Markt&Technik

Für GEOS 64

BOOK-
WARE

GeoBasic

GEOS-Programmierung leichtgemacht

BERKELEY
Softworks

Auf Diskette enthalten:
Basic-Interpreter mit Hilfsprogrammen, Konvertierungsprogramm
und Debugger zur Programmierung von GEOS-Applikationen

BOOK-
WARE

GeoBasic

GEOS-Programmierung leichtgemacht

Markt&Technik
Verlag AG

CIP-Titelaufnahme der Deutschen Bibliothek

GeoBasic : GEOS-Programmierung leichtgemacht ; für GEOS 64. –
Haar bei München : Markt-u.-Technik-Verl., 1990
(Bookware)
ISBN 3-89090-245-6

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1

93 92 91 90

ISBN 3-89090-245-6

Einbandgestaltung: Grafikdesign Heinz Rauner
Dieses Produkt wurde mit Desktop-Publishing-Programmen erstellt
und auf der Linotronic 300 belichtet.
Druck: Wiener Verlag
Printed in Austria

Inhaltsverzeichnis

4. Kapitel: Der Basic-Lernkurs

5. Kapitel: Programmierung für Fortgeschrittene

9. Kapitel: Der Piktogrammlisten-Editor

10. Kapitel: Der Grafik-Editor

11. Kapitel: Der Sprite-Editor

Vorwort

Als GEOS, die grafische Benutzeroberfläche für den C 64 und C 128, Anfang 1986 vorgestellt wurde, belächelte man es vielerseits.

Dank der außergewöhnlichen Programmierleistung, die in GEOS steckt und nicht zuletzt auch wegen der vielen jetzt erhältlichen Erweiterungen (GeoFile, GeoCalc usw.), schaffte es die Benutzeroberfläche aber doch, sich in der Beliebtheitsskala der C 64- und C 128-Anwender hochzukatapultieren.

Inzwischen sind bereits weit über eine Million Exemplare von GEOS und GEOS 128 verkauft und GEOS 2.0 und GEOS 128 2.0 haben eine neue, noch leistungsstärkere GEOS-Ära eingeläutet.

Schon früh in der »Geschichte« von GEOS wurde verständlicherweise nach zugehörigen Programmiersprachen verlangt. Diesem Wunsch konnte Berkeley Softworks bis jetzt aber nur mit dem Entwicklungspaket »GeoPROGRAMMER« – bestimmt für Assembler-Programmierer – entsprechen, was verständlicherweise keine Dauerlösung sein konnte.

Mit dem Produkt GeoBasic halten Sie nun einen kompletten Basic-Interpreter zur Benutzung mit einer beliebigen GEOS-Version in Händen. Einfach zu erlernende Befehle und Funktionen decken das gesamte Spektrum der GEOS-Funktionen ab. GeoBasic ermöglicht Ihnen die Programmierung eigener Applikationen im Stil von GeoPaint und GeoWrite.

Dieses Buch soll Ihnen helfen, die vielen Funktionen von GeoBasic nutzen zu lernen. Es ist wie folgt aufgebaut:

In Kapitel 1 wird das Erstellen von Sicherheitskopie und Arbeitsdisketten erläutert.

Kapitel 2 vermittelt Ihnen die allgemeinen Begriffe und Regeln der Basic-Programmierung und die diesbezüglichen Besonderheiten von GeoBasic.

Die Bedienung von GeoBasic und die Funktionen des Text-Editors werden ausführlich von Kapitel 3 behandelt.

Einen GeoBasic-Lernkurs finden Sie in Kapitel 4. Hier werden die grundlegenden Befehle und Funktionen ausführlich anhand von Beispielen erklärt.

Kapitel 5 schließlich erläutert die Befehle und Funktionen, die die Besonderheiten der GEOS-Benutzeroberfläche und des C 64 und C 128 unterstützen. Außerdem wird Ihnen die Programmierung von GEOS-Applikationen anhand eines Beispiels erklärt. Sie sollten sich dieses Kapitel erst dann zu Gemüte führen, wenn Sie in der Benutzung der grundlegenden Basic-Befehle (Kapitel 4) wirklich sicher sind.

Mit GeoBasic können auch Applikationen erstellt werden, die anschließend selbst ablauffähig sind und vom Desktop zu starten sind. Kapitel 6 beschreibt alle Schritte zur Ausnutzung dieser interessanten Funktion.

Die Kapitel 7 bis 11 behandeln die Funktionen der fünf in GeoBasic integrierten Hilfsprogramme (Menü-Editor, Dialogbox-Editor, Piktogrammlisten-Editor, Grafik-Editor und Sprite-Editor).

Der GeoBasic-Debugger und die sich durch ihn eröffnenden Möglichkeiten der Fehlersuche in Ihren eigenen Programmen werden ausführlich in Kapitel 12 beschrieben.

Kapitel 13 erläutert die Applikation Basic-Grabber, mit der Sie Commodore-Basic-Programme zu GeoBasic-Programmen konvertieren können.

In Kapitel 14 finden Sie den Anhang, unter anderem ein Glossar, das eventuell unbekannte Wörter erklärt, und viele zusätzliche Informationen zu GeoBasic und der Erstellung von eigenen Programmen.

Einführung

Basic ist eine Programmiersprache für Ihren Computer. Die Abkürzung »Basic« steht für Beginner's All Purpose Symbolic Instruction Code, was schon viel über den Sinn und die Aufgabe dieser Sprache aussagt: sie soll Anfängern (Beginner) die Erstellung von Programmen mit Anweisungen (Instruction Code) für jeden beliebigen Zweck (All Purpose) erleichtern.

In der Tat ist Basic recht einfach zu erlernen und bietet dennoch umfangreiche Möglichkeiten zur Programmierung und Ausnutzung der Fähigkeiten Ihres Computers. Sie müssen nicht jahrelang studiert haben oder irgendwelche Erfahrungen im Bereich Programmierung mitbringen, um unter Basic ansprechende Programme erstellen zu können. Und obwohl diese Programmiersprache für Anfänger prädestiniert ist, kann sogar ein Profi noch viel aus ihr herausholen und sich so beispielsweise aufwendige Arbeit in Assembler, einer wesentlich schwierigeren Programmiersprache, ersparen.

Um die Vorteile von Basic auch für alle nutzbar zu machen, die das Betriebssystem GEOS auf dem C 64 oder C 128 verwenden, wurde GeoBasic entwickelt.

Jetzt kann sich jeder selbst »seine« GEOS-Applikationen erstellen und alle Ideen und Wünsche verwirklichen, die im Laufe der Zeit entstanden sind. Und das mit Abrollmenüs, Dialogboxen, Piktogrammen, Zeichensätzen, eigenen Sprites, und ... und ... und ...

GeoBasic gibt Ihnen alle Mittel in die Hand, um Ihre Programme so zu gestalten, daß sie gleichzeitig einfach und leistungsstark sind – genau wie GEOS und die anderen Applikationen auch.

Das GeoBasic-Programmpaket bietet Ihnen

- über 100 Befehle und Funktionen zur Unterstützung von Abrollmenüs, Piktogrammen, Dialogboxen, Grafiken, Maus, Sprites, Labels, Variablen, Schleifenprogrammierung, Disketten- und Dateizugriffen, Unterroutinen, Berechnungen, zum Aufrufen von GEOS-Maschinenroutinen und zur Nutzung von vielen anderen wichtigen Dingen in Ihren eigenen Programmen,

- einen schnellen und umfangreichen Text-Editor zur Eingabe und Bearbeitung Ihrer Programme,
- eine Menüleiste mit vielen nützlichen Funktionen (z. B. Ändern der Numerierung oder Debugging),
- fünf Hilfsprogramme zur einfachen Erstellung der in Programmen verwendbaren Menüs, Dialogboxen, Grafiken, Piktogramme und Sprites sowie
- einen Basic-Grabber zur Übersetzung von Standard-Commodore-Basic-Programmen in GeoBasic-Programme.

1. Kapitel
Bevor Sie mit GeoBasic arbeiten

Sie sollten vor der ersten Nutzung von GeoBasic dieses Kapitel durcharbeiten. Es erläutert Ihnen die zum Betreiben notwendige Hard- und Software und teilt Ihnen mit, was Sie möglichst erledigen sollten, bevor Sie mit GeoBasic arbeiten, um eventuell später Schwierigkeiten durch Diskettenfehler zu vermeiden.

1.1 Benötigte Hard- und Software

Um mit GeoBasic arbeiten zu können, müssen Sie über folgende Hard- und Software verfügen:

- ein Commodore 64, 64c oder 128, ein Diskettenlaufwerk 1541, 1570 oder 1571, ein beliebiger Monitor, ein Eingabegerät (Joystick oder Maus) und ein GEOS-Grundsystem (Version 1.2 oder höher).

Zur Steigerung der Leistungsfähigkeit und Geschwindigkeit von GeoBasic und Ihrer gesamten sonstigen GEOS-Software ist folgende optionale Hardware empfehlenswert:

- ein GEOS-unterstützter Drucker (schlagen Sie notfalls in Ihrem GEOS-Benutzerhandbuch nach, ob Ihr Drucker unterstützt wird), ein Zusatzlaufwerk 1541, 1570 oder 1571 (ab GEOS 2.0 auch 1581) und eine Commodore-RAM-Erweiterung 1750 oder 1764.

Die Nutzung einer der oben aufgeführten RAM-Erweiterungen ist sehr empfehlenswert, da sich dann bei Verwendung als RAM-Laufwerk die Zugriffszeiten auf Diskette durch GeoBasic stark verkürzen lassen.

Die Commodore-RAM-Erweiterung 1700 bietet wegen ihrer eingeschränkten Speicherkapazität (128 Kbyte) kaum Vorteile bei der Benutzung mit einem GEOS-System.

Hinweis: GeoBasic ist nicht für die Benutzung unter einer GEOS-128-Version gedacht; es können Fehlfunktionen auftreten.

1.2 Anfertigen einer Sicherheitskopie

Um Probleme mit Diskettenfehlern zu vermeiden, sollten Sie eine Sicherheitskopie Ihrer GeoBasic-Originaldiskette anfertigen.

Hinweis: Schützen Sie vor dem Kopieren Ihre GeoBasic-Originaldiskette mit Schreibschutzaufklebern gegen unbeabsichtigtes Löschen.

Wenn Sie nur ein Laufwerk besitzen:
Starten Sie das Programm DISKETTENKOPIER (im GEOS-Grundsystem enthalten), und folgen Sie den Anweisungen auf dem Bildschirm. Die Quelldiskette ist dabei immer Ihre GeoBasic-Originaldiskette und die Zieldiskette die Leerdiskette, die später die Sicherheitskopie enthalten soll.

Eine andere Möglichkeit besteht darin, unter Desktop die GeoBasic-Diskette zu öffnen und aus dem »Diskette«-Menü »Kopieren« auszuwählen. Folgen Sie dann den Anweisungen auf dem Bildschirm.

Wenn Sie zwei Laufwerke besitzen:
Legen Sie unter Desktop die GeoBasic-Originaldiskette in das eine Laufwerk und die Leerdiskette für die Sicherheitskopie in das andere. Öffnen Sie nun die GeoBasic-Originaldiskette und wählen Sie aus dem »Diskette«-Menü »Kopieren«. Nun müssen Sie nur noch den Anweisungen der Dialogboxen folgen.

Wenn Sie ein Laufwerk und ein RAM-Laufwerk benutzen:
Öffnen Sie unter Desktop die GeoBasic-Originaldiskette und kopieren Sie diese (»Kopieren« aus dem »Diskette«-Menü) in das RAM-Laufwerk. Anschließend tauschen Sie die GeoBasic-Originaldiskette gegen die Leerdiskette für die Sicherheitskopie aus und kopieren den Inhalt des RAM-Laufwerks darauf. Wenn Sie während des Kopiervorgangs immer den Anweisungen auf dem Bildschirm nachkommen, kann nichts schiefgehen.

Wenn Sie dann mit der zweiten Seite des Originals ebenso verfahren, besitzen Sie eine vollständige Sicherheitskopie. Ihre GeoBasic-Originaldiskette können Sie ab jetzt getrost an einem sicheren Platz aufbewahren.

1.3 Anfertigen von Arbeitsdisketten

Nun ist es an der Zeit, Arbeitsdisketten anzufertigen, auf denen Sie (wie es der Name schon sagt) mit GeoBasic arbeiten können.
Kopieren Sie einfach den Desktop und GeoBasic (aber nicht von der Originaldiskette, sondern von der Sicherheitskopie) auf eine leere Diskette. Fertig ist Ihre erste Arbeitsdiskette!

Hinweis: Haben Sie noch Schwierigkeiten mit dem Kopieren von einzelnen Dateien, können Sie dies in Ihrem GEOS-Benutzerhandbuch nachschlagen.

Folgende Dateien können Sie ebenfalls auf Ihre Arbeitsdisketten kopieren:

- Basic-Grabber (wenn Sie Ihre alten Commodore-Basic-Programme in das GeoBasic-Format konvertieren möchten),
- Druckertreiber (für Ausdrucke) und
- Zeichensätze (zur Verwendung in eigenen GeoBasic-Programmen).

Sie sollten aber immer beachten, daß zusätzliche Dateien, die Sie nicht unbedingt benötigen, und die trotzdem auf der Diskette stehen, Ihnen wertvollen Speicherplatz stehlen, den Sie vielleicht später noch für das Programmieren dringend benötigen.

2. Kapitel
Grundbegriffe der GeoBasic-Programmierung

Dieses Kapitel ist für alle Personen gedacht, die mit GeoBasic ihre ersten Schritte in der Programmierung eines Computers in der Programmiersprache Basic unternehmen möchten. Es erklärt die wichtigsten Grundregeln und Begriffe der Basic-Programmierung, soll Ihnen aber noch kein Wissen über die vielen verfügbaren Befehle und Funktionen vermitteln; dafür sind die Kapitel 4 und 5 konzipiert.

Beim Durcharbeiten dieses Kapitels ist es übrigens nicht erforderlich, GeoBasic zu starten und alle Beispiele auszuprobieren. Haben Sie aber trotz allem den Wunsch, die Beispiele zu erproben, empfehle ich Ihnen, vorher in Kapitel 3 die Abschnitte über das Starten von GeoBasic und die Funktionen des Text-Editors durchzuarbeiten.

2.1 Aufbau von Basic-Befehlen und Funktionen

Der Sprachumfang von GeoBasic teilt sich in

- Befehle und Anweisungen und
- Funktionen

auf. »Befehle und Anweisungen« (ab jetzt nur noch Befehle genannt) sind die eigentlichen Bestandteile von Programmen und steuern dessen Ablauf (z. B. Bildschirmausgabe, Schleifen oder Programmsprünge). Einige Befehle können aber auch im Text-Editor zur Programmeditierung verwendet werden.

»Funktionen« stellen dem Programm bestimmte Werte und Systemzustände zur Verfügung, die vom Programm weiterverarbeitet werden können (z. B. Logarithmen oder Zeichenkettenverknüpfungen).

Wie Sie bereits erfahren haben, gibt es in GeoBasic mehr als 100 Befehle und Funktionen. In ihrer Schreibweise sind sie an die englische Sprache angelehnt, wie zum Beispiel der Befehl RECT für »RECTangle« steht, was im Deutschen »Rechteck« bedeutet. Dadurch können Sie auf die Wirkungsweise dieses Befehls schließen: er zeichnet Rechtecke auf den Bildschirm. Jemand, der ein wenig Englisch gelernt hat, dürfte keine Probleme haben, sich die Namen der

Befehle und Funktionen auf Anhieb merken zu können. Und alle, die kein Englisch gelernt haben, benötigen sicher auch nicht sehr viel mehr Zeit. Lassen Sie sich jetzt also nicht entmutigen, weil Sie kein Englisch können!

Kommen wir nun zum grundsätzlichen Aufbau (Syntax) der Befehle:

BEFEHLSWORT PARAMETERLISTE

BEFEHLSWORT steht für das eigentliche Befehlswort, wie z. B. PRINT (zur Ausgabe von Texten und Zahlen auf dem Bildschirm). Als Parameterliste bezeichnet man alle auf das Befehlswort folgenden Angaben, die den Befehl genauer beschreiben (definieren). Bei dem Befehlswort PRINT könnte das z. B. eine Zeichenkette sein: "GEOS läßt sich mit GeoBasic einfach programmieren!". Der vollständige Befehl würde jetzt also lauten

```
PRINT "GEOS läßt sich mit GeoBasic einfach programmieren!"
```

und entspricht damit vollkommen der obengenannten Syntax.

Nehmen wir an, wir möchten ein Rechteck auf den Bildschirm mit den Anfangskoordinaten 10, 10 und den Endkoordinaten 309, 189 zeichnen. Der Befehl in unserem Programm müßte so aussehen:

```
RECT 10, 10, 309, 189
```

Müssen mehrere Parameter angegeben werden, so können sie beispielsweise durch Kommata getrennt werden. Dies variiert aber von Befehl zu Befehl. Parameter werden z. B. bei einigen Befehlen auch durch Semikolon (»;«) oder durch ein Wort (»TO«) getrennt.

Ebenso gibt es bei Funktionen einen einheitlichen Aufbau:

FUNKTIONSNAME(PARAMETERLISTE)

Unsere Beispielfunktion lautet vollständig so:

LOG(7)

Wichtig für die Nutzung von Funktionen in Basic-Programmen ist außerdem, daß sie nie alleine stehen können. Sie müssen immer in Zusammenhang mit Befehlen gesetzt werden, beispielsweise so:

```
PRINT LOG(7)      -> als Parameter zu einem Befehl
A = LOG(7) + 9    -> als Bestandteil einer Berechnung
```

Dies sollten Sie bei der Verwendung von Funktionen immer beachten!

Hinweis: In GeoBasic-Programmen erscheinen alle Befehle und Funktionen automatisch in Großschrift. Dies erleichtert das Lesen der Programme ungemein. Natürlich müssen Sie diese Befehle und Funktionen nicht unbedingt in Großbuchstaben eingeben. GeoBasic erkennt jede beliebige Kombination aus Klein- und Großbuchstaben, sollte es sich dabei um einen Befehl oder eine Funktion handeln, und wandelt diese sofort in Großbuchstaben um.

2.2 Aufbau von Basic-Programmen

Vom Aufbau einzelner Befehle und Funktionen ist es jetzt nicht mehr weit bis zum Aufbau von ganzen Programmen. Betrachten wir einfach mal ein Beispiel:

```
10 PRINT "Hallo, wie geht es Ihnen?"
20 INPUT a$
30 IF a$="gut" THEN PRINT "Wunderbar!": END
40 PRINT "OK."
50 END
```

Wie Sie erkennen können, ist dies eine Ansammlung von Basic-Befehlen (die groß geschriebenen Wörter). Vor den Befehlen sind immer Nummern aufgeführt, und zwar fortlaufend in Zehner-Schritten. Eine solche Ansammlung von Befehlen nennt man »Programm«, wobei die Nummern vor den einzelnen Zeilen »Zeilennummern« heißen. Dieses Programm ist so aufgebaut:

ZEILENNUMMER BEFEHLE

ZEILENNUMMER BEFEHLE

ZEILENNUMMER ... ...

Und so ist auch jedes andere Programm aufgebaut. Anhand der Zeilennummer kann GeoBasic bei Ihrer Eingabe erkennen, an welcher Stelle im Programm die neue Zeile eingeordnet werden soll. Um auch nachträglich noch Zeilen einfügen zu können, ist dieses Programm in Zehner-Schritten, ausgehend von der Nummer 10, numeriert. Würden wir, wenn wir das Beispielprogramm eingegeben hätten, auch noch folgende Zeile eingeben:

```
45 PRINT "Hoffentlich geht es Ihnen bald wieder besser!"
```

würde das Programm so aussehen:

```
10 PRINT "Hallo, wie geht es Ihnen?"
20 INPUT a$
30 IF a$="gut" THEN PRINT "Wunderbar!": END
40 PRINT "OK."
45 PRINT "Hoffentlich geht es Ihnen bald wieder besser!"
50 END
```

Programme werden grundsätzlich in der Reihenfolge der Programmzeilen (von oben nach unten) abgearbeitet.

Mit Zeilennummern werden aber auch Sprünge von beliebigen Stellen an beliebige Stellen im Programm ermöglicht: Sie geben einfach nur nach einem Sprungbefehl (GOTO oder GOSUB) die gewünschte Zeilennummer an, und der Sprung kann ausgeführt werden.

Jede Zahl zwischen 1 und 65535 kann in GeoBasic als Zeilennummer benutzt werden.

Und was machen Sie nun, wenn Sie mehrere Befehle in eine Zeile packen möchten, sei es, weil dies nicht anders möglich ist oder weil es vielleicht besser verständlich ist? Schauen Sie sich dazu die Zeile 30 im Beispielprogramm noch einmal genau an. Dort steht der Befehl END (zum Beenden des Programms) hinter einem Doppelpunkt, vor dem ein IF-Befehl (zur Reaktion auf den eingegebenen Wert) zu sehen ist. Mehrere Befehle in einer Zeile trennen Sie also immer durch Doppelpunkte.

2.3 Die Verwendung von Labels

Wie wir im letzten Kapitel festgestellt haben, können sich in Programmen Sprungbefehle auf Zeilennummern beziehen. Eine wirkungsvollere Methode liegt in der Verwendung von Labels. Nehmen wir folgendes Programm:

```
10  INPUT "Welchen Computer besitzen Sie"; Co$
20  IF Co$="C64" THEN GOTO 100
30  IF Co$="C128" THEN GOTO 200
40  PRINT "Schwach!"
50  END
100 PRINT "Guter Computer!"
110 GOTO 10
200 PRINT "Spitzenmäßiger Computer!"
210 GOTO 10
```

Zugegeben, das Programm ist noch überschaubar und die Sprungbefehle (hier GOTO) auch gut nachvollziehbar. Haben Sie aber erst einmal ein Listing, das ausgedruckt nicht mehr auf eine DIN-A4-Seite paßt, kann das Verwirrspiel um die vielen GOTOs und GOSUBs leicht eskalieren. Dann sollten Sie spätestens Labels benutzen. Was aber sind Labels?

Labels sind Platzhalter und stehen für die Programmzeilen, in denen sie zu finden sind. Meist werden sie dann in Sprunganweisungen verwendet, wodurch sich die Lesbarkeit eines Programmes ganz beträchtlich erhöht.

Labels bestehen aus dem »@«-Zeichen (Shift + 3) sowie einem direkt darauf folgenden Wort (ohne Leerzeichen). Gültige Labels sind beispielsweise:

@schließen, @Ende, usw.

Nicht möglich sind dagegen folgende Varianten, da sie ein Leerzeichen enthalten:

@ schließen, @En de

Hinweis: Bitte beachten Sie die GEOS-Tastaturbelegung! (Abschnitt 14.7) Das auf der Tastatur abgebildete @-Zeichen ist in der GEOS-Tastaturbelegung das §-Zeichen. Im Text-Editor wird es als @-Zeichen dargestellt, während des Programmablaufs und in den Hilfsprogrammen erscheint es als §-Zeichen.

Versehen wir jetzt einfach unser Beispiel mit Labels:

```
10  @Start:
20  INPUT "Welchen Computer benutzen Sie"; Co$
30  IF Co$="C64" THEN GOTO @C64
40  IF Co$="C128" THEN GOTO @C128
50  PRINT "Schwach!"
60  END
100 @C64:
110 PRINT "Guter Computer!"
120 GOTO @Start
200 @C128:
210 PRINT "Spitzenmäßiger Computer!"
220 GOTO @Start
```

In diesem Listing fehlen bei allen GOTOs jetzt die Zeilennummern. Anstelle dessen sind Labels angegeben, unter denen Sie sich sicher mehr vorstellen können, als unter den »nackten« Zeilennummern.

Neben der verbesserten Lesbarkeit der Basic-Listings bieten Labels aber noch einen weiteren großen Vorteil. Sollten Sie eine Routine, die mit einem Label versehen ist, per Hand und nicht mit RENUMBER an eine andere Stelle im Programm verschieben, brauchen Sie nicht alle Sprungbefehle, die diese Routine aufrufen, mit einer neuen Zeilennummer zu versehen, da Sie das Label auch verschoben haben und dieses damit die neue Zeilennummer enthält.

Im Endeffekt bleibt es aber Ihnen überlassen, ob Sie Ihre Programme mit Labels versehen oder nicht. Haben Sie sich für die Nutzung von Labels entschieden, dann sollten Sie folgende Dinge immer beachten:

Das Wort hinter dem »@«-Zeichen darf zwar beliebig lang sein; beachtet werden aber nur die ersten sechs Buchstaben. Das Label »@Flaeche« ist also für GeoBasic mit dem Label »@Flaecheninhalt« identisch.

Versuchen Sie, zwei gleiche Labels in einem Programm zu verwenden, so wird ein »LABEL-REDEFINED«-Fehler ausgegeben. GeoBasic erkennt die folgenden Labels als verschiedene Labels an: »@Start« und »@start«, da sie sich in der Groß- und Kleinschreibung unterscheiden. Auch »@klickBehandlung« und »@klickbehandlung« werden aus diesem Grund beispielsweise als unterschiedlich ausgelegt.

Labels müssen immer als erstes nach der Zeilennummer in der Programmzeile stehen und werden (ebenso wie Befehle) durch einen Doppelpunkt von eventuell nachfolgenden Befehlen getrennt. Sie können Labels auch in Berechnungen einbauen. Ein »GOSUB @Unterroutinen * 100 + 5« ist also ebenso wie »a% = (@aktualisieren + 100) * 5« möglich. Sie dürfen maximal 127 verschiedene Labels in ein GeoBasic-Programm einbauen.

2.4 Was Sie tun und was Sie unterlassen sollten

Bevor wir zum eigentlichen Programmieren und zum Basic-Lernkurs kommen, möchte ich Ihnen noch ein paar Tips mit auf den Weg geben, die Sie immer berücksichtigen sollten.

1. Nutzen Sie Einrückungen, um Ihre Programme lesbarer zu machen.

FOR...NEXT-Schleifen, IF-Befehle und Unterroutinen können viel einfacher gelesen und verstanden werden, wenn Sie sie nach einem einfachen Schema einrücken. Dies ist besonders wirksam bei verschachtelten FOR...NEXT-Schleifen und bei IF-Befehlen, die länger als eine Bildschirmzeile sind.

```
10 FOR x = 1 TO 100
20   PRINT x
30   PRINT x + 5
40   FOR y = 1 TO 10
50     PRINT y
60     PRINT y + 2
70   NEXT y
80 NEXT x
```

oder:

```
10 IF umbenennen = 0 THEN
     neuName$ = "Beispiel":
     umbenennen = 1
```

Beachten Sie beim zweiten Beispiel bitte, daß es sich nur um eine Programmzeile handelt, die auf mehrere Bildschirmzeilen aufgeteilt ist. Obwohl hier die Befehle auf mehrere Bildschirmzeilen verteilt sind, müssen Sie noch durch Doppelpunkte getrennt werden!

2. Benutzen Sie so viele Labels wie möglich.

Labels sorgen dafür, daß das Programm lesbarer wird. Alle Unterroutinen (auch die Routinen, die von Menüs oder Piktogrammen aus angesprungen werden) sollten Labels erhalten. Dabei ist es anzuraten, immer Labelnamen zu benutzen, die die grundsätzliche Funktion der Routinen treffend beschreiben. Sie können sich unter »GOSUB @löschen« mehr vorstellen als unter »GOSUB @Label1« oder unter »GOSUB 2380«!

Bei Verwendung von Labels ist es außerdem einfacher, Routinen an andere Stellen im Programm zu verschieben. Sie müssen nicht alle Sprunganweisungen nachträglich ändern.

Sie sollten es vermeiden, hinter dem Label noch Befehle einzugeben:

```
10 @löschen:
20 FOR i=0 to 7
30 ...
```

3. Schreiben Sie leere Zeilen zwischen einzelne Routinen.

So können Sie einzelne Programmabschnitte besser auseinanderhalten. Um eine leere Programmzeile zu erhalten, geben Sie die Zeilennummer ein, drücken einmal auf die Leertaste und betätigen anschließend Return.

```
1000 @HandleClick:
1010 ....
1020 RETURN
1030
1040 @DoQuit:
1050 ...
```

4. Wählen Sie Variablennamen mit großen und kleinen Buchstaben.

Auch diese Maßnahme trägt wieder zur Lesbarkeit bei. Beispiele hierfür sind: geosVersion, Temporär und Fertig.

5. Vermeiden Sie mehrere Befehle in einer Programmzeile.

Es ergibt sich ansonsten der gefürchtete »Spaghetti«-Code, da die Programmzeilen sehr lang und vollgepackt sind. Sie können mehrere Befehle in einer Zeile aber nicht immer vermeiden, da beispielsweise alle von einem IF-Befehl abhängigen weiteren Befehle in einer Programmzeile stehen müssen. Dann bleibt Ihnen aber noch die Möglichkeit, die Befehle auf mehrere Bildschirmzeilen zu verteilen und einzurücken, wie es unter 1. beschrieben ist.

3. Kapitel
GeoBasic

Wenn Sie dieses Kapitel durchgearbeitet haben, wissen Sie, wie Sie GeoBasic ordnungsgemäß starten und den Text-Editor mit seinen gesamten Funktionen bedienen.

3.1 Starten von GeoBasic

Es gibt zwei Möglichkeiten, GeoBasic zu starten:

1. Einmal können Sie GeoBasic starten, indem Sie eine Arbeitsdiskette mit GeoBasic im Desktop öffnen und das Piktogramm, welches mit »GEOBASIC« untertitelt ist (Bild 3.1), doppelt anklicken.

Bild 3.1: Das GeoBasic-Piktogramm

Nach kurzer Zeit öffnet sich eine Dialogbox mit dem Startmenü (Abschnitt 3.2) auf dem Bildschirm.

2. Haben Sie schon ein GeoBasic-Programm geschrieben und möchten dies weiterbearbeiten, so können Sie auch unter Desktop das Piktogramm Ihres Programms (Bild 3.2) doppelt anklicken. GeoBasic wird dann automatisch nachgeladen, und Sie gelangen direkt in den Text-Editor (Abschnitt 3.3).

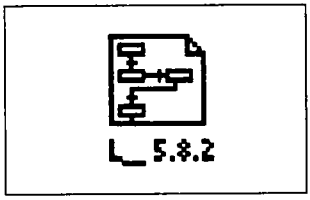


Bild 3.2: Das Piktogramm eines selbstgeschriebenen GeoBasic-Programms

3.2 Das GeoBasic-Startmenü

Das GeoBasic-Startmenü (Bild 3.3) stellt Ihnen folgende Auswahlmöglichkeiten zur Verfügung:

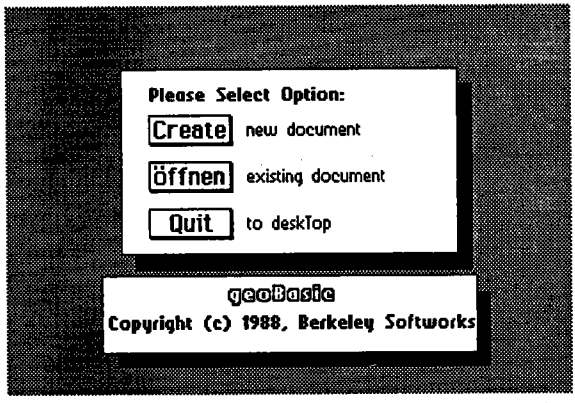


Bild 3.3: Das GeoBasic-Startmenü

3.2.1 »Create« new document (neues Dokument erstellen)

Klicken Sie »Create« an, wenn Sie ein vollkommen neues GeoBasic-Programm erstellen möchten. In einer Dialogbox (Bild 3.4) werden Sie aufgefordert, einen Dateinamen für Ihr Programm einzugeben.

Haben Sie den Namen eingegeben, müssen Sie [Return] drücken, um zum Text-Editor (Abschnitt 3.3) zu gelangen.

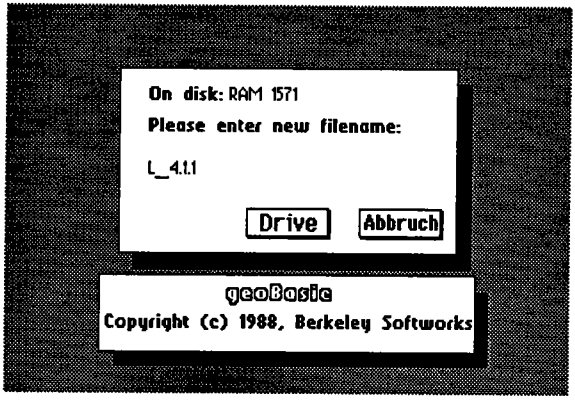


Bild 3.4: Erstellen eines neuen Programms

Eine Datei gleichen Namens darf auf der aktuellen Diskette noch nicht vorhanden sein. Ist dies aber der Fall oder tritt ein anderer Fehler auf, so wird dies in einer Dialogbox angezeigt.

Haben Sie »Create« ungewollt angewählt und möchten zum Startmenü zurück, dann klicken Sie einfach »Abbruch« an.

Möchten Sie auf einem anderen als dem aktuellen Laufwerk ein Programm erstellen, schalten Sie durch Anklicken von »Drive« zwischen den aktiven Laufwerken hin und her.

»Disk« ermöglicht Ihnen das Erstellen von Programmen auf anderen Disketten als im aktuellen Laufwerk. Nach dem Anklicken werden Sie aufgefordert, die Disketten auszutauschen. Legen Sie nun die gewünschte Diskette ein, und klicken Sie auf »OK«.

Hinweis: Das Wechseln von Disketten ohne ausdrückliche Aufforderung durch GeoBasic führt mit ziemlicher Sicherheit zu einem Absturz von GEOS und eventuell auch zu einem Verlust von wichtigen Daten!

3.2.2 »Öffnen« existing document (vorhandenes Dokument öffnen)

Wollen Sie ein bereits erstelltes Programm weiterbearbeiten, so wählen Sie »Öffnen« aus dem Startmenü aus. Eine Dialogbox (Bild 3.5) zeigt alle Programmnamen der aktuellen Diskette zur Auswahl an.

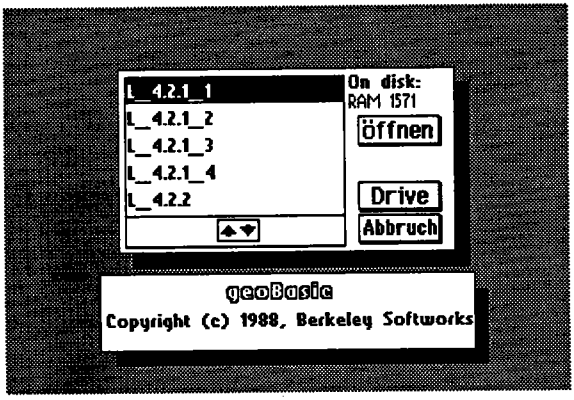


Bild 3.5: Auswählen eines vorhandenen Programms von der aktuellen Diskette

Passen nicht alle Programmnamen in die Dialogbox, erscheinen Rollpfeile, mit denen Sie weitere Namen in die Box rollen können.

Klicken Sie den Namen des gewünschten Programms einfach an. Die Auswahl bestätigt GeoBasic dadurch, daß der Name invers dargestellt wird. Anschließend klicken Sie auf »Öffnen«, um das Programm zu laden und in den Text-Editor (Abschnitt 3.3) zu gelangen. Haben Sie »Öffnen« aus dem Startmenü ungewollt angewählt, gelangen Sie mit »Abbruch« wieder dorthin zurück. Möchten Sie ein Programm von einer Diskette in einem anderen Laufwerk öffnen, so schalten Sie mit »Drive« zwischen den aktiven Laufwerken hin und her. »Disk« ermöglicht Ihnen das Öffnen von Programmen von einer Diskette, die derzeit in keinem Ihrer Laufwerke liegt. Nach Anklicken werden Sie zum Diskettenwechsel aufgefordert. Legen Sie nun die gewünschte Diskette ein und klicken Sie auf »OK«.

3.2.3 »Quit« to Desktop (zum Desktop verlassen)

Klicken Sie »Quit« an, wenn Sie die Arbeit mit GeoBasic beenden und zum Desktop zurückkehren möchten.

3.3 Der GeoBasic-Text-Editor

Während Ihre mit GeoBasic erstellten Programme ohne Ausnahme nur im hochauflösenden Grafikmodus ablaufen (wie z. B. GeoPaint und GeoWrite auch), steht Ihnen zur Eingabe und Bearbeitung Ihrer Programme ein leistungsstarker Editor im Text-Modus mit 24 Bildschirm-

zeilen zu je 40 Spalten (Editierfenster) zur Verfügung, der in seinem Aufbau und in seinen Funktionen an den Direktmodus des C 64 und C 128 (im 40-Spalten-Modus) erinnert. In der obersten Zeile finden Sie eine Menüleiste (Abschnitt 3.4) mit weiteren wichtigen Funktionen (siehe Bild 3.6).

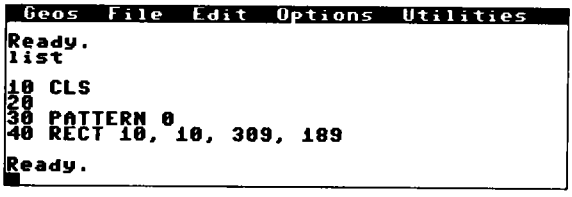


Bild 3.6: Der GeoBasic-Text-Editor

3.3.1 Der Cursor und Ihre Tastatureingaben

Das ständig blinkende Rechteck ist der »Cursor«, der sich mit den [Cursor]-Tasten unten rechts (bei GEOS 128 auch mit den zusätzlichen Tasten in der oberen Reihe) an jede beliebige Position im Editierfenster bringen läßt.

Klicken Sie mit dem Mauszeiger auf eine Bildschirmposition (nicht in der Menüleiste), so wird der Cursor automatisch an diese Stelle gesetzt. In die linke obere Ecke des Editierfensters läßt sich der Cursor am einfachsten mit der [Home]-Taste bringen. Jede Betätigung einer der alphanumerischen Tasten Ihres Computers führt dazu, daß das zugehörige Zeichen an der aktuellen Cursor-Position auf dem Bildschirm ausgegeben wird. Der Cursor bewegt sich danach um eine Spalte nach rechts. Sollte er dabei die 40. Spalte überschreiten, so wird er an den Anfang der nächsten Zeile gesetzt.

Die Großbuchstaben erreichen Sie, indem Sie gleichzeitig mit der jeweiligen Taste die [Shift]-Taste betätigen. Genauso können Sie auch die oberen Zeichen auf den mehrfach belegten Tasten ausgeben.

Die genaue Belegung der Tastatur unter GEOS und GeoBasic finden Sie in Abschnitt 14.7.

Hinweis: Manche Zeichen werden im Text-Editor anders angezeigt, als auf der Tastaturübersicht vorgesehen. So erscheinen z. B. die deutschen Umlaute und das »ß« als Grafikzeichen und das Paragraphenzeichen erscheint als Klammeraffe. Diese andere Anzeigeform betrifft aber nur den Text-Editor; während des Programmablaufs und in den Hilfsprogrammen erscheinen die Zeichen in ihrer korrekten Form.

3.3.2 Die Eingabemodi

Es gibt zwei verschiedene Eingabemodi im GeoBasic-Text-Editor: den »Überschreibe«- und den »Einfüge«-Modus. Sie schalten zwischen diesen Modi durch Drücken von Shift + Del um.

Der voreingestellte Modus ist der Überschreibe-Modus. Der Cursor ist hier immer ein blinkendes Rechteck. Geben Sie Text ein, dann wird dadurch der Text, der an der aktuellen Cursor-Position auf dem Bildschirm steht, einfach überschrieben.

Im Einfüge-Modus besteht der Cursor aus einem nicht blinkenden Rechteck. Bei einer Texteingabe werden alle Zeichen, die unter und rechts vom Cursor stehen, weiter nach rechts geschoben, um Platz für die neuen Zeichen zu schaffen. Passen Zeichen nicht mehr in die aktuelle Bildschirmzeile, werden sie einfach in die nächste geschoben.

3.3.3 Besondere Tastenfunktionen

Betätigen Sie die Return-Taste, so signalisieren Sie GeoBasic, daß Sie einen Befehl oder eine Programmzeile eingegeben haben. Die Programmzeile wird bei korrekter Eingabe in den Speicher übernommen, wobei ein Befehl – falls es sich um einen Befehl handelt, der im Text-Editor funktioniert und der unter Beachtung der Syntax eingegeben wurde – sofort ausgeführt wird. Stimmt irgendetwas nicht, so erscheint eine Fehlermeldung.

Haben Sie sich verschrieben, was auch mal vorkommen kann, so helfen Ihnen zwei Tasten weiter: Die Del-Taste löscht das Zeichen, das unmittelbar links vom Cursor auf dem Bildschirm steht und bewegt den Cursor, das unter ihm und alle rechts von ihm befindlichen Zeichen um eine Spalte nach links.

Nehmen wir an, Sie möchten den Befehl »RECT« eingeben, haben aber

```
RECZ
```

eingegeben, und der Cursor steht hinter dem »Z«, so bewegt sich der Cursor bei Betätigung der Del-Taste um eine Stelle nach links, zieht dabei das Leerzeichen unter sich mit und löscht somit das »Z«. Ähnlich verhält sich die ← -Taste: Das Zeichen unter dem Cursor wird gelöscht und alle Zeichen rechts vom Cursor werden um eine Position nach links verschoben. Dabei verändert der Cursor selbst aber nicht seine Position. Um die ganze Zeile zu löschen, in der sich der Cursor zur Zeit befindet, genügt es, Ctrl + Del zu drücken.

Mit der Tastenkombination [Control] + [1] (wie bei GeoWrite) nutzen Sie die eingebauten Tabulatoren des Text-Editors. Tabstopps befinden sich in den Spalten 8, 12, 16, 20, 24, 28, 32 und 36. Im Überschreibe-Modus wird der Cursor einfach nur an die neue Position bewegt, während im Einfüge-Modus eine entsprechende Anzahl von Leerzeichen an der alten Cursor-Position eingefügt wird. Dies ist besonders nützlich bei der nachträglichen Einrückung von Befehlen.

Wollen Sie das gesamte Editierfenster löschen, so betätigen Sie einfach [Shift] + [Home]. Das hochauflösende Grafikbild – also normalerweise das Bild Ihrer Applikation – können Sie sich nach Betätigen von [F7] ansehen. Mit [F7] gelangen Sie dann wieder zurück in den Text-Editor. Wird gerade ein Programm mit dem LIST-Befehl auf den Bildschirm gebracht, so können Sie das Scrollen durch Betätigen von [F5] stoppen und bei nochmaligem Betätigen wieder fortsetzen. Wollen Sie das Auflisten eines Programms mit LIST unterbrechen, so drücken Sie auf [Run Stop]. Mit dieser Taste können Sie auch laufende Programme unterbrechen. Sie gelangen dann zurück in den Text-Editor.

3.3.4 Das Eingeben, Verändern und Löschen von Programmzeilen

Möchten Sie eine Programmzeile eingeben, so bewegen Sie den Cursor an die erste Position einer Bildschirmzeile. Geben Sie die Zeilennummer, eventuell Leerzeichen sowie die Befehle und Parameter ein, die Sie wünschen. Beispiel:

```
10 PRINT "Dies ist ein Programm.": REM Bildschirmausgabe.
```

Schreiben Sie dabei über den rechten Bildschirmrand, so wird eine neue Zeile unter der aktuellen eingefügt, in der die Ausgabe fortgesetzt wird.

Haben Sie alle Befehle eingegeben, wird die Programmzeile nach Betätigen von [Return] in den Speicher übernommen. Der Cursor muß sich dabei nicht unbedingt hinter dem letzten Zeichen dieser Programmzeile befinden; es reicht, wenn er an einer beliebigen Stelle der Zeile steht.

Eine Programmzeile kann bis zu sechs Bildschirmzeilen (240 Zeichen) lang sein. Geben Sie mehr ein, so wird ein »Line-too-long«-Fehler ausgelöst und die Zeile wird nicht in das Programm übernommen. Sie müssen die Zeile nun komplett neu eingeben und dabei beachten, daß die maximale Länge von sechs Bildschirmzeilen nicht überschritten wird.

Eine Programmzeile kann sehr einfach wieder geändert werden, wenn sie auf dem Bildschirm komplett zu sehen ist. Bewegen Sie nur den Cursor an die Stelle der Zeile, an der Sie etwas ändern möchten, geben die Änderung ein und drücken [Return].

Steht die Programmzeile jedoch nicht komplett auf dem Bildschirm, dann müssen Sie sie zuerst durch Eingabe von »LIST« und der Zeilennummer mit anschließender Betätigung von Return auflisten. Beispiel:

`LIST 100` (um die Zeile 100 auf den Bildschirm zu bringen)

Möchten Sie eine Programmzeile löschen, dann geben Sie an der ersten Position einer Bildschirmzeile die Zeilennummer ein und betätigen direkt danach die Return-Taste. Es gibt im Text-Editor keine Funktion zum Löschen eines ganzen Programms, um z. B. ein neues eingeben zu können! Sie müssen erst das Basic-Programm schließen (»close«) und anschließend ein neues erstellen (Abschnitt 3.2.1). Das alte Programm können Sie nur unter dem Desktop löschen.

3.4 Die Menüleiste

Der Text-Editor stellt Ihnen weitere wichtige Funktionen in der Menüleiste am oberen Bildschirmrand zur Verfügung. Klicken Sie ein Schlagwort an, um ein Abrollmenü auf den Bildschirm zu bringen. Durch Anklicken wählen Sie die darin enthaltenen Funktionen; wollen Sie keine Funktion aufrufen, dann bewegen Sie den Mauszeiger vom Abrollmenü, ohne etwas anzuklicken.

3.4.1 »Geos«

GeoBasic info

Eine Dialogbox zeigt den Autor, die Versionsnummer und einen Copyright-Hinweis zu GeoBasic an.

Hilfsmittel

Im »Geos«-Abrollmenü werden auch die Namen der Hilfsmittel, die auf der Arbeitsdiskette gespeichert sind, angezeigt. Um ein Hilfsmittel zu starten, klicken Sie einfach auf dessen Namen.

3.4.2 »File« (Datei)

»close« (schließen)

Ihr GeoBasic-Programm wird auf Diskette gespeichert und Sie gelangen zum Startmenü zurück.

»update« (aktualisieren)

Ihr GeoBasic-Programm wird auf Diskette gespeichert. Sie sollten diesen Menüpunkt in regelmäßigen Abständen während Ihrer Arbeit aufrufen, um einen größeren Datenverlust zu vermeiden, z. B. bei einem Stromausfall oder wenn Sie versehentlich Ihren Computer ausschalten, ohne vorher GeoBasic beendet zu haben.

»rename« (umbenennen)

Ist der Name Ihres Programms eventuell unpassend und Sie möchten ihn ändern, dann wählen Sie diesen Menüpunkt an. Eine Dialogbox (Bild 3.7) erscheint, in der Sie den neuen Namen eingeben können.

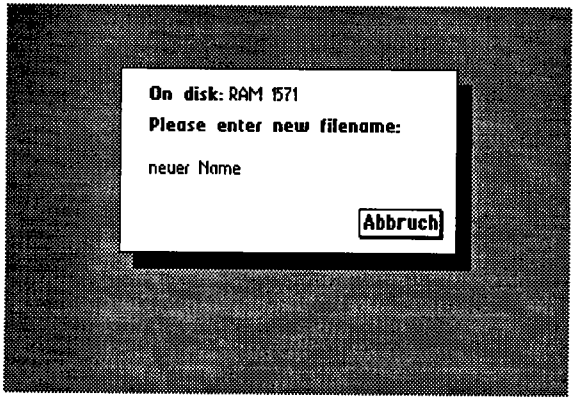


Bild 3.7: Umbenennen eines Basic-Programms

Drücken Sie Return, wenn Sie den Namen eingegeben haben. Das Programm wird nun im Speicher und auf Diskette umbenannt. Möchten Sie doch nicht den Namen ändern, dann klicken Sie »Abbruch« an oder drücken – ohne etwas einzugeben – auf Return.

»print« (drucken)

Dieser Menüpunkt gibt Ihr komplettes Programm als Listing auf dem Drucker aus. Ihr Drucker sollte bereits eingeschaltet sein, wenn Sie »print« aufrufen, da ansonsten ein »Device not found«-Fehler ausgelöst wird. Unterbrechen können Sie das Ausdrucken durch Betätigen von [Run Stop].

