

LASERTM 2001

HOME COMPUTER

**BASIC
PROGRAMMIERHANDBUCH**



ERSTAUSGABE — 1983

Alle Rechte vorbehalten. Eine Wiedergabe oder Verwendung ohne ausdrückliche Genehmigung des textlichen oder bildlichen Teils dieses Handbuchs in irgendeiner Weise ist untersagt. Die Verwendung der hierin enthaltenen Informationen wird nicht als Patentverbindlichkeit angenommen. Bei der Herstellung dieses Buches wurde mit großer Sorgfalt vorgegangen. Der Herausgeber übernimmt keine Verantwortlichkeit für Fehler oder Unterlassungen. Ferner wird keine Verbindlichkeit übernommen für Beschädigungen, die sich aus der Verwendung der hierin enthaltenen Informationen ergeben.

© Copyright: Video Technology Ltd. — 1983

EINLEITUNG

WENN SIE NICHTS ÜBER COMPUTER WISSEN, HIER GEHT ES LOS!

Dieses Buch ist für alle gedacht, die noch nichts von BASIC wissen. Auch wenn Sie zu denen gehören, die schon einmal einen Anlauf genommen haben - das Buch wird Ihnen verschiedene Probleme deutlicher machen.

BEVOR SIE WEITERLESEN:

- 1) Packen Sie Ihren LASER 2001 aus.
- 2) Schließen Sie ihn an ein Fernsehgerät an.
- 3) Verbinden Sie beide Geräte mit dem Netz.
- 4) Schalten Sie beide Geräte ein.
- 5) Wählen Sie den richtigen Empfangskanal.

Der Bildschirm zeigt nun



BASIC VERSION 1.0
(C) 1983 VTL

Alle Informationen zur Inbetriebnahme finden Sie in dem Begleitheft KURZINFORMATIONEN ZUM LASER 2001 SYSTEM. Haben Sie also Ihren Computer noch nicht in Betrieb, so schließen Sie ihn jetzt an, ansonsten lesen Sie weiter.

Mit diesem Buch finden Sie einen Einstieg in das erweiterte MICROSOFT BASIC. Obwohl es leicht zu erlernen ist, sollten Sie bei Schwierigkeiten an den Spruch: "Der schlaueste Computer ist immer noch dummer als ein Mensch" denken.

INHALT

KAPITEL 1

IHREN COMPUTER KENNEN

- 13 ● IHR ERSTES COMPUTER-PROGRAMM
- 16 ● WAS SIE NICHT UNBEDINGT WISSEN MÜSSEN
- 18 ● WAS SIE AUF JEDEN FALL WISSEN MÜSSEN
- 26 ● THEORIE DES PROGRAMMIERENS
- 30 ● DIE PRAXIS DER PROGRAMMIERUNG
- 34 ● DER VERFLIXTE 'SYNTAX ERROR'
- 36 ● EDITIEREN EINER ANWEISUNGSZEILE
- 40 ● 10 REGELN FÜR BASIC-PROGRAMMIERER

KAPITEL 2

GRAFIK-ANWEISUNGEN IM LASERBASIC

- 43 ● GR
- 44 ● TEXT
- 45 ● COLOR =
- 47 ● PLOT
- 49 ● UNPLOT
- 50 ● CIRCLE
- 51 ● RECT
- 52 ● HOME
- 53 ● VPEEK
- 55 ● VPOKE

KAPITEL 3

LASERBASIC 'DIE LEICHTEN ANWEISUNGEN'

- 58 ● END
- 59 ● NEW
- 60 ● LET
- 61 ● REM
- 62 ● PRINT
- 63 ● INPUT
- 64 ● LIST
- 65 ● STOP
- 66 ● CONT
- 67 ● BREAK
- 68 ● SO GEHT ES EINFACHER

KAPITEL 4

LASERBASIC BEHERRSCHT DAS PROGRAMM FÜR FAULE LEUTE

- 75 ● IF....THEN
- 77 ● FOR....TO....STEP....NEXT
- 79 ● READ....DATA
- 81 ● RESTORE
- 83 ● GOTO
- 85 ● GOSUB....RETURN
- 86 ● ON....GOTO/ON....GOSUB
- 87 ● CLEAR

KAPITEL 5

LASERBASIC BEFIEHLT DAS PROGRAMM FÜR TON UND MUSIK

- 91 ● WIE DER COMPUTER TÖNE ERZEUGT
- 92 ● SOUND
- 94 ● SGEN

KAPITEL 6

DAS SCHRECKLICHE RECHNEN, DAS SIE ZU VER- MEIDEN HOFFTEN

- 97 ● WAS IST EIN OPERATOR
ARITHMETISCHE OPERATOREN, LOGISCHE
OPERATOREN, RELATIONALE OPERATOREN
- 102 ● WAS IST EIN FUNKTION
ABS, ATN, COS, EXP, INT, LOG, RND, SGN, SIN,
SQR, TAN

KAPITEL 7

SCHRECKLICHE UND UNVERSTÄNDLICHE STRINGS

- 107 ● STRINGFUNKTIONEN UND VARIABLEN
- 108 ● LISTE DER STRING FUNKTIONEN
ASC, CHR\$, GET, LEFT\$, LEN\$, MID\$, RIGHT\$, STR\$,
VAL
- 110 ● STRINGVERGLEICHE

KAPITEL 8

LASERBASIC BEHERRSCHT DAS SCHWIERIGE PROGRAMM

- 113 ● DIM
- 114 ● PEEK
- 115 ● POKE
- 116 ● CALL

KAPITEL 9

LASERBASIC ANWEISUNGEN FÜR IHREN KASSETTENREKORDER

- 119 ● CLOAD
- 120 ● CSAVE
- 121 ● CRUN
- 122 ● BLOAD
- 123 ● BSAVE
- 124 ● BRUN
- 125 ● STORE
- 126 ● RECALL

KAPITEL 10

LASERBASIC WEIST DAS PROGRAMM FÜR DEN DRUCKER AN

- 129 ● LLIST
- 130 ● LPRINT

KAPITEL 11

ANHANG

- 133 ● FEHLERMELDUNGEN
- 137 ● ZEICHEN-CODE TABELLE

KAPITEL 1

IHREN COMPUTER KENNEN

- IHR ERSTES COMPUTER-PROGRAMM
- WAS SIE NICHT UNBEDINGT WISSEN MÜSSEN
- WAS SIE AUF JEDEN FALL WISSEN MÜSSEN
- THEORIE DES PROGRAMMIERENS
- DIE PRAXIS DER PROGRAMMIERUNG
- DER VERFLIXTE 'SYNTAX ERROR'
- EDITIEREN EINER ANWEISUNGSZEILE
- 10 REGELN FÜR BASIC-PROGRAMMIERER

IHR ERSTES COMPUTER-PROGRAMM!

Lassen Sie uns jetzt zur Sache kommen! Alles ist richtig angeschlossen, eingeschaltet, und der Bildschirm zeigt nun:



BASIC VERSION 1.0
(C) 1983 VTL

Tippen Sie jetzt das folgende Programm in den Computer ein. Lassen Sie kein Komma aus und vergessen Sie die Zahl am Anfang jeder Zeile nicht! Fügen Sie auch nichts hinzu!

Nur das Wort<RETURN> wird nicht geschrieben, sondern es wird die mit "RETURN" bezeichnete Taste der LASER 2001 - Tastatur betätigt. Das blinkende, kleine Quadrat wird damit zum Beginn der nächsten Zeile gestellt.

FERTIG? DANN SCHREIBEN SIE JETZT!

```
10 GR <RETURN>
20 COLOR=1, 3<RETURN>
30 CIRCLE (75, 50), 20<RETURN>
40 CIRCLE (175, 50), 20<RETURN>
50 PLOT 70, 145 TO 75, 150<RETURN>
60 PLOT 180, 145 TO 175, 150<RETURN>
70 FOR X=75 TO 175 STEP 1<RETURN>
80 PLOT X, 150<RETURN>
90 NEXT X<RETURN>
100 CIRCLE (85, 60), 5<RETURN>
110 CIRCLE (165, 60), 5<RETURN>
120 END<RETURN>
```

Das war es schon! Sie haben gerade ein Programm geschrieben! So schwer war es doch gar nicht? Starten Sie jetzt das Programm und lassen Sie uns sehen, was passiert. Sollten Sie Fehler gemacht haben, werden sie sich jetzt zeigen. Fertig?

Schreiben Sie:

RUN<RETURN>

Wenn Sie nun vom Bildschirm ein fröhlich grinsendes Gesicht auf grünem Hintergrund ansieht, haben Sie das Programm ohne Fehler eingegeben.

Das Gesicht verschwindet, wenn Sie nun die Taste<RESET> betätigen, und Ihr vorher geschriebenes Programm wird in grüner Schrift vor schwarzem Hintergrund wieder sichtbar. Sie können unter CSAVE in diesem Buch nachlesen, wie das Programm mit dem Kassettenrecorder abgespeichert wird.

Um nun weiter zu machen, schreiben Sie entweder NEW <RETURN> , oder Sie schalten den LASER 2001 einmal aus und wieder ein.

Wenn Sie das fröhliche Gesicht nicht gesehen haben, war die Eingabe des Programms nicht fehlerfrei. Drücken Sie die Taste <RESET> und schreiben dann LIST <RETURN> . Vergleichen Sie das Programm auf dem Bildschirm mit dieser Vorlage. Irgendwo wird ein Eingabefehler stecken. Das Wort <RETURN> wird nicht auf dem Bildschirm erscheinen. Sie haben es nicht geschrieben, es steht in der Liste nur als Hinweis auf das Betätigen der<RETURN>- Taste.

Sollten Sie keinen Fehler entdecken, schalten Sie den LASER aus und wieder ein und beginnen noch einmal von vorne.

Sie sollten, nachdem Sie das Gesicht ausreichen bewundert haben, weiterlesen und herausfinden, wie es entsteht.

WAS SIE NICHT UNBEDINGT WISSEN MÜSSEN

Auf dieser Seite finden Sie ein paar zwar grundsätzlich wichtige Informationen, die Ihnen aber zum Erlernen von BASIC nicht unbedingt hilfreich sind. Sie sind jedoch geeignet, Familie oder Freunde mit Ihrem neuen Wissen über Computer zu beeindrucken.

WAS IST EIN COMPUTER?

Es besteht kein Grund, zu umfangreichen Erklärungen auszuholen. Ein Computer - das ist erst mal der kleine, flache Kasten mit der Tastatur hier vor Ihnen.

WAS IST BASIC?

BASIC ist die Abkürzung der Bezeichnung BEGINNERS ALL-PURPOSE SYMBOLIC INSTRUCTION CODE. Die Sprache wurde 1964 am Dartmouth College mit dem Ziel entwickelt, programmieren zu können, ohne an binäre Zahlensysteme oder Maschinencode denken zu müssen.

WAS IST MS-BASIC?

BASIC wurde die am häufigsten benutzte Programmiersprache. Es existiert eine Vielzahl von BASIC-Versionen. MS-BASIC ist von der auf Software - Entwicklungen spezialisierten Firma MicroSoft entwickelt worden und hat sich weltweit als Programmiersprache für Microcomputer durchgesetzt. Wenn Sie also MS-BASIC mit dem LASER 2001 lernen, werden Sie auch viele andere Computer programmieren können.

WENN SIE MS-BASIC SCHON KENNEN UND KÖNNEN . .

Das für den LASER 2001 benutzte MS-BASIC wurde um einige Kommandos für Grafik- und Tonerzeugung von VIDEO TECHNOLOGY erweitert.

Wenn Sie also Bescheid wissen, blättern Sie einfach zur Seite weiter, Sie finden dort eine Liste der erweiterten Anweisungen.

WAS SIE AUF JEDEN FALL WISSEN MÜSSEN!

