DOKTOR A.TARI

SPIELER/GESCHOß-GRAPHIK

Hier zunächst einige Vorbemerkungen für Bandbenutzer und Besitzer von 16K-Maschinen:

Für 16K-Benutzer:

Alle Programme auf diesem Band außer den letzten 2, laufen auf 16k-Anlagen. Das letzte mit 16K lesbare Programm ist das mit dem ATARI LOGO als Beispiel für zwei Spieler nebeneinander.

Vorbemerkung zum Begleittext:

Die beigefügten Programme sind ohne diesen Text eher langweilig. Erst mit Erklärung ergeben diese Player Missile Graphiken einen Sinn. Der ideale Zustand ist dann erreicht, wenn man seine ersten eigenen PM-Programme geschrieben hat.

ALLGEMEINES :

Zunächst machen wir uns daran, ein Spiel zu schreiben. Sehr beliebt sind Verfolgungsspiele, die gewöhnlich niedliche kleine Figuren und ein Labyrinth benutzen. Ein solches Programm soll hier erstellt werden, ein Spiel ähnlich dem bekannten PAC MAN.

Zum Entwerfen der Hintergrundumrisse und Figuren gehört auch eine Portion Phantasie. Einige der jetzt entstehenden Gestalten werden in den weiteren 14 Programmen und einigen Demos immer wieder benutz. Alle Beispiele kann man mit eigenen Ideen ergänzen und verändern.

Was bedeutet"Spieler" und "Geschoß"?

Diese Bezeichnungen wurden aus dem ursprünglichen Anwendungsfeld solcher Figuren übernommen, als nähmlich immer ein Schütze mitspielte, der per Kanone Geschosse abfeuerte. Inzwischen macht man alles mögliche mit diesen Elementen, hat aber die alten Bezeichnungen beibehalten.

Programm 1:

Der Neuling sollte es mehrfach genau ansehen und sich gedanklich später darauf beziehen können.

Der Monitor erscheint in Beispiel 1 zuerst schwarz, weil der Bildprozessor anfangs abgeschaltet wurde, der sogenannte ANTIC-Ship, der zweierlei bewirkt:

1. Er macht den Ablauf ca 30% schneller,

2. Er läßt den Hintergrund,z.B. Sterne, plötzlich erscheinen, was eindrucksvoller ist, als eine allmähliche Aufzeichnung zu machen.

Man kann diese Methode mit der 2. vergleichen, bei der ein Hintergrund langsam aufgebaut wird und dann nach Neigung wählen. Wenn man die 1. Version , mit ANTIC, vorzieht, geht man so vor:

An dem Punkt, wo der Monitor ausgeblendet werden soll, programmiert man
100 ON=PEEK(559):POKE559,0
(hier der Code zum Zeichnen des Hintergrundes)
500 POKE 559,ON.

Später wird erklärt, welches die korrekte Codierung für "ON" ist; man kann aber jederzeit einfach die alten Werte abstellen, wie oben geschehen. Die Zahl in Speicherplatz 559 muß später noch geändert werden, damit die PMG richtig läuft.

Zwei Prozessoren:

Es stimmt, das der ATARI 2 Microprozessoren besitzt. Einer davon ist ANTIC, der die Aufgabe hat, das Monitorbild aufzubauen. Er arbeitet in Text und in Graphik. Wird er eingesetzt, so nimmt er Maschinentakte in Anspruch, die dem Hauptprozessor dann fehlen. Deshalb läuft der Computer bei abgeschaltetem ANTIC , d.h. wenn der Bildschirm leer bleibt, (559=0) schneller.

Wir sind noch immer in Programm 1. Der Computer hat jetzt ein Bündel "Sterne" aufs Bild gebracht, einige in rot, einige in blau. Wenn wir nähmlich einzelne Punkte in Graphikgang 8 ansprechen (PLOTTEN), kann der Monitor ein blaues oder ein rotes "Pixel" auf dem Schirm leuchten lassen. Diese kleinen Pixels in rot und blau kann man bei sehr genauem Sehen auf der Mattscheibe gut unterscheiden.

Diese Hintergrundpunkte nennt man Spielfeld, wenn von PMG die Rede ist. Bei einem Blick auf die Graphik-Tabelle im BASIC-Handbuch stellt man fest, daß die meisten Graphikgänge 1-4 Farben zulassen. Die Gänge 9-11 haben mehr. Doch in folgendem ist nur von vieren die Rede.

Farbe 1 (Color1) in Verbindung mit dem Aufzeichnen von Umrisser auf dem Bildschirm bildet das "Spielfeld 1", entsprechend gilt das für Color 2 und Color 3. Das 4. Spielfeld ist eigentlich die Farbe 0, die sogenannte Hintergrundfarbe, die am Anfang eines Graphikgangs erscheint. All diese Farben sind steuerbar, und der Umgang mit ihnen soll jetzt erklärt werden.

Der Rand ist eine Sache für sich. Gewöhnlich setzt man ihn auf die Farbe des Hintergrundes, sodaß der gesamte Monitor gleich aussieht. Die Farbsternchen reichen

eigentlich nicht ganz bis zur Bildkante, da aber die Spieler sich überall hin bewegen können, erscheint das Spielfeld größer, wenn der Rand in derselben Farbe gehalten wird wie die Mitte.

Endlich sind da auch noch zwei Spieler auf dem Bild. Sie bewegen sich. In der Horizontalen geht das mit BASIC sehr leicht. Beim lesen der Programmlistung stellt man fest, daß 5 Zeilen (400-440) genügen, um alle Geräuche und Bewegungen zu steuern.

Als nächstes wächst die Spielergröße an. Das wird ebenfalls durch einfaches Einfügen einer Zahl erreicht; Erklärung weiter unten.

Schließlich :

Das Raumspiel endet mit dem Abfeuern einer Rakete von einem Raumschiff auf das andere, bis beide zerstört sind und das Spiel abgeschlossen ist. Wir haben also folgende Elemente benutzt:

Einen Spieler, ein Geschoß, den Hintergrund, die Spielfelder 0-3, einen Rand, und Größenvariationen.

Soweit ist die Szene bereits komplett, jedoch müssen wir uns klarmachen, daß wir noch nicht soweit waren, ein richtiges Spiel wie das Verfolgungsspiel herzustellen.

Solche Programme werden in Maschinensprache mit speziellen Displayroutinen für sehr feine Graphiken geschrieben. Hier sollen nur alle Begriffe demonstriert werden, die später im Programm zu lernen sein werden.

Fortgeschrittene können das Spiel erweitern und viel Spaß damit haben. So wie es hier steht, paßt es noch in 16K-Anlagen, was wir auch anstrebten.

Folgende Spielregeln gelten:

Zuerst die Spielstäbe in die Steckdosen 1 und 2 stecken. Jeder Spieler bewegt jetzt seine Figur und versucht, Punkte zu erzielen. Ein Punkt ist dann erzielt, wenn 1 Spieler eine der beiden Energiepillen aufschluckt, die ab und zu erscheinen.

5 Punkte gibt es, wenn man seinen Gegner schluckt. Das geschieht, wenn einer als erster die blaue Aufladezone erreicht, die in der Mitte des Labyrinthes liegt. Der betreffende Spieler wechselt dort die Farbe und jetzt einen Punkt machen, sobald er den anderen berührt. Wenn auch der andere Spieler die Aufladezone erreicht, kann er die Farbe des ersteren wieder auf normalen Stand bringen.

Wir fangen beim Punktstand 3 für beide an, um zu zeigen, daß nicht unbedingt bei 0 begonnen werden muß. Die Spielstrategie besteht im Ausnutzen leichter Einzelpunktchancen und gleichzeitigem Anzielen des großen Gewinns von 5 Punkten.

So simple dies Spiel erscheint, so vollständig enthält es doch alle Elemente wirklicher Verfolgungsspiele. Alle nötigen Ergänzungen benötigen nur der Phantasie des Lernenden und natürlich der Geschwindigkeit. Verfolgungsspiele werden in Maschinensprache geschrieben, dadurch erreichen sie hohe Geschwindigkeit.

Man achte jetzt genauer auf die Bewegungen der Spielelemente:
Sie sind langsam, obwohl sie mit Maschinensprachen-Routinen programmiert sind (die später erklärt werden). Das liegt daran, daß diese Routinen per BASIC-Sprache angerufen werden.

Weiter fällt etwas auf, wenn die Spieler die Labyrinthwände berühren.

Zunächst aber weiter zu Beispiel 3:

Spielfelder.

Es handelt sich um ein sehr einfaches Programm. Alles was nicht als Spieler oder Geschoß anzusehen ist, gilt als Spielfeld, wie oben schon gesagt. Spielfelder werden durch das PLOTTEN von Punkten durch PLOT und DRAW- Befehlen erzeugt bzw. durch direkte POKE-Befehle, die Daten in den Bildbereich des Spielers bringen. Vergl. hierzu das Handbuch.

Durch drücken von 1 auf der Tastatur und Anschließen eines Spielhebels in Stecker 1, kann man nun bei entsprechender Bewegung des Hebels die Farbe 1 aufs Bild bringen. Dann drückt man 2 und trägt Farbe 2 auf, entsprechend Farbe 3. Schließlich drückt man 0 und arbeitet mit Farbe 0. Diese Farbe erscheint als Löschvorgang, wird aber in Wirklichkeit auch "aufgetragen", und zwar als Hintergrundfarbe, die alles andere in Hintergrund verwandelt.

Der Zweck der obigen Übung war es, die Spielfelder zu demonstrieren. Spielfeld 0 ist die Hintergrundfarbe, Spielfeld 1 ist alles, was in Farbe 1 erscheint, und so weiter.

Seite 53 des BASIC-Reference-Manual(ACC Berlin) zeigt , welche Spielfelder bei welchen Graphikgängen möglich sind.

Anmerkung zu den Farben:

Alle Farben, die in den Beispielen benutzt werden, lassen sich ändern:

Spielfeldfarben

Farbe Nr.	POKE-Adresse
0	710
1	708
2	709
3	711
Rand - und Textfenster	712

Der korrekte POKE-Wert ist:

COLOR * 256 + Helligkeit

Wobei der Farbcode von 0-15 geht und die Helligkeit von 0-14 (gerade Zahlen).

Beispiel 4:
(Spielfelddrucker):

Natürlich braucht jedes Bildprogramm einen Hintergrund, außer wenn es ein reines Textprogramm ist, was aber bei PMG wohl nicht vorkommt. Das bedeutet, daß wir einen bequemen Weg zum Aufbau von Feldern und ihrer Projektion suchen müssen.

Möchte man z.B. ein Weltraumspiel mit Bergen, feindlichen Basen und einigen Schiffen zum Beschießen haben, so muß man diese Elemente auf den Monitor PLOTTEN, bevor das eigentliche Spiel beginnt.

Man entwirft so etwas normalerweise auf Millimeterpapier und rechnet jeden PLOT-Punkt aus. Da dies unbequem ist, haben wir ein kleines Druckprogramm entworfen.

Für Plattenbenutzer gibt es keine Probleme. Sie drücken bei Erscheinen des Beispiels 4 "L" und dann "D", sodaß ein zuvor hier gespeichertes Bild als Testbild erscheint. Dabei fällt ein blitzender Punkt auf, ein Läufer.

Bandbenutzer müssen eine Zeichnung programmieren und sie abstellen, bevor die LOAD-Anwahl funktioniert. Das ist eben der Punkt, der die Zeichnung bildet, wenn man ihn über den Schirm bewegt.

Man hat die Wahl zwischen den Spielfeldern 0-3. Zuerst muß man feststellen, wieviele Spielfelder der gewählte Graphikgang überhaupt zuläßt. Graphikgang 5 z.B. kann mit allen 4 Farben arbeiten, bei Gang 6 hat man nur 2 zur Auswahl.

Man drückt die Nummer der Spielfeldfarbe und benutzt dann die Läufertaste (nicht CTRL!), um den Läufer zu bewegen. Um zu löschen, PLOTTET man die Hintergrundfarbe, 0, ein. Jetzt zeichnet man ein Gebäude oder ein Raumschiff. Diese Zeichnung kann mit dem Editor auf zweierlei Weise abstellen:

1. Man kann "S" drücken und dann "T" oder "D" für Band- bzw. Plattenaufnahme. Bandbenutzer sollten daran denken, den Anfangspunkt der Aufnahme auf das Band zu zeichnen. Besser ist es, ein freies Band zu verwenden. Plattenbenutzer finden ihr Programm immer unter dem selben Namen auf der Platte, PFDATA. Wenn man also die Zeichnungen aufheben möchte, benütze man den Namenslistenbefehl "E", von DOS, um die Daten sinnvoller bezeichnen zu können.

2. Will man die Zeichnung im Programm verwenden, braucht man nur eine einfache Subroutine. In der Listung dieses Programms findet man eine sehr geschickte Lösung mit Strings. Daraus kann jeder Teile kopieren, der ein eigenes Programm schreiben will. Hier zeigen wir aber auch einen einfachen Weg zum PLOTTEN der Daten:

Zuerst folgende Anmerkung:
Jede Farbe eines Feldes muß separat aufgebracht werden. Außerdem soll der Umriß vielleicht auch extra erscheinen, zum Beispiel ein Bau mit etwas Bodenfläche.

Dann werden eventuell andere Figuren wie Personen, Raumschiffe, Planeten gebraucht. Auch sie sind extra zu PLOTTEN.

Die Adressdaten für Umrisse bekommt man durch drücken von SELECT, danach wartet man ca 10 Sekunden. Im Textbereich des Monitors erscheinen jetzt Datastatements, welche die ansprechbaren Bildpunkte bezeichnen. Die linke obere Ecke des Quadrats, die den Umriß begrenzen würde, wenn er in Farbe erschiene, hat 1,1. Die Zahlen bedeuten X,Y gemäß dem Standart-Koordinatensystem, das bei der ATARI gewöhnlich für Graphik benutzt wird.

Um alle Daten pro Farbe zu sehen, muß man nach jedem Stop der Datenprojektion START drücken. Wenn mehr als 199 Punkte pro Farbe in einem Umriß getastet werden, meldet das Programm "zuviele Data", denn mehr als 199 konnten wir nicht auf 32K unterbringen.

Das Problem wird selten auftauchen. Wenn doch, tauscht man an einigen Stellen einer Figur die Farben aus.

Die Nummern der Datastatements sollten so abgeändert werden, daß sie ins übrige Programm passen. Man kann das so ordnen, daß etwa Bäume bei 5000 anfangen, Gebäude bei 6000 u.s.w. . Hier ein Beispiel, daß mit dem Spielfelddrucker ausgegeben werden kann:

Dies sind keine wirklich verwendeten Daten, sie zeigen nur die Form, die solch ein Programm haben kann.

Schlußbemerkung zum Spielfelddrucker :
Er verwendet eine Gebrauchs-Displayliste, die man genau studieren sollte. Die Methode wird im ersten Abschnitt dieses Lernprogramms erläutert.

FARBEN, Beispiel 5:

Bei Bandbenutzung gelangt dieses Programm etwas eigenartig aufs Bild. Man drückt einfach RESET und tastet RUN, um den Monitor zu korrigieren.

Es war zuvor schon vom Wechsel der Spielfeldfarbe die Rede (708-712). Folgendes Verfahren erlaubt die Auswahl der Farbe, ohne daß man die auszuwählende Adresse zuerst ausrechnen müßte. Die aufgeführten POKES gelten allerdings für die Spielerfarben, die ja woanders untergebracht sind (704-707 und 711). Darüber hinaus ist es das Hauptanliegen des Beispiels, den Leser mit dem Spielern und Geschossen bekannt zu machen. Auf dem Bild erscheinen Farbstreifen, jeder Streifen representiert einen halben Spieler, die untere Hälfte ist abgeschaltet, damit man die Farbinformation sehen kann.

Diese Streifen werden nur abgebildet, damit die Spielerfarben für den Wechsel besser zu erkennen sind. Man kann die kleinen Farbvierecke quer durch jeden Streifen erkennen. Im nächsten Beispiel wählen wir bestimmte Pixels aus und bauen einen brauchbaren Umriß auf. Denn jetzt wird die obere Hälfte jedes Spielers und Geschosses angeschaltet (alle 8 Pixels in Farbe, d.h. mit 1 programmiert) und die untere Hälfte des Streifens ist ausgeschaltet (d.h. 0 auf allen 8 Pixeladressen).

Nun fragt sich, was davon welcher Spieler ist. Spieler 0 ist ganz links, Spieler 3 ganz rechts. Ganz rechts stehen die entsprechenden Geschosse 0-3, zu jedem Spieler gehörend.

Nun wird ein Spielstab in Steckdose 1 angeschlossen. Das erscheinende Sternchen wird über irgendeinen Spieler gerückt.

Jetzt bewegt man den Hebel aufwärts, wenn man den Farbwert erhöhen will, abwärts, wenn er verblassen soll. Das bedeutet, daß man z.B. einen schönen Rotwert, z.B. Code 45, in irgend eines der Farbregister laden kann (POKE), die von 704-712 gehen, und dadurch dies Rot für die entsprechende Spielfigur bekommt.

Sobald man eine gewünschte Spielerfarbe bekommen hat, schiebt man das Sternchen zur nächsten Figur und verfährt dort entsprechend. Wenn alle 4 Spieler versorgt sind, setzt man die Hintergrundfarbe fest,

indem man das Sternchen ganz nach rechts schiebt und den Abzughebel drückt. Um diese Änderungen dauerhaft zu machen, muß man sie ins Programm einfügen.

Noch haben wir nicht über Farbregister für Geschosse gesprochen, denn es gibt sie nicht. Jedes Geschoss nimmt automatisch die Farbe des Spielers der gleichen Nummer an. Hier eine Übersicht :

REGISTER	BENUTZT FÜR SPIELER
704	Spieler 0
705	Spieler 1
706	Spieler 2
707	Spieler 3
711	Spieler 4

Dabei bedeutet Spieler 4 die Kombination aller 4 Geschosse. Diese können bei Bedarf für einen 5. Spieler (Nr.4) zusammengefaßt werden. Das wird später noch erklärt.

Nun wird SELECT gedrückt. Die Geschosse rücken zusammen und nehmen die Farbe an, die in Register 711 eingetragen worden war.

Um den Farbdrucker für Spieler 4 anzuwenden, bewegt man das Sternchen dorthin und benutzt den Abzugsknopf wie zuvor erklärt. Ein zweiter Druck auf SELECT würde den 5. Spieler in separate Geschosse verwandeln.

Die beschriebene Methode ist praktisch immer da, wo man Farben auswählen möchte.

Welches ist der Spieler?

Beim Ablauf des nächsten Beispiels, Nr. 6, erscheinen 2 Roboter auf dem Bild. Man muß nun herausfinden, welches von beiden ein echter Spieler ist und welches nur durch PLOTTEN auf den Monitor gezeichnet wurde.

Durch Augenschein kann man keinen Unterschied feststellen. Erst wenn man einen Spielhebel in Stecker 1 anschließt und die Figuren zu bewegen versucht, stellt sich heraus, daß der Spieler sich ohne Druck auf den Abzugsknopf bewegt, die Spielfeldfigur Bei Druck auf den Abzugknopf. Genauer gesagt, wird der Spieler durch eine kleine Routine in Maschinensprache bewegt.

Man wird später sehen, daß man sogar mit BASIC einen Spieler ziehmlich schnell in der Vertikalen und unmittelbar in der Horizontalen bewegen kann.

Bei diesem Programm ist es interessant, die Farbe im Programmteil des betreffenden Spielers zu ändern (POKE 704,Nr...) und dann den Spieler genau über die Spielfeldfigur zu setzen. Man erkennt erst dann richtig die Vorzüge der echten Spieler. Würde man 2 geplottete Umrisse übereinander schieben und sie dann wegrücken, so müßte man beide neu zeichnen, um die hinterlassenen Löcher aufzufüllen. Dies Beispiel sollte einige Hinweise zum Einsatz der Spielerstreifen geben. Statt des Roboters hätte man natürlich andere Figuren aufbauen können, die in die begrenzte Breite des Spielerstreifens passen.

Später wird gezeigt, wie man breitere Spieler konstruieren kann. Jetzt geht es zunächst generell um den Aufbau eines Spielers.-

Mit dem nächsten Programm beginnt der technische Teil des Kurses. Zuerst weisen wir darauf hin, daß das letzte Bild noch auf dem Monitor steht. Das kommt einfach daher, daß Spielfelder, die einmal mit PLOT eingerichtet wurden, solange stehen bleiben, bis man sie durch neue PLOT-Daten überschreibt oder bis man einen anderen Graphikgang abruft.

Wir laden das folgende Programm , während der Roboter (Spielfeld) noch auf dem Schirm war, sodaß er auch bei Beginn dieses Beispiels noch bleibt. Das geht ähnlich wie beim addieren von 32 zu einem Graphikabruf (Vergl. BASIC-Handbuch).

Daraus ergeben sich 4 interessante Möglichkeiten. Man kann Hintergründe mit dem einen Programm aufbauen und dann die Hauptprogrammschleife separat einladen. Dadurch passen Großprogramme in kleinere Speicherbereiche. Auch Spieler bleiben auf dem Bildschirm. In diesem Fall haben wir die Speilerumriße jedoch geänert.

Wieder ist Spielstab 1 anzuschließen. Mit ihm läßt sich der Umriß auf dem Monitor in jede Richtung bewegen, auch diagonal. Diese Routine ist vielseitig, weil sie alle die folgenden Grundlagen

erklärt:
1. wie man einen Spieler aufbaut,
2. wie man diese Figur vertikal bewegt,
3. wie man sie horizontal bewegt.

Das Beispiel kann in jedes beliebige Programm kopiert werden, gegebenenfalls unter Modifikation nach jeweiligem Bedarf. Man sollte es auf einem seperaten Band oder einer Liskette aufbewahren.

Das vollständige BASIC-Programm für dieses Beispiel befindet sich am Schluß der vorliegenden Texte. Wir werden die Hauptpunkte hier besprechen.

Das Programm beginnt damit, daß es auf Zeile 140 springt, um einige stets notwendige Standardverbereitungen durchzuführen. Zeile 160 setzt, wie man leicht erkennt, die Farbe von Spieler 0, die wir für den späteren Umriß (z.B. Raumschiff) benutzen wollen.

Die Zeilen 170-190 sehen so aus :

```
170 PMB =PEEK (106)-16
180 POKE 54279,   PMB
190 PMBASE =  PMB *256
```

Sie reservieren Platz für Spieler/und Geschoßumrisse, die noch einzufügen sind. Hierüber kann an dieser Stelle nicht erschöpfend gesprochen werden. Spieler und Geschosse werden auf dem ATARI mit Hilfe der sogenannten DMA (Direkte Speicheradressierung) konstruiert. Dieser exisiert auf allen größeren Computern (bei APPLE nicht!). Das bedeutet einfach, daß bei Einschaltung der DNA vom Computer alle Informationen , die er in einen Speicherbereich beginnend bei PMBASE vorfindet, auf den Monitor gebracht werden.

Diese Informationen werden horizontal (in Richtung X) angeordnet, entsprechend den Werten in den horizontalen Registern (53248- 53255). Ihre Ausdehnung in der X-Richtung wird durch die in den Größenregistern (53256-53260) gespeicherten Werte gesteuert (s.unten). Ihre Ausdehnung in Richtung Y wird durch 2 Faktoren kontrolliert. Der 1. ist, wieveil Pixels für ihren Umriß benutzt werden. Man muß wisen, daß Spieler so gebildet werden können, daß jede Zahl im Speicher als 1 Pixel erscheint oder so, daß sie als 2 Pixels erscheint (Eintelzeilen-Auflösung bzw. Doppelzeilen-Auflösung). Ihre Anordnung auf dem Monitor in der Richtung Y hängt einfach davon ab. an welcher Stelle im Streifen man Pixels anschaltet. Das alles wird noch einmal an Hand der Beispiele erklärt.

Jetzt betrachten wir wieder die Programmzeilen 170-190. Zeile 170 übernimmt den Wert von Stelle 106, der Stelle, die ganz oben im Speicherteil des Benutzers steht (als Anzahl von 256 - Byte-Seiten), und subtrahiert von ihm die jeweils vom Benutzer vorbestimmte Anzahl von Seiten. Diese Anzahl wir dann im Register 54279 eingespeichert (POKE). Dieses gibt an, wo die Spielerdaten beginnen.

Schließlich übernimmt Zeile 190 die PMG in Form von Seitenanzahlen und wandelt sie um in eine reguläre Speicheradresse, PMBASE. Diese Zahl werden wir oft benutzen.

Die von uns subtrahierte Zahl ist 16, doch ändert sich das je den benutzten Graphikgängen. Die Wahl des zu subtrahierenden Wertes wird in Beispiel 9 erklärt.

Als nächstes bringen wir auf Zeile 200 eine 3 in 53277. Dies Register nennt sich GRACTL; hierhin eine 1 zu bringen (nur für Geschosse)bzw. eine 3 (Spieler und Geschosse) ist ein Schritt zum Einschalten der oben erwähnten DMA.

Zeile 210 nennt die anfänglichen X- und Y- Positionen, auf denen der Spieler im Bild erscheinen soll. In Zeile 220 wird der Spieler 0 auf normale Position gerückt, nähmlich 0. 53256 hätte ohnehin enthalten, doch da wir viele Programme hintereinander laufen lassen, haben wir zur Sicherheit auf 0 geprüft.

Folgende Auswahl ist hinsichtlich der Größe gegeben:

POKE-Befehl mit	Spielergröße
0	normale Größe
1	doppelte Größe
3	vierfache Größe

Folgende Register stehen zur Steuerung der Spieler-und Geschossgröße zur Verfügung:

(Man setzt in sie die entsprechenden Werte aus der obigen Tafel):

REGISTER	PARAMETER
53256	Größe Spieler 0
53257	Größe Spieler 1
53258	Größe Spieler 2
53259	Größe Spieler 3
53260	Größe aller Geschosse

Um die Geschossgröße zu bestimmen, muß man etwas rechnen:

Man sucht eunfach die gewünschte Größe für jedes Geschoss von der unten abgebildeten Übersicht aus und addiert dann die Zahl von allen Geschossen, die man benutzen will. Schließlich läd man die Summe in Register 53260. Will man z.B. einen normalen Geschossdurchmesser für 0, doppeltes Geschoss 1, vierfaches Geschoss 2 und doppeltes Geschoss 3, so muß man 0+4+48+64 =116 in 53260 laden. Wenn man nur normales Missile 0 und vierfaches M3 will, ergibt sich 192 und so weiter.

GESCHOSS (M)	NORMAL	DOPPELT	VIERFACH
0	0	1	3
1	0	4	12
2	0	16	48
3	0	64	192

Zeile 230 überträgt NULLEN in den Bereich für die Spielerdaten. Dadurch wird zuerst einmal der Spielerstreifen weggelöscht. Das erscheint zunächst überflüssig, ist aber doch notwendig, wie man später sehen wird.

Da das Löschungsstatement jedoch einige Sekunden verbraucht, sollte man es ganz vorne ins Programm bringen, bevor der Hauptteil anfängt. Dies wird in späteren Beispielen gezeigt.

Für unser jetziges Beispiel braucht man lediglich PMBASE + 1024 und PMBASE + 1280 zu löschen.

Die Übersicht mit der Überschrift Einzelzeilen-Auflösung beginnt oben mit dem Kasten "oberes Speicherende"; von dort fährt man eine bestimmte Zahl von Seiten, die von dem Wert in 106 subtrahiert worden sind, nach unten. Damit gelangt man zum unteren Ende der TAFEL. Diese Rechnung wurde in den Zeilen 170-190 durchgeführt. Sie wird in Beispiel 9 noch einmal erklärt.

Nachdem man PMBASE gesetzt hat, geht man im Speicher nach oben zu PMBASE + 768. Die nächsten 256 Bytes sind der Streifen, der die Geschosse enthält. Wenn irgendwelche Register in diesem Speicherteil nicht auf 0 stehen und wenn die Vorbereitungspokes zum Anschalten der DMA durchgeführt wurden, dann leuchten diese NICHT-NULL-WERTE im Bild auf, und zwar mit den Farbwerten, die 704 gespeichert hat.

Um den Umriß auf dem Monitor auf-und abzubewegen, bringt man jetzt Nicht-Null-Bytes nach oben oder unten in den Speicherstreifen.

Hat man z.B. PMBASE bei 30000 plaziert, dann erscheinen alle Nicht-Null-Bytes bei PMBASE +1030 (31030) am oberen Ende des Bildschirms, die selben Werte bei PMBASE + 1200 erscheinen nahe dem unteren Ende.

Entsprechend dieser Logik kann man sich vorstellen, was passiert, wenn man Zeilenwerte in PMBASE + 1400 transportiert. Man hätte jetzt ein Umriß projeziert, der innerhalb von Spieler 1 auftauchen würde, falls die DMA arbeitet. Davon zunächst nichts weiter.

Die Zeilen 240-300 bringen den Umriß in den Speicher, so wie es oben besprochen wurde. Wichtig ist das Verständnis der Zeilen 250-300 , die so aussehen:

```
250 FOR I= PMBASE +1024+Y TO PMBASE +1280
260 READ B: IF B=0 THEN 310
270 CNT=CNT+1
280 POKE I,B
290 NEXT 1
300 DATA 8,60,126,195,60,24,126,16,0,0
```

Der Umriß wird mit einem Abstand von "Y" in den Speicher gebracht, d.h. mit einem Abstand von Y vom oberen Bildrand.

Falls Y=0, steht die Figur ganz oben an der Kante. Ein Wert von Y=252 setzt sie nach ganz unten, da der Umriß 8 Ziffern belegt, die in den 256 Bytes langen Streifen des Speichers passen müssen. Auf einigen weiteren Zeilen wollen wir ausrechen, wohin die Figur gesetzt werden soll und dann diese Routine benutzen, um sie an den richtigen Platz zu bringen.

Um eine eigene Figur zu konstruieren, muß man die Kästchen ausfüllen, die der Figur entsprechen. Um daraus die "Summe" zu ermitteln, geht man einfsach durch jede Reihe und addiert die Werte aller ausgefüllten Kästchens. Wenn man diese Zahlen in den richtigen Speicherbereich einträgt (POKE) , dann erscheint der gewünschte Umriß auf dem Schirm.

Zurück zum Programmcode:

Zeile 310 schaltet schleißlich den Spieler ein. Der korrekte Wert wird durch Addition der gewünschten Elemente aus folgender Liste erreicht:

DMACTL (559)

FÜR	ADDIERE
Breites Spielfeld	3* ALTANATIV
Standardspielfeld	2* ALTANATIV
Schmales Spielfeld	1* ALTANATIV
Geschoss-DMA ein	4
Spieler DMA ein	8
Doppelzeilen-Auflösung	0* ALTANATIV
Einzelzeilen-Auflösung	16* ALTANATIV
Haupt-DMA ein	32

In diesem Fall wollen wir eine DMA für Standardspielfeld (2) + für Geschoss (4) und für Speiler (8) + der Haupt-DMA (32) sowie Einzelzeilen - Auflösung haben. Das ergibt eine Summe von 2+4+8+32+16=62. Deshelb der POKE 559.62.

Damit ist die Aufbereitung der Spielerfiguren beendet. Dieselbe Vorbereitungcodierung kann für die meisten PMG-Programme benutzt werden. Man muß nur die wenigen Änderungen für Auflösung und Spielfeldgröße berücksichtigen.

Einzelzeilen-und Doppelzeilen-Auflösung wird noch in Beispiel 10 besprochen.

Das Programm verzweigt nun auf die Hauptschleife auf Zeile 50, wo Spielstab 0 abgefragt wird. Die Zeilen 60 und 70 erhöhen X und Y über den ursprünglichen Stand bei Zeile 210 hinaus. Für das Programm bleibt jetzt nur noch, in die Richtungen X und Y zu gehen. Bei X geht das leicht, weil die ATARI eingebaute Lageregister für die X - Achse hat. Auf Zeile 110 wird einfach der neue X-Wert in das entsprechende Register (53248) geladen.

Es gibt folgende Register für die X-Achse:

Registernummer	Geschoss/Spieler*
53248	M0
53249	M1
53250	M2
53251	M3
53252	P0
53253	P1
53254	P2
53255	P3

* M0 bezieht sich auf Geschoß 0, P1 auf Spieler 1 u.s.w.

Um in der Y-Richtung zu wandern, müssen wir wir ähnlich vorgehen wie bei der Verschiebung des Spielfeld-Robotors in Beispiel 6. Zum Glück handelt es sich nicht um derart viele Elemente im Speicher. Der Robotor hatte über 100 Datenpunkte. Für unsere Spieler übertragen wir lediglich 8 Zahlen in den Speicher, nähmlich auf den Zeilen 340 - 480. Zuerst, in Zeile 350, wird der laufende Wert von Y mit dem letzten Y-Wert verglichen. Sind beide gleich, kehrt man einfach zurück. Zeile 360 besagt "Wenn Y jetzt größer ist als letztes Mal, springe auf die Aufwärtsroutine bei 430, andernfalls bewege nach unten." -

Y bewirkt bei Anwachsen nähmlich eine Abwärtsbewegung auf dem Bild (vergl. Seite 47 des BASIC-Handbuchs).

Beide Routinen für die Auf- und die Abwärtsbewegung ähneln der ursprünglich verwandten Routine zum Eintragen des Umrisses in den Speicher. Wir versetzen die selbe Figur einfach um eine Speicherstelle nach unten oder oben.

Beachten:

Wenn man die Zeichnung bewegt, kommen 7 von 8 Werten, die zuvor im Speicher waren, an vorher schon besetzte Stellen, und nur eine wird durch POKE auf eine vorher UNBESETZTE Position gebracht.

Der letzte Wert wird unverändert stehen bleiben, wenn man ihn nicht durch POKE löschen würde. Das geschieht auf Zeile 420 oder 480. In dem Fall ließe jede Bewegung eine breite Spur hinter sich Diese Bwegungen sollte man ausreichend üben, damit alles klar wird.

Nun zu den Paddels:

Es bleibt zunächst zu erklären, warum die Bewegung auf der X-Achse so langsam verlief:

Die Spielstäbe, die in Verbindung mit dem Programmcode benutzt sind, erlauben nur Bewegungen von jeweils einem Schritt in Richtung X. Paddels dagegen können in einem Zuge von 0-228 schalten. Man sollte also jetzt einen Paddel anschließen und noch einmal die Bewegung üben; sie verläuft jetzt in Sprüngen.

Beispiel 8:

Mondlandung

Es handelt sich um ein einfaches Programm, das die selbe Grundroutine hat wie Beispiel 7, jedoch mehr wie ein Spiel abläuft.

3 Dinge kann man aus Beispiel 8 lernen:

1. Es ergibt sich hier ein Streifen von flackernden Resten auf dem Bild, wenn das Programm noch in der Startphase ist. Das kommt daher, daß die DM (wie oben besprochen) nur teilweise angeschaltet ist. Sobald alle Vorbereitungen abgeschlossen sind, erscheint jedoch die genaue Figuration.

2. Man sieht aus dem im Anhang wiedergegebenden Programm dessen genaue Struktur. Diese Struktur sollte jeder auch bei eigenen Programmen bilden. Das Programm verzweigt auf verschiedene Subroutinen, die A) die Berge zeichnen, B) die Sterne zeichnen,

3. Spieler 0 konstruieren, wie in der oben erwähnten Schleife, D) die eigentliche Aktion durchführen (Bewegung und Abschuß)

Zu beachten ist auch ist auch der Einsatz von Begleittönen in der Hauptschleife, deren Programmierung keine Schwierigkeiten machen dürfte.

Dieser einfache Aufbau wird in den meisten Spielen diesen Typs verwendet. Zum Schluß erklären wir noch 2 Verfahren in Maschinensprache, die im Programm enthalten sind. Beide sind kurz und können so leicht in andere Programme übernommen werden.

Das erste dient zum Bewegen von Spieler 0 allein. Es fragt Spielhebel 0 ab und bewegt die Daten entsprechend im Streifen des Spielers 0. Man muß hierfür Spieler mit Einzel-Zeilen-Auflösung verwenden, deren Umriß also bei PMBASE + 1024 bis PMBASE + 1279 abgestellt ist. Die benötigten Programmzeilen kopiert man am besten mit LIST auf einen seperaten Datenträger und nimmt sie per ENTER bei Bedarf ins Programm. Vor dem Kopieren sollte man nicht vergessen, die übrigen Programmteile zu löschen.

