DM 18,- öS 145,- sfr 18,- Fmk 80,- Lit 16 900,- hfl 23,90 Dr 2 200,-

DOS EXTRA Nr. 1 - 1993

DOS EXTRA

Sonderausgabe DOS EXTRA

Nr. 1 1993

ISSN 0934-2842
DMV-Verlag
Eine Publikation der
DOS INTERNATIONAL

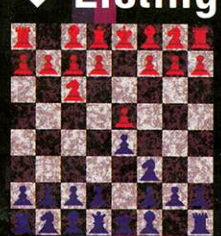
Top-Routinen zur Spieleprogrammierung

Rollenspiel im Eigenbau

- 3D-Grafik in Echtzeit
- Musik im Hintergrund
- Intelligenter Parser

Strategie-Geheimnisse gelüftet

- Die besten Algorithmen
- Listing: Profi-Schach





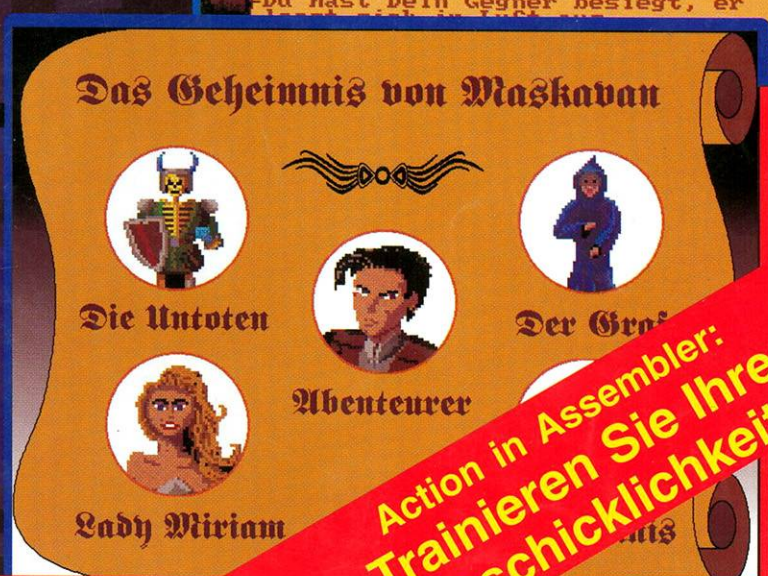
Bestellcoupon für alle Programme dieser Ausgabe im Heft











Action in Assembler: Trainieren Sie Ihre Geschicklichkeit

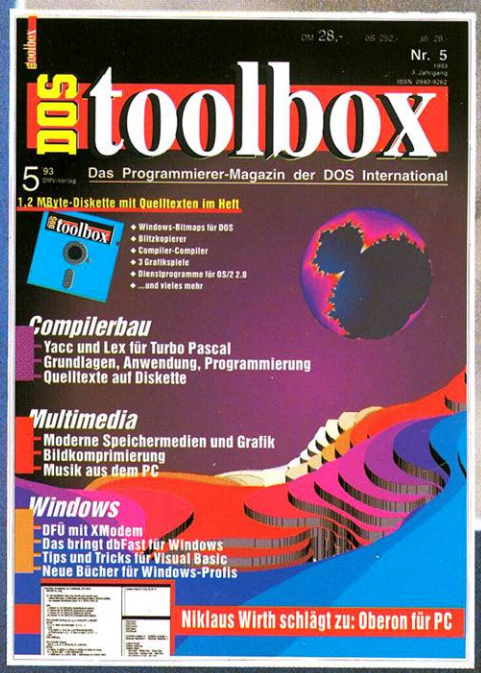
Aufgewacht!

Programmieren ist Zeit und Wissen

Sie haben die Idee. Wir die Informationen.

Der Schlüssel zum Compiler: Das vierteljährliche Programmierer-Magazin der DOS International bietet Ihnen qualifizierte und umfassende Informationen zu allen Programmierthemen.

Und weil wir wissen, daß Programmierer wenig Zeit haben, liefern wir sämtliche Programme und Quelltexte auf Diskette gleich mit. – Also anfordern.



Jetzt neu!

CeBIT'93 HANNOVER
24. — 31. 03. 1993
Halle 7, Stand E13/E19

DMV-Verlag · Postfach 250 · 3440 Eschwege

DMV Daten- und Medienverlag

Kann denn Spielen Sünde sein?

Der PC ist als »Spiele-Mühle« längst salonfähig geworden. Immer bessere Grafik- und Soundkarten machen es schwer, sich ein kleines Spielchen zu verkneifen. Man gönnt sich ja sonst nichts.

Vielleicht haben Sie sich schon einmal gefragt, wie Computerspiele gemacht werden. Sie denken, Scharen von Programmierern schlagen sich zahllose Nächte um die Ohren? Sie haben recht. Die Essenz dieser Mühen haben wir in dieses Heft gepackt. Es enthält den Stoff, der in endlosen Stunden, ja sogar Jahren ausgefeilter Programmierarbeit entstanden ist. Die Vorbereitungen für den Strategie-Schwerpunkt haben tatsächlich im Frühjahr 1992 begonnen.

Diese Ausgabe enthält Material, das Sie wahrscheinlich in keiner anderen Publikation finden, von dem aber viele Spielefirmen leben. Es war unser Bestreben, alle Tricks darzustellen und darüber hinaus umfassend zu dokumentieren. Wenn Sie sich mit den vorgestellten Programmen intensiv beschäftigen möchten, werden Sie sehr viel Zeit brauchen. Nicht zuletzt darin ist aber der Wert zu sehen: Sie erhalten eine Fülle an komprimiertem Know-how.

Das vergangene Jahr erlebte die Geburt einer neuen grafischen Darstellung von Spielewelten. Bis dato nicht gesehene, unglaublich schnelle 3D-Grafik war der Schlüssel zum Erfolg. Die Art der Grafikdarstellung fand jedoch sehr bald viele Freunde, so daß sich kurze Zeit später weitere, auch freundliche Spiele damit schmückten. Das Geheimnis dieses grafischen Leckerbissens lüftet unser Rollenspiel-Schwerpunkt. Dort zeigen wie Ihnen außerdem, wie Sie Musik im Hintergrund abspielen und wie Sie sich mit Ihrem Computer richtig unterhalten können. Ein deutscher Parser wird Ihnen verblüffende Antworten geben!

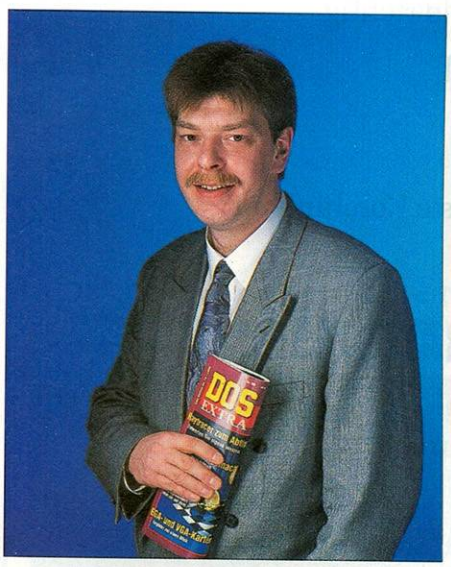
Können Maschinen denken? Dieser Frage gehen wir in unserem Strategie-Schwerpunkt auf den Grund. Wir zeigen Ihnen, wie Sie einfache Spiele immer gewinnen. Erprobte Strategien führen Sie weiter bis zum Spiel der Könige: Schach.

Viele andere, kleinere Projekte decken den Spielemarkt ziemlich vollständig ab. Die kleinen Programme zeigen insbesondere auch, daß sich mit wenig Aufwand interessante, pfiffige Spiele herstellen lassen.

Damit Ihnen beim Abtippen die Finger nicht zu kurz werden, können Sie mit dem in diesem Heft enthaltenen Gutschein alle Programme im Quelltext und als ausführbare Programme auf einer Databox zu einem günstigen Preis erwerben.

Ihr

Burkhard Müller



INHALT

DOS EXTRA
1/93



Ein komplettes Rollenspiel mit bewegter 3D-Grafik, Hintergrundmusik und Parser finden Sie ab Seite 6

ROLLENSPIELE

3D-Grafikprogrammierung 6
Ein 3D-Fenster zur Fantasiewelt

Textur-Generator 14
Ein Malprogramm für Maskavan

Simulation einer Welt 16
Das Geheimnis von Maskavan

Kommunikation und Parsing 48
Dialog mit dem PC

Soundprogrammierung 56
Musik aus dem Hintergrund

GESELLSCHAFTSSPIELE

Monopoly 58
Spekulieren Sie richtig

Das Suchspiel 73
Börsenfieber

Logikspiel 77
Wer ist der beste Kombinierer?

STRATEGIESPIELE

Das Nim-Spiel 86
Wer zuletzt nimmt

Tic Tac Toe 90
Der Tac-Tic auf der Spur

Reversi 95
Steinchen wechsle dich

Das Spiel der Könige 99
Schachmatt

ARCADE-SPIELE

Goldgräber 115
Sammeln Sie den Müll aus dem All

Tetris 134
Klötzchenweise

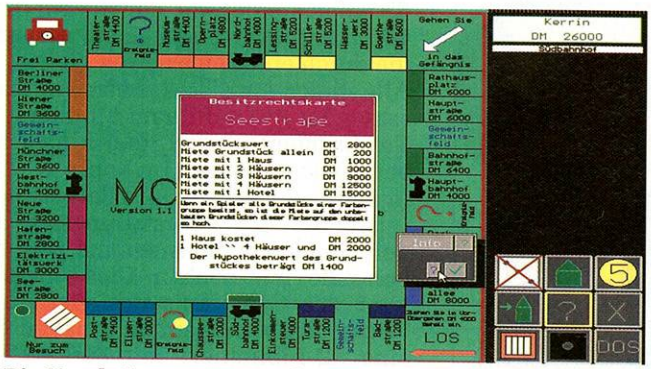
TIPS & TRICKS

Kleine Spiele 139

RUBRIKEN

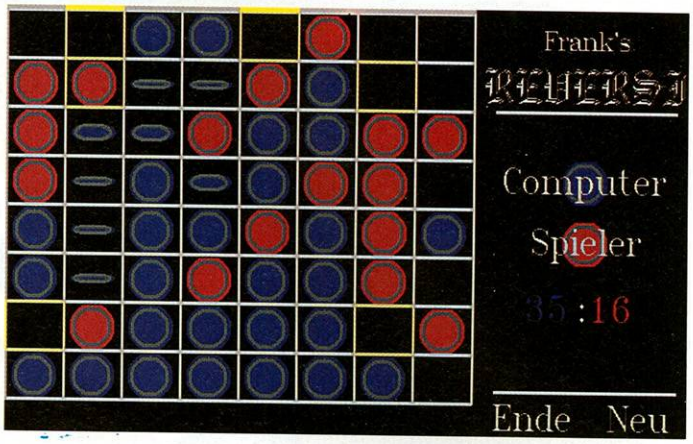
Editorial 3
Kann denn Spielen Sünde sein?

Impressum 148



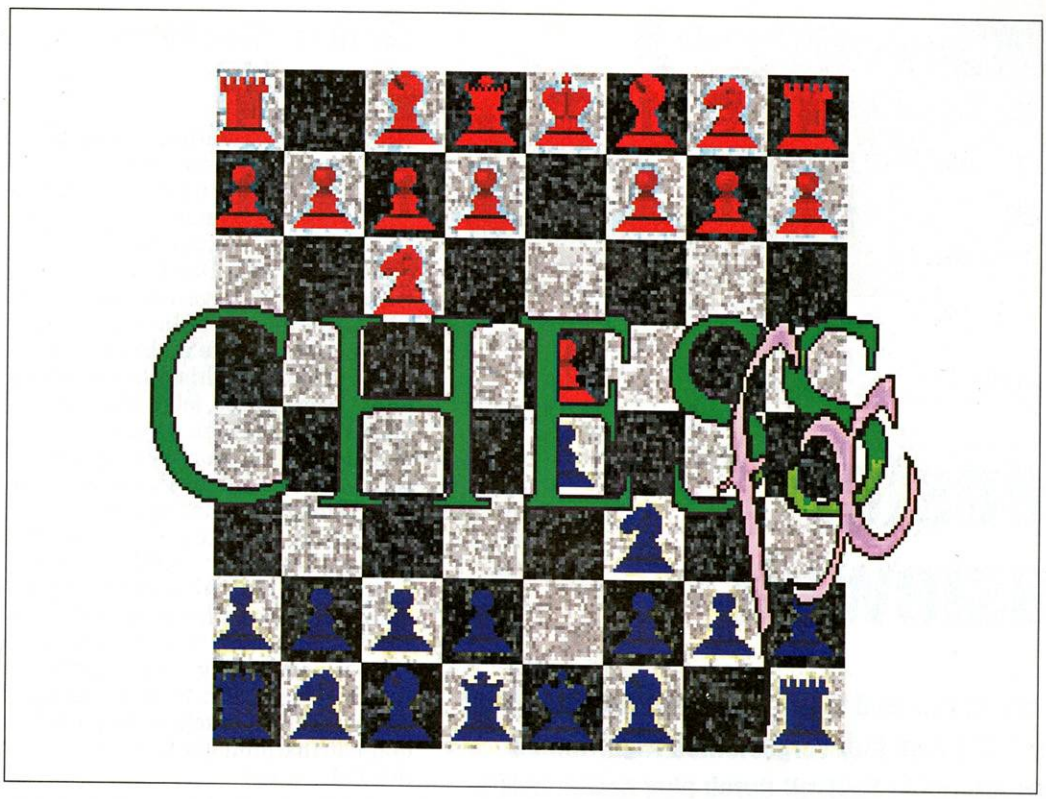
Die Simulation von Monopoly in Turbo Pascal wartet mit einer ansprechenden grafischen Spielführung auf und eignet sich auch für mehrere Mitspieler.

Seite 58



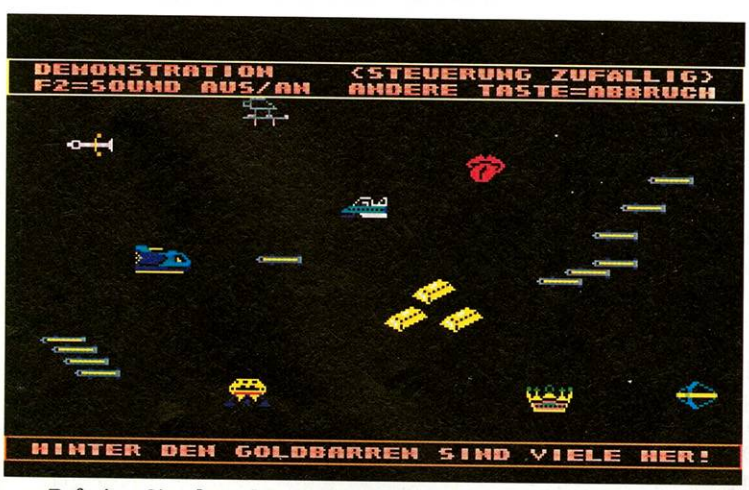
Reversi bereitet Ihnen mit einer Alpha-Beta-Strategie und einem Zuggenerator den Weg zur Programmierung komplexer Denkspiele.

Seite 95



Mit unserem schnellen Schachprogramm in Assembler bringen Sie Ihrem Computer Fähigkeiten bei, die dem richtigen Denken nahekommen.

Seite 99

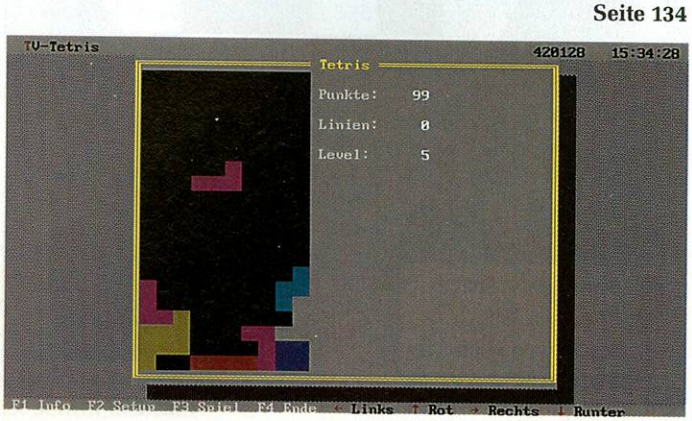


Befreien Sie als Goldgräber mit Treffsicherheit und Geschick das Weltall vom Zivilisationsmüll. Dieses schnelle Arcade-Game finden Sie ab

Seite 115

Den Dauerbrenner Tetris in einer Turbo-Pascal-Variante mit Zweispieler-Modus erhalten Sie ab

Seite 134





3D-Grafikprogrammierung

Ein 3D-Fenster zur Fantasiewelt

Sind Sie schon einmal vorbei an Bits und Bytes durch den Hauptspeicher Ihres Computers gewandert? Mit dem hier vorgestellten Algorithmus können Sie sich mausgesteuert und in Echtzeit durch eine dreidimensionale Welt bewegen, die Sie selbst erschaffen haben.

Stellen Sie sich vor, daß Sie mit Ihrer Maus eine Videokamera steuern, die durch eine Fantasiewelt gleitet. Alle Bilder, die diese Kamera einfängt, sehen Sie live auf Ihrem Monitor. Die Gefahr lauert überall, und hinten haben Sie keine Augen im Kopf.
Immer mehr Action- und Rollenspiele verwenden diese Darstellungsart, um den Spieler realitätsnah ins Geschehen miteinzubeziehen. Dieser Beitrag wird das Geheimnis des grafischen Tricks lüften. Außerdem erhalten Sie eine 3D-Bibliothek, mit deren Hilfe die Kreation eigener Welten zum Kinderspiel wird. Zunächst erfahren Sie allerdings mehr über den Hintergrund und die Idee dieser Darstellungsform.

Die Idee: Eine Dimension verschwindet

Die Ausgangssituation ist dieselbe wie bei früheren Action- und Rollenspielen auch: Sie haben eine Art Landkarte aus Bytes als Variablenfeld im Hauptspeicher Ihres Rechners deklariert. In dieser sind alle Objekte, wie Wände, Personen und Gegenstände, eingetragen. Während in üblichen Computerspielen diese Karte mehr oder weniger direkt auf den Bildschirm übertragen wird, werden Sie jetzt einen anderen Weg kennenlernen. Dieser geht zwar von denselben Voraussetzungen aus, sein Ziel ist jedoch die dreidimensionale Darstellung der Spielewelt.
Stellen Sie sich zunächst einmal vor, Sie befänden sich wirklich an einem bestimmten Ort auf dieser zweidimensionalen Landkarte im Hauptspeicher. Ein Speicherplatz mit dem Inhalt Null wäre durchsichtig, die übrigen undurchsichtig (Bild 1). Wie würden Sie dann die Objekte sehen, und nach welchen Prinzipien läuft dieser Vorgang des Sehens ab?
In der Natur ist das Licht verantwortlich dafür, daß Sie alle Gegenstände in

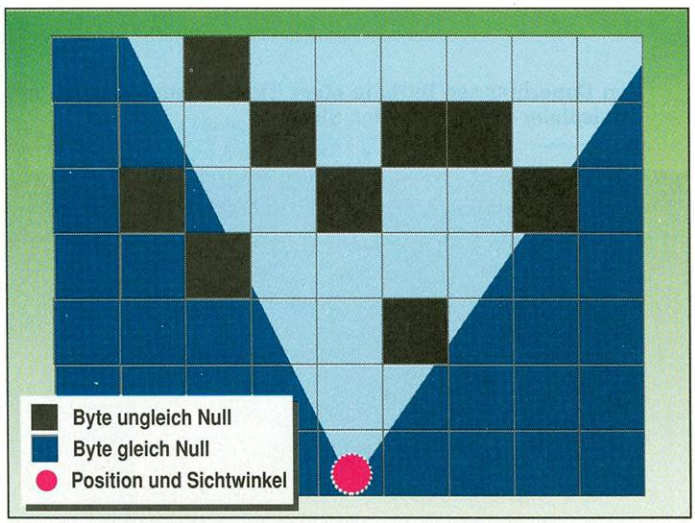


Bild 1. Die Landkarte der virtuellen Welt im Hauptspeicher

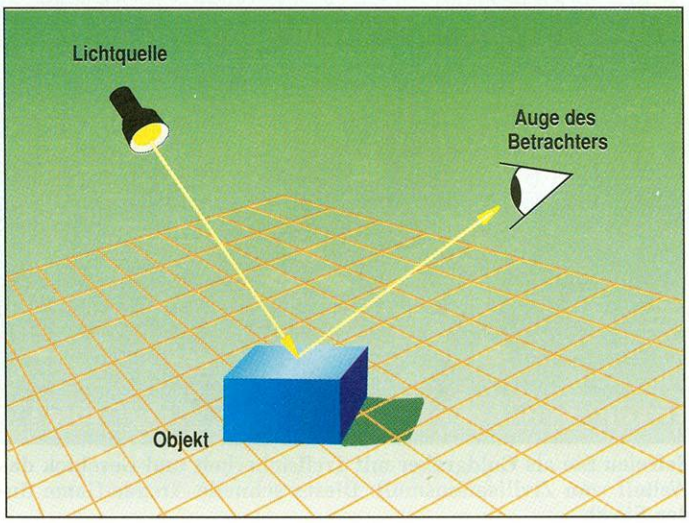


Bild 2. Der Weg eines Lichtstrahls

Der Pseudoalgorithmus für das Texture-Mapping

Name	Bedeutung
sb	Breite des Objektes
l1,l2	berechnete Anfangs- und Endhöhe des Objektes
t[l][b]	Array des Musters »t« mit Länge »l« und Breite »b«
b[s][z]	Bildschirm mit Spalte »s« und Zeile »z«
ht	Laufvariable: aktuell zu zeichnende Höhe
ax,ay	Startwert des Objektes auf dem Bildschirm

```
for x:=0 to sb
   begin
      ht = (l1 – (l1–l2)*x/sb)/2;
      for y:=0 to ht
         begin
            b[x+ax][ay+y]=t[b*x/sb][y*h/(ht*2)];
            b[x+ax][ay–y]=t[b*x/sb][h/2+y*h/(ht*2)]; end;
   end;
```

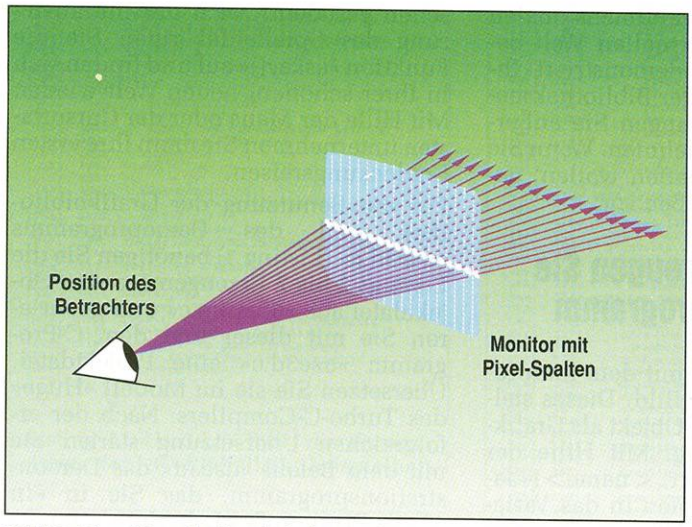


Bild 3. Eine Gerade durch jede Pixelspalte des Monitors

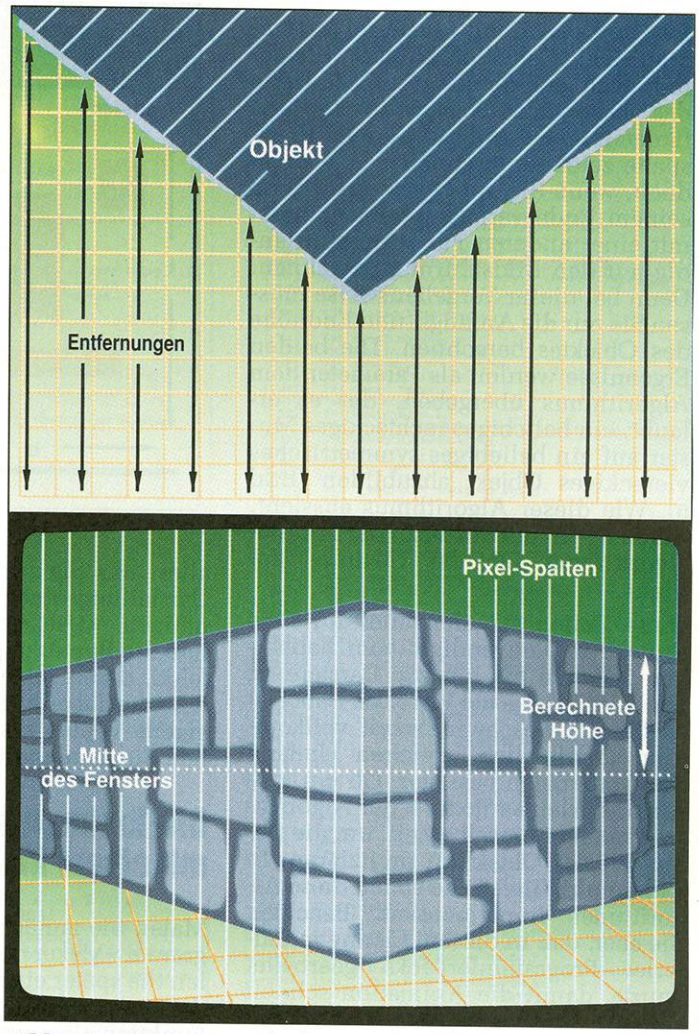


Bild 4. Der Weg zum nächstliegenden Objekt muß in jeder Pixelspalte gemessen werden. Damit erhalten Sie ein Maß für die Höhe des Objekts.

Ihrer Umgebung erkennen. Grundsätzlich passiert dabei folgendes: Aus einer Lichtquelle kommt ein Lichtstrahl, der auf ein Objekt stößt. Dort wird er reflektiert und trifft auf Ihr Auge (Bild 2). Wenn nun dieses Verhalten des Lichts mit dem Rechner nachgeahmt wird und die Lichtstrahlen mathematisch berechnet werden, entsteht ein dreidimensionales Bild, das dem natürlichen Bild entspricht.

Der Preis für die Qualität des Bildes: der Aufwand an Mathmatik und Berechnungszeit. Denn der kleine Nachteil dieses Verfahrens, das man Raytracing nennt, ist, daß die Berechnung eines Bildes oft mehrere Stunden Zeit in Anspruch nimmt. Eine Bewegung in Echtzeit ist natürlich ausgeschlossen. Abgesehen davon liegen die Objektdaten bei der Raytracing-Methode in drei Dimensionen vor, und unsere Landkarte hat nur zwei. Dennoch ist die mathematische Lichtstrahlenverfolgung die Lösung des Problems. Denn gerade weil die vorgegebene Landkarte nur in zwei Dimensionen vorliegt, ist es überhaupt nicht notwendig, dreidimensional alle Lichtstrahlen des Bildes zu berechnen. Das Fehlen der dritten Dimension bringt bei einer Berechnung nicht nur Vorteile, sondern vereinfacht sie zu einem Algorithmus, wie er zum Zeichnen von Linien benutzt wird.

Projektion eines Musters auf eine beliebige Fläche

Sie gehen dabei folgendermaßen vor: Sie wollen ein 3D-Fenster mit einer bestimmten Pixelbreite auf Ihren Bildschirm bringen. Wenn Sie mathematisch eine Gerade von Ihrer Ausgangsposition auf der Landkarte durch jedes Pixel in den zweidimensionalen Raum der Landkarte legen, so stößt diese Gerade unweigerlich früher oder später auf ein Objekt (Bild 3). Nun messen Sie die Entfernung zum gefundenen Objekt, das in dieser Blickrichtung liegt, und speichern sie zusammen mit der Objektnummer. Da in der Realität die Entfernung proportional zur Höhe ist, können Sie die Bildhöhe des gefundenen Objektes festlegen. Damit haben Sie das Ziel fast schon erreicht.

Wenn Sie nun einmal alle Pixelspalten durchgehen und jeweils die Bildhöhen der Objekte errechnen und darstellen, haben Sie Ihre virtuelle Welt schon vor Augen (Bild 4) – bisher natürlich noch einfarbig und ohne Muster.

Nachdem Sie nun die Höhe für jede Pixelspalte berechnet haben, verzieren

Die Rollenspiel-Komponenten

Dieser Beitrag leitet den Schwerpunkt »Rollenspiele« ein, in dem wir Ihnen die vollständige Programmierung eines aufwendigen Rollenspiels mit bewegter 3D-Grafik, Parsing und Sound zeigen. Er zeigt Ihnen die Programmierung der grafischen Grundlagen. Über das Parsing und den Sound gelangen Sie zum eigentlichen Spieleprogramm. Ein Malprogramm für eigene Spielewelten bildet den Abschluß dieser Serie. Alle darin vorgestellten Programme sind für sich allein lauffähig und ergeben zusammen das Rollenspiel »Das Geheimnis von Maskavan«.

Sie die Objekte mit Mustern (Texture-Mapping). Natürlich könnten Sie sofort nach der Berechnung der Höhe das Muster eines Objektes aus den Grafikdaten holen und die aktuelle Pixelspalte zeichnen. Schneller geht es jedoch, wenn Sie ein Objekt erst bis zu seinem Ende verfolgen, um es dann mit einer anderen Funktion vollständig auf den Bildschirm zu zeichnen. Denn bei dieser Vorgehensweise müssen Sie nur die Anfangs- und Endhöhe des Objektes berechnen. Die beiden Ergebnisse werden als Parameter dem Algorithmus übergeben, der es erlaubt, ein beliebiges rechteckiges Muster auf ein beliebiges symmetrisches viereckiges Objekt abzubilden (Bild 5). Wie dieser Algorithmus aussieht, sehen Sie in der Textbox »Der Pseudoalgorithmus für das Texture-Mapping«.

Diese Zeilen schaffen alles für eine Projektion Notwendige: vom beliebigen Zoomen bis hin zur Darstellung auch der schrägsten Vierecke. Dabei ist es für die Geschwindigkeit vollkommen unerheblich, welchen Umfang die Grafikdaten besitzen. Ob ein großes oder kleines Texture, der Algorithmus braucht dieselbe Zeit, um die Daten auf das Objekt zu projizieren. In »3danim.c« finden Sie genau diese Algorithmen implementiert. Beachten Sie aber, daß in diesem Listing mit allerlei Programmiertricks gearbeitet wurde, damit die Animation schnell abläuft.

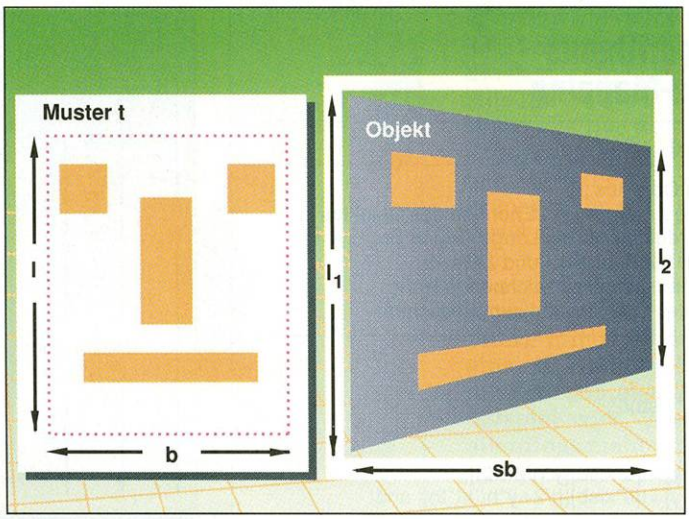


Bild 5. Ein rechteckiges Muster wird auf ein viereckiges, symmetrisches Objekt projiziert.

Die 3D-Bibliothek stellt Ihre Fantasiewelt dar

Auch ohne diese Grundlagen sind Sie in der Lage, Ihre Vorstellungen einer besseren Computerwelt zu realisieren. Denn in der 3D-Bibliothek finden Sie alles, was Sie zur dreidimensionalen Darstellung einer virtuellen Welt benötigen. »Use3d.c« demonstriert Ihnen die Benutzung der Bibliotheksbefehle, deren Bedeutungen Sie außerdem der Tabelle entnehmen. Wenn Sie eigene Welten erschaffen wollen, gehen Sie folgendermaßen vor:

Die Texturen erzeugen Sie mit einem Malprogramm

Malen Sie zunächst mit dem Malprogramm »Maker« ein Bild. Dieses stellen Sie später einem Objekt als Grafikdaten zur Verfügung. Mit Hilfe der Funktion »load(text *t, <name>)« laden Sie die Grafikdaten in das Variablenfeld desjenigen Objektes, das Sie verändern oder erzeugen möchten. (Weitere wichtige Hinweise über die Zusammenhänge der Strukturen finden Sie im Beitrag über das Rollenspiel »Das Geheimnis von Maskavan«.) Auch das Labyrinth des Spielfeldes können Sie nach Ihren eigenen Wünschen gestalten. Nach der Initialisierung des Spielfeldes rufen Sie die Funktion »askey()« auf und finden sich in Ihrer schönen, neuen Welt wieder. Mit Hilfe der Maus oder der Cursortasten unternehmen Sie dann Ihre ersten Entdeckungsreisen.

Für die Benutzung der Grafikbibliothek und des Demoprogramms »use3d.c« (Listing 1) benötigen Sie die Listings 2 bis 7. Erzeugen Sie eine Objektdatei aus »3danim.c« und generieren Sie mit dieser und dem C-Programm »use3d.c« eine Projektdatei. Übersetzen Sie sie im Modell »Huge« des Turbo-C-Compilers. Nach der erfolgreichen Übersetzung starten Sie mit dem Befehl »use3d« das Demonstrationsprogramm, das Sie in ein 16×16 Pixel großes Labyrinth mit passierbaren Wänden versetzt.

(Matthias Oberdorfer/bm)

Name	Bedeutung
showtext()	projiziert ein Muster auf ein symmetrisches Viereck (intern)
showpos(x,y,w)	findet nächstliegendes Objekt auf der Landkarte im Speicher (intern)
text	Struktur, die auf Grafikdaten eines Objektes zeigt (Näheres siehe Beitrag über Rollenspiel)
spielfeld	Struktur, die alle Daten einer virtuellen Welt verwaltet (Näheres siehe Beitrag über Rollenspiel)
setwinkel(w,sw)	initialisiert die internen Winkeltabellen je nach Breite und Größe des Sichtwinkels (intern)
showwelt(x,y,w, size,pf)	Darstellung eines 3D-Bildes ab Position »x,y« mit Blickwinkel »w« im Spielfeld »pf« mit der Breite »size«
BGVISIBLE	Farbwert für durchsichtige Teile von Wänden; außer dem Farbwert muß auch das Objekt als »OBJECT« attributiert sein (Näheres dazu siehe Beitrag über Rollenspiel).
TRANSPARENT, WALL,OBJECT	Attribute von Objekten der Landkarte (Näheres dazu siehe Beitrag über Rollenspiel)
initpf(pf,t,l,width, sw,ysize)	initialisiert die Darstellung des Fensters und des Spielfeldes. Das Fenster beginnt ab Position »t,l« mit der Breite »width« und der Höhe »ysize«. Das Spielfeld »pf« wird mit dem Sichtwinkel »sw« dargestellt.
askkey(*x,*y,*w, *pf,width)	gestattet über Maus- und Tastenabfrage, durch die dreidimensionale Welt zu gehen

Diese Befehle stellt Ihnen die 3D-Bibliothek zur Verfügung.

```
[W6Ø]    1: /*
[3V2]    2:  Programm: use3d.c
[EF6]    3:  Funktion: Demonstriert 3D-Darstellung
[DH4]    4:            mit Hilfe der 3D-Bibliothek
[K83]    5:  Sprache:  Turbo C 2.Ø
[KR4]    6:  Autor:    Matthias Oberdorfer
[8Z4]    7:  (c) 1992 DMV Widuch GmbH & Co. KG */
[ØØØ]    8:
[4L2]    9: #include"defs.h"
[6M4]   1Ø: static int feld[16][16] = {
[ØH5]   11: Øx4ØØ4,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,
[RG5]   12: Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,
[UN3]   13: Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,
[7Z4]   14: Øx4ØØ4,Ø,Ø,Ø,1,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Øx4ØØ4,
[4Z4]   15: Øx4ØØ4,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Øx4ØØ4,
[7Z4]   16: Øx4ØØ4,Ø,Ø,Ø,1,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Øx4ØØ4,
[CZ4]   17: Øx4ØØ4,1,1,1,1,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Øx4ØØ4,
[7Z4]   18: Øx4ØØ4,Ø,Ø,Ø,1,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Øx4ØØ4,
[7Z4]   19: Øx4ØØ4,Ø,Ø,Ø,1,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Øx4ØØ4,
[7Z4]   2Ø: Øx4ØØ4,Ø,Ø,Ø,1,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Øx4ØØ4,
[4Z4]   21: Øx4ØØ4,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Øx4ØØ4,
[NZ4]   22: Øx4ØØ4,Ø,Ø,Ø,1,Ø,1,1,1,1,1,Ø,1,Ø,Ø,Ø,Øx4ØØ4,
[HZ4]   23: Øx4ØØ4,Ø,1,1,1,Ø,Ø,Ø,Ø,Ø,1,Ø,1,Ø,Ø,Ø,Øx4ØØ4,
[TZ4]   24: Øx4ØØ4,Ø,1,1,1,1,1,Ø,1,Ø,1,Ø,1,1,Ø,Ø,Øx4ØØ4,
[7Z4]   25: Øx4ØØ4,Ø,Ø,Ø,Ø,Ø,Ø,Ø,1,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Øx4ØØ4,
[TZ4]   26: Øx4ØØ4,Ø,1,1,1,1,1,Ø,1,Ø,1,Ø,1,1,Ø,Ø,Øx4ØØ4,
[7Z4]   27: Øx4ØØ4,Ø,Ø,Ø,Ø,Ø,Ø,Ø,1,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Øx4ØØ4,
[4K5]   28: Øx4ØØ4,Øx4ØØ4,Øx4ØØ4,Øx4ØØ4,Øx4ØØ4,Øx4ØØ4,Øx4ØØ4,
[4K5]   29: Øx4ØØ4,Øx4ØØ4,Øx4ØØ4,Øx4ØØ4,Øx4ØØ4,Øx4ØØ4,Øx4ØØ4,
[344]   3Ø: Øx4ØØ4,Øx4ØØ4,Øx4ØØ4,Øx4ØØ4,};
[3K6]   31: static unsigned char texture[16][16] = {
[G13]   32: 79,79,79,79,79,79,79,79,
[G13]   33: 79,79,79,79,79,79,79,79,
[YS4]   34: 79,156,156,156,156,156,156,156,156,156,
[2Z2]   35: 156,156,156,156,156,79,
[VR2]   36: 79,156,79,79,79,79,79,
[5H3]   37: 79,79,79,79,79,79,79,79,156,79,
[VR2]   38: 79,156,79,79,79,79,79,
[5H3]   39: 79,79,79,79,79,79,79,79,156,79,
[VR2]   4Ø: 79,156,79,79,79,79,79,
[5H3]   41: 79,79,79,79,79,79,79,79,156,79,
[833]   42: 79,156,79,79,79,79,79,79,
[433]   43: 79,79,79,79,79,79,156,79,
[VR2]   44: 79,156,79,79,79,79,79,
[5H3]   45: 79,79,79,79,79,79,79,79,156,79,
[VR2]   46: 79,156,79,79,79,79,79,
[5H3]   47: 79,79,79,79,79,79,79,79,156,79,
[833]   48: 79,156,79,79,79,79,79,79,
[433]   49: 79,79,79,79,79,79,156,79,
[833]   5Ø: 79,156,79,79,79,79,79,79,
```

```
[433]   51: 79,79,79,79,79,79,156,79,
[833]   52: 79,156,79,79,79,79,79,79,
[433]   53: 79,79,79,79,79,156,79,
[833]   54: 79,156,79,79,79,79,79,79,
[433]   55: 79,79,79,79,79,79,156,79,
[833]   56: 79,156,79,79,79,79,79,79,
[433]   57: 79,79,79,79,79,79,156,79,
[833]   58: 79,156,79,79,79,79,79,79,
[433]   59: 79,79,79,79,79,79,156,79,
[AV2]   6Ø: 79,156,156,156,156,156,156,
[Y@3]   61: 156,156,156,156,156,156,156,
[SY1]   62: 156,156,156,79,
[G13]   63: 79,79,79,79,79,79,79,79,
[VE3]   64: 79,79,79,79,79,79,79,79,);
[@@@]   65:
[8L4]   66: unsigned char t[16][16] ={
[GN3]   67: 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
[G44]   68: 1,21,21,21,21,21,21,21,21,21,21,21,
[LA1]   69: 21,21,21,1,
[BV3]   7Ø: 1,21,1,1,1,1,1,1,1,1,1,1,1,1,21,1,
[BV3]   71: 1,21,1,1,1,1,1,1,1,1,1,1,1,1,21,1,
[BV3]   72: 1,21,1,1,1,1,1,1,1,1,1,1,1,1,21,1,
[BV3]   73: 1,21,1,1,1,1,1,1,1,1,1,1,1,1,21,1,
[BV3]   74: 1,21,1,1,1,1,1,1,1,1,1,1,1,1,21,1,
[BV3]   75: 1,21,1,1,1,1,1,1,1,1,1,1,1,1,21,1,
[BV3]   76: 1,21,1,1,1,1,1,1,1,1,1,1,1,1,21,1,
[BV3]   77: 1,21,1,1,1,1,1,1,1,1,1,1,1,1,21,1,
[BV3]   78: 1,21,1,1,1,1,1,1,1,1,1,1,1,1,21,1,
[BV3]   79: 1,21,1,1,1,1,1,1,1,1,1,1,1,1,21,1,
[BV3]   8Ø: 1,21,1,1,1,1,1,1,1,1,1,1,1,1,21,1,
[BV3]   81: 1,21,1,1,1,1,1,1,1,1,1,1,1,1,21,1,
[G44]   82: 1,21,21,21,21,21,21,21,21,21,21,21,
[LA1]   83: 21,21,21,1,
[P74]   84: 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,} ;
[LV6]   85: text te= { 4,&texture[Ø][Ø],4,&t[Ø][Ø],);
[Ø35]   86: thing Wand = {"", Ø,Ø,Ø,Ø,Ø,Ø,&te,Ø,Ø};
[8L6]   87: thing *allthings[]= { &Wand, &Wand,&Wand,
[Ø53]   88: &Wand,&Wand, &Wand,};
[LD3]   89: spielfeld spfld = {
[V55]   9Ø: &feld[Ø][Ø],4,11Ø,211,allthings};
[W35]   91:  adventurer ich =("hallo",Øx@2ØØ,
[DZ3]   92:  Øx@2ØØ,Ø,-1,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,
[633]   93:  Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,DEFWACH/2
[HG5]   94:  ,DEPHUNG/2,LE/3*2,TAG/4,Ø,MAENNLICH,
[P62]   95:  KRAFTANF,Ø);
[Ø82]   96: showarm() {}
[UWØ]   97: main()
[P7Ø]   98: {
[VE3]   99:  /* Bildschirmbreite */
[KS2]   1ØØ:  unsigned width;
[YB5]   1Ø1:  /* Winkel des Sichtbereichs 45 - 9Ø
[C53]   1Ø2:     sichtbarer Winkel */
[962]   1Ø3:  unsigned sw;
[6R1]   1Ø4:  width = 22Ø;
[GF1]   1Ø5:  sw    = 35;
[X13]   1Ø6:  if(!setgraphik(ON))
[CV5]   1Ø7:   {printf("VGA - Karte notwendig!\n");
[ØM1]   1Ø8:    exit(Ø); }
[SG3]   1Ø9:  setwinkel(width,sw);
[F45]   11Ø:  initpf(&spfld,5Ø,2Ø,width,1ØØ,sw);
[C22]   111:  animcount=Ø;
[8L2]   112:  spfld.ich = &ich;
[GV1]   113:  gotoxy(1,1);
[WD1]   114:  cls(13,Ø);
[LZ1]   115:  gotoxy(1,2Ø);
[536]   116:  print(" Demonstration von Echtzeit-"
[KU3]   117:  "3D-Fenstern", 13<<8|124);
[FT4]   118:  askkey(&(ich.posx),&(ich.posy)
[B74]   119:        ,&(ich.look),&spfld,width);
[UM2]   12Ø:  setgraphik(OFF);
[U7Ø]   121:  }
```

Listing 1. »Use3d.c« führt die Verwendung der 3D-Bibliothek beispielhaft vor.

```
[W6Ø]    1: /*
[PD3]    2:  Programm: schalter.c
[UU5]    3:  Funktion: Mausschalterfunktionen
[K83]    4:  Sprache:  Turbo C 2.Ø
[KR4]    5:  Autor:    Matthias Oberdorfer
[8Z4]    6:  (c) 1992 DMV Widuch GmbH & Co. KG */
[@ØØ]    7:
[ØR2]    8: #ifndef SCHALTER
[6R2]    9: #define SCHALTER
[SR2]   1Ø: #include"mouse.c"
[6R2]   11: #include"graph.h"
[7M2]   12: #include<dos.h>
[Ø43]   13: #include<stdlib.h>
[8W2]   14: #include<stdio.h>
[864]   15: typedef struct schalter
[FX6]   16:  {struct schalter *next;/*next Schalter*/
[WH5]   17:   unsigned left,top; /* wie groß */
[EN5]   18:   unsigned right,bottom; /* und wo */
[CY4]   19:   unsigned framecol,col,ilucol;
[KB3]   2Ø:   unsigned framecol2;
[GØ3]   21:   unsigned textcol;
[MF6]   22:   unsigned backcol;/*bleibt erleuchtet 2*/
[BY5]   23:  /* blinkt bei 1, tut sich nichts bei Ø*/
[KG3]   24:   char *msg; } schalter;
[RX6]   25: unsigned testsch(unsigned x,unsigned y,
[CV4]   26:   schalter *sch,schalter **sha);
[LR5]   27: void print(char *str,unsigned co);
[NN4]   28: void printnum(unsigned nr);
[DX3]   29: void getstr(char *str);
[SL4]   3Ø: void showsch(schalter *sch);
[GM6]   31: /* testsch((x,y,sch,sha): überprüft ob
[G86]   32: an Stelle x,y ein Schalter ist und gibt
[WN5]   33: ihn gegebenenfalls in sha zurück.
[ZH4]   34: sch ist erster Schalter; */
[VW4]   35: unsigned testsch(x,y,sch,sha)
[Ø62]   36: unsigned x,y;
[U53]   37: schalter *sch,**sha;
[UV2]   38:  {schalter *s=sch;
[M91]   39:   int i=1;
[DB1]   4Ø:   while(s)
[LE5]   41:    {if(x>=s->left-1 && x<=s->right+1 &&
[LK4]   42:        y>=s->top-1 && y<=s->bottom+1)
[SK2]   43:       {if(s->backcol)
[4K2]   44:         {mouse_hide();
[ØC5]   45:          drawbox(s->left,s->top,s->right
[8W3]   46:                 ,s->bottom, s->ilucol,Ø);
[WH2]   47:          if(s->backcol==1)
[P7Ø]   48:            {
[XF1]   49:          delay(6Ø);
[CD5]   5Ø:          drawbox(s->left,s->top,s->right,
[M14]   51:          s->bottom, s->framecol,Ø);
[WG4]   52:          drawbox(s->left-1,s->top-1,
[FL3]   53:          s->right+1,s->bottom+1,
[1F2]   54:          s->framecol,Ø);
[HC4]   55:          drawline(s->left-1,s->top-1,
[B95]   56:          s->right+1,s->top-1,s->framecol2);
[5B4]   57:          drawline(s->left-1,s->top-1
[7C3]   58:          ,s->left-1,s->bottom+1
[TE2]   59:          , s->framecol2);
```

```
[DB5]   6Ø:          drawline(s->left,s->top,s->right
[8R3]   61:          ,s->top, s->framecol2);
[4B5]   62:          drawline(s->left,s->top,s->left,
[F94]   63:          s->bottom, s->framecol2);}
[2P2]   64:          mouse_show();}
[D81]   65:        *sha=s;
[642]   66:        return(i);}
[UFØ]   67:     i++;
[N72]   68:     s=s->next;}
[D32]   69:   return(Ø);}
[K44]   7Ø: static char far *firstch;
[YG2]   71: static int i,j,c;
[RY5]   72: printchar(int ch,int col,int x,int y)
[3M2]   73:  {x=x<<3;y=y<<3;
[7Y5]   74:   firstch=*((char far **)(Øx@43L<<2))+8*ch;
[AF2]   75:   for(i=Ø; i<8;i++)
[VL2]   76:    {c=*(firstch+i);
[LD3]   77:     for(j=Ø;j<8;j++,c=c<<1)
[FL6]   78:     setp(x+j,y+i,c&Øx8Ø?col&Øxff:col>>8);}}
[WY4]   79: /* print(str, co):gibt String im
[WF5]   8Ø: Grafikmodus in der Farbe co aus. */
[RZ2]   81: void print(str,co)
[7L1]   82: char *str;
[T32]   83: unsigned co;
[KG1]   84:  {int i=Ø;
[7H2]   85:   while(*(str + i))
[PE5]   86:    {printchar(*(str+i),co,wherex()-1,
[PS1]   87:     wherey()-1);
[GE4]   88:     gotoxy(wherex()+1,wherey());
[B31]   89:     i++;}}
[9T6]   9Ø: /* getstr(str): holt String vom Benutzer
[YZ2]   91: im Graphikmodus */
[3X2]   92: void getstr(str)
[7L1]   93: char *str;
[L32]   94:  {char ch[2];
[7B1]   95:   ch[1]=Ø;
[1VØ]   96:   do{
[8C2]   97:     ch[Ø]=getch();
[M32]   98:     switch(ch[Ø])
[TR1]   99:      {case '\r':
[9K1]  1ØØ:       case '\n':
[PØ1]  1Ø1:        break;
[A51]  1Ø2:       case 8:
[211]  1Ø3:        str--;
[KE4]  1Ø4:        gotoxy(wherex()-1,wherey());
[232]  1Ø5:        print(" ",215);
[KE4]  1Ø6:        gotoxy(wherex()-1,wherey());
[PØ1]  1Ø7:        break;
[8E1]  1Ø8:       default:
[632]  1Ø9:        print(ch,215);
[ZC2]  11Ø:        *str ++ = ch[Ø];
[7M1]  111:        break;}}
[D43]  112:   while(ch[Ø]!='\r');
[3M1]  113:   *str = Ø;}
[VM6]  114: /* printnum(i):gibt Zahl im Grafikmodus
[N31]  115: aus. */
[6U2]  116: void printnum(i)
[@Ø2]  117: unsigned i;
[SE2]  118:  {char str[1Ø];
[ZB2]  119:   itoa(i,str,1Ø);
[RH2]  12Ø:   print(str,215);}
[MB6]  121: /* showsch(sch) : Stellt Schalter auf dem
[RN6]  122: Bildschirm nach structdefinitonen auf */
[LW2]  123: void showsch(sch)
[SA2]  124: schalter *sch;
[UV2]  125:  {schalter *s=sch;
[DB1]  126:   while(s)
[2X1]  127:    {if(s->msg)
[CD5]  128:      drawbox(s->left,s->top,s->right,
[Z33]  129:      s->bottom, s->col,1);
[CD5]  13Ø:     drawbox(s->left,s->top,s->right,
[M14]  131:     s->bottom, s->framecol,Ø);
[V66]  132:     drawbox(s->left-1,s->top-1,s->right+1,
[BA4]  133:     s->bottom+1, s->framecol,Ø);
[CY5]  134:     drawline(s->left-1,s->top-1,s->right+1
[5T3]  135:     ,s->top-1, s->framecol2);
[XT5]  136:     drawline(s->left-1,s->top-1,s->left-1
[LC4]  137:     ,s->bottom+1, s->framecol2);
[5E5]  138:     drawline(s->left,s->top,s->right,
[UF3]  139:     s->top, s->framecol2);
[4B5]  14Ø:     drawline(s->left,s->top,s->left,
[SX3]  141:     s->bottom, s->framecol2);
[VF1]  142:     if(s->msg)
[445]  143:      {gotoxy(s->left/8+2,s->top/8+2);
[SG6]  144:      print(s->msg,s->textcol|(s->col<<8));}
[KB2]  145:    s=s->next;})
[NZØ]  146: #endif
```

Listing 2. »Schalter.c« enthält die notwendigen Funktionen für die Bedienung mit der Maus.

```
[W6Ø]    1: /*
[7Z2]    2:  Programm: 3danim.c
[Z46]    3:  Funktion: Darstellung von 3D-Welten
[K83]    4:  Sprache:  Turbo C 2.Ø
[KR4]    5:  Autor:    Matthias Oberdorfer
[HN4]    6:  (c) 1992 DMV Widuch GmbH & Co. KG
[17Ø]    7: */
[RH1]    8: #ifndef D3
[XH1]    9: #define D3
[7M2]   1Ø: #include<dos.h>
[RN2]   11: #include<math.h>
[2T2]   12: #include<alloc.h>
[6R2]   13: #include"graph.h"
[SR2]   14: #include"mouse.c"
[GB3]   15: #include"schalter.c"
[CV2]   16: extern showarm();
[AX5]   17: /* showtext(x,l1,l2,sb,*text,anf,end)
[5H6]   18:  stellt ein Texture auf dem Bildschirm dar
[TC5]   19:  x      : ist der rechte Anfangswert
[8U2]   2Ø:           der Darstellung
[9D5]   21:  l1,l2 : Anfangs- und End-Höhe der
[942]   22:           Darstellung
[6M4]   23:  sb     : Breite der Darstellung
[9F5]   24:  *text : Zeiger auf das Texturebild
[PT4]   25:  anf,end: relative Anfangs und
[114]   26:           Endordinaten des Bildes
[WG5]   27:  Globale Variablen von Bedeutung:
[8V4]   28:  left,top: Linke obere Ecke des
[EW3]   29:            virtuellen Bildschirms
[8L4]   3Ø:  YSIZE    : Höhe des virtuellen
[242]   31:            Bildschirms
[385]   32:  floorcol: Farbe des Fussbodens
[CF2]   33:            Farbe der Decke
[ST5]   34:  vid      :Startadresse des Bildschirm
[S33]   35:            initialisieren mit
[DV1]   36:            initvid()  */
[KU2]   37: #ifndef SHOWTEXT
[SU2]   38: #define SHOWTEXT
[U@3]   39: #define XCOUNT  32Ø
[MW2]   4Ø: #define YSIZE   114
[Ø52]   41: #ifndef SHOWPOS
[V43]   42: #define YSIDE          1
[R43]   43: #define XSIDE          Ø
[NZØ]   44: #endif
[PX8]   45: #define initvid() vid=(char far *)(Øxa@@@@@
            Ø@+top*32Ø+left)
```

```
[MU3]   46: typedef struct text {
[K15]   47: int xdim; unsigned char *xmap;
[N96]   48: int ydim; unsigned char *ymap; } text;
[WX2]   49: #include"defadv.h"
[5L6]   5Ø: void showtext(int i,int l1,int l2,int sb,
[YB5]   51:   text *tex,int txbegin,int txend,
[5P1]   52:   int side);
[2A6]   53: /* links oben des virtuellen Schirms */
[B23]   54: static int left=2Ø;
[EU2]   55: static int top=1Ø;
[624]   56: static int ysize=YSIZE;
[DB3]   57: static int floorcol;
[Z53]   58: static int ceilcol;
[TV4]   59: static unsigned char far *vid;
[UN5]   6Ø: static unsigned char far *virtvid;
[U35]   61: static unsigned char far *vidb;
[HG6]   62: static unsigned char far *vide,far *vidu;
[K66]   63: static unsigned int xcount = XCOUNT;
[Y@7]   64: static unsigned int hy = XCOUNT*(YSIZE/2);
[NL2]   65: static long tt;
[272]   66: static int st;
[ZL2]   67: static long sx;
[C73]   68: static char *savet;
[7M2]   69: #include<dos.h>
[413]   7Ø: static unsigned w;
[S91]   71: long x;
[452]   72: int iht,dim;
[7N1]   73: char   *map;
[BD2]   74: int start,end;
[V52]   75: unsigned v,z;
[7Ø6]   76: static void showtext(i,l1,l2,sb,tex,
[SB3]   77:  txbegin,txend,side)
[F53]   78:  int i,l1,l2,sb,side;
[183]   79:  int txbegin,txend;
[ZP1]   8Ø: text *tex;
[F74]   81:  {register unsigned int j;
[YN2]   82:  register int y;
[KB1]   83:  int ix;
[8L2]   84:  if(side ==XSIDE)
[5R2]   85:   {dim = tex->xdim;
[724]   86:    map = (char  *)tex->xmap;}
[ZRØ]   87:  else
[7R2]   88:   {dim = tex->ydim;
[WØ4]   89:    map =(char  *) tex->ymap;}
[ZD4]   9Ø:  txbegin = txbegin>>(8-dim);
[LP3]   91:  txend = txend>>(8-dim);
[3D1]   92:  st=l1-l2;
[581]   93:  if(sb<=Ø)
[KC1]   94:   return;
[YK5]   95:  sx=(long)((txend-txbegin+1)<<6)/sb;
[XD3]   96:  v = (unsigned long) i;
[V83]   97:  z = v+hy+hy-xcount;
[FK6]   98:  for(x=txbegin<<6,tt=Ø,j=Ø;j<sb;j++,x+=sx,
[H42]   99:           tt+=st,v+=,z+=)
[LY1]  1ØØ:   {ix = x>>6;
[5U3]  1Ø1:    iht = l1-tt/sb&Øxffffe;
[D72]  1Ø2:    savet=map+ix;
[A52]  1Ø3:    if(iht>ysize)
[VXØ]  1Ø4:     {w=v;
[DS3]  1Ø5:      start = (iht-ysize)>>1;
[TG3]  1Ø6:      end   = start+ysize;}
[ZRØ]  1Ø7:     else
[8Z4]  1Ø8:      {w=v+xcount*((ysize-iht)>>1);
[KA1]  1Ø9:       start=Ø;
[GN1]  11Ø:       end=iht;}
[XM5]  111:     for(y=start;y<end;y ++,w +=xcount)
[414]  112:     *(vid+w)=*(savet+(((y<<dim )
[ML2]  113:                       /iht)<<dim)); })
[NZØ]  114: #endif
[ØS2]  115: #ifndef SHOWPOS
[4M4]  116: /* unsigned showpos(xØ,yØ,w):
[345]  117: ermittelt folgende Werte einer
[C54]  118: dreidimensionalen Welt:
[P86]  119: - Abstand des Beobachters zum nächsten
[EØ4]  12Ø:   Objekt in Blickrichtung:
[S14]  121:   globale Varibale laenge
[UL2]  122: - Art des Objekts
[HW5]  123:   (playfield + Rückgabe der Funktion)
[2D3]  124: - Nummer des Objekts:
[ØC4]  125:   Rückgabewert des Objekts
[FY4]  126: - Auffall des Blickes auf X oder
[YF3]  127:   Y-Seite des Objekts
[HG6]  128:   in globaler Varialer side: YSIDE, XSIDE
[937]  129: - X-Koordinate auf einer Objektseite lx,ly
[LP2]  13Ø: Parameter sind:
[RH6]  131:  xØ,yØ: Position in spezieller Codierung
[HD6]  132:  w    : Blickwinkel als Zeiger auf Tabellen
[EP6]  133: Globale Variablen auf die Bezug genommen
[9WØ]  134: wird:
[MD6]  135:  wsin[],wcos[] : Tabelle mit Sinus und
[TM4]  136:    Cosinus Werten. Wird durch
[SD4]  137:    setwinkel() initialisiert.
[PG6]  138:  *playfield: Zeiger auf ein Spielfeld mit
[UT5]  139:    Objekten (Größe maximal 256 * 256)
[5X6]  14Ø:  playy: y-Größe als Exponent zur Basis 2
[NW4]  141:    Bsp: Spielfeld y=16 playy=4; */
[CF2]  142: #define SHIFT 8
[F63]  143: #define WIDTHMAX 96Ø
[893]  144: #define PI 3.1415927464
[ØS2]  145: #ifndef SHOWPOS
[XE2]  146: #define YSIDE 1
[RE2]  147: #define XSIDE Ø
[NZØ]  148: #endif
[5D6]  149: #define BGVISIBLE Øxff /* für Textures*/
[EC5]  15Ø: const unsigned UNVISIBLE = Øx2Ø@@;
[Y94]  151: const unsigned AIR = Øx4@@@;
[MC5]  152: const unsigned INVISIBLE = Øx8@@@;
[W25]  153: unsigned showpos(unsigned xØ,
[H93]  154:   unsigned yØ,int w);
[MG3]  155: typedef long wsize;
[685]  156: static wsize wsin[4*WIDTHMAX];
[P65]  157: static wsize wcos[4*WIDTHMAX];
[1K3]  158: static int *playfield;
[T@5]  159: static int playy=4; /* 2hoch4=16*/
[M64]  16Ø: /* Vorzeichen-Variablen */
[7R2]  161: static int vx,vy;
[VY2]  162: static long ix,iy;
[2L3]  163: static wsize wsi,wco;
[4Z2]  164: static long lx,ly;
[2N2]  165: static int  side;
[CL3]  166: static long laenge=@L;
[S64]  167: static long stepx,stepy;
[9S6]  168: /* enthält Anzahl der gebrauchten Winkel*/
[8M1]  169: int numw;
[393]  17Ø: static unsigned pf;
[D54]  171: unsigned showpos(xØ,yØ,w)
[434]  172: register unsigned xØ,yØ;
[B21]  173: int w;
[395]  174:  { /* ax,ay sind die Abstände der
[S86]  175:     Schnittpunkte zu den Startpunkten */
[FC3]  176:  register long ay,ax;
[487]  177: /* px,py sind die Schnittpunkte der Geraden
[FP3]  178:    mit den x,y-Geraden */
[UE3]  179:  register int px,py;
[AB2]  18Ø:  ix=iy=Øx1@@@@L;
[GS1]  181:  vx = vy = 1;
[BA6]  182:  /* Einmalige Zuweisung ist schneller */
[WP4]  183:  wsi=wsin[w],wco=wcos[w];
[ØA1]  184:  if(wco<Ø)
[A93]  185:   {ay  = -(xØ & Øx@Øfffl);
[H63]  186:    iy  = -iy;vy  = -vy;}
```

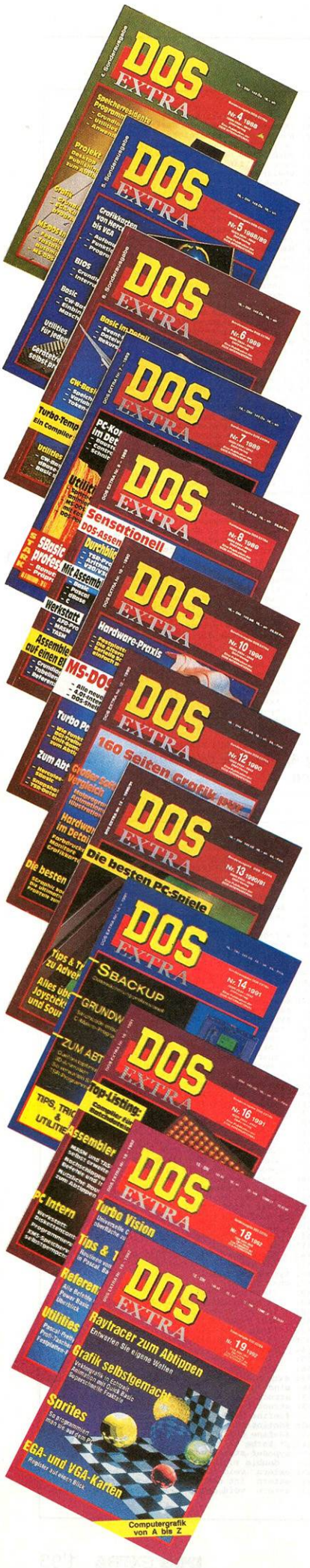
Listing 3. Die Bibliothek »3danim.c« enthält alle Funktionen zur Darstellung dreidimensionaler Welten.

```
[ZRØ]  187:   else
[6U3]  188:    ay = Øx91ØØL - (xØ & Øx ØØff);
[EA1]  189:   if(wsi<Ø)
[Ø93]  19Ø:    {ax  = - (yØ & ØxØØffL);
[T33]  191:     ix  = -ix; vx  = -vx;}
[ZRØ]  192:   else
[5U3]  193:    ax = Øx91ØØL - (yØ & ØxØØff);
[735]  194:   px =  xØ>>SHIFT; py =  yØ>>SHIFT;
[1N5]  195:   /* kann X-Gerade erreicht werden ? */
[T81]  196:   if(!wsi)
[EG4]  197:     {       /* nein parallel zu X */
[BE1]  198:      lx=ly=ØL;
[S62]  199:      side  = YSIDE;
[GF4]  2ØØ:      px += vy;      /* Erst eins vor */
[WX5]  2Ø1:      while(!*(playfield+px+(py<<playy))||
[996]  2Ø2:     *(playfield+px+(py<<playy))&UNVISIBLE)
[P7Ø]  2Ø3:        {
[SP4]  2Ø4:         if(*(playfield+px+(py<<playy))
[HX1]  2Ø5:                          &INVISIBLE)
[P7Ø]  2Ø6:           {
[F51]  2Ø7:            if(vy>Ø)
[ZW4]  2Ø8:             laenge=((px<<SHIFT)-(long)xØ);
[ZRØ]  2Ø9:           else
[DC3]  21Ø:             laenge=(((px<<SHIFT)-
[183]  211:             (long)xØ+ØxØ1ØØ)*vy);
[RØ3]  212:           saveobj(xØ,yØ,w,px
[2F3]  213:           +(py<<playy),laenge);
[U7Ø]  214:           }
[MA1]  215:          px += vy;
[U7Ø]  216:        }
[F51]  217:      if(vy>Ø)
[ZW4]  218:       laenge=((px<<SHIFT)-(long)xØ);
[ZRØ]  219:      else
[L93]  22Ø:        laenge=(((px<<SHIFT)
[YØ3]  221:         -(long)xØ+ØxØ1ØØ)*vy);
[ET3]  222:      return(px+(py<<playy));
[U7Ø]  223:     }
[T95]  224:     if(!wco) /* Kann Y-Gerade erreicht
[PN1]  225:                 werden ? */
[U84]  226:     {     /* nein parallel zu Y */
[BE1]  227:      lx=ly=ØL;
[P62]  228:      side  = XSIDE;
[564]  229:      py += vx; /*erst eins vor */
[GA5]  23Ø:      while(!*(playfield+px+(py<<playy))
[5W2]  231:      ||*(playfield+px
[ØJ4]  232:              +(py<<playy))&UNVISIBLE)
[P7Ø]  233:        {
[SP4]  234:         if(*(playfield+px+(py<<playy))
[HX1]  235:                           &INVISIBLE)
[P7Ø]  236:           {
[D51]  237:            if(vx>Ø)
[7X4]  238:             laenge=((py<<SHIFT)-(long)yØ);
[ZRØ]  239:            else
[M93]  24Ø:             laenge=(((py<<SHIFT)
[YØ3]  241:             -(long)yØ+ØxØ1ØØ)*vx);
[RØ3]  242:           saveobj(xØ,yØ,w,px
[2F3]  243:           +(py<<playy),laenge);
[U7Ø]  244:           }
[PA1]  245:          py += vx;
[U7Ø]  246:        }
[D51]  247:      if(vx>Ø)
[7X4]  248:       laenge=((py<<SHIFT)-(long)yØ);
[ZRØ]  249:      else
[M93]  25Ø:       laenge=(((py<<SHIFT)
[YØ3]  251:                -(long)yØ+ØxØ1ØØ)*vx);
[ET3]  252:      return(px+(py<<playy));
[U7Ø]  253:     }
[W94]  254:     lx = (ax<<SHIFT)/(long)wsi;
[K94]  255:     ly = (ay<<SHIFT)/(long)wco;
[EG3]  256:     stepx = ix/(long)wsi;
[TE3]  257:     stepy = iy/(long)wco;
[HP2]  258:     vx = vx<<playy;
[L63]  259:     pf= px+(py<<playy);
[571]  26Ø:     while(1)
[BN4]  261:      {ml: /* zur beschleunigung */
[D35]  262:       /* Schnittpunkt zur nächsten
[K43]  263:          X-Geraden näher ? */
[D32]  264:       while(lx < ly)
[P7Ø]  265:         {
[NL5]  266:          /* Sprung über X-Gerade also y ++*/
[X81]  267:           pf += vx;
[RY2]  268:          if(*(playfield+pf))
[P7Ø]  269:            {
[P62]  27Ø:             side  = XSIDE;
[MK4]  271:             if(*(playfield+pf)&UNVISIBLE)
[P7Ø]  272:               {
[8K4]  273:                if(*(playfield+pf)&INVISIBLE)
[MR3]  274:                 saveobj(xØ,yØ,w,pf,lx);
[U7Ø]  275:               }
[ZRØ]  276:             else
[GZ1]  277:               {   /* Mauer */
[VN1]  278:                laenge=lx;
[9W1]  279:                return(pf);
[U7Ø]  28Ø:               }
[U7Ø]  281:            }
[765]  282:          /* Nächste X-gerade einstellen */
[UY1]  283:          lx += stepx;
[U7Ø]  284:         }
[7T2]  285:       while(lx >=ly)
[P7Ø]  286:         {
[GF5]  287:          /* Sprung über Y-Gerade also x++*/
[Y81]  288:          pf += vy;
[RY2]  289:          if(*(playfield+pf))
[P7Ø]  29Ø:            {
[S62]  291:             side  = YSIDE;
[MK4]  292:            if(*(playfield+pf)&UNVISIBLE)
[P7Ø]  293:              {
[8K4]  294:              if(*(playfield+pf)&INVISIBLE)
[PR3]  295:                saveobj(xØ,yØ,w,pf,ly);
[U7Ø]  296:              }
[ZRØ]  297:            else
[GZ1]  298:              {   /* MAUER */
[XN1]  299:               laenge=ly;
[9W1]  3ØØ:               return(pf);
[U7Ø]  3Ø1:              }
[U7Ø]  3Ø2:            }
[865]  3Ø3:          /* Nächste Y-Gerade einstellen */
[1Z1]  3Ø4:          ly += stepy;
[U7Ø]  3Ø5:         }
[G85]  3Ø6:       goto ml; /* zur beschleunigung */
[U7Ø]  3Ø7:      }
[U7Ø]  3Ø8:   }
[PV7]  3Ø9: /* setwinkel: setzt Winkelwerte für cos und
[TW3]  31Ø:  sinus in wcos und wsin.
[586]  311:  Abhängig von Bildschirmbreite width
[2Z1]  312:  Ø<=width<=32Ø
[EV5]  313:  und sichtbarem Winkel sw: 3Ø<=sw<=9Ø.
[3T4]  314:  Liefert Anzahl der benutzen
[586]  315:  Variable zurück. Falls Ø, dann Fehler.
[DN5]  316:  Setzt numw auf Anzahl der Winkel. */
[964]  317: int setwinkel(width,sw)
[682]  318: int width,sw;
[A84]  319:  {int anzpi=9Ø*width/sw,i;
[143]  32Ø:   if(anzpi>WIDTHMAX)
[TLl]  321:    return(Ø);
[GF3]  322:   for(i=Ø;i<anzpi*4;i++)
[6V5]  323:    {wsin[i]=(wsize)(sin((double)sw*i
[AL4]  324:      *PI/18Ø.Ø/width)*(1L<<SHIFT));
[PK5]  325:     wcos[i]=(wsize)(cos((double)sw*i
[Y25]  326:      *PI/18Ø.Ø/width)*(1L<<SHIFT));}
[MT2]  327:   return(numw=i);}
[NZØ]  328: #endif
[YP5]  329: unsigned xkoord(int side,long lx,
[DØ7]  33Ø: long ly,int wink,unsigned x,unsigned y);
[5X2]  331: int normw(int w);
[LG6]  332: /* normw(): normalisiert Winkelnummer */
[N52]  333: int normw(w)
```

```
[B21]  334: int w;
[WA1]  335:  {if(w<Ø)
[FW1]  336:     w = numw+w;
[ZRØ]  337:   else
[YU1]  338:    if(w>=numw)
[DM1]  339:       w =w%numw;
[M42]  34Ø:  return(w);}
[D75]  341: /* xkoord(): berechnet relative x-
[714]  342:  Koordinate auf Texture */
[5B6]  343: unsigned xkoord(side,lx,ly,wink,x,y)
[SE2]  344: int side,wink;
[322]  345: long lx,ly;
[Ø62]  346: unsigned x,y;
[8Ø3]  347:  {if(side == XSIDE)
[B33]  348:    {if(wcos[wink]>Ø)
[FB4]  349:       return((unsigned)(((ly-lx)*
[9T2]  35Ø:       wcos[wink])>>8));
[ZRØ]  351:     else
[SP2]  352:      if(wcos[wink]<Ø)
[ZB4]  353:       return((unsigned)((Øx1ØØØØL+
[W24]  354:        (ly-lx)*wcos[wink])>>8));
[ZRØ]  355:      else
[KD3]  356:       return(256-(x&Øxff));}
[ZRØ]  357:   else
[Y33]  358:     {if(wsin[wink]>Ø)
[EB4]  359:       return((unsigned)(((lx-ly)*
[WT2]  36Ø:       wsin[wink])>>8));
[ZRØ]  361:     else
[KP2]  362:      if(wsin[wink]<Ø)
[ZB4]  363:       return((unsigned)((Øx1ØØØØL+
[E34]  364:        (lx-ly)*wsin[wink])>>8));
[ZRØ]  365:     else
[M14]  366:       return(256-(y&ØxØØff)); })
[BP6]  367: /* Speichert Objekte mit durchsichtigem
[RU3]  368:    Hintergrund in Array */
[3W2]  369: typedef struct (
[E92]  37Ø: unsigned pos;
[NB2]  371: long way,way2;
[G36]  372: int xi,xl;    /* Anfang- und End-Pixel */
[166]  373: long xanflx,xanfly; /* Koordinaten  */
[GH3]  374: long xendlx,xendly;
[3U2]  375: int xside,nside;
[XE3]  376: int wink,xØ,yØ,wink2;
[KZ1]  377: void *alive;
[HG1]  378: } object;
[F11]  379: int i;
[N32]  38Ø: long factor;
[AG2]  381: spielfeld *spf;
[VZ2]  382: object Object[3Ø];
[UA2]  383: int objcount;
[295]  384: int saveobj(int xØ,int yØ,int w,
[214]  385: unsigned pos,long way);
[M16]  386: void showobjtext(int i,int  l1,int l2,
[ØU6]  387: int sb,text *tex,int txbegin,int txend,
[5P1]  388: int side);
[AA5]  389: /* saveobj(): speichert gesehenes
[ZP1]  39Ø:  Objekt ab */
[YU4]  391: int saveobj(xØ,yØ,w,pos,way)
[E92]  392: unsigned pos;
[GX1]  393: int xØ,yØ,w;
[WG1]  394: long way;
[HC1]  395:  {int x;
[KV3]  396:   for(x=Ø;x<objcount;x++)
[YT3]  397:    if(Object[x].pos==pos)
[GP3]  398:      {Object[x].way2 =way;
[DT2]  399:       Object[x].xl=i;
[9R3]  4ØØ:       Object[x].xendlx=lx;
[ER3]  4Ø1:       Object[x].xendly=ly;
[173]  4Ø2:       Object[x].wink2=w;
[1R3]  4Ø3:       Object[x].nside=side;
[D32]  4Ø4:       return(Ø);}
[8B4]  4Ø5:   /* neues Object eintragen */
[SG3]  4Ø6:   Object[x].pos = pos;
[UT2]  4Ø7:   Object[x].xi = i;
[PS3]  4Ø8:   Object[x].xanflx = lx;
[TS3]  4Ø9:   Object[x].xanfly = ly;
[K24]  41Ø:   Object[x].xside = side;
[BB3]  411:   Object[x].wink = w;
[BY2]  412:   Object[x].xØ = xØ;
[GY2]  413:   Object[x].yØ = yØ;
[EH3]  414:   Object[x].way = way;
[W66]  415:   Object[x].alive = (*(spf->firstthing+
[EE2]  416:    (*(spf->playmap+
[C45]  417:     Object[x].pos)&ØxØfff)))->alive;
[N62]  418:   objcount++; }
[ZY5]  419: /* void showobjtext(): Stellt Objekt-
[LC2]  42Ø:  Texture dar */
[2W6]  421: void showobjtext(i,l1,l2,sb,tex,txbegin,
[EX1]  422: txend,side)
[F53]  423: int i,l1,l2,sb,side;
[183]  424: int txbegin,txend;
[ZP1]  425: text *tex;
[F74]  426:  {register unsigned int j;
[KU2]  427:  register int y,z;
[S91]  428:  long x;
[4L2]  429:  int ix,iht,dim;
[SH1]  43Ø:  char *map;
[EN2]  431:  unsigned char c;
[8L2]  432:  if(side ==XSIDE)
[5R2]  433:   {dim = tex->xdim;
[5W2]  434:    map = tex->xmap;}
[ZRØ]  435:  else
[7R2]  436:   {dim = tex->ydim;
[6W2]  437:    map = tex->ymap;}
[KF4]  438:  txbegin = (txbegin)>>(8-dim);
[ØW3]  439:  txend = (txend)>>(8-dim);
[3D1]  44Ø:  st=l1-l2;
[581]  441:  if(sb<=Ø)
[KC1]  442:   return;
[ZA5]  443:  sx=(long)((txend-txbegin)<<6)/sb;
[FV1]  444:  vidb = vid+i;
[KC2]  445:  vide = vidb+sb;
[M24]  446:  vidu = vidb+hy+hy-xcount;
[UF5]  447:  for(x=txbegin<<6,tt=Ø;vidb<vide;
[AE4]  448:      vidb++,vidu++,x+=sx,tt+=st)
[LY1]  449:    {ix = x>>6;
[5U3]  45Ø:     iht = l1-tt/sb&ØxffffE;
[852]  451:     if(ysize>iht)
[6H4]  452:      {j= xcount*((ysize-iht)>>1);
[YUØ]  453:       y=Ø;}
[ZUØ]  454:     else
[5Z3]  455:      {j=Ø; y=(iht-ysize)>>1;}
[UB3]  456:     savet=(char  *)map+ix;
[5A4]  457:     for(;j<hy;j += xcount,y ++)
[ZP3]  458:        {z=(((long)y<<dim)/iht);
[FX2]  459:         c=*(savet+(z<<dim));
[GK2]  46Ø:         if(c!=BGVISIBLE)
[EX1]  461:          *(vidb+j) = c;
[PS4]  462:         c=*(savet+(((l<<dim)-z-1)<<dim));
[GK2]  463:         if(c!=BGVISIBLE)
[AV2]  464:          *(vidu-j)=c; }}}
[WZ5]  465: /* void showobj(): Stellt ganzes
[7X1]  466:  Objekt dar */
[G32]  467:   unsigned hi;
[ØC3]  468:   unsigned tanf,tend;
[N24]  469:   int cx,dx,ex,fx,gx,ax,bx;
[UP3]  47Ø: void showobj(width,pf)
[US1]  471: int width;
[672]  472: spielfeld *pf;
[HC1]  473:  {int x;
[ZA4]  474:   for(x=objcount-1;x>=Ø;x--)
[KN3]  475:    {if(!Object[x].alive)
[864]  476:      {if(Object[x].way>=Øx2Ø)
[BC5]  477:       hi = (int)(factor/Object[x].way);
[ZRØ]  478:      else
[5SØ]  479:       hi=Ø;
[V31]  48Ø:      if(hi<Ø)
```

```
[5SØ]  481:       hi=Ø;
[Y25]  482:      tanf=xkoord(Object[x].xside,
[436]  483:      Object[x].xanflx,Object[x].xanfly,
[VZ4]  484:      Object[x].wink,Object[x].xØ,
[KF2]  485:      Object[x].yØ);
[Ø25]  486:      tend=xkoord(Object[x].nside,
[236]  487:      Object[x].xendlx,Object[x].xendly,
[M55]  488:      Object[x].wink2,Object[x].xØ,
[KF2]  489:      Object[x].yØ);
[4T2]  49Ø:      ax=tanf;bx=tend;
[HX3]  491:      if(Object[x].way2>=Øx2Ø)
[XB1]  492:       if(hi<8ØØ)
[G26]  493:       showobjtext(Object[x].xi+width,hi,
[HT4]  494:       (int)(factor/Object[x].way2),
[3V4]  495:      Object[x].xl-Object[x].xi-1,
[755]  496:      (*(pf->firstthing+(*(pf->playmap+
[EK5]  497:      Object[x].pos)&Øx9fff)))->texture,
[G64]  498:      ax,bx,Object[x].xside);}
[ZRØ]  499:     else
[VN3]  5ØØ:      { /* Höhe errechnen */
[8W3]  5Ø1:     if(Object[x].way2>=Øx2Ø
[9R3]  5Ø2:       &&Object[x].way>=Øx2Ø)
[ØY2]  5Ø3:      hi = (int)(2*factor/
[PC5]  5Ø4:       (Object[x].way+Object[x].way2));
[ZRØ]  5Ø5:     else
[5SØ]  5Ø6:      hi=Ø;
[V31]  5Ø7:      if(hi<Ø)
[5SØ]  5Ø8:       hi=Ø;
[4Z3]  5Ø9:      /* cx Breite des Objekts*/
[6L5]  51Ø:      cx=((Object[x].xl-Object[x].xi));
[M41]  511:      if(cx<Ø)
[C71]  512:       cx = -cx;
[CB1]  513:      if(cx>hi)
[DL3]  514:       {fx=(cx-hi)/2;cx=hi;}
[ZRØ]  515:      else
[ETØ]  516:       fx=Ø;
[Y25]  517:      tanf=xkoord(Object[x].xside,
[436]  518:      Object[x].xanflx,Object[x].xanfly,
[VZ4]  519:      Object[x].wink,Object[x].xØ,
[KF2]  52Ø:      Object[x].yØ);
[HC1]  521:      ax=tanf;
[ØD4]  522:      if(Object[x].xside==YSIDE
[EL4]  523:       &&wcos[Object[x].wink]>=Ø)
[TW1]  524:       ax=Øx1ØØ-ax;
[YC4]  525:      if(Object[x].xside==XSIDE
[VK4]  526:       &&wsin[Object[x].wink]<=Ø)
[TW1]  527:       ax=Øx1ØØ-ax;
[Ø25]  528:      tend=xkoord(Object[x].nside,
[236]  529:      Object[x].xendlx,Object[x].xendly,
[M55]  53Ø:      Object[x].wink2,Object[x].xØ,
[KF2]  531:      Object[x].yØ);
[LC1]  532:      bx=tend;
[2C4]  533:      if(Object[x].nside==YSIDE
[XT4]  534:       &&wcos[Object[x].wink2]>=Ø)
[YW1]  535:       bx=Øx1ØØ-bx;
[ØC4]  536:      if(Object[x].nside==XSIDE
[GT4]  537:       &&wsin[Object[x].wink2]<=Ø)
[YW1]  538:       bx=Øx1ØØ-bx;
[EC6]  539:      if(Object[x].nside==Object[x].xside)
[1U1]  54Ø:       {dx=bx-ax;
[X41]  541:        bx=Ø;}
[ZRØ]  542:      else
[2N1]  543:       dx=Øx1ØØ-ax;
[VG4]  544:      if(Object[x].xside==YSIDE)
[1Ø5]  545:       {if(wcos[Object[x].wink]>=Ø)
[BT5]  546:         dx=dx*wcos[Object[x].wink]/Øx1ØØ;
[ZRØ]  547:       else
[WC6]  548:         dx=dx*-wcos[Object[x].wink]/Øx1ØØ;}
[TG4]  549:      if(Object[x].xside==XSIDE)
[GØ5]  55Ø:       {if(wsin[Object[x].wink]<=Ø)
[2X5]  551:         dx=dx*-wsin[Object[x].wink]/Øx1ØØ;
[ZRØ]  552:       else
[R46]  553:         dx=dx*wsin[Object[x].wink]/Øx1ØØ;}
[XF4]  554:      if(Object[x].nside==YSIDE)
[455]  555:       {if(wcos[Object[x].wink2]>=Ø)
[ZØ6]  556:         bx=bx*wcos[Object[x].wink2]/Øx1ØØ;
[ZRØ]  557:       else
[F46]  558:         bx=bx*-wcos[Object[x].wink2]/Øx1ØØ;}
[VF4]  559:      if(Object[x].nside==XSIDE)
[H55]  56Ø:       {if(wsin[Object[x].wink2]<=Ø)
[CØ6]  561:         bx=bx*-wsin[Object[x].wink2]/Øx1ØØ;
[ZRØ]  562:       else
[8G6]  563:         bx=bx*wsin[Object[x].wink2]/Øx1ØØ;}
[VG4]  564:      if(Object[x].xside==YSIDE)
[1Ø5]  565:       {if(wcos[Object[x].wink]>=Ø)
[2T5]  566:         ax=ax*wcos[Object[x].wink]/Øx1ØØ;
[ZRØ]  567:       else
[LC6]  568:         ax=ax*-wcos[Object[x].wink]/Øx1ØØ;}
[TG4]  569:      if(Object[x].xside==XSIDE)
[GØ5]  57Ø:       {if(wsin[Object[x].wink]<=Ø)
[TW5]  571:         ax=ax*-wsin[Object[x].wink]/Øx1ØØ;
[ZRØ]  572:       else
[E46]  573:         ax=ax*wsin[Object[x].wink]/Øx1ØØ;}
[F92]  574:      bx=bx+dx+ax;
[AP1]  575:      if(bx>Øxff)
[5C1]  576:       bx=Øxff;
[XB1]  577:     if(hi<8ØØ)
[9H6]  578:      showobjtext(Object[x].xi+width+fx,hi,
[XX3]  579:      hi,cx-1,(*(pf->firstthing+
[ER4]  58Ø:      (*(pf->playmap+Object[x].pos)
[M86]  581:      &Øx9fff)))->texture,ax,bx,XSIDE); }}}
[LM6]  582: /* void showwelt(unsigned x,unsigned y,
[235]  583:  int w, int size,spielfeld *pf); :
[5E5]  584:  Darstellung eines 3D-Bildes von
[7X4]  585:  bestimmter Position x,y mit
[FB5]  586:  Blickwinkel w auf Spielfeld pf */
[L77]  587: void showwelt(unsigned x,unsigned y,int w,
[YØ4]  588: int size,spielfeld *pf);
[XH1]  589:    int oldi;
[ZX4]  59Ø:    int wink,lastwink,helpwink;
[8LJ]  591:    unsigned obj,lastobj;
[VK3]  592:    int height,oldheight;
[Y32]  593:    int oldside;
[7X2]  594:    long oldlx,oldly;
[ØC3]  595:    unsigned tanf,tend;
[XY4]  596: void showwelt(x,y,w,size,pf)
[Ø62]  597: unsigned x,y;
[212]  598: int w,size;
[672]  599: spielfeld *pf;
[C72]  6ØØ:  {objcount =Ø;
[Ø46]  6Ø1:   factor= ((long)size<<8)+((long)size<<7);
[5P2]  6Ø2:   size = size>>1;
[BU3]  6Ø3:   /* erstes Objekt sehen */
[VK2]  6Ø4:   oldi = i  = -size;
[4B4]  6Ø5:   lastwink  = normw(w+oldi);
[VT5]  6Ø6:   lastobj   = showpos(x,y,lastwink);
[RD2]  6Ø7:   if(laenge-Øx2Ø>Ø)
[DH4]  6Ø8:    oldheight = (factor/laenge);
[ZRØ]  6Ø9:   else
[XZ1]  61Ø:    oldheight=Ø;
[RL2]  611:   oldside   = side;
[E62]  612:   oldlx     = lx;
[L62]  613:   oldly     = ly;
[FS3]  614:   /* Hintergrund setzen*/
[A93]  615:    for(i=Ø;i<16ØØØ;i+=32Ø)
[UN5]  616:    {setmem(vid+i,(size<<1)-1,ceilcol);
[9L4]  617:     setmem(vid+i+16ØØØ,(size<<1)-1,
[772]  618:       floorcol);}
[PT3]  619:    for(i=-size;i<size;i++)
[V93]  62Ø:    {wink = normw(w+i);
[EX3]  621:     obj=showpos(x,y,wink);
```

Listing 3. Die Bibliothek »3danim.c« enthält alle Funktionen zur Darstellung dreidimensionaler Welten.



DOS EXTRA 4
Speicherresidente Programme
DTP-Programm
EGA-Grafik
MS-DOS intern

DOS EXTRA 5
PC-Grafikkarten
Das BIOS des PC
GW-Basic
Gerätetreiber programmieren

DOS EXTRA 6
Basic im Detail
GW-Basic intern
SBasic-Compiler
Basic-Utilities

DOS EXTRA 7
PC-Komponenten
Sprite-Editor
SBasic Plus
C-Grundlagen

DOS EXTRA 8
Assembler SAM III
Assembler-Werkstatt
Assembler-Grundlagen

DOS EXTRA 10
MS-DOS 4.01
Hardware-Praxis
Snapshot-Programm
Turbo-Pascal-Units

DOS EXTRA 12
Grafik in Basic
Hardcopy-Utility
Fraktalgrafik in Turbo Pascal
Basic für PC-Grafikkarten

DOS EXTRA 13
Die besten Spiele
Abenteuerspiele
Flugsimulator
Sharewarespiele

DOS EXTRA 14
Datensicherungsprogramm
Zeichensatzeditor
Barcode-Darstellung

DOS EXTRA 15
SBasic Version 2.0
Skeybgr Version 4.0
Schiffe versenken
Simulation von Tastatureingaben

DOS EXTRA 16
Batch-Compiler der S-Klasse
Präprozessoren für MASM und TASM
Assembler-Befehle und Interrupts
RS-232-Programmierung

DOS EXTRA 17
Transpiler: Basic nach C
Flexibles Kopierprogramm für alle Formate
Formatierer der S-Klasse
BGI-Treiber für Super-VGA
Kiloweise Hilfsprogramme für jeden Zweck

DOS EXTRA 18
universelle Entwickleroberfläche
PCX-Bilder in Basic
Pretty Printer für Turbo Pascal
Tips, Tricks und Tools
kiloweise Hilfsprogramme für jeden Zweck

DOS EXTRA 19
kompletter Raytracer in C
schnelle Sprites mit Pascal
Animation mit Quick Basic
Vektorgrafik in Echtzeit
Superschnelle Fraktale
Tips und Tricks

DOS EXTRA

CeBIT'93 HANNOVER
24. — 31. 03. 1993
Halle 7, Stand E13/E19



DOS EXTRA Nr. 21

Aus dem Inhalt

- Der Assembler-Kurs zum Mitmachen
- Der Makro-Assembler SAM 4 zum Abtippen
- Komplette Referenz aller BIOS-Interrupts und Prozessorbefehle
- MS-DOS-Funktionen durchleuchtet
- Gerätetreiber selbstgemacht

Vorsprung für PC-Benutzer mit DOS EXTRA im Abonnement!

4 Ausgaben bei vierteljährlicher Erscheinungsweise mit dem bequemen Zustellerservice per Post:

BRD	DM 60,–
Europa	DM 70,–
Außereuropäisches Ausland	DM 80,–

Keine Bestellprobleme mit dem Dauerbezug der DOS-EXTRA-DATABOX!

Bequemes Dauerbezugsverfahren mit Bankabbuchung und vierteljährlicher Kündigungsmöglichkeit:

BRD	DM 39,–
Europa	DM 41,–
Außereuropäisches Ausland	DM 42,–

DATABOX

Die Software zu den DOS-EXTRA-Heften

Sagenhaft günstig, extra viel gute Software fürs Geld! Bringt alle Programme der DOS EXTRA auf Datenträger

Den Aufbau der Databox-Diskette finden Sie in der Datei »read.me« erklärt. Geben Sie deshalb vor dem Start »type read.me« ein. Sämtliche Pascal- und C-Programme liegen auch in einer compilierten Form für Hercules- und Farbgrafikkarten vor. Es ist allerdings nicht gewährleistet, daß diese Versionen auf Hardware, die nicht vollständig kompatibel zum IBM-PC ist, problemlos laufen. In diesem Fall müssen Sie den Quellcode nach der Anpassung mit dem entsprechenden Compiler neu übersetzen.

Die Basic-Programme arbeiten mit dem GW-Basic-Interpreter von Microsoft einwandfrei. Bei Interpretern und Compilern anderer Hersteller kann es zu Problemen kommen, sofern diese nicht vollkompatibel sind.

jeweils **2 Disketten 5 1/4"**
1 Diskette 3 1/2"

35,– DM
Wenn Sie über den DMV-Bestellservice bestellen, gilt folgendes:

Inland:		**Ausland:**	
Einzelpreis	35,– DM	Einzelpreis	35,– DM
zzgl. Versandkosten	4,– DM	zzgl. Versandkosten	6,– DM
Endpreis	**39,– DM**	**Endpreis**	**41,– DM**

Unabhängig von der Anzahl der bestellten Produkte berechnen wir für das Inland 4,– DM bzw. für das Ausland 6,– DM Porto und Verpackung.

DMV-Verlag • Postfach 250 • 3440 Eschwege

DMV Daten- und Medienverlag

```
[KH4]  622:     /*siehst du neues Objekt ?*/
[KF5]  623:     if(obj!=lastobj ¦¦ side!=oldside)
[GY4]  624:      (/* Altes Objekt einstellen */
[2W2]  625:       obj=showpos(x,y,
[4Ø4]  626:                   helpwink=normw(wink-1));
[RD2]  627:       if(laenge-Øx2Ø>Ø)
[3Z3]  628:        height = (factor/laenge);
[ZRØ]  629:       else
[GF1]  63Ø:        height=Ø;
[UU1]  631:       if(height<Ø)
[GF1]  632:        height=Ø;
[BP5]  633:       tanf=xkoord(oldside,oldlx,oldly,
[LB2]  634:                                lastwink,x,y);
[XZ3]  635:       tend=xkoord(side,lx,ly
[6L2]  636:                                ,helpwink,x,y);
[ØE4]  637:       if(height<12ØØ&oldheight<12ØØ)
[Y76]  638:       showtext(oldi+size,oldheight,height
[214]  639:       ,i-oldi,(*(pf->firstthing+
[ØY4]  64Ø:       (*(pf->playmap+lastobj)&ØxØfff)))
[5M1]  641:       ->texture
[173]  642:      ,tanf,tend,oldside);
[W75]  643:       /* Aktuelles Objekt auf Anfang */
[SN2]  644:       lastwink = wink;
[HF5]  645:       lastobj  = showpos(x,y,lastwink);
[HK1]  646:       oldi     = i;
[RD2]  647:       if(laenge-Øx2Ø>Ø)
[GG4]  648:        oldheight=  (factor/laenge);
[ZRØ]  649:       else
[XZ1]  65Ø:        oldheight=Ø;
[3D2]  651:       if(oldheight<Ø)
[XZ1]  652:        oldheight=Ø;
[EF2]  653:       oldside  = side;
[KX1]  654:       oldlx    = lx;
[DD2]  655:       oldly    = ly; ))
[3X6]  656:     obj=showpos(x,y,helpwink=normw(wink-1));
[3Z3]  657:     height = (factor/laenge);
[UU1]  658:     if(height<Ø)
[GF1]  659:      height=Ø;
[BP5]  66Ø:     tanf=xkoord(oldside,oldlx,oldly,
[LB2]  661:       lastwink,x,y);
[5G6]  662:     tend=xkoord(side,lx,ly,helpwink,x,y);
[ØE4]  663:     if(height<12ØØ&oldheight<12ØØ)
[A96]  664:      showtext(oldi+size,oldheight,height,
[D34]  665:      i-oldi-1,(*(pf->firstthing
[L95]  666:      +(*(pf->playmap+lastobj)&ØxØfff)))
[KX4]  667:      ->texture,tanf,tend,oldside);
[KZ2]  668:     showobj(size,pf);
[P12]  669:     showarm(pf);
[E93]  67Ø:     for(i=Ø;i<32ØØØ;i+=32Ø)
[YY5]  671:     movmem(vid+i,virtvid+i,(size<<1)-1);}
[8S6]  672: /* initpf():initialisiert Darstellung und
[HK1]  673:  Spielfeld
[ZB2]  674:  pf: *Spielfeld
[ØE3]  675:  t : Top des Fensters
[LD3]  676:  l :left des Fensters
[FH2]  677:  sw:Sichtwinkel
[8S3]  678:  ys: Höhe des Fensters
[9K4]  679:  width: Breite des Fensters */
[XH4]  68Ø: static char virtual[32ØØØ];
[615]  681: int initpf(pf,l,t,width,ys,sw)
[S64]  682: unsigned t,l,width,sw,ys;
[672]  683: spielfeld *pf;
[AS3]  684:  (setwinkel(width,sw);
[PC3]  685:   /* Tabelle der Winkel */
[XC1]  686:   if(!numw)
[335]  687:    (printf("Keine Winkeltabelle "
[ØM2]  688:     "vorhanden\n");
[3E1]  689:     exit(Ø);)
[1Y3]  69Ø:   playfield = pf->playmap;
[4D1]  691:   spf = pf;
[Y83]  692:   playy = pf->playdim;
[TZ3]  693:   floorcol = pf->floorcol;
[7L3]  694:   ceilcol = pf->ceilcol;
[UA1]  695:   top = t;
[C81]  696:   left = l;
[YX1]  697:   ysize = ys;
[A53]  698:   hy = XCOUNT*(ys/2);
[PN1]  699:   initvid();
[962]  7ØØ:   virtvid=vid;
[GH4]  7Ø1:   vid=(char far *)&virtual[Ø];
[E32]  7Ø2:   return(1);}
[HN3]  7Ø3: #define sgn(c)  (c<Ø?-1:1)
[FE3]  7Ø4: #define MAXANIM        5
[XY4]  7Ø5: /* animate(): Animiert Objekte in
[NK2]  7Ø6:  animateListe */
[3W2]  7Ø7: typedef struct {
[9X5]  7Ø8:  text *t;      /* Adresse in Playfield */
[4R6]  7Ø9:  int textcount;  /* Anzahl der Animation */
[NL5]  71Ø:  int aktutext;   /* aktuelles Bild */
[SB6]  711:  text *ANIM[MAXANIM]; /* Animationen */
[DM1]  712: }animobj;
[ST2]  713: int animcount=Ø;
[F33]  714: animobj *animlist;
[9F1]  715: animate()
[4B1]  716:  (int i;
[RØ2]  717:   animobj *ao;
[ØX3]  718:   for(i=Ø;i<animcount;i++)
[A93]  719:    (ao=&(*(animlist+i));
[VA4]  72Ø:     memcpy((ao->t),*(ao->ANIM+ao
[VD4]  721:     ->aktutext),sizeof(text));
[WH2]  722:     ao->aktutext ++;
[5E5]  723:     if(ao->aktutext>=ao->textcount)
[V83]  724:      ao->aktutext=Ø; ))
[395]  725: /* int askkey(x,y,w,field,width):
[VE6]  726:  erlaubt ein Durchlaufen der 3D-Welt */
[EØ6]  727: int askkey(unsigned *x,unsigned *y,
[BS4]  728: unsigned *w,spielfeld *field,
[5S2]  729: unsigned width);
[LV2]  73Ø: #define KEYSTEP 2
[WT3]  731: #define KEYTURNSTEP 15
[434]  732: #define MOUSETURNSTEP 1
[U93]  733: #define MOUSESTEP 6
[RY2]  734: #define ANIMTIME 6
[2F5]  735: int  askkey(xb,yb,wb,field,width)
[UL4]  736: unsigned *xb,*yb,*wb,width;
[AR2]  737: spielfeld *field;
[1X4]  738:  (int fertig=Ø,xm,ym,wm,dx,dy;
[NK4]  739:   unsigned x=*xb,y=*yb,w=*wb;
[EN2]  74Ø:   unsigned char c;
[B73]  741:   unsigned anicount;
[GØ2]  742:   w=normw(w);
[FA6]  743:   mouse_dis(&xm,&ym); /*Einmal zu Beginn*/
[MG2]  744:   while(xm¦¦ym)
[FA6]  745:   mouse_dis(&xm,&ym); /*Einmal zu Beginn*/
[8Ø3]  746:   /* Erst mal sehen */
[3N4]  747:   showwelt(x,y,w,width,field);
[472]  748:   while(!fertig)
[915]  749:    (xm=ym=dx=wm=dy=anicount=Ø;
[WZ4]  75Ø:     while(!xm&&!ym&&!fertig&&!kbhit())
[TE3]  751:     (mouse_dis(&xm,&ym);
[VH2]  752:      if(mouse_key(1))
[VU5]  753:       (while(mouse_press());/*wait a sec*/
[GV1]  754:        fertig=1;)
[953]  755:      if(field->ich->le<=Ø)
[P74]  756:       (field->ich->status ¦=1;
[GV1]  757:       fertig=1;)
[8V3]  758:      if(anicount > ANIMTIME)
[3Z1]  759:       (animate();
[XW4]  76Ø:        showwelt(x,y,w,width,field);)
[ZRØ]  761:      else
[4X1]  762:       anicount ++;
[U7Ø]  763:     )
[2X1]  764:     if(!fertig)(
[8Y1]  765:     if(xm¦¦ym)
[P7Ø]  766:      (
[HVØ]  767:       if(xm)
[7G4]  768:         wm += xm/4*MOUSETURNSTEP/2;
```

```
[KVØ]  769:       if(ym)
[P7Ø]  77Ø:        (
[HR2]  771:         dy =(int) -(((long)
[7M4]  772:         wsin[w]*ym/8)>>MOUSESTEP);
[ER2]  773:         dx =(int) -(((long)
[ML4]  774:         wcos[w]*ym/8)>>MOUSESTEP);
[U7Ø]  775:        )
[U7Ø]  776:      )
[ZRØ]  777:     else
[P7Ø]  778:     (
[UK1]  779:     c=getch();
[6TØ]  78Ø:     if(!c)
[UK1]  781:      c=getch();
[PD1]  782:     switch(c)
[P7Ø]  783:      (
[A81]  784:       case '8':
[161]  785:       case 72:
[E44]  786:         dy =(wsin[w]>>KEYSTEP);
[V24]  787:         dx =(wcos[w]>>KEYSTEP);
[PØ1]  788:        break;
[K71]  789:       case '2':
[W51]  79Ø:       case 8Ø:
[N24]  791:         dy =-(wsin[w]>>KEYSTEP);
[X24]  792:         dx =-(wcos[w]>>KEYSTEP);
[PØ1]  793:        break;
[281]  794:       case '6':
[N61]  795:       case 77:
[HT2]  796:         w +=KEYTURNSTEP;
[PØ1]  797:        break;
[U71]  798:       case '4':
[D61]  799:       case 75:
[6U2]  8ØØ:         w -= KEYTURNSTEP;
[PØ1]  8Ø1:        break;
[BF2]  8Ø2:       case 27: /* ESC */
[TF1]  8Ø3:        fertig=1;
[8E1]  8Ø4:       default:
[DG1]  8Ø5:        break; ))
[GØ2]  8Ø6:     w=normw(w);
[KU2]  8Ø7:     if(sgn(dy)*dy>Øxff)
[3R2]  8Ø8:      dy = Øxff*sgn(dy);
[EU2]  8Ø9:     if(sgn(dx)*dx>Øxff)
[ZP2]  81Ø:      dx = Øxff*sgn(dx);
[YG5]  811:     if((*(playfield+(x>>SHIFT)+(((y+dy+
[LH5]  812:     sgn(dy)*Øx6Ø)>>SHIFT)<<playy))&AIR))
[DTØ]  813:       dy=Ø;
[RC5]  814:       if((*(playfield+((x+dx+sgn(dx)*Øx6Ø)
[FP5]  815:     >>SHIFT)+(((y)>>SHIFT)<<playy))&AIR))
[ATØ]  816:       dx=Ø;
[WN5]  817:     if((*(playfield+((x+dx+Øx6Ø)>>SHIFT)
[4Ø5]  818:     +(((y+dy)>>SHIFT)<<playy))&AIR)¦¦
[XZ4]  819:        (*(playfield+((x+dx-Øx6Ø)>>SHIFT)
[TN4]  82Ø:        +(((y+dy)>>SHIFT)<<playy))&AIR))
[D91]  821:       dy=dx=Ø;
[PE5]  822:     if((*(playfield+((x+dx)>>SHIFT)+(((y
[L75]  823:     +dy+Øx6Ø]>>SHIFT)<<playy))&AIR)¦¦
[SZ4]  824:        (*(playfield+((x+dx)>>SHIFT)+(((y
[AX4]  825:        +dy-Øx6Ø)>>SHIFT)<<playy))&AIR))
[C91]  826:       dx=dy=Ø;
[3H5]  827:     if(!(*(playfield+((x+dx+sgn(dx)*Øx4Ø)
[4Y4]  828:     >>SHIFT)+(((y+dy+sgn(dy)*Øx4Ø)>>
[F93]  829:     SHIFT)<<playy))&AIR))
[8X3]  83Ø:      (y +=dy;x +=dx;w +=wm;)
[UY2]  831:      *xb=x;*yb=y;*wb=w;
[UM1]  832:     animate();
[3N4]  833:     showwelt(x,y,w,width,field);
[B92]  834:     enable(); ))}
[NZØ]  835: #endif
```

Listing 3. Die Bibliothek »3danim.c« enthält alle Funktionen zur Darstellung dreidimensionaler Welten.

```
[4U3]   68:   intr(MOUSEINT, &reg);)
[HC5]   69: /* void mouse_def(): definiert das
[Ø74]   7Ø: Aussehen des Mauszeigers
[BB4]   71:  x,y : Position der Mitte */
[6W4]   72: void mouse_def(bitfield,x,y)
[6L2]   73: byte *bitfield;
[7V2]   74: unsigned int x,y;
[HU2]   75:  (reg.r_ax = ØxØ9;
[312]   76:   reg.r_bx = x;
[812]   77:   reg.r_cx = y;
[DL4]   78:   reg.r_es = FP_SEG(bitfield);
[XK4]   79:   reg.r_dx = FP_OFF(bitfield);
[4U3]   8Ø:   intr(MOUSEINT, &reg);}
[DK5]   81: /* void mouse_pos(x,y):liefert die
[Y73]   82: Mausposition x,y */
[GC3]   83: void mouse_pos(x,y)
[N93]   84: unsigned int *x,*y;
[BU2]   85:  (reg.r_ax = Øx3;
[DE3]   86:   intr(MOUSEINT, &reg);
[D12]   87:   *x=reg.r_cx;
[D92]   88:   *y=reg.r_dx;}
[AY6]   89: /* unsigned int mouse_press():liefert den
[X85]   9Ø:  Maustastenstatus: Bit Ø= linke,
[486]   91:  Bit 1 = rechte, Bit 2=mittlere Taste */
[PL4]   92: unsigned int mouse_press()
[BU2]   93:  (reg.r_ax = ØxØ3;
[DE3]   94:   intr(MOUSEINT, &reg);
[V73]   95:   return(reg.r_bx);}
[TC6]   96: /* unsigned int mouse_key(nr):liefert,
[8G6]   97: wie oft eine Maustaste gedrückt wurde
[L83]   98:  Ø = linke Maustaste
[K83]   99:  1 = rechte Maustaste
[XY3]  1ØØ:  2 = mittlere Maustaste */
[PH4]  1Ø1: unsigned int mouse_key(nr)
[DU2]  1Ø2: unsigned int nr;
[DU2]  1Ø3:  (reg.r_ax = ØxØ5;
[S72]  1Ø4:   reg.r_bx = nr;
[DE3]  1Ø5:   intr(MOUSEINT, &reg);
[V73]  1Ø6:   return(reg.r_bx);}
[PT5]  1Ø7: /* void mouse_move(x,y): bewegt den
[C95]  1Ø8: Mauscursor an die Position x,y */
[MM3]  1Ø9: void mouse_move(x,y)
[7V2]  11Ø: unsigned int x,y;
[CU2]  111:  (reg.r_ax = ØxØ4;
[RA4]  112:   reg.r_cx = x; reg.r_dx = y;
[4U3]  113:   intr(MOUSEINT, &reg);}
[B17]  114: /* void mouse_area(left,top,right,bottom):
[SY5]  115: legt Area fest, in der der Mauscursor
[N73]  116: bewegt werden darf */
[KX6]  117: void mouse_area(left,top ,right,bottom)
[C36]  118: unsigned int left,top, right,bottom;
[XG5]  119:  (reg.r_ax = ØxØ7; reg.r_cx = left;
[RD6]  12Ø:   reg.r_dx = right; intr(MOUSEINT, &reg);
[K55]  121:   reg.r_ax = ØxØ8; reg.r_cx = top;
[RR6]  122:   reg.r_dx = bottom;intr(MOUSEINT, &reg);}
[ØØ6]  123: /* void mouse_dis(x,y):ermittelt die
[6E6]  124: Entfernung in Punkten seit dem letzten
[ZX3]  125: Aufruf zurück in x,y */
[SB3]  126: void mouse_dis(x,y)
[5M1]  127: int *x,*y;
[UU2]  128:  (reg.r_ax = ØxØb;
[DE3]  129:   intr(MOUSEINT, &reg);
[D12]  13Ø:   *x=reg.r_cx;
[D92]  131:   *y=reg.r_dx;}
[NZØ]  132: #endif
```

Listing 4. »Mouse.c« enthält die Mausroutinen.

```
[W6Ø]    1: /*
[FN2]    2:  Programm: defs.h
[AE6]    3:  Funktion: Definitionen und Strukturen
[H45]    4:                zur Darstellung von 3D-Welten
[A52]    5:                mit Objekten
[K83]    6:  Sprache:  Turbo C 2.Ø
[KR4]    7:  Autor:    Matthias Oberdorfer
[8Z4]    8:  (c) 1992 DMV Widuch GmbH & Co. KG */
[ØØØ]    9:
[RU1]   1Ø: #ifndef D3H
[XU1]   11: #define D3H
[7M2]   12: #include<dos.h>
[RN2]   13: #include<math.h>
[L12]   14: /* für talk */
[U73]   15: #define UNWICHTIG Ø
[US2]   16: #define WICHTIG 1
[663]   17: #define SATZLEN 128
[1Ø3]   18: #define ANZWORD 3Ø
[8M2]   19: typedef struct
[142]   2Ø:  (char *lexem;
[ØF1]   21:   int imp;
[SP5]   22:   int num; /* Anzahl der Antworten */
[UP4]   23:   char **(ans[8]); /* Verweise */
[1B1]   24:  } Word;
[B32]   25: /* Personen */
[8M2]   26: typedef struct
[3S3]   27:  (char feel[SATZLEN];
[ØB4]   28:   char thoughts[SATZLEN];
[9L3]   29:   char will[SATZLEN];
[2S3]   3Ø:   char future[SATZLEN];
[UP3]   31:   char should[SATZLEN];
[7H3]   32:   char need[SATZLEN];
[SG3]   33:   char have[SATZLEN];
[XD4]   34:   char can[SATZLEN];}view;
[8M2]   35: typedef struct
[432]   36:  (char *menge;
[YE2]   37:   char *andmenge;
[B22]   38:   int infnum;
[LM2]   39:   char **inf;}info;
[3W2]   4Ø: typedef struct {
[H12]   41:   char *inhalt;
[T21]   42:   char *intro;
[A12]   43:   info *first;
[NX5]   44:   int type;/* wichtig oder unwichig*/
[VU3]   45:   int (*action)();}thema;
[3W2]   46: typedef struct {
[ØU1]   47:  char *name;
[6X2]   48:   float motivpart;
[ZV4]   49:   /* Wahrscheinlichkeit motiv */
[TD2]   5Ø:   int motivnum;
[HZ1]   51:   char **motiv;
[KT2]   52:   float viewpart;
[XW4]   53:   /* Wahrscheinlichkeit views */
[552]   54:   view *myview;
[B62]   55:   view *yoview;
[M83]   56:   thema *thema;} mind;
[Z46]   57: extern talk(char *name, char *partner,
[GX1]   58: mind *pers);
[MX2]   59: struct spielfeld;
[K52]   6Ø: struct text;
[MU1]   61: #define ON Ø
[612]   62: #define OFF 1
[MZ3]   63: #define DURCHSICHTIG 255
[1F1]   64: /* Farbe */
[CØ4]   65: typedef struct color {
[TC3]   66:   double r,g,b;} color;
[7L4]   67: extern  void init_pal();
[UG5]   68: extern  int setgraphik(int mode);
[836]   69: extern  void setp(int x,int y,int f);
```

```
[RF6]   7Ø: extern  unsigned char getp(int x,int y);
[4Z5]   71: extern  void showpal(int xØ,int yØ);
[EG3]   72: extern  normw(int w);
[775]   73: extern  drawline(int xØ,int yØ,
[G53]   74: int x1,int y1,int f);
[TE7]   75: extern  unsigned interpret_col(color *col);
[RP6]   76: extern  void drawbox(int xØ,int yØ,int x1
[CT3]   77: ,int y1,int f,int fill);
[KX6]   78: extern  void cls(unsigned f,unsigned rf);
[985]   79: extern  void display(int mode);
[P56]   8Ø: extern  unsigned mouse_check(void);
[RA5]   81: extern  void mouse_show(void);
[575]   82: extern  void mouse_hide(void);
[GS6]   83: extern  void mouse_move(unsigned int x
[SP2]   84: , unsigned int y);
[FG5]   85: extern  void mouse_def(unsigned
[6U4]   86: char *bitfield,unsigned int x
[BT2]   87: ,unsigned int y);
[YN6]   88: extern  void mouse_pos(unsigned int *x
[CT2]   89: ,unsigned int *y);
[6Y6]   9Ø: extern  unsigned int mouse_press(void);
[NF3]   91: extern  unsigned int
[DM4]   92: mouse_key(unsigned int nr);
[997]   93: extern  void mouse_area(unsigned int left,
[XQ6]   94: unsigned int top ,unsigned int right
[FN3]   95: ,unsigned int bottom);
[B56]   96: extern  void mouse_dis(int *x,int *y);
[864]   97: typedef struct schalter
[FX6]   98:  (struct schalter *next;/*next Schalter*/
[CS5]   99:   unsigned left,top;    /* wie groß */
[ZN5]   1ØØ:  unsigned right,bottom;/* und wo */
[CY4]   1Ø1:  unsigned framecol,col,ilucol;
[KB3]   1Ø2:  unsigned framecol2;
[GØ3]   1Ø3:  unsigned textcol;
[3X2]   1Ø4:  unsigned backcol;
[CM3]   1Ø5:  char *msg;) schalter;
[S86]   1Ø6: extern  unsigned testsch(unsigned x,
[CS6]   1Ø7: unsigned y,schalter *sch,schalter **sha);
[827]   1Ø8: extern  void print(char *str,unsigned co);
[AØ6]   1Ø9: extern  void printnum(unsigned nr);
[285]   11Ø: extern  void getstr(char *str);
[DY5]   111: extern  void showsch(schalter *sch);
[HD6]   112: extern  unsigned showpos(unsigned xØ,
[H93]   113: unsigned yØ,int w);
[3W2]   114: typedef struct (
[E92]   115: unsigned pos;
[WG1]   116: long way;
[G36]   117: int xi,x1;    /* Anfang- und End-Pixel */
[166]   118: long xanflx,xanfly; /* Koordinaten  */
[GH3]   119: long xendlx,xendly;
[ST1]   12Ø: int xside;
[GC5]   121: int wink,xØ,yØ,wink2; ) object;
[834]   122: extern object Object[3Ø];
[1E3]   123: extern int objcount;
[5V2]   124: #define XCOUNT 32Ø
[CM2]   125: #define YSIZE 114
[ØS2]   126: #ifndef SHOWPOS
[XE2]   127: #define YSIDE 1
[RE2]   128: #define XSIDE Ø
[NZØ]   129: #endif
[PX8]   13Ø: #define initvid() vid=(char far *)(ØxaØØØØØ
             ØØ+top*32Ø+left)
[MU3]   131: typedef struct text (
[K15]   132: int xdim; unsigned char *xmap;
[N96]   133: int ydim; unsigned char *ymap; ) text;
[CF2]   134: #define SHIFT 8
[F63]   135: #define WIDTHMAX 96Ø
[893]   136: #define PI 3.1415927464
[ØS2]   137: #ifndef SHOWPOS
[XE2]   138: #define YSIDE 1
[RE2]   139: #define XSIDE Ø
[NZØ]   14Ø: #endif
[BK3]   141: #define BGVISIBLE Øxff
[BN5]   142: extern const unsigned UNVISIBLE;
[LL4]   143: extern const unsigned AIR ;
[VL5]   144: extern const unsigned INVISIBLE;
[L26]   145: extern int numw;/* Anzahl der Winkel*/
[273]   146: #define OBJECT ØxaØØØ
[XØ3]   147: #define WALL Øx4ØØØ
[C94]   148: #define TRANSPARENT Øx2ØØØ
[WX2]   149: #include"defadv.h"
[HY5]   15Ø: extern int far initpf(spielfeld *pf,
[H76]   151: unsigned l,unsigned t,unsigned width,
[9F4]   152: unsigned ys,unsigned sw);
[HN3]   153: #define sgn(c)  (c<Ø?-1:1)
[FU2]   154: #define MAXANIM 5
[1V3]   155: extern void  animate();
[WD7]   156: extern void showwelt(unsigned x,unsigned y
[4Z4]   157: ,int w,int size,spielfeld *pf);
[3W2]   158: typedef struct (
[9X5]   159: text *t;        /* Adresse in Playfield */
[Z56]   16Ø: int textcount;  /* Anzahl der Animat. */
[NL5]   161: int aktutext;  /* aktuelles Bild */
[SB6]   162: text *ANIM[MAXANIM]; /* Animationen */
[DM1]   163: }animobj;
[4L3]   164: extern int animcount;
[274]   165: extern animobj *animlist;
[XV6]   166: extern int askkey(unsigned *x,unsigned *y,
[BS4]   167: unsigned *w,spielfeld *field,
[5S2]   168: unsigned width);
[LV2]   169: #define KEYSTEP 2
[WT3]   17Ø: #define KEYTURNSTEP 15
[834]   171: #define MOUSETURNSTEP 2
[P93]   172: #define MOUSESTEP 5
[VB3]   173: #define ANIMTIME 2ØØØ
[ØG5]   174: extern int printinbox(const char
[Ø22]   175: *format, ...);
[T84]   176: extern int blink(int col);
[SV5]   177: extern unsigned char pointer2[64];
[2A6]   178: #define MUSPORT *((unsigned far *)Øx18Ø)
[943]   179: extern mind dovan;
[9C3]   18Ø: extern mind miriam;
[ØD3]   181: extern mind miriam2;
[NZØ]   182: #endif
```

Listing 5. »Defs.h« enthält die Definitionen und Strukturen für die 3D-Welten.

```
[W6Ø]    1: /*
[LØ3]    2:  Programm: defadv.h
[HF6]    3:  Funktion: Defintionen für Rollenspiel
[K83]    4:  Sprache:  Turbo C 2.Ø
[KR4]    5:  Autor:    Matthias Oberdorfer
[HN4]    6:  (c) 1992 DMV Widuch GmbH & Co. KG
[17Ø]    7: */
[9P4]    8: typedef struct verfolger (
[414]    9: struct verfolger *next;
[T94]   1Ø: struct verfolger *prev;
[HV6]   11: unsigned posx,posy; /* Position intern */
[UL4]   12: unsigned type;      /* intern */
[D95]   13: int le;             /* LE des Monsters*/
[Z15]   14: int geschw;         /* wie schnell */
[MF4]   15: int counter;        /* intern*/
[P84]   16: int tp;             /* wie stark*/
[YS2]   17: )verfolger;
[6K3]   18: typedef struct thing
```

```
[ZH5]   19:  (char *name;          /* Name des Dings */
[XC6]   2Ø:   unsigned attribute; /* Eigenschaften */
[B45]   21:   unsigned status;    /* Zustand */
[AM5]   22:   int (*actpos)();    /* Bedienungsfunk */
[HE2]   23:   int (*actpos)();
[RC2]   24:   int (*actneg)();
[XR3]   25:   struct text *texture;
[ND5]   26:   void *alive;       /* Falls es lebt */
[UU4]   27:   verfolger *verf;   /* Monster */
[5E1]   28:  ) thing;
[TW2]   29: #define WEIBLICH 1
[U33]   3Ø: #define MAENNLICH 2
[EX2]   31: #define KRAFTANF 1
[3M2]   32: #define MAXCRY 8
[8M2]   33: typedef struct
[VE6]   34:  (char *name;/* Name unsere Abenteurers*/
[GT5]   35:   unsigned posx, posy; /* Position */
[Ø25]   36:   unsigned look; /* Blickwinkel */
[LS5]   37:   int selected; /* angklickte Sache */
[2F6]   38:   thing *carry[MAXCRY];/* Mitgefuehrte*/
[V85]   39:   unsigned nr[MAXCRY];/* Dinge */
[9G6]   4Ø:   unsigned status; /* Status Lebewesen */
[TP5]   41:   thing *waffe; /* Waffe zum Schlagen */
[HU4]   42:   int muede;    /* Ø ist zu muede */
[LA5]   43:   int hungrig;  /* Ø ist zu hugrig */
[266]   44:   int le;/* Lebensenergie: Null = tot */
[ZE3]   45:   unsigned long alter;
[HA4]   46:   int ae; /* Astralenergie */
[HH3]   47:   unsigned geschlecht;
[RS2]   48:   unsigned kraft;
[LG6]   49:   unsigned stufe; /* steigt automatisch */
[K92]   5Ø:  )adventurer;
[PW1]   51: #define LE 3Ø
[H53]   52: #define DEFWACH 2ØØØØ
[TB4]   53: #define DEFHUNG (DEFWACH/3)
[1U3]   54: #define TAG (DEFWACH*2L)
[VF4]   55: typedef struct spielfeld{
[482]   56: int *playmap;
[Z42]   57: int playdim;
[KA2]   58: int floorcol;
[F42]   59: int ceilcol;
[Y83]   6Ø: thing **firstthing;
[9S2]   61: adventurer *ich;
[R42]   62: }spielfeld;
```

Listing 6. In »defadv.h« sind wichtige Strukturen und Konstanten für das Rollenspiel definiert.

```
[W6Ø]    1: /*
[LW2]    2:  Programm: graph.h
[GX4]    3:  Funktion:  Graphikfunktionen
[K83]    4:  Sprache:  Turbo C 2.Ø
[KR4]    5:  Autor:    Matthias Oberdorfer
[8Z4]    6:  (c) 1992 DMV Widuch GmbH & Co. KG */
[ØØØ]    7:
[Y32]    8: #ifndef GRAPH
[442]    9: #define GRAPH
[7M2]   1Ø: #include<dos.h>
[RN2]   11: #include<math.h>
[MU1]   12: #define ON Ø
[612]   13: #define OFF 1
[MZ3]   14: #define DURCHSICHTIG 255
[PG1]   15: /*  Farbe */
[CØ4]   16: typedef struct color (
[MM3]   17:  double r,g,b; ) color;
[KP2]   18: void init_pal();
[564]   19: int setgraphik(int mode);
[LS4]   2Ø: void setp(int x,int y,int f);
[255]   21: unsigned char getp(int x,int y);
[FM4]   22: void showpal(int xØ,int yØ);
[SW5]   23: drawline(int xØ,int yØ,int x1,int y1,
[Z61]   24:   int f);
[446]   25: unsigned interpret_col(color *col);
[6G5]   26: void drawbox(int xØ,int yØ,int x1,
[8P3]   27:   int y1,int f,int fill);
[XK5]   28: void cls(unsigned f,unsigned rf);
[MX3]   29: void display(int mode);
[8R3]   3Ø: /* void init_pal (void);
[VA5]   31:   initialisiert RGB-Farbpalette */
[RM2]   32: void init_pal()
[MW1]   33:  (int r,g,b;
[R73]   34:   struct REGPACK reg;
[4A2]   35:   for(b=Ø;b<6;b++)
[2B2]   36:   for(g=Ø;g<6;g++)
[4D2]   37:   for(r=Ø;r<6;r++)
[M53]   38:    (reg.r_ax = Øx1Ø1Ø;
[463]   39:     reg.r_bx = r+g*6+b*36;
[YN5]   4Ø:     reg.r_cx = (unsigned int)((g)*12)<<8¦
[V92]   41:                (unsigned int)
[721]   42:                ((b)*12);
[BK5]   43:     reg.r_dx = (unsigned int)((r)*12)<<8;
[NM2]   44:     intr(Øx1Ø,&reg);)
[BL5]   45:    /* Dursichtige Farbe auf Schwarz */
[RZ4]   46:    reg.r_ax = Øx1Ø1Ø; reg.r_bx = 255;
[M54]   47:    reg.r_cx = Ø; reg.r_dx = Ø;
[KU2]   48:    intr(Øx1Ø,&reg); )
[TF6]   49: /* setgraphik(mode) : Ein- und Ausschalten
[AE5]   5Ø:  des  32Ø - Grafikmodus mit 256 Farben
[ZT5]   51:  Rückgabewerte : Ø := Keine VGA Karte
[7X2]   52:  1 := erfolgreich */
[EF3]   53: int setgraphik(mode)
[YH1]   54: int mode;
[MN3]   55:  (struct REGPACK reg;
[NE2]   56:   static int old;
[YZ2]   57:   /* test vga karte */
[9S2]   58:   reg.r_ax = Øx1aØØ;
[VC2]   59:   intr(Øx1Ø,&reg);
[6Ø8]   6Ø:   if((reg.r_bx&Øxff)<7)
[ØX5]   61:    (printf("VGA - Karte notwendig.\n");
[D32]   62:     return(Ø);)
[2H3]   63:   /* Videomodussetzen */
[MY1]   64:   if(mode==ON)
[NR3]   65:    (/* Videomodus holen */
[BM2]   66:     reg.r_ax=ØxØfØØ;
[VC2]   67:     intr(Øx1Ø,&reg);
[VD3]   68:     old = reg.r_ax & Øxff;
[BE2]   69:     reg.r_ax = Øx13;
[VC2]   7Ø:     intr(Øx1Ø,&reg);
[612]   71:     init_pal();)
[ZRØ]   72:   else
[7U2]   73:    (reg.r_ax = old;
[NM2]   74:     intr(Øx1Ø,&reg);)
[E32]   75:   return(1);)
[UK6]   76: /* display(ON¦OFF) : Schaltet Bildschirm
[NS1]   77:  an und aus */
[113]   78: void display(mode)
[YH1]   79: int mode;
[MN3]   8Ø:  (struct REGPACK reg;
[VV2]   81:   /* Bildschirm aus */
[PU3]   82:   reg.r_ax = Øx12ØØ¦mode;
[VU4]   83:   reg.r_bx = ØxØØ36; reg.r_cx = 1Ø;
[KU2]   84:   intr(Øx1Ø,&reg); )
[HZ5]   85: /* setp(x,y,f): setzt Bildpunkt in 32Ø
[8M5]   86:  Graphik an Stelle x,y mit Farbe f */
[3G8]   87: static unsigned char far *vidram=(char far
            *)ØxaØØØØØØØ;
```

```
[8V2]   88: void setp(x, y, f)
[DR1]   89: int x,y,f;
[LH3]   9Ø:  (*(vidram+y*32Ø+x)=f;)
[Z86]   91: /* getp(x,y,f): holt farbe Bildpunkt in
[H36]   92:  32Ø-Graphik an Stelle x,y mit Farbe f */
[XW3]   93: unsigned char getp(x, y)
[ZD1]   94: int x,y;
[UF4]   95:  (return(*(vidram+y*32Ø+x));)
[8C6]   96: /* showpal zeigt alle Farben der Palette
[8G6]   97:  an Position x,y auf dem Bildschirm an */
[163]   98: void showpal(xØ,yØ)
[6K1]   99: int xØ,yØ;
[D62]   1ØØ:  (int x,y,col;
[LH3]   1Ø1:  for(col=Ø;col<6;col++)
[EE2]   1Ø2:  for(y=Ø;y<6;y++)
[8E2]   1Ø3:   for(x=Ø;x<6;x++)
[2G5]   1Ø4:    (setp((x<<2)+Ø+xØ,(col<<2)*6+yØ+(y<<2),
[WY1]   1Ø5:     x+6*y+col*36);
[9D5]   1Ø6:     setp((x<<2)+1+xØ,(col<<2)*6+yØ+(y<<2),
[WY1]   1Ø7:     x+6*y+col*36);
[395]   1Ø8:     setp((x<<2)+Ø+xØ,(col<<2)*6+yØ+(y<<2)
[N72]   1Ø9:     +1,x+6*y+col*36);
[595]   11Ø:     setp((x<<2)+1+xØ,(col<<2)*6+yØ+(y<<2)
[N72]   111:     +1,x+6*y+col*36);
[795]   112:     setp((x<<2)+2+xØ,(col<<2)*6+yØ+(y<<2)
[K72]   113:     +Ø,x+6*y+col*36);
[795]   114:     setp((x<<2)+2+xØ,(col<<2)*6+yØ+(y<<2)
[N72]   115:     +1,x+6*y+col*36);
[H65]   116:     setp((x<<2)+Ø+xØ,(col<<2)*6+yØ+(y<<2
[3D2]   117:      )+2,x+6*y+col*36);
[595]   118:     setp((x<<2)+1+xØ,(col<<2)*6+yØ+(y<<2)
[S72]   119:     +2,x+6*y+col*36);
[795]   12Ø:     setp((x<<2)+2+xØ,(col<<2)*6+yØ+(y<<2)
[9Y2]   121:     +2,x+6*y+col*36); ))
[AN6]   122: /* drawline: zeichnet Line zwischen zwei
[K83]   123:  Punkten in Farbe f */
[ØN3]   124: drawline(xØ,yØ,x1,y1,f)
[PC4]   125: register int xØ,yØ,x1,y1,f;
[GG3]   126:  (int dx,dy,ix=1,iy=1;
[2T2]   127:   register int xf;
[CC3]   128:   dx = x1-xØ; dy = y1-yØ;
[N41]   129:   if(dx<Ø)
[C93]   13Ø:     (dx = -dx; ix = -ix;)
[R41]   131:   if(dy<Ø)
[S93]   132:     (dy = -dy; iy = -iy;)
[7B1]   133:   if(dx<dy)
[471]   134:     xf = -dx;
[ZRØ]   135:   else
[671]   136:     xf = dx;
[DG2]   137:   x1 +=ix; y1 +=iy;
[8B3]   138:   while(xØ!=x1 && yØ!=y1)
[1K2]   139:    (setp(xØ,yØ,f);
[K81]   14Ø:     if(xf<=Ø)
[833]   141:      (yØ += iy; xf += dx;)
[ZRØ]   142:     else
[DY3]   143:      (xf -= dy; xØ += ix; )))
[5T4]   144: /* drawbox: zeichnet Rechteck
[4W5]   145:  fill = Ø; nur Rand; 1 = ausgefüllt */
[P45]   146: void drawbox(xØ,yØ,x1,y1,f,fill)
[PP3]   147: int xØ,yØ,x1,y1,f,fill;
[8F1]   148:  (int i,j;
[UA1]   149:   if(!fill)
[RØ4]   15Ø:    (drawline(xØ,yØ,x1,yØ,f);
[ZP3]   151:     drawline(x1,yØ,x1,y1,f);
[ØR3]   152:     drawline(x1,y1,xØ,y1,f);
[NY3]   153:     drawline(xØ,y1,xØ,yØ,f);)
[ZRØ]   154:   else
[753]   155:    (for(i=xØ;i<=x1;i++)
[Z13]   156:      for(j=yØ;j<=y1;j++)
[ER2]   157:       setp(i,j,f); ))
[KØ7]   158: /* word interpret_col(col): interpretiert
[F16]   159:  RGB-Wert in ein Palettenregister */
[N13]   16Ø: #define BREITE 32767
[5F2]   161: #define WINK Ø.5
[2W4]   162: unsigned interpret_col(col)
[FU1]   163: color *col;
[VT5]   164:  (double r=col->r,g=col->g,b=col->b;
[R94]   165:   double rf=Ø.Ø,gf=Ø.Ø,bf=Ø.Ø,h;
[Ø34]   166:   double rs=Ø.Ø,gs=Ø.Ø,bs=Ø.Ø;
[ØD5]   167:   if(col->r<Ø¦¦col->g<Ø¦¦col->b<Ø)
[TL1]   168:    return(Ø);
[ML4]   169:   h=WINK*sqrt((r*r+g*g+b*b)/3.Ø);
[E21]   17Ø:   if(r>h)
[3PØ]   171:     rf++;
[Y11]   172:   if(b>h)
[8SØ]   173:     bf ++;
[3Z1]   174:   if(g>h)
[KSØ]   175:     gf ++;
[EXØ]   176:   r=5*r;
[2WØ]   177:   g=5*g;
[EVØ]   178:   b=5*b;
[MØ3]   179:     if(rs¦¦gs¦¦bs)
[P7Ø]   18Ø:        (
[NP1]   181:         if(r+rf<6.Ø)
[G91]   182:          r=r+rf;
[9N1]   183:         if(g+gf<6.Ø)
[S61]   184:          g=g+gf;
[NM1]   185:         if(b+bf<6.Ø)
[D91]   186:          b=b+bf;)
[R75]   187:    return((unsigned)(r)+(unsigned)
[YH3]   188:     (g)*6+(unsigned)(b)*36);)
[L45]   189: /* Löscht Bildschirm mit Farbe f
[HS4]   19Ø:    Vordergrund rf Rahmenfarbe */
[S52]   191: void cls(f,rf)
[WØ4]   192: register unsigned f,rf;
[E74]   193:  (register unsigned int i;
[R73]   194:   struct REGPACK reg;
[KU2]   195:   for(i=Ø;i<64ØØØ;i++)
[B42]   196:     *(vidram+i)=f;
[7P2]   197:   reg.r_ax = Øx1ØØ1;
[6Z3]   198:   reg.r_bx = ØxØØØØ¦(rf<<8);
[NM2]   199:   intr(Øx1Ø,&reg);)
[TZ5]   2ØØ: /* blink(col): bildschirm blinkt col */
[733]   2Ø1: void blink(int col)
[MN3]   2Ø2: (struct REGPACK reg;
[LG5]   2Ø3:  reg.r_ax = Øx1Ø1Ø; reg.r_dx=col<<8;
[NE5]   2Ø4:  reg.r_cx = col&ØxffØØ; reg.r_bx = Ø;
[VC2]   2Ø5:  intr(Øx1Ø,&reg);
[4R4]   2Ø6:  reg.r_ax = Øx1Ø1Ø; reg.r_dx= Ø;
[G54]   2Ø7:  reg.r_cx = Ø; reg.r_bx = Ø;
[KU2]   2Ø8:  intr(Øx1Ø,&reg); )
[U26]   2Ø9: /* mkfr(): zeichnet einfachen Rahmen */
[A54]   21Ø: mkfr(xØ,yØ,x1,y1,col1,col2)
[F44]   211: int xØ,yØ,x1,y1,col1,col2;
[5E4]   212:  (drawline(xØ,yØ,x1,yØ,col1);
[975]   213:   drawline(xØ-1,yØ-1,x1+1,yØ-1,col1);
[L75]   214:   drawline(xØ-2,yØ-2,x1+2,yØ-2,col1);
[X75]   215:   drawline(xØ-3,yØ-3,x1+3,yØ-3,col1);
[2A4]   216:   drawline(xØ,yØ,xØ,y1,col1);
[875]   217:   drawline(xØ-1,yØ-1,xØ-1,y1+1,col1);
[K75]   218:   drawline(xØ-2,yØ-2,xØ-2,y1+2,col1);
[W75]   219:   drawline(xØ-3,yØ-3,xØ-3,y1+3,col1);
[AA4]   22Ø:   drawline(x1,yØ,x1,y1,col2);
[875]   221:   drawline(x1+1,yØ-1,x1+1,y1+1,col2);
[K75]   222:   drawline(x1+2,yØ-2,x1+2,y1+2,col2);
[W75]   223:   drawline(x1+3,yØ-3,x1+3,y1+3,col2);
[9A4]   224:   drawline(xØ,y1,x1,y1,col2);
[375]   225:   drawline(xØ-1,y1+1,x1+1,y1+1,col2);
[D75]   226:   drawline(xØ-2,y1+2,x1+2,y1+2,col2);
[MB5]   227:   drawline(xØ-3,y1+3,x1+3,y1+3,col2);)
[NZØ]   228: #endif
```

Listing 7. In »graph.h« sind die Grafikfunktionen für das Spiel definiert.

Textur-Generator

Ein Malprogramm für Maskavan

Um wirkungsvolle Grafiken für Wände und Objekte der eigenen dreidimensionalen Welten zu entwerfen, finden Sie hier ein Malprogramm mit fünf Grundfunktionen für Texturen mit 256 Farben.

Nicht nur Programmierer brauchen Werkzeuge, sondern auch die Künstler eines Spiels. Das folgende Malprogramm bietet Ihnen eine Farbpalette an, mit deren Hilfe Sie nahezu alles malen können. Einer der großen Vorteile dieses Malprogramms ist zweifellos, daß Sie die fertigen Bilder als Texturen beim Rollenspiel »Das Geheimnis von Maskavan« verwenden können. Möchten Sie beispielsweise einen anderen Hintergrund für das Spiel schaffen, laden Sie einfach die entsprechende Textur in das Malprogramm ein und verändern sie nach eigenen Wünschen. So wird in Kürze aus einer finsteren Burg ein futuristisches Raumschiff.

Das Malprogramm starten Sie durch Eingabe von »maker«. Es präsentiert sich eine pseudo-dreidimensionale Oberfläche. Fertige Bilder laden Sie mit dem Befehl »load«. Dazu müssen Sie den Namen der gewünschten Datei wissen. Eine Dateiauswahlbox ist bei der Kürze des Listings nicht enthalten. Die fünf grundlegenden Malfunktionen im einzelnen: Mit »draw« setzen Sie im Freihandmodus Punkte.

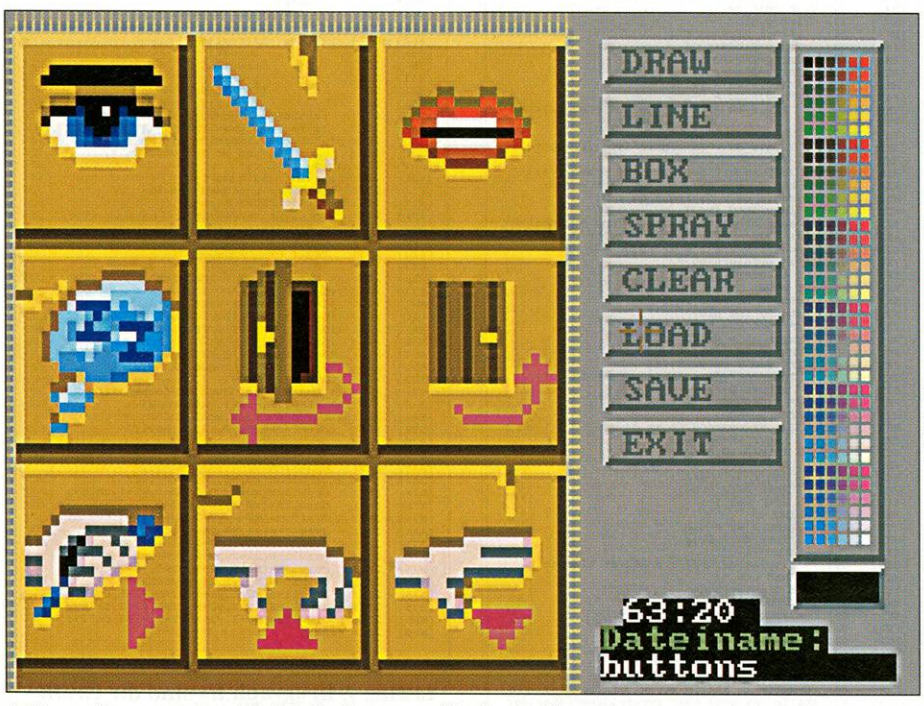
Texturen mit 256 Farben schnell selbst gestaltet

Dazu wählen Sie zuerst eine der 256 Farben, die am rechten Bildschirmrand dargestellt werden, und setzen dann mit der linken Maustaste Ihre Punkte. Die Funktion »line« zeichnet Linien für Sie, erfordert allerdings ein wenig Übung, da die Linien nicht als Gummiband dargestellt werden. Sie geben nur Start- und Endpunkt an und schätzen den Linienverlauf ungefähr ab. Ähnlich gestaltet sich auch die Erzeugung von gefüllten Boxen mit der Box-Funktion. Mit der Spray-Funktion verteilen Sie Punkte nach einem vorgegebenen Verteilungsschema über die Grafik. Für die relativ grobe Auflösung der Texturen reicht diese Art der Funktions-Implementierung zum schnellen Zeichnen völlig aus.

Für die erfolgreiche Kompilierung von »maker« benötigen Sie folgende Listings aus dieser Ausgabe: »mouse.c«, »schalter.c«, »maker.c« (Listing und Bild) und »graph.h«.

Kopieren Sie alle Teilprogramme in ein Verzeichnis und kompilieren Sie »maker« im Modell »Huge« des Turbo-C-Compilers. Für eine Verwendung der Bilder unabhängig vom Rollenspiel steht Ihnen mit dem Listing »load.c« eine Funktionsbibliothek zur Verfügung, die es erlaubt, Bilder wahlweise auf dem Bildschirm oder in den Hauptspeicher zu laden. Das Listing zu »load.c« finden Sie im Beitrag »Das Geheimnis von Maskavan«.

(Matthias Oberdorfer/bm)



Mit »maker.c« malen Sie leicht Texturen für das Rollenspiel.

```
[W6Ø]    1: /*
[BX2]    2:  Programm: maker.c
[KR3]    3:  Funktion: Malprogramm
[K83]    4:  Sprache:  Turbo C 2.Ø
[KR4]    5:  Autor:    Matthias Oberdorfer
[HN4]    6:  (c) 1992 DMV Widuch GmbH & Co. KG
[ØØØ]    7:
[1R3]    8: model huge verwenden */
[842]    9: #ifndef MAKER
[E42]   1Ø: #define MAKER
[KU2]   11: #include "graph.h"
[2W2]   12: #include "mouse.c"
[LA3]   13: #include "schalter.c"
[PS4]   14: void spray(int x,int y,int f);
[7G4]   15: void makeframe(int x,int y);
[E95]   16: int setpunkt(int x,int y,int f);
[3E5]   17: unsigned getpunkt(int x,int y);
[3Y6]   18: void dl(int xØ,int yØ,int x1,int y1,int f);
[MH6]   19: void db(int xØ,int yØ,int x1,int y1,int f
[ØU1]   2Ø: ,int fill);
[TT3]   21: int save(); int load();
[ØW1]   22: /* Buttons */
[KF5]   23: schalter EX =( /*11*/Ø,21Ø,118,27Ø,128,
[1C3]   24: 56,2ØØ,215,Ø,Ø,1,"EXIT",);
[MS5]   25: schalter SV =( /*1Ø*/&EX,21Ø,1Ø2,27Ø,112,
[FC3]   26: 56,2ØØ,215,Ø,Ø,1,"SAVE",);
[RU3]   27: schalter LD =( /*9*/&SV,
[665]   28: 21Ø,86,27Ø,96,56,2ØØ,215,Ø,Ø,1,"LOAD",);
[9C5]   29: schalter CL =( /*8*/&LD,21Ø,7Ø,27Ø,8Ø,
[KR3]   3Ø: 56,2ØØ,215,Ø,Ø,1,"CLEAR",);
[926]   31: schalter SPR =( /*7*/&CL,21Ø,54,27Ø,64,56,
[KA3]   32: 2ØØ,215,Ø,Ø,1,"SPRAY",);
[MX5]   33: schalter BOX =( /*6*/&SPR,21Ø,38,27Ø,48,
[S13]   34: 56,2ØØ,215,Ø,Ø,1,"BOX",);
[6Y5]   35: schalter LI =( /*5*/&BOX,21Ø,22,27Ø,32,56,
[E63]   36: 2ØØ,215,Ø,Ø,1,"LINE",);
[795]   37: schalter DR =( /*4*/&LI,21Ø,6,27Ø,16,
[GB3]   38: 56,2ØØ,215,Ø,Ø,1,"DRAW",);
[VC6]   39: schalter ZB =( /*Zeichenbrett 3*/&DR,4,4,
[6L3]   4Ø: 195,195,56,Ø,215,Ø,Ø,Ø,"",);
[HS5]   41: schalter PAL =( /*2*/&ZB,276,6,3Ø6,156,
[CK2]   42: 56,Ø,215,Ø,Ø,Ø,"",);
[GK5]   43: schalter ZF =(  /* nr 1*/ &PAL,  276,16Ø,
[TK3]   44: 3Ø6,17Ø,56,Ø,215, Ø,Ø,Ø,"", );
[BV3]   45: #define ZEICHENFARBE 1ØØ
[K46]   46: /* makeframe (): Zeichnet Gittermu-ter
[LP2]   47: auf den Schirm */
[463]   48: void makeframe(x,y)
[ZD1]   49: int x,y;
[4B1]   5Ø:  {int i;
[MU4]   51:   drawbox(x+3,y+3,x+3*64+4,y+3*64+4,
[3P2]   52:   DURCHSICHTIG,1);
[MU4]   53:   drawbox(x+3,y+3,x+3*64+4,y+3*64+4,
[LK2]   54:   ZEICHENFARBE,Ø);
[PP2]   55:   for(i=Ø;i<=64;i++)
[KØ5]   56:   (drawline(x,3*i+y+4,2+x,3*i+y+4,
[FC2]   57:    ZEICHENFARBE);
[YT3]   58:    drawline(x+4+3*64,3*i+y+4,
[VP4]   59:    4+3*64+4+x,3*i+y+4,ZEICHENFARBE);
[4A4]   6Ø:    drawline(3*i+x+4,y,3*i+x+4,y+3
[DK2]   61:    ,ZEICHENFARBE);
[XU4]   62:    drawline(3*i+x+4,y+4+3*64,3*i+x+4,
[BH4]   63:    y+4+3+3*64,ZEICHENFARBE);}))
[WY6]   64: /* setpunkt(): setzt Punkt in Rasterbreite
[8H3]   65: des Zeichenbrettes */
[4A3]   66: int setpunkt(x,y,f)
[DR1]   67: int x,y,f;
[8F1]   68:  {int i,j;
[SK1]   69:   x -=4; y -=4;
[AM1]   7Ø:    if(x<Ø) x=Ø;
[V62]   71:    if(x>=192) x=191;
[DM1]   72:    if(y<Ø) y=Ø;
[X62]   73:    if(y>=192) y=191;
[ØU2]   74:    x =x-x%3; y =y-y%3;
[2B2]   75:    for(i=Ø;i<3;i++)
[8B2]   76:     for(j=Ø;j<3;j++)
[H93]   77:      setp(4+x+i,4+y+j,f); }
[RL4]   78: /*  getpunkt(): Holt Farbe vom
[SD2]   79: Rasterbrett */
[HV3]   8Ø: unsigned getpunkt(x,y)
[ZD1]   81: int x,y;
[8F3]   82:  (return(getp(x,y));)
[HB5]   83: /* drawline: für Zeichenbrett */
[XB3]   84: void dl(xØ,yØ,xl,yl,f)
[PC4]   85: register int xØ,yØ,xl,yl,f;
[GG3]   86:  (int dx,dy,ix=1,iy=1;
[2T2]   87:   register int xf;
[CC3]   88:   dx = xl-xØ; dy = yl-yØ;
[N41]   89:   if(dx<Ø)
[S43]   9Ø:     (dx = -dx;ix = -ix;)
[R41]   91:   if(dy<Ø)
[253]   92:     (dy = -dy;iy = -iy;)
[DW2]   93:   if(dx<dy) xf = -dx;
```

```
[R12]   94:    else xf = dx;
[DG2]   95:    xl +=ix; yl +=iy;
[8B3]   96:    while(xØ!=xl && yØ!=yl)
[BB3]   97:     (setpunkt(xØ,yØ,f);
[K81]   98:      if(xf<=Ø)
[833]   99:       (yØ += iy; xf += dx;)
[ZRØ]  1ØØ:      else
[AØ4]  1Ø1:       (xf -= dy; xØ += ix;)))
[G87]  1Ø2: /* db : für Zeichenbrett: zeichnet Rechteck
[7K5]  1Ø3:  fill = Ø; nur rand. 1; ausgefüllt */
[Z64]  1Ø4: void db(xØ,yØ,xl,yl,f,fill)
[PP3]  1Ø5: int xØ,yØ,xl,yl,f,fill;
[8F1]  1Ø6:  (int i,j;
[UA1]  1Ø7:   if(!fill)
[6N5]  1Ø8:    (dl(xØ,yØ,xl,yØ,f); dl(xl,yØ,xl,yl,f);
[VL5]  1Ø9:     dl(xl,yl,xØ,yl,f); dl(xØ,yl,xØ,yØ,f);)
[ZRØ]  11Ø:   else
[853]  111:    (for(i=xØ;i<xl;i+=3)
[243]  112:      for(j=yØ;j<yl;j+=3)
[863]  113:       setpunkt(i,j,f);))
[ØF2]  114: #define RANGE 1Ø
[MT4]  115: /* Spray: Sprühdoseneffekt */
[WR2]  116: void spray(x,y,f)
[DR1]  117: int x,y,f;
[3K5]  118:  (long zx = (long)rand()*RANGE/32768;
[ME5]  119:   long zy = (long)rand()*RANGE/32768;
[XD5]  12Ø:   setpunkt((int)(x+RANGE/2-zx),(int)
[223]  121:     (y+RANGE/2-zy),f);)
[2D6]  122: /* save(name): Speichert Bild unter name
[7U3]  123: ab; 1 wenn erfolgreich */
[XN1]  124: int save()
[1L1]  125:  (FILE *fp;
[EN2]  126:   unsigned char c;
[1H2]  127:   char name[1ØØ];
[5B1]  128:   int i,j;
[V42]  129:   gotoxy(67,23);
[TH3]  13Ø:   print("Dateiname:",56);
[W42]  131:   gotoxy(67,24);
[5V2]  132:   print("          ",56);
[W42]  133:   gotoxy(67,24);
[852]  134:   getstr(name);
[8W2]  135:   /* Datei öffnen */
[7Z2]  136:   fp=fopen(name,"wb");
[521]  137:   if(!fp)
[TL1]  138:    return(Ø);
[ØG2]  139:   for(j=Ø;j<64;j++)
[TF2]  14Ø:    for(i=Ø;i<64;i++)
[5X3]  141:     (c=getpunkt(i*3+4,j*3+4);
[HH4]  142:      fwrite(&c,sizeof(c),1,fp);)
[4Y3]  143:   fclose(fp); return(1);)
[DS5]  144: /* load(name): lädt Bild mit name ein
[XØ5]  145: liefert 1 falls erfolgreich */
[1H2]  146: char name[1ØØ];
[4M1]  147: int load()
[1L1]  148:  (FILE *fp;
[2A1]  149:   int i, j;
[EN2]  15Ø:   unsigned char c;
[V42]  151:   gotoxy(67,23);
[TH3]  152:   print("Dateiname:",56);
[W42]  153:   gotoxy(67,24);
[5V2]  154:   print("          ",56);
[W42]  155:   gotoxy(67,24);
[852]  156:   getstr(name);
```

```
[EP2]  157:   /*Datei öffnen */
[2Z2]  158:   fp=fopen(name,"rb");
[521]  159:   if(!fp)
[TL1]  16Ø:    return(Ø);
[ØG2]  161:   for(j=Ø;j<64;j++)
[TF2]  162:     for(i=Ø;i<64;i++)
[724]  163:      (fread(&c,sizeof(c),1,fp);
[2R3]  164:       setpunkt(i*3+4,j*3+4,c);)
[4Y3]  165:   fclose(fp); return(1);)
[UX6]  166: /* Zeilennummer (x,y): Schreibt aktuelle
[DM5]  167: Mauposition auf den Bildschirm */
[EW2]  168: Zeilennummer(x,y)
[ZD1]  169: int x,y;
[F43]  17Ø:  (x=(x-4)/3; y=(y-4)/3;
[W42]  171:   gotoxy(68,22);
[LH3]  172:   if(x>=Ø&&y>=Ø&&x<64&&y<64)
[DD2]  173:    (printnum(x);
[NS3]  174:     print(":",ZEICHENFARBE);
[N32]  175:     printnum(y);
[CB4]  176:     print(" ",ZEICHENFARBE);))
[UWØ]  177: main()
[LW5]  178:  (unsigned x,y,xalt,yalt, fertig=Ø;
[EX2]  179:  int ART = Ø,farbe;
[P94]  18Ø:  int f=255; /* Zeichenfarbe */
[X91]  181:  color c;
[R84]  182:  schalter *first = &ZF,*sch;
[X13]  183:  if(!setgraphik(ON))
[961]  184:    exit(Ø);
[465]  185:  /* Zufallsgenerator anwerfen */
[7Y1]  186:  randomize();
[P34]  187:  /* Farbe initialisieren */
[F23]  188:  c.r=.5;c.b=.5; c.g=.5;
[W34]  189:  farbe=interpret_col(&c);
[7T1]  19Ø:  sch=first;
[KM1]  191:  while(sch)
[7V3]  192:   (sch->framecol = farbe;
[UB3]  193:    sch->textcol=farbe;
[CW2]  194:    sch = sch->next;)
[3B3]  195:  c.r=.8; c.b=.8; c.g=.8;
[W34]  196:  farbe=interpret_col(&c);
[7T1]  197:  sch=first;
[KM1]  198:  while(sch)
[RY3]  199:   (sch->framecol2 = farbe;
[CW2]  2ØØ:    sch = sch->next;)
[ØB3]  2Ø1:  c.r=.6; c.b=.6;c.g=Ø.6;
[7T1]  2Ø2:  sch=first;
[W34]  2Ø3:  farbe=interpret_col(&c);
[KM1]  2Ø4:  while(sch)
[KY2]  2Ø5:   (sch->col = farbe;
[CW2]  2Ø6:    sch = sch->next;)
[RX6]  2Ø7:  cls(interpret_col(&c),interpret_col(&c));
[3E3]  2Ø8:  c.r=Ø.Ø; c.b=Ø.Ø; c.g=Ø.Ø;
[LD1]  2Ø9:  ZF.col=f;
[3E3]  21Ø:  /* Bildschirm aufbau */
[WP4]  211:  showsch(&ZF); showpal(28Ø,1Ø);
[PA2]  212:  makeframe(Ø,Ø);
[9E6]  213:  mouse_def(pointer,16,6); mouse_show();
[DB4]  214:  do( while(!mouse_press())
[M13]  215:      (mouse_pos(&x,&y);
[ØB1]  216:       x=x>>1;
[LH3]  217:       Zeilennummer(x,y);)
[M62]  218:   mouse_hide();
[U35]  219:   switch(testsch(x,y,first,&sch))
```

```
[281]  22Ø:    (case 2:
[E24]  221:       f=getp(x&Øxfffe,y&Øxfffe);
[222]  222:       ZF.col=f;
[4K4]  223:       drawbox(ZF.left+1,ZF.top+1,
[TS5]  224:       ZF.right,ZF.bottom,ZF.col,1); break;
[HZ3]  225:      case 3: /* Zeichen Brett */
[6X1]  226:       switch(ART)
[U71]  227:        (case Ø:
[FR2]  228:           xalt=x; yalt=y;
[TB3]  229:           while(mouse_press())
[M13]  23Ø:            (mouse_pos(&x,&y);
[ØB1]  231:             x=x>>1;
[Z13]  232:             Zeilennummer(x,y);
[L53]  233:             dl(x,y,xalt,yalt,f);
[CV2]  234:             xalt=x; yalt=y;)
[N41]  235:         case 1:
[FR2]  236:           xalt =x; yalt =y;
[BA2]  237:           mouse_show();
[TB3]  238:           while(mouse_press())
[M13]  239:            (mouse_pos(&x,&y);
[ØB1]  24Ø:             x=x>>1;
[LH3]  241:             Zeilennummer(x,y);)
[M62]  242:           mouse_hide();
[ZT2]  243:           mouse_pos(&x,&y);
[ØB1]  244:           x=x>>1;
[L53]  245:           dl(x,y,xalt,yalt,f);
[PØ1]  246:           break;
[S41]  247:         case 2:
[FR2]  248:           xalt =x; yalt =y;
[BA2]  249:           mouse_show();
[TB3]  25Ø:           while(mouse_press())
[M13]  251:            (mouse_pos(&x,&y);
[ØB1]  252:             x=x>>1;
[LH3]  253:             Zeilennummer(x,y);)
[M62]  254:           mouse_hide();
[524]  255:           mouse_pos(&x,&y); x=x>>1;
[KC4]  256:           db(xalt,yalt,x,y,f,1); break;
[V41]  257:         case 3:
[TB3]  258:           while(mouse_press())
[M13]  259:            (mouse_pos(&x,&y);
[HK2]  26Ø:             x=x>>1;delay(1Ø);
[Z13]  261:             Zeilennummer(x,y);
[AX3]  262:             spray(x,y,f);) break;)
[PØ1]  263:       break;
[KE3]  264:      case 4: ART =Ø; break;
[SE3]  265:      case 5: ART =1; break;
[YE3]  266:      case 6: ART =2; break;
[4F3]  267:      case 7: ART =3; break;
[A51]  268:      case 8:
[WK3]  269:       drawbox(4,4,3*64+4,3*64+4,
[5X3]  27Ø:       DURCHSICHTIG,1); break;
[7C3]  271:      case 9: load(); break;
[XF3]  272:      case 1Ø: save(); break;
[344]  273:      case 11: fertig=1; break;)
[BA2]  274:     mouse_show();
[DP2]  275:    )while(!fertig);
[MW2]  276:  setgraphik(OFF);)
[NZØ]  277: #endif
```

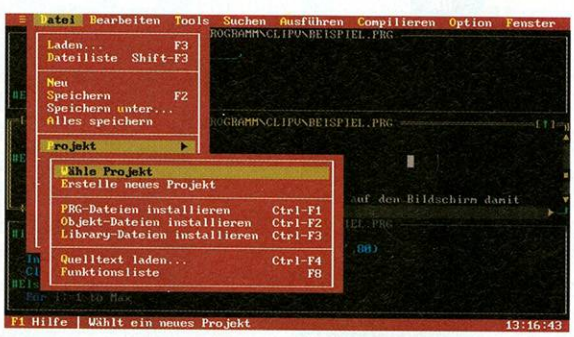
Neue Oberflächenstrukturen entwerfen Sie mit »maker.c«.

NEU

Clip and Vision 1.0

Die integrierte Entwicklungsumgebung für Clipper 5.0x

Clip and Vision 1.0 ist die maßgeschneiderte Entwicklungsumgebung, auf die Sie als Clipper-5.0x-Programmierer lange gewartet haben. ***Clip and Vision 1.0*** besticht mit einem SAA-konformen Editor mit modernster Mehrfenstertechnik und Mausbedienung. Zahlreiche Funkionen für die Projektverwaltung halten Ihnen den Kopf frei für das Wesentliche. Direkt aus ***Clip and Vision 1.0*** heraus compilieren, linken, starten und debuggen Sie Ihre Programme und erhalten damit eine erhebliche Zeitersparnis.



- SAA-Oberfläche mit modernster Menü- und Fenstertechnik
- Maussteuerung mit 'Mausbox'-Spezialmenüs
- Mehrfenster-Editor zur gleichzeitigen Bearbeitung verschiedener Quelltexte
- Quelltextformatierung und Templates für einheitliche, übersichtliche Listings
- Compilieren, linken und debuggen direkt aus der Oberfläche heraus
- Projektverwaltung für bis zu 20 PRG-, OBJ- und LIB-Dateien
- integrierter Datenbank-Browser
- kontextsensitive Online-Hilfe
- EGA-/VGA-Support
- ASCII-Tabelle für Sonderzeicheneingabe

CeBIT'93 HANNOVER 24. — 31.03.1993 Halle 7, Stand E13/E19

Systemanforderungen: DOS ab V. 3.2, Clipper 5.0 oder 5.01, Festplatte

Clip and Vision 1.0 DM 199,–*

Update Clip and Vision DM 99,–*

DMV-ProfiTools für Clipper DM 199,–*

* Unabhängig von der Anzahl der bestellten Produkte berechnen wir für das Inland DM 4,– bzw. für das Ausland DM 6,– Porto und Verpackung.

Bitte benutzen Sie die Bestellkarte.
Händleranfragen erwünscht

DMV-Verlag · Postfach 250 · W-3440 Eschwege
Tel.: (0 56 51) 8 09-0 · Fax: (0 56 51) 8 09-4 30

DMV Daten- und Medienverlag

Das Geheimnis von Maskavan - die Story

Du bist der Sohn eines verarmten, adligen Barons in der Zeit des grauen Mittelalters. Dir steht der Kopf mehr nach handfesten Abenteuern, als den Bauern den Zehnten abzufordern, wie es Dein Vater gewohnt ist zu tun. Zur Hebung seines Ansehen gibt Dein Vater einen großen Ball auf eurem Schloss. Alles was Rang und Namen hat kommt und feiert. Auch Mylady Miriam von Maskavan, eines der schönsten Mädchen im Königreich Legendy, ist an diesem Abend anwesend. Du hattest dieses bezaubernde Wesen schon früher einmal gesehen, aber es nicht gewagt sie anzusprechen. Denn ihr Vater, der Graf von Maskavan, ein einflußreicher und konservativer Mann, sieht es nicht so gern, wenn sich seine Tochter mit jungen Männern unterhält. Deshalb wohnt er mit ihr in einer weit abgelegenen Provinz auf dem geheimnisvollen Schloss Maskavan, von dem es heißt, daß es mehr Geheimgänge und geheime Zimmer hat, als normale Gänge und Zimmer. An diesem Abend geschieht etwas, was Dich völlig überrascht: Dieses wunderschöne Mädchen schaut Dich unablässig an und kommt schliesslich über ein paar Umwegen auf Dich zu. Dir bleibt fast das Herz stehen, als Du in ihre tiefblauen Augen blickst. Nach einer atemlosen kurzen Stille ergreift Sie das Wort und spricht Dich an. Obwohl Du vor Aufregung kaum Luft holen kannst, ist Dir klar, daß es jetzt von Dir abhängt, wie dieser Abend weitergeht...

Simulation einer Welt

Das Geheimnis von Maskavan

Rollenspiele und Simulationen anderer Welten und Zeiten erfreuen sich immer größerer Beliebtheit. »Das Geheimnis von Maskavan« verknüpft die Komponenten Action, Simulation und Rollenspiel.

Die Computerspiele der neuesten Generation gehen immer mehr dazu über, den Spieler in eine virtuelle, fantastische Welt eintauchen zu lassen. Unterstützt von digitalisiertem Sound und schneller Echtzeit-3D-Grafik bewegen Sie sich beispielsweise über fremde Planeten in einer unbekannten Zeit.

Weil in jedem guten Computerspiel sowohl Action und Strategie als auch Simulation geboten werden, kann man kaum noch ein Spiel einem bestimmten Genre zuordnen. Sicher gibt es Ausnahmen, wie zum Beispiel Tetris, das durch seine Einfachheit und überschaubaren Spielregeln überzeugt.

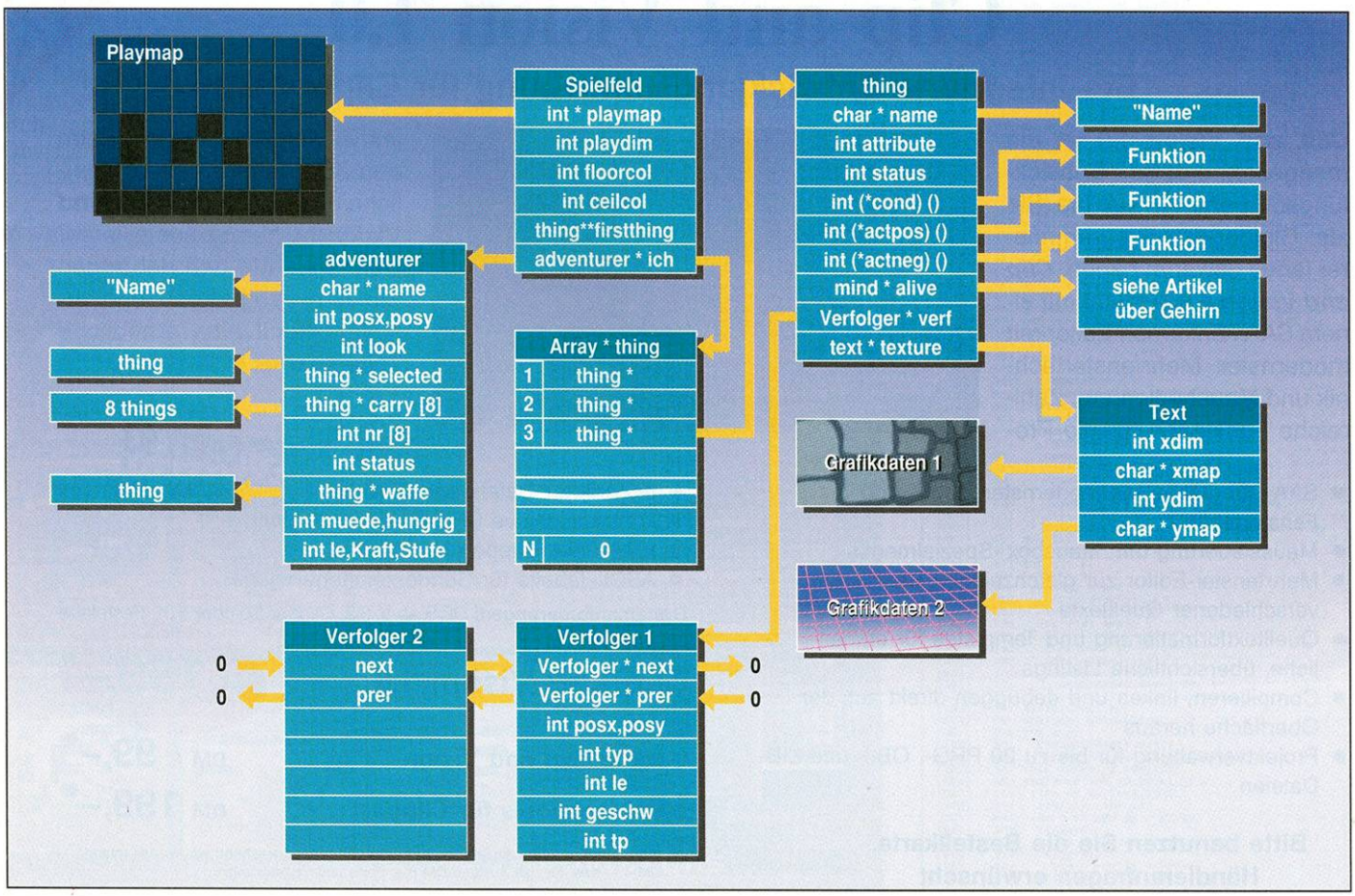


Bild 1. Die Zusammenhänge der Strukturen des Spiels

Aber der Trend hin zur Simulation einer virtuellen Welt ist deutlich zu erkennen und läßt auch eine Prognose über das Aussehen zukünftiger Computerspiele zu: Mehr Rechnerleistung und neue Eingabegeräte wie beispielsweise Cyberspace-Handschuhe werden zu einer immer perfekteren Nachbildung einer anderen Welt führen.

Nicht die technische Simulation eines Flugsimulators, sondern die Atmosphäre und Authentizität gehören bei dieser Art von Spielen zu den wichtigsten Faktoren. So ist auch »Das Geheimnis von Maskavan« eine Kombination aus Simulation, Action und Denkspiel (Bilder 2 bis 5).

Rollenspiele auf Computern

Hinter dem klassischen Rollenspiel ohne Computer steht vereinfacht gesagt die Idee, sich einen Abenteuerfilm anzuschauen und dabei die Rolle des Helden aktiv zu übernehmen. Mehrere Spieler bewegen sich durch eine nur in der Vorstellung existierende Welt mit eigenen Gesetzen und versuchen, eine bestimmte Aufgabe zu lösen. Auf dem Weg dahin stehen den Abenteurern viele Hindernisse im Weg. So gibt es je nach Weltbild und Zeit des Spiels verschiedene Rätsel, Strategien und Kämpfe zu absolvieren, um schließlich ein Abenteuer erfolgreich zu bestehen.

Die Hauptaufgabe des einzelnen Spielers ist dabei das Agieren in der von ihm angenommenen Rolle. Diese wird durch einen bestimmten Abenteurertyp festgelegt. Da ein Kämpfer beispielsweise andere Wertvorstellungen hat als ein Zwerg oder ein Druide, verhält er sich auch in bestimmten Situationen anders. Ein Spieler muß daher immer gemäß seiner Rolle reagieren. Mit dem Bestehen mehrerer Abenteuer sammelt er Erfahrungen und Reichtum. Dies drückt sich im Spiel dadurch aus, daß er mit mehr Geld bessere Ausrüstungen kaufen und mit mehr Erfahrung schwierigere Aufgaben lösen kann. Je nach Fantasie und schauspielerischem Talent entsteht eine realitätsnahe Atmosphäre im Spiel, die den eigentlichen Spielreiz erzeugt. Überwacht und gesteuert werden diese Abenteuer von einem menschlichen Spielleiter, der die Funktion des Erzählers, des Bösewichts und des Richters in dieser Fantasiewelt übernimmt.

Die Umsetzung von Rollenspielen auf Computern führte erst einmal zwangsweise zu Einschränkungen und Änderungen der Ursprungsidee. Abgesehen von den sogenannten »Multi User Dungeons« (MUDs), die mehreren Spielern die Chance bieten, sich in einer einzigen Computer-Spielewelt zu bewegen, werden die Rollenspiele auf dem PC meist von einer einzigen Person gespielt. Weil der einzige Mitspieler dabei der Computer ist, der die Rolle der Gegner und des Spielleiters einnimmt, bleibt natürlich viel Atmosphäre auf der Strecke. Die Spielehersteller versuchten diesen Nachteil zunächst durch realistische Grafiken und digitalisierten Sound auszugleichen. Der Effekt war zwar teilweise beeindruckend, macht aber noch kein Spiel zum Rollenspiel.

Heutzutage zeichnen sich gute Rollenspiele durch folgende Merkmale aus:

- Im Vordergrund stehen nicht mehr nur der Kampf gegen die Computergegner, sondern vielmehr Aktionen, die den Spieler tatsächlich ein persönliches Verhältnis zu den computergesteuerten Lebewesen entwickeln lassen.
- Die virtuellen Welten weisen solch eine Größe und Komplexität auf, daß der Spieler gar nicht merkt, daß irgendwo ein Ende existieren könnte. So trifft man in Ultima VI alte Freunde aus früheren Abenteuern. Simulation heißt das Zauberwort, das die Atmosphäre und Authentizität erzeugt, die man in frühreren Computer-Rollenspielen vermißte. Aber um Authentizität zu erreichen, genügt nicht nur eine einzige Simulation. So wird alles simuliert – von den Lebewesen über die Geräusche bis hin zu den Grafiken, aber auch die Reaktionen der Bösewichte.

Auf diese Merkmale wurde auch bei der Entwicklung von »Das Geheimnis von Maskavan« besonderer Wert gelegt. Durch das Zusammenspiel verschiedener Simulationen wird die Welt im Computer lebendig. So ist es Ihnen zum Beispiel gestattet, sich mit den künstlichen Lebewesen in natürlicher Sprache zu unterhalten oder in Echtzeit durch die Fantasiewelt zu wandern. Solcher Aufwand an Grafik und Programm verbirgt einen Umfang, der normalerweise mehrere Disketten füllt und kaum noch in einer Zeitschrift zu veröffentlichen ist. Dennoch ist es beim »Geheimnis von Maskavan« gelungen, durch extrem effektiven Programmcode und viele Tricks bei den Grafiken ein Spiel zu entwickeln, das in dieser Richtung neue Maßstäbe setzt.

Das Spiel setzt sich aus den Listings im Kasten »Listings für das Rollenspiel«, weiteren 23 Texturen und 15 »Büchern« zusammen (Listings 1 bis 43), die als Hexdumps abgedruckt sind. Tippen Sie zuerst den Entpacker »unpack.c« (Listing 1) ab. Mit ihm entpacken Sie dann die Hexdumps (Syntax: »unpack packed file«). Die ausgepackten Dateien dürfen keine Endungen mehr haben. Aus »buch.pak« machen Sie also »buch«. Um die Exe-Datei zu erzeugen, kopieren Sie alle Dateien in ein Verzeichnis, bilden aus den C-Quelltexten eine Projektdatei und kompilieren im Modell »Huge« des Turbo-C-Compilers. Die Grafikbibliothek von Turbo C muß ausgeschaltet sein. Ebenso sollten Sie die Warnungen – insbesondere das Prototyping – ausschalten. Das Programm



Bild 2. Wenn Lady Miriam Sie auf einen Festball einlädt, sollten Sie schnell zustimmen.



Bild 3. Dem Grafen entlocken Sie mit den richtigen Fragen einige Informationen.

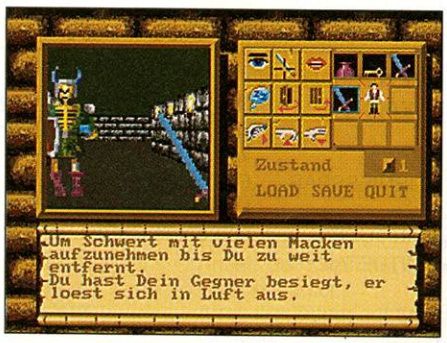


Bild 4. Um die Untoten zu besiegen, brauchen Sie ein gutes Schwert.



Bild 5. In der Bibliothek gibt es einige interessante Bücher.

»mus.c« erfordert eine gesonderte Übersetzung im Modell »Tiny«, wobei der Schalter »Test Stack Overflow« auf »OFF« stehen muß. Starten Sie schließlich alles durch den Aufruf »mus mask«.

Die Simulation einer Welt

Im Spiel »Das Geheimnis von Maskavan« ist eine ganze Reihe von Simulationen enthalten. Abgesehen vom künstlichen Gehirn der Spielfiguren und der 3D-Darstellungsart der Fantasiewelt finden Sie eine Organismus-Simulation für den Spieler. Damit sind die Merkmale des Spiels gemeint, die dem Spieler einen menschlichen Lebensrhythmus verleihen. Das bedeutet, daß der Spieler zum Beispiel hungrig wird und essen muß, müde wird und schlafen muß, die Tage vergehen, es Nacht wird, sein Erfahrungsstand steigt – und daß seine Verletzungen ausheilen, wenn er schläft oder Heiltränke zu sich nimmt. Die aktuellen Werte des Spielers sind in der Struktur »adventurer« gespeichert (Bild 1 und Tabelle 1). Alle Funktionen und Prozeduren dazu finden Sie im Programm »mask.c«. Damit verbunden existiert eine allgemeine Verwaltung der Zustände aller vorkommenden Gegenstände.

Gegenstände des Spiels sind Objekte, die als Strukturen vom Typ »thing« definiert sind. Falls ein Spieler mit einem Objekt eine bestimmte Aktion vorhat, wird dieses Objekt der allgemeinen Prüffunktion »generalcond()« übergeben, die testet, ob die Aktion mit diesem Gegenstand erlaubt ist. Ist sie dies, so wird die allgemeine Aktionsfunktion »generaldo()« aufgerufen, die die gewünschte Aktion mit dem Gegenstand ausführt.

Bei diesem Verfahren ist es auch gestattet, speziellen Gegenständen besondere Prüf- und Aktionsfunktionen mitzugeben, deren Adressen dann in der Struktur des Objektes eingetragen werden.

Beispielsweise kann ein Schlüssel nur eine bestimmte Tür öffnen und braucht deshalb eine Prüffunktion, die das Aufschließen nur bei der richtigen Tür gestattet. Alle Gegenstände werden durch diese Struktur vom Typ »thing« beschrieben.

In Tabelle 2 finden Sie die Attribute und deren Zustände, die Sie jedem Objekt mitgeben können. Falls Sie also keine Lust mehr haben, gegen Monster zu kämpfen, so tragen Sie in der Thing-Struktur der Monster im Feld »attribute« die Worte »GENIESSBAR|BEWEGBAR« ein, und schon ist es Ihnen gestattet, angreifende Monster aufzuessen. Um einen eigenen Gegenstand zu erzeugen und ihn ins Spiel mit aufzunehmen, müssen Sie



lediglich eine neue Thing-Struktur anlegen und den Zeiger in das Array »firstthing« eintragen. Überall, wo Sie nun die dem Gegenstand entsprechende, fortlaufende Array-Nummer in das Spielfeld eintragen, taucht das Objekt im Spiel auf. Dabei ist es noch wichtig, die zusätzlichen Attribute »TRANSPARENT«, »WALL« oder »OBJECT« zu beachten, die der Array-Nummer mitzugeben sind.

– »TRANSPARENT« bewirkt, daß das Objekt zwar da ist, aber nicht gesehen werden kann.

– Mit dem Attribut »OBJECT« stellen Sie ein Objekt dar, das nicht die Form eines großen Würfels besitzt (alle kleineren Objekte). Lassen Sie dieses Attribut weg, so wird das Objekt als große Wand dargestellt.

– Mit dem Attribut »WALL« legen Sie fest, ob man durch das Feld, auf dem ein Objekt liegt, hindurchgehen kann oder nicht. Das erste Element im Array, daß unter »firsthing« zu finden ist, ist eine Wand. Die entsprechende Nummer in der Landkarte »Playmap« ist »0x4001« oder »WALL4«.

Attribut	Status	Bedeutung
BEWEGBAR	–	Gegenstand läßt sich mitnehmen.
LEBENDIG	TOT,SCHLAFEND,BETRUNKEN,HUNGRIG,MUEDE,HELLWACH,SATT	Zustände von Lebewesen
GENIESSBAR	GIFTIG,NAHRHAFT,HEILEND,ALKOHOL	Gegenstand läßt sich essen.
MITTUER	OFFEN,VERSCHLOSSEN	Gegenstand läßt sich öffnen.
MITDETAIL	–	Es existieren nähere Informationen über diesen Gegenstand in der Datei mit dem Namen, der unter »void *alive« in dieser Struktur gespeichert ist.

Tabelle 1. Die Attribute und der dazugehörige Status eines Objektes

In Tabelle 2 finden Sie alle notwendigen Erläuterungen, um sämtliche Strukturen des Spiels nach Ihren eigenen Wünschen abzuändern und selbst ein neues oder erweitertes Spiel zu schaffen.
Über die Struktur »spielfeld« greifen Sie während des Spiels auf alle wichtigen Informationen zu, da alle Strukturen über Zeiger miteinander verbunden sind.

Tips zur Lösung

Gespräch mit Mylady Miriam: Seien Sie einfach Sie selbst: ein galanter, verführerischer, aber nicht aufdringlicher Abenteurer, dann löst sich das Problem von selbst.
In der Zelle: Wofür ist Schloß Maskavan bekannt?
Beim Grafen: Er besitzt nicht nur Schloß Maskavan, sondern auch noch andere Dinge, die Ihnen weiterhelfen könnten. Wenn Sie jedoch nicht wissen, was Sie wollen, kann er es Ihnen auch nicht geben.
In der Bibliothek: Hier gibt es viel lesenswerte Literatur über Schloß, Leute und Zeit. Auch die unscheinbarsten Bücher verbergen manches Geheimnis.
Geheimtüren werden meist gut bewacht.
In der anderen Dimension: Magiere sind nur mit bestimmten Waffen zu überwinden.

Warum die Fackeln flackern

Es existieren zwei unterschiedliche Arten der Animation. Zum einen sehen Sie, wie die Fackeln an der Wand brennen und dabei flackern. Außerdem bewegen sich auch die Monster, wenn Sie mit ihnen kämpfen. Der erste Effekt kommt durch eine spezielle Funktion zustande. Diese arbeitet auf der Basis, daß sie die Zeiger der Texturen anhand einer bestimmten Animationsliste verändert. Da bei jedem neuen Bild über diese Zeiger auf die Grafikdaten der Texturen zugegriffen wird, entsteht eine Animation. Je nachdem, wie viele Bilder in diese Animationsliste eingetragen sind, wird eine Bewegung fließender. Den Aufbau der Animationsstrukturen finden Sie in Tabelle 3. Dies zeigt auch die Grenze dessen, was an Bildumfang in einer Zeitschrift veröffentlicht werden kann. Um wirklich perfekte Animationen zu sehen, können Sie selbst noch mit dem in dieser Ausgabe enthaltenen Malprogramm einige Texturen zeichnen und in die global definierte Animationsliste eintragen. Den Rest erledigt dann das Programm. Die zweite Art von Animation ist die com-

Feldname	Bedeutung
Struktur »spielfeld«: Hauptstruktur, über die alle Informationen des Spiels erreichbar sind	
playmap	Zeiger auf die Landkarte des Spiels: zweidimensionales Array von Integern, die als Zeiger auf Objekte verwendet werden
playdim	Größe der Landkarte: Exponent zur Basis 2
floorcol	Farbregisternummer der Fußbodenfarbe
ceilcol	Farbregisternummer der Deckenfarbe
firstthing	Zeiger auf ein Array von allen Gegenständen des Spiels vom Typ »thing«
adventurer	Zeiger auf die Informationen über den Spieler
Struktur »thing«: Objektstruktur	
name	Zeiger auf Name des Objektes
attribute	siehe Tabelle 1
status	siehe Tabelle 1
cond()	Zeiger auf Funktion, falls dieses Objekt spezielle Bedingungen erfüllen muß
actpos()	Zeiger auf Funktion, die ausgeführt wird, falls die Bedingungsfunktion positiv antwortet
actneg()	Zeiger auf Funktion, die ausgeführt wird, falls die Bedingungsfunktion negativ antwortet
texture	Zeiger auf eine Struktur »text«, zur Darstellung des Objektes
alive	Zeiger auf eine Struktur »mind«, falls das Objekt ein Gehirn besitzt
verf	Zeiger auf eine Struktur »verfolger«
Struktur »text«: enthält Informationen für die Texturen eines Objektes	
xdim	Größe der Texturen auf den x-Seiten: Exponent zur Basis 2
xmap	Zeiger auf die Grafikdaten der Texturen für die x-Seiten
ydim	Größe der Texturen auf den y-Seiten: Exponent zur Basis 2
ymap	Zeiger auf die Grafikdaten der Texturen für die y-Seiten
Struktur »adventurer«: enthält alle aktuellen Informationen über den Zustand des Spielers	
name	Zeiger auf Zeichenkette mit Namen
posx,posy,look	Position und Blickwinkel des Spielers
selected	momentan angeklicktes Gepäckstück
carry	Array von acht Zeiger auf Strukturen vom Typ »thing« für die Gepäckstücke
nr	Array mit Datentyp »integer«, das die Nummern der Playmap aufnimmt, solange man den Gegenstand trägt
status	siehe Status für »LEBENDIG« in Tabelle 1
waffe	aktuelle Waffe: Zeiger auf Struktur vom Typ »thing«
muede	wie wach ist der Held (intern)
hungrig	wie satt ist der Held (intern)
le	Lebenskraft des Helden (intern)
alter	Alter des Helden (intern)
geschlecht	Geschlecht des Helden
kraft	wie stark ist der Held (intern)
stufe	Erfahrungsstufe (intern)
Struktur »verfolger«: Informationen über Monster	
next,prev	Zeiger auf nächste und vorhergehende Struktur
posx,posy	Integer mit der aktuellen Position des Verfolgers
typ	interne Verwendung
le	Lebensenergie des Monsters
geschw	Geschwindigkeit bei der Fortbewegung
tp	Trefferpunkte beim Zuschlagen

Tabelle 2. Die Strukturen und ihre Bedeutungen

Feldname	Bedeutung
Struktur »animobj«: Animation von Texturen; globaler Zeiger »animlist«	
t	Zeiger auf zu verändernde Struktur vom Typ »text«
textcount	Anzahl der Bilder der Animation
aktutext	interner Zähler
ANIM[]	Array von Zeiger auf Strukturen vom Typ »text«, die die Animationsbilder enthalten

Tabelle 3. Der Aufbau von Strukturen zu Animationen

putergesteuerte Bewegung von Objekten auf dem Spielfeld. Dazu gibt es einige wichtige Grundlagen: Nehmen Sie an, ein Monster greift Sie an. Dann muß es Sie zuerst wahrnehmen, verfolgen und schließlich mit Ihnen kämpfen. All diese Funktionen erledigt das Programm automatisch. Das einzige, was Sie selbst zu tun haben, um ein neues Monster zu schaffen, ist, es zu malen. Dann legen Sie eine Struktur vom Typ »thing« an und geben dem Monster über das Feld »verf« die ihm entsprechende Gefährlichkeit mit. Schließlich tragen Sie die Adresse der Struktur wie oben erklärt im Array der Thing-Zeiger ein und bringen die Nummer, die dem Array-Element entspricht, auf dem Spielfeld mit dem Attribut »OBJECT« unter.

Viel Spaß bei der Schöpfung neuer Welten mit weiteren Gefahren, Rätseln, Lebewesen und Abenteuern.

(Matthias Oberdorfer/bm)

Bildschirmaufbau und Bedienung

Während des Spiels gliedert sich der Bildschirm in drei Teile, denen verschiedene Aufgaben zukommen. Links oben befindet sich das 3D-Fenster zur Abenteuerwelt, in der Sie sich mit Hilfe der Maus fortbewegen. Rechts oben finden Sie die Mausfelder mit den Action-Buttons. Quer über dem unteren Teil liegt das Textfeld, in dem die Informationen ausgegeben werden und die Kommunikation abläuft.

• Das 3D-Fenster

Wenn Sie auf die rechte Maustaste drücken, verschwindet der Mauszeiger vom Bildschirm. Nun ist das 3D-Fenster aktiv: Ihr Auge in die virtuelle Welt. Es ist also immer der Teil der Welt sichtbar, den Sie gerade wirklich anschauen. Mit der Maus können Sie jetzt in Echtzeit durch die virtuelle Welt reisen. Analog zur Maus bewegen Sie sich vor oder zurück. Seitliche Bewegungen der Maus führen zu einer Drehung Ihrer Person. Auf diese Weise können Sie sich zum Beispiel an einen Mauervorsprung heranschleichen und vorsichtig um die Ecke spähen...

• Mausfelder

Drücken Sie auf die rechte Maustaste, schalten Sie zwischen der Steuerung im 3D-Fenster und den Mausfeldern hin und her. Mit der linken Maustaste wählen Sie alle Mausfelder an. Deren Bereich ist wiederum dreigeteilt.

Links oben sind die Action-Buttons. Wenn Sie sie durch Drücken der linken Maustaste aktivieren, lösen Sie die dargestellten Funktionen aus. Rechts oben finden Sie die Gegenstände abgebildet, die Sie als Gepäck mit sich führen. Die Felder unten dienen einerseits zum Speichern des aktuellen Spielstandes, Laden eines neuen oder Verlassen des gegenwärtigen Spiels. Andererseits wird hier der Zustand Ihres Abenteurers dargestellt.

Wenn Sie das Feld »Zustand« anklicken, erhalten Sie Informationen über Ihr Befinden im Spiel. In gleicher Höhe entdecken Sie einen grünen Punkt, der Ihren Überlebenswillen repräsentiert. Ist er intensiv grün, so sind Sie voller Tatendrang, ist er schwarz, dann bleibt Ihnen nicht mehr viel Lebenszeit. Rechts daneben wird Ihre Erfahrungsstufe angezeigt.

Je mehr Erfahrungen Sie sammeln, um so höher steigt diese Zahl an. Je höher die Erfahrungsstufe, desto leichter lösen Sie bestimmte Aufgaben und desto stärker sind Sie.

Die Listings für das Rollenspiel

mus.c	Musik aus dem Hintergrund
talk.c	Dialog mit dem PC
3danim.c	3D-Grafik
defs.h	3D-Grafik
defadv.h	3D-Grafik
graph.h	3D-Grafik
mouse.c	3D-Grafik
schalter.c	3D-Grafik
maker.c	Malprogramm
load.c	dieser Beitrag
mask.c	dieser Beitrag
text.c	dieser Beitrag
mask.prj	dieser Beitrag
*.pak	dieser Beitrag (Hexdumps, Buchinhalte)

Mathias Oberdorfer ist Geschäftsführer von Dolphin Data Systems, Unternehmensberater und EDV-Fachautor für Fragen zu Unix- und PC-Betriebssystemen.

```
[MB3]    1: /* Programm: unpack.c
[7B4]    2:    Funktion: Datei entpacken
[1E5]    3:    Sprache:  QC1,QC2,QCW1,CL5,CL51,CL6,
[2M3]    4:              CL7,TC15,TC2,BC1,BC2,BC3
[BØ4]    5:    Autor:    Wolfgang Lorenz
[8Z4]    6:    (C) 1992 DMV Widuch GmbH & Co. KG */
[ØØØ]    7:
[L13]    8: #include <stdio.h>
[6A3]    9: #include <string.h>
[KR4]   1Ø: typedef unsigned char byte;
[XB6]   11: int cdecl main (int argn, char *argv[])
[894]   12: {FILE *pInFile, *pOutFile;
[E95]   13:   byte abWindow[256], bIn, bOut, b;
[F11]   14:   int i;
[142]   15:   if (argn != 3) {
[1AB]   16:     fprintf (stderr, "Aufruf: unpack <Einga
        bedatei> <Ausgabedatei>\n");
[4U1]   17:     return 1;}
[6W4]   18:   pInFile = fopen (argv[1], "rb");
[AH2]   19:   if (!pInFile) {
[LNB]   2Ø:     fprintf (stderr, "Kann die Eingabedatei
         '%s' nicht finden\n", argv[1]);
[4U1]   21:     return 1;}
[N35]   22:   pOutFile = fopen (argv[2], "wb");
[TD2]   23:   if (!pOutFile) {
[PNB]   24:     fprintf (stderr, "Kann die Ausgabedatei
        '%s' nicht anlegen\n", argv[2]);
[4U1]   25:     return 1;}
[DW3]   26:   memset (abWindow, Ø, 256);
[C41]   27:   bIn = Ø;
[YL5]   28:   while ((i = fgetc (pInFile)) != EOF) {
[ØY1]   29:     if (i < 128) {
[GT4]   3Ø:       bOut = (byte) fgetc (pInFile);
```

Listing 1. Mit »unpack.c« lassen Sie die Hexdumps entpacken.

```
[4V1]   31:       while (i--) {
[NC3]   32:         b = abWindow[bOut++];
[Ø43]   33:         fputc (b, pOutFile);
[FY3]   34:         abWindow[bIn++] = b;})
[KA3]   35:       else while (i++ < 256) {
[H44]   36:       b = (byte) fgetc (pInFile);
[Ø43]   37:       fputc (b, pOutFile);
[FY3]   38:       abWindow[bIn++] = b;})
[LS2]   39:   fclose (pInFile);
[5Ø3]   4Ø:   fclose (pOutFile);
[3U1]   41:   return Ø;}
```

```
[W6Ø]    1: /*
[HS2]    2:  Programm: text.c
[337]    3:  Funktion: Speicherplätze für Textures
[K83]    4:  Sprache:  Turbo C 2.Ø
[KR4]    5:  Autor:    Matthias Oberdorfer
[8Z4]    6:  (c) 1992 DMV Widuch GmbH & Co. KG */
[ØØØ]    7:
[2L5]    8: unsigned char ttexture1[64][64];
[6L5]    9: unsigned char ttexture2[64][64];
[ØH5]   1Ø: unsigned char texture13[64][64];
[WG5]   11: unsigned char texture12[64][64];
[ZM5]   12: unsigned char texture3[64][64];
[2N5]   13: unsigned char texture4[64][64];
```

```
[5N5]   14: unsigned char texture5[64][64];
[8N5]   15: unsigned char texture6[64][64];
[BN5]   16: unsigned char texture7[64][64];
[EN5]   17: unsigned char texture8[64][64];
[HN5]   18: unsigned char texture9[64][64];
[MG5]   19: unsigned char texture1Ø[64][64];
[SG5]   2Ø: unsigned char texture11[64][64];
```

Listing 2. »Text.c« reserviert Speicherplatz für die Texturen.

```
[ST1]    1: /* usetalk */
[6Z1]    2: #define GAME
[RL2]    3: #include "talk.c"
```

Listing 3. »Usetalk.c« definiert die Konstante »GAME«.

```
[W6Ø]    1: /*
[XM2]    2:  Programm: load.c
[FP5]    3:  Funktion: Speichern und Laden von
[UY3]    4:            Bildern, Objekten & Maps
[K83]    5:  Sprache:  Turbo C 2.Ø
[KR4]    6:  Autor:    Matthias Oberdorfer
[8Z4]    7:  (c) 1992 DMV Widuch GmbH & Co. KG */
[ØØØ]    8:
[3Z1]    9: #ifndef LOAD
[9Z1]   1Ø: #define LOAD
[8W2]   11: #include<stdio.h>
[XU2]   12: #include<ctype.h>
[ML2]   13: #include<mem.h>
[4L2]   14: #include"defs.h"
[LU6]   15: /* load(te,name): lädt Texture mit name ein
[XØ5]   16: liefert 1 falls erfolgreich */
[FV2]   17: int load(te,name)
[BD2]   18: char *te,*name;
[1L1]   19:  {FILE *fp;
[PØ4]   2Ø:   if((fp=fopen(name,"rb"))==Ø)
[Ø36]   21:    {printf("Bider nicht zu finden\n");
[3E1]   22:     exit(Ø);}
[MW2]   23:   fread(te,4Ø96,1,fp);
[4Y3]   24:   fclose(fp); return(1);}
[KU4]   25: /* printinbox(format, string):
[C16]   26: Gibt string auf Schriftrolle aus. */
[6D6]   27: int printinbox(const char *format, ...);
[F33]   28: int col=(1Ø1<<8)¦15;
[ZC3]   29: static char str[2ØØ];
[225]   3Ø: static unsigned boxx=Ø,boxy=Ø;
[K64]   31: int printinbox(string, t)
[LK3]   32: const char *string,*t;
[K12]   33:  {char s[2];
[ØX1]   34:   int i=Ø,j,k;
[6L1]   35:   s[1]='\Ø';
[ØY3]   36:   sprintf(str,string,t);
[T64]   37:   while((s[Ø]= str[i ++])!=Ø)
[A24]   38:    {if(strchr(&str[i-1],' '))
[ZR5]   39:      {if(((unsigned)strchr(&str[i-1],' ')
[CZ4]   4Ø:        - (unsigned)&str[i-1]) > 32-boxx)
[YV2]   41:        {s[Ø]='\n';i--;}}
[TT3]   42:     else /* letztes Wort */
[815]   43:      if(strlen(&str[i-1] ) > 32-boxx)
[4M2]   44:         {s[Ø]='\n';i--;}
[WW3]   45:      gotoxy(boxx+5,boxy+18);
[R22]   46:      switch(s[Ø])
[HR1]   47:       {case '\n':
[1H2]   48:         boxx=32; break;
[KM1]   49:        case 'ü':
[KM1]   5Ø:        case 'Ü':
[V24]   51:         s[Ø] = s[Ø]=='Ü'?'U':'u';
[272]   52:         print(s,col);
[N53]   53:         str[--i]='e'; break;
[2L1]   54:        case 'Ä':
[2L1]   55:        case 'ä':
[BW3]   56:         s[Ø] = s[Ø]=='Ä'?'A':'a';
[272]   57:         print(s,col);
[N53]   58:         str[--i]='e'; break;
[EM1]   59:        case 'ö':
[EM1]   6Ø:        case 'Ö':
[714]   61:         s[Ø] = s[Ø]=='Ö'?'O':'o';
[272]   62:         print(s,col);
[N53]   63:         str[--i]='e'; break;
[EX1]   64:        case 'ß':
[TL1]   65:         s[Ø] = 's';
[272]   66:         print(s,col);
[D73]   67:         str[--i]='s'; break;
[ØB3]   68:        case 8: /* Backspace */
[7L1]   69:         if(boxx>1)
[PB1]   7Ø:         {boxx--;
[WW3]   71:          gotoxy(boxx+5,boxy+18);
[K72]   72:          print(" ",col);
[NK2]   73:          boxx--;} break;
[8E1]   74:        default:
```

```
[DP4]   75:          if(isgraph(s[Ø])¦¦s[Ø]==' ')
[272]   76:            print(s,col);
[ZRØ]   77:          else
[7L1]   78:           if(boxx>1)
[LB1]   79:            boxx--;}
[8T1]   8Ø:       if(boxx<32)
[C71]   81:        boxx++;
[ZRØ]   82:       else
[ØV2]   83:        {boxy++; boxx=Ø;}
[LL1]   84:       if(boxy>5)
[L93]   85:         {/*Zeilen kopieren */
[PB2]   86:          for(k=Ø;k<5;k++)
[WB2]   87:          for(j=Ø;j<8;j++)
[KZ4]   88:          movedata(ØxaØ ØØ,k*256Ø+46112+j*32Ø,
[WØ4]   89:          ØxaØ ØØ,k*256Ø+43552+j*32Ø,264);
[Ø71]   9Ø:          boxy --;
[553]   91:          /* Zeile löschen */
[WW3]   92:          gotoxy(boxx+5,boxy+18);
[Ø71]   93:          s[Ø]=' ';
[ØPØ]   94:          j=33;
[FK1]   95:          while (j--)
[272]   96:           print(s,col);
[531]   97:         }}}
[5W6]   98: /* loadbild(name,x,y,v): lädt Bild mit name
[F95]   99: auf den Bildschirm an Stelle x,y
[EY6]  1ØØ: v=1 einfache Vergrößerung sonst un das
[K76]  1Ø1: V-fache. Liefert 1 falls erfolgreich */
[834]  1Ø2: int loadbild(name,x,y,v)
[ØU1]  1Ø3: char *name;
[GM2]  1Ø4: unsigned x,y,v;
[1L1]  1Ø5:  {FILE *fp;
[RV1]  1Ø6:   int i, j,k,l;
[EN2]  1Ø7:   unsigned char c;
[EP2]  1Ø8:   /*Datei öffnen */
[PØ4]  1Ø9:   if((fp=fopen(name,"rb"))==Ø)
[836]  11Ø:    {printf("Bilder nicht zu finden\n");
[3E1]  111:     exit(Ø);}
[X43]  112:   for(j=Ø;j<64*v;j+=v)
[S43]  113:    for(i=Ø;i<64*v;i+=v)
[724]  114:     {fread(&c,sizeof(c),1,fp);
[RR2]  115:      if(i+y<32Ø&&y+j<2ØØ)
[UF2]  116:       for(k=Ø;k<v;k++)
[ØG2]  117:        for(l=Ø;l<v;l++)
[A63]  118:        setp(i+x+l,j+y+k,c);}
[4Y3]  119:   fclose(fp); return(1);}
[GP4]  12Ø: /* loadbg(name): lädt bild als
[AC2]  121: Hintergrund */
[YU2]  122: void loadbg(name)
[ØU1]  123: char *name;
[VG1]  124:  {int l=Ø;
[GM2]  125:   unsigned k=Ø,z;
[385]  126:   char far *anf=(char far *)ØxaØ ØØ ØØØ;
[C33]  127:   loadbild(name,Ø,Ø,1);
[CE2]  128:   for(z=Ø;z<4;z++)
[WB2]  129:    for(l=Ø;l<5;l++)
[RK3]  13Ø:      for(k=Ø;k<(z<3?64:8);k++)
[6E4]  131:     movedata(FP_SEG(anf+k*32Ø),
[6N2]  132:     FP_OFF(anf+k*32Ø),
[7D4]  133:     FP_SEG(anf+k*32Ø+l*64+2Ø48Ø*z),
[2D4]  134:     FP_OFF(anf+k*32Ø+l*64+2Ø48Ø*z),
[1RØ]  135:     64);}
[DA6]  136: /* loadpart(): Lädt Teil eines Bildes an
[TY5]  137: eine bestimmte Bildschirmpostion */
[1M5]  138: int loadpart(name,x,y,xord,width)
[ØU1]  139: char *name;
[ØG2]  14Ø: unsigned x,y;
[1L1]  141:  {FILE *fp;
[HR1]  142:   int i, j,k;
[EN2]  143:   unsigned char c;
[EP2]  144:   /*Datei öffnen */
[PØ4]  145:   if((fp=fopen(name,"rb"))==Ø)
[836]  146:    {printf("Bilder nicht zu finden\n");
[3E1]  147:     exit(Ø);}
[NH4]  148:   for(i=Ø,k=Ø;i<64&&k<width;i++,
[V82]  149:                 k +=i<xord?Ø:1)
[ØG2]  15Ø:    for(j=Ø;j<64;j++)
[724]  151:     {fread(&c,sizeof(c),1,fp);
[3U1]  152:      if(i>=xord)
[RR2]  153:       if(i+y<32Ø&&y+j<2ØØ)
[LY2]  154:        setp(x+k ,j+y,c);}
[PT1]  155:   fclose(fp);
[E32]  156:   return(1);}
[AG6]  157: /* saveall(): Speichert den Spielstand;
[L73]  158:  1 wenn erfolgreich */
[VT6]  159: extern verfolger *firstverf, *lastverf;
[BY4]  16Ø: extern char texture9[64][64];
[AR2]  161: int saveall(sp)
[X72]  162: spielfeld *sp;
[1L1]  163:  {FILE *fp;
[1H2]  164:   char name[1ØØ];
[SB4]  165:   thing **t=(sp->firstthing);
[G91]  166:   int i=Ø;
[ØC4]  167:   getsatz(name,"DATEINAME");
[8W2]  168:   /* Datei öffnen */
[7Z2]  169:   fp=fopen(name,"wb");
[521]  17Ø:   if(!fp)
[PC2]  171:    { return(Ø); }
```

Listing 4. Mit »load.c« laden und speichern Sie Objekte, Bilder und Maps.

```
[KE6]  172:    fwrite(sp->ich,sizeof(adventurer),1,fp);
[3V5]  173:    fwrite(sp->playmap,(1<<sp->playdim)*
[WD5]  174:    (1<<sp->playdim),sizeof(int),fp);
[GW1]  175:    while(*(t+i))
[BV5]  176:      {fwrite((*(t+i)),sizeof(thing),1,fp);
[MRØ]  177:       i++;}
[GK4]  178:    fwrite(sp->ich->name,5Ø,1,fp);
[PT1]  179:    fclose(fp);
[E32]  18Ø:    return(1);}
[RØ5]  181: /* loadall(): lädt den Spielstand
[XØ5]  182: liefert 1 falls erfolgreich */
[7Ø4]  183: static char adname[5Ø];
[1H2]  184: char name[1ØØ];
[HN2]  185: int loadall(sp)
[X72]  186: spielfeld *sp;
[1L1]  187:  {FILE *fp;
[E46]  188:    thing **t=(sp->firstthing),*thisthing;
[EF1]  189:    thing th;
[MZ1]  19Ø:    int i=Ø,x,y;
[4B5]  191:    unsigned xa[8]={24Ø,261,282,24Ø,282,
[9N1]  192:    24Ø,261,282};
[BP4]  193:    unsigned ya[8]={ 22, 22, 22, 43,
[S52]  194:    43, 64, 64, 64};
[K85]  195:    load(&texture9[Ø][Ø],"sword3");
[LK4]  196:    /* Verfolger abschütteln */
[XU2]  197:    while(firstverf)
[VR3]  198:     {lastverf=firstverf;
[9R4]  199:      firstverf=firstverf->next;
[6R2]  2ØØ:      free(lastverf);}
[NS3]  2Ø1:    lastverf=firstverf=Ø;
[ØC4]  2Ø2:    getsatz(name,"DATEINAME");
[D82]  2Ø3:    display(OFF);
[EP2]  2Ø4:    /*Datei öffnen */
[2Z2]  2Ø5:    fp=fopen(name,"rb");
[521]  2Ø6:    if(!fp)
[E92]  2Ø7:     {display(ON);
[D32]  2Ø8:      return(Ø);}
[GY3]  2Ø9:    loadbild("self3",24Ø,2Ø,1);
[UF6]  21Ø:    fread(sp->ich,sizeof(adventurer),1,fp);
[WX3]  211:    /* gegenstände setzen */
[WB3]  212:    for(i=Ø;i<MAXCRY;i++)
[BW2]  213:     if(sp->ich->nr[i])
[YK3]  214:      {sp->ich->carry[i]=
[656]  215:     thisthing=*(t+(sp->ich->nr[i]&ØxØfff));
[KK2]  216:       for(x=Ø;x<18;x++)
[SK2]  217:        for(y=Ø;y<17;y++)
[MP5]  218:        setp(x+xa[i],y+ya[i],*(thisthing->
[ZØ5]  219:        texture->xmap+(1<<thisthing->
[KR4]  22Ø:        texture->xdim)*(46+y)+x+23));}
[GR2]  221:    sp->ich->waffe=Ø;
[2P3]  222:    sp->ich->selected = -1;
[NS5]  223:    fread(sp->playmap,(1<<sp->playdim)*
[WD5]  224:    (1<<sp->playdim),sizeof(int),fp);
[4KØ]  225:    i=Ø;
[GW1]  226:    while(*(t+i))
[M15]  227:      {fread(&th,sizeof(thing),1,fp);
[6F4]  228:       (*(t+i))->status=th.status;
[3B5]  229:       (*(t+i))->attribute=th.attribute;
[TW2]  23Ø:       (*(t+i))->texture =
[W35]  231:       MK_FP(FP_SEG((*(t+i))->texture),
[YB3]  232:       FP_OFF(th.texture));
[452]  233:       (*(t+i))->verf =
[XG4]  234:       MK_FP(FP_SEG((*(t+i))->verf),
[9S2]  235:       FP_OFF(th.verf));
[MRØ]  236:       i++;}
[YD3]  237:    sp->ich->name=adname;
[SE4]  238:    fread(sp->ich->name,5Ø,1,fp);
[PT1]  239:    fclose(fp);
[D22]  24Ø:    display(ON);
[E32]  241:     return(1);}
[NZØ]  242: #endif
```

Listing 4. Mit »load.c« laden und speichern Sie Objekte, Bilder und Maps.

```
[W6Ø]    1: /*
[CP2]    2:  Programm: mask.c
[KA4]    3:  Funktion: das Rollenspiel
[K83]    4:  Sprache:  Turbo C 2.Ø
[KR4]    5:  Autor:    Matthias Oberdorfer
[8Z4]    6:  (c) 1992 DMV Widuch GmbH & Co. KG */
[ØØØ]    7:
[7M2]    8: #include<dos.h>
[8W2]    9: #include<stdio.h>
[ML2]   1Ø: #include<mem.h>
[XU2]   11: #include<ctype.h>
[ZP2]   12: #include<time.h>
[Ø43]   13: #include<stdlib.h>
[4L2]   14: #include"defs.h"
[DZ5]   15: verfolger *firstverf=Ø,*lastverf=Ø;
[DY6]   16: extern void getsatz(char *str,char *name);
[Y22]   17: /*Landkarte*/
[VC4]   18: static int feld[32][32] ={
[ZG5]   19: Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4Ø12,Øx4Ø12,
```

```
[ZG5]   2Ø: Øx4Ø12,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4Ø12,
[CH5]   21: Øx4Ø12,Øx4Ø12,Øx4Ø12,Øx4ØØ1,Øx4Ø12,Øx4Ø12,
[RG5]   22: Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,
[AP5]   23: Øx4ØØd,Øx4ØØd,Øx4Ø21,Øx4ØØd,Øx4ØØ1,Øx4ØØ1,
[SN5]   24: Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,ØxaØ34,Øxeø1d,Øx4ØØ1,
[MØ5]   25: ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4ØØ1,ØØØØØØ,ØØØØØØ,
[F55]   26: Øx4ØØ1,Øxeø1d,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[XW4]   27: ØØØØØØ,ØØØØØØ,Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[9G5]   28: Øx4ØØ1,Øx4ØØd,ØØØØØØ,ØØØØØØ,Øxeø1d,Øx4ØØd,
[4D5]   29: Øx4Ø26,ØØØØØØ,Øxaø2e,Øx4ØØ1,Øx4ØØ1,ØØØØØØ,
[YE5]   3Ø: Øxeø1d,Øx4ØØ1,ØØØØØØ,Øx4ØØ2,ØØØØØØ,Øx4ØØ1,
[Y15]   31: Øxeø3Ø,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[P35]   32: Øxeø1d,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[X75]   33: ØØØØØØ,ØØØØØØ,Øx4ØØ1,Øx4ØØd,ØØØØØØ,Øx4Ø27,
[9F5]   34: ØØØØØØ,Øx4ØØd,Øx4ØØ1,Øx4ØØ1,Øx4Ø2b,Øx4ØØ1,
[Z75]   35: Øx4ØØ1,ØØØØØØ,ØØØØØØ,Øx4ØØ1,ØØØØØØ,Øx4ØØ2,
[L55]   36: ØØØØØØ,Øx4ØØ1,ØØØØØØ,ØØØØØØ,Øx4ØØ1,Øx4ØØ1,
[CH5]   37: Øx4Ø12,Øx4Ø12,Øx4Ø12,Øx4ØØ1,Øx4Ø12,Øx4Ø12,
[GB5]   38: Øx4ØØ1,ØØØØØØ,Øxeø1d,ØØØØØØ,Øx4ØØ1,Øx4ØØd,
[665]   39: ØØØØØØ,ØØØØØØ,Øx4ØØd,ØØØØØØ,Øx4Ø22,Øx4ØØd,
[Z75]   4Ø: ØØØØØØ,Øx4ØØd,Øx4ØØ1,Øxaø39,ØØØØØØ,ØØØØØØ,
[3F5]   41: ØØØØØØ,Øx4ØØ2,Øx4ØØb,Øxaø2c,Øxeø3Ø,ØØØØØØ,
[RZ4]   42: Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[H15]   43: ØØØØØØ,Øxaø37,Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[Z55]   44: Øx4ØØ1,Øx4ØØd,Øx4ØØd,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[FD5]   45: Øx4ØØd,ØØØØØØ,ØØØØØØ,Øx4ØØd,Øx4ØØ1,ØØØØØØ,
[M65]   46: ØØØØØØ,Øx4ØØ1,ØØØØØØ,Øx4ØØ2,ØØØØØØ,Øx4ØØ1,
[H45]   47: ØØØØØØ,Øxaø38,Øx4ØØ1,ØØØØØØ,ØØØØØØ,Øx4ØØ1,
[Z45]   48: ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4ØØ1,Øx4ØØ1,ØØØØØØ,
[N95]   49: ØØØØØØ,ØØØØØØ,Øx4ØØ1,Øx4ØØd,Øx4Ø28,ØØØØØØ,
[SB5]   5Ø: Øx4ØØd,ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4ØØd,Øx4ØØd,
[SG5]   51: Øx4ØØ1,Øxaø2e,Øxeø1d,Øx4ØØ1,ØØØØØØ,Øx4ØØ2,
[E95]   52: ØØØØØØ,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,ØØØØØØ,
[KL5]   53: Øx4ØØ1,Øx4ØØ2,Øx4ØØ1,Øx4Ø1f,Øx4ØØ1,Øx4ØØ2,
[V55]   54: Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4ØØd,ØØØØØØ,
[P85]   55: ØØØØØØ,ØØØØØØ,Øx4Ø24,Øx4Ø2Ø,Øx4ØØd,ØØØØØØ,
[KR5]   56: Øx4ØØd,Øx4ØØd,Øx4ØØ1,Øxaø2c,Øxeø1d,ØØØØØØ,
[XW4]   57: ØØØØØØ,ØØØØØØ,Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[P95]   58: ØØØØØØ,ØØØØØØ,Øx4ØØ1,Øxaø2e,Øx4ØØ2,ØØØØØØ,
[U65]   59: Øx4ØØ2,Øxaø2e,Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[U95]   6Ø: Øx4ØØe,ØØØØØØ,Øx4ØØc,ØØØØØØ,Øx4ØØd,ØØØØØØ,
[L85]   61: ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4Ø25,Øx4ØØ1,Øx4ØØ1,
[7H5]   62: Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ9,
[H85]   63: Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,ØØØØØØ,Øx4ØØ1,ØØØØØØ,
[R55]   64: Øx4ØØ2,ØØØØØØ,Øx4ØØ2,ØØØØØØ,Øx4ØØ1,ØØØØØØ,
[MØ5]   65: ØØØØØØ,ØØØØØØ,Øx4ØØd,ØØØØØØ,ØØØØØØ,Øx4Ø23,
[595]   66: ØØØØØØ,ØØØØØØ,Øx4Ø29,Øx4ØØd,ØØØØØØ,Øx4ØØd,
[D95]   67: Øx4ØØ1,Øxaø2c,ØØØØØØ,ØØØØØØ,Øx4ØØ1,ØØØØØØ,
[695]   68: ØØØØØØ,ØØØØØØ,Øx4ØØ1,Øxaø3a,Øx4ØØ1,ØØØØØØ,
[R55]   69: Øx4ØØ1,ØØØØØØ,Øx4ØØ2,ØØØØØØ,Øx4ØØ2,ØØØØØØ,
[WC5]   7Ø: Øx4ØØ1,Øxaø38,ØØØØØØ,ØØØØØØ,Øx4ØØ1,Øx4ØØd,
[Ø35]   71: ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4ØØd,ØØØØØØ,ØØØØØØ,
[115]   72: ØØØØØØ,Øx4ØØd,Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[TG5]   73: Øx4ØØ8,ØØØØØØ,ØØØØØØ,Øxeø2a,Øx4ØØ1,Øx4Ø3b,
[815]   74: Øx4ØØ1,ØØØØØØ,Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[ØB5]   75: ØØØØØØ,ØØØØØØ,Øx4ØØ1,Øx4ØØ1,Øx4Ø1a,Øx4ØØ1,
[NP5]   76: Øx4ØØ1,Øx4ØØ1,Øx4ØØd,Øx4ØØd,Øx4ØØd,Øx4ØØd,
[BH5]   77: Øx4ØØd,Øx4ØØd,Øx4ØØd,Øx4ØØ1,Øx4ØØ1,ØØØØØØ,
[XW4]   78: ØØØØØØ,ØØØØØØ,Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[XF5]   79: Øx4ØØ1,Øxeø3Ø,Øx4ØØ1,ØØØØØØ,Øx4ØØ1,Øxeø1d,
[Z75]   8Ø: ØØØØØØ,ØØØØØØ,ØØØØØØ,Øxeø1d,Øx4ØØ1,ØØØØØØ,
[X75]   81: ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4ØØ1,Øx4ØØI,Øx4ØØ1,
[RG5]   82: Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,
[245]   83: Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4ØØ1,ØØØØØØ,
[N65]   84: ØØØØØØ,Øxaø2d,Øx4ØØ1,ØØØØØØ,Øx4ØØ1,ØØØØØØ,
[VL5]   85: Øx4ØØ1,Øx4ØØ1,Øx4ØØ2,Øxaø1e,Øx4ØØ2,Øx4ØØ1,
[N25]   86: Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4ØØ1,
[CV4]   87: ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[WM5]   88: Øxeø1d,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ3,Øx4ØØ1,
[2H5]   89: Øx4ØØ1,Øx4ØØ1,Øx4ØØ4,Øx4ØØ1,Øx4ØØ1,Øx4ØØ5,
[VA5]   9Ø: Øx4ØØ1,ØØØØØØ,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,ØØØØØØ,
[M15]   91: Øx4ØØ1,ØØØØØØ,Øx4Ø19,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[CV4]   92: ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[MA5]   93: ØØØØØØ,ØØØØØØ,Øxeø1d,Øx4Ø1c,Øx4ØØ1,ØØØØØØ,
[AY4]   94: ØØØØØØ,ØØØØØØ,Øxeø3Ø,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[ØG5]   95: ØØØØØØ,Øxeø3Ø,Øx4ØØ1,Øx4ØØa,Øx4ØØ1,Øx4ØØ1,
[TG5]   96: Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øxeø1d,Øx4ØØ1,ØØØØØØ,
[E75]   97: ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4ØØ1,Øxaø38,ØØØØØØ,
[M65]   98: ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,Øxeø1d,Øx4ØØ1,
[945]   99: Øx4ØØ1,Øxeø3Ø,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[RZ4]  1ØØ: ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4ØØ1,ØØØØØØ,
[CV4]  1Ø1: ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[F95]  1Ø2: Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,Øxaø38,Øx4ØØ1,
[RG5]  1Ø3: Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,
[RG5]  1Ø4: Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,
[BH5]  1Ø5: Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ6,Øx4ØØ1,Øx4ØØ1,
[RZ4]  1Ø6: Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[VG5]  1Ø7: ØØØØØØ,Øxeø1d,Øx4ØØ1,Øx4ØØ1,Øx4Ø1b,Øx4ØØ1,
[L55]  1Ø8: Øx4ØØ1,Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4ØØ1,
[Z45]  1Ø9: ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4ØØ1,Øx4ØØ1,ØØØØØØ,
[XW4]  11Ø: ØØØØØØ,ØØØØØØ,Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[VZ4]  111: ØØØØØØ,ØØØØØØ,Øx4ØØ1,ØØØØØØ,ØØØØØØ,Øx4ØØ1,
[TD5]  112: Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,ØØØØØØ,
[N25]  113: ØØØØØØ,Øxeø1d,Øx4ØØ1,Øx4ØØ1,ØØØØØØ,ØØØØØØ,
[815]  114: ØØØØØØ,Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4ØØ1,
[NC5]  115: Øx4ØØ1,ØØØØØØ,ØØØØØØ,Øxeø1d,Øx4Ø14,ØØØØØØ,
[F55]  116: ØØØØØØ,Øxeø1d,ØØØØØØ,ØØØØØØ,Øx4ØØ1,ØØØØØØ,
[H15]  117: ØØØØØØ,Øx4ØØ1,Øxaø2e,ØØØØØØ,ØØØØØØ,ØØØØØØ,
```

```
[M45]  118: Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4Ø35,ØØØØØØ,
[N35]  119: ØØØØØØ,ØØØØØØ,Øxeø3Ø,Øx4Ø11,ØØØØØØ,ØØØØØØ,
[685]  12Ø: ØØØØØØ,Øx4ØØ1,Øx4ØØ1,Øxaø2e,ØØØØØØ,ØØØØØØ,
[RZ4]  121: Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[ØG5]  122: Øx4ØØ1,ØØØØØØ,Øxaø2f,Øx4ØØ1,Øxaø2c,ØØØØØØ,
[XW4]  123: ØØØØØØ,ØØØØØØ,Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[N25]  124: Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4ØØ1,
[Z45]  125: ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4ØØ1,Øx4ØØ1,ØØØØØØ,
[XW4]  126: ØØØØØØ,ØØØØØØ,Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[RH5]  127: Øx4ØØ1,Øxaø2e,Øx4ØØ1,ØØØØØØ,Øxaø2f,Øx4ØØ1,
[P75]  128: Øxaø2e,ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4ØØ1,ØØØØØØ,
[G65]  129: ØØØØØØ,ØØØØØØ,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,ØØØØØØ,
[K45]  13Ø: ØØØØØØ,Øx4ØØ1,Øxaø2e,ØØØØØØ,ØØØØØØ,Øx4Ø12,
[8C5]  131: Øx4ØØ1,Øx4ØØ1,ØØØØØØ,Øx4ØØ1,Øx4ØØ1,ØØØØØØ,
[Z45]  132: ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4ØØ1,Øx4ØØ1,ØØØØØØ,
[K65]  133: ØØØØØØ,Øx4ØØ1,ØØØØØØ,Øxeø1d,ØØØØØØ,ØØØØØØ,
[RZ4]  134: ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4ØØ1,ØØØØØØ,
[8E5]  135: ØØØØØØ,Øxeø3Ø,Øx4ØØ1,Øx4ØØ1,Øx4Ø12,Øx4Ø12,
[RH5]  136: Øx4Ø1Ø,Øx4ØØ1,Øx4ØØ1,Øxaø2c,ØØØØØØ,Øxaø2e,
[135]  137: Øx4ØØ1,ØØØØØØ,Øxeø1d,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[BA5]  138: Øx4Ø13,ØØØØØØ,ØØØØØØ,Øx4ØØ1,Øx4Ø28,ØØØØØØ,
[XW4]  139: ØØØØØØ,ØØØØØØ,Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[WF5]  14Ø: Øx4ØØ1,ØØØØØØ,ØØØØØØ,Øx4ØØ1,Øxaø2e,Øx4ØØ1,
[UN5]  141: Øxeø31,Øx4Ø33,Øx4Ø12,Øx4ØØ1,Øx4ØØ1,Øxaø2c,
[X85]  142: Øxaø97,Øxaø2e,Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[VZ4]  143: ØØØØØØ,ØØØØØØ,Øx4ØØ1,ØØØØØØ,ØØØØØØ,Øx4ØØ1,
[TD5]  144: Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,ØØØØØØ,
[XW4]  145: ØØØØØØ,ØØØØØØ,Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[TF5]  146: Øxeø3Ø,Øx4ØØ1,ØØØØØØ,ØØØØØØ,Øxaø2e,Øx4ØØ1,
[RG5]  147: Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,
[2E5]  148: Øx4ØØ1,Øx4Ø15,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,ØØØØØØ,
[CV4]  149: ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[9E5]  15Ø: Øxeø1d,ØØØØØØ,Øxeø1d,ØØØØØØ,Øx4ØØ1,Øx4ØØ1,
[355]  151: ØØØØØØ,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,ØØØØØØ,ØØØØØØ,
[U75]  152: Øxaø2e,Øx4ØØ1,Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[RZ4]  153: Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[8Ø5]  154: Øx4Ø16,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[CV4]  155: ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[EK5]  156: Øx4ØØ1,ØØØØØØ,Øxeø3Ø,Øxaø2e,Øx4Ø12,Øx4ØØ1,
[355]  157: ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4ØØ2,Øx4ØØ1,ØØØØØØ,
[8A5]  158: Øxeø3Ø,ØØØØØØ,ØØØØØØ,ØØØØØØ,Øxeø1d,ØØØØØØ,
[E95]  159: ØØØØØØ,ØØØØØØ,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,
[8G5]  16Ø: Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øxaø38,ØØØØØØ,ØØØØØØ,
[815]  161: ØØØØØØ,ØØØØØØ,Øx4ØØ1,ØØØØØØ,Øx4ØØ1,ØØØØØØ,
[2K5]  162: Øx4Ø12,Øx4ØØ2,Øx4ØØ2,Øxeø32,Øx4ØØ2,Øx4ØØ2,
[245]  163: Øx4ØØ1,ØØØØØØ,ØØØØØØ,Øx4ØØ2,Øx4ØØ1,ØØØØØØ,
[YF5]  164: Øxaø2c,ØØØØØØ,ØØØØØØ,Øx4ØØ1,Øx4Ø12,Øx4Ø12,
[GH5]  165: Øx4Ø12,Øx4Ø12,Øx4Ø12,Øx4Ø12,Øx4Ø12,Øx4ØØ1,
[TD5]  166: Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,ØØØØØØ,
[US5]  167: Øxeø1d,Øx4ØØ1,Øx4Ø12,Øx4ØØ2,Øxeø3Ø,Øxeø3Ø,
[F65]  168: Øxeø3Ø,Øx4ØØ2,Øx4ØØ1,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[RG5]  169: Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,
[X95]  17Ø: Øx4Ø12,Øxaø2e,ØØØØØØ,ØØØØØØ,ØØØØØØ,Øxeø3Ø,
[RB5]  171: Øx4Ø12,Øx4ØØ1,ØØØØØØ,ØØØØØØ,Øx4ØØ1,Øxeø3Ø,
[B55]  172: ØØØØØØ,ØØØØØØ,Øx4ø3c,ØØØØØØ,Øx4Ø12,Øx4ØØ2,
[355]  173: ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4ØØ2,Øx4ØØ1,ØØØØØØ,
[Ø45]  174: Øxeø3Ø,ØØØØØØ,Øx4Ø36,ØØØØØØ,ØØØØØØ,ØØØØØØ,
[V95]  175: Øxeø3Ø,Øx4Ø17,ØØØØØØ,ØØØØØØ,ØØØØØØ,Øxeø3Ø,
[L15]  176: ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4Ø18,ØØØØØØ,ØØØØØØ,
[MØ5]  177: ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4ØØ1,ØØØØØØ,ØØØØØØ,
[W55]  178: Øx4Ø12,Øx4ØØ2,ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4ØØ2,
[S85]  179: Øx4ØØ1,Øxaø38,ØØØØØØ,ØØØØØØ,Øx4ØØ1,ØØØØØØ,
[LA5]  18Ø: ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4ØØ1,Øx4Ø12,Øxaø2e,
[U25]  181: ØØØØØØ,ØØØØØØ,ØØØØØØ,ØØØØØØ,Øx4Ø12,Øx4ØØ1,
[XL5]  182: ØØØØØØ,Øxeø1d,Øx4ØØ1,Øxaø2e,Øx4ØØ1,Øxeø3Ø,
[965]  183: ØØØØØØ,ØØØØØØ,Øxeø3Ø,Øx4Ø1f,ØØØØØØ,ØØØØØØ,
[ØL5]  184: Øxeø3Ø,Øx4ØØ2,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,
[RG5]  185: Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,
[LH5]  186: Øx4Ø12,Øx4Ø12,Øx4Ø12,Øx4Ø12,Øx4Ø12,Øx4Ø12,
[WG5]  187: Øx4Ø12,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,
[EH5]  188: Øx4ØØ1,Øx4Ø12,Øx4Ø12,Øx4Ø12,Øx4Ø12,Øx4Ø12,
[XA4]  189: Øx4ØØ1,Øx4ØØ1,Øx4ØØ1,Øx4ØØ1, };
[AA6]  19Ø: static unsigned char black[16][16] ={
[ØM3]  191: Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,
[MP3]  192: Ø,2,2,2,2,2,2,2,2,2,2,2,2,2,2,Ø,
[CM3]  193: Ø,2,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,2,Ø,
[CM3]  194: Ø,2,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,2,Ø,
[DW3]  195: Ø,2,Ø,Ø,Ø,Ø,Ø,Ø,12,12,Ø,Ø,Ø,Ø,Ø,2,Ø,
[SL4]  196: Ø,2,Ø,Ø,12,12,12,12,12,12,12,12,12,Ø,Ø,2,Ø,
[SL4]  197: Ø,2,Ø,Ø,12,12,12,12,12,12,12,12,12,Ø,Ø,2,Ø,
[K34]  198: Ø,2,Ø,Ø,12,Ø,Ø,12,12,Ø,Ø,12,Ø,Ø,2,Ø,
[K34]  199: Ø,2,Ø,Ø,12,Ø,Ø,12,12,Ø,Ø,12,Ø,Ø,2,Ø,
[K34]  2ØØ: Ø,2,Ø,Ø,12,Ø,Ø,12,12,Ø,Ø,12,Ø,Ø,2,Ø,
[DW3]  2Ø1: Ø,2,Ø,Ø,Ø,Ø,Ø,12,12,Ø,Ø,Ø,Ø,Ø,2,Ø,
[TV4]  2Ø2: Ø,2,Ø,12,12,12,12,12,12,12,12,12,12,Ø,2,Ø,
[CM3]  2Ø3: Ø,2,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,2,Ø,
[CM3]  2Ø4: Ø,2,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,2,Ø,
[MP3]  2Ø5: Ø,2,2,2,2,2,2,2,2,2,2,2,2,2,2,Ø,
[664]  2Ø6: Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,} ;
[3H6]  2Ø7: static unsigned char marble[16][16] ={
[8U4]  2Ø8: Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,86,86,
[8U4]  2Ø9: Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,86,86,
[8U4]  21Ø: 86,86,Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,
[8U4]  211: 86,86,Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,
[8U4]  212: Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,86,86,
[8U4]  213: Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,86,86,
[8U4]  214: 86,86,Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,
[8U4]  215: 86,86,Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,
```

```
[8U4]  216: Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,86,86,
[8U4]  217: Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,86,86,
[8U4]  218: 86,86,Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,
[8U4]  219: 86,86,Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,
[8U4]  22Ø: Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,86,86,
[8U4]  221: Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,86,86,
[8U4]  222: 86,86,Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,
[8U4]  223: 86,86,Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,86,86,Ø,Ø,
[6KØ]  224: } ;
[727]  225: extern unsigned char ttexture1[64][64];
[A27]  226: extern unsigned char ttexture2[64][64];
[YY6]  227: extern unsigned char texture13[64][64];
[VY6]  228: extern unsigned char texture12[64][64];
[N66]  229: static unsigned char texture[64][64];
[SG6]  23Ø: static unsigned char texture2[64][64];
[UL6]  231: extern unsigned char texture3[64][64];
[WL6]  232: extern unsigned char texture4[64][64];
[YL6]  233: extern unsigned char texture5[64][64];
[ØM6]  234: extern unsigned char texture6[64][64];
[2M6]  235: extern unsigned char texture7[64][64];
[4M6]  236: extern unsigned char texture8[64][64];
[6M6]  237: extern unsigned char texture9[64][64];
[NY6]  238: extern unsigned char texture1Ø[64][64];
[SY6]  239: extern unsigned char texture11[64][64];
[WB3]  24Ø: #define BEWEGBAR ØxØ1
[7B3]  241: #define LEBENDIG ØxØ2
[3U3]  242: #define GENIESSBAR ØxØ4
[5G5]  243: #define MITTUER Øx Ø8 /* öffenbar*/
[6X4]  244: #define MITDETAIL Øx1Ø /* Text */
[MA4]  245: #define WAFFE Øx2Ø /* WAFFE */
[M13]  246: #define NEHMEN Øx81
[LR2]  247: #define GEBEN Øxc1
[8H3]  248: #define VERLIEREN Øx41
[LE3]  249: #define SPRECHEN Øx86
[V24]  25Ø: #define KONSUMIEREN Øx44
[U83]  251: #define OEFFNEN Øx48
[LU3]  252: #define SCHLIESSEN Øx88
[993]  253: #define SCHAUEN Øx5Ø
[GB3]  254: #define KAEMPFEN Øx6Ø
[FD3]  255: #define SCHLAFEN Øxff
[VY2]  256: /* Status defines */
[VV4]  257: /* Gegenstände mit Inhalten */
[ZK2]  258: #define OFFEN Øx1
[K34]  259: #define VERSCHLOSSEN Øx2
[912]  26Ø: /* Lebewesen*/
[DK2]  261: #define TOT Øx1
[EC3]  262: #define BETRUNKEN Øx2
[N83]  263: #define SCHLAFEND Øx4
[NA3]  264: #define MONSTER Øx8
[AR2]  265: #define SATT Øx1Ø
[5A3]  266: #define HUNGRIG Øx2Ø
[7S2]  267: #define MUEDE Øx4Ø
[4C3]  268: #define HELLWACH Øx8Ø
[WG1]  269: /* Essen */
[DB3]  27Ø: #define NAHRHAFT Øx1Ø
[EØ3]  271: #define GIFTIG Øx2Ø
[K93]  272: #define ALKOHOL Øx4Ø
[783]  273: #define HEILEND Øx8Ø
[Y55]  274: /* findpos(): liefer die Position
[T96]  275: des Gegenstandes, auf den man schaut */
[3N5]  276: unsigned findpos(posx,posy,look)
[U84]  277: unsigned posx,posy,look;
[KH2]  278: {unsigned pos;
[EZ1]  279:  objcount=Ø;
[325]  28Ø:  pos=showpos(posx,posy,look);
[XØ2]  281:  if(objcount)
[F93]  282:   pos=Object[Ø].pos;
[682]  283:  return(pos);}
[285]  284: /* etwassehen(): Zeigt Text einer
[ØT4]  285: näheren Beschreibung an. */
[VM3]  286: etwassehen(pos,pf,ich)
[672]  287: spielfeld *pf;
[E92]  288: unsigned pos;
[9S2]  289: adventurer *ich;
[SU5]  29Ø:  (thing *thisthing=*(pf->firstthing
[VN4]  291:   +(*(pf->playmap+pos)&ØxØfff));
[PR1]  292:   char s[2];
[H21]  293:   int z;
[SB1]  294:   FILE *fp;
[EP2]  295:   /*Datei öffnen */
[U25]  296:   fp=fopen(thisthing->alive,"rb");
[521]  297:   if(!fp)
[TL1]  298:    return(Ø);
[MU1]  299:   s[1]=Ø; z=Ø;
[ZA4]  3ØØ:   while((s[Ø]=getc(fp))!=EOF)
[L93]  3Ø1:     { if(s[Ø]=='\n') z++;
[3M2]  3Ø2:       printinbox(s,"");
[E41]  3Ø3:      if(z>=5)
[P7Ø]  3Ø4:        {
[Ø67]  3Ø5: printinbox("<Weiter - Taste drücken>","");
[6LØ]  3Ø6:          z=Ø;
[E62]  3Ø7:         if(getch()==27)
[HG3]  3Ø8:          {printinbox("\n","");
[E32]  3Ø9:           return(1);}
[XN3]  31Ø:         printinbox("\n","");}}
[4Y3]  311:   fclose(fp); return(1);}
```

Listing 5. »Mask.c« enthält die Kernroutinen für das Rollenspiel.

```
[285]  312: /* etwasnehmen(): Aufnehmen eines
[DH2]  313: Gegenstandes */
[A24]  314: etwasnehmen(pos,pf,ich)
[672]  315: spielfeld *pf;
[E92]  316: unsigned pos;
[9S2]  317: adventurer *ich;
[5P5]  318:  {unsigned xa[8]={24Ø,261,282,24Ø,282,
[9N1]  319:   24Ø,261,282};
[Z45]  32Ø:   unsigned ya[8]={ 22, 22, 22, 43, 43,
[7R1]  321:   64, 64, 64};
[KD5]  322:  thing *thisthing=*(pf->firstthing
[VN4]  323:   +(*(pf->playmap+pos)&ØxØfff));
[HR1]  324:   int i,x,y;
[T64]  325:   /* leerer Eintrag suchen */
[WB3]  326:   for(i=Ø;i<MAXCRY;i++)
[1Ø3]  327:    if(!ich->carry[i])
[PØ1]  328:     break;
[HØ2]  329:  if(i<MAXCRY)
[VH4]  33Ø:   {ich->carry[i]=thisthing;
[WY2]  331:    /* Object löschen */
[Z45]  332:    ich->nr[i]=*(pf->playmap+pos);
[ZA3]  333:    *(pf->playmap+pos)=Ø;
[KK2]  334:    for(x=Ø;x<18;x++)
[SK2]  335:     for(y=Ø;y<17;y++)
[MP5]  336:      setp(x+xa[i],y+ya[i],*(thisthing->
[ZØ5]  337:      texture->xmap+(1<<thisthing->
[675]  338:      texture->xdim)*(46+y)+x+23));}}
[DW5]  339: /* etwasverlieren(): Einen Gegenstand
[ØT1]  34Ø: weglegen */
[7C4]  341: etwasverlieren(pos,pf,ich)
[672]  342: spielfeld *pf;
[E92]  343: unsigned pos;
[9S2]  344: adventurer *ich;
[565]  345:  {unsigned xa[8]={24Ø,261,282,24Ø,
[952]  346:   282,24Ø,261,282};
[BP4]  347:   unsigned ya[8]={ 22, 22, 22, 43,
[S52]  348:   43, 64, 64, 64};
[Ø62]  349:   unsigned x,y;
[7T4]  35Ø:   ich->carry[ich->selected]=Ø;
[WY2]  351:   /* Object löschen */
[975]  352:   *(pf->playmap+pos)=ich->nr[ich->
[FT1]  353:     selected];
[FA4]  354:   ich->nr[ich->selected]=Ø;
[KK2]  355:   for(x=Ø;x<18;x++)
[SK2]  356:    for(y=Ø;y<17;y++)
[5M4]  357:     setp(x+xa[ich->selected],y+
[D34]  358:     ya[ich->selected],22);}
[U15]  359: /* etwasgeben(): Einen Gegenstand
[AL1]  36Ø: wegeben */
[HL3]  361: etwasgeben(pos,pf,ich)
[672]  362: spielfeld *pf;
[E92]  363: unsigned pos;
[9S2]  364: adventurer *ich;
[565]  365:  {unsigned xa[8]={24Ø,261,282,24Ø,
[952]  366:   282,24Ø,261,282};
[BP4]  367:   unsigned ya[8]={ 22, 22, 22, 43,
[S52]  368:    43, 64, 64, 64};
[KD5]  369:   thing *thisthing=*(pf->firstthing
[VN4]  37Ø:   +(*(pf->playmap+pos)&ØxØfff));
[Ø62]  371:   unsigned x,y;
[7T4]  372:   ich->carry[ich->selected]=Ø;
[FA4]  373:   ich->nr[ich->selected]=Ø;
[KK2]  374:   for(x=Ø;x<18;x++)
[SK2]  375:    for(y=Ø;y<17;y++)
[E64]  376:     setp(x+xa[ich->selected],
[SZ3]  377:     y+ya[ich->selected],22);
[Y35]  378:   printinbox("%s bedankt sich.\n",
[R63]  379:             thisthing->name);}
[496]  38Ø: /* etwasessen(): Gegenstand aufessen */
[MN3]  381: etwasessen(pos,pf,ich)
[672]  382: spielfeld *pf;
[E92]  383: unsigned pos;
[9S2]  384: adventurer *ich;
[L56]  385:  {unsigned xa[8]={24Ø,261,282,24Ø,282,24Ø,
[X51]  386:   261,282};
[XL5]  387:   unsigned ya[8]={ 22, 22, 22, 43, 43, 64,
[F81]  388:    64, 64};
[Ø62]  389:   unsigned x,y;
[Ø67]  39Ø: thing *thismeal=ich->carry[ich->selected];
[7T4]  391:   ich->carry[ich->selected]=Ø;
[FA4]  392:   ich->nr[ich->selected]=Ø;
[2N5]  393:   printinbox("Du nimmst %s zu Dir. ",
[5M2]  394:   thismeal->name);
[5S4]  395:   if(thismeal->status&GIFTIG)
[ØM3]  396:    {ich->status ¦=TOT;
[AL1]  397:     ich->le=Ø;
[9F5]  398:     printinbox("Du fühlst Dich sehr "
[9C5]  399:     "schlecht, denn %s war vergiftet. "
[R43]  4ØØ:     ,thismeal->name);}
[LZ4]  4Ø1:   if(thismeal->status& ALKOHOL)
[ZV5]  4Ø2:    {printinbox("Das war Alkohol! Dir "
[4V5]  4Ø3: "dreht sich alles. ",thismeal->name);
[NE4]  4Ø4:     ich->status ¦=BETRUNKEN;
[HY3]  4Ø5:     if(ich->muede > DEFWACH/3)
[3U3]  4Ø6:      ich->muede = DEFWACH/3;}
[7Ø5]  4Ø7:   if(thismeal->status&NAHRHAFT)
[P7Ø]  4Ø8:    {
[4H6]  4Ø9: printinbox("Es schmeckt Dir sehr gut. "
```

```
[5FØ]  41Ø:      ,"");
[D14]  411:      ich->hungrig +=DEFHUNG/3;
[XZ2]  412:      if(ich->le+(LE/5)<LE)
[Ø23]  413:       ich->le += (LE/5);}
[1X4]  414:   if(thismeal->status&HEILEND)
[RP6]  415:     {printinbox("Deine Lebenskraft steigt."
[5FØ]  416:      ,"");
[VZ2]  417:      if(ich->le+(LE/3)<LE)
[AK2]  418:       ich->le += (LE/3);
[ZRØ]  419:      else
[372]  42Ø:       ich->le =LE;}
[C23]  421:   printinbox("\n","");
[KK2]  422:   for(x=Ø;x<18;x++)
[SK2]  423:    for(y=Ø;y<17;y++)
[E64]  424:     setp(x+xa[ich->selected],
[K74]  425:     y+ya[ich->selected],22);}
[DS2]  426: char *dream[]={
[H55]  427: "\"Wo bleibst Du, mein Retter!\"",
[KC6]  428: "\"Ich spuere das Du auf dem Weg zu mir "
[MU4]  429: "bist. Beeil Dich mein Held !\"",
[626]  43Ø: "\"Ich erwarte Dich, mein Abenteurer. "
[6V2]  431: "Komm schnell!\"",
[L65]  432: "\"Mein Held. Rette mich !\"",};
[U74]  433: etwasschlafen(pos,pf,ich)
[672]  434: spielfeld *pf;
[E92]  435: unsigned pos;
[9S2]  436: adventurer *ich;
[WC2]  437:  {int i = rand();
[Ø35]  438:   printinbox("Du schläfst eine "
[Ø52]  439:   "Weile.\n","");
[185]  44Ø:   if(ich->muede+(DEFWACH/3)<DEFWACH)
[DX3]  441:      ich->muede += (DEFWACH/3);
[ZRØ]  442:   else
[TA3]  443:      ich->muede = DEFWACH;
[294]  444:   drawbox(3Ø,2Ø,119+3Ø,99+2Ø,Ø,1);
[SF1]  445:   if(i>16ØØØ)
[XW1]  446:    {  /* dream */
[VØ2]  447:     i=rand()/8192;
[FS4]  448:     loadbild("miriam8",28+3Ø,2Ø+18,1);
[GY4]  449:     printinbox("Du traeumst von "
[3G4]  45Ø:     "Miriam:\n%s\n",*(dream+i));}
[5P1]  451:   delay(5ØØØ);
[VZ2]  452:   if(ich->le+(LE/3)<LE)
[AK2]  453:      ich->le += (LE/3);
[ZRØ]  454:   else
[1Ø2]  455:      ich->le = LE;
[FS4]  456:   ich->status &= (~BETRUNKEN);
[L13]  457:   ich->alter += TAG/7;
[GE4]  458:   if(ich->hungrig>=DEFHUNG/4)
[BD4]  459:    ich->hungrig -=DEFHUNG/4;}
[RF1]  46Ø: /* Dinge */
[SA5]  461: text w1 = {6,&texture5[Ø][Ø],6,
[ZG3]  462: &texture12[Ø][Ø],};
[LA5]  463: text w2 = {6,&texture2[Ø][Ø],6,
[ZG3]  464: &texture12[Ø][Ø],};
[ØB5]  465: text w3 = {6,&texture5[Ø][Ø],6,
[ZG3]  466: &texture12[Ø][Ø],};
[LA2]  467: thing Wand = {
[685]  468:   "Wand aus grossen Steinquadern",
[SK2]  469:   Ø,Ø,Ø,Ø,Ø,&w1,Ø,Ø,};
[MN4]  47Ø: text w4 = {4,&marble[Ø][Ø],
[6Z2]  471:   4,&marble[Ø][Ø],};
[YH3]  472: thing marmorWand ={
[2V4]  473:   "Wand aus marmoriertem Stein",
[YK2]  474:   Ø,Ø,Ø,Ø,Ø,&w4,Ø,Ø,};
[H74]  475: char geheimtext[2][2] ={
[RD2]  476:   255,255,255,255,};
[TY6]  477: text geheimtextzu = {6,&texture12[Ø][Ø],
[BD3]  478:   6,&texture12[Ø][Ø],};
[H37]  479: text geheimtextof = {1,&geheimtext[Ø][Ø],
[DR3]  48Ø:   1,&geheimtext[Ø][Ø],};
[DR4]  481: int oeffnegtuer(pos,pf,ich)
[E92]  482: unsigned pos;
[9S2]  483: adventurer *ich;
[672]  484: spielfeld *pf;
[SU5]  485:  {thing *thisthing=*(pf->firstthing
[VN4]  486:   +(*(pf->playmap+pos)&ØxØfff));
[LV1]  487:     MUSPORT=6;
[KK4]  488:     *(pf->playmap+pos) ¦= OBJECT;
[VØ6]  489:     thisthing->texture = &geheimtextof;
[LK4]  49Ø:     *(pf->playmap+pos) &= (~WALL);
[NX4]  491:     showwelt(ich->posx,ich->posy,
[GS2]  492:      ich->look,12Ø,pf);
[MZ4]  493:    thisthing->status ¦= OFFEN;}
[8D5]  494: void schliessegtuer(pos,pf,ich)
[E92]  495: unsigned pos;
[9S2]  496: adventurer *ich;
[672]  497: spielfeld *pf;
[SU5]  498:  {thing *thisthing=*(pf->firstthing
[VN4]  499:   +(*(pf->playmap+pos)&ØxØfff));
[NV1]  5ØØ:     MUSPORT=7;
[Y26]  5Ø1:     thisthing->texture = &geheimtextzu;
[NW4]  5Ø2:     *(pf->playmap+pos) &= (~OBJECT);
[ØD4]  5Ø3:     *(pf->playmap+pos) ¦= WALL;
[NX4]  5Ø4:     showwelt(ich->posx,ich->posy,
[GS2]  5Ø5:     ich->look,12Ø,pf);
[AF5]  5Ø6:     thisthing->status &= (~OFFEN);}
[AN5]  5Ø7: int geheimtuer(pos,command,pf,ich)
```

```
[HN3]  5Ø8: unsigned pos,command;
[9S2]  5Ø9: adventurer *ich;
[672]  51Ø: spielfeld *pf;
[6Z2]  511:  {switch(command)
[6D2]  512:    {case OEFFNEN:
[W14]  513:      oeffnegtuer(pos,pf,ich);
[UL1]  514:      return(1);
[8T2]  515:     case SCHLIESSEN:
[MK4]  516:      schliessegtuer(pos,pf,ich);
[E32]  517:      return(1);}
[D32]  518:    return(Ø);}
[FA3]  519: thing geheimtuer1 ={
[MV4]  52Ø:   "Geheimtuer in der Steinwand",
[GR5]  521:   MITTUER,Ø,Ø,geheimtuer,Ø,&w1,Ø,Ø,};
[CF3]  522: thing geheimtuer2 = {
[MV4]  523:   "Geheimtuer in der Steinwand",
[ZM6]  524:   MITTUER,Ø,Ø,geheimtuer,Ø,&geheimtextzu,
[VVØ]  525:   Ø,Ø,};
[EF3]  526: thing geheimtuer3 = {
[MV4]  527:   "Geheimtuer in der Steinwand",
[ZM6]  528:   MITTUER,Ø,Ø,geheimtuer,Ø,&geheimtextzu,
[VVØ]  529:   Ø,Ø,};
[GF3]  53Ø: thing geheimtuer4 = {
[MV4]  531:   "Geheimtuer in der Steinwand",
[ZM6]  532:   MITTUER,Ø,Ø,geheimtuer,Ø,&geheimtextzu,
[VVØ]  533:   Ø,Ø,};
[KF3]  534: thing geheimtuer5 = {
[MV4]  535:   "Geheimtuer in der Steinwand",
[ZM6]  536:   MITTUER,Ø,Ø,geheimtuer,Ø,&geheimtextzu,
[VVØ]  537:   Ø,Ø,};
[MF3]  538: thing geheimtuer6 = {
[MV4]  539:   "Geheimtuer in der Steinwand",
[ZM6]  54Ø:   MITTUER,Ø,Ø,geheimtuer,Ø,&geheimtextzu,
[VVØ]  541:   Ø,Ø,};
[BK4]  542: text t1 = { /* geschlossen */
[NG6]  543:   6,&texture4[Ø][Ø],6,&texture4[Ø][Ø],};
[YG3]  544: text t3 = { /* offen */
[1H6]  545:   6,&texture6[Ø][Ø],6,&texture6[Ø][Ø],};
[GA4]  546: int oeffnetuer(pos,pf,ich)
[E92]  547: unsigned pos;
[9S2]  548: adventurer *ich;
[672]  549: spielfeld *pf;
[SU5]  55Ø:  {thing *thisthing=*(pf->firstthing
[VN4]  551:   +(*(pf->playmap+pos)&ØxØfff));
[KK4]  552:      *(pf->playmap+pos) |= OBJECT;
[LV1]  553:     MUSPORT=6;
[W24]  554:     thisthing->texture = &t3;
[NX4]  555:     showwelt(ich->posx,ich->posy,
[GS2]  556:     ich->look,12Ø,pf);
[LK4]  557:     *(pf->playmap+pos) &= (~WALL);
[MZ4]  558:    thisthing->status |= OFFEN;}
[BØ5]  559: void schliessetuer(pos,pf,ich)
[E92]  56Ø: unsigned pos;
[9S2]  561: adventurer *ich;
[672]  562: spielfeld *pf;
[SU5]  563:  {thing *thisthing=*(pf->firstthing
[VN4]  564:   +(*(pf->playmap+pos)&ØxØfff));
[NW4]  565:     *(pf->playmap+pos) &= (~OBJECT);
[NV1]  566:     MUSPORT=7;
[U24]  567:     thisthing->texture = &t1;
[NX4]  568:     showwelt(ich->posx,ich->posy,
[GS2]  569:     ich->look,12Ø,pf);
[ØD4]  57Ø:     *(pf->playmap+pos) |= WALL;
[NW4]  571:     *(pf->playmap+pos) &= (~OBJECT);
[AF5]  572:      thisthing->status &= (~OFFEN);}
[VV4]  573: int tuer(pos,command,pf,ich)
[HN3]  574: unsigned pos,command;
[9S2]  575: adventurer *ich;
[672]  576: spielfeld *pf;
[6Z2]  577:  {switch(command)
[6D2]  578:    {case OEFFNEN:
[GU3]  579:      oeffnetuer(pos,pf,ich);
[UL1]  58Ø:      return(1);
[8T2]  581:     case SCHLIESSEN:
[ZE4]  582:      schliessetuer(pos,pf,ich);
[E32]  583:      return(1);}
[D32]  584:    return(Ø);}
[FN2]  585: thing Tuer1 = {
[ZT6]  586:  "Holztuer aus Eiche mit Inschrift >III<",
[8M6]  587:   MITTUER,VERSCHLOSSEN,Ø,tuer,Ø,&t1,Ø,Ø,};
[SE2]  588: thing Tuer2 ={
[2L6]  589: "Holztuer aus Eiche mit Inschrift >II<",
[8M6]  59Ø:  MITTUER,VERSCHLOSSEN,Ø,tuer,Ø,&t1,Ø,Ø,};
[RN2]  591: thing Tuer3 = {
[PF6]  592:   "Holztuer aus Eiche mit Inschrift >I<",
[8M6]  593:   MITTUER,VERSCHLOSSEN,Ø,tuer,Ø,&t1,Ø,Ø,};
[VN2]  594: thing Tuer4 = {
[276]  595:   "Holztuer aus Eiche",MITTUER,Ø,Ø,tuer,
[NN1]  596:   Ø,&t1,Ø,Ø,};
[6E5]  597: int keycond(pos,command,pf,ich)
[HN3]  598: unsigned pos,command;
[9S2]  599: adventurer *ich;
[672]  6ØØ: spielfeld *pf;
[SU5]  6Ø1:  {thing *thisthing=*(pf->firstthing
[VN4]  6Ø2:   +(*(pf->playmap+pos)&ØxØfff));
[6Z2]  6Ø3:    int x,y,distance;
[LA6]  6Ø4:   y=pos/(1<<pf->playdim)-(ich->posy>>8);;
[M36]  6Ø5:   x=(pos%(1<<pf->playdim))-(ich->posx>>8);
```

```
[PK6]  6Ø6:   distance=(unsigned)sqrt((float)x*x+y*y);
[DE2]  6Ø7:   if(distance<=2)
[AL2]  6Ø8:   switch(command)
[6D2]  6Ø9:    {case OEFFNEN:
[8Y3]  61Ø:      if(thisthing==&Tuer1||
[XS3]  611:         thisthing==&Tuer2||
[P13]  612:         thisthing==&Tuer3)
[P7Ø]  613:        {
[SC6]  614:       thisthing->status &= (~VERSCHLOSSEN);
[C16]  615:        printinbox("Der Schluessel oeffnet"
[M73]  616:        " das Schloß\n","");}
[ZRØ]  617:       else
[B76]  618:       printinbox("Das kannst du nicht mit "
[6C5]  619:        "diesem Schluessel öffnen\n","");
[UL1]  62Ø:       return(1);
[8T2]  621:      case SCHLIESSEN:
[8Y3]  622:       if(thisthing==&Tuer1||
[XS3]  623:          thisthing==&Tuer2||
[P13]  624:          thisthing==&Tuer3)
[EA6]  625:        {thisthing->status |= VERSCHLOSSEN;
[FT4]  626:         printinbox("Der Schluessel "
[H86]  627:         "verschließt das Schloß\n","");}
[ZRØ]  628:       else
[956]  629:        printinbox("Das kannst du nicht mit"
[9M6]  63Ø:     " diesem Schluessel abschließen\n","");
[E32]  631:       return(1);}
[ZRØ]  632:    else
[MU4]  633:     printinbox("Du bist zu weit "
[NR2]  634:     "entfernt.\n","");
[D32]  635:     return(Ø);}
[ZM1]  636: int mkey()
[W62]  637:  { return(1);}
[K35]  638:  text k1 = {6,&texture13[Ø][Ø],
[CD3]  639:  6,&texture13[Ø][Ø],};
[482]  64Ø: thing key1 = {
[VA4]  641:   "Schluessel aus Messing",
[TA6]  642:   BEWEGBAR|MITTUER,Ø,mkey,keycond,Ø,&k1,
[SVØ]  643:   -1,Ø);
[C82]  644: thing Tuer5={
[AK6]  645:   "Holztuer mit Inschrift: >Bibliothek<",
[8M6]  646:   MITTUER,VERSCHLOSSEN,Ø,tuer,Ø,&t1,Ø,Ø,};
[MK5]  647: int keycondb(pos,command,pf,ich)
[HN3]  648: unsigned pos,command;
[9S2]  649: adventurer *ich;
[672]  65Ø: spielfeld *pf;
[SU5]  651:  {thing *thisthing=*(pf->firstthing
[VN4]  652:   +(*(pf->playmap+pos)&ØxØfff));
[6Z2]  653:    int x,y,distance;
[LA6]  654:   y=pos/(1<<pf->playdim)-(ich->posy>>8);;
[M36]  655:   x=(pos%(1<<pf->playdim))-(ich->posx>>8);
[PK6]  656:   distance=(unsigned)sqrt((float)x*x+y*y);
[DE2]  657:   if(distance<=2)
[AL2]  658:   switch(command)
[6D2]  659:    {case OEFFNEN:
[KD3]  66Ø:      if(thisthing==&Tuer5)
[P7Ø]  661:       {
[SC6]  662:       thisthing->status &= (~VERSCHLOSSEN);
[C16]  663:        printinbox("Der Schluessel oeffnet"
[M73]  664:        " das Schloß\n","");}
[ZRØ]  665:       else
[956]  666:        printinbox("Das kannst du nicht mit"
[2K5]  667:        " diesem Schluessel öffnen\n","");
[UL1]  668:       return(1);
[8T2]  669:      case SCHLIESSEN:
[KD3]  67Ø:       if(thisthing==&Tuer5)
[EA6]  671:        {thisthing->status |= VERSCHLOSSEN;
[FT4]  672:         printinbox("Der Schluessel "
[H86]  673:         "verschließt das Schloß\n","");}
[ZRØ]  674:       else
[6G5]  675:        printinbox("Das kannst du nicht "
[3Z6]  676: "mit diesem Schluessel abschließen\n","");
[E32]  677:       return(1);}
[ZRØ]  678:     else
[MU4]  679:      printinbox("Du bist zu weit "
[NR2]  68Ø:      "entfernt.\n","");
[D32]  681:      return(Ø);}
[782]  682: thing key2 = {
[S17]  683:   "Schluessel aus Gold",BEWEGBAR|MITTUER,
[894]  684:   Ø,mkey,keycondb,Ø,&k1,-1,Ø);
[G82]  685: thing Tuer6={
[UP6]  686:   "Holztuer mit Inschrift: >Mademoiselle"
[DZ5]  687:   " Miriam<",MITTUER,Ø,Ø,tuer,Ø,&t1,Ø,Ø,};
[YØ2]  688: thing Tuer7=
[936]  689:  {"Holztuer aus Fichte mit Inschrift "
[P46]  69Ø:   ">Wache<",MITTUER,Ø,Ø,tuer,Ø,&t1,Ø,Ø,);
[LE2]  691: thing Tuer8= {
[2Ø6]  692:   "Holztuer aus Fichte mit Inschrift "
[X46]  693:   ">Lager<",MITTUER,Ø,Ø,tuer,Ø,&t1,Ø,Ø,);
[RE2]  694: thing Tuer9= {
[Y96]  695:   "Holztuer aus Tanne",MITTUER,Ø,Ø,tuer,
[NN1]  696:   Ø,&t1,Ø,Ø,};
[ZN2]  697: thing Tuer1Ø= {
[Y96]  698:   "Holztuer aus Tanne",MITTUER,Ø,Ø,tuer,
[NN1]  699:   Ø,&t1,Ø,Ø,};
[ØP2]  7ØØ: thing Tuer11= {
[CH4]  7Ø1:   "Holztuer aus Ebenholz mit "
```

Listing 5. »Mask.c« enthält die Kernroutinen für das Rollenspiel.

```
[V16]  7Ø2:    "Inschrift >Marmorzimmer<",MITTUER,
[1U2]  7Ø3:    Ø,Ø,tuer,Ø,&t1,Ø,Ø,};
[1P2]  7Ø4: thing Tuer12= {
[9F5]  7Ø5:    "Holztuer aus Ebenholz",MITTUER,
[1U2]  7Ø6:    Ø,Ø,tuer,Ø,&t1,Ø,Ø,};
[2P2]  7Ø7: thing Tuer13= {
[276]  7Ø8:    "Holztuer aus Eiche",MITTUER,Ø,Ø,tuer,
[NN1]  7Ø9:    Ø,&t1,Ø,Ø,};
[3P2]  71Ø: thing Tuer14= {
[595]  711:    "Holztuer aus Eiche",MITTUER,Ø,Ø,
[HB2]  712:    tuer,Ø,&t1,Ø,Ø,};
[9F2]  713: thing Tuer15={
[276]  714:    "Holztuer aus Eiche",MITTUER,Ø,Ø,tuer,
[NN1]  715:    Ø,&t1,Ø,Ø,};
[5P2]  716: thing Tuer16= {
[FX5]  717:    "Holztuer aus Eiche mit Inschrift "
[Z96]  718:    ">Schlosstor<",MITTUER,VERSCHLOSSEN,
[RX2]  719:    Ø,tuer,Ø,&t1,Ø,Ø,};
[6L5]  72Ø: int keycondt(pos,command,pf,ich)
[HN3]  721: unsigned pos,command;
[9S2]  722: adventurer *ich;
[672]  723: spielfeld *pf;
[SU5]  724:  {thing *thisthing=*(pf->firstthing
[VN4]  725:   +(*(pf->playmap+pos)&ØxØfff));
[6Z2]  726:    int x,y,distance;
[LA6]  727:   y=pos/(1<<pf->playdim)-(ich->posy>>8);;
[M36]  728:   x=(pos%(1<<pf->playdim))-(ich->posx>>8);
[PK6]  729:   distance=(unsigned)sqrt((float)x*x+y*y);
[DE2]  73Ø:   if(distance<=2)
[AL2]  731:   switch(command)
[6D2]  732:    {case OEFFNEN:
[3K3]  733:      if(thisthing==&Tuer16)
[P7Ø]  734:       {
[SC6]  735:       thisthing->status &= (~VERSCHLOSSEN);
[FT4]  736:        printinbox("Der Schluessel "
[XT4]  737:        "oeffnet das Schloß\n","");
[VZ1]  738:        showend2();}
[ZRØ]  739:      else
[B76]  74Ø:      printinbox("Das kannst du nicht mit "
[6C5]  741:       "diesem Schluessel öffnen\n","");
[UL1]  742:      return(1);
[8T2]  743:     case SCHLIESSEN:
[3K3]  744:      if(thisthing==&Tuer16)
[EA6]  745:       {thisthing->status |= VERSCHLOSSEN;
[FT4]  746:        printinbox("Der Schluessel "
[H86]  747:        "verschließt das Schloß\n","");}
[ZRØ]  748:      else
[6G5]  749:       printinbox("Das kannst du nicht "
[3Z6]  75Ø: "mit diesem Schluessel abschließen\n","");
[E32]  751:      return(1);}
[ZRØ]  752:   else
[MU4]  753:    printinbox("Du bist zu weit "
[NR2]  754:    "entfernt.\n","");
[D32]  755:    return(Ø);}
[D82]  756: thing key4 = {
[617]  757:   "Schluessel aus Eisen",BEWEGBAR|MITTUER,
[GB4]  758:   Ø,mkey,keycondt,Ø,&k1,-1,Ø};
[6P2]  759: thing Tuer17= {
[HB6]  76Ø:   "Holztuer mit Inschrift: >Gemaecher<",
[8M6]  761:   MITTUER,VERSCHLOSSEN,Ø,tuer,Ø,&t1,Ø,Ø,};
[7P2]  762: thing Tuer18= {
[HB6]  763:   "Holztuer mit Inschrift: >Gemaecher<",
[8M6]  764:   MITTUER,VERSCHLOSSEN,Ø,tuer,Ø,&t1,Ø,Ø,};
[TK5]  765: int keycondg(pos,command,pf,ich)
[HN3]  766: unsigned pos,command;
[9S2]  767: adventurer *ich;
[672]  768: spielfeld *pf;
[SU5]  769:  {thing *thisthing=*(pf->firstthing
[VN4]  77Ø:   +(*(pf->playmap+pos)&ØxØfff));
[6Z2]  771:    int x,y,distance;
[LA6]  772:   y=pos/(1<<pf->playdim)-(ich->posy>>8);;
[M36]  773:   x=(pos%(1<<pf->playdim))-(ich->posx>>8);
[PK6]  774:   distance=(unsigned)sqrt((float)x*x+y*y);
[DE2]  775:   if(distance<=2)
[AL2]  776:   switch(command)
[6D2]  777:    {case OEFFNEN:
[E94]  778:      if(thisthing==&Tuer17||
[383]  779:         thisthing==&Tuer18)
[P7Ø]  78Ø:       {
[SC6]  781:       thisthing->status &= (~VERSCHLOSSEN);
[626]  782:        printinbox("Der Schluessel oeffnet "
[NK3]  783:        "das Schloß\n","");}
[ZRØ]  784:      else
[956]  785:       printinbox("Das kannst du nicht mit"
[2K5]  786:       " diesem Schluessel öffnen\n","");
[UL1]  787:      return(1);
[8T2]  788:     case SCHLIESSEN:
[E94]  789:      if(thisthing==&Tuer17||
[383]  79Ø:         thisthing==&Tuer18)
[EA6]  791:       {thisthing->status |= VERSCHLOSSEN;
[FT4]  792:        printinbox("Der Schluessel "
[H86]  793:       "verschließt das Schloß\n","");}
[ZRØ]  794:      else
[6G5]  795:       printinbox("Das kannst du nicht "
[3Z6]  796: "mit diesem Schluessel abschließen\n","");
[E32]  797:      return(1);}
[ZRØ]  798:   else
[MU4]  799:    printinbox("Du bist zu weit "
[NR2]  8ØØ:    "entfernt.\n","");
[D32]  8Ø1:    return(Ø);}
[F22]  8Ø2: thing key3 ={
[A77]  8Ø3: "Schluessel aus Kupfer",BEWEGBAR|MITTUER,
[W94]  8Ø4:   Ø,mkey,keycondg,Ø,&k1,-1,Ø};
[Y17]  8Ø5: text geheimtextbzu = {6,&texture8[Ø][Ø],
[1K3]  8Ø6:   6,&texture8[Ø][Ø],};
[7K5]  8Ø7: void schliessegbtuer(pos,pf,ich)
[E92]  8Ø8: unsigned pos;
[9S2]  8Ø9: adventurer *ich;
[672]  81Ø: spielfeld *pf;
[SU5]  811:  {thing *thisthing=*(pf->firstthing
[VN4]  812:   +(*(pf->playmap+pos)&ØxØfff));
[NV1]  813:     MUSPORT=7;
[Ø76]  814:     thisthing->texture = &geheimtextbzu;
[NW4]  815:     *(pf->playmap+pos) &= (~OBJECT);
[ØD4]  816:     *(pf->playmap+pos) |= WALL;
[NX4]  817:     showwelt(ich->posx,ich->posy,
[GS2]  818:     ich->look,12Ø,pf);
[AF5]  819:     thisthing->status &= (~OFFEN);}
[U36]  82Ø: int geheimbtuer(pos,command,pf,ich)
[HN3]  821: unsigned pos,command;
[9S2]  822: adventurer *ich;
[672]  823: spielfeld *pf;
[6Z2]  824:  {switch(command)
[6D2]  825:    {case OEFFNEN:
[W14]  826:      oeffnegtuer(pos,pf,ich);
[UL1]  827:      return(1);
[8T2]  828:     case SCHLIESSEN:
[CT4]  829:      schliessegbtuer(pos,pf,ich);
[E32]  83Ø:      return(1);}
[D32]  831:    return(Ø);}
[5C2]  832: thing buch14 = {
[NC5]  833:   "Bücherschrank der Bibliothek",
[S76]  834:   MITDETAIL|MITTUER,Ø,Ø,geheimbtuer,Ø,
[ZK4]  835:   &geheimtextbzu,"Buch14",Ø};
[S35]  836: text m1 = {6,&texture13[Ø][Ø],
[CD3]  837:   6,&texture13[Ø][Ø],};
[KZ4]  838: text m2 = {6,&texture7[Ø][Ø],
[1C3]  839:   6,&texture7[Ø][Ø], };
[S35]  84Ø: text m3 = {6,&texture11[Ø][Ø],
[AD3]  841:   6,&texture11[Ø][Ø],};
[7A3]  842: verfolger mp ={ Ø,Ø,
[XK4]  843: 8,2,3|OBJECT|WALL,25,4,Ø,3};
[VW2]  844: thing monster = {
[ØW4]  845:   "Skelett mit gezogener Waffe",
[EE3]  846:   LEBENDIG,TOT,Ø,Ø,Ø,&m1,
[834]  847:   -1, /* rundes Object */&mp,};
[D53]  848: verfolger mp1 ={ Ø,Ø,
[AL4]  849: 8,2,3|OBJECT|WALL,5Ø,3,Ø,6};
[N13]  85Ø: thing monster2 = {
[ØC5]  851:   "Skelett mit starker magischer "
[H15]  852:   "Ausstrahlung",LEBENDIG,TOT,Ø,
[8A4]  853:   Ø,Ø,&m1, -1, /* rundes Object */
[ZA1]  854:   &mp1,};
[RV4]  855: text ma1 = {6,&texture3[Ø][Ø],
[HB3]  856:   6,&texture3[Ø][Ø], };
[5M2]  857: thing mann1 = {
[A64]  858:   "Mann, der angekettet ist",
[T33]  859:   LEBENDIG,Ø,Ø,Ø,Ø, &ma1,
[Y14]  86Ø:   &dovan, /* rundes Object */
[LSØ]  861:   Ø,};
[855]  862: text g1 = {6,&ttexture1[Ø][Ø],
[LR3]  863:   6,&ttexture1[Ø][Ø], };
[PK3]  864: static char help[1ØØ];
[D64]  865: askher(pos,command,pf,ich)
[HN3]  866: unsigned pos,command;
[9S2]  867: adventurer *ich;
[672]  868: spielfeld *pf;
[SU5]  869:  {thing *thisthing=*(pf->firstthing
[VN4]  87Ø:   +(*(pf->playmap+pos)&ØxØfff));
[TØ6]  871:   getsatz(help,"Was willst Du zu dem "
[B13]  872:   "Maedchen sagen ?");
[A82]  873:   strupr(help);
[1E5]  874:   if(!strcmp(help,"BIEGI VAN LIEGI "))
[C52]  875:     {MUSPORT=3;
[ZØ3]  876:      thisthing->cond=Ø;
[ØD3]  877:      thisthing->actpos=Ø;
[CN5]  878:      printinbox("Das Maedchen erwacht "
[SR5]  879:      "langsam aus dem magischen Bann:\n"
[AD5]  88Ø:      ,"(gähnt): Endlich bist du da mein "
[7S2]  881:      "Abenteurer.\n");
[T36]  882:     printinbox("Wo kommst Du denn her ?\n"
[9C2]  883:      ,"");return(1);}
[ZRØ]  884:    else
[E66]  885:    printinbox("Das Maedchen scheint in "
[KM3]  886:    "Trance zu sein.\n","");
[D32]  887:   return(Ø);}
[RL2]  888: thing girl1 = {
[NK5]  889:   "Mädchen mit zerrissenem Kleid",
[RE4]  89Ø:   LEBENDIG,Ø,askher,mkey,Ø,&g1,
[F75]  891:   &miriam2, /* rundes Object */ Ø,};
[GY4]  892:  text b1 = {6,&texture8[Ø][Ø],
[1K3]  893:   6,&texture8[Ø][Ø],};
[AB2]  894: thing buch1 ={
[NC5]  895:   "Bücherschrank der Bibliothek",
[1Ø5]  896:   MITDETAIL,Ø,Ø,Ø,Ø,&b1,"Buch1", Ø };
[9K2]  897: thing buch2 = {
```

```
[NC5]  898:    "Bücherschrank der Bibliothek",
[2L4]  899:    MITDETAIL,Ø,Ø,Ø,Ø,&b1,"Buch2",Ø};
[DK2]  9ØØ: thing buch3 = {
[VB5]  9Ø1:    "Bücherschrank",MITDETAIL,Ø,Ø,Ø,
[7C2]  9Ø2:    Ø,&b1,"Buch3",Ø };
[HK2]  9Ø3: thing buch4 = {
[NC5]  9Ø4:    "Bücherschrank der Bibliothek",
[4L4]  9Ø5:    MITDETAIL,Ø,Ø,Ø,Ø,&b1,"Buch4",Ø};
[NK2]  9Ø6: thing buch5 = {
[NC5]  9Ø7:    "Bücherschrank der Bibliothek",
[XT4]  9Ø8:    MITDETAIL,Ø,Ø,Ø,Ø,&b1,"Buch5",Ø };
[TK2]  9Ø9: thing buch6 = {
[NC5]  91Ø:    "Bücherschrank der Bibliothek",
[YT4]  911:    MITDETAIL,Ø,Ø,Ø,Ø,&b1,"Buch6",Ø };
[XK2]  912: thing buch7 = {
[NC5]  913:    "Bücherschrank der Bibliothek",
[ZT4]  914:    MITDETAIL,Ø,Ø,Ø,Ø,&b1,"Buch7",Ø };
[1L2]  915: thing buch8 = {
[NC5]  916:    "Bücherschrank der Bibliothek",
[8L4]  917:    MITDETAIL,Ø,Ø,Ø,Ø,&b1,"Buch8",Ø};
[5L2]  918: thing buch9 = {
[NC5]  919:    "Bücherschrank der Bibliothek",
[9L4]  92Ø:    MITDETAIL,Ø,Ø,Ø,Ø,&b1,"Buch9",Ø};
[1C2]  921: thing buch1Ø = {
[NC5]  922:    "Bücherschrank der Bibliothek",
[RT4]  923:    MITDETAIL,Ø,Ø,Ø,Ø,&b1,"Buch1Ø",Ø};
[2C2]  924: thing buch11 = {
[NC5]  925:    "Bücherschrank der Bibliothek",
[TT4]  926:    MITDETAIL,Ø,Ø,Ø,Ø,&b1,"Buch11",Ø};
[3C2]  927: thing buch12 = {
[NC5]  928:    "Bücherschrank der Bibliothek",
[VT4]  929:    MITDETAIL,Ø,Ø,Ø,Ø,&b1,"Buch12",Ø};
[4C2]  93Ø: thing buch13 = {
[NC5]  931:    "Bücherschrank der Bibliothek",
[XT4]  932:    MITDETAIL,Ø,Ø,Ø,Ø,&b1,"Buch13",Ø};
[6Ø5]  933:   text s1 = {6,&texture9[Ø][Ø],
[9D3]  934:    6,&texture1Ø[Ø][Ø],};
[B24]  935: scond(pos,command,pf,ich)
[HN3]  936: unsigned pos,command;
[9S2]  937: adventurer *ich;
[672]  938: spielfeld *pf;
[YS4]  939:   {*(pf->playmap+15+6*32) ¦= WALL;
[D32]  94Ø:    return(Ø);}
[WV2]  941: thing schwert = {
[FF5]  942:    "Schwert mit goldenem Knauf und "
[KH5]  943:    "magischem Glanz",BEWEGBAR|WAFFE,
[3S5]  944:    Ø,Ø,scond,Ø,&s1,5, /* rundes ObjectTP*/
[3PØ]  945:    Ø};
[PØ3]  946: thing schwert2 = {
[FM4]  947:    "Schwert mit vielen Macken",
[F44]  948:    BEWEGBAR|WAFFE,Ø,Ø,Ø,Ø,&s1,
[BM3]  949:    1, /* rundes Object */Ø};
[TØ3]  95Ø: thing schwert3 = {
[DG6]  951:    "exzellentes Schwert",BEWEGBAR¦WAFFE,
[C85]  952:    Ø,Ø,Ø,Ø,&s1,3, /* rundes Object */ Ø};
[4B4]  953:   text bw1 = {4,&black[Ø][Ø],
[7N2]  954:   4,&black[Ø][Ø],};
[YB3]  955: thing blackwall = {
[NT5]  956:    "unheimliche schwarze Marmorwand",
[2R2]  957:    Ø,Ø,Ø,Ø,Ø,&bw1,Ø,Ø, };
[664]  958: askhim(pos,command,pf,ich)
[HN3]  959: unsigned pos,command;
[9S2]  96Ø: adventurer *ich;
[672]  961: spielfeld *pf;
[E26]  962:   {getsatz(help,"Was willst Du zu der "
[1U5]  963:    "unheimlichen Marmorwand sagen ?");
[A82]  964:    strupr(help);
[5M5]  965:    if(!strcmp(help,"ALKA RANY HURZ "))
[C52]  966:      {MUSPORT=3;
[Ø75]  967:       *(pf->playmap+pos) ¦= INVISIBLE;
[LK4]  968:       *(pf->playmap+pos) &= (~WALL);
[E32]  969:       return(1);}
[D32]  97Ø:    return(Ø);}
[L44]  971: thing blackwalltuer = {
[NT5]  972:    "unheimliche schwarze Marmorwand",
[GL5]  973:    LEBENDIG,Ø,askhim,mkey,Ø,&bw1,Ø,Ø,};
[Z54]  974: askbar(pos,command,pf,ich)
[HN3]  975: unsigned pos,command;
[9S2]  976: adventurer *ich;
[672]  977: spielfeld *pf;
[E26]  978:   {getsatz(help,"Was willst Du zu der "
[1V5]  979:    "unsichtbare magische Wand sagen ?");
[A82]  98Ø:    strupr(help);
[365]  981:    if(!strcmp(help,"BARRI BE SERR "))
[C52]  982:      {MUSPORT=3;
[Ø75]  983:       *(pf->playmap+pos) ¦= INVISIBLE;
[LK4]  984:       *(pf->playmap+pos) &= (~WALL);
[E32]  985:       return(1);}
[D32]  986:    return(Ø);}
[DØ3]  987: thing barriere = {
[426]  988:    "unsichtbare magische Wand",LEBENDIG,
[DN5]  989:    Ø,askbar,mkey,Ø,&geheimtextof,Ø,Ø, };
[S55]  99Ø:   text mag = {6,&ttexture2[Ø][Ø],
[WG3]  991:    6,&ttexture2[Ø][Ø],};
[8X2]  992: verfolger magv ={
[LR3]  993: Ø,Ø,8,2,3¦WALL,5Ø,Ø,Ø,6,};
[PG2]  994: thing magier = {
[7Z5]  995:    "Maennerportraet in Oel",LEBENDIG,Ø,
[5E4]  996:    askbar,mkey,Ø,&mag,Ø,&magv,};
```

```
[645]  997: text tr1 = {6,&texture[Ø][Ø],6,
[RØ3]  998: &texture[Ø][Ø],};
[SP2]  999: thing trank1 =  {
[U76] 1ØØØ:   "guten Rotwein",BEWEGBAR¦GENIESSBAR,
[8K5] 1ØØ1:   NAHRHAFT¦ALKOHOL,Ø,Ø,Ø,&tr1,-1,Ø };
[YH2] 1ØØ2: thing trank2 = {
[HB6] 1ØØ3:   "Sirup",BEWEGBAR¦GENIESSBAR,NAHRHAFT,
[D22] 1ØØ4:   Ø,Ø,Ø,&tr1,-1,Ø};
[ZH2] 1ØØ5: thing trank3 = {
[CH3] 1ØØ6:   "magischen Heiltrank",
[FF6] 1ØØ7:   BEWEGBAR¦GENIESSBAR,HEILEND,Ø,Ø,Ø, &tr1,
[G31] 1ØØ8:   -1,Ø };
[ØK2] 1ØØ9: thing trank4 = {
[E17] 1Ø1Ø:  "klare Fluessigkeit",BEWEGBAR¦GENIESSBAR,
[KH3] 1Ø11:   GIFTIG,Ø,Ø,Ø,&tr1,-1,Ø};
[1K2] 1Ø12: thing trank5 = {
[4Ø6] 1Ø13:   "Kraftbruehe", BEWEGBAR¦GENIESSBAR,
[MØ4] 1Ø14:   NAHRHAFT,Ø,Ø,Ø,&tr1,-1,Ø };
[XR5] 1Ø15: thing *allthings[]= { &Wand, &Wand,
[3B6] 1Ø16: &blackwall,&Tuer1,&Tuer2,&Tuer3,&Tuer4,
[E55] 1Ø17: &key1,&geheimtuer1,&geheimtuer2,
[9B6] 1Ø18: &geheimtuer3,&geheimtuer4,&buch1,&buch2,
[AD6] 1Ø19: &Tuer5,&key2,&buch3,&Tuer6,&marmorWand,
[6S5] 1Ø2Ø: &Tuer7,&Tuer8,&Tuer9,&Tuer1Ø,&Tuer11,
[4B6] 1Ø21: &Tuer12,&Tuer13,&Tuer14,&Tuer15,&Tuer16,
[CK6] 1Ø22: &monster,&schwert,&blackwalltuer,&buch4,
[Y95] 1Ø23: &buch5,&buch6,&buch7,&buch8,&buch9,
[WU5] 1Ø24: &buch1Ø,&buch11,&buch12,&buch13,&mann1,
[8T6] 1Ø25: &buch14,&trank1,&trank2,&trank3,&schwert2,
[ØK6] 1Ø26: &monster2,&girl1,&barriere,&magier,&key3,
[SV6] 1Ø27: &Tuer17,&Tuer18,&trank4,&trank5,&schwert3,
[7T5] 1Ø28: &key4,&geheimtuer5,&geheimtuer6,Ø, };
[366] 1Ø29: animobj alist[2] ={{&w1,2,Ø,&w2,&w3,},
[U73] 1Ø3Ø: {&m1,2,Ø,&m2,&m3,}};
[NC6] 1Ø31: spielfeld spfld = {&feld[Ø][Ø],5,1Ø,11,
[H52] 1Ø32: allthings,Ø};
[9G5] 1Ø33: int generaldo(pos,command,pf,ich)
[HN3] 1Ø34: unsigned pos,command;
[9S2] 1Ø35: adventurer *ich;
[672] 1Ø36: spielfeld *pf;
[Y43] 1Ø37:  {thing *thisthing;
[DØ3] 1Ø38:  if(ich->selected<Ø)
[D74] 1Ø39:    {/* Etwas im Bildschirm */
[2F4] 1Ø4Ø:    thisthing=*(pf->firstthing
[F55] 1Ø41:    +(*(pf->playmap+pos)&Øxøfff));}
[ZRØ] 1Ø42:   else
[PA6] 1Ø43:     thisthing=ich->carry[ich->selected];
[AL2] 1Ø44:   switch(command)
[ZN2] 1Ø45:    {case SPRECHEN:
[PL5] 1Ø46:      if(thisthing->attribute&LEBENDIG)
[U75] 1Ø47:       talk(ich->name,thisthing->name,
[A23] 1Ø48:         thisthing->alive);
[ZRØ] 1Ø49:      else
[ØX3] 1Ø5Ø:        etwasessen(pos,pf,ich);
[PØ1] 1Ø51:      break;
[V12] 1Ø52:     case NEHMEN:
[LC5] 1Ø53:      etwasnehmen(findpos(ich->posx,
[1T4] 1Ø54:      ich->posy,ich->look),pf,ich);
[PØ1] 1Ø55:      break;
[1X1] 1Ø56:     case GEBEN:
[XU3] 1Ø57:      etwasgeben(pos,pf,ich);
[PØ1] 1Ø58:      break;
[RN2] 1Ø59:     case VERLIEREN:
[FW5] 1Ø6Ø:      etwasverlieren((ich->posx>>8)+(ich->
[WC5] 1Ø61:      posy>>8)*(1<<pf->playdim),pf,ich);
[PØ1] 1Ø62:      break;
[D52] 1Ø63:     case OEFFNEN:
[RV4] 1Ø64:      thisthing->status ¦= OFFEN;
[PØ1] 1Ø65:      break;
[8T2] 1Ø66:     case SCHLIESSEN:
[PZ4] 1Ø67:      thisthing->status &= (~OFFEN);
[NB2] 1Ø68:     case KAEMPFEN:
[976] 1Ø69:      ich->waffe=ich->carry[ich->selected];
[PØ1] 1Ø7Ø:      break;
[9B2] 1Ø71:     case SCHLAFEN:
[BD4] 1Ø72:      etwasschlafen(pos,pf,ich);
[PØ1] 1Ø73:      break;
[Y52] 1Ø74:     case SCHAUEN:
[7W3] 1Ø75:      etwassehen(pos,pf,ich);
[8E1] 1Ø76:     default:
[D32] 1Ø77:      return(Ø);}
[E32] 1Ø78:    return(1);}
[2C5] 1Ø79: generalcond(pos,command,pf,ich)
[HN3] 1Ø8Ø: unsigned pos,command;
[9S2] 1Ø81: adventurer *ich;
[672] 1Ø82: spielfeld *pf;
[Y43] 1Ø83:  {thing *thisthing;
[ZD1] 1Ø84:   int x,y;
[X43] 1Ø85:   unsigned distance;
[F11] 1Ø86:   int i;
[DØ3] 1Ø87:   if(ich->selected<Ø)
[D74] 1Ø88:    {/* Etwas im Bildschirm */
[2F4] 1Ø89:    thisthing=*(pf->firstthing
[VN4] 1Ø9Ø:    +(*(pf->playmap+pos)&Øxøfff));
[736] 1Ø91:   y=pos/(1<<pf->playdim)-(ich->posy>>8);
```