»quit« (Ende)

Ihr Programm wird auf Diskette gespeichert und Sie kehren zum Desktop zurück.

3.4.3 »Edit« (Editieren)

»list« (Programm listen)

Dieser Menüpunkt listet Ihr ganzes Programm auf dem Bildschirm. Sie können auch einfach »LIST« eingeben und [Return] drücken (Abschnitt 4.3.1). Beachten Sie hierbei die in Abschnitt 3.3.3 angegebenen Tastaturfunktionen ([Run Stop] und [F5]), um das Auflisten abzubrechen oder das Rollen des Bildschirms zu unterdrücken, wenn das Programm länger als 24 Bildschirmzeilen ist.

»sprcol« (Sprite-Farben)

Klicken Sie diesen Menüpunkt an, um die ersten beiden Farben für Ihre Sprites in einer Dialogbox (Bild 3.8) auszuwählen (Kapitel 5.6.1).

Bild 3.8: Auswahl der ersten beiden Farben für Sprites

Klicken Sie auf die mit »Multicolor 1« und »Multicolor 2« bezeichneten Rechtecke, um eine von 16 möglichen Farben auszuwählen. Sind Sie mit den gewählten Farben zufrieden, klicken Sie auf »OK«, um zum Text-Editor zurückzugelangen.

3.4.4 »Options« (Optionen)

»run« (starten)

Ihr Programm wird gestartet. Der Befehl RUN (Abschnitt 4.3.2) kann ebenfalls zum Starten des Programms verwendet werden.

»debug« (Fehlersuche)

In GeoBasic ist ein Debugger zur Fehlersuche in Ihren Basic-Programmen integriert. Starten Sie ihn über diesen Menüpunkt. Außerdem ist es möglich, den Debugger über Eingabe von »DEBUG« und Drücken von Return zu starten (Abschnitt 4.3.4). Eine ausführliche Beschreibung des Debuggers finden Sie in Kapitel 12.

»resize« (Programmspeicher definieren)

Für Ihre Programme stellt Ihnen GeoBasic ungefähr 10 Kbyte des Hauptspeichers zur Verfügung. Dieser relativ kleine Speicher (der Hauptspeicher umfaßt 64 Kbyte, und im Commodore Basic haben Sie immerhin mehr als 38 Kbyte für eigene Programme zur Verfügung) ist begründet durch die Größe des GEOS-Betriebssystems (ca. 32 Kbyte) und den Speicher, den GeoBasic für sich selbst beansprucht (ca. 21 Kbyte).

Die 10 Kbyte müssen Sie nun noch in Programm- und Variablenspeicher unterteilen, da alle Variablen auch Speicherplatz benötigen (Zeichenketten-Variablen benötigen sogar oft sehr viel Speicherplatz). Auch die Daten von Menüs, Dialogboxen, Zeichensätzen usw. werden im Variablenspeicher untergebracht.

GeoBasic macht diesen Speicherplatzmangel dadurch wieder wett, daß es immer nur einen kleinen Teil Ihres Programms (der Teil, welcher gerade abgearbeitet oder von Ihnen verändert wird) im Hauptspeicher hält und den Rest auf Diskette auslagert. Die Größe eines Programms ist also nicht auf die Größe des Programmspeichers beschränkt, sondern kann den gesamten verfügbaren Diskettenspeicherplatz umfassen. Auch bei größeren Programmen kann dank dieser Methode immer noch ein ausreichender Speicherbereich für Variablen usw. reserviert werden.

Voreingestellt ist ein Speicherbereich von 4 Kbyte für den aktuellen Teil des Hauptprogrammes und ca. 6 Kbyte für Variablen usw. Normalerweise reicht dies aus, doch spätestens dann, wenn Sie große Zeichensätze verwenden, müssen Sie den Variablenspeicher vergrößern.

Mit dem Menüpunkt »resize« können Sie nun in einer Dialogbox (Bild 3.9) die Größe des Programmspeichers (2 bis 8 Kbyte) bestimmen. Die Größe des Variablenspeichers ist immer 10 Kbyte abzüglich des Programmspeichers.

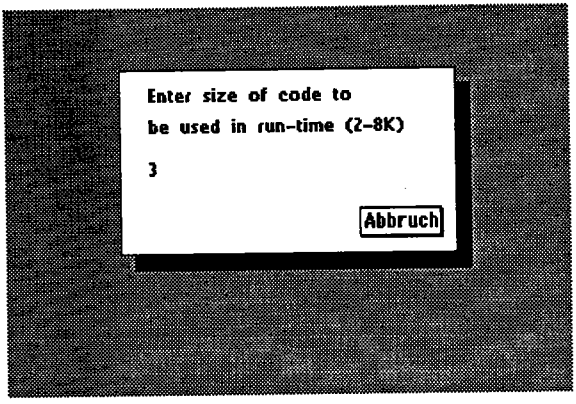


Bild 3.9: Bestimmung der Größe des Programmspeichers

»renumber« (neu numerieren)

Mit diesem Menüpunkt können Sie eine leistungsstarke Funktion zum Renumerieren eines bestehenden Basic-Programms aufrufen.

In einer Dialogbox (Bild 3.10) werden Sie aufgefordert, eine Zahl einzugeben, mit der GeoBasic das Programm neu numeriert. Nach der Eingabe drücken Sie auf Return.

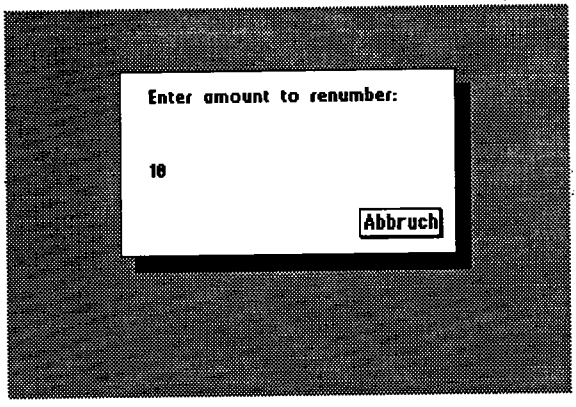


Bild 3.10: Nachträgliche Numerierung eines Programms

Geben Sie beispielsweise eine »100« ein, wird das Programm in 100er-Schritten neu numeriert, beginnend mit der Zahl 100. Ist Ihre Eingabe eine 5, dann fängt die Numerierung mit 5 an und verläuft in 5er-Schritten.

»make appl« (Applikation erstellen)

Aus Ihrem Programm wird eine eigenständig ablaufende Applikation erstellt. Näheres finden Sie in Kapitel 6.

3.4.5 »Utilities« (Hilfsprogramme)

»menu« (Menü-Editor)

Der Menü-Editor (Kapitel 7) wird aktiviert.

»dialog« (Dialogbox-Editor)

Der Dialogbox-Editor (Kapitel 8) wird aufgerufen.

»icon« (Piktogrammlisten-Editor)

Der Piktogrammlisten-Editor (Kapitel 9) wird aktiviert.

»bitmap« (Grafik-Editor)

Der Grafik-Editor (Kapitel 10) wird aufgerufen.

»sprite« (Sprite-Editor)

Der Sprite-Editor (Kapitel 11) wird aktiviert.

4. Kapitel
Der Basic-Lernkurs

In diesem Kapitel finden Sie alles über die grundlegenden GeoBasic-Befehle, die größtenteils in ihrer Funktion an gleichlautende Commodore-Basic-Befehle angelehnt sind. Auch wenn Sie sich mit Commodore Basic schon auskennen, sollten Sie dieses Kapitel wenigstens »überfliegen«, weil manche Befehle nicht ganz identisch mit denen des Commodore Basic sind. In diesem und dem folgenden Kapitel 5, das die Programmierung für Fortgeschrittene erklärt, sind die GeoBasic-Befehle aufgelistet und ausführlich erklärt.

Vorab müssen aber noch einige Begriffe, die im Zusammenhang mit den weiteren Befehlserklärungen stehen, definiert werden: Die GeoBasic-Befehle, Funktionen und Operatoren sind Schlüsselwörter und dürfen nicht in Variablennamen verwendet werden. In allen Kapiteln dieses Handbuches erscheinen sie zur besseren Erkennung in Großschrift.
PARAMETER, die mit den meisten Befehlen und Funktionen übergeben werden müssen, sind in den Kapiteln dieses Buches immer in Groß- und Kleinschrift angegeben. Parameter können sein: Dateinamen, Variablen, Konstanten, Zeilennummern, Labels, beliebige Ausdrücke und Operatoren.

ECKIGE KLAMMERN [] grenzen Parameter ein, die optional sind und nicht unbedingt angegeben werden müssen.

Stehen Parameter in APOSTROPHEN ('), dann müssen sie immer angeben werden.

Ein SCHRÄGSTRICH (/) trennt einzelne Parameter in einer Parameterliste. Sie haben dann einen der durch den Schrägstrich getrennten Parameter zu wählen.

EINE FOLGE VON PUNKTEN (...) bedeutet, daß ein Parameter auch mehrmals wiederholt werden kann.

Sind Parameter in RUNDE KLAMMERN () eingeschlossen, müssen diese Klammern in jedem Fall mit angegeben werden.

VARIABLE steht für einen beliebigen gültigen Variablennamen (Y, Z$, Q% usw.).

AUSDRUCK steht für einen beliebigen gültigen numerischen Ausdruck, wie z. B. »R * (4 / T)«.

STRING steht für eine beliebige Zeichenketten-Konstante, Variable oder für einen Zeichenketten-Ausdruck.

4.1 Konstanten, Variablen, Ausdrücke und Operatoren

Bevor wir nun mit der eigentlichen Programmierung beginnen, liegt noch ein ganzes Stück Theorie vor uns. Können Sie es schon überhaupt nicht mehr abwarten, bis Sie endlich programmieren können, dann rate ich Ihnen, diesen Abschnitt erst einmal zu überspringen und das Kapitel direkt nach dem PRINT-Befehl (Abschnitt 4.2) durchzuarbeiten.

4.1.1 Konstanten

Konstanten sind Werte, die während des Ablaufs eines Basic-Programms immer gleich bleiben (konstant). Es gibt drei verschiedene Typen von Konstanten:

1. Ganzzahl-Konstanten sind Zahlen ohne Dezimalpunkt im Bereich von –32767 bis 32767, die jeweils zwei Byte Speicherplatz belegen. Vorangestellte Nullen werden ignoriert. Beispiele für Ganzzahl-Konstanten:

   –10 1256 0 –2366

2. Fließkomma-Konstanten sind positive oder negative maximal neunstellige Zahlen, die einen Dezimalpunkt (trotz des Namens kein Komma!!!) enthalten dürfen. Wenn mehr als neun Stellen angegeben werden, dann wird die Konstante automatisch auf neun Stellen nach den geltenden Regeln gerundet. Beispiele für Fließkomma-Konstanten:

   14.6 3.1415927 –66666.66 .01

Ist die Zahl kleiner als –999999999 oder größer als 999999999, dann wird die Fließkomma-Konstante in Exponentialform ausgegeben. Dies sieht dann beispielsweise so aus:

1.23456E07

Der erste Teil der Zahl (die Ziffern links vom »E«), bezeichnet als Mantisse, ist eine Fließkomma-Zahl mit dem Dezimalpunkt hinter dem ersten Zeichen. Der Buchstabe »E« zeigt an, daß es sich hierbei um die Exponentialdarstellung einer Fließkomma-Konstanten handelt, und die Ziffern hinter dem »E« stellen den Exponenten zur Basis 10 dar, mit dessen Potenz ($10^{Exponent}$) die Mantisse multipliziert werden muß, um die eigentliche Fließkomma-Zahl zu erhalten.

Wenn wir beispielsweise die Zahl

3.12E02

in eine Fließkomma-Zahl umwandeln möchten, dann verfahren wir so:

3.12 * 10 ^ 2

und erhalten als Ergebnis 312.

Mantisse und Exponent können natürlich auch negativ sein. Wenn die Mantisse negativ ist, dann ist die gesamte Fließkomma-Zahl kleiner als 0. Ist der Exponent negativ, dann ist die Fließkomma-Zahl zwischen 0 und 1.

Demnach ist

3.12E-02

mit der Fließkomma-Zahl 0.0312 gleichzusetzen.

GeoBasic akzeptiert Fließkomma-Konstanten in Exponentialdarstellung zwischen ± 2.93873588E-39 und ± 1.70141183E38. Ist die Zahl zu groß, wird ein »OVERFLOW«-Fehler ausgegeben, ist sie zu klein, dann wird sie auf 0 gesetzt.

3. Zeichenketten-Konstanten sind eine Aneinanderreihung von alphanumerischen Zeichen (Buchstaben, Zahlen, Sonderzeichen und natürlich auch Leerzeichen), deren Länge nur durch die Länge einer Programmzeile beschränkt ist und die von Anführungszeichen (»"«) eingegrenzt sein müssen. Beispiele für Zeichenketten-Konstanten:

   "Hallo" "DM 14,–" "Ich sagte: 'Du mußt nach rechts abbiegen!'"

Das letzte Anführungszeichen einer Zeichenketten-Konstanten kann auch weggelassen werden, wenn nach der Zeichenkette in einer Programmzeile nichts mehr folgt.

4.1.2 Variablen

Variablen sind Werte, die sich während des Ablaufs Ihres Basic-Programms verändern können oder sogar unmittelbar vom Anwender des Programms beeinflußt werden dürfen.

Variablen haben immer einen Namen, an dem GeoBasic sie erkennen kann. Dieser Name kann beliebig lang sein, wobei aber nur die ersten drei Buchstaben beachtet werden. GeoBasic kann also beispielsweise »numDrives« nicht von »numPatt« unterscheiden und behandelt aus diesem Grund beide als eine Variable. Wie auch bei Labels, beachtet GeoBasic bei Variablen die Groß- und Kleinschreibung, »ABC« ist eine andere Variable als »aBc«! Die erste Ziffer des Variablennamens muß ein Buchstabe sein, die nachfolgenden können auch Zahlen sein, z. B. a01, Gu9.

Variablen dürfen keine Basic-Schlüsselworte (Befehlsnamen bzw. Funktionsnamen) enthalten, ansonsten tritt ein »Syntax«-Fehler beim Ablaufen auf. Auch bei Variablen gibt es verschiedene Typen wie bei den Konstanten, die auch wieder denselben Regeln und Einschränkungen unterliegen (Abschnitt 4.1.1): Ganzzahl-Variablen, Fließkomma-Variablen und Zeichenketten-Variablen.

Zur Unterscheidung dieser drei Typen hängen Sie an den Variablennamen noch ein Kennzeichen an; dies ist bei Ganzzahl-Variablen ein Prozentzeichen (»%«) und bei Zeichenketten-Variablen ein »$«. Bei Fließkomma-Variablen entfällt eine Kennung.

Gültige Variablen sind beispielsweise:

a	(Fließkomma-Variable)
Number%	(Ganzzahl-Variable)
Text$	(Zeichenketten-Variable)

Nicht möglich sind folgende Variablen:

to%	(enthält das Befehlswort »TO«)
1a$	(das erste Zeichen ist kein Buchstabe)

Die Wertzuweisung erfolgt bei einer Variablen, indem Sie diese mit einem Ausdruck (Abschnitt 4.1.3) gleichsetzen:

```
D$      = "Hallo Ihr"
aX      = (4.5 * 100 / 10) ^ 2
Test%   = Gurt%
Test    = A * B / C
Be%     = (14 + Ce) * 6
A       = A + 1
```

Sie sollten darauf achten, numerischen Variablen (Ganzzahl- und Fließkomma-Variablen) nur numerische Ausdrücke und Zeichenketten-Variablen nur Zeichenketten-Ausdrücke zuzuweisen. In allen anderen Fällen ist ein »Type mismatch«-Fehler die Folge.

Wenn Sie einer Ganzzahl-Variablen einen Wert mit Nachkommastellen zuweisen möchten, dann wird nur die Zahl vor dem Komma übernommen, die Nachkommastellen werden ignoriert, z. B.:

```
10 a% = 4.45 * 1.22
20 PRINT a%
```

Das Belegen einer Fließkomma-Variablen mit einem Ganzzahl-Ausdruck ist natürlich problemlos möglich.

Enthält ein Ausdruck eine Variable, die noch keinen Wert hat, dann nimmt diese Variable automatisch den Wert 0 an:

```
10 b = 4.12
20 a = b + c
30 PRINT a
```

Belegen Sie eine Variable mit einem Wert, wenn Sie bereits einen Wert hat, dann geht dabei der alte Wert verloren und die Variable nimmt den neuen Wert an:

```
10 a = 14
20 a = 7
30 PRINT a
```

Wie Sie wissen, ist der Speicher Ihrer Programme sehr stark eingeschränkt. Dies betrifft hauptsächlich den Variablenspeicher, da er – im Gegensatz zum Programmspeicher – nicht mit Disketteninhalten ausgetauscht werden kann (Abschnitt 3.4.4.3).

Zeichenketten benötigen besonders viel Speicherplatz des ohnehin nur kleinen Speichers. Und oftmals brauchen Sie diese Zeichenketten höchstens ein einziges Mal, danach können sie wieder gelöscht werden.

Die Zeichenketten-Variable

TMP$

kann eine einzelne Zeichenkette aufnehmen, ohne daß diese Speicherplatz im Variablenspeicher belegt. Einzig beim ersten Verwenden von TMP$ wird ein Bereich von wenigen Byte (unabhängig von der Länge der verwendeten Zeichenkette) reserviert. Bei jeder anschließenden Verwendung wird die Größe des Variablenspeichers nicht mehr beeinflußt.

Ein Beispiel:

```
10 TMP$="Diese Zeichenkette ist nach dem PRINT-Befehl nicht mehr
       wichtig!"
20 PRINT TMP$
```

Die Zeichenketten-Variable TMP$ in Zeile 10 verbraucht trotz ihrer enormen Länge keinen Speicherplatz! Dazu ist sie als einzige Variable in der Lage, jede andere Variable verkleinert den Variablenspeicher.

Schreiben Sie jetzt zur Übung ein kleines Programm, das nur aus einer Reihe von Variablenzuweisungen besteht:

```
10 @varKonst:
20    A  = 14.5
30    B% = 6
40    C$ = "Hallo"
50
60 @varAusdruck:
70    D  = A + B%
80    E% = 16 * (1/A)
```

```
90     F$ = C$ + " Ihr"
100
110 @varVar:
120    G$ = F$
130    G$ = G$ + F$
```

Nach Starten des Programms (mit RUN + RETURN) erhält in Zeile 20 die Fließkomma-Variable A die Fließkomma-Konstante 14.5 zugewiesen, in 30 wird B% (Ganzzahl-Variable) auf 6 (Ganzzahl-Konstante) gesetzt und in 40 bekommt die Zeichenketten-Variable C$ die Zeichenketten-Konstante "Hallo" zugewiesen.

Nach Abarbeitung der Zeilen 70, 80 und 90 haben die Variablen D, E% und F$ folgende Werte:

```
D  = 20.5
E% = 1 (da die Nachkommastellen .1034483 abgeschnitten werden)
F$ = "Hallo Ihr"
```

In Zeile 120 erhält die Zeichenketten-Variable G$ den Wert der Zeichenketten-Variablen F$, also "Hallo Ihr".
Zeile 130 sorgt dafür, daß an die Zeichenketten-Variable G$ die Zeichenkette F$ angehängt wird (Ergebnis: "Hallo IhrHallo Ihr").

4.1.3 Ausdrücke und Operatoren

Jetzt haben wir im letzten Abschnitt schon so viel von »numerischen Ausdrücken« gehört, ohne mehr darüber zu wissen, als daß es zu berechnende Formeln sind.

Nehmen wir uns einfach mal einen dieser numerischen Ausdrücke als Beispiel:

```
(20 - a%) / 4
```

Die Variablen und Konstanten in einem Ausdruck bezeichnet man als »Operanden«, die sie verbindenden Zeichen als »Operatoren«.

Operatoren in numerischen Ausdrücken

Operatoren, die in numerischen Ausdrücken genutzt werden können, sind:

1. ^ Potenzierung
2. * Multiplikation
   / Division
3. + Addition
   - Subtraktion

In der hier aufgeführten Reihenfolge werden die einzelnen Operatorengruppen von GeoBasic abgearbeitet. Die Operatoren innerhalb der einzelnen Gruppen (also beispielsweise Multiplikation und Division) werden in der Reihenfolge abgearbeitet, wie sie auftreten.

Durch das Setzen von runden Klammern kann die Reihenfolge der Abarbeitung von Operatoren beeinflußt werden. Eingeklammerte Bereiche des Ausdrucks werden von GeoBasic immer zuerst bearbeitet. (20 + 5) * 3 ergibt also 75 und nicht 35!

Vergleichsoperatoren, die Sie auch in numerischen Ausdrücken verwenden können, werden – wie der Name schon sagt – zum Vergleichen von zwei Teilen des Ausdrucks genutzt. Das Ergebnis eines Ausdrucks mit einer Vergleichsoperation kann nur –1 (wahr) oder 0 (falsch) sein. Mögliche Vergleichsoperatoren sind

< kleiner als,
= gleich,
> größer als,
<= kleiner als oder gleich,
>= größer als oder gleich sowie
<> ungleich.

Beispiele:

2 = 3 – 1 Ergebnis ist: –1 (wahr)

20 + 4 < 5 + 9 Ergebnis ist: 0 (falsch)

Die dritte und letzte Gruppe der Operatoren sind die logischen Operatoren. Sie beeinflussen numerische Ausdrücke auf der Ebene der kleinsten Informationseinheit im Computer: dem Bit. Ein Bit kann nur die Zustände »Aus« (0) und »Ein« (1) annehmen. Mit nur einem Bit könnte ein Computer also auch nur die Zahlen 0 und 1 darstellen. Um nun größere Zahlen und sogar Buchstaben verwalten zu können, sind immer 8 solcher Bit zu einer Einheit zusammengeschlossen – nämlich zu einem Byte. Mit einem Byte können schon bis zu 256 Zahlen (0 bis 255) dargestellt werden.

Die Zahl 5, aufgeteilt in 8 Bit, sieht so aus:

0 0 0 0 0 1 0 1

Dies ist die größte Zahl, die mit dieser Anzahl von Bits angezeigt werden kann:

1 1 1 1 1 1 1 1 (255)

Auf die eigentliche Umrechnung möchte ich hier nicht eingehen, da dies den Rahmen dieser Einführung in die GeoBasic-Programmierung sprengen würde. Möchten Sie noch größere Zahlen als 255 darstellen, so werden einfach noch mehr einzelne Bits zu einer Gruppe zusammengefaßt. Wichtig ist in diesem Zusammenhang, daß logische Operatoren in GeoBasic nur mit Zahlen zusammenarbeiten, die aus maximal 16 Bit zusammengesetzt sind und darum in einem Bereich von –32767 bis 32767 liegen dürfen. Größere oder kleinere Zahlen werden von GeoBasic mit einem »Illegal quantity«-Fehler quittiert.

Kommen wir aber nun zu den logischen Operatoren. Mit dem Operator NOT sieht die Beeinflussung der Bits wie folgt aus:

Operand	Ergebnis nach NOT
0	1
1	0

NOT dreht also die Bitwerte um: aus einer 0 wird eine 1 und aus einer 1 eine 0.
Beispiel für einen Ausdruck mit einer NOT-Operation:

NOT (2 = 1)

Da 2 ungleich 1 ist, wäre das Ergebnis der Klammer 0 (falsch). Dies ist – in 16 Bit ausgedrückt – 0000000000000000. Ein »NOT 0000000000000000« ergibt nun (da NOT alle Bits umdreht) 1111111111111111, und das ist die Dezimalzahl –1.

NOT (2 = 1) ergibt also –1 (wahr).

Der zweite Operator, AND, arbeitet folgendermaßen:

Operand 1	Operand 2	Ergebnis nach AND
0	0	0
1	0	0
0	1	0
1	1	1

Der Ausdruck

3 AND 5

würde also

```
    0000000000000011 (3)
AND 0000000000000101 (5)
--------------------------
    0000000000000001 (1)
```

ergeben, da AND im Ergebnis ein Bit nur dann setzt, wenn es in beiden Operanden auch gesetzt ist! Ist das Bit nur in einem Operanden oder in keinem gesetzt, dann wird es im Ergebnis gelöscht.

Beispiel für einen komplexeren Ausdruck mit AND:

```
(4 + 5) = 9 AND (10 - 3) = 7
```

Die Ausdrücke links und rechts von AND sind beide wahr (–1). Verknüpfen Sie nun beide Ergebnisse per AND-Operator miteinander, ergibt sich folgendes:

```
1111111111111111 (–1)
AND 1111111111111111 (–1)
----------------------
1111111111111111 (–1)
```

Das Ergebnis des gesamten Ausdrucks ist also –1 (wahr).

Der dritte logische Operator ist OR. Die »Verknüpfungstabelle« für OR sieht so aus:

Operand 1	Operand 2	Ergebnis nach OR
0	0	0
1	0	1
0	1	1
1	1	1

OR setzt das Bit im Ergebnis, wenn das Bit entweder in einem oder in beiden Operanden gesetzt ist. Die Verknüpfung von 3 und 5 mittels OR ergibt:

```
0000000000000011 (3)
OR 0000000000000101 (5)
-----------------------
0000000000000111 (7)
```

Ein weiteres Beispiel für die OR-Verknüpfung:

```
(4 - 3) = 0 OR (16 + 1) = 17
```

Der Ausdruck vor dem OR ergibt 0 (falsch), der Ausdruck hinter OR ist wahr, also –1.

```
0000000000000000 (0)
OR 1111111111111111 (–1)
------------------------
1111111111111111 (–1)
```

Das Ergebnis des ganzen Ausdrucks ist also –1 (wahr).

Nachdem Sie nun alle Operatoren von GeoBasic kennengelernt haben, finden Sie hier noch die Hierarchie dieser Operatoren (die Reihenfolge, in der sie vom Computer innerhalb eines Ausdrucks bearbeitet werden):

1. ()	Ausdrücke in Klammern
2. ^	Potenzierung
3. *	Multiplikation
/	Division
4. +	Addition
–	Subtraktion
5. < = >	Vergleichs-Operatoren
6. NOT, AND, OR	logische Operatoren

Operatoren in Zeichenketten-Ausdrücken

Neben den numerischen Ausdrücken gibt es noch Zeichenketten-Ausdrücke, z. B.

```
"Dies " + "ist " + "ein " + "Test."
"M" + "N" = "MN"
```

Die einzigen Operatoren, die auch bei Zeichenketten-Ausdrücken verwendet werden können, sind der Additions-Operator (+) und die Vergleichs-Operatoren.

```
"Hallo 2" + "Du."
```

ergibt beispielsweise

```
"Hallo Du."
```

und

```
"A" = "B"
```

ergibt 0 (falsch).

Bei Vergleichsoperationen ist »A« immer kleiner als »B«, das wiederum kleiner als »C« ist usw.

4.2 Bildschirmausgabe

In nahezu allen Fällen muß ein Programm mit dem Benutzer kommunizieren. Dies geschieht in der Regel über Bildschirmausgaben, die durch den im nächsten Abschnitt beschriebenen Befehl PRINT ausgelöst werden. Alle weiteren Befehle und Funktionen in den Abschnitten 4.2.2 bis 4.2.4 ergänzen den PRINT-Befehl.

4.2.1 PRINT

Mit dem PRINT-Befehl können Sie GeoBasic dazu bringen, etwas auf dem Bildschirm oder (wenn Sie den PRNTER-Befehl aus Abschnitt 5.9.3 verwenden) auch auf dem Drucker auszugeben. Geben Sie einfach einmal das folgende Beispiel ein, starten es durch Eingabe von RUN und Betätigen von [Return] und schauen sich das Resultat an:

```
10 PRINT "Dies ist ein Test des PRINT-Befehls."
20 A$ = "GeoBasic2"
30 PRINT A$
```

GeoBasic gibt – ausgelöst durch Zeile 10 – den Satz

```
Dies ist ein Test des PRINT-Befehls.
```

auf dem Bildschirm aus. Zeile 20 ist eine Variablenzuweisung (siehe Abschnitt 4.1) und in Zeile 30 wird der Inhalt dieser (Zeichenketten-)Variablen mittels des PRINT-Befehls auf den Bildschirm ausgegeben.

Drücken Sie eine beliebige Taste oder klicken Sie mit der Maus, um zum Text-Editor zurückzukehren, da das Programm beendet ist. Nun ergänzen Sie folgende Zeilen:

```
40 X = 10
50 PRINT X
60 PRINT "X hoch 2 ergibt: "; X ^ 2
```

Auch in Zeile 40 finden Sie wieder die Zuweisung einer Variablen – diesmal ist es aber eine Fließkomma-Variable.

Anschließend wird in Zeile 50 der Wert der Variablen (10), in Zeile 50 der Text

```
X hoch 2 ergibt:
```

und direkt dahinter das Ergebnis eines arithmetischen Ausdrucks (100) auf den Bildschirm ausgegeben. Vergessen Sie nicht, eine Taste zu drücken oder die Maus zu klicken, wenn Sie nach dem Ablaufen eines Programms in den Text-Editor zurückkehren möchten!

Man kann also sagen, daß man mit dem PRINT-Befehl Zeichenketten, Variablen, Ausdrücke etc. ausgeben kann. Fassen wir dies nun in eine Syntax-Beschreibung von PRINT zusammen:

PRINT ['Ausdruck'][,/;['Ausdruck']]...

Geben Sie hinter einem PRINT-Befehl keinen Ausdruck an, dann wird einfach eine leere Zeile ausgegeben.

Wie Sie sehen, ist es auch möglich, mehrere Ausdrücke mit einem PRINT-Befehl auszugeben. Diese müssen dann durch ein Komma (,) oder ein Semikolon (;) getrennt werden. In ihrer Funktion sind sich diese Zeichen sehr ähnlich, dienen sie doch beide als Trennzeichen, die Wirkung ist aber verschieden:

Das Semikolon bewirkt, daß der folgende Ausdruck ohne einen Zwischenraum hinter dem letzten Ausdruck ausgegeben wird, auch bei numerischen Ausdrücken! Um Zahlen, die Sie mit dem PRINT-Befehl und Trennzeichen Semikolon ausgeben, also korrekt lesen zu können, müssen Sie selbst für einen Zwischenraum sorgen (z. B. durch die Ausgabe einer Zeichenketten-Konstanten mit nur einem Leerzeichen darin).

Ist der letzte Parameter eines PRINT-Befehls ein Semikolon, bleibt der Cursor nach der Ausgabe des letzten Ausdrucks dieses PRINT-Befehls direkt hinter den ausgegebenen Zeichen stehen und springt nicht – wie es normalerweise ist – an den Anfang der nächsten Bildschirmzeile.

Der Grafik-Bildschirm von GeoBasic ist horizontal in acht gleichgroße Zonen unterteilt. Ein Komma sorgt dafür, daß der Cursor an den Anfang der nächsten dieser acht Zonen gebracht wird, und daß ein folgender Ausdruck dort ausgegeben wird. Mit dem Komma können Sie also sehr einfach Ausdrücke in Tabellenform auf den Bildschirm bringen.

Ist der letzte Parameter eines PRINT-Befehls ein Komma und ein Semikolon (,;), dann bleibt der Cursor nach der Ausgabe des letzten Ausdrucks dieses PRINT-Befehls in der nächsten Zone stehen und springt nicht – wie es sonst der Fall wäre – an den Anfang der folgenden Bildschirmzeile. Dazu ein Beispiel:

```
10 FOR I = 1 TO 50
20  PRINT I; " ";
30 NEXT I
```

Die beiden (wahrscheinlich unbekannten) Befehle in den Zeilen 10 und 30 interessieren uns jetzt nicht. Sie sorgen nur dafür, daß die Zeile 20 insgesamt 50mal abgearbeitet und die Variable I dabei von 1 bis 50 durchgezählt wird.

Mit dem PRINT-Befehl in Zeile 20 wird der Inhalt dieser Variablen ausgegeben – aufgrund der anderen Zeilen insgesamt 50mal und dabei jeweils um eins erhöht. Außerdem wird nach jeder Zahl ein Leerzeichen ausgegeben, um das Lesen der Zahlen überhaupt zu ermöglichen.

Beachten Sie die Art der Bildschirmausgabe (beeinflußt durch das Semikolon), wenn Sie das Beispiel nun starten (durch RUN + Return).

Fügen Sie anschließend vor dem letzten Semikolon in Zeile 20 ein Komma ein (vergessen Sie das Betätigen der Taste Return danach nicht) und starten das Programm nochmals.

Erkennen Sie die Veränderung in der Ausgabe? Wenn nicht, dann löschen Sie das Komma wieder und starten das Programm noch einmal.

Das letzte Beispiel für den PRINT-Befehl soll eine Besonderheit bei der Ausgabe von Zeichenketten erläutern:

```
10 A$ = "Karsten": B$ = "Jan": C$ = "Thomas"
20 PRINT A$B$; C$, B$
```

(Keine Angst, Zeile 20 ist nicht falsch!) Geben Sie weiter ein:

```
30 PRINT "Hi, "A$". ""Ich hoffe, Du bist angenehm überrascht!"
40 PRINT "Grüße Heike von mir."
```

Ausgegeben wird folgendes:

```
KarstenJanThomas     Jan
Hi, Karsten.
Ich hoffe, Du bist angenehm überrascht! Grüße Heike von mir.
```

Zwischen Zeichenketten-Konstanten und Variablen in einem PRINT-Befehl müssen keine Trennzeichen stehen!

Mit dem jetzt erworbenen Wissen über den PRINT-Befehl sollten Sie noch weiter experimentieren, bevor Sie mit dem nächsten Abschnitt weitermachen. PRINT ist ein sehr wichtiger Befehl, da Sie durch ihn mit dem Benutzer kommunizieren können.

4.2.2 SETPOS

Der SETPOS-Befehl verändert die Position von auszugebenden Zeichen auf dem Bildschirm. SETPOS hat folgende Syntax:

SETPOS 'Ausdruck','Ausdruck'

Der erste Ausdruck setzt die X-Position, an der das nächste Zeichen ausgegeben wird, und muß zwischen 0 und 319 liegen. Der andere Ausdruck setzt die zugehörige Y-Position auf einen beliebigen Wert zwischen 0 und 199. Die Y-Koordinate bezieht sich auf die Grundlinie der auszugebenden Zeichen (Bild 4.1).

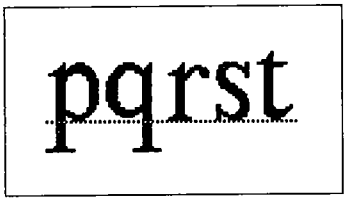


Bild 4.1: Die Grundlinie eines Zeichensatzes

Ein Beispiel:

```
10 SETPOS 10, 10
20 PRINT "Dieser Satz steht an (10/10)!"
30 SETPOS 140, 100
40 PRINT "Koordinaten: (140/100)"
```

SETPOS erlaubt als Parameter außer den numerischen Konstanten auch numerische Ausdrücke (beispielsweise X * 15 / 2) und numerische Variablen (wie A%, xP usw.)

4.2.3 XPOS

Die XPOS-Funktion übergibt die X-Koordinate des Cursors (dort, wo PRINT-Befehle Zeichen auf dem Bildschirm ausgeben). Die Koordinate ist eine ganze Zahl im Bereich von 0 bis 319.

Syntax: XPOS('Ausdruck')

Der Ausdruck in Klammern kann jeder beliebige numerische Ausdruck oder jede beliebige numerische Variable oder Konstante sein, da er nicht benutzt wird. Er muß aber immer angegeben werden! Ein Beispiel zu XPOS finden Sie im nächsten Abschnitt.

4.2.4 YPOS

Syntax: YPOS('Ausdruck')

Die YPOS-Funktion übergibt die Y-Koordinate des Cursors (0 bis 199). Diese Koordinate bezieht sich auf die Grundlinie des Zeichens (Bild 4.1). Auch bei YPOS wird der anzugebende Ausdruck nicht benötigt und kann beliebig sein. Beispiel für XPOS und YPOS:

```
10 SETPOS 0, 0
20 A = XPOS(0) + 5
30 B = YPOS(0) + 10
40 SETPOS A, B
50 PRINT "Position 1"
60 SETPOS 0, 0
70 PRINT "Linke obere Position."
80 SETPOS XPOS(1) + 100, YPOS(255) + 120
90 PRINT "Position 2"
```

Nachdem in Zeile 10 der Cursor in die linke obere Ecke des Bildschirms gesetzt wird, übertragen die Zeilen 20 und 30 die X- sowie Y-Position desselben in Fließkomma-Variablen und erhöhen sie um bestimmte Werte. Zeile 40 setzt den Cursor auf die neuen Werte aus den Fließkomma-Variablen. Der PRINT-Befehl in Zeile 40 bewirkt die Ausgabe von »Position 1« an dieser Position.

Zeile 60 setzt den Cursor wieder in die »Home«-Position (links oben am Bildschirm) und in Zeile 70 wird die Zeichenkette »Linke obere Position.« ausgegeben. Der Cursor wird auf eine neue Koordinate in Zeile 80 gesetzt, wobei die aktuelle X-Koordinate um 100 erhöht und die aktuelle Y-Position um 120 erhöht wird. Zeile 90 bewirkt die Ausgabe von »Position 2«.

4.3 Befehle im Text-Editor

Im Text-Editor (Abschnitt 3.3) können auch bestimmte GeoBasic-Befehle verwendet werden. Diese Befehle sind in den folgenden Abschnitten beschrieben. Alle anderen Befehle und alle Funktionen lassen sich nicht im Text-Editor verwenden und lösen einen »Syntax«-Fehler aus.

4.3.1 LIST

Bereits in Abschnitt 3.3.4 wurde die Möglichkeit beschrieben, einzelne Zeilen zwecks Veränderung mit dem LIST-Befehl auf den Bildschirm zu holen. Da die Möglichkeiten dieses Befehls aber wesentlich umfangreicher sind, erfahren Sie hier alles Weitere.

Die Funktion des LIST-Befehls ist das Anzeigen des Programms im Editierfenster. Paßt es nicht vollständig hinein, werden vorher ausgegebene Programmzeilen nach oben aus dem Fenster herausgerollt. Die vollständige Syntax des LIST-Befehls lautet:

LIST ['Ausdruck'][,['Ausdruck']]

Wird der LIST-Befehl ohne jeglichen Parameter eingegeben, dann listet GeoBasic das ganze Programm auf.

Möchten Sie nur eine ganz bestimmte Zeile sehen, dann geben Sie LIST und die Zeilennummer ohne ein Komma ein. Dann wird nur diese Zeile gelistet.

Geben Sie nach LIST noch die erste Zeile und ein Komma an, so werden diese Zeile und alle folgenden Zeilen des Programms auf den Bildschirm gebracht.

Wird außer LIST noch das Komma und die letzte Zeile eingegeben, erscheinen alle Zeilen vom Anfang des Programms bis einschließlich der eingegebenen Zeile.

Sind beide (die erste und die letzte Zeilennummer) angegeben, werden alle dazwischenliegenden einschließlich dieser beiden Zeilen gelistet. Anstelle der Zeilennummern können Sie auch beliebige Labels (und auch Variablen sowie ganze Ausdrücke) im LIST-Befehl verwenden. Beispiele für LIST:

```
LIST                  (das ganze Programm wird gelistet)
LIST 100              (nur Zeile 100 wird gelistet)
LIST 100,             (alle Zeilen von 100 bis zum Ende des Programms erscheinen)
LIST , 100            (listet alle Zeilen vom Anfang bis Zeile 100)
LIST 100, 500         (die Zeilen von 100 bis 500 werden gelistet)
LIST @Start, @Ende (die Zeilen von @Start bis @Ende werden gelistet).
```

4.3.2 RUN

RUN startet ein GeoBasic-Programm.

Syntax: RUN ['Zeilennummer']

Wird RUN ohne eine Zeilennummer (wie auch bei LIST und FIND sind auch Labels möglich) eingegeben, wird das Programm in seiner ersten Zeile gestartet, ansonsten in der angegebenen Zeile. Existiert diese aber nicht, wird ein »Undefined statement error«-Fehler ausgegeben.

Das laufende Programm wird beendet, sobald ein END-Befehl die letzte Programmzeile erreicht oder RUN STOP gedrückt wird.

Hinweis: Ist die letzte Zeile oder ein END-Befehl abgearbeitet, kehrt GeoBasic nicht sofort in den Text-Editor zurück, sondern wartet erst auf den Druck einer Taste oder des Mausknopfs. Beispiele für RUN:

RUN (startet ein Programm in der ersten Zeile)

RUN 200 (startet das Programm ab Zeile 200)

Hinweis: RUN ist ein Befehl im Text-Editor. Er sollte nicht in Programmen verwendet werden, da ansonsten ein Absturz von GeoBasic provoziert werden könnte. Verwenden Sie anstelle dessen den GOTO-Befehl (Abschnitt 4.5.1)!

4.3.3 FIND

Mit FIND können Sie Ihr Programm ganz oder teilweise nach bestimmten Befehlen, Variablen oder ganzen Zeichenpassagen durchsuchen lassen. Die Syntax von FIND ist folgende:

FIND "String"[,['Ausdruck'][,['Ausdruck']]]

FIND funktioniert genauso wie LIST, wobei aber immer nur die Zeilen ausgegeben werden, die auch die Suchzeichenkette enthalten.

Müssen Sie beispielsweise ganz schnell einen bestimmten POKE-Befehl finden, wissen aber nicht mehr die genaue Position im Programm, können Sie FIND so anwenden:

```
FIND "POKE"
```

GeoBasic listet jetzt alle Zeilen, die den Befehl POKE enthalten. Kennen Sie noch den ungefähren Bereich, in dem der POKE-Befehl stand, können Sie dies auch in den FIND-Befehl einarbeiten:

```
FIND "POKE", 100,        (nur in den Zeilen von 100 bis zum Ende suchen)
FIND "POKE",, 100        (nur bis Zeile 100 nach POKE suchen)
FIND "POKE", 100, 500    (von Zeile 100 bis Zeile 500 das Programm durchsuchen)
```

Auch hier ist wieder die Verwendung von Labels anstelle der Zeilennummern möglich, z.B.

```
FIND "POKE", @Start, @Ende
```

Möchten Sie nur eine bestimmte Zeile nach der Zeichenkette durchsuchen lassen, dann geben Sie deren Nummer getrennt durch ein Komma nach der Zeichenkette ein:

```
FIND "POKE", 100    (Zeile 100 durchsuchen)
```

4.3.4 DEBUG

Syntax: DEBUG ['Ausdruck']

Der DEBUG-Befehl arbeitet ähnlich wie der RUN-Befehl: er startet Ihr Programm. Vorher können Sie aber in der Debug-Dialogbox (Kapitel 12) noch einige Parameter festlegen.

Mit dem optionalen numerischen Ausdruck übergeben Sie die Nummer der Zeile (oder auch den Label), ab der (dem) das Programm gestartet werden soll. Wird dieser Parameter nicht angegeben, dann wird das Programm in seiner ersten Zeile gestartet.

4.4 Annahme von Tastatureingaben

Die Möglichkeit der Eingabe von bestimmten Werten finden Sie in fast jedem Programm, sei es beispielsweise die Eintragung des Namens in die High-Score-Liste eines Spieles oder aber das Umbenennen von Dateien.

Auch in eigenen GeoBasic-Programmen besteht die Möglichkeit, Eingaben zu verschiedensten Zwecken entgegenzunehmen.

4.4.1 INPUT

Mit INPUT können Sie sehr einfach komplexe Eingaben vom Benutzer holen. So erwartet

```
INPUT A$
```

die Eingabe einer Zeichenkette durch den Benutzer und übergibt diese anschließend in die Variable A$. Kommen wir aber erst einmal zur Syntax von INPUT:

INPUT ['String';]'Variable'['Variable']...

Lassen wir jetzt »String« erst einmal außer Betracht und kümmern uns nur um die Variablen. Jeder beliebige Typ (Zeichenketten-, Fließkomma- und Ganzzahl-Variable) kann als Parameter verwendet werden.

Trifft GeoBasic auf einen INPUT-Befehl, so wird der Ablauf des Programms angehalten, ein Fragezeichen an der aktuellen Position ausgegeben und der Cursor (ein blinkender Strich) eingeschaltet. Sie können dann Ihre Eingabe tätigen, die Sie anschließend mit RETURN beenden müssen.

Stehen mehrere Variablennamen hinter dem INPUT-Befehl, so trennen Sie die einzelnen Werte bei der Eingabe durch Kommata. Hierzu ein Beispiel: Steht im Programm

```
INPUT A, A$, B%,
```

so werden die Werte folgendermaßen eingegeben:

```
12.75,Hallo Du,-32
```

Geben Sie zu wenige Werte ein, dann erscheinen in der nächsten Bildschirmzeile zwei Fragezeichen (??), und Sie müssen die vergessenen Werte eingeben. Haben Sie aber zu viele Werte eingegeben, dann wird die Meldung »Extra ignored« ausgegeben, was bedeutet, daß die zusätzlichen Eingaben in keiner Variablen gespeichert – also einfach ignoriert – werden. Die einzelnen Werte, die Sie eingeben, dürfen natürlich kein Komma enthalten, da das Komma als Trennzeichen benutzt wird!

Wenn Sie versehentlich eine Zeichenkette eingegeben haben, und der Wert soll in einer Fließkomma- oder Ganzzahl-Variablen gespeichert werden, dann erscheint die Meldung »Redo from start«, und Sie müssen einen korrekten Wert für diese Variable eingeben. Möchten Sie den alten Wert der Variable im INPUT-Befehl erhalten, dann drücken Sie einfach auf RETURN, ohne einen Wert einzugeben.

Nun, da Sie den INPUT-Befehl kennengelernt haben, können Sie schon recht nützliche Programme gestalten, z. B. ein Programm zur Errechnung des Flächeninhalts eines Rechteckes, dessen Seitenlängen der Benutzer eingibt:

```
10 INPUT A, B
20 F = A * B
30 PRINT "Flächeninhalt: ";F
40 GOTO 10
```

In Zeile 10 erwartet GeoBasic vom Benutzer die Eingabe der Seitenlängen und speichert die Werte in den Variablen A und B. Anschließend wird der Variablen F in Zeile 20 der Flächeninhalt – errechnet aus den eingegebenen Werten – zugewiesen und in Zeile 30 auf dem Bildschirm ausgegeben. Zeile 40 läßt das Programm dann wieder von vorne ablaufen. Abbrechen können Sie das Programm durch Drücken der RUN STOP -Taste.

Kommen wir jetzt zu dem bisher überschlagenen »String« aus der Syntax-Angabe. Einem INPUT-Befehl kann eine beliebige Zeichenketten-Konstante (ein Text eingeschlossen in Anführungszeichen) folgen. Dieser Text wird dann vor dem Fragezeichen auf dem Bildschirm ausgegeben. So können Sie beispielsweise dem Benutzer erklären, was er eingeben soll.

Sie dürfen aber nie das Semikolon (kein Komma!!!) zwischen der Zeichenketten-Konstanten und der ersten Variablen vergessen. Gestalten Sie jetzt das Flächeninhaltsprogramm wie folgt um:

```
10 INPUT "Länge der Seite A in cm"; A
20 INPUT "Länge der Seite B in cm"; B
30 F = A * B
40 PRINT "Flächeninhalt: ";F;" cm^2"
50 GOTO 10
```

Probieren Sie diese Version einfach mal aus, um die Unterschiede zum Vorgänger erkennen zu können.

4.4.2 GET

GET funktioniert ähnlich wie INPUT. Dieser Befehl übergibt den Wert einer gedrückten Taste in Variablen. Die Syntax von GET lautet:

GET 'Variable'[,'Variable']...

Zeichenketten-, Fließkomma- und Ganzzahl-Variablen können als Parameter angegeben werden. Wird aber bei einer Fließkomma- oder Ganzzahl-Variablen ein Buchstabe eingegeben, so bricht GeoBasic das Programm mit der Meldung »Type mismatch« ab. Darum sollte man besser Zeichenketten-Variablen angeben und diese anschließend in eine numerische Variable konvertieren (Abschnitt 4.10.2), wenn eine Zahl zur Weiterverarbeitung unbedingt nötig ist.