Beispiel 9:

Ein dummes Programm.

Beim Abspielen dieses Beispiels trifft man den schon bekannten Roboter wieder an. Wenn man ihn jedoch bewegt (mit Hilfe derselben Routine in Maschinensprache wie oben gezeigt), so bleibt "0" auf dem Bild zurück.

Dies soll im Zusammenhang mit dem Aufbau des Player/Missile- Bereichs im Speicher erklärt werden.

Dazu ist die Programmlistung für Beispiel 9 im Anhang zu betrachten und die Methode, mit deren Hilfe wir den Wert für PMBASE ermittelten:

```
100 PMBASE =(PEEK(106)-16)*256
```

Auf Seite 45 des BASIC-Handbuchs steht eine Tafel, aus der man die Menge von Speicherplätzen (RAM) ersieht, die jeder Graphikgang benötigt. Wie die Zeichnung zeigt, muß man vermeiden, das die Daten der Spieler und Geschosse die übrigen Projektionsdaten verdecken. Erst recht muß man vermeiden, daß sie die Displayliste überschreiben, wie es den Bandbenutzern bei dem Farbbeispiel 5 passierte. Solbald dieses Programm durch RESET neu anlief, wurden die Displaylisteninhalte erfaßt.

Bei unserem jetzigen Beispiel sieht man die Spielerdaten, wie sie durch die Routine in Maschinensprache innerhalb des Spielerstreifens bewegt werden.

Man kann sich hier auch den Begriff "Umbruch" (WRAP AROUND) klar machen, wenn man den Spieler so lange nach oben rückt, bis er am unteren Bildrand wieder zurück kommt. Das funktioniert auch in der Horizontalen. Auch die Daten, die direkt auf den Monitor geschrieben werden, können auf diese Weise umlaufen.

Bei der früheren Erklärung dieser Methode sagten wir, daß der Wert 16 sich in andere Werte ändern kann, je nach Einzelfall. Warum ? -

Die hier vorliegende Programmlistung enthält nur den Wert 16 als abzuziehende Speicherplatzmenge, weil der hier benutzte Graphikgang 1 K oder weniger benötigt.

Andere Beispiele dagegen, etwa Nr. 8, Poken die Speichermenge um 40 Seiten zurück, undzwar deshalb, weil dort Gang 8 eingesetzt wird, der mehr Speicherplatz braucht.

Der einfachste Weg, Daten am falschen Ort zu vermeiden, besteht in der Benutzung der fogenden Übersicht. Spieler mit Einzelzeilenauflösung müsen stets an einer "1K-Grenze" beginnen, d.h. man muß von dem Wert in 106 8,16,24,32 oder andere Vielfache von 8 abziehen. Eine "SEITE" ist 256 Bytes groß (1/4 K). Handelt es sich um Spieler mit zweizeiliger Auflösung, so müssen deren Daten ebenfalls an einer 1K - Grenze anfangen, in diesem Fall wird 4,8,12,16 u.s.w. aus 106 abgezogen.

Die Doppelzeilen - Auflösung:

Beispiel 10:

Dies ebenfalls einfache Beispiel bringt das SQUIGLIY MONSTER ins Spiel. Es zeigt sich in 2 Größen. Die kleinere hat Einzelzeilenauflösung. Schlicht gesagt, wird jeder Wert im Datenbereich des Spielers als eine Zeile aufs Bild gebracht.

Der Datenbereich hat 256 Bytes, was mehr als genug ist, um den Monitor abzudecken. Bei genauem hinsehen kann man die einzelnen Pixels auf der Mattscheibe erkennen.

Die 2. Art, die Spieler und Geschosse zu konstruieren, ist die Doppelzeilenauflösung. Doppelzeilentechnik benutzt nur 128 Bytes pro Spieler, und die Maschine bringt für jeden im Datenbereich des Speichers 2 Zeilen aufs Bild. Die projezierte Figur erscheint daher dicker. Außerdem verbraucht man auf diese Weise weniger Platz im PM - Bereich.

Zum Abrufen des einzeiligen Spielers drückt man auf SELECT, zum Abrufen des zweizeiligen auf den START-Knopf.

Doppelzeilenspieler werden ähnlich aufgebaut wie einzeilige, doch mit folgenden Unterschieden:

1. Anstatt 16 zu der nach 559 zu bringenden Summe zu addieren, addiert man 0. Der sich ergebende Wert sagt der Maschine, wo die Daten Pro Spieler und Geschoß stehen. Unser Beispiel wechselt zwischen den beiden Auflösungen durch diesen einen POKE hin und her.

2. Anstatt den Spielerumriß mit POKE in PMBASE + 768 bis PMBASE 2048 zu bringen wie oben, plaziert man jetzt die Daten in PMBASE+ 384 bis PMBASE+ 1024.

Beispiel 11:

Es bleibt nun nicht mehr viel zu lernen, bevor das Pac Man - Spiel verständlich wird, dessen Spielerumriß nun bewegt werden soll. Hierzu Beispiel 11 anlaufen lassen!

Man sieht, daß der Speicherbereich für den Umriß gelöscht wird und das dann der Umriß erscheint, der allerdings nicht wie ein Pac Man aussieht. Jetzt bewegt man den Spielhebel. Sobald nun Bewegung eintritt, macht sich die Figur schon besser. Zum Öffnen und Schließen des Maul hätte man auch 3 oder 4 Elemente zeichnen können. Doch lassen sich 2 besser erklären!

Wenn man ein weiteres Element zufügt, dessen Maul nur leicht geöffnet ist, wird der Ablauf flüssiger.

Um die beiden Figuren in Bewegung zu sehen, braucht man nur ein paar Anstöße am Spielhebel zu geben. Man sieht dann zwar 2 Spieler, doch immer nur einen gleichzeitig. Das ist wie folgt programmiert:

Auf Zeile 50 (in der Programmlistung des Beispiels 11) beträgt die Speicherverschiebung nicht 16 sondern 32. Diese Größe braucht man für Graphikgang 7. Auf Zeile 60 folgen POKES in das Horizontale Register und in den Umfang von Spieler 2 ! Man muß die Spieler nähmlich nicht in einer bestimmten Reihenfolge aktivieren.

Die Zeilen 120 - 150 bringen lediglich die Daten für eine 2. Erscheinung von Spieler 2 in den entsprechenden Speicherbereich. Auf Zeile 225 dienen POKES dazu, Spieler 0 nach X und Spieler 2 nach 0 zu bringen , also aus dem Bild hinaus. Schließlich wird, bei Zeile 226, der Spieler 0 aus dem Bild geschoben und Spieler 2 nach X gerückt. Auf Zeile 225 ist eine Verzögerung eingelegt, durch die man das Umschalten der Figuren genauer verfolgen kann.

Um einen weiteren Spieler zu bauen, tut man folgendes:

1. Man gibt mit POKE die Daten für den neuen Spielerumriß in den richtigen Speicherbereich,

2. man gibt die gewünschte Größe in das Größenregister (muß nicht sein, weil automatisch 0 dort hin gerückt wird).

3. Man setzt die horizontale Position (X) in das Register für die Horizontale. Vergißt man das, rutscht die Figur automatisch auf Position 0, also links außerhalb des Bildes.

Zum Bewegen der Figur wechselt man einfach zwischen 2 oder mehr Umrissen am selben Platz auf dem Monitor hin und her. Diese ähnlichen Figuren werden dann wie eine einzige aussehen, die sich bewegt.

Wenn man an das Roboterbeispiel zurück denkt, bei dem gefragt wurde, welches von den beiden Figuren der Spieler ist, so wird klar, wie praktisch es ist, diese Spielertechnik einzusetzen. Mit Hilfe von BASIC würde das Setzen und Löschen der Figuren nur sehr langsam von statten gehen.

PRIORITÄT UND KOLLISION

Beispiel 12:

Beispiel 12 nimmt den Anfang eines Spieles wieder auf, daß schon in 11 enthalten war. Es schließt zusätzlich die Kollision ein, und auch sogenannte Energiepillen, die von einem Tier gefressen werden. Hinzu kommt der Effekt, daß die Figuren bei Überschreiten der "Wände" des Tunnels so erscheinen, als stünden Sie VOR der Wand. Am Ende des Tunnels erscheint die Figur jeweils innerhalb oder hinter den Wänden. Diese Wirkungen werden durch bestimmte Setzungen von Prioritäten erzielt, sodaß sich einige Farben anders verhalten als andere.

Die meisten Teile des Programms unterscheiden sich nicht von früheren Programmstrukturen. Es gilt überhaupt, daß man schon durch geringfügige Änderungen des Programms ganz neue Effekte erzielen kann.

Zeile 10 setzt alle Positionsregister für die Horizontale auf 0. Das würde für unser Programm 12 eigentlich nicht gebraucht, stellt aber sicher, daß alle früheren Spieler vom Bild verschwinden. Denn das ist mit dem Laden eines neuen Programms keineswegs garantiert.

Die Zeilen 47-57 setzen 3 verschiedene Teile auf den Hintergrund.

Das 1. sind die sogenannten Energiepillen, also kleine Flecken, die der PAC MAN auffrißt. Sie werden in Farbe 1 geplottet (Register 708). Dann in den Zeilen 50 und 52, werden die rechteckigen Felder, die sogenannten PAC MAN - Tunnels gezeichnet, undzwar in Farbe 2.

Die Zeilen 55-57 zeichnen die Hindernisse, die der Figur den Weg versperren, natürlich in Farbe 3.

Die Tatsache, daß die verschiedenen Elemente in verschiedenen Farben erscheinen, macht das Spiel kritisch. Der ATARI hat für diese Situation sogenannte Kollisionsregister, die man darauf hin abfragen kann, ob ein Element (z.B. ein Spieler) an ein anderes gestoßen ist (z.B. an eine Wand in Farbe 1).

Weiter, auf Zeile 205, lernen wir einen neuen Trick kennen:

Es gibt eine Speicherstelle mit der Bezeichnung HITCLR, bei 53278. Wenn in dieses Register eine 1 geladen wird, schaltet es die anderen Kollisionregister auf 0. Unterläßt man diese Rückschaltung, so bleiben die Kollisionregister auf dem jeweils durch Kollision gesetzten Wert stehen. Kollisionsregister und andere Register sind in der Speicher Listung zu haben. (ACC Berlin)

Es sind nicht wenige, ihr Gebrauch bietet jedoch keine Probleme. Man beginnt einfach bei dem Register, daß zu dem gerade ins Auge gefaßten Objekt gehört. In diesem Fall wollen wir wissen, ob die Figur des Spielers 0 mit dem Spielfeld zusammengestoßen ist:

Das bedeutet, wir müssen den Wert in 53252 abfragen, was auf Zeile 242 auch geschieht.

Wenn das Register nicht auf 0 steht, ist ein Zusammenstoß vorgekommen und wir verzweigen auf die Subroutine von Zeile 1000.

Bei 1000-1005 wird abgefragt, welchen Inhhalt das Kollisionsregister hat. Das es nicht auf 0 ist, ist schon bekannt,- wenn es auf 2 steht, bedeutet dies:

Der Spieler "traf" , also überlappte Spielfeld 2, nähmlich die Tunnelwände. Wir programmieren dann einen Kollisionsturm und gehen auf Zeile 1030. Das ist UNUMGÄNGLICH, damit die Kollisionsregister wieder auf 0 gesetzt werden.

Steht in unserem Register eine 1, so ist der genannte Spieler auf Spielfeld 1 gestoßen, also hier eine sogenannte Energiepille.

Daraufhin wird der Text "YUMMY" ausgedruckt und ein passender Begleitton gesetzt. Dann geben wir noch POKE auf HITCLR und kehren zurück.

Hier eine kurze Wiederholung:
Man sucht das Kollisionsregister auf, das die beiden Elemente betrifft, über die man eine Auskunft wünscht, entweder eine Spielernummer X oder eine Geschoßnummer X rechts und jeweils entweder ein Spieler, ein Geschoß bzw. ein Spielfeld Nummer Y links. Man fragt durch PEEK den Inhalt des Kollisionsregisters ab. Es kann 0,1,2 oder 4 enthalten. Bei 0 fand keine Kollision statt. Bei 1 ist man mit Spielfeld 1, Geschoß 1 oder Spieler 1 zusammen gestoßen, jenachdem wofür das Kollisionsregister zuständig ist. 2 bedeutet dementsprechend, daß man mit Spielfeld, Geschoß oder Spieler 2 kollidiert hat. 4 steht schließlich für Spielfeld, Spieler bzw. Geschoß Nr. 3. Auf Grund dieser Rückmeldung aus dem Kollisionsregister kann man dann das Programm nach Wunsch verzweigen.

Hier noch der Hinweis auf einen kleinen Trick:

Zeile 245 ermöglicht der Fifur einen Bildumlauf in der X-Ebene (WRap Around). Hier wird einfach abgefragt, ob der X-Wert über 200 liegt, einen bestimmten Mindestabstand vom rechten Rand hat.

Setzt man X = 40, so erscheint die Spielfigur plötzlich von links. Eine Einzelheit des Beispiels 12 ist nicht aus der Codierung ersichtlich:

Nähmlich die Priorität!
Dieser Begriff steht für die Frage "Was erscheint VOR bzw. HINTER etwas anderem ?"

Unser Beispiel hatte keine Prioritätsabfrage, sondern ließ der automatischen Regelung freien Lauf.

Um dies Kontrollregister einzusetzen, addiert man einfach alle Möglichkeiten auf und setzt die SUMME in 623. Man kann jedoch nur eine Gruppe von Prioritäten wählen, also nur z.B. 8,4,2 oder 1.

Der fünfte Spieler:

Man erinnert sich, daß in Beispiel 5 durch Druck auf SELECT, alle 4 Geschosse sich zu einem Spieler zusammenfügten. Das wurde so programmiert:

Man setzte 623 auf den Wert 17 sowohl für die gewünschten Prioritäten wie für Farbe des 5. Spielers (Teile 1000-1050). Dann schob man einfach die X-Positionen nebeneinander. Geschosse sind nur 2 Pixels breit; man kann sie gemeinsam genau wie einen normalen Spieler benutzen, nur daß das Ausrechnen der Position hier jeweils etwas schwieriger ist.

Spieler von 16 Pixels Breite, Beispiel 13.

Bisher standen uns zwar 128 bzw. 256 Pixels für die Höhe der Elemente zur Verfügung, jedoch nur 8 Pixels in der Breite! Um breitere Figuren zu bekommen, kann man ohne weiteres 2 Spieler so nebeneinander plazieren, daß sie optisch verschmelzen. Durch eine einfache Formel kann sie stehts auf 8 Pixel horizontal Abstand halten. Beispiel 13 zeigt ein solches Verfahren für ATARI.

Eine Überraschung (für 32K), Beispiel 14:

Unser letztes Programm ist ein Ausgabeprogramm für Spielerfiguren.
Es wird wie folgt eingesetzt:

1. Am Anfang des Ablaufs wird abgefragt, ob alte Daten übernommen oder neue geladen werden sollen. Wähle nun.

2. Als nächstes sollte die Nachricht erscheinen: "Spielernummer für Ausgabe." Wähle jetzt eine Ziffer von 0-4. Der 4. Spieler wird aus den Geschossen gebildet.

3. Ein blinkender Läufer erscheint jetzt im Ausgaberahmen links. Drücke D um zu zeichnen und führe danach den Läufer mit den 4 Steuertasten wie gewünscht hin und her (CTRL wird nicht gebraucht).

4. Um eine Zeichnung zu beenden, drücke E (ERASE). Bewege den Läufer an den Anfangspunkt der nächsten Zeichnung.

5. Man kann auch an einer Zeichnung neu anfangen: Hierzu drückt man START und korrigiert oder ergänzt nach Belieben mit dem Läufer.

6. Drücke OPTION zum ändern der Nummer des ausgegebenen Spielers.

7. Man kann den gerade entstehenden Spieler auf dem Spielhebel auch bewegen (Spielhebel 1).

8. Die Größe des ausgedruckten Spielers kann ebenfalls geändert werden (außer bei Spieler 4): Dazu drückt man SELECT, dann 0, 1 oder 3 für normal, doppelte bzw. vierfache Größe.

9. Drücke S, um die aufgebauten Figuren auf Band oder Platte zu bringen.

10. Drücke C, um irgendwelche aufgezeichneten Figuren in den Speicher zu laden.

Zu beachten:
Wenn man einen Satz Spieler in die Maschine läd, sieht man sie zunächst nicht, sondern erst nach drücken von OPTION und Einzelabruf.

Dieses Ausgabeprogramm macht ein schnelles entwerfen von Spielern für jeden Zweck leicht. Die Codierung für unser Verfahren enthält übrigens auch einige lehrreiche Beispiele für die Speicherung von Spielerdaten mit Hilfe von STRINGS, interessant für Fortgeschrittene.

Einige ergänzende Hinweise sollen die Lektion abschließen.

Zuerst gehen wir zum Beispiel 2 zurück, an dem wir noch einmal die Struktuierung des Spiels demonstrieren wollen, soweit diese in die VCodierung eingeht.

Die Zeilen 40-60 setzen die Farben, Zeilen 70 verweist auf eine Subroutine, die die Spielfelder zeichnet. Das Spielfeld sollte man stets am Schluß des Programms behandeln, damit die Hauptschleife schneller abläuft. In diesem Fall wird für die Spielfelderstellung lediglich eine größere Menge von Daten geplottet.

Zeile 80 sagt aus, daß das Programm nicht gestoppt ist, sondern sich nur neu ordnet. So etwas macht sich immer gut.

Zeile 110 ist kompliziert. Es handelt sich um eine weitere Routine in Maschinensprache, sogar weit nützlicher als die zuerst gezeigte.

Es werden hier 4 Anfangsregister festgelegt, undzwar in einer geschützten Zone des Speichers, nähmlich zwischen PMBASE und PMBASE+768, wo die Geschoßdaten beginnen. Da Zeile 20 diesen ganzen Bereich löscht, weiß man, daß er frei ist und gesichert.

Nun, auf den Zeilen 130-310 werden die Daten für 2 PAC MAN-Umrisse und 2 für das SQUIGILY MONSTER eingeladen.

Ihre Anfangspunkte sind leicht merkbar und können später gut gefunden werden.

Zeile 330 gibt den Maschinencode wieder, der in einem String gespeichert ist. Man kann diese Zeile für andere Programme abkopieren, zusammen mit dem Dimensionsstatement auf Zeile 50 und dem zugehörigen USR-Abruf.

Die Zeilen 380-440 erbringen die normale Rechenarbeit, die man zum Verfolgen der X- und Y- Werte zweier Spielhebel braucht.

Auf den Zeilen 450-480 wird in Maschinensprache gearbeitet. Es handelt sich um eine Routine zum Verschieben im Speicher. Sie nimmt die genannten Adressen auf und bewegt deren Inhalte an jede gewünschte Stelle. Man kann sie zur totalen Verschiebung eines vollständigen Bildinhaltes anwenden, wenn man will. In unserem Beispiel dient die Routine zum Bewegen der Figur, die in den Bereich zwischen den Daten der beiden Spieler abgelegt ist. Folgende Anweisungen werden in den einzelnen Zeilen gegeben:

450: "Die PAC MAN - Daten sind von Stelle "Figur 2" auf den Bereich für Spieler 0 gerückt, wobei 20 Bytes gebraucht werden." Diese Figur erscheint jetzt im Bild.

460: "Dasselbe wie vorher, doch die SQUIGILY-Daten in den Bereich von Spieler 1"(auch dieser wird jetzt gezeigt).

470: "Bewege 2. PAC MAN - Gestalt in den Bereich von Spieler 0"(neuer PAC MAN wird gezeigt).

480: "NBewege 2. Squigily-Gestalt in den Bereich für Spieler 1 !"(neue Gestalt wird gezeigt).

Alle oben genannten Bewegungen schließen auch das Auf- und Abrücken ein, zugleich mit der Addition auf der Y-Koordinate, die jeweils im Register für die Datenbewegungen eingegeben wird.

Diese Routine bewegt alle 5 Spieler und/oder Geschosse, alle sehr schnell. Die frühere Routine in Maschinensprache bewegte nur Spieler 0. Die hier benuzte Routine wird von uns oft eingesetzt.

Die Zeilen 490-570 registrieren alle Kollisionen, an denen wir interessiert sind. Jede dieser Stellen enthält eine Verzweigung zu einer Subroutine bzw. enthält bedingte Änderungen.

Die Trefferzählung und die Töne sind simple zu programmieren:

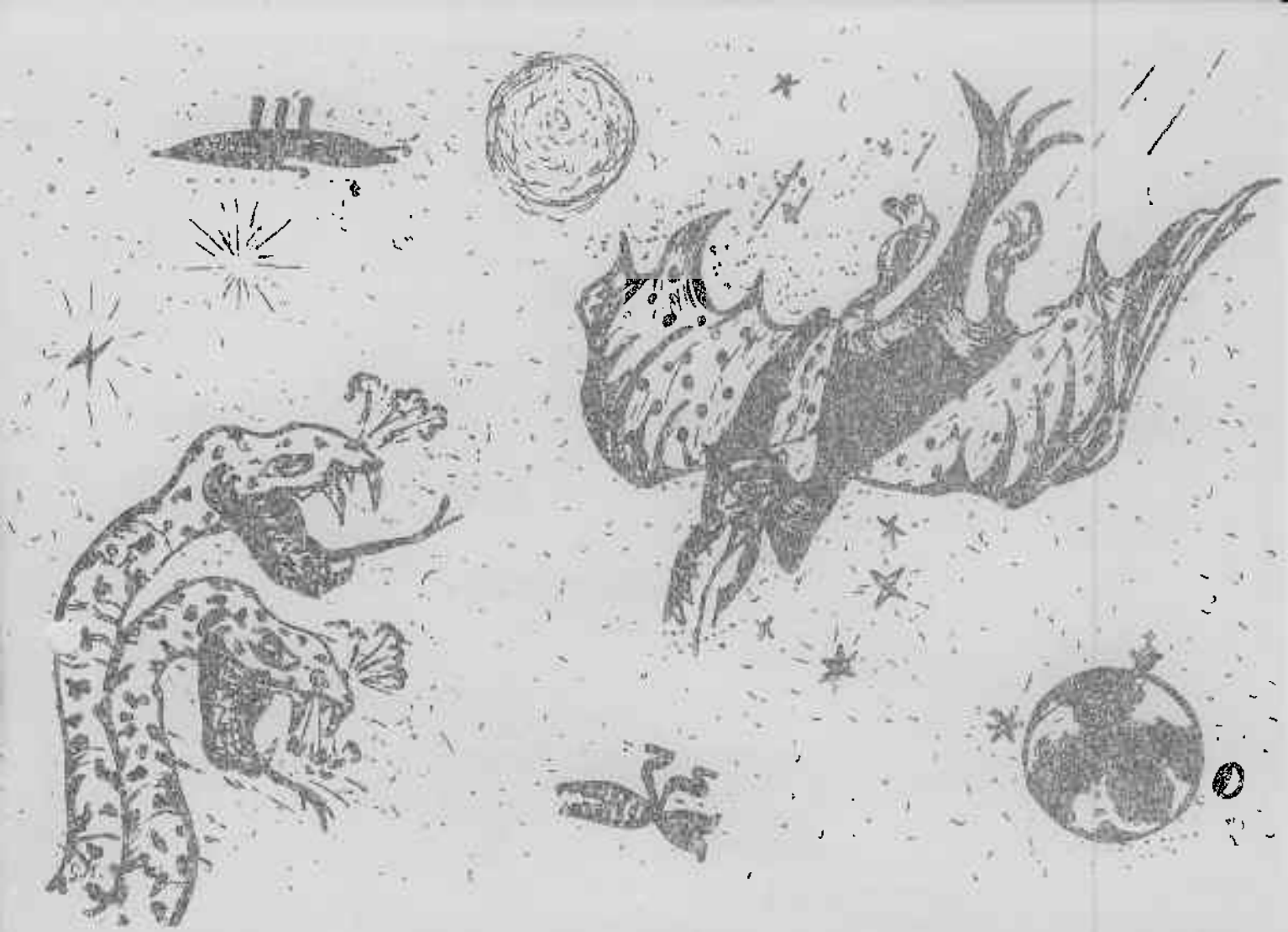
Man setzt einfach einen Toncode, zählt einen Treffer, wechselt eine Farbe. Diese Elemente eines Spiels hängen von der Phantasie des Programmierers ab, er setzt ein, was er für gut hält.

SCHLUßBEMERKUNG :

Wir haben den Gebrauch der Geschosse nicht sehr betont. Es wird klar geworden sein, daß man sie genau wie Spieler einsetzen kann. Ein gutes Beispiel für ihre Bewegung bietet Beispiel 1 in den Zeilen 640-710. Die hauptsächliche Einschränkung bei Geschossen besteht darin, daß sie nur 2 Pixels breit sind und daß die Geschoßdatenbereiche in ein und demselben Streifen stehen. Das bedeutet, daß man NULEN in diejenigen Geschoßpositionen poken muß, die nicht auf das Bild kommen sollen bzw. das man diese Geschosse mit Hilfe der Register für horizontale Positionen aus dem Bild schieben muß.

In den Programmen sind übrigens einige kleine Besonderheiten in Hinsicht auf Farben, Größen und Wegschieben von solchen Figuren, die gerade nicht gebraucht werden, versteckt.

Sie alle sollten sorgfältig studiert und unter Umständen verbessert werden.

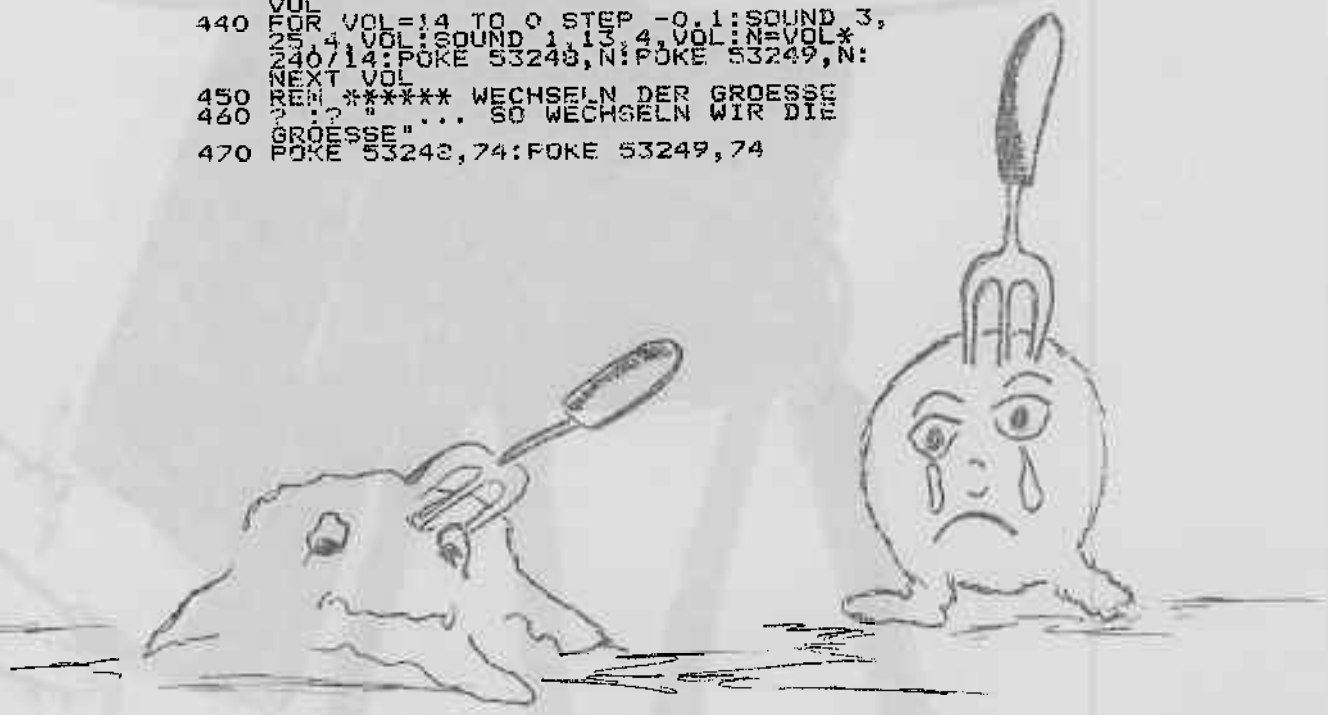


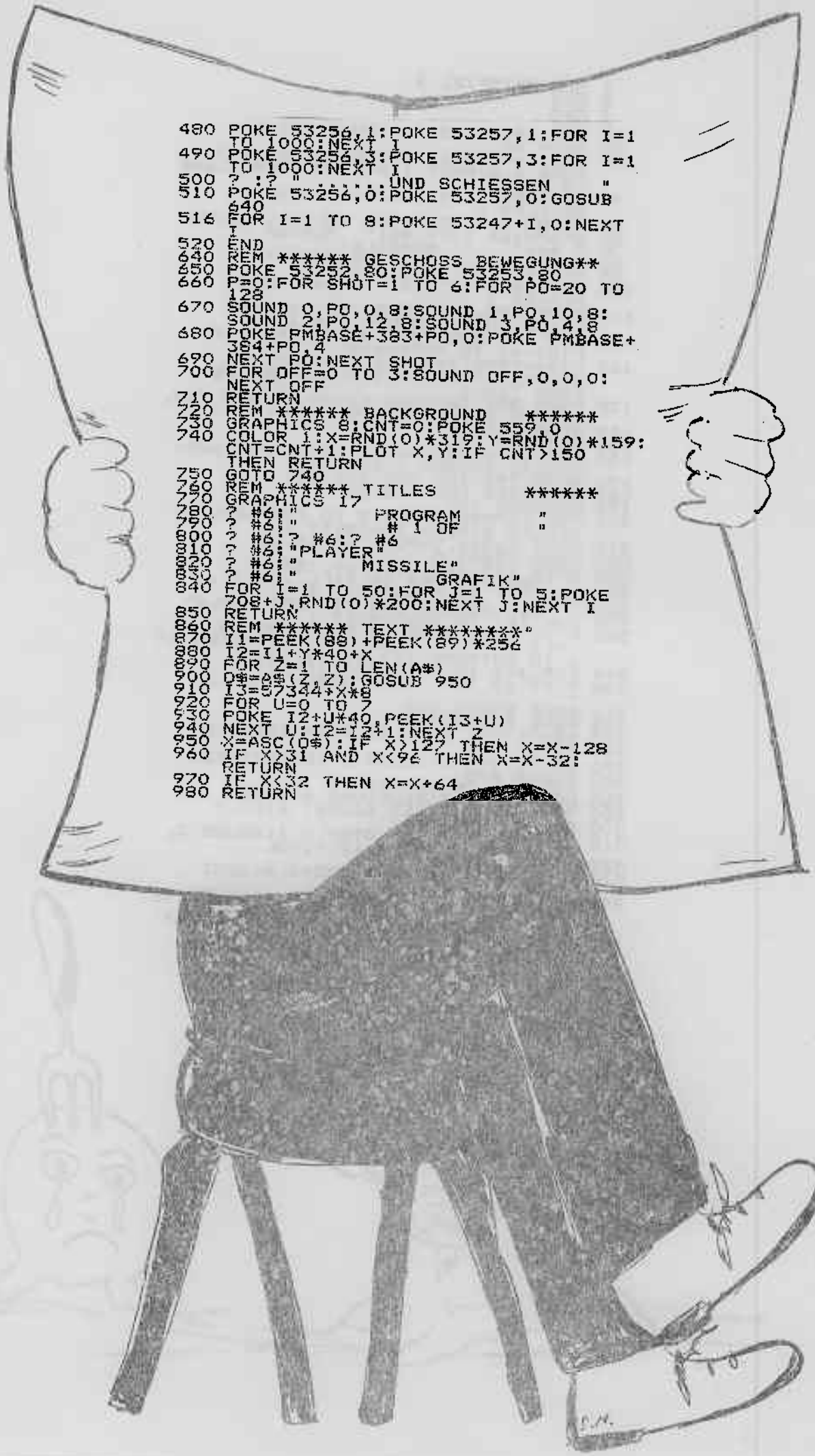
[illegible]

[illegible]

[illegible]

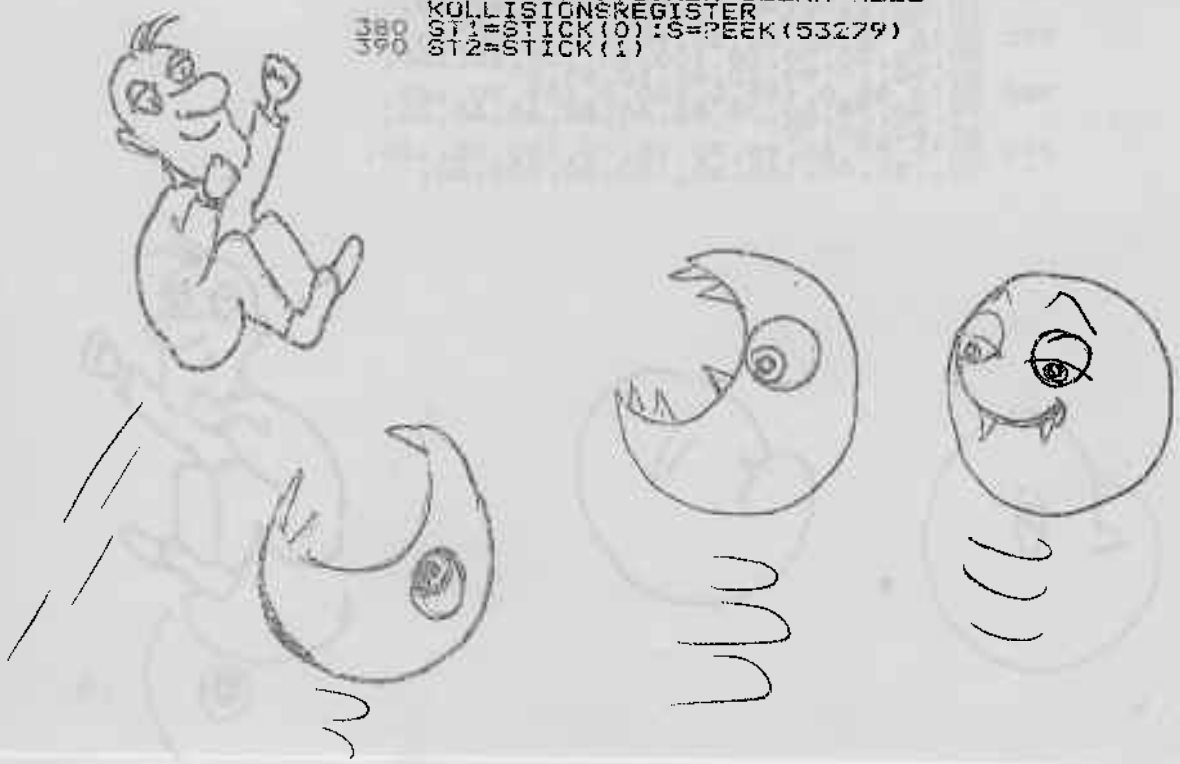
```
  1 REM BEISPIEL 1
  2 REM
  3 REM
  4 REM
 10 DIM A$(15),O$(1):POKE 623,8
 20 GOTO 40
 30 FOR I=PMBASE+384 TO PMBASE+1024:
    POKE I,0:NEXT I:RETURN :REM CLEAR
    PM AREA
 40 K=20:REM INITIALISIEREN VON
    SPIELER 0
 50 SOUND 3,13,8,2:REM WELTALL SOUND
 60 GRAPHICS 17:? #6;"  PLAYER "
 70 ? #6;"       MISSILE"
 80 ? #6;"              GRAPHICS"
 90 FOR I=1 TO 2000:NEXT I
100 Z=90:REM INITIAL Y POS. FUER
    PLAYER 1
110 POKE 704,149:POKE 705,249:REM
    FARBE FUER PLAYERS 0 & 1
115 REM 16K ANWENDER WECHSELN     P=
    P106-40 TO P=P106-8
120 P106=PEEK(106):P=P106-40:POKE
    54279,P:PMBASE=256*P:GOSUB 730:
    POKE 710,0
130 POKE 559,46:POKE 53277,3:POKE 752,
    1:POKE 53248,0:POKE 53249,0
140 GOSUB 30
150 RESTORE 120
160 FOR I=PMBASE+640+K TO PMBASE+644+
    K:READ B:POKE I,B:NEXT I
170 DATA 153,189,255,189,153
180 REM DATA FOR TIE SHAPE
190 RESTORE 170
200 FOR I=PMBASE+512+Z TO PMBASE+516+
    Z:READ A:POKE I,A:NEXT I
210 POKE 53256,0:POKE 53260,18
220 POKE 53257,0
230 DATA 153,189,255,189,153
240 ? "DAS IST DAS WELTALL ........."
250 FOR I=1 TO 3000:NEXT I
260 ? "WIR NENNEN DAS EIN SPIELFELD"
270 FOR I=1 TO 3000:NEXT I
280 ? :? "     SPIELFELD UMRANDUNG
    ":? "          UND   FARBEEN":FOR I=
    1 TO 20:POKE 712,RND(0)*255
    :NEXT I
290 FOR J=1 TO 100:NEXT J:NEXT I
300 ? :? :? "       WIR HABEN SPIELER
    "
310 POKE 53248,100
320 POKE 53249,100
330 A$="<==== SPIELER 1":X=15:Y=10
340 GOSUB 870
350 A$="<==== SPIELER 2":X=15:Y=150
360 GOSUB 870
370 FOR I=1 TO 30:NEXT I
380 REM ******* BEWEGUNG ******
390 ? :? "     ....ES BEWEGT SICH."
400 SOUND 2,0,8,2
410 FOR VOL=1 TO 15 STEP 0.1:SOUND 3,
    25,4,VOL:SOUND 1,13,4,VOL
420 N=VOL*240/15
430 POKE 53249,N:POKE 53248,N:NEXT
    VOL
440 FOR VOL=14 TO 0 STEP -0.1:SOUND 3,
    25,4,VOL:SOUND 1,13,4,VOL:N=VOL*
    240/14:POKE 53248,N:POKE 53249,N:
    NEXT VOL
450 REM ****** WECHSELN DER GROESSE
460 ? :? "... SO WECHSELN WIR DIE
    GROESSE"
470 POKE 53248,74:POKE 53249,74
```