Listing 5. »Mask.c« enthält die Kernroutinen für das Rollenspiel.

```
[M36] 1Ø92:    x=(pos%(1<<pf->playdim))-(ich->posx>>8);
[GU6] 1Ø93:    distance=(unsigned)sqrt((float)x*x+y*y);}
[ZRØ] 1Ø94:    else
[KF6] 1Ø95:     {thisthing=ich->carry[ich->selected];
[MA2] 1Ø96:      distance = Ø;}
[AL2] 1Ø97:    switch(command)
[A62] 1Ø98:     {case NEHMEN:
[RT4] 1Ø99:       if(thisthing&&ich->selected<Ø)
[HR2] 11ØØ:         {if(distance<=2)
[YM3] 11Ø1:           {for(i=Ø;i<MAXCRY;i++)
[GU3] 11Ø2:           /* gibt es noch platz ? */
[1Ø3] 11Ø3:            if(!ich->carry[i])
[PØ1] 11Ø4:             break;
[HØ2] 11Ø5:           if(i<MAXCRY)
[UL1] 11Ø6:            return(1);
[ZRØ] 11Ø7:            else
[TG5] 11Ø8:           printinbox("Du kannst nichts so "
[EG3] 11Ø9:           "viel tragen!\n","");}
[ZRØ] 111Ø:         else
[X45] 1111:           printinbox("Um %s aufzunehmen "
[SW4] 1112:          "bis Du zu weit entfernt.\n",
[R63] 1113:          thisthing->name);}
[ZRØ] 1114:        else
[3G2] 1115:         {if(thisthing)
[BU4] 1116:           printinbox("Du hast %s schon "
[DT4] 1117:           "bei dir\n", thisthing->name);
[ZRØ] 1118:          else
[7K4] 1119:           printinbox("Was willst du "
[LW2] 112Ø:           "nehmen ?\n","");}
[PØ1] 1121:       break;
[RN2] 1122:      case VERLIEREN:
[VD5] 1123:       if(!(ich->selected<Ø) && thisthing)
[8T1] 1124:         /*besitz ?*/
[X75] 1125:        {if(!*(pf->playmap+(ich->posx>>8)
[Ø55] 1126:        +(ich->posy>>8)*(1<<pf->playdim)))
[UL1] 1127:          return(1);
[ZRØ] 1128:         else
[RL5] 1129:          printinbox("An dieser Stelle ist "
[P96] 113Ø:      "kein Platz mehr um %s hinzulegen.\n",
[R63] 1131:          thisthing->name);}
[ZRØ] 1132:       else
[Z42] 1133:        if(thisthing)
[SE5] 1134:          printinbox("Du kannst nur etwas "
[YK5] 1135:          "hinlegen was du vorher genommen "
[SY1] 1136:          "hast.\n","");
[PØ1] 1137:       break;
[1X1] 1138:      case GEBEN:
[VD5] 1139:       if(!(ich->selected<Ø) && thisthing)
[8T1] 114Ø:        /*besitz ?*/
[8E3] 1141:         {if((*(pf->firstthing
[1H6] 1142:  +(*(pf->playmap+pos)&ØxØfff)))->attribute
[WK1] 1143:          &LEBENDIG)
[UL1] 1144:           return(1);
[ZRØ] 1145:          else
[RE4] 1146:           printinbox("Du kannst nur "
[ØX4] 1147:           "Lebewesen etwas geben.\n","");}
[ZRØ] 1148:       else
[Z42] 1149:        if(thisthing)
[SE5] 115Ø:          printinbox("Du kannst nur etwas "
[ØF4] 1151:          "geben was Du besitzt.\n","");
[PØ1] 1152:       break;
[7D2] 1153:      case SPRECHEN:
[D25] 1154:        if(ich->selected>=Ø && thisthing)
[8T1] 1155:        /*besitz ?*/
[P7Ø] 1156:         {
[856] 1157:        if(thisthing->attribute&GENIESSBAR)
[7C5] 1158:           {if(ich->hungrig<DEFHUNG/3*2¦¦
[FN4] 1159:          (thisthing->status&HEILEND))
[UL1] 116Ø:           return(1);
[ZRØ] 1161:          else
[8N5] 1162:         printinbox("Du bist so satt, daß "
[2L5] 1163:         "Du nicht noch %s essen kannst.\n",
[R63] 1164:             thisthing->name);}
[ZRØ] 1165:          else
[815] 1166:           printinbox("Da kannst %s nicht "
[3Z4] 1167:           "essen.\n", thisthing->name);}
[ZRØ] 1168:        else
[4H5] 1169:         if(thisthing->attribute&LEBENDIG
[W53] 117Ø:          && thisthing->alive)
[734] 1171:        {if(!thisthing->status)
[UL1] 1172:          return(1);
[CD5] 1173:      /* Lebenwesen lebt und ist gesund */
[GA4] 1174:        if(thisthing->status&TOT)
[GØ6] 1175:         printinbox("Das Reden ist zwecklos"
[ØT5] 1176:      ", denn %s ist tot!",thisthing->name);
[ZRØ] 1177:        else
[3C5] 1178:        if(thisthing->status&SCHLAFEND)
[ZZ4] 1179:         printinbox("Du redest, doch %s "
[F85] 118Ø:         "schläft.\n",thisthing->name);
[ZRØ] 1181:        else
[YE5] 1182:        if(thisthing->status&BETRUNKEN)
[XY5] 1183:         printinbox("Es kommt keine Antwort"
[ØG6] 1184:  ", %s ist betrunken.\n",thisthing->name);
[U7Ø] 1185:        }
[ZRØ] 1186:        else
[GØ5] 1187:         printinbox("Du redest, aber %s "
[HØ7] 1188: "gibt keine Antwort.\n", thisthing->name);
[PØ1] 1189:       break;
```

```
[D52] 119Ø:       case OEFFNEN:
[DE2] 1191:        if(distance<=2)
[KF5] 1192:         {if(!(thisthing->status&OFFEN))
[4A4] 1193:           {if(!(thisthing->status
[KL2] 1194:            &VERSCHLOSSEN))
[UL1] 1195:             return(1);
[ZRØ] 1196:            else
[X53] 1197:             printinbox("%s ist "
[TZ5] 1198:         "verschlossen.\n", thisthing->name);
[U7Ø] 1199:           }
[ZRØ] 12ØØ:          else
[XF4] 12Ø1:          printinbox("%s ist bereits "
[ØY4] 12Ø2:         "offen.\n", thisthing->name);}
[ZRØ] 12Ø3:         else
[AM5] 12Ø4:          printinbox("Du bist zum Oeffnen "
[UY4] 12Ø5:          "zu weit von %s entfernt.\n",
[ØV2] 12Ø6:          thisthing->name);
[PØ1] 12Ø7:         break;
[8T2] 12Ø8:       case SCHLIESSEN:
[DE2] 12Ø9:          if(distance<=2)
[ØY4] 121Ø:           {if(thisthing->status&OFFEN)
[4A4] 1211:             {if(!(thisthing->status
[KL2] 1212:                  &VERSCHLOSSEN))
[UL1] 1213:               return(1);
[ZRØ] 1214:              else
[X53] 1215:               printinbox("%s ist "
[L76] 1216: "verschlossen.\n", thisthing->name);}
[ZRØ] 1217:            else
[XF4] 1218:             printinbox("%s ist bereits "
[546] 1219: "geschlossen.\n", thisthing->name);}
[ZRØ] 122Ø:          else
[W74] 1221:           printinbox("Du bist zum "
[DL4] 1222:           "Schliessen zu weit von %s "
[675] 1223:          "entfernt.\n", thisthing->name);
[PØ1] 1224:         break;
[NB2] 1225:       case KAEMPFEN:
[D25] 1226:         if(ich->selected>=Ø && thisthing)
[8T1] 1227:          /*besitz ?*/
[P7Ø] 1228:          {
[E85] 1229:           if((ich->carry[ich->selected])->
[MU2] 123Ø:           attribute&WAFFE)
[UL1] 1231:             return(1);
[ZRØ] 1232:           else
[E34] 1233:            printinbox("%s ist keine "
[TS4] 1234:            "Waffe\n",thisthing->name);}
[ZRØ] 1235:         else
[AA5] 1236:          printinbox("Mit welcher Waffe "
[RL4] 1237:          "willst Du kämpfen ?.\n","");
[PØ1] 1238:         break;
[9B2] 1239:       case SCHLAFEN:
[Y92] 124Ø:        if(!firstverf)
[ØU4] 1241:        {if(ich->hungrig>=DEFHUNG/4)
[SG4] 1242:        {if(ich->muede<DEFWACH/4*3)
[UL1] 1243:          return(1);
[ZRØ] 1244:         else
[7F5] 1245:          printinbox("Du bist so wach, daß"
[2D5] 1246:         " Du nicht einschlafen kannst.\n",
[R63] 1247:         thisthing->name);}
[ZRØ] 1248:        else
[HU4] 1249:         printinbox("Du kannst nicht "
[8X4] 125Ø:         "einschlafen, weil Dein Magen "
[3N2] 1251:         "knurrt.\n","");}
[ZRØ] 1252:        else
[HU4] 1253:         printinbox("Du kannst nicht "
[XE5] 1254:         "schlafen, weil untote Kreaturen "
[H64] 1255:         "in der Nähe sind.\n","");
[PØ1] 1256:        break;
[Y52] 1257:       case SCHAUEN:
[2C3] 1258:        if(thisthing->alive)
[UL1] 1259:         return(1);
[ZRØ] 126Ø:        else
[UG5] 1261:         printinbox("Du schaust auf %s und "
[M35] 1262:         "siehst nichts Besonderes.\n",
[ØV2] 1263:         thisthing->name);
[AG1] 1264:        break;}
[D32] 1265:     return(Ø);}
[936] 1266: /* dowith(): Aktionen des Abenteuerers
[322] 1267: mit Dingen */
[4C5] 1268: void dowith(pos,command,pf,ich)
[HN3] 1269: unsigned pos,command;
[9S2] 127Ø: adventurer *ich;
[672] 1271: spielfeld *pf;
[Y43] 1272:  {thing *thisthing;
[DØ3] 1273:   if(ich->selected<Ø)
[D74] 1274:    {/* Etwas im Bildschirm */
[LR4] 1275:    thisthing= *(pf->firstthing
[F55] 1276:    +(*(pf->playmap+pos)&ØxØfff));}
[ZRØ] 1277:   else
[PA6] 1278:     thisthing=ich->carry[ich->selected];
[UG3] 1279:     /* Steck die waffe weg */
[832] 128Ø:          ich->waffe=Ø;
[XD3] 1281:  if(command==SCHLAFEN)
[W56] 1282:   {if(generalcond(pos,command,pf,ich))
[P94] 1283:    /* Allgemeine bedingungen */
[2E5] 1284:     generaldo(pos,command,pf,ich);}
[ZRØ] 1285:  else
[Z42] 1286:  if(thisthing)
[YU5] 1287:   {if(thisthing->attribute&command)
```

```
[A53] 1288:     /* Befehl erlaubt ? */
[BG3] 1289:      (if(thisthing->cond)
[WK5] 129Ø:        /* gibt es spezielle bedingungen */
[XS3] 1291:          (if(!(thisthing->cond)
[ME3] 1292:                (pos,command,pf,ich))
[GU3] 1293:             (if(thisthing->actneg)
[F83] 1294:               (thisthing->actneg)
[SF3] 1295:                 (pos,command,pf,ich);
[FG1] 1296:               return;}
[YD6] 1297:  /* Spezielle bedingung nicht erfüllt */
[1H3] 1298:            if(thisthing->actpos)
[C74] 1299:              (if(!(thisthing->actpos)
[ME3] 13ØØ:               (pos,command,pf,ich))
[2E5] 13Ø1:          generaldo(pos,command,pf,ich);}
[ZRØ] 13Ø2:             else
[2E5] 13Ø3:              generaldo(pos,command,pf,ich);}
[ZRØ] 13Ø4:         else
[GK5] 13Ø5:          if(generalcond(pos,command,pf,ich))
[P94] 13Ø6:           /* Allgemeine bedingungen */
[UV3] 13Ø7:           (if(thisthing->actpos)
[C74] 13Ø8:             (if(!(thisthing->actpos)
[ME3] 13Ø9:                   (pos,command,pf,ich))
[2E5] 131Ø:              generaldo(pos,command,pf,ich);}
[ZRØ] 1311:             else
[2E5] 1312:              generaldo(pos,command,pf,ich);}
[ZRØ] 1313:        else
[GU3] 1314:         (if(thisthing->actneg)
[F83] 1315:           (thisthing->actneg)
[3B4] 1316:                (pos,command,pf,ich);})
[ZRØ] 1317:      else
[AL2] 1318:       switch(command)
[ZN2] 1319:        (case SPRECHEN:
[TP5] 132Ø:          printinbox("Du redest und redest, "
[SP4] 1321:          "%s aber antwortet nicht.\n",
[ØV2] 1322:          thisthing->name);
[PØ1] 1323:          break;
[V12] 1324:         case NEHMEN:
[435] 1325:          printinbox("Du kannst %s nicht "
[FS4] 1326:          "nehmen.\n",thisthing->name);
[PØ1] 1327:          break;
[D52] 1328:         case OEFFNEN:
[435] 1329:          printinbox("Du kannst %s nicht "
[MZ4] 133Ø:          "öffnen.\n",thisthing->name);
[PØ1] 1331:          break;
[8T2] 1332:        case SCHLIESSEN:
[435] 1333:          printinbox("Du kannst %s nicht "
[WF5] 1334:          "schliessen.\n",thisthing->name);
[PØ1] 1335:          break;
[NB2] 1336:        case KAEMPFEN:
[AA5] 1337:          printinbox("Mit welcher Waffe "
[SZ3] 1338:          "willst Du kaempfen.\n",
[ØV2] 1339:          thisthing->name);
[PØ1] 134Ø:          break;
[Y52] 1341:        case SCHAUEN:
[SD5] 1342:          printinbox("Du schaust auf %s.\n",
[ØV2] 1343:          thisthing->name);
[PØ1] 1344:          break;
[1X1] 1345:        case GEBEN:
[4K5] 1346:          printinbox("Du %s nicht geben.\n",
[ØV2] 1347:          thisthing->name);
[PØ1] 1348:          break;
[RN2] 1349:       case VERLIEREN:
[L95] 135Ø:          printinbox("Du trägst %s nicht "
[TX4] 1351:          "bei Dir.\n",thisthing->name);
[1W1] 1352:          break;}})
[7U1] 1353: /*buttons*/
[7Z5] 1354: schalter inf ={Ø,175,84,261,1ØØ,22,22,133,
[VW2] 1355: 22,15,Ø," Zustand",};
[E55] 1356: schalter s ={&inf,218,92+8,26Ø,1Ø8+8,
[YN3] 1357: 22,22,133,22,15,Ø,"SAVE",};
[HA5] 1358: schalter l ={&s,175,92+8,175+42,1Ø8+8,
[UP3] 1359: 22,22,133,22,15,Ø," LOAD",};
[E95] 136Ø: schalter q ={&l,261,92+8,261+42,1Ø8+8,
[TR3] 1361: 22,22,133,22,15,Ø,"QUIT",};
[CK4] 1362: schalter c9 ={&q,282,64,3ØØ,79,
[UF2] 1363: 15,22,133,15,Ø,2,Ø,};
[FM4] 1364: schalter c8 ={&c9,261,64,279,79,
[UF2] 1365: 15,22,133,15,Ø,2,Ø,};
[WL4] 1366: schalter c7 ={&c8,24Ø,64,258,79,
[UF2] 1367: 15,22,133,15,Ø,2,Ø,};
[YR5] 1368: schalter c6 ={&c7,282,43,3ØØ,58,15,22,133,
[AL1] 1369: 15,Ø,2,Ø,};
[TK4] 137Ø: schalter c4 ={&c6,24Ø,43,258,58,
[UF2] 1371: 15,22,133,15,Ø,2,Ø,};
[DH4] 1372: schalter c3 ={&c4,282,22,3ØØ,37,
[UF2] 1373: 15,22,133,15,Ø,2,Ø,};
[5K4] 1374: schalter c2 ={&c3,261,22,279,37,
[UF2] 1375: 15,22,133,15,Ø,2,Ø,};
[KH4] 1376: schalter c1 ={&c2,24Ø,22,258,37,
[UF2] 1377: 15,22,133,15,Ø,2,Ø,};
[ZW4] 1378: schalter drop ={&c1,217,62,237,82,
[XX1] 1379: Ø,Ø,Ø,Ø,Ø,Ø,Ø,"",};
[VW5] 138Ø: schalter nehmen ={&drop,196,62,215,82,
[XX1] 1381: Ø,Ø,Ø,Ø,Ø,Ø,Ø,"",};
[9E5] 1382: schalter ge ={&nehmen,175,62,194,82,
[XX1] 1383: Ø,Ø,Ø,Ø,Ø,Ø,Ø,"",};
[3Y4] 1384: schalter clse ={&ge,217,41,237,61,
[XX1] 1385: Ø,Ø,Ø,Ø,Ø,Ø,Ø,"",};
```

```
[SG5] 1386: schalter opn ={&clse,196,41,215,61,
[XX1] 1387: Ø,Ø,Ø,Ø,Ø,Ø,Ø,"",};
[CY4] 1388: schalter tr ={&opn,175,41,194,61,
[XX1] 1389: Ø,Ø,Ø,Ø,Ø,Ø,Ø,"",};
[LU4] 139Ø: schalter spr ={ /* Sprechen*/
[Ø13] 1391: &tr,217,2Ø,237,4Ø,Ø,Ø,Ø,
[TG1] 1392: Ø,Ø,Ø,"",};
[Ø65] 1393: schalter fight ={ /* kämpfen */
[6P4] 1394: &spr,196,2Ø,215,4Ø,Ø,Ø,Ø,Ø,Ø,Ø,"",};
[NF4] 1395: schalter EX ={ /* Schauen */
[7T4] 1396: &fight,175,2Ø,194,4Ø,Ø,Ø,Ø,Ø,Ø,Ø,"",};
[BV6] 1397: /* ichinfo(ich): erzeugt Information über
[ZT2] 1398: unseren Helden */
[U12] 1399: ichinfo(ich)
[9S2] 14ØØ: adventurer *ich;
[1S5] 14Ø1:  (if(ich->status&TOT¦¦ich->le<=Ø)
[145] 14Ø2:   (printinbox("Du bist leider "
[Y84] 14Ø3:    "gestorben, %s", ich->name);
[3S2] 14Ø4:    if(ich->muede<=ØL)
[R35] 14Ø5:     printinbox(", weil Du zu wenig "
[AG3] 14Ø6:     "geschlafen hast", "");
[573] 14Ø7:    if(ich->hungrig<=ØL)
[4L5] 14Ø8:     printinbox(", weil Du verhungert ",
[2R1] 14Ø9:     "bist", "");}
[ZRØ] 141Ø:  else
[KB4] 1411:  (printinbox("Du bist","");
[834] 1412:  if(ich->status&BETRUNKEN)
[HT5] 1413:    printinbox(" völlig besoffen","");
[ZRØ] 1414:  else
[EF4] 1415:   printinbox(" nuechtern","");
[LT3] 1416:  switch((int)((long)ich->
[E63] 1417:   hungrig*3/DEFHUNG))
[281] 1418:   (case 2:
[V41] 1419:    case 3:
[LM3] 142Ø:      printinbox(", satt","");
[PØ1] 1421:      break;
[K41] 1422:    case Ø:
[8E1] 1423:    default:
[D84] 1424:      printinbox(", hungrig","");
[PØ1] 1425:      break;
[N41] 1426:    case 1:
[KD4] 1427:     printinbox(", noch satt","");
[AG1] 1428:     break;}
[LE6] 1429:  switch((int)((long)ich->muede*3/DEFWACH))
[U71] 143Ø:   (case Ø:
[8E1] 1431:    default:
[ZS3] 1432:     printinbox(", muede","");
[PØ1] 1433:     break;
[S41] 1434:    case 2:
[V41] 1435:    case 3:
[Y94] 1436:     printinbox(", hellwach","");
[PØ1] 1437:     break;
[N41] 1438:    case 1:
[XK3] 1439:     printinbox(", wach","");
[AG1] 144Ø:     break;}
[W56] 1441:   printinbox(" und Du fuehlst Dich","");
[925] 1442:   switch((int)((long)ich->le*6/LE))
[U71] 1443:    (case Ø:
[8E1] 1444:    default:
[NW4] 1445:     printinbox(" dem Tode nahe","");
[PØ1] 1446:     break;
[N41] 1447:    case 1:
[594] 1448:     printinbox(" ganz mies","");
[PØ1] 1449:     break;
[S41] 145Ø:    case 2:
[ØA4] 1451:     printinbox(" schlecht","");
[PØ1] 1452:     break;
[V41] 1453:    case 3:
[ØE5] 1454:     printinbox(" nicht besonders","");
[PØ1] 1455:     break;
[Y41] 1456:    case 4:
[474] 1457:     printinbox(" ganz gut","");
[PØ1] 1458:     break;
[151] 1459:    case 5:
[451] 146Ø:    case 6:
[D55] 1461:     printinbox(" ausgezeichnet","");
[AG1] 1462:     break;}
[LL6] 1463:   sprintf(help,"%d",(int)(ich->alter/TAG)+1
              );
[E66] 1464:   printinbox(". Dies ist Dein %s. Tag im "
[9K2] 1465:   "Schloss",help);
[B16] 1466:   printinbox(". Du glaubst, daß es","");
[TS5] 1467:   switch((int)((ich->alter%TAG)*4/TAG))
[U71] 1468:    (case Ø:
[K84] 1469:      printinbox(" nacht ist","");
[PØ1] 147Ø:      break;
[N41] 1471:     case 1:
[UT4] 1472:      printinbox(" morgens ist","");
[PØ1] 1473:      break;
[S41] 1474:     case 2:
[RE5] 1475:      printinbox(" nachmittags ist","");
[PØ1] 1476:      break;
[V41] 1477:     case 3:
[LL4] 1478:      printinbox(" abends ist","");
[7M1] 1479:      break;})
```

Listing 5. »Mask.c« enthält die Kernroutinen für das Rollenspiel.

```
[LM3] 148Ø:  printinbox(".\n","");}
[KF6] 1481: /* setstatus():Setzt und zeigt Status */
[SD2] 1482: setstatus(ich)
[9S2] 1483: adventurer *ich;
[963] 1484:  {static color col;
[311] 1485:   int c;
[K13] 1486:   if(ich->hungrig>=Ø)
[3K2] 1487:    ich->hungrig --;
[ZRØ] 1488:   else
[6A2] 1489:     {if(ich->le>Ø)
[MK1] 149Ø:        ich->le--;
[YV1] 1491:      if(ich->le>Ø)
[8B4] 1492:        ich->hungrig = DEFHUNG/5;}
[8K2] 1493:   if(ich->muede>=Ø)
[L52] 1494:    ich->muede --;
[ZRØ] 1495:   else
[6A2] 1496:     {if(ich->le>Ø)
[MK1] 1497:        ich->le--;
[YV1] 1498:      if(ich->le>Ø)
[6U3] 1499:        ich->muede = DEFWACH/4;}
[X42] 15ØØ:   ich->alter ++;
[CØ2] 15Ø1:   if(ich->le<=Ø)
[8B3] 15Ø2:    ich->status ¦=TOT;
[LE2] 15Ø3:   col.r=col.b=Ø.Ø;
[MV3] 15Ø4:   col.g=(float)ich->le/LE;
[BU3] 15Ø5:   c=interpret_col(&col);
[KF4] 15Ø6:   setp(28Ø,9Ø,c); setp(28Ø,91,c);
[RF4] 15Ø7:   setp(28Ø,92,c); setp(281,9Ø,c);
[WF4] 15Ø8:   setp(281,91,c); setp(281,92,c);
[PF4] 15Ø9:   setp(282,9Ø,c); setp(282,91,c);
[EB2] 151Ø:   setp(282,92,c);}
[8Y2] 1511: unsigned bef[9] =
[R86] 1512: {SCHAUEN,KAEMPFEN,SPRECHEN,SCHLAFEN,
[XF5] 1513: OEFFNEN,SCHLIESSEN,GEBEN,NEHMEN,
[W72] 1514: VERLIEREN};
[363] 1515: extern int endvid();
[UH6] 1516: extern showobjtext(int i,int l1,int l2,
[ØU6] 1517:  int sb,text *tex,int txbegin,int txend,
[5P1] 1518:  int side);
[ML2] 1519: #include<mem.h>
[2T2] 152Ø: #include<alloc.h>
[LN2] 1521: addverf(pos,sp)
[E92] 1522: unsigned pos;
[X72] 1523: spielfeld *sp;
[WV5] 1524:  {thing *thisthing=*(sp->firstthing
[ZP4] 1525:   +(*(sp->playmap+pos)&ØxØfff));
[AØ3] 1526:  verfolger *vf,*vf2;
[S83] 1527:  if(thisthing->verf)
[RW3] 1528:   {if((vf2=firstverf)!=Ø)
[7W1] 1529:    {while(vf2)
[U53] 153Ø:       {if(vf2->posx+(vf2->
[864] 1531:        posy<<sp->playdim)==pos)
[KC1] 1532:         return;
[KW2] 1533:        vf2=vf2->next;}}
[M76] 1534:   vf=malloc(sizeof(struct verfolger));
[1VØ] 1535:   if(vf)
[EW4] 1536:    {memcpy(vf,thisthing->verf,
[783] 1537:     sizeof(verfolger));
[9V1] 1538:     MUSPORT=1;
[9Ø5] 1539:     vf->posx=pos%(1<<sp->playdim);
[YØ5] 154Ø:     vf->posy=pos/(1<<sp->playdim);
[5W1] 1541:     vf->next=Ø;
[KF4] 1542:     vf->type=*(sp->playmap+pos);
[KC2] 1543:     vf->counter=Ø;
[D62] 1544:     if(firstverf)
[3H3] 1545:      {lastverf->next=vf;
[K33] 1546:       vf->prev=lastverf;
[DE2] 1547:       lastverf=vf;}
[ZRØ] 1548:     else
[7B4] 1549:       {lastverf=firstverf=vf;
[823] 155Ø:        vf->prev=Ø;}}}}
[HZ2] 1551: /* searchnextstep:
[MA6] 1552: Verfolgungsalgorithmus der Monster */
[U33] 1553: searchnextstep(sp)
[X72] 1554: spielfeld *sp;
[RC4] 1555:  {adventurer *ich=sp->ich;
[854] 1556:   verfolger *vf=firstverf;
[7R1] 1557:   int x,y,c;
[7F4] 1558:   int ix=(ich->posx)>>SHIFT;
[CF4] 1559:   int iy=(ich->posy)>>SHIFT;
[8F1] 156Ø:   while(vf)
[232] 1561:    {if(vf->le>Ø)
[AG5] 1562:    if(!vf->counter--¦¦!vf->geschw)
[SØ6] 1563:     {x=ix>vf->posx?1:(ix<vf->posx?-1:Ø);
[8U5] 1564:    y=iy>vf->posy?1:(iy<vf->posy?-1:Ø);
[D54] 1565:    vf->counter = vf->geschw;
[9F4] 1566:    c=*(sp->playmap+vf->posx+x+
[PH4] 1567:    ((vf->posy+y)<<sp->playdim));
[VF1] 1568:    if(c&WALL)
[5Ø5] 1569:     {if(!((c=*(sp->playmap+vf->posx+
[AL5] 157Ø:      ((vf->posy+y)<<sp->playdim)))&WALL))
[WUØ] 1571:      x=Ø;}
[VF1] 1572:    if(c&WALL)
[P95] 1573:     {if(!((c=*(sp->playmap+vf->posx+x+
[NC5] 1574:      ((vf->posy)<<sp->playdim)))&WALL))
[YUØ] 1575:       y=Ø;}
[YT1] 1576:    if(!(c&WALL))
[G73] 1577:     {ix=ix-vf->posx+x;
```

```
[ØW2] 1578:      iy=iy-vf->posy+y;
[GR3] 1579:     if((ix=(ix*ix+iy*iy))>9)
[CL4] 158Ø:      {/* Abhängen von Monstern */
[PC1] 1581:       if(ix>225)
[LE4] 1582:         {vf->prev->next=vf->next;
[C34] 1583:          vf->next->prev=vf->prev;
[UU2] 1584:          if(vf==firstverf)
[WG3] 1585:           firstverf=vf->next;
[ZN2] 1586:          if(vf==lastverf)
[V83] 1587:           lastverf=vf->prev;
[2A2] 1588:          /* Abgehängt */
[VU1] 1589:          free(vf);}
[ZRØ] 159Ø:       else
[6N2] 1591:       {if(vf->geschw)
[Ø61] 1592:         {if(!c)
[M34] 1593:          *(sp->playmap+vf->posx+x+
[876] 1594:     ((vf->posy+y)<<sp->playdim))=vf->type;
[RB4] 1595:          if(*(sp->playmap+vf->posx+
[W16] 1596:     ((vf->posy)<<sp->playdim))==vf->type)
[US3] 1597:          *(sp->playmap+vf->posx
[TG4] 1598:          +(vf->posy<<sp->playdim))=Ø;
[412] 1599:         vf->posx +=x;
[EØ3] 16ØØ:         vf->posy +=y;}}}
[ZRØ] 16Ø1:    else
[GH3] 16Ø2:     {if(!c) /* Freie Bahn ?*/
[FV2] 16Ø3:       {ich->le -=vf->tp;
[RV1] 16Ø4:      MUSPORT=8;
[TZ2] 16Ø5:      blink(Øx3f);}}}}
[FN2] 16Ø6:   vf=vf->next;}}
[VR5] 16Ø7: /* fightit(): Der Held bekämpft ein
[KP1] 16Ø8: Monster */
[5S2] 16Ø9: fightit(pos,sp)
[E92] 161Ø: unsigned pos;
[X72] 1611: spielfeld *sp;
[NM5] 1612:  {unsigned x=pos%(1<<sp->playdim);
[695] 1613:   unsigned y=pos/(1<<sp->playdim);
[2R1] 1614:   int ix,iy;
[854] 1615:   verfolger *vf=firstverf;
[8F1] 1616:   while(vf)
[L55] 1617:   {if(vf->posx==x && vf->posy==y)
[1E5] 1618:     {ix=((sp->ich->posx)>>SHIFT)-x;
[AZ4] 1619:      iy=((sp->ich->posy)>>SHIFT)-y;
[X23] 162Ø:     if((ix*ix+iy*iy)<=9)
[UØ5] 1621:      {if(vf->geschw¦¦(!vf->geschw
[MB4] 1622:        &&sp->ich->waffe==&schwert))
[SR3] 1623:         vf->le -= sp->ich->kraft+
[P95] 1624:       (unsigned)sp->ich->waffe->alive;
[K12] 1625:        MUSPORT = 9;
[B12] 1626:        blink(Øx3fØØ);
[5V1] 1627:        if(vf->le<=Ø)
[RP4] 1628:         {if(*(sp->playmap+vf->posx+
[W16] 1629:  ((vf->posy)<<sp->playdim))==vf->type)
[6Y3] 163Ø:          *(sp->playmap+vf->posx+
[9F4] 1631:          (vf->posy<<sp->playdim))=Ø;
[D34] 1632:          vf->prev->next=vf->next;
[C34] 1633:          vf->next->prev=vf->prev;
[UU2] 1634:          if(vf==firstverf)
[WG3] 1635:           firstverf=vf->next;
[ZN2] 1636:          if(vf==lastverf)
[V83] 1637:           lastverf=vf->prev;
[4L3] 1638:          /* erfahrungspunkte */
[YA3] 1639:          if(sp->ich->stufe>1ØØ)
[F23] 164Ø:           sp->ich->stufe +=2Ø/
[593] 1641:           (sp->ich->stufe/1ØØ);
[ZRØ] 1642:          else
[V23] 1643:           sp->ich->stufe +=2Ø;
[X45] 1644:  /* Der Geist entschwebt hier... */
[4F1] 1645:          free(vf);
[HV1] 1646:          MUSPORT=5;
[RF5] 1647:          printinbox("Du hast Dein Gegner "
[L94] 1648:          "besiegt, er loest sich in "
[U33] 1649:          "Luft aus.\n","");}
[7M1] 165Ø:        break;}}
[FN2] 1651:    vf=vf->next;}}
[RX5] 1652: /* lookmunster(): hält nach Monstern
[HU1] 1653: Ausschau */
[EG2] 1654: int counter=Ø;
[HU2] 1655: lookmunster(sp)
[X72] 1656: spielfeld *sp;
[GH2] 1657:  {if(!counter--)
[A22] 1658:    {counter=2;
[R55] 1659:     addverf(findpos(sp->ich->posx,
[3F5] 166Ø:     sp->ich->posy,sp->ich->look),sp);
[R55] 1661:     addverf(findpos(sp->ich->posx,
[P74] 1662:     sp->ich->posy,numw/8),sp);
[R55] 1663:     addverf(findpos(sp->ich->posx,
[LD4] 1664:     sp->ich->posy,numw/8*3),sp);
[R55] 1665:     addverf(findpos(sp->ich->posx,
[TD4] 1666:     sp->ich->posy,numw/8*5),sp);
[R55] 1667:     addverf(findpos(sp->ich->posx,
[ZD4] 1668:     sp->ich->posy,numw/8*7),sp);
[R55] 1669:     addverf(findpos(sp->ich->posx,
[UB3] 167Ø:     sp->ich->posy,Ø),sp);
[R55] 1671:     addverf(findpos(sp->ich->posx,
[G74] 1672:     sp->ich->posy,numw/2),sp);
[R55] 1673:     addverf(findpos(sp->ich->posx,
[FD4] 1674:     sp->ich->posy,numw/4*3),sp);
[R55] 1675:     addverf(findpos(sp->ich->posx,
```

```
[3V4] 1676:     sp->ich->posy,numw/4),sp);})
[972] 1677: int oldday=Ø;
[BP2] 1678: int oldstufe=Ø;
[ZØ2] 1679: showarm(sp)
[X72] 168Ø: spielfeld *sp;
[RC4] 1681: {adventurer *ich=sp->ich;
[EU2] 1682:  /* Monsteralarm */
[CW2] 1683:  lookmunster(sp);
[6B3] 1684:  searchnextstep(sp);
[ZM4] 1685:  if(sp->ich->alter/TAG>oldday)
[144] 1686:   {oldday=ich->alter/TAG;
[FG3] 1687:    sp->ich->stufe += 2Ø;}
[XØ5] 1688:  if(sp->ich->stufe/1ØØ>oldstufe)
[Ø93] 1689:   {sp->ich->kraft ++;
[MP4] 169Ø:    oldstufe=sp->ich->stufe/1ØØ;
[9Y4] 1691:    printinbox("Du erreichst die "
[ØX5] 1692: "%d. Erfahrungsstufe\n",oldstufe+1);
[L42] 1693:    gotoxy(37,12);
[EZ4] 1694:    print(itoa(oldstufe+1,help,1Ø),
[2Ø2] 1695:    15<<8¦22);}
[433] 1696:  if(ich->status&TOT)
[F62] 1697:   {MUSPORT = 5;
[G52] 1698:    ichinfo(ich);
[735] 1699:    load(&texture9[Ø][Ø],"skull");
[DL5] 17ØØ:    showobjtext(Ø,1ØØ,1ØØ,12Ø,&s1,Ø,255,Ø);}
[ZRØ] 17Ø1:  else
[US2] 17Ø2:   {setstatus(ich);
[H32] 17Ø3:    if(ich->waffe)
[WX2] 17Ø4:     {if(mouse_key(Ø))
[9R4] 17Ø5:      {if(ich->status&BETRUNKEN)
[F95] 17Ø6:         printinbox("Du bist betrunken "
[726] 17Ø7:    "und bist unfaehig zu kaempfen.\n","");
[ZRØ] 17Ø8:       else
[KU4] 17Ø9:         {fightit(findpos(ich->posx,
[B54] 171Ø:          ich->posy,ich->look),sp);
[RV3] 1711:          showobjtext(72,1ØØ,1ØØ,47,
[AØ6] 1712:        ich->waffe->texture,35,154,YSIDE);}}
[ZRØ] 1713:     else
[RV3] 1714:      showobjtext(72,1ØØ,1ØØ,47,
[X76] 1715:     ich->waffe->texture,2Ø,139,XSIDE);}}}
[G91] 1716: int i=Ø;
[VA3] 1717: int gibschluessel()
[K61] 1718:  {if(!i)
[YSØ] 1719:    {i++;
[374] 172Ø:   *((&spfld)->playmap+7+11*32) =
[134] 1721:   *((&spfld)->playmap+7+1Ø*32);
[6T5] 1722:   *((&spfld)->playmap+7+1Ø*32) = ØxaØØf;}
[D32] 1723:   return(Ø);}
[7Z4] 1724: /* Wartefunktion für Intro */
[BN1] 1725: wait(time)
[YK1] 1726: int time;
[HD1] 1727: {long i;
[HM3] 1728:  for(i=Ø;i<time*1ØØL;i++)
[7U1] 1729:   {delay(1Ø);
[KM1] 173Ø:  if(kbhit())
[TF2] 1731:   {while(kbhit())
[R61] 1732:    getch();
[AG1] 1733:    break;}
[ZRØ] 1734:  else
[MV2] 1735:   if(mouse_press())
[CW3] 1736:   {while(mouse_press());
[DH2] 1737:    /*mouse_key(Ø);
[VH2] 1738:    mouse_key(1);  */
[1W1] 1739:    break;}}}
[698] 174Ø: showpic(char *bild,char *text,char *text2,i
            nt time)
[SD2] 1741:  {display(OFF);
[MS3] 1742:  drawbox(2Ø,15,3ØØ,185,Ø,1);
[N71] 1743:  if(bild)
[5G3] 1744:   loadbild(bild,96,25,2);
[9Y4] 1745:  gotoxy(2Ø-(strlen(text)/2)+1,21);
[SH2] 1746:  print(text,129);
[Ø45] 1747:  gotoxy(2Ø-(strlen(text2)/2)+1,22);
[SP2] 1748:  print(text2,129);
[D22] 1749:  display(ON);
[MØ2] 175Ø:  wait(time);}
[VM6] 1751: newtext(char *text,char *text2,int time)
[T24] 1752:  {drawbox(2Ø,155,3ØØ,185,Ø,1);
[9Y4] 1753:   gotoxy(2Ø-(strlen(text)/2)+1,21);
[SH2] 1754:   print(text,129);
[Ø45] 1755:   gotoxy(2Ø-(strlen(text2)/2)+1,22);
[SP2] 1756:   print(text2,129);
[MØ2] 1757:  wait(time);}
[7K2] 1758: extern int col;
[C84] 1759: showintro(spielfeld *sp)
[3T1] 176Ø:  {int ende;
[D82] 1761:   display(OFF);
[WH2] 1762:   loadbg("mauer1");
[MS3] 1763:   drawbox(2Ø,15,3ØØ,185,Ø,1);
[D22] 1764:   display(ON);
[TX6] 1765:   newtext("Eines Tages findet ein grosser"
[XA4] 1766:   ,"Ball auf dem Schloss ...",5);
[YK5] 1767:   newtext("... Deines Vaters statt."
[EY4] 1768:   ,"Unter den Ballgaesten ...",5);
[MX5] 1769:   newtext("... siehst Du ein Maedchen,"
[NA5] 177Ø:   ,"dass Dir sofort auffaellt ...",5);
[9F5] 1771:   showpic("miriam8","Sie lernt Dich "
[4C2] 1772:   "kennen...","",5);
[EK6] 1773:   newtext("\"Na, edler Ritter, willst Du"
[PF3] 1774:   ," mich nicht mal ...",5);
[AH5] 1775:   newtext("... etwas unterhalten?\""
[583] 1776:   ,"fragt sie Dich.",5);
[WD5] 1777:   newtext("Das laesst Du Dir nicht"
[K55] 1778:   ,"zweimal sagen und meinst:",5);
[D82] 1779:   display(OFF);
[SK3] 178Ø:   drawbox(2Ø,5,3ØØ,185,Ø,1);
[LØ4] 1781:   loadbild("miriam8",96,1Ø,2);
[R71] 1782:   col=129;
[D22] 1783:   display(ON);
[1E5] 1784:   ende=talk(sp->ich->name,"Mylady",
[HE1] 1785:   &miriam);
[MN2] 1786:   col = (1Ø1<<8)¦15;
[H51] 1787:   wait(3);
[D82] 1788:   display(OFF);
[WH2] 1789:   loadbg("mauer1");
[WE1] 179Ø:   if(ende<Ø)
[DN6] 1791:   {showpic("miriam8","\"Du bist vielleicht"
[UW5] 1792:    " langweilig!\"","sagt sie und laesst "
[NL2] 1793:    "Dich stehen.",5);
[XG2] 1794:    return(ende);}
[MS3] 1795:   drawbox(2Ø,15,3ØØ,185,Ø,1);
[D22] 1796:   display(ON);
[KB6] 1797:   newtext("Der Ball ist vorueber und die"
[8N5] 1798:   ,"Gaeste sind wieder abgereist.",5);
[EF5] 1799:   newtext("Also machst Du Dich so"
[RW5] 18ØØ:   ,"schnell wie moeglich auf den Weg",5);
[3G5] 18Ø1:   newtext("Nach den Strapazen des"
[NF5] 18Ø2:   ,"langen Weges erreichst Du ...",5);
[YE5] 18Ø3:   showpic("ankunft","... das Schloss "
[133] 18Ø4:   "in der Nacht.","",5);
[1A6] 18Ø5:   newtext("Es ist der 23. November 1432 "
[R44] 18Ø6:   "und","es ist Vollmond.",5);
[UE5] 18Ø7:   newtext("Du weisst nicht warum,"
[TZ4] 18Ø8:   ,"aber Dich beschleicht ein",5);
[KK5] 18Ø9:   newtext("merkwuerdiges Gefuehl."
[8K3] 181Ø:   ," ... AUUAUUUAUU ... ",5);
[VA6] 1811:   newtext("Ploetzlich trifft Dich ein"
[VP5] 1812:   ,"schwerer Schlag am Hinterkopf",5);
[FS3] 1813:   newtext("und Dir wird"
[ØZ3] 1814:   ," schwarz vor Augen...",5);
[E32] 1815:   return(1);}
[PY1] 1816: showtitle()
[SD2] 1817: {display(OFF);
[191] 1818:  cls(Ø,Ø);
[DV1] 1819:  MUSPORT=3;
[ZH5] 182Ø:  showpic("titel1","(c) 1992 DMV Verlag"
[XØ4] 1821:  ,"DOS International",8);}
[ZN1] 1822: showinfo()
[SD2] 1823: {display(OFF);
[K12] 1824:    loadbg("bg3");
[ØB3] 1825:    mkfr(8Ø,15,24Ø,185,15,22);
[8X3] 1826:    drawbox(8Ø,15,24Ø,185,1Ø1,1);
[824] 1827:    loadbild("titel1",128,45,1);
[TZ1] 1828:    gotoxy(12,3);
[H26] 1829:    print("Das Geheimnis von",(1Ø1<<8)¦15);
[XZ1] 183Ø:    gotoxy(12,4);
[1Y5] 1831:    print("Schloss Maskavan",(1Ø1<<8)¦15);
[A42] 1832:    gotoxy(12,15);
[F65] 1833:    print("   Programming",(1Ø1<<8)¦15);
[B42] 1834:    gotoxy(12,16);
[555] 1835:    print("   Art & Design",(1Ø1<<8)¦15);
[C42] 1836:    gotoxy(12,17);
[AX4] 1837:    print("     Writing",(1Ø1<<8)¦15);
[D42] 1838:    gotoxy(12,18);
[SE5] 1839:    print("  Sound Effects",(1Ø1<<8)¦15);
[942] 184Ø:    gotoxy(12,2Ø);
[654] 1841:    print("       by",(1Ø1<<8)¦15);
[B42] 1842:    gotoxy(12,22);
[WE6] 1843:   print("Matthias Oberdorfer",(1Ø1<<8)¦15);
[D22] 1844:    display(ON);
[B91] 1845:    wait(1Ø);
[4M2] 1846:    display(OFF);}
[5N1] 1847: showende()
[SD2] 1848:  {display(OFF);
[WH2] 1849:   loadbg("mauer1");
[M12] 185Ø:   MUSPORT=1Ø;
[HV6] 1851:   showpic("miriam8","Ich hatte gehofft, das
            s"
[PC4] 1852:    ,"du das sagen wuerdest.",5);
[N67] 1853:   newtext("Ich liebe Dich,",spfld.ich->name
            ,5);
[NH6] 1854:   newtext("und ich will mit Dir jetzt ..."
[3UØ] 1855:   ,"",5);
[SV4] 1856:   newtext("gluecklich werden,"
[YG3] 1857:   ,"mein Abenteurer.",2Ø);
[ØH5] 1858:   newtext("ENDE dieses Abenteuers"
[L54] 1859:   ,"(Fortsetzung folgt)",5ØØ);
[UM2] 186Ø:   setgraphik(OFF);
[3E1] 1861:   exit(Ø);}
[KM1] 1862: showend2()
[SD2] 1863:  {display(OFF);
[WH2] 1864:   loadbg("mauer1");
[WM6] 1865:  showpic("ankunft","Schnell fluechtest Du"
```

Listing 5. »Mask.c« enthält die Kernroutinen für das Rollenspiel.

```
[NX3] 1866:    ,"aus diesem Schloss.",5);
[YM5] 1867:    newtext("Von weitem schaust Du,"
[SE3] 1868:    ,"Dich noch mal um.",5);
[HØ5] 1869:    newtext("Du hast das Gefuehl,"
[1B3] 187Ø:    ,"dass Du nicht ...",5);
[V35] 1871:    newtext("... das erreicht hast,"
[6E5] 1872:    ,"was Du eigentlich wollest.",2Ø);
[ØH5] 1873:    newtext("ENDE dieses Abenteuers"
[L54] 1874:    ,"(Fortsetzung folgt)",5ØØ);
[UM2] 1875:    setgraphik(OFF);
[3E1] 1876:    exit(Ø);}
[1S5] 1877: schalter quf ={Ø,1ØØ,116,22Ø,132,43,129
[T63] 1878: ,133,172,43,Ø,"Ende",};
[UC5] 1879: schalter enf ={&quf,1ØØ,92,22Ø,1Ø8,
[MN3] 188Ø: 43,129,133,172,43,Ø,"Info",};
[FK5] 1881: schalter introf ={&enf,1ØØ,44,22Ø,6Ø,
[F15] 1882: 43,129,133,172,43,Ø,"Neues Spiel",};
[VF5] 1883: schalter lf ={&introf,1ØØ,68,22Ø,84,
[MY4] 1884: 43,129,133,172,43,Ø,"Lade Spiel",};
[RL2] 1885: int firstpage()
[YB3] 1886:  {int fertig = Ø,mp=Ø;
[8X3] 1887:   unsigned mx=16Ø<<1,my=5Ø;
[ØR5] 1888:   schalter *first = &lf,*sch,*show=&lf;
[6K2] 1889:   while(fertig!=1)
[SD2] 189Ø:   {display(OFF);
[K12] 1891:    loadbg("bg3");
[ØB3] 1892:    mkfr(8Ø,15,24Ø,185,15,22);
[8X3] 1893:    drawbox(8Ø,15,24Ø,185,1Ø1,1);
[TZ1] 1894:    gotoxy(12,3);
[H26] 1895:    print("Das Geheimnis von",(1Ø1<<8)¦15);
[XZ1] 1896:    gotoxy(12,4);
[1Y5] 1897:    print("Schloss Maskavan",(1Ø1<<8)¦15);
[WE2] 1898:    showsch(show);
[D22] 1899:    display(ON);
[4TØ] 19ØØ:    mp=Ø;
[F62] 19Ø1:    mouse_key(Ø);
[443] 19Ø2:    mouse_move(mx,my);
[BA2] 19Ø3:    mouse_show();
[4F1] 19Ø4:      while(!mp)
[WX2] 19Ø5:       {if(mouse_key(Ø))
[5TØ] 19Ø6:          mp=1;
[T43] 19Ø7:        mouse_pos(&mx,&my);
[3V1] 19Ø8:        mx=mx>>1;}
[FF3] 19Ø9:      while(mouse_press());
[M62] 191Ø:      mouse_hide();
[9E5] 1911:      switch(testsch(mx,my,first,&sch))
[681] 1912:       {case 3:
[BX1] 1913:          showinfo();
[PØ1] 1914:          break;
[S41] 1915:        case 2:
[D82] 1916:         display(OFF);
[7A3] 1917:         mkfr(2Ø,15,3ØØ,185,15,22);
[WW3] 1918:         drawbox(2Ø,15,3ØØ,185,1Ø1,1);
[Ø24] 1919:         loadbild("advent",96,25,2);
[D22] 192Ø:         display(ON);
[V46] 1921:        getsatz(spfld.ich->name,"\n\n\nWie"
[Z35] 1922:         " ist Dein Name, Abenteurer?\n");
[ZU3] 1923:         if(showintro(&spfld)>Ø)
[TL1] 1924:          return(Ø);
[PØ1] 1925:         break;
[N41] 1926:        case 1:
[UL1] 1927:         return(1);
[Y41] 1928:        case 4:
[TF1] 1929:         fertig=1;
[7M1] 193Ø:         break;}}
[UM2] 1931:      setgraphik(OFF);
[3E1] 1932:      exit(Ø);}
[AR2] 1933: char myname[1ØØ];
[Ø55] 1934:  adventurer ich ={ myname,Øx2ØØ,
[484] 1935:  ØxØbØØ,3Ø8,-1,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,
[LM5] 1936:  Ø,Ø,Ø,Ø,Ø,Ø,HELLWACH¦SATT,Ø,DEFWACH/2
[HG5] 1937:  ,DEFHUNG/2,LE/3*2,TAG/4,Ø,MAENNLICH,
[P62] 1938:  KRAFTANF,Ø};
[UWØ] 1939: main()
[AD3] 194Ø:  {unsigned fertig=Ø;
[VE3] 1941:  /* Bildschirmbreite */
[KS2] 1942:  unsigned width;
[YB5] 1943:  /* Winkel des Sichtbereichs 45 - 9Ø
[C53] 1944:     sichtbarer Winkel */
[962] 1945:  unsigned sw;
[823] 1946:  unsigned col,col2;
[X91] 1947:  color c;
[172] 1948:  int loadit=Ø;
[DW2] 1949:  /* Mausposition */
[YS5] 195Ø:  unsigned mx=18Ø<<1,my=1ØØ,mp,schelp;
[4P5] 1951:  schalter *first = &EX,*sch,*show=&c1;
[X13] 1952:  if(!setgraphik(ON))
[CV5] 1953:   {printf("VGA - Karte notwendig!\n");
[3E1] 1954:    exit(Ø);}
[YE3] 1955:  /* Bildschirmaufbau */
[L45] 1956:   /* allgemeine Initialisierung */
[2Y2] 1957:   if(!mouse_check())
[NØ3] 1958:    {setgraphik(OFF);
[DF6] 1959:    printf("Kein Maustreiber gefunden\n");
[3E1] 196Ø:     exit(Ø);}
[XL4] 1961:   mouse_def(&pointer2[Ø],16,6);
[7Y1] 1962:   randomize();
[KY5] 1963: /* Intialisiersung von Spielfeld und
```

```
[4M1] 1964:  Figuren */
[4R1] 1965:  width = 12Ø;
[GF1] 1966:  sw    = 35;
[Y33] 1967:  c.r=.3; c.g=.3;c.b=.3;
[XN5] 1968:  spfld.floorcol=interpret_col(&c);
[793] 1969:  c.r=.1; c.g=.3; c.b=.Ø;
[ZE5] 197Ø:  spfld.ceilcol=interpret_col(&c);
[745] 1971:  initpf(&spfld,3Ø,2Ø,width,1ØØ,sw);
[5E3] 1972:  animlist=&alist[Ø];
[M22] 1973:  animcount=2;
[8L2] 1974:  spfld.ich = &ich;
[HØ2] 1975:  showtitle();
[LG3] 1976:  /* Laden der Grafiken */
[8E5] 1977:  load(texture,"trank");   /* trank */
[AT5] 1978:  load(texture2,"mauer1Ø"); /* mauer1Ø */
[MT5] 1979:  load(texture5,"mauer11"); /* mauer11 */
[MØ6] 198Ø:  load(texture4,"tuerzu");  /* Tuere zu */
[HP4] 1981:  load(texture3,"man");  /* Mann */
[EZ5] 1982:  load(texture6,"tuerof"); /* Tuere of */
[NV5] 1983:  load(texture7,"skell1"); /* Monster */
[BC5] 1984:  load(texture8,"buch"); /* Bibliok */
[BP5] 1985:  load(texture9,"sword3"); /*Schwert */
[4Ø6] 1986:  load(texture1Ø,"sword6"); /*Schwert */
[N16] 1987:  load(texture11,"skell2"); /* Monster */
[5H5] 1988:  load(texture12,"mauer1"); /* Mauer */
[Ø26] 1989:  load(texture13,"key1"); /* Schüssel*/
[VE5] 199Ø:  load(ttexture1,"mgirl7"); /* girl*/
[L35] 1991:  load(ttexture2,"bild"); /* girl*/
[BB1] 1992:  ausweg:
[S93] 1993:  loadit=firstpage();
[YE3] 1994:  /* Bildschirmaufbau */
[D82] 1995:  display(OFF);
[X43] 1996:  loadbg("bg3");  /*bg3*/
[ZY2] 1997:  /* Rahmen Fenster */
[MS4] 1998:  drawbox(23+7,19+5,155+7,12Ø+9,Ø,1);
[KR4] 1999:  drawbox(18Ø+7,19+5,3Ø5+7,12Ø+5,Ø,1);
[6R4] 2ØØØ:  drawbox(23+5,134+5,3ØØ+7,186+5,Ø,1);
[Z93] 2ØØ1:  c.r=.7; c.g=.5; c.b=.Ø;
[ØT3] 2ØØ2:  col=interpret_col(&c);
[124] 2ØØ3:  drawbox(23,19,155,12Ø,col,1);
[F74] 2ØØ4:  drawbox(174,19,3Ø4,12Ø,col,1);
[8A3] 2ØØ5:  c.r=.8; c.g=.6; c.b=.1;
[ØT3] 2ØØ6:  col=interpret_col(&c);
[G93] 2ØØ7:  c.r=.4; c.g=.2; c.b=.1;
[224] 2ØØ8:  col2=interpret_col(&c);
[W24] 2ØØ9:  mkfr(29,19,149,12Ø,col2,col);
[2Z3] 2Ø1Ø:  mkfr(25,17,153,122,col,col2);
[LA4] 2Ø11:  mkfr(173,18,3Ø5,121,col,col2);
[314] 2Ø12:  mkfr(28Ø,9Ø,282,92,col,col2);
[484] 2Ø13:  loadbild("buttons",175,2Ø,1);
[GY3] 2Ø14:  loadbild("self3",24Ø,2Ø,1);
[UC4] 2Ø15:  loadpart("rolle",23,128,Ø,32);
[7L4] 2Ø16:  loadpart("rolle",54,128,1Ø,44);
[ZL4] 2Ø17:  loadpart("rolle",97,128,1Ø,44);
[VK4] 2Ø18:  loadpart("rolle",14Ø,128,1Ø,44);
[EL4] 2Ø19:  loadpart("rolle",183,128,1Ø,44);
[WK4] 2Ø2Ø:  loadpart("rolle",226,128,1Ø,44);
[UM4] 2Ø21:  loadpart("rolle",269,128,29,36);
[WE2] 2Ø22:  showsch(show);
[DH1] 2Ø23:  if(loadit)
[E92] 2Ø24:   {display(ON);
[5Z2] 2Ø25:    if(loadall(&spfld))
[BVØ] 2Ø26:     {i=Ø;
[ZY1] 2Ø27:     oldstufe=Ø;
[Y46] 2Ø28:     printinbox("Erfolgreich geladen.\n"
[5FØ] 2Ø29:     ,"");
[FV1] 2Ø3Ø:     fertig=Ø;}
[ZRØ] 2Ø31:    else
[NV5] 2Ø32:     {printinbox("Fehler beim Laden.\n"
[5FØ] 2Ø33:      ,"");
[DL2] 2Ø34:      ich.selected=-1;
[WE2] 2Ø35:      showsch(show);
[E51] 2Ø36:      wait(2);
[9S2] 2Ø37:      goto ausweg;}}
[UN4] 2Ø38:  mouse_area(174<<1,19,3Ø4<<1,12Ø);
[WE2] 2Ø39:  showsch(show);
[1B2] 2Ø4Ø:  /* Spielstart */
[Z46] 2Ø41:  showwelt(ich.posx,ich.posy,ich.look,
[N62] 2Ø42:  width,&spfld);
[4T1] 2Ø43:  if(!loadit)
[AP6] 2Ø44: printinbox("Du erwachst wieder ...\n","");
[L42] 2Ø45:  gotoxy(37,12);
[UA6] 2Ø46:  print(itoa(oldstufe+1,help,1Ø),15<<8¦22);
[D22] 2Ø47:  display(ON);
[6K2] 2Ø48:  while(fertig!=1)
[624] 2Ø49:   {/* Steck die Waffe weg */
[MX1] 2Ø5Ø:    ich.waffe=Ø;
[F62] 2Ø51:    mouse_key(Ø);
[443] 2Ø52:    mouse_move(mx,my);
[BA2] 2Ø53:    mouse_show();
[DL2] 2Ø54:    ich.selected=-1;
[ØG1] 2Ø55:    do{mp=Ø;
[VN5] 2Ø56:     if(((ich.status&TOT) ¦¦ich.le<=Ø)
[AD1] 2Ø57:      &&!fertig)
[V23] 2Ø58:      {ich.status=TOT;
[Z46] 2Ø59:       showwelt(ich.posx,ich.posy,ich.look,
[N62] 2Ø6Ø:        width,&spfld);
[HV1] 2Ø61:       fertig=2;}
```

```
[4F1] 2Ø62:       while(!mp)
[WX2] 2Ø63:        {if(mouse_key(Ø))
[5TØ] 2Ø64:          mp=1;
[VH2] 2Ø65:         if(mouse_key(1))
[6TØ] 2Ø66:          mp=2;
[T43] 2Ø67:         mouse_pos(&mx,&my);
[CF1] 2Ø68:         mx=mx>>1;
[3G3] 2Ø69:         if(!(ich.status&TOT))
[3Z1] 2Ø7Ø:          {animate();
[HS4] 2Ø71:           showwelt(ich.posx,ich.posy,
[TX3] 2Ø72:            ich.look,width,&spfld);
[ØX2] 2Ø73:           if(ich.status&TOT)
[HS4] 2Ø74:            showwelt(ich.posx,ich.posy,
[G94] 2Ø75:             ich.look,width,&spfld);}}
[FF3] 2Ø76:       while(mouse_press());
[M62] 2Ø77:       mouse_hide();
[Z81] 2Ø78:       if(mp==1)
[XW2] 2Ø79:       {switch(schelp=
[3C4] 2Ø8Ø:        testsch(mx,my,first,&sch))
[Y71] 2Ø81:        {case 1:
[S41] 2Ø82:         case 2:
[V41] 2Ø83:         case 3:
[Y41] 2Ø84:         case 4:
[151] 2Ø85:         case 5:
[451] 2Ø86:         case 6:
[751] 2Ø87:         case 7:
[A51] 2Ø88:         case 8:
[D51] 2Ø89:         case 9:
[3G3] 2Ø9Ø:         if(!(ich.status&TOT))
[XZ5] 2Ø91:          {dowith(findpos(ich.posx,ich.posy
[845] 2Ø92:           ,ich.look),bef[schelp-1],&spfld,
[YVØ] 2Ø93:           &ich);
[HS4] 2Ø94:           showwelt(ich.posx,ich.posy,
[TX3] 2Ø95:           ich.look,width,&spfld);
[WE2] 2Ø96:           showsch(show);
[7V2] 2Ø97:           ich.selected=-1;}
[PØ1] 2Ø98:          break;
[751] 2Ø99:         case 1Ø:
[B51] 21ØØ:         case 11:
[F51] 21Ø1:         case 12:
[L51] 21Ø2:         case 13:
[R51] 21Ø3:         case 14:
[V51] 21Ø4:         case 15:
[Z51] 21Ø5:         case 16:
[361] 21Ø6:         case 17:
[3G3] 21Ø7:         if(!(ich.status&TOT))
[R24] 21Ø8:          {ich.selected=schelp-1Ø;
[WE2] 21Ø9:          showsch(show);
[DX4] 211Ø:          testsch(mx,my,first,&sch);}
[PØ1] 2111:          break;
[761] 2112:        case 18:
[7S4] 2113:         printinbox("Spiel wirklich "
[7S2] 2114:         "beenden ?(J/N)","");
[DN2] 2115:         help[Ø]=getch();
[AR1] 2116:         help[1]=Ø;
[A82] 2117:         strupr(help);
[EF2] 2118:         if(help[Ø]=='J')
[TF1] 2119:          fertig=1;
[WE2] 212Ø:         showsch(show);
[PØ1] 2121:         break;
[B61] 2122:        case 19:
[ZY1] 2123:         oldstufe=Ø;
[5Z2] 2124:         if(loadall(&spfld))
[BVØ] 2125:          {i=Ø;
[D94] 2126:           printinbox("Erfolgreich "
[SE2] 2127:           "geladen.\n","");
[L42] 2128:           gotoxy(37,12);
[YT4] 2129:           print(itoa(oldstufe+1,help,1Ø)
[8S1] 213Ø:           ,15<<8¦22);
[FV1] 2131:           fertig=Ø;}
[ZRØ] 2132:         else
[L34] 2133:          printinbox("Fehler beim "
[R32] 2134:          "Laden.\n","");
[DL2] 2135:         ich.selected=-1;
[WE2] 2136:         showsch(show);
[HS4] 2137:         showwelt(ich.posx,ich.posy,
[TX3] 2138:         ich.look,width,&spfld);
[PØ1] 2139:         break;
[A51] 214Ø:        case 2Ø:
[LØ3] 2141:         if(saveall(&spfld))
[D94] 2142:          printinbox("Erfolgreich "
[Ø83] 2143:          "gespeichert.\n","");
[ZRØ] 2144:         else
[L34] 2145:          printinbox("Fehler beim "
[3X2] 2146:          "Speichern.\n","");
[DL2] 2147:         ich.selected=-1;
[WE2] 2148:         showsch(show);
[PØ1] 2149:         break;
[8E1] 215Ø:        default:
[DL2] 2151:          ich.selected=-1;
[WE2] 2152:          showsch(show);
[M82] 2153:          ichinfo(&ich);
[AG1] 2154:          break;}
[BA2] 2155:       mouse_show();
[3G3] 2156:       if(!(ich.status&TOT))
[3Z1] 2157:        {animate();
[HS4] 2158:         showwelt(ich.posx,ich.posy,
[G94] 2159:         ich.look,width,&spfld);}}
[VW3] 216Ø:     }while(mp!=2&& fertig!=1);
```

```
[VU4] 2161:     if(!(ich.status&TOT)&&!fertig)
[8G3] 2162:      {mouse_pos(&mx,&my);
[WE2] 2163:       showsch(show);
[FT4] 2164:       askkey(&(ich.posx),&(ich.posy)
[2L4] 2165:        ,&(ich.look),&spfld,width);}}
[MW2] 2166:  setgraphik(OFF);}
```

Listing 5. »Mask.c« enthält die Kernroutinen für das Rollenspiel.

```
[MF5]     1: FE Ø1 Ø1 Ø3 FF FE Ø1 Ø8 Ø3 Ø3 Ø3 ØØ FF ØØ Ø3 ØB
[XH5]     2: Ø7 FA 2A D8 FC Ø8 Ø1 Ø8 Ø8 Ø3 Ø6 FE Ø8 ØØ Ø3 49
[PB5]     3: Ø4 4B FD Ø1 Ø2 Ø2 Ø3 44 Ø4 45 Ø5 Ø9 Ø4 Ø2 FF Ø2
[ZE5]     4: Ø7 66 Ø6 65 Ø3 57 Ø6 6E Ø3 7A FD ØØ Ø8 FF Ø5 56
[HA5]     5: Ø3 85 Ø4 81 FF Ø1 Ø5 81 Ø6 93 Ø8 Ø7 Ø5 9C Ø3 54
[GN5]     6: Ø4 8F Ø5 7A Ø4 4E Ø9 B6 FD ØF ØF ØF Ø4 BF Ø4 C6
[ZG5]     7: Ø7 C3 Ø5 BF Ø5 83 FF Ø8 Ø7 5E Ø5 71 Ø3 D5 Ø9 C2
[1P5]     8: ØB C5 FF 16 Ø6 ØØ Ø8 FF Ø4 ØE FF 3A Ø5 F8 Ø3 47
[DG5]     9: Ø6 74 FE Ø2 Ø1 Ø3 A7 Ø7 F4 ØD Ø4 ØC Ø7 ØF 46 Ø8
[HL5]    1Ø: D3 Ø4 1E Ø4 23 Ø4 25 ØD 32 ØB 3B ØE FF FA 3A ØF
[GK5]    11: 16 ØF 3A 16 Ø3 16 Ø3 65 FF 2C Ø4 9D Ø6 61 Ø8 FD
[XD5]    12: Ø3 91 Ø9 86 Ø4 91 Ø6 73 Ø3 BF Ø4 A8 Ø3 C8 Ø3 CC
[RM5]    13: FF 3A Ø3 93 FF 16 Ø5 59 Ø3 1A Ø4 9F FE Ø1 2C Ø7
[1L5]    14: 67 Ø4 CC Ø5 8F Ø4 A7 Ø4 F4 Ø5 F1 Ø3 FF Ø4 13 Ø3
[XS5]    15: EF Ø7 F1 FF 3A Ø3 D4 Ø5 56 Ø5 9D Ø3 E1 FD Ø2 2C
[LB5]    16: 2C Ø5 BF Ø4 Ø3 Ø5 Ø5 Ø3 ØB Ø6 37 Ø8 36 Ø6 39 Ø3
[TF5]    17: 35 Ø9 3E Ø4 D8 Ø6 5C FD Ø8 Ø1 2C Ø3 64 FE Ø8 16
[LK5]    18: Ø4 A6 12 6D FF Ø2 Ø9 78 Ø4 89 Ø9 7F Ø4 58 FE Ø2
[CC5]    19: Ø1 Ø4 DE Ø3 2Ø FC Ø2 Ø1 Ø2 Ø2 Ø5 68 Ø8 6A Ø5 8C
[YL5]    2Ø: Ø5 B8 FD Ø2 ØF Ø8 Ø3 BC FE Ø8 Ø8 Ø4 B5 FF Ø2 Ø6
[FT5]    21: C6 Ø5 AB Ø3 D7 Ø3 C4 FE Ø1 ØØ Ø5 DD Ø3 5A FD 2C
[VH5]    22: 2C Ø2 Ø6 D4 Ø3 D8 Ø8 B9 Ø3 D1 Ø4 8A FF Ø8 Ø3 CØ
[4K5]    23: FE Ø2 ØF Ø3 E7 Ø3 Ø3 Ø4 Ø4 Ø7 EB Ø4 A5 Ø4 1D FF
[WF5]    24: Ø8 Ø3 E2 Ø4 EF FE Ø2 Ø2 Ø3 E8 Ø5 25 Ø4 FD Ø3 D3
[FN5]    25: Ø5 FF FF Ø1 Ø4 C2 Ø5 C4 Ø5 43 FD 2C ØF 2C Ø3 E6
[FH5]    26: FD 2C Ø2 2C Ø4 ØC FE Ø1 Ø1 Ø7 1D Ø4 1A Ø3 A4 Ø3
[WE5]    27: 4F Ø3 Ø2 FF Ø8 Ø3 56 Ø3 55 Ø4 AD Ø3 76 Ø4 27 Ø4
[5H5]    28: ØD Ø7 86 Ø4 66 FF Ø1 Ø3 63 Ø4 A4 Ø3 95 Ø4 5D FE
[9L5]    29: Ø1 ØØ Ø7 7D Ø7 88 FF 2C Ø5 95 FF Ø1 Ø3 7C Ø3 4E
[WD5]    3Ø: Ø6 83 Ø7 A5 Ø3 C4 Ø4 92 Ø3 CE Ø4 D3 Ø4 DØ Ø6 9D
[1M5]    31: Ø3 DB Ø9 93 Ø5 EE ØA EA Ø5 D8 Ø5 CD Ø6 D8 Ø7 ØD
[CK5]    32: Ø9 ØA Ø9 DE Ø8 13 Ø6 Ø3 Ø5 D5 Ø5 CF 1Ø ØE ØE 3F
[NG5]    33: Ø3 1E Ø5 5F Ø4 5C Ø8 46 Ø5 9B ØA 4C ØB 75 ØC 59
[7N5]    34: ØE 96 ØE 8E Ø7 A5 Ø6 55 1E 95 13 C8 ØE BD ØA 5C
[SP5]    35: ØA F3 Ø9 BC 21 BE Ø8 16 F7 Ø1 FF FF Ø1 FF Ø8 Ø8
[FB5]    36: ØØ Ø8 Ø3 4Ø Ø3 4B Ø4 53 Ø3 52 Ø4 4F Ø5 4C FE ØØ
[5G5]    37: ØØ Ø3 46 Ø6 7D Ø3 67 FF FF 11 ØB Ø6 83 Ø6 54 Ø4
[UH5]    38: 58 Ø5 53 ØE 8D Ø5 58 Ø6 5E Ø8 A1 Ø4 7B Ø3 7E Ø4
[D16]    39: 6A FE FF FF Ø5 B1 FD ØF Ø8 ØF Ø4 CE ØC CD Ø6 DE
[YK5]    4Ø: ØB D9 Ø4 E5 Ø3 FØ FF Ø8 Ø3 49 Ø4 B9 Ø6 79 FF Ø2
[8H5]    41: Ø4 C9 Ø5 Ø9 FF 16 Ø4 ØF ØB ØE Ø6 14 Ø8 1E Ø9 2C
[UF5]    42: Ø4 F1 Ø7 F9 Ø4 Ø2 Ø3 Ø5 Ø9 ØB Ø9 2E ØF 28 11 5F
[XE5]    43: Ø6 38 Ø3 4Ø FF Ø8 Ø6 45 Ø9 6D Ø3 88 Ø3 8F ØD ØD
[UG5]    44: ØA 6C Ø8 AC Ø5 77 FF ØØ Ø4 7E Ø4 Ø5 Ø3 86 Ø9 A5
[9S5]    45: Ø4 96 FF 3A Ø5 34 FD ØF 3A 3A Ø7 8F Ø5 D1 ØB 9B
[FL5]    46: FD 3A ØF 3A Ø6 36 Ø5 7F Ø6 85 Ø7 BØ Ø4 Ø6 Ø3 DE
[BK5]    47: Ø4 D9 Ø6 48 Ø5 1C Ø8 1A Ø4 CF Ø5 DB Ø5 37 FF Ø8
[8D5]    48: Ø5 3E Ø7 Ø3 Ø9 17 Ø4 36 Ø7 F5 Ø4 5D FF Ø2 Ø5 53
[5G5]    49: Ø5 92 Ø5 5Ø FF 3A Ø3 72 Ø8 4B Ø5 7E FF Ø1 Ø4 45
[FE5]    5Ø: Ø4 6A F9 3A ØF Ø2 ØF Ø2 Ø2 Ø2 Ø8 76 Ø3 91 Ø6 4A
[KD5]    51: Ø4 9Ø Ø3 A4 Ø3 92 Ø3 74 Ø9 75 Ø3 7C Ø4 7D FF ØØ
[ØH5]    52: Ø3 42 Ø7 64 FF 3A Ø4 A8 Ø4 A5 Ø9 9F Ø5 B8 Ø4 D2
[PE5]    53: Ø5 A5 Ø6 B1 Ø4 C4 Ø3 E7 Ø3 C3 Ø7 BC Ø6 83 Ø9 AØ
[8L5]    54: Ø5 D3 FF Ø8 Ø4 CF Ø3 E2 Ø6 F9 Ø4 1F Ø6 DB Ø6 A8
[GB5]    55: Ø4 2E Ø7 2Ø FB ØØ ØØ ØØ Ø8 Ø1 Ø3 15 Ø5 C3 FE Ø8
[4D5]    56: Ø2 Ø4 25 FF Ø8 Ø3 12 Ø3 4D Ø4 46 Ø4 48 Ø5 5C Ø7
[6N5]    57: 47 Ø4 66 FE Ø2 2C Ø5 32 FC 2C 2C Ø1 2C Ø4 3F FE
[CN5]    58: Ø1 ØØ Ø3 6F FD Ø1 Ø8 2C Ø3 6D FF 2C Ø5 5B Ø6 5E
[4B5]    59: Ø4 95 Ø6 23 Ø5 A1 Ø3 78 Ø4 9F Ø3 4E Ø4 87 Ø4 75
[WH5]    6Ø: Ø4 AØ FF Ø1 Ø3 7D FD Ø1 Ø1 ØØ Ø3 1B Ø7 8D Ø6 CD
[MU5]    61: FF 2C Ø7 C6 Ø3 D5 Ø3 A8 FF 2C Ø3 DF Ø4 BØ FD 2C
[VK5]    62: Ø2 Ø2 Ø3 7B FF 2C Ø4 77 Ø3 DB Ø4 F4 Ø6 3D FF Ø1
[9F5]    63: Ø3 41 Ø4 D9 FE Ø1 Ø1 Ø5 F2 Ø4 D4 Ø3 11 Ø5 F3 Ø4
[VH5]    64: ED Ø4 79 Ø6 8C FB 2C 2C Ø2 2C Ø1 Ø4 31 Ø3 EB Ø3
[UD5]    65: 2F Ø3 BB Ø3 Ø2 Ø3 C2 Ø3 81 FF Ø2 Ø4 34 FE Ø1 Ø2
[SA5]    66: Ø3 4B Ø4 2B Ø3 24 Ø4 86 Ø3 29 Ø5 4D Ø3 56 Ø3 14
[DE5]    67: Ø3 5E Ø4 6B Ø3 68 Ø4 72 Ø5 6B Ø6 3C Ø5 81 Ø4 6E
[6F5]    68: Ø4 36 Ø7 87 Ø6 6C Ø6 99 Ø3 1B Ø4 C4 Ø6 8D Ø8 8D
[FD5]    69: Ø3 3B Ø5 7B Ø8 BE Ø8 76 Ø7 9Ø Ø6 7Ø Ø7 69 13 E2
[TN5]    7Ø: ØA BD Ø7 FF Ø6 F3 Ø8 DC Ø7 BØ FF Ø1 Ø4 Ø6 12 F5
[WK5]    71: Ø3 1D ØA 2B Ø8 7F ØF FB ØC 1F 13 41 14 2B 37 89
[CU5]    72: 4Ø 8Ø FC FF FF Ø8 FF Ø5 Ø4 Ø3 ØØ Ø8 ØC 12 B5 FF
[8K5]    73: Ø2 Ø4 Ø1 FF Ø8 Ø4 2A 13 3Ø Ø6 2E ØB 2E Ø3 4A FE
[UE5]    74: Ø2 Ø2 Ø6 1E Ø5 7F FC Ø2 Ø2 Ø8 Ø2 Ø6 4F FF ØF Ø3
[PL5]    75: 6D FF ØF Ø9 71 FD ØF Ø8 Ø1 Ø7 4E Ø3 6F 11 8Ø FF
[1F5]    76: Ø8 Ø3 6Ø Ø4 9D FF ØØ Ø8 66 Ø3 7B 24 AE Ø5 9Ø Ø5
[RU5]    77: 91 FF Ø2 Ø3 5F Ø3 9F Ø6 7F Ø5 CE FF 16 1Ø EF FF
[ØR5]    78: 3A 11 EF FE 16 3A Ø6 6F Ø4 99 Ø5 9E FE Ø2 Ø1 Ø4
[2V5]    79: 7E ØB EC Ø8 29 11 EE FD 3A 3A 3A Ø3 3B FC ØF ØF
[7E5]    8Ø: 3A 3A Ø3 34 Ø3 E3 Ø5 9F Ø4 64 Ø3 5A Ø6 EB Ø3 51
[NB5]    81: Ø3 52 Ø4 3B Ø7 74 Ø7 75 Ø7 73 Ø4 3C Ø3 4D Ø3 56
[LP5]    82: FD 3A Ø8 ØF Ø3 5B Ø3 5F FE Ø1 Ø1 Ø5 A6 Ø9 34 FD
[3E5]    83: ØF 3A ØF Ø4 ØØ Ø5 B8 Ø6 93 Ø7 C3 Ø3 C2 Ø5 72 Ø3
[9L5]    84: BE Ø5 55 Ø7 1A Ø3 AØ Ø4 A2 Ø6 A8 Ø9 C4 Ø8 F6 Ø5
```

Listing 6. bg3.pak ist eine gepackte Grafikdatei.

```
[EP5]    85: F4 Ø7 BF Ø3 B4 Ø5 BØ Ø7 F4 FE Ø8 ØF Ø8 DC FF Ø2
[VD5]    86: Ø3 63 Ø6 6A Ø6 Ø6 Ø3 1Ø Ø5 35 FB Ø8 ØF Ø2 Ø2 ØF
[1F5]    87: Ø5 3D Ø3 45 FF Ø2 Ø8 Ø3 Ø6 16 Ø4 D9 Ø3 5C Ø4 5E
[5G5]    88: FD Ø1 Ø2 Ø1 Ø4 43 Ø8 EC ØC F5 Ø6 42 FF Ø2 Ø3 83
[TC5]    89: Ø8 87 Ø9 84 Ø3 5A FE Ø2 Ø2 Ø5 5D Ø3 E2 Ø4 65 Ø7
[TG5]    9Ø: 9Ø Ø3 9B Ø4 A9 Ø4 A7 Ø3 1A Ø4 AC Ø4 B2 Ø4 BE Ø3
[SK5]    91: 3C Ø3 56 Ø4 25 Ø3 E6 FF Ø1 Ø3 CC Ø3 A4 Ø6 9F Ø5
[SN5]    92: E5 Ø4 B5 Ø4 D3 Ø3 ED Ø4 C8 Ø3 E5 Ø7 FB Ø4 D4 Ø5
[GM5]    93: FF Ø4 F9 Ø5 AF Ø5 F9 Ø3 13 Ø7 9D Ø3 1C Ø6 7F Ø5
[VF5]    94: 15 Ø5 F7 Ø4 C3 FF Ø2 Ø5 98 Ø4 32 Ø4 1A Ø5 EC Ø5
[AG5]    95: E9 Ø5 47 Ø3 25 Ø8 1B FF Ø2 Ø4 A4 Ø7 4F Ø5 17 Ø5
[ZC5]    96: Ø5 Ø4 E5 Ø5 43 Ø5 59 Ø5 5A Ø5 64 Ø6 59 Ø8 87 Ø6
[6K5]    97: 1F Ø4 9B FF Ø1 Ø5 44 Ø6 44 Ø7 42 Ø6 7E Ø4 CØ Ø9
[TR5]    98: 8F Ø6 8D Ø7 A3 Ø7 DC FF ØØ Ø3 CF Ø6 CD Ø9 81 Ø8
[3K5]    99: CØ Ø7 8A Ø6 CB Ø6 DF Ø4 FE Ø5 ØB Ø4 ØD Ø3 1C Ø6
[1G5]   1ØØ: EØ Ø5 FF Ø5 CF FE Ø1 Ø2 Ø7 21 Ø4 F9 Ø3 36 Ø8 15
[GK5]   1Ø1: ØF 48 Ø5 DB ØA 14 ØC D8 Ø9 16 Ø8 77 ØA 5C Ø9 63
[5L5]   1Ø2: Ø5 83 Ø4 85 Ø7 5B ØE D7 ØA 16 ØF 6B 18 CD Ø6 BC
[VW1]   1Ø3: Ø4 99 Ø8 99 Ø8 C1
```

Listing 6. bg3.pak ist eine gepackte Grafikdatei.

```
[ØE5]     1: Ø8 ØØ FE 48 48 Ø8 ØA F9 2B ØØ ØØ 24 48 ØØ 24 Ø3
[AE5]     2: Ø6 Ø8 Ø1 ØE 24 Ø3 12 FC 48 2B ØØ 2B Ø3 38 Ø7 23
[ØE5]     3: FF 48 Ø3 43 Ø4 21 Ø7 Ø8 Ø3 4B Ø6 4B Ø6 45 Ø6 1F
[9H5]     4: Ø4 5C Ø4 5F Ø3 6B FF 2B Ø4 36 Ø8 5C Ø6 65 Ø5 51
[XL5]     5: Ø5 41 Ø7 4Ø FF 24 Ø5 88 FF CA Ø3 9C Ø4 86 FF 74
[NE5]     6: Ø3 A4 Ø4 92 Ø4 AB FF 2B Ø5 91 Ø5 B5 Ø8 2Ø Ø4 84
[1S5]     7: ØA 5C Ø7 BA Ø3 D5 Ø4 9C FE DØ DØ Ø3 A5 Ø4 E4 ØA
[SH5]     8: A5 Ø8 22 Ø7 F1 Ø6 EF FC 48 ØØ 81 56 Ø8 F6 Ø5 52
[7S5]     9: FF 24 Ø5 9B FF CA Ø6 E2 Ø6 25 Ø9 A7 FE 48 2B ØB
[KP5]    1Ø: B6 Ø9 F1 Ø7 8F Ø7 81 FC 48 ØØ DØ CA Ø3 EØ ØA 21
[VF5]    11: Ø6 28 FF 74 Ø3 6E Ø7 4E ØA 43 F9 ØØ ØØ 81 ØØ 81
[6L5]    12: 81 56 Ø3 8A FD ØØ ØØ 2B Ø9 49 Ø3 DE Ø5 9C ØB 79
[HF5]    13: Ø4 E8 Ø3 AF Ø4 E1 Ø5 41 FF 48 Ø4 9Ø Ø3 8D Ø4 89
[FM5]    14: FD ØØ 56 56 Ø5 9Ø FF 56 Ø7 ØØ FE ØØ ØØ Ø4 9E FE
[CR5]    15: CA CA Ø5 D3 Ø6 38 Ø5 A9 Ø3 EF Ø5 AA Ø5 E5 FF 2B
[DG5]    16: Ø3 FD Ø4 8A Ø4 C8 Ø4 Ø5 Ø5 CD FE 56 2B Ø3 35 Ø4
[KT5]    17: 78 Ø3 1C FF CA Ø6 94 ØA E6 ØB A4 Ø5 D2 Ø3 3F FD
[XD5]    18: 2B 56 8F Ø3 45 FD 6B 16 2B Ø5 ØD Ø4 Ø7 Ø4 3E Ø4
[DP5]    19: BA Ø6 77 Ø5 5F Ø8 3B ØC 2E Ø6 FA F7 56 2B 56 16
[3K5]    2Ø: 6B 16 16 8F 16 Ø6 CB Ø4 81 FF 2B Ø3 8Ø Ø4 8E ØA
[RR5]    21: 6A Ø8 25 ØE E3 FF 74 Ø3 7E FE 2B 2B Ø3 12 FA 6B
[UG5]    22: 16 56 6B 16 ØØ Ø3 46 FF 16 Ø3 41 FC 47 47 47 16
[9G5]    23: Ø8 ØA Ø6 AF Ø5 55 Ø4 E3 Ø8 22 Ø8 B1 FF 74 Ø4 41
[KF5]    24: Ø3 C2 Ø6 C6 FF 16 Ø3 85 FE ØØ ØØ Ø3 86 FF 16 Ø3
[2K5]    25: 4A Ø3 94 Ø8 AA Ø8 DE 11 2B Ø4 5B Ø3 4B FC 56 ØØ
[MG5]    26: 47 16 Ø5 45 Ø3 D4 FB ØØ ØØ 16 6B 6B Ø5 8E Ø4 DA
[ZC5]    27: Ø5 E5 FD ØØ ØØ 24 Ø3 62 Ø4 F8 Ø9 37 ØA EA Ø5 58
[ØR5]    28: FC 56 1D 16 ØØ Ø5 86 FE 1D 16 Ø4 4F FF 16 Ø3 4E
[HH5]    29: Ø9 5A Ø4 6A Ø3 63 FF 24 Ø5 99 FF 24 Ø4 AD Ø3 65
[HF5]    3Ø: FE 48 48 Ø5 65 Ø3 A9 Ø3 CØ Ø3 84 Ø4 C5 Ø3 8A Ø8
[5R5]    31: C7 Ø3 8F FF 56 Ø5 59 Ø8 DD Ø7 B9 Ø8 B5 Ø4 AC Ø7
[PC5]    32: F8 Ø3 17 FE 56 56 Ø4 13 Ø3 D5 ØA Ø6 Ø4 97 Ø4 42
[LG5]    33: ØB DE Ø9 28 Ø4 AE Ø6 64 Ø4 F7 FE 2B 56 Ø3 ØØ Ø5
[DC5]    34: D7 Ø4 Ø1 Ø8 46 Ø4 18 Ø5 1B FC 2B 47 16 2B Ø9 4Ø
[3K5]    35: FD 56 23 16 Ø6 4E Ø5 AC FE 24 48 Ø4 A7 Ø3 62 Ø3
[9S5]    36: 82 FF AC Ø3 86 FF 23 Ø3 8A FC 81 AC AC 81 Ø3 8E
[DM5]    37: FE AC 6B Ø3 6E FF 2B Ø5 1A FF 16 Ø4 6E Ø4 92 Ø4
[RH5]    38: A5 FE 16 16 Ø4 A8 FC ØØ 81 ØØ AC Ø3 7B Ø3 B9 Ø6
[AF5]    39: FD Ø5 8F F8 81 47 47 16 16 23 23 16 Ø3 C6 Ø3 DØ
[KP5]    4Ø: Ø4 6D FE 2B 2B Ø5 DC Ø3 C1 FF 81 Ø4 D2 Ø4 E8 Ø7
[RG5]    41: E7 Ø3 AF Ø3 A3 FB 24 24 56 ØØ 2B Ø4 7F FF 81 Ø3
[LL5]    42: F7 FA 81 6B 6B 16 16 81 Ø3 ØD FB 81 81 23 23 3A
[UG5]    43: Ø3 11 Ø3 CD FF 2B Ø3 12 FD ØØ 23 16 Ø5 2Ø Ø3 6C
[DF5]    44: FF 23 Ø3 1Ø FF 16 Ø4 15 Ø3 ØF Ø3 35 Ø3 Ø2 Ø3 FE
[KR5]    45: Ø3 3F FF AC Ø3 38 F5 81 81 8F 8F 16 56 81 6B 8F
[PG5]    46: 81 56 Ø3 Ø9 FF 6B Ø3 53 ØA 53 FF 6B Ø4 62 Ø3 61
[ZC5]    47: FF 23 Ø6 5E Ø3 D7 Ø3 Ø3 44 Ø3 75 Ø4 BD Ø3 81
[FL5]    48: Ø4 79 Ø4 6F FE 2B 81 Ø3 89 FF 8F Ø3 48 FF 81 Ø3
[HE5]    49: 49 FE 8F 16 Ø4 91 FF ØØ Ø3 99 Ø7 A1 Ø5 A5 Ø4 47
[NH5]    5Ø: Ø4 36 Ø6 76 Ø3 3D Ø4 DE Ø4 Ø3 Ø4 CC Ø3 86 FD 16
[3H5]    51: 81 1D Ø4 ØD Ø3 19 FF 6B Ø3 23 FE 16 ØØ Ø8 1F Ø3
[8M5]    52: E5 Ø3 CD Ø4 CA Ø4 31 FE 56 56 Ø3 81 Ø6 7A FF ØØ
[WK5]    53: Ø3 FC Ø4 F7 FD 16 16 1D Ø3 ØA FC 16 16 56 3A Ø4
[PG5]    54: ØB FB 23 1D 2B 1D 16 Ø4 11 FD 3A 1D 16 Ø8 2Ø Ø3
[4F5]    55: 18 Ø3 2B Ø3 13 FF 81 Ø4 8Ø FF 2B Ø4 F6 Ø5 FD Ø4
[BC5]    56: Ø1 Ø3 3A Ø3 D2 Ø3 4B Ø3 38 Ø3 4C FF ØØ Ø3 4D Ø3
[SH5]    57: 56 FD 3A 3A 3A Ø4 48 FF 56 Ø5 62 Ø4 57 Ø4 66 Ø3
[HB5]    58: 71 Ø4 45 Ø4 37 Ø5 BE Ø3 4Ø Ø6 F5 Ø3 74 Ø5 34 Ø5
[ØK5]    59: 8E Ø5 96 Ø5 7D Ø5 Ø3 Ø4 8A Ø5 78 Ø6 76 Ø6 37 FF
[CK5]    6Ø: 56 Ø7 9B FE 56 ØØ Ø5 39 Ø5 F7 Ø6 F7 Ø5 CD Ø8 99
[UF5]    61: Ø6 A5 Ø3 44 Ø3 EA Ø6 A3 Ø9 AC FF ØØ Ø4 FC Ø4 C1
[1B5]    62: Ø5 98 Ø5 8A Ø6 97 Ø4 Ø9 Ø7 BC Ø6 Ø5 Ø6 77 Ø6 14
[AF5]    63: Ø4 2A Ø6 89 Ø8 BD Ø5 26 Ø4 FF Ø6 3C Ø5 2C ØD 58
[R95]    64: Ø6 4B Ø5 16 Ø6 Ø7 Ø3 25 Ø7 3B Ø4 25 Ø3 C3 ØE 58
[PG5]    65: 1F 95 Ø4 69 Ø3 B5 Ø5 87 Ø6 B9 Ø6 C4 Ø9 AF FB ØØ
[SR5]    66: 8F 16 8F 16 Ø3 D5 Ø3 DD FF 16 Ø7 D6 11 A5 FF 2B
[BF5]    67: Ø3 84 Ø6 56 Ø4 FB ØF 87 FD 23 16 23 Ø3 E6 FE 23
[1K5]    68: 16 Ø4 18 FF 23 Ø3 23 11 E6 Ø3 49 ØA Ø8 Ø3 83 FF
[8H5]    69: 56 Ø3 79 Ø9 4E Ø7 15 FD 56 23 16 Ø3 5C FF 2B Ø5
[LM5]    7Ø: 1D Ø6 48 Ø5 B5 Ø4 76 Ø3 69 Ø9 49 FA ØØ 2B AC AC
[ET5]    71: 81 AC Ø3 76 FF 56 Ø5 7D Ø3 FA FE 56 81 Ø4 DE FE
[ZL5]    72: AC AC Ø3 99 Ø8 9D FF 81 Ø8 48 Ø4 71 Ø6 76 FF 81
```

```
[EN5]    73: Ø7 8C Ø3 BØ FD AC 81 81 Ø5 A8 Ø8 8F FF AC Ø3 65
[GR5]    74: FD 16 81 81 Ø5 23 FF 81 Ø6 EØ Ø6 A8 Ø7 CD FF AC
[7P5]    75: Ø3 B8 Ø3 EA Ø7 3E FF 2B Ø3 BA Ø4 E8 Ø6 FB Ø4 D3
[7E5]    76: FE 81 81 Ø3 D9 FD 56 56 16 Ø3 1D Ø4 16 Ø3 1C Ø4
[NH5]    77: C7 Ø5 EC Ø4 D1 FE AC 81 Ø4 F9 Ø9 3B Ø3 12 Ø7 Ø6
[GH5]    78: Ø3 4E FD 81 81 AC Ø3 85 FF 56 Ø4 47 Ø5 49 Ø4 D2
[9L5]    79: Ø6 57 Ø6 ØB Ø4 B2 Ø4 2A Ø5 4C Ø6 BE Ø4 65 FF 1D
[WL5]    8Ø: Ø3 88 FF 16 Ø4 8D FF 1D Ø3 19 FC 81 81 1D 16 Ø3
[N85]    81: 84 Ø7 63 Ø4 99 Ø4 61 Ø4 75 Ø4 41 Ø3 Ø5 Ø4 ØA Ø6
[UR5]    82: FC Ø5 6D FE 81 56 Ø3 D8 Ø3 C9 Ø4 BC Ø6 E6 FE 23
[UD5]    83: 16 Ø3 78 Ø6 Ø8 Ø6 5E Ø4 73 Ø5 B8 Ø8 71 Ø7 FC Ø3
[ZP5]    84: F9 Ø4 E5 FE 23 16 Ø4 E6 Ø4 ØD Ø3 D1 Ø6 Ø4 FE 2B
[EH5]    85: 56 Ø5 88 Ø5 86 FF 16 Ø4 1B Ø3 8C Ø5 EE Ø4 EØ Ø9
[EK5]    86: 3A Ø3 2C Ø4 76 EF 16 8F 8F 8F 16 2B 47 23 23 16
[ØH5]    87: 23 23 23 6B 16 56 6B Ø3 D8 Ø3 16 FF 16 Ø3 D7 Ø4
[6M5]    88: C7 Ø6 61 Ø4 2C Ø3 F7 Ø5 31 Ø7 F9 Ø4 7F Ø6 42 FB
[1M5]    89: 16 16 6B 16 8F Ø3 88 Ø3 8D FD 8F 16 56 Ø3 94 FB
[SG5]    9Ø: 2B 2B 8F 16 81 Ø3 97 Ø3 47 FE 16 2B Ø4 A2 Ø7 F6
[FG5]    91: Ø5 82 Ø5 2C Ø6 BB Ø6 75 Ø3 C5 FF 23 Ø3 8A FE ØØ
[R95]    92: ØØ Ø3 57 Ø5 54 Ø3 Ø3 Ø5 D1 Ø4 86 Ø3 17 Ø4 E3 Ø5
[HC5]    93: C6 Ø8 7Ø ØA 79 Ø4 35 Ø3 51 Ø3 67 Ø5 4D Ø4 65 Ø3
[YH5]    94: Ø6 Ø3 6A FF 6B Ø5 52 Ø5 Ø8 Ø4 22 Ø6 B8 ØB B7 Ø7
[NM5]    95: BC FF 56 Ø3 F8 Ø3 8E FE 16 16 Ø6 46 Ø4 4B Ø4 4C
[ME5]    96: FE 2B 2B Ø4 4F Ø6 5A Ø4 56 ØB 79 Ø3 33 Ø3 44 Ø5
[MM5]    97: F8 Ø3 Ø2 Ø6 7A ØC 88 Ø6 33 ØF 33 1Ø 8B Ø9 FB FF
[BL5]    98: ØØ Ø4 C4 Ø6 6F 18 9C 1B AØ Ø4 8Ø 1C A5 15 C9 Ø4
[UL5]    99: 75 Ø7 Ø2 Ø3 39 Ø3 43 FC 8F 8F 16 16 Ø3 3A FE 1D
[RB5]   1ØØ: 1D Ø3 4A Ø3 52 Ø3 73 Ø3 24 FE 23 16 Ø8 95 Ø3 C2
[RK5]   1Ø1: ØF 68 Ø4 83 Ø3 79 Ø3 64 F1 56 56 AC 81 AC B3 B3
[BD5]   1Ø2: 23 8F 16 23 23 23 1D 16 Ø3 84 FF 81 Ø7 54 Ø6 5C
[CK5]   1Ø3: Ø3 Ø1 Ø3 83 FE 81 AC Ø3 91 Ø4 AA Ø7 AA FF 81 Ø7
[3K5]   1Ø4: Ø3 Ø4 7F F7 81 81 81 8F B3 B3 8F 23 6B Ø3 8C Ø3
[ML5]   1Ø5: 9Ø Ø3 D2 Ø5 96 Ø4 5B FF 2B Ø4 A1 Ø3 B6 Ø3 A7 Ø4
[ZK5]   1Ø6: D2 FF 81 Ø4 C3 Ø4 EF Ø4 FØ Ø7 21 Ø3 77 Ø4 C5 F8
[BS5]   1Ø7: 16 8F 8F 6B 16 23 6B 16 Ø6 F3 FE 56 2B Ø3 D6 FF
[7E5]   1Ø8: 23 Ø3 8B Ø3 5Ø Ø6 E9 Ø3 94 FF 56 Ø3 29 Ø4 F2 Ø4
[KM5]   1Ø9: 2A Ø3 B7 FF 81 Ø7 BA Ø4 32 FB 6B 6B 16 16 23 Ø3
[WF5]   11Ø: 49 Ø3 1F Ø4 CD FE 81 81 Ø4 52 Ø4 58 FE 16 2B Ø4
[EE5]   111: 5C Ø3 51 Ø4 4D Ø6 62 Ø3 4A Ø4 2F Ø5 FA Ø5 3D Ø3
[4H5]   112: 34 FF 6B Ø3 74 F9 16 16 81 8F 8F 16 ØØ Ø4 49 FF
[HM5]   113: 81 Ø3 48 FF 16 Ø5 46 FC 2B 2B 16 16 Ø3 ØF Ø3 9C
[EP5]   114: Ø3 ØE Ø5 A2 FC 23 6B 6B 6B Ø3 84 Ø3 35 Ø6 BC FE
[RL5]   115: 2B ØØ Ø3 17 FF 1D Ø3 AF F8 2B 56 81 B3 B3 16 56
[GS5]   116: 8F Ø4 8E Ø4 8D Ø5 D1 FE 2B 2B Ø4 DC Ø3 DB Ø3 Ø9
[ZR5]   117: Ø3 E9 Ø3 CF FC B3 16 16 B3 Ø3 DF FD 2B 2B 56 Ø5
[TU5]   118: BF Ø4 81 Ø3 C5 FF 1D Ø4 F6 Ø5 8D FF B3 Ø3 F4 FF
[7D5]   119: 56 Ø3 91 Ø4 12 Ø3 CE Ø5 19 FF 56 Ø3 1E Ø3 ØE Ø3
[YD5]   12Ø: 28 Ø3 E5 Ø3 8C Ø3 Ø9 Ø3 Ø4 Ø4 F7 Ø6 ØØ FF 2B Ø3
[NK5]   121: Ø7 FF ØØ Ø3 CB Ø4 4E Ø3 E2 Ø4 EB Ø3 ØF FE 8F 16
[KR5]   122: Ø6 E9 FE 8F 23 Ø3 92 Ø3 4E FF 2B Ø3 6D Ø4 72 FF
[1R5]   123: 2B Ø3 4F FF 56 Ø3 39 Ø5 BD Ø4 7C FF 2B Ø3 8A Ø5
[NE5]   124: F6 Ø4 87 Ø3 8E Ø6 9Ø Ø3 17 FF ØØ Ø3 9B Ø5 92 Ø5
[9E5]   125: 8D Ø5 96 Ø3 BØ Ø3 86 Ø3 44 Ø6 8Ø Ø4 BA FF ØØ Ø6
[1M5]   126: 77 Ø3 C6 Ø4 43 Ø5 C5 Ø3 BE Ø6 BF Ø6 7D Ø3 D1 Ø6
[SG5]   127: D1 Ø5 3B Ø4 CF FF ØØ Ø4 7B Ø9 BD Ø4 FØ Ø5 37 Ø4
[MD5]   128: Ø5 Ø6 C7 Ø3 Ø3 Ø5 19 Ø3 1D Ø4 D4 Ø4 CA Ø6 27 Ø4
[GE5]   129: 25 Ø4 Ø4 Ø7 DB ØB 4Ø Ø3 ØD Ø5 1A Ø9 4E Ø3 5C Ø9
[K95]   13Ø: 44 Ø3 39 Ø4 69 ØF 38 1Ø 39 Ø4 54 Ø6 15 Ø5 57 62
[Y9Ø]   131: 9D
```

Listing 7. titel1.pak ist eine gepackte Grafikdatei.

```
[5R5]     1: FF 2B Ø8 FC FF 2B Ø6 Ø9 Ø3 1Ø FC 56 2B 56 56 Ø7
[6C5]     2: Ø9 ØF 1D Ø7 Ø3 FF Ø7 Ø5 34 Ø6 FB Ø5 ØØ Ø4 45 Ø3
[ZG5]     3: 16 Ø7 44 FF 56 Ø3 15 Ø3 53 ØC 59 Ø5 52 Ø7 2B FE
[SG5]     4: 2B Ø7 Ø3 67 Ø7 38 Ø7 3E FF 56 Ø4 82 Ø4 49 Ø4 6F
[NS5]     5: Ø3 74 FF 81 Ø6 6Ø FD AC AC 81 Ø5 9C FE 88 88 Ø5
[XF5]     6: 63 Ø7 ØD Ø3 76 FF 56 Ø3 73 Ø8 37 Ø3 84 Ø3 B8 FF
[NH5]     7: 16 32 C7 Ø3 B5 Ø5 BB Ø5 97 FE 16 Ø8 3Ø Ø8 FF 16
[1P5]     8: Ø3 B9 FE Ø7 56 Ø3 AD Ø5 64 FD 16 Ø8 ØF 2E 49 FE
[EL5]     9: 16 1D Ø3 FB FD Ø7 Ø7 2B Ø6 Ø1 Ø4 46 Ø6 BC Ø3 8F
[ØG5]    1Ø: FA ØD ØØ ØØ 13 ØØ ØC Ø6 98 FE 13 ØC Ø6 97 FF 13
[9K5]    11: Ø3 A4 Ø4 92 Ø8 AE FD 1D 16 1D Ø6 81 FD 56 56 81
[EU5]    12: Ø3 C2 Ø9 86 Ø5 AC FE ØD ØØ Ø3 A6 FF 13 ØE D9 FF
[3R5]    13: ØØ Ø3 D7 FE ØØ ØD Ø3 ED Ø6 CD Ø3 B7 FF 81 Ø3 CØ
[VN5]    14: Ø3 3C Ø4 BF Ø9 85 Ø3 AA FF ØD Ø4 E8 Ø5 E4 Ø4 1A
[RP5]    15: FD 56 AC AC Ø3 1C Ø5 16 Ø3 A5 Ø9 CB 1Ø B6 FF 81
[RF5]    16: Ø5 Ø7 Ø4 31 Ø3 9F FF 13 Ø3 A7 Ø9 19 FF 56 Ø4 5A
[1F5]    17: Ø5 12 Ø4 67 Ø3 4A Ø6 FØ Ø6 F7 Ø4 7C FF 2B Ø5 41
[3H5]    18: Ø6 87 Ø3 D4 FF ØD Ø3 55 Ø4 69 FD ØØ 82 82 Ø3 5C
[WH5]    19: FF 56 Ø5 6Ø Ø3 9F Ø4 E9 Ø5 25 Ø4 7Ø Ø7 F4 FD 2B
[TN5]    2Ø: 2B 56 Ø3 BB Ø3 BD FF 2B Ø8 85 Ø4 CD Ø6 13 Ø3 98
[AY5]    21: FF 82 Ø3 CC FF AD Ø3 DE Ø4 9F Ø5 AA Ø3 6D FF ØC
[DM5]    22: Ø3 EB ØA B2 Ø4 C1 Ø3 BE ØB 43 FF ØC Ø3 ØC Ø5 D1
[GE5]    23: Ø5 D7 Ø4 DE Ø3 1F Ø6 E3 Ø5 92 Ø3 25 Ø6 6E Ø4 37
[UN5]    24: Ø5 BB Ø5 C2 FE AC AC Ø8 Ø7 Ø4 DØ Ø4 51 Ø3 18 FF
[HK5]    25: 82 ØB 1C Ø6 E5 Ø4 6C FF 13 Ø9 B1 Ø5 3E Ø3 7E Ø3
[HF5]    26: A2 FE 56 56 Ø7 C7 Ø3 91 Ø5 D2 FD 13 82 ØØ Ø5 5A
[8A5]    27: Ø3 1B Ø7 FØ Ø5 27 Ø4 14 Ø3 B1 Ø6 34 FE ØØ ØØ Ø5
[WT5]    28: 7E Ø6 CØ Ø6 C7 Ø3 EE Ø3 4E Ø6 D3 Ø4 1B Ø5 D7 FF
[PF5]    29: ØØ Ø3 DD Ø4 A5 Ø3 8E Ø4 EC FF ØD Ø4 33 Ø7 76 Ø6
[GT5]    3Ø: C1 Ø4 FB Ø7 C7 Ø3 E9 Ø6 EA FC 82 8F B3 ØØ Ø3 DC
[PR5]    31: FD B3 B3 82 Ø3 9D Ø6 E4 Ø4 2B Ø3 CE FF ØC Ø4 B3
[AL5]    32: Ø6 B6 Ø6 FF Ø3 41 Ø7 85 Ø3 6D Ø4 ØD Ø5 EC F9 B3
```

```
[FL5]    33: Ø5 B3 AD 82 B3 29 Ø3 1F F9 AD 82 82 ØØ 56 AC 56
[NG5]    34: Ø5 53 Ø5 CD Ø4 F2 Ø5 77 Ø7 3D FC 56 56 81 81 Ø6
[RD5]    35: 47 Ø5 ØF Ø6 12 FD 82 B3 82 Ø3 97 Ø5 5F Ø3 23 Ø8
[7L5]    36: 66 Ø3 91 Ø3 4C Ø6 B4 FF 56 Ø8 7A ØC 83 Ø6 91 Ø4
[6L5]    37: 95 Ø3 A2 Ø3 62 FF AD Ø3 DF Ø8 63 Ø4 CE Ø4 4C Ø8
[YU5]    38: B3 FC 56 56 ØØ 2B 1Ø BF Ø8 ØF FF 13 ØA DB FE AD
[DL5]    39: ØØ Ø9 A4 Ø5 ED Ø3 31 FF ØD Ø4 36 FE 81 81 Ø3 ØØ
[YH5]    4Ø: Ø3 3F Ø3 FA Ø6 Ø5 FC ØD ØD ØC ØØ Ø4 5Ø Ø6 92 Ø6
[ZF5]    41: 19 Ø3 D7 Ø4 31 Ø6 66 Ø3 53 Ø3 6F Ø5 2F Ø4 37 FF
[UK5]    42: 56 Ø3 BA ØD 7F Ø5 7Ø Ø6 6A Ø3 57 Ø5 99 FD 82 82
[BK5]    43: 2C Ø5 62 FF ØØ Ø7 A8 Ø3 4F Ø5 8C Ø5 77 Ø6 7D Ø4
[1T5]    44: C2 FF 56 Ø6 87 Ø3 B2 Ø6 52 Ø3 D5 FC AD 7C 7C 7C
[BF5]    45: Ø4 9E Ø6 A1 FF 56 Ø9 A8 Ø6 7Ø ØC 77 Ø3 C1 Ø9 86
[YK5]    46: Ø5 EB Ø3 9Ø Ø3 D7 Ø5 9B FF 5Ø Ø3 1F Ø6 E2 Ø6 EA
[XD5]    47: Ø5 3Ø Ø7 F4 Ø3 C2 Ø4 Ø2 Ø5 CØ Ø9 C7 FF ØØ Ø4 ØE
[UP5]    48: FF 99 Ø3 55 Ø9 1A Ø8 22 Ø4 CF FF ØC Ø3 2E FE ØD
[MN5]    49: ØD Ø6 B4 Ø3 E6 Ø6 FC Ø3 7D Ø5 46 FF ØD Ø3 4C Ø3
[ØS5]    5Ø: 4D FD ØC ØC 99 Ø5 5E FB AD 82 AD 5Ø D7 Ø5 95 Ø5
[VG5]    51: 62 Ø5 CF Ø3 14 FF ØD Ø3 8B Ø7 75 Ø8 FD Ø3 82 Ø7
[4W5]    52: C7 Ø5 4D FF 99 Ø4 94 FF D7 Ø4 9E Ø5 D8 FE 6F 6F
[5M5]    53: Ø5 A2 Ø3 ØD FF ØØ Ø3 AE Ø4 8A Ø3 F1 Ø4 B7 Ø9 BC
[CL5]    54: Ø3 Ø3 Ø9 47 Ø7 91 FF 6F Ø4 DC FF D7 Ø3 1C Ø5 14
[RS5]    55: FC 5Ø 5Ø 75 5Ø Ø3 E9 FF ØC Ø7 ED Ø6 74 ØC FB Ø6
[HG5]    56: Ø6 Ø5 B2 Ø5 D3 Ø5 E2 Ø7 5A F9 6F 5Ø 75 99 75 5Ø
[7L5]    57: 99 Ø5 E6 Ø6 CA Ø6 B4 Ø4 82 Ø4 7E Ø5 6E Ø8 C7 Ø3
[LT5]    58: 4F Ø4 5F FE 99 99 Ø4 57 FF 6F Ø3 9C Ø4 51 FF 99
[ZC5]    59: Ø4 94 Ø3 A1 Ø3 73 Ø4 6C Ø8 73 Ø4 7F Ø6 81 ØB 85
[4G5]    6Ø: Ø5 51 Ø3 A7 Ø5 98 Ø5 D5 Ø3 DE Ø5 E4 Ø3 DD Ø9 6A
[YH5]    61: Ø9 B6 Ø8 FE ØA C7 FF ØØ Ø5 D2 Ø4 56 Ø4 E8 Ø3 EØ
[KP5]    62: Ø6 E4 FC 6F 6F 75 75 Ø3 EC Ø8 2E Ø3 37 Ø6 31 FD
[4M5]    63: 56 56 2B Ø4 4Ø Ø3 47 FF 1D 2D 4A FD 16 1D 56 Ø7
[FP5]    64: 7A F8 2B 81 AC AC 88 16 Ø8 16 2F 89 FB 1D AC 81
[SG5]    65: 81 81 Ø4 43 Ø3 BB FD 81 81 16 2E 4A Ø4 F5 Ø5 BA
[KN5]    66: Ø3 42 FF 56 Ø6 C1 F9 88 88 88 56 2B 2B 2B Ø3 3F
[VL5]    67: Ø3 ØA Ø6 7D Ø4 ØD Ø4 1B Ø5 FF FF 2B Ø4 28 Ø4 1D
[9K5]    68: Ø3 26 Ø4 FA Ø5 BB Ø5 FE Ø4 3F Ø4 Ø7 Ø4 19 FF ØØ
[MF5]    69: Ø3 1Ø FF AC Ø5 Ø3 Ø6 1A Ø5 1F Ø6 Ø1 Ø3 62 Ø6 4D
[5B5]    7Ø: ØB 34 Ø6 41 Ø8 Ø1 Ø4 2A Ø3 3Ø Ø5 74 Ø5 19 FF ØØ
[KB5]    71: Ø6 2F Ø4 84 Ø5 27 Ø7 6D ØB 36 Ø5 2Ø Ø6 23 Ø5 58
[DH5]    72: Ø4 AF Ø6 79 Ø6 BD Ø3 5F Ø7 18 Ø6 E8 Ø3 5E Ø5 A5
[XM5]    73: Ø7 17 Ø9 E4 Ø8 F9 Ø5 AF Ø4 13 FF Ø7 Ø6 F8 Ø5 EF
[NN5]    74: Ø6 D8 FF 56 Ø4 5F Ø4 ØF Ø5 63 ØB FC Ø7 C2 Ø7 EA
[2M5]    75: ØC 3A FF ØØ Ø4 1F Ø9 3F Ø3 5D Ø3 6D ØB BB Ø6 65
[UH5]    76: ØB 53 Ø7 1D ØC 75 ØE 7Ø Ø4 6E Ø6 7F ØE 72 Ø7 1B
[6F5]    77: Ø6 5B Ø4 29 Ø5 B1 Ø5 B4 Ø5 31 Ø8 C4 Ø6 5D Ø4 AE
[7F5]    78: Ø4 EE Ø4 CB ØB A5 Ø7 CC Ø3 1Ø Ø4 12 Ø5 9Ø Ø5 Ø9
[PF5]    79: Ø8 BA Ø6 6B Ø8 66 Ø6 1D Ø7 87 Ø6 2B Ø8 2C Ø7 36
[WE5]    8Ø: Ø9 58 ØB A6 Ø6 4F Ø8 13 ØC FF ØC 88 Ø5 86 ØB 53
[SD5]    81: ØF 83 Ø9 51 Ø5 A3 Ø7 B3 Ø5 BB ØE 87 26 DB Ø4 35
[8H5]    82: 13 87 1D F6 ØA 14 FD 2B 2B 56 12 42 Ø3 4Ø Ø6 A5
[T85]    83: Ø5 36 Ø3 53 Ø5 51 Ø7 5Ø Ø4 6A Ø3 72 Ø5 56 F5 ØØ
[ZH5]    84: ØØ Ø7 56 56 81 81 AC AC 81 AC Ø3 83 Ø4 87 FF 81
[KG5]    85: Ø3 82 FF 81 Ø5 57 Ø8 C3 Ø4 79 Ø5 91 Ø3 8B FF 88
[XP5]    86: Ø4 AE FF AC Ø3 B3 Ø3 81 ØB 7A FF 81 Ø9 AF Ø3 C3
[FH5]    87: Ø3 A8 Ø6 6E Ø7 5A Ø6 7Ø Ø5 C1 FF 56 Ø3 CB Ø5 8C
[WK5]    88: Ø4 EC Ø5 D5 Ø7 3F Ø5 CF Ø6 CF Ø4 D3 ØC 95 Ø6 15
[TF5]    89: Ø8 E5 ØD Ø6 FE Ø7 Ø7 Ø9 3E Ø3 84 Ø3 EF Ø5 11 Ø7
[1F5]    9Ø: 13 Ø5 DA Ø5 9E ØA 22 Ø9 6C Ø4 57 Ø3 74 Ø5 5F Ø5
[DM5]    91: B7 Ø6 73 ØA 4F Ø6 DF Ø3 DE Ø5 94 Ø6 9E Ø6 63 Ø5
[KH5]    92: 8B Ø7 64 Ø6 5F Ø5 D6 Ø8 54 Ø8 14 Ø6 A6 Ø4 BD Ø8
[7N5]    93: EØ Ø5 C9 Ø5 7F Ø8 AB Ø6 7C 12 Ø1 Ø4 DF Ø4 FE Ø8
[1G5]    94: 5D Ø5 DF Ø8 12 Ø4 2C Ø8 AA Ø9 32 Ø7 37 ØB ØD Ø7
[LP5]    95: DA Ø6 3E Ø8 1D Ø6 E5 ØB 3F ØA 5D ØC 4D Ø4 86 Ø4
[EK5]    96: 4A Ø9 5C Ø8 D4 Ø8 9Ø ØA 73 Ø5 9C Ø8 C3 Ø9 BE Ø6
[2N5]    97: D5 Ø8 CF Ø6 6E Ø6 C8 ØF Ø6 Ø5 AE 12 C5 ØC Ø7 Ø7
[ØS5]    98: 8F ØA DB Ø5 FB Ø6 24 ØE BE ØC C1 Ø8 35 ØC ØC Ø9
[L43]    99: 52 ØB D9 31 7E 34 7C 1C E3
```

Listing 8. bild.pak ist eine gepackte Grafikdatei.

```
[6R5]     1: FF 41 13 ØØ FE ØF 1D ØC Ø9 FF ØF 13 ØE ØB 35 FF
[ZN5]     2: 16 11 41 FD Ø8 ØF 1D ØA 49 FD 1D ØF ØF 12 4E ØB
[ES5]     3: 49 21 4Ø FF 1D 22 62 ØC C4 Ø7 A5 FE D4 D6 ØA 58
[BT5]     4: 2Ø A3 ØC C4 Ø4 CE Ø4 E8 FF CC Ø9 D7 FF 1D Ø6 EA
[5N5]     5: 1F 68 FF ØF Ø6 47 ØA 4E ØB 17 ØE 63 Ø5 4C Ø7 4C
[AN5]     6: Ø7 7D Ø3 4B Ø7 ØA Ø5 76 Ø6 93 FF CE ØA D7 ØB A4
[BM5]     7: FC ØF ØF ØB ØB Ø3 84 Ø4 B1 Ø9 BA Ø4 86 FC B4 ØØ
[GH5]     8: ØØ D7 Ø4 C6 FF ØØ Ø3 B9 Ø7 A8 ØE 99 Ø6 E8 Ø3 B5
[4U5]     9: FD ØA ØA ØB Ø6 EF Ø5 B9 Ø5 FF FC D7 D7 C7 C1 Ø4
[KV5]    1Ø: Ø8 FC C1 C7 D7 D7 Ø3 CF Ø8 68 ØD DA Ø5 28 FF ØF
[ØK5]    11: Ø3 F5 FF D7 Ø6 31 Ø3 F6 Ø5 8F Ø3 4Ø FF 1D Ø4 Ø4
[GW5]    12: Ø3 CC FF B4 Ø4 ØC FE D7 1D ØA 67 ØC 1B Ø5 68 FE
[HK5]    13: Ø4 ØA Ø7 87 Ø4 75 FE ØA Ø4 Ø8 7C Ø4 43 FC CØ B4
[VS5]    14: BB CØ Ø3 4D FF 1D Ø3 6A Ø9 53 ØC 5B Ø5 A8 FD 1D
[XF5]    15: Ø4 Ø4 ØA 3Ø FF Ø4 Ø4 95 Ø5 7F FE 16 1D Ø3 84 Ø5
[CN5]    16: C8 Ø5 95 ØA 93 Ø6 9C FE 23 23 Ø9 E5 FC 1D 1D Ø3
[1T5]    17: ØA Ø6 F1 FF Ø3 Ø4 8F Ø8 BD Ø4 EC Ø5 Ø8 ØA A3 ØA
[6L5]    18: D7 FD 23 23 65 ØB 1Ø Ø3 EE Ø4 32 Ø7 ØC Ø9 FE ØC
[BP5]    19: 46 ØC 13 FD 16 CC D4 Ø5 23 ØB 52 ØA Ø9 24 3C FC
[8G5]    2Ø: 23 23 23 ØF 11 8F ØB 89 1F 8Ø Ø3 62 FF Ø9 Ø5 A2
[6T5]    21: 37 A8 FF 65 Ø3 63 FE Ø9 ØF Ø3 E4 37 E8 Ø5 5E FE
[YK5]    22: Ø9 23 3E 26 1C A4 14 BF FF ØF 29 CØ Ø3 E7 3F ØØ
[3K5]    23: FE 41 1D 12 41 FF ØF 13 41 16 53 FE 1D 1D 1E 8Ø
[1T5]    24: FE ØF ØF 2Ø AØ ØA 8Ø Ø5 3B ØD A4 FF ØF Ø3 FF 25
```

```
[2U5]    25: AØ FF 1D Ø4 C9 FF D6 Ø4 ØA Ø4 DE Ø7 E8 Ø3 DC FA
[TD5]    26: Ø2 ØF ØF Ø8 Ø8 Ø8 Ø5 15 Ø8 E8 Ø4 DC Ø5 31 Ø6 2A
[2K5]    27: Ø3 34 Ø3 42 FF 16 Ø5 Ø9 FF D5 Ø5 4B FF ØF Ø9 12
[FG5]    28: FF Ø2 Ø4 1C FD ØØ ØØ Ø2 1D 23 FF 41 Ø3 43 Ø3 45
[HS5]    29: FF BA Ø3 87 FE D5 D3 Ø4 4D FF CE Ø8 76 27 5A Ø4
[CX5]    3Ø: AD Ø4 8E F6 B4 CE D3 D3 D3 CE D5 D3 CE 1D 33 93
[Ø16]    31: F9 CE D5 BA D4 D4 CE B4 Ø3 ØC FF CE ØB D1 FF 23
[CP5]    32: Ø3 F4 23 EØ Ø3 C4 Ø5 ØB FC D5 D6 D6 B4 Ø3 CC ØB
[7N5]    33: 12 FF 23 18 5E Ø3 5C ØD 39 Ø3 4B FF D5 Ø3 CA Ø3
[2U5]    34: 4E ØD 5Ø 28 1D FD CE CE D3 Ø3 C4 Ø6 ØB FF 1D Ø8
[ZN5]    35: A7 Ø3 DA 1F 9D FF 52 Ø7 FD FF 16 Ø3 C9 Ø8 Ø7 Ø3
[CS5]    36: F9 ØA D3 FF Ø2 Ø6 1E FF 52 Ø4 FC 12 29 FF 52 Ø3
[3V5]    37: 24 Ø6 FF FD ØF ØF 1D Ø3 ØD FC C7 C7 1D 1D Ø4 2A
[TG5]    38: ØE 13 FF Ø2 Ø4 62 Ø3 3D 11 29 Ø3 3B FE 52 52 Ø5
[5N5]    39: 3F FB ØF ØF D6 D6 16 Ø3 4D Ø6 6A Ø8 67 FE 1D 1D
[LR5]    4Ø: Ø4 5C Ø7 89 FF 52 Ø8 A8 Ø4 AØ Ø6 B3 ØA FA FB CE
[PX5]    41: D6 D5 D5 CE Ø6 8D ØC E4 Ø3 BC FE ØF ØF Ø6 CD ØA
[UM5]    42: 66 ØC EF ØC FC ØB F1 Ø7 13 Ø4 24 Ø6 F7 ØC 24 Ø6
[YN5]    43: F9 Ø9 2E Ø6 7F Ø9 8A ØB ØE Ø5 7A Ø5 5D ØD CE Ø7
[NL5]    44: 1F Ø8 5F Ø6 BF ØC Ø7 12 11 11 63 ØC 9D Ø4 CØ ØD
[LT5]    45: 46 ØD D1 22 9E 14 FF FF ØF 29 ØØ Ø3 27 3F 4Ø FE
[AE5]    46: 41 1D 12 81 FF ØF 13 81 11 93 Ø7 A9 Ø4 CØ 1Ø FØ
[FR5]    47: 14 E9 1Ø E8 FE 1D ØF Ø6 FA 19 CØ Ø6 F9 1A CA Ø6
[CG5]    48: 2A 16 FF Ø4 55 ØF 2A 1D 28 Ø5 54 Ø7 89 Ø5 66 Ø3
[ES5]    49: 96 2Ø C4 ØB 79 FE ØF AD Ø7 C5 FD ØF C6 C6 Ø6 9Ø
[BM5]    5Ø: 22 D6 Ø8 E1 Ø3 ØØ Ø5 C4 FF 56 Ø4 Ø8 FB 9Ø C7 BB
[2S5]    51: BB 1D 22 D2 ØA 89 Ø3 3E FF ØF Ø4 Ø3 FD 56 ØØ CB
[TG5]    52: Ø3 44 FD AD AD 9Ø Ø3 1Ø Ø5 27 ØC 85 Ø4 3B Ø9 67
[ØK5]    53: Ø4 6Ø Ø7 C6 FF CB Ø4 5Ø FF 41 Ø5 Ø4 FF ØØ Ø3 ØA
[RB5]    54: Ø4 49 Ø5 97 Ø4 68 Ø8 C5 Ø6 C8 Ø8 DØ Ø9 71 Ø5 Ø8
[UM5]    55: FF 56 Ø5 7C Ø4 88 FF ØØ Ø7 47 FF A6 Ø5 6A 11 93
[7N5]    56: Ø4 A2 13 D3 Ø6 7B FE A6 A6 Ø3 C3 Ø5 87 FF A6 Ø5
[NK5]    57: 8E Ø6 D1 Ø4 B8 Ø3 Ø2 Ø3 ØB FF 1D Ø5 B2 ØD E7 Ø6
[VM5]    58: B6 Ø4 BB Ø7 BE FF 9Ø Ø3 C7 Ø4 ØB FF 52 Ø9 ØE Ø3
[UL5]    59: 1F Ø3 1B Ø6 ØD Ø4 89 FF 1D Ø7 13 Ø3 5A FD 56 56
[UL5]    6Ø: ØF Ø3 57 Ø3 77 Ø5 4F FF 41 Ø3 78 FF C6 Ø3 71 Ø5
[ØR5]    61: CD FF 52 Ø5 4D Ø5 A8 FF 16 Ø3 2B Ø7 5D FF 56 Ø3
[HL5]    62: C9 Ø5 67 Ø4 77 Ø4 E3 Ø6 9C Ø6 4E FB 41 16 16 C7
[ØN5]    63: 9Ø Ø8 77 FF 52 Ø5 8D Ø7 93 FD 1D AD 56 Ø7 9E Ø8
[2Y5]    64: A4 Ø5 D6 FD 56 AD CB ØB B3 Ø3 BF FE C6 9Ø Ø9 F5
[FU5]    65: Ø3 CD Ø3 CF Ø9 E8 F5 CB 16 16 4C 16 CB AD CB 56
[YN5]    66: CB 16 Ø9 E7 Ø3 EF FD 56 1D 4C Ø6 35 Ø6 FC Ø6 C7
[YC5]    67: Ø8 Ø8 ØA DØ Ø3 2F FF 16 Ø3 39 Ø7 75 Ø9 53 Ø5 41
[4E5]    68: Ø4 38 Ø6 8E Ø3 CØ ØA 82 ØD 8D Ø3 99 Ø8 37 ØD 9Ø
[8M5]    69: ØB 48 11 7D ØE A3 ØA 35 11 A6 Ø7 CC ØF BE 15 A2
[BV4]    7Ø: 1B ØD Ø3 3D 14 3F FF ØF 29 4Ø Ø3 67 7F 8Ø
```

Listing 9. buttons.pak ist eine gepackte Grafikdatei.

```
[B46]     1: FF FF 7F ØØ 7F 7F 7F FE 7F 7E 7F FD 7F 7C 7F FB
[KX5]     2: 7F 7A 7F F9 7F 78 7F F7 7F 76 7F F5 7F 74 7F F3
[926]     3: 7F 72 7F F1 7F 7Ø 7F EF 7F 6E 7F ED 7F 6C 7F EB
[K26]     4: 7F 6A 7F E9 7F 68 7F E7 7D 66 FE 6B 6B 3E A6 FE
[4R5]     5: 1D 1D 34 E4 FF 6B Ø4 59 FF 16 Ø3 5D FD 6B ØØ ØØ
[8T5]     6: 34 24 FF 1D Ø4 99 FF ØF Ø3 9D FB 1D ØØ FF 6B 16
[VU5]     7: 33 27 FF FF Ø3 62 Ø4 DD Ø3 6Ø 33 A6 FB FF 16 FF
[RH5]     8: 16 ØØ Ø5 1D FE ØØ 16 Ø7 1B 2F 2A FF ØØ Ø8 25 Ø9
[1S1]     9: 5A 7F 6B 15 EB
```

Listing 10. key1.pak ist eine gepackte Grafikdatei.

```
[RY5]     1: FF FF 7F ØØ 7F 7F 7F FE 6Ø 7E FF C1 3F AØ FF 9Ø
[PY5]     2: 3F EØ 3D 1E FF BB Ø3 5C 3B 5E FB C1 C1 C1 C6 BB
[NT5]     3: 3A 9E FE C1 C7 Ø5 1A FE C1 72 36 22 FC C2 C7 48
[AT5]     4: 7A Ø3 5B FD 48 72 C1 35 62 FE C6 B5 Ø3 5A FE 57
[SG5]     5: 57 Ø4 5D 35 21 Ø6 98 Ø4 9C 39 A1 FD 57 ØØ 57 Ø3
[KG5]     6: 1A Ø3 9F 38 E3 Ø5 5A 39 2Ø FC 48 82 57 82 Ø3 9A
[KR5]     7: FD 48 48 9Ø 34 A3 FD FF B4 48 Ø3 9C Ø3 9B 3A AØ
[VR5]     8: FC 48 82 58 58 39 DF FD FF B4 6C Ø3 19 Ø4 1E 37
[2T5]     9: 21 FD 9Ø 9Ø 48 Ø4 9A FB 6C 9Ø 9Ø C7 C7 32 A4 FA
[DP5]    1Ø: C6 C7 B5 9Ø 9Ø 6C Ø3 DB FB 9Ø 9Ø 73 BB C1 31 A2
[5N5]    11: F9 C7 C7 B5 B4 B4 73 9Ø Ø3 1B F9 73 73 6D 6D 9Ø
[HN5]    12: BB C1 2F A8 FE C1 B4 Ø7 18 FD 9Ø 9Ø 6D Ø3 6Ø Ø3
[DK5]    13: E2 3Ø 28 Ø3 61 Ø5 9A Ø3 2Ø FB 6D 6D B5 C1 BB 2D
[TW5]    14: E9 Ø5 95 FE 6C 6C Ø3 9C Ø3 DA F9 6D 73 6D B4 C1
[7T5]    15: BB 6D 2B A9 Ø3 94 Ø3 1E Ø3 DE FF 6C Ø3 1C Ø3 DF
[1N5]    16: 2F E3 FB C1 C1 B4 9Ø 9Ø Ø3 1B Ø5 9A Ø3 2Ø FF 48
[ML5]    17: 2E 23 Ø5 52 Ø6 99 Ø4 DB FF 6C Ø3 61 FF C1 2D 65
[7P5]    18: Ø4 93 FF 6D Ø3 A3 FB C1 C1 82 82 82 Ø6 2Ø FE C1
[ZU5]    19: B4 29 E7 FB C1 C1 9Ø 9Ø C1 Ø4 E3 FE B4 ØØ Ø3 DC
[VH5]    2Ø: FF 57 Ø7 2Ø 2B E6 Ø5 12 FF BB Ø5 17 Ø3 1E Ø3 5E
[ST5]    21: Ø3 15 FE BB 6D 29 67 FC 9Ø 9Ø 9Ø B4 Ø3 57 FA B4
[FL5]    22: B4 6C ØØ ØØ ØØ Ø5 5E FF C6 Ø3 1Ø 2C 66 Ø3 5Ø FF
[LP5]    23: B4 Ø3 9Ø Ø7 99 FA B4 C6 C6 BB 9Ø 9Ø 29 A7 Ø3 CE
[8S5]    24: FF 6C Ø4 12 FA 9Ø 6C 6D ØØ ØØ 6D Ø3 DD FF C6 2E
[CF5]    25: A2 Ø8 4E FD 48 48 6D Ø3 59 Ø5 1E Ø3 D6 31 26 Ø4
[7H5]    26: 55 Ø7 9D Ø3 63 32 25 Ø3 ØF FE 9Ø 6C Ø3 9B FE ØØ
```

Listing 11. man.pak ist eine gepackte Grafikdatei.

```
[XE5]    27: ØØ Ø4 23 33 A5 Ø6 D7 FC ØØ 82 82 ØØ Ø4 59 33 E5
[WK5]    28: FF B4 Ø4 19 FF 82 Ø6 1E 33 25 Ø7 57 FE 57 57 Ø3
[3K5]    29: 6Ø FF 48 36 64 FB 48 6C 48 57 ØØ Ø4 AØ FF B4 36
[ZM5]    3Ø: A4 FC 9Ø 6C 9Ø 48 Ø3 61 FE 48 9Ø 34 E3 FD B4 9Ø
[SS5]    31: 73 Ø4 1A FC 48 48 6C 6C 37 22 FF B4 Ø3 5A FF B4
[FL5]    32: Ø4 5D 3D 62 FF 6C Ø4 AØ 33 A3 Ø3 D7 FB 6D 73 9Ø
[9V5]    33: B4 6C Ø4 DF FF C1 32 E4 FF 48 Ø9 17 FD 6D 6C B5
[5P5]    34: 38 23 FF 9Ø Ø5 5C FF 73 33 62 FF 48 Ø8 96 F9 48
[4M5]    35: 73 6D 9Ø B5 C1 C1 32 66 Ø8 D7 FF 48 Ø3 EØ FF 9Ø
[3N5]    36: 34 E4 FF 73 Ø5 19 FF C1 Ø3 1F FE 9Ø 9Ø 31 E3 Ø3
[5N5]    37: D6 FF C1 Ø8 59 FC 6D 9Ø 73 B4 3Ø 65 FF C6 ØA 96
[HS5]    38: Ø4 5F FF B4 2F 64 FD 48 C6 C6 Ø8 D7 Ø5 9E 3Ø E4
[CP5]    39: FF C6 Ø6 D5 Ø3 22 Ø6 9D FE 6D C1 2D 25 Ø3 54 Ø7
[8H5]    4Ø: 96 Ø4 23 Ø3 61 FE 9Ø 9Ø 2D 65 Ø3 94 FF 6D Ø5 D7
[CL5]    41: FE 6D 6D Ø6 5D Ø3 D6 2B A7 Ø3 93 Ø4 A1 Ø3 14 Ø6
[3K5]    42: 21 FE 6D 48 Ø3 16 2B E6 FF 48 Ø7 19 Ø5 54 Ø6 9D
[GG5]    43: Ø5 24 2A A9 Ø3 93 Ø3 27 FF 6D Ø3 94 Ø5 5D Ø5 9A
[F22]    44: FF B4 2D 6A 2A D5
```

Listing 11. man.pak ist eine gepackte Grafikdatei.

```
[TF5]     1: Ø7 ØØ FE 2B 2B Ø4 Ø5 Ø3 Ø8 Ø7 Ø1 Ø3 15 1E EC Ø8
[BH5]     2: ØB FF 56 Ø5 4Ø Ø3 38 ØA 18 FE 2B 2B Ø6 4Ø ØD 5A
[4P5]     3: ØE 47 Ø3 75 Ø7 61 FE AC 81 Ø3 8Ø FE AC 81 Ø4 65
[4L5]     4: ØA 4D FE AC 81 Ø3 83 Ø3 84 Ø4 83 Ø4 97 FF 81 Ø4
[RH5]     5: 86 FF 56 ØE 4B Ø4 98 Ø9 99 Ø3 81 Ø8 A9 Ø6 92 Ø3
[TK5]     6: A5 Ø4 D5 Ø3 C8 Ø4 D8 Ø3 EØ Ø4 87 Ø6 CA Ø4 B3 Ø6
[VR5]     7: C1 Ø6 DA Ø5 C4 Ø5 DC Ø8 CC Ø5 FE Ø6 7A Ø6 DC Ø5
[UC5]     8: E1 Ø7 88 Ø8 91 Ø6 FD Ø5 EØ Ø3 36 Ø5 22 Ø7 2D Ø3
[XN5]     9: C7 Ø6 FE Ø5 26 Ø7 15 FF 81 Ø3 EA Ø8 4B Ø3 B8 Ø7
[KC5]    1Ø: 1E Ø6 6Ø Ø6 4Ø Ø6 A8 Ø4 72 Ø7 6Ø Ø6 64 Ø5 9E Ø6
[AE5]    11: 9F FF ØØ Ø4 2C FF 2B Ø3 69 Ø4 38 Ø6 26 Ø5 18 Ø6
[6G5]    12: 88 Ø7 2A Ø6 19 Ø7 BD Ø4 69 Ø3 DE Ø4 11 Ø8 E9 Ø5
[UF5]    13: F2 Ø7 96 Ø5 D9 Ø4 DF Ø6 Ø1 FF ØØ Ø5 EC Ø3 Ø5 Ø3
[7E5]    14: Ø7 Ø5 15 Ø3 1B Ø5 E3 Ø5 14 Ø6 EB Ø4 3Ø FF 2B Ø5
[9G5]    15: E8 Ø8 DF Ø7 1F Ø8 E9 Ø4 DØ Ø5 26 Ø6 Ø2 Ø4 42 Ø7
[CD5]    16: 25 ØA 2A Ø6 Ø6 Ø9 17 Ø5 5C Ø3 5Ø Ø5 CC Ø3 8A Ø7
[NA5]    17: 62 Ø7 77 Ø6 58 Ø9 48 Ø9 61 Ø5 4Ø Ø7 Ø1 Ø3 CØ Ø4
[XH5]    18: ØC Ø5 8C FF 2B Ø3 51 Ø4 11 Ø8 12 Ø7 DC Ø3 8B Ø6
[1D5]    19: EC Ø5 3Ø Ø9 D9 Ø7 8Ø Ø4 D3 Ø7 ØA Ø7 6Ø Ø5 29 Ø4
[EG5]    2Ø: D2 Ø6 59 Ø4 9Ø Ø6 8C Ø3 F1 Ø3 BØ Ø7 AD Ø9 Ø9 ØF
[AA5]    21: 47 Ø5 3F Ø3 2Ø Ø6 58 Ø7 35 Ø9 6B Ø7 8C Ø3 11 Ø3
[9M5]    22: 7E 13 44 14 81 1F A8 16 7F 1F AA 35 E2 11 BA Ø3
[HB5]    23: 66 FF 56 Ø3 63 Ø3 46 Ø4 4C Ø6 49 Ø8 17 Ø3 57 Ø3
[7M5]    24: 42 FF 2B Ø7 4A ØC 6C Ø7 C8 Ø4 41 FD 56 56 AC Ø3
[GH5]    25: 86 FF 81 Ø4 86 Ø3 88 Ø4 8A ØA 76 FF ØØ Ø4 82 Ø4
[ZE5]    26: 92 Ø7 A5 Ø8 A5 Ø4 4E Ø4 9B Ø4 81 Ø4 9Ø FF 81 Ø3
[MM5]    27: C8 Ø6 AC Ø6 C5 Ø4 97 Ø5 DC Ø5 C2 Ø6 C6 Ø8 E6 Ø3
[HS5]    28: ED FF 56 Ø3 D7 Ø3 56 Ø4 FC Ø3 E2 FF 56 Ø5 EE Ø3
[8E5]    29: F7 Ø5 Ø6 Ø6 Ø6 Ø8 98 Ø5 E2 Ø3 A3 Ø4 A2 Ø6 24 ØD
[9C5]    3Ø: 11 FC 2B ØØ 81 AC Ø8 2B Ø4 3Ø Ø7 49 ØA 16 Ø5 55
[BC5]    31: Ø6 45 Ø5 28 Ø8 6D Ø3 74 Ø3 BF Ø3 AØ Ø3 EC Ø7 29
[6E5]    32: Ø3 C1 Ø4 64 Ø6 6A Ø3 99 Ø5 5B FF 2B Ø4 Ø3 Ø7 71
[5D5]    33: Ø3 93 Ø3 8Ø Ø7 6A FF 81 Ø5 7B Ø3 6Ø Ø5 9Ø Ø4 8B
[KK5]    34: Ø5 A7 Ø4 BØ Ø4 C6 Ø3 C9 Ø8 5C Ø6 A8 Ø4 CF Ø9 ED
[EG5]    35: Ø5 EC Ø6 7C Ø4 Ø9 Ø6 F5 Ø4 D9 Ø7 F1 Ø3 D8 Ø6 9B
[PF5]    36: FF ØØ Ø7 F2 Ø6 18 Ø6 Ø1 Ø5 D8 Ø3 37 Ø3 9F Ø3 FC
[CF5]    37: Ø5 13 Ø9 14 ØD 29 Ø3 1E FF ØØ Ø6 ØE Ø6 4B Ø6 47
[1B5]    38: Ø5 44 Ø6 19 Ø6 3E Ø7 Ø2 Ø4 67 Ø9 43 Ø7 1A Ø3 9E
[BH5]    39: Ø8 86 Ø5 66 Ø8 23 Ø6 B5 Ø4 9F Ø7 AD Ø3 FB ØC E8
[TM5]    4Ø: Ø6 92 Ø5 9D Ø6 A5 Ø4 BF ØB 4Ø Ø4 EE Ø3 7D Ø3 E8
[5F5]    41: Ø5 BF Ø6 79 Ø6 Ø4 Ø5 EB Ø4 Ø1 Ø6 9C Ø3 C9 Ø3 19
[KM5]    42: Ø7 65 Ø6 7D Ø6 Ø2 Ø4 FE Ø4 FB Ø5 FE FE ØØ 56 Ø3
[KK5]    43: E1 Ø4 3D FF 56 Ø8 4Ø ØD 58 Ø6 3C Ø3 4E Ø6 97 Ø8
[6K5]    44: 37 Ø7 62 31 83 Ø6 FC Ø9 78 12 A6 49 D5 19 D3 Ø8
[NG5]    45: 1F FF 56 Ø9 4Ø Ø4 71 Ø5 34 Ø6 4C FF 56 Ø4 3E Ø4
[8H5]    46: 3F FF 2B Ø4 6Ø Ø4 62 Ø4 76 Ø4 5Ø Ø3 6B ØA 4Ø FF
[UN5]    47: AC Ø6 8Ø FE 81 AC Ø5 63 Ø6 4E Ø3 7D FF 81 Ø4 85
[GK5]    48: Ø6 83 Ø6 9F FF 81 Ø8 48 Ø4 71 Ø6 7D Ø3 A6 Ø7 BB
[WL5]    49: Ø5 C1 Ø4 A9 Ø7 8F Ø3 C5 Ø4 C6 ØE DB Ø5 A9 Ø5 CD
[6N5]    5Ø: Ø3 B6 Ø7 E4 Ø7 CØ FF 56 Ø4 FC Ø4 5F Ø6 3C Ø4 FF
[BC5]    51: Ø6 Ø7 Ø4 17 Ø6 19 Ø4 21 Ø5 EC Ø5 B4 Ø4 FE Ø5 1C
[TD5]    52: Ø3 3D Ø4 18 Ø8 16 Ø5 ØE Ø4 B8 ØA 1D Ø6 1B Ø4 59
[YD5]    53: Ø6 ØB Ø4 B2 Ø4 37 Ø7 F9 Ø7 5F Ø4 78 Ø7 8B Ø8 Ø5
[RK5]    54: Ø8 39 Ø5 3E Ø5 24 Ø6 CB FE 56 2B Ø5 68 Ø6 B6 Ø8
[EK5]    55: 99 Ø3 B2 Ø6 AC Ø5 B1 Ø4 A6 Ø3 31 Ø8 D4 Ø6 DD Ø4
[6K5]    56: C5 Ø3 CC Ø5 71 Ø5 BD Ø6 F3 Ø7 E3 Ø3 E1 Ø6 8B Ø6
[ZK5]    57: C7 ØA DE Ø7 17 Ø5 24 FF ØØ Ø5 EE Ø7 Ø7 Ø6 1F Ø4
[6D5]    58: F1 Ø7 41 Ø7 CA Ø7 47 Ø5 55 Ø9 42 Ø6 5C Ø3 2D Ø4
[DE5]    59: 66 Ø6 53 Ø5 3D Ø5 3C Ø7 77 Ø9 CA Ø5 87 Ø8 56 Ø7
[TH5]    6Ø: 62 Ø7 A1 Ø7 4F Ø7 A5 Ø6 54 Ø5 AC Ø5 2E Ø6 4E Ø8
[AK5]    61: 74 Ø8 8Ø Ø6 24 Ø5 ØB Ø6 EE Ø5 EØ Ø7 EF Ø6 91 ØA
[KL5]    62: Ø7 Ø4 FA FF ØØ Ø8 3C Ø6 E2 Ø3 1E Ø5 6D Ø4 ED Ø4
[ØK5]    63: EC Ø5 2C Ø3 2F Ø5 Ø6 Ø6 34 Ø5 2E ØE 4A Ø6 FB Ø8
[LL5]    64: 59 ØF 4D Ø6 6F ØF 64 ØB 69 Ø5 48 17 7E ØC 99 Ø8
[EP5]    65: 33 Ø8 6D 18 9C 1B E5 13 CC 12 49 1B A4 Ø3 36 FF
[GE5]    66: 56 11 44 Ø6 5F Ø8 34 FD 2B ØØ 56 11 43 Ø3 65 Ø6
[ØL5]    67: 59 Ø3 64 FA AC 81 AC 81 81 AC Ø3 84 Ø3 8C Ø3 85
[FG5]    68: FE AC 81 Ø4 53 Ø3 55 Ø6 5C FF 2B Ø3 95 Ø8 86 Ø4
[MV5]    69: 8B Ø5 AB Ø5 94 Ø6 5A FF 56 Ø4 85 FF 56 Ø3 C6 FF
[UD5]    7Ø: 81 Ø4 83 Ø3 B3 Ø3 D2 Ø5 55 Ø6 7D Ø3 A2 Ø5 DØ Ø5
[UM5]    71: D1 Ø4 C8 Ø5 EE Ø4 B6 Ø5 9C FC 2B ØØ 81 AC Ø4 C9
[ØF5]    72: Ø6 C5 FF 56 Ø5 C7 Ø5 D3 Ø4 D6 Ø4 1D Ø5 A4 Ø5 13
```

```
[NF5]    73: Ø4 ØE Ø6 C7 Ø3 2E Ø3 Ø2 Ø3 79 Ø6 FE Ø6 2D Ø6 ØA
[7D5]    74: Ø4 5Ø Ø6 14 Ø3 FA Ø5 6Ø Ø4 E3 Ø6 48 Ø8 EA Ø6 EF
[XF5]    75: Ø6 3A FE ØØ ØØ Ø7 4D Ø8 88 Ø8 49 Ø6 1B Ø3 DD Ø7
[5G5]    76: 11 Ø8 4C Ø8 8B Ø3 B8 Ø3 9F Ø3 99 Ø4 96 Ø7 C4 Ø3
[9K5]    77: B6 Ø3 CB Ø4 DØ Ø3 D7 Ø4 98 Ø3 DC Ø3 A1 Ø6 B6 Ø3
[NN5]    78: EØ Ø4 E9 Ø4 E1 Ø5 D5 FF 2B Ø5 5A Ø3 CØ Ø7 C7 Ø5
[EG5]    79: F6 Ø5 EE Ø7 EA Ø6 FC Ø3 15 Ø4 Ø9 Ø9 Ø6 Ø6 11 Ø6
[3G5]    8Ø: 2F Ø5 9D Ø6 15 Ø8 C4 Ø7 27 Ø8 97 Ø4 DE Ø3 19 Ø6
[VD5]    81: E7 Ø8 EA Ø7 DØ Ø6 1A Ø4 71 Ø6 4B Ø7 4Ø Ø9 89 Ø5
[KB5]    82: 9B Ø7 7F Ø7 87 Ø7 46 Ø7 22 Ø5 5D Ø3 AØ Ø3 Ø1 Ø6
[KH5]    83: 62 Ø5 C3 Ø4 B5 Ø6 DA Ø7 9B Ø4 DF Ø5 75 Ø8 28 Ø5
[WL5]    84: 17 Ø5 7B Ø7 5D Ø4 Ø1 ØD Ø8 Ø6 EØ ØA FA Ø4 CC Ø4
[CK5]    85: D6 Ø6 B6 Ø3 18 Ø5 D6 Ø5 14 Ø8 99 Ø7 DF Ø6 49 ØB
[FG5]    86: 41 Ø8 45 Ø6 3D Ø6 12 Ø5 BD ØA 49 ØC 16 2Ø 8F ØC
[YXØ]    87: 4Ø 44 BB
```

Listing 12. mauer1.pak ist eine gepackte Grafikdatei.

```
[5R5]     1: FF 2B Ø8 FC FF 2B Ø6 Ø9 Ø3 1Ø FC 56 2B 56 56 Ø7
[6C5]     2: Ø9 ØF 1D Ø7 Ø3 FF Ø7 Ø5 34 Ø6 FB Ø5 ØØ Ø4 45 Ø3
[ZG5]     3: 16 Ø7 44 FF 56 Ø3 15 Ø3 53 ØC 59 Ø5 52 Ø7 2B FE
[SG5]     4: 2B Ø7 Ø3 67 Ø7 38 Ø7 3E FF 56 Ø4 82 Ø4 49 Ø4 6F
[NS5]     5: Ø3 74 FF 81 Ø6 6Ø FD AC AC 81 Ø5 9C FE 88 88 Ø5
[9C5]     6: 63 Ø7 ØD Ø3 76 FF 56 Ø3 73 Ø8 37 Ø3 84 Ø4 B8 Ø9
[LM5]     7: 16 Ø8 93 FF 88 Ø3 9B Ø5 DC FF 23 Ø4 DE Ø6 61 Ø3
[6P5]     8: AC Ø7 95 Ø4 C5 Ø3 F9 Ø5 BE Ø7 6Ø FD 2B 81 81 Ø5
[VT5]     9: EB Ø3 9A Ø4 D6 FC 88 AC AC 47 Ø4 DF FE 23 23 Ø4
[KG5]    1Ø: 1C Ø3 A1 Ø6 E9 Ø3 ØB ØA FØ Ø3 C7 Ø5 87 Ø4 2A Ø5
[FU5]    11: 27 Ø4 AC FE 81 2B Ø5 DD F7 88 88 47 47 6B 88 AC
[AB5]    12: 23 8F Ø3 22 Ø3 19 Ø3 67 Ø5 A9 Ø3 3Ø Ø7 72 Ø3 FB
[WC5]    13: FE Ø7 Ø7 Ø4 1Ø Ø6 82 Ø9 49 FD 56 81 81 Ø3 64 Ø3
[AL5]    14: 65 F7 23 B3 6B 88 B3 8F 8F B3 47 Ø3 63 Ø3 89 Ø6
[FC5]    15: 29 Ø8 7Ø Ø9 71 Ø3 92 Ø4 3Ø Ø3 93 Ø6 8B Ø5 CØ Ø3
[NH5]    16: 57 F7 AC 47 47 B3 6B 23 B3 8F B3 Ø4 A2 Ø4 56 Ø7
[ØS5]    17: AA FF AC Ø4 B8 Ø7 F2 FF 81 Ø3 3C Ø7 F7 Ø4 93 Ø8
[6R5]    18: CC Ø4 E7 Ø3 5E FB 47 B3 8F 23 B3 Ø3 1F Ø4 A3 Ø6
[ZN5]    19: E7 Ø3 2D FF AC Ø3 F1 Ø4 32 ØA B7 Ø4 C2 Ø4 45 ØA
[HK5]    2Ø: ØA Ø5 96 Ø4 19 FD 6B 1D B3 Ø3 DF Ø4 23 FF 56 Ø4
[2P5]    21: 4A FF 56 Ø4 8F FF AC Ø8 42 Ø7 39 FF 2B Ø7 41 Ø3
[PT5]    22: 27 Ø9 ØC FE 81 88 Ø5 D8 Ø8 5C FE 88 88 Ø3 A6 FE
[4R5]    23: 88 2B Ø3 AB FE ØØ ØØ Ø3 3Ø Ø8 B2 FE 2B 2B Ø3 8D
[1U5]    24: Ø5 2B Ø5 87 FC 88 88 56 ØØ Ø5 CE FF 56 Ø3 EF FE
[9K5]    25: 88 88 Ø5 99 FB 47 B3 23 23 23 Ø5 E4 Ø5 68 Ø3 AD
[5L5]    26: FF 56 Ø4 56 Ø4 B1 FE 56 81 Ø4 BC Ø3 FE Ø6 82 Ø3
[DM5]    27: D6 Ø3 EC Ø6 CE Ø3 Ø5 Ø4 D7 FF 65 Ø3 E1 FC B3 23
[SG5]    28: 47 23 Ø4 27 Ø4 C9 Ø6 EB Ø5 FØ Ø3 38 Ø5 BC Ø5 FD
[LE5]    29: Ø4 C5 Ø5 E8 FA ØØ ØØ ØØ 2B ØØ Ø7 Ø5 12 Ø3 56 FB
[4L5]    3Ø: 47 47 23 23 1D Ø3 5B Ø5 E5 FE 88 AC Ø5 7C Ø4 2E
[6F5]    31: Ø7 71 Ø5 3E Ø4 7D Ø9 6F Ø3 4C FD ØØ Ø7 Ø7 Ø3 51
[FH5]    32: Ø3 53 Ø3 92 FD 88 47 1D Ø3 5C FE 17 1D Ø3 27 Ø6
[NH5]    33: 28 Ø5 ØB Ø6 7D Ø6 B5 ØB B1 Ø4 BB Ø5 C5 Ø3 8F Ø3
[WR5]    34: D2 Ø5 BØ FD ØF 1D 1D Ø3 9F FF ØF Ø9 3A Ø6 BC 12
[AM5]    35: B4 ØA BE Ø4 8D FF 81 Ø3 1Ø Ø3 93 Ø3 AA FE Ø8 ØF
[AG5]    36: Ø4 1C FF Ø8 Ø9 7Ø Ø9 86 ØC 33 Ø5 EA Ø5 52 Ø8 46
[LL5]    37: FE 81 81 Ø5 A2 FD 88 81 Ø8 Ø6 5B ØD 4D Ø6 A5 Ø3
[LN5]    38: 44 Ø4 78 Ø7 3D Ø5 45 FD AC AC AC Ø6 6B Ø3 7C Ø3
[FP5]    39: 88 FE AC AC ØB 35 Ø4 8D Ø4 98 Ø3 58 Ø4 95 Ø5 AE
[ØR5]    4Ø: FF AC Ø4 77 Ø6 7C Ø8 83 Ø6 CA FE 81 81 Ø7 9D FD
[GN5]    41: 81 ØØ Ø8 Ø5 DB Ø9 D2 Ø5 EA Ø7 CB Ø3 F4 Ø4 2D FF
[ZP5]    42: 2B ØB BF FE 56 56 Ø8 CE 1Ø 94 Ø5 F2 Ø5 E7 Ø3 ØE
[TG5]    43: Ø4 ØB Ø5 F5 FE 81 81 Ø4 79 Ø4 Ø1 Ø3 Ø4 FE AC 56
[CL5]    44: Ø4 38 Ø4 2C Ø6 63 Ø3 4E FF ØØ Ø7 DB Ø7 E9 Ø3 2F
[2D5]    45: Ø5 2C Ø7 7Ø Ø4 2E Ø3 BA Ø9 7F Ø5 69 Ø4 84 Ø5 Ø4
[AG5]    46: Ø5 6D ØD E3 Ø4 79 Ø8 5Ø Ø8 A3 Ø6 7D Ø4 C2 ØB AD
[MH5]    47: Ø7 83 Ø4 98 Ø7 DC Ø7 CC Ø8 84 Ø7 8A ØA 79 Ø3 C1
[3F5]    48: Ø5 F1 Ø6 F5 Ø4 Ø1 Ø6 F7 Ø8 9B Ø5 Ø7 Ø3 FB Ø6 E8
[8F5]    49: Ø8 E8 Ø3 ØF Ø5 FB Ø5 3D Ø4 32 Ø4 26 Ø8 ØD Ø4 4A
[5K5]    5Ø: FF 56 Ø6 DA Ø5 56 Ø5 3C Ø5 49 Ø5 57 Ø9 7Ø Ø6 7E
[1G5]    51: Ø7 54 ØA 61 Ø7 84 Ø5 9C Ø5 47 Ø8 4Ø ØE AD Ø8 FD
[TG5]    52: Ø3 A6 Ø5 83 Ø6 3B Ø7 BF Ø3 D8 FF Ø2 Ø4 DD Ø4 76
[8H5]    53: ØA 92 ØA B3 ØB CD Ø8 D2 Ø6 A6 ØA ØØ FF ØØ Ø3 1A
[8K5]    54: Ø3 9F ØC BD Ø4 C7 Ø4 E9 15 F8 ØB 14 Ø6 4F Ø4 5C
[YD5]    55: Ø6 2C Ø4 59 Ø7 ØD ØB 22 ØA 57 Ø3 5F Ø6 7F Ø4 52
[2H5]    56: ØA 86 Ø8 94 Ø5 9A ØF 3A ØA A1 14 C3 ØC BF Ø9 8Ø
[4H5]    57: ØA 9B ØC 55 22 Ø2 13 FD Ø8 25 Ø3 49 Ø7 4Ø ØA FA
[ZE5]    58: FF 56 Ø4 54 ØB 81 Ø4 47 Ø3 57 Ø6 BC Ø4 61 ØD 74
[DR5]    59: F9 2B 81 AC AC 88 88 88 Ø6 46 Ø3 6F Ø7 7C Ø6 6A
[TG5]    6Ø: Ø3 9D Ø5 65 Ø6 96 Ø4 59 Ø4 A4 Ø3 86 FA 88 88 AC
[XL5]    61: 81 81 81 Ø4 89 Ø3 BB FE 81 81 Ø3 85 FF AC Ø6 97
[HF5]    62: FD 56 56 81 Ø3 88 Ø6 A8 Ø5 AD Ø4 E1 Ø4 B3 Ø7 67
[5P5]    63: Ø5 B2 FF AC Ø4 F6 Ø4 BB Ø5 A4 Ø5 C2 Ø4 86 Ø6 ED
[WD5]    64: Ø4 87 ØA 94 Ø6 E2 Ø5 89 Ø6 69 Ø3 26 Ø4 FA Ø5 BB
[XH5]    65: Ø5 FE Ø4 3F Ø4 Ø7 Ø4 19 Ø4 52 FF AC Ø5 Ø3 Ø6 1A
[9B5]    66: Ø5 1F Ø6 Ø1 Ø4 A1 Ø5 4E ØB 34 Ø6 41 Ø8 Ø1 Ø4 2A
[4C5]    67: Ø3 9Ø Ø5 74 Ø6 EB Ø6 2F Ø4 84 Ø5 27 Ø7 6D ØB 36
[3M5]    68: Ø6 D8 Ø6 FF Ø8 AB Ø6 79 Ø6 BD Ø3 5F Ø7 18 Ø6 E8
[DK5]    69: Ø3 5E Ø5 A5 Ø7 17 Ø9 E4 Ø8 F9 Ø5 AF Ø4 13 FF Ø7
[LP5]    7Ø: Ø6 F8 Ø5 EF Ø6 D8 FF 56 Ø4 5F Ø4 ØF Ø5 63 ØB FC
[ØN5]    71: Ø7 C2 Ø7 EA ØC 3A FF ØØ Ø4 1F Ø9 3F Ø3 5D Ø3 6D
[PH5]    72: ØB BB Ø6 65 ØB 53 Ø7 1D ØC 75 ØE 7Ø Ø4 6E Ø6 7F
[SD5]    73: ØE 72 Ø7 1B Ø6 5B Ø4 29 Ø5 B1 Ø5 B4 Ø5 31 Ø8 C4
[GL5]    74: Ø6 5D Ø4 AE Ø4 EE Ø4 CB ØB A5 Ø7 CC Ø3 1Ø Ø4 12
[FE5]    75: Ø5 9Ø Ø5 Ø9 Ø8 BA Ø6 6B Ø8 66 Ø6 1D Ø7 87 Ø6 2B
[MF5]    76: Ø8 2C Ø7 36 Ø9 58 ØB A6 Ø6 4F Ø8 13 ØC FF ØC 88
```

```
[TC5]    77: Ø5 86 ØB 53 ØF 83 Ø9 51 Ø5 A3 Ø7 B3 Ø5 BB ØE 87
[XH5]    78: 26 DB Ø4 35 13 87 1D F6 ØA 14 FD 2B 2B 56 12 42
[P75]    79: Ø3 4Ø Ø6 A5 Ø5 36 Ø3 53 Ø5 51 Ø7 5Ø Ø4 6A Ø3 72
[GG5]    8Ø: Ø5 56 F5 ØØ ØØ Ø7 56 56 81 81 AC AC 81 AC Ø3 83
[3F5]    81: Ø4 87 FF 81 Ø3 82 FF 81 Ø5 57 Ø8 C3 Ø4 79 Ø5 91
[SS5]    82: Ø3 8B FF 88 Ø4 AE FF AC Ø3 B3 Ø3 81 ØB 7A FF 81
[6H5]    83: Ø9 AF Ø3 C3 Ø3 A8 Ø6 6E Ø7 5A Ø6 7Ø Ø5 C1 FF 56
[WM5]    84: Ø3 CB Ø5 8C Ø4 EC Ø5 D5 Ø7 3F Ø5 CF Ø6 CF Ø4 D3
[BE5]    85: ØC 95 Ø6 15 Ø8 E5 ØD Ø6 FE Ø7 Ø7 Ø9 3E Ø3 84 Ø3
[PG5]    86: EF Ø5 11 Ø7 13 Ø5 DA Ø5 9E ØA 22 Ø9 6C Ø4 57 Ø3
[3L5]    87: 74 Ø5 5F Ø5 B7 Ø6 73 ØA 4F Ø6 DF Ø3 DE Ø5 94 Ø6
[6G5]    88: 9E Ø6 63 Ø5 8B Ø7 64 Ø6 5F Ø5 D6 Ø8 54 Ø8 14 Ø6
[ØL5]    89: A6 Ø4 BD Ø8 EØ Ø5 C9 Ø5 7F Ø8 AB Ø6 7C 12 Ø1 Ø4
[HM5]    9Ø: DF Ø4 FE Ø8 5D Ø5 DF Ø8 12 Ø4 2C Ø8 AA Ø9 32 Ø7
[ØP5]    91: 37 ØB ØD Ø7 DA Ø6 3E Ø8 1D Ø6 E5 ØB 3F ØA 5D ØC
[UG5]    92: 4D Ø4 86 Ø4 4A Ø9 5C Ø8 D4 Ø8 9Ø ØA 73 Ø5 9C Ø8
[UP5]    93: C3 Ø9 BE Ø6 D5 Ø8 CF Ø6 6E Ø6 C8 ØF Ø6 Ø5 AE 12
[CS5]    94: C5 ØC Ø7 Ø7 8F ØA DB Ø5 FB Ø6 24 ØE BE ØC C1 Ø8
[5G4]    95: 35 ØC ØC Ø9 52 ØB D9 31 7E 34 7C 1C E3
```

Listing 13. mauer10.pak ist eine gepackte Grafikdatei.

```
[5R5]     1: FF 2B Ø8 FC FF 2B Ø6 Ø9 Ø3 1Ø FC 56 2B 56 56 Ø7
[6C5]     2: Ø9 ØF 1D Ø7 Ø3 FF Ø7 Ø5 34 Ø6 FB Ø5 ØØ Ø4 45 Ø3
[ZG5]     3: 16 Ø7 44 FF 56 Ø3 15 Ø3 53 ØC 59 Ø5 52 Ø7 2B FE
[SG5]     4: 2B Ø7 Ø3 67 Ø7 38 Ø7 3E FF 56 Ø4 82 Ø4 49 Ø4 6F
[NS5]     5: Ø3 74 FF 81 Ø6 6Ø FD AC AC 81 Ø5 9C FE 88 88 Ø5
[9C5]     6: 63 Ø7 ØD Ø3 76 FF 56 Ø3 73 Ø8 37 Ø3 84 Ø4 B8 Ø9
[1N5]     7: 16 Ø8 93 FF 88 Ø3 9B ØA DC Ø6 61 Ø3 AC Ø7 95 Ø4
[FP5]     8: C5 Ø3 F9 Ø5 BE Ø7 6Ø FD 2B 81 81 Ø5 EB Ø3 9A Ø4
[GN5]     9: D6 FF 88 Ø8 DF FF 23 Ø4 1E Ø3 A1 Ø6 E9 Ø3 ØB ØA
[KL5]    1Ø: FØ Ø3 C7 Ø5 87 Ø4 2A Ø5 27 Ø4 AC FE 81 2B Ø5 1D
[AC5]    11: Ø4 18 Ø3 59 FE 23 8F Ø3 22 Ø6 19 Ø5 A9 Ø3 3Ø Ø7
[HH5]    12: 72 Ø3 FB FE Ø7 Ø7 Ø4 1Ø Ø6 82 Ø9 49 FD 56 81 81
[MK5]    13: Ø3 64 Ø3 65 FF 23 Ø3 5C FB B3 8F 8F 23 47 Ø3 63
[N85]    14: Ø3 89 Ø6 29 Ø8 7Ø Ø9 71 Ø3 92 Ø4 3Ø Ø3 93 Ø6 8B
[NG5]    15: Ø5 CØ Ø5 63 F8 47 B3 88 23 B3 8F B3 47 Ø3 A3 Ø4
[5T5]    16: 56 Ø7 AA FF AC Ø4 B8 Ø7 F2 FF 81 Ø3 3C Ø7 F7 Ø4
[FL5]    17: 93 Ø8 CC Ø4 E7 Ø3 5E F9 47 B3 8F 23 B3 B3 B3 Ø5
[8M5]    18: A2 Ø6 E7 Ø3 2D FF AC Ø3 F1 Ø4 32 ØA B7 Ø4 C2 Ø4
[MR5]    19: 45 ØA ØA Ø6 96 Ø3 1A FB 6B 6B B3 B3 6B Ø5 22 FF
[5R5]    2Ø: 56 Ø4 4A FF 56 Ø4 8F FF AC Ø8 42 Ø7 39 FF 2B Ø7
[MN5]    21: 41 Ø4 64 Ø8 ØD FE 81 88 Ø3 18 FF 8F Ø3 5F FF 41
[GF5]    22: Ø4 5F Ø4 15 Ø3 49 FF 2B Ø3 AB FE ØØ ØØ Ø3 3Ø Ø8
[LK5]    23: B2 FE 2B 2B Ø3 8D Ø5 2B Ø5 87 FC 88 88 56 ØØ Ø5
[1R5]    24: CE FF 56 Ø3 EF FE 88 88 Ø3 5A Ø4 9C FD 23 23 23
[WG5]    25: Ø5 E4 Ø5 68 Ø3 AD FF 56 Ø3 56 Ø5 BØ FE 56 81 Ø4
[DP5]    26: BC Ø3 FE Ø6 82 Ø3 D6 Ø3 EC Ø6 CE Ø3 Ø5 FC 88 88
[YL5]    27: 47 47 Ø3 DF Ø3 1A FF 47 Ø3 E2 Ø3 A4 Ø4 Ø9 Ø5 EC
[VF5]    28: Ø5 FØ Ø3 38 Ø5 BC Ø5 FD Ø4 C5 Ø5 E8 FA ØØ ØØ ØØ
[LC5]    29: 2B ØØ Ø7 Ø5 12 Ø3 56 Ø4 1A FE 6B 47 Ø6 E1 Ø3 96
[7K5]    3Ø: Ø5 7C Ø4 2E Ø7 71 Ø5 3E Ø4 7D Ø9 6F Ø3 4C FD ØØ
[L95]    31: Ø7 Ø7 Ø3 51 Ø3 53 Ø3 92 F8 88 6B 41 23 41 1D 17
[3H5]    32: 1D Ø3 27 Ø6 28 Ø5 ØB Ø6 7D Ø6 B5 ØB B1 Ø4 BB Ø5
[EP5]    33: C5 Ø3 8F Ø3 D2 Ø5 BØ FD ØF 1D 1D Ø3 9F FF ØF Ø9
[ØK5]    34: 3A Ø6 BC 12 B4 ØA BE Ø4 8D FF 81 Ø3 1Ø Ø3 93 Ø3
[VN5]    35: AA FE Ø8 ØF Ø4 1C FF Ø8 Ø9 7Ø Ø9 86 ØC 33 Ø5 EA
[KG5]    36: Ø5 52 Ø8 46 FE 81 81 Ø5 A2 FD 88 81 Ø8 Ø6 5B ØD
[NP5]    37: 4D Ø6 A5 Ø3 44 Ø4 78 Ø7 3D Ø5 45 FD AC AC AC Ø6
[7N5]    38: 6B Ø3 7C Ø3 88 FE AC AC ØB 35 Ø4 8D Ø4 98 Ø3 58
[ØN5]    39: Ø4 95 Ø5 AE FF AC Ø4 77 Ø6 7C Ø8 83 Ø6 CA FE 81
[5N5]    4Ø: 81 Ø7 9D FD 81 ØØ Ø8 Ø5 DB Ø9 D2 Ø5 EA Ø7 CB Ø3
[CU5]    41: F4 Ø4 2D FF 2B ØB BF FE 56 56 Ø8 CE 1Ø 94 Ø5 F2
[SE5]    42: Ø5 E7 Ø3 ØE Ø4 ØB Ø5 F5 FE 81 81 Ø4 79 Ø4 Ø1 Ø3
[TM5]    43: Ø4 FE AC 56 Ø4 38 Ø4 2C Ø6 63 Ø3 4E FF ØØ Ø7 DB
[EG5]    44: Ø7 E9 Ø3 2F Ø5 2C Ø7 7Ø Ø4 2E Ø3 BA Ø9 7F Ø5 69
[3C5]    45: Ø4 84 Ø5 Ø4 Ø5 6D ØD E3 Ø4 79 Ø8 5Ø Ø8 A3 Ø6 7D
[ØL5]    46: Ø4 C2 ØB AD Ø7 83 Ø4 98 Ø7 DC Ø7 CC Ø8 84 Ø7 8A
[7E5]    47: ØA 79 Ø3 C1 Ø5 F1 Ø6 F5 Ø4 Ø1 Ø6 F7 Ø8 9B Ø5 Ø7
[WF5]    48: Ø3 FB Ø6 E8 Ø8 E8 Ø3 ØF Ø5 FB Ø5 3D Ø4 32 Ø4 26
[8K5]    49: Ø8 ØD Ø4 4A FF 56 Ø6 DA Ø5 56 Ø5 3C Ø5 49 Ø5 57
[ØC5]    5Ø: Ø9 7Ø Ø6 7E Ø7 54 ØA 61 Ø7 84 Ø5 9C Ø5 47 Ø8 4Ø
[KN5]    51: ØE AD Ø8 FD Ø3 A6 Ø5 83 Ø6 3B Ø7 BF Ø3 D8 FF Ø2
[9F5]    52: Ø4 DD Ø4 76 ØA 92 ØA B3 ØB CD Ø8 D2 Ø6 A6 ØA ØØ
[ZK5]    53: FF ØØ Ø3 1A Ø3 9F ØC BD Ø4 C7 Ø4 E9 15 F8 ØB 14
[ZF5]    54: Ø6 4F Ø4 5C Ø6 2C Ø4 59 Ø7 ØD ØB 22 ØA 57 Ø3 5F
[AF5]    55: Ø6 7F Ø4 52 ØA 86 Ø8 94 Ø5 9A ØF 3A ØA A1 14 C3
[CG5]    56: ØC BF Ø9 8Ø ØA 9B ØC 55 22 Ø2 13 FD Ø8 25 Ø3 49
[2K5]    57: Ø7 4Ø ØA FA FF 56 Ø4 54 ØB 81 Ø4 47 Ø3 57 Ø6 BC
[WM5]    58: Ø4 61 ØD 74 F9 2B 81 AC AC 88 88 88 Ø6 46 Ø3 6F
[TC5]    59: Ø7 7C Ø6 6A Ø3 9D Ø5 65 Ø6 96 Ø4 59 Ø4 A4 Ø3 86
[TN5]    6Ø: FA 88 88 AC 81 81 81 Ø4 89 Ø3 BB FE 81 81 Ø3 85
[UL5]    61: FF AC Ø6 97 FD 56 56 81 Ø3 88 Ø6 A8 Ø5 AD Ø4 E1
[9L5]    62: Ø4 B3 Ø7 67 Ø5 B2 FF AC Ø4 F6 Ø4 BB Ø5 A4 Ø5 C2
[AD5]    63: Ø4 86 Ø6 ED Ø4 87 ØA 94 Ø6 E2 Ø5 89 Ø6 69 Ø3 26
[2N5]    64: Ø4 FA Ø5 BB Ø5 FE Ø4 3F Ø4 Ø7 Ø4 19 Ø4 52 FF AC
[3B5]    65: Ø5 Ø3 Ø6 1A Ø5 1F Ø6 Ø1 Ø4 A1 Ø5 4E ØB 34 Ø6 41
[PB5]    66: Ø8 Ø1 Ø4 2A Ø3 9Ø Ø5 74 Ø6 EB Ø6 2F Ø4 84 Ø5 27
[EL5]    67: Ø7 6D ØB 36 Ø6 D8 Ø6 FF Ø8 AB Ø6 79 Ø6 BD Ø3 5F
[VH5]    68: Ø7 18 Ø6 E8 Ø3 5E Ø5 A5 Ø7 17 Ø9 E4 Ø8 F9 Ø5 AF
[NN5]    69: Ø4 13 FF Ø7 Ø6 F8 Ø5 EF Ø6 D8 FF 56 Ø4 5F Ø4 ØF
[ØP5]    7Ø: Ø5 63 ØB FC Ø7 C2 Ø7 EA ØC 3A FF ØØ Ø4 1F Ø9 3F
[2H5]    71: Ø3 5D Ø3 6D ØB BB Ø6 65 ØB 53 Ø7 1D ØC 75 ØE 7Ø
[2G5]    72: Ø4 6E Ø6 7F ØE 72 Ø7 1B Ø6 5B Ø4 29 Ø5 B1 Ø5 B4
```

```
[WN5]    73: Ø5 31 Ø8 C4 Ø6 5D Ø4 AE Ø4 EE Ø4 CB ØB A5 Ø7 CC
[9B5]    74: Ø3 1Ø Ø4 12 Ø5 9Ø Ø5 Ø9 Ø8 BA Ø6 6B Ø8 66 Ø6 1D
[DD5]    75: Ø7 87 Ø6 2B Ø8 2C Ø7 36 Ø9 58 ØB A6 Ø6 4F Ø8 13
[HE5]    76: ØC FF ØC 88 Ø5 86 ØB 53 ØF 83 Ø9 51 Ø5 A3 Ø7 B3
[6L5]    77: Ø5 BB ØE 87 26 DB Ø4 35 13 87 1D F6 ØA 14 FD 2B
[785]    78: 2B 56 12 42 Ø3 4Ø Ø6 A5 Ø5 36 Ø3 53 Ø5 51 Ø7 5Ø
[LF5]    79: Ø4 6A Ø3 72 Ø5 56 F5 ØØ ØØ Ø7 56 56 81 81 AC AC
[BG5]    8Ø: 81 AC Ø3 83 Ø4 87 FF 81 Ø3 82 FF 81 Ø5 57 Ø8 C3
[SL5]    81: Ø4 79 Ø5 91 Ø3 8B FF 88 Ø4 AE FF AC Ø3 B3 Ø3 81
[PK5]    82: ØB 7A FF 81 Ø9 AF Ø3 C3 Ø3 A8 Ø6 6E Ø7 5A Ø6 7Ø
[SN5]    83: Ø5 C1 FF 56 Ø3 CB Ø5 8C Ø4 EC Ø5 D5 Ø7 3F Ø5 CF
[DF5]    84: Ø6 CF Ø4 D3 ØC 95 Ø6 15 Ø8 E5 ØD Ø6 FE Ø7 Ø7 Ø9
[XG5]    85: 3E Ø3 84 Ø3 EF Ø5 11 Ø7 13 Ø5 DA Ø5 9E ØA 22 Ø9
[6H5]    86: 6C Ø4 57 Ø3 74 Ø5 5F Ø5 B7 Ø6 73 ØA 4F Ø6 DF Ø3
[HL5]    87: DE Ø5 94 Ø6 9E Ø6 63 Ø5 8B Ø7 64 Ø6 5F Ø5 D6 Ø8
[9K5]    88: 54 Ø8 14 Ø6 A6 Ø4 BD Ø8 EØ Ø5 C9 Ø5 7F Ø8 AB Ø6
[XK5]    89: 7C 12 Ø1 Ø4 DF Ø4 FE Ø8 5D Ø5 DF Ø8 12 Ø4 2C Ø8
[LL5]    9Ø: AA Ø9 32 Ø7 37 ØB ØD Ø7 DA Ø6 3E Ø8 1D Ø6 E5 ØB
[ØM5]    91: 3F ØA 5D ØC 4D Ø4 86 Ø4 4A Ø9 5C Ø8 D4 Ø8 9Ø ØA
[KP5]    92: 73 Ø5 9C Ø8 C3 Ø9 BE Ø6 D5 Ø8 CF Ø6 6E Ø6 C8 ØF
[PM5]    93: Ø6 Ø5 AE 12 C5 ØC Ø7 Ø7 8F ØA DB Ø5 FB Ø6 24 ØE
[SN5]    94: BE ØC C1 Ø8 35 ØC ØC Ø9 52 ØB D9 31 7E 34 7C 1C
[39Ø]    95: E3
```

Listing 14. mauer11.pak ist eine gepackte Grafikdatei.

```
[5R5]     1: FF FF 7F ØØ 22 7F FD 1D 1D 1D 3D 66 F8 16 16 16
[1T5]     2: 1D ØF FF Ø1 Ø1 Ø4 DD Ø5 E9 2F B4 F9 ØF 16 ØF ØF
[YR5]     3: 1D 16 Ø1 Ø6 E8 2D 2F FF Ø1 Ø3 A1 FF ØF Ø3 E2 Ø4
[2M5]     4: 24 36 AC F8 ØF 57 89 89 89 5E 57 16 Ø3 5E 36 69
[5R5]     5: Ø4 AØ FF 5E Ø4 A3 34 A8 Ø3 DD FC 57 FF 57 5E Ø3
[VS5]     6: 1F FD Ø8 16 1D Ø6 58 2E EF F1 Ø8 89 D7 97 D7 5E
[RT5]     7: D7 CØ D7 Ø8 Ø8 1D 1D FF 16 31 EC FF Ø8 Ø3 E1 Ø4
[1M5]     8: E1 FC 5E Ø8 Ø8 ØF 36 69 Ø3 A1 FF 57 Ø3 E3 FB Ø8
[FH5]     9: ØF Ø8 1D 16 32 AB F9 ØF 57 89 89 Ø3 Ø3 Ø3 Ø4 E4
[HR5]    1Ø: FE 16 1D 31 EB Ø4 DA FF 57 Ø4 2Ø FC 57 Ø8 16 16
[FR5]    11: Ø3 E8 Ø3 5A FF Ø1 3Ø EE F6 Ø8 Ø8 89 5E 57 FF Ø8
[DP5]    12: 16 ØF ØF 34 29 Ø4 1B FA 57 2C 2C FF FF ØF Ø3 27
[7R5]    13: 33 AB Ø3 1C Ø3 A1 FF 57 Ø3 E4 FE 16 16 Ø4 E8 FE
[GT5]    14: 1D ØF 35 EF Ø3 23 Ø3 9D 33 2A FB 89 89 5E 89 5E
[3T5]    15: Ø5 61 FE Ø8 1D Ø3 E7 FD 1D 1D 16 2F 33 FC 5E 57
[1V5]    16: 5E 5E Ø4 9F FF ØF Ø3 AC FC 16 1D 16 ØF 2D AE Ø3
[9P5]    17: 9E Ø3 6Ø FD 5E 89 57 Ø3 1B F9 1D 1D ØF ØF 16 16
[MØ6]    18: 16 2D 2E FE 5E CD Ø3 E3 FC 5E 89 CD CD Ø3 5E FF
[MZ5]    19: 57 Ø4 E8 FE FF 16 2D EE FB CD CD CD 57 57 Ø3 61
[CS5]    2Ø: FF C8 Ø4 5C F9 57 1D ØF Ø8 1D 16 Ø8 2C 7Ø FB C8
[LR5]    21: C8 C8 CD 57 Ø3 A1 Ø3 DC Ø4 64 FE 16 1D 2E ED Ø3
[HY5]    22: E2 FD C8 6C 98 Ø4 E1 FD C8 98 56 Ø4 A5 FF FF 2D
[AV5]    23: ED F8 FF 98 C8 98 98 6C 98 98 Ø3 22 F8 97 6C FF
[MM5]    24: FF 89 5E 57 57 2D 2D Ø3 58 FD 6D 6C 6C Ø3 5B Ø3
[VN5]    25: 5C Ø3 65 33 67 FE 89 6D Ø4 EØ FF 6D Ø3 9C Ø4 A5
[RX5]    26: 32 A8 FE 89 5E Ø4 DD FF C8 Ø3 E3 Ø5 E5 FD 5E 5E
[GP5]    27: 5E 3Ø ED Ø6 1C FE C8 98 Ø6 A5 2F 2C Ø8 5A Ø3 A2
[VT5]    28: 35 66 Ø4 E9 FF FF Ø4 9E FF 56 Ø3 A4 Ø4 DB 2F EA
[YU5]    29: FF 5E Ø5 DB FD C8 C8 81 Ø3 2A 32 A9 Ø7 E5 Ø3 DF
[TM5]    3Ø: FE 98 98 Ø3 25 31 29 Ø3 18 Ø4 5C FC C8 C8 C8 89
[DP5]    31: Ø3 A8 Ø3 E4 37 2A FF CD Ø3 A1 FF 89 32 A5 Ø3 E3
[TP5]    32: Ø7 DB Ø4 9F FE 56 6C 3Ø E6 FF 5E Ø4 19 Ø8 1C FB
[KU5]    33: C8 98 98 57 98 33 EA Ø4 5C FE CD C8 Ø3 5F FC 98
[NS5]    34: 57 57 57 2E 27 FE 89 5E Ø4 D3 FD CD C8 98 Ø4 AØ
[3R5]    35: FF 98 Ø3 57 2F A8 Ø5 16 Ø3 9F FE 98 98 Ø3 1C FF
[AT5]    36: 5E 3A E5 Ø4 DE Ø3 E4 Ø3 25 76 29 FE 6C 6C 7A A1
[CT5]    37: FF CD Ø7 1C FF 56 3B 24 FF 98 Ø9 2Ø FF 57 3Ø AA
[UZ5]    38: FF C8 Ø3 9C ØA 9E FF 5E 35 A9 FF 6C Ø5 1F FE 98
[WR5]    39: 89 33 E6 Ø3 1A Ø4 9D FF 98 Ø3 5C FF 2B 3B 25 Ø5
[LR5]    4Ø: 9B 37 65 Ø3 A2 FA 6C CD 98 C8 C8 2B Ø4 A4 36 A9
[ØN5]    41: Ø4 21 Ø4 62 FF 89 3C E8 Ø6 A2 32 2B Ø6 5C FF 6C
[EP5]    42: Ø3 22 32 E5 Ø5 59 Ø3 9D FE CD 56 3A A2 Ø4 1C FE
[WU5]    43: CD C8 Ø4 E2 FF 2B 31 E7 FF CD Ø3 58 FD 98 5E 57
[6R5]    44: Ø4 59 FC 6C C8 98 FF 32 27 Ø4 94 Ø3 68 Ø4 6A FF
[8N5]    45: 98 Ø4 65 Ø5 9F 2F AD Ø3 A7 11 D6 7F B1 2D 31 Ø9
[5A5]    46: 67 ØA 9C 2D 71 Ø3 DD Ø5 A9 Ø3 DC Ø7 DE 11 EF
```

Listing 15. mgirl7.pak ist eine gepackte Grafikdatei.

```
[AE5]     1: FA Ø3 Ø3 FF Ø1 Ø1 Ø3 Ø4 Ø5 FD Ø1 Ø3 Ø1 Ø5 Ø3 F9
[L75]     2: Ø1 Ø2 Ø2 Ø3 Ø2 Ø2 Ø4 12 FE Ø3 Ø3 Ø3 14 FE Ø3
[MH5]     3: Ø2 Ø3 ØE Ø4 1B Ø5 15 FE Ø1 Ø1 Ø4 21 FE FF FF Ø5
[MF5]     4: Ø6 FE Ø1 Ø3 Ø4 37 Ø3 ØA FD Ø3 Ø1 FF Ø3 44 Ø3 49
[U65]     5: Ø4 27 Ø4 24 Ø3 31 Ø3 11 Ø4 5A Ø6 23 Ø3 2C Ø6 29
[3P5]     6: FF FF Ø6 56 FF FF Ø3 3E Ø4 ØD Ø3 3F Ø4 ØC Ø4 84
[CD5]     7: Ø7 1C Ø5 4C Ø3 71 Ø4 72 Ø5 5F Ø3 6D Ø6 13 Ø7 A2
[NG5]     8: Ø3 97 Ø5 ØE Ø7 85 Ø6 B9 Ø5 83 Ø6 BC Ø3 5C Ø3 C8
[YM5]     9: Ø8 9F Ø8 AA Ø7 E7 FF FF Ø3 D2 Ø5 Ø5 Ø9 81 Ø7 ØC
[LR5]    1Ø: Ø6 CB Ø6 F4 FE Ø3 FF Ø5 EA Ø6 2F Ø3 77 FE FF 16
[YG5]    11: Ø4 25 Ø4 8A Ø4 48 Ø3 FØ Ø5 48 Ø5 F7 Ø4 7F Ø6 FE
[GG5]    12: Ø8 42 Ø5 4B Ø4 E3 Ø4 97 Ø3 15 Ø7 23 Ø4 67 FF FF
[DE5]    13: Ø3 6C Ø6 C6 Ø6 7C Ø4 7Ø ØA F6 Ø4 7E Ø4 B3 Ø4 54
```

Listing 16. miriam8.pak ist eine gepackte Grafikdatei.

```
[BM5]    14: Ø4 53 FE Ø1 FF Ø3 6Ø Ø3 9E FD 16 Ø8 Ø8 Ø8 64 FC
[1S5]    15: ØF ØF FF FF Ø5 4D Ø5 1B Ø9 85 Ø7 CB Ø4 31 Ø6 4B
[GR5]    16: Ø4 94 Ø4 32 Ø6 9E FF ØF Ø3 E1 FF ØF Ø7 66 FF ØF
[RF5]    17: Ø7 CF Ø4 D7 Ø6 ØØ Ø5 74 Ø5 BE Ø4 F5 Ø7 D1 Ø4 5B
[ØH5]    18: Ø7 19 Ø3 AB Ø3 AC Ø4 AB Ø5 E7 ØA BØ Ø3 6E Ø9 82
[ØC5]    19: Ø5 CA Ø4 3Ø Ø5 52 Ø4 54 Ø9 19 Ø3 E3 FF 16 Ø3 62
[5F5]    2Ø: Ø7 E7 FC 16 FF Ø3 Ø2 Ø5 94 Ø6 3E Ø7 Ø5 Ø5 85 Ø6
[VH5]    21: 7F Ø4 ØA Ø3 72 FE Ø2 FF Ø3 DB Ø6 A8 Ø4 66 Ø3 A2
[WK5]    22: Ø4 29 Ø4 2Ø Ø6 71 Ø4 14 Ø6 BE FF FF Ø7 FE FE Ø2
[NP5]    23: Ø3 Ø3 B4 Ø5 CA Ø4 38 FF Ø1 Ø5 99 FA ØF ØF 5E Ø8
[SR5]    24: 57 ØF Ø3 E1 FF 57 Ø3 23 FF Ø8 Ø5 29 FD ØF ØF FF
[UE5]    25: Ø5 95 Ø4 13 Ø3 49 FF FF Ø7 41 Ø4 Ø9 Ø5 14 Ø7 DØ
[PG5]    26: Ø5 98 Ø3 EB FF 57 ØA 1F Ø6 66 Ø3 FØ Ø3 F2 Ø4 11
[R85]    27: Ø5 9Ø Ø3 Ø7 Ø3 FF Ø5 Ø1 Ø4 B8 Ø5 B4 Ø3 52 Ø6 16
[LH5]    28: FC Ø8 57 57 5E Ø7 5F Ø5 26 FF Ø8 Ø3 2C FF Ø8 Ø4
[ØG5]    29: 31 Ø4 CC Ø4 B2 Ø3 7D FE Ø3 FF Ø3 8Ø Ø4 84 Ø4 Ø9
[ØH5]    3Ø: Ø4 4D Ø4 53 Ø4 14 Ø6 DA ØB 9E FE 57 57 Ø3 2D Ø3
[XE5]    31: 9A Ø3 97 Ø3 77 Ø5 B5 Ø5 85 Ø3 B3 Ø3 D6 Ø3 75 Ø4
[YL5]    32: C5 Ø4 8E Ø4 57 Ø5 98 ØA 5D Ø7 A4 FF Ø8 Ø6 2E Ø3
[ZB5]    33: 72 Ø4 32 Ø6 BB Ø3 82 Ø3 7C Ø6 48 Ø5 76 Ø4 54 Ø4
[UM5]    34: D6 FA ØF ØF Ø8 5E 5E ØØ Ø3 1F Ø3 E9 Ø6 1F Ø4 EC
[HL5]    35: Ø5 BØ Ø5 F3 Ø4 CF Ø4 FC Ø4 ØB Ø4 3B FE FF Ø2 Ø3
[2C5]    36: 45 Ø3 88 Ø7 13 FA 16 ØF Ø8 57 5E ØØ Ø3 68 Ø3 28
[NF5]    37: FF ØØ Ø5 5E Ø3 2B ØB 2D Ø3 F8 Ø4 3D Ø3 Ø4 Ø3 Ø2
[SF5]    38: Ø5 44 Ø4 87 FE Ø1 Ø2 Ø3 49 Ø4 94 Ø6 58 FC 5E D7
[7M5]    39: 97 97 Ø3 DC FB 57 57 97 97 D7 Ø4 2A Ø4 6D Ø4 FØ
[RA5]    4Ø: FE 16 16 Ø4 76 Ø5 3D Ø3 82 Ø5 Ø5 Ø3 89 Ø3 3A Ø4
[UM5]    41: 7A Ø3 D2 Ø5 16 Ø4 9B FF 5E Ø3 A7 Ø3 A3 Ø3 9F Ø4
[9H5]    42: E9 FF 16 Ø9 6D Ø4 B6 Ø3 CF Ø7 3F Ø3 Ø5 Ø5 3E Ø3
[DK5]    43: 12 Ø5 CC ØB D5 Ø5 1F Ø7 1C Ø3 2C FD Ø8 16 ØF Ø5
[5N5]    44: F1 Ø5 B6 FF Ø2 Ø4 Ø8 Ø7 C2 Ø4 Ø2 Ø5 C6 Ø3 DØ FF
[985]    45: Ø1 1Ø 15 Ø7 26 Ø3 2C Ø4 31 Ø4 37 ØA 37 Ø5 8Ø Ø3
[6M5]    46: 4F Ø4 CB Ø4 ØC ØF D1 Ø6 67 Ø6 AØ Ø5 6C Ø5 6D Ø6
[AR5]    47: 77 FF FF Ø3 BD FF Ø1 Ø4 41 Ø6 CB Ø3 FB Ø4 FA Ø9
[4N5]    48: 12 FF 16 Ø5 9B ØA 9C FF 57 Ø3 EE Ø3 B3 FF Ø8 Ø4
[9G5]    49: B3 Ø6 B8 FF Ø2 Ø4 49 Ø3 C1 Ø5 4E Ø6 ØF Ø3 Ø7 Ø5
[PN5]    5Ø: 54 Ø3 76 Ø8 9A Ø3 C3 ØA E6 Ø4 6F FF Ø8 Ø4 F6 Ø4
[BB5]    51: 38 Ø4 48 Ø3 3B Ø3 3C Ø3 3E Ø4 85 Ø4 11 FE Ø1 Ø3
[XH5]    52: Ø5 14 Ø3 D8 Ø5 1B FB Ø3 Ø3 D7 D7 D7 Ø3 24 Ø5 2A
[BB5]    53: Ø5 31 Ø3 3Ø Ø6 37 FF Ø2 Ø4 Ø2 Ø3 3F Ø4 FE Ø3 5Ø
[SG5]    54: Ø5 4E Ø4 CA Ø7 54 Ø3 6F Ø6 5D Ø4 23 FD 57 57 ØØ
[AA5]    55: Ø3 F2 Ø3 71 Ø3 6C Ø9 34 Ø4 47 Ø4 3C Ø5 Ø1 Ø4 D3
[BF5]    56: Ø3 8B Ø7 D2 Ø7 57 Ø7 1C Ø4 A7 Ø7 AA Ø3 73 Ø8 B4
[ZK5]    57: Ø3 CC FF Ø2 Ø6 BE Ø4 Ø5 Ø5 FA Ø4 D1 ØB D3 FC 16
[FL5]    58: Ø8 Ø8 Ø8 Ø5 E3 FB 2C 2C FF FF Ø8 Ø5 99 Ø4 FØ Ø6
[BB5]    59: 35 Ø5 1Ø Ø3 46 Ø5 B9 FE Ø1 Ø2 Ø3 53 Ø4 Ø6 Ø9 12
[DG5]    6Ø: FE FF ØØ Ø3 2D FF ØF Ø3 69 FF 2C Ø5 63 Ø3 29 Ø3
[CR5]    61: EB Ø6 2F Ø8 F6 FF FF Ø4 FF Ø4 ØØ Ø3 C3 Ø4 41 FF
[6C5]    62: Ø1 Ø4 3A Ø3 5Ø FE FF ØØ Ø5 96 Ø4 1A Ø3 E6 Ø3 26
[PN5]    63: Ø5 66 FF FF Ø3 71 Ø4 AC Ø8 73 Ø3 BA Ø5 BE Ø3 84
[UF5]    64: Ø4 4F Ø6 4E Ø3 4B Ø6 96 Ø6 97 FF ØØ Ø5 E4 Ø4 A5
[FP5]    65: Ø3 DC FF FF Ø3 1D Ø5 BØ Ø8 B4 FF ØØ Ø4 CØ FD Ø3
[9G5]    66: Ø1 FF Ø3 48 Ø4 CA Ø3 8E Ø3 Ø9 ØC D2 FD ØØ ØØ 57
[SB5]    67: Ø3 E1 Ø4 25 Ø7 E8 Ø3 F1 Ø8 32 Ø3 33 FD ØF ØØ Ø3
[2A5]    68: Ø4 89 Ø4 C9 Ø3 1Ø Ø4 Ø4 Ø3 DD ØE 51 Ø3 2Ø Ø6 EØ
[SH5]    69: FE 57 ØØ Ø3 AB FF Ø8 Ø5 31 Ø3 AE Ø5 F8 FF 16 Ø3
[S85]    7Ø: 76 Ø6 83 Ø4 44 Ø3 4B Ø3 87 FE ØØ Ø2 11 52 Ø4 6Ø
[RM5]    71: FF 57 Ø4 28 FD ØF FF Ø8 Ø4 2E Ø3 AE Ø5 35 Ø3 3Ø
[GG5]    72: Ø4 7E Ø4 3F Ø4 ØE FD Ø3 Ø2 ØØ Ø3 CD Ø7 99 Ø4 62
[9S5]    73: Ø9 D2 Ø5 63 FE 57 57 Ø4 AB Ø4 AE FD Ø8 16 FF ØC
[TR5]    74: B6 Ø4 8A FF FF Ø4 87 FF Ø3 Ø4 ØD ØA D1 ØD 19 Ø4
[VL5]    75: E7 FF 57 Ø4 EC Ø3 7Ø Ø4 7Ø Ø6 F8 Ø3 36 Ø5 8Ø FD
[7B5]    76: Ø3 Ø2 Ø3 Ø3 Ø8 FF Ø2 Ø3 Ø2 Ø3 CC Ø9 12 Ø6 6Ø ØB
[RD5]    77: 6Ø Ø7 AC Ø3 74 Ø3 73 Ø5 3Ø FF FF Ø3 42 Ø3 3E Ø4
[FL5]    78: 46 Ø3 49 Ø3 Ø3 Ø6 CD ØA 21 ØE 5F FE ØF FF Ø3 34
[RM5]    79: Ø4 33 Ø3 B5 FF Ø8 Ø5 7B Ø3 BF FE Ø1 Ø3 Ø3 ØD FF
[ZN5]    8Ø: Ø3 Ø3 Ø9 FE Ø1 ØØ ØD 4E Ø7 97 ØC 1D Ø4 78 FC FF
[RM5]    81: ØF ØF Ø8 Ø3 6E Ø4 B6 FD Ø8 FF ØF Ø3 Ø6 Ø5 BF Ø3
[RK5]    82: Ø7 Ø3 ØC Ø3 ØD ØE DØ ØE 1E Ø3 EB Ø4 F7 Ø3 77 Ø3
[KD5]    83: BD Ø3 38 FF Ø8 Ø3 34 Ø4 4Ø FF Ø1 Ø3 Ø3 Ø4 Ø8 Ø4
[SC5]    84: CD ØB 9Ø 13 15 Ø6 B2 Ø8 33 Ø4 7C Ø3 4Ø Ø7 Ø4 Ø3
[2H5]    85: C3 ØE 8D 14 5A FF 57 Ø4 B7 Ø4 72 Ø3 FØ Ø3 B3 Ø4
[2H5]    86: BE Ø4 81 Ø4 44 Ø3 ØA Ø4 8E 1D 91 Ø3 AC Ø4 B3 Ø5
[US5]    87: 6F Ø4 BC Ø3 BE FF Ø1 Ø4 42 Ø4 CB Ø6 8C Ø6 DØ FE
[RP5]    88: AC 57 ØE 18 ØA 16 FF 57 Ø4 6D FD ØF ØF Ø8 Ø3 37
[HH5]    89: FD 16 ØF Ø8 Ø5 7E Ø3 43 Ø3 86 Ø3 ØA FE Ø1 FF Ø3
[TH5]    9Ø: 12 Ø4 51 FE AC AC ØE 19 Ø4 64 Ø7 27 FF 57 Ø3 3Ø
[TP5]    91: FF ØF Ø3 34 Ø5 77 Ø5 3D Ø3 C8 FF Ø3 Ø3 44 FD FF
[EG5]    92: FF Ø2 Ø3 4C Ø4 51 Ø4 53 ØC 56 Ø3 95 FF 81 ØA 66
[UP5]    93: FF 5E Ø3 AA Ø4 33 Ø3 36 Ø6 FC FE Ø3 FF Ø4 CA Ø4
[ØN5]    94: ØA FF ØØ Ø7 ØD Ø5 64 Ø8 97 Ø5 D5 FF 81 ØB A6 Ø3
[MA5]    95: 7Ø Ø4 71 Ø6 B8 FF ØØ Ø3 8Ø Ø3 81 Ø4 Ø5 FD Ø3 Ø2
[TH5]    96: Ø3 Ø8 CE FF 81 Ø3 15 Ø5 D7 FE 57 57 Ø7 E1 Ø5 16
[6A5]    97: Ø4 EB Ø5 2C Ø5 F4 Ø5 9Ø Ø4 Ø2 Ø4 Ø4 Ø3 8B Ø4 Ø8
[8E5]    98: Ø4 ØE Ø6 24 Ø6 27 ØB 1F FC 81 81 56 56 Ø4 34 Ø7
[ZK5]    99: EE FE ØØ FF Ø7 3B Ø3 43 Ø5 Ø4 Ø3 4F Ø3 88 Ø6 D1
[3H5]   1ØØ: Ø7 66 FF 56 Ø6 D4 Ø9 94 Ø4 AB Ø6 EB Ø3 75 Ø4 79
[AD5]   1Ø1: Ø5 7E FE Ø3 Ø3 Ø3 49 Ø4 C8 FD Ø1 Ø1 ØØ Ø3 8F Ø7
[TT5]   1Ø2: 12 Ø4 AC ØA 98 ØF A6 Ø4 6D Ø3 CF Ø3 CE FE Ø3 FF
[TG5]   1Ø3: Ø4 C2 FF Ø2 Ø3 84 Ø6 89 Ø3 F9 Ø6 DB Ø9 D8 Ø4 1A
[CL5]   1Ø4: ØA E6 ØA AF FF ØØ Ø5 4D Ø3 ØØ Ø3 CØ Ø3 43 FE Ø2
[SG5]   1Ø5: Ø2 Ø3 41 Ø4 7A Ø3 1Ø Ø8 15 Ø5 59 ØF 6Ø ØA FØ FB
[585]   1Ø6: Ø2 Ø3 Ø2 Ø3 ØØ Ø3 C4 Ø4 4Ø Ø3 47 Ø3 C5 Ø6 ØC Ø8
[YF5]   1Ø7: 51 Ø4 5E ØA 26 Ø7 96 FF ØØ ØB 7Ø Ø3 49 Ø4 7E Ø3
[4P5]   1Ø8: 3F Ø5 C5 FF Ø1 Ø6 8E Ø4 DØ FF 5E Ø7 97 12 5D ØD
[TV5]   1Ø9: AF Ø3 BA FF FF Ø4 43 Ø4 84 Ø4 Ø5 Ø9 CB Ø3 F2 FF
[NW5]   11Ø: AC Ø7 D8 ØB 5B Ø4 AC ØB EF FF ØØ Ø4 F9 FF FF Ø4
[UR5]   111: Ø1 Ø3 Ø3 FF FF Ø4 88 ØA CC Ø3 16 FF AC Ø3 18 Ø7
[KS5]   112: 5C Ø6 5B Ø4 EC ØC 2E Ø4 FB FE ØØ FF Ø3 FE FF Ø1
```

```
[NS5]   113: Ø3 ØØ Ø4 FA FE Ø1 Ø3 ØB 4D FE 5E AC Ø6 67 ØD DB
[YU5]   114: 1Ø 6D Ø3 FD Ø6 8Ø Ø4 45 Ø3 85 FF FF ØC 8E FF AC
[5K5]   115: Ø4 99 Ø4 63 ØB 25 Ø9 AE FF 5E Ø3 38 FE Ø3 ØØ Ø4
[CT5]   116: BD Ø3 C9 Ø4 Ø4 Ø4 C1 Ø9 4A Ø5 F1 FD AC 81 AC Ø4
[GH5]   117: E6 Ø6 66 Ø5 E2 ØB EC FC 57 57 Ø2 ØØ Ø3 ØØ Ø3 3A
[RM5]   118: FF Ø2 Ø4 42 Ø3 FD Ø3 89 ØF ØC Ø3 1A FF AC ØA A3
[9H5]   119: Ø6 2A ØC 6F FF ØØ Ø4 42 Ø3 44 Ø5 C2 Ø5 48 Ø9 CD
[TF5]   12Ø: Ø3 7Ø Ø4 59 ØA 9C Ø3 26 FF 81 Ø7 6B Ø4 96 Ø7 B6
[EG5]   121: Ø4 7E Ø3 88 Ø3 45 Ø5 87 FF Ø2 Ø9 4D Ø4 B2 FE ØØ
[Z75]   122: AC Ø5 9A ØE 62 FF ØØ Ø3 B6 Ø7 B3 FF ØØ Ø5 7B
```

Listing 16. miriam8.pak ist eine gepackte Grafikdatei.

```
[3L5]     1: Ø3 ØØ FE 65 65 3C CD FA Ø8 16 65 65 16 Ø8 Ø9 47
[HM5]     2: ØA 46 1F 5A Ø7 41 FE Ø8 ØF Ø4 7A F8 ØF Ø8 Ø8 65
[PC5]     3: 65 65 ØF Ø8 Ø3 86 Ø4 9Ø Ø5 4F Ø4 91 Ø3 89 FF 65
[KD5]     4: Ø3 8D Ø8 98 Ø3 93 Ø4 A6 Ø6 9E Ø7 82 Ø7 B9 Ø4 B4
[DM5]     5: Ø3 8C Ø5 B4 Ø8 8D Ø4 CF Ø5 DØ Ø9 D9 ØB E4 1A B9
[DH5]     6: ØB ØA Ø7 F2 Ø9 1B ØA 17 ØB F8 FE Ø8 Ø8 Ø9 Ø5 Ø7
[5K5]     7: 24 FF ØF ØD 17 18 23 Ø3 EF Ø4 3E Ø5 AA Ø8 F1 11
[HR5]     8: 67 ØE 6Ø ØB 9A Ø4 81 Ø5 7A ØA C2 18 8C ØE D6 1A
[PN5]     9: F2 2C ØB 7F F8 7F 77 4B 36 FC Ø8 16 Ø8 65 Ø3 7F
[NN5]    1Ø: 7F 88 7F C7 7F 46 75 Ø5 FC Ø8 Ø8 Ø8 16 Ø3 7A 7F
[X36]    11: 81 7F CØ 7F 7F 7F FE 7F 7D 7F FC 7F 7B 7F FA 4Ø
[HK5]    12: 79 FF Ø8 Ø3 87 Ø3 B8 41 CØ FD Ø8 16 Ø8 Ø6 B7 7F
[4R5]    13: ØA 7F 49 7F Ø8 7F 87 7F Ø6 7F 85 7F Ø4 ØD 83 12
[3M5]    14: 87 17 A2 FF Ø8 Ø7 BA Ø5 B9 FF Ø8 2Ø 87 11 99 Ø6
[5C5]    15: F8 Ø3 BF Ø4 86 Ø5 ØØ Ø8 B2 1Ø Ø9 18 E2 Ø6 Ø2 Ø3
[1D5]    16: 4Ø Ø3 CØ ØA 46 Ø3 3E Ø8 32 2Ø DB Ø5 43 1Ø 8Ø Ø3
[XY5]    17: 4F 13 53 ØE 19 ØA F4 1C 7E ØC A7 FF ØF ØB C6 1A
[HR5]    18: B2 FF Ø2 Ø4 ØC Ø8 E7 FF ØF Ø8 99 Ø3 ØE ØA E5 Ø4
[HF5]    19: 22 17 F3 FC Ø8 Ø2 Ø2 ØØ Ø3 4D FB Ø2 Ø8 65 Ø2 Ø8
[ZC5]    2Ø: Ø3 55 Ø5 19 Ø3 4A Ø4 4C Ø4 57 Ø7 D4 FF ØØ Ø7 3Ø
[AH5]    21: Ø8 4Ø FE ØØ Ø8 Ø4 82 Ø3 7F Ø4 4D ØD 8C ØC 88 Ø4
[CE5]    22: 55 Ø5 A9 Ø4 4E Ø6 97 Ø7 81 Ø3 B7 Ø3 76 11 89 23
[4ZØ]    23: D6 Ø6 C2
```

Listing 17. rolle.pak ist eine gepackte Grafikdatei.

```
[6U5]     1: FD ØF ØF 1D 33 Ø2 FF 41 Ø5 36 FF ØF Ø4 3C FD 1D
[YH5]     2: 1D 16 38 43 FB 1D 1D ØF ØF 41 3A 42 Ø5 7A 3C 81
[KY5]     3: FD 16 16 Ø8 7F BF 7F 3E 7F FD 7F 7C 7F FB 7F 7A
[CR5]     4: 65 F9 FF Ø5 33 5F ØC 5E FF ØB 1F 52 15 BE ØB 9E
[KR5]     5: ØC 9E ØD 9E 1D F7 14 EA 16 5D 17 3E Ø8 14 FD ØØ
[FN5]     6: ØØ ØØ 1D ØC 18 3D Ø7 67 FC ØØ AD AD AD Ø7 5F 16
[6U5]     7: 68 19 7D Ø7 D6 Ø3 9D 1C C1 21 BC FF 16 1F DF 1D
[5R5]     8: FD F7 B3 Ø8 Ø9 B3 AD B3 Ø9 Ø8 B3 36 23 FF B3 Ø4
[SS5]     9: 5A FF B3 Ø4 5F 1B 62 1C 7E F7 AC Ø8 Ø9 Ø9 B3 Ø9
[GN5]    1Ø: Ø9 Ø8 AC 39 A3 FB Ø8 Ø9 AC Ø9 Ø8 37 E1 FC B3 B3
[SM5]    11: AC 16 Ø3 2Ø FC Ø8 Ø8 16 AC 1B A2 14 7E FC Ø5 ØB
[7F5]    12: ØB ØB Ø3 57 Ø3 64 FB ØØ Ø8 47 Ø8 ØØ Ø4 56 FE B3
[ER5]    13: 16 Ø3 93 FF ØB 2C AB FE D1 D1 Ø5 55 Ø5 97 Ø6 D6
[XN5]    14: 3Ø E9 Ø7 DA 1B 63 2Ø FB FA 16 12 ØD 16 ØD 12 1C
[LR5]    15: 22 1F 3D FE ØC 12 Ø4 5B 3B 22 FF ØC Ø4 9B 3A A2
[8M5]    16: FE ØØ ØØ Ø4 1B 7F E2 35 A1 FF Ø5 Ø5 98 FF AC Ø3
[US5]    17: 9D Ø3 DC Ø4 E3 FF ØB 2D E8 Ø5 D6 ØE 19 16 E7 16
[GG5]    18: 3E ØA 15 ØB 28 15 28 15 7E ØB 15 ØC 69 14 69 14
[NV5]    19: BE ØC 15 ØD AA 13 15 17 FE 29 14 7F FE 7F 7D 7F
[NS5]    2Ø: 3C 7F BB 7F 3A 7F B9 Ø8 38 FF 41 Ø4 Ø1 FF Ø8 3C
[WK5]    21: 43 FC Ø8 ØF ØF Ø8 3A 81 Ø4 BD 3B BF Ø6 FC Ø4 ØØ
[XY1]    22: 37 44 Ø6 3D 7E 81
```

Listing 18. self3.pak ist eine gepackte Grafikdatei.

```
[FØ6]     1: FF FF 7F ØØ 56 7F FF C8 Ø9 CD FF C7 34 A2 FF CD
[DM5]     2: ØB 16 33 EØ FF CE ØA D6 Ø3 1F 32 21 FF C8 ØC 62
[ØX5]     3: FE C8 C2 32 63 FF C1 3F 96 FF 9Ø 17 CA 28 ED F4
[1R5]     4: C2 9Ø 81 AC AC AC 81 81 81 56 C8 C1 ØB 15 2A 2B
[9T5]     5: FF 81 Ø4 5A Ø4 9A FF 56 34 61 F8 FF 56 81 81 1D
[9S5]     6: 1D 1D 56 Ø5 DB 36 A1 F8 1D 16 6B 6B 6B 16 16 ØF
[TP5]     7: 39 DF FF ØØ Ø3 D8 FF ØØ 3C 1D FD ØØ 6B ØØ 3B 5C
[8U5]     8: F9 24 6B 6B ØØ 6B 6B ØF 3A 9E F9 3A 1D 6B 1D 3A
[1P5]     9: ØF 24 36 1F FD FF 56 24 Ø3 97 FB ØØ 1D ØF 24 2B
[PK5]    1Ø: 35 21 Ø3 94 FF ØØ Ø3 19 FE ØF ØØ 38 6Ø Ø4 95 FF
[PN5]    11: 16 33 9C FC AC AC AC 81 Ø3 2Ø FD 16 1D 16 Ø3 13
[LR5]    12: FF ØF Ø3 ØB FE ØF ØF Ø3 1B FE 56 81 Ø3 ØE 29 EA
[5L5]    13: FA 56 31 37 3C 3C 16 Ø3 15 F6 16 1D ØF ØE ØE ØE
[9E5]    14: 14 31 Ø6 56 28 24 Ø3 4E Ø3 5Ø F5 ØD 37 3C 41 41
[BC5]    15: 41 47 41 16 16 16 Ø3 1F FE 14 ØD Ø4 63 Ø3 66 23
[FL5]    16: AA FF 81 Ø4 8F FB ØC 38 38 3C Ø6 Ø3 D6 FE 41 16
[RN5]    17: Ø4 D6 FF Ø6 Ø3 A1 Ø5 A3 FF 81 24 EA FF 56 Ø3 DØ
```

```
[SM5]    18: FD ØC 37 5B Ø3 95 FE 47 1D Ø3 59 Ø4 9C FD 14 31
[CS5]    19: 31 Ø4 E3 FF 56 24 29 FC ØC ØC ØC ØE Ø3 11 FF 37
[XL5]    2Ø: Ø5 D5 FF 1D Ø5 DB FE 14 14 Ø5 DC 24 29 Ø5 4B Ø3
[GR5]    21: E7 Ø4 92 FF 1D Ø8 18 FE 14 ØC Ø4 22 FD Ø6 ØC ØC
[4U5]    22: 23 AC FD AC AC AC Ø4 CB Ø4 CC ØA 57 FE 31 ØC 25
[FT5]    23: A3 Ø5 CA FE AC Ø8 Ø3 ØC Ø3 11 FD AC 81 1D Ø3 97
[EH5]    24: ØB 9C Ø3 E6 1F E8 FF 81 Ø3 1Ø FD Ø3 Ø3 Ø3 Ø4 11
[MH5]    25: Ø4 13 Ø8 57 Ø7 1F FD ØD ØC 31 23 29 Ø3 4B Ø8 4F
[ZR5]    26: FE ØD 1D Ø4 99 Ø5 1E FF D4 Ø3 A2 FF D3 21 27 Ø7
[FS5]    27: 47 Ø8 8F FE 81 ØD Ø4 D9 FD 14 14 31 Ø3 AØ FF CD
[NS5]    28: Ø3 E3 FD ØC ØC 31 25 AA ØC 8E Ø4 9C Ø3 63 F8 D4
[6Z5]    29: CD CE CE C7 ØC ØC ØD 1F E9 FF 81 ØE C9 ED 38 3C
[DN5]    3Ø: 16 16 14 ØD 31 31 ØE Ø6 CD CD C8 C8 C8 C7 ØC C9
[BR5]    31: ØD 2B EA F7 AC 81 Ø8 81 C9 81 Ø8 Ø8 Ø7 Ø3 9E FE
[9N5]    32: FF FF Ø3 64 FD C8 ØF ØF 21 29 Ø3 8B Ø8 4D Ø6 96
[FV5]    33: FE Ø2 Ø2 Ø5 9F F8 FF FF ØD C7 C2 ØF 16 16 2B 2A
[7P5]    34: Ø3 16 FC C9 C9 C9 ØD Ø4 1C FE Ø6 Ø6 Ø3 E3 FB ØF
[NP5]    35: ØF 16 4Ø 31 1F EB ØC C9 Ø3 16 FE ØD 14 Ø5 1C FF
[RT5]    36: ØE Ø4 21 FF C9 22 E8 Ø5 48 Ø8 ØC FF 81 Ø9 56 FF
[UK5]    37: Ø6 Ø3 6Ø Ø3 62 25 A5 Ø7 ØA Ø3 91 F7 AC AC 81 ØC
[UG5]    38: 14 ØC ØC ØC Ø6 Ø5 9F 28 A3 Ø8 CA Ø4 54 Ø3 D7 Ø4
[2U5]    39: D9 FF ØC 32 DE Ø6 11 FF ØC Ø4 D8 FE Ø6 Ø6 Ø3 1E
[HE5]    4Ø: 2B AØ Ø9 4B Ø4 55 Ø5 58 Ø4 DC Ø3 61 29 65 Ø4 8D
[LR5]    41: FE Ø3 Ø3 Ø4 D5 Ø5 18 FF ØE Ø4 DE 2D A2 Ø5 CD Ø5
[BG5]    42: 95 Ø3 99 Ø5 6Ø 2C 62 Ø7 ØB FA AC 81 Ø6 31 31 ØE
[FR5]    43: Ø6 59 32 21 FD FF FF ØD Ø6 1E Ø5 9A FF ØF 31 63
[AY5]    44: FC FF FF ØF ØF Ø8 AØ ØA D4 38 A8 FE 81 56 33 E2
[RS5]    45: FE 16 16 Ø7 4D FE 81 AC 7F 2Ø 38 9F FE 5Ø 5Ø Ø6
[TM5]    46: D9 FF 16 Ø9 D5 2B 29 FD 75 75 75 Ø6 17 FF 6F Ø6
[CT5]    47: 5D 31 25 FE 4A 4A Ø7 17 FF 4A Ø5 9E 31 24 FD 4A
[XM5]    48: 4A 75 Ø6 98 FD FF 6F 75 31 62 Ø5 91 Ø8 D7 FF 4A
[KB5]    49: Ø3 14 31 23 Ø3 13 ØA 17 32 A2 Ø5 1D FF 5Ø Ø6 58
[3N5]    5Ø: 36 5F ØA 54 FE 4A 6F 34 21 FC FF 4A 5Ø 5Ø Ø6 D8
[4T5]    51: FF 4A Ø4 1F 3C E4 FF 6F 35 21 Ø3 97 FF 5Ø Ø4 5A
[PN5]    52: FD 4A 6F 6F Ø3 6Ø 32 A4 Ø4 97 FF 75 Ø8 1C 1D A3
```

Listing 19. skell1.pak ist eine gepackte Grafikdatei.

```
[FØ6]     1: FF FF 7F ØØ 56 7F FF C8 Ø9 CD FF C7 34 A2 FF CD
[DM5]     2: ØB 16 33 EØ FF CE ØA D6 Ø3 1F 32 21 FF C8 ØC 62
[ØX5]     3: FE C8 C2 32 63 FF C1 3F 96 FF 9Ø 17 CA 28 ED F4
[1R5]     4: C2 9Ø 81 AC AC AC 81 81 81 56 C8 C1 ØB 15 2A 2B
[9T5]     5: FF 81 Ø4 5A Ø4 9A FF 56 34 61 F8 FF 56 81 81 1D
[9S5]     6: 1D 1D 56 Ø5 DB 36 A1 F8 1D 16 6B 6B 6B 16 16 ØF
[TP5]     7: 39 DF FF ØØ Ø3 D8 FF ØØ 3C 1D FD ØØ 6B ØØ 3B 5C
[8U5]     8: F9 24 6B 6B ØØ 6B 6B ØF 3A 9E F9 3A 1D 6B 1D 3A
[1P5]     9: ØF 24 36 1F FD FF 56 24 Ø3 97 FB ØØ 1D ØF 24 2B
[PK5]    1Ø: 35 21 Ø3 94 FF ØØ Ø3 19 FE ØF ØØ 38 6Ø Ø4 95 FF
[PN5]    11: 16 33 9C FC AC AC AC 81 Ø3 2Ø FD 16 1D 16 Ø3 13
[LR5]    12: FF ØF Ø3 ØB FE ØF ØF Ø3 1B FE 56 81 Ø3 ØE 29 EA
[5L5]    13: FA 56 31 37 3C 3C 16 Ø3 15 F6 16 1D ØF ØE ØE ØE
[9E5]    14: 14 31 Ø6 56 28 24 Ø3 4E Ø3 5Ø F5 ØD 37 3C 41 41
[BC5]    15: 41 47 41 16 16 16 Ø3 1F FE 14 ØD Ø4 63 Ø3 66 23
[FL5]    16: AA FF 81 Ø4 8F FB ØC 38 38 3C Ø6 Ø3 D6 FE 41 16
[RN5]    17: Ø4 D6 FF Ø6 Ø3 A1 Ø5 A3 FF 81 24 EA FF 56 Ø3 DØ
[SM5]    18: FD ØC 37 5B Ø3 95 FE 47 1D Ø3 59 Ø4 9C FD 14 31
[CS5]    19: 31 Ø4 E3 FF 56 24 29 FC ØC ØC ØC ØE Ø3 11 FF 37
[HG5]    2Ø: Ø4 D5 FF D4 Ø6 59 F9 14 14 ØC ØC Ø7 Ø6 31 24 29
[SK5]    21: Ø5 4B Ø3 E7 Ø4 92 FE 1D 47 Ø8 59 F8 ØC 31 ØC 31
[KP5]    22: 31 31 ØC Ø6 23 AC FD AC AC AC Ø4 CB Ø4 CC Ø3 57
[9T5]    23: FF CE Ø4 DA FE D4 D4 Ø3 A2 FE Ø6 Ø7 22 A6 Ø5 CA
[4S5]    24: FE AC Ø8 Ø3 ØC Ø3 11 FC AC 81 1D 1D Ø6 D9 Ø4 DE
[9C5]    25: FB Ø6 31 Ø7 31 31 21 E8 Ø3 1Ø FD Ø3 Ø3 Ø3 Ø4 11
[7B5]    26: Ø4 13 Ø8 D7 Ø4 1D Ø3 22 Ø3 26 1F 28 Ø4 48 Ø3 4B
[G26]    27: Ø8 4F FA ØD 1D D4 CD CD CD Ø3 5F F7 C7 C7 C7 CE
[SH5]    28: 31 41 31 ØC FF 26 28 Ø8 8F F7 81 ØD Ø6 1D 16 Ø6
[MF5]    29: 14 14 CD Ø3 AØ FB 41 41 31 31 Ø7 27 A8 ØB 8E F2
[ZG5]    3Ø: 1D 16 ØF 14 ØC Ø6 Ø6 C7 C2 41 4Ø 41 16 ØD 1F EA
[CB5]    31: Ø3 Ø7 ØE C9 F1 38 3C 16 16 14 ØD 31 31 ØE Ø6 41
[RE5]    32: 31 16 41 ØD 22 27 ØD Ø8 F7 AC 81 Ø8 81 C9 81 Ø8
[MM5]    33: Ø8 Ø7 Ø3 9E FC FF FF 16 16 Ø3 A1 25 69 Ø8 4D Ø6
[WW5]    34: 96 FE Ø2 Ø2 Ø5 9F FE FF FF Ø6 DF 2B 2A Ø3 16 FC
[PR5]    35: C9 C9 C9 ØD Ø4 1C FE Ø6 Ø6 23 E7 1Ø C5 Ø3 16 FE
[GN5]    36: ØD 14 Ø5 1C FF ØE 2C 21 Ø8 ØC FF 81 Ø9 56 FF Ø6
[YF5]    37: Ø3 6Ø 28 62 Ø7 ØA Ø3 91 F7 AC AC 81 ØC 14 ØC ØC
[9P5]    38: ØC Ø6 Ø5 9F 2A 62 Ø6 CC Ø4 54 Ø3 D7 Ø4 D9 FF ØC
[3M5]    39: 32 DE Ø6 11 FF ØC Ø4 D8 FE Ø6 Ø6 Ø3 1E 2B AØ Ø9
[RE5]    4Ø: 4B Ø4 55 Ø5 58 Ø4 DC Ø3 61 29 65 Ø4 8D FE Ø3 Ø3
[5N5]    41: Ø4 D5 Ø5 18 FF ØE Ø4 DE 2D A2 Ø5 CD Ø5 95 Ø3 99
[XF5]    42: Ø5 6Ø 2C 62 Ø7 ØB FA AC 81 Ø6 31 31 ØE Ø6 59 32
[NS5]    43: 21 FD FF FF ØF Ø6 1E Ø5 9A FF Ø2 31 63 Ø3 92 FE
[1P5]    44: ØF ØF Ø7 CD FE Ø2 Ø2 4Ø AØ FF 56 32 E2 FC 1D 1D
[MK5]    45: 16 16 Ø7 4C FC 56 81 56 2B 34 22 Ø8 8B FE 56 56
[8R5]    46: 3F 6Ø 4Ø 5F FD ØF ØF Ø2 31 22 FC 7C 7C 6F 4A Ø7
[AP5]    47: 57 FE 4A 6F Ø3 5F Ø8 56 29 2B FE 6F 4A Ø9 56 Ø5
[NS5]    48: 9E 2F A4 FE 75 75 ØB 95 FF 75 3Ø A3 Ø4 D1 FF 5Ø
[3P5]    49: Ø7 E1 FE 4A 4A ØA 94 34 A9 Ø3 A1 31 62 Ø3 51 FF
[PG5]    5Ø: 5Ø Ø7 95 Ø3 5E FE 6F 6F 31 62 ØC 92 Ø3 9D 32 A1
[4M5]    51: FF 4A Ø8 D4 36 9B FD 4A 75 6F Ø9 6Ø Ø4 5E 3Ø 21
[NG5]    52: FF 6F Ø3 53 Ø8 62 Ø4 DC Ø8 15 29 2A Ø4 13 ØA 56
[1E1]    53: ØA D5 15 EB
```

Listing 20. skell2.pak ist eine gepackte Grafikdatei.

```
[ZR5]     1: 7F ØØ 7F 7F 59 FE FF 2B ØC 57 3D 27 32 5E Ø3 A4
[YR5]     2: FF 56 ØC D6 2F A2 Ø3 D4 FF 81 ØB 15 Ø9 DE 29 AD
[GN5]     3: ØC 15 Ø8 1E Ø3 25 28 2A Ø3 5F FF AC Ø8 95 31 5D
[2P5]     4: ØE 9Ø ØD 9B 24 A9 Ø9 CF FF D7 Ø6 17 Ø7 DD 29 A3
[5W5]     5: FE 2B 2D Ø4 11 FC AD AD AD 2D Ø7 17 ØC 1D FF 2B
[NN5]     6: 22 2B FA 57 57 2D AC AC 82 Ø3 54 Ø9 56 FC 82 82
[8K5]     7: 82 57 Ø3 A3 Ø7 66 22 6E FF 57 Ø3 AØ Ø4 94 FF Ø9
[GN5]     8: Ø9 99 FF Ø3 Ø6 A1 FF 57 23 A9 Ø6 CE Ø4 D3 FE Ø3
[8W5]     9: Ø3 Ø7 D9 FB Ø3 Ø3 AC 27 AD Ø6 AØ 28 EB Ø4 D4 FE
[1G5]    1Ø: Ø3 2E Ø3 6Ø Ø6 5B FD 81 Ø3 27 Ø4 53 FE AD 82 24
[4E5]    11: 2B Ø7 12 ØA 57 F4 81 81 81 Ø3 Ø3 Ø6 Ø6 Ø6 Ø3 AD
[UL5]    12: 82 57 2Ø 6C Ø4 4D Ø3 92 FF ØA Ø5 1B Ø7 9C FE 81
[RN5]    13: 81 Ø5 D4 Ø4 A6 Ø4 AB 1D AE Ø8 CD Ø5 Ø7 Ø4 D7 Ø6
[DR5]    14: DF Ø5 15 FE Ø6 Ø6 Ø3 69 23 ED FB Ø9 ØA D7 FF Ø2
[EE5]    15: Ø4 54 Ø4 18 Ø4 1E FE ØØ ØØ Ø4 56 Ø3 26 FE Ø3 AD
[GG5]    16: 22 AC FC 82 82 27 D7 Ø4 62 Ø6 63 Ø4 9B Ø6 92 Ø5
[CS5]    17: 66 Ø3 2B 1D 6F Ø5 CB FD 27 D7 FF Ø3 AØ FF 6B Ø4
[3U5]    18: A4 FD ØØ AC AC Ø4 C8 Ø3 D6 Ø6 A5 FF 57 1F 6D Ø5
[XH5]    19: 8A FF 27 Ø3 9Ø Ø5 AØ Ø3 E6 Ø5 DC Ø7 15 FD 81 81
[NT5]    2Ø: 56 2Ø AC Ø7 CA FF 27 Ø6 47 FC AC AC AC FF Ø3 59
[VR5]    21: Ø6 53 FF D7 Ø4 29 21 EC Ø5 ØB Ø7 9Ø FD AD 27 AC
[LM5]    22: Ø3 12 FF D7 Ø3 1B Ø3 A1 Ø5 28 28 6A FF AD Ø5 D2
[ES5]    23: FF 82 Ø4 11 Ø3 58 FF D7 Ø4 EØ FE AC AC Ø4 A7 21
[XN5]    24: AD Ø5 CB Ø7 D2 FF 57 Ø6 52 Ø6 DF Ø4 E4 24 EB Ø8
[2F5]    25: CC Ø3 95 Ø8 18 FE 56 56 Ø5 AØ 28 27 Ø5 4E Ø3 56
[ZR5]    26: FD 57 27 AC Ø5 8B FE D7 56 Ø3 AØ 2B 65 Ø3 8E Ø3
[BK5]    27: 95 Ø3 91 FD 27 81 81 Ø4 22 FE 81 81 Ø4 AØ FF 56
[8M5]    28: 2A A7 Ø4 DØ Ø4 92 Ø3 D8 Ø5 1A Ø4 A4 FE 56 D7 2B
[3X5]    29: E6 Ø5 8F FD AD AD Ø7 18 FD AC D7 AC 2E 24 Ø4
[RK5]    3Ø: 51 Ø3 57 FE 27 27 Ø3 2Ø Ø3 9C FF D7 Ø3 63 2C E5
[2V5]    31: Ø6 ØE FE ØA ØA Ø4 62 Ø4 D8 FD D7 2B 2B 2F 63 FB
[4K5]    32: ØØ Ø3 Ø9 2B 2B Ø5 9F Ø4 1A FE 2B 2B Ø4 1C 2F A6
[6M5]    33: FE Ø3 2B Ø9 56 Ø3 22 FD Ø9 81 Ø9 2E 26 FC 57 ØA
[DE5]    34: Ø9 57 Ø4 6Ø FE Ø9 Ø9 Ø4 22 FC Ø9 Ø9 ØA Ø3 2D A6
[CP5]    35: F8 ØØ 57 ØA 82 82 81 Ø9 81 Ø3 DA FF Ø9 Ø3 63 FD
[KR5]    36: 81 AC ØØ 2F A5 Ø4 91 Ø4 DA Ø5 1B FE AC AC 32 E3
[ZU5]    37: FA ØA ØØ 56 AC AC D7 Ø5 5A FD AC AC 56 3Ø E6 Ø4
[8U5]    38: 92 FF 56 Ø3 57 Ø3 9A FF Ø9 Ø6 6Ø FF ØA 2F 68 FF
[KS5]    39: Ø4 Ø5 94 Ø6 DA Ø4 A2 FF 83 2F A5 Ø4 D5 ØB 18 FD
[KL5]    4Ø: Ø3 ØA Ø3 2E 27 FF Ø5 ØF 46 3Ø 26 FF 83 1Ø 55 2E
[3G5]    41: A6 31 24 11 C4 3E 56 37 53 FF 59 Ø3 8B FF 83 Ø4
[VK5]    42: 8F FF 59 13 Ø5 Ø4 8B Ø6 8D 11 94 Ø7 C3 FE 28 ØB
[MP5]    43: Ø9 CB FF 28 Ø8 85 Ø9 AF ØA CB Ø3 FØ 17 FF Ø4 ØA
[PN5]    44: FD 59 Ø4 Ø4 Ø4 ØB Ø6 D7 FF ØA Ø4 1C FE Ø9 ØA ØA
[1G5]    45: Ø6 FE ØA ØA Ø5 27 1C 7Ø FF Ø4 Ø9 33 FE 59 Ø5 Ø5
[YE5]    46: 51 FE Ø9 ØA ØA 51 ØA 62 65 37 FF Ø4 35 9D Ø8 11
[TL5]    47: FE 59 ØA Ø5 1Ø Ø3 E1 ØA 12 1E EE FD Ø3 Ø3 Ø4 Ø4
[UG5]    48: ØF FA Ø2 Ø4 Ø4 Ø5 Ø5 ØA Ø3 2B FB 59 59 59 83 83
[LG5]    49: Ø4 21 FF ØA Ø3 56 Ø3 5Ø Ø3 6C FD Ø3 2E 2E 18 72
[2D5]    5Ø: FA Ø2 Ø3 Ø4 Ø4 Ø4 2D Ø3 69 FE Ø4 Ø4 Ø3 65 FD Ø4
[8K5]    51: Ø4 Ø9 Ø3 95 Ø4 9D FD ØA Ø9 Ø9 Ø4 9F FE Ø4 Ø4 Ø4
[CF5]    52: 8B FB Ø3 Ø5 Ø3 2E Ø2 18 75 FE ØA ØA Ø3 A8 Ø3 DØ
[YN5]    53: Ø5 9D FD Ø4 ØA Ø9 Ø5 CF FF Ø4 Ø4 A4 Ø4 96 Ø6 E9
[ZL5]    54: FD 2E 2E Ø3 16 B3 FD ØØ Ø3 2E Ø3 D3 Ø3 5E FF 83
[4D5]    55: Ø7 D2 ØD 19 FE Ø4 Ø4 Ø4 1Ø Ø3 25 19 F1 Ø3 C8 Ø4
[VE5]    56: AB Ø3 4F Ø3 29 13 55 Ø3 AD Ø4 6B 18 72 Ø9 87 Ø8
[3A2]    57: E8 11 98 21 7Ø 35 CA
```

Listing 21. skull.pak ist eine gepackte Grafikdatei.

```
[B46]     1: FF FF 7F ØØ 7F 7F 7F FE 7F 7E 7F FD 7F 7C 7F FB
[5X5]     2: 7F 7A 7F F9 7F 78 7F F7 7F 76 7F F5 16 74 FD A4
[F16]     3: CE CE 3E 4D 43 8A FF D4 7F 4D 7F CC 7F 4B 43 CA
[1W5]     4: FF 8Ø 7F ØE 7F 8D 7F ØC 7F 8B 4D ØA 3D D8 7F 94
[6S5]     5: 7F 13 7F 92 7F 11 12 9Ø FE 6B 41 3C A4 FC B7 B7
[KP5]     6: 1D 16 3B E4 FD B7 74 74 3A 21 FB A4 B7 99 74 5Ø
[ET5]     7: 3A E4 FC FF B7 63 13 3B E3 F9 B7 B7 74 5Ø 5Ø 93
[5V5]     8: 9F 38 64 FE 6B 41 Ø3 61 FC 5Ø. 5Ø 9F 93 37 E4 FE
[SL5]     9: 41 16 Ø4 17 Ø3 EØ 3A 24 Ø5 1D FE 6B 1D 3E 24 FE
[NU2]    1Ø: 1D 1D Ø5 1C 3A A9 1C A5
```

Listing 22. sword3.pak ist eine gepackte Grafikdatei.

```
[B46]     1: FF FF 7F ØØ 7F 7F 7F FE 7F 7E 7F FD 7F 7C 7F FB
[KX5]     2: 7F 7A 7F F9 7F 78 7F F7 7F 76 7F F5 7F 74 7F F3
[R36]     3: 7F 72 7F F1 7F 7Ø 7F EF 62 6E FE CC CC 3E D2 FD
[916]     4: C6 CD CD 3E D3 FD C6 CE CE 3E 13 FF C7 7F 52 42
[WS5]     5: D1 FD C7 C8 CE 3E 15 ØE 54 FF 23 3E 63 FD 16 16
[9X5]     6: ØF 34 A4 FF CF Ø7 D9 FE B7 6E 33 E4 Ø7 15 Ø3 1B
[A26]     7: FD B7 6E 6E 34 24 FE FF C7 Ø3 58 FB AD 82 AD AD
[4X5]     8: AD 36 62 Ø4 56 FA AD AD 82 82 82 AD 37 A2 FE FF
[MW5]     9: C8 Ø3 DE Ø4 9C 38 E1 FF FF Ø3 1C FC 99 99 6E 82
[BU5]    1Ø: 38 22 F8 9F AD 82 6E 6E 99 99 75 38 62 FE 6E 6E
[YH5]    11: Ø4 9A FE 23 23 36 2Ø FD 23 6E 6E Ø3 14 FB 82 23
```

Listing 23. sword6.pak ist eine gepackte Grafikdatei.

```
[ØG5]    12: 16 16 ØF 34 E1 FD 23 23 16 Ø4 52 FA 82 82 ØF ØF
[5A5]    13: 82 82 34 22 Ø5 58 Ø3 5C Ø3 61 36 61 Ø6 D7 Ø3 AØ
[NL5]    14: FF AD 3C A1 Ø5 9D 3B E2 Ø4 1E Ø4 AØ 3E 25 42 64
[VS1]    15: FE AD 82 18 E8
```

Listing 23. sword6.pak ist eine gepackte Grafikdatei.

```
[B46]     1: FF FF 7F ØØ 7F 7F 7F FE 7F 7E 7F FD 7F 7C 7F FB
[KX5]     2: 7F 7A 7F F9 7F 78 7F F7 7F 76 7F F5 7F 74 7F F3
[926]     3: 7F 72 7F F1 7F 7Ø 7F EF 7F 6E 7F ED 7F 6C 7F EB
[WU5]     4: 7F 6A 7F E9 7F 68 35 E7 FA 6F 9A AØ 9A 6F 4A Ø3
[7P5]     5: 21 39 25 Ø4 21 3A 24 F9 FF FF AØ 9A 75 6F 75 3A
[FN5]     6: 25 FF 6F Ø5 9E 38 61 FF 4A Ø4 1C Ø3 EØ FF 6F 36
[7S5]     7: E3 Ø6 1B FE 75 6F Ø4 E1 34 25 FB 4A 6F 94 AØ 94
[HN5]     8: Ø3 2Ø Ø3 A1 FE 4A 5Ø 38 67 Ø5 AØ 7A A4 FF 9A 3B
[ZX5]     9: 9F FC FF 4A 75 AØ Ø3 5D Ø4 62 FF 4A 34 67 FE FF
[ZA2]    1Ø: 4A Ø4 5A Ø5 A1 1B A5
```

Listing 24. trank.pak ist eine gepackte Grafikdatei.

```
[VL5]     1: FF 2B ØF ØØ Ø3 1Ø F9 56 2B 56 56 2B 2B AC ØD 19
[B95]     2: Ø8 ØA Ø4 ØF 1Ø ØØ Ø6 42 Ø3 16 Ø5 4C Ø3 13 Ø4 23
[BC5]     3: FE 56 81 Ø5 59 Ø6 58 Ø5 53 Ø3 26 Ø5 2D Ø3 14 Ø4
[3L5]     4: 74 FF 81 Ø6 73 Ø4 13 FE 56 81 Ø4 48 Ø4 49 FD 2B
[AH5]     5: 56 ØØ Ø3 68 ØA 5B Ø9 5E Ø5 A1 FE 56 AC Ø4 3Ø Ø3
[SG5]     6: 83 Ø8 79 Ø3 B5 Ø3 BD ØD 3D Ø7 98 15 97 Ø3 91 Ø5
[ØF5]     7: 31 Ø7 B6 Ø8 B6 Ø6 B4 Ø3 Ø9 ØB 54 Ø8 B5 Ø7 1E Ø4
[8F5]     8: E6 Ø6 24 FE ØØ 81 Ø8 F2 Ø7 17 Ø3 2D Ø4 33 Ø4 48
[ØS5]     9: Ø6 26 Ø5 13 Ø3 AE Ø9 59 FE FF FF Ø7 A7 Ø5 E6 FF
[2B5]    1Ø: 81 Ø4 Ø8 Ø6 5Ø Ø4 38 Ø9 15 Ø4 ØA Ø3 8Ø Ø4 ØE FF
[FT5]    11: FF 12 94 Ø5 E9 Ø7 EØ FF AC Ø3 88 ØB 34 Ø4 DD Ø4
[CL5]    12: 87 Ø9 1Ø Ø3 62 12 D5 FE 56 81 Ø9 AB Ø5 7Ø Ø5 BB
[WB5]    13: Ø8 3A Ø5 22 Ø5 ØF Ø3 Ø6 15 D3 Ø6 A3 Ø8 A7 Ø9 B4
[ZC5]    14: Ø9 7C Ø3 4B Ø3 Ø9 Ø3 3F Ø5 6Ø 19 13 Ø9 A9 ØA 36
[7E5]    15: FF 2B Ø6 8Ø Ø3 7Ø Ø4 4A FB ØØ ØØ Ø7 Ø7 Ø7 1A 53
[TG5]    16: FF 56 Ø5 7Ø Ø4 C6 Ø3 85 Ø5 B7 Ø5 B4 FF 2B Ø4 87
[RS5]    17: Ø4 89 Ø3 8F FD Ø8 2B Ø8 1A 94 FF FF Ø5 31 Ø3 AE
[GG5]    18: Ø5 C2 Ø6 C1 Ø5 7C Ø6 AF Ø3 5Ø Ø3 DØ FF Ø1 1B 93
[NM5]    19: Ø3 6F Ø5 6E Ø4 BD Ø8 FA Ø4 C4 Ø6 EF FE ØØ ØØ Ø3
[NK5]    2Ø: 11 FD 2B Ø1 Ø1 1B D4 Ø3 AD Ø5 47 Ø8 3E FE 2B 2B
[MM5]    21: Ø3 ØD Ø6 Ø1 FF 56 Ø3 9Ø FF Ø9 Ø5 5Ø 1B 14 FE FF
[CM5]    22: 81 Ø6 Ø7 Ø5 81 Ø6 7F Ø3 BA FF AC Ø5 C6 25 8C FF
[BN5]    23: FF Ø7 B2 Ø9 BC Ø5 C2 Ø6 Ø8 FC Ø7 Ø8 Ø8 2B Ø3 ØF
[WH5]    24: FE Ø1 Ø7 1C 54 ØA B2 Ø5 Ø1 FE 2B 2B Ø3 F7 ØC Ø6
[EP5]    25: FF Ø1 Ø3 D3 FF 2B 25 D7 Ø5 82 Ø5 F8 ØB 46 FF Ø1
[MG5]    26: Ø4 12 1C 56 FF 56 Ø5 B2 Ø5 41 Ø4 7C FF 81 ØA 82
[5K5]    27: FD Ø7 Ø7 Ø8 Ø3 D1 2Ø 92 FF AC Ø5 F2 Ø3 B8 Ø3 8A
[9K5]    28: FF 2B Ø3 3F Ø4 76 Ø6 B2 FE ØØ Ø8 Ø8 ØE 24 56 Ø6
[FF5]    29: 77 FF 81 Ø3 Ø1 FE 81 2B Ø6 F2 FF ØØ Ø5 1Ø Ø5 81
[BH5]    3Ø: 24 57 Ø4 B6 Ø3 CØ FF 81 Ø3 7B Ø7 86 Ø3 7D FE Ø8
[FN5]    31: 2B Ø4 DØ 25 16 Ø5 3A FF 56 Ø4 B3 ØB 85 Ø3 CF Ø4
[HF5]    32: 51 25 97 Ø5 7C Ø4 CØ Ø8 C5 FE Ø7 Ø9 Ø3 9Ø 2Ø D2
[AN5]    33: Ø3 FA Ø5 FA FF 56 Ø6 B9 Ø5 34 ØD Ø6 FE Ø8 Ø7 1E
[7E5]    34: 95 Ø5 72 Ø7 3F 11 3F Ø4 13 28 D4 Ø6 91 Ø3 39 ØC
[AH5]    35: 85 FD Ø1 2B Ø8 Ø3 D2 24 D7 Ø8 BC Ø5 B7 Ø4 Ø7 Ø5
[MD5]    36: ØC FA Ø8 2B Ø1 Ø7 Ø1 Ø7 23 D7 ØC 39 Ø5 73 3Ø ØB
[1D5]    37: Ø7 37 FD ØØ ØØ ØØ ØB 85 2E 1Ø FF ØØ Ø4 81 Ø5 83
[2X5]    38: FE AC AC Ø5 CA Ø4 D1 1F 93 Ø6 Ø6 ØA 7C Ø6 C2 FE
[UK5]    39: 56 56 29 8A Ø5 32 Ø5 FC Ø3 3C FF 56 Ø5 7A Ø8 86
[8K5]    4Ø: FF Ø1 2B 4F Ø4 Ø8 Ø8 F6 34 86 Ø6 F2 Ø6 7E 37 86
[KK5]    41: Ø4 7C Ø5 4Ø Ø6 BC ØA ØC FF 2B 27 97 Ø3 FD Ø3 3E
[LL5]    42: Ø3 44 Ø5 46 Ø5 ØC FF 2B 2C 12 Ø6 7D FF 81 ØC 85
[RF5]    43: 21 51 Ø8 BC Ø5 87 Ø3 4Ø FF 81 3Ø C3 Ø5 F2 Ø3 B8
[TN5]    44: FF 56 Ø4 C7 15 CØ FF Ø8 1D D6 Ø8 33 Ø6 BA Ø4 BE
[WE5]    45: 36 Ø5 Ø3 3A Ø5 81 Ø3 B6 Ø8 86 FF Ø1 Ø3 51 FF Ø1
[DF5]    46: 28 53 Ø3 82 Ø3 BD Ø3 F9 Ø4 78 Ø4 C7 27 CC Ø7 B3
[NE5]    47: Ø6 B9 Ø3 F8 Ø4 Ø3 Ø5 Ø6 2E ØC Ø4 39 Ø6 37 FF ØØ
[PE5]    48: ØF 45 Ø3 12 1B 57 Ø6 C6 Ø6 38 Ø8 7D ØB 46 Ø3 53
[4F5]    49: Ø3 5Ø 1C 97 Ø5 B2 Ø4 41 FE ØØ ØØ Ø6 4Ø ØE 84 FE
[YM5]    5Ø: Ø1 Ø8 Ø3 C3 22 D7 Ø3 BB Ø6 FB Ø3 F7 ØE 45 FF 2B
[HM5]    51: 1F 94 Ø7 33 Ø6 FC FE 2B 2B Ø4 77 ØB 46 FF 2B Ø5
[4C5]    52: 9Ø 23 17 Ø4 42 Ø3 3E Ø4 7A FE 2B 56 Ø5 86 Ø4 C2
[VS5]    53: FF Ø1 Ø4 91 2B 55 FE 56 81 Ø4 B9 FF AC Ø4 C6 Ø4
[VH5]    54: 8B FD Ø8 Ø1 Ø8 Ø5 9Ø 23 97 FE 2B 81 Ø4 BB FE 56
[EL5]    55: 56 Ø4 FA Ø7 Ø6 FE 2B Ø8 Ø4 ØF FF Ø8 1E D4 Ø6 86
[4F5]    56: Ø9 B8 Ø5 81 Ø7 46 FF Ø8 Ø3 4E FE Ø1 Ø1 1F 13 Ø5
[WS5]    57: C6 FF AC Ø4 83 Ø3 3D Ø3 BF Ø4 FB ØB 46 Ø3 8F FF
[6F5]    58: Ø7 22 95 Ø3 82 Ø3 35 Ø5 7D Ø4 41 34 86 Ø8 7A FF
[HH5]    59: 56 ØA Ø3 Ø5 4D 2Ø 12 ØB F2 Ø4 3D FE 81 56 Ø3 3B
[SK5]    6Ø: Ø3 BA Ø4 BA FE Ø7 Ø8 Ø5 4F FE Ø7 Ø1 1B 17 ØB B1
[UM5]    61: Ø4 3C Ø5 39 FE 2B AC Ø5 86 FD ØØ Ø7 Ø1 Ø4 8F FF
[6R5]    62: Ø7 1C 56 FE FF FF Ø6 46 Ø3 83 Ø4 BA FF ØØ Ø4 79
[8S5]    63: FF 56 Ø8 C5 Ø4 4D FF Ø7 1C 16 Ø5 EE Ø5 F2 Ø3 BD
[CL5]    64: Ø3 FB Ø3 FC Ø5 ØØ Ø9 C6 1D 15 ØB 2C Ø7 B7 Ø4 7E
[VC5]    65: ØD Ø2 FF Ø8 21 11 1Ø 31 FF ØØ Ø4 81 Ø9 46 22 5Ø
[6L5]    66: ØA 71 Ø5 81 Ø6 BF FF 56 Ø5 C6 FF ØØ 11 A1 ØB BB
[683]    67: Ø8 E8 FE FF 81 Ø6 C7 Ø7 F8
```

Listing 25. tuerof.pak ist eine gepackte Grafikdatei.

```
[VL5]     1: FF 2B ØF ØØ Ø3 1Ø F9 56 2B 56 56 2B 2B AC ØD 19
[B95]     2: Ø8 ØA Ø4 ØF 1Ø ØØ Ø6 42 Ø3 16 Ø5 4C Ø3 13 Ø4 23
[BC5]     3: FE 56 81 Ø5 59 Ø6 58 Ø5 53 Ø3 26 Ø5 2D Ø3 14 Ø4
[3L5]     4: 74 FF 81 Ø6 73 Ø4 13 FE 56 81 Ø4 48 Ø4 49 FD 2B
[AH5]     5: 56 ØØ Ø3 68 ØA 5B Ø9 5E Ø5 A1 FE 56 AC Ø4 3Ø Ø3
[SG5]     6: 83 Ø8 79 Ø3 B5 Ø3 BD ØD 3D Ø7 98 15 97 Ø3 91 Ø5
[ØF5]     7: 31 Ø7 B6 Ø8 B6 Ø6 B4 Ø3 Ø9 ØB 54 Ø8 B5 Ø7 1E Ø4
[8F5]     8: E6 Ø6 24 FE ØØ 81 Ø8 F2 Ø7 17 Ø3 2D Ø4 33 Ø4 48
[AH5]     9: Ø6 26 Ø5 13 Ø3 AE ØA 59 FF Ø8 Ø7 A7 Ø5 E6 FF 81
[CA5]    1Ø: Ø4 Ø8 Ø6 5Ø Ø4 38 Ø9 15 Ø4 ØA Ø3 8Ø Ø4 ØE F9 Ø7
[9C5]    11: ØØ ØØ Ø7 Ø7 Ø8 Ø7 Ø5 9A Ø4 6Ø Ø3 9D Ø5 E9 Ø7 EØ
[XK5]    12: FF AC Ø3 88 ØB 34 Ø4 DD Ø4 87 Ø9 1Ø Ø5 A2 F6 Ø7
[YK5]    13: 56 2B Ø9 Ø8 Ø9 Ø7 Ø9 Ø9 Ø8 Ø6 D9 FE 56 81 Ø9 AB
[HE5]    14: Ø5 7Ø Ø5 BB Ø8 3A Ø5 22 Ø5 ØF Ø3 Ø6 Ø5 95 FD Ø9
[4P5]    15: Ø8 2B Ø3 16 Ø5 DD FF 2B Ø6 DA FE Ø8 Ø9 Ø9 EC Ø6
[ØE5]    16: F4 Ø8 FE Ø5 ØC Ø3 4B Ø3 Ø9 Ø3 3F Ø5 A1 Ø3 28 Ø6
[FL5]    17: 56 Ø4 1E FF Ø8 Ø3 5F Ø5 26 Ø3 29 Ø9 A9 ØA 36 FF
[SE5]    18: 2B Ø6 8Ø Ø3 7Ø Ø4 4A Ø6 D3 18 55 FE Ø9 56 Ø5 7Ø
[TG5]    19: Ø4 C6 Ø3 85 Ø5 B7 Ø5 B4 FF 2B Ø4 87 Ø4 89 Ø3 8F
[3H5]    2Ø: Ø3 A1 Ø7 A7 13 9B FF Ø9 Ø5 31 Ø3 AE Ø5 C2 Ø6 C1
[1H5]    21: Ø5 7C Ø6 AF Ø4 5Ø ØB E4 Ø6 12 Ø7 99 FC Ø8 Ø9 56
[ØZ5]    22: 2B Ø3 ED FF AC Ø6 6D Ø4 BD Ø8 FA Ø4 C4 Ø6 EF FB
[3A5]    23: ØØ ØØ Ø8 56 2B Ø7 1C Ø3 EØ Ø8 5B Ø8 64 Ø3 2C Ø4
[1K5]    24: AC Ø5 47 Ø8 3E FE 2B 2B Ø3 ØD Ø6 Ø1 FF 56 Ø3 9Ø
[FH5]    25: FE Ø9 2B Ø6 51 Ø5 62 Ø9 15 Ø9 A5 Ø3 AB FF 81 Ø6
[8P5]    26: Ø7 Ø5 81 Ø6 7F Ø3 BA FF AC Ø5 C6 Ø4 8C Ø7 1A Ø5
[ZK5]    27: 97 Ø7 DA ØD A3 FE Ø8 Ø9 Ø7 B2 Ø9 BC Ø5 C2 Ø6 Ø8
[SK5]    28: FE Ø7 Ø7 Ø8 E2 Ø4 ØF ØA 5D Ø4 25 Ø4 E1 ØC ED Ø8
[WL5]    29: FE FE 2B 2B Ø3 F7 ØB Ø6 FE Ø8 Ø8 Ø4 85 Ø7 DB Ø4
[1L5]    3Ø: E7 FF Ø9 Ø4 F7 Ø6 14 ØF ED Ø5 82 Ø5 F8 ØA 46 Ø6
[AM5]    31: DØ FE 2B 2B ØA 58 FE 56 2B Ø8 A4 Ø6 98 Ø3 54 Ø3
[BK5]    32: B4 Ø5 41 Ø4 7C FF 81 ØA 82 Ø3 CD Ø5 A9 FF Ø7 Ø3
[9G5]    33: C3 Ø6 1F Ø4 CD Ø4 D5 Ø4 26 Ø6 6C Ø3 FØ Ø5 F2 Ø3
[SR5]    34: B8 Ø3 8A FF 2B Ø3 3F Ø4 76 Ø6 B2 FF ØØ Ø4 EE Ø6
[9F5]    35: DF Ø4 28 Ø5 8Ø Ø3 ØF Ø4 14 Ø3 B6 Ø5 E8 Ø3 D9 Ø8
[PL5]    36: 72 Ø6 77 FF 81 Ø3 Ø1 FE 81 2B Ø6 F2 Ø3 7E Ø8 B2
[Z85]    37: Ø6 56 Ø9 54 Ø5 65 Ø4 2C Ø3 2B Ø9 72 Ø5 53 Ø4 4Ø
[6F5]    38: FF 56 Ø8 85 ØD 5E Ø3 7A Ø4 82 ØC 6Ø Ø5 D9 Ø9 32
[LL5]    39: Ø5 9F FF 56 Ø4 B3 16 85 1Ø DA Ø8 EB Ø9 B3 Ø5 7C
[EG5]    4Ø: Ø4 CØ Ø8 C5 FF Ø7 Ø4 ØE Ø6 21 Ø3 EE Ø7 66 Ø5 22
[ZH5]    41: Ø6 68 Ø5 6C Ø8 4F FF 56 Ø6 B9 Ø5 34 ØB Ø6 Ø3 2F
[3H5]    42: FF Ø7 Ø3 55 Ø6 EC Ø4 4D Ø3 3D Ø4 5C Ø6 EB Ø3 51
[TH5]    43: FF AC Ø5 72 Ø7 3F 16 3F Ø3 26 ØA 58 Ø5 15 Ø3 9E
[6S5]    44: Ø5 6C Ø4 6F ØD B3 Ø4 7E FF 2B ØE 85 ØB E7 Ø5 CD
[TK5]    45: Ø4 E3 14 A7 Ø8 BC Ø5 B7 Ø4 Ø7 ØA 8C Ø6 D8 Ø7 DC
[5D5]    46: Ø4 68 13 E7 ØC 39 Ø5 73 18 ØB Ø7 5A Ø7 2Ø ØA 31
[KD5]    47: Ø7 37 FD ØØ ØØ ØØ ØD 85 ØC 25 Ø7 54 16 65 Ø4 7F
[7R5]    48: Ø4 81 Ø5 83 FE AC AC ØD 8A Ø5 AD Ø7 A5 Ø6 DA Ø8
[FH5]    49: 1F FF Ø9 Ø6 Ø6 ØA 7C Ø6 C2 FE 56 56 Ø7 CA ØD 64
[XT5]    5Ø: ØB 5E Ø9 E9 Ø5 46 FF AC Ø5 FC Ø3 3C FF 56 Ø5 7A
[2K5]    51: ØB 86 ØA 9B Ø9 51 Ø8 24 FE 56 81 ØC 6E Ø4 Ø8 Ø8
[YM5]    52: F6 25 46 Ø3 78 FF 2B Ø3 E2 ØA 72 ØA F2 ØA 86 FE
[ZL5]    53: 56 2B Ø7 9C Ø3 AD Ø6 D2 Ø4 AD FF Ø8 Ø3 E7 Ø3 7C
[CG5]    54: Ø3 C3 ØD BØ Ø4 7C Ø5 4Ø Ø6 BC Ø4 ØC FF 2B Ø8 D1
[WH5]    55: Ø3 EE Ø6 1C Ø4 EE Ø3 E6 FE 56 56 Ø4 21 Ø3 2C ØC
[PR5]    56: B2 Ø3 FD Ø5 EA FF AC Ø5 46 1D 4C Ø4 AE Ø3 6C ØE
[ZG5]    57: 3Ø Ø6 7D FF 81 24 85 Ø4 22 FF Ø9 Ø4 E1 Ø8 BC Ø5
[ZH5]    58: 87 Ø3 4Ø FF 81 ØD C3 19 9Ø Ø3 18 Ø4 B5 Ø3 FØ Ø5
[CT5]    59: F2 Ø3 B8 FF 56 Ø4 C7 2A CØ Ø3 6E FF 2B Ø4 28 Ø9
[WG5]    6Ø: 32 Ø6 BA Ø4 BE 26 Ø5 Ø6 69 ØB 71 Ø5 3D FF 81 Ø3
[PN5]    61: 79 1E 85 ØA 5A ØE 6D Ø4 EB Ø3 83 Ø6 76 Ø4 C7 2E
[8N5]    62: 4C Ø6 B9 Ø3 F8 Ø4 Ø3 Ø5 Ø6 2F CC Ø5 F8 Ø4 3F FF
[4G5]    63: ØØ 2D 85 Ø6 C6 Ø6 38 Ø8 7D ØD 86 Ø4 3B ØC 97 Ø4
[GG5]    64: 77 ØC A7 Ø5 B2 Ø4 41 FE ØØ ØØ Ø6 4Ø ØF 84 Ø3 D3
[UR5]    65: FE 2B 2B ØA D8 FE 56 2B Ø3 62 12 E7 Ø3 BB Ø6 FB
[GG5]    66: Ø3 F7 ØF C5 Ø4 A3 Ø6 2Ø FF Ø7 Ø6 9Ø FF Ø8 Ø3 1D
[PG5]    67: 11 29 Ø6 FC Ø3 94 11 C3 Ø3 43 Ø5 5F Ø5 14 FF Ø8
[UF5]    68: Ø5 21 Ø3 76 1Ø 2A Ø4 5C Ø3 3E Ø4 7A FE 2B 56 Ø5
[2K5]    69: 86 Ø3 7F Ø4 8E Ø4 82 Ø6 96 ØC 91 Ø7 D6 FF Ø8 ØF
[ØT5]    7Ø: 71 FE 56 81 Ø4 B9 FF AC Ø4 C6 Ø3 8B ØE CD FE 56
[YT5]    71: 81 ØE 9D Ø6 4E Ø8 B2 FE 2B 81 Ø4 BB Ø3 DE Ø3 FB
[XP5]    72: Ø7 Ø6 ØF CD ØF 1B FF Ø9 Ø6 2B Ø6 86 Ø9 B8 Ø5 DF
[LK5]    73: Ø7 46 Ø5 6D Ø5 52 Ø3 AF Ø5 52 Ø4 EB Ø4 5D Ø6 5D
[DM5]    74: Ø6 ED Ø5 72 Ø4 D9 Ø3 3D Ø3 BF Ø4 FB ØE 46 FF Ø7
[7G5]    75: Ø8 A8 FE Ø9 Ø7 Ø3 9Ø Ø3 41 Ø4 9C Ø6 98 Ø8 EF Ø3
[1H5]    76: 82 Ø3 35 Ø5 7D Ø4 41 ØF 86 Ø5 A2 Ø8 9A Ø5 68 ØA
[5L5]    77: A7 FF Ø3 Ø8 B2 Ø8 7A FF 56 ØA Ø3 Ø6 EB Ø6 27 Ø5
[RT5]    78: 57 Ø4 DE Ø5 5B Ø8 E7 FF Ø8 ØD FØ Ø4 3D FE 81 56
[PG5]    79: Ø3 3B Ø3 BA Ø4 BA FF Ø7 Ø3 19 ØB E4 Ø3 52 Ø9 56
[9G5]    8Ø: 12 28 Ø4 42 FE ØØ ØØ Ø5 39 FE 2B AC Ø5 86 Ø3 4C
[UG5]    81: FF Ø9 Ø5 9Ø Ø7 CD 16 5C Ø6 46 Ø4 23 Ø3 BB FF ØØ
[BN5]    82: Ø4 79 FF 56 Ø8 C5 FD Ø7 Ø8 56 Ø5 E3 Ø4 AB Ø9 CF
[VR5]    83: Ø6 D9 Ø5 D6 FF Ø8 Ø3 ED FF Ø3 Ø6 86 Ø3 BD Ø3 FB
[XA5]    84: Ø3 FC Ø5 ØØ Ø9 C6 Ø3 15 Ø7 A5 Ø6 ØF Ø9 16 Ø3 28
[AR5]    85: FF 2B Ø3 EC FF Ø9 Ø7 3Ø Ø7 B7 Ø4 7E ØD Ø2 Ø6 8F
[SE5]    86: 19 55 13 2E FF ØØ Ø4 81 27 46 Ø4 A8 ØA 71 Ø5 81
[FA5]    87: Ø6 BF FF 56 Ø5 C6 ØB BB 1A D6 FE Ø3 81 ØD C7
```

Listing 26. tuerzu.pak ist eine gepackte Grafikdatei.

```
[YG5]    1: FF FF ØE ØØ FA Ø2 Ø9 Ø2 Ø2 Ø2 Ø9 Ø6 14 FF FF Ø7
[1T5]    2: 15 FE FF Ø9 Ø5 ØF FE Ø2 Ø2 Ø9 Ø7 FF FF ØD 27 ØD
[BA5]    3: Ø3 Ø3 39 Ø4 2Ø Ø3 56 Ø4 19 Ø6 56 Ø5 5D Ø3 34 ØD
[9E5]    4: 2D 14 3B Ø4 22 FF Ø2 Ø3 5B Ø8 78 Ø6 1F Ø6 1D Ø9
[6C5]    5: 76 Ø5 A9 13 78 Ø8 11 Ø3 A1 Ø3 25 Ø5 59 Ø3 D6 Ø3
[WH5]    6: CC Ø4 DF Ø6 E6 Ø5 73 16 35 Ø5 F3 Ø5 D6 Ø7 1A Ø3
[WK5]    7: E3 Ø6 9A Ø8 CA Ø3 D4 Ø5 94 Ø5 B3 12 B7 Ø5 D4 Ø3
[5D5]    8: 4A Ø8 E5 Ø9 58 Ø4 9Ø Ø6 AØ Ø6 ØA 11 B5 Ø5 Ø8 Ø3
[HL5]    9: 6E Ø4 23 Ø8 AØ Ø7 CE ØB 97 Ø7 9D Ø4 D8 14 F8 Ø5
[MM5]   1Ø: E3 Ø5 ØE Ø7 27 Ø3 D8 FF ØØ Ø7 DC Ø6 5F Ø9 A7 1Ø
[KE5]   11: 37 Ø5 84 Ø4 BØ Ø7 8D Ø3 E1 FD 81 81 81 Ø4 EØ Ø4
[PE5]   12: 18 Ø5 18 Ø5 CE Ø5 Ø8 Ø3 2E ØF 73 Ø6 85 ØA 94 Ø8
[XR5]   13: DC ØF 59 Ø9 49 14 B4 Ø4 8C Ø3 87 FF Ø2 Ø3 51 FE
[195]   14: 81 FF Ø6 17 1Ø 98 Ø5 47 Ø6 26 12 73 Ø3 46 Ø3 Ø7
[HN5]   15: FF FF Ø9 51 15 94 FF ØØ 14 6E Ø4 FF Ø5 2D Ø3 C9
[ØD5]   16: Ø6 12 ØA 1C Ø4 23 ØF 5D Ø5 AE ØE 34 Ø3 Ø2 Ø5 BØ
[2F5]   17: Ø4 ØB Ø7 17 Ø3 4E Ø5 4E ØD 1F FE ØØ Ø2 Ø5 2B Ø4
[DG5]   18: 48 1Ø F6 Ø5 72 Ø3 4A Ø9 D6 FF 5E Ø3 95 FF 57 Ø3
[PL5]   19: 99 ØF 9D FF Ø2 Ø5 44 Ø4 B1 ØE 38 Ø4 BF FF Ø2 Ø6
[K26]   2Ø: CD Ø5 CA FF 5E Ø7 D5 FF 57 Ø3 DD ØC 9B Ø6 BØ ØF
[5H5]   21: 74 Ø9 42 Ø4 ØF FE Ø9 Ø3 ØA D3 Ø8 D7 ØC E3 Ø5 2E
[UM5]   22: Ø6 FE FF Ø2 ØB 7A FF Ø9 Ø3 3A Ø5 8B FC Ø3 Ø3 ØØ
[FE5]   23: Ø3 ØE 13 Ø3 1F ØD 23 Ø3 47 Ø6 3A FC Ø2 FF Ø2 Ø2
[4H5]   24: Ø3 7A Ø3 C8 Ø5 84 Ø6 8A FF Ø3 Ø3 8F 1A 53 Ø5 3Ø
[DH5]   25: FF Ø2 Ø9 32 Ø4 7D Ø3 BC Ø4 43 Ø4 C2 Ø3 8C Ø3 8E
[LH5]   26: FF Ø9 1Ø 93 ØB A1 Ø5 FØ ØB B4 Ø4 7F Ø4 F7 Ø6 73
[5L5]   27: Ø4 E9 FE Ø3 Ø9 11 D1 FE 57 57 1Ø E4 Ø3 F2 Ø7 74
[GG5]   28: Ø6 3D Ø5 36 FD Ø9 FF ØØ Ø4 8Ø Ø3 DØ FE ØØ ØØ 19
[2L5]   29: 15 Ø4 AE Ø4 F2 Ø8 FA Ø6 BD FF Ø2 Ø4 C6 FE Ø9 Ø2
[KH5]   3Ø: Ø4 4B FF Ø2 Ø4 9Ø FF D7 ØE 53 Ø9 63 Ø4 6D Ø9 34
[AV5]   31: Ø6 F8 Ø6 3C Ø5 AF FE Ø2 Ø9 Ø3 C9 FF FF Ø3 9Ø FC
[7F5]   32: Ø3 D7 49 49 Ø7 96 Ø4 A8 Ø5 A2 Ø5 A4 Ø4 86 Ø5 CB
[FM5]   33: Ø7 37 Ø6 B4 Ø4 46 Ø4 EE Ø4 EF Ø4 C3 FB Ø3 5E 5E
[3N5]   34: D7 D7 Ø7 DA FB 5E 49 49 D7 5E Ø4 E7 Ø5 A3 Ø6 Ø3
[7L5]   35: Ø4 75 Ø5 FB Ø5 BB Ø5 37 Ø6 Ø5 Ø5 CA FE Ø9 Ø9 ØB
[SG5]   36: 5A FF D7 Ø3 D4 Ø5 18 Ø3 5A Ø8 2A Ø5 C6 Ø5 8Ø Ø9
[7K5]   37: FC Ø7 ED Ø7 CØ Ø9 53 Ø8 9B Ø4 59 Ø4 67 FF Ø9 Ø6
[XL5]   38: Ø4 Ø6 89 ØC F8 FF Ø2 Ø3 AC Ø3 C4 18 8B Ø5 A4 Ø4
[8K5]   39: A8 Ø5 44 Ø4 BA Ø3 78 ØB 7A Ø4 C4 Ø4 71 Ø4 C5 Ø9
[CN5]   4Ø: CF ØF D9 FD 16 ØØ ØØ Ø5 EC Ø5 46 Ø3 C7 ØC 79 Ø4
[CC5]   41: 82 Ø4 45 Ø4 2A Ø8 ØF Ø3 24 ØB 1B FC ØØ 16 ØØ 16
[DF5]   42: Ø5 4C Ø4 32 Ø3 FØ Ø4 6B ØD 3A Ø5 6A Ø5 Ø4 Ø3 8E
[SG5]   43: ØA 53 Ø3 98 Ø7 E4 Ø4 68 Ø7 2F Ø4 ØE Ø4 BØ ØA FB
[8G5]   44: Ø3 71 Ø3 8B Ø9 44 Ø8 1A Ø9 EØ Ø3 A4 FF ØØ Ø6 A8
[WG5]   45: Ø5 C6 Ø5 B1 ØB BA Ø4 C2 Ø6 EB Ø4 ØB Ø7 5Ø ØC D8
[DG5]   46: FF Ø3 Ø8 E6 Ø6 AA Ø4 Ø2 ØB 76 Ø3 39 Ø5 8A Ø5 F4
[5D5]   47: Ø3 47 FF Ø3 Ø4 1B Ø3 55 Ø9 1A FC Ø3 Ø3 ØØ Ø3 Ø6
[GB5]   48: AB Ø6 ØD Ø5 42 Ø4 Ø1 Ø5 3E Ø4 8Ø Ø4 48 Ø5 3F Ø4
[ZD5]   49: Ø9 FF Ø3 Ø4 52 Ø5 9B Ø6 5D FE 2C Ø3 Ø3 A3 Ø4 69
[KD5]   5Ø: Ø4 A4 Ø4 6B Ø5 84 Ø8 39 FE Ø2 Ø2 Ø4 A3 FE Ø3 Ø3
[ML5]   51: Ø3 8D Ø4 AD Ø4 4F Ø3 93 Ø3 9F Ø8 5A FF 2C Ø4 A2
[CG5]   52: Ø4 6F Ø3 D1 Ø3 9Ø Ø3 F1 Ø6 7F Ø8 3A FC FF Ø3 Ø3
[YK5]   53: AC Ø5 Ø4 FC 81 Ø3 Ø3 Ø1 Ø4 ØE Ø7 92 Ø6 DA FE 2C
[PP5]   54: 2C Ø4 61 FF Ø3 Ø3 B3 Ø3 AF Ø5 C4 FF Ø3 Ø4 33 Ø8
[6M5]   55: 78 Ø8 FF FC 81 81 81 56 Ø5 F4 FF Ø1 Ø5 4F Ø6 57
[6K5]   56: Ø6 1D FD 57 57 2B Ø3 31 Ø9 27 FF Ø9 Ø3 64 ØE F5
[RH5]   57: Ø6 44 FF ØØ Ø3 88 FF Ø8 Ø3 8C Ø6 4F Ø4 56 Ø3 5C
[3H5]   58: Ø3 5E FC 81 56 56 2B Ø3 63 Ø7 E7 Ø4 6C Ø4 32 ØB
[5M5]   59: 75 ØE 82 Ø7 CE FF 2B Ø4 5B Ø4 5E Ø5 AØ FE 2B 2B
[WK5]   6Ø: Ø3 93 Ø5 6C Ø7 AD ØB F5 Ø4 81 FF 56 Ø5 C6 FF Ø1
[WS5]   61: ØC CB FF AC Ø4 1C Ø5 DD Ø5 EØ Ø5 91 Ø3 EB FE Ø9
[TF5]   62: Ø2 Ø3 AC Ø5 B3 FF FF Ø7 F8 Ø5 Ø1 Ø4 Ø5 Ø4 27 Ø9
[MS5]   63: ØD FE AC AC Ø4 17 Ø8 1D FE 56 56 Ø7 8F FE Ø8 Ø8
[MB5]   64: Ø5 EA Ø4 73 Ø8 F6 Ø5 4Ø FF Ø1 Ø6 Ø7 Ø7 68 Ø6 53
[EF5]   65: FE AC 2B Ø5 41 Ø5 6Ø Ø5 91 Ø6 8C Ø4 6F Ø3 7A Ø6
[TM5]   66: B4 Ø9 7D FF Ø1 Ø3 A9 Ø4 A7 Ø8 A8 Ø3 97 FE 56 56
[RF5]   67: Ø3 D8 Ø5 21 FF 81 Ø8 4E Ø7 67 FE Ø3 Ø3 Ø6 74 Ø9
[SP5]   68: BB Ø4 C8 Ø5 9Ø Ø8 4C FF 81 Ø4 D6 Ø4 D7 Ø4 DE Ø9
[8N5]   69: E3 Ø9 E8 Ø3 BØ Ø9 F6 FE AC 81 Ø5 Ø2 Ø4 88 Ø6 C7
[AN5]   7Ø: Ø4 12 FF 81 Ø3 15 FF 2B Ø7 1B Ø6 A4 ØF 25 FF Ø8
[KH5]   71: Ø3 B9 Ø7 7A Ø9 41 Ø8 8E Ø7 52 Ø4 DA ØC EØ ØF 66
[ØN5]   72: FF Ø2 Ø3 B8 ØC 7C ØA ØB Ø5 92 Ø4 D6 Ø4 1F FE 81
[YY5]   73: 81 ØB 89 ØB EB Ø3 D9 FF 56 Ø4 B9 FF Ø9 Ø6 CØ FE
[CE5]   74: Ø1 ØØ ØC E2 Ø4 D8 Ø3 18 Ø4 DD Ø5 AØ Ø6 71 ØC EC
[6T5]   75: FE AC AC Ø3 F8 FE 56 2B ØD FD ØC ED Ø4 14 Ø5 54
[2N5]   76: ØC AB ØA 5F Ø3 36 Ø4 F7 FD 2B 2B Ø2 Ø7 3F FF ØØ
[UB5]   77: Ø4 Ø7 Ø8 25 Ø7 D5 ØE 5A Ø6 22 Ø7 A5 Ø4 76 Ø5 79
[GM5]   78: FF ØØ Ø4 7F ØB Ø4 Ø5 AA Ø6 94 Ø8 99 ØC 48 Ø7 C8
[CR5]   79: Ø5 75 Ø5 B9 FF 2B Ø5 BF FF 2B Ø5 C5 Ø8 AC ØF 53
[FL5]   8Ø: ØA Ø9 Ø9 E6 Ø6 F5 FF 2B Ø5 BA Ø3 4Ø FF 2B Ø6 Ø4
[KK5]   81: Ø8 2B Ø5 12 Ø6 D8 Ø6 EA ØC 68 ØC FØ Ø7 FB FF Ø1
[AB5]   82: Ø3 Ø3 Ø9 Ø8 FD Ø2 Ø8 Ø1 ØA 13 Ø4 88 Ø5 88 ØA 85
[EM5]   83: Ø3 AF FF 81 Ø9 73 Ø6 3C Ø7 82 Ø6 AC FE Ø2 Ø2 ØA
[1L5]   84: 91 Ø6 A4 Ø4 9D Ø6 2E Ø5 6Ø Ø4 71 Ø6 B4 FF FF ØE
[FP5]   85: 7B 11 C9 Ø6 9E ØC 65 Ø6 EA Ø3 F2 Ø4 F4 1D B9 Ø4
[ZF5]   86: 57 13 1A Ø4 6B Ø3 B2 Ø8 34 13 7C FE Ø8 Ø2 ØA 52
[NF5]   87: 1Ø 5B Ø6 DC 16 71 Ø5 9B Ø5 CB Ø9 13 Ø6 C7 ØC AØ
[4P5]   88: Ø4 D9 ØB B1 Ø6 3B FF AC Ø4 AD ØB 44 ØB 53 Ø5 EB
[UN5]   89: ØB A2 Ø3 28 Ø5 FØ Ø4 F4 Ø3 38 FF 2B Ø9 FC FF 2B
[HK5]   9Ø: ØA Ø6 Ø3 51 FF Ø8 Ø9 57 Ø6 6A ØA 24 FE Ø8 FF Ø8
[YH5]   91: FØ Ø4 36 1Ø 3B Ø6 61 FC 81 81 Ø1 56 ØC 1C Ø6 AB
[UR5]   92: Ø6 DE FE FF 81 ØA 2F Ø4 38 Ø9 FD Ø9 87 Ø3 EE Ø3
[1W3]   93: D2 Ø7 A6 ØC DB ØB 69 Ø9 F3 Ø4 BC
```

Listing 27. advent.pak ist eine gepackte Grafikdatei.

```
[KD5]    1: FF 6C Ø3 ØØ FD 48 6C 48 Ø3 Ø4 Ø6 Ø3 Ø3 ØC Ø5 12
[ØN5]    2: Ø4 ØØ FE 6C 6D Ø3 1C Ø4 ØB Ø6 17 FB 9Ø 9Ø 9Ø 6C
[Z75]    3: 9Ø Ø5 1Ø Ø6 22 Ø5 13 Ø3 1F Ø4 3D Ø6 13 Ø4 25 Ø7
[MP5]    4: 11 Ø6 3B FE 6D 4E Ø8 4D FD 6C 9Ø 6D Ø6 29 Ø6 5Ø
[KL5]    5: Ø9 5Ø Ø5 1D Ø8 84 FE 6C 48 Ø3 2F FF 9Ø Ø3 91 Ø3
[4M5]    6: 93 ØA 77 Ø5 8B Ø9 A7 Ø9 9C Ø8 57 FD 48 9Ø 6D Ø3
[SL5]    7: B6 Ø4 C6 Ø3 67 Ø3 6F Ø4 BØ Ø4 92 FE 6C 6D Ø3 CØ
[8N5]    8: Ø8 B2 FE 9Ø 6D ØA A7 Ø4 AF Ø7 1F Ø3 D9 FF 9Ø Ø3
[DE5]    9: DB Ø3 Ø1 Ø4 E2 Ø5 E8 Ø4 D3 Ø7 ØF Ø8 B4 Ø5 ØC Ø4
[VC5]   1Ø: 6C ØD 2C Ø4 93 Ø5 12 FF 9Ø Ø4 Ø5 Ø3 Ø3 Ø5 38 Ø3
[LG5]   11: 6A Ø7 61 Ø8 12 Ø7 22 Ø7 68 Ø3 53 ØD EA Ø4 E7 Ø8
[6B5]   12: 64 Ø3 8E Ø6 54 Ø8 52 Ø6 4B Ø6 47 Ø3 72 Ø5 84 Ø6
[7G5]   13: 5Ø ØA 7Ø Ø4 AC Ø7 84 Ø4 96 FC 6C 48 49 48 Ø4 D1
[7M5]   14: Ø9 94 ØE 62 Ø4 C9 Ø6 DD Ø7 B8 Ø6 B5 FF 6C Ø3 7D
[ØG5]   15: Ø7 D9 FD 49 49 6C Ø4 D5 Ø7 91 Ø8 3E Ø3 Ø5 Ø6 Ø2
[8F5]   16: Ø3 29 ØD F3 Ø5 7C Ø5 ØA Ø3 32 Ø6 B4 FE 48 49 Ø4
[LE5]   17: 4A Ø3 4C Ø4 56 Ø4 9A Ø3 A2 Ø3 E1 Ø4 36 Ø4 A7 Ø7
[GM5]   18: 1A Ø7 3A Ø3 69 Ø7 32 Ø8 33 Ø3 8E FF 49 Ø5 8F FF
[UL5]   19: 4E Ø3 9A Ø6 31 FF 4E Ø9 89 FE 9Ø 6C Ø5 B1 Ø6 6A
[5G5]   2Ø: Ø6 3E Ø7 8D Ø3 D4 FF 49 Ø7 CD Ø8 14 Ø5 C2 Ø4 54
[FH5]   21: Ø4 57 Ø7 18 Ø7 4A Ø7 FB FC 9Ø 6C 48 ØØ Ø5 E3 Ø5
[1P5]   22: ØE Ø6 E5 Ø7 B2 Ø8 D8 Ø5 Ø4 Ø3 E1 Ø5 ØF Ø7 1B FF
[HE5]   23: 6D Ø5 6D Ø6 5A FF ØØ Ø6 ØF Ø3 Ø9 Ø3 ØA Ø6 1D Ø4
[NM5]   24: DF Ø5 CC Ø3 47 Ø3 94 Ø6 EA Ø8 Ø2 Ø7 DA Ø8 EE Ø5
[RG5]   25: 2A Ø5 3C Ø5 24 Ø9 D9 Ø3 ØB Ø6 67 Ø9 6C ØC 14 ØA
[NB5]   26: 7Ø Ø3 4Ø Ø6 9C Ø4 53 Ø4 98 Ø3 D1 Ø7 12 Ø4 ØB Ø8
[1L5]   27: A8 Ø8 12 FF 4E 1Ø AC Ø5 4Ø Ø3 C1 ØA 13 Ø7 68 Ø3
[VK5]   28: DB Ø6 61 Ø4 1C Ø6 26 Ø5 2B FF 2A ØF AB FB ØØ ØØ
[995]   29: 6C ØØ ØØ Ø8 75 Ø4 17 Ø5 13 Ø5 5A Ø5 21 Ø7 79 Ø6
[ZF5]   3Ø: 5C FB 48 48 24 2A 9Ø Ø6 7F Ø5 4D Ø3 26 FF ØØ Ø3
[9E5]   31: 81 Ø5 47 Ø4 59 Ø8 ØE FE 49 49 Ø4 68 Ø6 5Ø Ø3 DØ
[7H5]   32: Ø5 D2 Ø5 3C FA 24 24 2A 48 9Ø 72 Ø5 53 FD 6C 49
[EC5]   33: 4E Ø4 A9 Ø3 44 Ø6 54 Ø5 58 Ø6 1D Ø3 D4 Ø5 66 Ø3
[MP5]   34: D7 Ø9 6Ø Ø5 AA FF 72 Ø3 AF Ø3 9C FF 4E Ø7 1D Ø4
[3T5]   35: BC Ø3 C1 FF ØØ Ø6 E8 Ø7 DF Ø6 D1 Ø4 E4 Ø4 D4 Ø3
[KN5]   36: 17 Ø5 9E Ø6 ØD FC 72 24 2B 2A Ø3 7E Ø6 35 FE 6C
[ØA5]   37: 6C Ø4 F5 Ø3 82 Ø5 1Ø Ø8 23 Ø5 ØB Ø4 32 Ø4 19 Ø3
[3L5]   38: 57 Ø3 DB Ø5 D6 Ø6 35 ØC 2F FF 4E Ø5 9C Ø6 41 Ø7
[PG5]   39: Ø4 Ø8 Ø9 Ø6 8C Ø4 58 Ø3 1F ØB 63 ØB 2E Ø5 D9 13
[YK5]   4Ø: 7E Ø8 C8 Ø5 8E Ø4 13 Ø5 6Ø Ø8 D3 ØB 6F Ø3 54 Ø9
[XF5]   41: BD FE ØØ ØØ Ø3 Ø5 Ø9 94 FF 6C Ø3 Ø8 Ø3 Ø6 Ø4 4E
[RH5]   42: Ø5 A3 FE 48 49 Ø4 Ø6 Ø5 99 FE 24 24 ØA B1 Ø3 BA
[XE5]   43: Ø9 FE Ø9 D1 Ø4 16 ØA 53 Ø4 1B Ø4 3A Ø4 5A ØE 2A
[TD5]   44: Ø5 DØ Ø4 79 Ø5 81 Ø7 7D Ø7 59 Ø8 93 Ø6 D8 Ø4 9A
[BM5]   45: FE ØØ 24 ØC 68 FF 2B Ø5 AA Ø9 7A Ø4 BB ØA 54 Ø7
[TP5]   46: BA Ø9 ØC Ø5 D3 Ø3 25 FA 2B 6C 2B 2B 9Ø Ø3 AF
[FH5]   47: FF 48 Ø3 EA Ø7 79 ØC 86 Ø6 91 Ø6 DØ Ø4 A2 Ø8 B6
[PN5]   48: Ø6 EØ FE ØØ 24 Ø4 E9 FF 24 Ø3 2C FD 2A 48 2A Ø3
[ZT5]   49: F1 Ø3 24 FF 2B Ø4 ØF Ø9 FF Ø6 19 FF 6C ØC CD Ø7
[5B5]   5Ø: 56 Ø5 5B Ø5 27 FB 48 24 2A 24 2A Ø3 33 Ø3 71 Ø3
[4F5]   51: 6C FE 2B 48 Ø5 3B Ø3 51 Ø7 46 Ø5 A2 Ø3 ØD Ø6 3B
[ZN5]   52: Ø4 86 FE 2A 6C Ø3 72 FF 6C Ø4 74 Ø4 67 FF 2A Ø3
[7K5]   53: 6E Ø4 6F Ø4 7Ø Ø3 9B Ø3 BA FF 24 Ø6 C4 ØA D3 Ø7
[CV5]   54: 7C Ø3 CE Ø3 ØC Ø3 38 FE 2B 2B Ø4 CF FD 2A 2B 2B
[FK5]   55: Ø5 A7 Ø3 EF ØD EE Ø3 BC Ø6 D1 Ø4 16 Ø5 D8 Ø6 93
[WT5]   56: Ø3 4D Ø6 88 ØA DD FF 2B Ø3 FD FD 6C 6C 24 Ø4 B9
[GD5]   57: Ø5 11 Ø4 72 Ø3 AB Ø3 1F ØA C1 Ø5 Ø9 Ø4 4B Ø4 4C
[CF5]   58: Ø5 47 ØD 1D FF 2B Ø5 73 Ø7 31 Ø7 37 Ø4 ØB ØC C2
[8L5]   59: Ø4 85 Ø6 4B Ø6 84 Ø7 5E Ø3 69 FD 2B 16 2B ØA 6C
[PC5]   6Ø: Ø4 36 FF 16 Ø3 7A Ø5 ØA Ø4 C2 Ø4 52 Ø8 54 Ø5 95
[HH5]   61: Ø5 CA Ø4 1B Ø5 61 Ø5 9B FD 2B 48 24 Ø7 EC Ø6 76
[WK5]   62: Ø3 A3 Ø4 97 ØE 81 Ø8 Ø5 Ø4 C8 Ø3 FD ØB DE FF 24
[WR5]   63: Ø3 FA FF 2A Ø6 FØ Ø3 F8 FF 16 Ø3 21 Ø3 FB FE 6C
[XD5]   64: 6C Ø3 Ø1 Ø4 C3 Ø4 19 FF ØØ Ø9 8A Ø9 11 Ø7 1E Ø4
[ØH5]   65: AB FE 24 24 Ø3 61 Ø3 3Ø Ø3 6F FC 6C 2A 24 48 Ø3
[CG5]   66: 76 FE 24 2A Ø6 BE Ø6 C3 Ø3 81 FF 72 Ø7 9Ø ØA Ø4
[AD5]   67: Ø6 5E Ø3 7A Ø4 39 Ø3 7Ø Ø4 33 Ø3 B2 Ø5 A9 ØC 7A
[AG5]   68: Ø4 CØ FD ØØ ØØ Ø7 Ø9 ØA Ø8 95 Ø8 9E FE 24 2A Ø5
[EH5]   69: C7 Ø4 AC Ø6 B1 FE ØØ ØØ Ø3 E3 Ø4 D6 ØC CØ FE ØØ
[RU5]   7Ø: 72 Ø3 8C Ø9 CD Ø8 DA FF 2A ØB E3 Ø3 ED Ø4 D5 Ø6
[DM5]   71: F5 Ø4 BC ØE FF Ø4 8B Ø3 51 Ø4 69 Ø4 13 FF 9Ø ØD
[DD5]   72: 1D Ø3 66 Ø6 2D Ø4 A4 Ø3 69 14 3A Ø4 81 Ø3 7E Ø4
[DE5]   73: 54 Ø3 8E ØE 5C FE ØØ 24 Ø3 94 Ø3 74 Ø4 72 Ø3 A8
[8L5]   74: Ø6 79 Ø6 8C ØC 85 Ø6 29 FF Ø7 13 98 Ø4 D3 Ø9 AF
[FM5]   75: Ø6 78 Ø7 8B 1Ø C5 Ø9 14 Ø3 1E FF 2A 14 E2 FE 4F
[NG5]   76: 4F Ø6 38 ØB FE FF Ø6 Ø4 4A Ø6 49 Ø3 53 F8 Ø9 9Ø
[FH5]   77: 9Ø Ø7 Ø7 9Ø 9Ø 2B Ø8 28 FF 24 ØD 28 Ø3 6A F8 4F
[FF5]   78: 4F 4F 24 2A Ø6 2B 2B Ø4 4D 13 Ø4 Ø3 59 FB Ø7 Ø7
[3M5]   79: 48 48 9Ø Ø4 52 ØC CB FE 24 24 Ø6 B1 FB 2A 2A Ø6
[8G5]   8Ø: 4F 4F Ø5 53 ØE C1 Ø7 A1 FC 48 48 ØØ 48 Ø7 99 Ø4
[YU5]   81: 1B Ø8 CE Ø8 AD Ø4 F6 Ø3 F8 FE 4E 4F Ø6 4D FF 48
[ØN5]   82: Ø3 C4 ØE 85 Ø3 DD Ø4 D9 FF Ø7 Ø6 DE Ø7 CB Ø5 E4
[UU5]   83: Ø5 E6 Ø4 BC Ø3 78 FE 4F 24 Ø3 FE FF Ø6 Ø3 Ø4 FF
[TH5]   84: 6C Ø3 Ø8 Ø3 44 ØB A1 FE 48 Ø7 ØB D9 Ø8 21 ØA 62
[EK5]   85: Ø4 39 FF Ø6 Ø3 B7 Ø3 7D FF 48 Ø3 81 FF Ø6 Ø3 43
[7R5]   86: Ø7 BD Ø9 CF FB 48 Ø7 2B 48 9Ø Ø9 5F Ø4 F8 ØA AA
[HD5]   87: FE 4E 4E Ø3 89 Ø6 7B Ø3 8Ø FF Ø6 Ø3 46 Ø4 Ø4 Ø3
[WP5]   88: 8E FE Ø7 Ø7 Ø8 6E FD 48 48 2A Ø3 9B ØA 9F ØE A4
[4T5]   89: FF 4E Ø6 F6 FF 2B Ø5 BF FF ØØ Ø4 88 Ø3 E7 Ø4 1A
[WT5]   9Ø: ØB 4D FE 2A 2A ØC DC Ø7 F6 Ø6 2E Ø3 8F FC Ø6 4E
[7G5]   91: 4E 48 Ø5 B5 Ø3 4Ø Ø3 Ø4 Ø7 EF Ø4 ØB Ø7 CF Ø7 D9
[YF5]   92: Ø4 37 Ø6 3Ø FD 48 Ø1 2B Ø3 ØØ Ø4 FF Ø5 39 Ø3 3D
[7E5]   93: Ø8 3C Ø4 46 Ø4 ØF Ø6 46 Ø3 82 Ø3 5F FD Ø1 48 9Ø
```

Listing 28. ankunft.pak ist eine gepackte Grafikdatei.

```
[VE5]    94: Ø4 5B Ø4 61 F9 2A 48 Ø6 48 2A 48 Ø1 Ø6 74 Ø5 AF
[5F5]    95: Ø4 81 Ø4 AB Ø3 FD Ø6 8Ø Ø4 35 Ø6 ØF Ø5 8E Ø3 D5
[AN5]    96: FD Ø7 48 9Ø Ø3 D8 Ø4 AØ Ø3 C8 FF 48 Ø4 A8 FB 4E
[TK5]    97: 2A 2A 2B 2B Ø5 D1 Ø7 63 FE 2B 2B Ø7 BD Ø4 5E Ø5
[3H5]    98: 88 Ø6 Ø6 Ø6 48 Ø6 99 FF ØØ Ø3 A4 Ø3 6E FF Ø7 Ø4
[PW5]    99: E9 FD 2A 2B Ø6 Ø4 EE FF 2A Ø5 ØA FE 2B ØØ Ø7 FA
[NE5]   1ØØ: Ø4 CC Ø6 4Ø Ø5 B5 Ø9 4F Ø7 99 FE 48 ØØ Ø3 6C Ø3
[4K5]   1Ø1: 62 Ø4 29 Ø4 2A Ø3 68 FE Ø7 Ø7 Ø5 CB Ø3 2C Ø3 3D
[MH5]   1Ø2: Ø5 75 Ø7 48 Ø4 F3 Ø3 6F FC Ø6 Ø1 Ø1 Ø1 Ø3 E3 FE
[3N5]   1Ø3: 9Ø ØØ Ø4 1C FF 2B Ø3 64 FD 2B 2A 4E Ø4 67 Ø7 AA
[VP5]   1Ø4: Ø5 Ø8 Ø3 3B Ø4 B9 Ø8 7E Ø4 6F FE 2A 2A Ø3 57 FF
[RH5]   1Ø5: Ø6 Ø5 CE Ø3 63 FF Ø1 Ø3 9C FF 2A Ø3 5D FE Ø6 48
[EG5]   1Ø6: Ø3 CB Ø5 27 Ø6 AD Ø4 89 Ø6 B6 Ø4 F4 Ø7 FF Ø6 C8
[TK5]   1Ø7: FE 4E 2A Ø5 CF Ø4 11 FD Ø1 Ø1 ØØ Ø3 D4 FF 2A Ø3
[GH5]   1Ø8: 1Ø Ø4 Ø7 Ø6 E9 Ø3 96 Ø5 2C FF 2B Ø4 3A Ø4 FD ØB
[6U5]   1Ø9: FB Ø5 55 FF 48 Ø9 CE FE ØØ 48 Ø8 84 FF 48 Ø9 E4
[2B5]   11Ø: Ø5 94 Ø5 4Ø 1Ø 33 Ø4 1F Ø3 1C Ø3 1D Ø4 8A Ø4 56
[UF5]   111: Ø5 52 Ø5 5E Ø8 61 FE Ø6 2B Ø3 15 Ø5 6B FF ØØ Ø5
[HK5]   112: 7D Ø7 76 Ø6 7E Ø3 9E Ø6 BE FE 48 48 Ø3 16 Ø5 79
[9N5]   113: FE 2B 2B ØA 2Ø FE 2A 2A Ø5 7B Ø3 AA Ø4 EC Ø5 59
[UC1]   114: Ø5 D4 Ø5 FB
```

Listing 28. ankunft.pak ist eine gepackte Grafikdatei.

Wir kämpfen für

WENIGER MÜLL!

