

SHARP

TASCHENCOMPUTER

MODELL **PC-1251**

BEDIENUNGSANLEITUNG





INHALTSVERZEICHNIS

	Seite
I. Einleitung	5
II. Betriebshinweise	6
III. Stromversorgung	8
1. Einschalten des Computers	10
2. Ausschalten des Computers	10
3. Die Bedienungselemente	11
4. Die Tastenfunktionen.	13
5. Die Anzeige	15
6. Wahl der Betriebsart.	17
7. Speicherorganisation	17
8. Rechnen ohne Programmunterstützung	18
8.1 Wissenschaftliche Notation	21
8.2 Rechengenauigkeit.	22
8.3 Kontrollieren und Verändern der Eingabe (Editieren).	23
8.4 Playback	23
8.5 Mathematische Funktionen.	24
8.6 Klammerregeln	25
8.7 Sedezimalzahlen	26
8.8 Textausdrücke	28
8.9 Vergleichsausdrücke.	29
9. Sprachelemente.	31
9.1 Numerische Konstante	31
9.2 Textkonstante	31
9.3 Numerische Variable	32
9.4 Textvariablen	32
9.5 Numerische Funktion, Textfunktion.	32
9.6 Numerischer Ausdruck	33
9.7 Textausdruck	34
9.8 Logischer Ausdruck	35
10. Variablen.	37
10.1 Variablen im PC-1251	37
10.2 Standardvariablen	38
10.3 Einfache Variablen	38
10.4 Indizierte Variablen	39
10.5 Besonderheit der Variablen A	40
10.6 Feldvariable	41
10.7 Numerische Feldvariablen.	42
10.8 Textfeldvariablen	43

11. Numerische Funktionen	44
Winkleinheitskommandos:	
DEGREE/RADIAN/GRAD.	45
DMS/DEG	46
Trigonometrische und inverse trigonometrische Funktionen	48
11.1 Mathematische Funktionen	48
LN/LOG	49
EXP	49
ABS	50
INT	50
SGN	51
SQR	51
PI.	52
RND.	52
RANDOM	54
MEM.	54
NOT-Funktion	55
AND.	56
OR.	57
12. Programmieren in BASIC	58
12.1 BASIC-Übersicht.	58
12.2 Programmerstellung	59
12.3 Programmaufbau.	59
12.4 Eingabe eines Programms	60
12.5 Überprüfen des Programms, Editieren und Auflisten	61
12.6 Programmausführung	62
12.7 Fehlermeldung/Fehlersuche.	63
12.8 PC-1251 BASIC	63
NEW.	65
CLEAR	65
DIM	66
LET	70
DATA.	72
READ.	73
RESTORE	74
REM.	75
RUN/GOTO/Definable Keys.	76
AREAD.	78
STOP	80
CONT.	82

END	82
LIST	84
TRACE	85
GOTO	86
ON GOTO	88
GOSUB	90
ON GOSUB	92
RETURN	93
IF . . . THEN	94
FOR . . . TO . . . STEP/NEXT	97
INPUT	101
PRINT	104
PAUSE	110
USING	111
WAIT	116
BEEP	117
PASS	118
12.9 Textfunktionen	120
ASC	121
CHR\$	122
LEN	124
STR\$	125
VAL	126
LEFT\$	128
RIGHT\$	129
MID\$	130
INKEY\$	131
13. Der ReSerVe-Mode	132
14. CE-125 (Option)	134
14.1 Einleitung	134
14.2 Anschluß des CE-125 an den Computer PC-1251	135
14.3 Bedienungs- und Kontroll-Einrichtungen	136
14.4 Betriebsvorbereitung	136
14.5 Auswechseln der Papierrolle	137
14.6 Einsetzen der Mikrokassette	138
14.7 Stromversorgung und Fehlermeldung 8	139
14.8 Druckerausgaben	140
14.9 Manuelles Drucken	140
14.10 LPRINT	142
LLIST	146

15. Datenspeicherung auf Magnetband	147
15.1. Benutzung des Magnetbandes.	149
15.2. Überprüfen der Abspeicherung und Zurückgewinnen der Information	150
15.2.1 Vorbereitung für das CLOAD?-Kommando	151
CSAVE	153
CLOAD	155
CLOAD?	156
MERGE	157
CHAIN	161
PRINT #	165
INPUT #	168
15.3 Rückladen von Programmen und Daten von einem externen Bandgerät	169
16. Programmbeispiele.	170
Anhang	181
A. ASCII-Tabelle.	182
B. Rechenbereich/Rechengenauigkeit	184
C. Fehlermeldungen.	185
D. Liste der Funktionen und Anweisungen	186
E. Anwendung der Kommandos, Anweisungen und Funktionen in den verschiedenen Betriebsarten.	193
F. Referenzliste BASIC	195
G. PC-1251 Technische Daten	197
H. CE-125 (Option) Technische Daten.	198
Programmbeispiele.	199

I. Einleitung

Der **PC-1251** hat echte Taschenrechnermaße bei einer Speicherkapazität, die früher bei dieser Gerätegröße noch nicht denkbar war.