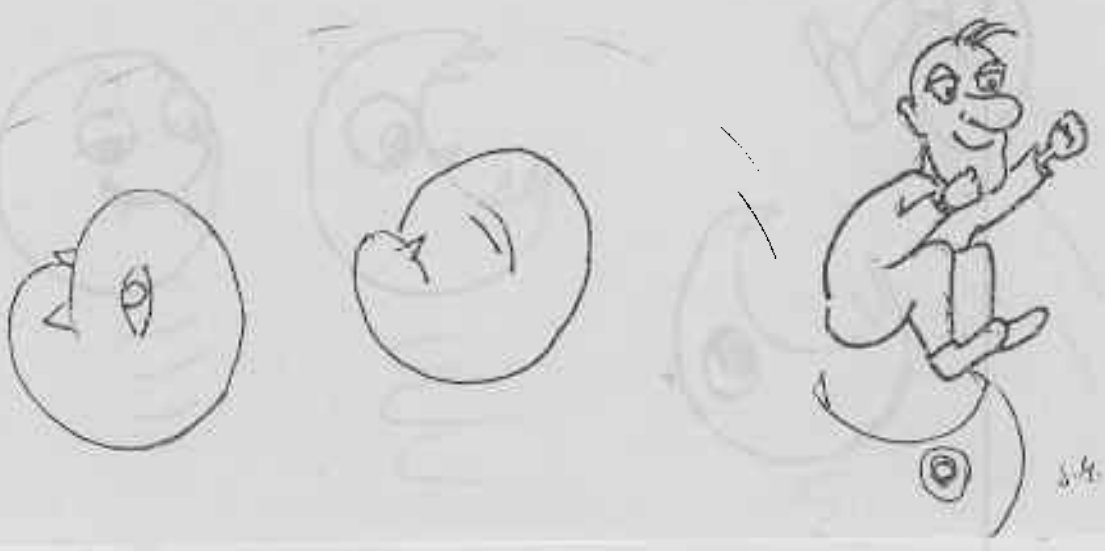


```
480 POKE 53256,1:POKE 53257,1:FOR I=1
    TO 1000:NEXT I
490 POKE 53256,3:POKE 53257,3:FOR I=1
    TO 1000:NEXT I
500 ? :? "   ...UND SCHIESSEN    "
510 POKE 53256,0:POKE 53257,0:GOSUB
    640
516 FOR I=1 TO 8:POKE 53247+I,0:NEXT
    I
520 END
640 REM ****** GESCHOSS BEWEGUNG**
650 POKE 53252,80:POKE 53253,80
660 P=0:FOR SHOT=1 TO 6:FOR PO=20 TO
    128
670 SOUND 0,PO,0,8:SOUND 1,PO,10,8:
    SOUND 2,PO,12,8:SOUND 3,PO,4,8
680 POKE PMBASE+383+PO,0:POKE PMBASE+
    384+PO,4
690 NEXT PO:NEXT SHOT
700 FOR OFF=0 TO 3:SOUND OFF,0,0,0:
    NEXT OFF
710 RETURN
720 REM ****** BACKGROUND    ******
730 GRAPHICS 8:CNT=0:POKE 559,0
740 COLOR 1:X=RND(0)*319:Y=RND(0)*159:
    CNT=CNT+1:PLOT X,Y:IF CNT>150
    THEN RETURN
750 GOTO 740
760 REM ****** TITLES         ******
770 GRAPHICS 17
780 ? #6;"          PROGRAM         "
790 ? #6;"              # 1  OF      "
800 ? #6;? #6;? #6
810 ? #6;"PLAYER"
820 ? #6;"      MISSILE"
830 ? #6;"              GRAFIK"
840 FOR I=1 TO 50:FOR J=1 TO 5:POKE
    708+J,RND(0)*200:NEXT J:NEXT I
850 RETURN
860 REM ****** TEXT ********"
870 I1=PEEK(88)+PEEK(89)*256
880 I2=I1+Y*40+X
890 FOR Z=1 TO LEN(A$)
900 O$=A$(Z,Z):GOSUB 950
910 I3=57344+X*8
920 FOR U=0 TO 7
930 POKE I2+U*40,PEEK(I3+U)
940 NEXT U:I2=I2+1:NEXT Z
950 X=ASC(O$):IF X>127 THEN X=X-128
960 IF X>31 AND X<96 THEN X=X-32:
    RETURN
970 IF X<32 THEN X=X+64
980 RETURN
```

```
  1 REM BEISPIEL 2
  2 REM
  3 REM
  4 REM
 10 GOTO 580
 20 FOR I=PMBASE TO PMBASE+2048:POKE
    I,0:NEXT I:RETURN
 30 REM *****PLAYER ERSTELLEN*****
 40 GRAPHICS 7:POKE 710,121:POKE 712,
    0:POKE 704,41:POKE 705,41:POKE
    559,62
 50 DIM A$(100):POKE 623,1
 60 POKE 708,103:POKE 752,1:POKE 709,
    148
 70 GOSUB 690:REM HINTERGRUND MALEN
 80 ?" *** AUGENBLICK BITTE ..... **
    *"
 90 A=PEEK(106)-32:POKE 54279,A:
    PMBASE=A*256
100 POKE 53277,3:POKE 53256,0:POKE
    53250,150
110 GOSUB 20:SHAPE1=PMBASE:SHAPE2=
    PMBASE+50:SHAPE3=PMBASE+100:
    SHAPE4=PMBASE+150
120 RESTORE 160:Y1=172:Y2=172:X1=60:
    X2=190
130 FOR I=PMBASE TO PMBASE+20:READ B
140 POKE I,B:NEXT I
150 REM ***** DATA FUER PACMAN1 *****
160 DATA 0,0,0,28,62,119,127,126,124,
    120,124,126,127,126,62,28,0,0,0,0,
    0
170 RESTORE 210
180 FOR I=PMBASE+50 TO PMBASE+70:READ
    B
190 POKE I,B:NEXT I
200 REM ***** DATA FUER PACMAN2 ******

210 DATA 0,0,0,28,62,118,126,126,126,
    126,127,125,126,62,28,0,0,0,0,0
220 POKE 559,62
230 RESTORE 260:FOR I=PMBASE+100 TO
    PMBASE+120:READ B
240 POKE I,B:NEXT I
250 REM ***** DATA FUER MONSTER1 *****

260 DATA 0,0,0,0,124,214,214,254,124,
    68,68,204,0,0,0,0,0,0,0,0,0
270 RESTORE 310
280 FOR I=PMBASE+150 TO PMBASE+170:
    READ B
290 POKE I,B:NEXT I
300 REM ***** DATA FUER MONSTER2 *****
    ***
310 DATA 0,0,0,0,124,254,214,214,124,
    68,68,102,0,0,0,0,0,0,0,0,0
320 SCOREMON=3:SCOREPAC=3
330 A$="h[I](C)[P]0h[(E)a]h[(E)(.)]h[
    (E)c]h[(E)b]h[(E)e]h[(E)d]](,)[
    (E)f(E)q(1(.)(Q)b](X)()](A)e[(.)
    (E)(.))](,)e[a(E)a](X)[)](A)e[b
    (E)b)](,)e[c(E)c](X)[)](A)e[f
    (E)f)](,)e[g(E)g](X)[%dE+PN%eEgPH]
    (.)"
340 ?:? :? :? "STICK1
    STICK2":FOR I=1 TO 300:NEXT I
350 REM *******HAUPT PROGRAMM********
360 GOSUB 1030
370 POKE 53278,1:REM CLEAR ALLE
    KOLLISIONSREGISTER
380 ST1=STICK(0):S=PEEK(53279)
390 ST2=STICK(1)
```

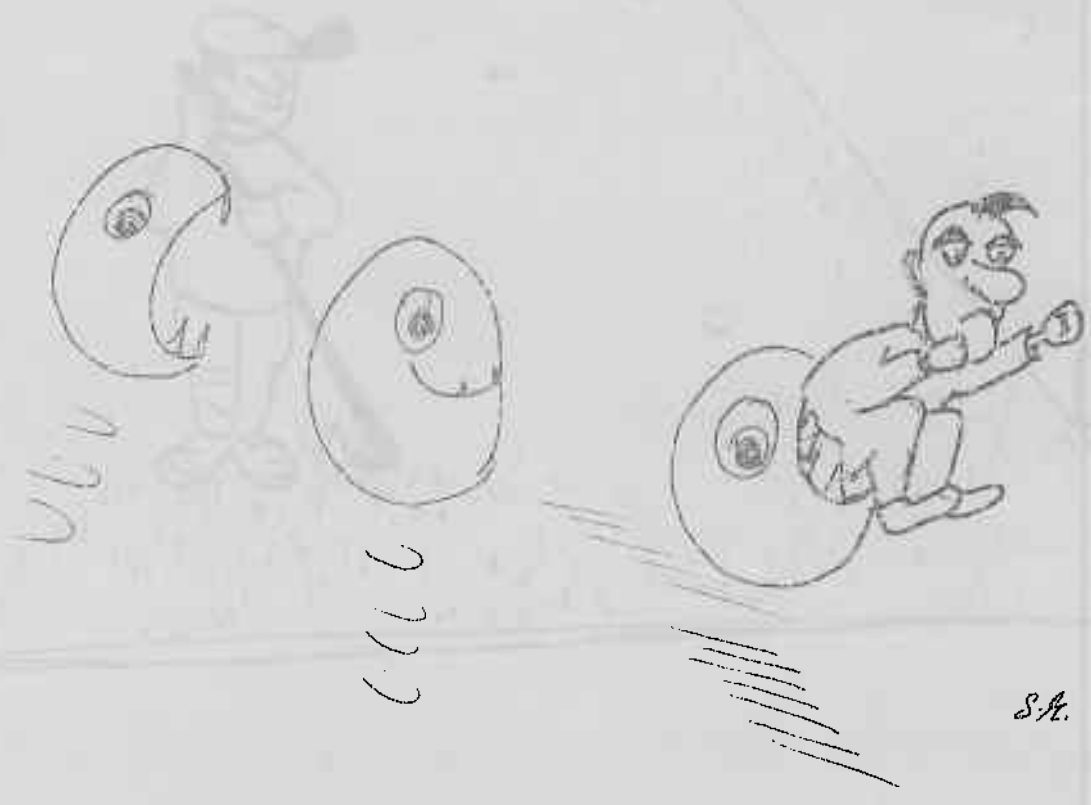


```
400 IF S=3 THEN POKE 764,12:RUN "D:
    EX5.3"
410 Y1=Y1+(ST1=13)*2-(ST1=14)*2
420 Y2=Y2+(ST2=13)*2-(ST2=14)*2
430 X1=X1+(ST1=7)-(ST1=11):POKE 53248,
    X1
440 X2=X2+(ST2=7)-(ST2=11):POKE 53249,
    X2
450 D=USR(ADR(A$),SHAPE2,PMBASE+1024+
    Y1,20)
460 D=USR(ADR(A$),SHAPE4,PMBASE+1280+
    Y2,20)
470 D=USR(ADR(A$),SHAPE1,PMBASE+1024+
    Y1,20)
480 D=USR(ADR(A$),SHAPE3,PMBASE+1280+
    Y2,20)
490 IF PEEK(53252)=2 THEN POKE 704,
    200:POKE 705,41:SOUND 0,200,10,8:
    FOR I=1 TO 30:NEXT I:SOUND 0,0,0,
    0
500 IF PEEK(53252)=1 THEN X1=X1-(ST1=
    7)+(ST1=11):POKE 53248,X1:Y1=Y1-
    (ST1=13)*3+(ST1=14)*3
510 IF PEEK(53253)=2 THEN POKE 705,
    200:POKE 704,41:SOUND 0,200,10,8:
    FOR I=1 TO 30:NEXT I:SOUND 0,0,0,
    0
520 IF PEEK(53253)=1 THEN X2=X2-(ST2=
    7)+(ST2=11):POKE 53249,X2:Y2=Y2-
    (ST2=13)*3+(ST2=14)*3
530 IF PEEK(53260)<>0 THEN GOSUB 920:
    GOTO 570
540 IF PEEK(53253)=4 THEN GOSUB 890
550 IF RND(0)*100>99 THEN COLOR 3:
    PLOT 35,55:PLOT 135,65
560 IF PEEK(53252)=4 THEN GOSUB 860
570 POKE 53278,0:GOTO 380
580 GRAPHICS 17
590 ? #6;"WIR SPIELEN JETZT    "
600 ? #6;"MIT EINEM KOMPLETTEM"
610 ? #6;"SPIEL                "
620 ? #6;"WIR WOLLEN JETZT DAS"
630 ? #6;"ERLERNEN WIE ES GE-  "
640 ? #6;"MACHT WIRD.          "
650 ? #6;"BITTE          PACMAN[?]"
660 FOR I=1 TO 3000:NEXT I:GOTO 30
670 REM **** HINTERGRUND**********
    *******
680 RESTORE 740:COLOR 1
690 READ X1,Y1,X2,Y2:IF X1=999 THEN
    820
700 PLOT X1,Y1:DRAWTO X2,Y2
710 GOTO 710
720 DATA 20,20,20,0,20,0,0,0,0,0,0,79,
    0,79,70,79,70,79,70,70,0,50,10,50,
    0,40,10,40,10,10,10,30,10,30,20,
    30
730 DATA 20,30,20,40,20,40,40,40,40,
    40,40,10,20,50,20,60,20,60,10,60,
    10,60,10,70,10,70,50,70,50,70,50,
    40
740 DATA 30,30,30,0,30,0,70,0,70,0,70,
    10,50,0,50,30,50,40,60,40,60,10,
    60,50,30,50,40,50,30,60,40,60
750 DATA 70,30,90,30,90,30,90,50,70,
    50,70,30,70,20,100,20,100,20,100,
    60,90,10,90,20,80,10,80,0
760 DATA 80,0,159,0,159,0,159,70,159,
    79,80,79,80,79,80,60,80,60,60,60,
    60,60,60,70
770 DATA 90,60,90,70,90,70,150,70,150,
    70,150,40,140,60,130,60,130,60,
```

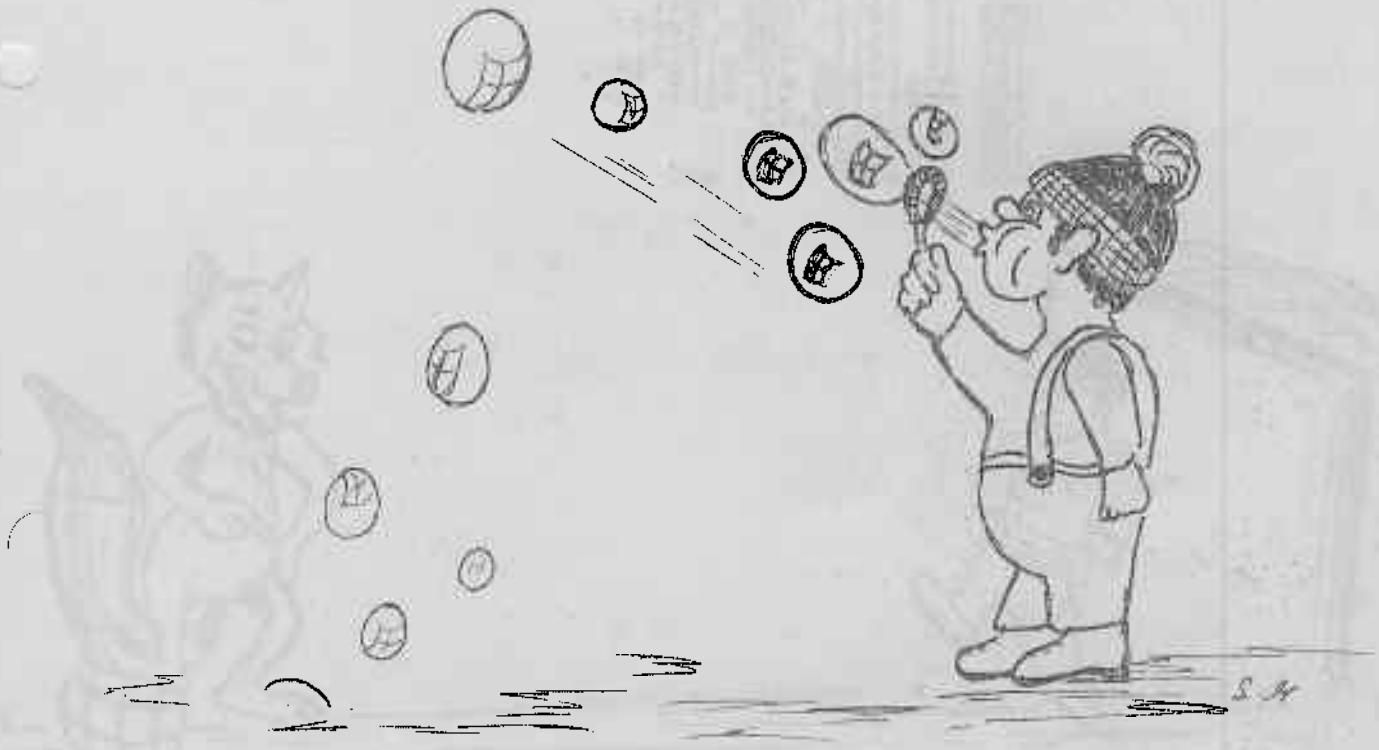


```
  1 REM BEISPIEL 3
  2 REM
  3 REM
  4 REM
  5 FOR I=1 TO 8:POKE 53247+I,0:NEXT
    I
 10 GRAPHICS 17:POKE 752,1:POKE 712,
    128
 30 ? #6;"     LEICHTER START     "
 40 ? #6;"                        "
 50 ? #6;"        MIT DEM         "
 60 ? #6;"                        "
 70 ? #6;"      SPIELFELD         "
 80 FOR I=1 TO 100:POKE 709,I:FOR J=1
    TO 30:NEXT J:NEXT I
 90 REM ******** EIGENE ZEICHNUNG***
100 GRAPHICS 5:? "MIT JOYSTICK 0
    ZEICHNE SPIELFELD  0-3":? "
    PRESS 0,1,2, OR 3"
110 POKE 708,245:POKE 712,130:POKE
    709,67:POKE 710,38
120 ST=STICK(0)
130 Y=Y+(ST=13)+(ST=9)+(ST=5)-(ST=10)-
    (ST=14)-(ST=6)
140 X=X+(ST=6)+(ST=7)+(ST=5)-(ST=10)-
    (ST=11)-(ST=9)
150 COLOR PFN
160 IF X<0 THEN X=0
170 IF X>79 THEN X=79
180 IF Y<1 THEN Y=1
190 IF Y>39 THEN Y=39
200 IF PEEK(764)=50 THEN PFN=0
210 IF PEEK(764)=31 THEN PFN=1
220 IF PEEK(764)=30 THEN PFN=2
230 IF PEEK(764)=26 THEN PFN=3
240 POKE 656,2:? "BENUTZE SPIELFELD
    #";PFN:POKE 752,1
250 PLOT X,Y:GOTO 120
```

```
      130,70,130,40,110,40,110,40,110,
      70
  800 DATA 100,30,130,30,110,10,110,20,
      110,20,130,20,140,20,140,50,140,
      50,120,50,120,50,120,60
  810 DATA 150,10,150,30,140,0,140,10,
      130,10,130,20,120,0,120,10,100,0,
      100,10,999,99,9,9
  820 REM ******* FARBE GEBEN *******
  830 COLOR 2:FOR I=1 TO 20:PLOT 70,30+
      I:DRAWTO 90,30+I:NEXT I
  840 COLOR 3
  850 RETURN
  860 REM ***** PUNKTE FUER PACMAN *****
      **
  870 COLOR 9:PLOT 35,55:PLOT 135,65:
      SCOREPAC=SCOREPAC+1:FOR I=50 TO 1
      STEP -1:SOUND 0,I,10,8:NEXT I
  880 GOSUB 1030:RETURN
  890 REM *** MONSTERPUNKTE *******
  900 COLOR 0:PLOT 35,55:PLOT 135,65:
      SCOREMON=SCOREMON+1:FOR I=50 TO 1
      STEP -1:SOUND 0,I,10,8:NEXT I
  910 GOSUB 1030:RETURN
  920 REM **** PAC FRASS MONSTER *******
      ****
  930 IF PEEK(53260)=2 AND PEEK(704)=
      200 THEN SCOREPAC=SCOREPAC+5:
      GOSUB 1030:GOTO 950
  940 GOTO 960
  950 FOR I=200 TO 0 STEP -1:SOUND 0,I,
      10,8:NEXT I
  960 REM **** MON FRASS PACMAN ******
  970 IF PEEK(53261)=1 AND PEEK(705)=
      200 THEN SCOREMON=SCOREMON+5:
      GOSUB 1030:GOTO 990
  980 GOTO 1000
  990 FOR I=0 TO 200 STEP 1:SOUND 0,I,
      10,8:NEXT I:SOUND 0,0,0,0
 1000 X1=40:X2=190:Y1=172:Y2=172
 1010 DUM=USR(ADR(A$),PMBASE+200,PMBASE+
      1024,Y,240)
 1020 DUM=USR(ADR(A$),PMBASE+200,PMBASE+
      1024,5,0)
 1030 POKE 656,1:POKE 657,2:? "PACMAN=
      ";SCOREPAC;"                MONSTER =
      ";SCOREMON:RETURN
```



```
  1 REM BEISPIEL 4
  2 REM
  3 REM
  4 REM
  5 ? CHR$(125)
100 I=PEEK(106)-8
110 POKE 54279,I
120 PMBAS=I*256
130 FOR Q=PMBAS+512 TO PMBAS+640
140 POKE Q,0
150 NEXT Q
160 X=180:Y=100
170 POKE 704,90:POKE 559,46:POKE
    53277,3
180 POKE 53248,X
190 FOR Q=0 TO 8
200 READ P
210 POKE PMBAS+512+Y+Q,P
220 NEXT Q
230 DATA 12,12,8,63,72,136,20,34,65
240 POKE 53256,1
250 GOSUB 1000
260 GOSUB 1100
270 FOR Q=1 TO 80
280 B=USR(UP,PMBAS+511+Y):Y=Y-1
290 NEXT Q
300 FOR Q=1 TO 120
310 X=X-1:POKE 53248,X
320 NEXT Q
330 FOR Q=1 TO 80
340 B=USR(DOWN,PMBAS+511+Y):Y=Y+1
350 NEXT Q
360 FOR Q=1 TO 120
370 X=X+1:POKE 53248,X
380 NEXT Q
390 GOTO 270
000 DIM UPCODE$(21):UP=ADR(UPCODE$)
010 FOR I=UP TO UP+20
020 READ B:POKE I,B
030 NEXT I:RETURN
040 DATA 104,104,133,204,104,133,203
050 DATA 160,1,177,203,136,145,203
060 DATA 200,200,192,1,208,245,96
100 DIM DOWNCODE$(21):DOWN=
    ADR(DOWNCODE$)
110 FOR I=DOWN TO DOWN+20
120 READ B:POKE I,B
130 NEXT I:RETURN
140 DATA 104,104,133,204,104,133,203
150 DATA 160,10,177,203,200,145,203
160 DATA 136,136,192,255,208,245,96
```



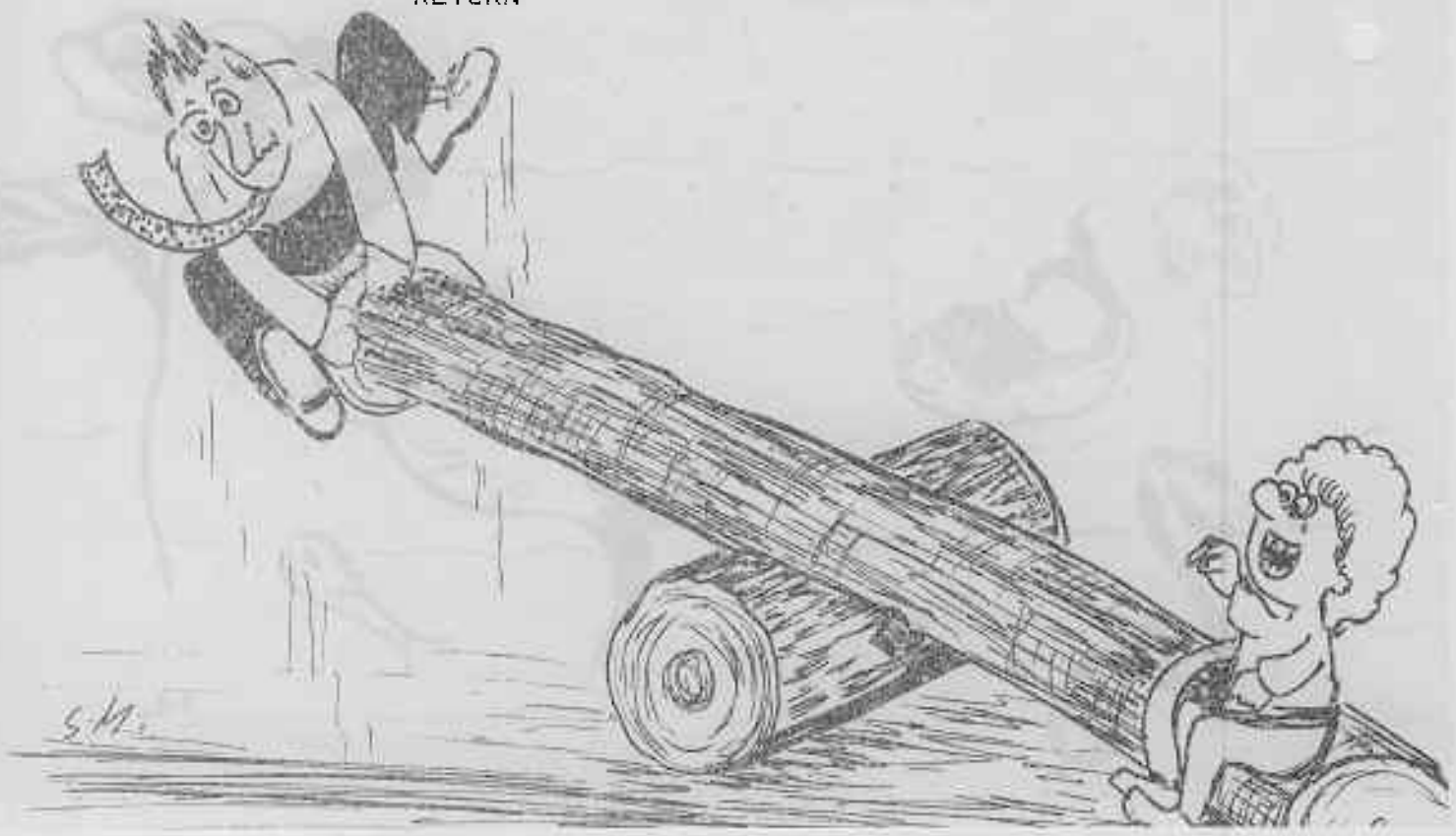
```
  1 REM BEISPIEL 5
  2 REM
  3 REM
  4 REM
  5 TRAP 10
 10 DIM X(8):MENU=1000:MOV=10000:COLR=
    11000:POKE 623,1
 20 FOR I=1 TO 4:X(I)=20+I*24:NEXT I:
    REM INITIAL POS OF P/M 1 TO 5
 30 FOR I=5 TO 8:X(I)=110+I*9:NEXT I
 40 TRAP 210:POKE 559,0
 50 PRINT "@CLEAR@"
 60 PC1=137:PC2=198:PC3=248:PC4=54:
    BAKC=0
 70 POKE 704,PC1:POKE 705,PC2:POKE
    706,PC3:POKE 707,PC4:PC5=28:REM
    SET UP INITIAL COLORS OF PLAYERS
 80 A=PEEK(106)-8:POKE 54279,A:PMBASE=
    256*A:REM PLAYER-MISSLE START
    ADDRESSEVOM PUNKT  8*256(2048)
 90 REM BYTES UNTERE BYTES MUESSEN
    NACH OBEN UND GEHE SICHER DAS ES
    AN DIE 2K GRENZE IST
100 REM POKE 559,46:POKE 53277,3:REM
    2 ZEILEN PM GRAFIK MIT TRICK
110 POKE 53277,3
120 POKE 752,1:REM MACHE CURSOR
    UNSICHTBAR UND BEWEGE IHN SPAETER
130 FOR I=PMBASE+384 TO PMBASE+1024:
    POKE I,0:NEXT I
140 FOR I=PMBASE+384 TO PMBASE+448:
    POKE I,255:NEXT I
150 FOR I=PMBASE+512 TO PMBASE+576:
    POKE I,255:NEXT I
160 FOR I=PMBASE+768 TO PMBASE+832:
    POKE I,255:NEXT I
170 FOR I=PMBASE+640 TO PMBASE+704:
    POKE I,255:NEXT I
180 FOR I=PMBASE+896 TO PMBASE+960:
    POKE I,255:NEXT I
190 POKE 53256,3:POKE 53257,3:POKE
    53258,3:POKE 53259,3:POKE 53260,
    255:REM INITIALISIERUNG DER
    SPIELER
200 FOR I=1 TO 8:POKE 53247+I,X(I):
    NEXT I
210 POKE 559,46
220 X=50:Y=50:POKE 710,BAKC:POKE 752,
    1:? "":GOSUB 720
230 FOR I=1 TO 5:POKE 53247+I,X(I):
    NEXT I:REM BEWEGE SPIELER AUF
    BILDSCHIRM
235 REM ***** HAUPT PROGRAMM****
240 S=STICK(0)
243 REM
245 IF PEEK(53279)=5 THEN GOSUB 1000
250 IF STRIG(0)=0 THEN GOSUB 380
260 IF S=15 THEN 240
270 X2=X1-(S=9)-(S=10)-(S=11)+(S=5)+
    (S=6)+(S=7)
280 Y2=Y1-(S=6)-(S=10)-(S=14)+(S=5)+
    (S=9)+(S=13)
290 IF X2<1 THEN X2=1
300 IF X2>37 THEN X2=37
310 IF Y2<1 THEN Y2=1
320 IF Y2>10 THEN Y2=10
340 POSITION X1,Y1:? "  "
350 POSITION X2,Y2:? "*"
360 X1=X2:Y1=Y2
370 GOTO 240
380 S=STICK(0)
400 IF S=15 THEN 380
```

```
  410 IF X2<7 THEN 460
  420 IF X2<13 THEN 510
  430 IF X2<19 THEN 560
  440 IF X2<25 THEN 610
  445 IF X2<34 THEN 655
  450 GOTO 660
  460 PC1=PC1+(S=14)-(S=13)
  470 IF PC1<2 THEN PC1=2:RETURN
  480 IF PC1>255 THEN PC1=255:RETURN
  490 POSITION 30,17:?  "    ":POSITION
      30,17:? PC1
  500 POKE 704,PC1:RETURN
  510 PC2=PC2+(S=14)-(S=13)
  520 IF PC2<0 THEN PC2=0:RETURN
  530 IF PC2>255 THEN PC2=255:RETURN
  540 POSITION 30,18:?  "    ":POSITION
      30,18:? PC2
  550 POKE 705,PC2:RETURN
  560 PC3=PC3+(S=14)-(S=13)
  570 IF PC3<0 THEN PC3=0:RETURN
  580 IF PC3>255 THEN PC3=255:RETURN
  590 POSITION 30,19:?  "    ":POSITION
      30,19:? PC3
  600 POKE 706,PC3:RETURN
  610 PC4=PC4+(S=14)-(S=13)
  620 IF PC4<0 THEN PC4=0:RETURN
  630 IF PC4>255 THEN PC4=255:RETURN
  640 POSITION 30,20:?  "    ":POSITION
      30,20:? PC4
  650 POKE 707,PC4:RETURN
  655 PC5=PC5+(S=14)-(S=13)
  656 IF PC5<0 THEN PC5=0:RETURN
  657 IF PC5>255 THEN PC5=255:RETURN
  658 POSITION 30,21:?  "    ":POSITION
      30,21:? PC5
  659 POKE 711,PC5:RETURN
  660 BAKC=BAKC+(S=14)-(S=13)
  670 IF BAKC<0 THEN BAKC=0:RETURN
  680 IF BAKC>255 THEN BAKC=255:RETURN
  690 POSITION 30,22:?  "    ":POSITION
      30,22:? BAKC
  700 POKE 710,BAKC:RETURN
  710 GOTO 380
  720 POSITION 10,15:? "SPIELER FARBEN":
      POSITION 10,16:? "**************"
  730 POSITION 10,17:? "SPIELER 0(POKE
      704)= ";PC1
  740 POSITION 10,18:? "SPIELER 1(POKE
      705)= ";PC2
  750 POSITION 10,19:? "SPIELER 2(POKE
      706)= ";PC3
  760 POSITION 10,20:? "SPIELER 3(POKE
      707)= ";PC4
  765 POSITION 10,21:? "SPIELER 4(POKE
      711)= ";PC5
  770 POSITION 10,22:? "HINTERGR.(POKE
      710) = ";BAKC
  780 RETURN
 1000 REM ***** 5. SPIELER *******
 1010 IF FLAG=0 THEN POKE 623,17:FLAG=1:
      GOTO 1030
 1020 IF FLAG=1 THEN POKE 623,1:FLAG=0
 1030 FOR I=5 TO 8:X(I)=110+I*8:NEXT I
 1040 FOR I=5 TO 8:POKE 53247+I,X(I):
      NEXT I
 1050 RETURN
10000 GRAPHICS 10:GOTO 10000
```