Ich möchte mehr über Greenpeace wissen!
Für Ihre Kosten habe ich 3,60 DM in Briefmarken beigelegt:

Vorname, Name

Straße, Hausnummer

Postleitzahl, Ort, Zustellpostamt

GREENPEACE

Greenpeace e.V., Vorsetzen 53, 2000 Hamburg 11
Spendenkonto: Nr. 2061-206, PGiroA Hamburg, BLZ 20010020

Z00020


```
[SG5]     1: FC ØF Ø9 Ø8 Ø9 3C ØØ FA Ø9 Ø9 Ø7 Ø9 Ø9 Ø8 3A 42
[LK5]     2: FC Ø8 Ø7 Ø7 Ø8 3C 8Ø Ø3 3F 3D C2 FE ØØ Ø7 Ø5 Ø1
[1K5]     3: FC 4E 72 4E ØØ 2F ØA Ø5 Ø2 FD ØØ ØØ Ø8 Ø3 41 FB
[WM5]     4: ØØ ØØ 81 AC 81 Ø3 3E 1A 4C FD 4A 51 4A Ø3 39 FD
[8K5]     5: ØØ 15 16 ØB 5C Ø6 41 FB Ø3 Ø9 Ø3 Ø8 3A Ø5 46 FB
[ZG5]     6: Ø7 Ø7 15 1C 15 Ø3 8A FF 54 Ø8 33 Ø7 94 FD 3Ø 38
[2L5]     7: 3Ø Ø8 66 FE 81 81 Ø5 9E FD Ø3 ØA Ø3 Ø4 78 FE Ø3
[PH5]     8: Ø8 Ø3 B7 Ø3 BC FD Ø8 16 ØØ Ø4 Ø7 FE 51 ØØ Ø5 8D
[6F5]     9: FD 54 7F 54 Ø6 Ø1 Ø8 DA Ø3 87 F9 ØØ 16 41 16 ØØ
[1V5]    1Ø: 55 ØD ØD AD FF Ø7 ØA BB FF ØØ Ø5 C6 FF AC Ø5 86
[TL5]    11: FF 51 Ø3 D1 FF 55 ØA 35 Ø3 AB Ø8 A3 Ø3 EE Ø3 6E
[4K5]    12: F6 9D C7 C1 ØØ Ø3 41 Ø3 ØØ 31 ØF 1Ø FB Ø5 CB F6
[RH5]    13: 51 52 AC Ø7 55 7F 55 Ø7 Ø7 1Ø Ø8 DB Ø6 22 FE 65
[GR5]    14: 4A Ø4 EA Ø4 2E FF AC Ø3 33 Ø3 F6 17 39 FD ØØ AC
[NR5]    15: 52 Ø5 11 FE ØF ØF Ø5 EE ØC 1F 26 6B FE ØØ 51 Ø3
[GK5]    16: 92 FD 55 7F 54 Ø3 2A FF 1Ø Ø5 59 2B A2 Ø3 ØD Ø3
[ZW5]    17: 9C Ø6 D2 FB 55 ØØ ØØ ØF ØF Ø5 DC ØF E3 FF A3 ØF
[3N5]    18: F3 FF Ø9 Ø8 43 FF AC Ø6 CC ØA 11 FD Ø7 Ø7 ØF Ø4
[3M5]    19: DB Ø4 ØD FD 16 65 16 Ø8 6A FF A4 Ø3 73 FE 41 Ø3
[7F5]    2Ø: Ø3 D9 FF Ø8 Ø3 41 ØC 3F Ø7 CB Ø4 51 Ø3 9D Ø3 19
[MS5]    21: Ø4 9B FE ØF ØF Ø4 22 FF Ø7 Ø3 A7 FC Ø7 55 ØD Ø7
[PN5]    22: Ø8 6E ØF B6 FF Ø9 35 C5 ØA FB 37 Ø4 Ø9 FC Ø6 9A
[US5]    23: 3Ø 4A 3A 3B FD ØE 15 ØE 2A 78 FE 73 4F Ø9 D9 Ø8
[ØS5]    24: C3 Ø8 B5 FF Ø9 Ø4 FE FC Ø9 Ø3 Ø8 Ø7 1Ø D2 F9 Ø7
[7S5]    25: 4A Ø7 Ø7 54 5C 54 ØC DD FC 33 3A 33 Ø7 Ø9 ED FF
[ØL5]    26: AC ØE B7 FD Ø7 33 16 Ø8 47 FF Ø7 Ø5 5Ø F4 4A 5Ø
[ZD5]    27: 75 Ø7 81 AC 81 Ø7 32 38 32 Ø7 Ø3 5B Ø4 64 Ø4 29
[ZP5]    28: F9 41 16 Ø7 4F 74 4F 4F Ø5 59 FE 81 56 ØC 3B FF
[9L5]    29: 81 Ø7 87 Ø9 5Ø FE 4A 4A Ø5 75 FB 87 81 Ø7 73 4F
[XE5]    3Ø: Ø5 9Ø Ø4 8C Ø8 6D Ø3 35 Ø3 61 Ø7 FB Ø5 82 Ø9 C6
[UG5]    31: Ø9 91 Ø4 19 FC Ø7 32 87 32 Ø4 A1 FD 51 57 51 Ø5
[AT5]    32: 68 Ø4 C3 FF 5E Ø3 EA Ø3 F5 ØE 78 FF 54 Ø9 Ø6 Ø9
[WL5]    33: D1 17 D9 Ø6 BØ 1Ø F6 Ø4 A8 Ø5 49 FC 3Ø Ø7 Ø7 ØF
[AE5]    34: Ø6 14 Ø6 19 Ø3 5F 24 22 FF 3Ø Ø9 86 FC Ø7 ØF 16
```

```
[ØP5]    35: 16 Ø5 55 1D 59 FF 81 ØE 77 FF 75 Ø8 C5 Ø4 11 FD
[WH5]    36: 16 16 ØF Ø4 96 1Ø 99 Ø3 4C Ø9 AC FA 81 AC 81 Ø7
[RR5]    37: AC 81 ØA FB FF 5Ø Ø7 48 Ø4 D6 FE 75 4A Ø6 D4 Ø5
[2M5]    38: F4 Ø7 DE Ø4 1A 1C 29 FF 5Ø Ø8 45 Ø4 16 Ø5 95 2E
[8N5]    39: 57 FF Ø9 35 85 ØA BB 37 C4 Ø9 FC FE Ø7 ØØ 33 Ø6
[3U5]    4Ø: FF Ø7 4B FB FD 4A 75 4A Ø5 75 FD 73 7A 73 Ø5 89
[1T5]    41: FD 31 5B 31 16 64 FD 4F 6C 4F 14 72 FD ØF 5E ØF
[PN5]    42: ØD 89 FD 81 AB 81 16 99 FD 33 5E 33 18 72 FD 31
[HS5]    43: 5C ØC Ø5 CD FD Ø2 27 Ø2 Ø6 95 FF 73 Ø9 B1 FF 6E
[EZ5]    44: Ø6 91 FE 5A 3Ø ØF AE FF Ø9 ØB FE FF Ø7 Ø3 ØA FF
[SL5]    45: Ø7 Ø3 ØE FF Ø7 Ø3 12 FF Ø7 Ø3 16 FB Ø7 Ø7 56 56
[7H5]    46: Ø7 Ø6 5D F5 6E 93 6E Ø6 ØF ØF Ø7 81 AC 81 Ø7 Ø3
[MT5]    47: 2F Ø7 5D FF Ø7 ØF 7A FC Ø7 33 39 33 ØE 4D FF 73
[NN5]    48: Ø7 71 Ø6 64 FE 16 ØF Ø3 2B Ø5 6E FF 3A Ø7 B3 ØF
[FS5]    49: 7B ØF 4A FE ØF Ø2 Ø7 9B Ø8 A3 Ø9 AA FE 33 81 Ø5
[LT5]    5Ø: F4 1F 3A Ø8 D9 Ø9 E2 FF Ø6 Ø9 EB Ø7 33 13 3B FD
[BM5]    51: 81 DØ 81 ØF 11 FA Ø8 ØF 6E Ø6 4A 4A Ø4 27 FF Ø7
[BS5]    52: Ø8 2B FF 39 Ø7 73 FE Ø2 Ø9 11 BD Ø8 4E FD 81 AB
[2H5]    53: 81 Ø5 59 FA Ø6 6E ØF Ø8 Ø6 ØF Ø3 86 Ø3 68 F6 Ø7
[VM5]    54: 81 AC 81 Ø7 33 5E 33 Ø7 55 Ø7 B3 ØB BB FD ØF 5E
[ZN5]    55: ØF Ø9 C9 FD 1Ø 3B 1Ø Ø6 95 FD 56 56 Ø6 Ø7 21 FE
[HP5]    56: 4A 4A Ø3 A8 Ø5 AA Ø4 EF FF Ø7 Ø6 F4 ØC BA Ø4 Ø6
[PK5]    57: FF ØC 13 ØB FE Ø6 6E Ø3 E8 Ø3 66 Ø3 2Ø FF Ø7 Ø9
[LR5]    58: 6A 12 B3 FF Ø9 35 45 ØA 3B 37 84 ØA BC FF ØØ 33
[2W5]    59: C6 FF Ø7 4B BB FD 4F 79 4F 34 Ø9 FF Ø9 Ø8 3E FD
[HD5]    6Ø: 81 AC 81 Ø5 75 F5 Ø2 Ø9 Ø2 ØØ 55 7F 55 ØØ 4A 75
[DS5]    61: 4A Ø5 89 FD 49 78 49 Ø5 99 FD 26 2D 26 Ø5 A1 FD
[XF5]    62: 5Ø 75 5Ø Ø5 A9 Ø4 46 ØC 7A Ø8 46 FE Ø2 17 Ø6 9Ø
[SP5]    63: F5 ØF 16 ØF ØØ ØF 3A ØF ØØ 9E A5 9E Ø9 A1 FD 3Ø
[LG5]    64: 7E 3Ø Ø5 D5 17 B2 FF Ø7 Ø4 Ø9 FC 34 17 34 Ø7 Ø3
[WF5]    65: D2 FF Ø7 Ø3 96 FF Ø7 Ø3 DA FF Ø7 Ø3 9E Ø5 Ø9 Ø3
[SG5]    66: E6 FF Ø7 Ø3 EA FF Ø7 Ø3 EE F7 Ø7 31 39 31 Ø7 56
[LK5]    67: 81 56 Ø7 1Ø FA FC 32 5D 32 Ø7 Ø3 CE 11 11 FD 54
[Z95]    68: 8Ø 54 Ø9 25 Ø3 AE Ø5 35 Ø5 Ø6 Ø3 42 ØC 3E Ø4 72
[2G5]    69: Ø3 8E 11 51 FD 81 AC 81 4D 65 Ø4 32 14 B6 Ø5 4A
[AM5]    7Ø: Ø3 CF Ø4 E2 Ø4 2E FD 5D 88 5D 15 5D 38 F2 Ø8 52
[BG5]    71: 14 32 Ø4 6A Ø4 4A FD 34 17 34 Ø5 91 Ø3 6E FF Ø6
[BS5]    72: Ø4 5A FD 9E A5 9E 25 61 ØC 46 Ø7 D2 FF Ø6 Ø8 DA
[2K5]    73: 23 22 FF Ø9 35 Ø5 ØA 3B 37 44 ØA 3C FF Ø9 Ø3 85
[PG5]    74: Ø3 Ø4 Ø5 89 ØA 9Ø ØD 99 ØE A2 Ø5 98 ØD 7C Ø7 C6
[ZA3]    75: ØB 9Ø ØE 98 Ø9 87 Ø7 E8 Ø7 B9
```

Listing 29. buch.pak ist eine gepackte Textdatei.

```
[GD5]     1: BD 44 75 2Ø 6C 69 65 73 74 2Ø 64 61 73 2Ø 42 75
[YL5]     2: 63 68 3A ØD ØA 5B 41 6E 6E 61 6C 65 6E 2Ø 64 65
[KE5]     3: 73 2Ø 53 63 68 6C 6F 73 73 65 73 ØD ØA 2Ø 4D 61
[1K5]     4: 73 6B 61 76 61 6E 5D ØD ØA ØD ØA 22 49 6D 2Ø 4A
[ØF5]     5: 61 68 72 65 Ø5 1C F6 48 65 72 72 6E 2Ø 31 33 33
[BD5]     6: 37 Ø3 11 EØ 2Ø 57 69 72 2C 2Ø 75 6E 73 65 72 2Ø
[YL5]     7: 57 6F 68 6C 67 65 62 6F 72 65 6E ØD ØA 47 72 61
[NK5]     8: 66 2Ø 44 6F Ø3 32 FC 2Ø 76 6F 6E Ø9 2C FC 2Ø 68
[WG5]     9: 61 62 Ø4 6A FA 62 65 66 75 6E 64 Ø3 1A FF 69 Ø3
[3M5]    1Ø: 1B Ø3 Ø4 F7 65 6D 2Ø 54 65 69 6C ØD ØA Ø4 44 FE
[GG5]    11: 4C 61 Ø3 91 FA 73 2Ø 65 69 6E 65 Ø3 55 FE 6E 74
[CH5]    12: Ø3 49 FD 65 73 69 Ø3 92 FA 7A ØD ØA 7A 75 2Ø Ø3
[8H5]    13: BA F2 69 63 68 74 65 6E 2E 2Ø 45 73 2Ø 67 65 68
[6M5]    14: Ø3 93 FC 7A 77 61 72 Ø3 6C FC 65 72 75 65 Ø4 CD
[BK5]    15: FC 2Ø 75 6D 2C Ø4 Ø8 Ø3 AF FB 73 2Ø 68 69 65 Ø3
[3E5]    16: EØ F7 76 65 72 68 65 78 74 2Ø 73 Ø3 B1 FB 2Ø 73
[2N5]    17: 6F 6C 6C Ø3 5A FF 64 Ø4 FØ FC ØD ØA 42 6F Ø3 9B
[ØN5]    18: FC 2Ø 72 65 67 Ø3 FA FD 74 2C 2Ø Ø3 87 FF 72 Ø6
[ØL5]    19: 5A FA 65 ØD ØA 52 69 74 Ø3 B9 Ø6 85 FC 2Ø 6E 75
[RF5]    2Ø: 72 Ø5 BØ Ø3 8A F6 6D 65 72 6B 77 81 72 64 69 67
[BF5]    21: Ø3 4Ø Ø4 E3 FE 63 68 Ø9 95 Ø5 EØ Ø3 57 F9 64 2Ø
[LL5]    22: 66 65 73 74 67 Ø3 75 FD 65 6C 6C Ø3 26 F8 77 6F
[7N5]    23: 62 65 69 ØD ØA 6E Ø4 CC F7 2Ø 61 75 73 7A 75 6D
[NG5]    24: 61 63 Ø4 D9 Ø3 DE Ø3 8Ø FE 65 6C Ø4 96 F6 ØD ØA
[2E5]    25: 55 72 73 7Ø 72 75 6E 67 Ø3 F4 EB 72 2Ø 69 73 74
[ZG5]    26: 2E ØD ØA ØD ØA 2Ø 49 6D 2Ø 4A 61 68 72 65 2Ø 64
[LH5]    27: Ø3 F6 F5 48 65 72 72 6E 2Ø 31 33 33 38 3A Ø3 BC
[KN5]    28: FD 41 75 66 Ø6 2D Ø3 5A FD 68 65 69 Ø3 F3 F8 77
[FC5]    29: 75 72 64 65 2Ø 76 6F Ø3 A6 FF 64 Ø3 E1 Ø3 47 Ø3
[4R5]    3Ø: E6 FD 6D 69 73 Ø6 A3 FA 42 65 76 94 6C 6B Ø3 5C
[EG5]    31: FA 6E 67 2Ø 6D 69 74 Ø3 C7 FB 6D 2Ø 42 61 75 Ø4
[7H5]    32: C7 Ø4 6B F3 6D 61 65 75 65 72 73 2Ø 62 65 67 6F
[KM5]    33: 6E Ø3 49 Ø3 DD FF 64 Ø4 FA FC ØD ØA 61 6E Ø3 78
[6N5]    34: Ø3 E5 FF 6E Ø3 F8 DF 53 63 68 6C 6F 73 73 61 72
[2F5]    35: 63 68 69 74 65 6B 74 ØD ØA 62 65 61 75 66 74 72
[6D5]    36: 61 67 74 2E 2Ø 45 72 2Ø Ø3 Ø3 FC 61 65 74 7A Ø3
[KL5]    37: 18 FA 69 65 ØD ØA 5A 65 Ø3 17 FF 66 Ø3 2C FA 2Ø
[SG5]    38: 64 61 73 2Ø 6E Ø3 2B Ø8 4E Ø3 41 Ø3 DB FD 6D 69
[PE5]    39: 6E Ø3 22 F6 74 65 6E 73 2Ø 32 3Ø 2D 32 35 Ø6 C1
[KN5]    4Ø: FE ØD ØA Ø3 3E FA 2E 2Ø 57 69 72 2C Ø7 DD F7 57
[BM5]    41: 6F 68 6C 67 65 62 6F 72 Ø4 Ø6 FB 44 6F 76 61 6E
[9L5]    42: Ø4 F1 FB 2Ø 4D 61 73 6B Ø3 D5 FB 2C 2Ø 68 61 62
[BD5]    43: Ø5 37 Ø3 97 F6 65 72 6C 61 75 62 74 2Ø 69 68 Ø3
[7N5]    44: E3 FF 64 Ø3 E9 FF 41 Ø8 57 FB 65 6E 2C ØD ØA Ø8
[SN5]    45: 16 FC 45 69 6E 62 Ø3 1F Ø3 D9 FD ØD ØA 76 Ø3 2D
[EL5]    46: Ø3 Ø4 FF 65 Ø3 FE Ø3 F1 FB 65 69 ØD ØA 47 Ø3 48
[5G5]    47: FB 69 6D 67 61 65 Ø3 44 Ø4 EA FD 64 2Ø 67 Ø5 37
[5L5]    48: Ø4 CF F5 5A 69 6D 6D 65 72 6E 2Ø 7A 75 2Ø Ø9 61
[2N5]    49: FF 65 Ø3 B8 FF 44 Ø4 4E FD 41 72 62 Ø3 7F Ø4 56
[NM5]    5Ø: FD 77 65 72 Ø4 FE FD 62 69 73 Ø3 59 FA 72 ØD ØA
```

```
[ZD5]    51: 42 61 75 Ø3 A3 F6 6C 6C 65 2Ø 64 69 65 2Ø 41 75
[GG5]    52: Ø4 4Ø Ø3 24 FF 62 Ø3 44 Ø3 66 FA ØD ØA ØD ØA 49
[3E5]    53: 6D Ø6 AE Ø3 13 FB 73 2Ø 48 65 72 Ø3 76 F7 31 33
[CE5]    54: 34 36 3A ØD ØA 2Ø 44 Ø5 94 FB 73 68 75 62 61 Ø6
[EN5]    55: 7Ø Ø6 42 FE 64 65 Ø6 86 Ø5 B3 FF 4B Ø4 8E FB 72
[7K5]    56: 73 2Ø 73 69 Ø3 DA FD 62 65 65 Ø3 A2 FF 74 Ø3 A6
[FM5]    57: Ø6 C5 Ø8 D1 FE 61 6E Ø3 E5 Ø3 6C FD 49 6E 6E Ø4
[WT5]    58: 3E FC 62 61 65 75 Ø4 7C FF 69 Ø3 9B FF 6F Ø3 EC
[KR5]    59: FF 6D Ø3 34 FF 61 Ø3 1B 1A A6 FE 35 32 Ø3 C2 F6
[WE5]    6Ø: 4E 61 63 68 64 65 6D 2Ø 6D 65 Ø3 26 FF 65 Ø3 4Ø
[EN5]    61: FF 6E Ø3 79 Ø3 74 FE ØD ØA Ø4 48 FD 2Ø 56 61 Ø3
[NH5]    62: 6A F7 2Ø 47 72 61 66 2Ø 44 6F 76 Ø3 ØF FE 76 6F
[AH5]    63: Ø3 14 FB 4D 61 73 6B 61 Ø4 81 FA 64 75 72 63 68
[ML5]    64: 2Ø Ø3 5E EE 65 ØD ØA 7Ø 6C 6F 65 74 7A 6C 69 63
[FM5]    65: 68 65 2Ø 6D 79 73 Ø3 76 FC 69 6F 65 73 Ø3 9C FA
[UL5]    66: 4B 72 61 6E 6B 68 Ø3 Ø9 FC 2Ø 64 61 68 Ø3 5F F9
[EC5]    67: 67 65 73 63 68 69 65 Ø5 12 FØ 69 73 74 2C 2Ø 75
[ZG5]    68: 65 62 65 72 6È 65 68 6D 65 2Ø Ø3 A6 FE 2Ø 73 Ø3
[BN5]    69: 99 F9 ØD ØA 53 6F 68 6E 2C Ø6 79 F9 48 65 6E 64
[2E5]    7Ø: 72 69 6B 1Ø 84 FB 69 65 2Ø 56 65 Ø3 B9 F4 74 77
[GM5]    71: 6F 72 74 75 6E 67 ØD ØA 75 6D Ø5 ØE FF 46 Ø3 68
[1P5]    72: FE 69 67 Ø3 AD FE 6C 6C Ø3 1D Ø4 43 Ø3 EC F4 63
[ZG5]    73: 68 6C 6F 73 73 65 73 2E ØD ØA 44 Ø3 26 FD 41 72
[7M5]    74: 62 Ø3 BE FD 65 6E 2Ø Ø3 2F FF 68 Ø3 53 E9 6B 75
[HH5]    75: 72 7A ØD ØA 76 6F 72 2Ø 69 68 72 65 6D 2Ø 45 6E
[NL5]    76: 64 65 2E 2Ø 4D Ø3 E9 FD 2Ø 56 61 Ø3 AE F4 ØD ØA
[WH5]    77: 68 61 74 2Ø 73 6F 67 61 72 2Ø Ø3 CA FF 6F Ø6 ØD
[9M5]    78: FE 4D 6F Ø3 DA FF 6C Ø3 2Ø FC 6E 64 2Ø 61 Ø3 6E
[KH5]    79: F5 72 65 2Ø 47 65 67 65 6E 73 74 84 Ø3 AØ FC 2C
[3H5]    8Ø: 2Ø 77 69 Ø3 B5 FD 52 81 73 Ø4 1C FF 65 Ø3 F1 Ø4
[CH5]    81: 74 FC 42 69 6C 64 Ø6 37 F9 4D 61 67 69 65 72 73
[ZF5]    82: Ø4 ØØ Ø3 4C F9 78 6B 61 68 6E 2Ø 28 Ø4 63 F6 64
[UL5]    83: 65 6D ØD ØA 69 63 68 2Ø 6D Ø4 EF FA 61 6C 73 2Ø
[RP5]    84: 4B 69 Ø3 9C FD 69 6D 6D Ø4 7B F5 66 81 72 63 68
[6H5]    85: 74 65 74 65 29 2Ø Ø3 C4 FC 64 61 73 2Ø Ø7 3D FB
[MN5]    86: ØD ØA 62 72 69 Ø4 BD FD 2Ø 6C 61 Ø3 42 FA 6E 2E
[GN5]    87: 2Ø 44 65 72 Ø3 DC FE 63 68 Ø3 51 FE 6B 74 Ø3 ED
[8H5]    88: FE 73 74 Ø3 12 FD 7A 77 69 Ø3 89 Ø3 28 FE 61 75
[TM5]    89: Ø3 F5 F7 67 65 73 74 6F 72 62 65 6E Ø6 99 F9 64
[6E5]    9Ø: 61 2Ø 65 73 2Ø 6B Ø3 C3 Ø3 B5 FB 41 75 66 7A 65
[5R5]    91: Ø3 F4 FF 6E Ø5 BC FE 2Ø 75 Ø3 95 Ø3 E8 Ø4 B4 F8
[LK5]    92: 47 65 68 65 69 6D 74 75 Ø3 A2 Ø3 7F Ø3 62 FF 5A
[ØE5]    93: Ø5 Ø1 F4 2Ø 67 69 62 74 2C ØD ØA 77 65 72 64 Ø3
[YF5]    94: 96 Ø3 87 F4 73 65 2Ø 77 6F 68 6C 2Ø 6E 69 65 2Ø
[FU5]    95: Ø3 5C FD 75 74 7A Ø3 3F Ø7 AB FB 6B 6F 65 6E 6E
[1N5]    96: Ø5 2F FF 61 Ø3 98 FC 73 2Ø 6E 75 Ø3 Ø5 Ø3 B2 FD
[NH5]    97: 2Ø 45 78 Ø3 42 FC 65 6E 7A 2Ø Ø4 6C FF 73 Ø3 55
[TL5]    98: Ø4 8E Ø4 5D FE 47 61 Ø3 7C FF 73 Ø3 CØ FC 6B 61
[4N5]    99: 6E 6E Ø3 44 FC 73 74 2C 2Ø Ø4 CD Ø3 DD Ø3 E4 Ø3
[NE5]    1ØØ: A1 F8 48 65 72 72 2Ø 56 61 74 Ø3 21 Ø3 F3 FD 6D
[NL5]    1Ø1: 61 6C Ø3 F8 Ø3 75 FF 67 Ø3 C7 FØ 68 61 74 2E ØD
[YE5]    1Ø2: ØA ØD ØA 49 6D 2Ø 4A 61 68 72 65 Ø3 16 FF 73 Ø5
[HC5]    1Ø3: 23 F2 6E 2Ø 31 33 35 33 3A ØD ØA 2Ø 57 69 72 2C
[VL5]    1Ø4: Ø4 1A Ø3 2C FF 57 Ø3 B9 FC 67 65 62 6F Ø3 95 Ø3
[4N5]    1Ø5: Ø1 FD 72 61 66 Ø3 53 EF 6E 64 72 69 6B 2Ø 76 6F
[5H5]    1Ø6: 6E 2Ø 4D 61 73 6B 61 76 61 Ø3 77 FE 67 61 Ø3 C1
[DH5]    1Ø7: Ø6 B1 FF 73 Ø5 49 Ø5 16 FB 67 72 6F 73 73 Ø4 76
[ØP5]    1Ø8: EE 45 72 6F 65 66 66 6E 75 6E 67 73 62 61 6C 6C
[HL5]    1Ø9: 2C 2Ø 61 Ø3 99 F8 65 6D ØD ØA 73 6F 67 61 Ø4 A5
[CL5]    11Ø: F8 72 2Ø 4B 6F 65 6E 69 67 Ø3 66 FD 64 2Ø 73 Ø4
[TE5]    111: F3 FD ØD ØA 41 Ø3 Ø5 F6 74 72 61 75 74 65 2Ø 74
[CM5]    112: 65 69 Ø3 7Ø FD 6E 6F 6D Ø5 FE FF 68 Ø4 96 1A 42
[ZK5]    113: FE 37 38 Ø4 5E FE 48 65 Ø4 FØ F9 72 69 63 68 74
[5K5]    114: 65 74 Ø3 A8 FF 77 Ø3 63 Ø7 78 FE 44 6F Ø3 9Ø Ø5
[XD5]    115: D2 F6 5A 77 65 69 74 65 2C 2Ø 7A 75 Ø3 B2 Ø3 11
[UM5]    116: FD 6E 2Ø 6D Ø4 E3 FB 73 2Ø 56 61 74 Ø3 49 Ø4 DE
[CK5]    117: FE 7A 75 Ø3 CA F7 32 35 2D 6A 61 65 68 72 69 Ø3
[CK5]    118: F8 F7 2Ø 4A 75 62 69 6C 65 75 6D Ø4 14 FA ØD ØA
[MH5]    119: 53 63 68 6C Ø4 AD FE 73 2Ø Ø3 5F F7 2Ø 46 65 73
[1K5]    12Ø: 74 2Ø 61 75 73 Ø3 Ø7 FC 41 75 63 68 Ø5 47 FC 4D
[YF5]    121: 61 67 69 Ø3 AE F9 41 72 78 73 6B 61 68 Ø3 FF Ø3
[8L5]    122: 8A FF 73 Ø3 7E F8 5Ø 6F 72 74 72 84 74 2Ø Ø3 9A
[UN5]    123: Ø5 65 Ø3 8C FD 41 72 62 Ø3 4E FD 73 7A 69 Ø3 FC
[TF5]    124: EE 72 2Ø 68 84 6E 67 74 2C 2Ø 77 61 72 ØD ØA 61
[FK5]    125: 6E 77 65 Ø3 C5 F8 64 2E 22 ØD ØA 3C 44 69 Ø3 2B
[DS5]    126: Ø3 9E FF 6C Ø3 2E Ø3 C6 FF 53 Ø4 4E FC ØD ØA 73
[AK5]    127: 69 Ø3 6C FD 68 65 72 Ø3 A2 FC 67 65 72 69 Ø4 C4
[7D2]    128: FA 3E ØD ØA ØD ØA 1A
```

Listing 30. buch1.pak ist eine gepackte Textdatei.

```
[DC5]     1: D7 44 75 2Ø 6C 69 65 73 74 2Ø 64 61 73 2Ø 42 75
[BF5]     2: 63 68 3A ØD ØA 5B 4D 61 67 69 73 63 68 65 2Ø 57
[YB5]     3: 65 6C 74 2Ø 42 61 6E 64 2Ø 37 Ø3 11 Ø9 15 FD 53
[BN5]     4: 7Ø 72 Ø3 ØE F2 73 61 6D 6D 6C 75 6E 67 5D ØD ØA
[5H5]     5: ØD ØA 22 Ø4 2C FD 65 2Ø 69 Ø3 Ø6 FA 68 69 65 72
[8F5]     6: 2Ø 64 Ø3 4C F6 4D 61 74 65 72 69 65 2C ØD ØA Ø4
[KG5]     7: 58 F8 64 65 6D 2Ø 6C 65 68 72 Ø3 3B F9 65 6E 2Ø
[YP5]     8: 45 6C 65 76 Ø3 75 FE 69 6E Ø5 64 FF 73 Ø3 6B F8
[SE5]     9: 57 65 72 6B 65 2Ø 6E 61 Ø3 32 F2 67 65 62 72 61
[6L5]    1Ø: 63 68 74 ØD ØA 77 65 72 64 Ø3 7C F2 73 6F 6C 6C
[YK5]    11: 2E 2Ø 46 72 65 69 6C 69 63 68 Ø5 4E EC 65 73 ØD
[MH5]    12: ØA 6E 75 72 2Ø 65 69 6E 65 2Ø 45 69 6E 66 81 68
```

```
[9F5]    13: 72 Ø3 4Ø Ø3 7E Ø4 57 Ø3 9D FC 75 6E 64 65 Ø5 71
[GL5]    14: Ø6 1D Ø3 DC Ø3 5B Ø3 4B F6 3B ØD ØA 75 6D 2Ø 77
[RK5]    15: 69 72 6B Ø4 BØ Ø3 E2 Ø5 DA FA 2Ø 7A 75 ØD ØA 76
[7M5]    16: Ø3 A7 FB 62 72 69 6E 67 Ø4 A3 FF 69 Ø3 25 FC 76
[2C5]    17: 69 65 6C Ø3 D6 F3 4A 61 68 72 65 2Ø 41 75 73 62
[NM5]    18: 69 6C 64 Ø4 CD FD 64 75 72 Ø3 B2 Ø4 C1 Ø3 8Ø FC
[SF5]    19: 77 65 69 73 Ø3 16 FF 47 Ø3 2Ø FD 68 72 74 Ø3 47
[WD5]    2Ø: F7 6E 6F 74 77 65 6E 64 69 67 Ø4 63 Ø3 51 Ø5 83
[WR5]    21: FC 73 2Ø 6B 6C Ø5 C1 FC 42 81 63 68 Ø4 6C Ø3 BB
[8P5]    22: FD 69 65 2Ø Ø3 DD FD 65 74 7A Ø3 61 EE 6B 61 6E
[3K5]    23: 6E 2E 2Ø 44 65 6E 6E 6F 63 68 ØD ØA 67 69 62 Ø4
[ML5]    24: B7 Ø4 3B FB 69 67 65 2Ø 47 Ø3 CC F7 64 62 65 67
[5H5]    25: 72 69 66 66 65 Ø4 5D Ø4 7D Ø3 D1 F8 74 61 6C 65
[ES5]    26: 6E 74 69 65 Ø3 4F FE 72 2Ø Ø3 A9 FC ØD ØA 69 6E
[KN5]    27: Ø3 C8 FD 65 73 73 Ø7 C5 FC 53 63 68 75 Ø3 4B FF
[SD5]    28: 72 Ø3 18 Ø4 94 F7 6D 69 74 2Ø 48 69 6C 66 65 Ø8
[KE5]    29: 63 F3 42 75 63 68 65 73 ØD ØA 73 65 6C 62 73 Ø3
[WL5]    3Ø: 9B FF 72 Ø3 E5 Ø3 3E Ø6 88 FB ØD ØA ØD ØA 44 ØA
[KN5]    31: FA FA 2Ø 73 74 65 6C 6C Ø3 ØD Ø3 6E E5 ØD ØA 5A
[1H5]    32: 75 73 61 6D 6D 65 6E 66 61 73 73 75 6E 67 2Ø 64
[6M5]    33: 65 72 ØD ØA 75 65 62 6C Ø3 EA Ø3 2D FA 6E 2Ø 5A
[GC5]    34: 61 75 62 Ø3 8Ø FC 7Ø 72 75 65 Ø3 Ø2 Ø4 CF F4 2Ø
[ØG5]    35: 61 6C 7Ø 68 61 62 65 74 68 69 73 Ø3 66 Ø3 4C FC
[3R5]    36: 52 65 69 68 Ø3 4Ø FE 6F 6C Ø3 A4 FC 64 61 72 3A
[ZH5]    37: Ø4 1D FD 4D 61 67 Ø6 77 FE 2Ø 42 Ø3 19 FD 2Ø 62
[7M5]    38: 72 Ø4 65 FF 6E Ø3 8E F4 3C 42 69 65 67 69 2Ø 76
[ES5]    39: 61 6E 2Ø 6C Ø4 AE FF 3E ØC 8F Ø3 9C FE 72 72 Ø3
[HG5]    4Ø: D9 F5 72 65 6E ØD ØA 61 75 66 68 65 62 Ø7 A7 Ø4
[KL5]    41: CB FB 2Ø 62 65 2Ø 73 Ø3 CF Ø5 BC F3 54 65 6C 65
[6N5]    42: 7Ø 6F 72 74 61 74 69 6F 6E Ø5 33 FC 73 ØD ØA 4C
[EN5]    43: Ø3 DA FF 77 Ø3 23 F4 6E 73 3A 2Ø 28 77 69 72 6B
[LF5]    44: 74 ØD ØA Ø3 42 FA 74 2Ø 6E 69 65 29 Ø3 DF Ø3 F4
[UH5]    45: F7 69 2Ø 69 74 2Ø 67 72 61 6E ØE BB FC 73 2Ø 54
[KU5]    46: 6F Ø3 7C FB 6F 65 66 66 6E Ø6 DC FB 61 6C 6B 61
[1M5]    47: 2Ø Ø3 32 FA 79 2Ø 68 75 72 7A ØE BC FC 57 61 66
[ØK5]    48: 66 Ø4 D2 Ø3 99 FB 73 74 65 6C 6C Ø6 4F F4 52 75
[4E5]    49: 78 2Ø 6B 72 75 79 2Ø 78 75 6D Ø5 63 F7 44 69 65
[KM5]    5Ø: 2Ø 52 65 64 61 6B Ø5 FD FF 75 Ø3 ØA FB 72 6E 69
[CH5]    51: 6D 6D Ø3 1A FF 6B Ø4 Ø2 F3 72 6C 65 69 2Ø 48 61
[9N5]    52: 66 74 75 6E 67 65 Ø3 A4 FB 6E 64 ØD ØA 56 Ø3 7A
[NN5]    53: FF 69 Ø3 6D FF 72 Ø3 CØ F8 73 61 6E 73 7Ø 72 75
[XE5]    54: 65 Ø4 6D FD 66 75 65 Ø3 47 F9 64 61 73 2Ø 66 61
[MN5]    55: 6C Ø5 6C FA 41 6E 77 65 6E 64 Ø3 C3 F5 6F 64 65
[AN5]    56: 72 2Ø 64 69 65 ØD ØA 4E Ø3 CF FB 74 2D 46 75 6E
[SU5]    57: Ø6 AØ FE 76 6F Ø3 76 FD 5A 61 75 Ø3 A8 Ø8 DA FE
[ØH5]    58: 6E 2E Ø6 66 EA 65 72 53 63 68 72 6F 74 74 2Ø 5Ø
[FX4]    59: 52 45 53 53 2Ø 31 33 39 33 2E 22 Ø4 94 FF 1A
```

Listing 31. buch10.pak ist eine gepackte Textdatei.

```
[BD5]     1: AC 44 75 2Ø 6C 69 65 73 74 2Ø 64 61 73 2Ø 42 75
[WE5]     2: 63 68 3A ØD ØA 5B 41 75 66 62 61 75 2Ø 75 6E 64
[2F5]     3: 2Ø 41 72 63 68 69 74 65 6B 74 75 72 2Ø 76 6F 6E
[7K5]     4: ØD ØA 53 63 68 6C 6F 73 73 2D 61 65 68 6E 6C 69
[LF5]     5: 63 68 65 6E 2Ø 47 65 62 61 65 75 64 65 6E 2C ØD
[NR5]     6: ØA 76 6F 6D 2Ø Ø7 31 FF 61 Ø8 21 FF 65 Ø3 2E FA
[DG5]     7: 4D 61 74 74 68 69 Ø3 ØA E6 4F 62 65 72 64 6F 72
[9F5]     8: 66 65 72 5D ØD ØA ØD ØA 22 41 6C 73 2Ø 64 65 72
[DA5]     9: 2Ø 45 72 Ø3 18 Ø3 86 Ø3 2C F7 2Ø 35 32 ØD ØA 42
[HG5]    1Ø: 75 72 67 Ø3 41 Ø4 1C FC 73 61 67 65 Ø6 9F F5 63
[RD5]    11: 68 72 65 69 62 65 ØD ØA 32 31 Ø4 53 F5 6F 65 73
[PK5]    12: 73 65 72 6E 2C 2Ø 67 65 Ø3 BC FD 72 65 2Ø Ø3 3E
[AL5]    13: EF ØD ØA 6F 68 6E 65 2Ø 5A 77 65 69 66 65 6C 2Ø
[HH5]    14: 7A 75 Ø3 84 Ø3 65 FC 65 72 66 61 Ø3 AF FF 6E Ø3
[RM5]    15: C1 Ø9 2F ØB 5B Ø4 84 Ø3 97 FD 72 61 6E Ø3 3F FD
[PE5]    16: 2E 2Ø 49 Ø3 Ø6 F8 69 65 73 65 6D 2Ø 42 75 Ø4 CF
[YG5]    17: Ø3 C6 Ø5 CC FC 2Ø 61 75 66 Ø4 18 F7 2Ø 73 7Ø 65
[GR5]    18: 7A 69 65 6C 6C Ø4 E4 EE 5Ø 72 6F 62 6C 65 6D 65
[HM5]    19: 2Ø 62 65 69 6D 2Ø 41 6E 6C 65 Ø4 9C Ø3 91 FC ØD
[UH5]    2Ø: ØA 57 61 Ø4 BF FB 67 72 61 65 62 Ø7 9D Ø5 Ø8 F6
[CK5]    21: 49 6E 73 74 61 6C 6C 61 74 69 Ø3 92 FF 62 Ø3 BD
[MH5]    22: Ø3 69 F6 47 65 74 69 65 72 73 ØD ØA 69 Ø4 Ø6 FC
[YK5]    23: 6E 73 65 6C Ø4 68 FD 65 69 6E Ø5 14 Ø7 F5 Ø5 F3
[XH5]    24: F7 77 61 72 7A 65 6E 61 75 2C Ø4 4B FE 73 7Ø Ø3
[RP5]    25: 3B FF 73 Ø3 D9 FD 73 65 2C Ø4 24 ED 6C 61 6E 67
[CD5]    26: 2Ø 65 73 2Ø 6D 69 72 2Ø 64 6F 63 68 2C 2Ø 7A Ø3
[WF5]    27: C6 F8 ØD ØA 6D 69 6E 64 65 73 Ø3 Ø4 F6 73 2Ø 38
[8H5]    28: 2Ø 4D 65 74 65 72 2Ø Ø4 DØ F5 65 ØD ØA 4B 72 6F
[CT5]    29: 6B 6F 64 69 6C Ø3 29 Ø5 9D Ø3 1D FF 34 Ø6 F4 FE
[DN5]    3Ø: ØD ØA Ø5 FB FA 6E 2Ø 42 75 72 67 Ø3 64 Ø3 9B FC
[LR5]    31: ØD ØA 75 6E Ø3 19 FB 7A 75 62 72 69 Ø4 2Ø FD 2E
[6L5]    32: 2Ø 44 Ø3 F8 FB 54 72 69 63 6B Ø3 CB Ø3 95 FA 2Ø
[4H5]    33: 69 63 68 2Ø 68 Ø3 8C FC 2Ø 77 69 65 Ø3 7Ø F6 68
[CP5]    34: 6F 6C 74 ØD ØA 61 6E 77 65 Ø3 EA F8 2C 2Ø 69 73
[HG5]    35: 74 2Ø 64 69 Ø3 FF F7 53 65 6E 6B 72 65 63 68 74
[MP5]    36: Ø3 ED FD 6C 6C 75 Ø3 D2 Ø3 5D FE ØD ØA Ø4 25 Ø3
[XH5]    37: 86 F5 65 68 65 75 65 72 2E ØD ØA 49 6D Ø8 A6 Ø6
[7M5]    38: 29 FD 2Ø 76 6F Ø3 2E FC 47 72 61 66 Ø4 AF F4 2Ø
[1F5]    39: 42 72 6F 6E 73 69 6B 2Ø 77 61 72 Ø4 5Ø FB ØD ØA
[LG5]    4Ø: 73 6F 67 Ø3 C7 FE 6D 69 Ø3 71 Ø3 57 F6 49 6E 73
[EK5]    41: 74 61 6C 6C 61 74 69 Ø4 B1 Ø4 ØF FC 73 2Ø 6B 6C
[VN5]    42: Ø4 EB F6 6E 2Ø 44 72 61 63 68 65 6E 73 Ø3 E9 F9
[2P5]    43: 72 66 6F 6C 67 72 65 Ø3 CA FD 2E 2Ø 5A Ø4 C6 FB
```

Listing 32. buch11.pak ist eine gepackte Textdatei.

```
[1D5]    44: 66 72 61 73 73 Ø6 89 Ø6 F9 FE 2Ø 67 Ø3 F2 Ø3 52
[6H5]    45: FA 64 61 72 61 75 66 Ø3 1A Ø3 E8 Ø7 A2 FA 68 65
[ZH5]    46: 72 72 6E 2Ø Ø3 31 FE 2C 2Ø Ø3 AB FF 72 Ø4 72 Ø3
[5D5]    47: ØØ F9 67 65 73 63 68 61 68 Ø3 46 Ø4 19 FF 6D Ø9
[ZH5]    48: 38 Ø7 A9 Ø3 41 Ø4 5F FA 28 21 29 ØD ØA 45 Ø3 F4
[4D5]    49: E9 2Ø 77 65 69 74 65 72 65 2Ø 53 7Ø 65 7A 69 61
[ZS5]    5Ø: 6C 69 74 61 65 74 ØD ØA Ø4 BA FE 6D 69 Ø3 C8 FF
[1K5]    51: 73 Ø3 D7 FD 61 73 2Ø Ø3 82 FF 72 Ø3 2A FF 74 Ø4
[4H5]    52: 36 Ø4 9D F6 47 65 68 65 69 6D 67 84 6E 67 Ø3 73
[PP5]    53: FD 75 6E 64 Ø4 55 Ø4 BF Ø3 C7 F5 5A 69 6D 6D 65
[PG5]    54: 72 6E 2E 2Ø 41 6C Ø3 54 FB 62 65 73 6F 6E Ø3 1B
[HE5]    55: F6 73 2Ø 67 75 74 2Ø 67 65 6C 75 Ø4 C5 Ø4 53 FØ
[BK5]    56: 42 65 69 73 7Ø 69 65 6C 2Ø 6E 65 6E 6E 65 ØD ØA
[MK5]    57: Ø4 2A Ø5 D9 FC 2Ø 77 69 65 Ø3 EB Ø4 5Ø F8 ØD ØA
[FG5]    58: 65 78 74 72 65 6D Ø3 F3 Ø5 D1 F5 2Ø 42 69 62 6C
[MM5]    59: 69 6F 74 68 65 6B Ø3 ØE FE 6E 2Ø Ø7 68 F8 2Ø 4D
[RK5]    6Ø: 61 73 6B 61 76 61 Ø3 DE FE 49 6E Ø3 62 F8 72 ØD
[MK5]    61: ØA 64 6F 72 74 69 Ø4 C6 FA 6E 6F 72 6D 61 6C Ø3
[WG5]    62: 64 ØC 35 FC 62 65 66 69 Ø3 EA FD 74 2Ø 73 Ø4 1Ø
[CP5]    63: FF 65 Ø3 41 FE 6B 6C Ø3 8A Ø4 FC Ø7 2D FF 73 Ø7
[2M5]    64: D7 FF 2C Ø3 A8 FF E1 Ø4 89 Ø3 23 F9 77 65 69 74
[EM5]    65: 65 72 65 Ø7 8D FF 2Ø ØE 26 Ø3 FE Ø9 71 Ø3 A7 F5
[RL5]    66: 72 73 74 65 6C 6C 74 2E ØD ØA 44 Ø3 1E FB 4F 65
[1L5]    67: 66 66 6E Ø3 F7 F5 73 6D 65 63 68 6E 69 73 6D 75
[2L5]    68: 73 Ø3 58 ØA 94 FF 6E Ø7 9F EE 73 2Ø 69 73 74 ØD
[GL5]    69: ØA 6E 75 72 2Ø 6D 69 72 2Ø 75 6E 64 Ø3 FF FF 6D
[KP5]    7Ø: Ø8 43 FB 68 65 72 72 6E Ø4 7A FB 6B 61 6E 6E 74
[AM5]    71: Ø5 22 Ø3 16 FD 2Ø 73 6F Ø7 C7 FF 2C Ø3 5C Ø3 A9
[XM5]    72: Ø4 86 FA 69 68 6E 2Ø 68 69 Ø3 E9 Ø3 BB FD 64 65
[8D5]    73: 72 Ø3 19 Ø3 59 F4 74 2Ø 76 65 72 94 66 66 65 6E
[TF5]    74: 74 6C Ø3 6F Ø3 ØB Ø4 3B Ø3 E5 FE 41 62 Ø3 63 Ø3
[3P5]    75: 8D F9 68 61 74 2Ø 77 61 73 Ø3 1E FF 74 Ø5 AB FF
[FP5]    76: 6D Ø3 CF FF 75 Ø3 5A F9 7A 75 2Ø 74 75 6E 2C Ø4
[HM5]    77: 48 FC 76 69 65 6C Ø5 83 FE ØD ØA Ø4 59 Ø3 74 FC
[NC5]    78: 72 61 74 65 Ø4 87 F8 2Ø 2Ø 4D 61 74 74 68 69 Ø3
[HH5]    79: 98 FF 4F Ø3 8C F1 64 6F 72 66 65 72 2E 31 33 33
[VM1]    8Ø: 36 22 ØD ØA 1A
```

Listing 32. buch11.pak ist eine gepackte Textdatei.

```
[FC5]     1: B9 44 75 2Ø 6C 69 65 73 74 2Ø 64 61 73 2Ø 42 75
[KN5]     2: 63 68 3A ØD ØA 5B 42 65 6B 6F 65 6D 6D 6C 69 63
[MC5]     3: 68 65 2Ø 52 65 7A 65 7Ø 74 65 ØD ØA 66 75 65 72
[TC5]     4: 2Ø 53 7Ø 65 69 73 73 2Ø 75 6E 64 2Ø 54 72 61 6E
[HN5]     5: 6B 5D ØD ØA ØD ØA 22 44 Ø3 Ø4 FF 65 Ø6 ØB D9 2C
[8F5]     6: ØD ØA 6E 69 65 64 65 72 67 65 73 63 68 72 69 65
[ØG5]     7: 62 65 6E 2Ø 76 6F 6E 2Ø 64 65 73 ØD ØA 4B 6F 65
[3F5]     8: 6E 69 67 73 2Ø 62 Ø3 Ø5 E6 65 6D 2Ø 4B 6F 63 68
[6F5]     9: 2Ø 67 69 62 74 ØD ØA 41 6E 6C 65 69 74 75 6E 67
[WH5]    1Ø: 2C 2Ø 77 Ø3 ØA FD 7A 75 72 Ø3 87 FD 75 73 72 Ø3
[GH5]    11: 1D Ø4 8E Ø5 65 F6 67 72 6F 73 73 65 6E ØD ØA 46
[XM5]    12: Ø3 78 FC 6D 61 68 6C Ø3 63 FØ 6E 6F 74 77 65 6E
[NH5]    13: 64 69 67 2Ø 69 73 74 2E ØD ØA Ø3 46 Ø6 2F Ø3 BE
[MS5]    14: FE 73 69 Ø3 38 FD 61 6C 6C Ø3 28 FC 77 6F 68 6C
[GF5]    15: Ø3 5F Ø3 1E Ø3 DF Ø3 37 F7 ØD ØA 76 65 72 75 72
[7S5]    16: 73 61 Ø4 EF FB 2Ø 6B 65 69 6E Ø3 E5 FF 56 Ø3 FB
[4K5]    17: FE 6E 72 Ø3 Ø7 FE 69 67 Ø3 A5 Ø3 Ø3 Ø8 6A FC 72
[7H5]    18: 7Ø 65 72 Ø6 35 Ø3 57 FA 2Ø 53 65 65 6C 65 Ø3 CE
[3G5]    19: F6 ØD ØA 47 65 62 72 61 74 65 6E Ø3 4A FC 57 69
[RK5]    2Ø: 6C 64 Ø3 5C FF 77 Ø3 13 FA 3A ØD ØA 42 65 6C Ø3
[YK5]    21: 6Ø FF 74 Ø3 A8 FF 72 Ø5 E1 FF 6D Ø3 76 F4 69 ØD
[KL5]    22: ØA 66 72 61 6E 7A 6F 65 73 69 Ø3 4D Ø3 1B FE 47
[WV5]    23: 61 Ø3 B7 FE 65 6E Ø3 39 FC 5A 75 6E 61 Ø3 EE FE
[CG5]    24: 73 74 Ø4 CA EF 2Ø 64 61 73 2Ø 54 69 65 72 ØD ØA
[7K5]    25: 7A 75 2Ø 74 6F 65 Ø3 81 FE 2C 2Ø Ø3 9F FF 76 Ø3
[9P5]    26: 9A FD 74 65 69 Ø3 BD FE ØD ØA Ø4 2C Ø3 AA Ø6 12
[TK5]    27: Ø5 82 FD 44 61 6E Ø3 DB FC 74 69 63 68 Ø3 93 Ø3
[MA5]    28: 31 FA 65 72 66 61 68 72 Ø3 44 Ø4 21 F7 63 68 2Ø
[GD5]    29: 64 69 65 2Ø 6C 61 Ø3 19 F8 2Ø 45 69 73 65 6E 73
[RH5]    3Ø: 74 Ø4 FØ Ø3 B6 Ø3 1E FE 2Ø 42 Ø3 41 FE 6F 66 Ø3
[4K5]    31: F8 F1 2Ø 68 69 6E 64 75 72 63 68 2E 22 ØD ØA 3C
[BU5]    32: 44 Ø3 D8 FF 52 Ø3 7F FF 2C Ø3 D6 FA 73 2Ø 73 65
[BF5]    33: 68 72 Ø3 68 F6 6E 75 74 7A 74 ØD ØA 61 75 73 Ø3
[1P5]    34: 2C FD 65 6E 64 Ø3 7A FE 42 75 Ø3 78 FE 73 2C Ø6
[HF5]    35: 9Ø Ø4 15 F4 ØD ØA 46 65 74 74 73 7Ø 72 69 74 7A
[XP5]    36: Ø3 1F F5 73 6F 2Ø 75 6E 6B 65 6E 6E 74 6C Ø3 D2
[HP5]    37: FF 2C Ø3 Ø1 FC 61 E1 2Ø 6E Ø4 D2 FD 73 2Ø 6D Ø4
[7K5]    38: 2D Ø3 BD F7 65 6E 74 7A 69 66 66 65 72 Ø3 ØØ Ø3
[A22]    39: 4E FC 3E ØD ØA 1A
```

Listing 33. buch12.pak ist eine gepackte Textdatei.

```
[7C5]     1: D5 44 75 2Ø 6C 69 65 73 74 2Ø 64 61 73 2Ø 42 75
[6F5]     2: 63 68 3A ØD ØA 5B 31 3Ø 3Ø 2Ø 75 6E 64 2Ø 65 69
[KN5]     3: 6E 65 2Ø 4D 65 74 68 6F 64 65 ØD ØA Ø5 1D E3 4A
[BD5]     4: 75 6E 67 66 72 61 75 2Ø 7A 75 2Ø 62 65 74 6F 65
[4H5]     5: 72 65 6E 2C ØD ØA 76 65 72 66 61 73 Ø3 Ø6 E5 76
[6E5]     6: 6F 6E 2Ø 43 61 73 61 62 6C 61 6E 63 61 2C 2Ø 64
```

```
[UH5]     7: 65 6D ØD ØA 4C 65 68 72 65 72 Ø3 5F FF 73 Ø5 53
[9H5]     8: E6 6E 6F 76 61 5D ØD ØA ØD ØA 22 44 69 65 2Ø 57
[XC5]     9: 65 67 65 2Ø 73 69 63 68 2Ø 69 6E Ø5 Ø8 FC 48 65
[6K5]    1Ø: 72 7A Ø6 29 FD 72 2Ø 6A Ø3 31 FA 65 2Ø 44 61 6D
[UL5]    11: 65 Ø3 38 F9 ØD ØA 73 63 68 6C 65 Ø3 88 FE 65 6E
[FG5]    12: Ø3 86 Ø3 1A F3 76 69 65 6C 66 61 65 6C 74 69 67
[XH5]    13: ØD ØA Ø4 19 Ø5 B1 FF 74 Ø4 38 F7 65 72 6C 65 72
[8S5]    14: 6E 65 6E 2E Ø3 A4 FF 73 Ø3 45 FE 6F 72 Ø3 Ø3 FB
[UN5]    15: 67 65 6E 64 65 Ø3 ØC FF 65 Ø5 AF Ø3 B7 FC 6F 6C
[ØP5]    16: 6C 74 Ø3 28 FF 64 Ø3 9C Ø3 48 FC 6F 6C 67 72 Ø5
[5L5]    17: B2 FD 2Ø 47 61 Ø3 59 F5 74 2Ø 69 6D 6D 65 72 ØD
[9D5]    18: ØA 2C 62 Ø3 4Ø FB 69 74 2Ø 66 75 Ø3 Ø7 FC 52 61
[SH5]    19: 74 2Ø Ø4 CB FD 54 61 74 Ø3 44 FD 62 65 69 Ø6 86
[NN5]    2Ø: FD 74 72 61 Ø3 EE FF 2E Ø4 79 F1 31 2E 2Ø 4D 65
[6P5]    21: 74 68 6F 64 65 3A ØD ØA 4D 61 Ø3 FC FD 65 74 7A
[RN5]    22: Ø7 85 FD 6E 65 62 Ø3 B6 Ø3 8F FC ØD ØA 67 65 Ø3
[YK5]    23: EB F9 62 74 65 2Ø 57 65 73 Ø3 73 Ø4 36 Ø3 AE FE
[HF5]    24: 61 75 Ø3 Ø3 F7 69 68 6D 2Ø 74 69 65 66 2Ø Ø3 FB
[NM5]    25: FA 64 69 65 2Ø 41 75 Ø4 4C Ø3 E2 F5 6E 6E ØD ØA
[FH5]    26: 6C 61 73 73 65 2Ø 6D Ø3 62 Ø3 FA Ø3 87 FE 45 6E
[YN5]    27: Ø3 7B FA 2Ø 64 75 72 63 68 Ø3 78 FF 5A Ø5 1E FC
[9T5]    28: 2Ø 67 65 68 Ø3 A9 FF 2E Ø3 4F FE 3C 41 Ø5 AØ FF
[6K5]    29: 73 Ø3 D5 F4 53 74 65 6C 6C 65 2Ø 77 65 69 73 74
[ØL5]    3Ø: Ø6 75 FE 42 75 Ø3 6D Ø3 BD F9 69 67 65 2Ø 4C 6F
[KL5]    31: 65 Ø3 13 F8 72 2Ø 61 75 66 3E ØD ØA Ø3 DD FF 2E
[PS5]    32: Ø6 89 FC 6F 2Ø 6C 65 Ø3 Ø8 Ø4 B9 FF 73 Ø4 BD F5
[FG5]    33: ØD ØA 48 61 6E 64 2C 2Ø 67 6C 65 Ø4 6C FF 64 Ø3
[8H5]    34: 1Ø FF 4C Ø3 36 FB 75 6E 67 ØD ØA Ø4 2F F9 72 2Ø
[MC5]    35: 54 61 75 62 65 Ø4 12 Ø4 Ø3 Ø3 CF Ø3 F4 FF 67 Ø3
[WL5]    36: ØE FE 6E 2C Ø5 5A Ø3 E6 FD ØD ØA 72 Ø3 ØD Ø3 81
[1K5]    37: FA 53 63 68 75 6C 74 Ø3 52 Ø3 42 FD ØD ØA 41 Ø3
[YL5]    38: C4 FD 62 65 74 Ø4 A9 FA 57 61 65 68 72 65 Ø4 1F
[VE5]    39: Ø5 3E Ø4 74 FD 41 75 67 Ø3 CØ Ø5 42 FF 69 Ø3 9Ø
[YS5]    4Ø: FE 6E 64 Ø3 AE FE 6E 84 Ø3 ØF F2 6E 2C ØD ØA 62
[ER5]    41: 65 67 69 6E 6E 74 2Ø 6E 75 Ø5 E4 FF 2Ø Ø4 35 F6
[RN5]    42: 2Ø 6C 61 6E 67 73 61 6D ØD ØA Ø4 1E FD 76 6F 72
[XM5]    43: Ø4 A1 F8 74 69 67 2Ø 7A 75 2Ø 77 Ø3 D9 Ø6 C2 FD
[DK5]    44: 69 6D 6D Ø3 B2 Ø3 65 Ø4 84 FC 69 6E 2Ø 52 Ø4 EE
[DP5]    45: Ø5 4A FF 69 Ø3 9B FE 73 2E Ø3 21 F9 22 ØD ØA 3C
[3K5]    46: 44 75 72 Ø6 A3 FE 2Ø 72 Ø3 E6 Ø3 BC FE 4C 94 Ø3
[VK5]    47: 69 Ø4 86 FB 73 ØD ØA 42 75 Ø3 2C FD 69 73 74 Ø5
[PF5]    48: BØ Ø3 7F Ø4 Ø2 F7 6C 7Ø 69 6C 7A ØD ØA 65 69 Ø3
[7M5]    49: 8E FF 64 Ø3 33 Ø3 AD F7 2E 2Ø 44 65 73 68 61 6C
[ZM5]    5Ø: 62 Ø3 AØ FC 6E 64 ØD ØA Ø5 2F FD 65 73 6C Ø3 13
[9M5]    51: Ø3 37 FF 53 Ø4 Ø9 Ø3 BD EB 75 72 2Ø 6E 6F 63 68
[4W4]    52: ØD ØA 46 65 74 7A 65 6E 3E ØD ØA ØD ØA 1A
```

Listing 34. buch13.pak ist eine gepackte Textdatei.

```
[TB5]     1: C1 44 75 2Ø 6C 69 65 73 74 2Ø 64 61 73 2Ø 42 75
[1H5]     2: 63 68 3A ØD ØA 5B 53 69 6D 7Ø 6C 65 2Ø 52 65 69
[SH5]     3: 6D 65 5D ØD ØA ØD ØA 22 44 69 65 2Ø 4B 61 74 7A
[NC5]     4: 65 2C ØD ØA 2Ø 6C 61 67 2Ø 61 75 66 2Ø 64 65 72
[KM5]     5: Ø3 31 FB 4D 61 74 72 61 Ø6 2E FB 75 6E 64 2Ø 6B
[ME5]     6: Ø3 45 FD 74 65 2E Ø4 22 F6 2Ø 4A 65 2Ø 77 65 69
[BE5]     7: 74 65 72 Ø4 3B FB 2Ø 57 75 72 66 Ø4 49 F6 6A 65
[EH5]     8: 2Ø 73 63 68 6E 65 6C 6C Ø4 64 Ø3 ØA FB 5Ø 66 65
[TN5]     9: 72 64 Ø4 6F FF 77 Ø3 28 FE 57 69 Ø3 4E F9 69 6D
[3K5]    1Ø: 2Ø 48 61 61 72 Ø4 89 FC 73 6F 2Ø 77 Ø3 4D FD 65
[KH5]    11: 72 62 Ø6 9B EC 69 73 74 27 73 2Ø 6E 69 63 68 74
[MH5]    12: 2Ø 76 65 72 6B 65 68 72 74 Ø6 57 FE 4F 62 Ø3 B4
[RN5]    13: FE 65 64 Ø4 7C Ø3 96 FA 47 65 6D 75 65 74 Ø4 AD
[RF5]    14: FB 65 73 2Ø 66 75 Ø4 C1 F8 2Ø 6B 65 69 6E 65 2Ø
[EH5]    15: 61 Ø4 A6 Ø4 68 FE 65 67 Ø3 EØ FA 6E 61 63 68 2Ø
[DM5]    16: 73 Ø3 EE FE 2Ø 6F Ø4 67 FA 68 61 62 65 6E 3F Ø5
[SP5]    17: C6 FA 53 6F 2Ø 67 75 74 Ø4 E2 FE 74 75 Ø5 DE F9
[RH5]    18: 77 65 6E 6E 2Ø 6D 61 Ø3 2F FA 69 74 2Ø 4D 75 74
[PR5]    19: Ø4 9E Ø3 B8 FF 2Ø Ø4 EE FD 6E 2Ø 48 Ø5 3A F9 61
[UG5]    2Ø: 75 66 73 65 74 7A Ø3 47 Ø3 25 Ø6 C5 FF 57 Ø3 8E
[BL5]    21: Ø7 76 Ø4 BC F8 67 65 68 6E 2Ø 64 69 65 Ø3 6Ø F7
[ØH5]    22: 4A 61 68 72 65 2C 2Ø 6E 6F Ø4 Ø3 Ø6 68 FE 65 72
[DG5]    23: Ø3 7A ØB 6F FA 2Ø 48 61 61 72 65 Ø3 5D FC 2Ø 55
[6L5]    24: 6E 64 Ø3 3E Ø5 AB FB 65 2Ø 66 6F 72 Ø6 27 F8 61
[PD5]    25: 65 63 68 73 74 2Ø 6E Ø3 4Ø FC 74 73 2Ø 6D Ø3 E8
[EL5]    26: FE 2Ø 64 Ø7 B8 FB 64 61 73 2Ø 69 Ø3 C4 FC 77 6F
[U65]    27: 68 6C Ø6 C6 Ø7 D8 FF 57 Ø4 7E Ø3 A6 FE 22 1A
```

Listing 35. buch14.pak ist eine gepackte Textdatei.

```
[FF5]     1: EØ 44 75 2Ø 6C 69 65 73 74 2Ø 65 69 6E 69 67 65
[KK5]     2: 2Ø 76 6F 65 6C 6C 69 67 ØD ØA 6C 61 6E 67 77 65
[LB5]     3: 69 Ø3 14 F2 65 2Ø 42 81 63 68 65 72 2Ø 75 6E 64
[3G5]     4: 2Ø 44 Ø3 Ø9 E3 65 ØD ØA 77 65 72 74 76 6F 6C 6C
[7H5]     5: 65 2Ø 5A 65 69 74 2Ø 76 65 72 69 6E 6E 74 2E ØD
[PKØ]     6: ØA 1A
```

Listing 36. buch2.pak ist eine gepackte Textdatei.

```
[CC5]     1: E6 44 75 2Ø 6C 69 65 73 74 2Ø 64 61 73 2Ø 42 75
[HF5]     2: 63 68 3A ØD ØA 5B 54 61 67 65 62 Ø3 ØE D8 2Ø 76
[DL5]     3: 6F 6E 2Ø 4D 69 72 69 61 6D 5D ØD ØA ØD ØA 22 31
[YB5]     4: 32 2E 31 3Ø 2E 31 34 33 32 ØD ØA 48 65 75 74 65
[6M5]     5: 2Ø 66 65 69 65 72 Ø3 3D FF 69 Ø3 1B ED 6D 65 69
[RK5]     6: 6E ØD ØA 6E 65 75 6E 7A 65 68 6E 74 65 73 2Ø 47
[7F5]     7: Ø3 18 F8 72 74 73 74 61 67 73 66 Ø3 Ø5 EE 2E ØD
[6D5]     8: ØA 56 61 74 65 72 2Ø 68 61 74 2Ø 6D 69 72 2Ø 64
[BK5]     9: Ø3 Ø4 Ø3 5B Ø8 15 F2 ØD ØA 67 65 73 63 68 65 6E
[TH5]    1Ø: 6B 74 2E 2Ø 49 Ø3 49 EF 62 69 6E 2Ø 67 61 6E 7A
[CL5]    11: 2Ø 73 74 6F 6C 7A ØD ØA 2C Ø4 Ø8 FF 73 Ø5 47 Ø3
[8F5]    12: 91 Ø3 1F Ø3 B9 FB 72 65 69 62 65 Ø3 4F F8 6B 61
[HG5]    13: 6E 6E 2C 2Ø 61 62 Ø3 73 Ø4 B5 FF 68 Ø3 DØ FE 2Ø
[5K5]    14: 61 Ø5 8A Ø3 4D FD 65 2Ø 67 Ø3 3C FC 6E 2Ø 4C 65
[BK5]    15: Ø3 C1 FC 72 2Ø 67 65 Ø3 D9 Ø4 6C FC ØD ØA 31 33
[4P5]    16: ØA 3Ø Ø4 9A FF 68 Ø3 B1 FF 65 Ø8 7D F9 53 63 68
[CN5]    17: 6C 6F 73 73 Ø3 FC FF 45 Ø3 B3 F3 73 74 2Ø 73 6F
[TL5]    18: 2Ø 75 6E 68 65 69 6D 6C Ø3 D5 Ø6 AE FE ØD ØA Ø4
[DE5]    19: D5 F8 6E 61 63 68 74 73 2Ø 6E Ø3 44 FE 74 2Ø Ø3
[CR5]    2Ø: E4 Ø3 BF FD 6C 61 66 Ø8 C6 FA 2E 2Ø 57 65 6E 6E
[2L5]    21: Ø5 D4 FF 6D Ø5 E4 FD 41 75 67 Ø4 5E Ø4 58 Ø3 18
[ZM5]    22: F7 73 65 2C 2Ø 66 81 68 6C 65 Ø5 6C F5 69 6D 6D
[4L5]    23: 65 72 ØD ØA 77 69 65 2Ø Ø3 F3 FF 6D Ø4 8F F5 61
[SD5]    24: 6E 73 74 61 72 72 74 2E 2Ø 53 Ø4 72 F4 ØD ØA 41
[5N5]    25: 55 47 45 4E 2Ø 73 69 6E 64 Ø3 2E FE 6F 2Ø Ø3 7E
[SG5]    26: FC 72 65 63 6B Ø5 38 Ø6 42 Ø6 11 FD 73 69 65 Ø6
[5L5]    27: FC FF 34 ØA Ø3 FC 48 65 75 74 Ø3 15 FD 75 72 66
[LK5]    28: Ø3 F5 Ø4 D4 FD 56 61 74 Ø3 9E FB 61 75 66 ØD ØA
[GE5]    29: Ø3 17 F2 2Ø 4A 61 67 64 2Ø 62 65 67 6C 65 69 74
[LH5]    3Ø: 65 Ø3 65 Ø3 BØ FA 74 ØD ØA 4D 75 74 Ø4 Ø5 F5 74
[6C5]    31: 6F 74 2Ø 69 73 74 2C 2Ø 67 65 Ø3 52 Ø5 Ø3 Ø3 7C
[NL5]    32: F9 65 68 72 2Ø 6F 66 74 Ø4 Ø8 FF 2Ø Ø8 ØE F4 2E
[MN5]    33: 2Ø 49 63 68 ØD ØA 76 65 72 6D 69 Ø7 DA Ø3 C1 FF
[FG5]    34: 2Ø Ø4 44 Ø7 59 FB 67 6C 61 75 62 Ø5 9C FC 61 75
[KD5]    35: 63 68 ØF E1 F2 2Ø 28 61 62 65 6E 64 73 29 ØD ØA
[GD5]    36: 44 69 65 Ø3 66 F6 41 75 67 65 6E 2Ø 6D 61 63 68
[1K5]    37: Ø4 AC F8 69 72 2Ø 41 6E 67 73 74 Ø3 89 FE 45 73
[ML5]    38: Ø4 32 Ø3 6C F8 2C 2Ø 61 6C 73 2Ø 6F 62 Ø5 68 FE
[2M5]    39: 6C 65 Ø3 9B F7 ØD ØA 77 75 65 72 64 65 6E Ø3 CB
[MH5]    4Ø: FF 62 Ø3 82 F6 64 61 73 2Ø 6B 61 6E 6E 2Ø 6E Ø3
[9P5]    41: FF Ø3 25 FD 73 65 69 Ø3 1F FF 44 Ø3 EE FD 73 61
[AN5]    42: 67 Ø9 3B FE 6A 65 Ø3 E3 FB 66 61 6C 6C 73 Ø3 2Ø
[9U5]    43: Ø4 6A FF 65 Ø4 B2 FE 73 6F Ø3 79 FE 65 6D Ø3 FE
[AN5]    44: F9 2Ø 75 6E 64 2Ø 62 6F Ø4 A5 FD 61 75 73 Ø8 CB
[SK5]    45: Ø3 FB Ø3 2Ø FF 6D Ø3 F7 FE 2Ø 64 Ø3 4C F7 67 61
[KT5]    46: 6E 7A 65 2Ø 5A 65 69 Ø3 FA Ø3 3E FF 6C Ø5 BØ FB
[VP5]    47: 2E 2Ø 47 6F 74 Ø3 C7 FE 65 69 Ø3 Ø2 F8 6E 6B ØD
[4R5]    48: ØA 66 61 68 72 Ø3 26 FF 77 Ø3 B7 FD 6D 6F 72 Ø4
[7M5]    49: AB F4 75 6E 73 65 72 65 ØD ØA 54 61 6E 74 Ø4 3C
[2L5]    5Ø: EF 66 2Ø 53 63 68 6C 6F 73 73 2Ø 52 6F 73 65 6E
[PD5]    51: 73 74 Ø3 3Ø FA ØD ØA 62 65 73 75 Ø5 69 FØ ØD ØA
[NE5]    52: ØD ØA 37 2E 31 31 2E 31 34 33 32 ØD ØA 57 Ø3 86
[8H5]    53: FE 73 69 Ø3 35 FE 77 69 Ø3 14 F3 72 2Ø 7A 75 72
[RF5]    54: 81 63 6B 2Ø 69 6D ØD ØA Ø7 A2 F8 2E 2Ø 45 73 2Ø
[EK5]    55: 68 61 74 Ø3 4E Ø3 DØ FE 65 68 Ø3 1Ø FE 67 65 Ø4
[ZP5]    56: 18 FC 65 6E 2C 2Ø Ø3 B6 FE 6F 6E Ø3 DA FE 73 2Ø
[ZG5]    57: Ø4 DA FD 53 6F 68 Ø3 B3 Ø4 2F Ø3 18 Ø5 98 Ø3 FØ
[8M5]    58: Ø4 DØ FD 65 68 74 Ø5 57 Ø3 Ø1 FE 75 74 Ø3 9D FF
[1D5]    59: 73 Ø3 8F F2 64 2Ø 69 73 74 2Ø 66 75 72 63 68 74
[8G5]    6Ø: 62 61 Ø3 ØØ FB 73 74 61 72 6B Ø3 BE F4 49 63 68
[HM5]    61: 2Ø 67 6C 61 75 62 65 2C 2Ø Ø3 25 FF 66 Ø3 D3 FF
[2M5]    62: 65 Ø4 F7 FB 63 68 ØD ØA 61 Ø3 B9 Ø5 37 FE 2Ø 69
[XE5]    63: Ø3 2A FC 72 65 73 73 Ø3 29 Ø4 2D Ø3 57 FE 61 67
[6R5]    64: Ø3 9B FF 6A Ø3 D9 FF 6E Ø4 Ø5 F9 73 2C 2Ø 64 61
[AG5]    65: 73 73 Ø3 6C F6 ØD ØA 69 6E 2Ø 6E 84 63 68 73 Ø3
[5L5]    66: 89 FC 2Ø 5A 65 69 Ø4 4Ø Ø3 7F Ø6 77 FD 6D 61 6C
[BM5]    67: Ø4 ØC Ø3 C5 FF 74 Ø3 5E F3 ØD ØA 38 2E 31 31 2E
[SE5]    68: 31 34 33 32 ØD ØA Ø7 F2 FA 67 65 73 63 68 6E Ø3
[NF5]    69: CØ F2 2C 2Ø 69 6D 2Ø 53 63 68 6C 6F 73 73 ØD ØA
[HH5]    7Ø: Ø4 ED FF 73 Ø3 CA FF 2Ø Ø4 22 Ø3 45 F1 61 6E 67
[SH5]    71: 65 6E 65 68 6D 65 ØD ØA 4B 61 65 6C Ø3 9B Ø3 42
[GT5]    72: FC 67 65 62 72 Ø3 F7 FF 65 Ø3 F9 EE 74 72 6F 74
[2H5]    73: 7A 64 65 6D ØD ØA 77 69 72 2Ø 81 62 65 72 Ø3 A4
[DL5]    74: EF 2Ø 5Ø 65 63 68 66 61 63 6B 65 6C 6E 2Ø 61 6E
[YN5]    75: ØD ØA Ø3 AØ FD 2Ø 57 84 Ø3 72 FF 6E Ø3 D2 FB 66
[ZM5]    76: 65 73 74 69 Ø3 9A FC 6E 2E 2Ø 49 Ø4 CB FC 68 61
[EK5]    77: 62 65 Ø4 A9 F9 2Ø 47 65 66 81 68 6C Ø7 A8 FE 64
[3M5]    78: 69 Ø4 2Ø FF 84 Ø4 26 FD 76 6F 6E Ø4 9Ø FF 73 Ø3
[LK5]    79: 66 FE 6B 61 Ø3 98 Ø3 5B FE 62 6F Ø5 A2 FE 41 75
[WN5]    8Ø: Ø3 1A F8 2Ø 68 65 72 6B 6F 6D 6D Ø6 D9 FF 39 ØA
[YG5]    81: EØ FD 48 65 75 Ø3 99 F2 6E 61 63 68 74 2Ø 69 73
[UG5]    82: 74 2Ø 65 74 77 61 Ø3 Ø5 FF 7Ø Ø3 8D FD 69 65 72
[1H5]    83: Ø3 35 Ø3 E6 Ø6 ØB FE 61 75 Ø4 77 F9 56 61 74 65
[1S5]    84: 72 2Ø 6E Ø3 FA Ø3 E2 FB 72 6B 6C 84 72 Ø5 A4 FE
[ZR5]    85: 6E 6E Ø3 C4 FD 44 69 65 Ø8 B5 Ø3 7C FF 6E Ø5 F8
[CG5]    86: FB ØD ØA 76 65 72 Ø4 63 FB 72 74 2E 2Ø 41 Ø3 47
[WN5]    87: Ø7 Ø9 FD 6E 75 72 Ø6 9Ø Ø5 24 FF 2C Ø5 FD FF 2Ø
[MP5]    88: Ø3 3B FB 2Ø 4D 75 6E 64 Ø5 8A F9 ØD ØA 67 61 6E
[UN5]    89: 7A 65 Ø3 83 Ø4 3Ø FF 74 Ø3 2F FF 65 Ø3 7B FC 6A
[MG5]    9Ø: 65 74 7A Ø3 7C EA 6F ØD ØA 61 75 73 2Ø 61 6C 73
[YB5]    91: 2Ø 77 81 72 64 65 2Ø 65 73 2Ø 6C 65 Ø3 2C Ø3 1D
[HP5]    92: FF 49 Ø3 5E Ø4 2A FF 2Ø Ø6 Ø4 FF 67 Ø3 AØ FF 74
[5M5]    93: Ø4 58 FF 65 Ø3 6D Ø4 8E F5 6D 65 69 6E 65 6D 2Ø
[8P5]    94: 5A 69 6D 6D Ø3 B3 F1 65 6E 74 66 65 72 6E 65 6E
[RG5]    95: ØD ØA 7A 75 2Ø 6C Ø3 EC Ø3 A2 FC 2Ø 45 72 2Ø Ø3
```

```
[4K5]    96: 36 FC 73 7Ø 72 61 Ø3 A8 Ø4 BF FA 6D 69 72 2E 2Ø
[FT5]    97: 4E Ø3 F4 FF 64 Ø3 CB Ø6 BØ Ø3 FC Ø6 4F FE 54 75
[RN5]    98: Ø3 ØC FF 7A Ø3 4D F7 42 69 62 6C 69 74 68 65 6B
[UL5]    99: Ø3 ØE FC 74 ØD ØA 73 Ø6 C8 FA 53 63 68 6C 75 65
[4M5]   1ØØ: Ø3 E5 FF 6C Ø4 C1 FC 66 67 65 73 Ø3 38 FF 6F Ø4
[DW5]   1Ø1: E5 Ø3 AA FC 74 74 65 2C Ø5 AA FE ØD ØA Ø3 79 FE
[EH5]   1Ø2: 2Ø 64 Ø4 4F FB 65 75 74 69 67 Ø3 64 FD 54 61 67
[7N5]   1Ø3: Ø6 2A FE 4C 65 Ø4 4E Ø3 EE FF 62 Ø4 F3 FF 74 Ø3
[2C5]   1Ø4: A4 FØ ØD ØA 32 32 2E 31 31 2E 31 34 33 32 ØD ØA
[ØF5]   1Ø5: 45 73 Ø4 51 FF 2Ø Ø3 48 F9 6F 6E 2Ø 77 69 65 64
[BL5]   1Ø6: Ø5 B3 Ø3 AØ FE 6E 65 Ø4 75 Ø3 63 FE 6E 6F Ø3 6Ø
[AL5]   1Ø7: Ø4 59 FF 2Ø Ø4 5F FØ 5Ø 6F 73 74 2Ø 76 6F 6D ØD
[SH5]   1Ø8: ØA 53 6F 68 6E 2Ø 6D Ø4 31 FF 72 Ø3 6F F3 6E 74
[6K5]   1Ø9: 65 2Ø 62 65 6B 6F 6D 6D 65 6E 3A Ø3 97 FF 72 Ø3
[WM5]   11Ø: A5 FA 6C 6C 2Ø 6D 6F 72 Ø4 6C FF 6E Ø3 84 Ø3 74
[8L5]   111: F9 74 61 67 ØD ØA 68 69 Ø3 DC Ø3 D9 F6 74 72 65
[PH5]   112: 66 66 65 6E 2E 2Ø 49 Ø3 C6 FC 66 72 65 75 Ø3 5C
[9G5]   113: FF 6D Ø4 C5 Ø5 AØ Ø3 88 F9 44 61 73 2Ø 47 65 73
[6N5]   114: Ø3 28 FD 74 2Ø 69 Ø3 CB FF 69 Ø3 E9 FE 72 2Ø Ø4
[ZU5]   115: BB Ø3 B6 FB 61 2Ø 75 6E 64 Ø5 C4 EF 66 81 68 6C
[ZD5]   116: 65 2C 2Ø 64 61 73 73 2Ø 65 73 ØD ØA 6A Ø3 A8 Ø8
[LN5]   117: D7 F9 42 65 77 65 67 75 6E Ø3 FB F6 ØD ØA 76 65
[7K5]   118: 72 66 6F 6C 67 74 Ø6 1A FC 7A 69 65 68 Ø3 6E Ø3
[4R5]   119: 57 FE ØD ØA Ø6 AØ FF 6E Ø5 3B FA 6D 65 68 72 2Ø
[PG5]   12Ø: 69 Ø7 D6 Ø3 DØ FF 5A Ø6 44 FE 75 6D Ø4 6Ø Ø3 Ø3
[VK5]   121: Ø3 66 Ø7 96 FB 62 65 6F 62 65 Ø3 ØØ FF 74 Ø3 FC
[KS5]   122: FD 6B 61 6E Ø3 19 FE 45 73 Ø4 3F F8 ØD ØA 61 6C
[WH5]   123: 73 2Ø 6F 62 Ø4 C9 FD 6E 75 72 Ø3 C3 F5 72 61 75
[BG5]   124: 66 2Ø 77 61 72 74 65 74 Ø3 31 FF 2E Ø4 ØB F9 2Ø
[3P5]   125: 22 ØD ØA 3C 48 69 Ø3 BD FD 65 6E 64 Ø3 DC F6 64
[FL5]   126: 69 65 2Ø 45 69 6E 74 72 61 Ø6 7B FC 3E ØD ØA 1A
```

Listing 37. buch3.pak ist eine gepackte Textdatei.

```
[DC5]     1: D7 44 75 2Ø 6C 69 65 73 74 2Ø 64 61 73 2Ø 42 75
[BF5]     2: 63 68 3A ØD ØA 5B 4D 61 67 69 73 63 68 65 2Ø 57
[VB5]     3: 65 6C 74 2Ø 42 61 6E 64 2Ø 35 Ø3 11 Ø9 15 FA 54
[YC5]     4: 6F 72 65 2Ø 75 Ø3 25 F1 42 61 72 72 69 65 72 65
[LM5]     5: 6E 5D ØD ØA ØD ØA 22 Ø4 2C FD 65 2Ø 69 Ø3 Ø6 FF
[9L5]     6: 68 Ø3 42 FE 2Ø 64 Ø3 5Ø FC 4D 61 74 65 Ø3 41 FD
[9H5]     7: 2C ØD ØA Ø4 5C EE 64 65 6D 2Ø 6C 65 68 72 73 61
[CP5]     8: 6D 65 6E 2Ø 45 6C 65 76 Ø3 79 FE 69 6E Ø5 68 FF
[CH5]     9: 73 Ø3 6F F8 57 65 72 6B 65 2Ø 6E 61 Ø3 32 F2 67
[YE5]    1Ø: 65 62 72 61 63 68 74 ØD ØA 77 65 72 64 Ø3 8Ø F2
[5G5]    11: 73 6F 6C 6C 2E 2Ø 46 72 65 69 6C 69 63 68 Ø5 52
[8K5]    12: E9 65 73 ØD ØA 6E 75 72 2Ø 65 69 6E 65 2Ø 45 69
[CF5]    13: 6E 66 81 68 72 75 6E 67 Ø3 82 Ø4 5B Ø3 A1 Ø3 3A
[7L5]    14: FF 65 Ø5 75 Ø6 1D Ø3 EØ Ø3 5F Ø3 4F F6 3B ØD ØA
[ML5]    15: 75 6D 2Ø 77 69 72 6B Ø4 B4 Ø3 E6 Ø5 DE FA 2Ø 7A
[VL5]    16: 75 ØD ØA 76 Ø3 AB FB 62 72 69 6E 67 Ø4 A7 FF 69
[9M5]    17: Ø3 3B FC 76 69 65 6C Ø3 DA FD 4A 61 68 Ø3 37 F9
[UE5]    18: 41 75 73 62 69 6C 64 Ø4 D1 FD 64 75 72 Ø3 B6 Ø4
[NN5]    19: C5 Ø3 84 FC 77 65 69 73 Ø3 1A FF 47 Ø3 24 FD 68
[DG5]    2Ø: 72 74 Ø3 4B F7 6E 6F 74 77 65 6E 64 69 67 Ø4 67
[SM5]    21: Ø3 55 Ø5 87 FC 73 2Ø 6B 6C Ø5 C5 FC 42 81 63 68
[5K5]    22: Ø4 7Ø Ø3 BF FD 69 65 2Ø Ø3 E1 FD 65 74 7A Ø3 65
[KR5]    23: EE 6B 61 6E 6E 2E 2Ø 44 65 6E 6E 6F 63 68 ØD ØA
[ØM5]    24: 67 69 62 Ø4 BB Ø4 3F FB 69 67 65 2Ø 47 Ø3 DØ F7
[XE5]    25: 64 62 65 67 72 69 66 66 65 Ø4 61 Ø4 81 Ø3 D5 F8
[2M5]    26: 74 61 6C 65 6E 74 69 65 Ø3 53 FE 72 2Ø Ø3 AD FC
[YP5]    27: ØD ØA 69 6E Ø3 CC FD 65 73 73 Ø7 C9 FC 53 63 68
[YG5]    28: 75 Ø3 4F FF 72 Ø3 1C Ø4 98 F7 6D 69 74 2Ø 48 69
[SF5]    29: 6C 66 65 Ø8 67 F3 42 75 63 68 65 73 ØD ØA 73 65
[MK5]    3Ø: 6C 62 73 Ø3 9F FF 72 Ø3 E9 Ø3 42 Ø6 8C F7 ØD ØA
[RG5]    31: 5A 75 6E 61 65 63 68 Ø5 1Ø FD 6B 84 72 Ø3 19 FD
[PN5]    32: 77 69 72 Ø7 A3 FE ØD ØA Ø3 E4 FB 6C 61 67 77 6F
[SP5]    33: Ø3 DF F6 3A ØD ØA ØD ØA 4D 61 67 69 73 Ø3 Ø6 FB
[1F5]    34: 2Ø 42 61 72 72 Ø3 DD Ø4 4B FF 4E Ø3 94 FC 74 2Ø
[BU5]    35: 6D 61 Ø3 7D F8 76 94 6C 6C 69 67 2Ø 6E Ø3 EE FF
[ZM5]    36: 74 Ø5 EC FC 74 62 61 72 Ø3 3E FC 48 69 6E 64 Ø3
[PB5]    37: 16 F6 69 73 73 65 2E 2Ø 45 69 6E 65 Ø3 6A Ø6 53
[ØL5]    38: FE ØD ØA Ø8 5A FF 2Ø Ø3 9E F7 81 74 7A 74 2Ø 61
[3K5]    39: 6C 73 6F Ø4 38 FF 65 Ø3 6D Ø4 51 Ø3 EA FF 6F Ø3
[FM5]    4Ø: 8B DD 2Ø 47 65 67 65 6E 73 74 61 6E 64 2Ø 76 6F
[KE5]    41: 72 ØD ØA 64 65 6D 2Ø 5A 75 67 72 69 66 66 2Ø 64
[DG5]    42: 75 72 63 68 2Ø Ø3 D4 Ø3 A9 F9 ØD ØA 4C 65 62 65
[1V5]    43: 77 Ø3 FF FF 6E ØD C7 FF 84 Ø3 FØ Ø3 2Ø FB 46 61
[GH5]    44: 6C 6C 73 Ø4 B9 FF 2Ø Ø7 C1 F3 62 65 69 73 7Ø 69
[FK5]    45: 65 6C 73 2D ØD ØA 77 Ø3 26 FE 65 2Ø Ø3 1B Ø5 BA
[NG5]    46: FE 2Ø 5A Ø3 31 FB 6B 61 6D 7Ø 66 Ø3 6E FØ 65 72
[CG5]    47: 77 69 63 6B 65 6C 74 2Ø 77 69 72 64 2C 2Ø Ø3 B7
[ME5]    48: FA 62 61 75 74 ØD ØA Ø3 22 Ø4 3A Ø9 99 FF 2Ø Ø8
[6M5]    49: A4 F9 ØD ØA 75 6D 2Ø 73 69 Ø3 EC FD 68 65 72 Ø3
[EN5]    5Ø: 8Ø FB 61 75 66 2Ø 75 Ø3 D5 FD 6B 61 6E Ø3 BE FD
[TK5]    51: 66 6F 72 Ø3 D3 FE 2Ø 6E Ø3 84 F9 74 2Ø 6D 65 68
[UH5]    52: 72 2Ø Ø3 4B FC 6C 65 74 7A Ø3 62 F1 77 65 72 64
[RD5]    53: 65 6E 2E 2Ø 45 73 2Ø 73 65 69 2Ø Ø3 BB FF 6E Ø3
[WL5]    54: 6C Ø3 99 FC 62 65 6E 75 Ø3 B3 Ø5 19 Ø8 6D FB 73
[GE5]    55: 2Ø 53 63 68 Ø3 B8 Ø3 B5 FC 28 73 69 65 Ø5 73 Ø3
```

Listing 38. buch4.pak ist eine gepackte Textdatei.

```
[2H5]    56: 93 FE 34 3A Ø5 1D Ø4 E1 E8 ØD ØA 57 61 66 66 65
[VA5]    57: 6E 29 2E 2Ø 46 81 72 2Ø 64 61 73 2Ø 41 75 66 68
[SU5]    58: 65 Ø3 D1 FE ØD ØA Ø8 DD FF 72 Ø9 75 FC 6E 2Ø 69
[6H5]    59: 73 Ø3 ED Ø3 5C F1 76 69 65 6C 2Ø 5A 61 75 62 65
[SK5]    6Ø: 72 6B 72 61 66 Ø3 3A F6 6E 6F 74 77 65 6E 64 69
[WK5]    61: 67 2C Ø3 15 FF E1 Ø3 C6 F6 72 6A 65 6E 69 67 65
[LK5]    62: 2C ØD ØA Ø3 61 FD 2Ø 64 69 Ø3 E4 Ø3 AE FE 73 75
[3K5]    63: Ø3 A5 FF 2C Ø3 A8 Ø5 38 Ø6 A3 FC 61 6D 2Ø 4C Ø4
[2L5]    64: 1E F5 2Ø 62 6C 65 69 62 74 2E ØD ØA 4E Ø4 88 F6
[5K5]    65: 64 65 73 74 6F 74 72 6F 74 7A Ø4 37 FD 2Ø 65 73
[4M5]    66: Ø3 22 FD 94 67 6C Ø3 AØ FE 2Ø 75 Ø3 F9 FA 6D 69
[PK5]    67: 72 2Ø 73 69 Ø3 CØ Ø3 D9 Ø3 67 F2 ØD ØA 46 84 6C
[YM5]    68: 6C 65 2Ø 62 65 6B 61 6E 6E Ø3 7D FE 69 6E Ø3 6Ø
[WM5]    69: FF 6E Ø4 2Ø FB 65 72 66 61 68 Ø3 34 FF 65 Ø5 FE
[ST5]    7Ø: Ø7 6E Ø4 B3 FD 55 6E 74 Ø4 ED FE 6E 67 Ø3 93 FF
[LR5]    71: 81 Ø3 48 FA 6C 65 62 74 65 6E Ø3 9C FD 4D 61 6E
[ØP5]    72: Ø3 29 FE 2Ø 6D Ø3 F8 FF 73 Ø4 22 FC 42 61 72 72
[2M5]    73: Ø3 FA Ø4 E9 FD 67 65 6C Ø3 19 F9 2Ø 6E 75 72 2Ø
[AR5]    74: 66 75 Ø3 FB FF 62 Ø3 A5 FC 69 6D 6D 74 Ø3 D1 Ø4
[GL5]    75: 9Ø FF 77 Ø3 ØØ FD 6E 2Ø 6F Ø4 6D FE 47 65 Ø3 ØE
[VL5]    76: FA 73 74 84 6E 64 65 Ø3 1C FD 56 69 65 Ø3 D7 Ø9
[FE5]    77: 26 F7 57 61 66 66 65 2Ø 77 65 72 Ø3 E6 FE ØD ØA
[BM5]    78: Ø3 E3 FD 73 6F 6C ØF 2B FD 61 75 66 Ø3 57 Ø3 FØ
[GM5]    79: FF 74 Ø3 6F Ø4 1D Ø7 7A F7 54 6F 72 65 3A ØD ØA
[NG5]    8Ø: 4E 65 Ø3 DE Ø3 77 Ø3 A1 F8 73 69 63 68 74 62 61
[DP5]    81: 72 Ø3 7F Ø4 6B FF 2C Ø4 FD FD 2Ø 69 6D Ø3 C1 E9
[WL5]    82: 6F 72 6D 61 6C 66 61 6C 6C 2Ø 75 6E 64 75 72 63
[DP5]    83: 68 64 72 69 6E 67 6C Ø3 CF Ø4 CC FA 6E 64 2E 2Ø
[PP5]    84: 57 65 Ø3 DA Ø5 C7 FE 2Ø 68 Ø3 9D FE 2Ø Ø5 8A F9
[HG5]    85: 65 6E 74 73 7Ø 72 65 Ø3 B8 Ø3 ØC FA 6E 2Ø 5A 61
[6L5]    86: 75 62 Ø3 17 FD 61 6E 2C Ø3 91 FB ØD ØA 76 65 72
[3N5]    87: Ø4 D9 FD 72 74 2Ø Ø4 CE Ø5 DE FC 4D 61 74 65 Ø3
[ZH5]    88: 9C FE ØD ØA Ø4 6Ø FB 57 61 6E 64 2Ø Ø3 8F Ø4 59
[4K5]    89: FD 41 72 74 Ø3 DD FC 61 E1 ØD ØA Ø4 11 Ø5 F4 Ø3
[SK5]    9Ø: 46 FF 65 Ø3 14 Ø6 F3 FD 2Ø 67 65 Ø3 27 FB ØD ØA
[5K5]    91: 6B 61 6E Ø3 36 FB 77 6F 62 65 69 Ø5 7C F9 74 72
[2K5]    92: 6F 74 7A 64 65 Ø3 E4 FD 77 65 69 Ø3 52 Ø3 81 Ø5
[RH5]    93: 46 Ø4 D2 F5 2Ø 62 6C 65 69 62 74 2E ØD ØA 49 Ø3
[PG5]    94: 75 F6 69 65 73 65 6D 2Ø 5A 75 73 74 Ø4 5E FA 73
[XH5]    95: 74 65 6C 6C 74 Ø3 57 Ø3 9E Ø5 5D F6 65 69 6E 65
[6D5]    96: 2Ø 54 75 65 72 65 Ø3 61 Ø3 1D Ø4 EB Ø3 E6 Ø4 F1
[UP5]    97: F8 44 69 6D 65 6E 73 69 6F Ø3 C8 FD 61 72 2C Ø6
[1H5]    98: DF F9 6D 65 69 73 74 2Ø 76 Ø3 ØC F9 73 65 68 72
[AK5]    99: 2Ø 6D 84 Ø3 B7 FE 69 67 Ø4 8D FC 5A 61 75 62 Ø3
[ZC5]    1ØØ: Ø1 ED 72 2Ø 65 72 73 63 68 61 66 66 65 6E 2Ø 77
[XN5]    1Ø1: 75 72 64 65 6E Ø3 C4 FF 28 Ø3 9D FC 68 65 2Ø 42
[1L5]    1Ø2: Ø4 E6 F9 33 3A 2Ø 4D 61 67 69 Ø3 3E FD 65 ØD ØA
[XM5]    1Ø3: Ø9 Ø5 FD 65 6E 29 Ø3 4D FF 44 Ø3 39 FE 41 75 Ø3
[GM5]    1Ø4: 43 FB 74 68 61 6C 74 Ø3 F4 Ø5 E9 Ø3 A8 FD 73 6F
[HK5]    1Ø5: 6C Ø3 64 FD 6E 2Ø 6D Ø7 6Ø FA 6E 2Ø 54 6F 72 2Ø
[ZL5]    1Ø6: Ø3 1B Ø3 F7 FD 78 74 72 Ø3 CE F5 67 65 66 84 68
[XG5]    1Ø7: 72 6C 69 63 68 2E Ø3 5E Ø3 3Ø Ø3 91 FD 6C 74 65
[BN5]    1Ø8: Ø3 ØE FF 68 Ø3 79 F9 6E 75 72 2Ø 73 6F 2Ø Ø3 9D
[XR5]    1Ø9: F9 6E 65 6C 6C ØD ØA 77 Ø4 16 FD 6F 65 67 Ø4 B9
[NL5]    11Ø: FE 2Ø 75 Ø3 59 Ø3 D5 FB 67 65 72 61 64 Ø3 66 ØD
[UK5]    111: E1 FD 64 75 72 Ø3 Ø7 Ø3 8A FE ØD ØA Ø8 99 FF 73
[1R5]    112: Ø5 A2 FD 68 69 6E Ø5 ØA FF 67 Ø3 53 Ø4 4C FE 4D
[LH5]    113: 65 Ø3 A7 Ø3 6C FC 2Ø 76 65 72 Ø3 19 FB 6C 69 65
[UH5]    114: 73 73 Ø3 AØ FF 73 Ø3 Ø6 FD ØD ØA 64 Ø3 FE Ø3 1F
[4T5]    115: FC 65 2Ø 6E 61 Ø6 ØD FB 6D 61 6C 69 67 Ø4 8D FF
[BT5]    116: 44 Ø9 27 FF 2Ø Ø3 FD FF 64 Ø3 79 FE 76 6F Ø4 C1
[4H5]    117: FB 65 6C 62 73 74 Ø3 3Ø FE 57 65 Ø4 93 FB 2Ø 5A
[LL5]    118: 61 75 62 Ø3 3D FE 7Ø 81 Ø4 8D Ø3 EE FC ØD ØA 55
[MK5]    119: 6E Ø4 37 FF 69 Ø3 42 FA 6E 2Ø 66 81 72 2Ø Ø3 5Ø
[7M5]    12Ø: FF 73 Ø3 FA F3 62 65 6E 94 74 69 67 74 2Ø 77 65
[WH5]    121: 72 64 Ø4 AC Ø3 24 FD 65 74 2Ø Ø3 78 Ø3 7F E9 74
[5F5]    122: 72 65 62 73 61 6D 65 2Ø 4C 65 68 72 6C 69 6E 67
[BG5]    123: 2Ø 69 6E 2Ø 42 61 Ø3 EF FE 37 3A Ø3 31 Ø7 16 F9
[DK5]    124: 2Ø 53 7Ø 72 75 63 68 Ø3 DB F6 6D 6C 75 6E 67 2E
[AB1]    125: 22 ØD ØA 1A
```

Listing 38. buch4.pak ist eine gepackte Textdatei.

```
[DC5]     1: D7 44 75 2Ø 6C 69 65 73 74 2Ø 64 61 73 2Ø 42 75
[BF5]     2: 63 68 3A ØD ØA 5B 4D 61 67 69 73 63 68 65 2Ø 57
[WB5]     3: 65 6C 74 2Ø 42 61 6E 64 2Ø 34 Ø3 11 ØA 15 F5 61
[7N5]     4: 66 66 65 6E 5D ØD ØA ØD ØA 22 Ø4 2C FD 65 2Ø 69
[9G5]     5: Ø3 Ø6 FA 68 69 65 72 2Ø 64 Ø3 44 F6 4D 61 74 65
[1L5]     6: 72 69 65 2C ØD ØA Ø4 5Ø EE 64 65 6D 2Ø 6C 65 68
[1M5]     7: 72 73 61 6D 65 6E 2Ø 45 6C 65 76 Ø3 6D FE 69 6E
[SG5]     8: Ø5 5C FF 73 Ø3 63 F8 57 65 72 6B 65 2Ø 6E 61 Ø3
[XC5]     9: 32 F2 67 65 62 72 61 63 68 74 ØD ØA 77 65 72 64
[WK5]    1Ø: Ø3 74 F2 73 6F 6C 6C 2E 2Ø 46 72 65 69 6C 69 63
[DG5]    11: 68 Ø5 46 E9 65 73 ØD ØA 6E 75 72 2Ø 65 69 6E 65
[GD5]    12: 2Ø 45 69 6E 66 81 68 72 75 6E 67 Ø3 76 Ø4 4F Ø3
[ØF5]    13: 95 FC 75 6E 64 65 Ø5 69 Ø6 1D Ø3 D4 Ø3 53 Ø3 43
[2K5]    14: F6 3B ØD ØA 75 6D 2Ø 77 69 72 6B Ø4 A8 Ø3 DA Ø5
[LL5]    15: D2 FA 2Ø 7A 75 ØD ØA 76 Ø3 9F FB 62 72 69 6E 67
[GM5]    16: Ø4 9B FF 69 Ø3 25 FC 76 69 65 6C Ø3 CE F3 4A 61
[YD5]    17: 68 72 65 2Ø 41 75 73 62 69 6C 64 Ø4 C5 FD 64 75
[MM5]    18: 72 Ø3 AA Ø4 B9 Ø3 78 FC 77 65 69 73 Ø3 ØE FF 47
[5L5]    19: Ø3 18 FD 68 72 74 Ø3 3F F7 6E 6F 74 77 65 6E 64
[TR5]    2Ø: 69 67 Ø4 5B Ø3 49 Ø5 7B FC 73 2Ø 6B 6C Ø5 B9 FC
[ØG5]    21: 42 81 63 68 Ø4 64 Ø3 B3 FD 69 65 2Ø Ø3 D5 FD 65
[8R5]    22: 74 7A Ø3 59 EE 6B 61 6E 6E 2E 2Ø 44 65 6E 6E 6F
```

```
[9H5]    23: 63 68 ØD ØA 67 69 62 Ø4 AF Ø4 33 FB 69 67 65 2Ø
[YE5]    24: 47 Ø3 C4 F7 64 62 65 67 72 69 66 66 65 Ø4 55 Ø4
[LL5]    25: 75 Ø3 C9 F8 74 61 6C 65 6E 74 69 65 Ø3 47 FE 72
[1S5]    26: 2Ø Ø3 A1 FC ØD ØA 69 6E Ø3 CØ FD 65 73 73 Ø7 BD
[AL5]    27: FC 53 63 68 75 Ø3 43 FF 72 Ø3 1Ø Ø4 8C F7 6D 69
[YE5]    28: 74 2Ø 48 69 6C 66 65 Ø8 5B F3 42 75 63 68 65 73
[SG5]    29: ØD ØA 73 65 6C 62 73 Ø3 93 FF 72 Ø3 DD Ø3 36 Ø6
[SH5]    3Ø: 8Ø F7 ØD ØA ØD ØA 4D 61 67 69 73 Ø3 FA FD 2Ø 57
[BP5]    31: 61 Ø3 A9 FE 6E 3A Ø3 FE F3 69 6E 64 2Ø 69 6D 2Ø
[FE5]    32: 61 6C 6C 67 65 6D Ø4 65 Ø5 71 FC 63 68 74 2Ø Ø3
[XG5]    33: 6E Ø3 42 F5 ØD ØA 68 65 72 7A 75 73 74 65 6C Ø3
[6D5]    34: B9 F7 2E 2Ø 4D 61 6E 2Ø 62 72 61 Ø3 F9 Ø3 4B Ø4
[7L5]    35: 39 FF 2Ø Ø3 4E FA 6B 6F 65 6D 6D 6C Ø3 48 Ø7 2Ø
[3P5]    36: Ø3 AC FF 77 Ø7 BØ Ø3 D8 FF 77 Ø3 D2 FB 2Ø 6F 64
[BD5]    37: 65 72 Ø7 67 F7 53 74 72 65 69 74 61 78 74 Ø4 C3
[LP5]    38: FE 2Ø 62 Ø3 DC FF 67 Ø3 66 Ø5 F1 FF 2Ø Ø4 E7 Ø4
[YF5]    39: 9A F7 6D 2Ø 65 6E 74 73 7Ø 72 65 Ø3 78 FE 6E 64
[CK5]    4Ø: Ø4 3C FA 5A 61 75 62 65 72 Ø3 5A FE 65 69 Ø3 Ø4
[CS5]    41: Ø3 8F Ø3 DØ Ø3 AC FE 73 6F Ø3 CF FF 72 Ø3 FD Ø3
[GH5]    42: A1 FC 63 68 2Ø 76 Ø3 4F FF 69 Ø3 9Ø Ø8 2Ø Ø3 E9
[EM5]    43: FC 6E 75 74 7A Ø3 B2 FB 2C 64 61 2Ø 73 Ø3 85 Ø3
[TH5]    44: 3Ø FA 4B 61 6D 7Ø 66 2Ø Ø5 4Ø FD ØD ØA 7A Ø3 FØ
[PP5]    45: FE 74 6F Ø4 9Ø Ø7 E3 Ø3 71 FE 6E 6E Ø3 58 FA ØD
[VR5]    46: ØA 4A 65 64 6F Ø3 F8 Ø4 DF Ø4 B5 FC 2Ø 6E 75 72
[1K5]    47: Ø6 BE F4 ØD ØA 73 65 68 72 2Ø 67 72 6F 73 73 Ø3
[DM5]    48: C2 Ø4 A9 F9 65 72 66 61 68 72 65 Ø5 5E Ø6 D5 F8
[WK5]    49: 65 72 2Ø 6D 6F 65 67 6C Ø3 27 Ø3 45 FB 4D 61 67
[RM5]    5Ø: 69 73 Ø3 CC Ø8 Ø4 FE 73 69 Ø3 71 FE 64 61 Ø3 F2
[DK5]    51: Ø3 5C EE 7A 69 67 65 2Ø 4D 69 74 74 65 6C 2Ø 75
[LS5]    52: 6D 2Ø 67 65 67 Ø4 D1 FE 6D 61 Ø3 CB FF 74 Ø5 B2
[SC5]    53: Ø3 95 Ø3 84 F8 2Ø 7A 75 ØD ØA 6B 61 65 Ø3 22 Ø3
[5D5]    54: 43 FA 2Ø 44 61 62 65 69 Ø5 4E Ø3 55 F8 65 67 61
[ZP5]    55: 6C ØD ØA 69 6E Ø3 36 FF 6C Ø3 99 FC 72 2Ø 45 72
[WG5]    56: Ø4 98 F7 69 6E 75 6E 67 73 66 6F 72 Ø3 7D Ø4 53
[UP5]    57: ØA CF FA 76 6F 72 68 61 6E Ø4 3A Ø4 A4 Ø3 91 FB
[PK5]    58: 44 69 65 2Ø 4C Ø4 C1 FE 64 65 Ø3 73 F9 7A 61 65
[RH5]    59: 68 6C 74 2C Ø4 A8 F5 73 2Ø 61 75 66 ØD ØA 53 63
[HK5]    6Ø: 68 6C Ø3 6A Ø3 1C F9 73 6B 61 76 61 6E 2Ø Ø3 ØA
[ØN5]    61: FD 6D 61 6C Ø4 66 FC ØD ØA 67 72 Ø3 5A Ø3 Ø2 Ø9
[7H5]    62: 8Ø FF 64 Ø3 81 Ø3 Ø7 FC 77 61 72 7A Ø4 C3 Ø5 1D
[AG5]    63: Ø4 6D Ø5 1C Ø4 Ø7 FE 73 2Ø Ø3 56 F6 77 65 72 74
[BP5]    64: ØD ØA 68 69 6E 74 Ø3 85 Ø3 99 FE 65 6D Ø9 9C F9
[UH5]    65: 6D 2Ø 54 6F 72 ØD ØA Ø3 B7 FB 67 65 6D 61 75 Ø3
[9Z5]    66: AB FE 2Ø 68 Ø3 E8 FE 6E 2C Ø3 FC FF 69 Ø3 6C FE
[2F5]    67: 73 2Ø Ø3 17 Ø5 CA F5 7A 69 67 65 2Ø 57 61 66 66
[VF5]    68: 65 2Ø Ø3 8B Ø4 E6 FD 2Ø 49 68 Ø3 9Ø FE 75 65 Ø3
[TL5]    69: 7E FF 77 Ø3 3Ø Ø3 2C FD 6B 6F 6E Ø3 B2 F9 2E 2Ø
[SL5]    7Ø: 28 73 69 65 68 Ø3 E9 FF 42 Ø3 29 FD 2Ø 35 2C Ø9
[CP5]    71: BC Ø4 C6 FB 65 2Ø 75 6E 64 Ø4 21 FD 72 72 69 Ø3
[TK5]    72: 7F FB 6E 29 2E ØD ØA Ø6 7B FB 73 7Ø 72 75 65 Ø4
[TF5]    73: 3Ø FF 68 Ø3 42 FB 7A 75 2Ø 69 6D Ø7 21 FC 37 3A
[SP5]    74: 2Ø 44 Ø3 FF FF 6D Ø7 2C FD ØD ØA 53 Ø3 52 EF 63
[1R5]    75: 68 73 61 6D 6D 6C 75 6E 67 2E 22 ØD ØA ØD ØA 1A
```

Listing 39. buch5.pak ist eine gepackte Textdatei.

```
[DC5]     1: D7 44 75 2Ø 6C 69 65 73 74 2Ø 64 61 73 2Ø 42 75
[BF5]     2: 63 68 3A ØD ØA 5B 4D 61 67 69 73 63 68 65 2Ø 57
[NB5]     3: 65 6C 74 2Ø 42 61 6E 64 2Ø 33 Ø3 11 Ø9 15 EF 44
[SN5]     4: 69 6D 65 6E 73 69 6F 6E 65 6E 5D ØD ØA ØD ØA 22
[RE5]     5: Ø4 2C FD 65 2Ø 69 Ø3 Ø6 FA 68 69 65 72 2Ø 64 Ø3
[LG5]     6: 49 F6 4D 61 74 65 72 69 65 2C ØD ØA Ø4 55 F6 64
[MG5]     7: 65 6D 2Ø 6C 65 68 72 73 61 Ø3 37 FB 2Ø 45 6C 65
[DL5]     8: 76 Ø3 72 FE 69 6E Ø5 61 FF 73 Ø3 68 F8 57 65 72
[NE5]     9: 6B 65 2Ø 6E 61 Ø3 32 F2 67 65 62 72 61 63 68 74
[DK5]    1Ø: ØD ØA 77 65 72 64 Ø3 79 F2 73 6F 6C 6C 2E 2Ø 46
[1K5]    11: 72 65 69 6C 69 63 68 Ø5 4B E9 65 73 ØD ØA 6E 75
[LG5]    12: 72 2Ø 65 69 6E 65 2Ø 45 69 6E 66 81 68 72 75 6E
[BK5]    13: 67 Ø3 7B Ø4 54 Ø3 9A FC 75 6E 64 65 Ø5 6E Ø6 1D
[5K5]    14: Ø3 D9 Ø3 58 Ø3 48 F6 3B ØD ØA 75 6D 2Ø 77 69 72
[9S5]    15: 6B Ø4 AD Ø3 DF Ø5 D7 FA 2Ø 7A 75 ØD ØA 76 Ø3 A4
[SM5]    16: FB 62 72 69 6E 67 Ø4 AØ FF 69 Ø3 25 FC 76 69 65
[3K5]    17: 6C Ø3 D3 F3 4A 61 68 72 65 2Ø 41 75 73 62 69 6C
[6R5]    18: 64 Ø4 CA FD 64 75 72 Ø3 AF Ø4 BE Ø3 7D FC 77 65
[RM5]    19: 69 73 Ø3 13 FF 47 Ø3 1D FD 68 72 74 Ø3 44 F7 6E
[6K5]    2Ø: 6F 74 77 65 6E 64 69 67 Ø4 6Ø Ø3 4E Ø5 8Ø FC 73
[CK5]    21: 2Ø 6B 6C Ø5 BE FC 42 81 63 68 Ø4 69 Ø3 B8 FD 69
[KT5]    22: 65 2Ø Ø3 DA FD 65 74 7A Ø3 5E EE 6B 61 6E 6E 2E
[6L5]    23: 2Ø 44 65 6E 6E 6F 63 68 ØD ØA 67 69 62 Ø4 B4 Ø4
[5F5]    24: 38 FB 69 67 65 2Ø 47 Ø3 C9 F7 64 62 65 67 72 69
[3H5]    25: 66 66 65 Ø4 5A Ø4 7A Ø3 CE F8 74 61 6C 65 6E 74
[BL5]    26: 69 65 Ø3 4C FE 72 2Ø Ø3 A6 FC ØD ØA 69 6E Ø3 C5
[EH5]    27: FD 65 73 73 Ø7 C2 FC 53 63 68 75 Ø3 48 FF 72 Ø3
[ØH5]    28: 15 Ø4 91 F7 6D 69 74 2Ø 48 69 6C 66 65 Ø8 6Ø F3
[VG5]    29: 42 75 63 68 65 73 ØD ØA 73 65 6C 62 73 Ø3 98 FF
[YL5]    3Ø: 72 Ø3 E2 Ø3 3B Ø6 85 F7 ØD ØA ØD ØA 4D 61 67 69
[FR5]    31: 73 Ø3 FF F7 2Ø 44 69 6D 65 6E 73 69 6F Ø3 11 FF
[DM5]    32: 3A Ø3 Ø3 FA 69 6E 64 2Ø 7A 75 Ø3 C8 FF 73 Ø3 D1
[2H5]    33: F4 72 2Ø 57 65 6C 74 ØD ØA 65 78 69 73 Ø4 C1 Ø3
[HK5]    34: 55 F7 65 ØD ØA 7Ø 61 72 61 6C 6C Ø3 E1 Ø5 44 FF
[7M5]    35: 65 Ø4 89 Ø4 F7 Ø3 EA FF 75 Ø3 D3 Ø3 12 FF 6E Ø3
[WK5]    36: E7 F5 74 2Ø 73 65 68 72 2Ø 67 72 6F E1 Ø3 7F FC
[FF5]    37: 69 6E ØD ØA Ø3 C9 F9 2Ø 62 65 73 74 65 68 Ø3 77
```

```
[LG5]    38: F6 6F 66 74 2Ø 6E 75 72 2Ø 61 75 Ø3 Ø2 Ø3 8B EF
[NK5]    39: 65 6D 2Ø 5A 69 6D 6D 65 72 2E 2Ø 53 6F 6C 63 68
[MC5]    4Ø: 2Ø Ø4 AA Ø3 7Ø 11 1F FØ 2Ø 77 69 72 64 ØD ØA 68
[6H5]    41: 61 65 75 66 69 67 2Ø 76 Ø3 D4 Ø6 AA Ø4 1E FE 65
[1N5]    42: 72 Ø3 A8 FF 72 Ø3 C8 F2 61 66 66 65 6E 2C 2Ø 75
[CG5]    43: 6D 2Ø 65 74 77 61 Ø3 A7 F6 61 75 66 7A 75 62 65
[RD5]    44: 77 61 68 Ø3 52 FE 2C 2Ø Ø3 Ø8 Ø3 79 FD 65 6D 61
[ØF5]    45: Ø3 91 Ø4 8C FA 64 69 65 2Ø 48 84 Ø3 54 FE 2Ø 66
[YP5]    46: Ø4 5C FB 6E 2Ø 73 6F 6C Ø3 65 Ø3 1A FD 65 69 6C
[1L5]    47: Ø3 49 FD 73 2Ø 64 Ø9 EC Ø3 89 FA 6C 62 73 74 2Ø
[AP5]    48: 53 Ø3 FA FE 64 65 Ø3 29 F6 6B 6F 65 6E 6E 74 65
[BG5]    49: 2E 2Ø 45 Ø3 Ø7 Ø4 E8 F6 2Ø 6D 61 65 63 68 74 69
[5H5]    5Ø: 67 65 Ø3 ØA FE 5A 61 Ø3 11 F9 72 69 6E 73 74 72
[WS5]    51: 75 Ø3 CF FC 74 2Ø 6F 64 Ø3 56 Ø6 BE FF 57 Ø4 FC
[ØV2]    52: F9 2E 22 ØD ØA ØD ØA 1A
```

Listing 40. buch6.pak ist eine gepackte Textdatei.

```
[DC5]     1: D7 44 75 2Ø 6C 69 65 73 74 2Ø 64 61 73 2Ø 42 75
[BF5]     2: 63 68 3A ØD ØA 5B 4D 61 67 69 73 63 68 65 2Ø 57
[LB5]     3: 65 6C 74 2Ø 42 61 6E 64 2Ø 31 Ø3 11 Ø8 15 FF 72
[DP5]     4: Ø4 22 F9 6E 5D ØD ØA ØD ØA 22 Ø4 2C FD 65 2Ø 69
[7M5]     5: Ø3 Ø6 FE 68 69 Ø3 33 FF 64 Ø3 43 F6 4D 61 74 65
[PN5]     6: 72 69 65 2C ØD ØA Ø4 4F EE 64 65 6D 2Ø 6C 65 68
[YL5]     7: 72 73 61 6D 65 6E 2Ø 45 6C 65 76 Ø3 6C FE 69 6E
[MG5]     8: Ø5 5B FF 73 Ø3 62 F8 57 65 72 6B 65 2Ø 6E 61 Ø3
[9E5]     9: 1B F2 67 65 62 72 61 63 68 74 ØD ØA 77 65 72 64
[UK5]    1Ø: Ø3 73 F2 73 6F 6C 6C 2E 2Ø 46 72 65 69 6C 69 63
[CG5]    11: 68 Ø5 45 E9 65 73 ØD ØA 6E 75 72 2Ø 65 69 6E 65
[CD5]    12: 2Ø 45 69 6E 66 81 68 72 75 6E 67 Ø3 75 Ø4 4E Ø3
[FE5]    13: 94 FC 75 6E 64 65 Ø5 68 Ø6 1D Ø3 D3 Ø3 52 Ø3 42
[RH5]    14: F6 3B ØD ØA 75 6D 2Ø 77 69 72 6B Ø4 A7 Ø3 D9 Ø5
[EL5]    15: D1 FA 2Ø 7A 75 ØD ØA 76 Ø3 9E FB 62 72 69 6E 67
[BM5]    16: Ø4 9A FF 69 Ø3 25 FC 76 69 65 6C Ø3 CD F3 4A 61
[VD5]    17: 68 72 65 2Ø 41 75 73 62 69 6C 64 Ø4 C4 FD 64 75
[6M5]    18: 72 Ø3 A9 Ø4 B8 Ø3 77 FC 77 65 69 73 Ø3 ØD FF 47
[ZK5]    19: Ø3 17 FD 68 72 74 Ø3 3E F7 6E 6F 74 77 65 6E 64
[ER5]    2Ø: 69 67 Ø4 5A Ø3 48 Ø5 7A FC 73 2Ø 6B 6C Ø5 B8 FC
[SF5]    21: 42 81 63 68 Ø4 63 Ø3 B2 FD 69 65 2Ø Ø3 D4 FD 65
[4R5]    22: 74 7A Ø3 58 EE 6B 61 6E 6E 2E 2Ø 44 65 6E 6E 6F
[5H5]    23: 63 68 ØD ØA 67 69 62 Ø4 AE Ø4 32 FB 69 67 65 2Ø
[WE5]    24: 47 Ø3 C3 F7 64 62 65 67 72 69 66 66 65 Ø4 54 Ø4
[DL5]    25: 74 Ø3 C8 F8 74 61 6C 65 6E 74 69 65 Ø3 46 FE 72
[7T5]    26: 2Ø Ø3 AØ FC ØD ØA 69 6E Ø3 BF FD 65 73 73 Ø7 BC
[SL5]    27: FC 53 63 68 75 Ø3 42 FF 72 Ø3 ØF Ø4 8B F7 6D 69
[VE5]    28: 74 2Ø 48 69 6C 66 65 Ø8 5A F3 42 75 63 68 65 73
[HG5]    29: ØD ØA 73 65 6C 62 73 Ø3 92 FF 72 Ø3 DC Ø3 35 Ø6
[HK5]    3Ø: 7F F7 ØD ØA ØD ØA 4D 61 67 69 73 Ø3 F9 FD 72 2Ø
[HK5]    31: 42 Ø3 1Ø FC 3A ØD ØA 44 Ø3 1F FF 6D Ø7 19 Ø5 21
[AN5]    32: FF 2Ø Ø3 ØB FE 6E 74 Ø3 2C Ø3 7Ø Ø4 AE FE 42 65
[YN5]    33: Ø3 B2 F8 66 6C 75 73 73 75 6E 67 Ø4 B1 Ø4 FB FB
[LN5]    34: 4C 65 62 65 77 Ø3 F2 FA 6E 73 2Ø 44 69 6E Ø3 9B
[PU5]    35: F9 7A 75 2Ø 74 75 6E 2C Ø4 EF FE ØD ØA Ø3 F4 FE
[8G5]    36: 61 75 Ø4 95 F1 67 65 6E 65 6D 2Ø 41 6E 74 72 69
[PD5]    37: 65 62 2Ø 6E Ø4 79 Ø3 73 F8 2Ø 77 75 65 72 64 65
[ZK5]    38: 2E Ø3 69 Ø3 7D FF 69 Ø4 Ø3 Ø3 B3 FC 73 65 68 72
[5N5]    39: Ø3 E4 F9 61 63 68 74 76 6F 6C Ø4 DC F7 5A 61 75
[3K5]    4Ø: 62 65 72 73 7Ø 72 Ø3 F8 FC 2Ø 75 6E 64 Ø3 FD Ø3
[DN5]    41: BE FE 74 65 Ø4 93 FA 2Ø 61 6E 2Ø 55 6E Ø3 31 FD
[4N5]    42: 75 6C 64 Ø4 85 FC 2C ØD ØA 77 Ø3 DE ED 62 65 73
[3E5]    43: 7Ø 69 65 6C 73 77 65 69 73 65 2Ø 56 65 72 66 75
[YG5]    44: Ø3 B3 Ø3 54 Ø3 7B Ø3 59 FD 72 2Ø 4A Ø3 ØD FC 66
[GK5]    45: 72 61 75 Ø4 AD FF 67 Ø3 64 FD 74 7A 74 Ø3 F1 Ø4
[DM5]    46: 9F FE 6E 2E ØA C3 EB 81 63 68 65 2Ø 68 69 65 72
[HG5]    47: 7A 75 ØD ØA 66 69 6E 64 65 74 2Ø 64 Ø3 15 FD 67
[8K5]    48: 65 6C Ø3 ØA Ø3 EC FB 2Ø 53 63 68 81 Ø3 CØ F7 ØD
[EK5]    49: ØA 42 61 6E 64 2Ø 37 3A Ø4 A4 FB 2Ø 6D 61 67 69
[7P5]    5Ø: Ø3 E6 Ø3 96 FF 53 Ø5 CB FB 73 61 6D 6D 6C Ø3 19
[Y12]    51: FB 2E 22 ØD ØA 1A
```

Listing 41. buch7.pak ist eine gepackte Textdatei.

```
[DC5]     1: D7 44 75 2Ø 6C 69 65 73 74 2Ø 64 61 73 2Ø 42 75
[BF5]     2: 63 68 3A ØD ØA 5B 4D 61 67 69 73 63 68 65 2Ø 57
[4D5]     3: 65 6C 74 2Ø 42 61 6E 64 2Ø 32 Ø3 11 Ø9 15 F3 4D
[ØM5]     4: 6F 6E 73 74 65 72 5D ØD ØA ØD ØA 22 Ø4 2C FD 65
[2D5]     5: 2Ø 69 Ø3 Ø6 FA 68 69 65 72 2Ø 64 Ø3 45 FE 4D 61
[6M5]     6: Ø3 39 FB 69 65 2C ØD ØA Ø4 51 EE 64 65 6D 2Ø 6C
[WK5]     7: 65 68 72 73 61 6D 65 6E 2Ø 45 6C 65 76 Ø3 6E FE
[CM5]     8: 69 6E Ø5 5D FF 73 Ø3 64 F8 57 65 72 6B 65 2Ø 6E
[WD5]     9: 61 Ø3 32 F2 67 65 62 72 61 63 68 74 ØD ØA 77 65
[TH5]    1Ø: 72 64 Ø3 75 F2 73 6F 6C 6C 2E 2Ø 46 72 65 69 6C
[6F5]    11: 69 63 68 Ø5 47 E9 65 73 ØD ØA 6E 75 72 2Ø 65 69
[4G5]    12: 6E 65 2Ø 45 69 6E 66 81 68 72 75 6E 67 Ø3 77 Ø4
[GG5]    13: 5Ø Ø3 96 FC 75 6E 64 65 Ø5 6A Ø6 1D Ø3 D5 Ø3 54
[ZL5]    14: Ø3 44 F6 3B ØD ØA 75 6D 2Ø 77 69 72 6B Ø4 A9 Ø3
```

```
[VR5]    15: DB Ø5 D3 FA 2Ø 7A 75 ØD ØA 76 Ø3 AØ FB 62 72 69
[7S5]    16: 6E 67 Ø4 9C FF 69 Ø3 25 FC 76 69 65 6C Ø3 CF F3
[VK5]    17: 4A 61 68 72 65 2Ø 41 75 73 62 69 6C 64 Ø4 C6 FD
[NH5]    18: 64 75 72 Ø3 AB Ø4 BA Ø3 79 FC 77 65 69 73 Ø3 ØF
[3P5]    19: FF 47 Ø3 19 FD 68 72 74 Ø3 4Ø F7 6E 6F 74 77 65
[7K5]    2Ø: 6E 64 69 67 Ø4 5C Ø3 4A Ø5 7C FC 73 2Ø 6B 6C Ø5
[FH5]    21: BA FC 42 81 63 68 Ø4 65 Ø3 B4 FD 69 65 2Ø Ø3 D6
[XS5]    22: FD 65 74 7A Ø3 5A EE 6B 61 6E 6E 2E 2Ø 44 65 6E
[AL5]    23: 6E 6F 63 68 ØD ØA 67 69 62 Ø4 BØ Ø4 34 FB 69 67
[FC5]    24: 65 2Ø 47 Ø3 C5 F7 64 62 65 67 72 69 66 66 65 Ø4
[MF5]    25: 56 Ø4 76 Ø3 CA F8 74 61 6C 65 6E 74 69 65 Ø3 48
[NL5]    26: FE 72 2Ø Ø3 A2 FC ØD ØA 69 6E Ø3 C1 FD 65 73 73
[EH5]    27: Ø7 BE FC 53 63 68 75 Ø3 44 FF 72 Ø3 11 Ø4 8D F7
[ZF5]    28: 6D 69 74 2Ø 48 69 6C 66 65 Ø8 5C F3 42 75 63 68
[PH5]    29: 65 73 ØD ØA 73 65 6C 62 73 Ø3 94 FF 72 Ø3 DE Ø3
[4N5]    3Ø: 37 Ø6 81 FB ØD ØA ØD ØA 45 Ø4 67 FF 65 Ø3 9C Ø3
[8P5]    31: 1C F8 41 72 6D 65 65 2Ø 61 75 Ø3 FE FC 4D 6F 6E
[PP5]    32: 73 Ø3 D5 FC 6E 2Ø 7A 75 Ø4 78 FD 63 68 61 Ø3 AA
[XN5]    33: F9 6E 2Ø 69 73 74 ØD ØA Ø5 66 FE 4B 75 Ø3 33 FE
[7M5]    34: 2Ø 64 Ø4 DF FB 63 68 77 61 72 Ø3 7E Ø3 2F FC 61
[HD5]    35: 67 69 65 Ø4 86 F6 73 68 61 6C 62 2Ø 77 69 72 64
[XN5]    36: Ø7 F1 Ø3 FF F8 63 68 6C 65 63 68 74 68 Ø3 B5 F8
[XH5]    37: 76 65 72 77 65 72 66 6C Ø3 E3 FC 65 ØD ØA 55 Ø4
[HH5]    38: CB FD 66 61 6E Ø3 21 F9 2Ø 6E 75 72 2Ø 76 6F Ø3
[NK5]    39: 64 F6 67 65 77 69 73 73 65 6E 6C 6F Ø3 B7 FD 2Ø
[SN5]    4Ø: 6D 61 Ø4 88 FC 75 6E 67 72 Ø4 2Ø FE ØD ØA Ø3 D9
[ZN5]    41: Ø4 5F FF 6D Ø4 68 Ø3 37 FB 62 65 67 6F 6E Ø3 ØD
[GL5]    42: FD 2E 2Ø 41 Ø3 CD Ø5 7B EA 72 2Ø 53 74 65 6C 6C
[EH5]    43: 65 2Ø 73 6F 6C 6C 74 65 2Ø 6A 65 64 6F 63 68 Ø3
[KK5]    44: 4B F9 72 77 84 68 6E 74 2Ø Ø3 92 F8 64 65 6E 2C
[AL5]    45: 2Ø 64 61 73 Ø3 8Ø Ø3 ED ØB 3C Ø3 BD F9 4B 72 65
[PP5]    46: 61 74 75 72 Ø4 CC FE 7A 75 Ø3 57 FE 6E 2Ø Ø3 9D
[7M5]    47: FE 6F 74 Ø3 4Ø FE 7A 84 Ø3 85 F4 6E 2Ø 75 6E 64
[WE5]    48: ØD ØA 61 6E 2Ø 81 62 Ø5 96 FD 6E 2Ø 57 Ø5 29 Ø5
[ZN5]    49: A8 FB 77 65 6E 69 67 Ø5 CE FF 61 Ø4 3F FB 65 72
[4G5]    5Ø: 6C 65 69 Ø3 79 EA 2E 2Ø 2Ø 57 69 72 6B 73 61 6D
[ØH5]    51: 65 72 ØD ØA 69 73 74 2Ø 65 69 6E 65 Ø3 BF FE 67
[KK5]    52: 69 Ø3 26 FF 65 Ø6 62 F9 2E ØD ØA 28 53 69 65 Ø3
[EP5]    53: A2 F6 42 61 6E 64 2Ø 34 3A ØD ØA 4D ØD 9D F9 6E
[VZ1]    54: 29 2E 22 ØD ØA 1A
```

Listing 42. buch8.pak ist eine gepackte Textdatei.

```
[DC5]     1: D7 44 75 2Ø 6C 69 65 73 74 2Ø 64 61 73 2Ø 42 75
[BF5]     2: 63 68 3A ØD ØA 5B 4D 61 67 69 73 63 68 65 2Ø 57
[YD5]     3: 65 6C 74 2Ø 42 61 6E 64 2Ø 36 Ø3 11 DE 54 65 6C
[6E5]     4: 65 7Ø 6F 72 74 61 74 69 6F 6E 2Ø 66 75 65 72 2Ø
[GF5]     5: 41 6E 66 61 65 6E 67 65 72 5D ØD ØA ØD ØA 22 Ø4
[1K5]     6: 15 FD 65 2Ø 69 Ø3 Ø6 FE 68 69 Ø3 3C FF 64 Ø3 51
[2K5]     7: F6 4D 61 74 65 72 69 65 2C ØD ØA Ø4 5D EE 64 65
[RF5]     8: 6D 2Ø 6C 65 68 72 73 61 6D 65 6E 2Ø 45 6C 65 76
[7S5]     9: Ø3 7A FE 69 6E Ø5 69 FF 73 Ø3 7Ø F8 57 65 72 6B
[AF5]    1Ø: 65 2Ø 6E 61 Ø3 1B F2 67 65 62 72 61 63 68 74 ØD
[NH5]    11: ØA 77 65 72 64 Ø3 81 F2 73 6F 6C 6C 2E 2Ø 46 72
[4H5]    12: 65 69 6C 69 63 68 Ø5 53 E9 65 73 ØD ØA 6E 75 72
[LG5]    13: 2Ø 65 69 6E 65 2Ø 45 69 6E 66 81 68 72 75 6E 67
[2F5]    14: Ø3 83 Ø4 5C Ø3 A2 FC 75 6E 64 65 Ø5 76 Ø6 1D Ø3
[UF5]    15: E1 Ø3 6Ø Ø3 5Ø F6 3B ØD ØA 75 6D 2Ø 77 69 72 6B
[XP5]    16: Ø4 B5 Ø3 E7 Ø5 DF FA 2Ø 7A 75 ØD ØA 76 Ø3 AC FD
[CM5]    17: 62 72 69 Ø3 44 Ø3 A9 FF 69 Ø3 25 FE 76 69 Ø3 2D
[4E5]    18: F1 ØD ØA 4A 61 68 72 65 2Ø 41 75 73 62 69 6C 64
[PH5]    19: Ø4 D2 FD 64 75 72 Ø3 B7 Ø4 C6 Ø3 85 FC 77 65 69
[6P5]    2Ø: 73 Ø3 1B FF 47 Ø3 25 FD 68 72 74 Ø3 4C F7 6E 6F
[KE5]    21: 74 77 65 6E 64 69 67 Ø4 68 Ø3 56 Ø5 88 FC 73 2Ø
[GK5]    22: 6B 6C Ø5 C6 FC 42 81 63 68 Ø4 71 Ø3 CØ FD 69 65
[GL5]    23: 2Ø Ø3 E2 FD 65 74 7A Ø3 66 EE 6B 61 6E 6E 2E 2Ø
[RH5]    24: 44 65 6E 6E 6F 63 68 ØD ØA 67 69 62 Ø4 BC Ø4 4Ø
[YF5]    25: FB 69 67 65 2Ø 47 Ø3 D1 F7 64 62 65 67 72 69 66
[6G5]    26: 66 65 Ø4 62 Ø4 82 Ø3 D6 F8 74 61 6C 65 6E 74 69
[6V5]    27: 65 Ø3 54 FE 72 2Ø Ø3 AE FC ØD ØA 69 6E Ø3 CD FD
[3H5]    28: 65 73 73 Ø7 CA FC 53 63 68 75 Ø3 5Ø FF 72 Ø3 1D
[6L5]    29: Ø4 99 F7 6D 69 74 2Ø 48 69 6C 66 65 Ø8 68 F3 42
[7F5]    3Ø: 75 63 68 65 73 ØD ØA 73 65 6C 62 73 Ø3 AØ FF 72
[5L5]    31: Ø3 EA Ø3 43 Ø6 8D FB ØD ØA ØD ØA 54 Ø3 E9 F6 7Ø
[NH5]    32: 6F 72 74 61 74 69 6F 6E 3A Ø3 7F Ø3 95 FC 74 2Ø
[UD5]    33: 6D 61 Ø5 67 F6 ØD ØA 42 65 66 6F 65 72 64 65 Ø3
[ØM5]    34: AD FF 67 Ø4 BF Ø4 Ø9 F6 47 65 67 65 6E 73 74 61
[PH5]    35: 6E 64 Ø3 Ø2 FF 6F Ø3 4B FA 2Ø 4C 65 62 65 77 Ø3
[ZD5]    36: ØØ FF 6E Ø3 56 F6 64 75 72 63 68 2Ø 52 61 75 6D
[9G5]    37: Ø4 DØ F6 2Ø 5A 65 69 74 2C 2Ø 6F 68 6E Ø3 42 Ø4
[GK5]    38: FE FD 2Ø 61 62 Ø3 68 F7 7A 75 2Ø 62 65 6E 6F 65
[8C5]    39: 74 Ø3 A8 Ø4 2Ø F8 56 65 72 73 63 68 77 69 Ø3 61
[1H5]    4Ø: F5 74 2Ø 61 6C 73 6F 2Ø 65 74 77 61 Ø3 95 FF 6E
[YH5]    41: Ø3 9Ø Ø3 99 Ø4 52 F2 6E 2Ø 53 74 65 6C 6C 65 2C
[VM5]    42: 2Ø 73 6F ØD ØA Ø5 AD Ø3 CC Ø3 12 FB 73 2Ø 67 6C
[3Z5]    43: 65 Ø3 EF FF 7A Ø3 87 FE 69 67 Ø3 DC Ø3 CD FF 6D
[GP5]    44: Ø3 C3 Ø3 C8 Ø3 CF FB 5Ø 6C 61 74 7A Ø3 A9 FD 5A
[YG5]    45: 61 75 Ø3 98 FC 73 7Ø 72 81 Ø3 E1 F9 2Ø 68 69 65
[KG5]    46: 72 7A 75 Ø3 33 FE 28 42 Ø3 FF F8 2Ø 37 3A 2Ø 4D
[LC5]    47: 61 67 69 Ø4 EØ F3 ØD ØA 53 7Ø 72 75 63 68 73 61
[184]    48: 6D 6D 6C Ø3 4E FA 29 2E 22 ØD ØA 1A
```

Listing 43. buch9.pak ist eine gepackte Textdatei.

Kommunikation und Parsing

Dialog mit dem PC

In nahezu allen Rollenspielen und Adventures unterhalten Sie sich auf einfache Weise mit den Lebewesen, die Ihnen dort begegnen. Da unsere Sprache jedoch sehr komplex und kontextabhängig ist, ist eine sinnvolle Kommunikation zwischen Rechner und Mensch nicht ganz einfach zu bewerkstelligen. Dennoch erlauben die hier vorgestellten Grundlagen und Mechanismen die Programmierung eines Kommunikationsparsers.

Wenn Sie in Zukunft über ein Terminal eine Bestellung aufgeben, könnte es sein, daß es sich bei Ihrem Gesprächspartner nicht um ein Lebewesen, sondern um einen Computer handelt. Auch am Telefon oder am Geldautomaten werden Sie sich wohl mit einer künstlichen Person unterhalten können. Die Forschungen der letzten Jahre waren zunehmend auf Spracherkennung und Kommunikation ausgerichtet. Wenn Sie als Spieler in Adventures oder Rollenspielen auf Lebewesen treffen, befinden Sie sich in einer ähnlichen Lage: Sie sprechen mit künstlichen Lebewesen in Ihrem Rechner. Während der Computer in den übrigen Beispielen nur Dienstleistungen anbieten soll, ist eine Kommunikation im Spiel weitaus mehr. Sie soll die Geisteshaltung, das Wissen und den Willen von Individuen repräsentieren. Dabei soll auch noch Atmosphäre entstehen, und Informationen oder Gegenstände sollen ausgetauscht werden. Ein gutes Beispiel dafür ist der Unterschied bei der Simulation zwischen der Kommunikation mit einem Geldautomaten und einem arabischen Händler. Sie unterscheiden sich durch Motivationen, Aktionen und Reaktionen. Wenn Sie mit einem Geldautomaten sprechen, wird sich folgendes Gespräch nie entwickeln:

Geldautomat: Frische Hunderter! Hier gibt's die besten.
Kunde: Ich brauche 250 Mark.
Geldautomat: Nimm doch 300 Mark, die Hunderter sind gerade frisch reingekommen!
Kunde: Ich brauch' aber nur 250!
Geldautomat: Was kannst Du mit 250 Mark schon anfangen? Die sind sowieso bald wieder futsch! Wenigstens 290 Mark solltest Du abheben!
Kunde: Ok, 260.
Geldautomat: 280!
Kunde: 270 Mark ist mein letztes Wort!
Geldautomat: Wenn das so ist, bekommst Du keinen Pfennig!
Kunde: Also gut, gib mir 280 Mark!
Geldautomat: Wunderbar, hier sind sie.

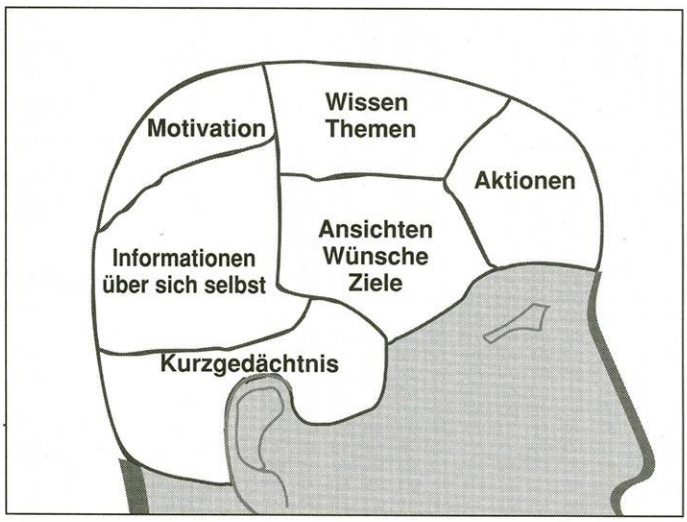


Bild 1. Das Gehirn einer künstlichen Person hat verschiedene interessante Bereiche.

Bisher werden solche Kommunikationsabläufe – bei Geldautomaten und Spielen – meist nach dem Multiple-choice-Verfahren erledigt. Dabei geht die Natürlichkeit einer Unterhaltung verloren, und der Eindruck, vom lebenden Gesprächspartner tatsächlich verstanden zu werden, entsteht auch nicht. Dies ist aber gerade für die zukünftigen computergestützten Dienstleistungen, für Adventures und Rollenspiele wichtig. Im folgenden lernen Sie Grundlagen, Problemstellungen, Prinzipien und Mechanismen kennen, die eine Basis für eine solche Kommunikation bilden.

Verstehen und verstanden werden

Ziel ist es, ein Programm zu entwickeln, das eine Diskussion über verschiedene Themengebiete erlaubt und dabei bestimmte Absichten verfolgt oder erkennt und den Austausch von Informationen, Aktionen oder Gegenständen gestattet und dabei noch die Persönlichkeitsstruktur eines bestimmten Individuums repräsentiert.
Die beiden Hauptprobleme bei einer Mensch-Maschine-Kommunikation sind:
– Wie versteht der Computer, was der Mensch meint?
– Wie versteht der Mensch, was der Computer meint?
Es gibt vielerlei Ansätze, diese vereinfacht formulierten Probleme zu lösen. Die klassische Methode zur Lösung des ersten Problems ist das sogenannte Parsing. Programme, die Wörter und Sätze einer Sprache aufschlüsseln und auf Korrektheit überprüfen, heißen Parser. Jeder Compiler oder Interpreter einer Programmiersprache enthält einen Parser im Umfang der entsprechenden Sprache. Die Anzahl der Kommandos einer modernen Programmiersprache ist im Vergleich zum Wortumfang der englischen oder deutschen Sprache jedoch sehr gering. Für Rollenspiele und die Geldautomaten der Zukunft bedeutet dies: Was passiert bei einem Parser, falls nun ein Kunde oder Spieler eine Frage stellt?

Ein Parser für den gesamten Wortumfang der deutschen Sprache wäre wohl dementsprechend groß. Es würde damit aber erstens sehr lange dauern, auch nur einen Satz zu parsen und zu überprüfen, und zweitens ist damit noch keine entsprechende Antwort auf die gestellte Frage gefunden, ganz zu schweigen von der Vermittlung von Natürlichkeit, Geisteshaltung des Sprechers und so weiter. Im Englischen hat man es da schon leichter, da sowohl Grammatik als auch der Wortgebrauch einfacher sind. Dennoch ist der Aufwand extrem und das Ergebnis nicht berauschend.

Deshalb sind gute englische oder gar deutsche Parser in Computerspielen sehr rar gesät. Sie sehen, ein Parser ist zwar notwendig, aber er löst noch lan-

ge nicht das Problem des Verstehens. Im Normalfall gibt es einen Unterschied zwischen dem, was wir sagen, und dem, was wir meinen. Wenn Sie diese Worte lesen, die hier stehen, und wissen, was damit gemeint ist, dann »verstehen« Sie diesen Satz. Um einen Algorithmus zu finden, damit Ihr Computer Sie »versteht«, müssen Sie von einer ganz anderen Sichtweise ausgehen.

Das Gehirn einer künstlichen Person

Stellen Sie sich erst einmal folgende Frage: Wer will was von wem? Im obigen Beispiel trifft ein Kunde auf einen Geldautomaten. Der Kunde will Geld, der Automat will welches hergeben. Dieses Beispiel zeigt schon einmal zwei grundlegende Gesprächssituationen auf. Einer will etwas haben, der andere will etwas geben. Dabei ist es egal, ob es sich um Geld, Informationen oder etwas anderes handelt. Immer werden Waren getauscht oder geschenkt. Jedes Gespräch ist also im allgemeinen ein Austausch. Falls dabei keine Waren getauscht werden, dann zumindest Informationen. Um einen solchen Austausch auf dem Computer in natürlicher Sprache zu simulieren, ist mehr notwendig als ein bloßes Parsing der eingegebenen Wörter.

Nehmen Sie an, Ihr Computer ist kein Computer, sondern ein Mensch. Und zwar ein kleiner Mensch: ein Kind, das noch nicht alles versteht und noch nicht alles weiß. Für dieses Kind entwickeln Sie jetzt einen Aktionsplan für eine Kommunikation. Dieser Plan stellt natürlich nur eine primitive Simulation dar, aber er genügt schon, um obige Unterhaltung zu realisieren. Als Grundlage dient das Gehirn einer künstlichen Person.

Zunächst erfahren Sie, wie ein Gehirn einer künstlichen Person aussehen kann, und nach welchen Prinzipien es arbeitet. Das künstliche Gehirn ist aus folgenden Teilen aufgebaut (Bild 1):

1. Motivation: Zu welchen Aktionen will die Person animieren, und was sind ihre Absichten?
2. Im Kurzzeitgedächtnis werden das Wissen über das Gespräch und die Ansichten des Gesprächspartners aufbewahrt. Das Kurzzeitgedächtnis ist dabei so programmiert, daß es sich Themengebiete in begrenztem Umfang merkt und nach einigen Sätzen über ein anderes Thema immer noch das ursprüngliche Thema kennt.
3. Die persönlichen Ansichten, allgemeinen Ziele und Wünsche
4. Informationen über die eigene Person (Name, Alter und ähnliches)
5. Wissen und Informationen zu Themen, die dieser künstlichen Person bekannt sind
6. Aktionen, die diese künstliche Person ausführen kann, falls sie danach gefragt wird

Dabei stellen Sie sicher fest, daß alle Teile des Gehirns sehr allgemein gehalten sind, so daß Lebewesen jedweder Art mit diesem künstlichen Gehirn ausgestattet werden können.

Nachdem die künstliche Person nun Gestalt angenommen hat, sollte sie auch verstehen und sich ausdrücken lernen. Hierzu wenden Sie sich ein zweites Mal von der Computerwelt ab und gehen in die Psychologie. Stellen Sie sich vor, wie ein kleines Kind mit Ihnen sprechen würde.

Simuliertes Denken und Gesprächsführung

Wir gehen hier von der These aus, daß der Mensch nicht in Wörtern oder grammatikalischen Regeln denkt, sondern diese nur Ausdruck und Hilfsmittel seiner Gedanken sind. Damit ist auch nicht die Erkennung von Grammatik der wesentliche Bestandteil der Gespächsführung unseres Programms, vielmehr ist es die Erkennung von Absichten und Gedankenabläufen. Deshalb wird davon ausgegangen, daß der Mensch einen Satz wie einen Behälter für Informationen benutzt. Nicht die Wörter und Grammatik sind entscheidend, sondern die Satzhülsen. So werden im minimalen Kurzzeitgedächtnis Satzhülsen gespeichert, und auch die Antworten sind als vollständig formulierte Sätze vorhanden. Gerade diese Formulierungen unterstützen die Natürlichkeit der Sätze des Kommunikationsprogramms. Abgesehen davon, daß verschiedene Gespräche nicht gleich verlaufen, sind die Gesprächssituation und der Kontext entscheidend für die gegebenen Antworten und demnach auch der Schlüssel zu den gesuchten Algorithmen.

Legen Sie also zunächst die Situationen fest, in denen sich das Gehirn befinden kann, und überlegen Sie dann, wie es auf eine Veränderung der Situation reagiert. Dabei gehen Sie davon aus, wieviel das Programm vom eingegebenen Satz versteht und welchen Hintergrund es schon besitzt.

- **Kein Gesprächsthema vorhanden**

1. Situation: Kein Wort wird verstanden.
2. Situation: Allgemeine Schlüsselwörter werden erkannt.
3. Situation: Ein Gesprächsthema wird erkannt.

- **Gesprächsthema vorhanden**

1. Situation: Kein Wort wird verstanden.
2. Situation: Allgemeine Schlüsselwörter werden erkannt.
3. Situation: Neues Gesprächsthema wird erkannt.
4. Situation: Frage oder Aussagesätze werden erkannt.
5. Situation: Absichten werden erkannt.

Jeder dieser Situationen wird nun ein Verhaltensmuster zugewiesen, das mit der jeweiligen Situation fertig wird. Falls Sie nun den Satz eingeben, durchläuft er die Erkennungsmechanismen von Situation 5 bis 1 beziehungsweise 3 bis 1 und reagiert entsprechend dem Verhaltensmuster der Stufe. Je höher die Situationsnummer bei einer Erkennung ausfällt, desto exakter kann unser Kommunikationsprogramm reagieren.

Nun legen Sie das Verhaltensmuster fest.

- **1. Verhaltensmuster für Situation 1**

Ausgangspunkt ist der »worst case«, also der schlimmste Fall. Das bedeutet, das Kommunikationsprogramm befindet sich in Situation 1 und versteht nicht ein Wort seines Gesprächspartners, der im folgenden »Benutzer« genannt wird. In diesem Fall gibt es genau zwei Alternativen, die auf den eingegebenen Satz zutreffen:

A. Der Gesprächspartner hat – absichtlich oder wegen Rechtschreibfehlern – tatsächlich etwas völlig Unverständliches eingegeben. Da das Kommunikationsprogramm so gut sein sollte, daß es nahezu alle sinnvollen Eingaben oberhalb dieser Situationsnummer bearbeitet, gibt es in diesem Fall zu, daß es nichts versteht, und fragt nach Erklärungen. Falls diese Situation mehrmals hintereinander eintritt, muß davon ausgegangen werden, daß der Benutzer absichtlich sinnlose Sätze eingibt oder er Analphabet ist. In diesen Fällen bricht das Programm von sich aus die Konversation ab.

B. Der Gesprächspartner spricht über ein so spezielles Thema, daß das Programm es nicht kennt. An dieser Stelle versucht er durch Nachfragen, das Gespräch zu einem bekannten Thema zu wenden.

- **2. Verhaltensmuster für Situation 2**

Oberhalb dieser ersten Stufe liegt die Ein-Wort-Erkennung. Das heißt, in einem Satz werden Schlüsselwörter erkannt, die keinen direkten Bezug zu einem Thema haben, aber von allgemeiner Bedeutung sind. Diese Phase ähnelt der Phase eines kleinen Kindes, das gerade sprechen und verstehen lernt und alles wissen will: das Fragealter, oder auch »Mama, warum ist der Himmel blau?«.

In diesem Fall muß aus den erkannten Schlüsselwörtern ein Fragesatz formuliert werden, um den Benutzer zu einer klareren oder anderen Aussage zu bewegen, die in einer Stufe oberhalb dieser, also bei den bekannten Themen, abge-

handelt werden kann. Diese Kommunikationsstufe versteht zwar etwas, sieht aber noch keinen Sinn oder Zusammenhänge. Durch Nachfragen und so tun, als ob es den Benutzer versteht, hat diese Stufe die zwei folgenden wichtigen Begleiterscheinungen: Zum einen gibt sie die Gelegenheit, die Unwissenheit des Programms durch Fragen und kontextabhängige Aussagen zu verschleiern. Weiterhin können an dieser Stelle Sätze eingeflochten werden, die den Benutzer zu bestimmten Aussagen animieren. So fühlt er sich einerseits verstanden und wird andererseits behutsam an ein Thema hingeführt. Für den ersten Teil dieser Stufe bietet sich ein Programm an, das auf der Basis von »Eliza« arbeitet. »Eliza« ist die 1966 von J. Weizenbaum entwickelte Simulation eines zuhörenden Pyschoanalytikers, die dem Benutzer durch Umstellung seiner eigenen Antworten in Fragesätze vorgaukelt, es verstehe ihn. Für den zweiten Teil, der Einflechtung von Animation oder Motivationssätzen, ist ein zufallsgesteuerter Satzgenerator notwendig.

● 3. Verhaltensmuster für Situation 3
Wenn ein Thema erkannt wird, über das das Kommunikationsprogramm Wissen besitzt, so kann entweder ein einleitender Satz zum Thema oder aber direkt schon speziell erfragtes Wissen als Antwort zurückgegeben werden. Bei einer Unterhaltung ist eine Einleitung zu einem Thema der Normalfall, oder ist es Ihnen schon passiert, daß eine fremde Person mit den Worten auf Sie zukam : »Aber weißt Du, das fand ich einfach schlecht von dem Geldautomaten!«. Vielmehr wird der Fremde das Gespräch mit dem Satz beginnen: »Sie kennen doch diese Geldautomaten bei den Banken? Da ist mir was Merkwürdiges passiert!« Wichtig ist in dieser Situation, daß sich das Programm das Thema der Unterhaltung merkt und sich auch noch beim nächsten Satz darauf bezieht. Falls ein anderes Thema auftaucht, muß die Entscheidung über einen Themenwechsel intern stattfinden.

● 4. Verhaltensmuster für Situation 4
Eine bestimmte Frage oder Aussage eines bekannten Themas wird vollständig erkannt. In diesem Fall wird das Wissen der Person als Antwort oder Gegenaussage zurückgegeben.

● 5. Verhaltensmuster für Situation 5
Wenn eine exakt bestimmte Aussage, die auf eine bestimmte Absicht schließen läßt, erkannt wird, wird eine bestimmte Aktion ausgelöst. Im Fall des Geldautomaten, der über die Abhebesumme verhandelt, ist diese Aktion das Feilschen um die Summe. Wenn eine solche Aktion stattfindet, sind die Absichten des Programms und des Benutzers klar umrissen und werden deshalb extern behandelt.
Unter Zuhilfenahme dieser Verhaltensmuster und des vorher beschriebenen Gehirns entsteht ein Gesprächsverhalten, das die anfangs aufgestellten Forderungen erfüllt. Den allgemeinen Ablaufplan einer Konversation mit dem Rechner entnehmen Sie aus Bild 2.

Struktur	Inhalt	Funktion
mind		simuliert Gedächtnis
	name	Zeiger auf Zeichenkette mit Name der Person
	motivpart	0>=x>=1.0 Zufallsanteil der Motivsätze
	motivnum	Anzahl der Motivsätze
	motiv	Zeiger auf Array der Motivsätze
	viewpart	0>=x>=1.0 Zufallsanteil der Ansichtensätze
	myview	Ansichten der künstlichen Person
	yoview	Ansichten des Gesprächspartners
	thema	Zeiger auf Array von Thema-Strukturen, das durch ein Nullelement abgeschlossen sein muß
view		Ansichten einer Person
	feel	Was empfindet die Person?
	thoughts	Was denkt sie?
	will	Was will sie?
	future	Was wird sie?
	should	Was soll sie?
	need	Was braucht sie?
	have	Was hat sie?
	can	Was kann sie?
thema		Thema, das der künstlichen Person bekannt ist. Die Themen werden im Array so angeordnet, daß zuerst der Typ »UNWICHTIG« und dann der Typ »WICHTIG« kommt. Außerdem entscheidet die Lage im Array über die Wichtigkeit der Themen vom Typ »WICHTIG«. Dabei ist das erste Element das unwichtigste Thema und das letzte Element des Arrays das wichtigste Thema.
	inhalt	Zeichenkette, die zu diesem Thema gehörende Oberbegriffe enthält, welche durch Kommata voneinander getrennt sind
	intro	falls nicht 0, enthält »intro« einen Zeiger auf einen das Thema einleitenden Satz
	first	Zeiger auf ein Array von Info-Strukturen, das durch ein Nullelement abgeschlossen sein muß
	type	»WICHTIG« oder »UNWICHTIG«; entscheidet, ob ein Themenwechsel stattfinden soll
	action()	Zeiger auf externe Funktion, die aufgerufen wird, falls das erste Info-Element des Themas angesprochen wird und der Zeiger nicht 0 ist
info		enthält Erkennungsmengen von zu einem Thema gehörenden Stichwörtern sowie Zeiger auf die dazu passenden Antworten. Falls die Mengenbedingungen erfüllt sind, wird hier eine Antwort ausgewählt.
	menge	Zeichenkette, die eine Oder-Menge von Stichwörtern enthält, welche im Satz vorkommen müssen. Dabei gilt: Nur eines dieser Wörter muß vorkommen. Jedes Wort wird durch ein Komma beendet.
	andmenge	Zeichenkette, die eine Und-Menge von Stichwörtern enthält, die im Satz vorkommen müssen. Dabei gilt: Alle diese Wörter müssen vorkommen. Jedes Wort wird durch ein Komma beendet.
	infnum	Anzahl der zu diesem Themenpunkt vorhandenen Antworten
	inf	Zeiger auf ein Array von Zeichenketten, das die Antworten enthält

Die Effekte und Interna des »Gehirns«

Alle bisher aufgestellten Theorien sind im Programm »talk.c« (Listing) realisiert. Ebenso finden sie in dem in dieser Ausgabe enthaltenen Rollenspiel Verwendung. Ein künstliches Gehirn wird durch die Struktur »mind« repräsentiert, die die oben definierten Eigenschaften aufweist. Diese Struktur und auch die damit verbundenen

Unterstrukturen finden Sie in Bild 3. Eine solche »Gehirnstruktur« enthält in dem Zeichenketten-Array »motiv« diejenigen Sätze, die den Gesprächspartner zu einer Aktion animieren sollen. In den beiden Strukturen vom Typ »view« sind die Ansichten der Person und des Gesprächspartners enthalten.

Das »Gehirn« wird im Programm als Struktur dargestellt

Weiterhin stehen die dem künstlichen Wesen bekannten Themen in der Struktur »thema«, die wiederum ein Array des Strukturtyps »info« enthält. Ein »thema« ist ein Oberbegriff eines Bereichs, der dem Wesen bekannt ist. Eine Info-Struktur definiert durch Mengen von Stichworten die Fragen und Antworten, die zu diesem Thema gehören. In einer solchen Struktur gibt es wiederum ein Array von Zeichenketten, das die eigentlichen Antworten enthält.

Die Mind-Struktur wird nun der Funktion »talk()« übergeben. Um Wörtern eine sehr realistische Simulation einer menschlichen Kommunikation zu erzeugen, finden mehrere Aktionen seitens dieser Funktion statt.

1. Die Ansichten des Gesprächspartners werden gespeichert, um später wieder darauf zurückkommen zu können.
2. Die Motivation und Ansichtensätze werden zufallsgesteuert an eine Antwort angehängt, um einerseits eigene Initiativen des Programms zu simulieren und andererseits einen Themenwechsel auf dem »Gehirn« bekannten Themengebieten zu erlauben.
3. Angesprochene Themen werden für eine Weile gespeichert, so daß das Programm tatsächlich weiß, worüber es gerade redet. Dadurch werden Unterhaltungen realisierbar wie beispielsweise die folgende:
   Du: Hast du Zeit?
   Computer: Ja, habe ich.
   Du: Wie lange?
   Computer: Zwei Stunden, dann muß ich abstürzen.
   Auch durch Motivationssätze angesprochene Themen können einen Themenwechsel durchführen.
4. Es gibt sogenannte »wichtige« und »unwichtige« Themen. Bei einem Gespräch kann ein Themenwechsel nur auf ein »wichtiges« Thema erfolgen. Die »unwichtigen« Themen sind als allgemeine Informationsquellen immer abfragbar.
5. Wenn mehrere zu einem Thema gespeicherte Informationen existieren, wird die Antwort zufallsgesteuert ausgewählt.
6. Absichten eines Themas werden erkannt, wodurch ein externes Programm gestartet wird, so daß verschiedene Reaktionen auf verschiedene Absichten folgen.

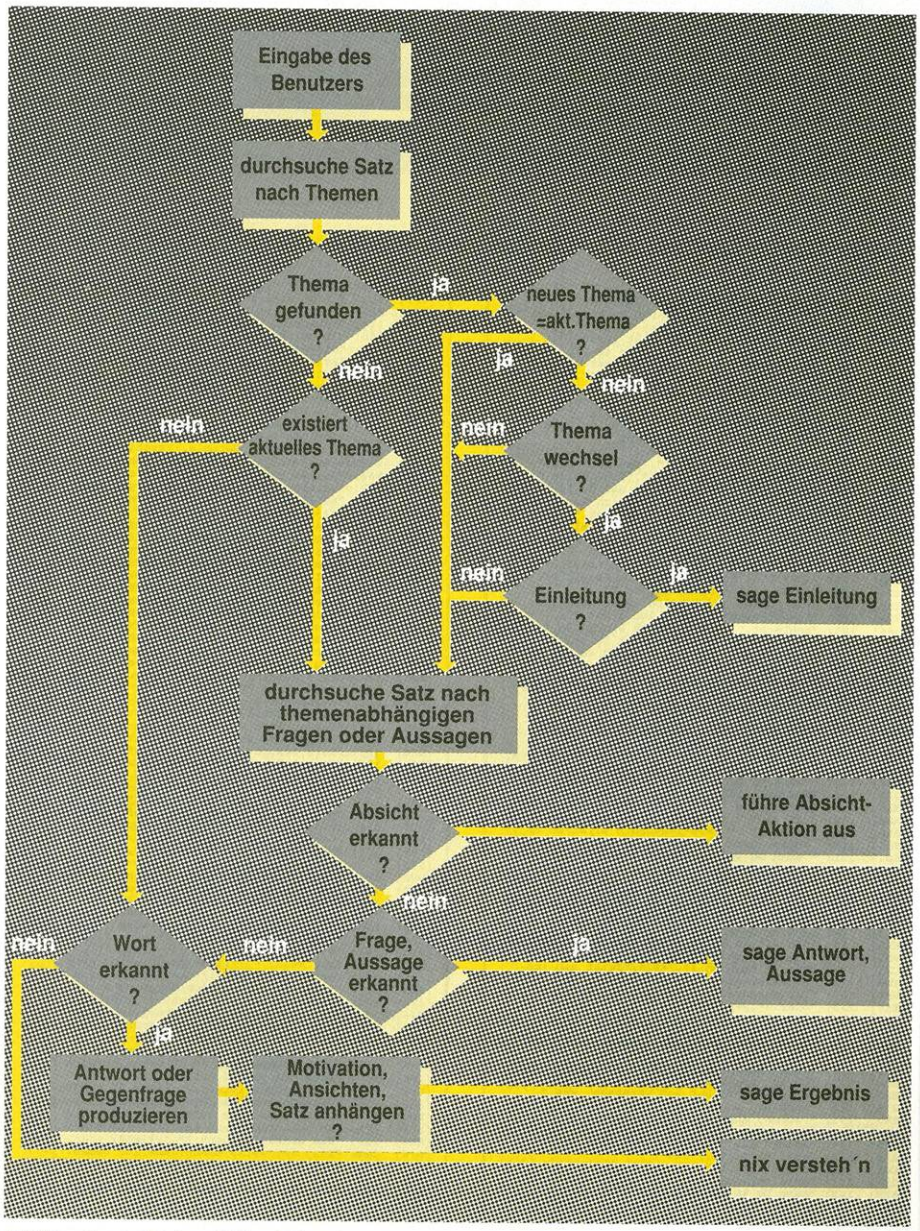


Bild 2. So sieht der Ablaufplan für eine Gesprächsführung aus.

Um eigene »Gehirne« entwerfen zu können, finden Sie in der Tabelle die Funktionen und alle Effekte beschrieben und sehen im Programm »talk.c« drei »Beispiel-Gehirne« verschiedenen Umfangs, die zu dem in dieser Ausgabe veröffentlichten Rollenspiel gehören. Das Programm »talk.c« ist jedoch auch unabhängig von diesem Spiel anwendbar. Durch die Angabe einer Nummer zwischen 1 und 3 als Parameter wählen Sie ein Gehirn aus und unterhalten sich mit der ausgewählten Person. Das bedeutet: Durch die Eingabe der Kommandozeilen »talk 1«, »talk 2« oder »talk 3« unterhalten Sie sich mit Graf Dovan, Mylady Miriam1 oder Mylady Miriam2. Weitere interne Abhängigkeiten der »Gehirnstruktur« finden Sie in Bild 3.

Mit dem hier vorgestellten Konzept realisieren Sie sehr einfach das – zu Beginn dargestellte – Beispiel Geldautomat-Kunde und erreichen bei ausgebauteren Gehirnen verblüffende Effekte. Auch ist es denkbar, eine sich ändernde Stimmung – wie Wut oder gute Laune – der künstlichen Person mitzugeben. Eine weitere sehr interessante Idee ist es, eine zusätzliche Kommunikationsstufe einzubauen, die ein primitives Lernen eingegebener Zusammenhänge erlaubt. Je öfter Sie sich dann mit dem Programm unterhalten, desto mehr weiß es. Die Optionen sind vielfältig, und der Effekt ist so verblüffend, daß Sie sich nicht wundern müssen, wenn Sie von Ihrer nächsten Einladung auf ein Fest aus der Unterhaltung mit Ihrem Computer erfahren.

(Matthias Oberdorfer/bm)

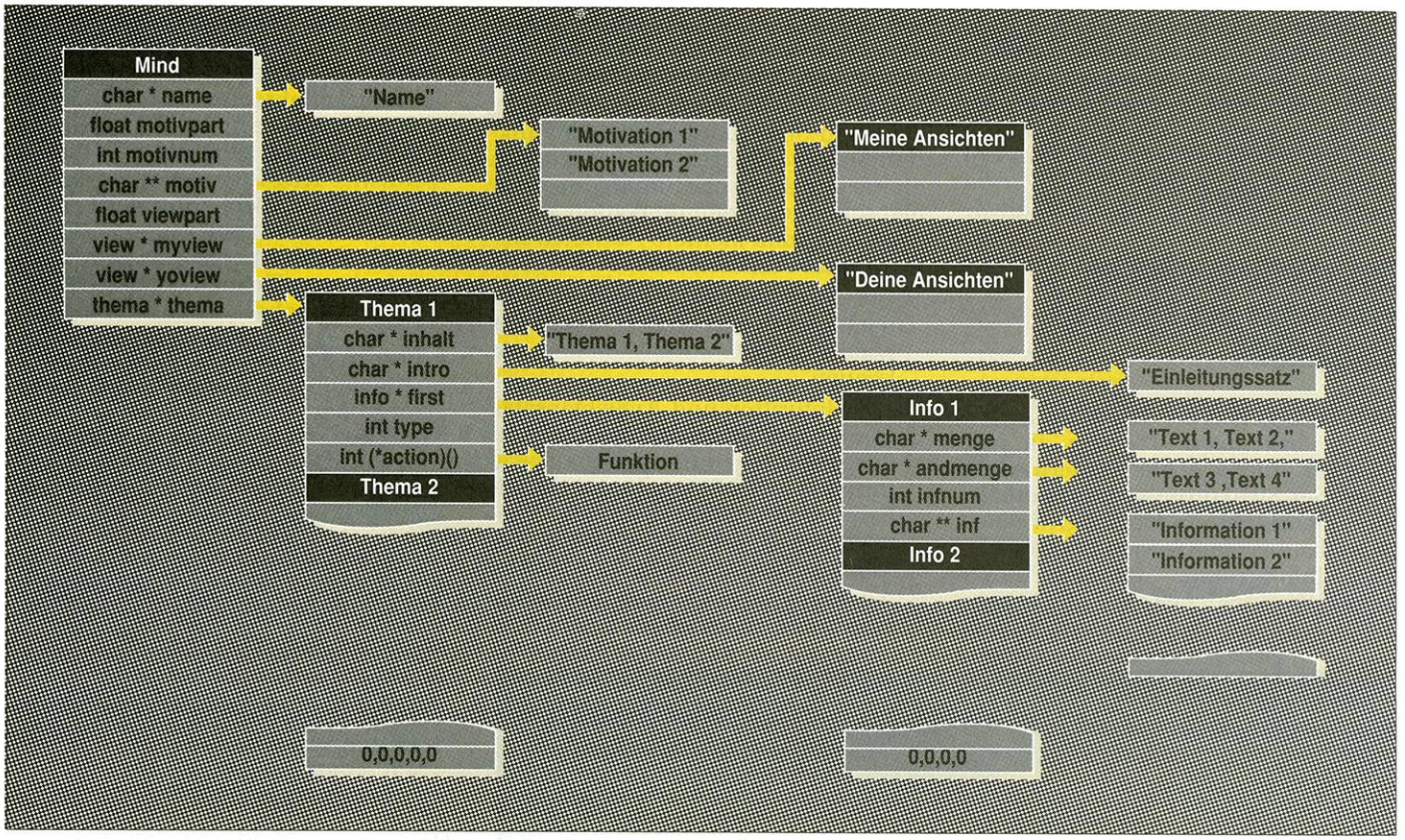


Bild 3. Die Struktur des »Gehirns« und ihre Zusammenhänge

```
[W6Ø]    1: /*
[UP2]    2:  Programm: talk.c
[ZM6]    3:  Funktion: simuliert Kommunikation mit
[LN4]    4:            unterschiedlichen Personen
[K83]    5:  Sprache:  Turbo C 2.Ø
[KR4]    6:  Autor:    Matthias Oberdorfer
[8Z4]    7:  (c) 1992 DMV Widuch GmbH & Co. KG */
[ØØØ]    8:
[WZ1]    9: #ifndef TALK
[2Ø2]   1Ø: #define TALK
[B53]   11: #include<string.h>
[Ø43]   12: #include<stdlib.h>
[ZP2]   13: #include<time.h>
[XU2]   14: #include<ctype.h>
[7M2]   15: #include<dos.h>
[8W2]   16: #include<stdio.h>
[EU2]   17: #include<conio.h>
[ØZ1]   18: #ifndef GAME
[E94]   19: #define printinbox printf
[2TØ]   2Ø: #else
[1P6]   21: extern int printinbox(const char *i, ...);
[B56]   22: extern void print(char *s,unsigned i);
[NZØ]   23: #endif
[3M3]   24: #define NIXVERSTEHN Ø
[HL3]   25: #define FRAGEZEICHEN 1
[DD2]   26: #define ICHDU 2
[2T2]   27: #define ONEWORD 6
[2C2]   28: #define FRAGE 3
[VW2]   29: #define ANTWORT 4
[2Y2]   3Ø: #define ADJEKTIV 5
[6G2]   31: #define GRUND 7
[ZD2]   32: #define THEMA 8
[773]   33: #define NEBENTHEMA 9
[TM2]   34: #define ACTION 1Ø
[1X3]   35: #define WIEDERHOLUNG 13
[6X3]   36: #define KEINEANTWORT 14
[X23]   37: #define ABSCHIED 15
[663]   38: #define SATZLEN 128
[1Ø3]   39: #define ANZWORD 3Ø
[U73]   4Ø: #define UNWICHTIG Ø
[US2]   41: #define WICHTIG 1
[8M2]   42: typedef struct
[UG5]   43:   (char *lexem; /* Schlüsselwort */
[TD4]   44:    int imp;     /* Wichtigkeit*/
[SØ6]   45:    int num;     /* Anzahl der Antworten */
[V26]   46:    char **(ans[8]); /* Verweise */ ) Word;
[5E3]   47: static char *aw[] = {
[G96]   48: "Du bist also %s ?","Ich bin mir da nicht"
[CS4]   49: " so sicher, dass Du %s bist.",
[D86]   5Ø: "Woher soll ich wissen, ob Du %s bist ?",
[K95]   51: "Keine Ahnung!","Ich weiss nicht.",
[8A4]   52: "Kann ich auch nicht sagen.",
[KX5]   53: "Tja, gute Frage.","Das weiss ich auch"
[9M5]   54: " nicht.","Mmmmh. Tja, ich weiss auch"
[HF5]   55: " nicht.","Finde es selbst heraus.",
[X36]   56: "Ich bin also %s ! Und was bist dann Du?",
[7V5]   57: "Ja ich weiss !","Ich bin zwar %s, aber"
[TK4]   58: " Du auch ein bischen. Oder ?",
[YG5]   59: "Sogar sehr %s. Meinst Du nicht ?",
[1E5]   6Ø: "Ich bin mindestens so %s, wie Du !",
[N24]   61: "Was hält Dich denn %s ?",
[H36]   62: "Denkst Du, es ist gut %s zu bleiben ?",
[FR3]   63: "Dem ist wohl so ! Und ?",
[SP4]   64: "Sieht so aus. Und was jetzt ?",
[W46]   65: "%s - Ist das wirklich notwendig für"
[L96]   66: " Dich ?","Bist Du sicher, daß Du nicht "
[6P6]   67: "ohne %s auskommst ?","Wer weiß, ob ich "
[1N5]   68: "%s bleibe? Was würdest Du tun ?",
[P96]   69: "Ja! Wenn Du mir %s gibst, verrate ich "
[236]   7Ø: "Dir was.","%s kann ich immer brauchen!"
[ØL6]   71: " Aber woher kriegen ?","Stimmt. Manchmal"
[BG6]   72: " finde sogar ich %s.","Aus welchem Grund"
[RU5]   73: " findest Du %s ?","Finde ich nett von "
[CL4]   74: "Dir, %s zu geben !","Gibst Du "
[TZ6]   75: "öfters %s ?","Wer erlaubt Dir, daß Du %s"
```

```
[XH4]   76: " darfst ?","Darf ich auch %s ?",
[ZB5]   77: "Ich würde %s gerne mitkommen.",
[B16]   78: "Wieso gerade %s ?","Aber Du bist nicht "
[B65]   79: "sicher ?","Wieso denkst Du %s ?",
[7G5]   8Ø: "Wenn Du Dich da nur nicht irrst !",
[X75]   81: "Echt ? Ich hätte auch gern %s .",
[MN5]   82: "Wie findest Du, daß Du %s hast ?",
[F25]   83: "Freu Dich doch darüber. (haha)",
[Ø53]   84: "Wieso hast Du %s ?",
[N96]   85: "Ach ? Und warum hättest Du gerne %s ?",
[395]   86: "Ich weiss nicht, ob ich %s habe.",
[GV5]   87: "Sicher, ich habe %s! Und was ist mit "
[2M4]   88: "Dir ?","Ah ja. Interessant.",
[Z66]   89: "So, so, na und ?","Aha !","Das hoffe ich"
[6W5]   9Ø: " auch.","Wer weiß ?!","Man soll die "
[3R5]   91: "Hoffnung nie aufgeben.","Bist Du da "
[GU5]   92: "sicher ?","Wer sagt Dir, daß Du %s "
[2C5]   93: "kannst?","Vielleicht kann ich %s.",
[AH5]   94: "Ich weiß nicht, ob ich %s kann.",
[6B5]   95: "Vielleicht.", /* 52*/"Ich bin mir da "
[8C6]   96: "nicht ganz sicher.","Wer kommt sonst "
[Ø25]   97: "noch %s ?","Warum kommst Du %s ?",
[KC6]   98: "Ich würde Dir gerne dabei mitmachen.",
[ZP5]   99: /*56*/"Ok, mach %s, wenn Du es willst.",
[MH4]  1ØØ: "Warum willst Du %s machen ?",
[EP5]  1Ø1: "Was denkst Du, wenn Du %s machst?",
[HA5]  1Ø2: "Einverstanden! Finde ich nicht "
[566]  1Ø3: "schlecht.",/*6Ø */"Warum willst Du %s ?",
[YW5]  1Ø4: "Aus welchem Grund möchtest Du %s?",
[EØ6]  1Ø5: "Weshalb %s?","Warum musst Du %s ?",/*64*/
[MR4]  1Ø6: "Wer sagt, daß Du %s musst ?",
[CP4]  1Ø7: "Ich denke, ich muss auch %s.",
[TK6]  1Ø8: "Ich weiß nicht, ob das gut ist",  /*67*/
[ZA4]  1Ø9: "Warum willst Du das tun ?",
[ØT4]  11Ø: "Glaubst Du das bringt was ?",
[CN5]  111: "Ich zweifle gar nicht daran.", /*7Ø*/
[674]  112: "Davon bin ich ueberzeugt.",
[KL6]  113: "Was weisst Du schon!","Wie meinst Du das"
[8Z5]  114: " ?",/* 73 */"Denkst Du, dass es gut ist,"
[Y36]  115: " wenn Du %s willst ?","Alle wollen %s! "
[3E5]  116: "Was soll ich dazu sagen ?","Warum "
[BA6]  117: "würdest Du %s ?",/*76*/"Das sagst Du so"
[A86]  118: " einfach.","Findest Du es gut, wenn Du "
[896]  119: "%s wirst ?",/*78*/"Weshalb wirst Du %s ?"
[5C6]  12Ø: ,"Wieso denn ?","Ich finde es nicht gut,"
[5Ø6]  121: " wenn Du %s vergisst.",/*81*/"%s scheint"
[SU5]  122: " Dir nicht so wichtig zu sein.","Ich "
[PU6]  123: "vergesse normalerweise nie etwas.",/*83*/
[W65]  124: "Also gut! ich habe %s vergessen.",
[HS5]  125: "Warum denkst Du %s ?", /*85*/"Ich denke"
[LC5]  126: " manchmal auch %s.","Rate mal wer !",
[ZZ3]  127:  /*87*/"Denk doch mal nach !",
[KR5]  128: "Was willst Du ?",/*89*/"Was soll ich"
[9N5]  129: " dazu sagen ?","Was hab' ich damit zu "
[CX5]  13Ø: "tun ?","Ach so !",/*92*/"Jetzt hab ich "
[TM5]  131: "es verstanden.","Deswegen also ?",
[YK5]  132: "Und warum das ?","Wurde auch Zeit, "
[K55]  133: "oder ?",/*96*/"Weshalb das ?",/*97*/
[4M5]  134: "Wieso denn ?","Warum ?","Ja! Genau.",
[A26]  135: /*1ØØ*/"Das ist klar.","Sicher darf ich "
[5Q5]  136: "%s.",/*1Ø2*/"Sag mir : Warum darf ich"
[NK4]  137: " %s ?","Das ist gut, wenn Du %s "
[FD6]  138: "verstehst.",/*1Ø4*/"Bist Du sicher, daß "
[E56]  139: "Du %s verstehst ?","Was verstehst Du "
[NN5]  14Ø: "schon ?","Sicher ich verstehe, was "
[276]  141: "dachtest Du denn ?",/*1Ø7*/"Ich verstehe"
[1V5]  142: " schon, aber auch Du verstehst %s !",
[HY5]  143: "Ich schon, aber wie sieht es bei Dir "
[AZ5]  144: "selbst aus ?","Vielleicht will ich "
[AZ2]  145: "aber nicht !",/*11Ø*/
[GC5]  146: "Eigentlich hab ich keine Lust !",
[5K6]  147: "Dazu muss mich erst jemand überreden.",
[5P5]  148: "Wenn Du %s kennst, dann aber los!",
[MN6]  149: /*113*/"Woher weißt Du so gut Bescheid ?",
[U35]  15Ø: "Aha.",/*115*/"Und dann?","Echt? Was "
```

```
[RC6]  151: "dann ?","Ich weiss nicht nur %s, sondern"
[YP3]  152: " noch viel mehr!", /*118*/
[YH4]  153: "Tja, ich weiss, ich weiss.",
[264]  154: "Natürlich weiss ich %s.",
[415]  155: "In welcher Beziehung ?", /*121*/
[NR3]  156: "Findest Du ? Wieso ?",
[NR5]  157: "Besser als so ärmlich, finde ich.",
[P54]  158: /*123*/"Tut mir echt leid !",
[FH5]  159: "Wer recht hat, sollte auch recht "
[HN2]  16Ø: "behalten.",/*125*/
[DR5]  161: "Die Wahrheit siegt am Ende immer.",
[W85]  162: "Ich bin gegen alle Unwahrheit.",
[A56]  163: "Ich bin klein, mein Herz ist rein, ...",
[KC6]  164: "Du und Deine Gefühle. Erzähl bitte "
[YF6]  165: "Näheres ?",/*129*/"Warum fühlst Du %s ?",
[HD5]  166: "Ja. Es geht.",/* 131 */"Naja. Wie man's"
[HZ5]  167: " nimmt.","Was heisst schon gut oder "
[FX5]  168: "schlecht ?","Was soll man da machen ?",
[4Z3]  169: /*134*/"Naja, so schlimm ?",
[EF6]  17Ø: "Tja, Klugheit macht sich bezahlt", /*136*/
[7D6]  171: "Ich sag nur: So schlau wie ein Fuchs",
[XG6]  172: "Gesundheit ist das Wichtigste !", /*138*/
[GX5]  173: "Echt ? Krankheit ?","Zufriedenheit "
[4C5]  174: "ist wichtig!", /*14Ø*/"Glück ist "
[356]  175: "Schicksal.","Unzufriedenheit bringt "
[VX5]  176: "nichts.",/*142*/"Ein Unglück kommt "
[D76]  177: "selten allein.","Naja, ich habe schon "
[S65]  178: "besseres Wetter erlebt.", /*144*/
[EP6]  179: "Nicht schlecht für diese Jahreszeit.",
[NL6]  18Ø: "Nichts geht über echte Freunde.",/*146*/
[W56]  181: "Ich habe nur wenig echte Freunde und "
[4Ø6]  182: "Du ?","Meine Freunde sind auch Deine "
[N26]  183: "Freunde !","Viel Feind viel Ehr, naja,"
[RT3]  184: " wenn das so wär.", /*149*/
[AN4]  185: "Die Feinde lauern überall."
[ØE6]  186: "Wer an Pech glaubt, der hat auch Pech.",
[ØP4]  187: /*151*/"Wer wagt, der gewinnt.",
[S84]  188: "Geld braucht jeder.", /*153*/
[P86]  189: "Weisst Du wie man schnell reich wird?",
[8C4]  19Ø: "Nichts ist Zufall.",/*155*/
[ØD6]  191: "Ich glaube, alles ist vorherbestimmt.",
[AW5]  192: "Hallo! Wie geht's?",/*157*/"Guten Tag!",
[VM6]  193: "Sei gegrüßt!","Wer sagt, daß ich %s "
[NU5]  194: "muss ?", /*16Ø*/"Tja, ich muss also %s,"
[8N3]  195: " und was ist mit Dir ?",
[E95]  196: "Wer weiss, ob ich %s kann.", /*162 */
[975]  197: "Nichts ist sicher.",          /* 163 */
[UV6]  198: "Vielleicht aber vielleicht auch nicht.",
[CX5]  199: "Du scheinst da unsicher zu sein.",
[KP4]  2ØØ: "Ich weiss auch nicht.", /* 166*/
[FQ6]  2Ø1: "Warum ?",/*167*/"Wieso ?","Warum denn ?",
[4P5]  2Ø2: /*169*/"Erkläre mir bitte wieso ?",
[8X4]  2Ø3: "Hmmm. Nicht so leicht !", /* 171 */
[465]  2Ø4: "Tja, so einfach ist das nicht !",
[MM5]  2Ø5: "Wer weiss.", /*173*/"Kann sein, aber "
[H36]  2Ø6: "warum ?","Was soll das denn heissen ?",
[YH5]  2Ø7: /*175*/"Mach richtige Sätze sonst "
[1X5]  2Ø8: "verstehe ich Dich nicht !","Sprechen "
[U46]  2Ø9: "wir über ein anderes Thema, wenn Du "
[CD5]  21Ø: "Dich nicht ausdrücken kannst.",
[H94]  211: "Willst Du mich ärgern ?",
[DØ6]  212: "So, jetzt habe ich genug von Deinem "
[6M6]  213: "dummen Geschwätz !","Unterhalte Dich "
[284]  214: "doch mit jemand anderen !",
[CS5]  215: "Hallo ? Was ist los mit Dir ?", /*181*/
[156]  216: "Bist Du noch da ?","Wenn Du keine Lust "
[KD5]  217: "hast, habe auch keine Lust mehr !",
[ZF4]  218: "Naja, gerade noch so.", /* 184 */
[MK4]  219: "Was denkst Du denn ?", /*185*/
[G16]  22Ø: "Tja, wer weiss schon, was ich denke.",
[G27]  221: "Ich fühle mich soweit ganz gut und Du ?",
[WW5]  222: /*187*/"Vielleicht fühle ich %s, aber "
[A16]  223: "ich bin nicht sicher.","Woher willst "
[TE5]  224: "Du wissen, ob ich %s will ?", /* 189*/
[836]  225: "Was willst Du denn ?","Ja. Ich werde "
```

```
[C65]  226: "vielleicht %s, aber ist das so"
[LD5]  227: " schlimm ?", /*191*/"Kann sein, daß "
[2P5]  228: "ich %s werde, aber Du ?","Bis dann.",
[3U4]  229: "Bitte, gern geschehen.", /*194*/
[NP6]  230: "Ich tue was ich kann.","Zu was man Lust "
[4Ø5]  231: "hat und was man tut ist sehr "
[495]  232: "verschieden, findest Du nicht?",
[EU5]  233: "Besser Lust als Frust, finde ich !",
[M31]  234: /*197*/};
[AZ3]  235: static Word sprache[] =
[KT6]  236: {{"BIN",ONEWORD,3,&aw[Ø],&aw[1],&aw[2]},
[ZU5]  237:  {"BIST", ONEWORD,5,&aw[1Ø],&aw[11],
[C64]  238:                &aw[12],&aw[13],&aw[14]},
[1T5]  239:  {"BLEIBE", ONEWORD,2,&aw[15],&aw[16]},
[MR4]  24Ø:  {"BLEIBST",ONEWORD,1,&aw[21]},
[ES5]  241:  {"SEIN",ONEWORD,2,&aw[17],&aw[18]},
[XW5]  242:  {"BRAUCHE",ONEWORD,2,&aw[19],&aw[2Ø]},
[GE6]  243:  {"BRAUCHST",ONEWORD,2,&aw[22],&aw[23]},
[4Z5]  244:  {"DENKE",ONEWORD,5,&aw[32],&aw[33],
[Y74]  245:                &aw[34],&aw[85],&aw[86]},
[3D6]  246:  {"DENKST",ONEWORD,2,&aw[185],&aw[186]},
[216]  247:  {"ERZAEHLE",ONEWORD,3,&aw[7Ø],&aw[71],
[LG1]  248:                &aw[72]},
[8A6]  249:  {"FINDE",ONEWORD,2,&aw[24],&aw[25]},
[TA6]  25Ø:  {"FUEHLE",ONEWORD,2,&aw[129],&aw[13Ø]},
[1H6]  251:  {"FUEHLST",ONEWORD,2,&aw[187],&aw[188]},
[VN5]  252:  {"GEBE",ONEWORD,2,&aw[26],&aw[27]},
[KP5]  253:  {"DARF",ONEWORD,2,&aw[28],&aw[29]},
[HA6]  254:  {"DARFST",ONEWORD,2,&aw[1Ø2],&aw[1Ø3]},
[ZW4]  255:  {"GAB",ONEWORD,3,&aw[42],&aw[43],
[LG1]  256:                &aw[44]},
[3P5]  257:  {"GEHE",ONEWORD,2,&aw[3Ø],&aw[31]},
[TP4]  258:  {"GEHT",ONEWORD,1,&aw[184]},
[CY5]  259:  {"GLAUBE",ONEWORD,3,&aw[32],&aw[33],
[HG1]  26Ø:                &aw[34]},
[295]  261:  {"HABE",ONEWORD,4,&aw[35],&aw[36],
[EW2]  262:                &aw[37],&aw[38]},
[ES5]  263:  {"HAST",ONEWORD,2,&aw[4Ø],&aw[41]},
[3V4]  264:  {"HAETTE",ONEWORD,1,&aw[39]},
[HC6]  265: {"HAT",ONEWORD,3,&aw[42],&aw[43],&aw[44]},
[7D6]  266: {"IST",ONEWORD,3,&aw[42],&aw[43],&aw[44]},
[4Z5]  267:  {"HOFFE",ONEWORD,2,&aw[45],&aw[46],
[WG1]  268:                &aw[47]},
[ES5]  269:  {"KANN",ONEWORD,2,&aw[48],&aw[49]},
[116]  27Ø:  {"KANNST",ONEWORD,2,&aw[5Ø],&aw[51],
[DL1]  271:                &aw[162]},
[VA6]  272:  {"KENNE",ONEWORD,2,&aw[113],&aw[114]},
[EZ5]  273:  {"KOENNTE",ONEWORD,2,&aw[52],&aw[53]},
[1C6]  274:  {"KOMME",ONEWORD,2,&aw[54],&aw[55]},
[PZ5]  275:  {"MACHE",ONEWORD,4,&aw[56],&aw[57],
[WW2]  276:                &aw[58],&aw[59]},
[LA5]  277:  {"MACH",ONEWORD,4,&aw[56],&aw[57],
[WW2]  278:                &aw[58],&aw[59]},
[HM5]  279:  {"MOECHTE",ONEWORD,6,&aw[6Ø],&aw[61],
[ZK5]  28Ø:  &aw[62],&aw[63],&aw[74],&aw[75]},
[695]  281:  {"MUSS",ONEWORD,3,&aw[64],&aw[65]
[GH1]  282:  ,&aw[66]},
[F56]  283:  {"MUSST",ONEWORD,5,&aw[16Ø],&aw[161],
[1D4]  284:  &aw[11Ø],&aw[111],&aw[112]},
[KB5]  285:  {"SAGE",ONEWORD,3,&aw[67],&aw[68],
[6H1]  286:  &aw[69]},
[TC5]  287:  {"SOLL",ONEWORD,3,&aw[64],&aw[65],
[XG1]  288:  &aw[66]},
[A16]  289:  {"SOLLST",ONEWORD,5,&aw[16Ø],&aw[161],
[1D4]  29Ø:  &aw[11Ø],&aw[111],&aw[112]},
[E26]  291:  {"SOLLTE",ONEWORD,6,&aw[6Ø],&aw[61],
[ZK5]  292:  &aw[62],&aw[63],&aw[74],&aw[75]},
[ZG6]  293:  {"VERGESSE",ONEWORD,2,&aw[81],&aw[82]},
[ØL6]  294:  {"VERGISST",ONEWORD,2,&aw[83],&aw[84]},
[P16]  295:  {"VERGISS",ONEWORD,2,&aw[83],&aw[84]},
[3L6]  296:  {"VERSTEHE",ONEWORD,3,&aw[1Ø4],&aw[1Ø5]
[5U1]  297:  ,&aw[1Ø6]},
[MW6]  298:  {"VERSTEHST",ONEWORD,3,&aw[1Ø7],&aw[1Ø8]
[8U1]  299:  ,&aw[1Ø9]},
[9E6]  3ØØ: {"WAR",ONEWORD,3,&aw[42],&aw[43],&aw[44]},
[P36]  3Ø1:  {"WEISS",ONEWORD,5,&aw[7Ø],&aw[71],
[P74]  3Ø2:  &aw[72],&aw[48],&aw[73]},
[7D6]  3Ø3:  {"WAERE",ONEWORD,2,&aw[48],&aw[73]},
[W16]  3Ø4:  {"WEISST",ONEWORD,3,&aw[118],&aw[119],
[TK1]  3Ø5:  &aw[12Ø]},
[97X]  3Ø6:  {"WUERDE",ONEWORD,2,&aw[76],&aw[77]},
[1Ø6]  3Ø7:  {"WERDE",  ONEWORD,3,&aw[78],&aw[79],
[FG1]  3Ø8:  &aw[8Ø]},
[WF6]  3Ø9:  {"WIRST",ONEWORD,2,&aw[191],&aw[192]},
[XF6]  31Ø:  {"WILL",ONEWORD,6,&aw[6Ø],&aw[61],&aw[62]
[2W3]  311:  ,&aw[63],&aw[74],&aw[75]},
[ØG6]  312:  {"WILLST",ONEWORD,2,&aw[189],&aw[19Ø]},
[766]  313:  {"WER",FRAGE,2,&aw[87],&aw[4],&aw[8]},
[HM5]  314:  {"WO",FRAGE,7,&aw[3],&aw[4],&aw[5],
[BT4]  315:  &aw[6],&aw[7],&aw[8],&aw[9]},
[NX5]  316:  {"WANN",FRAGE,7,&aw[3],&aw[4],&aw[5],
[BT4]  317:  &aw[6],&aw[7],&aw[8],&aw[9]},
[EB6]  318:  {"WIESO",FRAGE,7,&aw[3],&aw[4],&aw[5],
[BT4]  319:  &aw[6],&aw[7],&aw[8],&aw[9]},
[YB6]  32Ø:  {"WARUM",FRAGE,7,&aw[3],&aw[4],&aw[5],
[BT4]  321:  &aw[6],&aw[7],&aw[8],&aw[9]},
[5P5]  322:  {"WIE",FRAGE,7,&aw[3],&aw[4],&aw[5],
[BT4]  323:  &aw[6],&aw[7],&aw[8],&aw[9]},
[2X5]  324:  {"WAS",FRAGE,8,&aw[88],&aw[3],&aw[4],
[776]  325:  &aw[5],&aw[6],&aw[7],&aw[8],&aw[9]},
[765]  326:  {"WESHALB",FRAGE,7,&aw[3],&aw[4],
[776]  327:  &aw[5],&aw[6],&aw[7],&aw[8],&aw[9]},
[XM5]  328:  {"?",FRAGEZEICHEN,7,&aw[3],&aw[4],
[776]  329:  &aw[5],&aw[6],&aw[7],&aw[8],&aw[9]},
[685]  33Ø:  {"WIEVIEL",FRAGE,7,&aw[3],&aw[4],
[776]  331:  &aw[5],&aw[6],&aw[7],&aw[8],&aw[9]},
[U75]  332:  {"WOVON",FRAGE,7,&aw[3],&aw[4],
[776]  333:  &aw[5],&aw[6],&aw[7],&aw[8],&aw[9]},
[195]  334:  {"WODURCH",FRAGE,7,&aw[3],&aw[4],
[776]  335:  &aw[5],&aw[6],&aw[7],&aw[8],&aw[9]},
[E75]  336:  {"WOMIT",FRAGE,7,&aw[3],&aw[4],
[776]  337:  &aw[5],&aw[6],&aw[7],&aw[8],&aw[9]},
[C65]  338:  {"WOHIN",FRAGE,7,&aw[3],&aw[4],
[776]  339:  &aw[5],&aw[6],&aw[7],&aw[8],&aw[9]},
[G65]  34Ø:  {"WOHER",FRAGE,7,&aw[3],&aw[4],
[776]  341:  &aw[5],&aw[6],&aw[7],&aw[8],&aw[9]},
[Ø55]  342:  {"WOBEI",FRAGE,7,&aw[3],&aw[4],
[776]  343:  &aw[5],&aw[6],&aw[7],&aw[8],&aw[9]},
[V45]  344:  {"WORAUF",FRAGE,7,&aw[3],&aw[4],
[776]  345:  &aw[5],&aw[6],&aw[7],&aw[8],&aw[9]},
[865]  346:  {"WORAN",FRAGE,7,&aw[3],&aw[4],
[776]  347:  &aw[5],&aw[6],&aw[7],&aw[8],&aw[9]},
[FY4]  348:  {"WOZU",FRAGE,7,&aw[3],&aw[4],
[776]  349:  &aw[5],&aw[6],&aw[7],&aw[8],&aw[9]},
[B45]  35Ø:  {"WOFUER",FRAGE,7,&aw[3],&aw[4],
[776]  351:  &aw[5],&aw[6],&aw[7],&aw[8],&aw[9]},
[NM5]  352:  {"WORUEBER",FRAGE,7,&aw[3],&aw[4],
[776]  353:  &aw[5],&aw[6],&aw[7],&aw[8],&aw[9]},
[S85]  354:  {"WORRAUS",FRAGE,7,&aw[3],&aw[4],
[776]  355:  &aw[5],&aw[6],&aw[7],&aw[8],&aw[9]},
[AS6]  356:  {"DIR",ICHDU,3,&aw[89],&aw[9Ø],&aw[91]},
[9N6]  357:  {"DICH",ICHDU,3,&aw[89],&aw[9Ø],&aw[91]},
[X86]  358:  {"DU",ICHDU,3,&aw[89],&aw[9Ø],&aw[91]},
[GN6]  359:  {"WEIL",GRUND,6,&aw[92],&aw[93],&aw[94],
[CC4]  36Ø:  &aw[95],&aw[1ØØ],&aw[1Ø1]},
[9M6]  361:  {"DASS",GRUND,6,&aw[92],&aw[93],&aw[94],
[CC4]  362:  &aw[95],&aw[1ØØ],&aw[1Ø1]},
[ØX5]  363:  {"DANN",ANTWORT,3,&aw[115],&aw[116],
[KL1]  364:  &aw[117]},
[GK6]  365:  {"VIELLEICHT",GRUND,3,&aw[163],&aw[164],
[TL1]  366:  &aw[165]},
[PU5]  367:  {"SICHER",GRUND,3,&aw[163],&aw[164],
[TL1]  368:  &aw[165]},
[T14]  369:  {"LOS",GRUND,1,&aw[166]},
[2L5]  37Ø:  {"ABER",GRUND,2,&aw[169],&aw[17Ø]},
[GF6]  371:  {"JETZT",ADJEKTIV,2,&aw[167],&aw[168]},
[RU5]  372:  {"WAHRSCHEINLICH",GRUND,4,&aw[163],
```

```
[1F4]  373:  &aw[173],&aw[165],&aw[174]},
[DØ6]  374:  {"GENAU",ANTWORT,3,&aw[92],&aw[93],
[AL1]  375:  &aw[115]},
[6T4]  376:  {"ENDLICH",ONEWORD,1,&aw[96]},
[5W4]  377:  {"DANKE",ONEWORD,1,&aw[194]},
[VN4]  378:  {"DANK",ONEWORD,1,&aw[194]},
[FX4]  379:  {"BITTE",ONEWORD,1,&aw[195]},
[BU6]  38Ø:  {"MIR",ICHDU,3,&aw[97],&aw[98],&aw[99]},
[TR6]  381:  {"MICH",ICHDU,3,&aw[97],&aw[98],&aw[99]},
[1T6]  382:  {"ICH",ICHDU,3,&aw[97],&aw[98],&aw[99]},
[W72]  383:  /* Adjektive */
[HU6]  384: {"AEHNLICH",ADJEKTIV,2,&aw[121],&aw[122]},
[YR5]  385:  {"VERSCHIEDEN",ADJEKTIV,2,&aw[121],
[1L1]  386:  &aw[122]},
[EE5]  387:  {"LANGWEILIG",ADJEKTIV,2,&aw[121],
[1L1]  388:  &aw[122]},
[DT5]  389:  {"INTERESSANT",ADJEKTIV,2,&aw[121],
[1L1]  39Ø:  &aw[122]},
[4W4]  391:  {"REICH",ADJEKTIV,1,&aw[123]},
[2M4]  392:  {"ARM",ADJEKTIV,1,&aw[124]},
[RZ5]  393:  {"RECHT",ADJEKTIV,4,&aw[125],&aw[126],
[623]  394:  &aw[127],&aw[128]},
[GH6]  395:  {"RICHTIG",ADJEKTIV,4,&aw[125],&aw[126],
[623]  396:  &aw[127],&aw[128]},
[PY5]  397:  {"WAHR",ADJEKTIV,4,&aw[125],&aw[126],
[623]  398:  &aw[127],&aw[128]},
[9P6]  399:  {"WAHRHEIT",ADJEKTIV,4,&aw[125],&aw[126],
[623]  39Ø:  &aw[127],&aw[128]},
[BH6]  4Ø1:  {"UNRECHT",ADJEKTIV,4,&aw[125],&aw[126],
[623]  4Ø2:  &aw[127],&aw[128]},
[F36]  4Ø3:  {"FALSCH",ADJEKTIV,4,&aw[125],&aw[126],
[623]  4Ø4:  &aw[127],&aw[128]},
[GP6]  4Ø5:  {"UNRICHTIG",ADJEKTIV,4,&aw[125],&aw[126],
[623]  4Ø6:  &aw[127],&aw[128]},
[ØT5]  4Ø7:  {"GUT",ADJEKTIV,3,&aw[131],&aw[132],
[8L1]  4Ø8:  &aw[133]},
[ZL6]  4Ø9:  {"SCHLECHT",ADJEKTIV,3,&aw[131],&aw[132],
[8L1]  41Ø:  &aw[133]},
[R56]  411:  {"DUMM",ADJEKTIV,2,&aw[134],&aw[135]},
[A96]  412:  {"BLOED",ADJEKTIV,2,&aw[134],&aw[135]},
[P66]  413:  {"KLUG",ADJEKTIV,2,&aw[136],&aw[137]},
[3G6]  414:  {"SCHLAU",ADJEKTIV,2,&aw[136],&aw[137]},
[ZR5]  415:  {"INTELLIGENT",ADJEKTIV,2,&aw[136],
[SL1]  416:  &aw[137]},
[T45]  417:  {"GESUND",ADJEKTIV,1,&aw[138]},
[DX4]  418:  {"KRANK",ADJEKTIV,1,&aw[139]},
[BC5]  419:  {"ZUFRIEDEN",ADJEKTIV,2,&aw[14Ø],
[3L1]  42Ø:  &aw[141]},
[GB5]  421:  {"GLUECKLICH",ADJEKTIV,2,&aw[14Ø],
[3L1]  422:  &aw[141]},
[826]  423:  {"UNGLUECKLICH",ADJEKTIV,2,&aw[142],
[BL1]  424:  &aw[143]},
[SR6]  425:  {"TRAURIG",ADJEKTIV,2,&aw[142],&aw[143]},
[LT5]  426:  {"UNZUFRIEDEN",ADJEKTIV,2,&aw[142],
[BL1]  427:  &aw[143]},
[WK6]  428:  {"EINFACH",ADJEKTIV,2,&aw[171],&aw[172]},
[PH6]  429:  {"SCHWER",ADJEKTIV,2,&aw[171],&aw[172]},
[GF6]  43Ø:  {"LEICHT",ADJEKTIV,2,&aw[171],&aw[172]},
[9C5]  431:  {"SCHWIERIG",ADJEKTIV,2,&aw[171],
[GL1]  432:  &aw[172]},
[ØP2]  433:  /* Substantive  */
[VD6]  434:   {"WETTER",ONEWORD,2,&aw[144],&aw[145]},
[AZ5]  435:   {"FREUND",ONEWORD,3,&aw[146],&aw[147],
[ZL1]  436:   &aw[148]},
[U46]  437:   {"FREUNDE",ONEWORD,3,&aw[146],&aw[147],
[ZL1]  438:   &aw[148]},
[2L6]  439:   {"FREUNDIN",ONEWORD,3,&aw[146],&aw[147],
[ZL1]  44Ø:   &aw[148]},
[DA6]  441:   {"FEIND",ONEWORD,2,&aw[149],&aw[15Ø]},
[HA6]  442:   {"FEINDE",ONEWORD,2,&aw[149],&aw[15Ø]},
[Z96]  443:   {"GLUECK",ONEWORD,2,&aw[152],&aw[141]},
[CZ5]  444:   {"PECH",ONEWORD,2,&aw[151],&aw[143]},
[RY5]  445:   {"GELD",ONEWORD,2,&aw[153],&aw[154]},
[ZR6]  446:   {"REICHTUM",ONEWORD,2,&aw[153],&aw[154]},
[4D6]  447:   {"ZUFALL",ONEWORD,2,&aw[155],&aw[156]},
[M26]  448:   {"LUST",ONEWORD,2,&aw[196],&aw[197]},
[P36]  449:  {"HALLO",ONEWORD,3,&aw[157],&aw[158],
[6M1]  45Ø:  &aw[159]},
[G75]  451:  {"HI",ONEWORD,3,&aw[157],&aw[158],
[6M1]  452:  &aw[159]},
[8E5]  453:  {"TAG",ONEWORD,3,&aw[157],&aw[158],
[6M1]  454:  &aw[159]},
[WG5]  455:  {"TSCHUESS",ABSCHIED,1,&aw[193]},
[GØ6]  456:  {"WIEDERSEHEN",ABSCHIED,1,&aw[193]},
[BG4]  457:  {"ADE",ABSCHIED,1,&aw[193]},
[7M4]  458:  {"CIAO",ABSCHIED,1,&aw[193]},
[XZ1]  459:  {Ø,Ø,Ø,Ø,Ø});
[G14]  46Ø: Word *lang=&sprache[Ø];
[B32]  461: /* Personen */
[8M2]  462: typedef struct
[3S3]  463:  {char feel[SATZLEN];
[ØB4]  464:   char thoughts[SATZLEN];
[9L3]  465:   char will[SATZLEN];
[2S3]  466:   char future[SATZLEN];
[UP3]  467:   char should[SATZLEN];
[7H3]  468:   char need[SATZLEN];
[SG3]  469:   char have[SATZLEN];
[XD4]  47Ø:   char can[SATZLEN];}view;
[GA1]  471:  /* Info */
[8M2]  472: typedef struct
[432]  473:  {char *menge;
[YE2]  474:   char *andmenge;
[B22]  475:   int infnum;
[LM2]  476:   char **inf;}info;
[YF1]  477: /* Thema */
[3W2]  478: typedef struct {
[H12]  479: char *inhalt;
[TZ1]  48Ø: char *intro;
[A12]  481: info *first;
[UM1]  482: int type;
[VU3]  483: int (*action)();}thema;
[EK1]  484: /* Gehirn */
[3W2]  485: typedef struct {
[ØU1]  486: char *name;
[6X2]  487: float motivpart;
[TD2]  488: int motivnum;
[HZ1]  489: char **motiv;
[KT2]  49Ø: float viewpart;
[552]  491: view *myview;
[B62]  492: view *yoview;
[M83]  493: thema *thema;} mind;
[7H5]  494: int talk(char *name,char *partner,
       mind *pers);
[SF6]  496: /* newview(): Neu Ansichten des Partner
[SB1]  497:  merken */
[786]  498: newview(chg,word,zufall,abhier,str)
[5G1]  499: char *chg;
[CZ1]  5ØØ: Word *word;
[MU2]  5Ø1: char *abhier,*str;
[Ø32]  5Ø2: int zufall;
[NS3]  5Ø3:  {int savezuf=zufall;
[FK4]  5Ø4:   if(!strcmp(word->lexem,str))
[E33]  5Ø5:     {savezuf=zufall;
[6F2]  5Ø6:      if(word->num>1)
[944]  5Ø7:        {while(savezuf==zufall)
[694]  5Ø8:           zufall=random(word->num);
[PN5]  5Ø9:         sprintf(chg,*(word->ans[zufall])
[VC2]  51Ø:           ,abhier);}}}
[LK6]  511: /* themlen(): Ermittelt die Länge eines
[TU5]  512: Wortes in einer AND oder OR Menge */
[7X2]  513: int themlen(str)
[7L1]  514: char *str;
[TY1]  515:  {int len=Ø;
[CU5]  516:   while(*(str+len)!=',&&*(str+len)!='\Ø')
[HVØ]  517:         len++;
[Z62]  518:   return(len);}
[T37]  519: /* andif(): überprüft, ob in einem Satz die
```

```
[NL5]  52Ø: AND-Menge der Worte enthalten ist
[W33]  521:  Ø : Nicht enthalten
[A65]  522:  1 : Enthalten / keine AND-Menge */
[5S3]  523: int andinf(first,satz)
[1X1]  524: char *satz;
[A12]  525: info *first;
[3R2]  526:  {char *anf=satz;
[NM4]  527:   char *text=first->andmenge;
[AF1]  528:   int ret;
[5F4]  529:   if(strlen(text)<=1¦¦!text)
[LØ1]  53Ø:    ret=1;
[LP1]  531:   do{ret=Ø;
[4K1]  532:      satz=anf;
[2Ø2]  533:      while(*satz)
[YU2]  534:        {while(*satz==' ')
[P61]  535:           satz++;
[G96]  536:        if(!strncmp(text,satz,themlen(text)))
[MU2]  537:           {ret =1;break;}
[WN4]  538:         satz=satz + strlen(satz)+1;}
[H81]  539:       if(!ret)
[TL1]  54Ø:        return(Ø);
[5X3]  541:       text += themlen(text)+1;
[933]  542:     }while(*text!='\Ø');
[R72]  543:   return(ret);}
[S95]  544: /* srchinf(): Durchsucht Satz nach
[757]  545: OR-Menge und AND-Menge von Schlüsselworten
[6Ø5]  546: Rückgabe : AND Menge enthalten
[T44]  547:  ein Element der OR-Menge
[3B3]  548:  -1: Absicht erkannt
[3S4]  549:  1 : Frage oder Aussage erkannt
[L63]  55Ø:  Ø : Nichts entahlten
[HY4]  551:  Die Antwort ist in antwort */
[6C6]  552: int srchinf(char *antwort,info *first,
[YU1]  553:   char  *satz)
[S33]  554:  {info *save=first;
[PH2]  555:   char *anf=satz;
[ZW1]  556:   char *text;
[4B1]  557:   if(first)
[T93]  558:    while(first->menge)
[1U1]  559:     {satz=anf;
[2Ø2]  56Ø:     while(*satz)
[5B5]  561:     /* versuche Worte zu verstehen */
[YU2]  562:       {while(*satz==' ')
[P61]  563:          satz++;
[XA3]  564:        text = first->menge;
[NC4]  565:        do{if(!strncmp(text,satz,
[4K2]  566:                  themlen(text)))
[462]  567:             /* neue Info */
[S8Ø]  568:           {if(andinf(first,anf))
[Z25]  569:            {sprintf(antwort,*(first->inf
[RY3]  57Ø:             +(unsigned)((rand()/32767.Ø)
[N53]  571:             * first->infnum)));
[MH2]  572:            if(save==first)
[WT1]  573:             return(-1);
[ZRØ]  574:            else
[B72]  575:             return(1);}}
[5X3]  576:         text += themlen(text)+1;
[933]  577:       }while(*text!='\Ø');
[WN4]  578:     satz=satz + strlen(satz)+1;}
[CG1]  579:    first++;}
[D32]  58Ø:   return(Ø);}
[AM6]  581: /* srchtheme(satz): durchsucht Satz nach
[7Ø3]  582: Gesprächsthema */
[4W1]  583: char  t[1ØØ];
[XA6]  584: thema *srchtheme(mind *pers,char *satz,
[2R1]  585:   thema *akt)
[4B1]  586: {int i;
[ZW1]  587: char *text;
[1Ø1]  588: akt=Ø;
[2Ø2]  589: while(*satz)
[5B5]  59Ø: /* versuche Worte zu verstehen */
[YU2]  591:    {while(*satz==' ')
[P61]  592:       satz++;
[4KØ]  593:     i=Ø;
[HS4]  594:     while((pers->thema+i)->inhalt)
[R95]  595:      {text = (pers->thema+i)->inhalt;
[282]  596:       memset(t,Ø,9Ø);
[Ø85]  597:       strncpy(t,text,themlen(text));
[1VØ]  598:       do{
[Y66]  599:       if(!strncmp(text,satz,themlen(text))
[SH3]  6ØØ:        && (pers->thema+i)>akt)
[ØD2]  6Ø1:       /* neues Thema */
[943]  6Ø2:        akt=pers->thema+i;
[5X3]  6Ø3:        text += themlen(text)+1;
[933]  6Ø4:      }while(*text!='\Ø');
[MRØ]  6Ø5:      i++;}
[WN4]  6Ø6:    satz=satz + strlen(satz)+1;}
[K72]  6Ø7:   return(akt);}
[HP5]  6Ø8: /* sayview(): Ansicht generieren */
[EP4]  6Ø9: int sayview(antwort, pers)
[SA2]  61Ø: char *antwort;
[XX1]  611: mind *pers;
[E67]  612:  {int zufall=random(sizeof(view)/SATZLEN);
[RB2]  613:   if(random(2Ø)<1Ø)
[VY5]  614:     strcpy(antwort,pers->myview->feel
[DR3]  615:      +(int)(zufall*SATZLEN));
[ZRØ]  616:   else
[RY5]  617:     strcpy(antwort,pers->yoview->feel
[DR3]  618:     +(int)(zufall*SATZLEN));
[E32]  619:   return(1);}
[H67]  62Ø: /* saymotiv(): Motivationssatz generieren*/
[EV4]  621: int saymotiv(antwort, pers)
[SA2]  622: char *antwort;
[XX1]  623: mind *pers;
[DC5]  624:  {strcpy(antwort,pers->motiv[
[DC4]  625:   random(pers->motivnum)]);
[E32]  626:   return(1);}
[5B5]  627: /* saysth(): etwas sagen. Setzt Satz
[6Y5]  628: in Antwort. 1 als Rückgabe: gelungen
[392]  629: Ø: kein satz */
[WB4]  63Ø: int saysth(antwort, pers)
[SA2]  631: char *antwort;
[XX1]  632: mind *pers;
[Ø14]  633:  {int zufall=random(1ØØ);
[8P5]  634:   if(zufall<(int)(pers->viewpart*1ØØ))
[DD5]  635:     return(sayview(antwort, pers));
[ZRØ]  636:   else
[LL5]  637:    {if(zufall<(int)((pers->viewpart
[ML3]  638:     +pers->motivpart)*1ØØ))
[5L5]  639:      return(saymotiv(antwort, pers));
[ZRØ]  64Ø:    else
[A72]  641:      return(Ø);}}
[DS5]  642: /* split(): teilt Satz in Wörter auf
[E35]  643: hinter jedem Wort eine Ø. Hinter
[VV6]  644: letzter Doppelnull sind auch Wörter */
[PØ2]  645: split(satz)
[1X1]  646: char *satz;
[3Z2]  647:  {/* Alles groß */
[DA2]  648:   strupr(satz);
[LU3]  649:   while(*satz==' ') satz ++;
[2Ø2]  65Ø:   while(*satz)
[UB4]  651:    {if((*satz=='!'¦¦*satz=='.'
[9R4]  652:     ¦¦*satz=='?')&& *(satz-1)!='\Ø')
[F13]  653:      {*(satz+1)=*(satz);
[852]  654:       *(satz++)='\Ø';
[P61]  655:       satz++;
[FX1]  656:       *(satz)='\Ø';
[AG1]  657:       break;}
[Z52]  658:    if(*satz == ' ')
[HZ1]  659:     {*satz='\Ø';
[V51]  66Ø:      satz ++;
[7A4]  661:      while(*satz==' ') satz ++;}
```

»Talk.c« simuliert die Kommunikation mit verschiedenen Personen.

```
[ZRØ]  662:      else
[ND1]  663:        satz ++;)
[V51]  664:     satz ++;
[CW1]  665:     *satz = '\Ø';
[V51]  666:     satz ++;
[382]  667:     *satz = '\Ø';)
[V56]  668: /* findword(): Sucht Schlüsselwort in
[MF2]  669: einem String */
[GF2]  670: findword(word)
[TV1]  671: char *word;
[KG1]  672:  {int i=Ø;
[7L1]  673:   char *str;
[HT4]  674:   while((str=(lang+i)->lexem)!=Ø)
[DY3]  675:    {if(!strcmp(str,word))
[KM1]  676:       return(i);
[MRØ]  677:      i++;}
[TX1]  678:    return(-1);}
[7R6]  679: /* examine():Durchsucht Satz nach Wörter
[B47]  680: und Sätzen. anz = Anzahl der verstandenen
[AR6]  681: Wörter. Satzfolge enhält verstandene
[LN2]  682: Wortnummern. */
[8D7]  683: int examine(satz,satzfolge,importword,num)
[1X1]  684: char *satz;
[8R2]  685: int *satzfolge;
[TX2]  686: int *importword;
[1K1]  687: int *num;
[724]  688:  {int anz=Ø,count=Ø, help;
[YU5]  689: /* wichtigstes Wort falls kein Satz
[R53]  690: verstanden wurde */
[7R2]  691:   *importword = -1;
[H22]  692:   split(satz);
[2Ø2]  693:   while(*satz)
[5B5]  694:    /* versuche Worte zu verstehen */
[YU2]  695:    {while(*satz==' ')
[P61]  696:       satz++;
[LC1]  697:     if(*satz)
[1R3]  698:      {help=findword(satz);
[X65]  699:       if(help>=Ø) /*wort verstanden ? */
[3Ø4]  7ØØ:        {*satzfolge+anz)=help;
[D75]  7Ø1:         /* Suche nach wichtigstem wort */
[3V2]  7Ø2:        if(*importword<Ø)
[YY2]  7Ø3:          *importword=anz;
[ZRØ]  7Ø4:         else
[K83]  7Ø5:          if((lang+help)->imp>=
[K26]  7Ø6:       (lang+satzfolge[*importword])->imp)
[YY2]  7Ø7:            *importword=anz;
[RL1]  7Ø8:          anz++;})
[AB1]  7Ø9:     count ++;
[WN4]  71Ø:     satz=satz + strlen(satz)+1;}
[4H1]  711:    *num=anz;
[MW1]  712:    if(count>=2)
[RU5]  713:     /* Satz hat zwei Worte mindestens */
[WF5]  714:     if(anz)     /* Ein Wort verstanden ?*/
[UR2]  715:       return(ONEWORD);
[GU3]  716:    return(NIXVERSTEHN);}
[C96]  717: /* changesatz():  Verändert Satz für
[ZW1]  718: Antwort. */
[DR3]  719: changesatz(satz, save)
[7V2]  72Ø: char *satz, *save;
[2T2]  721:  {char *ptr=satz;
[2Ø2]  722:   while(*satz)
[924]  723:     {while(*satz==' ') satz++;
[BH6]  724:     if(*satz=='!'¦¦ *satz=='?'¦¦*satz=='.')
[CN3]  725:       *(save+((unsigned)satz
[2D3]  726:       -(unsigned)ptr))='\Ø';
[RP4]  727:     if(!strncmp(satz,"DEIN",4)¦¦
[B73]  728:     !strcmp(satz,"DICH")
[1S3]  729:     ¦¦!strcmp(satz,"DIR"))
[CN3]  73Ø:      *(save+((unsigned)satz
[M63]  731:      -(unsigned)ptr))='m';
[ZRØ]  732:     else
[F44]  733:       if(!strncmp(satz,"MEIN",4)
[SM3]  734:       ¦¦!strcmp(satz,"MIR")
[LW3]  735:       ¦¦!strcmp(satz,"MICH"))
[CN3]  736:        *(save+((unsigned)satz
[G53]  737:         -(unsigned)ptr))='d';
[RN3]  738:     if(!strcmp(satz,"ICH"))
[6Ø5]  739:      {memcpy((save+((unsigned)satz
[1C5]  74Ø:       -(unsigned)ptr)),(save+((unsigned)
[ØP3]  741:       satz-(unsigned)ptr))+1,
[KR4]  742:       strlen((save+((unsigned)satz
[A23]  743:       -(unsigned)ptr))+1);
[LP4]  744:       memcpy((save+((unsigned)satz
[MN3]  745:       -(unsigned)ptr)),"Du", 2);
[4A1]  746:       ptr ++;}
[ZRØ]  747:     else
[MH3]  748:       if(!strcmp(satz,"DU"))
[AB5]  749:        {memmove((save+((unsigned)satz
[DX2]  75Ø:         -(unsigned)ptr))+1,
[6E3]  751:         (save+((unsigned)satz
[CL2]  752:         -(unsigned)ptr)),
[KR4]  753:         strlen((save+((unsigned)satz
[A23]  754:         -(unsigned)ptr)))+1);
[LP4]  755:         memcpy((save+((unsigned)satz
[CY3]  756:         -(unsigned)ptr)),"ich", 3);
[EA1]  757:         ptr --;}
[PX4]  758:    satz=satz + strlen(satz)+1;})
[8E6]  759: /* changeverbs(): ändert die Verbform */
[ZZ3]  76Ø: changeverbs(satz, save)
[7V2]  761: char *satz, *save;
[2T2]  762:  {char *ptr=satz;
[G91]  763:   int i=Ø;
[XW1]  764:   int flag=Ø;
[CE2]  765:   while(*(save+i))
[UB5]  766:    {if(*(save+i)=='!'¦¦ *(save+i)=='?'
[ML2]  767:    ¦¦*(save+i)=='.')
[HB2]  768:      (*save+i)='\Ø';
[AG1]  769:        break;}
[MRØ]  77Ø:    i++;}
[2Ø2]  771:   while(*satz)
[924]  772:     {while(*satz==' ') satz++;
[1Z5]  773:     if(!flag) /*erstes Wort ich oder DU?*/
[S84]  774:       {if(!strncmp(satz,"ICH",3))
[U51]  775:          flag=3;
[1T3]  776:        if(!strncmp(satz,"DU",2)
[HE2]  777:        && *(satz+2)=='\Ø')
[R51]  778:          flag=2;
[A71]  779:        if(flag)
[6Ø5]  78Ø:         {memcpy((save+((unsigned)satz
[CL2]  781:           -(unsigned)ptr)),
[DE4]  782:           (save+((unsigned)satz+(flag)
[DX2]  783:           -(unsigned)ptr))+1,
[KR4]  784:           strlen((save+((unsigned)satz
[A23]  785:           -(unsigned)ptr)))+1);
[C43]  786:           ptr= ptr +(flag)+1;)
[H91]  787:         flag=1;}
[2L3]  788:     /* Verben verwandeln */
[HØ3]  789:     if(strlen(satz)>2&&
[6K3]  79Ø:     strncmp(satz,"DEIN",4)
[TR3]  791:     && strncmp(satz,"MEIN",4)
[BS3]  792:     && strncmp(satz,"SEIN",4)
[XZ3]  793:     && strncmp(satz,"KEIN",4)&&
[DF3]  794:      strncmp(satz,"EIN",3))
[EL5]  795:     {if(!strcmp(satz+strlen(satz)-2,
[BC3]  796:      "ST")&&strlen(satz)>3)
[5K4]  797:       {/*finde Anfang des Worts*/
[4K9]  798:        i=Ø;
[515]  799:        while(*(satz+strlen(satz)-2-i))
[UFØ]  8ØØ:          i++;
[NL4]  8Ø1:        if(islower(*(save+((unsigned)
[ER3]  8Ø2:        (satz+strlen(satz)-1-i)
[1R2]  8Ø3:        -(unsigned)ptr))))
[6Ø5]  8Ø4:         (memcpy((save+((unsigned)satz
[2B5]  8Ø5:       +(strlen(satz)-1)-(unsigned)ptr)),
[7K3]  8Ø6:         (save+((unsigned)satz+
[SH5]  8Ø7:       (strlen(satz)-1)-(unsigned)ptr))+1,
```

```
[KR4]  8Ø8:         strlen((save+((unsigned)satz
[TT5]  8Ø9:     +(strlen(satz)-1)-(unsigned)ptr)))+1);
[SU3]  81Ø:           *(save+((unsigned)satz+
[3U5]  811:       (strlen(satz)-2)-(unsigned)ptr))='e';
[YH1]  812:           ptr ++;})
[ZRØ]  813:     else
[T35]  814:      if(!strcmp(satz+strlen(satz)-1
[G63]  815:      ,"E")&&strlen(satz)>3)
[BVØ]  816:        {i=Ø;
[115]  817:         while(*(satz+strlen(satz)-1-i))
[UFØ]  818:           i++;
[NL4]  819:         if(islower(*(save+((unsigned)
[X86]  82Ø:    (satz+strlen(satz)-i)-(unsigned)ptr))))
[A85]  821:           {memmove((save+((unsigned)satz
[FK5]  822:       +(strlen(satz)-1)-(unsigned)ptr))+1,
[7K3]  823:           (save+((unsigned)satz+
[R75]  824:         (strlen(satz)-1)-(unsigned)ptr)),
[ZY3]  825:           strlen((save+((unsigned)
[CK6]  826: satz+(strlen(satz)-1)-(unsigned)ptr)))+1);
[C16]  827:  *(save+((unsigned)satz+(strlen(satz)
[ZS3]  828:  -2)-(unsigned)ptr+1))='s';
[ØE6]  829:  *(save+((unsigned)satz+(strlen(satz)-2)
[PG3]  83Ø:  -(unsigned)ptr+2))='t';
[ZX1]  831:           ptr --;})}
[PX4]  832:    satz=satz + strlen(satz)+1;})
[M77]  833: /* cleansatz():Reinigt Satz von Kommata und
[4W1]  834: Umlauten */
[ZN2]  835: cleansatz(satz)
[1X1]  836: char *satz;
[KG1]  837:  {int i=Ø;
[PG5]  838:   /* Kommata raus, ersetze äöü */
[TE2]  839:   while(*(satz+i))
[YØ3]  84Ø:    {switch(*(satz+i))
[HR1]  841:      {case '\n':
[DK1]  842:       case '\r':
[8S1]  843:        *(satz+i)=Ø;
[KC1]  844:        return;
[2L1]  845:       case 'ä':
[2L1]  846:       case 'Ä':
[KM1]  847:       case 'ü':
[KM1]  848:       case 'Ü':
[EM1]  849:       case 'ö':
[EM1]  85Ø:       case 'Ö':
[NØ4]  851:        memmove(satz+i+1,satz+i,
[WW2]  852:        strlen(satz+i)+2);
[ØP2]  853:        switch(*(satz+i))
[6H1]  854:         {case 'Ä':
[2L1]  855:          case 'ä':
[LA2]  856:            *(satz+ i++)='a';
[PØ1]  857:            break;
[KM1]  858:          case 'Ü':
[KM1]  859:          case 'ü':
[3D2]  86Ø:            *(satz+ i++)='u';
[PØ1]  861:            break;
[EM1]  862:          case 'Ö':
[EM1]  863:          case 'ö':
[BC2]  864:            *(satz+ i++)='o';
[AG1]  865:            break;}
[E72]  866:        *(satz+ i)='e';
[PØ1]  867:        break;
[U61]  868:       case ',':
[8Y1]  869:        *(satz+i)=' ';
[AG1]  87Ø:        break;}
[B31]  871:     i++;})
[X15]  872: /* getsatz(): erfragt Eingabe des
[VE4]  873: Kommmunikationpartners */
[AF5]  874: void getsatz(char *str,char *name)
[4B1]  875:  {int i;
[ZW1]  876:   char ch[2];
[6M2]  877:   char *satz=str;
[7B1]  878:   ch[1]=Ø;
[1U3]  879:   printinbox("%s: ",name);
[6L2]  88Ø:   strcpy(satz,"");
[LH3]  881:   while(kbhit())getch();
[P84]  882:   for(i=Ø;i<16*3Ø &&!kbhit();i++)
[ZF1]  883:    delay(62);
[4KØ]  884:    i=Ø;
[KM1]  885:   if(kbhit())
[TVØ]  886:    {do{
[8C2]  887:      ch[Ø]=getch();
[M32]  888:      switch(ch[Ø])
[TR1]  889:       {case '\r':
[9K1]  89Ø:        case '\n':
[PØ1]  891:         break;
[661]  892:        case 27:
[5C1]  893:         *satz=27;
[C23]  894:         printinbox("\n","");
[KC1]  895:         return;
[A51]  896:        case 8:
[XZØ]  897:         if(i>Ø)
[8TØ]  898:          {i--;
[211]  899:          str--;
[3A3]  9ØØ:          printinbox(ch,"");}
[PØ1]  9Ø1:         break;
[8E1]  9Ø2:        default:
[232]  9Ø3:         if(i<SATZLEN)
[YSØ]  9Ø4:          {i++;
[FU2]  9Ø5:           printinbox(ch,"");
[TM2]  9Ø6:           *str ++ = ch[Ø];}
[AG1]  9Ø7:         break;}
[T23]  9Ø8:      }while(ch[Ø]!='\r');
[4G1]  9Ø9:     *(str) = Ø;
[GZ2]  91Ø:     /* Eins anhängen */
[KZ2]  911:      strcat(satz," ");}
[ZRØ]  912:    else
[6L2]  913:     strcpy(satz,"");
[6A3]  914:    printinbox("\n","");}
[XV2]  915: nospace(char *str)
[PG3]  916:  {int l=strlen(str)-1;
[6L3]  917:   while(i>Ø & *(str+i)==' ')
[862]  918:    *(str+i--)=Ø;}
[GH5]  919: /* antworte(): Gibt Antwort aus */
[5W3]  92Ø: antworte(name,antwort)
[ØU1]  921: char *name;
[SA2]  922: char *antwort;
[484]  923:  {printinbox("%s: ", name);
[LX4]  924:   printinbox("%s\n", antwort);}
[R85]  925: /* talk(): Stellt ein Gespräch mit
[GG5]  926: bestimmten Gesprächspartner her.
[VP1]  927: Rückgabe:
[SG2]  928: -2: Analphabeth
[TD4]  929: -1: Normale Verabschiedung
[UC6]  93Ø: >Ø Spezielle Gesprächsbeendigungen */
[ØX4]  931: static char pname[SATZLEN];
[PU4]  932: static  char satz[SATZLEN];
[5K4]  933: static char save[SATZLEN];
[YS1]  934: static char antwort[2*SATZLEN];
[EG5]  935: static char antsave[2*SATZLEN];
[M95]  936: static char antzus[2*SATZLEN];
[V15]  937: static char einsave[SATZLEN];
[N95]  938: static int satzfolge[ANZWORD];
[5N3]  939: static char *anh[] = (
[4B2]  94Ø: " Uebrigens, ",
[P83]  941: " Nebenbei bemerkt, ",
[F82]  942: " Ausserdem, ",
[774]  943: " Davon einmal abgesehen, ",
[V43]  944: " Abgesehen davon, ",
[6Ø4]  945: " Was ich sagen wollte, ",
[VC4]  946: " Darüberhinaus gesagt, ",
[5P3]  947: " Noch etwas, und zwar ",
[4R3]  948: " Etwas ganz Anderes,  ",
[FV2]  949: " Anderes Thema, ",
[NØ3]  95Ø: " Aber Folgendes, ");
[RR4]  951: int talk(name,partner,pers)
[V73]  952: char  *name,*partner;
[GW1]  953: mind  *pers;
```

```
[SU3]  954:  {char *abhier,fertig=Ø;
[VS5]  955:   int nichtverstanden=Ø,noanswer=Ø;
[2H4]  956:   int nummer,anz=Ø, Art,act=Ø;
[2R4]  957:   int zufall, refreshthema=Ø;
[LH4]  958:   thema *aktthema=Ø,*neuthema;
[FV1]  959:   char *hstr;
[5Z2]  96Ø:   strcpy(antzus,"");
[P63]  961:   strcpy(antsave,"");
[PF5]  962:   printinbox("%s spricht mit",name);
[VH4]  963:   printinbox(" %s\n",partner);
[TT3]  964:   strcpy(pname,partner);
[472]  965:   while(!fertig)
[ØF3]  966:    {getsatz(satz,name);
[T83]  967:    strcpy(antwort,"");
[1X5]  968:    if(!strcmp(satz,"ciao ")¦¦*satz==27)
[WT1]  969:       return(-1);
[1N3]  97Ø:   nummer=Art=act=anz=Ø;
[3T2]  971:   if(!strlen(satz))
[233]  972:      Art=KEINEANTWORT;
[ZRØ]  973:   else
[V23]  974:   {cleansatz(satz);
[553]  975:    strcpy(save,satz);
[474]  976:    if(!strcmp(satz,einsave))
[133]  977:      Art=WIEDERHOLUNG;
[ZRØ]  978:    else
[X24]  979:     {strcpy(einsave,satz);
[N54]  98Ø:      /* Suche nach einem Wort */
[AH5]  981:      Art=examine(satz,&satzfolge[Ø],
[SA2]  982:                  &nummer,&anz);
[D42]  983:      if((neuthema=
[EK5]  984:      srchtheme(pers,satz,aktthema))!=Ø)
[GC4]  985:       {/* Neues Thema gefunden */
[3W2]  986:        /* Thema wechsel */
[7N4]  987:        if(neuthema->type==WICHTIG)
[NW3]  988:         {if(neuthema>aktthema)
[MV2]  989:           { /* Neues Thema */
[TS1]  99Ø:            Art=THEMA;
[4N2]  991:            refreshthema=2;
[W53]  992:            aktthema=neuthema;
[ØM3]  993:             /* Gibst einleitung ? */
[993]  994:            if(aktthema->intro)
[163]  995:             {sprintf(antwort,
[UY2]  996:               aktthema->intro);
[KW1]  997:              act = 2ØØ;}
[ZRØ]  998:            else
[RP3]  999:             { /*keine Einleitung */
[L64]  1ØØØ:             if((act=srchinf(antwort,
[364]  1ØØ1:              aktthema->first,satz))>Ø)
[TS1]  1ØØ2:              Art=THEMA;
[ZRØ]  1ØØ3:             else
[T91]  1ØØ4:              if(act<Ø)
[RZ2]  1ØØ5:                Art = ACTION;})
[235]  1ØØ6:          else/*altes neues Thema gef. */
[AV2]  1ØØ7:           (refreshthema=2;
[L64]  1ØØ8:           if((act=srchinf(antwort,
[364]  1ØØ9:            aktthema->first,satz))>Ø)
[TS1]  1Ø1Ø:              Art=THEMA;
[ZRØ]  1Ø11:            else
[T91]  1Ø12:             if(act<Ø)
[RZ2]  1Ø13:               Art = ACTION;})
[P34]  1Ø14:        else /* Thema unwichtig */
[F81]  1Ø15:         {act=Ø;
[DØ2]  1Ø16:          if(aktthema)
[TC4]  1Ø17:          {if((act=srchinf(antwort,
[364]  1Ø18:           aktthema->first,satz))>Ø)
[AV2]  1Ø19:           (refreshthema=2;
[D82]  1Ø2Ø:            Art=THEMA;)
[ZRØ]  1Ø21:           else
[T91]  1Ø22:            if(act<Ø)
[3G2]  1Ø23:              Art = ACTION;)
[971]  1Ø24:          if(!act)
[TC4]  1Ø25:           {if((act=srchinf(antwort,
[F64]  1Ø26:            neuthema->first,satz))>Ø)
[3S2]  1Ø27:             Art=NEBENTHEMA;
[ZRØ]  1Ø28:           else
[BL1]  1Ø29:            if(act --<Ø)
[M33]  1Ø3Ø:              Art = ACTION;}))
[XE5]  1Ø31:    else /* Altes Thema beibehalten */
[HH4]  1Ø32:     if(aktthema && refreshthema)
[TC4]  1Ø33:      {if((act=srchinf(antwort,
[364]  1Ø34:         aktthema->first,satz))>Ø)
[AV2]  1Ø35:         (refreshthema=2;
[D82]  1Ø36:          Art=THEMA;)
[ZRØ]  1Ø37:       else
[T91]  1Ø38:        if(act<Ø)
[3G2]  1Ø39:          Art = ACTION;)
[VH6]  1Ø4Ø:   /* Thema löscht sich aus Gedächtins */
[ZS2]  1Ø41:   if(!refreshthema)
[TM2]  1Ø42:    aktthema = NULL;
[ZRØ]  1Ø43:   else
[NL2]  1Ø44:    refreshthema--;
[XA3]  1Ø45:   /* Keine Themafrage */
[L41]  1Ø46:   if(anz)
[N54]  1Ø47:    (changesatz(satz,save);
[673]  1Ø48:    strcpy(satz,save);
[DA2]  1Ø49:    strupr(satz);
[L41]  1Ø5Ø:    if(anz)
[MX5]  1Ø51:     abhier=save+(unsigned)strstr(satz,
[LS5]  1Ø52:      (lang+satzfolge[nummer])->lexem)-
[9P4]  1Ø53:      (unsigned)satz+strlen((lang+
[6U4]  1Ø54:      satzfolge[nummer])->lexem);
[ZRØ]  1Ø55:     else
[B22]  1Ø56:      abhier=save;
[744]  1Ø57:     while(*abhier==' ')abhier++;
[H22]  1Ø58:     split(satz);
[275]  1Ø59:     changeverbs(satz+(abhier-save),
[G81]  1Ø6Ø:      abhier);
[1T3]  1Ø61:     nospace(abhier);}))
[ØM5]  1Ø62:  /* Neues Thema aber nix verstehn */
[MG3]  1Ø63:  if(Art==THEMA && act==Ø)
[R62]  1Ø64:    Art=ONEWORD;
[6X1]  1Ø65:  switch(Art)
[173]  1Ø66:   {case NIXVERSTEHN:
[Y55]  1Ø67:     nummer = 175 + nichtverstanden ++;
[785]  1Ø68:     sprintf(antwort,aw[nummer]);
[PØ1]  1Ø69:     break;
[653]  1Ø7Ø:    case KEINEANTWORT:
[Ø34]  1Ø71:     nummer = 181 + noanswer ++;
[73]   1Ø72:     sprintf(antwort,aw[nummer]);
[PØ1]  1Ø73:     break;
[Z82]  1Ø74:    case ONEWORD:
[SX1]  1Ø75:    case ICHDU:
[8E1]  1Ø76:    default:
[L41]  1Ø77:     if(anz)
[WN3]  1Ø78:      ( /* Ein Wort erkannt */
[XV4]  1Ø79:       nichtverstanden=noanswer=Ø;
[7X5]  1Ø8Ø:       if((lang+satzfolge[Ø])->imp==FRAGE)
[PH1]  1Ø81:        nummer=Ø;
[ZØ6]  1Ø82:       if((lang+satzfolge[nummer])->num>Ø)
[FD5]  1Ø83:        (hstr=(lang+satzfolge[nummer])
[RB1]  1Ø84:          ->lexem;
[RY4]  1Ø85:         zufall=random((lang+satzfolge
[YP2]  1Ø86:         [nummer])->num);
[L83]  1Ø87:         if(strlen(abhier)<=2)
[V14]  1Ø88:          (if(strcmp(hstr,"GEBE")&&
[G93]  1Ø89:            strcmp(hstr,"GEHE")&&
[YS3]  1Ø9Ø:            strcmp(hstr,"BLEIBE")&&
[VH3]  1Ø91:            strcmp(hstr,"KOMME")&&
[H63]  1Ø92:            strcmp(hstr,"WAR")&&
[XD3]  1Ø93:            strcmp(hstr,"WAERE"))
[P32]  1Ø94:          abhier="das";
[ZRØ]  1Ø95:        else
[242]  1Ø96:          abhier="\b";)
[US5]  1Ø97:        sprintf(antwort,*((lang+satzfolge
[RG5]  1Ø98:        [nummer])->ans[zufall]),abhier);
[NH5]  1Ø99:        if(saysth(antzus,pers)&&strstr(*((
```

```
[F84] 11Ø0:        lang+satzfolge[nummer])
[835] 11Ø1:        ->ans[zufall]),"%s")==NULL && *(*((
[W55] 11Ø2:        lang+satzfolge[nummer])->ans[
[824] 11Ø3:        zufall])+strlen(*((lang+
[3Y5] 11Ø4:        satzfolge[nummer])->ans[zufall]))
[PØ1] 11Ø5:        -1)!='?')
[LB3] 11Ø6:         (zufall=random(11);
[MZ4] 11Ø7:          strcat(antwort,anh[zufall]);
[HE4] 11Ø8:          strcat(antwort, antzus);)
[AA3] 11Ø9:        zufall=random((lang+
[7D4] 111Ø:        satzfolge[nummer])->num);
[EU4] 1111:   /* Merken von Informationen */
[LK5] 1112:       newview(pers->yoview->feel,lang+
[MB7] 1113: satzfolge[nummer],zufall,abhier,"FUEHLE");
[2E6] 1114:       newview(pers->yoview->thoughts,lang+
[PØ7] 1115: satzfolge[nummer],zufall,abhier,"DENKE");
[YL5] 1116:       newview(pers->yoview->will,lang
[ØX6] 1117: +satzfolge[nummer],zufall,abhier,"WILL");
[RØ6] 1118:       newview(pers->yoview->future,lang+
[527] 1119: satzfolge[nummer],zufall,abhier,"WERDE");
[VØ6] 112Ø:       newview(pers->yoview->should,lang+
[UX6] 1121: satzfolge[nummer],zufall,abhier,"SOLL");
[VK5] 1122:       newview(pers->yoview->need,lang+
[EH7] 1123: satzfolge[nummer],zufall,abhier,"BRAUCHE");
[4L5] 1124:       newview(pers->yoview->have,lang+
[9U6] 1125: satzfolge[nummer],zufall,abhier,"HABE");
[ME5] 1126:       newview(pers->yoview->can,lang+
[QV6] 1127: satzfolge[nummer],zufall,abhier,"KANN");
[VT4] 1128:       if((lang+satzfolge[nummer])
[WL2] 1129:         ->imp==ABSCHIED)
[642] 113Ø:           fertig=-1;)
[ZRØ] 1131:       else
[EX3] 1132:         if(saysth(antzus,pers))
[HB3] 1133:          (zufall=random(1Ø);
[MZ4] 1134:           strcat(antwort,anh[zufall]);
[8U4] 1135:           strcat(antwort, antzus);))
[ZRØ] 1136:       else
[2T4] 1137:         while(!saysth(antwort,pers));
[PØ1] 1138:       break;
[F63] 1139:    case WIEDERHOLUNG:
[FY6] 114Ø: sprintf(antwort,"Du wiederholst Dich.");
[PØ1] 1141:      break;
[ØY1] 1142:    case THEMA:
[NS2] 1143:    case NEBENTHEMA:
[PØ1] 1144:      break;
[E12] 1145:    case ACTION:
[AC6] 1146:  if((act==-1?aktthema:neuthema)->action)
[N64] 1147:       fertig=((act==-1?aktthema:
[Ø34] 1148:               neuthema)->action)(pers);
[AG1] 1149:      break;)
[YR4] 115Ø:   if(!strcmp(antwort,antsave))
[213] 1151: sprintf(antwort,
[PM5] 1152: "Ich sagte schon: \"%s\".",antsave);
[H84] 1153:    antworte(pname,antwort);
[ZD4] 1154:    strcpy(antsave, antwort);
[2D4] 1155:    /* auf motivation absuchen */
[UZ2] 1156:    if(strlen(antzus))
[T32] 1157:     {Art=THEMA;
[F74] 1158:      strcpy(antwort,antzus);
[TE3] 1159:      strcpy(antzus,"");)
[KD2] 116Ø:    if(Art == THEMA)
[6M3] 1161:     (cleansatz(antwort);
[ØD5] 1162:      examine(antwort,&satzfolge[Ø],
[SA2] 1163:                      &nummer,&anz);
[LU4] 1164:      if((neuthema=srchtheme(pers,
[7L3] 1165:                antwort,aktthema))!=Ø)
[GC4] 1166:       (/* Neues Thema gefunden */
[3W2] 1167:        /* Thema wechsel */
[7N4] 1168:        if(neuthema->type==WICHTIG)
[NW3] 1169:         (if(neuthema>aktthema)
[AV2] 117Ø:          (refreshthema=2;
[YC4] 1171:           aktthema=neuthema;))))
[XX5] 1172:    if(nichtverstanden>5¦¦noanswer>2)
[VX1] 1173:      return(-2);)
[RS2] 1174:   return(fertig);)
[NZØ] 1175: #endif
[CM3] 1176: static char *endl[]={
[DM6] 1177: "Das muss ich mir doch nicht anhören! Hau"
[KC6] 1178: " ab !","Sowas muss ich mir nicht bieten "
[8W1] 1179: "lassen!", };
[1T3] 118Ø: static info aus1[] = {
[ZB4] 1181: "DU,DICH,",",",2,end1,Ø,Ø,Ø,Ø,Ø;
[ZVØ] 1182: ciao()
[U82] 1183:  {return(-1);}
[P92] 1184: /* mind dovan */
[C64] 1185: static char *womainf[]={
[S75] 1186:  "Es gibt einen Geheimgang in der "
[CY5] 1187:  "Nordwand aus dieser Zelle. Er endet "
[NB5] 1188:  "vor einer Geheimtuer, die in den "
[3X5] 1189:  "Waffengang fuehrt.", "Unsere beiden "
[6D5] 119Ø:  "Zellen sind durch eine geheime "
[DH5] 1191:  "Tuer verbunden. Auch aus meiner "
[ZC5] 1192:  "Zelle fuehrt eine solche Tuer in "
[856] 1193:  "der Nordwand.", "Es gibt einen Ausweg "
[LV5] 1194:  "von hier, durch einen Geheimgang in "
[YV2] 1195:  "der Nordwand.",);
[CW4] 1196: static info geheiminf[] = {
[V56] 1197:  "GEHEIMTUER,GEHEIMGANG,GEHEIMGAENGE,"
[B96] 1198:  "AUSWEG,AUSGANG,RAUS,","WO",",2,womainf,
[VE4] 1199:  "WEG,","?,",2,womainf,Ø,Ø,Ø,Ø,Ø;
[XV3] 12ØØ: static char *mainf[]={
[HA6] 12Ø1:  "Ich weiss nicht, wo meine Tochter ist, "
[Y86] 12Ø2:  "vielleicht in einer andere Zelle.",);
[554] 12Ø3: static char *frmainf[]={
[837] 12Ø4:  "Miriam ist neunzehn oder zwanzig Jahre, "
[7P3] 12Ø5:  "fast noch ein Kind.");
[HX3] 12Ø6: static char *mainf[]={
[4T6] 12Ø7: "Mademoiselle Miriam ist der Name meiner "
[PØ2] 12Ø8: "Tochter.",);
[YX3] 12Ø9: static char *huinf[]={
[C15] 121Ø:   "Aber sicher ist meine Tochter "
[123] 1211:   "wunderschoen.",);
[B74] 1212: static char *hulinf[]={
[HE5] 1213:   "Meine Tochter Miriam ist von den "
[FV5] 1214:   "Untoten irgendwohin verschleppt "
[B42] 1215:   "worden.",);
[RØ4] 1216: static info mirinf[] = {
[NV4] 1217: "TOCHTER,MIRIAM,","","",1,hulinf,
[ML5] 1218: "TOCHTER,MIRIAM,SIE,","WO,",1,mainf,
[356] 1219: "SIE,TOCHTER,MIRIAM,","ALT,",1,frmainf,
[U35] 122Ø: "SIE,TOCHTER,","HEISST,",1,mninf,
[FD6] 1221: "SIE,TOCHTER,MIRIAM,","HUEBSCH",1,huinf,
[6A6] 1222: "SIE,TOCHTER,MIRIAM,","SCHOEN",1,huinf,
[L41] 1223: Ø,Ø,Ø;
[SN3] 1224: static char *iinf[]={
[F76] 1225:   "Ich bin Graf Dovan von Maskavan, der "
[VV4] 1226:   "Schlossbesitzer und Du ?",);
[GE3] 1227: neuername(mind *pers)
[BL4] 1228:  (strcpy(pname,pers->name);
[D32] 1229:   return(Ø);)
[554] 123Ø: static char *maginf[]={
[EW5] 1231:  "Ich weiss nicht, um welchen Magier "
[Y73] 1232:  "es sich handelt.",);
[254] 1233: static char *ichinf[]={
[LS5] 1234:   "Ich weiss nicht, wie Dein Name ist "
[4R4] 1235:   "edler Ritter, jedoch bist Du "
[RR6] 1236: "hoffentlich der Retter meiner Tochter.",
[BDØ] 1237: };
[3N3] 1238: static char *binf[]={
[YV6] 1239:  "Es gibt eine Bibliothek im Nordostteil,"
[YØ6] 124Ø:  " die Wohn- und Schlafräume sind im "
[X76] 1241:  "Suedosten.","Das Verliess, in dem wir "
[SØ5] 1242:  "hier sind, ist im Westeil des "
[RP4] 1243:  "Schlosses, edler Ritter.",);
[GV3] 1244: static char *hinf[]= {
[NE6] 1245:   "Ich bin von den unheimlichen Untoten "
[ØV4] 1246:   "gefangen genommen worden.",);
[2R3] 1247: static char *winf[]={
[CV5] 1248:   "Dies ist Schloss Maskavan im Lande "
[AR3] 1249:   "Koenigs Armax II.", };
[PV3] 125Ø: static char *alsk[]= {
[1K5] 1251:   "Diese Untoten laufen ueberall im "
[6S5] 1252:   "Schloss herum.","Die Untoten sind "
[4S5] 1253:   "Skelette und haben alle Bewohner "
[764] 1254:   "gefangen oder getoetet.",
[VH6] 1255:   "Hoffentlich haben die Untoten Skelette"
[415] 1256:    " meine Tochter verschont.",);
[AU3] 1257: static char *alsk1[]={
[D16] 1258:   "Wahrscheinlich hat sie ein boesen "
[776] 1259:   "Schwarzmagier zu Leben erweckt.",);
[VW3] 126Ø: static char *wainf[]={
[1V5] 1261:  "Du bekommst eine angemessene hohe "
[ØB6] 1262: "Belohnung und die Hand meiner Tochter.",
[BDØ] 1263:  };
[VF4] 1264: static info skellinf{} = {
[A75] 1265: "SKELETTE,UNTOTE,SIE,UNTOTEN,"
[RP2] 1266:       ,"WOHER,",1,alsk1,
[V26] 1267: "SKELETTE,UNTOTE,UNTOTEN,","","",3,alsk,
[731] 1268: Ø,Ø,Ø,Ø,Ø;
[SD4] 1269: static info name2inf[] = {
[3W3] 127Ø: "WER,","WER,BIST,",1,iinf,
[324] 1271: "WER,","BIN,ICH,",1,ichinf,
[NG4] 1272: "IST,","WER,MAGIER,",1,maginf,
[9W5] 1273: "IST,","WER,SCHWARZMAGIER,",1,maginf,
[EW6] 1274: "SKELETTE,UNTOTE,UNTOTEN,","WER,",1,alsk1,
[731] 1275: Ø,Ø,Ø,Ø;
[1F6] 1276: static char *mkom[]={
[WP2] 1278:  "Ich bin hier angekettet, ich kann nicht "
[P64] 1279: static info schlinf[] = {
[V64] 128Ø: "ZIMMER,RAEUME,","","",2,binf,
[UL5] 1281: "WIESO,WARUM,WESHALB,","DU,",1,hinf,
[6Y3] 1282: "WAS,","MACHST,DU,",1,hinf,
[HG4] 1283: "WAS,","BEKOMME,ICH,",1,wainf,
[PK4] 1284: "KOMMST,KOMM,","MIT,",1,mkom,
[B14] 1285: "WIE,","KOMMST,DU,",1,hinf,
[CN5] 1286: "SKELETTE,UNTOTE,","WOHER,",1,alsk1,
[913] 1287: "HIER,","WO,",1,winf,
[AS3] 1288: "SIND,","WIR,WO,",1,winf,
[9Ø7] 1289: "GEHEIMGANG,GEHEIMGAENGE,AUSWEG,AUSGANG,"
[L55] 129Ø: "RAUS,GEHEIMTUER,","?,",2,womainf,
[ML5] 1291: "TOCHTER,MIRIAM,SIE,","WO,",1,mainf,
[731] 1292: Ø,Ø,Ø,Ø,Ø;
[HØ4] 1293: static info kominf[] = {
[PK4] 1294: "KOMMST,KOMM,","MIT,",1,mkom,
[9Ø7] 1295: "GEHEIMGANG,GEHEIMGAENGE,AUSWEG,AUSGANG,"
[ØR5] 1296: "RAUS,GEHEIMTUER,","WIE,",2,womainf,
[B14] 1297: "WIE,","KOMMST,DU,",1,hinf,
[731] 1298: Ø,Ø,Ø,Ø,Ø;
[L64] 1299: static info nameinf[] = {
[VB3] 13ØØ: "HEISST,","DU,",1,iinf,
[3W3] 13Ø1: "WER,","WER,BIST,",1,iinf,
[P44] 13Ø2: "TOCHTER,","NAME,",1,mninf,
[E33] 13Ø3: "WIE,","NAME,",1,iinf,
[UC4] 13Ø4: "HEISST,","TOCHTER,",1,mninf,
[7B4] 13Ø5: "HEISST,","MAGIER,",1,maginf,
[NG4] 13Ø6: "IST,","WER,MAGIER,",1,maginf,
[HS5] 13Ø7: "HEISST,","SCHWARZMAGIER,",1,maginf,
[9W5] 13Ø8: "IST,","WER,SCHWARZMAGIER,",1,maginf,
[EW6] 13Ø9: "SKELETTE,UNTOTE,UNTOTEN,","WER,",1,alsk1,
[731] 131Ø: Ø,Ø,Ø,Ø,Ø;
[YY3] 1311: static char *matinf[] ={
[MK6] 1312:  "Matthias ist der Spieledesigner, er hat"
[285] 1313:  " hier jedes Pixel geschaffen.",);
[TS3] 1314: static info mati[] = {
[PL3] 1315: "MATTHIAS,","","",1,matinf,
[731] 1316: Ø,Ø,Ø,Ø,Ø;
[R14] 1317: static char *bainf[]= {
[HG5] 1318:   "Ich warte auf Dich, versuche auch "
[LH4] 1319:   "meine Tochter zu finden !",);
[T44] 132Ø: static info backinf[] = {
[ØW3] 1321: "ZURUECK,","ICH,",1,bainf,
[731] 1322: Ø,Ø,Ø,Ø,Ø;
[4X3] 1323: static char *bsinf[]={
[3L6] 1324:   "Den Schlüssel zur Bibliothek habe ich, "
[VL5] 1325: "aber ich gebe ihn dir nach unserem "
[YL2] 1326: "Gespraech.", );
[G34] 1327: static char *bsinf2[]={
[PT5] 1328:   "Den Schluessel habe ich bei mir.",);
[M34] 1329: static char *bsinf3[]={
[BR5] 133Ø:   "Es gibt viele Schluessel, welchen "
[EG2] 1331:   "meinst Du ?", );
[Z44] 1332: static char *ziinf[]= {
[XG5] 1333:   "Das Geheimzimmer befindet in der "
[SØ5] 1334:   "hintersten Ecke bei dem Buch "
[VA3] 1335:   ">Simple Reime<",);
[1Y3] 1336: static info bibinf[] = {
[NM5] 1337: "BIBLIOTHEK,","SCHLUESSEL,",1,bsinf,
[TZ4] 1338: "BIBLIOTHEK,","ZIMMER,",1,ziinf,
[XE4] 1339: "ZIMMER,RAUM,","WO,",1,ziinf,
[WZ3] 134Ø: "BIBLIOTHEK,","WO,",1,binf,
[731] 1341: Ø,Ø,Ø,Ø,Ø;
[314] 1342: static info keyinf[] = {
[CG3] 1343: "GIB,GEBEN,","",",1,bsinf,
[LZ2] 1344: "SCHLUESSEL,DER,",
[W34] 1345: "WO,BIBLIOTHEK,",1,bsinf2,
[125] 1346: "WIE,WO,","SCHLUESSEL,",1,bsinf3,
[UH5] 1347: "ZUR,FUER,","BIBLIOTHEK,",1,bsinf2,
[BA1] 1348: Ø,Ø,Ø,Ø,Ø,;
[R54] 1349: static char *schwinf[]={
[BE6] 135Ø:   "Am Ende des Geheimgangs im Waffengang"
[TW3] 1351:   " liegen Schwerter.",);
[UL4] 1352: static char *schwminf[]= {
[UD6] 1353:   "Es soll ein magisches Schwert hinter "
[HN6] 1354: "dem >Schwarzen Tor< eingemauert sein.",
[BDØ] 1355:  );
[W74] 1356: static info swinf[] = {
[KV4] 1357: "MAGISCH,MAGISCHES,MAGISCHE,",
[5G3] 1358: "SCHWERT,",1,schwminf,
[KV4] 1359: "MAGISCH,MAGISCHES,MAGISCHE,",
[N33] 136Ø: "WAFFE,",1,schwminf,
[Z96] 1361: "WAFFE,WAFFEN,SCHWERT,","WO,",1,schwinf,
[731] 1362: Ø,Ø,Ø,Ø,Ø;
[7L3] 1363: static char *tol[]= {
[5B6] 1364:   "Das >Schwarze Tor< ist eine schwarze "
[XZ3] 1365:   "Wand im Geheimgang.",);
[6U3] 1366: static info torf[] = {
[P82] 1367: "TOR,","","",1,tol,
[731] 1368: Ø,Ø,Ø,Ø,Ø;
[NK3] 1369: static char *no1[]= {
[GY5] 137Ø:   "Die Wand gegenueber der Tuer liegt "
[HM2] 1371:   "im Norden.",);
[UT3] 1372: static info norf[] = {
[Z25] 1373: "NORD,NORDEN,NORDWAND,","","",1,no1,
[731] 1374: Ø,Ø,Ø,Ø,Ø;
[QZ1] 1375: #ifndef GAME
[RU4] 1376: gibschluessel(){return(Ø);}
[TZ3] 1377: showende(){return(Ø);}
[2TØ] 1378: #else
[614] 1379: extern gibschluessel();
[D23] 138Ø: extern showende();
[NZØ] 1381: #endif
[BM4] 1382: static thema akothemen[] = {
[SR5] 1383:  ("WESHALB,WO,WARUM,WIESO,WAS,?,",Ø,
[KL3] 1384:   schlinf,UNWICHTIG,Ø),
[ND5] 1385:  ("MATTHIAS,",Ø,mati,UNWICHTIG,Ø),
[LY5] 1386:  ("HEISST,NAME,",Ø,nameinf,UNWICHTIG,
[M42] 1387:  neuername),
[FU6] 1388:  ("WER,",Ø,name2inf,UNWICHTIG,neuername},
[DP6] 1389:  ("IDIOT,ARSCH,TROTTEL,DUMM,IDIOT,DEPP,"
[3A5] 139Ø:  "BLOED,",Ø,aus1,UNWICHTIG,ciao),
[SA6] 1391:  ("KOMMST,KOMM,",Ø,kominf,UNWICHTIG,Ø),
[2R6] 1392:  ("UNTOTE,SKELETTE,UNTOTEN,SKELETTEN,",
[W64] 1393:  Ø,skellinf,UNWICHTIG,Ø),
[TB5] 1394:  ("WAFFEN,WAFFE,SCHWERT,",Ø,swinf,
[R62] 1395:  UNWICHTIG,Ø),
[R17] 1396:  ("GEGRUESST,HALLO,TAG,","Sei gegrüßt "
[256] 1397:  "edler Ritter. Wie kommst Du hierher ?",
[712] 1398:  Ø,WICHTIG,Ø),
[U95] 1399:  ("ZURUECK,",Ø,backinf,WICHTIG,Ø),
[EØ6] 14ØØ:  ("RETTE,BEFREIE,","Ich bin nicht ohne "
[KY5] 14Ø1:  " Hilfsmittel von meinen Ketten zu "
[YV4] 14Ø2:  "befreien, ich warte auf deine "
[E44] 14Ø3:  "Rueckkehr.",Ø,WICHTIG,Ø),
[DP6] 14Ø4:  ("VERLIESS,ZELLE,KERKER,","Das hier ist "
[WS5] 14Ø5:  "eine der drei Kerkerzellen meines "
[BL5] 14Ø6:  "Schlosses.",schlinf,WICHTIG,Ø),
[XA7] 14Ø7:  ("ANFUEHRER,VERANTWORTLICH,","Die Untoten
",
[MM5] 14Ø8:  "sind magisch am Leben, aber wieso "
[EN4] 14Ø9:  "weiss niemand.",Ø,WICHTIG,Ø),
[7X5] 141Ø:  ("TOR,BARRIERE,",Ø,torf,WICHTIG,Ø),
[GN6] 1411:  ("MIRIAM,TOCHTER,",Ø,mirinf,WICHTIG,Ø),
[CN7] 1412:  ("GEHEIMGANG,GEHEIMGAENGE,AUSGANG,WEG,AUSW
EG,"
[N76] 1413:  "GEHEIM,GANG,RAUS,HERAUS,GEHEIMTUER,"
[AG6] 1414:  ,"Das Schloss ist von Geheimgängen und "
[U76] 1415:  "geheimen Zimmern, durch die Du gehen "
[KB5] 1416:  "kannst.",geheiminf,WICHTIG,Ø),
[N87] 1417:  ("NORD,NORDEN,NORDWAND,",Ø,norf,WICHTIG,Ø)
,
[H75] 1418:  ("BIBLIOTHEK,",Ø,bibinf,WICHTIG,
[UV2] 1419:  gibschluessel),
[3D5] 142Ø:  ("SCHLUESSEL,",Ø,keyinf,WICHTIG,
[UV2] 1421:  gibschluessel),
[TS1] 1422:  (Ø,Ø,Ø,),);
[5Y3] 1423: static view akoview ={
[U47] 1424: "Ich fühle, daß es mit mir zu Ende geht.",
[RV6] 1425: "Ich denke manchmal, an die glücklichen "
[ZM6] 1426: "Tage zurück.","Ich will noch einmal das"
[AM6] 1427:  Licht der Sonne erblicken.","Wir werden"
[XD6] 1428:  sterben.","Du bist endlich ein Mensch "
[Ø66] 1429: "und keiner von diese Untoten.","Wir "
[8D5] 143Ø: "brauchen Hilfe.","Wir haben keine "
[UL5] 1431: "Chance mehr.","Wir koennen Ihnen "
[1A3] 1432: "nicht entkommen.",);
[XP3] 1433: static view akoask = {
[1S5] 1434: "Fuehlst Du auch jene beklemmende "
[A22] 1435: "Spannung ?",
[XR6] 1436: "Denkst Du, dass Du uns retten kannst ?",
[TF6] 1437: "Wie willst Du uns retten ?","Wirst Du "
[SP5] 1438: "einen Weg finden um mich und meine "
[586] 1439: "Tochter zu retten ?","Du sollst eine "
[ØL5] 144Ø: "angemessene Belohnung erhalten.",
[TL5] 1441: "Brauchst Du noch etwas von mir?",
[ML6] 1442: "Hast Du eine Idee, die uns retten kann?",
[EF4] 1443: "Du kannst mir helfen ?", );
[1P4] 1444: static char *akomotiv[] ={
[9L6] 1445: "Die Zeit wird knapp. Nur du kannst uns "
[8C2] 1446: "noch retten!",
[PN5] 1447: "Bitte rette meine Tochter Miriam !",
[996] 1448: "Wir sind nur hier, weil mein Schloss "
[9S4] 1449: "von Untoten belagert wird.",
[K56] 145Ø: "Wir sind in meinem eigenen Schloss "
[VX4] 1451: "gefangen. Was sagst du dazu ?",
[PZ5] 1452: "Ich muss mein Tochter befreien, die "
[S14] 1453: "verschleppt wurde!",);
[9B2] 1454: mind dovan = {
[865] 1455: "Graf Dovan",.1,5,&akomotiv[Ø],.2,
[7X4] 1456: &akoview,&akoask,akothemen);
[EX2] 1457: /**** MIND MIRIAM ***/
[Y54] 1458: static view mirview = {
[9Ø7] 1459: "Ich fühle, dass Du mich total anmachst. "
[WN6] 146Ø: "Abenteurer","Ich denke gerade über Dich "
[8F4] 1461: "nach.","Ich will Dich näher "
[ØE6] 1462: "kennenlernen.","Ich wuerde gerne mehr "
[7E6] 1463: "über Dich erfahren.","Ich sollte nicht "
[EM6] 1464: "in deiner Nähe sein.","Ich brauche Dich "
[2M6] 1465: "nur anzuschauen und meine Knie werden "
[WL5] 1466: "schwach.","Ich habe das Gefühl Du "
[VX6] 1467: "gefällst mir. Abenteurer","Ich kann mich "
[T16] 1468: "kaum zurückhalten, Dich zu ...", );
[1H3] 1469: static view mirask = {
[KE5] 147Ø:   "Fuehlst Du auch diese Spannung "
[W86] 1471:   "zwischen uns ?","Denkst Du manchmal, "
[RY3] 1472:   "ich durchschaue Dich ?",
[ZW5] 1473:  "Willst Du mit mir auf den nächsten "
[V76] 1474:  "Ball gehen ?","Wuerdest Du mich gerne "
[ZC5] 1475:  "kuessen ?","Warum sollte ich Dir "
[CY5] 1476:  "glauben ?","Du brauchst mir nichts "
[KE5] 1477:  "zu verheimlichen !","Was hast Du "
[XK4] 1478:  "gerade über mich gedacht ?",
[GB5] 1479:  "Du kannst mir deine geheimsten "
[EF4] 148Ø:  "Wünsche erzählen ?", );
[ØR4] 1481: static char *mirmotiv[] ={
[F75] 1482: "Warum schaust Du mir so tief in "
[9M6] 1483: "meine Augen ?","Auf Schloss Maskavan ist"
[CH5] 1484:  ein Festball. Kommst Du vorbei ?",
[FY4] 1485: "Ich gebe am 24. November einen "
[1H3] 1486: "Festball. Komm doch !",
[LØ7] 1487: "Fühlst Du auch diese Spannung, wenn Du "
[BH4] 1488: "mir in die Augen schaust?",
[HU3] 1489: "Was denkst du jetzt ?",
[T94] 149Ø: "Was hast Du mit mir vor ?",
[3X6] 1491: "Wenn Du mit mir tanzen willst, dann komm"
[164] 1492:  doch auf meinen Festball",
[356] 1493: "Wuerdest Du gerne auf einen Festball "
[GE6] 1494: "gehen ?","Du wirst mir zu gefährlich, "
[5Ø7] 1495: "mein Abenteurer.","Ich weiss nicht warum,"
[ØF6] 1496:  aber Du imponierst mir!","Wenn du mich "
[XK7] 1497: "wiedersehen willst, komm doch zum Festball
,"
[FU6] 1498: "Besuch mich doch mal, auf unserm Schloss!"
,
[976] 1499: "Kommst Du mich auch mal besuchen?",);
[YL3] 15ØØ: static char *nam2[]={
[NU3] 1501: "Der Name gefällt mir.",
[WW3] 1502: "Der Name passt zu Dir.",
[CT6] 1503: "Interessanter Name, woher kommt er?",);
[5L3] 1504: static char *ahal[]={
[HH6] 1505: "Interessant. Das dachte ich mir schon.",
[GK5] 1506: "Ja, das hab ich mir vorgestellt.",
[UP5] 1507: "Ach ja wirklich? Hätte ich nicht "
[2N1] 1508: "gedacht.",
[D85] 1509: "Echt? Naja, doch das kann ich mir "
[BB2] 151Ø: "vorstellen.",
[NL3] 1511: "Ja, denke ich auch.");
[FP3] 1512: static info ich1[] = {
[DB3] 1513: "HEISSE,","ICH,",3,nam2,
[HV2] 1514: "ALT,","ICH,",5,ahal,
[3H5] 1515: "NAME,","MEIN,IST,",3,nam2,
[731] 1516: Ø,Ø,Ø,Ø,Ø;
[SB3] 1517: static char *gl[]={
[YM6] 1518: "Mir geht es heute ausgezeichnet. Danke.",
[F96] 1519: "Ich fühle mich ganz gut. Wie geht es "
[GG6] 152Ø: "Dir ?","Ich fühle mich nicht schlecht. "
[HB4] 1521: "Wie fühlst Du Dich?",);
[644] 1522: static info geht1[] = {
[HP2] 1523: "WIE,","GEHT,",3,gl,
[UZ2] 1524: "GEHT,","MIR,",4,ahal,
[731] 1525: Ø,Ø,Ø,Ø,Ø;
[NV3] 1526: static char *wirf1[]={
[LL6] 1527: "Fände ich nicht schlecht, aber das geht"
[P16] 1528: " nicht jetzt.","Ich wuerde ja gerne, "
```

»Talk.c« simuliert die Kommunikation mit verschiedenen Personen.

```
[LB6] 1529: "aber nicht heute.","Eigentlich haette "
[UN5] 153Ø: "ich schon Lust, aber das geht doch "
[PX1] 1531: "nicht.",);
[5T3] 1532: static info wir1[] = {
[W54] 1533: "WIR,UNS,","GEHEN",3,wirf1,
[FD4] 1534: "WIR,UNS,","WOLLEN",3,wirf1,
[731] 1535: Ø,Ø,Ø,Ø);
[ØD3] 1536: static info se1[] = {
[9D3] 1537: "DU,DIR,","SEX,",2,end1,
[3D5] 1538: "WILL,","ICH,SCHLAFEN,DIR,",3,wirf1,
[GW4] 1539: "WILL,","KUESSEN,DICH,",3,wirf1,
[731] 154Ø: Ø,Ø,Ø,Ø);
[YC3] 1541: static char *nam[]={
[9L6] 1542: "Mein Name ist Mylady Graefin Miriam von "
[NØ2] 1543: "Maskavan.");
[V14] 1544: static char *alt1[] = {
[LT6] 1545: "Ich bin neunzehn Lenze jung und du ?");
[KD3] 1546: static info du1[] = {
[FR3] 1547: "HEISST,","WIE,DU,",1,nam,
[BH3] 1548: "BIST,","WER,DU,",1,nam,
[5P3] 1549: "NAME,","WIE,DEIN,",1,nam,
[VY2] 155Ø: "ALT,","WIE,",1,alt1,
[L86] 1551: "WILLST,MOECHTEST,","DU,MICH,",3,wirf1,
[D56] 1552: "WILLST,MOECHTEST,","DU,MIR,",3,wirf1,
[731] 1553: Ø,Ø,Ø,Ø);
[GYØ] 1554: goon()
[U42] 1555:  {return(1);}
[NB3] 1556: static char *f1[]={
[2K5] 1557: "Ich freue mich schon, auf Dich.");
[TB3] 1558: static char *f2[]={
[4Y5] 1559: "Auf unserem Schloss Maskavan am 24 "
[8V3] 156Ø: "November. Kommst du ?");
[G54] 1561: static info fest1[] = {
[C16] 1562: "WILL,MOECHTE,WERDE,ICH,","KOMM,",1,f1,
[2P3] 1563: "STATT,IST,?,","WO,",1,f2,
[Ø24] 1564: "STATT,IST,?,","WANN,",1,f2,
[YU3] 1565: "FESTBALL,BALL,","?,",1,f2,
[EH3] 1566: "BESUCHEN,","WILL,",1,f2,
[2M2] 1567: ",","BESUCHE,",1,f2,
[731] 1568: Ø,Ø,Ø,Ø);
[BD3] 1569: static char *uf1[]={
[FC6] 157Ø: "Wenn Du nicht mal weisst, was Du mit mir"
[4C6] 1571: " sprechen sollst, geh heim, Kleiner!");
[Ø73] 1572: static info u1[] = {
[7L4] 1573: "UNTERHALT,SPRECHEN,REDEN,"
[FØ5] 1574: ,"WORUEBER,",1,uf1,"REDEN,SAGEN,"
[RZ3] 1575: ,"SOLL,WAS,",1,uf1,Ø,Ø,Ø,Ø);
[DC3] 1576: static char *fl[]={
[956] 1577: "Du schmeichelst mir und Du weisst, "
[GR3] 1578: "daß das mir gefaellt!",
[NN5] 1579: "Danke, schoen für das Kompliment.",
[UB6] 158Ø: "Du Schmeichler. Ich warne Dich, werde "
[WC6] 1581: "nicht unverschaemt!","Aber nicht doch, "
[5L4] 1582: "ich werde ganz verlegen.",);
[BS3] 1583: static info kom1[] = {
[GB6] 1584: "SCHOEN,ATTRAKTIV,GEIL,HUEBSCH,SUPER,"
[H22] 1585: ,"HAST,",4,kf1,
[GB6] 1586: "SCHOEN,ATTRAKTIV,GEIL,HUEBSCH,SUPER,"
[ER2] 1587: ,"GEFALLEN,",4,kf1,
[GB6] 1588: "SCHOEN,ATTRAKTIV,GEIL,HUEBSCH,SUPER,"
[122] 1589: ,"BIST,",4,kf1,
[WR3] 159Ø: "MIR,","GEFAELLST,",4,kf1,
[1K3] 1591: "MIR,","GEFALLEN,",4,kf1,
[M93] 1592: ",","FINDE,DEIN,",4,kf1,
[K83] 1593: ",","FINDE,DICH,",4,kf1,
[R83] 1594: ",","LIEBE,DICH,",4,kf1,
[731] 1595: Ø,Ø,Ø,Ø);
[AM4] 1596: static thema mirthemen[] = {
[ND5] 1597:  ("MATTHIAS,",Ø,mat1,UNWICHTIG,Ø),
[3H6] 1598:  ("DU,DEIN,",Ø,du1,UNWICHTIG,neuername),
[RU4] 1599:  ("GEHT,",Ø,geht1,UNWICHTIG,Ø),
[455] 16ØØ:  ("ICH,MEIN,",Ø,ich1,UNWICHTIG,Ø),
[NB5] 16Ø1:  ("WIR,UNS,",Ø,wir1,UNWICHTIG,Ø),
[9M5] 16Ø2:  ("UNTERHALT,SPRECHEN,REDEN,",Ø,u1,
[613] 16Ø3:  UNWICHTIG,ciao),
[UU5] 16Ø4:  ("FINDE,GEFAELLST,HAST,LIEBE,BIST,",
[SC3] 16Ø5:  Ø,kom1,UNWICHTIG,Ø),
[DP6] 16Ø6:  ("IDIOT,ARSCH,TROTTEL,DUMM,IDIOT,DEPP,"
[3A5] 16Ø7:  "BLOED,",Ø,aus1,UNWICHTIG,ciao),
[Z36] 16Ø8:  ("GEGRUESST,HALLO,TAG,","Hallo, mein "
[1H4] 16Ø9:  "Abenteurer.",Ø,WICHTIG,Ø),
[R36] 161Ø:  ("SEX,KUESSEN,",Ø,sel,WICHTIG,ciao),
[AT6] 1611:  ("FESTBALL,BALL,",Ø,fest1,WICHTIG,goon),
[S65] 1612:  ("BESUCH,",Ø,fest1,WICHTIG,goon),
[TS1] 1613:  (Ø,Ø,Ø),);
[E83] 1614: static char *hr[]= {
[4Ø5] 1615:  "Ja, ich auch mein Abenteuer." );
[G64] 1616: static info geirinf[] = {
[2R4] 1617: "WILL,","HEIRAT,",1,hr,Ø,Ø,Ø,Ø);
[464] 1618: static info liebinf[] = {
[2Ø5] 1619: "LIEBE,","ICH,DICH,",1,hr,Ø,Ø,Ø,Ø);
[XR2] 162Ø: mind miriam = {
[WV5] 1621: "Mylady Miram",..3,13,&mirmotiv[Ø],.2,
[TØ5] 1622: &mirview,&mirask,mirthemen);
[KW4] 1623: static char *mirmotiv2[] ={
[PG5] 1624: "Ich freue mich, das Du hier bist.",
[M26] 1625: "Ich koennte mich an Dich gewoehnen.",
[A14] 1626: "Ich habe Dich vermisst.",
[ES5] 1627: "Ich glaube, ich habe mich verliebt "
[6M1] 1628: "in Dich!",
[686] 1629: "Was denkst Du über unsere Zukunft?",
[Y45] 163Ø: "Was hältst Du von heiraten?",
[XV4] 1631: "Was denkst Du über die Ehe?",
[BL3] 1632: "Bist Du verletzt?",);
[PT4] 1633: static thema mirthemen2[] = {
[ND5] 1634:  ("MATTHIAS,",Ø,mat1,UNWICHTIG,Ø),
[3H6] 1635:  ("DU,DEIN,",Ø,du1,UNWICHTIG,neuername),
[RU4] 1636:  ("GEHT,",Ø,geht1,UNWICHTIG,Ø),
[455] 1637:  ("ICH,MEIN,",Ø,ich1,UNWICHTIG,Ø),
[NB5] 1638:  ("WIR,UNS,",Ø,wir1,UNWICHTIG,Ø),
[816] 1639:  ("FINDE,GEFAELLST,HAST,BIST,",Ø,kom1,
[R62] 164Ø:  UNWICHTIG,Ø),
[DP6] 1641:  ("IDIOT,ARSCH,TROTTEL,DUMM,IDIOT,DEPP,"
[3A5] 1642:  "BLOED,",Ø,aus1,UNWICHTIG,ciao),
[2F6] 1643:  ("GEGRUESST,HALLO,TAG,","Endlich, mein "
[1H4] 1644:  "Abenteurer.",Ø,WICHTIG,Ø),
[WR5] 1645:  ("RETTE,RETTEN,BEFREIE,BEFREIEN,",
[396] 1646:  "Ich bin Dir sehr dankbar, dass Du mich "
[BE4] 1647:  "befreit hast.",Ø,WICHTIG,Ø),
[Y46] 1648:  ("WER,",Ø,name2inf,WICHTIG,neuername),
[895] 1649:  ("SEX,KUESSEN,",Ø,sel,WICHTIG,Ø),
[ER6] 165Ø:  ("HEIRAT,",Ø,heirinf,WICHTIG,showende),
[2D6] 1651:  ("LIEBE,",Ø,liebinf,WICHTIG,showende),
[TS1] 1652:  (Ø,Ø,Ø),);
[VH2] 1653: mind miriam2 = {
[5X5] 1654: "Mylady Miram",..2,8,&mirmotiv2[Ø],.3,
[5X4] 1655: &mirview,&mirask,mirthemen2);
[ØZ1] 1656: #ifndef GAME
[ØN4] 1657: main(int argc, char *argv[])
[2D1] 1658:  {mind *m;
[7Y1] 1659:   randomize();
[5F1] 166Ø:   if(argc>1)
[5X3] 1661:   {switch(atoi(*(argv+1)))
[NM1] 1662:    {default:
[N41] 1663:     case 1:
[NC1] 1664:       m=&dovan;
[P91] 1665:       break;
[S41] 1666:     case 2:
[EL1] 1667:       m=&miriam;
[PØ1] 1668:       break;
[V41] 1669:     case 3:
[ZT1] 167Ø:       m=&miriam2;
[AG1] 1671:       break;}
[D14] 1672:   talk("Du",m->name,m);}}
[NZØ] 1673: #endif
```

»Talk.c« simuliert die Kommunikation mit verschiedenen Personen.

Soundprogrammierung

Musik aus dem Hintergrund

Wie die Grafik tragen auch Musik und Geräuschkulisse wesentlich zur Atmosphäre eines Spiels bei. In diesem Beitrag lernen Sie eine Methode kennen, die es Ihnen erlaubt, für Ihre Spiele durch ein parallel laufendes Programm Soundeffekte im Hintergrund abzuspielen.

Jeder, der ein Spiel schreibt, kennt das Problem: Die Grafik funktioniert, das Spiel läuft, nur der Sound fehlt noch. Wie kann man den Sound entsprechend dem Spiel ablaufen lassen, ohne den Spielfluß zu verzögern? Falls beispielsweise in einem Weltraumspiel ein Raumschiff in einem Kampf abgeschossen wird und der Spieler jedesmal auf das Ende einer langen Siegermelodie wartet, wird er sehr bald die Musik abschalten.

Dazu gibt es drei alternative Lösungsvorschläge. Zum einen verwenden Sie sehr kurze Effekte, so daß die Wartezeit kurz, aber informativ ist. Oder Sie programmieren einen periodischen Aufruf einer Routine, die die Verwaltung der Soundeffekte übernimmt. Der dritte Weg, der hier beschrieben wird, ist die vollständige Trennung von Soundverwaltung und Spiel. Dabei läuft das Programm zur Soundsteuerung im Hintergrund parallel zum Spielprogramm als zweites Task. Über eine Art Briefkasten teilt das Spieleprogramm der Soundverwaltung mit, welche Musik oder Effekte gespielt werden sollen. Dieses Verfahren bringt viele Vorteile. Zum Beispiel brauchen Sie bei der Entwicklung eines Spiels keine Rücksicht mehr auf die Soundprogrammierung nehmen. Denn da die Verwaltung der Sounddaten ein völlig anderes Programm mit eigenem Daten- und Stacksegment ist, kommt es in diesen Bereichen zu keinerlei Platzproblemen. Auch können beide Programme unabhängig entwickelt und erweitert werden. Außerdem ist eine externe Soundverwaltung auch bei weiteren Spielen ohne große Änderungen einsetzbar, so daß Sie viel Zeit sparen.

Durch Parallelität von Programmen zur Musik

Das hier vorgestellte Beispiel stellt ein Konzept dar, das eine einfache, vom eigentlichen Spiel getrennte Verwaltung von Musikeffekten erlaubt, die parallel, aber dennoch vom Spiel gesteuert abgespielt werden. Dabei wird zwar der – in jedem PC vorhandene – Lautsprecher benutzt, jedoch ist dasselbe Konzept auch auf Soundkarten anwendbar. Zunächst erfahren Sie nun, wie Sie eine parallel ablaufende Soundverwaltung realisieren. Die Arbeit des Prozessors Ihres PC wird ständig von Interrupts unterbrochen. Deren Aufgabe besteht, vereinfacht gesagt, in der Verarbeitung extern auftretender Ereignisse. Wenn Sie also auf

Befehl	Funktion
L<n>	definiert die Länge aller folgenden Noten bis zum nächsten Kommando »L«
Buchstaben a, b, c, d, e, f, g	Töne
+ –	erhöht oder erniedrigt vorhergehende Note um einen Halbton
.	punktiert eine Note
N<n>	spielt Note mit Nummer n
P<n>	Pause von Länge <n>; P8 entspricht einer Pause von einer Achtelnote
MN	Töne haben ihre volle Länge
MS	Töne haben Dreiviertel der vollen Länge
O<n>	legt die Oktave für die folgenden Noten fest
>	erhöht die aktuelle Oktave um 1
<	verringert die aktuelle Oktave um 1

Befehle für Soundzeichenketten

eine Taste Ihres Rechners drücken, wird ein Interrupt ausgelöst, der den Tastaturcode vom Tastaturcontroller in den Tastaturpuffer überträgt. Für dieses Projekt ist in diesem Zusammenhang der Timer-Interrupt von Interesse, der periodisch 18,2mal in der Sekunde auftritt. Er ist dafür verantwortlich, die interne Uhr Ihres Rechners weiterzuzählen oder die Motoren der Diskettenlaufwerke wieder abzuschalten, wenn Lese- oder Schreibvorgänge beendet sind. Durch ein »Einhängen« in diesen Interrupt lassen Sie bei jedem Aufruf der Interrupt-Routine ein eigenes Programm mit ausführen. Dieses Verfahren, das üblicherweise alle TSR-Programme (Terminate Stay Resident) anwenden, wird in diesem Fall zweifach verwendet. Erstens wird der sogenannte »Briefkasten« überwacht, ob ein neuer Befehl zum Abspielen eines Soundeffekts erwartet wird. Zweitens werden aktuelle Soundeffekte gestartet, abgespielt und schließlich beendet. Diese beiden Funktionen arbeiten auch unabhängig voneinander.

Hier lernen Sie das Funktionsprinzip dieses Soundverwaltungsprogramms kennen. Nachdem das Soundverwaltungsprogramm installiert ist, wartet es auf einen »Brief« in Form einer Zahl an einer bestimmten Speicherzelle »MUSPORT« im Hauptspeicher. Wenn der Inhalt der Speicherzelle 0 ist, bedeutet das, daß kein »Brief« da ist. Schreibt nun Ihr Spieleprogramm durch einen Befehl der Form

```
MUSPORT=<Zahl>;
```

einen »Brief« in Form einer Zahl, so spielt das Soundverwaltungsprogramm das Musikstück ab, das dieser Zahl zugeordnet ist. Dieses Abspielen erfolgt dabei völlig im Hintergrund und beeinflußt das Spieleprogramm in keiner Weise. Im Programm »mus.c« (Listing 1) geben Sie die Tonhöhen und Tonlängen der Musikstücke in Zeichenketten an, deren Format größtenteils dem Play-Befehl von GW-Basic entspricht (Tabelle). Das Programm »usemus.c« (Listing 2) demonstriert Ihnen die Funktionen der vorgestellten Soundverwaltung. Der Aufruf der Programme lautet dazu

```
mus usemus
```

Das hier vorgestellte Werkzeug findet auch bei dem in dieser Ausgabe enthaltenenen Adventure-Rollenspiel Verwendung und versetzt Sie in die Lage, sehr schnell und vor allem nachträglich Ihr Spiel mit eigenen Soundeffekten auszustatten. Die Übersetzung von »mus.c« erfolgt im Modell »Tiny« des Turbo-C-Compilers, wobei der Schalter »Test Stack Overflow« auf »OFF« stehen muß.

(Matthias Oberdorfer/bm)

```
[W6Ø]   1: /*
[UM2]   2:  Programm: mus.c
[EE7]   3:  Funktion: ferngesteuerte Hintergrundmusik
[K83]   4:  Sprache:  Turbo C 2.Ø
[KR4]   5:  Autor:    Matthias Oberdorfer
[HN4]   6:  (c) 1992 DMV Widuch GmbH & Co. KG
[ØØØ]   7:
[FB4]   8:  Compilieren im TINY-Model
[TB4]   9:  TESTSTACK OVERFLOW auf OFF
[17Ø]  1Ø: */
[ØØØ]  11:
[7M2]  12: #include<dos.h>
[2T2]  13: #include<alloc.h>
[XU2]  14: #include<ctype.h>
[5Z3]  15: static char *songs[]=(
[PB2]  16: /*  kommissar*/
[TW4]  17: "o@ms18el8el8g-18el8gl8el8al8g",
[6Ø2]  18:   /*  oldsong */
[ØR5]  19: "o@ms14el8l8bl4gl8bl8bl4g-l8bl8bl8f"
[CP1]  2Ø: "l8fl8bl8b",
[353]  21:  /* mysterious hoch*/
[546]  22: "o6ml12bl2b-l2gl2g-l2bl2b-l2gl2g-l4bl4b-"
[Ø76]  23: "l4gl4g-l4bl4b-l4gl4g-l8bl8b-l8gl8g-l8bl"
[WT1]  24: "8b-l8gl8g-",
[VK1]  25:   /* bolero */
[G36]  26: "o2l4>c.l8c<l16b>l16c<l16d>c<ba l8>cl16c",
[K21]  27:  /* tot */
[ØG6]  28: "o1mSl4al4al8al4al4>c<l8bl4bl8al4al8a-l4a"
[GB3]  29: , /*  Dreiklang hoch */
[X95]  3Ø: "o2l16a>d-e", /* Dreiklang runter */
[ØR4]  31: "o3l16ed-<a",/* Kloing */"o4l16a",
[7C4]  32: /* Kloing */"o4l16c",/*heirat*/
[3C6]  33: "o4ms14el4al8al2al4el4bl8g+l2al8el4a>l4c+"
[6C6]  34: "l8el4c+<l4al4f+l4bl4g+l2ap2l4el4al8al2al"
[G76]  35: "4el4bl8g+l2al8el4a>l4c+l8el4c+<l4al4f+l4"
[AV1]  36: "bl4g+l2a",);
[NA3]  37: typedef struct tone /*Note*/  {
[BF2]  38: unsigned freq;
[HB3]  39: unsigned int dauer;
[GW4]  4Ø: struct tone far *next; }tone;
[D54]  41: tone far *mus_actu=NULL;
[6U4]  42: tone far *end=NULL, far *help;
[S24]  43: void far *reserved=NULL;
[363]  44: int hintergrund=Ø;
[7M4]  45: void interrupt (*org)(void);
[2A6]  46: #define MUSPORT *((unsigned far *)Øx18Ø)
[PT5]  47: /* void music(): spielt Noten einer
[UØ7]  48: Warteschlange im Hintergrund ab und freed
[ZW3]  49: die abgespielte Note */
[7M2]  5Ø: #include<dos.h>
[WN2]  51: unsigned flag=Ø;
[165]  52: unsigned oldss,oldsp,cs,sp,ss;
[CZ3]  53: #define STACKSIZE   Øx2ØØØ
[4U4]  54: unsigned stack[STACKSIZE];
[KG4]  55: void interrupt music(void)
[DX1]  56:   (disable();
[784]  57:   __emit__(Øx9e,/*PUSH CS*/
[GL2]  58:   Øx1f);/* POP DS */
[VK1]  59:   if(!flag++)
[A24]  6Ø:     {oldsp=_SP; oldss=_SS;
[LP2]  61:       cs = _DS; _SS=cs;
[DD5]  62:       _SP=FP_OFF(&stack[STACKSIZE]);
[M92]  63:       org(); enable();
[G52]  64:       if(mus_actu)
[VA3]  65:       /* Note vorhanden ? */
[UX3]  66:         (if(!mus_actu->dauer --)
[432]  67:            ( /*vorbei ?*/
[HT3]  68:              help = mus_actu->next;
[363]  69:              farfree(mus_actu);
[PY2]  7Ø:              mus_actu = help;)
[ZRØ]  71:            else
[Z94]  72:              sound(mus_actu->freq);)
[ZRØ]  73:         else
[D12]  74:          (end =NULL;
[LR1]  75:          nosound();
[N13]  76:          /* Überwache Port */
[7Y3]  77:          if(MUSPORT)
[XA4]  78:            ( /* starte neue Sounds */
[C64]  79:             PLAY(songs[MUSPORT-1]);
[SE2]  8Ø:             MUSPORT=Ø;))
[RM1]  81:         disable();
[Ø14]  82:         _SS=oldss; _SP=oldsp;)
[BB1]  83:         flag --;)
[867]  84: /* inst_mus() : installiert den Interrupt-
[NR6]  85: handler zum Abspielen im Hintergrund */
[2G2]  86: int inst_mus()
[TU3]  87:   (reserved = (void far *)
[4F4]  88:   farmalloc(sizeof(tone)*1ØØ);
[B13]  89:   org = getvect(Øx Ø8);
[F12]  9Ø:   MUSPORT =Ø;
[AT3]  91:   setvect(Øx Ø8, music);)
[Y26]  92: /* end_mus() : setzt Interruptvektor
[KR4]  93: auf alte Adresse zurück. */
[K88]  94: #define end_mus() (setvect(Øx Ø8, org);nosou
           nd();)
[4C6]  95: /* int SOUND(freq, dauer); : spielt einen
[134]  96: Ton mit freq und Dauer */
[9N2]  97: tone far *noten;
[VP3]  98: int SOUND(freq, dauer)
[9K1]  99:   int freq;
[HB3] 1ØØ:   unsigned int dauer;
[8S3] 1Ø1:   (if((noten = (tone far *)
[XB5] 1Ø2:   farmalloc(sizeof(tone))) == NULL)
[5B2] 1Ø3:     (if(reserved)
[H43] 1Ø4:     farfree(reserved);
[FK2] 1Ø5:     reserved=NULL;
[D32] 1Ø6:     return(Ø);)
[383] 1Ø7:   noten->freq = freq;
[WB3] 1Ø8:   noten->dauer = dauer;
[SA3] 1Ø9:   noten->next = NULL;
[A52] 11Ø:   if( mus_actu)
[NA3] 111:     (end->next = noten;
[972] 112:      end = noten;)
[ZRØ] 113:   else
[CG3] 114:     mus_actu =end= noten;
[D32] 115:   return(Ø);)
[V35] 116: /* int PLAY(s): spielt Noten nach
[FG4] 117:  übergebener Befehlskette.
[UV4] 118:  s: String mit Musikbefehlen */
[HB4] 119:  #define neu while(*s==' ')s++
[UY5] 12Ø:  int note[9] = (6272,7Ø4Ø,79Ø4,4192,47Ø4,
[3N2] 121:                 528Ø,5584,6272,7Ø4Ø);
[YS5] 122:  int l_note[13] = (4192,4448,47Ø4,4992,
[855] 123:  528Ø,5584,5928,6272,6656,7Ø4Ø,7472,79Ø4);
[X72] 124:  int octave=4;
[344] 125:  unsigned int not_len = 8;
[6A1] 126:  int l=Ø;
[N82] 127:  int pause = 8;
[VH3] 128:       int ton,ton2, oct_h;
[DU2] 129:       unsigned int nr;
[UX1] 13Ø:  int PLAY(s)
[T71] 131:       char *s;
[CY1] 132:       (octave=4 ;
[7G1] 133:        pause = Ø;
[5A1] 134:        while(*s)
[2XØ] 135:          (neu;
[U93] 136:           switch(toupper(*s++))
[TE1] 137:             (case 'A':
[K91] 138:              case 'B':
[P91] 139:              case 'C':
[U91] 14Ø:              case 'D':
[Y91] 141:              case 'E':
[2A1] 142:              case 'F':
[6A1] 143:              case 'G':
[TF3] 144:                nr = toupper(*(s-1))-'Ø';
```

```
[URØ] 145:                 neu;
[5K5] 146:       ton = ton2 = note[nr]>>(6-octave);
[H13] 147:       if(*s=='+' || *s =='#')
[WTØ] 148:       (s++;
[1G1] 149:       if(nr == 2)
[FU4] 15Ø:         ton2 = note[nr+1]>>(7-octave);
[ZRØ] 151:       else
[DU4] 152:         ton2 = note[nr+1]>>(6-octave);
[YG3] 153:         ton += (ton2 - ton)>>1;)
[GE2] 154:       else if(*s == '-')
[WTØ] 155:         (s++;
[4G1] 156:         if(nr == 3)
[5U4] 157:           ton2 = note[nr-1]<<(5-octave);
[ZRØ] 158:         else
[NU4] 159:           ton2 = note[nr-1]>>(6-octave);
[1P3] 16Ø:           ton -= (ton-ton2)>>1;)
[URØ] 161:         neu;
[342] 162:         nr = not_len;
[RX1] 163:         while(*s=='.')
[Ø12] 164:           (nr += nr>>1;
[2G1] 165:           s++;neu;)
[SD1] 166:         if(pause)
[P7Ø] 167:           (
[ST5] 168: SOUND(ton, (nr<<2)/l/pause*(pause-1));
[5Y4] 169: SOUND(32ØØØ, (nr<<2)/l/pause);)
[ZRØ] 17Ø:           else
[8F3] 171:             SOUND(ton, (nr<<2)/l );
[PØ1] 172:             break;
[6B1] 173:           case 'O':
[URØ] 174:             neu;
[4N2] 175:             octave = *s++ - 'Ø';
[PØ1] 176:             break;
[291] 177:           case '>':
[EV1] 178:             if(octave<6)
[ØE1] 179:               octave++;
[PØ1] 18Ø:             break;
[U81] 181:           case '<':
[WU1] 182:             if(octave>Ø)
[AE1] 183:               octave--;
[PØ1] 184:             break;
[2B1] 185:           case 'N':
[URØ] 186:             neu;
[3Y1] 187:             nr = *s++ - 'Ø';
[7A2] 188:             if(isdigit(*s))
[ZØ3] 189:    nr = 1Ø*nr + (*s++ - 'Ø');
[HVØ] 19Ø:             if(nr)
[YE5] 191: ton = l_note[(nr-1)%12]>>((nr-1)/12);
[ZRØ] 192:           else
[GN1] 193:             ton = 32ØØØ;
[URØ] 194:             neu;
[342] 195:             nr = not_len;
[RX1] 196:             while(*s=='.')
[Ø12] 197:               (nr += nr>>1;
[2G1] 198:               s++;neu;)
[SD1] 199:             if(pause)
[P7Ø] 2ØØ:               (
[WY5] 2Ø1: SOUND(ton, (nr<<2)/l/pause*(pause-1) );
[5Y4] 2Ø2: SOUND(32ØØØ, (nr<<2)/l/pause);)
[ZRØ] 2Ø3:                 else
[8F3] 2Ø4:                   SOUND(ton, (nr<<2)/l );
[PØ1] 2Ø5:                   break;
[UA1] 2Ø6:           case 'L':
[URØ] 2Ø7:             neu;
[ZU1] 2Ø8:             l = *s++ - 'Ø';
[7A2] 2Ø9:             if(isdigit(*s))
[1N2] 21Ø:             l = 1Ø*l + (*s++ - 'Ø');
[PØ1] 211:             break;
[AB1] 212:           case 'P':
[URØ] 213:             neu;
[DL2] 214:             oct_h = *s++ - 'Ø';
[7A2] 215:             if(isdigit(*s))
[S74] 216:         oct_h = 1Ø*oct_h + (*s++ - 'Ø');
[URØ] 217:             neu;
[342] 218:             nr = not_len;
[RX1] 219:             while(*s=='.')
[Ø12] 22Ø:               (nr += nr>>1;
[EGØ] 221:               s++;
[MZØ] 222:               neu;)
[B44] 223:             SOUND(32ØØØ, (nr<<2)/oct_h);
[PØ1] 224:             break;
[YA1] 225:           case 'M':
[URØ] 226:             neu;
[U93] 227:             switch(toupper(*s++))
[YE1] 228:               (case 'F':
[YL2] 229:                 hintergrund = Ø;
[PØ1] 23Ø:                 break;
[K91] 231:               case 'B':
[2M2] 232:                 hintergrund = 1;
[PØ1] 233:                 break;
[2B1] 234:               case 'N':
[9H1] 235:                 pause = 8;
[PØ1] 236:                 break;
[UA1] 237:               case 'L':
[TG1] 238:                 pause = Ø;
[PØ1] 239:                 break;
[PB1] 24Ø:               case 'S':
[1H1] 241:                 pause = 4;
[AG1] 242:                 break;)
[AC1] 243:               case 'X':
[Y81] 244:               case '~':
[P81] 245:               case ';':
[4W1] 246:                 break; )))
[7D3] 247: #include<process.h>
[ØN4] 248: main(int argc, char *argv[])
[752] 249:   (inst_mus();
[5F1] 25Ø:   if(argc>1)
[2K6] 251:   spawnlp(P_WAIT,*(argv+1),*(argv+1),NULL);
[772] 252:   end_mus();)
```

Listing 1. Mit »mus.c« lassen Sie im Hintergrund Geräusche ertönen.

```
[W6Ø]   1: /*
[T53]   2:  Programm: usemus.c
[1S6]   3:  Funktion: Hintergrund-Soundverwaltung
[K83]   4:  Sprache:  Turbo C 2.Ø
[KR4]   5:  Autor:    Matthias Oberdorfer
[2Z4]   6:  (c) 1992 DMV Widuch DmbH & Co. KG */
[ØØØ]   7:
[2A6]   8: #define MUSPORT *((unsigned far *)Øx18Ø)
[UWØ]   9: main()
[HP6]  1Ø: (printf("Bitte Taste drücken, um Brief "
[V53]  11:  "abzuschicken.\n");
[R61]  12:  getch();
[FV1]  13:  MUSPORT=4;
[A82]  14:  while(!kbhit())
[FC6]  15:  printf("Brief abgeschickt. Taste für "
[AM1]  16:  "Ende\n");
[R61]  17:  getch();
[U7Ø]  18: }
```

Listing 2. »Usemus.c« demonstriert die ferngesteuerte Musikverwaltung.