Die Kombination von Leistungsmerkmalen der SHARP-Taschencomputer PC-1211 und PC-1500 macht diesen Rechner so interessant.

Besitzer des PC-1211 können ihre Programme auch auf dem **PC-1251** laufen lassen; dabei bietet der **PC-1251** trotzdem die Vorteile des erweiterten BASIC, Sie können damit den Anwenderbereich von 3,7 KByte RAM optimal nutzen.

Zusammen mit der Option **CE-125** – einer Mikrokassetten-Rekorder/Thermodrucker-Einheit – ist dieser Taschencomputer der ideale Begleiter für alle, die ein wirklich portables System benötigen.

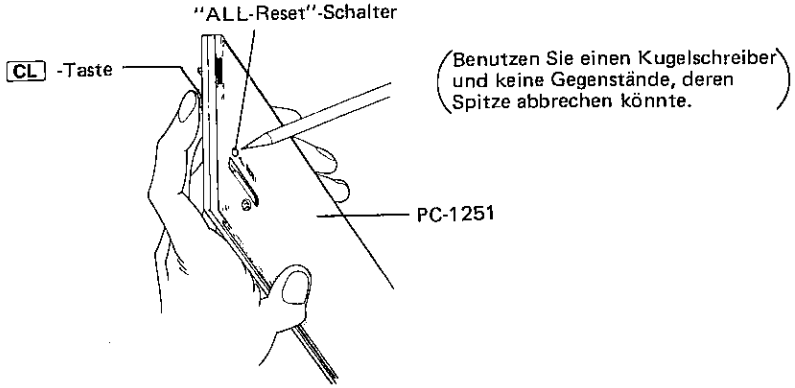
Die Bedienungsanleitung versucht sowohl dem Anfänger als auch dem geübten Programmierer gerecht zu werden. Dabei mußte man naturgemäß Kompromisse schließen. Wir empfehlen jedem Anfänger sich zusätzlich ein BASIC-Lehrbuch zuzulegen. An dieser Stelle sei insbesondere auf die Publikationen des Vieweg-Verlages verwiesen; eine Referenzliste mit genauen Titelangaben finden Sie im Anhang.

Die Bedienungsanleitung versteht sich in erster Linie nicht als eine Einführung in die Programmiersprache BASIC; sie soll die spezifischen Merkmale des **PC-1251/CE-125** herausarbeiten, die Wirkung und den Aufbau der einzelnen zugelassenen BASIC-Befehle erklären und damit die Grundlage für eine optimale Nutzung dieses Taschencomputers liefern. Wir empfehlen Ihnen, die Anleitung zunächst von Anfang an durchzuarbeiten; ihr Aufbau ist dabei so gehalten, daß sie später auch als Nachschlagewerk dienen kann. Wir hoffen, Ihnen damit die Arbeit mit Ihrem **PC-1251** so einfach und problemlos wie möglich gemacht zu haben.

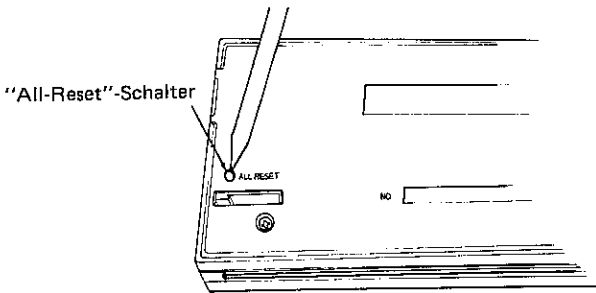
II. Betriebshinweise

1. Die Flüssigkristallanzeige des PC-1251 ist in Spezialglas eingebettet und kann daher bei Gewalteinwirkung zerbrechen. Behandeln Sie den Computer deshalb mit Sorgfalt. Beim Transport immer die Schutzhülle verwenden.
2. Schützen Sie den Computer vor Staub, Feuchtigkeit und allzu großen Temperaturschwankungen.
3. Zur Reinigung dient ein weiches, trockenes Tuch. Keine Reinigungs- oder Lösungsmittel verwenden.
4. Durch elektrostatische Entladungen über den Computer oder durch Fehlbedienung (der Computer blieb beim Batteriewechsel oder Anschluß der Option **CE-125/EA-23E** eingeschaltet) kann der **PC-1251** "abstürzen". Dadurch werden alle Tastenfunktionen blockiert einschließlich der **ON** -Taste. Sollte Ihnen dies passieren, müssen Sie den Computer **PC-1251** initialisieren. Dazu stehen zwei Möglichkeiten zur Wahl:

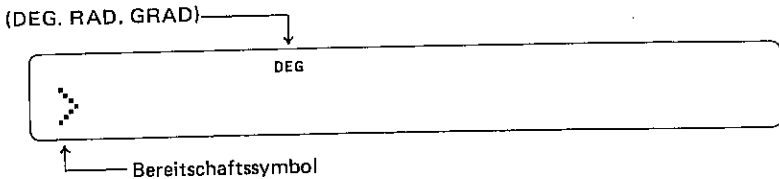
- 1) Sie drücken die Taste **CL** und betätigen dann den "ALL-RESET"-Schalter. Die Programme, Variablen und der Reservespeicher werden dabei nicht gelöscht.