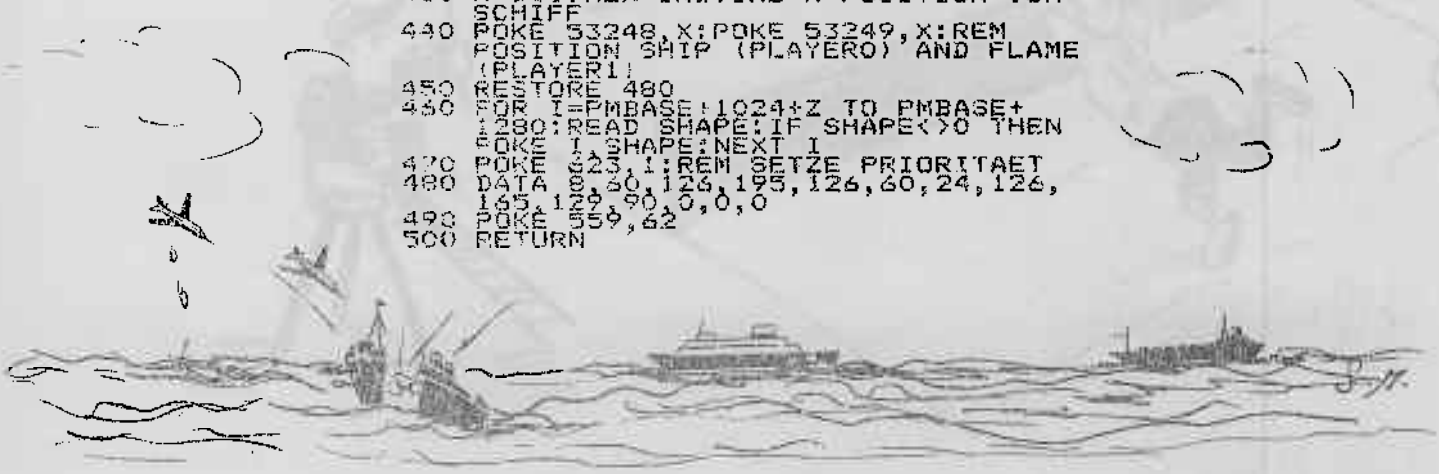
```
  1 REM BEISPIEL 6
  2 REM
  3 REM
  4 REM
 10 FOR I=1 TO 8:POKE 53247+I,0:NEXT
    I:GOTO 410
 20 REM ***** PLOT SHAPE ******
 30 COLOR 0
 40 D=D+(ST=6)+(ST=7)+(ST=5)-(ST=9)-
    (ST=10)-(ST=11)
 50 E=E+(ST=13)+(ST=9)+(ST=5)-(ST=6)-
    (ST=10)-(ST=14)
 60 FOR X=0 TO 9:FOR Y=0 TO 25:PLOT X+
    D,Y+E:NEXT Y:NEXT X
 70 RESTORE 150:COLOR 1
 80 FOR N=1 TO 100:READ X:READ Y
 90 IF X=0 AND Y=0 THEN RETURN
100 PLOT X+D,Y+E:NEXT N
110 E$(1,60)="hhhJH[O]{K}[ ]{B}[1K{H}
    {Q}KHHPw]hJH[O]{K}[ @BELL@1KH{Q}K
    {H}{H}Pw]hJH[O]{G}[FM%M{M}]
    {,}[P]hJ[O]{G}[FM%M{M}]{,}[P]{.}
    {,}{,}"
120 REM POKE 53248,30
140 SOUND 2,0,8,2:FOR VOL=1 TO 15
    STEP 0.1:SOUND 0,25,4,VOL:SOUND 1,
    13,4,VOL
150 DATA 5,0,3,1,4,1,5,1,6,1,2,2,3,2,
    4,2,5,2,6,2,7,2,1,3,2,3,7,3,8,3,2,
    4,3,4,4,4,5,4,6,4,7,4,3,5,4,5,5,5,
    6,5
160 DATA 4,6,5,6,2,7,3,7,4,7,5,7,6,7,
    7,7,1,8,3,8,6,8,8,8,1,9,8,9,2,10,
    4,10,5,10,7,10,2,11,4,11,5,11,7,
    11
170 DATA 2,12,7,12,3,13,6,13,3,14,6,
    14,4,15,5,15,4,16,5,16,4,17,5,17,
    4,18,5,18,3,19,4,19,5,19,6,19
180 DATA 2,20,3,20,4,20,5,20,6,20,7,
    20,1,21,2,21,3,21,4,21,5,21,6,21,
    7,21,8,21,2,22,3,22,4,22,5,22,6,
    22,7,22
190 DATA 3,23,4,23,5,23,6,23,4,24,5,
    24,0,0,0,0
200 REM *****PLAYER SETUP *****
210 DIM E$(60):GRAPHICS 6:D=20:E=50:
    POKE 559,0:GOSUB 20:? "
    WHICH IS THE PLAYER?":POKE 710,
    148:POKE 712,148
220 ? "SELECT = SIZES  OPTION = NEXT
    EX."
230 E$(1,60)="hhhJH[O]{K}[ ]{,}[1K{H}
    {Q}KHHPw]hJH[O]{K}[ ]{,}[1KH{Q}K
    {H}{H}Pw]hJH[O]{G}[FM%M{M}]
    {,}[P]hJ[O]{G}[FM%M{M}]{,}[P]{.}
    {,}{,}"
240 A=PEEK(106)-16:POKE 54279,A:POKE
    204,A+4:POKE 203,0:PMBASE=A*256
250 POKE 53277,3:POKE 704,40:POKE
    53248,150:POKE 205,120:POKE 53256,
    0
260 FOR I=PMBASE+1024 TO PMBASE+1280:
    POKE I,0:NEXT I
270 RESTORE 290
280 FOR I=PMBASE+1155 TO PMBASE+1209:
    READ B:POKE I,B:NEXT I
290 DATA 8,8,60,60,126,126,195,195,
    126,126,60,60,24,24,126,126,165,
    165
300 DATA 129,129,90,90,90,90,66,66,36,
    36,36,36,24,24,24,24,24,24,24,24
310 DATA 60,60,126,126,255,255,126,
```

```
    126,60,60,24,24,0,0,0,0,0
320 POKE 559,62
330 REM
340 REM ******HAUPTPROGRAMM***
350 REM
360 IF STRIG(0)=0 THEN GOSUB 20:FLAG=
    0
370 ST=STICK(0):S=PEEK(53279)
380 IF S=5 THEN GOSUB 670
390 IF S=3 THEN POKE 764,12:POKE
    53256,0:RUN
400 A=USR(ADR(E$),STICK(0)):GOTO 360
410 GRAPHICS 17:POKE 712,148
420 ? #6;"  DAS BEISPIEL ZEIGT "
430 ? #6;"  DEN VERGLEICH VON  "
440 ? #6;"NORMALEN SPIELER     "
450 ? #6;"UND SCHATTEN SPIELER"
460 ? #6;"NIMM DEN  STICK UND"
470 ? #6;"SIEH WER DER SPIELER"
480 ? #6;"IST.                 "
490 FOR I=1 TO 3000:NEXT I:GOTO 210
500 REM ***** PLOT SHAPE ******
510 COLOR 1
520 D=D+(ST=6)+(ST=7)+(ST=5)-(ST=9)-
    (ST=10)-(ST=11)
530 E=E+(ST=13)+(ST=9)+(ST=5)-(ST=6)-
    (ST=10)-(ST=14)
540 RESTORE 370
550 FOR N=1 TO 100:READ X:READ Y
560 IF X=0 AND Y=0 THEN 110
570 PLOT X+D,Y+E:NEXT N
580 IF FLAG=0 THEN COLOR 0:GOTO 130
590 RETURN
600 FOR X=1 TO 8:FOR Y=0 TO 25:PLOT X+
    D,Y+E:NEXT Y:NEXT X
610 RETURN
620 DATA 5,0,3,1,4,1,5,1,6,1,2,2,3,2,
    4,2,5,2,6,2,7,2,1,3,2,3,7,3,8,3,
    4,4,4,4,4,5,4,6,4,7,4,8,5,4,5,5,5,
    6,5
630 DATA 4,6,5,6,6,7,3,7,4,7,5,7,6,7,
    7,7,1,8,3,8,6,8,8,8,1,9,8,9,2,10,
    4,10,5,10,7,10,2,11,4,11,5,11,7,
    11
640 DATA 2,12,7,12,3,13,6,13,3,14,6,
    14,4,15,5,15,4,16,5,16,4,17,5,17,
    4,18,5,18,3,19,4,19,5,19,6,19
650 DATA 3,20,3,20,4,20,5,20,6,20,7,
    20,1,21,2,21,3,21,4,21,5,21,6,21,
    7,21,8,21,2,22,3,22,4,22,5,22,6,
    22,7,22
660 DATA 3,23,4,23,5,23,6,23,4,24,5,
    24,0,0,0
670 REM ***** SIZES *****
680 ? :? :? "PRESS 0, 1, OR 3 FOR
    SIZE":? "SELECT = SIZES    OPTION =
    NEXT EX."
690 POKE 764,255
700 PK=PEEK(764)
710 IF PK=50 THEN POKE 53256,0:RETURN
720 IF PK=31 THEN POKE 53256,1:RETURN
730 IF PK=26 THEN POKE 53256,3:RETURN
740 GOTO 700
```

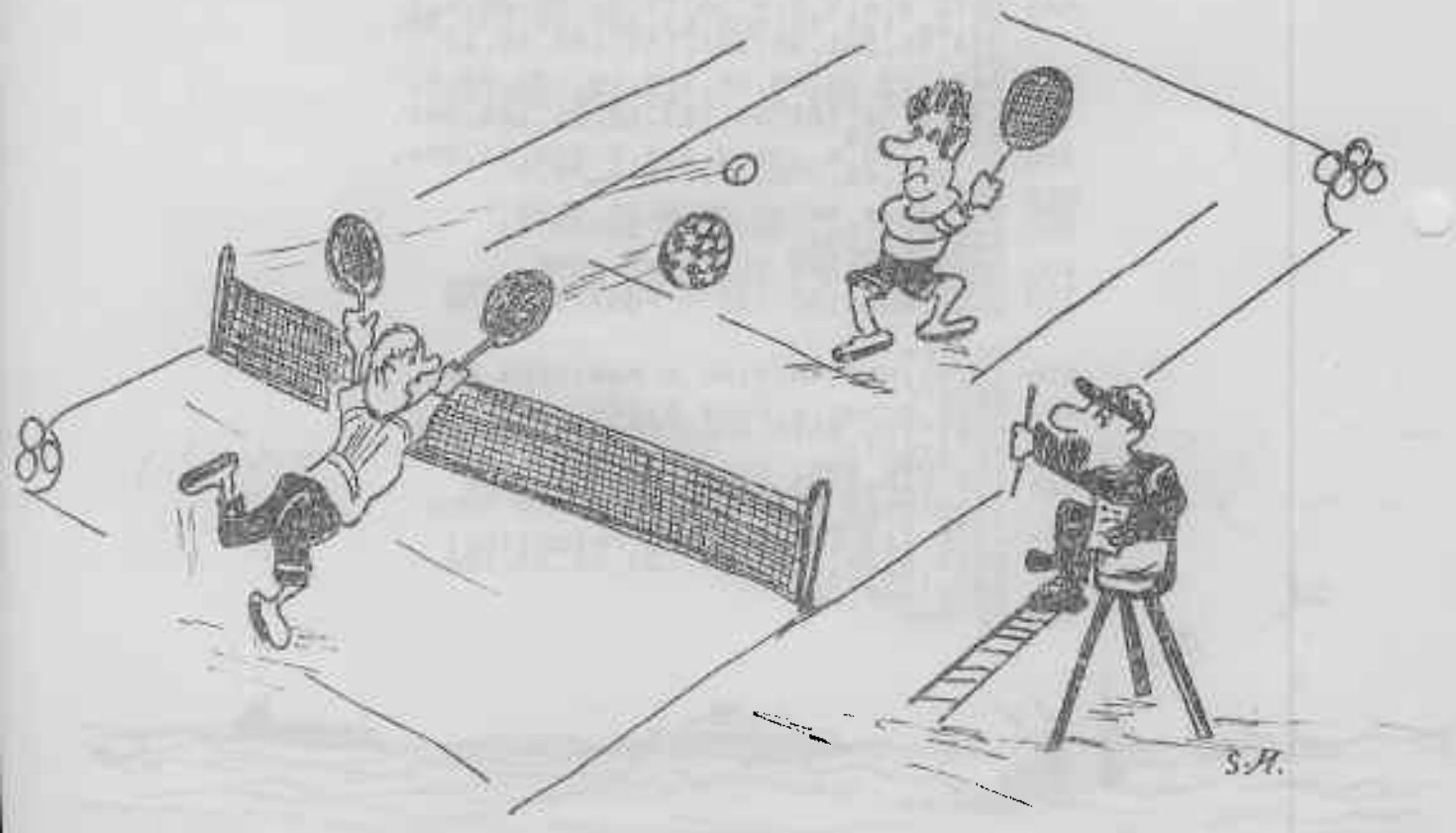
```
  1 REM BEISPIEL 7
  2 REM
  3 REM
  4 REM
 10 GOTO 140
 20 REM
 30 REM ****** HAUPT PROGRAMM ********
 40 REM
 50 ST=STICK(0)
 60 X=X+(ST=6)+(ST=7)+(ST=5)-(ST=9)-
    (ST=10)-(ST=11)
 70 Y=Y+(ST=13)+(ST=9)+(ST=5)-(ST=6)-
    (ST=10)-(ST=14)
 80 GOSUB 330
 85 IF PEEK(53279)=3 THEN POKE 764,12:
    RUN
 90 IF PADDLE(0)=228 THEN 110
100 X=PADDLE(0)
110 POKE 53248,X
120 GOTO 50
130 REM
140 REM PMB=PLAYER MISSILE BASIS
    ADRESSE    PAGES(256 BYTES PER
    PAGE)
150 REM
160 POKE 704,40
170 PMB=PEEK(106)-16
180 POKE 54279,PMB
190 PMBASE=PMB*256
200 POKE 53277,3
210 X=125:Y=100:YSAVE=100
220 POKE 53256,0
230 FOR I=PMBASE+1024 TO PMBASE+1280:
    POKE I,0:NEXT I
240 RESTORE 210:CNT=0
250 FOR I=PMBASE+1024+Y TO PMBASE+
    1280
260 READ B:IF B=0 THEN 310
270 CNT=CNT+1
280 POKE I,B
290 NEXT I
300 DATA 8,60,126,195,126,60,24,126,
    165,0,0
310 POKE 559,62
320 GOTO 50
330 REM
340 REM ***** HOCH UND RUNTER*****
350 IF YSAVE=Y THEN RETURN
360 IF YSAVE<Y THEN 430
370 RESTORE 210
380 FOR I=1 TO CNT
390 READ B
400 POKE PMBASE+1024+Y+I,B
410 NEXT I
420 YSAVE=Y:POKE PMBASE+1024+Y+CNT+1,
    0:RETURN
430 RESTORE 210
440 FOR I=1 TO CNT
450 READ B
460 POKE PMBASE+1024+Y+I-1,B
470 NEXT I
480 YSAVE=Y:POKE PMBASE+1024+Y-1,0:
    RETURN
```



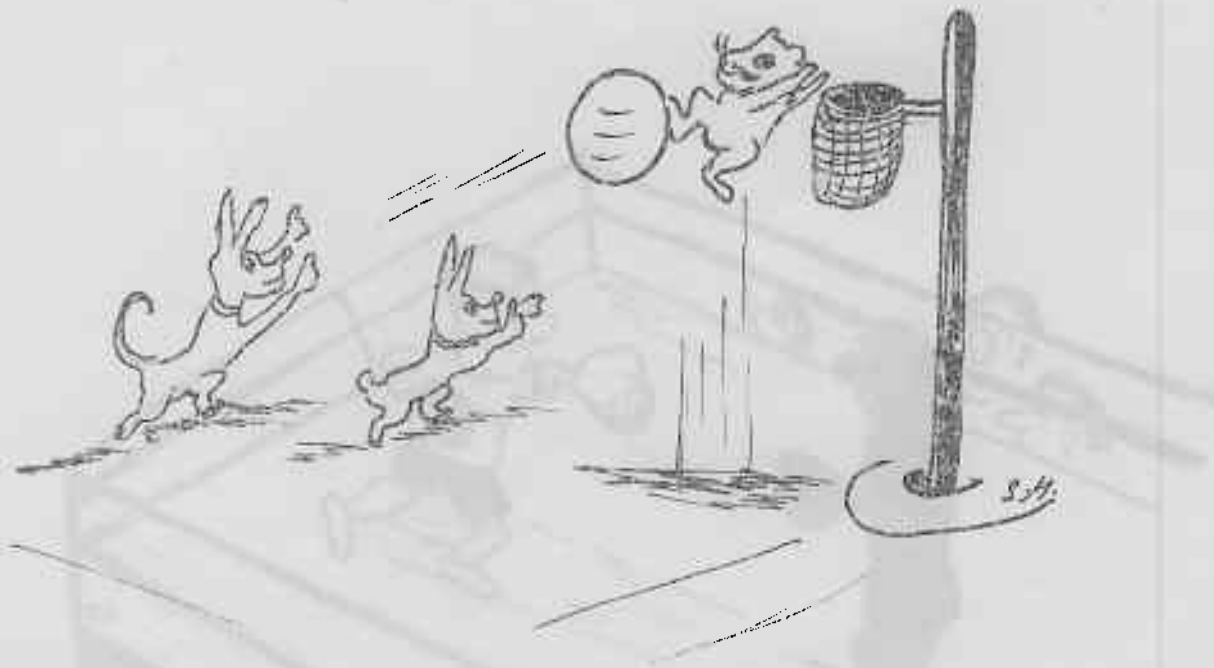
```
  1 REM BEISPIEL 8
  2 REM
  3 REM
  4 REM
 10 GOTO 30
 20 FOR I=PMBASE+1024 TO PMBASE+1536:
    POKE I,0:NEXT I:RETURN
 30 GOSUB 200:REM DRAW MOUNTAINS
 40 DIM E$(60)
 50 GOSUB 300:REM ZEICHNE STERNE
 60 POKE 710,0:POKE 712,0:POKE 704,
    123
 70 GOSUB 380:REM SETUP PLAYERS
 80 POKE 204,A+4:POKE 203,0:POKE 205,
    120
 90 E$(1,60)="hhhJH[O](K)[ 1(B)[1K(H)
    (Q)KHHPw]hJH[O](K)[ @BELL@1KH(@)K
    (H)(H)Pw]hJH[O](G)[FM%M(M)]
    (,)[P]hJ[O](G)[+M%M(M)](,)[P](.)
    (,)(,)"
100 REM POKE 53248,30
110 REM ***MAIN LOOP *********
120 SOUND 2,0,8,2:FOR VOL=1 TO 15
    STEP 0.1:SOUND 0,25,4,VOL:SOUND 1,
    13,4,VOL
130 A=USR(ADR(E$),STICK(0))
140 NEXT VOL
150 FOR VOL=14 TO 0 STEP -0.1:SOUND 0,
    25,4,VOL:SOUND 1,13,4,VOL
160 A=USR(ADR(E$),STICK(0))
170 NEXT VOL
180 IF PEEK(53279)=3 THEN POKE 764,12:
    RUN "D:EX5.9"
190 GOTO 110
200 REM **** PLOT HINTERGRUND*****
210 GRAPHICS 8+16:RESTORE 260
220 COLOR 1
230 READ X,Y:PLOT X,Y
240 READ X,Y:IF X>900 THEN RETURN
250 DRAWTO X,Y+10:GOTO 240
260 DATA 0,157,27,166,39,166,49,166,
    57,157,57,158,69,158,74,164,87,
    170,87,171,90,169,95,140
270 DATA 110,70,112,65,114,50,116,63,
    127,93,130,97,140,97,153,83,159,
    57,165,84,168,91,175,103,181,123
280 DATA 190,174,199,174,205,165,207,
    149,229,133,245,156,256,133,271,
    120,283,165,287,172,291,164,300,
    89,305,76
290 DATA 310,63,999,999
300 REM **** STERNE *****
310 TRAP 320
320 RESTORE 340
330 READ X,Y:PLOT X,Y:GOTO 330
340 DATA 2,4,5,8,9,10,13,43,34,65,73,
    23,66,123,45,300,67,301,65,246,34,
    234,49,261,68,241,133,123,55,67,
    98
350 DATA 12,64,75,58,119,10,133,43,4,
    65,7,93,203,66,12,45,178,67,145,
    65,48,34,187,49,123,88,21,133,243,
    55,118,74
360 DATA 123,5,156,7,213,8,295,12,296,
    34,294,56,230,56,230,6,78,9
370 RETURN
380 REM **** SETUP SPIELER ****
390 A=PEEK(106)-40:POKE 54279,A:
    PMBASE=A*256
400 GOSUB 20:REM CLEAR PM ZONE
410 POKE 53277,3:REM TURN ON PM DMA
420 Z=20:REM INITIAL Y POSITION VOM
    SCHIFF
430 X=150:REM INITIAL X POSITION VOM
    SCHIFF
440 POKE 53248,X:POKE 53249,X:REM
    POSITION SHIP (PLAYER0) AND FLAME
    (PLAYER1)
450 RESTORE 480
460 FOR I=PMBASE+1024+Z TO PMBASE+
    1280:READ SHAPE:IF SHAPE<>0 THEN
    POKE I,SHAPE:NEXT I
470 POKE 623,1:REM SETZE PRIORITAET
480 DATA 8,30,126,195,126,60,24,126,
    165,129,90,0,0,0
490 POKE 559,62
500 RETURN
```



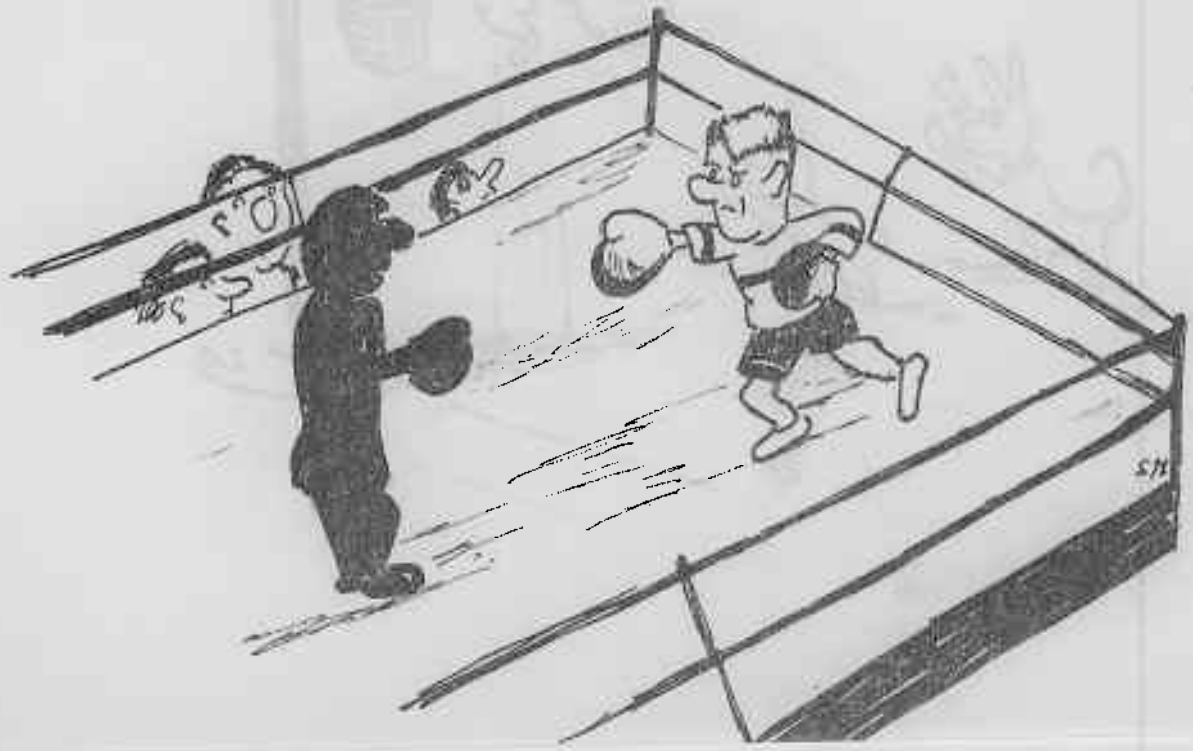
```
  1 REM BEISPIEL 9
  2 REM
  3 REM
  4 REM
  5 FOR I=1 TO 8:POKE 53247+I,0:NEXT
    I
  8 GOTO 100
 10 DIM E$(60)
 20 E$(1,60)="hhhJH[O](K)[ ](,)[1K(H)
    (Q)KHHPw]hJH[O](K)[ ](,)[1KH(Q)K
    (H)(H)Pw]hJH[O](G)[FM%M(M)]
    (,)[P]hJ[O](G)[fM%M(M)](,)[P1(.)
    (,)(.)"
 30 A=PEEK(106)-16:POKE 54279,A:POKE
    204,A+4:POKE 203,0:PMBASE=A*256
 40 POKE 559,62:POKE 53277,3:POKE 704,
    40:POKE 205,120:POKE 53256,1
 50 FOR I=PMBASE+1024 TO PMBASE+1280:
    POKE I,0:NEXT I
 60 FOR I=PMBASE+1100 TO PMBASE+1127:
    READ B:POKE I,B:NEXT I
 70 DATA 8,60,126,195,126,60,24,126,
    165,126,0,90,20,66,36,36,24,24,24,
    0,24,60,126,255,126,60,24,0
 75 POKE 53248,150
 80 A=USR(ADR(E$),STICK(0)):IF
    PEEK(53279)=3 THEN POKE 764,12:
    POKE 53256,0:RUN
 90 GOTO 80
100 GRAPHICS 17
102 ? #6;"DAS IST EIN BEISPIEL"
104 ? #6;"WAS FALSCH GEMACHT "
106 ? #6;"WERDEN KANN WENN SIE"
110 ? #6;"PLAYER/MISSILE DATA."
120 ? #6;"FALSCH EINGEBEN     "
125 ? #6;"                    "
130 ? #6;"                    "
140 ? #6;"                    "
150 ? #6;"                    "
160 ? #6;"                    "
162 FOR I=1 TO 3000:NEXT I:POKE 752,1
165 GRAPHICS 7:POKE 712,148
180 ? "BEWEGE SPIELER IN 2 SEITEN ":?
    "PRESS OPTION TO START"
190 GOTO 10
2000 REM ***** PLOT SCHATTEN ***
2010 COLOR 1
2015 RESTORE 2100
2020 FOR N=1 TO 100:READ X:READ Y
```



```
  1 REM BEISPIEL 10
  2 REM
  3 REM
  4 REM
 10 FOR I=1 TO 8:POKE 53247+I,0:NEXT
    I:POKE 559,0:GOTO 30
 20 FOR I=PMBASE+512 TO PMBASE+1280:
    POKE I,0:NEXT I:RETURN
 30 GRAPHICS 0:? :? :? " PRESS OPTION
    TO START":POKE 752,1
 40 ? :? " PRESS SELECT FUER EINFACHE
    LINIE          RESOLUTION PLAYER"
 50 ? :? " PRESS START FUER DOPPELTE
    LINIE          RESOLUTION PLAYER"
 70 POKE 704,40
 80 PMB=PEEK(106)-16
 90 POKE 54279,PMB
100 PMBASE=PMB*256
110 POKE 53277,3:GOSUB 20:POKE 559,34
120 X=125:Y=110:YSAVE=100
130 POKE 53256,0:GOTO 200
140 REM ***** HAUPTPROGRAMM****
150 P=PEEK(53279)
160 IF P=5 THEN POKE 559,62:POKE
    53248,100
170 IF P=6 THEN POKE 559,46:POKE
    53248,100
180 IF P=3 THEN POKE 764,12:RUN
190 GOTO 150
200 REM
210 REM SINGLE LINE RESOLUTION *******
    *
220 REM
230 RESTORE 290:CNT=0
240 FOR I=PMBASE+1024+Y TO PMBASE+
    1280
250 READ B:IF B=0 THEN 300
260 CNT=CNT+1
270 POKE I,B
280 NEXT I
290 DATA 124,214,214,254,124,68,68,
    204,0,0,0
300 REM
310 REM DOUBLE LINE RESOLUTION ******
320 REM
330 RESTORE 290:CNT=0
340 FOR I=PMBASE+512+Y TO PMBASE+640
350 READ B:IF B=0 THEN 390
360 CNT=CNT+1
370 POKE I,B
380 NEXT I
390 GOTO 150
```



```
  1 REM BEISPIEL II
  2 REM
  3 REM
  4 REM
 10 FOR I=1 TO 8:POKE 53247+I,0:NEXT
    I:GOTO 240
 20 FOR I=PMBASE+1024 TO PMBASE+1792:
    POKE I,0:NEXT I:RETURN
 30 REM *****PLAYER SETUP *****
 40 GRAPHICS 7:D=20:E=50:POKE 710,0:
    POKE 712,0:POKE 704,41:POKE 706,
    41:POKE 559,62
 45 COLOR 2:PLOT 1,57:DRAWTO 159,57:
    PLOT 1,65:DRAWTO 159,65
 50 A=PEEK(106)-32:POKE 54279,A:POKE
    204,A+4:POKE 203,0:PMBASE=A*256
 60 POKE 53277,3:X=150:POKE 208,120:
    POKE 53256,0:POKE 53256,150:POKE
    53250,0
 70 GOSUB 20
 80 RESTORE 110:Y=150
 90 FOR I=PMBASE+1024+Y TO PMBASE+
    1209:READ B:IF B<>0 THEN POKE I,B:
    NEXT I
100 REM ***** DATA FOR PACMAN1 *****
110 DATA 28,62,119,127,254,252,248,
    252,254,127,126,62,28,0,0
120 RESTORE 150
130 FOR I=PMBASE+1536+Y TO PMBASE+
    1792:READ B:IF B<>0 THEN POKE I,B:
    NEXT I
140 REM ***** DATA FOR PACMAN2 ******
150 DATA 28,62,119,255,255,255,255,
    255,126,126,62,28,0,0,0,0
160 POKE 559,62
170 REM
180 REM ******HAUPT PROGRAMM**
190 REM
210 ST=STICK(0):S=PEEK(53279)
220 IF S=3 THEN POKE 764,12:RUN
222 IF ST=15 THEN 210
225 X=X+(ST=7)-(ST=11):POKE 53248,X:
    POKE 53250,0:FOR J=1 TO 50:NEXT J
226 POKE 53250,X:POKE 53248,0
230 GOTO 210
240 GRAPHICS 17:POKE 712,69:POKE 710,
    19
250 ? #6;"HIER WERDEN DIE      "
260 ? #6;"SCHATTEN LEBENDIG    "
270 ? #6;"GEMACHT ZWISCHEN DEN"
280 ? #6;"2 SPIELERN UND PAC   "
290 ? #6;"MAN.                 "
300 ? #6;"NIMM DEN JOYSTICK    "
310 ? #6;"UND SCHAUE SELBER.   "
320 FOR I=1 TO 3000:NEXT I:GOTO 40
330 POKE 764,255
```

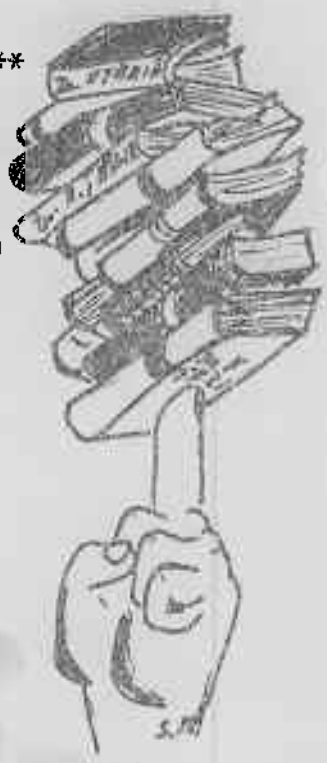


```
  1 REM BEISPIEL 12
  2 REM
  3 REM
  4 REM
 10 FOR I=1 TO 8:POKE 53247+I,0:NEXT
    I:GOTO 270
 20 FOR I=PMBASE+1024 TO PMBASE+1536:
    POKE I,0:NEXT I:RETURN
 30 REM *****PLAYER SETUP *****
 40 GRAPHICS 7:POKE 710,0:POKE 712,69:
    POKE 704,41:POKE 706,41:POKE 559,
    62
 45 POKE 708,103:POKE 752,1:POKE 709,
    139:POKE 710,100
 47 COLOR 1:FOR X=20 TO 150 STEP 30:
    PLOT X,61:PLOT X+1,61:NEXT X
 50 COLOR 2:PLOT 1,57:DRAWTO 159,57:
    DRAWTO 159,66:DRAWTO 1,66:DRAWTO
    1,57
 52 PLOT 157,57:DRAWTO 157,66:PLOT
    158,57:DRAWTO 158,66
 55 COLOR 3:FOR J=1 TO 4:FOR K=1 TO 5
 56 T=K+J*30
 57 PLOT T,57:DRAWTO T,66:NEXT K:NEXT
    J
 60 A=PEEK(106)-32:POKE 54279,A:POKE
    204,A+4:POKE 203,0:PMBASE=A*256
 70 POKE 53277,3:X=50:POKE 205,120:
    POKE 53256,0:POKE 53250,150
 80 GOSUB 20
 90 RESTORE 120:Y=150:ERAX=-10
100 FOR I=PMBASE+1024+Y TO PMBASE+
    1280:READ B:IF B=999 THEN 110
105 POKE I,B:NEXT I
110 REM ***** DATA FOR PACMAN1 *****
120 DATA 28,62,119,127,254,252,248,
    252,254,127,126,62,28,999
130 RESTORE 160
140 FOR I=PMBASE+1536+Y TO PMBASE+
    1792:READ B:IF B=999 THEN 150
145 POKE I,B:NEXT I
150 REM ***** DATA FOR PACMAN2 ******
160 DATA 28,62,118,255,255,255,255,
    255,126,126,62,28,999
170 POKE 559,62
180 REM
190 REM ******HAUPT PROGRAMM**
200 REM
205 POKE 53278,1:REM CLEAR ALL
    COLISION LOCATIONS (SO THEY HOLD
    0'S)
220 ST=STICK(0):S=PEEK(53279)
230 IF S=3 THEN POKE 764,12:RUN
240 X=X+(ST=7)-(ST=11):POKE 53248,X:
    POKE 53250,0:FOR J=1 TO 50:NEXT J
242 IF PEEK(53252)<>0 THEN GOSUB 1000:
    GOTO 220
245 IF X>220 THEN X=40
250 POKE 53250,X:POKE 53248,0
260 GOTO 220
270 GRAPHICS 17:POKE 712,245
280 ? #6;"WIR SEHEN UNS DIE    "
290 ? #6;"KOLLISIONS REGISTER  "
300 ? #6;"AN DER MAUER AN UND  "
310 ? #6;"DIE PRIORITAETEN     "
320 ? #6;"DER ANDEREN SCHATTEN"
330 ? #6;"NIMM DEN JOYSTICK    "
340 ? #6;"UND LAUFE MIT DEM    "
350 ? #6;"PACMAN REIN UND RAUS"
360 ? #6;"AUS DEM TUNNEL       "
365 FOR I=1 TO 2000:NEXT I
370 GOTO 30


1000 REM **** COLLISION SOUND ****
1002 COL=PEEK(53252)
1004 IF COL=2 THEN 1015
1005 IF COL<>1 THEN 1030
1006 ? "YUMMY!!":? :?
1007 FOR I=200 TO 0 STEP -1:SOUND 0,I,
     10,8:NEXT I:? :? :?
1008 COLOR 0:ERAX=ERAX+30:PLOT ERAX,61:
     PLOT ERAX+1,61
1009 X=X+8:POKE 53248,X:POKE 53250,X
1010 GOTO 1030
1015 SOUND 0,50,8,8
1020 FOR I=1 TO 30:NEXT I:SOUND 0,0,0,
     0
1030 POKE 53278,1:RETURN
```