Da das Programm beim Auffinden einer GET-Anweisung nicht automatisch gestoppt wird, um auf einen Tastendruck zu warten, wird bei gar keiner Eingabe im Moment der Abarbeitung von GET die Variable vollkommen leer übergeben. Es empfiehlt sich darum eine Schleife z. B. mit IF...THEN (Abschnitt 4.6.1), um auch wirklich den Wert eines Tastendrucks und nicht eine leere Variable zu erhalten:

```
10 GET A$
20 IF A$ = "" THEN GOTO 10
30 PRINT A$
```

In Zeile 10 wird eine Eingabe von der Tastatur geholt. Erfolgte keine Eingabe, dann ist die Variable A$ leer, und GeoBasic verzweigt aus Zeile 20 mit einem GOTO-Sprung (Abschnitt 4.5.1) wieder in Zeile 10, um die Tastatur nochmals abzufragen. Erst, wenn eine

Taste gedrückt wurde, enthält die Variable A$ einen Wert, der in Zeile 30 auf den Bildschirm geschrieben werden kann.

4.4.3 PROMPT

Ein Manko des GET-Befehls ist, daß der Cursor nicht eingeschaltet wird, um dem Benutzer zu signalisieren, daß er etwas eingeben soll. Dafür gibt es in GeoBasic den Befehl PROMPT.

Mit PROMPT können Sie den Cursor ein- oder ausschalten und ihn gleichzeitig an eine beliebige Position setzen. Parameter müssen PROMPT wie folgt übergeben werden:

PROMPT 'Ausdruck','Ausdruck','Ausdruck'

Alle Ausdrücke können beliebige numerische Ausdrücke, Konstanten oder Variablen sein. Der erste Ausdruck entscheidet, ob der Cursor eingeschaltet (ungleich 0) oder ausgeschaltet (gleich 0) wird. Mit den beiden anderen Ausdrücken, die übrigens auch dann anzugeben sind, wenn der Cursor ausgeschaltet wird, definieren Sie seine X-Position (von 0 bis 319) und Y-Position (0 bis 199). Ein Beispiel für PROMPT:

```
10 PRINT ">";
20 PROMPT 1, XPOS(0), YPOS(0) - 6
30 GET a$
40 IF a$="" THEN GOTO 30
50 PRINT a$;
60 GOTO 20
```

Wichtigste Stelle dieses Programms ist Zeile 20. Durch Sie wird der blinkende Cursor an der Stelle eingeblendet, wo die nächste Bildschirmausgabe stattfinden wird (XPOS und YPOS, Abschnitt 4.2). Durch regelmäßiges Anspringen (Zeile 60) ist gewährleistet, daß die Cursor-Position nach jeder erfolgten Bildschirmausgabe wieder korrigiert wird. Ergebnis: Der Cursor agiert genau wie bei der INPUT-Eingabe.

Natürlich handelt es sich bei der hier vorgestellten Routine noch nicht um eine komplette Eingaberoutine. Es lassen sich nämlich noch nicht alle Tastenfunktionen (wie DEL) nutzen und außerdem werden die eingegebenen Zeichen nirgendwo gespeichert.

4.5 Sprungbefehle und Unterroutinen

Schon im Abschnitt 2.2 wurde die Möglichkeit erwähnt, mit bestimmten Befehlen Sprünge innerhalb des Programms ausführen zu können, z.B. um eine sehr wichtige Funktion aufzurufen oder um eine für den Benutzer unwichtige Programmstelle zu umgehen.

4.5.1 GOTO

Der wohl einfachste Befehl zum Springen in Programmen ist GOTO.

Mit ihm haben Sie schon in einigen Beispielen der letzten Abschnitte Bekanntschaft gemacht. Seine Syntax ist folgende:

GOTO 'Ausdruck'

Der Ausdruck kann ein beliebiger numerischer Ausdruck, ein Label, eine numerische Konstante oder Variable sein und definiert die Zeilennummer, an der das Programm fortgesetzt werden soll. Hierzu ein Beispiel:

```
10   PRINT "Zeile 10 vor dem GOTO-Sprung."
20   GOTO @Routine
30
40   @Schleife:
50   GOTO @Schleife
60
100  @Routine:
110  PRINT "Wir sind in Zeile 110!!!"
120  GOTO 40
```

Zeile 20 könnte natürlich ebenso »GOTO 100« oder »GOTO 10 * 10« lauten.

Der erste Sprung führt – nach Abarbeitung des PRINT-Befehls in Zeile 10 – von Zeile 20 in Zeile @Routine (Zeile 100). Ist die Routine beendet, wird aus Zeile 120 in eine Schleife (Zeilen 40 und 50) gesprungen. Um das Programm zu beenden, müssen Sie RUN STOP betätigen.

4.5.2 GOSUB...RETURN

Auch der GOSUB-Befehl (kombiniert mit RETURN) dient dem Verzweigen innerhalb eines GeoBasic-Programmes.

Eine Ähnlichkeit von GOSUB und GOTO läßt sich schon rein vom Namen her nicht bestreiten. Selbst die Syntax

GOSUB 'Ausdruck'

zeigt das. Auch hier ist es möglich, jeden beliebigen numerischen Ausdruck oder eine beliebige numerische Variable oder Konstante sowie Labels zu verwenden. Der Unterschied zwischen GOTO und GOSUB ist folgender:

Bei Auffinden eines GOSUB-Befehls merkt sich GeoBasic automatisch die Programmstelle, an der dieser Befehl stand. Ist nun die mit GOSUB aufgerufene Routine (genannt: Unterroutine) mit ihrer Arbeit fertig und beabsichtigt, zum aufrufenden Punkt zurückzukehren, so kann

sie dies durch einen einfachen RETURN-Befehl machen. GeoBasic arbeitet anschließend direkt hinter dem GOSUB-Befehl weiter. Die Routine muß also nicht die aufrufende Stelle kennen.

Benötigen Sie einen bestimmten Programmteil mehrmals, so können Sie ihn – statt ihn mehrmals in ein Programm einzubauen – besser mit einem RETURN abschließen und dann beliebig oft mit GOSUB aufrufen!

Ist es nötig, von verschiedenen Stellen der Unterroutine zur Hauptroutine zurückkehren zu können, dann können Sie auch mehr als einen RETURN-Befehl in dieser Unterroutine verwenden!

Natürlich kann die aufgerufene Unterroutine ebenso wieder Unterroutinen aufrufen usw. (man nennt dies »Verschachtelung von Unterroutinen«). Versucht man aber, zu viele Unterroutinen zu verschachteln, wird mit einem »Out of memory«-Fehler das Programm abgebrochen. Dies ist damit zu erklären, daß der Speicherbereich, in dem sich GeoBasic die Rückkehradressen merkt, auch nur von begrenzter Größe ist. Nun aber erst einmal ein Beispiel zu GOSUB...RETURN:

```
10 FOR N = 1 TO 10
20  GOSUB @Routine
30 NEXT N
40 END
50
100 @Routine:
110  PRINT "Diese Unterroutine wird zum ";N;". Mal aufgerufen."
120 RETURN
```

Der FOR...NEXT-Befehl wird im Abschnitt 4.7.1 erklärt.

Wichtig sind die Zeilen 20 und 120: In Zeile 20 erfolgt der Aufruf der Unterroutine. Sie könnten z. B. auch »GOSUB 100« oder »GOSUB 10 * 10« einsetzen. Ist die Unterroutine beendet, wird durch den RETURN-Befehl das Programm nach dem GOSUB-Befehl (also in Zeile 30) fortgesetzt.

Vergessen Sie nicht den END-Befehl in Zeile 40 (Abschnitt 4.11.4). Fehlt er, dann wird nach Beendigung der FOR...NEXT-Schleife in Zeile 30 das Programm in Zeile 100 fortgesetzt und bei Erreichen der Zeile 110 ein »RETURN without GOSUB«-Fehler ausgegeben, da GeoBasic nicht weiß, wohin der RETURN-Befehl zurückspringen soll, wenn kein GOSUB-Aufruf vorangegangen ist.

4.6 Abhängige Verzweigungen

Die Befehle in diesem Abschnitt ermöglichen Ihnen das Abhängigmachen der Programmreaktionen von bestimmten Eingaben und Berechnungen.

4.6.1 IF...THEN

Durch die IF...THEN-Programmierung können Sie die Abarbeitung bestimmter Programmteile von Ergebnissen beliebiger Berechnungen, Zustände oder Eingaben abhängig machen.

Hierzu ein Beispiel aus der realen Welt: Sie gehen doch sicher nicht Kaffee kaufen, wenn Sie kein Geld mehr in der Tasche haben. Mit GeoBasic-Befehlen ließe sich dieser Sachverhalt wie folgt ausdrücken:

IF Geld = 0 THEN Nicht Kaffee kaufen

Der Umkehrschluß hiervon wäre dann:

IF NOT(Geld = 0) THEN Kaffee kaufen

Syntax: IF 'Ausdruck' THEN 'Befehle'

Jeder erdenkliche Ausdruck (numerisch oder Zeichenkette), jede Variable, jede Konstante kann als Ausdruck eingesetzt werden. Befehle können alle gültigen GeoBasic-Befehle sein, wobei mehrere Befehle durch Doppelpunkte getrennt werden.

Ist das Ergebnis des Ausdrucks wahr, werden die Befehle hinter THEN ausgeführt, stellt er sich als falsch heraus, dann wird mit der Bearbeitung der nächsten Zeile fortgefahren. Kommen wir noch einmal zu unserem Kaffee-Beispiel:

```
10 INPUT "Wieviel Geld haben Sie noch"; Geld
20 INPUT "Wieviel kostet Ihr Kaffee"; Preis
30 IF Geld < Preis THEN PRINT "Gehen Sie besser nicht mehr
einkaufen!": END
40 PRINT "Sie können beruhigt Ihren Kaffee kaufen!"
50 END
```

Der END-Befehl wird in Abschnitt 4.11.4 erklärt. Erst fordert das Programm (Zeilen 10 und 20) den Benutzer zur Eingabe des vorhandenen Geldbetrages und des Kaffee-Preises auf. Anschließend wird geprüft (Zeile 30), ob weniger Geld vorhanden ist als der Kaffee kostet. Ist dies der Fall, so wird dem Benutzer vom Kauf des Kaffees abgeraten und das Programm ist beendet. Ansonsten werden Sie in Zeile 40 zum Kaffee-Kauf ermuntert und das Programm endet. Noch ein Beispiel mit verstärktem IF...THEN-Einsatz:

```
10 @Start:
20 Count = 0
30 PRINT "Ich denke mir jetzt eine Zahl zwischen 1 und 10 aus."
40 Zahl = INT(RND(10))+1
50 PRINT "Raten Sie die Zahl!"
60 INPUT Eingabe
70 Count = Count + 1
80 IF Eingabe < 1 OR Eingabe > 10 THEN PRINT "Nein, nur im Bereich
von 1 bis 10!": GOTO 50
90 IF Eingabe = Zahl THEN GOTO @Richtig
100 IF Eingabe < Zahl THEN PRINT "Größer!": GOTO 50
110 PRINT "Kleiner!"
120 GOTO 50
130
140 @Richtig:
150 PRINT "Wow!!! Richtig!!!"
160 PRINT "In nur ";Count;" Zügen haben Sie es geschafft."
170 INPUT "Möchten Sie nochmal spielen (J/N)"; Again$ 180 IF Again$
= "J" OR Again$ = "j" THEN GOTO @Start
190 PRINT "Bis zum nächsten Mal ..."
200 END
```

Dies ist ein Zahlen-Ratespiel. Die Befehle INT und RND erzeugen in Kombination eine glatte Zufallszahl (Abschnitt 4.10.1).

Versuchen Sie einmal durch Ausprobieren, alle Funktionen des Programms und die IF...THEN-Befehle zu erkunden. Ein geglückter Versuch mit diesem Spiel sieht so aus (Bild 4.2):

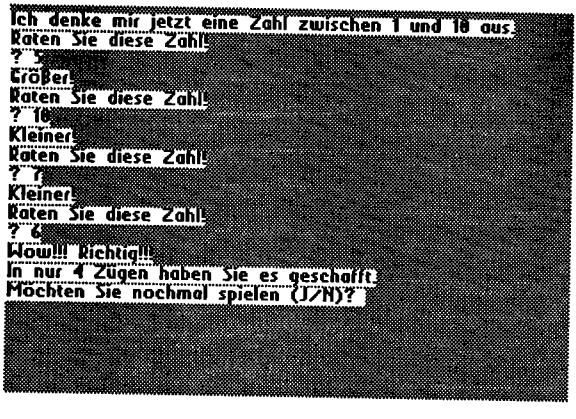


Bild 4.2: Das Zahlen-Ratespiel

4.6.2 ON...GOTO/GOSUB

Kommen wir nun zu einer Besonderheit der Befehle GOTO und GOSUB (Abschnitt 4.5). Mit ON...GOTO/GOSUB können Sie je nach Wert eines Ausdrucks an verschiedene Stellen des Programms verzweigen.

Syntax: ON 'Ausdruck' GOTO/GOSUB 'Ausdruck'[,'Ausdruck']...

Ist der Wert des Ausdrucks hinter ON gleich 1, dann verzweigt das Programm an die durch den ersten Ausdruck hinter GOTO/GOSUB bezeichnete Stelle, ist er 2, dann verzweigt es an die durch den zweiten Ausdruck hinter GOTO/GOSUB bezeichnete Stelle usw.

Der Ausdruck hinter ON muß numerisch sein, ist es ein Fließkomma-Ausdruck, dann werden die Nachkommastellen ignoriert. Ist er 0, dann verzweigt das Programm gar nicht, sondern fährt mit dem nächsten Befehl nach ON...GOTO/GOSUB fort. Wenn er negativ ist, dann wird ein »Illegal quantity«-Fehler ausgegeben.

Die Ausdrücke hinter GOTO/GOSUB dürfen beliebige ganzzahlige numerische Ausdrücke sein und bezeichnen die Sprungziele. Neben Variablen und Konstanten empfiehlt sich hier natürlich wieder die Verwendung von Labels.

Sollten Sie jetzt auf die Idee gekommen sein, daß der ON...GOTO/GOSUB eigentlich überflüssig ist, da eine Verzweigung je nach Wert eines Ausdrucks auch mit IF...THEN...GOTO/GOSUB realisierbar ist, gebe ich Ihnen grundsätzlich recht. In folgendem Falle bewährt sich aber der Einsatz von ON...GOTO/GOSUB:

```
10 INPUT A
20 IF A = 1 THEN GOSUB @Eingabe
30 IF A = 2 THEN GOSUB @Ausgabe
40 IF A = 3 THEN GOSUB @Ende
```

Mit ON...GOTO/GOSUB sieht das Ganze folgendermaßen aus:

```
10 INPUT A
20 ON A GOSUB @Eingabe, @Ausgabe, @Ende
```

Gespart haben Sie jetzt ganze zwei Programmzeilen und darum wird Ihr GeoBasic-Programm schneller ablaufen können. Weitere Beispiele:

```
10 X = 5
20 ON X - 4 GOTO 100, 200, 200
```

Ihr Programm würde in Zeile 20 nach Zeile 100 verzweigen, da X –4 den Wert 1 ergibt.

Hinweis: Vergessen Sie bei Verwendung von ON...GOSUB nicht die RETURN-Befehle am Ende der auszuführenden Routinen.

4.7 Programmschleifen

Dieser Abschnitt erläutert die in GeoBasic enthaltenen Befehle zur Schleifenprogrammierung.

4.7.1 FOR...TO...STEP...NEXT

Die genaue Wirkungsweise der Befehlskette FOR...TO...STEP...NEXT läßt sich am einfachsten anhand eines kleinen Beispiels erläutern:

```
10 FOR A = 1 TO 50 STEP 1
20  PRINT A,;
30 NEXT A
```

Geben Sie dieses Programm ein und starten es dann. Es gibt die Zahlen von 1 bis 50 auf dem Bildschirm aus. Und warum?

Ganz einfach: Der FOR...NEXT-Befehl leitet eine Schleife ein und setzt die Variable A auf den Anfangswert 1. Dieser wird in Zeile 20 ausgegeben. Der NEXT-Befehl in Zeile 30 bewirkt, daß sozusagen »die nächste Runde eingeläutet wird« und in Zeile 10 A um 1 (da mit STEP die Zählschritte auf 1 festgelegt wurden) auf 2 erhöht wird, die Variable A wieder ausgegeben wird usw. Erst, wenn der Endwert 50 überschritten ist (hier also Variable A auf 51 steht), wird die Schleife beendet und hinter Zeile 30 weitergearbeitet, wo nichts mehr folgt. Das Programm wird also beendet. Wir können also zur Syntax von FOR...NEXT sagen:

FOR 'Variable' = 'Ausdruck' TO 'Ausdruck' [STEP 'Ausdruck']
...
(hier können beliebige Befehle stehen!)
...
NEXT ['Variable'][,'Variable']...

Die Variable muß eine Fließkomma-Variable sein, Ganzzahl-Variablen und Zeichenketten-Variablen können Sie nicht verwenden! Der erste (numerische) Ausdruck bezeichnet den Anfangswert der Zählvariable, der zweite den Endwert.

Der Zähler wird in jedem Durchlauf um den hinter STEP stehenden numerischen Ausdruck erhöht (oder vermindert bei negativen Werten). Ist STEP nicht angegeben, so wird automatisch ein Wert von 1 für die Erhöhung eingesetzt.

Ist der Ausdruck hinter STEP negativ, dann muß auch der Endwert kleiner sein als der Anfangswert; ansonsten ist eine kontrollierte Schleife nicht möglich!

Zwischen FOR und NEXT dürfen alle gültigen GeoBasic-Befehle (auch weitere Programmschleifen) stehen – natürlich auch verteilt auf mehrere Zeilen. Verschachteln können Sie FOR...NEXT-Schleifen bis zur neunten Ebene. Beachten Sie dabei aber bitte, daß Sie, wenn

Sie Schleifen verschachteln, jeweils andere Zählvariablen verwenden, da ansonsten die Schleifen nicht mehr richtig funktionieren.

Ist kein Parameter zum NEXT-Befehl angegeben, wird automatisch die zuletzt definierte Schleife weitergezählt. Parameter müssen die Variablennamen der in den FOR-Befehlen als Zähler definierten Variablen sein und immer zuerst der Name des Zählers der zuletzt definierten Schleife, dann der Name des Zählers der davor definierten Schleife usw.

Findet GeoBasic ein NEXT, ohne daß vorher ein entsprechendes FOR im Programm stand, wird der Ablauf mit der Meldung »NEXT without FOR« abgebrochen. Beispiele für FOR...NEXT:

```
10 A = 5: B = 10: C = 0.5
20 FOR I = A TO B STEP C
30      PRINT I,;
40      FOR J = A + 5 TO B + 5 STEP C / 2
50          PRINT I * J
60 NEXT J, I
```

Anstelle des einen NEXT-Befehls für beide Schleifen hätten Sie auch jeweils einen eigenen NEXT-Befehl ansetzen können.

```
10 FOR A = 0 TO -15 STEP -1
20      PRINT A
30 NEXT A
```

Beim Herunterzählen müssen Sie den STEP-Befehl immer angeben (da der voreingestellte Wert von 1 nicht möglich ist).

Mit FOR...NEXT lassen sich auch ziemlich einfach Verzögerungsschleifen herstellen:

```
10 FOR i = 0 TO 500: NEXT i
20      REM Das Programm wird erst nach dieser Schleife fortgesetzt.
```

4.7.2 REPEAT...UNTIL

Die zweite Möglichkeit, eine Programmschleife herzustellen, liegt bei REPEAT...UNTIL. Die Syntax lautet:

REPEAT
...
(beliebige Befehle)
...
UNTIL 'Ausdruck'

Alle Befehle, die auf REPEAT folgen, werden von GeoBasic ausgeführt, bis UNTIL erreicht wird. Ist der Ausdruck, der auf UNTIL folgt, falsch (0), dann verzweigt das Programm zurück zu REPEAT und die Befehle zwischen REPEAT und UNTIL werden ein weiteres Mal ausgeführt.

Erst, wenn der Ausdruck wahr (–1) wird, verläßt das Programm die Schleife und wird mit dem nach UNTIL folgenden Befehl fortgesetzt.

Beachten Sie, daß der Programmteil zwischen REPEAT und UNTIL immer mindestens einmal ausgeführt wird, auch wenn die UNTIL-Bedingung von Anfang an wahr ist. Dies resultiert daraus, daß die Bedingung erst bei Erreichen von UNTIL geprüft wird und nicht schon zu Beginn der Schleife.

Wenn der Ausdruck hinter UNTIL immer falsch bleibt, dann wird eine endlose Schleife ausgeführt.

Mit REPEAT...UNTIL läßt sich sehr einfach eine Tastaturabfrage programmieren, wenn man auf die IF...THEN-Abfrage (Abschnitt 4.6.1) verzichten will:

```
10 REPEAT
20     GET A$
30 UNTIL A$<>""
```

Erst, wenn A$ nicht mehr leer ist (also eine Taste gedrückt wurde), ist die Bedingung hinter UNTIL wahr, und die Schleife kann verlassen werden. Ein weiteres Beispiel:

```
10 REPEAT
20     A = A + 1
30     PRINT A,;
40 UNTIL A = 20
```

Es werden alle Zahlen zwischen 1 und 20 ausgegeben.

4.7.3 WHILE...LOOP

WHILE...LOOP-Schleifen ähneln den REPEAT...UNTIL-Schleifen sehr stark. Die Syntax lautet:

WHILE 'Ausdruck'
...
(beliebige Befehle)
...
LOOP

Im Gegensatz zu REPEAT...UNTIL wird die WHILE...LOOP-Schleife ausgeführt, während das Ergebnis des Ausdrucks hinter WHILE wahr (–1) ist.

Wird LOOP erreicht, verzweigt das Programm zum WHILE-Befehl und prüft, ob die Bedingung immer noch wahr ist. Ist sie es nicht, dann wird die Schleife verlassen und die auf LOOP folgenden Befehle werden ausgeführt.

Der Programmteil zwischen WHILE und LOOP kann nie ausgeführt werden, wenn die Bedingung immer falsch ist.

Ist hingegen die Bedingung immer wahr, dann endet das ganze Spielchen in einer Endlos-Schleife. Beispiel für WHILE...LOOP:

```
10 X = 10
20 WHILE X < 20
30      X = X + 1
40      PRINT X,;
50 LOOP
```

Auf dem Bildschirm werden die Zahlen von 11 bis 20 ausgegeben.

4.8 Feldvariablen

In diesem Abschnitt finden Sie Erläuterungen zu einer Sonderform der Variablen – den Feldvariablen. Die Funktion dieser Variablenart läßt sich am besten an einem kleinen Beispiel erläutern:

Nehmen wir an, Sie möchten ein Vokabelprogramm entwickeln, das bis zu 100 Vokabeln und die zugehörigen deutschen Übersetzungen verwalten kann. So könnte eine Eingaberoutine für dieses Vokabelprogramm aussehen:

```
10 INPUT "Vokabel";a1$
20 INPUT "Übersetzung";b1$
30 INPUT "Vokabel";a2$
40 INPUT "Übersetzung";b2$
50 INPUT "Vokabel";a3$
60 INPUT "Übersetzung";b3$
70 INPUT "Vokabel";a4$
80 INPUT ...
90 ... ...
```

Diese Methode ist jedoch umständlich und verbraucht sehr viel Speicherplatz, da die einzelnen Routinen (z. B. zur Eingabe und Ausgabe) sehr lang werden.

Verwenden wir anstelle der je einhundert verschiedenen Variablen für Vokabeln und Übersetzung doch besser eine einzelne Variable, die mehrere Werte (Elemente) gleichzeitig (also ein Feld von Elementen) aufnehmen kann: eine Feldvariable.

Feldvariablen bestehen aus einem Variablennamen (Abschnitt 4.1.2) und eventuell dem Kennzeichen $ (für Zeichenketten-Variablen) oder % (für Ganzzahl-Variablen). Verwenden Sie keine Kennung, dann handelt es sich bei Ihrer Feldvariablen um eine Fließkomma-Variable.

Nach dem Namen müssen Sie in Klammern einen sogenannten »Index« angeben, an dem GeoBasic erkennen kann, auf welches Element des Feldes Sie zugreifen möchten.

Gestalten wir nun ein Feld A$ für unsere Vokabeln. Es muß 100 Elemente speichern können (von 0 bis 99) und füllen dies gleich mit ein paar Beispielvokabeln:

Index: 0 1 2 3 ... 98 99

Inhalt: to take to swim to laugh ... to run to swap

A$(99)

enthält beispielsweise »to swap« und läßt sich durch

```
1000 PRINT A$(99)
```

auf dem Bildschirm unter der Bedingung ausgeben, daß ein entsprechend der oben abgebildeten Übersicht aufgebautes Feld bereits im Speicher steht.

Das oben aufgeführte Feld nennt man »eindimensionales Feld«, da seine Elemente anhand eines einzelnen Indizes bestimmt werden können.

Möchten wir nun auch noch unsere Übersetzungen unterbringen, dann müssen wir entweder ein weiteres eindimensionales Feld (z. B. B$(0) – B$(99)) oder aber anstelle des eindimensionalen Feldes A$ ein zweidimensionales Feld A$ verwenden.

In einem zweidimensionalen Feld werden zwei Indizes benötigt, um das Element zu bestimmen, auf das man zugreifen will:

1. Dimension:

   0 (enthält die Vokabeln)
   1 (enthält die Übersetzungen)

2. Dimension:

0	to take	nehmen
1	to swim	schwimmen
2	to laugh	lachen
3	...	...
...	...	...
98	to run	rennen
99	to swap	tauschen

In der Variablen

A$(0,1)

unseres Beispielfeldes steht »to swim«. Die Übersetzung (»schwimmen«) finden Sie in

A$(1,1).

Vokabel und Übersetzung lassen sich durch

```
1000 PRINT "Vokabel: "; A$(0,1)
1010 PRINT "Übersetzung: "; A$(1,1)
```

ausgeben. Haben Sie sich nun endgültig für eine zweidimensionale Feldvariable A$ in unserem Vokabelprogramm entschieden (es ist am zweckmäßigsten), gestalten Sie die Eingaberoutine folgendermaßen:

```
10 FOR I =  0 TO 99
20          INPUT "Vokabel"; A$(0,I)
30          INPUT "Übersetzung"; A$(1,I)
40 NEXT I
```

Diese Eingaberoutine ist deutlich kürzer als die ganz oben vorgestellte. Bestimmen wir jetzt aber den Aufbau und die Bedingungen von Feldvariablen. Zuerst der Aufbau:

'Variablenname'('Ausdruck'[,'Ausdruck']...)

Mit dem Variablennamen haben wir uns weiter oben schon ausgiebig beschäftigt (Abschnitt 4.1.2). Die numerischen Ausdrücke stellen die Indizes dar, mit denen das Element bestimmt wird, auf das wir zugreifen möchten, und zwar immer der erste Ausdruck den Index der ersten Dimension, der zweite Ausdruck den Index der zweiten Dimension usw. Indizes müssen immer ganzzahlig sein!

Theoretisch kann eine Feldvariable bis zu 127 Dimensionen groß sein, in normalen Anwendungen reichen aber ein bis drei Dimensionen meist aus. Außerdem schränkt die Größe des Variablenspeichers die Größe eines Feldes ein.

In diesem Handbuch sind die Feldvariablen mit unter den Oberbegriff »Variablen« gefaßt. Sie können also sicher sein, daß Sie anstelle von normalen Variablen auch immer Feldvariablen als Parameter zu Befehlen angeben können, auch wenn dies nicht ausdrücklich erwähnt ist.

Mögliche Feldvariablen sind beispielsweise:

A$(0,1,3), curPatt(4,1000), P%(16) und Status(0)

4.8.1 DIM

Wegen ihres besonderen Aufbaus müssen Sie Feldvariablen vor der Benutzung sozusagen bei GeoBasic »anmelden«. Dies geschieht mit Hilfe des DIM-Befehls. Die Syntax von DIM lautet:

DIM 'Variable'('Größe')[,'Variable'('Größe')]...

Möchten Sie mehrere Variablen mit einem DIM-Befehl dimensionieren, dann trennen Sie diese durch Kommata. Für »Variable« setzen Sie den Namen der Variable ein, die Sie dimensionieren (einrichten) möchten.

Anstelle von »Größe« soll die maximale Größe aller Indizes stehen. Möchten Sie ein Feld B% mit drei Elementen (0 bis 2) in der ersten Dimension und 15 Elementen (0 bis 14) in der zweiten Dimension (insgesamt also 45 Elemente) dimensionieren, dann kann Ihr DIM-Befehl so aussehen:

```
10 DIM B%(2,14)
```

Anstelle der numerischen Konstanten in den runden Klammern ist es auch möglich, numerische Ausdrücke und Variablen zu verwenden (z. B. 1 + 1 oder 2 * 7).

Vor die Eingaberoutine unseres Vokabelprogramms gehört also noch folgende Zeile, um es komplett zu machen:

```
5 DIM A$(1,99)
```

Die Elemente eines numerischen Feldes erhalten bei der Dimensionierung alle den Wert Null, die Elemente von Zeichenketten-Feldern bleiben leer (Null-Zeichenketten). Eine Feldvariable kann nicht zweimal dimensioniert werden; ein »Redimensioned array«-Fehler wäre die Folge. Nachträgliche Änderungen sind also nicht möglich!

Wollen Sie in Ihrem Programm auf ein Element zugreifen, das nicht in diesem Feld existiert, dann wird dies mit einem »Bad subscript«-Fehler quittiert. Beispiel für DIM:

```
10 DIM aB(4,5,6), A$(1,999)
20 DIM Cutt%(14)
```

Der DIM-Befehl muß nicht angegeben werden, wenn Ihr Feld nur eindimensional sein wird und nicht mehr als 11 Elemente (0 bis 10) enthalten soll. Führt GeoBasic beispielsweise die Zeile

```
100 XY(8)=199.95
```

aus und eine Dimensionierung dieser Variable hat noch in keiner der vorherigen Zeilen stattgefunden, dann wird

```
DIM XY(10)
```

automatisch durchgeführt. Wie Sie sicher während Ihrer späteren Programmierarbeit noch erfahren werden, bieten Feldvariablen sehr viele Möglichkeiten und erleichtern die Lösung von vielen Problemen, wie auch in unserem Beispiel mit dem Vokabelprogramm.

Sie sollten sich nicht von der Komplexität dieser Variablenart abschrecken lassen, sondern mit einfachen Dingen (eindimensionalen Feldern) beginnen. Später, wenn Sie genügend Erfahrung gesammelt haben, läuft die Programmierung und Verwaltung auch mehrdimensionaler Felder schon (fast) von alleine ab!

4.9 Datentabellen

Dieser Abschnitt beschäftigt sich mit der Verwaltung von Datentabellen in GeoBasic-Programmen.

4.9.1 DATA

Nehmen wir an, Sie sind immer noch dabei, Ihr Vokabelprogramm (siehe Abschnitt 4.8) zu schreiben, und nun kommen Sie auf die Idee, eine Anzahl von Vokabeln mit Übersetzungen sozusagen resident in das Programm einzubauen. Diese sind dann sofort nach dem Starten verfügbar und müßten nicht erst mühsam eingegeben werden. Eine Möglichkeit, dies zu realisieren, ist:

```
10  DIM A$(1,99)
20
30  A$(0,0) = "to take"
40  A$(1,0) = "nehmen"
50
60  A$(0,1) = "to swim"
70  A$(1,1) = "schwimmen"
80
90  A$(0,2) = "to write"
100 A$(1,2) = "schreiben"
110
120 A$(0,3) = ...
130 ... ...
```

Weil sie zu umständlich ist, sollten Sie diese Methode aber nicht anwenden. Einfacher ist die Verwendung von Datentabellen mit dem DATA-Befehl.

Syntax: DATA 'Konstante'[,'Konstante']...

Sie können beliebige numerische oder Zeichenketten-Konstanten als Parameter übergeben. Mehrere Parameter werden durch Kommata getrennt. Obwohl dies nicht vorgeschrieben ist, sollten Sie Zeichenketten-Konstanten immer in Anführungszeichen setzen, da Sie sonst keine Leerzeichen und Sonderzeichen in diesen Zeichenketten verwenden können!

Die Werte Null oder leere Zeichenketten als Parameter können Sie getrost weglassen. Sie müssen anstelle dessen nur das Komma setzen:

DATA 0, 100, 0, 200.2, "Hallo Du!", 0

ist mit der Zeile

DATA , 100, , 200.2, "Hallo Du!",

identisch. Ausgestattet mit dem Wissen über diesen Befehl können Sie nun die Vokabeln und Übersetzungen für das Vokabelprogramm in DATA-Zeilen schreiben:

```
10 DIM A$(1,99)
20
30 DATA "to take", "nehmen", "to swim", "schwimmen"
40 DATA "to write", "schreiben"
```

4.9.2 READ

Nun benötigen wir nur noch eine Routine, die die Daten (siehe auch RESTORE) nach dem Starten einliest:

```
100 FOR I = 0 TO 2
110       READ A$(0,I), A$(1,I)
120 NEXT I
```

Kommen wir zur Funktionsweise dieser Routine: GeoBasic verwaltet einen internen Zeiger, der zu Beginn des Programms auf das erste Element der ersten DATA-Zeile des Programms (»to take«) deutet.

DATA-Befehle werden von GeoBasic nicht direkt ausgeführt, sondern einfach ignoriert, wenn es während des Programmablaufs auf sie trifft. Nachdem also unser Programm gestartet wurde, wird in Zeile 10 das zweidimensionale Feld A$ für die Vokabeln und Übersetzungen eingerichtet und dann passiert lange Zeit gar nichts mehr. Bis GeoBasic auf Zeile 100 stößt ...

Die besagte Zeile weist die Einrichtung einer Schleife mit der Zählvariablen I an, die genau dreimal durchlaufen werden soll. Dabei wird I von 0 bis 2 gezählt. Die Zahl 3 sollte eigentlich bei Ihnen einen »Aha-Effekt« auslösen, wenn Sie sich die DATA-Befehle mal genauer anschauen. Richtig, es sind genau drei Vokabeln inklusive Übersetzungen darin enthalten!

Mit dem READ-Befehl der Zeile 110 wird nun das erste DATA-Element – eine Vokabel – gelesen und in das erste Element der Feldvariablen abgelegt. Anschließend wird der interne Zeiger auf das nächste Element gesetzt und eine weitere Zeichenkette (diesmal ist es die Übersetzung) in die Feldvariable übertragen.

Durch den NEXT-Befehl wird I weitergezählt und der Vorgang wiederholt, bis der Wert 2 überschritten ist. Aber nun zur Syntax von READ:

READ 'Variable'['Variable']...

Als Parameter zu READ können beliebige numerische oder Zeichenketten-Variablen, getrennt durch Kommata fungieren. Jeder READ-Befehl liest das Element ein, auf das der DATA-Zeiger weist, übergibt diesen in der angegebenen Variablen und verschiebt anschließend diesen Zeiger auf das nächste Element, bis zu dem Zeitpunkt, an dem kein Element mehr vorhanden ist. Wird dann noch einmal versucht, ein Element mit READ zu lesen, kehrt GeoBasic mit einem »Out of DATA«-Fehler zum Text-Editor zurück.

Natürlich muß der Typ der eingelesenen Konstanten mit dem der hinter READ angegebenen Variablen übereinstimmen; der Versuch, eine Zeichenkette in einer numerischen Variable unterzubringen, endet verständlicherweise mit einer Fehlermeldung. Weiteres Beispiel für READ:

```
10 READ A,B,C,D,E$,F%
20 DATA ,100,,200.2,"Hallo Du!",
```

4.9.3 RESTORE

Die Wichtigkeit des RESTORE-Befehls wird meist unterschätzt, da Sie ihn eigentlich nur brauchen, wenn Sie mehrere Male auf dieselben DATA-Elemente zugreifen.

Wir erinnern uns: Bei Benutzung des READ-Befehls wird immer ein Zeiger weitergesetzt, bis dann schließlich kein Element mehr vorhanden ist, das nicht schon einmal gelesen wurde. Versuchen Sie nun noch einmal, den READ-Befehl anzuwenden, so wird der Zeiger *nicht* automatisch auf das erste DATA-Element zurückgesetzt sondern der Fehler »Out of DATA« erscheint.

Was nun? Abhilfe bringt da der RESTORE-Befehl. Seine Syntax lautet:

RESTORE ['Ausdruck']

Als Ausdruck gelten hier beliebige numerische und ganzzahlige Konstanten, Variablen, Ausdrücke und natürlich auch Labels. Er bestimmt die Zeile, auf deren erstes DATA-Element der interne Zeiger gerichtet werden soll.

Wird kein Ausdruck angegeben, setzt GeoBasic den Zeiger auf das erste DATA-Element des Programms.

Möchten Sie aber den Zeiger auf eine bestimmte Zeile richten, muß diese immer als Parameter angegeben werden!

Nach dem Ausführen des RESTORE-Befehls können Sie dann wieder problemlos weiter mit READ arbeiten. Beispiel:

```
10  RESTORE 110
20  READ A$
30  PRINT A$
40
100 DATA "to swim"
110 DATA "to take"
```

Nach Ausführung dieses Programms wird »to take« auf dem Bildschirm ausgegeben, da der DATA-Zeiger durch Zeile 10 auf das Element in Zeile 110 gerichtet wird.

4.10 Funktionen

4.10.1 Mathematische Funktionen

GeoBasic stellt Ihnen eine ganze Reihe von mathematisch wichtigen Funktionen (wie z. B. Sinus oder Quadratwurzel) zur Verwendung in Ihren Programmen zur Verfügung.

ATN

Syntax: ATN('Ausdruck')

Diese Funktion liefert den Arcus Tangens (Gegenfunktion zum Tangens) des als Parameter anzugebenden numerischen Ausdrucks im Bogenmaß.

Das Ergebnis liegt im Bereich von –/2 bis /2. Beispiel:

```
10 PRINT ATN(2)
```

Auf dem Bildschirm wird »1.10714872« ausgegeben.

COS

Syntax: COS('Ausdruck')

Der Kosinus des numerischen im Bogenmaß angegebenen Ausdrucks wird übergeben. Beispiel:

```
10 PI = 3.14159265
20 PRINT COS(20)
30 Y = COS(4 * PI / 180)
```

In Zeile 30 wird innerhalb der Klammer eine Zahl vom Gradmaß in das Bogenmaß umgerechnet.

EXP

Syntax: EXP('Ausdruck')

Diese Funktion potenziert die Zahl e (e = ca. 2.71828) mit dem anzugebenden numerischen Ausdruck. Ist dieser größer als 88.0296919, wird ein »Overflow«-Fehler ausgelöst. Beispiel:

```
10 PRINT EXP(4)
```

Auf dem Bildschirm wird »54.59815« ausgegeben.

INT

Syntax: INT('Ausdruck')

Der Ganzzahl-Wert des anzugebenden numerischen Ausdrucks wird übergeben. Ist der Wert des Ausdrucks positiv, dann werden die Nachkommastellen einfach ignoriert, ist er hingegen negativ, wird die nächst kleinere ganze Zahl übergeben. Beispiel:

```
10 PRINT INT(10.5), INT(-10.5)
```

Die Werte »10« und »–11« erscheinen auf dem Bildschirm. INT ist vor allem im Zusammenhang mit Zufallszahlen (siehe RND weiter unten) sehr wirkungsvoll, da diese immer als Fließkommazahlen übergeben werden, und Sie meist ganze Zufallszahlen brauchen:

```
10 PRINT INT(RND(11))
```

LOG

Syntax: LOG('Ausdruck')

Diese Funktion errechnet den natürlichen Logarithmus des numerischen Ausdrucks, der größer als Null sein muß. Der Fehler »Illegal quantity« wird Sie ansonsten auf einen falschen Wert aufmerksam machen.

```
10 PRINT LOG(10 / 7)
```

RND

Syntax: RND('Ausdruck')

RND dient der Erstellung einer Zufallszahl zwischen Null und dem angegebenen numerischen Ausdruck. Die Zufallszahl hat immer die Form einer Fließkomma-Konstanten. Beispiel:

```
10 A = RND(5): PRINT A
20 A = INT(RND(10)) + 1
```

Das Zahlenrate-Spiel in Abschnitt 4.6.1 ist übrigens auch ein sehr gutes Beispiel für die Anwendungsmöglichkeiten von RND.

SGN

Syntax: SGN('Ausdruck')

SGN übergibt einen der folgenden Werte an Ihr Programm zurück:

1, wenn der angegebene numerische Ausdruck größer als Null ist, 0, wenn der Ausdruck gleich Null ist oder –1, wenn der Ausdruck kleiner als Null ist. Beispiel:

```
10 PRINT SGN(4-2)
20 X = 0: PRINT SGN(X)
30 X = -15: PRINT SGN(X)
```

Ausgegeben werden die Zahlen »1«, »0« und »–1«.

SIN

Syntax: SIN('Ausdruck')

Diese Funktion errechnet den Sinus des numerischen Ausdrucks im Bogenmaß. Beispiel:

```
10 PRINT SIN(4)
```

Auf dem Bildschirm erscheint die Zahl »–.756802495«.

SQR

Syntax: SQR('Ausdruck')

Diese Funktion berechnet die Quadratwurzel des Ausdrucks, der eine positive Zahl sein muß. Jeder Versuch, einen negativen Parameter zu übergeben, scheitert an einem »Illegal quantity«-Fehler. Beispiel:

```
10 FOR X = 1 TO 10
20         PRINT SQR(X)
30 NEXT X
```

TAN

Syntax: TAN('Ausdruck')

Es wird der Tangens des numerischen Ausdrucks im Bogenmaß errechnet. Beispiel:

```
10 X = 1 : Y = TAN(X * 2)
20 PRINT Y
```

Auf dem Bildschirm erscheint die Zahl »–2.18503986«.

4.10.2 Konvertierfunktionen

Wie Sie wissen, gibt es verschiedene Variablentypen, numerische Variablen (Fließkomma- und Ganzzahl-) sowie Zeichenketten-Variablen. Eine Konvertierung zwischen diesen Formaten nach dieser Methode:

A$ = Wert

oder

Var% = BAS$

erzeugt einen »Type mismatch«-Fehler. Da jedoch manchmal eine Umwandlung sehr sinnvoll ist, sind zu diesem Zweck in GeoBasic zwei Funktionen eingebaut worden:

STR$

Syntax: STR$('Ausdruck')

Möchten Sie eine numerische Variable in eine Zeichenketten-Variable übertragen, dann können Sie dies mit STR$ erledigen. Geben Sie den Ausdruck einfach als Parameter an. Beispiel:

```
10 A% = 16
20 A$ = STR$(A%)
30 B$ = STR$(3.14159265)
```

VAL

Syntax: VAL('Ausdruck')

Diese Funktion wandelt einen Zeichenketten-Ausdruck in eine Zahl um. Buchstaben und Sonderzeichen im Ausdruck können zur Verfälschung des Ergebnisses führen. Zur Erläuterung von VAL möchte ich noch einmal zum GET-Befehl (Abschnitt 4.2) zurückkehren:

Möchten Sie mit GET eine numerische Eingabe vom Benutzer holen, sollten Sie trotzdem eine Zeichenketten-Variable als Parameter zu GET angeben, um zu vermeiden, daß ein »Type mismatch«-Fehler ausgegeben wird, wenn ein Buchstabe eingeben wird. Anschließend können Sie dann die Eingabe mit VAL in eine numerische Variable übergeben:

```
10 PRINT "Bitte geben Sie eine Zahl von 0 bis 9 ein: ";
20
30 REPEAT
40       GET A$
50 UNTIL A$<>""
60
70 A = VAL(A$)
80 PRINT A
```

4.10.3 Sonderfunktionen für Zeichenketten

Die hier aufgeführten Funktionen erleichtern die Arbeit mit Teilen von bestimmten Zeichenketten-Konstanten, Variablen und Ausdrücken.

LEFT$

Syntax: LEFT$('String','Ausdruck')

Diese Funktion übergibt den linken Teil des Zeichenketten-Ausdrucks 'String', dessen Länge durch den numerischen Ausdruck definiert ist.

Der numerische Ausdruck muß eine ganze Zahl zwischen 0 und 255 sein. Ist er 0, dann wird eine leere Zeichenkette übergeben, ist er größer als oder gleich der Länge der Zeichenkette, so wird die ganze Zeichenkette übergeben. Beispiel:

```
10 A$ = "Berkeley Softworks"
20 B$ = LEFT$(A$, 8)
30 PRINT B$
40
50 PRINT LEFT$("Britta F.", 6)
```

Dieses Beispiel schreibt »Berkeley« und »Britta« auf den Bildschirm.

MID$

Syntax: MID$('String','Ausdruck',['Ausdruck'])

Ebenso wie LEFT$ übergibt diese Funktion eine Teilzeichenkette des Zeichenketten-Ausdrucks 'String'. Der erste numerische Ausdruck gibt die Anfangsposition dieser Teilzeichenkette an, der zweite numerische Ausdruck seine Länge.

Ist der erste Ausdruck größer als die Länge der Zeichenkette, dann wird eine leere Zeichenkette übergeben. Wird der zweite Ausdruck nicht angegeben, oder ist er größer als die Länge der Zeichenkette 'String' ab der Anfangsposition, dann wird der gesamte rechte Teil ab der Anfangsposition übergeben.

Wenn der zweite Ausdruck gleich Null ist, so liefert MID$ eine leere Zeichenkette.

Beide numerischen Ausdrücke müssen in dem Bereich von 0 bis 255 liegen. Beispiel:

```
10 A$ = "Ich wünsche Euch einen "
20 B$ = "schönenangenehmenguten"
30 C$ = " Tag!"
40
50 PRINT A$; MID$(B$, 8, 10); C$
```

Ausgegeben wird:

```
Ich wünsche Euch einen angenehmen Tag!
```

RIGHT$

Syntax: RIGHT$('String','Ausdruck')

Die letzte Funktion zur Übergabe einer Teilzeichenkette ist mit der LEFT$-Funktion bis auf eine einzige Kleinigkeit vollkommen identisch:

Der anzugebende Ausdruck gibt die Anzahl der Zeichen an, die von der rechten Seite der Zeichenkette übergeben werden sollen. Beispiel:

```
10 A$ = "Berkeley Softworks"
20 PRINT "Berkeley "; RIGHT$(A$, 9);
30 B$ = " programmierte auch "
40 C$ = "GEOSGEOS 128GeoBasic"
50 PRINT B$; RIGHT$(C$, 8); "."
```

Ausgegeben wird der Satz:

```
Berkeley Softworks programmierte auch GeoBasic.
```

4.10.4 Sonstige Funktionen

ASC

Syntax: ASC('String')

Diese Funktion übergibt den ASCII-Wert des ersten Zeichens der Zeichenkette 'String'.

'String' darf eine Zeichenketten-Konstante, eine Zeichenketten-Variable oder ein Zeichenketten-Ausdruck sein und muß mindestens ein Zeichen lang sein, sonst wird ein »Illegal quantity«-Fehler ausgegeben.

Eine Tabelle aller Zeichen und der zugehörigen ASCII-Werte finden Sie in Anhang 14.5. Beispiel:

```
10 PRINT ASC("A")
20 PRINT ASC("Hallo")
30 J$ = "Hallo"
40 PRINT ASC(J$)
```

Dieses kleine Programm gibt in Zeile 10 »65« (ASCII-Wert für »A«), »72« in Zeile 20 (ASCII-Wert für »H«, den ersten Buchstaben von »Hallo«) und ebenso in Zeile 40 »72« auf dem Bildschirm aus.

CHR$

Syntax: CHR$('Ausdruck')

Die CHR$-Funktion ist die genaue Umkehrung der ASC-Funktion, die ein ASCII-Zeichen in eine Zahl umwandelt (in den Wert des Zeichens). CHR$ verarbeitet den anzugebenden numerischen Ausdruck, der eine beliebige Zahl im Bereich von 0 bis 128 sein darf, und bestimmt das zu übergebende ASCII-Zeichen.

Über CHR$ läßt sich jeder Buchstabe, jede Zahl und jedes Sonderzeichen erzeugen. Eine Spezialität von CHR$, die in Verbindung mit dem PRINT-Befehl (Abschnitt 4.2) genutzt werden kann, ist es aber, den Schriftstil der auszugebenden Zeichen zu wechseln:

```
10 PRINT CHR$(14);"Dieser Satz erscheint unterstrichen!";CHR$(15)
20 A = 18
30 PRINT CHR$(A);"INVERS läßt sich einschalten mit
CHR$(";A;").";CHR$(A + 1)
```

Folgende Schriftstile sind möglich:

CHR$(14)	schaltet den Unterstreich-Modus ein
CHR$(15)	schaltet den Unterstreich-Modus wieder aus
CHR$(18)	aktiviert den Invers-Modus
CHR$(19)	deaktiviert den Invers-Modus wieder
CHR$(24)	schaltet den Fettdruck-Modus ein
CHR$(25)	schaltet die Kursiv-Schrift ein
CHR$(26)	aktiviert die Kontur-Schrift
CHR$(27)	kehrt zur normalen Schrift zurück

Bei Verwendung der Werte 1 bis 7 oder 29 bis 31 können Systemabstürze ausgelöst werden!