- 2) Sollte die Blockade dadurch noch nicht aufgehoben sein, betätigen Sie den "ALL-RESET"-Schalter, ohne dabei die **CL** -Taste zu drücken. Alle Speicherinhalte werden dadurch gelöscht.



Auf der Anzeige erscheint:



III. Stromversorgung

Der Taschencomputer PC-1251 wird mit Lithiumbatterien betrieben. Die Batterien sind werksseitig bereits eingesetzt, bei erschöpften Batterien muß ein Wechsel wie folgt vorgenommen werden:

1. Sie schalten den Computer aus.
2. Lösen Sie die zwei Gehäuseschrauben auf der Rückseite des Geräts. (Abb. 1.)
3. Heben Sie den Deckel auf der Schraubenseite etwas an und schieben Sie ihn nach hinten.
4. Schieben Sie die Batterieabdeckung nach hinten; sie läßt sich dann abheben. (Abb. 2.)
5. Erneuern Sie die Batterien. (Abb. 3.)
Achten Sie auf richtige Polung (+, -)!
6. Sie setzen die Batterieabdeckung wieder ein und verriegeln sie.
7. Sie setzen die Klauen des Deckels in die entsprechenden Aussparungen am PC-1251, klappen ihn zu und befestigen ihn mit den Gehäuseschrauben. (Abb. 4.)
8. Sie schalten den PC-1251 ein, in der Anzeige steht:

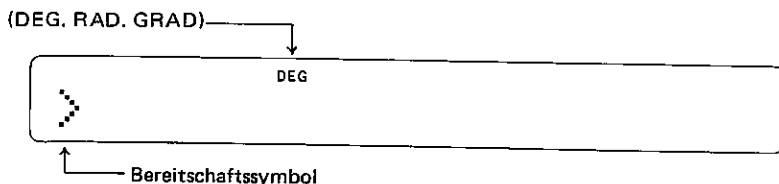


Abbildung 1.)

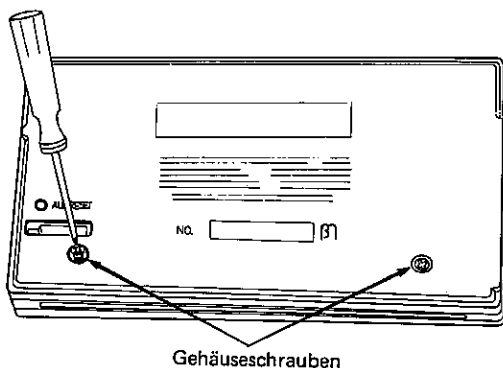


Abbildung 2.)

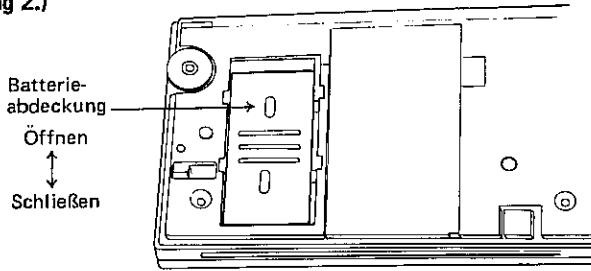


Abbildung 3.)

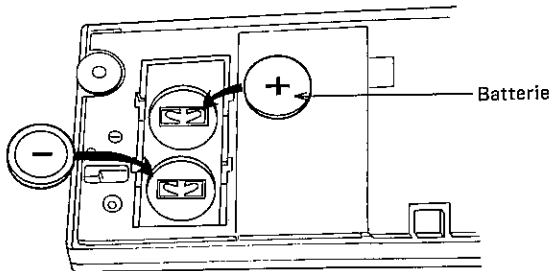
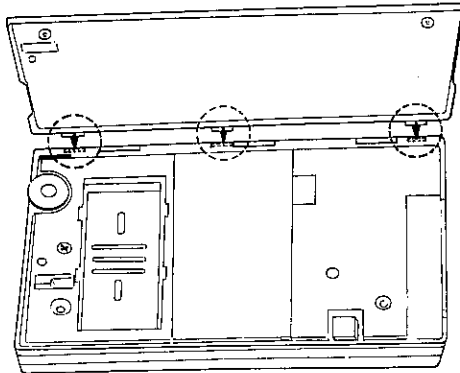


Abbildung 4.)