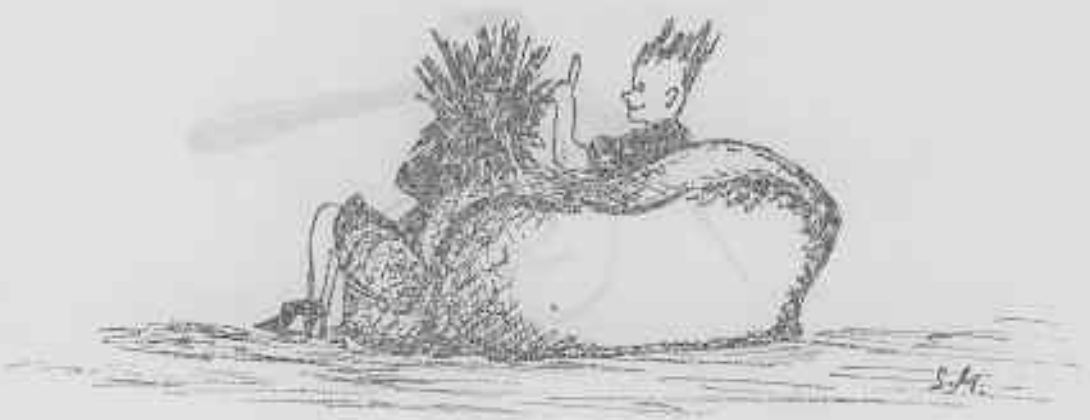
```
  1 REM BEISPIEL 13
  2 REM
  3 REM
  4 REM
 10 FOR I=1 TO 8:POKE 53247+I,0:NEXT
    I:GOTO 240
 20 FOR I=PMBASE+1024 TO PMBASE+1792:
    POKE I,0:NEXT I:RETURN
 30 REM *****PLAYER SETUP *****
 40 GRAPHICS 7:POKE 710,0:POKE 712,0:
    POKE 704,41:POKE 708,41:POKE 559,
    62
 50 A=PEEK(106)-24:POKE 54279,A:POKE
    204,A+4:POKE 203,0:PMBASE=A*256
 60 POKE 53277,3:X=150:POKE 205,120:
    POKE 53256,0:POKE 53250,150
 70 GOSUB 20
 80 RESTORE 110:Y=150
 90 FOR I=PMBASE+1024+Y TO PMBASE+
    1209:READ B:IF B<>0 THEN POKE I,B:
    NEXT I
100 REM ***** DATA FOR FIRST HALF ***
110 DATA 2,2,2,2,2,2,2,2,2,4,4,4,8,
    112,0,0,0
120 RESTORE 150
130 FOR I=PMBASE+1536+Y TO PMBASE+
    1792:READ B:IF B<>0 THEN POKE I,B:
    NEXT I
140 REM ***** DATA FOR 2ND HALF *****
150 DATA 160,160,160,160,160,160,160,
    160,160,144,144,144,136,135,0,0,0
160 POKE 559,62
165 ? "                16 BIT WEITER
    SCHATTEN"
167 POKE 752,1
168 ? "          <====NIMM JOYSTICK =====
    >"
170 REM
180 REM ******HAUPT PROGRAMM**
190 REM
200 REM
210 ST=STICK(0):S=PEEK(53279)
220 IF S=3 THEN POKE 764,12:RUN
225 X=X+(ST=7)-(ST=11):POKE 53248,X:
    POKE 53250,X+8
230 GOTO 200
240 GRAPHICS 17
250 ? #6;"DAS NAECHSTE          "
260 ? #6;"          BEISPIEL    "
270 ? #6;"DER BEWEGUNG          "
320 FOR I=1 TO 3000:NEXT I:GOTO 40
```

```
  1 GRAPHICS 2+16:SETCOLOR 4,5,5:
    POSITION 2,5:? #6;"INTERMISSION":
    FOR I=1 TO 1000:NEXT I:POKE 559,0
  5 DIM X(4),M(19,7),DAT(19),PWR(7),
    FF(5)
  6 DIM MO$(20),M1$(20),M2$(20),
    M3$(20),M4$(20),MX$(20),MZ$(22),
    MC$(100),P$(128)
  9 XPN0=90:XPN1=150:XPN2=75:XPN3=120:
    XPN4A=176:XPN4B=174:XPN4C=172:
    XPN4D=170:F0=24:F1=24:F2=44:F3=44:
    F5=44
 10 RESTORE 30010:FOR I=0 TO 7:READ
    RD:PWR(I)=RD:NEXT I
 11 FF(0)=0:FF(1)=24:FF(2)=24:FF(3)=
    44:FF(4)=44:FF(5)=44
 12 FOR I=0 TO 19:FOR J=0 TO 7:M(I,J)=
    0:NEXT J:NEXT I
 13 FOR I=0 TO 4:X(I)=0:NEXT I
 14 FOR I=0 TO 19:DAT(I)=0:NEXT I
 15 MC$(1)="(,)":MC$(100)="(,)":
    MC$(2)=MC$(1):MO$="":M1$="":M2$=
    "":M3$="":M4$="":MX$="":MZ$="(,)
    (,)(,)(,)(,)(,)(,)(,)(,)(,)(,)
    (,)(,)(,)(,)(,)(,)(,)(,)(,)"
 16 OPC=0:RWC=0:CLC=0:MLB=7010:OPN1=
    7040:RAW=7070:CLS1=7090:TRAP MLB
 19 GOTO 30
 20 IF LEN(MX$)<>20 THEN RETURN
 21 IF PM<>5 THEN 23
 22 POKE PMBASE+383+F,0
 23 POKE PMBASE+511+F+128*(PN-1),0
 24 FOR I=0 TO 19
 26 NUM=ASC(MX$(I+1,I+1))
 28 IF PN<>5 THEN 32
 30 POKE PMBASE+384+I+F,NUM:GOTO 34
 32 POKE PMBASE+512+F+128*(PN-1)+I,
    NUM
 34 NUM=0
 35 NEXT I
 36 IF PN<>5 THEN 38
 37 POKE PMBASE+384+I+F,0
 38 POKE PMBASE+512+F+128*(PN-1)+I,0
 39 RETURN
 50 POKE 53248,0:POKE 53249,0:POKE
    53250,0:POKE 53251,0:POKE 53252,0:
    POKE 53253,0:POKE 53254,0:POKE
    53255,0
 59 GOTO 300
 60 REM *** BLANK PLOT-ZERO MATRIX ***
    "
 62 POKE 87,5:POKE 88,P882:POKE 89,
    P892
 64 COLOR 1:FOR Y=0 TO 19:FOR X=1 TO
    8:PLOT X,Y:NEXT X:NEXT Y
 65 FOR I=0 TO 19:FOR J=0 TO 7:M(I,J)=
    0:NEXT J:NEXT I
 66 RETURN
 70 REM *** BLANK PLOT-ZERO MATRIX ***
    "
 72 POKE 87,5:POKE 88,P882:POKE 89,
    P892
 74 COLOR 1:FOR Y=0 TO 19:FOR X=1 TO
    8:PLOT X,Y:NEXT X:NEXT Y
 76 FOR I=0 TO 19:FOR J=0 TO 7:M(I,J)=
    0:NEXT J:NEXT I
 78 ON PN GOSUB 80,81,82,83,84
 80 MO$="":GOTO 86
 81 M1$="":GOTO 86
 82 M2$="":GOTO 86
 83 M3$="":GOTO 86
 84 M4$="":GOTO 86
```

```
  86 MX$="":X(PN-1)=0
  88 RETURN
  90 REM REBUILD THE MATRIX
  92 ON PN GOTO 94,95,96,97,98
  94 MX$=M0$:GOTO 101
  95 MX$=M1$:GOTO 101
  96 MX$=M2$:GOTO 101
  97 MX$=M3$:GOTO 101
  98 MX$=M4$:GOTO 101
 100 REM REBUILD THE MATRIX
 101 IF LEN(MX$)<>20 THEN RETURN
 102 F=FF(PN)
 104 POKE 87,5:POKE 88,P882:POKE 89,
     P892
 106 FOR J=0 TO 19
 108 NUM=ASC(MX$(J+1,J+1))
 114 IF PN<>5 THEN 118
 116 POKE PMBASE+384+J+F,NUM:GOTO 120
 118 POKE PMBASE+512+F+128*(PN-1)+J,
     NUM
 120 FOR I=0 TO 7
 122 IF NUM<PWR(I) THEN M(J,I)=0:GOTO
     132
 124 NUM=NUM-PWR(I):M(J,I)=PWR(I)
 125 COLOR 2:PLOT I+1,J
 132 NEXT I
 133 NEXT J
 134 X(PN-1)=255
 135 RETURN
 200 REM CONVERT PLAYER TO CHAR
 201 MX$="":NUM=0
 202 F=FF(PN)
 205 POKE 53248,XPN0:POKE 53249,XPN1:
     POKE 53250,XPN2:POKE 53251,XPN3
 206 POKE 53252,XPN4A:POKE 53253,XPN4B:
     POKE 53254,XPN4C:POKE 53255,XPN4D
 207 POKE 87,5:POKE 88,P882:POKE 89,
     P892
 209 X(PN-1)=255
 210 FOR I=0 TO 19:FOR J=0 TO 7:NUM=
     NUM+M(I,J):NEXT J
 220 MX$(LEN(MX$)+1)=CHR$(NUM)
 221 REM PUT ONE LINE TO SCREEN
 222 IF PN<>5 THEN 226
 224 POKE PMBASE+384+I+F,NUM:GOTO 228
 226 POKE PMBASE+512+F+128*(PN-1)+I,
     NUM
 228 NUM=0
 229 NEXT I
 230 ON PN GOTO 231,232,233,234,235
 231 M0$=MX$:RETURN
 232 M1$=MX$:RETURN
 233 M2$=MX$:RETURN
 234 M3$=MX$:RETURN
 235 M4$=MX$:RETURN
 249 RETURN
 300 REM ******* SETUP ROUTINES *******
     *
 320 P559=PEEK(559)
 325 POKE 623,17:REM SET PLAYER
     PRIORITIES
 330 POKE 712,BAK
 335 A=PEEK(106)-32:POKE 54279,A:
     PMBASE=256*A:REM SET PLAYER-
     MISSLE STARTING ADDRESS AT A
     POINT 8*256(2048)
 340 REM BYTES BELOW THE TOP OF RAM TO
     ALLOW ROOM AND BE SURE IT'S ON A
     2K BOUNDRY.
 345 POKE 53277,3:REM ENABLE PM
     GRAPHICS WITH 2-LINE THICKNESS
 350 POKE 752,1:REM MAKE CURSOR
```



```
    INVISABLE
360 FOR I=PMBASE+384 TO PMBASE+1024:
    POKE I,0:NEXT I:REM CLEAR OUT P/M
    AREA
365 REM THIS IS TO PREVENT RANDOM
    JUNK.
370 POKE 53256,0:POKE 53260,18:REM
    INITIAL SIZE OF PLAYERS AND ALL
    INITIAL SIZE OF PLAYERS AND ALL
    MISSLES
372 REM *** SET UP SPLIT SCREEN ***
374 GRAPHICS 5:DL=PEEK(560)+256*
    PEEK(561)
376 PC1=133:PC2=198:PC3=246:PC4=0:
    TEXT=40:POKE 708,0
377 POKE 704,PC1:POKE 705,PC2:POKE
    706,PC3:POKE 707,PC4:POKE 712,
    TEXT:REM SET UP INITIAL COLORS OF
    PLAYERS
378 PDL4=PEEK(DL+4):PDL5=PEEK(DL+5)
379 I=1:RESTORE 30000
380 READ A:IF A=0 THEN 400
381 POKE (DL+I-1),A
382 IF I=5 THEN POKE DL+4,PDL4
383 IF I=6 THEN POKE DL+5,PDL5
384 I=I+1:GOTO 380
400 REM **** START OF MODES ****
401 REM ****   SCREEN DATA    ****
410 START1=PDL4+256*PDL5:REM START OF
    SCREEN DATA AREA
420 START2=START1+80:P892=
    INT(START2/256):P882=START2-P892*
    256:REM CALCULATE WHERE DATA FOR
    MODE 2 STARTS
430 START3=START2+20*20:REM # BYTES
    IN SECOND AREA
440 P893=INT(START3/256):P883=START3-
    P893*256:REM START OF MODE 3 DATA
    IN MEMORY
450 POKE 752,1:POKE 87,0:POKE 88,P883:
    POKE 89,P893
454 POKE 87,0:POKE 88,PDL4:POKE 89,
    PDL5:POSITION 0,1:? #6;"8    1
      PLAYER 0          PLAYER 1"
458 POKE 87,0:POKE 88,P883:POKE 89,
    P893:POKE 559,46
462 POSITION 0,0:? #6;"       PLAYER 2
      PLAYER 3    MIS/PL. #4"
470 GOSUB 60
474 X=1:Y=0
476 GOSUB 7000
478 POSITION 0,10:? #6;"EDITING NEW
    (N)  OR OLD (L) DATA?"
479 POKE 764,255
480 D=PEEK(764)
482 IF D=35 THEN 490
484 IF D=0 THEN 487
486 GOTO 480
487 POSITION 0,10:? #6;"PREPARE TAPE
    OR DISK THEN ENTER L       ":GOTO
    500
490 GOSUB 3000
500 REM ****** MAIN MOVE ROUTINE *****

501 POKE 559,46:POKE 764,255
502 POKE 87,5:POKE 88,P882:POKE 89,
    P892
508 XX=0:YY=0:QQ=1:NN=2
510 D=PEEK(764)
511 COLOR NN:PLOT XX+1,YY
512 COLOR QQ:PLOT XX+2,YY
513 IF PEEK(53272)=3 THEN GOSUB 3000
514 IF PEEK(53279)=6 THEN GOSUB 200
```

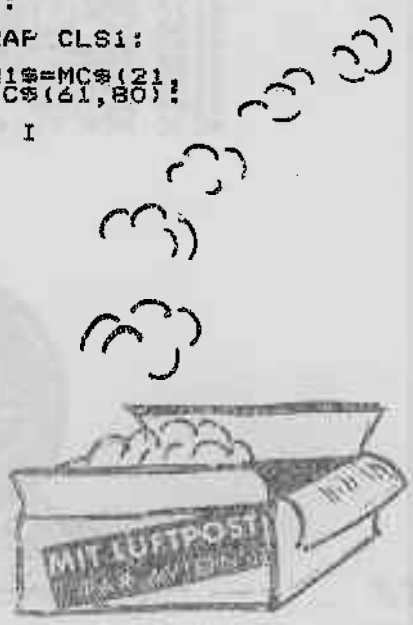


```
515 IF PEEK(53279)=5 THEN GOSUB 2000
516 POKE 87,5:POKE 88,P882:POKE 89,
    P892
517 IF D=255 AND STICK(0)=15 THEN 510
518 IF D=6 AND XX<>0 THEN XX=XX-1:
    GOTO 750
520 IF D=7 AND XX<>7 THEN XX=XX+1:
    GOTO 750
522 IF D=14 AND YY<>0 THEN YY=YY-1:
    GOTO 750
524 IF D=15 AND YY<>19 THEN YY=YY+1:
    GOTO 750
526 IF D=58 THEN 700:REM DRAW (D)
528 IF D=42 THEN 710:REM ERASE (E)
530 TT=STICK(0):IF TT<>15 THEN GOSUB
    800
540 IF D=18 THEN GOSUB 70:GOTO 548
542 IF D=62 THEN GOSUB 200:GOSUB 4100:
    REM SAVE CHANGE
544 IF D=0 THEN GOSUB 600:REM LOAD
    CHANGE
548 POKE 764,255
549 GOTO 510
600 REM INPUT DATA FROM TAPE AND DISK
602 POKE 87,0:POKE 88,P883:POKE 89,
    P893
604 POSITION 0,10:? #6;"INPUT FROM
    TAPE (T) OR DISK (D)?          "
606 POSITION 0,11:? #6;"           "
608 POSITION 0,12:? #6;"           ":POKE
    764,255
610 D=PEEK(764)
620 IF D=45 THEN GOTO 650:REM TAPE
630 IF D=58 THEN GOTO 4600:REM DISK
640 GOTO 610
650 REM INPUT FROM TAPE
654 TRAP OPN1:OPEN #1,4,0,"C:"
659 TRAP RAW:INPUT #1,P$:P$=""
660 INPUT #1,MC$:TRAP CLS1:CLOSE #1
662 TRAP MLB:M0$=MC$(1,20):M1$=MC$(21,
    40):M2$=MC$(41,60):M3$=MC$(61,80):
    M4$=MC$(81,100)
671 FOR I=0 TO 4:X(I)=0:NEXT I
672 FOR I=1 TO 20
673 IF M0$(I,I)<>"{,}" THEN X(0)=255
674 IF M1$(I,I)<>"{,}" THEN X(1)=255
675 IF M2$(I,I)<>"{,}" THEN X(2)=255
676 IF M3$(I,I)<>"{,}" THEN X(3)=255
677 IF M4$(I,I)<>"{,}" THEN X(4)=255
678 NEXT I
680 GOSUB 3000:OPN=255
689 RETURN
700 REM TURN ON SPOT
701 QQ=2:NN=1
702 M(YY,XX)=PWR(XX)
704 GOTO 510
710 REM TURN OFF SPOT
712 QQ=1:NN=2
713 M(YY,XX)=0
714 GOTO 510
750 IF QQ=2 THEN M(YY,XX)=PWR(XX):
    GOTO 548
752 IF QQ=1 THEN M(YY,XX)=0:GOTO 548
754 GOTO 548
799 GRAPHICS 0
800 REM ROUTINE FOR MOVING PLAYERS
810 TT=STICK(0):X1=0:Y1=0
812 IF TT=15 THEN RETURN
813 IF TT=14 THEN Y1=-1
814 IF TT=7 THEN X1=1
```

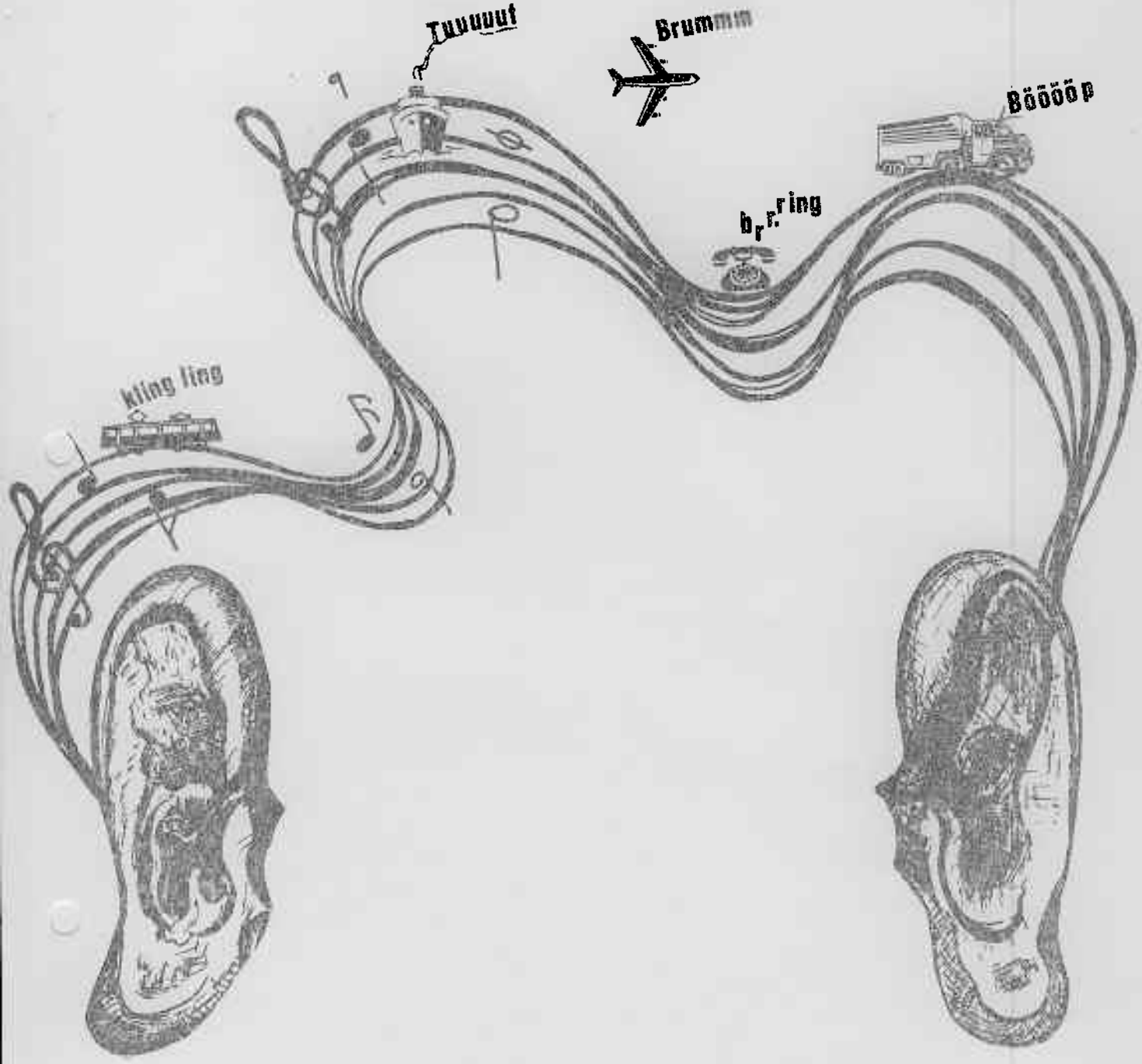
```
  815 IF TT=13 THEN Y1=1
  816 IF TT=11 THEN X1=-1
  817 FF(PN)=FF(PN)+Y1:F=FF(PN)
  818 ON PN GOTO 820,822,824,826,828
  820 MX$=M0$:XPN0=XPN0+X1:GOTO 837
  822 MX$=M1$:XPN1=XPN1+X1:GOTO 837
  824 MX$=M2$:XPN2=XPN2+X1:GOTO 837
  826 MX$=M3$:XPN3=XPN3+X1:GOTO 837
  828 MX$=M4$:XPN4A=XPN4A+X1
  829 MX$=M4$:XPN4B=XPN4B+X1
  830 MX$=M4$:XPN4C=XPN4C+X1
  831 MX$=M4$:XPN4D=XPN4D+X1:GOTO 837
  832 IF X1=0 THEN 840
  838 POKE 53248,XPN0:POKE 53249,XPN1:
      POKE 53250,XPN2:POKE 53251,XPN3
  839 POKE 53252,XPN4A:POKE 53253,XPN4B:
      POKE 53254,XPN4C:POKE 53255,XPN4D:
      GOTO 810
  840 IF Y1<>0 THEN GOSUB 20:GOTO 810
  849 GOTO 810
  850 NOXX=STICK(0)
  860 ? NOXX
  870 GOTO 850
 2000 REM SET PLAYER SIZE
 2010 GOSUB 7000
 2020 POKE 87,0:POKE 88,P883:POKE 89,
      P893
 2030 POSITION 0,11:? #6;"      [PRESS]
      [0],[1] OR [3] FOR PLAYER SIZE
      "
 2040 POKE 764,255
 2050 P=PEEK(764):IF P=255 THEN 2050
 2060 IF P=50 THEN V=0
 2062 IF P=31 THEN V=1
 2064 IF P=26 THEN V=3
 2066 IF P<>50 AND P<>31 AND P<>26 THEN
      2050
 2068 POKE 53255+PN,V
 2069 GOSUB 4000
 2070 RETURN
 3000 REM SELECT THE PLAYER TO EDIT
 3004 GOSUB 7000
 3006 IF OPN<6 THEN 3024
 3008 POSITION 0,0:? #6;"      "
 3024 POSITION 5,10:? #6;"ENTER PLAYER
      # TO EDIT?"
 3028 POKE 764,255:OPN=PN
 3030 A=PEEK(764):IF A=255 THEN 3030
 3040 IF A=50 THEN PN=1:GOTO 3056
 3042 IF A=31 THEN PN=2:GOTO 3056
 3044 IF A=30 THEN PN=3:GOTO 3056
 3046 IF A=26 THEN PN=4:GOTO 3056
 3048 IF A=24 THEN PN=5:GOTO 3056
 3049 POKE 87,0:POKE 88,P883:POKE 89,
      P893
 3050 POSITION 5,10:? #6;"**ERROR!
      INPUT 0,1,2,3, OR 4":POKE 764,255:
      GOTO 3030
 3056 IF X(PN-1)<>0 THEN GOTO 3100
 3062 GOSUB 4000
 3063 POKE 87,0:POKE 88,P883:POKE 89,
      P893
 3070 POSITION 0,0:? #6;"P# ";PN-1
 3080 GOSUB 60:GOTO 3198
 3100 REM PUT IT UP
 3110 POSITION 0,0:? #6;"P# ";PN-1
 3120 IF PN=OPN THEN RETURN
 3130 MX$="":NUM=0
 3140 GOSUB 60:GOSUB 90:GOSUB 4000
 3198 X=1:Y=0:XX=0:YY=0:D=255
 3199 RETURN
 4000 REM CLEAR MESSAGE LINES
```

S.M.

```
4002 POKE 752,1:POKE 87,0:POKE 88,P883:
     POKE 89,P893
4004 POSITION 0,12:? #6;"
                         "
4010 POSITION 0,10:? #6;"
                         "
4020 POSITION 0,11:? #6;"
                         ":POKE
     764,255
4050 REM DISPLAY OPTION-SELECT-START
4052 POKE 752,1:POKE 87,0:POKE 88,P883:
     POKE 89,P893:POSITION 5,10:?
     #6;"PRESS OPTION TO CHANGE #
     "
4054 POSITION 5,11:? #6;"PRESS SELECT
     FOR SIZE
                "
4056 POSITION 5,12:? #6;"PRESS START
     FOR P/M
4057 POKE 87,5:POKE 88,P882:POKE 89,
     P892
4059 RETURN
4060 REM DISPLAY OPTION
4100 REM OUTPUT DATA TO TAPE AND DISK
4102 POKE 87,0:POKE 88,P883:POKE 89,
     P893
4104 POSITION 0,10:? #6;"OUTPUT TO
     TAPE (T) OR DISK (D)?
4110 POSITION 0,11:? #6;"
                         "
4120 POSITION 0,12:? #6;"
                         ":POKE
     764,255
4130 D=PEEK(764)
4140 IF D=45 THEN GOTO 4200:REM TAPE
4150 IF D=58 THEN GOTO 4300:REM DISK
4160 GOTO 4130
4200 REM WRITE DATA TO TAPE
4220 FOR I=1 TO 128:P$(LEN(P$)+1)=" ":
     NEXT I
4224 MC$(1,20)=M0$:MC$(21,40)=M1$:
     MC$(41,60)=M2$:MC$(61,80)=M3$:
     MC$(81,100)=M4$
4230 TRAP OPN1:OPEN #1,8,0,"C:"
4240 TRAP RAW:? #1;P$:P$=""
4242 ? #1;MC$
4243 TRAP CLS1:CLOSE #1
4244 TRAP MLB:MC$(1)="(,)":MC$(100)="
     (,)":MC$(2)=MC$(1)
4248 GOSUB 4000
4249 RETURN
4300 REM SAVE TO DISK
4320 MC$(1,20)=M0$:MC$(21,40)=M1$:
     MC$(41,60)=M2$:MC$(61,80)=M3$:
     MC$(81,100)=M4$
4330 TRAP OPN1:OPEN #1,8,0,"D:
     PMISDATA"
4340 TRAP RAW:? #1;MC$:TRAP CLS1:CLOSE
     #1
4344 TRAP MLB:MC$(1)="(,)":MC$(100)="
     (,)":MC$(2)=MC$(1)
4346 GOSUB 4000
4349 RETURN
4600 REM INPUT FROM DISK
4604 TRAP OPN1:OPEN #1,4,0,"D:
     PMISDATA"
4610 TRAP RAW:INPUT #1,MC$:TRAP CLS1:
     CLOSE #1
4620 TRAP MLB:M0$=MC$(1,20):M1$=MC$(21,
     40):M2$=MC$(41,60):M3$=MC$(61,80):
     M4$=MC$(81,100)
4621 FOR I=0 TO 4:X(I)=0:NEXT I
4622 FOR I=1 TO 20
```

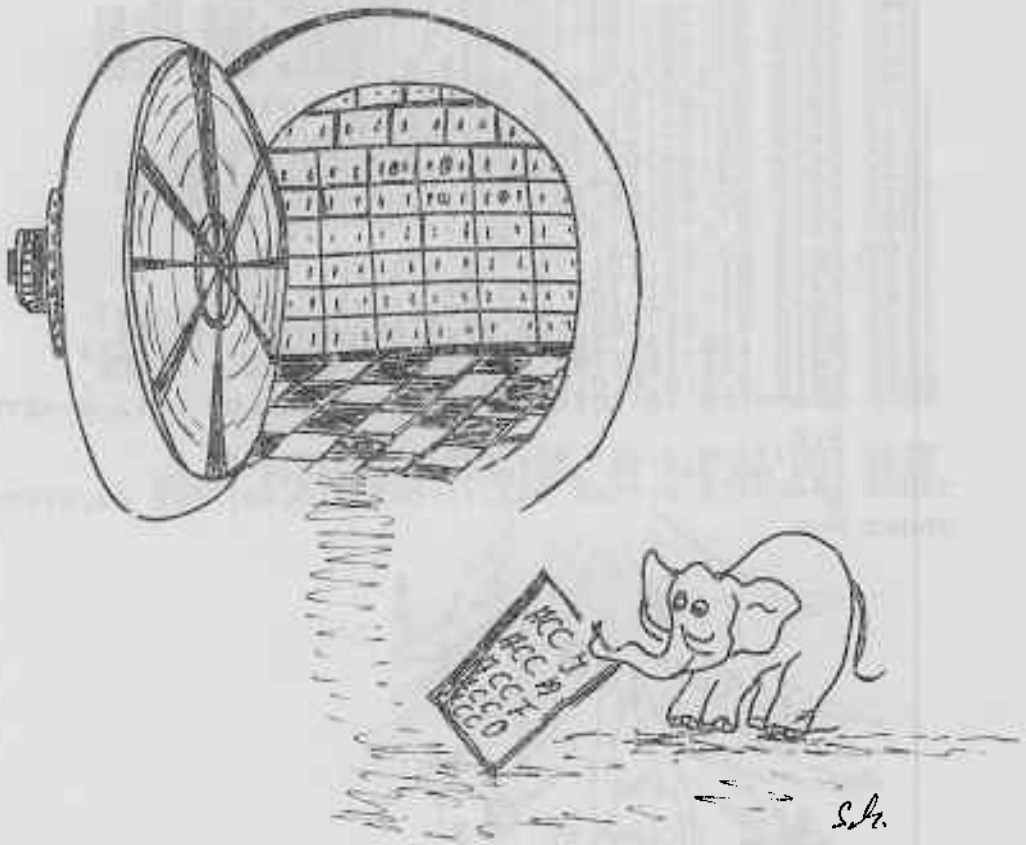


```
4623 IF MO$(I,I)<>"{,}" THEN X(0)=255
4624 IF M1$(I,I)<>"{,}" THEN X(1)=255
4625 IF M2$(I,I)<>"{,}" THEN X(2)=255
4626 IF M3$(I,I)<>"{,}" THEN X(3)=255
4627 IF M4$(I,I)<>"{,}" THEN X(4)=255
4628 NEXT I
4630 GOSUB 3000:OPN=255
4670 RETURN
7000 REM CLEAR MESSAGE LINES
7002 POKE 87,0:POKE 88,P883:POKE 89,
     P893
7004 POSITION 0,10:? #6;"
                         "
7006 POSITION 0,11:? #6;"
                         "
7008 POSITION 0,12:? #6;"
                         ":POKE
     764,255
7009 RETURN
7010 REM TRAP FOR NON I/O
7012 GRAPHICS 0:? "I'M TOTALLY
     CONFUSED.  TURN ME OFF...TAKE A
     BREAK...AND...START OVER."
7013 ? "FAREWELL FOR NOW..."
7014 ? "ON THE OTHER HAND, I'M WILLING
     TO TRY IT AGAIN."
7015 ? " THAT IS IF YOU DON'T DO WHAT
     YOU JUST DID TO ME."
7039 FOR I=0 TO 1000:NEXT I:GOTO 8
7040 REM TRAP FOR OPEN ERROR
7050 OPC=OPC+1:CLOSE #1
7052 IF OPC>5 THEN 7010
7054 POKE 87,0:POKE 88,P883:POKE 89,
     P893
7055 POSITION 5,10:? #6;"I CAN'T OPEN
     FOR YOUR I/O"
7056 POSITION 5,11:? #6;"PLEASE CHECK
     YOUR I/O DEVICE"
7057 POSITION 5,12:? #6;"IF OPEN FOR
     READ, IS THERE DATA?"
7058 FOR I=0 TO 1500:NEXT I
7059 GOSUB 3000:GOSUB 4000:RETURN
7070 REM TRAP FOR READ/WRITE ERROR
7080 RWC=RWC+1:CLOSE #1
7082 IF RWC>5 THEN 7010
7084 POKE 87,0:POKE 88,P883:POKE 89,
     P893
7085 POSITION 5,10:? #6;"I CAN'T
     EXECUTE THAT I/O"
7086 POSITION 5,11:? #6;"YOU'D BETTER
     LOOK AT YOUR I/O DEVICE"
7087 POSITION 5,11:? #6;"YOU'D BETTER
     LOOK AT YOUR I/O DEVICE"
7088 FOR I=0 TO 500:NEXT I
7089 GOSUB 3000:GOSUB 4000:RETURN
7090 REM TRAP FOR CLOSE ERROR
7091 CLC=CLC+1
7092 IF CLC>5 THEN 7010
7094 POKE 87,0:POKE 88,P883:POKE 89,
     P893
7095 POSITION 5,10:? #6;"I JUST TRIED
     TO CLOSE AND COULDN'T "
7096 POSITION 5,11:? #6;"YOU'D BETTER
     LOOK AT YOUR I/O DEVICE"
7097 FOR I=0 TO 500:NEXT I
7098 GOSUB 3000:GOSUB 4000:RETURN
8000 REM DEBUG TOOL FOR I/O
8010 GRAPHICS 0
8020 ? MC$:? MO$:? M1$;? M2$;? M3$;?
     M4$
8030 ? X(0),X(1),X(2),X(3),X(4)
8040 STOP

20080 DATA 112,112,112,66,64,156,2,10,
      10,10,10,10,10,10,10,10,10,10,10,
      10,10,10,10,10,10,10,2,2,2,2,2,
      2,2
20090 DATA 2,2,2,2,2,2,65,32,124,0,0,0
30000 DATA 112,112,112,66,64,156,2,10,
      10,10,10,10,10,10,10,10,10,10,10,
      10,10,10,10,10,10,10,2,2,2,2,2,
      2,2
30001 DATA 2,2,2,2,2,2,65,32,124,0,0,0
30010 DATA 128,64,32,16,8,4,2,1
```