Monopoly

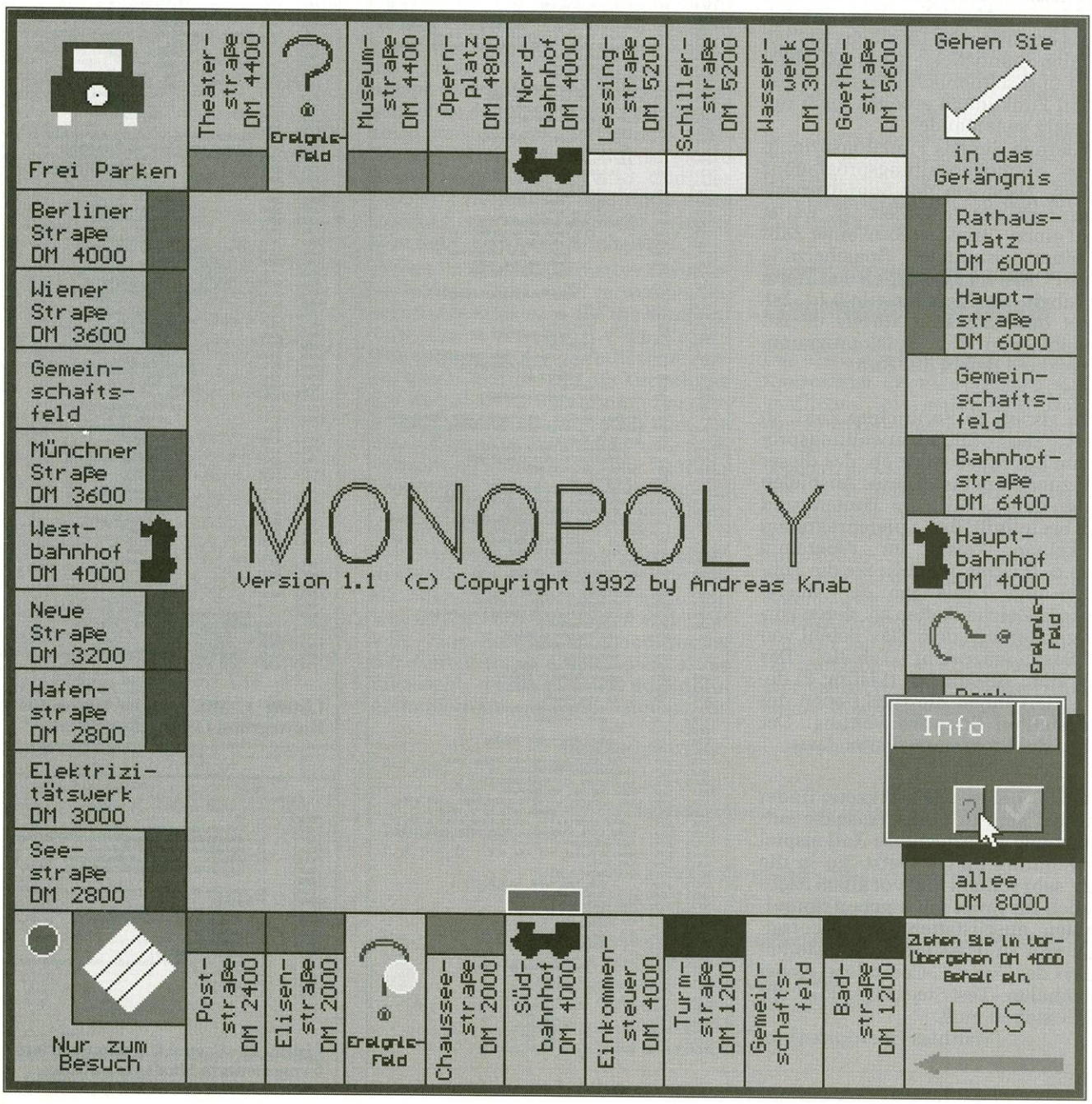
Spekulieren Sie richtig

»Gehen Sie sofort ins Gefängnis. Gehen Sie nicht über »Los«. Ziehen Sie keine 4000 Mark ein.« Wenn Sie jetzt eine Gänsehaut bekommen, ist das altbewährte Gesellschaftsspiel Monopoly genau das richtige für Sie.

Mit »monopoly.exe« können bis zu sechs Personen am Computer gegeneinander Monopoly spielen. Das Programm können Sie ohne Probleme mit Turbo Pascal 6.0 übersetzen oder ergänzen. Die benötigten Bildschirmtreiber und -schriften werden automatisch eingebunden.

Nach dem Start des Programms erscheint das Spielfeld in VGA-Qualität auf dem Bildschirm. Rechts am Rand ist eine Spalte frei, die die zum Spiel nötigen Daten und Menüfelder aufnimmt. Bevor Sie die Teilnehmer Ihrer Spielrunde eingegeben haben, ist außer einem Eingabefeld noch nichts zu sehen.

Jeder Spieler (zwei bis sechs können teilnehmen) gibt zunächst über die Tastatur einen Namen ein, unter dem er später ansprechbar ist. Drücken Sie aber noch nicht [Enter] zur Bestätigung! Öffnen Sie vorab mit der Tabulator-Taste ein Dialogfenster (Bild 1), das Ihnen 16 Farben anzeigt, die Ihre Spielfigur annehmen kann. Sind Sie nicht der erste Spieler, so fehlen natürlich schon die Farben, die Ihre Vorgänger gewählt haben. Bestätigen Sie erst nach der Farbwahl mit [Enter], so daß sich der nächste Spieler eintragen



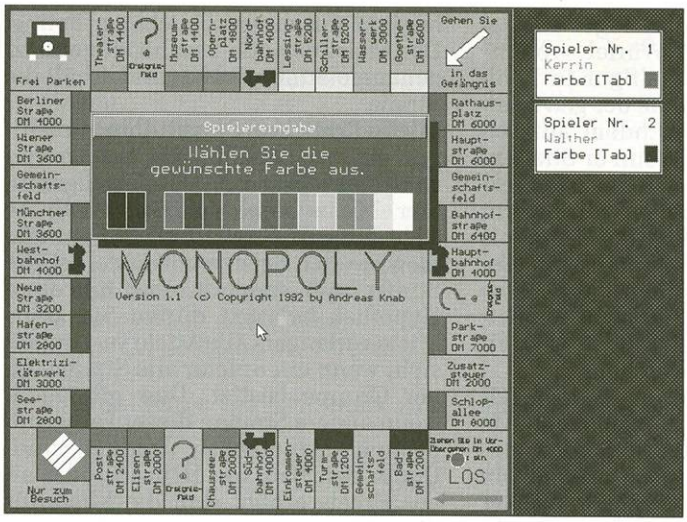


Bild 1. Unter diesen Farben kann der Spieler auswählen.

kann. Haben sich sechs Spieler auf diese Weise »angemeldet«, erscheint sofort das Hauptmenü. Andernfalls drücken Sie nach Eingabe der letzten Person [Enter] ohne Namenseingabe.

Während das Programm (Listings 1 bis 10) abläuft, werden Sie immer wieder Dialogfenstern begegnen, die Ihnen wichtige Informationen zeigen oder Eingaben von Ihnen erwarten. Ein Dialogfenster ist folgendermaßen aufgebaut:

In der Mitte des Fensters finden Sie eine Information oder eine Eingabeaufforderung, die Ihnen mitteilt, was zu tun ist. Die Dialogknöpfe rechts unten stellen Ihre Auswahlvarianten dar. Es gibt vier Arten von Knöpfen:

- Fragezeichen: Drücken Sie diesen Knopf, erscheint eine Information über die aktuellen Besitztümer aller Spieler.
- Rotes Kreuz: Es steht immer für »Abbruch« oder für »Nein«.
- Grüner Haken: Dieser Knopf steht immer für »Ok«.
- Beliebiger Text: Dieser Knopf steht für Abfragen, die nicht mit Ja oder Nein zu beantworten sind.

Das Fragezeichen rechts oben (es muß nicht immer vorhanden sein) öffnet ein Fenster mit einer kleinen Hilfestellung zu Ihrer aktuellen Situation.

Um einen Knopf zu aktivieren, halten Sie die linke Maustaste gedrückt, wenn der Cursor auf dem Knopf steht, und lassen dann los, oder Sie drücken einfach auf [Enter].

Der Fenstertitel erlaubt es Ihnen, das Fenster an einer anderen Stelle des Bildschirms zu plazieren, wenn es etwa ungünstig über anderen Informationen steht. Dazu bewegen Sie den Cursor auf den Titel, halten die linke Maustaste gedrückt und verschieben den erscheinenden Rahmen an die gewünschte Bildschirmposition. Dann lassen Sie die Taste los. Falls es sich um ein Eingabefenster handelt, erscheint im Informationsbereich zusätzlich ein Rahmen, in dem Sie Ihre Eingabe vornehmen können. Alle Tasten, die Sie drücken, werden dann in das Fenster eingetragen. Die Rückschritt-Taste löscht das letzte Zeichen, wenn Sie sich einmal vertippt haben.

Nach der Eingabe aller Spieler finden Sie sich im Hauptmenü wieder. Den überwiegenden Teil des Bildschirms nimmt das Spielbrett ein: Rechts oben erscheint der Name des Spielers, der gerade an der Reihe ist, mit dem Betrag seines Barvermögens, darunter die Straßen, die er bereits erworben hat (dieser Platz ist zu Beginn leer).

• Die neun Menüsymbole

Ganz rechts unten befinden sich die neun Menüsymbole, von denen einige jetzt durchgestrichen, das heißt nicht aktivierbar sind. Das mittlere Feld enthält zusätzlich einen gelben Rahmen, der den jeweils gewählten Modus anzeigt.

Würfelsymbol: Wenn Sie dieses anklicken, setzen sich zwei Würfel in Bewegung, um die Weite ihres nächsten Zugs festzulegen. Der Wurf wird angezeigt, anschließend bewegt sich Ihre Spielfigur um die entsprechende Anzahl Felder nach vorne. Das Würfelsymbol wird dann durchgestrichen, da Sie in jeder Runde nur einmal würfeln dürfen. Eine Ausnahme bildet der Pasch (beide Würfel zeigen die gleiche Augenzahl an). Falls Sie auf ein Feld kommen, das bestimmte Aktionen verlangt, werden diese sofort ausgeführt.

Häusersymbol: Dieses Symbol ist erst aktivierbar, wenn Sie Straßen besitzen. Es ist ein Modussymbol, das heißt, es passiert nichts, bis Sie ein Straßenfeld anwählen. Dann erscheint ein Dialogfenster, das Ihnen mitteilt, ob Sie auf der betreffenden Straße bauen können, und wenn ja, ob Sie ein Haus bauen oder abreißen wollen.

Hypothekensymbol: Auch dieses Symbol ist ein Modussymbol, das erst bei Straßenbesitz aktivierbar ist. Wählen Sie im aktivierten Modus ein Straßenfeld an, teilt Ihnen ein Dialogfenster mit, ob Sie auf diese Straße eine Hypothek aufnehmen beziehungsweise abzahlen können, und fragt, ob Sie dies auch tun wollen.

Handelssymbol: Dieses Modussymbol erlaubt Ihnen, Straßen aus Ihrem Besitz zu verkaufen und fremde Straßen zu erwerben. Wählen Sie einfach die Straße an, die Sie kaufen oder verkaufen wollen.

Informationssymbol: Dieses Modussymbol ist standardmäßig aktiviert (das heißt durch den gelben Rahmen markiert). Wählen Sie hier ein Straßenfeld an, erscheint die Besitzrechtskarte zu derjenigen Karte, mit der Sie die Mieten sowie die Häuserpreise abfragen können. Ist die Straße mit einer Hypothek belastet, so wird eine Karte angezeigt, die den Hypothekenbetrag angibt.

Außer auf Straßenfelder ist diese Option auch auf das Feld «Frei parken» anwendbar. Es wird dann der Geldbetrag angezeigt, der sich auf diesem Feld befindet.

Weitersymbol: Dieses Symbol und das Würfelsymbol sind abwechselnd aktivierbar. Haben Sie bereits gewürfelt, können Sie über dieses Symbol Ihren

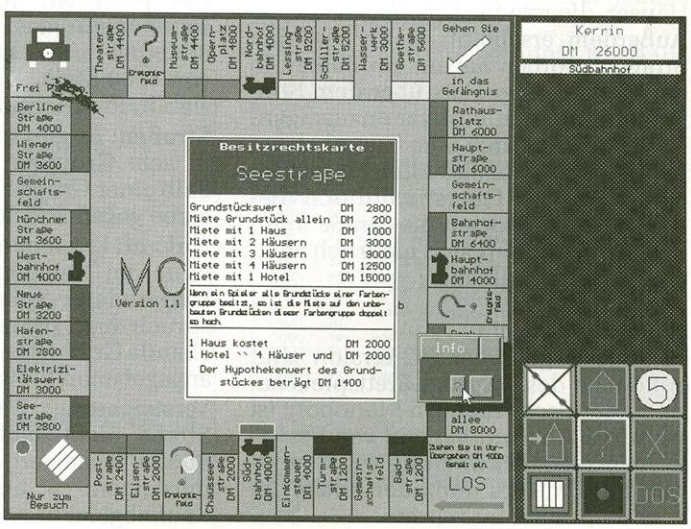


Bild 2. Das Informationsblatt enthält alle wichtigen Angaben für den Kauf.

Zug beenden, so daß der nächste Spieler an die Reihe kommt.
Gefängnissymbol: Wählen Sie dieses Symbol an, öffnet sich ein Dialogfenster, das Ihnen verschiedene Optionen bietet, Gefängnisfreikarten zu erwerben oder zu verkaufen, oder – falls Sie gerade im Gefängnis sind – eine Strafe von 1000 Mark zu bezahlen und so wieder auf freien Fuß zu kommen.
Diskettensymbol: Hiermit öffnen Sie ein Dialogfenster, mit dem Sie das aktuelle Spiel auf Diskette speichern oder ein bereits gespeichertes Spiel von Diskette laden. Letzteres erreichen Sie auch, wenn Sie bei Spielbeginn gleich im ersten Eingabefeld [Enter] ohne Namenseingabe drücken.
DOS-Symbol: Mit diesem Symbol verlassen Sie das Monopoly-Programm und kehren auf die DOS-Ebene zurück.

- **Die Tastaturbelegung und die Mausfunktionen**

Machen Sie sich jetzt die Funktionenvielfalt der beiden Maustasten klar. Die besondere Belegung verschiedener Tasten finden Sie in der Tabelle.
Linker Mausknopf: Dessen Spektrum hängt von der Position des Mauscursors ab, die angewählte Funktion wird ausgeführt.
Rechter Mausknopf: Er hat situationsbedingt verschiedene Aufgaben.

- Im Hauptmenü wird das Würfelnsymbol aktiviert. Hat der Spieler bereits gewürfelt, erscheint das Weitersymbol.
- In einem Dialogfenster wird, sofern vorhanden, immer der OK- sprich Ja-Knopf aktiviert (dargestellt durch einen grünen Haken).

Bei dieser Belegung können Sie ganze Spielzüge nur mit den Mausknöpfen durchführen.

- **Die Darstellung des Spielbretts**

Zur Verdeutlichung der Spielsituation erscheinen im Laufe des Spiels weitere Symbole auf dem Spielfeld. Dazu gehören Gebäude auf den Straßenfeldern: Die grüne Farbe steht dabei für Häuser, die rote für Hotels.
Außerdem erscheint, wenn Sie eine Straße erworben haben, ein Balken in der Farbe des Spielers über dem Straßenfeld, um den Besitzer anzuzeigen. Ist die Straße durch eine Hypothek belastet, so verläuft zusätzlich ein weißer oder schwarzer Querstrich durch diesen Balken. Wechselt die Straße ihren Besitzer, so ändert sich natürlich auch die Farbe des Balkens.

- **Die Spielregeln**

Die Spielregeln der PC-Version sind mit denen des Original-Brettspiels nahezu identisch. Ziel von Monopoly ist es, durch An- und Verkauf von Besitzrechten sowie Vermietung von Grundstücken, Häusern und/oder Hotels der reichste Spieler zu werden.

Jeder Spieler setzt seine Figur auf das Los-Feld, läßt zwei Würfel »rollen« und bewegt seine Spielfigur in Pfeilrichtung um die Summe der gewürfelten Augen weiter. Je nachdem, auf welchem Feld er landet, kann er Straßen, Bahnhöfe, das Wasser- oder das Elektrizitätswerk kaufen oder ersteigern, muß Steuern bezahlen, ins Gefängnis gehen oder eine Ereignis- oder Gemeinschaftskarte ziehen. Anschließend ist der nächste Spieler an der Reihe. Es können auch zwei oder mehr Figuren auf einem Feld stehen.
Bei einem Pasch führt der Spieler seinen Zug mit allen sich ergebenden Aktionen durch und würfelt noch einmal. Bei erneutem Pasch muß er ein drittes Mal würfeln. Wirft der Spieler dreimal hintereinander einen Pasch, darf er den dritten Zug nicht mehr ausführen, sondern muß sich mit seiner Spielfigur sofort ins Gefängnis begeben.

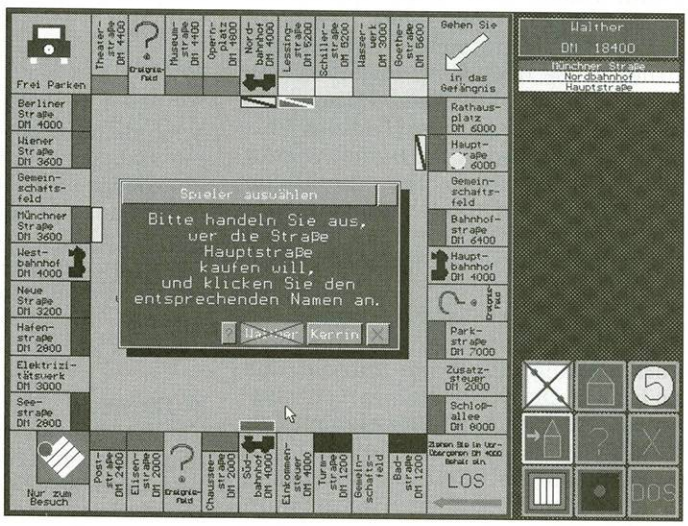


Bild 3. Handeln Sie aus, wer die Straße kauft.

Die Bedeutung der Felder des Spielplans

Los: Jedesmal, wenn ein Spieler mit seiner Figur das Feld »Los« erreicht oder passiert, bezieht er ein Gehalt von 4000 Mark. Dies gilt nur, wenn er sich in Pfeilrichtung bewegt.
Straßen: Es gibt insgesamt 22 Straßen in acht Gruppen. Jede Straße innerhalb einer Gruppe ist mit der gleichen Farbe gekennzeichnet.
Ankauf: Erreichen Sie mit Ihrer Spielfigur ein Straßenfeld, auf das noch kein anderer Spieler einen Besitzanspruch hat, können Sie die entsprechende Grundstückskarte (Bild 2) zum angegebenen Preis kaufen.
Versteigerung: Wenn der Spieler die Straße nicht kaufen will, wird die Straße versteigert (Bild 3). Jeder kann mitsteigern – auch der, der einen Kauf abgelehnt hat. Die Versteigerung beginnt mit einem beliebigen Betrag. Der Meistbietende bezahlt die zuletzt genannte Summe sofort und erhält die Straße.
Miete: Erreichen Sie mit Ihrer Spielfigur ein Straßenfeld, das bereits verkauft ist, müssen Sie an den Eigentümer Miete bezahlen. Die Miethöhe steht für jede Straße fest: Sie entnehmen sie der Besitzrechtskarte. Ist die Straße zwar verkauft, aber mit einer Hypothek belastet, dürfen Sie keine Miete verlangen. Die Miete verdoppelt sich, wenn ein Spieler alle Straßen einer Gruppe besitzt. Dies gilt auch, wenn andere Straßen dieser Farbgruppe mit Hypotheken belastet sind. Durch den Bau von Häusern oder eines Hotels erhöht sich die Miete für diese Straße.
Bahnhöfe, Wasser- und Elektrizitätswerk: Für diese Felder gilt das gleiche wie für die Straßenfelder. Dort können jedoch weder Häuser noch Hotels gebaut werden.
Ereignis- und Gemeinschaftsfeld: Kommen Sie mit Ihrer Spielfigur auf eines dieser Felder, müssen Sie die Anweisungen der obersten Karte befolgen. Der Spieler, der die Karte »Du kommst aus dem Gefängnis frei« zieht, behält diese zurück, um sie bei Bedarf einzusetzen. Er kann sie aber auch irgendwann an einen Spieler zu einem frei auszuhandelnden Preis verkaufen.
Zusatz- und Einkommensteuerfeld: Beide Steuern müssen Sie sofort bezahlen. Bei der Einkommensteuer haben Sie die Wahl, entweder 4000 Mark oder 10 Prozent Ihres gesamten Vermögens zu zahlen. Die Entscheidung müssen Sie fällen, bevor die gesamte Höhe des Vermögens festgestellt wird. Das Vermögen besteht aus dem Bargeld, den aufgedruckten Werten der Besitzrechtskarten sowie den Kaufwerten von Häusern und Hotels.
Gefängnisfeld: Im Gefängnis landen Sie, wenn Sie

- auf das Feld »Geh in das Gefängnis« kommen,
- eine Karte »Geh in das Gefängnis« ziehen,
- dreimal hintereinander einen Pasch werfen.

Wenn ein Spieler ins Gefängnis geht, ist sein Zug damit beendet. Kommen Sie im normalen Spielverlauf auf das Gefängnisfeld, sind Sie dort nur zu Besuch und haben keinerlei Nachteile.

Sie kommen aus dem Gefängnis frei, wenn Sie im nächsten oder übernächsten Zug

- eine Geldbuße von 1000 Mark zahlen, bevor Sie würfeln;
- einen Pasch würfeln. Sie ziehen sofort um die Summe der Würfelaugen weiter, aber ohne noch einmal zu würfeln;
- die in Ihrem Besitz befindliche Karte »Du kommst aus dem Gefängnis frei« benutzen;
- die Karte »Du kommst aus dem Gefängnis frei« von einem anderen Spieler kaufen und einsetzen.

Hat der Spieler mit seinem dritten Wurf noch keinen Pasch erzielt, muß er 1000 Mark Geldbuße bezahlen und zieht mit der Summe der Augen seines dritten Wurfs weiter. Ein Spieler, der im Gefängnis sitzt, kann Miete kassieren und Besitztümer erwerben und verkaufen.

Frei Parken: Alle Steuern, Renovierungs- und Bußgelder werden auf das Feld »Frei Parken« gelegt. Der Spieler, der dieses als nächster erreicht oder sich gerade dort befindet, wenn eine Einzahlung erfolgt, darf die Summe kassieren.

Häuser und Hotels: Ein Spieler darf eine Straße nur dann bebauen, wenn er die dazugehörige Straßengruppe insgesamt besitzt. Das zuerst gekaufte Haus kann er frei plazieren. Kauft er ein zweites, muß er es auf einer noch unbebauten Straße dieser Gruppe bauen oder an einer Straße einer anderen Gruppe, die ihm gehört. Achten Sie auf gleichmäßige Bebauung werden, das heißt, auf einer Straße dürfen Sie erst dann ein weiteres Haus bauen, wenn an den anderen Straßen dieser Farbgruppe jeweils die gleiche Anzahl

Taste	Funktion
[Enter]	entspricht Drücken der linken Maustaste
[Esc]	entspricht Drücken der rechten Maustaste
[Cursor]	bewegt den Cursor in die entsprechende Richtung
[Pos1]	setzt den Cursor in die linke obere Ecke des Spielfeldes (Frei-Parken-Feld)
[Bild auf]	setzt den Cursor in die rechte obere Ecke des Spielfeldes (»Gehen Sie in das Gefängnis-Feld«)
[Ende]	setzt den Cursor in die linke untere Ecke des Spielfeldes (Gefängnis-Feld)
[Bild ab]	setzt den Cursor in die rechte untere Ecke des Spielfeldes (Los-Feld).
[F1] bis [Fn]	setzt in Dialogfenstern den Cursor auf den ersten bis n-ten Knopf
[1] bis [9]	setzt den Cursor auf ein Menüsymbol, wobei die Zahlen wie auf der numerischen Tastatur angeordnet sind

Diese Tasten sind mit neuen Funktionen belegt.

Häuser steht. Ein Hotel entspricht dabei fünf Häusern und fällt somit auch unter diese Regel. Im übrigen ist Ihnen die Bebauung freigestellt. Wird ein Haus abgebaut, so erhalten Sie nur den halben Einkaufspreis zurück.

Immobilienverkauf: Unbebaute Grundstücke, das Elektrizitäts- und das Wasserwerk sowie die Bahnhöfe können jederzeit unter den Spielern frei ausgehandelt werden. Es kann jedoch kein Grundstück an einen anderen Spieler verkauft werden, wenn auf einer Straße dieser Farbgruppe bereits Gebäude stehen. Diese Häuser müssen zuerst abgebaut werden.

Hypotheken gibt nur die Bank aus. Der Hypothekenwert beträgt 50 Prozent des Straßenwertes, der Zinssatz 10 Prozent des Hypothekenwertes. Die Zinsen fallen an, wenn die Hypothek zurückgezahlt wird. Bei Übertragung eines hypothekarisch belasteten Besitztum kann der neue Inhaber die Hypothek sofort zurückerstatten. Er muß in diesem Falle außer der Hypothek auch die nun fälligen 10 Prozent Zinsen bezahlen. Kündigt er die Hypothek nicht, sind dennoch 10 Prozent Zinsen zu zahlen. Hebt er die Hypothek für später auf, fallen nochmals Zinsen an. Um ein Grundstück hypothekarisch zu belasten, müssen vorher alle Gebäude auf diesem Grundstück abgebaut werden.

Ein Spieler, der die Zahlungen einstellt, also mehr schuldet, als er bezahlen kann, muß dem Gläubiger seine gesamte Habe übergeben und darf nicht mehr weiterspielen. Der übrigbleibende Restbetrag wird von der Bank übernommen. Straßen, die sich noch in seinem Besitz befinden, werden meistbietend versteigert.

Wie Sie den Listings entnehmen, bilden Sie die komplexe Spielsituation mit allen raffinierten Regeln am PC nach. Die Programmiersprache Turbo Pascal mit dem System der Units erleichtert hierbei die Verständlichkeit der Struktur. Unumgänglich dabei bleibt allerdings, daß Sie sämtliche Spielregeln in den Hilfsdateien übernehmen müssen, wobei allein schon »monhilfe.txt« (Listing 10) fast 9 KByte beansprucht. Die Initialisierungsdatei »monoinit.pas« braucht gleich über 31 KByte. Doch eben die Arbeit garantiert das Spielvergnügen.

• Probleme des Programms

In Ausnahmefällen soll es bereits zu Schwierigkeiten mit dem Programm gekommen sein. Angeblich tauchten sie häufig wegen eines nicht vorhandenen Mauscursors auf.

Dazu folgendes: Monopoly arbeitet mit dem höchstauflösenden Standard-VGA-Modus, der erst vor wenigen Jahren entwickelt wurde. Viele Maustreiber sind jedoch älter und kennen diesen Modus nicht. Der Einsatz eines neueren Maustreibers behebt jedoch das Problem meistens.

(Andeas Knab/uk)

```
[DW3]    1: (Unit:      monokart.pas
[7G5]    2:  Funktion: Festlegung der Karten
[G64]    3:  Sprache:  Turbo Pascal 6.Ø
[ZC3]    4:  Autor:    Andreas Knab
[915]    5:  (c) 1993 DMV Widuch GmbH & Co. KG)
[ØØØ]    6:
[VA5]    7: {$A+,B-,D-,E+,F-,I-,L-,N-,O-,R-,S-,V-}
[BF2]    8: unit MonoKart;
[SK1]    9: Interface
[8E5]   1Ø: uses MonoInit, MonoCnst, Graph;
[ST Ø]  11: type
[US2]   12:  KarteObj=Object
[VØ3]   13:    Ereignis: Boolean;
[ZP3]   14:    EreignisText: String;
[BY6]   15:    Constructor Init(e: Boolean; t: String);
[YØ6]   16:    Procedure ZeigeKarte(X, Y: Integer);
[ZNØ]   17:  end;
[3L3]   18:  KartePtr=^KarteObj;
[NM3]   19: var EreigKartenStack,
[3D6]   2Ø:     GemKartenStack: Array [1..16] of Byte;
[P95]   21:     EreigKartenZ, GemKartenZ: Byte;
[9S1]   22:     GemKarten,
[926]   23:     ErgKarten: Array [1..16] of KartePtr;
[ØØØ]   24:
[ØR3]   25: Procedure InitKarten;
[4L2]   26: Implementation
[ØØØ]   27:
```

```
[ZR4]   28: Constructor KarteObj.Init;
[MUØ]   29: begin
[CZ4]   3Ø:   Ereignis:=e; EreignisText:=t;
[ZNØ]   31: end;
[ØØØ]   32:
[N65]   33: Procedure KarteObj.ZeigeKarte;
[GØ5]   34: var s: String; w: Word; b: Byte;
[MUØ]   35: begin
[YG5]   36:   SetFillStyle (SolidFill, White);
[RK3]   37:   Bar (X, Y, X + 3ØØ, Y + 15Ø);
[CP2]   38:   if Ereignis then
[Z65]   39:     SetFillStyle (SolidFill, Cyan)
[6E7]   4Ø:  else SetFillStyle(SolidFill,LightMagenta);
[ØY3]   41:   Bar(X+1Ø, Y+1Ø, X+29Ø, Y+14Ø);
[6U5]   42:   SetzeSchrift(SmallFont, HorizDir,
[TE2]   43:                7, CenterText);
[UT2]   44:   SetColor (Black);
[CP2]   45:   if Ereignis then
[A96]   46:    OutTextXY (X+15Ø, Y+3Ø, 'Ereigniskarte')
[ZRØ]   47:   else
[ZK3]   48:    OutTextXY (X+15Ø, Y+3Ø,
[8S3]   49:                         'Gemeinschaftskarte');
[5S6]   5Ø:   SetTextStyle (DefaultFont, HorizDir, 1);
[734]   51:   w:=Y+6Ø; s:=EreignisText;
[VE2]   52:   while (s <> '') Do
[MUØ]   53:   begin
[R72]   54:     b:=Pos ('\', s);
```

```
[EB5]   55:     if (b = Ø) then b:=Succ (Length (s));
[ØT5]   56:     OutTextXY(X+15Ø,w,Copy(s,1,Pred(b)));
[3W3]   57:     Delete (s, 1, b); Inc (w, 1Ø);
[ZNØ]   58:   end;
[ZNØ]   59: end;
[ØØØ]   6Ø:
[ØR3]   61: Procedure InitKarten;
[EF4]   62: var b: Byte;  u: Set of 1..16;
[MUØ]   63: begin
[PH2]   64:   for b:=1 to 16 Do
[MUØ]   65:   begin
[B63]   66:     New(ErgKarten [b],
[Z45]   67:                Init (True, EreignisTxt [b]));
[W73]   68:     New (GemKarten [b],
[C85]   69:                Init (False, Gemeinschtxt [b]));
[ZNØ]   7Ø:   end;
[EZ3]   71:   EreigKartenZ:=Ø; u:=[];
[RB4]   72:   while (EreigKartenZ < 16) Do
[MUØ]   73:   begin
[WA3]   74:     b:=Succ (Random (15));
[WD2]   75:     while (b In u) Do
```

Listing 1. »Monokart.pas« legt die Ereignis- und die Gemeinschaftskarten fest.

```
[MUØ]    76:      begin
[X41]    77:        Inc (b);
[GZ2]    78:        if (b > 16) then b:=1;
[ZNØ]    79:      end;
[8U4]    8Ø:      Inc (EreigKartenZ); u:=u+[b];
[W86]    81:      EreigKartenStack [b]:=EreigKartenZ;
[ZNØ]    82:    end;
[8G3]    83:    GemKartenZ:=Ø; u:=[];
[8Ø4]    84:    while (GemKartenZ < 16) Do
[MUØ]    85:    begin
[WA3]    86:      b:=Succ (Random (15));
[WD2]    87:      while (b In u) Do
[MUØ]    88:      begin
[X41]    89:        Inc (b);
[GZ2]    9Ø:        if (b > 16) then b:=1;
[ZNØ]    91:      end;
[LM4]    92:      Inc (GemKartenZ); u:=u+[b];
[GM5]    93:      GemKartenStack [b]:=GemKartenZ;
[ZNØ]    94:    end;
[W45]    95:    GefangFreiE:=Ø; GefangFreiG:=Ø;
[ZNØ]    96: end;
[KNØ]    97: end.
```

Listing 1. »Monokart.pas« legt die Ereignis- und die Gemeinschaftskarten fest.

```
[AW3]     1: (Unit:      monoinit.pas
[LØ5]     2:  Funktion: Initialisierungen
[G64]     3:  Sprache:  Turbo Pascal 6.Ø
[ZC3]     4:  Autor:    Andreas Knab
[915]     5:  (c) 1993 DMV Widuch GmbH & Co. KG)
[ØØØ]     6:
[VA5]     7: ($A+,B-,D-,E+,F-,I-,L-,N-,O-,R-,S-,V-)
[GG2]     8: unit MonoInit;
[SK1]     9: interface
[TL5]    1Ø: uses Mouse, Crt, Graph, MonoCnst;
[ØØØ]    11:
[4C4]    12: type NameStr=String [2Ø];
[UY4]    13:      MietTyp=Array [1..6] of Word;
[LV3]    14:      SetofByte=Set of Byte;
[V14]    15: type StrasseObj=Object
[E14]    16:        Gruppe, Position: Byte;
[RG4]    17:        Name: NameStr;  Wert: Word;
[DR5]    18:        Miete: MietTyp; Hypothek: Boolean;
[X25]    19:        Bebauung: Byte; Spieler: Byte;
[335]    2Ø:        Constructor Init(n: NameStr;
[GØ3]    21:                         g, p: Byte; w: Word);
[Z17]    22: procedure SetzeMiete(l,h1,h2,h3,h4,h:Word);
[YØ6]    23:       procedure ZeigeKarte(X, Y: Integer);
[T74]    24:       procedure ZeigeBebauung;
[8Ø6]    25:       procedure ZeigeTitel(X, Y: Integer);
[XB5]    26:       procedure SetzeSpieler(n: Byte);
[ØB4]    27:       function BebauungErlaubt
[E33]    28:                          (h: Byte): Boolean;
[9D4]    29:       function HoleMiete: Word;
[ZNØ]    3Ø:     end;
[TC4]    31:     StrassePtr=^StrasseObj;
[3Ø3]    32:     SpielerObj=Object
[PB3]    33:       Farbe, Nummer  : Byte;
[YZ2]    34:       Name: String [2Ø];
[SN3]    35:       Strassen: Set of 1..4Ø;
[HZ4]    36:       Geld: LongInt; Feld: ShortInt;
[S65]    37:       Puffer: Pointer; Fx, Fy: Word;
[UA5]    38:       Gefangen: Byte; Pleite: Boolean;
[293]    39:       Bezahlend: Boolean;
[ØX6]    4Ø:     Constructor Init(n,f:Byte;na:String);
[CN4]    41:       procedure StrDazu(n: Byte);
[8G4]    42:       procedure StrWeg(n: Byte);
[CR3]    43:       procedure ZeigeFigur;
[N26]    44:       procedure BewegeFigur(n: ShortInt);
[AL4]    45:       procedure Bezahlen(p: Word;
[7U1]    46:                          f: Boolean);
[C53]    47:       Destructor Fertig;
[ZNØ]    48:     end;
[T94]    49:     SpielerPtr=^SpielerObj;
[PTØ]    5Ø: var Strasse: Array [1..4Ø] of StrassePtr;
[XL5]    51:     LetzterWurf, Pasch, SpielerDran,
[BC6]    52:     SpielerZahl: Byte; Spielende: Boolean;
[SØ6]    53:     Spieler: Array [1..6] of SpielerPtr;
[WK3]    54:     FreiParken: LongInt;
[975]    55:     GefangFreiE, GefangFreiG: Byte;
[ØØØ]    56:
[184]    57: procedure InitStrassen;
[WC3]    58: procedure InitBild;
[MD7]    59: function StrNum(w:LongInt;n:Byte): String;
[H66]    6Ø: procedure SetzeSchrift(Art, Dir, Gr,
[AX1]    61:                         Ausr: Byte);
[WL4]    62: function FeldNummer: Byte;
[N65]    63: procedure BauAktion(Sp: Byte);
[8V4]    64: procedure Hypothek(Sp: Byte);
[VR6]    65: function WaehleSpieler(s: string): Byte;
[1M4]    66: procedure Handel(Spi: Byte);
[C85]    67: procedure ZeigeSpielerStatus;
[YL4]    68: procedure ZeigeSpielstand;
[K26]    69: procedure AufFreiParken(l: LongInt);
[ØØ]     7Ø:
[4L2]    71: Implementation
[LE2]    72: uses MonoDial;
[ØØØ]    73:
[MD7]    74: function StrNum(w:LongInt;n:Byte): String;
[DA2]    75: var s: String;
[MUØ]    76: begin
[TØ4]    77:   Str (w:n, s); StrNum:=s;
[ZNØ]    78: end;
[ØØØ]    79:
[H66]    8Ø: procedure SetzeSchrift(Art, Dir, Gr,
[AX1]    81:                         Ausr: Byte);
[MUØ]    82: begin
[FH4]    83:   SetTextStyle (Art, Dir, Gr);
[MØ6]    84:   SetTextJustify (Ausr, CenterText);
[ZNØ]    85: end;
[ØØØ]    86:
[C85]    87: procedure ZeigeSpielerStatus;
[GV3]    88: var y: Integer; b: Byte;
[MUØ]    89: begin
[8V6]    9Ø:   SetFillStyle (ClosedotFill, LightGray);
[XX2]    91:   Bar (48Ø, Ø, 639, 325);
[LS6]    92:   if (Spieler [SpielerDran]^.Farbe<8) then
[3F5]    93:     SetFillStyle (SolidFill, White)
[7R6]    94:   else SetFillStyle (SolidFill, DarkGray);
[5S2]    95:   Bar (485, 5, 634, 4Ø);
[PX6]    96:   SetColor (Spieler [SpielerDran]^.Farbe);
[HM3]    97:   Rectangle (485, 5, 634, 4Ø);
[676]    98:   SetzeSchrift (SmallFont, HorizDir, 5,
[142]    99:                       CenterText);
[GY2]   1ØØ:   OutTextXY (56Ø, 1Ø,
[Ø55]   1Ø1:                Spieler [SpielerDran]^.Name);
[TWA]   1Ø2:   OutTextXY (56Ø, 3Ø, 'DM ' + StrNum (Spiel
er [SpielerDran]^.Geld, 6));
[3XØ]   1Ø3:   y:=45;
[AH2]   1Ø4:   for b:=1 To 4Ø do
[VX3]   1Ø5:     if (StrWert [b] <> Ø) And
[GC7]   1Ø6: (b In Spieler [SpielerDran]^.Strassen) then
[MUØ]   1Ø7:     begin
[CM5]   1Ø8:       Strasse [b]^.ZeigeTitel (485, y);
[4N1]   1Ø9:       Inc (y, 1Ø);
[ZNØ]   11Ø:     end;
```

```
[AK5]   111:   if (GefangFreiE=SpielerDran) then
[MUØ]   112:   begin
[E75]   113:     SetFillStyle(SolidFill, Cyan);
[GR2]   114:     Bar (622, 7, 632, 22);
[ZNØ]   115:   end;
[KK5]   116:   if (GefangFreiG=SpielerDran) then
[MUØ]   117:   begin
[1P6]   118:     SetFillStyle(SolidFill, LightMagenta);
[3W2]   119:     Bar (622, 23, 632, 38);
[ZNØ]   12Ø:   end;
[ZNØ]   121: end;
[ØØØ]   122:
[YL4]   123: procedure ZeigeSpielstand;
[7H4]   124: var d: DialogObj; b, c: Byte;
[W76]   125:     y: Integer; str, hyp, haeu, hot: byte;
[TL4]   126:     ges : LongInt; s: string[1Ø];
[MUØ]   127: begin
[B26]   128:  d.Zeige(8Ø,8Ø,519,36Ø-(6-SpielerZahl)*25);
[DP5]   129:  d.SetOpt (#Ø#32'Spielstand', '', 'Ok');
[XG3]   13Ø:  SetColor(LightGray);
[736]   131:  SetzeSchrift(SmallFont, VertDir, 5,
[142]   132:                       CenterText);
[L95]   133:  OutTextXY(285, 15Ø, 'Straßen');
[1W4]   134:  OutTextXY(3Ø5, 15Ø, 'Hypoth.');
[7X4]   135:  OutTextXY(335, 15Ø, 'Häuser');
[9N4]   136:  OutTextXY(355, 15Ø, 'Hotels');
[656]   137:  SetzeSchrift(SmallFont, HorizDir, 5,
[1S1]   138:                       LeftText);
[XU4]   139:  OutTextXY(1Ø5, 16Ø, 'Spieler');
[734]   14Ø:  OutTextXY(2Ø5, 16Ø, 'Geld');
[ØM4]   141:  OutTextXY(375, 16Ø, 'Gesamt');
[EM3]   142:  SetColor (LightCyan);
[RF4]   143:  for b:=1 to SpielerZahl do
[MUØ]   144:  begin
[7Y1]   145:    y:=162 + b * 25;
[DK3]   146:    with Spieler[b]^ do
[MUØ]   147:    begin
[FP3]   148:      OutTextXY(1ØØ, y, Name);
[F46]   149:     OutTextXY(2ØØ,y,StrNum(Geld,6)+' DM');
[7U3]   15Ø:      str:=Ø; hyp:=Ø; haeu:=Ø;
[8Z2]   151:      hot:=Ø; ges:=Geld;
[CH2]   152:      for c:=1 to 4Ø do
[DP3]   153:        if (c in Strassen) then
[MUØ]   154:        begin
[TH1]   155:          inc(str);
[1N3]   156:          with Strasse[c]^ do
[MUØ]   157:          begin
[BR2]   158:            if hypothek then
[MUØ]   159:            begin
[LP4]   16Ø:              inc(hyp); inc(ges,Wert div 2);
[4MØ]   161:            end
[T73]   162:            else inc(ges, Wert);
[FH4]   163:            if (Bebauung=5) then inc(hot)
[EZ3]   164:            else inc(haeu, Bebauung);
[5C1]   165:            inc(ges,
[3R5]   166:       Bebauung*succ(position div 1Ø)*1ØØØ)
        ;
[ZNØ]   167:          end;
[ZNØ]   168:        end;
[ØC5]   169:      OutTextXY(28Ø, y, StrNum(Str, 2));
[DB5]   17Ø:      OutTextXY(3ØØ, y, StrNum(Hyp, 2));
[KF5]   171:      OutTextXY(32Ø, y, StrNum(Haeu, 3));
[EA5]   172:      OutTextXY(35Ø, y, StrNum(Hot, 2));
[EV5]   173:      OutTextXY(37Ø,y,StrNum(Ges,6)+' DM');
[ØØ6]   174:  s:=StrNum(Trunc(Ges/3ØØØ*1ØØ-1ØØ),5)+'%';
[2K4]   175:      if (Pos('-', s)=Ø) then s[1]:='+'
[ZRØ]   176:      else
[MUØ]   177:      begin
[PB4]   178:        s[Pos('-', s)]:=' ';  s[1]:='-';
[ZNØ]   179:      end;
[M83]   18Ø:      OutTextXY(455, y, s);
[ZNØ]   181:    end;
[ZNØ]   182:  end;
[LR2]   183:  d.Eingabe;  d.Weg;
[ZNØ]   184: end;
[ØØØ]   185:
[184]   186: procedure InitStrassen;
[255]   187: var b, c, g: Byte; s: String [2Ø];
[MUØ]   188: begin
[AH2]   189:   for b:=1 To 4Ø do
[MUØ]   19Ø:     begin
[UY2]   191:       g:=StrGruppe [b];
[XL2]   192:       s:=StrName [b];
[4T3]   193:       if (Pos ('-', s) <> Ø) then
[WL3]   194:           Delete (s, Pos ('-', s), 1);
[AG5]   195:       New (Strasse[b],Init(s,g Mod 9,b,
[353]   196:            StrWert [b] * 1ØØ));
[Y42]   197:       if (g <> Ø) then
[MUØ]   198:       begin
[CQ3]   199:         c:=Pred (g Mod 9) * 12 + 1;
[CR3]   2ØØ:         if (g > 8) then Inc (c, 6);
[R27]   2Ø1:    Strasse [b]^.SetzeMiete(StrMiete[c    ],
[TP2]   2Ø2:         StrMiete [c + 1],
[2G5]   2Ø3:         StrMiete [c + 2], StrMiete [c + 3],
[XH5]   2Ø4:         StrMiete [c + 4], StrMiete [c + 5]);
[ZNØ]   2Ø5:       end;
[ZNØ]   2Ø6:     end;
[ZNØ]   2Ø7: end;
[ØØØ]   2Ø8:
[WC3]   2Ø9: procedure InitBild;
[4F6]   21Ø:  procedure SchreibeStrasse(x, y: Word;
[FL1]   211:                            n, d: Byte);
[XX3]   212:   var t: Byte; s: String;
[MUØ]   213:   begin
[886]   214:     SetTextJustify(LeftText, TopText);
[KA5]   215:     SetTextStyle (SmallFont, d, 4);
[3U4]   216:     s:=StrName [n]; t:=Pos ('-', s);
[CD3]   217:     if (s [t - 1] <> ' ') then
[MUØ]   218:       begin
[XZ3]   219:         Insert ('-', s, t); Inc (t);
[ZNØ]   22Ø:       end;
[E85]   221:     OutTextXY (x, y, Copy (s, 1, t - 1));
[DX4]   222:     if (d=HorizDir) then Inc (y, 1Ø)
[BG2]   223:     else Inc (x, 1Ø);
[RF5]   224:     OutTextXY (x, y, Copy (s, t + 1, 2Ø));
[DX4]   225:     if (d=HorizDir) then Inc (y, 1Ø)
[BG2]   226:     else Inc (x, 1Ø);
[7L2]   227:     OutTextXY(x,y,
[E75]   228:      'DM '+StrNum(StrWert[n]*1ØØ,4));
[ZNØ]   229:   end;
[ØØØ]   23Ø:
[NU6]   231:   procedure Lokomotive(x,y: Word; d: Byte);
[CW6]   232:   type LokTyp=Array [1..12] of Pointtype;
[NK5]   233:        RadTyp=Array [1..2] of Pointtype;
[2V3]   234:   const LokDaten: LokTyp
[535]   235:    =((x: Ø;y: 7), (x: 3;y: 5), (x: 5;y: 5),
[R45]   236:     (x: 3;y: Ø), (x: 1Ø;y: Ø), (x: 8;y: 5),
[735]   237:     (x: 2Ø;y: 5),(x: 2Ø;y: Ø),(x: 3Ø;y: Ø),
[3E5]   238:   (x: 3Ø; y: 12),(x: 3;y: 12),(x: Ø;y: 7));
[GT2]   239:     RadDaten: RadTyp
[SX3]   24Ø:    =((x: 8;y: 11), (x: 22;y: 11));
[7N6]   241:   var b, c: Byte; Lok: LokTyp; Rad: RadTyp;
[MUØ]   242:   begin
[LU4]   243:     Lok:=LokDaten; Rad:=RadDaten;
[B33]   244:     if (d=VertDir) then
[MUØ]   245:     begin
[AH2]   246:       for b:=1 To 12 do
[MUØ]   247:       begin
[FZ1]   248:         c:=Lok [b].x;
[9G3]   249:         Lok [b].x:=Lok [b].y;
[W72]   25Ø:         Lok [b].y:=c;
[ZNØ]   251:       end;
[LF2]   252:       for b:=1 To 2 do
[MUØ]   253:       begin
[4Y1]   254:         c:=Rad [b].x;
[4D3]   255:         Rad [b].x:=Rad [b].y;
[262]   256:         Rad [b].y:=c;
```

```
[ZNØ]   257:       end;
[ZNØ]   258:     end;
[AH2]   259:     for b:=1 To 12 do
[MUØ]   26Ø:     begin
[433]   261:       Inc (Lok [b].x, x);
[933]   262:       Inc (Lok [b].y, y);
[ZNØ]   263:     end;
[LF2]   264:     for b:=1 To 2 do
[MUØ]   265:     begin
[R13]   266:       Inc (Rad [b].x, x);
[W13]   267:       Inc (Rad [b].y, y);
[ZNØ]   268:     end;
[RE5]   269:     SetFillStyle (SolidFill, Black);
[9V5]   27Ø:     SetColor (Black); FillPoly (12, Lok);
[GF5]   271:     Sector(Rad[1].x,Rad[1].y,Ø,36Ø,4,4);
[LF5]   272:     Sector(Rad[2].x,Rad[2].y,Ø,36Ø,4,4);
[ZNØ]   273:   end;
[ØØØ]   274:
[C77]   275: const Gefaengnis: Array [1..5] of Pointtype
[KL5]   276:  =((x:35;y:424),(x:55;y:4Ø4),(x:75;y:424),
[4F4]   277:    (x: 55; y: 444), (x: 35; y: 424));
[595]   278:    Pfeil: Array [1..8] of Pointtype
[UØ4]   279:    =((x:414;y:459),(x:414;y:461),
[PU3]   28Ø:     (x:469;y:461),(x:469;y:467),
[GT3]   281:     (x:414;y:467),(x:414;y:469),
[WW3]   282:     (x:4Ø4;y:464),(x:414;y:459));
[VV5]   283:    GefPfeil: Array [1..8] of Pointtype
[TX5]   284:  =((x:416;y:54),(x:416;y:39),(x:421;y:44),
[ØE5]   285:    (x:451;y:19),(x:456;y:24),(x:426;y:49),
[RU3]   286:    (x:431;y:54),(x:416;y:54));
[LX3]   287:    GemRich: Array[1..3] of
[F16]   288:       Byte=(VertDir, HorizDir, HorizDir);
[WA5]   289:    GemXY: Array [1..9] of Pointtype
[XY5]   29Ø: =((x:328;y:418),(x:12;y:153),(x:422;y:153),
[1H5]   291:  (x:338;y:418),(x:12;y:163),(x:422;y:163),
[KP5]   292:  (x:348;y:418),(x:12;y:173),(x:422;y:173));
[Y45]   293: var w, x, y, t: Word; s: String;
[MUØ]   294: begin
[LE6]   295:   SetFillStyle (SolidFill, LightGreen);
[6L2]   296:   Bar (5, 5, 474, 474);
[YS3]   297:   SetColor (LightGreen);
[EM3]   298:   Rectangle (Ø, Ø, 479, 479);
[B55]   299:   SetFillStyle (SolidFill, Red);
[FF4]   3ØØ:   FloodFill (1, 1, LightGreen);
[8V6]   3Ø1:   SetFillStyle (ClosedotFill, LightGray);
[LP5]   3Ø2:   Bar (48Ø, Ø, 639, 479); SetColor(Black);
[XS3]   3Ø3:   Rectangle (8Ø, 8Ø, 399, 399);
[AH2]   3Ø4:   for w:=Ø To 9 do
[MUØ]   3Ø5:   begin
[4M3]   3Ø6:     x:=8Ø + Trunc (319 / 9 * w);
[SW4]   3Ø7:     Line(x,5,x,79); Line(x,4ØØ,x,474);
[GU4]   3Ø8:     Line(5,x,79,x); Line(4ØØ,x,474,x);
[ZNØ]   3Ø9:   end;
[8H2]   31Ø:   for w:=Ø To 8 do
[MUØ]   311:   begin
[C63]   312:     x:=8Ø+Trunc(319/9*w);
[L34]   313:     y:=8Ø+Trunc(319/9*Succ(w));
[B54]   314:     if(StrName [21+w] <> '') then
[MUØ]   315:     begin
[VU4]   316:       if(StrGruppe [21+w] <> Ø) then
[MUØ]   317:       begin
[1N2]   318:         Line(x, 62, y, 62);
[874]   319:         SetFillStyle(SolidFill,
[PN6]   32Ø:    GruppenFarbe [StrGruppe [21+w] Mod 9]);
[XU3]   321:         FloodFill(x+1, 63, Black);
[ZNØ]   322:       end;
[C66]   323:       SchreibeStrasse(x+2, 9, 21+w, VertDir);
[ZNØ]   324:     end;
[D54]   325:     if(StrName [31+w] <> '') then
[MUØ]   326:     begin
[ZU4]   327:       if(StrGruppe [31+w] <> Ø) then
[MUØ]   328:       begin
[BX2]   329:         Line(416, x, 416, y);
[874]   33Ø:         SetFillStyle(SolidFill,
[TN6]   331:       GruppenFarbe [StrGruppe [31+w] Mod 9]);
[PZ3]   332:         FloodFill(4ØØ, x+1, Black);
[ZNØ]   333:       end;
[KB6]   334:       SchreibeStrasse(422,x+3,31+w,HorizDir);
[ZNØ]   335:     end;
[KK4]   336:     if(StrName [19 - w] <> '') then
[MUØ]   337:     begin
[CT4]   338:       if(StrGruppe [19 - w] <> Ø) then
[MUØ]   339:       begin
[TP2]   34Ø:         Line(62, x, 62, y);
[874]   341:         SetFillStyle(SolidFill,
[PN6]   342:       GruppenFarbe[StrGruppe[19-w] Mod 9]);
[ØW3]   343:         FloodFill(63, x+1, Black);
[ZNØ]   344:       end;
[H76]   345:       SchreibeStrasse(12,x+3,19-w,HorizDir);
[ZNØ]   346:     end;
[P94]   347:     if(StrName [9 - w] <> '') then
[MUØ]   348:     begin
[WW4]   349:       if(StrGruppe [9 - w] <> Ø) then
[MUØ]   35Ø:       begin
[CX2]   351:         Line(X, 417, y, 417);
[874]   352:         SetFillStyle(SolidFill,
[RH6]   353:       GruppenFarbe [StrGruppe[9-w] Mod 9]);
[634]   354:         FloodFill(x+1, 4ØØ, Black);
[ZNØ]   355:       end;
[756]   356:       SchreibeStrasse(x+2,418,9-w,VertDir);
[ZNØ]   357:     end;
[ZNØ]   358:   end;
[4N2]   359:   SetColor(Blue);
[ZG2]   36Ø:   for w:=1 To 3 do
[MUØ]   361:   begin
[WW6]   362:     SetTextStyle(SmallFont, GemRich[w], 4);
[M64]   363:     OutTextXY(GemXY[w    ].x,
[Z64]   364:               GemXY[w    ].y, 'Gemein-');
[L64]   365:     OutTextXY(GemXY[w+3].x,
[F44]   366:               GemXY[w+3].y, 'schafts-');
[YG6]   367:     OutTextXY(GemXY[w+6].x, GemXY[w+6].y,
[W21]   368:               'feld');
[ZNØ]   369:   end;
[5R2]   37Ø:   SetColor(Black);
[MV6]   371:   SetzeSchrift(SansSerifFont, HorizDir, 6,
[142]   372:                       CenterText);
[UY4]   373:   OutTextXY(24Ø, 24Ø, 'MONOPOLY');
[PD6]   374:   SetTextStyle(Smallfont, HorizDir, 7);
[C44]   375:   OutTextXY(437, 44Ø, 'LOS');
[AD6]   376:   SetTextStyle(Smallfont, HorizDir, 4);
[EZ2]   377:   OutTextXY(24Ø, 273,
[L56]   378:  '(c) 1993 by Andreas Knab -- DMV-Verlag');
[HS4]   379:   OutTextXY(42, 457, 'Nur zum');
[7C4]   38Ø:   OutTextXY(42, 465, 'Besuch');
[135]   381:   OutTextXY(42, 7Ø, 'Frei Parken');
[7S4]   382:   OutTextXY(437, 1Ø, 'Gehen Sie');
[H64]   383:   OutTextXY(437, 6Ø, 'in das');
[BY4]   384:   OutTextXY(437, 7Ø, 'Gefängnis');
[LZ4]   385:   OutTextXY(437, 337, 'Zusatz-');
[BR4]   386:   OutTextXY(437, 345, 'steuer');
[7H4]   387:   OutTextXY(437, 353, 'DM 2ØØØ');
[UM3]   388:   Rectangle(3Ø, 399, 8Ø, 449);
[SY5]   389:   SetFillStyle(SolidFill, LightRed);
[ØU2]   39Ø:   Bar(31, 4ØØ, 79, 448);
[F43]   391:   FillPoly(8, Pfeil);
[RF5]   392:   SetFillStyle(SolidFill, White);
[PX3]   393:   FillPoly(5, Gefaengnis);
[ZG2]   394:   for w:=1 To 3 do
[585]   395:     Line(35+w*5, 424+w*5, 55+w*5, 4Ø4+w*5);
[6D6]   396:   SetTextStyle(SmallFont, HorizDir, 3);
[S56]   397:   OutTextXY(439,4Ø8,'Ziehen Sie im Vor-');
[836]   398:   OutTextXY(439,416,'übergehen DM 4ØØØ');
[7Z4]   399:   OutTextXY(439,424,'Gehalt ein.');
[EU4]   4ØØ:   OutTextXY(135, 55, 'Ereignis-');
[N24]   4Ø1:   OutTextXY(135, 63, 'Feld');
[BØ5]   4Ø2:   OutTextXY(17Ø, 455, 'Ereignis-');
```

```
[934]  4Ø3:   OutTextXY(17Ø, 463, 'Feld');
[596]  4Ø4:   SetTextStyle(SmallFont, VertDir, 3);
[NØ5]  4Ø5:   OutTextXY(455, 271, 'Ereignis-');
[M34]  4Ø6:   OutTextXY(463, 271, 'Feld');
[896]  4Ø7:   SetTextStyle(SmallFont, VertDir, 4);
[185]  4Ø8:   OutTextXY(263, 437, 'Einkommen-');
[4R4]  4Ø9:   OutTextXY(273, 437, 'steuer');
[9H4]  41Ø:   OutTextXY(283, 437, 'DM 4ØØØ');
[4N2]  411:   SetColor(Blue);
[W37]  412:   SetTextStyle(SansSerifFont, HorizDir, 6);
[6Y5]  413:   OutTextXY(136, 2Ø, '?');  SetColor(Red);
[4K3]  414:   OutTextXY(171, 42Ø, '?');
[SZ6]  415:   SetTextStyle(SansSerifFont, VertDir, 6);
[YK3]  416:   OutTextXY(417, 27Ø, '?');
[GS4]  417:   Lokomotive(224, 6Ø, HorizDir);
[RY4]  418:   Lokomotive(224, 4Ø3, HorizDir);
[SH4]  419:   Lokomotive(6Ø, 224, VertDir);
[3P4]  42Ø:   Lokomotive(4Ø3, 224, VertDir);
[A15]  421:   SetFillStyle(SolidFill, Red);
[TØ5]  422:   Bar(25, 15, 55, 3Ø); Bar(2Ø, 3Ø, 6Ø, 45);
[RF5]  423:   SetFillStyle(SolidFill, White);
[725]  424:   Bar(3Ø, 18, 5Ø, 27); Bar(23, 46, 28, 51);
[WG2]  425:   Bar(52, 46, 57, 51);
[MK3]  426:   FillPoly(8, GefPfeil);
[E86]  427:   SetColor(Blue); DrawPoly(8, GefPfeil);
[H66]  428:   SetColor(White); Sector(4Ø,38,Ø,36Ø,5,5);
[1C5]  429:   SetColor(Black); Circle(4Ø, 38, 5);
[LN3]  43Ø:   PutPixel(4Ø, 38, Black);
[ZNØ]  431: end;
[MR4]  432: { --- Objekt: StrasseObj --- }
[265]  433: Constructor StrasseObj.Init;
[MUØ]  434: begin
[114]  435:   Gruppe:=g; Position:=p;
[ST2]  436:   Name:=n; Wert:=w;
[2Y4]  437:   Hypothek:=False; Bebauung:=Ø;
[YY1]  438:   Spieler:=Ø;
[ZNØ]  439: end;
[ØØØ]  44Ø:
[WR5]  441: procedure StrasseObj.SetzeMiete;
[MUØ]  442: begin
[4L4]  443:   Miete [1]:=1; Miete [2]:=h1;
[MK4]  444:   Miete [3]:=h2; Miete [4]:=h3;
[MF4]  445:   Miete [5]:=h4; Miete [6]:=h;
[ZNØ]  446: end;
[ØØØ]  447:
[AR5]  448: procedure StrasseObj.ZeigeKarte;
[RV3]  449: var p: Pointer; b: Byte;
[MUØ]  45Ø: begin
[7H2]  451:   if(Gruppe=Ø) and
[966]  452:    Not(Position In [12,28,5,15,25,35]) then
[ØW9]  453:    Exit;
[5R2]  454:   SetColor(Black);
[RF5]  455:   SetFillStyle(SolidFill, White);
[B46]  456:   SetLineStyle(SolidLn, Ø, NormWidth);
[973]  457:   Bar(X, Y, X+2ØØ, Y+25Ø);
[G96]  458:   SetzeSchrift(DefaultFont,HorizDir,1,
[142]  459:                CenterText);
[LA3]  46Ø:   if Not Hypothek then
[MUØ]  461:   begin
[3N4]  462:     Rectangle(X+2, Y+2, X+198, Y+248);
[HP5]  463:     if(Position In [12,28,5,15,25,35]) then
[MUØ]  464:     begin
[PD6]  465:       SetTextStyle(SmallFont, HorizDir, 7);
[AG4]  466:       OutTextXY(X+1ØØ, Y+32, Name);
[7Z5]  467:       SetzeSchrift(SmallFont,HorizDir,4,
[2Ø2]  468:                    RightText);
[PE3]  469:       OutTextXY(X+195,Y+66,
[ØL3]  47Ø:                 'DM '+StrNum(Wert, 4));
[LK3]  471:       OutTextXY(X+195,Y+232,
[A74]  472:                 'DM '+StrNum(Wert Div 2,4));
[HS6]  473:       SetTextJustify(LeftText, CenterText);
[7T6]  474:      OutTextXY(X+5,Y+66,'Grundstückswert');
[5L6]  475:      OutTextXY(X+5,Y+232,'Hypothekenwert');
[CU5]  476:      if(Position=12) Or(Position=28) then
[MUØ]  477:      begin
[N73]  478:        OutTextXY(X+5, Y+85,
[185]  479:        'Wenn man Besitzer des Wasserwer-');
[PF8]  48Ø:        OutTextXY(X+5, Y+96, 'kes ist, so ist
      t die Miete 8Ømal');
[6N8]  481:        OutTextXY(X+5, Y+1Ø7, 'so hoch, wie
      Augen auf den Wür-');
[9F5]  482:        OutTextXY(X+5, Y+118, 'feln sind.');
[B49]  483:        OutTextXY(X+5, Y+14Ø, 'Wenn man Besi
      tzer beider Werke,');
[U49]  484:        OutTextXY(X+5, Y+151, 'Wasser- und E
      lektrizitätswerk');
[ØH8]  485:        OutTextXY(X+5, Y+162, 'ist, so ist d
      ie Miete 2ØØmal so');
[EW8]  486:        OutTextXY(X+5, Y+173, 'hoch, wie Aug
      en auf den Würfeln');
[AS4]  487:        OutTextXY(X+5, Y+184, 'sind.');
[4MØ]  488:       end
[WP1]  489:       else begin
[UP4]  49Ø:         OutTextXY(X+5, Y+9Ø, 'Miete');
[A85]  491:         OutTextXY(X+5, Y+11Ø, 'Wenn man');
[H93]  492:         OutTextXY(X+5, Y+121,
[E34]  493:                   'zwei Bahnhöfe besitzt');
[R85]  494:         OutTextXY(X+5, Y+141, 'Wenn man');
[Y93]  495:         OutTextXY(X+5, Y+152,
[SØ4]  496:                   'drei Bahnhöfe besitzt');
[495]  497:         OutTextXY(X+5, Y+172, 'Wenn man');
[BA3]  498:         OutTextXY(X+5, Y+183,
[W14]  499:                   'vier Bahnhöfe besitzt');
[5U6]  5ØØ:     SetTextJustify(RightText, CenterText);
[BZ4]  5Ø1:         OutTextXY(X+195, Y+9Ø, 'DM  5ØØ');
[DZ4]  5Ø2:         OutTextXY(X+195,Y+121, 'DM 1ØØØ');
[LZ4]  5Ø3:         OutTextXY(X+195,Y+152, 'DM 2ØØØ');
[UZ4]  5Ø4:         OutTextXY(X+195,Y+183, 'DM 4ØØØ');
[ZNØ]  5Ø5:       end;
[4MØ]  5Ø6:      end
[WP1]  5Ø7:      else begin
[874]  5Ø8:        SetFillStyle(SolidFill,
[T64]  5Ø9:                     GruppenFarbe [Gruppe]);
[ØV3]  51Ø:        Bar(X+4, Y+4, X+196, Y+55);
[NT6]  511:     if (GruppenFarbe[Gruppe]<DarkGray) then
[CT2]  512:        SetColor(White);
[WG3]  513:       OutTextXY(X+1ØØ, Y+11,
[TG3]  514:                 'Besitzrechtskarte');
[5B6]  515:       SetTextStyle(SmallFont,HorizDir,7);
[AG4]  516:       OutTextXY(X+1ØØ, Y+32, Name);
[5R2]  517:       SetColor(Black);
[Z18]  518:       SetzeSchrift(SmallFont, HorizDir, 4,
      LeftText);
[NZ8]  519:       for b:=1 To 7 do OutTextXY(X+5, Y+55
      +b*11, PreisTxt [b]);
[YA4]  52Ø:       Line(X+5, Y+142, X+195, Y+142);
[7T4]  521:       SetUserCharSize(83, 1ØØ, 9, 1Ø);
[YZB]  522:       OutTextXY(X+5,Y+147,'Wenn ein Spiele
      r alle Grundstücke einer Farben-');
[7NB]  523:       OutTextXY(X+5,Y+156,'gruppe besitzt,
      so ist die Miete auf den unbe-');
[LDC]  524:       OutTextXY(X+5,Y+165,'bauten Grundstü
      cken dieser Farbengruppe doppelt');
[425]  525:       OutTextXY(X+5, Y+174, 'so hoch.');
[EB4]  526:       Line(X+5, Y+182, X+195, Y+182);
[AD6]  527:       SetTextStyle(SmallFont, HorizDir, 4)
      ;
[F16]  528:       OutTextXY(X+5, Y+195, '1 Haus kostet
      ');
[XH7]  529:       OutTextXY(X+5, Y+2Ø6, '1 Hotel `` 4
      Häuser und');
[5U6]  53Ø:       SetTextJustify(RightText, CenterText
      );
[YF2]  531:       for b:=2 To 7 do
[L85]  532:         OutTextXY(X+195, Y+55+b*11, 'DM '
      +
```

```
[4E4]  533:         StrNum(Miete [Pred(b)], 5));
[6T3]  534:       OutTextXY(X+195, Y+195,
[T46]  535:  'DM '+StrNum(Succ(Gruppe) Div 2*1ØØØ,4));
[SS3]  536:       OutTextXY(X+195, Y+2Ø6,
[P86]  537:  'DM '+StrNum(Succ(Gruppe) Div 2*1ØØØ, 4));
[1X3]  538:       OutTextXY(X+195, Y+55+11,
[1LJ]  539:  'DM '+StrNum(Wert, 5));
[TX6]  54Ø:     SetTextJustify(CenterText,CenterText);
[VP3]  541:     OutTextXY(X+1ØØ, Y+22Ø,
[2L5]  542:  'Der Hypothekenwert des Grund-');
[3R3]  543:     OutTextXY(X+1ØØ, Y+232,
[4B7]  544:  'stückes beträgt DM '+StrNum(Wert Div 2, 4
      ));
[ZNØ]  545:    end;
[4MØ]  546:  end
[WP1]  547:  else begin
[TD2]  548:   SetColor(Red);
[1K6]  549:   OutTextXY(X+1ØØ,Y+45,'Hypothekenbrief');
[RW3]  55Ø:   Line(X+5, Y+9Ø, X+195, Y+9Ø);
[PD6]  551:   SetTextStyle(SmallFont, HorizDir, 7);
[LP4]  552:   OutTextXY(X+1ØØ, Y+11Ø, Name);
[CB4]  553:   Line(X+5, Y+135, X+195, Y+135);
[AD6]  554:   SetTextStyle(SmallFont, HorizDir, 4);
[6R3]  555:   OutTextXY(X+1ØØ, Y+18Ø,
[LH4]  556:                'Die Karte muß auf diese');
[ER3]  557:   OutTextXY(X+1ØØ, Y+192,
[TD4]  558:                'Seite umgedreht werden,');
[3R3]  559:   OutTextXY(X+1ØØ, Y+2Ø4,
[Y75]  56Ø:         'wenn das Besitztum belastet');
[4W4]  561:   OutTextXY(X+1ØØ, Y+216, 'ist.');
[HS6]  562:   SetTextJustify(LeftText, CenterText);
[V66]  563:   OutTextXY(X+15, Y+16Ø, 'Belastet mit');
[5U6]  564:   SetTextJustify(RightText, CenterText);
[ES3]  565:   OutTextXY(X+185, Y+16Ø,
[7A4]  566:                'DM '+StrNum(Wert Div 2, 4));
[ZNØ]  567:  end;
[ZNØ]  568: end;
[ØØØ]  569:
[896]  57Ø: procedure StrasseObj.ZeigeBebauung;
[W75]  571: var x, y: Word; xs, ys: Integer;
[8X3]  572:     add : ShortInt; b: Byte;
[MUØ]  573: begin
[EA3]  574:   if(Wert=Ø) then Exit;
[LT3]  575:   Case(Position Div 1Ø) of
[G85]  576:     Ø: x:=81+Trunc(319/9*(9-Position));
[N41]  577:     1: x:=64;
[LF6]  578:     2: x:=81+Trunc(319/9*Pred(Position Mod
      1Ø));
[891]  579:     3: x:=4ØØ;
[ZNØ]  58Ø:   end;
[LT3]  581:   Case(Position Div 1Ø) of
[591]  582:     Ø: y:=4Ø1;
[B56]  583:     1: y:=81+Trunc(319/9*(9 - Position Mod
      1Ø));
[S41]  584:     2: y:=64;
[PF6]  585:     3: y:=81+Trunc(319/9*Pred(Position Mod
      1Ø));
[ZNØ]  586:   end;
[KG5]  587:   if Odd(Position Div 1Ø) then add:=Ø
[WY1]  588:   else add:=-1;
[ST5]  589:   if(StrGruppe [Position] <> Ø) then
[MUØ]  59Ø:   begin
[874]  591:     SetFillStyle(SolidFill,
[6K7]  592:   GruppenFarbe[StrGruppe[Position] Mod 9]);
[TU5]  593:     Bar(x,y,x+15-18*Add,y+15+18*Succ(Add));
[ZNØ]  594:   end;
[G33]  595:   if(Spieler <> Ø) then
[874]  596:     SetFillStyle(SolidFill,
[9X5]  597:          MonoInit.Spieler[Spieler]^.Farbe)
[ZRØ]  598:   else
[W96]  599:     SetFillStyle(SolidFill, LightGreen);
[ML2]  6ØØ:   xs:=17*succ(add);
[6V1]  6Ø1:   ys:= 17*add;
[666]  6Ø2:   if(Position < 1Ø) or(Position > 3Ø) then
[MUØ]  6Ø3:   begin
[634]  6Ø4:     if(xs <> Ø) then xs:=-xs+4;
[E34]  6Ø5:     if(ys <> Ø) then ys:=-ys+4;
[ZNØ]  6Ø6:   end;
[4N3]  6Ø7:   inc(xs, x); inc(ys, y);
[NA6]  6Ø8:   if(Position in [1,9,11,19,21,29,31,39]) t
      hen
[MUØ]  6Ø9:   begin
[CVØ]  61Ø:     b:=12;
[SS2]  611:     case Position of
[FL2]  612:       9, 21 : Inc(xs, 11);
[XP2]  613:       19, 31: Inc(ys, 11);
[ZNØ]  614:     end;
[4MØ]  615:   end
[LS1]  616:   else b:=23;
[VB3]  617:   Bar(xs, ys, xs+1Ø - b*Add,
[FX2]  618:                ys+1Ø+b*Succ(add));
[G33]  619:   if(Spieler <> Ø) then
[MUØ]  62Ø:   begin
[WCA]  621:     if(MonoInit.Spieler [Spieler]^.Farbe >
      7) then SetColor(Black)
[LM3]  622:     else SetColor(White);
[NV4]  623:     Rectangle(xs, ys, xs+1Ø - b*add,
[FX2]  624:                       ys+1Ø+b*succ(add));
[1B6]  625:     SetLineStyle(SolidLn, Ø, ThickWidth);
[BR2]  626:     if Hypothek then
[GA4]  627:       Line(xs+1, ys+1, xs+9 - b*add,
[SP2]  628:                        ys+9+b*succ(add));
[B46]  629:     SetLineStyle(SolidLn, Ø, NormWidth);
[ZNØ]  63Ø:   end;
[2Y3]  631:   if(Bebauung=Ø) then Exit;
[C13]  632:   if(Bebauung=5) then
[MUØ]  633:   begin
[245]  634:     Dec(x, 1Ø*Add); Inc(y, 1Ø*Succ(Add));
[CT2]  635:     SetColor(White);
[A15]  636:     SetFillStyle(SolidFill, Red);
[7Ø3]  637:     Bar(x, y, x+15, y+15);
[154]  638:     Rectangle(x, y, x+15, y+15);
[7E3]  639:     Line(x+8, y, x+8, y+15);
[ØWØ]  64Ø:     Exit;
[ZNØ]  641:   end;
[1R3]  642:   SetColor(LightGreen);
[8F5]  643:   SetFillStyle(SolidFill, Green);
[XN3]  644:   b:=32 Div(Bebauung*2) - 2;
[MW3]  645:   Inc(x, 4*Succ(Add) - b*Add);
[UV3]  646:   Inc(y, b*Succ(Add) - 4*Add);
[WY3]  647:   for b:=1 To Bebauung do
[MUØ]  648:   begin
[CN2]  649:     Bar(x, y, x+7, y+7);
[UX3]  65Ø:     Rectangle(x, y, x+7, y+7);
[DC5]  651:     Line(x, y+4, x+7, y+4); Dec(x, 8*Add);
[AX2]  652:     Inc(y, 8*Succ(Add));
[ZNØ]  653:   end;
[ZNØ]  654: end;
[ØØØ]  655:
[9S5]  656: procedure StrasseObj.ZeigeTitel;
[MUØ]  657: begin
[A15]  658:   if Hypothek then SetColor(Red)
[ZRØ]  659:   else
[8T6]  66Ø:  if(GruppenFarbe [Gruppe] < DarkGray) and
[K95]  661:     (GruppenFarbe [Gruppe] > Ø) then
[HR2]  662:     SetColor(White)
[SG3]  663:  else SetColor(Black);
[D73]  664:  if Not Hypothek And
[WD5]  665:     (GruppenFarbe [Gruppe] <> Ø) then
[874]  666:   SetFillStyle(SolidFill,
[DZ3]  667:                GruppenFarbe [Gruppe])
[H86]  668:   else SetFillStyle(SolidFill, White);
[613]  669:   Bar(X, Y, X+15Ø, Y+8);
[256]  67Ø:   SetzeSchrift(SmallFont, HorizDir, 4,
[142]  671:                              CenterText);
```

```
[4D4]  672:   OutTextXY(X+75, Y+3, Name);
[ZNØ]  673: end;
[ØØØ]  674:
[4L6]  675: function StrasseObj.BebauungErlaubt;
[VM4]  676: var b   : Array [1..3] of Byte;
[815]  677:     z, c: ShortInt; min, max: Byte;
[MUØ]  678: begin
[XZ3]  679:   BebauungErlaubt:=True;
[7W6]  68Ø:  if(Spieler=Ø) then BebauungErlaubt:=False;
[FPØ]  681:   c:=1;
[EW5]  682:   for z:=Position - 3 To Position+3 do
[BH3]  683:    if(z > Ø) And(z < 41) then
[246]  684:      if(Strasse [z]^.Gruppe=Gruppe) then
[MUØ]  685:      begin
[K15]  686:        b [c]:=Strasse [z]^.Bebauung;
[X41]  687:        Inc(c);
[7M6]  688:      if(Strasse [z]^.Spieler<>Spieler) then
[C54]  689:        BebauungErlaubt:=False;
[ZNØ]  69Ø:     end;
[HM3]  691:   Dec(c); min:=5; max:=Ø;
[5K2]  692:   for z:=1 To c do
[MUØ]  693:   begin
[Y45]  694:     if(b [z] < min) then min:=b [z];
[W45]  695:     if(b [z] > max) then max:=b [z];
[ZNØ]  696:   end;
[YH2]  697:   if(max=min) then
[MUØ]  698:   begin
[8U3]  699:     if(max < 5) then Inc(max);
[GS3]  7ØØ:     if(min > Ø) then Dec(min);
[ZNØ]  7Ø1:   end;
[ØY3]  7Ø2:   if(h < min) Or(h > max) then
[C54]  7Ø3:               BebauungErlaubt:=False;
[ZNØ]  7Ø4: end;
[ØØØ]  7Ø5:
[YD5]  7Ø6: function StrasseObj.HoleMiete;
[643]  7Ø7: var b, c, d: Integer;
[MUØ]  7Ø8: begin
[EU5]  7Ø9:   HoleMiete:=Miete [Succ(Bebauung)];
[AA6]  71Ø:   if(Bebauung<>Ø) Or(Spieler=Ø) then Exit;
[573]  711:   if Odd(Position) And
[EM3]  712:        (Position Mod 5=Ø) then
[MUØ]  713:   begin
[R11]  714:     b:=5ØØ;
[PF2]  715:     for c:=Ø To 3 do
[837]  716:   if(Strasse[c*1Ø+5]^.Spieler=Spieler) then
[XB1]  717:       Inc(b, b);
[543]  718:     HoleMiete:=b Div 2;
[ØWØ]  719:     Exit;
[ZNØ]  72Ø:   end;
[CU5]  721:   if(Position=12) Or(Position=28) then
[MUØ]  722:   begin
[U79]  723:     if (Strasse [12]^.Spieler=Strasse [28]^
      .Spieler) then
[UF4]  724:       HoleMiete:=LetzterWurf*2Ø
[AG5]  725:     else HoleMiete:=LetzterWurf*8Ø;
[ØWØ]  726:     Exit;
[ZNØ]  727:   end;
[GPØ]  728:   d:=Ø;
[YU5]  729:   for b:=Position - 3 to Position+3 do
[WL3]  73Ø:     if(b >= 1) and(b <= 4Ø) and
[BP5]  731:       (strasse[b]^.gruppe=gruppe) and
[546]  732:       (strasse[b]^.spieler=spieler) then
[Z41]  733:         inc(d);
[GP5]  734:   if(d=3) then holemiete:=2*Miete[1];
[ZNØ]  735: end;
[ØØØ]  736:
[H36]  737: procedure StrasseObj.SetzeSpieler;
[MUØ]  738: begin
[UR4]  739:   Spieler:=n;  ZeigeBebauung;
[ZNØ]  74Ø: end;
[ØØØ]  741:
[UP4]  742: { --- Objekt: SpielerObj --- }
[K45]  743: Constructor SpielerObj.Init;
[MUØ]  744: begin
[VU4]  745:   Farbe:=f; Nummer:=n; Name:=na;
[R64]  746:   Strassen:=[]; Geld:=3ØØØØ;
[GB1]  747:   Feld:=4Ø;
[NN5]  748:   GetMem(Puffer, ImageSize(Ø, Ø, 14, 14));
[AW1]  749:   Fx:=Ø; Fy:=Ø;
[DH4]  75Ø:   Gefangen:=Ø; Pleite:=False;
[FX2]  751:   Bezahlend:=False;
[ZNØ]  752: end;
[ØØØ]  753:
[P15]  754: Destructor SpielerObj.Fertig;
[MUØ]  755: begin
[6D3]  756:   Feld:=-1; ZeigeFigur;
[F26]  757:   FreeMem(Puffer, ImageSize(Ø, Ø, 14, 14));
[ZNØ]  758: end;
[ØØØ]  759:
[565]  76Ø: procedure SpielerObj.StrDazu;
[MUØ]  761: begin
[R84]  762:   Strassen:=Strassen+[n];
[ZNØ]  763: end;
[ØØØ]  764:
[VØ5]  765: procedure SpielerObj.StrWeg;
[MUØ]  766: begin
[UB4]  767:   Strassen:=Strassen - [n];
[ZNØ]  768: end;
[ØØØ]  769:
[5R5]  77Ø: procedure SpielerObj.ZeigeFigur;
[243]  771: var add: ShortInt;
[MUØ]  772: begin
[Y72]  773:   if(Fx <> Ø) then
[TD6]  774:   PutImage(Fx, Fy, Puffer^, NormalPut);
[9D3]  775:   if(Feld=-1) then Exit;
[EA3]  776:   if(Gefangen <> Ø) then
[MUØ]  777:   begin
[GB2]  778:     Fx:=31; Fy:=41Ø;
[4MØ]  779:   end
[WP1]  78Ø:   else begin
[HY2]  781:    Case(Feld Div 1Ø) of
[RN4]  782:      Ø: Fx:=81+Trunc(319/9*(9 - Feld));
[AB1]  783:      1: Fx:=1Ø;
[3Ø6]  784:      2: Fx:=81+Trunc(319/9*Pred(Feld Mod 1Ø
      ));
[4T1]  785:    3,4: Fx:=42Ø;
[ZNØ]  786:    end;
[HY2]  787:    Case(Feld Div 1Ø) of
[WH1]  788:      Ø: Fy:=42Ø;
[SF5]  789:      1:Fy:=81+Trunc(319/9*(9-Feld Mod 1Ø));
[EB1]  79Ø:      2: Fy:=1Ø;
[7Ø6]  791:  3: Fy:=81+Trunc(319/9*Pred(Feld Mod 1Ø));
[4K1]  792:      4: Fy:=42Ø;
[ZNØ]  793:    end;
[ZNØ]  794:   end;
[8S4]  795:   if(Gefangen=Ø) And(Feld=1Ø) then
[YX2]  796:     if(Nummer < 4) then
[MUØ]  797:     begin
[BØ5]  798:       Fx:=1Ø; Fy:=4Ø5+Pred(Nummer)*17;
[4MØ]  799:     end
[WP1]  8ØØ:     else begin
[LL4]  8Ø1:       Fx:=38+(Nummer - 4)*17; Fy:=455;
[4MØ]  8Ø2:     end
[WP1]  8Ø3:   else begin
[5R4]  8Ø4:     if Odd(Feld Div 1Ø) then add:=-1
[WV1]  8Ø5:     else add:=Ø;
[5D5]  8Ø6:       Dec(Fx,Trunc(Nummer/2-Ø.5)*add*17);
[DB6]  8Ø7:   Inc(Fy,Trunc(Nummer/2-Ø.5)*Succ(add)*17);
[3Ø5]  8Ø8:       Dec(Fy, Succ(Nummer) Mod 2*add*17);
[5B6]  8Ø9:   Inc(Fx, Succ(Nummer) Mod 2*Succ(add)*17);
[ZNØ]  81Ø:   end;
```

Listing 2. »Monoinit.pas« führt die notwendigen Initialisierungen durch, etwa für die Mieteinnahmen.

```
[N86]  811:   GetImage(Fx, Fy, Fx+14, Fy+14, Puffer^);
[KE5]  812:   SetFillStyle(SolidFill, Farbe);
[2P2]  813:   SetColor(Farbe);
[LA4]  814:   Sector(Fx+7, Fy+7, Ø, 36Ø, 7, 7);
[H46]  815:   SetColor(White); Circle(Fx+7, Fy+7, 7);
[ZNØ]  816: end;
[ØØØ]  817:
[TS5]  818: procedure SpielerObj.BewegeFigur;
[MUØ]  819: begin
[EØ2]  82Ø:   Inc(Feld, n);
[4L4]  821:   if(Feld > 4Ø) then Dec(Feld, 4Ø);
[YG4]  822:   if(Feld < Ø) then Inc(Feld, 4Ø);
[ES3]  823:   ZeigeFigur; Delay(1ØØ);
[ZNØ]  824: end;
[ØØØ]  825:
[W75]  826: procedure SpielerObj.Bezahlen;
[775]  827: var d: AuswahlObj; e: DialogObj;
[PL1]  828:     s: String;
[MUØ]  829: begin
[2U2]  83Ø:   Bezahlend:=True;
[P31]  831:   Repeat
[4N4]  832:     if(Nummer=SpielerDran) then
[RK3]  833:                ZeigeSpielerStatus;
[9D3]  834:     if(Strassen=[]) then
[2D3]  835:       s:='#Hypothek\#Bauen\#Verkaufen\Ok\?'
[WP1]  836:     else begin
[ØB6]  837:       s:='Hypothek\Bauen\Verkaufen\Ok\?';
[Z95]  838:       if(geld < p) then Insert('#', s, 26);
[ZNØ]  839:     end;
[GH4]  84Ø:     d.HoleEingabe(#Ø#33'Bezahlen',
[DP8]  841:       Name+', Sie müssen\DM '+ StrNum(p, Ø)
+' bezahlen.', s);
[7Ø5]  842:     if(d.Result=4) then Dec(Geld, p)
[DW1]  843:     else  begin
[6L3]  844:       e.Zeige(15Ø, 19Ø, 33Ø, 29Ø);
[Z73]  845:       e.SetOpt('Bezahlen',
[YT7]  846:     'Wählen Sie die\gewünschte\Straße aus.'
, '');
[FX3]  847:       HoleTaste(true);  e.Weg;
[XM2]  848:       Case d.Result of
[S93]  849:         1: Hypothek(Nummer);
[WE3]  85Ø:         2: BauAktion(Nummer);
[4T2]  851:         3: Handel(Nummer);
[ZNØ]  852:       end;
[ZNØ]  853:     end;
[1Y2]  854:     Pleite:=(Geld < Ø);
[DS4]  855:   until(d.Result=4) Or Pleite;
[FD4]  856:   if f then AufFreiParken(p);
[FX2]  857:   Bezahlend:=False;
[ZNØ]  858: end;
[ØØØ]  859:
[F74]  86Ø: procedure AufFreiParken;
[6M4]  861: var n, k: byte; s: set of byte;
[764]  862:     a: AuswahlObj; p: Longint;
[MUØ]  863: begin
[4Ø2]  864:   n:=Ø; s:=[];
[9G4]  865:   for k:=1 to SpielerZahl do
[GM4]  866:     if(Spieler[k]^.Feld=2Ø) then
[MUØ]  867:     begin
[MP2]  868:       inc(n); s:=s+[k];
[ZNØ]  869:     end;
[2S1]  87Ø:   if(n=Ø) then
[MUØ]  871:   begin
[834]  872:     Inc(FreiParken,.1); exit;
[ZNØ]  873:   end;
[AT1]  874:   p:=1 div n;
[9G4]  875:   for k:=1 to SpielerZahl do
[EA2]  876:     if(k in s) then
[MUØ]  877:     begin
[3E4]  878:       a.HoleEingabe('Frei Parken',
[SL16] 879:   Spieler[k]^.Name+', Sie erhalten\' +
[HR7]  88Ø:   'DM '+StrNum(p, 6)+' von 'Frei Parken« !'
, 'Ok');
[MB4]  881:       Inc(Spieler[k]^.Geld, p);
[ZNØ]  882:     end;
[ZNØ]  883: end;
[ØØØ]  884:
[WL4]  885: function FeldNummer: Byte;
[PB2]  886: var v, w: Byte;
[MUØ]  887: begin
[R93]  888:   FeldNummer:=Ø;  w:=Ø;
[7P4]  889:   if (CurX > 475) Or (CurX > 8Ø) And
[NG4]  89Ø:     (CurX < 4ØØ) And (CurY > 8Ø) And
[NT5]  891:     (CurY < 4ØØ) then Exit;
[NT5]  892:   if (CurX<8Ø) And (CurY > 4ØØ) then w:=1Ø;
[ET5]  893:   if(CurX < 8Ø) And(CurY < 8Ø) then w:=2Ø;
[8V5]  894:   if(CurX > 4ØØ) And(CurY < 8Ø) then w:=3Ø;
[VX5]  895:   if(CurX>4ØØ) And (CurY>4ØØ) then w:=3Ø;
[D52]  896:   if(w <> Ø) then
[MUØ]  897:   begin
[AE3]  898:     FeldNummer:=w; Exit;
[ZNØ]  899:   end;
[4G3]  9ØØ:   if(CurX < 81) then w:=1Ø
[5Ø6]  9Ø1:   else if(CurX > 4ØØ) And(CurX < 475) then
[3RØ]  9Ø2:     w:=3Ø
[XN3]  9Ø3:   else if(CurX < 4ØØ) then
[1L4]  9Ø4:    w:=(CurX - 8Ø) Div Trunc(319/9)+1
[GS1]  9Ø5:   else Exit;
[145]  9Ø6:   if(CurY > 4ØØ) And(CurY < 475) then
[361]  9Ø7:     w:=1Ø - w
[3Ø5]  9Ø8:   else if(CurY < 81) then Inc(w, 2Ø)
[YM3]  9Ø9:   else if(CurY < 4ØØ) then
[MUØ]  91Ø:   begin
[HS4]  911:     v:=(CurY - 8Ø) Div Trunc(319/9)+1;
[D84]  912:     if(CurX < 2ØØ) then v:=1Ø - v;
[VD1]  913:     Inc(w, v);
[4MØ]  914:   end
[GS1]  915:   else Exit;
[FF2]  916:   FeldNummer:=w;
[ZNØ]  917: end;
[ØØØ]  918:
[3H2]  919: procedure BauAktion;
[5A4]  92Ø: var b, h: Byte; s, t: String;
[9P5]  921:     k, u: ShortInt; weg, dazu: Boolean;
[VH2]  922:     d : AuswahlObj;
[MUØ]  923: begin
[RC2]  924:   b:=FeldNummer;
[7S2]  925:   if(b=Ø) then Exit;
[TH3]  926:   With Strasse [b]^ do
[MUØ]  927:   begin
[8Ø2]  928:     h:=Bebauung;
[LJ5]  929:     weg:=BebauungErlaubt(Pred(h));
[EH5]  93Ø:     dazu:=BebauungErlaubt(Succ(h)) and
[5N6]  931:       not MonoInit.Spieler [Sp]^.Bezahlend;
[ZNØ]  932:     end;
[UP3]  933:   with Spieler [Sp]^ do
[DW3]  934:     if Not(b In Strassen) Or
[334]  935:             ((weg Or dazu)=False) then
[MUØ]  936:     begin
[H14]  937:       d.HoleEingabe(#Ø#34'Bauen',
[A96]  938:   'Da können Sie\nicht bauen !', 'Ok\?');
[ØWØ]  939:       Exit;
[4MØ]  94Ø:     end
[WP1]  941:     else begin
[DA1]  942:       Case h of
[3H2]  943:         Ø: s:='unbebaut';
[L22]  944:         1: s:='1 Haus';
[XP2]  945:         2: s:='2 Häuser';
[ØR2]  946:         3: s:='3 Häuser';
[3R2]  947:         4: s:='4 Häuser';
[G62]  948:         5: s:='1 Hotel';
[ZNØ]  949:       end;
[9E5]  95Ø:       t:='Ab\1 Haus dazu\1 Haus weg\?';
[G45]  951:       if not weg  then Insert('#', t, 16);
[EB5]  952:       if not dazu then Insert('#', t, 4);
```

```
[N14]  953:       d.HoleEingabe(#Ø#35'Bauen',
[656]  954:     'Wie wollen Sie die\Bebauungsform ' +
[FK4]  955:     'ändern ?\Aktuell: '+s, t);
[TH3]  956:       With Strasse [b]^ do
[MUØ]  957:       begin
[633]  958:         if(d.Result=2) then
[MUØ]  959:         begin
[HU4]  96Ø:           Inc(Bebauung); ZeigeBebauung;
[UL5]  961:  Bezahlen(Succ(Position Div 1Ø)*1ØØØ,
[361]  962:             False);
[ZNØ]  963:         end;
[833]  964:       if(d.Result=3) then
[MUØ]  965:       begin
[CT4]  966:         Dec(Bebauung); ZeigeBebauung;
[8H5]  967:         Inc(Geld,Succ(Position Div 1Ø)*5ØØ);
[ZNØ]  968:       end;
[ZNØ]  969:     end;
[ZNØ]  97Ø:   end;
[ZNØ]  971: end;
[ØØØ]  972:
[GF3]  973: procedure Hypothek;
[ZC4]  974: var d: AuswahlObj; b: Byte;
[MUØ]  975: begin
[RC2]  976:   b:=FeldNummer;
[7S2]  977:   if(b=Ø) then Exit;
[UP3]  978:   with Spieler [Sp]^ do
[MUØ]  979:   begin
[DW3]  98Ø:     if Not(b In Strassen) Or
[ZA5]  981:      (Strasse [b]^.Bebauung <> Ø) then
[MUØ]  982:     begin
[LL4]  983:       d.HoleEingabe(#Ø#36'Hypothek',
[BD8]  984: 'Auf diese Straße können\Sie keine Hypothek
auf' +
[HV2]  985: 'nehmen !', 'Ok\?');
[ØWØ]  986:       Exit;
[ZNØ]  987:     end;
[TH3]  988:     With Strasse [b]^ do
[BR2]  989:       if Hypothek then
[KC3]  99Ø:        if not Bezahlend then
[MUØ]  991:        begin
[RL4]  992:          d.HoleEingabe(#Ø#37'Hypothek',
[ØH7]  993: 'Wollen Sie die Hypothek\in Höhe von DM ' +
[DR5]  994:  StrNum(Wert Div 2, Ø)+'\+ 1Ø% Zins ab' +
[LN3]  995:  'bezahlen ?', 'Ok\Ab\?');
[6Ø4]  996:          if(d.Result=2) then Exit;
[D25]  997:          Bezahlen(Round((Wert Div 2)*1.1),
[361]  998:                   False);
[DV2]  999:          Hypothek:=False;
[PF2]  1ØØØ:         ZeigeBebauung;
[4MØ]  1ØØ1:       end
[PH5]  1ØØ2:       else d.HoleEingabe(#Ø#38'Hypothek',
[MR7]  1ØØ3: 'Sie können jetzt keine\Hypothek abzahlen !
',
[U21]  1ØØ4:       'Ok\?')
[WP1]  1ØØ5:       else begin
[ZL4]  1ØØ6:         d.HoleEingabe(#Ø#39'Hypothek',
[7H7]  1ØØ7: 'Wollen Sie eine Hypothek\in Höhe von DM '+
[7T5]  1ØØ8:       StrNum(Wert Div 2, Ø)+'\aufnehmen ?',
[FV1]  1ØØ9:            'Ok\Ab\?');
[6Ø4]  1Ø1Ø:         if(d.Result=2) then Exit;
[HA5]  1Ø11:         Hypothek:=True; ZeigeBebauung;
[7C3]  1Ø12:         Inc(Geld, Wert Div 2);
[ZNØ]  1Ø13:       end;
[ZNØ]  1Ø14:   end;
[ZNØ]  1Ø15: end;
[ØØØ]  1Ø16:
[364]  1Ø17: function WaehleSpieler;
[GC6]  1Ø18: var t, u: string; b: byte; d: AuswahlObj;
[MUØ]  1Ø19: begin
[G41]  1Ø2Ø:   t:='Ab';
[RF4]  1Ø21:   for b:=1 To SpielerZahl do
[MUØ]  1Ø22:   begin
[H26]  1Ø23:     if(Length(Spieler [b]^.Name) > 9) then
[646]  1Ø24:    u:='\'+Copy(Spieler [b]^.Name, 1, 8)+'.'
[U35]  1Ø25:     else u:='\'+Spieler [b]^.Name;
[436]  1Ø26:     if(b=SpielerDran) then Insert('#',u,2);
[Z61]  1Ø27:       t:=t+u;
[ZNØ]  1Ø28:     end;
[RG1]  1Ø29:   t:=t+'\?';
[ZC6]  1Ø3Ø:   d.HoleEingabe(#Ø#4Ø'Spieler auswählen',
[TSØ]  1Ø31:                          s,t);
[K55]  1Ø32:   WaehleSpieler:=Pred(d.Result);
[ZNØ]  1Ø33: end;
[ØØØ]  1Ø34:
[UW2]  1Ø35: procedure Handel;
[K95]  1Ø36: var d: AuswahlObj; f: EingabeObj;
[7F3]  1Ø37:     n, s: Byte; b,sp: Byte;
[WW5]  1Ø38:     t, u: String; p: LongInt; q: Integer;
[MUØ]  1Ø39: begin
[FD2]  1Ø4Ø:   n:=FeldNummer;
[LS2]  1Ø41:   if(n=Ø) then Exit;
[294]  1Ø42:   s:=Strasse [n]^.Spieler;
[7S1]  1Ø43:   if(s=Ø) then
[MUØ]  1Ø44:   begin
[DN5]  1Ø45:     d.HoleEingabe(#Ø#41'Straße kaufen',
[RK8]  1Ø46:  'Mit dieser Straße kann\nicht gehandelt we
rden !',
[N41]  1Ø47:       'Ok\?');
[ØWØ]  1Ø48:     Exit;
[ZNØ]  1Ø49:   end;
[AA5]  1Ø5Ø:   if(Strasse [n]^.Gruppe <> Ø) then
[573]  1Ø51:   for b:=n - 3 To n+3 do
[143]  1Ø52:     if(b > Ø) And(b < 41) And
[NED]  1Ø53:    (Strasse [b]^.Gruppe=Strasse [n]^.Gruppe)
 And (Strasse [b]^.Bebauung <> Ø) then
[MUØ]  1Ø54:     begin
[HN5]  1Ø55:       d.HoleEingabe(#Ø#42'Straße kaufen',
[MA7]  1Ø56:  'Hier müssen zuerst noch einige\Häuser ' +
[V64]  1Ø57:                  'abgebaut werden !', 'Ok\?');
[ØWØ]  1Ø58:       Exit;
[ZNØ]  1Ø59:     end;
[H82]  1Ø6Ø:   if(s=Spi) then
[MUØ]  1Ø61:   begin
[ZBB]  1Ø62:     d.Result:=WaehleSpieler('Bitte handeln
Sie aus,\wer die Straße\' +
[EN6]  1Ø63:     Strasse [n]^.Name+'\kaufen will,\und '+
[SH6]  1Ø64:     'klicke den\entsprechenden Namen an.');
[2Ø4]  1Ø65:     if(d.Result=Ø) then Exit;
[T72]  1Ø66:     sp:=d.Result;
[4MØ]  1Ø67:   end
[ZRØ]  1Ø68:   else
[8Z5]  1Ø69:     if Not Spieler [Spi]^.Bezahlend then
[MUØ]  1Ø7Ø:     begin
[NN5]  1Ø71:       d.HoleEingabe(#Ø#43'Straße kaufen',
[WB3]  1Ø72:         Spieler [s]^.Name +
[CE5]  1Ø73:       ', verkaufen\Sie die Straße\' +
[Y64]  1Ø74:        Strasse [n]^.Name+'\an ' +
[WW5]  1Ø75:        Spieler [Spi]^.Name+' ?', 'Ok\Ab\?');
[6Ø4]  1Ø76:       if(d.Result=2) then Exit;
[GD1]  1Ø77:       sp:=Spi;
[4MØ]  1Ø78:     end
[WP1]  1Ø79:     else begin
[TN5]  1Ø8Ø:       d.HoleEingabe(#Ø#44'Straße kaufen',
[EX5]  1Ø81:       Spieler [Spi]^.Name+', Sie können' +
[5X6]  1Ø82:     ' jetzt\keine Straße kaufen.', 'Ok\?');
[ØWØ]  1Ø83:       exit;
[ZNØ]  1Ø84:     end;
[KN5]  1Ø85:   Val(f.Eingabe(#Ø#45'Straße kaufen',
[SC3]  1Ø86:       'Geben Sie den ausge-' +
[VA5]  1Ø87:       '\machten Preis ein:', '1234567890',
[ØH1]  1Ø88:         15Ø), p, q);
[T44]  1Ø89:   if(p=Ø) Or(q <> Ø) then Exit;
[EZ4]  1Ø9Ø:   if Strasse[n]^.Hypothek then
[MUØ]  1Ø91:   begin
[5E4]  1Ø92:     d.HoleEingabe(#Ø#46'Hinweis',
```

```
[UR6]  1Ø93:       'Auf der Straße liegt eine Hypothek,' +
[368]  1Ø94:       '\auf die 1Ø% Zinsen bezahlt werden müs
sen.', 'Ok');
[CA5]  1Ø95:       Inc(p, Strasse[n]^.Wert div 2Ø);
[ZNØ]  1Ø96:     end;
[UP3]  1Ø97:     with Spieler [sp]^ do
[MUØ]  1Ø98:     begin
[285]  1Ø99:       Bezahlen(p, False); StrDazu(n);
[ZNØ]  11Ø9:     end;
[CH3]  11Ø1:     with Spieler [s]^ do
[MUØ]  11Ø2:     begin
[NT3]  11Ø3:       Inc(Geld, p); StrWeg(n);
[ZNØ]  11Ø4:     end;
[XE5]  11Ø5:     Strasse [n]^.SetzeSpieler(sp);
[ZNØ]  11Ø6: end;
[KNØ]  11Ø7: end.
```

Listing 2. »Monoinit.pas« führt die notwendigen Initialisierungen durch, etwa für die Mieteinnahmen.

```
[YT3]    1: (Unit:     monofeld.pas
[8H6]    2:  Funktion: Vereinbarung von Spielregeln
[G64]    3:  Sprache:  Turbo Pascal 6.Ø
[ZC3]    4:  Autor:    Andreas Knab
[915]    5:  (c) 1993 DMV Widuch GmbH & Co. KG)
[ØØØ]    6:
[VA5]    7: ($A+,B-,D-,E+,F-,I-,L-,N-,O-,R-,S-,V-)
[6E2]    8: unit MonoFeld;
[SK1]    9: Interface
[PT5]   1Ø: uses MonoInit, MonoDial, MonoCnst,
[PR2]   11:      MonoKart, Graph;
[ØØØ]   12:
[834]   13: procedure Ereignisfeld;
[UY4]   14: procedure Gemeinschaftsfeld;
[2A4]   15: procedure Zusatzsteuer;
[7Z6]   16: procedure VersteigereStrasse(n, t: Byte;
[9F2]   17:                              alle: Boolean);
[154]   18: procedure KaufeStrasse;
[W45]   19: procedure VerarbeiteStrasse;
[ØØØ]   2Ø:
[4L2]   21: Implementation
[834]   22: procedure Ereignisfeld;
[UC4]   23: var z, b: Byte; d: DialogObj;
[184]   24:     a: AuswahlObj; p: Pointer;
[SC4]   25:     i: Integer; r, o: ShortInt;
[3W1]   26:     l: LongInt;
[MUØ]   27: begin
[WD4]   28:   i:=ImageSize(1ØØ, 165, 4ØØ, 315);
[PY1]   29:   GetMem(p, i);
[VE4]   3Ø:   GetImage(1ØØ, 165, 4ØØ, 315, p^);
[Y23]   31:   Inc(EreigKartenZ);
[5W6]   32:  if (EreigKartenZ>16) then EreigKartenZ:=1;
[SD6]   33:   z:=EreigKartenStack [EreigKartenZ];
[CP5]   34:   ErgKarten [z]^.ZeigeKarte(1ØØ, 165);
[VK3]   35:   d.Zeige(41Ø, 25Ø, 49Ø, 315);
[B15]   36:   d.SetOpt(#Ø#22'Karte', '', 'Ok\?');
[RK2]   37:   d.Eingabe; d.Weg;
[4E5]   38:   PutImage (1ØØ, 165, p^, NormalPut);
[1D2]   39:   FreeMem (p, i);
[LH5]   4Ø:   with Spieler [SpielerDran]^ do
[MUØ]   41:   begin
[Z91]   42:     o:=Feld;
[3C1]   43:     Case z Of
[351]   44:     1: begin
[ØRØ]   45:          l:=Ø;
[AH2]   46:          for b:=1 to 4Ø do
[TH3]   47:            with Strasse [b]^ do
[155]   48:              if (Spieler = SpielerDran) then
[D43]   49:                if (Bebauung<5) then
[UZ2]   5Ø:                  Inc(l, Bebauung*8ØØ)
[9M2]   51:                else Inc(l, 23ØØ);
[6T4]   52:                if (l>Ø) then Bezahlen(l, True);
[ZNØ]   53:        end;
[551]   54:     2: begin
[BØ5]   55:          if (Feld<1Ø) then r:=1 else r:=-1;
[VS4]   56:            if (r=1) then for b:=Feld to 8 do
[CC2]   57:                       BewegeFigur(l)
[6M4]   58:            else for b:=Feld downto 12 do
[VL2]   59:                       BewegeFigur(-1);
[RG4]   6Ø:            Gefangen:=3; BewegeFigur(r);
[AE1]   61:            Pasch:=Ø;
[ZNØ]   62:        end;
[FH3]   63:     3: Bezahlen(4ØØ, True);
[951]   64:     4: begin
[ØRØ]   65:          l:=Ø;
[AH2]   66:          for b:=1 to 4Ø do
[TH3]   67:           with Strasse [b]^ do
[EZ4]   68:             if (Spieler=SpielerDran) then
[D43]   69:               if (Bebauung<5) then
[U43]   7Ø:                 Inc(l, Bebauung * 5ØØ)
[ØM2]   71:               else Inc(l, 2ØØØ);
[PU4]   72:             if (l>Ø) then Bezahlen(l,True);
[ZNØ]   73:        end;
[MS2]   74:     5: if (Feld<15) then
[PG5]   75:          for b:=Feld to 14 do BewegeFigur(1)
[WP1]   76:          else begin
[882]   77:            Inc(Geld, 4ØØØ);
[W23]   78:            for b:=Feld to 54 do
[SL2]   79:              BewegeFigur(1);
[ZNØ]   8Ø:          end;
[3L2]   81:     6: Inc(Geld, 1ØØØ);
[FS1]   82:     7: begin
[XK5]   83:          for b:=Feld to 39 do BewegeFigur(1);
[W92]   84:            Inc (Geld, 4ØØØ);
[ZNØ]   85:          end;
[UE5]   86:     8: for b:=1 to 3 do BewegeFigur (-1);
[YY5]   87:     9: for b:=Feld to 38 do BewegeFigur(1);
[2F6]   88:   1Ø: if (GefangFreiE<>Ø) then Ereignisfeld
[V75]   89:       else GefangFreiE:=SpielerDran;
[HM3]   9Ø:    11: Bezahlen (3ØØ, True);
[ZØ3]   91:    12: if (Feld<24) then
[NG5]   92:          for b:=Feld to 23 do BewegeFigur(1)
[WP1]   93:        else begin
[W92]   94:          Inc (Geld, 4ØØØ);
[MK5]   95:          for b:=Feld to 63 do BewegeFigur(1);
[ZNØ]   96:        end;
[XT2]   97:    13: Inc (Geld, 2ØØØ);
[6W3]   98:    14: Bezahlen (3ØØØ, True);
[4U2]   99:    15: Inc (Geld, 3ØØØ);
[M63]  1ØØ:    16: if (Feld < 11) then
[EG5]  1Ø1:          for b:=Feld to 1Ø do BewegeFigur(1)
[WP1]  1Ø2:        else begin
[W92]  1Ø3:          Inc (Geld, 4ØØØ);
[DK5]  1Ø4:        for b:=Feld to 5Ø do BewegeFigur(1);
[ZNØ]  1Ø5:        end;
[ZNØ]  1Ø6:   end;
[V26]  1Ø7:   if (o <> Feld) then VerarbeiteStrasse;
[ZNØ]  1Ø8:  end;
[ZNØ]  1Ø9: end;
[ØØØ]  11Ø:
[UY4]  111: procedure Gemeinschaftsfeld;
[UC4]  112: var z, b: Byte; d: DialogObj;
```

```
[184]  113:     a: AuswahlObj; p: Pointer;
[SC4]  114:     i: Integer; r, o: ShortInt;
[MUØ]  115: begin
[YH4]  116:   i:=ImageSize (1ØØ, 165, 4ØØ, 315);
[3Ø2]  117:   GetMem (p, i);
[ZE4]  118:   GetImage (1ØØ, 165, 4ØØ, 315, p^);
[YW2]  119:   Inc (GemKartenZ);
[L96]  12Ø:   if (GemKartenZ > 16) then GemKartenZ:=1;
[6N5]  121:   z:=GemKartenStack [GemKartenZ];
[YT5]  122:   GemKarten [z]^.ZeigeKarte (1ØØ, 165);
[5P3]  123:   d.Zeige (41Ø, 25Ø, 49Ø, 315);
[VZ4]  124:   d.SetOpt (#Ø#23'Karte', '', 'Ok\?');
[LR2]  125:   d.Eingabe;  d.Weg;
[4E5]  126:   PutImage (1ØØ, 165, p^, NormalPut);
[1D2]  127:   FreeMem (p, i);
[LH5]  128:   with Spieler [SpielerDran]^ do
[MUØ]  129:   begin
[Z91]  13Ø:     o:=Feld;
[3C1]  131:     Case z Of
[TK3]  132:       1: Bezahlen (2ØØØ, True);
[CH2]  133:       2: Inc (Geld, 1ØØ);
[2K2]  134:       3: Inc (Geld, 4ØØØ);
[WH2]  135:       4: Inc (Geld, 4ØØ);
[YH2]  136:       5: Inc (Geld, 2ØØØ);
[ØK2]  137:       6: Inc (Geld, 2ØØØ);
[KV3]  138:       7: for b:=Feld downto 2 do
[VL2]  139:            BewegeFigur(-1);
[H51]  14Ø:       8: begin
[XK5]  141:          for b:=Feld to 39 do BewegeFigur(1);
[W92]  142:           Inc (Geld, 4ØØØ);
[ZNØ]  143:          end;
[W34]  144:       9: if (GefangFreiG <> Ø) then
[ØØ3]  145:            Gemeinschaftsfeld
[X75]  146:          else GefangFreiG:=SpielerDran;
[EA1]  147:     1Ø: begin
[Ø42]  148:           a.HoleEingabe
[H64]  149:                (#Ø#24'Gemeinschaftskarte',
[DW3]  15Ø:                 'Was wollen Sie tun ?',
[SC6]  151:      'DM 2ØØ zahlen\Ereigniskarte nehmen');
[143]  152:           if (a.Result=1) then
[ZØ3]  153:                   Bezahlen(2ØØ, True)
[M43]  154:           else Ereignisfeld;
[ZNØ]  155:          end;
[HA1]  156:     11: begin
[BØ5]  157:           if (Feld<1Ø) then r:=1 else r:=-1;
[C85]  158:           if (r = 1) then for b:=Feld to 8 do
[CC2]  159:                 BewegeFigur(1)
[6M4]  16Ø:           else for b:=Feld downto 12 do
[VL2]  161:                 BewegeFigur(-1);
[VX1]  162:          Gefangen:=3;
[774]  163:          BewegeFigur (r); Pasch:=Ø;
[ZNØ]  164:          end;
[XN2]  165:     12: Inc (Geld, 5ØØ);
[VV3]  166:     13: Bezahlen (1ØØØ, True);
[KM3]  167:     14: Bezahlen (1ØØ, True);
[3P2]  168:     15: Inc (Geld, 2ØØ);
[3Ø5]  169:     16: for b:=1 to SpielerZahl do
[VZ3]  17Ø:           if (b = SpielerDran) then
[725]  171:            Inc(Geld,Pred(SpielerZahl)*2ØØ)
[ZRØ]  172:           else
[8G5]  173:            Spieler [b]^.Bezahlen(2ØØ,False);
[ZNØ]  174:     end;
[146]  175:     if (o<>Feld) then VerarbeiteStrasse;
[ZNØ]  176:   end;
[ZNØ]  177: end;
[2A4]  178: procedure Zusatzsteuer;
[2A6]  179: var d: AuswahlObj; b: Byte; l: LongInt;
[MUØ]  18Ø: begin
[F25]  181:   d.HoleEingabe (#Ø#25'Steuerfeld',
[TA6]  182:     'Sie müssen\Zusatzsteuer zahlen.',
[NG4]  183:          '1Ø% des Vermögens\DM 4ØØØ');
[LH5]  184:   with Spieler [SpielerDran]^ do
[N56]  185:    if (d.Result=2) then Bezahlen(4ØØØ,True)
[WP1]  186:    else begin
[U91]  187:     l:=Geld;
[AH2]  188:     for b:=1 to 4Ø do
[BP3]  189:       if (b In Strassen) then
[TH3]  19Ø:         with Strasse [b]^ do
[6L5]  191:          if Hypothek then Inc(l,Wert Div 2)
[ED8]  192:          else Inc(l,Wert+Bebauung*Succ(Posi
       tion Div 1Ø)*1ØØØ);
[XS3]  193:          Bezahlen(l Div 1Ø, True);
[ZNØ]  194:     end;
[ZNØ]  195: end;
[Y95]  196: procedure VersteigereStrasse;
[EK6]  197: const Meldung: Array[1..2] of String[8]
[6H3]  198:                = ('kauft', 'steigert');
[HG6]  199: var k: byte; p: longint; e: EingabeObj;
[C54]  2ØØ:     a: AuswahlObj; q: integer;
[MUØ]  2Ø1: begin
[K75]  2Ø2:   if alle then inc(SpielerDran, 6);
[ØlK]  2Ø3:   k:=WaehleSpieler('Die Straße\' + Strasse[
       n]^.Name + '\wird ver' + Meldung[t] + '.\We
       r bietet am meisten ?');
[Z55]  2Ø4:   if alle then dec(SpielerDran, 6);
[8RØ]  2Ø5:   p:=Ø;
[9D7]  2Ø6:   Spieler[Strasse[n]^.Spieler]^.StrWeg(n);
[G25]  2Ø7:   Strasse[n]^.SetzeSpieler(Ø);
[652]  2Ø8:   if (k <> Ø) then
[MUØ]  2Ø9:   begin
[NK5]  21Ø:     Val(e.Eingabe(#Ø#26'Versteigerung',
[RX4]  211:        'Geben Sie den ausge-\machten '+
[ØV4]  212:        'Preis ein:',Ø123456789',15Ø),p,q);
[G52]  213:     if (p <> Ø) then
[MUØ]  214:     begin
[EZ4]  215:       if Strasse[n]^.Hypothek then
[MUØ]  216:       begin
[ØL5]  217:         a.HoleEingabe(#Ø#27'Versteigerung',
[XV5]  218:         'Auf der Straße liegt eine Hypo' +
[DW8]  219:  'thek,\auf die 1Ø% Zinsen bezahlt werden m
       üssen.','Ok');
[X84]  22Ø:         Inc(p, Strasse[n]^.Wert);
[ZNØ]  221:       end;
[155]  222:       Strasse[n]^.SetzeSpieler(k);
[TC4]  223:       Spieler[k]^.StrDazu(n);
[1G5]  224:       Spieler[k].Bezahlen(p, False);
[ZNØ]  225:     end;
[ZNØ]  226:   end;
[632]  227:   if (p = Ø) then
[4L5]  228:    a.HoleEingabe(#Ø#28'Versteigerung',
[XH4]  229:    'Damit geht die Straße\' +
[Y96]  23Ø:     Strasse[n]^.Name +'\an die Bank','Ok');
[ZNØ]  231: end;
[ØØØ]  232:
[154]  233: procedure KaufeStrasse;
[Z56]  234: var d: AuswahlObj; s: String; k: byte;
[PU5]  235:     p: longint; q: integer; e: DialogObj;
[MUØ]  236: begin
[LH5]  237:   with Spieler [SpielerDran]^ do
[MUØ]  238:     begin
[M94]  239:      with Strasse [Feld]^ do
[Y91]  24Ø:       s:=Name+
[956]  241:      '\zum Preis von DM '+StrNum (Wert,Ø);
[WL3]  242:      Delete (s, Pos ('~', s), 1);
[WØ6]  243:      d.HoleEingabe(#Ø#29'Straßenverkauf',
[E45]  244:        'Wollen Sie die Straße\' + s +
[934]  245:        '\erwerben ?', 'Ok\Ab\?');
[743]  246:      if (d.Result=1) then
[MUØ]  247:      begin
[XH6]  248:       Bezahlen(Strasse [Feld]^.Wert,False);
[CG2]  249:       StrDazu(Feld);
[3N7]  25Ø:       Strasse [Feld]^.SetzeSpieler(SpielerD
       ran);
[4MØ]  251:      end
[DD6]  252:      else VersteigereStrasse(Feld,2,true);
```

```
[ZNØ]  253:     end;
[ZNØ]  254: end;
[ØØØ]  255:
[W45]  256: procedure VerarbeiteStrasse;
[MR5]  257: var d: AuswahlObj; w: Word; k: Char;
[MUØ]  258: begin
[LH5]  259:   with Spieler [SpielerDran]^ do
[MUØ]  26Ø:     begin
[VC4]  261:      if (StrWert[Feld]<>Ø) then
[LA6]  262:       if (Strasse [Feld]^.Spieler<>Ø) then
[EL4]  263:        if Not (Feld In Strassen) And
[A36]  264:           Not Strasse [Feld]^.Hypothek then
[MUØ]  265:        begin
[UY4]  266:         w:=Strasse [Feld]^.HoleMiete;
[1TØ]  267:         with
[5L6]  268:         Spieler[Strasse[Feld]^.Spieler]^ do
[MUØ]  269:         begin
[ZM4]  27Ø:           d.HoleEingabe(#Ø#3Ø'Miete','DM '+
[CM4]  271:           StrNum(w, Ø)+' an '+Name,'Ok\?');
[LØ2]  272:           Inc (Geld, w);
[ZNØ]  273:         end;
[863]  274:         Bezahlen (w, False);
[4MØ]  275:        end
[ML1]  276:        else  else
[492]  277:           KaufeStrasse
[ZRØ]  278:       else
[PX1]  279:        Case Feld Of
[C84]  28Ø:     2, 17, 33: Gemeinschaftsfeld;
[YD3]  281:     7, 22, 36: Ereignisfeld;
[594]  282:     38       : Bezahlen (2ØØØ, True);
[9D3]  283:     4        : Zusatzsteuer;
[DP4]  284:     2Ø       : if (FreiParken <> Ø) then
[MUØ]  285:                begin
[825]  286:         d.HoleEingabe (#Ø#31'Frei parken',
[ØM5]  287:         'Glückspilz !\' +'Sie erhalten DM '+
[GP4]  288:              StrNum (FreiParken, 5), 'Ok');
[KP3]  289:              Inc (Geld, FreiParken);
[ZB2]  29Ø:              FreiParken:=Ø;
[ZNØ]  291:              end;
[ØX2]  292:  end; end; end; end.
```

Listing 3. Die Unit »monofeld.pas« legt verschiedene Regeln fest.

```
[3U3]    1: (Unit:     monodial.pas
[KS5]    2:  Funktion: Oberflächengestaltung
[G64]    3:  Sprache:  Turbo Pascal 6.Ø
[ZC3]    4:  Autor:    Andreas Knab
[915]    5:  (c) 1993 DMV Widuch GmbH & Co. KG)
[434]    6: {$A+,B-,D+,I-,L+,N-,O-,R-,S-}
[NE2]    7: unit MonoDial;
[SK1]    8: Interface
[YL3]    9: uses Crt,Graph,Mouse;
[STØ]   1Ø: type
[692]   11:   BoxObj=Object
[1Y2]   12:     p           : Pointer;
[ZL3]   13:     x1, y1, x2, y2: Integer;
[X23]   14:     Groesse     : Word;
[ØE5]   15:     Constructor Zeige (px1, py1, px2,
[694]   16:       py2: Integer; hf, rf: Byte);
[GN2]   17:     Destructor Weg;
[ZNØ]   18:   end;
[ZV2]   19:  BoxPtr=^BoxObj;
[LT2]   2Ø:  KnopfObj=Object
[ZL3]   21:   x1, y1, x2, y2: Integer;
[EZ3]   22:   Txt           : String[3Ø];
[5N2]   23:   Info          : Byte;
[ØE5]   24:   Constructor Zeige (px1, py1, px2,
[FU6]   25:   py2: Integer; pTxt: String; pInfo: Byte);
[RZ3]   26:   procedure SetCurMitte;
[ID6]   27:   procedure ZeigeGedrueckt(t: Boolean);
[DY4]   28:   function Gedrueckt: Boolean;
[TV6]   29:   function InBereich(x,y:Integer):Boolean;
[965]   3Ø:   procedure Bewege(x, y: Integer);
[GN2]   31:   destructor Weg;
[ZNØ]   32:  end;
[Ø44]   33: DialogObj=Object (BoxObj)
[AA2]   34:   Result: Byte;
[L95]   35:   KOpt  : Array[1..1Ø] Of KnopfObj;
[AA2]   36:   MaxOpt: Byte;
[YU3]   37:   Titel, Hilfe : KnopfObj;
[7A3]   38:   Ttl   : String[8Ø];
[NY2]   39:   Txt, Opt: String;
[XC7]   4Ø: Constructor Zeige(px1,py1,px2,py2:Integer);
[Z87]   41:   procedure SetOpt(pTtl,pTxt,pOpt: String);
[A23]   42:   procedure Eingabe;
[TB6]   43:   procedure Bewege(MitTasten: Boolean);
[MN3]   44:   procedure ZeigeHilfe;
[GN2]   45:   Destructor Weg;
[ZNØ]   46:  end;
[8R3]   47:  DialogPtr=^DialogObj;
[ØØØ]   48:
[UT4]   49:  AuswahlObj=Object (DialogObj)
[LH5]   5Ø:   procedure HoleEingabe(pTtl,pTxt,
[PC2]   51:                        pOpt: String);
[ZNØ]   52:  end;
[CP4]   53:  EingabeObj=Object (DialogObj)
[2T4]   54:    function Eingabe(pTtl,pTxt,
[CT6]   55:    Erlaubt: String; minb: Integer): String;
[ZNØ]   56:  end;
[FD5]   57: const KeinInfo  =Ø; InfoStatus=1;
[274]   58:       InfoOk    =2; InfoNMoegl=3;
[SF4]   59:       InfoTitel =8; InfoHilfe =9;
[Ø34]   6Ø: var CurX, CurY : Integer;
[SC3]   61:     MausKnopf  : Integer;
[S33]   62:     LastReadKey: Char;
[G46]   63: procedure HoleTaste(taste: boolean);
[ØØØ]   64:
[4L2]   65: Implementation
[EG2]   66: uses MonoInit;
[ØØØ]   67:
[ØG4]   68: Constructor BoxObj.Zeige;
[MUØ]   69: begin
[DP2]   7Ø:   x1:=px1; y1:=py1;
[TP2]   71:   x2:=px2; y2:=py2;
[ZK3]   72:   inc(px2, 8);  inc(py2, 8);
[PS5]   73:   Groesse:=ImageSize(px1,py1,px2,py2);
[P13]   74:   GetMem(p,Groesse);
[T35]   75:   GetImage(px1,py1,px2,py2,p^);
[924]   76:   SetFillStyle (SolidFill, hf);
[1B2]   77:   SetColor (rf);
[Ø13]   78:   Bar (x1, y1, x2, y2);
[8Ø6]   79:   Rectangle (Succ(x1), Succ(y1), Pred(x2),
[XH1]   8Ø:                Pred(y2));
[MD5]   81:   SetFillStyle(SolidFill, Black);
[UU3]   82:   Bar(x2+1, y1+8, px2, py2);
[DG3]   83:   Bar(x1+8, y2+1, x2, py2);
[ZNØ]   84: end;
[KV3]   85: Destructor BoxObj.Weg;
[MUØ]   86: begin
[AC5]   87:   PutImage (x1, y1, p^, NormalPut);
[ZF3]   88:   FreeMem (p, Groesse);
[ZNØ]   89: end;
[ØØØ]   9Ø:
[RØ5]   91: Constructor KnopfObj.Zeige;
```

```
[674]   92: const Symbol: String[2*3]
[7Ø6]   93:    ='Ok'+Chr(LightGreen)+'Ab'+Chr(LightRed);
[5C4]   94: var mx: Integer;  p: Byte;
[MUØ]   95: begin
[W32]   96:   Info:=pInfo;
[6H6]   97:   SetTextStyle (SmallFont, HorizDir, 5);
[Y76]   98:   SetFillStyle(SolidFill, LightGray);
[317]   99:   SetTextJustify(CenterText, CenterText);
[VS2]  1ØØ:   x1:=px1;  y1:=py1;
[5T2]  1Ø1:   x2:=px2;  y2:=py2;
[8S2]  1Ø2:   mx:=(x1+x2) div 2;
[5X1]  1Ø3:   Txt:=pTxt;
[ØJ3]  1Ø4:   Bar (x1, y1, x2, y2);
[193]  1Ø5:   p:=Pos(Txt, Symbol);
[9W1]  1Ø6:   if (p=Ø) then
[MUØ]  1Ø7:   begin
[M43]  1Ø8:     if (Txt[1]='?') then
[9X3]  1Ø9:       if (Info=InfoHilfe) then
[BL3]  11Ø:           SetColor(LightGreen)
[L74]  111:       else SetColor(LightBlue)
[LM3]  112:     else SetColor(White);
[615]  113:     OutTextXY (succ(mx), y1+8, Txt);
[4MØ]  114:   end
[ZRØ]  115:   else
[MUØ]  116:   begin
[PP4]  117:     SetColor(Ord(Symbol[p+2]));
[1B6]  118:     SetLineStyle(SolidLn, Ø, ThickWidth);
[5B1]  119:     Case p of
[351]  12Ø:       1: begin
[S14]  121:            Line(mx-5, y1+9, mx, y1+14);
[X24]  122:            Line(mx, y1+14, mx+6, y1+4);
[ZNØ]  123:          end;
[951]  124:       4: begin
[P54]  125:            Line(mx-5, y1+5, mx+5, y1+15);
[944]  126:            Line(mx-5, y1+15, mx+5, y1+5);
[ZNØ]  127:          end;
[ZNØ]  128:     end;
[B46]  129:     SetLineStyle(SolidLn, Ø, NormWidth);
[ZNØ]  13Ø:   end;
[UF4]  131:   if (Info <> InfoNMoegl) then
[MUØ]  132:     begin
[F75]  133:       SetColor (Black);  MoveTo (x2, y1);
[WY4]  134:       LineTo (x2, y2); LineTo (x1, y2);
[HH5]  135:       SetColor (White); LineTo (x1, y1);
[7B2]  136:       LineTo (x2, y1);
[4MØ]  137:     end
[ZRØ]  138:   else
[MUØ]  139:     begin
[CK2]  14Ø:       SetColor (Red);
[193]  141:       Line (x1, y1, x2, y2);
[193]  142:       Line (x1, y2, x2, y1);
[ZNØ]  143:     end;
[ZNØ]  144: end;
[ØØØ]  145:
[4H5]  146: procedure KnopfObj.SetCurMitte;
[MUØ]  147: begin
[C25]  148:   SetPos((x1+x2) div 2, (y1+y2) div 2);
[ZNØ]  149: end;
[ØØØ]  15Ø:
[CW5]  151: procedure KnopfObj.ZeigeGedrueckt;
[MUØ]  152: begin
[XY1]  153:   if not t then
[MUØ]  154:   begin
[HE5]  155:     SetColor (Black); MoveTo (x2, y1);
[WY4]  156:     LineTo (x2, y2); LineTo (x1, y2);
[HH5]  157:     SetColor (White); LineTo (x1, y1);
[7B2]  158:     LineTo (x2, y1);
[4MØ]  159:   end
[ZRØ]  16Ø:   else
[MUØ]  161:   begin
[XG3]  162:     SetColor(LightGray);
[214]  163:     Rectangle(x1, y1, x2, y2);
[ZNØ]  164:   end;
[ZNØ]  165: end;
[ØØØ]  166:
[HY4]  167: function KnopfObj.InBereich;
[MUØ]  168: begin
[MØ5]  169:   InBereich:=(x >= x1) and (x <= x2)
[ST3]  17Ø:    and (y >= y1) and (y <= y2);
[ZNØ]  171: end;
[ØØØ]  172:
[LL5]  173: function KnopfObj.Gedrueckt;
[MUØ]  174: begin
[CW2]  175:   Gedrueckt:=false;
[FK4]  176:   if not InBereich(XPosition,
[MK3]  177:                    YPosition) then Exit;
[WE5]  178:   if (MausKnopf=5) then Gedrueckt:=
[N83]  179:                      (Info<>InfoNMoegl);
[NA3]  18Ø:   if not LeftButton or
[WP4]  181:           (Info=InfoNMoegl) then Exit;
[GD3]  182:   while LeftButton do
[MUØ]  183:   begin
[M66]  184:     ZeigeGedrueckt(InBereich(XPosition,
[812]  185:                    YPosition));
[WW2]  186:     HoleTaste(false);
[ZNØ]  187:   end;
[HS3]  188:   ZeigeGedrueckt(false);
[7B7]  189:  Gedrueckt:=InBereich(XPosition,YPosition);
[ZNØ]  19Ø: end;
[BF4]  191: procedure KnopfObj.Bewege;
[MUØ]  192: begin
[RE3]  193:   Inc(x1, x); Inc(x2, x);
[ZE3]  194:   Inc(y1, y); Inc(y2, y);
[ZNØ]  195: end;
[P74]  196: Destructor KnopfObj.Weg;
[MUØ]  197: begin
[ZNØ]  198: end;
[N25]  199: Constructor DialogObj.Zeige;
[MUØ]  2ØØ: begin
[F15]  2Ø1:   BoxObj.Zeige (px1, py1, px2, py2,
[WU2]  2Ø2:                 DarkGray, White);
[YP1]  2Ø3:   MaxOpt:=Ø;
[ZNØ]  2Ø4: end;
[LW4]  2Ø5: procedure DialogObj.SetOpt;
[6Ø5]  2Ø6: var q, n: Byte;  x, y, b: Integer;
[BC3]  2Ø7:     s: String; f: Boolean;
[VV1]  2Ø8:     Info: Byte;
[MUØ]  2Ø9: begin
[CM3]  21Ø:   Ttl:=pTtl; Opt:=pOpt;
[5X1]  211:   Txt:=pTxt;
[663]  212:   if (Ttl[1] <> #Ø) then
[P85]  213:     Titel.Zeige(x1+3, y1+3, x2-3, y1+23,
[XN2]  214:                 Ttl, InfoTitel)
[ZRØ]  215:   else
[MUØ]  216:   begin
[X65]  217:     Titel.Zeige(x1+3, y1+3, x2-23, y1+23,
[HG4]  218:                 Copy(Ttl, 3, 255), InfoTitel);
[125]  219:     Hilfe.Zeige(x2-21, y1+3, x2-3, y1+23,
[CC2]  22Ø:                 '?', InfoHilfe);
[ZNØ]  221:   end;
[7H6]  222:   SetTextStyle (SmallFont, HorizDir, 6);
[L57]  223:   SetTextJustify (CenterText, CenterText);
[CT2]  224:   SetColor(White);
[VØ4]  225:   y:=y1+35; x:=(x1+x2) Div 2;
[MZ2]  226:   while (pTxt <> '') Do
[MUØ]  227:     begin
[4Z2]  228:       q:=Pos ('\', pTxt);
[H25]  229:       if (q=Ø) then q:=Length (pTxt)+1;
[MZ5]  23Ø:       OutTextXY(x,y,Copy(pTxt,1,Pred(q)));
[NZ2]  231:       Delete (pTxt, 1, q);
[LN1]  232:       Inc (y, 15);
[ZNØ]  233:     end;
```

Listing 4. »Monodial.pas« ist für die Dialogarbeit zuständig.

```
[ZK3]  234:    y:=y2-25; x:=x2-1Ø; n:=1;
[6H6]  235:    SetTextStyle (SmallFont, HorizDir, 5);
[KY2]  236:    while (pOpt <> '') Do
[MUØ]  237:    begin
[8Y2]  238:      q:=Pos ('\', pOpt);
[N15]  239:      if (q=Ø) then q:=Length (pOpt)+1;
[A44]  24Ø:      s:=Copy (pOpt, 1, Pred (q));
[6Y2]  241:      Delete (pOpt, 1, q);
[TW5]  242:      if (s='?')  then Info:=InfoStatus else
[B75]  243:      if (s='Ok') then Info:=InfoOk else
[NG5]  244:      if (s [1] <> '#') then Info:=KeinInfo
[WP1]  245:      else begin
[172]  246:        Delete (s, 1, 1);
[9Y2]  247:        Info:=InfoNMoegl;
[ZNØ]  248:      end;
[CU2]  249:      b:=TextWidth (s);
[136]  25Ø:      KOpt[n].Zeige(x-b-5,y,x,y+2Ø,s,Info);
[7X1]  251:      Dec (x, b+1Ø);
[161]  252:      Inc (n);
[ZNØ]  253:    end;
[AU2]  254:    MaxOpt:=Pred (n);
[ZNØ]  255: end;
[ØØØ]  256:
[KV4]  257: procedure DialogObj.Eingabe;
[8Y1]  258: var b: Byte;
[MUØ]  259: begin
[HV3]  26Ø:   if (MaxOpt=Ø) then Exit;
[VN4]  261:   KOpt[MaxOpt].SetCurMitte;
[P31]  262:   Repeat
[P31]  263:     Repeat
[TR2]  264:       HoleTaste(true);
[6V3]  265:       if (MausKnopf >= 59) then
[MUØ]  266:       begin
[ØL5]  267:         MausKnopf:=MaxOpt-(MausKnopf-59);
[6L3]  268:         if (MausKnopf > Ø) then
[7A5]  269:                KOpt[MausKnopf].SetCurMitte;
[472]  27Ø:         MausKnopf:=Ø;
[ZNØ]  271:       end;
[CS3]  272:     until (MausKnopf <> Ø);
[YP1]  273:     Result:=Ø;
[DC3]  274:     if (MausKnopf=2) then
[MUØ]  275:     begin
[7G4]  276:       if (MaxOpt=1) then Result:=1
[T84]  277:       else  for b:=1 To MaxOpt do
[MØ5]  278:        if (KOpt[b].Info=InfoOk) then
[2S1]  279:                          Result:=b;
[4MØ]  28Ø:     end
[ZRØ]  281:     else
[8V5]  282:       if Titel.InBereich(CurX, CurY) then
[293]  283:         Bewege(MausKnopf=5)
[UL3]  284:       else  if (Ttl[1]=#Ø) and
[NG5]  285:         Hilfe.Gedrueckt then ZeigeHilfe
[DW1]  286:        else  begin
[YP1]  287:         Result:=Ø;
[DPØ]  288:         b:=1;
[KN3]  289:         while (b <= MaxOpt) and
[FD5]  29Ø:               not KOpt[b].Gedrueckt do inc(b);
[473]  291:         if (b <= MaxOpt) then
[SV5]  292:         if (KOpt[b].Info=InfoStatus) then
[MUØ]  293:         begin
[ZX2]  294:           ZeigeSpielstand;
[VN4]  295:           KOpt[MaxOpt].SetCurMitte;
[4MØ]  296:         end
[ØR2]  297:         else Result:=b;
[ZNØ]  298:       end;
[ZØ3]  299:   until (Result <> Ø);
[ZNØ]  3ØØ: end;
[MT4]  3Ø1: procedure DialogObj.Bewege;
[X35]  3Ø2: var px1, py1, px2, py2: Integer;
[S54]  3Ø3:     mx, my: Integer; b: Byte;
[HW1]  3Ø4:     q: Pointer;
[MUØ]  3Ø5: begin
[ØW3]  3Ø6:   SetWriteMode(XorPut);
[ZD3]  3Ø7:   SetColor(LightRed);
[D44]  3Ø8:   px1:=x1; py1:=y1; px2:=x2;
[3R4]  3Ø9:   py2:=y2; mx:=CurX; my:=CurY;
[TU4]  31Ø:   SetXRange(mx-px1, 631-(px2-mx));
[ZU4]  311:   SetYRange(my-py1, 471-(py2-my));
[P31]  312:   repeat
[HL4]  313:     Rectangle(px1, py1, px2, py2);
[WW2]  314:     HoleTaste(false);
[HL4]  315:     Rectangle(px1, py1, px2, py2);
[UU5]  316:     Inc(px1, CurX-mx); Inc(py1, CurY-my);
[1V5]  317:     Inc(px2, CurX-mx); Inc(py2, CurY-my);
[PB3]  318:     mx:=CurX; my:=CurY;
[X24]  319:   until not (LeftButton or
[MZ4]  32Ø:             (MitTasten and (MausKnopf=Ø)));
[P94]  321:   SetWriteMode(NormalPut);
[UE3]  322:   for b:=1 to MaxOpt do
[PW4]  323:       KOpt[b].Bewege(px1-x1, py1-y1);
[NS4]  324:   Titel.Bewege(px1-x1, py1-y1);
[HØ3]  325:   if (Ttl[1]=#Ø) then
[GP4]  326:         Hilfe.Bewege(px1-x1, py1-y1);
[Z33]  327:   GetMem(q, Groesse);
[WF4]  328:   GetImage(x1, y1, x2, y2, q^);
[GU4]  329:   Weg;  Zeige(px1, py1, px2, py2);
[TG5]  33Ø:   PutImage(px1, py1, q^, NormalPut);
[U36]  331:   FreeMem(q, Groesse);  SetXRange(Ø, 639);
[8M4]  332:   SetYRange(Ø, 479);  MaxOpt:=b;
[ZNØ]  333: end;
[ØØØ]  334:
[YF5]  335: procedure DialogObj.ZeigeHilfe;
[Y44]  336: var h: DialogObj;  f: File;
[813]  337:     s: String; w: Word;
[MUØ]  338: begin
[DK3]  339:   h.Zeige(13Ø, 13Ø, 51Ø, 35Ø);
[F55]  34Ø:   h.SetOpt('Dialoghilfe', '', 'Ok');
[VX6]  341:   SetTextStyle(TriplexFont, HorizDir, 1);
[6U6]  342:   SetTextJustify(LeftText, BottomText);
[N84]  343:   Assign(f, 'MONOPOLY.HLP');
[US1]  344:   Reset(f, 1);
[LM3]  345:   if (IOResult <> Ø) then
[K68]  346:     OutTextXY(14Ø, 25Ø, 'MONOPOLY.HLP nicht
       gefunden.')
[DW1]  347:   else  begin
[Ø46]  348:     Seek(f, (Ord(Ttl[2])-1)*SizeOf(Word));
[FB4]  349:     Blockread(f, w, SizeOf(w));
[6V2]  35Ø:     Seek(f, w); w:=17Ø;
[LE3]  351:     BlockRead(f, s[Ø], 1);
[ST4]  352:     BlockRead(f, s[1], Ord(s[Ø]));
[1B6]  353:     SetLineStyle(SolidLn, Ø, ThickWidth);
[UR2]  354:     while (s <> '@') do
[MUØ]  355:     begin
[MT2]  356:       if (s[1]='_') then
[MUØ]  357:       begin
[G42]  358:         Delete(s, 1, 1);
[UB5]  359:         Line(16Ø,w+3,16Ø+TextWidth(s),w+3);
[ZNØ]  36Ø:       end;
[7X4]  361:       OutTextXY(16Ø, w, s); Inc(w, 18);
[LE3]  362:       BlockRead(f, s[Ø], 1);
[ST4]  363:       BlockRead(f, s[1], Ord(s[Ø]));
[ZNØ]  364:     end;
[KD1]  365:     close(f);
[B46]  366:     SetLineStyle(SolidLn, Ø, NormWidth);
[ZNØ]  367:   end;
[3L2]  368:   h.Eingabe; h.Weg;
[ZNØ]  369: end;
[ØØØ]  37Ø:
[594]  371: Destructor DialogObj.Weg;
[8Y1]  372: var b: Byte;
[MUØ]  373: begin
[3T5]  374:   for b:=1 to MaxOpt do KOpt[b].Weg;
[YR3]  375:   Titel.Weg; BoxObj.Weg;
[ZNØ]  376: end;
[ØØØ]  377:
[XN5]  378: procedure AuswahlObj.HoleEingabe;
[7U4]  379: var h, b: Integer; mb: Integer;
```

```
[373]  38Ø:     q: Byte;  s: String;
[MUØ]  381: begin
[3H3]  382:   h:=7Ø;  mb:=Ø;  s:=pTxt;
[7H6]  383:   SetTextStyle (SmallFont, HorizDir, 6);
[VE2]  384:   while (s <> '') do
[MUØ]  385:     begin
[N82]  386:       q:=Pos ('\', s);
[YG4]  387:       if (q=Ø) then q:=Length (s)+1;
[DH5]  388:       b:=TextWidth(Copy(s,1,Pred(q)))+3Ø;
[1B2]  389:       Delete (s, 1, q);
[3D3]  39Ø:       if (b > mb) then mb:=b;
[ØM1]  391:       Inc (h, 15);
[ZNØ]  392:     end;
[FM3]  393:   b:=TextWidth(pTtl)+2Ø;
[3D3]  394:   if (b > mb) then mb:=b;
[6H6]  395:   SetTextStyle (SmallFont, HorizDir, 5);
[YC2]  396:   s:=pOpt;  b:=2Ø;
[VE2]  397:   while (s <> '') do
[MUØ]  398:     begin
[N82]  399:       q:=Pos ('\', s);
[YG4]  4ØØ:       if (q=Ø) then q:=Length (s)+1;
[KS5]  4Ø1:       b:=b+TextWidth(Copy(s,1,Pred(q)))+15;
[1B2]  4Ø2:       Delete (s, 1, q);
[ZNØ]  4Ø3:     end;
[3D3]  4Ø4:   if (b > mb) then mb:=b;
[8R3]  4Ø5:   mb:=mb Div 2;  h:=h Div 2;
[S15]  4Ø6:   if (mb > 24Ø) then b:=32Ø Else b:=24Ø;
[PE4]  4Ø7:   Zeige (b-mb, 24Ø-h, b+mb, 24Ø+h);
[WB4]  4Ø8:   SetOpt (pTtl, pTxt, pOpt);
[V72]  4Ø9:   Eingabe;  Weg;
[ZNØ]  41Ø: end;
[ØØØ]  411:
[7V4]  412: function EingabeObj.Eingabe;
[TØ5]  413: var h, b: Integer;  mb: Integer;
[Z75]  414:     q: Byte;  s: String;  x: Integer;
[GD1]  415:     w : Byte;
[MUØ]  416: begin
[3H3]  417:   h:=7Ø;  mb:=Ø;  s:=pTxt;
[7H6]  418:   SetTextStyle (SmallFont, HorizDir, 6);
[VE2]  419:   while (s <> '') do
[MUØ]  42Ø:     begin
[N82]  421:       q:=Pos ('\', s);
[YG4]  422:       if (q=Ø) then q:=Length (s)+1;
[DH5]  423:       b:=TextWidth(Copy(s,1,Pred(q)))+3Ø;
[1B2]  424:       Delete (s, 1, q);
[3D3]  425:       if (b > mb) then mb:=b;
[ØM1]  426:       Inc (h, 15);
[ZNØ]  427:     end;
[FM3]  428:   b:=TextWidth(pTtl)+2Ø;
[3D3]  429:   if (b > mb) then mb:=b;
[P94]  43Ø:   if (minb > mb) then mb:=minb;
[614]  431:   mb:=mb Div 2;  h:=h Div 2+2Ø;
[S15]  432:   if (mb > 24Ø) then b:=32Ø else b:=24Ø;
[PE4]  433:   Zeige (b-mb, 24Ø-h, b+mb, 24Ø+h);
[875]  434:   SetOpt (pTtl, pTxt, '?\Ab\Ok');
[CK2]  435:   SetColor (Red);
[385]  436:   Rectangle(x1+2Ø, y2-55, x2-2Ø, y2-35);
[VN4]  437:   KOpt[MaxOpt].SetCurMitte;
[EV@]  438:   s:='';
[P31]  439:   repeat
[P31]  44Ø:     repeat
[FE3]  441:       x:=x1+21+TextWidth(s);
[ZD3]  442:       SetColor(LightRed);
[KX3]  443:       SetWriteMode (XORPut);
[7K3]  444:       Line (x, y2-54, x, y2-36);
[TR2]  445:       HoleTaste(true);
[7K3]  446:       Line (x, y2-54, x, y2-36);
[E34]  447:       SetWriteMode (CopyPut);
[6V3]  448:       if (MausKnopf >= 59) then
[MUØ]  449:       begin
[ØL5]  45Ø:         MausKnopf:=MaxOpt-(MausKnopf-59);
[6L3]  451:         if (MausKnopf > Ø) then
[7A5]  452:              KOpt[MausKnopf].SetCurMitte;
[472]  453:         MausKnopf:=Ø;
[ZNØ]  454:       end;
[664]  455:       if (LastReadKey <> #Ø) then
[MUØ]  456:       begin
[AY5]  457:         SetFillStyle(SolidFill, DarkGray);
[3B6]  458:         SetTextStyle(SmallFont,HorizDir,6);
[L57]  459:   SetTextJustify (CenterText, CenterText);
[3S4]  46Ø:         w:=TextWidth (LastReadKey);
[WU3]  461:         if (LastReadKey=#8) then
[R72]  462:           if (s <> '') then
[MUØ]  463:           begin
[Y66]  464:     Dec(x,TextWidth(Copy(s,Length(s),1)));
[8V3]  465:               Bar (x, y2-54, x2-21, y2-36);
[KH3]  466:               Delete (s, Length (s), 1);
[3D1]  467:           end else
[WV4]  468:           else if (LastReadKey > #31) and
[U15]  469:             (Pos (LastReadKey, Erlaubt) <> Ø)
[GB4]  47Ø:               and (x+w <= x2-21) then  begin
[BV2]  471:             s:=s+LastReadKey;
[DH6]  472:     OutTextXY(x+w Div 2,y2-45,LastReadKey);
[ZNØ]  473:         end;
[ZNØ]  474:       end;
[CS3]  475:     until (MausKnopf <> Ø);
[YP1]  476:     Result:=Ø;
[DC3]  477:     if (MausKnopf=2) then
[MUØ]  478:     begin
[7G4]  479:       if (MaxOpt=1) then Result:=1
[ZRØ]  48Ø:       else
[UE3]  481:         for b:=1 To MaxOpt do
[MØ5]  482:           if (KOpt[b].Info=InfoOk) then
[2S1]  483:             Result:=b;
[4MØ]  484:     end
[ZRØ]  485:     else
[8V5]  486:     if Titel.InBereich(CurX, CurY) then
[124]  487:           Bewege(MausKnopf=5) else
[A76]  488:     if (Ttl[1]=#Ø) and Hilfe.Gedrueckt then
[VS1]  489:       ZeigeHilfe
[DW1]  49Ø:     else  begin
[AE2]  491:       Result:=Ø; b:=1;
[KN3]  492:       while (b <= MaxOpt) and
[FD5]  493:        not KOpt[b].Gedrueckt do inc(b);
[473]  494:       if (b <= MaxOpt) then
[SV5]  495:       if (KOpt[b].Info=InfoStatus) then
[MUØ]  496:       begin
[ZX2]  497:         ZeigeSpielstand;
[VN4]  498:         KOpt[MaxOpt].SetCurMitte;
[4MØ]  499:       end
[ØR2]  5ØØ:       else Result:=b;
[ZNØ]  5Ø1:     end;
[ZØ3]  5Ø2:   until (Result <> Ø);
[MPØ]  5Ø3:   Weg;
[HL4]  5Ø4:   if (Result=2) then Eingabe:=''
[GP2]  5Ø5:                 else Eingabe:=s;
[ZNØ]  5Ø6: end;
[ØØØ]  5Ø7:
[LK3]  5Ø8: procedure HoleTaste;
[GH4]  5Ø9: var c: char; x, y, k: Integer;
[MUØ]  51Ø: begin
[2Z4]  511:   MousePos(x, y, k); ShowCursor;
[P31]  512:   repeat
[1M5]  513:     MousePos (CurX, CurY, MausKnopf);
[RM2]  514:     LastReadKey:=#Ø;
[Y23]  515:     if keypressed then
[MUØ]  516:     begin
[3X1]  517:       c:=readkey;
[Y91]  518:       case c of
[ZS2]  519:         #13: MausKnopf:=5;
[CT2]  52Ø:         #27: MausKnopf:=2;
[K91]  521:         #Ø: begin
[3X1]  522:        c:=readkey;
[Y91]  523:        case c of
[U26]  524:        #75: SetPos(XPosition-15,YPosition);
[W26]  525:        #77: SetPos(XPosition+15,YPosition);
```

```
[636]  526:        #72: SetPos(XPosition,YPosition-15);
[Z26]  527:        #8Ø: SetPos(XPosition,YPosition+15);
[AT2]  528:        #71: SetPos(4Ø, 4Ø);
[PY2]  529:        #79: SetPos(4Ø, 44Ø);
[YX2]  53Ø:        #73: SetPos(44Ø, 4Ø);
[X23]  531:        #81: SetPos(44Ø, 44Ø);
[X74]  532:        #59..#68: MausKnopf:=Ord(c);
[ZNØ]  533:       end;
[ZNØ]  534:       end;
[ND3]  535:         else LastReadKey:=c;
[ZNØ]  536:     end;
[ZNØ]  537:   end;
[S86]  538:   until (MausKnopf<>k) or (((CurX<>x) or
[S74]  539:    (CurY<>y)) and not taste) or
[NØ3]  54Ø:    (LastReadKey <> #Ø);
[Ø32]  541:   HideCursor;
[2Ø6]  542:   CurX:=XPosition; CurY:=YPosition;
[ZNØ]  543: end;
[KNØ]  544: end.
```

Listing 4. »Monodial.pas« ist für die Dialogarbeit zuständig.

```
[2X3]    1: (Unit:     monocnst.pas
[ØH7]    2:  Funktion: Festlegung von Konstantenfeldern
[G64]    3:  Sprache:  Turbo Pascal 6.Ø
[G63]    4:  Autor:    Andeas Knab
[7D4]    5:  (c) DMV Widuch GmbH & Co. KG)
[ØØØ]    6:
[VA5]    7: ($A+,B-,D-,E+,F-,I-,L-,N-,O-,R-,S-,V-)
[8G2]    8: unit MonoCnst;
[SK1]    9: interface
[5H1]   1Ø: uses Crt;
[6XØ]   11: const
[WT5]   12: GruppenFarbe: Array[Ø..8] Of Byte =
[AP5]   13:     (White, Blue, Cyan, Magenta, Brown,
[7Z5]   14:      LightRed, Yellow, Green, LightBlue);
[836]   15: PreisTxt : Array[1..7] Of String[23] =
[NE3]   16:     ('Grundstückswert',
[TH4]   17:      'Miete Grundstück allein',
[XU2]   18:      'Miete mit 1 Haus',
[4N3]   19:      'Miete mit 2 Häusern',
[5N3]   2Ø:      'Miete mit 3 Häusern',
[6N3]   21:      'Miete mit 4 Häusern',
[K33]   22:      'Miete mit 1 Hotel');
[D16]   23: StrName : Array[1..4Ø] Of String[2Ø] =
[CG7]   24:     ('Bad-straße',      '',                'T
        urm-straße',
[DS7]   25:      '',                'Süd-bahnhof',     'Cha
        ussee-straße',
[BD7]   26:      '',                'Elisen-straße',   'Pos
        t-straße',
[6H8]   27:      '',                'See-straße',      'Ele
        ktrizi-tätswerk',
[XX7]   28:      'Hafen-straße',    'Neue -Straße',    'Wes
        t-bahnhof',
[4T7]   29:      'Münchner -Straße', '',               'Wie
        ner -Straße',
[UG7]   3Ø:      'Berliner -Straße', '',               'The
        ater-straße',
[657]   31:      '',                'Museum-straße',   'Ope
        rn-platz',
[VE9]   32:      'Nord-bahnhof',    'Lessing-straße',  'Sch
        iller-straße',
[U76]   33:      'Wasser-werk',     'Goethe-straße',   '',
[Y16]   34:      'Rathaus-platz',   'Haupt-straße',    '',
[1F6]   35:      'Bahnhof-straße',  'Haupt-bahnhof',   '',
[5S7]   36:      'Park-straße',     '',                'Schl
        oß-allee', '');
[4T5]   37: StrGruppe : Array [1..4Ø] Of Byte =
[7E3]   38:     (1, Ø,1Ø, Ø, Ø, 2, Ø, 2,11, Ø, 3,
[XR3]   39:      Ø, 3,12, Ø, 4, Ø, 4,13, Ø, 5, Ø, 5,
[YG3]   4Ø:      14, Ø, 6, 6, Ø,15, Ø, 7, 7, Ø,16,
[EX1]   41:      Ø, Ø, 8, Ø,17, Ø);
[FC5]   42: StrWert : Array [1..4Ø] Of Byte =
[M84]   43:     (12,Ø,12,Ø,4Ø,2Ø,Ø,2Ø,24,Ø,28,3Ø,
[F94]   44:      28,32,4Ø,36,Ø,36,4Ø,Ø,44,Ø,44,48,
[W74]   45:      4Ø,52,52,3Ø,56,Ø,6Ø,6Ø,Ø,64,4Ø,Ø,
[TG1]   46:      7Ø,Ø,8Ø,Ø);
[ZV3]   47:       ( Aufbau der Mietskala :
[HX5]   48: - Gruppe 1 /Straße 1/ Miete 1,2,3,4,5,6
[CF6]   49: - Gruppe 1 /Straße 2 o. 3/Miete 1,2,3,4,5,6
[M96]   5Ø: - Gruppe 2 /Straße 1/Miete 1,2,3,4,5,6 ..)
[NE5]   51: StrMiete : Array [1..96] Of Word =
[WR4]   52:  (ØØØ4Ø, ØØ2ØØ, ØØ6ØØ, Ø18ØØ, Ø32ØØ, Ø5ØØØ,
[3L4]   53:   ØØØ8Ø, ØØ1ØØ, Ø12ØØ, Ø36ØØ, Ø64ØØ, Ø9ØØØ,
[ØL4]   54:   ØØ12Ø, Ø3ØØØ, Ø18ØØ, Ø54ØØ, Ø8ØØØ, 11ØØØ,
[ØL4]   55:   ØØ16Ø, ØØ8ØØ, Ø2ØØØ, Ø6ØØØ, Ø9ØØØ, 12ØØØ,
[WK4]   56:   ØØ2ØØ, Ø1ØØØ, Ø3ØØØ, Ø9ØØØ, 125ØØ, 15ØØØ,
[9K4]   57:   ØØ24Ø, Ø12ØØ, Ø36ØØ, 1ØØØØ, 14ØØØ, 18ØØØ,
[HK4]   58:   ØØ28Ø, Ø14ØØ, Ø4ØØØ, 11ØØØ, 15ØØØ, 19ØØØ,
[LK4]   59:   ØØ32Ø, Ø16ØØ, Ø44ØØ, 12ØØØ, 16ØØØ, 2ØØØØ,
[RL4]   6Ø:   ØØ36Ø, Ø18ØØ, Ø5ØØØ, 14ØØØ, 175ØØ, 21ØØØ,
[9L4]   61:   ØØ4ØØ, Ø2ØØØ, Ø6ØØØ, 15ØØØ, 185ØØ, 22ØØØ,
[8M4]   62:   ØØ44Ø, Ø22ØØ, Ø66ØØ, 16ØØØ, 195ØØ, 23ØØØ,
[ML4]   63:   ØØ448, Ø24ØØ, Ø72ØØ, 17ØØØ, 2Ø5ØØ, 24ØØØ,
[FM4]   64:   ØØ52Ø, Ø26ØØ, Ø78ØØ, 18ØØØ, 22ØØØ, 255ØØ,
[NK4]   65:   ØØ56Ø, Ø3ØØØ, Ø9ØØØ, 2ØØØØ, 24ØØØ, 28ØØØ,
[ØL4]   66:   ØØ7ØØ, Ø35ØØ, 1ØØØØ, 22ØØØ, 26ØØØ, 3ØØØØ,
[MN4]   67:   Ø1ØØØ, Ø4ØØØ, 12ØØØ, 28ØØØ, 34ØØØ,4ØØØØ);
[XF6]   68: EreignisTxt : Array [1..16] Of String =
[VN9]   69:  ('Du wirst zu Straßenausbesserungs-\arbeit
        en herange'+
[MP8]   7Ø:  'zogen. Zahle für\Deine Häuser und Hotels\
        \DM  8ØØ '+
[W16]   71:  ' je Haus \DM 23ØØ je Hotel\an die Bank.',
[SR8]   72:  '\Gehe in das Gefängnis.\Begib Dich direkt
        dorthin.'+
[D28]   73:  '\Gehe nicht über Los.\\Ziehe nicht DM 4ØØ
        Ø ein.',
[PN6]   74:  '\\Betrunken im Dienst.\\Strafe DM 4ØØ',
[PV8]   75:  'Lasse alle Deine Häuser\renovieren.\\Zahl
        e an die '+
[GE8]   76:  'Bank\für jedes Haus  DM  5ØØ\für jedes Ho
        tel DM 2Ø' +
[XGØ]   77:  'ØØ',
[UW8]   78:  'Mache einen Ausflug nach dem\West-Bahnhof
        ,\\und we' +
[SW6]   79:  'nn Du über Los kommst,\ziehe DM 4ØØØ ein.
        ',
[KØ8]   8Ø:  '\Die Bank zahlt Dir\eine Dividende von\\D
        M 1ØØØ',
[ZM4]   81:  '\\\Rücke bis auf Los vor.',
[XK4]   82:  '\\\Gehe 3 Felder zurück.',
[PL6]   83:  '\\Rücke vor bis zur\\Schloßallee.',
[749]   84:  'Du kommst aus dem Gefängnis frei.\\Diese
        Karte mu' +
[199]   85:  'ß behalten wer-\den, bis sie gebraucht od
        er\verkau' +
[6T1]   86:  'ft wird.',
[U67]   87:  '\\Strafe für zu schnelles Fahren.\\DM 3ØØ
        ',
[5F9]   88:  '\Rücke vor bis zum Opernplatz.\\Wenn Du ü
        ber Los k' +
[L34]   89:  'ommst,\ziehe DM 4ØØØ ein.',
```

DAS ABO



Ihr Vorsprung

- **Sichern Sie sich Ihre Vorteile:**
- **Hochkarätige Informationen**
- **Regelmäßig frei Haus lieferbar**
- **Über 7 Prozent Preisvorteil**

CeBIT'93 HANNOVER
24. — 31. 03. 1993
Halle 7, Stand E13/E19

Widerrufsrecht: Jeder Abonnent hat das Recht, seine Bestellung innerhalb von zehn Tagen beim DMV-Verlag, Postfach 250, 3440 Eschwege schriftlich zu widrerrufen. Die rechtzeitige Absendung des Widerrufschreibens genügt zur Fristwahrung.

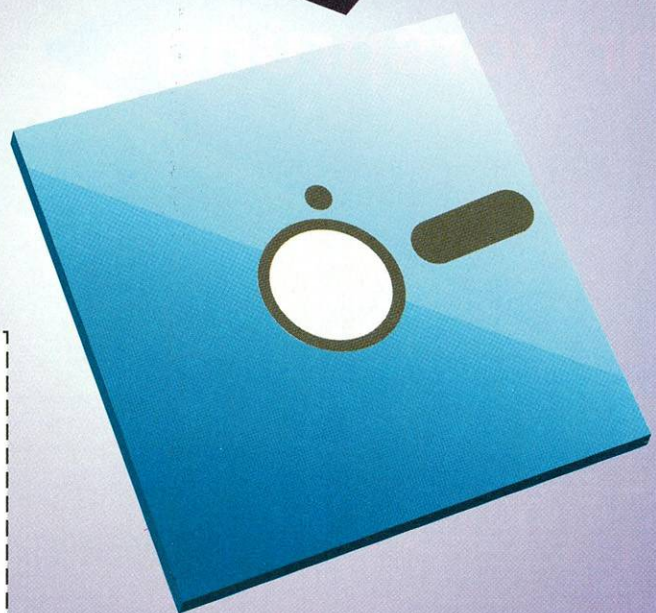
DMV Verlag • Postfach 250 • 3440 Eschwege

DMV Daten- und Medienverlag



Alle Programme dieser Ausgabe auf Diskette

nur 12 Mark



☐ Hiermit bestelle ich die Databox der DOS Extra zu einem Preis von 12 Mark zuzüglich 4 Mark Versandkosten (Ausland: 6 Mark)

Diskettenformat: 5 1/4 Zoll
3.5 Zoll

Wie beurteilen Sie den Listing-Anteil der DOS Extra?
☐ zu viel
☐ genau richtig
☐ zu wenig

Wieviele Themen sollen in der DOS Extra behandelt werden?
☐ 1 Thema
☐ 2 bis 3 Themen
☐ mehr als 3 Themen

Über welche Themen würden Sie in der DOS Extra gerne mehr lesen?

_____

_____

In welcher Form würden Sie die Listings der DOS Extra vorzugsweise beziehen?
☐ Databox
☐ Mailbox
☐ Btx

Mit welchen Compilern arbeiten Sie?

_____

_____

Welche zusätzliche Hardware besitzen Sie?

_____

_____

_____

Datum **Unterschrift**

Bitte ausschneiden und auf einer Postkarte abschicken an:

DMV-Verlag
DOS Extra
Postfach 250
W-3440 Eschwege

```
[D99]   9Ø:   '\Du hast in einem Kreuzworträtsel-\Wettbe
              werb gewo' +
[A54]   91:   'nnen.\\Ziehe DM 2ØØØ ein.',
[SR4]   92:   '\\Zahle Schulgeld\\DM 3ØØØ',
[PE9]   93:   '\Miete und Anleihezinsen\werden fällig.\\
              Die Bank ' +
[UR2]   94:   'zahlt Dir DM 3ØØØ',
[ØR9]   95:   '\Rücke vor bis zur Seestraße.\\Wenn Du üb
              er Los ko' +
[H74]   96:   'mmst,\ziehe DM 4ØØØ ein.');
[ØØØ]   97:
[MF6]   98: GemeinschTxt : Array [1..16] Of String =
[A86]   99: ('\\Zahle an das Krankenhaus\\DM 2ØØØ',
[F87]   1ØØ: '\\Aus Lagerverkäufen erhälst Du\\DM 1ØØ',
[3Y8]   1Ø1: '\\Bank-Irrtum zu Deinen Gunsten.\\Ziehe D
             M 4ØØØ ei' +
[FLØ]   1Ø2: 'n.',
[1T8]   1Ø3: '\\Einkommensteuer-Rückzahlung\\Ziehe DM 4
             ØØ ein.',
[XH3]   1Ø4: '\\Du erbst\\DM 2ØØØ',
[5G8]   1Ø5: '\\Die Jahresrente wird fällig.\\Ziehe DM
             2ØØØ ein.',
[9K6]   1Ø6: '\\Gehe zurück nach der\\Badstraße',
[ZX4]   1Ø7: '\\Rücke vor bis auf\\Los.',
[749]   1Ø8: '\Du kommst aus dem Gefängnis frei.\\Diese
             Karte mu' +
[199]   1Ø9: 'ß behalten wer-\den, bis sie gebraucht od
             er\verkau' +
[6T1]   11Ø: 'ft wird.',
[YZ8]   111: '\Zahle eine Strafe von\\DM 2ØØ\\oder nimm
             eine Ere' +
[Z92]   112: 'igniskarte.',
[3V8]   113: '\Gehe in das Gefängnis.\Begib Dich direkt
             dorthin.' +
[D28]   114: '\Gehe nicht über Los.\\Ziehe nicht DM 4ØØ
             Ø ein.',
[X19]   115: '\Du erhälst auf Vorzugs-Aktien\7% Dividen
             de\\DM 5Ø' +
[6DØ]   116: 'Ø',
[PE7]   117: '\\Zahle Deine Versicherungssumme.\\DM 1ØØ
             Ø',
[X45]   118: '\\Arzt-Kosten.\\Zahle DM 1ØØ',
[849]   119: '\Du hast den II. Preis in einer\Schönheit
             skonkurre' +
[1M5]   12Ø: 'nz gewonnen.\\Ziehe DM 2ØØ ein.',
[639]   121: '\\Es ist Dein Geburtstag.\\Ziehe von jede
             m Spieler' +
[P52]   122: '\DM 2ØØ ein.');
[4L2]   123: Implementation
[KNØ]   124: End.
```

Listing 5. »Monocnst.pas« legt die Konstantenfelder fest.

```
[A53]    1: (Unit:     mouse.pas
[HØ5]    2:  Funktion: Steuerung der Maus
[G64]    3:  Sprache:  Turbo Pascal 6.Ø
[ZC3]    4:  Autor:    Andreas Knab
[585]    5:  (c) 1992 DMV Widuch GmbH & Co. KG )
[ØØØ]    6:
[4H2]    7: {$D-,I-,L-,R-,S-}
[512]    8: unit Mouse;
[SK1]    9: INTERFACE
[GL3]   1Ø: procedure InitMouse;
[296]   11: function InitAndGetStatus: Integer;
[MT3]   12: procedure ShowCursor;
[XN3]   13: procedure HideCursor;
[CN6]   14: procedure MousePos (var x,y,b: Integer);
[545]   15: function XPosition: Integer;
[845]   16: function YPosition: Integer;
[E35]   17: function LeftButton: Boolean;
[U85]   18: function RightButton: Boolean;
[8H5]   19: function CenterButton: Boolean;
[X95]   2Ø: procedure SetPos(x,y: Integer);
[EH6]   21: procedure ButtonPressed(But:Integer;
[HB4]   22:                  var x,y,b,Count:Integer);
[KL6]   23: procedure ButtonReleased(But: Integer;
[BC4]   24:                  var x,y,b,Count: Integer);
[TN6]   25: procedure SetXRange (min, max: Integer);
[WN6]   26: procedure SetYRange (min, max: Integer);
[T37]   27: procedure SetGraphicCursor(x,y: Integer;
[4X3]   28:                  CursorDaten: Pointer);
[MF6]   29: procedure SetTextCursor(CursorMode,
[V85]   3Ø:                  ScreenMask, CursorMask: Byte);
[A56]   31: procedure MouseMotion (var XMickeys,
[5A3]   32:                  YMickeys: Integer);
[DR6]   33: procedure MouseProc (CallMask: Integer;
[973]   34:                  ProcPtr: pointer);
[ER3]   35: procedure LightPenOn;
[DX3]   36: procedure LightPenOff;
[W17]   37: procedure SetMickeys (hor, vert: Integer);
[ØØØ]   38:
[4L2]   39: IMPLEMENTATION
[ØØØ]   4Ø:
[SM2]   41: uses Dos, Graph;
[ØØØ]   42:
[RNØ]   43: var
[LU2]   44:   Regs: Registers;
[GL3]   45: procedure InitMouse;
[MUØ]   46: begin
[HW1]   47:   Regs.AX:=Ø;
[MW1]   48:   Regs.BX:=Ø;
[RW1]   49:   Regs.CX:=Ø;
[UW1]   5Ø:   Regs.DX:=Ø;
[HC2]   51:   Intr (51,Regs);
[ZNØ]   52: end;
[ØØØ]   53:
[296]   54: function InitAndGetStatus: Integer;
[MUØ]   55: begin
[4R2]   56:   if (GetMaxX = 719)
[X33]   57:     then Mem[$4Ø:$49]:=6;
[HW1]   58:   Regs.AX:=Ø;
[MW1]   59:   Regs.BX:=Ø;
[RW1]   6Ø:   Regs.CX:=Ø;
[UW1]   61:   Regs.DX:=Ø;
[HC2]   62:   Intr (51,Regs);
[GA2]   63:   if (Regs.AX = Ø)
[D74]   64:     then InitAndGetStatus:=Ø
[XL5]   65:     else InitAndGetStatus:=Regs.BX;
[ZNØ]   66: end;
[ØØØ]   67:
[MT3]   68: procedure ShowCursor;
[MUØ]   69: begin
[MW1]   7Ø:   Regs.AX:=1;
[MW1]   71:   Regs.BX:=Ø;
[RW1]   72:   Regs.CX:=Ø;
[UW1]   73:   Regs.DX:=Ø;
[272]   74:   Intr(51,Regs);
[ØL3]   75: end { ShowCursor };
[ØØØ]   76:
[XN3]   77: procedure HideCursor;
[MUØ]   78: begin
[RW1]   79:   Regs.AX:=2;
[MW1]   8Ø:   Regs.BX:=Ø;
[RW1]   81:   Regs.CX:=Ø;
[UW1]   82:   Regs.DX:=Ø;
[272]   83:   Intr(51,Regs);
```

```
[9F3]   84: end { HideCursor };
[ØØØ]   85:
[ØF6]   86: procedure MousePos(var x,y,b: Integer);
[MUØ]   87: begin
[UW1]   88:   Regs.AX:=3;
[MW1]   89:   Regs.BX:=Ø;
[RW1]   9Ø:   Regs.CX:=Ø;
[UW1]   91:   Regs.DX:=Ø;
[272]   92:   Intr(51,Regs);
[9Y1]   93:   b:=Regs.BX;
[RZ1]   94:   x:=Regs.CX;
[VZ1]   95:   y:=Regs.DX;
[EW2]   96: end { MousePos };
[ØØØ]   97:
[545]   98: function XPosition: Integer;
[MUØ]   99: begin
[UW1]  1ØØ:   Regs.AX:=3;
[MW1]  1Ø1:   Regs.BX:=Ø;
[RW1]  1Ø2:   Regs.CX:=Ø;
[UW1]  1Ø3:   Regs.DX:=Ø;
[272]  1Ø4:   Intr(51,Regs);
[LG3]  1Ø5:   XPosition:=Regs.CX;
[V93]  1Ø6: end { XPosition };
[ØØØ]  1Ø7:
[845]  1Ø8: function YPosition: Integer;
[MUØ]  1Ø9: begin
[UW1]  11Ø:   Regs.AX:=3;
[MW1]  111:   Regs.BX:=Ø;
[RW1]  112:   Regs.CX:=Ø;
[UW1]  113:   Regs.DX:=Ø;
[272]  114:   Intr(51,Regs);
[RG3]  115:   YPosition:=Regs.DX;
[Z93]  116: end { YPosition };
[ØØØ]  117:
[E35]  118: function LeftButton: Boolean;
[MUØ]  119: begin
[UW1]  12Ø:   Regs.AX:=3;
[MW1]  121:   Regs.BX:=Ø;
[RW1]  122:   Regs.CX:=Ø;
[UW1]  123:   Regs.DX:=Ø;
[272]  124:   Intr(51,Regs);
[CA4]  125:   LeftButton:=ODD(Regs.BX);
[5H3]  126: end { LeftButton };
[ØØØ]  127:
[U85]  128: function RightButton: Boolean;
[MUØ]  129: begin
[UW1]  13Ø:   Regs.AX:=3;
[MW1]  131:   Regs.BX:=Ø;
[RW1]  132:   Regs.CX:=Ø;
[UW1]  133:   Regs.DX:=Ø;
[272]  134:   Intr(51,Regs);
[LD5]  135:   RightButton:=((Regs.BX AND 2) <> Ø);
[LR3]  136: end { RightButton };
[ØØØ]  137:
[8H5]  138: function CenterButton: Boolean;
[MUØ]  139: begin
[UW1]  14Ø:   Regs.AX:=3;
[MW1]  141:   Regs.BX:=Ø;
[RW1]  142:   Regs.CX:=Ø;
[UW1]  143:   Regs.DX:=Ø;
[272]  144:   Intr(51,Regs);
[TP5]  145:   CenterButton:=((Regs.BX AND 4) <> Ø);
[BL3]  146: end { CenterButton };
[ØØØ]  147:
[X95]  148: procedure SetPos(x,y: Integer);
[MUØ]  149: begin
[XW1]  15Ø:   Regs.AX:=4;
[MW1]  151:   Regs.BX:=Ø;
[FØ2]  152:   Regs.CX:=x;
[NØ2]  153:   Regs.DX:=y;
[272]  154:   Intr(51,Regs);
[NU2]  155: end { SetPos };
[ØØØ]  156:
[EH6]  157: procedure ButtonPressed(But:Integer;
[HB4]  158:                  var x,y,b,Count:Integer);
[MUØ]  159: begin
[ØX1]  16Ø:   Regs.AX:=5;
[W72]  161:   Regs.BX:=But;
[RW1]  162:   Regs.CX:=Ø;
[UW1]  163:   Regs.DX:=Ø;
[272]  164:   Intr(51,Regs);
[RZ1]  165:   x:=Regs.CX;
[VZ1]  166:   y:=Regs.DX;
[7Y1]  167:   b:=Regs.AX;
[2S2]  168:   Count:=Regs.BX;
[KY3]  169: end { ButtonPressed };
[ØØØ]  17Ø:
[KL6]  171: procedure ButtonReleased(But: Integer;
[BC4]  172:                  var x,y,b,Count: Integer);
[MUØ]  173: begin
[3X1]  174:   Regs.AX:=6;
[W72]  175:   Regs.BX:=But;
[RW1]  176:   Regs.CX:=Ø;
[UW1]  177:   Regs.DX:=Ø;
[272]  178:   Intr(51,Regs);
[RZ1]  179:   x:=Regs.CX;
[VZ1]  18Ø:   y:=Regs.DX;
[7Y1]  181:   b:=Regs.AX;
[2S2]  182:   Count:=Regs.BX;
[374]  183: end { ButtonReleased };
[ØØØ]  184:
[TN6]  185: procedure SetXRange (min, max: Integer);
[MUØ]  186: begin
[6X1]  187:   Regs.AX:=7;
[MW1]  188:   Regs.BX:=Ø;
[A72]  189:   Regs.CX:=min;
[R62]  19Ø:   Regs.DX:=max;
[272]  191:   Intr(51,Regs);
[M83]  192: end { SetXRange };
[ØØØ]  193:
[WN6]  194: procedure SetYRange (min, max: Integer);
[MUØ]  195: begin
[9X1]  196:   Regs.AX:=8;
[MW1]  197:   Regs.BX:=Ø;
[A72]  198:   Regs.CX:=min;
[R62]  199:   Regs.DX:=max;
[272]  2ØØ:   Intr(51,Regs);
[R83]  2Ø1: end { SetYRange };
[ØØØ]  2Ø2:
[T37]  2Ø3: procedure SetGraphicCursor(x,y: Integer;
[4X3]  2Ø4:                  CursorDaten: pointer);
[MUØ]  2Ø5: begin
[CX1]  2Ø6:   Regs.AX:=9;
[CØ2]  2Ø7:   Regs.BX:=x;
[KØ2]  2Ø8:   Regs.CX:=y;
[RU4]  2Ø9:   Regs.DX:=Ofs(CursorDaten^);
[2U4]  21Ø:   Regs.ES:=Seg(CursorDaten^);
[272]  211:   Intr(51,Regs);
[RC4]  212: end { SetGraphicCursor };
[ØØØ]  213:
[MF6]  214: procedure SetTextCursor(CursorMode,
[V85]  215:                  ScreenMask, CursorMask: Byte);
[MUØ]  216: begin
[3X1]  217:   Regs.AX:=1Ø;
[8L3]  218:   Regs.BX:=CursorMode;
[KH3]  219:   Regs.CX:=ScreenMask;
[RM3]  22Ø:   Regs.DX:=CursorMask;
[272]  221:   Intr(51,Regs);
[S24]  222: end { SetTextCursor };
[ØØØ]  223:
[A56]  224: procedure MouseMotion (var XMickeys,
[5A3]  225:                  YMickeys: Integer);
[MUØ]  226: begin
[7X1]  227:   Regs.AX:=11;
[MW1]  228:   Regs.BX:=Ø;
[RW1]  229:   Regs.CX:=Ø;
[UW1]  23Ø:   Regs.DX:=Ø;
```

```
[272]  231:   Intr(51,Regs);
[N53]  232:   XMickeys:=Regs.CX;
[S53]  233:   YMickeys:=Regs.DX;
[TN3]  234: end { MouseMotion };
[ØØØ]  235:
[DR6]  236: procedure MouseProc (CallMask: Integer;
[973]  237:                  ProcPtr: pointer);
[MUØ]  238: begin
[BX1]  239:   Regs.AX:=12;
[MW1]  24Ø:   Regs.BX:=Ø;
[H23]  241:   Regs.CX:=CallMask;
[844]  242:   Regs.DX:=Ofs(ProcPtr^);
[L34]  243:   Regs.ES:=Seg(ProcPtr^);
[272]  244:   Intr(51,Regs);
[C73]  245: end { MouseProc };
[ØØØ]  246:
[ER3]  247: procedure LightPenOn;
[MUØ]  248: begin
[FX1]  249:   Regs.AX:=13;
[MW1]  25Ø:   Regs.BX:=Ø;
[RW1]  251:   Regs.CX:=Ø;
[UW1]  252:   Regs.DX:=Ø;
[272]  253:   Intr(51,Regs);
[UG3]  254: end { LightPenOn };
[ØØØ]  255:
[DX3]  256: procedure LightPenOff;
[MUØ]  257: begin
[LX1]  258:   Regs.AX:=14;
[MW1]  259:   Regs.BX:=Ø;
[RW1]  26Ø:   Regs.CX:=Ø;
[UW1]  261:   Regs.DX:=Ø;
[272]  262:   Intr(51,Regs);
[AM3]  263: end { LightPenOff };
[ØØØ]  264:
[W17]  265: procedure SetMickeys (hor, vert: Integer);
[MUØ]  266: begin
[RX1]  267:   Regs.AX:=15;
[MW1]  268:   Regs.BX:=Ø;
[R72]  269:   Regs.CX:=hor;
[ZE2]  27Ø:   Regs.DX:=vert;
[272]  271:   Intr(51,Regs);
[ØK3]  272: end { SetMickeys };
[2T3]  273: end { of unit Mouse ).
```

Listing 6. »Mouse.pas« ist für die einwandfreie Steuerung der Maus zuständig.

```
[MW3]    1: (Unit:     Monomaus.pas
[F56]    2:  Funktion: enthält Routinen für das
                       Hauptprogramm
[NC2]    3:
[G64]    4:  Sprache:  Turbo Pascal 6.Ø
[ZC3]    5:  Autor:    Andreas Knab
[585]    6:  (c) 1992 DMV Widuch GmbH & Co. KG )
[ØØØ]    7:
[355]    8: {$A+,B-,D-,E+,I-,L-,N-,O+,R-,S-,V-}
[KF2]    9: Unit Monomaus;
[SK1]   1Ø: Interface
[CØ2]   11: uses Graph;
[ØØØ]   12:
[V97]   13: procedure MausInterrupt(Flags,CS,IP,AX,BX,
[HE6]   14:   CX,DX,SI,DI,DS,ES,BP: Word); interrupt;
[ØØØ]   15:
[4L2]   16: Implementation
[ØØØ]   17:
[NB4]   18: var MausX, MausY: Integer;
[HW1]   19:     MinX, MinY,
[GG3]   2Ø:     MaxX, MaxY  : Integer;
[EF3]   21:     HinterGr    : Pointer;
[893]   22:     Sichtbar    : Boolean;
[N76]   23:     Cursor2     : array[1..8] of PointType;
[ØØØ]   24:
[EG6]   25: const Cursor : array[1..8] of PointType
[8S4]   26:  =((x:Ø; y:Ø),(x:Ø; y:9),(x:3; y:8),
[EU4]   27:    (x:8; y:13),(x:11; y:11),(x:6; y:6),
[Z83]   28:    (x:9; y:5),(x:Ø; y:Ø));
[ØØØ]   29:
[VD4]   3Ø: procedure MausInterrupt;
[PZ3]   31:   procedure ZeigeCursor;
[X22]   32:   var b : Byte;
[MUØ]   33:   begin
[EA5]   34:     GetImage(MausX, MausY, MausX + 15,
[VN3]   35:                  MausY + 15, HinterGr^);
[ZF2]   36:     for b:=1 to 8 do
[MUØ]   37:     begin
[SW5]   38:       Cursor2[b].x:=Cursor[b].x + MausX;
[XW5]   39:       Cursor2[b].y:=Cursor[b].y + MausY;
[ZNØ]   4Ø:     end;
[5R2]   41:     SetColor(Black);
[RF5]   42:     SetFillStyle(SolidFill, White);
[WE3]   43:     FillPoly(8, Cursor2);
[KE3]   44:     DrawPoly(8, Cursor2);
[BP2]   45:     Sichtbar:=true;
[ZNØ]   46:   end;
[ØØØ]   47:
[SS4]   48:   procedure VersteckeCursor;
[MUØ]   49:   begin
[RF7]   5Ø:   PutImage(MausX,MausY,HinterGr^,NormalPut);
[XR2]   51:     Sichtbar:=false;
[ZNØ]   52:   end;
[ØØØ]   53:
[8R2]   54: var os : Boolean;
[MUØ]   55: begin
[DP1]   56:  case AX of
[151]   57:  Ø: begin
[CY5]   58:       GetMem(HinterGr,ImageSize(Ø,Ø,15,15));
[XR2]   59:       Sichtbar:=False;
[B91]   6Ø:       MinX:=Ø;
[C91]   61:       MinY:=Ø;
[DF2]   62:       MaxX:=GetMaxX;
[GF2]   63:       MaxY:=GetMaxY;
[ZNØ]   64:     end;
[TY5]   65:  1: if Not Sichtbar then ZeigeCursor;
[226]   66:  2: if Sichtbar then VersteckeCursor;
[751]   67:  3: begin
[2YØ]   68:       BX:=Ø;
[SS1]   69:       CX:=MausX;
[WS1]   7Ø:       DX:=MausY;
[ZNØ]   71:     end;
[HF5]   72:  4: if (CX <= MaxX) and (CX >= MinX) and
[Ø95]   73:       (DX <= MaxY) and (DX >= MinY) then
[MUØ]   74:     begin
[B62]   75:       os:=Sichtbar;
[ZM5]   76:       if Sichtbar then VersteckeCursor;
[UR1]   77:       MausX:=CX;
[XR1]   78:       MausY:=DX;
[5W3]   79:       if os then ZeigeCursor;
[ZNØ]   8Ø:     end;
[F51]   81:  7: begin
[AG1]   82:       MinX:=CX;
```

Listing 7. »Monomaus.pas« enthält wichtige Routinen für das Hauptprogramm.

```
[V12]   83:        MaxX:=DX - 15;
[ZNØ]   84:      end;
[H51]   85:   8: begin
[BG1]   86:        MinY:=CX;
[W12]   87:        MaxY:=DX - 15;
[ZNØ]   88:      end;
[ZNØ]   89:  end;
[ZNØ]   9Ø: end;
[KNØ]   91: end.
```

Listing 7. »Monomaus.pas« enthält wichtige Routinen für das Hauptprogramm.

```
[Y94]    1: (Programm: monopoly.pas
[884]    2:  Funktion: Spielprogramm
[G64]    3:  Sprache:  Turbo Pascal 6.Ø
[ZC3]    4:  Autor:    Andreas Knab
[585]    5:  (c) 1992 DMV Widuch GmbH & Co. KG )
[ØØØ]    6:
[VA5]    7: ($A+,B-,D-,E+,F-,I-,L-,N-,O-,R-,S-,V-)
[6X2]    8: ($M 32768,Ø,655369)
[E33]    9: program Monopoly;
[X36]   1Ø: uses Crt, Dos, Graph, Mouse, MonoMaus,
[MR6]   11:      MonoCnst, MonoInit, MonoDial, MonoKart,
[TG4]   12:      MonoFeld, BGIDriv, BGIFont;
[ØØØ]   13:
[G64]   14: var Gewuerfelt : Boolean;
[HH5]   15:     Icons      : Array[1..9] of Pointer;
[ØØØ]   16:
[RP3]   17: procedure Wuerfeln;
[LN6]   18:   procedure Kreis (x, y: Word; b: Boolean);
[MUØ]   19:   begin
[965]   2Ø:     if b then Sector (x, y, Ø, 36Ø, 4, 4);
[ZNØ]   21:   end;
[ØØØ]   22:
[4U2]   23: var b, w1, w2: Byte;
[5P2]   24:     r        : ShortInt;
[XU1]   25:     k        : Char;
[HM2]   26:     d        : DialogObj;
[8Z2]   27:     a        : AuswahlObj;
[MUØ]   28: begin
[3A3]   29:   w1 := Succ (Random (6));
[6A3]   3Ø:   w2 := Succ (Random (6));
[ZN3]   31:   LetzterWurf := w1 + w2;
[624]   32:   if (w1 = w2) then Inc (Pasch)
[LB2]   33:    else Pasch := Ø;
[HN3]   34:   d.Zeige (235, 18Ø, 4Ø5, 3ØØ);
[6A5]   35:   d.SetOpt (#Ø#1'Würfeln', '', 'Ok\?');
[UT2]   36:   SetColor (Black);
[YG5]   37:   SetFillStyle (SolidFill, White);
[M13]   38:   Bar (27Ø, 22Ø, 315, 265);
[513]   39:   Bar (325, 22Ø, 37Ø, 265);
[G34]   4Ø:   Rectangle (27Ø, 22Ø, 315, 265);
[944]   41:   Rectangle (325, 22Ø, 37Ø, 265);
[RE5]   42:   SetFillStyle (SolidFill, Black);
[ØB5]   43:   Kreis (28Ø,23Ø,Not Odd (w1) or (w1=5));
[4H3]   44:   Kreis (3Ø5, 23Ø, (w1 > 2));
[GH3]   45:   Kreis (28Ø, 242, (w1 = 6));
[CS3]   46:   Kreis (292, 242, Odd (w1));
[GH3]   47:   Kreis (3Ø5, 242, (w1 = 6));
[YH3]   48:   Kreis (28Ø, 255, (w1 > 2));
[ØS5]   49:   Kreis(3Ø5, 255, Not Odd (w1) or (w1=5));
[SR5]   5Ø:   Kreis(335, 23Ø, Not Odd (w2) or (w2=5));
[2H3]   51:   Kreis (36Ø, 23Ø, (w2 > 2));
[RH3]   52:   Kreis (335, 242, (w2 = 6));
[LS3]   53:   Kreis (347, 242, Odd (w2));
[EH3]   54:   Kreis (36Ø, 242, (w2 = 6));
[5K3]   55:   Kreis (335, 255, (w2 > 2));
[4S5]   56:   Kreis(36Ø, 255, Not Odd (w2) or (w2=5));
[LL1]   57:   d.Eingabe;
[KZØ]   58:   d.Weg;
[LH5]   59:   With Spieler [SpielerDran]^ Do
[MUØ]   6Ø:   begin
[7F3]   61:     if (Gefangen <> Ø) then
[MUØ]   62:     begin
[KW2]   63:       if (Pasch <> Ø) then
[MUØ]   64:       begin
[KM4]   65:         a.HoleEingabe (#Ø#4'Würfeln',
[BP6]   66: 'Das war e. Pasch!\Sie sind frei!','Ok\?');
[W42]   67:         Gefangen := Ø;
[3P1]   68:         Pasch := Ø;
[4MØ]   69:       end
[DW1]   7Ø:       else  begin
[PM4]   71:         a.HoleEingabe (#Ø#5'Würfeln',
[PW5]   72: 'Das war leider\kein Pasch.', 'Ok\?');
[KE2]   73:         Dec (Gefangen);
[523]   74:         if (Gefangen=Ø) then
[SB3]   75:           Bezahlen (1ØØØ, True);
[122]   76:         ZeigeFigur;
[ZNØ]   77:       end;
[ZNØ]   78:     end;
[CA5]   79:     if (Gefangen = Ø) And (Pasch < 3) then
[MUØ]   8Ø:     begin
[4K4]   81:       For b:=1 To LetzterWurf Do
[MUØ]   82:       begin
[9L2]   83:         BewegeFigur (1);
[7V4]   84:         if (Feld=4Ø) then Inc(Geld, 4ØØØ);
[ZNØ]   85:       end;
[393]   86:       VerarbeiteStrasse;
[ZNØ]   87:     end;
[6K4]   88:     if (Feld=3Ø) or (Pasch=3) then
[MUØ]   89:     begin
[UM4]   9Ø:       a.HoleEingabe (#Ø#6'Würfeln',
[5S7]   91: 'Sie müssen leider\ins Gefängnis!','Ok\?');
[V25]   92:       if (Feld < 1Ø) then r:=1 else r:=-1;
[VS4]   93:       if (r=1) then for b:=Feld to 8 do
[EH2]   94:         BewegeFigur (1)
[6M4]   95:        else for b:=Feld downto 12 do
[1R2]   96:         BewegeFigur (-1);
[VX1]   97:       Gefangen:=3;
[CP2]   98:       BewegeFigur (r);
[AE1]   99:       Pasch:=Ø;
[ZNØ] 1ØØ:      end;
[ZNØ] 1Ø1:    end;
[ZNØ] 1Ø2: end;
[ØØØ] 1Ø3:
[KP2] 1Ø4: procedure Info;
[314] 1Ø5: var w: Byte; p: Pointer;
[MN2] 1Ø6:     c: Char; v: Word;
[XF4] 1Ø7:     d: DialogObj; a: AuswahlObj;
[MUØ] 1Ø8: begin
[1E2] 1Ø9:   w:=FeldNummer;
[712] 11Ø:   if (w=2Ø) then
[MUØ] 111:     begin
[TC4] 112:       a.HoleEingabe (#Ø#7'Hinweis',
[FL4] 113: 'Zur Zeit befinden sich\DM '+
[HS3] 114:  StrNum (Freiparken, Ø)+
[C66] 115: ' auf dem\Feld 'Frei parken¤.', 'Ok\?');
[ØWØ] 116:       Exit;
[ZNØ] 117:     end;
[Y56] 118:   if (w=Ø) or (StrWert [w]=Ø) then Exit;
[1M4] 119:   v:=ImageSize (17Ø, 115, 371, 366);
[GØ2] 12Ø:   GetMem (p, v);
[8G4] 121:   GetImage (17Ø, 115, 371, 366, p^);
[XU5] 122:   Strasse [w]^.ZeigeKarte (17Ø, 115);
[KR3] 123:   d.Zeige (39Ø, 3Ø1, 47Ø, 366);
[ØP4] 124:   d.SetOpt (#Ø#2'Info', '', 'Ok\?');
[LR2] 125:   d.Eingabe;  d.Weg;
[1E5] 126:   PutImage (17Ø, 115, p^, NormalPut);
[VD2] 127:   FreeMem (p, v);
[ZNØ] 128: end;
[ØØØ] 129:
[CH4] 13Ø: procedure Gefaengniskarte;
[Z13] 131: var a: AuswahlObj;
[EL1] 132:     o: String;
[H71] 133:     b: Byte;
[C72] 134:     e: EingabeObj;
[BW1] 135:     p: LongInt;
[SV1] 136:     q: Integer;
[WD1] 137:     k: ^Byte;
[MUØ] 138: begin
[ZGA] 139:   o:='Ab\Ereign. kaufen\Gemeinsch. kaufen\B
             enutzen\Bezahlen\?';
[GU6] 14Ø:   if (Spieler[SpielerDran]^.Gefangen=Ø) or
[KD5] 141:      Gewuerfelt then Insert('#', o, 46);
[UV5] 142:   if ((GefangFreiE <> SpielerDran) and
[H95] 143:      (GefangFreiG <> SpielerDran)) or
[VS6] 144:     (Spieler[SpielerDran]^.Gefangen=Ø) then
[9N2] 145:      Insert('#', o, 37);
[KK5] 146:   if (GefangFreiG=SpielerDran) then
[Ø23] 147:     Insert('ver', o, 3Ø)
[DH4] 148:   else if (GefangFreiG=Ø) then
[BN2] 149:     Insert('#', o, 19);
[AK5] 15Ø:   if (GefangFreiE=SpielerDran) then
[223] 151:     Insert('ver', o, 12)
[BH4] 152:   else if (GefangFreiE=Ø) then
[SE2] 153:     Insert('#', o, 4);
[EE6] 154:   a.HoleEingabe(#Ø#8'Gefängniskarte','',o);
[MM2] 155:   Case a.Result of
[B71] 156:     1: Exit;
[R13] 157:     2: k:=@GefangFreiE;
[Z13] 158:     3: k:=@GefangFreiG;
[951] 159:     4: begin
[965] 16Ø:          with Spieler[SpielerDran]^ do
[MUØ] 161:          begin
[FX1] 162:            Gefangen:=Ø;
[122] 163:            ZeigeFigur;
[ZNØ] 164:          end;
[KK5] 165:          if (GefangFreiG=SpielerDran) then
[PA2] 166:            GefangFreiG:=Ø
[Y93] 167:          else GefangFreiE:=Ø;
[ØWØ] 168:          Exit;
[ZNØ] 169:        end;
[FH5] 17Ø:     5: with Spieler[SpielerDran]^ do
[MUØ] 171:        begin
[FX1] 172:          Gefangen:=Ø;
[122] 173:          ZeigeFigur;
[9A3] 174:          Bezahlen(1ØØØ, False);
[ØWØ] 175:          Exit;
[ZNØ] 176:        end;
[ZNØ] 177:   end;
[KY3] 178:   if (k^=SpielerDran) then
[MUØ] 179:   begin
[US2] 18Ø:     b:=WaehleSpieler
[FV6] 181: ('Wer will die Gefängnisfreikarte von\' +
[AU6] 182: Spieler[SpielerDran]^.Name + ' kaufen ?');
[LX2] 183:     if (b=Ø) then Exit;
[777] 184: Val(a.Eingabe(#Ø#9'Gefängniskartenverkauf',
[9X6] 185: 'Geben Sie den ausge-\machten Preis ein :',
[823] 186: '1234567890', 15Ø), p, q);
[G52] 187:     if (p <> Ø) then
[MUØ] 188:     begin
[FF5] 189:       Spieler[b]^.Bezahlen(p, False);
[946] 19Ø:       Inc(Spieler[SpielerDran]^.Geld, p);
[AØ1] 191:       k^:=b;
[ZNØ] 192:     end;
[4MØ] 193:   end
[WP1] 194:   else begin
[SK6] 195:  a.HoleEingabe(#Ø#1Ø'Gefängniskartenkauf',
[AN3] 196:  Spieler[k^].Name + ',
[CF5] 197:  verkaufen Sie\Ihre Gefängnis' +
[WD2] 198:  'freikarte an ' +
[697] 199:  Spieler[SpielerDran]^.Name+' ?','Ok\Ab\?');
[2MØ] 2ØØ:     if (a.Result=2) then Exit;
[NVD] 2Ø1: Val(a.Eingabe(#Ø#11'Gefängniskartenverkauf'
           ,'Geben Sie den ausge-\machten Preis ein:',
[823] 2Ø2:       '1234567890', 15Ø), p, q);
[G52] 2Ø3:     if (p <> Ø) then
[MUØ] 2Ø4:     begin
[9T5] 2Ø5:       Spieler[SpielerDran]^.Bezahlen(p,
[361] 2Ø6:                                      False);
[BD4] 2Ø7:       Inc(Spieler[k^].Geld, p);
[1X2] 2Ø8:       k^:=SpielerDran;
[ZNØ] 2Ø9:     end;
[ZNØ] 21Ø:   end;
[ZNØ] 211: end;
[ØØØ] 212:
[ØK5] 213: function HoleDateiname: String;
[1Y2] 214: var d: EingabeObj;
[7Ø3] 215:     s: string;  b: byte;
[MUØ] 216: begin
[333] 217:   s[Ø]:=chr(255 - 32 + 1);
[255] 218:   for b:=32 to 255 do s[b]:=chr(b);
[YØ6] 219:   s:=d.Eingabe(#Ø#12'Datei laden/sichern',
[RØ7] 22Ø: 'Geben Sie d. gewünschten Dateinamen ein:',
[PXØ] 221:                                    s, 35Ø);
[RV1] 222:   if (s<>'') and
[H46] 223:  (Pos('.',Copy(s,Pos('\',s)+1,255))=Ø) then
[FY1] 224:     s:=s + '.MON';
[MW2] 225:   HoleDateiname:=s;
[ZNØ] 226: end;
[ØØØ] 227:
[1M2] 228: const Kennung =
[PW7] 229:   #8'Monopoly 1.1-Spieledatei'#13#1Ø'Andreas
             Knab'#26;
[385] 23Ø: procedure LadeDatei(s: String);
[WW1] 231: var f: file;
[H71] 232:     b: byte;
[8B2] 233:     a: AuswahlObj;
[MUØ] 234: begin
[S12] 235:   inoutres:=Ø;
[HØ2] 236:   assign(f, s);
[US1] 237:   reset(f, 1);
[GL3] 238:   if (inoutres <> Ø) then
[MUØ] 239:   begin
[ØB6] 24Ø:     while (pos('\',s)<>Ø) do s[pos('\',s)]:
             ='/';
[HY4] 241:     a.HoleEingabe(#Ø#13'Datei laden',
[AA6] 242:   '['+StrNum(inoutres,2)+ '] Die Datei\'+s+
[636] 243:     '\kann nicht geöffnet werden !', 'Ok');
[ØWØ] 244:     Exit;
[ZNØ] 245:   end;
[HZ1] 246:   s:=Kennung;
[YM5] 247:   BlockRead(f, s[Ø], Succ(Length(s)));
[PV4] 248:   if (s <> Kennung) or eof(f) then
[MUØ] 249:   begin
[KD1] 25Ø:     Close(f);
[NY4] 251:     a.HoleEingabe(#Ø#14'Datei laden',
[3G5] 252:     'Datei hat falsches format !', 'Ok');
[ØWØ] 253:     Exit;
[ZNØ] 254:   end;
[RF4] 255:   for b:=1 to SpielerZahl do
[MM3] 256:     Spieler[b]^.Fertig;
[FA7] 257: BlockRead(f,FreiParken,SizeOf(FreiParken));
[X64] 258:   BlockRead(f, SpielerZahl,
[CK3] 259:                    SizeOf(SpielerZahl));
[A64] 26Ø:   BlockRead(f, SpielerDran,
[XF3] 261:                    SizeOf(SpielerDran));
[RP5] 262:   BlockRead(f, Pasch, SizeOf(Pasch));
[1A4] 263:   BlockRead(f, LetzterWurf,
[PN3] 264:                    SizeOf(LetzterWurf));
[X34] 265:   BlockRead(f, GefangFreiE,
[4E3] 266:                    SizeOf(GefangFreiE));
[Z34] 267:   BlockRead(f, GefangFreiG,
[AE3] 268:                    SizeOf(GefangFreiG));
[X34] 269:   BlockRead(f, Gewuerfelt,
[5C3] 27Ø:                    SizeOf(Gewuerfelt));
[145] 271:   BlockRead(f, EreigKartenStack,
[DG4] 272:                  SizeOf(EreigKartenStack));
[PD4] 273:   BlockRead(f, EreigKartenZ,
[YU3] 274:                   SizeOf(EreigKartenZ));
[ET4] 275:   BlockRead(f, GemKartenStack,
[Y14] 276:                   SizeOf(GemKartenStack));
[Z34] 277:   BlockRead(f, GemKartenZ,
[LB3] 278:                   SizeOf(GemKartenZ));
[AH2] 279:   for b:=1 to 4Ø do
[YM3] 28Ø:     with Strasse[b]^ do
[MUØ] 281:   begin
[RS6] 282:   BlockRead(f, Hypothek, SizeOf(Hypothek));
[8L6] 283:   BlockRead(f, Bebauung, SizeOf(Bebauung));
[LK6] 284:   BlockRead(f, Spieler, SizeOf(Spieler));
[ZNØ] 285:   end;
[RF4] 286:   for b:=1 to SpielerZahl do
[DK3] 287:    with Spieler[b]^ do
[MUØ] 288:    begin
[UØ2] 289:      Init(b, Ø, '');
[TM5] 29Ø:      BlockRead(f, Farbe, SizeOf(Farbe));
[186] 291:      BlockRead(f, Nummer, SizeOf(Nummer));
[M45] 292:      BlockRead(f, Name, SizeOf(Name));
[3X6] 293:      BlockRead(f,Strassen,SizeOf(Strassen));
[F35] 294:      BlockRead(f, Geld, SizeOf(Geld));
[835] 295:      BlockRead(f, Feld, SizeOf(Feld));
[XK6] 296:      BlockRead(f,Gefangen,SizeOf(Gefangen));
[122] 297:      ZeigeFigur;
[ZNØ] 298:    end;
[AH2] 299:   for b:=1 to 4Ø do
[GV4] 3ØØ:     Strasse[b]^.ZeigeBebauung;
[KD1] 3Ø1:   close(f);
[ZNØ] 3Ø2: end;
[ØØØ] 3Ø3:
[ØØ6] 3Ø4: procedure SchreibeDatei(s: String);
[RB3] 3Ø5: var f: file;  b: byte;
[8B2] 3Ø6:     a: auswahlobj;
[MUØ] 3Ø7: begin
[T84] 3Ø8:   inoutres:=Ø;  assign(f, s);
[F52] 3Ø9:   rewrite(f, 1);
[GL3] 31Ø:   if (inoutres <> Ø) then
[MUØ] 311:   begin
[NS3] 312:     while (pos('\', s) <> Ø) do
[1U2] 313:       s[pos('\', s)]:='/';
[BS5] 314:     a.HoleEingabe(#Ø#15'Spiel speichern',
[HC6] 315: '['+StrNum(inoutres,2)+'] Die Datei\'+s+
[636] 316:   '\kann nicht erstellt werden !', 'Ok');
[GWØ] 317:     Exit;
[ZNØ] 318:   end;
[HZ1] 319:   s:=Kennung;
[PY5] 32Ø:   BlockWrite(f, s[Ø], Succ(Length(s)));
[R54] 321:   BlockWrite(f, FreiParken,
[AA3] 322:                 SizeOf(FreiParken));
[MD4] 323:   BlockWrite(f, SpielerZahl,
[CK3] 324:                 SizeOf(SpielerZahl));
[LD4] 325:   BlockWrite(f, SpielerDran,
[XF3] 326:                 SizeOf(SpielerDran));
[EX5] 327:   BlockWrite(f, Pasch, SizeOf(Pasch));
[7K4] 328:   BlockWrite(f, LetzterWurf,
[PN3] 329:                 SizeOf(LetzterWurf));
[4A4] 33Ø:   BlockWrite(f, GefangFreiE,
[4E3] 331:                 SizeOf(GefangFreiE));
[8A4] 332:   BlockWrite(f, GefangFreiG,
[AE3] 333:                 SizeOf(GefangFreiG));
[L64] 334:   BlockWrite(f, Gewuerfelt,
[5C3] 335:                 SizeOf(Gewuerfelt));
[4D5] 336:   BlockWrite(f, EreigKartenStack,
[DG4] 337:       SizeOf(EreigKartenStack));
[1P4] 338:   BlockWrite(f, EreigKartenZ,
[YU3] 339:        SizeOf(EreigKartenZ));
[PV4] 34Ø:   BlockWrite(f, GemKartenStack,
[Y14] 341:        SizeOf(GemKartenStack));
[V54] 342:   BlockWrite(f, GemKartenZ,
[LB3] 343:        SizeOf(GemKartenZ));
[AH2] 344:   for b:=1 to 4Ø do
[YM3] 345:    with Strasse[b]^ do
[MUØ] 346:    begin
[KY6] 347:   BlockWrite(f,Hypothek,SizeOf(Hypothek));
[AT6] 348:   BlockWrite(f,Bebauung,SizeOf(Bebauung));
[YL6] 349:     BlockWrite(f,Spieler,SizeOf(Spieler));
[ZNØ] 35Ø:   end;
[RF4] 351:   for b:=1 to SpielerZahl do
[DK3] 352:    with Spieler[b]^ do
[MUØ] 353:    begin
[PU5] 354:     BlockWrite(f, Farbe, SizeOf(Farbe));
[6D6] 355:     BlockWrite(f, Nummer, SizeOf(Nummer));
[FG5] 356:     BlockWrite(f, Name, SizeOf(Name));
[PØ7] 357:    BlockWrite(f,Strassen,SizeOf(Strassen));
[CF5] 358:     BlockWrite(f, Geld, SizeOf(Geld));
[7F5] 359:     BlockWrite(f, Feld, SizeOf(Feld));
[ØP6] 36Ø:   BlockWrite(f,Gefangen,SizeOf(Gefangen));
[ZNØ] 361:   end;
[KD1] 362:   close(f);
[ZNØ] 363: end;
[ØØØ] 364:
[ZF3] 365: procedure SpielOpt;
[UR4] 366: var a: AuswahlObj; s: String;
[MUØ] 367: begin
[FL4] 368:   a.HoleEingabe(#Ø#16'Optionen',
[R44] 369:        '', 'Ab\Laden\Sichern\?');
[K44] 37Ø:   if (a.Result=1) then Exit;
[LW2] 371:   s:=holedateiname;
[ØK4] 372:   while (s <> '') and (s[1]=' ') do
[G42] 373:     delete(s, 1, 1);
[2S5] 374:   while (s<>'') and (s[length(s)]=' ') do
[RH3] 375:     delete(s, length(s), 1);
[5Y2] 376:   if (s='') then Exit;
[825] 377:   if (a.Result=2) then LadeDatei(s)
[EL3] 378:   else SchreibeDatei(s);
[ZNØ] 379: end;
[ØØØ] 38Ø:
[6Y3] 381: procedure SpielerInit;
[9X4] 382: var z, f: Byte; n: String [2Ø];
[KE3] 383:     k: Char;   y, c, d: Word;
[5M4] 384:     Farben: Set Of Black..White;
[PL1] 385:     s: String;
[ØØØ] 386:
[CP4] 387:  function WaehleFarbe: byte;
[RNØ] 388:  var
[BE3] 389:    d: DialogObj;  b: byte;
[MUØ] 39Ø:  begin
[NK3] 391:    d.Zeige(8Ø, 1ØØ, 4ØØ, 22Ø);
[SA4] 392:    d.SetOpt('Spielereingabe',
[ZA7] 393: 'Wählen Sie die\gewünschte Farbe aus.','');
[UY3] 394:    for b:=black to white do
[3T3] 395:      if not (b in Farben) then
[MUØ] 396:      begin
[LR4] 397:        setfillstyle(solidfill, b);
[6W3] 398:        bar(96+b*18,175,114+b*18,21Ø);
[CT2] 399:        setcolor(white);
[2V4] 4ØØ:        rectangle(96+b*18,175,114+b*18,21Ø);
[ZNØ] 4Ø1:      end;
[P31] 4Ø2:    repeat
[P31] 4Ø3:     repeat
[TR2] 4Ø4:       HoleTaste(true);
[C75] 4Ø5:     Until (CurX>95) and (CurX<385) and
[6S3] 4Ø6:           (CurY>174) and (CurY<211);
```

```
[4Ø3]  4Ø7:      b:=(CurX - 96) div 18;
[NT3]  4Ø8:        until not (b in Farben);
[KZØ]  4Ø9:        d.Weg;
[HL2]  41Ø:        WaehleFarbe:=b;
[ZNØ]  411:      end;
[ØØØ]  412:
[MUØ]  413: begin
[XH5]  414:   for z:=1 to 6 do New(Spieler[z]);
[WRØ]  415:   z:=Ø;
[B12]  416:   Farben:=[];
[P31]  417:   Repeat
[571]  418:     Inc (z);
[YG5]  419:     SetFillStyle (SolidFill, White);
[UT2]  42Ø:     SetColor (Black);
[676]  421:     SetzeSchrift (SmallFont, HorizDir, 5,
[1S1]  422:                   LeftText);
[FY2]  423:     y:=2Ø + Pred (z) * 7Ø;
[KD3]  424:     Bar (5ØØ, y, 62Ø, y + 65);
[7P4]  425:     Rectangle(5Ø1,Succ(y),619,y+64);
[E63]  426:     OutTextXY(51Ø,y+15,
[3U4]  427:      'Spieler Nr. '+StrNum(z,2)+':');
[2S5]  428:     OutTextXY (51Ø, y + 45, 'Farbe [Tab]');
[GK1]  429:     n:=''; f:=Ø;
[1F4]  43Ø:     c:=51Ø; SetColor (LightRed);
[TT4]  431:     Rectangle (6Ø4, y + 39, 616, y + 54);
[P31]  432:     Repeat
[PE4]  433:       While (f in Farben) do Inc(f);
[BS4]  434:       SetFillStyle (SolidFill, f);
[LU3]  435:       Bar (6Ø5, y + 4Ø, 615, y + 53);
[KX3]  436:       SetWriteMode (XORPut);
[P31]  437:       Repeat
[DG4]  438:         Line(c,y+25,c,y+35); Delay(2Ø);
[DG4]  439:         Line(c,y+25,c,y+35); Delay(2Ø);
[K13]  44Ø:       Until KeyPressed;
[E34]  441:       SetWriteMode (CopyPut);
[LX1]  442:       k:=ReadKey;
[PA1]  443:       Case k Of
[C13]  444:       #9: f:=WaehleFarbe;
[3T2]  445:       #8: if (c > 51Ø) then
[MUØ]  446:             begin
[RF5]  447:               SetFillStyle(SolidFill, White);
[1Y4]  448:               d:=TextWidth (n [Length (n)]);
[WA1]  449:               Dec(c, d);
[L53]  45Ø:               Bar(c, y+25, c+d, y+38);
[3H3]  451:               Delete(n, Length(n), 1);
[ZNØ]  452:             end;
[WR4]  453:         else if (k>#31) and (k<>'\') and
[5G4]  454:         (k<>'#') and (Length (n)< 2Ø) And
[CW3]  455:         (TextWidth(k)+c<615) then
[MUØ]  456:         begin
[E41]  457:           n:=n + k;
[ZD3]  458:           SetColor(LightRed);
[LM6]  459:        SetTextJustify(LeftText,CenterText);
[1R3]  46Ø:           OutTextXY (c, y + 3Ø, k);
[XH3]  461:           Inc (c, TextWidth (k));
[ZNØ]  462:         end;
[ZNØ]  463:       end;
[9Y4]  464:     until (k=#13) and ((n<>'') or (z<>2));
[F52]  465:     if (n<>'') then
[MUØ]  466:     begin
[6D3]  467:       Farben:=Farben + [f];
[3H4]  468:       Spieler [z]^.Init (z, f, n);
[TF4]  469:       Spieler [z]^.ZeigeFigur;
[ZNØ]  47Ø:     end;
[773]  471:   until (n='') or (z=6);
[U53]  472:   if (n='') then Dec (z);
[ØU2]  473:   Spielerzahl:=z;
[8V6]  474:   SetFillStyle (ClosedotFill, LightGray);
[EY2]  475:   Bar (48Ø, Ø, 639, 479);
[ØW3]  476:   while (Spielerzahl=Ø) do
[MUØ]  477:   begin
[LW2]  478:     s:=HoleDateiname;
[AZ1]  479:     if (s='') then
[MUØ]  48Ø:     begin
[522]  481:       closegraph;
[DUØ]  482:       halt;
[ZNØ]  483:     end;
[542]  484:     ladedatei(s);
[ZNØ]  485:   end;
[ZNØ]  486: end;
[ØØØ]  487:
[AØ6]  488: procedure LoescheSpieler (n: Byte);
[175]  489: var a: AuswahlObj; e: DialogObj;
[D22]  49Ø:     s: Set Of 1..4Ø;
[PG3]  491:     b, k: Byte; p: LongInt;
[SV1]  492:     q: Integer;
[MUØ]  493: begin
[5E4]  494:   a.HoleEingabe (#Ø#17'Hinweis',
[ST9]  495:  Spieler[n]^.Name + ' hat schlecht gewirtsc
       haftet\und so' +
[PX9]  496:   ' sein Vermögen verloren. Er ist pleite !\
       Sein restliches' +
[N4A]  497:   ' Geld geht wie die Häuser an die Bank,\se
       ine Straßen wer' +
[E24]  498:  'den versteigert.', 'Ok');
[HL3]  499:   with Spieler[n]^ do
[MUØ]  5ØØ:   begin
[S32]  5Ø1:     s:=Strassen;
[NA1]  5Ø2:     Feld:=-1;
[122]  5Ø3:     ZeigeFigur;
[ZNØ]  5Ø4:   end;
[5P4]  5Ø5:   if (SpielerZahl=2) then Exit;
[WX2]  5Ø6:   Dec(SpielerZahl);
[MG4]  5Ø7:   for b:=n to SpielerZahl do
[MUØ]  5Ø8:   begin
[7R4]  5Ø9:     Spieler[b]:=Spieler[b + 1];
[Ø24]  51Ø:     Spieler[b]^.Nummer:=b;
[ZNØ]  511:   end;
[AH2]  512:   for b:=1 to 4Ø do
[1A2]  513:     if (b in s) then
[155]  514:       VersteigereStrasse(b, 2, true);
[ZNØ]  515: end;
[ØØØ]  516:
[YZ6]  517: procedure ZeigeIcon (n: Byte; a: Boolean);
[XM2]  518: var b, c, d: Byte;
[MUØ]  519: begin
[4A3]  52Ø:   c:=Pred (n) Mod 3 * 5Ø;
[AB3]  521:   d:=Pred (n) Div 3 * 5Ø;
[P24]  522:   if (Icons[n] <> nil) then
[X25]  523:     PutImage(489+c,329+d,Icons[n]^,
                       NormalPut)
[WT1]  524:   else
[ZRØ]  525:   begin
[MUØ]  526:     SetColor (LightGray);
[GN3]  527:     SetFillStyle(SolidFill,DarkGray);
[1W5]  528:     SetLineStyle(SolidLn,Ø,ThickWidth);
[V26]  529:     SetLineStyle(SolidLn,Ø,ThickWidth);
[4W3]  53Ø:     Bar(49Ø+c,33Ø+d,535+c,375+d);
[4Z4]  531:     Rectangle(49Ø+c,33Ø+d,535+c,375+d);
[7Y5]  532:     SetLineStyle(SolidLn,Ø,NormWidth);
[TX6]  533:     SetTextJustify(CenterText,CenterText);
[ZA1]  534:     Case n Of
[3S1]  535:     1: begin
[ZZ3]  536:          Rectangle(493, 333, 532, 372);
[EM2]  537:          Circle(5Ø9, 34Ø, 3);
[1N2]  538:          Circle(512, 352, 3);
[VN2]  539:          Circle(525, 365, 3);
[2A5]  54Ø:          SetFillStyle(SolidFill,White);
[NM4]  541:          FloodFill(494,334,LightGray);
[ZNØ]  542:        end;
[551]  543:     2: begin
[FG5]  544:          SetFillStyle (SolidFill, Green);
[YS3]  545:          SetColor (LightGreen);
[L44]  546:          Rectangle (55Ø, 35Ø, 575, 37Ø);
[F73]  547:          Line (55Ø, 35Ø, 562, 335);
[H83]  548:          Line (562, 335, 575, 35Ø);
[DØ5]  549:          FloodFill (551, 351, LightGreen);
[415]  55Ø:          FloodFill (562, 337, LightGreen);
```

```
[ZNØ]  551:        end;
[751]  552:     3: begin
[1B6]  553:        SetTextStyle(SmallFont,HorizDir,1Ø);
[UT2]  554:          SetColor (Black);
[5U5]  555:          SetFillStyle (SolidFill, Yellow);
[5W2]  556:          Circle (612, 352, 19);
[H94]  557:          FloodFill (612, 352, Black);
[MR3]  558:          OutTextXY (616, 347, '5');
[ZNØ]  559:        end;
[951]  56Ø:     4: begin
[FG5]  561:          SetFillStyle (SolidFill, Green);
[YS3]  562:          SetColor (LightGreen);
[934]  563:          Rectangle (51Ø, 4ØØ, 525, 42Ø);
[V73]  564:          Line (51Ø, 4ØØ, 517, 385);
[Ø83]  565:          Line (517, 385, 525, 4ØØ);
[SZ4]  566:          FloodFill (511, 4Ø1, LightGreen);
[615]  567:          FloodFill (517, 387, LightGreen);
[V26]  568:        SetLineStyle(SolidLn,Ø,ThickWidth);
[U63]  569:          Line (494, 4ØØ, 5Ø5, 4ØØ);
[K73]  57Ø:          Line (5Ø1, 396, 5Ø5, 4ØØ);
[V63]  571:          Line (5Ø1, 4Ø4, 5Ø5, 4ØØ);
[7Y5]  572:          SetLineStyle(SolidLn,Ø,NormWidth);
[ZNØ]  573:        end;
[MA1]  574:     5,6:begin
[1B6]  575:        SetTextStyle(SmallFont,HorizDir,1Ø);
[PE5]  576:        if (n=5) then OutTextXY(564,399,'?')
[MN4]  577:          else OutTextXY (614, 399, 'X');
[ZNØ]  578:        end;
[F51]  579:     7: begin
[HT5]  58Ø:          SetFillStyle(SolidFill,LightRed);
[423]  581:          Bar (495, 435, 53Ø, 47Ø);
[YG5]  582:          SetFillStyle (SolidFill, White);
[L13]  583:          Bar (5ØØ, 44Ø, 525, 465);
[UT2]  584:          SetColor (Black);
[854]  585:          Rectangle (495, 435, 53Ø, 47Ø);
[144]  586:          Rectangle (5ØØ, 44Ø, 525, 465);
[NF2]  587:          for b:=1 to 3 do
[2X3]  588:            Line(5ØØ+b*6,44Ø,5ØØ+b*6,465);
[ZNØ]  589:        end;
[H51]  59Ø:     8: begin
[RE5]  591:          SetFillStyle (SolidFill, Black);
[X13]  592:          Bar (545, 435, 58Ø, 47Ø);
[ØC3]  593:          SetColor (DarkGray);
[1W5]  594:          SetFillStyle(SolidFill,DarkGray);
[EZ3]  595:          Sector (562, 452, Ø, 36Ø, 5, 5);
[C23]  596:          Bar (579, 439, 58Ø, 44Ø);
[ZNØ]  597:        end;
[L51]  598:     9: begin
[7B6]  599:         SetTextStyle(SmallFont,HorizDir,8);
[L24]  6ØØ:         OutTextXY (614, 449, 'DOS');
[ZNØ]  6Ø1:       end;
[ZNØ]  6Ø2:   end;
[226]  6Ø3:   GetMem(Icons[n],ImageSize(489,329,536,
                                              376));
[FSØ]  6Ø4:   if (Icons[n] <> nil) then
[DG4]  6Ø6:     GetImage(489+c,329+d,536+c,376+d,
                         Icons[n]^);
[M22]  6Ø7:
[ZNØ]  6Ø8:   end;
[AY1]  6Ø9:   if Not a then
[MUØ]  61Ø:     begin
[CK2]  611:       SetColor (Red);
[V26]  612:       SetLineStyle(SolidLn,Ø,ThickWidth);
[KY3]  613:       Line(49Ø+c,33Ø+d,535+c,375+d);
[KY3]  614:       Line(535+c,33Ø+d,49Ø+c,375+d);
[7Y5]  615:       SetLineStyle(SolidLn,Ø,NormWidth);
[ZNØ]  616:     end;
[ZNØ]  617: end;
[ØØØ]  618:
[7D3]  619: procedure MacheZug;
[GP3]  62Ø: var x, y: Word; b: Byte;
[XW4]  621:     Ferlaubt: Boolean;
[KA3]  622:     Modus: Byte; k: Char;
[4A3]  623:     Erlaubt: Set Of 1..9;
[8B2]  624:     a: AuswahlObj;
[MUØ]  625: begin
[H53]  626:   Gewuerfelt:=False;
[GC2]  627:   Fertig:=False;
[Z33]  628:   Modus:=5; Pasch:=Ø;
[AL3]  629:   Erlaubt:=[1, 4..5, 7..9];
[P31]  63Ø:   Repeat
[X13]  631:     if Gewuerfelt then
[3A4]  632:      Erlaubt:=Erlaubt-[1]+[6];
[UY7]  633:     if (Spieler [SpielerDran]^.Strassen<>[]
       ) then
[MY3]  634:       Erlaubt:=Erlaubt + [2..3]
[LU4]  635:     else Erlaubt:=Erlaubt-[2..3];
[E96]  636:    if Not (Modus In Erlaubt) then Modus:=5;
[CD3]  637:     NeuZeichnen:=False;
[RK3]  638:     ZeigeSpielerStatus;
[VF2]  639:     for b:=1 to 6 do
[794]  64Ø:       ZeigeIcon(b, b In Erlaubt);
[1B6]  641:     SetLineStyle(SolidLn, Ø, ThickWidth);
[B43]  642:     SetColor (Yellow);
[E44]  643:     x:=Pred (Modus) Mod 3 * 5Ø;
[M54]  644:     y:=Pred (Modus) Div 3 * 5Ø;
[Ø75]  645:     Rectangle(49Ø+x,33Ø+y,535+x,375+y);
[7Y5]  646:     SetLineStyle(SolidLn,Ø,NormWidth);
[P31]  647:     Repeat
[TR2]  648:       HoleTaste(true);
[BT3]  649:       if (LastReadKey>='Ø') and
[CM3]  65Ø:          (LastReadKey<='9') then
[5G6]  651:  SetPos(512+(ord(LastReadKey)-49) mod 3*5Ø,
[RT5]  652:  352+(2-(ord(LastReadKey)-49) div 3)*5Ø);
[DC3]  653:       if (MausKnopf=2) then
[MUØ]  654:       begin
[5N3]  655:         if not Gewuerfelt then
[MUØ]  656:         begin
[ZB3]  657:           CurX:=495;  CurY:=335;
[4MØ]  658:         end
[WP1]  659:         else begin
[A83]  66Ø:           CurX:=63Ø; CurY:=39Ø;
[ZNØ]  661:         end;
[ZNØ]  662:       end;
[9C5]  663:       if (CurX > 489) And (CurY > 329) then
[MUØ]  664:       begin
[ST3]  665:         x:=(CurX - 489) Div 5Ø + 1;
[VW2]  666:         if (x > 3) then x:=3;
[ET3]  667:         y:=(CurY - 329) Div 5Ø + 1;
[ZW2]  668:         if (y > 3) then y:=3;
[CM2]  669:         b:=x + Pred (y) * 3;
[4B4]  67Ø:         NeuZeichnen:=b In Erlaubt;
[ØD3]  671:         if (b In Erlaubt) then
[V91]  672:         Case b Of
[351]  673:          1: begin
[2L1]  674:               Wuerfeln;
[BL2]  675:               if (Pasch=Ø) then
[H23]  676:                  Gewuerfelt:=true;
[ZNØ]  677:             end;
[BB2]  678:          2..5: Modus:=b;
[FL2]  679:          6: Fertig:=True;
[U53]  68Ø:          7: Gefaengniskarte;
[5Z1]  681:          8: SpielOpt;
[LS1]  682:          9: begin
[CHA]  683:               a.HoleEingabe(#Ø#18'Spielend
       e','Sie verlassen das Spiel.','Ok\Ab');
[4Z3]  684:                  Spielende:=(a.Result=1);
[ZNØ]  685:             end;
[ZNØ]  686:         end;
[4MØ]  687:       end
[WP1]  688:     else begin
[Z32]  689:       Case Modus Of
[A94]  69Ø:       2: BauAktion(SpielerDran);
[H74]  691:       3: Hypothek(SpielerDran);
[XS3]  692:       4: Handel(SpielerDran);
[661]  693:       5: Info;
[ZNØ]  694:       end;
[9W3]  695:       if (Modus In [2..4]) then
```

```
[X73]  696:                   NeuZeichnen:=True;
[ZNØ]  697:       end;
[RF4]  698:       for b:=1 to SpielerZahl do
[HL4]  699:         if Spieler [b]^.Pleite then
[Z43]  7ØØ:           LoescheSpieler (b);
[PX6]  7Ø1:   until NeuZeichnen or Spielende or Fertig;
[CD3]  7Ø2:     NeuZeichnen:=False;
[5F4]  7Ø3:   Until Spielende or Fertig;
[ZNØ]  7Ø4: end;
[ØØØ]  7Ø5:
[VB5]  7Ø6: Const GraphDriver: Integer=VGA;
[FG4]  7Ø7:       GraphMode  : Integer=VGAHi;
[YG4]  7Ø8: var b: Byte; a: AuswahlObj;
[TT5]  7Ø9:     p: PaletteType;  Old33Vec: Pointer;
[MUØ]  71Ø: begin
[XE3]  711:   DirectVideo:=False;
[L13]  712:   CheckBreak:=False;
[4Z7]  713:   Writeln(#13#1Ø'MONOPOLY  Version 1.1'#13#
       1Ø'(c) 1992 ' +
[BP3]  714:  'by Andreas Knab'#13#1Ø);
[XN1]  715:   Delay(5ØØ);
[WH6]  716:   RegisterBGIDriver(@EGAVGADriverProc);
[ED6]  717:   RegisterBGIFont(@SansSerifFontProc);
[4R5]  718:   RegisterBGIFont(@SmallFontProc);
[X56]  719:   RegisterBGIFont(@TriplexFontProc);
[L86]  72Ø:   InitGraph (GraphDriver, GraphMode, '');
[F34]  721:   if (GraphResult <> Ø) then
[MUØ]  722:   begin
[ZN5]  723:     Writeln ('Grafikmodus kann nicht ',
[GM4]  724:     'eingeschaltet werden !'#13#1Ø +
[NS6]  725:  'Entweder VGA-Karte oder Grafiktreiber ',
[Y43]  726:     'nicht vorhanden !');
[DUØ]  727:       Halt;
[ZNØ]  728:     end;
[YN4]  729:   if (InitAndGetStatus=Ø) then
[MUØ]  73Ø:     begin
[ET3]  731:       GetIntVec($33, Old33Vec);
[G85]  732:       SetIntVec($33, @MausInterrupt);
[XU1]  733:       InitMouse;
[FB7]  734:       a.HoleEingabe(#Ø#19'Programminitialis
       ierung'+
[YL6]  735:  'Maus kann nicht initialisiert werden !\'+
[PL8]  736:  'Entweder k. Maus vorhanden oder\kein Maus
       treiber'+
[BC7]  737:  'installiert.\Interne Mausemulation wird'+
[8K2]  738:  'installiert.');
[4MØ]  739:     end
[M83]  74Ø:   else Old33Vec:=nil;
[1TL]  741:   Randomize;
[G66]  742:   for b:=1 to 15 do SetPalette (b, Black);
[PC2]  743:   InitStrassen;
[922]  744:   InitKarten;
[XH1]  745:   InitBild;
[3N3]  746:   GetDefaultPalette (p);
[BZ2]  747:   SetAllPalette (p);
[VP2]  748:   SetXRange(Ø, 639);
[6R2]  749:   SetYRange(Ø, 479);
[T72]  75Ø:   SpielerInit;
[665]  751:   for b:=1 to 9 do Icons[b]:=nil;
[SP5]  752:   for b:=7 to 9 do ZeigeIcon (b, True);
[4P2]  753:   SpielerDran:=1;
[1Y2]  754:   Spielende:=False;
[ZB2]  755:   FreiParken:=Ø;
[P31]  756:   Repeat
[UK1]  757:     MacheZug;
[113]  758:     Inc (SpielerDran);
[7W5]  759:     if (SpielerDran > SpielerZahl) then
[4P2]  76Ø:       SpielerDran:=1;
[EU2]  761:   Until Spielende;
[3T3]  762:   if (SpielerZahl=1) then
[3T4]  763:     a.HoleEingabe (#Ø#2Ø'Spielende',
[7P7]  764: 'Es gibt einen Sieger: ' + Spieler[1]^.Name
       +' !\' +
[H98]  765: 'Als letzter Spieler, der noch Besitz hat,\
       hat er ' +
[KDA]  766: 'diese Runde Monopoly gewonnen.\\HERZLICHEN
       GLÜCKWUNSCH !',
[N41]  767:  'Ok\?');
[XT5]  768:   a.HoleEingabe (#Ø#21'Spiel verlassen',
[XM6]  769:  'Monopoly Version 1.1\\(c) 1992 Andreas '+
[N94]  77Ø:  'Knab', 'Auf Wiedersehen');
[9X3]  771:   if (Old33Vec <> nil) then
[6U3]  772:       SetIntVec($33, Old33Vec);
[522]  773:   CloseGraph;
[KNØ]  774: end.
```

Listing 8. Das Programm »monopoly.pas« ist sehr einfach zu bedienen.

```
[G74]    1: (Programm: hilfconv.pas
[UP6]    2:  Funktion: Konvertierung der Hilfedatei
[G64]    3:  Sprache:  Turbo Pascal 6.Ø
[ZC3]    4:  Autor:    Andreas Knab
[915]    5:  (c) 1993 DMV Widuch GmbH & Co. KG)
[ØØØ]    6:
[P26]    7: ($A+,B-,D-,E+,F-,G-,I-,L-,N-,O-,R-,S-,V-,X+
         )
[SW2]    8: ($M 16384,Ø,65536Ø)
[BW4]    9: program  HilfeKonvertierer;
[ØØØ]   1Ø:
[S93]   11: var Eingabe : Text;
[SG2]   12:     Ausgabe : File;
[5L2]   13:     Zeile   : String;
[AY4]   14:     Tabelle : Array[1..5Ø] of Word;
[WU1]   15:     b       : Byte;
[X92]   16:     e       : Integer;
[ØØØ]   17:
[MUØ]   18: begin
[165]   19:   assign(Eingabe, 'MONHILFE.TXT');
[X75]   2Ø:   assign(Ausgabe, 'MONOPOLY.HLP');
[9H2]   21:   reset(Eingabe);
[T93]   22:   rewrite(Ausgabe, 1);
[W26]   23:   FillChar(Tabelle, SizeOf(Tabelle), Ø);
[2S4]   24:   Blockwrite(Ausgabe, Tabelle,
[4R2]   25:                    SizeOf(Tabelle));
[HU3]   26:   while not eof(Eingabe) do
[MUØ]   27:   begin
[2S3]   28:     readln(Eingabe, Zeile);
[ØM3]   29:     if (Zeile[1] <> '#') then
[HW4]   3Ø:       BlockWrite(Ausgabe, Zeile[Ø],
[NM2]   31:                  Length(Zeile) + 1)
[WP1]   32:     else begin
[434]   33:       Val(Copy(Zeile, 2, 4), b, e);
[K25]   34:       Tabelle[b] := FilePos(Ausgabe);
[ZNØ]   35:     end;
[ZNØ]   36:   end;
[3K2]   37:   Seek(Ausgabe, Ø);
[2S4]   38:   Blockwrite(Ausgabe, Tabelle,
[4R2]   39:                    SizeOf(Tabelle));
[6G2]   4Ø:   Close(Ausgabe);
[ØG2]   41:   Close(Eingabe);
[KNØ]   42: end.
```

Listing 9. »Hilfconv.pas« konvertiert die Hilfedatei »monhilf.pas«.

```
[T6Ø]    1: #1
[DX2]    2: _Würfelanzeige
[ØØØ]    3:
[9C5]    4: Dieses Dialogfenster zeigt Ihnen
[RC5]    5: den letzten Wurf an, den Sie ge-
[MP4]    6: worfen haben. Ihre Spielfigur
[695]    7: wird jetzt um die entsprechende
[DØ5]    8: Anzahl von Augen im Uhrzeiger-
[FL5]    9: sinn auf dem Spielfeld vorwärts-
[P91]   1Ø: bewegt.
[Ø4Ø]   11: @
[W6Ø]   12: #2
[9K2]   13: _Kartenanzeige
[ØØØ]   14:
[2C5]   15: Sie haben im Info-Modus ein Feld
[2P5]   16: auf dem Spielbrett angewählt und
[UR5]   17: sehen jetzt die Besitzrechtskar-
[WS2]   18: te zu diesem Feld.
[Ø4Ø]   19: @
[27Ø]   2Ø: #4
[331]   21: _Pasch
[ØØØ]   22:
[X95]   23: Sie befinden sich zur Zeit im Ge-
[985]   24: fängnis und haben jetzt einen
[ES5]   25: Pasch geworfen, was bedeutet, daß
[G56]   26: Sie sofort ohne Bußgeld zu zahlen
[RM5]   27: aus dem Gefängnis entlassen wer-
[SF5]   28: den. Sie dürfen aber nicht noch
[8R2]   29: einmal würfeln.
[Ø4Ø]   3Ø: @
[57Ø]   31: #5
[8Ø2]   32: _Kein Pasch
[ØØØ]   33:
[X95]   34: Sie befinden sich zur Zeit im Ge-
[WE5]   35: fängnis, haben aber keinen Pasch
[4Y5]   36: gewürfelt und dürfen so auch nicht
[GM5]   37: weiterziehen. Ist dies bereits ihr
[6G5]   38: dritter Wurf im Gefängnis, dann
[4P5]   39: müssen Sie ein Bußgeld von DM 1ØØØ
[995]   4Ø: bezahlen und werden entlassen.
[Ø4Ø]   41: @
[87Ø]   42: #6
[331]   43: _Pasch
[ØØØ]   44:
[RC5]   45: Sie haben jetzt dreimal hinter-
[HY4]   46: einander einen Pasch geworfen,
[RG5]   47: was bedeutet, daß Sie jetzt so-
[4P5]   48: fort in das Gefängnis müssen und
[3Ø5]   49: nicht noch einmal würfeln dür-
[1PØ]   5Ø: fen.
[Ø4Ø]   51: @
[B7Ø]   52: #7
[R32]   53: _Frei Parken
[ØØØ]   54:
[B75]   55: Alle Steuergelder und die Buß-
[K75]   56: gelder, die durch Ereignis- und
[YS5]   57: Gemeinschaftskarten bezahlt wer-
[585]   58: den müssen, werden im Frei Par-
[EP5]   59: ken gelagert. Derjenige Spieler,
[BZ4]   6Ø: der als erster auf dieses Feld
[HY3]   61: kommt, erhält das Geld.
[Ø4Ø]   62: @
[E7Ø]   63: #8
[KW3]   64: _Gefängniskartenmenü
[ØØØ]   65:
[765]   66: In diesem Menü stehen Optionen
[PH4]   67: bereit, die es erlauben, ein
[HR5]   68: Bußgeld von DM 1ØØØ zu bezahlen,
[B65]   69: um aus dem Gefängnis entlassen
[7V4]   7Ø: zu werden, Karten zu benutzen
[Z44]   71: oder mit ihnen zu handeln.
[Ø4Ø]   72: @
[H7Ø]   73: #9
[YF4]   74: _Gefängniskartenverkauf
[ØØØ]   75:
[VV4]   76: Geben Sie in diesem Dialogfen-
[EW4]   77: ster den Betrag ein, der für
[XM4]   78: die Karte, die Sie zum Verkauf
[8A5]   79: angeboten haben, geboten wurde.
[EB4]   8Ø: Der Spieler, der die Karte
[1K4]   81: kauft, bezahlt diesen Betrag
[FZ1]   82: dann an Sie.
[Ø4Ø]   83: @
[TCØ]   84: #1Ø
[AU3]   85: _Gefängniskartenkauf
[ØØØ]   86:
[N55]   87: Der aktuelle Spieler will eine
[2C4]   88: Karte kaufen, die sich im Be-
[L85]   89: sitz eines zweiten Spielers be-
[G15]   9Ø: findet. Ist dieser zum Verkauf
[1Ø5]   91: bereit, drücken Sie Ok. Dann
[765]   92: wird in einem weiteren Fenster
[UZ4]   93: der Preis der Karte abgefragt.
[Ø4Ø]   94: @
[XCØ]   95: #11
[YF4]   96: _Gefängniskartenverkauf
[ØØØ]   97:
[N55]   98: Der aktuelle Spieler will eine
[2C4]   99: Karte kaufen, die sich im Be-
[L85]  1ØØ: sitz eines zweiten Spielers be-
[8M4]  1Ø1: findet. Geben Sie jetzt den
[WX4]  1Ø2: Preis ein, zu dem die Karte den
[M24]  1Ø3: Besitzer wechseln soll.
[Ø4Ø]  1Ø4: @
[1DØ]  1Ø5: #12
[PB3]  1Ø6: _Datei laden/sichern
[ØØØ]  1Ø7:
[TK4]  1Ø8: Geben Sie hier den Dateinamen
[S15]  1Ø9: ein, unter dem Sie das aktuelle
[FN4]  11Ø: Spiel abspeichern bzw. ein an-
[YØ5]  111: deres laden wollen. Bei Bedarf
[TG5]  112: wird automatisch die Endung .MON
[AZ4]  113: an den eingegebenen Namen ange-
[B71]  114: hängt.
[Ø4Ø]  115: @
[5DØ]  116: #13
[H22]  117: _Ladefehler
[ØØØ]  118:
[P55]  119: Die angegebene Datei kann nicht
[M65]  12Ø: geladen werden, wahrscheinlich
[YS4]  121: haben Sie nicht die korrekte
[E15]  122: Diskette im Laufwerk oder Sie
[5W4]  123: befinden sich im falschen Pfad
[GL4]  124: oder Sie haben sich ganz ein-
[GC2]  125: fach vertippt.
[Ø4Ø]  126: @
[9DØ]  127: #14
[XM3]  128: _Falsches Dateiformat
[ØØØ]  129:
[WX4]  13Ø: Die angegebene Datei ist keine
[T54]  131: Monopoly 1.1-Spieledatei.
[Ø4Ø]  132: @
[DDØ]  133: #15
[XV2]  134: _Speicherfehler
[ØØØ]  135:
[P55]  136: Die angegebene Datei kann nicht
[7R5]  137: auf Diskette/Platte gespeichert
[XV4]  138: werden, Sie haben entweder un-
[7G5]  139: gültige Zeichen verwendet oder
[9S4]  14Ø: Ihre Diskette kann keine wei-
[714]  141: teren Dateien aufnehmen.
[Ø4Ø]  142: @
[HDØ]  143: #16
[MK2]  144: _Optionenmenü
[ØØØ]  145:
```

```
[XA5]  146: In diesem Menü sind die Befehle
[NT4]  147: zum Speichern und Laden von
[PB5]  148: Spielstanddateien auf Diskette/
[YK3]  149: Festplatte enthalten.
[Ø4Ø]  15Ø: @
[NDØ]  151: #17
[2L3]  152: _Spieler scheidet aus
[ØØØ]  153:
[265]  154: Der angegebene Spieler hat keine
[6R5]  155: Möglichkeit mehr, seine Schulden
[925]  156: zu bezahlen, somit scheidet er
[7Y4]  157: aus dem aktuellen Spiel aus.
[3Z4]  158: Seine Häuser gehen an die Bank,
[WT5]  159: die Straßen werden versteigert.
[Ø4Ø]  16Ø: @
[TDØ]  161: #18
[7V1]  162: _Spielende
[ØØØ]  163:
[ØØ5]  164: Sie haben das DOS-Zeichen ange-
[6Ø5]  165: wählt. Wenn Sie das Programm
[W45]  166: verlassen wollen, bestätigen
[KT2]  167: Sie jetzt mit Ok.
[5H4]  168: Achtung : Das aktuelle Spiel
[KR4]  169: geht, wenn nicht gespeichert,
[9L2]  17Ø: dabei verloren.
[Ø4Ø]  171: @
[XDØ]  172: #19
[C74]  173: _Keine Maus installiert
[ØØØ]  174:
[6L4]  175: An ihrem Computer ist keine
[ZR4]  176: Maus angeschlossen bzw. kein
[1G5]  177: Maustreiber installiert, somit
[VL4]  178: kann Monopoly 1.1 nur mit der
[274]  179: Tastatur bedient werden.
[Ø4Ø]  18Ø: @
[WCØ]  181: #2Ø
[TV2]  182: _Sieger gefunden
[ØØØ]  183:
[695]  184: Bis auf einen Spieler sind alle
[Z15]  185: Spieler ausgeschieden, es gibt
[AB5]  186: nur noch eine Person, die Besitz
[9K5]  187: hat und somit das Spiel gewinnt.
[Ø4Ø]  188: @
[ØDØ]  189: #21
[H74]  19Ø: _Programm wird verlassen
[ØØØ]  191:
[GZ4]  192: Sie haben den Befehl Spiel ver-
[TØ5]  193: lassen angewählt, Monopoly 1.1
[Ø52]  194: wird beendet.
[Ø4Ø]  195: @
[4DØ]  196: #22
[L14]  197: _Ereigniskartenanzeige
[ØØØ]  198:
[MX4]  199: Sie sind auf ein Ereignisfeld
[X85]  2ØØ: gekommen und müssen jetzt die
[GW4]  2Ø1: Anweisung der obersten Ereig-
[XN3]  2Ø2: niskarte ausführen.
[Ø4Ø]  2Ø3: @
[8DØ]  2Ø4: #23
[G15]  2Ø5: _Gemeinschaftskartenanzeige
[ØØØ]  2Ø6:
[L95]  2Ø7: Sie sind auf ein Gemeinschafts-
[C65]  2Ø8: feld gekommen und müssen jetzt
[2Ø5]  2Ø9: die Anweisung der obersten Ge-
[T35]  21Ø: meinschaftskarte ausführen.
[Ø4Ø]  211: @
[CDØ]  212: #24
[EK3]  213: _Gemeinschaftskarte
[ØØØ]  214:
[HZ4]  215: Die von Ihnen gezogene Gemein-
[615]  216: schaftskarte fordert Sie auf,
[WG4]  217: entweder DM 2ØØ Strafe zu be-
[Z45]  218: zahlen oder eine Ereigniskarte
[1N4]  219: zu ziehen. Sie haben die Wahl !
[Ø4Ø]  22Ø: @
[GDØ]  221: #25
[ØL2]  222: _Zusatzsteuer
[ØØØ]  223:
[Y85]  224: Sie sind auf das Zusatzsteuer-
[C65]  225: feld gekommen und müssen jetzt
[S75]  226: Zusatzsteuer bezahlen : Entwe-
[294]  227: der DM 4ØØØ oder 1Ø% Ihres Ge-
[AN5]  228: samtvermögens (Straßen, Bargeld,
[YS1]  229: Häuser...)
[Ø4Ø]  23Ø: @
[MDØ]  231: #26
[8C4]  232: _Straßenversteigerung
[ØØØ]  233:
[BC5]  234: Es wurde eine Straße verstei-
[KK4]  235: gert, der Preis, der geboten
[B15]  236: wurde, muß jetzt in das Dia-
[KW4]  237: logfenster eingegeben werden.
[Ø4Ø]  238: @
[SDØ]  239: #27
[PE3]  24Ø: _Hypothekenhinweis
[ØØØ]  241:
[BC5]  242: Es wurde eine Straße verstei-
[CR4]  243: gert. Da sie hypothekarisch
[CU5]  244: belastet ist, müssen zusätzlich
[725]  245: zum Verkaufspreis 1Ø% Hypothe-
[784]  246: kenzinsen bezahlt werden.
[Ø4Ø]  247: @
[WDØ]  248: #28
[ZH4]  249: _Straße geht an die Bank
[ØØØ]  25Ø:
[BC5]  251: Es wurde eine Straße verstei-
[7L4]  252: gert, doch kein Spieler hat
[A75]  253: sich für sie interessiert und
[514]  254: geboten. Deshalb geht die
[CE5]  255: Straße jetzt in den Besitz der
[LU1]  256: Bank über.
[Ø4Ø]  257: @
[QEØ]  258: #29
[7C3]  259: _Straßenverkauf
[ØØØ]  26Ø:
[SW4]  261: Der aktuelle Spieler ist auf
[Z75]  262: ein Straßenfeld gekommen, das
[UV4]  263: noch niemand gehört. Er hat
[EN4]  264: jetzt die Möglichkeit, diese
[TH3]  265: Straße zu erwerben.
[Ø4Ø]  266: @
[ZCØ]  267: #3Ø
[541]  268: _Miete
[ØØØ]  269:
[SW4]  27Ø: Der aktuelle Spieler ist auf
[Z75]  271: ein Straßenfeld gekommen, das
[WX4]  272: einem anderen Spieler gehört
[NU4]  273: und nicht mit einer Hypothek
[Ø55]  274: belastet ist. Er muß an diesen
[945]  275: Miete bezahlen, die von der Be-
[PG4]  276: bauung der Straße abhängt.
[Ø4Ø]  277: @
[3DØ]  278: #31
[R32]  279: _Frei Parken
[ØØØ]  28Ø:
[RY4]  281: Sie sind auf das Feld Frei Par-
[UX4]  282: ken gekommen und erhalten den
[935]  283: Geldbetrag, der sich hier ange-
[T42]  284: sammelt hat.
[Ø4Ø]  285: @
[7DØ]  286: #32
[A62]  287: _Spielstand
[ØØØ]  288:
[AV4]  289: Dieses Fenster zeigt eine Sta-
```

```
[ET4]  29Ø: tistik zum aktuellen Besitz
[ZB2]  291: aller Spieler.
[Ø4Ø]  292: @
[BDØ]  293: #33
[XK1]  294: _Bezahlen
[ØØØ]  295:
[LZ4]  296: Sie müssen einen bestimmten
[2W4]  297: Geldbetrag bezahlen und haben
[DV4]  298: jetzt die Möglichkeit, hierzu
[R65]  299: Straßen zu verkaufen, Häuser
[LØ5]  3ØØ: abzubauen oder Hypotheken auf-
[8T4]  3Ø1: zunehmen, falls Ihr Bargeld
[1S2]  3Ø2: nicht ausreicht.
[Ø4Ø]  3Ø3: @
[FDØ]  3Ø4: #34
[VØ3]  3Ø5: _Bauen unmöglich
[ØØØ]  3Ø6:
[UT4]  3Ø7: An der von Ihnen angewählten
[HK4]  3Ø8: Position kann kein Haus ge-
[M64]  3Ø9: baut bzw. abgebaut werden.
[Ø4Ø]  31Ø: @
[LDØ]  311: #35
[SG1]  312: _Baumenü
[ØØØ]  313:
[4F4]  314: Dieses Dialogfenster zeigt
[YF4]  315: Ihnen die aktuelle Bebauung
[H55]  316: des von Ihnen angewählten Fel-
[KE4]  317: des an und bietet Ihnen die
[W75]  318: Möglichkeit, ein Haus zu bauen
[185]  319: bzw. abzubauen (zu entsprechen-
[M62]  32Ø: den Preisen).
[Ø4Ø]  321: @
[RDØ]  322: #36
[N75]  323: _Hypothek aufnehmen unmöglich
[ØØØ]  324:
[UT4]  325: An der von Ihnen angewählten
[E85]  326: Position können Sie keine Hy-
[LC3]  327: potheken aufnehmen.
[Ø4Ø]  328: @
[VDØ]  329: #37
[RK3]  33Ø: _Hypothek abbezahlen
[ØØØ]  331:
[TX4]  332: Sie haben mit der Option Hypo-
[6N4]  333: thek ein Feld angewählt, das
[DY4]  334: hypothekarisch belastet ist.
[UR5]  335: Sie können diese Hypothek jetzt
[U74]  336: abbezahlen oder abbrechen.
[Ø4Ø]  337: @
[2DØ]  338: #38
[TF5]  339: _Hypothek abbezahlen unmöglich
[ØØØ]  34Ø:
[645]  341: Sie haben aus dem Bezahlenmenü
[2N5]  342: die Option Hypothek ausgewählt
[ØV4]  343: und jetzt ein Feld angewählt,
[NY4]  344: das bereits belastet ist. Aus
[375]  345: dem Bezahlenmenü können Sie
[Ø75]  346: aber keine Hypothek abbezahlen.
[Ø4Ø]  347: @
[3EØ]  348: #39
[BG3]  349: _Hypothek aufnehmen
[ØØØ]  35Ø:
[TX4]  351: Sie haben mit der Option Hypo-
[6N4]  352: thek ein Feld angewählt, das
[5V4]  353: noch nicht hypothekarisch be-
[4X1]  354: lastet ist.
[GC5]  355: Sie können jetzt eine Hypothek
[K34]  356: aufnehmen oder abbrechen.
[Ø4Ø]  357: @
[2DØ]  358: #4Ø
[H72]  359: _Spielerwahl
[36Ø]  36Ø:
[6N5]  361: Sie wollen eine Straße bzw. eine
[L15]  362: Gefängnisfreikarte verkaufen,
[LU5]  363: oder es wird gerade eine Straße
[D85]  364: versteigert. Handeln Sie jetzt
[ZN4]  365: untereinander aus, wer das be-
[CU4]  366: treffende Objekt kaufen will
[Y73]  367: oder brechen Sie ab.
[Ø4Ø]  368: @
[6DØ]  369: #41
[TZ3]  37Ø: _Handel unmöglich
[ØØØ]  371:
[R95]  372: Die angewählte Straße gehört
[V74]  373: noch niemand, deshalb kann
[N25]  374: mit ihr auch noch nicht gehan-
[322]  375: delt werden.
[Ø4Ø]  376: @
[ADØ]  377: #42
[VDJ]  378: _Verkauf unmöglich
[ØØØ]  379:
[FL5]  38Ø: Die angewählte Straße kann nicht
[YB5]  381: verkauft werden, da sie oder an-
[BW5]  382: dere Straßen der gleichen Gruppe
[MA5]  383: noch bebaut sind. Diese Häuser
[N65]  384: müssen zuerst abgebaut werden.
[Ø4Ø]  385: @
[EDØ]  386: #43
[TG1]  387: _Verkauf
[ØØØ]  388:
[BU4]  389: Es wurde mit der Handeloption
[HM5]  39Ø: ein Straßenfeld angewählt, das
[P85]  391: nicht dem aktuellen Spieler ge-
[XY4]  392: hört. Ist der Besitzer dieser
[8W4]  393: Straße mit dem Handel einver-
[1K5]  394: standen, so wählen Sie jetzt Ok,
[WD5]  395: ansonsten können Sie abbrechen.
[Ø4Ø]  396: @
[KDØ]  397: #44
[NL4]  398: _Straßenkauf unmöglich
[ØØØ]  399:
[645]  4ØØ: Sie haben aus dem Bezahlenmenü
[YV4]  4Ø1: die Option Handel ausgewählt
[ØV4]  4Ø2: und jetzt ein Feld angewählt,
[4U4]  4Ø3: das Ihnen nicht gehört. Aus
[375]  4Ø4: dem Bezahlenmenü können Sie
[VT5]  4Ø5: aber keine fremde Straße kaufen,
[SØ5]  4Ø6: sondern nur eigene verkaufen.
[Ø4Ø]  4Ø7: @
[PDØ]  4Ø8: #45
[LB4]  4Ø9: _Straßenverkaufspreis
[ØØØ]  41Ø:
[LW4]  411: Geben Sie in dieses Dialogfen-
[WP4]  412: ster den Preis ein, zu dem die
[6P5]  413: betreffende Straße den Besitzer
[7D2]  414: wechseln soll.
[Ø4Ø]  415: @
[UDØ]  416: #46
[PE3]  417: _Hypothekenhinweis
[ØØØ]  418:
[RK5]  419: Es wurde eine Straße verkauft.
[HØ5]  42Ø: Da sie hypothekarisch belastet
[V15]  421: ist, müssen zusätzlich zum
[ØA4]  422: Verkaufspreis 1Ø% Hypothe-
[784]  423: kenzinsen bezahlt werden.
[Ø4Ø]  424: @
```

Listing 10. »Monhilfe.txt« enthält alle wichtigen Hilfetexte.

Das Suchspiel

Börsenfieber

Haben Sie nicht auch schon mal Lust gehabt, an einer Börse zu spekulieren? Jetzt können Sie anhand des Spiels »boerse.exe« feststellen, ob Sie beispielsweise an der Frankfurter Börse Chancen haben, »das große Geld« zu verdienen.

Nach dem Start von »boerse.exe« (das Listing präsentiert die Pas-Datei) erscheint zuerst das Titelbild, das durch Drücken der linken Maustaste wieder verschwindet. Anschließend werden bis zu vier Spieler aufgefordert, ihre Namen einzugeben. Ist der benötigte Drucker bereit, so gibt er nach erneutem Mausklick die Kurskarten eines jeden Spielers aus. anach erscheint das Spielfeld. Am oberen Bildschirmrand sind sieben verschiedene Aktiensorten in unterschiedlichen Farben dargestellt (Bild 1). Jeder Mitspieler erhält bei Spielbeginn ein Grundkapital von 2000 Mark, das er durch Spekulation, Ankauf und Verkauf von Aktien sowie ein kleines Quäntchen Glück vervielfachen kann. Dabei muß er natürlich seine Mitaktionäre im Auge behalten und ihnen finanziellen Schaden zufügen, wo immer es geht.
In der ersten Spielrunde ist der Ankauf neuer Aktien oder das Aktivieren einer Kurskarte erlaubt.

- **An- und Verkauf:** Suchen Sie das Feld »Ankauf« der gewünschten Aktie mit dem Mauszeiger auf und klicken Sie es an. Am Bildschirm links unten erscheint der Ankaufbalken, eingeteilt in 1 bis 20 (Stück), wobei das zur Verfügung stehende Bargeld die zu kaufende Höchstmenge begrenzt. Ist der Ankauf von weniger als 20 Aktien möglich, so wird dies durch die Darstellung in schwarzer Schrift dokumentiert, um lange Kopfrechenarbeit zu ersparen. Klicken Sie die gewünschte Aktienanzahl mit der linken Maustaste an, verschwindet der Ankaufbalken, und alle Werte werden auf den aktuellen Stand gebracht. Das Bargeld verringert sich, die Aktienanzahl und der Gesamtwert der Aktien steigen. In jeder Runde dürfen Sie allerdings nur ein einziges Mal eine bestimmte Aktie ansprechen (Ankauf oder direkter Verkauf). Eine andere Aktiensorte können Sie hingegen problemlos verschieben.
Schon ab der zweiten Spielrunde können Sie bereits erworbene Aktien wieder verkaufen. Dies geschieht analog zum Ankauf.

- **Kurskarte:** Die 15 Kurskarten jedes einzelnen Spielers werden durch Zufall ermittelt. Eine jede bewirkt eine wertmäßige Veränderung der angegebenen drei Aktien um den jeweils genannten Betrag. Die Veränderung be-



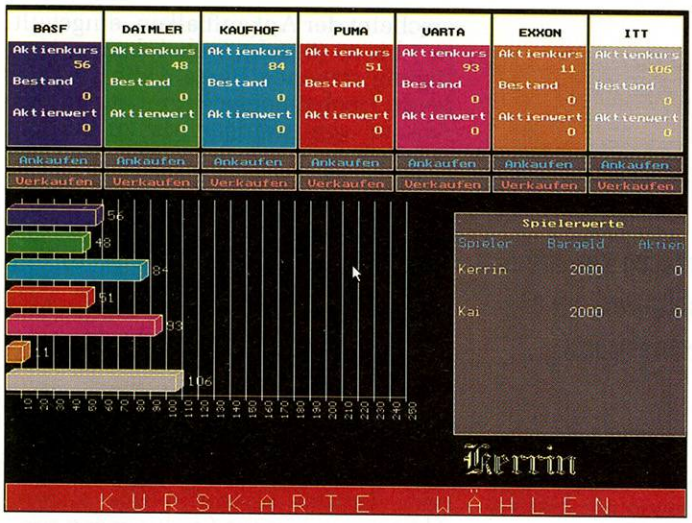


Bild 1. Der gelungene Bildschirmaufbau des Spiels »boerse.exe«

einflußt somit das Aktienvermögen aller Aktionäre der entsprechenden Spielrunde.

Um eine Kurskarte zu setzen, müssen Sie den am unteren Bildschirmrand liegenden Kurskartenbalken mit der linken Maustaste anklicken. Es erscheint ein Nummernbalken, der sämtliche Kurskarten darstellt: Bereits genommene erscheinen in schwarzer Schrift und können nicht mehr aktiviert werden. Zum Aktivieren klicken Sie die entsprechende Zahl an (linke Maustaste). Daraufhin werden die einzelnen Aktien- und Spielerwerte aktualisiert, das Balkendiagramm neu aufgebaut, und der nachfolgende Spieler kommt an die Reihe – sofern kein Börsenkrach dazwischenfunkt.

Vorsicht Börsenkrach – ein gefährliches Spiel

● **Börsenkrach:** Ein- bis zweimal pro Spiel kündigt ein Pfeifton einen unmittelbar bevorstehenden Börsenkrach an. Der Computer setzt dann die Aktienkurse nach oben oder nach unten. Am Bildschirm erscheint zuerst die aufregende Meldung und sogleich ein Balkendiagramm, das die Kursabweichungen einzeln darstellt (Bild 2):

– Von der Mitte aus nach links flüchtende Balken bedeuten, daß der Aktienkurs gefallen ist; der genaue Betrag steht neben dem Balken.

– Von der Mitte aus nach rechts flüchtende Balken bedeuten, der Aktienkurs ist gestiegen. Auch hier wird der Betrag eingeblendet.

Alle Spieler, die Aktienbesitzer sind, werden von diesem Börsenkrach betroffen und erfahren eine Auf- oder Abwertung ihrer Aktienpakete. Liegt der neue Aktienkurs infolge einer Aufwertung über 250 Mark, wird der übersteigende Betrag pro Aktie als Bargeld Ihrem Spielerkonto gutgeschrieben. Sollte jedoch durch eine Abwertung der Aktienwert negativ werden, so müssen Sie den Minusbetrag je Aktie durch Ihr Bargeld ausgleichen. Ist es aufgebraucht, und der PC stellt immer noch Forderungen an Sie, so wird das entsprechende Aktienpaket eingezogen, und Sie sind nicht mehr Aktionär dieser Aktie. Durch Anklicken der linken Maustaste wird der Bildschirm neu aufgebaut, und das Spiel kann weitergehen.

● **Spielende:** Hat jeder Spieler seine 15 Kurskarten gesetzt, oder hat ein Mitspieler während einer Spielrunde sowohl sein Bargeld als auch sein Aktienkapital verloren, so ist das Spiel beendet. Es erscheint eine grafische Gegenüberstellung, die den Kapitalstand eines jeden Spielers widerspiegelt. Per Mausklick (linke Taste) werden die besten Spieler in einen Highscore aufgenommen, der abhängig von der Mitspielerzahl angelegt ist.

● **Programmstruktur:** »Boerse.exe« wurde in Turbo Pascal geschrieben, wobei besonders die Grafikfähigkeiten dieser Sprache zum Ausdruck kommen. Die Schriftdateien sind fest in das Programm eingebunden. Das setzt jedoch voraus, daß die benötigten Dateien mit der Erweiterung »obj« vorliegen: Dies läßt sich leicht realisieren, wenn Sie »binobj.exe« verwenden. Mit dem Aufruf

```
binobj Quelldatei.bin Zieldatei.obj Prozedurname
```

lassen sich die Bgi-Datei und die Chr-Dateien, die ein binäres Format haben, passend konvertieren. Mit dem Compiler-Befehl »{$L}« werden die Obj-Dateien als Prozeduren aufgenommen. Das Einbinden des Bildschirmtreibers erfolgt zum Beispiel in folgender Form:

```
procedure EGAVGADriver; external;
{$L egavga}
procedure Installation;
begin
ft:=registerbgidriver(@EGAVGADriver);
end;
```

Verwenden Sie den zuständigen Interrupt 51hex, um die Maus einsetzen zu können. Der ständige Wechsel von »Maus_ausschalten« und »Maus_anschalten« dient als Vorsorge, um die immer wieder neu aufzubauende Grafik nicht zu zerstören (was bei Programmen im Handel leider hin und wieder der Fall ist). Der Quellcode enthält eindeutige Variablennamen und ist dadurch sehr übersichtlich und nachvollziehbar.

Das Hauptprogramm führt wichtige Installationen aus wie den Aufruf der Prozedur »Grafikinstallation«, welche die Treiber für EGA- und VGA-Karten sowie die benötigten Zeichensätze einbindet. Es folgen die Initialisierung der Anfangswerte und die Darstellung des Titelbildes. Sie werden dann aufgefordert, Ihren Namen einzugeben, und wählen Ihre Kurskarten, die sogleich ausgedruckt werden. In einer Repeat-Schleife erwartet das Programm für die Variable »Beenden« den Wert »True«, woraufhin Sie mit der linken Maustaste das Spiel beenden. Dies ist jedesmal nach einem »Börsenkrach« der Fall.

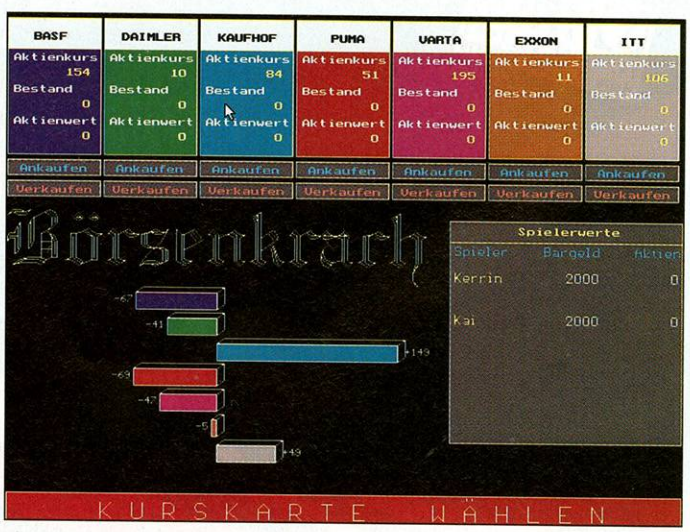


Bild 2. Der Börsenkrach kann Ihnen einen dicken Strich durch die Rechnung machen.

Die Anfangswerte der Aktien, die Kurskartenwerte und der Zeitpunkt des Börsenkrachs werden durch einen Zufallsgenerator bestimmt, um das Spiel so interessant wie nur möglich zu gestalten und eine Berechenbarkeit auszuschließen. Dadurch behält das Spiel seinen Reiz, selbst wenn Sie es immer wieder spielen. Wie durch die Ablaufbeschreibung deutlich geworden ist, benötigen Sie unbedingt eine Maus und einen Drucker.

Auf dem Drucker werden zu Beginn eines jeden Spiels die notwendigen Kurskarten ausgedruckt.