Wichtige Hinweise

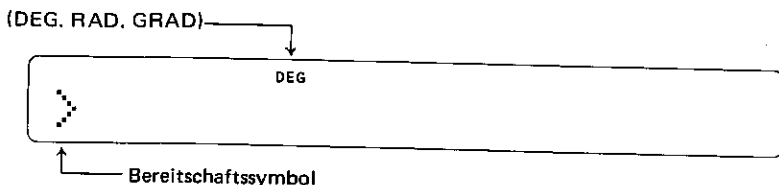
1. Wechseln Sie grundsätzlich immer beide Batterien vom gleichen Fabrikat (z.B. VARTA CR2032).
2. Entfernen Sie verbrauchte Batterien sofort aus dem Gerät.
3. Der Kontrast der LCD-Anzeige ist abhängig von der Batteriespannung und dem Blickwinkel. Bei neuen Batterien bzw. bei Verwendung der Option **CE-125/EA-23E** sollte der Kontrast der Anzeige mit dem Kontrastknopf nachgeregelt werden. Schwächerer Kontrast bedeutet auch weniger Stromverbrauch.
4. Zur Entlastung der Batterien wird die Stromversorgung des **PC-1251** von der Option **CE-125/EA-23E** übernommen, sofern ihre momentane Betriebsspannung höher ist.

1. Einschalten des Computers

Das Einschalten des **PC-1251** erfolgt zum einen über den Schiebeschalter rechts neben der Anzeige, zum anderen über die Taste **ON BRK**.

Über den Schiebeschalter wird mit dem Einschalten gleichzeitig die gewünschte Betriebsart eingestellt.

Auf der Anzeige erscheinen die folgenden Symbole



- 1) > Das Symbol zeigt an, daß der **PC-1251** betriebsbereit ist.
- 2) DEG, RAD, oder GRAD gibt die Einheit an, in der Winkel vom Computer verarbeitet werden.

Hat sich der Rechner automatisch ausgeschaltet, muß er über die Taste **ON BRK** wieder aktiviert werden. Das Zeilendisplay erscheint beim Einschalten unverändert auf der Anzeige.

2. Ausschalten des Computers

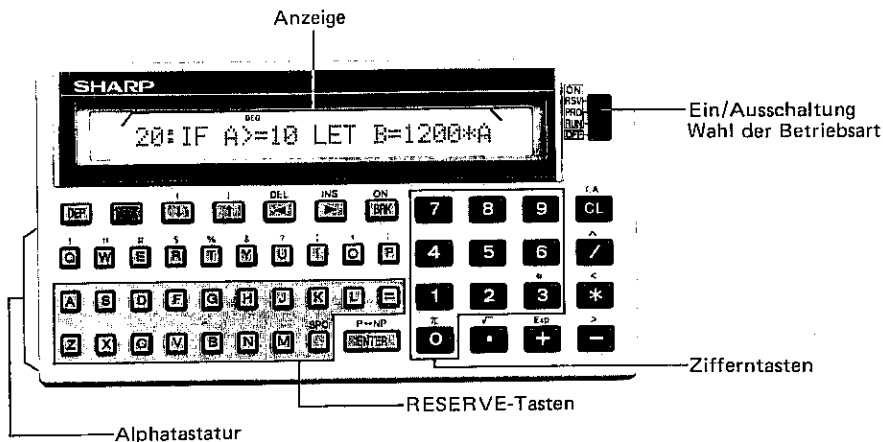
1. Automatisch

Wenn der Computer kein Programm abarbeitet, schaltet er sich ca. 11 Minuten nach der letzten Tastenbedienung automatisch aus. Dabei bleiben alle Daten- und Programminformationen gespeichert.

2. Manuell

Soll der Computer sofort abgeschaltet werden, muß der Schiebeschalter rechts neben der Anzeige auf OFF gestellt werden. Die Daten- und Programminformationen bleiben erhalten, jedoch wird das Zeilendisplay gelöscht. Der Computer läßt sich während der Programmausführung nicht ausschalten.

3. Die Bedienungselemente



Das Bedienungsfield des PC-1251 besteht aus 52 Tasten, die zu 3 Blöcken zusammengefaßt sind:

- die Schreibmaschinen-Alphatastatur incl. RESERVE-Tasten und **ENTER** -Taste
- in der oberen Reihe die Tasten mit Sonderfunktionen
- die Zifferntasten mit den mathematischen Operatoren in zwei Ebenen

Die Alphatastatur ist in der oberen Reihe mit zwei Ebenen belegt. Bei einfachem Tastendruck gelten die auf den Tasten stehenden Bezeichnungen; will man die über den Tasten stehenden Sonderzeichen abrufen, muß man vorher die Doppelfunktionstaste **SHIFT** drücken. Die Tasten A, S, D, F, G, H, J, K, L, =, Z, X, C, V, B, N, M, SPC dienen in der Doppelfunktion als Definable Keys bzw. als RESERVE-Tasten.

Über die Definable Keys können benannte Programme gestartet werden; man gibt dazu die Tastenfolge **DEF** < Definable Keys > ein. Die RESERVE-Tasten können über die Doppelfunktionstaste **SHIFT** < RESERVE-Taste > angesprochen werden.