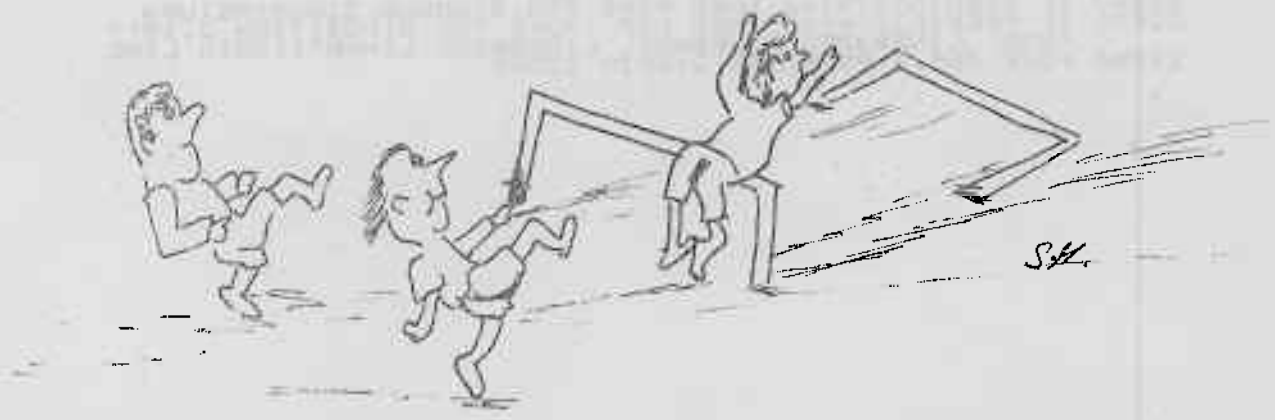
SOUND

```
  1 REM BEISPIEL 1
  2 REEM
  3 REEM
  4 REEM
100 P=100:D=10:V=2:GRAPHICS 17
120 ? #6:? #6;"   [THE ]sound[ COMMAND]"
125 ? #6:? #6;"       screen 9"
130 ? #6:? #6;"      CT=CTRL KEY"
140 ? #6:? #6;"    [p=increase pitch]"
160 ? #6:? #6;"CT+[p=decrease pitch]"
180 ? #6:? #6;"    [d=increase dist.]"
200 ? #6:? #6;"CT+[d=decrease dist.]"
220 ? #6:? #6;"    [v=increase vol.]"
240 ? #6:? #6;"CT+[v=decrease vol.]"
260 ? #6:? #6;"    [space=no sound]"
280 ? #6:? #6;"     [e=end of job]"
300 POSITION 2,22:? #6;"sound 0,";P;",";D;",";V;"    ":
    SOUND 0,P,D,V:GOTO 400
340 POSITION 2,22:? #6;"sound 0,0,0,0       ":SOUND 0,0,0,
    0
400 POKE 764,255
420 IF PEEK(764)=10 AND P<255 THEN P=P+1:GOTO 320
440 IF PEEK(764)=138 AND P>0 THEN P=P-1:GOTO 320
460 IF PEEK(764)=16 AND V<15 THEN V=V+1:GOTO 320
480 IF PEEK(764)=144 AND V>0 THEN V=V-1:GOTO 320
500 IF PEEK(764)=58 AND D<14 THEN D=D+2:GOTO 320
520 IF PEEK(764)=186 AND D>0 THEN D=D-2:GOTO 320
540 IF PEEK(764)=33 THEN 340
560 IF PEEK(764)=42 OR PEEK(764)=28 THEN RUN
580 GOTO 420
```



```
  1 REM BEISPIEL2
 20 GRAPHICS 18:SETCOLOR 2,4,8:SETCOLOR 0,12,6:SETCOLOR 4,
    4,0
 40 DIM WORD$(20),DISP$(20),BLANK$(20):BLANK$="
            ":GOTO 300
100 POKE 540,HOLD:SOUND 0,V0,10,8:SOUND 1,V1,10,4:SOUND 2,
    V2,10,4:SOUND 3,V3,10,4
101 IF PEEK(540)<>0 THEN 101
102 SOUND 0,0,0,0:SOUND 1,0,0,0:SOUND 2,0,0,0:SOUND 3,0,0,
    0:RETURN
300 READ V0,V1,V2,V3,HOLD,SWITCH,WORD$
330 IF SWITCH=2 THEN WORD$=DISP$
340 IF SWITCH=9 THEN 9000
360 LW=LEN(WORD$):DISP$=BLANK$:IF LW>18 THEN DISP$=WORD$:
    GOTO 380
370 FP=11-INT(LW/2):DISP$(FP,FP+LW-1)=WORD$
380 POSITION 0,4:? #6;DISP$:GOSUB 100:GOTO 300
600 DATA 81,96,121,162,89,1,SILENT NIGHT]
610 DATA 72,72,144,144,29,0,SILENT NIGHT]
620 DATA 81,96,121,162,59,0,SILENT [NIGHT]
630 DATA 96,121,81,193,119,0,SILENT NIGHT
640 DATA 0,0,0,0,59,2,REST
650 DATA 81,96,121,162,89,1,HOLY NIGHT]
660 DATA 72,72,144,144,29,0,HOLY NIGHT]
670 DATA 81,96,121,162,59,0,HOLY [NIGHT]
680 DATA 96,121,81,193,119,0,HOLY NIGHT
690 DATA 0,0,0,0,59,2,REST
700 DATA 53,64,81,108,119,1,ALL [IS CALM]
710 DATA 53,64,81,108,59,0,ALL IS [CALM]
720 DATA 64,81,108,128,119,0,ALL IS CALM
730 DATA 0,0,0,0,59,2,REST
740 DATA 60,81,96,121,119,0,ALL [IS BRIGHT]
750 DATA 60,81,96,121,59,0,ALL IS [BRIGHT]
760 DATA 81,96,121,162,119,0,ALL IS BRIGHT
770 DATA 0,0,0,0,59,2,REST
800 DATA 72,91,121,144,119,1,ROUND [YON]
810 DATA 72,91,121,144,59,0,ROUND YON
820 DATA 60,72,91,121,89,1,VIR[GIN]
830 DATA 64,64,128,128,29,0,VIR[GIN]
840 DATA 72,71,121,144,59,0,VIRGIN
850 DATA 81,96,121,162,89,1,MO[THER AND CHILD]
860 DATA 72,72,144,144,29,0,MOTHER [AND CHILD]
870 DATA 81,96,121,162,59,0,MOTHER AND [CHILD]
880 DATA 96,121,81,193,119,0,MOTHER AND CHILD
890 DATA 0,0,0,0,59,2,REST
900 DATA 72,91,121,144,119,1,HO[LY INFANT SO]
910 DATA 72,91,121,144,59,0,HOLY [INFANT SO]
920 DATA 60,72,91,121,89,0,HOLY INFANT SO]
930 DATA 64,64,128,128,29,0,HOLY INFANT [SO]
940 DATA 72,71,121,144,59,0,HOLY INFANT SO
950 DATA 81,96,121,162,89,1,TEN[DER AND MILD]
960 DATA 72,72,144,144,29,0,TENDER [AND MILD]
970 DATA 81,96,121,162,59,0,TENDER AND [MILD]
980 DATA 96,121,81,193,119,0,TENDER AND MILD
990 DATA 0,0,0,0,59,2,REST
1000 DATA 53,64,81,108,119,1,SLEEP [IN]
1010 DATA 53,64,81,108,59,0,SLEEP IN
1020 DATA 45,53,64,81,89,1,HEA[VENLY PEACE]
1030 DATA 53,64,81,108,29,0,HEAVEN[LY PEACE]
1040 DATA 64,81,108,128,59,0,HEAVENLY [PEACE]
1050 DATA 60,81,96,121,179,0,HEAVENLY PEACE
1060 DATA 47,60,81,96,119,0,HEAVENLY PEACE
1070 DATA 0,0,0,0,59,2,REST
1100 DATA 60,81,96,121,59,1,SLEEP [IN]
1110 DATA 81,96,121,162,59,0,SLEEP [IN]
1120 DATA 96,121,162,193,59,0,SLEEP IN
1130 DATA 81,108,126,162,89,1,HEA[VENLY PEACE]
1140 DATA 91,91,182,182,29,0,HEAVENLY PEACE]
1150 DATA 108,128,182,217,59,0,HEAVENLY [PEACE]
1160 DATA 121,162,193,243,179,0,HEAVENLY PEACE
8000 DATA 0,0,0,0,0,9,END
9000 GRAPHICS 18:SETCOLOR 2,4,8:SETCOLOR 0,12,6:SETCOLOR 4,
    4,0
9100 POSITION 3,4:? #6;"MERRY CHRISTMAS"
9200 FOR HOLD=1 TO 500:NEXT HOLD:GOTO 30000
30000 GRAPHICS 0:POKE 752,1:POKE 710,48:POKE 82,2:POKE 201,
    8
30005 RUN
```

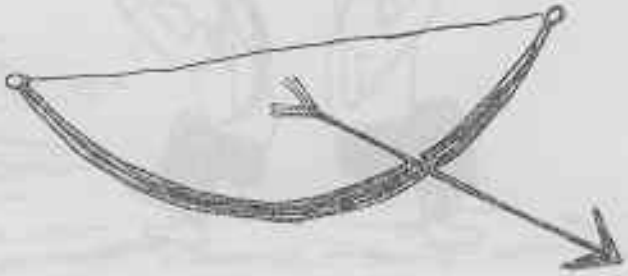
```
  1 REM BEISPIEL 2
  2 REM
  3 REM
  4 REM
 10 DIM TYPE$(40),N(50):GOTO 100
 50 POKE 540,WAIT
 52 IF PEEK(540)<>0 THEN 52
 54 RETURN
 60 FOR OFF=0 TO 3:SOUND OFF,0,0,0:NEXT OFF:RETURN
 70 LINE=LEN(TYPE$):POSITION HOR,VER
 72 FOR ME=1 TO LINE:? TYPE$(ME,ME);:IF TYPE$(ME,ME)=" "
    THEN 76
 74 REM SOUND 0,25,4,6:FOR DECAY=6 TO 0 STEP -0.5:SOUND 0,
    10,0,DECAY:NEXT DECAYECAY
 76 NEXT ME:RETURN
100 DATA 14,15,16,17,18,19,21,22,23,24,26,27,29,31,33,35,
    37,40,42,45,47,50,53,57,60,64,68,72,76,81,85,91,96
110 DATA 102,108,114,121,128,136,144,153,162,173,182,193,
    204,217,230,243,255
120 PAGE=9:GOSUB 21000:FOR X=1 TO 50:READ IT:N(X)=IT:NEXT
    X
130 TYPE$="Although there are pitches":HOR=6:VER=5:GOSUB
    70
140 TYPE$="numbered from 0 thru 255, we need":HOR=3:VER=7:
    GOSUB 70
150 TYPE$="only the ones that correspond to":HOR=3:VER=9:
    GOSUB 70
160 TYPE$="musical notes to write music.":HOR=5:VER=11:
    GOSUB 70:WAIT=60:GOSUB 50
170 TYPE$="I put these specific pitches":HOR=5:VER=13:
    GOSUB 70
180 TYPE$="into an array as shown in the":HOR=5:VER=15:
    GOSUB 70
190 TYPE$="printed PITCH CHART.":HOR=9:VER=17:GOSUB 70
200 GOSUB 20000:GOSUB 21000
300 LIST 310
310 FOR P=1 TO 50:FOR V=15 TO 0 STEP -3:SOUND 0,N(P),10,V:
    NEXT V:NEXT P
320 IF LINE<>310 THEN ? :LIST 330
330 FOR P=50 TO 1 STEP -1:FOR V=15 TO 0 STEP -3:SOUND 0,
    N(P),10,V:NEXT V:NEXT P
335 IF LINE=310 THEN 400
340 TYPE$="We can now access these notes":HOR=5:VER=13:
    GOSUB 70
350 TYPE$="using a 50 element array as shown":HOR=3:VER=
    15:GOSUB 70
360 TYPE$="in the FOR NEXT loops above.":HOR=5:VER=17:
    GOSUB 70
400 LINE=310:GOSUB 22000
410 TYPE$="We also need some way of telling":HOR=5:VER=5:
    GOSUB 70
420 TYPE$="BASIC how long to hold onto a note.":HOR=3:VER=
    7:GOSUB 70:WAIT=60:GOSUB 50
430 TYPE$="One way is to use one of the counters":HOR=2:
    VER=9:GOSUB 70
440 TYPE$="built into your computer.":HOR=9:VER=11:GOSUB
    70:GOSUB 50
450 TYPE$="RAM location 540 contains a counter":HOR=3:VER=
    13:GOSUB 70
460 TYPE$="which counts backwards at the rate":HOR=3:VER=
    15:GOSUB 70
470 TYPE$="of 60 per second.":HOR=12:VER=17:GOSUB 70
500 GOSUB 20000:GOSUB 21000
510 TYPE$="1/60 of a second is called a JIFFY.":HOR=3:VER=
    5:GOSUB 70:GOSUB 50
520 TYPE$="We can poke any number of jiffies":HOR=4:VER=7:
    GOSUB 70
530 TYPE$="(up to 255) into location 540,":HOR=5:VER=9:
    GOSUB 70:GOSUB 50
540 TYPE$="then loop there until that location":HOR=3:VER=
    11:GOSUB 70
550 TYPE$="contains a zero.":HOR=12:VER=13:GOSUB 70:GOSUB
```



```
    50
  600 GOSUB 20000:GOSUB 21000
  610 TYPE$="This is how we turn on a note for":HOR=4:VER=5:
      GOSUB 70
  620 TYPE$="one second.":HOR=15:VER=7:GOSUB 70:GOSUB 50
  630 TYPE$="(1) Turn on a SOUND":HOR=3:VER=9:GOSUB 70:
      GOSUB 50
  640 TYPE$="(2) POKE 540,60":HOR=3:VER=11:GOSUB 70:GOSUB
      50
  650 TYPE$="(3) IF PEEK(540)<>0 THEN PEEK AGAIN":HOR=3:VER=
      13:GOSUB 70:GOSUB 50
  660 TYPE$="(4) LOCATION 540=0 SO TURN OFF SOUND":HOR=3:
      VER=15:GOSUB 70:GOSUB 50
  700 GOSUB 20000:GOSUB 21000
  710 TYPE$="Since this will be done often,":HOR=6:VER=5:
      GOSUB 70:GOSUB 50
  720 TYPE$="we will use a WAIT subroutine.":VER=7:GOSUB 70:
      GOSUB 50
  730 TYPE$="For example, this program uses":VER=9:GOSUB 70
  740 TYPE$="the variable WAIT as the number":VER=11:GOSUB
      70
  750 TYPE$="of jiffies to POKE into location 540.":HOR=2:
      VER=13:GOSUB 70:GOSUB 50
  760 TYPE$="We set WAIT=60 to wait one second":HOR=4:VER=
      15:GOSUB 70:GOSUB 50
  770 TYPE$="then GOSUB 50.":HOR=13:VER=17:GOSUB 70
  800 GOSUB 20000:GOSUB 21000
  810 TYPE$="This is our wait subroutine:":HOR=7:VER=5:
      GOSUB 70:GOSUB 50
  820 POKE 82,7:? " ":LIST 50,54:POKE 82,2:? " ":WAIT=120:
      GOSUB 50
  830 TYPE$="And here is a subroutine to turn":HOR=4:VER=12:
      GOSUB 70
  840 TYPE$="off all SOUNDS:":HOR=12:VER=14:GOSUB 70:? :
      LIST 60
  900 GOSUB 20000:GOSUB 21000
  910 TYPE$="In order to write a song in Basic,":HOR=4:VER=
      5:GOSUB 70:WAIT=60:GOSUB 50
  920 TYPE$="we can store the NOTE numbers":HOR=6:VER=7:
      GOSUB 70
  930 TYPE$="and durations in DATA statements.":HOR=4:VER=9:
      GOSUB 70:GOSUB 50
  940 TYPE$="We read the data, then let our":HOR=6:VER=11:
      GOSUB 70
  950 TYPE$="subroutines play the music as":HOR=6:VER=13:
      GOSUB 70
  960 TYPE$="demonstrated by later programs.":HOR=4:VER=15:
      GOSUB 70
 1000 POKE 201,10:? "@PFEIL OBEN@";:GOSUB 20000:RUN "D:
      MAJRMINR"
20000 POSITION 2,20:POKE 201,5:POKE 752,1
20010 ? ;"(0)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)
      (R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(E)"
20020 ? ;"| Press [START] to continue |"
20030 ? ;"(Z)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)
      (R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(C)"
20040 IF PEEK(53279)<>6 THEN POKE 755,3:POKE 755,2:GOTO
      20040
20050 POKE 755,2:RETURN
21000 GRAPHICS 0:SETCOLOR 2,9,0:SETCOLOR 4,9,0:SETCOLOR 1,9,
      12:POKE 752,1:POKE 82,2:POKE 83,39:POKE 201,11
21005 PAGE=PAGE+1
21010 ? ;"(0)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(E)"
21020 ? ;"|| BASIC MUSIC ||"
21025 ? ;"|  screen #";PAGE;"  |"
21030 ? ;"(Z)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(C)"
21040 POKE 77,0:RETURN
22000 POSITION 2,19:POKE 201,5:POKE 752,1
22010 ? ;"(0)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)
      (R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(E)"
22015 ? ;"| Press [OPTION] to repeat  |"
22020 ? ;"| Press [START] to continue |"
22030 ? ;"(Z)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)
      (R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(C)"
22040 IF PEEK(53279)=6 THEN POKE 755,2:GOSUB 21000:RETURN
22050 IF PEEK(53279)=3 THEN POP :POKE 755,2:POSITION 2,19:?
      "@DEL LINE@@DEL LINE@@DEL LINE@@DEL LINE@";:GOTO LINE
22060 POKE 755,3:POKE 755,2:GOTO 22040
```

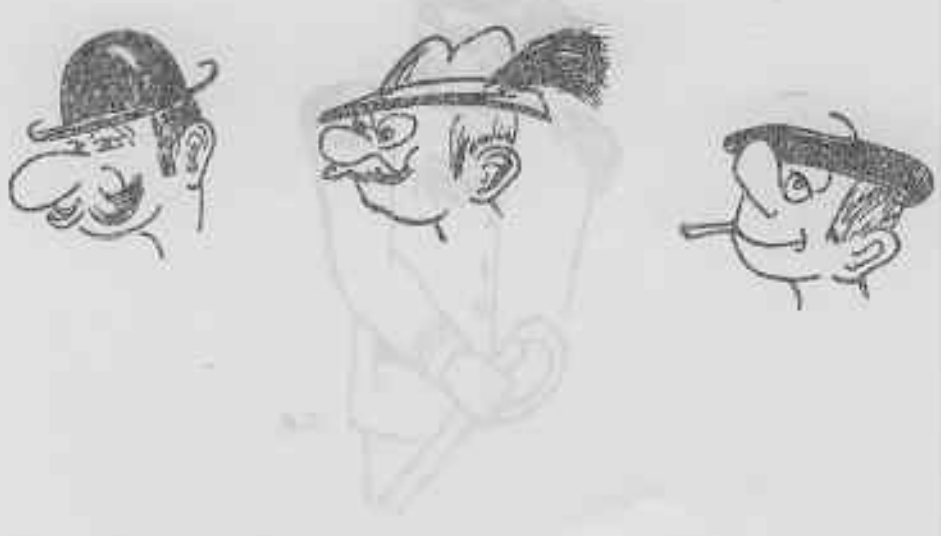
```
  1 REM BEISPIEL 4
  2 REM
  3 REM
  4 REM
 10 DIM TYPE$(40),N(50):V0=0:V1=1:V2=2:V3=3:POKE 82,2:? "@PFEIL
    ";?"@PFEIL OBEN@":GOTO 100
 50 POKE 540,WAIT
 52 IF PEEK(540)<>0 THEN 52
 54 RETURN
 60 FOR OFF=0 TO 3:SOUND OFF,0,0,0:NEXT OFF:RETURN
 70 LINE=LEN(TYPE$):POSITION HOR,VER
 72 FOR ME=1 TO LINE:? TYPE$(ME,ME);:IF TYPE$(ME,ME)=" "
    THEN 76
 74 REM SOUND 0,25,4,6:FOR DECAY=6 TO 0 STEP -0.5:SOUND 0,
    10,0,DECAY:NEXT DECAY
 76 NEXT ME:RETURN
100 DATA 14,15,16,17,18,19,21,22,23,24,26,27,29,31,33,35,
    37,40,42,45,47,50,53,57,60,64,68,72,76,81,85,91,96
110 DATA 102,108,114,121,128,136,144,153,162,173,182,193,
    204,217,230,243,255
120 PAGE=17:GOSUB 21000:FOR X=1 TO 50:READ IT:N(X)=IT:
    NEXT X
130 TYPE$="Using the array method makes it":HOR=5:VER=6:
    GOSUB 70
140 TYPE$="possible to find the notes of any":HOR=4:VER=8:
    GOSUB 70
150 TYPE$="CHORD with a BASIC subroutine.":HOR=6:VER=10:
    GOSUB 70:WAIT=60:GOSUB 50
160 TYPE$="This way, we can create a 3 or 4":HOR=4:VER=12:
    GOSUB 70
170 TYPE$="note chord by entering only one":HOR=4:VER=14:
    GOSUB 70
180 TYPE$="number or the ROOT of the chord.":HOR=4:VER=16:
    GOSUB 70:GOSUB 20000:GOSUB 21000
190 TYPE$="MAJOR CHORDS:":HOR=14:VER=6:GOSUB 70:GOSUB 50:
    ? :? :LIST 200,210
200 FOR X=13 TO 50:SOUND 0,N(X),10,14:SOUND 1,N(X-4),10,6:
    SOUND 2,N(X-7),10,6:SOUND 3,N(X-12),10,6
210 GOSUB 60:NEXT X
215 IF LINE<>200 THEN ? :LIST 220,230
220 FOR X=50 TO 13 STEP -1:SOUND 0,N(X),10,14:SOUND 1,N(X-
    4),10,6:SOUND 2,N(X-7),10,6:SOUND 3,N(X-12),10,6
230 GOSUB 60:NEXT X
240 LINE=200:GOSUB 22000
250 TYPE$="MINOR CHORDS:":HOR=14:VER=6:GOSUB 70:GOSUB 50:
    ? :LIST 260,270
260 FOR X=13 TO 50:SOUND 0,N(X),10,14:SOUND 1,N(X-3),10,6:
    SOUND 2,N(X-7),10,6:SOUND 3,N(X-12),10,6
270 GOSUB 60:NEXT X
280 IF LINE<>260 THEN ? :LIST 290,300
290 FOR X=50 TO 13 STEP -1:SOUND 0,N(X),10,14:SOUND 1,N(X-
    3),10,6:SOUND 2,N(X-7),10,6:SOUND 3,N(X-12),10,6
300 GOSUB 60:NEXT X
310 LINE=260:GOSUB 22000
320 X=49
330 TYPE$="C MAJOR":HOR=17:VER=6:GOSUB 70:GOSUB 50:? :
    LIST 340:?
340 X=49:SOUND 0,N(X),10,14:SOUND 1,N(X-4),10,6:SOUND 2,
    N(X-7),10,6:SOUND 3,N(X-12),10,6
350 WAIT=60:GOSUB 50:GOSUB 60
360 IF LINE<>340 THEN TYPE$="C MINOR":HOR=17:VER=12:GOSUB
    70:GOSUB 50:? :LIST 370:?
370 X=49:SOUND 0,N(X),10,14:SOUND 1,N(X-3),10,6:SOUND 2,
    N(X-7),10,6:SOUND 3,N(X-12),10,6
380 WAIT=60:GOSUB 50:GOSUB 60
390 LINE=340:GOSUB 22000
400 TYPE$="Let's go back to standard SOUND":HOR=5:VER=6:
    GOSUB 70
410 TYPE$="commands now to make a comparison.":HOR=3:VER=
    9:GOSUB 70:GOSUB 50
420 TYPE$="The lowest note we can get using":HOR=4:VER=10:
    GOSUB 70
```

```
  430 TYPE$="DISTORTION 10 is pitch 255, a low B.":HOR=2:
      VER=12:GOSUB 70:GOSUB 50
  440 TYPE$="SOUND V0,255,10,8":HOR=12:VER=14:GOSUB 70:
      GOSUB 60:SOUND V0,255,10,8:GOSUB 50:GOSUB 60
  450 GOSUB 20000:GOSUB 21000
  500 TYPE$="Now compare these two sounds...":HOR=5:VER=6:
      GOSUB 70:GOSUB 50
  510 TYPE$="SOUND V0,255,10,8":HOR=12:VER=8:GOSUB 70:GOSUB
      50
  520 SOUND V0,255,10,8:GOSUB 50:GOSUB 60:GOSUB 50
  530 TYPE$="SOUND V0,33,12,8":HOR=12:VER=10:GOSUB 70:GOSUB
      50
  540 SOUND V0,33,12,8:GOSUB 50:GOSUB 60:GOSUB 50
  550 TYPE$="The tone quality is different but":HOR=4:VER=
      12:GOSUB 70
  560 TYPE$="the actual PITCH is the same low B.":HOR=3:VER=
      14:GOSUB 70:GOSUB 50
  570 POSITION 12,8:? "[SOUND V0,255,10,8]":POSITION 12,10:
      ? "SOUND V0,33,12,8":SOUND V0,255,10,8
  580 GOSUB 50:GOSUB 60:GOSUB 50:POSITION 12,8:? "SOUND V0,
      255,10,8"
  590 POSITION 12,10:? "[SOUND V0,33,12,8]":SOUND V0,33,12,
      8:GOSUB 50:GOSUB 60:GOSUB 50
  600 POSITION 12,8:? "[SOUND V0,255,10,8]":SOUND V0,255,10,
      8:POSITION 12,10:? "[SOUND V1,33,12,8]":SOUND V1,33,
      12,8
  610 GOSUB 50:GOSUB 60:LINE=570:GOSUB 22000
  620 TYPE$="We can get much deeper BASS NOTES":HOR=3:VER=6:
      GOSUB 70
  630 TYPE$="in DISTORTION 12 than in 10.":HOR=5:VER=8:
      GOSUB 70:GOSUB 50
  640 TYPE$="FOR EXAMPLE:":HOR=14:VER=10:GOSUB 70:GOSUB 50
  700 DATA 25,27,28,30,31,33,36,37,40,42,45,48,51,52,57,60,
      63,67,72,75,82,85,90,97,102
  710 FOR BASS=1 TO 25:READ PO:FOR DECAY=15 TO 0 STEP -1
  720 POSITION 12,12:? "SOUND V0,";PO;",12,";DECAY;" ":
      SOUND V0,PO,12,DECAY:NEXT DECAY:NEXT BASS
  750 RESTORE 700:LINE=710:GOSUB 22000
  800 TYPE$="The next program will demonstrate":HOR=3:VER=6:
      GOSUB 70
  910 TYPE$="a complete song using these":HOR=3:VER=8:GOSUB
      70
  820 TYPE$="methods. Later you will be given":HOR=3:VER=10:
      GOSUB 70
  830 TYPE$="a small subroutine to add your own ":HOR=3:VER=
      12:GOSUB 70
  835 TYPE$="words and music to. ":HOR=3:VER=14:GOSUB 70:
      GOSUB 20000
  840 RUN "D:KEYOFC"
20000 POSITION 2,20:POKE 201,5:POKE 752,1
20010 ? #6;"{Q}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}
      {R}{R}{R}{R}{R}{R}{R}{R}{R}{E}"
20020 ? #6;"! Press [START] to continue !"
20030 ? #6;"{Z}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}
      {R}{R}{R}{R}{R}{R}{R}{R}{R}{C}"
20040 IF PEEK(53279)<>6 THEN POKE 755,3:POKE 755,2:GOTO
      20040
20050 POKE 755,2:RETURN
21000 GRAPHICS 0:SETCOLOR 2,9,0:SETCOLOR 4,9,0:SETCOLOR 1,9,
      12:POKE 752,1:POKE 82,2:POKE 83,39:POKE 201,8
21005 PAGE=PAGE+1
21010 ? #6;"{Q}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}
      {R}{R}{R}{R}{E}"
21020 ? #6;"! The [SOUND] command !"
21025 ? #6;"!      screen #";PAGE;"      !"
21030 ? #6;"{Z}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}
      {R}{R}{R}{R}{R}{R}{R}{R}{R}{C}"
21040 POKE 77,0:RETURN
22000 POSITION 2,19:POKE 201,5:POKE 752,1
22010 ? #6;"{Q}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}
      {R}{R}{R}{R}{R}{R}{R}{R}{R}{E}"
22015 ? #6;"! Press [OPTION] to repeat  !"
22020 ? #6;"! Press [START] to continue !"
22030 ? #6;"{Z}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}
      {R}{R}{R}{R}{R}{R}{R}{R}{R}{C}"
22040 IF PEEK(53279)=6 THEN POKE 755,2:GOSUB 21000:RETURN
22050 IF PEEK(53279)=3 THEN POP :POKE 755,2:POSITION 2,19:?
      "@DEL LINE@@DEL LINE@@DEL LINE@@DEL LINE@";:GOTO LINE
22060 POKE 755,3:POKE 755,2:GOTO 22040
```



```
  1 REM BEISPIEL 5
  2 REM
  3 REM
  4 REM
120 POKE 82,5:GOTO 480
140 FOR WAIT=1 TO 200:NEXT WAIT:RETURN
160 COLOR 1:PLOT X+1,Y:DRAWTO X+2,Y:PLOT X,Y+1:DRAWTO X+3,
    Y+1
180 PLOT X-1,Y+2:DRAWTO X+4,Y+2:PLOT X-1,Y+3:DRAWTO X+4,Y+
    3
200 PLOT X,Y+4:DRAWTO X+3,Y+4:PLOT X+1,Y+5:DRAWTO X+2,Y+5
220 PLOT X+4,Y-10:DRAWTO X+4,Y+2:RETURN
240 FOR HOLD=1 TO 500:NEXT HOLD:FOR OFF=0 TO 3:SOUND OFF,
    0,0,0:NEXT OFF:RETURN
260 ? "     PRESS ANY KEY TO CONTINUE.";:POKE 764,255
280 IF PEEK(764)=255 AND PEEK(53279)=7 THEN 280
300 GOSUB 380:RETURN
320 ? "     PRESS ANY KEY TO CONTINUE.";:POKE 764,255
340 IF PEEK(764)=255 AND PEEK(53279)=7 THEN 340
360 RETURN
380 GRAPHICS 7:SETCOLOR 0,7,8:SETCOLOR 1,12,8:SETCOLOR 2,
    2,12:SETCOLOR 3,4,6:SETCOLOR 4,2,12
400 COLOR 2:PLOT 20,20:DRAWTO 140,20:PLOT 20,26:DRAWTO
    140,26:PLOT 20,32:DRAWTO 140,32
420 POKE 752,1:PLOT 20,38:DRAWTO 140,38:PLOT 20,44:DRAWTO
    140,44:RETURN
440 COLOR 1:PLOT X+3,Y+6:DRAWTO X+5,Y-1:PLOT X+7,Y+6:
    DRAWTO X+9,Y-1
460 PLOT X+2,Y+1:DRAWTO X+10,Y+1:PLOT X+2,Y+4:DRAWTO X+10,
    Y+4:RETURN
480 GOSUB 380:? :? "   THE C SCALE":? :? "C D E F G A B C"
500 X=40:Y=47:GOSUB 160:FOR VOL=10 TO 0 STEP -0.2:SOUND 0,
    121,10,VOL:NEXT VOL
520 COLOR 2:PLOT 33,50:DRAWTO 48,50:GOSUB 140:COLOR 1
540 X=50:Y=44:GOSUB 160:FOR VOL=10 TO 0 STEP -0.2:SOUND 0,
    108,10,VOL:NEXT VOL:GOSUB 140
560 X=60:Y=41:GOSUB 160:FOR VOL=10 TO 0 STEP -0.2:SOUND 0,
    96,10,VOL:NEXT VOL:GOSUB 140
580 X=70:Y=38:GOSUB 160:FOR VOL=10 TO 0 STEP -0.2:SOUND 0,
    91,10,VOL:NEXT VOL:GOSUB 140
600 X=80:Y=35:GOSUB 160:FOR VOL=10 TO 0 STEP -0.2:SOUND 0,
    81,10,VOL:NEXT VOL:GOSUB 140
620 X=90:Y=32:GOSUB 160:FOR VOL=10 TO 0 STEP -0.2:SOUND 0,
    72,10,VOL:NEXT VOL:GOSUB 140
640 X=100:Y=29:GOSUB 160:FOR VOL=10 TO 0 STEP -0.2:SOUND
    0,64,10,VOL:NEXT VOL:GOSUB 140
660 X=110:Y=26:GOSUB 160:FOR VOL=10 TO 0 STEP -0.2:SOUND
    0,60,10,VOL:NEXT VOL:GOSUB 140
680 GOSUB 240
700 ? "THE C MAJOR CHORD":? "     C + E + G"
720 X=60:Y=47:GOSUB 160:FOR VOL=10 TO 0 STEP -0.2:SOUND 0,
    121,10,VOL:NEXT VOL
740 PLOT 55,50:DRAWTO 68,50
760 X=80:Y=41:GOSUB 160:FOR VOL=10 TO 0 STEP -0.2:SOUND 0,
    96,10,VOL:NEXT VOL
780 X=100:Y=35:GOSUB 160:FOR VOL=10 TO 0 STEP -0.2:SOUND
    0,81,10,VOL:NEXT VOL:GOSUB 320
800 ? :? :? "SOUND 0,121,10,8":? "SOUND 1,96,10,8":? ,
    "SOUND 2,81,10,8"
820 SOUND 0,121,10,8:SOUND 1,96,10,8:SOUND 2,81,10,8:
    GOSUB 240
840 GOSUB 320
```

```
  1 REM BEISPIEL 6
  2 REM
  3 Q0=0:Q1=1:Q2=2:Q3=3:Q4=4:Q5=5:Q6=0.2:Q8=8:Q10=10:Q12=
    12:Q13=13:Q16=16:Q17=17:Q19=19:Q20=20:Q22=22:Q23=23:
    Q26=26:Q29=29
  4 Q35=35:Q38=38:Q40=40:Q41=41:Q42=42:Q44=44:Q45=45:Q47=
    47:Q50=50:Q53=53:Q55=55:Q57=57:Q60=60:Q64=64:Q68=68:
    Q70=70:Q72=72
  5 Q76=76:Q80=80:Q81=81:Q85=85:Q91=91:Q95=95:Q96=96:Q102=
    102:Q108=108:Q114=114:Q121=121:Q122=122:Q130=130:Q138=
    138:Q154=1
  6 Q178=178:Q255=255:Q256=256:Q300=300:Q764=764:Q100=100:
    Q7=6
 20 DIM ZA$(Q100):GOTO 150
 40 ZL=PEEK(560)+PEEK(561)*Q256
 42 ZM=PEEK(ZL+Q4)+PEEK(ZL+Q5)*Q256:FOR ZW=Q1 TO LEN(ZA$):
    ZT=57344+((ASC(ZA$(ZW,ZW))-Q32)*Q8):ZC=ZM+ZY*Q40+ZX+
    ZW-Q1
 44 FOR ZR=Q0 TO 7:POKE ZC+ZR*Q40,PEEK(ZT+ZR):NEXT ZR:ZY=
    ZY+ZZ:NEXT ZW:RETURN
 50 COLOR Q1:PLOT X+Q1,Y:DRAWTO X+Q2,Y:PLOT X,Y+Q1:DRAWTO
    X+Q3,Y+Q1
 52 PLOT X-Q1,Y+Q2:DRAWTO X+Q4,Y+Q2:PLOT X-Q1,Y+Q3:DRAWTO
    X+Q4,Y+Q3
 53 PLOT X,Y+Q4:DRAWTO X+Q3,Y+Q4:PLOT X+Q1,Y+Q5:DRAWTO X+
    Q2,Y+Q5
 55 PLOT X+Q4,Y-Q10:DRAWTO X+Q4,Y+Q2:RETURN
 60 FOR HOLD=Q1 TO 500:NEXT HOLD:FOR OFF=Q0 TO Q3:SOUND
    OFF,Q0,Q0,Q0:NEXT OFF:RETURN
 70 ? "           PRESS ANY KEY TO CONTINUE.";:POKE Q764,Q255
 71 IF PEEK(Q764)=Q255 AND PEEK(53279)=7 THEN 71
 72 RETURN
 80 GRAPHICS Q8:SETCOLOR Q0,Q2,Q0:SETCOLOR Q1,Q2,Q0:
    SETCOLOR Q2,Q2,Q8:SETCOLOR Q4,Q2,Q8:POKE 752,Q1:COLOR
    Q1
 81 PLOT Q20,Q20:DRAWTO Q300,Q20:PLOT Q20,Q26:DRAWTO Q300,
    Q26:PLOT Q20,Q32:DRAWTO Q300,Q32:PLOT Q20,Q38:DRAWTO
    Q300,Q38
 85 PLOT Q20,Q44:DRAWTO Q300,Q44:ZX=Q13:ZY=125:ZA$="BASIC
    CHORDS":GOSUB Q40:RETURN
100 FOR VOL=Q10 TO Q0 STEP -Q6:SOUND Q0,P,Q10,VOL:NEXT
    VOL:RETURN
150 GRAPHICS Q17:SETCOLOR Q2,Q8,Q12:? #Q7:? #Q7:? #Q7;"
    [SELECT CHORD]":? #Q7;"          BY NUMBER":? #Q7
160 ? #Q7:? #Q7;"      1= [D  MAJOR]":? #Q7:? #Q7;"      2=
    [D# MAJOR]"
170 ? #Q7:? #Q7;"      3= [E  MAJOR]":? #Q7:? #Q7;"      4=
    [F  MAJOR]"
180 ? #Q7:? #Q7;"      5= [F# MAJOR]":? #Q7:? #Q7;"      6=
    [G  MAJOR]":? #Q7:? #Q7;"          7= [G# MAJOR]"
200 ? #Q7:? #Q7;"          8= [Q U I T]":P=Q10:GOSUB Q100
210 POKE Q764,Q255:CLOSE #Q1:OPEN #Q1,Q4,Q0,"K:":GET #Q1,
    GC:CLOSE #Q1:GC=GC-48
215 IF GC=Q8 THEN 30000
220 IF GC<Q1 OR GC>7 THEN 210
230 GOSUB Q80:ON GC GOSUB 1000,1300,1600,1900,2200,2500,
    2800
240 GOTO 150
1000 ZA$="THE D MAJOR CHORD":ZX=Q12:ZY=Q75:GOSUB Q40:? ,
    "500 SOUND 1,108,10,2":ZA$="D":ZX=Q16:ZY=Q55
1042 GOSUB Q40:P=Q108:GOSUB Q100:X=Q130:Y=Q44:GOSUB Q50:? ,
    "510 SOUND 1,85,10,2"
1110 ZA$="F#":ZX=Q19:ZY=Q55:GOSUB Q40:P=Q85:GOSUB Q100:X=
    Q154:Y=Q38:GOSUB Q50
1140 ZA$="A":ZX=Q20:ZY=Q38:GOSUB Q40:? ,"520 SOUND 2,72,10,
    2":ZA$="A":ZX=Q22:ZY=Q55:GOSUB Q40
1220 P=Q72:GOSUB Q100:X=Q178:Y=Q32:GOSUB Q50
1250 ZA$="D+F#+A=D MAJOR":ZX=Q13:ZY=Q95:GOSUB Q40
1260 SOUND Q0,Q108,Q10,Q2:SOUND Q1,Q85,Q10,Q2:SOUND Q2,Q72,
    Q10,Q2:GOSUB Q60:GOSUB Q70:RETURN
1300 ZA$="THE D# MAJOR CHORD":ZX=Q12:ZY=Q75:GOSUB Q40:? ,
    "500 SOUND 0,102,10,2":ZA$="D#":ZX=Q16:ZY=Q55:GOSUB
```