Eine Tabelle aller Zeichen und der zugehörigen ASCII-Werte finden Sie in Anhang 14.5. Beispiel:

```
10 A = 65
20 PRINT CHR$(A)
30 PRINT CHR$(A + 1)
```

Ausgegeben werden durch dieses Programm ein »A« und ein »B«.

FRE

Syntax: FRE('Ausdruck')

Diese Funktion übergibt die Größe des freien Speichers für Ihre Variablen in Byte. Der numerische Ausdruck kann beliebig sein, er wird nicht benötigt, muß aber aus Syntax-Gründen immer angegeben werden. Beispiel:

```
10 PRINT FRE(0)
20 A = 14: REM Verkleinerung des Variablenspeichers.
30 PRINT FRE(0)
```

LEN

Syntax: LEN('String')

Die Länge (in Buchstaben, nicht in Bildschirmpunkten) der Zeichenketten-Konstanten oder Variablen 'String' wird übergeben. Dabei werden auch Leerzeichen und Steuerzeichen berücksichtigt. Beispiel:

```
10 A$ = "Berkeley Softworks"
20 B = LEN(A$)
30 C = LEN("GeoBasic")
40 PRINT B, C
```

Ausgegeben werden die Zahlen »18« und »8«.

SPC

Syntax: SPC('Ausdruck')

SPC sollte immer im Zusammenhang mit einem PRINT oder einem PRASCII-Befehl stehen. Diese Funktion gibt Leerzeichen auf dem jeweiligen Ausgabegerät aus. Die Anzahl der Leerzeichen bestimmen Sie mit dem numerischen Ausdruck, der in einem Bereich von 0 bis 255 liegen darf.

Bei Nutzung von SPC mit dem Befehl PRASCII (zur Ausgabe von Zeichen auf dem Drucker) ergibt sich noch eine Besonderheit: Wird während der Ausgabe der Leerzeichen das Zeilenende erreicht, dann springt der Drucker in die nächste Zeile und die Ausgabe der Leerzeichen wird abgebrochen. Beispiel:

```
10 PRINT "Links und";
20 PRINT SPC(20); "weiter rechts."
```

Ausgegeben wird:

```
Links und weiter rechts.
```

TAB

Syntax: TAB('Ausdruck')

Die TAB-Funktion bewegt den Cursor auf die durch den numerischen Ausdruck definierte X-Position in der aktuellen Zeile.

Der Ausdruck muß im Bereich von 0 bis 319 liegen. Liegt die angegebene Position rechts von der aktuellen X-Position am Bildschirm, dann bewegt sich der Cursor an die gewünschte Position, liegt die neue Position aber links von der alten Position, so hat TAB keine Auswirkung.

TAB muß immer im Zusammenhang mit PRINT verwendet werden, ansonsten zeigt diese Funktion keine Wirkung. Beispiel:

```
10 PRINT "Vor TAB"; TAB(100); "Nach TAB"
```

TAB eignet sich hervorragend für die Erstellung von tabellarischen Ausführungen auf dem Bildschirm!

4.10.5 Definieren eigener Funktionen

GeoBasic ermöglicht Ihnen auch die Definition eigener Funktionen. So ersparen Sie sich dann später die wiederholte Verwendung von oftmals platzraubenden Formeln in Ihren Programmtexten.

DEF FN

Mit DEF FN können Sie eine Funktion definieren:

DEF FN 'Name'('Variable') = 'Ausdruck'

'Name' muß ein beliebiger zugelassener Variablenname sein (Abschnitt 4.1.2).

Die 'Variable' in Klammern ist aufgrund der Syntax immer anzugeben, es ist Ihnen aber nicht vorgeschrieben, daß Sie diese für die Berechnung in Ihrer Funktion nutzen. Rufen Sie später die Funktion auf (siehe unter FN), dann erhält die bei DEF FN in Klammern gesetzte Variable den Wert, den Sie beim Aufrufen der Funktion in Klammern gesetzt haben.

Der auf das »=«-Zeichen folgende Ausdruck kann ein beliebiger numerischer oder Zeichenketten-Ausdruck sein.

Eine eigene Funktion muß während des Ablaufs eines Programms erst definiert werden, bevor sie aufgerufen werden kann. Beispiel:

```
10 DEF FN SEC(x) = 1 / COS(x)
```

Es wird eine Funktion SEC definiert, die den Sekans von »x« liefert.

FN

Syntax: FN 'Name'('Ausdruck')

Diese Funktion übergibt den Wert einer vorher mit DEF FN erstellten Funktion. Der Ausdruck wird in die mit DEF FN bezeichnete Variable eingesetzt und die eigene Funktion wird automatisch ausgeführt.

Ergänzen wir zum besseren Verständnis das Beispiel für DEF FN um folgende Zeile:

```
20 PRINT FN SEC(4)
```

In Zeile 20 wird jetzt die in Zeile 10 definierte Funktion genutzt und der Sekans von »4« ausgegeben. Anstelle der »4« hätten Sie beispielsweise auch »2 * 2« oder »3 + 1« einsetzen können.

Beachten Sie, daß die Variable x nach Ablauf der Zeile 20 den Wert 4 hat, da sie als Variable im DEF-FN-Befehl angegeben wurde. Verwenden Sie die SEC-Funktion wie in Zeile 10 unseres Beispielprogramms also in eigenen Programmen, und die Variable x wird für andere wichtige Dinge in diesem Programm benötigt, dann müssen Sie eine andere (weniger wichtige) Variable in den DEF-FN-Befehl einsetzen!

4.11 Sonstige grundlegende Befehle

4.11.1 SYSINFO

SYSINFO übergibt in einer beliebigen Variablen bestimmte Systemzustände an das Programm.

Syntax: SYSINFO 'Ausdruck','Variable'

Der numerische Ausdruck ist eine ganze Zahl im Bereich von 0 bis 16 und bezeichnet den in der Variablen zu übergebenden Systemzustand:

Wert des Ausdrucks	*Übergibt in die Variable*
0	Benutzter Computer (0 = C 64, 1 = C 128)
1	Versionsnummer des GEOS-KERNELS
2	Nationalität des GEOS-KERNELS (0 = Amerikanisch, 1 = Deutsch ...)
3	Aktuelle Laufwerksnummer (8, 9)
4	Größtmögliche X-Koordinate des aktuellen Grafikbildschirms (319 bei GEOS 64)

Wert des Ausdrucks	*Übergibt in die Variable*
5	Größtmögliche Y-Koordinate des aktuellen Grafikbildschirms (199 bei GEOS 64)
6	Kennzahl für den zuletzt aufgetretenen Basic-Fehler
7	Aktuelle Stunde
8	Aktuelle Minute
9	Aktuelle Sekunde
10	Aktueller Monat
11	Aktueller Tag
12	Aktuelles Jahr
13	Grund, warum das Basic-Programm gestartet wurde (0 = ohne Zusätze, 1 = zugehöriges Dokument geöffnet, 2 = zugehöriges Dokument ist auszudrucken)
14	Dokumentname zu 13
15	Aktuelle Zeit (Stunden : Minuten : Sekunden)
16	Aktuelles Datum (Monat/Tag/Jahr)

Die Variable, die Sie bei Abfrage der Parameter 0 bis 13 benutzen, sollte eine numerische Variable sein, bei den restlichen ist eine Zeichenketten-Variable anzugeben.

Möchten Sie beispielsweise erfahren, auf welchem Computer Ihr Programm gerade betrieben wird, dann können Sie das so:

```
10 SYSINFO 0, A
20 IF A = 0 THEN PRINT "C 64": END
30 PRINT "C 128"
40 END
```

Weitere Einsatzmöglichkeiten des SYSINFO-Befehls finden Sie im nächsten Abschnitt und in Kapitel 6.

4.11.2 ONERR

Leider läßt es sich nie ausschließen, daß der Benutzer eines Programms Fehler macht, oder daß Ihnen bei der Programmierung Fehler unterlaufen.

Normalerweise beendet GeoBasic beim Eintreten eines solchen Fehlers ein laufendes Programm und aktiviert den Text-Editor, um eine entsprechende Meldung auszugeben (z. B. »Syntax error«). Möchten Sie, daß solche Fehler vom Programm selbst abgefangen und gemeldet werden können, dann sollten Sie sich des ONERR-Befehls bedienen.

Syntax: ONERR 'Ausdruck'

Der numerische Ausdruck bezeichnet die erste Zeilennummer einer Routine, die GeoBasic (per GOTO) anspringen soll, sobald ein Fehler auftritt.

ONERR fängt immer nur einen (den nächsten auftretenden) Fehler ab, danach wird der Befehl deaktiviert. Um mit allen weiteren Fehlern ebenso verfahren zu können, muß ONERR nach jeder Fehlerbehandlung wiederholt werden. Hierzu ein Beispiel:

```
10 ONERR @Fehler ... ... ...
39998 END
39999
40000 @Fehler: ...
40100 GOTO 10
```

Ist ein Fehler aufgetreten, dann wird Zeile 40000 angesprungen (an die Sie noch beliebige Zeilen anhängen können) und dann die Zeile 10 aufgerufen, um die Fehlerbehandlung erneut zu ermöglichen.

Leider läßt sich allein mit dem ONERR-Befehl nicht viel anfangen. Sie müssen nun noch einen Befehl haben, der zumindest die Nummer des aufgetretenen Fehlers an das Programm übergeben kann. Dieser Befehl wurde bereits im letzten Abschnitt beschrieben: SYSINFO.

Im Zusammenhang mit der Fehlerbehandlungsroutine interessiert uns nur der Parameter 6 dieses Befehls. Durch ihn erfahren wir, welcher Fehler aufgetreten ist und zum Aufrufen der ONERR-Routine geführt hat.

Zum Abfangen der Taste RUN STOP, mit der ein Programm unterbrochen werden kann, genügt folgende Routine:

```
10 Break = 47
20 ONERR @Fehler ... ...
39998 END 39999
40000 @Fehler:
40010 SYSINFO 6, checkErr
40020 IF checkErr = Break THEN PRINT "Sie können das Programm nicht
so beenden!": GOTO 20
```

4.11.3 REM

Sicherlich ist es sehr schwer, ein größeres GeoBasic-Programm wieder zu verstehen, wenn Sie es eine längere Zeit zur Seite gelegt haben und dann mit der Programmierung fortfahren möchten.

Gerade deshalb sollte man es sich angewöhnen, Kommentare in den Programmtext einzubinden, die beispielsweise darüber informieren, welche Funktionen bestimmte Routinen haben oder welche Parameter übergeben werden müssen.

Für diesen Zweck gibt es den REM-Befehl, der eigentlich kein richtiger Befehl ist, da er beim Ablauf des GeoBasic-Programms gar nichts bewirkt.

Seine Syntax lautet:

REM ['Text']

Beim Ablaufen Ihres GeoBasic-Programms wird der REM-Befehl und alles Weitere bis zum Ende der Programmzeile ignoriert. Beispiel:

```
10 PRINT "Bitte warten ..."
20 REM Gibt "Bitte warten ..." auf dem Bildschirm aus
```

Kommentare helfen Ihnen, Ihr Programm besser zu verstehen. Seien Sie aber trotzdem nicht zu großzügig mit Kommentaren, da auch sie Speicherplatz belegen und zudem noch die Ablaufgeschwindigkeit eines Programms herabsetzen.

4.11.4 END

Beim Auffinden des END-Befehls wird die Ausführung des Programms sofort beendet. Ein Tastendruck bringt den Benutzer dann zum Text-Editor zurück.

Syntax: END

Es ist nicht erforderlich, ein Programm mit einem END-Befehl abzuschließen, wird es aber trotzdem gemacht, so zeugt dies von einem guten Programmierstil. Sie können beliebig viele END-Befehle in Ihr Programm einbauen. Beispiel:

```
10 PRINT "Abbruch (J/N)?"
20
30 REPEAT
40  GET A$
50 UNTIL A$<>""
60
70 IF A$ = "J" OR A$ = "j" THEN END
80
90 REM Rest des Programms
```

5. Kapitel
Programmierung für Fortgeschrittene

Sie sollten dieses Kapitel erst dann durcharbeiten, wenn Sie sich in der Programmierung der Befehle und Funktionen, die in Kapitel 4 erläutert sind, sicher fühlen. Schreiben Sie notfalls noch einige kleinere Programme zur Übung.

Die in diesem Kapitel aufgeführten Befehle und Funktionen dienen dem Nutzen der Besonderheiten von C 64 und C128 und von GEOS. Bei richtigem Einsatz können Sie Basic-Programme entwickeln, die Sie vorher nie für möglich gehalten haben oder nur unter Einsatz von Assembler-Routinen verwirklichen konnten.

5.1 Grafiken

GEOS ist ein grafisches System. Grundsätzlich läuft es komplett im hochauflösenden Grafik-Modus des C 64 und C 128 ab.

Deshalb wurde auch eine Vielzahl von Befehlen zur Unterstützung der hochauflösenden Grafik in GeoBasic-Programmen geschaffen:

5.1.1 CLS

Syntax: CLS

Der CLS-Befehl bewirkt ein Löschen des Grafikbildschirms. Ist ein Fenster mit dem WINDOW-Befehl definiert worden, dann wird nur das Fenster gelöscht.

Es ist nützlich, ein Programm mit einem CLS-Befehl zu beginnen, um sich eine leere Fläche für die Bildschirmausgaben zu schaffen. Beispiel:

```
10 CLS
```

5.1.2 BITMAP

Syntax: BITMAP 'String','Ausdruck','Ausdruck'['Ausdruck']

Mit dem BITMAP-Befehl können Sie eine Grafik, die Sie vorher im Grafik-Editor (Kapitel 10) erstellt haben, auf den Bildschirm ausgeben lassen. Die Zeichenkette 'String' muß der Name sein, den Sie der Grafik im Grafik-Editor gegeben haben.

Der erste numerische Ausdruck steht für die X-Position der linken oberen Ecke der Grafik auf dem Bildschirm (0 bis 39) und der zweite numerische Ausdruck bezeichnet die zugehörige Y-Position (0 bis 199). Wenn Sie eine Grafik auf dem Bildschirm so plazieren, daß sie über den rechten oder unteren Rand hinausragt, dann werden die überstehenden Teile einfach abgeschnitten.

Der optionale numerische Ausdruck gibt an, ob die Grafik »deckend« (0) oder »transparent« (< > 0) auf den Bildschirm gebracht werden soll. Deckend bedeutet, daß die unter der Grafik liegende Fläche vollkommen gelöscht wird. Fügen Sie eine Grafik hingegen transparent ein, dann wird die Grafik über den bestehenden Untergrund gelegt, ohne daß dieser dabei gelöscht wird. Geben Sie diesen Parameter nicht an, dann setzt GeoBasic automatisch das »deckende« Verfahren ein. Beispiel:

```
10 BITMAP "Bild1", 10, 20, 1
20 REM Gibt die Grafik "Bild1" an der Position X: 10, Y: 20 transpa-
rent aus
```

Sie müssen vor dem Starten des Programms eine Grafik mit dem Namen »Bild1« im Grafik-Editor erstellen, da sonst GeoBasic eine Fehlermeldung ausgibt!

5.1.3 POINT

Syntax: POINT 'Ausdruck','Ausdruck'

Dieser Befehl setzt oder löscht einen Punkt des Bildschirms. Der erste numerische Ausdruck definiert die X-Position (0 bis 319), der zweite numerische Ausdruck die Y-Position (0 bis 199). Mit dem SETCOL-Befehl können Sie bestimmen, ob der Punkt gesetzt oder gelöscht werden soll (siehe Abschnitt 5.1.8). Beispiel:

```
10 POINT 10, 10
```

5.1.4 LINE

Syntax: LINE 'Ausdruck','Ausdruck' TO 'Ausdruck','Ausdruck'

LINE zeichnet eine Linie auf den Bildschirm. Die beiden ersten numerischen Ausdrücke definieren X- und Y-Position des Startpunktes der Linie, die numerischen Ausdrücke hinter TO sind die X- und Y-Position des Endpunktes der Linie. Die X-Koordinaten müssen zwischen 0 und 319 liegen, die Y-Koordinaten zwischen 0 und 199.

Ob die Linie gezeichnet oder gelöscht werden soll, bestimmen Sie mit SETCOL (Abschnitt 5.1.8). Beispiel:

```
10 x1 =   0: y1 =   0
20 x2 = 319: y2 = 199
30 LINE x1, y1 TO x2, y2
40 LINE 10, 10 TO 10, 90
```

5.1.5 PATTERN

Syntax: PATTERN 'Ausdruck'

Dieser Befehl bestimmt das Füllmuster, mit dem gefüllte Rechtecke gezeichnet werden sollen (Abschnitt 5.1.6). Der numerische Ausdruck muß eine ganze Zahl zwischen 0 und 33 sein.

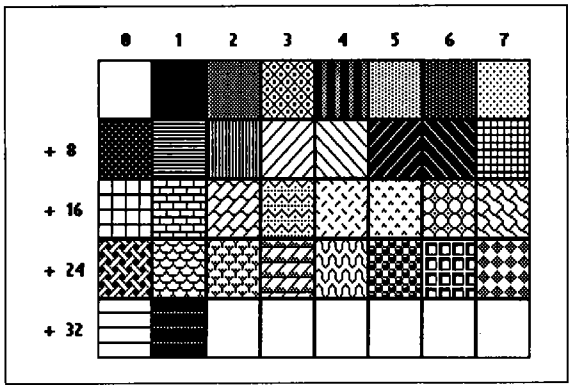


Bild 5.1: Die GEOS-Füllmuster

Beispiel:

```
10 PATTERN 2
```

Bestimmt das Füllmuster, mit dem die meisten GEOS-Programme einschließlich Desktop den Hintergrund füllen.

5.1.6 RECT

Syntax: RECT 'Ausdruck','Ausdruck','Ausdruck','Ausdruck'

RECT zeichnet ein gefülltes Rechteck auf den Bildschirm.

Die beiden ersten numerischen Ausdrücke bestimmen die X- und Y-Position der oberen linken Ecke des Rechtecks, die beiden anderen die X- und Y-Position der unteren rechten Ecke. Die X-Koordinaten dürfen in dem Bereich von 0 bis 319 liegen, die Y-Koordinaten im Bereich von 0 bis 199. Das Füllmuster bestimmen Sie mit dem PATTERN-Befehl (Abschnitt 5.1.5).

Beispiel:

```
10 PATTERN 10
20 RECT 10, 10, 40, 40
```

5.1.7 FRECT

Syntax: FRECT 'Ausdruck','Ausdruck','Ausdruck','Ausdruck'

Der FRECT-Befehl zieht einen Rahmen um einen beliebigen rechteckigen Bildschirmausschnitt. Die beiden ersten numerischen Ausdrücke bestimmen die X- und Y-Position der oberen linken Ecke des Rahmens, die beiden anderen die X- und Y-Position der unteren rechten Ecke. Die X-Koordinaten dürfen in dem Bereich von 0 bis 319 liegen, die Y-Koordinaten im Bereich von 0 bis 199. Das zu verwendende Linien-Füllmuster bestimmen Sie mit Hilfe des SETCOL Befehls (Abschnitt 5.1.8).

Beispiel:

```
10 SETCOL 255
20 FRECT 10, 10, 40, 40
```

5.1.8 SETCOL

Syntax: SETCOL 'Ausdruck'

Dieser Befehl setzt Parameter für bestimmte Grafikbefehle, die auf SETCOL folgen. Der numerische Ausdruck muß zwischen 0 und 255 liegen.

1. Ist der Ausdruck gleich 0, dann wird ein Punkt mit dem POINT-Befehl (Abschnitt 5.1.3) gezeichnet, ist der Ausdruck ungleich 0, dann wird der Punkt gelöscht.
2. Für LINE (Abschnitt 5.1.4) gilt das unter 1. Beschriebene entsprechend.

3. Bei farbigen Rechtecken (COLRECT, Abschnitt 5.1.9) bestimmt der Ausdruck die Zeichenfarben: die unteren vier Bit die Hintergrundfarbe, die oberen vier Bit die Vordergrundfarbe. Es gibt 16 verschiedene Farben (durchnumeriert von 0 bis 15). Um die Vordergrundfarbe 5 und die Hintergrundfarbe 8 zu verwenden, schreiben Sie den Befehl folgendermaßen: SETCOL 8 + 16 * 5.

4. Die acht Bit des angegebenen Ausdrucks bestimmen bei rechteckigen Rahmen (FRECT, Abschnitt 5.1.7) das Linien-Füllmuster. Ist das Bit gesetzt, dann wird der zugehörige Punkt der Linie gezeichnet, ist es gelöscht, dann wird der Punkt ebenfalls gelöscht. Die Zahl 255 (in Bits [binär]: 11111111) hat alle Bits gesetzt und bewirkt damit auch eine durchgezogene Linie, die Zahl 85 (binär: 01010101) produziert eine gepunktete Linie. Die Umrechnung ist ziemlich einfach:

Bit:	7	6	5	4	3	2	1	0
Wert:	0	1	0	1	0	1	0	1
Dezimal:	128	64	32	16	8	4	2	1

Nun zählt man die Dezimalzahlen aller gesetzten Bits (1) zusammen und erhält die Zahl, die man mit SETCOL angeben muß:

64 + 16 + 4 + 1 = 85

Beispiel:

```
10 SETCOL 85
20 FRECT 40, 40, 80, 80
```

5.1.9 COLRECT

Syntax: COLRECT 'Ausdruck','Ausdruck','Ausdruck','Ausdruck'

COLRECT färbt einen rechteckigen Bildschirmbereich ein. Die ersten beiden Ausdrücke bestimmen die X- und Y-Position der linken oberen Ecke des Rechtecks, die beiden anderen Ausdrücke die X- und Y-Position der rechten unteren Ecke des Rechtecks. Die X-Koordinaten dürfen dabei zwischen 0 und 39 liegen, die Y-Koordinaten zwischen 0 und 24. Mit SETCOL (Abschnitt 5.1.8) stellen Sie die gewünschten Zeichenfarben ein. Beispiel:

```
10 SETCOL 5 + 4 * 16
20 COLRECT 4, 5, 6, 7
```

5.1.10 INVRECT

Syntax: INVRECT 'Ausdruck','Ausdruck','Ausdruck','Ausdruck'

Ein rechteckiger Bildschirmausschnitt wird invertiert, d. h., daß in diesem Bereich jedes Pixel, das gesetzt ist, gelöscht wird und jedes Pixel, das gelöscht ist, gesetzt wird.

Die ersten beiden Ausdrücke bestimmen die X- und Y-Position der linken oberen Ecke des Rechtecks, die beiden anderen Ausdrücke die X- und Y-Position der rechten unteren Ecke des Rechtecks. Die X-Koordinaten dürfen dabei zwischen 0 und 319 liegen, die Y-Koordinaten zwischen 0 und 199. Beispiel:

```
10 INVRECT 10, 10, 40, 40
```

5.1.11 WINDOW

Syntax: WINDOW 'Ausdruck','Ausdruck','Ausdruck','Ausdruck'

Der WINDOW-Befehl definiert ein Ausgabefenster auf dem Bildschirm. Alle folgenden Bildschirmausgaben durch PRINT erfolgen nur noch in dem Fenster.

Die ersten beiden Ausdrücke bestimmen die X- und Y-Position der linken oberen Ecke des Rechtecks, die beiden anderen Ausdrücke die X- und Y-Position der rechten unteren Ecke des Rechtecks. Die X-Koordinaten dürfen dabei zwischen 0 und 319 liegen, die Y-Koordinaten zwischen 0 und 199. Beispiel:

```
10 WINDOW 16, 16, 303, 183
20 PRINT "Dieser Text wird im definierten Fenster ausgegeben!"
```

Ein mit WINDOW erzeugtes Fenster ist kein vom Hintergrund unabhängiges Fenster, das Sie problemlos wieder verschwinden lassen können, wenn Sie es nicht mehr brauchen, und wobei automatisch der Bildschirm wieder in den Zustand vor Erstellen des Fenster zurückversetzt wird. Der WINDOW-Befehl ist vielmehr nur zum Setzen der Grenzen des Bereichs da, in dem mit PRINT Bildschirmausgaben erfolgen können.

Möchten Sie ein Fenster, welches unabhängig vom Hintergrund ist und nach Gebrauch wieder spurlos verschwindet (eine Dialogbox), dann verwenden Sie zur Erstellung den Dialogbox-Editor (Kapitel 10) und die in Abschnitt 5.4 erläuterten Befehle, um Dialogboxen in Ihre Programme zu integrieren.

5.2 Menüs

Aus Menüs (Abschnitt 7.1) können in GEOS-Applikationen normalerweise übergeordnete Funktionen ausgewählt werden. Hierzu gehören beispielsweise die Dateiverwaltungsfunktionen »schließen, aktualisieren ...« (üblicherweise im Menü »Datei«) und die Funktionen zur Erstellung von Scrap-Dateien im »Edit«-Menü.

Auch in GeoBasic wird die Bereitstellung von Menüs in eigenen Programmen unterstützt. Die Verwaltung dieser Menüs wird dabei komplett von GEOS übernommen. Sie müssen nur noch in Ihre Programme die Reaktionsroutinen für die Menüpunkte integrieren. Dadurch gestaltet sich der Einbau – wie Sie weiter unten auch sehen werden – denkbar einfach!

Es ist dazu natürlich nötig, seine Programme an bestimmte Anforderungen anzupassen. Sie sollten sich in diesem Zusammenhang unbedingt die Ausführungen über die GEOS-Mainloop in Abschnitt 5.11 ansehen.

5.2.1 MENU

Syntax: MENU 'String'

MENU plaziert ein vorher mit dem Menü-Editor (Kapitel 7) erstelltes Menü in der oberen linken Ecke des Bildschirms. Die Zeichenketten-Konstante oder Variable 'String' enthält den Namen des Menüs, den Sie im Menü-Editor bestimmt haben.

Allein mit dem MENU-Befehl ist die korrekte Funktion des Menüs natürlich noch nicht gegeben. Sie müssen auch alle Routinen einbauen, die Sie bei den einzelnen Funktionen im Menü-Editor angegeben haben. Diese Routinen müssen mit einem RETURN-Befehl enden, da sie durch GeoBasic mit einem GOSUB aufgerufen werden.

Es kann immer nur eine Menüleiste aktiv sein. Wenn Sie zwei MENU-Befehle in Ihrem Programm verwenden, dann ist immer die durch den zuletzt ausgeführten MENU-Befehl bezeichnete Menüleiste aktiv.

MENU ist nur zum Zeichnen des Menüs auf dem Bildschirm zuständig. Die Kontrolle der Mausklicks auf dem Menü erfolgt durch die GEOS-Mainloop (Abschnitt 5.11), die Sie mit dem MAINLOOP-Befehl aktivieren müssen, nachdem das Menü gezeichnet wurde. Beispiel:

```
10 REDRAW 20
15 REM Nach einem Hilfsprogramm soll Zeile 20 aufgerufen werden.
16
20 PATTERN 2
25 RECT 0, 0, 319, 199
26
```

```
30      MENU "Menü"
40      MAINLOOP
50      REM Kontrolle an GEOS-Mainloop übergeben.
9999
10000 REM Hier finden Sie alle Routinen zu den Menüfunktionen.
10050
10100 @info:
10110 REM Routine zum Menüpunkt »program info«, »Geos«-Abrollmenü
10120 DIALOG "InfoB"
10130 RETURN
10150
10200 @quit:
10210 REM Routine zum Menüpunkt »Ende«, »Datei«-Abrollmenü
10220 END
10230 REM Da diese Routine das Programm verläßt, darf sie nicht mit
10240 REM RETURN enden!
```

Das Programm setzt voraus, daß Sie ein Menü mit dem Namen »Menü« im Menü-Editor erstellt haben, das zwei Abrollmenüs (»Geos« und »Datei«) mit je einem Menüpunkt enthält. Der Menüpunkt »program info« (im »Geos«-Abrollmenü) muß zur Routine »@info« verzweigen, der Menüpunkt »Ende« im »Datei«-Abrollmenü muß zur Routine »@quit« verzweigen.

Außerdem müssen Sie eine Dialogbox »InfoB« mit dem Dialogbox-Editor erstellt haben. Diese kann den Info-Dialogboxen von GeoPaint oder GeoWrite nachempfunden sein.

5.2.2 REDRAW

Syntax: REDRAW 'Ausdruck'

Wenn Sie in Ihrem GeoBasic-Programm ein Menü verwenden (Abschnitt 5.1), dann können aus dem »Geos«-Menü auch Hilfsprogramme gestartet werden. Diese Hilfsprogramme verändern den Bildschirminhalt, stellen ihn aber nicht mehr wieder her, wenn zu Ihrem Programm zurückgekehrt wird.

REDRAW ermöglicht Ihnen, daß Sie selbst den Bildschirm wiederherstellen.

Der numerische Ausdruck bezeichnet die Zeilennummer, an die GeoBasic mit GOTO verzweigen soll, sobald ein Hilfsprogramm beendet wurde. Die dort stehende Routine kann dann z. B. die Menüs und Piktogramme wiederherstellen. Beispiel:

```
10      REDRAW @nachDA ... ...
40000 @nachDA:
40010 REM Stellt den Bildschirm nach einem Hilfsprogramm wieder her
40020 MENU "Stoff"
40030 PRINT "Alles klar!"
40040 MAINLOOP
```

5.3 Piktogramme

Neben den Menüleisten stellen GEOS-Applikationen ihre Funktionen auch unter Zuhilfenahme von Piktogrammen (Abschnitt 9.1) dem Benutzer zur Verfügung. Bei den durch Piktogrammen bezeichneten Funktionen handelt es sich zumeist um die eigentlichen Editierfunktionen (z. B. Löschen, Invertieren, Weiter- bzw. Zurückblättern ...).

Durch den Einsatz von Piktogrammen in GeoBasic-Programmen können Sie diese sehr anwenderfreundlich gestalten. So können auch Anfänger diese Programme sofort ohne Schwierigkeiten einsetzen und müssen nicht erst eine umfangreiche Anleitung wälzen, um die vielen Tastenkombinationen zum Aufrufen von Funktionen zu lernen.

GEOS verwaltet die Piktogramme ebenso wie die Menüs nach der Aktivierung vollkommen selbständig. Aus diesem Grunde muß der Aufbau Ihres Programms ein wenig vom »normalen« Aufbau abweichen. Lesen Sie dazu in Abschnitt 5.11 nach.

5.3.1 ICON

Syntax: ICON 'String'

ICON sorgt dafür, daß die Piktogramme einer Liste, die Sie vorher im Piktogrammlisten-Editor (Kapitel 9) erstellt haben, auf den Bildschirm gebracht werden.

Die Zeichenketten-Konstante oder Variable 'String' enthält den Namen der Piktogrammliste, den Sie im Piktogrammlisten-Editor bestimmt haben.

Um die korrekte Funktion der Piktogramme zu gewährleisten, müssen natürlich alle Routinen, die Sie im Piktogrammlisten-Editor als Sprungziele angegeben haben, vorhanden sein. Diese Routinen werden von GeoBasic aus mit einem GOSUB aufgerufen und müssen darum mit einem RETURN zurückkehren.

Wenn Sie den ICON-Befehl mehrmals anwenden, sind immer die zuletzt auf den Bildschirm gebrachten Piktogramme aktiviert – auch wenn noch alte Piktogramme auf dem Bildschirm zu sehen sind.

Wie auch beim MENU-Befehl muß die GEOS-Mainloop (Abschnitt 5.11) mit dem MAINLOOP-Befehl eingeschaltet werden, sobald das Anklicken von Piktogrammen automatisch kontrolliert werden soll. Beispiel:

```
10 ICON "Pikto"
20 MAINLOOP
30    REM Die Kontrolle wird hier an die GEOS-Mainloop übergeben.
9999
```

```
10000 REM Hier befinden sich die Routinen zu den Piktogrammen
10050
10100 @ic1:
10110 REM Beispielpiktogramm 1
10120 PRINT "Piktogramm 1 wurde geklickt"
10130 RETURN
10150
10200 @ic2:
10210 REM Beispielpiktogramm 2
10220 END
10230 REM Da bei Aufruf dieses Piktogramms das Programm beendet wird,
10240 REM muß kein RETURN-Befehl die Routine abschließen
```

Das Programm setzt voraus, daß Sie eine Piktogrammliste »Pikto« im Piktogrammlisten-Editor mit zwei Piktogrammen entworfen haben. Das erste verzweigt zur Routine »@ic1«, das zweite zur Routine »@ic2«.

5.4 Dialogboxen

Dialogboxen (Abschnitt 8.1) können zu vielen Zwecken genutzt werden. Mit ihnen kann der Benutzer nach Anwahl einer bestimmten Funktion beispielsweise gefragt werden, ob er die Funktion auch wirklich ausgeführt haben möchte (»Sind Sie sicher?«). Oder der Benutzer kann aus einer Dialogbox den Namen einer Datei anwählen usw.

Trotz der vielfältigen Einsatzmöglichkeiten von Dialogboxen ist deren Einbau in eigene Programme völlig unproblematisch. In den allermeisten Fällen reicht das Definieren der Dialogbox im Dialogbox-Editor (Kapitel 8) und ein einzelner Befehl im GeoBasic-Programm, um sie zu aktivieren. Bei den bereits vordefinierten Dialogboxen genügt ein Befehl im Programm, und schon läuft alles wie gewünscht.

5.4.1 DIALOG

Syntax: DIALOG 'String'[,'Variable']

Mit DIALOG können Sie eine Dialogbox, die Sie im Dialogbox-Editor (Kapitel 8) erstellt haben, auf den Bildschirm bringen. Die Zeichenketten-Konstante oder Variable 'String' enthält den Namen der Dialogbox.

In der optional anzugebenden Fließkomma-Variable wird, wenn die Dialogbox durch Anklicken eines Piktogramms verlassen wurde, die zugehörige Kennzahl übergeben:

Kennzahl	*Piktogramm*
1	OK
2	Abbruch
3	Ja
4	Nein
5	Öffnen
6	Disk
20 – 255	Eigenes Piktogramm

Weitere Details zu den Kennzahlen finden Sie in Kapitel 8. Beispiel:

```
10 DIALOG "Test", Klick
20 IF Klick = 1 THEN PRINT "»OK« angeklickt!": END
30 PRINT "»Abbruch« angeklickt."
40 END
```

Das Programm setzt voraus, daß Sie eine Dialogbox namens »Test« entwickelt haben, die ein »OK«- und ein »Abbruch«-Piktogramm enthält.

5.4.2 DBFILE

Syntax: DBFILE 'Variable'

Es gibt keine Möglichkeit, mit dem Dialogbox-Editor Dialogboxen zur Auswahl von Dateien von Diskette zu entwerfen. Für diesen Zweck gibt es eine spezielle fest eingebaute Dialogbox, die mit DBFILE aufgerufen wird.

In der Zeichenketten-Variablen 'Variable' wird nach dem Löschen der Dialogbox der Name der ausgewählten Datei übergeben. Wird in der Dialogbox »Abbruch« angewählt, dann ist die Zeichenkette leer.

Auch wenn keine Datei zur Auswahl stand, wird eine leere Zeichenkette übergeben.

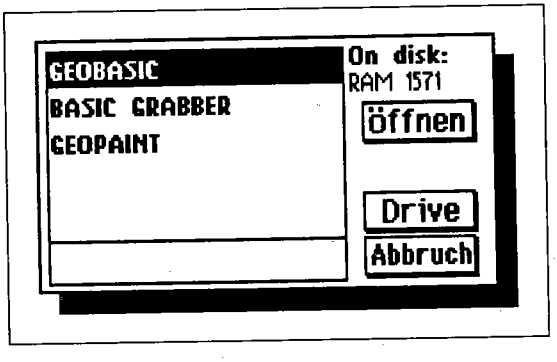


Bild 5.2: Die mit DBFILE erzeugte Dialogbox

Den Typ und die Klasse der anzuzeigenden Dateien bestimmen Sie mit dem HEADER-Befehl (Abschnitt 5.8.1).

Sollte der Benutzer eine Datei ausgewählt haben, dann wird diese automatisch von GeoBasic geöffnet. Eine Anwendung des OPEN-Befehls entfällt hier also (Abschnitt 5.8.3). Vor der Benutzung von DBFILE müssen Sie eine eventuell geöffnete Datei zur Vermeidung eines Datenverlustes immer schließen (Abschnitt 5.8.4)! Beispiel:

```
10 HEADER 6, ""
20 DBFILE A$
30 IF A$ = "" THEN PRINT "Keine Datei wurde ausgewählt!": END
40 PRINT "Die Datei ";A$;" wurde ausgewählt."
50 END
```

5.4.3 DBSTRN

Syntax: DBSTRN 'String', 'Variable'

DBSTRN bringt eine spezielle Dialogbox auf den Bildschirm, in der der Benutzer etwas eingeben kann. Die Zeichenketten-Konstante oder Variable 'String' enthält den Text, der in der Dialogbox zur Erklärung der Eingabe ausgegeben werden soll.

Die Zeichenketten-Variable 'Variable' enthält nach der Rückkehr aus der Dialogbox den vom Benutzer eingegebenen Text. Wurde das Piktogramm »Abbruch« angeklickt, oder haben Sie ohne jegliche Eingabe auf Return gedrückt, dann bleibt die Variable leer.

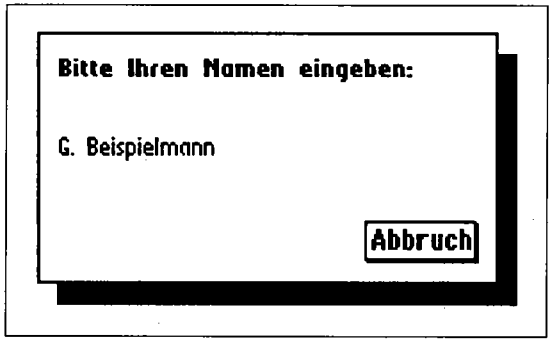


Bild 5.3: Eine mit DBSTRN erzeugte Dialogbox

Beispiel:

```
10 DBSTRN "Bitte geben Sie Ihren Namen ein:", A$
20 IF A$ = "" THEN END
```

```
30 PRINT "Sie heißen also "; A$; "."
40 END
```

5.5 Maus

Das Betriebssystem GEOS und die meisten Applikationen sind für die Bedienung durch Maus oder Joystick ausgelegt. Nur in unumgänglich notwendigem Maße wird auch die Tastatur Ihres C 64 und C 128 unterstützt.

Auch in GeoBasic finden Sie eine ganze Reihe von Befehlen und Funktionen speziell zur Beeinflussung und Abfrage der Maus.

5.5.1 MOUSE

Syntax: MOUSE 'Ausdruck','Ausdruck','Ausdruck'

Dieser Befehl schaltet den Mauszeiger entweder aus oder ein.

Ist der numerische Ausdruck gleich Null, dann wird der Zeiger abgeschaltet, ist er ungleich Null, dann wird der Mauszeiger eingeschaltet.

Der zweite numerische Ausdruck gibt die gewünschte X-Position (0 – 319) des Mauszeigers an, der dritte numerische Ausdruck die zugehörige Y-Position (0 – 199) an. Beide Ausdrücke müssen immer – also auch beim Abschalten des Mauszeigers – angegeben werden. Beispiel:

```
10 MOUSE 0, 0, 0
20 REM Der Mauszeiger ist jetzt deaktiviert
30 MOUSE 1, 10, 10
40 REM Jetzt ist er wieder eingeschaltet
```

5.5.2 BUTTON

Syntax: BUTTON 'Ausdruck'

Durch den BUTTON-Befehl kann eine Routine deklariert werden, die bei jedem Mausklick aufgerufen wird. Der anzugebende numerische Ausdruck stellt die erste Zeilennummer der bei einem Mausklick aufzurufenden Unterroutine dar.

Beachten Sie in diesem Zusammenhang wieder die Ausführungen über die GEOS-Mainloop (Abschnitt 5.11), die Ihre Routine aufruft.

Die Routine muß – nachdem sie ihre Arbeit beendet hat – mit einem RETURN-Befehl abschließen.

Alle Mausklicks, also auch Mausklicks auf der Menüleiste und auf Piktogrammen führen zur Aktivierung der Routine. Deshalb sollte zuerst von dieser Routine eine Bereichsüberprüfung stattfinden, um diese Klicks abzufangen. Beispiel:

```
10 BUTTON @Klick
20 MAINLOOP
30     REM Kontrolle an GEOS-Mainloop übergeben. ... ...
20000 @Klick:
20010 PRINT "Hi!"
20020 RETURN
```

5.5.3 MOUSEIN

Syntax: MOUSEIN('Ausdruck','Ausdruck','Ausdruck','Ausdruck')

Die MOUSEIN-Funktion überprüft, ob sich die Maus innerhalb eines bestimmten rechteckigen Bereiches aufhält.

Die X- und Y-Position der linken oberen Ecke des Bereiches bestimmen Sie durch die ersten beiden numerischen Ausdrücke, die X- und Y-Position der rechten unteren Ecke durch die beiden letzten numerischen Ausdrücke. Die X-Koordinaten müssen zwischen 0 und 319, die Y-Koordinaten zwischen 0 und 199 liegen.

MOUSEIN übergibt –1 (wahr), wenn sich der Mauszeiger im Bereich befindet. Im anderen Fall wird 0 (falsch) übergeben. Beispiel:

```
10 Mpos = MOUSEIN(10, 20, 50, 100)
20 IF NOT(Mpos) THEN GOTO 10
30 PRINT "OK!"
40 END
```

5.5.4 MOUSEX

Syntax: MOUSEX('Ausdruck')

Die MOUSEX-Funktion übergibt die aktuelle X-Position des Mauszeigers (zwischen 0 und 319).

Der Ausdruck kann jeder beliebige gültige numerische Ausdruck sein – er muß nur aus Syntax-Gründen angegeben werden und wird nicht wirklich benötigt. Beispiel:

```
10 PRINT MOUSEX(1),;
20 GOTO 10
```

5.5.5 MOUSEY

Syntax: MOUSEY('Ausdruck')

Die MOUSEY-Funktion übergibt die aktuelle Y-Position des Mauszeigers (zwischen 0 und 199).

Der Ausdruck kann jeder beliebige gültige numerische Ausdruck sein – er ist nur aus Syntax-Gründen anzugeben. Beispiel:

```
10 PRINT MOUSEY(0),;
20 GOTO 10
```

5.6 Sprites

GeoBasic enthält neben einem Sprite-Editor (Kapitel 11) auch noch leistungsstarke Befehle, die die Programmierung von Sprites vereinfachen.

5.6.1 SPRCOL

Syntax: SPRCOL 'Ausdruck','Ausdruck'

Sprites können dreifarbig sein, zwei von diesen Farben müssen aber bei allen Sprites gleich sein. Diese beiden Farben können Sie mit dem SPRCOL-Befehl (oder durch Auswahl von »sprcol« aus dem »Edit«-Menü (Abschnitt 3.4.3) bestimmen.

Die numerischen Ausdrücke nach SPRCOL bezeichnen die Farben und dürfen zwischen 0 und 15 liegen:

0	schwarz	8	orange
1	weiß	9	braun
2	rot	10	hellrot
3	grün	11	dunkelgrau
4	violett	12	grau
5	dunkelgrün	13	hellgrün
6	blau	14	hellblau
7	gelb	15	hellgrau

Beispiel:

```
10 SPRCOL 2, 5
```

5.6.2 SPRITE

Syntax: SPRITE 'String'

Bringt Sprites, die mit dem Sprite-Editor (Kapitel 11) erstellt wurden, auf den Bildschirm.

Die Zeichenketten-Konstante oder Variable 'String' muß den Namen des Sprites enthalten, unter dem Sie es im Sprite-Editor gespeichert haben. Ist eine bestimmte Bewegung, Animation oder ein Zusammenschluß von mehreren Sprites zu einem großen Sprite definiert, so werden diese Bewegungen und Animationen usw. sofort von GeoBasic ausgeführt.

Das Sprite erscheint nach der Aktivierung an der Stelle, die Sie im Sprite-Editor angegeben haben. Beispiel:

```
10 SPRITE "NewSp"
20 GOTO 20
```

Sie müssen vor Aktivierung dieses Beispiels ein Sprite mit dem Namen »NewSp« entworfen haben.

Die Endlosschleife in Zeile 20 ist in diesem Beispiel enthalten, um die Beendigung des Programms und damit das Stoppen sämtlicher Spritebewegungen und Animationen zu verhindern!

5.6.3 SPRT

Syntax: SPRT 'Ausdruck','Ausdruck','Ausdruck'

Mit dem SPRT-Befehl können Sie bestehende Einstellungen zu Sprites entweder verändern oder abfragen.

Der Wert des ersten numerischen Ausdrucks stellt die Nummer des Sprites dar, auf das zugegriffen werden soll (3 bis 8). Der nächste numerische Ausdruck enthält die Nummer des Parameters, der geändert oder abgefragt werden soll:

0	Farbe
1	X-Geschwindigkeit (oder X-Position des Zielpunkts bei Verwendung von »trail«)
2	Y-Geschwindigkeit (oder Y-Position des Zielpunkts bei Verwendung von »trail«)
3	X-Position (0 bis 512)
4	Y-Position (0 bis 256)
5	»timeout« Zeit

Die Bedeutung der einzelnen Sprite-Parameter entnehmen Sie bitte Kapitel 11, in dem der Sprite-Editor beschrieben ist.

Wird dieser Parameter mit einem negativen Vorzeichen angegeben, dann wird der Wert abgefragt und Ihnen übergeben, ansonsten wird der Wert nach Ihren Wünschen neu gesetzt.

Hinweis: Es gibt keine negative Null, die Farbe eines Sprites kann also nicht abgefragt werden!

Bei einer Abfrage muß als letzter Parameter kein numerischer Ausdruck, sondern eine numerische Variable übergeben werden. In dieser speichert GeoBasic dann die von Ihnen gewünschte Information. Setzen Sie aber einen Sprite-Parameter neu, dann genügt ein beliebiger numerischer Ausdruck.

Möchten Sie beispielsweise, daß Ihr Sprite bei der Bewegung auch noch beschleunigt, dann können Sie in Ihr Programm folgendes einfügen:

```
500 FOR xGeschw = 50 TO 200 STEP 10
510       SPRT 3, 1, xGeschw
520       REM Erhöht die Geschwindigkeit von Sprite 3 ...
590 NEXT xGeschw
```

Interessiert Sie, an welcher Stelle sich ein Sprite gerade befindet, dann reicht diese Zeile:

```
600 SPRT 3, -3, X : SPRT 3, -4, Y
610 REM X enthält jetzt die X-Position, Y die Y-Position
```

5.7 Sound

Der C 64 und C 128 besitzt einen leistungsstarken Soundchip (SID). Mit den in diesem Kapitel aufgeführten Befehlen können Sie diesem in Ihren Programmen jeden nur erdenklichen Ton entlocken.

5.7.1 VOICE

Syntax: VOICE 'Ausdruck','Ausdruck'

Mit VOICE bestimmen Sie für jede Stimme, mit welchem Instrument diese spielen soll.

Der erste numerische Ausdruck (zwischen 1 und 3) bezeichnet die Stimme, der zweite das Instrument, welches diese Stimme spielen soll:

0 Schlagzeug
1 Flöte
2 Streicher
3 Bläser
4 Glocke

Beispiel:

```
10 VOICE 1, 2
```

5.7.2 SOUND

Syntax: SOUND 'Ausdruck','Ausdruck','Ausdruck','Ausdruck'

Dieser Befehl erzeugt Töne. Ein vom Soundchip des C 64 und C 128 zu erzeugender Ton wird durch eine Hüllkurve definiert, die aus der Anschlagzeit (Attack), der Abschwellzeit (Decay), der Haltezeit (Sustain) und der Ausklingzeit (Release) besteht. Durch dieses Verfahren lassen sich ziemlich viele verschiedene Klänge nachahmen.