ENTER beschließt jede Eingabe; über **SHIFT** **ENTER** kann man wählen, ob Druck oder Nichtdruck manueller Rechnungen gewünscht wird. Die Druckbetriebsart sollte nur gewählt werden, wenn die Option CE-125 auch tatsächlich angeschlossen ist.

In der Reihe über der Alphatastatur befinden sich Tasten mit Sonderfunktionen. **DEF** dient zum Start von Programmen über Definable Keys.

Bei Betätigen der Doppelfunktionstaste **SHIFT** erscheint zur Kontrolle der Schriftzug SHIFT in der Anzeige. Zweimaliges Drücken der **SHIFT** -Taste löscht den Doppelfunktionsbefehl. Die SHIFT-Funktion ist nur gültig für das unmittelbar folgende Zeichen.

DEF und **SHIFT** sind Wechselschalter.

4. Die Tastenfunktionen



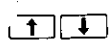
Schiebeschalter zum Ein- und Ausschalten des Rechners;
Wahl der Betriebsart RUN, PRO oder Reserve (RSV).
Der Rechner wird mit der Wahl einer der Betriebsarten automatisch auch eingeschaltet.



In Verbindung mit den Tasten A, S, D, F, G, H, J, K, L, =, Z, X, C, V, B, N, M, SPC ermöglicht diese Taste das Starten des Programmlaufs über einen Markennamen.



Doppelfunktionstaste



Editiertasten; Sonderfunktion



Cursor nach links; Editieraufruf



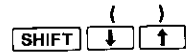
Cursor nach rechts; Editieraufruf



Unterbrechung des Programmlaufs. In der Anzeige erscheint BREAK IN < Zeilennummer > ; unterbricht auch Drucker- und Magnetbandbetrieb.



Löscht die Anzeige und die Blockade durch Fehlermeldungen.



Klammern



Löschen einzelner Zeichen



Platzhalter für nachträgliches Einfügen von Zeichen



ON; Einschalten des Rechners nach Abschalten durch die Automatik



Löschen der Anzeige; der Computer wird in folgenden Zustand gesetzt:

- WAIT-Intervall ist gelöscht
- Ausgabeformat ist gelöscht
- TRACE-Zustand ist gelöscht
- Fehlermeldung ist gelöscht



Alphatastatur; Variablennamen



Leerraum

ENTER

Beschließt die Eingabe von Programmzeilen, Reserve-Ausdrücken etc.; beim Rechnen ohne Programmunterstützung ersetzt es das "="; Programmstart in Verbindung mit RUN oder GOTO.

SHIFT **Q**



SHIFT **P**

Sonderzeichen (!, % haben reine Kommentarbeutung)

DEF **A**



DEF **SPC**

Definable Keys

SHIFT **A**



SHIFT **SPC**

Abruf der Reserve-Ausdrücke

SHIFT **ENTER**



Umschalter für Druck/Nichtdruck beim Rechnen ohne Programmunterstützung; in der Anzeige erscheint ein P

. **0**



9

Dezimalpunkt; numerische Zeichen

SHIFT **0**



Konstante PI

SHIFT **.**



Quadratwurzel

SHIFT **3**



Zeichen @ als Sonderfunktion im Reserve-Mode

+

-

/

Mathematische Operationszeichen; Sonderfunktion

SHIFT **+**



Exponent im wissenschaftlichen Format

SHIFT **>**

-

SHIFT **<**

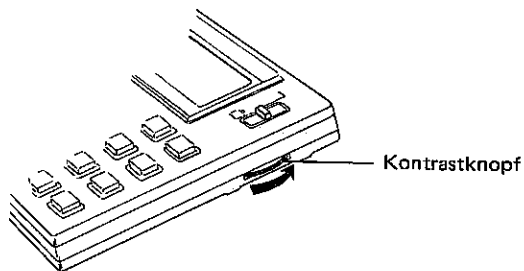
Vergleichsoperatoren

SHIFT **^**



Zeichen ^; Sonderfunktion; Exponentiation

Kontrastknopf



5. Die Anzeige

Der PC-1251 verfügt über eine 24-stellige alphanumerische Flüssigkristallanzeige. Die einzelnen Zeichen werden in einer 5 x 7 Punktmatrix dargestellt.

Das Eingaberegister faßt 80 Zeichen; alles, was über die 24 möglichen Zeichen der Anzeige hinausgeht, wird im Rollschreiberverfahren durchgezogen.

Der Cursor zeigt die Stelle, auf welche die nächste Information geschrieben wird. Er steht normalerweise auf der ersten freien Stelle und wird durch einen kurzen waagerechten Strich dargestellt.

Eine Ausnahme besteht, wenn auf der Anzeige links das Bereitschaftssymbol steht. Durch das nächste einzugebende Zeichen wird dieses überschrieben. Erst dann sieht man den Cursor auf der ersten freien Stelle nach dem zuerst eingegebenen Zeichen.