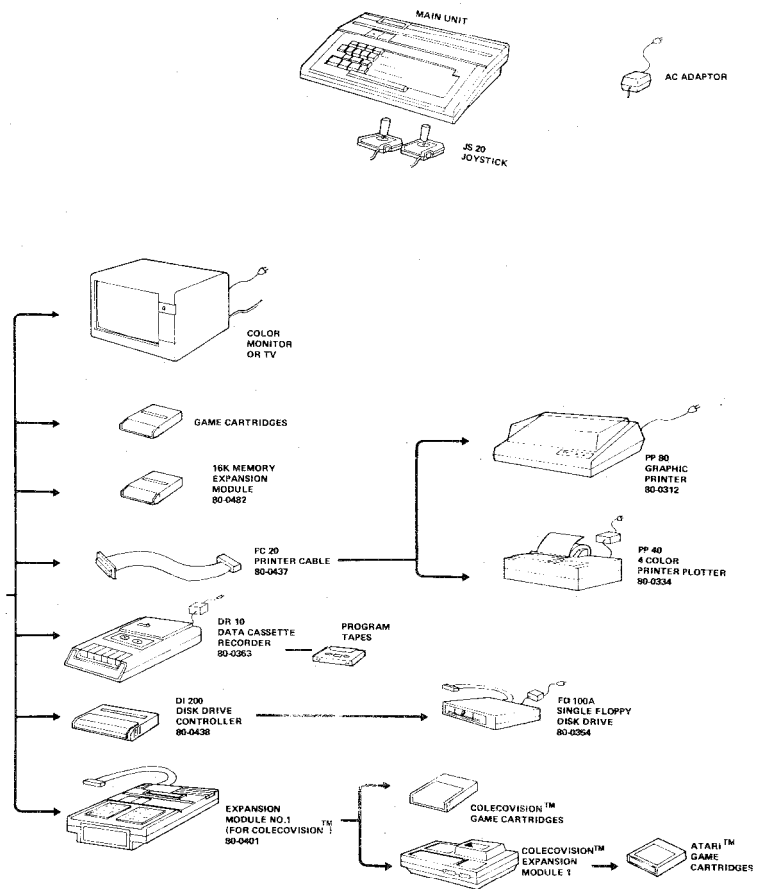
Alles, was in diesem Kapitel besprochen wird, muß auf jeden Fall von Ihnen beherrscht werden. Fangen Sie jetzt nicht an, gleich große Programme zu schreiben! Falls Sie es versuchen: Es wird nicht funktionieren, und der Weg in die Frustration wäre vorgezeichnet. Sie werden die Lust am Programmieren verlieren und niemals wissen, was für ein erstklassiger Programmierer Sie hätten werden können.

DIE TASTATUR, DER SCHLÜSSEL ZUM UNIVERSIUM!

Die Tastatur Ihres LASER 2001 ist einfach, entspricht ergonomischen Erfordernissen und ist leicht zu bedienen. Jedoch sind einige Regeln zu erlernen. Nur sie öffnen den Weg zum Programmieren in BASIC. Wenn Sie diese Regeln beherrschen, können sie alles mit Ihrem Computer machen.

WARUM SIE KEIN SYSTEM-DIAGRAMM BRAUCHEN.

Wenn Sie sich erinnern, waren wir uns einig, daß Sie Ihren Computer betriebsklar machen und anfangen. Das letzte, was Sie also brauchen, wäre ein Diagramm - richtig?



DREI FUNKTIONEN MIT EINER TASTE

Ihr LASER 2001 hat 49 Tasten in verschiedenen Größen. Für alle ist eine AUTO-REPEAT-Funktion vorgesehen. Das bedeutet, daß die Ausgabe des Zeichens auf den Bildschirm so lange automatisch wiederholt wird, wie Sie die Taste drücken. Fast alle der 49 Tasten sind mit drei Funktionen belegt:

Jede Taste ist mit einer großen Ziffer, einem Buchstaben oder einem Wort bezeichnet. Wenn Sie jetzt eine Taste niederdrücken, wird das entsprechende Zeichen auf dem Bildschirm erscheinen oder die mit dem Wort bezeichnete Tätigkeit ausgeführt. Wird z.B. die Taste mit der Ziffer '1' gedrückt, so wird die '1' auch auf dem Bildschirm erscheinen.

DIE SHIFT-TASTE: 16 Tasten, die gesamte obere Tastenreihe und vier andere, sind mit einem zusätzlichen roten Symbol versehen, sind eine der beiden SHIFT-Taste und die entsprechende Symbol-Taste gleichzeitig zu betätigen. Z.B.: Wird die SHIFT-Taste und die Taste mit der '1' gleichzeitig gedrückt, so wird auf dem Bildschirm das Symbol '!' erscheinen.

Wird die SHIFT-Taste gleichzeitig mit einer Buchstaben-Taste gedrückt, so wird der Buchstabe in Kleinschrift auf dem Bildschirm erscheinen. Werden Taste 'A' und SHIFT gedrückt, wird das 'a' ausgegeben.

Alle Kleinbuchstaben dürfen zum Schreiben von BASIC-Programmen nicht benutzt werden. Der LASER 2001 erkennt BASIC-Anweisungen nur in Großschrift. Um Fehler zur Laufzeit des Programmes zu vermeiden, sollten Sie sich an diese Regel halten.

Die SHIFT-Tasten sind leicht zu erkennen: Es steht das Wort SHIFT auf der Taste, sie sind links und rechts auf der unteren Tastenreihe angeordnet.

DIE CTRL-TASTE: Mit dieser Taste wird der dritte Funktionsbereich der Tastatur eingeschaltet. Mit einem Tastendruck werden die kompletten Schlüsselworte des MS-BASIC eingegeben. Mühsame Schreibarbeit wird so eingespart.

Die BASIC-Worte sind den Tasten zugeordnet. An der Taste '1' finden Sie zum Beispiel das BASIC-Wort 'CSAVE'.

Um dieses BASIC-Wort nun einzugeben, müssen wie bei 'SHIFT' die Tasten 'CTRL' und die Wort-Taste gleichzeitig gedrückt werden. Um den Bildschirm zu löschen, drücken Sie also gleichzeitig die CTRL-Taste und die Taste 'A' für das Wort 'HOME', danach die Taste 'RETURN'.

Alle diese Worte an den Tasten können Sie selbstverständlich auch mit den Buchstabentasten der Tastatur ausschreiben. Die Eingabe mit der CTRL-Taste spart Zeit und verhindert Schreibfehler.

CTRL ist die Abkürzung für CONTROL. Sie finden diese Taste in der dritten Reihe links der Tastatur.

WEITERE TASTEN MIT BESONDEREN FUNKTIONEN

DIE RETURN-TASTE: In der zweiten Reihe rechts der LASER 2001 -Tastatur finden Sie die RETURN-Taste. Die Taste wird zur Eingabe der von Ihnen geschriebenen Anweisungen oder Kommandos benutzt. Zum Beispiel wird mit CTRL-A das Kommando 'HOME' auf den Bildschirm geschrieben. Die Ausführung jedoch wird erst durch Drücken der Taste RETURN veranlaßt.

Die Eingabe einer Programmzeile wird ebenfalls erst durch die RETURN-Taste vorgenommen. Nach RETURN schreiben Sie immer in der nächsten Zeile des Bildschirms. Sollten Sie am Ende einer BASIC-Zeile die RETURN-Taste vergessen, werden zwei Zeilen zu einer zusammengefaßt. Sie müssen dann noch einmal geschrieben werden.

RETURN steht als Kurzwort für die Bezeichnung CARRIAGE RETURN. Es ist dies eine Funktion aus der Schreibmaschinenteknik und bedeutet: WAGEN RÜCKLAUF. Eine andere Abkürzung hierfür ist 'CR'. Sie finden diese Bezeichnung in vielen BASIC-Fachbüchern. Andere Computer, um die Nähe zur Schreibmaschinenteknik zu vermeiden, verwenden statt dessen die Bezeichnung 'ENTER'. Lassen Sie sich aber davon nicht verwirren, alle diese Namen und Abkürzungen bedeuten ganz einfach 'RETURN'.

DIE SPACE-TASTE: Die nicht zu übersehende große, lange Taste unter der Tastatur ist die SPACE-Taste. Sie erzeugt den Leerraum zwischen den Worten. Benutzen Sie diese Taste nicht, wird Ihr Text schwer zu lesen sein. Ein Leerzeichen wird auch mit SHIFT SPACE auf den Bildschirm gegeben.

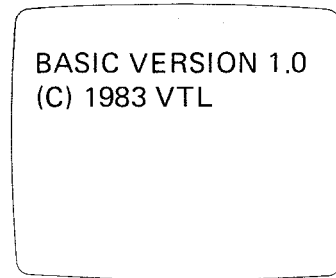
DIE RESET-TASTE: Diese Taste werden Sie sicherlich häufig benutzen. Wird bei Fehlern im Programm der Computer blockiert, so können Sie ohne Programmverlust die Tastenfunktionen wieder aktivieren.

Die RESET-Taste finden Sie über der Tastatur neben der Kontroll-LED. Suchen Sie nach einer großen, quadratischen Taste, und Sie finden sie bestimmt!

DAS BLINKENDE KLEINE QUADRAT AUF DEM BILDSCHIRM

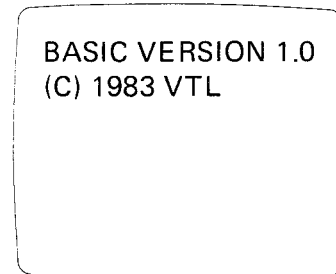
Sie haben eben die Tastatur ausprobiert, und der Bildschirm ist jetzt gefüllt mit allen möglichen Worten und Zeichen, Um diesen unmöglichen Zustand zu beenden, schalten Sie den LASER 2001 einfach aus und wieder ein. (Das Ein- und Ausschalten des TV-Gerätes löscht nicht den Bildschirm. Nach dem Einschalten haben Sie den gleichen Unsinn auf dem Schirm!)

Wenn Sie den Bildschirm gelöscht haben, zeigt er wieder:



Das kleine, blinkende Quadrat wird CURSOR genannt. Der Name hat keinen Bezug zu seiner Funktion, lernen Sie ihn einfach!

Die einzige Aufgabe des CURSOR ist, Ihnen zu zeigen, wo auf dem Bildschirm das nächste Zeichen oder Wort erscheinen wird. Wenn Sie jetzt die Taste 'A' betätigen, erscheint das 'A' an der Stelle des CURSOR, der CURSOR rückt eine Stelle weiter. Der Bildschirm zeigt also jetzt:



DER CURSOR WIRD BEWEGT

Sie bewegen den CURSOR nach rechts mit der Taste '→' und nach links mit der Taste '←'. Für einen CURSOR-Schritt nach oben drücken Sie gleichzeitig SHIFT und '←', die Bewegung nach unten wird mit SHIFT und '→' erreicht. Mit diesen vier Funktionen wird die Ausgabe des mit den Tasten erzeugten Zeichens oder Wort an jeden beliebigen Punkt des Bildschirms dirigiert.

Üben Sie jetzt. Wir werden warten!

DER CURSOR ALS PROMPT

Als PROMT wird die Aufforderung zur Eingabe bezeichnet. Immer, wenn Sie den CURSOR blinken sehen, ist der Computer bereit, Programmzeilen, Kommandos oder Daten hereinzunehmen. Eine Eingabe ohne blinkenden CURSOR ist nicht möglich.

THEORIE DES PROGRAMMIERENS

Denken Sie immer daran, daß ein computer erst einmal grundsätzlich dumm ist. Auch Ihr LASER 2001. Er ist zwar besser als manch anderer Computer, aber ein zweijähriges Kind macht selbst einem Großcomputer etwas vor: Es lernt täglich aus Erfahrung und Erziehung. Ein Computer nicht, er ist nur so schlau, wie sein Programmierer. Ihm muß alles gesagt werden.

DER DUMME BUTLER

Stellen Sie sich vor, Sie hätten einen sehr dummen Butler und Sie müßten ihm jede Tätigkeit detailliert erklären. Der Gang zum Briefkasten, um die Zeitung zu holen, würde dann so dem Butler erklärt werden müssen:

Schritt 1: Tue keinen Schritt, bevor ich 'LOS' sage.

Schritt 2: Gehe zur Haustür.

Schritt 3: Öffne die Tür.

Schritt 4: Geh raus.

Schritt 5: Schließe die Tür.

Schritt 6: Gehe zum Briefkasten.

Schritt 7: Öffne den Briefkasten.

Schritt 8: Nehme die Zeitung.

Schritt 9: Schließe den Briefkasten.

Schritt 10: Drehe dich um.

Schritt 11: Gehe zurück zum Haus.

Schritt 12: Öffne die Tür.

Schritt 13: Geh rein.

Schritt 14: Schließe die Tür.

Schritt 15: Gehe weiter zum Wohnzimmer.

Schritt 16: Gebe mir die Zeitung.

Schritt 17: Hau ab, bis ich dich wieder rufe,

Schritt 18: 'LOS'

Diese Anweisungsliste ist ein Programm. Schritt 1 ist eine Vorsichtsmaßnahme, weil Ihr dummer Butler sonst sofort zur Tür rennt und dort auf weitere Anweisungen warten würde. Schritt 18 ist genauso wichtig, weil der dumme Butler sonst vor Ihnen stehen bleibt und Sie weiter ansieht.

Sehr sorgfältig müssen Sie auch die Reihenfolge der Schritte planen. Wenn Sie dem Butler befehlen, die Tür zu öffnen, dann zu schließen, wird er mit dem nächsten Schritt durch die Tür laufen.

Ein Computer aber ist noch dummer als unser Butler. Wenn Sie diese Anweisungen einem Computer geben, müssen Sie ihm auch noch erklären, wie er laufen muß. Sie würden bis zum Briefkasten "rechten Fuß vor", "linken Fuß vor" rufen müssen. Sie würden zum Öffnen der Tür die Anweisungen "Hand heben", "Griff anfassen", "Griff drücken" usw. geben müssen. Jeder Gang zum Briefkasten müßte immer wieder neu von diesen Anweisungen begleitet werden, da der Computer alles wieder vergessen hat, wenn sein Programm neu gestartet wird.