(Joachim Ofenstein/uk)

```
[ZM3]    1: (Programm: boerse.pas
[HX4]    2:  Funktion: Nachgestaltung der
[4V4]    3:            Börsentätigkeit als Spiel
[G64]    4:  Sprache:  Turbo Pascal 6.0
[AG4]    5:  Autor:    Joachim Ofenstein
[9I5]    6:  (c) 1993 DMV Widuch GmbH & Co. KG)
[0000]   7:
[1X3]    8: program Boersenfieber;
[RX4]    9: uses Printer, Graph, Dos, Crt;
[0000]  10:
[RN0]   11: var
[A33]   12:   LosWert, LosVorz,
[T76]   13:   LosAktie: Array[1..15,1..4,1..3] of Byte;
[WL5]   14:   Akt_Zahl: Array[1..7, 1..4] of Word;
[7H5]   15:   Karte: Array[1..15,1..4] of Boolean;
[S06]   16:   ScoreName: Array[1..14] of String[8];
[EV5]   17:   ScoreWert: Array[1..14] of LongInt;
[ZS6]   18:   ScoreDatum: Array[1..14] of String[10];
[685]   19:   Aktie: Array[1..7] of String[8];
[TP4]   20:   Kurs: Array[1..7] of Integer;
[RH6]   21:   Kr_Wert, Kr_Vorz : Array[1..7] of Byte;
[UY5]   22:   Spieler: Array[1..4] of String[8];
[2C3]   23:   Bonitaet, Akt_Wert,
[SE5]   24:   Bargeld: Array[1..4] of LongInt;
[K74]   25:   Krach: Array[1..2] of Byte;
[E94]   26:   HDatum, Datum: String[10];
[AV4]   27:   StrDummy, HName: String[8];
[2U2]   28:   JahrS: String[4];
[444]   29:   TagS, MonatS: String[2];
[A95]   30:   Punkte, LongIntDummy: LongInt;
[TK6]   31:   Day,Tag,Monat,Jahr,x_pos,y_pos: Word;
[7U3]   32:   ft, gd, gm, x, y: Integer;
[452]   33:   faktor: Real;
[YW6]   34:   Druckbereit, Beenden, Sortiert: Boolean;
[HK3]   35:   Register: Registers;
[P04]   36:   Bearbeitet: Set of Byte;
[H02]   37:   Datei: Text;
[935]   38:   Durchgang, Menge, AktNr, Runde,
[XU6]   39:   Person, Akt_Zeiger, ByteDummy, Moeglich,
[5U6]   40:   Sp_Anzahl, Player , KursKarte, Los: Byte;
[0000]  41:
[1S4]   42: procedure Maus_anschalten;
[MU0]   43: begin
[KM2]   44:   Register.ax:=1;
[0Z2]   45:   Intr(51,Register);
[ZN0]   46: end;
[T15]   47: procedure Maus_ausschalten;
[MU0]   48: begin
[NM2]   49:   Register.ax:=2;
[0Z2]   50:   Intr(51,Register);
[ZN0]   51: end;
[ZY4]   52: procedure Mausinstallation;
[MU0]   53: begin
[FM2]   54:   Register.ax:=0;
[0Z2]   55:   Intr(51,Register);
[8X2]   56:   Maus_anschalten;
[ZN0]   57: end;
[HL4]   58: function Maus_Taste:Byte;
[MU0]   59: begin
[SM2]   60:   Register.ax:=3;
[0Z2]   61:   Intr(51,Register);
[EA4]   62:   Maus_Taste:=Register.bx;
[ZN0]   63: end;
[E95]   64: function MausPos_x:Integer;
[MU0]   65: begin
[SM2]   66:   Register.ax:=3;
[0Z2]   67:   Intr(51,Register);
[X84]   68:   MausPos_x:=Register.cx;
[ZN0]   69: end;
[H05]   70: function MausPos_y:Integer;
[MU0]   71: begin
[SM5]   72:   Register.ax:=3; Intr(51,Register);
[194]   73:   MausPos_y:=Register.dx;
[ZN0]   74: end;
[DE4]   75: procedure setze_MausPos;
[MU0]   76: begin
[B96]   77:   Register.ax:=4; Register.cx:=x_pos;
[PL6]   78:   Register.dx:=y_pos; Intr(51,Register);
[ZN0]   79: end;
[PT5]   80: procedure EGAVGADriver; External;
[G52]   81: {$L egavga}
[C35]   82: procedure GothFont; External;
[Z81]   83: {$L goth}
[Y45]   84: procedure LittFont; External;
[3P1]   85: {$L litt}
[255]   86: procedure Grafikinstallation;
[MU0]   87: begin
[WF6]   88:   ft:=registerbgidriver(@EGAVGADriver);
[GH5]   89:   ft:=registerbgifont(@GothFont);
[ZH5]   90:   ft:=registerbgifont(@LittFont);
[TY1]   91:   gd:=Detect;
[333]   92:   InitGraph(gd,gm,'');
[RY1]   93:   SetColor(0);
[ZN0]   94: end;
[0B4]   95: procedure Spielerabfrage;
[MU0]   96: begin
[RT5]   97:   Maus_ausschalten; ClearViewPort;
[2A4]   98:   SetColor(15); y:=15;  x:=222;
[992]   99:   Sp_Anzahl:=0;
[5C4]  100:   for Person:=1 to 4 do begin
[FG3]  101:     Spieler[Person]:='';
[ET3]  102:     Str(Person,StrDummy);
[CS2]  103:     OutTextxy(13,x,
[636]  104:       'Spielernamen eingeben:');
[US3]  105:     Write(#7);  Gotoxy(43,y);
[RN6]  106:     TextColor(14); ReadLn(Spieler[Person]);
```

```
[8S3]  107:     Spieler[Person][1]:=
[KZ4]  108:       UpCase(Spieler[Person][1]);
[SR2]  109:     Inc(Sp_Anzahl);
[9G5]  110:     if Spieler[Person] = '' then begin
[3E4]  111:       Dec(Sp_Anzahl); Person:=4;
[ZN0]  112:     end;
[AP2]  113:     Inc(y); Inc(x,16);
[ZN0]  114:   end;
[8X2]  115:   Maus_anschalten;
[ZN0]  116: end;
[F44]  117: procedure Anfangswerte;
[MU0]  118: begin
[1T1]  119:   Randomize;
[9Z2]  120:   Aktie[1]:='  BASF';
[BD3]  121:   Aktie[2]:=' DAIMLER';
[CE3]  122:   Aktie[3]:=' KAUFHOF';
[M03]  123:   Aktie[4]:='   PUMA';
[943]  124:   Aktie[5]:='  VARTA';
[T63]  125:   Aktie[6]:='  EXXON';
[SY2]  126:   Aktie[7]:='   ITT';
[3E3]  127:   Druckbereit:=False;
[VX2]  128:   Beenden    :=False;
[7B2]  129:   Durchgang  :=0;
[M22]  130:   Player     :=1;
[E14]  131:   for Akt_Zeiger:=1 to 7 do
[XG5]  132:     Kurs[Akt_Zeiger]:=Random(241)+10;
[5C4]  133:   for Person:=1 to 4 do begin
[MY3]  134:     Bargeld[Person] :=2000;
[5M3]  135:     Akt_Wert[Person]:=0;
[SZ3]  136:     Bonitaet[Person]:=2000;
[E14]  137:     for Akt_Zeiger:=1 to 7 do
[YS5]  138:       Akt_Zahl[Akt_Zeiger,Person]:=0;
[ZN0]  139:   end;
[R93]  140:   for Person:=1 to 4 do
[C03]  141:     for Los:=1 to 15 do
[7C4]  142:       Karte[Los,Person]:=True;
[ZN0]  143: end;
[075]  144: procedure Kurskarten_waehlen;
[MU0]  145: begin
[C03]  146:   for Los:=1 to 15 do
[A06]  147:     for Person:=1 to Sp_Anzahl do begin
[SP2]  148:       Bearbeitet:=[];
[C15]  149:       for ByteDummy:=1 to 3 do begin
[P31]  150:         repeat
[NZ5]  151:           LosAktie[Los,Person,ByteDummy]:=
[HS1]  152:                                 Random(7)+1;
[N34]  153:         until not (LosAktie[Los,
[WN5]  154:           Person,ByteDummy] in Bearbeitet);
[5Y3]  155:         Bearbeitet:=Bearbeitet+
[E66]  156:           [LosAktie[Los,Person,ByteDummy]];
[ZV5]  157:         LosVorz[Los,Person,ByteDummy]:=
[XH1]  158:                                 Random(2);
[MU5]  159:         LosWert[Los,Person,ByteDummy]:=
[912]  160:                                 Random(126)+5;
[ZN0]  161:       end;
[ZN0]  162:     end;
[SP2]  163:   Bearbeitet:=[];
[ZN0]  164: end;
[YH3]  165: procedure Titelbild;
[MU0]  166: begin
[786]  167:   Maus_ausschalten; SetFillStyle(1,12);
[NV3]  168:   Bar3D(50,50,590,430,0,True);
[PZ2]  169:   SetFillStyle(1,4);
[NW3]  170:   Bar3D(90,90,550,390,0,True);
[RY1]  171:   SetColor(0);
[0C5]  172:   Line(50,50,90,90); Line(590,430,550,390);
[RC5]  173:   Line(590,50,550,90); Line(50,430,90,390);
[N83]  174:   SetTextStyle(2,0,6);
[5V2]  175:   OutTextxy(125,200,
[A17]  176:  'Das Spiel um Aktien,Spekulation u. Geld');
[K83]  177:   SetTextStyle(2,0,5);
[AAA]  178:   OutTextxy(170,370,'Drücke die linke Maus-T
                aste für den Start');
[A32]  179:   SetColor(14);
[193]  180:   SetTextStyle(4,0,8);
[LM5]  181:   OutTextxy(115,100,'Börsenfieber');
[232]  182:   SetColor(12);
[K83]  183:   SetTextStyle(2,0,5);
[F66]  184:   OutTextxy(250,260,'Joachim Ofenstein');
[465]  185:   SetColor(11); Maus_anschalten;
[P31]  186:   repeat
[355]  187:     Sound(Random(1000)+800); Delay(30);
[VH3]  188:   until Maus_Taste = 1;
[3G1]  189:   NoSound;
[ZN0]  190: end;
[7H3]  191: procedure Spielfeld;
[MU0]  192: begin
[RT5]  193:   Maus_ausschalten; ClearViewPort;
[083]  194:   SetTextStyle(0,0,1);
[2F5]  195:   Krach[1]:=Random(15)*Sp_Anzahl+1;
[WR3]  196:   Krach[2]:=Random(45)+1;
[P45]  197:   SetColor(14); SetFillStyle(1,8);
[X84]  198:   Bar3D(420,190,639,400,0,False);
[UW2]  199:   Line(420,210,639,210);
[HK5]  200:   OutTextxy(485,196,'Spielerwerte');
[R03]  201:   x:=1; SetColor(14);
[W35]  202:   for Akt_Zeiger:=1 to 7 do begin
[163]  203:     SetFillStyle(1,15);
[D24]  204:     Bar3D(x,1,x+88,30,0,False);
[TT4]  205:     SetFillStyle(1,Akt_Zeiger);
[Y54]  206:     Bar3D(x,30,x+88,130,0,False);
[UZ2]  207:     SetFillStyle(1,8);
[TA4]  208:     Bar3D(x,135,x+88,150,0,False);
[UZ2]  209:     SetFillStyle(1,8);
[7B4]  210:     Bar3D(x,155,x+88,170,0,False);
[WF1]  211:     Inc(x,91);
[ZN0]  212:   end;
[MW0]  213:   x:=10;
[W35]  214:   for Akt_Zeiger:=1 to 7 do begin
[RY1]  215:     SetColor(0);
[526]  216:     OutTextxy(x,15,Aktie[Akt_Zeiger]);
[E32]  217:     SetColor(15);
[385]  218:     OutTextxy(x-3,35,'Aktienkurs');
[LG4]  219:     OutTextxy(x-3,65,'Bestand');
[485]  220:     OutTextxy(x-3,95,'Aktienwert');
[Y22]  221:     SetColor(11);
[6V4]  222:     OutTextxy(x+3,140,'Ankaufen');
[232]  223:     SetColor(12);
[TV4]  224:     OutTextxy(x,160,'Verkaufen');
[WF1]  225:     Inc(x,91);
[ZN0]  226:   end;
[A32]  227:   SetColor(14);
[PZ2]  228:   SetFillStyle(1,4);
[V83]  229:   SetTextStyle(2,0,8);
[S24]  230:   Bar3D(1,450,639,479,0,False);
[BS2]  231:   OutTextxy(90,452,
[WM4]  232:     'K U R S K A R T E   W Ä H L E N');
[8X2]  233:   Maus_anschalten;
[ZN0]  234: end;
[E74]  235: procedure Balkenanzeige;
[MU0]  236: begin
[T55]  237:   Maus_ausschalten; SetColor(0);
[NE5]  238:   SetFillStyle(1,0); Bar(1,171,419,420);
[4W4]  239:   SetColor(15); Line(1,365,375,365);
[G83]  240:   SetTextStyle(2,1,4);
[ZZ2]  241:   for x:=15 to 375 do
[3T3]  242:     if x MOD 15 = 0 then begin
[E32]  243:       SetColor(15);
[WP4]  244:       Str(Trunc(x/1.5),StrDummy);
[MM4]  245:       OutTextxy(x,368,StrDummy);
[NS4]  246:       SetColor(7); Line(x,180,x,369);
[ZN0]  247:     end;
[L31]  248:   y:=190;
[W35]  249:   for Akt_Zeiger:=1 to 7 do begin
[A32]  250:     SetColor(14);
[TT4]  251:     SetFillStyle(1,Akt_Zeiger);
```

```
[GA6]  252:     Bar3D(1,y,Trunc(Kurs[Akt_Zeiger]*1.5),
[3W1]  253:                          y+15,7,True);
[ZV2]  254:     FloodFill(5,y-2,14);
[UH6]  255:     FloodFill(Trunc(Kurs[Akt_Zeiger]*1.5)+
[WA1]  256:                          2,y+3,14);
[UV5]  257:     Str(Kurs[Akt_Zeiger],StrDummy);
[2E5]  258:     SetColor(15); SetTextStyle(2,0,5);
[DM6]  259:     OutTextxy(Trunc(Kurs[Akt_Zeiger]*1.5+
[8U2]  260:                          12),y-3,StrDummy);
[7F1]  261:     Inc(y,26);
[ZN0]  262:   end;
[8X2]  263:   Maus_anschalten;
[ZN0]  264: end;
[7W5]  265: procedure Spielerwerte_anzeigen;
[MU0]  266: begin
[Z34]  267:   Maus_ausschalten; x:=80;
[ZD5]  268:   SetColor(14); SetTextStyle(3,0,5);
[RD3]  269:   SetTextJustify(2,2);
[W35]  270:   for Akt_Zeiger:=1 to 7 do begin
[TT4]  271:     SetFillStyle(1,Akt_Zeiger);
[CS2]  272:     Bar(x-75,45,x+5,55);
[VS2]  273:     Bar(x-75,75,x+5,85);
[QJ3]  274:     Bar(x-75,105,x+5,115);
[UV5]  275:     Str(Kurs[Akt_Zeiger],StrDummy);
[EE4]  276:     OutTextxy(x,49,StrDummy);
[VZ5]  277:     Str(Akt_Zahl[Akt_Zeiger,Player],
[FU1]  278:                          StrDummy);
[ME4]  279:     OutTextxy(x,79,StrDummy);
[716]  280:     Str((Akt_Zahl[Akt_Zeiger,Player]*
[Z25]  281:          Kurs[Akt_Zeiger]),StrDummy);
[2M4]  282:     OutTextxy(x,109,StrDummy);
[WF1]  283:     Inc(x,91);
[ZN0]  284:   end;
[0T6]  285:   SetTextJustify(0,2); SetTextStyle(2,0,5);
[LN5]  286:   SetFillStyle(1,8); Bar(422,233,638,398);
[Y22]  287:   SetColor(11);
[KV2]  288:   OutTextxy(425,210,
[4N4]  289:     'Spieler    Bargeld    Aktien');
[R31]  290:   y:=235;
[A06]  291:   for Person:=1 to Sp_Anzahl do begin
[A32]  292:     SetColor(14);
[ND3]  293:     SetTextJustify(0,2);
[LY5]  294:     OutTextxy(425,y,Spieler[Person]);
[E32]  295:     SetColor(15);
[2K5]  296:     Str(Bargeld[Person],StrDummy);
[RD3]  297:     SetTextJustify(2,2);
[PL4]  298:     OutTextxy(562,y,StrDummy);
[5M3]  299:     Akt_Wert[Person]:=0;
[E14]  300:     for Akt_Zeiger:=1 to 7 do
[7V3]  301:       Inc(Akt_Wert[Person],
[S85]  302:         Akt_Zahl[Akt_Zeiger,Person]*
[RA3]  303:         Kurs[Akt_Zeiger]);
[5W5]  304:     Str(Akt_Wert[Person],StrDummy);
[1M4]  305:     OutTextxy(637,y,StrDummy);
[FN6]  306:     Bonitaet[Person]:=Akt_Wert[Person]+
[GZ2]  307:                          Bargeld[Person];
[9F1]  308:     Inc(y,40);
[ZN0]  309:   end;
[ND3]  310:   SetTextJustify(0,2);
[8X2]  311:   Maus_anschalten;
[ZN0]  312: end;
[000]  313:
[AC5]  314: procedure Anzahlleiste_Aktien;
[MU0]  315: begin
[6C5]  316:   Maus_ausschalten; SetColor(14);
[VZ2]  317:   SetFillStyle(1,9);
[F83]  318:   SetTextStyle(2,0,4);
[414]  319:   Bar3D(1,420,378,440,0,False);
[TR0]  320:   x:=1;
[H65]  321:   for ByteDummy:=1 to 21 do begin
[ZG2]  322:     Line(x,420,x,440);
[3F1]  323:     Inc(x,18);
[HY4]  324:     if ByteDummy-1 = Moeglich then
[5V4]  325:       SetColor(0) else SetColor(14);
[PN4]  326:     Str(ByteDummy-1,StrDummy);
[4X4]  327:     OutTextxy(x-15,425,StrDummy);
[ZN0]  328:   end;
[8X2]  329:   Maus_anschalten;
[ZN0]  330: end;
[8F3]  331: procedure Wieviele;
[MU0]  332: begin
[PR4]  333:   Maus_ausschalten; Menge:=0;
[8X2]  334:   Maus_anschalten;
[P31]  335:   repeat
[F73]  336:     x_pos:=MausPos_x;
[K73]  337:     y_pos:=MausPos_y;
[5T6]  338:   until (Maus_Taste=1) and (MausPos_y>405);
[2S4]  339:   Menge:=Trunc(MausPos_x / 18);
[833]  340:   Maus_ausschalten;
[624]  341:   if Menge > 20 then Menge:=0;
[KZ2]  342:   SetFillStyle(1,9);
[CK2]  343:   Bar(1,390,379,441);
[8X2]  344:   Maus_anschalten;
[ZN0]  345: end;
[TY4]  346: procedure Karte_ausfuehren;
[MU0]  347: begin
[BB6]  348:   if Karte[KursKarte,Player] then begin
[833]  349:     Maus_ausschalten;
[7S5]  350:     Karte[KursKarte,Player]:=False;
[ZN3]  351:     for Los:=1 to 3 do begin
[C37]  352:       if LosVorz[KursKarte,Player,Los]=0 then
[MU0]  353:       begin
[B85]  354:         Inc(Kurs[LosAktie[KursKarte,
[AR5]  355:           Player,Los]],LosWert[KursKarte,
[782]  356:           Player,Los]);
[4Y4]  357:         if Kurs[LosAktie[KursKarte,
[AK4]  358:           Player,Los]] > 250 then begin
[A06]  359:           for Person:=1 to Sp_Anzahl do begin
[HM3]  360:             Inc(Bargeld[Person],
[0W5]  361:               ((Kurs[LosAktie[KursKarte,Player,
[BY6]  362:     Los]]-250)*Akt_Zahl[LosAktie[KursKarte,
[6Y3]  363:                          Player,Los],Person]));
[GY5]  364:             Inc(Akt_Zahl[LosAktie[KursKarte,
[6P3]  365:                          Player,Los],Person],
[095]  366:               Akt_Zahl[LosAktie[KursKarte,
[TT3]  367:                          Player,Los],Person]);
[ZN0]  368:           end;
[VK4]  369:           Kurs[LosAktie[KursKarte,
[NM4]  370:             Player,Los]]:=Random(51)+100;
[ZN0]  371:         end;
[ZN0]  372:       end;
[F37]  373:       if BysVorz[KursKarte,Player,Los]=1 then
[MU0]  374:       begin
[3G7]  375:     Dec(Kurs[LosAktie[KursKarte,Player,Los]],
[HR5]  376:       LosWert[KursKarte,Player,Los]);
[026]  377:       if Kurs[LosAktie[KursKarte,Player,
[493]  378:       Los]] <= 0 then begin
[A06]  379:         for Person:=1 to Sp_Anzahl do begin
[HM3]  380:           Inc(Bargeld[Person],
[9N4]  381:             ((Kurs[LosAktie[KursKarte,
[Player,Los]]*Akt_Zahl[LosAktie[KursKarte,
[5K4]  383:                  Player,Los],Person])));
[MU0]  384:           if Bargeld[Person] <= 0 then
[095]  385:           begin
[A64]  386:             Akt_Zahl[LosAktie[KursKarte,
[5F3]  387:                  Player,Los],Person]:=0;
[ZN0]  388:             Bargeld[Person]:=0;
[ZN0]  389:           end;
[ZN0]  390:         end;
[VK4]  391:         Kurs[LosAktie[KursKarte,
[UU2]  392:                      Player,Los]]:=100;
[ZN0]  393:       end;
```

»Boerse.pas« ist ein interessantes Gesellschaftsspiel.

```
[ZNØ]  394:        end;
[ZNØ]  395:      end;
[GZ1]  396:      Inc(Player);
[236]  397:      if Player > Sp_Anzahl then Player:=1;
[6B5]  398:      Bearbeitet:=[]; Inc(Durchgang);
[Y47]  399:      Maus_anschalten; Spielerwerte_anzeigen;
[HD2]  4ØØ:      Balkenanzeige;
[ZNØ]  4Ø1:    end;
[ZNØ]  4Ø2: end;
[ØØØ]  4Ø3:
[B54]  4Ø4: procedure Kartenleiste;
[MUØ]  4Ø5: begin
[6C5]  4Ø6:   Maus_ausschalten; SetColor(14);
[ED6]  4Ø7:   SetFillStyle(1,9); SetTextStyle(2,Ø,6);
[C34]  4Ø8:   Bar3D(2Ø,4Ø6,621,449,Ø,False);
[LZ2]  4Ø9:   SetFillStyle(1,1);
[XY4]  41Ø:   Bar3D(25,411,616,444,Ø,False); x:=2Ø;
[ØW3]  411:   for Los:=1 to 15 do begin
[PN4]  412:     SetColor(14); Line(x,411,x,444);
[1C4]  413:     if Karte[Los,Player] then
[1V4]  414:        SetColor(14) else SetColor(Ø);
[5L4]  415:     Inc(x,4Ø); Str(Los,StrDummy);
[GX4]  416:     OutTextxy(x-26,418,StrDummy);
[ZNØ]  417:   end;
[XA5]  418:   KursKarte:=Ø; Maus_anschalten;
[P31]  419:   repeat
[F73]  42Ø:     x_pos:=MausPos_x;
[K73]  421:     y_pos:=MausPos_y;
[D24]  422:   until (Maus_Taste = 1) and
[296]  423:   ((MausPos_y>4Ø6) and (MausPos_y<449));
[B76]  424:   KursKarte:=1+Trunc((MausPos_x-2Ø)/4Ø);
[833]  425:   Maus_ausschalten;
[V33]  426:   if (KursKarte < 1) or
[5P5]  427:      (KursKarte > 15) then KursKarte:=Ø;
[XE5]  428:   SetFillStyle(1,Ø); Bar(1,4Ø5,631,449);
[Z46]  429:   Maus_anschalten; Karte_ausfuehren;
[ZNØ]  43Ø: end;
[ØØØ]  431:
[5E6]  432: procedure Mausaktivitaet_beobachten;
[MUØ]  433: begin
[NJ6]  434:   Maus_ausschalten; SetFillStyle(1,Ø);
[5T2]  435:   Bar(42Ø,4Ø1,639,449);
[2E5]  436:   SetTextStyle(4,Ø,5); SetColor(14);
[626]  437:   OutTextxy(43Ø,395,Spieler[Player]);
[376]  438:   SetTextStyle(2,Ø,5); Maus_anschalten;
[P31]  439:   repeat
[F73]  44Ø:     x_pos:=MausPos_x;
[K73]  441:     y_pos:=MausPos_y;
[VH3]  442:   until Maus_Taste = 1;
[9R3]  443:   if (MausPos_y > 135) and
[AB4]  444:      (MausPos_y < 17Ø) then begin
[G45]  445:     AktNr:=Trunc(MausPos_x / 91)+1;
[TD5]  446:     if AktNr in Bearbeitet then Exit;
[HG5]  447:     Bearbeitet:=Bearbeitet + [AktNr];
[ZNØ]  448:   end;
[9R3]  449:   if (MausPos_y > 135) and
[2B4]  45Ø:      (MausPos_y < 15Ø) then begin
[833]  451:     Maus_ausschalten;
[T54]  452:     SetColor(14); Moeglich:=Ø;
[HR5]  453:     Moeglich:=Trunc(Bargeld[Player]/
[EE2]  454:                          Kurs[AktNr]);
[DM2]  455:     OutTextxy(1,4Ø5,
[936]  456:      'Wieviel Aktien wollen Sie kaufen ?');
[8X2]  457:     Maus_anschalten;
[LR3]  458:     Anzahlleiste_Aktien;
[ZH1]  459:     Wieviele;
[NL7]  46Ø: if (Menge*Kurs[AktNr])>Bargeld[Player] then
[UD1]  461:       Menge:=Ø;
[4Y5]  462:     Inc(Akt_Zahl[AktNr,Player],Menge);
[C47]  463:   Dec(Bargeld[Player],(Kurs[AktNr]*Menge));
[Ø24]  464:     Spielerwerte_anzeigen;
[L33]  465:     Menge:=Ø; AktNr:=Ø;
[4MØ]  466:   end
[ZRØ]  467:   else
[VD3]  468:     if (MausPos_y>155) and
[Ø94]  469:        (MausPos_y<17Ø) then begin
[R55]  47Ø:     Maus_ausschalten; Moeglich:=Ø;
[316]  471:     Moeglich:=Akt_Zahl[AktNr,Player];
[DM2]  472:     OutTextxy(1,4Ø5,
[RL6]  473:  'Wieviel Aktien wollen Sie verkaufen ?');
[9L6]  474:     Maus_anschalten; Anzahlleiste_Aktien;
[ZH1]  475:     Wieviele;
[KD6]  476:     if Menge>Akt_Zahl[AktNr,Player] then
[UD1]  477:       Menge:=Ø;
[ZW5]  478:     Dec(Akt_Zahl[AktNr,Player],Menge);
[F47]  479:   Inc(Bargeld[Player],(Menge*Kurs[AktNr]));
[Ø24]  48Ø:     Spielerwerte_anzeigen;
[L33]  481:     Menge:=Ø; AktNr:=Ø;
[4MØ]  482:   end
[ZRØ]  483:   else
[SG4]  484:     if MausPos_y >45Ø then begin
[MB2]  485:       Kartenleiste;
[ZNØ]  486:     end;
[ZNØ]  487: end;
[ØØØ]  488:
[XW5]  489: procedure Kurskarten_ausdrucken;
[MUØ]  49Ø: begin
[833]  491:   Maus_ausschalten;
[P31]  492:   repeat
[DXØ]  493:     {$I-}
[Y9Ø]  494:     Write(Lst,' ');
[BXØ]  495:     {$I+}
[LC4]  496:     if IOResult <> Ø then begin
[ØE5]  497:       SetColor(14); SetTextStyle(2,Ø,7);
[SY9]  498:       OutTextxy(85,41Ø,'Drucker nicht berei
t!  Bitte einschalten!');
[1S1]  499:       Sound(25ØØ);
[4MØ]  5ØØ:     end
[T34]  5Ø1:     else Druckbereit:=True;
[Y43]  5Ø2:   until Druckbereit;
[EV4]  5Ø3:   NoSound; SetTextStyle(2,Ø,5);
[KZ2]  5Ø4:   SetFillStyle(1,Ø);
[WH2]  5Ø5:   Bar(1,41Ø,42Ø,435);
[AQ6]  5Ø6:   for Person:=1 to Sp_Anzahl do begin
[TP4]  5Ø7:     ClearViewPort; SetColor(14);
[SSF]  5Ø8:     OutTextxy(6Ø,22Ø,'Maustaste drücken, um
 die Kurskarten von '+Spieler[Person]+' zu
drucken !');
[8X2]  5Ø9:     Maus_anschalten;
[P31]  51Ø:     repeat
[YM6]  511:     until Maus_Taste=1; Maus_ausschalten;
[9S6]  512:     WriteLn(Lst,'Kurskarten von Spieler ',
[E53]  513:                          Spieler[Person]);
[P75]  514:     WriteLn(Lst,'-------------------------',
[SØ4]  515:       '-----------------------------------',
[TZ1]  516:       '-------------');
[PZ3]  517:     for Los:= 1 to 15 do begin
[EA5]  518:      Write(Lst,'Karte Nr. (',Los:2,')');
[C15]  519:      for ByteDummy:=1 to 3 do begin
[YW6]  52Ø:     if LosVorz[Los,Person,ByteDummy]=Ø then
[C42]  521:         StrDummy:='+'
[S83]  522:       else StrDummy:='-';
[FZ6]  523:     Case LosAktie[Los,Person,ByteDummy] of
[7V3]  524:       1: Write(Lst,'      BASF     ',
[SØ2]  525:          StrDummy,' ',
[BY5]  526:          LosWert[Los,Person,ByteDummy]:3);
[V64]  527:       2: Write(Lst,'     DAIMLER ',
[SØ2]  528:          StrDummy,' ',
[BY5]  529:          LosWert[Los,Person,ByteDummy]:3);
[U74]  53Ø:       3: Write(Lst,'     KAUFHOF ',
[SØ2]  531:          StrDummy,' ',
[BY5]  532:          LosWert[Los,Person,ByteDummy]:3);
[8X3]  533:       4: Write(Lst,'      PUMA     ',
[SØ2]  534:          StrDummy,' ',
[BY5]  535:          LosWert[Los,Person,ByteDummy]:3);
[3Z3]  536:       5: Write(Lst,'      VARTA    ',
[SØ2]  537:          StrDummy,' ',
[BY5]  538:          LosWert[Los,Person,ByteDummy]:3);
[N14]  539:       6: Write(Lst,'      EXXON    ',
[SØ2]  54Ø:          StrDummy,' ',
[BY5]  541:          LosWert[Los,Person,ByteDummy]:3);
[SW3]  542:       7: Write(Lst,'      ITT      ',
[SØ2]  543:          StrDummy,' ',
[BY5]  544:          LosWert[Los,Person,ByteDummy]:3);
[ZNØ]  545:       end;
[ZNØ]  546:     end;
[TM2]  547:     WriteLn(Lst,'');
[ZNØ]  548:   end;
[D44]  549:   for ByteDummy:=1 to 15 do
[ZNØ]  55Ø:     WriteLn(Lst,'');
[ZNØ]  551: end;
[8X2]  552:   Maus_anschalten;
[ZNØ]  553: end;
[ØØØ]  554:
[1S3]  555: procedure Auswertung;
[MUØ]  556: begin
[833]  557:   Maus_ausschalten;
[P31]  558:   repeat
[YU2]  559:     Sortiert:=True;
[7C5]  56Ø:     for Person:=1 to Sp_Anzahl-1 do
[4L3]  561:       if Bonitaet[Person]<
[ZØ5]  562:          Bonitaet[Person+1] then begin
[GW2]  563:         Sortiert:=False;
[SX5]  564:         LongIntDummy:=Bonitaet[Person];
[V84]  565:         HName:=Spieler[Person];
[MM6]  566:       Bonitaet[Person]:=Bonitaet[Person+1];
[ØM6]  567:         Spieler[Person]:=Spieler[Person+1];
[DØ6]  568:         Bonitaet[Person+1]:=LongIntDummy;
[XD4]  569:         Spieler[Person+1] :=HName;
[ZNØ]  57Ø:       end;
[NU2]  571:   until Sortiert;
[TP4]  572:   ClearViewPort; SetColor(14);
[PZ2]  573:   SetFillStyle(1,4);
[LT3]  574:   Bar3D(1,1,639,195,Ø,False);
[493]  575:   SetTextStyle(4,Ø,9);
[1D5]  576:   OutTextxy(8Ø,1,'Börsenfieber');
[N83]  577:   SetTextStyle(4,Ø,4);
[7Z9]  578:   OutTextxy(5Ø,115,'Wochenbericht in abfall
ender Reihenfolge');
[2A5]  579:   SetColor(Ø); SetTextStyle(2,Ø,5);
[SDA]  58Ø:   OutTextxy(165,165,'Maustaste drücken, um
Ansicht zu beenden !');
[A32]  581:   SetColor(14);
[MM3]  582:   if Bonitaet[1] > Ø then
[EØ4]  583:     Faktor:=59Ø/Bonitaet[1]
[GL2]  584:   else Faktor:=1;
[431]  585:   y:=21Ø;
[AØ6]  586:   for Person:=1 to Sp_Anzahl do begin
[ND3]  587:     SetTextJustify(Ø,2);
[D14]  588:     SetFillStyle(1,Person);
[WV5]  589:     OutTextxy(1,y-5,Spieler[Person]);
[NT1]  59Ø:     Bar3D(1,y+2Ø,
[NP5]  591:       (Round(Bonitaet[Person]*faktor)),
[422]  592:       y+4Ø,1Ø,True);
[ZS5]  593:     Str(Bonitaet[Person],StrDummy);
[RD3]  594:     SetTextJustify(2,2);
[PT4]  595:     OutTextxy(15Ø,y-5,StrDummy);
[HF1]  596:     Inc(y,6Ø);
[ZNØ]  597:   end;
[M65]  598:   for ByteDummy:=1 to 5Ø do begin
[9H3]  599:     Sound(Random(8ØØ)+2ØØ);
[DF1]  6ØØ:     Delay(2Ø);
[ZNØ]  6Ø1:   end;
[X15]  6Ø2:   NoSound; SetTextJustify(Ø,2);
[8X2]  6Ø3:   Maus_anschalten;
[P31]  6Ø4:   repeat
[PG3]  6Ø5:   until Maus_Taste=1;
[CE2]  6Ø6:   Beenden:=True;
[ZNØ]  6Ø7: end;
[ØØØ]  6Ø8:
[WH3]  6Ø9: procedure Highscore;
[MUØ]  61Ø: begin
[PL4]  611:   GetDate(Jahr,Monat,Tag,day);
[CD4]  612:   Str(Sp_Anzahl,StrDummy);
[8F6]  613:   Assign(Datei,'HSCORE'+StrDummy+'.BSF');
[DXØ]  614:   {$I-}
[192]  615:   Append(Datei);
[BXØ]  616:   {$I+}
[TP2]  617:   Str(Jahr,JahrS);
[663]  618:   Str(Monat,MonatS);
[3N5]  619:   if Monat < 1Ø then MonatS:='Ø'+MonatS;
[2D2]  62Ø:   Str(Tag,TagS);
[9M4]  621:   if Tag < 1Ø then TagS:='Ø'+TagS;
[L45]  622:   Datum:=TagS+'.'+MonatS+'.'+JahrS;
[263]  623:   if IOResult = Ø then
[GL1]  624:     Write(#7#7)
[ZRØ]  625:   Else
[4P2]  626:     Rewrite(Datei);
[AØ6]  627:     for Person:=1 to Sp_Anzahl do begin
[MP5]  628:       WriteLn(Datei,Spieler[Person]);
[ET5]  629:       WriteLn(Datei,Bonitaet[Person]);
[BN3]  63Ø:       WriteLn(Datei,Datum);
[ZNØ]  631:     end;
[642]  632:     Close(Datei);
[CD4]  633:   Str(Sp_Anzahl,StrDummy);
[8F6]  634:   Assign(Datei,'HSCORE'+StrDummy+'.BSF');
[DXØ]  635:   {$I-}
[F52]  636:   Reset(Datei);
[BXØ]  637:   {$I+}
[MS6]  638:   for Person:=1 to 1Ø+Sp_Anzahl do begin
[6L5]  639:     ReadLn(Datei,ScoreName[Person]);
[5P5]  64Ø:     ReadLn(Datei,ScoreWert[Person]);
[7U5]  641:     ReadLn(Datei,ScoreDatum[Person]);
[ZNØ]  642:   end;
[P31]  643:   repeat
[YU2]  644:     Sortiert:=True;
[SL5]  645:     for Person:=1 to 1Ø+Sp_Anzahl-1 do
[SV3]  646:       if ScoreWert[Person] <
[LC5]  647:          ScoreWert[Person+1] then begin
[VW4]  648:         if ScoreName[Person] = '' then
[PØ5]  649:           ScoreName[Person]:='pusher';
[GW2]  65Ø:         Sortiert:=False;
[CE4]  651:         HName:=ScoreName[Person];
[ØW4]  652:         Punkte:=ScoreWert[Person];
[WØ5]  653:         HDatum:=ScoreDatum[Person];
[R77]  654:       ScoreName[Person]:=ScoreName[Person+1];
[6C7]  655:       ScoreWert[Person]:=ScoreWert[Person+1];
[KD7]  656:     ScoreDatum[Person]:=ScoreDatum[Person+1];
[ØV4]  657:         ScoreName[Person+1]:=HName;
[W85]  658:         ScoreWert[Person+1] :=Punkte;
[N35]  659:         ScoreDatum[Person+1]:=HDatum;
[ZNØ]  66Ø:       end;
[NU2]  661:   until Sortiert;
[CD4]  662:   Str(Sp_Anzahl,StrDummy);
[8F6]  663:   Assign(Datei,'HSCORE'+StrDummy+'.BSF');
[432]  664:   Erase(Datei);
[5Ø5]  665:   Rewrite(Datei); Append(Datei);
[MD4]  666:   for Person:=1 to 1Ø do begin
[225]  667:     WriteLn(Datei,ScoreName[Person]);
[Y16]  668:     WriteLn(Datei,ScoreWert[Person]);
[H56]  669:     WriteLn(Datei,ScoreDatum[Person]);
[ZNØ]  67Ø:   end;
[4H5]  671:   Close(Datei); Maus_ausschalten;
[322]  672:   ClearDevice;
[BØ3]  673:   SetFillStyle(9,1);
[D24]  674:   Bar3D(14Ø,5Ø,5ØØ,42Ø,Ø,False);
[MZ2]  675:   SetFillStyle(1,2);
[S84]  676:   Bar3D(14Ø,425,5ØØ,455,Ø,False);
[RY1]  677:   SetColor(Ø);
[MEA]  678:   OutTextxy(155,43Ø,'Maustaste drücken, um
das Spiel zu beenden !');
[E32]  679:   SetColor(15);
[LW2]  68Ø:   Line(14Ø,14Ø,5ØØ,14Ø);
[RW2]  681:   Line(14Ø,142,5ØØ,142);
[V83]  682:   SetTextStyle(4,Ø,6);
[HW4]  683:   OutTextxy(155,6Ø,'Highscore');
[F83]  684:   SetTextStyle(2,Ø,4);
[CD4]  685:   Str(Sp_Anzahl,StrDummy);
[GB6]  686:   OutTextxy(425,95,StrDummy+' Spieler');
[8F6]  687:   Assign(Datei,'HSCORE'+StrDummy+'.BSF');
[GC4]  688:   y:=16Ø; SetTextStyle(Ø,Ø,2);
[A32]  689:   SetColor(14);
[DXØ]  69Ø:   {$I-}
[F52]  691:   Reset(Datei);
[Z44]  692:   if IOResult = Ø then begin
[MD4]  693:     for Person:=1 to 1Ø do begin
[D73]  694:       ReadLn(Datei,HName);
[BH3]  695:       ReadLn(Datei,Punkte);
[RF3]  696:       ReadLn(Datei,HDatum);
[CZ3]  697:       if (HName=Spieler[1]) and
[DD4]  698:          (Punkte=Bonitaet[1]) then
[HV4]  699:          SetColor(4) Else SetColor(14);
[383]  7ØØ:       SetTextStyle(Ø,Ø,2);
[ND3]  7Ø1:       SetTextJustify(Ø,2);
[SY3]  7Ø2:       OutTextxy(155,y,HName);
[K83]  7Ø3:       SetTextStyle(2,Ø,5);
[E24]  7Ø4:       OutTextxy(298,y,HDatum);
[383]  7Ø5:       SetTExtStyle(Ø,Ø,2);
[RD3]  7Ø6:       SetTextJustify(2,2);
[NT3]  7Ø7:       Str(Punkte,StrDummy);
[2Ø6]  7Ø8:       OutTextxy(485,y,StrDummy); Inc(y,25);
[ZNØ]  7Ø9:     end;
[ZNØ]  71Ø:   end;
[ØØØ]  711:   Close(Datei); Maus_anschalten;
[P31]  712:   repeat
[VH3]  713:   until Maus_Taste = 1;
[RT5]  714:   Maus_ausschalten; ClearViewPort;
[522]  715:   CloseGraph;
[ZNØ]  716: end;
[ØØØ]  717:
[L54]  718: procedure Boersenkrach;
[MUØ]  719: begin
[833]  72Ø:   Maus_ausschalten;
[W35]  721:   for Akt_Zeiger:=1 to 7 do begin
[B96]  722:     Kr_Wert[Akt_Zeiger]:=Random(146)+5;
[1U5]  723:     Kr_Vorz[Akt_Zeiger]:=Random(2);
[ZNØ]  724:   end;
[3P3]  725:   Delay(1ØØØ); SetColor(Ø);
[NE5]  726:   SetFillStyle(1,Ø); Bar(1,171,419,42Ø);
[3E5]  727:   SetColor(14); SetTextStyle(4,Ø,8);
[8E5]  728:   OutTextxy(1,155,'Börsenkrach');
[S16]  729:   for ByteDummy:=2ØØ downto 6Ø do begin
[F24]  73Ø:     Sound(ByteDummy*1Ø); Delay(35);
[ZNØ]  731:   end;
[EN3]  732:   NoSound; SetColor(15);
[F83]  733:   SetTextStyle(2,Ø,4);
[E72]  734:   x:=2ØØ; y:=26Ø;
[W35]  735:   for Akt_Zeiger:=1 to 7 do begin
[EC6]  736:     if Kr_Vorz[Akt_Zeiger]=Ø then begin
[ND3]  737:       SetTextJustify(Ø,2);
[TT4]  738:       SetFillStyle(1,Akt_Zeiger);
[FØ2]  739:       Bar3D(x,y,x+
[7H5]  74Ø:        Trunc(Kr_Wert[Akt_Zeiger]*1.15),
[4W1]  741:        y+16,7,True);
[BG6]  742:       Str(Kr_Wert[Akt_Zeiger],StrDummy);
[M47]  743:       OutTextxy(x+Trunc(Kr_Wert[Akt_Zeiger]*
[1E3]  744:        1.15)+7,y,'+'+StrDummy);
[5F1]  745:       Inc(y,24);
[4MØ]  746:     end
[ZRØ]  747:     else
[EA6]  748:       if Kr_Vorz[Akt_Zeiger] = 1 then begin
[RD3]  749:       SetTextJustify(2,2);
[TT4]  75Ø:       SetFillStyle(1,Akt_Zeiger);
[LS6]  751:       Bar3D(x-Trunc(Kr_Wert[Akt_Zeiger]*
[VE3]  752:        1.15),y,x,y+16,7,True);
[BG6]  753:       Str(Kr_Wert[Akt_Zeiger],StrDummy);
[K92]  754:       OutTextxy(x-
[GS5]  755:         Trunc(Kr_Wert[Akt_Zeiger]*1.15)-2,
[3G2]  756:            y,'-'+StrDummy);
[5F1]  757:       Inc(y,24);
[ZNØ]  758:     end;
[ND3]  759:     SetTextJustify(Ø,2);
[BA6]  76Ø:     if Kr_Vorz[Akt_Zeiger] = Ø then begin
[WU3]  761:       Inc(Kurs[Akt_Zeiger],
[XØ4]  762:           Kr_Wert[Akt_Zeiger]);
[2Y5]  763:       if Kurs[Akt_Zeiger] > 25Ø then begin
[AQ6]  764:         for Person:=1 to Sp_Anzahl do begin
[HM3]  765:           Inc(Bargeld[Person],
[A14]  766:            ((Kurs[Akt_Zeiger]-25Ø)*
[UG5]  767:              Akt_Zahl[Akt_Zeiger,Person]));
[AY5]  768:           Inc(Akt_Zahl[Akt_Zeiger,Person],
[DC5]  769:              Akt_Zahl[Akt_Zeiger,Person]);
[ZNØ]  77Ø:         end;
[EH5]  771:         Kurs[Akt_Zeiger]:=Random(31)+14Ø;
[ZNØ]  772:       end;
[ZNØ]  773:     end;
[EA6]  774:     if Kr_Vorz[Akt_Zeiger] = 1 then begin
[RT3]  775:       Dec(Kurs[Akt_Zeiger],
[XØ4]  776:                    Kr_Wert[Akt_Zeiger]);
[MZ5]  777:       if Kurs[Akt_Zeiger] <= Ø then begin
[AQ6]  778:         for Person:=1 to Sp_Anzahl do begin
[HM3]  779:           Inc(Bargeld[Person],
[853]  78Ø:           Kurs[Akt_Zeiger]*
[DC5]  781:           Akt_Zahl[Akt_Zeiger,Person]);
[TB5]  782:         if Bargeld[Person]<=Ø then begin
[YS5]  783:           Akt_Zahl[Akt_Zeiger,Person]:=Ø;
[5F3]  784:           Bargeld[Person]:=Ø;
[ZNØ]  785:         end;
[ZNØ]  786:       end;
[9R3]  787:       Kurs[Akt_Zeiger]:=1Ø;
[ZNØ]  788:     end;
[ZNØ]  789:   end;
[ZNØ]  79Ø:   end;
[8X2]  791:   Maus_anschalten;
[P31]  792:   repeat
[VH3]  793:   until Maus_Taste = 1;
[BF6]  794:   Balkenanzeige; Spielerwerte_anzeigen;
[GS2]  795:   Krach[Runde]:=Ø;
[ZNØ]  796: end;
[ØØØ]  797:
[MUØ]  798: begin
[7N6]  799:   Grafikinstallation; Mausinstallation;
[784]  8ØØ:   Anfangswerte; Titelbild;
[RS2]  8Ø1:   Spielerabfrage;
[9M3]  8Ø2:   Kurskarten_waehlen;
[C44]  8Ø3:   Kurskarten_ausdrucken;
[864]  8Ø4:   Spielfeld; Balkenanzeige;
[Ø24]  8Ø5:   Spielerwerte_anzeigen;
[P31]  8Ø6:   repeat
[WU4]  8Ø7:     Mausaktivitaet_beobachten;
[B75]  8Ø8:     if Durchgang>=Sp_Anzahl*14 then
[Z53]  8Ø9:       Bearbeitet:=[1..7];
[S83]  81Ø:     for Runde:=1 to 2 do
[4C5]  811:       if Durchgang = Krach[Runde] then
[Ø82]  812:         Boersenkrach;
[WX4]  813:     for Person:=1 to Sp_Anzahl do
[KM4]  814:       if (Bonitaet[Person]<=Ø) or
[N15]  815:          (Durchgang = Sp_Anzahl*15) then
[652]  816:         Auswertung;
[8C2]  817:   until Beenden;
[781]  818:   Highscore;
[KNØ]  819: end.
```

»Boerse.pas« ist ein interessantes Gesellschaftsspiel.

Das Logikspiel

Wer ist der beste Kombinierer?

Das Spiel »logikal.pas« stellt eine Trainingsvariante für das logische Denken dar. Anhand von Hinweisen führen Sie logische Verknüpfungen durch, um das richtige Ergebnis zu erhalten. Als Hilfestellung finden Sie ein Rätselgitter, in das Sie Ihre Antworten eintragen.

Nach dem Start des Denkspiels »logikal.exe« sehen Sie das Spielfeld, das sich in drei Teile aufgliedert (Bild 1). Auf der rechten Seite des Bildschirms erscheint das Rätselgitter, das für alle neun Rätsel gleich bleibt. An der rechten Bildschirmseite werden die neun verschiedenen Rätsel und die Taste »Spiel beenden« eingeblendet. Am unteren Bildschirmrand erscheinen die Hinweise zu jedem einzelnen Rätsel, wobei durch Anklicken des Feldes »Nächster Hinweis« alle Hinweise durchgeblättert werden können (Bild 2).

Anhand der gegebenen Hinweise werden durch logische Verknüpfungen Übereinstimmungen in positiver oder negativer Hinsicht festgestellt. So können zum Beispiel aus folgendem Hinweis mehrere Rückschlüsse gezogen werden.

»Peter, der keine 20 Jahre alt ist, trägt einen grünen Pullover. Der Träger des grünen Pullovers fährt einen Audi.«

Hieraus läßt sich also folgendes schließen: Peter ist nicht 20 Jahre alt, Peter trägt einen grünen Pullover, Peter fährt einen Audi.

Vergleichen Sie das interessante Rätselgitter

Durch den horizontalen und vertikalen Vergleich der im Rätselgitter angegebenen Tatbestände wird nun für Übereinstimmungen in dem entsprechenden Kästchen ein »+« (durch Drücken der linken Maustaste), bei Unstimmigkeit ein »–« (durch Drücken der rechten Maustaste) eingetragen. Sollten Sie sich geirrt und aus Versehen ein falsches Zeichen in das Kästchen eingesetzt haben, läßt sich das Zeichen durch Drücken der mittleren Maustaste entfernen. Sind alle Kästchen durch Zeichen besetzt, so nimmt der Computer automatisch die Auswertung vor und meldet sich mit dem entsprechenden Ergebnis.

Logikal wurde in Turbo Pascal 6.0 entwickelt, wobei die Grafikfähigkeit ausgenutzt und die benötigte Schriftdatei sowie der benötigte Grafiktreiber fest in das Programm eingebunden wurden. Die Steuerung der Maus wird über den zuständigen Interrupt 51hex realisiert. Die benötigten Rätselelemente, die Hinweise und die für die Auswertung benötigte Summe, die sich aus der Multiplikation der x- und y-Werte der richtigen »+« zusammensetzt, wurden als Konstanten am Programmanfang abgelegt. Der Mauszeiger wird vor jedem Grafikaufbau ab- und nach Beendigung des Aufbaus wieder eingeschaltet, um die Zerstörung der Grafik zu vermeiden. Die aktuellen Positionen »x_pos« und »y_pos« stellen Sie fest über die Programmzeilen

```
Feld[Trunc((x_pos-100-Trunc(x_pos/
   100))/20)+1
```

Die Übersichtlichkeit des Quellcodes (Listing) wird durch die Verwendung von eindeutigen Variablen, Konstanten und Einrückungen gewahrt.

Sie können dieses Spiel natürlich noch beliebig erweitern und ausbauen. Auch neue logische Verknüpfungen lassen sich leicht integrieren.

(Joachim Ofenstein/uk)

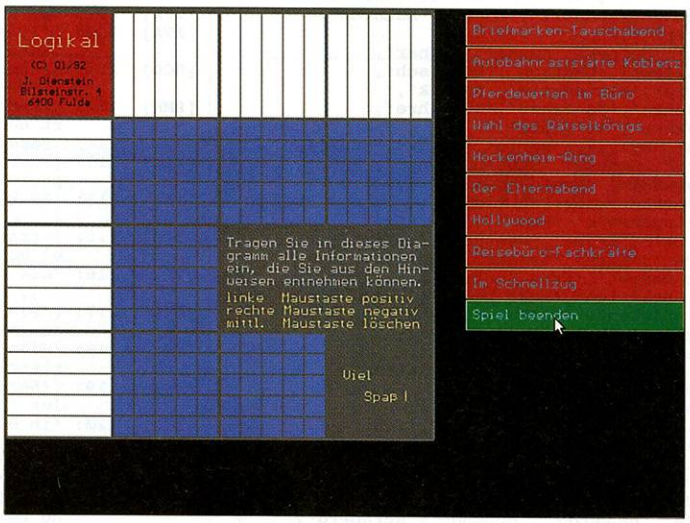


Bild 1. Der Bildschirmaufbau des Spiels »logikal.pas«

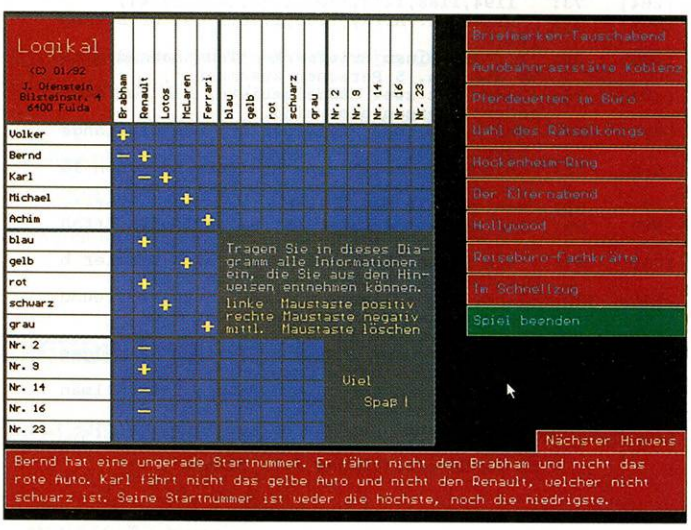


Bild 2. Zahlreiche Hinweise führen hoffentlich zum richtigen Ziel.

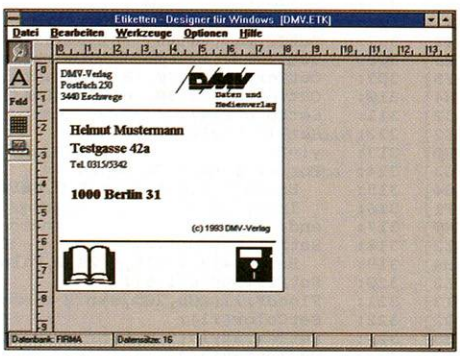
```
[HT3]     1: {Programm: logikal.pas
[7P3]     2:  Funktion: Logik-Spiel
[G64]     3:  Sprache:  Turbo Pascal 6.Ø
[AG4]     4:  Autor:    Joachim Ofenstein
[915]     5:  (c) 1993 DMV Widuch GmbH & Co. KG}
[ØØØ]     6:
[2W2]     7: program Logikal;
[W83]     8: uses Crt, Dos, Graph;
[6XØ]     9: const
[N86]    1Ø: Werte: Array[1..9,1..2Ø] of String[16] =
[CK4]    11:  (('Blanke', 'Wehner', 'Grosch',
[GM5]    12:    'Kollmann', 'Jordan', 'Deutschland',
[DY5]    13:    'Dänemark','England','Griechenland',
[FG5]    14:    'USA',' 5 Jahre','1Ø Jahre','15 Jahre',
[425]    15:    '2Ø Jahre','25 Jahre','Frankreich',
[L55]    16:    'Italien','Portugal','Schweden',
[FS1]    17:    'Schweiz'),
[SD5]    18:   ('Dirk','Roger','Reinhold','Peter',
[AB4]    19:    'Christoph','Brot','Kaffee',
[Ø36]    2Ø:    'Honigmelonen','Rüben','Zigaretten',
[A74]    21:    'Bozen','Kiel','Offenburg',
[G64]    22:    'Regensburg','Wiesbaden',
[C65]    23:    'Guthmann','Müller','Köster',
[AU3]    24:    'Schrehardt','Wiegard'),
[C25]    25:   ('Marion','Anja','Inge','Kathrin',
[VC6]    26:    'Anita','Wilder Bulle','Schwarze Rose',
[436]    27:    'Spring-Lady','Schnecke','Lahme Ente',
[KB4]    28:    'Platz 4','Platz 5','Platz 8',
[7T5]    29:    'Platz 11','Platz 12','Auszubildende',
[VX4]    3Ø:    'Buchhalterin','Empfangsdame',
[UY4]    31:    'Schreibkraft','Sekretärin'),
[AX5]    32:   ('Zenker','Krah','Müller','Schweimer',
[784]    33:    'Reuscher','Bastelgitter',
[XC6]    34:    'Kreuzworträtsel','Schwedenrätsel',
[195]    35:    'Silbenrätsel','Zahlenrätsel',
[6N4]    36:    '1. Platz','2. Platz','3. Platz',
[1P4]    37:    '4. Platz','5. Platz','Näherin',
[F95]    38:    'Sekretärin','Student','Lehrer',
[2N2]    39:    'Arbeitsloser'),
[7E6]    4Ø: ('Volker','Bernd','Karl','Michael','Achim',
[9G5]    41:    'blau','gelb','rot','schwarz','grau',
[TK5]    42: 'Nr. 2','Nr. 9','Nr. 14','Nr. 16','Nr. 23',
[DW5]    43:    'Brabham','Renault','Lotos','McLaren',
[RR1]    44:    'Ferrari'),
[XC5]    45:   ('Hörl','Hofner','Kunze','Wagner',
[M75]    46:    'Schweimer','Deutsch','Englisch',
[5Z4]    47:    'Chemie','Mathematik','Physik',
[ØH4]    48:    '33 Jahre','36 Jahre','39 Jahre',
[MZ2]    49:    '41 Jahre','44 Jahre',
[4E6]    5Ø:   'Petra','Sylvia','Mario','Peter','Rudi'),
[WA6]    51:   ('Björn','Dennis','Stefan','Paul','Knut',
[185]    52:    'Kameramann','Regie-Assistent',
[7V4]    53:    'Beleuchter','Maskenbildner',
[CK5]    54:    'Ton-Ingenieur','Komödie','Drama',
[DS5]    55:    'Musical','Krimi','Western','Berger',
[YH5]    56:    'Janßen','Gerk','Saager','Wulff'),
[ND4]    57:   ('Frau Kampen','Frau Becker',
[TL4]    58:    'Herr Helfrich','Herr Kühne',
[P65]    59:    'Herr Birkenfeld','Bali','Hawaii',
[UX4]    6Ø:    'Hongkong','Malediven','Miami',
[KW4]    61:    'Bremen','Flensburg','Lübeck',
[YS5]    62:    'Hamburg','Kiel','Abteilungsleiter',
[GV4]    63:    'Ausbilder','Bürobesitzer',
[ZB5]    64:    'Büroleiter','Verkaufskraft'),
[ØD6]    65:   ('Benisch','Ehlers','Merkens','Schäfer',
[N16]    66:    'Weber','Krimi-Leser','Western-Leser',
[TF5]    67:    'Zeitungs-Leser','Rätsel-Rater',
[CS4]    68:    'Zeichner','2Ø Jahre','3Ø Jahre',
[UF4]    69:    '4Ø Jahre','5Ø Jahre','6Ø Jahre',
[3M6]    7Ø:    'Bebra','Hannover','München','Nürnberg',
[LB2]    71:    'Würzburg'));
[DG5]    72:   Wert: Array[1..9] of Word=(121Ø,1194,
[EB4]    73:   1194,1188,1219,12Ø3,1178,1178,1214);
[D67]    74:   Hinweise: Array[1..9,1..6,1..3] of String
               [81]=
[T7C]    75:  ((('Bei einem Briefmarken-Tauschabend sitz
               en die o.a. 5 Personen zusammen.',
[CXA]    76: 'Jeder hat zwei Sammelgebiete, die von den
               anderen verschieden sind.',
[7ØC]    77: 'Wer hat welche Sammelgebiete und wie lange
                betreibt jeder sein Hobby ?'),
[ULD]    78: ('Blanke sammelt seine Breifmarken schon lä
               nger als Grosch, aber doch fünf Jahre',
[677]    79: 'weniger als der Griechenland-Spezialist.',
[MN8]    8Ø: 'Der Übersee-Sammler hat am wenigsten Erfah
               rung.'),
[D1D]    81: ('Die Deutschland- und Frankreich-Sammler b
               etreiben ihr Hobby zusammen so viel',
[R7D]    82: 'Jahre wie Kollmann und der Portugal-Freund
               . Wehner ist seit 25 Jahren für die',
[NV3]    83: 'Schweiz zuständig.'),
[ENC]    84: ('','Keines der zusammengehörigen Sammelgeb
               iete hat gemeinsame Grenzen.',''),
[GUC]    85: ('Obwohl erst fünf Jahre dabei, hat Kollman
               n schon einige Raritäten aus dem',
[PN6]    86: 'nördlichsten Land in seinem Album.',''),
[VPC]    87: ('Herr Wehner sammelt Marken, auf denen kei
               ne Länderbezeichnung ist. Blanke',
[1BA]    88: 'bezieht viele Neuerscheinungen direkt vom
               Postschalter.','')),
[26C]    89: (('Die o.a. 5 Fernfahrer treffen sich auf d
               er Autobahnraststätte Koblenz.',
[CVA]    9Ø: 'Sie unterhalten sich über ihre Ladung und
               ihre Heimatstädte.',
[B97]    91: 'Wer hat was geladen und woher kommen sie ?
               '),
[DLD]    92: ('Der Fahrer, der Kaffee transportiert, ist
                am zweitweitesten von seiner Heimat-',
[WMH]    93: 'stadt entfernt, und er heißt nicht Wiegard
               . Kösters Wohnort liegt der Rast-','stätte
               am nächsten.'),
[Z7C]    94: ('','Weder der Hesse Peter, noch Guthmann t
               ransportieren Lebensmittel.',''),
[FFD]    95: ('','Roger ist Bayer und Müller hat auf sei
               nem Brummer Honigmelonen geladen.',''),
[S9F]    96: ('Christoph ist Tiroler, und der Fahrer mit
                der Brotladung ist kein Württem-','berger.
               ',''),
[XØD]    97: ('Die Rüben, die weder von Dirk noch von Ch
               ristoph gefahren werden, befördert',
[W25]    98: 'der Schleswig-Holsteiner.','')),
[A8D]    99: (('Die 5 Damen eines Büros sind alle Pferde
               - und Wettnarren. Beim letzten Derby',
[4K6]   1ØØ: 'haben sie wieder Wetten abgeschlossen.',
[SDC]   1Ø1: 'Wer hat welche Position im Betrieb und hat
                auf welches Pferd gesetzt ?'),
[3R9]   1Ø2: ('Anjas Pferd wurde knapp von dem der Empfa
               ngsdame besiegt.',
[387]   1Ø3: 'Diese hatte nicht auf Schnecke gesetzt.','
               '),
[SZC]   1Ø4: ('Die Auszubildende hatte zusätzlich mit An
               ita gewettet, daß ihr Pferd sich',
[V8A]   1Ø5: 'besser plazieren würde als Anitas.','Sie v
               erlor die Wette.'),
[R6B]   1Ø6: ('Das Pferd der Buchhalterin kam 3 Plätze h
               inter Marions ins Ziel.',
[1A8]   1Ø7: 'Marion hatte nicht auf Wilder Bulle getipp
               t.',''),
[E69]   1Ø8: ('Inges Pferd war schneller als das der Sch
               reibkraft.',
[U95]   1Ø9: 'Es schlug auch Spring-Lady.',''),
[ETA]   11Ø: ('','Kathrins Pferd war schneller als Lahme
                Ente und Schnecke.','')),
[HCD]   111: (('Zur Wahl des Rätselkönigs sitzen die o.a
               . 5 Personen in der Entausscheidung.',
[PMH]   112: 'Finden Sie die Namen, die Lieblingsrätsela
               rt und den erreichten Platz der','einzelnen
                Personen heraus !'),
[XLD]   113: ('Der Student hatte Pech, daß seine Rätsels
               pezialität nicht gefordert wurde.',
[1A7]   114: 'Zenker errang einen guten Mittelplatz.','
               '),
[3TC]   115: ('Frau Reuscher konnte mit ihrem Zahlenräts
               el so viel Zeit gutmachen, daß',
[PP7]   116: 'sie einen der vorderen Plätze erreichte.',
               ''),
[RØC]   117: ('Die Näherin schnitt besser ab als Schweim
               er, aber nicht so gut wie der',
[XV8]   118: 'Silbenrätsel-Experte.','Der Lehrer wurde D
               ritter.'),
[MFC]   119: ('Herr Müller war knapp zwei Minuten schnel
               ler fertig als der Student, was',
[LRC]   12Ø: 'in der Endabrechnung jedoch gleich vier Pl
               ätze Unterschied ausmachte.',''),
[K68]   121: ('','Krahs Spezialität sind die Bastelgitte
               r.','')),
[48F]   122: (('Beim großen Rennen auf dem Hockenheim-Ri
               ng befinden sich die o.a. 5 Fahrer','in der
                Endrunde.',
[MØA]   123: 'Wer steuert welches Fahrzeug und welche Fa
               rbe hat dieses ?'),
[CKE]   124: ('Volker fährt keinen Lotos und nicht das g
               raue Auto. Seine Startnummer ist niedri-',
[Y9D]   125: 'ger als die von McLaren, welcher nicht die
               Nr. 23 hat. Der Lotos ist nicht gelb.',
[6ØC]   126: 'Bernd fährt nicht das schwarze Auto und de
               r Ferrari hat nicht die Nr. 2.'),
[8AD]   127: ('Bernd hat eine ungerade Startnummer. Er f
               ährt nicht den Brabham und nicht das',
[PAD]   128: 'rote Auto. Karl fährt nicht das gelbe Auto
                und nicht den Renault, welcher nicht',
[G1D]   129: 'schwarz ist. Seine Startnummer ist weder d
               ie höchste, noch die niedrigste.'),
[TDD]   13Ø: ('Achim fährt nicht den Ferrari. Seine Star
               tnummer ergibt sich aus der Addition',
[FP7]   131: 'von Bernds Nummer und der des gelben Autos
               .','',''),
[Y6D]   132: ('Michael fährt nicht den McLaren und nicht
                das schwarze Auto. Er hat eine zwei-',
[GS7]   133: 'stellige Startnummer auf seinem Fahrzeug.'
               ,''),
[RØD]   134: ('Der Brabham, der nicht von Volker pilotie
               rt wird, hat eine niedrigere Start-',
[RFD]   135: 'nummer als das graue Auto.','Der Lotus ist
                schwarz, aber hat nicht sie Nr. 16.')),
```

```
[Z2D]  136: (('Bei einem Elternabend beraten 5 Väter üb
            er die Abhilfemöglichkeiten in den',
[TK5]  137: 'schwachen Fächern ihrer Kinder.',
[P6D]  138: 'Wer ist der Vater von wem, wie heißen sie
            und welche Schwächen hat das Kind ?'),
[END]  139: ('Herr Schweimer ist genauso viele Jahre jü
            nger als Petras Vater, wie der Vater',
[WY8]  14Ø: 'des Mathematik-schwachen Kindes zum 41-jäh
            rigen.','',
[8ZC]  141: ('Während Herr Hofner für seinen Sohn einen
             Nachhilfelehrer voschlägt, ist',
[D7D]  142: 'Rudis Vater mehr für gegenseitige Hilfeste
            llung. Der 41-jährige dagegen will',
[ZCC]  143: 'mit seinem Kind eine Reise nach London buc
            hen, um dem Übel beizukommen.'),
[NND]  144: ('Die Väter von Petra und Peter sind zusamm
            en zehn Jahre älter als Herr Schwei-',
[4CD]  145: 'mer und Sylvias Vater. Herr Wagner und Her
            r Hörl sind zusammen genauso alt',
[2V5]  146:  'wie Herr Kunze und Rudis Vater.'),
[1YB]  147:  ('Herr Wagner, als Senior in dieser Runde,
             kann seiner Tochter in Deutsch',
[784]  148:  'sowieso nicht helfen.','',
[FVA]  149: ('', 'Herr Wagner stimmt der Idee des Chemi
            e-geplagten Hofner zu.','')),
[U9D]  15Ø: (('Hollywood - Die Stadt der Stars und des
            Filmes. Doch ohne die Männer im Hinter-',
[ØDB]  151: 'grund läuft absolut nichts. Hier sind 5 di
            eser fleißigen Helfer.',
[YA9]  152: 'Wer hat welchen Job und woran arbeiten sie
             im Moment ?'),
[Z3D]  153: ('Knut arbeitet bereits seit 12 Jahren in d
            er Filmbranche; Saager und Wulff seit',
[YCE]  154: 'zehn Jahren, während Stefan und der Beleuc
            hter zusammen erst acht Jahre','dabei sind.
            '),
[Z8D]  155: ('Sowohl der Regieassistent als auch derjen
            ige, der an dem Musical mitarbeitet,',
```

```
[2CD]  156: 'haben kürzlich Preise gewonnen. Gerk und J
            anßen haben ihre ersten Filme vor',
[4R3]  157: 'vier Jahren gemacht.'),
[9DD]  158: ('Weder Janßen noch der Beleuchter haben bi
            sher gemeinsam mit dem Toningenieur',
[K4C]  159: 'an einem Film gearbeitet, während dies bei
             Knut und Wulff der Fall ist.',
[AF8]  16Ø: 'Paul ist weder Ton-Ingenieur noch der Kame
            ramann.'),
[8SD]  161: ('Die fünf Männer sind z.Z. mit Außenaufnah
            men beschäftigt: Berger in Spanien',
[S9H]  162: 'und der Beleuchter in Arizona; der Maskenb
            ildner ist in New York und Paul und','Björn
             sind auf Hawaii.'),
[38D]  163: ('Die Männer, die mit der Produktion der Ko
            mödie und des Musicals zu tun haben,',
[MUB]  164: 'sehen sich jeden Tag. Der Mann, der mit de
            m Krimi zu tun hat, ist nicht in',
[1GD]  165: 'Europa. Der Mann in Arizona, arbeitet wede
            r am Krimi noch an einem Western mit.')),
[8FE]  166: (('Bei einer Fachtagung für Reisebüro-Fachk
            räfte treffen sich die o.a. Personen und',
[X4E]  167: 'unterhalten sich über ihr persönliches Url
            aubsziel und ihre Stellung im Beruf.',
[66A]  168: 'Wer bekleidet welchen Posten und was ist s
            ein Traumurlaub ?'),
[YZC]  169: ('Frau Kampen wohnt in einer Hansestadt, ab
            er nicht in Lübeck. Sie ist weder Ab-',
[1MD]  17Ø: 'teilungsleiterin, noch war sie auf Hawaii.
             Der Büroleiter kommt nicht aus Ham-',
[4AA]  171: 'burg oder Lübeck, und der Ausbilder war ni
            cht auf einer Insel.'),
[W9D]  172: ('Frau Becker ist eine echte Sprotte. Sie w
            ar im Fernen Osten, aber nicht als Ver-',
[DCD]  173: 'kaufskraft oder Abteilungsleiterin. Herr B
            irkenfeld und der Ausbilder kommen',
[5HC]  174: 'nicht aus Bremen oder Lübeck und waren som
            it auch nicht auf den Malediven.'),
```

Der Quellcode von »logikal.pas« ist sehr übersichtlich.

NEU



Etiketten-Designer

für Windows

CeBIT'93 HANNOVER
24. — 31. 03. 1993
Halle 7, Stand E13/E19

Repräsentative Etiketten und Poster einfach und schnell

Überzeugen Sie Ihren Schriftverkehrs-Partner schon beim ersten Eindruck. Mit dem ***Etiketten-Designer für Windows*** haben Sie die Leistungsbreite eines kleinen DTP-Programms kombiniert mit den Datenimportfunktionen einer Profi-Datenbank. Ob es nun ein Mailing sein soll oder die Gestaltung einer Glückwunschkarte, ***Etiketten-Designer für Windows*** bietet Leistung nach Maß mit dem vollen Komfort von Windows.

- frei definierbare Etikettenformate bis DIN A-4
- 24 vordefinierte Etikettengrößen
- Import von dBASE-, ANSI- und ASCII-Daten für Seriendrucke
- Grafikeinbindung aus BMP-, PCX- und LBM-Dateien
- Nutzung aller Windows-Schriftarten

Windows ist ein eingetragenes Warenzeichen der Firma Microsoft

Bitte benutzen Sie die Bestellkarte im Heft.
Händleranfragen erwünscht.

* Unabhängig von der Anzahl der bestellten Produkte berechnen wir für das Inland DM 4,– bzw. für das Ausland DM 6,– Porto und Verpackung.

Etiketten-Designer für Windows DM 99,95*

ERGO-Calc für Windows DM 199,95*

DMV-Verlag · Postfach 250 · W-3440 Eschwege
Tel.: (0 56 51) 8 09-0 · Fax: (0 56 51) 8 09-4 30

DMV Daten- und Medienverlag

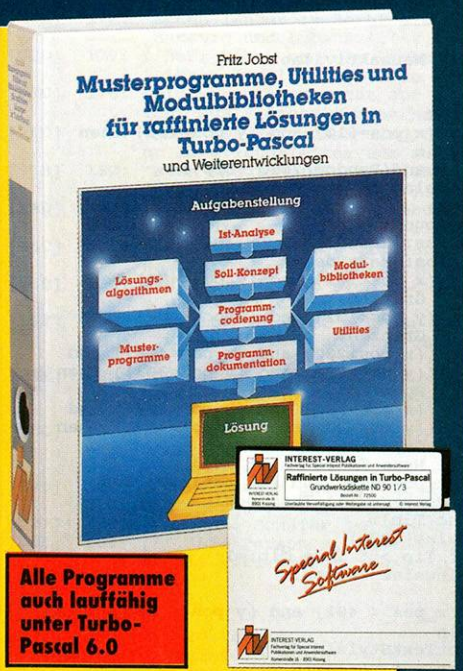
```
[6ZC]  175: ('Herr Helfrich ist jünger als der Bürobesi
            tzer und der Bali-Besucher, aber',
[H8B]  176: 'schon länger im Beruf als die Personen aus
             Hamburg und Lübeck.',''),
[52D]  177: ('Herr Kühne und die Verkaufskraft haben au
            f ihren Inseln das Surfen erlernt.',
[FRC]  178: 'Sie waren aber nicht auf Bali und kommen n
            icht aus Flensburg oder Hamburg.',''),
[2FD]  179: ('Die USA-Reisenden sind der Besitzer und d
            er Abteilungsleiter aus der nördlich-',
[3LB]  18Ø: 'sten Stadt. Sie sind nicht weiblich und he
            ißen nicht Helfrich.','')),
[EAD]  181: (('Im Schnellzug von Hamburg nach München s
            itzen die 5 Personen in einem Abteil.',
[Z6C]  182: 'Jeder beschäftigt sich auf eine andere Wei
            se bis zu seiner Endstation.',
[86B]  183: 'Wer beschäftigt sich mit was, steigt wo au
            s und wie alt ist er ?'),
[15F]  184: ('Weber ist älter als der Krimi-Leser, aber
             jünger als Benisch und der Bebra-','Besuch
            er.','),
[HKE]  185: ('Der Rätsel-Rater ist nur halb so alt wie
            Merkens. Dafür dauert seine Fahrt ','länger
            .','),
[XCG]  186: ('Nur bei dem Zeichner stimmen die Anfangsb
            uchstaben des Namens und des Reise-','ziele
            s überein.','),
[G2D]  187: ('Der älteste und der jüngste Fahrgast besc
            häftigen sich mit dem Lesen eines',
[V27]  188: 'Romanes. Schäfer ist nicht darunter.',''),
[SDE]  189: ('','Die Person mit dem längsten Reiseweg h
            at die kurzweiligste Beschäftigung.','')));
[ØØØ]  19Ø:
[ØØØ]  191:
[RNØ]  192: var
[KG5]  193:   Feld: Array[1..15,1..15] of Boolean;
[HK3]  194:   Register: Registers;
[FG5]  195:   x_pos, y_pos, ft, gd, gm : Integer;
[ZV3]  196:   x, y, Summe             : Word;
[FM4]  197:   Nr, Tip, Positive       : Byte;
[K45]  198:   Gewaehlt, Beenden       : Boolean;
[ØØØ]  199:
[1S4]  2ØØ: procedure Maus_anschalten;
[MUØ]  2Ø1: begin
[KM2]  2Ø2:   Register.ax:=1;
[ØZ2]  2Ø3:   Intr(51,Register);
[ZNØ]  2Ø4: end;
[ØØØ]  2Ø5:
[T15]  2Ø6: procedure Maus_ausschalten;
[MUØ]  2Ø7: begin
[NM2]  2Ø8:   Register.ax:=2;
[ØZ2]  2Ø9:   Intr(51,Register);
[ZNØ]  21Ø: end;
[ØØØ]  211:
[DE4]  212: procedure setze_MausPos;
[MUØ]  213: begin
[VM2]  214:   Register.ax:=4;
[GL3]  215:   Register.cx:=x_pos;
[PL3]  216:   Register.dx:=y_pos;
[ØZ2]  217:   Intr(51,Register);
[ZNØ]  218: end;
[ØØØ]  219:
[ZY4]  22Ø: procedure Mausinstallation;
[MUØ]  221: begin
[FM2]  222:   Register.ax:=Ø;
[ØZ2]  223:   Intr(51,Register);
[8X2]  224:   Maus_anschalten;
[SX1]  225:   x_pos:=61Ø;
[4Y1]  226:   y_pos:=28Ø;
[4P2]  227:   setze_MausPos;
[ZNØ]  228: end;
[ØØØ]  229:
[HL4]  23Ø: Function Maus_Taste:Byte;
[MUØ]  231: begin
[SM2]  232:   Register.ax:=3;
[ØZ2]  233:   Intr(51,Register);
[EA4]  234:   Maus_Taste:=Register.bx;
[ZNØ]  235: end;
[ØØØ]  236:
[EØ5]  237: Function MausPos_x:Integer;
[MUØ]  238: begin
[SM2]  239:   Register.ax:=3;
[ØZ2]  24Ø:   Intr(51,Register);
[X84]  241:   MausPos_x:=Register.cx;
[ZNØ]  242: end;
[ØØØ]  243:
[HØ5]  244: Function MausPos_y:Integer;
[MUØ]  245: begin
[SM2]  246:   Register.ax:=3;
[ØZ2]  247:   Intr(51,Register);
[194]  248:   MausPos_y:=Register.dx;
[ZNØ]  249: end;
[ØØØ]  25Ø:
[PT5]  251: procedure EGAVGADriver; External;
[G52]  252: {$L egavga}
[ØØØ]  253:
[Y45]  254: procedure LittFont; External;
[3P1]  255: {$L litt}
```

```
[ØØØ]  256:
[255]  257: procedure Grafikinstallation;
[MUØ]  258: begin
[WF6]  259:   ft:=registerbgidriver(@EGAVGADriver);
[ZH5]  26Ø:   ft:=registerbgifont(@LittFont);
[TY1]  261:   gd:=Detect;
[333]  262:   InitGraph(gd,gm,'');
[RY1]  263:   SetColor(Ø);
[ZNØ]  264: end;
[ØØØ]  265:
[7H3]  266: procedure Spielfeld;
[MUØ]  267: begin
[833]  268:   Maus_ausschalten;
[FZ1]  269:   SetColor(8);
[C53]  27Ø:   SetLineStyle(Ø,Ø,3);
[BG2]  271:   Line(  Ø,  Ø,4Ø4,  Ø);
[BS2]  272:   Line(4Ø4,  Ø,4Ø4,4Ø4);
[LV2]  273:   Line(4Ø4,2Ø2,2Ø2,2Ø2);
[FV2]  274:   Line(2Ø2,2Ø2,2Ø2,3Ø3);
[RV2]  275:   Line(2Ø2,3Ø3,3Ø3,3Ø3);
[ZV2]  276:   Line(3Ø3,3Ø3,3Ø3,4Ø4);
[BS2]  277:   Line(4Ø4,4Ø4,  Ø,4Ø4);
[BG2]  278:   Line(  Ø,4Ø4,  Ø,  Ø);
[LR2]  279:   Line(1Ø1,  Ø,1Ø1,4Ø4);
[LR2]  28Ø:   Line(  Ø,1Ø1,4Ø4,1Ø1);
[PZ2]  281:   SetFillStyle(1,4);
[GH2]  282:   FloodFill(2,2,8);
[163]  283:   SetFillStyle(1,15);
[2W2]  284:   FloodFill(1Ø3,3,8);
[EW2]  285:   FloodFill(3,1Ø3,8);
[VZ2]  286:   SetFillStyle(1,9);
[GØ3]  287:   FloodFill(1Ø3,1Ø3,8);
[LR2]  288:   Line(  Ø,2Ø2,2Ø2,2Ø2);
[VR2]  289:   Line(  Ø,3Ø3,2Ø2,3Ø3);
[VR2]  29Ø:   Line(2Ø2,  Ø,2Ø2,4Ø4);
[VR2]  291:   Line(3Ø3,  Ø,3Ø3,2Ø2);
[653]  292:   SetLineStyle(Ø,Ø,1);
[ØG3]  293:   For x:=1 to 4 do begin
[C74]  294:    Line(Ø,1Ø1 + x*2Ø,4Ø4,1Ø1 + x*2Ø);
[A74]  295:    Line(Ø,2Ø2 + x*2Ø,2Ø2,2Ø2 + x*2Ø);
[U74]  296:    Line(Ø,3Ø3 + x*2Ø,3Ø3,3Ø3 + x*2Ø);
[RA4]  297:    Line(1Ø1 + x*2Ø, Ø,1Ø1 + x*2Ø,4Ø4);
[AC4]  298:    Line(2Ø2 + x*2Ø,  Ø,2Ø2 + x*2Ø,2Ø2);
[MF4]  299:    Line(2Ø2 + x*2Ø,3Ø3,2Ø2 + x*2Ø,4Ø4);
[MC4]  3ØØ:    Line(3Ø3 + x*2Ø,  Ø,3Ø3 + x*2Ø,2Ø2);
[ZNØ]  3Ø1:   end;
[A32]  3Ø2:   SetColor(14);
[S83]  3Ø3:   SetTextStyle(2,Ø,7);
[MD4]  3Ø4:   OutTextxy(13,15,'Logikal');
[F83]  3Ø5:   SetTextStyle(2,Ø,4);
[RY1]  3Ø6:   SetColor(Ø);
[R44]  3Ø7:   OutTextxy(25,45,'(C) Ø1/92');
[B25]  3Ø8:   OutTextxy(17,6Ø,'J. Ofenstein');
[RF5]  3Ø9:   OutTextxy(15,7Ø,'Bilsteinstr. 4');
[UH4]  31Ø:   OutTextxy(22,8Ø,'64ØØ Fulda');
[A32]  311:   SetColor(14);
[PZ2]  312:   SetFillStyle(1,4);
[WRØ]  313:   y:=2;
[AG3]  314:   For x:=1 to 9 do begin
[3B4]  315:     Bar3D(435,y,639,y+28,Ø,False);
[5F1]  316:     Inc(y,3Ø);
[ZNØ]  317:   end;
[MZ2]  318:   SetFillStyle(1,2);
[3B4]  319:     Bar3D(435,y,639,y+28,Ø,False);
[UZ2]  32Ø:   SetFillStyle(1,8);
[Ø13]  321:   FloodFill(2Ø5,2Ø5,8);
[Y22]  322:   SetColor(11);
[K83]  323:   SetTextStyle(2,Ø,5);
[3T2]  324:   OutTextxy(442,  7,
[ØL4]  325:        'Briefmarken-Tauschabend');
[PT2]  326:   OutTextxy(442, 37,
[6C5]  327:       'Autobahnraststätte Koblenz');
[TT2]  328:   OutTextxy(442, 67,
[G54]  329:             'Pferdewetten im Büro');
[WT2]  33Ø:   OutTextxy(442, 97,
[DB4]  331:           'Wahl des Rätselkönigs');
[4V5]  332:   OutTextxy(442,127,'Hockenheim-Ring');
[AR5]  333:   OutTextxy(442,157,'Der Elternabend');
[Y15]  334:   OutTextxy(442,187,'Hollywood');
[967]  335:  OutTextxy(442,217,'Reisebüro-Fachkräfte');
[XN5]  336:   OutTextxy(442,247,'Im Schnellzug');
[AH5]  337:   OutTextxy(442,277,'Spiel beenden');
[CZ1]  338:   SetColor(7);
[ZU2]  339:   OutTextxy(21Ø,21Ø,
[KG4]  34Ø:             'Tragen Sie in dieses Dia-');
[4V2]  341:   OutTextxy(21Ø,222,
[GK4]  342:             'gramm alle Informationen');
[9V2]  343:   OutTextxy(21Ø,234,
[Ø44]  344:             'ein, die Sie aus den Hin-');
[EV2]  345:   OutTextxy(21Ø,246,
[3P4]  346:             'weisen entnehmen können.');
[A32]  347:   SetColor(14);
[8V2]  348:   OutTextxy(21Ø,262,
[BK4]  349:             'linke  Maustaste positiv');
[DV2]  35Ø:   OutTextxy(21Ø,274,
[VN4]  351:             'rechte Maustaste negativ');
[KV2]  352:   OutTextxy(21Ø,286,
[RM4]  353:             'mittl. Maustaste löschen');
```

```
[114]  354:    OutTextxy(32Ø,335,'Viel');
[4S4]  355:    OutTextxy(34Ø,355,'Spaß !');
[ZK2]  356:    For x:=1 to 15 do
[1L2]  357:      For y:=1 to 15 do
[5X2]  358:        Feld[x,y]:=False;
[8X2]  359:    Maus_anschalten;
[ZNØ]  36Ø: end;
[ØØØ]  361:
[SS6]  362: procedure Werte_in_Tabelle_eintragen;
[MUØ]  363: begin
[833]  364:    Maus_ausschalten;
[RY1]  365:    SetColor(Ø);
[ND3]  366:    SetTextJustify(Ø,2);
[F83]  367:    SetTextStyle(2,Ø,4);
[G31]  368:    y:=1Ø5;
[DN3]  369:    For x:=1 to 15 do begin
[6U4]  37Ø:      OutTextxy(5,y,Werte[Nr,x]);
[1F1]  371:      Inc(y,2Ø);
[ZNØ]  372:    end;
[G83]  373:    SetTextStyle(2,1,4);
[FD3]  374:    SetTextJustify(Ø,Ø);
[E31]  375:    x:=1Ø5;
[FM3]  376:    For y:=16 to 2Ø do begin
[UØ5]  377:      OutTextxy(x,95,Werte[Nr,y]);
[ZE1]  378:      Inc(x,2Ø);
[ZNØ]  379:    end;
[MN3]  38Ø:    For y:=6 to 15 do begin
[UØ5]  381:      OutTextxy(x,95,Werte[Nr,y]);
[ZE1]  382:      Inc(x,2Ø);
[ZNØ]  383:    end;
[8X2]  384:    Maus_anschalten;
[ZNØ]  385: end;
[ØØØ]  386:
[1S3]  387: procedure Auswertung;
[MUØ]  388: begin
[832]  389:    Positive:=Ø;
[PP1]  39Ø:    Summe   :=Ø;
[ZK2]  391:    For x:=1 to 15 do
[1L2]  392:      For y:=1 to 15 do
[3W3]  393:        If Feld[x,y] Then begin
[NF2]  394:          Inc(Positive);
[YL2]  395:          Inc(Summe,(x*y));
[ZNØ]  396:        end;
[NE4]  397:    If (Positive=3Ø) Then begin
[833]  398:      Maus_ausschalten;
[LK2]  399:      ClearViewPort;
[ØA2]  4ØØ:      Write(#7#7#7#7#7);
[383]  4Ø1:      SetTextStyle(Ø,Ø,2);
[2T4]  4Ø2:      If Summe=Wert[Nr] Then begin
[ZU2]  4Ø3:        OutTextxy(15Ø,15Ø,
[NU4]  4Ø4:                'Herzlichen Glückwunsch');
[D25]  4Ø5:        OutTextxy(2ØØ,2ØØ,'zum Logiker');
[F15]  4Ø6:        OutTextxy(25Ø,25Ø,'des Jahres !');
[4MØ]  4Ø7:      end
[WP1]  4Ø8:      else begin
[ZU2]  4Ø9:        OutTextxy(15Ø,15Ø,
[LZ3]  41Ø:                'Das tut mir aber leid,');
[XU2]  411:        OutTextxy(2ØØ,2ØØ,
[UN3]  412:                  'aber das war nichts !');
[ZNØ]  413:      end;
[9P1]  414:    Delay(6ØØØ);
[CE2]  415:    Beenden:=True;
[ZNØ]  416:    end;
[ZNØ]  417: end;
[ØØØ]  418:
[ØP4]  419: procedure Hinweise_Zeigen;
[MUØ]  42Ø: begin
[833]  421:    Maus_ausschalten;
[B25]  422:    If (Nr > Ø) and (Nr < 1Ø) Then begin
[E32]  423:      SetColor(15);
[PZ2]  424:      SetFillStyle(1,4);
[X14]  425:      Bar3D(Ø,411,639,466,Ø,False);
[A94]  426:      Bar3D(5ØØ,39Ø,639,411,Ø,False);
[K83]  427:      SetTextStyle(2,Ø,5);
[FD3]  428:      SetTextJustify(Ø,Ø);
[A32]  429:      SetColor(14);
[EB6]  43Ø:      OutTextxy(51Ø,4Ø3,'Nächster Hinweis');
[E32]  431:      SetColor(15);
[R31]  432:      y:=425;
[U94]  433:      If Tip=1 Then SetColor(14);
[YF3]  434:      For x:=1 to 3 do begin
[C26]  435:        OutTextxy(1Ø,y,Hinweise[Nr,Tip,x]);
[4F1]  436:        Inc(y,17);
[ZNØ]  437:      end;
[ZNØ]  438:    end;
[8X2]  439:    Maus_anschalten;
[ZNØ]  44Ø: end;
[ØØØ]  441:
[H73]  442: procedure Sperren;
[MUØ]  443: begin
[ZX1]  444:    x_pos:=55Ø;
[PY1]  445:    y_pos:=395;
[4P2]  446:    setze_MausPos;
[ZNØ]  447: end;
[ØØØ]  448:
[R63]  449: procedure Grenzen;
[MUØ]  45Ø: begin
[ØY1]  451:    x_pos:=1Ø5;
[2Y1]  452:    y_pos:=1Ø5;
[4P2]  453:    setze_MausPos;
[ZNØ]  454: end;
[ØØØ]  455:
[FE4]  456: procedure Mausaktivitaet;
[MUØ]  457: begin
[P31]  458:   Repeat
[P31]  459:     Repeat
[VB5]  46Ø:       If (x_pos=61Ø) and (y_pos=28Ø) Then
[MUØ]  461:       begin
[LG3]  462:         Sound(Random(2ØØØ)+8ØØ);
[DN1]  463:         Delay(1ØØ);
[ZNØ]  464:       end;
[3G1]  465:       NoSound;
[F73]  466:       x_pos:=MausPos_x;
[K73]  467:       y_pos:=MausPos_y;
[CB5]  468:       If (x_pos<1Ø2) and (y_pos<4Ø4) Then
[WE1]  469:          Grenzen;
[CB5]  47Ø:       If (y_pos<1Ø2) and (x_pos<4Ø4) Then
[WE1]  471:          Grenzen;
[8L5]  472:       If ((x_pos>2ØØ) and (x_pos<4Ø4)) and
[R95]  473:          ((y_pos>2ØØ) and (y_pos<3Ø3)) Then
[2F1]  474:          Sperren;
[9L5]  475:       If ((x_pos>3ØØ) and (x_pos<4Ø4)) and
[BA5]  476:          ((y_pos>3Ø3) and (y_pos<4Ø4)) Then
[WE1]  477:          Grenzen;
[AT4]  478:     Until Maus_Taste in [1..4];
[3D5]  479:     If (x_pos>5ØØ) and ((y_pos>39Ø) and
[FT3]  48Ø:        (y_pos<411)) Then begin
[KG1]  481:       Inc(Tip);
[HN1]  482:       Delay(2ØØ);
[VE3]  483:       If Tip > 6 Then Tip:=1;
[9Z2]  484:       Hinweise_zeigen;
[ZNØ]  485:     end;
[WV5]  486:     If (x_pos < 4Ø2) and (y_pos < 4Ø2) Then
[MUØ]  487:     begin
[383]  488:       SetTextStyle(Ø,Ø,2);
[ND3]  489:       SetTextJustify(Ø,2);
[833]  49Ø:       Maus_ausschalten;
[VZ2]  491:       SetFillStyle(1,9);
[KZ1]  492:       SetColor(9);
[MS5]  493:       x:=Trunc((x_pos-Trunc(x_pos/1Ø1))/
[NU3]  494:                2Ø)*2Ø+Trunc(x_pos/1Ø1)+3;
[YS5]  495:       y:=Trunc((y_pos-Trunc(y_pos/1Ø1))/
[UU3]  496:                2Ø)*2Ø+Trunc(y_pos/1Ø1)+4;
[6X4]  497:       Bar3D(x-1,y-2,x+14,y+14,Ø,False);
[A32]  498:       SetColor(14);
[CA4]  499:       If Maus_Taste=1 Then begin
[373]  5ØØ:         OutTextxy(x,y,'+');
[HN1]  5Ø1:         Delay(2ØØ);
[8X5]  5Ø2:         Feld[Trunc((x_pos-1ØØ-Trunc(x_pos/
[NB1]  5Ø3:                    1ØØ))/2Ø)+1,
[XT4]  5Ø4:              Trunc((y_pos-1ØØ-Trunc(y_pos/
[SP2]  5Ø5:                    1ØØ))/2Ø)+1]:=True;
[ZNØ]  5Ø6:       end;
[GA4]  5Ø7:       If Maus_Taste=2 Then begin
[573]  5Ø8:         OutTextxy(x,y,'-');
[HN1]  5Ø9:         Delay(2ØØ);
[8X5]  51Ø:         Feld[Trunc((x_pos-1ØØ-Trunc(x_pos/
[NB1]  511:                  1ØØ))/2Ø)+1,
[XT4]  512:            Trunc((y_pos-1ØØ-Trunc(y_pos/
[AS2]  513:                  1ØØ))/2Ø)+1]:=False;
[ZNØ]  514:       end;
[VB3]  515:       If Maus_Taste=4 Then
[8X5]  516:         Feld[Trunc((x_pos-1ØØ-Trunc(x_pos/
[NB1]  517:                    1ØØ))/2Ø)+1,
[XT4]  518:              Trunc((y_pos-1ØØ-Trunc(y_pos/
[AS2]  519:                    1ØØ))/2Ø)+1]:=False;
[652]  52Ø:       Auswertung;
[ZNØ]  521:     end;
[TP2]  522:     If (x_pos>435) and
[S54]  523:        (Trunc(y_pos/3Ø)+1=1Ø) Then
[CE2]  524:       Beenden:=True;
[EØ6]  525:     If (x_pos > 435) and not Gewaehlt Then
[MUØ]  526:     begin
[WZ3]  527:       Nr:=Trunc(y_pos / 3Ø) + 1;
[RØ3]  528:       If Nr < 1Ø Then begin
[KS2]  529:         Gewaehlt:=True;
[X25]  53Ø:         Werte_in_Tabelle_eintragen;
[9Z2]  531:         Hinweise_zeigen;
[ZNØ]  532:       end;
[ZNØ]  533:     end;
[8X2]  534:     Maus_anschalten;
[8C2]  535:   Until Beenden;
[ZNØ]  536: end;
[ØØØ]  537:
[MUØ]  538: begin
[4U2]  539:   Gewaehlt:=False;
[8P2]  54Ø:   Beenden :=False;
[SC1]  541:   Nr      :=Ø;
[DK1]  542:   Tip     :=1;
[DG3]  543:   Grafikinstallation;
[U33]  544:   Mausinstallation;
[2S1]  545:   Spielfeld;
[AW2]  546:   Mausaktivitaet;
[522]  547:   CloseGraph;
[KNØ]  548: end.
```

Der Quellcode von »logikal.pas« ist sehr übersichtlich.

Jetzt ist Ihr Erfolg vorprogrammiert!



Alle Programme auch lauffähig unter Turbo-Pascal 6.0

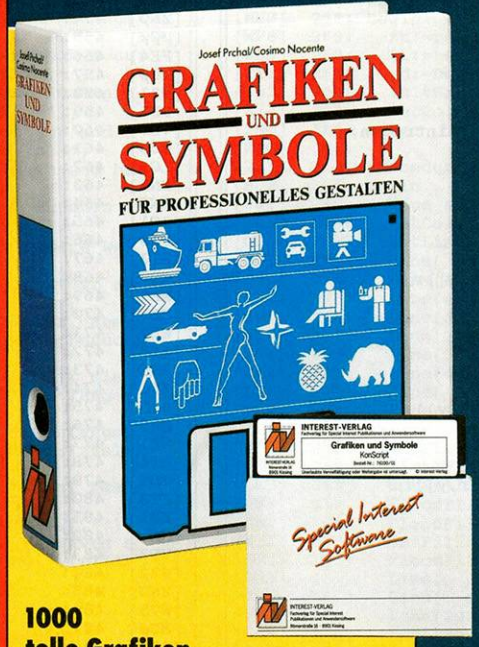
Lösungen in Turbo-Pascal

Dieses Nachschlagewerk bietet Ihnen u.a.:

- **detailliertes Know-how zum Turbo-Pascal-System;**
- **Betriebssystem-Routinen:** u.a. ROM-BIOS und IBM-Hardware (BIOS-Variable, Adapter für asynchrone Datenübertragung, Druckeransteuerung);
- **hardwarenahe Programmierung:** u.a. Disk-Optimizer, Floppy-Tuning, Prefix-Segment und Environment, Expanded Memory (EMS);
- **raffinierte Utilities: Schnittstellenutilities** (u.a. Turbo-Inliner, Zugriff auf dBASE-Dateien), **Grafikutilities** (Ausdruck von Farbgrafiken als Schwarz/Weiß-Muster), **Datenschutz** (u.a. Verschlüsseln von Dateien);
- **grafische Benutzeroberfläche SISGRAF:** GEM-ähnliche Benutzeroberfläche zur Einbindung in eigene Grafikprogramme oder zur Programmierung von Systemsteuerungen à la WINDOWS;
- **Anwenderprogramme für Technik/Mathematik und Betriebswirtschaft:** u.a. statistische Qualitätskontrolle, Schrittmotor-Simulation, Finanzbuchhaltung SOBUCH;
- **Entwicklungsprogramme:** Editoren (Texteditor in Pascal, Vektor-Zeichensatzeditor), Datenbankanwendungen (INTERDAT);
- **Turbo-Pascal-Modulbibliotheken,** u.a. Datum-, String- und DOS-Routinen, Mausfunktionen, EMS-Unterstützung und virtuelle Arrays.

Grundwerk ca. 600 Seiten, DIN A4, inkl. drei 5¼"-Disketten, Bestell-Nr.: 2500, Preis: DM 99,–, wahlweise zwei 3½"-Disketten (gegen DM 4,– Aufpreis)

Alle 2–3 Monate erhalten Sie Erweiterungen mit je ca. 130 Seiten, inkl. Diskette, zum Preis von DM 59,50 (Abbestellung jederzeit möglich).



1000 tolle Grafiken

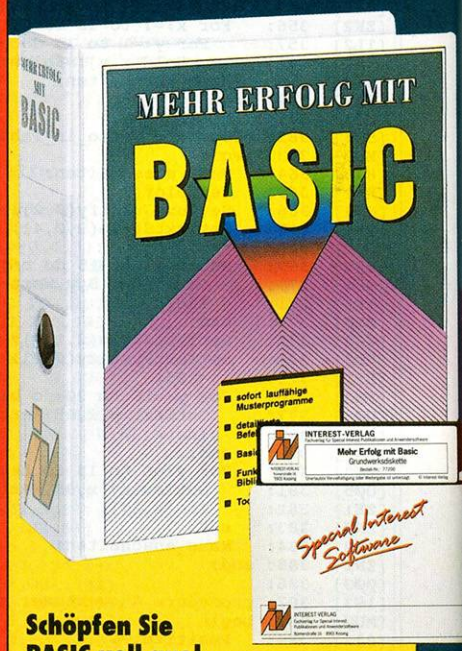
Dieses neue Nachschlagewerk bietet Ihnen Grafikdateien und reprofähige Vorlagen in PostScript-Qualität.

Aus dem Inhalt:

- **umfangreiche Grafikbibliothek** aus den Bereichen Mensch und Freizeit, Beruf, Architektur, Technik, Pflanzen und Tiere, Karikaturen …
- **Symbole und Piktogramme** u.a. als Orientierungshilfe oder zur Wiedererkennung;
- **Blickfänger:** u.a. Buttons, Sterne, Pfeile, Rahmen, Schattenlinien zur Gestaltung origineller Privatdrucksachen;
- **PostScript-Qualität** für erfolgreichen DTP-Einsatz. 20 Disketten mit sämtlichen Grafiken, die Sie ohne Schärfeverlust nutzen können;
- **Konvertierungsprogramm „KonScript"** für die Umwandlung der Grafikdateien in Pixel- (PCX, TIFF, IMG, GIF) oder Vektorformat (EPS, CGM, GEM).
- **problemlose Übernahme** der Grafiken in Ihre Software (u.a. Word, WinWord, WordPerfect, PageMaker, Ventura Publisher, Corel Draw, Ami Pro, Geo Works Ensemble, HG…).

Grundwerk ca. 250 Seiten, DIN A4, inkl. zwanzig 5¼"-Disketten + gratis Demodiskette, Bestell-Nr.: 6100, Preis: DM 199,–, wahlweise zehn 3½"-Disketten (gegen DM 15,– Aufpreis)

Alle 2–3 Monate erhalten Sie Erweiterungen mit über 70 Seiten, inkl. ca. zehn 5¼"-Disketten bzw. fünf 3½"-Disketten (gegen DM 2,– Aufpreis pro Diskette) zum Preis von DM 99,– (Abbestellung jederzeit möglich).



Schöpfen Sie BASIC voll aus!

Dieses speziell für den Programmierer entwickelte Nachschlagewerk zeigt Ihnen, wie Sie schnell und sicher in Basic programmieren.

Sie erhalten u.a.:

- **eine detaillierte Befehls- und Funktionsreferenz** zu allen wichtigen Basic-Dialekten, u.a. GW-, Power-, Quick-, Q- und Visual-Basic.
- **praxisorientierte Musterprogramme,** u.a.: **Datenbanksystem SOUNDEX:** Sie bearbeiten Datensätze, editieren Texte, schreiben Serienbriefe. Zusätzlich erhalten Sie fertige Datenbank-Lösungen wie z.B. Adreßverwaltung, Etikettendruckprogramm; **Personalzeitsystem INPERS:** komfortable Verwaltung von bis zu 60 Mitarbeitern;
- **Basic-Programmierkurs:** für den sicheren Umgang mit Datentypen, Kontrollstrukturen, Einsatz der Interrupt-Programmierung …;
- **Konvertierungsprogramm CONBAS:** zur Portierung Ihrer Programme von GW-Basic nach Quick- oder Visual-Basic;
- **Visual-Basic unter Windows:** mit umfassenden Befehlsübersichten und VB-Musterlösungen erstellen Sie Ihre eigenen Windows-Programme!
- **raffinierte Tools und Utilities:** z.B. einen Ausgabemasken-Interpreter, mit dem Sie ASCII-Bildschirmmasken bei Laufzeit der Basic-Applikationen interpretieren und anzeigen können, komplette Directory-Verwaltung, komfortable Mausemulation u.v.m.

Grundwerk ca. 450 Seiten, DIN A5, inkl. zwei 5¼"-Disketten, Bestell-Nr.: 7200, Preis: DM 99,–, wahlweise 3½"-Diskette (gegen DM 2,– Aufpreis)

Alle 2–3 Monate erhalten Sie Erweiterungen mit je ca. 100 Seiten, inkl. Diskette, zum Preis von DM 59,50 (Abbestellung jederzeit möglich).

INTEREST-VERLAG
Fachverlag für Special Interest
Publikationen und Anwendersoftware
Römerstraße 16
W-8901 Kissing
Tel. 0 82 33 / 2 11-0
Fax 0 82 33 / 2 11-2 99

Prüfen Sie jedes Werk in Ruhe zu Hause. Wenn Sie nicht überzeugt sind, schicken Sie dieses mit den Disketten innerhalb von 10 Tagen an uns zurück. Damit ist die Sache für Sie erledigt!

Für Ihre Anforderung verwenden Sie bitte nebenstehende Bestellkarte!

Sichern Sie sich heute noch
Ihr **ERGO Base/Calc Paket**
zum Superpreis von nur
DM 299,95

PC Software

5 1/4	3 1/2		DM
81787029 ❑	81787030 ❑	DMV-Profi-Tools f. Clipper	199,–
81787062 ❑	81787063 ❑	Clip and Vision 1.0	199,–
81787065 ❑	81787066 ❑	Clip and Vision-Update	99,–
81780013 ❑	81780014 ❑	Fraktal Animator 3D	69,–
81780025 ❑	81780026 ❑	Etiketten-Designer f. Windows	99,95
		Power Hits	
81784001 ❑	81784002 ❑	Multi Copy	29,80
81784010 ❑	81784011 ❑	Tippsy	29,80
81784013 ❑	81784014 ❑	Menue	29,80
81784016 ❑	81784017 ❑	Terminplaner	29,80
81784019 ❑	81784020 ❑	PC-Astro	29,80
81784022 ❑	81784023 ❑	Text Compiler	29,80
81784025 ❑	81784026 ❑	Fraktal 3D	29,80
81784028 ❑	81784029 ❑	PC-Spiel des Wissens 1	29,80
		Happy Windows Collection	
81785001 ❑	81785002 ❑	Adressen-Manager	49,80
81785004 ❑	81785005 ❑	Icon-Designer	49,80
81785007 ❑	81785008 ❑	Literatur-Verwaltung	49,80
81785010 ❑	81785011 ❑	Briefmarken-Verwaltung	49,80
81785013 ❑	81785014 ❑	Musik-Verwaltung	49,80
81785016 ❑	81785017 ❑	Video-Verwaltung	49,80
81785019 ❑	81785020 ❑	Im Tal der Könige	49,80
81785022 ❑	81785023 ❑	Bildschirmschoner	49,80
81785028 ❑	81785029 ❑	Biorhythmus	49,80
81785025 ❑	81785026 ❑	Mega-Album I	49,80
81785031 ❑	81785032 ❑	Diaschau	49,80

60 Pfennig, die sich lohnen

Antwortkarte

DMV-Verlag
Versandservice
Postfach 250

W-3440 Eschwege



Die
DMV-Sammelordner

Sonderhefte - Sammelordner - Bücher

Ausgabe	je DM 18,– Bestell-Nr.	Databox je DM 35,– 5 1/4"	Databox je DM 35,– 3 1/2"
DOS Extra Nr. 3	99900306 ❑	81789024 ❑	81789025 ❑
DOS Extra Nr. 4	99900406 ❑	81789026 ❑	81789027 ❑
DOS Extra Nr. 5	99900506 ❑	81789028 ❑	81789029 ❑
DOS Extra Nr. 6	99900606 ❑	81789030 ❑	81789031 ❑
DOS Extra Nr. 7	99900706 ❑	81789032 ❑	81789033 ❑
DOS Extra Nr. 8	99900806 ❑	81789034 ❑	81789035 ❑
DOS Extra Nr. 10	99901006 ❑	81789038 ❑	81789039 ❑
DOS Extra Nr. 12	99901206 ❑	81789042 ❑	81789043 ❑
DOS Extra Nr. 13	99901306 ❑	–	–
DOS Extra Nr. 14	99901406 ❑	81789044 ❑	81789045 ❑
DOS Extra Nr. 15	99901506 ❑	81789046 ❑	81789047 ❑
DOS Extra Nr. 16	99901606 ❑	81789048 ❑	81789049 ❑
DOS Extra Nr. 17	99901706 ❑	81789050 ❑	81789051 ❑
DOS Extra Nr. 18	99901806 ❑	81789052 ❑	81789053 ❑
DOS Extra Nr. 19	99901906 ❑	81789054 ❑	81789055 ❑
DOS Extra Nr. 20	99902006 ❑	81789056 ❑	81789057 ❑
DOS Extra Nr. 21	99902106 ❑	81789058 ❑	81789059 ❑
WinDOS Heft Nr. 1	88880020 ❑		
WinDOS Heft Nr. 2	88880021 ❑		

Ausgabe	je DM 18,– Bestell-Nr.	Databox je DM 35,– 5 1/4"	Databox je DM 35,– 3 1/2"
DMV Computerwissen Band 1	88880022 ❑	81789001 ❑	81789002 ❑
DMV Computerwissen Band 2	88880023 ❑	81789003 ❑	81789004 ❑
DMV Computerwissen Band 3	88880024 ❑	81789005 ❑	81789006 ❑
DMV Computerwissen Band 4	88880025 ❑	81789007 ❑	81789008 ❑
	je DM 28,–		
DOS-Toolbox Nr. 1	99900102 ❑		
DOS-Toolbox Nr. 2	99900202 ❑		
DOS-Toolbox Nr. 3	99900302 ❑		
DOS-Toolbox Nr. 4	99900402 ❑		
DOS-Toolbox Nr. 5	99900502 ❑		
Bücher	**je DM 19,80**		
Context 4.0 Taschenbuch	92617710 ❑		
Windows 3.0 Taschenbuch	92617709 ❑		
Sammelmappen	**2 St. DM 15,80**		
Sammelmappen 2 St. DOS	81780909 ❑		
Sammelmappen 2 St. WinDOS	81780908 ❑		

+ Porto/Verpackung
(Inland 4,– DM, Ausland 6,– DM) _____ DM = Gesamtbetrag _____ DM



Programmieren Sie Ihren Erfolg!

PRÜFEN SIE

Ihr Nachschlagewerk in Ruhe zu Hause. Wenn Sie nicht überzeugt sind, schicken Sie dieses mit den Disketten innerhalb von 10 Tagen an uns zurück. Damit ist die Sache für Sie erledigt.

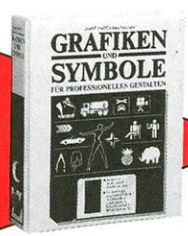
Bitte abtrennen oder ausschneiden!

Ja, senden Sie mir sofort:



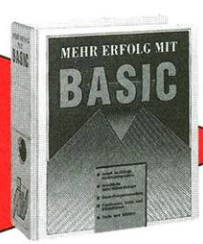
☐ Expl. **„Musterprogramme, Utilities und Modulbibliotheken für raffinierte Lösungen in Turbo-Pascal"**

Praktischer Ringbuchordner, DIN A4, ca. 600 Seiten, inkl. drei 5 1/4"-Disketten, Bestell-Nr.: 2500, Preis: DM 99,–, wahlweise zwei 3 1/2"-Disketten.



☐ Expl. **„Grafiken und Symbole für professionelles Gestalten"**

Prakt. Ringbuchordner, DIN A4, ca. 250 Seiten, inkl. zwanzig 5 1/4"-Disketten + gratis Demodiskette, Bestell-Nr.: 6100, Preis: DM 199,–, wahlweise zehn 3 1/2"- Disketten (gegen DM 15,– Aufpreis)



☐ Expl. **„Mehr Erfolg mit Basic"**

Praktischer Ringbuchordner, DIN A5, ca. 450 Seiten, inkl. zwei 5 1/4"-Disketten, Bestell-Nr.: 7200, Preis: DM 99,–, wahlweise 3 1/2"-Diskette.

Disketten-Service
Ich wünsche die Disketten im
☐ 3 1/2"-Format (gegen DM 2,– Aufpreis pro Diskette)

Zu jedem dieser Werke erhalten Sie alle 2–3 Monate Erweiterungsausgaben zu den in der Anzeige angegebenen Preisen.

Nicht vergessen!
Geben Sie bitte umseitig Ihre vollständige Adresse an.

▶▶▶

Ja, senden Sie mir

Das sensationelle Anwender-Paket für Windows

☐ Expl. **ERGO Base f. Windows**
Zum Einzelpreis von 199,95

☐ 5,25 Zoll **Bestell-Nr. 81783050**
☐ 3,5 Zoll **Bestell-Nr. 81783051**

☐ Expl. **ERGO Calc f. Windows**
Zum Einzelpreis von 199,95

☐ 5,25 Zoll **Bestell-Nr. 81783056**
☐ 3,5 Zoll **Bestell-Nr. 81783057**

☐ Expl. **Paket ERGO Base/Calc f. Windows**
zum Paketpreis von 299,95

☐ 5,25 Zoll **Bestell-Nr. 81783064**
☐ 3,5 Zoll **Bestell-Nr. 81783065**

Unabhängig von der Anzahl der bestellten Produkte berechnen wir für das Inland DM 4,– /Ausland DM 6,– Porto und Verpackung.

Bitte kreuzen Sie die gewünschte Zahlungsweise an:

☐ beigefügter Verrechnungsscheck
☐ nach Erhalt der Rechnung

Meine Anschrift:

Name

Vorname

Straße, Hausnummer

PLZ/Ort

Telefon für eventuelle Rückfragen

Datum Unterschrift (bei Minderjährigen des gesetzlichen Vertreters)

Sichern Sie sich heute noch
Ihr **ERGO Base/Calc Paket**
zum Superpreis von nur
DM 299,95

PC-Sonderhefte

Ja, ich mache von Ihrem Angebot Gebrauch und bestelle die umseitig ausgewählten Produkte. Bitte liefern Sie an folgende Anschrift:

Name

Straße / Postfach

PLZ / Ort

❑ Den Betrag bezahle ich mit dem beigefügten Verrechnungsscheck.
❑ Gegen Rechnung ab einem Warenwert von DM 50,–

Datum Unterschrift

Bitte ausreichend frankieren

Antwortkarte

DMV-Verlag
DOS International
Postfach 250

W-3440 Eschwege

Informieren

DOS-Taschenbücher und DOS-EXTRA-Sonderhefte sind eine zuverlässige Informationsquelle für jeden PC-Besitzer. Nutzen auch Sie dieses Wissen, und vertiefen Sie es mit den dazu angebotenen Programmen aus dem Disketten-Service. Den PC beherrschen mit

- **DOS-EXTRA-Sonderheften**
- **DOS-Taschenbüchern**
- **DOS-Disketten-Service**

Zur Erinnerung:
Meine DOS-Sonderhefte, DOS-Taschenbücher, DOS-Disketten wurden bestellt am

Meine Anschrift

Name / Firma

Vorname / Ansprechpartner

Straße, Haus-Nr.

PLZ, Ort

Unterschreiben Sie hier bitte Ihre Bestellung!
Bei Minderjährigen ist die Unterschrift eines gesetzlichen Vertreters erforderlich. Ohne Ihre Unterschrift kann die Bestellung nicht bearbeitet werden.

✗

Datum Unterschrift

Bitte unterschreiben Sie auch Ihre Sicherheitsgarantie, mit der Sie folgendes zur Kenntnis nehmen: Sie haben das Recht, Ihr angefordertes Werk innerhalb von 10 Tagen ab Lieferung an den INTEREST-VERLAG GmbH, Römerstraße 16, W-8901 Kissing, zurückzusenden, wobei für die Fristwahrung das Datum der Absendung genügt. Sie kommen dadurch von allen Verpflichtungen aus der Bestellung frei.

✗

Datum Unterschrift

IN 110 427

60 Pfennig, die sich lohnen!

Postkarte / Antwort

INTEREST-VERLAG

Fachverlag für Special Interest
Publikationen und Anwendersoftware
z. Hd. Herrn Michel

Römerstraße 16

W-8901 Kissing

Verlagsgarantie

- Sie erhalten von mir PC-Fachinformationen, von absoluten Spezialisten für Sie geschrieben.
- Ihr Nachschlagewerk können Sie in aller Ruhe 10 Tage zu Hause prüfen und bei Nichtgefallen innerhalb dieser Frist zurücksenden. Der INTEREST-VERLAG gewährt nach Zahlung des Rechnungsbetrages das Recht, die beiliegende Software zu testen und zeitlich unbeschränkt zu nutzen.
- Ihr Nachschlagewerk ist immer aktuell. Dafür sorgt unser Erweiterungsservice.
- Den Erweiterungsservice können Sie jederzeit kündigen. Sie gehen also kein Risiko ein. Darauf gebe ich Ihnen mein Wort.

Ihr INTEREST-VERLAG
Fachverlag für Special Interest
Publikationen und Anwendersoftware

W. Michel

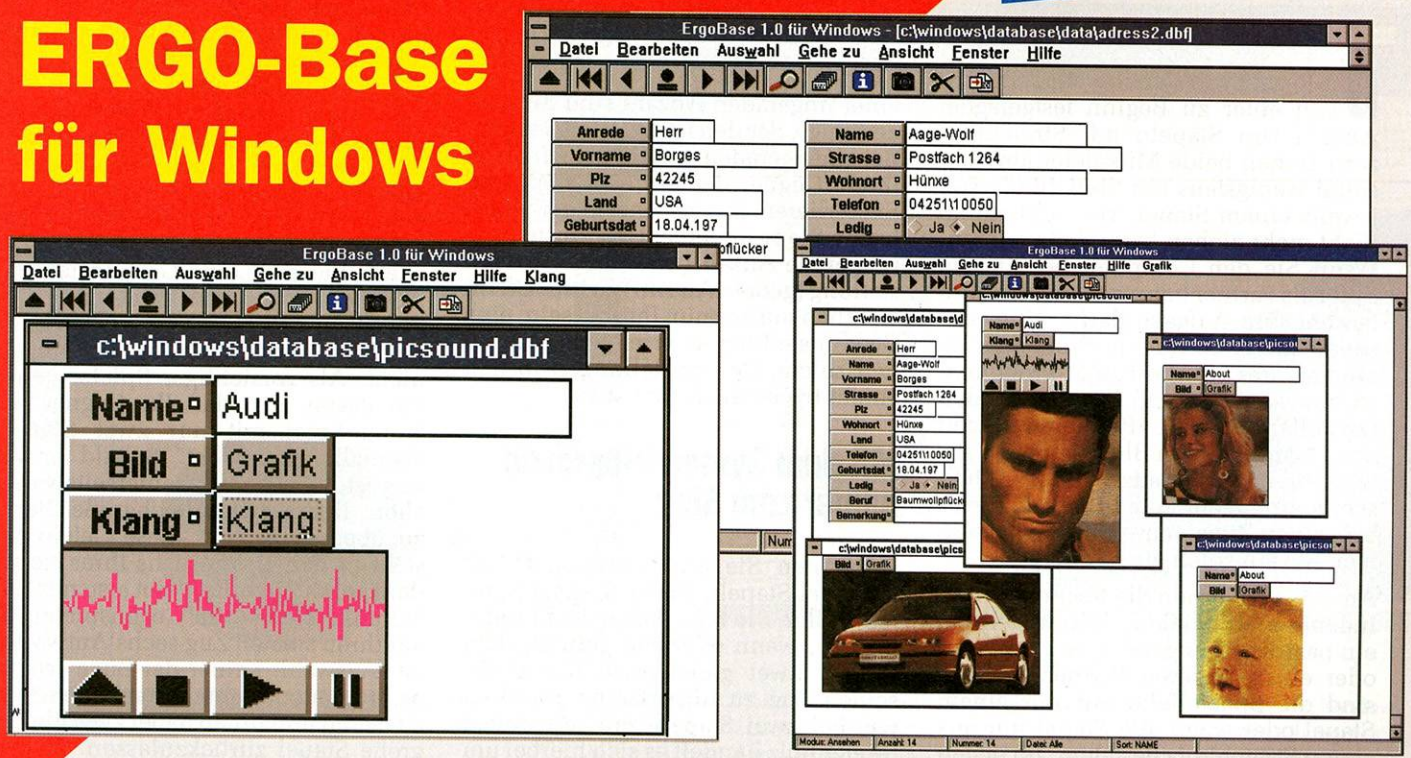
Wolfgang Michel
Verlagsleiter

Unglaublich!

Datenbank mit Bild und Ton

Fantastische Windows-Leistung zum Superpreis

ERGO-Base für Windows



Ton und Bild machen Ihre Datenbank "dreidimensional"

Jetzt ist es endlich möglich: Ihre Adressen, Briefmarken, Videos oder CDs in einer Datenbank mit integrierten Bildern und Ton. Ihre Daten erreichen bisher unmöglich geglaubte Aussagekraft. ***Praktikables Multimedia zum erschwinglichen Preis.***

ERGO-Base für Windows bietet aber auch das Leistungsspektrum dBase-kompatibler Datenbanken für den professionellen Einsatz, wie zum Beispiel als Personaldatenbank mit Paßbildern und Sprechprobe. ERGO-Base hat durch "C-Baum-Technik" auch die "Power" für Datenbanken, die Hunderte von Megabyte groß sind.

Sicherheit und leichte Einarbeitung

ERGO-Base ist übersichtlich, weil es Sie immer nur mit der Funktionsauswahl "behelligt", die in Ihrer aktuellen Arbeitssituation auch sinnvoll ist. Möchten Sie z.B. mit Grafiken arbeiten, werden Ihnen die entsprechenden Menübefehle zur Verfügung gestellt. Wenn Sie zur normalen Datenbearbeitung zurückwechseln, verschwindet das überflüssig gewordene Grafikmenü. ERGO-Base hält Ihnen den Blick auf das für Sie Wesentliche frei.

Individuelle Gestaltung und intuitive Bedienung

ERGO-Base verschont Sie mit Fachbegriffen wie "Eingabemodus" oder "Maskenmodus". Die Gestaltung und das Aussehen Ihrer Datenbank können Sie jederzeit ändern, unabhängig davon, was Sie gerade mit der Datenbank machen. Einfaches "Anfassen" und "Verschieben" mit der Maus genügt – ERGO-Base für Windows macht "Einschalten und Loslegen" wahr!

Etiketten, Listendruck und Serienbriefe par excellence

Mit ERGO-Base sind Sie im Handumdrehen in der Lage, sowohl ansprechende Listen (Reports) zu erstellen als auch Etiketten zu generieren. Zur Gestaltung der Etiketten steht Ihnen ein komplettes Programm-Modul ***mit DTP-Funktionen*** zur Verfügung, mit dem Sie auch Geburtstagskarten oder Miniposter erstellen können. Durch die Serienbriefanbindung werden ERGO-Base und Ihre Textverarbeitung zu einem unschlagbaren Team.

Schon eingebaut: Unterstützung für Word für Windows, AmiPro und WordStar für Windows

ERGO-Base für Windows im Überblick

● Hoher Bedienungskomfort ● Extrem hohe Geschwindigkeit ● dBase-kompatibel ● Multimedia-Unterstützung mit Sound und Grafik ● Über 2 Milliarden Datensätze, 255 Felder pro Datensatz ● Feldtypen-Text, numerisch, Gleitpunkt, Datum, Logik, Memo, Grafik und Sound ● Unbegrenzte Datenmengen bei Memo, Grafik und Sound ● Leistungsfähige Such- und Reportfunktionen ● Speicherplatzersparnis durch Kompression ● Beliebige, variable Eingabemasken ● Serienbriefanbindung und Ettikettendruck ● DTP-Funktionen zur Gestaltung von Etiketten, Karten oder Minipostern ● Kompatibilität mit allen gängigen Windows-Textverarbeitungen

ERGO-Base für Windows nur DM **199,95***

ICON-Designer für Windows **49,80***

* Unabhängig von der Anzahl der bestellten Produkte berechnen wir für das Inland DM 4,— bzw. für das Ausland DM 6,— Porto und Verpackung. Bitte benutzen Sie die Bestellkarte im Heft.

Word für Windows, AmiPro, WordStar für Windows, dBase sind Warenzeichen der jeweiligen Hersteller.

Händleranfragen erwünscht

CeBIT'93 HANNOVER
24. — 31. 03. 1993
Halle 7. Stand E13/E19

DMV-Verlag • Postfach 250 • W-3440 Eschwege
Tel.: (0 56 51) 809-0 • Fax: (0 56 51) 809-430

DMV Daten- und Medienverlag

Das Nimm-Spiel

Wer zuletzt nimmt

Den Reigen der Strategiespiele in dieser Ausgabe eröffnet »Nimm«. Es wurde bereits um 1902 von C.L. Bouton erfunden und zählt heute zu den berühmtesten mathematischen Zweipersonenspielen.

Die Regeln von »Nimm« sind einfach: Von einer zu Beginn festgelegten Anzahl von Stapeln mit Streichhölzern ziehen beide Mitspieler abwechselnd wenigstens ein Streichholz von jeweils einem Stapel. Wer schließlich nicht mehr ziehen kann, hat verloren. Wenn Sie den Computer bei diesem Spiel als Partner benutzen möchten, so besteht Ihre Aufgabe darin, eine passende Taktik zu finden. Probieren Sie einmal das Turbo-Pascal-Programm »nimm.pas« (Listing) aus. Sie werden feststellen, daß mit »Probieren« gegen den Computer kein Blumentopf zu gewinnen ist. Er besitzt offensichtlich schon eine recht gute Strategie, die er bei seinen Zügen anwendet. Wie kommen Sie auf solch eine Strategie?

Am meisten lernen Sie über ein Spiel, indem Sie es spielen. Also wagen Sie ein paar Runden gegen den Computer oder einen anderen Partner. Einfach sind die Spezialfälle mit nur einem Stapel oder wenn alle Stapel nur aus einem Streichholz bestehen. Im ersten Fall, wenn nur ein Stapel existiert, nehmen Sie einfach den kompletten Stapel auf, der Nachfolger kann nicht ziehen, Sie haben gewonnen. Im anderen Fall stehen Ihre Chancen nicht so eindeutig fest. Es kommt darauf an, wie viele Stapel noch existieren. Bei einer ungeraden Anzahl sind Sie stets derjenige, der den letzten Stapel räumt. Bei einer geraden Anzahl jedoch bleibt Ihrem Gegner der letzte Haufen, und Sie verlieren. Je nach Anzahl der Stapel haben Sie es mit einer Gewinnstellung (ungerade Anzahl) oder einer Verluststellung (gerade Anzahl) zu tun. Offensichtlich muß es nun Ihr Ziel sein, eine Gewinnstellung zu erreichen. Wie sehen solche Gewinnstellungen in den »nicht trivialen« Fällen aus?

Mit dem Symmetrieprinzip immer zum Sieg

Betrachten Sie ein Nimm-Spiel mit nur zwei Stapeln. Nach ein paar Spielen stellen Sie fest, daß es recht nützlich ist, wenn es Ihnen gelingt, dem Gegner zwei gleichgroße Stapel für seinen Zug zu hinterlassen. Ähnlich wie bei zwei Stapeln mit nur einem Streichholz handelt es sich hierbei um eine Gewinnstellung (beziehungsweise Verluststellung für Ihren Gegner). Wenn Ihr Gegenüber nämlich einen kompletten Stapel aufnimmt, so gewinnen Sie, indem Sie den anderen Stapel komplett aufnehmen.

Reduziert Ihr Gegenspieler einen Stapel hingegen nur um ein paar Hölzer, stellen Sie das Gleichgewicht wieder her, indem Sie vom anderen Stapel dieselbe Anzahl von Hölzern entfernen (Erhaltung der Symmetrieeigenschaft). Mit der Zeit erreichen Sie so den trivialen Spielstand, bei dem auf beiden Stapeln nur ein Streichholz liegt, und gewinnen dann mit Ihrem nächsten Zug.

Soweit ist das Nimm-Spiel noch überschaubar. Was passiert aber bei drei Stapeln? Bild 1 zeigt den Zuggraphen eines Nimm-Spiels, bei dem der erste Stapel aus vier, der zweite Stapel aus einem und der dritte Stapel aus zwei Streichhölzern besteht. Sie sind am Zug; die Pfeile deuten an, welche Züge sich Ihnen anbieten. Mit dem Wissen um die Beherrschung des Nimm-Spiels mit zwei Stapeln versuchen Sie natürlich zunächst einen Zug, der dem Gegner nur zwei gleichgroße Stapel hinterläßt. Mit solch einem Zug sichern Sie sich eine Gewinnstellung. Aber ein Blick auf Ihre möglichen Züge bringt Ernüchterung, denn so einen Zug erlauben die Spielregeln nicht. Als vorsichtiger Spieler gehen Sie davon aus, daß Ihr Gegner das Nimm-Spiel mit nur zwei Stapeln ebenfalls beherrscht. Sobald er am Zug ist, wird er also seinerseits versuchen, Ihnen zwei gleichgroße Stapel zu übergeben. Und diese Gefahr besteht in den ersten sechs Ihrer sieben denkbaren Züge. Es bleibt also nur Zug Nummer sieben. Ihr Gegenspieler hat auf Ihren siebten Zug sechs Antworten zur Auswahl. Keiner der Züge erlaubt es ihm jedoch, einen Stapel komplett zu leeren und Ihnen dabei zwei gleichgroße Stapel zurückzulassen. Im Gegenteil: Zu jedem seiner Züge existiert ein Folgezug Ihrerseits, der das Spiel durch Erreichen einer Gewinnstellung entscheidet. Sie werden sich also (ausschließlich) mit dem siebten Zug auf die Siegerstraße begeben.

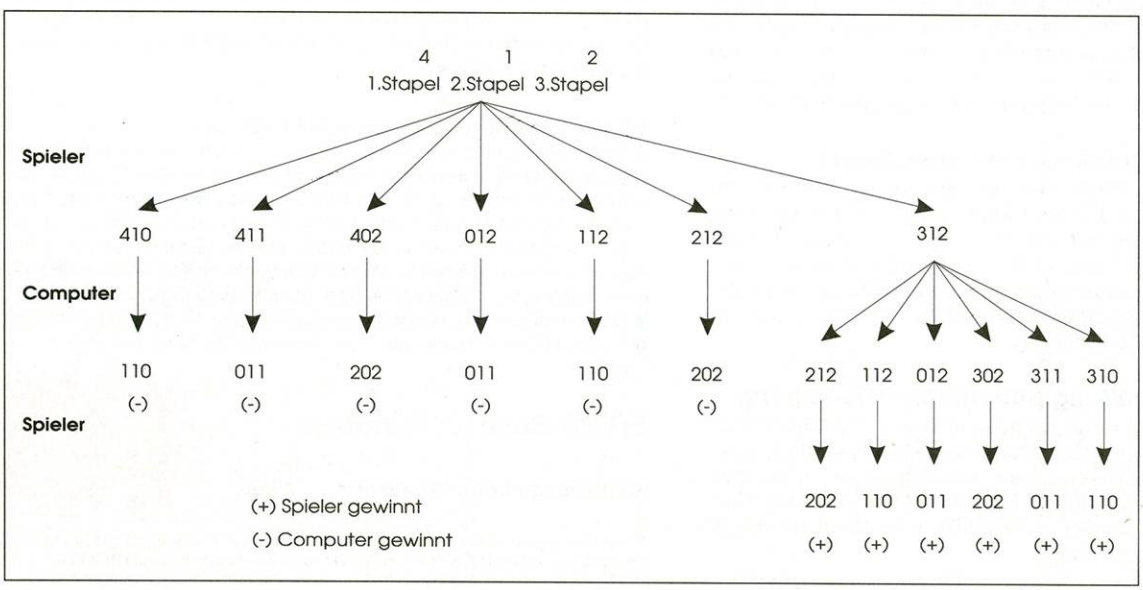


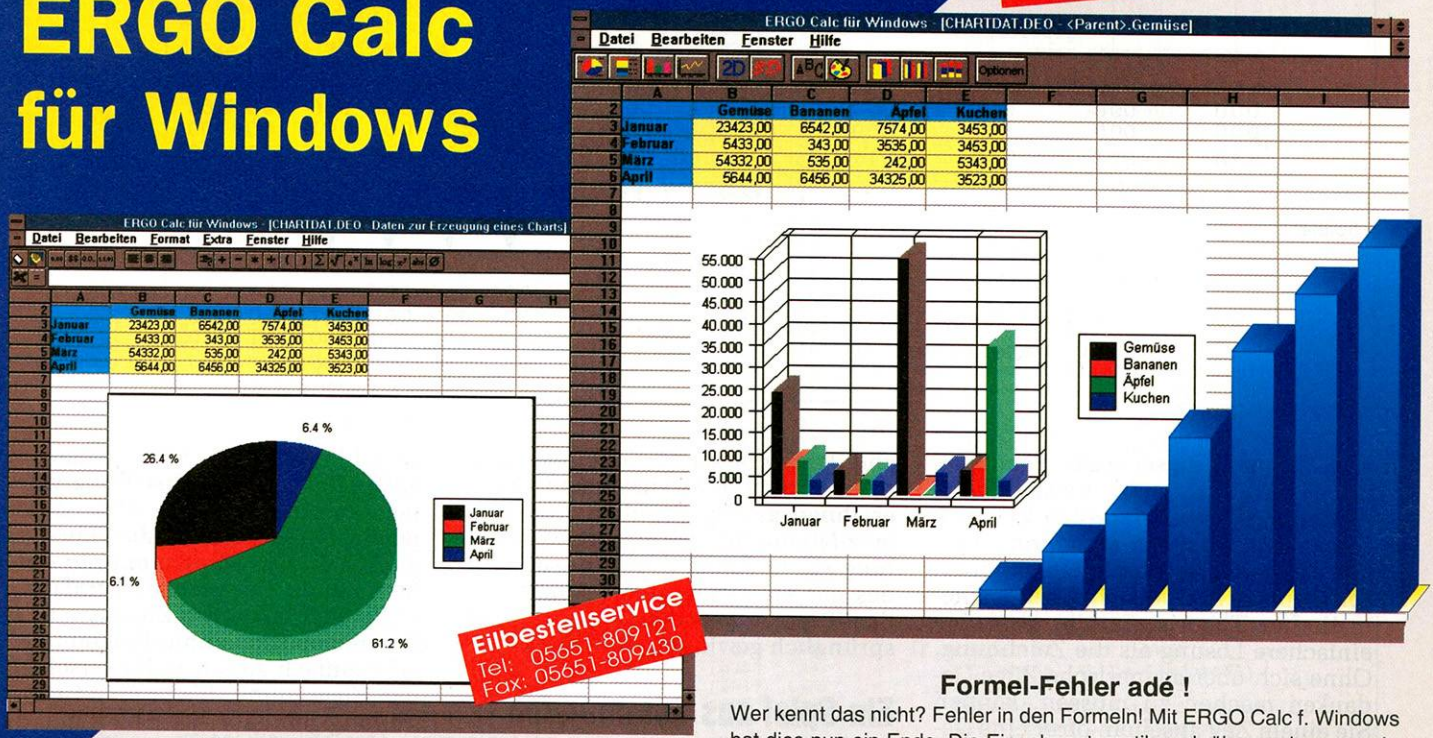
Bild 1. Bei drei Stapeln sind einige Überlegungen notwendig, um den besten Zug zu finden.

Sensationell

Aussagekräftige Kalkulation mit Farbe und Form

Tabellenkalkulation und Präsentation mit sensationeller Leistung zum Top-Preis

ERGO Calc für Windows



Eilbestellservice
Tel: 05651-809121
Fax: 05651-809430

ERGO-Calc f. Windows ist Ihr Werkzeug für den Umgang mit Zahlen jedweder Art. Ob Sie Ihre Haushalts-, Urlaubs- oder Bausparkosten planen, Ihre Steuererklärung tätigen oder Ihre Vereinsabrechnung machen wollen, ERGO Calc f. Windows stellt Ihre Berechnungen auf eine solide Basis.
Auch im beruflichen Bereich ist ERGO Calc f. Windows eine unentbehrliche Unterstützung, die Sie bald nicht mehr missen möchten. Erstellen Sie mit ERGO Calc f. Windows Ihre Betriebsplanung, Angebote oder auch Fahrtkostenabrechnung ohne Probleme in Rekordzeit!

Leichte Einarbeitung und Handhabung

Durch die übersichtliche Oberfläche mit verschiedenen Sinnbildern, die in einer jederzeit zugängliche Liste angeordnet sind, ist das Bedienen von ERGO Calc für Sie eine Leichtigkeit. Kein langes Einarbeiten oder Wälzen von Handbüchern, statt dessen finden Sie eine intuitiv zu bedienende und logische Benutzerführung vor.

Ausdrucksstarke Kalkulationen mit Farbe und Grafik

Mit ERGO Calc f. Windows geben Sie Ihren Kalkulationen endlich den besonderen Schliff. Heben Sie wichtige Zahlen, Ergebnisse oder Informationen durch Farbe, verschiedene Schrifttypen und -größen hervor! Für professionelle Präsentationen fügen Sie aus Ihren Zahlen erzeugte Grafiken in Ihre Tabellen ein. Sie markieren mit der Maus das gewünschte Zahlenmaterial, klicken das Diagramm-Symbol an und fertig ist Ihre Grafik. Je nach Bedarf wählen Sie bei der Darstellung Ihrer Grafiken zwischen 2D- oder 3D-Grafiken und zusätzlich zwischen verschiedensten Charttypen.

Formel-Fehler adé !

Wer kennt das nicht? Fehler in den Formeln! Mit ERGO Calc f. Windows hat dies nun ein Ende. Die Eingabeautomatik und -überwachung macht Fehler in den Formeln zu einer Seltenheit. Sie klicken selbst die kompliziertesten Formeln einfach mit der Maus zusammen.

Mehrzeilige Beschriftungen leichtgemacht

Schluß mit dem Zwang zu einzeiligen Texten in den Tabellenzellen. Mit ERGO Calc f. Windows können Sie jetzt in einer Zelle ohne Probleme mehrzeilige Texte unterbringen. So können Sie z.B. den Text "Gesamtumsatz in DM" auf einfachste Weise zweizeilig darstellen. Kein langes Überlegen, Knobeln, Jonglieren oder Ausrichten des Textes mehr, dank ERGO Calc.

Die wichtigsten Merkmale auf einen Blick :

● Leichte Handhabung ● Formel-Eingabeautomatik und -überwachung ● alle wichtigen math. Funktionen ● einfache grafische Umsetzung von Zahlenmaterial ● 2D und 3D Grafiken ● Grafiken einfach und schnell in Tabellen einbettbar ● farbige Markierung zur optischen Hervorhebung in Tabellen ● Formatierung mit verschiedensten Zeichensätzen, Schriftgrößen, Ausrichtungen und Zahlenformaten ● Mini-Editor mit Textumbruch innerhalb einer Tabellenzelle für mehrzeilige Texte ● Tabellengröße nur vom Hauptspeicher begrenzt ● Verknüpfte Tabellen für größere Kalkulationsprojekte ● Ausführliches Hilfesystem ● einfachste Übernehme von Tabellen in gängige Windows-Textverarbeitungen

ERGO Calc f. Windows

nur DM **199,95***

ERGO Base f. Windows DM 199,95*

* Unabhängig von der Anzahl der bestellten Produkte berechnen wir für das Inland DM 4,- bzw.für das Ausland DM 6,- Porto und Verpackung.

DMV-Verlag • Postfach 250 • W-3440 Eschwege

ERGO Base/Calc das Paket zum Superpreis von nur DM 299,95*
Bitte benutzen Sie die Bestellkarte im Heft.

DMV Daten- und Medienverlag

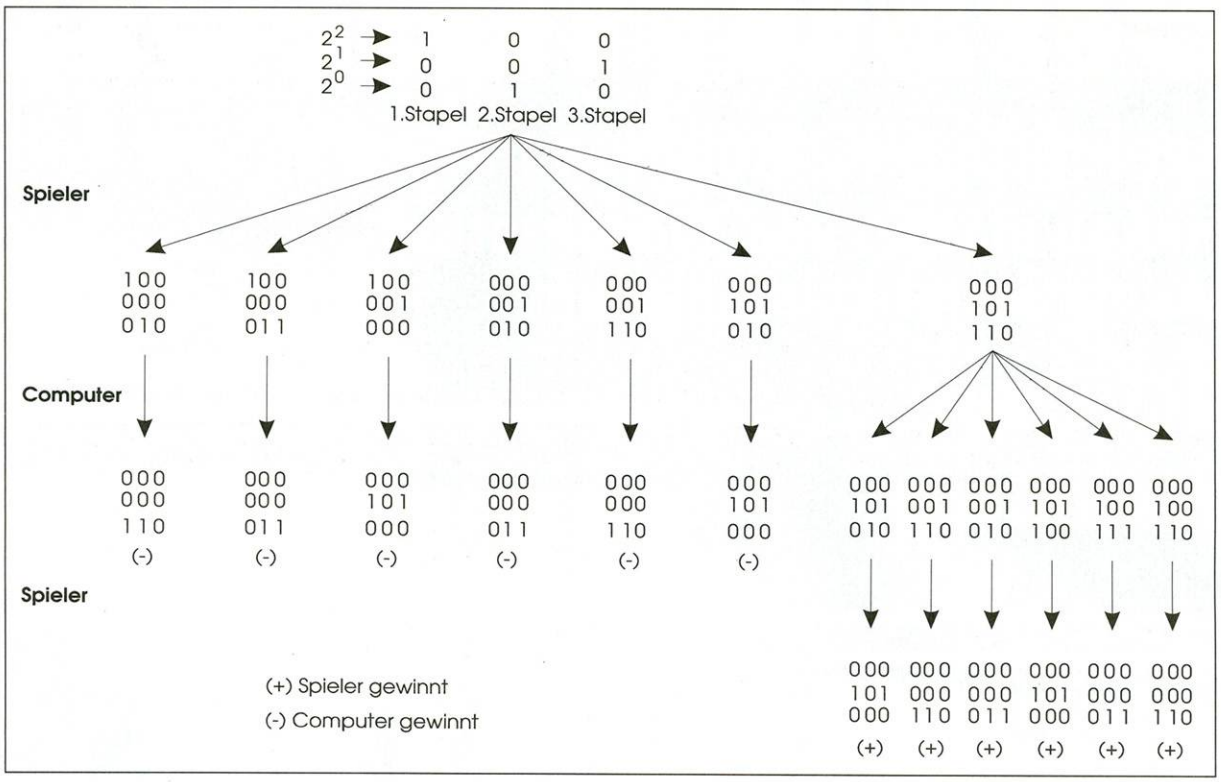


Bild 2. Entsprechend den Überlegungen aus Bild 1 finden Sie aufgrund der Darstellung im Binärsystem schnell eine einfache Strategie.

Wieso gerade dieser Zug? Benötigen Sie jedes Mal eine umständliche Zeichnung, bevor Sie einen Zug machen? »Nimm« ist ja nicht von ungefähr eines der berühmtesten Zweipersonenspiele mit vollständiger mathematischer Theorie. Es gibt also eine einfachere Lösung als die Zeichnung. Ohne sich über gegnerische Züge Gedanken machen zu müssen, können Sie einem Zug ansehen, ob er eine Gewinnstellung bringt oder nicht. In seinem Buch »Spiel, Idee und Strategie« führt Rüdeger Baumann ein komplexes Nimm-Spiel auf eines mit nur einem Haufen zurück und kommt so zur allgemeinen Lösungsstrategie. Insgesamt ist der beschriebene Weg jedoch immer noch recht kompliziert, und der erwünschte Aha-Effekt bleibt aus.

Aber es geht auch anschaulich! Besinnen Sie sich noch einmal auf das bereits Bekannte. Nimm-Spiele aus Stapeln mit nur einem Streichholz beherrschen Sie bereits. Was liegt folglich näher, als das allgemeine Nimm-Spiel auf triviale Nimm-Spiele zurückzuführen. Ideal wäre ein System aus mehreren Nimm-Spielen, die aus jeweils nur einelementigen Stapeln bestehen, die in ihrer Gesamtheit das ursprüngliche Nimm-Spiel repräsentieren. So kompliziert die Forderung klingen mag, diese Reduktion führt Ihr Computer ohnehin automatisch aus, indem er nämlich jede Anzahl von Streichhölzern intern im Binärsystem darstellt, das nur aus Einsen und Nullen besteht. Jede Zahl besteht im Binärsystem aus einer Summe von Zweierpotenzen. Jeweils Summanden gleicher Potenz fassen Sie zu einem separaten Nimm-Spiel zusammen. Wenn es Ihnen gelingt, in allen Zweierpotenz-Nimm-Spielen eine Gewinnstellung zu erreichen, so gewinnen Sie jedes der einzelnen Nimm-Spiele und damit auch jenes, welches Sie ursprünglich gewinnen wollten.

Ein Spiel aus Summanden gleicher Potenz

Betrachten Sie den eben diskutierten Fall aus Bild 1 noch einmal in Bild 2. Dieses Mal ist jede Zahl im Binärsystem dargestellt. Daher erkennen Sie sofort, daß der siebte Zug der einzige ist, bei dem alle Zweierpotenz-Nimm-Spiele auf Gewinnstellung stehen. Denn in jedem dieser Spiele existiert eine gerade Anzahl von Stapeln mit nur einem Streichholz. Das ist der Trick.

Binärzahlen können »Exklusiv-Oder« (kurz: »XOR«) miteinander verknüpft werden. Bei dieser Verknüpfung zweier Zahlen werden alle Zweierpotenzen verglichen; nur wenn die Zweierpotenz genau in der einen oder genau in der anderen Zahl enthalten ist, enthält auch das Ergebnis diese Zweierpotenz. Tritt eine Zweierpotenz aber in beiden Zahlen auf, so fehlt sie im Ergebnis. Diese Verknüpfung läßt sich also hervorragend zur Überprüfung auf eine Gewinnstellung verwenden. Denn wenn Sie alle Stapel »Exklusiv-Oder« verknüpfen und anschließend das Ergebnis Null ist, bedeutet dies, daß alle Zweierpotenzen in gerader Anzahl auftraten. Und genau dies entspricht unserer Vorstellung von einer Gewinnstellung: Alle Zweierpotenz-Nimm-Spiele haben eine gerade Anzahl von Stapeln mit nur einem Streichholz. Entsprechend einfach gestaltet sich auch die Zugauswahl des Computers im Nimm-Programm. Er geht einfach so lange die verschiedenen Züge durch, bis er eine Gewinnstellung gefunden hat. Sollte ihm dies nicht gelingen, wenn er sich also selbst in einer Verluststellung befindet, macht er einen zufällig ausgewählten Verlegenheitszug. Es bleibt ihm nur zu »hoffen«, daß sein Gegenspieler einen Fehler macht und er selbst anschließend eine Gewinnstellung findet.

Einen Nachteil haben solche Spiele mit vollständiger Theorie allerdings. Wenn Sie das Geheimnis einmal gefunden haben, eignet sich das Spiel bestenfalls noch dafür, Uneingeweihte zu beeindrucken. Das Spiel gegen den Computer verliert seinen Reiz, da er in sehr kurzer Zeit jede Gewinnstellung findet. Dank der Vollständigkeit des Algorithmus bleiben Ihnen dann auch keine Trickzüge mehr, mit denen Sie das Blatt wenden könnten, denn der Computer ist »voll informiert«. Bis zur Auffindung einer Theorie bleiben solche Spiele jedoch interessant; das Entdecken der Strategie kann unter Umständen sogar noch spannender sein als das Spiel selbst. Erkennen lassen sich solche Spiele meist daran, daß Sie kaum eine Chance haben, auch einmal zu gewinnen.

Bevor Sie jetzt enttäuscht schnell weiterblättern – wer will schon immer verlieren – sei Ihnen versichert, daß »Nimm« das erste und letzte vollinformierte Spiel in dieser Strategiespiel-Reihe ist. Anders sieht es da schon bei Reversi oder Schach in den folgenden Beiträgen aus, die nicht mehr geschlossen lösbar sind, komplizierte Strategien erfordern, dafür aber sehr viel länger interessant bleiben.

(Frank Höppner/bm)

```
[KF3]    1: {Programm: nimm.pas
[LE3]    2:  Funktion: Nimm-Spiel
[G64]    3:  Sprache:  Turbo Pascal 6.Ø
[K34]    4:  Autor:    Frank Höppner
[585]    5:  (c) 1992 DMV Widuch GmbH & Co. KG }
[ØØØ]    6:
[2A2]    7: program nimm;
[STØ]    8: type
[H15]    9:   tspieler    = (computer,mensch);
[UH3]   1Ø:   tspielstand = record
[2B5]   11:     stapel : array[1..9] of integer;
[HY2]   12:     AmZug  : tspieler;
[ZNØ]   13:   end;
[DZ6]   14: procedure Start(var stand:tspielstand);
[FB6]   15:   var anzstapel,maxstapel,i : integer;
[YE3]   16:       s                     : string;
[MUØ]   17:   begin
[XF1]   18:     writeln;
[XF1]   19:     writeln;
[1M2]   2Ø:     writeln('Nimm');
[EA2]   21:     writeln('----');
[XF1]   22:     writeln;
[CG6]   23:     writeln('Ziehen Sie abwechselnd Str'+
[196]   24:       'eichhölzer von den Stapeln, bis');
[726]   25:     writeln('keine Hölzer mehr da sind.'+
[Z35]   26:           ' Wer nicht mehr ziehen kann');
[6F3]   27:     writeln('verliert.');
[686]   28:     writeln('Ein Zug besteht in der Ein'+
[9W4]   29:           'gabe der Nummer des Feldes,');
[866]   3Ø:     writeln('auf das Sie ziehen wollen.'+
[UN4]   31:           ' Die Felder sind von links');
[156]   32:     writeln('oben nach rechts unten dur'+
[1R2]   33:                          'chnummeriert.');
[XF1]   34:     writeln;
[F26]   35:     write('Mit wievielen Stapel wollen '+
[563]   36:                     'Sie spielen? (2-9)');
[X13]   37:     readln(anzstapel);
[LF5]   38:     if anzstapel>9 then anzstapel:=9;
[LE5]   39:     if anzstapel<2 then anzstapel:=2;
[656]   4Ø:     write('Wieviele Hölzer maximal pro '+
[YR2]   41:                       'Stapel? (15-15Ø)');
[913]   42:     readln(maxstapel);
[CZ5]   43:     if maxstapel>15Ø then maxstapel:=15Ø;
[7S5]   44:     if maxstapel<15 then maxstapel:=15;
[NØ6]   45:     for i:=1 to anzstapel do with stand
[9Y5]   46:      do stapel[i]:=RANDOM(maxstapel-5)+5;
[654]   47:     for i:=anzstapel+1 to 9 do
[DD3]   48:      stand.stapel[i]:=Ø;
[HC5]   49:     write('Wer beginnt? (s=Spieler)');
[LH1]   5Ø:     readln(s);
[VM2]   51:     if (s='s') or (s='S')
[X64]   52:       then stand.AmZug:=mensch
[HW4]   53:       else stand.AmZug:=computer;
[ZNØ]   54:   end;
[6F6]   55: procedure Anzeige(stand:TSpielstand);
[8X2]   56:   var i,j : integer;
[MUØ]   57:   begin
[XF1]   58:     writeln;
[DV4]   59:     for i:=1 to 9 do with stand do
[863]   6Ø:      if stapel[i]>Ø then
[MUØ]   61:       begin
[VH3]   62:         write('Stapel #',i:1,' (',
[GW2]   63:           stapel[i]:3,') : ');
[4K4]   64:         for j:=1 to stapel[i] div 3
[W62]   65:           do write(#24Ø);
[PV3]   66:         case stapel[i] mod 3 of
[4E2]   67:          1 : writeln('-');
[PF2]   68:          2 : writeln('=');
[5Y1]   69:          Ø : writeln;
[ZNØ]   7Ø:         end;
[ZNØ]   71:       end;
[XF1]   72:     writeln;
[ZNØ]   73:   end;
[H13]   74: function GameOver
[YW4]   75:                (stand:TSpielstand):boolean;
[2P2]   76:   var i : integer;
[MUØ]   77:   begin
[UPØ]   78:     i:=Ø;
[P31]   79:     repeat
[951]   8Ø:       Inc(i);
[NN5]   81:     until (i=9) or (stand.stapel[i]>Ø);
[MY4]   82:     gameover:=(stand.stapel[i]=Ø)
[ZNØ]   83:   end;
[UN3]   84: procedure SpielerZug
[Y34]   85:                    (var stand:TSpielstand);
[LS4]   86:   var stapel,anzahl : integer;
[MUØ]   87:   begin
[P31]   88:     repeat
[7K5]   89:       write('Von welchem Stapel wollen '+
[KF2]   9Ø:                            'Sie ziehen?');
[ZG2]   91:       readln(stapel);
[F65]   92:     until stand.stapel[stapel]>Ø;
[P31]   93:     repeat
[G26]   94:       write('Wieviele Hölzer möchten Si'+
[UU1]   95:                              'e nehmen?');
[LG2]   96:       readln(anzahl);
[VE6]   97:     until stand.stapel[stapel]>=anzahl;
[NS5]   98:     Dec(stand.stapel[stapel],anzahl);
[NX3]   99:     stand.AmZug:=computer;
[ZNØ]  1ØØ:   end;
[LU3]  1Ø1: procedure ComputerZug
[Y34]  1Ø2:                    (var stand:TSpielstand);
[LS4]  1Ø3:   var stapel,anzahl : integer;
[6Ø4]  1Ø4:       zug           : TSpielstand;
[4A3]  1Ø5:       found         : boolean;
[ØØØ]  1Ø6:
[X94]  1Ø7:   function GewinnStellung
[554]  1Ø8:                    (s:TSpielstand):boolean;
[8X2]  1Ø9:     var i,j : integer;
[MUØ]  11Ø:     begin
[EP2]  111:       j:=s.stapel[1];
[H22]  112:       for i:=2 to 9
[T44]  113:         do j:=j xor s.stapel[i];
[6S3]  114:       gewinnstellung:=(j=Ø);
[ZNØ]  115:     end;
[MUØ]  116:   begin
[NP5]  117:     stapel:=1; anzahl:=Ø; found:=false;
[P31]  118:     repeat
[RH5]  119:       {vom alten Spielstand ausgehend}
[132]  12Ø:       zug:=stand;
[ØØØ]  121:
[9U3]  122:       { nächsten Stapel oder
[9H4]  123:         nächste Anzahl wählen }
[XC5]  124:       if stand.stapel[stapel]>anzahl
[GP2]  125:         then Inc(anzahl)
[WP1]  126:         else begin
[LZ1]  127:                Inc(stapel);
[GG1]  128:                anzahl:=1
[ZNØ]  129:              end;
[ØØØ]  13Ø:
[FC5]  131:       { ist die Wahl ein korrekter Zug,
[UG5]  132:         dann Zug erstellen und prüfen, ob
[FL5]  133:         es sich um Gewinnstell. handelt }
[1Ø5]  134:       if (stapel<1Ø) and (anzahl>=1) and
[H35]  135:          (zug.stapel[stapel]>=anzahl)
[ZN1]  136:         then begin
[GH5]  137:           Dec(zug.stapel[stapel],anzahl);
[1N4]  138:           found:=Gewinnstellung(zug)
[ZNØ]  139:         end;
[2C4]  14Ø:     until (stapel=1Ø) or found;
[KR5]  141:     { wenn keine Gewinnstellung einen
[M25]  142:       Verlegenheitszug erfinden }
[PT3]  143:     if not found then begin
[P31]  144:       repeat
[133]  145:         stapel:=random(9)+1
[F65]  146:       until stand.stapel[stapel]>Ø;
[ED1]  147:       anzahl:=
[SF5]  148:         random(stand.stapel[stapel])+1;
[ZNØ]  149:     end;
[7A3]  15Ø:     { Zug ausführen }
[NS5]  151:     Dec(stand.stapel[stapel],anzahl);
[DF3]  152:     stand.AmZug:=mensch;
[S13]  153:     { Zug mitteilen }
[TF5]  154:     writeln('Ich nehme vom ',stapel:1,
[H15]  155:       '. Stapel ',anzahl,' Hölzer.');
[ZNØ]  156:   end;
[3V2]  157: procedure spiel;
[L15]  158:   var spielstand : tspielstand;
[MUØ]  159:   begin
[243]  16Ø:     start(spielstand);
[P31]  161:     repeat
[4H3]  162:       anzeige(spielstand);
[A74]  163:       case spielstand.AmZug of
[VG5]  164:        mensch  :SpielerZug(spielstand);
[526]  165:        computer:ComputerZug(spielstand);
[ZNØ]  166:       end;
[EN4]  167:     until GameOver(spielstand);
[XF1]  168:     writeln;
[A74]  169:     case spielstand.AmZug of
[CS4]  17Ø:      computer: write('Sie haben');
[R74]  171:      mensch  : write('Ich habe');
[ZNØ]  172:     end;
[9P3]  173:     writeln(' gewonnen!!!');
[ZNØ]  174:   end;
[MUØ]  175: begin
[RWØ]  176:   spiel
[KNØ]  177: end.
```

Nur wenn Sie das Geheimnis der Nimm-Spiele kennen, haben Sie eine Chance gegen dieses Nimm-Programm

Tic Tac Toe

Der Tac-Tic auf der Spur

Auf dem Weg zu komplexen Strategiespielen wie Reversi oder Schach gilt es zunächst einige grundlegende Verfahren zu finden, mit deren Hilfe der Computer zum würdigen Gegner wird. Das einfache Spiel »Tic Tac Toe«, auch bekannt als »Kringel und Kreuze«, eignet sich gut als Einstieg in die Geheimnisse der Strategie.

Es gibt wohl niemanden, der als Dreikäsehoch dieses Spiel nicht bis zum Abwinken auf Schmierzetteln oder beschlagenen Fensterscheiben gespielt hätte: Wer drei Felder in einer Reihe auf dem 3×3 Felder großen Spielfeld besetzt, hat gewonnen. Es stellt nicht gerade hohe Anforderungen an seine Spieler, aber jeder hat wohl seine ersten Strategiespiel-Erfahrungen mit »Tic Tac Toe« gemacht (zuerst die Mitte, dann die Ecken...!). Das Praktische an diesem Spiel ist seine Überschaubarkeit. Schließlich verläuft keine Partie über mehr als neun Züge. Damit fällt es leicht, nach wenigen Spielen allgemeine Regeln aufzustellen, die eine echte Entscheidungshilfe bedeuten. Genau dies ist auch die Arbeit, die Sie leisten müssen, wenn Sie Strategiespiele programmieren wollen.

Kein Wunder also, wenn dieses Spiel immer wieder als Paradebeispiel herhalten muß, wie zum Beispiel im Kinofilm »Wargames – Kriegsspiele«. Der (künstlich) intelligente Verteidigungscomputer Norad versucht sich hier am klassischen »Kringel und Kreuze«, bis er schließlich resümiert: »Ein seltsames Spiel. Die einzige Möglichkeit zu gewinnen liegt darin, nicht zu spielen«. Das ist zwar kein Lob für das Spiel, aber es wirft die Frage auf, wie Norad auf diese Aussage kommt.

Ohne dem Computer bereits Erfahrung mitzugeben, kann er bei einer Tic-Tac-Toe-Partie nur eines entscheiden: ob sie gewonnen, verloren oder noch nicht entschieden ist. Darüber geben die Spielregeln Auskunft. Ohne über sinnvolle oder unsinnige Züge weiter nachzudenken, können Sie den Computer einfach alle überhaupt denkbaren Partien einmal durchspielen lassen, um den Ausgang der Spiele zu bewerten. Eine Tic-Tac-Toe-Partie besteht aus maximal neun Zügen, also sind höchstens Neun-Fakultät ($9! = 9 \times 8 \times 7 \times 6 \times 5 \times 4 \times 3 \times 2 \times 1 = 362\,880$) verschiedene Partien denkbar.

Simulieren Sie einfach zwei Spieler, die von Tic Tac Toe keine Ahnung haben und folglich ihre Kringel und Kreuze ganz zufällig verteilen. Nach jedem Zug überprüfen Sie, ob die Partie entschieden ist oder nicht. Welche Informationen erhalten Sie damit?

Die Berechnung aller Züge dauert viel zu lange

Nachdem Sie bei einer bestimmten Spielstellung einen Spieler alle möglichen Züge haben durchführen lassen, können Sie feststellen, ob es einen Zug gibt, der bei dieser Stellung zum Sieg führt. Aus dem Beitrag über das Nim-Spiel kennen Sie sogenannte Gewinnstellungen, die Ihnen einen Sieg sichern. Gibt es solche Stellungen auch bei Tic Tac Toe? Aus Ihrer Erfahrung mit dem Spiel wissen Sie, daß es sie nicht gibt; bei Tic Tac Toe können Sie gegen einen aufmerksamen Gegner nicht gewinnen. Kann Ihr Programm diese Erfahrung auch machen?

Eine Gewinnstellung bedeutet per Definition, daß alle folgenden Züge zu Ihrem Sieg führen. Dabei kalkulieren Sie an keiner Stelle etwa eine Unaufmerksamkeit des Gegners ein, sondern es darf einfach überhaupt keine Alternative zu Ihrem Sieg bestehen. Die Suche nach solch einem Zug ist gleichzeitig die Strategie für Ihr Programm, um zu einem anspruchsvollen Computergegner zu kommen. Die Programmierung dieser Suche finden Sie im Programm »noradttt.pas« (Listing 1) in einer ersten Version.

Die Prozedur »suche« probiert nacheinander alle denkbaren Züge auf dem als Parameter übergebenen Spielbrett aus. Dazu setzt sie ein Kreuz an die Stelle des jeweiligen Zuges und läßt dann den Gegenspieler durch den Aufruf der Funktion »spielerwert« zum Zug kommen. Diese liefert als Ergebnis, ob die Partie unentschieden ausgeht oder wer gewinnt. Das stellt sie fest, indem sie ihrerseits alle verbliebenen Züge ausprobiert und – falls die Partie noch nicht beendet ist – mit der Funktion »computerwert« wieder den Computer ziehen läßt. Beide Funktionen rufen sich gegenseitig immer wieder auf, bis das Spiel zu Ende ist. Die Vielfalt der denkbaren

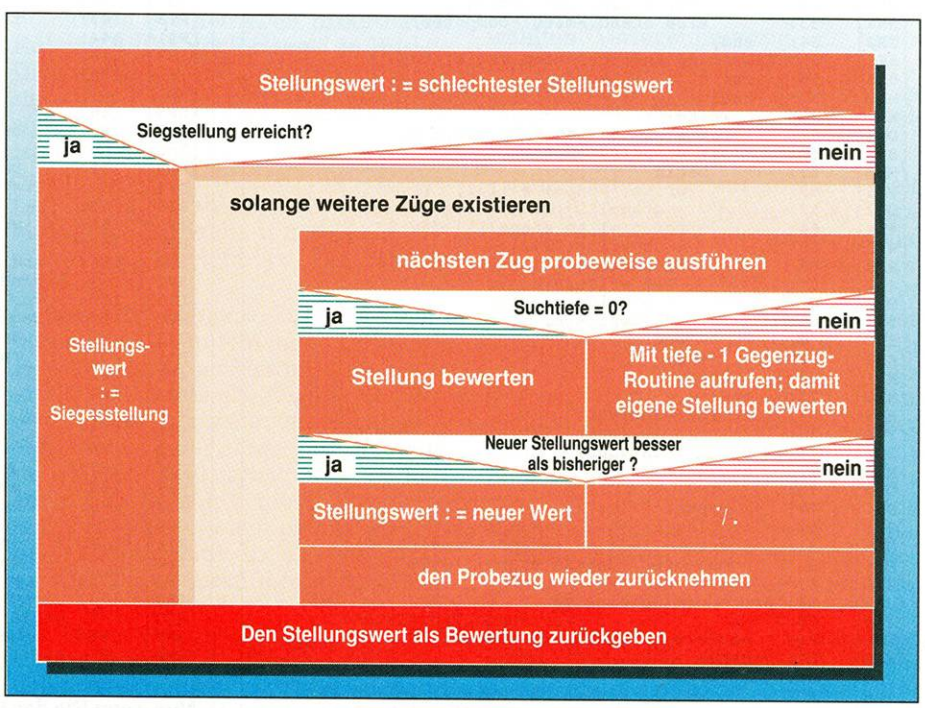


Bild 1. Nach dem Minimax-Verfahren ermitteln Sie den optimalen Gegenzug für Ihren Computerspieler.

Partieverläufe ergibt den Funktionswert für die Prozedur »suche«. Auf diese Weise finden Sie heraus, welches Ergebnis mit diesem Zug maximal zu erreichen ist. Wenn Sie das Programm starten und den Computer beginnen lassen, meldet das Programm zu jedem der neun Züge, daß keiner der beiden Spieler den Sieg sicher hat, also keine Gewinnstellung vorliegt. Diese Information bedeutet – schließlich hat das Spiel noch nicht einmal begonnen –, daß keiner gewinnen kann; wenn Sie also unbedingt gewinnen wollen, können Sie sich die Partie sparen. Soviel zu Norads Feststellung. Auch Ihren Erfahrungswert, zunächst in die Mitte zu setzen, finden Sie bestätigt, denn in diesem Fall ermittelt das Programm die meisten (theoretischen) Siege.

Zusammen mit der Feststellung, daß kein Zug echte Gewinnchancen enthält, gibt Ihnen das Programm die Anzahl der bisher erfolgten Bewertungen aus. Eine Bewertung entspricht dem Aufruf der Funktion »wert«, die anhand der Spielregeln entscheidet, ob jemand gewonnen hat oder nicht. Bis das Programm die Analyse der Anfangssituation abgeschlossen hat, hat es fast 600 000 Bewertungen vorgenommen. Entsprechend lange braucht das Programm, bis es diese Informationen auf den Bildschirm liefert. Bei einem so primitiven Spiel mit maximal neun Zügen dauert die Analyse etliche Sekunden. Wieviel Rechenzeit soll denn dann ein komplexes Spiel benötigen? Diese Frage ist berechtigt, denn Spiele wie Reversi oder gar Schach sind um einige Potenzen komplexer als Tic Tac Toe. Wer bringt jedoch die Geduld auf, einen Tag lang auf den Gegenzug eines Schachprogramms zu warten? Also müssen Sie versuchen, die Anzahl der vorgenommen Analysen zu verringern, und zwar drastisch.

Wie im Leben: Maximierung des eigenen Nutzens

● **Das Minimax-Theorem:** Zunächst einmal ist es überhaupt nicht nötig, wirklich sämtliche Züge durchzurechnen, zumal sich das Gesamtergebnis der Analyse dadurch nicht unbedingt verändert. Wenn Sie beispielsweise einen Zug finden, mit dem Sie die Partie gewinnen können, macht es keinen Sinn, die weiteren Züge dieser Stellung auszuprobieren. Mehr als gewinnen können Sie nicht. Wenn Sie am Ende der Repeat-Schleifen in den beiden Funktionen »computerwert« und »spielerwert« aus Listing 1 die ausgeklammerte Oder-Abfrage einsetzen, brechen die Routinen die Suche ab, sobald ein Zug die Partie entscheiden kann. Allein dadurch vermindert sich die Zahl der Bewertungen von 596 016 auf 105 121. Eine gute Ausbeute also!

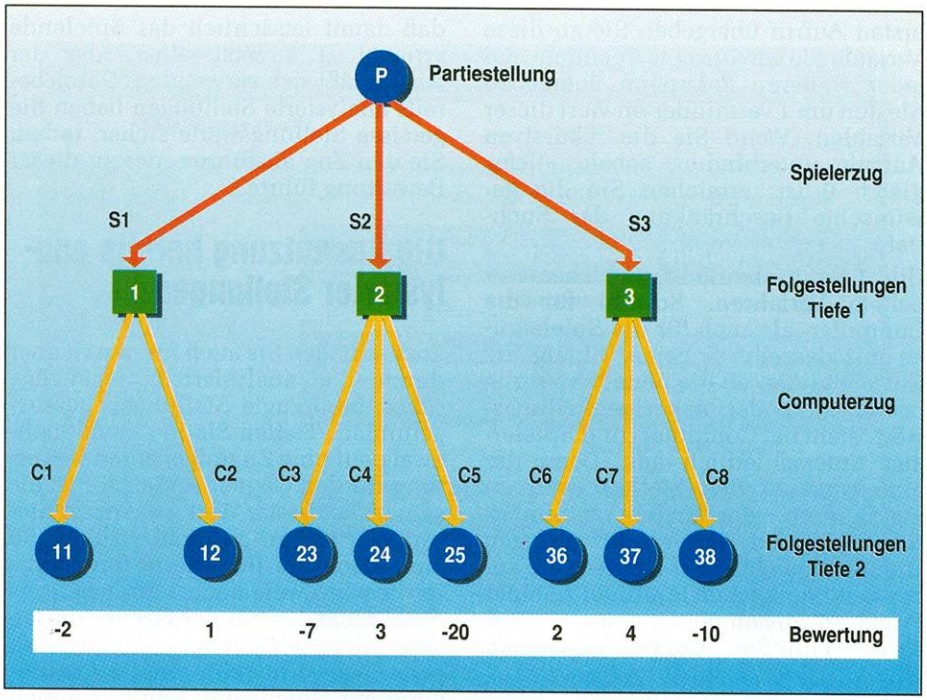


Bild 2. Bei der Eindämmung des Aufwands hilft Mitdenken. Die konsequente Ausnutzung der Minimax-Theorie führt zum Alpha-Beta-Algorithmus.

Die Annahme, daß der Gegner stets den Zug wählt, der seinen eigenen Nutzen maximiert, den Vorteil des Gegners hingegen minimiert, verhilft Ihnen in diesem Fall zu einer Reduzierung des Aufwandes um nahezu fünf Sechstel. In der Theorie der Strategiespiele hat sich dieses Verfahren unter dem Namen »Minimax-Theorem« etabliert.

Auf die richtige Bewertung kommt es wesentlich an

Nun stellt Tic Tac Toe auf dem Gebiet der Strategiespiele natürlich eine große Ausnahme dar, denn kaum ein Spiel hat eine so eng begrenzte Spieldauer von nur neun Zügen. Mit jedem weiteren Zug jedoch erhöht sich der Aufwand der erschöpfenden Suche bis zu Sieg oder Niederlage drastisch. Viele Spiele sind in der Anzahl ihrer Züge nahezu unbegrenzt. Da können Sie Ihr Programm nicht bis zum Spielende alle möglichen Züge ausprobieren lassen. Statt dessen beschränken Sie sich notgedrungen darauf, nur ein paar Züge vorauszublicken. Mit der bisherigen Bewertungsfunktion erzielen Sie dann jedoch keine guten Erfolge. Denn nach nur ein paar Zügen ist noch nicht damit zu rechnen, daß auf Sieg, Niederlage oder Patt entschieden werden kann. In den meisten Fällen wird das Spiel nicht einmal beendet sein. Sie müssen also eine feinfühligere Bewertung der jeweiligen Spielsituation entwerfen, die auch leichte Vorteile für einen Spieler erkennt.

Hier liegt die eigentliche Kunst der Strategiespiele-Programmierung. Denn nur die richtige Bewertung eines Spielstandes führt zu einem ausgereiften Programm. Wie könnte solch eine Bewertung für Ihr Tic Tac Toe aussehen? Lassen Sie Ihre Erfahrung mit dem Spiel einfließen: Züge in Zentrum- oder Eckpositionen honorieren Sie mit mehr Punkten als die Züge auf die Mittelpunkte der Seiten. Züge, nach denen zwei Steine in einer Reihe liegen und deren dritter Platz noch leer ist, werden mit einem zusätzlichen Bonus belohnt.

Diese wenigen Kriterien sind für eine verbesserte Bewertungsfunktion schon ausreichend. Selbstverständlich spielt es eine große Rolle, ob Sie oder Ihr Gegenspieler positionelle Vorteile auf dem Spielfeld erkämpfen. Bei nur zwei Mitstreitern bietet es sich an, dem Spieler positive Werte, dem Computer negative Werte zuzuordnen. Der Stellungswert ist die Summe beider Einzelwertungen. Als Spieler versuchen Sie den Stellungswert zu maximieren, der Computer versucht entsprechend zu minimieren. Getreu dem Minimax-Prinzip bedeutet eine Maximierung der eigenen Werte automatisch eine Minimierung der Computerbewertung (das heißt hohe Computerwerte). Das Programm »alphattt.pas« (Listing 2) zeigt am Beispiel der Funktion »wert«, wie Sie die einzelnen Faktoren wichten können. Um nun die Beschränkung der Zugsuche auf eine bestimmte Anzahl von Stellungen zu erreichen, führen Sie eine weitere Variable »tiefe« in den Kopf der Funktionen »computerwert« und »spielerwert« ein. Beim

ersten Aufruf übergeben Sie an diese Variable die gewünschte Suchtiefe. Bei jeder weiteren Rekursion übergeben Sie den um 1 verminderten Wert dieser Variablen. Wenn Sie die rekursiven Aufrufe unterbinden, sobald »tiefe« gleich 0 ist, erreichen Sie die gewünschte Beschränkung der Suchtiefe.

Bild 1 veranschaulicht das bisher erarbeitete Verfahren. Sowohl für die Computer- als auch für die Spielerzüge gilt dasselbe Schema. Einzig die Entscheidung, ob die neue Bewertung besser ist als der bisherige Stellungswert, sieht bei Computer- und Spielerzug unterschiedlich aus. Denn der eine versucht den Stellungswert zu maximieren, der andere minimiert ihn. Aufgrund dieser sehr geringen Unterschiede bietet es sich an, die ansonsten identischen Routinen zu einer Funktion zusammenzufassen – was in der Funktion »suche« geschehen ist.

Eine allgemein anwendbare Strategie wird entwickelt

Sie haben jetzt ein Grundschema für die Ermittlung von Zügen, das sich auf nahezu jedes Spiel anwenden läßt. Wie umfangreich ist jedoch die Suche? Nehmen Sie zum Beispiel den Fall an, daß es im Mittel jeweils 40 denkbare Folgezüge zu einer Spielsituation gibt (bei Schach eine realistische Anzahl). Die Tabelle zeigt die Entwicklung der Anzahl der zu bewertenden Stellungen in Abhängigkeit zur Suchtiefe. Bei einer Suchtiefe von fünf Zügen und der hohen Anzahl von 1000 Stellungsbewertungen pro Sekunde (!) benötigen Sie immer noch länger als einen Tag, um alle Stellungen zu bewerten. Die Zeit, die zur Ermittlung der möglichen Züge und zum Aufbau der Stellungen notwendig ist, bleibt dabei noch unberücksichtigt. Sie sehen schon, wie wichtig es ist, nicht alle Züge gleichwertig zu behandeln, sondern unsinnige Züge rasch zu erkennen. Dabei laufen Sie jedoch stets Gefahr, versehentlich einen nur auf den ersten Blick schlechten Zug zu vernachlässigen – das darf einem guten Computergegner nicht passieren.

● **Das Alpha-Beta-Verfahren:** Ein sehr hilfreicher Ansatz zur Reduzierung der nötigen Züge stellt der Alpha-Beta-Algorithmus dar. Unter dieser (nicht gerade aussagekräftigen) Bezeichnung versteckt sich eine Weiterentwicklung der ersten Verbesserung des Minimax-Prinzips. Sie erinnern sich, daß die Suche in einem Zweig abbrach, sobald die Partie gewonnen war, denn mehr kann ein Zug nicht bringen. Bei komplexen Spielen mit langer Spieldauer ist jedoch der Fall, daß damit tatsächlich das Spielende erreicht ist, äußerst selten. Aber der Ansatz läßt sich verwenden. Durch bereits analysierte Stellungen haben Sie gewisse Stellungswerte sicher, indem Sie den Zug ausführen, der zu dieser Bewertung führte.

Die Ausnutzung bereits analysierter Stellungen

Ebenso haben Sie auch für den Gegner durch die analysierten Gegenzüge schon bevorzugte Stellungen herausgefunden. Treffen Sie in einem Suchzweig auf eine Zugfolge, deren Bewertung für den Gegner schlechter ist als sein bisher bester Wert, so wird er diesen Zug sicherlich nicht wählen. So fantastisch die Bewertung Ihrer Züge in diesem Zweig auch sein mag – es nützt nichts, wenn der Gegner diesen Weg nicht einschlägt, sondern ein paar Züge vorher eine andere Abzweigung im Spielbaum wählt. Folglich können Sie sich die Bewertung solcher Züge gleich sparen. Genau diese Erkenntnis nutzt der Alpha-Beta-Algorithmus aus. »Alpha« und »Beta« stehen dabei für zwei Hilfsvariablen, in denen die für jeden Spieler erreichbaren Stellungswerte gespeichert sind. Bild 2 zeigt einen fiktiven Spielbaum, bei dem Sie aufgrund der ersten Bewertungen spätere Zugfolgen einfach ignorieren dürfen. Man spricht hier von sogenannten »Cut-Offs«, der Zugbaum wird beschnitten. Dadurch reduziert sich die Anzahl der zu untersuchenden Stellungen.

● **Die Alpha-Beta-Suche im Detail:** Zu Beginn der Suche setzen Sie Alpha und Beta auf Extremwerte, die eine besonders schlechte Bewertung bedeuten. Denn bisher hat keiner der Spieler einen Stellungswert erreicht. Setzen Sie also Alpha, die Variable für die Computerbewertung, beispielsweise auf –32 000. Wenn Sie den Computer seinen Stellungswert maximieren lassen, dann ist der Wert –32 000 unakzeptabel, und jede erstbeste Stellung wird einen besseren Stellungswert liefern. Entsprechend setzen Sie Beta für Ihre eigenen Bewertungsergebnisse auf 32 000. Ihre Aufgabe ist es, eine Stellung mit minimaler Bewertung zu erreichen. Sie sind am Zug und haben, ausgehend von der aktuellen Situation, die drei Züge S1, S2 und S3 zur Auswahl. Die Suche beginnt mit Zug S1. Daraufhin bleiben dem Computer zwei Züge C1 und C2 als Antwort. Weil bisher noch kein Zug bewertet wurde, sind Alpha und Beta noch auf Ihren ursprünglichen Werten. Die Zugtiefe ist damit erschöpft, die Bewertungsfunktion liefert für die Zugfolge S1-C1 den Wert –2.

Suchtiefe	Anzahl Stellungen
1	40
2	1600
3	64 000
4	2 560 000
5	1 024 000 000 000

Der Suchtiefe sind durch den Aufwand der Stellungsbewertung enge Grenzen gesetzt. Schon bei geringen Suchtiefen kann die Bewertung Tage dauern.

Da der Computer am Zug ist, bedeutet dieser Wert gegenüber dem Inhalt von Alpha eine deutliche Verbesserung. Der Computer merkt sich diesen Zug und behält –2 als bisherigen Stellungswert in Alpha. Im Anschluß liefert die Bewertungsfunktion mit dem Wert 1 für den Zug S1-C2 eine noch bessere Stellung; wieder ändert sich Alpha auf den neuen Bestwert.

Die Ausführung des Zuges S1 bedeutet für Sie also, daß im Gegenzug der Computer einen Stellungswert von maximal 1 erreichen kann; damit wird diese Zahl zur Bewertung Ihres Zuges S1 herangezogen. S1 stellt Ihren ersten bewerteten Zug dar und liegt mit seiner Bewertung deutlich besser als Ihr bisheriger Beta-Wert. Also merken Sie sich diesen Zug als aktuellen Favoriten und setzen Sie Beta auf den Stellungswert 1. Anschließend probieren Sie Ihren nächsten Zug S2 aus.

Über die Brauchbarkeit von S2 entscheiden die Folgezüge des Computers C3 bis C5. Für Ihren zweiten Zug S2 besitzt Ihr Gegner noch keine Bewertung, der Alpha-Wert für diesen Zweig liegt also noch bei –32 000. Die Bewertungsfunktion liefert für S2-C3 den Wert –7, eine für Sie positive Entwicklung, denn schließlich versuchen Sie den Stellungswert zu minimieren. Der Computer ist hingegen mit dem Zug unzufrieden, ihm bleibt jedoch im Fall des Zuges S2 bisher keine bessere Alternative. Er übernimmt den Wert –7 in die Alpha-Variable. Als nächstes steht der Zug S2-C4 an. Er wird mit 3 bewertet. An dieser Stelle ist Ihnen klar, daß der Zug S2 eine für den Computer bessere Bewertung erlaubt als der Zug S1, der bestenfalls die Bewertung 1 lieferte (Beta). Also verzichten Sie auf eine weitere Betrachtung des Zuges S2. Der Zug S2-C5, der zwar eine gute Bewertung für Sie liefert, fällt unter den Tisch – der Computer hätte ihn sowieso nicht gewählt, da mit C4 ein für ihn besserer Zug vorliegt. Da S2 keine Verbesserung für Sie gebracht hat, gehen Sie nun zu S3 über. Wieder betrachten Sie die Computerreaktion. Der Zug S3-C6 liefert den Wert 2. Wieder erkennen Sie am Inhalt von Beta, daß Sie schon einen Zug gefunden hatten, der den Computer auf einen Wert von 1 drückte. Es erfolgt also ein weiterer Cut-Off. Der Suchbaum ist abgearbeitet, und Ihre Wahl fällt auf S1.

Auf diese Weise haben Sie etwa ein Drittel der denkbaren Züge verworfen, bevor sie zur Bewertung kommen konnten. Dennoch ist das Ergebnis identisch mit dem Ergebnis der Minimax-Suche, bei der Sie jedoch erheblich mehr Zeit bis zum Resultat benötigten. Der Alpha-Beta-Algorithmus stellt eine äußerst nützliche Erweiterung des Suchverfahrens dar. Sie erreichen mit ihm, daß nur Züge untersucht werden, die innerhalb des sogenannten Alpha-Beta-Fensters liegen. Außerhalb dieses Fensters liegende Züge fördern nicht die Entwicklung des Spiels zugunsten des ziehenden Spielers.

Die Bearbeitung eines Suchbaums im Detail

Bei der Implementation des Algorithmus in Ihrem Programm ist entscheidend, daß Sie im richtigen Moment die Alpha- und Beta-Werte mit den Werten 32 000 beziehungsweise –32 000 belegen. Dies ist dann der Fall, wenn Sie neu in eine Suchtiefe einsteigen. Verlassen Sie diese Suchtiefe schließlich, geben Sie den Wert als Funktionswert an die aufrufende Funktion zurück, wo er dann erneut als Stellungswert behandelt wird. Bild 3 zeigt das Schaubild für die Suche mit dem Alpha-Beta-Algorithmus.

- **Die Effizienz der Suche:** Wie effizient die Suche mit dem Alpha-Beta-Verfahren ist, liegt an der Anzahl der erreichten Cut-Offs. Im besten Fall liegen alle Folgezüge in ihrer Bewertung schlechter als der erste Zug, denn dann werden alle Untersuchungen sofort abgebrochen. Der schlechteste Fall tritt ein, wenn sich die Bewertungen von Zug zu Zug verbessern, weil sie das Alpha-Beta-Fenster nur langsam verkleinern und somit kaum Züge außerhalb des Bereichs liegen, die Sie dann abschneiden dürfen. Der Aufwand kann also genauso hoch sein wie bei der normalen Minimax-Suche. Unter Umständen liegt er jedoch auch nur bei der Hälfte oder noch niedriger. Wenn beim Tic Tac Toe der Alpha-Beta-Algorithmus auch keine spürbare Verbesserung bringt, weil die Bewertungsfunktion zu ungenau ist, stellt das Verfahren dennoch die Grundlage der Spielbarkeit aller komplexeren Strategiespiele dar, wie Sie noch sehen werden. Insbesondere dann, wenn es durch geschickte Zugauswahl gelingt, potentielle Cut-Off-Züge zuletzt, beziehungsweise gute Züge zuerst zu untersuchen.

Daß in Sachen Alpha-Beta-Algorithmus noch nicht das letzte Wort gesprochen ist, sehen Sie spätestens bei der Programmierung des Schachprogramms in einem der folgenden Beiträge. Dort lernen Sie darüber hinaus einige spielabhängige Klippen dieses Verfahrens kennen und wie Sie sie umschiffen.

(Frank Höppner/bm)

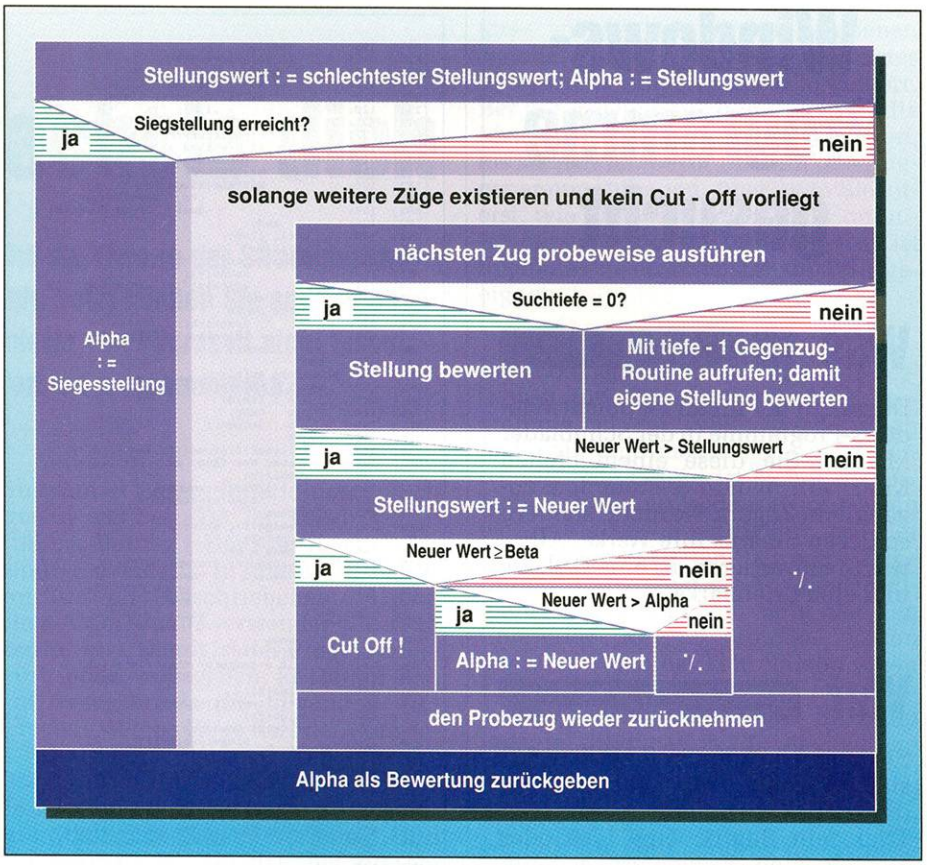


Bild 3. Das Schema des Alpha-Beta-Verfahrens bedeutet gegenüber dem Minimax-Verfahren eine deutliche Steigerung.

```
[N84]    1: {Programm: noradttt.pas
[3T4]    2:  Funktion: NORADs Tic Tac Toe
[G64]    3:  Sprache:  Turbo Pascal 6.Ø
[K34]    4:  Autor:    Frank Höppner
[585]    5:  (c) 1992 DMV Widuch GmbH & Co. KG }
[ØØØ]    6:
[NY4]    7: program NORAD_Tic_Tac_Toe;
[5H1]    8: uses crt;
[ST9]    9: type
[H22]    1Ø:   tfeld    = -1..1;
[Y12]    11:   tzug     = 1..9;
[C85]    12:   tbrett   = array[tzug] of tfeld;
[6XØ]    13: const
[GS1]    14:   mensch   = 1;
[U52]    15:   computer = -1;
[AH1]    16:   leer     = Ø;
[W85]    17:   zeichen  : array[tfeld] of char
[RT1]    18:            = ('x','.','o');
[AG5]    19:   sieger   : array[tfeld] of string
[P32]    2Ø:            = ('Computer',
[YD1]    21:               'Keiner',
[TU1]    22:               'Spieler');
[RNØ]    23: var
[V72]    24:   zug      : tzug;
[BWØ]    25:   gwst,
[NU2]    26:   ctr      : longint;
[VC6]    27: function spielende(b:tbrett):boolean;
[2P2]    28:   var i : integer;
[MUØ]    29:   begin
[UPØ]    3Ø:     i:=Ø;
[P31]    31:     repeat
[835]    32:       inc(i); spielende:=b[i]<>leer;
[8A4]    33:     until (b[i]=leer) or (i=9);
[ZNØ]    34:   end;
[UW4]    35: procedure zeige( b : tbrett );
[2P2]    36:   var i : integer;
[MUØ]    37:   begin
[FG2]    38:     for i:=1 to 9 do
[UT1]    39:       if i mod 3 = Ø
[EY4]    4Ø:         then writeln(zeichen[b[i]])
[6M4]    41:         else write(zeichen[b[i]]);
[ZNØ]    42:   end;
[LH5]    43: function wert( b : tbrett ) : tfeld;
[RA6]    44: const gewp : array[1..8,1..3] of integer
[V33]    45:            = (( ,2,3),(4,5,6),(7,8,9),
[HP2]    46:              (1,4,7),(2,5,8),(3,6,9),
[9X1]    47:              (1,5,9),(3,5,7));
[Z33]    48:   var i,w  : integer;
[MUØ]    49:   begin
[TG1]    5Ø:     Inc(ctr);
[ZS2]    51:     wert:=leer; i:=Ø;
[P31]    52:     repeat
```

```
[951]    53:       inc(i);
[955]    54:       w:=b[gewp[i,1]]+b[gewp[i,2]]
[3D2]    55:         +b[gewp[i,3]];
[ZT3]    56:     until (i=8) or (abs(w)=3);
[VM4]    57:     if w=3 then wert:=mensch else
[7R2]    58:      if w=-3 then begin
[D45]    59:        wert:=computer; inc(gwst) end;
[ZNØ]    6Ø:   end;
[2N5]    61: function suche( b : tbrett ):TZug;
[G93]    62:   var i,j,k : integer;
[D12]    63:       w,h  : tfeld;
[UP6]    64:   function computerwert(b:tbrett):tfeld;
[VE1]    65:     forward;
[UC6]    66:   function spielerwert(b:tbrett):tfeld;
[WP2]    67:     var i   : integer;
[NW1]    68:         w,h : tfeld;
[MUØ]    69:     begin
[NC2]    7Ø:       if spielende(b)
[H94]    71:         then spielerwert:=wert(b)
[WP1]    72:         else begin
[CE4]    73:           w:=computer; i:=Ø; h:=leer;
[P31]    74:           repeat
[951]    75:             inc(i);
[KV2]    76:             if b[i]=leer then
[MUØ]    77:               begin
[T82]    78:                 b[i]:=mensch;
[CV1]    79:                 h:=wert(b);
[L72]    8Ø:                 if h=leer then
[8C3]    81:                   h:=computerwert(b);
[6K2]    82:                 if h>w then w:=h;
[212]    83:                 b[i]:=leer;
[ZNØ]    84:               end;
[8R4]    85:           until (i=9) {or (h=mensch)};
[EM2]    86:           spielerwert:=w
[4MØ]    87:         end
[ZNØ]    88:     end;
[UP6]    89:   function computerwert(b:tbrett):tfeld;
[WP2]    9Ø:     var i   : integer;
[NW1]    91:         w,h : tfeld;
[MUØ]    92:     begin
[NC2]    93:       if spielende(b)
[FK4]    94:         then computerwert:=wert(b)
[WP1]    95:         else begin
[GU3]    96:           w:=mensch; i:=Ø; h:=leer;
[P31]    97:           repeat
[951]    98:             inc(i);
[KV2]    99:             if b[i]=leer then
[MUØ]    1ØØ:              begin
[UT2]    1Ø1:                b[i]:=computer;
[CV1]    1Ø2:                h:=wert(b);
[L72]    1Ø3:                if h=leer then
[6Ø3]    1Ø4:                  h:=spielerwert(b);
[2K2]    1Ø5:                if h<w then w:=h;
[212]    1Ø6:                b[i]:=leer;
[ZNØ]    1Ø7:              end;
[E25]    1Ø8:           until (i=9) {or (h=computer)};
[3X2]    1Ø9:           computerwert:=w
[4MØ]    11Ø:         end
[ZNØ]    111:     end;
[MUØ]    112:   begin
[LC3]    113:     i:=Ø; j:=Ø; w:=mensch;
[P31]    114:     repeat
```

Listing 1. »Noradttt.pas« ist eine erste einfache Ausführung des Tic-Tac-Toe-Spiels, bei der Sie den Computer nur über die elementaren Spielregeln aufklären.

Windows-Programme gesucht

Was

Dämmern bei Ihnen vielleicht Windows-Programme in der Schublade? Machen Sie diese einem breiten Kreis von Windows-Freunden zugänglich. Zögern Sie nicht, sondern schicken Sie uns Ihre Werke. Alles, was Sie investieren, ist eine Diskette und eine Briefmarke.

Wer

Sie brauchen kein Profi zu sein. Oft sind es gerade die Kleinigkeiten, die das Leben erleichtern. Ein Trick, ein Bugfix, eine Idee oder einfach eine Erleichterung beim Umgang mit dem PC unter Windows mögen Anregungen sein.

Wie

Ihre Zuschrift sollte den Quelltext sowie das ausführbare Programm enthalten. Die Features sollten Sie in einer kleinen Textdatei kurz dokumentieren.

Wieviel

Bei Veröffentlichung wird Ihre Mühe natürlich mit einem angemessenen Honorar belohnt. Außerdem wird Ihr Werk durch zigtausendfache Verbreitung für viele unserer Leser eine Hilfestellung sein.

Wohin

Bitte richten Sie Ihre Zuschriften an folgende Adresse:

Redaktion DOS International
Stichwort: Windows-Programme

Gruberstr. 46c
8011 Poing bei München

```
[951]  115:       Inc(i);
[KV2]  116:       if b[i]=leer then
[MUØ]  117:         begin
[252]  118:           k:=i; gwst:=Ø;
[B15]  119:           WRITE('Sicherer Sieger bei Zug '
[251]  12Ø:                 ,i,' : ');
[UT2]  121:           b[i]:=computer;
[6Ø3]  122:           h:=spielerwert(b);
[MS1]  123:           if h<w then
[893]  124:             begin w:=h; j:=i end;
[212]  125:           b[i]:=leer;
[LX3]  126:           WRITELN(sieger[h],'   (',
[Y83]  127:                   ctr,' Bewertungen, ',
[RS3]  128:                   gwst,' Siege bewertet');
[ZNØ]  129:         end;
[KS1]  13Ø:     until (i=9);
[Ø11]  131:     if j<>Ø
[ZN1]  132:      then begin
[5F1]  133:             suche:=j;
[DL4]  134:             writeln('Ich wähle Zug ',j)
[4MØ]  135:           end
[WP1]  136:      else begin
[6F1]  137:             suche:=k;
[GL4]  138:             writeln('Ich wähle Zug ',k)
[ZNØ]  139:           end;
[ZNØ]  14Ø:   end;
[3V2]  141: procedure spiel;
[PK2]  142:   var f : tbrett;
[1H1]  143:       s : tfeld;
[3F1]  144:       z : tzug;
[NZ1]  145:       n : integer;
[MUØ]  146:   begin
[855]  147:     fillchar(f,sizeof(f),leer); n:=Ø;
[DK5]  148:     write('Wer fängt an ? (1=spieler)');
[UH1]  149:     readln(z);
[Y73]  15Ø:     if z=1 then s:=mensch
[PY2]  151:            else s:=computer;
[P31]  152:     repeat
[GZ2]  153:       zeige(f); writeln;
[U71]  154:       ctr:=Ø;
[EB1]  155:       case s of
[4T3]  156:        computer : z:=suche(f);
[9H2]  157:        mensch   : repeat
[HU2]  158:                     write('Ihr Zug :');
[UH1]  159:                     readln(z);
[EZ2]  16Ø:                   until (z in [1..9])
[9P2]  161:                      and (f[z]=leer);
[ZNØ]  162:       end;
[MG1]  163:       f[z]:=s;
[7N4]  164:       if s=computer then s:=mensch
[PY2]  165:                     else s:=computer;
[L51]  166:       inc(n);
[WU4]  167:     until (wert(f)<>leer) or (n=9);
[7D1]  168:     zeige(f);
[W54]  169:     writeln('Gewonnen hat : ',
[H13]  17Ø:             sieger[wert(f)]);
[ZNØ]  171:   end;
[MUØ]  172: begin
[DB1]  173:   clrscr;
[XF1]  174:   writeln;
[MA6]  175:   writeln('NORAD-TicTacToe Simulator');
[XF1]  176:   writeln;
[696]  177:   writeln('Spielen Sie das klassische'+
[A24]  178:          ' Tic-Tac-Toe gegen den');
[236]  179:   writeln('Computer. Ihren Zug geben '+
[NN3]  18Ø:          'Sie durch die Nummer');
[Y26]  181:   writeln('des Feldes an, auf das Sie'+
[1X4]  182:          ' setzen wollen. Die Felder');
[W36]  183:   writeln('sind von links oben nach r'+
[K85]  184:        'echts unten durchnummeriert.');
[XF1]  185:   writeln;
[821]  186:   spiel;
[KNØ]  187: end.
```

Listing 1. »Noradttt.pas« ist eine erste einfache Ausführung des Tic-Tac-Toe-Spiels, bei der Sie den Computer nur über die elementaren Spielregeln aufklären.

```
[N84]    1: {Programm: alphattt.pas
[XF5]    2:  Funktion: Alpha-Beta-Tic-Tac-Toe
[G64]    3:  Sprache:  Turbo Pascal 6.Ø
[K34]    4:  Autor:    Frank Höppner
[815]    5:  (c) 1992 DMV Widuch GmbH & Co. KG}
[ØØØ]    6:
[YX5]    7: { Tic-Tac-Toe mit Alpha-Beta Algo-  }
[ZG6]    8: { rythmus und Suchtiefe 2 [maxtiefe]}
[8W5]    9: program Alpha_Beta_Tic_Tac_Toe;
[5H1]   1Ø: uses crt;
[ST Ø]  11: type
[Y12]   12:   tzug     = 1..9;
[WK5]   13:   tbrett   = array[tzug] of integer;
[6XØ]   14: const
[GS1]   15:   mensch   = 1;
[W52]   16:   computer = 1Ø;
[AH1]   17:   leer     = Ø;
[EZ1]   18:   maxtiefe = 2;
[KN6]   19:   zeichen  : array[leer..computer] of char
[WT2]   2Ø:            = ('.','o',' ',' ',' ',' ',
[932]   21:               ' ',' ',' ',' ','x');
[RNØ]   22: var
[WM2]   23:   zugwahl  : tzug;
[NU2]   24:   ctr      : longint;
[VC6]   25: function spielende(b:tbrett):boolean;
[2P2]   26:   var i : integer;
[MUØ]   27:   begin
[UPØ]   28:     i:=Ø;
[P31]   29:     repeat
[835]   3Ø:       inc(i); spielende:=b[i]<>leer;
[8A4]   31:     until (b[i]=leer) or (i=9);
[ZNØ]   32:   end;
[UW4]   33: procedure zeige( b : tbrett );
[2P2]   34:   var i : integer;
[MUØ]   35:   begin
[FG2]   36:     for i:=1 to 9 do
[UT1]   37:       if i mod 3 = Ø
[EY4]   38:         then writeln(zeichen[b[i]])
[6M4]   39:         else write(zeichen[b[i]]);
[ZNØ]   4Ø:   end;
[3W5]   41: function wert( b : tbrett ) : integer;
[RA6]   42: const gewp : array[1..8,1..3] of integer
[V33]   43:            = ((1,2,3),(4,5,6),(7,8,9),
[HP2]   44:               (1,4,7),(2,5,8),(3,6,9),
[9X1]   45:               (1,5,9),(3,5,7));
[BL4]   46:       bonus: array[1..9] of integer
[PP2]   47:            = (7,Ø,7,Ø,15,Ø,7,Ø,7);
[183]   48:   var i,w,h  : integer;
[MUØ]   49:   begin
[TG1]   5Ø:     Inc(ctr);
[SPØ]   51:     h:=Ø;
[FG2]   52:     for i:=1 to 9 do
[H12]   53:       case b[i] of
```

```
[A43]   54:         1 : Dec(h,bonus[i]);
[F93]   55:         1Ø : Inc(h,bonus[i]);
[ZNØ]   56:       end;
[UPØ]   57:     i:=Ø;
[P31]   58:     repeat
[951]   59:       inc(i);
[955]   6Ø:       w:=b[gewp[i,1]]+b[gewp[i,2]]
[3D2]   61:         +b[gewp[i,3]];
[UB1]   62:       case w of
[ZØ2]   63:        2Ø : inc(h,25);
[EZ1]   64:        Ø2 : dec(h,25);
[Ø12]   65:        3Ø : inc(h,3ØØ);
[GZ1]   66:        Ø3 : dec(h,3ØØ);
[ZNØ]   67:       end;
[FZ3]   68:     until (i=8) or (abs(h)>1ØØ);
[VH3]   69:     if h>1ØØ then h:=1ØØ else
[243]   7Ø:       if h<-1ØØ then h:=-1ØØ;
[ZC1]   71:     wert:=h;
[ZNØ]   72:   end;
[TL4]   73: function suche( b : tbrett;
[5E2]   74:           spieler,tiefe,
[5B4]   75:           alpha,beta:integer ):TZug;
[NM5]   76:   var i,gegner,h,stellung : integer;
[HT3]   77:       cutoff              : boolean;
[MUØ]   78:   begin
[MD2]   79:     cutoff:=false;
[2D3]   8Ø:     if spieler=computer
[ZN1]   81:       then begin
[LG2]   82:              stellung:=-1ØØØ;
[RW1]   83:              alpha:=-1ØØØ;
[YD2]   84:              gegner:=mensch
[4MØ]   85:            end
[WP1]   86:       else begin
[LG2]   87:              stellung:=1ØØØ;
[ZP1]   88:              beta:=1ØØØ;
[FV2]   89:              gegner:=computer
[ZNØ]   9Ø:            end;
[UPØ]   91:     i:=Ø;
[P31]   92:     repeat
[951]   93:       inc(i);
[N12]   94:       if b[i]=leer
[ZN1]   95:         then begin
[8G2]   96:           b[i]:=spieler;
[3G1]   97:           if tiefe=1
[KF2]   98:             then h:=wert(b)
[2V3]   99:             else h:=suche(b,gegner,
[933]   1ØØ:                  tiefe-1,alpha,beta);
[212]   1Ø1:           b[i]:=leer;
[NN2]   1Ø2:           case spieler of
[6B1]   1Ø3:            mensch  :
[LB4]   1Ø4:             {mensch minimiert beta}
[Y14]   1Ø5:             if h<=stellung then begin
[Y42]   1Ø6:              stellung:=h;
[TS4]   1Ø7:              if h<alpha then cutoff:=true;
[4M3]   1Ø8:              if h<beta then beta:=h;
[ZNØ]   1Ø9:            end;
[RL1]   11Ø:            computer:
[MY4]   111:             {computer maximiert alpha}
[MUØ]   112:             begin
[6M3]   113:              if tiefe=maxtiefe then
[C74]   114:                writeln('Bewertung für '+
[W62]   115:                        'Zug ',i,' : ',h);
[224]   116:              if h>=stellung then begin
[Y42]   117:               stellung:=h;
[3H4]   118:               if h>beta then cutoff:=true;
[X83]   119:               if h>alpha then begin
[GE1]   12Ø:                 alpha:=h;
[US2]   121:                 if tiefe=maxtiefe
[KT2]   122:                   then zugwahl:=i;
[ZNØ]   123:               end;
[ZNØ]   124:              end;
[ZNØ]   125:            end;
[ZNØ]   126:          end;
[ZNØ]   127:       end;
[LD3]   128:       until (i=9) or cutoff;
[2D3]   129:       if spieler=computer
[6X2]   13Ø:         then suche:=alpha
[DX2]   131:         else suche:=beta;
[ZNØ]   132:     end;
[3V2]   133: procedure spiel;
[PK2]   134:   var f   : tbrett;
[XM1]   135:       z   : tzug;
[N72]   136:       s,n : integer;
[MUØ]   137:   begin
[855]   138:     fillchar(f,sizeof(f),leer); n:=Ø;
[DK5]   139:     write('Wer fängt an ? (1=spieler)');
[UH1]   14Ø:     readln(z);
[Y73]   141:     if z=1 then s:=mensch
[PY2]   142:            else s:=computer;
[P31]   143:     repeat
[GZ2]   144:       zeige(f); writeln;
[U71]   145:       ctr:=Ø;
[EB1]   146:       case s of
[4M2]   147:        computer : begin
[VC3]   148:                   z:=suche(f,computer,
[2Y2]   149:                     maxtiefe,-1ØØØ,1ØØØ);
[C12]   15Ø:                   z:=zugwahl;
[YA4]   151:                   writeln('Mein Zug : ',z,' (',
[HA3]   152:                           ctr,' Bewertungen)');
[ZNØ]   153:                 end;
[ØF1]   154:        mensch   :
[P31]   155:          repeat
[HU2]   156:            write('Ihr Zug :');
[UH1]   157:            readln(z);
[EZ2]   158:          until (z in [1..9])
[9P2]   159:                and (f[z]=leer);
[ZNØ]   16Ø:       end;
[MG1]   161:       f[z]:=s;
[7N4]   162:       if s=computer then s:=mensch
[PY2]   163:                     else s:=computer;
[L51]   164:       inc(n);
[M15]   165:     until (abs(wert(f))=1ØØ) or (n=9);
[7D1]   166:     zeige(f);
[FX3]   167:     write('Gewonnen hat : ');
[UX2]   168:     if wert(f)=1ØØ then
[U44]   169:       writeln('Computer') else
[FØ3]   17Ø:      if wert(f)=-1ØØ then
[1Y3]   171:        writeln('Spieler') else
[L23]   172:        writeln('Keiner');
[ZNØ]   173:   end;
[MUØ]   174: begin
[DB1]   175:   clrscr;
[XF1]   176:   writeln;
[EG3]   177:   writeln('TicTacToe');
[XF1]   178:   writeln;
[696]   179:   writeln('Spielen Sie das klassische'+
[A24]   18Ø:          ' Tic-Tac-Toe gegen den');
[236]   181:   writeln('Computer. Ihren Zug geben '+
[NN3]   182:          'Sie durch die Nummer');
[Y26]   183:   writeln('des Feldes an, auf das Sie'+
[1X4]   184:          ' setzen wollen. Die Felder');
[W36]   185:   writeln('sind von links oben nach r'+
[K85]   186:        'echts unten durchnummeriert.');
[XF1]   187:   writeln;
[821]   188:   spiel;
[KNØ]   189: end.
```

Listing 2. »Alphattt.pas« ist in dieser Form als Muster für eigene Strategiespiele geeignet.

Reversi

Steinchen wechsle dich

Nach den vorangegangenen Beiträgen ist die Theorie der Strategiespiel-Programmierung für Sie ein offenes Buch. Jetzt können Sie endlich das tun, was schon die Windows-Programmierer von Microsoft nicht lassen konnten, nämlich Ihre eigene Reversi-Version programmieren.

Falls Sie dem Reiz von Reversi noch nicht erlegen sind, hier vorab die Regeln: Sie spielen wie bei Dame oder Schach auf einem 8×8 Felder großen Brett. Die Spielsteine sind Plättchen, deren Vorder- und Rückseite verschiedenfarbig sind. Zu Beginn sind die vier Mittelfelder bereits je zur Hälfte mit den Spielsteinen beider Spieler belegt (Bild 1). Abwechselnd setzen die Spieler einen Stein ihrer Farbe. Dabei ist wenigstens ein Stein des Gegners zwischen dem neuen und den bereits vorher gesetzten Steinen einzuschließen (horizontal, vertikal oder diagonal). Sollte dies einmal nicht gehen, setzt der betreffende Spieler eine Runde aus. Die eingeschlossenen Steine werden anschließend umgedreht, das heißt, sie wechseln die Farbe und damit den Besitzer. Wenn keiner der Spieler mehr setzen kann oder alle Felder belegt sind, ist das Spiel zu Ende. Sind die Steine Ihrer Farbe auf dem Brett in der Überzahl, so haben Sie gewonnen.

Aus den Spielregeln entnehmen Sie sofort, daß ein Spiel maximal über 60 Züge geht (das Brett hat 64 Felder, davon sind vier Felder zu Beginn bereits belegt). Gegenüber »Tic Tac Toe« mit neun Zügen bedeutet dies eine erhebliche Steigerung. Das Prinzip, mit dem Sie den Computer seinen »besten« Zug auswählen lassen, funktioniert jedoch genau wie bei »Tic Tac Toe«. Den in diesem Beitrag (»Der »TacTic« auf der Spur«, ebenfalls in dieser Ausgabe) ausführlich beschriebenen Alpha-Beta-Algorithmus verwenden Sie auch bei Reversi. Damit ist das Programmgerüst schon vorgegeben. Es stellen sich im wesentlichen die folgenden drei Fragen: Welche Züge darf der Computer überhaupt machen? (Bei »Tic Tac Toe« durfte der Computer auf jedes freie Feld setzen.) Wie bewerten Sie eine beliebige Reversi-Stellung? Und schließlich: Bieten sich bei diesem Spiel Methoden an, die beim Alpha-Beta-Algorithmus zu möglichst vielen Cutoffs führen, damit die Suche nach dem besten Zug beschleunigt wird?

Die Zugwahl übernimmt ein Generator

Welche Züge dem jeweiligen Spieler zur Auswahl stehen, besagen die Spielregeln. Jedoch ist es nicht sinnvoll, die Auswahl der denkbaren Züge innerhalb der Suche zu programmieren, wie Sie es bei »Tic Tac Toe« gesehen haben. Dazu ist die Auswahl zu kompliziert.

- **Der Zuggenerator:** Es bietet sich die Programmierung eines »Zuggenerators« an, der aus einem gegebenen Spielfeld alle potentiellen Züge eines Spielers in einer Zugliste speichert. Bei der Suche brauchen Sie dann die Liste nur der Reihe nach abzuarbeiten, bis sie leer ist. Bevor Sie nun den Zuggenerator aufbauen, überlegen Sie zuerst, wie Sie das Spielfeld im Computer darstellen möchten. In Pascal ist folgendes zweidimensionale Feld geeignet:

```
type feld = array[1..8,1..8] of figur;
```

Diese Deklaration erhöht zwar die Lesbarkeit und Übersichtlichkeit des Programms, in der Praxis wird jedoch meist ein eindimensionales Feld verwendet. Ein Grund ist sicher die Bequemlichkeit, außerdem erhöht es ein wenig die Geschwindigkeit, mit der das Programm letztendlich seine Züge findet. Bild 2 zeigt ein solches eindimensionales Feld, auf dem Sie den Computer Reversi spielen lassen könnten. Noch hat kein Spieler einen Zug gemacht, es besteht die Grundstellung. Jetzt überprüfen Sie zum Beispiel die Position C5, ob sie einen gültigen Zug für den roten Spieler darstellt. Weil das eindimensionale Spielfeld in der Größe 10 × 10 gewählt ist, fällt die Umrechnung der zweidimensionalen Position C5 in das eindimensionale Feld leicht: zehn mal die Zeile plus die Spalte. Da die C-Spalte die dritte Spalte ist, handelt es sich bei C5 also um Position 53 in unserem Feld. Der Zug auf C5 ist legal, wenn Sie damit gegnerische Figuren einschließen. Um dies festzustellen, müssen Sie entlang allen acht horizontalen, vertikalen und diagonalen Richtungen suchen, bis Sie nach gegnerischen Figuren direkt anschließend auf eigene treffen. Nur dann ist C5 ein ausführbarer Zug. Diese Suche programmieren Sie als einfache Schleife, bei der Sie alle Richtungen nacheinander verfolgen. In einem zweidimensionalen Feld verändern Sie dabei jeweils zwei Variablen: die Spalten- und die Zeilenposition. Für die Richtung diagonal nach rechts unten sind beispielsweise beide Werte um 1 zu erhöhen. Bei der eindimensionalen Darstellung gelangen Sie zur neuen Position aber schon nach nur einer Addition: Sie addieren zum alten Wert 11 – das entspricht einer Spalte (plus 10) und einer Zeile (plus 1). Im Grunde legt Ihr Pascal-Compiler natürlich auch ein zweidimensionales Spielbrett eindimensional im Speicher ab, weil der Speicher nur linear adressierbar ist. Indem Sie diese Linearität durch etwas Vorarbeit berücksichtigen, verbleibt dem Compiler weniger Arbeit und das Programm wird kompakter.

Aber diese Darstellung hat noch einen weiteren Vorteil: Bei der zweidimensionalen Brettdefinition sind je nach

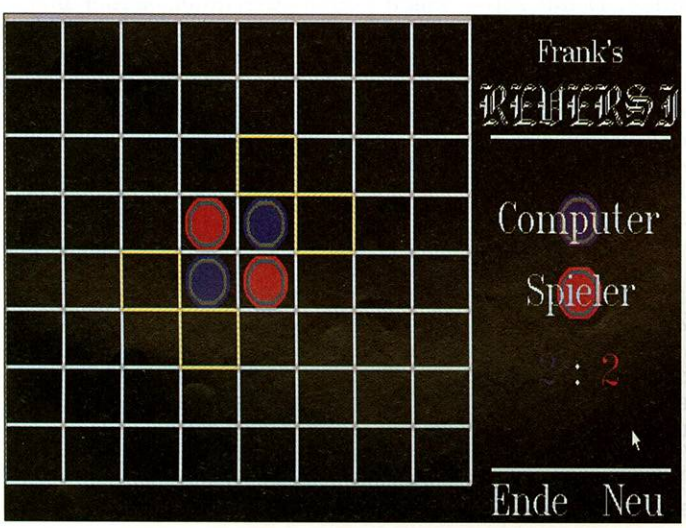


Bild 1. Zu Beginn des Spiels stehen die Steine in der Grundstellung. Ist der Spieler am Zug, zeigt der Computer die legalen Züge gelb an.

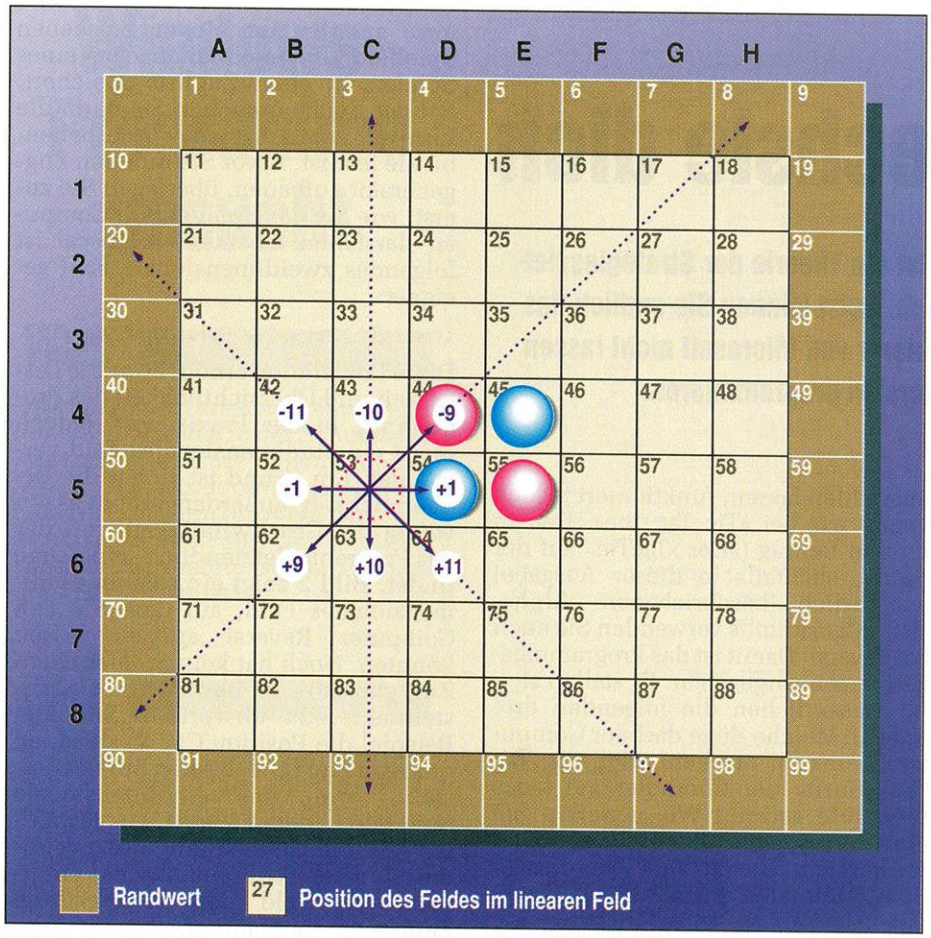


Bild 2. Das Speichern des zweidimensionalen Brettes in einem eindimensionalen Array erleichtert die Programmierung.

Bewegungsrichtung bis zu zwei Randabfragen notwendig, damit die Suchschleife das Spielfeld nicht an seinen Begrenzungen überschreitet.

Ein lineares Brett vereinfacht die Randabfrage

Insgesamt gibt es sogar vier verschiedene Randabfragen, je eine für oben, unten, links und rechts. Indem Sie aber die überzähligen Felder des eindimensionalen Feldes mit einem während des Spielverlaufs nicht verwendeten Wert belegen, erkennen Sie leicht, ob Sie den Spielfeldrand überschritten haben. Denn wenn Sie den Inhalt des jeweils in der Suchrichtung nächsten Feldes kontrollieren, brechen Sie einfach ab, sobald Sie statt einer Figur den Randwert gefunden haben. So verhindern Sie, daß Sie bei der steten Addition der Richtungskonstante am normalen zweidimensionalen Spielfeld austreten und an anderer Stelle regelwidrig wieder in das Spielfeld gelangen (»wrap around«). Legen Sie nun noch die acht Suchrichtungen in einem weiteren Konstantenfeld an, so erledigen Sie die Überprüfung auf einen legalen Zug mit zwei Schleifen, einer für die Richtung und einer, die entlang derselben sucht. Mit einer dritten Schleife arbeiten Sie alle leeren Spielfelder ab, und fertig ist der Zuggenerator.

● **Die Bewertung:** Damit ist es an der Zeit, sich über die Bewertung des jeweiligen Spielstandes Gedanken zu machen. Hier kommen Sie ohne Spielerfahrung nicht aus. Sie können aber mit dem Computer gemeinsam lernen. Zunächst liegt es nahe, einfach die Differenz von eigenen und gegnerischen Figuren zu zählen. Damit setzen Sie das Ziel des Spiels direkt um, denn schließlich gewinnen Sie nur, wenn die Steine eigener Farbe auf dem Spielfeld in der Überzahl sind. Nach ein paar Runden merken Sie jedoch, daß diese Bewertungsfunktion noch nicht optimal ist. Denn der Alpha-Beta-Algorithmus untersucht in einer bestimmten Suchtiefe das Gleichgewicht der Farben. Dabei bleibt unberücksichtigt, daß Sie eventuell schon einen Zug später noch mehr Spielsteine zurückgewinnen können, als der Computer vorher bei seinem besten Zug gewann. Hingegen kann ein Zug für den Computer langfristig die bessere Wahl sein, obwohl er auf den ersten Blick weniger Gewinn bringt, Ihnen aber die Gelegenheit zum Kontern nimmt. Durch die Beschränkung der Bewertung auf eine festgelegte Tiefe, den Suchhorizont, geschehen in so einem Fall Bewertungsfehler. Es hilft zwar etwas, die Suchtiefe zu erhöhen, doch ein Horizont bleibt immer bestehen, solange Sie nicht bis zum Ende des Spiels suchen. Aber schon bei sehr geringen Suchtiefen dauert eine Bewertung viel zu lange, als daß dies durchführbar wäre. Ihre Bewertungsfunktion muß also zusätzlich positionelle Vorteile der Spieler bewerten. Dabei ist jedoch Vorsicht angesagt, weil falsche Einschätzungen meist schwere Folgen haben. Am besten probieren Sie Ihre Änderungen zunächst in der Suchtiefe 1 aus, denn nur so können Sie mit Sicherheit beurteilen, ob sie in die von Ihnen gewünschte Richtung wirken. Erst dann erhöhen Sie die Suchtiefe, weil dadurch die Auswirkungen durch die erfolgten Züge und Gegenzüge vielschichtiger in die Bewertung eingehen.

Einige Felder bieten einen großen Stellungsvorteil

Wie schätzen Sie aber positionelle Vorteile eines Spielers ab? Oft kann ein gewinnbringender Zug des Gegners noch übertroffen werden, indem er einfach seinen Stein hinter den vorangegangenen Zug setzt. In diesem Fall gewinnt er nicht selten alle Steine zurück, die er vorher verloren hat. Dieses »Dranhängen« an einen vorangegangenen Zug funktioniert natürlich nicht, wenn der Spielfeldrand bereits erreicht ist. Deswegen bewerten Sie Steine am Spielfeldrand mit einem Bonus. Dadurch belohnen Sie automatisch die Beherrschung der Flanken. Besonders hohe positionelle Bewertungen gibt es für die Eckfelder, denn sie erlauben es, komplette Flanken des Gegners für sich zu gewinnen. Außerdem sind die Eckfelder die einzigen Felder, die Ihnen keiner mehr umdrehen kann. Die Nachbarfelder der Ecken bewerten Sie hingegen schlecht, da ein eigener Stein in dieser Position bedeutet, daß der Gegner meist das zugehörige Eckfeld erreicht. Aus diesem Schema ergibt sich eine erste positionelle Bewertung. Im Reversi-Listing dieses Beitrags finden Sie die positionelle Bewertung für jedes einzelne Spielfeld im konstanten Feld »Korrektur«. Weil Ihnen die Felder am Spielfeldrand so viele Stellungspunkte wert sind, sollte Ihnen gerade hier durch die beschränkte Suchtiefe tunlichst kein Bewertungsfehler einen Strich durch die Rechnung machen. Wenn der Gegenspieler den Stein am Brettrand wegen einer Fehleinschätzung aber sofort zurück-

gewinnen kann, fördern Sie statt der eigenen die gegnerische Flankenentwicklung. Bei der Bewertung solcher Steine sollten Sie also schon in der Bewertungsfunktion eventuelle Folgezüge berücksichtigen. Innerhalb des Suchalgorithmus werden die Gegenzüge zwar analysiert, aber – wie bereits erwähnt – nur bis zum festgesetzten Suchhorizont. Sie müssen kontrollieren, ob der Stein unmittelbar vom Gegner gewonnen werden kann. Ein Stein an einer Flanke kann nur durch seine beiden direkten Nachbarn auf derselben Flanke gewonnen werden, das heißt, der Aufwand bleibt vertretbar gering. Zeigt die Analyse, daß der Stein nicht sicher ist, bestrafen Sie die Stellung mit einem entsprechenden Punktabzug. Die Dominanz Ihrer Farbe auf einer Flanke ist Ihnen hingegen einen Bonus wert. Auf diese Weise ist die Bewertungsfunktion »brettwert« im Listing entstanden.

Ein kleiner Trick beschleunigt die Suche

Schließlich bleibt die Frage zu klären, ob der Alpha-Beta-Suche etwas unter die Arme gegriffen werden kann. Sie erinnern sich, daß die Geschwindigkeit der Suche davon abhängt, wie viele Cutoffs auftreten, und daß Sie Cutoffs fördern, indem Sie die voraussichtlich guten Züge zuerst bewerten (wie bei »Tic Tac Toe«). Natürlich läßt sich nicht vorhersagen, welcher Zug in einer bestimmten Situation die meisten Punkte bringen wird, aber aufgrund Ihrer Bewertungsfunktion wissen Sie am besten, welche Züge die meisten Punkte bringen. Bauen Sie nun die Zugliste so auf, daß Sie die Züge, deren positionelle Bewertung auf eine gute Gesamtbewertung hoffen läßt, zuerst eintragen.

Dies geschieht beim Zuggenerator des Listings, indem er die Reihenfolge der zu untersuchenden Felder in einem Konstantenfeld speichert. Das Feld ist so organisiert, daß die Reihenfolge der untersuchten Felder von den Rand- zu den Mittelfeldern verläuft. Denn insbesondere Züge auf die Randfelder erreichen eine hohe Bewertung. Dadurch, daß Sie die potentiellen Kandidaten für die nächste Zugauswahl dann zu Beginn der Liste finden, erhöhen Sie die Anzahl der Cutoffs.

● **Die Spielegrafik:** Wenn Ihre Suche schließlich gut auf Ihre Züge pariert, sollten Sie Ihrem Reversi auch eine passende Umgebung angedeihen lassen. Mit den Methoden, die Turbo Pascal Ihnen zur Verfügung stellt, ist auch dies kein Problem. Eine Maussteuerung ist schnell mit Standard-Prozeduren gezaubert und erleichtert die Bedienung. Mit den Prozeduren »GetImage« und »PutImage« der Unit Graph hauchen Sie den Reversi-Steinen auf einfache Weise Leben ein: Das Umdrehen der gewonnenen Steine können Sie in animierter Grafik betrachten (Bild 3). Durch einen zusätzlichen Aufruf des Zuggenerators ermitteln Sie auch die möglichen Züge des Spielers, die Sie ihm bei seiner Zugwahl auf dem Spielfeld als kleine Hilfe markieren. Auf ähnliche Weise können Sie den Fortschritt der Suche des Computerzugs kommentieren.

Die Spiele, bei denen der Computer nur mit niederen Aufgaben wie Brettdarstellung und Regelkontrolle beauftragt wird, stellen für einen Programmierer keine große Herausforderung dar. Insofern haben die eingangs erwähnten Programmierer von Microsoft ihren spieletechnischen Höhepunkt bereits hinter sich, denn einen Computergegner gibt es in den Spielen von Windows 3.1 nicht mehr. Kein Grund für Sie jedoch, sich ebenfalls auf Ihren Lorbeeren auszuruhen: Im nächsten Beitrag wartet das Spiel der Spiele auf Sie: Schach!

(Frank Höppner/bm)

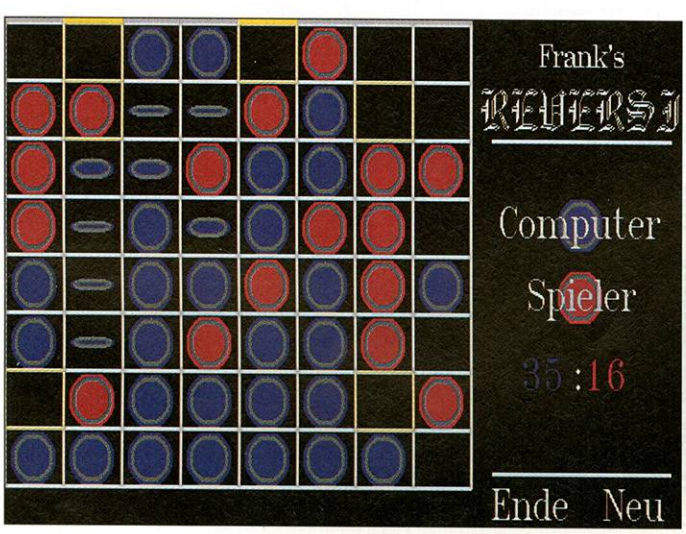


Bild 3. Die Animation des Steineumdrehens gibt der grafischen Spielfelddarstellung den letzten Schliff.

```
[2V3]    1: (Programm: reversi.pas
[S64]    2:  Funktion: Reversi-Spiel
[G64]    3:  Sprache:  Turbo Pascal 6.0
[K34]    4:  Autor:    Frank Höppner
[585]    5:  (c) 1992 DMV Widuch GmbH & Co. KG )
[000]    6:
[KA6]    7: ( O)ptions C)ompiler eX)tended Syntax
[ML5]    8: Die Dateien goth.chr, trip.chr und
[TG5]    9: egavga.bgi ins aktuelle Directory
[5X1]   10: kopieren )
[000]   11:
[0Z2]   12: program reversi;
[ZM2]   13: uses graph,crt;
[ST9]   14: type
[3D5]   15:  tspiel     = (beenden,spielen,neu);
[RD4]   16:  tinhalt    = (jenseits,leer,
[XX2]   17:                mensch,computer);
[SY4]   18:  tspieler   = mensch..computer;
[P36]   19:  tspielfeld = array[0..99] of tinhalt;
[GY2]   20:  tzug       = integer;
[9N5]   21:  tzugliste  = array[1..99] of tzug;
[MP5]   22:  tround     = array[1..8] of integer;
[VL5]   23:  tbitblt    = array[0..1023] of byte;
[6X0]   24: const
[563]   25:  umgebung   : tround
[253]   26:              = (-11,-10,-9,-1,1,9,10,11);
[TM6]   27:  gegner     : array[tspieler] of tspieler
[L73]   28:              = (computer,mensch);
[EP6]   29:  name       : array[tspieler] of STRING[8]
[314]   30:              = ('Spieler ','Computer');
[XY6]   31:  startwert  : array[tspieler] of integer
[XX1]   32:              = (32000,-32000);
[WX5]   33:  korrektur  : array[0..99] of integer
[2L2]   34:              = (0,0,0,0,0,0,0,0,0,0,
[SR4]   35:   0,1888,-200,200,200,200,200,-200,1888,0,
[8S3]   36:   0,-200,-400,8,8,8,8,-400,-200,0,
[M83]   37:   0,200,8,20,20,20,20,8,200,0,
[M83]   38:   0,200,8,20,20,20,20,8,200,0,
[M83]   39:   0,200,8,20,20,20,20,8,200,0,
[M83]   40:   0,200,8,20,20,20,20,8,200,0,
[8S3]   41:   0,-200,-400,8,8,8,8,-400,-200,0,
[SR4]   42:   0,1888,-200,200,200,200,200,200,-200,1888,0,
[3C2]   43:   0,0,0,0,0,0,0,0,0,0);
[EA5]   44:  rand       : array[0..99] of integer
[2L2]   45:              = (0,0,0,0,0,0,0,0,0,0,
[292]   46:                 0,0,1,1,1,1,1,1,0,0,
[W82]   47:                 0,2,0,0,0,0,0,0,2,0,
[W82]   48:                 0,2,0,0,0,0,0,0,2,0,
[W82]   49:                 0,2,0,0,0,0,0,0,2,0,
[W82]   50:                 0,2,0,0,0,0,0,0,2,0,
[W82]   51:                 0,2,0,0,0,0,0,0,2,0,
[W82]   52:                 0,2,0,0,0,0,0,0,2,0,
[292]   53:                 0,0,1,1,1,1,1,1,0,0,
[3C2]   54:                 0,0,0,0,0,0,0,0,0,0);
[RN0]   55: var
[T36]   56:  animate      : array[0..9] of tbitblt;
[P74]   57:  mousex,mousey : integer;
[TG3]   58:  status       : TInhalt;
[VK3]   59:  zustand      : TSpiel;
[VW3]   60:  restfelder   : integer;
[000]   61:
[9E5]   62: ( Routinen zur Maussteuerung )
[LK5]   63: procedure MouseInit; assembler;
[FN0]   64:   asm
[GH1]   65:     xor ax,ax
[B81]   66:     int 33h
[GB1]   67:     mov ax,7
[DB1]   68:     mov cx,0
[XV1]   69:     mov dx,640d
[B81]   70:     int 33h
[HB1]   71:     mov ax,8
[DB1]   72:     mov cx,0
[WV1]   73:     mov dx,350d
[B81]   74:     int 33h
[AB1]   75:     mov ax,1
[B81]   76:     int 33h
[ZN0]   77:   end;
[3F4]   78: procedure WaitMouseClick;
[9A3]   79:   assembler; label lp;
[FN0]   80:   asm
[AB1]   81:     mov ax,1
[B81]   82:     int 33h
[WZ1]   83: lp: mov ax,3
[B81]   84:     int 33h
[P81]   85:     cmp bl,1
[E41]   86:     jnz lp
[VV2]   87:     mov ds:mousex,cx
[1W2]   88:     mov ds:mousey,dx
[BB1]   89:     mov ax,2
[B81]   90:     int 33h
[ZN0]   91:   end;
[8C6]   92: ( Routinen zur Anzeige- und Spielfeld-
[5R2]   93: ( verwaltung.)
[7R2]   94: procedure init;
[034]   95:   var grdr,grmd : integer;
[BL2]   96:       i          : integer;
[MU0]   97:   begin
[1T1]   98:    randomize;
[C04]   99:    grdr:=vga; grmd:=vgamed;
[0T3]  100:    initgraph(grdr,grmd,'');
[EK6]  101:    SetTextStyle(GothicFont,HorizDir,15);
[KE4]  102:    OutTextXY(30,100,'REVERSI');
[V26]  103:    SetLineStyle(solidLn,0,ThickWidth);
[1F3]  104:    for i:=0 to 4 do begin
[FV4]  105:      SetFillStyle(SolidFill,red);
[TD2]  106:      SetColor(red);
[XR4]  107:      ellipse(27,20,0,360,20,(4-i+1)*4-2);
[G63]  108:      floodfill(27,20,red);
[RC3]  109:      SetColor(darkgray);
[3K4]  110:      ellipse(27,20,0,360,17,(4-i+1)*3);
[ZW4]  111:      GetImage(3,3,51,38,animate[i]);
[Z65]  112:      SetFillStyle(SolidFill,black);
[V12]  113:      Bar(0,0,55,40);
[ZN0]  114:    end;
[GF3]  115:    for i:=5 to 9 do begin
[M35]  116:      SetFillStyle(SolidFill,blue);
[4N2]  117:      SetColor(blue);
[2H4]  118:      ellipse(27,20,0,360,20,(i-4)*4-2);
[KD3]  119:      floodfill(27,20,blue);
[RC3]  120:      SetColor(darkgray);
[894]  121:      ellipse(27,20,0,360,17,(i-4)*3);
[ZW4]  122:      GetImage(3,3,51,38,animate[i]);
[Z65]  123:      SetFillStyle(SolidFill,black);
[V12]  124:      Bar(0,0,55,40);
[ZN0]  125:    end;
[LK2]  126:    ClearViewPort;
[CZ1]  127:    SetColor(7);
[CG2]  128:    for i:=0 to 8 do
[K33]  129:      Line(0,i*40,8*55,i*40);
[CG2]  130:    for i:=0 to 8 do
[E43]  131:      Line(i*55,0,i*55,8*40);
[KM6]  132:    SetTextStyle(TriplexFont,HorizDir,3);
[UB4]  133:    OutTextXY(505,5,'Frank''s');
[6E6]  134:    SetTextStyle(GothicFont,HorizDir,5);
[0F4]  135:    OutTextXY(450,10,'REVERSI');
[PM6]  136:    SetTextStyle(TriplexFont,HorizDir,4);
[RL6]  137:    PutImage(520,120,animate[9],normalput);
[GL6]  138:    PutImage(520,170,animate[0],normalput);
[4V4]  139:    OutTextXY(468,115,'Computer');
[9G4]  140:    OutTextXY(495,165,'Spieler');
[UF3]  141:    OutTextXY(540,220,':');
[NT4]  142:    OutTextXY(460,310,'Ende  Neu');
```

Das vollständige Reversi-Programm setzt sich aus den Überlegungen dieses Beitrags zusammen.

```
[7X2]  143:    Line(46Ø,31Ø,63Ø,31Ø);
[ØN2]  144:    Line(46Ø,8Ø,63Ø,8Ø);
[Z65]  145:    SetFillStyle(SolidFill,black);
[ZNØ]  146:  end;
[5L4]  147: procedure neues_spielfeld(
[583]  148:                 var f:tspielfeld );
[2P2]  149:   var i : integer;
[242]  15Ø:       s : TSpieler;
[MUØ]  151:   begin
[P45]  152:     fillchar(f,sizeof(f),jenseits);
[4S2]  153:     for i:=11 to 88 do
[X74]  154:       if (i mod 1Ø>Ø) and (i mod 1Ø<9)
[DT2]  155:         then f[i]:=leer;
[CP2]  156:     if random(1ØØØ)>498
[KE5]  157:       then s:=computer else s:=mensch;
[F53]  158:     f[44]:=s; f[55]:=s;
[X66]  159:     f[45]:=gegner[s]; f[54]:=gegner[s];
[ZNØ]  16Ø:   end;
[ØM5]  161: procedure feldSetzen( p:integer;
[H12]  162:                       s:TSpieler);
[MB3]  163:   var a,x,y : integer;
[MUØ]  164:   begin
[KR2]  165:     x:=(p mod 1Ø-1)*55+3;
[XP2]  166:     y:=(p div 1Ø-1)*4Ø+3;
[EB1]  167:     case s of
[FS5]  168:      mensch   : PutImage(x,y,animate[Ø],
[812]  169:                          normalput);
[K16]  17Ø:      computer : PutImage(x,y,animate[9],
[812]  171:                          normalput);
[ZNØ]  172:     end;
[ZNØ]  173:   end;
[4L5]  174: procedure loesche_spielfeld;
[2Ø3]  175:   var x,y : integer;
[MUØ]  176:   begin
[8H2]  177:     for x:=Ø to 7 do
[AH2]  178:       for y:=Ø to 7 do
[ØU4]  179:         Bar(x*55+2,y*4Ø+3,x*55+52,y*4Ø+38);
[RC4]  18Ø:   SetLineStyle(solidln,Ø,3);
[CZ1]  181:   SetColor(7);
[AH2]  182:   for x:=Ø to 8 do
[E53]  183:     Line(Ø,x*4Ø,8*55,x*4Ø);
[AH2]  184:   for x:=Ø to 8 do
[T63]  185:     Line(x*55,Ø,x*55,8*4Ø);
[ZNØ]  186:  end;
[H44]  187: procedure mark(p,c:byte);
[UW2]  188:   var x,y:integer;
[MUØ]  189:   begin
[GØ2]  19Ø:     SetColor(c);
[NC4]  191:     SetLineStyle(solidln,Ø,1);
[KR2]  192:     x:=(p mod 1Ø-1)*55+3;
[XP2]  193:     y:=(p div 1Ø-1)*4Ø+3;
[ZB2]  194:     MoveTo(x-3,y-3);
[674]  195:     LineRel(55,Ø); LineRel(Ø,4Ø);
[9R4]  196:     LineRel(-55,Ø); LineRel(Ø,-4Ø);
[ZNØ]  197:  end;
[7P6]  198: procedure feld_anzeigen(f:tspielfeld);
[2P2]  199:   var i : integer;
[MUØ]  2ØØ:   begin
[4S2]  2Ø1:     for i:=11 to 88 do
[BA3]  2Ø2:       feldSetzen(i,f[i]);
[ZNØ]  2Ø3:   end;
[9G4]  2Ø4: procedure ZugAusfuehrung(
[G83]  2Ø5:               var f : TSpielfeld;
[242]  2Ø6:                   s : TSpieler;
[3F1]  2Ø7:                   z : TZug;
[LL3]  2Ø8:                   animation : boolean );
[NP3]  2Ø9:   var n,i,j,p  : integer;
[TL2]  21Ø:       g        : TSpieler;
[1T4]  211:       turn     : array[1..56] of byte;
[7R3]  212:   procedure TurnPieces;
[SK3]  213:     var a,i,x,y : integer;
[MUØ]  214:     begin
[9W1]  215:       if s=mensch
[4U3]  216:         then for a:=9 downto Ø do
[MUØ]  217:           begin
[PG3]  218:             for i:=1 to n do begin
[Y84]  219:               x:=(turn[i] mod 1Ø-1)*55+3;
[V94]  22Ø:               y:=(turn[i] div 1Ø-1)*4Ø+3;
[Y74]  221:               PutImage(x,y,animate[a],
[812]  222:                        normalput);
[ZNØ]  223:             end;
[HS1]  224:             delay(3Ø-n);
[4MØ]  225:           end
[L33]  226:         else for a:=Ø to 9 do
[MUØ]  227:           begin
[PG3]  228:             for i:=1 to n do begin

[Y84]  229:               x:=(turn[i] mod 1Ø-1)*55+3;
[V94]  23Ø:               y:=(turn[i] div 1Ø-1)*4Ø+3;
[Y74]  231:               PutImage(x,y,animate[a],
[812]  232:                        normalput);
[ZNØ]  233:             end;
[HS1]  234:             delay(3Ø-n);
[ZNØ]  235:           end;
[ZNØ]  236:     end;
[MUØ]  237:   begin
[F73]  238:     g:=Gegner[s]; n:=Ø;
[9M5]  239:     if animation then feldSetzen(z,s);
[MG1]  24Ø:     f[z]:=s;
[AF3]  241:     for i:=1 to 8 do begin
[MZ2]  242:       p:=z+umgebung[i];
[RM5]  243:       while f[p]=g do p:=p+umgebung[i];
[KL5]  244:       if (f[p]=s) and (p<>z+umgebung[i])
[ZU1]  245:         then begin
[EZ2]  246:           p:=p-umgebung[i];
[822]  247:           while p<>z do
[MUØ]  248:             begin
[VM2]  249:               f[p]:=s; Inc(n);
[172]  25Ø:               turn[n]:=p;
[SV2]  251:               p:=p-umgebung[i]
[ZNØ]  252:             end;
[ZNØ]  253:       end;
[ZNØ]  254:     end;
[ZNØ]  255:     if animation and (n>Ø)
[6D3]  256:       then TurnPieces;
[NW2]  256:   end;
[ZNØ]  257:
[F44]  258: ( Zuglistengenerator )
[D65]  259: procedure ZugListenGenerator(
[TH2]  26Ø:                   f : TSpielfeld;
[242]  261:                   s : TSpieler;
[CP2]  262:                 var n : integer;
[483]  263:                 var l : TZugListe );
[2A5]  264:   const folge : array[1..6Ø] of byte
[D73]  265:               = (11,18,81,88,12,13,14,15,
[DV2]  266:                  16,17,21,31,41,51,61,71,
[7Z2]  267:                  28,38,48,58,68,78,82,83,
[GX2]  268:                  84,85,86,87,22,23,24,25,
[MY2]  269:                  26,27,37,47,57,67,77,76,
[PW2]  27Ø:                  75,74,73,72,62,52,42,32,
[BX2]  271:                  33,34,35,36,46,56,66,65,
[YH1]  272:                  64,63,53,43);
[4H3]  273:   var i,j,p,k : integer;
[CM2]  274:       g       : TSpieler;
[ED2]  275:       ok      : boolean;
[MUØ]  276:   begin
[EG6]  277:     ( Benutzung des Arrays erhöht durch
[4L6]  278:     automatische Sortierung der Zugliste
[X66]  279:     die Cut-Off-Rate des Alpha-Beta Alg.)
[NR3]  28Ø:     fillchar(l,SizeOf(l),Ø);
[K73]  281:     g:=Gegner[s]; n:=1;
[WH2]  282:     for i:=1 to 6Ø do
[X65]  283:       if f[folge[i]]=leer then begin
[5D2]  284:         ok:=false; j:=1;
[WX4]  285:         while (j<=8) and not ok do begin
[KE4]  286:           p:=folge[i]+umgebung[j];
[8M2]  287:           while (f[p]=g) do
[EZ2]  288:             p:=p+umgebung[j];

[7R2]  289:             ok:=(f[p]=s) and
[3L4]  29Ø:                 (p<>folge[i]+umgebung[j]);
[B51]  291:           Inc(j);
[ZNØ]  292:         end;
[8E2]  293:         if ok then begin
[MØ4]  294:           l[n]:=folge[i]; Inc(n);
[ZNØ]  295:         end;
[ZNØ]  296:       end;
[E41]  297:     Dec(n);
[ZNØ]  298:   end;
[AS4]  299: ( Zuganalyse und Auswahl )
[753]  3ØØ: procedure ZugWahl(
[TH2]  3Ø1:                   f : TSpielfeld;
[242]  3Ø2:                   s : TSpieler;
[7A2]  3Ø3:                 var z : TZug);
[Y73]  3Ø4:   var maxt : integer;
[8U5]  3Ø5:   function BrettWert( f:TSpielfeld;
[Y12]  3Ø6:                       s:TSpieler;
[423]  3Ø7:                       zs,zc:integer):integer;
[HP3]  3Ø8:     var i,add,nb : integer;
[GD4]  3Ø9:         tab      : array[1..4,tinhalt]
                                 of integer;
[SW1]  31Ø:         wert,ctr : array[tinhalt]
[R94]  311:                    of integer;
[SW1]  312:         g        : TSpieler;
[TL2]  313:     begin
[MUØ]  314:       ( Statistik )
[MD2]  315:       fillchar(ctr,SizeOf(ctr),Ø);
[KM4]  316:       fillchar(wert,SizeOf(wert),Ø);
[TY4]  317:       add:=Ø; g:=Gegner[s]; nb:=Ø;
[NE4]  318:       for i:=11 to 88 do begin
[RM3]  319:         wert[f[i]]:=wert[f[i]]+
[ZG4]  32Ø:           2Ø+korrektur[i];
[4W2]  321:         Inc(ctr[f[i]]);
[MS2]  322:         case rand[i] of
[1P2]  323:       Ø : ;
[YHØ]  324:       1 : if (f[i+1]=g) and (f[i-1]=g) then d
                      ec(nb,54) else
[SN6]  326:           if (f[i+1]=g) or (f[i-1]=g) then in
                      c(nb,135);
[548]  327:       2 : if (f[i+1Ø]=g) and (f[i-1Ø]=g) then
                      dec(nb,54) else
[FT6]  328:           if (f[i+1Ø]=g) or (f[i-1Ø]=g) then
                      inc(nb,135);
[ZNØ]  329:         end;
[ZNØ]  33Ø:       end;
[5Ø4]  331:       ( Partie entschieden ? )
[D63]  332:       if (ctr[leer]=Ø) or
[2H4]  333:          ((zc=Ø) and (zs=Ø)) then begin
[W35]  334:         if ctr[computer]>ctr[mensch]
[7W4]  335:           then brettwert:=MaxInt else
[675]  336:           if ctr[mensch]>ctr[computer]
[874]  337:             then brettwert:=-MaxInt
[D53]  338:             else brettwert:=Ø;
[ØWØ]  339:         Exit;
[ZNØ]  34Ø:       end;
[KG4]  341:       fillchar(tab,SizeOf(tab),Ø);
[2M3]  342:       for i:=11 to 18 do begin
[LW2]  343:         inc(tab[1,f[i]]);
[D33]  344:         inc(tab[2,f[i+7Ø]])
[ZNØ]  345:       end;
[AF3]  346:       for i:=1 to 8 do begin
[KE3]  347:         inc(tab[3,f[1+i*1Ø]]);
[H93]  348:         inc(tab[4,f[8+i*1Ø]])
[ZNØ]  349:       end;
[2F3]  35Ø:       for i:=1 to 4 do begin
[BT3]  351:         if (tab[i,mensch]=Ø) and
[Z73]  352:            (tab[i,computer]>1)
[5P2]  353:           then inc(add,12Ø);
[K54]  354:         if (tab[i,computer]=Ø) and
[5U2]  355:            (tab[i,mensch]>1)
[KM2]  356:           then dec(add,12Ø);
[ZNØ]  357:       end;
[8Z3]  358:       if s=mensch then nb:=-nb;
[5S4]  359:       brettwert:=wert[computer]-
                             wert[mensch]+add-nb;
[ZNØ]  361:     end;
[WK5]  362:   function ZugWert( f:TSpielfeld;
[Y12]  363:                     s:TSpieler;
[R42]  364:                     t,p:integer;
[LK3]  365:                     a,b: integer):integer;
[6P3]  366:     var sf      : TSpielfeld;
[LL2]  367:         n,i,h   : integer;
[2T2]  368:         l       : TZugListe;
[HV2]  369:         cutoff  : boolean;
[MUØ]  37Ø:     begin
[AP4]  371:       ZugListenGenerator(f,s,n,l);
[BY2]  372:       if (n=Ø) then exit;
[L73]  373:       i:=1; cutoff:=false;
[V62]  374:       if s=computer
[DF4]  375:         then b:=startwert[mensch]
[755]  376:         else a:=startwert[computer];
[P31]  377:       repeat
[UYØ]  378:         sf:=f;
[3G5]  379:         ZugAusfuehrung(sf,s,l[i],false);
[5UØ]  38Ø:         if t=1
[XF4]  381:           then h:=brettwert(sf,s,p,n)
[V25]  382:           else h:=ZugWert(sf,gegner[s],
[5A2]  383:                           pred(t),n,a,b);
[EB1]  384:         case s of
[AC5]  385:          mensch  : (mensch minimiert (b))
[MUØ]  386:            begin
[624]  387:              if h<a then cutoff:=true;
[9K2]  388:              if h<b then begin
[4RØ]  389:                b:=h;
[ET3]  39Ø:                if t=maxt then z:=l[i];
[ZNØ]  391:              end;
[ZNØ]  392:            end;
[716]  393:          computer: (computer maximiert(a))
[MUØ]  394:            begin
[8D4]  395:              if t=maxt then mark(l[i],7);
[D24]  396:              if h>b then cutoff:=true;
[AK2]  397:              if h>a then begin
[2RØ]  398:                a:=h;
[W43]  399:                if t=maxt then begin
[6H3]  4ØØ:                  if z<>Ø then mark(z,7);
[Y63]  4Ø1:                  z:=l[i]; mark(z,14);
[ZNØ]  4Ø2:                end;
[ZNØ]  4Ø3:              end;
[ZNØ]  4Ø4:            end;
[ZNØ]  4Ø5:       end;
[951]  4Ø6:       inc(i);
[MF3]  4Ø7:     until (i>n) or cutoff;
[V62]  4Ø8:     if s=computer
[LS2]  4Ø9:       then zugwert:=a
[FV2]  41Ø:       else zugwert:=b;
[ZNØ]  411:   end;
[MUØ]  412:   begin
[B62]  413:     maxt:=4; z:=Ø;
[F35]  414:     if restfelder<2Ø then inc(maxt);
[B35]  415:     if restfelder<1Ø then inc(maxt);
[W35]  416:     if restfelder<6 then inc(maxt);
[KA3]  417:     ZugWert(f,s,maxt,1,
[HV3]  418:       startwert[computer],
[HL3]  419:       startwert[mensch])  ;
[ZNØ]  42Ø:   end;
[LV2]  421: ( Spielroutine )
[3V2]  422: procedure spiel;
[854]  423:   var feld      : tspielfeld;
[BH3]  424:       spieler    : tspieler;
[A93]  425:       ctrwait    : integer;
[SØ3]  426:   procedure punkte;
[EV3]  427:     var stand : STRING[8];
[NK2]  428:         c,m,i : integer;
[MUØ]  429:     begin
[TD2]  43Ø:       SetColor(red);
[SB3]  431:       m:=Ø; for i:=11 to 88 do

[AS4]  432:         if feld[i]=mensch then inc(m);
[HH2]  433:       str(m:2,stand);
[2R2]  434:       Bar(55Ø,22Ø,61Ø,26Ø);
[B14]  435:       OutTextXY(55Ø,22Ø,stand);
[4N2]  436:       SetColor(blue);
[4B3]  437:       c:=Ø; for i:=11 to 88 do
[CA5]  438:         if feld[i]=computer then inc(c);
[XG2]  439:       str(c:2,stand);
[GR2]  44Ø:       Bar(49Ø,22Ø,53Ø,26Ø);
[B14]  441:       OutTextXY(49Ø,22Ø,stand);
[L53]  442:       restfelder:=64-c-m;
[ØR4]  443:       if (c+m=64) or (ctrwait>1) then
[PF5]  444:         if c>m then status:=computer else
[594]  445:           if c<m then status:=mensch
[MZ3]  446:             else status:=jenseits;
[ØØØ]  447:
[ZNØ]  448:     end;
[HK3]  449:   procedure aussetzen;
[MUØ]  45Ø:     begin
[L23]  451:       SetColor(yellow);
[ER2]  452:       OutTextXY(48,314,
[8V2]  453:         'A U S S E T Z E N !');
[NN1]  454:       delay(1ØØØ);
[DK2]  455:       Bar(48,322,4ØØ,349);
[X82]  456:       Inc(ctrwait);
[ZNØ]  457:     end;
[4Ø4]  458:   procedure computerzug;
[FN2]  459:     var zug : TZug;
[EN3]  46Ø:         zugliste : TZugliste;
[S43]  461:         anzahl,i : integer;
[MUØ]  462:     begin
[CZ1]  463:       SetColor(7);
[LL2]  464:       CASE RANDOM(5) OF
[WH4]  465:         Ø : OutTextXY(49Ø,26Ø,'Hmm..');
[4M4]  466:         1 : OutTextXY(49Ø,26Ø,'Tja,...');
[RL4]  467:         2 : OutTextXY(49Ø,26Ø,'Ähm...');
[DS4]  468:         3 : OutTextXY(49Ø,26Ø,'Pffft..');
[VH4]  469:         4 : OutTextXY(49Ø,26Ø,'Ooh...');
[ZNØ]  47Ø:       END;
[A84]  471:       ZugListenGenerator(feld,
[4N4]  472:         computer,anzahl,zugliste);
[9E3]  473:       for i:=1 to anzahl do
[ZE3]  474:         mark(zugliste[i],Ø);
[9P1]  475:       delay(8ØØ);
[TK4]  476:       ZugWahl(feld,spieler,zug);
[BR2]  477:       Bar(49Ø,26Ø,62Ø,3ØØ);
[171]  478:       if zug=Ø
[ZG2]  479:         then Aussetzen
[WP1]  48Ø:         else begin
[322]  481:                mark(zug,14);
[ØH3]  482:                ZugAusfuehrung(feld,
[F83]  483:                  spieler,zug,true);
[5Y1]  484:                mark(zug,7);
[YZ1]  485:                ctrwait:=Ø;
[ZNØ]  486:              end;
[ZNØ]  487:     end;
[GT3]  488:   procedure spielerzug;
[TN4]  489:     var zug_iste    : tzugliste;
[AL3]  49Ø:         i,nr,anzahl : integer;
[MUØ]  491:     begin
[A84]  492:       ZugListenGenerator(feld,
[H24]  493:         mensch,anzahl,zugliste);
[CH2]  494:       if anzahl=Ø then
[5B4]  495:         begin aussetzen; exit end;
[9E3]  496:       for i:=1 to anzahl do
[KL3]  497:         mark(zugliste[i],14);
[P31]  498:       repeat
[BF3]  499:         mousex:=Ø; mousey:=Ø;
[PT2]  5ØØ:         WaitMouseClick;
[NØ5]  5Ø1:         if (mousex>46Ø) and (mousey>31Ø)
[ZN1]  5Ø2:           then begin
[RF2]  5Ø3:             case mousex of
[Y74]  5Ø4:               46Ø..54Ø : zustand:=beenden;
[YL3]  5Ø5:               541..639 : zustand:=neu;
[ZNØ]  5Ø6:             end;
[ØWØ]  5Ø7:             exit;
[ZNØ]  5Ø8:           end;
[8G3]  5Ø9:         nr:=(mousex div 55+1)+
[B53]  51Ø:             (mousey div 4Ø+1)*1Ø;
[UPØ]  511:         i:=Ø;
[P31]  512:         repeat
[XXØ]  513:           inc(i)
[554]  514:         until (zugliste[i]=nr) or
[TU1]  515:               (i>anzahl);
[FB4]  516:         if i>anzahl then write(#7);
[YØ4]  517:       until (zugliste[i]=nr);
[UZ4]  518:       ZugAusfuehrung(feld,spieler,
[CF1]  519:         nr,true);
[YZ1]  52Ø:       ctrwait:=Ø;
[9E3]  521:       for i:=1 to anzahl do
[MF3]  522:         mark(zugliste[i],7);
[ZNØ]  523:     end;
[1R3]  524:   procedure neuesSpiel;
[2Ø3]  525:     var x,y : integer;
[MUØ]  526:     begin
[9P2]  527:       if random(1ØØØ)>488
[5G3]  528:         then spieler:=mensch
[T54]  529:         else spieler:=computer;
[B23]  53Ø:       zustand:=spielen;
[B54]  531:       status:=leer; ctrwait:=Ø;
[Z53]  532:       loesche_spielfeld;
[KX3]  533:       neues_spielfeld(feld);
[WA3]  534:       feld_anzeigen(feld);
[5C1]  535:       punkte;
[ZNØ]  536:     end;
[MUØ]  537:   begin
[P31]  538:     repeat
[642]  539:       neuesSpiel;
[P31]  54Ø:       repeat
[3M5]  541:         if (zustand=neu) then neuesSpiel;
[2D3]  542:         if spieler=computer
[NZ2]  543:           then ComputerZug
[ZW2]  544:           else SpielerZug;
[1H4]  545:         spieler:=Gegner[spieler];
[5C1]  546:         punkte;
[5Ø4]  547:       until (status<>leer) or
[K23]  548:             (zustand=beenden);
[4V4]  549:       if zustand=beenden then exit;
[ØØØ]  55Ø:
[ØN4]  551:       SetTextStyle(TriplexFont,
[RY1]  552:         HorizDir,4);
[L23]  553:       SetColor(yellow);
[7H2]  554:       case status of
[794]  555:        mensch   : OutTextXY(52,314,
[ØS3]  556:                   'Ok, Sie sind SIEGER !');
[YH4]  557:        computer : OutTextXY(72,314,
[BA3]  558:                   'Ich hab'' gewonnen');
[RH4]  559:        jenseits : OutTextXY(92,314,
[PY2]  56Ø:                   'UNENTSCHIEDEN!');
[ZNØ]  561:       end;
[PT2]  562:       WaitMouseClick;
[VH2]  563:       Bar(5Ø,314,4ØØ,349);
[TØ4]  564:     until (zustand=beenden);
[ZNØ]  565:   end;
[MUØ]  566: begin
[DV1]  567:   mouseinit;
[AVØ]  568:   init;
[821]  569:   spiel;
[522]  57Ø:   closegraph;
[KNØ]  571: end.
```

Das vollständige Reversi-Programm setzt sich aus den Überlegungen dieses Beitrags zusammen.

Das Spiel der Könige

Schachmatt

Kein Spiel der Welt ist gleichzeitig so populär und doch so anspruchsvoll wie Schach. Dabei müssen Sie kein Schachprofi sein, um den Reiz des Spiels zu entdecken. Die Programmierung eines Schachspiels vertieft Ihre Kenntnisse der Strategiespielprogrammierung ebenso wie das Verständnis um das Schachspiel selbst.

Anfang März 1949 veröffentlichte Claude Shannon seinen Aufsatz »Programmieren eines Digital-Rechners zum Schachspiel«. Damit nahm die Geschichte des Computerschachs ihren Anfang. Die Grunderkenntnisse aus Shannons Aufsatz gelten noch heute. Er unterteilte alle Schachprogramme in zwei elementare Typen.

● **Der A-Typ:** Zunächst benannte er den Typ A, der alle denkbaren Züge und Gegenzüge kompromißlos durchrechnet. Diese Shannon-A-Klasse bezeichnet man auch als Brute-force Typ (»rohe Kraft«). Aus den bisherigen Beiträgen zur Strategiespielprogrammierung wissen Sie bereits, daß bei durchschnittlich 40 in Frage kommenden Antworten auf einen Zug die Zahl der Zugkombinationen explosionsartig ansteigt.

● **Der B-Typ:** Da bei dieser Methode sehr viel Zeit für »unsinnige« Züge verbraucht wird, ist es erstrebenswert, nur eine Auswahl von besseren Zügen zu untersuchen. Programme dieser Art faßt Shannon unter dem B-Typ zusammen. Wie ein Typ-B-Algorithmus allerdings die zu untersuchenden Züge selektieren soll, ließ Shannon offen.

● **Der Turing-Test:** Ebenfalls große Erfolge auf dem Gebiet der Schachprogrammierung hat der Mathematiker Alan Turing erzielt. Bei ihm scheiterte die Funktionstüchtigkeit eines Schachprogramms nur an den eingeschränkten Fähigkeiten der damaligen Rechner. Interessanterweise ist es mit einem Schachprogramm auch erstmals gelungen, den sogenannten Turing-Test erfolgreich zu absolvieren. Dieser Test ist eine Herausforderung an die Künstliche Intelligenz: Ein Außenstehender soll entscheiden, ob er mit einem Menschen oder mit einem Computer kommuniziert (hier: Schach spielt). Ist eine Entscheidung ausgeschlossen, ist der Test positiv ausgefallen. Allen Bemühungen zum Trotz ist es bisher nur auf dem Gebiet der Schachprogrammierung gelungen, diesen Test erfolgreich zu absolvieren: eine geschichtsträchtige Angelegenheit also, die Schachprogrammierung.

Die wesentlichen Techniken der Programmierung von Strategiespielen haben Sie bereits in den vorangegangenen Beiträgen kennengelernt – falls nicht, so lesen Sie zuerst den Beitrag über die Grundlagen der Strategiespielprogrammierung »Tic Tac Toe«. Viele Begriffe, wie etwa das Minimax-Prinzip, der Spielbaum oder der Stellungswert, gehen auf Shannon zurück. Besonders im Schachspiel ist es sehr schwer, eine selektive Zugauswahl zu treffen, weshalb das Schachprogramm dieses Beitrags sich in die Mehrzahl der Shannon-A-Programme einreiht. Das hindert Sie natürlich nicht daran, zeitsparende Verfahren wie die Alpha-Beta-Suche zu benut-



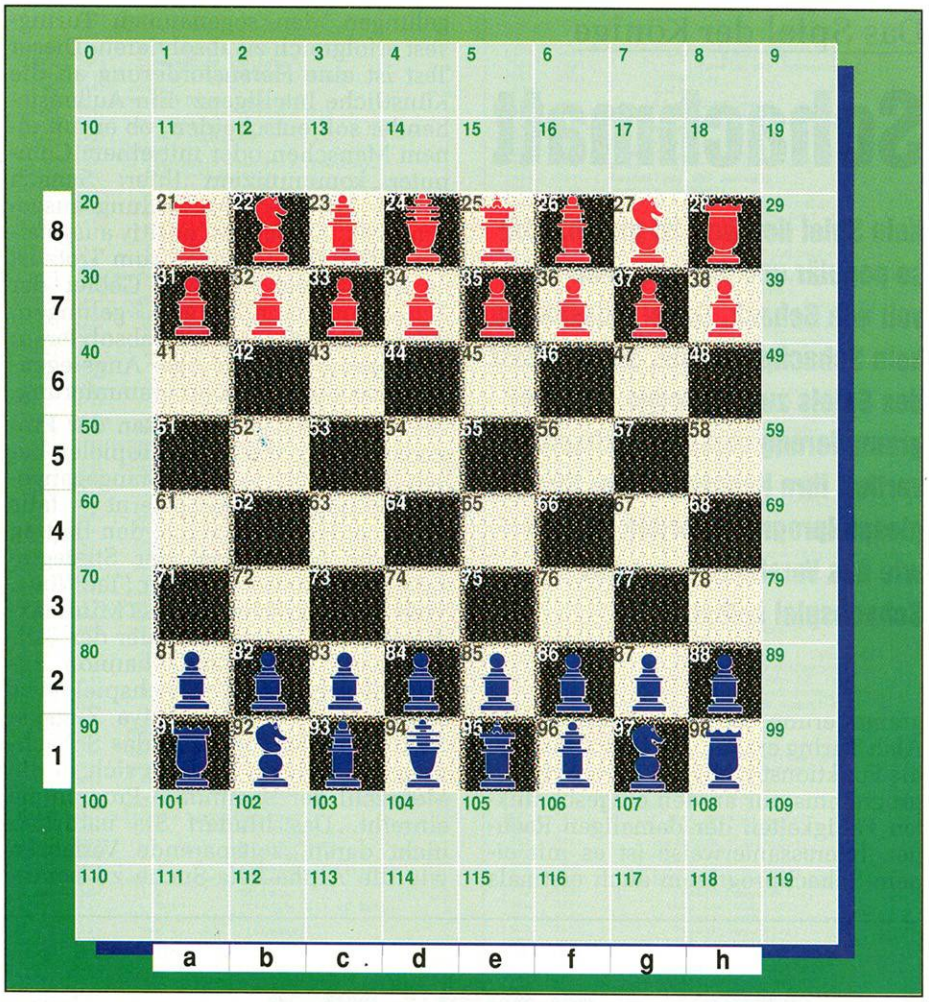


Bild 1. In dieser Datenstruktur speichert das Programm das Schachbrett.

zen, so daß die Antwortzeiten des Computers in einem erträglichen Rahmen bleiben.

Wie auch beim Strategiespiel Reversi, stellen sich Ihnen beim Schach drei Hauptprobleme: Wie ermitteln Sie die denkbaren Züge? Wie bewerten Sie die Stellungen? Wie beschleunigen Sie das Programm?

● **Das Spielfeld:** Entscheidend für fast alle dieser Fragen ist die interne Darstellung des Spielbretts. Es ist sehr wahrscheinlich, daß Sie aufgrund späterer Einsichten die Definition Ihres Spielfeldes ändern. Meistens legen die vielen unterschiedlichen Problemstellungen der Spieleprogrammierung verschiedene Brettdarstellungen nahe. Oft ist ein Kompromiß notwendig, um alle Probleme zufriedenstellend zu lösen. Gerade im Schachspiel ist die Beziehung verschiedener Spielfiguren zueinander von großer Bedeutung. Von diesem Gesichtspunkt aus ist ein internes Spielbrett gar nicht notwendig. Statt dessen benötigen Sie eine Reihe von Listen, in denen Sie bestimmte Figuren rasch finden, ohne vorher das gesamte Brett nach ihnen absuchen zu müssen. Für die Bestimmung der denkbaren Züge hingegen spielt es keine Rolle, ob ein Feld von einem gegnerischen Bauer oder der feindlichen Dame besetzt ist – schlagen dürfen Sie so oder so. Der Zug bleibt formal derselbe. Für den Zuggenerator ist also eine herkömmliche, ans Spielbrett angelehnte Darstellung sinnvoll. Großrechner, die intern mit 64-Bit-Zahlen umgehen, speichern in Bitmaps Informationen über das Spielbrett. Jedes der 64 Bits steht für ein Feld auf dem Brett. Für jeden Figurtyp und jeden Spieler steht nun eine solche Bitmap zur Verfügung, auf der entsprechend der Lage auf dem Spielbrett die Bits gesetzt sind. Durch einfache logische Vergleiche ermitteln diese Rechner dann rasend schnell die notwendigen Informationen. Falls Sie einen 386er besitzen, sind Sie an den 64 Bit schon recht dicht dran, aber in diesem Beitrag geht es nicht um die Programmierung eines hochgezüchteten 386er-Schachs.

Die ersten Vorüberlegungen haben gezeigt, daß sich sowohl die Listen- als auch die Brettdarstellung für Ihre Zwecke eignet. Benutzen Sie vorzugsweise die Brettdarstellung, da es einfacher ist, sie aktuell als ein Dutzend Listen auf dem laufenden zu halten. Aber sobald es ernst wird, ermitteln Sie aus diesem Spielbrett durch eine Hilfsroutine die entsprechenden Listen, um auf deren Vorzüge nicht verzichten zu müssen.

Eindimensionale Verwaltung des Schachbretts

Bild 1 zeigt die interne Brettdarstellung. Wenn Sie den Beitrag über Reversi gelesen haben, kommt Ihnen die eindimensionale Organisation sicherlich vertraut vor. Neu ist allerdings, daß das Brett zu Beginn und am Ende zwei Spielfeldzeilen »zuviel« besitzt. Sie erinnern sich, weshalb dieser zusätzliche Rand eingeführt wurde. Bei der Beurteilung, ob ein Zug legal ist, führen Sie ihn zunächst einfach aus. Landen Sie auf einem leeren oder gegnerischen Feld, so ist der Zug ausführbar. Vorsicht ist am Spielfeldrand geboten, damit Sie nicht aufgrund der li-

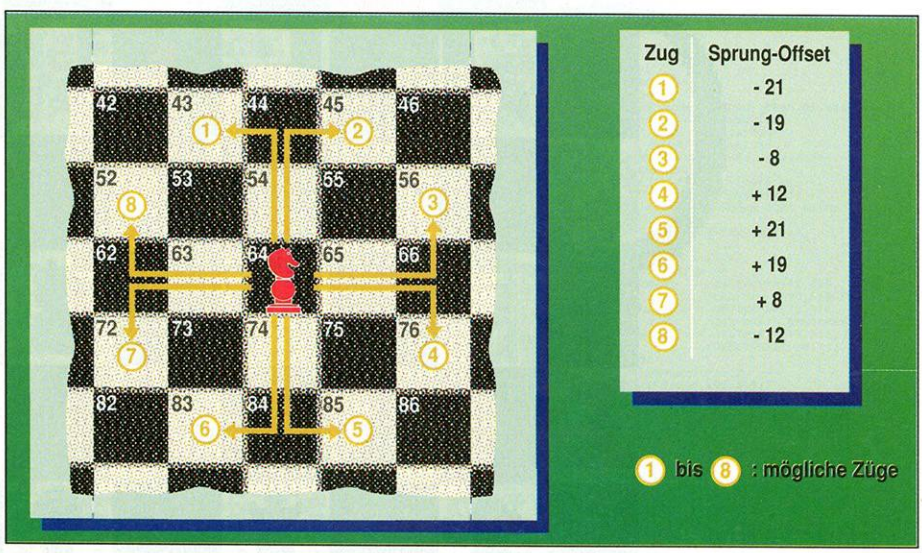


Bild 2. Aus dieser Skizze entstehen die Konstanten, mit denen Sie die Springerzüge ermitteln.

nearen Organisation des Speichers regelwidrig auf der gegenüberliegenden Spielfeldseite eintreten. Hier sind umfangreiche Abfragen notwendig, die bei Erreichen des Spielfeldrandes die weitere Suche abbrechen.

Das Spielfeld wird um einen Rand erweitert

Den komplizierten Abfragen gehen Sie durch die zusätzlichen Felder aus dem Weg, denn der erste Schritt aus dem Spielfeld heraus führt immer in diese Zusatzfelder. An deren Belegung erkennen Sie sehr leicht, ob Ihr Zug regelwidrig war. Da beim Schach ein Springerzug stets über zwei Felder führt, tragen Sie dem durch Ausweiten der Zusatzfelder Rechnung. Mit diesen Korrekturen bläht sich das Spielfeld zwar insgesamt um fast das Doppelte auf, aber die Vorteile bei der Programmierung des Zuggenerators wiegen den Speicherverlust wieder auf. Außerdem verbuchen Sie dadurch noch Geschwindigkeitsvorteile, für die Sie während des Spiels dankbar sein werden.

● **Die Spielfiguren:** Jetzt ist es an der Zeit, die Darstellung der Spielfiguren festzulegen. Einem leeren, aber gültigen Feld geben Sie die Kennziffer 7. Auf die sechs schwarzen Spielfiguren verteilen Sie die Nummern 1 bis 6, die sechs weißen Figuren kodieren Sie mit Werten größer als 7. Ein einziger Vergleich mit der Ziffer für das leere Feld zeigt Ihnen dann, ob es sich beim betrachteten Feld um eine weiße (größer als 7) oder schwarze (kleiner als 7) Figur handelt. Für die Zusatzfelder außerhalb des bespielbaren Bereichs legen Sie 0 fest.

● **Der Zuggenerator:** Nun ist die Programmierung des Zuggenerators nur noch eine Fleißarbeit. Jede Figur darf sich nur nach bestimmten Regeln bewegen. Sie probieren der Reihe nach alle Bewegungsvarianten aus, und falls das Zielfeld nicht außerhalb des Schachbretts, aber leer oder mit einem gegnerischen Stein besetzt ist, so war der Zug legal. Alle legalen Züge tragen Sie dann in eine Zugliste ein. Den Aufbau einer solchen Zugliste sehen Sie im Listing an der Struktur »Zug«. Sie speichern für jeden Zug Startfeld (»von«) und Zielfeld (»nach«).

Da das Spielfeld trotz der Erweiterungen nur 120 Byte groß ist, reichen hierfür Byte-Variablen. Der Eintrag »Option« enthält bei einigen Zügen Zusatzinformationen. Bei Bauernumwandlungen finden Sie hier die Figur, in die der Bauer verwandelt werden soll. Außerdem markieren Sie mit dieser Variable Schlag- und Rochadenzüge. Die Bedeutung dieser Markierung sowie der beiden noch folgenden Felder »Next« und »Value« ist später bei der Suche nach dem besten Zug von Bedeutung.

Die Bauern: Wenn Sie die Prozeduren zur Erzeugung der Zuglisten studieren, fällt Ihnen sicherlich auf, daß es für weiße und schwarze Bauern zwei fast identische Routinen gibt. Warum zwei separate Routinen? Die Regeln sind doch für beide Bauern gleich. Die Begründung liegt hier in der Wahl der Programmiersprache. Das komplette Schachprogramm ist in Assembler geschrieben. Das bringt neben der gewünschten hohen Geschwindigkeit aber auch eine hohe Fehleranfälligkeit mit sich. In Pascal oder C einfach anmutende Konstruktionen fallen in Assembler schwer und verschleiern den Blick für das Wesentliche. Ein Beispiel: Unbewegte Bauern dürfen zu Beginn einen Doppelschritt machen. Woran merkt Ihr Zuggenerator, daß der Bauer noch nicht bewegt wurde? Nun, er steht noch auf seiner Ursprungsreihe, werden Sie antworten.

Legale Züge werden in der Zugliste gespeichert

Das ist aber für den schwarzen Bauern die zweite Reihe, für den weißen Bauern die siebte.

Wann ist eine Bauernumwandlung durchführbar? Sobald die gegnerische Grundreihe erreicht ist. Wieder ist dies für schwarze Bauern eine andere Reihe als für weiße. Ein Bauer kann ferner nur in eine Richtung ziehen: nach vorne. Erneut ist diese Richtung abhängig von der Farbe des Spielers. Sie sehen, daß sich die Bauernzüge aus der Sicht des Zuggenerators sehr wohl unterscheiden. Zwar nur in wenigen Details, aber doch an genügend Stellen, um die Prozeduren unübersichtlich werden zu lassen, wenn Sie jede Kleinigkeit an Ort und Stelle neu entscheiden. Deshalb erleichtern Sie die Lesbarkeit und vermindern die Fehleranfälligkeit, wenn Sie in solchen Fällen zwei Routinen schreiben. Außerdem interessieren Sie bei der Aufstellung einer Zugliste immer nur die Züge der Figuren einer bestimmten Farbe. Wenn Sie die Zugliste für Schwarz aufbauen, rufen Sie den Bauernzuggenerator nur für Felder mit schwarzen Bauern auf.

Damit sind die Detailfragen eigentlich schon geklärt. Jede erneute Abfrage der Farbe, die bei einer allgemeinen Routine nötig wäre, kostet Zeit. Ihr Zuggenerator ist einer der am häufigsten durchlaufenen Programmabschnitte, also macht sich jeder noch so kleine Geschwindigkeitsgewinn später bezahlt. Beachten Sie aber, daß Sie Korrekturen unter Umständen an zwei Stellen vornehmen müssen: einmal für jede Farbe.

Zwei Zuggeneratoren für die Bauern zur Übersichtlichkeit

Dame, Turm und Läufer: Weniger kritisch sind die Teile des Zuggenerators, die Turm, Läufer oder Dame betreffen. Hier gibt es keine Sonderregeln, die in Ihre Arbeit als Programmierer Stolpersteine legen. Die Züge der Dame erschlagen Sie sogar ganz elegant, indem Sie nacheinander den Turm- und den Läuferzuggenerator aufrufen.

Die Springer: Die Springerzüge sind etwas komplizierter als die bisher betrachteten Züge. Im Schachprogramm sind aber auch sie nur eine Handvoll Konstanten, die Sie zur jeweiligen Springerposition addieren. Bild 2 zeigt die Werte für die acht denkbaren Züge des Springers.

Der König: Eigentlich sollten die Königszüge für Schwarz und Weiß identisch sein, wäre da nicht noch die Rochade als Sonderfall. Eine Rochade ist nur möglich, solange der König und der beteiligte Turm unbewegt bleiben, die Felder zwischen Turm und König frei sind und der König während seines Zuges nicht bedroht ist. Das sind recht umfangreiche Forderungen, bei denen wieder Einzelheiten von der Farbe des rochierenden Königs abhängen.

Die Rochade erfordert einige Überprüfungen

Besonders brisant ist die Forderung, daß der König während seines Zuges nicht bedroht sein darf. Dies ist keine Frage, die sich mit ein paar Abfragen beantworten läßt. Grundsätzlich bieten sich zwei Methoden an: Sie können den Zuggenerator ein weiteres Mal aufrufen und überprüfen, ob ein Zug existiert, der den König schlägt.

Allerdings ist der Aufwand des Zuggenerators sehr hoch, berechnet er doch viel mehr Züge, als Sie in diesem Fall interessieren. Die andere Variante geht von der Königsposition aus und überprüft »rückwärts« gezielt die Positionen, auf denen eine gegnerische Figur stehen müßte, wollte sie den König bedrohen. Der Aufwand bleibt immens, da wegen der hohen Mobilität von Dame, Läufer und Turm viele Felder in Frage kommen, liegt aber dennoch unter dem des Zuggenerators. Im Listing finden Sie die entsprechende Prozedur unter dem Namen »KoenigImSchach«. Im Rochadenzuggenerator wird das letzte Feld des Königszuges nicht auf eine Bedrohung untersucht. Sie können an dieser Stelle auf die

Überprüfung verzichten, da dies ohnehin einen regelwidrigen Zug darstellt. Schließlich darf der Spieler seinen eigenen König nicht so stellen, daß er vom Gegner bedroht ist. Eine Schachsituation kann grundsätzlich aber aus jedem anderen Zug aus der Zugliste entstehen.

Es wäre sicher unsinnig, die Frage nach einem drohenden Schach bereits zu klären, bevor Sie einen Zug in die Zugliste eintragen. Denn die Zahl solcher Züge ist im allgemeinen sehr gering, der Aufwand der Überprüfung jedoch sehr hoch. Aber im Rahmen der Suche nach dem besten Zug rufen Sie ohnehin bis zu einer bestimmten Suchtiefe Zug- und Gegenzuggenerator auf. Damit sind Sie in der Lage, nach der ersten der angesprochenen Methoden die Bedrohung des Königs zu erkennen. Im Listing ist die Routine »KoenigBedroht« dafür verantwortlich. Sie wird einmal beim Eintreten in eine neue Suchtiefe im Anschluß an den Zuggenerator aufgerufen. Ist der König bei dieser Stellung bedroht, wird jede weitere Berechnung wertlos, da ein regelwidriger Zug die Grundlage der Betrachtungen bildet. Das Programm bricht die Suche an dieser Stelle ab und liefert die denkbar schlechteste Bewertung für diesen Zug. Damit stellen Sie sicher, daß der Zug niemals gewählt wird.

Um noch einmal auf die Brettdarstellung zurückzukommen: An dieser Stelle wünschen Sie sich als Programmierer eine Art von Bedrohungslisten, die für jedes Spielfeld angeben, durch welche gegnerischen Figuren es gedeckt ist. Eine inkrementale Verwaltung solcher Listen ist sicherlich auch wesentlich schneller als die ständig neue Suche nach bedrohenden Figuren. Allerdings benötigen Sie zweimal 64 solcher Bedrohungslisten (für jedes Feld und jeden Spieler). Dies bedeutet großen Speicheraufwand und eine ganze Menge Programmierfleiß.

Durch den Einsatz von mehr Speicher läßt sich offensichtlich Zeit sparen. Der benötigte Speicher für die Darstellung der Stellung schlägt aber gleich mehrfach zu Buche. Denn bei dem wiederholten rekursiven Aufruf der Suchroutine brauchen Sie in jeder Rekursionstiefe alle Variablen noch einmal.

Das Suchergebnis jeder Stufe wird gespeichert

In Pascal übernimmt der Compiler die Organisation dieser rekursiven Variablen, in Assembler müssen Sie sich selbst darum kümmern. Üblicherweise reservieren Sie sich einen großen Stack für solche Speicheranforderungen. Im Schachprogramm dieses Beitrags ist der Speicher jedoch als »künstlicher Stack« im normalen Datenspeicher angelegt. Dadurch wissen Sie immer, wo sich die Variablen der jeweiligen Suchtiefe befinden. Der Abstand von einem kompletten Spielstand zum nächsten ist in diesem »Stack« stets konstant. Auf dem richtigen Stack legt das Programm zusätzlich Sprungadressen und Zwischenergebnisse ab, die den gezielten Zugriff auf bestimmte Suchtiefen nicht zulassen. Die Struktur »Spielstand« umfaßt alle Informationen, die zur Beschreibung des jeweiligen Spielverlaufs notwendig sind. Acht dieser Spielstrukturen bilden den »GameStack«. Die Variable »AktStack« ist der Index, der auf den jeweils aktuellen Spielstand im »GameStack« zeigt. Ohne zusätzliche Variablen einzuführen, erkennen Sie sofort, daß Sie sich in Suchtiefe 0 befinden, wenn »AktStack« und der Offset der Adresse »GameStack« übereinstimmen.

Bis zu dieser Entwicklungsstufe brauchten Sie keinerlei Schacherfahrung, die Zugvariationen für die einzelnen Figuren entnehmen Sie direkt den Spielregeln.

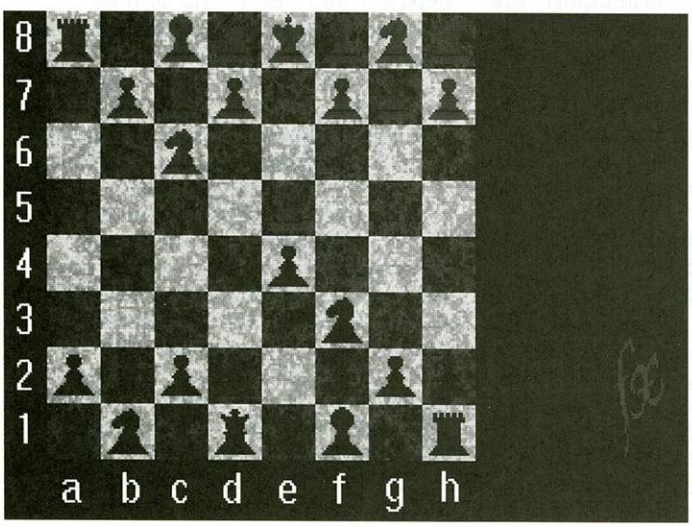


Bild 3. Bei konstanter Suchtiefe von nur einem Halbzug droht bei dieser Stellung dem Spieler der blauen Figuren eine Fehlentscheidung.

● **Die Bewertung der Spielsituation:** Anders sieht es jedoch aus, wenn es nun an die Bewertung einer Spielsituation mit konkreten Zahlenwerten geht. Um hier zu vernünftigen Ergebnissen zu gelangen, die schließlich die Stärke des Schachprogramms ausmachen, brauchen Sie ein gehöriges Maß an Erfahrung im Schachspiel. »Das große Computerschachbuch« (Data Becker) der Autoren Bartel, Kraas und Schrüfer bietet eine beispielhafte Diskussion von Bewertungskriterien für ein Schachspiel. Die dort genannten Kriterien sind auch im Listing herangezogen worden. Wenn Ihnen der folgende Einblick nicht ausreicht, sei Ihnen das Buch sehr empfohlen. Darin finden Sie die Erfahrungswerte vieler Schachprogrammierer verewigt, die Sie allein niemals alle herausfinden könnten – schließlich wollen Sie ja irgendwann auch mal gegen Ihr Programm spielen.

Natürlich müssen Sie jede Figur einzeln und abhängig vom Figurtyp bewerten. Dabei unterscheiden Sie grundsätzlich zwischen materieller und positioneller Bewertung. Unter der materiellen Bewertung verstehen Sie die Punkte, die Sie allein für die Existenz der Figuren vergeben.

Bewertet werden Materialwert und Stellung

Hier ist Ihnen die Dame natürlich mehr Punkte wert als ein Bauer. In die Bewertung der Stellung geht dann die Differenz der Materialsummen beider Spieler ein. Wie schon beim Reversi-Spiel reichen diese wenigen Informationen noch nicht aus, um einer vernünftigen Einschätzung der individuellen Spielsituation zu genügen. Es kommen also positionelle Bewertungen hinzu, mit denen Sie die Lage Ihrer Figuren untereinander, aber auch zum Gegner berücksichtigen. An dieser Stelle ist es wichtig, sich darüber klar zu werden, in welchem Verhältnis beide Bewertungsarten zueinander stehen sollen. Auf keinen Fall darf eine positionelle Bewertung beispielsweise den Verlust des Königs ausgleichen. Was nützt eine tolle Stellung, wenn Sie verloren haben? Sie können sich natürlich entschließen, gewisse materielle Opfer für außerordentliche positionelle Vorteile zu bringen. Im Schachprogramm dieses Beitrags hingegen ist die positionelle Bewertung so ausgelegt, daß sie den Materialwert eines Bauern nicht überschreitet. Damit stellen Sie sicher, daß das Programm mit seinen Figuren wenigstens sorgsam umgeht.

Der Wert der Bauern: Bei der Bewertung der einzelnen Spielfiguren verdienen die Bauern besondere Beach-

tung. Das mag Sie zunächst verwundern, denn schließlich haben die Bauern den geringsten Wert. Aber dafür besitzen Sie acht solcher Figuren, die geschickt eingesetzt eine große Schlagkraft bringen können. Beim Endspiel sind Sie für jeden zusätzlichen Bauern dankbar – und damit Sie dann überhaupt noch Bauern haben, verdienen diese bereits von Anfang an besondere Beachtung.

Bauern lassen sich leicht in ein Schema pressen

Außerdem sind sie durch ihren eingeschränkten Bewegungsraum recht starr und müssen mit Bedacht eingesetzt werden. Alle Bauernzüge sind endgültig, ein Zurück gibt es nicht. Aus diesem Grund existieren für die Bauernbewertung auch fundierte und erprobte Bewertungstheorien. Die Figurentypen Springer, Läufer, Turm und Dame lassen sich hingegen schlecht in ein starres Wertesystem zwängen, dazu ist die Flexibilität dieser Figuren zu hoch. Eine genaue Bewertung wäre unendlich detailliert, eine ungenaue unterbindet die volle Entfaltung.

Welche Kriterien gibt es also für die Bauern? Da das Zentrum – also die Felder D4, D5, E4 und E5 – eine Schlüsselrolle im Schachspiel einnehmen, bewerten Sie eine Blockierung Ihrer Bauern auf der D- und E-Linie negativ. Ferner liegt die Stärke der Bauern in ihrem massiven Auftreten. Wenn einem Bauern also linker und rechter Nachbar fehlen (isolierter Bauer), bedeutet das ebenfalls einen Nachteil für die eigene Stellung. Aus der Tatsache, daß Bauern nur schräg schlagen, folgt für ihre Bewertung, daß mehrere Bauern auf einer Linie weniger Wert haben, als wenn Sie nebeneinander stehen. Nur so können sie sich gegenseitig decken. Die Blockierung des vorderen Bauern reicht zudem aus, alle Bauern dieser Linie lahmzulegen. Solche »Mehrfachbauern« sind also in der Bewertung ebenfalls mit Abzügen zu bedenken.

Schlecht ist auch eine ungleichmäßige Entwicklung der Bauernfront. Konkret sind rückständige Bauern zu vermeiden, die beim Weiterrücken durch gegnerische Bauern bedroht werden, und denen keine noch weiter zurückliegenden eigenen Bauern Unterstützung bieten.

Sie gewähren hingegen Pluspunkte für Freibauern. Denn sie können durch gegnerische Bauern weder geschlagen noch blockiert werden.

Generell positiven Einfluß hat während des Spiels das Vorrücken der mittleren Bauern, um gegnerischen Figuren im Zentrum die Entfaltung zu erschweren. Gegen Ende der Partie ist allerdings das Vorrücken aller Bauern wünschenswert, um rasch in den Genuß einer wertvollen Bauernumwandlung zu kommen.

Positiv ist das Vorrücken mittlerer Bauern

Für die Umsetzung dieser Kriterien in Assembler sind diverse Listen über Bauernpositionen in Abhängigkeit von der jeweiligen Linie oder Reihe erforderlich. Diese Listen stellt die Prozedur »Analyse« den Routinen »BauernWertWeiss« beziehungsweise »BauernWertSchwarz« zur Verfügung.

Die Springer-Bewertung: Der Aktionsradius des Springers beispielsweise ist immer auf maximal acht Züge beschränkt, je nach eigenem Standort. Seine größte Wirkung entfaltet er folglich in Zentrumsnähe. Hinzu kommt, daß er auf der Grundreihe die Entwicklung des eigenen Königs behindert, indem er die Rochade blockiert.

Der Wert der Läufer: Dasselbe gilt natürlich auch für die Läufer. Bei ihnen kommt hinzu, daß ein Läufer stets nur Felder einer Farbe beherrscht. Den Besitz des kompletten Läuferpaares sollten Sie also honorieren. Der Aktionsradius des Läufers hängt – ebenso wie bei Turm und Dame – nicht nur von seiner eigenen Position, sondern auch vom Umfeld ab. Der Springer dagegen bewegt sich notfalls auch über andere Figuren hinweg. Das erlauben in diesem Fall die Regeln nicht. Sinnvoll sind also Läuferstellungen, bei denen Sie viele Felder bedrohen. Die Anzahl der Felderkontrollen ermittelt indirekt der Zuggenerator, indem er alle erreichbaren Felder der Figur in die Zugliste einträgt. Es bietet sich an, diese Bewertung bereits durch den Zuggenerator zu erledigen. Dabei gewichten Sie einzelne Felderkontrollen in Abhängigkeit vom Abstand zum Zentrum, da besonders die Zentrumskontrolle die gegnerische Entwicklung stört.

Die Turm-Bewertung: Der Turm entfaltet seine Fähigkeiten besonders leicht, wenn er auf einer Felderreihe steht, in der keine beziehungsweise keine eigenen Bauern stehen. Ansonsten ist seine vertikale Bewegungsfreiheit stark eingeschränkt. Stehen zudem beide Türme in einer Linie, so erlauben sie eine optimale Vorbereitung des eigenen Angriffs auf gegnerische Reihen. Gelingt es einem Ihrer Türme, auf die gegenüberliegende Bauernreihe vorzudringen, erlangen Sie die Kontrolle über die feindliche Grundreihe. Steht der gegnerische König noch auf derselben, so nehmen Sie ihm durch Ihren Turm beinahe jede Bewegungsfreiheit. Das sollte Ihnen ein paar Pluspunkte wert sein.

Die Stellung der Dame: Zusätzlich nutzen Sie für die Bewertung des Turmes, wie auch der Dame, die bereits beim Läufer eingeführte Anzahl der kontrollierten Felder.

Da Sie der Verlust Ihrer Dame empfindlich trifft, sollten Sie diese erst einsetzen, wenn durch andere Figuren genügend Raum gewonnen ist. Sonst müssen Sie die Dame ständig vor gegnerischen Figuren in Sicherheit bringen und vernachlässigen die Entwicklung Ihrer Leichtfiguren. Eine zu frühe Aktivierung bestrafen Sie daher mit Punktabzug.

Der König: Er greift aktiv meistens erst im Endspiel ins Geschehen ein. Seine »Unersetzbarkeit« veranlaßt Sie, ihn nicht zu gefährden. Entsprechend unterbinden Sie ein Vorrücken des Königs, solange Ihr Gegner noch über starke Figuren verfügt. Kommt eine Rochade noch in Frage, so bedeutet dies meistens ein hohes Sicherheitspotential. Dies ist abhängig davon, wie stark der Bauernschutz der jeweiligen Seite noch geblieben ist.

Im Endspiel schließlich sollten Sie sehen, daß Sie mit dem König in Zentrumsnähe ziehen, um nicht in eine Spielecke gedrängt zu werden, aus der es kein Entrinnen mehr gibt. Da die Bauern oft die letzten eigenen Figuren sind, sollte sich der König in deren Nähe aufhalten, um die Bauernumwandlung voranzutreiben. Wenn es um eine Umwandlung geht, so ist der König natürlich besonders gut beraten, wenn er sich an seine Freibauern hält.

● **Das Zeitproblem bei der Suche:** Stellen Sie sich vor, der Computer würde eine Zugfolge untersuchen, an dessen Ende er eine gegnerische Figur schlägt. Für die Suchtiefe von nur einem Zug zeigt Bild 3 solch eine Stellung. Der Zug »Springer F3 schlägt Bauer E5« gibt viele Punkte, weil der Materialverlust des Gegners jeden positionellen Vorteil bei der Bewertung übertrifft.

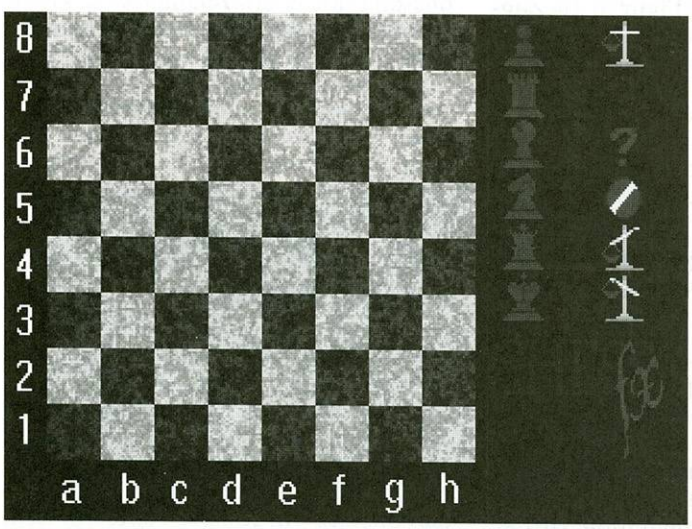
Beim Schlagen wird die Suchtiefe erhöht

Ihr Programm wählt also diesen Zug aus, ohne zu sehen, daß die eben noch schlagende Figur einen Zug später selbst geschlagen wird – Springer C6 schlägt E5. So viel Material besitzen Sie im Schachspiel jedoch nicht, als daß Sie sich ein solches Fehlverhalten leisten könnten.

Statt dessen sollten Sie bei Schlagzügen den gesetzten Suchhorizont ignorieren und die Schlagfolgen bis zum Ende durchprobieren. Da unter Umständen sehr lange Abtauschfolgen

entstehen, kann der Zeitaufwand für diese Erweiterung der Suche immens werden. Leider gibt es keinen anderen Weg. Allerdings beschränken Sie die Untersuchung der Folgestellungen jenseits der regulären Suchtiefe konsequent auf die Schlagzüge und führen auch nicht mehr eine komplette Stellungsbewertung durch. Entscheidend ist in diesem Fall ohnehin nur der Materialwert, positionelle Vorteile vernachlässigen Sie aus Zeitgründen. Glücklicherweise sind durch die starke Wichtung der materiellen Bewertung gegenüber der positionellen keine weiteren Fehleinschätzungen zu erwarten.

Nun ist die Forderung nach einer Beschleunigung der Suche natürlich noch nachdrücklicher geworden. Sie erinnern sich, daß Sie den gewünschten Erfolg durch viele Cutoffs in der Alpha-Beta-Suche erlangen (»Der TacTic auf der Spur...«). Wieder leistet Ihnen die Kenntnis über Ihre eigene Stellungsbewertung wertvolle Dienste. Viele Punkte gibt es für Schlagzüge, die Sie schon im Zuggenerator durch einen Eintrag im Feld »Option« besonders markiert haben. Diese Zugvarianten haben also gute Chancen, ein paar Cutoffs zu erzwingen, wenn Sie diese zuerst untersuchen. Das Autorenteam des bereits erwähnten Computerschach-Buches führte eine sogenannte Hauptvariante in ihrem Programm ein. Dort speichern sie jeweils die bislang besten Zugfolgen. Meistens bleibt eine dieser »Hauptvarianten« auch bei geringfügig abweichenden Stellungen die beste Wahl.
Das Schachprogramm dieses Beitrags erreicht seine Cutoffs durch sorgfältige Buchführung über die bewerteten Züge einer Stellung.



Jeder bewertete Zug wird zusammen mit seiner Bewertung in eine zweite Zugliste »savelist« einsortiert. Die Sortierordnung ist je nach Spieler auf- oder absteigend, abhängig davon, ob für den Spieler hohe oder niedrige Punktzahlen eine gute Stellung bedeuten. Wenn Sie dann für dieselbe Suchtiefe das nächste Mal den Zuggenerator aufrufen, können Sie die neuen Züge eventuell durch einen Eintrag in der alten Bewertungsliste von vornherein abschätzen. Finden Sie den neuen Zug in der alten Liste, so dient die alte Bewertung als Kriterium für die Position in der neuen Zugliste. Vorher gut bewertete Stellungen gelangen damit automatisch an den Anfang der Liste, schlechte Züge ans Ende. Die Zugsuche beginnt die Bewertung also mit den Top-Kandidaten der letzten Runde – und liegt mit ihnen oftmals wieder richtig.

Bewertete Züge werden gespeichert

So einfach die Angelegenheit klingt, so viel Arbeit steckt in der technischen Umsetzung. Denn die vielen Ein- und Aussortierungen dürfen natürlich kaum Zeit in Anspruch nehmen, sonst verbrauchen Sie die mühsam durch Cutoffs gewonnene Zeit sofort wieder. Eine ausgeklügelte verkettete Listenstruktur ist gefragt. Zu diesem Zweck sind in der Definition der Zugstruktur die Einträge »Value« und »Next« enthalten. Hier speichern Sie die Bewertung des Zuges beim letzten Durchlauf und den Verweis auf den in der Sortierreihenfolge nächsten Zug.

● **Das Spielende:** Zu guter Letzt muß Ihr Programm natürlich auch erkennen, wann es verloren oder gewonnen hat. Dies erreichen Sie durch einen festen, in der regulären Bewertung unerreichbaren Wert. Die Zugsuche wählt diesen Zug aufgrund seiner guten Bewertung – und Sie erkennen an dem charakteristischen Wert, daß das Spiel zu Ende ist. Damit das Programm nicht sofort abbricht, sobald die Analyse ein paar Züge später das Spielende voraussieht, müssen Sie die Suchtiefe in diesen Wert eingehen lassen. Erst wenn bei Suchtiefe 0 das Spiel entschieden ist, beenden Sie es.

● **Das Programm:** Der Stand des Schachprogramms bis hier umfaßt etwa 3000 Zeilen Assembler, und nach außen sehen Sie nichts von der vielen Arbeit, die Sie geleistet haben. Also versehen Sie Ihr Schachprogramm mit einer ansprechenden Oberfläche. Die Routinen zur Realisierung schöner VGA-Spielfiguren und eines ansprechenden Spielfelds sind bereits in Ausgabe 19 der DOS Extra erschienen. Die Erklärungen zur Funktionsweise dieser Routinen lesen Sie bitte dort nach.

Im Programm ist eine Prozedur zum Einlesen von PCX-Bildern ebenfalls enthalten. Sie erleichtert das Erzeugen von Brett und Figuren, da Sie nun jedes beliebige Malprogramm verwenden können. Das Schachprogramm lädt zu Beginn des Spiels die Bilder »titel.pcx« (Bild 4) und »chess.pcx« ein. Ersteres enthält den Programmtitel quasi als Vorspann, die zweite Datei enthält Brett und Figuren. Malen Sie also ein PCX-Bild wie in Bild 5. Es zeigt Figuren und Spielfeld in einer ansprechenden Ausführung. Das Brett ist mit einer Oberflächenstruktur versehen, die das Erscheinungsbild des ganzen Programms aufwertet. Grundsätzlich können Sie das Spielfeld nach eigenen Vorstellungen gestalten. Sie müssen sich nur an die vom Programm erwartete Aufteilung halten. Das Brett beginnt auf dem 320 × 200 Punkte großen Bildschirm in Spalte 20. Jedes Feld ist 25 Punkte breit und 22 Punkte hoch. Die Spielfiguren beginnen in Spalte 230. Jede Figur ist maximal 24 Punkte

Bild 4. Einen Titel hat sich das Schachprogramm verdient: »titel.pcx«.

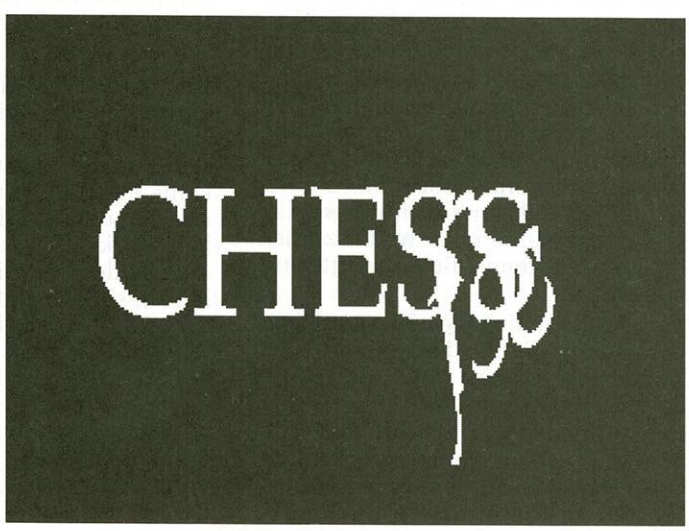


Bild 5. Mit einem Malprogramm zaubern Sie eine ansprechende Umgebung für Ihr Schachprogramm.

breit und 20 Punkte hoch. Die Reihenfolge Bauer, Turm, Läufer, Springer, Dame und König von oben nach unten müssen Sie streng einhalten. Der Hintergrund der Figuren im PCX-Bild muß in der Farbe Null gefüllt sein, sonst funktioniert die Animation der Spielfiguren auf dem Brett nicht korrekt. Die Piktogramme, die Sie im Anschluß an die Figuren für Schwarz und Weiß in Bild 4 sehen, versinnbildlichen den jeweiligen Programmstatus. Die verschieden ausschlagenden Waagen stehen für den Sieg der jeweiligen Farbe beziehungsweise das Remis. Die Farbe des Fragezeichens gibt an, welcher Spieler am Zug ist. Das Verbotszeichen signalisiert Ihnen, daß Ihre letzte Zugeingabe regelwidrig war. In jedem Fall müssen Sie die Existenz beider PCX-Bilder (für Brett und Titel) im Format 320 × 200 in 256 Farben sicherstellen, bevor Sie das Programm erstmals starten.

Während des Spielverlaufs teilen Sie Ihren Zug durch Eingabe von Start- und Zielfeld mit. Erreichen Sie mit einem Bauern die gegnerische Grundreihe, so geben Sie noch ein »D« oder ein »S« für die Bauernumwandlung ein. So wählen Sie zwischen einer Dame oder einem Springer (andere Figuren sind nicht sinnvoll). Mit [Esc] beenden Sie das Programm vorzeitig. Die Zeitdauer, die Sie dem Computer für seinen Zug zur Verfügung stellen, variieren Sie durch die Variable »Denkzeit«. Die Voreinstellung beträgt zehn Sekunden. Auf einem schnellen Rechner ist diese Zeit völlig ausreichend. Unter Umständen erhöhen Sie die Zeitspanne, bis Ihnen die Spielstärke des Programms zusagt.

Mit dem Wissen um die Programmierung eines Schachspiels sollten Sie sich nun gewappnet fühlen, selbst tief in die Tasten zu greifen, um Ihr Lieblingsstrategiespiel zu programmieren. Viel Erfolg!