Rückt man den Cursor mit Hilfe der  oder  -Taste auf ein Zeichen, so erkennt man seine Position am Blinken an dieser Stelle.

Anhaltender Druck auf die Cursor-Tasten läßt den gesamten Inhalt des Eingaberegisters schnell in die gewünschte Richtung über die Anzeige rollen; ein kurzer Druck verschiebt die Anzeige nur um einen Schritt.

Versucht man mehr als 80 Zeichen einzulesen, werden diese vom Rechner nicht mehr angenommen. Jede zusätzliche Eingabe überschreibt das letzte eingegebene Zeichen. Der Cursor verändert hier sein Aussehen; er blinkt auf dem letzten Zeichen als Warnsignal in der vollen Punktmatrix.

In der Anzeige können folgende Symbole erscheinen:



- >** Das Bereitschaftssymbol. Es erscheint, wenn der Computer bereit ist, eine Eingabe aufzunehmen.
- Cursor. Der Cursor ist nicht zu sehen, wenn das Bereitschaftssymbol in der Anzeige steht.
- BUSY** BUSY erscheint während der Ausführung eines Programms oder einer Berechnung. Der Schriftzug verschwindet nach Beendigung der Ausführung oder bei Programmunterbrechungen. Solange BUSY angezeigt ist, ist die Abschaltautomatik ausgesetzt; auch nimmt der Computer keine weiteren Eingaben an.
- P** Das Symbol zeigt an, daß die Druckbetriebsart beim Rechnen ohne Programmunterstützung gewählt wurde.
- DEF** Der Schriftzug erscheint, wenn die **DEF** -Taste gedrückt wurde.
- DEG/
RAD/
GRAD** Anzeige der Winkleinheit
- SHIFT** Der Schriftzug erscheint, wenn die Doppelfunktionstaste betätigt wurde.
- E** Fehleranzeige, erscheint immer dann, wenn ein Fehler auftritt.

Der Kontrast der Anzeige kann durch den Regler an der rechten oberen Seite des Computers eingestellt werden.

6. Wahl der Betriebsart

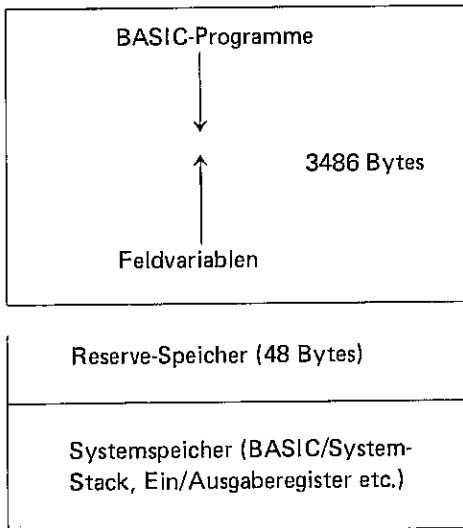
Der PC-1251 unterscheidet drei Betriebsarten:

- 1. RUN-Mode: der Computer kann manuell bedient werden und gespeicherte Programme können ausgeführt werden.
- 2. PROgramm-Mode: BASIC-Programme können eingegeben, verändert und gelistet werden.
- 3. RSV-Mode (Reserve-Mode): die 18 Tasten A, S, D, F, G, H, J, K, L, =, Z, X, C, V, B, N, M, SPC können mit Funktionen belegt werden.

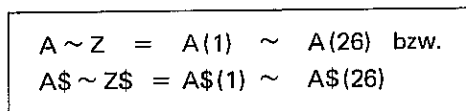
Die Wahl der Betriebsart erfolgt über den Schiebeschalter rechts neben der Anzeige.

7. Speicherorganisation

Hauptspeicher



Standardvariablenspeicher (208 Bytes)



8. Rechnen ohne Programmunterstützung

Der **PC-1251** kann auch wie ein normaler Taschenrechner verwendet werden. Dabei stehen alle gebräuchlichen mathematischen Funktionen zur Verfügung. Ist der Drucker **CE-125** (Option) angeschlossen, können alle Eingaben und Ergebnisse protokolliert werden.

Der **PC-1251** kann in der Programmiersprache BASIC programmiert werden. Zum Teil können Anweisungen und Funktionen dieser Sprache auch direkt eingegeben und ausgeführt werden.

Im RUN-Mode kann man den **PC-1251** auch wie einen einfachen Taschenrechner benutzen. Die allgemeine Form einer solchen Rechnung ist:

CL < numerischer Ausdruck > **ENTER**

Als < numerischer Ausdruck > wird jede mathematische Formel bezeichnet, die einen Zahlenwert als Ergebnis hat.

Beispiele:

a) Addition

CL

2 + 4

ENTER

Ausgabe

6.

b) Subtraktion

CL

2 - 4

ENTER

Ausgabe

-2.

c) Division

CL

5.5/2.7

ENTER

Ausgabe

2.037037037

j) Multiplikation

CL

2*3.1415

ENTER

Ausgabe

6.283

e) Benutzung von Klammern, Exponentiation

CL

27*(15+3)+1513

ENTER

Ausgabe

1999.