Der Gang zum Briefkasten muß also für den Computer in Tausenden von Einzelschritten programmiert werden. Ein falscher Schritt, und die Zeitung landet in Brasilien.

WICHTIGE REGELN ZUR PROGRAMMIERSTELLUNG:

- 1) Planen Sie vor der Eingabe jeden Schritt!
- 2) Achten Sie auf die richtige Reihenfolge der Schritte!

DIE PRAXIS DER PROGRAMMIERUNG

Es folgen noch mehr von diesen theoretischen, langweiligen Erklärungen, die einen Mann der Praxis nur hemmen. Überschlagen Sie aber trotzdem die folgenden Seiten nicht, es wird sich später rächen!

WIE WIRD EINE PROGRAMMANWEISUNG EINGEGEBEN?

Programmanweisungen in MS-BASIC werden zeilenweise eingegeben. Jede Zeile beginnt mit der Schrittnummer (Zeilennummer). Geben Sie keine Zeilennummer vor, wird der Computer wie der dumme Butler die Anweisung sofort ausführen. Erinnern Sie sich: Die Schrittnummer stand vor der Anweisung.

Ein typisches Programm wird also so aussehen:

```
10 TUE DIES  
20 TUE JENES  
30 TUE SONSTWAS
```

WARUM ABSTAND ZWISCHEN DEN ZEILENNUMMERN?

Grundsätzlich startet man die Numerierung der Zeilen mit der 10 und nummeriert dann in 10er-Schritten weiter. Natürlich können Sie auch in direkter Reihenfolge die Nummern vergeben (1, 2, 3, . . .). Wenn Sie jedoch so vorgehen, ist es später nicht einfach, noch irgenwelche vergessenen Anweisungen einzufügen. Ganze Abschnitte des

Programms müssen dann neu geschrieben werden. Es ist also erst einmal richtig, ein Programm so aufzubauen:

```
100 TUE DIES  
200 TUE JENES  
300 TUE SONSTWAS
```

In diesem Beispiel ist zwischen den zeilennummern ausreichend Raum zum Einfügen größerer Programmblöcke. Ihr LASER 2001 verarbeitet Anweisungszeilen mit Nummern bis 63999. Es spielt Keine Rolle, welche Nummern die Zeilen tragen. Der Computer verarbeitet die Zeilen in der Reihenfolge der Nummern, mit der niedrigsten beginnend, bis zum Ende.

63999. Es spielt keine Rolle, welche Nummern die Zeilen in der Reihenfolge der Nummern, mit der niedrigsten bis zum Ende.

Außerdem, wenn Sie Zeilen ab 100 nummeriert haben, können Sie ohne weiteres später Zeilen mit niedrigeren Nummern eingeben. Der Computer ordnet Sie bei der Eingabe richtig vor oder zwischen die schon geschriebenen Anweisungen ein.

DIE EINGABE EINER PROGRAMMZEILE

Die Eingabeeiner auf den Bildschirm geschriebenen Anweisungszeile wird durch Drücken der Taste RETURN vorgenommen. Der Computer übernimmt die Zeile und ordnet sie entsprechend der Zeilennummer in seinen Arbeitsspeicher ein.

Wird die Betätigung am Ende der Zeile vergessen, verbindet der Computer mehrere Zeilen miteinander, und die Ausführung des Programmes wird mit einer Fehlermeldung abgebrochen. Die Zeilen müssen dann noch einmal geschrieben werden.

Unser Kurzprogramm sollte jetzt so aussehen:

```
100 TUE DIES <RETURN>
200 TUE DAS <RETURN>
300 TUE SONSTWAS <RETURN>
```

Das RETURN-Tastenkommando wird nicht auf dem Bildschirm angezeigt. Vergessen Sie es aber auf keinen Fall!

ES MUSS AM ENDE JEDER ANWEISUNGSZEILE DAS <RETURN>-TASTEN-KOMMANDO GEGEBEN WERDEN!

DAS STARTEN DES PROGRAMMES MIT <RUN>

Unabhängig davon, wieviel Anweisungszeilen Sie schon eingegeben haben, der Computer wird nichts tun, außer sie in seinem Speicher abzulegen. Er wartet auf Ihr Kommando zum Start des Programms. Sie geben ihm dieses Kommando mit dem geschriebenen Wort RUN. Vergessen Sie hier nicht, die Ausführung durch Drücken der RETURN-Taste zu veranlassen. Dieses Kommando soll den sofortigen Start des Programms bewirken. Sie dürfen daher keine Zeilennummer vor das Wort RUN schreiben!

Wollen Sie also das Programm starten, muß unser Beispielprogramm jetzt so aussehen:

```
100 TUE DIES <RETURN>
200 TUE DAS <RETURN>
200 TUE SONSTWAS RETURN
RUN <RETURN>
```

GRAMMATIK UND ZEICHENSETZUNG

Sicher haben Sie schon Leute getroffen, die penibel auf richtige Schreibweise und Zeichensetzung achten. Sie wissen, wie frustrierend die Korrektur Ihrer Arbeiten durch Lehrer, Vorgesetzte oder Bürokraten sein kann. Aber warten Sie ab, bis Ihr Computer die von Ihnen geschriebenen Anweisungen zensiert.

Ihr Computer wird schon verrückt, wenn nur ein Komma falsch gesetzt ist oder fehlt. Auch mit einem Schreibfehler können Sie ihn nicht mehr bewegen, an Ihrem Programm zu arbeiten. Er wird sich erst zur Arbeit bewegen lassen, wenn Sie jeden Fehler korrigiert haben.

Die einzige Möglichkeit, dieser Besserwisserei zu entgehen, ist tatsächlich alles richtig zu schreiben und die Regeln der Sprache BASIC zu lernen.

DER VERFLIXTE 'SYNTAX ERROR'

Ihr LASER 2001 ist in der Lage, auf viele Ihrer Fehler durch Hinweise auf dem Bildschirm zu reagieren. Ärgern Sie sich nicht darüber. Viele andere Computer geben diese Fehlermeldungen gar nicht oder nur in verschlüsselter Form.

In vielen Fällen wird der Computer sie nicht nur auf die Art des Fehlers hinweisen, sondern Ihnen auch noch mitteilen, in welcher Anweisungszeile der Fehler aufgetreten ist. Sie brauchen dann nur noch das Kommando LIST xx geben, und die fehlerhafte Zeile erscheint auf dem Bildschirm. <xx> steht für die Nummer der Anweisungszeile. Alles, was Sie dann noch zu tun haben, ist die Zeile zu untersuchen und den Fehler zu verbessern. Hier werden wir uns mit der Fehlermeldung 'SYNTAX ERROR' beschäftigen.

SYNTAX ERROR

Wenn Sie diese Fehlermeldung noch nicht auf dem Bildschirm gesehen haben, trösten Sie sich, sie kommt noch! Sie werden diesen Hinweis noch öfter, als Ihnen lieb ist, zu sehen bekommen, es sei denn, Sie sind ein Spitzenprogrammierer. Ihr LASER 2001 ist trotz allem noch ein höflicher Computer. Wenn er sagt: SYNTAX ERROR, dann meint er: SIE MACHEN SCHON WIEDER EINEN FEHLER, MANSCH!

Wenn diese Fehlermeldung auf dem Bildschirm erscheint, dann haben Sie ein Wort falsch geschrieben oder irgendwelche Zeichen falsch gesetzt. Jede BASIC-Anweisung hat ihre eigene Schreibregel, und wenn Sie sich daran halten, wird der Computer sie auch nicht beanstanden.

Wie verbessern Sie nun eine falsch geschriebene Anweisung? Der Computer wird also melden: SYNTAX ERROR IN 80, was heißt, in Zeile 80 wurde ein Fehler festgestellt. (Es kann natürlich jede Zeilennummer sein, aber in diesem Beispiel haben wir Zeile 80 gewählt.) Wenn Sie diese Meldung sehen, schreiben Sie einfach 'LIST 80 <RETURN>' und die fehlerhafte Zeile erscheint auf dem Bildschirm. Prüfen Sie nun die Schreibweise und die Zeichensetzung. Finden Sie keinen Fehler, so schreiben Sie diese Zeile noch einmal mit Leerzeichen zwischen den Worten und Werten. Wird nach RUN diese Zeile wieder vom Computer beanstandet, so trinken Sie am besten Kaffee. Danach wird Ihnen Ihr Fehler bestimmt auffallen.

EDITIEREN EINER ANWEISUNGSZEILE

Der LASER 2001 gibt Ihnen die Möglichkeit, auf dem Bildschirm nach Belieben Anweisungen zu ergänzen, sie zu verbessern, zu ändern oder zu löschen. Sie editieren das Programm! Lassen Sie uns das an dem Programm, welches das fröhliche Gesicht erzeugt, demonstrieren.

```
10 GR RETURN
20 COLOR=1,3 RETURN
30 CIRCLE (75,50),20 RETURN
40 CIRCLE (175,50),20 RETURN
50 PLOT 70,145 TO 75,150 RETURN
60 PLOT 180,145 TO 175,150 RETURN
70 FOR X=75 TO 175 STEP 1<RETURN>
80 PLOT X,150 RETRUN
90 NEXT X RETURN
100 CIRCLE (85,60),5 RETURN
110 CIRCLE (165,60),50 RETURN
120 END RETURN
```

Um eine Zeile nun zu verändern, holen Sie sie sich mit dem LIST-Kommando auf den Bildschirm. Da manche BASIC-Programme seitenlang sind, sollten Sie sich schon die Anweisung gezielt auf den Bildschirm holen. Um in unserem Beispiel die Hintergrundfarbe zu ändern, schreiben Sie also nicht LIST, sondern LIST 20. Schreiben Sie:

```
LIST 20 <RETURN>
```

Der bildschirm zeigt dann:

LIST 20

20 COLOR=1, 3

ZEICHEN IN EINE ZEILE EINFÜGEN

Um die Hintergrundfarbe von Grün zu purpur zu ändern, muß die Anweisung COLOR=1, 3 zu COLOR=1, 13 geändert werden. Zwischen das Komma und die '3' muß eine ',1' eingefügt werden.

Benutzen sie die Tasten SHIFT und '←' gleichzeitig, um den CURSOR nach oben auf die Zeile 20 zu setzen, und bringen Sie ihn dann mit der Taste '→' genau über die '3'. Drücken Sie dann die Tasten CTRL und 'P' gleichzeitig. Damit wird die '3' eine Stelle nach rechts verschoben und zwischen dem komma und der '3' entsteht ein Leerraum, in dem der CURSOR blinkt. Dort hinein schreiben Sie nun die '1' und geben dann <RETURN>. Die geänderte Zeile wird nun in den Speicher übernommen. Mit LIST 20 <RETURN> können Sie sich davon überzeugen.

LIST 20

20 COLOR=1, 13

LÖSCHEN IN EINER ZEILE

Doch was machen wir, wenn wir die Hintergrundfarbe zu Rot verändern wollen? Rot hat den Farbcode 5. Wir müssen nun die '1' zur '5' ändern und die '3' löschen.

Bringen Sie also den CURSOR auf die '1' und schreiben dann in diese Position die '5'. Auf dem Bildschirm sehen Sie dann.



```
LIST 20
```

```
20 COLOR=1, 53
```

Nach dieser Änderung blinkt der CURSOR über der '3'. Er ist vom Computer in diese Position gesetzt worden. Mit der gleichzeitigen Betätigung von CTRL und der Taste 'O' wird die '3' gelöscht und Ihr Bildschirm zeigt nun:



```
LIST 20
```

```
20 COLOR=1, 5
```


So einfach ist das!

Mit der Tastenkombination 'CTRL' und 'P' schaffen Sie also in der Zeile Leerraum zum Einfügen weiterer Zeichen.

Bezeichnung: INSERT

Mit der Tastenkombination 'CTRL' und 'O' werden Zeichen gelöscht. (Bezeichnung: RUBOUT)

10 REGELN FÜR BASIC-PROGRAMMIERER

- 1) Schreiben Sie nur mit Großbuchstaben!
- 2) Beginnen Sie jede Anweisungszeile mit einer Nummer!
- 3) Beenden Sie jede Anweisungszeile mit <RETURN>!
- 4) Ordnen Sie die Anweisungen in der Reihenfolge, in der sie ausgeführt werden sollen!
- 5) Schreiben Sie keine Anweisungsnummer vor
RUN / LIST / NEW / CONT!
- 6) Machen Sie keine Rechtschreibfehler!
- 7) Machen Sie keine Fehler bei der Zeichensetzung!
- 8) Schlagen Sie nicht Ihren Computer mit einem Hammer!
- 9) Behalten Sie dieses Buch immer in Griffweite!
- 10) Geben Sie niemals auf!