Der erste numerische Ausdruck setzt die Stimme (1 bis 3), mit dem der Ton gespielt werden soll. Das zu benutzende Instrument hängt von der genutzten Stimme ab und kann mit dem VOICE-Befehl gesetzt werden. Der nächste Ausdruck bestimmt die Frequenz (0 bis 65535) des Tons.

Für die Definition der Anschlag- und Abschwellzeit ist der dritte numerische Parameter zu verwenden. Die unteren vier Bit stellen die Anschlag-, die oberen vier Bit die Abschwellzeit dar. Durch den vierten Ausdruck wird die Halte- und die Ausklingzeit gesetzt. Die unteren vier Bit stehen für die Haltezeit, die oberen vier Bit für die Ausklingzeit. Die beiden letzten Ausdrücke müssen im Bereich zwischen 0 und 255 liegen. Anschlag-, Abschwell-, Halte- und Ausklingzeit dürfen Werte von 0 bis 15 annehmen.

Durch Multiplikation der Abschwellzeit mit 16 und anschließender Addierung der Anschlagzeit erhalten Sie den dritten Parameter für SOUND, ebenso verfahren Sie mit der Zusammenfassung der Halte- und Ausklingzeit im vierten Parameter. Beispiel:

```
10 SOUND 1, 10000, 5 * 16 + 10, 8 * 16 + 3
20 REM Attack: 10, Decay: 5, Sustain: 3, Release: 8
```

Möchten Sie den Ton wieder ausschalten, verwenden Sie folgenden Befehl:

```
SOUND 1, 0, 0, 0
```

Weil dadurch der Rahmen dieses Handbuches gesprengt würde, kann die Programmierung von Tönen nicht bis ins letzte Detail erklärt werden. Greifen Sie, wenn Sie trotzdem ein größeres Interesse daran haben und mit eigenen Versuchen noch nicht weit gekommen sind, auf die einschlägige Fachliteratur über die Musikprogrammierung mit dem C 64 zurück.

5.8 Disketten- und Dateizugriffe

GEOS ist, bedingt durch seine Größe, ein diskettenorientiertes Betriebssystem. Es wäre beispielsweise vollkommen unmöglich gewesen, ein Textverarbeitungssystem wie GeoWrite zu entwickeln, das vollständig im Speicher steht, während es unter GEOS abläuft. Spätestens wenn Sie einen mehrseitigen Text hätten eingeben wollen, wäre es zu ernsthaften Speicherplatz-Problemen gekommen.

Selbst GeoBasic kann nicht ohne ständiges Nachladen von Programmteilen korrekt funktionieren; immerhin ist es ca. 79 Kbyte groß, mehr als doppelt so groß wie der verfügbare Speicher im GEOS-Betrieb (31 Kbyte).

Gerade wegen des ständigen Speicherplatzmangels sind in GEOS und GeoBasic leistungsstarke Disketten- und Dateiroutinen integriert. Ein »diskTurbo« verkürzt die Zugriffszeiten auf die Diskette auf ein erträgliches Maß und »last not least« wurde auch noch ein neues Dateiformat geschaffen, durch das die Übersichtlichkeit der Disketten-Inhaltsverzeichnisse verbessert und die Verwaltung von enorm großen Datenmengen stark vereinfacht wird:

Das VLIR-(Variable-Length-Indexed-Record-)Format stellt eine Weiterentwicklung des im normalen C64- und C128-Betrieb verfügbaren relativen Speicherverfahrens dar. Allein durch das VLIR-Format sind so große Programme wie GeoBasic erst möglich geworden. Gäbe es dieses Dateiformat nicht, dann hätte GeoBasic in mehreren einzelnen Dateien gespeichert werden müssen!

Ein gutes Beispiel für die Verwendung des VLIR-Formats sind außerdem noch die Text-Dateien, die Sie mit GeoWrite erstellen. Jeder Text wird von GeoWrite automatisch in Seiten aufgeteilt und auch so gespeichert. Bei der Verwendung des normalen sequentiellen Speicherverfahrens (bei dem alles an einem Stück gespeichert wird) müßte jede Seite in einer einzelnen Datei gesichert werden. Im Inhaltsverzeichnis der Diskette würde dann erscheinen: »Seite 1«, »Seite 2« ...

GeoWrite geht einen anderen Weg: Jede Seite wird als Datensatz in einer großen Datei vom Format VLIR gespeichert. Diese Datensätze sind sehr einfach auffindbar, da ihre Position auf der Diskette automatisch in einem Index-Block gespeichert wird.

Sie können eine VLIR-Datei auch als eine Zusammenfassung mehrerer (sequentiell gespeicherter) Datensätze zu einer Datei bezeichnen.

Jede VLIR-Datei darf maximal 127 solcher Datensätze umfassen (0 bis 126), und alle Datensätze können unterschiedlich lang sein. Möglich sind Datensätze bis zu einer Länge von 32.258 Byte. Die maximale Speicherkapazität einer VLIR-Datei beträgt somit 4.096.766 Byte (ca. 4 Mbyte), eine Größe, die aber wegen des niedrigen Disketten-Speicherplatzes (Floppy 1541 und 1570: 165 Kbyte, Floppy 1571: 330 Kbyte und Floppy 1581: 790 Kbyte) in der Praxis nicht erreicht werden kann.

Im folgenden werden alle Befehle, die die Zugriffe auf sequentielle und VLIR-Dateien betreffen, erläutert.

5.8.1 HEADER

Syntax: HEADER 'Ausdruck','String'[,'Ausdruck']

GEOS speichert zusätzliche Informationen zu einer Datei (beispielsweise das Datei-Piktogramm, die Dateiklasse und eventuell den Autor, wenn es sich um eine Programm-Datei handelt) in einem Info-Block. Den Inhalt dieses Info-Blocks können Sie sich nach dem Markieren einer Datei unter Desktop durch Anklicken von »Info« aus dem »Datei«-Menü anzeigen lassen.

Unter Zuhilfenahme dieser Datei-Informationen können Sie beispielsweise die Dateien einer bestimmten Dateiklasse bei der Auswahl aus der DBFILE-Dialogbox selektieren.

Der HEADER-Befehl ermöglicht Ihnen in Ihren GeoBasic-Programmen die Bestimmung der wichtigsten dateispezifischen Werte und Informationen für die Erstellung eigener Dateien (Abschnitt 5.8.2 CREATE) oder aber im Zusammenhang mit der Dialogbox zur Auswahl von Dateien (Abschnitt 5.4.2 DBFILE).

Mit dem ersten numerischen Ausdruck geben Sie den gewünschten Dateityp an:

1	Basic
2	Assembler
3	nur Daten
4	Systemdatei
5	Hilfsmittel
6	Applikation
7	Dokument
8	Schriftart
9	Druckertreiber
10	Eingabetreiber

11	Disk-Treiber
12	System-Boot
13	temporär
14	selbstausführend
15–128	Eingabetreiber

Sinnvoll sind bei der Erstellung von eigenen Dateien nur die Dateitypen 3, 4 und 7, wobei Sie aber grundsätzlich Dateien aller Typen in den Speicher laden können!

In der Zeichenketten-Konstanten oder Variablen 'String' übergeben Sie die Dateiklasse, die maximal 13 Zeichen lang sein darf.

Der optionale numerische Ausdruck gibt das Dateiformat an, das Sie wünschen:

0	sequentiell
1	VLIR

Wird der HEADER-Befehl nicht genutzt, dann ist der Dateityp 3 und die Dateiklasse »Basic DATA« voreingestellt. Beispiel:

```
10 HEADER 7, "Write Image V", 1
20 DBFILE A$
30 REM Läßt den Benutzer eine GeoWrite Text-Datei auswählen.
```

5.8.2 CREATE

Syntax: CREATE 'String'['Ausdruck']

Erstellt eine neue sequentielle oder eine neue VLIR-Datei auf Diskette.

Die Zeichenketten-Konstante oder Variable 'String' enthält den Namen der neuen Datei (höchstens 16 Zeichen lang). Eine bereits vorhandene Datei mit dem gleichen Namen wird automatisch gelöscht!

Der zweite optionale Parameter übergibt die Nummer des Laufwerks, auf dem die Datei erstellt werden soll (8 bis 11). Wird dieser Ausdruck nicht angegeben, dann wird die Datei auf dem aktuellen Laufwerk eingerichtet.

Hinweis: Auch wenn Ihnen durch diesen Parameter die Möglichkeit gelassen wird, auf ein ganz bestimmtes Laufwerk zuzugreifen, sollten Sie dies nicht machen. Jede typische GEOS-Applikation greift zuerst auf das Laufwerk zu, von dem es gestartet wurde (das aktuelle Laufwerk) und läßt dann dem Programmbenutzer eventuell eine Wahlmöglichkeit. Möchten Sie den Parameter doch verwenden, dann sollte sich das Programm das aktuelle Laufwerk nach

dem Programmstart merken und am Ende des Programms wieder auf dieses Laufwerk zurückschalten, um Probleme mit GeoBasic zu vermeiden.

Die dateispezifischen Informationen (Dateityp ...) müssen vor dem CREATE-Befehl mittels HEADER (Abschnitt 5.8.1) gesetzt werden.

Bei der Erstellung einer VLIR-Datei müssen Sie darauf achten, nach CREATE immer mindestens einen APPEND- (Abschnitt 5.8.9) oder INSERT-Befehl (Abschnitt 5.8.10) anzuwenden, um einen Datensatz zu erstellen. Ansonsten kann es passieren, daß Sie nach dem erneuten Öffnen der Datei (Abschnitt 5.8.3) keine Daten speichern können, da keine Datensätze vorhanden sind.

CREATE öffnet eine Datei automatisch nach deren Erstellung. Die Benutzung des CLOSE-Befehls (Abschnitt 5.8.4) zum Schließen ist deshalb unerläßlich. Beispiel:

```
10 CREATE "Test"
20 REM Erzeugt eine Datei >>Test<< auf der Diskette im aktuellen
Laufwerk
30 CLOSE
```

5.8.3 OPEN

Syntax: OPEN 'String'[,'Ausdruck']

Dieser Befehl öffnet eine bereits vorhandene Datei (sequentiell oder VLIR) für Eingabe- oder Ausgabe-Befehle.

Die Zeichenketten-Konstante oder Variable 'String' enthält den Namen der zu öffnenden Datei.

Wenn Sie den optionalen numerischen Ausdruck angeben, dann bestimmt dieser die Nummer des Laufwerks, auf dem die Datei geöffnet werden soll (Kapitel 8 bis 11). Wird er nicht angegeben, dann versucht GeoBasic, die Datei auf dem aktuellen Laufwerk zu öffnen.

Hinweis: Der zweite Parameter sollte grundsätzlich nicht verwendet werden (Abschnitt 5.8.2).

Es kann immer nur eine einzige Datei geöffnet werden. Wenden Sie den OPEN-Befehl nochmals an, ohne vorher eine bereits geöffnete Datei mit CLOSE (Abschnitt 5.8.4) wieder geschlossen zu haben, dann verlieren Sie die zuletzt in dieser Datei gespeicherten Daten. Beispiel:

```
10 OPEN "Test"
20 REM Versucht, die Datei >>Test<< zu öffnen.
```

5.8.4 CLOSE

Syntax: CLOSE

Dieser Befehl schließt die Datei wieder, die mit OPEN (Abschnitt 5.8.3), CREATE (Abschnitt 5.8.2) oder dem DBFILE-Befehl (Abschnitt 5.4.2) geöffnet wurde.

Da immer nur eine Datei zu einem Zeitpunkt geöffnet sein kann, bedarf es keiner Parameterangabe zu CLOSE. Beispiel:

```
10 OPEN "Test"
20 REM Hier können Sie Datei-Operationen einbauen.
30 CLOSE
40 REM Die Datei >>Test<< wird wieder geschlossen.
```

5.8.5 DREAD

Syntax: DREAD 'Variable'['Variable']...

DREAD liest Daten aus einer geöffneten Datei und übergibt diese den als Parameter zu DREAD angegebenen Variablen.

Der Datentyp (numerisch, Zeichenkette) muß natürlich mit den Variablen-Typen übereinstimmen. Bevor Sie mit dem DREAD-Befehl aus einer VLIR-Datei Daten herauslesen können, müssen Sie den betreffenden Datensatz mit Hilfe des PTREC-Befehls (Abschnitt 5.8.8) bestimmen!

Sind keine Daten mehr in der Datei vorhanden, die noch ausgelesen werden können, dann wird ein »Buffer overflow«-Fehler ausgegeben. Beispiel:

```
10 OPEN "Test"
20 DREAD tstVar, A$
```

Dieses Programm setzt voraus, daß mit dem WRITE-Befehl eine numerische und anschließend eine Zeichenketten-Konstante in der Datei »Test« (sequentielles Format) gespeichert wurde.

5.8.6 RDBYTE

Syntax: RDBYTE 'Variable'['Variable']...

Mit RDBYTE können Sie einzelne Bytes aus einer Datei auslesen, die in den angegebenen Variablen gespeichert werden.

Obwohl auch numerische Variablen zugelassen sind, sollten Sie nur Zeichenketten-Variablen in der Parameterliste verwenden, da bei Einlesen eines Zeichenketten-Werts in eine numerische Variable eine Fehlermeldung erscheint und das Programm abgebrochen wird. Möchten Sie dann anschließend die eingelesenen Werte in eine numerische Variable übertragen, dann verwenden Sie dazu die VAL-Funktion (Abschnitt 4.10.2.2).

Die Datei muß vorher mit OPEN (Abschnitt 5.8.3) oder DBFILE (Abschnitt 5.4.2) geöffnet worden sein.

Bei VLIR-Dateien müssen Sie vor Benutzung des RDBYTE-Befehls den PTREC-Befehl (Abschnitt 5.8.8) anwenden, um den Datensatz zu bestimmen, auf den Sie zugreifen möchten.

Sind keine auslesbaren Daten mehr in der Datei vorhanden, dann wird ein »Buffer overflow«-Fehler ausgegeben. Beispiel:

```
10 OPEN "Test"
20 RDBYTE Wert, St$
```

Sie müssen vor Ausführung dieses Beispiels eine Datei mit Namen »Test« anlegen, in der die beiden vom Programm geforderten Werte (ein numerischer und ein beliebiger Wert) enthalten sind.

5.8.7 WRITE

Syntax: WRITE 'Ausdruck'[,'Ausdruck']...

Der WRITE-Befehl ist dazu bestimmt, Daten in eine geöffnete Datei auf Diskette zu schreiben.

Diese Daten sind in Form von beliebigen numerischen oder Zeichenketten-Ausdrücken an den WRITE-Befehl als Parameter zu übergeben.

Die geschriebenen Daten können anschließend mit DREAD (Abschnitt 5.8.5) und RDBYTE (Abschnitt 5.8.6) wieder gelesen werden.

In einer VLIR-Datei ist vor Benutzung des WRITE-Befehls der gewünschte Datensatz mit dem PTREC-Befehl (Abschnitt 5.8.8) zu bestimmen.

Daten werden in VLIR-Dateien nur gespeichert, wenn der Datensatz, auf den Sie zugreifen möchten, vorher mit APPEND (Abschnitt 5.8.9) oder INSERT (Abschnitt 5.8.10) neu geschaffen wurde. Möchten Sie Daten, die Sie aus einem Datensatz ausgelesen haben, auch wieder in diesen zurückschreiben, müssen Sie diesen Datensatz erst mit DELETE (Abschnitt 5.8.11) löschen und anschließend (nach APPEND oder INSERT) komplett neu beschreiben. Ist der betroffene Datensatz nicht vorhanden, dann wird das Programm mit der Fehlermeldung »Bad record« abgebrochen. Beispiel:

```
10 OPEN "Test"
20  WRITE "Hallo, Du", A, B%
30 CLOSE
```

Eine sequentielle Datei mit Namen »Test« muß vor Ausführung dieses Programms bereits auf der Diskette im aktuellen Laufwerk enthalten sein.

5.8.8 PTREC

Syntax: PTREC 'Ausdruck'

Dieser Befehl ist nur verwendbar, wenn Sie mit einer VLIR-Datei arbeiten. Er bestimmt, auf welchen Datensatz (Record) Sie in einer geöffneten Datei zugreifen möchten.

Im numerischen Ausdruck muß die Nummer des Datensatzes übergeben werden (0 bis 126).

Ist der Datensatz, den Sie angegeben haben, noch nicht mittels APPEND (Abschnitt 5.8.9) oder INSERT (Abschnitt 5.8.10) angelegt worden, dann wird die Fehlermeldung »Bad record« ausgegeben.

Alle auf PTREC folgenden Schreib- und Lesezugriffe auf die Datei beziehen sich nur noch auf den gewünschten Datensatz. Beispiel:

```
10 HEADER 7, "Test V1.0", 1
20 CREATE "TEST"
30 APPEND 0
40 CLOSE
50
60 OPEN "TEST"
70 PTREC 0
80 REM Positioniert den Zeiger auf Record 0.
```

5.8.9 APPEND

Syntax: APPEND 'Ausdruck'

Dieser Befehl hängt einen neuen Datensatz hinter dem im numerischen Ausdruck angegebenen Datensatz in einer geöffneten VLIR-Datei an.

Alle folgenden Datensätze werden um einen Datensatz nach hinten verschoben, Datensatz 122 wird beispielsweise Datensatz 123 und Datensatz 123 wird Datensatz 124. Beispiel:

```
10 HEADER 7, "Test V1.0", 1
20 CREATE "TEST"
30 APPEND 0
40 REM Hängt einen Record hinter Record 0 an.
50 CLOSE
```

5.8.10 INSERT

Syntax: INSERT 'Ausdruck'

Der INSERT-Befehl arbeitet ähnlich wie der APPEND-Befehl. Ein neuer Datensatz wird nur nicht hinter dem angegebenen angehängt, sondern vor dem im numerischen Ausdruck angegebenen Datensatz eingefügt.

Der numerische Ausdruck muß im Bereich von 0 bis 126 liegen. Alle folgenden Datensätze einschließlich des als Parameter angegebenen werden um einen Datensatz nach hinten verschoben.

Wenn das Einfügen eines Datensatzes nicht mehr möglich ist (alle 127 Datensätze sind bereits belegt), dann wird die Meldung »Out of records« ausgegeben. Beispiel:

```
10 HEADER 7, "Test V1.0", 1
20 CREATE "TEST"
30 INSERT 0
40 REM Fügt einen Record vor Record 0 ein.
50 CLOSE
```

5.8.11 DELETE

Syntax: DELETE 'Ausdruck'

Dieser Befehl ist das genaue Gegenstück zu dem INSERT-Befehl; der als Parameter angegebene Datensatz wird gelöscht.

Der numerische Ausdruck muß im Bereich von 0 bis 126 liegen. Alle auf den gelöschten Datensatz folgenden Datensätze werden um einen Datensatz nach vorne verschoben. Beispiel:

```
10 HEADER 7, "Test V1.0", 1
20 CREATE "TEST"
30 INSERT 1
40 REM Fügt einen Record vor Record 1 ein.
50
60 DELETE 1
70 REM Löscht den neu geschaffenen Record 1 wieder.
80 CLOSE
```

5.8.12 LOAD

Syntax: LOAD 'String','Ausdruck'[,'Ausdruck']

Lädt eine sequentiell gespeicherte Datei (z. B. mit dem SAVE-Befehl aus Abschnitt 5.8.13) komplett in den Speicher.

Der Name der Datei (maximal 16 Zeichen lang) wird in der Zeichenketten-Konstanten oder Variablen 'String' übergeben.

Der erste numerische Ausdruck bezeichnet die Adresse im Speicher, an die die Datei geladen werden soll (0 bis 65535). Der zweite numerische Ausdruck gibt das Laufwerk an, von dem die Datei geladen werden soll (Kapitel 8 bis 11). Ist dieser Parameter nicht angegeben, dann wird die Datei vom aktuellen Laufwerk geladen.

Hinweis: Der letzte Parameter sollte normalerweise nicht angegeben werden (Abschnitt 5.8.2). Beispiel:

```
10 LOAD "Pic", 40960
20 REM Lädt >>Pic<< in den Bildschirmspeicher.
```

5.8.13 SAVE

Syntax: SAVE 'String','Ausdruck','Ausdruck'[,'Ausdruck']

Mit dem SAVE-Befehl können Sie einen beliebigen Bereich des Speichers in einer Datei auf Diskette sichern. Diese Datei wird im sequentiellen Format (nicht im VLIR-Format) angelegt. Der Name der Datei, in der der Bereich gesichert werden soll, wird in der Zeichenketten-Konstanten oder Variablen 'String' übergeben (maximal 16 Zeichen).

Der erste numerische Ausdruck gibt die Adresse an, ab der der Speicher gesichert werden soll, der zweite numerische Ausdruck die Anzahl der Bytes, die gesichert werden sollen. Geben Sie den letzten numerischen Ausdruck auch an, dann enthält er die Nummer des Laufwerks, auf dem die Datei gespeichert werden soll. Ansonsten wird die Datei auf der Diskette im aktuellen Laufwerk gespeichert.

Hinweis: Der letzte Parameter sollte grundsätzlich keine Verwendung finden (Abschnitt 5.8.2). Beispiel:

```
10 A$ = "Pic"
20 SAVE A$, 40960, 8000
30 REM Speichert den gesamten Bildschirm auf Diskette.
```

5.9 Ansteuern des Druckers

Durch das offene System von GEOS ist die Verwendung von fast allen Druckertypen möglich. Die Verwaltung des von Ihnen unter dem Desktop ausgewählten Druckertreibers wird automatisch von GEOS übernommen, Sie müssen also in Ihren GeoBasic-Programmen vor Starten des Druckmodus nicht erst überprüfen, welcher Drucker angeschlossen ist und welchen Treiber Sie dafür benötigen.

Sollte bei Verwendung der in diesem Abschnitt verwendeten Befehle allerdings nicht der voreingestellte Druckertreiber auf der aktuellen Diskette sein, dann wird GeoBasic den Druckversuch mit der Meldung »No Print Driver« abbrechen und Sie müssen zuerst den Druckertreiber auf die Diskette kopieren.

5.9.1 PRASCII

Syntax: PRASCII ['Ausdruck'][,/;['Ausdruck']]...

Der PRASCII-Befehl funktioniert genauso wie der PRINT-Befehl, die Ausgabe der Zeichen erfolgt allerdings nicht auf den Bildschirm sondern auf den Drucker.

Die Ausdrücke können beliebig sein (Formeln, Zeichenketten, Variablen, Konstanten ...).

Erfolgt die Trennung durch Semikolon, dann wird kein Zwischenraum zwischen den einzelnen Ausdrücken gelassen, wird hingegen ein Komma als Trennzeichen verwendet, dann bewegt sich der Druckkopf vor der Ausgabe in die nächste der acht gleich großen horizontalen Druckzonen oder, wenn die letzte Zone bereits überschritten wird, in die nächste Zeile.

Steht als letztes Zeichen in einem PRASCII-Befehl ein Semikolon, dann wird der Druckkopf nicht automatisch in die nächste Zeile gefahren, sondern verbleibt hinter dem letzten ausgegebenen Zeichen in der aktuellen Zeile.

Die Zeichen »,;« bewirken, daß der Druckkopf in die nächste Druckzone gebracht wird und dort verbleibt. Zeichenketten-Konstanten und Variablen brauchen nicht durch ein Semikolon oder Komma getrennt zu werden.

Wird kein Ausdruck nach PRASCII angegeben, dann wird eine leere Zeile gedruckt. Beispiel:

```
10 X = 10
20 PRASCII "X = "; X
30 PRASCII "X ^ 2 = "; X ^ 2
40 A$ = "Thomas": B$ = "Karsten": C$ = "Jan"
50 PRASCII A$B$,C$;" ""HI!"
```

5.9.2 NEWPAGE

Syntax: NEWPAGE

Löst beim Drucker einen »Form Feed« aus, so daß entweder das aktuelle Blatt ausgeworfen oder der Druckkopf – bei Endlospapier – an die erste Position des nächsten Blattes bewegt wird. Beispiel:

```
10 PRASCII "Dies ist auf Blatt 1."
20
30 NEWPAGE
40 PRASCII "Dies ist auf Blatt 2."
```

5.9.3 PRNTER

Syntax: PRNTER 'Ausdruck'

Mit diesem Befehl können Sie zwischen Drucker und Bildschirm als Ausgabegerät für den PRINT-Befehl wechseln.

Ist der numerische Ausdruck gleich Null, dann werden bei nachfolgenden PRINT-Befehlen die Zeichen auf dem Bildschirm ausgegeben, ist der Ausdruck ungleich Null, dann erfolgt die Ausgabe auf dem Drucker. Beispiel:

```
10 PRNTER 1
20 REM Ausgabe ab jetzt auf dem Drucker.
30
40 PRINT "Dieser Satz sollte auf dem Drucker ausgegeben werden,"
50 PRINT "nicht auf dem Bildschirm."
```

5.9.4 PRSCREEN

Syntax: PRSCREEN 'Ausdruck'

Gibt eine Hardcopy des Bildschirminhalts auf dem Drucker aus.

Ist der numerische Ausdruck gleich Null, dann erfolgt die Druckerausgabe in normaler Größe, bei einem Wert ungleich Null wird die Hardcopy in doppelter Größe erstellt. Beispiel:

```
10 PRSCREEN 0
20 REM Hardcopy in Normalgröße
```

Hinweis: Beachten Sie, daß bei Druckern mit einer Druckdichte von nur 60 dpi (z. B. MPS 801) ein Teil des rechten Randes verschluckt wird, wenn die Hardcopy in doppelter Größe erfolgt.

5.10 Prozesse

Die GEOS-Mainloop (Abschnitt 5.11) ist in der Lage, Prozesse zu steuern.

Prozesse sind Routinen, die in bestimmten Zeitabständen aufgerufen werden und dabei eine kleine Aufgabe erledigen. Mittels eines Prozesses kann der Desktop in der Version 2.0 beispielsweise ständig die aktuelle Uhrzeit und das Datum anzeigen.

Prozesse sollten kleine Routinen sein, da sie, wenn sie zu lang sind, die GEOS-Mainloop stark verlangsamen und dann eventuell wichtige Aufgaben, die die GEOS-Mainloop noch ausführen muß, nicht mehr zeitgerecht abgearbeitet werden.

Es können maximal acht Prozesse gleichzeitig ablaufen. Wie bereits gesagt, ist für die Prozeßsteuerung die Mainloop zuständig; Sie sollten sich aus diesem Grunde auch noch mit Abschnitt 5.11 beschäftigen.

5.10.1 PROCESS

Syntax: PROCESS 'Ausdruck','Ausdruck'

Mit diesem Befehl können Sie einen Prozeß starten.

Der erste numerische Ausdruck, der natürlich auch Labels enthalten kann, bezeichnet die erste Zeilennummer der Unterroutine, die den Prozeß darstellt. Diese Unterroutine wird von der GEOS-Mainloop mit einem GOSUB-Befehl aufgerufen und muß daher mit einem RETURN-Befehl abschließen.

Der zweite numerische Ausdruck bezeichnet die Zeit in 60stel Sekunden, die ablaufen soll, bevor der Prozeß aufgerufen werden soll. Nach jedem Ablauf wird diese Zeit erneut abgewartet und dann der Prozeß wieder aufgerufen. Beispiel:

```
10 PROCESS @time, 6
20 MAINLOOP
999
1000 @time:
1010     REM Prozeßbeginn
1020     x = XPOS(0)
1030     y = YPOS(0)
1040         REM Die alte Position des Cursors retten.
1050
1060     SETPOS 260, 10
1070     SYSINFO 15, a$
1080     PRINT " "; a$; " "
```

```
1090       REM Zeit ausgeben.
1100
1110    SETPOS x, y
1120      REM Alte Cursorposition wiederherstellen.
1130
1140 RETURN
1150     REM Prozeßende
```

Dieser Prozeß sorgt dafür, daß ständig die aktuelle Zeit auf dem Bildschirm angezeigt wird.

Um das eigentliche Programm nicht durch die ständige Änderung der Cursorposition für die Ausgabe der Zeit zu verwirren, wird vor jeder Ausgabe die Cursorposition in zwei Variablen gerettet und nach der Ausgabe wiederhergestellt.

5.10.2 DELPROC

Syntax: DELPROC 'Ausdruck'

Mit diesem Befehl können Sie einen Prozeß, den Sie mit PROCESS (Abschnitt 5.10.1) gestartet haben, wieder abbrechen.

Der numerische Ausdruck übergibt die erste Zeilennummer des Prozesses. Beispiel:

```
10 PROCESS @time, 6
...
100 DELPROC @time
```

Hinweis: Die Verwendung des DELPROC-Befehls ist auch innerhalb von Prozessen erlaubt, ein Prozeß kann sich durch einen solchen Befehl sogar selbst ausschalten.

5.11 Die GEOS-Mainloop

Verglichen mit dem originalen Betriebssystem des C 64 und C 128 ist mit GEOS ein großer Schritt nach vorne gemacht worden.

Dies zeigt sich schon bei der rein äußerlichen Betrachtung; GEOS ist Benutzeroberflächen größerer und teurerer Computer wie dem Apple Macintosh nachempfunden und unterstützt Menüs, Maus, Fenster, Piktogramme und vieles mehr.

Es ist gar nicht so weit hergeholt, daß jemand, der GEOS einmal in Aktion erlebt hat, meint, daß die Benutzung dieser vielen zusätzlichen Funktionen in eigenen Programmen ungemein schwer sein muß.

Dies ist aber nicht der Fall! Das GEOS-Betriebssystem ist so aufgebaut, daß dem Programm sämtliche im Zusammenhang mit der Benutzerführung stehenden Dinge abgenommen werden. Die Maus bewegt sich automatisch, Mausklicks über Piktogrammen und Menüs werden vom GEOS-Betriebssystem selbständig behandelt. So können Sie sich bei der Erstellung eines Programms nur auf die eigentlichen Funktionen konzentrieren.

GEOS ist also – man kann es fast so ausdrücken – ein eigenes großes Programm und stellt seine Funktionen den Applikationen (wie GeoWrite und GeoPaint) und auch Ihren GeoBasic-Programmen zur Verfügung.

Sie müssen aber etwas vom normalen Programmierstil abweichen: Es ist nicht mehr nötig, daß ein Programm eigene Abfrageroutinen für Maus, Menüs oder andere Dinge durchläuft, da GEOS alle diese Funktionen schon eingebaut hat.

Vielmehr ist es jetzt so, daß ein Programm sich selbst sozusagen in das GEOS-Betriebssystem einfügt, indem es beispielsweise ein Menü auf den Bildschirm bringt (MENU-Befehl aus Abschnitt 5.2) und dabei GEOS die Programmstellen mitteilt, die bei Anklicken einer Funktion aus diesem Menü aufgerufen werden sollen. Ebenso verfahren Sie mit Piktogrammen (ICON aus Abschnitt 5.3), Prozessen (Abschnitt 5.10) und der Abfrage von Mausklicks, die nicht über Menüs oder Piktogrammen stattfinden, (Abschnitt 5.5).

Haben Sie nun durch diesen »Initialisierungs«-Programmteil Ihr Programm »mit GEOS verbunden«, dann übergeben Sie die Kontrolle an das GEOS-Betriebssystem und lassen dieses alle nötigen Dinge selbst ausführen.

GEOS ruft dann vollkommen selbständig die angegebenen Routinen bei erfolgenden Mausklicks über Menüs usw. auf, wobei diese nach ihrer Beendigung in das GEOS-Betriebssystem zurückkehren. Dieses wartet wieder auf den nächsten Mausklick, arbeitet eventuell Prozesse ab oder verarbeitet Tastatureingaben und ruft bei erfolgenden Mausklicks (bei anderen Bereichen als Menüs und Piktogrammen) eine Unterroutine auf – eine fortlaufende Schleife also.

Das Gebilde, das diesen ganzen Ablauf managt, nennt sich GEOS-Mainloop (GEOS-Hauptschleife) und ist für Sie die einzige (und auch eine ziemlich einfache) Methode, die besonderen GEOS-Funktionen in Ihren GeoBasic-Programmen korrekt zu nutzen.

5.11.1 MAINLOOP

Syntax: MAINLOOP

Mit diesem Befehl springen Sie nach dem Initialisierungs-Programmteil in die GEOS-Mainloop, um die Verwaltung von Menüs, Piktogrammen, Prozessen usw. dem Betriebssystem zu überlassen.

Alle Routinen, die Sie vorher z. B. als Sprungziele bei Anwahl einer Funktion aus einem Menü oder eines Piktogramms angegeben haben, werden von der GEOS-Mainloop mit einem GOSUB aufgerufen und müssen demnach mit einem RETURN-Befehl beendet werden, um wieder in die GEOS-Mainloop zurückzukehren. Beispiel:

```
10    MENU "Menü"
20    ICON "Pikto"
30    REM Bindet das Menü und das Piktogramm in die GEOS-Mainloop ein.
40
50    MAINLOOP
60    REM GEOS-Mainloop übernimmt jetzt die Kontrolle über Ihr Programm.
999
1000  REM Die nun folgenden Routinen springt GEOS an, wenn eine Funk-
      tion aus
1010  REM dem Menü ausgewählt wurde.
1020
1100  @info:
1110  REM Reaktionsroutine zur »program-info«-Funktion aus dem »geos«-
1120     REM Menü.
1130
1140     DIALOG "InfoB"
1150  REM Bringt eine Dialogbox mit Informationen auf den Bildschirm.
1160
1170  RETURN
1180  REM Nach Beendigung zur GEOS-Mainloop zurück.
1190
1200  @quit:
1210  REM Reaktionsroutine zur Funktion »Ende« aus dem »Datei« Menü.
1220
1230  END
1240  REM Da dies das Programmende ist, wird die GEOS-Mainloop nicht mehr
1250  REM angesprungen.
1260
1300  REM Die folgende Routine springt GEOS an, wenn ein Piktogramm
      angeklickt
1310  REM wurde.
1320
1400  @ic1:
1410  REM Für Piktogramm 1.
1420
1430     PRINT "Piktogramm 1 angeklickt."
```

```
1440
1450  RETURN
1460  REM Zurück zur GEOS-Mainloop.
```

Das Programm setzt voraus, daß Sie ein Menü mit Namen »Menü« und zwei Funktionen, die »@info« und »@quit« aufrufen, erstellt haben und außerdem noch eine Piktogrammliste mit einem Piktogramm, das die Routine »@ic1« aufruft.

Ein weiteres Beispiel für die Programmierung der Mainloop finden Sie in Abschnitt 5.14, wo das Erstellen einer GEOS-typischen Applikation genau beschrieben wird.

5.12 Direkter Zugriff auf den Speicher und die Maschinenroutinen

Die Programmiersprache GeoBasic ist sehr umfassend und bietet für alle grundsätzlichen Dinge Befehle und Funktionen an. Sollte Ihnen allerdings doch irgendwann eine Funktion oder Information fehlen, die GeoBasic durch keinen Befehl und keine Funktion bereitstellen kann, dann hilft nur noch der direkte Zugriff auf den Speicher und das Aufrufen von Maschinenroutinen mittels der hier vorgestellten Befehle.

5.12.1 POKE

Syntax: POKE 'Ausdruck','Ausdruck'

Mit diesem Befehl können Sie einen Ein-Byte-Wert in einer bestimmten Speicherstelle ablegen. Der erste numerische Ausdruck bestimmt die Speicherstelle, in der Sie einen Wert ablegen (0 bis 65535). Der zweite numerische Ausdruck enthält den Wert, der in dieser Speicherstelle abgelegt wird (0 bis 255). Eine Aufführung von wichtigen Speicherstellen finden Sie in Abschnitt 14.6.

Hinweis: Bei der Verwendung des POKE-Befehls sollten Sie sehr vorsichtig sein; das Schreiben eines falschen Werts in die falsche Speicherstelle kann zum Absturz des Computers führen! Beispiel:

```
10 currentMode  = 46
20 underlne     = 128
30 bold     = 64
40 invers   = 32
50 italic   = 16
60 outlne   = 8
```

```
70  plText  = 0
80
90  POKE currentMode, underlne + bold + outlne
100 REM Schaltet Unterstreichen, Fettschrift und Konturschrift ein.
```

Die Speicherstelle »currentMode« (dezimal 46, hexadezimal $2e) enthält den aktuellen Schriftstil, in dem Zeichen auf dem Bildschirm ausgegeben werden. Durch »POKE current-Mode, plText« kann auf Normalschrift zurückgeschaltet werden.

5.12.2 PEEK

Syntax: PEEK('Ausdruck')

Mit der PEEK-Funktion können Sie den Inhalt (Ein-Byte-Wert) einer bestimmten Speicherstelle auslesen.

Der in Klammern anzugebende numerische Ausdruck bestimmt die Speicherstelle, deren Wert an das Programm übergeben werden soll (0 bis 65535).

Der übergebene Wert liegt zwischen 0 und 255. Beispiel:

```
10 currentMode = 46
20
30 A = PEEK(currentMode)
40 PRINT "Aktueller Schriftstil: "; A
```

Dieses Beispiel gibt den Inhalt der Speicherstelle »currentMode« aus.

5.12.3 DPOKE

Syntax: DPOKE 'Ausdruck','Ausdruck'

Dieser Befehl arbeitet ähnlich wie der POKE-Befehl; er schreibt aber nicht Ein-Byte-Werte (0 bis 255) sondern Zwei-Byte-Werte (0 bis 65535) in die Speicherstellen.

Der erste numerische Ausdruck bestimmt die Speicherstelle, in der der Wert abgelegt werden soll. Der zweite numerische Ausdruck enthält den Wert, dessen Low-Byte in der angegebenen Speicherstelle und dessen High-Byte in der direkt darauf folgenden Speicherstelle abgelegt wird. Beide Ausdrücke müssen im Bereich von 0 bis 65535 liegen.

Hinweis: Bei der Verwendung des DPOKE-Befehls sollten Sie sehr vorsichtig sein; das Schreiben eines falschen Werts in die falsche Speicherstelle kann zum Absturz des Computers führen! Beispiel:

```
10 lftMargin  = 53
20 rghtMargin = 55
30
40 DPOKE lftMargin, 16
50 DPOKE rghtMargin, 303
60 REM Der linke Rand wird auf 16, der rechte Rand auf 303 gesetzt.
70    REM Läßt sich auch mit dem WINDOW-Befehl einstellen.
```

Die Speicherstellen »lftMargin« (dezimal 53/54, hexadezimal $35/$36) enthalten den linken Rand (0 bis 319), die Speicherstellen »rghtMargin« (dezimal 55/56, hexadezimal $37/$38) enthalten den rechten Rand (0 bis 319).

5.12.4 DPEEK

Syntax: DPEEK('Ausdruck')

Mit dieser Funktion können Sie einen Zwei-Byte-Wert aus zwei direkt aufeinanderfolgenden Speicherstellen auslesen.

Der in Klammern anzugebende numerische Ausdruck bestimmt die erste der beiden Speicherstellen, deren Wert übergeben werden soll (0 bis 65535).

Der übergebene Wert liegt im Bereich von 0 bis 65535. Beispiel:

```
10 lftMargin  = 53
20 rghtMargin = 55
30
40 PRINT "Linker Rand:  "; DPEEK(lftMargin)
50
60 B = DPEEK(rghtMargin)
70 PRINT "Rechter Rand: "; B
```

5.12.5 CALL

Syntax: CALL 'Ausdruck'[,['Ausdruck'][,['Ausdruck'][,['Ausdruck'][,['Ausdruck']]]]]

Mit diesem Befehl können Sie eine Maschinenroutine aufrufen.

Der erste numerische Ausdruck wird als einziger unbedingt benötigt. Er bestimmt die Adresse der aufzurufenden Routine im Speicher (0 bis 65535).

Die restlichen numerischen Ausdrücke müssen zwischen 0 und 255 liegen und werden (in ihrer

Reihenfolge) im Akkumulator, im X-Register, im Y-Register und im Status-Register an die Maschinenroutine übergeben.

Mit CALL können Sie GEOS-Routinen aufrufen, die nicht direkt von GeoBasic unterstützt werden. Beispiel:

```
10 GtScanLne = 49603
20
30 X = 199
40
50 CALL GtScanLne, 0, X
```

Aufruf der Routine »GtScanLne« (49603) mit dem Wert 199 im X-Register.

5.12.6 USR

Syntax: USR('Ausdruck')

Dieser Befehl bewirkt die Ausführung einer Maschinenroutine, deren Adresse zuvor mit dem DPOKE-Befehl (Abschnitt 5.12.3) in den Speicherstellen 785/786 (Low-Byte/High-Byte) abgelegt wurde.

Der Wert des anzugebenden numerischen Ausdrucks wird in Fließkomma-Akkumulator ab Speicherstelle 97 gespeichert, so daß die aufgerufene Maschinenroutine darauf zugreifen kann.

Das Ergebnis der mit USR aufgerufenen Routine sollte ebenfalls im Fließkomma-Akkumulator abgelegt werden. Beispiel:

```
10 REM Die Adresse der Routine ist 32000
20 DPOKE 785, 32000
30
40 USR(1)
```

5.13 Zeichensätze

Eine Stärke von GEOS ist die Möglichkeit der Verwendung von verschiedenen Zeichensätzen für die Bildschirmdarstellung. Deshalb werden die GeoWrite-Texte am Bildschirm bereits so angezeigt, wie sie später im Ausdruck aussehen (What You See Is What You Get (WYSIWYG)).

Auch GeoBasic unterstützt diese Funktion für Ihre Programme. Dadurch läßt sich die Bildschirmdarstellung durch gekonnte Verwendung verschiedener Zeichensätze und Schriftgrößen sehr auflockern, was sich stark auf die Bedienerfreundlichkeit auswirkt.

Es ist aber empfehlenswert, die Möglichkeit der Zeichensatzumschaltung nicht zu sorglos zu verwenden, da Zeichensätze meist sehr viel Speicherplatz schlucken (normalerweise zwischen einem und vier Kbyte pro Zeichensatz), der Ihnen letztendlich bei Ihren Variablen fehlen wird.

5.13.1 FONT

Syntax: FONT 'String','Ausdruck'

Dieser Befehl lädt einen Zeichensatz aus einer VLIR-Zeichensatz-Datei in den Speicher und lenkt alle Bildschirmausgaben auf ihn um.

Ist der Zeichensatz bereits im Speicher, dann wird nur die Bildschirmausgabe auf ihn umgestellt. Wird der gewünschte Zeichensatz auf der aktuellen Diskette nicht gefunden und steht er auch nicht im Speicher, dann erfolgen alle weiteren Bildschirmausgaben mit dem Systemzeichensatz »BSW« in 9 Punkt Größe.

Die Zeichenketten-Konstante oder Variable 'String' gibt dabei den Namen des Zeichensatzes an (vergessen Sie bei deutschen Zeichensätzen die Kennung »GE« am Ende des Namens nicht).

Der numerische Ausdruck bestimmt die Größe des Zeichensatzes in Punkten, der geladen werden soll. Im deutschen GEOS-Grundsystem sind folgende Zeichensätze vorhanden:

Zeichensatz:	*Größe in Punkt:*
BSW	9 (im GEOS-KERNEL integriert, nicht auf Diskette)
University GE	6, 10, 12, 14, 18, 24
California GE	10, 12, 14, 18
Roma GE	9, 12, 18, 24
Dwinelle GE	18
Cory GE	12, 24

Zusätzlich zu diesen Zeichensätzen finden Sie im deutschen GEOS-2.0-Grundsystem noch weitere:

Commodore GE	10
LW Barrows GE	9, 10, 12, 14, 18, 24
LW Cal GE	9, 10, 12, 14, 18, 24
LW Greek GE	9, 10, 12, 14, 18, 24
LW Roma GE	9, 10, 12, 14, 18, 24

Ungefähr 190 weitere Schriften erhalten Sie mit dem »Mega Pack 1« (Bestell-Nr. 90772, Markt & Technik Verlag AG) und 20 Schriften und einen komfortablen Zeichensatz-Editor (GeoFont) gibt es als »International FONTPACK« (Bestell-Nr. 50321, Markt & Technik Verlag AG). Weitere 33 Schriftarten sind im »Mega Pack 2« erschienen (Bestell-Nr. 90350, Markt & Technik Verlag AG).

Hinweis: Mega Fonts, die im Programmpaket »GeoPUBLISH« enthalten sind, (z. B. Mega Roma), lassen sich nicht in GeoBasic-Programmen verwenden! Beispiel:

```
10 FONT "University GE", 18
20 PRINT "Der Zeichensatz University."
```

5.14 Ihre erste selbstprogrammierte Applikation

Um Ihnen den Einstieg in die Programmierung von GEOS-typischen Applikationen, die Sie mit GeoBasic erstellen können, zu erklären, wird in diesem Abschnitt eine kleine Beispielapplikation in ihre Einzelteile zerlegt und ausführlich beschrieben.

5.14.1 Der Aufbau von GEOS-Applikationen

Wie Sie aus den vorangegangenen Abschnitten wissen, ist bei GEOS-Applikationen aufgrund der Mainloop eine etwas andere Programmstruktur nötig. GEOS-Applikationen gliedern sich in den sogenannten Initialisierungsteil und die Reaktionsroutinen.

Der Initialisierungsteil ist dafür zuständig, Menüs, Piktogramme und sonstige GEOS-Funktionen zu installieren und eventuell den Benutzer des Programms mittels Dialogboxen durch eine Dateiauswahl zu leiten. Am Ende des Initialisierungsteiles wird dann die Kontrolle über das Programm an die Mainloop gegeben. Die Mainloop prüft nun unter anderem, ob ein Menüpunkt oder ein Piktogramm z.B. angeklickt wurde und ruft die Reaktionsroutinen in der Applikation auf.

Die Reaktionsroutinen reagieren dann entsprechend des ausgewählten Menüpunkts (z. B. »schließen«, »aktualisieren«, »herausschneiden«) oder Piktogramms (z. B. Rechteck zeichnen). Am Ende der Reaktionsroutine wird normalerweise wieder die Mainloop angesprungen, um weitere Aktionen zu ermöglichen. Nur bestimmte Reaktionsroutinen (z. B. der Menüpunkt »Ende« im »Datei«-Menü) ermöglichen ein Verlassen der Applikation.

5.14.2 Der Initialisierungsteil

Wie bereits oben gesagt, ist der Initialisierungsteil einer Applikation dazu da, um beispielsweise Menüs und Piktogramme einzubinden.

Kopieren Sie jetzt das Foto-Scrap von der GeoBasic-Sicherheitskopie auf Ihre GeoBasic-Arbeitsdiskette und starten Sie GeoBasic von dieser Arbeitsdiskette.

Erstellen Sie eine neue Datei mit dem Namen »Applikation.bsp«. Unter diesem Namen können Sie die Datei auch auf der GeoBasic-Sicherheitskopie finden. Sobald der Text-Editor erscheint, geben Sie folgende Programmzeilen ein:

```
10 REM ********************************
       *  GEOS  Beispiel-Applikation  *
       * von Holger Latzel 1989, 1990 *
       ********************************
11
20 REM ********************************
       *     Initialisierungsteil     *
       ********************************
21
30 MENU "Menue"
40 REDRAW @afterDA
41
50 ICON "Pikt"
51
60 MAINLOOP
9999
```

Hiermit ist der Initialisierungsprogrammteil fertiggestellt. Zuerst wird in Zeile 30 ein Menü, welches noch im Menü-Editor erstellt werden muß, auf den Schirm gebracht.

Der REDRAW-Befehl weist GeoBasic an, die Routine mit dem Label @afterDA aufzurufen, sobald ein Hilfsmittel, das aus dem »geos«-Untermenü aufgerufen werden kann, beendet ist. Diese Routine (die wir gleich noch ergänzen) soll den Bildschirm wieder so herstellen, wie er vor dem Aufrufen des Hilfsmittels ausgesehen hat.

Zeile 50 bewirkt, daß GeoBasic die Piktogramme aus der Liste »Pikt« auf den Bildschirm schreibt. Auch diese Liste muß noch im Grafik-/Piktogrammlisten-Editor erstellt werden.

Anschließend wird mit Zeile 60 der Initialisierungsprogrammteil beendet und die Mainloop aufgerufen. Dann ist die Menüleiste aktiv und Piktogramme können angeklickt werden.