Hinweis:

1. Erst nach Eingabe von **ENTER** wird die Rechnung oder das Kommando ausgeführt oder das Ergebnis angezeigt.
2. Das Ergebnis einer Rechnung kann in der nächsten Rechnung weiter verwendet werden. Dazu wird die Zahl dann nicht mit der **CL** -Taste gelöscht, sondern als nächstes das Operationszeichen der neuen Formel eingegeben (+ - * / ^).

Beispiel:

a) **CL**

2*3.141527

ENTER

Ausgabe

6.283054

Eingabe: /180

6.283054/180

ENTER

Ausgabe

3.490585556E-02

8.1 Wissenschaftliche Notation

Man kann Zahlen in den **PC-1251** auch im wissenschaftlichen Format eingeben. Dadurch ist es möglich, Zahlen bis zu einer Größe von $\pm 9.999999999E \pm 99$ einzugeben und zu verarbeiten.

Wird dieser Wertebereich überschritten, erscheint eine Fehlermeldung (ERROR 2).

Ist der Betrag einer Zahl kleiner als $1.E-99$, so wird die Zahl auf Null gesetzt.

Wird eine Zahl mit einem mehr als 2-stelligen Exponenten eingegeben, so werden die überzähligen Ziffern ignoriert. Es werden dabei die beiden zuletzt eingegebenen Zahlen bewertet.

Hinweis:

1. Wie im englischen Sprachraum üblich wird nicht das Komma, sondern der Punkt zum Trennen von ganzem und gebrochenem Anteil von Zahlen benutzt.
2. Für den Operator "geteilt durch" wird ein Schrägstrich / anstelle des Doppelpunktes gesetzt.
3. Für "multipliziert mit" muß ein Stern * gesetzt werden.
4. Als Zeichen für die Exponentiation wird ein Dach ^ verwendet; $15 \wedge 3$ bedeutet z.B., daß die Grundzahl 15 mit 3 exponenziert wird. Ist die Basis negativ, muß sie in Klammern gesetzt werden. Als Exponenten sind bei negativen Grundzahlen nur ganze Zahlen zulässig. Ansonsten erscheint eine Fehlermeldung (ERROR 2).
5. Sind mehr als 10 Ziffern für die Darstellung eines Ergebnisses erforderlich, wird es im wissenschaftlichen Format angezeigt. Das Anzeigeformat kann durch die BASIC-Anweisung USING verändert werden.

8.2 Rechengenauigkeit

Der PC-1251 verarbeitet Zahlen mit einer Genauigkeit von 10 Stellen. Dies kann bei längeren Rechnungen zu Rundungsfehlern führen. Diese Genauigkeitsbeschränkung tritt auch beim Exponenzieren auf oder wenn Zahlen sehr unterschiedlicher Größe addiert werden.

Beispiel:

a) Eingabe

41^3

ENTER

Ausgabe

68920.99999

b) Eingabe

41^3-41*41*41

ENTER

Ausgabe

-0.0000055

c) Eingabe

1E12+1-1E12





ENTER





Ausgabe

0.



8.3 Kontrollieren und Verändern der Eingabe (Editieren)

Eventuell auftretende Tippfehler können beim **PC-1251** ebenso während der Eingabe korrigiert werden, wie man schon eingegebene Formeln überprüfen und verändern kann.




Das passiert mit Hilfe der Cursor-Tasten  und . Über diese Tasten bewegt man den Cursor nach rechts oder links, ohne daß eingegebene Formeln verloren gehen. Man kann jedes Zeichen verändern, indem man den Cursor darüberstellt und ein neues Zeichen eingibt. Will man ein Zeichen löschen, kann man das über die Tasten  . Das über dem Cursor stehende Zeichen wird gelöscht.

Zum Einfügen von Zeichen betätigt man die Tasten  . Es wird ein Platzhalterzeichen  in die Formel vor dem Zeichen eingefügt, das über dem Cursor steht. Das Platzhalterzeichen kann dann durch ein neues Zeichen überschrieben werden. Mit  werden alle überflüssigen Platzhalter aus der Formel gelöscht.

8.4 Playback

Ist das Ergebnis einer Berechnung schon ausgegeben oder erschien beim Ausrechnen eine Fehlermeldung, kann die Ausgangsformel durch die Tasten  oder  in die Anzeige zurückgeholt und wie oben beschrieben geändert werden.

Hinweis:

BASIC-Schlüsselwörter wie LET, SIN, COS, PRINT etc. werden nach Betätigung der -Taste intern abgekürzt. Obwohl das Schlüsselwort unverändert auf der Anzeige erscheint, wird es beim Editieren durch ein einziges neues Zeichen vollständig überschrieben. Hat man sehr lange Programmzeilen einzugeben, kommt man zunächst nur bis zum 80. sten Zeichen. Geben Sie dann  ein und fahren mit dem Cursor auf das Ende der Zeile. Sie können dann noch zusätzliche Zeichen eingeben. Die Eingabe muß wieder mit  abgeschlossen werden.