KAPITEL 2

GRAFIK-ANWEISUNGEN IM LASERBASIC

- GR
- TEXT
- COLOR=
- PLOT
- UNPLOT
- CIRCLE
- RECT
- HOME
- VPEEK
- VPOKE

WARUM GRAFIK ZUERST?

Die meisten Handbücher verbannen das Grafikkapitel irgendwo an das Ende zwischen die Kapitel über hexadezimale Zahlen und Maschinensprache. Dort gehört es auch hin, wenn es nicht so kinderleicht zu programmieren ist wie im LASER BASIC.

Das BASIC des LASER 2001 gestattet es, komplexe Grafiken auch ohne Führerschein zu programmieren. Statt zur Erstellung der Grafik Sinus-, Cosinus- und logarithmische Funktionen zu formulieren, werden einfache, verständliche Kommandos benutzt.

GR

WAS BEDEUTET SIE?

Diese Anweisung schaltet den Bildschirm in die Grafik-Betriebsart. Bevor irgendeine Grafik oder Illustration auf den Bildschirm gegeben wird, muß diese Umschaltung vorgenommen werden.

Während des Grafikbetriebes ist die PRINT-Anweisung wirkungslos.

SCHREIBWEISE: 10 GR

WAS KANN MAN DAMIT MACHEN?

Sie können Zeichnungen, Illustrationen, Diagramme, graphische Darstellungen, bewegte Figuren und Spiele in dieser Betriebsart erzeugen. Aber bevor Sie nicht das GR-Kommando gegeben haben, wird nichts davon auf dem Bildschirm erscheinen.

TEXT

WAS BEDEUTET SIE?

Diese Anweisung schaltet vom Grafikbetrieb wieder in die Textausgabe zurück. Die Grafikdarstellung auf dem Bildschirm wird durch die vorher erzeugten Texte ersetzt.

Durch abwechselndes Geben der Kommandos GR und TEXT kann blitzschnell zwischen vorbereiteten Texten und Grafiken umgeschaltet werden.

SCHREIBWEISE: 10 TEXT

WAS KÖNNEN SIE DAMIT ANFANGEN?

Eine Grafik wird gezeigt, und mit der Anweisung TEXT werden anschließend wieder Texte auf den Bildschirm gegeben oder Daten angefordert.

COLOR =

WAS BEDEUTET SIE?

Diese Anweisung schaltet den Bildhintergrund auf eine gewählte Farbe und definiert für den Vordergrund (die Schriftzeichen) eine weitere Farbe.

DER FARBCODE:

1	Schwarz
2	Grün
3	Hellgrün
4	Dunkelblau
5	Blau
6	Hellrot
7	Magenta
8	Rot
9	Dunkelrot
10	Gelb
11	Hellgelb
12	Dunkelgrün
13	Purpur
14	Grau
15	Weiß

SCHREIBWEISE: 10 COLOR=1, 3

(Die erste Ziffer beschreibt den Vordergrund, die zweite den Hintergrund.)

WAS KÖNNEN SIE DAMIT ANFANGEN?

Mit COLOR=I, J werden im TEXT-Betrieb Mitteilungen und Daten farbig auf einen farbigen Hintergrund ausgegeben. Im Grafik-Betrieb lassen sich farbige Grafiken und Zeichnungen in jeder Farbkombination erstellen. Aber, . . . Sie können immer nur mit zwei Farben arbeiten. Es ist nicht möglich, auf dem Bildschirm mit mehreren Farben gleichzeitig zu arbeiten!

PLOT

WAS BEDEUTET SIE?

Im Grafik-Betrieb wird der Computer mit dieser Anweisung einen Punkt auf dem Bildschirm in der definierten Vordergrundfarbe setzen. Der Ort des Punktes wird mit den nach PLOT folgenden Werten, X für die horizontale Achse und Y für die vertikale Achse, beschrieben.

Der Bildschirm bildet ein Gitter aus setzbaren Punkten (PIXEL): 255 in der Horizontalen und 192 in der Vertikalen. Die linke, obere Ecke ist mit den Koordinaten 0, 0, die rechte, untere Ecke mit 255. 191 beschrieben.

In Verbindung mit der Anweisung TO wird aus dem PLOT-Kommando ein leistungsfähiger Befehl zur Erstellung von Grafiken. PLOT X,Y TO X,Y zieht auf dem Bildschirm eine Gerade vom ersten zum zweiten Koordinatenpaar.

Die PLOT-Anweisung kann nur im Grafik-Betrieb angewandt werden.

WIE SELZEN SIE DIESE ANWEISUNG NUN EIN?

```
10 GR
20 COLOR=1, 3
30 PLOT 100, 100 TO 150, 100
40 REM ZEILE 30 ERZEUGT EINE GERADE
50 PLOT 150, 100 TO 125, 50 TO 100, 100
60 REM ZEILE 50 ERGAENZT DIE GERADE ZU EINEM
   DREIECK
70 END
```

Anmerkung: Die Koordinaten werden durch ein Komma getrennt. Koordinatenpaare werden zum Zeichnen von Geraden durch die Anweisung 'TO' getrennt. Sie können so viele Koordinatenpaare in einer PLOT-Anweisung aneinanderreihen, wie sie in zwei Bildschirmzeilen unterbringen können.

Alles, was Sie sich vorstellen können, kann auch mit dem PLOT-Kommando gezeichnet werden. Wenn Sie zum Beispiel dickere Linien zeichnen wollen:

```
5 GR
7 COLOR 1,3
10 FOR X=50 TO 55 STEP 1
20 PLOT X, 100 TO X, 180
30 NEXT X
40 END
```

UNPLOT

WAS BEDEUTET SIE?

Mit dieser Anweisung löscht der Computer Punkte und Geraden der Grafik entsprechend den gegebenen Koordinaten. Es gelten die selben Regeln wie für die PLOT-Anweisung.

SCHREIBWEISE: 10 UNPLOT X1,Y1 TO X2, Y2

Sie können UNPLOT . . . TOTO . . . TO . . . geben, bis die Zeichnung gelöscht ist.

WAS LÄßT SICH DAMIT NUN ANFANGEN?

Mit UNPLOT werden Sie nicht nur Fehler korrigieren, sondern auch bewegte Grafiken erstellen. Figuren werden mit PLOT / UNPLOT auf dem Bildschirm erscheinen und wieder verschwinden.

CIRCLE

WAS BEDEUTET SIE?

Mit dieser Anweisung erzeugen Sie einen Kreis um den Punkt X, Y mit dem Radius R. Beachten Sie, daß der Radius der halbe Durchmesser des Kreises ist! Wenn Sie also einen Radius von 20 bestimmen, wird der Kreis einen Durchmesser von 40 Punkten haben.

Vielfach lassen sich die gewünschten Koordinaten berechnen, anstatt sie mühsam zu bestimmen. Lassen Sie das doch von Ihrem Computer machen: Wenn Sie zum Beispiel einen Kreis in den Mittelpunkt des Gitters von 256 Punkten waagrecht und 192 Punkten senkrecht zeichnen wollen, so kann das Programm wie folgt aussehen:

```
10 GR
20 COLOR=1,3
30 X=256/2
40 Y=192/2
50 CIRCLE (X,Y), 10
60 END
```

WAS KÖNNEN SIE NUN DAMIT MACHEN?

Kreise zeichnen! !
Was sonst.

RECT

WAS BEDEUTET SIE?

Der Computer wird angewiesen, ein Rechteck zu zeichnen. Sie brauchen ihm nur noch die Koordinaten der linken, oberen Ecke und der rechten, unteren Ecke mitteilen.

WIE BENUTZEN SIE DIESE ANWEISUNG?

```
10 GR
20 COLOR=1, 3
30 RECT (30, 30), (90, 90)
40 END
```

Wie bei allen Grafik-Anweisungen, muß auch hier die X-Koordinate (Waagerecht) und dann die Y-Koordinate (senkrecht) angegeben werden. Und vergessen Sie nicht: Keine Klammer, kein Komma dürfen fehlen!

HOME

WAS BEDEUTET SIE?

Mit dieser Anweisung löschen Sie den Bildschirm. Der CURSOR wird auf die linke, obere Ecke des Bildschirms gestellt. Der im Textspeicher vorhandene Programtext wird durch diese Anweisung nicht beeinflußt. HOME können Sie nur im TEXT-Betrieb geben.

DIE SCHREIBWEISE: 50 HOME

VPEEK

WAS ES BEDEUTET

Dies ist wie die PEEK-Anweisung, ausser dass die PEEK-Anweisung Daten vom Benutzer RAM und VPEEK Daten vom Bildschirm RAM überträgt.

WIE SIE ES SCHREIBEN

10 PRINT PEEK (A)

wobei A die Adresse ist, die Sie einlesen möchten. Die Adresse sollte sich auf die Video RAM Tabelle beziehen.

VIDEO RAM TABELLE

0	Das Muster in GR-Mode speichern
6114	Das sich bewegende Objekt in der GR-Mode speichern
8192	Den Farbcode in der GR-Mode speichern
14236	Den Text im Text-mode speichern
15360	Die Adresse des Musters in GR-mode speichern
16128	Die Adresse des sich bewegenden Objektes speichern
16384	

VPOKE

WAS ES BEDEUTET

Diese Anweisung ist wie die POKE - Anweisung, ausser dass die POKE-Anweisung Daten zum Benutzer-RAM überträgt und VPOKE die Daten zum Bildschirm-RAM.

WIE ES GESCHRIEBEN WIRD

10 POKE A, B

wobei A die Stelle ist, die Sie übertragen möchten und B die Daten, die Sie speichern möchten. Die Adresse sollte sich auf die Video RAM Tabelle beziehen.

KAPITEL 3

LASERBASIC *** DIE LEICHTEN ANWEISUNGEN ***

- END
- NEW
- LET
- REM
- PRINT
- INPUT
- LIST
- STOP
- CONT
- BREAK

WARUM 'DIE LEICHTEN ANWEISUNGEN'?

Zur Anwendung der Kommandos in diesem Abschnitt benötigen Sie keine Mathematikkenntnisse. Diesen Computer haben Sie doch auch deshalb gekauft, weil er mathematische Probleme für Sie lösen kann. Auf jeden Fall können Sie mit diesen Anweisungen ohne solche Kenntnisse arbeiten. Das ist es, was an den Anweisungen in diesem Kapitel leicht ist!

END

WAS BEDEUTET SIE?

Computer sind so dumm, daß man ihnen sagen muß, wann ein Programm zu Ende ist. Sie merken es sonst gar nicht. Also, wenn sie ein Programm beenden, schreiben Sie als letzte Anweisungszeile . . . mit einer Zeilennummer. . . . die END-Anweisung.

SCHREIBWEISE: 1000 END

WAS KÖNNEN SIE MIT DIESEM KOMMANDO ERREICHEN?

Sie teilen dem Computer mit, daß ein Programm hier zu Ende ist. So können Sie mehrere Programme im Speicher haben, und der Computer kann sie trennen.

NEW

WAS BEDEUTET SIE?

Mit der NEW-Anweisung löschen Sie ein im Speicher befindliches BASIC-Programm. Wenn Sie ein neues Programm schreiben wollen, ist es notwendig, das alte Programm zu löschen. Die Anweisungen des alten Programms würden sich mit den neuen Zeilen untrennbar vermischen.

Sollten Sie die NEW-Anweisung geben, so vergewissern Sie sich, daß das alte Programm auf die Kassette geschrieben ist oder nicht gebraucht wird.

DIE SCHREIBWEISE: NEW RETURN

(Eine Zeilennummer ist nicht notwendig!)

WAS KÖNNEN SIE NUN MIT DIESER ANWEISUNG ANFANGEN?

Nun, wie schon gesagt, ein nicht mehr benötigtes Programm löschen.

LET

WAS BEDEUTET SIE?

Mit der Anweisung LET wird zweierlei bewirkt: Ein für den BASIC-Programmierer stets wieder auffindbarer Speicher wird angelegt, und Zahlen, Zeichen oder die Ergebnisse mathematischer Operationen werden in ihn eingespeichert.

WIE SCHREIBEN SIE DIESE ANWEISUNG?

```
10 LET X=5
```

WAS BEWIRKT DIESE ANWEISUNG GENAU?

Sie ist vor allem keine algebraische Formel, wie die Schreibweise $X=5$ suggerieren würde, sondern der Computer wird angewiesen, einen Speicherbereich mit 'Namen' X anzulegen, oder falls schon vorhanden, zu öffnen und dorthinein die Zahl '5' zu speichern. Mit weiteren LET-Anweisungen kann der Speicherinhalt dann immer wieder modifiziert werden:
50 LET X=500.

REM

WAS KÖNNEN SIE MIT IHR ANFANGEN?

Bemerkungen in Ihre Programmliste schreiben, die der dumme Computer nicht wörtlich nimmt und versucht, auszuführen. REM ist die Abkürzung von REMARK, Anmerkung. REM weist also den Computer an, die in der Anweisungszeile folgenden Texte zu ignorieren.