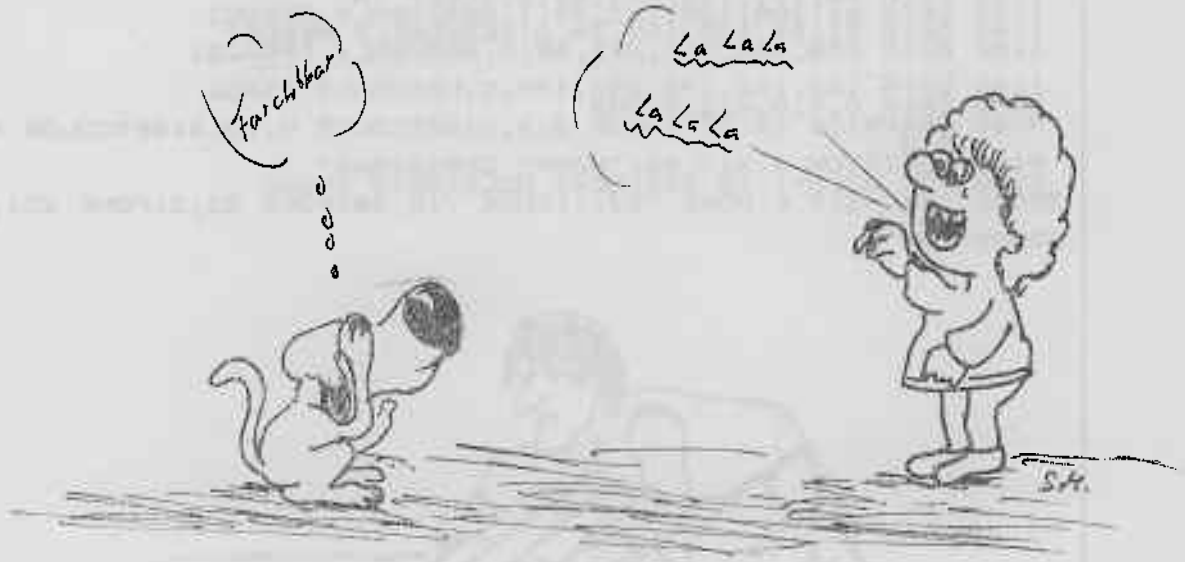
```
      Q40
1342  P=Q102:GOSUB Q100:X=Q130:Y=Q44:GOSUB Q50:ZA$="#":ZX=
      Q17:ZY=Q44:GOSUB Q40
1400  ? :? "510 SOUND 1,81,10,2":ZA$="G":ZX=Q19:ZY=Q55:GOSUB
      Q40:P=Q81:GOSUB Q100
1430  X=Q154:Y=Q35:GOSUB Q50:? :? "520 SOUND 2,68,10,2":ZA$=
      "A#":ZX=Q22:ZY=Q55:GOSUB Q40
1520  P=Q68:GOSUB Q100:X=Q178:Y=Q32:GOSUB Q50:ZA$="#":ZX=
      Q23:ZY=Q32:GOSUB Q40
1550  ZA$="D#+G+A#=D# MAJOR":ZX=Q13:ZY=Q95:GOSUB Q40
1560  SOUND Q0,Q102,Q10,Q2:SOUND Q1,Q81,Q10,Q2:SOUND Q2,Q68,
      Q10,Q2:GOSUB Q60:GOSUB Q70:RETURN
1600  ZA$="THE E MAJOR CHORD":ZX=Q12:ZY=Q75:GOSUB Q40:? :?
      "500 SOUND 0,96,10,2":ZA$="E":ZX=Q16:ZY=Q55:GOSUB Q40
1642  P=Q96:GOSUB Q100:X=Q130:Y=Q41:GOSUB Q50:? :? "510 SOUND
      1,76,10,2"
1710  ZA$="G#":ZX=Q19:ZY=Q55:GOSUB Q40:P=Q76:GOSUB Q100:X=
      Q154:Y=Q35:GOSUB Q50
1740  ZA$="#":ZX=Q20:ZY=Q35:GOSUB Q40:? :? "520 SOUND 2,64,10,
      2":ZA$="B":ZX=Q22:ZY=Q55:GOSUB Q40
1820  P=Q64:GOSUB Q100:X=Q178:Y=Q29:GOSUB Q50
1850  ZA$="E+G#+B=E MAJOR":ZX=Q13:ZY=Q95:GOSUB Q40
1860  SOUND Q0,Q96,Q10,Q2:SOUND Q1,Q76,Q10,Q2:SOUND Q2,Q64,
      Q10,Q2:GOSUB Q60:GOSUB Q70:RETURN
1900  ZA$="THE F MAJOR CHORD":ZX=Q12:ZY=Q75:GOSUB Q40:? :?
      "500 SOUND 0,91,10,2":ZA$="F":ZX=Q16:ZY=Q55:GOSUB Q40
1942  P=Q91:GOSUB Q100:X=Q130:Y=Q38:GOSUB Q50:? :? "510 SOUND
      1,72,10,2"
2010  ZA$="A":ZX=Q19:ZY=Q55:GOSUB Q40:P=Q72:GOSUB Q100:X=
      Q154:Y=Q32:GOSUB Q50
2100  ? :? "520 SOUND 2,60,10,2":ZA$="C":ZX=Q22:ZY=Q55:GOSUB
      Q40:P=Q60:GOSUB Q100
2130  X=Q178:Y=Q26:GOSUB Q50:ZA$="F+A+C=F MAJOR":ZX=Q13:ZY=
      Q95:GOSUB Q40
2160  SOUND Q0,Q91,Q10,Q2:SOUND Q1,Q72,Q10,Q2:SOUND Q2,Q60,
      Q10,Q2:GOSUB Q60:GOSUB Q70:RETURN
2200  ZA$="THE F# MAJOR CHORD":ZX=Q12:ZY=Q75:GOSUB Q40:? :?
      "500 SOUND 0,85,10,2":ZA$="F#":ZX=Q16:ZY=Q55:GOSUB
      Q40
2242  P=Q85:GOSUB Q100:X=Q130:Y=Q38:GOSUB Q50:ZA$="#":ZX=
      Q17:ZY=Q38:GOSUB Q40
2300  ? :? "510 SOUND 1,68,10,2":ZA$="A#":ZX=Q19:ZY=Q55:GOSUB
      Q40:P=Q68:GOSUB Q100
2330  X=Q154:Y=Q32:GOSUB Q50:ZA$="#":ZX=Q20:ZY=Q32:GOSUB
      Q40:? :? "520 SOUND 2,57,10,2":ZA$="C#":ZX=Q22:ZY=Q55:
      GOSUB Q40
2420  P=Q57:GOSUB Q100:X=Q178:Y=Q26:GOSUB Q50:ZA$="#":ZX=
      Q23:ZY=Q26:GOSUB Q40
2450  ZA$="F#+A#+C#=F# MAJOR":ZX=Q13:ZY=Q95:GOSUB Q40
2460  SOUND Q0,Q85,Q10,Q2:SOUND Q1,Q68,Q10,Q2:SOUND Q2,Q57,
      Q10,Q2:GOSUB Q60:GOSUB Q70:RETURN
2500  ZA$="THE G MAJOR CHORD":ZX=Q12:ZY=Q75:GOSUB Q40:? :?
      "500 SOUND 0,81,10,2":ZA$="G":ZX=Q16:ZY=Q55:GOSUB Q40
2542  P=Q81:GOSUB Q100:X=Q130:Y=Q35:GOSUB Q50:? :? "510 SOUND
      1,64,10,2"
2610  ZA$="B":ZX=Q19:ZY=Q55:GOSUB Q40:P=Q64:GOSUB Q100:X=
      Q154:Y=Q29:GOSUB Q50
2700  ? :? "520 SOUND 2,53,10,2":ZA$="D":ZX=Q22:ZY=Q55:GOSUB
      Q40:P=Q53:GOSUB Q100
2730  X=Q178:Y=Q23:GOSUB Q50:ZA$="G+B+D=G MAJOR":ZX=Q13:ZY=
      Q95:GOSUB Q40
2760  SOUND Q0,Q81,Q10,Q2:SOUND Q1,Q64,Q10,Q2:SOUND Q2,Q53,
      Q10,Q2:GOSUB Q60:GOSUB Q70:RETURN
2800  ZA$="THE G# MAJOR CHORD":ZX=Q12:ZY=Q75:GOSUB Q40:? :?
      "500 SOUND 0,76,10,2":ZA$="G#":ZX=Q16:ZY=Q55:GOSUB
      Q40
2842  P=Q76:GOSUB Q100:X=Q130:Y=Q35:GOSUB Q50:ZA$="#":ZX=
      Q17:ZY=Q35:GOSUB Q40
2900  ? :? "510 SOUND 1,60,10,2":ZA$="C":ZX=Q19:ZY=Q55:GOSUB
      Q40:P=Q60:GOSUB Q100
2930  X=Q154:Y=Q26:GOSUB Q50:? :? "520 SOUND 2,50,10,2":ZA$=
      "D#":ZX=Q22:ZY=Q55:GOSUB Q40

3020  P=Q50:GOSUB Q100:X=Q178:Y=Q23:GOSUB Q50:ZA$="#":ZX=
      Q23:ZY=Q23:GOSUB Q40
3050  ZA$="G#+C+D#=G# MAJOR":ZX=Q13:ZY=Q95:GOSUB Q40
3060  SOUND Q0,Q76,Q10,Q2:SOUND Q1,Q60,Q10,Q2:SOUND Q2,Q50,
      Q10,Q2:GOSUB Q60:RETURN
30000 GRAPHICS 0:POKE 752,1:POKE 710,48:POKE 82,2:POKE 201,
      0
```

```
  1 REM BEISPIEL 7
  2 REM
  3 REM MAJOR CHORDS IN DISTORTION LEVEL 12 BASS RANGE
  4 REM ALL PITCHES IN DISTORTION LEVEL 10 CLEAR SOUND
  5 REM FOUR TEXT COLORS USING GRAPHICS MODE 18
 20 DIM A$(8),AS$(8),B$(8),C$(8),CS$(8),D$(8),DS$(8),
    E$(8),F$(8),FS$(8),G$(8),GS$(8)
 22 DIM P1(13),P2(13),P3(13),MC$(11):MC$="[MAJOR CHORD]"
 30 A$="A   C# E ":AS$="A# D  F ":B$="B  D#  F#":C$="C  E
    G ":CS$="C# F  G#"
 32 D$="D  F# A ":DS$="D# G  A#":E$="E  G# B ":F$="F  A
    C ":FS$="F# A#  C#"
 34 G$="G  B  D ":GS$="G# C  D#":GOTO 200
 35 ? #6;"   A  ";MC$:RETURN
 40 ? #6;"  A# ";MC$:RETURN
 45 ? #6;"   B ";MC$:RETURN
 50 ? #6;"   C ";MC$:RETURN
 55 ? #6;"  C# ";MC$:RETURN
 60 ? #6;"   D ";MC$:RETURN
 65 ? #6;"  D# ";MC$:RETURN
 70 ? #6;"   E ";MC$:RETURN
 75 ? #6;"   F ";MC$:RETURN
 80 ? #6;"  F# ";MC$:RETURN
 85 ? #6;"   G ";MC$:RETURN
 90 ? #6;"  G# ";MC$:RETURN
 95 ? #6;"   A ";MC$:RETURN
 99 FOR WAIT=1 TO 200:NEXT WAIT:RETURN
100 DATA 75,60,51,72,57,48,67,55,45,63,51,42,60,48,40,57,
    45,37,55,42,36
102 DATA 51,40,33,48,37,31,45,36,30,42,33,28,40,31,27,37,
    30,25
200 FOR CHORD=1 TO 13:READ P1,P2,P3:P1(CHORD)=P1:
    P2(CHORD)=P2:P3(CHORD)=P3:NEXT CHORD
1000 FOR CHORD=1 TO 13:GRAPHICS 18:POKE 559,0:? #6
1020 ON CHORD GOSUB 35,40,45,50,55,60,65,70,75,80,85,90,95
1040 POKE 559,34:? #6;" sound 1,";P1(CHORD);",12,4"
1050 SOUND 1,P1(CHORD),12,4:GOSUB 99
1060 ? #6:? #6;"   sound 2,";P2(CHORD);",12,2"
1070 SOUND 2,P2(CHORD),12,2:GOSUB 99
1080 ? #6:? #6;"   sound 3,";P3(CHORD);",12,2"
1090 SOUND 3,P3(CHORD),12,2:GOSUB 99
1400 FOR HOLD=1 TO 200:NEXT HOLD
1500 SOUND 1,0,0,0:SOUND 2,0,0,0:SOUND 3,0,0,0
1600 ? #6:? #6;"   [press start]"
1700 IF PEEK(53279)<>6 THEN 1700
1800 NEXT CHORD
2000 GRAPHICS 18:? #6:? #6;"   [THE clear ]sound"
2100 FOR PITCH=255 TO 0 STEP -1:POSITION 3,4
2120 SETCOLOR 0,PITCH,6:SETCOLOR 3,PITCH+8,6:SETCOLOR 1,
    PITCH+4,6
2200 ? #6;"SOUND 0,";PITCH;",10,V  "
2220 FOR VOLUME=14 TO 0 STEP -2:SOUND 0,PITCH,10,VOLUME:
    NEXT VOLUME:NEXT PITCH
30000 GRAPHICS 0:POKE 752,1:POKE 710,48:POKE 82,2:POKE 201,
    9
30006 RUN
```

```
  1 REM BEISPIEL 8
  2 REM
  3 REM
  4 REM
 10 DIM LINE$(40),N(50):VO=0:V1=1:V2=2:V3=3:GOTO 100
 30 SOUND VO,N(P),10,14:GOTO 50
 40 PO=N(P):P1=N(CHORD):P2=N(CHORD-4):P3=N(CHORD-7)
 42 SOUND VO,PO,10,14:SOUND V1,P1,10,6:SOUND V2,P2,10,6:
    SOUND V3,P3,10,6
 50 POKE 540,WAIT
 52 IF PEEK(540)<>0 THEN 52
 54 SOUND VO,0,0,0:RETURN
 60 FOR OFF=0 TO 3:SOUND OFF,0,0,0:NEXT OFF:RETURN
 70 PO=N(P):P1=N(CHORD):P2=N(CHORD-4):P3=N(CHORD-7)
 72 FOR DECAY=8 TO 0 STEP -1:SOUND VO,PO,10,DECAY:SOUND
    V1,P1,10,DECAY:SOUND V2,P2,10,DECAY:SOUND V3,P3,10,
    DECAY
 74 NEXT DECAY:RETURN
100 DATA 14,15,16,17,18,19,21,22,23,24,26,27,29,31,33,35,
    37,40,42,45,47,50,53,57,60,64,68,72,76,81,85,91,96
110 DATA 102,108,114,121,128,136,144,153,162,173,182,193,
    204,217,230,243,255
120 GOSUB 21000:FOR X=1 TO 50:READ IT:N(X)=IT:NEXT X
200 TRAP 19000:POKE 82,8:? "  ";
210 READ LINE$,CHORD,P,WAIT:? LINE$:GOSUB 40
220 FOR ME=1 TO 6:READ P,WAIT:GOSUB 30:NEXT ME:GOSUB 60:
    WAIT=10:GOSUB 50
230 READ LINE$,CHORD,P,WAIT:? :? LINE$:GOSUB 40
240 FOR ME=1 TO 6:READ P,WAIT:GOSUB 30:NEXT ME:GOSUB 60:
    WAIT=30:GOSUB 50
250 READ LINE$,CHORD,P,WAIT:? :? LINE$:GOSUB 40
260 FOR ME=1 TO 6:READ P,WAIT:GOSUB 30:NEXT ME:GOSUB 60:
    WAIT=10:GOSUB 50
270 READ LINE$,CHORD,P,WAIT:? :? LINE$:GOSUB 40
280 FOR ME=1 TO 6:READ P,WAIT:GOSUB 30:NEXT ME:GOSUB 60:
    WAIT=30:GOSUB 50
290 READ LINE$,CHORD,P,WAIT:? :? LINE$:GOSUB 40
300 FOR ME=1 TO 5:READ P,WAIT:GOSUB 30:NEXT ME
310 READ CHORD,P,WAIT:GOSUB 40:GOSUB 60:WAIT=30:GOSUB 50
320 READ LINE$,CHORD,P,WAIT:? :? LINE$:GOSUB 40
330 FOR ME=1 TO 5:READ P,WAIT:GOSUB 30:NEXT ME
340 READ CHORD,P,WAIT:GOSUB 40:GOSUB 60:WAIT=30:GOSUB 50
350 READ LINE$,CHORD,P,WAIT:? :? LINE$:GOSUB 40
360 FOR ME=1 TO 5:READ P,WAIT:GOSUB 30:NEXT ME
370 READ CHORD,P,WAIT:GOSUB 40:GOSUB 60:WAIT=10:GOSUB 50
380 READ LINE$,CHORD,P,WAIT:? :? LINE$:GOSUB 40
390 READ P,WAIT:GOSUB 30
400 READ CHORD,P,WAIT:GOSUB 40
410 READ P,WAIT:GOSUB 30
420 READ CHORD,P,WAIT:GOSUB 40
430 READ P,WAIT:GOSUB 30
440 READ CHORD,P,WAIT:GOSUB 40
450 GOSUB 60:WAIT=10:GOSUB 50:FOR DECAY=15 TO 0 STEP -0.5:
    SOUND VO,N(1),10,DECAY:NEXT DECAY
460 GRAPHICS 18:? #6;? #6;"      [major chords]"
510 FOR ME=1 TO 8:READ CHORD,P,LINE$:POSITION ME*2,10-ME:
    ? #6;LINE$:GOSUB 70:NEXT ME
530 FOR ME=8 TO 1 STEP -1:READ CHORD,P,LINE$:POSITION ME*
    2,10-ME:? #6;LINE$:GOSUB 70:NEXT ME
540 WAIT=15:GOSUB 50:POSITION 4,11:? #6;"PRESS"
550 FOR DECAY=15 TO 0 STEP -0.5:SOUND VO,N(6),10,DECAY:
    NEXT DECAY
560 WAIT=15:GOSUB 50:POSITION 10,11:? #6;"START"
570 FOR DECAY=15 TO 0 STEP -0.5:SOUND VO,N(1),10,DECAY:
    NEXT DECAY
600 SETCOLOR 0,PEEK(20),10:IF PEEK(53279)<>6 THEN 600
700 GRAPHICS 18:SETCOLOR 0,1,10:SETCOLOR 1,11,12:SETCOLOR
    3,4,12
710 ? #6:? #6:? #6;"     PRESS option":? #6;"       TO
    RERUN"
720 WAIT=60:GOSUB 50
730 ? #6:? #6;"     PRESS [start]":? #6;"     TO CONTINUE"
```

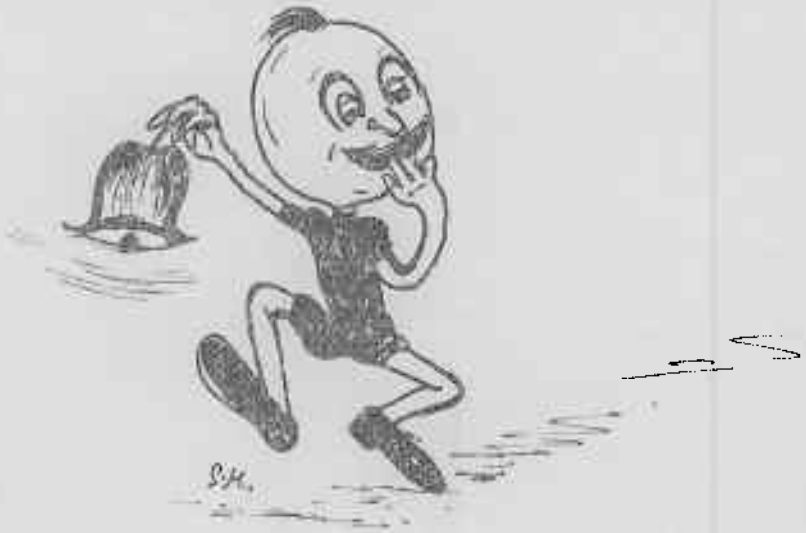
```
  740 IF PEEK(53279)=3 THEN RUN
  750 IF PEEK(53279)=6 THEN 900
  760 GOTO 740
  900 RUN "D:BIRTHDAY.BAS"
 1000 DATA DOE A DEER A FEMALE DEER
 1010 DATA 49,37,45,35,15,33,45,37,15,33,30,37,30,33,45
 1020 DATA RAY A DROP OF GOLDEN SUN
 1030 DATA 42,35,45,33,15,32,15,32,15,33,15,35,15,32,90
 1040 DATA ME A NAME I CALL MYSELF
 1050 DATA 49,33,45,32,15,30,45,33,15,30,30,33,30,30,45
 1060 DATA FA A LONG LONG WAY TO RUN
 1070 DATA 44,32,45,30,15,28,15,28,15,30,15,32,15,28,90
 1080 DATA SEW A NEEDLE PULLING THREAD
 1090 DATA 49,30,45,37,15,35,15,33,15,32,15,30,15,44,28,90
 1100 DATA LA A NOTE TO FOLLOW SEW
 1110 DATA 44,28,45,35,15,33,15,31,15,30,15,28,15,42,26,90
 1120 DATA TEA A DRINK WITH JAM AND BREAD
 1130 DATA 42,26,45,33,15,31,15,29,15,28,15,26,15,49,25,90
 1140 DATA THAT WILL BRING US BACK TO DOE
 1150 DATA 49,25,15,26,15
 1160 DATA 44,28,30,32,30
 1170 DATA 42,26,30,30,30
 1180 DATA 40,25,90
 1200 DATA 49,37,C
 1210 DATA 47,35,D
 1220 DATA 45,33,E
 1230 DATA 44,32,F
 1240 DATA 42,30,G
 1250 DATA 40,28,A
 1260 DATA 38,26,B
 1270 DATA 37,25,C
 1300 DATA 37,25,c
 1310 DATA 38,26,b
 1320 DATA 40,28,a
 1330 DATA 42,30,g
 1340 DATA 44,32,f
 1350 DATA 45,33,e
 1360 DATA 47,35,d
 1370 DATA 49,37,c
21000 GRAPHICS 0:SETCOLOR 2,9,0:SETCOLOR 4,9,0:SETCOLOR 1,9,
      12:POKE 752,1:POKE 82,2:POKE 83,39:POKE 201,7
21010 ? " ":? "{Q}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}
      {R}{R}{R}{R}{R}{R}{R}{R}{E}"
21020 ? "[ MAJOR CHORD HARMONY ]:"
21030 ? "{Z}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}
      {R}{R}{R}{R}{R}{R}{C}"
21040 POKE 77,0:RETURN
```

```
    1 REM BEISPIEL 9
    2 REM
    3 REM
    4 REM
   20 GRAPHICS 0:DIM NAME$(20):? :? "BIRTHDAY SONG PROGRAM:
      ":? :? "ENTER NAME:";:INPUT NAME$
   40 DIM WORD$(20),DISP$(20),BLANK$(20):BLANK$="
                ":GOTO 300
  100 POKE 540,HOLD:SOUND 0,V0,10,8:SOUND 1,V1,10,4:SOUND 2,
      V2,10,4:SOUND 3,V3,10,4
  101 IF PEEK(540)<>0 THEN 101
  102 SOUND 0,0,0,0:SOUND 1,0,0,0:SOUND 2,0,0,0:SOUND 3,0,0,
      0:RETURN
  300 READ V0,V1,V2,V3,HOLD,SWITCH,WORD$
  310 IF SWITCH=3 THEN WORD$=NAME$:GOTO 360
  330 IF SWITCH=1 THEN GRAPHICS 18:GOTO 360
  340 IF SWITCH=9 THEN 9000
  360 LW=LEN(WORD$):DISP$=BLANK$:IF LW>18 THEN DISP$=WORD$:
      GOTO 380
  370 FP=11-INT(LW/2):DISP$(FP,FP+LW-1)=WORD$
  380 POSITION 0,4:? #6;DISP$:GOSUB 100:GOTO 300
  500 DATA 121,121,243,243,19,1,HA[PPY BIRTHDAY]
  510 DATA 121,121,243,243,19,0,HAPPY [BIRTHDAY]
  520 DATA 108,108,217,217,39,0,HAPPY BIRTH[DAY]
  530 DATA 121,121,243,243,39,0,HAPPY BIRTHDAY
  540 DATA 91,91,182,182,39,1,TO[ YOU]
  550 DATA 96,96,193,193,79,0,TO YOU
  600 DATA 121,121,243,243,19,1,HA[PPY BIRTHDAY]
  610 DATA 121,121,243,243,19,0,HAPPY [BIRTHDAY]
  620 DATA 108,108,217,217,39,0,HAPPY BIRTH[DAY]
  630 DATA 121,121,243,243,39,0,HAPPY BIRTHDAY
  640 DATA 81,81,162,162,39,1,TO[ YOU]
  650 DATA 91,91,182,182,79,0,TO YOU
  700 DATA 121,121,243,243,19,1,HA[PPY BIRTHDAY]
  710 DATA 121,121,243,243,19,0,HAPPY [BIRTHDAY]
  720 DATA 60,60,121,121,39,0,HAPPY BIRTH[DAY]
  730 DATA 72,72,144,144,39,0,HAPPY BIRTHDAY
  740 DATA 91,91,182,182,39,1,DEAR
  750 DATA 96,96,193,193,39,3,NAME
  760 DATA 108,108,217,217,79,3,NAME
  770 DATA 0,0,0,0,19,3,NAME
  800 DATA 68,68,136,136,19,1,HA[PPY BIRTHDAY]
  810 DATA 68,68,136,136,19,0,HAPPY [BIRTHDAY]
  820 DATA 72,72,144,144,39,0,HAPPY BIRTH[DAY]
  830 DATA 91,91,182,182,39,0,HAPPY BIRTHDAY
  840 DATA 81,96,121,162,39,1,TO [YOU]
  850 DATA 91,121,144,182,79,0,TO YOU
 8000 DATA 0,0,0,0,0,9,END
 9000 GRAPHICS 18:SETCOLOR 2,4,8:SETCOLOR 0,12,6:SETCOLOR 4,
      4,0
 9100 POSITION 3,4:? #6;"HAPPY BIRTHDAY"
 9200 FOR HOLD=1 TO 500:NEXT HOLD:GOTO 30000
30000 GRAPHICS 0:POKE 752,1:POKE 710,48:POKE 82,2:POKE 201,
      9
30006 RUN
```

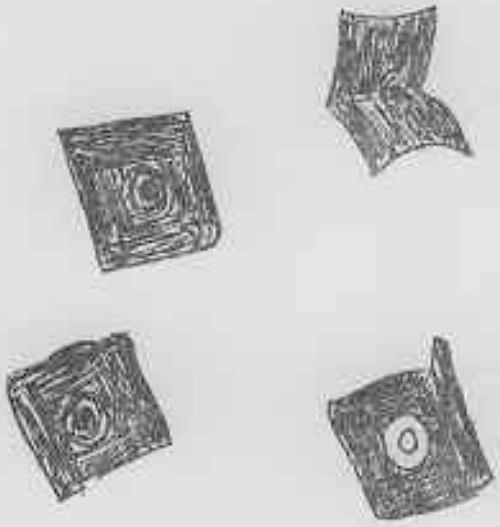


```
    1 REM BEISPIEL 10
    2 REM
    3 REM
    4 REM
   20 GRAPHICS 18:SETCOLOR 2,4,8:SETCOLOR 0,12,6:SETCOLOR 4,
      4,0
   40 DIM WORD$(20),DISP$(20),BLANK$(20):BLANK$="
                ":GOTO 300
  100 POKE 540,HOLD:SOUND 0,V0,10,8:SOUND 1,V1,10,4:SOUND 2,
      V2,10,4:SOUND 3,V3,10,4
  101 IF PEEK(540)<>0 THEN 101
  102 SOUND 0,0,0,0:SOUND 1,0,0,0:SOUND 2,0,0,0:SOUND 3,0,0,
      0:RETURN
  300 READ V0,V1,V2,V3,HOLD,SWITCH,WORD$
  330 IF SWITCH=2 THEN WORD$=DISP$
  340 IF SWITCH=9 THEN 9000
  360 LW=LEN(WORD$):DISP$=BLANK$:IF LW>18 THEN DISP$=WORD$:
      GOTO 380
  370 FP=11-INT(LW/2):DISP$(FP,FP+LW-1)=WORD$
  380 POSITION 0,4:? #6;DISP$:GOSUB 100:GOTO 300
  600 DATA 81,96,121,162,89,1,SI[LENT NIGHT]
  610 DATA 72,72,144,144,29,0,SI[LENT NIGHT]
  620 DATA 81,96,121,162,59,0,SILENT [NIGHT]
  630 DATA 96,121,81,193,119,0,SILENT NIGHT
  640 DATA 0,0,0,0,59,2,REST
  650 DATA 81,96,121,162,89,1,HO[LY NIGHT]
  660 DATA 72,72,144,144,29,0,HO[LY NIGHT]
  670 DATA 81,96,121,162,59,0,HOLY [NIGHT]
  680 DATA 96,121,81,193,119,0,HOLY NIGHT
  690 DATA 0,0,0,0,59,2,REST
  700 DATA 53,64,81,108,119,1,ALL [IS CALM]
  710 DATA 53,64,81,108,59,0,ALL IS [CALM]
  720 DATA 64,81,108,128,119,0,ALL IS CALM
  730 DATA 0,0,0,0,59,2,REST
  740 DATA 60,81,96,121,119,0,ALL [IS BRIGHT]
  750 DATA 60,81,96,121,59,0,ALL IS [BRIGHT]
  760 DATA 81,96,121,162,119,0,ALL IS BRIGHT
  770 DATA 0,0,0,0,59,2,REST
  800 DATA 72,91,121,144,119,1,ROUND [YON]
  810 DATA 72,91,121,144,59,0,ROUND YON
  820 DATA 60,72,91,121,89,1,VIRGIN]
  830 DATA 64,64,128,128,29,0,VIR[GIN]
  840 DATA 72,71,121,144,59,0,VIRGIN
  850 DATA 81,96,121,162,89,1,MO[THER AND CHILD]
  860 DATA 72,72,144,144,29,0,MOTHER [AND CHILD]
  870 DATA 81,96,121,162,59,0,MOTHER AND [CHILD]
  880 DATA 96,121,81,193,119,0,MOTHER AND CHILD
  890 DATA 0,0,0,0,59,2,REST
  900 DATA 72,91,121,144,119,1,HO[LY INFANT SO]
  910 DATA 72,91,121,144,59,0,HOLY [INFANT SO]
  920 DATA 60,72,91,121,89,0,HOLY INI[FANT SO]
  930 DATA 64,64,128,128,29,0,HOLY INFANT [SO]
  940 DATA 72,71,121,144,59,0,HOLY INFANT SO
  950 DATA 81,96,121,162,89,1,TEN[DER AND MILD]
  960 DATA 72,72,144,144,29,0,TENDER [AND MILD]
  970 DATA 81,96,121,162,59,0,TENDER AND [MILD]
  980 DATA 96,121,81,193,119,0,TENDER AND MILD
  990 DATA 0,0,0,0,59,2,REST
 1000 DATA 53,64,81,108,119,1,SLEEP [IN]
 1010 DATA 53,64,81,108,59,0,SLEEP IN
 1020 DATA 45,53,64,81,89,1,HEA[VENLY PEACE]
 1030 DATA 53,64,81,108,29,0,HEAVENLY PEACE]
 1040 DATA 64,81,108,128,59,0,HEAVENLY [PEACE]
 1050 DATA 60,81,96,121,179,0,HEAVENLY PEACE
 1060 DATA 47,60,81,96,119,0,HEAVENLY PEACE
 1070 DATA 0,0,0,0,59,2,REST
 1100 DATA 60,81,96,121,59,1,SLEEP [IN]
 1110 DATA 81,96,121,162,59,0,SLEEP [IN]
 1120 DATA 96,121,162,193,59,0,SLEEP IN
 1130 DATA 81,108,128,162,89,1,HEA[VENLY PEACE]
 1140 DATA 91,91,182,182,29,0,HEAVENLY PEACE]
 1150 DATA 108,128,162,217,59,0,HEAVENLY [PEACE]
 1160 DATA 121,162,193,243,179,0,HEAVENLY PEACE
 8000 DATA 0,0,0,0,0,9,END
 9000 GRAPHICS 18:SETCOLOR 2,4,8:SETCOLOR 0,12,6:SETCOLOR 4,
      4,0
 9100 POSITION 3,4:? #6;"MERRY CHRISTMAS"
 9200 FOR HOLD=1 TO 500:NEXT HOLD:GOTO 30000
30000 GRAPHICS 0:POKE 752,1:POKE 710,48:POKE 82,2:POKE 201,
      9
30005 RUN
```

```
1090 DATA 40,53,64,81,19,0,*
1100 DATA 40,53,64,81,19,0,*
1110 DATA 45,53,64,81,19,1,OPEN SLEIGH
1120 DATA 53,64,81,108,19,0,*
1130 DATA 60,81,96,121,79,0,*
1140 DATA 0,0,0,0,9,2,REST
1200 DATA 81,96,121,162,19,1,DASHING THRU
1210 DATA 47,47,96,96,19,0,*
1220 DATA 53,53,108,108,19,0,*
1230 DATA 60,60,121,121,19,1,THE SNOW
1240 DATA 81,96,121,162,59,0,*
1245 DATA 0,0,0,0,9,2,REST
1250 DATA 81,96,121,162,9,1,IN A ONE HORSE
1260 DATA 81,96,121,162,9,0,*
1270 DATA 81,96,121,162,19,0,*
1280 DATA 47,47,96,96,19,0,*
1290 DATA 53,53,108,108,19,1,OPEN SLEIGH
1300 DATA 60,60,121,121,19,0,*
1310 DATA 72,91,121,144,79,0,*
1320 DATA 0,0,0,0,9,2,REST
1330 DATA 72,91,121,144,19,1,OVER THE FIELDS
1340 DATA 45,45,91,91,19,0,*
1350 DATA 47,47,96,96,19,0,*
1360 DATA 53,53,108,108,19,1,WE GO
1370 DATA 64,81,108,128,79,0,*
1380 DATA 0,0,0,0,9,2,REST
1390 DATA 40,47,60,81,19,1,LAUGHING
1400 DATA 40,47,60,81,19,0,*
1410 DATA 45,45,91,91,19,1,ALL THE WAY
1420 DATA 53,53,108,108,19,0,*
1430 DATA 47,60,81,96,79,0,*
1440 DATA 0,0,0,0,9,2,REST
1500 DATA 81,96,121,162,19,1,BELLS ON BOB
1510 DATA 47,47,96,96,19,0,*
1520 DATA 53,53,108,108,19,0,*
1530 DATA 60,60,121,121,19,1,TAILS RING
1540 DATA 81,96,121,162,79,0,*
1545 DATA 0,0,0,0,9,2,REST
1550 DATA 81,96,121,162,19,1,MAKING SPIRITS
1560 DATA 47,47,96,96,19,0,*
1570 DATA 53,53,108,108,19,0,*
1580 DATA 60,60,121,121,19,0,*
1590 DATA 72,91,121,144,79,1,BRIGHT
1600 DATA 0,0,0,0,9,2,REST
1610 DATA 72,91,121,144,9,1,WHAT FUN IT IS
1620 DATA 72,91,121,144,19,0,*
1630 DATA 45,45,91,91,19,0,*
1640 DATA 47,47,96,96,19,0,*
1650 DATA 53,53,108,108,19,1,TO RIDE AND SING
1660 DATA 40,53,64,81,19,0,*
1670 DATA 40,53,64,81,19,0,*
1680 DATA 40,53,64,81,29,0,*
1700 DATA 40,53,64,81,9,1,A SLEIGHING
1710 DATA 35,35,72,72,9,0,*
1720 DATA 40,53,64,81,19,0,*
1730 DATA 45,45,91,91,19,1,SONG TONIGHT
1740 DATA 53,53,108,108,19,0,*
1750 DATA 60,81,96,121,29,0,*
2000 POKE 752,1:POKE 559,62:I=PEEK(106)-32:POKE 54279,I:
     POKE 623,33
2040 VTAB=PEEK(134)+PEEK(135)*256
2050 ATAB=PEEK(140)+PEEK(141)*256
2060 OFFS=I*254+1024-ATAB
2070 HI=INT(OFFS/256):LO=OFFS-HI*256
2090 POKE VTAB+2,LO:POKE VTAB+3,HI
2100 DIM MC$(42),PN$="hh[(E)L]hl(E)K]hl(E)N]hl(E)M&N ]
     (,)[])[(,)l(G)K(H)P(;)fL]]O(F)[Pt$MPoFL ]( ,)[(Q)K](.)":
     CLEAR=USR(ADR(MC$),ADR(PN$),1024):GOTO 10000
8000 DATA 0,0,0,0,0,9,END
9000 POKE 201,10:? CHR$(125)," MERRY CHRISTMAS":? ,"from
     Jerry White"
9200 FOR HOLD=0 TO 231 STEP 0.5:POKE 704,HOLD:POKE 705,
```



```
    HOLD+8:POKE 706,HOLD+16:POKE 707,HOLD+24:NEXT HOLD
 9300 GOSUB 20000:POKE 53277,0:GOTO 30000
 9500 TIMES=TIMES+1:FOR HOLD=1 TO 40:NEXT HOLD:RESTORE :
    GOTO 300
10000 PLL$(1,1)=CHR$(32):PLL$(2,2)=CHR$(112):PLL$(3,3)=
    CHR$(248):PLL$(4,4)=PLL$(2,2):PLL$(5,5)=PLL$(1,1)
10003 PLR$(1,1)=CHR$(4):PLR$(2,2)=CHR$(14):PLR$(3,3)=
    CHR$(31):PLR$(4,4)=PLR$(2,2):PLR$(5,5)=PLR$(1,1)
10004 FOR ME=70 TO 150 STEP 20:PM$(ME+256,ME+260)=PLR$:
    PM$(ME+266,ME+270)=PLL$
10005 PM$(ME+512,ME+516)=PLR$:PM$(ME+522,ME+526)=PLL$:NEXT
    ME
10006 PM$(114,118)=PLR$:PM$(134,138)=PLR$:PM$(154,158)=PLL$
10007 PM$(882,892)=PLL$:PM$(902,906)=PLL$:PM$(922,926)=PLR$
10008 POKE 708,196:POKE 710,65:POKE 712,14:POKE 709,72:POKE
    201,15:POKE 65,0:POKE 82,2:? :GOSUB 20000
10009 SOUND 0,0,0,0:POKE 53761,168:POKE 53763,164:POKE
    53765,164:POKE 53767,164
10010 POKE 765,1:FOR X=2 TO 6:A=8*X+80:B=11*X:C=80-8*X
10020 COLOR 1:PLOT 81,9:PLOT A-16,B-13:DRAWTO A,B:PLOT C-1,
    B+1
10030 POSITION C+16,B-13:XIO 18,#6,12,0,"S:"
10040 COLOR 2:PLOT A,B:DRAWTO A-1,B+1:PLOT C,B:DRAWTO C+1,B+
    1:DRAWTO C-1,B+1
10042 PLOT A,B+1:DRAWTO A+1,B+1:NEXT X
10050 COLOR 0:PLOT 81,9:COLOR 3:POKE 765,3:PLOT 90,67
10060 DRAWTO 90,79:DRAWTO 70,79:POSITION 70,67:XIO 18,#6,12,
    0,"S:"
10070 COLOR 2:POKE 765,2:PLOT 80,1:DRAWTO 85,10:DRAWTO 83,9
10080 POSITION 74,4:XIO 18,#6,12,0,"S:":DRAWTO 86,4:DRAWTO
    75,10
10090 DRAWTO 80,1:DRAWTO 80,5:DRAWTO 76,9:PLOT 83,5:POKE
    201,13:POKE 53277,2
11000 POKE 53248,102:POKE 53249,116:POKE 53250,132:POKE
    53251,146:RETURN
20000 FOR ME=53248 TO 53251:POKE ME,0:NEXT ME:RETURN
30000 GRAPHICS 0:POKE 752,1:POKE 710,48:POKE 82,2:POKE 201,
    9
30005 RUN
```