Starten Sie nun den Menü-Editor (Kapitel 7), indem Sie »menu« aus dem »Utilities«-Menü wählen. Erstellen Sie folgendes Menü:

Name: Menue
Anzahl der Menüpunkte: 2

Menüpunkt 1:	geos (bereits vorgegeben, nicht änderbar)	
	Name des Untermenüpunkts:	aufzurufende Routine:
	Applikation Info	@info
Menüpunkt 2:	Datei	
	Ende	@quit

Verlassen Sie den Menü-Editor. Jetzt muß nur noch die Piktogrammliste erstellt werden. Erst einmal wird das Aussehen des Piktogramms festgelegt. Wählen Sie »bitmap« aus dem »Utilities«-Menü, um den Grafik-Editor (Kapitel 10) zu starten. Die Grafik soll »Quit« heißen und wird ein Piktogramm mit der Aufschrift »Quit« (Verlassen) sein.

Das Erstellen der Grafik von Hand wäre in diesem Falle zu aufwendig. Deshalb kleben wir ein Foto-Scrap ein, das schon die richtige Grafik enthält. Wählen Sie aus dem »options«-Menü »paste photo scrap« aus. Das Foto-Scrap wird jetzt eingeklebt und die Grafik erscheint im Editierfenster. Änderungen an dieser Grafik müssen nicht mehr vorgenommen werden. Verlassen Sie also den Grafik-Editor.

Jetzt muß noch eine Liste mit den Piktogrammen erstellt werden, die das Programm haben soll und mit dem ICON-Befehl eingebunden werden (siehe oben). Rufen Sie den Piktogrammlisten-Editor (Kapitel 9) auf.

Wir möchten nur ein Piktogramm in dieser Applikation und erstellen darum folgende Liste:

Name:	Pikt
Anzahl der Piktogramme:	1
Piktogramm 1:	
X-Position:	264
Y-Position:	176
Name der Grafik:	Quit (vorher im Grafik-Editor erstellt)
Aufzurufende Routine:	@quit

Wenn das Piktogramm »Quit« angeklickt wird, dann wird also die Routine @quit aufgerufen. Dies geschieht auch, wenn Sie »Ende« aus dem »Datei«-Menü auswählen.

Die Piktogrammliste ist jetzt komplett und der Piktogrammlisten-Editor kann nun wieder verlassen werden.

5.14.3 Die Reaktionsroutinen

Nun können die Reaktionsroutinen ergänzt werden. Wie Sie aus dem Initialisierungsprogrammteil ersehen können, benötigen wir die Routinen @afterDA, @info und @quit:

```
10000 REM ****************************
          *     Reaktionsroutinen    *
          ****************************
10010 REM ****************************
          * @afterDA stellt den      *
          * Bildschirm nach Beendi-  *
          * gung eines Hilfsmittels  *
          * wieder her.              *
          ****************************
10011
10020 @afterDA:
10021
10030   PATTERN 2
10040   RECT 0, 16, 319, 199
10050
10060   GOTO 50
19999
20000 REM ****************************
          * @info gibt eine Dialogbox *
          * zum Menuepunkt           *
          * >>Applikation Info<< aus. *
          ****************************
20001
20010 @info:
20011
20020   DIALOG "InfoB"
20030
20040   RETURN
20050
20100 REM ****************************
          * @quit verlaesst die      *
          * Applikation ueber Menuep. *
          * >>Ende<< oder ueber das  *
          * Piktogramm.              *
          ****************************
20101
20110 @quit:
20111
20120   SETPOS 16, 80
20130   PRINT "See you!"
20140
20150   END
```

Die Routine @afterDA arbeitet folgendermaßen: In Zeile 10030 wird für den RECT-Befehl das Füllmuster 2 eingestellt. Es handelt sich hierbei um das Hintergrund-Füllmuster der Applikation. Damit wird dann ein ausgefülltes Rechteck gezeichnet (Zeile 10040).

Die oberen 15 Zeilen des Bildschirms müssen nicht gelöscht werden, da kein Hilfsmittel diese Zeilen verändern darf. Folglich muß auch die Menüleiste nicht nochmal gezeichnet werden.

Anschließend wird in Zeile 50 gesprungen. Dort befindet sich der letzte Teil der Initialisierung. Es wird das Piktogramm angezeigt und anschließend wieder in die Mainloop gesprungen.

@info gibt in Zeile 20020 einfach nur eine Dialogbox auf dem Bildschirm aus. Anschließend kehrt die Routine zurück in die Mainloop (Zeile 20040).

Definieren wir jetzt die Dialogbox im Dialogbox-Editor (Kapitel 8). Wählen Sie dazu »dialog« aus dem »Utilities«-Menü.

Die Dialogbox sieht wie folgt aus:

Name:	InfoB
Anzahl der Objekte:	5
Objekt #1:	Fixed Text
X-Position:	16
Y-Position:	16
Text:	Beispiel Applikation von
Objekt #2:	Fixed Text
X-Position:	16
Y-Position:	32
Text:	Holger Latzel
Objekt #3:	Fixed Text
X-Position:	16
Y-Position:	48
Text:	Version 1.0 für den C64
Objekt #4:	Fixed Text
X-Position:	16
Y-Position:	64
Text:	Copyright (c) 1989, 1990

Objekt #5:	Fixed Text
X-Position:	16
Y-Position:	80
Text:	Markt & Technik Verlag AG

Verlassen Sie jetzt den Dialogbox-Editor. Damit ist die Beispielapplikation komplett. Die Struktur ist offen gehalten, so daß Sie problemlos weitere Routinen ergänzen können. Sehen Sie dazu in den nächsten Abschnitt.

5.14.4 Ergänzen der Beispielapplikation

Die Beispielapplikation, die wir in den letzten Abschnitten aufgebaut haben, ist noch nicht sehr umfangreich. Sie möchten sicher noch nützliche Funktionen einbauen. Dieser Abschnitt soll Ihnen zeigen, wie das möglichst einfach realisierbar ist. Auch beim Ergänzen sollten Sie die Reihenfolge einhalten: erst den Initialisierungsteil, dann die Reaktionsroutinen.

Starten Sie jetzt – falls noch nicht geschehen – GeoBasic und öffnen Sie die Applikation »Applikation.bsp«. Wir ergänzen jetzt eine Routine, die die aktuelle Zeit ständig rechts oben auf dem Bildschirm anzeigt. Dazu eignen sich die Prozeßroutinen: Nehmen wir uns nun den Initialisierungsteil vor. Geben Sie folgende Zeilen ein:

```
60 PROCESS @time, 6 61
70 MAINLOOP
```

Mit der ersten Eingabe haben wir die alte Zeile 60 gelöscht und müssen diese aus diesem Grund in Zeile 70 anhängen.

Zeile 60 dient dazu, einen Prozeß zu initialisieren, der mit der Zeile @time beginnt und jede Zehntelsekunde aufgerufen werden soll. Ergänzen wir nun die Reaktionsroutinen:

```
29999
30000 REM ****************************
          * @time gibt ständig die    *
          * aktuelle Zeit aus.        *
          ****************************
30001
30010 @time:
30011
30020   x = XPOS(0)
30030   y = YPOS(0)
30040
30050  SETPOS 260, 10
30060  SYSINFO 15, a$
30070  PRINT " "; a$; " "
```

```
30080
30090    SETPOS x, y
30100
30110    RETURN
```

In den Zeilen 30020 und 30030 wird die aktuelle Cursorposition in die Variablen x und y gerettet, um nicht die normalen Reaktionsroutinen dadurch zu stören, daß die Cursorposition ständig geändert wird.

Anschließend wird der Cursor nach rechts oben auf den Bildschirm gesetzt. Durch SYSINFO 15, a$ wird die aktuelle Systemzeit in die Zeichenketten-Variable a$ übertragen und in Zeile 30070 auf den Bildschirm ausgegeben. Nun muß nur noch die alte Cursorposition wiederhergestellt werden (Zeile 30090) und der Prozeß kann beendet werden (Zeile 30110).

Wenn Sie die Applikation jetzt starten, wird die aktuelle Zeit ständig in der rechten oberen Bildschirmecke angezeigt – als würde ein zweites Programm gleichzeitig mit dem Hauptprogramm ablaufen.

Ergänzen Sie doch jetzt einfach weitere Funktionen. Wie wäre es denn, wenn Sie aus dem Programm eine Adreßverwaltung machen?! Suchen Sie noch weitere Anregungen, dann sehen Sie sich einfach die anderen Beispielprogramme auf Seite A der GeoBasic-Sicherheitskopie an.

6. Kapitel
Eigenständig lauffähige GeoBasic-Programme

Normalerweise laufen alle mit GeoBasic erstellten Programme auch nur mit GeoBasic ab, d. h., daß beim Starten eines Programms GeoBasic entweder schon im Speicher stehen muß oder noch nachträglich von Diskette geladen wird, um den Ablauf des Programms zu ermöglichen.

Möchten Sie aber beispielsweise ein GeoBasic-Programm einem Ihrer Freunde auf seiner GEOS-Version vorführen und dieser Freund hat kein GeoBasic, dann könnten Sie ihm das Programm normalerweise nicht zeigen.

Darum stellt Ihnen GeoBasic die Möglichkeit bereit, Programme eigenständig ablaufen zu lassen – ohne die Präsenz von GeoBasic im Speicher. Zu diesem Zwecke wird eine »Run-Time-Bibliothek« mit den Befehlen und Funktionen von GeoBasic an das Programm angehängt. Das Verfahren verbraucht sehr viel Diskettenspeicherplatz, Sie sollten deshalb nicht jedes x-beliebige Programm eigenständig lauffähig machen!

6.1 Wie machen Sie ein Programm eigenständig lauffähig?

Um eine »Run-Time-Bibliothek« an das Programm anzuhängen und es damit eigenständig lauffähig zu machen, wählen Sie einfach die Funktion »make appl« aus dem Menü »Options« im Text-Editor aus.

Sie müssen jetzt in einer Dialogbox den Namen des Autors des Programms eingeben (Bild 6.1). Dieser wird dann später im »Datei«/»Info«-Fenster des Desktop angezeigt.

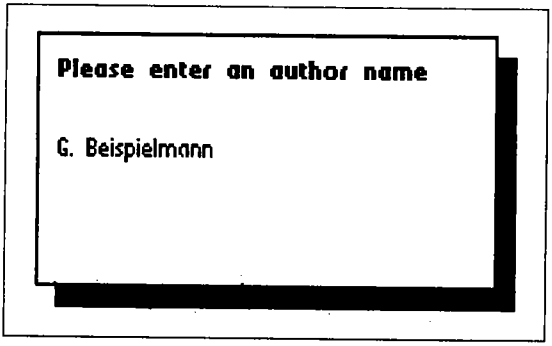


Bild 6.1: Eingabe des Autors

Drücken Sie nach der Eingabe die [Return]-Taste. Eine Dialogbox erscheint auf dem Bildschirm, die Sie zur Eingabe eines neuen Namens auffordert (Bild 6.2).

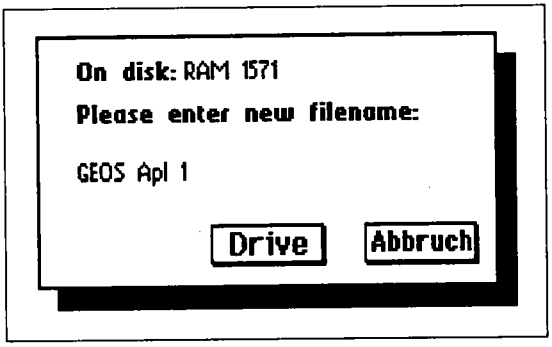


Bild 6.2: Geben Sie hier einen neuen Namen für das eigenständig lauffähige Programm ein

Geben Sie einen Namen ein und drücken Sie [Return], oder klicken Sie »Abbruch« an, um zum Text-Editor zurückzugelangen.

Möchten Sie, daß das eigenständig lauffähige Programm auf einer Diskette in einem anderen Laufwerk stehen soll, dann klicken Sie »Drive« an, um zwischen den Laufwerken hin- und herzuschalten. Klicken Sie »Disk« an, um eine neue Diskette in das aktuelle Laufwerk einlegen zu können, wenn darauf das eigenständig lauffähige Programm erstellt werden soll. Unter dem eingegebenen Namen wird Ihr Programm später in der eigenständig lauffähigen Form gespeichert.

Jetzt wird Ihr Programm in eigenständig lauffähiger Form auf Diskette geschrieben. Nachdem Sie GeoBasic verlassen haben, werden Sie eine neue Datei (Ihr Programm in eigenständig lauffähiger Form) im Disketten-Inhaltsverzeichnis antreffen, deren Piktogramm dem eines GeoBasic-Programms ähnelt (Bild 6.3).

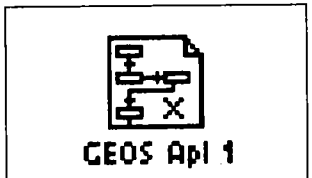


Bild 6.3: Das Piktogramm eines eigenständig lauffähigen Programms

Mit dem Icon-Editor, erhältlich im Programmpaket »Deskpack und GeoDex« (Best.-Nr. 50322, Markt & Technik Verlag AG), können Sie das Piktogramm nachträglich noch ändern.

Das eigenständig lauffähige Programm läßt sich vom Desktop aus – wie alle anderen Applikationen auch – durch doppeltes Anklicken auf das Piktogramm starten. Der Ablauf des Programms erfolgt nun vollkommen unabhängig von GeoBasic.

Hinweis: Jeder END-Befehl oder aber das Erreichen des Programmendes führt bei eigenständig lauffähigen Programmen zu einem Rücksprung zum Desktop. Sie sollten dem Benutzer immer mindestens eine Möglichkeit geben, aus dem laufenden Programm zurück zum Desktop zu gelangen!

6.2 Besonderheiten von eigenständig lauffähigen GeoBasic-Programmen

Wenn Sie ein Programm, das eigene Dokument-Dateien erstellt, eigenständig lauffähig machen, dann können Sie dieses Programm anschließend auch durch doppeltes Anklicken auf das Piktogramm eines Dokuments oder durch Ablage eines Dokument-Piktogramms auf dem Druckersymbol des Desktop starten – genauso, wie Sie z. B. GeoBasic starten können, indem Sie das Piktogramm eines GeoBasic-Programms doppelt anklicken oder auf dem Druckersymbol ablegen!

Für diese Fälle muß Ihr Programm also auch gewappnet sein. Hierbei kann Ihnen wieder der SYSINFO-Befehl (Abschnitt 4.12) behilflich sein:

```
980  REM Bis hier steht der Initialisierungs-Programmteil.
990
1000 @whyStarted:
1010
1020 REM Diese Routine überprüft, warum das Programm gestartet wurde
     und
1030 REM ruft die nötigen Funktionen auf.
1040
1050 SYSINFO 13, A 1060 ON A+1 GOTO @Normal, @DoppelKlick, @Drucken
1070
1080 REM Wenn A = 0, also ein normaler Aufruf stattgefunden hat,
     dann
1090 REM wird weiterhin der normale Initialisierungs-Programmteil
1100 REM abgearbeitet.
1110 REM Ist A = 1 (Doppelklick auf Piktogramm eines Dokuments),
     dann
1120 REM wird diese Datei jetzt geöffnet.
```

```
1130 REM Versucht man, ein Dokument auszudrucken (A = 2), dann wird
1140 REM entsprechend verzweigt.
1150
1160 @Normal:
1170
1180 REM Hier Options- und Dateiauswahl.
1190
1200 MAINLOOP
1210
1220 @DoppelKlick:
1230
1240 SYSINFO 14, A$
1250 REM Dateinamen holen.
1260
1270 REM Hier Datei öffnen
1280
1290 MAINLOOP
1300 1310 @Drucken:
1320
1330 SYSINFO 14, A$
1340 REM Dateinamen holen.
1350
1360 REM Hier Datei drucken.
1370
1380 END
1390 REM Nach dem Ausdrucken zu Desktop zurückkehren.
```

Dies ist nur ein Vorschlag, wie Sie verfahren können, um die unterschiedlichen Programmstart-Möglichkeiten korrekt zu verarbeiten. Sie müssen sich also nicht unbedingt an dieses Schema halten.

Wichtig ist nur, daß Ihr Programm (wenn es die Erstellung von Dokumenten auf Diskette zuläßt), nachdem es eigenständig lauffähig gemacht wurde, auf verschiedene Arten gestartet werden kann, und daß es auch ganz professionell erscheinen sollte, indem es vernünftig reagiert!

7. Kapitel
Der Menü-Editor

Es ist ein Hilfsprogramm in GeoBasic integriert, mit dem Sie komplette Menüs für eigene Basic-Programme entwerfen können. Beachten Sie bitte in diesem Zusammenhang die Abschnitte 5.2 (über den Einbau von Menüs in die Basic-Programme) und 5.11 (über das Steuern der Menüs durch die GEOS-Mainloop).

7.1 Was ist ein Menü?

Bevor Sie Menüs erstellen, sollten Sie erst einmal erfahren, was überhaupt Menüs sind. Als Menü bezeichnet man die Schlagworte an der Oberseite des Bildschirms (Bild 7.1).

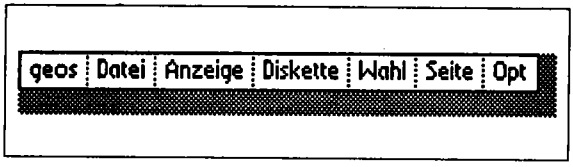


Bild 7.1: Eine »Menüleiste«

Jedes der Schlagworte stellt einen Menüpunkt dar, dem ein Abrollmenü (Bild 7.2) mit Funktionen (wie z. B. Speichern von Dokumenten, Einstellen einer neuen Schriftart usw.) zugeordnet ist.

Bild 7.2: Ein »Abrollmenü«

Ein Abrollmenü öffnen Sie auf dem Bildschirm, indem Sie den zugehörigen Menüpunkt in der Menüleiste anklicken. Auch die einzelnen Funktionen im Abrollmenü rufen Sie durch Anklicken auf. Möchten Sie hingegen keine Funktion aufrufen, dann bewegen Sie den Mauszeiger ohne irgendetwas anzuklicken vom Abrollmenü herunter. Es schließt sich nun automatisch.

7.2 Starten des Menü-Editors

Den Menü-Editor starten Sie durch Auswahl von »menu« aus dem »Utilities«-Menü im GeoBasic-Text-Editor (Bild 7.3).

Bild 7.3: Das »Utilities«-Menü

7.3 Auswahl eines Menüs zur Bearbeitung

Nach dem Starten des Menü-Editors erscheint eine Dialogbox auf dem Bildschirm (Bild 7.4), die der Dialogbox zur Auswahl eines vorhandenen GeoBasic-Dokuments (Abschnitt 3.2.2) sehr ähnelt.

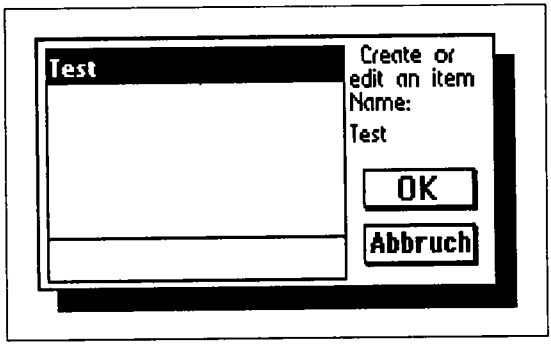


Bild 7.4: Auswahl eines Menüs

Mit »Abbruch« gelangen Sie von hier wieder zurück in den Text-Editor. Möchten Sie ein neues Menü erstellen, dann geben Sie einfach einen Namen für dieses Menü ein und drücken Sie Return. Ist bereits ein Name im Eingabefeld vorgegeben, dann müssen Sie diesen natürlich erst mit der Del-Taste löschen, bevor Sie den neuen Namen eingeben. Ein Menü-Name darf bis zu fünf Zeichen lang sein.

Hinweis: Ist bereits ein Menü mit dem eingegebenen Namen vorhanden, dann wird dieses zur Weiterbearbeitung geöffnet und kein neues Menü erstellt.

Wenn Sie ein schon erstelltes Menü weiterbearbeiten möchten, dann klicken Sie den zugehörigen Namen in der Dialogbox (er wird dann invertiert) und anschließend »OK« an.

Haben Sie den Namen eines existierenden Menüs ausgewählt, dann können Sie in einer weiteren Dialogbox bestimmen, was Sie mit diesem Menü zu tun gedenken (Bild 7.5).

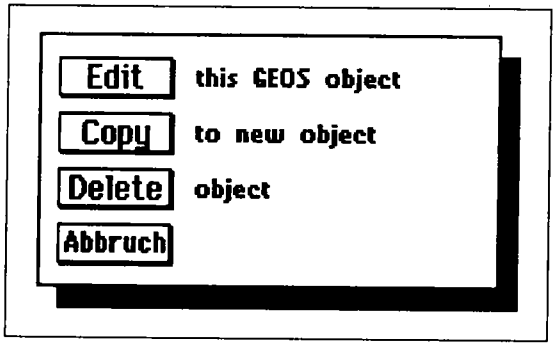


Bild 7.5: Auswahl der Funktion des Menü-Editors

Durch Anklicken von »Edit« gelangen Sie in den eigentlichen Menü-Editor (Abschnitt 7.4).

Mit »Copy« können Sie eine identische Kopie Ihres Menüs anfertigen. Eine neue Dialogbox erscheint (Bild 7.6), die von Ihnen die Eingabe eines neuen Namen fordert. Geben Sie nun den Namen ein und drücken Sie Return. Haben Sie Ihre Meinung geändert und möchten nun doch nicht kopieren, dann klicken Sie »Abbruch« an.

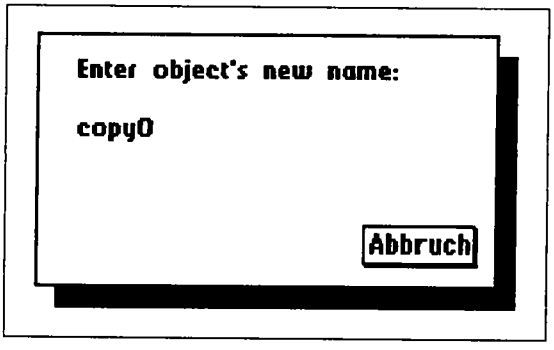


Bild 7.6: Eingabe eines Namens für die Kopie

»Delete« ermöglicht Ihnen das Löschen des aktuellen Menüs, beispielsweise wenn es nicht mehr benötigt wird. Mit dieser Funktion sollten Sie sehr vorsichtig verfahren, da sie nicht mehr rückgängig gemacht werden kann. Wollen Sie den Menü-Editor verlassen, dann klicken Sie auf »Abbruch«.

7.4 Bearbeiten eines Menüs

Haben Sie den Menü-Namen und die Edit-Funktion ausgewählt, dann erscheint der Menü-Editor-Bildschirm (Bild 7.7).

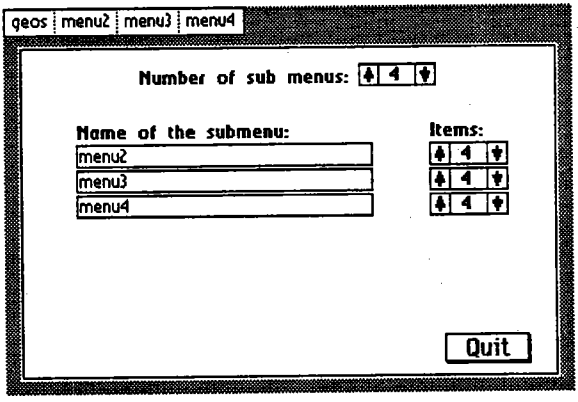


Bild 7.7: Der Menü-Editor-Bildschirm

Erstellen Sie ein neues Menü, dann erscheint im oberen Bildschirmbereich ein Menü mit fünf Schlagwörtern: »geos«, »menu1« bis »menu4«.

Ansonsten erscheint Ihr eigenes Menü in der zuletzt gespeicherten Fassung. Werfen wir jetzt einen Blick auf die verschiedenen Bildschirmelemente: Ihr Menü wird automatisch immer in seiner aktuellsten Fassung angezeigt; geben Sie beispielsweise ein neues Schlagwort ein oder verändern Sie die Anzahl der Menüpunkte, dann wird dies sofort auch auf dem Bildschirm angezeigt.

Der erste Menüpunkt »geos« kann nicht umbenannt oder sonstwie beeinflußt werden! Alle anderen Menüpunkte können Sie nach Belieben verändern; neue Menüpunkte hinzufügen, alte Menüpunkte löschen und Schlagwörter verändern.

Direkt unter dem Menü befindet sich die Zeile mit der Beschriftung »Number of submenus«, was für die Anzahl der Menüpunkte steht. Direkt daneben steht eine Zahl auf dem Bildschirm und links davon ein Pfeil nach oben sowie rechts davon ein Pfeil nach unten.

Mit dem Aufwärts-Pfeil vergrößern Sie die Anzahl der Menüpunkte – der Abwärts-Pfeil verkleinert sie. Die maximale Anzahl von Menüpunkten ist acht. Klicken Sie auf den Aufwärts-Pfeil, wenn bereits acht Menüpunkte vorhanden sind, dann schlägt die Zahl wieder auf eins um, wobei nur noch das »geos«-Menü erscheint.

Unterhalb der Zeile, in der Sie die Anzahl der Menüpunkte einstellen können, finden Sie Felder zur Eingabe der Schlagwörter für die einzelnen Menüpunkte. Jede Zeile steht für einen Menüpunkt. Klicken Sie das Feld an, dessen Schlagwort Sie eingeben oder ändern möchten. Der Cursor erscheint in diesem Feld und Sie können Ihre Eingabe tätigen. Um ein Zeichen zu löschen, benutzen Sie wie gewohnt die Del-Taste. Sind Sie mit der Änderung fertig, drücken Sie auf Return oder klicken eines der anderen Felder an.

Hinweis: Sie sollten darauf achten, nicht zu lange Schlagworte für die Menüpunkte zu verwenden. Andernfalls werden eventuell einige Zeichen nicht auf dem Bildschirm ausgegeben, damit andere Menüpunkte auch noch Platz haben.

Rechts neben jedem Schlagwort-Feld befindet sich ein Feld, das die Anzahl der Funktionen des Abrollmenüs anzeigt. In einem neu erstellten Menü hat anfänglich jedes Abrollmenü vier Funktionen (»action1« bis »action4«). Die Abrollmenüs müssen aber nicht alle die gleiche Anzahl an Funktionen aufweisen!

Links und rechts neben dem Feld gibt es ebenfalls wieder einen Aufwärts- und eine Abwärts-Pfeil zum Einstellen der Anzahl der Funktionen. Maximal können Sie 12 Funktionen jedem Abrollmenü zuordnen; mehr passen leider nicht auf den Bildschirm.

Um den Text einer Funktion in einem Abrollmenü einzugeben, klicken Sie das zugehörige Schlagwort im Menü an und anschließend die Funktion, deren Namen Sie eingeben möchten. Eine Dialogbox erscheint, die Sie auffordert, den neuen Namen einzugeben (Bild 7.8). Löschen können Sie Zeichen mit Del. Sind Sie fertig mit der Eingabe, so drücken Sie die Return-Taste.

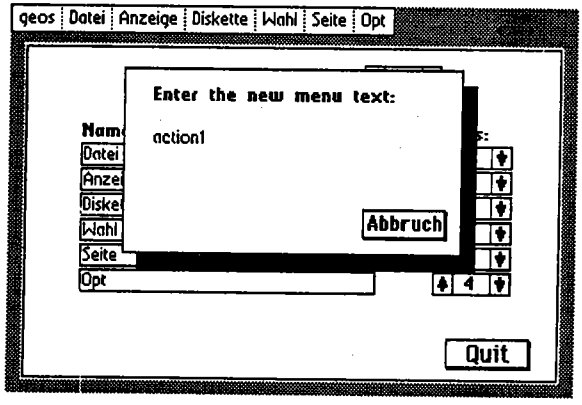


Bild 7.8: Eingabe eines neuen Namens für eine Funktion im Abrollmenü

Mit »Abbruch« können Sie die Änderung abbrechen. Haben Sie Return gedrückt, dann fragt Sie eine weitere Dialogbox (Bild 7.9) »Gosub where if selected?« (»Welche Unterroutine aufrufen, wenn die Funktion aufgerufen wird?«). Geben Sie nun die Zeile (entweder eine Zeilennummer oder ein Label) ein, an die GeoBasic mit einem GOSUB springen soll, wenn diese Funktion angeklickt wird.

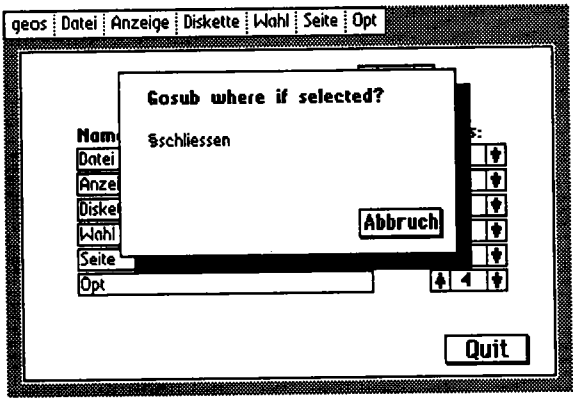


Bild 7.9: Eingabe der Zeile, die angesprungen werden soll

Bestätigen Sie die Eingabe mit Return oder klicken Sie auf »Abbruch«, wenn Sie nun doch keine Änderung mehr vornehmen möchten. Tippen Sie nichts ein, dann passiert beim Aufrufen dieser Funktion später in Ihrem Programm überhaupt nichts.

Hinweis: In Ihrem eigenen Interesse sollten Sie anstelle einer Zeilennummer immer ein Label hier angeben (z. B. @schließen oder @info). Dann brauchen Sie sich vor Aufruf des Menü-Editors keine Zeilennummern merken und können auch nach Einbau des Menüs in das eigene Programm die zugehörigen Routinen immer noch beliebig verschieben, ohne eine Zeilennummer im Menü-Editor ändern zu müssen.

Das »geos«-Abrollmenü beinhaltet nur eine Funktion, deren Namen und Einsprungzeile Sie wie bei den Funktionen in anderen Menüs ändern können. In fast allen GEOS-Programmen (GeoWrite, GeoPaint ...) wird diese Funktion als Info für das Programm genutzt, so daß beim Aufrufen eine Dialogbox erscheint, die Versionsnummer, Copyright-Hinweis und Namen des Autors des Programms anzeigt. Hier können Sie sich bei Ihren eigenen GeoBasic-Programmen ebenso verewigen!

7.5 Verlassen des Menü-Editors

Haben Sie alle Änderungen erledigt, wollen Sie verständlicherweise den Menü-Editor wieder verlassen, um das neue Menü in Ihr Programm einzubauen. Klicken Sie dazu einfach auf »Quit« am unteren rechten Bildschirmrand.

Es erscheint eine Dialogbox (Bild 7.10) und fragt, ob Sie die Änderungen speichern möchten oder nicht (»Save changes before quitting?«).

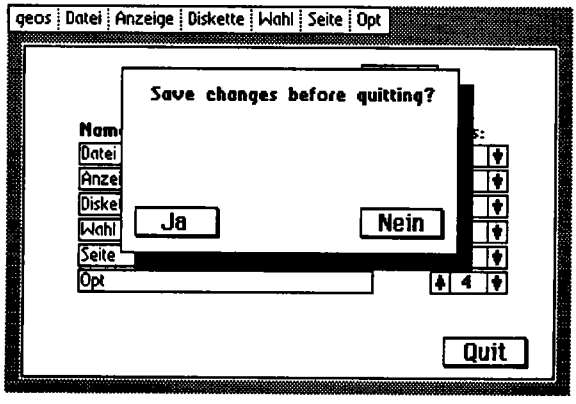


Bild 7.10: Verlassen des Menü-Editors

Wenn Sie auf »Ja« klicken, dann wird das Menü unter dem vorher von Ihnen bestimmten Namen auf Diskette gespeichert.

Beabsichtigen Sie, das Menü nicht zu speichern, so klicken Sie auf »Nein«. Sie gelangen dann in den Text-Editor zurück.

7.6 Nutzung des Menüs in eigenen GeoBasic-Programmen

Zögern Sie nicht, Menüs in Ihren Programmen zu verwenden. Die Bedienbarkeit wird dadurch stark verbessert und außerdem ist die Verwendung wirklich nicht sehr kompliziert.

Lesen Sie bitte in den Abschnitten 5.2 und 5.11 nach, wie sich das Einbauen und Aktivieren von Menüs mit MENU und MAINLOOP gestaltet.

8. Kapitel
Der Dialogbox-Editor

Mit dem Dialogbox-Editor können Sie bequem Dialogboxen mit Texten und Piktogrammen für Ihre eigenen GeoBasic-Programme entwerfen. Beachten Sie bitte in diesem Zusammenhang Abschnitt 5.4 (über den Einbau von Dialogboxen in eigene Programme).

8.1 Was ist eine Dialogbox?

Eine Dialogbox ist ein rechteckiger Bereich, der sich über den aktuellen Bildschirm legt, eine Information an den Benutzer übergibt oder vom Benutzer holt und anschließend wieder verschwindet. Dabei wird der Bildschirm wieder so hergestellt wie vor dem Erscheinen der Dialogbox.

Eine Dialogbox kann Text, Zahlen und Piktogramme enthalten. Normalerweise stellen dabei Text und Zahlen Erklärungen dar und der Benutzer klickt ein Piktogramm an, um dem Programm mitzuteilen, was er beabsichtigt.

Jedes GEOS-Programm – einschließlich GeoBasic – benutzt Dialogboxen zu diesem Zweck und Sie sollten möglichst auch diese sehr einfache Möglichkeit zur Kommunikation des Programms mit dem Benutzer nutzen. Ein Beispiel für eine Dialogbox sehen Sie in Bild 8.1.

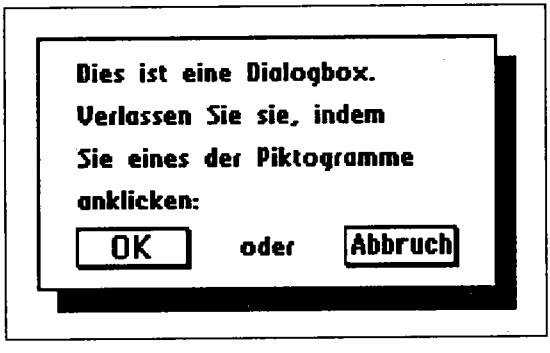


Bild 8.1: Eine Dialogbox

8.2 Starten des Dialogbox-Editors

Sie starten den Dialogbox-Editor durch Auswahl von »dialog« aus dem »Utilities«-Menü im GeoBasic-Text-Editor (Bild 8.2).

Bild 8.2: Das »Utilities«-Menü

8.3 Auswahl einer Dialogbox zur Bearbeitung

Nach dem Starten des Dialogbox-Editors erscheint eine Dialogbox auf dem Bildschirm (Bild 8.3), die der Dialogbox zur Auswahl eines vorhandenen GeoBasic-Dokuments (Abschnitt 3.2.2) ähnelt.

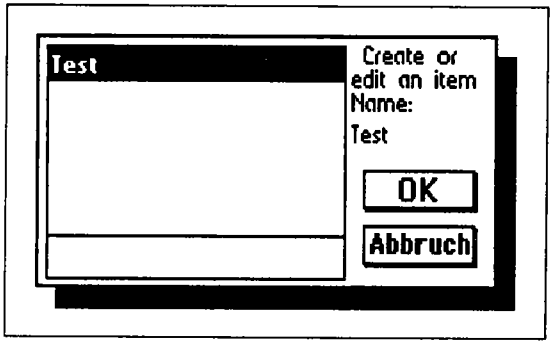


Bild 8.3: Auswahl einer Dialogbox

Mit »Abbruch« gelangen Sie wieder zurück in den Text-Editor. Möchten Sie eine neue Dialogbox erstellen, so geben Sie einfach einen Namen für diese Dialogbox ein und drücken auf [Return]. Ist bereits ein Name im Eingabefeld vorgegeben, dann müssen Sie diesen natürlich erst mit der [Del]-Taste löschen, bevor Sie den neuen Namen eingeben. Ein Dialogbox-Name darf bis zu fünf Zeichen lang sein.

Hinweis: Ist bereits eine Dialogbox mit dem eingegebenen Namen vorhanden, dann wird diese zur Weiterbearbeitung geöffnet und keine neue Dialogbox erstellt.

Wenn Sie eine schon erstellte Dialogbox weiterbearbeiten möchten, dann klicken Sie den zugehörigen Namen in der Auswahl-Dialogbox (er wird dann invertiert) und anschließend »OK« an.

Haben Sie den Namen einer bereits existierenden Dialogbox ausgewählt, dann bestimmen Sie im nächsten Bildschirm, was Sie mit dieser Dialogbox realisieren möchten (Bild 8.4).

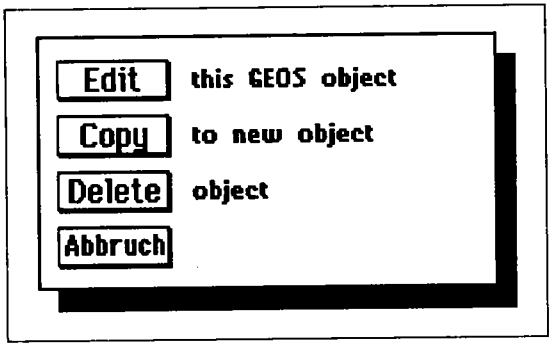


Bild 8.4: Auswahl der Funktion des Dialogbox-Editors

Durch Anklicken von »Edit« gelangen Sie in den eigentlichen Dialogbox-Editor (Abschnitt 8.4).

Mit »Copy« können Sie eine identische Kopie Ihrer Dialogbox anfertigen. Nach Anklicken des Piktogramms werden Sie zur Eingabe eines neuen Namens aufgefordert (Bild 8.5). Geben Sie den Namen ein und drücken Sie [Return]. Haben Sie Ihre Meinung geändert und möchten nun doch nicht kopieren, dann klicken Sie »Abbruch« an.

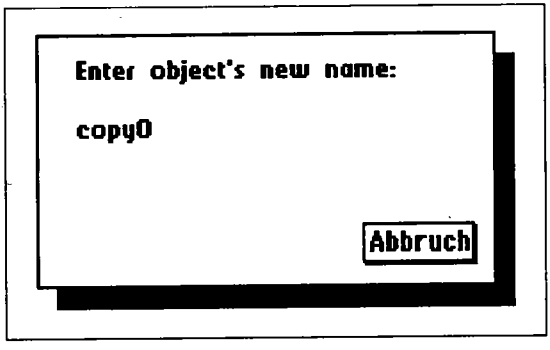


Bild 8.5: Eingabe eines Namens für die Kopie

»Delete« ermöglicht Ihnen das Löschen der aktuellen Dialogbox, beispielsweise wenn sie nicht mehr benötigt wird. Mit dieser Funktion sollten Sie sehr vorsichtig verfahren, da sie nicht mehr rückgängig gemacht werden kann.

Wollen Sie den Dialogbox-Editor verlassen, so klicken Sie auf »Abbruch«.

8.4 Bearbeiten einer Dialogbox

Haben Sie den Dialogbox-Namen und die Edit-Funktion ausgewählt, so erscheint der Dialogbox-Editor-Bildschirm (Bild 8.6).

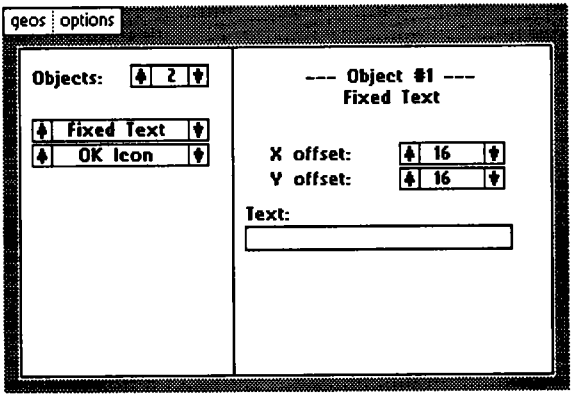


Bild 8.6: Der Dialogbox-Editor-Bildschirm

Der Dialogbox-Editor-Bildschirm ist in zwei Fenster aufgeteilt. Im linken Fenster können Sie die Anzahl und Arten der Objekte (Text, Piktogramme ...) einstellen, die in Ihrer Dialogbox erscheinen sollen. Auf der rechten Seite bestimmen Sie die Detail-Informationen für jedes Objekt.

Im linken Fenster befindet sich ein Feld mit einer Zahl darin und dem Wort »Objects:« (Objekte) daneben. Hier können Sie durch Anklicken der Pfeile links und rechts neben dem Feld die Anzahl der Objekte (maximal 8) einstellen. Bei jeder Änderung wechselt auch die Anzahl der Felder unter dieser Zeile, in denen Sie die Arten der Objekte einstellen können.

Durch Anklicken der Pfeile neben dem jeweilen Feld können Sie die Objekt-Art bestimmen. Für jedes Objekt müssen zusätzliche Daten, z. B. die Plazierung in der Dialogbox, im rechten Fenster spezifiziert werden. Um die Daten eines bestimmten Objekts im rechten Fenster anzuzeigen, klicken Sie auf das Feld dieses Objekts im linken Fenster.

Objekt-Arten sind:

1. **OK Icon** – Ein Piktogramm mit der Inschrift »OK«.

   »OK« wird normalerweise dazu benutzt, um die nächste Aktion eines Programms zu billigen. Im rechten Fenster müssen X- und Y-Position des Piktogramms angegeben werden. »OK« übergibt den Wert 1 bei Verwendung mit dem GeoBasic-DIALOG-Befehl.

2. **CANCEL Icon** – Ein Piktogramm mit der Inschrift »Abbruch«.

   »Abbruch« wird normalerweise dazu benutzt, um die nächste oder eine laufende Funktion abzubrechen. Im rechten Fenster müssen X- und Y-Position des Piktogramms angegeben werden. »Abbruch« übergibt den Wert 2 bei Verwendung mit dem GeoBasic-DIALOG-Befehl.

3. **YES Icon** – Ein Piktogramm mit der Inschrift »Ja«.

   Im rechten Fenster müssen X- und Y-Position des Piktogramms angegeben werden. »Ja« übergibt den Wert 3 bei Verwendung mit dem GeoBasic-DIALOG-Befehl.

4. **NO Icon** – Ein Piktogramm mit der Inschrift »Nein«.

   Im rechten Fenster müssen X- und Y-Position des Piktogramms angegeben werden. »Nein« übergibt den Wert 4 bei Verwendung mit dem GeoBasic-DIALOG-Befehl.

5. **OPEN Icon** – Ein Piktogramm mit der Inschrift »Öffnen«.

   »Öffnen« wird normalerweise dazu benutzt, um eine Datei zu öffnen. Im rechten Fenster müssen X- und Y-Position des Piktogramms angegeben werden. »Öffnen« übergibt den Wert 5 bei Verwendung mit dem GeoBasic-DIALOG-Befehl.

6. **DISK Icon** – Ein Piktogramm mit der Inschrift »Disk«.

   »Disk« wird oft dazu benutzt, um das Wechseln der Diskette dem Programm mitzuteilen. Im rechten Fenster müssen X- und Y-Position des Piktogramms angegeben werden. »Disk« übergibt den Wert 6 bei Verwendung mit dem GeoBasic-DIALOG-Befehl.

7. **Fixed Text** – Eine Zeichenketten-Konstante.

   Normalerweise werden Zeichenketten-Konstanten in Dialogboxen verwendet, um dem Benutzer wichtige Informationen über die nächste Funktion zu vermitteln. Im Gegensatz zu den obigen Piktogrammen können Sie durch Anklicken des Textes die Dialogbox nicht verlassen, wobei kein Wert an DIALOG übergeben wird.

   Die Zeichenketten-Konstante wird automatisch in Fettschrift (Boldface) in der Dialogbox ausgegeben. Möchten Sie einen anderen Schriftstil für die Ausgabe verwenden, so müssen Sie die Zeichenkette per »Variable Text« (Kapitel 8) in die Dialogbox einbinden.

Im rechten Fenster müssen X- und Y-Position der Zeichenkette und die Zeichenketten-Konstante (maximal 30 Zeichen) selbst bestimmt werden.

8. **Variable Text** – Ein beliebiger Zeichenketten-Ausdruck.

   Der Ausdruck darf Konstanten und Variablen (aber keine Funktionen wie z. B. CHR$) enthalten, beispielsweise "Sein Name ist " + A$ + ". ". Das Ergebnis des Ausdrucks wird in der Dialogbox ausgegeben. Großer Vorteil dieser Möglichkeit gegenüber »Fixed Text« ist, daß der Text (als auch der Schriftstil) veränderlich ist und so mit einer Dialogbox verschiedene Informationen an den Benutzer übergeben werden können. Der »Variable Text« kann nicht angeklickt werden und übergibt auch keinen Wert an den DIALOG-Befehl.

   Der Zeichenketten-Ausdruck sollte ein Leerzeichen als Abschlußkennung (zum Beispiel ausgabe$ + " ") enthalten, da ansonsten das letzte Zeichen des Ausdrucks nicht mit angezeigt wird.
   Die Ausgabe des Zeichenketten-Ausdrucks erfolgt normalerweise in Normalschrift (Plaintext). Durch Verwendung von Steuercodes innerhalb des Ausdrucks kann aber dieser Stil gewechselt werden. Zwar können Sie keine Funktionen innerhalb des Ausdrucks verwenden (z. B. CHR$(24) + "Dies ist Fettschrift. "), Sie können aber Variablen mit den jeweiligen Steuercodes belegen (z. B. fett$ = CHR$(24)) und dann diese Variablen in den Ausdruck einbauen: fett$ + "Dies ist Fettschrift. ".

   Im rechten Fenster müssen X- und Y-Position des Ausdrucks und der Ausdruck selbst (maximal 30 Zeichen) angegeben werden.

9. **User Icon** – Ein Piktogramm, dessen Aussehen Sie im Grafik-Editor (Kapitel 10) frei definieren können.

   Sie können keine Grafiken für diese Funktion wählen, die nicht speicherresident sind (Abschnitt 10.6.2). Es wird ansonsten ein »Bad data in object«-Fehler ausgegeben.

   Im rechten Fenster müssen X- und Y-Position des Piktogramms, der Name, den Sie der Grafik bei der Erstellung gegeben haben und der Wert, der an den Befehl DIALOG übergeben werden soll (von 20 bis 255), angegeben werden.

   Die Bestimmung der näheren Werte im rechten Fenster erfolgt bei Zahlen mit den bekannten Pfeilen und bei Texten und Ausdrücken durch Eingabe in die vorgesehenen Felder mit abschließender Betätigung von Return.

   Die X-Koordinate von Piktogrammen, die im rechten Fenster zu bestimmen ist, kann nur in 8-Pixel-Schritten variiert werden. Dies ist bedingt durch den internen Aufbau von GEOS-Grafiken und leider nicht zu ändern.

Hinweis: Haben Sie in Ihrer Dialogbox keine Möglichkeit gelassen, durch Anklicken eines Piktogramms diese zu verlassen, so kann sie durch Betätigen der Maustaste an einer beliebigen Bildschirmposition verlassen werden.

8.5 Das »options«-Menü

Im Dialogbox-Editor stellt Ihnen eine Menüleiste weitere Funktionen zur Erleichterung der Arbeit an Dialogboxen zur Verfügung.

8.5.1 »guides on/off«

Bei Aktivierung des Dialogbox-Editors ist eine Hilfsfunktion eingeschaltet, mit der sich die Objekte (z. B. Text) innerhalb der Dialogbox leichter positionieren lassen. Es werden dann nämlich nur bestimmte Koordinaten zur Auswahl freigegeben.

Möchten Sie nun die Objekte an eine Position bringen, die nicht zu den Standard-Positionen gehört, dann wählen Sie aus dem »options«-Menü »guides off« aus. Sie können jetzt alle Koordinaten frei wählen (X: 0 bis 176 (bei Grafiken und Piktogrammen in 8-Pixel-Schritten), Y: 0 bis 95).

Sollten Sie dann wieder die Positionierhilfe aktivieren wollen, genügt es, den Menüpunkt (der dann »guides on« heißt) nochmal anzuklicken.

Hinweis: Lassen Sie sich nicht durch den Menüpunkttext irritieren: angezeigt werden immer Funktionen, keine Zustände. Darum erscheint auch bei eingeschalteter Positionierhilfe der Text »guides off« und nicht »guides on«!

8.5.2 »display dialog box«

Ihre selbstkonstruierte Dialogbox wird auf dem Bildschirm angezeigt. »User Icons« werden als dunkle Balken dargestellt und anstelle des »Variable Text« wird »(String expression)« ausgegeben.