DIE SCHREIBWEISE: 10 REM TESTPROGRAMM

UND WAS KÖNNEN SIE NUN GENAU MIT DIESER ANWEISUNG TUN?

Sie können und Sie sollten die einzelnen Abschnitte Ihres Programms damit versehen und in diesen REM-Zeilen die Funktion der folgenden Zeilen knapp beschreiben. So haben Sie auch nach längerer Zeit noch die Übersicht, was diese verwirrende Anweisungsliste überhaupt bewirkt.

PRINT

WAS BEWIRKT SIE?

Sie bewirkt die Ausgabe von Texten und Daten auf den Bildschirm. Sie ermöglicht es dem 'BUTLER', auf Fragen zu antworten. Mit PRINT werden im Programm von beliebiger Stelle her Texte auf den Bildschirm gebracht.

DIE SCHREIBWEISE:

```
10 PRINT "ICH HASSE COMPUTER"
```

(Zwischen Anführungszeichen stehende Texte werden ausgegeben. Zwischen Anführungszeichen kann auch Kleinschreibung zur Ausgabe auf den Bildschirm in das Programm aufgenommen werden.)

```
10 LET X=5 «RETURN»  
20 LET Y=5 «RETURN»  
30 PRINT X+Y «RETURN»  
40 END «RETURN»  
RUN «RETURN»
```

(Das vorstehende Programm zeigt einen Weg, den Computer zu einer Antwort auf das Problem 5+5 zu bewegen!)

WAS KANN GRUNDSÄTZLICH MIT DEM PRINT-KOMMANDO ANGEFANGEN WERDEN?

Sie können überall auf den Bildschirm Texte in Groß- und Kleinschrift ausgeben oder mit Text bestimmte Daten von der Tastatur anfordern. Die Ergebnisse der Berechnungen werden als Zahlen mit PRINT ausgegeben.

INPUT

WAS BEDEUTET SIE?

Diese Anweisung veranlaßt den Computer, das Programm zu stoppen und auf die Eingabe von Daten zu warten. Es können mit einer INPUT-Anweisung bis zu 5 Daten von der Tastatur her eingegeben werden. Diese Daten sind entweder Zahlen oder Texte. Texteingaben werden in der Input-Anweisung besonders gekennzeichnet. Wird der Versuch gemacht, Texte einzugeben, wenn der Computer Zahlen erwartet, wird vom Computer die erneute Eingabe mit der Aufforderung: 'REENTER' gefordert.

DIE SCHREIBWEISE:

50 INPUT A

(Ein numerischer Wert wird für den Speicher 'A' angefordert)

50 INPUT A, B, C, D, E

(Fünf numerische Werte werden angefordert)

50 INPUT A\$

(Hier wird zur Eingabe in einen Text Speicher aufgefordert)

WIE WENDEN SIE NUN DIESE ANWEISUNG AN?

Setzen Sie sie überall dort ein, wo eine interaktive Verbindung zwischen Computer und Mensch im Programm gebraucht wird.

LIST

WAS BEWIRKT SIE?

Sie veranlaßt den Computer, Teile des Programms oder das ganze Programm auf den Bildschirm zu bringen. Bearbeitungen, Einfügungen oder Löschungen durch Sie werden so möglich.

In langen Programmen werden Sie die LIST-Funktion brauchen, um den Überblick zu erhalten.

WIE GEBEN SIE DIE KOMMANDOS?

LIST ␣RETURN␣ *(zeigt das gesamte Programm)*

LIST 10,100 ␣RETURN␣ *(listet die Zeilen 10 bis 100)*

LIST 50, ␣RETURN␣ *(listet ab Zeile 50 bis zum Ende)*

LIST , 50 ␣RETURN␣ *(listet vom Anfang bis Zeile 50)*

LIST 50 ␣RETURN␣ *(listet nur Zeile 50)*

WIE KÖNNEN SIE ES ANWENDEN?

Wenn Sie Programme von der Kassette untersuchen wollen, holen Sie das Programm mit LIST A, B blockweise auf den Bildschirm. Dies ist eine gute Möglichkeit, um Tricks der Ton- und Grafikprogrammierung zu lernen. Mit LIST bringen Sie Programmteile zum Editieren auf den Bildschirm.

LIST kann außerdem benutzt werden, um sicher zu gehen, daß keine ungewünschten Programmzeilen im Computer vorhanden sind.

STOP

WAS SIE BEDEUTET:

Sie bedeutet das, was sie besagt: STOP für den Programmlauf, bis von der Tastatur das Kommando 'CONT' den Programm-
lauf fortsetzt.

DIE SCHREIBWEISE:

```
10 LET X=5  
20 LET Y=23  
30 PRINT X+Y  
40 STOP
```

```
50 PRINT 2X+2Y  
RUN
```

```
BREAK IN 35
```

DER EINSATZ:

Das STOP-Kommando setzen Sie zur Fehlersuche in pro-
grammen ein. Es wird an kritischen Stellen in des Programm
eingefügt und erlaubt es dann, zur Laufzeit mit dem PRINT-
Kommando die entsprechenden Speicher zu untersuchen.

Der Abbruch des Programms wird mit 'BREAK IN 35'
gemeldet. Das Programm kann mit CONT «RETURN» fort-
gesetzt werden.

CONT

WAS SIE BEDEUTET:

CONT ist die Abkürzung von CONTINUE, fortsetzen. Sie weist den Computer an, das Programm an der Stelle fortzusetzen, an der es unterbrochen wurde.

DIE SCHREIBWEISE: CONT «RETURN»

UND WIE SETZEN SIE ES EIN?

Mit CONT starten Sie das Programm, wenn Sie es mit STOP oder BREAK unterbrochen haben, Es wird dann mit den bereits erarbeiteten Daten an der Stelle des Unterbruchs fortgesetzt.

BREAK CTRL/C

WOZU DIENT ES?

Ist der Computer auf Grund falscher Anweisungen nicht mehr in der Lage, das Programm zu beenden, so wird ein Abbruch mit dem BREAK-Kommando erzwungen.

WIE GEBEN SIE ES?

Drücken Sie die Tasten CTRL und 'C' gleichzeitig!

WAS KÖNNEN SIE NUN DAMIT MACHEN?

Ein laufendes Programm kann unterbrochen werden. Eine Rückkehr aus endlosen Schleifen wird möglich.

```
1000 PRINT "1234"  
2000 GOTO 1000  
RUN     RETURN
```

Aus dieser Endlosschleife ist eine Rückkehr zur Tastatur nur mit dem BREAK-Kommando möglich. Probieren Sie es!

Die Rückkehr wird dann mit der Meldung: BREAK IN 1000 oder BREAK IN 2000 angezeigt. Sie sollten dann die Anweisungen korrigieren.

SO GEHT ES EINFACHER

Dieses Buch sollte Ihnen die korrekte BASIC-Programmierung vermitteln. Es tut es auch! Aber wir wollen Ihnen auch einen Weg zum einfachen, schnellen und effektiven Programmieren zeigen.

Es gibt eine Reihe von Möglichkeiten im BASIC, außerhalb der korrekten Regeln zu arbeiten. Sie müssen sich nun entscheiden, ob Sie den konservativen Weg einschlagen wollen oder ob Sie mit Abkürzungen schneller arbeiten möchten. Entscheiden Sie sich für die erste Möglichkeit, dann überspringen Sie dieses Kapitel, sonst lesen Sie weiter.

Aber erst, wenn Sie dieses moralische Problem gelöst haben!

SIE BRAUCHEN NICHT MIT NEW STARTEN!

Ein Programm braucht nicht mit 10 NEW starten. Es genügt, von der Tastatur NEW zu geben und mit LIST nachzusehen, ob der Programmspeicher gelöscht ist.

DAS PROGRAMM MUß NICHT MIT END ABGESCHLOSSEN WERDEN!

Hat der Computer alle Anweisungen abgearbeitet, so beendet er selbständig das Programm. Die Anweisung 1000 END als letzte Anweisung im Programm kann entfallen.

Ausnahme: Sie haben mehrere Programme im Speicher, die Sie mit RUN 10, RUN 200 usw. starten wollen. Dann muß jedes Programm mit END abgeschlossen werden.

**AUCH AUF DIE ANWEISUNG LET KANN VERZICHTET
WERDEN.**

Anstelle von 10 LET X=5 können Sie auch 10 X=5 in die Anweisungsliste schreiben. Der Computer wird die Anweisung exakt ausführen.

Auch die PRINT-Anweisung läßt sich einsparen!

Statt der PRINT-Anweisung kann ein '?' geschrieben werden Schreiben Sie: 10 ? A «RETURN» und geben dann LIST «RETURN» . Der Bildschirm wird jetzt 10 PRINT A zeigen.

MEHR ABKÜRZUNGEN!

DIE REM-ANWEISUNG-NOTWENDIG ODER NICHT?

Einerseits ist sie zur sauberen Dokumentation des Programmes unentbehrlich, andererseits braucht sie gerade in großen Programmen viel Speicherplatz, Ausführungszeit und Ladezeit auf der Kassette. Sie sollten hier einen vernünftigen Kompromiß eingehen, eventuell zwei Programmkopien auf Kassette anlegen: eine mit Anmerkungen und eine ohne.

NICHT JEDE ANWEISUNG MUß MIT EINER ZEILEN- NUMMER BEGINNEN!

Sie können mehrere Anweisungen in eine Zeile schreiben und damit Zeit und Speicherplatz sparen. Die Anweisungen müssen mit einem ':' getrennt werden.

Statt:

```
10 X=5
20 Y=23
30 A=X+Y
```

Schreiben Sie:

```
10 X=5 : Y=23 : A=X+Y
```

Es lassen sich noch mehr Anweisungen in eine Zeile schreiben. Eine Zeile darf aber nicht länger als 255 Zeichen sein!

RESET STATT BREAK!

Es ist einfacher. Das Programm wird unterbrochen, und statt zwei Tasten haben Sie nur eine zu bedienen. Sie brauchen keinen Programmverlust, wie bei anderen Computern, zu befürchten.

Wenn Sie aber das Programm mit RESET abbrechen, können Sie es nicht wieder mit CONT starten. Es muß dann mit RUN gestartet werden.

KAPITEL 4

LASERBASIC BEHERRSCHT DAS PROGRAMM FÜR FAULE LEUTE

- IF....THEN
- FOR....TO....STEP....NEXT
- READ....DATA
- RESTORE
- GOTO
- GOSUB....RETURN
- ON....GOTO / ON....GOSUB
- CLEAR

WARUM DIESE ANWEISUNGEN FÜR BEQUEME LEUTE SIND

Dies sind die zeitersparenden Anweisungen. Sie weisen den Computer an, die eintönigen, sich wiederholenden Jobs zu tun, für die Computer so gut geeignet sind. Anstelle von Seiten und Seiten an Programmen zu schreiben, wird der Computer eine Menge Arbeit tun mit nur ein oder zwei Anweisungen. Sie bequeme Person!

IF...THEN

WAS ES BEDEUTET

Mit dieser Anweisung geben Sie dem Computer das Mittel, eine Entscheidung zu treffen, die jede Art von Verzweigung haben kann. Was Sie sagen ist, wenn (IF) dies wahr ist, dann (THEN) tun dies. (Wenn IF nicht wahr ist, dann wird das Programm weiterlaufen).

WIE SIE ES SCHREIBEN

```
10 A = 3
20 B = 7
30 IF A + B > 9 THEN PRINT "IT'S BIGGER"
    (wenn A + B > 9, dann schreibe "es ist
    grösser").
```

Sie können praktisch jede Bedingung, die Sie wünschen, unter "IF" aufstellen. Und Sie können praktisch jede Anweisung unter "THEN" aufstellen.

Zeile 30 sollte sein:

```
30 IF A + B > 9 THEN END
```

oder

```
30 IF A + B > 9 THEN A = -3
```

WAS SIE DAMIT TUN KÖNNEN

Wenn Sie wünschen, dass der Computer eine Wahlmöglichkeit hat, können Sie diese Anweisung fast immer benutzen. Zum Beispiel, in Zeichentrickfilmen könnten Sie eingeben, dass, wenn (IF) das sich bewegende Objekt zum Rand der Bildfläche kommt, es sich dann (THEN) herumwendet und in die andere Richtung läuft.

Wenn Sie mit dieser Anweisung vertraut sind, werden Sie viele Möglichkeiten entdecken, in denen Sie sie benutzen können. Sehr wahrscheinlich sind Sie dieser Anweisung schon bei Ihrem Computer begegnet. Im Laserbasic gibt es wahrscheinlich eine Programmzeile, die sagt:

10998 Wenn der dumme Mensch herumpfuscht, dann
schreibe "SYNTAX ERROR".

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

WAS ES BEDEUTET

Es ist nicht so schwierig wie es aussieht. Die Anweisung FOR . . . TO . . . STEP . . . NEXT ist einfach eine Reihe von Schritten, die den Computer veranlasst, eine Menge Arbeit für Sie zu tun, ohne dass es Sie viel kostet.