(Frank Höppner/bm)

Bartel, Kraas, Schrüfer, Das große Computerschachbuch, Data Becker (zur Zeit nicht lieferbar)

```
[F33]    1: ;Programm: chessfx
[R94]    2: ;Funktion: Schachprogramm
[8G2]    3: ;Sprache:  TASM
[944]    4: ;Autor:    Frank Höppner
[7X4]    5: ; (c) 1992 DMV Widuch GmbH & Co. KG
[ØØØ]    6:
[6R4]    7: ;Das Programm nicht ohne die
[195]    8: ;beiden folgenden PCX-Bilder im
[FD5]    9: ;aktuellen Verzeichnis starten:
[PE5]   1Ø: ;chess.pcx : Spielbrett & Figuren
[RF5]   11: ;titel.pcx : beliebiges Titelbild
[ØØØ]   12:
[5A1]   13:             .RADIX 1Ø
[XY1]   14:             .MODEL SMALL
[FF1]   15:             .STACK 2ØØ
[2SØ]   16:             .DATA
[ØØØ]   17:
[PF3]   18:             ; ********************************
[SB5]   19:             ;     Schach-Konstanten / Variablen
[PF3]   2Ø:             ; ********************************
[ØØØ]   21:
[S13]   22: Denkzeit        EQU 1Ø
[MF2]   23: Rand            EQU Ø
[BW2]   24: KoenigS         EQU 1
[ZH2]   25: DameS           EQU 2
[TN2]   26: TurmS           EQU 3
[LW2]   27: LaeuferS        EQU 4
[Z13]   28: SpringerS       EQU 5
[UR2]   29: BauerS          EQU 6
[CF2]   3Ø: leer            EQU 7
[AS2]   31: BauerW          EQU 8
[F23]   32: SpringerW       EQU 9
[113]   33: LaeuferW        EQU 1Ø
[GU2]   34: TurmW           EQU 11
[VP2]   35: DameW           EQU 12
[M13]   36: KoenigW         EQU 13
[BL2]   37: weiss           EQU Ø
[423]   38: schwarz         EQU -1
[DH2]   39: false           EQU Ø
[NP2]   4Ø: true            EQU -1
[GM3]   41: betastart       EQU -32ØØØ
[RN3]   42: alphastart      EQU 32ØØØ
[ØØØ]   43:
[MØ3]   44: Schlagzug       EQU 3
[UY2]   45: klRochade       EQU 4
[6Z2]   46: grRochade       EQU 5
[DN3]   47: VideoSeg        EQU ØB8ØØh
[ØØØ]   48:
[ML2]   49: Zug             STRUC
[F52]   5Ø:   von             DB Ø
[H12]   51:   nach            DB Ø
[CG2]   52:   option          DW Ø
[V92]   53:   value           DW Ø
[W82]   54:   next            DW Ø
[6C2]   55: Zug             ENDS
[ØØØ]   56:
[545]   57:                 ; Wortgröße vorausgesetzt
[E93]   58: Spielstand      STRUC
[XM3]   59: feld              DB 12Ø DUP (Ø)
[ØØØ]   6Ø:
[1N4]   61: list              Zug 1ØØ DUP (<Ø,Ø,Ø>)
[ØM2]   62: anzahl            DW Ø
[RF2]   63: start             DW Ø
[ØØØ]   64:
[UU2]   65: EndSpielS         DB Ø
[2V2]   66: EndSpielW         DB Ø
[YV2]   67: MatWertS          DW Ø
[2W2]   68: MatWertW          DW Ø
[4R2]   69: MatDiff           DW Ø
[D93]   7Ø: StellungTyp       DB Ø
[ØL2]   71: KoenPS            DB Ø
[CL2]   72: KoenPW            DB Ø
[C13]   73: klRochWmgl        DB Ø
[WØ3]   74: klRochSmgl        DB Ø
[P13]   75: grRochWmgl        DB Ø
[613]   76: grRochSmgl        DB Ø
[N73]   77: RochadeFaktor     DB Ø
[PB3]   78: RochadeErfolgt    DB Ø
[WF2]   79: dummy             DB Ø
[ØØØ]   8Ø:
[PF2]   81: AmZug             DB Ø
[SG2]   82: ZugNr             DW Ø
[4L2]   83: cutoff            DB Ø
[7E2]   84: alpha             DW Ø
[VB2]   85: beta              DW Ø
[SV2]   86: ruhewert          DW Ø
[5A3]   87: bedrohungwert     DW Ø
[ØØØ]   88:
[735]   89: savelist          Zug 1ØØ DUP (<Ø,Ø,Ø>)
[623]   9Ø: saveanzahl        DW Ø
[XX2]   91: savestart         DW Ø
[Ø23]   92: Spielstand      ENDS
[ØØØ]   93:
[RA5]   94: GameStack       Spielstand 8 DUP (?)
[5M2]   95: aktStack        DW Ø
[ØØØ]   96:
[PK4]   97: DepthØ   EQU OffSet GameStack
[6S5]   98: Depth1   EQU DepthØ+SIZE Spielstand
[VX5]   99: Depth7   EQU DepthØ+7*SIZE Spielstand
[ØØØ]  1ØØ:
[HP2]  1Ø1: Brett    DB 2Ø DUP (Ø)
[VH2]  1Ø2:          DB Ø,3,5,4,2,1,4,5,3,Ø
[6L2]  1Ø3:          DB Ø,6,6,6,6,6,6,6,6,Ø
[PL2]  1Ø4:          DB Ø,7,7,7,7,7,7,7,7,Ø
[PL2]  1Ø5:          DB Ø,7,7,7,7,7,7,7,7,Ø
[PL2]  1Ø6:          DB Ø,7,7,7,7,7,7,7,7,Ø
[PL2]  1Ø7:          DB Ø,7,7,7,7,7,7,7,7,Ø
[6M2]  1Ø8:          DB Ø,8,8,8,8,8,8,8,8,Ø
[7A3]  1Ø9:          DB Ø,11,9,1Ø,12,13,1Ø,9,11,Ø
[7K1]  11Ø:          DB 2Ø DUP (Ø)
[ØØØ]  111:
[4U2]  112: Spalte   DB 2Ø DUP (Ø)
[3R3]  113:          DB 6 DUP (Ø,1,2,3,4,5,6,7,8,Ø)
[WU4]  114: Zeile    DB 2 DUP (Ø,1,2,3,4,5,6,7,8,Ø)
[7K1]  115:          DB 1Ø DUP (1)
[3Y1]  116:          DB Ø,8 DUP (2),Ø
[5Y1]  117:          DB Ø,8 DUP (3),Ø
[7Y1]  118:          DB Ø,8 DUP (4),Ø
[9Y1]  119:          DB Ø,8 DUP (5),Ø
[BY1]  12Ø:          DB Ø,8 DUP (6),Ø
[DY1]  121:          DB Ø,8 DUP (7),Ø
[FY1]  122:          DB Ø,8 DUP (8),Ø
[ØØØ]  123:
[DK6]  124: SprTW    DW 2Ø DUP (Ø) ; Springerwert weiß
[8V2]  125:          DW Ø,-42Ø,-3ØØ,-18Ø,-6Ø
[HV2]  126:          DW -6Ø,-18Ø,-3ØØ,-42Ø,Ø
[ØG2]  127:          DW Ø,-3ØØ,-18Ø,-6Ø,6Ø
[TF2]  128:          DW 6Ø,-6Ø,-18Ø,-3ØØ,Ø
[DE2]  129:          DW Ø,-18Ø,-6Ø,6Ø,18Ø
[VC2]  13Ø:          DW 18Ø,6Ø,-6Ø,-18Ø,Ø
[PD2]  131:          DW Ø,-6Ø,6Ø,18Ø,3ØØ
[9B2]  132:          DW 3ØØ,18Ø,6Ø,-6Ø,Ø
[PD2]  133:          DW Ø,-6Ø,6Ø,18Ø,3ØØ
[9B2]  134:          DW 3ØØ,18Ø,6Ø,-6Ø,Ø
[DE2]  135:          DW Ø,-18Ø,-6Ø,6Ø,18Ø
[VC2]  136:          DW 18Ø,6Ø,-6Ø,-18Ø,Ø
[ØG2]  137:          DW Ø,-3ØØ,-18Ø,-6Ø,6Ø
[TF2]  138:          DW 6Ø,-6Ø,-18Ø,-3ØØ,Ø
[8V2]  139:          DW Ø,-42Ø,-3ØØ,-18Ø,-6Ø
[HV2]  14Ø:          DW -6Ø,-18Ø,-3ØØ,-42Ø,Ø
[9S5]  141: ZAbst    DW 2Ø DUP (Ø) ; Zentrumabstand
[LE2]  142:          DW Ø,6ØØ,5ØØ,4ØØ,3ØØ
[FC2]  143:          DW 3ØØ,4ØØ,5ØØ,6ØØ,Ø
[7E2]  144:          DW Ø,5ØØ,4ØØ,3ØØ,2ØØ
[BC2]  145:          DW 2ØØ,3ØØ,4ØØ,5ØØ,Ø
[VD2]  146:          DW Ø,4ØØ,3ØØ,2ØØ,1ØØ
[7C2]  147:          DW 1ØØ,2ØØ,3ØØ,4ØØ,Ø
[Z52]  148:          DW Ø,3ØØ,2ØØ,1ØØ,Ø
[Z52]  149:          DW Ø,1ØØ,2ØØ,3ØØ,Ø
[Z52]  15Ø:          DW Ø,3ØØ,2ØØ,1ØØ,Ø
[Z52]  151:          DW Ø,1ØØ,2ØØ,3ØØ,Ø
[VD2]  152:          DW Ø,4ØØ,3ØØ,2ØØ,1ØØ
[7C2]  153:          DW 1ØØ,2ØØ,3ØØ,4ØØ,Ø
[7E2]  154:          DW Ø,5ØØ,4ØØ,3ØØ,2ØØ
[BC2]  155:          DW 2ØØ,3ØØ,4ØØ,5ØØ,Ø
[LE2]  156:          DW Ø,6ØØ,5ØØ,4ØØ,3ØØ
[FC2]  157:          DW 3ØØ,4ØØ,5ØØ,6ØØ,Ø
[FS5]  158: ZWert    DB 2Ø DUP (Ø) ; Bedrohungswert
[YF3]  159:          DB Ø,2Ø,2Ø,2Ø,2Ø,2Ø,2Ø,2Ø,2Ø,Ø
[YF3]  16Ø:          DB Ø,2Ø,2Ø,2Ø,2Ø,2Ø,2Ø,2Ø,2Ø,Ø
[6H3]  161:          DB Ø,2Ø,2Ø,6Ø,6Ø,6Ø,6Ø,2Ø,2Ø,Ø
[WR3]  162:          DB Ø,2Ø,2Ø,6Ø,1ØØ,1ØØ,6Ø,2Ø,2Ø,Ø
[WR3]  163:          DB Ø,2Ø,2Ø,6Ø,1ØØ,1ØØ,6Ø,2Ø,2Ø,Ø
[6H3]  164:          DB Ø,2Ø,2Ø,6Ø,6Ø,6Ø,6Ø,2Ø,2Ø,Ø
[YF3]  165:          DB Ø,2Ø,2Ø,2Ø,2Ø,2Ø,2Ø,2Ø,2Ø,Ø
[YF3]  166:          DB Ø,2Ø,2Ø,2Ø,2Ø,2Ø,2Ø,2Ø,2Ø,Ø
[Ø85]  167:          ; Vorrückwerte für Bauern W/S
[PU2]  168: VorBW    DW 1Ø DUP (Ø)
[EU2]  169: VorBS    DW 1Ø DUP (Ø)
[ME4]  17Ø:          ; Default Werte Vorrücken
[DK3]  171:          ; für Mittel/Endspiel
[WU4]  172: VorMS    DW Ø,Ø,Ø,39Ø,54Ø,7ØØ,23Ø,Ø,Ø,Ø
[1M3]  173: VorES    DW Ø,2ØØ,2ØØ,2ØØ,2ØØ
[5C2]  174:          DW 2ØØ,2ØØ,2ØØ,2ØØ,Ø
[ØØØ]  175:
[F85]  176:          ; alle Daten zwischen POSBW und
[ØB4]  177:          ; HORIZON werden bei neuer
[ML3]  178:          ; Stellung gelöscht
[ØØØ]  179:
[CD5]  18Ø:          ; Bauernposition, bei Null Anzahl
[FS2]  181: PosBW    DB 1Ø DUP (Ø)
[GP2]  182: SpaBW    DB 1Ø DUP (Ø)
[XP2]  183: ZeiBW    DB 1Ø DUP (Ø)
[7S2]  184: PosBS    DB 1Ø DUP (Ø)
[8P2]  185: SpaBS    DB 1Ø DUP (Ø)
[NP2]  186: ZeiBS    DB 1Ø DUP (Ø)
[2B5]  187:          ; Bauern-Zähler pro Spalte W/S
[ER2]  188: SZaBW    DB 1Ø DUP (Ø)
[6R2]  189: SZaBS    DB 1Ø DUP (Ø)
[TM5]  19Ø:          ; am wenigsten vorgerückte Bauern
[CK2]  191:          ; je Spalte W/S
[CW2]  192: SlowBW   DB 1Ø DUP (Ø)
[ØW2]  193: SlowBS   DB 1Ø DUP (Ø)
[KC4]  194:          ; Bauernliste Position S/W
[7W2]  195: ListBW   DB 6Ø DUP (Ø)
[VV2]  196: ListBS   DB 6Ø DUP (Ø)
[384]  197:          ; Freibauern in Spalte S/W
[EU2]  198: FreiBW   DB 1Ø DUP (Ø)
[2U2]  199: FreiBS   DB 1Ø DUP (Ø)
[9A4]  2ØØ:          ; rückständige Bauern S/W
[9V2]  2Ø1: RuckBW   DB 1Ø DUP (Ø)
[XU2]  2Ø2: RuckBS   DB 1Ø DUP (Ø)
[NL4]  2Ø3:          ; Position aller Figuren S/W
[N14]  2Ø4:          ; bei Null Figurenzähler
[1T2]  2Ø5: PosFW    DB 17 DUP (Ø)
[TS2]  2Ø6: PosFS    DB 17 DUP (Ø)
[MP5]  2Ø7: SumFW    DB Ø  ; Summe Figuren weiß
[BC5]  2Ø8: TurmSW   DB Ø  ; Turm-Spalte weiß
[3T5]  2Ø9: DamePW   DB Ø  ; Dame-Position weiß
[AF6]  21Ø: GrlFW    DB Ø  ; Sum.Leichtfiguren Grundr.
[FG6]  211: LaufW    DB Ø  ; Läufer weiß vorhanden
[H84]  212: SumFS    DB Ø  ; dito schwarz
[TV1]  213: TurmSS   DB Ø
[VR1]  214: DamePS   DB Ø
[YM1]  215: GrlFS    DB Ø
[TM1]  216: LaufS    DB Ø
[TC6]  217: schach   DB Ø  ; aktueller König bedroht?
[L22]  218: DrohWrt  DW Ø
[ØØØ]  219:
[422]  22Ø: horizon  DW Ø
[MY2]  221: OutZone  DW Depth7
[DØ2]  222: zugwahl  DW Ø
[712]  223: schutzL  DW Ø
[Z12]  224: schutzR  DW Ø
[GY1]  225: gameovr  DB Ø
[UR1]  226: sieger   DB Ø
[7Ø2]  227: timeout  DB Ø
[WL1]  228: abort    DB Ø
[PV4]  229: Wert     DW Ø,Ø,-9ØØ,-5ØØ,-325,-325,-1ØØ
[P53]  23Ø:          DW Ø,1ØØ,325,325,5ØØ,9ØØ,Ø
[9H4]  231: msgCZ    EQU 2  ; Computerzug
[9C4]  232: msgSZ    EQU 1  ; Spielerzug
[EK4]  233: msgCG    EQU 4  ; ComputerSieg
[SB4]  234: msgSG    EQU 5  ; SpielerSieg
[P83]  235: msgKG    EQU Ø  ; Remis
[D85]  236: msgEr    EQU 3  ; ungültiger Zug
[8C6]  237: txtCG    DB 'Tja, die Partie ging an mich'
[XY4]  238:          DB '. Vielen Dank fürs Spielen!$'
[DG6]  239: txtSG    DB 'Gratuliere, Sie haben gewonn'
[KL4]  24Ø:          DB 'en. Wann gibts Revanche?$'
[S36]  241: txtKG    DB 'Remis! Verschieben wir die E'
[UL5]  242:          DB 'ntscheidung auf die nächste'
[GZ1]  243:          DB ' Partie!$'
[ZA5]  244: txtAB    DB 'Chess-fx abgebrochen!$'
[ØØØ]  245:
[PF3]  246:          ; ********************************
[5M5]  247:          ; VGA-Grafik Konstanten/Variablen
[PF3]  248:          ; ********************************
[ØØØ]  249:
[M13]  25Ø: gCRTC            EQU 3D4h
[TA3]  251: gPixelX          EQU 32Ød
[UT4]  252: gPixelXHalf      EQU gPixelX/2
[NA3]  253: gPixelY          EQU 2ØØd
[2U4]  254: gPixelYHalf      EQU gPixelY/2
[664]  255: gScreenSize      EQU 2ØØd*8Ød
[2S3]  256: gScreenPageØ     EQU ØØØ0h
[G85]  257: gScreenPage1     EQU gScreenSize
[RA5]  258: gScreenPage2     EQU 2*gScreenSize
[UA5]  259: gScreenPage3     EQU 3*gScreenSize
[N84]  26Ø: gScreenSegment   EQU ØAØØØh
[ØØØ]  261:
[514]  262: gScreen DW  gScreenPage1
[R82]  263: gColor  DB  Ø2h
[8E4]  264: FallDown DB  2ØØ,19Ø,18Ø,17Ø,16Ø
[5U2]  265:          DB  15Ø,14Ø,13Ø,12Ø,11Ø
[AY2]  266:          DB  1ØØ,9Ø,8Ø,7Ø,6Ø,5Ø,4Ø
[5W2]  267:          DB  3Ø,2Ø,1Ø,Ø,5,1Ø,15,2Ø
[LW2]  268:          DB  25,3Ø,33,36,38,39,4Ø
[1W2]  269:          DB  4Ø,4Ø,39,38,36,33,3Ø
[ZS2]  27Ø:          DB  25,2Ø,15,1Ø,5,Ø,3,5
[SU2]  271:          DB  7,9,11,11,9,7,5,3,Ø
[K92]  272:          DB  1,2,3,4,3,2,1,Ø
[SR2]  273:          DB  1,2,3,2,1,Ø,1,2,1,Ø
[9V1]  274:          DB  1,Ø,1,Ø,255
[1R4]  275: Aufbau   DB  23Ø,2Ø,2Ø,Ø,23Ø,2Ø,195,Ø
[4V3]  276:          DB  254,2Ø,2Ø,154,254,2Ø,195,154
[N93]  277:          DB  23Ø,6Ø,45,Ø,23Ø,6Ø,17Ø,Ø
[VV3]  278:          DB  254,6Ø,45,154,254,6Ø,17Ø,154
[F93]  279:          DB  23Ø,4Ø,7Ø,Ø,23Ø,4Ø,145,Ø
[1V3]  28Ø:          DB  254,4Ø,7Ø,154,254,4Ø,145,154
[5D3]  281:          DB  23Ø,8Ø,95,Ø,23Ø,1ØØ,12Ø,Ø
[5X3]  282:          DB  254,8Ø,95,154,254,1ØØ,12Ø,154
[WX4]  283: FigurPos DW Ø,Ø,23Ø,3ØØ,23Ø,28Ø,23Ø
[5K3]  284:          DW 22Ø,23Ø,24Ø,23Ø,26Ø,23Ø,2ØØ
[D43]  285:          DW Ø,Ø,254,2ØØ,254,26Ø,254
[UL3]  286:          DW 24Ø,254,22Ø,254,28Ø,254,3ØØ
[ØØØ]  287:
[ZF2]  288: xcount           DW Ø
[1G2]  289: ycount           DW Ø
[S32]  29Ø: pack             DB Ø
[ZE2]  291: bufpos           DW Ø
[515]  292: gPCXfileTitel    DB 'TITEL.PCX',Ø
[UX4]  293: gPCXfileBrett    DB 'CHESS.PCX',Ø
[LB4]  294: gPCXBuffer       DB 1Ø24 DUP (Ø)
[ØØØ]  295:
[FTØ]  296:          .CODE
[ØØØ]  297:
[PF3]  298:          ; ********************************
[UH3]  299:          ;                          CHESS-fx
[PF3]  3ØØ:          ; ********************************
[ØØØ]  3Ø1:
[XR1]  3Ø2:          call Intro
[4A3]  3Ø3:          mov ax,seg GameStack
[1H1]  3Ø4:          mov ds,ax
[3H1]  3Ø5:          mov es,ax
[W24]  3Ø6:          mov ax,offset GameStack
[235]  3Ø7:          mov word ptr[ds:aktStack],ax
[ØØØ]  3Ø8:
[US2]  3Ø9:          mov si,aktStack
[NR1]  31Ø:          call Spiel
[MP1]  311:          call gExit
[XH1]  312:          call Quit
[ØØØ]  313:
[VM1]  314:          mov ah,4Ch
[381]  315:          int 21h
[ØØØ]  316:
[PF3]  317:          ; ********************************
[LC5]  318:          ;       Zuggenerator-Zug eintragen
```

»Chessfx.asm« verspricht ungetrübten Schachgenuß.

```
[PF3]  319:          ; ********************************
[ØØØ]  32Ø:
[YC2]  321: NeuerZug PROC
[8L3]  322:          ; Neuen Zug eintragen
[BL2]  323:          ; al=von, ah=nach
[V32]  324:          ; dx=variante
[8T4]  325:          ; (Schlagzüge werden autom.
[MU3]  326:          ; erkannt und markiert)
[ØØØ]  327:
[PC1]  328:          push ax
[TC1]  329:          push bx
[GC1]  33Ø:          push si
[US2]  331:          mov si,AktStack
[3F1]  332:          xor bh,bh
[ØØØ]  333:
[MY1]  334:          ; klRochade?
[9R2]  335:          cmp dl,klRochade
[X11]  336:          je nzØ2
[AY1]  337:          ; grRochade?
[HR2]  338:          cmp dl,grRochade
[X11]  339:          je nzØ2
[ØØØ]  34Ø:
[3H2]  341:          ; Zielfeld leer
[8F1]  342:          mov bl,ah
[C15]  343:          cmp BYTE PTR [ds:si+bx],leer
[X11]  344:          je nzØ2
[225]  345:          cmp BYTE PTR [ds:si+bx],rand
[T11]  346:          je nzØ1
[2L2]  347:          ; wer schlägt?
[GF1]  348:          mov bl,al
[C15]  349:          cmp BYTE PTR [ds:si+bx],leer
[P21]  35Ø:          jl nzØ3
[BX2]  351:          ; weiss schlägt!
[8F1]  352:          mov bl,ah
[C15]  353:          cmp BYTE PTR [ds:si+bx],leer
[FA5]  354:          ; sich selbst schlagen gilt nicht
[Z11]  355:          jg nzØ1
[SW2]  356:          mov dx,Schlagzug
[RB1]  357:          jmp nzØ2
[ØØØ]  358:
[H73]  359:          ; schwarz schlägt!
[TF2]  36Ø: nzØ3:    mov bl,ah
[C15]  361:          cmp BYTE PTR [ds:si+bx],leer
[FA5]  362:          ; sich selbst schlagen gilt nicht
[E21]  363:          jl nzØ1
[SW2]  364:          mov dx,Schlagzug
[ØØØ]  365:
[MM2]  366:          ; Zug eintragen
[ØØØ]  367:
[CD2]  368: nzØ2:    push cx
[6F1]  369:          xor ch,ch
[H15]  37Ø:          mov cl,byte ptr[ds:ZWert+bx]
[L25]  371:          add word ptr[ds:DrohWrt],cx
[V31]  372:          pop cx
[ØØØ]  373:
[L53]  374:          ; noch volle Suche?
[QF2]  375:          cmp si,horizon
[8F2]  376:          jle nzØ2ok  ; ja!
[RY3]  377:          ; nein, also Sonderzug?
[6V2]  378:          cmp dx,Schlagzug
[DA1]  379:          jne nzØ1
[ØØØ]  38Ø:
[FS5]  381:          ; wurde Zug schon einmal bewertet?
[TN2]  382: nzØ2ok: push ax
[ZC1]  383:          push dx
[GH1]  384:          mov dx,ax
[7V3]  385:          ; Bewertung in ax holen
[NG2]  386:          ; (aus savelist)
[813]  387:          call Aussortieren
[ØØØ]  388:
[SG5]  389:          ; Zug eintragen (von,nach,option)
[W25]  39Ø:          mov bx,word ptr[ds:si.anzahl]
[EN2]  391:          pop cx ; option
[3M6]  392:      mov word ptr[ds:si.list+bx.option],cx
[ZØ2]  393:          pop cx ; zug
[5K5]  394:          mov word ptr[ds:si.list+bx],cx
[ØØØ]  395:
[MN4]  396:          ; nach letzter Bewertung(ax)
[LD2]  397:          ; einsortieren
[Ø35]  398:          mov dx,word ptr[ds:si.anzahl]
[RF1]  399:          mov di,si
[7M2]  4ØØ:          add di,size feld
[K15]  4Ø1:          mov bx,word ptr[ds:si.start]
[ZZ2]  4Ø2:          call Einsortieren
[H15]  4Ø3:          mov word ptr[ds:si.start],bx
[4A6]  4Ø4:         add word ptr[ds:si.anzahl],size zug
[TH1]  4Ø5:          xor dx,dx
[ØØØ]  4Ø6:
[X32]  4Ø7: nzØ1:    pop si
[T31]  4Ø8:          pop bx
[R31]  4Ø9:          pop ax
[3PØ]  41Ø:          ret
[WB2]  411: NeuerZug ENDP
[ØØØ]  412:
[PF3]  413:          ; ********************************
[HB4]  414:          ;                Bauernzug -- weiss
[PF3]  415:          ; ********************************
[ØØØ]  416:
[7H3]  417: BauernZugWeiss PROC
[TC1]  418:          push bx
[TH1]  419:          xor dx,dx
[FF1]  42Ø:          mov al,bl
[ØØØ]  421:
[FX2]  422:          ; Schlagzug links
[SB1]  423:          sub bl,11
[3F1]  424:          mov ah,bl
[C15]  425:          cmp BYTE PTR [ds:si+bx],leer
[VM1]  426:          jge bzgwØ3
[ZY2]  427:          CALL bzwEintragen
[ØØØ]  428:
[V33]  429:          ; Schlagzug rechts
[P82]  43Ø: bzgwØ3: inc bl
[GØ1]  431:          inc bl
[3F1]  432:          mov ah,bl
[C15]  433:          cmp BYTE PTR [ds:si+bx],leer
[NM1]  434:          jge bzgwØ1
[ZY2]  435:          CALL bzwEintragen
[ØØØ]  436:
[254]  437: bzgwØ1: ; ein weiter leer?
[GF1]  438:          mov bl,al
[PB1]  439:          sub bl,1Ø
[3F1]  44Ø:          mov ah,bl
[C15]  441:          cmp BYTE PTR [ds:si+bx],leer
[DN1]  442:          jne bzgwØ2
[ZY2]  443:          CALL bzwEintragen
[ØØØ]  444:
[YF4]  445:          ; zwei vorwärts, wenn leer
[VB1]  446:          cmp al,8Ø
[LD1]  447:          jl bzgwØ2
[PB1]  448:          sub bl,1Ø
[3F1]  449:          mov ah,bl
[C15]  45Ø:          cmp BYTE PTR [ds:si+bx],leer
[DN1]  451:          jne bzgwØ2
[392]  452:          CALL NeuerZug
[ØØØ]  453:
[WB2]  454: bzgwØ2: pop bx
[3PØ]  455:          ret
[7H3]  456: BauernZugWeiss ENDP
[ØØØ]  457:
[PF3]  458:          ; ********************************
[W56]  459:          ;   Hilfroutine für BauernZugWeiss
[PF3]  46Ø:          ; ********************************
[ØØØ]  461:
[R23]  462: bzwEintragen PROC
[PD3]  463:          ; Bauernumwandlung?
[AB1]  464:          cmp ah,3Ø
```

```
[AD1]  465:          jg bzgwØ5
[XD2]  466:          ; Bauer in Dame
[R22]  467:          mov dx,DameW
[392]  468:          CALL NeuerZug
[W73]  469:          ; Bauer in Springer
[PW2]  47Ø:          mov dx,SpringerW
[392]  471:          CALL NeuerZug
[3PØ]  472:          ret
[UB5]  473: bzgwØ5: ; oder einfach vorwärts
[392]  474:          CALL NeuerZug
[3PØ]  475:          ret
[N13]  476: bzwEintragen ENDP
[ØØØ]  477:
[PF3]  478:          ; ********************************
[YH4]  479:          ;              Bauernzug -- schwarz
[PF3]  48Ø:          ; ********************************
[ØØØ]  481:
[D83]  482: BauernZugSchw PROC
[TC1]  483:          push bx
[TH1]  484:          xor dx,dx
[FF1]  485:          mov al,bl
[ØØØ]  486:
[FX2]  487:          ; Schlagzug links
[F61]  488:          add bl,9
[3F1]  489:          mov ah,bl
[C15]  49Ø:          cmp BYTE PTR [ds:si+bx],leer
[6N1]  491:          jle bzgsØ3
[VY2]  492:          CALL bzsEintragen
[ØØØ]  493:
[V33]  494:          ; Schlagzug rechts
[K82]  495: bzgsØ3: inc bl
[GØ1]  496:          inc bl
[3F1]  497:          mov ah,bl
[C15]  498:          cmp BYTE PTR [ds:si+bx],leer
[ØN1]  499:          jle bzgsØ1
[VY2]  5ØØ:          CALL bzsEintragen
[ØØØ]  5Ø1:
[Y44]  5Ø2: bzgsØ1: ; ein weiter leer?
[GF1]  5Ø3:          mov bl,al
[791]  5Ø4:          add bl,1Ø
[3F1]  5Ø5:          mov ah,bl
[C15]  5Ø6:          cmp BYTE PTR [ds:si+bx],leer
[9N1]  5Ø7:          jne bzgsØ2
[VY2]  5Ø8:          call bzsEintragen
[ØØØ]  5Ø9:
[YF4]  51Ø:          ; zwei vorwärts, wenn leer
[RB1]  511:          cmp al,4Ø
[MC1]  512:          jg bzgsØ2
[791]  513:          add bl,1Ø
[3F1]  514:          mov ah,bl
[C15]  515:          cmp BYTE PTR [ds:si+bx],leer
[9N1]  516:          jne bzgsØ2
[392]  517:          CALL NeuerZug
[ØØØ]  518:
[SB2]  519: bzgsØ2: pop bx
[3PØ]  52Ø:          ret
[E73]  521: BauernZugSchw ENDP
[ØØØ]  522:
[PF3]  523:          ; ********************************
[436]  524:          ;   Hilfsroutine für BauernZugSchw
[PF3]  525:          ; ********************************
[ØØØ]  526:
[723]  527: bzsEintragen PROC
[Ll4]  528:          ; dann Bauernumwandlung?
[GB1]  529:          cmp ah,9Ø
[9D1]  53Ø:          jl bzgsØ5
[XD2]  531:          ; Bauer in Dame
[L22]  532:          mov dx,DameS
[392]  533:          CALL NeuerZug
[W73]  534:          ; Bauer in Springer
[KW2]  535:          mov dx,SpringerS
[392]  536:          CALL NeuerZug
[3PØ]  537:          ret
[ØØØ]  538:
[P85]  539: bzgsØ5: ; oder einfach vorwärts
[392]  54Ø:          CALL NeuerZug
[3PØ]  541:          ret
[513]  542: bzsEintragen ENDP
[ØØØ]  543:
[PF3]  544:          ; ********************************
[FV3]  545:          ;                       Springerzug
[PF3]  546:          ; ********************************
[ØØØ]  547:
[2A3]  548: SpringerZug     PROC
[FF1]  549:          mov al,bl
[2F1]  55Ø:          mov ah,al
[TH1]  551:          xor dx,dx
[ØØØ]  552:
[Z52]  553:          add ah,21     ; 21
[392]  554:          CALL NeuerZug
[5K2]  555:          sub ah,21+21  ; -21
[392]  556:          CALL NeuerZug
[T52]  557:          add ah,21+8   ; 8
[392]  558:          CALL NeuerZug
[YB2]  559:          sub ah,8 +8   ; -8
[392]  56Ø:          CALL NeuerZug
[V82]  561:          add ah,8 +12  ; 12
[392]  562:          CALL NeuerZug
[4K2]  563:          sub ah,12+12  ; -12
[392]  564:          CALL NeuerZug
[MA2]  565:          add ah,12+19  ; 19
[392]  566:          CALL NeuerZug
[NL2]  567:          sub ah,19+19  ; -19
[392]  568:          CALL NeuerZug
[ØØØ]  569:
[3PØ]  57Ø:          ret
[1Y2]  571: SpringerZug ENDP
[ØØØ]  572:
[PF3]  573:          ; ********************************
[524]  574:          ;                    diagonale Züge
[PF3]  575:          ; ********************************
[ØØØ]  576:
[AW2]  577: DiagonalZug PROC
[TC1]  578:          push bx
[FF1]  579:          mov al,bl
[2F1]  58Ø:          mov ah,al
[TH1]  581:          xor dx,dx
[ØØØ]  582:
[FC3]  583:          ; nach rechts oben (-9)
[HD2]  584: dzro:    sub ah,9
[8F1]  585:          mov bl,ah
[392]  586:          CALL NeuerZug
[C15]  587:          cmp BYTE PTR [ds:si+bx],leer
[D81]  588:          je dzro
[2F1]  589:          mov ah,al
[ØØØ]  59Ø:
[VM3]  591:          ; nach rechts unten (+9)
[8B2]  592: dzru:    add ah,9
[8F1]  593:          mov bl,ah
[392]  594:          CALL NeuerZug
[C15]  595:          cmp BYTE PTR [ds:si+bx],leer
[591]  596:          je dzru
[2F1]  597:          mov ah,al
[ØØØ]  598:
[GA3]  599:          ; nach links oben (-11)
[LF2]  6ØØ: dzlo:    sub ah,11
[8F1]  6Ø1:          mov bl,ah
[392]  6Ø2:          CALL NeuerZug
[C15]  6Ø3:          cmp BYTE PTR [ds:si+bx],leer
[V71]  6Ø4:          je dzlo
[2F1]  6Ø5:          mov ah,al
[ØØØ]  6Ø6:
[EA3]  6Ø7:          ; nach links oben (+11)
[AD2]  6Ø8: dzlu:    add ah,11
[8F1]  6Ø9:          mov bl,ah
[392]  61Ø:          CALL NeuerZug
```

```
[C15]  611:          cmp BYTE PTR [ds:si+bx],leer
[L81]  612:          je dzlu
[ØØØ]  613:
[T11]  614:          pop bx
[3PØ]  615:          ret
[9U2]  616: DiagonalZug ENDP
[ØØØ]  617:
[PF3]  618:          ; ********************************
[XD5]  619:          ;       horizontale/vertikale Züge
[PF3]  62Ø:          ; ********************************
[ØØØ]  621:
[863]  622: HorizVertZug PROC
[TC1]  623:          push bx
[FF1]  624:          mov al,bl
[2F1]  625:          mov ah,al
[TH1]  626:          xor dx,dx
[ØØØ]  627:
[3B2]  628:          ; nach links (-1)
[R42]  629: hvzl:    dec ah
[8F1]  63Ø:          mov bl,ah
[392]  631:          CALL NeuerZug
[C15]  632:          cmp BYTE PTR [ds:si+bx],leer
[N81]  633:          je hvzl
[2F1]  634:          mov ah,al
[ØØØ]  635:
[MF2]  636:          ; nach rechts (+1)
[262]  637: hvzr:    inc ah
[8F1]  638:          mov bl,ah
[392]  639:          CALL NeuerZug
[C15]  64Ø:          cmp BYTE PTR [ds:si+bx],leer
[D91]  641:          je hvzr
[2F1]  642:          mov ah,al
[ØØØ]  643:
[4A2]  644:          ; nach oben (-1Ø)
[5H2]  645: hvzo:    sub ah,1Ø
[8F1]  646:          mov bl,ah
[392]  647:          CALL NeuerZug
[C15]  648:          cmp BYTE PTR [ds:si+bx],leer
[191]  649:          je hvzo
[2F1]  65Ø:          mov ah,al
[ØØØ]  651:
[CF2]  652:          ; nach unten (+1Ø)
[WE2]  653: hvzu:    add ah,1Ø
[8F1]  654:          mov bl,ah
[392]  655:          CALL NeuerZug
[C15]  656:          cmp BYTE PTR [ds:si+bx],leer
[T91]  657:          je hvzu
[ØØØ]  658:
[T31]  659:          pop bx
[3PØ]  66Ø:          ret
[653]  661: HorizVertZug ENDP
[ØØØ]  662:
[ØØØ]  663:
[PF3]  664:          ; ********************************
[NQ4]  665:          ;                          Königszug
[PF3]  666:          ; ********************************
[ØØØ]  667:
[YP2]  668: KoenigsZug PROC
[FF1]  669:          mov al,bl
[2F1]  67Ø:          mov ah,al
[TH1]  671:          xor dx,dx
[ØØØ]  672:
[GZ1]  673:          inc ah        ; 1
[392]  674:          CALL NeuerZug
[A92]  675:          sub ah,1+1    ; -1
[392]  676:          CALL NeuerZug
[792]  677:          add ah,1+11   ; 11
[392]  678:          CALL NeuerZug
[XH2]  679:          sub ah,11+11  ; -11
[392]  68Ø:          CALL NeuerZug
[B92]  681:          add ah,11+1Ø  ; 1Ø
[392]  682:          CALL NeuerZug
[PH2]  683:          sub ah,1Ø+1Ø  ; -1Ø
[392]  684:          CALL NeuerZug
[W52]  685:          add ah,1Ø+9   ; 9
[392]  686:          CALL NeuerZug
[2B2]  687:          sub ah,9+9    ; -9
[392]  688:          CALL NeuerZug
[ØØØ]  689:
[3PØ]  69Ø:          ret
[YP2]  691: KoenigsZug ENDP
[ØØØ]  692:
[PF3]  693:          ; ********************************
[M45]  694:          ;     Rochade nach Regeln möglich?
[HC5]  695:          ;     dx linkes Byte der zu unter-
[HG5]  696:          ;     suchenden Doppelposition des
[V94]  697:          ;                  Koenig (jetzt&test)
[ØM5]  698:          ;  (Update in KoenPs/w außerhalb!)
[PF3]  699:          ; ********************************
[ØØØ]  7ØØ:
[TG3]  7Ø1: RochadeMoeglich PROC
[PC1]  7Ø2:          push ax
[TC1]  7Ø3:          push bx
[WC1]  7Ø4:          push cx
[FH1]  7Ø5:          mov bx,dx
[WB4]  7Ø6:          mov ax,word ptr[ds:si+bx]
[NM1]  7Ø7:          xchg ah,al
[RE4]  7Ø8:          mov word ptr[ds:si+bx],ax
[ZC1]  7Ø9:          push dx
[NC3]  71Ø:          call KoenigImSchach
[T31]  711:          pop bx
[WB4]  712:          mov ax,word ptr[ds:si+bx]
[NM1]  713:          xchg ah,al
[RE4]  714:          mov word ptr[ds:si+bx],ax
[V31]  715:          pop cx
[T31]  716:          pop bx
[R31]  717:          pop ax
[3PØ]  718:          ret
[SE3]  719: RochadeMoeglich ENDP
[ØØØ]  72Ø:
[PF3]  721:          ; ********************************
[8L5]  722:          ;  Rochade nur erlaubt, wenn König
[6L4]  723:          ;            nicht im Schach steht
[PF3]  724:          ; ********************************
[ØØØ]  725:
[UA3]  726: RochadeErlaubt PROC
[PC1]  727:          push ax
[TC1]  728:          push bx
[WC1]  729:          push cx
[NC3]  73Ø:          call KoenigImSchach
[V31]  731:          pop cx
[T31]  732:          pop bx
[R31]  733:          pop ax
[3PØ]  734:          ret
[UA3]  735: RochadeErlaubt ENDP
[ØØØ]  736:
[PF3]  737:          ; ********************************
[TG4]  738:          ;                RochadenZug weiss
[PF3]  739:          ; ********************************
[ØØØ]  74Ø:
[8Ø3]  741: RochadenZugW PROC
[6C1]  742:          cmp al,95
[DV1]  743:          jne rzwend
[FF1]  744:          mov al,bl
[ØØØ]  745:
[DM2]  746:          ; kleine Rochade
[MX4]  747:          cmp BYTE PTR [ds:si+96],leer
[7G1]  748:          jne rzwg
[RX4]  749:          cmp BYTE PTR [ds:si+97],leer
[7G1]  75Ø:          jne rzwg
[RM6]  751:      cmp BYTE PTR [ds:si.klRochWmgl],true
[7G1]  752:          jne rzwg
[LE3]  753:          call RochadeErlaubt
[481]  754:          jc rzwg
[G45]  755:          mov byte ptr[ds:si.KoenPW],96
[2F1]  756:          mov dx,95
```

```
[BE3]  757:          call RochadeMoeglich
[E45]  758:          mov byte ptr[ds:si.KoenPW],95
[481]  759:          jc rzwg
[2T2]  76Ø:          mov dx,klRochade
[GD1]  761:          mov ah,97
[392]  762:          CALL NeuerZug
[ØØØ]  763:
[ØØ3]  764:          ; große Rochade
[H36]  765: rzwg:    cmp BYTE PTR [ds:si+92],leer
[DV1]  766:          jne rzwend
[BX4]  767:          cmp BYTE PTR [ds:si+93],leer
[DV1]  768:          jne rzwend
[EX4]  769:          cmp BYTE PTR [ds:si+94],leer
[DV1]  77Ø:          jne rzwend
[1N6]  771:        cmp BYTE PTR [ds:si.grRochWmgl],true
[DV1]  772:          jne rzwend
[LE3]  773:          call RochadeErlaubt
[KE1]  774:          jc rzwend
[C45]  775:          mov byte ptr[ds:si.KoenPW],94
[ØF1]  776:          mov dx,94
[BE3]  777:          call RochadeMoeglich
[E45]  778:          mov byte ptr[ds:si.KoenPW],95
[KE1]  779:          jc rzwend
[7U2]  78Ø:          mov dx,grRochade
[8D1]  781:          mov ah,93
[392]  782:          CALL NeuerZug
[ØØØ]  783:
[642]  784: rzwend: ret
[6Z2]  785: RochadenZugW ENDP
[ØØØ]  786:
[ØØØ]  787:
[PF3]  788:          ; ********************************
[SG4]  789:          ;                RochadenZug schwarz
[PF3]  79Ø:          ; ********************************
[ØØØ]  791:
[4Ø3]  792: RochadenZugS PROC
[1C1]  793:          cmp bl,25
[XU1]  794:          jne rzsend
[FF1]  795:          mov al,bl
[ØØØ]  796:
[DM2]  797:          ; kleine Rochade
[6X4]  798:          cmp BYTE PTR [ds:si+26],leer
[RF1]  799:          jne rzsg
[9X4]  8ØØ:          cmp BYTE PTR [ds:si+27],leer
[RF1]  8Ø1:          jne rzsg
[7M6]  8Ø2:        cmp BYTE PTR [ds:si.klRochSmgl],true
[RF1]  8Ø3:          jne rzsg
[LE3]  8Ø4:          call RochadeErlaubt
[S71]  8Ø5:          jc rzsg
[945]  8Ø6:          mov byte ptr[ds:si.KoenPW],26
[VE1]  8Ø7:          mov dx,25
[BE3]  8Ø8:          call RochadeMoeglich
[745]  8Ø9:          mov byte ptr[ds:si.KoenPW],25
[871]  81Ø:          jc rzsg
[2T2]  811:          mov dx,klRochade
[9D1]  812:          mov ah,27
[392]  813:          CALL NeuerZug
[ØØØ]  814:
[ØØ3]  815:          ; große Rochade
[L26]  816: rzsg:    cmp BYTE PTR [ds:si+22],leer
[XU1]  817:          jne rzsend
[XW4]  818:          cmp BYTE PTR [ds:si+23],leer
[XU1]  819:          jne rzsend
[ØX4]  82Ø:          cmp BYTE PTR [ds:si+24],leer
[XU1]  821:          jne rzsend
[HM6]  822:        cmp BYTE PTR [ds:si.grRochSmgl],true
[XU1]  823:          jne rzsend
[LE3]  824:          call RochadeErlaubt
[6E1]  825:          jc rzsend
[545]  826:          mov byte ptr[ds:si.KoenPW],24
[TE1]  827:          mov dx,24
[BE3]  828:          call RochadeMoeglich
[745]  829:          mov byte ptr[ds:si.KoenPW],25
[6E1]  83Ø:          jc rzsend
[7U2]  831:          mov dx,grRochade
[1D1]  832:          mov ah,23
[392]  833:          CALL NeuerZug
[ØØØ]  834:
[P32]  835: rzsend: ret
[2Z2]  836: RochadenZugS ENDP
[ØØØ]  837:
[ØØØ]  838:
[PF3]  839:          ; ********************************
[E35]  84Ø:          ;          gesamte Zugliste löschen
[PF3]  841:          ; ********************************
[ØØØ]  842:
[RØ3]  843: LoescheListe PROC
[ULØ]  844:          cld
[WR2]  845:          mov di,aktStack
[1Z4]  846:          mov word ptr[ds:di.Anzahl],Ø
[LV4]  847:          mov word ptr[ds:di.Start],Ø
[FB6]  848:        mov word ptr[ds:di.BedrohungWert],Ø
[7M2]  849:          add di,SIZE feld
[GH1]  85Ø:          xor ax,ax
[RV2]  851:          mov cx,SIZE List
[ØA1]  852:          sar cx,1
[KN1]  853:          rep stosw
[3PØ]  854:          ret
[NZ2]  855: LoescheListe ENDP
[ØØØ]  856:
[PF3]  857:          ; ********************************
[8D5]  858:          ;        sortierte Zugliste löschen
[PF3]  859:          ; ********************************
[ØØØ]  86Ø:
[5N3]  861: LoescheSaveList PROC
[WR2]  862:          mov di,aktStack
[YK5]  863:          mov word ptr[ds:di.savestart],Ø
[CP5]  864:          mov word ptr[ds:di.saveanzahl],Ø
[ET6]  865:    add di,(size spielstand-size savelist-4)
[GH1]  866:          xor ax,ax
[XL3]  867:          mov cx,size savelist
[ØA1]  868:          sar cx,1
[KN1]  869:          rep stosw
[3PØ]  87Ø:          ret
[4L3]  871: LoescheSaveList ENDP
[ØØØ]  872:
[PF3]  873:          ; ********************************
[5H5]  874:          ;          mögliche Züge für weiss
[PF3]  875:          ; ********************************
[ØØØ]  876:
[R13]  877: ZugGenWeiss PROC
[GC1]  878:          push si
[VY2]  879:          CALL LoescheListe
[US2]  88Ø:          mov si,aktStack
[DE1]  881:          mov bx,2Ø
[6F1]  882:          mov cx,79
[ØØØ]  883:
[CN2]  884:          ; weisse Figur?
[T72]  885: zgwØ1:   inc bx
[WC1]  886:          push cx
[ZL4]  887:          mov cl,BYTE PTR [ds:si+bx]
[MY1]  888:          cmp cl,leer
[5H1]  889:          jle zgwØ2
[ØØØ]  89Ø:
[VT4]  891:          mov word ptr[ds:DrohWrt],Ø
[ØØØ]  892:
[Y52]  893:          cmp cl,BauerW
[1M1]  894:          jnz zgwØ3
[DH3]  895:          CALL BauernZugWeiss
[7Ø4]  896: zgwØ3:  cmp cl,SpringerW
[3M1]  897:          jnz zgwØ4
[UX2]  898:          CALL SpringerZug
[MK1]  899:          jmp zgwØ2
[H93]  9ØØ: zgwØ4:  cmp cl,TurmW
[5M1]  9Ø1:          jnz zgwØ5
[L43]  9Ø2:          CALL HorizVertZug
[235]  9Ø3:          mov dx,word ptr[ds:DrohWrt]
[A91]  9Ø4:          sal dx,1
[MD6]  9Ø5:        add word ptr[ds:si.BedrohungWert],dx
[MK1]  9Ø6:          jmp zgwØ2
[7Y3]  9Ø7: zgwØ5:   cmp cl,LaeuferW
[7M1]  9Ø8:          jnz zgwØ6
[BV2]  9Ø9:          CALL DiagonalZug
[235]  91Ø:          mov dx,word ptr[ds:DrohWrt]
[A91]  911:          sal dx,1
[MD6]  912:        add word ptr[ds:si.BedrohungWert],dx
[MK1]  913:          jmp zgwØ2
[E63]  914: zgwØ6:   cmp cl,DameW
[9M1]  915:          jnz zgwØ7
[L43]  916:          CALL HorizVertZug
[BV2]  917:          CALL DiagonalZug
[235]  918:          mov dx,word ptr[ds:DrohWrt]
[MD6]  919:        add word ptr[ds:si.BedrohungWert],dx
[MK1]  92Ø:          jmp zgwØ2
[4M3]  921: zgwØ7:   cmp cl,KoenigW
[ZL1]  922:          jnz zgwØ2
[565]  923:          mov byte ptr[ds:si.KoenPW],bl
[WS2]  924:          CALL KoenigsZug
[NØ3]  925:          CALL RochadenZugW
[ØØØ]  926:
[2A2]  927: zgwØ2:  pop cx
[5P1]  928:          loop zgwØ1
[ØØØ]  929:
[RK3]  93Ø:          call LoescheSaveList
[E31]  931:          pop si
[3PØ]  932:          ret
[PZ2]  933: ZugGenWeiss ENDP
[ØØØ]  934:
[PF3]  935:          ; ********************************
[8G5]  936:          ;          mögliche Züge für schwarz
[PF3]  937:          ; ********************************
[ØØØ]  938:
[DE3]  939: ZugGenSchwarz   PROC
[GC1]  94Ø:          push si
[VY2]  941:          CALL LoescheListe
[US2]  942:          mov si,aktStack
[DE1]  943:          mov bx,2Ø
[6F1]  944:          mov cx,79
[ØØØ]  945:
[CN2]  946:          ; weisse Figur?
[972]  947: zgsØ1:   inc bx
[WC1]  948:          push cx
[ZL4]  949:          mov cl,BYTE PTR [ds:si+bx]
[MY1]  95Ø:          cmp cl,leer
[6G1]  951:          jge zgsØ2
[ØØØ]  952:
[VT4]  953:          mov word ptr[ds:DrohWrt],Ø
[ØØØ]  954:
[P52]  955:          cmp cl,BauerS
[HL1]  956:          jnz zgsØ3
[T63]  957:          CALL BauernZugSchw
[LZ3]  958: zgsØ3:   cmp cl,SpringerS
[LL1]  959:          jnz zgsØ4
[UX2]  96Ø:          CALL SpringerZug
[4K1]  961:          jmp zgsØ2
[X83]  962: zgsØ4:   cmp cl,TurmS
[NL1]  963:          jnz zgsØ5
[L43]  964:          CALL HorizVertZug
[235]  965:          mov dx,word ptr[ds:DrohWrt]
[A91]  966:          sal dx,1
[3G6]  967:        sub word ptr[ds:si.BedrohungWert],dx
[4K1]  968:          jmp zgsØ2
[7X3]  969: zgsØ5:   cmp cl,LaeuferS
[RL1]  97Ø:          jnz zgsØ6
[BV2]  971:          CALL DiagonalZug
[235]  972:          mov dx,word ptr[ds:DrohWrt]
[A91]  973:          sal dx,1
[3G6]  974:        sub word ptr[ds:si.BedrohungWert],dx
[4K1]  975:          jmp zgsØ2
[U53]  976: zgsØ6:   cmp cl,DameS
[TL1]  977:          jnz zgsØ7
[L43]  978:          CALL HorizVertZug
[BV2]  979:          CALL DiagonalZug
[235]  98Ø:          mov dx,word ptr[ds:DrohWrt]
[3G6]  981:        sub word ptr[ds:si.BedrohungWert],dx
[4K1]  982:          jmp zgsØ2
[8L3]  983: zgsØ7:   cmp cl,KoenigS
[FL1]  984:          jnz zgsØ2
[X55]  985:          mov byte ptr[ds:si.KoenPS],bl
[WS2]  986:          CALL KoenigsZug
[D93]  987:          CALL RochadenZugS
[ØØØ]  988:
[K92]  989: zgsØ2:  pop cx
[1P1]  99Ø:          loop zgsØ1
[ØØØ]  991:
[RK3]  992:          call LoescheSaveList
[E31]  993:          pop si
[3PØ]  994:          ret
[K73]  995: ZugGenSchwarz ENDP
[ØØØ]  996:
[ØØØ]  997:
[PF3]  998:          ; ********************************
[XK4]  999:          ;                Fuehre Zug Nr BX aus
[PF3]  1ØØØ:         ; ********************************
[ØØØ]  1Ø01:
[W23]  1Ø02: FuehreZugAus PROC
[TC1]  1Ø03:         push bx
[BR5]  1Ø04:         mov ax,word ptr [ds:si.list+bx]
[6U6]  1Ø05:    mov dx,word ptr [ds:si.list+bx.option]
[ØØØ]  1Ø06:
[LH1]  1Ø07:         xor bx,bx
[T23]  1Ø08:         mov bl,ah  ; bx=nach
[LL4]  1Ø09:         mov ch,byte ptr [ds:si+bx]
[ØØØ]  1Ø1Ø:
[H44]  1Ø11:         ; Rochadenrechte klären
[GP3]  1Ø12:         ; (Turm/König bewegt?)
[AF5]  1Ø13:         cmp al,25  ; Pos. schwarzer König
[LL1]  1Ø14:         jne fzaØØ1
[EH6]  1Ø15:       mov byte ptr[ds:si.klRochSmgl],false
[PH6]  1Ø16:       mov byte ptr[ds:si.grRochSmgl],false
[DS2]  1Ø17:         cmp dx,klRochade
[YK1]  1Ø18:         jge fzaØØØ
[GE1]  1Ø19:         jmp fzaw
[651]  1Ø2Ø: fzaØØØ:
[ØC6]  1Ø21:       sub byte ptr[ds:si.RochadeFaktor],1ØØ
[8K6]  1Ø22:       mov byte ptr[ds:si.RochadeErfolgt],dl
[3D5]  1Ø23:         mov byte ptr[ds:si+bx],KoenigS
[UV4]  1Ø24:         mov byte ptr[ds:si+25],leer
[X55]  1Ø25:         mov byte ptr[ds:si.KoenPS],bl
[K91]  1Ø26:         add bl,25
[F91]  1Ø27:         shr bl,1
[Y25]  1Ø28:         mov byte ptr[ds:si+bx],TurmS
[DS2]  1Ø29:         cmp dx,klRochade
[VE1]  1Ø3Ø:         je fzaSKR
[KV4]  1Ø31:         mov byte ptr[ds:si+21],leer
[4H1]  1Ø32:         jmp fza99
[586]  1Ø33: fzaSKR: mov byte ptr[ds:si+28],leer
[4H1]  1Ø34:         jmp fza99
[4A6]  1Ø35: fzaØØ1: cmp al,21  ; Pos.links schw Turm
[PL1]  1Ø36:         jne fzaØØ2
[PH6]  1Ø37:       mov byte ptr[ds:si.grRochSmgl],false
[GE1]  1Ø38:         jmp fzaw
[EH6]  1Ø39: fzaØØ2: cmp al,28  ; Pos.rechts schw Turm
[TL1]  1Ø4Ø:         jne fzaØØ3
[EH6]  1Ø41:       mov byte ptr[ds:si.klRochSmgl],false
[GE1]  1Ø42:         jmp fzaw
[1L6]  1Ø43: fzaØØ3: cmp al,95  ; Pos. weißer König
[WL1]  1Ø44:         jne fzaØØ4
[UH6]  1Ø45:       mov byte ptr[ds:si.klRochWmgl],false
[2K6]  1Ø46:       mov byte ptr[ds:si.grRochWmgl],false
[DS2]  1Ø47:         cmp dx,klRochade
[H81]  1Ø48:         jl fzaw
[H96]  1Ø49:       add byte ptr[ds:si.RochadeFaktor],1ØØ
[8K6]  1Ø5Ø:       mov byte ptr[ds:si.RochadeErfolgt],dl
[FD5]  1Ø51:         mov byte ptr[ds:si+bx],KoenigW
[1W4]  1Ø52:         mov byte ptr[ds:si+95],leer
[565]  1Ø53:         mov byte ptr[ds:si.KoenPW],bl
[T91]  1Ø54:         add bl,95
[F91]  1Ø55:         shr bl,1
[235]  1Ø56:         mov byte ptr[ds:si+bx],TurmW
[DS2]  1Ø57:         cmp dx,klRochade
[BF1]  1Ø58:         je fzaWKR
[TV4]  1Ø59:         mov byte ptr[ds:si+91],leer
[4H1]  1Ø6Ø:         jmp fza99
[G86]  1Ø61: fzaWKR: mov byte ptr[ds:si+98],leer
[4H1]  1Ø62:         jmp fza99
[617]  1Ø63: fzaØØ4: cmp al,91  ;Pos.links weißer Turm
[ZL1]  1Ø64:         jne fzaØØ5
[2K6]  1Ø65:       mov byte ptr[ds:si.grRochWmgl],false
[GE1]  1Ø66:         jmp fzaw
[497]  1Ø67: fzaØØ5: cmp al,98  ;Pos.rechts weißer Turm
[7D1]  1Ø68:         jne fzaw
[UH6]  1Ø69:       mov byte ptr[ds:si.klRochWmgl],false
[ØØØ]  1Ø7Ø:
[K12]  1Ø71:         ; Schlagzug?
[H13]  1Ø72: fzaw:   cmp ch,leer
[T41]  1Ø73:         je fzaØ9
[651]  1Ø74:         jl fzaØ1
[ØØØ]  1Ø75:
[253]  1Ø76:         ; weiß geschlagen
[TC1]  1Ø77:         push bx
[3F1]  1Ø78:         xor bh,bh
[AF1]  1Ø79:         mov bl,ch
[691]  1ØØØ:         sal bx,1
[M25]  1Ø81:         mov dx,word ptr[ds:Wert+bx]
[T31]  1Ø82:         pop bx
[GU5]  1Ø83:         sub word ptr[ds:si.MatWertW],dx
[TD5]  1Ø84:         sub word ptr[ds:si.MatDiff],dx
[VG1]  1Ø85:         jmp fzaØ9
[ØØØ]  1Ø86:
[TA3]  1Ø87:         ; schwarz geschlagen
[RG2]  1Ø88: fzaØ1:  push bx
[3F1]  1Ø89:         xor bh,bh
[AF1]  1Ø9Ø:         mov bl,ch
[691]  1Ø91:         sal bx,1
[M25]  1Ø92:         mov dx,word ptr[ds:Wert+bx]
[T31]  1Ø93:         pop bx
[GU5]  1Ø94:         sub word ptr[ds:si.MatWertS],dx
[TD5]  1Ø95:         sub word ptr[ds:si.MatDiff],dx
[ØØØ]  1Ø96:
[D84]  1Ø97: fzaØ9:  mov bl,al  ; bx=von
[ZL4]  1Ø98:         mov cl,byte ptr [ds:si+bx]
[ØØØ]  1Ø99:
[ØR3]  11ØØ:         ; neue Königsposition
[LD2]  1101:         cmp cl,KoenigW
[NN1]  1102:         jne fzaØ9a
[W55]  1103:         mov byte ptr[ds:si.KoenPW],ah
[1R1]  1104:         jmp fzaØ9b
[KN3]  1105: fzaØ9a: cmp cl,KoenigS
[SN1]  1106:         jne fzaØ9b
[M55]  1107:         mov byte ptr[ds:si.KoenPS],ah
[ØØØ]  1108:
[PD3]  1109:         ; Bauernumwandlung?
[CE3]  111Ø: fzaØ9b: cmp cl,BauerW
[1F1]  1111:         jne fzalØ
[AC1]  1112:         cmp bl,39
[V41]  1113:         jg fzal1
[NF1]  1114:         mov cl,dl
[TC1]  1115:         push bx
[FH1]  1116:         mov bx,dx
[691]  1117:         sal bx,1
[M25]  1118:         mov dx,word ptr[ds:Wert+bx]
[ZS5]  1119:         sub dx,word ptr[ds:Wert+2*BauerW]
[T31]  112Ø:         pop bx
[1S5]  1121:         add word ptr[ds:si.MatWertW],dx
[AB5]  1122:         add word ptr[ds:si.MatDiff],dx
[CG1]  1123:         jmp fzal1
[ØØØ]  1124:
[L93]  1125: fzalØ:  cmp cl,BauerS
[3F1]  1126:         jne fzal1
[XB1]  1127:         cmp bl,8Ø
[A51]  1128:         jl fzal1
[NF1]  1129:         mov cl,dl
[TC1]  113Ø:         push bx
[FH1]  1131:         mov bx,dx
[691]  1132:         sal bx,1
[M25]  1133:         mov dx,word ptr[ds:Wert+bx]
[VS5]  1134:         sub dx,word ptr[ds:Wert+2*BauerS]
[T31]  1135:         pop bx
[HR5]  1136:         add word ptr[ds:si.MatWertS],dx
[AB5]  1137:         add word ptr[ds:si.MatDiff],dx
[ØØØ]  1138:
[7L2]  1139: fzal1:  mov bl,ah
[VR4]  114Ø:         mov byte ptr [ds:si+bx],cl
[GF1]  1141:         mov bl,al
[Y25]  1142:         mov byte ptr [ds:si+bx],leer
[ØØØ]  1143:
[G82]  1144: fza99:  pop bx
[3PØ]  1145:         ret
[U13]  1146: FuehreZugAus ENDP
[ØØØ]  1147:
[PF3]  1148:         ; ********************************
[463]  1149:         ; Stellungsanalyse
[PF3]  115Ø:         ; ********************************
[ØØØ]  1151:
[G52]  1152: Analyse PROC
[WE3]  1153:         ; alte Daten löschen
[GH1]  1154:         xor ax,ax
[4P3]  1155:         mov cx,offset Horizon
[8D3]  1156:         sub cx,offset PosBW
[SE3]  1157:         mov bx,offset PosBW
[ØA1]  1158:         sar cx,1
[4G1]  1159:         mov di,bx
[KN1]  116Ø:         rep stosw
[ØØØ]  1161:
[672]  1162:         ; Endspiel S/W
[AS5]  1163:         cmp word ptr[ds:si.MatWertW],2ØØØ
[UØ1]  1164:         jg anØ1
[Ø66]  1165:         mov byte ptr[ds:si.EndspielW],true
[MU6]  1166: anØ1:   cmp word ptr[ds:si.MatWertS],-2ØØØ
[D11]  1167:         jl anØ2
[W56]  1168:         mov byte ptr[ds:si.EndspielS],true
[ØØØ]  1169:
[XX2]  117Ø:         ; Bauernstandorte
[CR3]  1171: anØ2:   mov si,aktStack
[DE1]  1172:         mov bx,2Ø
[6F1]  1173:         mov cx,79
[ØØØ]  1174:
[7Ø2]  1175: anØ3:   inc bx
[YW4]  1176:         cmp byte ptr[ds:si+bx],leer
[441]  1177:         jl anØ3s
[L31]  1178:         je anØ3w
[845]  1179:         cmp byte ptr[ds:si+bx],BauerW
[L31]  118Ø:         je anØ3w
[FE4]  1181:         ; weiße Figur eintragen
[9F1]  1182:         xor dh,dh
[RF4]  1183:         mov dl,byte ptr[ds:PosFW]
[MØ1]  1184:         inc dl
[6G1]  1185:         mov di,dx
[DL4]  1186:         mov byte ptr[ds:PosFW],dl
[P15]  1187:         mov byte ptr[ds:PosFW+di],bl
[7B4]  1188:         mov dl,byte ptr[ds:si+bx]
[8F1]  1189:         jmp anØ3w
[ØØØ]  119Ø:
```

»Chessfx.asm« verspricht ungetrübten Schachgenuß.

```
[4C4] 1191:                ; schwarze Figur eintragen
[KY5] 1192: anØ3s:  cmp byte ptr[ds:si+bx],rand
[L31] 1193:         je anØ3w
[Ø45] 1194:         cmp byte ptr[ds:si+bx],BauerS
[ØJ1] 1195:         je anØ4
[9F1] 1196:         xor dh,dh
[LF4] 1197:         mov dl,byte ptr[ds:PosFS]
[MØ1] 1198:         inc dl
[6G1] 1199:         mov di,dx
[5L4] 12ØØ:         mov byte ptr[ds:PosFS],dl
[E15] 12Ø1:         mov byte ptr[ds:PosFS+di],bl
[7B4] 12Ø2:         mov dl,byte ptr[ds:si+bx]
[8F1] 12Ø3:         jmp anØ3w
[ØØØ] 12Ø4:
[U54] 12Ø5:         ; Konstr. wegen near jump
[HG2] 12Ø6: anØ3a:  loop anØ3
[W91] 12Ø7:         jmp anØ5
[ØØØ] 12Ø8:
[376] 12Ø9: anØ3w:  cmp byte ptr[ds:si+bx],BauerW
[K81] 121Ø:         jne anØ4
[S52] 1211:         ; neuer Bauer
[644] 1212:         inc byte ptr[ds:PosBW]
[8W4] 1213:         ; di=wievielter weißer Bauer
[9F1] 1214:         xor dh,dh
[7F4] 1215:         mov dl,byte ptr[ds:PosBW]
[6G1] 1216:         mov di,dx
[515] 1217:         ; Bauerspalte/zeile ermitteln
[WA3] 1218:         ; ah=Zeile, al=Spalte
[WY4] 1219:         mov ah,byte ptr[ds:bx+Zeile]
[555] 122Ø:         mov al,byte ptr[ds:bx+Spalte]
[XU2] 1221:         ; Bauernposition
[424] 1222:         mov byte ptr[ds:di+PosBW],bl
[FX4] 1223:         mov byte ptr[ds:di+SpaBW],al
[MX4] 1224:         mov byte ptr[ds:di+ZeiBW],ah
[224] 1225:         ; Bauern-Spaltenzähler
[Y63] 1226:         ; Position/Slowest
[ØF1] 1227:         xor ah,ah
[8F3] 1228:         mov di,ax  ; di=spalte
[TY4] 1229:         mov dh,byte ptr[ds:di+SZaBW]
[SC4] 123Ø:         inc byte ptr[ds:di+SZaBW]
[R15] 1231:         ; jeweils neuer Bauer  ist auch
[CW4] 1232:         ; langsamster weißer Bauer
[H65] 1233:         mov byte ptr[ds:di+SlowBW],bl
[F81] 1234:         shl dh,1
[FF1] 1235:         mov dl,dh
[F81] 1236:         shl dh,1
[F81] 1237:         shl dh,1
[EB1] 1238:         add dl,dh
[9F1] 1239:         xor dh,dh
[MC1] 124Ø:         add dx,di
[6G1] 1241:         mov di,dx
[465] 1242:         mov byte ptr[ds:di+ListBW],bl
[WD1] 1243:         jmp anØ3a
[ØØØ] 1244:
[M26] 1245: anØ4:   cmp byte ptr[ds:si+bx],BauerS
[LC1] 1246:         jne anØ3a
[S52] 1247:         ; neuer Bauer
[Y34] 1248:         inc byte ptr[ds:PosBS]
[PF5] 1249:         ; di=wievielter schwarzer Bauer
[9F1] 125Ø:         xor dh,dh
[3F4] 1251:         mov dl,byte ptr[ds:PosBS]
[6G1] 1252:         mov di,dx
[515] 1253:         ; Bauerspalte/zeile ermitteln
[WA3] 1254:         ; ah=Zeile, al=Spalte
[WY4] 1255:         mov ah,byte ptr[ds:bx+Zeile]
[555] 1256:         mov al,byte ptr[ds:bx+Spalte]
[XU2] 1257:         ; Bauernposition
[ØZ4] 1258:         mov byte ptr[ds:di+PosBS],bl
[BX4] 1259:         mov byte ptr[ds:di+SpaBS],al
[GX4] 126Ø:         mov byte ptr[ds:di+ZeiBS],ah
[MS5] 1261:         ; Bauern-Spaltenzähler/Position
[ØF1] 1262:         xor ah,ah
[AC3] 1263:         mov di,ax ; di=spalte
[9Y4] 1264:         mov dh,byte ptr[ds:di+SZaBS]
[MC4] 1265:         inc byte ptr[ds:di+SZaBS]
[D81] 1266:         cmp dh,Ø
[9F1] 1267:         jnz anØ3b
[ZV3] 1268:         ; erster Bauer in Spalte
[B45] 1269:         ; (bei schwarz gleichzeitig der
[UR4] 127Ø:         ; langsamste Bauer in Spalte)
[965] 1271:         mov byte ptr[ds:di+SlowBS],bl
[ØA2] 1272: anØ3b:  shl dh,1
[FF1] 1273:         mov dl,dh
[F81] 1274:         shl dh,1
[F81] 1275:         shl dh,1
[EB1] 1276:         add dl,dh
[9F1] 1277:         xor dh,dh
[MC1] 1278:         add dx,di
[6G1] 1279:         mov di,dx
[W55] 128Ø:         mov byte ptr[ds:di+ListBS],bl
[WD1] 1281:         jmp anØ3a
[ØØØ] 1282:
[T64] 1283:         ; Stellungstyp ermitteln
[337] 1284: anØ5:   cmp byte ptr[ds:si.EndSpielW],true
[811] 1285:         je anØ6
[A46] 1286:         cmp byte ptr[ds:si.EndSpielS],true
[811] 1287:         je anØ6
[S91] 1288:         ; normal
[2Ø7] 1289: anØ8:   mov byte ptr[ds:si.StellungTyp],4
[Y91] 129Ø:         jmp anØ7
[ØØØ] 1291:
[PA3] 1292:         ; noch weisse Bauern?
[MR5] 1293: anØ6:   cmp byte ptr[ds:di+PosBW],Ø
[P11] 1294:         jg anØ8
[TY3] 1295:         ; noch schwarze Bauern?
[US4] 1296:         cmp byte ptr[ds:di+PosBS],Ø
[P11] 1297:         jg anØ8
[ØØØ] 1298:
[S94] 1299:         ; Mattführung gg Schwarz?
[3F5] 13ØØ:         cmp word ptr[ds:si.MatDiff],5ØØ
[921] 13Ø1:         jl anØ9
[885] 13Ø2:         cmp word ptr[ds:si.MatWertS],-65Ø
[521] 13Ø3:         jl anØ8
[816] 13Ø4:         mov byte ptr[ds:si.StellungTyp],3
[Y91] 13Ø5:         jmp anØ7
[ØØØ] 13Ø6:
[VU3] 13Ø7:         ; Mattführung gg Weiss
[BF6] 13Ø8: anØ9:   cmp word ptr[ds:si.MatDiff],-5ØØ
[TØ1] 13Ø9:         jg anlØ
[AP5] 131Ø:         cmp word ptr[ds:si.MatWertW],65Ø
[P11] 1311:         jg anØ8
[416] 1312:         mov byte ptr[ds:si.StellungTyp],1
[Y91] 1313:         jmp anØ7
[ØØØ] 1314:
[2N3] 1315:         ; Remis-Stellungstyp
[EM6] 1316: anlØ:   cmp word ptr[ds:si.MatWertW],325
[P11] 1317:         jg anØ8
[3S5] 1318:         cmp word ptr[ds:si.MatWertS],-325
[521] 1319:         jl anØ8
[616] 132Ø:         mov byte ptr[ds:si.StellungTyp],2
[ØØØ] 1321:
[UM1] 1322: anØ7:   ret
[FJ2] 1323: Analyse ENDP
[ØØØ] 1324:
[ØØØ] 1325:
[PF3] 1326:         ; ********************************
[BX4] 1327:         ;          Bauernbewertung weiss
[PF3] 1328:         ; ********************************
[ØØØ] 1329:
[WU3] 133Ø: BauernWertWeiss PROC
[US2] 1331:         mov si,aktStack
[ØØØ] 1332:
[WY3] 1333:         ; überhaupt Bauern da?
[BC4] 1334:         cmp byte ptr[ds:PosBW],Ø
[BF1] 1335:         jnz bww
[3PØ] 1336:         ret
```

```
[ØØØ] 1337:
[FM2] 1338: bww:    xor ax,ax
[ØØØ] 1339:
[HB6] 134Ø:         cmp byte ptr[ds:si.EndSpielW],false
[961] 1341:         je bwwØ1
[GC1] 1342:         push si
[WE3] 1343:         mov si,offset VorES
[5E3] 1344:         mov di,offset VorBW
[NB1] 1345:         mov cx,8
[LN1] 1346:         rep movsw
[E31] 1347:         pop si
[ØØØ] 1348:
[3Z4] 1349:         ; Blockierung Zentrumsbauern
[V76] 135Ø: bwwØ1:  cmp byte ptr[ds:si+84],BauerW
[BH1] 1351:         jne bwwØ2
[BU4] 1352:         cmp byte ptr[ds:si+74],leer
[A61] 1353:         je bwwØ2
[8R1] 1354:         sub ax,1ØØØ
[Z76] 1355: bwwØ2:  cmp byte ptr[ds:si+85],BauerW
[DH1] 1356:         jne bwwØ3
[DU4] 1357:         cmp byte ptr[ds:si+75],leer
[B61] 1358:         je bwwØ3
[8R1] 1359:         sub ax,1ØØØ
[ØØØ] 136Ø:
[R34] 1361:         ; cx=Anzahl weiße Bauern
[WZ3] 1362:         ; ax=Bewertung der Bauern
[EN5] 1363: bwwØ3:  mov cl,byte ptr[ds:PosBW]
[6F1] 1364:         xor ch,ch
[ØØØ] 1365:
[865] 1366:         ; cx Nummer des weissen Bauern
[W64] 1367:         ; (wir zählen rückwärts)
[RH5] 1368:         ; di=cx,dh=Pos,bx=Spalte,dl=Zeile
[AN2] 1369: bww1:   mov di,cx
[3F1] 137Ø:         xor bh,bh
[US2] 1371:         mov si,aktStack
[CY4] 1372:         mov bl,byte ptr[ds:PosBW+di]
[1D1] 1373:         add si,bx
[9F1] 1374:         mov dh,bl
[VW4] 1375:         mov bl,byte ptr[ds:SpaBW+di]
[8X4] 1376:         mov dl,byte ptr[ds:ZeiBW+di]
[ØØØ] 1377:
[U33] 1378:         ; isolierte Bauern
[N75] 1379:         ; (isolierte Bauern haben links
[TC5] 138Ø:         ; und rechts keine eigenen Bauern)
[ØØØ] 1381:
[125] 1382:         cmp byte ptr[ds:SZaBW+bx-1],Ø
[FH1] 1383:         jne bwwØ4
[X15] 1384:         cmp byte ptr[ds:SZaBW+bx+1],Ø
[FH1] 1385:         jne bwwØ4
[KR1] 1386:         sub ax,15ØØ
[ØØØ] 1387:
[4R2] 1388:         ; Mehrfachbauern
[F45] 1389:         ; (in einer Spalte mehrere eigene
[ØC1] 139Ø:         ; Bauern)
[ØØØ] 1391:
[516] 1392: bwwØ4:  cmp byte ptr[ds:SZaBW+bx],2
[271] 1393:         jl bwwØ5
[AH1] 1394:         sub ax,3ØØ
[UU4] 1395:         cmp byte ptr[ds:SZaBW+bx],2
[D61] 1396:         je bwwØ5
[SR1] 1397:         sub ax,18ØØ
[ØØØ] 1398:
[KB4] 1399:         ; Freibauern (können durch
[TX3] 14ØØ:         ; gegn. Bauern nicht mehr
[E93] 14Ø1:         ; geschlagen werden)
[ØØØ] 14Ø2:
[S81] 14Ø3:         ; links
[686] 14Ø4: bwwØ5:  cmp byte ptr[ds:SZaBS+bx-1],Ø
[EC1] 14Ø5:         jz bwwØ5a
[FF5] 14Ø6:         cmp byte ptr[ds:SlowBS+bx-1],dh
[ZP1] 14Ø7:         jge bwwØ5a
[WK1] 14Ø8:         jmp bwwØ6
[ØY1] 14Ø9:         ; geradeaus
[M46] 141Ø: bwwØ5a: cmp byte ptr[ds:SZaBS+bx],Ø
[GC1] 1411:         jz bwwØ5b
[Y45] 1412:         cmp byte ptr[ds:SlowBS+bx],dh
[2R1] 1413:         jge bwwØ5b
[WK1] 1414:         jmp bwwØ6
[4B1] 1415:         ; rechts
[2C6] 1416: bwwØ5b: cmp byte ptr[ds:SZaBS+bx+1],Ø
[KC1] 1417:         jz bwwØ5c
[9F5] 1418:         cmp byte ptr[ds:SlowBS+bx+1],dh
[5R1] 1419:         jge bwwØ5c
[WK1] 142Ø:         jmp bwwØ6
[XR3] 1421:         ; ok, ist also Freibauer
[517] 1422: bwwØ5c: mov byte ptr[ds:FreiBW+bx],true
[6M3] 1423:         ; bewertung=+ESFaktor
[NT4] 1424:         ;          *UmwFaktor*Zeile*Zeile
[TC1] 1425:         push bx
[PC1] 1426:         push ax
[DL1] 1427:         mov bx,115
[5U4] 1428:         cmp byte ptr[ds:si-1Ø],leer
[BG2] 1429:         ; gegner davor?
[4T1] 143Ø:         jge bwwØ5x
[FD1] 1431:         mov bl,8Ø
[SF4] 1432:         ; links oder rechts hinten
[UX2] 1433:         ; eigener Bauer?
[SD6] 1434: bwwØ5x: cmp byte ptr[ds:si+11],BauerW
[HU1] 1435:         jnz bwwØ5d
[L91] 1436:         add bl,35
[9T1] 1437:         jmp bwwØ5e
[396] 1438: bwwØ5d: cmp byte ptr[ds:si+9],BauerW
[MU1] 1439:         jnz bwwØ5e
[L91] 144Ø:         add bl,35
[P91] 1441: bwwØ5e:
[DB6] 1442:         cmp byte ptr[ds:si.endspielS],false
[SC1] 1443:         jz bwwØ5f
[R81] 1444:         shl bl,1
[FT1] 1445:         jmp bwwØ5g
[T91] 1446: bwwØ5f:
[HB6] 1447:         cmp byte ptr[ds:si.endspielW],false
[UC1] 1448:         jz bwwØ5g
[R81] 1449:         shl bl,1
[9A3] 145Ø:         ; Produkt errechnen
[BU2] 1451: bwwØ5g: mov al,bl
[ØF1] 1452:         xor ah,ah
[BU3] 1453:         ; mal Quadrat der Zeile
[FA1] 1454:         mov bl,8
[1E1] 1455:         sub bl,dl
[H21] 1456:         mul bl
[H21] 1457:         mul bl
[CH1] 1458:         mov bx,ax
[R31] 1459:         pop ax
[AD1] 146Ø:         add ax,bx
[T31] 1461:         pop bx
[ØØØ] 1462:
[C54] 1463:         ; Vorrücken der Bauern
[8X4] 1464:         ; ax:=ax+(8-zeile)*Vorrückwert
[EL2] 1465: bwwØ6:  push dx
[PC1] 1466:         push ax
[V91] 1467:         shl bx,1
[GB1] 1468:         mov ax,7
[ZD1] 1469:         sub al,dl
[PK4] 147Ø:         mul word ptr[ds:VorBW+bx]
[GH1] 1471:         mov dx,ax
[Y91] 1472:         sar bx,1
[R31] 1473:         pop ax
[CD1] 1474:         add ax,dx
[X31] 1475:         pop dx
[ØØØ] 1476:
[ØP3] 1477:         ; rückständige bauern
[9X4] 1478:         ; (Bauern, die nicht durch eigene
[765] 1479:         ; Bauern gedeckt sind (neben oder
[5Y4] 148Ø:         ; schräg-hinter) aber am Vor-
[MH4] 1481:         ; rücken durch gegn. Bauern
[F93] 1482:         ; gehindert werden)
```

```
[ØØØ] 1483:
[GY4] 1484:         cmp byte ptr[ds:si-1],BauerW
[E81] 1485:         jz bwwØ7
[8Y4] 1486:         cmp byte ptr[ds:si+1],BauerW
[E81] 1487:         jz bwwØ7
[GY4] 1488:         cmp byte ptr[ds:si+9],BauerW
[E81] 1489:         jz bwwØ7
[915] 149Ø:         cmp byte ptr[ds:si+11],BauerW
[E81] 1491:         jz bwwØ7
[T15] 1492:         cmp byte ptr[ds:si-19],BauerS
[9M1] 1493:         jnz bwwØ7
[9H1] 1494:         sub ax,2ØØ
[D55] 1495:         sub byte ptr[ds:RuckBS+di],2ØØ
[A15] 1496:         cmp byte ptr[ds:si-21],BauerS
[9M1] 1497:         jnz bwwØ7
[9H1] 1498:         sub ax,2ØØ
[D55] 1499:         sub byte ptr[ds:RuckBS+di],2ØØ
[ØØØ] 15ØØ:
[272] 15Ø1: bwwØ7:  dec cx
[F81] 15Ø2:         jz bwwØ8
[CG1] 15Ø3:         jmp bww1
[ØØØ] 15Ø4:
[PW1] 15Ø5: bwwØ8:  ret
[VS3] 15Ø6: BauernWertWeiss ENDP
[ØØØ] 15Ø7:
[PF3] 15Ø8:         ; ********************************
[2Z4] 15Ø9:         ;          Bauernbewertung schwarz
[PF3] 151Ø:         ; ********************************
[ØØØ] 1511:
[ØK3] 1512: BauernWertSchw PROC
[US2] 1513:         mov si,aktStack
[ØØØ] 1514:
[C75] 1515:         ; überhaupt Bauern vorhanden?
[3C4] 1516:         cmp byte ptr[ds:PosBS],Ø
[VE1] 1517:         jnz bws
[3PØ] 1518:         ret
[ØØØ] 1519:
[ZL2] 152Ø: bws:    xor ax,ax
[ØØØ] 1521:
[DB6] 1522:         cmp byte ptr[ds:si.EndSpielS],false
[X51] 1523:         je bwsØ1
[GC1] 1524:         push si
[WE3] 1525:         mov si,offset VorES
[ND3] 1526:         mov di,offset VorBS
[NB1] 1527:         mov cx,8
[LN1] 1528:         rep movsw
[E31] 1529:         pop si
[ØØØ] 153Ø:
[3Z4] 1531:         ; Blockierung Zentrumsbauern
[Y66] 1532: bwsØ1:  cmp byte ptr[ds:si+34],BauerS
[VG1] 1533:         jne bwsØ2
[8U4] 1534:         cmp byte ptr[ds:si+44],leer
[Y51] 1535:         je bwsØ2
[TM1] 1536:         add ax,1ØØØ
[276] 1537: bwsØ2:  cmp byte ptr[ds:si+35],BauerS
[XG1] 1538:         jne bwsØ3
[AU4] 1539:         cmp byte ptr[ds:si+45],leer
[Z51] 154Ø:         je bwsØ3
[TM1] 1541:         add ax,1ØØØ
[ØØØ] 1542:
[KS4] 1543:         ; cx=Anzahl schwarzen Bauern
[WZ3] 1544:         ; ax=Bewertung der Bauern
[UM5] 1545: bwsØ3:  mov cl,byte ptr[ds:PosBS]
[6F1] 1546:         xor ch,ch
[ØØØ] 1547:
[SF5] 1548:         ; cx Nummer des schwarzen Bauern
[W64] 1549:         ; (wir zählen rückwärts)
[RH5] 155Ø:         ; di=cx,dh=Pos,bx=Spalte,dl=Zeile
[UM2] 1551: bwsl:   mov di,cx
[3F1] 1552:         xor bh,bh
[US2] 1553:         mov si,aktStack
[8Y4] 1554:         mov bl,byte ptr[ds:PosBS+di]
[1D1] 1555:         add si,bx
[9F1] 1556:         mov dh,bl
[RW4] 1557:         mov bl,byte ptr[ds:SpaBS+di]
[4X4] 1558:         mov dl,byte ptr[ds:ZeiBS+di]
[ØØØ] 1559:
[U33] 156Ø:         ; isolierte Bauern
[N75] 1561:         ; (isolierte Bauern haben links
[TC5] 1562:         ; und rechts keine eigenen Bauern)
[ØØØ] 1563:
[T15] 1564:         cmp byte ptr[ds:SZaBS+bx-1],Ø
[ZG1] 1565:         jne bwsØ4
[N15] 1566:         cmp byte ptr[ds:SZaBS+bx+1],Ø
[ZG1] 1567:         jne bwsØ4
[3N1] 1568:         add ax,15ØØ
[ØØØ] 1569:
[4R2] 157Ø:         ; Mehrfachbauern
[F45] 1571:         ; (in einer Spalte mehrere eigene
[ØC1] 1572:         ; Bauern)
[ØØØ] 1573:
[DØ6] 1574: bwsØ4:  cmp byte ptr[ds:SZaBS+bx],2
[P61] 1575:         jl bwsØ5
[VE1] 1576:         add ax,3ØØ
[KU4] 1577:         cmp byte ptr[ds:SZaBS+bx],2
[161] 1578:         je bwsØ5
[9N1] 1579:         add ax,18ØØ
[ØØØ] 158Ø:
[KB4] 1581:         ; Freibauern (können durch
[TX3] 1582:         ; gegn. Bauern nicht mehr
[E93] 1583:         ; geschlagen werden)
[ØØØ] 1584:
[S81] 1585:         ; links
[Y76] 1586: bwsØ5:  cmp byte ptr[ds:SZaBW+bx-1],Ø
[2C1] 1587:         jz bwsØ5a
[VF5] 1588:         cmp byte ptr[ds:SlowBW+bx-1],dh
[YP1] 1589:         jle bwsØ5a
[CK1] 159Ø:         jmp bwsØ6
[ØY1] 1591:         ; geradeaus
[C46] 1592: bwsØ5a: cmp byte ptr[ds:SZaBW+bx],Ø
[4C1] 1593:         jz bwsØ5b
[A55] 1594:         cmp byte ptr[ds:SlowBW+bx],dh
[1R1] 1595:         jle bwsØ5b
[CK1] 1596:         jmp bwsØ6
[4B1] 1597:         ; rechts
[UB6] 1598: bwsØ5b: cmp byte ptr[ds:SZaBW+bx+1],Ø
[6C1] 1599:         jz bwsØ5c
[NF5] 16ØØ:         cmp byte ptr[ds:SlowBW+bx+1],dh
[4R1] 16Ø1:         jle bwsØ5c
[CK1] 16Ø2:         jmp bwsØ6
[XR3] 16Ø3:         ; ok, ist also Freibauer
[DC6] 16Ø4: bwsØ5c: mov byte ptr[ds:FreiBS+bx],-1
[4S3] 16Ø5:         ; bewertung=+ESFaktor*
[5F4] 16Ø6:         ;        UmwFaktor*Zeile*Zeile
[TC1] 16Ø7:         push bx
[PC1] 16Ø8:         push ax
[9K1] 16Ø9:         mov bl,115
[XT4] 161Ø:         cmp byte ptr[ds:si+1Ø],leer
[BG2] 1611:         ; gegner davor?
[3T1] 1612:         jle bwsØ5x
[FD1] 1613:         mov bl,8Ø
[DB3] 1614:         ; links oder rechts
[CL3] 1615:         ; hinten eigener Bauer
[8D6] 1616: bwsØ5x: cmp byte ptr[ds:si-11],BauerS
[1U1] 1617:         jnz bwsØ5d
[L91] 1618:         add bl,35
[TS1] 1619:         jmp bwsØ5e
[R86] 162Ø: bwsØ5d: cmp byte ptr[ds:si-9],BauerS
[4U1] 1621:         jnz bwsØ5e
[L91] 1622:         add bl,35
[691] 1623: bwsØ5e:
[DB6] 1624:         cmp byte ptr[ds:si.endspielS],false
[CC1] 1625:         jz bwsØ5f
[R81] 1626:         shl bl,1
[ZS1] 1627:         jmp bwsØ5g
[991] 1628: bwsØ5f:
```

```
[HB6] 1629:         cmp byte ptr[ds:si.endspielW],false
[EC1] 163Ø:         jz bwsØ5g
[R81] 1631:         shl bl,1
[9A3] 1632:         ; Produkt errechnen
[VT2] 1633: bwsØ5g: mov al,bl
[ØF1] 1634:         xor ah,ah
[BU3] 1635:         ; mal Quadrat der Zeile
[LF1] 1636:         mov bl,dl
[H21] 1637:         mul bl
[H21] 1638:         mul bl
[CH1] 1639:         mov bx,ax
[R31] 164Ø:         pop ax
[TF1] 1641:         sub ax,bx
[T31] 1642:         pop bx
[ØØØ] 1643:
[C54] 1644:         ; Vorrücken der Bauern
[745] 1645:         ; ax:=ax-(spalte)*Vorrückwert
[YK2] 1646: bwsØ6:  push dx
[PC1] 1647:         push ax
[V91] 1648:         shl bx,1
[ØF1] 1649:         xor ah,ah
[HF1] 165Ø:         mov al,dl
[9ZØ] 1651:         dec al
[92Ø] 1652:         dec al
[EK4] 1653:         mul word ptr[ds:VorBS+bx]
[GH1] 1654:         mov dx,ax
[Y91] 1655:         sar bx,1
[R31] 1656:         pop ax
[VF1] 1657:         sub ax,dx
[X31] 1658:         pop dx
[ØØØ] 1659:
[ØP3] 166Ø:         ; rückständige bauern
[ØØØ] 1661:
[CY4] 1662:         cmp byte ptr[ds:si-1],BauerS
[281] 1663:         jz bwsØ7
[4V4] 1664:         cmp byte ptr[ds:si+1],BauerS
[281] 1665:         jz bwsØ7
[MY4] 1666:         cmp byte ptr[ds:si-9],BauerS
[281] 1667:         jz bwsØ7
[915] 1668:         cmp byte ptr[ds:si-11],BauerS
[281] 1669:         jz bwsØ7
[T15] 167Ø:         cmp byte ptr[ds:si+19],BauerW
[TLl] 1671:         jnz bwsØ7
[UE1] 1672:         add ax,2ØØ
[A35] 1673:         add byte ptr[ds:RuckBW+di],2ØØ
[A15] 1674:         cmp byte ptr[ds:si+21],BauerW
[TLl] 1675:         jnz bwsØ7
[UE1] 1676:         add ax,2ØØ
[A35] 1677:         add byte ptr[ds:RuckBW+di],2ØØ
[ØØØ] 1678:
[K62] 1679: bwsØ7:  dec cx
[381] 168Ø:         jz bwsØ8
[WF1] 1681:         jmp bws1
[ØØØ] 1682:
[6W1] 1683: bwsØ8:  ret
[ØK3] 1684: BauernWertSchw ENDP
[ØØØ] 1685:
[PF3] 1686:         ; ********************************
[PT4] 1687:         ;                 Figurenwert weiß
[PF3] 1688:         ; ********************************
[ØØØ] 1689:
[XF3] 169Ø: FigurWertWeiss PROC
[US2] 1691:         mov si,aktStack
[GH1] 1692:         xor ax,ax
[ØØØ] 1693:
[Y44] 1694:         ; cx Anzahl Figuren weiß
[TD4] 1695:         ; di Spalte aktuelle Figur
[2M4] 1696:         ; bx Position aktuelle Figur
[6F1] 1697:         xor ch,ch
[NF4] 1698:         mov cl,byte ptr[ds:PosFW]
[KN2] 1699: fww1:   mov di,cx
[WY4] 17ØØ:         mov bl,byte ptr[ds:PosFW+di]
[3F1] 17Ø1:         xor bh,bh
[625] 17Ø2:         mov dl,byte ptr[ds:Spalte+bx]
[9F1] 17Ø3:         xor dh,dh
[6G1] 17Ø4:         mov di,dx
[ØØØ] 17Ø5:
[8S5] 17Ø6:         cmp byte ptr[ds:si+bx],SpringerW
[5M1] 17Ø7:         jnz fwwØ1
[YA3] 17Ø8:         ; weißer Springer
[V91] 17Ø9:         shl bx,1
[4Z4] 171Ø:         add ax,word ptr[ds:SprTW+bx]
[Y91] 1711:         sar bx,1
[A44] 1712:         inc byte ptr[ds:SumFW]
[YB1] 1713:         cmp bl,9Ø
[171] 1714:         jl fwwØ9
[534] 1715:         inc byte ptr[ds:GrLFW]
[TH1] 1716:         sub ax,47Ø
[NT2] 1717: fwwØ9:  jmp fwwØ9
[ØØØ] 1718:
[HW6] 1719: fwwØ1:  cmp byte ptr[ds:si+bx],LaeuferW
[7M1] 172Ø:         jnz fwwØ2
[6B3] 1721:         ; weißer Läufer
[5B4] 1722:         cmp byte ptr[ds:LauFW],Ø
[EC1] 1723:         jz fwwØ1a
[R64] 1724:         ; noch zwei Läufer-Bonus
[WE1] 1725:         add ax,4ØØ
[2R5] 1726: fwwØ1a: mov byte ptr[ds:LauFW],1
[A44] 1727:         inc byte ptr[ds:SumFW]
[YB1] 1728:         cmp bl,9Ø
[6B1] 1729:         jl fwwØ1b
[534] 173Ø:         inc byte ptr[ds:GrlFW]
[PH1] 1731:         sub ax,55Ø
[RX2] 1732: fwwØ1b: jmp fwwØ9
[ØØØ] 1733:
[Z76] 1734: fwwØ2:  cmp byte ptr[ds:si+bx],TurmW
[9M1] 1735:         jnz fwwØ3
[P56] 1736:         ; weißer Turm vor gegn.Grundreihe?
[A44] 1737:         inc byte ptr[ds:SumFW]
[TB1] 1738:         cmp bl,4Ø
[5B1] 1739:         jl fwwØ2a
[AC1] 174Ø:         cmp bl,39
[PA1] 1741:         jg fwwØ2a
[XE1] 1742:         add ax,5ØØ
[AP4] 1743:         cmp byte ptr[ds:PosFS],3Ø
[PA1] 1744:         jg fwwØ2a
[YE1] 1745:         add ax,6ØØ
[WP2] 1746:         ; Turm-Bauer Lage
[R46] 1747: fwwØ2a: cmp byte ptr[ds:SZaBW+di],Ø
[HC1] 1748:         jz fwwØ2b
[TM1] 1749:         add ax, 15Ø
[3U1] 175Ø:         cmp byte ptr[ds:SZaBS+di],Ø
[DU1] 1751:         jnz fwwØ2b
[4F1] 1752:         add ax,25Ø
[SX4] 1753:         ; zweiter Turm gleiche Spalte?
[4V2] 1754: fwwØ2b: mov dx,di
[YX4] 1755:         cmp byte ptr[ds:TurmSW],dl
[WR1] 1756:         jne fwwØ2c
[ØF1] 1757:         add ax,8ØØ
[2A6] 1758: fwwØ2c: mov byte ptr[ds:TurmSW],dl
[5A5] 1759:         ; Freibauern auf dieser Spalte?
[ØX4] 176Ø:         cmp byte ptr[ds:FreiBS+di],Ø
[NC1] 1761:         jz fwwØ2d
[TM1] 1762:         add ax,19ØØ
[YK1] 1763:         jmp fwwØ3
[EØ7] 1764: fwwØ2d: cmp byte ptr[ds:FreiBW+di],false
[E81] 1765:         jz fwwØ3
[TM1] 1766:         add ax,19ØØ
[AL1] 1767:         jmp fwwØ9
[ØØØ] 1768:
[GWØ] 1769:         ; Dame
[U46] 177Ø: fwwØ3:  cmp byte ptr[ds:si+bx],DameW
[BM1] 1771:         jnz fwwØ4
[3A4] 1772:         add byte ptr[ds:SumFW],3
[EV4] 1773:         mov byte ptr[ds:DamePW],bl
[YB1] 1774:         cmp bl,9Ø
[A71] 1775:         jl fwwØ9
[Y84] 1776:         add byte ptr[ds:GrlFW],3
[AL1] 1777:         jmp fwwØ9
[ØØØ] 1778:
[3W5] 1779:         ; König (nur Position feststellen)
[GK6] 178Ø: fwwØ4:  cmp byte ptr[ds:si+bx],KoenigW
[1K1] 1781:         jne fwwØ9
[565] 1782:         mov byte ptr[ds:si.KoenPW],bl
[ØØØ] 1783:
[E72] 1784: fwwØ9:  dec cx
[F81] 1785:         jz fwwlØ
[MG1] 1786:         jmp fww1
[FW1] 1787: fwwlØ:  ret
[XF3] 1788: FigurWertWeiss ENDP
[ØØØ] 1789:
[PF3] 179Ø:         ; ********************************
[YM4] 1791:         ;                 Figurenwert schwarz
[PF3] 1792:         ; ********************************
[ØØØ] 1793:
[373] 1794: FigurWertSchw PROC
[US2] 1795:         mov si,aktStack
[GH1] 1796:         xor ax,ax
[ØØØ] 1797:
[2R4] 1798:         ; cx Anzahl Figuren schwarz
[TD4] 1799:         ; di Spalte aktuelle Figur
[2M4] 18ØØ:         ; bx Position aktuelle Figur
[6F1] 18Ø1:         xor ch,ch
[HF4] 18Ø2:         mov cl,byte ptr[ds:PosFS]
[2N2] 18Ø3: fwsl:   mov di,cx
[SY4] 18Ø4:         mov bl,byte ptr[ds:PosFS+di]
[3F1] 18Ø5:         xor bh,bh
[625] 18Ø6:         mov dl,byte ptr[ds:Spalte+bx]
[9F1] 18Ø7:         xor dh,dh
[6G1] 18Ø8:         mov di,dx
[ØØØ] 18Ø9:
[4S5] 181Ø:         cmp byte ptr[ds:si+bx],SpringerS
[NL1] 1811:         jnz fwsØ1
[TH3] 1812:         ; schwarzer Springer
[V91] 1813:         shl bx,1
[L15] 1814:         sub ax,word ptr[ds:SprTW+bx]
[Y91] 1815:         sar bx,1
[244] 1816:         inc byte ptr[ds:SumFS]
[SB1] 1817:         cmp bl,3Ø
[661] 1818:         jg fwsØØ
[X24] 1819:         inc byte ptr[ds:GrLFS]
[AF1] 182Ø:         add ax,47Ø
[NS2] 1821: fwsØØ:  jmp fwsØ9
[ØØØ] 1822:
[HV6] 1823: fwsØ1:  cmp byte ptr[ds:si+bx],LaeuferS
[RL1] 1824:         jnz fwsØ2
[RC3] 1825:         ; schwarzer Läufer
[XA4] 1826:         cmp byte ptr[ds:LauFS],Ø
[2C1] 1827:         jz fwsØ1a
[9M4] 1828:         ; noch zwei Läufer => Bonus
[BH1] 1829:         sub ax,4ØØ
[AP5] 183Ø: fwsØ1a: mov byte ptr[ds:LauFS],1
[244] 1831:         inc byte ptr[ds:SumFS]
[SB1] 1832:         cmp bl,3Ø
[BA1] 1833:         jg fwsØ1b
[X24] 1834:         inc byte ptr[ds:GrlFS]
[7F1] 1835:         add ax,55Ø
[RW2] 1836: fwsØ1b: jmp fwsØ9
[ØØØ] 1837:
[B76] 1838: fwsØ2:  cmp byte ptr[ds:si+bx],TurmS
[TL1] 1839:         jnz fwsØ3
[ØH5] 184Ø:         ; schw. Turm vor gegn.Grundreihe?
[244] 1841:         inc byte ptr[ds:SumFS]
[XB1] 1842:         cmp bl,8Ø
[TA1] 1843:         jl fwsØ2a
[FC1] 1844:         cmp bl,89
[AA1] 1845:         jg fwsØ2a
[CH1] 1846:         sub ax,5ØØ
[9G4] 1847:         cmp byte ptr[ds:PosFW],89
[TA1] 1848:         jl fwsØ2a
[DH1] 1849:         sub ax,6ØØ
[WP2] 185Ø:         ; Turm-Bauer Lage
[Z36] 1851: fwsØ2a: cmp byte ptr[ds:SZaBS+di],Ø
[5C1] 1852:         jz fwsØ2b
[8R1] 1853:         sub ax, 15Ø
[BU4] 1854:         cmp byte ptr[ds:SZaBW+di],Ø
[XT1] 1855:         jnz fwsØ2b
[LH1] 1856:         sub ax,25Ø
[SX4] 1857:         ; zweiter Turm gleiche Spalte?
[MU2] 1858: fwsØ2b: mov dx,di
[KX4] 1859:         cmp byte ptr[ds:TurmSS],dl
[CR1] 186Ø:         jne fwsØ2c
[FH1] 1861:         sub ax,8ØØ
[696] 1862: fwsØ2c: mov byte ptr[ds:TurmSS],dl
[5A5] 1863:         ; Freibauern auf dieser Spalte?
[TN5] 1864:         cmp byte ptr[ds:FreiBW+di],false
[9C1] 1865:         jz fwsØ2d
[8R1] 1866:         sub ax,19ØØ
[EK1] 1867:         jmp fwsØ3
[576] 1868: fwsØ2d: cmp byte ptr[ds:FreiBS+di],Ø
[281] 1869:         jz fwsØ3
[8R1] 187Ø:         sub ax,19ØØ
[UK1] 1871:         jmp fwsØ9
[ØØØ] 1872:
[GWØ] 1873:         ; Dame
[646] 1874: fwsØ3:  cmp byte ptr[ds:si+bx],DameS
[VL1] 1875:         jnz fwsØ4
[V94] 1876:         add byte ptr[ds:SumFS],3
[2V4] 1877:         mov byte ptr[ds:DamePS],bl
[SB1] 1878:         cmp bl,3Ø
[F61] 1879:         jg fwsØ9
[P84] 188Ø:         add byte ptr[ds:GrlFS],3
[UK1] 1881:         jmp fwsØ9
[ØØØ] 1882:
[3W5] 1883:         ; König (nur Position feststellen)
[MH6] 1884: fwsØ4:  cmp byte ptr[ds:si+bx],KoenigS
[HH1] 1885:         jne fwsØ9
[X55] 1886:         mov byte ptr[ds:si.KoenPS],bl
[ØØØ] 1887:
[Y62] 1888: fwsØ9:  dec cx
[381] 1889:         jz fwslØ
[4G1] 189Ø:         jmp fwsl
[ZV1] 1891: fwslØ:  ret
[463] 1892: FigurWertSchw ENDP
[ØØØ] 1893:
[PF3] 1894:         ; ********************************
[RS4] 1895:         ;                 Königswert weiß
[PF3] 1896:         ; ********************************
[ØØØ] 1897:
[Y85] 1898:         ; gibt in ax den Abstand von cl,ch
[T42] 1899: Abstand PROC
[GH1] 19ØØ:         xor ax,ax
[3F1] 19Ø1:         xor bh,bh
[KF1] 19Ø2:         mov bl,cl
[625] 19Ø3:         mov dl,byte ptr[ds:spalte+bx]
[EZ4] 19Ø4:         mov al,byte ptr[ds:zeile+bx]
[AF1] 19Ø5:         mov bl,ch
[UL5] 19Ø6:         mov dh,byte ptr[ds:spalte+bx]
[2Z4] 19Ø7:         mov ah,byte ptr[ds:zeile+bx]
[ØØØ] 19Ø8:
[ZN2] 19Ø9:         ; Zeilenabstand
[MD1] 191Ø:         cmp al,ah
[9C1] 1911:         jg abstØ1
[NM1] 1912:         xchg ah,al
[RM2] 1913: abstØ1: sub al,ah
[ØØØ] 1914:
[ES2] 1915:         ; Spaltenabstand
[3N2] 1916: abstØ2: cmp dl,dh
[DC1] 1917:         jg abstØ3
[4N1] 1918:         xchg dh,dl
[6N2] 1919: abstØ3: sub dl,dh
[ØØØ] 192Ø:
[CØ2] 1921:         ; summieren
[GB1] 1922:         add al,dl
[ØF1] 1923:         xor ah,ah
[ØØØ] 1924:
[3PØ] 1925:         ret
[S22] 1926: Abstand ENDP
[ØØØ] 1927:
[PF3] 1928:         ; ********************************
[M16] 1929:         ;   bewertet Abstände Bauern-König
[GE4] 193Ø:         ;          in cl den Standort des
[YX5] 1931:         ;   betreffenden Königs übergeben,
[XW4] 1932:         ;          addiert Bewertung zu ax
[PF3] 1933:         ; ********************************
[ØØØ] 1934:
[WN3] 1935: BauerKoenigAbst PROC
[PC1] 1936:         push ax
[GH1] 1937:         xor ax,ax
[TH1] 1938:         xor dx,dx
[FZ5] 1939:         ; gewichteter Bauernabstand (weiß)
[3F1] 194Ø:         xor bh,bh
[3F4] 1941:         mov bl,byte ptr[ds:PosBW]
[661] 1942:         or bl,bl
[981] 1943:         jz bkas
[126] 1944: bkaØ1:  mov ch,byte ptr[ds:PosBW+bx]
[WC1] 1945:         push cx
[PC1] 1946:         push ax
[ZC1] 1947:         push dx
[TC1] 1948:         push bx
[AØ2] 1949:         call Abstand
[X31] 195Ø:         pop dx
[FH1] 1951:         mov bx,dx
[HY4] 1952:         mov bl,byte ptr[ds:SpaBW+bx]
[3N5] 1953:         cmp byte ptr[ds:FreiBW+bx],true
[FH1] 1954:         mov bx,dx
[FD1] 1955:         jne bkaØ2
[5L1] 1956:         mov dx,6ØØ
[T31] 1957:         mul dx
[X31] 1958:         pop dx
[M71] 1959:         add dx,6
[UE1] 196Ø:         jmp bkaØ3
[CN2] 1961: bkaØ2:  mov dx,2ØØ
[T31] 1962:         mul dx
[X31] 1963:         pop dx
[G71] 1964:         add dx,2
[VL2] 1965: bkaØ3:  mov cx,ax
[R31] 1966:         pop ax
[BD1] 1967:         add ax,cx
[V31] 1968:         pop cx
[FØ1] 1969:         dec bx
[1G1] 197Ø:         jnz bkaØ1
[ØØØ] 1971:
[LG2] 1972: bkas:   xor bh,bh
[ZE4] 1973:         mov bl,byte ptr[ds:PosBS]
[661] 1974:         or bl,bl
[H81] 1975:         jz bkadd
[326] 1976: bkaØ4:  mov ch,byte ptr[ds:PosBS+bx]
[WC1] 1977:         push cx
[PC1] 1978:         push ax
[ZC1] 1979:         push dx
[TC1] 198Ø:         push bx
[AØ2] 1981:         call Abstand
[X31] 1982:         pop dx
[FH1] 1983:         mov bx,dx
[DY4] 1984:         mov bl,byte ptr[ds:SpaBS+bx]
[RM5] 1985:         cmp byte ptr[ds:FreiBS+bx],true
[FH1] 1986:         mov bx,dx
[ND1] 1987:         jne bkaØ5
[5L1] 1988:         mov dx,6ØØ
[T31] 1989:         mul dx
[X31] 199Ø:         pop dx
[M71] 1991:         add dx,6
[ØF1] 1992:         jmp bkaØ6
[KN2] 1993: bkaØ5:  mov dx,2ØØ
[T31] 1994:         mul dx
[X31] 1995:         pop dx
[G71] 1996:         add dx,2
[1M2] 1997: bkaØ6:  mov cx,ax
[R31] 1998:         pop ax
[BD1] 1999:         add ax,cx
[V31] 2ØØØ:         pop cx
[FØ1] 2ØØ1:         dec bx
[7G1] 2ØØ2:         jnz bkaØ4
[ØØØ] 2ØØ3:
[5C5] 2ØØ4:         ; mitteln, dann wert=-6*(Mittel-6)
[HH1] 2ØØ5:         mov cx,dx
[TH1] 2ØØ6:         xor dx,dx
[931] 2ØØ7:         div cx
[DH1] 2ØØ8:         sub ax,6ØØ
[4T1] 2ØØ9:         mov bx,-6ØØ
[ZB1] 2Ø1Ø:         imul bx
[W72] 2Ø11: bkadd:  pop bx
[AD1] 2Ø12:         add ax,bx
[3PØ] 2Ø13:         ret
[VL3] 2Ø14: BauerKoenigAbst ENDP
[ØØØ] 2Ø15:
[PF3] 2Ø16:         ; ********************************
[UM6] 2Ø17:         ; König Bewertung für weißen König
[PF3] 2Ø18:         ; ********************************
[ØØØ] 2Ø19:
[UT3] 2Ø2Ø: KoenigWertWeiss PROC
[SE5] 2Ø21:         ; Endspiel oder Normalbewertung
[E46] 2Ø22:         cmp byte ptr[ds:si.EndSpielW],true
[TH1] 2Ø23:         jne kwwØØ
[5L1] 2Ø24:         jmp kww11
[ØØØ] 2Ø25:
[WZ2] 2Ø26:         ; Normalbewertung
[N11] 2Ø27: kwwØØ:
[KB6] 2Ø28:       mov al,byte ptr[ds:si.rochadefaktor]
[8NØ] 2Ø29:         cbw
[ØØØ] 2Ø3Ø:
[SN5] 2Ø31:         ; Abzug für exponierte Stellung
[H25] 2Ø32:         cmp byte ptr[ds:si.KoenPW],8Ø
[S61] 2Ø33:         jg kwwØ1
[UM1] 2Ø34:         add ax,2ØØØ
[LH1] 2Ø35:         xor bx,bx
[3L1] 2Ø36:         jmp kww1Ø
[ØØØ] 2Ø37:
[HR5] 2Ø38:         ; Ermitteln der Gefährdungswerte
[WZ1] 2Ø39:         ; der Flügel
[6Z5] 2Ø4Ø: kwwØ1:  mov word ptr[ds:schutzL],Ø
[7T4] 2Ø41:         mov word ptr[ds:schutzR],Ø
[VM2] 2Ø42:         ; linker Flügel
[CB1] 2Ø43:         mov bx,1
[LR6] 2Ø44: kwwØ2:  cmp byte ptr[ds:si+bx+8Ø],BauerW
[M61] 2Ø45:         je kwwØ3
[XU4] 2Ø46:         add word ptr[ds:schutzL],5Ø
[KU4] 2Ø47:         cmp byte ptr[ds:SZaBW+bx],Ø
[ZH1] 2Ø48:         jne kwwØ3
[XU4] 2Ø49:         add word ptr[ds:schutzL],5Ø
[F82] 2Ø5Ø: kwwØ3:  inc bx
[X91] 2Ø51:         cmp bx,4
[871] 2Ø52:         jl kwwØ2
[Ø85] 2Ø53:         ; Rochade auf linkem Flügel mgl?
[AB6] 2Ø54:       cmp byte ptr[ds:si.RochadeErfolgt],Ø
[1K1] 2Ø55:         jne kwwØ4
[XU4] 2Ø56:         add word ptr[ds:schutzL],5Ø
[YE6] 2Ø57:       cmp byte ptr[ds:si.grRochWmgl],true
[N61] 2Ø58:         je kwwØ4
[NW4] 2Ø59:         add word ptr[ds:schutzL],15Ø
[ØØØ] 2Ø6Ø:
[7T2] 2Ø61:         ; rechter Flügel
[FK2] 2Ø62: kwwØ4:  mov bx,7
```

»Chessfx.asm« verspricht ungetrübten Schachgenuß.

```
[TR6] 2Ø63: kww Ø5:   cmp byte ptr[ds:si+bx+8Ø],BauerW
[R61] 2Ø64:           je kww Ø6
[NV4] 2Ø65:           add word ptr[ds:schutzR],5Ø
[KU4] 2Ø66:           cmp byte ptr[ds:SZaBW+bx],Ø
[5K1] 2Ø67:           jne kww Ø6
[NV4] 2Ø68:           add word ptr[ds:schutzR],5Ø
[N82] 2Ø69: kww Ø6:   inc bx
[2A1] 2Ø7Ø:           cmp bx,9
[B71] 2Ø71:           jl kww Ø5
[PC5] 2Ø72:           ; Rochade auf rechtem Flügel mgl?
[AB6] 2Ø73:       cmp byte ptr[ds:si.RochadeErfolgt],Ø
[7K1] 2Ø74:           jne kww Ø7
[NV4] 2Ø75:           add word ptr[ds:schutzR],5Ø
[PE6] 2Ø76:         cmp byte ptr[ds:si.klRochWmgl],true
[S61] 2Ø77:           je kww Ø7
[DX4] 2Ø78:           add word ptr[ds:schutzR],15Ø
[ØØØ] 2Ø79:
[A35] 2Ø8Ø:           ; je nachdem, ob rochiert wurde
[BS4] 2Ø81:           ; den Flügel mit dem besseren
[415] 2Ø82:           ; Gefährdungswert aussuchen
[321] 2Ø83: kww Ø7:
[AB6] 2Ø84:       cmp byte ptr[ds:si.RochadeErfolgt],Ø
[9K1] 2Ø85:           jne kww Ø8
[N35] 2Ø86:           mov bx,word ptr[ds:schutzr]
[H15] 2Ø87:           cmp bx,word ptr[ds:schutzl]
[A71] 2Ø88:           jl kww lØ
[335] 2Ø89:           mov bx,word ptr[ds:schutzl]
[3L1] 2Ø9Ø:           jmp kww lØ
[ØØØ] 2Ø91:
[521] 2Ø92: kww Ø8:
[4A6] 2Ø93:       cmp [ds:si.RochadeErfolgt],klRochade
[BK1] 2Ø94:           jne kww Ø9
[N35] 2Ø95:           mov bx,word ptr[ds:schutzR]
[3L1] 2Ø96:           jmp kww lØ
[AA6] 2Ø97: kww Ø9:   mov bx,word ptr[ds:schutzL]
[ØØØ] 2Ø98:
[A55] 2Ø99:           ; gefundene Summe mit gegnerischem
[N93] 21ØØ:           ; Figurenwert mult.
[ØM2] 21Ø1: kww lØ:   add ax,bx
[X24] 21Ø2:           dec byte ptr[ds:SumFS]
[364] 21Ø3:           mul byte ptr[ds:SumFS]
[H11] 21Ø4:           neg ax
[3PØ] 21Ø5:           ret
[ØØØ] 21Ø6:
[6E3] 21Ø7:           ; Endspielbewertung
[W23] 21Ø8:           ; Zentralisierung
[VN2] 21Ø9: kww ll:   xor bh,bh
[K45] 211Ø:           mov bl,byte ptr[ds:si.KoenPW]
[V91] 2111:           shl bx,1
[RØ5] 2112:           mov ax,word ptr[ds:ZAbst+bx]
[T91] 2113:           shl ax,1
[T91] 2114:           shl ax,1
[H11] 2115:           neg ax
[ØØØ] 2116:
[M45] 2117:           mov cl,byte ptr[ds:si.KoenPW]
[6F3] 2118:           call BauerKoenigAbst
[3PØ] 2119:           ret
[TR3] 212Ø: KoenigWertWeiss ENDP
[ØØØ] 2121:
[PF3] 2122:           ; ******************************
[MØ6] 2123:           ;  König Bewertung für schw. König
[PF3] 2124:           ; ******************************
[ØØØ] 2125:
[YG3] 2126: KoenigWertSchw PROC
[SE5] 2127:           ; Endspiel oder Normalbewertung
[A46] 2128:           cmp byte ptr[ds:si.EndSpielS],true
[9H1] 2129:           jne kwsØØ
[NK1] 213Ø:           jmp kws ll
[ØØØ] 2131:
[WZ2] 2132:           ; Normalbewertung
[511] 2133: kwsØØ:
[KB6] 2134:         mov al,byte ptr[ds:si.rochadefaktor]
[8NØ] 2135:           cbw
[ØØØ] 2136:
[SN5] 2137:           ; Abzug für exponierte Stellung
[525] 2138:           cmp byte ptr[ds:si.KoenPS],4Ø
[V61] 2139:           jl kwsØ1
[UM1] 214Ø:           add ax,2ØØØ
[LH1] 2141:           xor bx,bx
[LK1] 2142:           jmp kws lØ
[ØØØ] 2143:
[HR5] 2144:           ; Ermitteln der Gefährdungswerte
[WZ1] 2145:           ; der Flügel
[PY5] 2146: kwsØ1:    mov word ptr[ds:schutzL],Ø
[7T4] 2147:           mov word ptr[ds:schutzR],Ø
[VM2] 2148:           ; linker Flügel
[CB1] 2149:           mov bx,1
[BP6] 215Ø: kwsØ2:    cmp byte ptr[ds:si+bx+3Ø],BauerS
[861] 2151:           je kwsØ3
[XU4] 2152:           add word ptr[ds:schutzL],5Ø
[AU4] 2153:           cmp byte ptr[ds:SZaBS+bx],Ø
[FH1] 2154:           jne kwsØ3
[XU4] 2155:           add word ptr[ds:schutzL],5Ø
[Z72] 2156: kwsØ3:    inc bx
[X91] 2157:           cmp bx,4
[W61] 2158:           jl kwsØ2
[Ø85] 2159:           ; Rochade auf linkem Flügel mgl?
[AB6] 216Ø:       cmp byte ptr[ds:si.RochadeErfolgt],Ø
[HH1] 2161:           jne kwsØ4
[XU4] 2162:           add word ptr[ds:schutzL],5Ø
[KE6] 2163:         cmp byte ptr[ds:si.grRochSmgl],true
[961] 2164:           je kwsØ4
[NW4] 2165:           add word ptr[ds:schutzL],15Ø
[ØØØ] 2166:
[7T2] 2167:           ; rechter Flügel
[ZH2] 2168: kwsØ4:    mov bx,7
[HP6] 2169: kwsØ5:    cmp byte ptr[ds:si+bx+3Ø],BauerS
[B61] 217Ø:           je kwsØ6
[NV4] 2171:           add word ptr[ds:schutzR],5Ø
[AU4] 2172:           cmp byte ptr[ds:SZaBS+bx],Ø
[NH1] 2173:           jne kwsØ6
[NV4] 2174:           add word ptr[ds:schutzR],5Ø
[582] 2175: kwsØ6:    inc bx
[2A1] 2176:           cmp bx,9
[Z61] 2177:           jl kwsØ5
[PC5] 2178:           ; Rochade auf rechtem Flügel mgl?
[AB6] 2179:       cmp byte ptr[ds:si.RochadeErfolgt],Ø
[RH1] 218Ø:           jne kwsØ7
[NV4] 2181:           add word ptr[ds:schutzR],5Ø
[AE6] 2182:         cmp byte ptr[ds:si.klRochSmgl],true
[C61] 2183:           je kwsØ7
[DX4] 2184:           add word ptr[ds:schutzR],15Ø
[ØØØ] 2185:
[P45] 2186:           ; je nachdem, ob rochiert wurde,
[BS4] 2187:           ; den Flügel mit dem besseren
[415] 2188:           ; Gefährdungswert aussuchen
[Ll1] 2189: kwsØ7:
[AB6] 219Ø:       cmp byte ptr[ds:si.RochadeErfolgt],Ø
[TH1] 2191:           jne kwsØ8
[N35] 2192:           mov bx,word ptr[ds:schutzr]
[H15] 2193:           cmp bx,word ptr[ds:schutzl]
[Y61] 2194:           jl kwsl Ø
[335] 2195:           mov bx,word ptr[ds:schutzl]
[LK1] 2196:           jmp kwsl Ø
[ØØØ] 2197:
[N11] 2198: kwsØ8:
[4A6] 2199:       cmp [ds:si.RochadeErfolgt],klRochade
[VH1] 22ØØ:           jne kwsØ9
[N35] 2201:           mov bx,word ptr[ds:schutzR]
[LK1] 2202:           jmp kwsl Ø
[U96] 2203: kwsØ9:    mov bx,word ptr[ds:schutzL]
[ØØØ] 2204:
[AS5] 2205:           ; gefundene Summe mit gegnerischem
[N93] 2206:           ; Figurenwert mult.
[GL2] 2207: kwsl Ø:   add ax,bx
[534] 2208:           dec byte ptr[ds:SumFW]
[B64] 2209:           mul byte ptr[ds:SumFW]
```

```
[3PØ] 221Ø:           ret
[ØØØ] 2211:
[6E3] 2212:           ; Endspielbewertung
[W23] 2213:           ; Zentralisierung
[BN2] 2214: kwsll:    xor bh,bh
[E45] 2215:           mov bl,byte ptr[ds:si.KoenPS]
[V91] 2216:           shl bx,1
[RØ5] 2217:           mov ax,word ptr[ds:ZAbst+bx]
[T91] 2218:           shl ax,1
[T91] 2219:           shl ax,1
[ØØØ] 222Ø:
[G45] 2221:           mov cl,byte ptr[ds:si.KoenPS]
[6F3] 2222:           call BauerKoenigAbst
[3PØ] 2223:           ret
[YG3] 2224: KoenigWertSchw ENDP
[ØØØ] 2225:
[PF3] 2226:           ; ******************************
[3G5] 2227:           ;       Routine zur Mattbewertung,
[MN4] 2228:           ;    ax:=ax+16*(14-Abstand cl-ch)
[PF3] 2229:           ; ******************************
[ØØØ] 223Ø:
[ØY2] 2231: AbstandWert PROC
[PC1] 2232:           push ax
[AØ2] 2233:           call Abstand
[H11] 2234:           neg ax
[HA1] 2235:           add ax,14
[491] 2236:           sal ax,1
[491] 2237:           sal ax,1
[491] 2238:           sal ax,1
[491] 2239:           sal ax,1
[V31] 224Ø:           pop cx
[BD1] 2241:           add ax,cx
[3PØ] 2242:           ret
[ZV2] 2243: AbstandWert ENDP
[ØØØ] 2244:
[PF3] 2245:           ; ******************************
[XG6] 2246:           ; Wert für:weiß setzt schwarz matt
[PF3] 2247:           ; ******************************
[ØØØ] 2248:
[D83] 2249: MattWertWeiss PROC
[W36] 225Ø:           ; Zentralisierung des schw. Königs
[3F1] 2251:           xor bh,bh
[E45] 2252:           mov bl,byte ptr[ds:si.KoenPS]
[V91] 2253:           shl bx,1
[RØ5] 2254:           mov ax,word ptr[ds:ZAbst+bx]
[T91] 2255:           shl ax,1
[T91] 2256:           shl ax,1
[T91] 2257:           shl ax,1
[E73] 2258:           ; Abstand der Könige
[G45] 2259:           mov cl,byte ptr[ds:si.KoenPS]
[845] 226Ø:           mov ch,byte ptr[ds:si.KoenPW]
[3T2] 2261:           call AbstandWert
[BR3] 2262:           ; Abstand zu Springern
[3F1] 2263:           xor bh,bh
[LF4] 2264:           mov bl,byte ptr[ds:PosFW]
[AB7] 2265: mwwØØ:    cmp byte ptr[ds:PosFW+bx],SpringerW
[ZH1] 2266:           jne mwwØ1
[G45] 2267:           mov cl,byte ptr[ds:si.KoenPS]
[8Ø5] 2268:           mov ch,byte ptr[ds:PosFW+bx]
[3T2] 2269:           call AbstandWert
[A72] 227Ø: mwwØ1:    dec bx
[HM1] 2271:           jnz mwwØØ
[GA1] 2272:           mov cl,7
[6F1] 2273:           sar ax,cl
[Ø75] 2274:           add ax,word ptr[ds:si.MatDiff]
[3PØ] 2275:           ret
[E73] 2276: MattWertWeiss ENDP
[ØØØ] 2277:
[PF3] 2278:           ; ******************************
[176] 2279:           ; Wert für:schwarz setzt weiß matt
[PF3] 228Ø:           ; ******************************
[ØØØ] 2281:
[D23] 2282: MattWertSchw PROC
[HK6] 2283:           ;Zentralisierung des weißen Königs
[3F1] 2284:           xor bh,bh
[K45] 2285:           mov bl,byte ptr[ds:si.KoenPW]
[V91] 2286:           shl bx,1
[RØ5] 2287:           mov ax,word ptr[ds:ZAbst+bx]
[T91] 2288:           shl ax,1
[T91] 2289:           shl ax,1
[T91] 229Ø:           shl ax,1
[E73] 2291:           ; Abstand der Könige
[G45] 2292:           mov cl,byte ptr[ds:si.KoenPS]
[845] 2293:           mov ch,byte ptr[ds:si.KoenPW]
[3T2] 2294:           call AbstandWert
[BR3] 2295:           ; Abstand zu Springern
[3F1] 2296:           xor bh,bh
[FF4] 2297:           mov bl,byte ptr[ds:PosFS]
[AA7] 2298: mwsØØ:    cmp byte ptr[ds:PosFS+bx],SpringerS
[FH1] 2299:           jne mwsØ1
[M45] 23ØØ:           mov cl,byte ptr[ds:si.KoenPW]
[4Ø5] 23Ø1:           mov ch,byte ptr[ds:PosFS+bx]
[3T2] 23Ø2:           call AbstandWert
[U62] 23Ø3: mwsØ1:    dec bx
[1M1] 23Ø4:           jnz mwsØØ
[GA1] 23Ø5:           mov cl,7
[6F1] 23Ø6:           sar ax,cl
[H11] 23Ø7:           neg ax
[Ø75] 23Ø8:           add ax,word ptr[ds:si.MatDiff]
[3PØ] 23Ø9:           ret
[B13] 231Ø: MattWertSchw ENDP
[ØØØ] 2311:
[PF3] 2312:           ; ******************************
[RB5] 2313:           ;    kann gegner. König geschlagen
[WZ5] 2314:           ;    werden? Überprüfung anhand der
[6D5] 2315:           ;  Zuglisten (kein Check ob Listen
[755] 2316:           ;  leer...) Resultat in AX (Ø wenn
[CH5] 2317:           ; nicht im Schach, sonst Bewertung
[EN3] 2318:           ;                    des Zuges)
[PF3] 2319:           ; ******************************
[ØØØ] 232Ø:
[YK3] 2321: KoenigSchlagbar PROC
[YT5] 2322:           cmp byte ptr[ds:si.AmZug],weiss
[K91] 2323:           jne ksØ1
[K45] 2324:           mov dl,byte ptr[ds:si.KoenPS]
[AL4] 2325:           mov ax,3ØØØØ+size spielstand
[CF1] 2326:           sub ax,si
[WA1] 2327:           jmp ksØ2
[ØØØ] 2328:
[A46] 2329: ksØ1:     mov dl,byte ptr[ds:si.KoenPW]
[NL4] 233Ø:           mov ax,-3ØØØØ-size spielstand
[XC1] 2331:           add ax,si
[ØØØ] 2332:
[8H2] 2333: ksØ2:     xor bx,bx
[4W6] 2334: ksØ3:   cmp byte ptr[ds:si.list+bx.nach],dl
[M11] 2335:           je ksØ4
[GN2] 2336:           add bx,size zug
[A15] 2337:           cmp bx,word ptr[ds:si.anzahl]
[521] 2338:           jl ksØ3
[N22] 2339:           ; Entwarnung
[PLØ] 234Ø:           clc
[3PØ] 2341:           ret
[ØØØ] 2342:
[5N1] 2343: ksØ4:     stc
[3PØ] 2344:           ret
[XG3] 2345: KoenigSchlagbar ENDP
[ØØØ] 2346:
[PF3] 2347:           ; ******************************
[645] 2348:           ;     ist eigener König im Schach?
[PF3] 2349:           ; ******************************
[ØØØ] 235Ø:
[SA3] 2351: KoenigImSchach PROC
[3F1] 2352:           xor bh,bh
[795] 2353:           mov byte ptr[ds:schach],false
[YT5] 2354:           cmp byte ptr[ds:si.AmZug],weiss
[3G1] 2355:           jne kisØ2
[ØØØ] 2356:
```

```
[XC3] 2357:           ; Routine für weiss
[N24] 2358:           ; kann ein Bauer schlagen?
[NL4] 2359:           mov ax,-3ØØØØ-size spielstand
[XC1] 236Ø:           add ax,si
[PC1] 2361:           push ax
[K45] 2362:           mov bl,byte ptr[ds:si.KoenPW]
[NF1] 2363:           mov dl,bl
[WA6] 2364:       cmp byte ptr[ds:si.feld+bx-11],BauerS
[LA2] 2365:           je kis_schach2
[786] 2366:        cmp byte ptr[ds:si.feld+bx-9],BauerS
[LA2] 2367:           je kis_schach2
[PA3] 2368:           ; Figuren festlegen
[XØ2] 2369:           mov ah,DameS
[SS2] 237Ø:           mov al,LaeuferS
[PC1] 2371:           push ax
[G42] 2372:           mov al,TurmS
[CV2] 2373:           mov cl,SpringerS
[EH1] 2374:           jmp kisØ3
[ØØØ] 2375:
[LK3] 2376:           ; Routine für schwarz
[9S5] 2377: kisØ2:    mov ax,3ØØØØ+size spielstand
[CF1] 2378:           sub ax,si
[PC1] 2379:           push ax
[E45] 238Ø:           mov bl,byte ptr[ds:si.KoenPS]
[NF1] 2381:           mov dl,bl
[WA6] 2382:       cmp byte ptr[ds:si.feld+bx+11],BauerW
[LA2] 2383:           je kis_schach2
[386] 2384:        cmp byte ptr[ds:si.feld+bx+9],BauerW
[LA2] 2385:           je kis_schach2
[PA3] 2386:           ; Figuren festlegen
[112] 2387:           mov ah,DameW
[8T2] 2388:           mov al,LaeuferW
[PC1] 2389:           push ax
[M42] 239Ø:           mov al,TurmW
[GV2] 2391:           mov cl,SpringerW
[EH1] 2392:           jmp kisØ3
[ØØØ] 2393:
[822] 2394: kis_schach1:
[X31] 2395:           pop dx
[C22] 2396: kis_schach2:
[285] 2397:           mov byte ptr[ds:schach],true
[R31] 2398:           pop ax
[ENØ] 2399:           stc
[3PØ] 24ØØ:           ret
[ØØØ] 2401:
[1F3] 2402:           ; Springer-Bedrohung
[2S6] 2403: kisØ3:    cmp byte ptr[ds:si.feld+bx+21],cl
[GA2] 2404:           je kis_schach1
[9L5] 2405:           cmp byte ptr[ds:si.feld+bx-21],cl
[GA2] 2406:           je kis_schach1
[SL5] 2407:           cmp byte ptr[ds:si.feld+bx-19],cl
[GA2] 2408:           je kis_schach1
[GL5] 2409:           cmp byte ptr[ds:si.feld+bx+19],cl
[GA2] 241Ø:           je kis_schach1
[AL5] 2411:           cmp byte ptr[ds:si.feld+bx-12],cl
[GA2] 2412:           je kis_schach1
[2L5] 2413:           cmp byte ptr[ds:si.feld+bx+12],cl
[GA2] 2414:           je kis_schach1
[1G5] 2415:           cmp byte ptr[ds:si.feld+bx-8],cl
[GA2] 2416:           je kis_schach1
[TF5] 2417:           cmp byte ptr[ds:si.feld+bx+8],cl
[GA2] 2418:           je kis_schach1
[ØØØ] 2419:
[Z55] 242Ø:           ; Damen/Läufer/Turm-Bedrohung
[ØD1] 2421:           mov dh,1Ø
[MR1] 2422:           call suche
[AA2] 2423:           jc kis_schach1
[EH1] 2424:           mov dh,-1Ø
[MR1] 2425:           call suche
[AA2] 2426:           jc kis_schach1
[ØA1] 2427:           mov dh,1
[MR1] 2428:           call suche
[AA2] 2429:           jc kis_schach1
[YC1] 243Ø:           mov dh,-1
[MR1] 2431:           call suche
[AA2] 2432:           jc kis_schach1
[R31] 2433:           pop ax
[8A1] 2434:           mov dh,9
[MR1] 2435:           call suche
[DA2] 2436:           jc kis_schach2
[ED1] 2437:           mov dh,-9
[MR1] 2438:           call suche
[DA2] 2439:           jc kis_schach2
[2D1] 244Ø:           mov dh,11
[MR1] 2441:           call suche
[DA2] 2442:           jc kis_schach2
[HH1] 2443:           mov dh,-11
[MR1] 2444:           call suche
[DA2] 2445:           jc kis_schach2
[R31] 2446:           pop ax
[PLØ] 2447:           clc
[3PØ] 2448:           ret
[SA3] 2449: KoenigImSchach ENDP
[ØØØ] 245Ø:
[PF3] 2451:           ; ******************************
[NØ6] 2452:           ;  Hilfsroutine für KoenigImSchach
[C35] 2453:           ;    - sucht von dl in richtung dh
[VF4] 2454:           ;              nach Figur ah oder al
[PF3] 2455:           ; ******************************
[ØØØ] 2456:
[WP1] 2457: suche PROC
[LF1] 2458:           mov bl,dl
[8M2] 2459: suche1:   add bl,dh
[CR5] 246Ø:           cmp byte ptr[ds:si.feld+bx],leer
[GB1] 2461:           je suche1
[TB5] 2462:           cmp byte ptr[ds:si.feld+bx],ah
[KB1] 2463:           je suche2
[5C5] 2464:           cmp byte ptr[ds:si.feld+bx],al
[KB1] 2465:           je suche2
[743] 2466:           clc  ; nicht bedroht
[3PØ] 2467:           ret
[ØØØ] 2468:
[VG3] 2469: suche2:   stc  ; bedroht
[3PØ] 247Ø:           ret
[XN1] 2471: suche ENDP
[ØØØ] 2472:
[PF3] 2473:           ; ******************************
[YR4] 2474:           ;              Bewertungsfunktion
[PF3] 2475:           ; ******************************
[ØØØ] 2476:
[PF2] 2477: BrettWert PROC
[SZ5] 2478:           cmp byte ptr[ds:si.StellungTyp],4
[G21] 2479:           jl bwØ
[ØØØ] 248Ø:
[S93] 2481:           ; normale Bewertung
[7C5] 2482:           ; CutOff-Vorhersage bei zu hoher
[964] 2483:           ; Differenz Materialwert
[HF2] 2484:           ; zu alpha/beta
[1B5] 2485:           mov ax,word ptr[ds:si.MatDiff]
[CH1] 2486:           sub ax,12Ø
[2V4] 2487:           cmp ax,word ptr[ds:si.alpha]
[W11] 2488:           jl bwØ1
[3PØ] 2489:           ret
[GE2] 249Ø: bwØ1:     add ax,24Ø
[5V4] 2491:           cmp ax,word ptr[ds:si.beta]
[H11] 2492:           jg bwØ2
[3PØ] 2493:           ret
[ØØØ] 2494:
[FF3] 2495:           ; nur Abschätzung bei
[YS2] 2496:           ; Schlagzugsuche
[GA6] 2497: bwØ2:     mov ax,word ptr[ds:si.MatDiff]
[VM5] 2498:           add ax,word ptr[ds:si.ruhewert]
[ØF2] 2499:           cmp si,horizon
[891] 25ØØ:           jle bwØ3
[3PØ] 251Ø:           ret
[ØØØ] 2512:
[BØ3] 2513:           ; positionelle und
```

```
[XT3] 25Ø4:          ; materielle Bewertung
[D43] 25Ø5: bwØ3:    call PosWert
[ØT5] 25Ø6:          mov word ptr[ds:si.ruhewert],ax
[ØT5] 25Ø7:          add ax,word ptr[ds:si.MatDiff]
[3PØ] 25Ø8:          ret
[ØØØ] 25Ø9:
[6E3] 251Ø:          ; Endspielbewertung
[ZZ6] 2511: bw8Ø:    cmp byte ptr[ds:si.StellungTyp],1
[CA1] 2512:          jne bw81
[Ø83] 2513:          call MattWertWeiss
[3PØ] 2514:          ret
[ØØØ] 2515:
[4Ø7] 2516: bw81:    cmp byte ptr[ds:si.StellungTyp],3
[DA1] 2517:          jne bw82
[U22] 2518:          call MattWertSchw
[3PØ] 2519:          ret
[ØØØ] 252Ø:
[AM3] 2521: bw82:    xor ax,ax  ; patt
[3PØ] 2522:          ret
[RE2] 2523: BrettWert ENDP
[ØØØ] 2524:
[PF3] 2525:          ; ********************************
[R24] 2526:          ;          ermittelt positionelle
[G54] 2527:          ;                Bewertung in ax
[PF3] 2528:          ; ********************************
[ØØØ] 2529:
[N92] 253Ø: PosWert PROC
[GC1] 2531:          push si
[N32] 2532:          call Analyse
[YK3] 2533:          call BauernWertWeiss
[PC1] 2534:          push ax
[GG3] 2535:          call BauernWertSchw
[PC1] 2536:          push ax
[VM3] 2537:          call FigurWertWeiss
[PC1] 2538:          push ax
[7A3] 2539:          call FigurWertSchw
[PC1] 254Ø:          push ax
[WL3] 2541:          call KoenigWertWeiss
[PC1] 2542:          push ax
[EH3] 2543:          call KoenigWertSchw
[ØØØ] 2544:
[CØ2] 2545:          ; summieren
[T31] 2546:          pop bx
[AD1] 2547:          add ax,bx
[T31] 2548:          pop bx
[AD1] 2549:          add ax,bx
[T31] 255Ø:          pop bx
[AD1] 2551:          add ax,bx
[T31] 2552:          pop bx
[AD1] 2553:          add ax,bx
[T31] 2554:          pop bx
[AD1] 2555:          add ax,bx
[DB6] 2556:        add ax,word ptr[ds:si.BedrohungWert]
[L72] 2557:          sar ax,1 ; ax/2
[V72] 2558:          sar ax,1 ; ax/4
[B82] 2559:          sar ax,1 ; ax/8
[ØX4] 256Ø:          ; Strafpunkte für leichte Fig.
[R53] 2561:          ; auf der Grundlinie
[3F1] 2562:          xor bh,bh
[VE4] 2563:          mov bl,byte ptr[ds:grlfw]
[TF1] 2564:          sub ax,bx
[RE4] 2565:          mov bl,byte ptr[ds:grlfs]
[AD1] 2566:          add ax,bx
[592] 2567:          sar ax,1 ; ax/16
[992] 2568:          sar ax,1 ; ax/32
[R92] 2569:          sar ax,1 ; ax/64
[HC2] 257Ø:          sar ax,1 ; ax/128
[E31] 2571:          pop si
[3PØ] 2572:          ret
[M72] 2573: PosWert ENDP
[ØØØ] 2574:
[PF3] 2575:          ; ********************************
[MB5] 2576:          ;       Zug/Save-Listen Verwaltung
[PF3] 2577:          ; ********************************
[ØØØ] 2578:
[Y94] 2579: ; ****************************************
[N87] 258Ø: ;    löst einen Zug aus einer Liste heraus
[UZ6] 2581: ;  löst den Zug DX (von/nach) aus SAVELIST
[UT6] 2582: ;  heraus, falls Zug vorhanden war enthält
[NV5] 2583: ;              AX dessen letzte Bewertung,
[2R5] 2584: ;                 sonst AX alpha/betastart
[Y94] 2585: ; ****************************************
[ØØØ] 2586:
[W33] 2587: Aussortieren PROC
[FV3] 2588:          ; ist Liste schon leer?
[SN5] 2589:          cmp word ptr[ds:si.saveanzahl],Ø
[225] 259Ø:          ; ja, also ex. Zug nicht in Liste!
[U51] 2591:          je ausØ3
[EA6] 2592:        cmp word ptr[ds:si.savestart],ØffffH
[T15] 2593:          je ausØ3 ; ja, schon "leergesucht"
[T24] 2594:          ; ist Zug der Anfang der Liste?
[UT5] 2595:          mov bx,word ptr[ds:si.savestart]
[RØ6] 2596:          cmp dx,word ptr[ds:si.savelist+bx]
[T51] 2597:          je ausØ2
[AX4] 2598:          ; nein, dann Vorgänger finden
[UP2] 2599: ausØØ:  mov cx,bx
[N27] 26ØØ:     mov bx,word ptr[ds:si.savelist+bx.next]
[542] 26Ø1:          cmp bx,ØffffH
[1B5] 26Ø2:          ; Zugliste zuende, nicht gefunden
[U51] 26Ø3:          je ausØ3
[RØ6] 26Ø4:          cmp dx,word ptr[ds:si.savelist+bx]
[GX4] 26Ø5:          je ausØ1  ; Zug gefunden bei BX
[SH1] 26Ø6:          jmp ausØØ
[ØØØ] 26Ø7:
[E53] 26Ø8:          ; in Liste gefunden
[CK2] 26Ø9: ausØ1:   push dx
[R27] 261Ø:    mov ax,word ptr[ds:si.savelist+bx.value]
[YH3] 2611:          ; in dx Nachfolger des
[2G4] 2612:          ; auszusortierenden Zuges
[T27] 2613:     mov dx,word ptr[ds:si.savelist+bx.next]
[EH1] 2614:          mov bx,cx
[1N3] 2615:          ; in Vorgänger diesen
[5P3] 2616:          ; Nachfolger eintragen
[B67] 2617:     mov word ptr[ds:si.savelist+bx.next],dx
[X31] 2618:          pop dx
[ØK1] 2619:          jmp ausØ4
[ØØØ] 262Ø:
[EL2] 2621:          ; Zug ist Erster
[AX3] 2622:          ; Nachfolger wird Anfang
[FØ1] 2623: ausØ2:
[R27] 2624:    mov ax,word ptr[ds:si.savelist+bx.value]
[DB6] 2625:     mov cx,word ptr[ds:si.savelist+bx.next]
[ØT5] 2626:          mov word ptr[ds:si.savestart],cx
[ØK1] 2627:          jmp ausØ4
[ØØØ] 2628:
[WD3] 2629:          ; Zug nicht gefunden,
[2Y4] 263Ø:          ; also Startwert zurückgeben
[FZ6] 2631: ausØ3:   cmp byte ptr[ds:si.AmZug],weiss
[8B1] 2632:          je ausØ3w
[N94] 2633:          ; schwarz minimiert alpha
[3Ø3] 2634:          mov ax,alphastart
[ØK1] 2635:          jmp ausØ4
[1V3] 2636:          ; weiß maximiert beta
[F64] 2637: ausØ3w: mov ax,betastart
[ØØØ] 2638:
[PV1] 2639: ausØ4:  ret
[U23] 264Ø: Aussortieren ENDP
[ØØØ] 2641:
[Y94] 2642: ; ****************************************
[ØX6] 2643: ;   Einsortieren eines Zuges in eine Liste
[E84] 2644: ;    DX : Position des Zuges
[ØT6] 2645: ;    AX : Wert des Zuges, Sortierkriterium
[9E6] 2646: ;    DI : zeigt auf die Liste (sort/save)
[2K5] 2647: ;    BX : enthält den Listenanfang
[RX6] 2648: ;  je nachdem, ob schwarz oder weiß am Zug
[S17] 2649: ;  ist erfolgt unterschiedliche Sortierung
[MD6] 265Ø: ;     Es wird in BX der Listenanfang (eve.
[6P5] 2651: ;              jetzt neu!) zurückgegeben.
[W46] 2652: ;          WICHTIG: bei gleicher Bewertung
[ME6] 2653: ;          zweier Züge, hintendran anfügen
[Y94] 2654: ; ****************************************
[ØØØ] 2655:
[C23] 2656: Einsortieren PROC
[EV3] 2657:          ; Liste noch ganz leer?
[M71] 2658:          or dx,dx
[N41] 2659:          je einØ1
[ØØØ] 266Ø:
[335] 2661:          push bx ; Listenanfang sichern
[1B6] 2662:          ;ist es schwarze oder weiße Liste?
[MZ5] 2663:          cmp byte ptr[ds:si.AmZug],schwarz
[V91] 2664:          je einØ1s
[ØØØ] 2665:
[Z36] 2666:          ;in "weißer" Liste Position suchen
[WG5] 2667:          cmp ax,word ptr[ds:di+bx.value]
[2P4] 2668:          ; Position vor Listenanfang
[Y41] 2669:          jg einØ4
[S33] 267Ø: einØ1wl: mov cx,bx
[695] 2671:          mov bx,word ptr[ds:di+bx.next]
[8Ø4] 2672:          ; Position am Listenende
[542] 2673:          cmp bx,ØffffH
[R41] 2674:          je einØ3
[WG5] 2675:          cmp ax,word ptr[ds:di+bx.value]
[BL4] 2676:          jg einØ2  ; Position gefunden
[Z12] 2677:          jmp einØ1wl
[ØØØ] 2678:
[FC5] 2679:          ;in "schw." Liste Position suchen
[CU6] 268Ø: einØ1s: cmp ax,word ptr[ds:di+bx.value]
[NB4] 2681:          ; Position am Listenanfang
[D51] 2682:          jl einØ4
[C33] 2683: einØ1sl: mov cx,bx
[695] 2684:          mov bx,word ptr[ds:di+bx.next]
[542] 2685:          cmp bx,ØffffH
[R41] 2686:          je einØ3
[WG5] 2687:          cmp ax,word ptr[ds:di+bx.value]
[UL4] 2688:          jl einØ2  ; Position gefunden
[L12] 2689:          jmp einØ1sl
[ØØØ] 269Ø:
[4A5] 2691:          ; allererster Eintrag der Liste
[LV6] 2692: einØ1:   xor bx,bx ; Listenanfang ganz vorn
[826] 2693:          mov word ptr[ds:di+bx.next],ØffffH
[7N5] 2694:          mov word ptr[ds:di+bx.value],ax
[3PØ] 2695:          ret
[ØØØ] 2696:
[655] 2697:          ; mitten in Liste einsortieren
[DN2] 2698: einØ2:   mov bx,cx
[895] 2699:          mov cx,word ptr[ds:di+bx.next]
[4F5] 27ØØ:          mov word ptr[ds:di+bx.next],dx
[FH1] 27Ø1:          mov bx,dx
[2F5] 27Ø2:          mov word ptr[ds:di+bx.next],cx
[7N5] 27Ø3:          mov word ptr[ds:di+bx.value],ax
[PP5] 27Ø4:          pop bx ; Listenanfang unverändert
[3PØ] 27Ø5:          ret
[ØØØ] 27Ø6:
[HU4] 27Ø7:          ; am Listenende einsortieren
[FN2] 27Ø8: einØ3:   mov bx,cx
[4F5] 27Ø9:          mov word ptr[ds:di+bx.next],dx
[FH1] 271Ø:          mov bx,dx
[826] 2711:          mov word ptr[ds:di+bx.next],ØffffH
[7N5] 2712:          mov word ptr[ds:di+bx.value],ax
[PP5] 2713:          pop bx ; Listenanfang unverändert
[3PØ] 2714:          ret
[ØØØ] 2715:
[4E5] 2716:          ; vor Listenanfang einsortieren
[5W2] 2717: einØ4:   xchg bx,dx
[4F5] 2718:          mov word ptr[ds:di+bx.next],dx
[7N5] 2719:          mov word ptr[ds:di+bx.value],ax
[WM4] 272Ø:          pop dx ; neuer Listenanfang
[3PØ] 2721:          ret
[A13] 2722: Einsortieren ENDP
[ØØØ] 2723:
[PF3] 2724:          ; ********************************
[5Z4] 2725:          ;            erhöht die Stacktiefe
[PF3] 2726:          ; ********************************
[ØØØ] 2727:
[B92] 2728: IncDepth PROC
[7H1] 2729:          mov cx,ds
[5H1] 273Ø:          mov es,cx
[US2] 2731:          mov si,aktStack
[WR2] 2732:          mov di,aktStack
[1U3] 2733:          add di,SIZE Spielstand
[DZ3] 2734:          mov cx,SIZE Spielstand
[994] 2735:          ; die Savelist & SaveStart
[FY2] 2736:          ; nicht löschen!
[G44] 2737:          sub cx,(SIZE savelist + 4)
[ØA1] 2738:          sar cx,1
[LN1] 2739:          rep movsw
[6V4] 274Ø:          add aktstack,SIZE Spielstand
[US2] 2741:          mov si,aktstack
[EL4] 2742:          not byte ptr[ds:si.AmZug]
[3PØ] 2743:          ret
[982] 2744: IncDepth ENDP
[ØØØ] 2745:
[PF3] 2746:          ; ********************************
[DX4] 2747:          ;           mindert die Stacktiefe
[PF3] 2748:          ; ********************************
[ØØØ] 2749:
[682] 275Ø: DecDepth PROC
[NX4] 2751:          sub aktStack,SIZE Spielstand
[US2] 2752:          mov si,aktStack
[3PØ] 2753:          ret
[472] 2754: DecDepth ENDP
[ØØØ] 2755:
[PF3] 2756:          ; ********************************
[K65] 2757:          ;   Weiss an der Reihe (maximiert)
[PF3] 2758:          ; ********************************
[ØØØ] 2759:
[GC2] 276Ø: WeissZug PROC
[2V4] 2761:          cmp ax,word ptr[ds:si.alpha]
[TA1] 2762:          jle wzØ1
[ZR5] 2763:          mov byte ptr[ds:si.cutoff],true
[ØØØ] 2764:
[6W5] 2765: wzØ1:    cmp ax,word ptr[ds:si.beta]
[UA1] 2766:          jle wzØ2
[XY4] 2767:          mov word ptr[ds:si.beta],ax
[P52] 2768:          cmp si,Depth1
[ØB1] 2769:          jne wzØ2
[TR4] 277Ø:          cmp byte ptr[ds:timeout],Ø
[MD1] 2771:          jnz wzØ2
[925] 2772:          mov cx,word ptr[ds:si.zugNr]
[E45] 2773:          mov word ptr[ds:ZugWahl],cx
[ØØØ] 2774:
[5R1] 2775: wzØ2:    ret
[EB2] 2776: WeissZug ENDP
[ØØØ] 2777:
[PF3] 2778:          ; ********************************
[PD5] 2779:          ; Schwarz an der Reihe (minimiert)
[PF3] 278Ø:          ; ********************************
[ØØØ] 2781:
[CS2] 2782: SchwarzZug PROC
[5V4] 2783:          cmp ax,word ptr[ds:si.beta]
[2A1] 2784:          jge szØ1
[ZR5] 2785:          mov byte ptr[ds:si.cutoff],true
[ØØØ] 2786:
[VV5] 2787: szØ1:    cmp ax,word ptr[ds:si.alpha]
[EB1] 2788:          jge sz99
[XZ4] 2789:          mov word ptr[ds:si.alpha],ax
[P52] 279Ø:          cmp si,Depth1
[3C1] 2791:          jne sz99
[TR4] 2792:          cmp byte ptr[ds:timeout],Ø
[RE1] 2793:          jnz sz99
[925] 2794:          mov cx,word ptr[ds:si.zugnr]
[E45] 2795:          mov word ptr[ds:ZugWahl],cx
[ØØØ] 2796:
[8S1] 2797: sz99:    ret
[CS2] 2798: SchwarzZug ENDP
[ØØØ] 2799:
[PF3] 28ØØ:          ; ********************************
[SG4] 28Ø1:          ;                 restauriert Brett
[PF3] 28Ø2:          ; ********************************
[ØØØ] 28Ø3:
[G72] 28Ø4: Restore PROC
[3B1] 28Ø5:          push di
[GC1] 28Ø6:          push si
[WC1] 28Ø7:          push cx
[US2] 28Ø8:          mov si,aktStack
[EX3] 28Ø9:          sub si,SIZE Spielstand
[WR2] 281Ø:          mov di,aktStack
[DH3] 2811:          ; Brett restaurieren
[K25] 2812:          mov cx,5Ø ; Brett in Bytes 1Ø*1Ø/2
[LN1] 2813:          rep movsw
[1A4] 2814:          ; Statistik restaurieren
[US2] 2815:          mov si,aktStack
[2Y6] 2816:    add si,offset EndSpielS-SIZE Spielstand
[WR2] 2817:          mov di,aktStack
[XZ3] 2818:          add di,offset EndSpielS
[PB1] 2819:          mov cx,9
[LN1] 282Ø:          rep movsw
[V31] 2821:          pop cx
[E31] 2822:          pop si
[G21] 2823:          pop di
[3PØ] 2824:          ret
[F52] 2825: Restore ENDP
[ØØØ] 2826:
[PF3] 2827:          ; ********************************
[CD5] 2828:          ; Suche,ermittelt den "besten" Zug
[PF3] 2829:          ; ********************************
[ØØØ] 283Ø:
[HT2] 2831: ZugAuswahl PROC
[W85] 2832:          ; weitere Vertiefung möglich?
[C25] 2833:          cmp si,word ptr[ds:OutZone]
[811] 2834:          jl za Ø1
[SG3] 2835: zaØØ:    call BrettWert
[3PØ] 2836:          ret
[ØØØ] 2837:
[K53] 2838: za Ø1:   call IncDepth
[HT5] 2839:          mov byte ptr[ds:si.cutoff],false
[ØØØ] 284Ø:
[MZ5] 2841:          cmp byte ptr[ds:si.AmZug],schwarz
[V81] 2842:          jne zaØ2
[3A3] 2843:          call ZugGenSchwarz
[7E6] 2844:        mov word ptr[ds:si.alpha],alphastart
[5A1] 2845:          jmp zaØ3
[VV3] 2846: zaØ2:    call ZugGenWeiss
[M46] 2847:          mov word ptr[ds:si.beta],betastart
[ØØØ] 2848:
[XA5] 2849:          ; überhaupt Züge vorhanden?
[BX5] 285Ø: zaØ3:    cmp word ptr[ds:si.anzahl],Ø
[AE1] 2851:          jne zaØ3w
[XH2] 2852:          call Brettwert
[352] 2853:          call DecDepth
[3PØ] 2854:          ret
[ØØØ] 2855:
[693] 2856:          ; illegale Stellung?
[CH4] 2857: zaØ3w:   call KoenigSchlagbar
[GF4] 2858:          ; legt auch Bewertung fest!
[4E1] 2859:          jnc zaØ3x
[352] 286Ø:          call DecDepth
[3PØ] 2861:          ret
[ØØØ] 2862:
[V84] 2863: zaØ3x:   ; noch volle Suche?
[ØF2] 2864:          cmp si,horizon
[WC1] 2865:          jle zaØ3c
[RW5] 2866:          ; nein, nur noch Schlagzüge werden
[UE5] 2867:          ; untersucht. Als Basis RUHEWERT
[VZ1] 2868:          ; ermitteln
[XH2] 2869:          call brettwert
[MZ5] 287Ø:          cmp byte ptr[ds:si.AmZug],schwarz
[YC1] 2871:          jne zaØ3a
[MT2] 2872:          call SchwarzZug
[9E1] 2873:          jmp zaØ3b
[BC3] 2874: zaØ3a:   call WeissZug
[CS6] 2875: zaØ3b:   cmp word ptr[ds:si.cutoff],true
[2D1] 2876:          jne zaØ3c
[352] 2877:          call DecDepth
[3PØ] 2878:          ret
[ØØØ] 2879:
[G36] 288Ø: zaØ3c:   mov bx,word ptr[ds:si.start]
[962] 2881:          ; zugschleife
[T96] 2882: zaØ4:    mov word ptr[ds:si.ZugNr],bx
[442] 2883:          call Restore
[2Z2] 2884:          call FuehreZugAus
[ØØØ] 2885:
[TC1] 2886:          push bx
[ØF2] 2887:          cmp si,horizon
[M64] 2888:          jl zaØ6  ; noch volle Suche
[EC6] 2889:        cmp word ptr[ds:si.list+bx.option],Ø
[Z81] 289Ø:          jne zaØ6
[NC3] 2891:          call KoenigImSchach
[511] 2892:          jc zaØ7
[XH2] 2893:          call BrettWert
[9A1] 2894:          jmp zaØ7
[XT3] 2895: zaØ6:    call ZugAuswahl
[ØØØ] 2896:
[V22] 2897: zaØ7:    pop bx
[FH1] 2898:          mov cx,bx
[MZ5] 2899:          cmp byte ptr[ds:si.AmZug],schwarz
[191] 29ØØ:          jne zaØ8
[MT2] 29Ø1:          call SchwarzZug
[BA1] 29Ø2:          jmp zaØ9
[M93] 29Ø3: zaØ8:    call WeissZug
[ØØØ] 29Ø4:
[TW4] 29Ø5:          ; bewerteten Zug in savelist
[LD2] 29Ø6:          ; einsortieren
[396] 29Ø7: zaØ9:    mov cx,word ptr[ds:si.list+bx]
[TC1] 29Ø8:          push bx
[4V5] 29Ø9:          mov bx,word ptr[ds:si.saveanzahl]
[G96] 291Ø:          mov word ptr[ds:si.savelist+bx],cx
[HH1] 2911:          mov dx,bx
[RF1] 2912:          mov di,si
[ET6] 2913:    add di,(size spielstand-size savelist-4)
[UT5] 2914:          mov bx,word ptr[ds:si.savestart]
[ZZ2] 2915:          call Einsortieren
[WS5] 2916:          mov word ptr[ds:si.savestart],bx
[T31] 2917:          pop bx
[HL5] 2918:          add word ptr[ds:si.saveanzahl],8
[ØØØ] 2919:
[2P4] 292Ø:          ; Ende der Suche durch CUTOFF?
[DP5] 2921:          cmp byte ptr[ds:si.cutoff],true
[KØ1] 2922:          je za1Ø
[XH2] 2923:          ; Ende der Liste?
[DA6] 2924:        mov bx,word ptr[ds:si.list+bx.next]
[542] 2925:          cmp bx,ØffffH
[KØ1] 2926:          je za1Ø
[AD2] 2927:          ; Zeit zu Ende?
[TR4] 2928:          cmp byte ptr[ds:timeout],Ø
[X81] 2929:          jne za1Ø
[WC2] 293Ø: za9Ø1:   jmp zaØ4
[ØØØ] 2931:
[LY6] 2932: za1Ø:    cmp byte ptr[ds:si.AmZug],schwarz
[PØ1] 2933:          je za11
[B35] 2934:          ; weiss am Zug: Bewertung=beta
[RW4] 2935:          mov ax,word ptr[ds:si.beta]
[FB1] 2936:          jmp za99
[NH5] 2937:          ; schwarz am Zug: Bewertung=alpha
[MV5] 2938: za11:    mov ax,word ptr[ds:si.alpha]
[ØØØ] 2939:
```

»Chessfx.asm« verspricht ungetrübten Schachgenuß.

```
[B53] 294Ø: za99:   call DecDepth
[3PØ] 2941:         ret
[HT2] 2942: ZugAuswahl ENDP
[ØØØ] 2943:
[PF3] 2944:         ; ********************************
[BW3] 2945:         ;                          Computerzug
[PF3] 2946:         ; ********************************
[ØØØ] 2947:
[7Ø3] 2948: ComputerZug PROC
[6A3] 2949:         mov bx,seg GameStack
[2H1] 295Ø:         mov ds,bx
[4H1] 2951:         mov es,bx
[CW2] 2952:         ; GameStack-Reset
[GH1] 2953:         xor ax,ax
[A62] 2954:         mov di,Depth1
[ØE4] 2955:         mov cx,Size Spielstand*7/2
[KN1] 2956:         rep stosw
[WC3] 2957:         ; Timer rücksetzen
[U91] 2958:         mov ah,1
[PH1] 2959:         xor cx,cx
[TH1] 296Ø:         xor dx,dx
[D81] 2961:         int 15h
[N74] 2962:         ; Timer-Intervall setzen
[ZL1] 2963:         mov ah,83h
[MF1] 2964:         xor al,al
[WN3] 2965:         mov bx,offset timeout
[33J] 2966:         mov cx,15*Denkzeit
[TH1] 2967:         xor dx,dx
[D81] 2968:         int 15h
[6W2] 2969:         ; Flags löschen
[BT4] 297Ø:         mov byte ptr[ds:timeout],Ø
[ØØØ] 2971:
[KB1] 2972:         mov cx,5
[H72] 2973:         mov bx,Depth Ø
[ØØØ] 2974:
[6N2] 2975:         ; Zug ermitteln
[2V4] 2976: czØØ:   add bx,size Spielstand
[YM2] 2977:         mov horizon,bx
[TC1] 2978:         push bx
[WC1] 2979:         push cx
[WU2] 298Ø:         call ZugAuswahl
[V31] 2981:         pop cx
[T31] 2982:         pop bx
[TR4] 2983:         cmp byte ptr[ds:timeout],Ø
[R91] 2984:         jne czØ1
[KG1] 2985:         loop czØØ
[ØØØ] 2986:
[RP4] 2987: czØ1:   ; Stellung analysieren
[CD3] 2988:         test ax,1ØØØØØØØØØØØØØØb
[CC1] 2989:         jnz czØ2
[2V4] 299Ø:         cmp ax,3ØØØØ-2*size spielstand
[B21] 2991:         jl czØ3
[7C5] 2992:         mov byte ptr[ds:gameovr],true
[LC5] 2993:         mov byte ptr[ds:sieger],weiss
[3B1] 2994:         jmp czØ4
[136] 2995: czØ2:   cmp ax,-3ØØØØ+2*size spielstand
[W11] 2996:         jg czØ3
[7C5] 2997:         mov byte ptr[ds:gameovr],true
[ZX5] 2998:         mov byte ptr[ds:sieger],schwarz
[3B1] 2999:         jmp czØ4
[G33] 3ØØØ: czØ3:   call Analyse
[MZ5] 3ØØ1:         cmp byte ptr[ds:si.StellungTyp],2
[U91] 3ØØ2:         jne czØ4
[7C5] 3ØØ3:         mov byte ptr[ds:gameovr],true
[ZK4] 3ØØ4:         mov byte ptr[ds:sieger],3
[ØØØ] 3ØØ5:
[BP2] 3ØØ6:         ; Fertigmeldung
[RM2] 3ØØ7: czØ4:   mov ah,ØEh
[LD1] 3ØØ8:         mov al,Ø7
[Y71] 3ØØ9:         int 1Øh
[ØØØ] 3Ø1Ø:
[RW2] 3Ø11:         ; Zug übergeben
[ZU3] 3Ø12:         add si,size spielstand
[W25] 3Ø13:         mov bx,word ptr[ds:ZugWahl]
[V95] 3Ø14:         mov ax,word ptr[ds:si.list+bx]
[7G6] 3Ø15:       mov dx,word ptr[ds:si.list+bx.option]
[L24] 3Ø16:         mov si,offset gamestack
[S15] 3Ø17:         mov word ptr[ds:si.list],ax
[A55] 3Ø18:         mov word ptr[ds:si.list+2],dx
[LH1] 3Ø19:         xor bx,bx
[ØØØ] 3Ø2Ø:
[3PØ] 3Ø21:         ret
[6Y2] 3Ø22: ComputerZug ENDP
[ØØØ] 3Ø23:
[PF3] 3Ø24:         ; ********************************
[KV4] 3Ø25:         ;          Spielverlauf-Schleife
[LG5] 3Ø26:         ;     liest Koordinate von Tastatur
[D65] 3Ø27:         ;       liefert in bx Brettposition
[PF3] 3Ø28:         ; ********************************
[ØØØ] 3Ø29:
[5Y2] 3Ø3Ø: KoordEingabe PROC
[B82] 3Ø31: keØ1:   mov ah,8
[381] 3Ø32:         int 21h
[Z84] 3Ø33:         ; Umwandlung klein->groß
[3C1] 3Ø34:         cmp al,27
[721] 3Ø35:         je ke99
[1G1] 3Ø36:         cmp al,'a'
[231] 3Ø37:         jl keØ1a
[UB1] 3Ø38:         sub al,32
[7H2] 3Ø39: keØ1a:  cmp al,'A'
[111] 3Ø4Ø:         jl keØ1
[FG1] 3Ø41:         cmp al,'H'
[KØ1] 3Ø42:         jg keØ1
[MF1] 3Ø43:         mov dl,al
[7C1] 3Ø44:         sub dl,64
[ØØØ] 3Ø45:
[C82] 3Ø46: keØ2:   mov ah,8
[381] 3Ø47:         int 21h
[3C1] 3Ø48:         cmp al,27
[721] 3Ø49:         je ke99
[1F1] 3Ø5Ø:         cmp al,'1'
[511] 3Ø51:         jl keØ2
[FF1] 3Ø52:         cmp al,'8'
[PØ1] 3Ø53:         jg keØ2
[ZG1] 3Ø54:         mov dh,':'
[PD1] 3Ø55:         sub dh,al
[ØØØ] 3Ø56:
[L22] 3Ø57:         ; 1Ø Mal Zeile
[F81] 3Ø58:         shl dh,1
[BF1] 3Ø59:         mov bl,dh
[F81] 3Ø6Ø:         shl dh,1
[F81] 3Ø61:         shl dh,1
[AB1] 3Ø62:         add bl,dh
[ØØØ] 3Ø63:
[922] 3Ø64:         ; plus Spalte
[KB1] 3Ø65:         add bl,dl
[ØØØ] 3Ø66:
[3F1] 3Ø67:         xor bh,bh
[3PØ] 3Ø68:         ret
[ØØØ] 3Ø69:
[NC3] 3Ø7Ø:         ; ESC-Abort Programm
[626] 3Ø71: ke99:   mov word ptr[ds:abort],true
[3PØ] 3Ø72:         ret
[3X2] 3Ø73: KoordEingabe ENDP
[ØØØ] 3Ø74:
[PF3] 3Ø75:         ; ********************************
[CK5] 3Ø76:         ;    liest Spielerzug von Tastatur
[PF3] 3Ø77:         ; ********************************
[ØØØ] 3Ø78:
[DN2] 3Ø79: ZugEingabe PROC
[XA4] 3Ø8Ø: ze_neu: call KoordEingabe
[VØ5] 3Ø81:         cmp byte ptr[ds:abort],true
[EØ2] 3Ø82:         je ze_abort
[YW4] 3Ø83:         cmp byte ptr[ds:si+bx],leer
[2W1] 3Ø84:         jle ze_neu
[FF1] 3Ø85:         mov al,bl
[ØØØ] 3Ø86:
[PC1] 3Ø87:         push ax
[ZV2] 3Ø88:         call KoordEingabe
[VØ5] 3Ø89:         cmp byte ptr[ds:abort],true
[EØ2] 3Ø9Ø:         je ze_abort
[R31] 3Ø91:         pop ax
[3F1] 3Ø92:         mov ah,bl
[ØØØ] 3Ø93:
[WZ5] 3Ø94:         mov bx,word ptr[ds:si.Anzahl]
[XY6] 3Ø95: ze_suche: cmp ax,word ptr[ds:si.list+bx]
[FZ1] 3Ø96:         je ze_found
[ZR2] 3Ø97:         sub bx,SIZE Zug
[M82] 3Ø98:         jns ze_suche
[ØØØ] 3Ø99:
[732] 31ØØ:         mov ax,msgEr
[XØ2] 31Ø1:         call Message
[ØØØ] 31Ø2:
[HX1] 31Ø3:         jmp ze_neu
[ØØØ] 31Ø4:
[5G5] 31Ø5:         ; eve. Bauernumwandlung gewählt?
[RX2] 31Ø6: ze_found: push bx
[3F1] 31Ø7:         xor bh,bh
[GF1] 31Ø8:         mov bl,al
[856] 31Ø9:         cmp byte ptr[ds:si.feld+bx],BauerW
[R62] 311Ø:         jne ze_quit
[AC1] 3111:         cmp bl,39
[HS1] 3112:         jg ze_quit
[ØØØ] 3113:
[RY2] 3114: ze_bauer: mov ah,8
[381] 3115:         int 21h
[1G1] 3116:         cmp al,'a'
[LC1] 3117:         jl ze_bØ1
[UB1] 3118:         sub al,32
[3U2] 3119: ze_bØ1: cmp al,'D'
[9R1] 312Ø:         jne ze_bØ2
[R22] 3121:         mov dx,DameW
[TT1] 3122:         jmp ze_b99
[4V2] 3123: ze_bØ2: cmp al,'S'
[U62] 3124:         jne ze_bauer
[PW2] 3125:         mov dx,SpringerW
[2E2] 3126: ze_b99: pop bx
[4M6] 3127:         mov word ptr[ds:si.list+bx.option],dx
[3PØ] 3128:         ret
[ØØØ] 3129:
[1W2] 313Ø: ze_quit: pop bx
[PA2] 3131: ze_abort: ret
[DN2] 3132: ZugEingabe ENDP
[ØØØ] 3133:
[PF3] 3134:         ; ********************************
[SM5] 3135:         ; abwechselnd Computer/Spieler Zug
[PF3] 3136:         ; ********************************
[ØØØ] 3137:
[GR1] 3138: Spiel PROC
[ST6] 3139: spØØ:   cmp byte ptr[ds:si.AmZug],weiss
[T91] 314Ø:         jne spØ1
[ØØØ] 3141:
[752] 3142:         mov ax,msgSZ
[XØ2] 3143:         call Message
[YW2] 3144:         call ZugGenWeiss
[5U3] 3145: spØ1a:  call ZugEingabe
[VØ5] 3146:         cmp byte ptr[ds:abort],true
[XØ2] 3147:         je sp_abort
[CC4] 3148:         ; Zug korrekt (König nicht
[WR3] 3149:         ; ins Schach gestellt?)
[TC1] 315Ø:         push bx
[P62] 3151:         call IncDepth
[2Z2] 3152:         call FuehreZugAus
[3A3] 3153:         call ZugGenSchwarz
[6E3] 3154:         call KoenigSchlagbar
[T31] 3155:         pop bx
[ND1] 3156:         jnc spØ1b
[352] 3157:         call DecDepth
[732] 3158:         mov ax,MsgEr
[XØ2] 3159:         call Message
[4F1] 316Ø:         jmp spØ1a
[ØØØ] 3161:
[W73] 3162: spØ1b:  call DecDepth
[Ø12] 3163:         call Anzeige
[2Z2] 3164:         call FuehreZugAus
[EL4] 3165:         not byte ptr[ds:si.AmZug]
[732] 3166:         mov ax,msgCZ
[XØ2] 3167:         call Message
[3B1] 3168:         jmp spØ2
[ØØØ] 3169:
[9L5] 317Ø: spØ1:   not byte ptr[ds:si.AmZug]
[SY2] 3171:         call ComputerZug
[Ø12] 3172:         call Anzeige
[2Z2] 3173:         call FuehreZugAus
[ØØØ] 3174:
[HA6] 3175: spØ2:   cmp byte ptr[ds:gameovr],true
[S91] 3176:         jne spØØ
[1B5] 3177:         cmp byte ptr[ds:sieger],weiss
[V91] 3178:         jne spØ3
[M42] 3179:         mov ax,msgSG
[EC1] 318Ø:         jmp sp99
[AW6] 3181: spØ3:   cmp byte ptr[ds:sieger],schwarz
[W91] 3182:         jne spØ4
[M22] 3183:         mov ax,msgCG
[EC1] 3184:         jmp sp99
[K33] 3185: spØ4:   mov ax,msgKG
[423] 3186: sp99:   call Message
[5M1] 3187:         mov ah,86h
[2L1] 3188:         mov cx,12Ø
[TH1] 3189:         xor dx,dx
[D81] 319Ø:         int 15h
[9B2] 3191: sp_abort: ret
[HP1] 3192: Spiel ENDP
[ØØØ] 3193:
[PF3] 3194:         ; ********************************
[UØ5] 3195:         ;        Mitteilung No AX anzeigen
[PF3] 3196:         ; ********************************
[ØØØ] 3197:
[852] 3198: Message PROC
[EC1] 3199:         push es
[TC1] 32ØØ:         push bx
[PF2] 32Ø1:         mov dx,gScreen
[ZC1] 32Ø2:         push dx
[2B3] 32Ø3:         cmp dx,gScreenPageØ
[V41] 32Ø4:         je msgØØ
[284] 32Ø5:         mov gScreen,gScreenPageØ
[WG1] 32Ø6:         jmp msgØ1
[HC5] 32Ø7: msgØØ:  mov gScreen,gScreenPage2
[WK2] 32Ø8: msgØ1:  mov bx,2Ø
[N31] 32Ø9:         mul bx
[5M1] 321Ø:         mov cx,278
[UE1] 3211:         add ax,2ØØ
[MN1] 3212:         call sGrab
[LL1] 3213:         mov cx,244
[WL1] 3214:         mov ax,158
[ER1] 3215:         call sDrop
[X31] 3216:         pop dx
[HH2] 3217:         mov gScreen,dx
[T31] 3218:         pop bx
[G31] 3219:         pop es
[3PØ] 322Ø:         ret
[732] 3221: Message ENDP
[ØØØ] 3222:
[PF3] 3223:         ; ********************************
[DH4] 3224:         ;            Anzeige des Zuges BX
[D55] 3225:         ;      Aktualisierung des Brettes
[5Y4] 3226:         ;       Eingabe : Spielfeldpos BX
[FM4] 3227:         ;                 Ausgabe : x/y in cx/ax
[PF3] 3228:         ; ********************************
[ØØØ] 3229:
[RX2] 323Ø: Schirmkoord PROC
[625] 3231:         mov dl,byte ptr[ds:spalte+bx]
[8Z4] 3232:         mov dh,byte ptr[ds:zeile+bx]
[FZØ] 3233:         dec dl
[3ZØ] 3234:         dec dh
[GU4] 3235:         ; x-Position in cx ermitteln
[ØF1] 3236:         xor ah,ah
[HF1] 3237:         mov al,dl
[RE1] 3238:         mov bx,25
[ZC1] 3239:         push dx
[N31] 324Ø:         mul bx
[X31] 3241:         pop dx
[EH1] 3242:         mov cx,ax
[EA1] 3243:         add cx,2Ø
[GU4] 3244:         ; y-Position in ax ermitteln
[ØF1] 3245:         xor ah,ah
[9F1] 3246:         mov al,dh
[HE1] 3247:         mov bx,22
[N31] 3248:         mul bx
[3PØ] 3249:         ret
[PV2] 325Ø: Schirmkoord ENDP
[ØØØ] 3251:
[PF3] 3252:         ; ********************************
[PP3] 3253:         ;                          Rochadezug
[NP5] 3254:         ;      als zwei Spielzüge anzeigen
[PF3] 3255:         ; ********************************
[ØØØ] 3256:
[TA3] 3257: RochadeAnzeige PROC
[V95] 3258:         mov ax,word ptr[ds:si.list+bx]
[PC1] 3259:         push ax
[7G6] 326Ø:       mov dx,word ptr[ds:si.list+bx.option]
[ZC1] 3261:         push dx
[DS2] 3262:         cmp dx,klRochade
[BØ1] 3263:         je raØ1
[ØØ3] 3264:         ; große Rochade
[2Ø1] 3265:         inc ah
[R81] 3266:         sub al,4
[1K5] 3267:         mov word ptr[ds:si.list+bx],ax
[ØE6] 3268:       mov word ptr[ds:si.list+bx.option],Ø
[Ø12] 3269:         call Anzeige
[M91] 327Ø:         jmp raØ2
[DM2] 3271:         ; kleine Rochade
[9X1] 3272: raØ1:   dec ah
[761] 3273:         add al,3
[1K5] 3274:         mov word ptr[ds:si.list+bx],ax
[ØE6] 3275:       mov word ptr[ds:si.list+bx.option],Ø
[Ø12] 3276:         call Anzeige
[1N3] 3277:         ; ok, Zug restaurieren
[A22] 3278: raØ2:   pop dx
[R31] 3279:         pop ax
[1K5] 328Ø:         mov word ptr[ds:si.list+bx],ax
[4M6] 3281:       mov word ptr[ds:si.list+bx.option],dx
[3PØ] 3282:         ret
[TA3] 3283: RochadeAnzeige ENDP
[ØØØ] 3284:
[PF3] 3285:         ; ********************************
[RB5] 3286:         ;       Animation eines Spielzuges
[PF3] 3287:         ; ********************************
[ØØØ] 3288:
[962] 3289: Anzeige PROC
[ZØ5] 329Ø:         ; kleine oder große Rochade?
[KC6] 3291:       cmp word ptr[ds:si.list+bx.option],4
[522] 3292:         jl anzNormal
[TC1] 3293:         push bx
[X95] 3294:         mov bx,word ptr[ds:si.list+bx]
[3F1] 3295:         xor bh,bh
[7B4] 3296:         mov dl,byte ptr[ds:si+bx]
[T31] 3297:         pop bx
[Ø62] 3298:         cmp dl,BauerW
[G12] 3299:         je anzNormal
[S52] 33ØØ:         cmp dl,BauerS
[G12] 33Ø1:         je anzNormal
[PD3] 33Ø2:         call RochadeAnzeige
[ØØØ] 33Ø3:
[553] 33Ø4: anzNormal: push si
[EC1] 33Ø5:         push es
[ØR1] 33Ø6:         call gCopy
[XM4] 33Ø7:         ; zuerst Startfeld löschen
[YW2] 33Ø8:         ; (mit Feld-Backup)
[TC1] 33Ø9:         push bx
[X95] 331Ø:         mov bx,word ptr[ds:si.list+bx]
[3F1] 3311:         xor bh,bh
[GV2] 3312:         call SchirmKoord
[PC1] 3313:         push ax
[WC1] 3314:         push cx
[UE1] 3315:         add ax,2ØØ
[MN1] 3316:         call sGrab
[V31] 3317:         pop cx
[R31] 3318:         pop ax
[ER1] 3319:         call sDrop
[ØØØ] 332Ø:
[PT3] 3321:         ; dann Spielfigur grabben
[T31] 3322:         pop bx
[TC1] 3323:         push bx
[X95] 3324:         mov bx,word ptr[ds:si.list+bx]
[3F1] 3325:         xor bh,bh
[3B4] 3326:         mov bl,byte ptr[ds:si+bx]
[3F1] 3327:         xor bh,bh
[V91] 3328:         shl bx,1
[V91] 3329:         shl bx,1
[RU5] 333Ø:         mov cx,word ptr[ds:FigurPos+bx]
[RV5] 3331:         mov ax,word ptr[ds:FigurPos+bx+2]
[MN1] 3332:         call sGrab
[ØØØ] 3333:
[RS5] 3334:         ; Toggle-Schleife verschiebt Figur
[T31] 3335:         pop bx
[TC1] 3336:         push bx
[V95] 3337:         mov ax,word ptr[ds:si.list+bx]
[PC1] 3338:         push ax
[8F1] 3339:         mov bl,ah
[3F1] 334Ø:         xor bh,bh
[GV2] 3341:         call SchirmKoord
[GY4] 3342:         mov word ptr[ds:xcount],cx
[EY4] 3343:         mov word ptr[ds:ycount],ax
[R31] 3344:         pop ax
[GF1] 3345:         mov bl,al
[3F1] 3346:         xor bh,bh
[GV2] 3347:         call SchirmKoord
[ØØØ] 3348:
[GX1] 3349:         ; Animation
[8K2] 335Ø: anzØØ:  push ax
[WC1] 3351:         push cx
[8G3] 3352:         call sToggle ; setzen
[GE2] 3353:         call gSwitchEm
[RØ3] 3354:         call gWaitDisplay
[V31] 3355:         pop cx
[R31] 3356:         pop ax
[PC1] 3357:         push ax
[WC1] 3358:         push cx
[5T3] 3359:         call sToggle ; löschen
[V31] 336Ø:         pop cx
[R31] 3361:         pop ax
[RR4] 3362:         cmp ax,word ptr[ds:ycount]
[M61] 3363:         jl anzØ1
[661] 3364:         jg anzØ2
[5K1] 3365:         jmp anzØ3
[672] 3366: anzØ1:  inc ax
[5K1] 3367:         jmp anzØ3
[362] 3368: anzØ2:  dec ax
[KX5] 3369: anzØ3:  cmp cx,word ptr[ds:xcount]
[R61] 337Ø:         jl anzØ4
[961] 3371:         jg anzØ5
[BK1] 3372:         jmp anzØ6
[G72] 3373: anzØ4:  inc cx
[BK1] 3374:         jmp anzØ6
[D62] 3375: anzØ5:  dec cx
[SX5] 3376: anzØ6:  cmp cx,word ptr[ds:xcount]
[PG1] 3377:         jne anzØØ
```

```
[RR4] 3378:         cmp ax,word ptr[ds:ycount]
[PG1] 3379:         jne anzØØ
[ØØØ] 3380:
[N33] 3381:         ; Zielfeld löschen
[T31] 3382:         pop bx
[TC1] 3383:         push bx
[V95] 3384:         mov ax,word ptr[ds:si.list+bx]
[3F1] 3385:         xor bh,bh
[8F1] 3386:         mov bl,ah
[GV2] 3387:         call SchirmKoord
[UE1] 3388:         add ax,2ØØ
[PC1] 3389:         push ax
[WC1] 339Ø:         push cx
[MN1] 3391:         call sGrab
[V31] 3392:         pop cx
[R31] 3393:         pop ax
[9H1] 3394:         sub ax,2ØØ
[ER1] 3395:         call sDrop
[ØØØ] 3396:
[ØX2] 3397:         ; Zugfigur setzen
[T31] 3398:         pop bx
[TC1] 3399:         push bx
[UE6] 34ØØ:      mov di,word ptr[ds:si.list+bx.option]
[X95] 3401:         mov bx,word ptr[ds:si.list+bx]
[3F1] 3402:         xor bh,bh
[7B4] 3403:         mov dl,byte ptr[ds:si+bx]
[9F1] 3404:         xor dh,dh
[NF3] 3405:         ; Bauernumwandlung??
[Ø62] 3406:         cmp dl,BauerW
[4H1] 3407:         jne anzØ7
[AC1] 3408:         cmp bl,39
[C61] 3409:         jg anzØ8
[NG1] 341Ø:         mov dx,di
[FK1] 3411:         jmp anzØ8
[SB3] 3412: anzØ7:  cmp dl,BauerS
[6H1] 3413:         jne anzØ8
[XB1] 3414:         cmp bl,8Ø
[V61] 3415:         jl anzØ8
[NG1] 3416:         mov dx,di
[DX4] 3417:         ; dx nun Figur (bei Bauernumw.
[9W2] 3418:         ; die Ersatzfigur)
[HR2] 3419: anzØ8:  mov bx,dx
[V91] 342Ø:         shl bx,1
[V91] 3421:         shl bx,1
[RU5] 3422:         mov cx,word ptr[ds:FigurPos+bx]
[RV5] 3423:         mov ax,word ptr[ds:FigurPos+bx+2]
[MN1] 3424:         call sGrab
[T31] 3425:         pop bx
[TC1] 3426:         push bx
[V95] 3427:         mov ax,word ptr[ds:si.list+bx]
[3F1] 3428:         xor bh,bh
[8F1] 3429:         mov bl,ah
[GV2] 343Ø:         call SchirmKoord
[N32] 3431:         call sToggle
[GE2] 3432:         call gSwitchEm
[ØØØ] 3433:
[T31] 3434:         pop bx
[G31] 3435:         pop es
[E31] 3436:         pop si
[3PØ] 3437:         ret
[842] 3438: Anzeige ENDP
[ØØØ] 3439:
[PF3] 344Ø:         ; ********************************
[3L4] 3441:         ;              Quit (Verabschiedung)
[PF3] 3442:         ; ********************************
[ØØØ] 3443:
[6L1] 3444: Quit PROC
[3M1] 3445:         mov ah,Ø9h
[VØ5] 3446:         cmp byte ptr[ds:abort],true
[GTØ] 3447:         je qØ4
[1B5] 3448:         cmp byte ptr[ds:sieger],weiss
[4TØ] 3449:         je qØØ
[DW5] 345Ø:         cmp byte ptr[ds:sieger],schwarz
[7TØ] 3451:         je qØ1
[DH4] 3452:         cmp byte ptr[ds:sieger],3
[ATØ] 3453:         je qØ2
[W84] 3454: qØ4:    mov dx,offset txtAB
[381] 3455:         int 21h
[3PØ] 3456:         ret
[PA4] 3457: qØØ:    mov dx,offset txtSG
[381] 3458:         int 21h
[3PØ] 3459:         ret
[A94] 346Ø: qØ1:    mov dx,offset txtCG
[381] 3461:         int 21h
[3PØ] 3462:         ret
[6A4] 3463: qØ2:    mov dx,offset txtKG
[381] 3464:         int 21h
[3PØ] 3465:         ret
[4K1] 3466: Quit ENDP
[ØØØ] 3467:
[PF3] 3468:         ; ********************************
[ET3] 3469:         ;                        Intro & Init
[PF3] 347Ø:         ; ********************************
[ØØØ] 3471:
[NS1] 3472: Intro PROC
[VW4] 3473:         ; VGA-Grafikkarte vorbereiten
[UP1] 3474:         call gInit
[UM1] 3475:         call gHide
[L44] 3476:         mov dx,seg gPCXFileTitel
[4H1] 3477:         mov ds,dx
[7W4] 3478:         mov dx,offset gPCXFileTitel
[U62] 3479:         call gLoadPCX
[HS1] 348Ø:         call gShow
[ØØØ] 3481:
[N64] 3482:         ; Titel CHESSfx anzeigen
[ØB1] 3483:         mov si,Ø
[U12] 3484: titelloop1:
[ØF1] 3485:         xor ah,ah
[WR5] 3486:         mov al,byte ptr[ds:Falldown+si]
[ZM1] 3487:         mov bx,8Ød
[N31] 3488:         mul bx
[NP1] 3489:         add ax,16ØØØ
[CH1] 349Ø:         mov bx,ax
[ZS2] 3491:         call gStartAddr
[5M1] 3492:         mov ah,86h
[PH1] 3493:         xor cx,cx
[NU1] 3494:         mov dx,22ØØØ
[D81] 3495:         int 15h
[911] 3496:         inc si
[WL5] 3497:         cmp byte ptr[ds:Falldown+si],255
[4G2] 3498:         jne titelloop1
[ØØØ] 3499:
[384] 35ØØ:         ; Schachbrett vorbereiten
[RV4] 35Ø1:         mov dx,offset gPCXFileBrett
[284] 35Ø2:         mov gScreen,gScreenPageØ
[U62] 35Ø3:         call gLoadPCX
[AM2] 35Ø4:         ; Bauern setzen
[FE1] 35Ø5:         mov cx,2Ø
[ST1] 35Ø6: SetBauern:
[WC1] 35Ø7:         push cx
[9B1] 35Ø8:         mov ax,Ø
[5L1] 35Ø9:         mov cx,23Ø
[MN1] 351Ø:         call sGrab
[FE1] 3511:         mov ax,22
[V31] 3512:         pop cx
[WC1] 3513:         push cx
[N32] 3514:         call sToggle
[9B1] 3515:         mov ax,Ø
[NL1] 3516:         mov cx,254
[MN1] 3517:         call sGrab
[6L1] 3518:         mov ax,132
[V31] 3519:         pop cx
[WC1] 352Ø:         push cx
[N32] 3521:         call sToggle
[V31] 3522:         pop cx
[SA1] 3523:         add cx,25
```

```
[HH1] 3524:         cmp cx,22Ø
[912] 3525:         jl SetBauern
[PR2] 3526:         ; Figuren setzen
[UB3] 3527:         mov si,offset Aufbau
[M32] 3528: SetFiguren:
[6F1] 3529:         xor ch,ch
[ØF1] 353Ø:         xor ah,ah
[1Z3] 3531:         mov cl,byte ptr[ds:si]
[911] 3532:         inc si
[XY3] 3533:         mov al,byte ptr[ds:si]
[911] 3534:         inc si
[MN1] 3535:         call sGrab
[6F1] 3536:         xor ch,ch
[ØF1] 3537:         xor ah,ah
[1Z3] 3538:         mov cl,byte ptr[ds:si]
[911] 3539:         inc si
[XY3] 354Ø:         mov al,byte ptr[ds:si]
[911] 3541:         inc si
[N32] 3542:         call sToggle
[GS3] 3543:         cmp si,offset Aufbau+64
[872] 3544:         jl SetFiguren
[ØØØ] 3545:
[PB4] 3546:         ; Figuren rechts löschen
[H44] 3547:         mov si,offset gPCXBuffer
[4N4] 3548:         mov word ptr[ds:ycount],Ø
[ZZ4] 3549:         mov word ptr[ds:xcount],23Ø
[VE3] 355Ø: tclear1: call gSetPix
[ZD4] 3551:         inc word ptr[ds:xcount]
[Z75] 3552:         cmp word ptr[ds:xcount],23Ø+73
[2K1] 3553:         jl tclear1
[ZZ4] 3554:         mov word ptr[ds:xcount],23Ø
[1E4] 3555:         inc word ptr[ds:ycount]
[AY4] 3556:         cmp word ptr[ds:ycount],12Ø
[2K1] 3557:         jl tclear1
[ØØØ] 3558:
[9E3] 3559:         ; Spielfeld anzeigen
[SU1] 356Ø:         mov bx,16ØØØ
[YP3] 3561: titelloop2: sub bx,8Ø
[ZS2] 3562:         call gStartAddr
[5M1] 3563:         mov ah,86h
[PH1] 3564:         xor cx,cx
[7T1] 3565:         mov dx,8ØØØ
[D81] 3566:         int 15h
[A71] 3567:         or bx,bx
[VK2] 3568:         jnz titelloop2
[RØ3] 3569:         call gWaitDisplay
[ØØØ] 357Ø:
[773] 3571:         ; Brett-Backup laden
[RV4] 3572:         mov dx,offset gPCXFileBrett
[384] 3573:         mov gScreen,gScreenPage1
[U62] 3574:         call gLoadPCX
[484] 3575:         mov gScreen,gScreenPage2
[ØØØ] 3576:
[1T4] 3577:         ; interne Schachwerte setzen
[4R2] 3578:         ; Brett aufbauen
[4P2] 3579:         mov ax,seg Brett
[3H1] 358Ø:         mov es,ax
[1H1] 3581:         mov ds,ax
[6D3] 3582:         mov si,offset Brett
[N14] 3583:         mov di,offset GameStack
[FS2] 3584:         mov aktStack,di
[LE1] 3585:         mov cx,6Ø
[LN1] 3586:         rep movsw
[ØØØ] 3587:
[XU4] 3588:         ; Vorrück-Werte definieren
[MF3] 3589:         mov si,offset VorMS
[5E3] 359Ø:         mov di,offset VorBW
[NB1] 3591:         mov cx,8
[LN1] 3592:         rep movsw
[MF3] 3593:         mov si,offset VorMS
[ND3] 3594:         mov di,offset VorBS
[NB1] 3595:         mov cx,8
[LN1] 3596:         rep movsw
[ØØØ] 3597:
[US2] 3598:      mov si,aktStack
[7E6] 3599:      mov word ptr[ds:si.alpha],alphastart
[M46] 36ØØ:      mov word ptr[ds:si.beta],betastart
[V55] 36Ø1:      mov word ptr[ds:si.matdiff],Ø
[7U5] 36Ø2:      mov word ptr[ds:si.matwertw],4Ø5Ø
[7Y5] 36Ø3:      mov word ptr[ds:si.matwerts],-4Ø5Ø
[A16] 36Ø4:      mov byte ptr[ds:si.stellungtyp],4
[8G6] 36Ø5:      mov byte ptr[ds:si.klRochWmgl],true
[WF6] 36Ø6:      mov byte ptr[ds:si.klRochSmgl],true
[GG6] 36Ø7:      mov byte ptr[ds:si.grRochWmgl],true
[4G6] 36Ø8:      mov byte ptr[ds:si.grRochSmgl],true
[Z35] 36Ø9:      mov byte ptr[ds:si.koenps],25
[E45] 361Ø:      mov byte ptr[ds:si.koenpw],95
[Z35] 3611:      mov byte ptr[ds:abort],false
[ØØØ] 3612:
[3PØ] 3613:         ret
[PR1] 3614: Intro ENDP
[ØØØ] 3615:
[PF3] 3616:         ; ********************************
[MT4] 3617:         ;                  Initialisiert Grafik
[PF3] 3618:         ; ********************************
[ØØØ] 3619:
[BR1] 362Ø: gInit PROC
[PC1] 3621:         push ax
[2C1] 3622:         push dx
[ØØØ] 3623:
[8N1] 3624:         mov ax,13h
[Y71] 3625:         int 1Øh
[ØØØ] 3626:
[PX1] 3627:         mov dx,Ø3C4h
[5M1] 3628:         mov al,Ø4h
[6H1] 3629:         out dx,al
[511] 363Ø:         inc dx
[161] 3631:         in al,dx
[ZD2] 3632:         and al,(not 8h)
[6H1] 3633:         out dx,al
[ØØØ] 3634:
[TX1] 3635:         mov dx,Ø3D4h
[6M1] 3636:         mov al,14h
[6H1] 3637:         out dx,al
[511] 3638:         inc dx
[161] 3639:         in al,dx
[MG2] 364Ø:         and al,(not 4Øh)
[6H1] 3641:         out dx,al
[ØØØ] 3642:
[LØ1] 3643:         dec dx
[CM1] 3644:         mov al,17h
[6H1] 3645:         out dx,al
[511] 3646:         inc dx
[161] 3647:         in al,dx
[181] 3648:         or al,4Øh
[6H1] 3649:         out dx,al
[ØØØ] 365Ø:
[ØS3] 3651:         mov ax,gScreenSegment
[3H1] 3652:         mov es,ax
[GH1] 3653:         xor ax,ax
[EF1] 3654:         xor di,di
[V42] 3655:         mov cx,Ø7FFFh
[KN1] 3656:         rep stosw
[ØØØ] 3657:
[X31] 3658:         pop dx
[R31] 3659:         pop ax
[3PØ] 366Ø:         ret
[CP1] 3661: gInit ENDP
[ØØØ] 3662:
[PF3] 3663:         ; ********************************
[BV4] 3664:         ;                  versteckt Bildaufbau
[PF3] 3665:         ; ********************************
[ØØØ] 3666:
[HN1] 3667: gHide PROC
[PL1] 3668:         mov ah,12h
[ZL1] 3669:         mov al,Ø1h
```

```
[EM1] 367Ø:         mov bl,36h
[Y71] 3671:         int 1Øh
[3PØ] 3672:         ret
[KM1] 3673: gHide ENDP
[ØØØ] 3674:
[PF3] 3675:         ; ********************************
[DW4] 3676:         ;              zeigt Bildschirm wieder
[PF3] 3677:         ; ********************************
[ØØØ] 3678:
[XR1] 3679: gShow PROC
[PL1] 368Ø:         mov ah,12h
[XL1] 3681:         mov al,ØØh
[EM1] 3682:         mov bl,36h
[Y71] 3683:         int 1Øh
[3PØ] 3684:         ret
[YP1] 3685: gShow ENDP
[ØØØ] 3686:
[PF3] 3687:         ; ********************************
[SY4] 3688:         ;              setzt wieder Textmodus
[PF3] 3689:         ; ********************************
[ØØØ] 3690:
[7S1] 3691: gExit PROC
[PC1] 3692:         push ax
[ØF1] 3693:         xor ah,ah
[SE1] 3694:         mov al,3h
[Y71] 3695:         int 1Øh
[R31] 3696:         pop ax
[3PØ] 3697:         ret
[8R1] 3698: gExit ENDP
[ØØØ] 3699:
[PF3] 37ØØ:         ; ********************************
[E75] 37Ø1:         ;              setzt einen Graphikpunkt
[PF3] 37Ø2:         ; ********************************
[ØØØ] 37Ø3:
[Z72] 37Ø4: gSetPix PROC
[TC1] 37Ø5:         push bx
[EC1] 37Ø6:         push es
[ZC1] 37Ø7:         push dx
[WC1] 37Ø8:         push cx
[1Z3] 37Ø9:         mov cl,byte ptr[ds:si]
[8T4] 371Ø:         mov ax,word ptr[ds:xcount]
[BT4] 3711:         mov bx,word ptr[ds:ycount]
[WC1] 3712:         push cx
[PC1] 3713:         push ax
[PC1] 3714:         push ax
[ØØØ] 3715:
[V91] 3716:         shl bx,1
[V91] 3717:         shl bx,1
[V91] 3718:         shl bx,1
[V91] 3719:         shl bx,1
[BH1] 372Ø:         mov ax,bx
[V91] 3721:         shl bx,1
[V91] 3722:         shl bx,1
[AD1] 3723:         add ax,bx
[ØØØ] 3724:
[T31] 3725:         pop bx
[LA1] 3726:         shr bx,1
[LA1] 3727:         shr bx,1
[AD1] 3728:         add ax,bx
[ØØØ] 3729:
[9E2] 373Ø:         mov di,gScreen
[2C1] 3731:         add di,ax
[ØS3] 3732:         mov ax,gScreenSegment
[3H1] 3733:         mov es,ax
[ØØØ] 3734:
[V31] 3735:         pop cx
[TB1] 3736:         and cl,3h
[WF1] 3737:         mov bx,1h
[5F1] 3738:         shl bx,cl
[ZS2] 3739:         pop cx       ; Color
[3F1] 374Ø:         mov ah,bl
[RE1] 3741:         mov al,2h
[PX1] 3742:         mov dx,Ø3C4h
[YH1] 3743:         out dx,ax
[Y64] 3744:         mov byte ptr es:[di],cl
[ØØØ] 3745:
[V31] 3746:         pop cx
[X31] 3747:         pop dx
[G31] 3748:         pop es
[T31] 3749:         pop bx
[3PØ] 375Ø:         ret
[Y52] 3751: gSetPix ENDP
[ØØØ] 3752:
[PF3] 3753:         ; ********************************
[V75] 3754:         ;      setzt die Bildschirm-Start-
[XE3] 3755:         ;                        auf ØAØØØ:BX
[PF3] 3756:         ; ********************************
[ØØØ] 3757:
[BR2] 3758: gStartAddr PROC
[RØ3] 3759:         call gWaitDisplay
[BC1] 376Ø:         push ds
[N42] 3761:         mov dx,gCRTC
[VE1] 3762:         mov ah,bh
[3N1] 3763:         mov al,ØCh
[YH1] 3764:         out dx,ax
[3F1] 3765:         mov ah,bl
[EØ1] 3766:         inc al
[YH1] 3767:         out dx,ax
[ØØØ] 3768:
[PH1] 3769:         xor cx,cx
[3H1] 377Ø:         mov ds,cx
[2L4] 3771:         mov WORD PTR ds:[Ø44Eh],bx
[ØØØ] 3772:
[E31] 3773:         pop ds
[3PØ] 3774:         ret
[BR2] 3775: gStartAddr ENDP
[ØØØ] 3776:
[PF3] 3777:         ; ********************************
[L35] 3778:         ;          Hilfsroutine für LoadPCX
[PF3] 3779:         ; ********************************
[ØØØ] 378Ø:
[LP2] 3781: gNextPixel PROC
[ZD4] 3782:         inc word ptr[ds:xcount]
[CY4] 3783:         cmp word ptr[ds:xcount],32Ø
[232] 3784:         je gnp_next
[3PØ] 3785:         ret
[AN1] 3786: gnp_next:
[1E4] 3787:         inc word ptr[ds:ycount]
[2N4] 3788:         mov word ptr[ds:xcount],Ø
[3PØ] 3789:         ret
[LP2] 379Ø: gNextPixel ENDP
[ØØØ] 3791:
[PF3] 3792:         ; ********************************
[E25] 3793:         ;    Lädt ein 32Øx2ØØx256 PCX Bild
[KC4] 3794:         ;              Dateiname bei ds:dx
[PF3] 3795:         ; ********************************
[ØØØ] 3796:
[S82] 3797: gLoadPCX PROC
[1D3] 3798:         ; Dateiname bei ds:dx
[ØE2] 3799:         ; Datei öffnen
[WM1] 38ØØ:         mov ah,3Dh
[W82] 38Ø1:         mov al,1ØØ1ØØØØb
[381] 38Ø2:         int 21h
[ØØØ] 38Ø3:
[ML2] 38Ø4:         ; 128 Bytes lesen
[CH1] 38Ø5:         mov bx,ax
[ØN1] 38Ø6:         mov ah,3Fh
[UL1] 38Ø7:         mov cx,128
[ØK3] 38Ø8:         mov dx,seg gPCXBuffer
[4H1] 38Ø9:         mov ds,dx
[Ø54] 381Ø:         mov dx,offset gPCXBuffer
[381] 3811:         int 21h
```

»Chessfx.asm« verspricht ungetrübten Schachgenuß.

```
[ØØØ] 3812:
[9Z4] 3813:         ; gültige 32Øx2ØØx256 PCX-Datei?
[H44] 3814:         mov si,offset gPCXBuffer
[YY3] 3815:         cmp byte ptr[ds:si],1Ø
[V72] 3816:         jne glPCX_No
[924] 3817:         cmp byte ptr[ds:si+3],8
[V72] 3818:         jne glPCX_No
[492] 3819:         jmp glPCX_Ok
[XN1] 382Ø: glPCX_No:
[YM1] 3821:         mov ah,3Eh
[381] 3822:         int 21h
[3PØ] 3823:         ret
[ØØØ] 3824:
[XN1] 3825: glPCX_Ok:
[2N4] 3826:         mov word ptr[ds:xcount],Ø
[4N4] 3827:         mov word ptr[ds:ycount],Ø
[N54] 3828:         mov byte ptr[ds:pack],Ø
[ØØØ] 3829:
[2C4] 383Ø: glPCX_load:       ; nachladen
[ØN1] 3831:         mov ah,3Fh
[MT1] 3832:         mov cx,1Ø24
[Ø54] 3833:         mov dx,offset gPCXBuffer
[381] 3834:         int 21h
[H44] 3835:         mov si,offset gPCXBuffer
[ØØØ] 3836:
[2M4] 3837:         mov word ptr[ds:bufpos],Ø
[R52] 3838: glPCX_loop:
[YV4] 3839:         cmp word ptr[ds:bufpos],cx
[CH2] 384Ø:         jge glPCX_load
[344] 3841:         cmp byte ptr[ds:pack],Ø
[ØS2] 3842:         jz glPCX_nopack
[ØØØ] 3843:
[WC1] 3844:         push cx
[6F1] 3845:         xor ch,ch
[984] 3846:         mov cl,byte ptr[ds:pack]
[KC2] 3847: glPCX_unpack:
[Z32] 3848:         call gSetPix
[VT2] 3849:         call gNextPixel
[E33] 385Ø:         loop glPCX_unpack
[V31] 3851:         pop cx
[N54] 3852:         mov byte ptr[ds:pack],Ø
[6P2] 3853:         jmp glPCX_next
[ØØØ] 3854:
[RB2] 3855: glPCX_nopack:
[XY3] 3856:         mov al,byte ptr[ds:si]
[SS1] 3857:         and al,ØCØh
[9U1] 3858:         cmp al,ØcØh
[ØØ3] 3859:         jnz glPCX_pixel
[XY3] 386Ø:         mov al,byte ptr[ds:si]
[9K1] 3861:         and al,3Fh
[5A4] 3862:         mov byte ptr[ds:pack],al
[6P2] 3863:         jmp glPCX_next
[ØØØ] 3864:
[M92] 3865: glPCX_pixel:
[Z32] 3866:         call gSetPix
[VT2] 3867:         call gNextPixel
[ØØØ] 3868:
[L62] 3869: glPCX_next:
[911] 387Ø:         inc si
[ZC4] 3871:         inc word ptr[ds:bufpos]
[6Y4] 3872:         cmp word ptr[ds:ycount],2ØØ
[ZØ2] 3873:         je glPCX_end
[AN2] 3874:         jmp glPCX_loop
[ØØØ] 3875:
[UU1] 3876: glPCX_end:
[TC1] 3877:         push bx
[RL2] 3878:         ; palette finden
[TL1] 3879:         mov ah,42h
[7A1] 388Ø:         mov al,2
[T52] 3881:         mov cx,ØFFFFh
[Z62] 3882:         mov dx,-(3*256)
[381] 3883:         int 21h
[CK2] 3884:         ; palette laden
[ØN1] 3885:         mov ah,3Fh
[4V1] 3886:         mov cx,3*256
[Ø54] 3887:         mov dx,offset gPCXBuffer
[381] 3888:         int 21h
[MØ3] 3889:         ; palette anpassen
[3V1] 389Ø:         mov cx,2*256
[H44] 3891:         mov si,offset gPCXBuffer
[9W1] 3892: glPCX_pal:
[7V3] 3893:         shr byte ptr[ds:si],1
[7V3] 3894:         shr byte ptr[ds:si],1
[911] 3895:         inc si
[6L2] 3896:         loop glPCX_pal
[NR2] 3897:         ; palette setzen
[UH3] 3898:         mov ax,seg gPCXBuffer
[3H1] 3899:         mov es,ax
[Ø54] 39ØØ:         mov dx,offset gPCXBuffer
[2M1] 39Ø1:         mov al,12h
[KL1] 39Ø2:         mov ah,1Øh
[LH1] 39Ø3:         xor bx,bx
[VL1] 39Ø4:         mov cx,256
[Y71] 39Ø5:         int 1Øh
[T31] 39Ø6:         pop bx
[ØØØ] 39Ø7:
[RY2] 39Ø8:         ; File schließen
[YM1] 39Ø9:         mov ah,3Eh
[381] 39ØA:         int 21h
[ØØØ] 3911:
[3PØ] 3912:         ret
[P72] 3913: gLoadPCX ENDP
[ØØØ] 3914:
[PF3] 3915:         ; ********************************
[V24] 3916:         ;                    Spritegrabber
[6M4] 3917:         ;          bei cx,ax in gPCXBuffer
[PF3] 3918:         ; ********************************
[ØØØ] 3919:
[1R1] 392Ø: sGrab PROC
[ULØ] 3921:         cld
[BC1] 3922:         push ds
[GC1] 3923:         push si
[9D1] 3924:         mov bl,2Ø
[5D1] 3925:         mov bh,24
[BC2] 3926:         mov dx,(32Ø-24)/4
[ZC1] 3927:         push dx
[TC1] 3928:         push bx
[WH3] 3929:         mov bx,seg gPCXBuffer
[4H1] 393Ø:         mov es,bx
[L34] 3931:         mov di,offset gPCXBuffer
[2L1] 3932:         mov bx,32Ø
[ØØØ] 3933:
[LA1] 3934:         shr bx,1
[LA1] 3935:         shr bx,1
[N31] 3936:         mul bx
[EH1] 3937:         mov bx,cx
[LA1] 3938:         shr bx,1
[LA1] 3939:         shr bx,1
[AD1] 394Ø:         add ax,bx
[Y24] 3941:         mov si,ax  ; si Screenpos
[T31] 3942:         pop bx
[X31] 3943:         pop dx
[ØØØ] 3944:
[ØS3] 3945:         mov ax,gScreenSegment
[KG4] 3946:         mov ds,ax  ; ds:si Screenpos
[ØØØ] 3947:
[NP1] 3948: sg_yloop:
[WC1] 3949:         push cx
[ZC1] 395Ø:         push dx
[TC1] 3951:         push bx
[UZ1] 3952:         mov dx,Ø3CEh
[TE1] 3953:         mov al,4h
[MP1] 3954: sg_xloop:
[4F1] 3955:         mov ah,cl
[9B1] 3956:         and ah,3h
[YH1] 3957:         out dx,ax
[GØ3] 3958:         movsb    ; Transfer
[UC1] 3959:         cmp ah,3h
[F12] 396Ø:         jz sg_noinc
[152] 3961:         dec si   ; div 4
[DN1] 3962: sg_noinc:
[PL1] 3963:         inc cx
[ZY9] 3964:         dec bh
[6B2] 3965:         jnz sg_xloop
[T31] 3966:         pop bx
[X31] 3967:         pop dx
[3D1] 3968:         add si,dx
[BZØ] 3969:         dec bl
[V31] 397Ø:         pop cx
[7B2] 3971:         jnz sg_yloop
[ØØØ] 3972:
[E31] 3973:         pop si
[E31] 3974:         pop ds
[3PØ] 3975:         ret
[2P1] 3976: sGrab ENDP
[ØØØ] 3977:
[PF3] 3978:         ; ********************************
[ZX4] 3979:         ;            Spielfigur gPCXBuffer
[754] 398Ø:         ;                  bei cx,ax setzen
[PF3] 3981:         ; ********************************
[ØØØ] 3982:
[2S1] 3983: sDrop PROC
[ULØ] 3984:         cld
[BC1] 3985:         push ds
[GC1] 3986:         push si
[9D1] 3987:         mov bl,2Ø
[5D1] 3988:         mov bh,24
[BC2] 3989:         mov dx,(32Ø-24)/4
[ZC1] 399Ø:         push dx
[TC1] 3991:         push bx
[WH3] 3992:         mov bx,seg gPCXBuffer
[2H1] 3993:         mov ds,bx
[H44] 3994:         mov si,offset gPCXBuffer
[KU1] 3995:         mov bx,32Ø/4
[N31] 3996:         mul bx
[EH1] 3997:         mov bx,cx
[LA1] 3998:         shr bx,1
[LA1] 3999:         shr bx,1
[AD1] 4ØØØ:         add ax,bx
[Ø24] 4ØØ1:         mov di,ax  ; si Screenpos
[8A2] 4ØØ2:         add di,gScreen
[T31] 4ØØ3:         pop bx
[X31] 4ØØ4:         pop dx
[ØØØ] 4ØØ5:
[ØS3] 4ØØ6:         mov ax,gScreenSegment
[MG4] 4ØØ7:         mov es,ax  ; ds:si Screenpos
[ØØØ] 4ØØ8:
[CP1] 4ØØ9: sd_yloop:
[WC1] 4Ø1Ø:         push cx
[ZC1] 4Ø11:         push dx
[TC1] 4Ø12:         push bx
[PX1] 4Ø13:         mov dx,Ø3C4h
[RE1] 4Ø14:         mov al,2h
[BP1] 4Ø15: sd_xloop:
[WC1] 4Ø16:         push cx
[TB1] 4Ø17:         and cl,3h
[U91] 4Ø18:         mov ah,1
[LD1] 4Ø19:         shl ah,cl
[V31] 4Ø2Ø:         pop cx
[ØØØ] 4Ø21:
[YH1] 4Ø22:         out dx,ax
[GØ3] 4Ø23:         movsb    ; Transfer
[ZC1] 4Ø24:         cmp ah,8h
[912] 4Ø25:         jz sd_noinc
[342] 4Ø26:         dec di   ; div 4
[4N1] 4Ø27: sd_noinc:
[P11] 4Ø28:         inc cx
[ZY9] 4Ø29:         dec bh
[XA2] 4Ø3Ø:         jnz sd_xloop
[T31] 4Ø31:         pop bx
[X31] 4Ø32:         pop dx
[5C1] 4Ø33:         add di,dx
[BZØ] 4Ø34:         dec bl
[V31] 4Ø35:         pop cx
[YA2] 4Ø36:         jnz sd_yloop
[ØØØ] 4Ø37:
[E31] 4Ø38:         pop si
[E31] 4Ø39:         pop ds
[3PØ] 4Ø4Ø:         ret
[3R1] 4Ø41: sDrop ENDP
[ØØØ] 4Ø42:
[PF3] 4Ø43:         ; ********************************
[V95] 4Ø44:         ;     Spielfigur bei cx,ax togglen
[PF3] 4Ø45:         ; ********************************
[ØØØ] 4Ø46:
[Ø72] 4Ø47: sToggle PROC
[ULØ] 4Ø48:         cld
[BC1] 4Ø49:         push ds
[3B1] 4Ø5Ø:         push di
[GC1] 4Ø51:         push si
[9D1] 4Ø52:         mov bl,2Ø
[5D1] 4Ø53:         mov bh,24
[PU1] 4Ø54:         mov dx,32Ø/4
[ZC1] 4Ø55:         push dx
[TC1] 4Ø56:         push bx
[WH3] 4Ø57:         mov bx,seg gPCXBuffer
[4H1] 4Ø58:         mov es,bx
[L34] 4Ø59:         mov di,offset gPCXBuffer
[KU1] 4Ø6Ø:         mov bx,32Ø/4
[N31] 4Ø61:         mul bx
[EH1] 4Ø62:         mov bx,cx
[LA1] 4Ø63:         shr bx,1
[LA1] 4Ø64:         shr bx,1
[AD1] 4Ø65:         add ax,bx
[Y24] 4Ø66:         mov si,ax  ; si Screenpos
[6B2] 4Ø67:         add si,gScreen
[T31] 4Ø68:         pop bx
[X31] 4Ø69:         pop dx
[ØØØ] 4Ø7Ø:
[ØS3] 4Ø71:         mov ax,gScreenSegment
[KG4] 4Ø72:         mov ds,ax  ; ds:si Screenpos
[ØØØ] 4Ø73:
[WR1] 4Ø74: st_yloop:
[WC1] 4Ø75:         push cx
[ZC1] 4Ø76:         push dx
[TC1] 4Ø77:         push bx
[GC1] 4Ø78:         push si
[ØØØ] 4Ø79:
[VR1] 4Ø8Ø: st_xloop:
[YL3] 4Ø81:         mov al,byte ptr es:di
[BK1] 4Ø82:         cmp al,ØØh
[BH2] 4Ø83:         jz st_skipbyte
[ØØØ] 4Ø84:
[4F1] 4Ø85:         mov ah,cl
[9B1] 4Ø86:         and ah,3h
[PC1] 4Ø87:         push ax
[UZ1] 4Ø88:         mov dx,Ø3CEh
[TE1] 4Ø89:         mov al,4h
[YH1] 4Ø9Ø:         out dx,ax
[PXØ] 4Ø91:         movsb
[4Ø1] 4Ø92:         dec si
[6ZØ] 4Ø93:         dec di
[R31] 4Ø94:         pop ax
[WC1] 4Ø95:         push cx
[NM1] 4Ø96:         xchg ah,al
[EH1] 4Ø97:         mov cx,ax
[U91] 4Ø98:         mov ah,1
[LD1] 4Ø99:         shl ah,cl
[PX1] 41ØØ:         mov dx,Ø3C4h
[RE1] 41Ø1:         mov al,2h
[YH1] 41Ø2:         out dx,ax
[BM3] 41Ø3:         mov byte ptr ds:si,ch
[ØØØ] 41Ø4:
[V31] 41Ø5:         pop cx
[ØB2] 41Ø6: st_skipbyte:
[WC1] 41Ø7:         push cx
[TB1] 41Ø8:.        and cl,3h
[AD1] 41Ø9:         cmp cl,3h
[6B2] 411Ø:         jnz st_noinc
[911] 4111:         inc si
[MP1] 4112: st_noinc:
[V31] 4113:         pop cx
[BØ1] 4114:         inc di
[P11] 4115:         inc cx
[ZY9] 4116:         dec bh
[DC2] 4117:         jnz st_xloop
[E31] 4118:         pop si
[T31] 4119:         pop bx
[X31] 412Ø:         pop dx
[3D1] 4121:         add si,dx
[V31] 4122:         pop cx
[BZØ] 4123:         dec bl
[EC2] 4124:         jnz st_yloop
[ØØØ] 4125:
[E31] 4126:         pop si
[G21] 4127:         pop di
[E31] 4128:         pop ds
[3PØ] 4129:         ret
[Z42] 413Ø: sToggle ENDP
[ØØØ] 4131:
[PF3] 4132:         ; ********************************
[L45] 4133:         ;       gSwitchEm tauscht Anzeige-
[SC5] 4134:         ;    und Bearbeitungs- Grafikseite
[PF3] 4135:         ; ********************************
[ØØØ] 4136:
[NF2] 4137: gSwitchEm PROC
[TC1] 4138:         push bx
[213] 4139:         mov ax,SEG gScreen
[1H1] 414Ø:         mov ds,ax
[MR5] 4141:         cmp WORD PTR gScreen,gScreenPageØ
[X13] 4142:         je gSwitchEm_DispØ
[ØØØ] 4143:
[AX2] 4144: gSwitchEm_Disp2:
[SC3] 4145:         mov bx,gScreenPage2
[ZS2] 4146:         CALL gStartAddr
[6T5] 4147:         mov WORD PTR gScreen,gScreenPageØ
[ØR1] 4148:         call gCopy
[T31] 4149:         pop bx
[3PØ] 415Ø:         ret
[ØØØ] 4151:
[2X2] 4152: gSwitchEm_DispØ:
[GC3] 4153:         mov bx,gScreenPageØ
[ZS2] 4154:         CALL gStartAddr
[AT5] 4155:         mov WORD PTR gScreen,gScreenPage2
[ØR1] 4156:         call gCopy
[T31] 4157:         pop bx
[3PØ] 4158:         ret
[PE2] 4159: gSwitchEm ENDP
[ØØØ] 416Ø:
[PF3] 4161:         ; ********************************
[H95] 4162:         ;      Bildschirm-Kopie herstellen
[H55] 4163:         ;          (für flüssige Animation)
[PF3] 4164:         ; ********************************
[ØØØ] 4165:
[ER1] 4166: gCopy PROC
[TC1] 4167:         push bx
[PC1] 4168:         push ax
[WC1] 4169:         push cx
[BC1] 417Ø:         push ds
[GC1] 4171:         push si
[MX1] 4172:         mov dx,3C4h
[7A1] 4173:         mov al,2
[6H1] 4174:         out dx,al
[RX1] 4175:         mov dx,3C5h
[9N1] 4176:         mov al,ØFh
[6H1] 4177:         out dx,al
[T91] 4178:         mov ah,Ø
[6A1] 4179:         mov al,1
[482] 418Ø:         call gSetMode
[754] 4181:         mov di,word ptr gScreen
[N93] 4182:         cmp di,gScreenPageØ
[3UØ] 4183:         je gc1
[5C3] 4184:         mov si,gScreenPageØ
[U71] 4185:         jmp gc2
[294] 4186: gc1:    mov si,gScreenPage2
[LR2] 4187: gc2:    mov cx,16ØØØ
[ØS3] 4188:         mov ax,gScreenSegment
[3H1] 4189:         mov es,ax
[1H1] 419Ø:         mov ds,ax
[9M1] 4191:         rep movsb
[T91] 4192:         mov ah,Ø
[5A1] 4193:         mov al,Ø
[482] 4194:         call gSetMode
[E31] 4195:         pop si
[E31] 4196:         pop ds
[V31] 4197:         pop cx
[R31] 4198:         pop ax
[T31] 4199:         pop bx
[3PØ] 42ØØ:         ret
[FP1] 4201: gCopy ENDP
[ØØØ] 4202:
[PF3] 4203:         ; ********************************
[9V5] 4204:         ; setzt VGA-Schreibmodus r=ah,w=al
[PF3] 4205:         ; ********************************
[ØØØ] 4206:
[S82] 4207: gSetMode PROC
[7B1] 4208:         and ah,1h
[981] 4209:         shl ah,1
[981] 421Ø:         shl ah,1
[981] 4211:         shl ah,1
[NB1] 4212:         and al,3h
[1B1] 4213:         add ah,al
[T71] 4214:         or ah,4Øh
[UZ1] 4215:         mov dx,Ø3CEh
[UE1] 4216:         mov al,5h
[YH1] 4217:         out dx,ax
[3PØ] 4218:         ret
[P72] 4219: gSetMode ENDP
[ØØØ] 422Ø:
[PF3] 4221:         ; ********************************
[6E5] 4222:         ;        wartet auf Strahlrücklauf
[PF3] 4223:         ; ********************************
[ØØØ] 4224:
[TZ2] 4225: gWaitDisplay PROC
[PC1] 4226:         push ax
[DC2] 4227:         mov dx,gCRTC+6
[XB2] 4228: wd_r:   in al,dx
[KN1] 4229:         test al,8d
[6B1] 423Ø:         jz wd_r
[FB2] 4231: wd_d:   in al,dx
[KN1] 4232:         test al,8d
[ØK1] 4233:         jnz wd_d
[R31] 4234:         pop ax
[3PØ] 4235:         ret
[RY2] 4236: gWaitDisplay ENDP
[ØØØ] 4237:
[4MØ] 4238:         END
[ØØØ] 4239:
```

»Chessfx.asm« verspricht ungetrübten Schachgenuß.

Goldgräber

Sammeln Sie den Müll aus dem All

Spätestens nach überstandener Tetris-Entziehungskur denkt sich so mancher Computerbesitzer: »So etwas müßte ich doch auch programmieren können!« Eine Spielidee ist schnell (nach)erfunden, und eine Programmiersprache hat man schließlich auch in der Hinterhand. Der Computer läuft schon warm, also fix den Quelltext geschrieben.

So oder ähnlich beginnen wohl alle Versuche, ein eigenes Spiel in den Arbeitsspeicher zu bringen. Das Ergebnis ist meist ernüchternd. Die Kritiken reichen von verhaltenem Wohlwollen – aber gespielt wird es nicht – bis zur schroffen Ablehnung. Auf dem einen Computer läuft das Spiel zu schnell, auf einem anderen zu langsam oder verursacht einen sofortigen Systemabsturz. Die Grafik hat man sich ganz anders vorgestellt, und wieso die Steuerung des Spiels kompliziert sein soll, bleibt das Geheimnis der undankbaren Spieler.

Für die Programmierung eines Spielehits gibt es sicher ebensowenig ein Patentrezept wie für die Komposition eines Popsongs für die Top ten. Hier soll es aber nicht darum gehen, den großen Softwareschmieden Konkurrenz zu machen.

Die Umsetzung einer Spielidee ist wegen der im allgemeinen hohen Komplexität der Probleme eine äußerst reizvolle Herausforderung und damit ein ideales Übungsfeld. Bei der Suche nach einer Idee können Sie sich folgende Fragen stellen:

- Welchem Genre sollte das Spiel angehören? Ohne eine starre Einteilung vornehmen zu wollen, seien Kategorien wie Adventures, Rollenspiele, technische und sportliche Simulationen, Denk- und Strategiespiele, Jump'n'Run- sowie Actionspiele genannt.
- Bietet sich die Modifizierung einer bereits existierenden Idee an? Hierbei ist es nicht so einfach, tatsächlich das Original zu übertreffen.

Die Spielidee

Mit dem vorliegenden Spielprogramm wurden folgende Ziele umgesetzt:

- Umsetzung eines relativ einfachen Reaktionsspiels
- Einbringen von Elementen, die verhindern, daß das Spiel zu einem reinen Ballerspiel ausartet
- einfache Steuerung über Tastatur
- Unterstützung der Hercules-, CGA- und VGA-Karte
- Sicherung der Spielbarkeit auf Rechnern beliebiger Geschwindigkeiten, also vom XT mit 4,77 MHz bis zum 486er mit 66 MHz

- Wendet sich das Spiel an erfahrene Computerfreaks oder an den sporadisch spielenden Anwender?
- Ist das Programm auch für ein kurzes Spiel zwischendurch geeignet oder muß man viel Zeit einkalkulieren?

Die wichtigste Frage aber ist wohl die nach der eigenen Fantasie und den programmtechnischen Rahmenbedingungen zur Umsetzung der Idee. Daraus ergibt sich eventuell die Notwendigkeit, geeignete Partner zu suchen, die Teilprobleme lösen. Haben Sie eine Spielidee, so ergeben sich weitere Fragen: Welche Programmiersprache ist geeignet? In den meisten Fällen wird man um eine Assembler-Programmierung nicht umhin kommen.

Taste	Bedeutung
F1	Panik-(Boß-)Taste, ruft »command.com« auf; mit Kommando »exit« ins Spiel zurück
F2	Aus-/Einschalten der Soundeffekte
Cursor links	Tempo verringern (Bremse)
Cursor auf	Sprung (Höhe und Weite sind vom momentanen Tempo abhängig)
Cursor rechts	Tempo erhöhen (Gas)
Umschalt (links)	Schießen

Tabelle 1. Das Programm Goldgräber 3000 steuern Sie mit diesen Tasten.



Bild 1. Die verschiedenen Sprites

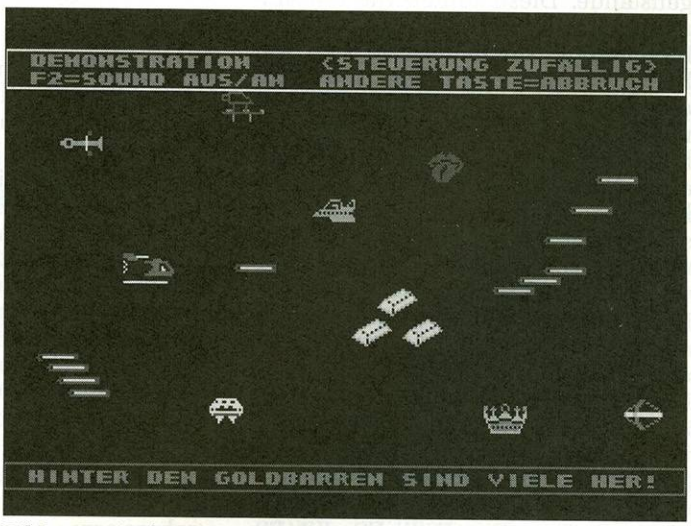


Bild 2. Typische Szene aus Goldgräber 3000



Bild 3. In diese drei Abschnitte ist der Bildschirm aufgeteilt.

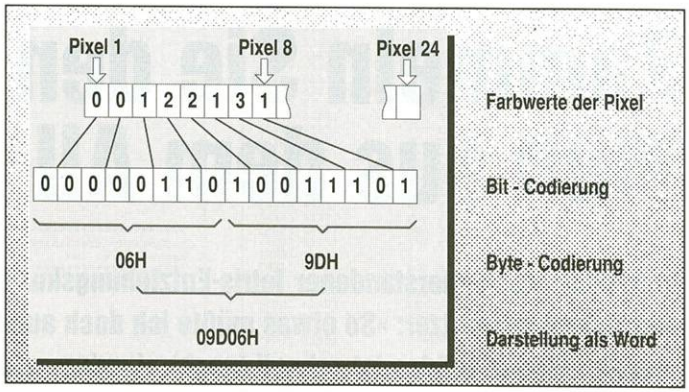


Bild 4. So wird ein Sprite codiert.

Welche Grafikkarten sollen unterstützt werden? Reicht eine Tastatursteuerung aus oder wird Joystick und/oder Mausbedienung angeboten? In welchem Umfang wird Sound eingesetzt? Welche zusätzlichen Programmelemente werden verwendet? Erfahrungsgemäß hat zum Beispiel eine Bestenliste einen positiven Einfluß auf die Spielmotivation. Weitere Elemente wären Informationen zum Spiel, ein Demonstrationsmodus und eine Boß-Taste, die das Spiel sofort unterbricht und »seriöse« Bildschirmarbeit simuliert.

Nach erfolgreicher Programmierung sollten Sie sich genügend Zeit für die Fehlersuche und -beseitigung, besonders aber für das Abrunden der Spielidee und die Spielbarkeit nehmen.

Auch wenn man sich sowohl um eine sehr gute Idee als auch um eine ebensolche Grafik- und Soundumsetzung bemühen sollte, sei hier folgende Erkenntnis des Autors bezüglich der minimalen Gestaltung eines Spielprogramms gestattet: An der Grafik und dem Sound eines Spielprogramms sind in dem Maße Abstriche zulässig, wie die Spielidee hervorragend ist.

- **Die Goldgräber-Geschichte:** Sie schlüpfen in die Rolle eines Raumfahrers im Jahre 3000, der den in das Weltall entsorgten Zivilisationsmüll (Bild 1) zu beseitigen hat. Dies erreichen Sie durch Beschuß mit einer Laserkanone. Unter dem Müll befinden sich aber, wie so oft, auch recht brauchbare Gegenstände. Diese sollten Sie einsammeln. Darüber hinaus gibt es noch alte Raumbasen, auf denen Sie sich mit Energie versorgen können, sowie andere freundlich oder feindlich gesonnene Astronautenkollegen. Um eine hohe Punktzahl zu erreichen, sind also schnelles Erfassen der entgegenkommenden Teile und richtige Entscheidungen über Abschuß, Einsammeln (durch Kollisionen) oder Ausweichen angebracht.

Die Müllteile bewegen sich von rechts nach links über den Bildschirm, während sich der Spieler (Goldgräber) links befindet und seine Position nur in der Höhe ändert (Bild 2). Das Spiel ist beendet, wenn die Schutzenergie (Energie) aufgebraucht ist. Diese verringert sich durch Kollisionen mit einigen Sprites. Der Energievorrat für die Laserkanone (Laser) wird unabhängig von der Schutzenergie verwaltet und verringert sich mit jedem Schuß. Die Steuerung des Spiels ist aus Tabelle 1 ersichtlich.

Taste	Bedeutung
Cursortasten	Positionsveränderung
0, 1, 2, 3	aktuellen Farbwert auswählen (0 = Hintergrund)
Enter	Pixel mit aktueller Farbe setzen
Z	Pinsel ein/aus, bei »ein« setzen auch die Cursortasten die Pixel
F2	Inhalt des aktuellen Fensters kopieren (neue Position, dann nochmals F2)
F3	Fensterrand sichtbar/nicht sichtbar
F4	Fenster definieren, Schritte: 1. Position wählen, dann nochmals F4 drücken. 2. Mit Cursortasten rechte untere Ecke festlegen und letztmalig F4 drücken.
F5	Inhalt des aktuellen Fensters als Asm-Quelltext retten mit Auswahl, ob byte- oder wortweise (DB oder DW). Format entspricht einem Speicherabzug (320×200 Bildpunkte bei vier Farben), je Pixel zwei Bit.
F6	gesamte Zeichenfläche speichern
F7	gespeicherte Fläche wieder laden
F8	Grafik im Fenster in Originalgröße (CGA) zeigen
F9	Grafik im Fenster in Originalgröße monochrom zeigen

Tabelle 2. Mit diesen Tasten bedienen Sie das Malprogramm »zeidef.pas«.

Verschiedene Grafikkarten werden unterstützt

Damit Goldgräber auf verschiedenen Grafikkarten läuft, wird zwischen der physikalisch zur Verfügung stehenden, grafikkartenabhängigen Bildschirmauflösung und einer logischen, programminternen Aufteilung unterschieden.

Hier bietet sich eine logische Auflösung von 320×174 Bildpunkten an. Für die CGA- und VGA-Karte bedeutet das den Verzicht auf 26 Pixelzeilen. Um die Hercules-Karte optimal auszunutzen, wird jedes logische Pixel im Format 2×2, also viermal ausgegeben, so daß diese Karte im Format 640×348 ausgenutzt wird. Damit wird auf 80 Pixelspalten verzichtet.

Die Anpassung zwischen den logischen und physischen Gegebenheiten erfolgt in den drei Grafikmodulen »ggherc.asm«, »ggcga.asm« und »ggvga.asm« (Listings 1 bis 3). Das eigentliche Spielprogramm braucht damit auf die unterschiedlichen Grafikkarten keine Rücksicht mehr zu nehmen. Im weiteren wird nur noch von der logischen Auflösung ausgegangen.

Der zur Verfügung stehende Bildschirmausschnitt wird entsprechend Bild 3 in die drei Bereiche Spielstandanzeige, Spielfläche und Untertitel aufgeteilt.

- **So werden die Sprites gemacht:** Die in Bild 1 gezeigten Sprites haben ein Format von 24×12 Pixeln, wobei tatsächlich nur eine Breite von 22 Pixeln ausgenutzt wird. Da auch die Darstellung der Sprites im Bildwiederholspeicher abhängig von der Grafikkarte ist, wurde für die interne Ablage der Aufbau entsprechend der CGA-Auflösung als Kompromiß gewählt. Es werden so für jedes Pixel im Sprite zwei Bit benötigt, und jedes Sprite ist damit intern mit maximal vier verschiedenen Farbwerten codiert (Bild 4). Für die Ausgabe auf einer Hercules-Karte werden die Farbwerte 1 bis 3 als gesetzt, der Farbwert 0 als nicht gesetzt (Hintergrund) interpretiert. Um die Farbenvielfalt der VGA-Karte zu nut-

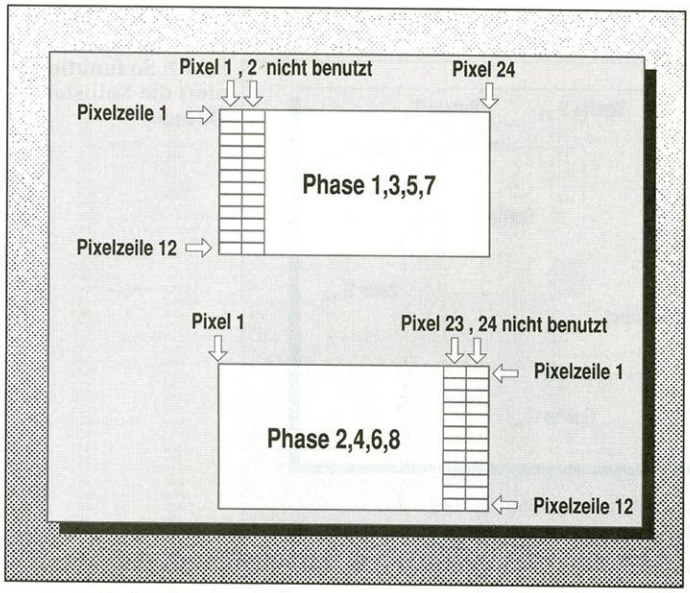
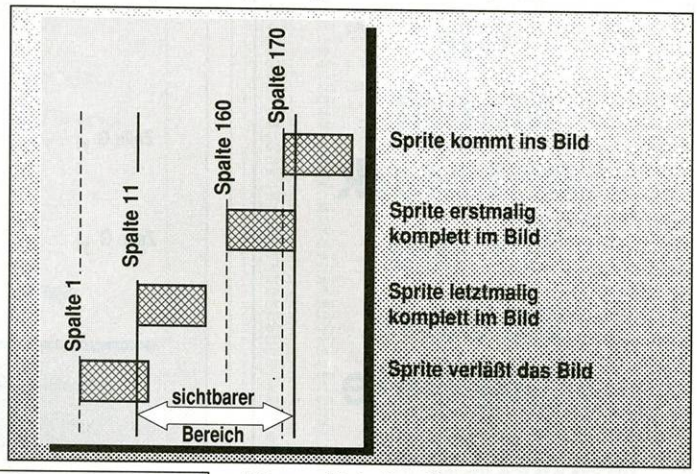


Bild 5. Der verfügbare Pixelbereich zur Gestaltung eines Sprites

Bild 6. Die logischen Spaltenpositionen der Sprites



zen, enthält das Modul »ggvga.asm« eine Tabelle verschiedener Farbzusammenstellungen aus jeweils vier Komponenten. Bei der Ausgabe eines Sprites wird entsprechend dessen interner Nummer eine Farbkombination aus dieser Tabelle zugeordnet und für die Darstellung verwendet.

Jedes sich über den Bildschirm bewegende Sprite ist mit mindestens zwei Phasen abgelegt. Dabei sind die Phasen mit gerader Nummer (2, 4,...) innerhalb der 24 Pixel um zwei Pixel nach links verschoben (Bild 5). Damit wird das Bildschirmrollen der Sprites vorbereitet.

Alle Grafiken liegen in der Datei »ggobj.exe« in komprimierter Form vor. Die Datei »u01.pic« enthält die Codierung eines verwendeten Sprites. Entsprechend Bild 4 können Sie die Werte und damit das Aussehen des Sprites manipulieren.

Scrolling auf verschieden schnellen Rechnern

Um eine weitgehend »ruckelfreie« Bewegung der Sprites über den Bildschirm zu erreichen, aber gleichzeitig auch auf langsamen Rechnern ein annehmbares Tempo zu gewährleisten, erfolgt die Animation in Schritten von zwei Pixeln. Bei einer Bildschirmbreite von 320 Pixeln werden also 160 Schritte (Takte) für eine Überquerung gebraucht. Da die Sprites fließend aus dem Bildschirm wandern, kommen bei einer Spritebreite von 22 Pixeln noch elf Takte dazu. Damit bietet sich zur bisherigen Aufteilung des Bildschirms noch die Einführung von 171 logischen Spalten an (Bild 6). Die logische Spaltenposition 170 steht also für den Eintritt des Sprites auf den Bildschirm, Spaltenposition 11 für die letztmalige komplette Anzeige, Spaltenposition 1 für die letztmalige Sichtbarkeit; Spaltenposition 0 wird im Programm als Zeichen der Deaktivierung des Sprites verwendet. Die Ausgabe der Sprites erfolgt durch eine Exklusiv-oder-Verknüpfung (XOR) im Bildwiederholspeicher, so daß das Löschen durch eine zweite Ausgabe auf die gleiche Position erreicht wird.

Die Bewegung der Sprites erfolgt nach folgendem Prinzip:

- Nach Ausgabe der ersten Phase eines Sprites wird die zweite Phase an der gleichen Bildschirmposition ausgegeben. Aus Bild 5 ist ersichtlich, daß damit eine Bewegung um zwei Pixel nach links erreicht wird.
- Anschließend wird die Ausgabeposition um vier Pixel nach links gesetzt, was bei einer CGA-Karte eine Veränderung um 1 Byte im Bildwiederholspeicher entspricht, und die erste oder die nächste ungerade Phase ausgegeben. Das bewirkt letztlich eine Verschiebung des Sprites um zwei Pixel.

Erzeugen des Spielprogramms

Die Datei »ggobj.exe« (Listing 14) enthält Grafikdaten in gepackter Form. Wenn Sie also diese Datei aufrufen, erhalten Sie automatisch die Dateien »gglogos.obj«, »ggraum.obj« und »ggsprit1.obj«. Die ausführbare Datei »gg3000.exe« erhalten Sie aus den Listings 1 bis 15 nach Ausführen der Stapeldatei »go.bat« (Listing 17).

Kollision des Goldgräbers mit anderen Sprites

Ähnlich wie bei einem Trickfilm kann durch die unterschiedliche Gestaltung der einzelnen Phasen eines Sprites erreicht werden, daß das Sprite neben der eigentlichen Vorwärtsbewegung in sich selbst eine eigene Bewegung vollführt (zum Beispiel das Tor zu den Extraräumen). Hierzu sollten Sie vier oder acht Phasen bereitstellen (Malprogramm »zeidef.pas«, Listing 16, Tabelle 2).

Wesentlich für die Auswertung der jeweiligen Spielsituation ist das Erkennen von Kollisionen des Goldgräbers mit anderen Sprites und das Treffen beim Schießen. Aus Bild 7 erkennen Sie, daß eine Kollision dann vorliegt, wenn sich die durch den Goldgräber und ein Sprite bedeckten Flächen überschneiden. Die Flächen selbst sind jeweils durch ihre linke Spalten- und obere Pixelzeilenposition und die Spaltenbreite beziehungsweise Pixelzeilenhöhe definiert. Deshalb wird ein eventueller Zusammenstoß vom Programm durch Koordinatenvergleich ermittelt.

Ein Treffer beim Schießen des Goldgräbers wird zunächst durch Auslesen der entsprechenden Pixelzeile im Bildwiederholspeicher erkannt, wobei die Trefferspalte berechnet wird. Durch Kontrolle der Spritetabelle »TeileTab« aus »gghaupt.asm« wird sowohl bei einer Kollision als auch bei einem Treffer das dazugehörige Sprite identifiziert. Die Reaktionen der Sprites auf Treffer durch den und Kollisionen mit dem Goldgräber sind im Modul »ggjumps.inc« aufgeführt. Auch für die Bewegung der Sprites gibt es verschiedene Arten und damit Unterprogramme. Die Zuordnung der jeweiligen Unterprogramme zum richtigen Sprite erfolgt mit der internen Spritenummer über drei Sprungleisten (»TrefferLeiste«, »CrashLeiste« und »AusgabeLeiste«).

- **Die Steuerung des Spiels:** Die Reaktionen des Programms auf Tastatureingaben und die Steuerung des Programmablaufs erfolgen über ein Prinzip der Nachrichtenübermittlung an alle relevanten Programmteile, die jeweils »ihre« Information herausnehmen und entsprechend reagieren. Im Programm »gghaupt.asm« erkennen Sie, daß beim Spielablauf (Prozedur »mp_spiele«) die Hauptschleife ab der Marke »game« jedes Unterprogramm aufruft, ohne hier schon zu prüfen, ob überhaupt eine Aktivität

Wettbewerb

Monat für Monat 1024 Mark für die besten 1024 Byte

Programmieren Sie in Assembler, Basic, C oder Pascal? Haben Sie ein oder mehrere Programme geschrieben, die das eine oder andere Problem auf geniale Weise lösen?
Dann zögern Sie nicht, und senden Sie den Quelltext und die lauffähige Version auf 5 1/4- oder 3 1/2-Zoll-Diskette zusammen mit Ihrer kompletten Anschrift und Angaben über den verwendeten Compiler an die

Redaktion
DOS International
»1024«
Gruberstraße 46 c
8011 Poing bei München

Jeden Monat präsentieren wir das beste eingegangene Listing mit 1024 Mark. Die einzige Bedingung: Der Quelltext darf nicht länger als 1024 Byte (1 KByte) sein. Nicht prämierte Listings werden bei Veröffentlichung nach den üblichen Sätzen honoriert.

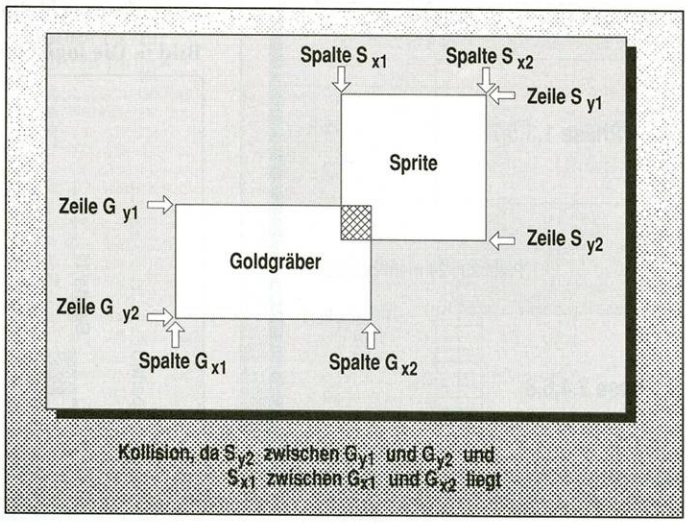


Bild 7. So funktioniert die Kollisionsabfrage.

notwendig ist. Die Unterprogramme kontrollieren zunächst, ob sie abgearbeitet werden müssen. Als Beispiel prüft »PunkteOut« zuerst, ob sich der Punktestand verändert hat. Die Nachrichten an die Unterprogramme werden zum Beispiel über die Variablen »gefeuert«, »escape«, »gameover« und andere übermittelt.

- **Hardwareanpassung:** Das Modul »ggenv.inc« enthält mit der Prozedur »grtest« eine Routine zur automatischen Erkennung der vorhandenen Grafikkarte. Falls Sie die Einstellung der verwendeten Grafikkarte selbst festlegen möchten, geben Sie beim Aufruf von »gg3000.exe« einen entsprechenden Parameter an. Dabei steht 1 für die Hercules-, 2 für die CGA- und 3 für die VGA-Karte.
Ab der Marke »grtest8« wird die hardwareabhängige Tempokonstante »tcount« ermittelt. Sie gewährleistet, daß das Programm auf verschiedenen Rechnerkonfigurationen mit gleicher Geschwindigkeit läuft.
- **Arbeit mit Tabellen:** Ein wichtiges Hilfsmittel zur Speicherung verschiedener Programm- und Spielsituationen sind Tabellen. So enthält »TeileTab« aus »gghaupt.asm« alle Informationen über die Sprites auf dem Bildschirm, »Grundbild« aus »ggr.asm« die Folge der Sprites, wie sie auf dem Bildschirm erscheinen, »Extra01« (»ggr.asm«) die Bezeichnungen und den Aufbau der sogenannten Extraräume.

(Detlef Paulin/bm)

Literatur:
– Martin Althaus: Programmierung der Grafikkarten, DOS International 4'91, S. 238
– Detlef Paulin: Eine Warteschleife für jeden Rechner, DOS International 9'91, S. 328

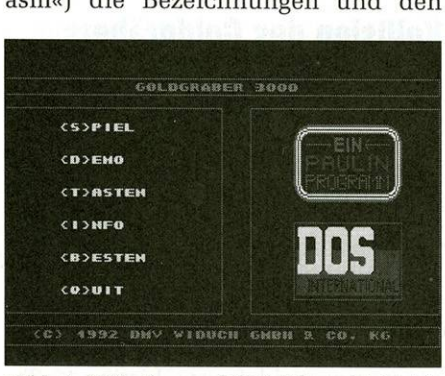


Bild 8. Mit diesem Titelbild meldet sich das Spiel.

```
[ØD3]    1: ; Programm: ggcga.asm
[BT6]    2: ; Funktion: Grafikroutine für CGA-Modus
[BC3]    3: ;           (320x2ØØ 4 Farben)
[913]    4: ; Sprache:  TASM 2.Ø1
[VT3]    5: ; Autor:    Detlef Paulin
[7A4]    6: ; (c) 1992 Widuch GmbH & Co. KG
[ØØØ]    7:
[XY1]    8:             .MODEL SMALL
[T8]     9: video                    equ     Øb8ØØh
[1K3]   10: zeile1                   equ     Ø
[S24]   11: zeile2                   equ     2ØØØh
[ZW4]   12: zeilenkorrektur          equ     16
[AU4]   13: spaltenkorrektur         equ     Ø
[ØØØ]   14:
[2SØ]   15:             .DATA
[ØØØ]   16:
[7M7]   17: ; Konstanten für horizontale und vertikale
[891]   18: ; Linien
[MK5]   19: HLTab    db      Øffh,3fh,Øfh,Ø3h,Øc0h,Øf0h
[ØE4]   20:          db      Øfch,Øffh,Øc0h,3Øh,Øch,Ø3h
[ØØØ]   21:
[MF3]   22: ; Inhalt der Routinen
[ØØØ]   23:
[PM3]   24: ; Name:       XORmitWORD
[TM5]   25: ; Funktion:   Ausgabe eines Sprites
[AX5]   26: ; Eingang:    AH=Pixelzeile, AL=Spalte
[7M4]   27: ;             DH=Pixelzeilenhöhe
[XW3]   28: ;             DL=Breite in Word
[NB5]   29: ;             BX=Zeiger auf Spritephase
[RB6]   3Ø: ; Ausgang:    BX=Zeiger auf nächste Phase
[984]   31: ; Rettet :    AX,DX,ES,DI,SI
[ØØØ]   32:
[ZM3]   33: ; Name:       XORmitBYTE
[UN6]   34: ; wie XORmitWORD, aber DL=Breite in Byte
[ØØØ]   35:
[BU2]   36: ; Name:       CHAR8x5
[BK6]   37: ; Funktion:   Ausgabe eines Zeichens des
[PE5]   38: ;             (8x5) Pixel Zeichensatztes
[AX5]   39: ; Eingang:    AH=Pixelzeile, AL=Spalte
[L25]   4Ø: ;             DL=Zeichen im ASCII-Code
[MU4]   41: ;             DH=Farbe (Ø,55h,Øaah,Øffh)
[VG5]   42: ;             BX-Adresse Zeichentabelle
[CW2]   43: ; Ausgang:    AL=AL+2
[2M4]   44: ; Rettet:     AH,BX,DX,ES,DI,SI
[ØØØ]   45:
[UA3]   46: ; Name:       TextModus
[FX4]   47: ; Funktion:   setzt Text-Modus
[ØØØ]   48:
[FM3]   49: ; Name:       GrafikModus
[HN5]   5Ø: ; Funktion:   setzt CGA-Grafikmodus
[ØØØ]   51:
[S03]   52: ; Name:       HoriLine
[KR6]   53: ; Funktion:   Zeichnet horizontale Linie
[HR5]   54: ; Eingang:    AH=Pixelzeile, AL=Farbe
[4L4]   55: ;             BX=von Pixelspalte
[B86]   56: ;             DX=bis Pixelspalte (DX-BX>3 !)
[DM3]   57: ; Rettet:     AX,BX,CX,DX
[ØØØ]   58:
[G13]   59: ; Name:       VertLine
[V96]   6Ø: ; Funktion:   Zeichnet vertikale Linie
[553]   61: ; Eingang:    AL=Farbe
[LV3]   62: ;             BX=Pixelspalte
[A74]   63: ;             DL=von Pixelzeile
[M54]   64: ;             DH=bis Pixelzeile
[DM3]   65: ; Rettet:     AX,BX,CX,DX
[ØØØ]   66:
[4S3]   67: ; Name:       WasGetroffen
[2Y5]   68: ; Funktion:   Prüft, ob es Treffer gab
[AX5]   69: ; Eingang:    AH=Pixelzeile, AL=Spalte
[XL6]   7Ø: ; Ausgang:    BX=log. Spalte des Treffers
[C46]   71: ;             BX>=151, falls nicht getroffen
[R96]   72: ;             ES:[DI]=Adresse Trefferstelle
[784]   73: ;             auf dem Bildschirm
[ØØØ]   74:
[FTØ]   75:             .CODE
[ØØØ]   76:
[7Y5]   77: ; Über Prozedur CGA_UP erfolgt die
[GW5]   78: ; Kommunikation mit Hauptprogramm
[ØØØ]   79:
[393]   8Ø:             GLOBAL  CGA_UP:PROC
[ØØØ]   81:
[2F3]   82: ; Macro-Definitionen
[ØØØ]   83:
[AØ4]   84: PUSHEN  MACRO   RegLeiste
[DA3]   85:          IRP     Reg,RegLeiste
[2S1]   86:          push    Reg
[HPØ]   87:          ENDM
```

```
[HPØ]   88:          ENDM
[ØØØ]   89:
[HV3]   90: POPEN    MACRO    RegLeiste
[DA3]   91:          IRP      Reg,RegLeiste
[WN1]   92:          pop      Reg
[HPØ]   93:          ENDM
[HPØ]   94:          ENDM
[ØØØ]   95:
[NH6]   96: ; PIXADR berechnet Adresse im Bildwieder-
[PM5]   97: ; holspeicher für log. Koordinaten
[AU5]   98: ; Eingang: AH=Pixelzeile, AL=Spalte
[875]   99: ; Ausgang: ES:[DI]=Adresse von AH
[UU4]  100: ;           ES:[SI]=Adresse von AH+1
[ØØØ]  101:
[L42]  102: PIXADR   MACRO
[NV1]  103:          LOCAL    pixØ
[8F2]  104:          mov      di,zeile1
[8G2]  105:          mov      si,zeile2
[SY6]  106: ; Korrektur der Position log.->physisch
[U44]  107:          add      ah,zeilenkorrektur
[5F4]  108:          add      al,spaltenkorrektur
[ER4]  109:          test     ah,1     ;ungerade Zeile?
[6T2]  110:          jz       pixØ     ;nein
[GL2]  111:          xchg     di,si    ;ja
[7M1]  112:          add      si,8Ø
[7D3]  113: pixØ:    mov      cx,Øfeh
[FY3]  114:          and      cl,ah    ;cx=zeile(-1)
[LU3]  115:          xor      ah,ah    ;ax=spalte
[XL1]  116:          shl      cx,1
[XL1]  117:          shl      cx,1
[MV3]  118:          shl      cx,1     ;cx=zeile*8
[BR1]  119:          add      ax,cx
[XL1]  120:          shl      cx,1
[KW1]  121:          shl      cx,1     ;cx=zeile*32
[D85]  122:          add      ax,cx    ;ax=4Ø*zeile+spalte
[GV3]  123:          add      di,ax    ;di  1. Zeile
[YX3]  124:          add      si,ax    ;si  2. zeile
[GC2]  125:          mov      ax,video
[3V1]  126:          mov      es,ax
[HPØ]  127:          ENDM
[ØØØ]  128:
[967]  129: ; Unterprogrammverteiler entsprechend der
[X47]  130: ; in CX übergebenen Unterprogrammnummer
[ØØØ]  131:
[DØ2]  132: CGA_UP   PROC
[XF1]  133:          push     ax
[BC4]  134:          mov      ax,offset CGALeiste
[BR1]  135:          add      ax,cx
[BR1]  136:          add      ax,cx
[BR1]  137:          add      ax,cx
[DU1]  138:          xchg     ax,cx
[RD1]  139:          pop      ax
[LC1]  140:          call     cx
[3PØ]  141:          ret
[8R1]  142: CGALeiste:
[PZ2]  143:          jmp      XORmitWORD
[LX2]  144:          jmp      XORmitBYTE
[V82]  145:          jmp      CHAR8x5
[SP2]  146:          jmp      TextModus
[K53]  147:          jmp      GrafikModus
[TG2]  148:          jmp      HoriLine
[2H2]  149:          jmp      VertLine
[W73]  150:          jmp      WasGetroffen
[CY1]  151: CGA_UP   ENDP
[ØØØ]  152:
[G63]  153: XORmitWORD       PROC
[9Y3]  154:          PUSHEN   <AX,DX,ES,DI,SI>
[U41]  155:          PIXADR
[6T1]  156:          xor      ch,ch
[7L1]  157:          shr      dh,1
[YX4]  158:          mov      cl,dh    ;cx=Zeilenläufe
[ZB2]  159: @@1:     push     cx
[PE1]  160:          push     di
[MF1]  161:          push     si
[A35]  162:          mov      cl,dl    ;cx=Spaltenläufe
[PA3]  163: @@2:     mov      ax,[bx]
[SW2]  164:          xor      es:[di],ax
[CH1]  165:          add      bx,2
[3G1]  166:          add      di,2
[TK1]  167:          loop     @@2
[NT1]  168:          mov      cl,dl
[UA3]  169: @@3:     mov      ax,[bx]
[PX2]  170:          xor      es:[si],ax
[CH1]  171:          add      bx,2
[1H1]  172:          add      si,2
[XK1]  173:          loop     @@3
[ED1]  174:          pop      si
[GC1]  175:          pop      di
[7M1]  176:          add      si,8Ø
[9L1]  177:          add      di,8Ø
[VD1]  178:          pop      cx
[NK1]  179:          loop     @@1
[KS3]  180:          POPEN    <SI,DI,ES,DX,AX>
[3PØ]  181:          ret
[F43]  182: XORmitWORD       ENDP
[ØØØ]  183:
[D43]  184: XORmitBYTE       PROC
[9Y3]  185:          PUSHEN   <AX,DX,ES,DI,SI>
[U41]  186:          PIXADR
[6T1]  187:          xor      ch,ch
[7L1]  188:          shr      dh,1
[YX4]  189:          mov      cl,dh    ;cx=Zeilenläufe
[BC2]  190: @@4:     push     cx
[PE1]  191:          push     di
[MF1]  192:          push     si
[A35]  193:          mov      cl,dl    ;cx=Spaltenläufe
[Y93]  194: @@5:     mov      al,[bx]
[MV2]  195:          xor      es:[di],al
[MB1]  196:          inc      bx
[BA1]  197:          inc      di
[5L1]  198:          loop     @@5
[NT1]  199:          mov      cl,dl
[2A3]  200: @@6:     mov      al,[bx]
[KW2]  201:          xor      es:[si],al
[MB1]  202:          inc      bx
[9B1]  203:          inc      si
[9L1]  204:          loop     @@6
[ED1]  205:          pop      si
[GC1]  206:          pop      di
[7M1]  207:          add      si,8Ø
[9L1]  208:          add      di,8Ø
[VD1]  209:          pop      cx
[1L1]  210:          loop     @@4
[KS3]  211:          POPEN    <SI,DI,ES,DX,AX>
[3PØ]  212:          ret
[C23]  213: XORmitBYTE       ENDP
[ØØØ]  214:
[N12]  215: CHAR8x5 PROC
[PG4]  216:          PUSHEN   <AX,BX,DX,ES,DI,SI>
[U41]  217:          PIXADR
[GV1]  218:          xor      ax,ax
[HT1]  219:          mov      al,dl
[YP1]  220:          sub      ax,32
[TL1]  221:          shl      ax,1
[BR1]  222:          add      bx,ax
[TL1]  223:          shl      ax,1
[TL1]  224:          shl      ax,1
[FE4]  225:          add      bx,ax       ;bx=Zeiger auf
[FG2]  226:                                ;Zeichenbitmap
[DG4]  227:          mov      dl,dh       ;dx=Farbmaske
[ME2]  228:          mov      ax,[bx]
[AS1]  229:          and      ax,dx
[KW2]  230:          mov      es:[di],ax
[9L1]  231:          add      di,8Ø
[SF2]  232:          mov      ax,[bx+2]
[AS1]  233:          and      ax,dx
[GX2]  234:          mov      es:[si],ax
[7M1]  235:          add      si,8Ø
```

```
[UF2]  236:          mov      ax,[bx+4]
[AS1]  237:          and      ax,dx
[KW2]  238:          mov      es:[di],ax
[9L1]  239:          add      di,8Ø
[WF2]  240:          mov      ax,[bx+6]
[AS1]  241:          and      ax,dx
[GX2]  242:          mov      es:[si],ax
[YF2]  243:          mov      ax,[bx+8]
[AS1]  244:          and      ax,dx
[KW2]  245:          mov      es:[di],ax
[5C4]  246:          POPEN    <SI,DI,ES,DX,BX,AX>
[6G1]  247:          add      al,2
[3PØ]  248:          ret
[MZ1]  249: CHAR8x5 ENDP
[ØØØ]  250:
[C83]  251: GrafikModus      PROC
[DN1]  252:          mov      ax,4
[YH1]  253:          int      1Øh
[3PØ]  254:          ret
[B63]  255: GrafikModus      ENDP
[ØØØ]  256:
[KØ3]  257: TextModus        PROC
[CN1]  258:          mov      ax,3
[YH1]  259:          int      1Øh
[3PØ]  260:          ret
[HY2]  261: TextModus        ENDP
[ØØØ]  262:
[LW2]  263: HoriLine         PROC
[MF1]  264:          push     si
[2V1]  265:          mov      si,bx
[2H1]  266:          add      si,3
[KT1]  267:          cmp      si,dx
[ED1]  268:          pop      si
[6Y1]  269:          jc       HoriGo
[3PØ]  270:          ret
[525]  271: HoriGo:  PUSHEN   <AX,BX,CX,DX,AX,BX>
[LM1]  272:          shr      bx,1
[LM1]  273:          shr      bx,1
[HV1]  274:          mov      cx,dx
[NM1]  275:          shr      cx,1
[NM1]  276:          shr      cx,1
[XT1]  277:          sub      cx,bx
[FT1]  278:          mov      al,bl
[1G1]  279:          push     cx
[U41]  280:          PIXADR
[2V2]  281:          POPEN    <CX,BX,AX>
[7S1]  282:          xchg     al,ah
[FT1]  283:          mov      al,bl
[5H1]  284:          and      al,3
[ØN3]  285:          mov      bx,offset HLTab
[CR0]  286:          xlat
[3P1]  287:          and      al,ah
[ULØ]  288:          cld
[NXØ]  289:          stosb
[HA1]  290:          dec      cx
[2P2]  291:          jz       HoriLine1
[6T1]  292:          mov      al,ah
[8V1]  293:          rep      stosb
[ES1]  294: HoriLine1:
[HT1]  295:          mov      al,dl
[5H1]  296:          and      al,3
[LT3]  297:          mov      bx,offset HLTab+4
[CR0]  298:          xlat
[3P1]  299:          and      al,ah
[NXØ]  3ØØ:          stosb
[A83]  3Ø1:          POPEN    <DX,CX,BX,AX>
[3PØ]  3Ø2:          ret
[KU2]  3Ø3: HoriLine         ENDP
[ØØØ]  3Ø4:
[WW2]  3Ø5: VertLine         PROC
[5Z3]  3Ø6:          PUSHEN   <AX,BX,CX,DX,BX>
[TH1]  3Ø7:          sub      dh,dl
[FV1]  3Ø8:          xor      cx,cx
[DT1]  3Ø9:          mov      cl,dh
[WV3]  31Ø:          inc      cx       ;cx=Zähler
[L34]  311:          mov      dh,al    ;Farbe merken
[FT1]  312:          mov      al,bl
[5H1]  313:          and      al,3
[VT3]  314:          mov      bx,offset HLTab+8
[CR0]  315:          xlat
[6P1]  316:          and      al,dh
[VT4]  317:          mov      dh,al    ;dh=Ausgabebyte
[TD1]  318:          pop      bx
[LM1]  319:          shr      bx,1
[LM1]  32Ø:          shr      bx,1
[KT3]  321:          mov      al,bl    ;al=Spalte
[5K4]  322:          mov      ah,dl    ;ah=Pixelzeile
[RS1]  323: VertLine1:
[1G1]  324:          push     cx
[XF1]  325:          push     ax
[U41]  326:          PIXADR
[8V2]  327:          mov      es:[di],dh
[RD1]  328:          pop      ax
[2A1]  329:          inc      ah
[VD1]  33Ø:          pop      cx
[UN2]  331:          loop     VertLine1
[A83]  332:          POPEN    <DX,CX,BX,AX>
[3PØ]  333:          ret
[VU2]  334: VertLine         ENDP
[ØØØ]  335:
[P93]  336: WasGetroffen     PROC
[LV1]  337:          xor      bx,bx
[GT1]  338:          mov      bl,al
[PN1]  339:          sub      al,11
[DL1]  34Ø:          shr      al,1
[ZF1]  341:          push     bx
[U41]  342:          PIXADR
[TD1]  343:          pop      bx
[8S3]  344: WG1:     mov      al,es:[di]
[SR4]  345:          test     al,ØfØh     ;1.Halbbyte=Ø ?
[M65]  346:          jnz      WGEnde      ;nein -> Treffer
[MB1]  347:          inc      bx
[4B4]  348:          or       al,al       ;2.Halbbyte=Ø ?
[242]  349:          jnz      WGEnde
[BA1]  35Ø:          inc      di
[MB1]  351:          inc      bx
[RV1]  352:          cmp      bx,151
[4C1]  353:          jc       WG1
[691]  354:          dec      di
[7Ø2]  355: WGEnde: ret
[N73]  356: WasGetroffen     ENDP
[ØØØ]  357:
[4MØ]  358:          END
```

Listing 1. »Ggcga.asm« enthält Routinen zur Steuerung der CGA-Karte.

```
[8N3]    1: ; Programm: ggherc.asm
[AY6]    2: ; Funktion: Grafikroutine Hercules-Modus
[CD3]    3: ;           (720x348 2 Farben)
[913]    4: ; Sprache:  TASM 2.Ø1
[VT3]    5: ; Autor:    Detlef Paulin
[7A4]    6: ; (c) 1992 Widuch GmbH & Co. KG
[ØØØ]    7:
[XY1]    8:          .MODEL SMALL
[T74]    9: video                 equ     ØbØØØh
[VR4]   10: zeilenkorrektur       equ     Ø
[MU4]   11: spaltenkorrektur      equ     5
[ØØØ]   12:
```

```
[2SØ]   13:          .DATA
[ØØØ]   14:
[S16]   15: ; Tabelle für Schalten in Grafikmodus
[ØØØ]   16:
[4N6]   17: grtable db  35h,2dh,2eh,Ø7h,5bh,Ø2h,57h,57h
[HX4]   18:         db  Ø2h,Ø3h,ØØh,ØØh,ØØh,ØØh,ØØh,ØØh
[ØØØ]   19:
[CØ6]   2Ø: ; Tabelle für Schalten in Textmodus
[ØØØ]   21:
[LK6]   22: txtable db  61h,5Øh,52h,Øfh,19h,Ø6h,19h,19h
[L25]   23:         db  Ø2h,Ødh,Øbh,Øch,ØØh,ØØh,ØØh,ØØh
[ØØØ]   24:
[MD7]   25: ; Tabelle zur Umwandlung der CGA-spezifisch
[647]   26: ; vorliegenden Sprites in die für Hercules-
[6G6]   27: ; grafik notwendigen (Byte mit dem Wert n
[CE6]   28: ; wird durch das n-te Byte aus BR ersetzt)
[ØØØ]   29:
[425]   3Ø: BR      db  ØØØH,ØØ3H,ØØ3H,ØØ3H,ØØCH,ØØFH
[UD4]   31:         db  ØØFH,ØØFH,ØØCH,ØØFH,ØØFH,ØØFH
[YA4]   32:         db  ØØCH,ØØFH,ØØFH,ØØFH,Ø3ØH,Ø33H
[EC4]   33:         db  Ø33H,Ø33H,Ø3CH,Ø3FH,Ø3FH,Ø3FH
[VE4]   34:         db  Ø3CH,Ø3FH,Ø3FH,Ø3FH,Ø3CH,Ø3FH
[F94]   35:         db  Ø3FH,Ø3FH,Ø3ØH,Ø33H,Ø33H,Ø33H
[VE4]   36:         db  Ø3CH,Ø3FH,Ø3FH,Ø3FH,Ø3CH,Ø3FH
[DF4]   37:         db  Ø3FH,Ø3FH,Ø3CH,Ø3FH,Ø3FH,Ø3FH
[XB4]   38:         db  Ø3ØH,Ø33H,Ø33H,Ø33H,Ø3CH,Ø3FH
[DF4]   39:         db  Ø3FH,Ø3FH,Ø3CH,Ø3FH,Ø3FH,Ø3FH
[CF4]   4Ø:         db  Ø3CH,Ø3FH,Ø3FH,Ø3FH,ØCØH,ØC3H
[YM4]   41:         db  ØC3H,ØC3H,ØCCH,ØCFH,ØCFH,ØCFH
[BR4]   42:         db  ØCCH,ØCFH,ØCFH,ØCFH,ØCCH,ØCFH
[XK4]   43:         db  ØCFH,ØCFH,ØFØH,ØF3H,ØF3H,ØF3H
[YS4]   44:         db  ØFCH,ØFFH,ØFFH,ØFFH,ØFCH,ØFFH
[GT4]   45:         db  ØFFH,ØFFH,ØFCH,ØFFH,ØFFH,ØFFH
[ØL4]   46:         db  ØFØH,ØF3H,ØF3H,ØF3H,ØFCH,ØFFH
[GT4]   47:         db  ØFFH,ØFFH,ØFCH,ØFFH,ØFFH,ØFFH
[ZN4]   48:         db  ØFCH,ØFFH,ØFFH,ØFFH,ØFØH,ØF3H
[HP4]   49:         db  ØF3H,ØF3H,ØFCH,ØFFH,ØFFH,ØFFH
[YS4]   5Ø:         db  ØFCH,ØFFH,ØFFH,ØFFH,ØFCH,ØFFH
[MK4]   51:         db  ØFFH,ØFFH,ØCØH,ØC3H,ØC3H,ØC3H
[BR4]   52:         db  ØCCH,ØCFH,ØCFH,ØCFH,ØCCH,ØCFH
[XR4]   53:         db  ØCFH,ØCFH,ØCFH,ØCFH,ØCFH,ØCFH
[ØL4]   54:         db  ØFØH,ØF3H,ØF3H,ØF3H,ØFCH,ØFFH
[GT4]   55:         db  ØFFH,ØFFH,ØFCH,ØFFH,ØFFH,ØFFH
[ZN4]   56:         db  ØFCH,ØFFH,ØFFH,ØFFH,ØFØH,ØF3H
[HP4]   57:         db  ØF3H,ØF3H,ØFCH,ØFFH,ØFFH,ØFFH
[YS4]   58:         db  ØFCH,ØFFH,ØFFH,ØFFH,ØFCH,ØFFH
[KL4]   59:         db  ØFFH,ØFFH,ØFØH,ØF3H,ØF3H,ØF3H
[YS4]   6Ø:         db  ØFCH,ØFFH,ØFFH,ØFFH,ØFCH,ØFFH
[GT4]   61:         db  ØFFH,ØFFH,ØFCH,ØFFH,ØFFH,ØFFH
[DH4]   62:         db  ØCØH,ØC3H,ØC3H,ØC3H,ØCCH,ØCFH
[XR4]   63:         db  ØCFH,ØCFH,ØCCH,ØCFH,ØCFH,ØCFH
[1N4]   64:         db  ØCCH,ØCFH,ØCFH,ØCFH,ØFØH,ØF3H
[HP4]   65:         db  ØF3H,ØF3H,ØFCH,ØFFH,ØFFH,ØFFH
[YS4]   66:         db  ØFCH,ØFFH,ØFFH,ØFFH,ØFCH,ØFFH
[KL4]   67:         db  ØFFH,ØFFH,ØFØH,ØF3H,ØF3H,ØF3H
[YS4]   68:         db  ØFCH,ØFFH,ØFCH,ØFFH,ØFFH,ØFFH
[GT4]   69:         db  ØFFH,ØFFH,ØFCH,ØFFH,ØFFH,ØFFH
[ØL4]   7Ø:         db  ØFØH,ØF3H,ØF3H,ØF3H,ØFCH,ØFFH
[GT4]   71:         db  ØFFH,ØFFH,ØFCH,ØFFH,ØFFH,ØFFH
[PC3]   72:         db  ØFCH,ØFFH,ØFFH,ØFFH
[ØØØ]   73:
[TF7]   74: ; Hilfswerte für Zeichnen horizontaler und
[XB3]   75: ; vertikaler Linien
[MK5]   76: HLTab   db      Øffh,3fh,Øfh,Ø3h,Øc0h,Øf0h
[ØE4]   77:         db      Øfch,Øffh,Øc0h,3Øh,Øch,Ø3h
[ØØØ]   78:
[SU5]   79: ; Beschreibung der Funktionen siehe
[S83]   8Ø: ; Programm ggcga.asm
[ØØØ]   81:
[FTØ]   82:          .CODE
[CH3]   83:          GLOBAL   HERC_UP:PROC
[ØØØ]   84:
[A04]   85: PUSHEN   MACRO    RegLeiste
[DA3]   86:          IRP      Reg,RegLeiste
[2S1]   87:          push     Reg
[HPØ]   88:          ENDM
[HPØ]   89:          ENDM
[ØØØ]   9Ø:
[HV3]   91: POPEN    MACRO    RegLeiste
[DA3]   92:          IRP      Reg,RegLeiste
[WN1]   93:          pop      Reg
[HPØ]   94:          ENDM
[HPØ]   95:          ENDM
[ØØØ]   96:
[L42]   97: PIXADR   MACRO
[NV1]   98:          LOCAL    pixØ
[PV1]   99:          xor      cx,cx
[LS1]  1ØØ:          xchg     cl,al
[DG4]  1Ø1:          add      cx,Spaltenkorrektur
[253]  1Ø2:          xor      di,di     ;di=ØH
[ØL3]  1Ø3:          mov      si,4ØØØH  ;si=4ØØØH
[6V4]  1Ø4:          test     ah,1      ;Vorbelegung von
[RG2]  1Ø5:                             ;di,si richtig?
[7H2]  1Ø6:          jz       pixØ      ;ja
[7L5]  1Ø7:          xchg     di,si     ;sonst vertauschen
[B84]  1Ø8:          add      si,9Ø     ;und 2.Zeile+9Ø
[X75]  1Ø9:          xor      ah,1      ;und Bit Ø löschen
[TF5]  11Ø: pixØ:    xchg     al,ah     ;ax=Zeile AND 1
[DR1]  111:          add      cx,ax
[TL1]  112:          shl      ax,1
[TL1]  113:          shl      ax,1
[RT3]  114:          add      cx,ax     ;cx=5*Zeile
[TL1]  115:          shl      ax,1
[2Y3]  116:          add      cx,ax     ;cx=13*Zeile
[TL1]  117:          shl      ax,1
[TL1]  118:          shl      ax,1
[LY3]  119:          add      cx,ax     ;cx=45*Zeile
[4P1]  12Ø:          add      di,cx
[2R1]  121:          add      si,cx
[655]  122:          mov      ax,video  ;es:[di]=1.Zeile
[TX4]  123:          mov      es,ax     ;es:[si]=2.Zeile
[HPØ]  124:          ENDM
[ØØØ]  125:
[W82]  126: HERC_UP PROC
[XF1]  127:          push     ax
[6H4]  128:          mov      ax,offset HERCLeiste
[BR1]  129:          add      ax,cx
[BR1]  13Ø:          add      ax,cx
[BR1]  131:          add      ax,cx
[DU1]  132:          xchg     ax,cx
[RD1]  133:          pop      ax
[LC1]  134:          call     cx
[3PØ]  135:          ret
[712]  136: HERCLeiste:
[PZ2]  137:          jmp      XORmitWORD
[LX2]  138:          jmp      XORmitBYTE
[V82]  139:          jmp      CHAR8x5
[SP2]  14Ø:          jmp      TextModus
[K53]  141:          jmp      GrafikModus
[TG2]  142:          jmp      HoriLine
[2H2]  143:          jmp      VertLine
[W73]  144:          jmp      WasGetroffen
[V62]  145: HERC_UP ENDP
[ØØØ]  146:
[G63]  147: XORmitWORD       PROC
[3H4]  148:          PUSHEN   <AX,DX,ES,DI,SI,BP>
[U41]  149:          PIXADR
[3Ø3]  15Ø:          mov      bp,offset BR
[PT1]  151:          xchg     bp,bx
[6T1]  152:          xor      ch,ch
[7L1]  153:          shr      dh,1
[YX4]  154:          mov      cl,dh    ;cx=Zeilenläufe
[VB2]  155: @@Ø:     push     cx
[PE1]  156:          push     di
[MF1]  157:          push     si
[A35]  158:          mov      cl,dl    ;cx=Spaltenläufe
[MR3]  159: @@1:     mov      ax,ds:[bp]
```

Listing 2. Mit »ggherc.asm« läuft Goldgräber auch auf einer Hercules-Karte.

```
[CR0]  160:          xlat
[3S1]  161:          xchg    ah,al
[CR0]  162:          xlat
[3S1]  163:          xchg    ah,al
[SW2]  164:          xor     es:[di],ax
[YE3]  165:          xor     es:[di+2000H],ax
[MG1]  166:          add     bp,2
[3G1]  167:          add     di,2
[NK1]  168:          loop    @@1
[NT1]  169:          mov     cl,dl
[SR3]  170: @@2:     mov     ax,ds:[bp]
[CR0]  171:          xlat
[3S1]  172:          xchg    ah,al
[CR0]  173:          xlat
[3S1]  174:          xchg    ah,al
[PX2]  175:          xor     es:[si],ax
[WF3]  176:          xor     es:[si+2000H],ax
[MG1]  177:          add     bp,2
[1H1]  178:          add     si,2
[TK1]  179:          loop    @@2
[ED1]  180:          pop     si
[GC1]  181:          pop     di
[8M1]  182:          add     si,90
[AL1]  183:          add     di,90
[VD1]  184:          pop     cx
[HK1]  185:          loop    @@0
[XU1]  186:          mov     bx,bp
[WB4]  187:          POPEN   <BP,SI,DI,ES,DX,AX>
[3P0]  188:          ret
[F43]  189: XORmitWORD       ENDP
[000]  190:
[D43]  191: XORmitBYTE       PROC
[3H4]  192:          PUSHEN  <AX,DX,ES,DI,SI,BP>
[U41]  193:          PIXADR
[303]  194:          mov     bp,offset BR
[PT1]  195:          xchg    bp,bx
[6T1]  196:          xor     ch,ch
[7L1]  197:          shr     dh,1
[YX4]  198:          mov     cl,dh   ;cx=Zeilenläufe
[BC2]  199: @@4:     push    cx
[PE1]  200:          push    di
[MF1]  201:          push    si
[A35]  202:          mov     cl,dl   ;cx=Spaltenläufe
[ØR3]  203: @@5:     mov     al,ds:[bp]
[CR0]  204:          xlat
[MV2]  205:          xor     es:[di],al
[KE3]  206:          xor     es:[di+2000H],al
[WA1]  207:          inc     bp
[BA1]  208:          inc     di
[5L1]  209:          loop    @@5
[NT1]  210:          mov     cl,dl
[4R3]  211: @@6:     mov     al,ds:[bp]
[CR0]  212:          xlat
[KW2]  213:          xor     es:[si],al
[GF3]  214:          xor     es:[si+2000H],al
[WA1]  215:          inc     bp
[9B1]  216:          inc     si
[9L1]  217:          loop    @@6
[ED1]  218:          pop     si
[GC1]  219:          pop     di
[8M1]  220:          add     si,90
[AL1]  221:          add     di,90
[VD1]  222:          pop     cx
[1L1]  223:          loop    @@4
[XU1]  224:          mov     bx,bp
[WB4]  225:          POPEN   <BP,SI,DI,ES,DX,AX>
[3P0]  226:          ret
[C23]  227: XORmitBYTE       ENDP
[000]  228:
[N12]  229: CHAR8x5 PROC
[PG4]  230:          PUSHEN  <AX,BX,DX,ES,DI,SI>
[U41]  231:          PIXADR
[UB3]  232:          or      dh,dh   ;Farbe=Ø?
[7Y2]  233:          jnz     MC1Ø    ;nein
[UK4]  234:          mov     dl,32   ;bei  Farbe=Ø,
[133]  235:                          ;Leerzeichen geben
[VT2]  236: MC1Ø:    xor     ax,ax
[HT1]  237:          mov     al,dl
[YP1]  238:          sub     ax,32
[TL1]  239:          shl     ax,1
[BR1]  240:          add     bx,ax
[TL1]  241:          shl     ax,1
[TL1]  242:          shl     ax,1
[FB4]  243:          add     bx,ax   ;bx=Zeiger auf
[WF3]  244:                          ;Zeichensatztabelle
[ME2]  245:          mov     ax,[bx]
[KW2]  246:          mov     es:[di],ax
[SE3]  247:          mov     es:[di+2000H],ax
[AL1]  248:          add     di,90
[SF2]  249:          mov     ax,[bx+2]
[GX2]  250:          mov     es:[si],ax
[PF3]  251:          mov     es:[si+2000H],ax
[8M1]  252:          add     si,90
[UF2]  253:          mov     ax,[bx+4]
[KW2]  254:          mov     es:[di],ax
[SE3]  255:          mov     es:[di+2000H],ax
[AL1]  256:          add     di,90
[WF2]  257:          mov     ax,[bx+6]
[GX2]  258:          mov     es:[si],ax
[PF3]  259:          mov     es:[si+2000H],ax
[YF2]  260:          mov     ax,[bx+8]
[KW2]  261:          mov     es:[di],ax
[SE3]  262:          mov     es:[di+2000H],ax
[5C4]  263:          POPEN   <SI,DI,ES,DX,BX,AX>
[6G1]  264:          add     al,2
[3P0]  265:          ret
[MZ1]  266: CHAR8x5 ENDP
[000]  267:
[LW2]  268: HoriLine         PROC
[MF1]  269:          push    si
[2V1]  270:          mov     si,bx
[2H1]  271:          add     si,3
[KT1]  272:          cmp     si,dx
[ED1]  273:          pop     si
[6Y1]  274:          jc      HoriGo
[3P0]  275:          ret
[5Z5]  276: HoriGo:  PUSHEN  <AX,BX,CX,DX,AX,BX>
[LM1]  277:          shr     bx,1
[LM1]  278:          shr     bx,1
[HV1]  279:          mov     cx,dx
[NM1]  280:          shr     cx,1
[NM1]  281:          shr     cx,1
[XT1]  282:          sub     cx,bx
[FT1]  283:          mov     al,bl
[1G1]  284:          push    cx
[U41]  285:          PIXADR
[2V2]  286:          POPEN   <CX,BX,AX>
[VN3]  287:          xchg    al,ah       ;ah=Farbe
[FT1]  288:          mov     al,bl
[5H1]  289:          and     al,3
[ØN3]  290:          mov     bx,offset HLTab
[CR0]  291:          xlat
[974]  292:          and     al,ah       ;al=Farbe CGA
[VØ3]  293:          mov     bx,offset BR
[YT4]  294:          xlat                ;al=Farbe Hercules
[UL0]  295:          cld
[CE3]  296:          mov     es:[di+2000H],al
[NX0]  297:          stosb
[HA1]  298:          dec     cx
[2F2]  299:          jz      HoriLine1
[VØ3]  300:          mov     bx,offset BR
[6T1]  301:          mov     al,ah
[CR0]  302:          xlat
[CS1]  303: HoriLineØ:
[CE3]  304:          mov     es:[di+2000H],al
[NX0]  305:          stosb
[FN2]  306:          loop    HoriLineØ
[ES1]  307: HoriLine1:
[HT1]  308:          mov     al,dl
[5H1]  309:          and     al,3
[LT3]  310:          mov     bx,offset HLTab+4
[CR0]  311:          xlat
```

```
[3P1]  312:          and     al,ah
[VØ3]  313:          mov     bx,offset BR
[CR0]  314:          xlat
[CE3]  315:          mov     es:[di+2000H],al
[NX0]  316:          stosb
[A83]  317:          POPEN   <DX,CX,BX,AX>
[3P0]  318:          ret
[KU2]  319: HoriLine         ENDP
[000]  320:
[WW2]  321: VertLine         PROC
[5Z3]  322:          PUSHEN  <AX,BX,CX,DX,BX>
[TR1]  323:          sub     dh,dl
[PV1]  324:          xor     cx,cx
[DT1]  325:          mov     cl,dh
[WV3]  326:          inc     cx      ;cx=Zähler
[L34]  327:          mov     dh,al   ;Farbe merken
[FT1]  328:          mov     al,bl
[5H1]  329:          and     al,3
[VT3]  330:          mov     bx,offset HLTab+8
[CR0]  331:          xlat
[6P1]  332:          and     al,dh
[VØ3]  333:          mov     bx,offset BR
[CR0]  334:          xlat
[VT4]  335:          mov     dh,al   ;dh=Ausgabebyte
[TD1]  336:          pop     bx
[LM1]  337:          shr     bx,1
[LM1]  338:          shr     bx,1
[KT3]  339:          mov     al,bl   ;al=Spalte
[5K4]  340:          mov     ah,dl   ;ah=Pixelzeile
[RS1]  341: VertLine1:
[1G1]  342:          push    cx
[XF1]  343:          push    ax
[U41]  344:          PIXADR
[8V2]  345:          mov     es:[di],dh
[ME3]  346:          mov     es:[di+2000H],dh
[RD1]  347:          pop     ax
[2A1]  348:          inc     ah
[VD1]  349:          pop     cx
[UN2]  350:          loop    VertLine1
[A83]  351:          POPEN   <DX,CX,BX,AX>
[3P0]  352:          ret
[VU2]  353: VertLine         ENDP
[000]  354:
[P93]  355: WasGetroffen     PROC
[LV1]  356:          xor     bx,bx
[GT1]  357:          mov     bl,al
[PN1]  358:          sub     al,11
[DL1]  359:          shr     al,1
[U41]  360:          PIXADR
[8S3]  361: WG1:     mov     al,es:[di]
[VP4]  362:          test    al,ØfØh    ;1.Halbbyte=Ø ?
[PD5]  363:          jnz     WGEnde     ;nein-> getroffen
[MB1]  364:          inc     bx
[6A4]  365:          or      al,al      ;2.Halbbyte=Ø ?
[242]  366:          jnz     WGEnde
[BA1]  367:          inc     di
[MB1]  368:          inc     bx
[RV1]  369:          cmp     bx,151
[4C1]  370:          jc      WG1
[691]  371:          dec     di
[702]  372: WGEnde:  ret
[N73]  373: WasGetroffen     ENDP
[000]  374:
[C83]  375: GrafikModus      PROC
[9Z3]  376:          push    es          ;Rette Reg.
[MF1]  377:          push    ds
[892]  378:          mov     dx,3bfh
[8N1]  379:          mov     al,3
[6V1]  380:          out     dx,al
[ØT1]  381:          xor     ah,ah
[1F5]  382:          mov     al,2        ;Wähle Grafikmodus
[S75]  383:          lea     si,grtable ;SI= Param.-Tab.
[BN1]  384:          mov     bx,Ø
[952]  385:          mov     cx,4000h
[BX4]  386:          call    setmode     ;Modus setzen
[ED1]  387:          pop     ds
[GD1]  388:          pop     es
[3P0]  389:          ret
[B63]  390: GrafikModus      ENDP
[000]  391:
[KØ3]  392: TextModus        PROC
[PF1]  393:          push    es
[MF1]  394:          push    ds
[ZX1]  395:          mov     al,20h
[9N2]  396:          lea     si,txtable
[662]  397:          mov     bx,72Øh
[562]  398:          mov     cx,8ØØh
[NC2]  399:          call    setmode
[ED1]  400:          pop     ds
[GD1]  401:          pop     es
[3P0]  402:          ret
[HY2]  403: TextModus        ENDP
[000]  404:
[562]  405: SetMode PROC
[MF1]  406:          push    ds
[PF1]  407:          push    es
[XF1]  408:          push    ax
[ZF1]  409:          push    bx
[1G1]  410:          push    cx
[GL4]  411:          mov     dx,Ø3b8h    ;Ausgabe: DCP
[6V1]  412:          out     dx,al
[3V1]  413:          mov     ax,ds
[DV4]  414:          mov     es,ax       ;ES= Param.-Tab.
[CJ4]  415:          mov     dx,Ø3b4h    ;Index-Register
[Y74]  416:          mov     cx,16       ;16 Parameter
[B14]  417:          xor     ah,ah       ;Register Ø
[UZ2]  418: param:   mov     al,ah
[S85]  419:          out     dx,al       ;Ausgaberegister
[SB1]  420:          inc     dx
[CV0]  421:          lodsb
[5K4]  422:          out     dx,al       ;Datenausgabe
[KL4]  423:          inc     ah          ;nächster Wert
[LA1]  424:          dec     dx
[RY1]  425:          loop    param
[VD1]  426:          pop     cx
[GC2]  427:          mov     ax,video
[UL0]  428:          cld
[3V1]  429:          mov     es,ax
[KS1]  430:          mov     di,0h
[RD1]  431:          pop     ax
[KZ1]  432: rep      stosw
[382]  433:          mov     dx,Ø3b8h
[RD1]  434:          pop     ax
[CG1]  435:          add     al,8
[6V1]  436:          out     dx,al
[GD1]  437:          pop     es
[ED1]  438:          pop     ds
[3P0]  439:          ret
[442]  440: SetMode ENDP
[000]  441:
[4M0]  442:          END
```

Listing 2. Mit »ggherc.asm« läuft Goldgräber auch auf einer Hercules-Karte.

```
[CF3]    1: ; Programm: ggvga.asm
[RV6]    2: ; Funktion: Grafikroutine für VGA-Modus
[9B3]    3: ;           (320x200 256 Farben)
[913]    4: ; Sprache:  TASM 2.01
[VT3]    5: ; Autor:    Detlef Paulin
[7A4]    6: ; (c) 1992 Widuch GmbH & Co. KG
[000]    7:
[XY1]    8:             .MODEL SMALL
[P74]    9: video                    equ     0A000h
[ZW4]   10: zeilenkorrektur          equ     16
```

```
[AU4]   11: spaltenkorrektur         equ     Ø
[000]   12:
[2S0]   13:             .DATA
[000]   14:
[NK6]   15: ; NR_VGA übergibt Nr. der gewünschten
[CM4]   16: ; Farbpalette für die Ausgabe
[000]   17:
[EC3]   18:             GLOBAL  nr_vga:WORD
[000]   19:
[4B6]   20: ; Tabelle der mögliche Farbpaletten zu
[LV2]   21: ; je 4 Farben (Byte)
[8Z5]   22: ; z.B. werden die Farbwerte Ø,1,2,3 des
[KP6]   23: ; ersten Sprite durch die VGA-Farbwerte
[PH2]   24: ; Ø,6,8,42 ersetzt
[000]   25:
[UA5]   26: Palette db  Ø,6,8,42,Ø,25,6,15,Ø,34,35,36
[664]   27:             db  Ø,25,6,44,Ø,63,27,32,Ø,11,42,32
[H74]   28:             db  Ø,62,43,32,Ø,7,32,60,Ø,37,40,43
[K64]   29:             db  Ø,49,42,7,Ø,51,53,55,Ø,10,12,32
[U54]   30:             db  Ø,13,40,32,Ø,6,32,39,Ø,13,32,23
[524]   31:             db  Ø,44,46,3,Ø,28,4,32,Ø,46,50,33
[B74]   32:             db  Ø,63,44,6,Ø,39,11,15,Ø,23,44,32
[YT3]   33:             db  Ø,48,49,50,Ø,11,29,4,Ø,7,8,9
[414]   34:             db  Ø,40,41,43,Ø,51,41,53,Ø,7,8,10
[YV3]   35:             db  Ø,23,32,44,Ø,2,4,6,Ø,34,37,40
[D64]   36:             db  Ø,7,50,32,Ø,39,44,32,Ø,47,12,11
[3E2]   37:             db  Ø,54,42,28,Ø,51,45,32
[M74]   38:             db  Ø,30,10,63,Ø,15,49,43,Ø,6,37,15
[464]   39:             db  Ø,29,5,49,Ø,40,38,43,Ø,62,14,40
[214]   40:             db  Ø,24,26,10,Ø,42,6,49,Ø,26,2,15
[F74]   41:             db  Ø,48,49,50,Ø,6,20,30,Ø,42,43,44
[Ø74]   42:             db  Ø,55,6,24,Ø,24,39,45,Ø,30,44,43
[KT3]   43:             db  Ø,22,19,4,Ø,54,7,11,Ø,6,61,4
[D74]   44:             db  Ø,25,23,2,Ø,15,15,15,Ø,39,39,39
[Ø74]   45:             db  Ø,12,43,48,Ø,31,32,4,Ø,39,35,42
[HØ3]   46:             db  Ø,57,59,11,Ø,29,45,38
[3Y2]   47:             db  Ø,21,21,21,Ø,48,50,14
[F74]   48:             db  Ø,42,49,63,Ø,6,11,60,Ø,23,23,23
[LY2]   49:             db  Ø,23,23,23,Ø,35,35,35
[RØ3]   50:             db  Ø,37,37,37,Ø,38,38,38
[EE2]   51:             db  Ø,39,39,39,Ø,41,41,41
[X74]   52:             db  Ø,43,43,43,Ø,39,44,28,Ø,14,41,3
[S74]   53:             db  Ø,12,26,44,Ø,43,52,47,Ø,3,32,44
[384]   54:             db  Ø,25,27,29,Ø,11,57,8,Ø,45,48,51
[DW3]   55:             db  Ø,40,24,31,Ø,7,8,9,Ø,44,40,23
[000]   56:
[KB5]   57: ; Die Funktionen entsprechen den
[ND6]   58: ; Erläuterungen im Programm ggcga.asm
[000]   59:
[FT0]   60:             .CODE
[9A3]   61:             GLOBAL  VGA_UP:PROC
[000]   62:
[A04]   63: PUSHEN   MACRO   RegLeiste
[DA3]   64:          IRP     Reg,RegLeiste
[2S1]   65:          push    Reg
[HP0]   66:          ENDM
[HP0]   67:          ENDM
[000]   68:
[HV3]   69: POPEN    MACRO   RegLeiste
[DA3]   70:          IRP     Reg,RegLeiste
[WN1]   71:          pop     Reg
[HP0]   72:          ENDM
[HP0]   73:          ENDM
[000]   74:
[L42]   75: PIXADR   MACRO
[U44]   76:          add     ah,zeilenkorrektur
[PV1]   77:          xor     cx,cx
[ET1]   78:          xor     di,di
[YS1]   79:          mov     ch,ah
[B24]   80:          xchg    di,cx   ;di=256*Zeile
[ZC5]   81:          xchg    cl,ah   ;ax=Spalte cx=Zeile
[XL1]   82:          shl     cx,1
[XL1]   83:          shl     cx,1
[XL1]   84:          shl     cx,1
[XL1]   85:          shl     cx,1
[195]   86:          add     cx,ax   ;cx=16*Zeile+Spalte
[XL1]   87:          shl     cx,1
[PD4]   88:          shl     cx,1    ;cx=64*Zei+4*Spa
[3H4]   89:          add     di,cx   ;di=320*Zei+4*Spa
[6U1]   90:          mov     si,di
[PS1]   91:          add     si,320
[GC2]   92:          mov     ax,video
[3V1]   93:          mov     es,ax
[HP0]   94:          ENDM
[000]   95:
[L12]   96: VGA_UP   PROC
[XF1]   97:          push    ax
[RE4]   98:          mov     ax,offset VGALeiste
[BR1]   99:          add     ax,cx
[BR1]  100:          add     ax,cx
[BR1]  101:          add     ax,cx
[DU1]  102:          xchg    ax,cx
[RD1]  103:          pop     ax
[LC1]  104:          call    cx
[3P0]  105:          ret
[ES1]  106: VGALeiste:
[PZ2]  107:          jmp     XORmitWORD
[LX2]  108:          jmp     XORmitBYTE
[V82]  109:          jmp     CHAR8x5
[SP2]  110:          jmp     TextModus
[K53]  111:          jmp     GrafikModus
[TG2]  112:          jmp     HoriLine
[2H2]  113:          jmp     VertLine
[W73]  114:          jmp     WasGetroffen
[KZ1]  115: VGA_UP   ENDP
[000]  116:
[G63]  117: XORmitWORD       PROC
[3G1]  118:          push    dx
[VK1]  119:          shl     dl,1
[XW2]  120:          call    XORmitBYTE
[XD1]  121:          pop     dx
[3P0]  122:          ret
[F43]  123: XORmitWORD       ENDP
[000]  124:
[D43]  125: XORmitBYTE       PROC
[9Y3]  126:          PUSHEN  <AX,DX,ES,DI,SI>
[U41]  127:          PIXADR
[CY3]  128:          mov     si,offset Palette
[225]  129:          mov     cx,1        ;1=Standardpalette
[273]  130:          xchg    cx,[nr_vga]
[XL1]  131:          shl     cx,1
[XL1]  132:          shl     cx,1
[8U4]  133:          add     si,cx   ;si = Farbpalette
[KX4]  134:          xchg    si,bx   ;bx = Farbpalette
[CT2]  135:                          ;si = Bildtabelle
[PV1]  136:          xor     cx,cx
[H55]  137:          mov     cl,dh   ;cx = Zeilenläufe
[BC2]  138: @@4:     push    cx
[PE1]  139:          push    di
[DE5]  140:          mov     cl,dl   ;cx = Spaltenläufe
[YR2]  141: @@5:     xor     ax,ax
[ZC2]  142:          mov     dh,[si]
[FK1]  143:          shl     dh,1
[4K1]  144:          rcl     al,1
[FK1]  145:          shl     dh,1
[4K1]  146:          rcl     al,1
[CR0]  147:          xlat
[YB4]  148:          xchg    ah,al   ;erstes Pixel
[FK1]  149:          shl     dh,1
[4K1]  150:          rcl     al,1
[FK1]  151:          shl     dh,1
[4K1]  152:          rcl     al,1
[CR0]  153:          xlat
[CF4]  154:          xchg    ah,al   ;ax= Pixelpaar
[SW2]  155:          xor     es:[di],ax
[BA1]  156:          inc     di
[BA1]  157:          inc     di
[GV1]  158:          xor     ax,ax
[FK1]  159:          shl     dh,1
[4K1]  160:          rcl     al,1
[FK1]  161:          shl     dh,1
[4K1]  162:          rcl     al,1
```

```
[CRØ]  163:          xlat
[3S1]  164:          xchg    ah,al
[FK1]  165:          shl     dh,1
[4K1]  166:          rcl     al,1
[FK1]  167:          shl     dh,1
[4K1]  168:          rcl     al,1
[CRØ]  169:          xlat
[3S1]  17Ø:          xchg    ah,al
[SW2]  171:          xor     es:[di],ax
[BA1]  172:          inc     di
[BA1]  173:          inc     di
[9B1]  174:          inc     si
[5L1]  175:          loop    @@5
[GC1]  176:          pop     di
[2R5]  177:          add     di,32Ø  ;nächste Pixelzeile
[VD1]  178:          pop     cx
[1L1]  179:          loop    @@4
[1U1]  18Ø:          xchg    bx,si
[KS3]  181:          POPEN   <SI,DI,ES,DX,AX>
[3PØ]  182:          ret
[C23]  183: XORmitBYTE       ENDP
[ØØØ]  184:
[N12]  185: CHAR8x5 PROC
[PG4]  186:          PUSHEN  <AX,BX,DX,ES,DI,SI>
[U41]  187:          PIXADR
[GV1]  188:          xor     ax,ax
[HT1]  189:          mov     al,dl
[YP1]  19Ø:          sub     ax,32
[TL1]  191:          shl     ax,1
[BR1]  192:          add     bx,ax
[TL1]  193:          shl     ax,1
[TL1]  194:          shl     ax,1
[FB4]  195:          add     bx,ax   ;bx=Zeiger auf
                                     ;Zeichensatztabelle
[CY3]  197:          mov     si,offset palette
[YY1]  198:          add     si,75*4
[1U1]  199:          xchg    bx,si
[9T1]  2ØØ:          mov     al,dh
[PK4]  2Ø1:          and     al,3    ;Farbnr. filtern
[VS3]  2Ø2:          xlat            ;al=Farbbyte
[KN1]  2Ø3:          mov     cx,5
[9E2]  2Ø4: MCØ1:    push    cx
[FE2]  2Ø5:          mov     dx,[si]
[GS1]  2Ø6:          xchg    dl,dh
[HN1]  2Ø7:          mov     cx,4
[HU5]  2Ø8: MCØ11:   xor     bx,bx   ;erstmal Farbe=Ø
[ZL1]  2Ø9:          shl     dx,1
[ZL1]  21Ø:          shl     dx,1
[NA3]  211:          jnc     MCØ2    ;bleibt Ø
[YU4]  212:          mov     bl,al   ;Farbbyte setzen
[8K2]  213: MCØ2:    shl     dx,1
[ZL1]  214:          shl     dx,1
[ØK1]  215:          jnc     MCØ3
[4T1]  216:          mov     bh,al
[YU3]  217: MCØ3:    mov     es:[di],bx
[BA1]  218:          inc     di
[BA1]  219:          inc     di
[7S1]  22Ø:          loop    MCØ11
[9B1]  221:          inc     si
[9B1]  222:          inc     si
[TF4]  223:          add     di,312  ;nächste Zeile
[VD1]  224:          pop     cx
[5N1]  225:          loop    mcØ1
[5C4]  226:          POPEN   <SI,DI,ES,DX,BX,AX>
[6G1]  227:          add     al,2
[3PØ]  228:          ret
[MZ1]  229: CHAR8x5 ENDP
[ØØØ]  23Ø:
[C83]  231: GrafikModus      PROC
[8Z1]  232:          mov     ax,13H
[YH1]  233:          int     1Øh
[3PØ]  234:          ret
[B63]  235: GrafikModus      ENDP
[ØØØ]  236:
[KØ3]  237: TextModus        PROC
[CN1]  238:          mov     ax,3
[YH1]  239:          int     1Øh
[3PØ]  24Ø:          ret
[HY2]  241: TextModus        ENDP
[ØØØ]  242:
[LW2]  243: HoriLine         PROC
[C33]  244:          cmp     dx,bx   ;Anfang>Endpunkt?
[C33]  245:          jnc     HoriGo  ;nein
[3PØ]  246:          ret
[AB5]  247: HoriGo:  PUSHEN  <AX,BX,CX,AX,BX>
[LM1]  248:          shr     bx,1
[LM1]  249:          shr     bx,1
[FT1]  25Ø:          mov     al,bl
[U41]  251:          PIXADR
[UF2]  252:          POPEN   <BX,AX>
[HV1]  253:          mov     cx,dx
[XT1]  254:          sub     cx,bx
[8N5]  255:          inc     cx      ;cx=Schleifenzähler
[TW4]  256:          and     bx,3    ;Adresskorrektur
[3P1]  257:          add     di,bx
[RY3]  258:          mov     bx,offset palette
[335]  259:          add     bx,74*4 ;Linien=Palette 74
[PL4]  26Ø:          and     al,3    ;Farbnr. filtern
[CRØ]  261:          xlat
[UL@]  262:          cld
[8Y1]  263:          rep     stosb
[2V2]  264:          POPEN   <CX,BX,AX>
[3PØ]  265:          ret
[KU2]  266: HoriLine         ENDP
[ØØØ]  267:
[WW2]  268: VertLine         PROC
[T35]  269:          cmp     dh,dl   ;Anfang>Endezeile?
[N33]  27Ø:          jnc     VertGo  ;nein
[3PØ]  271:          ret
[LB5]  272: VertGo:  PUSHEN  <AX,BX,CX,AX,BX>
[LM1]  273:          shr     bx,1
[LM1]  274:          shr     bx,1
[FT1]  275:          mov     al,bl
[5T1]  276:          mov     ah,dl
[U41]  277:          PIXADR
[TD1]  278:          pop     bx
[RD1]  279:          pop     ax
[PV1]  28Ø:          xor     cx,cx
[DT1]  281:          mov     cl,dh
[3S1]  282:          sub     cl,dl
[WV3]  283:          inc     cx      ;cx=Zähler
[BK1]  284:          and     bx,3
[HY4]  285:          add     dl,bx   ;Adresskorrektur
[RY3]  286:          mov     bx,offset palette
[7Z1]  287:          add     bx,74*4
[PL4]  288:          and     al,3    ;Farbnr. filtern
[CRØ]  289:          xlat
[RS1]  29Ø: VertLine1:
[EV2]  291:          mov     es:[di],al
[SR1]  292:          add     di,32Ø
[UN2]  293:          loop    VertLine1
[2V2]  294:          POPEN   <CX,BX,AX>
[3PØ]  295:          ret
[VU2]  296: VertLine         ENDP
[ØØØ]  297:
[P93]  298: WasGetroffen     PROC
[LV1]  299:          xor     bx,bx
[GT1]  3ØØ:          mov     bl,al
[PN1]  3Ø1:          sub     al,11
[DL1]  3Ø2:          shr     al,1
[ZF1]  3Ø3:          push    bx
[U41]  3Ø4:          PIXADR
[TD1]  3Ø5:          pop     bx
[CT3]  3Ø6: WG1:     mov     ax,es:[di]
[2S4]  3Ø7:          or      ax,ax   ;1. und 2. Pixel=Ø?
[CD5]  3Ø8:          jnz     WGEnde  ;nein --> getroffen
[MB1]  3Ø9:          inc     bx
[BA1]  31Ø:          inc     di
[BA1]  311:          inc     di
[BV2]  312:          mov     ax,es:[di]
[CS4]  313:          or      ax,ax   ;3. und 4. Pixel=Ø?
[CD5]  314:          jnz     WGEnde  ;nein --> getroffen
[BA1]  315:          inc     di
[BA1]  316:          inc     di
[MB1]  317:          inc     bx
[RV1]  318:          cmp     bx,151
[4C1]  319:          jc      WG1
[691]  32Ø:          dec     di
[691]  321:          dec     di
[7Ø2]  322: WGEnde:  ret
[N73]  323: WasGetroffen     ENDP
[ØØØ]  324:
[4MØ]  325:          END
```