Klicken Sie, wenn Sie zum Editor zurückgelangen möchten, einfach eines der Piktogramme an oder – wenn kein Piktogramm vorhanden ist – klicken Sie auf eine beliebige Stelle des Bildschirms. Ein Beispiel für diese Funktion sehen Sie in Bild 8.7.

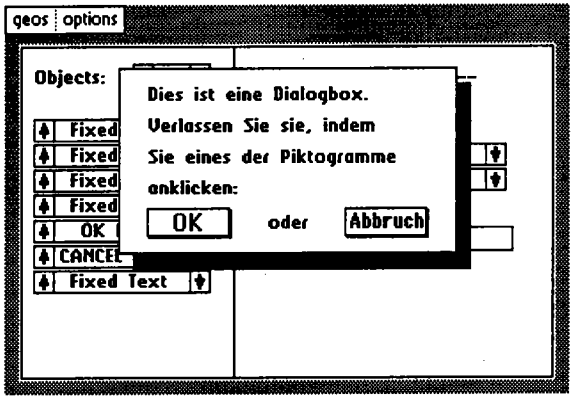


Bild 8.7: Eine selbsterstellte Dialogbox

8.5.3 »quit«

Sie können den Dialogbox-Editor über diese Funktion verlassen. Es erscheint eine Dialogbox mit der Frage »Save changes before quitting?« (Änderungen vor dem Verlassen speichern?) (Bild 8.8).

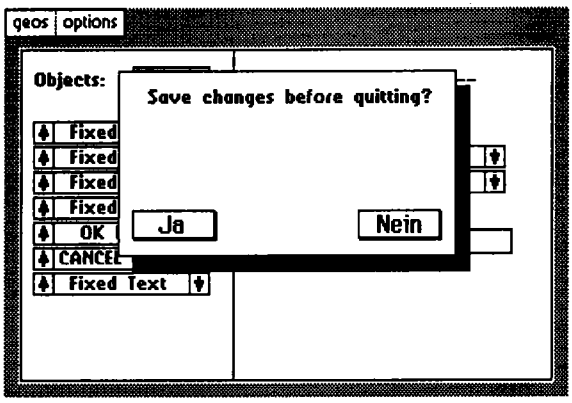


Bild 8.8: Verlassen des Dialogbox-Editors

Wenn Sie »Ja« anklicken, dann wird Ihre Dialogbox unter dem vorher von Ihnen bestimmten Namen gespeichert. Möchten Sie die Dialogbox nicht speichern, dann klicken Sie auf »Nein«.

8.6 Nutzung der Dialogbox in eigenen Programmen

Alle Informationen zum Einbauen von Dialogboxen (Befehle DIALOG, DBFILE, DBSTRN) in GeoBasic-Programme können Sie in Abschnitt 5.4 nachlesen.

9. Kapitel
Der Piktogrammlisten-Editor

Mit dem Piktogrammlisten-Editor können Sie bequem Piktogramme für die Verwendung in eigenen GeoBasic-Programmen zusammenstellen.

Beachten Sie bitte in diesem Zusammenhang die Abschnitt 5.3 und 5.11 über den Einbau und die Steuerung von Piktogrammlisten in eigenen Programmen.

9.1 Was ist ein Piktogramm und eine Piktogrammliste?

Piktogramme sind kleine Grafiken (Sinnbilder), die der Benutzer eines Programms anklicken kann, um eine bestimmte Funktion aufzurufen. Verschiedene Piktogramme, die beispielsweise GeoPaint zur Verfügung stellt, finden Sie in Bild 9.1.

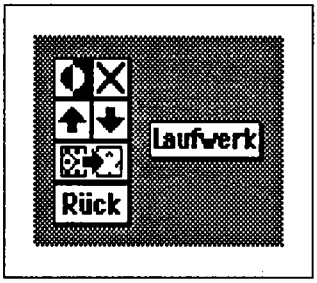


Bild 9.1: Piktogramme

Die Piktogramme werden intern in einer Liste (»Piktogrammliste«), die alle Daten der Piktogramme enthält, verwaltet.

Zuerst müssen die Grafiken für die Piktogramme im Grafik-Editor (Kapitel 10) entworfen werden. Anschließend können Sie mit dem Piktogrammlisten-Editor die Namen dieser Grafiken und noch einige zusätzliche Informationen zu einer Piktogrammliste zusammenstellen, die Sie dann in GeoBasic-Programmen verwenden können.

9.2 Starten des Piktogrammlisten-Editors

Den Piktogrammlisten-Editor starten Sie durch Auswahl von »icon« aus dem »Utilities«-Menü im GeoBasic-Text-Editor (Bild 9.2).

Bild 9.2: Das »Utilities«-Menü

9.3 Auswahl einer Piktogrammliste zur Bearbeitung

Nach dem Starten des Piktogrammlisten-Editors erscheint eine Dialogbox auf dem Bildschirm (Bild 9.3), die sehr der Dialogbox zur Auswahl eines vorhandenen GeoBasic-Dokuments (Abschnitt 3.2.2) ähnelt.

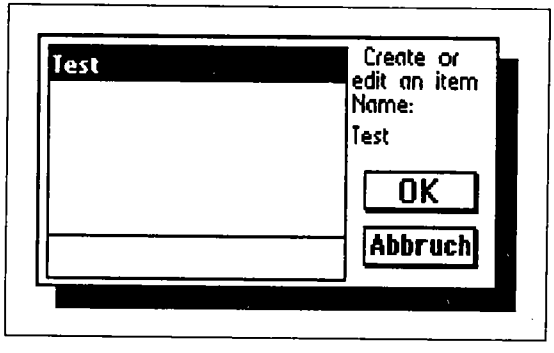


Bild 9.3: Auswahl einer Piktogrammliste

Mit »Abbruch« gelangen Sie wieder zurück in den Text-Editor. Möchten Sie eine neue Piktogrammliste erstellen, so geben Sie einfach einen Namen für diese Piktogrammliste ein und drücken Sie Return. Ist bereits ein Name im Eingabefeld vorgegeben, dann können Sie diesen natürlich erst mit der Del-Taste löschen. Ein Piktogrammlisten-Name darf bis zu fünf Zeichen lang sein.

Hinweis: Ist bereits eine Piktogrammliste mit dem eingegebenen Namen vorhanden, dann wird diese zur Weiterbearbeitung geöffnet und keine neue Piktogrammliste angelegt.

Wenn Sie eine schon erstellte Piktogrammliste weiterbearbeiten möchten, dann klicken Sie den zugehörigen Namen in der Auswahl-Dialogbox (er wird dann invertiert) und anschließend »OK« an.

Haben Sie den Namen einer existierenden Piktogrammliste ausgewählt, dann bestimmen Sie in der nächsten Dialogbox, was Sie mit dieser Piktogrammliste realisieren möchten (Bild 9.4).

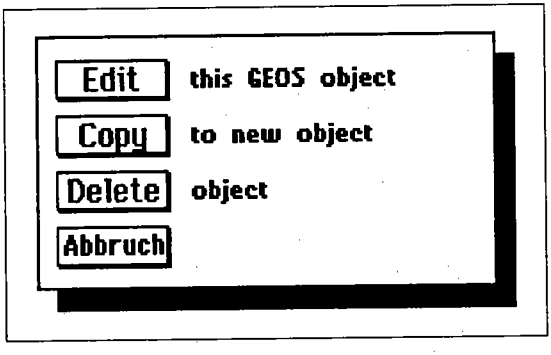


Bild 9.4: Auswahl der Funktion des Piktogrammlisten-Editors

Durch Anklicken von »Edit« gelangen Sie in den eigentlichen Piktogrammlisten-Editor (Abschnitt 9.4).

Mit »Copy« können Sie eine identische Kopie Ihrer Piktogrammliste anfertigen. Nach dem Anklicken werden Sie zur Eingabe eines neuen Namens aufgefordert (Bild 9.5). Geben Sie den Namen ein und drücken Sie Return. Haben Sie Ihre Meinung geändert und möchten nun doch nicht kopieren, dann klicken Sie »Abbruch« an.

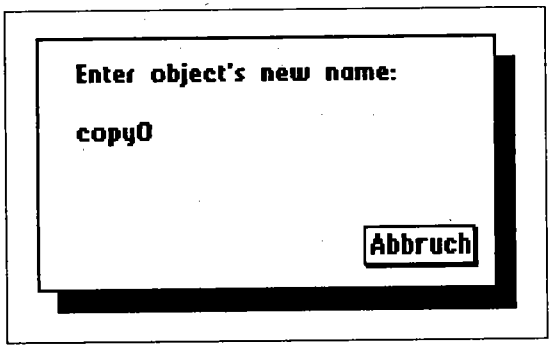


Bild 9.5: Eingabe eines Namens für die Kopie

»Delete« ermöglicht Ihnen das Löschen der aktuellen Piktogrammliste, beispielsweise wenn sie nicht mehr benötigt wird. Mit dieser Funktion sollten Sie sehr vorsichtig verfahren, da sie nicht mehr rückgängig gemacht werden kann.

Wollen Sie den Piktogrammlisten-Editor verlassen, so klicken Sie auf »Abbruch«.

9.4 Bearbeiten einer Piktogrammliste

Haben Sie einen Listen-Namen und die Edit-Funktion ausgewählt, so erscheint der Piktogrammlisten-Editor-Bildschirm (Bild 9.6).

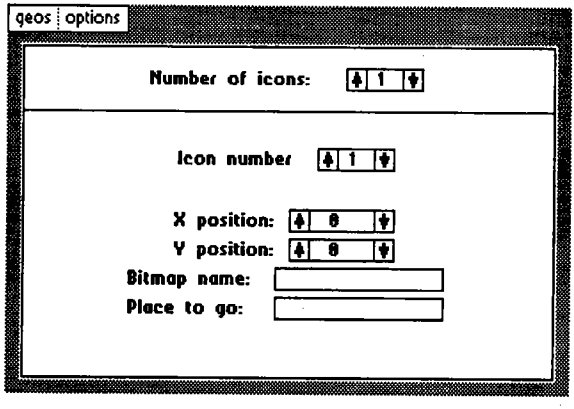


Bild 9.6: Der Piktogrammlisten-Editor-Bildschirm

Im oberen Bereich des Bildschirms finden Sie das Feld »Number of icons:« (Anzahl der Piktogramme:), das – wie der Name schon sagt – die Anzahl der Piktogramme enthält. Die maximale Anzahl von Piktogrammen ist 31. Zur Verwaltung von noch mehr Piktogrammen ist GEOS nicht in der Lage; 31 ist aber trotz allem eine stolze Zahl, die es erst einmal zu erreichen gilt!

Mit den Rollpfeilen an den Seiten des Feldes nehmen Sie die Einstellung vor und können diese natürlich bei Bedarf auch nachträglich noch ändern!

Unter dem Feld »Number of icons:« befindet sich ein weiteres Feld, in dem Sie die Nummer des Piktogramms in der Liste (ebenfalls mit Rollpfeilen) festlegen, das Sie bearbeiten möchten (»Icon number:«).

Weiter unten können Sie die Position des gerade bearbeiteten Piktogramms auf dem Bildschirm festlegen (»X position« und »Y position«). Die X-Koordinate kann zwischen 0 und 312

variieren und läßt sich nur in 8-Pixel-Schritten ändern, während die Y-Position eine beliebige Zahl zwischen 0 und 199 sein darf.

Unter den Feldern für die Einstellung der X- und Y-Position des Piktogramms ist eine Zeile zur Eingabe des Namens der Grafik (»Name of the bitmap«), die das Piktogramm darstellen soll, reserviert. Geben Sie hier den (höchstens fünf Zeichen langen) Namen an, den Sie im Grafik-Editor (Kapitel 10) für Ihre Grafik verwendet haben.

Hinweis: Eine Grafik, die nicht speicherresident ist (siehe Abschnitt 10.6.2), kann nicht als Grafik für ein Piktogramm verwendet haben. Falls Sie dies nicht beachten, wird ein »Bad data in object«-Fehler ausgegeben.

Die Routine, die angesprungen werden soll, wenn das Piktogramm vom Benutzer angeklickt wird, können Sie im Feld »Place to go:« (Zeile, in die gesprungen werden soll:) eingeben. Verwenden Sie hier möglichst Labels, da diese Ihnen eine Menge Arbeit ersparen können, wenn Sie nachträglich Ihr Programm ändern. Die Eingabe einer Zeilennummer ist aber auch möglich. Geben Sie in diesem Feld nichts an, dann reagiert Ihr Programm nicht auf das Anklicken des Piktogramms.

Um den Cursor in eines der beiden Felder (Name der Grafik und Zeile, an die gesprungen werden soll) zu setzen, klicken Sie einfach auf die jeweilige Zeile. Mit der Return-Taste können Sie den Cursor zwischen den beiden Feldern hin- und herspringen lassen.

9.5 Das »options«-Menü

9.5.1 »quit«

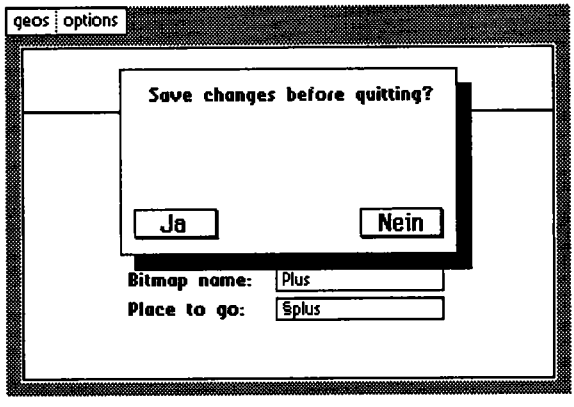


Bild 9.7: Verlassen des Piktogrammlisten-Editors

Sie können den Piktogrammlisten-Editor über diese Funktion verlassen. Es erscheint eine Dialogbox mit der Frage »Save changes before quitting?« (Änderungen vor dem Verlassen speichern?) (Bild 9.7).

Wenn Sie »Ja« anklicken, dann wird Ihre Piktogrammliste unter dem vorher von Ihnen bestimmten Namen gespeichert. Möchten Sie die Piktogrammliste nicht speichern, so klicken Sie auf »Nein«.

9.6 Nutzung der Piktogrammliste in eigenen Programmen

Es informiert Sie über die Nutzung von Piktogrammlisten in eigenen Programmen (mit ICON) Abschnitt 5.3. Sie finden alle Informationen über die Steuerung der Piktogramme durch die GEOS-Mainloop (über den MAINLOOP-Befehl) in Abschnitt 5.11.

10. Kapitel
Der Grafik-Editor

Mit dem Grafik-Editor können Sie kleine Grafiken für Ihre eigenen GeoBasic-Programme zeichnen.

Beachten Sie bitte in diesem Zusammenhang Abschnitt 5.1.2 beim Einbau als reine Illustration, Kapitel 8, wenn Sie die Grafik in einer Dialogbox verwenden möchten, oder Kapitel 9, wenn Sie die Grafik für Ihre Piktogramme gebrauchen.

10.1 Was ist eine Grafik?

Eine Grafik ist ein Bild, das aus vielen kleinen Punkten, genannt Pixel, zusammengesetzt ist. Jedes Bild in diesem Buch ist beispielsweise eine Grafik.

Unter GEOS sind alle Grafiken zweifarbig (schwarzweiß), mehrfarbige Grafiken sind nicht ohne weiteres möglich. Ein Beispiel für eine Grafik und ihren Aufbau finden Sie in Bild 10.1.

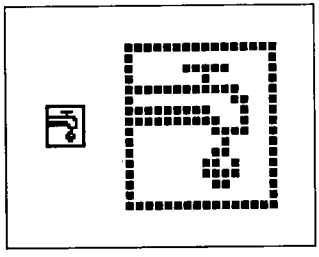


Bild 10.1: Links eine Grafik in Originalgröße, rechts in stark vergrößerter Form

Grafiken, die Sie mit dem Grafik-Editor erstellen oder verändern möchten, dürfen maximal 48 Pixel breit und maximal 42 Pixel hoch sein. Größere Grafiken können nur mit GeoPaint erstellt werden und müssen per Foto-Scrap eingebunden werden.

10.2 Starten des Grafik-Editors

Den Grafik-Editor starten Sie durch Auswahl von »bitmap« aus dem »Utilities«-Menü im GeoBasic-Text-Editor (Bild 10.2).

Bild 10.2: Das »Utilities«-Menü

10.3 Auswahl einer Grafik zur Bearbeitung

Nach dem Starten des Grafik-Editors erscheint eine Dialogbox auf dem Bildschirm (Bild 10.3), die der Dialogbox zur Auswahl eines vorhandenen GeoBasic-Dokuments (Abschnitt 3.2.2) ähnelt.

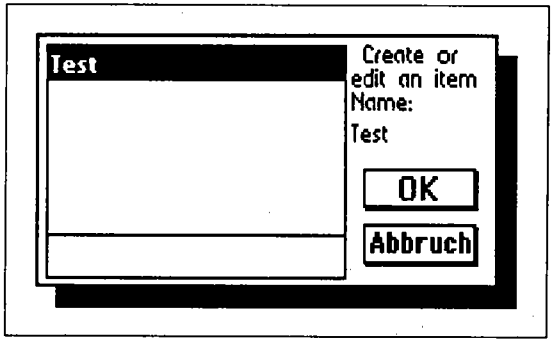


Bild 10.3: Auswahl einer Grafik

Mit »Abbruch« gelangen Sie wieder zurück in den Text-Editor. Möchten Sie eine neue Grafik erstellen, dann geben Sie einfach einen Namen für diese Grafik ein und drücken auf Return. Ist bereits ein Name im Eingabefeld vorgegeben, dann können Sie diesen natürlich erst mit der Del-Taste löschen. Ein Grafikname darf bis zu fünf Zeichen lang sein.

Hinweis: Ist bereits eine Grafik mit dem eingegebenen Namen vorhanden, dann wird diese zur Weiterbearbeitung geöffnet und keine neue Grafik erstellt.

Wenn Sie eine schon erstellte Grafik weiterbearbeiten möchten, dann klicken Sie deren Namen in der Auswahl-Dialogbox (er wird dann invertiert) und anschließend »OK« an.

Haben Sie den Namen einer existierenden Grafik ausgewählt, dann bestimmen Sie in der nächsten Dialogbox, was Sie mit dieser Grafik realisieren möchten (Bild 10.4).

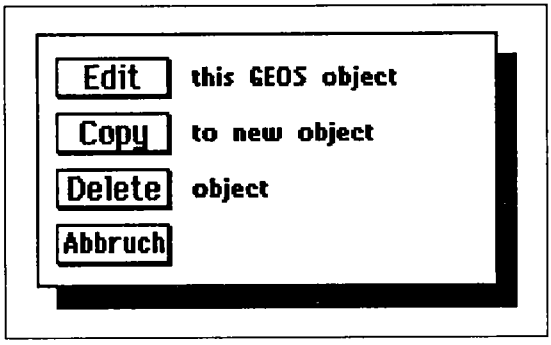


Bild 10.4: Auswahl der Funktion des Grafik-Editors

Durch Anklicken von »Edit« gelangen Sie in den eigentlichen Grafik-Editor (Abschnitt 10.4).

Sollte das »Edit«-Piktogramm nicht erscheinen, so haben Sie eine Grafik ausgewählt, die nicht speicherresident und nicht mehr veränderbar ist (Abschnitt 10.6.2).

Mit »Copy« können Sie eine identische Kopie Ihrer Grafik anfertigen. Nach dem Anklicken werden Sie zur Eingabe eines neuen Namens aufgefordert (Bild 10.5). Geben Sie den Namen ein und drücken Sie Return. Haben Sie Ihre Meinung geändert und möchten nun doch nicht kopieren, dann klicken Sie »Abbruch« an.

»Delete« ermöglicht das Löschen der aktuellen Grafik, beispielsweise wenn sie nicht mehr benötigt wird. Mit dieser Funktion sollten Sie sehr vorsichtig verfahren, da sie nicht mehr rückgängig gemacht werden kann. Wollen Sie den Grafik-Editor verlassen, so klicken Sie auf »Abbruch«.

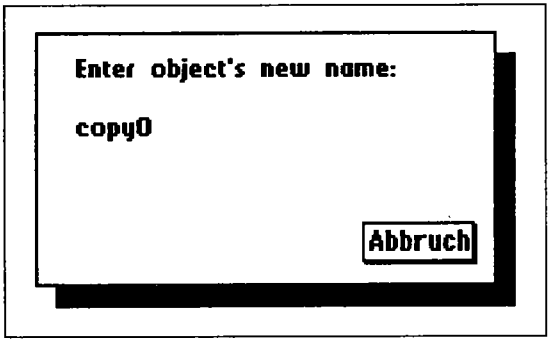


Bild 10.5: Eingabe eines Namens für die Kopie

10.4 Bearbeiten einer Grafik

Haben Sie einen Grafiknamen und die Edit-Funktion ausgewählt, so erscheint der Grafik-Editor-Bildschirm (Bild 10.6).

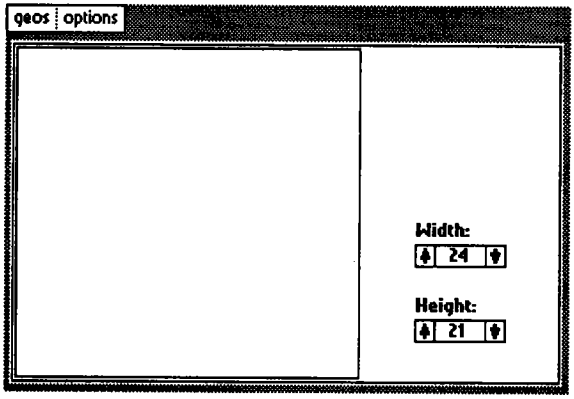


Bild 10.6: Der Grafik-Editor-Bildschirm

Der Bildschirm ist in zwei Fenster aufgeteilt:

Das linke Fenster ist das Editier-Fenster, in dem Sie das Aussehen der Grafik Punkt für Punkt ändern können.

Im rechten Fenster sehen Sie Ihre Grafik in Originalgröße, also so, wie sie später in Ihren GeoBasic-Programmen erscheint. Außerdem können Sie hier noch die Größe Ihrer Grafik einstellen (auch nachträglich). Das Feld mit der Bezeichnung »Width« beinhaltet die Breite Ihrer Grafik, die Sie in 8-Pixel-Schritten mit den beiden Pfeilen verändern können. Im Feld »Height« stellen Sie die Höhe Ihrer Grafik mit den Pfeilen ein.

Wie bereits erwähnt, hat die größte Grafik, die Sie im Grafik-Editor erstellen können, eine Breite von 48 und eine Höhe von 42 Pixel. Bei der Einstellung der Größe ändert sich natürlich auch die Größe des Editier-Fensters.

Zusätzlich kann sich dabei auch der Vergrößerungsmodus ändern: Ist die Grafik klein, dann wird sie im Editier-Fenster in einer ziemlich starken Vergrößerung angezeigt, ist die Grafik hingegen sehr groß, dann erscheint sie in einer weniger starken Vergrößerung.

Verändern können Sie die Grafik im Editier-Fenster. Jedesmal, wenn Sie den Mauszeiger darüberbewegen, wird der Mauszeiger zu einem Zeichenstift. Der Zeichenstift kann eine von drei Farben annehmen: Ist er blau, dann wird die Grafik beim Überqueren des Editier-Fensters nicht beeinflußt.

Um mit dem Zeichnen zu beginnen, plazieren Sie den Zeichenstift auf einem leeren Punkt und klicken einmal. Seine Farbe wechselt jetzt nach schwarz und jeder Punkt, über den seine Spitze bewegt wird, wird gesetzt. Möchten Sie wieder aufhören zu zeichnen, dann klicken Sie nochmals.

Möchten Sie einige Punkte löschen, dann plazieren Sie den Zeichenstift auf einen gesetzten Punkt und klicken einmal. Seine Farbe wechselt nach rot und alle Punkte, über die seine Spitze bewegt wird, werden gelöscht. Um mit dem Löschen aufzuhören, klicken Sie noch einmal.

10.5 Das »Geos«-Menü

Neben dem üblichen Menüpunkt »Info«, der Informationen über den Grafik-Editor bereitstellt, werden im »Geos«-Abrollmenü alle Hilfsprogramme der aktuellen Diskette bereitgestellt. Sie können diese durch Anklicken auswählen.

Sehr hilfreich ist dies bei der Einbindung von Grafiken aus GeoPaint, die Sie vorher in einem Foto-Album archiviert haben und nun aus dem Foto-Manager per Foto-Scrap direkt in den Grafik-Editor übertragen können.

10.6 Das »options«-Menü

10.6.1 »paste photo scrap«

Foto-Scraps enthalten Ausschnitte aus einem GeoPaint-Dokument. Ihre Erzeugung können Sie in Ihrem GEOS-Benutzerhandbuch nachlesen. Mit diesem Menüpunkt können Sie den Grafik-Ausschnitt aus dem Foto-Scrap in die gerade editierte Grafik übertragen.

Ist die Grafik breiter als 48 oder höher als 42 Pixel, dann kann sie nicht speicherresident gemacht werden, da sie zu groß für das Editierfeld ist. In diesem Falle wird die Grafik auf Diskette ausgelagert und der Grafik-Editor verlassen, da eine Veränderung wegen der Größe sowieso nicht möglich ist. In gewisser Weise ist dies als Vorteil zu sehen, da die Grafik keinen Variablen-Speicherplatz verbraucht (Abschnitt 10.6.2).

10.6.2 »disk loadable«

Mit dieser Funktion können Sie Grafiken, die normalerweise immer im Variablenspeicher stehen und trotz eines komprimierten Speicherverfahrens viel Platz verbrauchen, auf Diskette auslagern. Die Grafik ist dann nicht mehr speicherresident, d. h. sie ist nur in dem Moment im Speicher, in dem sie auch wirklich benötigt wird.

Den Vorteil des Platzsparens erkaufen Sie sich aber mit dem Nachteil, daß die Grafik nicht mehr mit dem Grafik-Editor verändert werden kann und nur noch als Illustration in Ihrem Programm dienen kann. Eine Verwendung in Dialogboxen oder als Piktogramm ist dann nicht mehr möglich.

Wählen Sie diese Funktion im »options«-Menü an, so müssen Sie noch einmal extra bestätigen, daß Sie dies auch wirklich möchten (Bild 10.7).

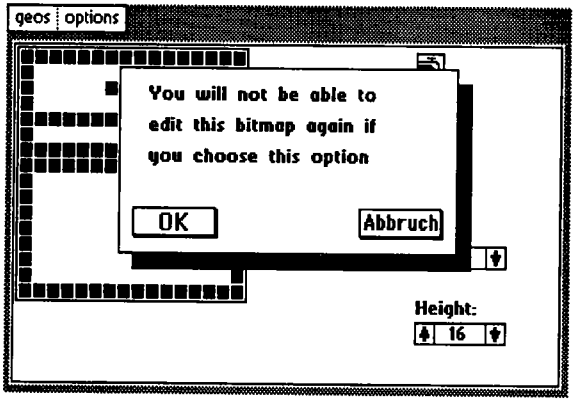


Bild 10.7: Auslagern einer Grafik auf Diskette zum Sparen von Speicherplatz

Nach erfolgreichem Abschluß dieser Funktion wird der Grafik-Editor automatisch verlassen.

10.6.3 »erase bitmap«

Die gesamte Grafik wird gelöscht. Sie sollten sehr vorsichtig mit dieser Funktion umgehen, da sie nicht rückgängig gemacht werden kann!

10.6.4 »quit«

Sie können den Grafik-Editor über diese Funktion verlassen. Es erscheint eine Dialogbox mit der Frage »Save changes before quitting?« (Änderungen vor dem Verlassen speichern?) (Bild 10.8).

Wenn Sie »Ja« anklicken, dann wird Ihre Grafik unter dem vorher von Ihnen bestimmten Namen gespeichert. Möchten Sie die Grafik nicht speichern, so klicken Sie auf »Nein«.

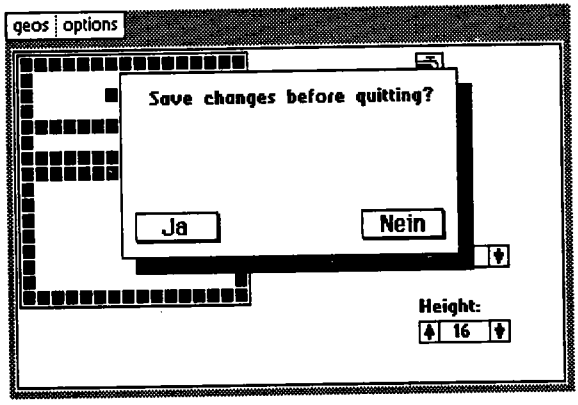


Bild 10.8: Verlassen des Grafik-Editors

10.7 Nutzung der Grafik in eigenen Programmen

Die mit dem Grafik-Editor erstellten Grafiken lassen sich auf vielfache Weise in Ihren eigenen Programmen nutzen. Sie können sie in Dialogboxen einbauen (Kapitel 8), als reine Augenweide auf den Bildschirm bringen (Abschnitt 5.1.2) oder für Piktogramme verwenden (Kapitel 9).

11. Kapitel
Der Sprite-Editor

Mit dem Sprite-Editor können Sie Sprites für Ihre eigenen GeoBasic-Programme entwerfen und ihre Parameter bestimmen. Beachten Sie in diesem Zusammenhang auch Abschnitt 5.6 über den Einbau von Sprites in GeoBasic-Programme.

11.1 Was ist ein Sprite?

Sprites sind kleine Bilder, die auf dem Bildschirm erscheinen können, ohne den Hintergrund zu verändern.

Ihnen stehen sechs Sprites zur Verfügung. Zwar ermöglichen der C 64 und C128 eigentlich die Verwendung von acht Sprites, wobei aber zwei vom GEOS-System (zur Darstellung des Mauszeigers und des Text-Cursors) benötigt werden. Die Möglichkeiten, die Ihnen Sprites bieten, werden im folgenden beschrieben.

11.2 Starten des Sprite-Editors

Den Sprite-Editor starten Sie durch Auswahl von »sprite« aus dem »Utilities«-Menü im GeoBasic-Text-Editor (Bild 11.1).

Bild 11.1: Das »Utilities«-Menü

11.3 Auswahl der Sprites zur Bearbeitung

Nach dem Starten des Sprite-Editors erscheint eine Dialogbox auf dem Bildschirm (Bild 11.2), die der Dialogbox zur Auswahl eines vorhandenen GeoBasic-Dokuments (Abschnitt 3.2.2) ähnelt.

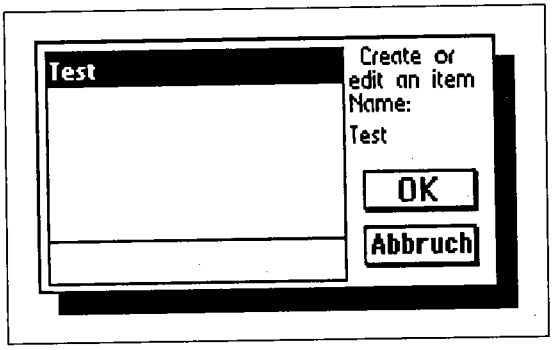


Bild 11.2: Auswahl des Sprites

Mit »Abbruch« gelangen Sie wieder zurück in den Text-Editor. Möchten Sie ein neues Sprite erstellen, dann geben Sie einfach einen Namen für dieses Sprite ein und drücken Sie Return. Ist bereits ein Name im Eingabefeld vorgegeben, dann können Sie diesen natürlich erst mit der Del-Taste löschen. Ein Sprite-Name darf bis zu fünf Zeichen lang sein.

Hinweis: Ist bereits ein Sprite mit dem eingegebenen Namen vorhanden, dann wird dieses zur Weiterbearbeitung geöffnet und kein neues Sprite erstellt.

Wenn Sie ein schon erstelltes Sprite weiterbearbeiten möchten, dann klicken Sie dessen Namen in der Auswahl-Dialogbox (er wird dann invertiert) und anschließend »OK« an.

Haben Sie den Namen eines existierenden Sprites ausgewählt, dann bestimmen Sie in der nächsten Dialogbox, was Sie mit diesem Sprite realisieren möchten (Bild 11.3).

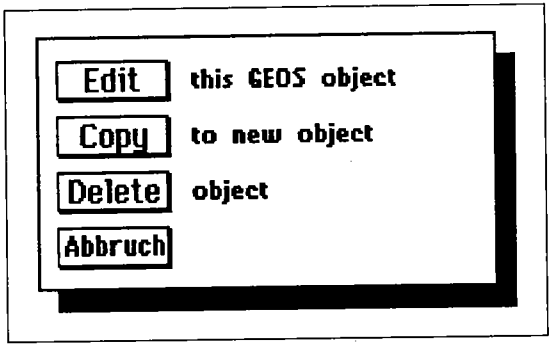


Bild 11.3: Auswahl der Funktion des Sprite-Editors

Durch Anklicken von »Edit« gelangen Sie in den eigentlichen Sprite-Editor (Abschnitt 11.4).

Mit »Copy« fertigen Sie eine identische Kopie Ihres Sprites an. Nach dem Anklicken werden Sie zur Eingabe eines neuen Namens aufgefordert (Bild 11.4). Geben Sie den Namen ein und drücken Sie Return. Haben Sie Ihre Meinung geändert und möchten nun doch nicht kopieren, so klicken Sie »Abbruch« an.

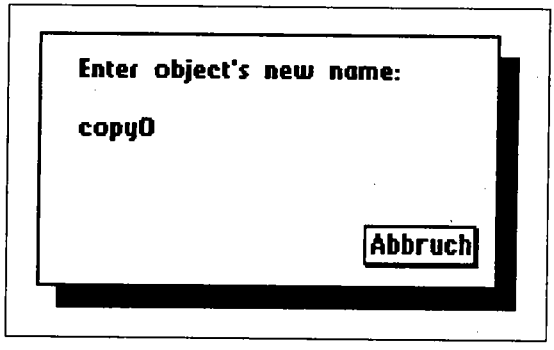


Bild 11.4: Eingabe eines Namens für die Kopie

»Delete« ermöglicht das Löschen des aktuellen Sprites, beispielsweise wenn es nicht mehr benötigt wird. Mit dieser Funktion sollten Sie sehr vorsichtig verfahren, da sie nicht mehr rückgängig gemacht werden kann. Wollen Sie den Sprite-Editor verlassen, so klicken Sie auf »Abbruch«.

11.4 Bearbeiten eines Sprites

Haben Sie einen Sprite-Namen und die Edit-Funktion ausgewählt, so erscheint der Attribut-Bildschirm (Bild 11.5).

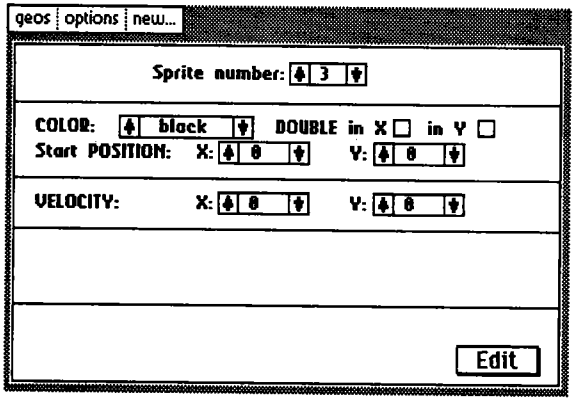


Bild 11.5: Der Attribut-Bildschirm

Erstellen Sie ein neues Sprites, so erscheinen nur die nötigsten Einstellungsmöglichkeiten auf dem Bildschirm, alle weiteren können Sie über das »new«- sowie das »options«-Menü hinzuschalten.

In der oberen Bildschirmzeile stellen Sie die »Sprite number« (Nummer des Sprites) ein. Verändern dürfen Sie eines der Sprites 3 bis 8, die Sprites 1 und 2 werden (wie gesagt) vom GEOS-System benötigt. Klicken Sie auf die Pfeile an den Seiten des Feldes, bis das richtige Sprite angewählt ist.

Im Feld »Color« können Sie die Farbe des Sprites einstellen. Zur Auswahl stehen 16 verschiedene Farben. Obwohl Ihre Sprites dreifarbig sein dürfen, können zwei dieser Farben nicht für jedes Sprite einzeln gesetzt werden, sondern sind bei allen Sprites gleich. Diese beiden Farben werden in GeoBasic mit dem SPRCOL-Befehl oder mit der Funktion »sprcol« aus dem »Edit«-Menü ausgewählt. Nur eine Farbe kann im Feld »Color« gesetzt werden, und zwar für jedes Sprite einzeln. Klicken Sie hierzu auf die Pfeile an den Seiten des Feldes.

Die Größe von Sprites kann in X- und Y-Richtung verdoppelt werden. Dabei verändert sich nicht die Anzahl der veränderbaren Pixels eines Sprites, sondern die Größe jedes Pixels selbst. Verwenden Sie zur Einstellung der X- und Y-Vergrößerung die Felder »Double in X« und »in Y«.
Die Anfangsposition eines Sprites kann bei »Start POSITION« ausgewählt werden. Die X-Koordinate kann zwischen 0 und 500 liegen, die Y-Koordinate zwischen 0 und 250. Die linke obere Ecke des Bildschirms entspricht den Koordinaten X: 31, Y: 31. Durch das große Koordinatenfeld (normal sind eigentlich X: 0 bis 319, Y: 0 bis 199) ist es auch möglich, Sprites unter den Rahmen des Bildschirms zu setzen und damit »unsichtbar« zu machen und anschließend auf den sichtbaren Bereich zu bewegen.

11.5 Das »new«-Menü

In diesem Menü können Sie weitere Einstellungsmöglichkeiten aktivieren. Ein Sternchen vor dem Schlagwort einer Funktion besagt, daß diese Funktion eingeschaltet ist.

11.5.1 »sprite«

Initialisiert die grundsätzlichen Einstellungsmöglichkeiten wie Sprite-Farbe, Anfangsposition und Vergrößerung des Sprites.

11.5.2 »velocity«

Die Sprite-Geschwindigkeit in X- und Y-Richtung kann über diese Funktion in Schritten von 8 Pixel pro Sekunde eingestellt werden. Klicken Sie einfach auf die Pfeile neben dem X- und dem Y-Feld, um die Geschwindigkeit nach Ihren Wünschen zu verändern. Möglich sind Geschwindigkeiten von –101 bis 99. Ist ein negativer Wert eingestellt, dann bewegt sich das Sprite nach links (X) oder nach oben (Y), bei positiven Werten bewegt es sich nach rechts (X) oder nach unten (Y).

Bei einer Geschwindigkeit von 0 in beiden Richtungen bleibt das Sprite auf der Anfangsposition stehen.

Bei Aktivierung von »velocity« wird automatisch die Funktion »trail« ausgeschaltet.

11.5.3 »picture«

Durch Anklicken dieser Funktion blenden Sie das Piktogramm »Edit« am unteren Bildschirmrand ein. Mit diesem Piktogramm gelangen Sie in den Editor-Bildschirm, der in Abschnitt 11.6.1 erläutert ist.

11.5.4 »sequence«

Die Möglichkeit, Sprites zu animieren, ist eine der interessantesten Funktionen von GeoBasic. Animation bedeutet, daß ein Sprite sein eigenes Aussehen verändern kann. Damit können Sie beispielsweise einen gezeichneten Menschen über den Bildschirm laufen lassen – mit bewegten Armen und Beinen.

Eine Animation wird dadurch erzeugt, daß mehrere einzelne Bilder in schneller Abfolge hintereinander gezeigt werden, und so eine Bewegung vorgetäuscht wird. Wie in einem Film unterscheidet sich jedes dieser Einzelbilder (genannt Frames) nur in einer Kleinigkeit von seinem Vorgänger. Jedes Einzelbild wird im Editor-Bildschirm (Abschnitt 11.6.1) gezeichnet.

Zuerst müssen Sie die Anzahl der »ANIMATION Frames« (Einzelbilder) einstellen. Maximal sieben solcher Einzelbilder sind möglich. Die hier bestimmte Zahl gibt auch an, wie viele Einzelbilder Sie im Editor-Bildschirm zeichnen müssen.

Das Feld »ANIMATION Rate« (Geschwindigkeit der Animation) sagt aus, in welcher Geschwindigkeit (in 60stel Sekunden) die Einzelbilder hintereinander gezeigt werden sollen. Der Wert muß zwischen 1 und 240 (4 Sekunden) liegen.

Durch Anklicken des Feldes »Continuous ANIMATION« (Endlose Animation) können Sie einstellen, ob die Animation nur ein einziges Mal ablaufen soll und dann nur noch das letzte Einzelbild gezeigt wird oder ob die Animation endlos wiederholt werden soll.

11.5.5 »timeout«

Diese Funktion ermöglicht den automatischen Aufruf einer Routine Ihres Programms, nachdem ein Sprite auf den Bildschirm gebracht wurde.

Im Feld »TIMEOUT Time« (Zeit) stellen Sie die Zeit in 60stel Sekunden ein, die nach dem Einschalten des Sprites noch ablaufen soll, bevor die Routine aufgerufen werden soll. Möglich ist ein Wert zwischen 1 und 240 (4 Sekunden).

Klicken Sie »Stop motion« an, dann wird die Animation und Bewegung des Sprites gleichzeitig mit dem Aufruf der Routine gestoppt und nicht mehr fortgeführt. Ansonsten erfolgt die Animation und die Bewegung des Sprites auch noch nach dem Aufruf der Routine.

Nun müssen Sie GeoBasic noch die erste Zeilennummer oder den Label der Routine im Feld »At TIMEOUT gosub« mitteilen. Die Routine muß mit einem RETURN-Befehl abschließen, um einen korrekten Ablauf des Programms zu gewährleisten. Ist die Routine ausgeführt, dann wird das Programm an der Stelle fortgesetzt, an der es unterbrochen wurde, als die Zeit (TIMEOUT Time) abgelaufen war. »timeout« kann nicht gleichzeitig mit »trail« benutzt werden.

11.5.6 »trail«

Mit der »trail«-Funktion können Sie eine Bildschirmposition bestimmen, zu der sich das Sprite hinbewegen soll. Dabei wird automatisch die X- und Y-Bewegung errechnet, die nötig ist, um den Punkt zu erreichen. »trail« deaktiviert die Menüpunkte »velocity« und »timeout«.

Die X- und Y-Position des gewünschten Zielpunkts stellen Sie in den Feldern »TRAIL to X« und »to Y« mit den Pfeilen ein. Die X-Koordinate darf zwischen 0 und 500 liegen, die Y-Koordinate zwischen 0 und 250. Die obere linke Bildschirmposition ist – wie Sie sich vielleicht erinnern – in diesem Koordinatensystem 31 / 31. So können Sie ein Sprite auch an eine Position außerhalb des sichtbaren Bereichs des Bildschirms bewegen.

Die Geschwindigkeit der Gesamtbewegung können Sie im Feld »Velocity« in Schritten von 8 Pixel pro Sekunde einstellen.

Durch das Feld »Stop motion at trail end« (Bewegung am Zielpunkt beenden) teilen Sie GeoBasic mit, ob das Sprite noch weiter animiert werden soll, wenn der Zielpunkt erreicht ist oder nicht.

Im Feld »At end gosub« können Sie die erste Zeilennummer oder das Label einer Routine angeben, die aufgerufen werden soll, sobald der Zielpunkt erreicht ist. Diese Routine muß nach ihrer Beendigung mit einem RETURN-Befehl abschließen. Dann wird das Programm an der Stelle fortgesetzt, an der es abgebrochen wurde, weil das Sprite den Zielpunkt erreicht hat.

11.6 Das »options«-Menü

11.6.1 »editor screen/attribute screen«

Steht an dieser Stelle »editor screen«, dann gelangen Sie durch Anklicken in den Editor-Bildschirm, wo Sie die Einzelbilder des Sprites in ihrem Aussehen verändern können (Bild 11.6). Durch Anklicken von »attribute screen« gelangen Sie wieder zum Attribut-Bildschirm zurück.

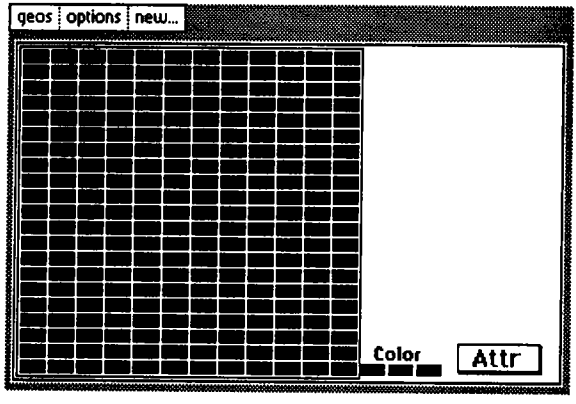


Bild 11.6: Der Editor-Bildschirm

Den Editor-Bildschirm können Sie auch einschalten, indem Sie im Attribut-Bildschirm das Piktogramm »Edit« anklicken. Befindet sich das Piktogramm derzeit nicht auf dem Bildschirm, dann machen Sie es mit »picture« aus dem »new«-Menü sichtbar. Aus dem Editor-Bildschirm schalten Sie durch Anklicken des Piktogramms »Attr« wieder auf den Attribut-Bildschirm zurück.

Der Editor-Bildschirm besteht aus zwei Fenstern, dem Editier-Fenster auf der linken Seite und einem Fenster auf der rechten Seite mit einigen Informationen und Einstellmöglichkeiten zum aktuellen Sprite.

Im Editier-Fenster sehen Sie das aktuelle Sprite in stark vergrößerter Form. Jedesmal, wenn der Mauszeiger in das Editier-Fenster bewegt wird, wird er zu einem Zeichenstift.

Um Punkte zu setzen, klicken Sie einmal über einen nicht gesetzten Punkt des Sprites (der Zeichenstift wird jetzt schwarz). Wenn Sie jetzt den Zeichenstift bewegen, dann werden an allen Stellen über die seine Spitze fährt, Punkte in der aktuellen Farbe gesetzt. Klicken Sie nochmals, um wieder mit dem Zeichnen aufzuhören (der Zeichenstift ist dann blau).

Möchten Sie Punkte löschen, dann klicken Sie einmal auf einem gesetzten Punkt des Sprites (der Zeichenstift wird rot). Alle Punkte, über die die Spitze des Zeichenstifts hinwegfährt, werden nun gelöscht. Klicken Sie nochmals, um das Löschen zu beenden.

Im unteren Bereich des rechten Fensters (links neben dem »Attr«-Piktogramm) befinden sich drei farbige Rechtecke. Diese zeigen die aktuellen Sprite-Farben an. Durch Anklicken eines Rechtecks wählen Sie die jeweilige Farbe als Zeichenfarbe aus. Das Rechteck, dessen Farbe als aktuelle Zeichenfarbe fungiert, ist eingerahmt.

Direkt über den farbigen Rechtecken befindet sich – falls Sie im Attribut-Bildschirm die Animation (»sequence«) eingeschaltet haben – ein Feld mit der Beschriftung »Current FRAME« (Aktuelles Einzelbild). Hier können Sie die Nummer des Einzelbildes auswählen, welches Sie verändern möchten.

Im oberen Teil des rechten Fensters sehen Sie Ihr Sprite in Originalgröße. Eine eventuell aktivierte Animation wird mit der im Attribut-Bildschirm gesetzten Geschwindigkeit angezeigt.

11.6.2 »link sprites«

Wenn die 12 x 21 Pixel eines einzelnen Sprites nicht für Ihr Vorhaben reichen, dann können Sie auch bis zu vier normale Sprites zu einem großen Sprite zusammenfügen. Alle Parameter des großen Sprites (Animation, Geschwindigkeit ...) werden durch ein Haupt-Sprite gesetzt und müssen nicht für jedes Sprite einzeln definiert werden.

Wählen Sie die Funktion »link sprites« aus, so müssen Sie bestimmen, wie viele Sprites zu einem großen Sprite zusammengefügt werden sollen (2, 3 oder 4). Das Sprite, das Sie zuletzt verändert haben, wird zum Haupt-Sprite und die nachfolgenden Sprites werden als Bestandteile des großen Sprites verarbeitet. Haben Sie also beispielsweise zuletzt Sprite 4 verändert und setzen 4 Sprites zu einem zusammen, dann werden die Sprites 4, 5, 6 und 7 verwendet. Haben Sie aber zuletzt Sprite 6 oder 7 editiert, dann können Sie natürlich nicht vier Sprites zusammenlinken, da es kein Sprite 9 und 10 gibt!