Sie können damit eine ähnliche Operation für eine zunehmende oder abnehmende Variable durchführen. Zum Beispiel, wenn Sie aus irgendeinem dummen Grund die Zahl 3 nehmen und zu jeder Zahl zwischen 1 und 100 addieren möchten, dann brauchen Sie nur 3 Programmlinien:

WIE SIE ES SCHREIBEN

```
10 FOR X = 1 TO 100 STEP 1
20 PRINT X + B
30 NEXT X
```

Für jede siebte Zahl zwischen 1 und 100 schreiben Sie

```
10 FOR X = 1 TO 100 STEP 7
```

Als "STEP" können Sie irgendeine Zahl nehmen, auch wenn sie nicht voll geteilt werden kann. Sie können als "STEP" auch Teilzahlen wie 0.5 oder 0.3 nehmen. Und wenn Sie als STEP "1" benutzen, brauchen Sie es nicht einmal zu schreiben. Der Computer wird STEP 1 automatisch durchführen. (Die Zeile wäre: 10 FOR X = 1 TO 100). Aber jede andere Zahl als +1 muss eingegeben werden. Auch wenn Sie um 1 nach rückwärts gehen wollen, müssen Sie schreiben: FOR X = 100 TO 1 STEP -1.

Vergessen Sie niemals "NEXT" in Ihrer Anweisung. Wenn Sie NEXT vergessen, wird der Computer 3 zur ersten Zahl (1) addieren und Sie bekommen als Antwort 4. Wenn Sie nicht "NEXT" schreiben, wird der Computer nicht die nächste Zahl durchführen.

Wenn Sie schlau sein wollen, und die "NEXT"-Anweisung ohne "FOR" gebrauchen wollen, wird der Computer sich über Sie lustig machen und Ihnen folgende Mitteilung geben: NEXT WITHOUT FOR ERROR IN 30. (Die 30 ist die Zeilennummer).

WAS SIE DAMIT TUN KÖNNEN

Diese Anweisung kann für unangenehme Aufgaben wie Arithmetik oder für feine Arbeiten wie Grafiken benutzt werden. Sie können es benutzen, um Linien langsam zu zeichnen, so dass Sie sie auf dem Bildschirm erscheinen sehen.

READ...DATA

WAS ES BEDEUTET

Dies ist ein anderer Arbeitssparer. Sie können damit die gleiche Operation an verschiedenen Werten mit nur einigen Zeilen von Programmschreiben durchführen.

Die DATA-Anweisung sagt dem Computer die wirklichen Datenwerten. Diese können Zahlen oder Wörter oder Sätze sein. Aber alle Werte, Zahlen oder Text MÜSSEN durch Kommas getrennt sein.

Die READ-Anweisung befiehlt dem Computer, die Daten zu lesen. Die erste Variable (X,Y oder was immer) in der READ-Anweisung wird den ersten Wert in der DATA-Anweisung benutzen. Der zweiten Variablen in READ wird der zweite Wert in DATA zugeteilt, und so weiter. Jede READ-Variable muss durch ein Komma getrennt sein.

Die Anzahl der Werte in der DATA-Anweisung muss gleich oder grösser sein als die Anzahl der Variablen in der READ-Anweisung. Andernfalls erhalten Sie die Mitteilung: OUT OF DATA ERROR. (Keine Daten, Irrtum).

WIE SIE ES SCHREIBEN

```
10 DATA 50, 30 "FRED", 2.5
20 READ A, B, C$, D
30 PRINT A, B, C$, D
40 END
RUN
```

Der Computer wird schreiben : 50 ... 30 ... FRED... 2.5

WAS SIE DAMIT TUN KÖNNEN

Sie können eine Menge Arbeit bei schrecklichen Jobs, wie zum Beispiel Durchschnittszahlen errechnen, ersparen. Aber Sie können auch interessante Variationen damit tun. Diese Anweisung erlaubt Ihnen, den gleichen Brief mit verschiedenen Namen zu schreiben.

RESTORE

WAS ES BEDEUTET

Die RESTORE-Anweisung sagt dem Computer, die Daten, die in einem READ zuvor benutzt wurden, für ein neues READ zu gebrauchen. Wenn Sie dies tun, ist es immer eine gute Idee, neue Variablen für des zweite READ zu benutzen. Wenn Sie X und Y im ersten READ hatten, dann benutzen Sie A und B im zweiten. Es geht nicht darum, dass der Computer durcheinander gebracht würde, er wird die neuen Variablen ganz froh akzeptieren, vielmehr könnten Sie verwirrt werden und nicht mehr sicher sein, von welchem X Sie reden, wenn das Programm weiterläuft.

WIE SIE ES SCHREIBEN

```
10 DATA 13, "FRED", 2.7
20 READ X, T$
30 RESTORE
40 READ A, B$
50 PRINT X, Y$
60 PRINT A, B$
RUN
```

Die Antwort, die Sie bekommen, ist:

```
13. . . . FRED
13 . . . .FRED
```

Die RESTORE-Anweisung veranlasst, dass der erste READ-Ausdruck nach RESTORE Daten von dem ersten DATA-Ausdruck, dem er begegnet, zuordnet. Dieser DATA-Ausdruck kann vor RESTORE stehen wie im Beispiel gezeigt ist. Er kann aber auch nach RESTORE stehen, und sogar ganz am Ende des Programmes, wenn Sie es möchten.

WAS SIE DAMIT TUN KÖNNEN

RESTORE ermöglicht es, verschiedene Operationen mit der gleichen Serie von Daten durchzuführen, ohne dass Sie alles noch einmal schreiben müssen. Ein anderes Werkzeug für die bequeme Person! Nun, Bequemlichkeit war wahrscheinlich der Grund, weshalb Sie überhaupt einen Computer gekauft haben.

GOTO

WAS ES BEDEUTET

Genau so wie es aussieht, die GOTO-Anweisung befiehlt dem Computer, hierhin zu gehen (go to) und dorthin zu gehen (go to), und das zu tun, was getan werden muss, wenn er an der Stelle ist. In allen Fällen, wenn Sie Ihrem Computer GOTO anweisen, müssen Sie ihn zu einer Zeilennummer im Programm senden, auch wenn Zeiten kommen, wo Sie ihn lieber zur . . . (einem anderen Platz als Himmel) senden möchten.

Anders als bei der GOSUB/RETURN Anweisung wird der Computer nicht nach einer GOTO-Anweisung zurückkommen. Wenn er zu einer neuen Adresse gegangen ist, wird er nicht zurückkommen, es sei denn, er findet eine andere GOTO-Anweisung nahe der neuen Adresse, die ihm sagt, zurück-zukehren.

WIE ES GESCHRIEBEN WIRD

```
10 PRINT "DAS IST LEBEN"  
20 PRINT "WAS IST LEBEN"  
30 PRINT "ES IST EINE ZEITSCHRIFT"  
40 PRINT "WIEVIEL KOSTET ES"  
50 PRINT "ES KOSTET EINEN DOLLAR"  
60 PRINT "ABER ICH HABE NUR 50 CENTS"  
70 GOTO 10
```

Diese Art von Programmen wird endlose Schleife genannt. Wenn Sie es tatsächlich ausprobieren, werden Sie herausfinden, dass es niemals aufhört. Der einzige Weg es zu beenden, ist, die Anweisung BREAK oder die RESET-Taste zu benutzen. GOTO-Anweisungen werden nicht immer in dieser Art von Schleife enden, aber dies passiert, wenn Sie nicht vorsichtig im Gebrauch von GOTO sind.

WAS SIE DAMIT TUN KÖNNEN

Sie können in Ihrem Programm vorwärts und rückwärts gehen, um sich wiederholende Arbeiten durchzuführen, wie zum Beispiel mathematische Operationen, oder Text für Anleitungen anzuzeigen, oder die gleiche Grafik immer wieder zu zeichnen. Es wird auch oft in Computer-Entscheidungssituationen mit der IF . . . THEN Anweisung gebraucht. (IF X = Y THEN GOTO 50).

GOSUB.....RETURN

WAS ES BEDEUTET

Mit dieser Anweisung sagen Sie dem Computer, eine Subroutine oder ein Unterprogramm an einer bestimmten Stelle durchzuführen und nicht zurückzukommen, bis er die Anweisung RETURN findet. Der Computer wird auf die angegebene Zeilennummer springen und die folgenden Zeilen durchführen, bis er auf die Anweisung RETURN trifft. Dann wird er zur Programmzeile zurückgehen, die der Zeile folgt, die ihm die GOSUB-Anweisung gab. Dies ist eine so grossartige Anweisung, dass es schade ist, dass sie nicht auf gewisse Leute angewendet werden kann.

WIE ES GESCHRIEBEN WIRD

```
10 PRINT "ENTER ANY TWO NUMBERS"  
20 PRINT "I'LL ADD THEM UP FOR YOU"  
30 PRINT "GIVE ME THE NUMBERS"  
40 INPUT A, B  
50 GOSUB 70  
60 GOTO 30  
70 C = A + B  
80 PRINT C  
90 RETURN
```

Dieses Programm wird solange laufen, bis Sie BREAK oder RESET eingeben. Selbstverständlich, wenn Sie wirklich ganz grosse Zahlen eingeben, wird es verrückt und von selbst kaputt gehen. Aber Sie würden es nicht dazu kommen lassen, oder?

ANMERKUNG: In diesem Falle muss das Wort RETURN voll ausgeschrieben werden, oder Sie bekommen eine grosse Überraschung.

ON...GOTO/ON...GOSUB

WAS ES BEDEUTET

Es richtet den Programmfluss je nach Bedeutung eines Ausdruckes.

WIE SIE ES SCHREIBEN

ON (Ausdruck) GOTO (Liniennummer 1, Liniennummer 2,)

ON (Ausdruck) GOSUB (Liniennummer 1, Liniennummer 2,)

WIE ES GEBRAUCHT WIRD

Der Wert eines Ausdruckes muss immer eine Zahl kleiner als oder gleich 255 sein. Wenn der Wert bestimmt ist, wird das Programm zur korrespondierenden Zeilennummer in der Liste geleitet, die entweder der GOSUB oder GOTO-Anweisung folgt. Z. Bsp. Wenn der Wert des Ausdruckes 5 ist, dann wird das Programm auf die fünfte Zeilennummer gehen, und wenn es 9 ist, auf die neunte Zeilennummer.

CLEAR

WAS ES BEDEUTET

Diese Anweisung löscht alle Programmvariablen, stellt "FOR" und "GOSUB" neu ein und speichert Daten um.

WIE SIE ES SCHREIBEN

```
10 CLEAR
```


KAPITEL 5

LASERBASIC BEFIEHLT DAS PROGRAMM FÜR TON UND MUSIK

- WIE DER COMPUTER TÖNE ERZEUGT
- SOUND
- SGEN

WIE DER COMPUTER TÖNE ERZEUGT

Im Laser 2001 ist ein Schaltsystem eingebaut, das herrliche Töne erzeugen kann. Wenn Sie spezielle Toneffekte oder Ihre eigene Musik komponieren möchten, können Sie die Befehle SGEN oder SOUND benutzen. Das integrierte Schaltsystem nennt man den Ton-Erzeuger. Dieser Ton-Erzeuger hat drei Ton kanäle und einen Geräusch kanal. Indem Sie diese vier kanäle kontrollieren, können Sie jede Art von Ton, Stimme, Geräusch oder Musik erzeugen.

SOUND

WAS ES BEDEUTET

Dieser Befehl kann irgendeinen Kanal des Ton-Erzeugers einstellen mit einer verschiedenen Frequenz Dauer und Lautstärke.

WIE SIE ES SCHREIBEN

10	SOUND	(P,D,V),	(P,D,V)	(P,D,V)	(w/P,D,V,CH)
		Kanal 1	Kanal 2	Kanal 3	Geräuschkanal

P = Tonhöhe von 0 bis 255; 0 ist die tiefste Frequenz.

D = Tondauer von 0 bis 255; 0 ist die kürzeste Tondauer.

V = Lautstärke von 0 bis 15; ist am lautesten.

W/P = weisses Rauschen oder periodisches Geräusch;
1 für periodisches Geräusch
2 für weisses Rauschen

CH = irgendein Kanal von 1 bis 3 oder der Geräuschkanal
kann ausgewählt werden.

DIE FREQUENZKONVERSIONSTABELLE IST WIE FOLGEND;

GEGENWÄRTIGE FREQUENZ = 62500 (Tonhöhe 255)

KONVERSIONSTABELLE TONHÖHE – FREQUENZ

Tonhöhe		Frequenz
0	—	245.098039
10	—	255.102041
20	—	265.957447
30	—	277.777778
40	—	290.697674
50	—	304.878049
60	—	320.512821
70	—	339.837838
80	—	357.142857
90	—	378.787879
100	—	403.225807
110	—	431.034483
120	—	462.962963
130	—	500
140	—	543.478261
150	—	595.238095
160	—	657.894737
170	—	735.294118
180	—	833.333333
190	—	961.538461
200	—	1136.36364
210	—	1388.88889
220	—	1785.71429
230	—	2500
240	—	4166.66667
250	—	12500

SGEN

WAS ES BEDEUTET

Mit dieser Anweisung können Sie den Ton-Erzeuger direkt kontrollieren.