Listing 3. Die VGA-Karte wird mit »ggvga.asm« ausgenutzt.

```
[E64]    1: ; Programm: ggzsatz.asm
[P67]    2: ; Funktion: selbstdefinierter Zeichensatz
[9L5]    3: ;           (8x5 Pixel), nicht alle Codes
[8Ø3]    4: ;           sind belegt
[913]    5: ; Sprache:  TASM 2.Ø1
[VT3]    6: ; Autor:    Detlef Paulin
[7A4]    7: ; (c) 1992 Widuch GmbH & Co. KG
[ØØØ]    8:
[XY1]    9:   .MODEL SMALL
[2SØ]   1Ø:   .DATA
[EØ3]   11:   GLOBAL  zs8x5:WORD
[5L5]   12: zs8x5   dw  Ø,Ø,Ø,Ø,Ø,Ø              ;Leerzeichen
[VC5]   13:         dw  ØC@ØFH,ØC@ØFH,ØC@ØFH,ØC@ØFH,ØC@ØFH ;!
[275]   14:         dw  ØF@3CH,ØF@3CH,Ø@@@@H,Ø@@@@H,Ø@@@@H ;"
[VS5]   15:         dw  ØF@3CH,ØFCFFH,ØF@3CH,ØFCFFH,ØF@3CH ;#
[FY5]   16:         dw  ØCØFFH,ØC@CFH,ØFCFFH,ØC@ØFH,ØFCFFH ;$
[BC5]   17:         dw  Ø3ØF@H,ØC@FØH,Ø@@@3H,Ø3C@CH,Ø3C3@H ;%
[SR5]   18:         dw  ØC@FFH,ØCCF3H,ØF@3FH,ØC@F3H,Ø3CFFH ;&
[X55]   19:         dw  ØC@@FH,Ø@@3FH,Ø@@@@H,Ø@@@@H,Ø@@@@H ;'
[NB5]   2Ø:         dw  ØF@@3H,Ø@@@FH,Ø@@3CH,Ø@@@FH,ØF@@3H ;
[C95]   21:         dw  Ø@@3FH,ØC@@3H,ØF@@@H,ØC@@3H,Ø@@3FH ;)
[NC5]   22:         dw  Ø3@33H,ØC@@FH,ØC@@FH,ØC@@FH,Ø3@33H ;*
[8N5]   23:         dw  ØF@@3H,ØC@@FH,ØF@@FH,Ø@@3FH,ØFCFFH ;+
[X75]   24:         dw  Ø@@@@H,Ø@@@@H,Ø@@@@H,ØC@@FH,Ø@@3FH ;,
[S85]   25:         dw  Ø@@@@H,Ø@@@@H,ØFCFFH,Ø@@@@H,Ø@@@@H ;-
[R95]   26:         dw  Ø@@@@H,Ø@@@@H,Ø@@@@H,ØC@@FH,ØC@@FH ;.
[YA5]   27:         dw  ØF@@@H,ØC@@3H,Ø@@@FH,Ø@@3CH,Ø@@F@H ;/
[MW5]   28:         dw  ØF@3FH,ØFCFCH,ØFCFCH,ØFCFCH,ØF@3FH ;Ø
[MF5]   29:         dw  ØF@@3H,ØF@@FH,ØF@3FH,ØF@@3H,ØF@@3H ;1
[MR5]   3Ø:         dw  ØF@3FH,ØFCC@H,ØF@@FH,Ø@@3FH,ØFCFFH ;2
[2R5]   31:         dw  ØF@FFH,ØFC@3H,ØF@FFH,ØFC@3H,ØF@FFH ;3
[TX5]   32:         dw  ØFCFCH,ØFCFCH,ØFCFFH,ØFC@@H,ØFC@@H ;4
[CT5]   33:         dw  ØFCFFH,Ø@@FCH,ØF@FFH,ØFC@@H,ØF@FFH ;5
[1T5]   34:         dw  ØF@3FH,Ø@@FCH,ØF@FFH,ØFCFCH,ØF@3FH ;6
[WE5]   35:         dw  ØFCFFH,ØF@@@H,ØC@@3H,Ø@@@FH,Ø@@3CH ;7
[HU5]   36:         dw  ØF@3FH,ØFCFCH,ØF@3FH,ØFCFCH,ØF@3FH ;8
[MS5]   37:         dw  ØF@3FH,ØFCFCH,ØFC3FH,ØFC@@H,ØF@3FH ;9
[M55]   38:         dw  Ø@@@@H,Ø@@@FH,Ø@@@@H,Ø@@@@H,Ø@@@@H ;:
[575]   39:         dw  Ø@@@@H,Ø@@@FH,Ø@@@@H,Ø@@@@H,Ø@@3FH ;;
[BC5]   4Ø:         dw  ØC@@3H,Ø@@@FH,Ø@@3CH,Ø@@@FH,ØC@@3H ;<
[EG5]   41:         dw  Ø@@@@H,ØFCFFH,Ø@@@@H,ØFCFFH,Ø@@@@H ;=
[GA5]   42:         dw  Ø@@@FH,ØC@@3H,ØF@@@H,ØC@@3H,Ø@@@FH ;>
[FG5]   43:         dw  ØF@3FH,ØFCC@H,ØF@@3H,Ø@@@@H,ØC@@FH ;?
[GU5]   44:         dw  ØF@3FH,ØFCCFH,ØFCCFH,Ø@@C@H,ØFC3FH ;@
[V@6]   45:         dw  ØF@3FH,ØFCFCH,ØFCFFH,ØFCFCH,ØFCFCH ;A
[4Y5]   46:         dw  ØF@FFH,ØFCFCH,ØF@FFH,ØFCFCH,ØF@FFH ;B
[YV5]   47:         dw  ØF@3FH,ØFCFCH,Ø@@FCH,ØFCFCH,ØF@3FH ;C
[2Z5]   48:         dw  ØF@FFH,ØFCFCH,ØFCFCH,ØFCFCH,ØF@FFH ;D
[SX5]   49:         dw  ØFCFFH,Ø@@FCH,ØFCFFH,Ø@@FCH,ØFCFFH ;E
[6T5]   5Ø:         dw  ØFCFFH,Ø@@FCH,ØFCFFH,Ø@@FCH,Ø@@FCH ;F
[HS5]   51:         dw  ØF@3FH,Ø@@FCH,ØFCFCH,Ø3CFCH,ØF@3FH ;G
[246]   52:         dw  ØFCFCH,ØFCFCH,ØFCFFH,ØFCFCH,ØFCFCH ;H
[TL5]   53:         dw  ØC@@FH,ØC@@FH,ØC@@FH,ØC@@FH,ØC@@FH ;I
[1F5]   54:         dw  ØF@@3H,ØF@@3H,ØF@@3H,ØF@33H,ØC@@FH ;J
[1X5]   55:         dw  Ø3CFCH,ØF@FCH,ØC@FFH,ØF@FCH,Ø3CFCH ;K
[AR5]   56:         dw  Ø@@FCH,Ø@@FCH,Ø@@FCH,Ø@@FCH,ØFCFFH ;L
[D16]   57:         dw  Ø3CF@H,ØFCFCH,ØFCFFH,ØFCFCH,ØFCFCH ;M
[F16]   58:         dw  ØFCF@H,ØFCFCH,ØFCFFH,ØFCFCH,Ø3CFCH ;N
[KY5]   59:         dw  ØF@3FH,ØFCFCH,ØFCFCH,ØFCFCH,ØF@3FH ;O
[DT5]   6Ø:         dw  ØF@FFH,ØFCFCH,ØF@FFH,Ø@@FCH,Ø@@FCH ;P
[FØ6]   61:         dw  ØF@3FH,ØFCFCH,ØFCFCH,ØF@FFH,ØCC3FH ;Q
[YY5]   62:         dw  ØF@FFH,ØFCFCH,ØF@FFH,ØF@FCH,Ø3CFCH ;R
[XN5]   63:         dw  ØF@3FH,Ø@@FCH,ØF@3FH,ØFC@@H,ØF@3FH ;S
[UR5]   64:         dw  ØFCFFH,ØC@@FH,ØC@@FH,ØC@@FH,ØC@@FH ;T
[R16]   65:         dw  ØFCFCH,ØFCFCH,ØFCFCH,ØFCFCH,ØF@3FH ;U
[NM5]   66:         dw  Ø3CF@H,Ø3CF@H,ØF@3CH,ØC@@FH,Ø@@@3H ;V
[HS5]   67:         dw  Ø3CF@H,Ø3CF3H,ØF@3FH,ØF@3FH,ØC@@CH ;W
[VN5]   68:         dw  Ø3CF@H,ØF@3CH,ØC@@FH,ØF@3CH,Ø3CF@H ;X
[EN5]   69:         dw  Ø3CF@H,ØF@3CH,ØC@@FH,ØC@@FH,ØC@@FH ;Y
[YS5]   7Ø:         dw  ØFCFFH,ØF@@3H,ØC@@FH,Ø@@3FH,ØFCFFH ;Z
[BK5]   71:         dw  ØF@3FH,Ø@@3FH,Ø@@3FH,Ø@@3FH,ØF@3FH ;[
[4E5]   72:         dw  ØF@@FH,Ø@@3CH,Ø@@@FH,ØC@@3H,Ø@@@@H ;\
[PH5]   73:         dw  ØF@3FH,ØF@@3H,ØF@@3H,ØF@@3H,ØF@3FH ;]
[HA5]   74:         dw  Ø@@@3H,ØC@@FH,ØF@3CH,Ø@@@@H,Ø@@@@H ;^
[2C5]   75:         dw  Ø@@@@H,Ø@@@@H,Ø@@@@H,Ø@@@@H,ØFFFFH ;_
[XW6]   76:         ; Pseudografikzeichen für Energieanzeige `
[M76]   77:         dw  ØFFFFH,ØFFFFH,ØFFFFH,ØFFFFH,ØFFFFH ;`
[B@6]   78:         dw  ØF@FFH,ØFF3FH,ØFF3FH,ØFF3FH,ØFFFFH ;a
[KZ5]   79:         dw  ØF@FFH,ØF@FFH,ØF@FFH,ØF@FFH,ØFFFFH ;b
[WU5]   8Ø:         dw  ØFFFFH,ØFF@3H,ØFF@3H,ØFF@3H,ØFFFFH ;c
[6U5]   81:         dw  ØFFFFH,ØFF@@H,ØFF@@H,ØFF@@H,ØFFFFH ;d
[3P5]   82:         dw  ØFFFFH,Ø3F@@H,Ø3F@@H,Ø3F@@H,ØFFFFH ;e
[GN5]   83:         dw  ØFFFFH,Ø@F@@H,Ø@F@@H,Ø@F@@H,ØFFFFH ;f
[ØK5]   84:         dw  ØFFFFH,Ø@3@@H,Ø@3@@H,Ø@3@@H,ØFFFFH ;g
[GH5]   85:         dw  ØFFFFH,Ø@@@@H,Ø@@@@H,Ø@@@@H,ØFFFFH ;h
[S45]   86:         dw  29*5 dup(Ø)             ; nicht belegt
[CM4]   87:         ; Pfeile für Cursortasten
[WD5]   88:         dw  Ø@@@3H,ØC@@FH,ØF@3FH,Ø@@@3H,Ø@@@3H ;A
[AG5]   89:         dw  Ø@@@3H,Ø@@@3H,ØF@3FH,ØC@@FH,Ø@@@3H ;C
[TF5]   9Ø:         dw  Ø@@@3H,ØC@@3H,ØF@3FH,ØC@@3H,Ø@@@3H ;B
[NE5]   91:         dw  Ø@@@3H,Ø@@@FH,ØF@3FH,Ø@@@FH,Ø@@@3H ;D
[W45]   92:         dw  4*5 dup(Ø)              ; nicht belegt
[346]   93:         dw  ØCCCFH,ØF@3FH,ØFCFCH,ØFCFFH,ØFCFCH ;Ä
[V35]   94:         dw  1Ø*5 dup(Ø)             ; nicht belegt
[M16]   95:         dw  ØCCCFH,ØF@3CH,ØFCFCH,ØFCFCH,ØF@3FH ;Ö
[N56]   96:   dw  ØCCCCH,ØFCFCH,ØFCFCH,ØFCFCH,ØF@3FH ;Ü
 [4MØ]  97:   END
```

Listing 4. Ein 5×8 Punkte großer Zeichensatz wird in »ggzsatz.asm« definiert.

```
[S34]    1: ; Programm: gghaupt.asm
[PB4]    2: ; Funktion: Spielprogramm
[913]    3: ; Sprache:  TASM 2.Ø1
[VT3]    4: ; Autor:    Detlef Paulin
[7A4]    5: ; (c) 1992 Widuch GmbH & Co. KG
[ØØØ]    6:
[CD6]    7: ; Speichermodell SMALL verwenden, d.h.,
[N15]    8: ; 64kB Code- und 64kB Data-Bereich
[ØØØ]    9:
[622]   1Ø:          .MODEL  SMALL
[LF4]   11:          .STACK  2ØØh    ;512 Byte Stack
[2SØ]   12:          .DATA
[ØØØ]   13:
```

```
[6Y6]   14: ; Globale Symbole für spätere Linken def.
[ØØØ]   15:
[9W5]   16:          GLOBAL pictab:WORD,ExtraFolge:BYTE
[2K5]   17:          GLOBAL grexplb:WORD, implod:WORD
[6X4]   18:          GLOBAL ich:WORD, MaxRaum:WORD
[8U5]   19:          GLOBAL grundbild:WORD,extraØ1:BYTE
[HP5]   2Ø:          GLOBAL epplogo:WORD,doslogo:WORD
[Z45]   21:          GLOBAL zs8x5:WORD, nr_vga:WORD
[BG4]   22:          GLOBAL TeileBemerkung:BYTE
[ØØØ]   23:
[U16]   24: include ggequ.inc       ;EQU's einlesen
[ØØØ]   25:
[E75]   26: ; Konstante für Textausgaben
[ØØØ]   27:
[KN4]   28: kopf1    db      'LASER  :                '
[C33]   29:          db      'PRÄMIE: @@@@@$'
[7W4]   3Ø: kopf2    db      'ENERGIE:                '
[7S2]   31:          db      'TEMPO :    Ø@$'
[6Y4]   32: kopf3    db      ' DEMONSTRATION      '
[DR4]   33:          db      '(STEUERUNG ZUFÄLLIG)$'
[NV5]   34: kopf4    db      ' F2=SOUND AUS/AN  ANDERE '
[RB3]   35:          db      'TASTE=ABBRUCH$'
[S25]   36: txtg1    db      '    WIEDER AUF DER '
[9G4]   37:          db      'MÜLL-UMLAUFBAHN     $'
[K45]   38: txtg2    db      '         AUF DER MÜLL-'
[5P3]   39:          db      'UMLAUFBAHN          $'
[ØØØ]   4Ø:
[ØB5]   41: ; Puffer für Parameterausgaben
[ØØØ]   42:
[C53]   43: cpunkte db       '14@92$'
[X62]   44: ctempo  dw       Ø
[KE1]   45:         db       '/64$'
[1N3]   46: cpseud  db       '                    $'
[ØØØ]   47:
[LT3]   48: diatab  db       '`abcdefgh'
[MR3]   49: ziffern db       'Ø123456789'
[ØØØ]   5Ø:
[MU6]   51: ; Tabelle der aktuell genutzten Sprites.
[GW6]   52: ; Maximal 64 Sprites gleichzeitig möglich
[ØU4]   53: ; (auf 32 logischen Zeilen je 2).
[8V6]   54: ; Abstand zweier logischer Zeile beträgt
[756]   55: ; 4 Pixelzeilen, d.h., die beiden ersten
[SK6]   56: ; Sprites werden auf Pixelzeile 16, die
[7V5]   57: ; nächsten zwei auf Zeile 2Ø und die
[PK6]   58: ; letzten zwei Sprites auf Pixelzeile 14Ø
[LØ2]   59: ; ausgegeben.
[6W6]   6Ø: ; Ist in einer Zeile nur ein Sprite aktiv,
[SN7]   61: ; so steht dessen Information grundsätzlich
[Ø57]   62: ; an der ersten Stelle. Gegebenenfalls wird
[837]   63: ; die Information von der zweiten Position
[GK6]   64: ; an die erste kopiert (siehe MACRO COPY4
[3Ø3]   65: ; in ggmacros.inc).
[B46]   66: ; Je Sprite werden 6 Informationen in
[9A5]   67: ; insgesamt 1Ø Byte gespeichert.
[LC1]   68: ; Inhalt:
[N36]   69: ;       Byte      aktuelle Spaltenposition
[NU3]   7Ø: ;                 =Ø -> nicht belegt
[ZB5]   71: ;       Byte      aktuelle Spritephase
[SK4]   72: ;       Word      Nummer des Sprite
[E36]   73: ;       Word      Anfangsadresse des Sprites
[EN5]   74: ;       Word      Ausgabezeile (konstant)
[MM5]   75: ;       Word      Platz für spezifische
[9V5]   76: ;                 Information einiger Sprites
[@@@]   77:
[B25]   78: TeileTab        dw   Ø,Ø,Ø,16,Ø,Ø,Ø,Ø,16,Ø
 [3K6]  79: ; 31 Wiederholungen mit Initialisieren
 [4Z3]  8Ø: ; der Ausgabepixelzeile
 [GP2]  81: initw            =       2Ø
 [ØA1]  82:          REPT    31
 [FT4]  83:          dw      Ø,Ø,Ø,initw,Ø,Ø,Ø,Ø,initw,Ø
 [FP3]  84: initw            =       initw+4
 [HP@]  85:          ENDM
 [ØØØ]  86:
 [NL5]  87: ; Puffer für Ausgabe der Sprites
 [ØØØ]  88:
 [UP2]  89: vleer            dw      Ø
 [GR2]  9Ø: mitte            dw      Ø
 [AP2]  91: nleer            dw      Ø
 [8A4]  92: teilpuffer       dw      36 dup(?)
 [ØØØ]  93:
 [827]  94: ; Tabelle für Sprite 'WANDLER' wandelt sich
 [B76]  95: ; bei Treffer in ein anderes Sprite um
 [ØØØ]  96:
 [YH5]  97: RNDwandler       db  4,6,8,1Ø,12,15,16,18,2Ø
 [VW2]  98:                  db  33,38,43,46,50,56,59
 [8C5]  99: ZufallsTab       db  18,2Ø,2Ø,3Ø,3Ø,2,2,2,7
 [ØZ2] 1ØØ:                  db  12,Ø,Ø,49,55,56,15,19
 [X33] 1Ø1:                  db  51,52,35,26,28,49,38,Ø
 [NE2] 1Ø2:                  db  1,22,21,3,3,41,44
 [ØØØ] 1Ø3:
 [9K6] 1Ø4: ; Tasten für entsprechende Bewegungen
 [ØØØ] 1Ø5:
 [B66] 1Ø6: Sprung           dw  Kursorh ;Kursor-hoch
 [3A6] 1Ø7: Gas              dw  Kursorr ;Kursor-rechts
 [KA6] 1Ø8: Bremse           dw  Kursorl ;Kursor-links
 [ØØØ] 1Ø9:
 [T66] 11Ø: ; logische Spalte, wo Sprite getroffen
 [C73] 111: TrefferSpalte    dw  Ø
 [ØØØ] 112:
 [T76] 113: ; Zeiger auf Position des getroffenen
 [544] 114: ; Sprites in der TeileTab
 [BK3] 115: ParameterZeiger dw  Ø
 [ØØØ] 116:
 [F86] 117: ; Ø/1, wenn 1./2. Teil der aktuellen Zeile
 [1T2] 118: ; bearbeitet wird
 [FC3] 119: ErstesZweites    dw  Ø
 [ØØØ] 12Ø:
 [ØK6] 121: ; Parameter für den Goldgräber (ICH)
 [X25] 122: springt      dw  Ø   ;<>Ø -> springt
 [796] 123: hoch         dw  Ø   ;Sprunghöhe in Zeilen
 [ND6] 124: flug         dw  Ø   ;Dauer der Sprungphase
 [L76] 125: gefeuert     dw  Ø   ;<>Ø -> hat geschossen
 [UF6] 126: shiftlks     dw  Ø   ;2/Ø Shift/nicht Shift
 [WK4] 127: tempo        dw  25,Ø ;Tempowert
 [G26] 128: escape       dw  1   ;Ø-> Abbruch durch ESC
 [UM5] 129: gameover     dw  1   ;Ø-> Spiel zu Ende
 [194] 13Ø: energie      dw  MaxEnergie,Ø
 [674] 131: munivorrat   dw  MaxMuni,Ø
 [E46] 132: punkte       dw  Ø,1 ;Punktestand neu/alt
 [LP5] 133: TMax         dw  Ø   ;akt. Maximaltempo
 [VF5] 134: TMin         dw  Ø   ;akt. Minimaltempo
 [8B5] 135: TAkt         dw  Ø   ;akt. Tempo merken
 [ØØØ] 136:
 [FN6] 137: SpielerBest dw  Ø    ;Zeile, für Eingabe
 [4Z1] 138:                      ;des Namens
 [LR6] 139: ; Anzahl geschaffter Extraräume je Spiel
 [BE3] 14Ø: SpielerExtras   dw  Ø
 [ØØØ] 141:
 [WV5] 142: ; Nummer des nächsten Extraraumes
 [UZ2] 143: Extralevel   dw  Ø
 [NN5] 144: ImExtraBild  dw  Ø       ;Ø/1-> nein/ja
 [ØØØ] 145:
 [5Ø7] 146: ; Zeiger auf Position in der Bildertabelle
 [3D4] 147: BildPos      dw      Grundbild
 [ØØØ] 148:
 [8P6] 149: ; Grundbildposition während Extrabild
 [3X2] 15Ø: PosMerker    dw  Ø
 [ØØØ] 151:
 [X96] 152: ; Taktezähler für Ausgabe der Sprites
 [F@3] 153: FolgenTakt   dw  Ø
 [G55] 154: Random       dw  Ø       ;Zufallswert
 [ØR6] 155: IPixZeile    dw  14Ø     ;Pixelzeile des ICH
 [DP6] 156: IPixZeileN   dw  14Ø     ;neue Zeile des ICH
 [2D6] 157: IPhase       dw  Ø       ;Phasennummer des I
            CH
 [986] 158: NR_VGA       dw  Ø       ;Code für VGA-Farbe
            n
```

Listing 5. »Gghaupt.asm« ist das Hauptprogramm für Goldgräber.

```
[Ø00]  159:
[Z84]  160: ; Multiplikationstabelle
[CM4]  161: multi72     dw   0,72,2*72,3*72,4*72
[1C2]  162:             dw   5*72,6*72,7*72
[Ø00]  163:
[YF6]  164: LetzteTaste  dw   Ø      ;gedrückte Taste
[NZ5]  165: SoundSpiel   dw   1      ;Ø/1=Sound aus/an
[Ø00]  166:
[D96]  167: ; Tabelle für verschiedene Geräusche
[D76]  168: ; Aufbau: Tempo,Frequenz,Frequenz...,Ø
[Ø00]  169:
[5R3]  170: knall   dw   2,2000,500,7000,Ø
[4D4]  171: gegnerknall  dw      3,200,620,Ø
[5C5]  172: grexpls dw   4,1000,Ø,4,1900,Ø,4,2800,Ø,4
[ØW3]  173:         dw   3700,Ø,6, 4600,5000,Ø,5,5500
[D24]  174:         dw   6000,Ø,4,6400,7000,Ø,9,7300,Ø
[EP4]  175: klexpls dw   2,500,Ø,1,400,Ø,2,300,Ø
[N73]  176:         dw   1,200,Ø,2,100,Ø,1,900,Ø
[7L2]  177:         dw   2,2000,Ø,1,5000,Ø
[E55]  178: implods dw   1,5000,Ø,2,5000,Ø,3,3000,Ø
[VP3]  179:         dw   4,3000,Ø,5,1000,Ø,6,1000,Ø
[DB2]  180:         dw   6,200,Ø,6,200,Ø
[GZ4]  181: bexpls  dw   2,Ø,2,Ø,2,Ø,2,300,Ø,2,Ø,2,Ø
[4W1]  182:         dw   2,Ø,2,300,Ø
[GL4]  183: mexpls  dw   6,400,800,Ø,8,200,900,Ø
[1S3]  184:         dw   10,1000,5000,Ø,6,800,100,Ø
[ZS3]  185:         dw   8,5000,400,Ø,10,3000,2000,Ø
[C83]  186:         dw   12,1000,Ø,8,500,1000,Ø
[TL5]  187: ExtraSound   dw   10,100,Ø,10,200,Ø,10,300
[SU3]  188:         dw   Ø,10,400,Ø,10,500,Ø,10,600,Ø
[3S3]  189:         dw   10,700,Ø,10,800,Ø,10,900,Ø
[7Z3]  190:         dw   10,1000,Ø,10,1100,Ø,10,1200,Ø
[WZ3]  191:         dw   10,1300,Ø,10,1400,Ø,10,1500,Ø
[HØ4]  192:         dw   10,1600,Ø,10,1700,Ø,10,1800,Ø
[4Ø4]  193:         dw   10,1900,Ø,10,2000,Ø,10,2100,Ø
[XZ3]  194:         dw   10,2200,Ø,10,2300,Ø,10,2400,Ø
[KØ4]  195:         dw   10,2500,Ø,10,2600,Ø,10,2700,Ø
[BØ4]  196:         dw   10,2800,Ø,10,2900,Ø,10,3000,Ø
[KØ4]  197:         dw   10,2800,Ø,10,2600,Ø,10,2400,Ø
[VZ3]  198:         dw   10,2200,Ø,10,2000,Ø,10,1500,Ø
[FU2]  199:         dw   10,1000,Ø,10,3800,Ø
[Ø00]  200:
[374]  201: ; Weitere Textkonstante
[Y56]  202: gamevorbei db '  DAS SPIEL IST VORBEI ! $'
[MA5]  203: gamebreak  db 'SPIEL ABGEBROCHEN !$'
[Ø00]  204:
[U36]  205: NEText   db     'GEBEN SIE IHREN NAMEN EIN'
[Y52]  206:          db     '... ENTER$'
[GR4]  207: TIText   db     'GOLDGRÄBER 3000$'
[SC3]  208: SPText   db     '(S)PIEL$'
[M43]  209: DEText   db     '(D)EMO$'
[TH3]  210: TAText   db     '(T)ASTEN$'
[Ø73]  211: INText   db     '(I)NFO$'
[DG3]  212: BEText   db     '(B)ESTEN$'
[Y93]  213: QUText   db     '(Q)UIT$'
[YR5]  214: COText   db     '(C) 1992 DMV WIDUCH GMBH'
[X12]  215:          db     ' & CO. KG$'
[NB5]  216: FiText   db     'F1          PANIK (MIT EXIT '
[RC2]  217:          db     'ZURÜCK)$'
[FY4]  218: F2Text   db     'F2          SOUND AUS/AN$'
[544]  219: CLText   db     'ß           BREMSE$'
[H74]  220: CUText   db     'Ä           SPRUNG$'
[YM3]  221: CRText   db     'ö           GAS$'
[LC4]  222: SHText   db     'SHIFT       FEUER$'
[PF6]  223: IHilfe   db     ' &çöß=AUSWAHL, ESC=ENDE$'
[WA4]  224: BETitel  db     'BESTENLISTE$'
[395]  225: TATitel  db     'TASTENBELEGUNGEN$'
[Ø00]  226:
[Ø85]  227: ; Variable für die Stringeingabe
[US3]  228: Ipuffer   db       6Ø dup(?)
[NS2]  229: IPosition dw       Ø
[BG2]  230: ILaenge   dw       Ø
[KE2]  231: IBSPos    dw       Ø
[Ø00]  232:
[8W6]  233: ; Konstanten für die Ausgabe der Besten
[MS4]  234: Platz    db       '  2Ø1918171615141312ll'
[663]  235:          db       '1Ø 9 8 7 6 5 4 3 2 1'
[MX4]  236: BLKopf   db       'PLATZ       NAME          '
[SL3]  237:          db       'PRÄMIE   RÄUME$'
[G53]  238: BLAP     db       36 dup(' ')
[441]  239:          db       '$'
[Ø00]  240:
[AH3]  241: ;Tabelle der 2Ø Besten
[DM6]  242: ;Name 18 Byte, Punkte 5 Byte, Anzahl 1 Byte
[SW5]  243: BestenListe  db 'PAULIN              Ø5ØØØ',Ø
[MA1]  244:          REPT     19
[E54]  245:          db       'PAULIN              '  ;Name
[RØ4]  246:          db       'Ø1ØØ'                  ;Punkte
[B65]  247:          db       Ø    ;Anzahl der Extraräume
[HPØ]  248:          ENDM
[Ø00]  249:
[G34]  250: ; Begin des Programmcode
[Ø00]  251:
[FTØ]  252:          .CODE
[Ø00]  253:
[5W4]  254: ; Namen externer Prozeduren
[Ø00]  255:
[5M5]  256:          GLOBAL  CGA_UP:PROC,HERC_UP:PROC
[9A3]  257:          GLOBAL  VGA_UP:PROC
[CR5]  258:          assume  cs:@code,ds:@data,ss:stack
[Ø00]  259:
[ZT4]  260: ; Macro-Vereinbarungen holen
[Ø00]  261:
[LG3]  262: include ggmacros.inc
[Ø00]  263:
[DS2]  264: ; Hauptprogramm
[Ø00]  265:
[FT1]  266: epp      PROC
[NC2]  267:          mov     ax,@data
[1V1]  268:          mov     ds,ax
[PF1]  269:          push    es
[V43]  270:          call    SpeicherFrei
[1A3]  271:          call    ComspecSuchen
[GD1]  272:          pop     es
[CW3]  273:          call    Aufrufparameter
[Ø62]  274:          call    grtest
[US4]  275: start:   mov     cx,GrafikModus
[132]  276:          call    upgraf
[6V2]  277:          call    HauptMenue
[CK4]  278: start1:  mov     cx,TextModus
[132]  279:          call    upgraf
[VY1]  280:          mov     ah,4ch
[3K1]  281:          int     21h
[ER1]  282: epp      ENDP
[Ø00]  283:
[R45]  284: ; eigentliche Programmschleife
[Ø00]  285:
[P23]  286: Hauptmenue      PROC
[Ø00]  287:
[4B5]  288: ; Anfangsbild mit Menü ausgeben
[993]  289: hmAnf:   call    ClrScr
[RC3]  290:          BOX     Ø,7,32Ø,11,ColorM
[ØV3]  291:          BOX     Ø,25,16Ø,127,ColorM
[AW3]  292:          BOX     18Ø,25,14Ø,127,ColorM
[EU3]  293:          BOX     Ø,159,32Ø,13,ColorM
[N14]  294:          TEXT8x5 1Ø,24,TIText,ink2
[4Ø4]  295:          TEXT8x5 35,1Ø,SPText,ink3
[1Ø4]  296:          TEXT8x5 55,1Ø,DEText,ink3
[124]  297:          TEXT8x5 75,1Ø,TAText,ink3
[614]  298:          TEXT8x5 95,1Ø,INText,ink3
[F14]  299:          TEXT8x5 115,1Ø,BEText,ink3
[434]  300:          TEXT8x5 135,1Ø,QUText,ink3
[X04]  301:          TEXT8x5 162,5,COText,ink2
[AT3]  302:          mov     bx,offset epplogo
[5M2]  303:          mov     ax,35*256+53
[HB3]  304:          mov     dx,LogoFormat
[ZL4]  305:          mov     WORD PTR [nr_vga],81
[9D3]  306:          mov     cx,XORmitWORD
[132]  307:          call    upgraf
[BY3]  308:          mov     bx,offset doslogo
```

```
[N32]  309:          add     ax,60*256
[1M4]  310:          mov     WORD PTR [nr_vga],83
[9D3]  311:          mov     cx,XORmitWORD
[132]  312:          call    upgraf
[UU1]  313: hmØ:     WTAST
[Ø00]  314:
[AB7]  315: ; Entsprechend gedrückter Taste verzweigen
[223]  316:          cmp     ax,ESCTaste
[KX1]  317:          jz      hmEnde
[R35]  318:          or      al,2ØH  ;AL=Kleinbuchstabe
[5V1]  319:          cmp     al,'s'
[1M1]  320:          jnz     hm1
[3P2]  321:          call    mp_spiele
[GV1]  322:          jmp     hmAnf
[WR2]  323: hm1:     cmp     al,'d'
[5M1]  324:          jnz     hm2
[WF2]  325:          call    mp_demo
[GV1]  326:          jmp     hmAnf
[ØT2]  327: hm2:     cmp     al,'t'
[9M1]  328:          jnz     hm3
[KS2]  329:          call    mp_steuer
[GV1]  330:          jmp     hmAnf
[ES2]  331: hm3:     cmp     al,'i'
[DM1]  332:          jnz     hm4
[HK2]  333:          call    mp_infos
[GV1]  334:          jmp     hmAnf
[4S2]  335: hm4:     cmp     al,'b'
[HM1]  336:          jnz     hm5
[1S2]  337:          call    mp_besten
[GV1]  338:          jmp     hmAnf
[6T2]  339: hm5:     cmp     al,'q'
[XL1]  340:          jnz     hmØ
[VZ1]  341: hmEnde:  ret
[NØ3]  342: Hauptmenue      ENDP
[Ø00]  343:
[GØ5]  344: ; Ausgabe der Tastenbelegungen
[Ø00]  345:
[233]  346: mp_steuer       PROC
[642]  347:          call    ClrScr
[F14]  348:          TEXT8x5 5,2Ø,TATitel,ink1
[6Z3]  349:          TEXT8x5 3Ø,13,F1Text,ink2
[BZ3]  350:          TEXT8x5 5Ø,13,F2Text,ink2
[XZ3]  351:          TEXT8x5 7Ø,13,CLText,ink2
[A04]  352:          TEXT8x5 9Ø,13,CUText,ink2
[W14]  353:          TEXT8x5 11Ø,13,CRText,ink2
[Y14]  354:          TEXT8x5 13Ø,13,SHText,ink2
[X32]  355:          call    WTaste
[3PØ]  356:          ret
[113]  357: mp_steuer       ENDP
[Ø00]  358:
[7L4]  359: ; Durchführen eines Spiels
[Ø00]  360:
[LØ3]  361: mp_spiele       PROC
[ØN2]  362:          call    SpielInit
[Ø00]  363:
[2E2]  364: ; Bild aufbauen
[Ø00]  365:
[642]  366:          call    CLRSCR
[713]  367:          BOX     Ø,Ø,32Ø,16,ink1
[1U3]  368:          BOX     Ø,163,32Ø,11,ColorM
[TF3]  369:          TEXT8x5 2,1,kopf1,ink1
[9G3]  37Ø:          TEXT8x5 9,1,kopf2,ink1
[DD6]  371:          TEXT8x5 TextZei,TextSp,Txtg2,ColorG
[LH2]  372:          call    TempoOut
[RP2]  373:          call    PunkteOut
[1P2]  374:          call    SchussOut
[HU2]  375:          call    EnergieOut
[G23]  376:          call    IchAusgeben
[Ø00]  377:
[SM4]  378: ; eigentliche Spielschleife
[Ø00]  379:
[FW4]  38Ø: game:    call    TastaturAbfrage
[K43]  381:          call    TeileSetzen
[2U1]  382:          call    Tout
[1W2]  383:          call    Geschossen
[ØT2]  384:          call    IchBewegen
[LH2]  385:          call    TempoOut
[1P2]  386:          call    SchussOut
[HU2]  387:          call    EnergieOut
[RP2]  388:          call    PunkteOut
[BM2]  389:          WARTE   [tempo]
[UØ3]  39Ø:          WARTE   [hightech]
[AC4]  391:          cmp     WORD PTR [escape],Ø
[ZR1]  392:          je      game1Ø
[FT4]  393:          cmp     WORD PTR [gameover],Ø
[WN1]  394:          jne     game
[Ø00]  395:
[653]  396: ; Spiel ist vorbei
[Ø00]  397:
[S43]  398:          call    OverMeldung
[VY1]  399:          jmp     game3Ø
[VB4]  4Ø0: game1Ø:  call    BreakMeldung
[EA3]  4Ø1: game3Ø:  call    CLRSCR
[ØR4]  4Ø2:          call    SpielerInBestenListe
[3PØ]  4Ø3:          ret
[KY2]  4Ø4: mp_spiele       ENDP
[Ø00]  4Ø5:
[2P3]  4Ø6: ; Bestenliste ausgeben
[Ø00]  4Ø7:
[H23]  4Ø8: mp_besten       PROC
[642]  4Ø9:          call    ClrScr
[MY2]  41Ø: mp_bestenSpiel:
[LC3]  411:          BOX     Ø,11,32Ø,9,ColorM
[VU3]  412:          BOX     Ø,19,32Ø,144,ColorM
[XX3]  413:          TEXT8x5  13,1,blkopf,ink1
[134]  414:          TEXT8x5  2,26,BETitel,ink3
[G52]  415:          mov     ax,15Ø1H
[4N4]  416:          mov     si,offset BestenListe
[FS1]  417:          mov     cx,2Ø
[RP2]  418: MPBLØØ:  push    cx
[XF1]  419:          push    ax
[MF1]  42Ø:          push    ds
[GD1]  421:          pop     es
[LT3]  422:          mov     bx,offset Platz
[DR1]  423:          add     bx,cx
[DR1]  424:          add     bx,cx
[ME2]  425:          mov     ax,[bx]
[YA3]  426:          mov     di,offset BLAP
[XY2]  427:          mov     ds:[di+1],ax
[XN4]  428:          mov     BYTE PTR ds:[di+3],'.'
[6G1]  429:          add     di,5
[UL0]  43Ø:          cld
[PM1]  431:          mov     cx,9
[4G2]  432:          rep     movsw
[3G1]  433:          add     di,2
[KN1]  434:          mov     cx,5
[9Y1]  435:          rep     movsb
[CV0]  436:          lodsb
[TL1]  437:          mov     ah,Ø
[AR1]  438:          mov     cl,1Ø
[5C1]  439:          div     cl
[2Z1]  44Ø:          or      ax,3Ø3Øh
[5G1]  441:          add     di,4
[ZYØ]  442:          stosw
[EA4]  443:          mov     BYTE PTR ds:[di],'$'
[RD1]  444:          pop     ax
[XF1]  445:          push    ax
[MF1]  446:          push    si
[ZY3]  447:          TEXT8x5  ah,al,BLAP,ink2
[ED1]  448:          pop     si
[RD1]  449:          pop     ax
[ZF1]  45Ø:          add     ah,7
[VD1]  451:          pop     cx
[ZØ2]  452:          loop    MPBLØØ
[ME3]  453:          mov     ax,[SpielerBest]
[XX4]  454:          or      ax,ax          ;NamensEingabe?
[B43]  455:          jz      MPBL1Ø         ;nein
[3PØ]  456:          ret
[NA3]  457: MPBL1Ø:  call    WTaste
[3PØ]  458:          ret
```

```
[GØ3]  459: mp_besten       ENDP
[Ø00]  46Ø:
[977]  461: ; Anzeige aller Sprites mit Erläuterungen
[Ø00]  462:
[1Z2]  463: mp_infos        PROC
[642]  464:          call    ClrScr
[WY3]  465:          TEXT8x5  166,1,IHilfe,ink1
[2U3]  466:          BOX     Ø,164,32Ø,1Ø,ColorM
[BU3]  467:          BOX     Ø,11,32Ø,13Ø,ColorM
[ZT3]  468:          BOX     Ø,14Ø,32Ø,23,ColorM
[MF1]  469:          push    ds
[GD1]  47Ø:          pop     es
[9W3]  471:          mov     di,offset IPuffer
[952]  472:          mov     ax,2Ø2Øh
[UL0]  473:          cld
[MS1]  474:          mov     cx,13
[KZ1]  475:          rep     stosw
[TT1]  476:          mov     al,'$'
[NXØ]  477:          stosb
[CH3]  478:          mov     cx,64       ;64 Teile
[8U2]  479: mp_iØØ:  push    cx
[RS1]  48Ø:          mov     ax,64
[YK4]  481:          sub     ax,cx       ;ax=Spritenr.
[VB4]  482:          mov     dx,ax       ;in dx merken
[AP4]  483:          mov     word ptr [nr_vga],ax
[TL1]  484:          shl     ax,1
[TL1]  485:          shl     ax,1
[KB3]  486:          mov     bx,offset pictab
[165]  487:          add     bx,ax       ;bx=Spritezeiger
[ME2]  488:          mov     ax,[bx]
[V75]  489:          mov     bx,ax       ;bx=Spriteanfang
[G84]  49Ø:          add     bx,72       ;zur 2.Phase
[Ø63]  491: mp_i Ø4: mov     ax,dx
[ER3]  492:          and     dl,7        ;dl=Spalte
[VL1]  493:          mov     ah,2
[VK1]  494:          shl     dl,1
[4N1]  495:          add     ah,dl
[VK1]  496:          shl     dl,1
[VK1]  497:          shl     dl,1
[7Z4]  498:          add     ah,dl       ;ah=echte Spalte
[V34]  499:          and     al,ØF8H     ;al=Zeile*8
[NK1]  5ØØ:          shl     al,1
[9F4]  5Ø1:          add     al,14       ;al=PixelZeile
[7S1]  5Ø2:          xchg    al,ah
[TH3]  5Ø3:          mov     dx,TeileFormat
[9D3]  5Ø4:          mov     cx,XORmitWORD
[132]  5Ø5:          call    upgraf
[VD1]  5Ø6:          pop     cx
[G42]  5Ø7:          loop    mp_iØØ
[BF4]  5Ø8:          mov     dx,Ø        ;Zeile/Spalte
[991]  5Ø9: mp_i1Ø:
[PV1]  51Ø:          xor     cx,cx
[LV1]  511:          xor     bx,bx
[NT1]  512:          mov     cl,dl
[XL1]  513:          shl     cx,1
[XL1]  514:          shl     cx,1
[XL1]  515:          shl     cx,1
[DR1]  516:          add     bx,cx
[XL1]  517:          shl     cx,1
[XL1]  518:          shl     cx,1
[DR1]  519:          add     bx,cx
[135]  52Ø:          add     bx,4        ;bx=PixelSpalte
[XS1]  521:          mov     ah,dh
[DM1]  522:          mov     cl,4
[LR1]  523:          shl     ah,cl
[XN4]  524:          add     ah,12       ;ah=Pixelzeile
[ZF1]  525:          push    bx
[XF1]  526:          push    ax
[3G1]  527:          push    dx
[CT3]  528:          BOX     bx,ah,32,16,ColorM
[XD1]  529:          pop     dx
[ZL2]  53Ø:          call    mp_itext
[TU2]  531: mp_i Ø5: push    dx
[X32]  532:          call    WTaste
[XD1]  533:          pop     dx
[223]  534:          cmp     ax,ESCTaste
[S23]  535:          je      mp_infosEnde
[7V2]  536:          cmp     ax,Kursorr
[322]  537:          jne     mp_i Ø6
[MA1]  538:          inc     dl
[FH1]  539:          and     dl,7
[9M2]  54Ø:          jmp     mp_inext
[E54]  541: mp_i Ø6: cmp     ax,Kursorl
[622]  542:          jne     mp_i Ø7
[F91]  543:          dec     dl
[FH1]  544:          and     dl,7
[9M2]  545:          jmp     mp_inext
[554]  546: mp_i Ø7: cmp     ax,Kursorh
[922]  547:          jne     mp_i Ø8
[391]  548:          dec     dh
[3H1]  549:          and     dh,7
[9M2]  55Ø:          jmp     mp_inext
[C64]  551: mp_i Ø8: cmp     ax,Kursort
[Ø22]  552:          jne     mp_i Ø5
[8A1]  553:          inc     dh
[3H1]  554:          and     dh,7
[9M2]  555:          jmp     mp_inext
[Ø00]  556:
[RD1]  557:          pop     ax
[TD1]  558:          pop     bx
[CT3]  559:          BOX     bx,ah,32,16,ColorM
[X32]  56Ø:          call    WTaste
[9C2]  561: mp_infosEnde:
[RD1]  562:          pop     ax
[TD1]  563:          pop     bx
[3PØ]  564:          ret
[Ø00]  565:
[2R1]  566: mp_inext:
[RD1]  567:          pop     ax
[TD1]  568:          pop     bx
[3G1]  569:          push    dx
[VH3]  57Ø:          BOX     bx,ah,32,16,paper
[XD1]  571:          pop     dx
[W22]  572:          jmp     mp_i1Ø
[Ø00]  573:
[ER1]  574: mp_itext:
[3G1]  575:          push    dx
[285]  576:          mov     si,offset TeileBemerkung
[8N1]  577:          or      dx,dx
[A42]  578:          je      mp_it ØØ
[FK1]  579:          shl     dh,1
[FK1]  58Ø:          shl     dh,1
[FK1]  581:          shl     dh,1
[EN1]  582:          add     dl,dh
[PV1]  583:          xor     cx,cx
[NT1]  584:          mov     cl,dl
[UL0]  585:          cld
[Y72]  586: mp_it1:  lodsb
[ZN3]  587:          cmp     al,'$'  ;Textende?
[W32]  588:          jne     mp_it1
[V62]  589:          loop    mp_it1
[1C1]  59Ø: mp_it ØØ:
[MF1]  591:          push    si
[5E4]  592:          TEXT8x5 147,1Ø,IPuffer,paper
[TD1]  593:          pop     bx
[623]  594:          TEXT8x5 147,1Ø,,ink3
[XD1]  595:          pop     dx
[3PØ]  596:          ret
[ØX2]  597: mp_infos        ENDP
[Ø00]  598:
[6L4]  599: ; Demonstration vorführen
[897]  6ØØ: ; Tastatureingabe wird zufällig simuliert
[Ø00]  6Ø1:
[C92]  6Ø2: mp_demo PROC
[ØN2]  6Ø3:          call    SpielInit
[642]  6Ø4:          call    CLRSCR
[713]  6Ø5:          BOX     Ø,Ø,32Ø,16,ink1
[XF3]  6Ø6:          TEXT8x5 2,1,kopf3,ink1
[DG3]  6Ø7:          TEXT8x5 9,1,kopf4,ink1
[DD6]  6Ø8:          TEXT8x5 TextZei,TextSp,TxtG2,ColorG
[Y94]  6Ø9:          mov     WORD PTR [tempo],7
[1U3]  61Ø:          BOX     Ø,163,32Ø,11,ColorM
```

```
[G23]  611:          call    IchAusgeben
[635]  612: demostart:       call    DemoTasten
[K43]  613:          call    TeileSetzen
[2U1]  614:          call    Tout
[1W2]  615:          call    Geschossen
[ØT2]  616:          call    IchBewegen
[BN2]  617:          WARTE   [tempo]
[UØ3]  618:          WARTE   [hightech]
[355]  619:          and     WORD PTR [ExtraLevel],15
[AD1]  620:          PLAYKEY
[KØ2]  621:          jnz     demolØ
[MM2]  622:          jmp     demostart
[RW1]  623: demolØ: ret
[ØØØ]  624:
[6N6]  625: ; simulierte Taste zufällig bestimmen
[ØØØ]  626:
[122]  627: Demotasten:
[YV4]  628:          mov     WORD PTR [gefeuert],Ø
[WD1]  629:          RND     14Ø
[MS1]  630:          test    al,59
[YR1]  631:          jnz     dtlØ
[ØW4]  632:          mov     WORD PTR [gefeuert],1
[N72]  633: dtlØ:    RND     65
[MK1]  634:          sub     al,1
[6L1]  635:          jnc     dt2Ø
[GM4]  636:          cmp     WORD PTR [springt],Ø
[2P1]  637:          jnz     dt2Ø
[BC4]  638:          inc     WORD PTR [springt]
[X43]  639:          mov     ax,MinTempo
[KU2]  640:          sub     ax,[tempo]
[HM1]  641:          shr     ax,1
[KM2]  642:          mov     [flug],ax
[HM1]  643:          shr     ax,1
[KB1]  644:          inc     ax
[MM2]  645:          mov     [hoch],ax
[GRØ]  646: dt2Ø:
[3PØ]  647:          ret
[ØØØ]  648:
[B72]  649: mp_demo ENDP
[ØØØ]  650:
[EK6]  651: ; Initialisierung notwendiger Variablen
[ØP4]  652: ; vor dem Start eines Spiels
[ØØØ]  653:
[GZ2]  654: SpielInit       PROC
[GV1]  655:          xor     ax,ax
[CN1]  656:          mov     bx,1
[YG4]  657:          mov     [ErstesZweites],ax
[683]  658:          mov     [springt],ax
[MM2]  659:          mov     [hoch],ax
[3G4]  660:          mov     WORD PTR [tempo],25
[R23]  661:          mov     [tempo+2],ax
[1L5]  662:          mov     WORD PTR [TMax],MaxTempo
[3K5]  663:          mov     WORD PTR [TMIN],MinTempo
[M53]  664:          mov     [escape],bx
[CC3]  665:          mov     [gameover],bx
[793]  666:          mov     dx,MaxEnergie
[963]  667:          mov     [energie],dx
[TH3]  668:          mov     [energie+2],ax
[PU2]  669:          mov     dx,MaxMuni
[S14]  670:          mov     [munivorrat],dx
[Y44]  671:          mov     [munivorrat+2],ax
[583]  672:          mov     [punkte],ax
[NB3]  673:          mov     [punkte+2],bx
[SY3]  674:          mov     [SpielerBest],ax
[4H4]  675:          mov     [SpielerExtras],ax
[RY3]  676:          mov     [ExtraLevel],ax
[BZ3]  677:          mov     [ImExtraBild],ax
[ZC4]  678:          mov     dx,offset Grundbild
[S63]  679:          mov     [BildPos],dx
[W95]  680:          mov     WORD PTR [FolgenTakt],1ØØ
[8X1]  681:          mov     dx,14Ø
[BS3]  682:          mov     [IPixZeile],dx
[MZ3]  683:          mov     [IPixZeileN],dx
[Z53]  684:          mov     [IPhase],ax
[W34]  685:          mov     bx,offset TeileTab
[VS1]  686:          mov     cx,64
[SE3]  687: SIØØ:    mov     [bx],ax
[BM1]  688:          add     bx,1Ø
[2P1]  689:          loop    SIØØ
[3PØ]  690:          ret
[FX2]  691: SpielInit        ENDP
[ØØØ]  692:
[GF5]  693: ; restliche Programmteile holen
[T33]  694: include ggouts.inc
[HV2]  695: include ggenv.inc
[2G3]  696: include ggdivers.inc
[AF3]  697: include ggjumps.inc
[ØØØ]  698:
[YM1]  699:          END     EPP
[4MØ]  7ØØ:          END
```

Listing 5. »Gghaupt.asm« ist das Hauptprogramm für Goldgräber.

```
[NB3]    1: ; Programm: ggr.asm
[GB7]    2: ; Funktion: Rahmenprogramm für die Tabellen
[6P6]    3: ;           des Grundbildes und Extraräume
[913]    4: ; Sprache:  TASM 2.Ø1
[VT3]    5: ; Autor:    Detlef Paulin
[7A4]    6: ; (c) 1992 Widuch GmbH & Co. KG
[ØØØ]    7:
[XY1]    8:          .MODEL SMALL
[4N5]    9: ; EQU-Vereinbarungen zur besseren
[X24]   1Ø: ; Lesbarkeit der Tabellen
[ØØØ]   11:
[DZ2]   12: tnormal equ     Ø,63
[8G4]   13: ende    equ     224+2ØØ*256,ØffffH
[BC2]   14: tx      equ     224
[7B2]   15: t3Ø     equ     192
[7B2]   16: t25     equ     16Ø
[7B2]   17: t2Ø     equ     128
[M52]   18: t15     equ     96
[C32]   19: t9      equ     64
[522]   2Ø: tØ      equ     32
[SX1]   21: t5      equ     Ø
[ØØØ]   22:
[5S6]   23: ; Folgender Block stellt 73 Symbole gØ-g72
[RK5]   24: ; mit gi=i*256 zur Verfügung, um die
[TB5]   25: ; Sprites Nr. Ø - 72 zu bezeichnen
[ØØØ]   26:
[VP2]   27: DEFSYM  MACRO Num
[4G2]   28:          g&Num EQU Num*256
[HPØ]   29:          ENDM
[ØØØ]   3Ø:
[4MØ]   31:          x = Ø
[S51]   32:          REPT 73
[1U1]   33:          DEFSYM %Ø+X
[MUØ]   34:          x=x+1
[HPØ]   35:          ENDM
[ØØØ]   36:
[8H5]   37: ; Bezeichnung für Zufallssprite
[AW2]   38: gz      equ     128*256
[ØØØ]   39:
[2SØ]   4Ø:          .DATA
[8U5]   41:          GLOBAL Grundbild:WORD,extraØ1:BYTE
[M66]   42:          GLOBAL ExtraFolge:BYTE,MaxRaum:WORD
[ØØØ]   43:
[5L6]   44: ; Anzahl der Elemente in der Extrafolge-1
[8B2]   45: MaxRaum     dw   9
[ØØØ]   46:
[PU4]   47: ; Reihenfolge der Extraräume
[NB6]   48: ; - Anzahl der Bytes darf nicht kleiner
[1K2]   49: ;   MaxRaum sein
[Z47]   5Ø: ; - jeder Wert darf nicht größer sein, als
```

```
[UH6]   51: ;   die Anzahl der Extraräume-1, die nach
[ØE4]   52: ;   ExtraØ1 definiert wurden
[ØØØ]   53:
[9N4]   54: ExtraFolge db   Ø,Ø,1,1,Ø,1,1,1,Ø,Ø
[ØØØ]   55:
[K56]   56: ; Tabelle des Grundbildes: enthält die
[Y37]   57: ; Informationen über die Folge der Sprites
[5H5]   58: ; und damit über den Spieleffekt.
[GG6]   59: ; Jeweils ein WORD (2 Byte) enthält eine
[BM5]   6Ø: ; abgeschlossenen Information mit
[PN3]   61: ; folgender Struktur:
[4L3]   62: ; Beispiel 1:      g7+16+t2Ø
[CS5]   63: ; g7 -> Ausgabe des Sprite Nr. 7  (Ø-72)
[EC7]   64: ; Hinweis 1: gz -> Spritenr. wird zufällig
[Ø23]   65: ;                 gewählt
[BE5]   66: ; Hinweis 2: g24-> Tor zu Extraraum
[CS5]   67: ; 16 -> logischen Ausgabezeile 16 (Ø-31)
[TC7]   68: ; t2Ø-> Abstand zum nächste Sprite 2Ø Takte
[YV5]   69: ;       (bzw. log. Spalten) s. obige EQU's
[ØØØ]   7Ø:
[193]   71: ; Beispiel 2:      tx+g5Ø
[5P7]   72: ; tx -> Einfügen einer größeren Abstandes
[XE4]   73: ;       zwischen zwei Sprites
[6D6]   74: ; g5Ø-> Abstand beträgt 5Ø log. Spalten
[P3Ø]   75: ;
[ØØØ]   76:
[VL5]   77: GrundBild  dw   g7+2Ø,g5+3Ø+t25,g5+28+t3Ø
[RA4]   78:     dw  g12+2Ø+t25,g5+3+t3Ø,g12+29+t3Ø
[ZP4]   79:     dw  g67+31+t9,g24+31+t3Ø          ;TorØ1
[T84]   8Ø:     dw  g1+18+t5,g49+2Ø+t5,g1+18+t3Ø
[DC4]   81:     dw  gz+8+t3Ø,g28+3Ø+t3Ø,g7+1+t3Ø
[YE4]   82:     dw  g39+5+t2Ø,g39+3+t2Ø,gØ+1Ø+t15
[1H4]   83:     dw  gØ+14+t15,g17+31+t5,g42+25+t3Ø
[Y94]   84:     dw  g2+2Ø+t9,g2+2Ø+t3Ø,g25+1Ø+t3Ø
[A45]   85:     dw  g37+14+t25,g4Ø+22+t2Ø,tx+g63,g7+16
[W43]   86:     dw  g21+14+tØ,g21+17+t3Ø
[V34]   87:     dw  g24+19+t3Ø                    ;TorØ2
[1V6]   88: ; Hier eventuell weitere Beschreibung des
[V42]   89: ; Grundbildes
[1A1]   9Ø: ;   dw  ...
[ØØØ]   91:
[VX5]   92: ; Endekennzeichen (nicht vergessen!!)
[P71]   93:     dw  ende
[ØØØ]   94:
[RV6]   95: ; Tabelle der Extraräume (Reihenfolge ist
[YT5]   96: ;         gleich Nummer für ExtraFolge)
[EA1]   97: ; Aufbau:
[VØ6]   98: ;   db    39 Byte Text (Großbuchstaben!)
[K45]   99: ;         mit $ als Endekennzeichen
[T37]   1ØØ: ;  dw    Marke ExtraRaum,MinTempo,MaxTempo
[AY6]   1Ø1: ; Hinweis: tnormal steht für Tempo analog
[E04]   1Ø2: ;          dem des Grundbildes
[TV4]   1Ø3: ;          Tempokonstanten von Ø-63
[ØØØ]   1Ø4:
[MP5]   1Ø5: ExtraØ1 db     ' HINTER DEN GOLDBARREN'
[2G3]   1Ø6:         db     ' SIND VIELE HER! $'
[SH2]   1Ø7:         dw     FØØ,tnormal
[P74]   1Ø8:         db     ' HIER HAT EINER SEINE '
[PØ4]   1Ø9:         db     'PRÄMIEN VERLOREN $'
[UH2]   11Ø:         dw     FØ1,tnormal
[B66]   111: ; hier eventuell weitere Extraräume
[K04]   112: ;       db     '...' 39 Byte Text+'$'
[UE6]   113: ;       dw     Start Extraraum,Min,MaxTempo
[ØØØ]   114:
[ZZ6]   115: ; Der Aufbau der Extraräume ist analog zum
[ØV2]   116: ; Grundbildaufbau
[ØØØ]   117:
[B85]   118: FØØ     dw  g37+25+t3Ø,g14+Ø+t3Ø,g49+22+t5
[FH4]   119:        dw  g49+19+t5,g49+22+t2Ø,g2Ø+21+t5
[884]   12Ø:        dw  g2Ø+18+t5,g2Ø+15+tØ,g2Ø+12+t5
[CR3]   121:        dw  g2Ø+9,g2Ø+6,g15+3Ø,ende
[9F5]   122: FØ1    dw  g4Ø+15+t3Ø,g57+3Ø+t25,g2+25+t3Ø
[A94]   123:        dw  g21+17+t5,g21+14+t5,g21+17+tØ
[HD4]   124:        dw  g21+11+t5,g21+14+t5,g21+18+t3Ø
[TA4]   125:        dw  g3Ø+21+t3Ø,g36+29,g3Ø+25,ende
[NE6]   126: ; Hier eventuell Beschreibung weiterer
[A62]   127: ; Extraräume
[ØØØ]   128:
[4MØ]   129:        END
```

Listing 6. Die Verwaltung von Tabellen und Extraräumen übernimmt »ggr.asm«.

```
[2D6]   6Ø:     dw  mc,1,sauerstoff,1,rechner,1,papier,1
[NU5]   61:     dw  stones,1,krone,1,schuh,1,abombe,1
[NB5]   62:     dw  sojus,1,farbtopf,1,tarnkapp,1
[1Ø3]   63:     dw  feder,1,truhe,1
[XW5]   64:     dw  contain,1,loch1,3,loch2,3,fress4,1
[N46]   65:     dw  fress6,1,fress6,1,fress6,1,fress6,1
[ZT1]   66:     dw  fress6,1
[ØØØ]   67:
[TL6]   68: ; Zum Probieren das Sprite 'Radar' (Nr. 4)
[ØØØ]   69:
[HH2]   7Ø: include uØ1.pic
[ØØØ]   71:
[M97]   72: ; Bemerkungen zu den Sprites in der Reihen-
[RR6]   73: ; folge wie PICTAB für Menüpunkt 'INFO'
[ØØØ]   74:
[PM6]   75: TeileBemerkung  db      'KLEINER ASTEROID$'
[EN4]   76:         db      'TEIL DES EIFFELTURMES$'
[XR3]   77:         db      'LEERER CONTAINER$'
[CH5]   78:         db      'SEGMENT EINER RAUMSTATION$'
[ZK4]   79:         db      'DEFEKTE RADARSTATION$'
[TV3]   8Ø:         db      'PERPETUUM MOBILES'
[185]   81:         db      'ANNIHILIEREN SIE DAS DING$'
[5R4]   82:         db      'UNBEMANNTE RAUMKAPSEL$'
[KY4]   83:         db      'ALTES SOJUS-RAUMSCHIFF$'
[ØW3]   84:         db      'IHR FREUND, ODER?$'
[AN3]   85:         db      'EIN KARUSSELL??$'
[3X4]   86:         db      'FRANGÖDISCHER SCHROTT!$'
[R85]   87:         db      'TV-SONDE (LINDENSTRASSE!)$'
[2D4]   88:         db      'UNBRAUCHBARE >BLASE<$'
[GM3]   89:         db      'LAUFENDES ETWAS$'
[SD3]   9Ø:         db      'ALTE RAUMBASIS$'
[4B3]   91:         db      'SPACE SHUTTLE$'
[PF4]   92:         db      'FERNGESTEUERTE SONDE$'
[7G4]   93:         db      'NEUMODISCHER ANTRIEB$'
[5T4]   94:         db      'DIE ERSTE RAUMSTATION$'
[3F4]   95:         db      'ZIELSUCHENDE RAKETE!$'
[6H3]   96:         db      '1ØØ-PUNKTE AKTIE$'
[LS4]   97:         db      'VOLLER TREIBSTOFFTANK$'
[2R4]   98:         db      'VORSICHT! 25Ø KILO BOMBE$'
[ØV4]   99:         db      'TOR IN DIE EXTRARÄUME!$'
[KT4]   1ØØ:        db      'KINDISCHE WUNDERTÜTE$'
[L94]   1Ø1:        db      'FIESER BOMBENWERFER$'
[3Ø4]   1Ø2:        db      'RAKETENTERRORIST$'
[NF3]   1Ø3:        db      'NETTE TRETMINE$'
[5K4]   1Ø4:        db      'WOZU WAR DAS MAL GUT?$'
[Z75]   1Ø5:        db      'SCHIESSENDER KONKURRENT$'
[EA4]   1Ø6:        db      'EHEMALIGE MARSSONDE$'
[DX3]   1Ø7:        db      'SPIONAGESATELLIT$'
[MZ3]   1Ø8:        db      'DAS FLIEGENDE AUGE$'
[1C4]   1Ø9:        db      'SELBSTGEBAUTE SONDE$'
[6L4]   11Ø:        db      'ALTE ANGRIFFSROBOTER$'
[TX4]   111:        db      'GEFÜLLTE VORRATSBASIS$'
[XD4]   112:        db      'EINE LAMPE, ODER WAS?$'
[AØ5]   113:        db      'PERSÖNLICHER COMPUTER$'
[375]   114:        db      'KOMET AUF KOLLISIONSKURS$'
[HH4]   115:        db      'DARAUF SCHIESSEN ALLE$'
[FB4]   116:        db      'WAS ROLLT DENN HIER?$'
[9L3]   117:        db      'RUF DICH MAL AN!$'
[435]   118:        db      'PROGRAMM GOLDGRÄBER 3ØØØ$'
[FE4]   119:        db      'RIESENGROSSER RUBIN$'
[AC4]   12Ø:        db      'MIKROPROZESSOR 8Ø486$'
[ZU4]   121:        db      'SCHWERTER ZU MUNITION$'
[HY4]   122:        db      'BEHALTEN SIE IHREN KOPF!$'
[NV4]   123:        db      'CONTAINER MIT ATOMMÜLL$'
[VØ5]   124:        db      'DER REST VON FORT KNOX!?$'
[RH4]   125:        db      'IT',39,'S NOT A TRICK. IT'
[9Z1]   126:        db       39,'S...$'
[G74]   127:        db      'MTV',39,'S UNPLUGGED$'
[384]   128:        db      'SAUERSTOFFRESERVE$'
[4Z4]   129:        db      'SIMPLER TASCHENRECHNER$'
[1G4]   13Ø:        db      'ECHTES BÜTTENPAPIER$'
[CV4]   131:        db      'CD-SAMMLUNG DER STONES$'
[EG5]   132:        db      'REICHLICH VERZIERTE KRONE$'
[UE4]   133:        db      'UND WO IST DER RECHTE?$'
[ZR4]   134:        db      'DIE LETZTE(?) ATOMBOMBE$'
[SB4]   135:        db      'OLDTIMER-RAUMSCHIFF$'
[ML4]   136:        db      'WO IST DENN DER MALER?$'
[534]   137:        db      'TARNKAPPENFLIEGER$'
[V74]   138:        db      'SPRING AUF DIE FEDER!$'
[9A5]   139:        db      'ENDLICH! DIE SCHATZTRUHE.$'
[DY2]   14Ø:        db      Ø       ;TextEnde
[4MØ]   141:        END
```

Listing 7. »Ggsprit2.asm« enthält exemplarisch die Definition eines Sprites.

```
[554]    1: ; Programm: ggsprit2.asm
[GB7]    2: ; Funktion: Rahmenprogramm für die Tabellen
[Z43]    3: ;           der Sprites
[913]    4: ; Sprache:  TASM 2.Ø1
[VT3]    5: ; Autor:    Detlef Paulin
[7A4]    6: ; (c) 1992 Widuch GmbH & Co. KG
[ØØØ]    7:
[XY1]    8:  .MODEL SMALL
[2SØ]    9:  .DATA
[N76]   1Ø: ; Marken für den Tabellenanfang der
[TD3]   11: ; jeweiligen Sprites
[Z23]   12:  GLOBAL pictab:WORD
[W57]   13:  GLOBAL fels1:WORD,stahl:WORD,contain:WORD
[FX5]   14:  GLOBAL fels4:WORD,uØ2:WORD,uØ3:WORD
[1C5]   15:  GLOBAL uØ4:WORD,uØ5:WORD,uØ6:WORD
[BY6]   16:  GLOBAL uØ7:WORD,uØ8:WORD,uØ9:WORD,ul0:WORD
[CX6]   17:  GLOBAL ul1:WORD,u12:WORD,u13:WORD,u14:WORD
[C76]   18:  GLOBAL ul5:WORD,u16:WORD,missile:WORD
[XV6]   19:  GLOBAL bonusl0Ø:WORD,oel1:WORD,bombe:WORD
[D37]   2Ø:  GLOBAL tor:WORD,wandler:WORD,bomber:WORD
[HK6]   21:  GLOBAL bomberg:WORD,mine:WORD,rad:WORD
[797]   22:  GLOBAL gegner:WORD,sonde:WORD,spindel:WORD
[1G6]   23:  GLOBAL auge:WORD,ufo17:WORD,ufo18:WORD
[MØ7]   24:  GLOBAL basis:WORD,ballon:WORD,komet:WORD
[T77]   25:  GLOBAL computer:WORD,pulsar:WORD,mc:WORD
[437]   26:  GLOBAL treppe:WORD,telefon:WORD,uran:WORD
[W87]   27:  GLOBAL rubin:WORD,rubin:WORD,gold:WORD
[A27]   28:  GLOBAL 18Ø486:WORD,schwert:WORD,schuh:WORD
[1K5]   29:  GLOBAL fallbeil:WORD,camera:WORD
[F56]   3Ø:  GLOBAL sauerstoff:WORD,rechner:WORD
[A47]   31:  GLOBAL papier:WORD,stones:WORD,krone:WORD
[317]   32:  GLOBAL abombe:WORD,sojus:WORD,truhe:WORD
[VY5]   33:  GLOBAL farbtopf:WORD,tarnkapp:WORD
[V47]   34:  GLOBAL leder:WORD,truhe:WORD,contain:WORD
[UV6]   35:  GLOBAL loch1:WORD,loch2:WORD,fress4:WORD
[333]   36:  GLOBAL fress6:WORD
[BG4]   37:  GLOBAL TeileBemerkung:BYTE
[ØØØ]   38:
[1V5]   39: ; Tabelle der verfügbaren Sprites
[EA1]   4Ø: ; Aufbau:
[PG6]   41: ;  dw Marke des Sprites, Anzahl Phasen-1
[UL6]   42: ; Aus dieser Tabelle wird das Sprite zur
[HW4]   43: ; gegebenen Nummer zugeordnet
[8M6]   44: ; Beispiel: tor steht an 25. Stelle, also
[2F5]   45: ;           ist Tor das Sprite mit der
[1B5]   46: ;           Nummer 24 (Zählbeginn bei Ø)
[ØØØ]   47:
[LP6]   48: PICTAB dw fels1,1,stahl,1,contain,1,fels4,1
[9B5]   49:     dw  uØ1,1,uØ2,1,uØ3,1,uØ4,1,uØ5,1,uØ6,1
[YD5]   5Ø:     dw  uØ7,1,uØ8,7,uØ9,1,ulØ,3,ul1,7,u12,3
[1B5]   51:     dw  ul3,1,ul4,1,ul5,3,ul6,1,missile,1
[285]   52:     dw  bonusl0Ø,1,oel1,1,bombe,1,tor,7
[446]   53:     dw  wandler,1,bomber,1,bomberg,1,mine,1
[MF5]   54:     dw  rad,1,gegner,1,sonde,1,spindel,1
[KW4]   55:     dw  auge,1,ufo17,1,ufo18,1,basis,1
[CF6]   56:     dw  ballon,1,computer,1,komet,3,pulsar,3
[ØA5]   57:     dw  treppe,7,telefon,1,diskette,1
[VF4]   58:     dw  rubin,1,18Ø486,1,schwert,1
[2N5]   59:     dw  fallbeil,7,uran,1,gold,1,camera,1
```

```
[8R3]    1: ; Programm: ggouts.inc
[G27]    2: ; Funktion: Routinen zur Spritesausgabe,
[546]    3: ;           Steuerung des Goldgräbers u.a.
[ZF5]    4: ;           für Programm gghaupt.asm
[913]    5: ; Sprache:  TASM 2.Ø1
[VT3]    6: ; Autor:    Detlef Paulin
[7A4]    7: ; (c) 1992 Widuch GmbH & Co. KG
[ØØØ]    8:
[ØØØ]    9:
[9A6]   1Ø: ; Routine führt Bewegung der Sprites
[KX5]   11: ; entsprechend der TeileTabelle aus
[3S4]   12: ; und bearbeitet Kollisionen
[ØØØ]   13:
[KW1]   14: Tout    PROC
[9D4]   15:         mov     bp,offset teiletab-1Ø
[TS5]   16:         mov     WORD PTR [ErstesZweites],1
[GP1]   17: ToutNext:
[AG4]   18:         cmp     bp,offset teiletab+63Ø
[GF2]   19:         jnc     ToutEnde
[AX4]   2Ø:         ;1. <-> 2. Teil je Zeile schalten
[ZS5]   21:         xor     WORD PTR [ErstesZweites],1
[LL1]   22:         add     bp,1Ø
[AT2]   23:         mov     ax,TSpalte
[3L1]   24:         or      al,al
[SY4]   25:         jz      ToutNext ;kein Sprite da
[Y84]   26:         cmp     WORD PTR TInfo,1ØØh
[2C5]   27:         jc      tout02; ;Sprite bearbeiten
[Y84]   28:         sub     WORD PTR TInfo,1ØØh
[RL2]   29:         jmp     ToutNext
[5X4]   3Ø: tout02: mov     cx,[IPixZeile]
[W63]   31:         mov     ax,TPixZeile
[3M1]   32:         sub     cx,8
[ZX2]   33:         ; auf Crash testen
[UT1]   34:         cmp     ax,cx
[HC5]   35:         jc      ToutNormal  ;höher als ICH
[TM1]   36:         add     cx,16
[WT1]   37:         cmp     cx,ax
[9M5]   38:         jc      ToutNormal  ;tiefer als ICH
[CW4]   39:         mov     ax,TSpalte          ;Crash?
[NH3]   4Ø:         mov     ah,ISpalteBis-2
[GR1]   41:         cmp     ah,al
[ØW4]   42:         jc      ToutNormal  ;vor dem ICH
[NM3]   43:         mov     ah,ISpalteVon-7
[MR1]   44:         cmp     al,ah
[3L5]   45:         jc      ToutNormal  ;hinter dem ICH
[YØ6]   46:         call    CrashUP ;BP=ParameterZeiger
[RL2]   47:         jmp     ToutNext
[952]   48: ToutNormal:
[AL2]   49:         call    AusgabeUP
[RL2]   5Ø:         jmp     ToutNext
[UL1]   51: ToutEnde:
```

Listing 8. »Ggouts.inc« erledigt unter anderem die Sprite-Ausgabe und die Kollisionsabfrage.

```
[3PØ]   52:          ret
[HU1]   53: Tout     ENDP
[ØØØ]   54: '
[PV2]   55: ; Sprite ausgeben
[ØØØ]   56:
[283]   57: TeilAusgeben     PROC
[9V2]   58:          mov     ax,TNummer
[7M5]   59:          mov     [nr_vga],ax ;Nr.=VGA-Farbe
[AT2]   60:          mov     ax,TSpalte
[LV1]   61:          xor     bx,bx
[UH4]   62:          mov     bl,ah       ;bl=Phasennr.
[FM1]   63:          and     bl,15
[VL1]   64:          shl     bx,1
[XK3]   65:          mov     si,multi72[bx]
[NC5]   66:          add     si,TPhaseØ   ;si=Zeiger auf
[YH2]   67:                               ;Sprite-BitMap
[6P3]   68:          cmp     al,StartSpalte-8
[VC4]   69:          jnc     TeilKommt    ;ins Bild
[HY3]   70:          cmp     al,StartSpalte-158
[2T4]   71:          jc      TeilGeht     ;aus dem Bild
[L22]   72: TeilNormal:
[PM1]   73:          sub     al,11
[G94]   74:          shr     al,1         ;AL=Spalte
[C53]   75:          mov     ah,TPixZeile
[375]   76:          mov     dx,TeileFormat  ;12x3 Word
[ØV1]   77:          mov     bx,si
[9D3]   78:          mov     cx,XORmitWORD
[132]   79:          call    upgraf
[ZH3]   80:          jmp     TeilAusgegeben
[1U1]   81: TeilKommt:
[ZP3]   82:          mov     ah,StartSpalte+3
[GR1]   83:          sub     ah,al
[1L1]   84:          shr     ah,1
[TV1]   85:          xor     dx,dx
[W03]   86:          mov     [vleer],dx
[CT1]   87:          mov     dl,ah
[VZ2]   88:          mov     [mitte],dx
[HM1]   89:          mov     dl,6
[UR1]   90:          sub     dl,ah
[403]   91:          mov     [nleer],dx
[PN1]   92:          sub     al,11
[DL1]   93:          shr     al,1
[4D3]   94:          jmp     TeilByteweise
[CH1]   95: TeilGeht:
[EA1]   96:          inc     al
[DL1]   97:          shr     al,1
[TV1]   98:          xor     dx,dx
[403]   99:          mov     [nleer],dx
[MT1]   100:         mov     dl,al
[VZ2]   101:         mov     [mitte],dx
[HM1]   102:         mov     dl,6
[2S1]   103:         sub     dl,al
[W03]   104:         mov     [vleer],dx
[BY4]   105:         mov     al,Ø         ;AusgabeSpalte=Ø
[RH2]   106: TeilByteweise:
[YG4]   107:         push    ax           ;Spalte retten
[MF1]   108:         push    ds
[815]   109:         pop     es           ;es=Datensegment
[5H4]   110:         mov     bx,offset teilpuffer
[4U1]   111:         mov     di,bx
[ULØ]   112:         cld
[505]   113:         mov     cx,12        ;12 Pixelzeilen
[RZ3]   114: teilb2: add     si,[vleer]
[1G1]   115:         push    cx
[KW2]   116:         mov     cx,[mitte]
[9Y1]   117:         rep     movsb
[YP2]   118:         add     si,[nleer]
[VD1]   119:         pop     cx
[K22]   120:         loop    teilb2
[LL4]   121:         pop     ax           ;Spalte holen
[C53]   122:         mov     ah,TPixZeile
[MW2]   123:         mov     dx,[mitte]
[4R1]   124:         mov     dh,12
[PC3]   125:         mov     cx,XORmitBYTE
[132]   126:         call    upgraf
[CS2]   127: TeilAusgegeben:
[3PØ]   128:         ret
[163]   129: TeilAusgeben    ENDP
[ØØØ]   130:
[W87]   131: ; Neu kommendes Sprite in die Spritetabelle
[493]   132: ; TeileTab aufnehmen
[N95]   133: ; BL=Ausgabezeile, BH=Teilnummer
[ØØØ]   134:
[GV2]   135: TeilInit        PROC
[BV1]   136:         mov     ax,bx
[GX3]   137:         and     al,31        ;al=Zeile
[3E3]   138:         xor     cx,cx        ;cx=Ø
[CD4]   139:         mov     cl,ah        ;cx=Teilnr.
[934]   140:         mov     ah,ch        ;ax=Zeile
[DG3]   141:         cmp     cx,ZufallsTeil
[9X1]   142:         jnz     TITØØ
[SL5]   143:         push    ax       ;zufälliges Sprite
[ZG3]   144:         RND     31           ;1 aus 32
[HH4]   145:         mov     bx,offset ZufallsTab
[CRØ]   146:         xlat
[EV1]   147:         mov     cx,ax
[RD1]   148:         pop     ax
[N85]   149: TITØØ:  mov     bp,offset TeileTab
[TL1]   150:         shl     ax,1
[TL1]   151:         shl     ax,1
[RG4]   152:         add     bp,ax            ;bp+4*Zeile
[TL1]   153:         shl     ax,1
[TL1]   154:         shl     ax,1
[LP1]   155:         add     bp,ax
[PH4]   156:         ;bp=Offset TeileTab+2Ø*Zeile
[Y54]   157:         cmp     BYTE PTR TSpalte,Ø
[335]   158:         jz      TIT01    ;als 1.Teil setzen
[2I4]   159:         add     bp,1Ø    ;auf 2.Teil setzen
[Y54]   160:         cmp     BYTE PTR TSpalte,Ø
[DD4]   161:         jnz     TITEnde  ;kein Platz
[UV6]   162: TIT01:  mov     TNummer,cx   ;Nr. eintragen
[KS3]   163:         mov     bx,offset pictab
[XL1]   164:         shl     cx,1
[XL1]   165:         shl     cx,1
[235]   166:         add     bx,cx    ;bx=Zeiger PhaseØ
[ME2]   167:         mov     ax,[bx]
[3T2]   168:         mov     TPhaseØ,ax
[ØF5]   169:         mov     cx,[bx+2] ;cx=Phasenanzahl
[CV1]   170:         mov     bx,ax
[273]   171:         sub     bx,InfoLaenge
[1U2]   172:         mov     ax,TWieOft
[YD2]   173:         mov     TInfo,ax
[XL1]   174:         shl     cx,1
[XL1]   175:         shl     cx,1
[XL1]   176:         shl     cx,1
[465]   177:         shl     cx,1     ;cl=16*Phasenanzahl
[5B4]   178:         inc     cx       ;Anfangsphase 1
[UX4]   179:         mov     ch,17Ø   ;Anfangsspalte 17Ø
[DS1]   180:         xchg    cl,ch
[3Y2]   181:         mov     TSpalte,cx
[EE1]   182: TITEnde:
[3PØ]   183:         ret
[FT2]   184: TeilInit        ENDP
[ØØØ]   185:
[ØØØ]   186:
[FH6]   187: ; Aus der Sprite-Folge das nächste holen,
[247]   188: ; wobei der Abstand zwischen zwei Sprites
[AZ6]   189: ; als Taktzahl in [FolgenTakt] steht, d.h.,
[496]   190: ; [FolgenTakt]=Ø -> nächstes Sprite
[ØØØ]   191:
[283]   192: TeileSetzen    PROC
[FY3]   193:         mov     ax,[FolgenTakt]
[ZM1]   194:         or      ax,ax
[HA5]   195:         jz      TSZAnfang    ;neues Sprite
[MG5]   196:         jmp     TeilNochNicht ;noch warten
[6U1]   197: TSZAnfang:
[E45]   198:         mov     bx,[BildPos]  ;BX=Zeiger
[UV2]   199:                                 ;auf Folgesprite
[DC5]   200:         mov     ax,[bx]    ;al=Zeile, ah=Nr
[CH1]   201:         add     bx,2
[P65]   202:         mov     [BildPos],bx   ;Zeiger auf
[AS2]   203:                                 ;nächstes Teil
[DL4]   204:         cmp     ax,0ffffh      ;RaumEnde?
[RC4]   205:         jz      BildEnde       ;Bildende!
[CV1]   206:         mov     bx,ax
[GG5]   207:         cmp     al,224  ;FolgenTakt setzen?
[RU3]   208:         jnc     FTaktSetzen    ;ja
[RY3]   209:         cmp     al,192         ;3Ø Takte?
[M63]   210:         jnc     FTakt3Ø        ;ja
[SY3]   211:         cmp     al,16Ø         ;25 Takte?
[573]   212:         jnc     FTakt25        ;ja
[TY3]   213:         cmp     al,128         ;2Ø Takte?
[H63]   214:         jnc     FTakt2Ø        ;ja
[EX3]   215:         cmp     al,96          ;15 Takte?
[273]   216:         jnc     FTakt15        ;ja
[L44]   217:         cmp     al,64          ;Pause von
[8K1]   218:                                ;1Ø Takten
[E63]   219:         jnc     FTakt1Ø        ;ja
[S74]   220:         cmp     al,32          ;nächstes
[N72]   221:                                ;Teil sofort?
[B43]   222:         jnc     FTaktØ         ;ja
[RG4]   223:         jmp     FTakt5         ;nächstes
[D43]   224:                             ;Teil nach 5 Takten
[D42]   225: ImGrundBild:
[TC4]   226:         mov     ax,offset grundbild
[C63]   227:         mov     [BildPos],ax
[5F2]   228:         jmp     TSZEnde
[K81]   229: FTakt3Ø:
[CS1]   230:         mov     ax,3Ø
[1W1]   231:         jmp     FTSØ1
[391]   232: FTakt25:
[NS1]   233:         mov     ax,25
[1W1]   234:         jmp     FTSØ1
[F81]   235: FTakt2Ø:
[BS1]   236:         mov     ax,2Ø
[1W1]   237:         jmp     FTSØ1
[Ø91]   238: FTakt15:
[MS1]   239:         mov     ax,15
[1W1]   240:         jmp     FTSØ1
[C81]   241: FTakt1Ø:
[AS1]   242:         mov     ax,1Ø
[1W1]   243:         jmp     FTSØ1
[RR3]   244: FTaktØ: call    TeilInit
[LN2]   245:         jmp     TSZAnfang
[MY2]   246: FTakt5: mov     ax,5
[1W1]   247:         jmp     FTSØ1
[N62]   248: FTaktSetzen:
[MT1]   249:         xor     al,al
[8V4]   250:         xchg    al,ah        ;ax=neuer Takt
[3Y3]   251:         mov     [FolgenTakt],ax
[3PØ]   252:         ret
[R35]   253: FTSØ1:  mov     [FolgenTakt],ax
[ØF2]   254:         call    TeilInit
[3PØ]   255:         ret
[5F2]   256: TeilNochNicht:
[8R4]   257:         dec     WORD PTR [FolgenTakt]
[4G1]   258: TSZEnde:
[3PØ]   259:         ret
[GG1]   260: BildEnde:
[ØA5]   261:         mov     WORD PTR [FolgenTakt],2ØØ
[L95]   262:         cmp     WORD PTR [ImExtraBild],Ø
[NY2]   263:         jz      ImGrundBild
[7B3]   264:         ; Extraraum beendet
[G54]   265: BE1Ø:   mov     ax,[MaxRaum]
[8W3]   266:         cmp     ax,[ExtraLevel]
[EE1]   267:         jz      BE15
[1G5]   268:         ; auf nächsten Extraraum stellen
[KT4]   269:         inc     WORD PTR [ExtraLevel]
[VC6]   270: BE15:   inc     WORD PTR [SpielerExtras]
[KG3]   271:         ; wieder im Grundbild
[5B5]   272:         mov     WORD PTR [ImExtraBild],Ø
[TL2]   273:         mov     ax,[TAkt]
[ØL5]   274:         mov     [Tempo],ax     ;altes Tempo
[1L5]   275:         mov     WORD PTR [TMax],MaxTempo
[3K5]   276:         mov     WORD PTR [TMin],MinTempo
[6N3]   277:         mov     ax,[PosMerker]
[A65]   278:         ; auf alte Position im Grundbild
[C63]   279:         mov     [BildPos],ax
[AN3]   280:         mov     bx,offset TxtG1
[GH4]   281:         mov     si,offset Extrasound
[CM1]   282:         add     bx,2Ø
[FS1]   283:         mov     cx,2Ø
[ØE2]   284: BE2Ø:   push    cx
[FA1]   285:         dec     bx
[IZ1]   286:         push    bx
[SL5]   287:         TEXT8x5  TextZei,TextSp,,ColorG
[ØV1]   288:         mov     bx,si
[2Y1]   289:         call    Sound
[1H1]   290:         add     si,2
[TD1]   291:         pop     bx
[VD1]   292:         pop     cx
[WM1]   293:         loop    BE2Ø
[3PØ]   294:         ret
[163]   295: TeileSetzen     ENDP
[ØØØ]   296:
[ØØØ]   297:
[2Ø5]   298: ; Anzeige des aktuellen Tempos
[ØØØ]   299:
[3Y2]   300: TempoOut        PROC
[4W2]   301:         mov     ax,[tempo]
[1Z2]   302:         cmp     ax,[tempo+2]
[GF2]   303:         jz      TEMPOEnde
[R23]   304:         mov     [tempo+2],ax
[VS1]   305:         mov     cx,64
[FU1]   306:         xchg    cx,ax
[UT1]   307:         sub     ax,cx
[ES1]   308:         mov     cx,1Ø
[5C1]   309:         div     cl
[2I1]   310:         or      ax,3Ø3Øh
[H73]   311:         mov     [ctempo],ax
[MY3]   312:         TEXT8x5  9,67,ctempo,ink2
[WS1]   313: TEMPOEnde:
[3PØ]   314:         ret
[2W2]   315: TempoOut        ENDP
[ØØØ]   316:
[SA6]   317: ; Anzeige der bisher erreichten Punkte
[ØØØ]   318:
[713]   319: PunkteOut       PROC
[173]   320:         mov     ax,[punkte]
[L63]   321:         cmp     ax,[punkte+2]
[MP2]   322:         jz      PunkteEnde
[MB3]   323:         mov     [punkte+2],ax
[3R1]   324:         mov     di,1Ø
[ØX1]   325:         mov     dx,1ØØ
[W22]   326:         mov     bx,1ØØØ
[E42]   327:         mov     cx,1ØØØØ
[HB3]   328:         PUSHEN  <di,dx,bx,cx>
[2M1]   329:         mov     di,Ø
[GV1]   330:         mov     dx,ax
[KY3]   331:         mov     bx,offset ziffern
[GH2]   332: PKTØ:   pop     cx
[GW1]   333:         mov     al,255
[4E2]   334: PKT1:   inc     al
[ØU1]   335:         sub     dx,cx
[CR1]   336:         jnc     PKT1
[HR1]   337:         add     dx,cx
[CRØ]   338:         xlat
[1P3]   339:         mov     cpunkte[di],al
[BA1]   340:         inc     di
[MK1]   341:         cmp     di,4
[7U1]   342:         jnz     PKTØ
[HT1]   343:         mov     al,dl
[CRØ]   344:         xlat
[1P3]   345:         mov     cpunkte[di],al
[134]   346:         TEXT8x5  2,67,cpunkte,ink2
[U12]   347: PunkteEnde:
[3PØ]   348:         ret
[6Z2]   349: PunkteOut       ENDP
[ØØØ]   350:
[6N6]   351: ; Anzeige der restlichen Munition (LASER)
[ØB7]   352: ; über Pseudografik des 8x5 Zeichensatzes
[ØØØ]   353:
[H03]   354: SchussOut      PROC
[L04]   355:         mov     ax,[munivorrat]
[504]   356:         cmp     ax,[munivorrat+2]
[YN2]   357:         jz      SchussEnde
[Y44]   358:         mov     [munivorrat+2],ax
[FW1]   359:         call    pseud
[3Y3]   360:         TEXT8x5  2,17,cpseud,ink2
[412]   361: SchussEnde:
[3PØ]   362:         ret
[GY2]   363: SchussOut       ENDP
[ØØØ]   364:
[FZ6]   365: ; Stellt Anzeigestreifen in cpseud bereit
[ØØØ]   366:
[CP2]   367: pseud:  push    ds
[GD1]   368:         pop     es
[Ø74]   369:         mov     si,offset cpseud+16
[1S3]   370:         mov     di,offset cpseud
[ULØ]   371:         cld
[BH1]   372:         add     ax,3
[HM1]   373:         shr     ax,1
[HM1]   374:         shr     ax,1
[CV1]   375:         mov     bx,ax
[2T1]   376:         mov     ah,al
[XG1]   377:         and     ah,7
[LM1]   378:         shr     bx,1
[LM1]   379:         shr     bx,1
[LM1]   380:         shr     bx,1
[2N1]   381:         or      bx,bx
[CG1]   382:         jz      PSØ1
[FV1]   383:         mov     cx,bx
[H23]   384:         mov     al,diatab[8]
[8Y1]   385:         rep     stosb
[WV5]   386: PSØ1:   cmp     di,si      ;alles erledigt?
[DG1]   387:         jz      PSØ2
[8T1]   388:         mov     bl,ah
[E93]   389:         mov     al,diatab[bx]
[NXØ]   390:         stosb
[LL4]   391:         cmp     di,si      ;jetzt fertig?
[5D2]   392:         jz      PSØ2       ;ja
[2V1]   393:         mov     cx,si
[1T1]   394:         sub     cx,di
[713]   395:         mov     al,[diatab]
[8Y1]   396:         rep     stosb
[2P1]   397: PSØ2:   ret
[ØØØ]   398:
[XR6]   399: ; Anzeige des aktuellen Energievorrates
[CL3]   400: ; über Pseudografik
[ØØØ]   401:
[123]   402: EnergieOut     PROC
[843]   403:         mov     ax,[energie]
[ZC3]   404:         cmp     ax,[energie+2]
[UW2]   405:         jz      EnergieEnde
[TH3]   406:         mov     [energie+2],ax
[FW1]   407:         call    pseud
[SY3]   408:         TEXT8x5  9,17,cpseud,ink2
[K22]   409: EnergieEnde:
[3PØ]   410:         ret
[ØØ3]   411: EnergieOut     ENDP
[ØØØ]   412:
[GK5]   413: ; Ausgabe des Golgräberfahrzeuges
[ØØØ]   414:
[WV1]   415: Iout    PROC
[G23]   416:         call    IchAusgeben
[DX3]   417:         mov     ax,[IPixZeileN]
[5B3]   418:         mov     [IPixZeile],ax
[Ø53]   419:         mov     ax,[IPhase]
[KB1]   420:         inc     ax
[DK1]   421:         and     ax,7
[Z53]   422:         mov     [IPhase],ax
[542]   423: IchAusgeben:
[1M4]   424:         mov     word ptr [nr_vga],77
[6L3]   425:         mov     ax,[IPixZeile]
[K93]   426:         mov     ah,ISpalteVon
[AN1]   427:         sub     ah,11
[1L1]   428:         shr     ah,1
[7S1]   429:         xchg    al,ah
[R43]   430:         mov     si,[IPhase]
[GL1]   431:         shl     si,1
[S53]   432:         mov     bx,offset ich
[UE3]   433:         add     bx,multi72[si]
[TH3]   434:         mov     dx,TeileFormat
[9D3]   435:         mov     cx,XORmitWORD
[132]   436:         call    upgraf
[3PØ]   437:         ret
[VT1]   438: Iout    ENDP
[ØØØ]   439:
[E97]   440: ; Auswertung, falls Goldgräber geschossen
[ØØØ]   441:
[H33]   442: Geschossen      PROC
[CU4]   443:         cmp     WORD PTR [gefeuert],Ø
[K22]   444:         jnz     geschØ
[C73]   445:         jmp     GeschossEnde
[UV4]   446: geschØ: mov     ax,[IPixZeile]
[BM3]   447:         mov     ah,ISpalteVon+6
[EA1]   448:         inc     al
[7S1]   449:         xchg    al,ah
[XF1]   450:         push    ax
[3V3]   451:         mov     cx,WasGetroffen
[RK5]   452:         call    upgraf    ;BX=TrefferSpalte
[RD1]   453:         pop     ax
[772]   454:         mov     al,ink3
[XF1]   455:         push    ax
[3V3]   456:         mov     ah,[grafikkarte]
[XA3]   457:         cmp     ah,3            ;VGA?
[YR3]   458:         jnz     geschA          ;nein
[L52]   459:         mov     ax,Ø3Ø3H
[SW2]   460:         xor     es:[di],ax
[6Z2]   461:         xor     es:[di+2],ax
[RD1]   462:         pop     ax
[X22]   463:         jmp     geschB
[YR2]   464: geschA: pop     ax
[MV2]   465:         xor     es:[di],al
[1P4]   466: geschB: PUSHEN  <AX,ES,DI,BX>
[LL2]   467:         call    geschoss3
[HP3]   468:         mov     bx,offset knall
[2Y1]   469:         call    sound
[TD1]   470:         pop     bx
[ZF1]   471:         push    bx
[RV1]   472:         cmp     bx,151
[MK2]   473:         jnc     geschoss1
[MA3]   474:         call    teilgetroffen
[7M2]   475:         jmp     geschoss2
[SS1]   476: geschoss1:
[V71]   477:         WARTE   2
[US1]   478: geschoss2:
[TD1]   479:         pop     bx
[GC1]   480:         pop     di
[GD1]   481:         pop     es
[3V3]   482:         mov     ah,[grafikkarte]
[XA3]   483:         cmp     ah,3            ;VGA?
[4S3]   484:         jnz     geschC          ;nein
[L52]   485:         mov     ax,Ø3Ø3H
[SW2]   486:         xor     es:[di],ax
[6Z2]   487:         xor     es:[di+2],ax
[RD1]   488:         pop     ax
[332]   489:         jmp     geschD
[4S2]   490: geschC: pop     ax
[MV2]   491:         xor     es:[di],al
[DP3]   492: geschD: mov     al,paper
[6M3]   493:         ; Geschosspur löschen
[WS1]   494: geschoss3:
[SR4]   495:         sub     bx,13           ;log.Spalte-13
[6A5]   496:         and     bx,0fffeh       ;Bit Ø löschen
[6P2]   497:         mov     bx,1            ;+2
[XL2]   498:         add     bx,3            ;+3
[734]   499:         mov     dx,2*ISpalteVon-1Ø
[GU1]   500:         xchg    bx,dx
[T23]   501:         mov     cx,HoriLine
[132]   502:         call    upgraf
[5A2]   503: GeschossEnde:
[3PØ]   504:         ret
[G13]   505: Geschossen      ENDP
[ØØØ]   506:
[5R5]   507: ; Testet Bewegungsmöglichkeit des
[182]   508: ; Goldgräbers
[KK5]   509: ; KannIchHoch/KannIchTief liefert
[2Z6]   510: ; AX=Ø bzw.<>Ø,und ZF gesetzt/nicht gesetzt
```

```
[826]  511: ; wenn es nicht geht bzw. wenn es geht
[ØØØ]  512:
[132]  513: KannIchTief:
[6L3]  514:         mov     ax,[IPixZeile]
[SY4]  515:         cmp     ax,14Ø          ;unterer Rand?
[VN3]  516:         jz      TiefGehtNicht ;ja
[EV1]  517:         cmp     ax,13Ø
[VS2]  518:         jnc     TiefGehtEs
[VL1]  519:         sub     ax,4
[P23]  52Ø:         jmp     HochWieTief
[332]  521: KannIchHoch:
[6L3]  522:         mov     ax,[IPixZeile]
[GP4]  523:         cmp     ax,16           ;oberer Rand?
[FN3]  524:         jz      HochGehtNicht ;ja
[TP1]  525:         cmp     ax,2Ø
[7N2]  526:         jz      HochGehtEs
[FV4]  527:         sub     ax,24           ;Mitte testen
[N32]  528: HochWieTief:
[434]  529:         mov     bp,offset TeileTab
[LP1]  53Ø:         add     bp,ax
[TL1]  531:         shl     ax,1
[TL1]  532:         shl     ax,1
[LP1]  533:         add     bp,ax
 [Y54] 534:         cmp     BYTE PTR TSpalte,Ø
 [C15] 535:         jz      HochGehtEs      ;Zeile frei
 [9V2] 536:         mov     ax,TNummer
 [7V4] 537:         cmp     ax,4            ;fester Grund?
 [R93] 538:         jnc     KIHØ1           ;nein
 [AT2] 539:         mov     ax,TSpalte
 [Ø63] 54Ø:         cmp     al,ISpalteBis
 [WK2] 541:         ja      HochGehtEs
 [RL3] 542:         cmp     al,ISpalteVon-9
 [Z73] 543:         jnc     HochGehtNicht
 [6U5] 544: KIHØ1:  add     bp,1Ø           ;stört 2.Teil
 [Y54] 545:         cmp     BYTE PTR TSpalte,Ø
 [HT3] 546:         jz      HochGehtEs      ;nein
 [9V2] 547:         mov     ax,TNummer
 [8V4] 548:         cmp     ax,4            ;festes Grund?
 [TW3] 549:         jnc     HochGehtEs      ;nein
 [AT2] 55Ø:         mov     ax,TSpalte
 [Ø63] 551:         cmp     al,ISpalteBis
 [4Y4] 552:         ja      HochGehtEs      ;2.Teil davor
 [RL3] 553:         cmp     al,ISpalteVon-9
 [Z73] 554:         jnc     HochGehtNicht
 [DØ2] 555: HochGehtEs:
 [TØ2] 556: TiefGehtEs:
 [AN1] 557:         mov     ax,1
 [T22] 558:         jmp     KIHend
 [3E2] 559: HochGehtNicht:
 [FE2] 56Ø: TiefGehtNicht:
 [GV1] 561:         xor     ax,ax
 [4G6] 562: KIHend: or      ax,ax   ;nicht möglich->Z=1
 [3PØ] 563:         ret
 [ØØØ] 564:
 [TM6] 565: ; Bewegung des Goldgräbers durchführen
 [ØØØ] 566:
 [GØ3] 567: IchBewegen      PROC
 [ZK2] 568:         mov     ax,[hoch]
 [CV1] 569:         mov     bx,ax
 [C15] 57Ø:         or      ax,ax       ;Aufstiegsphase?
 [444] 571:         jz      IchNichtHoch    ;nein
 [E13] 572:         call    KannIchHoch
 [UP3] 573:         jz      IchKannNichtHoch
 [6L3] 574:         mov     ax,[IPixZeile]
 [VL1] 575:         sub     ax,4
 [Y95] 576:         mov     [IPixZeileN],ax ;Zeile-4
 [BV1] 577:         mov     ax,bx
 [DA1] 578:         dec     ax
 [A23] 579:         jmp     IchAusgabeH
 [CY2] 58Ø: IchKannNichtHoch:
 [GV1] 581:         xor     ax,ax
 [ØZ5] 582:         mov     [flug],ax  ;oben angestoßen
 [932] 583: IchAusgabeH:
 [MM2] 584:         mov     [hoch],ax
 [AT2] 585:         jmp     IchAusgabe
 [X62] 586: IchNichtHoch:
 [C13] 587:         call    KannIchTief
 [2E5] 588:         mov     [springt],ax  ;Ø/1->nein/ja
 [P15] 589:         jnz     FallFlug      ;kann fallen
 [KM2] 59Ø:         mov     [flug],ax
 [AT2] 591:         jmp     IchAusgabe
 [AK1] 592: FallFlug:
 [U14] 593:         cmp     WORD PTR [flug],Ø
 [BR5] 594:         jz      IchFalle ;keine Sprungphase
 [4V3] 595:         dec     WORD PTR [flug]
 [AT2] 596:         jmp     IchAusgabe
 [GG1] 597: IchFalle:
 [6L3] 598:         mov     ax,[IPixZeile]
 [CH1] 599:         add     ax,4
 [MH5] 6ØØ:         mov     [IPixZeileN],ax    ;Zeile+4
 [RZ1] 6Ø1: IchAusgabe:
 [CT1] 6Ø2:         call    Iout
 [3PØ] 6Ø3:         ret
 [FY2] 6Ø4: IchBewegen      ENDP
 [ØØØ] 6Ø5:
 [BS5] 6Ø6: ; Abfrage der Tastatur im Spiel und
 [3P5] 6Ø7: ; belegen entsprechender Variabler
 [ØØØ] 6Ø8:
 [1M3] 6Ø9: Tastaturabfrage PROC
 [MF1] 61Ø:         push    ds
 [5Z1] 611:         mov     ax,4Øh
 [1V1] 612:         mov     ds,ax
 [KZ1] 613:         mov     bx,17H
 [ME2] 614:         mov     ax,[bx]
 [ED1] 615:         pop     ds
 [8L5] 616:         and     ax,ShiftLinks  ;ShiftLinks?
 [SL4] 617: tast1Ø: mov     [gefeuert],ax
[W61]  618:         jz      Tast0Ø      ;nicht geschossen
 [L04] 619:         mov     ax,[munivorrat]
 [ZM1] 62Ø:         or      ax,ax
 [KX4] 621:         jz      Tast1Ø      ;keine Muni mehr
 [DA1] 622:         dec     ax
 [F14] 623:         mov     [munivorrat],ax
 [GH5] 624:         ; liefert Taste und erledigt F1,F2
 [DN2] 625: Tast0Ø: PLAYKEY
 [SG5] 626:         jnz     Tast11   ;ja,Taste gedrückt
 [8Y3] 627:         jmp     TastEnde        ;nein
[RT6]  628: Tast11: cmp     ax,Sprung    ;Sprung-Taste?
[XN3]  629:         jnz     tast01          ;nein
[GM4]  63Ø:         cmp     WORD PTR [springt],Ø
[RP5]  631:         jnz     Tast21   ;Sprung unmöglich
[ZG4]  632:         cmp     WORD PTR [energie],Ø
[1R4]  633:         jz      Tast21          ;Energie alle
[N74]  634:         dec     WORD PTR [energie]
[BC4]  635:         inc     WORD PTR [springt]
[P64]  636:         ; Sprungkurve ermitteln
[X43]  637:         mov     ax,MinTempo
[KU2]  638:         sub     ax,[tempo]
[HM1]  639:         shr     ax,1
[V1X]  64Ø:         mov     [flug],ax       ;Flugphasen
[HM1]  641:         shr     ax,1
[KB1]  642:         inc     ax
[UB5]  643:         mov     [hoch],ax       ;Steigphasen
[TR3]  644: Tast21: jmp     TastEnde
[TD5]  645: tast01: cmp     ax,Gas                  ;Gas?
[Y84]  646:         jnz     tast02                  ;nein
[ZG4]  647:         cmp     WORD PTR [energie],Ø
[1R4]  648:         jz      Tast21          ;Energie alle
[4W2]  649:         mov     ax,[tempo]
[SE5]  65Ø:         cmp     ax,[TMax]   ;schon Vollgas?
[WV3]  651:         jz      Tast21                  ;ja
[N74]  652:         dec     WORD PTR [energie]
[DA1]  653:         dec     ax
[9Ø3]  654:         mov     [tempo],ax
[7Y5]  655:         ; Tempoänderung bewirkt Änderung
[AX2]  656:         ; der Sprungkurve
[U14]  657:         cmp     WORD PTR [flug],Ø
[Y15]  658:         jz      Tast21      ;nicht im Sprung
[9W3]  659:         inc     WORD PTR [flug]
[FF2]  66Ø:         jmp     TastEnde
[XZ5]  661: tast02: cmp     ax,Bremse               ;Bremse?
[194]  662:         jnz     tast03                  ;nein
[4W2]  663:         mov     ax,[tempo]
```

```
[B95]  664:         cmp     ax,[TMin]    ;bereits Min.?
[E54]  665:         jz      TastEnde                ;ja
[KB1]  666:         inc     ax
[9Ø3]  667:         mov     [tempo],ax
[U14]  668:         cmp     WORD PTR [flug],Ø
[PA2]  669:         jz      TastEnde
[4V3]  67Ø:         dec     WORD PTR [flug]
[FF2]  671:         jmp     TastEnde
[Y96]  672: tastØ3: cmp     ax,ESCTaste         ;Abbruch
[HK4]  673:         jnz     TastEnde                ;nein
[GV1]  674:         xor     ax,ax
[H53]  675:         mov     [escape],ax
[8K1]  676: TastEnde:
[3P0]  677:         ret
[ØK3]  678: TastaturAbfrage ENDP
[ØØØ]  679:
[MB5]  68Ø: ; Ermittelt getroffenes Sprite
[UW6]  681: ; und führt entsprechende Reaktion durch
[6V4]  682: ; BX=log.Spalte des Treffers
[ØØØ]  683:
[4B3]  684: TeilGetroffen   PROC
[6L3]  685:         mov     ax,[IPixZeile]
[4R1]  686:         sub     ax,16
[H35]  687:         shr     ax,1        ;al=Schusszeile
[GV1]  688:         mov     dx,ax
[VN2]  689:         shl     dx,1        ;*2
[434]  69Ø:         mov     bp,offset TeileTab
[PP1]  691:         add     bp,dx
[ZL1]  692:         shl     dx,1
[GP2]  693:         shl     dx,1        ;*8
[PP1]  694:         add     bp,dx
[UT4]  695:         ; maximal drei Zeilen beachten
[GN1]  696:         mov     cx,3
[VL1]  697:         cmp     ax,4
[FT1]  698:         jnc     wenØØ
[HA1]  699:         dec     cx
[ZM1]  7ØØ:         or      ax,ax
[BW1]  7Ø1:         jnz     wenØØ
[HA1]  7Ø2:         dec     cx
[C76]  7Ø3: wenØØ:  xor     al,al       ;al=Ø für 1.Teil
[286]  7Ø4: wenØ1:  push    cx          ;cx=Zeilenanzahl
[Y15]  7Ø5:         mov     ah,TSpalte ;ah=log.Spalte
[PR1]  7Ø6:         cmp     bl,ah
[LV4]  7Ø7:         ; Trefferspalte<TeilSpalte?
[1T4]  7Ø8:         jc      wenØ3       ;das Teil nicht
[1G1]  7Ø9:         add     ah,9
[HR1]  71Ø:         cmp     ah,bl
[9V4]  711:         ;Trefferspalte>TeilSpalte+9?
[1T4]  712:         jc      wenØ3       ;das Teil nicht
[315]  713:         pop     cx          ;Stack bereinigen
[VF5]  714:         jmp     trefferup   ;Teil gefunden
[YH6]  715: wenØ3:  add     bp,1Ø   ;2.Teil überprüfen
[NZ4]  716:         xor     al,1        ;al=1 für 2.Teil
[UR2]  717:         mov     ah,TSpalte
[PR1]  718:         cmp     bl,ah
[CM1]  719:         jc      wenØ5
[1G1]  72Ø:         add     ah,9
[HR1]  721:         cmp     ah,bl
[CM1]  722:         jc      wenØ5
[315]  723:         pop     cx          ;Stack bereinigen
[VF5]  724:         jmp     TrefferUP   ;Teil gefunden
[R36]  725: wenØ5:  sub     bp,3Ø       ;nächste Zeile
[CM1]  726:         xor     al,1
[5F5]  727:         ;al=Ø, wieder 1.Teil in der Zeile..
[VD1]  728:         pop     cx
[SW1]  729:         loop    wenØ1
[P55]  73Ø:         ; Return, wenn nichts gefunden.
[9Ø5]  731:         ; Dürfte aber eigentlich nicht
[F62]  732:         ; passieren !!
[3P0]  733:         ret
[393]  734: TeilGetroffen   ENDP
```

Listing 8. »Ggouts.inc« erledigt unter anderem die Sprite-Ausgabe und die Kollisionsabfrage.

```
[244]    1: ; Programm: ggmacros.inc
[456]    2: ; Funktion: MACRO-Vereinbarungen zu
[143]    3: ;                ggHaupt.asm
[913]    4: ; Sprache:  TASM 2.01
[VT3]    5: ; Autor:    Detlef Paulin
[7A4]    6: ; (c) 1992 Widuch GmbH & Co. KG
[ØØØ]    7:
[4L5]    8: ; Macro-Aufruf: PUSHEN RegLeiste,
[XW4]    9: ;                mit RegLeiste=<AX,...>
[DD4]   1Ø: ; Funktion    : push-Befehle
[V75]   11: ;               entsprechend RegLeiste
[ØØØ]   12:
[AØ4]   13: PUSHEN  MACRO   RegLeiste
[DA3]   14:         IRP     Reg,RegLeiste
[2S1]   15:         push    Reg
[HPØ]   16:         ENDM
[HPØ]   17:         ENDM
[ØØØ]   18:
[BF5]   19: ; Macro-Aufruf: POPEN RegLeiste,
[XW4]   2Ø: ;               mit RegLeiste=<AX,...>
[VA4]   21: ; Funktion    : pop-Befehle
[V75]   22: ;               entsprechend RegLeiste
[ØØØ]   23:
[HV3]   24: POPEN   MACRO   RegLeiste
[DA3]   25:         IRP     Reg,RegLeiste
[WN1]   26:         pop     Reg
[HPØ]   27:         ENDM
[HPØ]   28:         ENDM
[ØØØ]   29:
[WP4]   3Ø: ; Macro-Aufruf: WARTE laenge
[KP6]   31: ; Funktion    : softwaremäßige Warte-
[ØN5]   32: ;               schleife für laenge in ms
[DT6]   33: ; Referenz    : tinit (tcount muß gesetzt
[V54]   34: ;               sein) in ggenv.inc
[ØØØ]   35:
[A83]   36: WARTE   MACRO   laenge
[ZW3]   37:         LOCAL   wait0,wait1,wait2
[5D3]   38:         PUSHEN  <AX,BX,CX,DX>
[3H2]   39:         mov     cx,laenge
[VL1]   4Ø:         cmp     cx,Ø
[FP1]   41:         jz      wait2
[ØT1]   42:         xor     ah,ah
[YT4]   43: wait0:  mov     bx,ds:[tcount]
[RB4]   44: wait1:  inc     ds:[tcount+2]
[1G1]   45:         push    cx
[HL1]   46:         int     1ah
[VD1]   47:         pop     cx
[FA1]   48:         dec     bx
[2N1]   49:         or      bx,bx
[HW1]   5Ø:         jnz     wait1
[UW1]   51:         loop    wait0
[5D4]   52: wait2:  POPEN   <DX,CX,BX,AX>
[HPØ]   53:         ENDM
[ØØØ]   54:
[YH3]   55: ; Macro-Aufruf: INKEY
[N45]   56: ; Funktion    : Abfrage Tastatur
[3G5]   57: ; Ergebnis    : AX=Ø --> keine Eingabe
[ZT3]   58: ;               AH=Ø --> AL=ASCII
[455]   59: ;               AH=1 --> AL-Sondertaste
[443]   6Ø: ; Veränderte  : AX
[ØØØ]   61:
[SZ1]   62: INKEY   MACRO
[1H2]   63:         LOCAL   endinkey
[1V4]   64:         mov     ah,Øbh      ;Pufferstatus
[3K1]   65:         int     21h
[HU1]   66:         and     ax,255
[9S4]   67:         jz      endinkey    ;Puffer leer
```

```
[PU4]   68:         mov     ah,Ø8h      ;Zeichen holen
[3K1]   69:         int     21h
[HU1]   7Ø:         and     ax,255
[D55]   71:         jnz     endinkey    ;ASCII-Zeichen
[ØY4]   72:         mov     ah,Ø8h      ;Sonderzeichen
[3K1]   73:         int     21h
[UL1]   74:         mov     ah,1
[BH1]   75: endinkey:
[HPØ]   76:         ENDM
[ØØØ]   77:
[924]   78: ; Macro-Aufruf: PLAYKEY
[3G7]   79: ; Funktion    : Tastaturabfrage mit Löschen
[NN5]   8Ø: ;               des Tasturpuffers und
[SY4]   81: ;               Bearbeiten von F1 und F2
[5W5]   82: ; Ergebnis    : AX=SCAN-Code der Taste
[ØØØ]   83:
[392]   84: PLAYKEY MACRO
[T24]   85:         LOCAL   ik1,ik2,ik3,ik4,ik6
[MF1]   86:         push    ds
[5Z1]   87:         mov     ax,4ØH
[1V1]   88:         mov     ds,ax
[BZ1]   89:         mov     bx,8Øh
[SE2]   9Ø:         mov     cx,[bx]
[2N5]   91:         ;cx=Offset Tastaturpufferanfang
[6Ø2]   92:         mov     bx,1ah
[ME2]   93:         mov     ax,[bx]
[AW5]   94:         cmp     ax,[bx+2] ;Tasten gedrückt?
[NH2]   95:         jnz     ik1         ;ja
[ED1]   96:         pop     ds
[C45]   97:         xor     ax,ax       ;ax=Ø keine Taste
[UK1]   98:         jmp     ik3
[WT4]   99: ik1:    mov     WORD PTR [bx],cx
[384]   1ØØ:        mov     WORD PTR [bx+2],cx
[CV1]   1Ø1:        mov     bx,ax
[545]   1Ø2:        mov     ax,[bx]   ;ax=Tastencode
[ED1]   1Ø3:        pop     ds
[3Ø4]   1Ø4:        mov     [letzteTaste],ax
[7F3]   1Ø5:        xor     al,al       ;Ascii=Ø
[C35]   1Ø6:        xchg    al,ah       ;ax=Tastennummer
[PØ6]   1Ø7: ik3:   cmp     ax,STaste ;Sound an/aus?
[NH1]   1Ø8:        jne     ik4
[Y75]   1Ø9:        xor     WORD PTR [SoundSpiel],1
[9A5]   11Ø:        xor     ax,ax       ;Taste bearbeitet
[CU4]   111:        jmp     ik6
[BR5]   112: ik4:   cmp     ax,PanikTaste   ;Panik?
[XH1]   113:        jne     ik6
[NB3]   114:        call    PanikProgramm
[9A5]   115:        xor     ax,ax       ;Taste bearbeitet
[8S5]   116: ik6:   or      ax,ax       ;keine Taste ZF=1
[HPØ]   117:        ENDM
[ØØØ]   118:
[8K3]   119: ; Macro-Aufruf: WTAST
[837]   12Ø: ; Funktion    : wartet, bis Taste gedrückt
[7S6]   121: ; Referenz     : nutzt inkey --> AX enthält
[354]   122: ;                gedrückte Taste
[Z23]   123: ; Verändert    : AX
[ØØØ]   124:
[2Ø2]   125: WTAST  MACRO
[652]   126:        LOCAL   wtast1
[372]   127: wtast1: INKEY
[ZM1]   128:        or      ax,ax
[YX1]   129:        jz      wtast1
[HPØ]   13Ø:        ENDM
[ØØØ]   131:
[S57]   132: ; Macro-Aufruf: TEXT8x5 Pixelzeile,Spalte,
[M13]   133: ;               Text,Farbe
[CW6]   134: ; Funktion    : Textausgabe im Grafikmodus
[KJ6]   135: ; Eingang     : Text ist Textmarke oder
[Y24]   136: ;               BX zeigt auf Text
[SF6]   137: ; Referenz    : Zeichenausgabe des UPGRAF
[RB7]   138: ; Verändert   : BX=Zeiger Endekennzeichen+1
[ØØØ]   139:
[357]   14Ø: TEXT8x5 MACRO  pixzeile,spalte,text,farbe
[A33]   141:        mov     ah,pixzeile
[TG2]   142:        mov     al,spalte
[BØ2]   143: IFNB <text>
[2G3]   144:        mov     bx,offset text
[TU0]   145: ENDIF
[UA2]   146:        mov     dh,farbe
[C42]   147:        call    txt8x5
[HPØ]   148:        ENDM
[ØØØ]   149:
[5U3]   15Ø: txt8x5: mov    dl,[bx]
[DS1]   151:        cmp     dl,'$'
[2P2]   152:        jz      txt8x5Ende
[ZF1]   153:        push    bx
[8P3]   154:        mov     bx,offset zs8x5
[2R2]   155:        mov     cx,CHAR8x5
[132]   156:        call    upgraf
[TD1]   157:        pop     bx
[MB1]   158:        inc     bx
[252]   159:        jmp     txt8x5
[812]   16Ø: txt8x5Ende:
[MB1]   161:        inc     bx
[3P0]   162:        ret
[ØØØ]   163:
[F37]   164: ; Macro-Aufruf: BOX vonPixSpa,vonPixZei,
[AØ4]   165: ;                   AnzahlPixSpa,
[PV4]   166: ;                   AnzahlPixZei,Farbe
[WV5]   167: ; Funktion    : zeichnet ein Rechteck
[KM6]   168: ; Referenz    : Unterprogramme aus UPGRAF
[5E7]   169: ; Restriktion : Pixelbreite>4, Pixelhöhe>3,
[ØV4]   17Ø: ;               LPS,LPZ=Konstanten!
[ØØØ]   171:
[D26]   172: BOX    MACRO   VPS,VPZ,LPS,LPZ,FARBE
[ØB2]   173:        mov     al,FARBE
[D22]   174:        mov     ah,VPZ
[A32]   175:        mov     bx,VPS
[HV1]   176:        mov     dx,bx
[862]   177:        add     dx,LPS-1
[T23]   178:        mov     cx,HoriLine
[132]   179:        call    upgraf
[752]   18Ø:        add     ah,LPZ-1
[T23]   181:        mov     cx,HoriLine
[132]   182:        call    upgraf
[ØT1]   183:        mov     dh,ah
[391]   184:        dec     dh
[CT1]   185:        mov     dl,ah
[982]   186:        sub     dl,LPZ-2
[Z33]   187:        mov     cx,VertLine
[132]   188:        call    upgraf
[462]   189:        add     bx,LPS-1
[Z33]   19Ø:        mov     cx,VertLine
[132]   191:        call    upgraf
[HPØ]   192:        ENDM
[ØØØ]   193:
[H85]   194: ; Macro-Aufruf: COPY4 von,nach
[4T6]   195: ; Funktion    : kopiert die TeileParameter
[AL5]   196: ;               von [BP+von]->[BP+nach]
[KD5]   197: ;               und löscht das von-Teil
[K26]   198: ; Eingang     : BP zeigt auf Spritepara-
[YM5]   199: ;               meterbereich von TeileTab
[ØØØ]   2ØØ:
[XL3]   2Ø1: COPY4  MACRO   VON,NACH
[XP1]   2Ø2:        push    ax
[R85]   2Ø3:        mov     ax,ds:[bp+von]    ;TSpalte
[MN3]   2Ø4:        mov     ds:[bp+nach],ax
[5A5]   2Ø5:        mov     ax,ds:[bp+von+2]  ;TNummer
[3T3]   2Ø6:        mov     ds:[bp+nach+2],ax
[U45]   2Ø7:        mov     ax,ds:[bp+von+4]  ;TPhaseØ
[7T3]   2Ø8:        mov     ds:[bp+nach+4],ax
[4X4]   2Ø9:        mov     ax,ds:[bp+von+8]  ;TInfo
[FT3]   21Ø:        mov     ds:[bp+nach+8],ax
[GV1]   211:        xor     ax,ax
[PC5]   212:        mov     ds:[bp+von],ax    ;löschen
[RD1]   213:        pop     ax
[HPØ]   214:        ENDM
[ØØØ]   215:
```

Listing 9. »Ggmacro.inc« enthält allgemein verwendete Makrodefinitionen.

```
[ØØØ]  216:
[BV6]  217: ; Macro-Aufruf : RND Maximum   Maximum<256)
[U17]  218: ; Funktion     : AL bekommt pseudozufällig
[5G4]  219: ;                Wert aus Ø - Maximum
[DB6]  220: ; Hinweis      : nutzt Systemuhr und die
[SA5]  221: ;                RANDOM-Zelle des Spiels
[ØØØ]  222:
[AH3]  223: RND     MACRO   Maximum
[3G1]  224:         push    dx
[ZF1]  225:         push    bx
[MF1]  226:         push    ds
[634]  227:         inc     WORD PTR [random]
[5Z1]  228:         mov     ax,40h
[1V1]  229:         mov     ds,ax
[FØ2]  230:         mov     bx,6ch
[ME2]  231:         mov     ax,[bx]
[ED1]  232:         pop     ds
[F13]  233:         add     ax,[random]
[Ø33]  234:         add     ax,[punkte]
[7Ø3]  235:         add     ax,[energie]
[KW3]  236:         add     ax,[munivorrat]
[HF2]  237: IF Maximum GT 254
[PX1]  238:         mov     bx,255
[ZRØ]  239: ELSE
[H13]  240:         mov     bx,Maximum+1
[TUØ]  241: ENDIF
[R65]  242:         xor     dx,dx       ;DX:AX=Dividend
[Ø44]  243:         div     bx          ;Rest in DX
[YB5]  244:         mov     ax,dx       ;AX=Zufallswert
[TD1]  245:         pop     bx
[XD1]  246:         pop     dx
[HPØ]  247:         ENDM
```

Listing 9. »Ggmacro.inc« enthält allgemein verwendete Makrodefinitionen.

```
[T24]    1: ; Programm: ggjumps.inc
[3C7]    2: ; Funktion: Reaktionsroutinen für Abschuß
[XM6]    3: ;           eines Sprite, Zusammenstoß mit
[HD6]    4: ;           einem Sprite und Bewegung eines
[M42]    5: ;           Sprite
[913]    6: ; Sprache:  TASM 2.Ø1
[VT3]    7: ; Autor:    Detlef Paulin
[7A4]    8: ; (c) 1992 Widuch GmbH & Co. KG
[P3Ø]    9: ;
[ØØØ]   10:
[PN6]   11: ; Verschiedene Explosionsarten bei einem
[RB6]   12: ; Treffer und eventueller Vernichtung
[756]   13: ; BX=Zeiger auf InfoLeiste zum Sprite
[ØØØ]   14:
[7Ø3]   15: TeilAngeschossen:
[PT3]   16:         mov     bx,offset implod
[KZ3]   17:         mov     si,offset klexpls
[TH3]   18:         mov     dx,TeileFormat
[W63]   19:         mov     ax,TPixZeile
[UR2]   20:         mov     ah,TSpalte
[AN1]   21:         sub     ah,11
[1L1]   22:         shr     ah,1
[7S1]   23:         xchg    al,ah
[NN1]   24:         mov     cx,8
[VW1]   25:         jmp     expl1
[ØV2]   26: Teilexplodiert:
[BF2]   27:         mov     ax,TExpl
[ZM1]   28:         or      ax,ax
[BK4]   29:         jz      teilexplodiertgross
[DA1]   30:         dec     ax
[FG4]   31:         jz      teilexplodiertklein
[DA1]   32:         dec     ax
[1F3]   33:         jz      teilimplodiert
[DA1]   34:         dec     ax
[RX4]   35:         jz      teilexplodiertgewaltig
[DA1]   36:         dec     ax
[TT3]   37:         jz      tbombeexplodiert
[B13]   38: tbombeexplodiert:
[PT3]   39:         mov     bx,offset implod
[KZ3]   40:         mov     si,offset bexpls
[ØH2]   41:         jmp     teilexpl
[7U2]   42: teilimplodiert:
[PT3]   43:         mov     bx,offset implod
[HY3]   44:         mov     si,offset implods
[ØH2]   45:         jmp     teilexpl
[X84]   46: teilexplodiertgewaltig:
[HY3]   47:         mov     bx,offset grexplb
[KU3]   48:         mov     si,offset mexpls
[ØH2]   49:         jmp     teilexpl
[7P3]   50: teilexplodiertklein:
[PT3]   51:         mov     bx,offset implod
[KZ3]   52:         mov     si,offset klexpls
[ØH2]   53:         jmp     teilexpl
[3S3]   54: teilexplodiertgross:
[HY3]   55:         mov     bx,offset grexplb
[8Z3]   56:         mov     si,offset grexpls
[TL1]   57: teilexpl:
[TN4]   58:         mov     bp,[ParameterZeiger]
[W63]   59:         mov     ax,TPixZeile
[UR2]   60:         mov     ah,TSpalte
[AN1]   61:         sub     ah,11
[1L1]   62:         shr     ah,1
[7S1]   63:         xchg    al,ah
[TH3]   64:         mov     dx,TeileFormat
[LE3]   65:         PUSHEN  <AX,BX,DX,SI>
[K53]   66:         call    teilausgeben
[G74]   67:         mov     BYTE PTR TSpalte,Ø
[DF4]   68:         mov     ax,[ErstesZweites]
[3L1]   69:         or      al,al
[N42]   70:         jnz     texplØ
[6K1]   71:         COPY4   1Ø,Ø
[MK4]   72: texplØ: POPEN   <SI,DX,BX,AX>
[FØ3]   73: texpl1: mov     cx,8
[6V3]   74: expl1:  PUSHEN  <CX,BX>
[9D3]   75:         mov     cx,XORmitWORD
[XL4]   76:         mov     word ptr [nr_vga],73
[132]   77:         call    upgraf
[ØV1]   78:         mov     bx,si
[2Y1]   79:         call    sound
[L15]   8Ø:         add     si,2   ;SI=nächster Sound
[TD1]   81:         pop     bx
[ZF1]   82:         push    bx
[9D3]   83:         mov     cx,XORmitWORD
[XL4]   84:         mov     word ptr [nr_vga],73
[132]   85:         call    upgraf
[YF2]   86:         POPEN   <BX,CX>
[ZH5]   87:         add     bx,72 ;BX=nächste ExplPhase
[HY1]   88:         loop    expl1
[3PØ]   89:         ret
[ØØØ]   9Ø:
[E56]   91: ; verschiedene Routinen bei Treffer,
[SP7]   92: ; Zusammenstoß und für Bewegung der Sprite
[756]   93: ; BX=Zeiger auf InfoLeiste zum Sprite
[ØØØ]   94:
[LE7]   95: ; Punkteaktualisierung bei Zusammenstoß
[ØØØ]   96:
[N62]   97: CrashPunkte:
[S43]   98:         mov     ax,CRPunkte
[612]   99:         jmp     TPKTØ1
[ØØØ]  1ØØ:
[XS5]  1Ø1: ; Punkteaktualisierung bei Treffer
[ØØØ]  1Ø2:
[RH2]  1Ø3: TrefferPunkte:
[6V2]  1Ø4:         mov     ax,TPunkte
[886]  1Ø5: TPKTØ1: or      ax,ax     ;Punkte beachten?
[UE3]  1Ø6:         jz      TPKTEnde  ;nein
[W84]  1Ø7:         test    ax,Øff ØØh ;-/+Punkte?
[NØ4]  1Ø8:         jz      TPKTPlus  ;+Punkte
[XV4]  1Ø9:         xor     ah,ah     ;Vorzeichen zu +
[EV1]  11Ø: TPKTMinus:
[L63]  111:         sub     [punkte],ax
[BE2]  112:         jnc     TPKTEnde
[EH4]  113:         mov     WORD PTR [punkte],Ø
[WF2]  114:         jmp     TPKTEnde
[RN1]  115: TPKTPlus:
[443]  116:         add     [punkte],ax
[NK1]  117: TPKTEnde:
[3PØ]  118:         ret
[ØØØ]  119:
[Ø87]  12Ø: ; Energieaktualisierung bei Zusammenstoß
[ØØØ]  121:
[5Ø2]  122: CrashEnergie:
[E43]  123:         mov     ax,CREnergie
[X15]  124:         or      ax,ax     ;Energie beachten?
[B93]  125:         jz      CREEnde   ;nein
[H74]  126:         test    ax,Øff ØØh  ;+/-Energie
[5A2]  127:         jz      CREPlus
[LV4]  128:         xor     ah,ah     ;Vorzeichen zu +
[ML1]  129: CREMinus:
[B43]  13Ø:         sub     [energie],ax
[P35]  131:         jnc     CREEnde   ;kein Spielende
[6H3]  132:         xor     ax,ax         ;ax=Ø
[X53]  133:         mov     [energie],ax
[395]  134:         mov     [gameover],ax   ;Game over
[CB2]  135:         jmp     CREEnde
[XF1]  136: CREPlus:
[7Ø3]  137:         add     ax,[energie]
[4B5]  138:         cmp     ax,MaxEnergie+1 ;>Maximum?
[S93]  139:         jc      CREØ1           ;nein
[193]  14Ø:         mov     ax,MAxEnergie
[D94]  141: CREØ1:  mov     [energie],ax
[BC1]  142: CREEnde:
[3PØ]  143:         ret
[ØØØ]  144:
[YZ7]  145: ; Munitionsaktualisierung bei Zusammenstoß
[ØØØ]  146:
[T62]  147: CrashSchuss:
[W43]  148:         mov     ax,CRSchuss
[GZ4]  149:         or      ax,ax         ;Munizuschuß?
[NK3]  15Ø:         jz      CRSEnde       ;nein
[NH1]  151: CRSPlus:
[KW3]  152:         add     ax,[munivorrat]
[RY4]  153:         cmp     ax,MaxMuni+1  ;>Maximum?
[EA3]  154:         jc      CRSØ1         ;nein
[GU2]  155:         mov     ax,MaxMuni
[R65]  156: CRSØ1:  mov     [munivorrat],ax
[3E1]  157: CRSEnde:
[3PØ]  158:         ret
[ØØØ]  159:
[994]  16Ø: ; Standardtrefferroutine
[ØØØ]  161:
[8Ø3]  162: TrefferAllgemein:
[SN2]  163:         mov     bx,TPhaseØ
[273]  164:         sub     bx,Infolaenge
[595]  165:         cmp     BYTE PTR TInfo,Ø  ;Action?
[EM3]  166:         jz      TRAAction         ;ja
[YD5]  167:         dec     BYTE PTR TInfo    ;Zähler-1
[SW3]  168:         call    TeilAngeschossen
[3PØ]  169:         ret
[2T1]  17Ø: TRAAction:
[EC3]  171:         call    TrefferPunkte
[XL3]  172:         call    Teilexplodiert
[3PØ]  173:         ret
[ØØØ]  174:
[K17]  175: ; Trefferroutine Sprite 'Energiefresser'
[ØØØ]  176:
[VU2]  177: TrefferFresser:
[SN2]  178:         mov     bx,TPhaseØ
[273]  179:         sub     bx,InfoLaenge
[AS3]  18Ø:         cmp     BYTE PTR TInfo,Ø
[GA2]  181:         jz      TFAction
[ØF3]  182:         dec     BYTE PTR TInfo
[3PØ]  183:         ret
[2K1]  184: TFAction:
[K53]  185:         call    TeilAusgeben
[HW3]  186:         inc     WORD PTR TNummer
[F94]  187:         add     WORD PTR TPhaseØ,156
[4U3]  188:         mov     WORD PTR TInfo,5
[K53]  189:         call    TeilAusgeben
[3PØ]  19Ø:         ret
[ØØØ]  191:
[H36]  192: ; Trefferroutine Sprite 'Atombombe'
[ØØØ]  193:
[D13]  194: TrefferAtombombe:
[SN2]  195:         mov     bx,TPhaseØ
[273]  196:         sub     bx,Infolaenge
[6X1]  197:         mov     ax,16Ø
[S84]  198:         sub     ax,[TrefferSpalte]
[5G2]  199:         call    CREMinus
[XL3]  2ØØ:         call    Teilexplodiert
[3PØ]  2Ø1:         ret
[ØØØ]  2Ø2:
[Z16]  2Ø3: ; Trefferroutine Sprite 'Wandler' (?)
[ØØØ]  2Ø4:
[1H3]  2Ø5: TrefferTeiltausch:
[SW3]  2Ø6:         call    TeilAngeschossen
[K53]  2Ø7:         call    TeilAusgeben
[9F3]  2Ø8:         RND     15              ;1 aus 16
[8G4]  2Ø9:         mov     bx,offset RNDwandler
[L75]  21Ø:         xlat                ;ax=neue Teilnummer
[TY2]  211:         mov     TNummer,ax
[KS3]  212:         mov     bx,offset pictab
[TL1]  213:         shl     ax,1
[TL1]  214:         shl     ax,1
[XD5]  215:         add     bx,ax ;bx=Zeiger auf PhaseØ
[ME2]  216:         mov     ax,[bx]
[3T2]  217:         mov     TPhaseØ,ax
[BN1]  218:         mov     ax,2
[YD2]  219:         mov     TInfo,ax
[SF2]  22Ø:         mov     ax,[bx+2]
[DM1]  221:         mov     cl,4
[MA5]  222:         shl     ax,cl   ;ax=16*Phasenanzahl
[ET2]  223:         mov     cx,TSpalte
[C14]  224:         and     ch,1            ;Phase=Ø/1
[S35]  225:         add     ch,al   ;ch=Phasenanzahl*16
[H12]  226:                 ;               +akt.Phase
[3Y2]  227:         mov     TSpalte,cx
[K53]  228:         call    TeilAusgeben
[3PØ]  229:         ret
[ØØØ]  23Ø:
[RB6]  231: ; Standardroutine bei Zusammenstoß
[ØØØ]  232:
[WP2]  233: CrashAllgemein:
[SN2]  234:         mov     bx,TPhaseØ
[273]  235:         sub     bx,InfoLaenge
[Ø53]  236:         call    CrashPunkte
[P43]  237:         call    CrashEnergie
[453]  238:         call    CrashSchuss
[XL3]  239:         call    Teilexplodiert
[3PØ]  24Ø:         ret
[ØØØ]  241:
[546]  242: ; Crashroutine für Sprite 'Atombombe'
[ØØØ]  243:
[9R2]  244: CrashAtombombe:
[SN2]  245:         mov     bx,TPhaseØ
[273]  246:         sub     bx,InfoLaenge
[2X1]  247:         mov     ax,52Ø
[5G2]  248:         call    CREMinus
[XL3]  249:         call    Teilexplodiert
[3PØ]  25Ø:         ret
[817]  251: ; Crashroutine für Sprite 'Schwarzes Loch'
[ØØØ]  252:
[4S1]  253: CrashLoch:
[SN2]  254:         mov     bx,TPhaseØ
[273]  255:         sub     bx,InfoLaenge
[2X1]  256:         mov     ax,52Ø
[5G2]  257:         call    CREMinus
[XL3]  258:         call    Teilexplodiert
[3PØ]  259:         ret
[ØØØ]  26Ø:
[GL6]  261: ; Crashrourine für Sprite 'Sprungfeder'
[ØØØ]  262:
[NE1]  263: CrashFeder:
[6L3]  264:         mov     ax,[IPixZeile]
[GH1]  265:         add     ax,8
[A53]  266:         cmp     ax,TPixZeile
[CD2]  267:         jne     CFNormal
[3P4]  268:         mov     ax,TSpalte      ;al=Spalte
[XH3]  269:         mov     ah,ISpalteBis-4
[915]  27Ø:         cmp     ah,al   ;Teil noch vor ICH?
[FE3]  271:         jc      CFNormal        ;ja
[BL3]  272:         cmp     al,ISpalteVon-6
[A43]  273:                         ;Teil schon durch?
[FE3]  274:         jc      CFNormal        ;ja
[3P4]  275:         mov     WORD PTR [springt],1
[Y74]  276:         mov     WORD PTR [flug],1Ø
[T74]  277:         mov     WORD PTR [hoch],25
[K53]  278:         call    TeilAusgeben
[F94]  279:         add     WORD PTR TPhaseØ,156
[K53]  28Ø:         call    TeilAusgeben
[VL5]  281:         call    AUTØØ ;noch normal ausgeben
[3PØ]  282:         ret
[EH1]  283: CFNormal:
[TF3]  284:         call    CrashAllgemein
[3PØ]  285:         ret
[ØØØ]  286:
[VW5]  287: ; Crashroutine für Sprite 'Bonus 1ØØ'
[ØØØ]  288:
[Y12]  289: CrashBonus:
[SN2]  29Ø:         mov     bx,TPhaseØ
[273]  291:         sub     bx,InfoLaenge
[Ø53]  292:         call    CrashPunkte
[P43]  293:         call    CrashEnergie
[453]  294:         call    CrashSchuss
[TN4]  295:         mov     bp,[ParameterZeiger]
[K53]  296:         call    teilausgeben
[3PØ]  297:         ret
[ZW6]  298: ; Crashroutine für Sprite 'Extraraumtor'
[ØØØ]  299:
[1L1]  3ØØ: CrashTor:
[L95]  3Ø1:         cmp     WORD PTR [ImExtraBild],Ø
[G75]  3Ø2:         jz      CRTorØ  ;nur aus Grundbild
[FG3]  3Ø3:         jmp     CrashAllgemein
[676]  3Ø4:         ; eventuelle Aufstiegsphase beenden
[DD5]  3Ø5: CRTorØ: mov     WORD PTR [hoch],Ø
[GN3]  3Ø6:         ; alle Sprites löschen
[434]  3Ø7:         mov     bp,offset TeileTab
[VS1]  3Ø8:         mov     cx,64
[VU2]  3Ø9: CRTor1: push    cx
[7F1]  31Ø:         push    bp
[6S2]  311:         mov     al,TSpalte
[LK1]  312:         cmp     al,Ø
[RY1]  313:         jz      CRTor2
[RN3]  314:         cmp     al,StartSpalte+1
[SW1]  315:         je      CRTor2
[MB4]  316:         cmp     WORD PTR TNummer,61
[M3P]  317:         jne     CRTor4
[8G4]  318:         ;Sprite 'Tarnkappenbomber'
[F93]  319:         ;besonders beachten
[DU1]  32Ø:         cmp     al,131
[XU4]  321:         jnc     CRTorA      ;noch sichtbar
[7P1]  322:         cmp     al,86
[ET5]  323:         jnc     CRTorB ;unsichtbar,löschen!
[WH4]  324: CRTorA: call    TeilAusgeben
[XL5]  325: CRTorB: mov     BYTE PTR TSpalte,Ø
[YR2]  326: CRTor2: pop     bp
[LL1]  327:         add     bp,1Ø
[VD1]  328:         pop     cx
[352]  329:         loop    CRTor1
[875]  33Ø:         mov     WORD PTR [FolgenTakt],4Ø
[6B5]  331:         mov     WORD PTR [ImExtraBild],1
[X63]  332:         mov     ax,[BildPos]
[US3]  333:         mov     [PosMerker],ax
[UX3]  334:         mov     ax,[ExtraLevel]
[TW2]  335:         ; Extraraum holen
[SF4]  336:         mov     bx,offset ExtraFolge
[CRØ]  337:         xlat
[1V3]  338:         mov     bx,offset ExtraØ1
[ZM1]  339:         or      ax,ax
[XY1]  34Ø:         jz      CRTor4
[EV1]  341:         mov     cx,ax
[ZS1]  342:         mov     dx,46
[E23]  343: CRTor3: add     bx,dx
[952]  344:         loop    CRTor3
[6S4]  345:         ; Bildunterschrift ausgeben
[343]  346: CRTor4: mov     cx,38
[GH4]  347:         mov     si,offset ExtraSound
[7V2]  348: CRTor5: push    cx
[ZF1]  349:         push    bx
[DR1]  35Ø:         add     bx,cx
[FA1]  351:         dec     bx
[RC5]  352:         TEXT8x5 TextZei,TextSp,,ColorE
[ØV1]  353:         mov     bx,si
[2Y1]  354:         call    sound
[1H1]  355:         add     si,2
[TD1]  356:         pop     bx
[VD1]  357:         pop     cx
[F52]  358:         loop    CRTor5
[RG5]  359:         mov     ax,[bx+4Ø]   ;ax=Raumzeiger
[C63]  36Ø:         mov     [BildPos],ax
[U85]  361:         mov     ax,[bx+42]   ;zeitweises
[PD5]  362:         mov     [TMax],ax    ;Maximaltempo
[YØ5]  363:         mov     ax,[bx+44]   ;zeitweises
[YC5]  364:         mov     [Tmin],ax    ;Minimaltempo
[YØ5]  365:         mov     ax,[tempo]   ;aktuelles
[WB5]  366:         mov     [TAkt],ax    ;Tempo merken
[UM2]  367:         mov     cx,[TMin]
[UT1]  368:         cmp     ax,cx
[1X1]  369:         jc      CRTor7
[242]  37Ø:         jmp     CRTor8
[9Z3]  371: CRTor7: mov     cx,[TMax]
[UT1]  372:         cmp     ax,cx
[4X1]  373:         jc      CRTor8
[542]  374:         jmp     CRTor9
[WB4]  375: CRTor8: mov     [tempo],cx
[N12]  376: CRTor9: ret
[ØØØ]  377:
[X85]  378: ; Crashroutine für Sprite 'Mine'
[ØØØ]  379:
[MR1]  38Ø: CrashMine:
[Y63]  381:         mov     bx,TPixZeile
[6L3]  382:         mov     ax,[IPixZeile]
[TT1]  383:         cmp     ax,bx
[LP2]  384:         jz      MineKracht
[EH1]  385:         add     bx,4
[TT1]  386:         cmp     ax,bx
[LP2]  387:         jz      MineKracht
[EH1]  388:         add     bx,4
[TT1]  389:         cmp     ax,bx
[LP2]  39Ø:         jz      MineKracht
[WV1]  391:         jmp     AUTØØ
[T12]  392: MineKracht:
[FG3]  393:         jmp     CrashAllgemein
[ØØØ]  394:
[LT5]  395: ; Crashroutine für ersten 4 Sprite
[ØØØ]  396:
[A@1]  397: CRTØØ:
[W63]  398:         mov     ax,TPixZeile
[GH1]  399:         add     ax,8
[MH3]  4ØØ:         cmp     ax,[IPixZeile]
[HE2]  4Ø1:         jnz     CRTØØEX
[6V1]  4Ø2:         call    AUTØØ
[3PØ]  4Ø3:         ret
[5E1]  4Ø4: CRTØØEX:
[EP4]  4Ø5:         call    teilexplodiertgross
[ZP1]  4Ø6: CRTØØEnde:
[3PØ]  4Ø7:         ret
[ØØØ]  4Ø8:
[ØØØ]  4Ø9:
[TY3]  41Ø: ; Standardroutine für Sprite-Weiterbewegung
[ØØØ]  411:
[TY3]  412: AUTØØ:  mov     ax,TSpalte
[RN3]  413:         cmp     al,StartSpalte+1
[TY3]  414:         jz      AUTØØa      ;Teil neu
[DL5]  415:         call    teilausgeben ;Teil löschen
[AT2]  416:         mov     ax,TSpalte
[YS1]  417:         mov     ch,ah
```

```
[9A5]  418:          and     ch,ØFØH ;ch=Phasenanzahl*16
[AT1]  419:          mov     cl,ah
[HL1]  420:          shr     cl,1
[HL1]  421:          shr     cl,1
[HL1]  422:          shr     cl,1
[ØM4]  423:          shr     cl,1            ;cl=AND-Maske
[BG3]  424:          inc     ah              ;Phase+1
[1P1]  425:          and     ah,cl
[VM1]  426:          add     ah,ch
[GØ5]  427: AUTØØa: dec     al              ;Spalte-1
[ZX2]  428:          mov     TSpalte,ax
[3L1]  429:          or      al,al
[1Y4]  430:          jz      AUTØØb      ;Teil ist durch
[7W4]  431:          call    teilausgeben    ;Teil neu
[3PØ]  432:          ret
[T55]  433:          ; Teil hat Bildschirm verlassen
[6Z4]  434:          ; -> aus Spriteliste löschen
[Ø37]  435: AUTØØb: test    WORD PTR [ErstesZweites],1
[WL2]  436:          jnz     AUTØØEnde
[6K1]  437:          COPY4   1Ø,Ø
[ZE6]  438:        add WORD PTR [ParameterZeiger],Øfff6h
[TS5]  439:          mov     WORD PTR [ErstesZweites],1
[4R1]  440: AUTØØEnde:
[3PØ]  441:          ret
[ØØØ]  442:
[LG6]  443: ; Bewegung Sprite 'schnelles Raumschiff'
[8W4]  444: ; führt zwei Bewegungen aus
[M73]  445: AUTRak: call    AUTØØ
[Y54]  446:          cmp     BYTE PTR TSpalte,Ø
[2L4]  447:          je      AUTRak1         ;gelöscht!
[6V1]  448:          call    AUTØØ
[4C1]  449: AUTRak1:
[3PØ]  450:          ret
[ØØØ]  451:
[SF5]  452: ; Bewegung Sprite 'schwarzes Loch'
[YK6]  453: ; Das 'schwarze Loch' enzieht Goldgräber
[PF6]  454: ; bei Annäherung Energie, ist Energie Ø,
[GM6]  455: ; so fällt Goldgräber ins 'Loch' -> Ende
[ØØØ]  456:
[9K1]  457: AUTLoch1:
[A43]  458:          mov     bx,[Energie]
[US3]  459:          or      bx,bx     ;Energie=Ø?
[EC4]  460:          jz      AUTL1Ø    ;ja->Zieht an
[BX1]  461:          mov     ax,171
[MP2]  462:          sub     al,TSpalte
[EM1]  463:          mov     cl,5
[K25]  464:          shr     ax,cl     ;ax=Minus-Energie
[195]  465:          sub     bx,ax     ;genug Energie da?
[MR2]  466:          jnc     AUTLØ1    ;ja
[LV1]  467:          xor     bx,bx
[TE4]  468: AUTLØ1: mov     [Energie],bx
[S36]  469: AUTLØ2: jmp     AUTØØ     ;normale Ausgabe
[AV5]  470:          ;Goldgräber bewegt sich zum 'Loch'
[XV4]  471: AUTL1Ø: mov     ax,[IPixZeile]
[ZL1]  472:          sub     ax,8
[A53]  473:          cmp     ax,TPixZeile
[GU1]  474:          jc      AUTLØ2
[S24]  475:          mov     WORD PTR [hoch],1
[C34]  476:          mov     WORD PTR [flug],Ø
[UY4]  477:          jmp     AUTØØ     ;normale Ausgabe
[ØØØ]  478:
[3H6]  479: ; Bewegung des Sprite 'Tarnkappenbomber'
[NN5]  480: ; Sprite ist zeitweise unsichtbar
[ØØØ]  481:
[BC2]  482: AUTTarnkappe:
[AT2]  483:          mov     ax,TSpalte
[8S4]  484:          cmp     al,131    ;bis 135 zu sehen
[W85]  485:          je      AUTTK1    ;unsichtbar mache
       n
[A53]  486:          jnc     AUTTKNormal
[525]  487:          cmp     al,86     ;sichtbar machen?
[CM2]  488:          je      AUTTK2    ;ja
[BW4]  489:          jc      AUTTKNormal  ;Spalte< 66
[8S2]  490Ø:                           ;normale Ausgabe
[FT4]  491:          dec     al        ;Teil unsichtbar
[ZX2]  492:          mov     TSpalte,ax
[3PØ]  493:          ret
[LU2]  494: AUTTK1: push    ax
[K53]  495:          call    TeilAusgeben
[RD1]  496:          pop     ax
[991]  497:          dec     al
[ZX2]  498:          mov     TSpalte,ax
[3PØ]  499:          ret
[2M2]  500: AUTTK2: dec     al
[ZX2]  501:          mov     TSpalte,ax
[K53]  502:          call    TeilAusgeben
[3PØ]  503:          ret
[U72]  504: AUTTKNormal:
[6V1]  505:          call    AUTØØ
[3PØ]  506:          ret
[ØØØ]  507:
[XK4]  508: ; Bewegung des Sprite 'Bomber'
[ZØ7]  509: ; klinkt Sprite 'Bombe' aus, wenn darunter
[353]  510: ; 3 Zeilen frei sind
[ØØØ]  511:
[NS2]  512: KlinktBombeAus:
[2T4]  513:          cmp     WORD PTR TPixZeile,12Ø
[PB5]  514:          jnc     KlinktNichtAus      ;zu tief
[AT3]  515:          mov     ax,TSpalte
[G55]  516:          cmp     al,StartSpalte-5Ø  ;jetzt?
[VE4]  517:          jc      KlinktVielleichtAus
[TV2]  518: KlinktNichtAus:
[975]  519:          call    AUTØØ      ;normale Ausgabe
[3PØ]  520:          ret
[SP3]  521: KlinktVielleichtAus:
[RR4]  522:          RND     BombenWKT  ;'würfeln'
[ZM1]  523:          or      ax,ax
[RP3]  524:          jnz     KlinktNichtAus
[CR5]  525:          ; Prüfen, ob 3 Zeilen darunter frei
[GT2]  526:          mov     dx,TSpalte
[EG1]  527:          add     dl,4
[NL1]  528:          add     bp,3Ø
[4R5]  529:          cmp     WORD PTR [ErstesZweites],Ø
[MV1]  530:          jnz     KVAØ1
[LL1]  531:          add     bp,1Ø
[1A6]  532: KVAØ1:  mov     cx,3     ;3 müssen frei sein
[8X3]  533: KVAØ2:  mov     ax,TSpalte
[YM4]  534:          or      al,al    ;1.Teil besetzt?
[3Ø5]  535:          jz      KVAØ3    ;nein->Zeile frei
[TN4]  536:          mov     bp,[ParameterZeiger]
[CN3]  537:          jmp     KlinktNichtAus
[HK5]  538: KVAØ3:  add     bp,2Ø    ;nächste Zeile
[1W1]  539:          loop    KVAØ2
[YR1]  540: KlinkeAus:
[W65]  541:          ; 'Bombe' in Spritetab. eintragen
[7P1]  542:          sub     bp,6Ø
[KS3]  543:          mov     bx,offset pictab
[T35]  544:          add     bx,4*23   ;bx=Zeiger PhaseØ
[FB4]  545:          mov     ax,[bx]   ;ax=PhaseØ
[3T2]  546:          mov     TPhaseØ,ax
[SF2]  547:          mov     ax,[bx+2]
[DM1]  548:          mov     cl,4
[F35]  549:          shl     ax,cl     ;al=16*Phasenzahl
[MG3]  550:          and     dh,1      ;Phase=Ø/1
[7N1]  551:          add     dh,al
[5Y2]  552:          mov     TSpalte,dx
[RB4]  553:          mov     ax,23     ;Bombe=Teil 23
[TY2]  554:          mov     TNummer,ax
[D45]  555:          mov     ax,2Ø1h   ;alle 2* ausgeben
[YD2]  556:          mov     TInfo,ax
[K53]  557:          call    TeilAusgeben
[TN4]  558:          mov     bp,[ParameterZeiger]
[8X4]  559:          call    AUTØØ     ;Bomber ausgeben
[3PØ]  560:          ret
[ØØØ]  561:
[KC4]  562: ; Bewegung des Sprite 'Bombe'
[395]  563: ; Sie fällt nach unten (logisch)
[ØØØ]  564:
[K42]  565: BombeFaellt:
[GT2]  566:          mov     dx,TSpalte
[XN3]  567:          cmp     dl,StartSpalte+1
[CM1]  568:          jnz     BFØØ
[CD5]  569:          call    AUTØØ ;Bombe kommt ins Bild
```

```
[3PØ]  57Ø:          ret
[SL2]  571: BFØØ:    cmp     dl,12
[HH1]  572:          jnc     BFØ1
[6W5]  573:          call    AUTØØ ;Bombe verläßt Bild
[3PØ]  574:          ret
[1R5]  575: BFØ1:    cmp     WORD PTR TPixZeile,14Ø
[G74]  576:          jnz     BFØ2  ;nicht am Boden
[SY2]  577: BombeExplodiert:
[TN4]  578:          mov     bp,[ParameterZeiger]
[G85]  579:          mov     bx,TPhaseØ  ;expl. am Boden
[273]  58Ø:          sub     bx,InfoLaenge
[XL3]  581:          call    teilexplodiert
[3PØ]  582:          ret
[MS5]  583:          ; wenn Bombe anderes Sprite trifft
[UM3]  584:          ; explodiert die Bombe
[XN2]  585: BFØ2:    mov     si,1Ø
[7R5]  586:          cmp     WORD PTR [ErstesZweites],1
[FM1]  587:          jnz     BFØ3
[2P1]  588:          sub     bp,1Ø
[2S1]  589:          mov     si,2Ø
[T23]  59Ø:          call    BombenCrash
[4Z3]  591:          jnz     BombeExplodiert
[Ø53]  592:          jmp     ZeileTiefer
[GH2]  593: BFØ3:    add     bp,1Ø
[T23]  594:          call    BombenCrash
[4Z3]  595:          jnz     BombeExplodiert
[Z52]  596: ZeileTiefer:
[7P1]  597:          add     bp,si
[RT1]  598:          mov     di,si
[3S1]  599:          mov     si,3Ø
[MS1]  6ØØ:          sub     si,di
[Y54]  601:          cmp     BYTE PTR TSpalte,Ø
[GM1]  602:          jnz     BFØ4
[6L1]  603:          jmp     BFØ5
[YH2]  604: BFØ4:    add     si,1Ø
[LL1]  605:          add     bp,1Ø
[Y54]  606:          cmp     BYTE PTR TSpalte,Ø
[4Z3]  607:          jnz     BombeExplodiert
[2P1]  608:          sub     bp,1Ø
[GL2]  609: BFØ5:    mov     cx,4
[Z64]  61Ø: BFZT:    call    BombenCrash
[4Z3]  611:          jnz     BombeExplodiert
[LL1]  612:          add     bp,1Ø
[3V1]  613:          loop    BFZT
[Z95]  614:          ; Bombe fällt eine Zeile tiefer
[WS1]  615: ZeileFrei:
[MF1]  616:          push    si
[TN4]  617:          mov     bp,[ParameterZeiger]
[RØ6]  618:          call    TeilAusgeben ;Bombe löschen
[ED1]  619:          pop     si
[TN4]  62Ø:          mov     bp,[ParameterZeiger]
[1A5]  621:          COPY4   Ø,si    ;Bombe Zeile tiefer
[7P1]  622:          add     bp,si
[AT2]  623:          mov     ax,TSpalte
[YS1]  624:          mov     ch,ah
[P22]  625:          and     ch,ØFØH
[AT1]  626:          mov     cl,ah
[HL1]  627:          shr     cl,1
[HL1]  628:          shr     cl,1
[HL1]  629:          shr     cl,1
[HL1]  63Ø:          shr     cl,1
[2A1]  631:          inc     ah
[1P1]  632:          and     ah,cl
[VM1]  633:          add     ah,ch
[991]  634:          dec     al
[ZX2]  635:          mov     TSpalte,ax
[3L1]  636:          or      al,al
[HW1]  637:          jz      BFEnde
[K53]  638:          call    TeilAusgeben
[GD2]  639:          mov     ax,TInfo
[U12]  64Ø:          add     ax,2ØØh
[YD2]  641:          mov     TInfo,ax
[UY1]  642: BFEnde: ret
[ØØØ]  643:
[ER7]  644: ; Routine überprüft, ob Bombe ein gegebenes
[1S2]  645: ; Sprite trifft
[UM6]  646: ; BP=Zeiger auf zu untersuchendes Teil,
[AU2]  647: ; dl=Bombenspalte
[N86]  648: ; wenn kein Treffer --> Z-Flag gesetzt!
[ØØØ]  649:
[E42]  65Ø: BombenCrash:
[JG1]  651:          push    dx
[AT2]  652:          mov     ax,TSpalte
[AG1]  653:          add     al,6
[VL4]  654:          cmp     al,dl   ;Teil vor Bombe?
[Ø74]  655:          jc      KeinBombenCrash ;ja
[DL1]  656:          add     dl,11
[E85]  657:          cmp     dl,al   ;Teil hinter Bombe?
[8L3]  658:          jc      KeinBombenCrash
[AN1]  659:          mov     ax,1
[PL3]  66Ø:          jmp     IstBombenCrash
[5V2]  661: KeinBombenCrash:
[GV1]  662:          xor     ax,ax
[3U2]  663: IstBombenCrash:
[XD1]  664:          pop     dx
[ZM1]  665:          or      ax,ax
[3PØ]  666:          ret
[ØØØ]  667:
[B26]  668: ; Bewegung des Sprite 'Missileträger'
[ØM5]  669: ; klinkt Missile ausklinken, wenn
[7Ø4]  67Ø: ; 3 Zeilen darunter frei!
[ØØØ]  671:
[K33]  672: KlinktMissileAus:
[2T3]  673:          cmp     WORD PTR TPixZeile,12Ø
[Ø25]  674:          jnc     MissileNicht      ;zu tief
[AT2]  675:          mov     ax,TSpalte
[E55]  676:          cmp     al,StartSpalte-3Ø  ;jetzt?
[FV3]  677:          jc      MissileVielleicht
[F92]  678: MissileNicht:
[TN4]  679:          mov     bp,[ParameterZeiger]
[975]  68Ø:          call    AUTØØ      ;normale Ausgabe
[3PØ]  681:          ret
[Ø73]  682: MissileVielleicht:
[EX4]  683:          RND     MissileWKT   ;'würfeln'
[ZM1]  684:          or      ax,ax
[183]  685:          jnz     MissileNicht
[GT2]  686:          mov     dx,TSpalte
[DU4]  687:          add     dl,4            ;weiter hinten
[GL2]  688:                                  ;ausklinken
[LL1]  689:          add     bp,1Ø
[4R5]  69Ø:          cmp     WORD PTR [ErstesZweites],Ø
[TU1]  691:          jnz     KMAØ1
[LL1]  692:          add     bp,1Ø
[FF4]  693:          ; 3 Zeilen müssen frei sein
[HS2]  694: KMAØ1:   mov     cx,3
[DW3]  695: KMAØ2:   mov     ax,TSpalte
[YM4]  696:          or      al,al     ;1.Teil besetzt?
[8Z4]  697:          jz      KMAØ3     ;nein->Zeile frei
[TN4]  698:          mov     bp,[ParameterZeiger]
[P63]  699:          jmp     MissileNicht
[PH5]  7ØØ: KMAØ3:   add     bp,2Ø     ;nächste Zeile
[6V1]  7Ø1:          loop    KMAØ2
[PG4]  7Ø2:          ; Sprite 'Missile' ausgeben
[R72]  7Ø3: MissileRaus:
[7P1]  7Ø4:          sub     bp,6Ø
[KS3]  7Ø5:          mov     bx,offset pictab
[SY1]  7Ø6:          add     bx,4*2Ø
[1B4]  7Ø7:          mov     ax,[bx]     ;ax=PhaseØ
[3T2]  7Ø8:          mov     TPhaseØ,ax
[SF2]  7Ø9:          mov     ax,[bx+2]
[DM1]  71Ø:          mov     cl,4
[155]  711:          shl     ax,cl  ;al=16*Phasenanzahl
[TG3]  712:          and     dh,1       ;Phase=Ø/1
[7N1]  713:          add     dh,al
[5Y2]  714:          mov     TSpalte,dx
[ZM4]  715:          mov     ax,2Ø      ;Missile=Nr. 2Ø
[TY2]  716:          mov     TNummer,ax
[3K4]  717:          mov     ax,2Ø1h    ;2 x ausgeben
[YD2]  718:          mov     TInfo,ax
[K53]  719:          call    TeilAusgeben
[TN4]  72Ø:          mov     bp,[ParameterZeiger]
[WC5]  721:          call    AUTØØ      ;'Missileträger'
[1V2]  722:                              ;normal ausgeben
```

```
[3PØ]  723:          ret
[ØØØ]  724:
[ØX4]  725: ; Bewegung des Sprite 'Missile'
[BZ6]  726: ; es folgt der Höhe des Goldgräbers und
[LR3]  727: ; meidet Schußlinie
[ØØØ]  728:
[8G2]  729: MissileFliegt:
[GT2]  73Ø:          mov     dx,TSpalte
[6T3]  731:          cmp     dl,StartSpalte-2Ø
[EE5]  732:          jnc     MFØ1 ;Sprite kommt ins Bild
[VN1]  733:          cmp     dl,2Ø
[X15]  734:          jnc     MFØ2   ;geht nicht aus Bild
[AM5]  735: MFØ1:    mov     bp,[ParameterZeiger]
[6V1]  736:          call    AUTØØ
[3PØ]  737:          ret
[E54]  738: MFØ2:    mov     ax,TPixZeile
[8L3]  739:          mov     bx,[IPixZeile]
[9U5]  74Ø:          ; Sprite folgt Goldgräberbewegung
[7U4]  741:          cmp     ax,bx       ;gleiche Höhe
[RN2]  742:          jz      MFØ1        ;ja
[R23]  743:          jnc     MissileHoch
[CH1]  744:          add     ax,4
[7U4]  745:          cmp     ax,bx       ;gleiche Höhe
[RN2]  746:          jz      MFØ1        ;ja
[SE3]  747:          jmp     MissileRunter
[752]  748: MissileHoch:
[UM4]  749:          cmp     WORD PTR TPixZeile,16
[4Ø4]  75Ø:          jz      MFØ1        ;schon oben
[2S1]  751:          mov     si,2Ø
[4R5]  752:          cmp     WORD PTR [ErstesZweites],Ø
[5F1]  753:          jz      MHØ1
[3S1]  754:          mov     si,3Ø
[4M5]  755:          ; Test, ob hoch/runter möglich ist
[DR2]  756: MHØ1:    sub     bp,si
[Y54]  757:          cmp     BYTE PTR TSpalte,Ø
[PM4]  758:          jz      MHGeht      ;alles frei
[IA5]  759:          call    MissileCrash   ;stört es?
[V53]  76Ø:          jnz     MFØ1        ;ja
[LL1]  761:          add     bp,1Ø
[Y54]  762:          cmp     BYTE PTR TSpalte,Ø
[EX3]  763:          jnz     MFØ1        ;belegt
[FP1]  764:          sub     si,1Ø
[SD6]  765: MHGeht: push    bp      ;Zeiger freie Pos.
[MF1]  766:          push    si
[7P1]  767:          add     bp,si
[WØ5]  768:          call    TeilAusgeben     ;löschen
[ED1]  769:          pop     si
[1D1]  77Ø:          pop     bp
[NP1]  771:          COPY4   si,Ø
[AT2]  772:          mov     ax,TSpalte
[YS1]  773:          mov     ch,ah
[P22]  774:          and     ch,Øf0h
[AT1]  775:          mov     cl,ah
[HL1]  776:          shr     cl,1
[HL1]  777:          shr     cl,1
[HL1]  778:          shr     cl,1
[HL1]  779:          shr     cl,1
[2A1]  78Ø:          inc     ah
[1P1]  781:          and     ah,cl
[VM1]  782:          add     ah,ch
[991]  783:          dec     al
[ZX2]  784:          mov     TSpalte,ax
[3L1]  785:          or      al,al
[H62]  786:          jz      MHGEnde
[K53]  787:          call    TeilAusgeben
[9C1]  788: MHGEnde:
[3PØ]  789:          ret
[BL2]  79Ø: MissileRunter:
[VL1]  791:          sub     ax,4
[GV1]  792:          cmp     ax,14Ø
[824]  793:          jz      MRØØ      ;schon unten
[6N4]  794:          cmp     ax,132    ;1 oder 2 Zeilen
[U52]  795:                            ;über Boden?
[R84]  796:          jnc     MR1Ø      ;ja, also frei
[ØE3]  797:          ; Test, ob Zeilen frei
[SL1]  798:          add     bp,6Ø
[4R5]  799:          cmp     WORD PTR [ErstesZweites],Ø
[7G1]  8ØØ:          jz      MR11
[2P1]  8Ø1:          sub     bp,1Ø
[U64]  8Ø2: MR11:    call    MissileCrash
[PH4]  8Ø3:          jnz     MRØØ          ;nicht frei
[LL1]  8Ø4:          add     bp,1Ø
[173]  8Ø5:          call    MissileCrash
[PH4]  8Ø6:          jnz     MRØØ          ;nicht frei
[HN5]  8Ø7: MR1Ø:    mov     bp,[ParameterZeiger]
[W63]  8Ø8:          mov     ax,TPixZeile
[DM1]  8Ø9:          add     ax,12
[D15]  81Ø:          cmp     ax,bx      ;Schußlinie?
[ZB3]  811:          jnz     MRØ1       ;nein
[CU4]  812:          cmp     WORD PTR [gefeuert],Ø
[4Y4]  813:          jz      MRØ1       ;kein Schießen
[1S5]  814: MRØØ:    jmp     MFØ1       ;sonst normal
[RR2]  815: MRØ1:    mov     si,2Ø
[4R5]  816:          cmp     WORD PTR [ErstesZweites],Ø
[4G1]  817:          jz      MRØ2
[1S1]  818:          mov     si,1Ø
[XN2]  819: MRØ2:    add     bp,si
[Y54]  82Ø:          cmp     BYTE PTR TSpalte,Ø
[AF4]  821:          jz      MRGeht      ;alles frei
[IC5]  822:          call    MissileCrash   ;stört es?
[SZ2]  823:          jnz     MRØØ        ;ja
[LL1]  824:          add     bp,1Ø
[Y54]  825:          cmp     BYTE PTR TSpalte,Ø
[AK4]  826:          jnz     MRØØ        ;Teil stört
[ØM1]  827:          add     si,1Ø
[TV2]  828: MRGeht: push    si
[TN4]  829:          mov     bp,[ParameterZeiger]
[P45]  83Ø:          call    TeilAusgeben   ;löschen
[ED1]  831:          pop     si
[TN4]  832:          mov     bp,[ParameterZeiger]
[DR1]  833:          COPY4   Ø,si
[7P1]  834:          add     bp,si
[AT2]  835:          mov     ax,TSpalte
[YS1]  836:          mov     ch,ah
[AT1]  837:          mov     cl,ah
[P22]  838:          and     ch,Øf0h
[HL1]  839:          shr     cl,1
[HL1]  84Ø:          shr     cl,1
[HL1]  841:          shr     cl,1
[HL1]  842:          shr     cl,1
[2A1]  843:          inc     ah
[1P1]  844:          and     ah,cl
[VM1]  845:          add     ah,ch
[991]  846:          dec     al
[ZX2]  847:          mov     TSpalte,ax
[3L1]  848:          or      al,al
[F72]  849:          jz      MRGEnde
[K53]  85Ø:          call    TeilAusgeben
[GD2]  851:          mov     ax,TInfo
[T12]  852:          add     ax,1ØØh
[YD2]  853:          mov     TInfo,ax
[7D1]  854: MRGEnde:
[3PØ]  855:          ret
[ØØØ]  856:
[MF7]  857: ; Routine überprüft, ob Missile gegebenes
[1S2]  858: ; Sprite trifft
[UM6]  859: ; BP=Zeiger auf zu untersuchendes Teil,
[TZ4]  86Ø: ; dl=Missile-SpaltenPosition
[NU5]  861: ; wenn kein Crash --> Z-Flag gesetzt!
[ØØØ]  862:
[GA2]  863: MissileCrash:
[XF1]  864:          push    ax
[3G1]  865:          push    dx
[AT2]  866:          mov     ax,TSpalte
[3L1]  867:          or      al,al
[4V4]  868:          jz      KeinMCrash  ;kein Teil da
[5L1]  869:          add     al,1Ø
[RB4]  87Ø:          cmp     al,dl       ;Teil davor?
[PB3]  871:          jc      KeinMCrash  ;ja
[CL1]  872:          add     dl,2Ø
```

Listing 10. Sprites werden mit Routinen aus »ggjumps.inc« gesteuert.

```
[DY4]  873:          cmp     dl,al       ;Teil dahinter?
[PB3]  874:          jc      KeinMCrash  ;ja
[AN1]  875:          mov     ax,1
[6N2]  876:          jmp     IstMCrash
[912]  877: KeinMCrash:
[GV1]  878:          xor     ax,ax
[TT1]  879: IstMCrash:
[XD1]  880:          pop     dx
[ZM1]  881:          or      ax,ax
[RD1]  882:          pop     ax
[3P0]  883:          ret
[000]  884:
[LW6]  885: ; Bewegung des Sprite 'schießender Gegner'
[HF7]  886: ; Dieses Sprite schießt auf Goldgräber,
[937]  887: ; sobald die Schußlinie frei ist und die
[5C6]  888: ; Schußweite unterschritten wird
[000]  889:
[KT2]  890: GegnerSchiesst:
[AT2]  891:          mov     ax,TSpalte
[ZA5]  892:          cmp     al,145      ;in Schußweite?
[CL3]  893:          jnc     GSNormal    ;nein
[055]  894:          cmp     al,ISpaltebis+5  ;zu nahe?
[9G5]  895:          jnc     GS01     ;nein->Schußversuch
[XK1]  896: GSNormal:
[TN4]  897:          mov     bp,[ParameterZeiger]
[WV1]  898:          jmp     AUTØØ
[MK4]  899: GS01:    mov     bx,[IPixZeile]
[W63]  900:          mov     ax,TPixZeile
[KF5]  901:          cmp     ax,bx    ;im Trefferbereich?
[2S3]  902:          jz      GSVersuch        ;ja
[XL1]  903:          sub     bx,4
[KF5]  904:          cmp     ax,bx    ;im Trefferbereich?
[2S3]  905:          jz      GSVersuch        ;ja
[XL1]  906:          sub     bx,4
[KF5]  907:          cmp     ax,bx    ;im Trefferbereich?
[8Ø4]  908:          jnz     GSNormal         ;nein
[PW1]  909: GSVersuch:
[XR5]  910:          add     ax,8     ;ax=Schußpixelzeile
[BH3]  911:          mov     ah,ISpalteBis+1
[7S1]  912:          xchg    al,ah
[3V3]  913:          mov     cx,WasGetroffen
[VE5]  914:          call    UPGraf      ;Schußbahn frei?
[AT2]  915:          mov     ax,TSpalte
[D35]  916:          dec     bx          ;bl=Trefferspalte
[YR1]  917:          cmp     bl,al
[9V4]  918:          jnz     GSNormal  ;kein Schuß
[332]  919: GSSchuß:
[W63]  920:          mov     ax,TPixZeile
[CG1]  921:          add     al,8
[L62]  922:          mov     ah,ink1
[755]  923:          xchg    al,ah     ;ah=Zeile,al=Farbe
[SD4]  924:          sub     bx,13     ;log.Spalte-13
[6A5]  925:          and     bx,Øfffeh     ;Bit Ø löschen
[6P2]  926:          shl     bx,1          ;*2
[XL2]  927:          add     bx,3          ;+3
[324]  928:          mov     dx,2*ISpalteBis-2Ø
[UL4]  929:          xchg    bx,dx      ;bx=von ,dx=bis
[772]  930:                             ;Pixelspalte
[SZ2]  931:          PUSHEN  <AX,BX,DX>
[T23]  932:          mov     cx,HoriLine
[132]  933:          call    UPGraf
[PT3]  934:          mov     bx,offset implod
[HZ1]  935:          add     bx,6*72
[XK4]  936:          mov     si,offset gegnerknall
[TH3]  937:          mov     dx,TeileFormat
[W63]  938:          mov     ax,TPixZeile
[9G1]  939:          add     al,5
[TM3]  940:          mov     ah,ISpalteVon-8
[1L1]  941:          shr     ah,1
[7S1]  942:          xchg    al,ah
[EN1]  943:          mov     cx,1
[5W1]  944:          call    expl1
[5V2]  945:          POPEN   <DX,BX,AX>
[XB2]  946:          mov     al,paper
[T23]  947:          mov     cx,HoriLine
[132]  948:          call    UPGraf
[9Y4]  949:          mov     ax,2     ;-2 Energiepunkte
[5G2]  950:          call    CREMinus
[495]  951:          jmp     GSNormal ;Gegner ausgeben
[000]  952:
[000]  953:
[R58]  954: ; Sprungleisten für Bewegung, Zusammenstoß
[1K4]  955: ; und Treffer eines Sprite,
[M97]  956: ; gegebenenfalls BP=Zeiger getroffenes Teil
[S94]  957: ;                BX=Trefferspalte
[367]  958: ; [ErstesZweites]=Ø/1 für 1/2.Teil in Zeile
[000]  959:
[LS1]  960: AusgabeUP:
[Y25]  961:          mov     cx,offset AusgabeLeiste
[KV1]  962:          jmp     CUP01
[WE1]  963: CrashUP:
[RM4]  964:          mov     cx,offset CrashLeiste
[KV1]  965:          jmp     CUP01
[YT1]  966: TrefferUP:
[T55]  967:          mov     cx,offset TrefferLeiste
[BE4]  968:          mov     [TrefferSpalte],bx
[1A5]  969:          ; Sprung berechnet sich aus der
[2H2]  970:          ; Spritenummer
[1U5]  971: CUP01:   mov     [ParameterZeiger],bp
[FV2]  972:          mov     dx,TNummer
[GR1]  973:          add     cx,dx
[GR1]  974:          add     cx,dx
[GR1]  975:          add     cx,dx
[LC1]  976:          call    cx
[TN4]  977:          mov     bp,[ParameterZeiger]
[3P0]  978:          ret
[4H2]  979: TrefferLeiste:
[XD5]  980:          ;ersten 25 Teile Standardtreffer
[AA1]  981:          REPT    25
[FX3]  982:          jmp     Trefferallgemein
[HP0]  983:          ENDM
[ML5]  984:          jmp     TrefferTeileTausch  ;Nr. 25
[3A1]  985:          REPT    32
[FX3]  986:          jmp     Trefferallgemein
[HP0]  987:          ENDM
[9D5]  988:          jmp     TrefferAtombombe    ;Nr. 58
[R51]  989:          REPT    8
[FX3]  990:          jmp     Trefferallgemein
[HP0]  991:          ENDM
[2X4]  992:          jmp     TrefferFresser          ;67
[T91]  993:          REPT    1Ø
[FX3]  994:          jmp     TrefferAllgemein
[HP0]  995:          ENDM
[000]  996:
[M42]  997: CrashLeiste:
[F51]  998:          REPT    4
[583]  999:          jmp     CRTØØ    ;Nr. Ø-4
[HP0] 1000:          ENDM
[V91] 1001:          REPT    2Ø
[FG3] 1002:          jmp     CrashAllgemein
[HP0] 1003:          ENDM
[484] 1004:          jmp     CrashTor        ;Nr. 24
[FG3] 1005:          jmp     CrashAllgemein
[FG3] 1006:          jmp     CrashAllgemein
[FG3] 1007:          jmp     CrashAllgemein
[D94] 1008:          jmp     CrashMine       ;Nr. 28
[PA1] 1009:          REPT    29
[FG3] 1010:          jmp     CrashAllgemein
[HP0] 1011:          ENDM
[HU4] 1012:          jmp     CrashAtombombe  ;Nr. 58
[FG3] 1013:          jmp     CrashAllgemein
[FG3] 1014:          jmp     CrashAllgemein
[FG3] 1015:          jmp     CrashAllgemein
[8C4] 1016:          jmp     CrashFeder      ;Nr. 62
[FG3] 1017:          jmp     CrashAllgemein
[FG3] 1018:          jmp     CrashAllgemein
[X94] 1019:          jmp     CrashLoch       ;Nr. 65
[1A4] 1020:          jmp     CrashLoch       ;Nr. 66
[T9L] 1021:          REPT    1Ø
[FG3] 1022:          jmp     CrashAllgemein
[HP0] 1023:          ENDM
[000] 1024:
[TF2] 1025: AusgabeLeiste:
[N51] 1026:          REPT    7
[WV1] 1027:          jmp     AUTØØ
[HP0] 1028:          ENDM
[MY3] 1029:          jmp     AUTRak          ;Nr. 7
[Z91] 1030:          REPT    12
[WV1] 1031:          jmp     AUTØØ
[HP0] 1032:          ENDM
[4N4] 1033:          jmp     MissileFliegt   ;Nr. 2Ø
[WV1] 1034:          jmp     AUTØØ
[WV1] 1035:          jmp     AUTØØ
[5G4] 1036:          jmp     BombeFaellt     ;Nr. 23
[WV1] 1037:          jmp     AUTØØ
[WV1] 1038:          jmp     AUTØØ
[XS4] 1039:          jmp     KlinktBombeAus  ;Nr. 26
[F25] 1040:          jmp     KlinktMissileAus ;Nr. 27
[WV1] 1041:          jmp     AUTØØ
[WV1] 1042:          jmp     AUTØØ
[ST4] 1043:          jmp     GegnerSchiesst  ;Nr. 3Ø
[X91] 1044:          REPT    3Ø
[WV1] 1045:          jmp     AUTØØ
[HP0] 1046:          ENDM
[KN4] 1047:          jmp     AUTTarnkappe    ;Nr. 61
[WV1] 1048:          jmp     AUTØØ
[WV1] 1049:          jmp     AUTØØ
[WV1] 1050:          jmp     AUTØØ
[M54] 1051:          jmp     AUTLoch1        ;Nr. 65
[T91] 1052:          REPT    1Ø
[WV1] 1053:          jmp     AUTØØ
[HP0] 1054:          ENDM
```

Listing 10. Sprites werden mit Routinen aus »ggjumps.inc« gesteuert.

```
[1F3]    1: ; Programm: ggequ.inc
[EP5]    2: ; Funktion: EQU-Vereinbarungen zu
[143]    3: ;           gghaupt.asm
[913]    4: ; Sprache:  TASM 2.Ø1
[VT3]    5: ; Autor:    Detlef Paulin
[7A4]    6: ; (c) 1992 Widuch GmbH & Co. KG
[000]    7:
[2W5]    8: ; Farbenmasken für AND-Verknüpfung
[000]    9:
[5X5]   1Ø: paper   equ     Ø       ;Hintergrundfarbe
[CT4]   11: ink1    equ     55h     ;Farbwert 1
[RU4]   12: ink2    equ     Øaah    ;Farbwert 2
[ZV4]   13: ink3    equ     Øffh    ;Farbwert 3
[000]   14:
[TW3]   15: ; Grafikunterprogramme
[000]   16:
[CK3]   17: XORmitWord      equ     Ø
[BG3]   18: XORmitBYTE      equ     1
[N13]   19: CHAR8x5         equ     2
[MC3]   2Ø: TextModus       equ     3
[GM3]   21: GrafikModus     equ     4
[T83]   22: HoriLine        equ     5
[493]   23: VertLine        equ     6
[ØP3]   24: WasGetroffen    equ     7
[000]   25:
[1Z5]   26: ; Zugriff auf TeileTabelle (TeileTab)
[000]   27:
[5T5]   28: TSpalte         equ     DS:[BP]     ;byte
[CG5]   29: TPhase          equ     DS:[BP+1]   ;byte
[PS5]   3Ø: TNummer         equ     DS:[BP+2]   ;word
[2L5]   31: TPhaseØ         equ     DS:[BP+4]   ;word
[7V5]   32: TPixZeile       equ     DS:[BP+6]   ;word
[RD5]   33: TInfo           equ     DS:[BP+8]   ;word
[000]   34:
[TR6]   35: ; Zugriff auf Spriteparameter (s. uØ1.pic)
[000]   36:
[AZ3]   37: TPunkte         equ     [BX]
[KZ3]   38: TExpl           equ     [BX+2]
[D94]   39: TWieOft         equ     [BX+4]
[YA4]   4Ø: CRPunkte        equ     [BX+6]
[CA4]   41: CREnergie       equ     [BX+8]
[RK4]   42: CRSchuss        equ     [BX+1Ø]
[000]   43:
[AE3]   44: ; diverse Konstanten
[000]   45:
[V14]   46: StartSpalte     equ     169
[TE3]   47: StartPhase      equ     1
[WX5]   48: TeileFormat     equ     ØC Ø3H  ;12x3 WORD
[BX5]   49: LogoFormat      equ     3ØØAH  ;48x1Ø WORD
[6G3]   5Ø: InfoLaenge      equ     12
[000]   51:
[P26]   52: ; Parameter der Raumunterschriften
[000]   53:
[636]   54: TextZei         equ     166     ;PixelZeile
[C25]   55: TextSp          equ     1       ;Spalte
[VP5]   56: ColorG          equ     ink2    ;Grundbild
[4R5]   57: ColorE          equ     ink1    ;Extrabild
[5C5]   58: ColorI          equ     ink2    ;String
[1A5]   59: ColorM          equ     ink2    ;Rahmen
[000]   6Ø:
[YW6]   61: ; Nummer des zufällig gewählten Sprite
[000]   62:
[614]   63: ZufallsTeil     equ     128
[000]   64:
[PP5]   65: ; Parameter zum Goldgräber (Ich)
[000]   66:
[MN3]   67: ISpalteVon      equ     39
[CM3]   68: ISpalteBis      equ     5Ø
[293]   69: MaxTempo        equ     Ø
[RD3]   7Ø: MinTempo        equ     63
[LT3]   71: MaxEnergie      equ     512
[SK3]   72: MaxMuni         equ     512
[000]   73:
[755]   74: ; Scan-Code verwendeter Tasten
[000]   75:
[EY5]   76: ShiftLinks      equ     2     ;Shift-Maske
[TD3]   77: Kursorh         equ     72
[5F3]   78: Kursort         equ     8Ø
[KE3]   79: Kursorl         equ     75
[GF3]   8Ø: Kursorr         equ     77
[383]   81: Return          equ     28
[683]   82: ESCTaste        equ     1
[D54]   83: STaste          equ     6Ø      ;F2
[PK4]   84: PanikTaste      equ     59      ;F1
[000]   85:
[H26]   86: ; Maß der Wahrscheinlichkeiten für
[9G4]   87: ; entsprechendes Auftreten
[000]   88:
[9C3]   89: BombenWKT       equ     7
[4N3]   9Ø: MissileWKT      equ     56
```

Listing 11. Konstanten werden in »ggequ.inc« definiert.

```
[ØF3]    1: ; Programm: ggenv.inc
[497]    2: ; Funktion: Routinen zur Programmumgebung
[913]    3: ; Sprache:  TASM 2.Ø1
[VT3]    4: ; Autor:    Detlef Paulin
[7A4]    5: ; (c) 1992 Widuch GmbH & Co. KG
[P3Ø]    6: ;
[2SØ]    7:          .DATA
[000]    8:
[SL6]    9: ;Fehlermeldungen nach Grafikkartentest
[000]   1Ø:
[LA6]   11: falschegrafik   db  13,1Ø,'Sie benötigen '
[HH3]   12:                 db  'eine Grafikkarte!$'
[ØX5]   13: warnunggrafik   db  13,1Ø,'Grafikkarte '
[C13]   14:                 db  'nicht erkannt.'
[000]   15:
[SX6]   16: ; Nummer der erkannten oder als Parameter
[XE4]   17: ; gewünschten Grafikkarte
[GP4]   18: ; 1=Hercules, 2=CGA (EGA), 3=VGA
[000]   19:
[043]   2Ø: grafikkarte    db  Ø
[000]   21:
[RA7]   22: ; von Programmstück ab grtest8: ermittelte
[VY5]   23: ; hardwareabhängige Zeitkonstante
[000]   24:
[RW2]   25: tcount          dw  Ø,Ø
[000]   26:
[7M7]   27: ; zusätzliche Tempokorrektur bei besonders
[6E3]   28: ; schnellen Rechnern
[000]   29:
[RS2]   3Ø: hightech        dw  Ø
[000]   31:
[ES7]   32: ; Segmentadresse für angeforderten Speicher
[ØT6]   33: ; wenn Ø, dann nicht genug Speicher für F1
[000]   34:
[483]   35: panikpuffer     dw  Ø
[000]   36:
[4K6]   37: ; Variable und Konstanten für den Aufruf
[CV5]   38: ; von COMMAND.COM nach Drücken von F1
[000]   39:
[9B4]   4Ø: Comsp           db  'COMSPEC='
[FM2]   41: KDZ             db  Ø,13
[N74]   42: PanikDatei      db  128 dup (Ø)
[HB5]   43: PrPaBl          dw  Ø,KDZ,Ødata,Ø,Ø,Ø,Ø,Ø
[000]   44:
[FT0]   45:          .CODE
[000]   46:
[YE6]   47: ; Programmteil führt Test auf Grafikkarte
[PK6]   48: ; durch und bestimmt hardwareabhängige
[YU2]   49: ; Zeitkonstante
[000]   5Ø:
[G32]   51: grtest  proc
[3V3]   52:         mov     ah,[grafikkarte]
[YD4]   53:         or      ah,ah       ;Grafik bereits
[FM2]   54:                             ;als Parameter?
[UØ5]   55:         jnz     grtest8     ;ja-> kein Test
[GV1]   56:         xor     ax,ax
[3V1]   57:         mov     es,ax
[F95]   58:         mov     byte ptr [grafikkarte],2
[FC5]   59:         mov     ax,es:[463H]  ;Portadresse
[LØ3]   6Ø:                             ;Video-Controller
[TM4]   61:         cmp     ax,3b4h     ;3b4h=MDA/Herc
[NL2]   62:                             ;3d4h=CGA,EGA,VGA
[HE2]   63:         jnz     grtest4
[5P4]   64:         mov     dx,Ø3bah    ;Test, ob MDA
[5K4]   65:         mov     bl,5        ;oder Hercules
[6L1]   66:         in      al,dx
[SS4]   67:         and     al,8Øh      ;Bit 7 isolieren
[2T1]   68:         mov     ah,al
[D52]   69:         mov     cx,8ØØØh
[Y26]   7Ø: grtest1:in      al,dx       ;erneutes Lesen
[SS4]   71:         and     al,8Øh      ;Bit 7 isolieren
[BY3]   72:         cmp     al,ah       ;Änderung?
[XA3]   73:         je      grtest2     ;nein
[B91]   74:         dec     bl
[982]   75:         jz      grtest3
[L47]   76: grtest2:loop    grtest1     ;kein Bit-Wechsel
[6R2]   77:         mov     ah,9        ;MDA
[Y35]   78:         mov     dx,offset falschegrafik
[3K1]   79:         int     21h
[W94]   8Ø:         mov     ah,4ch      ;und Schluß
[3K1]   81:         int     21h
[723]   82: grtest3:mov     al,1
[MT5]   83:         mov     [grafikkarte],al  ;Hercules
[PD2]   84:         jmp     grtest8
[RK5]   85: grtest4:mov     ah,es:[Ø4a8h]  ;Ø=CGA
[FK1]   86:         or      ah,ah
[PY4]   87:         jnz     grtest5     ;<>Ø=EGA/VGA
[PD2]   88:         jmp     grtest8
[LS6]   89: grtest5:mov     ah,es:[48ah]   ;Ø=keine VGA
[FK1]   9Ø:         or      ah,ah
[N82]   91:         jz      grtest6
[G95]   92:         mov     byte ptr [grafikkarte],3
[PD2]   93:         jmp     grtest8
[BD5]   94: grtest6:mov     ax,12ØØh        ;Status
[4Z1]   95:         mov     bx,1Øh
[YH1]   96:         int     1Øh
[HV3]   97:         cmp     bl,1Øh          ;bl<>1ØH
[EN2]   98:                                 ;deutet auf
    EGA
[715]   99:         jz      grtest7         ;alles unkla
    r
[PD2]  1ØØ:         jmp     grtest8
[L23]  1Ø1: grtest7:mov     ah,9
[H85]  1Ø2:         mov     dx,offset warnunggrafik
[3K1]  1Ø3:         int     21h
[5A3]  1Ø4: grtest8:xor     ax,ax
[7P3]  1Ø5:         mov     ds:[tcount],ax
[RT3]  1Ø6:         mov     ds:[hightech],ax
[HL1]  1Ø7:         int     1ah
[HV1]  1Ø8:         mov     cx,dx
[UY2]  1Ø9: grtest9:push    cx
[HL1]  11Ø:         int     1ah
[VD1]  111:         pop     cx
[ZT1]  112:         cmp     cx,dx
[192]  113:         jz      grtest9
[PB1]  114:         inc     cx
[PB1]  115:         inc     cx
[LN4]  116: grtesta:inc     ds:[tcount]
[1G1]  117:         push    cx
[HL1]  118:         int     1ah
[VD1]  119:         pop     cx
[ZT1]  12Ø:         cmp     cx,dx
[5G2]  121:         jnz     grtesta
[9N3]  122:         mov     cx,ds:[tcount]
[MS1]  123:         mov     ax,91
[RD1]  124:         mul     cx
[7A4]  125:         mov     bx,5ØØØ         ;1/1ØØØ sec
[7D1]  126:         div     bx
[KB1]  127:         inc     ax
[7P3]  128:         mov     ds:[tcount],ax
[DM1]  129:         mov     cl,4
[VT1]  13Ø:         shr     ax,cl
[YL1]  131:         cmp     ax,7
[RU1]  132:         jc      grtt1Ø
[FN1]  133:         mov     ax,6
[T25]  134: grtt1Ø: mov     ds:[hightech],ax
[FV3]  135:         mov     al,[grafikkarte]
[452]  136:         mov     bx,1ØØØh
[PK1]  137:         cmp     al,3
[EW1]  138:         jz      grtt11
[1X4]  139:         mov     bx,Ø4ØØh        ;Speicher für
[1X5]  14Ø: grtt11: mov     ah,48h          ;Panik-Taste
[374]  141:         int     21h             ;anfordern
[5L3]  142:         jc      grtt12          ;Fehler
[ØN5]  143:         mov     [panikpuffer],ax  ;Segment-
[2F3]  144:                                 ;adresse des Puffers
[BZ1]  145: grtt12: ret
[F12]  146: grtest  endp
[000]  147:
[MP6]  148: ; Verteiler für Grafikunterprogramme
[000]  149:
[H02]  15Ø: upgraf  proc
[XF1]  151:         push    ax
[FV3]  152:         mov     al,[grafikkarte]
[MK1]  153:         cmp     al,1
[Y42]  154:         jz      upgrafØ
[NK1]  155:         cmp     al,2
[252]  156:         jz      upgraf1
[RD1]  157:         pop     ax
[342]  158:         call    VGA_UP
[3P0]  159:         ret
[DS2]  16Ø: upgrafØ:pop     ax
[CF2]  161:         call    HERC_UP
[3PØ]  162:         ret
[HS2]  163: upgraf1:pop     ax
```

```
[X22]  164:          call    CGA_UP
[3PØ]  165:          ret
[GY1]  166: upgraf   endp
[ØØØ]  167:
[X47]  168: ; Programm rette Bildschirminhalt und ruft
[GK2]  169: ; COMMAND.COM auf
[TE7]  170: ; Mit Kommando EXIT kommt man wieder zurück
[ØØØ]  171:
[5C3]  172: PanikProgramm   PROC
[SY3]  173:          mov     ax,[panikpuffer]
[3X4]  174:          or      ax,ax       ;genug Speicher?
[DX2]  175:          jnz     PPLos       ;ja
[VP2]  176:          ret                 ;nein
[RR2]  177: PPLos:   push    ds
[SY3]  178:          mov     ax,[panikpuffer]
[3V1]  179:          mov     es,ax
[FV3]  180:          mov     al,[grafikkarte]
[KK4]  181:          xor     di,di       ;es:[di]=Ziel
[VU1]  182:          xor     si,si
[CU3]  183:          cmp     al,1        ;Hercules?
[PT2]  184:          jz      PPHerc      ;ja
[EW2]  185:          cmp     al,2        ;CGA?
[TM2]  186:          jz      PPCGA       ;ja
[V63]  187:          mov     ax,ØAØØØH   ;VGA
[UØ5]  188:          mov     ds,ax       ;ds:[si]=Quelle
[MF1]  189:          push    ds
[MF1]  190:          push    si
[PF1]  191:          push    es
[PE1]  192:          push    di
[D52]  193:          mov     cx,8ØØØH
[LZ1]  194:          rep     movsw
[NC2]  195:          mov     ax,@data
[1V1]  196:          mov     ds,ax
[H83]  197:          mov     cx,TextModus
[132]  198:          call    upgraf
[ST3]  199:          call    PanikDateiRufen
[CH3]  200:          mov     cx,GrafikModus
[132]  201:          call    upgraf
[ED1]  202:          pop     si
[ED1]  203:          pop     ds
[GC1]  204:          pop     di
[GD1]  205:          pop     es
[D52]  206:          mov     cx,8ØØØh
[LZ1]  207:          rep     movsw
[432]  208:          jmp     PPEnde
[U34]  209: PPCGA:   mov     ax,ØB8ØØH   ;ja
[1V1]  210:          mov     ds,ax
[FW4]  211:          push    es          ;ds:[si]=Quelle
[UE4]  212:          push    di          ;es:[di]=Ziel
[MF1]  213:          push    ds
[MF1]  214:          push    si
[752]  215:          mov     cx,2ØØØh
[LZ1]  216:          rep     movsw
[NC2]  217:          mov     ax,@data
[1V1]  218:          mov     ds,ax
[H83]  219:          mov     cx,TextModus
[132]  220:          call    upgraf
[ST3]  221:          call    PanikDateiRufen
[CH3]  222:          mov     cx,GrafikModus
[132]  223:          call    upgraf
[VE4]  224:          pop     di          ;Quelle<->Ziel
[GD1]  225:          pop     es
[ED1]  226:          pop     si
[ED1]  227:          pop     ds
[752]  228:          mov     cx,2ØØØh
[LZ1]  229:          rep     movsw
[432]  230:          jmp     PPEnde
[7N3]  231: PPHerc:  mov     ax,ØB8ØØH
[1V1]  232:          mov     ds,ax
[FW4]  233:          push    es          ;ds:[si]=Quelle
[UE4]  234:          push    di          ;es:[di]=Ziel
[MF1]  235:          push    ds
[MF1]  236:          push    si
[LD3]  237:          mov     cx,1ØØØH    ;1.Bank
[LZ1]  238:          rep     movsw
[LD4]  239:          add     si,2ØØØH    ;3.Bank retten
[652]  240:          mov     cx,1ØØØH
[LZ1]  241:          rep     movsw
[NC2]  242:          mov     ax,@data
[1V1]  243:          mov     ds,ax
[H83]  244:          mov     cx,TextModus
[132]  245:          call    upgraf
[ST3]  246:          call    PanikDateiRufen
[CH3]  247:          mov     cx,GrafikModus
[132]  248:          call    upgraf
[VE4]  249:          pop     di          ;Quelle<->Ziel
[MD4]  250:          pop     es          ;es:[di]=Ziel
[7U4]  251:          pop     si          ;ds:[si]=Quelle
[ED1]  252:          pop     ds
[7K3]  253:          mov     cx,1ØØØh    ;1. Bank
[LZ1]  254:          rep     movsw
[832]  255:          sub     si,2ØØØH
[KG4]  256:          mov     cx,1ØØØH    ;2. Bank wie 1.
[LZ1]  257:          rep     movsw
[BK3]  258:          mov     cx,1ØØØH    ;3. Bank
[LZ1]  259:          rep     movsw
[WE4]  260:          sub     si,2ØØØH    ;4. Bank wie 3.
[652]  261:          mov     cx,1ØØØH
[LZ1]  262:          rep     movsw
[ZR2]  263: PPEnde:  pop     ds
[TL1]  264:          WARTE   1ØØØ
[3PØ]  265:          ret
[ØØØ]  266:
[DV2]  267: PanikDateiRufen:
[ZF1]  268:          push    bx
[1G1]  269:          push    cx
[3G1]  270:          push    dx
[PE1]  271:          push    di
[MF1]  272:          push    si
[7F1]  273:          push    bp
[NC2]  274:          mov     ax,@data
[1V1]  275:          mov     ds,ax
[MF1]  276:          push    ds
[PF1]  277:          push    es
[3V1]  278:          mov     es,ax
[962]  279:          mov     ax,4BØØH
[8G4]  280:          mov     dx,offset PanikDatei
[V53]  281:          mov     cs:merkss,ss
[K53]  282:          mov     cs:merksp,sp
[AS3]  283:          mov     bx,offset PrPaBl
[3K1]  284:          int     21h
[EMØ]  285:          cli
[263]  286:          mov     ss,cs:merkss
[P53]  287:          mov     sp,cs:merksp
[6PØ]  288:          sti
[GD1]  289:          pop     es
[ED1]  290:          pop     ds
[1D1]  291:          pop     bp
[ED1]  292:          pop     si
[GC1]  293:          pop     di
[XD1]  294:          pop     dx
[VD1]  295:          pop     cx
[TD1]  296:          pop     bx
[3PØ]  297:          ret
[692]  298: merkss   dw      ?
[X82]  299: merksp   dw      ?
[4A3]  300: PanikProgramm    ENDP
[ØØØ]  301:
[WZ6]  302: ; Freigeben des restlichen Hauptspeichers
[8X3]  303: ; nach dem Programmstart
[ØØØ]  304:
[A82]  305: SpeicherFrei:
[MV1]  306:          mov     bx,ss
[4V1]  307:          mov     ax,es
[UT1]  308:          sub     bx,ax
[CV1]  309:          mov     ax,sp
[DM1]  310:          mov     cl,4
[VT1]  311:          shr     ax,cl
[BR1]  312:          add     bx,ax
[MB1]  313:          inc     bx
[RY1]  314:          mov     ah,4ah
```

```
[3K1]  315:          int     21h
[3PØ]  316:          ret
[ØØØ]  317:
[6M4]  318: ; Pfad für COMMAND.COM suchen
[ØØØ]  319:
[AF2]  320: CompspecSuchen:
[MF1]  321:          push    ds
[H54]  322:          mov     ah,62h      ;PSP holen
[YA4]  323:          int     21h         ;bx=Segment
[2V1]  324:          mov     ds,bx
[ØØ2]  325:          mov     si,2ch
[DK4]  326:          lodsw                ;ax=Umgebungs-
[CL2]  327:                               ;stringsegment
[1V1]  328:          mov     ds,ax
[VU1]  329:          xor     si,si
[NC2]  330:          mov     ax,@data
[3V1]  331:          mov     es,ax
[ULØ]  332:          cld
[ST3]  333: CSØØ:    mov     al,ds:[si]
[4K4]  334:          cmp     al,1        ;Endezeichen?
[ZW2]  335:          je      GEFEnd      ;ja
[QV1]  336:          mov     bx,si
[2P4]  337:          xor     cx,cx       ;cx=Zähler
[VA2]  338: CSØ2:    inc     cx
[CVØ]  339:          lodsb
[TT1]  340:          cmp     al,'='
[2F4]  341:          jne     CSØ2        ;noch kein '='
[TE4]  342:          cmp     cx,8        ;8 Byte lang?
[CM4]  343:          je      CS2Ø        ;->vergleichen
[KU1]  344: CSØ3:    lodsb
[1R3]  345:          or      al,al       ;Ø suchen
[SN1]  346:          jnz     CSØ3
[XK4]  347:          jmp     CSØØ        ;und von vorn
[DU2]  348: CS2Ø:    mov     si,bx
[6P3]  349:          mov     di,offset comsp
[NN1]  350:          mov     cx,8
[TC4]  351:          repe    cmpsb       ;COMSPEC=?
[9T3]  352:          je      Gefunden    ;gleich
[DM1]  353:          jmp     CSØ3
[8G1]  354: Gefunden:
[VE4]  355:          mov     di,offset PanikDatei
[ZX1]  356: GEFØØ:   lodsb
[NXØ]  357:          stosb
[LG4]  358:          or      al,al       ;Endezeichen?
[NB3]  359:          jnz     GEFØØ       ;nein
[SP2]  360: GEFEnd:  pop     ds
[3PØ]  361:          ret
[ØØØ]  362:
[1V5]  363: ; Auswerten eventueller Parameter
[EP3]  364: ; beim Programmaufruf
[ØØØ]  365:
[1Y2]  366: AufrufParameter:
[MF1]  367:          push    ds
[PF1]  368:          push    es
[ED1]  369:          pop     ds
[ØZ1]  370:          mov     si,8Øh
[CN1]  371:          mov     bx,1
[GV1]  372:          xor     ax,ax
[ULØ]  373:          cld
[145]  374:          lodsb                ;al=Parameterlänge
[HF3]  375:          or      al,al       ;Länge=Ø?
[7N2]  376:          jz      APEnde      ;ja
[4Ø4]  377:          mov     cx,ax       ;cx=Länge
[3U1]  378: APØ1:    lodsb
[UL4]  379:          or      al,2Øh      ;Kleinbuchstabe
[1T1]  380:          cmp     al,'1'
[FK4]  381:          jz      APØ12       ;Grafik wird vom
[7Z2]  382:                              ;Nutzer bestimmt
[3T1]  383:          cmp     al,'2'
[9K1]  384:          jz      APØ12
[5T1]  385:          cmp     al,'3'
[9K1]  386:          jz      APØ12
[7P2]  387: APØ11:   loop    APØ1
[1R2]  388: APEnde:  pop     ds
[8Y3]  389:          mov     [Grafikkarte],ah
[3PØ]  390:          ret
[WP2]  391: APØ12:   sub     al,48
[R55]  392:          mov     ah,al       ;AH = Grafiknummer
[ØR1]  393:          jmp     APØ11
```

Listing 12. Die Auswahl der Grafikkarte und die Auswertung der Kommandozeilenparameter erfolgt in »ggenv.inc«.

```
[H34]    1: ; Programm: ggdivers.inc
[MZ5]    2: ; Funktion: ergänzende Routinen zu
[143]    3: ;           gghaupt.asm
[913]    4: ; Sprache:  TASM 2.Ø1
[VT3]    5: ; Autor:    Detlef Paulin
[7A4]    6: ; (c) 1992 Widuch GmbH & Co. KG
[ØØØ]    7:
[XD2]    8: ; Name:     Sound
[WB6]    9: ; Funktion: Ausgabe eines Geräusches
[RZ6]   10: ; Eingang:  BX=Adresse der Geräuschtabelle
[UZ5]   11: ;           [BX]  =Längenkonstante (in ms)
[WØ4]   12: ;           [BX+2]=Frequenz 1
[EK4]   13: ;           [BX+4]=Frequenz 2 ....
[5A5]   14: ;           [BX+n]=Ø, Endekennzeichen
[A04]   15: ; Rettet:   AX,BX,CX,DX,DI
[ØØØ]   16:
[KY1]   17: Sound    PROC
[TC3]   18:          PUSHEN  <AX,CX,DX,DI>
[2V1]   19:          mov     si,bx
[TV2]   20: Sound1:  add     si,2
[2D2]   21:          mov     di,[si]
[GK1]   22:          cmp     di,Ø
[NE2]   23:          je      SoundEnde
[265]   24:          cmp     WORD PTR [SoundSpiel],Ø
[AC2]   25:          jz      SoundOff
[CW1]   26:          mov     al,182
[ØØ2]   27:          out     43h,al
[CZ1]   28:          mov     dx,12h
[M92]   29:          mov     ax,34dch
[YB1]   30:          div     di
[XZ1]   31:          out     42h,al
[6T1]   32:          mov     al,ah
[XZ1]   33:          out     42h,al
[2R1]   34:          in      al,61h
[2T1]   35:          mov     ah,al
[XD1]   36:          or      al,3
[YZ1]   37:          out     61h,al
[GY1]   38:          WARTE   [bx]
[6T1]   39:          mov     al,ah
[YZ1]   40:          out     61h,al
[MK2]   41: Sound2:  WARTE   1
[B32]   42:          jmp     sound1
[ØU1]   43: SoundEnde:
[H63]   44:          POPEN   <DI,DX,CX,AX>
[3PØ]   45:          ret
[WL1]   46: SoundOff:
[GY1]   47:          WARTE   [bx]
[B32]   48:          jmp     sound1
[HW1]   49: Sound    ENDP
[ØØØ]   50:
[UM6]   51: ; Erzeugt Pause von ax ms und bearbeitet
[VG3]   52: ; Sondertasten (F1, F2)
[ØØØ]   53:
[M22]   54: WarteX   PROC
[1H1]   55:          WARTE   ax
[AD1]   56:          PLAYKEY
[3PØ]   57:          ret
```

```
[LØ2]   58: WarteX   ENDP
[ØØØ]   59:
[VR6]   60: ; Wartet, bis eine Taste gerückt wird und
[H85]   61: ; bearbeitet Sondertasten (F1, F2)
[ØØØ]   62:
[ØB1]   63: WTaste:
[9K1]   64:          WARTE   1ØØ
[AD1]   65:          PLAYKEY
[UZ1]   66:          jz      WTaste
[3PØ]   67:          ret
[ØØØ]   68:
[2U3]   69: ; Name:      Stringeingabe
[U76]   70: ; Funktion: Eingabe eines Strings mit
[KB5]   71: ; Eingang:  AH=Pixelzeile, AL=Spalte
[9H2]   72: ;           BX=LÄnge
[K96]   73: ; Ausgang:  Ablage in IPuffer und Ausgabe
[G44]   74: ;           auf dem Bildschirm
[ØØØ]   75:
[RA3]   76: StringEingabe    PROC
[FV1]   77:          mov     cx,bx
[M53]   78:          mov     [ILaenge],bx
[8Ø5]   79:          mov     WORD PTR [IPosition],Ø
[W63]   80:          mov     [IBSPos],ax
[LV1]   81:          xor     bx,bx
[XL6]   82: STRIØØ:  mov     BYTE PTR IPuffer[bx],'_'
[MB1]   83:          inc     bx
[M32]   84:          loop    STRIØØ
[635]   85:          mov     BYTE PTR IPuffer[bx],'$'
[2G4]   86: STRI2Ø:  mov     ax,[IBSPos]
[UM4]   87:          TEXT8x5  ah,al,IPuffer,Paper
[BS1]   88:          mov     ax,2Ø
[452]   89:          call    WarteX
[Z64]   90:          pushf               ;Z-Flag retten
[XF1]   91:          push    ax
[L63]   92:          mov     ax,[IBSPos]
[RU4]   93:          TEXT8x5  ah,al,IPuffer,ColorI
[RD1]   94:          pop     ax
[XN4]   95:          popf                ;Z-Flag auswerten
[UX4]   96:          jnz     STRØ2       ;Taste gedrückt
[YW1]   97:          mov     ax,12Ø
[452]   98:          call    WarteX
[9X1]   99:          jz      STRI2Ø
[TW1]  100:          jmp     STRØ2
[7A3]  101: STRØ1:   call    WTaste
[AY2]  102: STRØ2:   jz      STRØ1
[ØØ4]  103:          mov     ax,[LetzteTaste]
[6X2]  104:          or      ah,ah       ;ah=Ø?
[GN2]  105:          jz      STRØ1       ;ja
[6Y2]  106:          or      al,al       ;al=Ø?
[YR1]  107:          jz      STRØ1
[ØT1]  108:          xor     ah,ah
[FE4]  109:          cmp     al,13       ;ENTER-Taste?
[MN2]  110:          jz      STRØ3       ;ja
[UG4]  111:          cmp     al,' '      ;Steuertasten
[V24]  112:          jc      STRØ1       ;ignorieren
[IU1]  113:          cmp     al,'ä'
[S14]  114:          jc      STRØ3       ;Zeichen ok
[465]  115:          and     al,5Fh      ;Bit 7,5 löschen
[2Ø3]  116: STRØ3:   mov     dl,al
[VM3]  117:          mov     bx,[IPosition]
[L63]  118:          mov     ax,[IBSPos]
[ZA4]  119:          add     al,bl       ;al=aktuelle
[XH4]  120:          add     al,bl       ;Ausgabespalte
[MH2]  121:          mov     dh,ColorI
[ØP1]  122:          cmp     dl,13
[FD2]  123:          jz      STREnter
[6Ø6]  124:          cmp     bx,[ILaenge] ;Maximallänge?
[SW2]  125:          jz      STRØ1       ;ja
[P95]  126:          mov     BYTE PTR IPuffer[bx],dl
[MB1]  127:          inc     bx
[DT3]  128:          mov     [IPosition],bx
[8P3]  129:          mov     bx,offset zs8x5
[2R2]  130:          mov     cx,Char8x5
[132]  131:          call    upgraf
[RW1]  132:          jmp     STRØ1
[1N1]  133: STREnter:
[1T5]  134:          cmp     bx,[ILaenge]  ;volle Länge?
[3A3]  135:          jz      STRRaus     ;ja
[T33]  136:          mov     cx,[ILaenge]
[XT1]  137:          sub     cx,bx
[2G6]  138: STREØ1:  mov     BYTE PTR IPuffer[bx],2Øh
[MB1]  139:          inc     bx
[L32]  140:          loop    STREØ1
[3K1]  141: STRRaus:
[L63]  142:          mov     ax,[IBSPos]
[RU4]  143:          TEXT8x5  ah,al,IPuffer,ColorI
[3PØ]  144:          ret
[PB3]  145: StringEingabe    ENDP
[ØØØ]  146:
[NG3]  147: ;Löscht Bildschirm
[ØØØ]  148:
[P12]  149: ClrScr   PROC
[SX1]  150:          mov     cx,174
[1D2]  151:          mov     ax,paper
[LV1]  152:          xor     bx,bx
[ZX1]  153:          mov     dx,319
[HM2]  154: CLRØØ:   push    cx
[XP1]  155:          push    ax
[T23]  156:          mov     cx,HoriLine
[132]  157:          call    upgraf
[RD1]  158:          pop     ax
[2A1]  159:          inc     ah
[VD1]  160:          pop     cx
[LW1]  161:          loop    CLRØØ
[3PØ]  162:          ret
[NZ1]  163: ClrScr   ENDP
[ØØØ]  164:
[D62]  165: OverMeldung:
[3B3]  166:          BOX     7Ø,146,2Ø5,14,ink1
[GR4]  167:          TEXT8x5 15Ø,18,gamevorbei,ink3
[KS1]  168:          mov     cx,5Ø
[7W2]  169: OM1Ø:    mov     ax,1ØØ
[1G1]  170:          push    cx
[452]  171:          call    WarteX
[VD1]  172:          pop     cx
[442]  173:          jnz     OMEnde
[AP1]  174:          loop    OM1Ø
[9Ø2]  175: OMEnde: ret
[ØØØ]  176:
[692]  177: BreakMeldung:
[UB3]  178:          BOX     75,146,164,14,ink1
[WM4]  179:          TEXT8x5  15Ø,2Ø,gamebreak,ink3
[GS1]  180:          mov     cx,3Ø
[DV2]  181: BM1Ø:    mov     ax,1ØØ
[1G1]  182:          push    cx
[452]  183:          call    WarteX
[VD1]  184:          pop     cx
[A32]  185:          jnz     BMEnde
[GN1]  186:          loop    BM1Ø
[FZ1]  187: BMEnde: ret
[ØØØ]  188:
[RP6]  189: ; Fügt aktuellen Spieler gegebenfalls in
[ØB3]  190: ; die Bestenliste ein
[ØØØ]  191:
[8H4]  192: SpielerInBestenListe PROC
[MF1]  193:          push    ds
[GD1]  194:          pop     es
[GM3]  195:          mov     cx,2Ø       ;2Ø Beste
[FN4]  196:          mov     bx,offset BestenListe
[ZL1]  197:          sub     bx,6
[4V2]  198: SIBLØ1:  add     bx,24
[1G1]  199:          push    cx
[ZF1]  200:          push    bx
[4X3]  201:          call    PunkteVergleich
[TD1]  202:          pop     bx
[VD1]  203:          pop     cx
[ØS4]  204:          jz      SIBLØ2      ;ja, in die Liste
[TØ2]  205:          loop    SIBLØ1
[K35]  206:          jmp     SIBLEnde    ;nicht in Liste
```

Listing 13. »Ggdivers.inc« enthält ergänzende Routinen für den Goldgräber.

```
[MP2]  207: SIBL02: push    cx
[9N4]  208:         cmp     cx,1        ;letzter Platz?
[2G2]  209:         je      SIBL10      ;ja
[6M4]  210:         mov     di,offset BestenListe
[VS1]  211:         add     di,478
[6U1]  212:         mov     si,di
[SP1]  213:         sub     si,24
[KN0]  214:         std
[HA1]  215:         dec     cx
[XL1]  216:         shl     cx,1
[XL1]  217:         shl     cx,1
[CV1]  218:         mov     ax,cx
[XL1]  219:         shl     cx,1
[HY4]  220:         add     cx,ax       ;cx=zu verschie-
[TD2]  221:                             ;bende Zeilen*12
[LZ1]  222:         rep     movsw
[Z35]  223: SIBL10: mov     si,offset cpunkte
[4U1]  224:         mov     di,bx
[KN1]  225:         mov     cx,5
[ULØ]  226:         cld
[9Y1]  227:         rep     movsb
[TD4]  228:         mov     ax,[SpielerExtras]
                    ;
[NXØ]  229:         stosb
[EE4]  230:         sub     bx,18       ;bx=Zeiger auf
[PY1]  231:                             ;neuen Namen
[CB5]  232:         mov     WORD PTR [SpielerBest],1
[FT2]  233:                             ;für mp_besten
[RD1]  234:         pop     ax
[FS1]  235:         mov     cx,2Ø
[EV3]  236:         sub     cx,ax       ;cx=Platz
[CV1]  237:         mov     ax,cx
[XL1]  238:         shl     cx,1
[BR1]  239:         add     ax,cx
[BR1]  240:         add     ax,cx
[KØ4]  241:         add     ax,cx       ;ax=Platz*7
[CH4]  242:         add     ax,21       ;ax=Pixelzeile
[WP1]  243:         mov     ah,11
[7S1]  244:         xchg    al,ah
[ZF1]  245:         push    bx
[XF1]  246:         push    ax
[1S2]  247:         call    mp_besten
[BB5]  248:         mov     WORD PTR [SpielerBest],Ø
[924]  249:         TEXT8x5 166,1,NerText,ink3
[2U3]  250:         BOX     Ø,164,32Ø,1Ø,ColorM
[RD1]  251:         pop     ax
[WS1]  252:         mov     bx,18
[7A3]  253:         call    StringEingabe
[NB3]  254:         pop     di          ;Ziel
[MF1]  255:         push    ds
[GD1]  256:         pop     es
[7X3]  257:         mov     si,offset IPuffer
[PN1]  258:         mov     cx,9
[LZ1]  259:         rep     movsw
[KG1]  260: SIBLEnde:
[3PØ]  261:         ret
[ØØØ]  262:
[TF5]  263: ; Ermittelt, ob bestenlistereif
[KK6]  264: ; ZF=1, falls ja (cpunkte>=Bestenpunkte)
[ØØØ]  265:
[TY2]  266: PunkteVergleich:
[ULØ]  267:         cld
[YS3]  268:         mov     cx,5        ;5 Zeichen
[GX3]  269:         mov     si,offset cpunkte
[KV1]  270: PVØØ:   lodsb
[KC5]  271:         cmp     al,[bx]     ;Beste größer?
[F74]  272:         je      PVØ1        ;=, also weiter
[SL4]  273:         jnc     PVØ2        ;CPunkte>Beste
[2P4]  274:         jmp     PVØ3        ;Beste>CPunkte
[VB2]  275: PVØ1:   inc     bx
[3R1]  276:         loop    PVØØ
[SW5]  277: PVØ2:   xor     cx,cx       ;CPunkte>=Beste
[H94]  278: PVØ3:   or      cx,cx       ;ZF -> 1/Ø
[3PØ]  279:         ret
[6G4]  280: SpielerInBestenliste ENDP
```

Listing 13. »Ggdivers.inc« enthält ergänzende Routinen für den Goldgräber.

```
[5K5]    1: 4D 5A 76 Ø1 Ø6 ØØ ØØ ØØ Ø2 ØØ Ø2 Ø6 FF FF FØ FF
[R95]    2: 7Ø 67 ØØ ØØ ØØ Ø1 FØ FF 1E ØØ ØØ ØØ ØØ ØØ ØØ ØØ
[LS5]    3: B8 7Ø 67 A3 4E ØC BF 56 ØC B9 7Ø 5F 2B CF 32 C0
[AL5]    4: F3 AA B4 3Ø CD 21 A3 52 ØC A1 2C ØØ A3 5Ø ØC E8
[AM5]    5: 7C ØØ B9 72 ØØ E8 4Ø ØØ B8 24 ØØ E8 31 ØØ E8 31
[ØL5]    6: ØØ B8 22 52 E8 2B ØØ E8 D1 Ø1 E8 3BØØ A0 48 5F
[9G5]    7: E9 ØE ØØ BB 3E ØB 5Ø 53 92 E8 34 ØØ 58 E8 28 ØØ
[5P5]    8: 58 B4 4C CD 21 C6 Ø6 48 5F Ø1 5Ø B8 34 ØB E8 1F
[SS5]    9: ØØ 58 E8 FA Ø2 8B FØ E8 23 ØØ 8B 1E 54 ØC 8B D6
[VT5]   1Ø: 91 B4 4Ø CD 21 E9 F4 Ø2 E8 E7 FF B8 21 ØB EB E2
[XV5]   11: 50 E8 F7 FF B8 19 ØB E8 D8 FF 58 EB D5 51 57 97
[GW5]   12: B9 FF ØF 33 CØ F2 AE F7 D1 49 91 5F 59 C3 E8 2C
[KM5]   13: Ø3 BF 8Ø ØØ 8A ØD 32 ED 41 D1 E9 33 DB 47 51 57
[1V5]   14: B8 2D 6F F2 AF 5F 59 74 ØE 51 57 B4 4F F2 AF 5F
[8R5]   15: 59 74 Ø4 4B 7A E7 C3 A2 7A 52 C3 E8 91 Ø2 Ø0 3E
[SP5]   16: 52 ØC Ø3 72 3F 8E Ø6 5Ø ØC 33 FF 32 CØ B9 FF FF
[RR5]   17: F2 AE AE 75 FB 47 47 1E Ø6 1F Ø7 8B F7 BF 22 52
[5N5]   18: E8 F8 Ø0 E8 ØØ Ø1 72 Ø3 E9 71 Ø2 FF ØE 4A 5F 7A
[TW5]   19: 13 BA 3F Ø8 Ø8 22 52 B0 Ø2 E9 21 FF AC 8A 4C 5F
[HT5]   2Ø: 7A C3 EB ED B8 2Ø 35 A2 A8 52 CD 21 Ø6 B4 52 CD
[HN5]   21: 21 8C C3 58 3B C3 72 Ø1 93 8E CØ B8 4C ØB 33 D2
[LU5]   22: B9 FF FF E8 C6 Ø1 85 CØ 74 D2 96 FD Ø6 1E Ø7 1F
[AR5]   23: AC 3C 2Ø 73 FB FC 46 AC Ø4 4Ø BF 22 52 57 AA B0
[SL5]   24: 3A AA E8 96 ØØ 5E B9 Ø8 ØØ AC 3C 2Ø 76 Ø4 3C 2E
[2P5]   25: E0 F7 8Ø 7C FF B8 2E 45 AB B8 58 45 AB C6 Ø5 ØØ
[YM5]   26: 13 BA 3F Ø8 Ø8 22 52 B0 Ø2 E9 21 FF AC 8A 4C 5F
[6L5]   27: 48 E8 EE Ø1 E8 6F ØØ 72 Ø3 E9 E0 Ø1 8E Ø6 56 ØC
[NL5]   28: 33 FF B9 Ø5 ØØ BE 50 ØB 32 CØ 51 56 57 F3 A6 5F
[UV5]   29: 5E 59 74 Ø8 F2 AE AE 75 F1 E9 6Ø FF B8 F7 57 BB
[6K5]   3Ø: Ø1 ØØ 26 AC 3C 3B 75 Ø3 32 CØ 43 AA B4 CØ 75 F2
[TT5]   31: 5E C6 Ø6 4C 5F 2F 8E 1E 50 ØC BF 22 52 57 AC AA
[MP5]   32: 84 CØ 75 FA 16 1F 58 BA 4C 5F E8 A5 Ø1 E8 16 ØØ
[WV5]   33: 72 Ø3 E9 7C ØØ 75 DF E9 21 FF AC 8A C0 75
[ØK5]   34: FA 16 1F 16 Ø7 C3 BA 22 52 B8 ØØ 3D B0 3E 52 ØC
[BH5]   35: Ø3 72 Ø2 B0 2Ø CD 21 A3 66 5F C3 E8 51 Ø1 33 C9
[8N5]   36: 33 D2 88 ØE A8 52 8B 1E 66 5F B8 Ø2 42 CD 21 8B
[BF5]   37: F0 85 D2 75 Ø5 3D ØØ 1Ø 72 Ø3 BE ØØ 1Ø 2B C6 83
[4X5]   38: DA ØØ 95 8B FA 83 EE 12 8B D5 8B CF E8 F3 ØØ BA
[RL5]   39: 62 ØC 8D 4C 12 E8 FØ ØØ 8B CE C7 Ø6 57 ØB Ø5 Ø6
[WL5]   4Ø: B8 55 ØB E8 A6 Ø2 E8 Ø3 53 B4 3F 8B C5 Ø8 C7 74 11
[FZ5]   41: 81 ED EA ØF 83 DF ØØ 7D CF Ø3 F5 33 ED 33 FF EB
[Ø7S]   42: C7 B0 Ø3 BA 59 ØB E9 CA FD 8B 3E 44 57 8B 4D 14
[8N5]   43: E3 31 56 8D 75 16 33 DB AC 3C 1B 74 Ø2 3C 13 75
[9T5]   44: Ø3 43 EB Ø5 92 B4 Ø2 CD 21 E2 ED 5E E8 DC FD 85
[ZR5]   45: DB 74 1Ø 8B 75 Ø8 E8 9A ØØ 72 Ø5 B0 Ø8 E9 A1 FD
[XN5]   46: E8 C8 FD 8B 36 4E ØC 8B 55 1Ø 8B 4D 12 E8 72 ØØ
[UG5]   47: 81 E3 FØ ØF 9E 7Ø 72 E8 ØC A1 Ø6 ØØ 2B C3 8B C1
[6R5]   48: 72 22 8B D6 E8 61 ØØ 8B 4D ØA E3 15 8B 5C 1C 8B
[DS5]   49: 54 1E 8D 78 2E Ø3 FA Ø3 7C 2Ø E8 1B Ø1 8B F7 E2
[UP5]   5Ø: EB E9 78 ØØ B0 Ø7 BA A4 ØB E9 47 FD E8 6Ø ØØ 96
[7T5]   51: 33 CØ A3 44 57 E3 25 8B FA AD 47 4F AF E0 FC 83
[4S5]   52: F9 Ø1 76 18 A7 74 Ø6 4F 4F 4E 4E EB E8 5Ø 8D 45
[ZM5]   53: FC A3 44 57 58 8Ø 3E A8 52 ØØ 74 EB A1 44 57 E9
[NW5]   54: 3A ØØ 53 B8 ØØ 42 E8 Ø3 53 B4 3F 8B 1E 66 5F CD
[FS5]   55: 21 5B C3 E8 1C FD B8 ØØ ØC CD 21 24 DF 3C 59 74
[5T5]   56: Ø4 3C 4E 75 F1 92 B4 Ø2 CD 21 8Ø EA 4F F5 C3 8F
[ZG5]   57: Ø6 4Ø 57 55 56 57 53 51 FF 26 4Ø 57 59 5B 5F 5E
[TR5]   58: 5D C3 5Ø 56 57 97 8B F2 AC AA B4 CØ 75 FA 5F 5E
[GS5]   59: 58 C3 52 56 8B F0 E4 FD Ø3 C6 5E 5A EB E3 Ø0
[D16]   60: E8 CC FF B8 F2 8B E9 8B ØE 6A 5F 8B 16 6C 5F BF
[GT5]   61: 3E 53 FC 33 CØ 45 EB 17 26 AC 8B D8 32 D9 8A CD
[FP5]   62: 8A EA 8A D8 Ø9 8A F7 D1 E3 D1 E3 33 Ø9 33 51 Ø2 4D
[SZ5]   63: 75 E6 89 ØE 6A 5F 89 16 6C 5F E9 9F FF E8 8F FF
[8T5]   64: FD 1E Ø7 BF 3C 57 BD FF ØØ B9 Ø8 ØØ 8B D5 33 CØ
[XU5]   65: D1 E8 D1 DA 73 Ø7 81 F2 2Ø 83 35 B8 ED E2 F1 AB
[HX5]   66: 92 AB 4D 79 E4 FC E9 73 FF B8 B3 ØB E8 56 FC B8
```

```
[V@6]   67: FE 52 E8 5D FC E9 64 FF E8 54 FF E8 DC ØØ 8A 44
[6L5]   68: ØA 3C Ø6 74 Ø4 84 CØ 75 E0 Ø3 D3 83 C2 1E 33 C9
[ØL5]   69: Ø3 54 27 A3 47 2C B8 AE 52 B8 Ø9 FF E8 54 ØØ B5 CØ 74 4D
[8Y5]   7Ø: E8 48 FC B8 FF FF A3 6A 5F A3 6C 5F B8 44 ØA 48
[HS5]   71: 78 Ø5 E8 F8 ØØ EB Ø3 E8 B3 ØØ A1 6A 5F 8B 16 6C
[ZM5]   72: 5F F7 Ø6 F7 D2 2B 44 1Ø 1B 54 12 ØB C2 74 ØB B8
[GS5]   73: ØØ ØC E8 FØ FB C6 Ø6 48 5F Ø1 8B 1E 3E 57 8B 4C
[9W5]   74: ØC 8B 54 ØE B8 Ø1 57 CD 21 B4 3E CD 21 E9 EC FE
[C56]   75: E8 DC FE 47 FF FE 52 8B CB Ø3 FB 4F FD B0 2F F2 AE
[PT5]   76: 75 Ø1 47 2A 47 FC B8 AE 52 8B D7 E8 D5 FE B0 74 A7
[KL5]   77: 52 ØØ 75 2A 5Ø BA 7C 52 B4 1A CD 21 5A B4 4E B9
[ZE5]   78: Ø7 ØØ CD 21 72 18 B8 ØØ 43 CD 21 72 11 92 E8 94
[8X5]   79: FB B8 D4 ØB E8 7C FE 72 Ø5 33 CØ E9 9E FE B9 2Ø
[9S5]   8Ø: ØØ B4 3C BA AE 52 CD 21 73 ØA B8 DB DA BA F1 ØB B0
[CW5]   81: Ø5 E9 62 FB A3 3E 57 E9 82 FE E8 72 FE BF FE 52
[7S5]   82: 8D 74 2E BB CB F3 A4 32 CØ AA E9 6F FE E8 5F FE
[1X5]   83: B8 15 ØC E8 5C FB B8 AE 52 E8 56 FB 8B 7C 14 8B
[XR5]   84: 74 16 B9 CØ 45 85 F6 75 ØA 85 FF 74 1D 3B CF 72
[Ø55]   85: Ø2 8B CF BA 62 ØC 52 E8 ØE FE 5A 85 CØ 74 ØB 2B
[CW5]   86: F8 83 DE ØØ 91 E8 1D Ø4 EB DB E9 2F FE E8 1F FE
[2U5]   87: B8 22 ØC E8 1C FB B8 AE 52 E8 16 FB 8B 44 14 A3
[TF5]   88: 6Ø 5F 8B 44 16 A3 62 5F 33 CØ F7 44 Ø8 Ø4 ØØ 74
[6H5]   89: 14 B9 ØØ Ø1 BB 22 4F E8 C9 ØØ BA 22 5Ø E8 D0 Ø1
[1D5]   9Ø: E8 7B Ø1 B0 Ø1 A2 72 52 FF 36 Ø8 Ø2 ØØ 77 25 C7
[P95]   91: Ø6 74 52 ØØ 2Ø C7 Ø6 78 52 ØØ 1Ø C6 Ø6 68 5F Ø2
[NR5]   92: C6 Ø6 5A 5F 3F C6 Ø6 5E 5F Ø6 C7 Ø6 64 5F 62 1C
[BH5]   93: EB 23 E8 6C C7 Ø6 74 52 ØØ 1Ø C7 Ø6 78 52 ØØ 2Ø
[BR5]   94: C6 Ø6 68 5F Ø3 C6 Ø6 5A 5F 7F C6 Ø6 5E 5F Ø7 C7
[VH5]   95: Ø6 64 5F 62 2C A0 68 5F Ø4 3F A2 7B 52 B9 4Ø ØØ
[AT5]   96: BB ØØ Ø1 Ø0 BB A2 4D E8 C9 ØØ BB 22 5Ø E8 DØ Ø1
[PU5]   97: A2 4E E8 5B Ø1 BB A2 4D BA E2 4D E8 52 E8 77
[8R5]   98: ØØ 32 CØ BF 62 ØC 8B ØE 78 52 81 C1 4Ø Ø1 F3 AA
[NS5]   99: E8 E4 Ø1 91 8B 16 64 5F 2B CA E8 48 Ø3 E9 5C FD
[VL5]  1ØØ: BA 2F ØC B0 Ø4 E9 2B FA 51 B9 Ø1 ØØ 29 ØE 60 5F
[1S5]  1Ø1: 83 1E 62 5F ØØ BA 6Ø ØC E8 ØD FD 48 59 75 E1 A1
[GZ5]  1Ø2: 6Ø ØC C3 E8 29 FD E8 DF FF 3B C1 73 D3 91 5Ø 41
[G2S]  1Ø3: BF 62 ØC 8B FF D0 FF A0 E2 FA 5A 8B FB 33 DB
[NS5]  1Ø4: AC 8A D8 B1 Ø4 D2 E8 43 8A D3 24 4Ø D3 72
[746]  1Ø5: AF F3 AA 77 EB E9 Ø4 FD E8 F4 FC BB 3F ØØ BA FF
[LL5]  1Ø6: ØØ 8A 8F A2 4D 8Ø F9 1Ø 73 1D 32 E4 8B F3 8A 8Ø
[GR5]  1Ø7: E2 4D 8B F0 BF Ø1 ØØ D3 E7 88 8C A2 4C 88 9C A2
[DV5]  1Ø8: 4B Ø3 F7 3B F2 76 F2 8A C3 Ø2 Ø6 68 5F 8B F3 8B
[XP5]  1Ø9: B0 A2 4E 8A 8F 62 4E 8Ø F9 Ø8 77 Ø5 BD A2 3D EB
[QV5]  11Ø: 2Ø 85 FF 74 Ø7 8B FE 23 FA 88 95 A2 3D BD A2 3F
[9S5]  111: B1 Ø4 EB Ø5 BD A2 41 B1 Ø8 D3 D2 E2 8F F2 4E 76
[FP5]  112: D9 E8 Ø8 Ø0 4B 7D 9A E9 92 FC 8A C3 BF Ø1 ØØ D3
[3P5]  113: E7 88 Ø2 88 8A Ø0 Ø1 Ø3 F7 3B F2 76 F4 C3 E8 6E
[TT5]  114: FC BA FF Ø0 8B DA 8B F3 8B B0 22 50 8A 8F 22 4F
[YP5]  115: 8Ø F9 Ø8 77 Ø5 BD A2 43 EB 2D 85 F2 74 15 8B FE
[YP5]  116: 23 FA 88 95 A2 43 F7 C6 3F 00 74 0E BD A2 45 B1
[5R5]  117: Ø4 EB ØC BD A2 49 B1 Ø8 EB Ø5 BD A2 47 B1 Ø6 D3
[FY5]  118: EE 2A 8F 22 4F F6 D9 E8 A8 FF 4B 7D B9 E9 2C FC
[7N5]  119: 49 8F E2 ØC 89 55 14 E8 5F 12 B9 Ø8 1Ø 57
[ZP5]  12Ø: B9 Ø8ØØ 33 CØ F3 AB 5F 4F 8B 75 13 8B 4D 11 AC
[FL5]  121: 8B D8 FE Ø1 E2 F9 33 CØ B9 1Ø ØØ 47 Ø2 Ø5 AA E2
[Ø55]  122: FB BF 61 ØC 8B 75 13 8B 4D 11 Ø3 F1 4E FD 33 CØ
[KV5]  123: 33 DB AC 8B E8 8A 1B FE CB 88 1B 88 41 17 E2 F2
[VL5]  124: 8B 5D 11 8D 71 16 8D B9 15 Ø1 Ø3 FB 33 D2 33 CØ
[GR5]  125: 3C Ø9 Ø3 C2 8A ØC 4E 3A E9 74 Ø7 8A E9 33 D2 F9
[8T5]  126: D3 DA AB 4B 75 EC BD 61 ØC 8B 7E 15 8B 76 13 8B
[AU5]  127: 4E 11 BB 78 ØD FC 57 AC 32 E4 8B F8 8A Ø3 FE Ø3
[TP5]  128: 97 D1 E7 8B 11 51 B9 1Ø ØØ D1 E2 D1 D8 E2 FA 59
[DV5]  129: 5F AB E2 E2 E9 85 FB E8 75 FB 33 CØ A2 76 52 E8
[BR5]  13Ø: 42 Ø1 8B 3E 64 5F 4B 8A 14 46 32 ED EB 5A 8A 1Ø
[NP5]  131: B8 A2 47 E8 BE ØØ Ø3 D8 8A Ø7 8A 8F ØØ Ø1 EB 23
[VE5]  132: B1 Ø2 BB A2 4F E8 EC FE 6Ø 3F 74 E2 B1 Ø4 BB A2
[TV5]  133: 45 EB Ø2 84 D2 74 E9 8A 87 A2 43 3C F7 74 E8 8A
[8R5]  134: 8F A2 44 E8 8E ØØ AA EB 7A B0 3E 72 52 ØØ 75 E3
[9R5]  135: 8A C2 B1 Ø8 EB ED B1 Ø8 B8 A2 41 E8 76 ØØ Ø3 D8
[2S5]  136: 8A Ø7 8A 8F ØØ Ø1 EB 3A B1 Ø1 E8 67 ØØ 72 DA 8A
[CR5]  137: E2 22 26 5A 5F A0 5E 5F 8A C8 E8 57 ØØ 8A 8F A2
[ØU5]  138: 4C 8A 9F A2 4B 91 D3 E3 91 ØA DC 89 1E 42 57 E8
[9M5]  139: 42 Ø2 84 D2 74 CØ 8A 87 A2 3C 3C 74 78 8A 8F
[CN5]  14Ø: A2 3E E8 2F ØØ 32 E4 38 Ø6 7B 52 75 Ø9 8B DA 32
[2P5]  141: FF Ø3 C3 E8 1C ØØ 91 56 8B F7 2B 36 42 57 4E F3
[ZN5]  142: A4 91 5E 81 FF 62 3C 72 9F EB 29 B1 Ø4 B8 A2 3F
[ST5]  143: EB 89 B1 Ø8 2A E9 73 15 F6 DD 2A CD D3 EA 4D 79
[HN5]  144: Ø3 E8 41 Ø0 8A 34 46 8A CD B5 Ø8 2A E9 32 FF D3
[5R5]  145: EA 8A DA C3 51 52 56 57 8B ØE 74 52 8B 16 64 5F
[VK5]  146: 8B 4E ØE 2 ØC 8B 36 64 5F Ø3 36 74 52 8B ØE
[YH5]  147: 78 52 2B F1 81 C1 4Ø Ø1 F3 A4 5F 2B 3E 74 52 5E
[GH5]  148: 5A 59 E9 43 FF 80 3E 76 52 Ø1 74 12 50 51 52 E8
[BD5]  149: 12 Ø0 5A 59 58 4D 79 Ø5 C6 Ø6 76 52 Ø1 C3 58 58
[TL5]  15Ø: 97 E9 3A FA B9 Ø0 8B 6Ø 5F 29 ØC 83 5C Ø2 ØØ
[NG5]  151: 73 Ø9 Ø3 ØC 33 CØ 89 Ø4 89 44 Ø2 BA 48 57 8B F2
[4T5]  152: E8 E5 F9 95 C3 53 52 E8 36 FA 5A 8B 1E 3E 57 B4
[SP5]  153: 54 CD 21 5B 3B C1 75 Ø1 C3 B4 4C CD 21 90 48
[SP5]  154: B4 41 CD 21 BA 44 ØC B0 Ø6 E9 C7 F6 Ø0 ØA 50 4B
[A65]  155: 53 46 58 2Ø 28 52 29 2Ø 2Ø 46 41 53 54 21 21 2Ø
[5F5]  156: 2Ø 4D 69 6E 69 2Ø 53 65 6C 66 2Ø 45 78 74 72 61
[EB5]  157: 63 74 2Ø 55 74 69 6C 69 74 79 2Ø 2Ø 56 65 72 73
[ZB5]  158: 69 6F 6E 2Ø 31 2E 31 2Ø 2Ø 3Ø 33 2D 31 35 2D 39
[GE5]  159: 3Ø 0D 0A 43 6F 70 72 2E 2Ø 31 39 38 39 2D 31 39
[SC5]  16Ø: 39Ø 3Ø 4Ø 54 41 54 52 45 2Ø 4F 52 4D 45 49 6E 54
[RD5]  161: 6C 6C 2Ø 52 69 67 68 74 73 2Ø 52 65 73 65 72 76
[BC5]  162: 65 64 2E 0D 0A 50 4B 53 46 58 2Ø 52 65 67 2E 2Ø
[NH5]  163: 55 2E 53 2E 2Ø 5Ø 61 74 2E 2Ø 61 6E 64 2Ø 54 6D
[BD5]  164: 2E 2Ø 4F 66 66 2E 0D 0A Ø0 5Ø 4B 53 46 58 3A 2Ø
[BC5]  165: Ø0 0D 0A Ø0 53 65 61 72 63 68 69 6E 67 2Ø 45 58
[BB5]  166: 45 3A 2Ø Ø0 57 61 72 6E 69 6E 67 21 2Ø Ø0 Ø0 43
[2C5]  167: 61 6E 2Ø 27 74 2Ø 4F 70 65 6E 3A 2Ø Ø0 45 58 4D
[XB5]  168: 50 41 54 4B 3D 50 4B Ø0 Ø0 45 72 72 6F 72 2Ø 69
[6C5]  169: 6E 2Ø 5A 49 50 2Ø 2D 2Ø 55 73 65 2Ø 50 4B 5A 69
[LC5]  17Ø: 70 46 69 78 Ø0 44 6F 2Ø 79 6F 75 2Ø 77 61 6E 74
[5A5]  171: 2Ø 74 6F 2Ø 65 78 74 72 61 63 74 2Ø 74 68 65 73
[AH5]  172: 65 Ø0 66 69 6C 65 73 2Ø 6E 6F 77 2Ø 28 79 2F 6E
[XF5]  173: 29 3F 2Ø Ø0 54 6F 6F 2Ø 6D 61 6E 79 2Ø 66 69 6C
[5L5]  174: 65 73 Ø0 55 6E 6E 6F 77 6E 2Ø 63 6F 6D 2Ø 46 46
[VM5]  175: 65 72 53 69 6F 6E 2Ø 6D 65 74 6F 64 2Ø 66 6F
[VA5]  176: 72 3A 2Ø Ø0 2Ø 61 6C 72 65 61 64 79 2Ø 65 78 69
[UB5]  177: 73 74 73 21 2Ø 4F 76 65 72 77 72 69 74 65 3F 2Ø
[1B5]  178: Ø0 43 61 6E 27 74 2Ø 63 72 65 61 74 65 3A 2Ø Ø0
[YB5]  179: 66 69 6C 65 2Ø 66 61 69 6C 73 2Ø 43 52 43 2Ø 63
[EF5]  18Ø: 68 65 63 6B Ø0 45 78 74 72 61 63 74 69 6E 67 3A
[SD5]  181: 2Ø 62 2Ø 27 74 2Ø 47 70 65 6E 3A 2Ø Ø0 45 58 4D
[B95]  182: 69 6C 65 2Ø 68 61 73 2Ø 61 2Ø 62 61 64 2Ø 74 61
[YD5]  183: 62 6C 65 Ø0 64 69 73 6B 2Ø 66 75 6C 6C Ø0 Ø0 Ø0
[L35]  184: Ø0 Ø0 Ø0 Ø0 Ø1 Ø0 5Ø 4B Ø3 Ø4 Ø0 Ø0 Ø0 Ø0 Ø6 Ø0
[AF5]  185: 6A 7F DD 18 13 A0 9C 54 12 Ø7 Ø0 Ø0 BC ØF Ø0 Ø0
[NK5]  186: Ø8 Ø0 Ø0 Ø0 47 47 4C 4F 47 4F 53 2E 4F 42 4A ØF
[1F5]  187: Ø0 12 Ø3 24 15 36 27 38 39 6A 7B 4C 9D 6E 1F 09
[BH5]  188: Ø6 Ø1 13 34 E5 F6 96 F7 Ø1 37 AB 8B 6E 5C 59 B6
[8V5]  189: 6F CF BE 9D EB 32 E8 D4 A6 67 11 83 A0 76 AA 75
[DR5]  19Ø: E5 8A 7D Ø8 32 E8 DC B9 65 DB 8A 65 5B 56 2E 48
[DR5]  191: 9Ø 56 88 9F 9B F6 AD 5B 9Ø 32 5D C2 8C 79 18 F1
[HS5]  192: 44 8Ø 4D D3 C3 CE DB 1D Ø3 3E AC 44 C4 1Ø AA 9D
[WK5]  193: 79 C4 9Ø Ø0 11 96 C5 Ø8 97 AD 4B B8 69 C7 5A 44
[VP5]  194: 2C 41 DA 29 22 9Ø 5E 61 DA 36 D2 BF 25 43 AD 07
[7R5]  195: F1 3F AØ 76 B4 1F 33 5B 3C 9A 55 98 72 7Ø 80
[6V5]  196: 9F 19 91 B1 C5 85 65 E1 98 1B D4 55 EB BA 6A FE
[4R5]  197: 3C 35 B6 B2 27 13 76 D5 38 5A C4 Ø3 AA 9D A9 65
[FT5]  198: Ø1 63 A5 CC 43 53 B6 CC 1Ø 6Ø C1 AF 54 8B 62 A5
[Ø45]  199: 4A 7Ø E8 53 A2 45 2D 63 3E Ø8 1Ø C9 F6 Ø2 Ø7 Ø6
[CG5]  2ØØ: 9C 8Ø CD 89 Ø6 A5 1A 94 ØØ B1 Ø4 C3 56 AØ ØE Ø9
[MR5]  2Ø1: 16 ØC 18 DC 32 42 8Ø Ø6 89 1E 95 FA B4 2A D4 C5
[2M2]  2Ø2: 9A 69 2B FF 5F ED 56 C9 ØE Ø1 Ø6 14 78 4Ø BA A2
[SH5]  2Ø3: 87 CA 54 88 9F 9D 28 64 Ø5 D0 3E 3Ø E9 5Ø A4 8Ø
[VK5]  2Ø4: CØ 47 Ø8 B9 Ø1 B3 1B 4C DA 64 B0 Ø7 82 F7 53
[8U5]  2Ø5: DA 69 A2 4F A7 3C 7F F4 29 AØ F4 6F 41 D3 45 A1
[LS5]  2Ø6: 8Ø 1B F4 CØ ØD Ø2 E4 8B A8 7E AØ A2 Ø3 46 Ø6 8D
[CH5]  2Ø7: 7E AØ ØØ 34 22 1F 1Ø EB 8Ø E2 79 1Ø 3B B0 E4 81
[VU5]  2Ø8: 3F 1C F6 D7 F3 81 ØF 38 1E 38 3C 6C FF Ø9 AF 39
[ØN5]  2Ø9: F1 3F AØ 76 B4 1F 33 5B 3C AD C8 3C 9A 55 98 72 7Ø 80
[MS5]  21Ø: 3B 9A 7E ØD 1Ø C5 D1 8E 1E D4 1F 7E 38 7A 2Ø C2
[RR5]  211: F9 8Ø Ø7 82 7E 88 EC 68 57 B8 53 9E DC 83 38 7A
[CG5]  212: D1 Ø2 Ø1 2D 77 B4 Ø3 16 F0 D5 2B 94 6B E6 68 9Ø
[6N5]  213: AA 3C 4Ø 8A ØØ 83 9E ØE 88 92 6Ø C0 82 68 8E F6
[TE5]  214: 82 A1 Ø3 22 9C 4A 3B 2Ø E8 82 48 A1 Ø6 D8 3F A3
[8D5]  215: 54 81 Ø1 21 36 2C 5Ø E4 Ø9 15 F1 8Ø Ø1 83 Ø2 ØD
[ØE5]  216: 7Ø F4 29 Ø1 43 ØC 8Ø B4 2Ø 4Ø Ø2 F8 Ø6 3E Ø4 A4
[9P5]  217: 84 6A 65 7Ø BD 1Ø C5 D1 8E 1E D4 1F 7E 38 7A 2Ø C2
[VV5]  218: D1 Ø3 3B BA A7 9E 4B B8 21 E8 63 4F DD 13 AØ FE
[7R5]  219: 8F D2 85 37 84 D8 FF A6 FC E6 8C Ø1 1E ØC 3E 7Ø
```

```
[HP5]  22Ø: 28 DØ 87 CE D4 3C 6Ø 1E 47 5F FC 21 43 78 47 A3
[EK5]  221: B4 3Ø 5Ø 14 FD 23 64 C8 57 C4 D1 27 38 9A 46 18
[LL5]  222: BD C4 D1 13 38 9A 86 76 F4 7A 3F 66 17 DA 1Ø 63
[GV5]  223: D7 9B FA 9B 33 69 3F 7C EA 25 CC 3B 47 78 53 D4
[5W5]  224: D1 34 BE A3 19 E5 FF FF C1 D1 3F 5D C9 B0 27 7F
[8L5]  225: 86 7F 47 26 55 9D 9A 25 CF 87 Ø6 71 8B 46 74 F4
[YW5]  226: EA 46 FF D9 87 1B AA 2B D2 A8 7F Ø2 54 A3 BE EA
[HT5]  227: 87 18 B5 A2 56 D9 9A 59 F7 E0 53 AE 75 F6 D4 F8
[KY5]  228: 7F 8A E6 77 F4 FD 9F 1D AD ØD 8E D6 77 47 AB 4A
[YN5]  229: 8Ø 3D D1 33 7C F3 D4 AØ 43 A3 8Ø 2A C9 5F 2Ø E2
[LR5]  23Ø: C8 24 8E 1E AØ 88 A3 Ø7 57 3C C2 9D Ø0 C1 A8 D1
[3M5]  231: 57 1C DD D4 B7 44 8D D0 2Ø E8 99 2Ø 5D 2B 74 34
[FK5]  232: F5 82 79 1D 7D EA 47 47 43 94 61 18 57 7E 88 62
[TG5]  233: 36 DD 65 CE 4Ø 4F 8E 3Ø 3Ø AØ 61 75 B2 A3 23 76
[GE5]  234: 84 33 9Ø C0 57 42 04 72 F4 2Ø 11 23 24 Ø8 E2 E8
[LH5]  235: Ø5 B1 87 1F 7A 47 13 18 47 33 E2 Ø9 52 AD 12 66
[8N5]  236: AF 5A C1 D8 94 ØF 4A 91 22 D4 8A 1Ø AB 46 8C 7Ø
[7T5]  237: 5C 6B 97 D9 47 8D C0 DC B0 39 5F 94 6A D4 8A Ø5
[SH5]  238: 88 AF 48 94 48 F1 45 8B 56 A5 18 B0 22 C1 A8 54
[4L5]  239: A1 Ø2 A8 3F ØA BF Ø3 Ø8 5A 9Ø 76 2Ø C4 76 B4 53
[AM5]  24Ø: 39 9A 5A FC 21 5A ØD 1B 11 Ø2 D1 A7 45 48 44 F6
[YT5]  241: AF 28 52 D2 D4 Ø7 8C A8 14 6D 94 E2 ØE AB A3 8B
[U16]  242: FD 8E 16 BF 3B 5A EE ED E8 6E Ø0 7C C1 8A 50 AC
[NV5]  243: 9D AA 9Ø A3 DB 6A 88 5A 8E E6 AA 11 1D CD 45 2D
[TP5]  244: 77 B4 A3 8E 52 7C 51 B4 2A 8E 26 Ø1 1C CD EE 15
[VS5]  245: ØA A0 8Ø AD 4Ø AC 14 8E 2E CD ØE 51 6D E8 Ø0 BD
[NR5]  246: E9 AA 55 13 AA DF A2 68 D1 A1 82 5F A2 62 D4 91
[KR5]  247: A2 2F 8A 2A DA E2 DE B0 A2 A0 BD C0 D1 03 41 44
[1H5]  248: 60 54 93 16 8C 82 D8 41 04 5D 21 BB A2 E2 11 55
[HR5]  249: 47 BC 6A 51 CB 15 D3 ØA A2 45 3A EC 41 A9 AB Ø8
[WU5]  25Ø: 4B Ø2 B4 A6 69 D5 A8 Ø1 D4 73 4Ø CA FE 5A 5D BD
[CZ5]  251: AB 7C B5 AF Ø2 A2 19 D8 2B 6A C7 Ø9 A9 BA 92 EB
[EU5]  252: 4C 98 6Ø 60 55 DC AD ØA 9Ø 4E 59 FF E8 13 DA 15
[EU5]  253: FE 56 ØF C8 6C 1C C3 Ø5 Ø3 Ø2 E9 F9 F9 F9 F9 F9
[HT5]  254: 45 BA C2 FF B1 AC 3C 96 15 C0 A5 4A 3Ø 90 AB D4
[PY5]  255: 89 57 FD FA AF D8 4B 1D F1 82 5Ø AE EC A5 AB 54
[US5]  256: D7 65 AF Ø8 14 ØA A8 AD B2 3Ø 53 ØA CB FA B4 4Ø
[7K5]  257: F1 AØ AE AF A2 D6 Ø5 88 B7 72 27 Ø7 Ø5 24 Ø9 76
[6V5]  258: D4 FF FF Ø5 3B B0 A3 81 AC E3 68 2E E3 68 AE E3
[KB5]  259: 68 Ø0 48 65 47 73 61 47 73 65 47 03 60 F5 5D 19
[AT5]  26Ø: A8 FA 8E Ø6 60 EA 3B 1A D8 47 7Ø 4A 53 75 7C Ø4
[1H5]  261: Ø0 52 D8 D1 EC 4B Ø0 68 65 47 03 55 D9 19 Ø1 C0
[VL5]  262: 8E 66 67 
```

```
[HN5]  372: 49 14 85 33 E9 6A 55 CF 34 55 A9 45 A1 42 2D CA
[XU5]  373: EB 7B 53 51 D3 53 9F 2A AD 3A 15 BA 4C 80 C6 9C
[5L5]  374: 09 20 8D 94 29 C8 8C 00 4F F9 45 06 85 9A A4 E9
[YK5]  375: F9 C3 E1 04 D8 16 72 82 67 17 DB 74 28 CA 97 00 F7
[1X5]  376: AC 4D A5 2F D5 E9 82 EF A1 B2 9D 09 D0 CA 91 58
[WN5]  377: 26 5B 5D 14 88 9E 97 B6 4F 1B 49 FB A9 53 6D 02
[8M5]  378: AD 09 90 C5 33 3C 4D D2 34 01 2E 53 9B 58 F3 55
[NR5]  379: 91 F2 62 DC 25 38 D3 4B CE 58 FF 34 90 FA 4E 11
[AR5]  380: E9 2A 09 6F 28 73 BC 0A 33 0E 93 7B 47 80 B1 98
[6L5]  381: 71 99 9C 0D 34 21 66 BC 26 E9 8F 00 C5 31 E3 34
[1U5]  382: 91 E1 E1 0A 6B 1C B8 4D 68 15 6B 5A CC F8 4C 04 E6
[MH5]  383: 03 62 B6 58 0E E8 03 02 3C C6 8C 0B C8 24 C2 3E 19
[XL5]  384: 66 BC 26 56 8D 00 69 30 E3 37 69 E1 DE ED 98 71
[AR5]  385: 9B 18 98 57 C4 8C E7 04 E2 D4 45 98 71 9C F4 E6
[PR5]  386: AF C4 8C D3 E4 13 BB A7 80 33 B5 C4 A2 EC 53 31
[ES5]  387: C7 04 58 CB F1 71 A9 42 A4 87 6A 94 1E 66 AF 8F
[TP5]  388: 8F B8 93 C4 19 2F 49 80 47 FE 44 1F 7D CA 94 28
[8S5]  389: 21 99 5A 34 E4 08 CD AE E0 F6 AE A2 C7 86 76 A3
[EV5]  390: 46 89 CD F1 2E B0 4F 7D FE 4C 58 19 47 38 5E 1B
[XP5]  391: 93 7A 5C 22 F0 94 4D A2 AF C6 99 EA FC A4 78 51
[0R5]  392: 04 62 1C D3 14 89 EE 8B 52 C5 92 BF 1D A0 28 F6
[AS5]  393: 9D AC 7E 11 CB 01 2C FE 35 12 9B 08 5D A3 9E D7
[3W5]  394: F2 D9 FE A2 DA 5E 2A 4E 05 7F E1 41 B8 E6 58 DA
[SM5]  395: 1A 89 3F 4D 24 77 5E E2 74 9D 3B C2 15 AD 67 E1
[TN5]  396: FC 59 71 A2 73 AE 10 43 9C 81 00 18 10 3A 8F FA
[GR5]  397: 72 51 32 B7 BB 70 7A 68 10 A0 7A AC 41 90 FE
[1R5]  398: 7E C0 E9 29 8E 5B 9C 9D 29 C5 92 00 21 C8 59 AA
[HV5]  399: 73 43 0F AB F9 33 DE 84 E4 7D E5 7D AA 40 5E 7B
[NN5]  400: 66 40 A6 8D 16 69 5F 2A E4 8A 00 A3 B4 D3 4F F4
[LR5]  401: 8F C9 89 AF 51 71 63 9B 79 8C E4 01 D6 73 46 D2
[FW5]  402: E4 9D A1 2F 5B 9A 5F 99 8C 89 28 95 6E B6 FD 2E
[YX5]  403: 68 CA 77 EB BA B6 1C 8C 9F 0F 06 4D 25 B9 EF 8D
[4L5]  404: F4 79 58 89 E0 45 41 D3 55 9A B3 85 33 35 D1 27
[TN5]  405: CC 59 71 A2 73 AE 10 43 9C 81 00 18 10 3A 8F FA
[4G5]  406: 81 02 7E 22 5D 6D D8 09 06 40 96 38 5E 10 2B C0
[LT5]  407: 8A 54 85 4A 8C 32 D6 50 AD 42 AD 0C 50 AB C0 D2
[2K5]  408: 55 9E 5C 99 16 58 21 FA 8A 02 8B 22 51 FC 29 52
[LS5]  409: 0E 4B B5 4A BA 47 25 4B 26 FA 8A 95 41 52 2A BE
[VL5]  410: B4 84 D8 AA 78 A8 15 8B E8 7D 41 20 48 0F 14 00
[9L5]  411: 6D 0C B4 2A E0 80 55 89 86 2C 09 B1 60 C5 82 65
[TN5]  412: 13 42 2D 00 5A 45 A5 56 9C 63 BE B2 28 CA 98 50
[9L5]  413: 49 52 0D 54 7C 52 23 EF 43 49 77 8B 46 81 06 8C
[FP5]  414: 9A 55 6A 82 E2 B5 46 2D 19 06 AD AB 06 A4 A2 76
[5M5]  415: 71 27 D6 14 86 62 6B 35 3C 82 AD 78 D5 2A B8 32
[DL5]  416: 80 4F 50 F0 6D 01 82 79 36 28 2C 56 A5 6A 91 8D
[XU5]  417: 8A 47 AD 58 75 A3 68 A2 9B 4A A7 58 2C 90 BB CD
[VL5]  418: AF 12 1A 37 16 3C 74 03 CF 52 BC 48 D5 A2 51 14
[UR5]  419: AF 12 71 6A 56 D1 58 37 4B A7 8C 06 B2 28 CA 98 9E
[4S5]  420: E8 00 D3 48 1E C1 26 08 FF 09 90 BF 20 90 F3 C3
[FP5]  421: E1 F0 15 03 54 39 F0 3F 70 55 47 FB 3F A4 AD 98
[RT5]  422: 8A A6 BA A9 74 AA 9E 0A C6 07 0C F3 87 C0 8A 7F
[PP5]  423: 09 FC 27 02 58 7F 34 7C FE 08 41 0B 04 FD 1F BE
[DW5]  424: 42 50 56 FE 6F BD 2B 57 FC D7 7E C5 45 FF F0 5F
[4T5]  425: 74 F4 90 7C 88 AC 78 90 FC 0F E2 4F 75 7C
[UN5]  426: 00 F5 36 CC 29 30 4D 22 D8 0A 1E 86 E9 73 E7 AA
[YK5]  427: 55 97 4A 7F C4 B2 32 44 A5 5E F1 32 35 D7 00 C5
[7S5]  428: 01 2A 37 DA A8 D8 96 A9 6B 6A 9C 7A A7 F6 89 60
[TT5]  429: F6 39 36 87 01 CE 54 77 5D F1 38 AE AD 78 E5 8F
[4S5]  430: 8A 7D 60 4A D4 D5 66 FF 81 24 C7 88 79 20 0B E7
[MN5]  431: 51 C5 3E 50 39 A3 6E 85 09 AB 58 1F EA 78 97 AD
[ZL5]  432: 14 AA 07 A4 49 2F 5B 21 F6 8A C9 29 69 04 40 9E
[DY5]  433: E1 B5 29 CD 5F 21 A3 E2 1F 0A 54 FD 28 7A FF AA
[7P5]  434: 01 6D 6D C5 54 34 D5 4D A5 53 F5 54 14 E8 1B DB
[BV5]  435: FF CF 23 D2 80 A8 6F 0E A7 4F 8E AA D8 E1 AB 44
[NM5]  436: 15 FD 43 85 A8 11 58 57 4C 45 53 DD 44 3A AB 27
[SR5]  437: 82 E6 FF 03 50 55 F0 C8 03 3E B8 13 1C 75 AD 4F
[9N5]  438: 89 2B 4C 15 7E 56 E0 8A 07 15 AB 8A 62 1F 08 52
[WL5]  439: 20 20 2A 3F 10 A5 0A 2D 7B 85 D6 B3 3E 1D 34 48
[EM5]  440: B0 01 61 0A C1 76 38 90 0F C6 66 0D F5 F9 D2 87
[BG5]  441: 92 E4 0C 11 A2 FA 93 33 14 50 2C 26 7F 14 59 BB
[7K5]  442: 4C C1 A0 50 C3 6A 57 08 33 A0 65 B4 62 5A 81 1C
[RW5]  443: 0E 8D 02 DF AC 52 06 C4 D2 C4 82 AD D8 76 BF 4E
[4K5]  444: 5E 85 B2 01 A2 5D AE 03 18 8A 84 82 A5 E1 17 28
[KM5]  445: 26 A2 B9 12 55 57 F2 03 C8 09 78 4A DF 84 66 C1
[0M5]  446: 86 6C 4A B8 02 E3 20 D7 05 2C 2B 20 F2 7F F2 07
[AP5]  447: A0 14 31 74 4E 9D 5A 62 29 96 EC D0 F4 21 B3 33
[RH5]  448: 77 50 41 D3 10 C0 B2 48 55 01 62 89 DA C3 E3 13
[PR5]  449: 8B 93 0E 6B 1E 18 12 39 8C 6B FD C7 6F C9 0F 04
[FW5]  450: 50 8C 8E 39 45 D2 EA EE BA E0 68 BD 6D 0D 3F FF
[KR5]  451: F1 89 8A 93 C2 48 2D 59 B6 7A B9 6E C9 3A 40 6A
[5M5]  452: 29 C6 A9 C2 A4 52 96 19 ED F0 0C 5F 82 21 9E 02
[MG5]  453: 10 6B 0E AC 08 E0 29 92 40 30 A8 A7 00 D4 03 2B
[EM5]  454: 7D 30 4F 11 1E BF 60 58 4F 01 A8 01 7A 5B 4C 1F
[ZN5]  455: 4F F1 FE 00 A4 C4 8A A1 9E 02 50 AF 02 A4 43 79
[1E5]  456: 72 B4 43 93 12 55 0C F5 14 80 5A 11 06 48 B8 C2
[FL5]  457: 97 08 29 B1 62 A8 A7 00 F4 82 03 0B C2 69 F9 1F
[8H5]  458: F8 12 25 25 56 0C 39 C0 2A 4E 26 D8 32 91 03 E8
[3V5]  459: 1D 42 79 1E CC F6 D5 3C C4 72 1E C1 BC 72 D5 B4
[0F5]  460: 55 AD 14 13 D1 54 37 95 4E D5 13 41 13 84 C1 79
[ST5]  461: 40 59 C0 FF 21 2E 5E 85 6D C8 3F 68 7B 01 99 BD
[4S5]  462: 1D 42 79 1E CC F6 D5 3C C4 72 1E C1 BC 72 D5 B4
[2K5]  463: 0E 8B 85 54 B7 32 C0 32 33 55 07 80 B4 F0 9C 8A
[ZU5]  464: 89 68 AA 9B 48 A7 EA A9 A0 E9 07 D0 FC 87 59 9C
[PS5]  465: 9F 5C 1C 1E B6 02 93 93 0A 38 AA A0 AD 98 8A A6
[FU5]  466: BA 89 74 AA 9E 0A 9A 37 50 BB E4 01 63 4F A6 EA
[SN5]  467: 52 E3 0D 8E AC 9A A3 C0 82 C4 6A 15 5F 51 40 E5
[TN5]  468: 44 09 C8 3C EF 5F 50 2D 8A 62 2C 9A EA 26 D2 A9
[9P5]  469: 7A 2A 68 73 80 77 B0 58 5C 9A 8B D8 7F 7D 4E 1C
[6M5]  470: 64 E8 D0 1D 28 0A 40 E8 D0 AD 8A 06 54 4C 44 53
[UK5]  471: DD 44 3A 55 4F 04 AD 64 0A 89 1C 91 89 30 C1 00
[MM5]  472: FD 81 B0 56 00 24 2E 55 D5 C2 95 65 EA 9A 3A 1A
[DL5]  473: 40 CA 05 C2 2B 0C 91 F3 65 41 64 1A CA 35 0D C0
[UG5]  474: 56 58 58 58 4A 07 5F 25 35 19 78 0D 75 0A 29 C3
[TN5]  475: 81 58 0C 77 29 74 82 E3 14 B3 B8 46 28 46 74 B3
[HR5]  476: 74 5E C5 A6 83 2D B0 36 03 2D 89 C4 3B 72 0F 6D
[CG5]  477: 28 C5 21 E4 23 12 29 BE 42 31 0F 48 A4 E9 8A 51
[HP5]  478: DD 2C 9D 17 08 8B B8 4D 0E C0 A1 80 56 91 78 46
[9W5]  479: 08 56 C3 5B 9E 2A 20 0F 1C F8 F3 44 CB 33 7F FE
[AT5]  480: 6C D9 99 2D C3 3E 5B 96 EE 87 2D 5B BA 27 A2 02
[6P5]  481: 47 35 B3 A1 41 2D EE 0A 01 1B 7C C5 9D 80 D8 20
[RS5]  482: 48 E5 F1 15 C7 49 9A 39 BF 65 56 A6 86 0C CB 6F 59
[GS5]  483: BA D7 51 FC 8B F4 96 1B 3D A9 6F 7F 2C 9D 6C 62
[EN5]  484: 95 62 B8 71 A0 05 CD EA 4C 51 1A 87 7E 60 21 B2
[1P5]  485: 62 C8 82 50 8A 11 DD A0 EA D9 9D 88 A7 E8 27 D1
[1S5]  486: 1E 1F 20 93 91 27 C5 E0 AB 14 C3 75 83 A9 A7 9E
[TS5]  487: 82 DB 03 F1 58 CE AF 38 9A 29 C5 A8 A4 2A 46 5C
[1K5]  488: C3 A5 53 87 0F F3 25 A8 C0 B1 62 07 84 B2 A5 18
[HN5]  489: AF 52 8C 77 0D 97 4E 7D 9F 4C D6 3A 2A 55 57 D6
[UM5]  490: 0B B8 22 1C A6 03 2A 36 E5 8A 05 4A 5D 31 54
[7N5]  491: 37 4B A7 AC 02 42 31 F5 30 AC D8 64 98 41 14 6B
[MN5]  492: 95 62 B8 68 D6 4D D9 01 68 5F C3 14 63 0B 5C C5
[KM5]  493: 40 4B 07 E3 DA 58 30 4D C1 D4 E4 A4 D2 A4 89 02
[6M5]  494: 8C F0 80 D6 A4 55 8D 86 17 96 2A 48 B8 E5 6E C0
[VS5]  495: 84 5B 1E AA C2 02 CB 86 2A B6 6C 1D CA 96 27 4C
[4W5]  496: 14 84 16 16 C0 11 51 12 35 52 4F 04 CD 4E 2A
[1M5]  497: 49 E4 83 3B 22 88 21 47 A3 1D C4 95 F1 0F FB AB
[ST5]  498: 3D D9 51 2C CB 4C 69 59 50 C5 96 AD 43 59 71 FE
[4R5]  499: 0D 5D 71 4A F7 F4 20 EA A9 A0 69 0F 11 97 BC 20
[5G5]  500: C8 14 46 96 67 87 18 21 97 BD 22 04 41 1B E0 C4
[6R5]  501: 1E 9B 12 45 E6 AB A6 1A 31 62 C4 8F 36 12 5B 2E
[BE5]  502: 17 14 46 09 A4 2B 2C 70 40 2F 48 65 D3 25 25 1C
[TM5]  503: 40 E6 78 D0 A8 04 04 92 CF 8C 76 24 CB 4B 61 0B
[8L5]  504: 3D DA 73 C3 68 AC 40 0F 1C 4E 0A 6D 06 A1 D6
[0C5]  505: 03 6C 41 00 09 04 95 93 58 44 14 62 77 2D 52 87
[SH5]  506: F1 28 19 23 C6 F5 08 97 39 52 A4 B8 01 74 87 8B
[ZL5]  507: 11 2B 3B DA 57 8A 44 37 F3 5D F8 32 02 A3 30 D7
[5S5]  508: B9 0C 65 38 AC E3 6A C0 2A 96 0F B7 CC 3E BC 11
[UR5]  509: BB F8 11 7A 74 C4 C8 F8 0F 3C 50 A4 4C 97 22 F6
[4N5]  510: 47 BF 98 91 95 5D A6 6C 17 20 C2 19 B2 C4 35 D2
[HV5]  511: D5 21 AC 88 AB C0 47 1F 2D 60 67 96 8F 5A D6 08
[MG5]  512: 20 8D 6C A0 0B C8 1F 14 8F 03 6B 36 88 35 01 E1
[CU5]  513: 00 4E CC 51 FB A8 E4 14 DC EB 51 43 D1 CB 8A DB
[6U5]  514: 00 6B C2 3F 43 ED 17 5D F7 3D EC 24 DC 0F A4 52
[RS5]  515: CF 07 04 8D B0 B8 CA 83 87 E7 C0 51 3A 79 9F 1F
[WM5]  516: 40 D1 81 FA 28 B6 3A D4 5D 48 48 C3 2B 1E 30 48
[UR5]  517: 44 D1 4E 11 88 CB B6 EE A1 43 61 D9 AA 90 86 4C
[5R5]  518: 56 60 0E 93 FF 1F 0E 0F 44 E0 03 1E 98 80 E8 FC
[KH5]  519: 05 12 46 37 EE 05 7F 3C 25 50 96 A7 F4 C1 53 42
[1H5]  520: B7 86 26 5A 90 4F B7 57 2C F0 14 07 4D 40 13 32
[ZN5]  521: 91 25 27 8B 3F 78 8A C3 90 98 34 4A 20 C8 F3 7F
[2N5]  522: 38 C5 2D D7 0F E0 29 12 D4 53 6C DE C2 4A 14 D4
[XH5]  523: 6D 31 E6 27 02 47 15 F8 3C 01 49 2C 20 F2 0C 87
[VU5]  524: 4F 26 30 D0 D9 30 0A D3 8F A8 BA 42 8A 39 6A C9 F4
[VU5]  525: 79 1F 5E D4 84 EB 68 10 2F FD BA AA 7E 2D C5 44
[WE5]  526: 34 D5 4D A4 53 07 83 05 83 18 10 08 4C E7 D0 12
[EW5]  527: 48 4E 8B D8 B5 D6 FF 5A AA 18 8B A6 BA 89 74 2A
[SF5]  528: 18 10 1C 32 B0 8E 58 92 B2 62 54 34 E5 81 C0 04
[6H5]  529: 03 54 01 7E B1 A4 2A 46 0A 58 79 10 4B 06 23 DB
[SP5]  530: B2 0F E1 B0 92 9B 52 58 F2 41 8B 0B 91 B0 AA 0A
[AN5]  531: B5 A5 B0 30 26 98 43 29 76 8A 44 A9 AA 82 B9 4C
[4N5]  532: DF A9 12 2D 0A E4 84 24 0B 2C 3B 94 19 12 9B B7
[3N5]  533: A0 39 A8 15 8A BA 5A 84 4D 0D 8B 7B 15 2C 4C B0
[6X5]  534: 76 95 E1 15 3C DE 45 2E A8 DE C5 DD 2B 12 99 BA
[5H5]  535: 81 4D 15 DB F2 55 96 31 01 43 68 C5 67 CE 08 36
[KK5]  536: 58 24 8A E9 A8 51 8B 16 2D E2 93 26 1A 34 82 83
[2T5]  537: 38 4A F1 45 0B 62 7B D4 88 11 AE 0D DB 34 21 85
[9H5]  538: 32 15 34 1F 40 AC 30 5C 89 2A 26 2B 52 A4 4A 85
[ZP5]  539: 17 DB 4F BF E5 B8 3C 45 28 CB 96 D1 8E 20 89 57
[8M5]  540: C0 76 28 9A 10 66 99 0A 9A 26 E3 C8 5C 66 00 55
[VW5]  541: E4 55 AF FE F3 01 EB 3B 9F 73 8C BE 0D 56 66 50
[7M5]  542: 95 03 AF 3A 60 D9 81 53 1C B0 AC 14 96 B5 78 D5
[7S5]  543: 29 CB 56 6C CB 1E 3C B8 90 65 8F 0B B0 C9 B2 73
[KA5]  544: 04 4B B3 4E 0D D4 D7 FF ED B7 AE 0D E7 DE 82 56
[WP5]  545: F0 79 13 5B 9E 49 F1 31 96 E7 50 B8 D3 58 B6 E2
[5T5]  546: 96 45 A8 96 A5 B8 65 B9 C9 2D FB 11 56 3B DA F2
[ND5]  547: 11 EA 27 0A 25 68 6A 01 60 38 10 15 8C A0 31 1C
[FW5]  548: A0 CF 5F CA 6A 1D 15 B3 FB D4 EA C8 91 5D 3D 6A
[YM5]  549: F5 BC 42 73 C7 8E 57 45 9F 13 8D B2 03 D4 BF 45
[WN5]  550: D9 34 73 4C CA 16 3C 7E 60 05 8E 58 1D F0 03 D7
[XS5]  551: 44 0A 8D 23 C8 B7 A3 3E B0 D4 D7 FF ED B7 AE 0D
[5L5]  552: F4 E8 82 F6 81 96 94 F5 61 69 5D 41 6B FE 45 41
[CL5]  553: EB 43 46 05 0E 07 B0 15 BD C4 77 21 0C CF 04 47
[YP5]  554: 2F 14 87 0B 43 05 31 FC 34 EE 45 06 18 33 FC BB
[6P5]  555: 57 C4 08 D0 48 9A AF A0 70 B7 62 06 2B CE F3 2F
[PM5]  556: 52 EC 51 96 19 E5 3B 40 0D FA 1C CB D1 1C 42 C5
[5T5]  557: E3 50 C9 A8 BA D2 9D F0 01 7A C0 BA 50 66 F7 50
[WU5]  558: 3D 4D E8 B3 A6 4A 80 DB FC 78 A3 05 0F 4C B8 7C
[KP5]  559: 15 AB 07 5D B0 0C D7 26 28 17 A5 B0 AC 8E 96 C5
[ED5]  560: 68 1B 1F 14 42 E4 80 13 34 90 3C C3 69 81 1D B0
[9T5]  561: 17 5B 92 26 54 CD AA D8 6B 57 E1 4C 5F C9 B0 02
[ZF5]  562: 23 B8 22 6C F0 52 45 88 B5 E3 82 48 15 49 35 68
[6S5]  563: 95 A2 96 AB ED 74 9E 1D 7C 61 1D A5 E2 65 39 1A
[DL5]  564: 41 CA 0A 62 49 5C B3 2A 2E F9 02 F1 96 17 08 3B
[UF5]  565: B8 56 D8 01 B7 3A 2C 28 12 64 41 B4 E0 A9 22 22
[AY5]  5661: B9 41 E2 A6 96 43 D2 A8 8C 2E 32 DC 0C BB B0 9F
[KK5]  567: 60 10 86 21 A5 FF 42 19 EE E5 1A 0C E4 6A 54 66
[ZR5]  568: 1E DE D5 02 4C 45 18 8E 16 65 09 F6 2B 14 C7 9A
[MU5]  569: C9 8F 16 60 C9 2A CB C4 35 35 BE A4 71 7B A7 F6
[DK5]  570: 89 60 6A 25 DB F5 06 8A D3 E8 03 B4 F0 19 57 82
[XN5]  571: DE AA 60 41 7B 1D A0 D8 00 FD 82 56 34 08 87 B2
[SN5]  572: 4C 5D 53 B3 E5 3C 6E EF D4 3E 05 27 F0 A0 41 85
[NM5]  573: 58 2E 2D A6 62 55 51 05 2F F8 37 3E 90 62 FF
[WV5]  574: 59 AC 58 9F 16 17 8A AA D8 2B 58 F8 DB 35 CB 53
[HT5]  575: 17 B8 EC 96 95 4B B1 F0 C9 1F 0C B6 2A E0 14 AE
[VP5]  576: A2 98 A3 2A 16 A9 2B B6 D7 9A 42 B1 4F E5 4F A5
[EV5]  577: D8 A9 14 AF A2 4C 53 8D 4F 65 E0 59 B1 95 C9 A9
[AG5]  578: 16 5B D6 2A C5 8C 4F 43 D9 78 2A 20 75 39 01 00
[LL5]  579: 0A D0 78 85 E4 31 DA 7C E0 B0 1B 04 41 DB 44 B4
[QG5]  580: 8C 1D 2F 0A 14 EC 68 20 71 53 F9 0F BE DA 48
[4R5]  581: EA 4D 44 31 15 AD 05 AF 74 5F 1D 4A 4D C4 B6 82
[6L5]  582: 2F 60 C3 E2 C0 62 10 30 37 0B DC 0F 0C 53 00 C1
[4H5]  583: 11 EA 00 AB 6D 85 81 45 C2 64 08 A0 18 CD 10 46
[AG5]  584: 31 E2 29 98 6B 98 74 11 6A B8 00 3B 09 07 80 A8
[1H5]  585: 20 05 E3 10 8C 69 02 2E 28 DB 06 9A 37 11 8C 60
[5V5]  586: D8 20 A8 63 DA 28 98 0D 14 A4 E1 E4 49 4A AA AE
[5M5]  587: 58 2E E8 0D 84 7B 30 D9 60 82 01 3A 90 A6 C0 70
[HL5]  588: 05 34 25 C1 03 98 C8 18 D3 C0 A1 0D 58 5B A1 94
[8G5]  589: 06 25 24 A0 45 0D 58 E4 68 20 46 03 49 B0 80 DB
[EE5]  590: 12 98 22 02 D6 5F 40 B9 4C 40 4B 22 30 C1 68 69
[UT5]  591: BA 8D ED 82 05 8B 06 A5 AC 40 FA 3E 22 01 E2 89
[9M5]  592: 58 ED 14 08 3B DB 04 50 B1 A5 EC 83 42 04 D2 4D
[YN5]  593: AA BA A4 51 A8 F8 29 40 C0 E4 55 70 9B 79 EB 66
[8D5]  594: C1 4F 34 4D D2 19 1F 60 68 27 40 B4 15 17 3D 81
[CS5]  595: AD EB 26 0A 6A 71 2F 18 54 5D 80 6E 80 C1 E1 AA
[7P5]  596: 0C 0A EB 69 B0 10 21 04 5B 0D FB 84 60 BB 86 4A
[0M5]  597: A7 9E 06 0A 03 CB 41 F5 34 18 AD 0A 36 B0 8E 19
[9H5]  598: 23 B8 8A 0A 95 4E 1D 0D 64 09 C7 04 8A 81 11 50
[MN5]  599: FE 8F 98 15 3F F3 59 B1 0A 1F 04 40 69 FE 37 C5
[XV5]  600: 52 C2 89 DA 05 29 EC D9 F5 FA 21 3C 89 C2 EA
[0M5]  601: D9 D6 50 D0 0E 0F B0 39 CC 81 3C 97 0B 0A 07
[3W5]  602: D0 E1 3F B4 AC CF 80 B2 62 2A FA 1A 47 0F A7 FB
[YN5]  603: C6 5C 96 4E D5 13 FF 1C 01 81 24 54 91 F2 9A 3F
[EL5]  604: E0 19 0C 54 95 41 31 29 EC 03 B4 00 EA 64 2F EA
[3S5]  605: 84 63 8A 89 68 6A 1C A6 9E 0A 9A 6F 50 75 E5 B9
[5M5]  606: C1 84 27 84 FD 41 C1 E6 00 20 18 0E 1A 18 AD 18
[BG5]  607: 02 5A 1C 9E 2B 28 10 E7 90 18 54 31 03 B2 2B 18 C7
[1P5]  608: 3C C9 EA 59 D0 84 55 E0 51 C8 CD 2D 53 28 44 B5
[2L5]  609: A4 88 1C 45 B1 A3 B0 44 6E A9 58 61 5A B2 8A A3
[UP5]  610: B0 0A DC 53 E0 51 AC CD 04 7B 4A 2D 29 22 47 51
[PM5]  611: AC 28 2C 91 5B 7A 4A 8A 65 DC 51 3C 05 C8 28 84
[ST5]  612: C4 D4 7B 4A 29 6D C2 0F 8E A2 18 2E 5A BA A9 74 EA
[6K5]  613: 29 20 A2 58 9B C5 2D 7D 53 49 61 55 51 61 49 5B
[2L5]  614: 54 5E D4 53 80 55 67 16 24 AA 49 54 96 88 32 51
[3H5]  615: 74 62 7D A2 92 48 7D 09 B0 9E 92 28 B0 2E 50 69
[ML5]  616: 20 EA 40 05 3E 6B 7C EA 0D 20 1F 98 15 B2 80 B1
[7K5]  617: 7C EE 14 7E B2 0B 28 36 A2 1D 30 05 D6 55 90 14
[VK5]  618: 11 60 2B 05 10 DC 79 0A A4 08 A8 63 FF 93 BF 25
[WN5]  619: E1 34 B6 A3 1C A6 82 B0 04 AC BD 2E 18 E4 06 77
[WS5]  620: 18 2C 65 AA 0C B4 12 AE 1C 01 A5 F5 6E 32 C0 40
[7L5]  621: 01 15 00 03 0B 29 5B 43 22 C9 D8 52 1C C0 F0
[1W5]  622: FF C9 C7 EC AB 4C C6 30 6C 6E 41 C1 14 38 D5 0A
[CL5]  623: 01 D1 58 CA 05 81 02 5B 4D 0E FF 82 6B 93 0C 02
[3P5]  624: 09 50 76 20 B3 99 CA 2C 68 1F 38 CF 87 6F 6A 67
[MV5]  625: CE 9C 39 3C 5C 03 C1 D7 80 F0 64 D9 9F F9 B3 A3
[3M5]  626: FD 07 D0 01 74 03 DA 9F 03 60 60 39 30 5C 03 C9
[4H5]  627: D7 80 1B 0A 28 41 8B 4C B1 65 67 F6 14 C0 0D 05
[7L5]  628: E8 03 80 96 29 E5 41 81 7A 0A 16 85 8F 20 51 AC
[8N5]  629: 80 78 54 BC 90 A3 8C A6 EA 0A F2 43 A2 72 04 39
[HT5]  630: 81 D9 B6 95 15 25 C8 AD 1F 04 41 7F 2A 80 A0 A2
[PK5]  631: B5 3E 29 8E 07 73 28 1D 5A 46 94 A5 22 98 19 84
[AP5]  632: 8B 28 E6 80 8A 55 A0 A8 E2 7F 00 42 2B 1E 18 E4
[HN5]  633: E9 80 17 CA AB 84 24 E6 21 8D 07 13 16 C9 A0 3E
[TH5]  634: 07 43 24 1D 88 05 D2 4C C6 16 38 87 92 55 2D 58
[TK5]  635: 89 77 16 A1 AD 98 8A A6 BA A9 74 AC F0 40 A8 49
[GK5]  636: 70 EF 43 A0 DB F0 68 A8 33 D3 41 AB 04 5D 31 11 4D 75
[RF5]  637: 13 EF 10 48 03 D6 53 52 DC 53 22 B4 15 13 D1 54
[PH5]  638: 37 85 34 30 4A 41 14 B3 A7 A4 B8 A4 24 E8 8A 89
[SU5]  639: 68 AA 9B 48 47 ED E3 FF 28 DD 42 B6 CA 18 9B 07
[0E5]  640: AC 12 32 44 89 57 20 30 D8 15 68 38 84 EE 0D 01
[0M5]  641: 0E D1 CD 81 07 AB 59 C3 40 F5 96 33 7F 00 60 D6
[HK5]  642: 4D 88 18 D2 8A 02 27 41 93 15 16 5F 3C B3 07 4E
[BV5]  643: 0E 07 51 3E 8A D0 F0 68 0A 9D 80 A8 DD 31 11 4D 75
[SF5]  644: 3F C2 6F 39 41 75 98 10 BA 29 17 43 16 54 93 C1
[AG5]  645: 48 D0 1C 8A 4C 25 54 17 18 14 89 80 5B 41 0A D2
[7H5]  646: 32 47 04 B1 38 A6 62 30 17 0C C1 13 01 BD 80 EF
[RD5]  647: 72 81 08 41 30 2B 22 D9 E5 99 8A 41 57 38 48 48
[1M5]  648: 64 77 32 C3 52 23 5B E1 FF 42 5B 07 8E 08 ED 36
[GL5]  649: C5 06 43 01 1D 2C 97 CA 62 AB A3 46 38 B2 5F 59
[ET5]  650: 1D EF 77 3B 36 5E F1 0F FB AB 1D D5 18 5A 85 68
[UN5]  651: AE 92 90 3E 27 E1 11 E1 24 86 A5 BB 8C 55 19 18
[8N5]  652: 18 46 0E 7B 80 94 5C 84 A6 79 C4 08 65 AE 86 EA
[RR5]  653: 68 87 3E 40 12 15 5A DB 5B 57 FD 27 F0 4D F4 9A
[7F5]  654: B2 0A 44 95 65 2E 10 48 99 49 64 20 18 49 66 2B
[PX5]  655: 0C 2D FF 09 AC 01 16 A0 F7 6F 46 BB 8C 36 BE EC
[NR5]  656: 5A 1A C1 A2 2A EA 80 44 ED 55 59 92 4D 28 99 F2
[0W5]  657: 9B 07 B8 02 59 A6 D6 76 7A 9F 01 51 B6 DD 71 F5 82 57
[EH5]  658: C6 A5 07 F0 75 40 48 90 40 CD C1 8C 32 9A 03 86
[ZT5]  659: C8 45 D5 95 80 9D 27 D2 B4 83 C1 FC A4 CF 0E D9
[NM5]  660: 40 D1 66 D6 09 61 FF A8 05 B6 4C 1F 28 45 85 1A
[BT5]  661: A0 81 25 1A 9E CA C8 8A CC 30 E3 00 2E 18 68 8D
[8G5]  662: CC 2B C8 41 C4 E8 11 16 08 80 0B 06 A2 42 50 B1
[CG5]  663: C3 04 45 C4 E8 15 5A 99 7C F0 26 D3 7C 6D 18 60
[GH5]  664: E0 B0 98 A9 30 20 EC 80 2E DF 15 52 58 39 1D 80
[DP5]  665: 4E 31 05 DE DD BC 79 58 86 57 04 3D 5C 21 95 62
[EE5]  666: 22 9A E8 66 E9 54 3D 15 34 63 3D 07 54 32 55 25
[WF5]  667: AA 67 07 5E 5A 41 72 07 E0 85 05 56 06 4A 50 80
[PN5]  668: 60 44 C4 03 6A C9 21 2F 0D A4 98 8A 26 56 C0 A5
[9L5]  669: 74 97 3E 68 2D 8A 2F F5 12 34 1D C1 F4 88 82 C5
[PR5]  670: 5A 1A C1 A2 2A EA 80 44 ED 55 59 92 4D 28 99 F2
[0M5]  671: 08 0C A2 83 38 B1 6C 79 01 51 B6 DD 71 F5 82 57
[0R5]  672: 21 C1 15 F2 04 15 F3 1D 21 4F 70 5B 25 FA A5 FC
[1L5]  673: 16 BF 41 34 DA 6F 5F C4 1D 21 28 13 40 A4 6E 6A
[4L5]  674: 99 50 26 46 6B 4C 17 61 A3 A9 E2 4B 18 3D 96 48
[3L5]  675: 74 D4 EA 61 C9 86 B3 56 B0 32 1C F8 60 61 5B 39
[5N5]  676: BC 99 FF 83 98 3F 7F 86 C0 7F C2 37 11 32 48 04
[2E5]  677: 93 0F 8A 0B 3C 20 10 61 25 F8 34 78 14 43 59 42
[FV5]  678: B1 ED F6 4A A2 3D D4 F2 45 87 C6 5B 9E 40 B1 43
[MM5]  679: 18 27 01 30 3E 50 0B F8 1E A7 20 CC 89 F2 AF B0
[6R5]  680: A4 03 92 28 1E 43 9C 26 E5 BC 86 38 4D 92 EA 8F
[JK5]  681: E5 AE 2B 3E C2 38 5E 80 07 21 06 0B 83 AC 48 38
[XM5]  682: 70 94 AE 2B 01 A2 53 76 DE 55 65 30 8F 30 DE C3
[PS5]  683: ED 4E 8F 2C 1E 74 4A 60 FE B8 38 C0 A9 F2 96 89
[3M5]  684: 6B 6A DC 7D 02 7A 27 F6 89 60 5A 05 11 84 0B 0B
[WT5]  685: E0 66 DD FA 88 D6 D3 4F C5 54 08 E6 09 18 21 3F
[4R5]  686: CA FD 27 88 65 EA 9A 1A BF F9 94 B8 15 B6 43 40
[UP5]  687: FB A4 82 3B 11 2C 8A 2E C0 67 BA 88 5E 34 C5 01
[6T5]  688: 12 57 29 B8 B1 1B DA FF F1 FF 04 A0 38 B2 62 AE
[AH5]  689: 2C 9A E8 A6 D2 A9 7A 22 68 2A 07 20 92 48 51 71
[WM5]  690: 18 8D 68 C5 65 9C 09 F0 86 A5 FD 3C 44 73 10 C5
[7P5]  691: 54 34 D5 4D A5 13 F5 44 CE 60 5B C1 7F 50 75 45
[DN5]  692: 6F 22 01 05 F3 8B 0C 67 33 27 3B A0 BE 6D 65 46
[4G5]  693: 88 36 11 7E 58 D0 2C 3C 71 14 93 CB 55 39 8E 0D 41
[9G5]  694: 55 8B 58 56 01 20 19 C4 0A 3E 83 AA 33 17 50 25
[UH5]  695: DB 7B 84 02 9C 52 10 72 F2 80 42 19 79 C1 EA 25
[KH5]  696: 74 8A C8 0A 52 A1 68 15 CB AA 02 24 53 89 90 46
[9P5]  697: E6 F9 B2 F0 EC F1 6A 6B 82 47 2E 08 0F 18 91 26
[MX5]  698: 91 AD BC E9 F1 38 41 FB 4D 9A DC 78 D7 89 1F 9F
[VW5]  699: F0 42 28 4D E2 C4 BE 9E FB F1 1B 6A AD B6 2A CD
[VH5]  700: AA A9 8A 57 5D 55 51 34 92 2A 45 A5 F2 8F 39 20
[5M5]  701: 2E 88 13 8D 73 67 8F 38 91 88 01 E2 04 B9 DF F1
[SM5]  702: D9 3E 71 6E 13 FD 01 32 65 1A 37 C6 0B E1 33 8E
[8R5]  703: AB 84 77 65 3E B7 2E 54 83 A5 A7 16 D5 82 CB AB
[SH5]  704: 2E AD 55 55 B1 1D ED 90 0B B0 32 29 60 C
```

```
[AP5]  825: 94 E0 7F 42 5E E8 B2 62 F4 8A D3 31 7E E7 78 1D
[7L5]  826: 7Ø F0 92 C4 ØF DE 15 E9 55 A4 65 91 34 57 46 5D
[TD5]  827: 19 73 25 F5 95 58 Ø1 Ø4 6F FØ 64 46 Ø8 16 2B 66
[9E5]  828: Ø7 C8 12 6Ø 52 ØA C3 A7 A4 38 37 66 43 8Ø 51 35
[AZ5]  829: B6 Ø6 18 6C DE FF Ø2 AE 9E 11 DØ D7 9F FC B4 D4
[AP5]  83Ø: 8Ø E5 F2 Ø4 4E D1 3A CØ E4 16 BD 3B FC Ø8 93 23
[DY5]  831: CC EF 3Ø 8B 9F A4 EE D3 2C C1 E6 Ø4 AF E7 1C 7E
[CL5]  832: DØ 33 CØ 29 C1 4E 13 6Ø 74 8C 4F 81 BE Ø4 CA 92
[3L5]  833: A5 74 9D C1 Ø1 5B Ø7 8C 1D E5 73 97 13 51 D6 44
[AG5]  834: E9 11 E5 45 94 1C 51 66 47 49 9C 22 D5 81 34 Ø7
[ØL5]  835: 5A A7 68 74 4Ø FØ 27 Ø7 92 9D 99 7Ø E6 47 8E DC
[DP5]  836: 67 CE 9D Ø8 73 E2 4C 8C D7 27 66 9E A8 DD 63 7Ø
[ML5]  837: 8D C5 35 72 Ø7 Ø9 F8 E4 4F 8Ø 39 A6 1A 2Ø 4F 8Ø
[3H5]  838: 73 Ø1 56 Ø5 58 69 49 5B 34 EE 31 F9 C1 8D 14 39
[PP5]  839: 52 8E 4E 31 73 CC 8F 1E 3D 8A A1 3F F4 ØC 5B 34 A9 E9
[7P5]  84Ø: D6 F4 77 99 C7 13 Ø2 3F A9 FC A0 76 81 ØC 2B A6
[SU5]  841: AC F8 BØ 22 F3 8A DF FA EC 14 Ø5 1C 75 FØ 93 DØ
[XK5]  842: Ø1 Ø5 4F 1D 3Ø 6F 9E 5F 1D 76 54 A9 55 25 77 15
[YS5]  843: 49 17 7C 56 8F CD 55 EA E6 91 F5 13 A2 3Ø CØ E9
[PR5]  844: 26 59 1F FC FB B2 7B 86 BA 8E 4A BØ 6A CØ 9A Ø1
[BU5]  845: 4Ø 67 AC 1A 96 8F 88 F1 F8 41 6B AA FB 49 A4 Ø3
[5T5]  846: 51 ØE 44 3Ø 35 1A 38 C6 86 Ø2 DB 8C 74 A1 13 BA C6
[XW5]  847: 8A 1D 37 5B 1C ØE D8 77 C7 DD 19 ED 7A E4 EE 91
[AY5]  848: 6E C7 EF ØE 55 97 EC EE D3 B5 4Ø E6 16 6B 46 FC
[YS5]  849: 91 95 67 A5 E8 31 E6 C6 98 DE 63 8A 3E 79 FC 24
[FP5]  85Ø: 4C 98 33 51 5A 21 AE 78 8C D6 61 AE 14 88 5C A6
[VS5]  851: 74 85 BF 39 4E Ø7 BC E5 DE 85 1F 94 9D 11 B2 47
[5X5]  852: AF 28 F9 A2 FC 8E 32 AB 4A EC 2A A7 ØA 9C 2E BØ
[BL5]  853: 2Ø 4Ø 3F 1Ø A3 35 Ø3 8E 86 75 A6 13 BA C6
[PL5]  854: 11 34 85 6Ø 3C 3B 6Ø 8F 2Ø 77 82 BC ØA 92 23 C8
[LH5]  855: 89 2E 71 82 E4 ØC 52 23 88 CØ 3A 61 4C A6 ØØ 9E
[ØP5]  856: DE 2A 4Ø CB ØØ 21 26 CC E8 51 25 CØ C3 3F 4B 26
[ØW5]  857: 7C FØ BB 27 DC 9C 33 F4 C9 A9 Ø9 AF 7A 98 9E A6
[BK5]  858: C7 AD ØØ 71 62 E4 Ø8 5Ø 36 CØ 86 Ø7 8D 1F F4 6A
[XU5]  859: BØ B7 C1 B8 15 D7 7E B4 FE F1 E9 CØ 87 Ø3 E3 C6
[WR5]  86Ø: BC 3E 1Ø E1 C4 D5 14 B9 Ø3 EE 3A B1 FB 44 EA 19
[GW5]  861: 8Ø CC 1D 6Ø B1 AB C6 CB ED 9Ø 2A C6 FØ 26 C9 CØ
[FN5]  862: 9D ØØ FF Ø2 48 C6 BF Ø2 84 91 F4 18 B1 4D C1 82
[1S5]  863: FF F6 97 63 7F 48 91 EØ 45 8B D4 Ø9 26 3F E9 55
[BG5]  864: A0 77 8F 44 4F 12 3F 19 3D C6 16 Ø6 59 52 24 C5
[Ø5S]  865: C1 14 8A 75 3B C5 EF 39 A5 1F AC D6 F4 Ø9 BD 14
[5M5]  866: F6 Ø9 7D B3 FC 9C 3Ø 73 42 AA Ø9 ØF 26 D4 9E 7Ø
[KS5]  867: 7A 86 3C 5E 1D E0 C0 96 46 5B 1A 6F D9 51 2B B5
[5Z5]  868: CA F8 3C 7Ø 7B 69 A1 FB 85 B4 9E 1A 63 F8 DB
[DV5]  869: 31 71 7A DC 6B 61 F8 ED 16 BE 34 6C 99 DE Ø1 5E
[4M5]  87Ø: F7 29 1C 43 1E FE ØE 58 A1 C2 86 ØA 21 2A 9C A8
[5U5]  871: DØ A2 C2 8C ØA 39 2A FC AE 2Ø 8B 1B 3F 59 31 A2
[FR5]  872: CC 88 35 63 62 8E 38 33 E2 F5 88 A9 4B 4E E7 F9
[HV5]  873: DD A7 C2 ØA D4 13 AD 78 BØ A2 F2 8A C0 ØF 36 3F
[TT5]  874: 1Ø DA 53 3F 78 ED F1 3E D8 F3 A0 18 DF 18 4Ø 8B FØ
[3L5]  875: 31 Ø3 32 ØD D8 A2 B7 ØE 98 3Ø 44 F1 38 21 4E ØD
[PN5]  876: D8 13 A2 CB B0 31 31 56 ØF 28 DD 63 74 8F 14 5A
[FT5]  877: 7F 93 B4 F2 F8 C1 89 23 B5 8F 8C Ø8 F8 37 6Ø 4A
[2R5]  878: 2Ø 69 82 BC 4D 11 95 AA 2B 39 D3 19 6Ø 4D 8E 37
[GU5]  879: 39 E6 E4 B8 93 63 4F ØE 7E 7E E0 41 99 25 6B FF
[KX5]  88Ø: B4 ED 81 B7 9F CA D3 BB CF 9F 1E 9B A7 DØ 14 A6
[CL5]  881: E9 4Ø 91 C3 9F 32 4C C1 D8 E7 44 A5 1Ø E7 44 A5
[WP5]  882: Ø3 Ø3 1E 3C 49 6Ø 7A 9D 1E 19 Ø9 AC 41 39 63 FC
[PW5]  883: EC B3 B8 Ø8 A3 D6 3E F3 2B C7 ED 1A 27 B6 B4 D8
[VW5]  884: 72 76 CB B1 3D 51 F7 E4 DB 33 F6 CD 2B 6Ø 8E FØ
[SG5]  885: 17 C9 5Ø E4 43 91 12 45 56 14 89 5B 84 E1 6F 3B
[3S5]  886: 61 56 1B 5Ø 7C BA CD 5E 5A 1D DØ 39 Ø9 C6 D9 66
[YM5]  887: CØ 5C 8F 7F 4Ø 83 37 D3 28 CA 1E 6F 7E 54 1B 7Ø
[8N5]  888: AØ 47 Ø6 89 3A C6 3B 65 3F BA 96 Ø9 E6 88 FC CØ
[DP5]  889: 69 B4 DA 53 E7 A9 3D E3 DF ØC 4B 2B D4 85 2D 17
[FU5]  89Ø: 4E 5C Ø8 F1 A7 CE Ø2 C7 4F 5D E8 72 A1 F4 85 DD
[NP5]  891: ØB 96 74 88 Ø8 D2 3D B5 C3 EC Ø5 E5 AE F4 BB 72
[3U5]  892: D7 5D FB 58 A5 Ø9 5A 3C ØC 3Ø 91 E6 1B 21 57 DF
[VL5]  893: F6 CF 31 6A Ø5 A8 1D 2Ø F5 87 DF ØF 86 Ø6 2C 33
[TK5]  894: 61 DD ØA 8D B1 F5 ØA 3B 13 6E AF Ø8 33 E1 2C 26
[GZ5]  895: EE F1 AB C7 E8 1E 91 9F EC F3 A6 Ø3 6A 8F E9 34
[UM5]  896: BØ 9C 5E 4C C0 92 DC 25 A8 E6 9Ø B8 Ø5 D9 ØE 57
[ØN5]  897: 7F 29 93 22 Ø6 5D 3D 3A 4Ø 12 B1 74 4C 6E 4Ø AC
[MK5]  898: 94 87 4Ø 87 95 63 18 76 12 B4 79 DD 8C D8 33 E2
[AT5]  899: CD 88 39 A3 67 9E 64 ØF CE 1C 82 E5 73 16 E5 69
[XS5]  9ØØ: F4 FB D1 9B 43 75 ØF 45 7B B2 E6 5Ø 9C 43 B7 ØF
[3R5]  9Ø1: E5 7E 53 66 AD 39 D4 FB D1 BB 51 3B 53 ØE 65
[WY5]  9Ø2: 39 F4 F9 DØ EE 1E AD FA 9Ø D5 51 ØE 5D 35 FA F9
[5R5]  9Ø3: 5Ø 1D 3E 1E 3C 49 6Ø 7A 9D 1E 19 Ø9 AC 41 39 63
[1L5]  9Ø4: 35 76 2D D8 BB 6Ø 1Ø C3 71 B1 1A 85 6B 28 C5 76
[UV5]  9Ø5: E3 EF 16 65 8F CC 3C 12 FA C8 DB 23 C7 EA D4 CC
[1N5]  9Ø6: 83 52 35 AA C7 9A Ø3 25 A6 7C 98 B2 E6 86 2C 6C
[RH5]  9Ø7: 73 2Ø CB 81 49 1F 14 97 3C CC 81 33 Ø7 5E 97 8Ø
[XS5]  9Ø8: 2A 77 1F 28 7D 6Ø E2 81 CD 2C E7 81 AB 24 ED 83
[YN5]  9Ø9: 61 11 6A 47 48 7D 6Ø 5E B4 CF Ø9 7E 9C C9 71 E6
[TR5]  91Ø: F7 99 B4 DA 11 56 25 1Ø C7 A7 E2 9Ø 36 42 AF
[2T5]  911: ØE 42 3F CF 59 2B C1 9Ø Ø1 B1 C6 D4 4E DØ 35 C2
[VT5]  912: AE ØE BB 3B 94 4A A0 ØA 4F 91 E4 Ø9 12 77 48 E5
[9P5]  913: 4D A9 C6 Ø4 4F Ø6 24 8A D1 38 41 E8 ØE 74 77 E1
[GM5]  914: 37 F7 1E E4 7B BØ EF 41 BF Ø7 8A 38 15 5F 56 13
[HR5]  915: 73 FA 3A Ø9 D5 7E 18 2D Ø9 5Ø 7A C6 EC 3D 48 43
[6V5]  916: F3 F6 78 E3 2B 34 7D 8F 1D 7B A0 7A 3Ø 7F A1 1E
[6M5]  917: 75 F6 3Ø F8 1D C6 C6 Ø1 A2 1C B9 7Ø A4 CB 91 11
[DP5]  918: 47 D2 9C 73 F9 71 A4 CE 91 DD 47 7E 1F 7D Ø4 6E
[4N5]  919: Ø8 FD A4 45 9B D9 6D Ø2 A5 AØ 8A 77 1E A9 38 83
[1T5]  92Ø: AA BC 9D Ø7 C5 3D 6A Ø8 D5 Ø8 A5 FE FE A2 5A 21
[1S5]  921: AC D2 2C A1 C3 74 98 73 79 CE 47 96 8A 8C D4 ØB
[TL5]  922: B1 4F 99 6Ø F3 76 88 A3 78 E6 Ø1 A9 27 B4 DA 73
[MU5]  923: 66 CF 9C 3D AD 63 F4 9E 5Ø 22 4A 9A 28 A1 A3 D4
[EW5]  924: 6A B2 EB CA BE 2B 6C 7E E7 E1 F6 51 Ø3 3E 21 E5
[KP5]  925: 6Ø 6F 4B Ø3 26 79 FC 35 Ø6 B6 Ø4 5D 36 46 72 CD
[8T5]  926: 4D 77 FD C8 Ø7 DD 93 F4 3Ø FE Ø9 C6 38 AØ 4A 8F
[6P5]  927: 59 Ø3 6E ØD D8 3D 2Ø 2B CE 3A 2Ø C4 98 D4 Ø3 7A
[5L5]  928: CF 28 15 21 F2 8C 56 11 5A CF 88 15 21 76 8C 9E
[3E5]  929: 64 CB 2F Ø4 71 ØØ 83 73 71 CØ 59 Ø1 92 31 B9 2F
[TL5]  93Ø: 9Ø CD 6C 81 Ø7 84 89 D3 Ø6 2C 74 1E B2 E6 D3 Ø8
[TM5]  931: DE Ø9 55 24 FD 43 16 86 6A 79 AC 1F BØ F6 Ø5 A5
[NT5]  932: FB CE 8Ø 3B 63 7C 47 A9 43 79 5D 2F BE 7Ø FB B2
[AU5]  933: EB E6 8F 51 3F D1 Ø8 4D 1C A3 F3 85 DE 17 66 5F
[KV5]  934: 28 29 EC 2E E7 ØØ 41 CF 15 4Ø 6D EE 9D E5 FØ 93
[FZ5]  935: DC 5B A8 DC 65 54 BE ØD FE 58 Ø6 DF 35 ØE AC 48
[VN5]  936: A2 78 AD Ø3 A2 69 EC Ø3 44 E1 33 3D C4 AC AB AØ
[PV5]  937: B2 6F 12 F8 7B 3C 3A FØ F8 4Ø EA 2D F1 4A D4 2E
[ZP5]  938: F1 23 6Ø 59 4E E1 6C E3 5Ø C7 AC 52 F2 DF D6 E8
[BK5]  939: 43 3D 12 77 F8 4Ø 66 FA 73 Ø8 DD 21 36 Ø7 ØC 71
[PM5]  94Ø: 6Ø C3 8F DØ Ø7 EC F5 Ø5 9Ø 3F CC EB Ø2 17 Ø2 45
[HN5]  941: 18 B2 27 68 5F AA 9E 63 BE 12 28 F4 8D DD 65 16
[Ø5P5]  942: 3F 9Ø CA DE 71 A2 4Ø 88 21 B3 6F 54 DE 91 7B 47
[RU5]  943: EB 4Ø 93 41 BC Ø5 E2 D6 89 AC 1Ø 0C 54 F9 C3 DD
[EP5]  944: 25 73 97 D8 1E A6 45 ØB 8D BC ØØ 29 ØF 1B 6Ø 5D
[VS5]  945: ØØ 33 27 C7 93 96 E9 26 5A E6 92 Ø1 A4 F4 97 55
[DL5]  946: E2 E1 32 8Ø 84 17 21 77 D9 E6 AD 54 Ø2
[GM5]  947: DF ØE ØØ 34 54 21 CØ 83 37 2E 41 23 F3 2A F1 3E
[XK5]  948: 65 DD 2A DE 6C 72 75 8Ø 14 29 5E A3 F4 Ø3 16 64
[AP5]  949: 89 3C 6Ø 5E 91 74 45 1A DF 78 7C 23 F1 8D C9 37
[PM5]  95Ø: 7E F7 5Ø BØ ØF 19 94 B8 1Ø 22 6D 76 43 7Ø D5 B9
[8R5]  951: 55 29 E5 B8 27 D7 98 66 AA 37 3E D7 F8 FF 1F 44
[NL5]  952: BA 68 52 A7 45 41 1A 2D 4A Ø5 24 ØF 12 E5 54 A8
[DM5]  953: 41 99 2D 9D 3A 14 69 D1 29 CE 1E 31 29 55 E4
[YG5]  954: A5 74 94 88 1B 1F 24 49 1A Ø7 65 ØØ Ø5 6B 1Ø D9
[1R5]  955: 1D 94 9D 2E 7D DA B4 69 51 A7 2Ø 99 BØ A4 A7 18
[8T5]  956: 1B 4B 71 1Ø E3 3B 41 8E 7F 29 4A 7F 24 DA DA D7
[RL5]  957: EA BØ 24 8Ø E9 9Ø ØF A2 F4 54 BE 3F 1A 82 6C 32
[EU5]  958: 69 11 F8 A6 1B 77 A6 45 CE 6D E9 93 52 ØD B1 B5
[RX5]  959: CA D9 9A Ø6 11 8C D6 2C C4 EC CE EE 28 DB 67 D8
[TW5]  96Ø: 7Ø 2Ø 2A 6C 11 E2 EA 5A 4E 2Ø 2A E9 A9 1Ø
[8T5]  961: B5 96 A2 EC 53 AA 6B 6A 2B 67 Ø7 76 6A C9 FE 24
[1T5]  962: 81 B5 63 EB 8E 84 24 84 14 4B 7E CA 53 EC AD 1D
[UN5]  963: 37 B9 8E 7E 2B D1 33 7D D2 AF 24 B9 25 45 3C 84
[7N5]  964: 8D 22 CD 14 ED 2C 41 26 21 6A 1Ø ØB 21 ED 89 3E
[KM5]  965: 41 F6 25 F6 56 E4 2C 7C 59 99 55 4E 12 BF C3 9Ø
[RT5]  966: ØØ 5B B4 9F 54 ØA 93 9D E8 95 A4 43 EA 22 6E 4E
[VL5]  967: 6D 3A 28 93 A8 9E 62 76 E7 95 5D B2 4B B9 4B 15
[5T5]  968: E4 5E B2 54 99 85 68 3D 5A Ø5 BD F6 BE 5D 33 55
[XØ6]  969: AD 54 D2 DD CA DF C1 79 AA 53 A4 F8 11 76 D6 3E
[DK5]  97Ø: C9 Ø7 49 31 36 19 15 37 6A C4 CC 66 BD 1Ø 38 1A
[BY5]  971: 59 DE C2 EA 9F ØC CD A3 56 89 B7 A4 68 21 3B DE
[DU5]  972: 82 AB E6 4D B2 6C F4 6B 96 52 F5 4C B2 BA Ø9 52
[GW5]  973: 12 EF 57 BA 54 88 AD F8 A7 48 99 FE 68 77 B3 ØF
[CV5]  974: DE 88 2F CD EC D4 49 9D D8 5E 13 E1 Ø2 33 62 CB
[DS5]  975: 35 F5 52 27 78 68 CD 3B FD 91 14 B7 D4 4E 0Ø DB
[HS5]  976: 9D 1C 89 BB FC 9D 59 2B 51 6B 2A 67 32 F6 5B FØ
[UV5]  977: 25 F5 A5 BE 25 A5 D6 DA 5D 21 8Ø D7 A7 C8 6C C4
[DU5]  978: 5D A8 A5 3A 31 7C 5B EA 29 EE 99 5E EE 11 61 26
[ES5]  979: 9D DF EE 5D 54 C8 1B 15 23 84 DC 47 1A D5 D4 55
```

```
[1R5]  98Ø: 4B 91 53 EA C8 8B 3E 15 F2 7B 92 39 FF C8 7D 12
[HT5]  981: 48 3B 3B 8A 61 7B 91 C7 67 57 6Ø 6D 4B 7E CE 2F
[DK5]  982: 89 83 5E 51 6C D9 6B 92 52 B6 86 CB 12 E3 6B 26
[GK5]  983: 6F 3A 68 93 24 1A 83 74 27 AD 53 5D 53 A9 5C 24
[1R5]  984: 9E AD 69 4E A5 39 71 6A AA B2 C0 E2 02 88 15 Ø6
[UP5]  985: B1 7A 21 56 46 C4 AA 8B 58 D1 11 BB 71 ØB C4 4A
[5N5]  986: 84 58 E5 1D 2B 28 62 75 46 AC FC 88 FD B8 ØA 62
[TW5]  987: C5 42 AC 86 B8 95 16 B1 87 Ø7 2D 2C D3 16 BC DØ
[NS5]  988: A8 63 A7 47 E6 42 D5 9E 53 65 92 66 3D D2 D4 5A
[YU5]  989: FA B2 D5 AC 7Ø 9E 12 FB 9E 33 24 49 56 7B C1 82
[ZX5]  99Ø: CØ 83 EØ AE 8Ø A9 B8 CC CF F2 D7 45 A1 42 2D FA
[EN5]  991: 63 2F 1C 5D 3B E1 45 A5 E2 19 56 B8 54 69 88 7B
[9R5]  992: 66 68 ØD E6 E6 7E 6F 1Ø F3 31 D1 DA DD 63 84 76
[WN5]  993: C6 2E 53 D8 25 6A 6Ø 79 BC FØ B5 Ø8 6E 4B 53 E5
[BX5]  994: B1 22 86 43 BB 94 D5 53 AA 11 92 82 33 AA 42 97
[WN5]  995: 75 9F 3A 65 B1 A8 CØ 6C D9 36 Ø3 5E 51 3B 2E D4
[7L5]  996: D1 EB 26 Ø6 73 82 25 63 7E AC D1 C3 C2 4C 5Ø 57
[DV5]  997: 66 87 B5 BB AD AD E1 62 37 F5 2E CD 93 7C 89 69
[EY5]  998: 65 5D 1B 9B 1A 6A A6 4E CA 8C A6 D6 8E 1Ø 4D CE
[WN5]  999: Ø4 EB D3 8B 38 89 D6 82 A8 B6 9A 5Ø ED 8E 5Ø 42 Ø4
[ZU5]  99Ø: 26 7C 62 98 6D B0 E3 49 BC 9D 1F BA 4F D1 A1 6E 4C
[HU5]  991: 5F 21 5C 23 6D 2A A2 84 AD 89 CD 6Ø AB D5 97 4D
[PR5]  992: B3 E5 EF 68 AB Ø5 B7 8B 21 86 C7 14 C5 A2 64 BØ
[BR5]  993: C7 49 EA 68 A7 AA 6E 33 2D E9 26 D9 E9 C9 8D 14
[3U5]  994: 96 C5 ØA E3 8F Ø8 53 4D FA D4 49 A9 B2 F6 BD 6C
[UL5]  995: 4D 22 7D 72 E8 C5 3E 76 6C A5 A8 49 89 CØ 36 29
[KM5]  996: A5 8F Ø2 79 6D ED 9E 92 42 4F 91 2D 22 B2 A5 13
[ZU5]  997: ED A9 EE AD EB 5Ø 93 B1 B2 9E 99 36 3F A1 CC 38
[4N5]  998: 5F 2F 1C 5D 3B E1 45 A5 E2 19 56 B8 54 69 88 7B
[PH5]  999: 11 4A 17 DØ ØE 27 2C 3E 7C D3 2Ø 55 AØ E8 3D 29
[SN5]  99Ø: 12 C9 32 F9 EA AA 12 38 95 57 C4 5A B3 76 3E D1
[8M5]  991: 43 8D DC A2 C7 96 1E F3 ØC 2F 3Ø A3 CØ 98 31 42
[7P5]  992: F0 59 ØA D4 7Ø EC D4 63 BE F4 28 83 24 B7 B1 F7
[XS5]  993: Ø8 1E 59 C1 FB 7B 82 E0 BD Ø5 5B C9 4B 64 96 1C
[4T5]  994: C1 6A 78 ØF D7 6C 81 1B AB 4Ø EC Ø2 3F Ø4 7C 2E
[PC5]  995: DØ A8 4Ø Ø3 31 DØ ØF 91 ØA Ø8 5E A9 4Ø 26 Ø1 74
[9E5]  996: C1 ØE 53 6Ø D3 98 5Ø 81 CA Ø4 7A 55 27 55 2Ø 23
[9S5]  997: D4 56 82 6D BC AF 28 E3 5E 6F 67 4Ø E9 1F B2
[6S5]  998: 99 67 Ø4 1F C4 A6 ØC FE Ø6 4B CF Ø3 A9 5A B3 97
[RL5]  999: 68 Ø4 5B 8C 39 31 A6 E5 98 15 29 4A A4 F8 92 22
[MP5]  99Ø: Ø3 ØA 33 45 Ø3 27 F4 ØF B6 F5 39 4A DF 39 B3 ØB
[SM5]  991: 8C 58 D1 61 Ø5 EE 17 14 BB F4 ØA 39 FC 1D 49 66
[7V5]  992: 1Ø 21 2Ø 5F 77 F6 Ø9 EE 66 DD D3 59 F1 4F FB ØC DØ
[PT5]  993: CE 22 D6 3B 48 DC ØØ A1 C7 6C 79 BØ 1B 49 FB CØ
[FS5]  994: ØF B6 34 48 6F EF 29 64 8B 83 99 CE 94 7B 59 7B
[EU5] 1Ø25: DC 39 D2 EØ 46 2C 33 B8 4F D8 BC 54 DA 2C 9E Ø7
[MR5] 1Ø26: A8 A1 7B CE 9C 52 E8 89 A3 68 74 27 6E 85 58 Ø9
[7K5] 1Ø27: 11 AB 2C 62 Ø5 87 69 EA Ø8 58 79 1Ø AB 1A 62 C5
[ØR5] 1Ø28: 44 AC C6 88 95 1E A6 A9 Ø8 62 85 42 AC 7E 88 95
[YS5] 1Ø29: 15 B1 FF C3 E0 15 E8 83 B9 FF DA D0 73 C3 2C Ø8
[XW5] 1Ø2A: 4Ø DC 1B 66 6F C8 B6 21 F5 B5 7D 1B 7A 6F F8
[MS5] 1Ø31: 2B DA 2D 36 D4 DE FØ 7B 43 8C ØD E2 E2 1C 46 C8
[8T5] 1Ø32: FB 72 E0 11 1D AF 91 63 A6 12 64 8E 5E Ø8 38 DF
[EU5] 1Ø33: 56 EC DA AE D9 73 79 CC 4Ø 58 41 AF B4 42 9Ø 45
[WF5] 1Ø34: 81 3E Ø8 13 E2 Ø4 1C FE 21 3Ø 65 ØC 32 C5 3C 74
[FM5] 1Ø35: BD 77 ØC ØØ 33 4B A9 Ø3 A7 DØ 14 EB AB E1 48 BØ
[6S5] 1Ø36: 9D B1 94 ØF 72 1D 9Ø CØ 9E 73 E8 C5 Ø1 A5 7D 15
[4U5] 1Ø37: 4E 5C 1F AB 2B 2Ø DF A8 F1 C1 3F AD BØ 2Ø D5 8E
[BG5] 1Ø38: 71 45 9A 46 BØ E0 7E 91 51 78 41 DE 22 D9 26 44
[2H5] 1Ø39: ED 9Ø BA 84 12 32 91 58 FØ D3 3A 3Ø 3F 61 6Ø 53
[FV5] 1Ø4Ø: D4 3D BA 8Ø E0 3A E3 1E 59 2Ø 84 1A CF 35 7A FC
[XS5] 1Ø41: E8 91 C3 ØB D9 B3 Ø5 AD C4 13 E0 98 7B F6 A4 ØØ
[YX5] 1Ø42: E0 1C 62 CF 9F 3D 6F F6 9C 7Ø C3 EC 51 C2 ØA BA
[WN5] 1Ø43: C7 ØA 18 6A 96 D5 Ø7 AC 4Ø 47 Ø7 FC E1 4C 7B Ø7
[9P5] 1Ø44: 54 F7 B8 3D 4Ø Ø1 A9 F3 26 12 95 B8 3B BD 59 95
[HG5] 1Ø45: CF 49 25 84 BØ 83 34 52 12 C3 A3 64 49 55 67 B5
[TS5] 1Ø46: 1F A0 B5 9E B1 Ø1 7C CD 85 DD E1 74 91 16 79 94
[ØL5] 1Ø47: AB 3Ø Ø7 F1 82 12 68 A9 16 86 95 F6 C2 A7 1E A1
[HL5] 1Ø48: ØD D9 16 DØ 95 2E 5C 19 1Ø A9 85 61 55 BD BØ A8
[NK5] 1Ø49: 87 Ø9 6Ø E0 1B 81 B5 6Ø E6 E3 96 B3 36 FF 75 Ø2
[TW5] 1Ø5Ø: 3B 85 3D 57 83 3D D0 17 CA BC 83 7B D2 D9 Ø7 3A
[9N5] 1Ø51: 1C B8 DØ A2 E3 81 Ø9 Ø7 1C 98 98 Ø8 C3 DB Ø3 BA
[PR5] 1Ø52: F6 81 1F 51 ØA 95 3B E8 CC 4B Ø4 CF B3 61 8Ø 9C
[9Z5] 1Ø53: 3C 41 A1 CD 2C D8 A2 C4 3D ØE DC BØ 44 EB CB 6A
[RM5] 1Ø54: 1Ø 67 ØØ C1 17 EC 3E 4A 6F 8F 38 Ø6 CB F2 DC 62
[KH5] 1Ø55: CD D4 A3 82 Ø9 62 92 6A Ø7 42 1D Ø8 79 ØØ 72 49
[2Y5] 1Ø56: 3B 8D E1 EA A5 73 F7 32 BA 1F FØ 91 E1 FØ FD F1
[2L5] 1Ø57: 1F 92 9C 57 91 A1 9D A2 85 D4 C4 85 CA 1Ø 85 C4
[ZH5] 1Ø58: B8 52 ØB 19 FØ 38 4A 77 8B CØ 32 A2 52 DC 7E 5Ø
[8M5] 1Ø59: A0 86 7A C5 39 D2 F8 Ø8 ØØ 73 E8 83 2A Ø9 ØC F5
[NK5] 1Ø6Ø: CB 21 7D 3Ø 15 75 CØ 9B 1A A5 24 46 39 D8 ØF 86
[RF5] 1Ø61: Ø7 AD 41 8Ø D2 A5 16 11 7A 7C 1B 1Ø 4E 57 Ø7 6C
[AP5] 1Ø62: BØ 67 EB 32 2F F8 BD CØ E7 3F ØD ØØ AB C9 ØF 74
[CW5] 1Ø63: AA 91 AD 4F C7 3E 7C FØ D7 CC ED ØD Ø7 2A ØC C1
[DN5] 1Ø64: 61 37 18 92 98 93 F7 4Ø 6Ø 1E B6 CE 7B 4E 42 E4 Ø8
[7P5] 1Ø65: DC 3Ø 23 44 ØB A3 37 44 E8 1Ø 1E 3F Ø8 51 22 C4
[5N5] 1Ø66: 8A 1Ø 1A E6 1B 5Ø 7B Ø8 DD 72 DA 3B F1 5Ø 2E 43
[WT5] 1Ø67: C6 64 CØ 5A FØ 56 Ø7 EE 1C 9Ø BB ØD 4B E3 7E 47
[4K5] 1Ø68: 46 FB D4 Ø7 67 3C 21 4Ø 83 1E 34 E4 4A 38 64 87
[PS5] 1Ø69: 37 CB 86 1B E6 75 FA 43 6D 36 AC C5 CØ 5B 3C DØ
[BR5] 1Ø7Ø: AC 15 Ø3 CA E4 5Ø E6 1C BB 35 DC D5 Ø7 35 ØD 13
[ZL5] 1Ø71: A2 2F 6F BC B4 76 9F 1Ø 39 32 9C Ø9 5D 2F 9Ø
[NU5] 1Ø72: 47 ØE E3 4A FB A5 F5 97 7Ø 71 2Ø BØ CF Ø6 F9 F1
[HM5] 1Ø73: A5 F6 97 94 71 3E 77 59 F6 A5 F7 97 55 71 26 C7
[AN5] 1Ø74: 29 9D 65 F7 97 1A 5F 8C ØB Ø3 1E Ø3 E9 Ø4 69 4C
[2U5] 1Ø75: CØ ØF A9 63 2Ø E2 99 ØF 8A E5 B1 6F FB C1 A4 95
[ZP5] 1Ø76: C7 99 A6 BØ E0 4F D2 2Ø 4F 31 D4 47 2D 1C 65 84
[NL5] 1Ø77: 6Ø 8A 53 29 68 DØ FE 6Ø 78 E4 Ø4 3A FB ØD 67 66
[7R5] 1Ø78: 3E DC B9 58 58 9F 64 C1 74 C4 86 E8 7B 8E 51 14
[7P5] 1Ø79: DC 3Ø 2C 6Ø E7 73 8C 5E A1 Ø2 5B 2E 71 8C 66
[PF5] 1Ø8Ø: F1 57 1F CØ 01 3D Ø4 DE 43 49 ØE 21 CØ 76 39 94
[6P5] 1Ø81: E6 5Ø AD 43 A6 29 41 4D 75 12 F8 A3 D4 E9 F2 21
[PN5] 1Ø82: 4D 9D 3A 75 FE D5 F9 5Ø C7 D9 4A 93 24 45 12 C7
[YP5] 1Ø83: 72 3A D5 29 1E 16 ØB D5 5F 3E C4 19 72 Ø7 67 7D
[AE5] 1Ø84: B3 38 28 Ø6 62 81 24 27 4E Ø5 C0 8D 4Ø 11 1E 13
[YR5] 1Ø85: 4F A5 ØE 8Ø B5 1A ØE 5F 45 Ø6 2D 2E D4 66 DØ EC
[PQ5] 1Ø86: 42 2Ø ØØ 7D ØØ F1 12 ØØ ØØ 47 47 53 5Ø 52 49 54
[YB5] 1Ø87: 7Ø BA 5Ø 4B Ø1 Ø2 ØB ØØ ØA ØØ ØØ ØØ Ø6 ØA 6A 7F
[EE5] 1Ø88: DD 18 13 AØ 9C 54 12 Ø7 ØØ ØØ BC ØF ØØ ØØ ØB ØØ
[815] 1Ø89: ØØ ØØ ØØ ØØ ØØ ØØ 2Ø ØØ ØØ ØØ 76 ØB ØØ ØØ
[ØL5] 1Ø9Ø: 47 47 4C 4F 47 4F 53 2E 4F 42 4A 5Ø 4B Ø1 Ø2 ØB
[WF5] 1Ø91: ØØØA ØØ ØØ ØØ Ø6 ØØ 6A 7F DD 18 B2 36 96 E1 D1
[4A5] 1Ø92: 1E ØØ ØØ ED 3F ØØ ØØ ØC ØØ ØØ ØØ ØØ ØØ ØØ ØØ
[B45] 1Ø93: ØØ 2Ø ØØ ØØ ØØ F1 12 ØØ ØØ 47 47 53 5Ø 52 49 54
[V95] 1Ø94: 31 2E 4F 42 4A 5Ø 4B Ø1 Ø2 ØB ØØ ØA ØØ ØØ Ø6
[XG5] 1Ø95: ØØ 6A 7F DD 18 5A 61 CB A6 ØE 12 ØØ ØØ 17 15 ØØ
[745] 1Ø96: ØØ ØA ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ 2Ø ØØ ØØ ØØ AC
[BC5] 1Ø97: 31 ØØ ØØ 47 47 52 41 55 4D 2E 4F 42 4A 5Ø 4B Ø5
[535] 1Ø98: Ø6 ØØ ØØ ØØ ØØ Ø3 ØØ Ø3 ØØ AB ØØ ØØ ØØ E2 43 ØØ
[GUØ] 1Ø99: ØØ ØØ ØØ
```