Im rechten Fenster des Editor-Bildschirms erscheinen bei zusammengesetzten Sprites noch einige zusätzliche Felder:

In der Mitte des rechten Fensters finden Sie das »Current SPRITE«-Feld (Aktuelles Sprite). In diesem Feld stellen Sie das Teilsprite ein, das Sie bearbeiten möchten. Es erscheint dann im Editier-Fenster und kann verändert werden.

Die Felder »X Off« und »Y Off« bestimmen, wie weit die Teilsprites vom Haupt-Sprite entfernt sind. Es kann immer nur die Entfernung des Sprites verändert werden, das Sie zur Zeit verändern (»Current SPRITE«). Die Entfernung des Haupt-Sprites läßt sich natürlich nicht einstellen! Ein negativer Offset (Abstand) positioniert ein Sprite links vom (X) oder oberhalb des (Y) Hauptsprites, ein positiver Offset rechts vom (X) oder unterhalb des (Y) Hauptsprites.

11.6.3 »paste frame«

Gestalten Sie animierte Sprites, so kann es sehr hilfreich sein, mit dieser Funktion den Inhalt eines anderen Einzelbildes in das aktuelle Einzelbild zu kopieren. So müssen Sie nicht jedes Einzelbild neu zeichnen!

Eine Dialogbox fragt nach dem Aufrufen dieser Funktion nach der Nummer des Einzelbildes, das in das aktuelle Einzelbild kopiert werden soll (Bild 11.7).

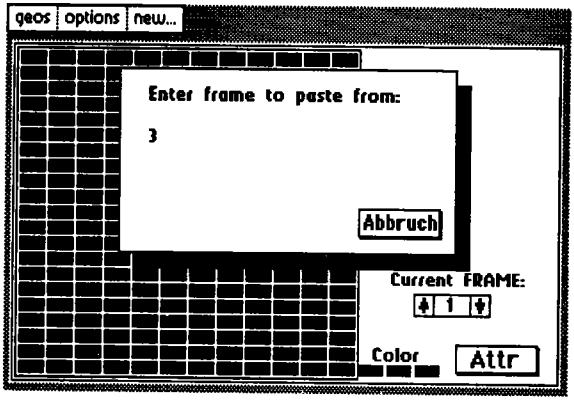


Bild 11.7: Kopieren eines Einzelbildes

Geben Sie die Nummer des Einzelbildes ein und drücken Sie Return. Mit dem »Abbruch«-Piktogramm können Sie diese Funktion auch vorzeitig beenden.

11.6.4 »quit«

Sie können den Sprite-Editor über diese Funktion verlassen. Es erscheint eine Dialogbox mit der Frage »Save changes before quitting?« (Änderungen vor dem Verlassen speichern?) (Bild 11.8).

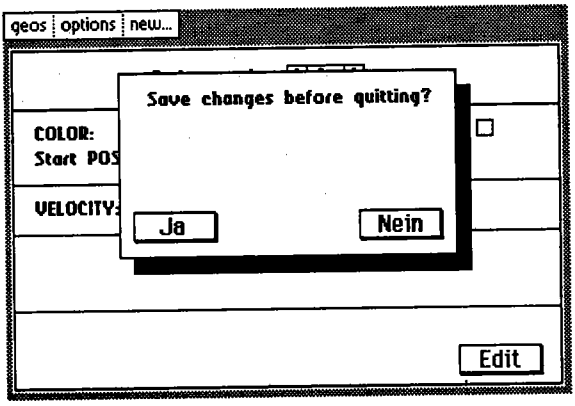


Bild 11.8: Verlassen des Sprite-Editors

Wenn Sie »Ja« anklicken, so wird Ihr Sprite unter dem vorher von Ihnen bestimmten Namen gespeichert. Möchten Sie das Sprite nicht speichern, so klicken Sie auf »Nein«.

11.7 Nutzung der Sprites in eigenen Programmen

Die mit dem Sprite-Editor erstellten Sprites lassen sich mit den in Abschnitt 5.6 beschriebenen Befehlen in eigenen Programmen verwenden.

12. Kapitel
Fehlersuche in GeoBasic-Programmen (Debuggen)

Niemand ist perfekt. Warum sollte es Ihnen also nicht auch passieren, daß Sie in Ihre Programmkreation ein paar kleine logische Fehler eingebaut haben? Und diese logischen Fehler müssen nicht einmal beim ersten Durchlaufen auffallen; oftmals merken Sie erst, daß ein Programm fehlerhaft ist, wenn es sich nach mehrmaligem Ablauf mit einer Fehlermeldung verabschiedet. Solche Fehler zu finden, ist außerdem ziemlich oft eine Sisyphus-Arbeit! Selten stoßen Sie beim ersten Nachschauen schon auf den »Täter«. Um Sie in der Fehlersuche zu unterstützen, wurde ein »Debugger« mit zahlreichen leistungsstarken Funktionen in GeoBasic integriert.

12.1 Starten des Debuggers

Um den Debugger zu starten, können Sie sich des DEBUG-Befehls (Abschnitt 4.3.4) bedienen. Eine weitere Möglichkeit, den Debugger zu starten, ist die Funktion »debug« aus dem »options«-Menü (Abschnitt 3.4.4) im Text-Editor.

12.2 Die Debugger-Dialogbox

Nach dem Starten des Debuggers und immer dann, wenn Sie die [Run Stop]-Taste während des Programmablaufs im Debugger drücken, gelangen Sie in die Debugger-Dialogbox (Bild 12.1).

Hier können Sie Abbruch-Parameter setzen, den Ablauf-Modus des Programms und anzuzeigende Ausdrücke bestimmen.

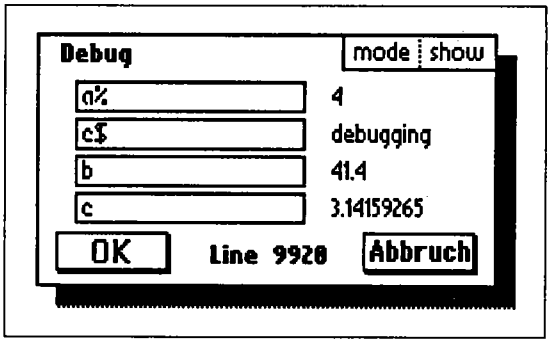


Bild 12.1: Die Debugger-Dialogbox

Mit dem »OK«-Piktogramm können Sie Ihr Programm starten oder (wenn Sie sich bereits mitten im Programm befinden) fortsetzen. Durch Anklicken von »Abbruch« gelangen Sie in den Text-Editor zurück.

Zwischen den beiden Piktogrammen wird die Zeilennummer, in der das Programm angehalten wurde, um die Debugger-Dialogbox zu öffnen, angezeigt (z. B. »Line: 10«). Ist das Programm noch nicht gestartet worden, dann wird hier natürlich keine gültige Zeilennummer angezeigt.

Die linke Seite der Dialogbox zeigt vier Felder, in die Sie, nachdem Sie sie einmal angeklickt haben, bestimmte Werte eingeben können. Diese Werte können beliebige Ausdrücke (z. B. a + 2), Abbruchstellen im Programm (Line breakpoints) oder Ausdrücke, die als Abbruchstellen verwendet werden (Expression breakpoints), sein. Die Art der Werte können Sie mit Hilfe des »display«-Menüs ändern (Abschnitt 12.4).

Voreingestellt ist die Eingabe von anzuzeigenden Ausdrücken. Dies können Sie daran erkennen, daß über den vier Feldern das Wort »Debug« steht. Geben Sie im »Debug«-Modus Ausdrücke mit beliebigen Variablen, Konstanten, Operatoren und Labels in die Felder ein, dann erscheinen die zugehörigen Werte (oder auch Zeichenketten) rechts neben den Feldern. Funktionen (wie SIN, LEFT$ oder LEN) sind nicht in den Debugger-Ausdrücken verwendbar!

Die Ausdrücke unterscheiden sich von den Ausdrücken, die Sie in GeoBasic benutzen, nur durch geänderten logischen Operatoren. Anstelle von »AND« müssen Sie im Debugger das Zeichen »&« eingeben, »ö« steht für »OR«, und »NOT« ersetzen Sie durch »!«.

Wegen des geänderten »OR«-Operators dürfen Sie keinesfalls Variablen eingeben, die ein »ö« enthalten. Der Debugger würde das ansonsten falsch interpretieren.

Nachfolgend finden Sie einige Beispiele für mögliche Ausdrücke im Debugger und deren »Übersetzung« in die GeoBasic-Schreibweise:

Debugger	*GeoBasic*
x	x
x + y	x + y
(2 * x) + y	(2 * x) + y
(x < 10) & (x > 15)	(x < 10) AND (x > 15)
(x < 10) ö (x > 15)	(x < 10) OR (x > 15)
x <> 5	x <> 5
! (x = 5)	NOT (x = 5)
z%(144)	z%(144)
3.141 * x	3.141 * x
x ^ 3	x ^ 3
a$	a$
a$ + b$	a$ + b$

Nicht möglich sind solche Ausdrücke:

SIN(x)
ABS(x + y)
SQR(x)
LEN(a$)
LEFT$(a$)

Finden Sie anstelle des Wortes »Debug« die Wörter »Line breakpoints« in der oberen Dialogbox-Zeile, dann werden in den vier Feldern die Abbruchstellen (Breakpoints) innerhalb des Programms angezeigt.

Das Erreichen einer der Zeilen, deren Nummern (oder deren Labels) Sie in die Felder eingegeben haben, führt im laufenden Programm zum Erscheinen der Debugger-Dialogbox. Sie haben mit diesen Breakpoints die Möglichkeit, ein Programm an einer genau definierten Stelle abzubrechen, um dann wieder im »Debug«-Modus Variablen oder Ausdrücke testen zu können. Geben Sie folgendes Programm im Text-Editor ein und starten Sie den Debugger:

```
10 a = 3
20 b = 2
30 c = 4
40 c = c + b + a
50     REM c hat nach Zeile 40 den Wert 9.
```

Möchten Sie nun in Zeile 40 einen Breakpoint setzen, dann geben Sie im »Line-breakpoints«-Modus die Zahl 40 in eines der vier Felder ein und klicken auf »OK«.

Schalten Sie in den »Expression-breakpoints«-Modus (in der oberen Zeile finden Sie jetzt »Expression breakpoints«, dann können Sie in die Felder Ausdrücke (in der oben beschriebenen Form) eingeben, die als Breakpoints dienen sollen.

Stellt der Debugger im Verlauf des Programms fest, daß einer (oder mehrere) der angegebenen Ausdrücke wahr werden (–1), dann erscheint wieder die Debugger-Dialogbox, und der erste Ausdruck, der wahr geworden ist, wird mit einem Sternchen an der linken Seite versehen (es können natürlich auch die anderen Ausdrücke wahr sein).

Möchten Sie, daß der Debugger (in unserem oben angeführten Beispiel) eingeschaltet wird, wenn c = 9 ist, dann geben Sie im »Expression breakpoints«-Modus in eines der vier Felder ein: c = 9. Nach dem Anklicken von »OK« wird Ihr Programm nach Abarbeitung der Zeile 40 angehalten.

12.3 Das »mode«-Menü

Es gibt drei verschiedene Ablauf-Modi im Debugger.

12.3.1 »run«

Wenn der »run«-Modus durch Anklicken angewählt wird, dann wird ein Programm nach Anklicken von »OK« ganz normal abgearbeitet, und die Debugger-Dialogbox erscheint nur dann, wenn ein Breakpoint oder ein Basic-Fehler gefunden wurde, oder wenn Sie Run Stop drücken.

12.3.2 »step«

Das Programm wird Befehl für Befehl abgearbeitet. Nach jedem Befehl oder bei Erreichen eines Breakpoints (und natürlich auch, wenn Sie Run Stop drücken oder ein Basic-Fehler aufgetreten ist) wird die Debugger-Dialogbox angezeigt.

Die aktuelle Zeilennummer wird dabei zwischen den Piktogrammen »OK« und »Abbruch« angezeigt (Line ...).

12.3.3 »trace«

Diese Funktion bewirkt einen Ablauf des Programms wie im »run«-Modus. Wird aber ein Breakpoint erreicht, dann erscheint die Debugger-Dialogbox nur, um die Ausdrücke, die Sie

vorher im »Debug«-Modus eingegeben haben, mit ihrem Wert anzuzeigen. Die Menüleiste wird nicht angezeigt.

Der »trace«-Modus kann durch Betätigen von [Run Stop] verlassen werden.

12.4 Das »display«-Menü

Hier können Sie die in Abschnitt 12.2 beschriebenen Anzeigearten einstellen.

12.4.1 »values«

Die eingegebenen Werte werden in die Felder gebracht und können verändert werden (»Debug«-Modus). Dieser Modus ist voreingestellt.

12.4.2 »ln brk«

Die Abbruchstellen (Breakpoints) werden angezeigt und sind veränderbar (»Line-breakpoints«-Modus).

12.4.3 »ex brk«

Die Ausdruck-Breakpoints werden gezeigt und können verändert werden (»Expression-breakpoints«-Modus).

12.4.4 »cl brk«

Die Breakpoints werden gelöscht und können neu belegt werden.

12.5 Auftreten eines Basic-Fehlers

Bei Auftreten eines Basic-Fehlers während des Programmablaufs erscheint eine Dialogbox, die »Check #« (Überprüfen #) und die Nummer des Fehlers anzeigt. Durch Klicken auf »OK« gelangen Sie zur Debugger-Dialogbox.

Die Liste der Basic-Fehler im Klartext mit den zugehörigen Nummern siehe Anhang 14.4.

13. Kapitel
Basic-Grabber

Haben Sie bereits unter Commodore-Basic eigene Programme entwickelt und möchten diese mit GeoBasic weiterverwenden, dann müssen Sie nicht erst das Programm unter Commodore-Basic ausdrucken und dann unter GeoBasic wieder abtippen. Für diesen Fall gibt es den Basic-Grabber:

13.1 Starten des Basic-Grabbers

Öffnen Sie unter Desktop eine Ihrer Arbeitsdisketten, die den Basic-Grabber enthält.

Anschließend klicken Sie das Piktogramm mit der Untertitelung »BASIC GRABBER« (Bild 13.1) doppelt an.

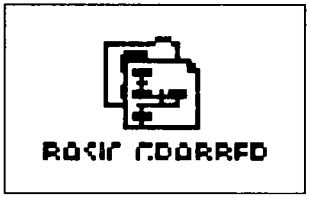


Bild 13.1: Das Basic-Grabber-Piktogramm

13.2 Auswählen einer Commodore-Basic-Datei

Nach dem Starten öffnet sich der Basic-Grabber-Auswahl-Bildschirm (Bild 13.2).

Die Dialogbox in der Mitte des Bildschirms zeigt alle Nicht-GEOS-Dateien der aktuellen Diskette an. Suchen Sie – gegebenenfalls unter Zuhilfenahme der Rollpfeile, wenn nicht alle Namen in die Dialogbox passen, den Namen des zu konvertierenden Commodore-Basic-Programms aus der Auflistung heraus und klicken ihn einmal an (er wird dann mit einem schwarzen Balken versehen). Anschließend klicken Sie auf »Öffnen«.

Mit »Quit« können Sie den Basic-Grabber wieder verlassen und zum Desktop zurückkehren.

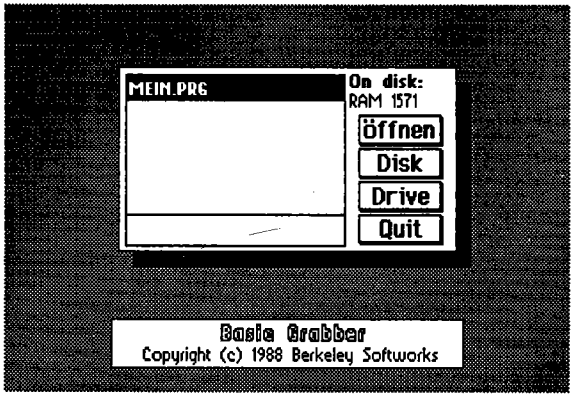


Bild 13.2: Der Basic-Grabber-Auswahl-Bildschirm

Möchten Sie ein Commodore-Basic-Programm von einer Diskette in einem anderen Laufwerk in das GeoBasic-Format konvertieren, so schalten Sie mit »Drive« zwischen den aktiven Laufwerken hin und her.

»Disk« ermöglicht Ihnen das Konvertieren von Commodore-Basic-Programmen einer anderen Diskette, die derzeit in keinem Ihrer Laufwerke liegt. Nach Anklicken werden Sie zum Diskettenwechsel aufgefordert. Legen Sie nun die gewünschte Diskette in das aktuelle Laufwerk ein, und klicken Sie auf »OK«.

13.3 Konvertierung in das GeoBasic-Format

Nachdem Sie das Commodore-Basic-Programm angewählt haben, erscheint eine Dialogbox, die Sie zur Eingabe eines neuen Namens auffordert (Bild 13.3). Den Namen, den Sie jetzt eingeben, trägt das GeoBasic-Programm nach der Konvertierung.

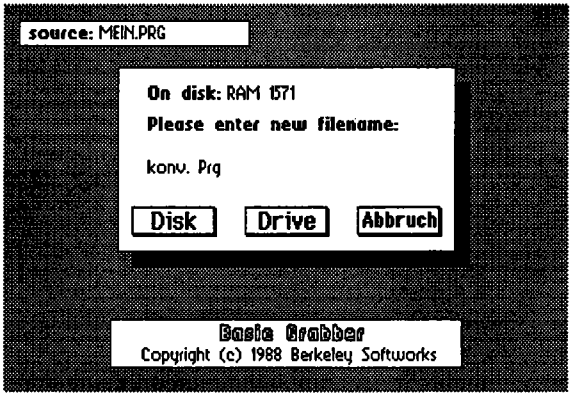


Bild 13.3: Eingabe des Namens für das GeoBasic-Programm

Mit [Del] können Sie einzelne Buchstaben löschen und mit [Return] beenden Sie Ihre Eingabe. Klicken Sie auf »Abbruch«, um zum Auswahl-Bildschirm zurückzukehren.

Möchten Sie, daß das GeoBasic-Programm auf eine Diskette in einem anderen Laufwerk erstellt wird, dann schalten Sie mit »Drive« zwischen den aktiven Laufwerken hin und her.

Soll das GeoBasic-Programm auf eine andere Diskette geschrieben werden, die in keinem der aktiven Laufwerke liegt, dann klicken Sie »Disk« an. Basic-Grabber fordert Sie nun zum Diskettenwechsel auf, nach dessen Beendigung Sie auf »OK« klicken müssen. Anschließend startet Basic-Grabber automatisch die Konvertierung.

Basic-Grabber ändert die Syntax der Commodore-Basic-Befehle und Funktionen – die RND-Funktion ausgenommen – nicht. Allen Befehlen, die GeoBasic nicht bearbeiten kann, stellt er ein REM voraus, gibt aber keine Fehlermeldung aus.

Vergessen Sie nicht nach der Konvertierung das Entfernen der Commodore-Basic spezifischen POKE-Befehle – sie können unter GeoBasic zu einem Absturz führen, da viele Systemadressen geändert wurden!

Nach erfolgreicher Konvertierung erscheint eine entsprechende Meldung in einer Dialogbox (siehe Bild 13.4). Auf der von Ihnen gewünschten Diskette befindet sich jetzt das konvertierte GeoBasic-Programm.

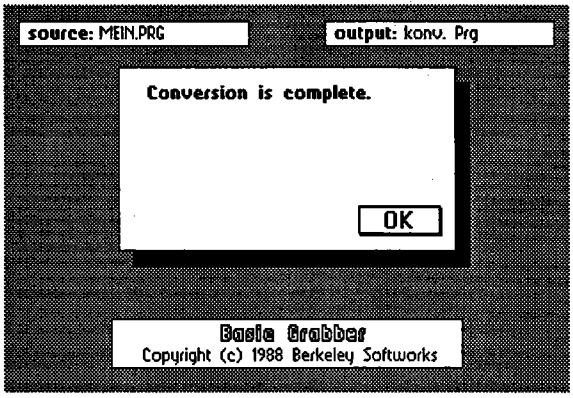


Bild 13.4: Die Konvertierung ist komplett

Durch Klicken auf »OK« gelangen Sie wieder zum Auswahl-Bildschirm zurück. Sind Fehler während der Konvertierung aufgetreten, so werden diese von Basic-Grabber in einer zusätzlichen Dialogbox angezeigt.

14. Kapitel
Anhang

14.1 Glossar

Begriffe, die mit einem »→« gekennzeichnet sind, werden an anderer Stelle im Glossar erläutert.

Akkumulator

Ein Register des Hauptprozessors Ihres C 64 und C 128. Der Akkumulator ist nur für die Programmierung in Maschinensprache und Assembler wichtig.

Anklicken

So wird die Betätigung der linken Maustaste oder des Feuerknopfs am Joystick genannt, wenn der →Mauszeiger über einem → Piktogramm, einem Schlagwort in einem →Menü oder an einer anderen bedeutsamen Stelle des Bildschirms (z. B. in einem Editierfeld) positioniert wurde.

Anweisungen

Anweisungen steuern den Ablauf von →Programmen und sind die eigentlichen Bestandteile von →Programmen. In diesem Buch wird aber anstelle des Worts Anweisung immer das wohl gebräuchlichere Wort →Befehl verwendet, um unnötige Verwirrungen zu unterbinden.

Applikationen

Als Applikationen bezeichnet man die Programme, die unter GEOS ablaufen und vom Desktop aus aufgerufen werden können, z. B. GeoWrite, GeoPaint und →GeoBasic.

Ausdrücke

Als Ausdruck bezeichnet man →Konstanten und →Variablen (→Operanden), die auch mit →Operatoren kombiniert werden können, z. B. 2, a, x$, 4 + r%, 16 / (4 – k), "Hallo " + A$ + "!".

Basic

Abkürzung für »**B**eginner's **A**ll Purpose **S**ymbolic **I**nstruction **C**ode«. Es handelt sich um eine einfach zu erlernende Programmiersprache für Einsteiger in die Welt des Computers.

Basic-Grabber

Eine im →GeoBasic-Programmpaket enthaltene →Applikation zur Umwandlung von Standard-Commodore-→Basic-→Programmen in →GeoBasic-→Programme.

Befehle

Sie dienen dem Verwalten eines →Programms im →Text-Editor (z. B. LIST, RUN, ...) und steuern den Programmablauf (z. B. PRINT, MAINLOOP ...).

Bit

Die kleinste Informationseinheit in Ihrem C 64 und C 128. Ein Bit kann nur zwei Zustände annehmen, 0 (aus) und 1 (ein). Um größere Werte verwalten zu können, sind immer acht Bits zu einem →Byte zusammengefaßt.

Byte

Ein Byte besteht aus acht →Bits. So lassen sich Werte im Bereich von 0 bis 255 darstellen. Möchten Sie noch größere Werte darstellen, dann müssen Sie mehrere Byte zu größeren Einheiten zusammenfassen, wobei zwei Byte beispielsweise einen Wertebereich von 0 bis 65535 haben. Der C 64 hat 65536 (0 bis 65535) Byte.

Cursor

Das Rechteck, an dessen Position im →Text-Editor die eingegebenen Zeichen ausgegeben werden. Anstelle des Rechtecks tritt bei Eingaben, die mit dem INPUT- →Befehl im →Programm entgegengenommen werden, ein vertikaler Balken.

debug

(engl. = entwanzen): Hierbei handelt es sich um die Tätigkeit, eigene →Programme von lästigen Fehlern zu befreien.

Debugger

Der Debugger ist im →GeoBasic-Programmpaket eingebunden und hilft einem Programmierer bei der Fehlersuche.

Dialogbox-Editor

Ein in →GeoBasic enthaltenes →Hilfsprogramm zur Erstellung von →Dialogboxen.

Dialogboxen

Ein rechteckiger Bereich, der sich über den aktuellen Bildschirm legt und vom Programmbenutzer eine bestimmte Entscheidung verlangt. Der Benutzer kann seine Entscheidung durch →Anklicken von Piktogrammen oder aber durch Eingabe von Text und Zahlen an das Programm übergeben. Anschließend verschwindet die Dialogbox wieder vom Bildschirm, wobei der Hintergrund automatisch wiederhergestellt wird.

Eigenständig lauffähige GeoBasic-Programme

Normalerweise sind→GeoBasic-→Programme nur unter →GeoBasic ablauffähig, d. h. →GeoBasic muß im Speicher stehen, wenn Sie Ihr Programm starten wollen. Durch eine besondere Funktion von →GeoBasic kann ein →Programm aber auch eigenständig lauffähig gemacht werden, so daß es auch ohne →GeoBasic gestartet werden kann.

Foto-Scrap

Eine Datei, die einen Ausschnitt aus einem GeoPaint-Bild enthält. Foto-Scraps können von Ihnen im →Grafik-Editor eingeklebt (eingelesen) werden.

Funktionen

Funktionen stellen dem Programm bestimmte Werte (z. B. einen Logarithmus oder den Teil einer →Zeichenkette) zur Verfügung.

GeoBasic

Eine GEOS- →Applikation, die Ihnen zu Programmierzwecken eine Vielzahl von →Basic-→Befehlen zur Verfügung stellt. Durch die Möglichkeit der Nutzung der besonderen GEOS-Bestandteile (→Menüs, →Piktogramme ...) bietet GeoBasic selbst Fortgeschrittenen in der →Basic-Programmierung noch großartige Möglichkeiten.

Grafik-Editor

Ein in →GeoBasic integriertes →Hilfsprogramm zur Erstellung von →Grafiken.

Grafiken

Bilder, die aus gesetzten und gelöschten →Pixels bestehen. Grafiken können entweder mit dem →Grafik-Editor erstellt werden oder aus GeoPaint-Dokumenten per →Foto-Scrap eingeklebt werden.

Hauptschleife

Siehe →Mainloop.

Hilfsprogramme

Programme wie der Foto-Manager, die aus →Applikationen heraus gestartet werden können. In →GeoBasic sind fünf Hilfsprogramme eingebaut, mit denen →Dialogboxen, →Grafiken, →Menüs, →Piktogrammlisten und →Sprites erstellt werden können.

Klicken

Siehe →Anklicken.

Konstanten

Werte, die während des Ablaufs eines Basic-Programms immer gleich bleiben und nicht verändert werden können. Es gibt Ganzzahl-Konstanten, Fließkomma-Konstanten und Zeichenketten-Konstanten.

Labels

Ähneln →Variablen, enthalten aber die Nummer von Zeilen, denen eine besondere Bedeutung zukommt (wie z. B. die erste Zeile einer Unterroutine).

Mainloop

Das Herz des GEOS-Betriebssystems. Die Mainloop steuert und verwaltet →Menüs, →Piktogramme und einige andere wichtige Dinge für die ablaufenden →Applikationen. Als Applikationen gelten auch die eigenen →GeoBasic-→ Programme, nicht nur, wenn diese →eigenständig lauffähig sind.

Mauszeiger

Der kleine Pfeil auf dem Bildschirm. Mit ihm können Sie z. B. Funktionen aus dem →Menü oder auch →Piktogramme anwählen. Die Positionierung des →Cursors im →Text-Editor kann ebenfalls über den Mauszeiger erfolgen.

Menü-Editor

Ein in →GeoBasic integriertes →Hilfsprogramm, mit dem →Menüs erstellt werden können.

Menü

Als Menü bezeichnet man die durch Rechtecke eingegrenzten Schlagworte in der linken oberen Ecke des Bildschirms. Durch →Anklicken eines dieser Schlagworte öffnen Sie ein Abrollmenü, aus dem Sie (wiederum durch →Anklicken) eine Funktion auswählen können.

Operanden

→Konstanten und →Variablen, wenn Sie in →Ausdrücken Verwendung finden.

Operatoren

Die Zeichen, die angeben, was mit den →Operanden in →Ausdrücken geschehen soll (^, *, / , +, -, <, =, >, NOT, AND, OR). Operatoren stehen normalerweise zwischen zwei →Operanden.

Parameter

Angaben, die an einen →Befehl angehängt werden, um ihn genau zu definieren. Parameter können →Konstanten, →Variablen, →Labels aber auch ganze →Ausdrücke sein.

Piktogramme

Kleine →Grafiken, die →angeklickt werden können, um eine bestimmte Funktion aufzurufen (z. B. Löschen des Editierfelds).

Piktogrammlisten-Editor

Ein in →GeoBasic integriertes →Hilfsprogramm zur Zusammenfassung von zu verwendenden →Piktogrammen zu einer Liste.

Pixel

Der Bildschirm des C 64 und C 128 wird unter GEOS in 64.000 einzeln ansprechbare Punkte, genannt Pixels, aufgeteilt.

Programme

Eine Ansammlung von einzelnen →Befehlen, die →GeoBasic ausführen kann.

Prozeß

Eine →Unterroutine, die durch die →Mainloop in bestimmten Zeitabständen automatisch aufgerufen wird.

Run-Time-Bibliothek

Wird an ein →GeoBasic-→Programm angehängt, um es →eigenständig lauffähig zu machen. Die Run-Time-Bibliothek enthält alle →Befehle und →Funktionen, die Sie in →GeoBasic-→Programmen verwenden können.

Schleife

Ein Teil eines →Programms, der mehrmals durchlaufen wird.

Sprite-Editor

Ein in →GeoBasic integriertes →Hilfsprogramm, mit dem Sie →Sprites erstellen können.

Sprites

Kleine bewegliche →Grafiken, die auch mehrfarbig sein dürfen. Bei der Bewegung von Sprites wird der Hintergrund nicht zerstört.

Status-Register

Ein Register des Hauptprozessors Ihres C 64 und C 128. Wichtig ist das Status-Register nur bei der Programmierung in Maschinensprache und Assembler.

Syntax

Der Aufbau eines Befehls, an den Sie sich halten müssen.

Text-Editor

Der Teil von →GeoBasic, in dem Sie ein →Programm eingeben und verändern können.

Unterroutine

Ein Teil Ihres →Programms, der eine bestimmte oft benötigte Funktion enthält.

Variablen

Werte, die sich während des Ablaufs eines →Programms ändern können und sogar vom Benutzer des →Programms beeinflußt werden dürfen.

X-Register

Ein Register des Hauptprozessors Ihres C 64 und C 128. Wichtig ist das X-Register nur bei der Programmierung in Maschinensprache und Assembler.

Y-Register

Ein Register des Hauptprozessors Ihres C 64 und C 128. Wichtig ist das Y-Register nur bei der Programmierung in Maschinensprache und Assembler.

Zeichenkette

Eine Aneinanderreihung von alphanumerischen Zeichen (Buchstaben, Zahlen und Sonderzeichen). Zeichenketten sind an den sie umschließenden Anführungszeichen (z. B. "DM 12,70") erkennbar.

14.2 Übersichten über die Menüs

14.2.1 GeoBasic

geos

GeoBasic info – Informationen zu GeoBasic.

Außerdem erscheinen im »geos«-Menü die Namen aller Hilfsmittel auf der Diskette.

file

close – das aktuelle Programm wird gespeichert und das Hauptmenü erscheint.

update – die letzten Programmänderungen werden auf Diskette gespeichert.

rename – ermöglicht die Änderung des Programmnamens.

print – gibt das Programm auf den Drucker aus.

quit – Sie beenden Ihre Arbeit mit GeoBasic und kehren zum Desktop zurück.

edit

list – das Programm wird auf dem Bildschirm aufgelistet.

sprcol – ermöglicht die Änderung der Sprite-Farbeinstellungen.

options

run – startet das Programm.

debug – aktiviert den Debugger.

resize	– die Größe des Programm- und des Variablenspeichers kann bestimmt werden.
renumber	– Ihr Programm wird nachträglich neu numeriert.
make appl	– macht ein Programm eigenständig lauffähig.

Utilities

menu	– der Menü-Editor wird aktiviert.
dialog	– ruft den Dialogbox-Editor auf.
icon	– der Piktogrammlisten-Editor wird gestartet.
bitmap	– aktiviert den Grafik-Editor.
sprite	– der Sprite-Editor wird aufgerufen.

14.2.2 Dialogbox-Editor

geos

dialog utility info	– Informationen zum Dialogbox-Editor.

options

guides on/off	– Positionierungshilfen für Objekte ein- bzw. ausschalten.
display dialog box	– zeigt die erstellte Dialogbox an.
quit	– Rückkehr zum Text-Editor.

14.2.3 Piktogrammlisten-Editor

geos

icon list utility info	– Informationen zum Piktogrammlisten-Editor.

options

quit	– Rückkehr zum Text-Editor.

14.2.4 Grafik-Editor

geos

bitmap info – Informationen zum Grafik-Editor.

Alle Hilfsmittel der aktuellen Diskette stehen auch im »geos«-Menü zur Auswahl.

options

paste photo scrap – klebt ein Foto-Scrap in die aktuelle Grafik ein.

disk loadable – die aktuelle Grafik wird auf die Diskette ausgelagert.

erase bitmap – löscht die aktuelle Grafik.

quit – Rückkehr zum Text-Editor.

14.2.5 Sprite-Editor

geos

sprite utility info – Informationen zum Sprite-Editor.

options

editor screen/
attribute screen – der jeweilige Bildschirm wird angezeigt.

link sprites – bis zu vier Sprites werden zu einem großen Objekt zusammengefügt.

paste frame – Einzelbild einer Sprite-Animation kopieren.

quit – Rückkehr zum Text-Editor.

new

sprite – Grundeinstellungen initialisieren.

velocity – Sie ändern die Geschwindigkeit.

picture – das Piktogramm »Edit« erscheint am unteren Bildschirmrand.

sequence – aktiviert die Animation eines Sprites.

timeout – ermöglicht den Aufruf einer Programmroutine nach Ablauf einer bestimmten Zeit seit der Aktivierung des Sprites.

trail – das Sprite soll auf direktem Wege an eine bestimmte Bildschirmposition bewegt werden.

14.2.6 Debugger

mode

run	– normaler ununterbrochener Ablauf des Programms und Anzeigen der Debugger-Dialogbox nur nach Auffinden eines Breakpoints.
step	– die Debugger-Dialogbox wird nach jedem ausgeführten Befehl angezeigt.
trace	– wie »run«, die Debugger-Dialogbox wird aber nur für einen kurzen Moment angezeigt.

show

values	– Ausdrücke werden angezeigt und können geändert werden.
ln brk	– zeigt die veränderbaren Zeilen-Breakpoints an.
ex brk	– die Ausdruck-Breakpoints werden angezeigt und können editiert werden.
cl brk	– löscht alle eingegebenen Breakpoints, so daß das Programm ohne Unterbrechung abläuft.

14.3 Befehle und Funktionen im Kurzüberblick

Hier finden Sie alle Befehle und Funktionen in alphabetischer Reihenfolge mit Seitenverweis aufgeführt.

14.3.1 Befehle

14.3.2 Funktionen

14.4 GeoBasic-Fehlermeldungen

Die in Klammern aufgeführten Werte werden vom SYSINFO-Befehl in Parameter 6 übergeben.

Syntax (0)

Sie haben sich beispielsweise in der Schreibweise eines Befehls, einer Funktion oder eines Parameters geirrt. Möglicherweise haben Sie aber auch einen numerischen Parameter angegeben, obwohl eine Zeichenkette nötig gewesen wäre oder umgekehrt. Dieser Fehler tritt ebenfalls auf, wenn Sie einen Befehl im Text-Editor eingeben, der nicht für die Benutzung im Text-Editor zugelassen ist oder wenn Sie zu viele oder zu wenige Parameter angeben.

Out of blocks (1)

Es sind zu wenig Blöcke auf Diskette frei, um fortfahren zu können.

Bad track (2)

Sie benutzen zwischen einem OPEN und CREATE und dem zugehörigen CLOSE Schreib- und Lesebefehle. Zugelassen ist aber immer nur eine dieser beiden Möglichkeiten (entweder Schreiben oder Lesen).

Disk full (3)

Es ist kein Platz mehr auf der Diskette, so daß die Ausführung eines weiteren Schreibzugriffs mit CREATE ... unmöglich ist.

Full directory (4)

Es können maximal 144 einzelne Dateien auf Diskette gespeichert werden. Dieser Fehler tritt auf, wenn Sie versucht haben, eine 145. Datei zu erstellen.

File not found (5)

Die Datei, die geöffnet oder geladen werden soll, ist auf der Diskette nicht vorhanden.

Bad BAM (6)

Die Block-Availability-Map (BAM) der Diskette, auf die Sie zugreifen möchten ist fehlerhaft. Dieser Fehler kann nur durch direkten Zugriff auf die BAM (nicht mit GeoBasic) behoben werden.

Unopened file (7)

Sie haben versucht, einen Lese- oder Schreibzugriff durchzuführen, obwohl keine Datei vorher mit OPEN geöffnet wurde.

Bad Record (8)

Der Record einer VLIR-Datei, auf den Sie zugreifen möchten, existiert noch nicht und muß erst noch mit APPEND oder INSERT erschaffen werden.

Out of records (9)

Es können maximal 127 Records in einer VLIR-Datei belegt werden, weitere INSERT- oder APPEND-Befehle sind erst nach einem DELETE-Befehl möglich.

Buffer overflow (11)

Sie haben versucht, weitere Werte aus einer Datei oder einem Record zu lesen, obwohl in dieser Datei oder diesem Record keine Daten mehr vorhanden sind.

No print driver (12)

Sie haben unter dem Desktop noch keinen Druckertreiber gewählt oder der gewählte Druckertreiber befindet sich nicht auf der GeoBasic-Diskette und Sie versuchen trotzdem, den Drucker zu verwenden.

Device not found (13)

Das angesprochene Gerät (z. B. der Drucker) ist nicht vorhanden oder ist nicht eingeschaltet

Next without for (14)

Beim Ablauf eines Programms stößt GeoBasic auf einen NEXT-Befehl, wobei noch kein FOR-Befehl ausgeführt wurde.

Return without Gosub (15)

GeoBasic ist in Ihrem Programm an einen RETURN-Befehl gestoßen, obwohl die Routine nicht mit einem GOSUB-Befehl aufgerufen wurde.

Out of data (16)

Sie haben versucht, weiter mit READ Daten zu lesen, wobei das letzte DATA-Element des Programms bereits ausgelesen wurde. Dieser Fehler läßt sich eventuell durch Benutzung von RESTORE vermeiden.

Illegal quantity (17)

Ein nicht zugelassener Wert wurde benutzt, z. B. 320 als X-Koordinate in einem LINE-Befehl, der maximal 319 als X-Koordinate verarbeitet.

Overflow (18)

GeoBasic kann eine Berechnung nicht korrekt weiterführen, da das Ergebnis zu hoch oder zu niedrig ist.

Out of memory (19)

Es ist nicht mehr genügend Speicherplatz vorhanden, um eine neue Variable anzulegen oder eine alte neu zu belegen.

Undefined statement (20)

Die Zeilennummer, die als Parameter zu einem Sprungbefehl (GOTO ...) angegeben wurde, existiert nicht.

Bad subscript (21)

Ein Index, den Sie bei einem Zugriff auf eine Feldvariable verwendet haben, ist negativ oder übersteigt den vorher im DIM-Befehl angegebene Größe.

Redimensioned array (22)

Die Feldvariable, die Sie dimensionieren wollten, ist bereits vorher dimensioniert worden.

Division by zero (23)

In einer Berechnung wird eine Zahl durch Null dividiert, was nicht möglich ist. Meistens tritt dieser Fehler auf, wenn Sie in einer Berechnung durch eine Variable teilen lassen, die vorher nicht mit einem Wert belegt wurde.

Type mismatch (24)

Sie haben eine Zeichenkette in einem numerischen Ausdruck oder umgekehrt verwendet, ohne dabei diese mit STR$ oder VAL umzuwandeln.

String too long (25)

Zeichenketten-Ausdrücke dürfen maximal 255 Zeichen lang sein. Dieser Fehler tritt auf, wenn Sie z. B. versucht haben, zwei Zeichenketten mit einer Länge von jeweils 200 Zeichen zu verbinden.

Formula too complex (26)

Ein Ausdruck ist zu komplex, um von GeoBasic ausgeführt werden zu können. Teilen Sie diesen Ausdruck in mehrere kleinere Teilstücke auf, um diesen Fehler zu beseitigen.

Undefined function (28)

Eine Funktion wurde mit FN benutzt, ohne daß diese vorher mit DEF FN definiert wurde.

Redefined label (30)

Ein Label mit dem gleichen Namen existiert bereits an einer anderen Stelle im Programm. Dieser Fehler kann auch auftreten, wenn Sie den Namen eines Labels in einer Programmzeile ändern, ohne die ganze Zeile zu löschen, und anschließend das alte Label an einer anderen Stelle im Programm zu verwenden. GeoBasic erkennt erst, daß Sie ein Label streichen möchten, wenn Sie die Zeile, in der dieses Label steht, löschen. Danach müssen Sie die Zeile mit dem neuen Label nochmals eingeben.

Line too long (31)

Eine Programmzeile ist länger als 240 Zeichen (6 Bildschirmzeilen).

No disk (33)

Im angesprochenen Laufwerk findet GeoBasic keine oder nur eine unformatierte Diskette.

Bad data (35)

Dieser Fehler tritt auf, wenn Sie eine nicht speicherresidente Grafik in einer Dialogbox oder in einer Piktogrammliste verwendet haben.

Write protected (38)

Ihr Schreibzugriff auf die Diskette kann nicht ausgeführt werden, weil diese Diskette schreibgeschützt ist.

Font too big (40)

Der einzulesende Zeichensatz ist zu groß, um noch in den Variablenspeicher zu passen.

Wrong disk (41)

Sie haben eine Diskette ohne Aufforderung durch GeoBasic gewechselt. Legen Sie die alte Diskette zurück in das Laufwerk.

Hinweis: Die Ausgabe dieser Meldung beim unaufgeforderten Wechseln von Disketten kann nicht garantiert werden – es ist ohne weiteres möglich, daß GeoBasic beim ersten Zugriff auf die falsche Diskette abstürzt!

Missing label (42)

Das Label, das Sie in einem Ausdruck oder in einem Menü, einer Dialogbox, einer Piktogrammliste, einer Grafik oder einem Sprite verwendet haben, ist nicht definiert worden.

Labels full (43)

Sie können maximal 127 Labels in einem GeoBasic-Programm nutzen.

Loop not found (44)

GeoBasic hat keinen LOOP- oder UNTIL-Befehl nach einem REPEAT- oder einem WHILE-Befehl gefunden.

Loop without do (45)

Es wurde vor Auffinden eines LOOP oder UNTIL kein REPEAT- oder WHILE-Befehl ausgeführt.

Break (47)

Die Taste RUN STOP wurde betätigt, um das Programm zu stoppen.

Too many processes (48)

Es können maximal acht Prozesse gleichzeitig ablaufen. Sie müssen vor der Definition eines weiteren Prozesses erst einen alten Prozeß wieder deaktivieren.

14.5 Die ASCII-Zeichen

In diesem Anhang finden Sie alle ASCII-Zeichen und ihre zugehörigen Werte bei der Verwendung von CHR$ und ASC.

Sind bestimmte Werte in dieser Tabelle nicht aufgeführt, dann können Sie entweder nicht verwendet werden oder führen zu unkontrollierten Reaktionen des Betriebssystems (z. B. Absturz).

Wert	*Zeichen/Funktion*	*Wert*	*Zeichen/Funktion*
8	Backspace (Cursor um ein Zeichen nach rechts)	40	(
9	Tab (Sprung zum nächsten Tabulator-Stopp)	41	)
10	Linefeed (Cursor an erste Position in aktueller Zeile)	42	*
11	Home (Cursor in die obere linke Bildschirmposition)	43	+
12	Upline (Cursor eine Zeile nach unten)	44	,
13	CR (Cursor an die erste Position der nächsten Zeile)	45	-
14	Unterstreichen ein	46	.
15	Unterstreichen aus	47	/
18	Invers ein	48	0
19	Invers aus	49	1
24	Fettschrift	50	2
25	Kursivschrift	51	3
26	Konturschrift	52	4
27	Normalschrift	53	5
32	Leerzeichen	54	6
33	!	55	7
34	"	56	8
35	#	57	9
36	$	58	:
37	%	59	;
38	&	60	<
39	'	61	=

Wert	*Zeichen/Funktion*	*Wert*	*Zeichen/Funktion*
62	>	96	'
63	?	97	a
64	@	98	b
65	A	99	c
66	B	100	d
67	C	101	e
68	D	102	f
69	E	103	g
70	F	104	h
71	G	105	i
72	H	106	j
73	I	107	k
74	J	108	l
75	K	109	m
76	L	110	n
77	M	111	o
78	N	112	p
79	O	113	q
80	P	114	r
81	Q	115	s
82	R	116	t
83	S	117	u
84	T	118	v
85	U	119	w
86	V	120	x
87	W	121	y
88	X	122	z
89	Y	123	ä
90	Z	124	ö
91	Ä	125	ü
92	Ö	126	ß
93	Ü	127	Delete (letztes Zeichen löschen)
94	^	128	Commodore-Zeichen (C=)
95	_		

Folgende Bytewerte sind möglich:

0 = kein Laufwerk
1 = 1541
2 = 1571
3 = 1581
65 = schattierte 1541
129 = RAM 1541
130 = RAM 1571

33971 (Blinkgeschwindigkeit für Piktogramme)

Beim Anklicken eines Piktogramms oder eines Menüpunkts blinkt dieses kurz auf. Diese Speicherstelle bestimmt die Geschwindigkeit dieses Blinkens. Standardmäßig beträgt der Wert 10.

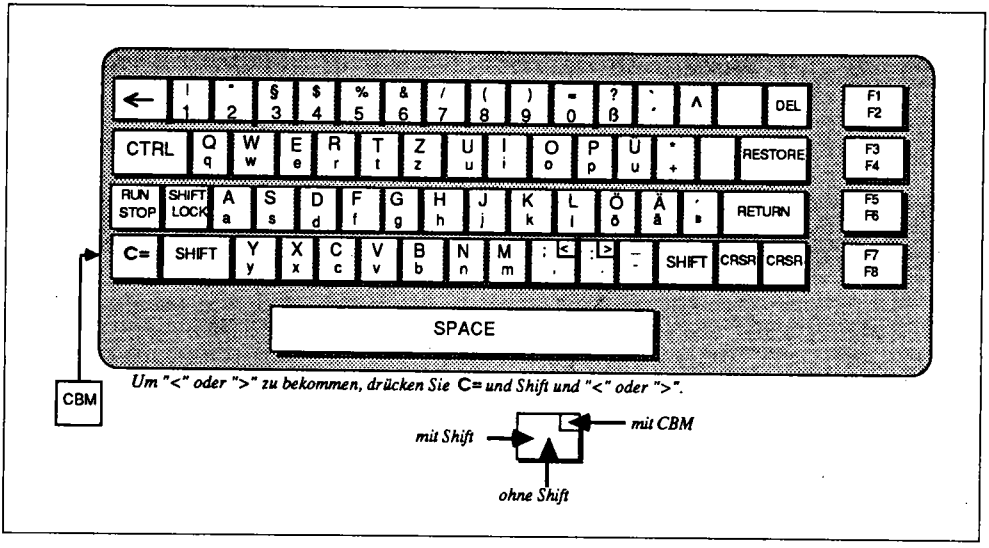
35036 – 35053 (Diskettenname und ID Laufwerk C)

Wie 33822 – 33839, nur für Laufwerk C.

35054 – 35071 (Diskettenname und ID Laufwerk D)

Wie 33822 – 33839, nur für Laufwerk D.

14.7 Die GEOS-Tastaturbelegung



14.8 Die GeoBasic-Diskette

Hier finden Sie eine Übersicht aller Dateien, die sich auf der GeoBasic-Diskette befinden:

Seite A:

Datei	*Dateityp*
GeoBasic	Applikation
parachutes3	Dokument
Sample Appl	Dokument
Samp Appl2	Applikation
Applikation.bsp	Dokument
Basic Grabber	Applikation
Foto Scrap	Systemdatei

Seite B:

Auf Seite B finden Sie die meisten der in diesem Handbuch abgedruckten Listings. Sie sind wie folgt gekennzeichnet:

L_(Kapitel)_(lfd. Nummer)

z. B.

L_4.4.1_1.

Die auf der GeoBasic-Diskette enthaltenen Beispielprogramme dienen der Erklärung wichtiger Features von GeoBasic und können von Ihnen in eigenen Programmen verwendet werden.

Stichwortverzeichnis

Bei Stichwörtern in Großbuchstaben handelt es sich um GeoBasic-Befehle, kursiv eingetragene Stichwörter bezeichnen Menüpunkte.