WAS SIE ES SCHREIBEN:

10 SGEN A, B, C, D

wobei A, B, C, D etc. eine Reihe von Daten ist, die eingegeben werden, um den Ton-Erzeuger direkt zu kontrollieren.

Das folgende Beispiel zeigt Ihnen, wie die SGEN-Anweisung funktioniert. Sie können die Daten nach der SGEN-Anweisung ändern, um zu sehen, was geschieht.

```
10  REM PROGRAM FOR BELL SOUND (Programm für
    Glockenton)
20  REM TURN OFF ALL SOUND CHANNEL (alle
    Tonkanäle abstellen)
30  SGEN 159, 191, 223, 255
40  REM SET CH 1 = 679 Hz, CH 2 = 694 Hz
50  SGEN 140, 5, 170, 5
60  FOR B=0 to 11 (für B=0 bis 11)
70  REM B=No. for bell sound (b = nr. für Glockenton)
80  SGEN I, I + 32
90  FOR D = 0 to 75: next D
100 NEXT I
110 NEXT B
120 PRINT "END OF SOUND" (Tonende)
```


KAPITEL 6

DAS SCHRECKLICHE RECHNEN, DAS SIE ZU VERMEIDEN HOFFTEN

- WAS IST EIN OPERATOR
 - ARITHMETISCHE OPERATOREN
 - LOGISCHE OPERATOREN
 - RELATIONALE OPERATOREN
- WAS IST EINE FUNKTION?
 - ABS
 - ATN
 - COS
 - EXP
 - INT
 - LOG
 - RND
 - SGN
 - SIN
 - SQR
 - TAN

WAS SIND OPERATOREN

Operatoren arbeiten zusammen mit einem Ausdruck oder einer Variablen, um einen einzigen Wert zu produzieren.

Es gibt drei Arten von Operatoren:

arithmetische

logische

relationale

ARITHMETISCHE OPERATOREN

Dies sind die gewöhnlichen mathematischen Operatoren. Sie werden immer in einer bestimmten Rangordnung durchgeführt. Die untenstehende Liste zeigt diese Rangordnung:

Operator	Operation	Beispiel
**	Exponentialrechnung	$A^{**}B$
-	negativer Wert	$-A$
*, /	Multiplikation	$A*B$
	Division	A/B
+, -	Addition	$A+B$
	Subtraktion	$A-B$

Diese Reihenfolge der operatoren kann durch den Gebrauch von Klammern geändert werden, da die Formel in der Klammer Vorrang hat.

Innerhalb der Klammern wird die obenstehende Reihenfolge auch eingehalten.

Beispiel:	Formel	Ergebnis
	$2 + 10/2$	7
	$(2 + 10)/2$	6

LOGISCHE OPERATOREN

Diese Operatoren arbeiten Schritt für Schritt an Werten, um ein Ergebnis zu produzieren, das entweder 1 oder 0, d.h. richtig oder falsch ist.

Logische Operatoren haben eine Reihenfolge, und diese sowohl als die Schritt für Schritt Operation sind untenstehend aufgelistet.

OPERATOR	A	B	ERGEBNIS
NOT (NICHT)	1	—	0
	0	—	1
AND (UND)	1	1	1
	1	0	0
	0	1	0
	0	0	0
OR (ODER)	1	1	1
	1	0	1
	0	1	1
	0	0	0

Beispiele:

63 AND 16=16 Da 63 = binarisch 111111 und 16= binarisch 100000 ist, ist das Ergebnis AND 100000 oder 16.

4 AND 2=0 Da 4 = binarisch 100 und 2 = binarisch 10 ist, ist das Ergebnis binarisch 0.

4 OR 2=6 Binarisch 100 OR 10 = 110 oder 6 dezimal.

NOT 0=-1 Das Bit-Komplement von 0 ist 1111 1111 1111 1111 oder -1 dezimal.

RELATIONALE OPERATOREN

Diese Operatoren haben - genau wie die logischen Operatoren - immer nur eines von 2 Dingen als Ergebnis: "falsch" oder "richtig".

Operator	Relation	Beispiel
=	Gleichheit	$A = B$
<>	Ungleichheit	$A < > B$
<	weniger als	$A < B$
>	grösser als	$A > B$
<= oder =<	weniger als oder gleich	$A < = B$
>= oder =>	grösser als oder gleich	$A > = B$

Wenn arithmetische, relationale oder logische Operatoren in einem Ausdruck vorhanden sind, dann ist die Reihenfolge der Evaluation : arithmetisch, relational und dann logisch.

Die logischen und relationale Operatoren können benutzt werden, um Entscheidungen für den Programmablauf zu treffen.

WAS IST EINE FUNKTION?

Eine Funktion ist ein Gesetz, das für einen bestimmten Wert angewendet, einen neuen Wert ergibt.

ZUM BEISPIEL, SQR IST DIE FUNKTION DER QUADRATWURZEL.

WENN WIR SCHREIBEN:

PRINT SQR (9)

erhalten wir die Antwort : 3.

NACHSTEHEND GEBEN WIR EINE LISTE VON NUMERISCHEN FUNKTIONEN UND EINE KURZE ERKLÄRUNG.

ABS (X)	ergibt den absoluten Wert von X.
ATN (X)	ergibt den Wert des Tangensbogens von X. Er ist in Radianen ausgedrückt.
COS (X)	ergibt den Wert des Cosinus von X. X ist ausgedrückt in Radianen.
EXP (X)	ergibt den Wert der Konstanten "E" (2.71828) zur Basis X.
INT (X)	ergibt die grösste Ganzzahl, die kleiner oder gleich X ist.
LOG (X)	ergibt den natürlichen Logarithmus von X.

RND (X)	ergibt eine Zufallszahl zwischen 0 und 1. X kontrolliert das Ergebnis der Zufallszahlen. $X > 0$ ergibt eine neue Zufallszahl. $X=0$ ergibt immer die gleiche Zufallszahl, $X < 0$ startet eine neue Folge von Zufallszahlen.
SGN (X)	ergibt 1, wenn $X > 0$ ist. ergibt 0, wenn $X=0$ ist. ergibt -1, wenn $X < 0$ ist.
SIN (X)	ergibt den Wert des Sinus von X. X ist in Radianen ausgedrückt.
SQR (X)	ergibt die Quadratwurzel von X.
TAN (X)	ergibt den Wert des Tangens von X. X ist in Radianen ausgedrückt.

KAPITEL 7

SCHRECKLICHE UND UNVERSTÄNDLICHE STRINGS

- STRINGFUNKTIONEN UND VARIABLEN

- ASC
 - CHR\$
 - GET
 - LEFT\$
 - LEN\$
 - MID\$
 - RIGHT\$
 - STR\$
 - VAL

- STRINGVERGLEICHE

STRINGS UND STRINGVARIABLEN

Ein "String" ist eine Kette von Zeichen wie z. Bsp. "LASER", "THIS IS A TEST" oder "ABCD". Genau wie numerischen Variablen kann Stringvariablen ein bestimmter Wert zugeteilt werden. Stringvariable werden von numerischen Variablen unterschieden, indem man ein "\$" nach der Variablen schreibt.

Z. BSP. VERSUCHEN SIE FOLGENDES:

```
A$ = "LASER 2001"  
PRINT A$
```

In diesem Beispiel geben wir der Stringvariablen A\$ den Wert "LASER 2001". Beachten Sie, dass wir den zu A\$ gehörenden Zeichenstring in Anführungszeichen schreiben.

WIR KÖNNEN AUCH FUNKTIONEN ZUM MANIPULIEREN VON STRINGS BENUTZEN. NACHSTEHEND FINDEN SIE DIE LISTE DER STRINGFUNKTIONEN.

ASC(X\$)	ergibt den numerischen ASCII-Wert des ersten Zeichens von String X\$. (Siehe Appendix für ASCII-Code).
CHR\$(X)	ERGIBT EIN Zeichen, und zwar das ASCII-Equivalent für den Ausdruck (I). (Siehe Appendix für ASCII-Code).
GET A\$	Geben Sie ein einzelnes Zeichen von der Tastatur ein. Wenn die Daten von der Tastatur sind, werden sie in die Variable gesetzt, die in GET spezifiziert ist. Wenn keine Daten zur Verfügung stehen, wird das Basic Programm weiterwarten.
LEFT\$(X\$,I)	ergibt die ganz links stehenden Zeichen von I aus dem String X\$.
LEN (X\$)	ergibt die Länge des Strings X\$ in Zeichen. Nicht gedruckte Zeichen und Leerstellen werden auch als Teil der Länge gezählt.
MID\$(X\$,I,J)	ergibt die Zeichen von String X\$, beginnend mit dem Zeichen I für eine Länge von J Zeichen.

<code>RIGHT\$(X\$, I)</code>	ergibt die ganz rechts stehenden Zeichen von I des Strings X\$.
<code>STR\$(X)</code>	ergibt einen String, der die Zeichendarstellung des numerischen Wertes X ist. z. Bsp. <code>STR\$(1.23) = "1.23"</code>
<code>VAL(X\$)</code>	wandelt den Stringausdruck (X\$) in eine Zahl um. z. Bsp. <code>VAL("1.23") = 1.23</code>

STRINGVERGLEICHE

Der Vergleich von Strings wird mit Hilfe von ASCII- Codes durchgeführt, ein Zeichen nach dem anderen, bis ein Unterschied gefunden wird. Wenn während eines Vergleichs von 2 Strings das Ende eines Strings erreicht ist, wird der kürzere String als kleiner im Wert betrachtet.

Die Stringvergleichsoperatoren sind die gleichen wie die relationalen Operatoren.

Diese sind:

- =
- >
- <
- < = oder =<
- > = oder =>
- < >

Für Stringverkettung benutzen wir das Symbol "+". Z. Bsp. A\$ = B\$ + C\$. Der resultierende String muss weniger als 256 Zeichen haben.

KAPITEL 8

LASERBASIC BEHERRSCHT DAS SCHWIERIGE PROGRAMM

- DIM
- PEEK
- POKE
- CALL

DIM

WAS ES BEDEUTET

Die Anweisung DIM teilt Platz für Matrizen zu. Alle Matrizenbestandteile werden durch diese Eingabe auf Null gestellt.

WIE SIE ES SCHREIBEN

1) DIM Variable (Grösse 1, Grösse 2 . . .)

z. Bsp. 100 DIM A (3)

2) Matrizen können Dimensionen von 1 bis 255 haben.

z. Bsp. 100 DIM R3 (5,5), D\$ (2,2,2)

3) Matrizen können während des Programmierens dynamisch dimensioniert werden.

z. Bsp. 100 DIM Q1 (N)

Wenn eine Matrix nicht ausdrücklich mit einer DIM-Anweisung dimensioniert ist, so wird angenommen, dass es sich um eine ein-dimensionale Matrix handelt, deren einziger Subscript zwischen 0 und 10 liegt.

PEEK

WAS ES BEDEUTET

Die PEEK-Funktion zeigt den Inhalt einer Adresse im Speicher in dezimaler Form an. Der angezeigte Wert liegt zwischen 0 und 255. Ein Versuch, eine nicht vorhandene Adresse im Speicher zu lesen, wird einen unbekannten Wert anzeigen.

WIE SIE ES SCHREIBEN

10 PRINT PEEK (I)

wobei I die Speicheradresse in dezimaler Form ist.

POKE

WAS ES BEDEUTET

Die Anweisung POKE speichert das Byte in die Speicheradresse. Das zu speichernde Byte muss zwischen 0 und 255 liegen.

WIE SIE ES SCHREIBEN

10 POKE I, J

wobei I die Speicheradresse ist und J das zu speichernde Byte.

ACHTUNG

Unachtsame Benutzung der POKE-Anweisung kann Störungen im Programm hervorrufen oder es zerstören.

WOFÜR ES GEBRAUCHT WIRD

Zusammen mit der PEEK-Funktion. Sie können Argumente zur Unteroutine der Maschinensprache übertragen. Sie können auch PEEK und POKE benutzen, um eine Speicherdiagnose oder einen Assembler in BASIC zu schreiben.

CALL

WAS ES BEDEUTET

Diese Anweisung veranlasst das Übertragen des Program-
mablaufs zu einer Unteroutine der Assemblersprache.