Listing 14. Die Grafiken sind in »ggobj.exe« in komprimierter Form enthalten.

```
[Ø53]    1: ; Programm: u01.pic
[EB6]    2: ; Funktion: Tabelle zum Sprite 'Radar'
[NZ6]    3: ;           zur Einbindung in ggsprit2.asm
[913]    4: ; Sprache:  TASM 2.01
[VT3]    5: ; Autor:    Detlef Paulin
[7A4]    6: ; (c) 1992 Widuch GmbH & Co. KG
[ØØØ]    7:
[ZT3]    8: ; Aufbau der Kopfzeile:
[YV5]    9: ; dw 2Ø -> bei Treffer gibt es 2Ø Punkte
[3E5]   1Ø: ;           (dw 256+x -> x Punkte Minus)
[XG4]   11: ; dw  Ø -> Explosionsart Ø (Ø-4)
[XG5]   12: ; dw  Ø -> Ø+1 Treffer sind notwendig,
[G25]   13: ;          bevor Sprite explodiert
[Y65]   14: ; dw  Ø -> bei Crash gibt es Ø Punkte
[365]   15: ;          (dw 256+x -> x Punkte Minus)
[6Ø6]   16: ; dw 256+1Ø->bei Crash werden 1Ø Energie-
[NY3]   17: ;          punkte abgezogen
[FE4]   18: ;          (dw x -> x Punkte Plus)
[DX5]   19: ; dw  Ø -> bei Crash gibt es Ø Munitions-
[F52]   2Ø: ;          zuwachs
```

```
[ØØØ]   21:
[PT2]   22:          dw      2Ø,Ø,Ø,Ø,256+1Ø,Ø
[ØØØ]   23:
[TZ4]   24: ; eigentliche Spritecodierung
[K93]   25: ; eine Phase=12*3 Word
[2K4]   26: ; u01 besteht aus zwei Phasen
[ØØØ]   27:
[FØ7]   28: ; die ersten drei Word entsprechen 24 Pixel
[EM4]   29: ; mit folgenden Farbwerten:
[H53]   3Ø: ; ØØØØØH -> Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø
[Ø93]   31: ; ØAAØ8H -> Ø,Ø,2,Ø,2,2,2,2
[Z53]   32: ; ØØØ2ØH -> Ø,2,Ø,Ø,Ø,Ø,Ø,Ø
[U
```

```
  hex:=h[bl*4+b2];
end;
begin
 escape:=false;
 Size:=ImageSize(1Ø0,1Ø,319,7Ø);
 GetMem(hilfep,Size);
 GetImage(1Ø0,1Ø,319,7Ø,hilfep^);
 f:=GetColor; SetColor(2);
 Rectangle(1Ø0,1Ø,319,7Ø);
 SetViewPort(1Ø1,11,318,69,Clipoff);
 ClearViewPort;
 SetViewport(Ø,Ø,319,199,ClipOff);
 OutTextXY(1Ø8,12,
     'Fenster als ASM-Quelltext');
 if ((xu-xo+1) MOD 8)<>Ø then wb:='B' else
  begin
  OutTextXY(1Ø8,22,'(B)yte oder (W)ord?');
  repeat Read_n_Char(wb,1,29Ø,22);
  until (wb='b') or (wb='B') or (wb='w')
                 or (wb='W') or escape
  end;
 if not escape then begin
 if (wb='b') or (wb='B') then
      OutTextXY(1Ø8,32,' ==> BYTE-weise')
 else OutTextXY(1Ø8,32,' ==> WORD-weise');
 OutTextXY(1Ø8,52,
                'Dateiname:          .asm');
 Read_n_char(dateiname,8,196,52);
 if not escape then begin
 assign(quelle,dateiname+'.asm');
 Rewrite(quelle);
 writeln(quelle,dateiname);
 for i:=yo to yu do
 begin
  p:=chr(9)+'dw'+chr(9); pk:='Ø'; j:=xo;
  if (wb='w') or (wb='W') then
  repeat
    p:=p+pk+hex(pict[j+4,i],pict[j+5,i])
       +hex(pict[j+6,i],pict[j+7,i])
       +hex(pict[j,i],pict[j+1,i])
       +hex(pict[j+2,i],pict[j+3,i])+'H';
    pk:=',Ø';j:=j+8;
  until j>=xu
  else
   repeat
    p:=p+pk+hex(pict[j,i],pict[j+1,i])
        +hex(pict[j+2,i],pict[j+3,i])+'H';
    pk:=',Ø';j:=j+4;
   until j>=xu;
   writeln(quelle,p);
  end;
  writeln(quelle,chr(26));
  close(quelle);
 end;
 end;
 PutImage(1Ø0,1Ø,hilfep^,Normalput);
 FreeMem(hilfep,size); SetColor(f);
end;

procedure SaveBs;
var size,f,i,j:integer;
begin
 escape:=false;
 Size:=ImageSize(1Ø0,1Ø,319,7Ø);
 GetMem(hilfep,Size);
 GetImage(1Ø0,1Ø,319,7Ø,hilfep^);
 f:=GetColor; SetColor(2);
 Rectangle(1Ø0,1Ø,319,7Ø);
 SetViewPort(1Ø1,11,318,69,Clipoff);
 ClearViewPort;
 SetViewport(Ø,Ø,319,199,ClipOff);
 OutTextXY(1Ø8,12,
  'Komplettes Bild retten');
 OutTextXY(1Ø8,52,
  'Dateiname:          .zfb');
 Read_n_char(dateiname,8,196,52);
 if not escape then begin
 assign(zfb,dateiname+'.zfb');
 Rewrite(zfb);
 for i:=Ø to 79 do for j:=Ø to 47 do
          pzfb[i,j]:=byte(pict[i,j]);
 write(zfb,pzfb); close(zfb);
end;
 PutImage(1Ø0,1Ø,hilfep^,Normalput);
 FreeMem(hilfep,size); SetColor(f);
end;

procedure ZeichnePkt(spalte,zeile,
                     farbe:integer);
var i,j:integer;
begin
 i:=spalte*4; j:=zeile*4;
 SetColor(farbe);
 Rectangle(i,j,i+3,j+3);
 Rectangle(i+1,j+1,i+2,j+2);
 if farbe=Ø then PutPixel(i+2,j+2,1);
end;

procedure ZeichneBild;
const
 fbyte: array[Ø..3] of byte=(Ø,85,17Ø,255);
var i,j,k,f,c: integer;
begin
 for i:=Ø to 47 do for j:=Ø to 79 do
  begin
   k:=i+j; c:=fbyte[pict[j,i]];
   halbbild1[k,j]:=c; halbbild1[k+1,j]:=c;
   halbbild2[k,j]:=c; halbbild2[k+1,j]:=c;
   if c=Ø then PutPixel(4*j+2,4*i+2,1);
  end;
end;

procedure ZeichneKreuz;
begin
 GetImage(xpos*4,ypos*4,xpos*4+3,
                        ypos*4+3,kreuzp);
 PutImage(xpos*4,ypos*4,kreuz,XorPut);
end;

procedure LoescheKreuz;
begin
 PutImage(xpos*4,ypos*4,kreuzp,NormalPut);
 if zeichnenflag then
  begin ZeichnePkt(xpos,ypos,aktf);
        pict[xpos,ypos]:=aktf; end;
end;

procedure OriginalBild(a,b,c,d:byte);
var size,i,j,f: integer;
    fpa: array[Ø..3] of byte;
begin
 fpa[Ø]:=a; fpa[1]:=b;fpa[2]:=c;fpa[3]:=d;
 Size:=ImageSize(238,1Ø0,319,149);
 GetMem(hilfep,Size);
 GetImage(238,1Ø0,319,149,hilfep^);
 f:=GetColor; SetColor(1);
 Rectangle(238,1Ø0,319,149);
 SetViewPort(239,1Ø1,318,148,ClipOff);
 ClearViewPort;
 SetViewPort(Ø,Ø,319,199,ClipOff);
 for i:=xo to xu do for j:=yo to yu do
     PutPixel(239+i,1Ø1+j,fpa[pict[i,j]]);
 WarteAufTaste;tastl:=' ';
 PutImage(238,1Ø0,hilfep^,NormalPut);
 FreeMem(hilfep,Size); SetColor(f);
end;

procedure Status(nr:integer);
var f:word; xs,ys,fs:string[2];
begin
 f:=GetColor;
 case nr of
 Ø : begin str(xpos-xo:2,xs);
      str(ypos-yo:2,ys); str(aktf:2,fs);
      SetColor(2);
      OutTextXY(Ø,192,
```

```
   'F1: Hilfe X='+xs+' Y='+ys+' Farbe='+fs);
     SetColor(aktf);
     OutTextXY(232,192,'  '); end;
 1 : begin str(xpos-xo:2,xs);SetColor(Ø);
     OutTextXY(96,192,'  '); SetColor(2);
     OutTextXY(96,192,xs); end;
 2 : begin str(ypos-yo:2,ys); SetColor(Ø);
     OutTextXY(136,192,'  '); SetColor(2);
     OutTextXY(136,192,ys); end;
 3 : begin str(aktf:2,fs); SetColor(Ø);
     OutTextXY(2Ø8,192,'  '); SetColor(2);
     OutTextXY(2Ø8,192,fs); SetColor(aktf);
     OutTextXY(232,192,'  '); end;
 4 : PutImage(26Ø,192,pinsel,XorPut)
end;
SetColor(f);
end;

procedure Hilfe;
const m: array [1..19] of string[35]
  =('              Tastenbelegung          ',
    'Kursortasten - Positionsveränderung',
    'Ø,1,2,3      - Farbwert auswählen    ',
    'Enter        - Farbwert setzen       ',
    'Z            - Pinsel ein/aus        ',
    'ESC          - Programm beenden      ',
    'F2           - Fenster kopieren      ',
    '     1. verschieben 2. F2 best&tigt',
    'F3           - Fensterrand an/aus    ',
    'F4           - Fenster festlegen     ',
    '     1. Linke  obere  Ecke, dann F4',
    '     2. Rechte untere Ecke, dann F4',
    'F5           - Fensterinhalt als     ',
    '               ASM-Quelltext retten',
    'F6           - Gesamtgrafik retten   ',
    'F7           - Gesamtgrafik laden    ',
    'F8           - Originalbild (CGA)    ',
    'F9           - Originalbild mono-    ',
    '               chromatisch zeigen    ');
var size,f:word;
    i:integer;
begin
 Size:=ImageSize(1Ø,1Ø,3Ø9,189);
 GetMem(hilfep,Size);
 GetImage(1Ø,1Ø,3Ø9,189,hilfep^);
 f:=GetColor; SetColor(2);
 Rectangle(1Ø,1Ø,3Ø9,189);
 SetViewPort(11,11,3Ø8,188,ClipOff);
 ClearViewPort;
 SetViewPort(Ø,Ø,319,199,ClipOff);
 for i:=1 to 19 do
      OutTextXY(12,i*9+3,' '+m[i]+' ');
 WarteAufTaste; tastl:=' ';
 PutImage(1Ø,1Ø,hilfep^,NormalPut);
 FreeMem(hilfep,size); SetColor(f);
end;

procedure FensterRand(a,b,c,d:integer);
var i,f: integer;
begin
 f:=GetColor;
 ZeichnePkt(xpos,ypos,pict[xpos,ypos]);
 for i:=a to c do
  begin ZeichnePkt(i,b,pict[i,b]);
        ZeichnePkt(i,d,pict[i,d]); end;
 for i:=b to d do
 begin ZeichnePkt(a,i,pict[a,i]);
       ZeichnePkt(c,i,pict[c,i]); end;
 ZeichneKreuz;SetColor(f);
end;

procedure DefFenster;
var f:integer; ch: char; zf: boolean;
begin
 if fenster then FensterRand(xo,yo,xu,yu);
 ZeichnePkt(xpos,ypos,pict[xpos,ypos]);
 f:=GetColor; SetColor(3);
 SetWriteMode(XorPut);
 xow:=xpos; yow:=ypos;xuw:=xpos;yuw:=ypos;
 Rectangle(xow*4,yow*4,xuw*4+3,yuw*4+3);
 repeat
  repeat until KeyPressed;
  Rectangle(xow*4,yow*4,xow*4+3,yow*4+3);
  ch:=ReadKey;
  if ch=#Ø then ch:=ReadKey;
  if (ch=#75) and (xow>Ø) then dec(xow);
  if (ch=#77) and (xow<76) then inc(xow);
  if (ch=#72) and (yow>Ø) then dec(yow);
  if (ch=#8Ø) and (yow<47) then inc(yow);
  if (ch<>#27) and (ch<>'>') then
  Rectangle(xow*4,yow*4,xow*4+3,yow*4+3);
 until (ch=#27) or (ch='>');
 if ch='>' then begin
 xuw:=xow+3;yuw:=yow+1;
 Rectangle(xow*4,yow*4,xuw*4+3,yuw*4+3);
 repeat
  repeat until KeyPressed;
  Rectangle(xow*4,yow*4,xuw*4+3,yuw*4+3);
  ch:=ReadKey;
  if ch=#Ø then ch:=ReadKey;
  if (ch=#75) and (xuw>xow+3) then
                                 xuw:=xuw-4;
  if (ch=#77) and (xuw<76) then xuw:=xuw+4;
  if (ch=#72) and (yuw>yow+1) then
                                 dec(yuw);
  if (ch=#8Ø) and (yuw<47) then inc(yuw);
  Rectangle(xow*4,yow*4,xuw*4+3,yuw*4+3);
 until (ch=#27) or (ch='>');
 Rectangle(xow*4,yow*4,xuw*4+3,yuw*4+3);
 if ch<>#27 then begin xpos:=xow;ypos:=yow;
  xo:=xow; yo:=yow; xu:=xuw; yu:=yuw;
  fenster:=true; end;
 end;
 SetColor(f);SetWriteMode(CopyPut);
 ZeichneKreuz; Status(1);Status(2);
end;

procedure CopyFenster;
var f,xp,yp,xd,yd,i,j,id,jd:integer;
    ch: char;
    zf: boolean;
    tpic: array[Ø..79,Ø..47] of integer;
begin
 ZeichnePkt(xpos,ypos,pict[xpos,ypos]);
 f:=GetColor; SetColor(3);
 SetWriteMode(XorPut);
 xp:=xo; yp:=yo; xd:=xu-xo; yd:=yu-yo;
 if not fenster then Rectangle(xp*4,yp*4,
                        (xp+xd)*4+3,(yp+yd)*4+3);
 repeat
  repeat until KeyPressed;
  if (xp<>xo) or (yp<>yo) then Rectangle(
        xp*4,yp*4,(xp+xd)*4+3,(yp+yd)*4+3);
  ch:=ReadKey;
  if ch=#Ø then ch:=ReadKey;
  if (ch=#75) and (xp>Ø) then dec(xp);
  if (ch=#77) and (xp+xd<79) then inc(xp);
  if (ch=#72) and (yp>Ø) then dec(yp);
  if (ch=#8Ø) and (yp+yd<47) then inc(yp);
  if (ch<>#27) then Rectangle(xp*4,yp*4,
                        (xp+xd)*4+3,(yp+yd)*4+3);
 until (ch=#27) or (ch='<');
 if ch='<' then begin
  id:=xp-xo;jd:=yp-yo;
  for i:=xo to xu do for j:=yo to yu do
                     tpic[i+id,j+jd]:=pict[i,j];
  for i:=xp to xp+xd do for j:=yp to yp+yd
                     do pict[i,j]:=tpic[i,j];
 end;
 SetColor(f);SetWriteMode(CopyPut);
 ZeichneKreuz; Status(1);Status(2);
end;

procedure LoadBs;
var size,f,i,j:integer;
```

```
    fehler: boolean;
begin
 fehler:=true;
 escape:=false;
 Size:=ImageSize(1Ø0,1Ø,319,7Ø);
 GetMem(hilfep,Size);
 GetImage(1Ø0,1Ø,319,7Ø,hilfep^);
 f:=GetColor; SetColor(2);
 Rectangle(1Ø0,1Ø,319,7Ø);
 SetViewPort(1Ø1,11,318,69,Clipoff);
 ClearViewPort;
 SetViewport(Ø,Ø,319,199,ClipOff);
 OutTextXY(1Ø8,12,'Komplettes Bild laden');
 OutTextXY(1Ø8,25,
 'Altes Bild wird gelöscht!');
 OutTextXY(1Ø8,52,
 'Dateiname:          .zfb');
 Read_n_char(dateiname,8,196,52);
 if not escape then begin
 assign(zfb,dateiname+'.zfb');
 {$I-} Reset(zfb); {$I+}
 if IOResult=Ø then fehler:=false;
 if not fehler then
 begin
 Read(zfb,pzfb);
 for i:=Ø to 79 do for j:=Ø to 47 do
          pict[i,j]:=integer(pzfb[i,j]);
 close(zfb);
 end
 else begin OutTextXY(1Ø8,6Ø,
              'Datei existiert nicht !');
      WarteAufTaste; tastl:=' ';
      end;
end;
 PutImage(1Ø0,1Ø,hilfep^,Normalput);
 FreeMem(hilfep,size);
 SetColor(f);
end;

procedure Beenden;
var size,f:integer;
begin
 escape:=false;
 Size:=ImageSize(6Ø,9Ø,26Ø,1Ø2);
 GetMem(hilfep,Size);
 GetImage(6Ø,9Ø,26Ø,1Ø2,hilfep^);
 f:=GetColor; SetColor(3);
 Rectangle(6Ø,9Ø,26Ø,1Ø2);
 SetViewPort(61,91,259,1Ø1,Clipoff);
 ClearViewPort;
 SetViewport(Ø,Ø,319,199,ClipOff);
 OutTextXY(76,93,'Programm beenden (J/N)');
 WarteAufTaste;
 if (tastl='J') or (tastl='j') then
                 tastl:=#27 else tastl:=' ';
 PutImage(6Ø,9Ø,hilfep^,Normalput);
 FreeMem(hilfep,size); SetColor(f);
end;

procedure action;
var ch:char; f,code:integer;
begin
 if KeyPressed then
 begin
  ch:=ReadKey; if ch=#Ø then ch:=ReadKey;
 end;
 case ch of
  'Ø'..'3' : begin val(ch,aktf,code);
                   Status(3);end;
  'Z','z'  : begin
          zeichnenflag:= not zeichnenflag;
             Status(4);end;
  'K'      : begin LoescheKreuz;
             if xpos>xo then dec(xpos) else xpos:=xu;
              ZeichneKreuz;Status(1); end;
  'M'      : begin LoescheKreuz;
             if xpos<xu then inc(xpos) else xpos:=xo;
              ZeichneKreuz;Status(1); end;
  'H'      : begin LoescheKreuz;
  if ypos>yo then dec(ypos) else ypos:=yu;
              ZeichneKreuz;Status(2);end;
  'P'      : begin LoescheKreuz;
  if ypos<yu then inc(ypos) else ypos:=yo;
              ZeichneKreuz;Status(2);end;
  #13      : begin LoescheKreuz;
              ZeichnePkt(xpos,ypos,aktf);
              pict[xpos,ypos]:=aktf;
              ZeichneKreuz; end;
  ';'      : hilfe;
  '<'      : CopyFenster;
  'C'      : OriginalBild(Ø,3,3,3);
  'B'      : OriginalBild(Ø,1,2,3);
  '='      : begin fenster:= not fenster;
              if not fenster then
              FensterRand(xo,yo,xu,yu);
             end;
  '>'      : if (xpos<76) and (ypos<47)
                          then DefFenster;
  '?'      : SaveQuelle;
  '@'      : SaveBs;
  'A'      : LoadBs;
  #27      : Beenden;
 end;
end;

begin
 Init;
 ZeichneBild;
 ZeichneKreuz;
 Status(Ø);
 repeat
 action;
 until tastl=#27;
end.
```

Listing 16. »Zeidef.pas« ist ein einfaches Malprogramm für Sprites.

```
1: tasm ggr
2: tasm gghaupt
3: tasm ggherc
4: tasm ggcga
5: tasm ggvga
6: tasm ggzsatz
7: tasm ggsprit2
8:
9: tlink ggr+gghaupt+ggherc+ggcga+ggvga+ggzsat
   z+ggsprit2+ggsprit1+gglogos+ggraum,gg3ØØØ
```

Listing 17. »Go.bat« assembliert und linkt die verschiedenen Programmteile.

Tetris-Clone

Klötzchenweise

Es ist noch nicht lange her, da war das Spiel Tetris der Hit unter den Geschicklichkeitsspielen. Das Anordnen herabfallender Klötzchen hat eine magische Anziehungskraft auf den Spieltrieb. Dieser Tetris-Clone verfügt zusätzlich über einen Zwei-Spieler-Modus, mit dem Sie die Klötzchen Ihres Gegenspielers zu bestimmten Zeiten manipulieren können, so daß Sie im weiteren Spielverlauf bald mehr auf die Aktionen Ihres Mitspielers achten werden als auf Ihre eigenen.

Ziel des Spiels Tetris ist es, die herabfallenden Teilchen optimal aneinanderzufügen, so daß keine Löcher entstehen. Wird eine komplette Reihe gebildet, löst sich diese auf, und der obere Teil rückt eine Reihe nach unten. Das Spiel ist zu Ende, wenn das Spielfeld bis oben aufgefüllt ist. Damit es aber nicht ganz so einfach ist, nimmt die Fallgeschwindigkeit mit zunehmender Zahl der gebildeten Reihen zu.

- **Start von der MS-DOS-Ebene:** Sie starten Tetris von MS-DOS aus mit dem Aufruf »tvtetris.exe«. Es erscheint sogleich ein Info-Fenster. Haben Sie eine Maus installiert, so können Sie diese zur Spielsteuerung auf einer CUA-Oberfläche einsetzen. Um das Info-Fenster zu schließen, drücken Sie die Enter-Taste.

Die Bedienung im Ein-Spieler-Modus (Bilder 1 und 2) entnehmen Sie Tabelle 1, eine Beschreibung der Funktionstasten finden Sie in Tabelle 2. Die herabfallenden Teilchen steuern Sie mit den Tasten aus Tabelle 3 und 4.

- **Der hinzugekommene Zwei-Spieler-Modus:** In diesem Modus (Bilder 1 und 3) können die Spieler gegenseitig ihre Teilchen manipulieren: Nach verschiedenen Zeitabständen erscheint am Spielfeldrand die Meldung »Versenke eine Reihe«. Können Sie in diesem Moment eine volle Reihe bilden, so erhalten Sie fünf Bonuspunkte (Credits). Diese Credits können Sie nun gegen den Mitspieler einsetzen. Dazu verwenden Sie die Tasten aus den Tabellen 5 und 6.

Im Zwei-Spieler-Modus macht der Einsatz der Maus keinen großen Sinn, da ja nur Spieler 1 sie zum Steuern der Teilchen verwenden kann.

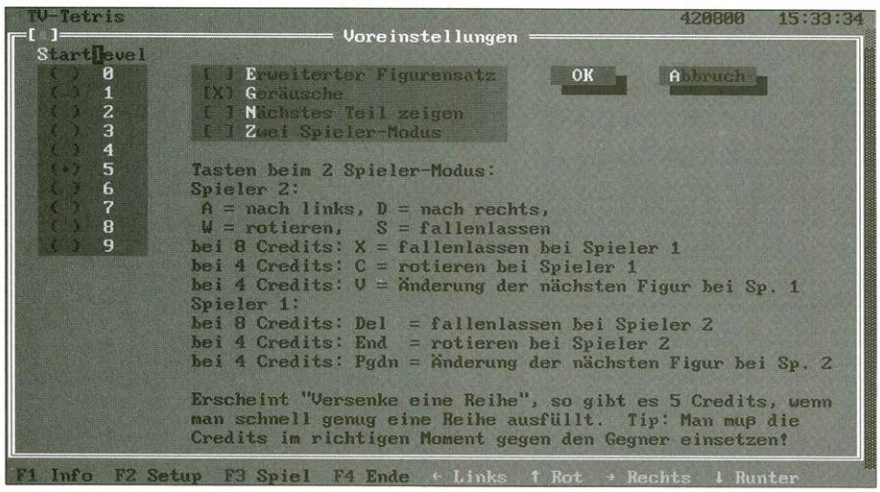


Bild 1. Die verschiedenen Einstellungen

Taste	Funktion
F1	Information
F2	Voreinstellung (Setup)
F3	Spiel
F4	Ende

Tabelle 1. Mit diesen Tasten bedienen Sie Tetris.

Das Einstellungsfenster (Bild 1) erlaubt die Wahl verschiedener Zustände des Spiels, zeigt die Tastaturbelegung im Zwei-Spieler-Modus und weist darauf hin, bei welchen Credits Sie die Klötzchen Ihres Mitspielers steuern und mit welchen Tasten Sie dies tun.

Tetris verfügt über zehn Schwierigkeitsgrade, die sich in der Geschwindigkeit der herabfallenden Teile unterscheiden. Außerdem können Sie einen erweiterten Figurensatz einstellen und Spielgeräusche ein- oder ausschalten. Mit der Option »Nächstes Teil anzeigen« blenden Sie ein kleines Fenster ein, in dem das Teilchen, das als nächstes hinabfällt, eingeblendet wird.

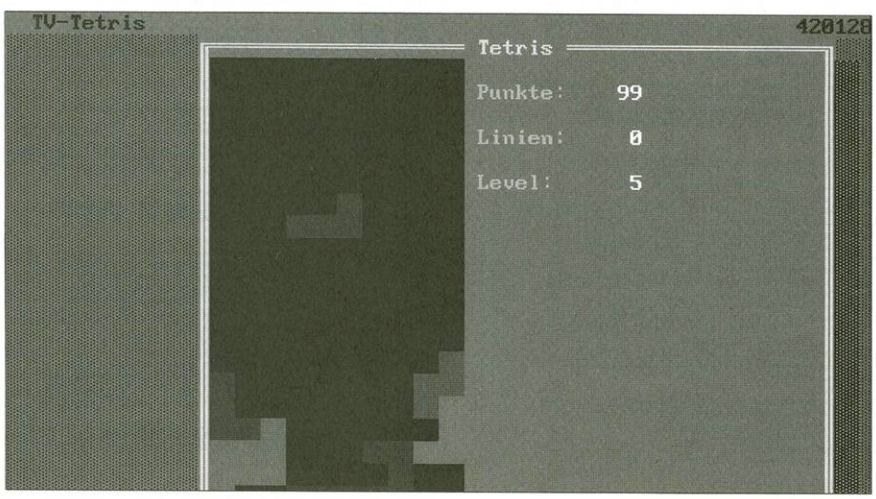


Bild 2. Der Ein-Spieler-Modus des Tetris-Clones

Taste	Bedeutung
F1	Hier erhalten Sie einen Überblick über das Programm, den Autor und anderes.
F2	führt Sie ins Setup. Dort haben Sie die Gelegenheit, verschiedene Spieloptionen auszuwählen: ● **Startlevel:** bestimmt den Schwierigkeitsgrad des Spiels. Zur Auswahl reicht es, die entsprechende Zifferntaste zu drücken. ● **Erweiterter Figurensatz:** Drücken der Taste [E] aktiviert den erweiterten Figurensatz. Statt der üblichen erscheinen nun auch komplexere Figuren im Spiel, die schwieriger zu handhaben sind. ● **Geräusche:** Die Taste [G] schaltet die Geräusche ein oder aus. ● **Nächstes Teil:** Ist diese Option mit der Taste [N] aktiviert, so wird während des Spiels schon das nächste herunterfallende Teilchen angezeigt. Diese Einrichtung ist sehr hilfreich, da Sie so eine Taktik aufstellen können. ● **Zwei-Spieler-Modus:** Mit der Taste [Z] stellen Sie den Zwei-Spieler-Modus ein. Ist die Option markiert, so spielen zwei Personen gegeneinander. Haben Sie alles eingestellt, wird das Setup mit der Enter-Taste verlassen. Drücken Sie dagegen [Esc], so werden die Einstellungen nicht übernommen.
F3	Damit starten Sie das Spiel. Per [Esc] können Sie das Spiel jederzeit abbrechen und gelangen wieder ins Hauptmenü zurück. Geben Sie am Ende des Spiels Ihren Namen ein, wird er mit der entsprechenden Punktzahl in die Highscore-Tabelle eingetragen. Die Tabelle wird auf Diskette beziehungsweise Festplatte gespeichert und geht auch nach Beendigung des Spiels nicht verloren.
F4	Hiermit beenden Sie Tetris.

Tabelle 2. Die Funktionen der verschiedenen Tasten

Taste	Aktion
[Cursor ab]	Teilchen fallenlassen
[Cursor auf]	Teilchen rotieren
[Cursor links]	Teilchen nach links verschieben
[Cursor rechts]	Teilchen nach rechts verschieben

Tabelle 3. Während des Spiels steuern Sie das herabfallende Teilchen mit den Cursortasten.

Taste	Aktion
S	Teilchen fallenlassen
W	Teilchen rotieren
A	Teilchen nach links verschieben
D	Teilchen nach rechts verschieben

Tabelle 4. Im Zwei-Spieler-Modus steuert der zweite Spieler mit diesen Tasten.

Noch ein Tip zum Schluß: Die Credits müssen gezielt gegen den Gegner eingesetzt werden. Vorsicht – besonders der Zwei-Spieler-Modus macht süchtig!

Dieser Tetris-Clone ist vollständig objektorientiert in Borlands Turbo Vision programmiert und kann deshalb als lehrreiches Beispiel für den Einsatz dieser Technik herangezogen werden. Auf diese Weise können Sie sowohl die objektorientierte Programmiertechnik in Pascal als auch den Einsatz der Klassenbibliothek Turbo Vision an zahlreichen Objekten detailliert studieren.

(Hartmut Günther/bm)

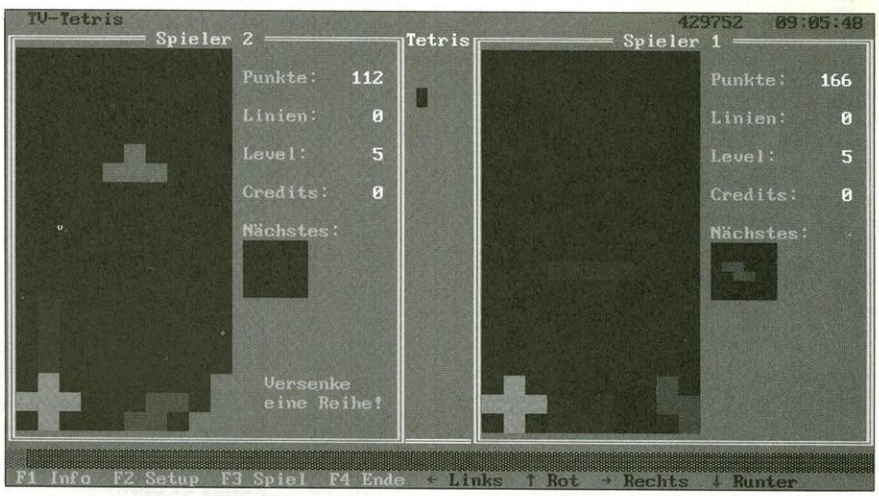


Bild 3. So sieht der Zwei-Spieler-Modus aus.

Anwendungen gesucht

Haben Sie eine schnelle und komfortable Textverarbeitung geschrieben? Oder eine Datenbank mit frei wählbaren Eingabemasken? Eventuell ein Zeichenprogramm? Wenn Ihr Programm mit professionellen Programmen konkurrieren kann, ist es mit Sicherheit zu umfangreich, um als Listing in der DOS International zu erscheinen. Zögern Sie deshalb nicht, schicken Sie uns unverbindlich die aktuelle Version Ihres Programms und einen Ausdruck der Dokumentation. Eventuell wird Ihr Programm der Verkaufsschlager im Rahmen der neuen DOS-Software-Schiene.

Schicken Sie Ihre Unterlagen an folgende Adresse:

Redaktion
DOS International,
Gruberstraße 46c
8011 Poing bei München

Bonuspunkte	Taste	Aktion
mit 8 Credits	[Entf]	Teilchen von Spieler 2 fallenlassen
mit 4 Credits	[Ende]	Teilchen von Spieler 2 rotieren
mit 4 Credits	[Bild ab]	Änderung der nächsten Figur von Spieler 2

Tabelle 5. Die Bonus-Tastensteuerung für Spieler 1

Bonuspunkte	Taste	Aktion
mit 8 Credits	[X]	Teilchen von Spieler 1 fallenlassen
mit 4 Credits	[C]	Teilchen von Spieler 1 rotieren
mit 4 Credits	[V]	Änderung der nächsten Figur von Spieler 1

Tabelle 6. Die Bonus-Tastensteuerung für Spieler 2

```
[V13]    1: ($M 4000,20000,64000)
[694]    2: (Programm: tvtetris.pas
[6X3]    3:  Funktion: Tetris-Clone
[F64]    4:  Sprache:  Turbo Pascal 5.0
[V44]    5:  Autor:    Hartmut Günther
[685]    6:  (c) 1993 DMV Widuch GmbH & Co. KG )
[000]    7:
[WD3]    8: program TV_TETRIS;
[AS0]    9: uses
[XS6]   10:   crt,objects,drivers,app,menus,dialogs,
[NC2]   11:    views,gadgets;
[000]   12:
[6X0]   13: const
[FE5]   14:   E_Zeit : String[12] = 'Ver 22.8.92';
[RY1]   15:   cminfo  = 100;
[672]   16:   cmsetup = 101;
[PX1]   17:   cmgame  = 102;
[9Z1]   18:   cmdown  = 103;
[000]   19:
[TZ6]   20: (Im Zweispieler-Modus befindet sich aus
[ZH6]   21: technischen Gründen Spieler 1 auf der
[757]   22: rechten und Spieler 2 auf der linken Seite.
[895]   23: Dies kann man über die Variable
[9V5]   24: gamedlg.Spieler nachvollziehen)
[000]   25:
[TC4]   26: (Kommandos für Spieler 1)
[V52]   27:   cmright = 104;
[LZ1]   28:   cmleft  = 105;
[F72]   29:   cmrotate= 106;
[AZ1]   30:   cmdrop  = 107;
[000]   31:
[UC4]   32: (Kommandos für Spieler 2)
[DT1]   33:   cmri    = 108;
[LS1]   34:   cmle    = 109;
[4T1]   35:   cmro    = 110;
[HR1]   36:   cmdr    = 111;
[000]   37:
[9Y6]   38: (Das Kommando cmnxfig1 wird von Spieler 1
[U07]   39: abgeschickt, um bei Sp2 das nächste Teil
[327]   40: zu verändern. cmnxfig2 ändert das nächste
[CL6]   41: Teil von Sp1. Diese Aktion macht nur bei
[K17]   42: eingeschalteter Figurenvorschau Sinn.)
[N42]   43:   cmnxfig1= 112;
[S42]   44:   cmnxfig2= 113;
[AX5]   45:   cmidle  = 114; (Gadgets aktivieren)
[ST0]   46: type
[GX5]   47:   figuretyp = array[1..4,1..5] of Word;
[P26]   48:   fieldtyp  = array[1..20,1..20] of Word;
[3L5]   49:   top_Typ   = array[1..20] of record
[7X2]   50:     name : String[10];
[GN2]   51:     pktzahl : Word;
[ZN0]   52:   end;
[K93]   53:   dlgdatatyp = record
[7P2]   54:    startlevelflag,
[612]   55:    extsetflag,
[T54]   56:    Soundflag, (Ton an / aus)
[F86]   57:    nextflag,  (Figurenvorschau an / aus)
[T16]   58:    twoflag,   (Ein- / Zweispieler-Modus)
[CR5]   59:    spnr : Word; (Nummer des Spieler)
[ZN0]   60:   end;
[000]   61:
[F25]   62:   mainapp = object(tapplication)
[7X2]   63:     name : String[10];
[SF3]   64:     dlgdata : dlgdatatyp;
[MA5]   65:     heap : pheapview;  (Zwei Gadgets)
[KA3]   66:     clock : pclockview;
[U43]   67:     constructor init;
[2G5]   68:     procedure initmenubar; virtual;
[556]   69:     procedure initstatusline; virtual;
[TX3]   70:     procedure information;
[7U3]   71:     procedure handleevent(
            var event : tevent); virtual;
[5W4]   72:     procedure idle; virtual;
[H64]   73:   end;
[ZN0]   74:
[000]   75:
[BU3]   76:   psetupdlg = ^setupdlg;
[D84]   77:   setupdlg = object(tdialog)
[U43]   78:     constructor init;
[ZN0]   79:   end;
[T84]   80:   (Eine Highscore-Tabelle)
[945]   81:   phighscoredlg = ^highscoredlg;
[9X4]   82:   highscoredlg = object(Tdialog)
[WU3]   83:     top_twenty : top_Typ;
[D55]   84:     constructor init(sn : String;
[ZX1]   85:       pkt : Word);
[ZN0]   86:   end;
[000]   87:
[1X3]   88:   pinfoview = ^infoview;
[004]   89:   infoview = object(tview)
[FZ1]   90:     sc,cd : Word;
[U02]   91:     sl,fl : Byte;
[RU5]   92:     constructor init(bounds : trect);
[8B4]   93:     procedure showinfo(score,
              credits : Word;fullines,
[T64]   94:       startlevel : Byte);
[563]   95:     procedure draw;virtual;
[554]   96:   end;
[ZN0]   97:
[000]   98:
[L54]   99:   pfieldview = ^fieldview;
[734]  100:   fieldview = object(tview)
[CT2]  101:     field : fieldtyp;
[G96]  102:     procedure showfield(data : fieldtyp);
```

```
[B74]  103:     procedure draw; virtual;
[ZN0]  104:   end;
[000]  105:
[FW3]  106: (Die Figurenvorschau)
[6B5]  107:   pnextview = ^nextview;(neu)
[W14]  108:   nextview = object(tview)
[UK2]  109:     nextfig : Byte;
[RU5]  110:     constructor init(bounds : trect);
[945]  111:     procedure shownext(nt : Byte);
[B74]  112:     procedure draw; virtual;
[ZN0]  113:   end;
[000]  114:
[9B7]  115: (Eingabebox für den Namen des Spielers)
[S83]  116:   pnamedlg = ^namedlg;
[V34]  117:   namedlg = object(tdialog)
[WX6]  118:     constructor init(s : String;pkt :Word);
[ZN0]  119:   end;
[000]  120:
[TH6]  121:   (zwei Instanzen des Objekts gamedlg)
[GC3]  122:   pgamedlg2 = ^gamedlg;
[EC3]  123:   pgamedlg1 = ^gamedlg;
[B64]  124:   gamedlg = object(twindow)
[NR4]  125:     gamefieldview : pfieldview;
[8D4]  126:     gameinfoview  : pinfoview;
[BF4]  127:     gamenextview  : pnextview;
[222]  128:     creditview;
[MW4]  129:     gameoverview  : pstatictext;
[B83]  130:     dostime :^Longint;
[XC5]  131:     oct, (Beginn der Creditchance)
[213]  132:     oldtime : Longint;
[CT2]  133:     field : fieldtyp;
[QT2]  134:     fig : figuretyp;
[WP2]  135:     col : shortint;
[4C6]  136:     level,startlevel,soundeff,nextshow,
[K55]  137:     spieler,rot,line,linedropped,
[MQ7]  138:     maxfigures,fullines,num,nextnum : Byte;
[E94]  139:     credits, (Bonuspunkte)
[C42]  140:     score : Word;
[TF1]  141:     c_aktiv,
[S63]  142:     gameover : Boolean;
[YS6]  143:     constructor init(dlgdata : dlgdatatyp);
[7U3]  144:     procedure handleevent(
              var event : tevent);virtual;
[VU4]  145:     procedure getfigure;
[GH3]  146:     procedure movefigure(key : Word);
[566]  147:     procedure sfx_row; (Sound-Effekte)
[WR3]  148:     procedure sfx_credit;
[704]  149:     procedure sfx_gameovr;
[U55]  150:     function check(showf : Boolean)
[FK1]  151:       : Boolean;
[ZN0]  152:   end;
[000]  153:
[4W2]  154:   pstart = ^start;
[5T3]  155:   start = object(tdialog)
[YS6]  156:     constructor init(dlgdata : dlgdatatyp);
[KG5]  157:     procedure handleevent(var event
[K43]  158:       : tevent);virtual;
[ZN0]  159:   end;
[000]  160:
[RN0]  161: var
[TL2]  162:   main    : mainapp;
[313]  163:   twomode : Boolean;
[YW4]  164:   (Punkte von Spieler 1 und 2)
[5Z3]  165:   punkte1,punkte2 : Word;
[000]  166:
[9K4]  167: constructor mainapp.init;
[H92]  168: var r : trect;
[MU0]  169: begin
[T63]  170:   tapplication.init;
[8P2]  171:   with dlgdata do
[MU0]  172:   begin
[U53]  173:     startlevelflag := 5;
[5G2]  174:     extsetflag := 0;
[1C2]  175:     soundflag  := 1;
[982]  176:     nextflag   := 0;
[N72]  177:     twoflag    := 0;
[ZN0]  178:   end;
[000]  179:
[2L3]  180: (Gadgets einfügen)
[782]  181:   getextent(R);
[6S1]  182:   dec(R.B.X);
[R85]  183:   R.A.X := R.B.X - 19; R.B.Y := R.A.Y+1;
[BN2]  184:   R.B.X := R.A.X + 8;
[435]  185:   heap := new(pheapview, Init(R));
[000]  186:
[782]  187:   GetExtent(R);
[H25]  188:   R.A.X := R.B.X - 9; R.B.Y := R.A.Y+1;
[0C5]  189:   clock := new(pclockview,init(r));
[162]  190:   insert(heap);
[FB2]  191:   insert(clock);
[S52]  192:   information;
[3G1]  193:   nosound;
[ZN0]  194: end;
[000]  195:
[W85]  196: procedure mainapp.initmenubar;
[RN0]  197: var
[UK1]  198:   r : trect;
[MU0]  199: begin
[782]  200:   getextent(r);
[4C2]  201:   r.b.y := r.a.y+1;
[WP6]  202:   menubar := new(pmenubar,init(r, newmenu(
[TU6]  203:     newsubmenu('~T~V-Tetris',hcnocontext,
[PF6]  204:     newmenu(newitem('~I~nformation','F1',
[9R5]  205:       kbf1,cminfo,hcnocontext,newitem(
```

```
[WD5]  206:       '~V~oreinstellungen','F2',kbf2,
[175]  207:       cmsetup,hcnocontext,newitem(
[894]  208:       '~S~piel','F3',kbf3,cmgame,
[GP3]  209:       hcnocontext,newitem(
[764]  210:       '~E~nde','F4',kbf4,cmquit,
[FD3]  211:       hcnocontext,nil))))),
[W31]  212:       nil))));
[ZN0]  213: end;
[000]  214:
[NW5]  215: procedure mainapp.initstatusline;
[RN0]  216: var
[UK1]  217:   r : trect;
[MU0]  218: begin
[782]  219:   getextent(r);
[9C2]  220:   r.a.y := r.b.y-1;
[F56]  221:   statusline := new(pstatusline,init(r,
[ES2]  222:    newstatusdef(0,0,
[SC6]  223:     newstatuskey('~F1~ Info',kbf1,cminfo,
[PW6]  224:     newstatuskey('~F2~ Setup',kbf2,cmsetup,
[NE6]  225:     newstatuskey('~F3~ Spiel',kbf3,cmgame,
[8D6]  226:     newstatuskey('~F4~ Ende',kbf4,cmquit,
[B86]  227:     newstatuskey('~#27~ Links',kbleft,
[U81]  228:       cmleft,
[FE5]  229:     newstatuskey('~#24~ Rot',kbup,
[WK1]  230:       cmrotate,
[PG6]  231:     newstatuskey('~#26~ Rechts',kbright,
[0F1]  232:       cmright,
[7C6]  233:     newstatuskey('~#25~ Runter',kbdown,
[D81]  234:       cmdrop,
[G83]  235:       (Luckenfüller)
[Z24]  236:     newstatuskey('  ',kbf10,cmmenu,
[W95]  237: (es folgen die Kommandos für)
[TX4]  238: (Spieler 2, es werden sowohl)
[T85]  239: (Klein- als auch Großbuch-)
[UE4]  240: (staben berücksichtigt )
[C26]  241:     newstatuskey('~#27~ Li2',$1E61,cmle,
[426]  242:     newstatuskey('~#27~ Li2',$1E41,cmle,
[626]  243:     newstatuskey('~#24~ Ro2',$1177,cmro,
[Y16]  244:     newstatuskey('~#24~ Ro2',$1157,cmro,
[G16]  245:     newstatuskey('~#26~ Re2',$2064,cmri,
[816]  246:     newstatuskey('~#26~ Re2',$2044,cmri,
[346]  247:     newstatuskey('~#25~ Ru2',$1F73,cmdr,
[V36]  248:     newstatuskey('~#25~ Ru2',$1F53,cmdr,
[VZ4]  249:     (wegen dem Zweispielermodus)
[VX4]  250:     newstatuskey('',$011B,cmcancel,
[EG2]  251:     nil))))))))))))))))),
[941]  252:   nil)));
[ES5]  253:   disablecommands([cmrotate,cmleft,
[G23]  254:    cmright,cmdrop]);
[ZN0]  255: end;
[000]  256:
[895]  257: procedure mainapp.information;
[RN0]  258:   var
[XW1]  259:     d : pdialog;
[UK1]  260:     r : trect;
[Z52]  261:     dummy : Word;
[MU0]  262:   begin
[D13]  263:     r.assign(0, 0, 54, 18);
[6T3]  264:     d := New(pdialog, init(r,
[5L2]  265:       'Information'));
[UT1]  266:     with d^ do
[MU0]  267:     begin
[XH5]  268:       options := options or ofcentered;
[832]  269:       r.grow(-1, -1);
[VY1]  270:       dec(R.B.Y, 3);
[AB5]  271:       insert(new(pstatictext, init(r,
[094]  272:         '~C~TV-TETRIS V2.2'#13 + #13#13 +
[KE9]  273:         '~C~Dieses Programm wurde in Turbo P
ascal 6.0 geschrieben.'
[6W0]  274:         +#13#13+
[PA8]  275:         '~C~Autor von TV-TETRIS V2.2: Hartmu
t Günther'+#13+
[NM5]  276:         '      (c) 1993 DMV Widuch GmbH &
Co. KG')));
[WV2]  277:       r.assign(2,10,53,14);
[AB5]  278:       insert(new(pstaticText, init(r,
[NC8]  279:         'Mit der Taste F2 kann alles einges
tellt werden.'+#13+
[VL8]  280:         'Im Zweispieler-Modus werden die Ta
sten a,s,d,e'+#13+
[7Z7]  281:         'von Spieler 2 benützt.          Pla
y TV-TETRIS!'+#13
[CD0]  282:         )));
[AB3]  283:       r.assign(19, 15, 29, 17);
[LR5]  284:       insert(new(pbutton, init(r, 'O~K',
[A33]  285:         cmOk, bfDefault)));
[ZN0]  286:     end;
[C24]  287:     dummy := desktop^.execview(d);
[7M2]  288:     dispose(d,done);
[ZN0]  289:   end;
[000]  290:
[000]  291:
[000]  292:
[N07]  293: procedure mainapp.handleevent(var event :
[BC1]  294:  tevent);
[587]  295:   procedure setup(var dlgdata :dlgdatatyp);
[N03]  296:   var p : psetupdlg;
[MU0]  297:   begin
[244]  298:     p := new(psetupdlg,init);
[MB3]  299:     if validview(p) <> nil
[ZN1]  300:     then begin
[L73]  301:         p^.setdata(dlgdata);
[DV5]  302:         if desktop^.execview(p) <> cmcancel
[0A4]  303:         then p^.getdata(dlgdata);
[ZM2]  304:         dispose(p,done);
[4M0]  305:     end
[ZN0]  306:   end;
[000]  307:
[KP4]  308: (Gibt im Zwei-Spielermodus
[PG3]  309: den Sieger bekannt)
[CY3]  310:   procedure Info_Sieger;
[RN0]  311:   var
[XW1]  312:     d : pdialog;
[UK1]  313:     r : trect;
[Z52]  314:     dummy : Word;
[6K0]  315:     sp,
[7U2]  316:     ps1, ps2 : String;
[MU0]  317:   begin
[D63]  318:     if punkte1 > punkte2
[412]  319:     then sp := '1'
[S54]  320:     else if punkte2 > punkte1
[612]  321:          then sp := '2'
[HC4]  322:          else sp := '1 und Spieler 2';
[L73]  323:     str(punkte1:5,ps1);
[T73]  324:     str(punkte2:5,ps2);
[R03]  325:     r.assign(0, 0, 50, 11);
[2G4]  326:     d := new(pdialog, init(r, ''));
[UT1]  327:     with d^ do
[MU0]  328:     begin
[XH5]  329:       options := options or ofCentered;
[832]  330:       r.grow(-1, -1);
[VY1]  331:       dec(r.b.y, 3);
[AB5]  332:       insert(new(pstatictext, init(r,
[VQ0]  333:         #13 +
[L43]  334:         '  Spieler 1 hat '+PS1+
[L22]  335:         ' Punkte'+#13 +
[R43]  336:         '  Spieler 2 hat '+PS2+
[G24]  337:         ' Punkte erreicht.'+#13+#13+
[ET3]  338:         '  Damit hat Spieler '+Sp+
[TR2]  339:         ' gewonnen!!'+#13)));
[B63]  340:       r.assign(19, 8, 29, 10);
[LR5]  341:       insert(new(pbutton, init(r, 'O~K',
[A33]  342:         cmOk, bfDefault)));
[ZN0]  343:     end;
[C24]  344:     dummy := desktop^.execview(d);
[7M2]  345:     dispose(d,done);
```

```
[ZNØ]  346:   end;
[ØØØ]  347:
[ZE7]  348: (Ermittelt den Namen des Spielers und zeigt
[WW5]  349: ggf. die Highscoretabelle an. Im Ein-
[MD6]  35Ø: Spielermodus wird snr = Ø übergeben)
[BG5]  351:   procedure Auswertung(Snr : Byte;
[ZX1]  352:    Pkt : Word);
[RNØ]  353:   var
[DY2]  354:     h : phighscoredlg;
[212]  355:     n : pnamedlg;
[NL4]  356:     d : Word;   (Dummy-Variable)
[YW1]  357:     S : String;
[MUØ]  358:   begin
[RA1]  359:     if snr > Ø
[ZN1]  36Ø:     then begin
[312]  361:            str(snr,s);
[ML2]  362:            s := 'Spieler '+S;
[4MØ]  363:          end
[5S1]  364:     else s := '';
[UX4]  365:     n := new(pnamedlg,init(s,pkt));
[EB3]  366:     if validview(n) <> nil
[ZN1]  367:     then begin
[XM5]  368:       if desktop^.execview(n) = cmcancel
[ZN1]  369:       then begin
[VM2]  37Ø:         dispose(n,done);
[ØWØ]  371:         exit;
[ZNØ]  372:       end;
[D72]  373:       n^.getdata(s);
[VM2]  374:       dispose(n,done);
[AV2]  375:       if s = '' then exit;
[ØU5]  376:       h := new(phighscoredlg,init(s,pkt));
[WA3]  377:       if validview(h) <> nil
[ZN1]  378:       then begin
[T94]  379:         if (snr <> 1) or (punkte2 = Ø)
[H35]  38Ø:         then d := desktop^.execview(h);
[FM2]  381:         dispose(h,done);
[ZNØ]  382:       end;
[ZNØ]  383:     end;
[ZNØ]  384:   end;
[ØØØ]  385:
[T36]  386:   procedure game(dlgdata : dlgdatatyp);
[RNØ]  387:   var
[YX1]  388:     p : pstart;
[EY4]  389:     d : Word;   ( Dummy-Variable )
[MUØ]  39Ø:   begin
[A55]  391:     p := new(pstart,init(dlgdata));
[MB3]  392:     if validview(p) <> nil
[ZN1]  393:     then begin
[KM5]  394:       disablecommands([cmquit,cminfo,
[R23]  395:        cmsetup,cmgame]);
[8K5]  396:       enablecommands([cmrotate,cmleft,
[AD6]  397:        cmright,cmdrop,cmle,cmri,cmdr,cmro])
                ;
[664]  398:       d := desktop^.execview(p);
[ZM2]  399:       dispose(p,done);
[ES5]  4ØØ:       disablecommands([cmrotate,cmleft,
[AD6]  4Ø1:        cmright,cmdrop,cmle,cmri,cmdr,cmro])
                ;
[735]  4Ø2:       enablecommands([cmquit,cminfo,
[R23]  4Ø3:        cmsetup,cmgame]);
[BY2]  4Ø4:       if (punkte1 = Ø) and
[UR3]  4Ø5:        (punkte2 = Ø) then exit;
[ØA3]  4Ø6:       if dlgdata.twoflag = Ø
[5P4]  4Ø7:       then auswertung(Ø,punkte1)
[WP1]  4Ø8:       else begin
[372]  4Ø9:              info_sieger;
[DP6] 41Ø: if punkte1 > Ø then auswertung(1,punkte1);
[KP6] 411: if punkte2 > Ø then auswertung(2,punkte2);
[ZNØ]  412:       end;
[ZNØ]  413:     end;
[ZNØ]  414:   end;
[ØØØ]  415:
[X45]  416: begin (mainapp.handleevent)
[DK5]  417:   tapplication.handleevent(event);
[G24]  418:   if event.what = evcommand
[ZN1]  419:   then begin
[PE3]  42Ø:          case event.command of
[AM3]  421:            cminfo  : information;
[6B4]  422:            cmsetup : setup(dlgdata);
[TK3]  423:            cmgame  : game(dlgdata);
[GS1]  424:            else exit;
[ZNØ]  425:          end;
[P23]  426:          clearevent(event);
[ZNØ]  427:        end;
[ZNØ]  428: end;
[ØØØ]  429:
[514]  43Ø: procedure mainapp.Idle;
[MUØ]  431: begin
[P43]  432:   tapplication.idle;
[3H5]  433:   heap^.update;(Gadgets updaten)
[M72]  434:   clock^.update
[ZNØ]  435: end;
[ØØØ]  436:
[K22]  437: (setupdlg)
[YT4]  438: constructor setupdlg.init;
[RNØ]  439: var
[1C3]  44Ø:   dlgsubview : pview;
[UK1]  441:   r : trect;
[MUØ]  442: begin
[TS2]  443:   r.assign(Ø,Ø,79,23);
[P36]  444:   tdialog.init(r,'Voreinstellungen');
[XH5]  445:   options := options or ofcentered;
[UY1]  446:   r.grow(-1,-1);
[NR2]  447:   r.assign(3,2,13,12);
[PS6]  448:   dlgsubview := new(pradiobuttons,init(r,
[Y65]  449:     newsitem(' ^Ø^',newsitem(' ^1^',
[675]  45Ø:     newsitem(' ^2^',newsitem(' ^3^',
[E75]  451:     newsitem(' ^4^',newsitem(' ^5^',
[P75]  452:     newsitem(' ^6^',newsitem(' ^7^',
[Y75]  453:     newsitem(' ^8^',newsitem(' ^9^',
[GX1]  454:   nil))))))))));
[3D3]  455:   insert(dlgsubview);
[ØØØ]  456:
[5R2]  457:   r.assign(2,1,13,2);
[2V6]  458:   insert(new(plabel,init(r,'^S^tartlevel',
[BA2]  459:    dlgsubview)));
[TS2]  46Ø:   r.assign(17,2,46,3);
[K25]  461:   insert(new(pcheckboxes,init(r,
[CX6]  462:    newsitem(' ^E^rweiterter Figurensatz',
[W31]  463:     nil))));
[YS2]  464:   r.assign(17,3,46,4);
[K25]  465:   insert(new(pcheckboxes,init(r,
[YC5]  466:    newsitem(' ^G^eräusche',nil))));
[3T2]  467:   r.assign(17,4,46,5);
[K25]  468:   insert(new(pcheckboxes,init(r,
[MA7]  469:    newsitem(' ^N^ächstes Teil zeigen',nil)))
              );
[8T2]  47Ø:   r.assign(17,5,46,6);
[K25]  471:   insert(new(pcheckboxes,init(r,
[WX6]  472:    newsitem(' ^Z^wei Spieler-Modus',nil))));
[VX2]  473:   R.assign(17,7,78,14);
[Ø34]  474:   insert(new(pstatictext,
[A26]  475:     init(r,'Tasten beim 2 Spieler-Modus:'
              +#13+'Spieler 2:'+#13+
[CV2]  476:     ' A = nach links, D = nach rechts,'#13+
[CG5]  477:     ' W = rotieren,   S = fallenlassen'#13+
[TX5]  478:     'bei 8 Credits: X = fallenlassen bei Sp
              ieler 1'#13+
[X57]  48Ø:     'bei 4 Credits: C = rotieren bei Spiele
              r 1'#13+
[N2A]  481:     'bei 4 Credits: V = Änderung der nächst
              en Figur bei Sp. 1')));
[L13]  482:   r.assign(17,14,78,18);
[Ø34]  483:   insert(new(pstatictext,
[6M3]  484:     init(r,'Spieler 1:'+#13+
[PA8]  485:     'bei 8 Credits: Del  = fallenlassen be
              i Spieler 2'#13+
[2N7]  486:     'bei 4 Credits: End  = rotieren bei Sp
              ieler 2'#13+
[R2A]  487:     'bei 4 Credits: Pgdn = Änderung der nä
              chsten Figur bei Sp. 2')));
[Ø23]  488:   r.assign(17,19,78,22);
[Ø34]  489:   insert(new(pstatictext,
[CHB]  49Ø:     init(r,'Erscheint "Versenke eine Reihe"
              , so gibt es 5 Credits, wenn '+#13+
[AY9]  491:          'man schnell genug eine Reihe au
              sfüllt.  Tip: Man muß die '+
[9U9]  492:          'Credits im richtigen Moment geg
              en den Gegner einsetzen!')));
[4F4]  493:   r.assign(21+38,16-14,32+38,18-14);
[2G6]  494:   insert(new(pbutton,init(r,'^A^bbruch',
[2B3]  495:    cmcancel,bfnormal)));
[RF4]  496:   r.assign(11+38,16-14,19+38,18-14);
[WG6]  497:   insert(new(pbutton,init(r,'^O^K',cmok,
[ØØ2]  498:    bfdefault)));
[ZNØ]  499: end;
[ØØØ]  5ØØ:
[UE5]  5Ø1: constructor Highscoredlg.init;
[RNØ]  5Ø2: var
[UK1]  5Ø3:   r : trect;
[1Y2]  5Ø4:   io_fehler : Word;
[MC1]  5Ø5:   i : Byte;
[EV3]  5Ø6:   dvar : file of top_typ;
[CE1]  5Ø7:   topname,
[TT3]  5Ø8:   s_punkte : String[1Ø];
[NR2]  5Ø9:   rang : String[3];
[AV2]  51Ø:   sp_ins : Boolean;
[MUØ]  511: begin
[3S2]  512:   r.assign(Ø,Ø,51,19);
[3L4]  513:   tdialog.init(r,'Highscore');
[EN4]  514:   (Highscore-Tabelle öffnen)
[ZS4]  515:   assign(dvar,'TVTETRIS.TOP');
[DXØ]  516: {$I-}
[1Z1]  517:   reset(dvar);
[6V3]  518:   io_fehler := ioresult;
[6E2]  519:   if io_fehler = Ø
[ZN1]  52Ø:   then begin
[VX3]  521:          read(dvar,top_twenty);
[6V3]  522:          io_fehler := ioresult;
[LC3]  523:          system.close(dvar);
[6V3]  524:          io_fehler := ioresult;
[ZNØ]  525:        end;
[BXØ]  526: {$I+}
[ØØØ]  527:
[XH5]  528:   options := options or ofcentered;
[UY1]  529:   r.grow(-1,-1);
[8S2]  53Ø:   r.assign(8,2,35,3);
[Ø65]  531:   insert(new(pstatictext,init(r,
[944]  532:    'Die 2Ø besten Spieler:')));
[ØØØ]  533:
[XN2]  534:   sp_ins := false;
[XP2]  535:   for i := 1 to 2Ø do
[MUØ]  536:   begin
[9C6]  537:     if (pkt > top_twenty[i].pktzahl) and
[HP2]  538:        (sp_ins = false)
[ZN1]  539:     then begin
[XR6]  54Ø:       move(top_twenty[i],top_twenty[i+1],
[PØ5]  541:        sizeof(top_twenty[i])*(2Ø-i));
[NB4]  542:       top_twenty[I].name := sn;
[Z55]  543:       top_twenty[I].pktzahl := pkt;
[BM2]  544:       sp_ins := true;
[ZNØ]  545:     end;
[S1I]  546:     if i > 1Ø
[DR4]  547:     then r.assign(28,i-1Ø+4,49,i-1Ø+5)
[TØ4]  548:     else r.assign(2,i+4,23,i+5);
[ØØØ]  549:
[127]  55Ø:     str(top_twenty[I].pktzahl:5,s_punkte);
[X26]  551:     fillchar(topname,sizeof(topname),Ø);
[W95]  552:     topname := top_twenty[I].name;
[WZ5]  553:     topname[Ø] := chr(sizeof(topname)-1);
[8S4]  554:     str(i:2,rang); rang := rang+'.';
[636]  555:     insert(new(pstatictext,init(r,rang+
[RW3]  556:      topname+'  '+s_punkte)));
[ZNØ]  557:   end;
[ØØØ]  558:
[CQ3]  559:   r.assign(2Ø,16,3Ø,18);
[PN5]  56Ø:   insert(new(pbutton,init(r,'^O^K',
[7W2]  561:    cmok,bfdefault)));
[ØØØ]  562:
[ØR5]  563:   if (io_fehler = Ø) or (io_fehler = 2)
[ZN1]  564:   then begin
[NB3]  565: ($I-) rewrite(dvar);
[6V3]  566:     io_fehler := ioresult;
[KE4]  567:     if io_fehler <> Ø then exit;
[Ø44]  568:     write(dvar,top_twenty);
[6V3]  569:     io_fehler := ioresult;
[LC3]  57Ø:     system.close(dvar);
[6V3]  571:     io_fehler := ioresult;
[6N1]  572: ($I+) end;
[ZNØ]  573: end;
[ØØØ]  574:
[8H4]  575: constructor namedlg.init;
[RNØ]  576: var
[UK1]  577:   R : Trect;
[5L4]  578:   Txt,S_Punkte : String[1Ø];
[5T3]  579:   Nsubview : Pinputline;
[MUØ]  58Ø: begin
[5S2]  581:   R.assign(Ø,Ø,48,7);
[8Z2]  582:   tdialog.init(r,S);
[XH5]  583:   Options := Options or ofCentered;
[ØØØ]  584:
[FY1]  585:   case Pkt of
[LK2]  586:     Ø..999 : Txt := '';
[B34]  587:     1ØØØ..2999 : Txt := 'Bravo! ';
[C54]  588:     3ØØØ..4999 : Txt := 'Super! ';
[DU3]  589:     else Txt := 'Spitze! ';
[ZNØ]  59Ø:   end;
[ØØØ]  591:
[WB3]  592:   str(Pkt,S_Punkte);
[TR2]  593:   R.assign(3,1,45,2);
[V66]  594:   insert(new(Pstatictext,init(R,#3+Txt+
[3L3]  595:    'Sie haben '+S_Punkte+
[ØK3]  596:    ' Punkte erreicht!')));
[ØØØ]  597:
[US2]  598:   R.assign(25,4,33,6);
[PN5]  599:   insert(new(Pbutton,init(R,'^O^K',
[7W2]  6ØØ:    cmok,bfdefault)));
[4T2]  6Ø1:   R.assign(35,4,46,6);
[Q2G]  6Ø2:   insert(new(Pbutton,init(R,'^A^bbruch',
[2B3]  6Ø3:    cmcancel,bfnormal)));
[TR2]  6Ø4:   R.assign(3,4,15,5);
[AN6]  6Ø5:   Nsubview := new(PInputLine, init(R, 1Ø));
[UZ2]  6Ø6:   insert(Nsubview);
[LR2]  6Ø7:   R.assign(2,3,15,4);
[ØW5]  6Ø8:   insert(new(Plabel, init(R,'Ihr Name:',
[9Y1]  6Ø9:    Nsubview)));
[ZNØ]  61Ø: end;
[ØØØ]  611:
[6A4]  612: constructor start.init;
[RNØ]  613: var
[UK1]  614:   r : trect;
[MUØ]  615: begin
[7C6]  616:   r.assign(Ø,Ø,5Ø+Byte(dlgdata.twoflag)*3Ø,
[TEØ]  617:    22);
[S44]  618:   tdialog.init(r,'Tetris');
[ØØØ]  619:
[3A3]  62Ø:   if dlgdata.twoflag = 1
[ZN1]  621:   then begin
[LX2]  622:     dlgdata.spnr := 2;
[ØY5]  623:     insert(new(pgamedlg2,init(dlgdata)));
[ZNØ]  624:   end;
[ØØØ]  625:
[HX2]  626:   dlgdata.spnr := 1;
[ZX5]  627:   insert(new(pgamedlg1,init(dlgdata)));
[6BB]  628:   options := options or ofpreprocess or ofp
              ostprocess or ofcentered;
[ZNØ]  629: end;
[ØØØ]  63Ø:
[G15]  631: procedure start.handleevent;
[H92]  632: var r : trect;
[MUØ]  633: begin
[GB4]  634:   if event.what = evkeyboard
[G94]  635:   then if event.charcode = #13
[ZN1]  636:     then begin
[EZ4]  637:       (Spiel abbrechen bei <ENTER>)
[BZ3]  638:       event.what := evcommand;
[174]  639:       event.command := cmcancel;
[ZNØ]  64Ø:     end;
[ØØØ]  641:
[154]  642:   if event.what = evnothing
[ZN1]  643:   then begin
[V77]  644: ( Damit wird gamedlg.handleev ausgeführt)
[BZ3]  645:     event.what := evcommand;
[WW3]  646:     event.command := cmidle;
[ZNØ]  647:   end;
[HT4]  648:   tdialog.handleevent(event);
[ZNØ]  649: end;
[ØØØ]  65Ø:
[3R1]  651: (gamedlg)
[UG4]  652: constructor gamedlg.init;
[RNØ]  653: var
[EWØ]  654:   Ctxt,
[SZ1]  655:   Sn : String;
[UK1]  656:   r : trect;
[MUØ]  657: begin
[695]  658:   Twomode := (dlgdata.twoflag = 1);
[YZ3]  659:   Spieler := dlgdata.spnr;
[VU2]  66Ø:   str(Spieler,Sn);
[ØØØ]  661:
[DP2]  662:   if Twomode = True
[ZN1]  663:   then begin
[994]  664:     r.assign((2-Spieler)*43,Ø,37+
[4P2]  665:      (2-Spieler)*43,22);
[RY2]  666:     Sn := 'Spieler '+Sn;
[4MØ]  667:   end
[WP1]  668:   else begin
[CR2]  669:     r.assign(Ø,Ø,5Ø,22);
[K92]  67Ø:     Sn := 'Tetris';
[ZNØ]  671:   end;
[ØØØ]  672:
[UE3]  673:   twindow.init(r,Sn,Ø);
[YZ3]  674:   Spieler := dlgdata.spnr;
[ØØØ]  675:
[ØZ5]  676:   r.assign(1,1,36+(1-Byte(Twomode))*13,21);
[AG4]  677:   insert(new(pview,init(r)));
[ØØØ]  678:
[8R2]  679:   r.assign(1,1,21,21);
[7T6]  68Ø:   gamefieldview := new(pfieldview,init(r));
[1T3]  681:   insert(gamefieldview);
[ØØØ]  682:
[KN3]  683:   if dlgdata.nextflag > Ø
[ZN1]  684:   then begin
[XZ2]  685:     r.assign(22,1Ø,31,11);
[Ø65]  686:     insert(new(pstatictext,init(r,
[3N2]  687:      'Nächstes:')));
[GØ3]  688:     r.assign(22,11,28,14);
[KN6]  689:     gamenextview := new(pnextview,init(r));
[LR3]  69Ø:     insert(gamenextview);
[ZNØ]  691:   end;
[ØØØ]  692:
[DN1]  693:   if Twomode
[3E4]  694:   then Ctxt := #13#13+'Credits:'
[9B2]  695:   else Ctxt := '';
[GS2]  696:   r.assign(22,2,3Ø,9);
[EUB]  697:   insert(new(pstatictext,init(r,'Punkte:'#1
              3#13'Linien:'#13#13'Level:'+Ctxt)));
[ØØØ]  698:
[GS2]  699:   r.assign(3Ø,2,35,9);
[FL6]  7ØØ:   gameinfoview := new(pinfoview,init(r));
[PN3]  7Ø1:   insert(gameinfoview);
[ØØØ]  7Ø2:
[ZØ3]  7Ø3:   r.assign(24,15,34,16);
[EP5]  7Ø4:   gameoverview := new(pstatictext,
[8D3]  7Ø5:    init(r,'Spielende!'));
[NP3]  7Ø6:   insert(gameoverview);
[MD3]  7Ø7:   gameoverview^.hide;
[ØØØ]  7Ø8:
[813]  7Ø9:   r.assign(24,18,35,2Ø);
[U35]  71Ø:   Creditview := new(pstatictext,
[B25]  711:    init(r,'Versenke eine Reihe!'));
[4C3]  712:   insert(Creditview);
[E33]  713:   Creditview^.hide;
[ØØØ]  714:
[2N1]  715:   flags := Ø;
[323]  716:   eventmask := $FFFF;
[ZF3]  717:   dostime := ptr($4Ø,$6C);
[Z25]  718:   fillchar(field,sizeof(field),Ø);
[V66]  719:   startlevel := Dlgdata.startlevelflag;
[F35]  72Ø:   Soundeff   := Dlgdata.soundflag;
[E25]  721:   Nextshow   := Dlgdata.nextflag;
[X13]  722:   gameover  := false;
[XP1]  723:   level := Ø;
[YH2]  724:   maxfigures := 7;
[EY1]  725:   Punktel := Ø;
[KY1]  726:   Punkte2 := Ø;
[ØØØ]  727:
[8P1]  728:   score := Ø;
[LC8]  729:   inc(maxfigures,byte(dlgdata.extsetflag an
              d 1=1)*3);
[1T1]  73Ø:   randomize;
[395]  731:   Nextnum := random(maxfigures)+1;
[VT1]  732:   getfigure;
[A33]  733:   (alle Ereignisse)
[A19]  734:   options := options or ofpreprocess or ofp
              ostprocess;
[Ø55]  735:   state := state and not sfshadow;
[ZNØ]  736: end;
[ØØØ]  737:
[U55]  738: procedure gamedlg.handleevent;
[164]  739:   procedure Calc_Credits;
[MUØ]  74Ø:   begin
[CK4]  741:     if Twomode = false then exit;
[WN1]  742:     if C_aktiv
[ZN1]  743:     then begin
[S56]  744:       ( die Chance ist nach 4Ø Ticks vorbei)
[T83]  745:       if Dostime^-Oct > 4Ø
[ZN1]  746:       then begin
[E33]  747:         Creditview^.hide;
[5Y2]  748:         C_aktiv :=  false;
[ZNØ]  749:         end;
[4MØ]  75Ø:       end
                  else if (random(5ØØØØ) = 1) or (Dostime
                  ^-Oct > 72Ø)
[ZN1]  752:       then begin
[KS2]  753:         Oct := Dostime^;
[Z53]  754:         Creditview^.show;
[YP2]  755:         C_aktiv := True;
[ZNØ]  756:       end;
[ZNØ]  757:   end;
[ØØØ]  758:
[WZ6]  759: (Diese Prozedur prüft, ob Spieler 1 eine
[3X6]  76Ø: Aktion gegen Spieler 2 startet. Sind genug
[N36]  761: Credits vorhanden, so wird über eine
[U96]  762: Message das Kommando an alle Desktop-
```

Der Tetris-Clone »tvtetris.pas« ist mit Turbo Vision geschrieben.

```
[ØU6]  763: Objekte geschickt. Das View-Objekt von Sp2
[3T5]  764: verarbeitet dann die Botschaft und
[X95]  765: läßt z.B. ein Teil fallen.)
[ØØØ]  766:
[WU3]  767:   procedure Check_Keys1;
[RNØ]  768:   var
[DW2]  769:     Dummyp : Pointer;
[MUØ]  77Ø:   begin
[VR3]  771:     case event.scancode of
[BL3]  772:       $53 : if Credits >= 8
[ZN1]  773:             then begin
[WH2]  774:                  dec(Credits,8);
[2A4]  775:                  Dummyp := Message(desktop,
[ØG3]  776:                   evcommand, cmdr, nil);
[ZNØ]  777:                 end;
[C33]  778:       $4F : if Credits >= 4
[ZN1]  779:             then begin
[MH2]  78Ø:                  dec(Credits,4);
[2A4]  781:                  Dummyp := Message(desktop,
[YG3]  782:                   evcommand, cmro, nil);
[ZNØ]  783:                 end;
[VØ3]  784:       $51 : if Credits >= 4
[ZN1]  785:             then begin
[MH2]  786:                  dec(Credits,4);
[2A4]  787:                  Dummyp := Message(desktop,
[754]  788:                   evcommand, cmnxfig1, nil);
[ZNØ]  789:                 end;
[GS1]  79Ø:       else exit;
[ZNØ]  791:     end;
[P23]  792:     clearevent(Event);
[6G6]  793:     gameinfoview^.showinfo(score,credits,
[BT3]  794:      fullines,startlevel);
[ZNØ]  795:   end;
[ØØØ]  796:
[875]  797: (Gegenstück zu Check_keys1)
[YU3]  798:   procedure Check_Keys2;
[RNØ]  799:   var
[DW2]  8ØØ:     Dummyp : Pointer;
[MUØ]  8Ø1:   begin
[5S3]  8Ø2:     case event.charcode of
[8M3]  8Ø3:       'x','X' : if Credits >= 8
[ZN1]  8Ø4:                 then begin
[WH2]  8Ø5:                      dec(Credits,8);
[2A4]  8Ø6:                      Dummyp := Message(desktop,
[DP3]  8Ø7:                       evcommand, cmdrop, nil);
[ZNØ]  8Ø8:                     end;
[2G3]  8Ø9:       'c','C' : if Credits >= 4
[ZN1]  81Ø:                 then begin
[MH2]  811:                      dec(Credits,4);
[2A4]  812:                      Dummyp := Message(desktop,
[484]  813:                       evcommand, cmrotate, nil);
[ZNØ]  814:                     end;
[ML3]  815:       'v','V' : if Credits >= 4
[ZN1]  816:                 then begin
[MH2]  817:                      dec(Credits,4);
[2A4]  818:                      Dummyp := Message(desktop,
[B54]  819:                       evcommand, cmnxfig2, nil);
[ZNØ]  82Ø:                     end;
[GS1]  821:       else exit;
[ZNØ]  822:     end;
[P23]  823:     clearevent(Event);
[6G6]  824:     gameinfoview^.showinfo(score,Credits,
[BT3]  825:      fullines,startlevel);
[ZNØ]  826:   end;
[ØØØ]  827:
[RNØ]  828: var
[UK1]  829:   R : Trect;
[MUØ]  83Ø: begin
[7X4]  831:   Twindow.handleevent(event);
[2H2]  832:   if not gameover
[ZN1]  833:   then begin
[CA2]  834:    Calc_Credits;
[356]  835:    if dostime^-(28-startlevel*3)>oldtime
[ZN1]  836:    then begin
[ER4]  837:     inc(oldtime,28-startlevel*3);
[EA3]  838:     movefigure(cmdown);
[4MØ]  839:    end
[X25]  84Ø:    else if event.what = evcommand
[8C4]  841:     then case event.command of
[N54]  842:      cmleft,cmright,cmrotate,
[UP3]  843:      cmdrop : if Spieler = 2
[2K1]  844:               then exit
[SC5]  845:      else movefigure(event.command);
[TH2]  846:      cmle,cmri,cmro,
[WG3]  847:      cmdr :   if Spieler = 1
[2K1]  848:               then exit
[WP1]  849:               else begin
[ØØØ]  85Ø:
[AV6]  851: (movefigure versteht nur die Kommandos
[GT6]  852: cmdrop,cmleft usw. deshalb Konvertierung
[8A4]  853: von z.B cmri zu cmright)
[ØØØ]  854:
[ØØØ]  855:
[UA5]  856:       event.command := event.command-4;
[5F4]  857:       movefigure(event.command);
[ZNØ]  858:       end;
[LL2]  859:       cmnxfig1: begin
[DT3]  86Ø:        if Spieler = 1 then exit;
[395]  861:        Nextnum := random(maxfigures)+1;
[WF2]  862:        if Nextshow > Ø
[CP6]  863:       then gamenextview^.shownext(Nextnum);
[ZNØ]  864:       end;
[ML2]  865:       cmnxfig2: begin
[GT3]  866:        if Spieler = 2 then exit;
[395]  867:        Nextnum := random(maxfigures)+1;
[WF2]  868:        if Nextshow > Ø
[CP6]  869:       then gamenextview^.shownext(Nextnum);
[ZNØ]  87Ø:        end;
[ØØØ]  871:
[GS1]  872:        else exit;
[4MØ]  873:       end
[4N8]  874:       else if (event.what = evkeyboard) an
       d (Twomode = True)
[ZN1]  875:       then begin
[212]  876:        if Spieler = 1
[GW2]  877:        then Check_Keys1
[413]  878:        else Check_Keys2;
[ØWØ]  879:        exit;
[ZNØ]  88Ø:        end;
[P23]  881:       clearevent(event);
[UX1]  882:       if gameover
[ZN1]  883:       then begin
[BH3]  884:        gameoverview^.show;
[492]  885:        SFX_Gameovr;
[212]  886:        if Spieler = 1
[MG3]  887:        then Punkte1 := Score
[3R3]  888:        else Punkte2 := Score;
[ZNØ]  889:       end;
[ZNØ]  89Ø:     end;
[ZNØ]  891: end;
[ØØØ]  892:
[AV4]  893: procedure gamedlg.getfigure;
[6XØ]  894: const
[CM5]  895:   figures: array[1..1Ø,1..4] of word=
[R33]  896:     (($ØØ66,$ØØ66,$ØØ66,$ØØ66),
[UZ2]  897:      ($Ø936,$Ø462,$Ø936,$Ø462),
[AY2]  898:      ($Ø963,$Ø132,$Ø963,$Ø132),
[X23]  899:      ($ØØØF,$4444,$ØØØF,$4444),
[6Y2]  9ØØ:      ($Ø974,$Ø223,$Ø17Ø,$Ø311),
[PY2]  9Ø1:      ($Ø972,$Ø232,$Ø27Ø,$Ø262),
[8Y2]  9Ø2:      ($Ø971,$Ø322,$Ø47Ø,$Ø226),
[6Z2]  9Ø3:      ($Ø957,$Ø313,$Ø75Ø,$Ø646),
[KZ2]  9Ø4:      ($Ø272,$Ø272,$Ø272,$Ø272),
[H63]  9Ø5:       ($Ø117,$Ø711,$Ø744,$Ø447));
[ØØØ]  9Ø6:
[6E4]  9Ø7:   value:array[Ø..15] of word=
[V34]  9Ø8:     ($ØØØ1,$ØØØ2,$ØØØ4,$ØØØ8,$ØØ1Ø,$ØØ2Ø,
[WØ4]  9Ø9:      $ØØ4Ø,$ØØ8Ø,$Ø1ØØ,$Ø2ØØ,$Ø4ØØ,$Ø8ØØ,
[WS2]  91Ø:      $1ØØØ,$2ØØØ,$4ØØØ,$8ØØØ);
[ØØØ]  911:
[RNØ]  912: var
[TK2]  913:   i,j,k : integer;
[ØØØ]  914:
[MUØ]  915: begin
[NY2]  916:   Num     := Nextnum;
[VE5]  917:   Nextnum :=random(maxfigures)+1;
[WF2]  918:   if Nextshow > Ø
[CP6]  919:   then gamenextview^.shownext(Nextnum);
[2G2]  92Ø:   for i := 1 to 4 do
[MUØ]  921:   begin
[SYØ]  922:     j := 1;
[ZP2]  923:     for k := Ø to 15 do
[MUØ]  924:     begin
[RY6]  925:       if figures[num,i] and value[k] = valu
       e[k]
[ZN1]  926:       then begin
[C35]  927:        fig[i,j] := k mod 4+(k shr 2) shl 8;
[B51]  928:        inc(j);
[ZNØ]  929:       end;
[8G5]  93Ø:     if j = 5 then fig[i,j] := fig[i,j-1];
[ZNØ]  931:     end;
[ZNØ]  932:   end;
[964]  933:   col := 7; line := Ø; rot := 1;
[7W2]  934:   linedropped := 2Ø;
[WT6]  935:   movefigure(cmdown); oldtime := dostime^;
[ZNØ]  936: end;
[ØØØ]  937:
[415]  938: procedure gamedlg.movefigure;
[RNØ]  939: var
[CF3]  94Ø:   dummycol : Shortint;
[1K4]  941:   dummyrot,dummyline : Byte;
[KØ3]  942:   onecube : Longint;
[3D3]  943:   linefilled : Boolean;
[ØØØ]  944:
[MUØ]  945: begin
[EB6]  946:   if key=cmdrop then linedropped := line;
[P31]  947:   repeat
[RP2]  948:     dummycol  := col;
[GZ2]  949:     dummyline := line;
[1U2]  95Ø:     dummyrot  := rot;
[DX1]  951:     case key of
[T23]  952:       cmleft   : dec(col,2);
[U93]  953:       cmright  : inc(col,2);
[Z74]  954:       cmdown,cmdrop : inc(line);
[GS6]  955:       cmrotate : inc(rot,1-byte(rot=4) shl
       2);
[ZNØ]  956:     end;
[AY3]  957:     if not check(key<>cmdrop)
[ZN1]  958:     then begin
[6U2]  959:      col  := dummycol;
[X13]  96Ø:      line := dummyline;
[TX2]  961:      rot  := dummyrot;
[E34]  962:      if key in [cmdown,cmdrop]
[ZN1]  963:      then begin
[WL1]  964:       if line > Ø
[ZN1]  965:       then begin
[LWA]  966:         inc(score,startlevel+num+byte(num >
        7) shl 6+(2Ø-linedropped) shr 2);
[212]  967:         if Spieler = 1
[MG3]  968:         then Punkte1 := Score
[3R3]  969:         else Punkte2 := Score;
[CAE]  97Ø:         onecube := 219+word((num+byte(num >
        7))) shl 8 + 219 shl 16+longint((num+Byte(
       num > 7))) shl 24;
[GW3]  971:         for dummycol := 1 to 5 do
[EDG]  972:           move(onecube,field[line+hi(fig[rot
       ,dummycol]),col+lo(fig[rot,dummycol]) shl 1
       ],sizeof(onecube));
[874]  973:         for dummyline := 2 to 2Ø do
[MUØ]  974:          begin
[463]  975:          linefilled := true;
[U34]  976:          for dummycol := 1 to 2Ø do
[AMA]  977:           if lo(field[dummyline,dummycol])
       <> 219 then linefilled := false;
[T32]  978:          if linefilled
[ZN1]  979:          then begin
[GU2]  98Ø:           if C_aktiv = False
[K62]  981:          then SFX_Row
[WP1]  982:           else begin
[VK2]  983:           inc(Credits,5);
[542]  984:           SFX_Credit;
[ZNØ]  985:           end;
[XC2]  986:           inc(fullines);
[KH3]  987:           if fullines mod (1Ø) = Ø
[ZN1]  988:           then begin
[MB6]  989:           if (level < 6) or (fullines mod
       2Ø = Ø) then
[6Z3]  99Ø:             inc(level,byte(level<9));
[VP7]  991:           if level > startlevel then start
       level := level;
[ZNØ]  992:           end;
[C77]  993:           move(field[1,1],field[2,1],4Ø*(d
       ummyline-1));
[944]  994:           fillchar(field[1,1],4Ø,Ø);
[TG5]  995:           gamefieldview^.showfield(field);
[ZNØ]  996:           end;
[K21]  997:           end; (for)
[VT1]  998:           getfigure;
[4MØ]  999:           end
[PN3] 1ØØØ:           else gameover := true;
[N9A] 1ØØ1:           gameinfoview^.showinfo(score,Cre
       dits,fullines,startlevel);
[ZNØ] 1ØØ2:          end;
[ZNØ] 1ØØ3:        end;
[N77] 1ØØ4:   until (key <> cmdrop) or (line = 1) or ga
       meover;
[ZNØ] 1ØØ5: end;
[ØØØ] 1ØØ6:
[HV6] 1ØØ7: (Sound-FX wenn eine Reihe versenkt wird)
[ØØØ] 1ØØ8:
[AM4] 1ØØ9: procedure gamedlg.SFX_Row;
[MUØ] 1Ø1Ø: begin
[H44] 1Ø11:   if soundeff = Ø then exit;
[X23] 1Ø12:   sound(3ØØ);delay(15);
[T23] 1Ø13:   sound(2ØØ);delay(15);
[5N4] 1Ø14:   sound(4ØØ);delay(2Ø); nosound;
[ZNØ] 1Ø15: end;
[ØØØ] 1Ø16:
[247] 1Ø17: (Sound-FX wenn man 5 Credits bekommen hat)
[ØØØ] 1Ø18:
[P15] 1Ø19: procedure gamedlg.SFX_Credit;
[MUØ] 1Ø2Ø: begin
[H44] 1Ø21:   if soundeff = Ø then exit;
[WU5] 1Ø22:   sound(742 div (3-Spieler));  delay(2Ø);
[U43] 1Ø23:   nosound; delay(7Ø);
[1P5] 1Ø24:   sound(883 div (3-Spieler)); delay(2Ø);
[U43] 1Ø25:   nosound; delay(7Ø);
[7T5] 1Ø26:   sound(14Ø9 div (3-Spieler)); delay(2Ø);
[3G1] 1Ø27:   nosound;
[ZNØ] 1Ø28: end;
[ØØØ] 1Ø29:
[566] 1Ø3Ø: (Sound-FX wenn das Spiel vorbei ist)
[1A5] 1Ø31: procedure gamedlg.SFX_Gameovr;
[MUØ] 1Ø32: begin
[H44] 1Ø33:   if Soundeff = Ø then exit;
[333] 1Ø34:   sound(15Ø);delay(45);
[E13] 1Ø35:   sound(6Ø); delay(45);
[P23] 1Ø36:   sound(3Ø);delay(45);
[3G1] 1Ø37:   nosound;
[ZNØ] 1Ø38: end;
[ØØØ] 1Ø39:
[314] 1Ø4Ø: function gamedlg.check;
[RNØ] 1Ø41: var
[H82] 1Ø42:   i:      Integer;
[AM2] 1Ø43:   buf:    Fieldtyp;
[FE2] 1Ø44:   x:      Shortint;
[UP1] 1Ø45:   y:      Byte;
[4V2] 1Ø46:   moveok: Boolean;
[ØØØ] 1Ø47:
[MUØ] 1Ø48: begin
[YH2] 1Ø49:   moveok := true;
[PW4] 1Ø5Ø:   move(field,buf,sizeof(field));
[6G2] 1Ø51:   for i := 1 to 5 do
[MUØ] 1Ø52:   begin
[XW4] 1Ø53:     x := col+lo(fig[rot,i]) shl 1;
[G24] 1Ø54:     y := line+hi(fig[rot,i]);
[5X8] 1Ø55:     if (x > Ø) and (x < 2Ø) and (y < 21) an
       d (lo(field[y,x]) <> 219)
[7S7] 1Ø56:       then movechar(buf[y,x],#219,num+byte(nu
       m > 7),2)
[TE3] 1Ø57:       else moveok := false;
[ZNØ] 1Ø58:   end;
[AE1] 1Ø59:   if moveok
[DA8] 1Ø6Ø:   then if showf then gamefieldview^.showfie
       ld(buf);
[7L2] 1Ø61:   check := moveok;
[ZNØ] 1Ø62: end;
[ØØØ] 1Ø63:
[PS4] 1Ø64: constructor infoview.init;
[MUØ] 1Ø65: begin
[LA3] 1Ø66:   tview.init(bounds);
[ZL2] 1Ø67:   showinfo(Ø,Ø,Ø,Ø);
[ZNØ] 1Ø68: end;
[ØØØ] 1Ø69:
[815] 1Ø7Ø: procedure infoview.showinfo;
[MUØ] 1Ø71: begin
[1E4] 1Ø72:   sc := score; fl := fullines;
[F65] 1Ø73:   sl := startlevel; Cd := Credits;
[FUØ] 1Ø74:   draw;
[ZNØ] 1Ø75: end;
[ØØØ] 1Ø76:
[G84] 1Ø77: procedure infoview.draw;
[RNØ] 1Ø78: var
[UK1] 1Ø79:   r : trect;
[WB2] 1Ø8Ø:   dummy:String;
[MUØ] 1Ø81: begin
[Ø12] 1Ø82:   tview.draw;
[782] 1Ø83:   getextent(r);
[7R2] 1Ø84:   str(sc:5,dummy);
[4V4] 1Ø85:   writestr(r.a.x,r.a.y,dummy,1);
[8R2] 1Ø86:   str(fl:5,dummy);
[YY4] 1Ø87:   writestr(r.a.x,r.a.y+2,dummy,1);
[2S2] 1Ø88:   str(sl:5,dummy);
[4Z4] 1Ø89:   writestr(r.a.x,r.a.y+4,dummy,1);
[DN1] 1Ø9Ø:   if Twomode
[ZN1] 1Ø91:   then begin
[AP2] 1Ø92:         str(Cd:5,dummy);
[A24] 1Ø93:         writestr(r.a.x,r.a.y+6,dummy,1);
[ZNØ] 1Ø94:       end;
[ZNØ] 1Ø95: end;
[ØØØ] 1Ø96:
[ØØØ] 1Ø97:
[C95] 1Ø98: procedure fieldview.showfield;
[MUØ] 1Ø99: begin
[B32] 11ØØ:   field := data;
[FUØ] 11Ø1:   draw;
[ZNØ] 11Ø2: end;
[ØØØ] 11Ø3:
[AB4] 11Ø4: procedure fieldview.draw;
[RNØ] 11Ø5: var
[UK1] 11Ø6:   r : trect;
[MUØ] 11Ø7: begin
[782] 11Ø8:   getextent(r);
[K46] 11Ø9:   writebuf(r.a.x,r.a.y,r.b.x,r.b.y,field);
[ZNØ] 111Ø: end;
[ØØØ] 1111:
[KU4] 1112: constructor nextview.init;
[N32] 1113: var n : Byte;
[MUØ] 1114: begin
[LA3] 1115:   tview.init(bounds);
[FZ1] 1116:   shownext(Ø);
[ZNØ] 1117: end;
[ØØØ] 1118:
[355] 1119: procedure nextview.shownext;
[MUØ] 112Ø: begin
[M82] 1121:   Nextfig := Nt;
[FUØ] 1122:   draw;
[ZNØ] 1123: end;
[ØØØ] 1124:
[KZ6] 1125: (Diese Prozedur zeigt das Teil an, welches
[ØE7] 1126: als nächstes herunterfällt. Die Felder Fi
[GS5] 1127: und Fi_Ext enthalten die Icons der
[S84] 1128: verschiedenen Figuren )
[ØØØ] 1129:
[EA4] 113Ø: procedure nextview.draw;
[6XØ] 1131: const
[5N4] 1132:   Fi : array[1..1Ø,1..4] of Word =
[763] 1133:        (($Ø1ØØ,$Ø1db,$Ø1db,$Ø1ØØ),
[V53] 1134:         ($Ø2dc,$Ø2db,$Ø2df,$Ø2ØØ),
[363] 1135:         ($Ø3df,$Ø3db,$Ø3dc,$Ø3ØØ),
[693] 1136:         ($Ø4dc,$Ø4dc,$Ø4dc,$Ø4dc),
[863] 1137:         ($Ø5dc,$Ø5dc,$Ø5db,$Ø5ØØ),
[K63] 1138:         ($Ø6dc,$Ø6db,$Ø6dc,$Ø6ØØ),
[S63] 1139:         ($Ø7db,$Ø7dc,$Ø7dc,$Ø7ØØ),
[873] 114Ø:         ($Ø9db,$Ø9df,$Ø9db,$Ø9ØØ),
[W93] 1141:         ($Øadf,$Øadb,$Øadf,$ØaØØ),
[FA3] 1142:         ($Øbdb,$ØbØØ,$ØbØØ,$ØbØØ));
[ØØØ] 1143:
[UG5] 1144:   Fi_ext : array[9..1Ø,1..4] of Word =
[583] 1145:           (($ØaØØ,$Øadc,$ØaØØ,$ØaØØ),
[PG3] 1146:            ($Øbdc,$Øbdc,$ØbdcØ,$ØbØØ));
[RNØ] 1147: var
[UK1] 1148:   r : trect;
[ØD4] 1149:   Patt : array[1..18] of Word;
[MUØ] 115Ø: begin
[Ø12] 1151:   tview.draw;
[782] 1152:   getextent(r);
[LW4] 1153:   fillchar(Patt,sizeof(Patt),Ø);
[F22] 1154:   if Nextfig > Ø
[P26] 1155:   then move(Fi[Nextfig,1],Patt[8],8);
[MK8] 1156:   if Nextfig > 8 then move(Fi_ext[Nextfig],
       Patt[2],8);
[B36] 1157:   writebuf(r.a.x,r.a.y,r.b.x,r.b.y,Patt);
[ZNØ] 1158: end;
[ØØØ] 1159:
[SZ3] 116Ø: begin (Hauptprogramm)
[BX2] 1161:   writeln(E_zeit);
[ØR1] 1162:   main.init;
[TH1] 1163:   main.run;
[WN1] 1164:   main.done;
[DB3] 1165:   textbackground(red);
[3X2] 1166:   textcolor(black);
[2Ø3] 1167:   writeln('              ');
[ZB3] 1168:   writeln('     Ende!    ');
[6P2] 1169:   write  ('              ');
[ZG4] 117Ø:   textcolor(white);writeln;
[KNØ] 1171: end.
```

Der Tetris-Clone »tvtetris.pas« ist mit Turbo Vision geschrieben.

Kleine Spiele

Die Spieleprogrammierung muß nicht zwangsläufig in meterlange Programme ausarten, wie die folgenden Seiten zeigen. Das Motto dieser Rubrik lautet: »In der Kürze liegt die Würze«. So kommt es weniger auf die grafische Gestaltung eines Spiels an als vielmehr auf den Spielwitz, den Sie auch mit »Bildschirm-Klötzchen« erreichen können.

Richtig geschoben

Beim Schiebespiel »quuzzle.c« (Listing 1) verschieben Sie die Steine auf dem Spielbrett in aufsteigender Reihenfolge. Sie beginnen links oben und ordnen Zeile für Zeile bis zur letzten. Mit den Funktionstasten [F3] bis [F8] wählen Sie die Größe des Spielfelds von 33 bis 88 Kästchen Kantenlänge. Mit [Esc] beenden Sie das Spiel jederzeit.

Die Steuerung erfolgt mit den Cursortasten. Wie in der Brettversion verschieben Sie die Kästchen um das Leerfeld, nicht das Leerfeld selbst.

Ziel des Spiels ist es, das Spielfeld in wenigen Zügen zu restaurieren. Die Anzahl der Züge wird rechts neben dem Spielfeld dargestellt. Einen ungewollten Zug nehmen Sie einfach zurück, indem Sie den Stein wieder zurückschieben.

Um schnell zum Ziel zu gelangen, gehen Sie am besten zeilenweise von oben vor. Bei den größeren Spielfeldern sammeln Sie erst möglichst viele Steine zusammen und schieben sie dann geschlossen in eine Zeile.

(Marc Palms/bm)

```
[W6Ø]    1: /*
[UF3]    2:  Programm: quuzzle.c
[VX2]    3:  Funktion: Puzzle
[ZY4]    4:  Sprache:  C (diverse Compiler)
[H23]    5:  Autor:    Marc Palms
[HN4]    6:  (c) 1992 DMV Widuch GmbH & Co. KG
[17Ø]    7: */
[2R2]    8: #include"conio.h"
[SZ2]    9: #include"stdlib.h"
[PK2]   1Ø: #define v while
[61C]   11: #define u(r,s,l)if(c==r)if(l){l--;z++;f=s;}
            if(c==s)if(l+l<n){l++;z++;f=r;}
[9ED]   12: #define k if(27==(e=getch()))goto q;else if
            (!e){c=getch();if(6Ø<c&&c<67){n=c-58;goto r
            ;}
[7W6]   13: int a[64],n=4,b,c,g,h,d,e,f,t,x,y,z,time();
[FØ2]   14: char*s="              ";
[AE4]   15: w(int t,int x,int y,char*h)
[4H6]   16: {textattr(t);gotoxy(x,y);cputs(h);}
[V81]   17: p(int x)
[R35]   18: {g=13-n*3/2+x/n*3; h=35-n*7/2+x%n*7;
[UØ4]   19: if(x=e=a[x]) e=31+((x-1)%n+
[RB2]   2Ø:   (x-1)/n)%(n-1)*16;
[2R4]   21: s[2]=s[3]=s[4]=' ';w(e,h,g,s);
[MW1]   22: w(e,h,g+2,s);
[436]   23: s[3-(x>9)]=48+x/1Ø; s[3+(x>9)]=48+x%1Ø;
[KW1]   24: w(e,h,g+1,s);
[VE4]   25: if(t>b) w(142,h,g+1," Ready!");}
[UW0]   26: main()
[K72]   27: {randomize();
[7SØ]   28:  v(1){
[5M5]   29: r:w(t=z=a[x=y=b=n*n-1]=Ø,d=1,1,"");
[ØE1]   3Ø:   clrscr();
[HW3]   31:   v(y--) {a[y]=y+1;p(y);}
[ØK1]   32:   v(b*4>=y++)
[KF6]   33:    {c=a[f=random(b-n)]; a[f]=a[f+n]; p(f);
[T23]   34:    a[f+n]=c; p(f+n);}
[375]   35:   do{w(t=Ø,1,1,"");k x=(e=x)%n; y=e/n;
[MB4]   36:      if(c==f&&d==z) z-=2; u(77,75,x);
[GA1]   37:      u(8Ø,72,y)
[YV5]   38:      x+=y*n; a[e]=a[x];p(e); a[x]=Ø;p(x);
[SX4]   39:      (a[t]>t++); w((t>b)?79:12Ø,67,13,"");
[7R3]   4Ø:      cprintf(" Züge: %d ",z);
[ZV1]   41:      if(z>d)d=z;}
[663]   42:    } v(t<=b); p(b); k)}
[GE3]   43: q:w(7,1,1,"");clrscr();}
```

Listing 1. Das interessante Schiebespiel »quuzzle.c«

Mastermind

Das C-Programm »ramory.c« (Listing 2) ist eine Variante des bekannten Masterminds, wobei Ihnen der Computer eine vierstellige Zahlenfolge vorgibt, die Sie mit möglichst wenigen Versuchen erraten sollen. Die Zahlenfolge besteht aus den Ziffern 1 bis 6. Der Computer bewertet Ihre Eingabe mit einem ausgefüllten Block für eine richtige Zahl an der richtigen Stelle oder einem grauen Block für eine richtige Zahl, die aber falsch positioniert ist.

Als kleine Mogelhilfe erhalten Sie mit der F1-Taste die richtige Zahl an der gerade bearbeiteten Stelle. Mit [Esc] verlassen Sie das Programm.

(Marc Palms/bm)

```
[W6Ø]    1: /*
[B53]    2:  Programm: ramory.c
[XK3]    3:  Funktion: Mastermind
[RV1]    4:  Sprache:  C
[H23]    5:  Autor:    Marc Palms
[HN4]    6:  (c) 1992 DMV Widuch GmbH & Co. KG
[17Ø]    7: */
[2R2]    8: #include"conio.h"
[SZ2]    9: #include"stdlib.h"
[1F4]   1Ø: #define f(x)for(t=Ø;t<x;t++)
[PK2]   11: #define v while
[U73]   12: #define k break;case
[LL6]   13: typedef int z[4],j;j n,i,c,t,b,time();
[VM1]   14:   z q,r,x,y;
[AC5]   15: char*h="┘ \Ø┌─┐\Ø┌──┐\Ø┌  ┐\Ø┌─┐\Ø┌─ \
            Ø │ \Ø└─┘\Ø└──┤\Ø└──┼\Ø└──┤\Ø├─┤\Ø ┴ \Ø
            └──┘\Ø   ┴\Ø└──┘\Ø└──┘";
[TP6]   16: a(j t,j x,j y) {textattr(t);gotoxy(x,y);}
[XE1]   17: s(j i,j o)
[4B5]   18: {f(3){a((i+q[i])<<(4*o),i*6+2Ø,21+t);
[636]   19: putch(32);cputs(&h[5*(t*6+q[i])]);}}
[ØX2]   2Ø: p() {j t;f(4)s(t,Ø);}
[CS3]   21: m(j o) {p();s(i=(i+o)&3,1);}
[UW0]   22: main()
[K72]   23: {randomize();
[196]   24: do{a(n=Ø,Ø,Ø);clrscr();f(4){q[t]=r[t];
[5U2]   25: r[t]=random(6);}
[5N4]   26: do{a(Ø,Ø,23);v(t++<11)putch(1Ø);
[BV4]   27: a(7+(n>5),7,22);cprintf("%5d",++n);
[Z71]   28: m(b=i=Ø);
[LX6]   29: do{switch(c=getch()){k 27:f(4)q[t]=r[t];
[3A6]   3Ø: k Ø:switch(c=getch()){k 77:m(1);k 75:m(-1);
[8U3]   31: k 59:q[i]=r[i];m(1);}}
[735]   32: if(48<c&&c<55){q[i]=c-49;m(1);}
[VW1]   33: }v(c>27);p();
[5P5]   34: f(4)if((x[t]=q[t])==(y[t]=r[t]))
[SE3]   35: {b++;x[t]=y[t]=7;}
[BV2]   36: a(7+((c=b)&4)*32,51,22);
[LØ4]   37: v(Ø<b--)cputs("■■  ");
[GK3]   38: f((i=4));{v(x[t]<7&&Ø<i--)
[GY5]   39: if(x[t]==y[i]){cputs("▒▒  ");
[1X2]   4Ø: x[t]=y[i]=7;}}
[K61]   41: }v(c<4);
[RC5]   42: }v(getch()!=27);a(7,Ø,Ø);clrscr();}
```

Listing 2. Eine Mastermind-Variante in C zeigt »ramory.c«.

Gegeneinander

»spring.c« (Listing 3) ist ein kleines Spiel für zwei Personen. Die Regeln sind einfach: Jeder Spieler muß in jeder Runde auf eines der 36 Felder einen Stein setzen. Der erste Spieler hat die Zahlen, der zweite die Buchstaben. In der linken oberen Ecke wird angezeigt, wer am Zug ist (»B« für Buchstaben, »Z« für Zahlen). Gesetzt werden darf auf ein leeres oder ein dem Spieler gehörendes Feld. Mit dem Ziffernblock ([Num] aktiviert) steuern Sie die Felder an, mit [Enter] setzen Sie einen Stein. Die Zahlen und Buchstaben auf den Feldern zeigen an, wie viele Steine auf dem Feld liegen (»A« für 1, »B« für 2 und so weiter). Wenn nun auf einem Feld so viele Steine liegen, wie es gerade angrenzende Felder hat – in der Ecke sind das zwei, am Rand drei und in der Mitte vier –, so ist es »voll«. Dann springen die Steine des Feldes auf die Nachbarfelder. Dort liegende gegnerische Steine werden bei diesem Vorgang »gekapert«. Da eines der Zielfelder auch wieder »voll« werden kann, entstehen auf diese Weise Kettenreaktionen. Gewonnen hat, wer alle gegnerischen Steine vom Brett räumt.

Am besten, Sie probieren ein paar Runden aus, dann bekommen Sie ein Gefühl für das Spiel. Noch ein Tip: Besetzen Sie die Eckfelder zuerst, sie sind am schnellsten voll. Auch Randfelder sind wichtig.

(Jens Woinowski/bm)

```
[W6Ø]    1: /*
[M43]    2:  Programm: spring.c
[2F5]    3:  Funktion: Spiel für 2 Personen
[RV1]    4:  Sprache:  C
[7X3]    5:  Autor:    Jens Woinowski
[HN4]    6:  (c) 1992 DMV Widuch GmbH & Co. KG
[17Ø]    7: */
[ØØØ]    8:
[L13]    9: #include <stdio.h>
[ØR2]   1Ø: #include <bios.h>
[2Ø3]   11: #include <conio.h>
[NY2]   12: #include <alloc.h>
[ØØØ]   13:
[GW4]   14: int feld[6][6],sp=1,zahl[3];
[ØØØ]   15:
[L81]   16: loesch()
[P7Ø]   17: {
[5B1]   18:   int i,j;
[ØE1]   19:   clrscr();
[1N2]   2Ø:   for(i=6;i<19;i+=2)
[8R2]   21:     for(j=28;j<53;j++)
[P7Ø]   22:     {
[HY1]   23:       GoToXY(j,i);
[SU1]   24:       putch('=');
[U7Ø]   25:     }
[8Y2]   26:   for(i=28;i<53;i+=4)
[9G2]   27:     for(j=6;j<19;j++)
[P7Ø]   28:     {
[LY1]   29:       GoToXY(i,j);
[5U1]   3Ø:       putch('¤');
[U7Ø]   31:     }
[VR2]   32:   memset(feld,Ø,72);
[B63]   33:   zahl[Ø]=zahl[2]=Ø;
[U7Ø]   34: }
[Z53]   35: druck(int x, int y)
[P7Ø]   36: {
[913]   37:   GoToXY(x*4+3Ø, y*2+7);
[4F2]   38:   if(feld[x][y])
[P7Ø]   39:   {
[6S2]   4Ø:     if(feld[x][y]<Ø)
[WR3]   41:       putch(64-feld[x][y]);
```

Listing 3. »spring.c« ist ein Spiel für zwei Personen.

```
[ZR0]   42:      else
[MR3]   43:        putch(48+feld[x][y]);
[U70]   44:    }
[ZR0]   45:    else
[VH1]   46:      putch(32);
[U70]   47: }
[B53]   48: setzen(int x, int y)
[P70]   49: {
[EA1]   50:   int t=2;
[N24]   51:   if((x>=0)&&(y>=0)&&(x<6)&&(y<6))
[P70]   52:   {
[F93]   53:     if(feld[x][y]*sp<0)
[182]   54:       --zahl[1-sp];
[VC3]   55:     if(feld[x][y]*sp<=0)
[L72]   56:       ++zahl[1+sp];
[WR4]   57:     feld[x][y]= feld[x][y]<0?
[RL3]   58:       (-feld[x][y])*sp+sp :
[LZ3]   59:       feld[x][y]*sp+sp;
[8Y1]   60:     druck(x,y);
[9Z0]   61:     if(x%5)
[2L0]   62:       ++t;
[AZ0]   63:     if(y%5)
[2L0]   64:       ++t;
[XB6]   65:     while((feld[x][y]*sp>=t)&&zahl[1-sp])
[P70]   66:     {
[344]   67:       if((feld[x][y]-t*sp)==0)
[N32]   68:         --zahl[sp+1];
[8Y1]   69:       druck(x,y);
[KG4]   70:       setzen(x-1,y); setzen(x+1,y);
[KG4]   71:       setzen(x,y-1); setzen(x,y+1);
[U70]   72:     }
[U70]   73:   }
[U70]   74: }
[L5]    75: spiel()
[P70]   76: {
[MT2]   77:   int x=0,y=0,r=0,t;
[9C1]   78:   loesch();
[NB0]   79:   do
[P70]   80:   {
[GV1]   81:     GoToXY(1,1);
[U41]   82:     if(sp>0)
[XT1]   83:       printf("Z");
[ZR0]   84:     else
[DS1]   85:       printf("B");
[NB0]   86:     do
[P70]   87:     {
[913]   88:       GoToXY(x*4+3, y*2+7);
[5N2]   89:       switch(t=getch())
[P70]   90:       {
[XZ3]   91:         case '8': y=(y+5)%6; break;
[7Z3]   92:         case '2': y=(y+7)%6; break;
[6Z3]   93:         case '4': x=(x+5)%6; break;
[GZ3]   94:         case '6': x=(x+7)%6; break;
[1R4]   95:         case 13: if(sp*feld[x][y]>=0)
[GP3]   96:          setzen(x,y),sp=-sp,r++;
[0W2]   97:          else putch(7),t=0;
[U70]   98:       }
[U70]   99:     }
[1Z1]  100:     while(t!=13);
[U70]  101:   }
[NX4]  102:   while((zahl[sp+1]!=0)||(r<2));
[U70]  103: }
[UW0]  104: main()
[P70]  105: {
[E71]  106:   spiel();
[U70]  107: }
```

Listing 3. »spring.c« ist ein Spiel für zwei Personen.

Spielerparadies

Wer kennt sie nicht, die Augenblicke, in denen man nichts Neues am Rechner beginnen will, da eine andere Aktion noch nicht abgeschlossen ist. So versorgt zum Beispiel der Spooler den Drucker, der leise vor sich »hinnadelt«. Das ist die Zeit für ein kleines Spielchen zwischendurch. »Larry« zu laden wäre zu umständlich und zu langwierig.

Wollen Sie für kurze Zeit die Luft von Las Vegas schnuppern und sich mit einem »einarmigen Banditen« vergnügen, laden Sie »rolfomat.pas« (Listing 4), die Simulation eines Spielautomaten. Da er sich an seinen amerikanischen Vorbildern orientiert, gibt er dementsprechend Gewinn und Verlust in Dollar an. Wenn Ihnen dies nicht »vaterländisch« genug ist, können Sie natürlich im Quelltext das schnöde $-Zeichen in »DM« umwandeln. Eine Spielrunde kostet jeweils 10 Dollar. Bei drei gleichen Farben gewinnen Sie 100 Dollar, bei zwei gleichen 20 Dollar.

Gibt der Automat von links nach rechts eine aufsteigende Farbfolge (von hell nach dunkel) an, gibt es zumindest ein Freispiel.

Nach der Begrüßungsmelodie, die das Flair des »großen Geldes« vermittelt, »drehen« sich die Laufräder des Spielautomaten unter Sound und Lauflicht. Durch Drücken der Tasten [1], [2] oder [3] des alphanumerischen Tastenblocks halten Sie das entsprechende Laufrad an. Aber Vorsicht! Wird eine andere Taste als eine der drei vorher erwähnten gedrückt, führt dies automatisch zum Verlust der Spielrunde. Wählen Sie dagegen mehrfach dieselbe Taste an, so bleibt dies ohne Folgen – der »Rolfomat« meckert kurz, indem er dreimal piepst, und läuft einfach weiter. Haben Sie alle drei Walzen gewählt, kommt es je nach Gewinn oder Verlust zu einer Neuanzeige des Spielerkapitals, das einen Startwert in Höhe von 100 Dollar hat. Danach erscheint eine Statuszeile, die abfragt, ob Sie abbrechen oder weiterspielen möchten.

Das Spiel wurde mit Turbo Pascal 6.0 kompiliert und für ein System mit einer Hercules-Karte geschrieben. Es läuft aber auch auf Computern mit einer CGA-Karte; nur müssen Sie hier die Prozedur »CursorOn« wegen der unterschiedlichen Auflösung etwas verändern – die entsprechende Zeile muß dann

```
R.AH := 1; R.CH := 8; R.CL := 9;
INTR($10,R);
```

lauten. Diese Routine ist vielleicht neben der Prozedur »Ton« auch interessant für andere Programme, da sich mit ihr die Cursorform verändern läßt. In bezug auf eine Hercules-Karte ergibt

```
R.CH := 1; R.CL := 12;
```

einen großen Block-Cursor,

```
R.CH := 2; R.CL := 11;
```

einen dezenteren Block-Cursor und

```
R.CH := 12; R.CL := 1;
```

einen Cursor aus Ober- und Unterstrich. Verwenden Sie eine CGA-Karte, sollten Sie beachten, daß nicht 12, sondern nur neun Zeilen zur Cursoranzeige zur Verfügung stehen. Die anderen Prozeduren bedürfen keiner weiteren Erklärung, da sie im Listing so weit wie nötig dokumentiert sind. Verbesserungen ergeben sich, wenn Sie ein Color-System besitzen und Farbe ins Spiel bringen. Auch könnten Sie bei einem Gewinnstand von 1000 Dollar eine Extra-Melodie spielen lassen – vielleicht »Money« von Pink Floyd. Beim Erreichen der Null-Dollar-Marke bietet sich »Üb' immer Treu und Redlichkeit« an.

(Rolf Olland/bm)

```
[Z64]    1: {Programm: rolfomat.pas
[ZM4]    2:  Funktion: Simulation eines
[493]    3:                Geldspielautomaten
[KF4]    4:  Sprache:  ab Turbo Pascal 4.0
[S63]    5:  Autor:    Rolf Olland
[585]    6:  (c) 1992 DMV Widuch GmbH & Co. KG }
[000]    7:
[923]    8: PROGRAM Autspiel;
```

```
[000]    9:
[X82]   10: USES CRT, DOS;
[000]   11:
[RF4]   12: const Kapital: INTEGER = 100;
[1A5]   13: var Taste: CHAR; Frequenz: REAL;
[AF6]   14:   Stop1, Stop2, Stop3, Verloren: BOOLEAN;
[YL5]   15:   Halbton, Rad1, Rad2, Rad3, Ergebnis1,
[G95]   16:   Ergebnis2, Ergebnis3, i: INTEGER;
[000]   17:
[ZK7]   18: procedure CursorOff; {schaltet Cursor aus}
[TU2]   19: VAR R: Registers;
[MU0]   20: BEGIN
[5P5]   21:   R.AH:=1; R.CH:=15; R.CL:=0; INTR($10,R);
[ZN0]   22: END;
[557]   23: PROCEDURE CursorOn; {schaltet Cursor an}
[TU2]   24: VAR R: Registers;
[MU0]   25: BEGIN
[BS5]   26:   R.AH:=1; R.CH:=11; R.CL:=12; INTR($10,R);
[ZN0]   27: END;
[DL7]   28: PROCEDURE Ton(Oktav,Halbton,Dauer:INTEGER);
[F07]   29: {Routine zum Spielen von Tönen, die in
[RN6]   30:  ihrer Oktavlage, Tonhöhe und Tondauer
[2L4]   31:  festgelegt werden können.}
[1K2]   32: VAR j: INTEGER;
[MU0]   33: BEGIN
[FZ2]   34:   Frequenz := 130.815;
[YB3]   35:   FOR j := 1 TO Oktav DO
[CY3]   36:     Frequenz := Frequenz * 2;
[EZ3]   37:     FOR j := 1 TO Halbton - 1 DO
[EA5]   38:       Frequenz := Frequenz * 1.059463094;
[686]   39:   SOUND(ROUND(Frequenz)); DELAY(Dauer);
[3G1]   40:   NOSOUND;
[ZN0]   41: END;
[NW2]   42: PROCEDURE Dallas;
[MU0]   43: BEGIN
[7L5]   44:  Ton(1,8,150); Ton(2,1,450); Ton(1,8,150);
[EK5]   45:  Ton(2,8,450); Ton(2,1,150); Ton(2,5,300);
[XH5]   46:  Ton(2,3,150); Ton(2,5,150); Ton(2,1,450);
[CP5]   47:  Ton(1,8,150); Ton(2,1,300); Ton(2,10,300);
[MK5]   48:  Ton(2,8,300); Ton(2,5,150); Ton(2,6,150);
[FU1]   49:  Ton(2,8,450);
[ZN0]   50: END;
[XF3]   51: PROCEDURE Dreiklang;
[MU0]   52: BEGIN
[GH5]   53:   Ton(2,1,200); Ton(2,5,200); Ton(2,8,200);
[AT1]   54:   Ton(3,1,600);
[ZN0]   55: END;
[UK3]   56: PROCEDURE Glissando;
[1K2]   57: VAR j: INTEGER;
[MU0]   58: BEGIN
[414]   59:   FOR j := 2000 DOWNTO 200 DO
[MU0]   60:   BEGIN
[CC4]   61:     DELAY(1); SOUND(j); DELAY(1);
[N85]   62:     SOUND(j * 3); DELAY(1); SOUND(200);
[ZN0]   63:   END;
[3G1]   64:   NoSound;
[ZN0]   65: END;
[F27]   66: PROCEDURE Print(y,x: BYTE; Satz: STRING);
[MU0]   67: BEGIN
[M64]   68:   GOTOXY(x,y); WRITE(Satz);
[ZN0]   69: END;
[TZ2]   70: PROCEDURE Invers;
[MU0]   71: BEGIN
[6E5]   72:   TEXTCOLOR(0); TEXTBACKGROUND(15);
[ZN0]   73: END;
[T73]   74: PROCEDURE Zurueck;
[FV7]   75: {schaltet auf den Normal-Bildschirm zurück}
[MU0]   76: BEGIN
[XB5]   77:   TEXTCOLOR(15); TEXTBACKGROUND(0);
[EL1]   78:   LOWVIDEO;
[ZN0]   79: END;
[6U2]   80: PROCEDURE Maske;
[MU0]   81: BEGIN
[743]   82:   CLRSCR; HIGHVIDEO;
[EZ1]   83:   GOTOXY(20,1);
[3G5]   84:   WRITELN('(c) 1992 DMV Widuch GmbH & '
[AB1]   85:   + 'Co. KG');
[GS4]   86:   LOWVIDEO; WRITELN; WRITELN;
[2DC]   87:   WRITELN('╔═══════════════════════╗');
[516]   88:   WRITELN('║   R O L F O M A T     ║'+
[YH6]   89:   'Zuerst laufen  alle drei Räder. Die');
[HZ4]   90:   WRITELN('║                       ║'+
[NS5]   91:   'Eingabe von 1  stoppt Rad1,  eine 2');
[E86]   92:   WRITELN('║  ▄   ▄   ▄   ▄   ▄    ║'+
[L06]   93:   'hält Rad2 an,... Die anderen beiden');
[HZ4]   94:   WRITELN('║                       ║'+
[V56]   95:   'laufen weiter,  bis die Eingabe der');
[FS5]   96:   WRITELN('║   D E L U X E         ║'+
[6P6]   97:   'übrigen Radzahlen  auch sie stoppt.');
[HZ4]   98:   WRITELN('║                       ║'+
[V76]   99:   'Der 3. Durchlauf entscheidet alles.');
[FD4]  100:   WRITELN('╚═══════════════════════╝');
[HZ4]  101:   WRITELN('                         '+
[2P5]  102:   'Drei Farben gleich:     $100 Gewinn');
[D35]  103:   WRITELN('┌──┬──┬──┬──┬──┐          '+
[PP5]  104:   'Zwei Farben gleich:     $ 20 Gewinn');
[MA8]  105:   WRITELN('│  │  │  │  │  │          '+
[K96]  106:   'Aufsteigende Farbfolge:   Freispiel');
[2DC]  107:   WRITELN('└──┴──┴──┴──┴──┘');
[ZN0]  108: END;
[U27]  109: PROCEDURE Anfang; {Begrüßungsmelodie}
[MU0]  110: BEGIN
[2G5]  111:   CursorOff; RANDOMIZE; LOWVIDEO;
[LG3]  112:   Maske; Dallas; Dallas;
[ZN0]  113: END;
[7R2]  114: PROCEDURE Init;
[2H6]  115: {setzt Spiel nach drei Durchläufen auf
[ED7]  116:  die Parameter für einen neuen Spielbeginn}
[MU0]  117: BEGIN
[KY4]  118:   Stop1 := FALSE; Stop2 := FALSE;
[GG5]  119:   Stop3 := FALSE; Verloren := FALSE;
[5S3]  120:   {10 $ für jedes Spiel}
[YK3]  121:   Kapital := Kapital - 10;
[EW1]  122:   {Rad-Marken anneuern}
[073]  123:   PRINT (13,6,'1    2    3');
[UR2]  124:   PRINT (10,8,'          ');
[Y45]  125:   GOTOXY(8,10); WRITE('$ ',Kapital);
[MH7]  126:   {löscht die Statuszeile - 51 Leerspalten}
[PY3]  127:   PRINT (22,1,'                                                  '
[DC2]  128:     + '                  ');
[YV2]  129:   {79 Leerspalten}
[UY3]  130:   PRINT (24,1,'                                                  '
[E53]  131:     + '                  ');
[UU2]  132:     + '                  '); i := 1;
[KZ2]  133: END; {PROCEDURE}
[MP2]  134: PROCEDURE Lauf;
[MU0]  135: BEGIN
[P31]  136:   REPEAT
[YR5]  137:     IF NOT Stop1 THEN PRINT (7,6,'▀');
[YH4]  138:     PRINT (11,6,'·'); Ton(1,1,50);
[M52]  139:     PRINT (11,6,' ');
[YB5]  140:     IF NOT Stop2 THEN PRINT (7,11,'▒');
[AC5]  141:     IF NOT Stop3 THEN PRINT (7,16,'▀');
[AK4]  142:     PRINT (11,8,'·'); Ton(1,3,50);
[W52]  143:     PRINT (11,8,' ');
[UK5]  144:     IF NOT Stop1 THEN PRINT (7,6,'▒');
[ZB5]  145:     IF NOT Stop2 THEN PRINT (7,11,'▀');
[UV4]  146:     PRINT (11,10,'·'); Ton(1,5,50);
[K92]  147:     PRINT (11,10,' ');
[MD5]  148:     IF NOT Stop3 THEN PRINT (7,16,'▒');
[PK5]  149:     IF NOT Stop1 THEN PRINT (7,6,'▒');
[ZV4]  150:     PRINT (11,12,'·'); Ton(1,6,50);
[M92]  151:     PRINT (11,12,' ');
```

```
[ØC5]  152:      IF NOT Stop2 THEN PRINT (7,11,'▓');
[9C5]  153:      IF NOT Stop3 THEN PRINT (7,16,'▒');
[YV4]  154:      PRINT (11,14,'·'); Ton(1,5,5Ø);
[P92]  155:      PRINT (11,14,' ');
[KK5]  156:      IF NOT Stop1 THEN PRINT (7,6,'▒');
[9D5]  157:      IF NOT Stop2 THEN PRINT (7,11,'▓');
[UV4]  158:      PRINT (11,16,'·'); Ton(1,3,5Ø);
[S92]  159:      PRINT (11,16,' ');
[BC5]  16Ø:      IF NOT Stop3 THEN PRINT (7,16,'▓');
[K13]  161:    UNTIL KEYPRESSED;
[HR2]  162:    Taste := READKEY;
[GN4]  163:    IF (Taste = '1') AND Stop1 THEN
[MUØ]  164:    BEGIN
[KW1]  165:      WRITE(#7#7#7);
[BVØ]  166:      Lauf;
[ZNØ]  167:    END;
[PN4]  168:    IF (Taste = '2') AND Stop2 THEN
[MUØ]  169:      BEGIN
[KT2]  17Ø:        WRITE(#7#7#7); Lauf;
[ZNØ]  171:      END;
[WN4]  172:    IF (Taste = '3') AND Stop3 THEN
[MUØ]  173:      BEGIN
[KT2]  174:        WRITE(#7#7#7); Lauf;
[ZNØ]  175:      END;
[ZNØ]  176: END;
[TZ2]  177: PROCEDURE Zufall;
[MUØ]  178: BEGIN
[6G5]  179:   Rad1:=RANDOM(4)+1; Rad2:=RANDOM(4)+1;
[XS2]  18Ø:   Rad3:=RANDOM(4)+1;
[WW1]  181:   CASE Rad1 OF
[CM5]  182:     1:IF NOT Stop1 THEN PRINT (7,6,'▓');
[WH5]  183:     2:IF NOT Stop1 THEN PRINT (7,6,'▓');
[WH5]  184:     3:IF NOT Stop1 THEN PRINT (7,6,'▓');
[L66]  185:     ELSE IF NOT Stop1 THEN PRINT (7,6,'▒');
[ZNØ]  186:   END;
[YW1]  187:   CASE Rad2 OF
[9X5]  188:     1:IF NOT Stop2 THEN PRINT (7,11,'▓');
[GT5]  189:     2:IF NOT Stop2 THEN PRINT (7,11,'▓');
[FT5]  19Ø:     3:IF NOT Stop2 THEN PRINT (7,11,'▓');
[4B6]  191:    ELSE IF NOT Stop2 THEN PRINT (7,11,'▒');
[ZNØ]  192:   END;
[ØX1]  193:   CASE Rad3 OF
[ØV5]  194:     1:IF NOT Stop3 THEN PRINT (7,16,'▓');
[7U5]  195:     2:IF NOT Stop3 THEN PRINT (7,16,'▓');
[6U5]  196:     3:IF NOT Stop3 THEN PRINT (7,16,'▓');
[NB6]  197:    ELSE IF NOT Stop3 THEN PRINT (7,16,'▒');
[ZNØ]  198:   END;
[ZNØ]  199: END;
[WB6]  2ØØ: PROCEDURE Abfrage; {Setzen von Flags }
[MUØ]  2Ø1: BEGIN
[Z12]  2Ø2:   CASE Taste OF
[B81]  2Ø3:   '1':BEGIN
[3V4]  2Ø4:         Stop1:=TRUE; Ergebnis1:=Rad1;
[S52]  2Ø5:         PRINT (13,6,' ');
[ZNØ]  2Ø6:       END;
[E81]  2Ø7:   '2':BEGIN
[9V4]  2Ø8:         Stop2:=TRUE; Ergebnis2:=Rad2;
[R92]  2Ø9:         PRINT (13,11,' ');
[ZNØ]  21Ø:       END;
[H8]   211:   '3':BEGIN
[FV4]  212:         Stop3:=TRUE; Ergebnis3:=Rad3;
[W92]  213:         PRINT (13,16,' ');
[ZNØ]  214:       END;
[WP1]  215:     ELSE BEGIN
[KD5]  216:       WRITE(#7#7#7); {bei Fehlbedienung}
[PB4]  217:       Ergebnis1:=4; Ergebnis2:=2;
[EY4]  218:       Ergebnis3:=1; Verloren:=TRUE;
[PV9]  219:       i:=1Ø;
[ZNØ]  22Ø:     END;
[692]  221:   END; {CASE}
[K21]  222:   i:=i+1;
[ZNØ]  223: END;
[1G5]  224: PROCEDURE Zuweisung; {Zuweisungen}
[MUØ]  225: BEGIN
[KW3]  226:   IF (Stop1 AND Stop2) THEN
[6U2]  227:     Ergebnis3 := Rad3;
[PW3]  228:   IF (Stop1 AND Stop3) THEN
[2U2]  229:     Ergebnis2 := Rad2;
[SW3]  23Ø:   IF (Stop2 AND Stop3) THEN
[YT2]  231:     Ergebnis1 := Rad1;
[K23]  232:   PRINT (13,6,'           ');
[ZNØ]  233: END;
[1Z5]  234: PROCEDURE Resultat; {Auswertung}
[MUØ]  235: BEGIN
[ZF4]  236:   IF (Ergebnis1=Ergebnis2) AND
[MB4]  237:     (Ergebnis1=Ergebnis3) THEN
[MUØ]  238:   BEGIN
[HU3]  239:     PRINT (22,1,'Du hast 1ØØ $ '
[B92]  24Ø:       + 'gewonnen!');
[N75]  241:     Kapital:=Kapital+1ØØ; Dreiklang;
[4MØ]  242:   END
[HN5]  243:   ELSE IF (Ergebnis1 < Ergebnis2) AND
[BE4]  244:     (Ergebnis2 < Ergebnis3) THEN
[MUØ]  245:   BEGIN
[YN4]  246:     PRINT (22,1,'Du hattest ein '+
[X62]  247:       'Freispiel!');
[285]  248:     Kapital:=Kapital+1Ø; Dreiklang;
[4MØ]  249:   END
[7A5]  25Ø:   ELSE IF (Ergebnis1=Ergebnis2) OR
[PZ3]  251:     (Ergebnis1=Ergebnis3) OR
[RB4]  252:     (Ergebnis2=Ergebnis3) THEN
[MUØ]  253:   BEGIN
[NT3]  254:     PRINT (22,1,'Du hast 2Ø $ '
[B92]  255:     + 'gewonnen!');
[585]  256:     Kapital:=Kapital+2Ø; Dreiklang;
[4MØ]  257:   END
[MM3]  258:   ELSE IF Verloren THEN
[MUØ]  259:   BEGIN
[UR1]  26Ø:     HIGHVIDEO;
[X66]  261:     PRINT (22,1,'Du mußt die richtigen '
[MV4]  262:       + 'Tasten drücken: 1, 2 oder 3!');
[EL1]  263:     LOWVIDEO;
[4MØ]  264:   END
[DW1]  265:   ELSE  BEGIN
[F15]  266:    PRINT (22,1,'Du hast verloren!');
[EC3]  267:    Glissando; DELAY(5ØØ);
[ZNØ]  268:   END;
[MS2]  269:   PRINT (1Ø,8,'        ');
[Y45]  27Ø:   GOTOXY(8,1Ø); WRITE('$ ',Kapital);
[WE7]  271:   {Statuszeile: Weiterspielen oder Abbruch}
[2D1]  272:   Invers;
[2F6]  273:   PRINT (24,1,'Drücke eine Taste, wenn du '
[N46]  274:    + 'weiter spielen willst - ESC beendet '
[CY2]  275:    + 'das Programm...');
[RA4]  276:   Zurueck; Taste := READKEY;
[ZNØ]  277: END;
[SZ3]  278: BEGIN {Hauptprogramm}
[291]  279:   Anfang;
[P31]  28Ø:   REPEAT
[AVØ]  281:     Init;
[P31]  282:     REPEAT
[RL3]  283:       Lauf; Zufall; Abfrage;
[TW6]  284:     UNTIL i>2; {Auswertung des Fehlercodes}
[MK3]  285:     IF i > 1Ø THEN Resultat
[DW1]  286:     ELSE  BEGIN
[962]  287:       Lauf; Zufall;
[BN3]  288:       Zuweisung; Resultat;
[ZNØ]  289:     END;
[3U2]  29Ø:   UNTIL Taste = #27;
[123]  291:   CLRSCR; CursorOn;
[KNØ]  292: END.
```

Listing 4. »rolfomat.pas« simuliert einen Geldspielautomaten.

Tron-Clone

Das Programm »vector.c« (Listing 5) wurde in Turbo C + + entwickelt und stellt ein kleines, aber dennoch nicht einfaches Geschicklichkeitsspiel dar. Der Computer und der Spieler steuern je einen Rennwagen mit gleichen Eigenschaften. Wenn nun diese Rennwagen losfahren, hinterlassen sie hinter sich eine Mauer. Das Ziel des Spiels ist es nun, einfach länger durchzuhalten als der Computer, ohne gegen eine Mauer zu fahren. Dabei stellen Sie sich den Spielplan von der Vogelperspektive aus vor. Weiß ist der Spieler, blau der Computer. Sie steuern den Rennwagen genauso wie ein normales Auto, das heißt, man kann nur nach links oder nach rechts fahren. Genau dies übernimmt die linke beziehungsweise rechte Cursortaste.

Wem das Spiel zu leicht oder zu schwer erscheint, der kann den Schwierigkeitsgrad ändern, indem er der Variablen »de« ganz am Anfang des Quelltextes einen niedrigeren beziehungsweise höheren Wert zuweist. In Zeile 12 geben Sie den Namen des Pfads ein, in dem der entsprechende BGI-Treiber vorhanden ist.

Es ist zu empfehlen, das Spiel von der integrierten Entwicklungsumgebung aus zu starten und nicht als Exe-Datei aus der DOS-Kommandoebene, weil dadurch die Ausgaben der Grafik auf dem Bildschirm flüssiger ablaufen.

(Timo Vogel/bm)

```
[DL3]   1: /*  Programm : vector.c
[ZP5]   2:     Funktion : Geschicklichkeitsspiel
[A23]   3:     Sprache  : Turbo C++
[NK3]   4:     Autor    : Timo Vogel  */
[ØØØ]   5:
[ØE3]   6: #include <graphics.h>
[9L2]   7: #include <dos.h>
[ØØØ]   8:
[UW0]   9: main()
[FV5]  1Ø:  { int x, y, a, d, cx, cy, tg, de=7, lo, r;
[864]  11:    int wi, ri, ric, e=DETECT;
[Ø15]  12:    initgraph(&e,Ø,"\\tc\\bgi\\");
[Z33]  13:    while ( inp(96) != 49 )
[N53]  14:    { clearviewport();
[XY1]  15:      setcolor(2);
[HG5]  16:      rectangle (Ø,Ø,getmaxx(),getmaxy());
[CS1]  17:      lo=Ø; ri=2;
[C62]  18:      if( x%2==Ø ) r=1;
[9L1]  19:      else r=-1;
[ND5]  2Ø:      y=getmaxy()/2; x=3Ø; a=getmaxx()-x;
```

```
[ØP2]  21:      d=y; ric=4; wi=Ø;
[D93]  22:      while ( lo==Ø && wi==Ø )
[RS3]  23:      { if ( inp(96)==75 && tg==Ø )
[ØK2]  24:          { ri--; tg=1; }
[BL3]  25:        if ( inp(96)==77 && tg==Ø )
[PH2]  26:          { ri++; tg=1; }
[9A4]  27:        if ( inp(96)!=75 && inp(96)!=77 )
[RSØ]  28:          tg=Ø;
[FG2]  29:        if ( ri==5 ) ri=1;
[6G2]  3Ø:        if ( ri==Ø ) ri=4;
[AK1]  31:        delay(de);
[NZ1]  32:        switch ( ri )
[DØ3]  33:        { case 4:x--; break;
[9R2]  34:          case 2:x++; break;
[HR2]  35:          case 1:y--; break;
[AZ2]  36:          case 3:y++; break; }
[4A4]  37:        if ( getpixel (x,y) != Ø ) lo=1;
[FX2]  38:        putpixel (x,y,15);
[U15]  39:        if ( getpixel (cx,cy) != Ø ) ric-=r;
[DU2]  4Ø:        if ( ric==5 ) ric=1;
[5U2]  41:        if ( ric==Ø ) ric=4;
[142]  42:        switch ( ric )
[7Z4]  43:        { case 4:a--; cx=a-5; cy=d; break;
[DT4]  44:          case 2:a++; cx=a+5; cy=d; break;
[RP4]  45:          case 1:d--; cx=a; cy=d-5; break;
[245]  46:          case 3:d++; cx=a; cy=d+5; break; }
[D74]  47:        if ( getpixel (a,d) != Ø ) wi=1;
[GN2]  48:        putpixel (a,d,9);
[U7Ø]  49:      }
[K83]  5Ø:      if (wi==1) outtextxy
[7H4]  51:           (27,12,"Sie haben gewonnen !!");
[UN2]  52:      else outtextxy
[SK4]  53:           (27,12,"Sie haben verloren !!");
[DT1]  54:      outtextxy
[CH4]  55:           (27,27,"Noch ein Spiel (J/N)");
[ET4]  56:      while ( inp(96)!=36 && inp(96)!=49 );
[U7Ø]  57:    }
[S42]  58:    closegraph();
[CE1]  59:    return Ø;
[U7Ø]  6Ø:  }
```

Listing 5. »vector.c« stellt ein kleines, aber nicht einfaches Geschicklichkeitsspiel dar.

Die Turbo-Schlange

Das Programm »snake.pas« (Listing 6), ein Remake des bekannten Schlangenspiels, zeigt auf eindrucksvolle Weise, wie einfach sich mit Turbo Vision ansprechende Applikationen im SAA-Look erzeugen lassen (Bild 1). Die Unit »usnake.pas« (Listing 7) wurde in erster Linie entwickelt, um den Desktop des mitgelieferten Demoprogramms »tvdemo.pas« um ein weiteres Spiel zu bereichern.

Um das Schlangenspiel in den Desktop einzufügen, nehmen Sie folgende Änderungen in »tvdemo.pas« vor:

1. Einbinden der Unit »USnake«,
2. Deklarieren der Konstante für den Befehl »CONST cmSnake = 150;«,
3. in der Typdeklaration von »TTVDemo« Angabe eines zusätzlichen Feldes »Data« vom Typ »Word« (zur Übergabe der Geschwindigkeit),
4. vor »TApplication.HandleEvent« ist die nachfolgende Startprozedur zur Verfügung zu stellen:

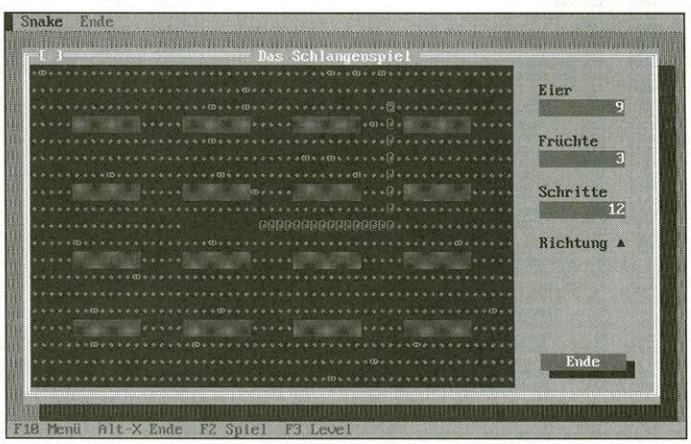


Bild 1. Das Schlangenspiel im CUA-Look

```
procedure Snake(Data:Word);
var S:PSnakeDlg;
begin
Data:=2; {Speed}
S:=New(PSnakeDlg,Init(Data));
if ValidView(S)<>nil then begin
Desktop+.ExecView(S);
dispose(S,Done);
end;
end;
```

5. In der Prozedur »TApplication.HandleEvent« ist die Aktion für »cmSnake« zuzuordnen:

```
cmSnake: Snake(Data);
```

6. Schließlich muß eine Gelegenheit zum Aufruf des Spiels geschaffen werden, zum Beispiel im ersten Untermenü von »TTVDemo.InitMenuBar«:

```
NewItem('Schlangenspiel', '', kbNoKey,
cmSnake, hcNoContext,
```

(Zusätzliche Klammer nach NIL nicht vergessen!)

Nach erfolgreicher Kompilierung (gegebenenfalls mit TPC) bleiben nach dem Start des Programms ohne die integrierte Entwicklungsumgebung über 400 KByte Arbeitsspeicher frei – ein Wert, der zusätzliche Erweiterungen geradezu herausfordert.

Wenn Sie das Spiel unabhängig von »TVDemo« benutzen möchten, geben Sie »snake.pas« ein. Es wird dann eine eigenständige Applikation erzeugt, die über einen zusätzlichen Dialog die Veränderung des Schwierigkeitsgrades erlaubt.

(Peter Fliesges/bm)

```
{Programm: snake.pas
 Funktion: Schlangenspiel
 Sprache:  Turbo Pascal 6.0
 Autor:    Peter Fliesges
 (c) 1992 DMV Widuch GmbH & Co. KG }

{$X+}
program Snake;

uses Objects,Menus,Views,Drivers,App,
     Dialogs,USnake;

const
  cmSnake      = 100;
  cmLevel      = 101;

type
  TMyApp=object(TApplication)
    Schlange: PSnakeDlg;
    DlgData: Word;
    procedure InitMenuBar; virtual;
    procedure InitStatusLine; virtual;
    procedure HandleEvent(var Event:
      TEvent); virtual;
  end;

var
  MyApp:TMyApp;

{--- TMyApp ---}

procedure TMyApp.InitMenuBar;
var R:TRect;
begin
  R.Assign(0,0,80,1);
  MenuBar:=New(PMenuBar,Init(R,NewMenu(
    NewSubMenu('~S~nake', hcNoContext,
      NewMenu(
      NewItem('~S~piel','F2',kbF2,cmSnake,
        hcNoContext,
      NewLine(
      NewItem('~L~evel','F3',kbF3,cmLevel,
        hcNoContext,Nil)))),
    NewItem('~E~nde','',kbESC,cmQuit,
      hcNoContext,Nil))))));
end;

procedure TMyApp.InitStatusLine;
var R: TRect;
begin
  R.Assign(0,24,80,25);
  StatusLine:=New(PStatusLine,Init(R,
    NewStatusDef(0,0,
    NewStatusKey('~F10~ Menü',kbF10,cmMenu,
    NewStatusKey('~Alt-X~ Ende',kbAltX,
      cmQuit,
    NewStatusKey('~F2~ Spiel',kbF2,cmSnake,
    NewStatusKey('~F3~ Level',kbF3,cmLevel,
      nil)))),
    nil))));
end;

constructor TMyApp.Init;
begin
  TApplication.Init;
  DisableCommands([cmSnake,cmLevel]);
end;

destructor TMyApp.Done;
begin
  TApplication.Done;
end;

procedure TMyApp.HandleEvent(Event:TEvent);
begin
  TApplication.HandleEvent(Event);
  if Event.What=evCommand then
    case Event.Command of
      cmSnake : Schlange:=New(PSnakeDlg,Init(DlgData));
      cmLevel : begin
                  DlgData:=DlgData+1;
                  if DlgData>3 then DlgData:=0;
                end;
    end;
end;

var MyApp:TMyApp;
begin
  MyApp.Init;
  MyApp.Run;
  MyApp.Done;
end.
```

Listing 6. Das Schlangenspiel in Turbo Vision zeigt »snake.pas«

```
{Programm: usnake.pas
 Funktion: Unit für Schlangenspiel
 Sprache:  Turbo Pascal 6.0
 Autor:    Peter Fliesges
 (c) 1992 DMV Widuch GmbH & Co. KG }

unit USnake;

{$F+,O+,X+,S-,D-}

interface

uses
  Drivers,Objects,App,Views,Dialogs,Crt;

type
  FieldType = array[1..19,1..57] of Word;

  SnakeArrType = array[0..2000] of Word;

  PFieldView=^TFieldView;
  TFieldView=object(TView)
    Field:FieldType;
    procedure MakeField(Matrix:FieldType);
    procedure Draw;virtual;
  end;

  PScoreView=^TScoreView;
  TScoreView=object(TView)
    Eggs,Fruits,Steps:Word;
    Dir:Char;
    procedure SetScore(EggC,FruitC,
      StepC:Word;SDir:Char);
    procedure Draw; virtual;
  end;

  PSnakeDlg=^TSnakeDlg;
  TSnakeDlg=object(TDialog)
    SnakeFieldView:PFieldView;
    ScoreView:PScoreView;
    GameOverView:PStaticText;
    BravoView:PStaticText;
    Field:FieldType;
    Ln,Col:Byte;
    Speed:Word;
    Quelle, Ziel : ^Word;
    SnakeLen,SnakePos,EggCount,FruitCount,
      StepCount: Word;
    SDir:Char;
    GameOver,AllEaten:Boolean;
    SnakeArr:SnakeArrType;

    Ticker:^LongInt;
    mTicker:LongInt;
    constructor Init(Data:Word);
    procedure InitField;
    procedure InitVar;
    procedure HandleEvent
      (var Event:TEvent);virtual;
    procedure TestMove;
    procedure MoveSnake;
    procedure Crash;
  end;

implementation

{--- Snake Dialog ---}

constructor TSnakeDlg.Init;
var
  R:TRect;
begin
  R.Assign(1,1,76,22);
  TDialog.Init(R,'Das Schlangenspiel');
  R.Assign(60,18,72,20);
  Insert(New(PButton,Init(R,'~E~nde',
    cmCancel,bfDefault)));
  R.Assign(1,1,58,20);
  SnakeFieldView:=New(PFieldView,Init(R));
  Insert(SnakeFieldView);
  R.Assign(61,2,71,12);
  ScoreView:=New(PScoreView,Init(R));
  Insert(ScoreView);
  R.Assign(16,8,43,13);
  GameOverView:=
    New(PStaticText,Init(R,#13#13^C+
      'Das war'+#39+'s mal wieder'));
  R.Assign(16,8,43,13);
  BravoView:=
    New(PStaticText,Init(R,#13#13^C+
      'Gut gemacht!'));
  Options:=Options or ofCentered;
  InitVar;
  InitField;
  Speed:=Data;
end;

procedure TSnakeDlg.InitField;
var
  anz,n,z,s:Byte;
begin
  Fillchar(Field,SizeOf(Field),0);
  for z := 1 to 19 do
    for s:=1 to 57 do Field[z,s]:=256*3+7;
  Randomize;
  anz:=Random(20)+35;
  for n:=1 to anz do
  begin
    z:=Random(19)+1;
    s:=Random(57)+1;
    Field[z,s]:=256*12+236;
  end;
  for z:=1 to 4 do
    for s:=1 to 4 do
      for n:=0 to 7 do
        Field[z*4,s*13+n-7]:=256*19+177;
  for n := 0 to 19 do
  begin
    if n=0 then
      Field[Hi(SnakeArr[n]),
        Lo(SnakeArr[n])]:=256*9+2
    else
      Field[Hi(SnakeArr[n]),
        Lo(SnakeArr[n])]:=256*9+64;
  end;
  SnakeFieldView^.MakeField(Field);
end;

procedure TSnakeDlg.InitVar;
var
  n:Byte;
begin
  Randomize;
  EggCount     := 0;
  FruitCount   := 0;
  Col          := 38;
  Ln           := 10;
  SnakePos     := 256*Ln+Col;
  SDir         := 'O';
  SnakeLen     := 20;
  Ticker       :=Ptr($40,$6C);
  mTicker      :=Ticker^;
  GameOver     :=False;
  AllEaten     :=False;
  for n := 0 to 19 do
    SnakeArr[n] := SnakePos-n;
end;

procedure TSnakeDlg.HandleEvent;
var
  n:Byte;
begin
  TDialog.HandleEvent(Event);
  if not GameOver then
  begin
    if Event.What=EvKeyDown then
    begin
      case Event.KeyCode of
        $4B00 : SDir:='W';
        $4D00 : SDir:='O';
        $4800 : SDir:='N';
        $5000 : SDir:='S';
      end;
    end;
    if Ticker^ - Speed > mTicker then
      TestMove;
    ClearEvent(Event);
    If GameOver then begin
      if AllEaten then
        Insert(BravoView)
      else
        Insert(GameOverView);
      for n:=1 to 10 do begin
        sound(500*n);
        Delay(3);
        NoSound;
        Delay(3);
      end;
    end;
  end;
end;

procedure TSnakeDlg.TestMove;
var
  z,s:Byte;
begin
  case SDir of
    'N' : Dec(Ln);
    'S' : Inc(Ln);
    'O' : Inc(Col);
    'W' : Dec(Col);
  end;
  SnakePos := 256*Ln + Col;
  if (Ln<1) or (Ln>19) or (Col<1)
    or (Col>57) or (Hi(Field[Ln,Col])=19)
    or (Hi(Field[Ln,Col])=9) then
    Crash
  else begin
    if (Hi(Field[Ln,Col])=3) then
    begin
      Sound(250);
```

```
[XA1]  201:          Delay(3);
[3G1]  202:          NoSound;
[UC1]  203:          inc(EggCount);
[ZN0]  204:        end;
[DR4]  205:      if (Hi(Field[Ln,Col])=12) then
[MU0]  206:        begin
[DS1]  207:          Sound(750);
[HF1]  208:          Delay(300);
[3G1]  209:          NoSound;
[8U2]  210:          inc(FruitCount);
[ZN0]  211:        end;
[DR2]  212:      AllEaten:=True;
[2S2]  213:      GameOver:=True;
[FL2]  214:      for z:=1 to 19 do
[3L2]  215:        for s:=1 to 57 do
[5N3]  216:          if (Hi(Field[z,s])=12) or
[6P4]  217:            (Hi(Field[z,s])=3) then begin
[ZS2]  218:            AllEaten:=False;
[MT2]  219:            GameOver:=False;
[ZN0]  220:          end;
[YT1]  221:      MoveSnake;
[UP2]  222:      inc(StepCount);
[975]  223:      ScoreView^.SetScore(EggCount,
[MW4]  224:        FruitCount,StepCount,SDir);
[ZN0]  225:    end;
[ZN0]  226: end;
[000]  227:
[2B5]  228: procedure TSnakeDlg.MoveSnake;
[MU0]  229: begin
[PP5]  230:   Field[Hi(SnakePos),Lo(SnakePos)]:=
[S51]  231:     256*9 + 2;
[PF4]  232:   Quelle := Addr(SnakeArr[0]);
[Z84]  233:   Ziel   := Addr(SnakeArr[1]);
[AG5]  234:   Move(Quelle^,Ziel^,SnakeLen *2 );
[244]  235:   SnakeArr[0] := SnakePos;
[KU6]  236:   Field[Hi(SnakeArr[1]),Lo(SnakeArr[1])]:=
[U31]  237:     256*9+64;
[S64]  238:   if StepCount Mod 5 = 0 then
[H52]  239:     inc(SnakeLen)
[ZR0]  240:   else
[165]  241:     Field[Hi(SnakeArr[SnakeLen]),
[AF5]  242:       Lo(SnakeArr[SnakeLen])]:=256*1+32;
[HR5]  243:   SnakeFieldView^.MakeField(Field);
[223]  244:   mTicker:=Ticker;
[ZN0]  245: end;
[000]  246:
[SH4]  247: procedure TSnakeDlg.Crash;
[RN0]  248: var
[E81]  249:   n:Byte;
[MU0]  250: begin
[2S2]  251:   GameOver:=True;
[HA7]  252:   ScoreView^.SetScore(EggCount,FruitCount,
[YT2]  253:     StepCount,SDir);
[B94]  254:   for n:=30 DownTo 1 do begin
[R92]  255:     Sound(n*2+100);
[DN1]  256:     Delay(100);
[K81]  257:     Dec(n,2);
[3G1]  258:     NoSound;
[ZN0]  259:   end;
[ZN0]  260: end;
[000]  261:
[335]  262: {--- FieldView (Spielfeld) ---}
[000]  263:
[ZF5]  264: procedure TFieldView.MakeField;
[MU0]  265: begin
[ED2]  266:   Field:=Matrix;
[FU0]  267:   Draw;
[ZN0]  268: end;
[000]  269:
[8L4]  270: procedure TFieldView.Draw;
[RN0]  271: var
[KE1]  272:   R:TRect;
[MU0]  273: begin
[782]  274:   GetExtent(R);
[R26]  275:   WriteBuf(R.A.X,R.A.Y,R.B.X,R.B.Y,Field)
[ZN0]  276: end;
[000]  277:
[W45]  278: {--- ScoreView (Spielstand) ---}
[000]  279:
[5C5]  280: procedure TScoreView.SetScore;
[MU0]  281: begin
[SS4]  282:   Eggs:=EggC; Fruits:=FruitC;
[924]  283:   Steps:=StepC; Dir:=SDir;
[FU0]  284:   Draw;
[ZN0]  285: end;
[000]  286:
[0N4]  287: procedure TScoreView.Draw;
[RN0]  288: var
[8L1]  289:   S:String;
[S71]  290:   c:Byte;
[MU0]  291: begin
[0Y3]  292:   WriteStr(0,0,'Eier      ',1);
[0F2]  293:   Str(Eggs:10,S);
[ZZ2]  294:   WriteStr(0,1,S,12);
[5H3]  295:   WriteStr(0,2,'          ',1);
[BB4]  296:   WriteStr(0,3,'Früchte   ',1);
[UX2]  297:   Str(Fruits:10,S);
[203]  298:   WriteStr(0,4,S,12);
[8H3]  299:   WriteStr(0,5,'          ',1);
[DD4]  300:   WriteStr(0,6,'Schritte  ',1);
[GM2]  301:   Str(Steps:10,S);
[503]  302:   WriteStr(0,7,S,12);
[BH3]  303:   WriteStr(0,8,'          ',1);
[DA4]  304:   WriteStr(0,9,'Richtung ',1);
[1X1]  305:   case Dir of
[YG1]  306:     'N': c:=30;
[FH1]  307:     'O': c:=16;
[GH1]  308:     'S': c:=31;
[AK1]  309:     'W': c:=17;
[ZN0]  310:   end;
[2P3]  311:   WriteStr(9,9,char(c),1);
[ZN0]  312: end;
[000]  313:
[KN0]  314: end.
```

Listing 7. Die Unit »usnake.pas« benötigen Sie für das Schlangenspiel.

Schiffe versenken

Das Basic-Programm »versenke.bas« (Listing 8) ist eine Version des bekannten Papierspiels »Schiffe versenken«. Es läuft mit den Sprachen Turbo Basic, Power Basic und QBasic.

Nach dem Start werden vier Schiffe für Sie unsichtbar auf dem Spielfeld plaziert. Diese sind jeweils zwei, drei, vier und fünf Kästchen groß. Ihre Aufgabe besteht nun darin, durch Angabe der Koordinaten Schüsse abzugeben und möglichst schnell alle Schiffe zu treffen.

Falls Sie das Spiel mittendrin beenden möchten, verlassen Sie es mit [Q]. Wenn Sie alle Schiffe versenkt haben, beendet sich das Programm automatisch.

(Marc Stimm/bm)

```
[T14]    1: ' Programm: versenke.bas
[2Y4]    2: ' Funktion: Schiffe versenken
[T53]    3: ' Sprache:  PB, QB, TB
[TC3]    4: ' Autor:    Marc Stimm
[ZV4]    5: ' (c) 1992 DMV Widuch GmbH & Co. KG
[000]    6:
[6M5]    7: CLS : DEFINT A-Z: DIM p(255, 99), s(5):
[US2]    8: RANDOMIZE TIMER
[V72]    9: FOR y = 1 TO 10
[8U3]   10:   FOR x = 1 TO 10: p(x, y) = 1
[ZS0]   11:   NEXT
[ZS0]   12: NEXT
[X12]   13: FOR s = 2 TO 5
[HL0]   14: no:
[YU2]   15: x = INT(RND * 10) + 1
[0V2]   16: y = INT(RND * 10) + 1
[YS2]   17: xp = INT(RND * 2) - 1
[0T2]   18: yp = INT(RND * 2) - 1
[525]   19: IF xp <> 0 AND yp <> 0 THEN GOTO no
[EN4]   20: IF xp = 0 AND yp = 0 THEN GOTO no
[B42]   21: FOR n = 1 TO s
[7F5]   22:   IF p(x+xp*n,y+yp*n) <> 1 THEN GOTO no
[921]   23: NEXT n
[B42]   24: FOR n = 1 TO s
[695]   25:   p(x + xp * n, y + yp * n) = -s: NEXT n
[S21]   26: NEXT s
[DV0]   27: such:
[NN1]   28: GOSUB bild
[1A3]   29: LOCATE 22, 65: INPUT a$
[LH3]   30: IF a$="q" then GOTO Ende
[GF2]   31: a$ = UCASE$(a$) + "x"
[TU2]   32: x = ABS(ASC(a$) - 64)
[8K5]   33: y = ABS(VAL(MID$(a$, 2, 2))): p = p(x, y)
[BM5]   34: IF p > 1 OR p = 0 THEN BEEP: GOTO such
[NS1]   35: IF p < 0 THEN
[2P3]   36: p = ABS(p): s(p) = s(p) + 1
[DP1]   37: p(x, y) = p
[504]   38: FOR t = 999 TO 499 STEP -39
[CL1]   39:   SOUND t, 1
[L62]   40: NEXT 'treffer
[801]   41: END IF
[FF5]   42: IF p = 1 THEN SOUND 200, 6: p(x, y) = 0
[YX2]   43: V = V + 1: GOTO such
[7U0]   44: bild:
[3W5]   45: LOCATE 1, 1: PRINT "Versuch ": PRINT V
[MS4]   46: FOR n = 1 TO 10: LOCATE 2, 6 + n * 5
[Z35]   47: PRINT CHR$(64 + n): LOCATE 2 + n * 2, 4
[062]   48: PRINT n: NEXT
[7E1]   49: COLOR 0, 7
[MB4]   50: FOR x = 1 TO 10: FOR y = 1 TO 10
[RD3]   51: y2 = 2 + y * 2: x2 = 5 + x * 5
[0F3]   52: LOCATE y2, x2: p = p(x, y)
[MG4]   53: IF p = 1 OR p < 0 THEN PRINT "   "
[WG3]   54: IF p = 0 THEN PRINT " X"
[WS1]   55: IF p > 1 THEN
[NA2]   56:   IF s(p) < p THEN
[AN2]   57:     PRINT " "; CHR$(173)
[ZR0]   58:   ELSE
[GA1]   59:     PRINT p
[801]   60:   END IF
[801]   61: END IF
[WS1]   62: NEXT: NEXT
[EE1]   63: COLOR 7, 0
[RY5]   64: FOR n = 2 TO 5: IF s(n) <> n THEN RETURN
[ZS0]   65: NEXT
[XT0]   66: Ende:
[6P2]   67: PRINT "Game over"
```

Listing 8. Mit »versenke.bas« versenken Sie Schiffe auf Ihrem PC.

Taktik mit Nim

Wahrscheinlich kennen Sie die einfachste und wohl älteste Art der Nim-Spiele, bei der die beiden Spieler abwechselnd von einem Haufen nach eigenem Ermessen entweder einen oder zwei Steine nehmen. Es verliert, wer die letzten ein oder zwei Steine entfernt. Die Bezeichnung »Nim« hat der amerikanische Mathematiker Charles L. Bouton eingeführt, der Anfang des 20. Jahrhunderts eine Abhandlung einiger Varianten nebst ihren optimalen Strategien veröffentlichte. Das englische Wort »nim« bedeutet soviel wie »stehlen, mopsen, stibitzen«. Und eben darum geht es in der Familie der Nim-Spiele.

Der Däne Piet Hein – bekannt als Erfinder des Somawürfels – steuerte eine ebenso leicht verständliche wie anspruchsvolle Spielart bei. Sein Spiel »Tactics« ist in dem Programm »tactics.c« (Listing 9) umgesetzt.

Bei Tactics werden abwechselnd Münzen von einem 44 Felder großen Spielfeld genommen. Sie können bei jedem Zug nur Münzen aus einer einzigen Reihe (oder Spalte) nehmen. Diese müssen außerdem zusammenhängend (ohne Lücke) liegen. Gewonnen haben Sie, wenn der Computer die letzte(n) Münze(n) nimmt.

Mit den Cursortasten fahren Sie die einzelnen Spielfelder an, markieren sie mit der Leerschritt-Taste und lösen den Zug mit der Enter-Taste aus. [Pos1] veranlaßt einen Neustart des Spiels, und mit [Ende] erreichen Sie einen Seitenwechsel. Mit [Esc] verlassen Sie Tactics jederzeit.

Im Programm wird die Spielstellung in der Variablen »unsigned b«(board) dargestellt. Bit 0 entspricht Münze 0 und so weiter.

Fehlt zum Beispiel die Münze 2, so dürfen Sie die übrigen Münzen der oberen Reihe wegen der Lücke nicht zusammen aufnehmen. Zulässig ist also hier das Entfernen einer beliebigen Münze allein oder des Paares 0/1.

Beim ersten Zug gibt es 16 Varianten, eine einzelne Münze zu entwenden, 24 Varianten für die Wegnahme eines Pärchens, 16 für das Entfernen von drei Münzen und acht für das Entfernen einer kompletten Reihe.

Die Bitmuster dieser insgesamt 64 Züge sind (nach Anzahl der Münzen aufsteigend sortiert) im Feld »unsigned z[64]« abgelegt. Das Makro »W()« dupliziert eine Zugvariante für die Initialisierung in waagerechter, und das Makro »S()« in senkrechter Richtung. Das Makro »v()« prüft, ob sich alle Steine, die für den Zug nötig sind, noch auf dem Brett befinden.

Die Funktion »y(i)« positioniert den Cursor auf Münze »i«. Den Ausdruck der Spielstellung besorgt die Funktion »p(m)«, wobei »m« das Bitmuster der markierten Steine enthält. Sämtliche Tastatureingabe-Funktionen werden durch Funktion »g()« erlaubt. »g()« verhindert das Markieren unzulässiger Züge. Die Hauptfunktion »main()« gestaltet den Spielablauf, lobt Sie, falls Sie gewinnen, oder läßt Sie weiterhin üben.

Das Herz des Programms ist die knapp 150 Zeichen umfassende rekursive Funktion »t(b)«. Nach dem Aufruf von »t(b)« wird ein noch denkbarer Zug aus dem Feld »z[]« ausgewählt, vom Brett probehalber entfernt und als Zwischenstellung an »t()« übergeben. »t()« ruft sich selbst so lange auf, bis das Brett leer ist, und kehrt dann mit der Anzahl der dazu benötigten Züge (»int d«) zurück. Ist diese Anzahl gerade, dann gewinnt der Rechner auf jeden Fall und stibitzt die entsprechenden Münzen. Das schließt aus, daß

sein Gegner daraufhin eine gewinnträchtige Zugfolge entwickeln kann. Die Variable »unsigned e« (Rekursionstiefe), die anfangs 0 enthält, wird vor jedem weiteren Aufruf um 1 erhöht und danach um 1 erniedrigt. So ermittelt der Rechner, ob er für sich (e ist gerade) oder für seinen Gegner (e ist ungerade) rechnet. Die zweite Bedingung, um gewinnen zu können, ist also: Wenn e ungerade ist, dann darf bei exaktem Spiel keine gerade Zugtiefe vorhanden sein.

Der Ausdruck »(d + e)&1« ist für Gewinn gleich 1. Wenn diese Bedingung durch keinen der noch möglichen Züge erfüllt wird, dann wird zufällig (pseudozufällig – Stand der Systemuhr s) eine der Münzen entfernt. Da die Berechnung aller Züge recht lange dauern kann, wird nach 40 »timerticks« (à 55 ms, also nach 2,2 Sekunden) die Funktion zum Abbruch gezwungen.

Wenn Ihnen das Programm noch zu schwach spielt, können Sie die zeitliche Abbruchbedingung »(*w-s>40)« durch Ersetzen von 40 durch eine größere Zahl noch hinauszögern. »w« enthält übrigens die Adresse der Systemuhr im BIOS-Variablensegment 0040:006Chex, verkleidet als »1132L«.

Tactics ist für Turbo C ab Version 2.0 geschrieben. Zum Übersetzen benötigen Sie die Dateien »tcc.exe«, »tlink.exe«, »c0t.obj« und »cs.lib«. Die Übersetzung starten Sie mit:

```
tcc -mt -lt -w- -f- tactics
```

(Andreas Westfeld/bm)

```
[W6Ø]    1: /*
[BC3]    2: Programm: tactics.c
[YD3]    3: Funktion: NIM-Spiel
[WU2]    4: Sprache:  Turbo C
[174]    5: Autor:    Andreas Westfeld
[HN4]    6: (c) 1992 DMV Widuch GmbH & Co. KG
[17Ø]    7: */
[9D6]    8: unsigned s,b,x,far *w=11321,e=Ø,z[64]={
[ZU4]    9:  1,1<<4,1<<8,1<<12,2,2<<4,2<<8,2<<12,4,
[8P4]   1Ø:  4<<4,4<<8,4<<12,8,8<<4,8<<8,8<<12,3,
[3S4]   11:  3<<4,3<<8,3<<12,6,6<<4,6<<8,6<<12,12,
[PØ5]   12:  12<<4,12<<8,12<<12,17,17*2,17*4,17*8,272,
[CD4]   13:  272*2,272*4,272*8,4352,4352*2,4352*4,
[D35]   14:  4352*8,7,7<<4,7<<8,7<<12,14,14<<4,14<<8,
[RX4]   15:  14<<12,273,273*2,273*4,273*8,4368,4368*2,
[Y35]   16:  4368*4,4368*8,15,15<<4,15<<8,15<<12,4369,
[7W2]   17:  4369*2,4369*4,4369*8};
[ØØØ]   18:
[DV0]   19: y(i){
[3L4]   2Ø:   gotoxy(9+(i*4&12),8+(i/2&14));}
[ØØØ]   21:
[7U0]   22: t(b){
[XC1]   23:   int *q,d;
[ØØØ]   24:
[GL2]   25:   if(!b) return e+1;
[N33]   26:   if(*w-s>4Ø) return Ø;
[L73]   27:   for(q=z+63;q>=z;q--)
[RG2]   28:     if(!(b&*q)==*q)) {
[KFØ]   29:       e++;
[HK1]   3Ø:       d=t(b^*q);
[ØGØ]   31:       e--;
[WD4]   32:       if((d^e)&1) return e?d:*q;}
[ØE2]   33:       if(e) return e;
[G51]   34:       else {
[815]   35:         while(!((b&z[s&15])==z[s&15]))s++;
[5L2]   36:         return z[s&15];
[U7Ø]   37:       }
[U7Ø]   38: }
[BVØ]   39: p(m){
[DC1]   4Ø:   int i,t;
[L13]   41:   for(i=Ø;i<16;i++) {
[DRØ]   42:     y(i);
[KA1]   43:     t=b&1<<i;
[R27]   44: cputs(t?t&m?"\xB2\xB2":"\xBØ\xBØ":"\xFA ");
[U7Ø]   45:   }
[U7Ø]   46: }
[5HØ]   47: g(){
[UU1]   48:   int m,c,*h;
[ZZØ]   49:   m=x=Ø;
[6T5]   5Ø:   while((p(m),y(x),c=getch())!=13¦¦!m) {
[3T3]   51:   switch(c?c:256+getch()) {
[791]   52:   case 328:
[9M1]   53:     if(x>3)x-=4;
[PØ1]   54:     break;
[991]   55:   case 336:
[7T1]   56:     if(x<12)x+=4;
[PØ1]   57:     break;
[491]   58:   case 331:
[HH1]   59:     if(x>Ø)x--;
[PØ1]   6Ø:     break;
```

```
[691]   61:   case 333:
[WL1]   62:     if(x<15)x++;
[PØ1]   63:     break;
[Y21]   64:   case' ':
[ØD1]   65:     m^=1<<x;
[8U1]   66:     if(!m)break;
[YT1]   67:     if(((b&m)==m))
[583]   68:     for(h=z;h<&z[64];h++)
[4B2]   69:       if(m==*h) goto j;
[ØD1]   7Ø:     m^=1<<x;
[2AØ]   71: j:
[PØ1]   72:   break;
[691]   73:   case 327:
[PMØ]   74:     b=-1;
[PØ1]   75:     break;
[891]   76:   case 335:
[KC1]   77:     return;
[661]   78:   case 27:
[7TØ]   79:     y(16);
[961]   8Ø:     exit(Ø);
[U7Ø]   81:     }
[U7Ø]   82:   }
[NUØ]   83:   b^=m;
[U7Ø]   84: }
[ØØØ]   85:
[6C1]   86: main(){
[F91]   87:   for(;;){
[PMØ]   88:     b=-1;
[ØE1]   89:     clrscr();
[YL5]   9Ø:     puts("TACTICS - (C) 1992 DMV Widuch\
[ØØ2]   91:  GmbH & Co. KG");
[AE1]   92:     for(;;) {
[5FØ]   93:       g();
[RLØ]   94:       p(Ø);
[ZØ1]   95:       if(!b) {
[7TØ]   96:         y(16);
[XN2]   97:         puts("Üben!\a");
[PØ1]   98:         break;
[U7Ø]   99:       }
[KPØ]  1ØØ:       s=*w;
[V61]  1Ø1:       b^=t(b);
[ZØ1]  1Ø2:       if(!b) {
[7TØ]  1Ø3:         y(16);
[NP2]  1Ø4:         puts("Bravo!\a");
[PØ1]  1Ø5:         break;
[U7Ø]  1Ø6:       }
[ZK1]  1Ø7:       p(Ø);}}}
```

Listing 9. »tactics.c« ist eine Nimm-Variante.

Weltraumspiel

Bei dem in Turbo Pascal geschriebenen Computerspiel "sp__arrow.pas" (Listing 10) geht es um das Abschießen von Sternen. Sie bewegen Ihr Zielgerät mit dem numerischen Zehnerblock in acht Richtungen und schießen mit der Leertaste. Ihre verschossene Munition sowie Ihre Treffer werden im unteren Teil des Bildschirms angezeigt. Mit [E] brechen Sie das Programm jederzeit ab.

Die im Spiel verwendeten Prozeduren können auch in anderen, selbstgeschriebenen Spielen verwendet werden. Auch können Sie das Programm noch verbessern, indem Sie zum Beispiel eine Highscore-Liste oder einen Titelbildschirm hinzugefügen.

Die Geschwindigkeit verändern Sie mit einer Verzögerungsschleife in der Hauptschleife, wie zum Beispiel »delay(30)«. Dieses recht einfache Turbo-Pascal-Programm zeigt, daß auch mit wenigen Mitteln animierte Spielprogramme in einer Hochsprache geschrieben werden können.

(Christian Schneider/bm)

```
[AA4]    1: {Programm: sp_arrow.pas
[BB4]    2:  Funktion: Weltraumspiel
[HH4]    3:  Sprache:  Turbo Pascal ab 4.Ø
[DS4]    4:  Autor:    Christian Schneider
[585]    5:  (c) 1992 DMV Widuch GmbH & Co. KG }
[ØØØ]    6:
[PP3]    7: program Space_Arrow;
[5H1]    8: uses crt;
[YY4]    9: var punktzahl,durchlauf,x,y,
[P65]   1Ø:     anzahl_schuesse,x_tastatur,
[AM3]   11:     y_tastatur:integer;
[AR4]   12:     abbruch,punkt,ende:boolean;
[PL5]   13: procedure bewege_feindflugzeug;
[1G5]   14: var x_random,y_random:integer;
```

```
[MUØ]    15: begin
[1T1]    16:   randomize;
[FØ2]    17:   gotoxy(x,y);
[FM1]    18:   write(' ');
[MN3]    19:   x_random:=random(2)+1;
[PN3]    2Ø:   y_random:=random(2)+1;
[AY4]    21:    if x_random=1 then inc(x,1) else
[291]    22:       dec(x,1);
[CY4]    23:    if y_random=1 then inc(y,1) else
[491]    24:       dec(y,1);
[FE2]    25:   if x<2 then x:=2;
[7X2]    26:   if x>77 then x:=77;
[KE2]    27:   if y<2 then y:=2;
[8V2]    28:   if y>21 then y:=21;
[FØ2]    29:   gotoxy(x,y);
[5R2]    3Ø:   textcolor(blue);
[TM1]    31:   write('*');
[ZNØ]    32: end;
[RB4]    33: procedure punkt_gemacht;
[MUØ]    34: begin
[LP2]    35:   inc(punktzahl);
[842]    36:   gotoxy(1Ø,23);
[DØ3]    37:   textcolor(brown);
[3Ø3]    38:   write(punktzahl);
[LW3]    39:   bewege_feindflugzeug;
[ZNØ]    4Ø: end;
[E13]    41: procedure schuss;
[MUØ]    42: begin
[HU3]    43:   inc(anzahl_schuesse);
[DØ3]    44:   textcolor(brown);
[B42]    45:   gotoxy(11,24);
[734]    46:   write(anzahl_schuesse);
[GR1]    47:   sound(1ØØ);
[DN1]    48:   delay(1ØØ);
[3G1]    49:   nosound;
[LK6]    5Ø:   if (x_tastatur+1=x) and (y_tastatur=y)
[ZN1]    51:   then begin
[K52]    52:     punkt:=true;
[WR1]    53:     sound(2ØØØ);
[NN1]    54:     delay(3ØØ);
[SR1]    55:     sound(1ØØØ);
[XN1]    56:     delay(5ØØ);
[3G1]    57:     nosound;
[KK2]    58:     punkt_gemacht;
[ZNØ]    59:     end;
[ZNØ]    6Ø: end;
[KG3]    61: procedure tastatur;
[3Y1]    62: var c:char;
[MUØ]    63: begin
[HZ3]    64:   if keypressed then begin
[3X1]    65:    c:=readkey;
[BG5]    66:    gotoxy(x_tastatur,y_tastatur);
[NN1]    67:    write(' ');
[Y91]    68:    case c of
[G73]    69:      'e'    :abbruch:=true;
[NV3]    7Ø:      '8'    :dec(y_tastatur,1);
[7V3]    71:      '4'    :dec(x_tastatur,1);
[8W3]    72:      '2'    :inc(y_tastatur,1);
[KW3]    73:      '6'    :inc(x_tastatur,1);
[L12]    74:      ' '    :schuss;
[XK1]    75:      '7'    :begin
[933]    76:                dec(y_tastatur,1);
[733]    77:                dec(x_tastatur,1);
[ZNØ]    78:              end;
[3L1]    79:      '9'    :begin
[933]    8Ø:                dec(y_tastatur,1);
[C43]    81:                inc(x_tastatur,1);
[ZNØ]    82:              end;
[BK1]    83:      '1'    :begin
[733]    84:                dec(x_tastatur,1);
[E43]    85:                inc(y_tastatur,1);
[ZNØ]    86:              end;
[HK1]    87:      '3'    :begin
[C43]    88:                inc(x_tastatur,1);
[E43]    89:                inc(y_tastatur,1);
[ZNØ]    9Ø:              end;
[ZNØ]    91:    end;
[B66]    92:    if x_tastatur<2 then x_tastatur:=2;
[D86]    93:    if x_tastatur>77 then x_tastatur:=77;
[C66]    94:    if y_tastatur<2 then y_tastatur:=2;
[Y66]    95:    if y_tastatur>21 then y_tastatur:=21;
[BG5]    96:  gotoxy(x_tastatur,y_tastatur);
[NN2]    97:  textcolor(red);
[7X1]    98:  write('[ ]');
[MP1]    99:  end; {if}
[ZNØ]   1ØØ: end;
[LY2]   1Ø1: procedure Rahmen;
[UW2]   1Ø2: var x,y:integer;
[MUØ]   1Ø3: begin
[RY2]   1Ø4:   textcolor(green);
[GV1]   1Ø5:   gotoxy(1,1);
[5P1]   1Ø6:   write(#218);
[VS2]   1Ø7:   for x := 2 to 79 do
[EP1]   1Ø8:     write(#196);
[ZN1]   1Ø9:   write(#191);
[VZ1]   11Ø:   gotoxy(1,22);
[2P1]   111:   write(#192);
[VS2]   112:   for x := 2 to 79 do
[EP1]   113:     write(#196);
[2P1]   114:   write(#217);
[SK2]   115:   for y:=2 to 21 do
[MUØ]   116:   begin
[8Z1]   117:     gotoxy(1,y);
[LP1]   118:     write(#179);
[M42]   119:     gotoxy(8Ø,y);
[LP1]   12Ø:     write(#179);
[ZNØ]   121:   end;
[N42]   122:   gotoxy(35,24);
[M73]   123:   textcolor(yellow);
[W93]   124:   write('SPACE ARROW');
[M42]   125:   gotoxy(35,23);
[NN2]   126:   textcolor(red);
[944]   127:   write('Abbruch durch "e"');
[ØØ2]   128:   gotoxy(2,23);
[DØ3]   129:   textcolor(brown);
[ND3]   13Ø:   writeln('Punkte: Ø');
[5M3]   131:   write  (' Schüsse: Ø');
[ZNØ]   132: end;
[MUØ]   133: begin
[4B2]   134:   punktzahl:=Ø;
[KH3]   135:   anzahl_schuesse:=Ø;
[CL2]   136:   abbruch:=false;
[D12]   137:   ende:=false;
[1S2]   138:   x_tastatur:=4Ø;
[ZR2]   139:   y_tastatur:=2Ø;
[DB1]   14Ø:   clrscr;
[842]   141:   gotoxy(4Ø,2Ø);
[NN2]   142:   textcolor(red);
[7X1]   143:   write('[ ]');
[RW0]   144:   x:=4Ø;
[UW0]   145:   y:=12;
[M91]   146:   rahmen;
[P31]   147:   repeat
[P31]   148:    repeat
[LW3]   149:      bewege_feindflugzeug;
[G56]   15Ø:      for durchlauf:=1 to 1ØØØØ do tastatur;
[V95]   151:    until (punkt=true) or (ende=true)
[8Z2]   152:      or (abbruch=true);
[AE3]   153:   until abbruch=true;
[KNØ]   154:   end.
```

Listing 10. »sp__arrow.pas« verwickelt Sie in eine Weltraum-Ballerei.



Bild 2. Ein bunter Zeichensatz wertet Ihre Programme optisch auf.

Farbige Zeichensätze

Oft brauchen Sie bei der Programmierung von Spielen eigene schöne Zeichensätze, zum Beispiel für eine Anzeige des Spielstands oder Highscore. Mit dem Programm »fontdrv.pas« (Listing 11) gelingt Ihnen das Einbinden selbstdefinierter Zeichensätze in Turbo Pascal 6.0. In dieser Version funktioniert das Programm im VGA-Mode 13hex (320 × 200 Pixel, 256 Farben). Die Zeichen haben eine Größe von 16 × 16 Pixel und sind im Programm »fontpas0« (Listing 12) definiert.

Die Tabelle zeigt Ihnen die Definition des Buchstabens »S« im Binär- und Hexadezimalformat. »fontpas0« enthält keinen kompletten Zeichensatz, sondern nur einige Buchstaben für den Schriftzug »Score« (Bild 2) und Zahlen. Außerdem können Sie die Zeichen auch ein- und ausblenden (Prozedur »blend1«).

In der Prozedur »writefont1« werden die Zeichen auf den Bildschirm gebracht. Zur Verwendung anderer Zeichenformate ändern Sie dort die Zahl 16 in den Zeilen 170 und 177 entsprechend. Außerdem ändern Sie in Zeile 200 die Sub-Anweisung.

(Manolis Dählmann/bm)

Binär	Hexadezimal
0000111111000000	3FF0
0111110000011100	7806
0011100000000000	3800
0001111000000000	1E00
0000000111111000	01F8
0110000000001110	600E
0111000000001111	700F
0001111100111110	03F8

So sieht die Definition des Buchstabens »S« aus.

Stapeldatei für Remember

```
REM Konvertierung der Fonts und Treiber
BINOBJ      EGAVGA.BGI      VGA     VGA_Treiber
BINOBJ      TRIP.CHR        TRIP    Font_1
BINOBJ      LITT.CHR        LITT    Font_2
```

```
[5X3]    1: {Programm: fontdrv.pas
[LK4]    2:  Funktion: eigene Fonts für
[PX2]    3:            den VGA-Mode 13hex
[G64]    4:  Sprache:  Turbo Pascal 6.0
[W94]    5:  Autor:    Manolis Dählmann
[585]    6:  (c) 1992 DMV Widuch GmbH & Co. KG }
[Ø9Ø]    7:
[A22]    8: program fontdrv;
[5H1]    9: uses crt;
[STØ]   1Ø: type
[382]   11:   Fname = record
[BD4]   12:     name: Array [Ø..11] of Char;
[7V1]   13:     endk: Byte;
[ZNØ]   14:   end;
[6XØ]   15: const
[LL5]   16:   { Dateilaenge von FONTPASØ.DAT ! }
[WF3]   17:   file_len : Word  = 1ØØØ;
[2X2]   18:   file_name: fname =
[WK4]   19:     (name:'FONTPASØ.DAT';endk:Ø);
[RNØ]   2Ø: var
[DX6]   21:   { Zeiger auf Buffer für FONTPASØ.DAT }
[FW1]   22:   p: Pointer;
[DZ4]   23:   Handle: Word;{ Filehandle    }
[CL4]   24:   I: Word;     { = Register AX }
[ØR5]   25: function VGA_on: Word; assembler;
[FNØ]   26: asm
```

```
[2W5]   27:   mov   ax,$1aØØ (* testen, ob VGA-Karte *)
[D51]   28:   int   $1Ø
[6F6]   29:   cmp   al,$1a   (* Funktion unterstützt?*)
[PR5]   3Ø:   jne   @novga   (* keine VGA - Karte    *)
[YD1]   31:   cmp   bl,7
[DF1]   32:   jne   @or8
[6X1]   33:   jmp   @javga
[1VØ]   34: @or8:
[1E1]   35:   cmp   bl,8
[XW1]   36:   jne   @novga
[H71]   37: @javga:
[VV4]   38:   mov   ax,$13    (* 32Ø*2ØØ,256 Farben *)
[D51]   39:   int   $1Ø
[KC4]   4Ø:   mov   ax,1      (* erfolgreich *)
[ØB1]   41:   jmp   @ok
[N91]   42: @novga:
[RF5]   43:   xor   ax,ax     (* Keine VGA - Karte *)
[LPØ]   44: @ok:
[ZNØ]   45: end;
[415]   46: procedure VGA_off;assembler;
[FNØ]   47: asm
[5X1]   48:   mov   ax,$12ØØ
[CX1]   49:   mov   bx,$ØØ31
[D51]   5Ø:   int   $1Ø
[8X1]   51:   mov   ax,$ØØØ3
[D51]   52:   int   $1Ø
[ZNØ]   53: end;
[4U5]   54: procedure set_palette;assembler;
[FNØ]   55: asm
[BC1]   56:   push ds
[RC5]   57:   mov  cx,33 (* Farben  Ø - 32 setzen *)
[E94]   58:   xor  bl,bl (* Farbnr. Ø - 32 *)
[RP5]   59:   lds  si,p  (* Palettendaten, Qelle *)
[5W5]   6Ø:   mov  si,[si](* Zeiger auf Palbytes *)
[2X2]   61:                (* in FONTPASØ.DAT *)
[AZ1]   62:   mov  dx,$Ø3C6
[GW1]   63:   mov  al,$ff
[BS1]   64:   out  dx,al
[F11]   65: @setp:
[6P1]   66:   mov  al,bl
[EZ1]   67:   mov  dx,$Ø3C8
[L35]   68:   out  dx,al (* Farbnummer in al *)
[9U5]   69:   inc  dx    (* zu setzendes Farbbyte *)
[F33]   7Ø:   lodsb      (* auf $Ø3c9 *)
[754]   71:   out  dx,al(* AL Ø - 63 rot *)
[CVØ]   72:   lodsb
[LL4]   73:   out  dx,al(* AL Ø - 63 grün *)
[CVØ]   74:   lodsb
[U74]   75:   out  dx,al(* AL Ø - 63 blau *)
[GØ1]   76:   inc bl
[UT1]   77:   loop @setp
[E31]   78:   pop ds
[ZNØ]   79: end;
[EB4]   8Ø: (* Ein/Ausblenden des Fonts
[VU3]   81:    blnr = Helligkeit          *)
[RE6]   82: procedure blend1(blnr:byte);assembler;
[FNØ]   83: asm
[BC1]   84:   push ds
[LM1]   85:   mov  cx,33
[FP1]   86:   xor  bl,bl
[Ø22]   87:   mov  ah,blnr
[2Ø3]   88:   and  ah,63  (* max 63 *)
[SR5]   89:   lds  si,p   (* Palettendaten, Qelle *)
[5W5]   9Ø:   mov  si,[si](* Zeiger auf Palbytes *)
[2X2]   91:                (* in FONTPASØ.DAT *)
[AZ1]   92:   mov  dx,$Ø3C6
[GW1]   93:   mov  al,$ff
[BS1]   94:   out  dx,al
[F11]   95: @setp:
[6P1]   96:   mov  al,bl
[EZ1]   97:   mov  dx,$Ø3C8
[L35]   98:   out  dx,al (* Farbnummer in al *)
[EG5]   99:   inc  dx    (* zu setzende Farbbyte *)
[F33]  1ØØ:   lodsb      (* auf $Ø3c9 *)
[SR1]  1Ø1:   cmp  al,ah
[FU1]  1Ø2:   jbe  @setr
[AT1]  1Ø3:   mov  al,ah
[L11]  1Ø4: @setr:
[754]  1Ø5:   out  dx,al(* AL Ø - 63 rot *)
[CVØ]  1Ø6:   lodsb
[SR1]  1Ø7:   cmp  al,ah
[3T1]  1Ø8:   jbe  @setg
[AT1]  1Ø9:   mov  al,ah
[XØ1]  11Ø: @setg:
[LL4]  111:   out  dx,al(* AL Ø - 63 grün *)
[CVØ]  112:   lodsb
[SR1]  113:   cmp  al,ah
[FS1]  114:   jbe  @setb
[AT1]  115:   mov  al,ah
[LØ1]  116: @setb:
[U74]  117:   out  dx,al(* AL Ø - 63 blau *)
[W45]  118:   inc  bl
[AY1]  119:   loop @setp
[G71]  12Ø:   pop  ds
[ZNØ]  121: end;
[BT6]  122: procedure writefont1(xp:word;yp:word;
[XY3]  123:   str:string);assembler;
[FNØ]  124: asm
[BC1]  125:   push ds
[7T1]  126:   mov  bx,sp
[322]  127:   lds  si,str
[X75]  128:   lodsb    (* Stringlaenge nach cx *)
[YS1]  129:   mov  cx,ax
[SN1]  13Ø:   xor  ch,ch
[EM5]  131:   mov  dx,32Ø (* XYstart ausrechnen *)
[GT1]  132:   mov  ax,yp
[NA1]  133:   mul  dx
[DN1]  134:   add  ax,xp
[Ø35]  135:   mov  di,ax (* XYstart in ES:DI *)
[921]  136:   mov  ax,$aØØØ
[GS1]  137:   mov  es,ax
[UR1]  138: @nextchar:
[WC1]  139:   push cx
[BC1]  14Ø:   push ds
[TC1]  141:   push bx
[XP3]  142:   lodsb (* Zeichen laden *)
[GC1]  143:   push si
[A62]  144:   call @loadptr
[AA1]  145:   pop  si
[LA1]  146:   pop  bx
[5A1]  147:   pop  ds
[PA1]  148:   pop  cx
[6G2]  149:   loop @nextchar
[5A1]  15Ø:   pop  ds
[4R1]  151:   jmp  @wend
[FL1]  152: @loadptr:
[PC1]  153:   push ax
```

```
[TB1]  154:   push bp
[XR1]  155:   mov  bp,bx
[752]  156:   mov  ax,[bp]
[DS1]  157:   mov  ds,ax
[L91]  158:   pop  bp
[MU5]  159:   lds  si,p (* DS:SI auf FONTPASØ.DAT *)
[SW6]  16Ø:   inc  si (* Palettenzeiger überspringen *)
[581]  161:   inc  si
[GA1]  162:   pop  ax
[EN1]  163:   xor  ah,ah
[UE5]  164:   sub  al,$3Ø (* Offsettabelle ab 'Ø' *)
[VX2]  165:   shl  ax,1   (* ax * 2 *)
[HM1]  166:   add  si,ax
[6T5]  167:   (* si zeigt auf Pixelfont Zeichen *)
[S52]  168:   mov  si,[si]
[9B4]  169:   (* 16 * 16 Pixel-Fond ploten *)
[SM1]  17Ø:   mov  cx,16
[VZ4]  171:   (* bh Hintergrunfarbe,bl Vord.*)
[8L1]  172:   mov  bl,17
[LD1]  173:   mov  bh,1
[MB1]  174: @nextz: '
[WC1]  175:   push cx
[3B1]  176:   push di
[SM1]  177:   mov  cx,16
[KW3]  178:   lodsw (* 16 Pixel laden *)
[CE1]  179: @nextpi:
[ZD1]  18Ø:   shl  ax,1
[PC1]  181:   push ax
[TT5]  182:   (* Hintergrund oder Vordergrund ? *)
[H12]  183:   jnc  @ground
[UN1]  184:   mov  al,bh
[NXØ]  185:   stosb
[6R1]  186:   jmp  @okpl
[AE1]  187: @ground:
[6P1]  188:   mov  al,bl
[H56]  189:   stosb (* durch 'inc di' ersetzen dann *)
[F11]  19Ø: @okpl:
[WN4]  191:   inc  bh (* kein Hintergrund *)
[GA1]  192:   pop  ax
[362]  193:   loop @nextpi
[LN1]  194:   xor  bh,bh
[Y61]  195:   inc  bl
[X81]  196:   pop  di
[PM1]  197:   add  di,32Ø
[PA1]  198:   pop  cx
[252]  199:   loop @nextz
[RN2]  2ØØ:   sub  di,(32Ø*16)-16
[3PØ]  2Ø1:   ret
[XØ1]  2Ø2: @wend:
[ZNØ]  2Ø3: end;
[DZ4]  2Ø4: function loadfont (p:pointer;
[AT4]  2Ø5:   f:pointer): word; assembler;
[FNØ]  2Ø6: asm
[BC1]  2Ø7:   push ds
[484]  2Ø8:   mov  ax,$3dØØ (* nur lesen *)
[S16]  2Ø9:   lds  DX,f     (* Dateinamen übergeben *)
[564]  21Ø:   int  $21      (* Datei öffnen *)
[KD1]  211:   jnc  @nfe
[5A1]  212:   pop  ds
[N32]  213:   mov  handle,Ø
[KØ2]  214:   jmp  @fehler
[VTØ]  215: @nfe:
[5A1]  216:   pop  ds
[CY4]  217:   mov  handle,ax (* Datei lesen *)
[VS1]  218:   mov  bx,ax
[BC1]  219:   push ds
[7U1]  22Ø:   mov  ah,$3F
[ST6]  221:   mov  cx,file_len (* Dateigröße merken *)
[VB6]  222:   lds  dx,p (* Speicherplatz übergeben *)
[H41]  223:   int  $21
[6U1]  224:   jc   @flesen
[D55]  225:   mov  ah,$3e (* Funktionsnummer *)
[3N4]  226:   int  $21    (* Datei schließen *)
[6U1]  227:   jc   @flesen
[RA4]  228:   xor  ax,ax  (* Kein Fehler *)
[ED1]  229: @flesen:
[5A1]  23Ø:   pop  ds
[5H5]  231: @fehler: (* Loadfont = Fehlercode *)
[LC5]  232: end;    (* 2 = File nicht gefunden *)
[B35]  233: function memresfont:Boolean;
[MUØ]  234: begin
[A53]  235:   memresfont:= True;
[XB5]  236:   if MaxAvail < file_len then begin
[TA3]  237:   memresfont:= False;
[3D1]  238:   end else
[MUØ]  239:   begin
[M83]  24Ø:     GetMem(p,file_len);
[ZNØ]  241:   end;
[ZNØ]  242: end;
[CX3]  243: procedure memfreefont;
[MUØ]  244: begin
[7D3]  245:   FreeMem(p,file_len);
[ZNØ]  246: end;
[CG5]  247: { Demoaufruf der Programmteile }
[RNØ]  248: var
[WP6]  249:   fptr: pointer;{ Zeiger auf File_name }
[5S5]  25Ø:   Zahl: longint;{ Demovariablen }
[SF2]  251:   s   : string[11];
[8H2]  252:   sØ  : string[11];
[CA2]  253:   strlen: byte;
[M22]  254:   blend : Byte;
[MUØ]  255:   begin
[PF1]  256:   Zahl := Ø;
[X82]  257:   sØ := 'ØØØØØØØØØØ';
[MY1]  258:   i := vga_on;
[LM1]  259:   blend := Ø;
[TG2]  26Ø:   if i=1 then begin
[SY3]  261:     if memresfont then begin
[6Y2]  262:       fptr:=@File_name;
[ØA3]  263:       i:=loadfont(p,fptr);
[PG2]  264:         if i=Ø then Begin
[LS2]  265:           (* set_palette; *)
[SD1]  266:           blend1(Ø);
[6S4]  267:           writefont1(1Ø,1Ø,'<::SCORE::<');
[KT2]  268:           repeat inc (Zahl);
[UV1]  269:             inc (Blend);
[D52]  27Ø:             blend1(blend);
[WL1]  271:             delay (15);
[512]  272:             str(Zahl,s);
[EC3]  273:             strlen:= ord(s[Ø]);
```

Listing 11. Mit »fontdrv.pas« binden Sie selbstdefinierte farbige Zeichensätze in Ihre Spiele ein.

```
[9R3]  274:              s@[@] := chr(11-strlen);
[EL1]  275:              s:= s@ + s;
[322]  276:              writefont1(1@,3@,s)
[K13]  277:            until keypressed;
[362]  278:            memfreefont;
[7D1]  279:            VGA_off;
[@C2]  28@:          end else begin
[7D1]  281:            VGA_off;
[NM5]  282:            WriteLn('Kann FONTPAS@.DAT nicht '+
[PC2]  283:                    'finden !');end;
[@C2]  284:      end else begin
[7D1]  285:      VGA_off;
[FW3]  286:      WriteLn ('Nicht genug'+
[AV3]  287:               ' Speicherplatz !');end;
[N46]  288:    end else WriteLn ('Keine VGA - Karte !')
[KN@]  289: end.
```

Listing 11. Mit »fontdrv.pas« binden Sie selbstdefinierte farbige Zeichensätze in Ihre Spiele ein.

```
[@14]    1: ;Programm: fontpas@.asm
[DP5]    2: ;Funktion: Zeichensatzdefinition
[8G2]    3: ;Sprache:  TASM
[2K4]    4: ;Autor:    Manolis Dählmann
[7X4]    5: ; (c) 1992 DMV Widuch GmbH & Co. KG
[@@@]    6:
[XV3]    7: code  segment para 'CODE'
[HH1]    8:       org    1@@h
[E21]    9: start:
[UX3]   1@: F_group  group fontdat
[FZ4]   11: assume es:F_group,ds:F_group
[9U2]   12: fontdat  segment
[WD4]   13: ; Zeiger auf Palettendaten
[FN3]   14: palp  dw offset paldat
[L96]   15: ; Offset-Tabelle der einzelnen Zeichen
[EP3]   16: ptr@  dw offset z@font
[KP3]   17: ptr1  dw offset z1font
[PP3]   18: ptr2  dw offset z2font
[UP3]   19: ptr3  dw offset z3font
[YP3]   2@: ptr4  dw offset z4font
[2R3]   21: ptr5  dw offset z5font
[6R3]   22: ptr6  dw offset z6font
[AR3]   23: ptr7  dw offset z7font
[ER3]   24: ptr8  dw offset z8font
[KR3]   25: ptr9  dw offset z9font
[SN2]   26:       dw offset lzoffs
[SN2]   27:       dw offset lzoffs
[T44]   28: grptr dw offset grafont
[KU3]   29: glptr dw offset glfont
[544]   3@: klptr dw offset klafont
[SN2]   31:       dw offset lzoffs
[SN2]   32:       dw offset lzoffs
[ST3]   33: Aptr  dw offset lzoffs
[UT3]   34: Bptr  dw offset lzoffs
[TG3]   35: Cptr  dw offset Cfont
[YT3]   36: Dptr  dw offset lzoffs
[1H3]   37: Eptr  dw offset Efont
[2U3]   38: Fptr  dw offset lzoffs
[4U3]   39: Gptr  dw offset lzoffs
[6U3]   4@: Hptr  dw offset lzoffs
[8U3]   41: Iptr  dw offset lzoffs
[AU3]   42: Jptr  dw offset lzoffs
[CU3]   43: Kptr  dw offset lzoffs
[EU3]   44: Lptr  dw offset lzoffs
[GU3]   45: Mptr  dw offset lzoffs
[KU3]   46: Nptr  dw offset lzoffs
[9K3]   47: Optr  dw offset ofont
[PU3]   48: Pptr  dw offset lzoffs
[SU3]   49: Qptr  dw offset lzoffs
[NK3]   5@: Rptr  dw offset rfont
[TK3]   51: Sptr  dw offset sfont
[YU3]   52: Tptr  dw offset lzoffs
[@V3]   53: Uptr  dw offset lzoffs
[2V3]   54: Vptr  dw offset lzoffs
[4V3]   55: Wptr  dw offset lzoffs
[242]   56: ;Leerzeichen
[V35]   57: lzoffs dw @,@,@,@,@,@,@,@,@,@,@,@,@,@,@,@,@,@,@,@
[2D3]   58: ;Buchstaben  S C O R E
[S21]   59: Sfont:
[HM4]   6@: dw @@@@111111@@@@@@b,@@111111111@@@@b
[FM4]   61: dw @11111@@@@@111@@b,@1111@@@@@@@11@b
[VL4]   62: dw @@111@@@@@@@@@@b,@@111@@@@@@@@@2@b
[6M4]   63: dw @@@111111@@@@@@@b,@@@@@111111@@@@@b
[5M4]   64: dw @@@@@@@111111@@@b,@@@@@@@@@@111@@b
[AM4]   65: dw @11@@@@@@@@1110b,111@@@@@@@@1111b
[GM4]   66: dw @111@@@@@@@1111b,@@111@@@@@@@1111b
[SM4]   67: dw @@@111111@@111110b,@@@@@@111111@@@b
[336]   68: Cfont dw @@7C@h,@1FF8h,@381Ch,@7@@Eh,@7@@@h
[ZV4]   69:  dw @F@@@h,@F@@@h,@F@@@h,@F@@@h,@F@@@h,@7@@@h
[RZ4]   7@:  dw @7@@@h,@38@3h,@3C@7h,@1E@Eh,@@7FCh
[E81]   71:  dw @@3F@h
[U46]   72: Ofont dw @@FC@h,@1C7@h,@3818h,@7@@Ch,@F@@6h
[YY4]   73: dw @E@@7h,@E@@7h,@E@@7h,@E@@7h,@E@@7h
[H15]   74: dw @6@@7h,@7@@Eh,@38@Eh,@1C1Ch,@1E78h
[P81]   75: dw @@7E@h
[N96]   76: Rfont dw @3F@@h,@FFE@h,@E@78h,@C@1Ch,@C@1Eh
[A35]   77: dw @C@3Ch,@C@7@h,@C1C@h,@FF8@h,@E38@h
[525]   78: dw @C1C@h,@C@E@h,@C@7@h,@C@38h,@E@1Eh
[V71]   79: dw @F@3Fh
[B76]   8@: Efont dw @FFFCh,@FFFEh,@6@@7h,@6@@@h,@6@@@h
[LZ4]   81: dw @6@@@h,@6@@@h,@7FFCh,@7FFCh,@6@@@h
[KX4]   82: dw @6@@@h,@6@@@h,@6@@@h,@6@3h,@FFFEh
[6C1]   83: dw @FFFCh
[785]   84: glfont dw @@@@@h,@AABCh,@5544h,@2AA4h
[YW4]   85: dw @1554h,@@AA8h,@@554h,@@2A8h,@@554h
[KY4]   86: dw @@AA4h,@1544h,@2A84h,@55@4h,@2A@4h
[X42]   87: dw @55FCh,@@@@@h
[7B1]   88: ; Muster
[VB5]   89: klafont dw @@@@@h,@@@@@h,@3D55h,@22AAh
[6Y4]   9@: dw @2554h,@2AA8h,@155@h,@2AA@h,@154@h
[PY4]   91: dw @2AA@h,@255@h,@22A8h,@2154h,@2@A8h
[T32]   92: dw @3F54h,@@@@@h
[CF5]   93: grafont dw @@@@@h,@@@@@h,@AABCh,@5544h
[WZ4]   94: dw @2AA4h,@1554h,@@AA8h,@@554h,@@2A8h
[DX4]   95: dw @@554h,@@AA4h,@1544h,@2A84h,@15@4h
[V52]   96: dw @2AFCh,@@@@@h
[FB1]   97: ;Zahlen
[Z25]   98: z@font dw @@@@@h,@@7F@h,@@E18h,@1C@Ch
[MU4]   99: dw @38@Eh,@7@@7h,@7@@7h,@7@@7h,@7@@7h
[4Z4]  1@@: dw @7@@7h,@3@@7h,@38@Eh,@1C@Ch,@@E38h
[642]  1@1: dw @@7F@h,@@@@@h
[B75]  1@2: z1font dw @@1F@h,@@3F8h,@@7F8h,@@FF8h
[@Y4]  1@3: dw @1F78h,@3E78h,@@@78h,@@@78h,@@@78h
[FU4]  1@4: dw @@@78h,@@@78h,@@@78h,@@@78h,@@@78h
[FA2]  1@5: dw @@3FCh,@@FFEh
[H85]  1@6: z2font dw @3FE@h,@7FF8h,@F@3Ch,@E@1Eh
[R15]  1@7: dw @7@1Eh,@381Eh,@1C3Ch,@@C78h,@@@F@h
[5X4]  1@8: dw @@1C@h,@@38@h,@@7@@h,@1E@@h,@3C@6h
[FC2]  1@9: dw @F83Eh, @FFFCh
[R85]  11@: z3font dw @3FF@h,@7FF8h,@F@3Ch,@E@1Eh
[145]  111: dw @6@1Eh,@@@3Eh,@@@7Ch,@1FF8h,@@FF8h
[G35]  112: dw @@@78h,@6@3Ch,@E@1Eh,@E@1Eh,@F@7Eh
```

```
[3B2]  113: dw @7FFCh,@1FF8h
[2D5]  114: z5font dw @FFFCh,@FFFCh,@C@1Ch,@@@1Ch
[W15]  115: dw @@@1Ch,@@@1Ch,@@3FCh,@@FFCh,@3E@@h
[355]  116: dw @7C@@h,@F@3Ch,@E@1Eh,@E@1Eh,@F@7Eh
[3A2]  117: dw @7FFCh,@1FF@h
[WD5]  118: z7font dw @FFFEh,@FFFEh,@C@1Eh,@@@1Eh
[7Z4]  119: dw @@@3Ch,@@@F8h,@@3E@h,@@F8@h,@1F@@h
[KX4]  12@: dw @1E@@h,@1E@@h,@1E@@h,@1E@@h,@1E@@h
[562]  121: dw @1F@@h,@3FC@h
[A55]  122: z4font dw @@1F8h,@@3B8h,@@738h,@@E38h
[V45]  123: dw @1C38h,@3838h,@7@38h,@E@3Bh,@FFFEh
[@Y4]  124: dw @7FFEh,@@@38h,@@@38h,@@@38h,@@@38h
[W82]  125: dw @@@FCh,@@1FEh
[V55]  126: z6font dw @@@7Eh,@@1F8h,@@3E@h,@@7C@h
[945]  127: dw @@F8@h,@3F@@h,@7FC@h,@FFF@h,@F@7Ch
[G25]  128: dw @E@1Eh,@E@@Eh,@E@@Eh,@7@@Eh,@383Ch
[A82]  129: dw @1FF8h,@@7E@h
[@65]  13@: z8font dw @@FE@h,@3E78h,@781Ch,@7@@Ch
[E15]  131: dw @7@1Ch,@3878h,@3FF@h,@7FF8h,@F@3Ch
[G25]  132: dw @E@1Eh,@E@@Eh,@E@@Eh,@7@@Eh,@383Ch
[A82]  133: dw @1FF8h,@@7E@h
[B95]  134: z9font dw @@FF8h,@3E7Ch,@781Eh,@F@@Eh
[Z55]  135: dw @F@@Eh,@781Eh,@3C3Eh,@1FFCh,@@7FCh
[MX4]  136: dw @@@3Ch,@@@3Ch,@@@38h,@@@78h,@38F@h
[W72]  137: dw @1FF@h,@@7C@h
[132]  138: paldat:  ;rgb
[8H3]  139: db    @, @, @ ;Hintergrund
[TP4]  14@: db  5@,15,18,52,16,19,4@,17,2@,32,18,21
[AP4]  141: db  31,19,22,3@,2@,23,29,21,24,28,22,26
[GR4]  142: db  27,23,28,26,24,3@,25,25,32,24,26,34
[8S4]  143: db  23,27,36,22,28,5@,21,29,52,2@,3@,58
[5K4]  144: db   8,11,11,  9,11,11,1@,11,11,11,11,11
[WL4]  145: db  12,11,11,13,11,11,14,11,11,15,11,11
[4N4]  146: db  16,11,11,17,11,11,18,11,11,19,11,11
[KL4]  147: db  2@,11,11,21,11,11,22,11,11,23,11,11
[R42]  148: fontdat ends
[TF1]  149: code ends
[4K1]  15@: end start
```

Listing 12. In »fontpas0.asm« definieren Sie Ihre Zeichen.

VGA-Angriff auf Neuronen

Vielleicht erinnern Sie sich noch an das Spiel Senso, ein Veteran der elektronischen Spiele, das wie ein kleines Ufo aussieht und ebensolche Ausdrucksformen besitzt. Auf der Oberfläche dieses Spiels (Bild 3) finden Sie vier Felder, von denen jedes eine eigene Farbe und ein eigenes Geräusch besitzt. Bei Spielbeginn fängt eines dieser Felder an, sich optisch und akustisch bemerkbar zu machen. Ihre Aufgabe ist es, anschließend eben dieses Feld anzuklicken. Waren Sie erfolgreich, blinkt dieses Feld noch einmal und gleich darauf ein neues. Wiederum müssen Sie diese beiden Felder in der richtigen Reihenfolge drücken. In jedem Durchgang gesellt sich ein neues, zufälliges Feld dazu. Die wachsende Zahl an Feldern bietet dem menschlichen Gehirn schon bald eine echte Herausforderung.

Das Programm »remember.pas« (Listing 13) stellt die PC-Simulation dieses Spielprinzips dar. Die auf dem Bildschirm blinkenden Sektoren wählen Sie mit den entsprechenden Cursortasten an. Es existieren zusätzlich drei verschiedene Schwierigkeitsgrade, welche die Spielgeschwindigkeit und die akustische Unterscheidbarkeit der Felder beeinflussen.

Das Programm ist ab Turbo Pascal 5.0 lauffähig und nur für Rechner mit VGA-Grafikkarten geeignet. Um das kompilierte Programm auch Pascalunabhängig zu starten, sind die Stapeldatei »konvert.bat« (Textbox »Stapeldatei für Remember«) und die Unit »modul.pas« (Listing 14) erforderlich, welche die benötigten Schriftarten und den BGI-Grafiktreiber einbindet. Zuerst starten Sie »konvert.bat« von der MS-DOS-Ebene. Die Dateien aus dem Lieferumfang von Turbo Pascal »egavga.bgi«, »trip.chr«, »litt.chr« und »binobj.exe« müssen sich bei diesem Vorgang im selben Verzeichnis befinden. Lohn dieser Mühe sind drei Objektdateien (»trip.obj«, »litt.obj« und »vga.obj«), welche die Unit »modul.pas« benötigt.

Die Prozedur »initgr_vga« schaltet in den VGA-Modus mit 640 × 480 Punkten bei 16 Farben und erledigt die nötigen Farbeinstellungen. Besondere Beachtung verdient die Prozedur »wait«. Sie entspricht dem Befehl »Delay«, ist jedoch taktfrequenzunabhängig (das wurde von Borland fälschlicherweise auch für Delay versprochen – ein kurzes Spiel mit dem Turbo-Taster am Gehäuse beweist jedoch das Gegenteil). Der Übergabewert wird in Hundertstelsekunden gewertet und ist zwischen 10 und 360 000 (60 min) gültig.

Die Prozedur »blink« läßt die einzelnen Felder in den levelbedingten Tonhöhen und Wartezeiten aufleuchten. Das gesamte Erscheinungsbild des Spiels ist in den Prozeduren »writedialogbox« und »playground« festgelegt. Der Spielverlauf findet in der Prozedur »action« statt, die das gesamte Handling der verzeigerten Liste, in der

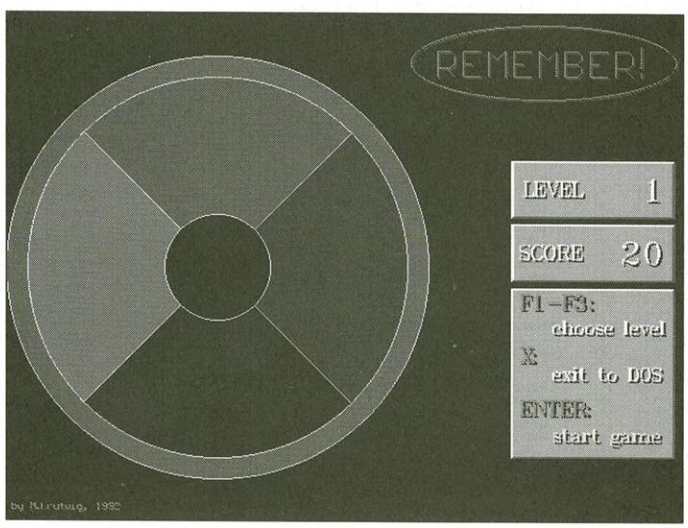


Bild 3. Remember reizt Ihre Konzentrationsfähigkeit aus.

die Felder gespeichert werden, und die Tastaturabfragen übernimmt. Die Kommentarklammern trennen die einzelnen Unterfunktionen. Die Menübox (Dialogbox) wird durch die Prozedur »select« gesteuert. Die Prozedur »sc« ist mit dem Befehl »Setcolor« identisch und dient nur zur Verkürzung des Quelltextes.

Dem geübten Programmierer wird es nicht schwerfallen, »remember.pas« zu erweitern und zu verbessern. Einige Anregungen: Spielfeld optisch aufwerten, Einbau einer Highscore-Datei, meherere Levels, zum Beispiel ohne Ton, schönes Ladebild voranstellen oder eine Joystick-Steuerung.

(Michael Krutwig/bm)

```
{P54}    1: {Programm: remember.pas
{D85}    2:  Funktion: Konzentrationsspiel
{G64}    3:  Sprache:  Turbo Pascal 6.0
{514}    4:  Autor:    Michael Krutwig
{585}    5:  (c) 1992 DMV Widuch GmbH & Co. KG }
{000}    6:
{WZ2}    7: program remember;
{V54}    8: uses graph,crt,dos,modul;
{GK4}    9: type zeigertyp = ^feldtyp;
{8U2}   10:      feldtyp = record
{1W1}   11:                  feld :byte;
{AN3}   12:                  naechster :zeigertyp;
{ZN0}   13:                end;
{UR2}   14: var level :byte;
{GY3}   15: procedure sc(col :byte);
{R04}   16: begin setcolor(col); end;
{WR3}   17: procedure initgr_vga;
{003}   18:   var tr,mo :integer;
{MU0}   19: begin
{VW1}   20:   tr:=9; mo:=2;
{Z53}   21:   initgraph(tr,mo,'');
{MM3}   22:   setrgbpalette(1,0,0,32);
{PM3}   23:   setrgbpalette(2,0,32,0);
{SM3}   24:   setrgbpalette(4,32,0,0);
{V44}   25:   setrgbpalette(5,160,0,236);
{NX3}   26:   setrgbpalette(20,32,32,0);
{ZN0}   27: end;
{9L4}   28: procedure wait(d :longint);
{HG3}   29:   var reg  :registers;
{FC5}   30:       x,min_A,min_B,sek,hun  :longint;
{MU0}   31: begin
{NX4}   32:   if (d<360000) and (d>=5) then begin
{E24}   33:     reg.AH:=$2C; intr($21,reg);
{HC4}   34:     min_A:=reg.CL; sek:=reg.DH;
{Z22}   35:     hun:=reg.DL;
{W54}   36:     d:=d+hun+sek*100+min_A*6000;
{P31}   37:     repeat
{E24}   38:       reg.AH:=$2C; intr($21,reg);
{LC4}   39:       min_B:=reg.CL; sek:=reg.DH;
{Z22}   40:       hun:=reg.DL;
{893}   41:       if min_B<min_A then
{CN2}   42:         min_B:=min_B+60;
{784}   43:       x:=hun + sek*100 + min_B*6000;
{NZ1}   44:     until x>=d;
{ZN0}   45:   end;
{ZN0}   46: end;
{WY4}   47: procedure blink(sector :byte;
{9L2}   48: pause :boolean);
{LR3}   49:   var w,s1,s2,s3,s4 :word;
{MU0}   50: begin
{H12}   51:   case level of
{DR4}   52:     1 : begin w:=80; s1:=100; s2:=200;
{UQ3}   53:           s3:=400; s4:=700; end;
{DR4}   54:     2 : begin w:=50; s1:=200; s2:=300;
{M03}   55:           s3:=400; s4:=500; end;
{BR4}   56:     3 : begin w:=20; s1:=300; s2:=320;
{G13}   57:           s3:=340; s4:=360; end;
{ZN0}   58:   end;
{XR3}   59:   if not pause then w:=15;
{RE2}   60:   case sector of
{Y81}   61:     1 : begin
{NS3}   62:           setrgbpalette(2,0,252,0);
{TM4}   63:           sound(s1); wait(w); nosound;
{PM3}   64:           setrgbpalette(2,0,32,0);
{ZN0}   65:         end;
{091}   66:     2 : begin
{FT3}   67:           setrgbpalette(4,252,0,0);
{UM4}   68:           sound(s2); wait(w); nosound;
{SM3}   69:           setrgbpalette(4,32,0,0);
{ZN0}   70:         end;
{291}   71:     3 : begin
{3T3}   72:           setrgbpalette(1,0,0,252);
{VM4}   73:           sound(s3); wait(w); nosound;
{MM3}   74:           setrgbpalette(1,0,0,32);
{ZN0}   75:         end;
{491}   76:     4 : begin
{054}   77:           setrgbpalette(20,252,252,0);
{WM4}   78:           sound(s4); wait(w); nosound;
{NX3}   79:           setrgbpalette(20,32,32,0);
{ZN0}   80:         end;
{ZN0}   81:   end;
{ZN0}   82: end;
{MR5}   83: procedure Box(x1,y1,x2,y2 :integer);
{MU0}   84: begin
{TZ2}   85:   setfillstyle(1,7);
{AH4}   86:   bar(x1+1,y1+1,x2-1,y2-1); sc(15);
{BK4}   87:   rectangle(x1+1,y1+1,x2-1,y2-1);
{NK4}   88:   rectangle(x1+2,y1+2,x2-2,y2-2);
{NK4}   89:   rectangle(x1+2,y1+2,x2-2,y2-2);
{UN3}   90:   line(x1+3,y1+3,x1+3,y2-3);
{7G4}   91:   sc(8); line(x2-1,y1+1,x2-1,y2-1);
{YN3}   92:   line(x2-2,y1+2,x2-2,y2-2);
{8P3}   93:   line(x2-3,y1+3,x2-3,y2-3);
{NN3}   94:   line(x1+2,y2-1,x2-1,y2-1);
{2N3}   95:   line(x1+3,y2-2,x2-2,y2-2);
{ZN0}   96: end;
{TR3}   97: procedure writelevel;
{Z63}   98:   var st :string[1];
{MU0}   99: begin
{NE2}  100:   str(level,st);
{9D6}  101:   settextstyle(1,0,4); setfillstyle(1,7);
{XT2}  102:   bar(555,142,628,185);
```

```
{LG4}  103:   sc(0);  outtextxy(627,177,st);
{UK4}  104:   sc(15); outtextxy(625,175,st);
{ZN0}  105: end;
{1E4}  106: procedure writedialogbox;
{MU0}  107: begin
{U94}  108:   settextstyle(1,0,2); sc(0);
{855}  109:   outtextxy(558,280,'F1-F3:'); sc(12);
{F34}  110:   outtextxy(556,278,'F1-F3:');
{R94}  111:   settextstyle(1,0,1); sc(0);
{TC5}  112:   outtextxy(630,303,'choose level');
{H01}  113:   sc(15);
{4D5}  114:   outtextxy(628,301,'choose level');
{U94}  115:   settextstyle(1,0,2); sc(0);
{UL4}  116:   outtextxy(505,330,'X:'); sc(12);
{1K3}  117:   outtextxy(503,328,'X:');
{R94}  118:   settextstyle(1,0,1); sc(0);
{X66}  119:   outtextxy(626,347,'exit to DOS'); sc(15);
{G45}  120:   outtextxy(624,345,'exit to DOS');
{U94}  121:   settextstyle(1,0,2); sc(0);
{NB5}  122:   outtextxy(558,380,'ENTER:'); sc(12);
{W94}  123:   outtextxy(556,378,'ENTER:');
{R94}  124:   settextstyle(1,0,1); sc(0);
{D06}  125:   outtextxy(624,405,'start game'); sc(14);
{VX4}  126:   outtextxy(622,403,'start game');
{ZN0}  127: end;
{YA4}  128: procedure cleardialogbox;
{MU0}  129: begin
{EM5}  130:   setfillstyle(1,7); bar(480,261,630,417);
{ZN0}  131: end;
{PR5}  132: procedure writescore(s :integer);
{173}  133:   var st :string[3];
{MU0}  134: begin
{ZP1}  135:   str(s,st);
{6D6}  136:   setfillstyle(1,7); settextstyle(1,0,4);
{RP3}  137:   bar(555,202,628,245); sc(0);
{YF4}  138:   outtextxy(627,237,st); sc(15);
{9H3}  139:   outtextxy(625,235,st);
{ZN0}  140: end;
{HS3}  141: procedure playground;
{MU0}  142: begin
{HD3}  143:   settextjustify(2,0);
{ZH5}  144:   setbkcolor(0); cleardevice; sc(15);
{0N2}  145:   circle(200,240,200);
{FN2}  146:   circle(200,240,180);
{VM2}  147:   circle(200,240,50);
{DT2}  148:   line(73,112,165,204);
{NV2}  149:   line(73,368,165,276);
{BY2}  150:   line(327,368,235,276);
{DW2}  151:   line(327,112,235,204);
{266}  152:   setfillstyle(1,2); floodfill(150,100,15);
{966}  153:   setfillstyle(1,4); floodfill(300,240,15);
{776}  154:   setfillstyle(1,1); floodfill(150,380,15);
{D26}  155:   setfillstyle(1,6); floodfill(50,240,15);
{U06}  156:   setfillstyle(1,8); floodfill(2,240,15);
{U74}  157:   sc(5); settextstyle(2,0,20);
{1S4}  158:   outtextxy(610,60,'REMEMBER!');
{2R5}  159:   outtextxy(609,59,'REMEMBER!'); sc(13);
{SK5}  160:   outtextxy(608,58,'REMEMBER!'); sc(5);
{RY3}  161:   ellipse(510,47,0,360,125,35);
{K15}  162:   ellipse(509,46,0,360,125,35); sc(13);
{EZ3}  163:   ellipse(508,45,0,360,125,35);
{DA4}  164:   settextstyle(2,0,4); sc(5);
{C96}  165:   outtextxy(110,470,'DOS International');
{D01}  166:   sc(13);
{EA6}  167:   outtextxy(109,469,'DOS International');
{683}  168:   settextstyle(1,0,2);
{6W2}  169:   box(477,137,633,193);
{N55}  170:   sc(0); outtextxy(550,173,'LEVEL');
{W95}  171:   sc(15); outtextxy(548,171,'LEVEL');
{EW2}  172:   box(477,197,633,253);
{Z55}  173:   sc(0); outtextxy(550,233,'SCORE');
{RA5}  174:   sc(15); outtextxy(548,231,'SCORE');
{RV2}  175:   box(477,258,633,420);
{ZN0}  176: end;
{5Y2}  177: procedure action;
{VS6}  178:   var anker,lzeiger1,lzeiger2 :zeigertyp;
{7H2}  179:       n          :integer;
{D42}  180:       ch,st      :char;
{XB3}  181:       break,lost :boolean;
{XF2}  182:       sector     :byte;
{MU0}  183: begin
{935}  184:   cleardialogbox; writescore(0);
{RV2}  185:   box(537,308,573,370);
{Y83}  186:   randomize; wait(100);
{H15}  187:   n:=0; anker:=nil; break:=false;
{P31}  188:   repeat
{1H5}  189:     while keypressed do ch:=readkey;
{BV4}  190:       { *** neues Feld bestimmen *** }
{7V3}  191:     if anker=nil then begin
{RY1}  192:       new(anker);
{YT2}  193:       lzeiger1:=anker;
{0C2}  194:     end else begin
{BD4}  195:       new(lzeiger1^.naechster);
{UA5}  196:       lzeiger1:=lzeiger1^.naechster;
{2N0}  197:     end;
{SU5}  198:     lzeiger1^.feld:=random(1000) MOD 4 +1;
{1D4}  199:     lzeiger1^.naechster:=NIL;
{CW3}  200:       { *** Felder anzeigen *** }
{TZ2}  201:     setfillstyle(1,7);
{TT2}  202:     bar(543,312,568,366);
{K83}  203:     settextstyle(1,0,6);
{A94}  204:     sc(5); outtextxy(565,358,'!');
{9H4}  205:     sc(13); outtextxy(563,356,'!');
{ZT2}  206:     lzeiger2:=anker;
{P31}  207:     repeat
{H12}  208:       case level of
{YR1}  209:         1 : wait(40);
{ES1}  210:         2 : wait(25);
{TR1}  211:         3 : wait(10);
{ZN0}  212:       end;
{9P4}  213:       blink(lzeiger2^.feld,true);
{YA5}  214:       lzeiger2:=lzeiger2^.naechster;
{LA3}  215:     until lzeiger2=nil;
{1H5}  216:     while keypressed do ch:=readkey;
{G51}  217:     ch:='x';
{3N4}  218:       {  ***** Felder eingeben **** }
{TZ2}  219:     setfillstyle(1,7);
{TT2}  220:     bar(543,312,568,366);
{F83}  221:     settextstyle(1,0,5);
{2D4}  222:     sc(5); outtextxy(568,355,'?');
{7K4}  223:     sc(13); outtextxy(566,353,'?');
{ZT2}  224:     lzeiger2:=anker;
{P31}  225:     repeat
{BW3}  226:       sector:=1; lost:=false;
{P31}  227:       repeat
{2Z4}  228:         if keypressed then ch:=readkey
{WP1}  229:         else begin
{AA4}  230:           sc(3); circle(200,240,sector);
{H12}  231:           case level of
{YR1}  232:             1 : wait(40);
{UR1}  233:             2 : wait(20);
{TR1}  234:             3 : wait(10);
{ZN0}  235:           end;
{7A4}  236:           sc(0); circle(200,240,sector);
{Y72}  237:           inc(sector,2);
{ZN0}  238:         end;
{8W5}  239:         if sector=51 then begin { Zeit um }
{464}  240:           lost:=true; break:=true;
{GM4}  241:           sound(50); wait(200); nosound;
{ZN0}  242:         end;
{B54}  243:       until (ch in [#72,#77,#80,#75])
{5L2}  244:       or (lost=true);
{VP3}  245:       if not break then begin
{LM1}  246:         case ch of
```

```
{KB2}  247:           #72 : sector:=1;
{AC2}  248:           #77 : sector:=2;
{NB2}  249:           #80 : sector:=3;
{AC2}  250:           #75 : sector:=4;
{EQ2}  251:         end; ch:='x';
{RA3}  252:         blink(sector,false);
{FV4}  253:         if lzeiger2^.feld=sector then
{MU0}  254:         begin
{YA5}  255:           lzeiger2:=lzeiger2^.naechster;
{FU3}  256:           inc(n,1); writescore(n);
{0C2}  257:         end else begin
{R22}  258:           lost:=true;
{D22}  259:           break:=true;
{3Y2}  260:           sound(40); wait(80);
{TA4}  261:           for sector:=1 to 3 do begin
{NN4}  262:             blink(lzeiger2^.feld,false);
{612}  263:             wait(10); end;
{3G1}  264:           nosound;
{ZN0}  265:         end;
{ZN0}  266:       end;
{FC5}  267:       if lzeiger2=nil then lost:=true;
{K12}  268:     until lost;
{R91}  269:     wait(50);
{L12}  270:   until break;
{UN3}  271:     { *** Heap säubern **** }
{2T2}  272:   lzeiger2:=anker;
{7L4}  273:   while lzeiger2<>nil do begin
{XA5}  274:     lzeiger1:=lzeiger2^.naechster;
{F53}  275:     dispose(lzeiger2);
{SB3}  276:     lzeiger2:=lzeiger1;
{ZN0}  277:   end;
{ZN0}  278: end;
{9Y2}  279: procedure select;
{T02}  280:   var ch :char;
{MU0}  281: begin
{SB4}  282:   writescore(0); writelevel;
{CU2}  283:   writedialogbox;
{P31}  284:   repeat
{P31}  285:     repeat
{702}  286:       ch:=readkey
{PB5}  287:     until ch in [#13,'x','X',#59,#60,#61];
{LM1}  288:     case ch of
{8T5}  289:       #13 : begin action; cleardialogbox;
{BH3}  290:             writedialogbox; end;
{Z83}  291:       #59 : if level<>1 then
{NL4}  292:               begin level:=1; writelevel;
{T23}  293:                 writescore(0); end;
{083}  294:       #60 : if level<>2 then
{SL4}  295:               begin level:=2; writelevel;
{T23}  296:                 writescore(0); end;
{683}  297:       #61 : if level<>3 then
{VL4}  298:               begin level:=3; writelevel;
{T23}  299:                 writescore(0); end;
{ZN0}  300:     end;
{KN3}  301:   until (ch='x') or (ch='X');
{ZN0}  302: end;
{MU0}  303: begin
{6K3}  304:   level:=1; initgr_vga;
{2E3}  305:   playground; select;
{522}  306:   closegraph;
{E93}  307:   directvideo:=true;
{H74}  308:   restorecrtmode; clrscr;
{KN0}  309: end.
```

Listing 13. Mit »remember.pas« überprüfen Sie Ihre Konzentrationsfähigkeit.

```
{MG3}    1: {Programm: modul.pas
{BH5}    2:  Funktion: Unit für remember.pas
{G64}    3:  Sprache:  Turbo Pascal 6.0
{514}    4:  Autor:    Michael Krutwig
{585}    5:  (c) 1992 DMV Widuch GmbH & Co. KG }
{000}    6:
{D12}    7: unit modul;
{SK1}    8: interface
{C02}    9:   uses graph;
{2Z3}   10:   procedure VGA_Treiber;
{X03}   11:   procedure Font_1;
{Y03}   12:   procedure Font_2;
{4L2}   13: implementation
{2Z3}   14:   procedure VGA_Treiber;
{R04}   15:   external; {$L VGA.OBJ}
{X03}   16:   procedure Font_1;
{4B4}   17:   external; {$L TRIP.OBJ}
{Y03}   18:   procedure Font_2;
{XA4}   19:   external; {$L LITT.OBJ}
{PF3}   20:   procedure einbinden;
{TK2}   21:     var i :integer;
{MU0}   22:   begin
{F66}   23:     i:=registerbgidriver(@VGA_Treiber);
{UP4}   24:     i:=registerbgifont(@Font_1);
{XP4}   25:     i:=registerbgifont(@Font_2);
{ZN0}   26:   end;
{MU0}   27: begin
{XR1}   28:   einbinden;
{KN0}   29: end.
```

Listing 14. Die Unit »modul.pas« zum Einbinden der Zeichensätze und des Grafiktreibers

Chaotisches Fraktal

Computerbilder – insbesondere Fraktale Gebilde – erfreuen sich großer Beliebtheit bei den Anwendern. Seit Mandelbrot das Apfelmännchen entdeckt hatte, wandelten weitere Forscher auf seinen Spuren und fanden eine schier unendlich große Anzahl dieser »selbstähnlichen« Gebilde. Dauerte deren jeweilige Berechnung in den Anfängen der Fraktalforschung noch sehr lange, so lassen sie sich

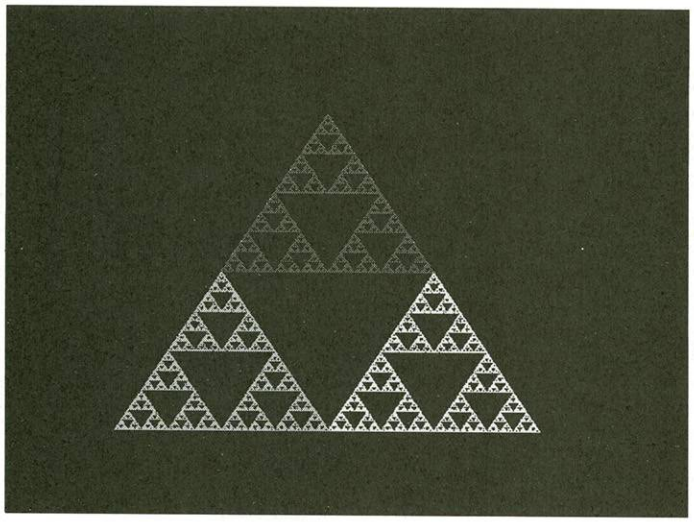


Bild 4. So ein »chaotisches« Fraktal erzeugen Sie mit dem Programm »chaos.pas«.

heutzutage schon in wenigen Sekunden zeichnen.

Das Programm »chaos.pas« (Listing 15) zeichnet ein besonderes Fraktal auf Ihrem VGA-Bildschirm. Es verwendet dazu den folgenden Algorithmus: Sie zeichnen ein Dreieck und benennen die Eckpunkte mit 1, 2 und 3. Sodann wählen Sie einen beliebigen Startpunkt, der auch außerhalb des Dreiecks liegen kann. Nun würfeln Sie und verbinden den Startpunkt mit dem gewürfelten Eckpunkt. Würfeln Sie eine 4, so verbinden Sie mit Ecke 1, bei einer 5 mit 2 und bei einer 6 mit 3. Halbieren Sie die Strecke und zeichnen Sie den Punkt. Dieser wird dann zum erneuten Startpunkt.

Das überraschende Ergebnis sehen Sie in Bild 4.

(Cord Krey/bm)

```
[2F3]    1: {Programm: chaos.pas
[Z15]    2:  Funktion: zeichnet ein Fraktal
[G64]    3:  Sprache:  Turbo Pascal 6.Ø
[ZØ3]    4:  Autor:    Cord Krey
[585]    5:  (c) 1992 DMV Widuch GmbH & Co. KG }
[ØØØ]    6:
[7G3]    7: program chaosspiel;
[A93]    8: uses graph,crt,dos;
[STØ]    9: TYPE
[3F1]   1Ø: P=RECORD
[BA4]   11:   X:INTEGER;Y:INTEGER;END;
[RNØ]   12: VAR
[ZH2]   13: P1,P2,P3,PB,PM:P;
[NS2]   14: Z,T,M,F:INTEGER;
[MUØ]   15: BEGIN
[3V2]   16: P1.X:=1ØØ;P1.Y:=4ØØ;
[RV2]   17: P2.X:=5ØØ;P2.Y:=4ØØ;
[DV2]   18: P3.X:=3ØØ;P3.Y:=1ØØ;
[1T1]   19: RANDOMIZE;
[4P1]   2Ø: T:=DETECT;
[MT2]   21: INITGRAPH(T,M,'');
[D85]   22: MOVETO(P1.X,P1.Y);LINETO(P2.X,P2.Y);
[YP5]   23: LINETO(P3.X,P3.Y);LINETO(P1.X,P1.Y);
[PT2]   24: PM.X:=RANDOM(639);
[1U2]   25: PM.Y:=RANDOM(479);
[P31]   26: REPEAT
[FA2]   27:  Z:=RANDOM(3)+1;
[3C1]   28:  CASE Z OF
[Y71]   29:   1:BEGIN
[AA4]   3Ø:      PB.X:=ROUND((PM.X+P1.X)/2);
[GA4]   31:      PB.Y:=ROUND((PM.Y+P1.Y)/2);
[NPØ]   32:      F:=1;
[ZNØ]   33:      END;
[Ø81]   34:    2:BEGIN
[BA4]   35:      PB.X:=ROUND((PM.X+P2.X)/2);
[HA4]   36:      PB.Y:=ROUND((PM.Y+P2.Y)/2);
[SVØ]   37:      F:=14;
[ZNØ]   38:      END;
[281]   39:    3:BEGIN
[CA4]   4Ø:      PB.X:=ROUND((PM.X+P3.X)/2);
[KA4]   41:      PB.Y:=ROUND((PM.Y+P3.Y)/2);
[SPØ]   42:      F:=4;
[ZNØ]   43:      END;
[ZNØ]   44:  END;
[5P3]   45: PUTPIXEL(PB.X,PB.Y,F);
[MW1]   46: PM.X:=PB.X;
[SW1]   47: PM.Y:=PB.Y;
[K13]   48: UNTIL KEYPRESSED;
[522]   49: CLOSEGRAPH;
[KNØ]   5Ø: END.
```

Listing 15. Dieses sehr regelmäßige Fraktal erzeugt das Programm »chaos.pas«. Alle neu hinzukommenden Punkte bilden diese Formen. Bei genauem Hinsehen finden Sie allerdings einige Ausreißer.

Impressum

Herausgeber: Michael Scharfenberger

Redaktion

Chefredakteur: Ulrich Eike (ue)

Chefin vom Dienst: Gabi Börger (gb)

Leitender Redakteur: Wolfgang Hofner (wh)

Redaktion: Burkhard Müller (bm), Kerrin Uhlmann (uk)

Redaktionsmarketing: Ingo Everts

Textredaktion: Annette Baumhof (ab), Anne Brückner (br), Vera Eich (ve), Thomas Bruer (tb)

Redaktionsassistenz: Anke Ludwig, Helga Menne, Lydia Roitzheim

Gestaltung und Layout: Christiana Seiser, Michaela Küppers

Grafik: Leonhard Büttner

Fotografie: Heinz Harcuba

Anschrift Redaktion: Redaktion DOS International, Gruberstr. 46c, 8011 Poing bei München, Telefon: (0 81 21) 769-0, Telefax: (0 81 21) 7 90 46

Sie erreichen die Redaktion werktags telefonisch von 15 bis 17 Uhr.

Manuskripte, Programme bitte an die Redaktion DOS International, Gruberstr. 46 c, 8011 Poing bei München senden.
Für unverlangt eingesandte Manuskripte und Datenträger sowie Fotos übernehmen der Verlag und die Redaktion keine Haftung. Die Zustimmung zum Abdruck wird vorausgesetzt. Das Urheberrecht für veröffentlichte Manuskripte liegt ausschließlich beim Verlag. Nachdruck sowie Vervielfältigung oder sonstige Verwertung von Texten nur mit schriftlicher Genehmigung des Verlages. Für veröffentlichte Programme und Basteleien übernehmen wir weder Gewähr noch Haftung. Schaltungen und verwendete Bezeichnungen müssen nicht frei von gewerblichen Schutzrechten sein. Namentlich gekennzeichnete Fremdbeiträge geben nicht in jedem Fall die Meinung der Redaktion wieder.

Anzeigenverkauf

Gesamtanzeigenleitung: Stefan Grajer
Anzeigenverkauf für PLZ 6 – 8
DMV-Verlagsbüro München, Gruberstr. 46 c, 8011 Poing bei München, Telefon : (0 81 21) 769-0, Telefax: (0 81 21) 769-399
Leitung: Britta Fiebig 769-375
Anzeigenverkauf: Peter Schätzle 769-378, Ilona Sehm 769-370, Christoph Knittel 769-376, Katharina Küspert 769-374

Anzeigenverkauf für PLZ 1, 4, 5:
DMV-Verlagsbüro Eschwege, Fuldaer Str. 6, W-3440 Eschwege
Leitung: Thomas Goldmann
Karina Ehrlich (0 56 51) 809-371, Telefax: (0 56 51) 809-444
Bernd Heckmann (0 56 51) 809-381,
Christina Wabra (0 56 51) 809-390,
Sylvia Stephani (0 56 51) 809-380

Anzeigenverkauf für PLZ 2 + 3:
DMV-Verlagsbüro Hamburg, Kunaustr. 4 a, 2000 Hamburg 65, Telefon: (0 40) 6 00 30 75-77, Telefax: (0 40) 6 01 54 75
Leitung: Sylvia Ehrenpfordt
Anzeigenverkauf: Sandra Neubäumer
Coordinating Manager Asia: Ralph Streng
Coordinating Manager USA: Tobias J. Hoffmann

Repräsentanten im Ausland:
Great Britain: Huson European Media, Gerry Rhoades-Brown, 10-11 The green Business Centre, The Causeway, Staines, Middlesex, TW18 3AL, Telephone No: 0784 46 99 00, Fax No: 0784 46 99 96

Taiwan: PRISCO. Mr. Parson Lee, 2F, No.17, Alley 7, Lane 217. Chung Hsiao E. Rd. Sec. 3, Taipei, Taiwan, R.O.C., Phone: [886] -2-711 48 33-6, FAX: [886]-2-741 51 10

Hong Kong: ELECTRONIC (HK) PUBLISHING CO: Mr. Herman Lau. 1301 Po Sang Bank Building, 33 Argyle Street, Kin., Hong Kong, Tel.: 391 2977, 391 5088, FAX: 852 789 8027, Cable: ELPUB

Singapore: Major Media Private Ltd., Mr. Hoo Siew-Sai, 6th Fl., 52 Chin Swee Road, Singapore 0316. Phone (65)-738 0122, Fax: (65)-738 2108

Anzeigenverwaltung: Andrea Giese, Beate Kranz
Druckunterlagen-Disposition
Christina Wabra (0 56 51) 809-372
Anzeigenpreise
Es gilt die Anzeigenpreisliste Nr. 7 vom 01.12.1991

Verlag

Produktionsleitung: Uwe Siebert

Koordination: Petra Hugo

Fotosatz: Silvia Führer, Jürgen Ludwig, Martina Siebert

Montage/Reprografie: Margarete Schenk (Ltg.) Helmut Skoupy (Ltg.), Manuela Eska, Monika Martin

Titel: Mohamed Hawa

Werbegrafik: Uwe Kalischinsky

Anschrift Verlag: DMV Daten- und Medienverlag Widuch GmbH & Co. KG, Fuldaer Str. 6, 3440 Eschwege, Telefon: (0 56 51) 809-0. Telefax: (0 56 51) 809-333

Geschäftsführung: Michael Scharfenberger, Helmut Schmitz

Vertrieb: Verlagsunion Erich Pabel-Arthur Moewig KG (VPM), Friedrich-Bergius-Straße 20, 6200 Wiesbaden

Druck: Druckerei Jungfer, 3420 Herzberg

Bezugspreise:
DOS EXTRA erscheint dreimonatlich. Einzelpreis DM 18,–/sfr 18,–/öS 145,–/Lit 15.900

Abonnementpreise
Die Preise verstehen sich grundsätzlich einschließlich Porto und Verpackung.

Inland:
Bezugspreis 4 Ausgaben: DM 60,–

Europäisches Ausland:
Bezugspreis 4 Ausgaben: DM 70,–

Außereuropäisches Ausland:
Bezugspreis 4 Ausgaben: DM 80,–

Bankverbindungen:
Postscheck Frankfurt/M: Kto.-Nr. 23043-608
Raiffeisenbank Eschwege: BLZ 522 603 85, Kto.-Nr. 245 7008

Die Abonnementbestellung kann innerhalb einer Woche nach Auftrag beim DMV-Verlag, Postfach 250, W-3440 Eschwege schriftlich widerrufen werden. Zur Wahrung der Frist reicht der Poststempel. Das Abonnement verlängert sich automatisch um 6 bzw. 12 Monate, wenn es nicht mindestens 6 Wochen vor Ablauf beim Verlag schriftlich gekündigt wird.

Noch erhältlich:

DOS EXTRA Nr. 4
- Desktop-Publishing-Programm »SDTP«
- Kommandoeditor für MS-DOS
- Editor für Disketten und Festplatten
- Speicherresidente Programme verwalten
- Paßwort-Schutz für MS-DOS
- Tastatureingabe automatisiert
- Clock-Utility für alle Grafikkarten
- Cursor und Zeichensatz für EGA-Karte
- Fenstertechnik für Turbo C

DOS EXTRA Nr. 5
- 16 Farben für die CGA-Farbgrafikkarte
- Druckerzeichensatz selbstgemacht
- Mausgesteuertes Zeichenprogramm
- Editor in GW-Basic
- Setup-Programm für ATs
- Maschinensprache für GW-Basic

DOS EXTRA Nr. 6
- SBasic-Compiler im Quelltext
- Datenbankprogramm »Parat«
- Fensterfunktionen für GW-Basic und SBasic
- Analyse von Grafikkarten
- Maskengenerator für Binärdateien
- dBase-Datenbanken in Basic
- Superhirn für Quick Basic 4.0

DOS EXTRA Nr. 7
- MS-DOS-Benutzeroberfläche Easydos
- Benutzeroberfläche für SBasic
- Sprite-Editor für Farbgrafikkarten
- DOS-Packer
- Virenschutzprogramm
- Weltkarte im Computer
- Einfacher Prozessor-Emulator

DOS EXTRA Nr. 8
- Makroassembler SAM III
- Exe2bin-Ersatz durch »exe2com«
- Speicherverwaltung unter MS-DOS

DOS EXTRA Nr. 10
- Units-Bibliotheken für Turbo Pascal: Datum, Sound, Bildschirmausgabe
- Snapshot-Utility, Version 2.7
- TSR-Datei-Manager
- DOS-Assembler SAM III, Version 1.1
- Schnelles Kopierprogramm
- Komfortable Dateiumleitung
- Hercules-Grafik in Basic programmiert
- Patch für BGI-Treiber von Turbo Pascal

DOS EXTRA Nr. 12
- Kreative Grafik in Basic zum Mitmachen
- Grafik-Anpassung von Basic-Programmen
- Außergewöhnliche Fraktalgrafiken
- Das ultimative Hardcopy-Utility
- Testbild für Monitore in Basic programmiert

DOS EXTRA Nr. 14
- Komfortables Datensicherungsprogramm
- Zeichensatzeditor für 24-Nadel-Drucker
- Pascal-Quelltexte optimieren
- Utilities zur Bearbeitung von Dateien und Verzeichnissen

DOS EXTRA Nr. 15
- Der Nachbrenner für GW-Basic: SBasic Version 2.0
- Der Tastaturtreiber »SKeybgr« in der Version 4.0
- »Schiffe versenken« mit einem Quick-Basic-Programm
- Simulation von Tastatureingaben mit »Prokey«

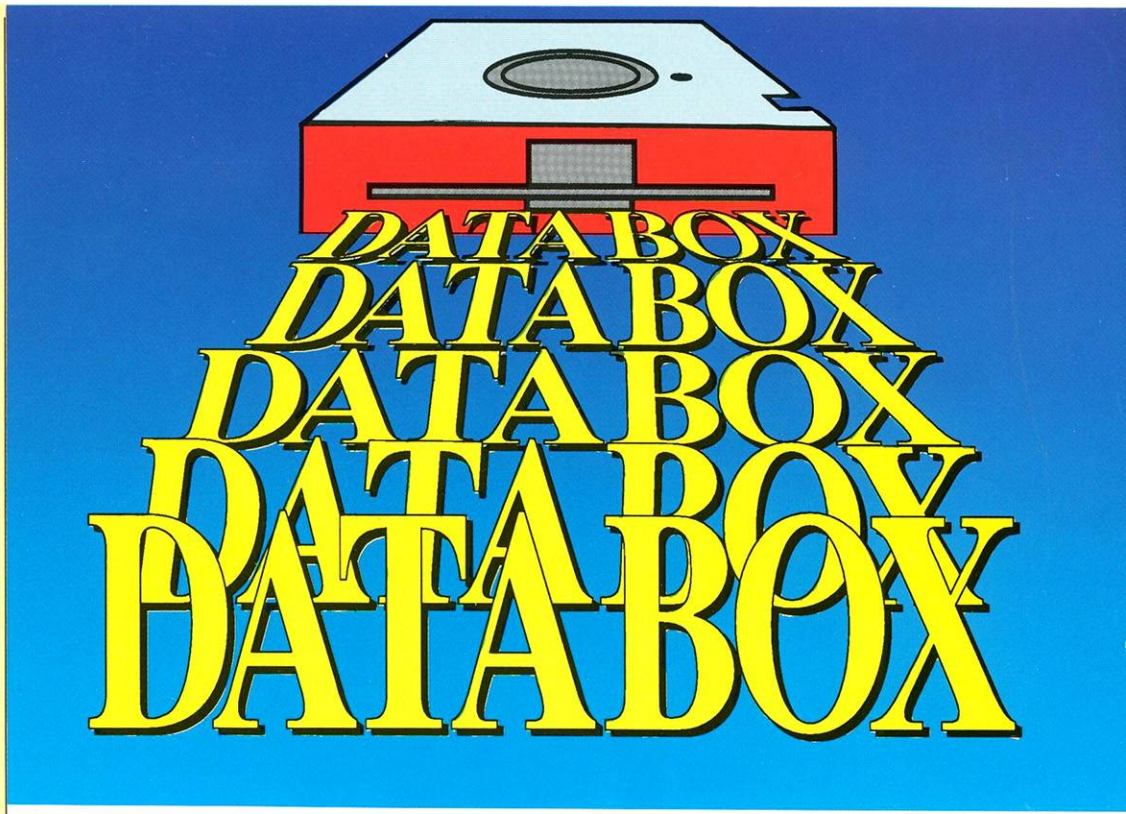
DOS EXTRA Nr. 16
- Batch-Compiler der S-Klasse
- Config-Editor beim Booten
- Assembler-Preprozessor für MASM und TASM
- GEM-API-Unit für Turbo Pascal
- SCheck-Prüfsummenprogramm in der Version 3.0

DOS EXTRA Nr. 17
- Transpiler: Basic nach C
- Flexibles Kopierprogramm für alle Formate
- Formatierer der S-Klasse
- BGI-Treiber für Super-VGA
- Kiloweise Hilfsprogramme für jeden Zweck

DOS EXTRA Nr. 18
- universelle Entwickleroberfläche
- PCX-Bilder in Basic
- Pretty Printer für Turbo Pascal
- Tips, Tricks und Tools
- kiloweise Hilfsprogramme für jeden Zweck

DOS EXTRA Nr. 19
- komplettes Raytracing-Programm
- Sprite-Programmierung
- Fraktal-Explorer
- Units zur Darstellung von 2D- und 3D-Grafiken
- Tips und Tricks zur Grafik

DOS EXTRA Nr. 20
- Objektorientierte Programmierung: Roboter im Labyrinth
- Programmierung der Soundblasterkarte
- Arbeiten mit dem integrierten Assembler
- BGI-Grafiktreiber intern
- Interrupts in eigenen Programmen nutzen



Die Software zu den DOS-EXTRA-Heften

Sagenhaft günstig, extra viel gute Software fürs Geld!

CeBIT'93 HANNOVER 24. — 31. 03. 1993 Halle 7, Stand E13/E19

DATABOX:

- **mehr als der übliche Softwareservice**
- **bringt alle Programme der DOS EXTRA auf Datenträger**

Für alle PCs unter MS-DOS



Den Aufbau der Databox-Diskette finden Sie in der Datei »read.me« erklärt. Geben Sie deshalb vor dem Start »type read.me« ein. Sämtliche Pascal- und C-Programme liegen auch in einer compilierten Form für Hercules- und Farbgrafikkarten vor. Es ist allerdings nicht gewährleistet, daß diese Versionen auf Hardware, die nicht vollständig kompatibel zum IBM PC ist, problemlos laufen. In diesem Fall müssen Sie den Quellcode nach der Anpassung mit dem entsprechenden Compiler neu übersetzen.
Die Basic-Programme arbeiten mit dem GW-Basic-Interpreter von Microsoft einwandfrei. Bei Interpretern und Compilern anderer Hersteller kann es zu Problemen bei der Programmausführung kommen, sofern diese nicht voll kompatibel sind.

jeweils 2 Disketten 5 1/4“
1 Diskette 3 1/2“

35,– DM (unverbindliche Preisempfehlung)
Wenn Sie über den DMV-Bestellservice bestellen, gilt folgendes:

Inland:		**Ausland:**	
Einzelpreis	35,– DM	Einzelpreis	35,– DM
zzgl. Versandkosten	4,– DM	zzgl. Versandkosten	6,– DM
Endpreis	**39,– DM**	**Endpreis**	**41,– DM**

Unabhängig von der Anzahl der bestellten Produkte berechnen wir für das Inland 4,– DM bzw. für das Ausland 6,– DM Porto und Verpackung.

– Bitte benutzen Sie die Bestellkarte. –

DOS EXTRA Nr. 21

- Programmieren mit der Assemblersprache
- SAM 4.0 – Der DOS-Makroassembler
- Dateiauswahlboxen leicht selbstgemacht
- Geräteprogrammierung intern
- Programmierung von Laserdruckern

DMV-Verlag · Postfach 250 · 3440 Eschwege

DMV Daten- und Medienverlag

Die Quelle Ihrer Kompetenz



Was Sie schon immer über Windows wissen wollten:

Zuverlässige Informationen von A – wie Aktuelles bis Z – wie Zubehör bietet Ihnen jetzt WinDOS, das monatliche Windows-Magazin der DOS International für den kompetenten Windows-Anwender.

Ab sofort im Handel!

CeBIT'93 HANNOVER
24. — 31. 03. 1993
Halle 7, Stand E13/E19

DMV-Verlag • Postfach 250 • 3440 Eschwege

DMV Daten- und Medienverlag