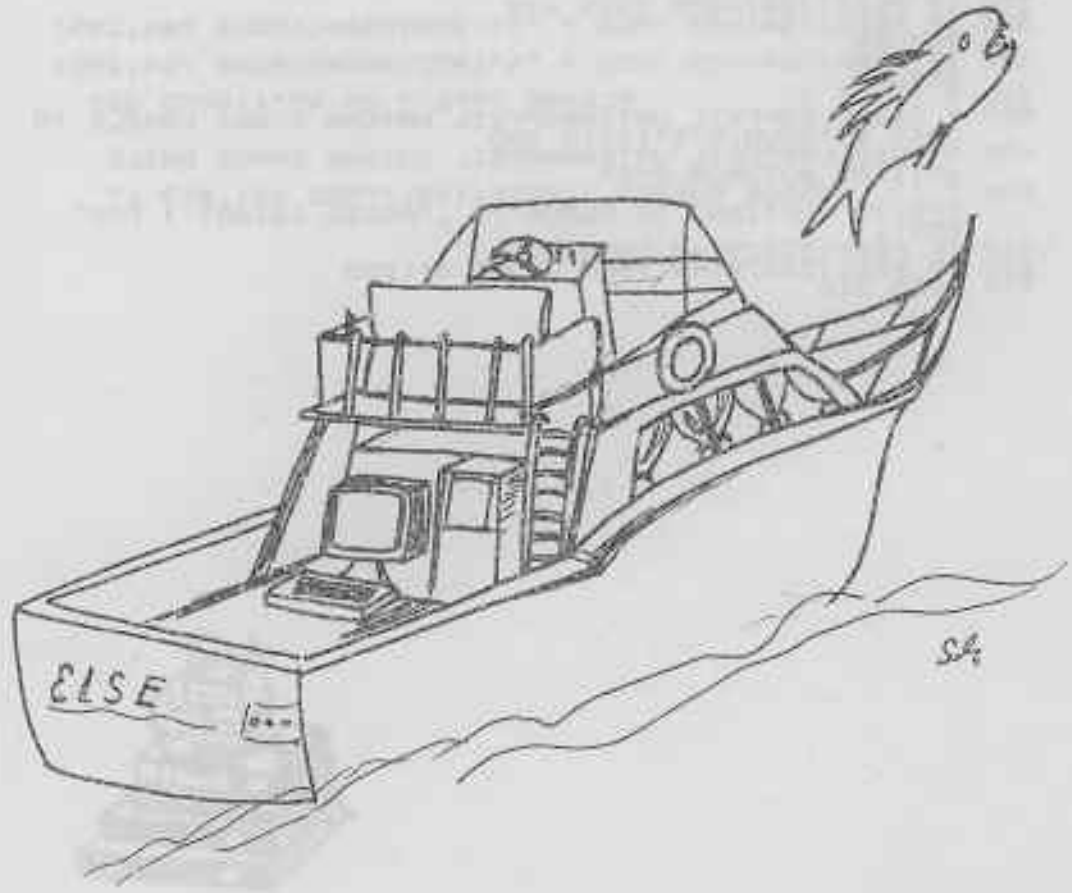


DISKETTEN
AUFBEWAHREN

```
  0 REM FILE PROGRAMM
  1 REM
  2 REM
100 DIM DRIVE$(3),FILE$(12),DRIVEFILE$(15),RECORD$(10),
    ANSWER$(1)
110 DIM SECTOR(20),BYTE(20),DIRECTORY$(20):REM DIMENSION
    STRINGS AND ARRAYS
111 REM
120 GRAPHICS 0:POKE 82,2:POKE 83,39:REM CLEAR SCREEN AND
    SET MARGINS
130 POKE 201,5:SETCOLOR 2,1,0:REM SET PRINT TAB WIDTH TO
    5 SPACES AND COLOR
140 ? :? "TYPE OPTION NUMBER THEN PRESS RETURN"
150 ? :? ,"(1) CREATE A DISK FILE":REM GOTO 1000
160 ? :? ,"(2) READ A DISK FILE":REM GOTO 2000
170 ? :? ,"(3) ADD TO A DISK FILE":REM GOTO 3000
180 ? :? ,"(4) UPDATE A DISK FILE":REM GOTO 4000
190 ? :? ,"(5) DISPLAY DISK DIRECTORY":REM GOTO 5000
200 ? :? ,"(6) RUN MENU":REM GOTO 9140
210 ? :? ,"YOUR CHOICE";:GOSUB 7000
220 TRAP 8000:LINE=120:HIGHNUMBER=6:NUMBER=VAL(ANSWER$)
230 IF NUMBER<1 OR NUMBER>6 THEN GOTO 8000
240 ON NUMBER GOTO 1000,2000,3000,4000,5000,9140
250 REM
1000 LINE=6100:GOSUB 7100:TRAP 9100:GRAPHICS 0:SETCOLOR 2,
    4,0
1010 CLOSE #1:OPEN #1,8,0,DRIVEFILE$
1020 ? :? "CREATING ";DRIVEFILE$:? :RECORD$="1234567890"
1030 FOR DEMO=1 TO 10
1040 ? #1;RECORD$
1050 ? "WRITING RECORD NUMBER ";DEMO
1060 NEXT DEMO
1070 ? :? "10 RECORD DEMO FILE CREATED"
1080 ? :? "CLOSING ";DRIVEFILE$
1090 CLOSE #1
1100 GOTO 6100
1110 REM
2000 LINE=6100:GOSUB 7100:TRAP 9100:GRAPHICS 0:SETCOLOR 2,
    5,0:? :?
2010 CLOSE #2:OPEN #2,4,0,DRIVEFILE$:RECNUM=0:LINE=6100
2020 INPUT #2,RECORD$
2030 RECNUM=RECNUM+1
2040 ? "RECORD NUMBER ";RECNUM;
2050 ? ,RECORD$
2060 GOTO 2020
2070 REM
3000 LINE=3000:GOSUB 7100:TRAP 9100:GRAPHICS 0:SETCOLOR 2,
    7,0
3010 CLOSE #3:OPEN #3,9,0,DRIVEFILE$
3020 GRAPHICS 0:? :? "ADD RECORD(S) ROUTINE:"
3030 ? :? ,"ENTER 10 CHARACER RECORD"
3040 ? :? ,"OR JUST PRESS RETURN TO EXIT":? :GOSUB 6000
3050 RECLEN=LEN(RECORD$):IF RECLEN=0 THEN 3200
3060 IF RECLEN=10 THEN 3090
3070 FOR BLANK=RECLEN+1 TO 10:RECORD$(LEN(RECORD$)+1)=" ":
    NEXT BLANK
3090 PRINT #3;RECORD$
3100 ? :? "PRESS START TO ENTER ANOTHER RECORD"
3110 ? :? "PRESS OPTION FOR OTHER OPTIONS...";
3120 IF PEEK(53279)=6 THEN 3020
3130 IF PEEK(53279)=3 THEN 3200
3140 GOTO 3120
3200 ? :? :? ,"ADDING RECORD(S) TO DISK":CLOSE #3:GOTO 120
3210 REM
4000 LINE=4100:GOSUB 7100:TRAP 9100:GRAPHICS 0:SETCOLOR 2,
    10,0
4010 CLOSE #4:OPEN #4,12,0,DRIVEFILE$:LINE=4100
4020 ? :? ,"CREATING INDEX":RECNUM=0
4030 NOTE #4,SECTOR,BYTE
4040 RECNUM=RECNUM+1
4050 SECTOR(RECNUM)=SECTOR:BYTE(RECNUM)=BYTE
4060 INPUT #4,RECORD$:? ," RECORD ";RECNUM,RECORD$
```

```
4070 ? "SECTOR=";SECTOR,"BYTE=";BYTE
4080 ? :GOTO 4030
4100 RECNUM=RECNUM-1
4110 ? :? "PRESS START TO UPDATE A RECORD"
4120 ? :? "PRESS OPTION FOR OTHER OPTIONS";
4130 IF PEEK(53279)=6 THEN 4200
4140 IF PEEK(53279)=3 THEN CLOSE #4:GOTO 120
4150 GOTO 4130
4200 GRAPHICS 0:REM RANDOM ACCESS RECORD UPDATE ROUTINE
4210 ? :? "DISKFILE CONTAINS ";RECNUM;" RECORDS"
4220 ? :? "ENTER RECORD NUMBER TO BE UPDATED";
4230 TRAP 4220:INPUT UPDATE:TRAP 40000
4240 UPDATE=INT(UPDATE):IF UPDATE<1 OR UPDATE>RECNUM THEN
     4230
4250 POINT #4,SECTOR(UPDATE),BYTE(UPDATE)
4260 INPUT #4,RECORD$:? :? RECORD$
4270 ? :? "ENTER NEW RECORD #";UPDATE;:INPUT RECORD$
4280 RECLEN=LEN(RECORD$):IF RECLEN=10 THEN 4300
4290 FOR BLANK=RECLEN+1 TO 10:RECORD$(LEN(RECORD$)+1)=" ":
     NEXT BLANK
4300 POINT #4,SECTOR(UPDATE),BYTE(UPDATE)
4310 PRINT #4;RECORD$:? :? ,"RECORD HAS BEEN UPDATED"
4320 GOTO 4110
4330 REM
5000 GRAPHICS 0:POKE 752,1:SETCOLOR 2,12,0:POKE 201,8:? :?
     "    DISK DIRECTORY":? :TRAP 9100
5010 CLOSE #5:OPEN #5,6,0,"D:*.*":REM OPEN DISK DIRECTORY
     FOR ALL ENTRIES
5020 LINE=6100
5030 INPUT #5,DIRECTORY$
5040 ? ,DIRECTORY$
5050 GOTO 5030
5060 REM
6000 RECORD$="":POKE 764,255:REM RECORD STRING AND LAST
     KEY PRESSED=NULL
6010 INPUT RECORD$:RETURN
6020 REM
6100 FOR FILE=1 TO 5:CLOSE #FILE:NEXT FILE:REM CLOSE ALL
     FILES
6110 POKE 201,5:? :? ,"PRESS RETURN FOR OPTIONS";
6120 GOSUB 7000:GOTO 120:REM PAUSE TO READ SCREEN THEN GO
     TO OPTIONS
6130 REM
7000 ANSWER$="":POKE 764,255:INPUT ANSWER$:RETURN :REM 1
     CHARACTER INPUT
7010 REM
7100 GRAPHICS 0:SETCOLOR 2,0,0:REM DRIVE NUMBER AND
     FILENAME INPUT ROUTINE
7110 ? :? "TYPE DISK DRIVE NUMBER (1-4)";:HIGHNUMBER=4:
     GOSUB 7000
7120 LINE=7110:TRAP 8000:NUMBER=VAL(ANSWER$):TRAP 9100
7130 IF NUMBER<1 OR NUMBER>4 THEN 8000
7140 DRIVE$="D":DRIVE$(LEN(DRIVE$)+1)=ANSWER$
7150 DRIVE$(LEN(DRIVE$)+1)=":"
7200 ? :? "TYPE FILE NAME";:INPUT FILE$:IF LEN(FILE$)=0
     THEN 7200
7210 DRIVEFILE$=DRIVE$
7220 DRIVEFILE$(LEN(DRIVEFILE$)+1)=FILE$:RETURN
7230 REM
8000 ? :? "PLEASE TYPE A NUMBER FROM 1 THRU ";HIGHNUMBER:
     REM ERROR ROUTINE
8010 GOSUB 9000:GOTO LINE:REM GO BACK TO LINE NUMBER
     (LINE)
9000 ? CHR$(253):REM RING ERROR BELL
9010 FOR COUNT=1 TO 300:NEXT COUNT:RETURN
9020 REM
9100 POKE 752,0:IF PEEK(195)=136 THEN GOTO LINE:REM ERROR
     WAS END OF FILE
9110 REM DISPLAY ERROR NUMBER AND LINE AT WHICH ERROR
     OCCURRED THEN END
9120 ? :? " ERROR ";PEEK(195);" AT LINE ";PEEK(186)+
     PEEK(187)*256

9130 LIST PEEK(186)+PEEK(187)*256:GOSUB 9000
9140 TRAP 40000:GRAPHICS 0:POKE 764,255:POKE 752,1:POKE
     201,10
9150 RUN
```

```
  1 REM
  9 REM [SET TEXT MODE, BLACK & WHITE, MARGINS, PRINTTAB WIDTH, NO
    CURSOR.]
 10 GRAPHICS 0:SETCOLOR 2,0,0:POKE 82,2:POKE 83,39:POKE 201,10:POKE
    752,1:?
 20 ? "FORMAT DISK ON DR.1":? :? "         PRESS START WHEN READY.":
    GOTO 80
 30 REM [STORE CURRENT SUCCESSFUL FORMATS+1 THEN SAY WHAT WE ARE UP
    TO.]
 40 F=F+1:? CHR$(125):? :? :? :? "FORMATTING DISK #";F
 50 TRAP 200:XIO 254,#1,0,0,"D1:":TRAP 40000:REM [SET DRIVE# & START
    FORMATTING]
 55 POKE 66,1:? CHR$(253):POKE 66,0:REM [RING THE BELL BUT NOT TOO
    LOUD]
 60 ? "DISK FORMATTED":? :POKE 77,0:REM [DISPLAY RESULT & KILL
    ATTRACT MODE]
 70 ? " PRESS START TO FORMAT ANOTHER DISK":? " PRESS OPTION TO END
    THIS PROGRAM"
 80 IF PEEK(53279)=6 THEN 40:REM INSTANT REPLAY.
 90 IF PEEK(53279)=3 THEN 300:REM THAT'S ALL FOLKS.
100 GOTO 80:REM [WHATCHA WANT? PRESSA BUTTON!]
180 REM [A TRAP TO LINE 200 WAS SET IN LINE 50 FOR UNFORMATTABLE
    DISKS]
190 REM [ABORT BY TAPPING THE BREAK KEY (IN MOST CASES)]
195 REM [YOU MIGHT HAVE TO REMOVE THE DISK, THEN TURN THE DRIVE OFF &
    ON]
200 ? CHR$(253):? :? "I WAS UNABLE TO FORMAT THAT DISKETTE":?
210 BAD=BAD+1:F=F-1:GOTO 70:REM [ONE LESS SUCCESSFUL FORMAT]
300 ? CHR$(125):? :? F;" DISKS FORMATTED":? :? "UNABLE TO FORMAT
    ";BAD;" DISK(S)"
310 REM [SAY WHAT WE ACCOMPLISHED THEN WAIT FOR THE USER TO PRESS A
    KEY]
320 ? :? "          END OF PROGRAM FORMAT1":? :? "         PRESS ANY KEY
    FOR MENU":POKE 764,255
330 IF PEEK(764)=255 AND PEEK(53279)=7 THEN 330
340 POKE 764,255:RUN
```



```
110 DIM A$(128),FN$(14):FN$="D1:AUTORUN.SYS":GOTO 280
120 FOR ME=1 TO 88:READ IT:PUT #1,IT:NEXT ME:RETURN
130 FOR ME=1 TO 20:READ IT:PUT #1,IT:NEXT ME:RETURN
140 FOR I=1 TO 123:READ D:IF I=64 THEN PUT #1,LEN(A$)-1:
    GOTO 160
150 PUT #1,D
160 NEXT I
170 FOR I=LEN(A$) TO 1 STEP -1:PUT #1,ASC(A$(I,I)):NEXT I
180 PUT #1,255:PUT #1,255:PUT #1,226:PUT #1,2:PUT #1,227:
    PUT #1,2:PUT #1,0:PUT #1,6
190 ? :? "  CLOSING  ";FN$:CLOSE #1:POKE 82,2:POKE 83,39:?
    :POKE 752,0:END
200 DATA 255,255,0,6,5,6,169,148,141,197,2,96,255,255,226,
    2,227,2,0,6
210 DATA 255,255,0,56,75,56,169,80,141,0,3,169,1,141,1,3,
    169,63,141,2,3,169,64,141,3,3,169,5,141,6
220 DATA 3,141,5,3,169,0,141,4,3,141,9,3,141,10,3,141,11,
    3,169,12,141,8,3,32,89,228,48,1,96,162
230 DATA 11,189,0,5,157,0,3,202,16,247,32,89,228,48,6,32,
    6,5,108,12,0,96,226,2,227,2,0,56
240 DATA 162,0,189,26,3,201,69,240,5,232,232,232,208,244,
    232,142,105,6,189,26,3,133,205,189,107,157,26,3,232,
    189
250 DATA 26,3,133,206,169,6,157,26,3,160,0,162,16,177,205,
    153,107,6,200,202,208,247,169,67,141,111,6,169,6,141
260 DATA 112,6,169,10,141,106,6,96,172,106,6,48,9,185,123,
    6,206,106,6,160,1,96,138,72,174,105,6,165,205,157
270 DATA 26,3,232,165,206,157,26,3,104,170,169,155,160,1,
    96,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
280 GRAPHICS 0:POKE 710,144:POKE 82,1:POKE 83,38:POKE 201,
    10:? :? "  [AUTORUN.SYS CREATE PROGRAM FOR BASIC]"
290 ? "    [DOS VERSION 2]":? :? "
300 ? :? "  BOOT INTERFACE (Y OR N)?";:GOSUB 360
310 ? :? "  [ENTER COMMANDS:]":? :INPUT A$:POKE 752,1:IF
    LEN(A$)=0 THEN 310
320 ? :? "  OPENING  ";FN$:TRAP 600:? :? "  CREATING ";FN$
330 IF BOOT850 THEN RESTORE 200:GOSUB 130:RESTORE 210:
    GOSUB 120:GOTO 350
340 RESTORE 200:GOSUB 130
350 PUT #1,255:PUT #1,255:PUT #1,0:PUT #1,6:L=123+LEN(A$)-
    1:PUT #1,L:PUT #1,6:RESTORE 240:GOTO 140
360 POKE 764,255
370 IF PEEK(764)=255 THEN 370
380 IF PEEK(764)=43 THEN ? "Y":BOOT850=1:POKE 764,255:
    RETURN
390 IF PEEK(764)=35 THEN ? "N":BOOT850=0:POKE 764,255:
    RETURN
400 ? :? :? "
500 ? "@BELL@@PFEIL UNTEN@@PFEIL UNTEN@  PLEASE TYPE Y OR N?";:GOTO 360
600 ? "@BELL@@PFEIL UNTEN@@PFEIL UNTEN@ I WAS UNABLE TO
    OPEN AUTORUN.SYS":GOTO 900
900 ? "@BELL@@PFEIL UNTEN@@PFEIL UNTEN@ ERROR WHILE
    WRITING AUTORUN.SYS"
    ? :? "ERROR NUMBER ";PEEK(195):POKE 201,8:? :?
    "PRESS [OPTION] TO RERUN":? ,"PRESS [START ] FOR'
    DOS";
910 IF PEEK(53279)=3 THEN RUN
920 IF PEEK(53279)=6 THEN CLOSE #1:DOS
930 GOTO 910
```

```
  0 REM [RPMTEST] 4/29/82
100 GRAPHICS 0:POKE 559,0:FOR I=0 TO 72:READ A:POKE 1536+
    I,A:NEXT I:POKE 710,0:POKE 82,1:POKE 201,10
120 ?"@CLEAR@":? "[{@}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}
    {R}{R}{R}{R}{E}]":? "[|  R P M  T E S T  |]":? "[{Z}
    {R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{R}{C}]":
    POKE 559,34:POKE 752,0
150 ? "@PFEIL UNTEN@@PFEIL UNTEN@@PFEIL UNTEN@@TAB@TEST
    WHICH DRIVE NUMBER";:TRAP 150:INPUT DRIVE:TRAP 40000
160 DRIVE=INT(DRIVE):IF DRIVE<1 OR DRIVE>4 THEN ? :? "
    ENTER DRIVE NUMBER 1, 2, 3, OR 4":GOTO 150
180 POKE 752,1:? :? "      TESTING 100 DISK REVOLUTIONS":?
    :? "   THIS SHOULD TAKE APPROX. 22 SECONDS";
200 POKE 1610,DRIVE:X=USR(1536):A=PEEK(1611):B=PEEK(1612):
    MIN=(256*B+A)/3600:RPM=INT(100/MIN+0.5)
220 ? :? :? "@TAB@@TAB@[RPM=]";RPM:IF RPM>284 AND RPM<291
    THEN ? :? "@TAB@    DRIVE SPEED IS O.K.":POKE 710,178:
    GOTO 250
230 POKE 710,66:IF RPM<285 THEN ? :? "@TAB@DRIVE SPEED IS
    TOO SLOW":GOTO 250
240 ? :? "@TAB@DRIVE SPEED IS TOO FAST"
250 ? :? "PRESS [OPTION] FOR MENU OR [START] TO RERUN";
300 IF PEEK(53279)=6 THEN 120
350 IF PEEK(53279)=3 THEN POKE 82,2:? "@CLEAR@":STOP
400 GOTO 300
500 REM DATA FOR MACHINE LANGUAGE ROUTINE
1000 DATA 104,169,1,141,10,3,169,0
1010 DATA 141,11,3,141,4,3,169,5
1020 DATA 141,5,3,173,74,6,141,1
1030 DATA 3,169,82,141,2,3,169,5
1040 DATA 141,73,6,32,83,228,206,73
1050 DATA 6,208,248,169,100,141,73,6
1060 DATA 169,0,133,19,133,20,32,83
1070 DATA 228,206,73,6,208,248,165,20
1080 DATA 164,19,141,75,6,140,76,6,96
2000 REM
2010 REM THIS IS A MODIFIED VERSION OF THE RPMTEST PROGRAM
     ORIGINALLY PUBLISHED
2020 REM IN THE MAY 1982 ISSUE OF COMPUTE! MAGAZINE.
2030 REM WE WISH TO THANK BOB CHRISTIANSEN FOR HIS
     PERMISSION TO INCLUDE THIS
2040 REM PROGRAM AS PART OF THIS TUTORIAL/UTILITY PACKAGE.
```

```
100 REM
110 GOTO 940
120 REM
130 REM **  [DISPLAY SECTOR (CHARACTER FORMAT)] **
140 REM
150 FOR X=A TO B:IF SEC$(X,X)=CHR$(155) THEN ? " ";:GOTO
    170
160 ? SEC$(X,X);
170 NEXT X:RETURN
180 REM
190 REM ** [DISPLAY SECTOR (HEX FORMAT)] **
200 REM
210 FOR X=A TO B:BYTE=ASC(SEC$(X,X)):HX=INT(BYTE/16)+1:
    HEX=BYTE-16*(HX-1)+1:HEX$(1,1)=HEXSTR$(HX,HX)
220 HEX$(4,4)=HEXSTR$(HEX,HEX):? HEX$;:NEXT X:RETURN
230 REM
240 REM ** [EXAMINE DIRECTORY FILE] **
250 REM
260 POKE 764,255:POKE 82,2:? "@CLEAR@":? "[S FN FILE
    EXT     SSEC     NSEC]":?
270 POKE 779,1:FOR LOOP=1 TO 8:POKE 778,104+LOOP:IO=
    USR(ADR(CIO$))
280 FOR ENTRY=1 TO 8:STATE$=SEC$(ENTRY*16-15,ENTRY*16-14):
    IF STATE$="( )( )" THEN 340
290 IF STATE$(1,1)="[( )]" THEN ? "D ";:GOTO 320
300 LOCK=ASC(STATE$(1,1)):LOCK=LOCK-INT(LOCK/64)*64:IF
    LOCK>=32 THEN ? "L ";:GOTO 320
310 ? "  ";
320 ? 8*(LOOP-1)+ENTRY-1;"@TAB@";SEC$(ENTRY*16-10,ENTRY*
    16);"@TAB@";
330 ? ASC(SEC$(ENTRY*16-12))+256*ASC(SEC$(ENTRY*16-
    *16-11));"@TAB@";ASC(SEC$(ENTRY*16-14))+256*
    ASC(SEC$(ENTRY*16-13))
340 NEXT ENTRY:NEXT LOOP:GOTO 750
350 REM
360 REM ** [EXAMINE SECTOR] **
370 REM
380 POKE 764,255:GOSUB 820:POKE 82,1
390 ? "@CLEAR@[SECTOR NUMBER] ";NUM:? :IO=USR(ADR(CIO$)):
    IF C=1 THEN POKE 766,1
400 ? "[BYTE# 000000000011111111112222222222333]":? "[
     0123456789012345678901234567890123456789012]"
410 ? "[      ";:A=1:B=32:IF C=1 THEN GOSUB 150:GOTO 430
420 GOSUB 210
430 ? :? :? :? "[BYTE# 333333334444444444555555555566666]":
    ? "[      234567890123456789012345678901231]"
440 ? "[      ";:A=33:B=64:IF C=1 THEN GOSUB 150:GOTO 460
450 GOSUB 210
460 ? :? :? :? "[BYTE# 666666777777777788888888889999999]":
    ? "[      4567890123456789012345678901234567]"
470 ? "[      ";:A=65:B=96:IF C=1 THEN GOSUB 150:GOTO 490
480 GOSUB 210
490 ? :? :? :? "[BYTE#      11111111111111111111111111111111]"
500 ? "[       99999000000000011111111112222222]"
510 ? "[       67890123456789012345678901234567]"
520 ? "[      ";:A=97:B=128:IF C=1 THEN GOSUB 150:GOTO 540
530 GOSUB 210
540 POKE 766,0:? :GOTO 750
550 REM
560 REM ** [EXAMINE FILE] **
570 REM
580 POKE 764,255:TRAP 580:? "@CLEAR@":? :? "      ENTER
    FILENAME";:INPUT USER$:FILE$=DRIVE$:FILE$(LEN(FILE$)+
    1)=USER$
590 TRAP 690:CLOSE #1:OPEN #1,4,0,FILE$:COUNT=0:SCOUNT=0:
    POKE 201,10:POKE 82,3:POKE 83,39:FLAG=1
600 POKE 752,1:POKE 766,0:? "@CLEAR@":POKE 766,1:? "[BYTE
    DEC VALUE  HEX VALUE   CHAR]"
610 GET #1,BYTE:COUNT=COUNT+1:SCOUNT=SCOUNT+1:HX=
    INT(BYTE/16)+1:HEX=BYTE-16*(HX-1)+1:HEX$(1,1)=
    HEXSTR$(HX,HX)
620 HEX$(4,4)=HEXSTR$(HEX,HEX):? "  ";COUNT,BYTE,HEX$(1,
```

```
    1);HEX$(4,4);:IF BYTE=155 THEN BYTE=32
630 ? #6;CHR$(BYTE)
640 IF SCOUNT=20 THEN GOSUB 1230:SCOUNT=0:GOTO 600
650 GOTO 610
660 REM
670 REM ** [TRAP ERRORS] **
680 REM
690 POKE 66,0:ERROR=PEEK(195):IF ERROR=136 THEN ? :? ,
    "END OF FILE":? :? :GOSUB 750:RUN
700 IF ERROR=170 THEN ? :? ,"FILE NOT FOUND":? :? :GOTO
    750
710 ? :? ,"ERROR ";ERROR;" AT LINE ";PEEK(186)+PEEK(187)*
    256:?':? :GOTO 750
720 REM
730 REM ** [TEMPORARY HALT] **
740 REM
750 POSITION 7,23:? "[PRESS ANY KEY FOR OPTIONS]";:POKE
    764,255
760 IF PEEK(764)=255 THEN 760
770 IF FLAG THEN RETURN
780 POKE 764,255:GOTO 1040
790 REM
800 REM ** [INPUT SECTOR NUMBER & FORMAT] **
810 REM
820 TRAP 820:? :? "     TYPE SECTOR NUMBER [RETURN]";:
    INPUT NUM:TRAP 40000
830 NUM=INT(NUM):IF NUM<1 OR NUM>720 THEN ? :? ,"[ENTER 1
    THRU 720]":GOTO 820
840 ? :? "[TYPE 1 FOR CHARACTER OR 2 FOR HEXDUMP]";:POKE
    764,255
850 LASTKEY=PEEK(764):IF LASTKEY=255 THEN 850
860 IF LASTKEY=31 THEN C=1:GOTO 890
870 IF LASTKEY=30 THEN C=2:GOTO 890
880 ? :GOTO 840
890 POKE 778,NUM-INT(NUM/256)*256
900 POKE 779,INT(NUM/256):RETURN
910 REM
920 REM ** [PROGRAM SETUP] **
930 REM
940 DIM CIO$(5),SEC$(128),STATE$(2),HEX$(5),HEXSTR$(16):
    CIO$="h S[4](."
950 HEX$=" @PFEIL UNTEN@@PFEIL LINKS@ @PFEIL OBEN@":
    HEXSTR$="0123456789ABCDEF"
960 DIM DRIVE$(3),USER$(12),FILE$(15):DRIVE$="D1:"
970 POKE 772,ADR(SEC$)-INT(ADR(SEC$)/256)*256:REM LOW
    BYTE
980 POKE 773,INT(ADR(SEC$)/256):REM HIGH BYTE
990 POKE 769,1:REM DRIVE 1
1000 POKE 770,82:REM READ SECTOR (87=WRITE SECTOR)
1010 REM
1020 REM ** [WHATCHA WANNA DO] **
1030 REM
1040 GRAPHICS 0:POKE 710,144:POKE 201,9:POKE 752,1:POKE 82,
     2:POKE 83,39:FLAG=0:POKE 766,0
1050 ? :? "(Q)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)
     (R)(R)(E)"
1060 ? ,"| DISK INSPECTOR |"
1070 ? ,"(Z)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)(R)
     (R)(C)"
1080 ? :? :? ,"[TYPE OPTION NUMBER]"
1090 ? :? ,"1= EXAMINE DIRECTORY"
1100 ? :? ,"2= EXAMINE SECTOR"
1110 ? :? ,"3= EXAMINE FILE"
1120 ? :? ,"4= RUN MENU"
1130 POKE 764,255:SEC$(1)=" ":SEC$(128)=" ":SEC$(2)=
     SEC$(1)
1140 LASTKEY=PEEK(764):IF LASTKEY=255 THEN 1140
1150 IF LASTKEY=31 THEN 240
1160 IF LASTKEY=30 THEN 360
1170 IF LASTKEY=24 THEN POKE 752,0:GOTO 560
1180 IF LASTKEY=24 THEN POKE 201,10:GRAPHICS 0:POKE 752,1:
     POKE 764,255:? :? :STOP
1190 GOTO 1130
1200 REM
1210 REM ** [START OR OPTION?] **
1220 REM
1230 POSITION 5,23:? "[OPTION=OPTIONS  START=CONTINUE]";
1240 IF PEEK(53279)=3 THEN RUN
1250 IF PEEK(53279)=6 THEN RETURN
1260 GOTO 1240
```