WIE SIE ES SCHREIBEN

10 CALL I

*Wobei I die Speicheradresse ist, die der Anfang der Unter-
routine ist.*

KAPITEL 9

LASERBASIC ANWEISUNGEN FÜR IHREN KASSETTENREKORDER

- CLOAD
- CSAVE
- CRUN
- BLOAD
- BSAVE
- BRUN
- STORE
- RECALL

**SIE BRAUCHEN EINEN KASSETTENREKORDER,
BEVOR IHNEN DIESE KAPITEL ETWAS NÜTZT.**

Diese Anweisungen helfen Ihnen, Programme von einem Kassettenrekorder zu speichern und zurückzurufen.

Ihr User Reference Guide (Gebrauchsreferenzbuch) wird Ihnen zeigen, wie der Recorder mit dem Laser 2001 verbunden wird. Wenn alles verbunden ist und funktioniert, können Sie die Anweisungen in diesem Kapitel benutzen.

CLOAD

WAS ES BEDEUTET

Es meint einfach LOAD. "C" steht für Kassette. Wenn Sie fertige Programme oder Ihre eigenen Programme auf Band gespeichert haben, dann verbinden Sie einfach Ihren Kassettenrekorder mit dem Computer und drücken Sie auf die Spieltaste. Dann schreiben Sie CLOAD auf der Tastatur.

Die Kassette wird zu laufen beginnen und die Programmzeile werden auf dem Bildschirm erscheinen.

Um besondere Programme einzugeben, schreiben Sie:

CLOAD

CSAVE

WAS ES BEDEUTET

Diese Anweisung ermöglicht es Ihnen, ein Programm auf Kassettenband zu speichern. "C" steht für Kassette.

SIE SCHREIBEN:

CSAVE

und der Computer wird das Programm auf das Band übertragen.

CRUN

WAS ES BEDEUTET

Dies weist den Computer an, ein Programm vom Kassettenband zu laden und es sofort durchzuführen. Wenn Sie nur CLOAD benutzen, wird es nur geladen. Sie müssen dann selbst die Anweisung RUN geben.

SIE SCHREIBEN:

CRUN

BLOAD

Obwohl dies sich so anhört, als würde es jemanden beschreiben, der zu viel gegessen hat, weist der BLOAD-Befehl den Computer an, ein Binärprogramm vom Band zu laden. Binärprogramme sind meistens Spiele, aber sie können auch andere Software einschliessen wie Textverarbeitungen oder Mehrfachtabellen-Analysen.

Sie müssen die BLOAD-Eingabe benutzen, weil Ihr Computer meistens mit BASIC arbeitet und nicht eine Binärsprache erkennt, ohne dass Sie es ihm sagen.

Um ein Binärprogramm vom Band zu laden, schreiben Sie nur:

BLOAD

BSAVE

WAS ES BEDEUTET

Diese Anweisung ermöglicht Ihnen, ein Binärprogramm auf einem Kassettenband zu speichern. B steht für Binärprogramm. Bevor Sie ein Binärprogramm speichern können, müssen Sie wissen, wo sich das Programm befindet und welche Länge es hat.

Um ein Binärprogramm zu speichern, schreiben Sie:

BSAVE A, L

wobei A die Anfangsadresse des Programms ist und B die Länge des Programmes.

BRUN

WAS ES BEDEUTET

Genau wie CRUN weist dieser Befehl den Computer an, ein Programm zu laufen, sobald es vom Kassettenband geladen wurde. Der einzige Unterschied ist, dass BRUN Binärprogramme eingibt und läuft anstelle von BASIC Programmen.

Sie brauchen nur zu schreiben:

BRUN

STORE

WAS ES BEDEUTET

Diese Anweisung überträgt Werte von spezifischen Variablen und Daten auf ein Kassettenband. Selbstverständlich muss der Rekorder richtig verbunden und auf Aufnahme gestellt sein, wenn diese Anweisung durchgeführt werden soll.

WIE SIE ES SCHREIBEN

10 STORE A\$

WOFÜR ES GEBRAUCHT WIRD

Diese Anweisung wird benutzt, um Daten auf Kassettenband zu speichern, besonders gut geeignet für Programme wie z. Bsp. Adressenkartei.

RECALL

WAS ES BEDEUTET

Liest die spezifischen zu speichernden Daten auf dem Kassettenband und ordnet sie spezifischen Variablen zu.

WIE SIE ES SCHREIBEN

10 RECALL A\$

WOFÜR ES GEBRAUCHT WIRD

Diese Anweisung wird zusammen mit der STORE Anweisung gebraucht, um Daten auf das Kassettenband zu lesen und zu schreiben.

KAPITEL 10

LASERBASIC WEIST DAS PROGRAMM FÜR DEN DRUCKER AN

- LLIST
- LPRINT

Sie brauchen einen Drucker, bevor dieses Kapitel Ihnen etwas nützt.

Wenn Sie einen Drucker haben, schauen Sie im Anleitungsbuch nach, wie er angeschlossen wird. Wenn er richtig verbunden ist und läuft, können Sie mit den obenstehenden Anweisungen Programme, Texte und sogar Grafiken drucken.

LLIST

Anstelle die Anweisung LIST zu gebrauchen, um eine Liste von allen Programmzeilen oder einer speziellen Zeile zu erhalten, können Sie die Anweisung LLIST benutzen und die Liste wird auf dem Drucker ausgedruckt.

WIE SIE ES SCHREIBEN

LLIST

WAS SIE DAMIT TUN KÖNNEN

Diese Anweisung hat zwei Hauptfunktionen. Die erste ist, dass Sie eine Kopie Ihres Programmes als Referenz haben, die andere ist zum Austesten eines Programmes, da die meisten BASIC Programme zu lang sind, um vollständig auf dem Bildschirm gesehen zu werden. Mit einer LLIST Kopie kann das ganze Programm vor Ihnen ausgeschrieben werden anstelle von nur kleinen Ausschnitten auf dem Bildschirm zu sehen.

LPRINT

WAS ES BEDEUTET

Diese Anweisung veranlasst den Computer, anstelle des Bildschirms mit dem Drucker zu drucken. Sie können LPRINT benutzen, während das Programm läuft, um zu drucken. Oder Sie können LPRINT benutzen, um all auszudrucken, was Sie auf dem Bildschirm haben (in diesem Falle brauchen Sie keine Zeilennummer).

WIE ES GESCHRIEBEN WIRD

```
10 LET A = 5
20 LET B = 6
30 LPRINT A + B
40 END
```

ANMERKUNG: Genau wie bei der Anweisung PRINT müssen Sie Text, den Sie ausdrucken möchten, in Anführungszeichen setzen. Zum Beispiel, in dem obigen Beispiel wäre das Ausdrucken 11. Wenn Sie die Zeile 30 umändern würden in 30 LPRINT "A + B" dann wäre die Antwort A + B.

WAS SIE DAMIT TUN KÖNNEN

Benutzen Sie den Drucker wofür Sie ihn gekauft haben.

KAPITEL 11

ANHANG

- FEHLERMELDUNGEN
- ZEICHEN-CODE TABELLE

FEHLERMELDUNGEN

BAD SUBSKRIPT (SUBSCRIPT OUT OF RANGE)

Ein Feldelement wird mit einem Index angesprochen, der außerhalb des mit **DIM** festgelegten Bereiches liegt. Ein Feldelement wird mit einer Anzahl Indizes angesprochen, die nicht in der entsprechenden **DIM**-Anweisung festgelegt ist.

CAN'T CONT (CAN'T CONTINUE)

Ein Programm kann mit der **CONT**-Anweisung nicht fortgesetzt werden, weil . . .

- . . . ein Fehler im Programm die Unterbrechung verursacht hat.
- . . . es nach der Unterbrechung verändert wurde.
- . . . es nicht mehr existiert.
- . . . es noch nicht gestartet wurde.

DIVISION BY ZERO

Eine Division mit Null wurde im Verlauf der Lösung eines numerischen Ausdrucks versucht. Wird im Verlauf der Lösung eines Ausdrucks ein Wert = Null zu einer negativen Basis erhoben, wird diese Fehlermeldung ebenfalls ausgegeben.

FORMULA TOO COMPLEX (STRING FORMULA TOO COMPLEX)

Ein String-Ausdruck ist zu lang oder zu komplex. Der Ausdruck sollte in zwei kleinere Ausdrücke aufgelöst werden.

ILLEGAL DIRECT

Eine Anweisung, die nur innerhalb einer Programmliste zulässig ist, wurde direkt von der Tastatur eingegeben.

ILLEGAL QUANTITY

Ein Parameter einer mathematischen oder String-Funktion ist außerhalb des zulässigen Bereiches. Die Fehlermeldung wird auch ausgegeben:

1. Subskriptor negativ oder unzulässig groß.
2. Das Argument der LOG-Anweisung ist = 0 oder negativ.
3. Das Argument der SQR-Funktion ist negativ.
4. Eine negative Mantisse mit einem nicht-integer Exponent tritt auf.
5. Aufruf der USR-Funktion mit fehlender Startadresse.
6. Allgemein unkorrektes Argument in vielen Anweisungen.

NEXT WITHOUT FOR

Eine Variable in einer **NEXT**-Anweisung hat keinen Bezug zu Variablen in offenen **FOR**-Anweisungen, oder es existiert keine offene **FOR**-Anweisung.

OUT OF DATA

Für eine **READ**-Anweisung existiert kein ungelesener Wert in der **DATA**-Liste.

OUT OF MEMORY

Diese Fehlermeldung wird ausgegeben, wenn ein Programm zu lang ist, zuviele **FOR . . . NEXT**-Schleifen oder **GOSUB**-Anweisungen aktiv sind oder wenn zu viele Variablen definiert wurden. Tief verschachtelte Ausdrücke werden ebenfalls mit dieser Meldung angezeigt.

OVERFLOW

Das Ergebnis einer Berechnung ist größer als der zulässige höchste Zahlenwert. Wird das Ergebnis kleiner als der kleinste zulässige Wert, so wird es zu Null gesetzt, und es wird keine Fehlermeldung ausgegeben.

REDIM'D ARRAY (REDIMENSIONED ARRAY)

Zwei **DIM**-Anweisungen sind auf das gleiche Feld bezogen oder eine **DIM**-Anweisung wird gegeben, nachdem der Defaultwert 10 für dieses Feld vergeben wurde.

RET'N WITHOUT GOSUB

Eine **RETURN**-Anweisung wird gefunden, für die keine aktive **GOSUB**-Anweisung besteht.

STRING TOO LONG

Es wird der Versuch gemacht, eine Zeichenkette mit mehr als 255 Zeichen zu generieren.

SYNTAX (SYNTAX ERROR)

Eine Anweisungszeile oder ein direktes Kommando ist unkorrekt eingegeben worden. Falsche Zeichen, fehlende Klammern, falscher Gebrauch von “.,,:” usw. können die Fehlermeldung verursachen.

TYPE MISMATCH

Ein numerischer Wert wird einer Stringvariablen zugewiesen, oder umgekehrt wird einer numerischen Variablen eine Zeichenkette zugewiesen. Ein Stringargument wird in einer numerischen Funktion formuliert, oder ein numerisches Argument wird in einer Stringfunktion benutzt.

UNDEF'D STATEMENT (UNDEFINED LINE)

Eine nicht existierende Anweisungszeile wird mit **GOTO**, **GOSUB**, **IF ... THEN... ELSE** oder **RUN** angesprochen.

ZEICHEN-CODE TABELLE

ASCII CODE	CHARACTER
32	(Space)
33	! (exclamation point)
34	" (quote)
35	# (number or pound sign)
36	\$ (dollar)
37	% (percent)
38	& (ampersand)
39	' (apostrophe)
40	((open parenthesis)
41) (close parenthesis)
42	* (asterisk)
43	+ (plus)
44	, (comma)
45	- (minus)
46	. (period)
47	/ (slant)
48	0
49	1
50	2
51	3
52	4
53	5
54	6
55	7
56	8
57	9
58	: (colón)
59	; (semicolon)
60	< (less than)
61	= (equals)

62	>	(greater than)
63	?	(question mark)
64	@	(at sign)
65	A	
66	B	
67	C	
68	D	
69	E	
70	F	
71	G	
72	H	
73	I	
74	J	
75	K	
76	L	
77	M	
78	N	
79	O	
80	P	
81	Q	
82	R	
83	S	
84	T	
85	U	
86	V	
87	W	
88	X	
89	Y	
90	Z	
091	[
092	\	
093]	
094	^	

095	—
096	‘
097	a
098	b
099	c
100	d
101	e
102	f
103	g
104	h
105	i
106	j
107	k
108	l
109	m
110	n
111	o
112	p
113	q
114	r
115	s
116	t
117	u
118	v
119	w
120	x
121	y
122	z

MICROSOFT ist eine registrierte Handelsmarke von Microsoft Inc.

COLECO, COLECOVISION ist eine registrierte Handelsmarke von Coleco Industries, Inc.

ATARI, VCS ist eine registrierte Handelsmarke von ATARI, Inc.

LASER™ 2001

HOME COMPUTER

©1983 VTL. MADE IN HONG KONG

91-0205-03