8.5 Mathematische Funktionen

Der PC-1251 verfügt über alle gebräuchlichen mathematischen Funktionen. Diese werden jedoch nicht über spezielle Tasten abgerufen, sondern über die Buchstabetasten in der Form eingegeben, wie sie in mathematischen Formeln gebräuchlich ist.

Beispiel:

SIN (2 * PI)

CL S I N (2 * P I) ENTER

Hinweis:

Rechnet man häufig mit bestimmten Funktionen, kann man zur Vereinfachung der Eingabe die "reservierbaren Tasten" verwenden (s. RESERVE-Mode).

8.6 Klammerregeln

Die Reihenfolge der einzelnen Schritte beim Ausrechnen eines komplexen numerischen Ausdrucks wird durch Klammern geregelt.

Wie in der Mathematik gibt es dabei eine implizite Klammerung, d. h. daß bestimmte Operationen vor anderen Vorrang haben. Der **PC-1251** verfügt über 15 Klammerebenen.

Beispiel:

$$3 * 5 + 7 * 4$$

ist gleichbedeutend mit

$$(3 * 5) + (7 * 4)$$

Die **Rangfolge** der mathematischen Operatoren ist:

1. Abruf von PI, MEM, INKEY\$ und der Variablen
2. Exponentiation, wenn eine Multiplikation ohne $\boxed{*}$ -Befehl vorausgeht.
z. B. $2 A^3 = 2 * (A^3)$
3. Multiplikation ohne $\boxed{*}$ -Befehl;
z. B. 2A, TIB, AB
4. Funktionsoperationen in Bezug auf das folgende Argument;
z. B. $SIN 3 + 4 = (SIN 3) + 4$
5. Exponentiation;
z. B. $2 * A^3 = 2 * (A^3)$
6. Vorzeichen + und -
7. Multiplikation * und Division /
8. Addition + und Subtraktion -
9. Vergleichsoperatoren
10. Logische Operatoren

Hinweise:

1. Werden in einem numerischen Ausdruck mehrere Operationen gleichen Ranges nebeneinander gestellt, so werden sie von links nach rechts ausgeführt; z. B.

$$24/3/2 = (24/3)/2$$

2. Ausdrücke in Klammern werden immer zuerst berechnet.
Bei geschachtelten Klammern wird der innerste Ausdruck zuerst berechnet.

8.7 Sedezimalzahlen

Natürliche Zahlen im Bereich zwischen 0 und 65535 können in den PC-125, auch als Sedezimalzahlen eingegeben werden. Der Zahl wird dann ein & vorangestellt. Zur Darstellung dienen die Zahlen 0 bis 9 und die Buchstaben A ~ F.

Beispiel:

a) Eingabe

& A

ENTER

Ausgabe

10.

b) Eingabe

& 10

ENTER

Ausgabe

16.

ENTER

c) Eingabe

& FFFF

Ausgabe

65535.

- d) Die Ausgabe von Zahlen in Sedezimalform ist auf dem **PC-1251** nicht vorgesehen, kann aber durch das folgende Programm gelöst werden:

```
10: DIM H$(0)
20: INPUT "DEZIMALZAHL?"
   ;N
30: N= INT N: A$=""
40: H$(0)="0123456789ABC
   DEF"
50: M=N:N= INT (N/16)
60: M=M-N*16+1
70: A$= MID$( H$(0),M,1)
   +A$
80: IF N>0 THEN 50
90: PRINT "HEXZAHL=" ; A$
```

Eingabe

RUN

ENTER

Ausgabe bei Zeile 20

DEZIMALZAHL?_

Eingabe z. B.

DEZIMALZAHL? 10

ENTER

Ausgabe bei Zeile 90

HEXZAHL=A

8.8 Textausdrücke

Textausdrücke sind Bestandteile der BASIC-Sprache. Sie können auch im RUN-Mode ohne Programmunterstützung eingegeben werden.

Man unterscheidet Textkonstanten, Textvariablen, Textfunktionen und zusammengesetzte Texte. Eine Textkonstante ist eine beliebige Zeichenfolge, die durch Anführungszeichen begrenzt ist. (Die Anführungszeichen sind nicht Bestandteil des Textes).

Beispiel:

CL

Eingabe

"SHARP"

ENTER

Ausgabe

SHARP

Textvariablen können bis zu 7 Zeichen enthalten. Über die DIM-Anweisung lassen sich Textvariablen mit Längen bis zu 80 Zeichen erzeugen.

Beispiel:

Eingabe

A\$ = "SHARP"

ENTER

Ausgabe

SHARP

Texte können mit dem "+"-Zeichen aneinandergesetzt werden.

Beispiel:

Eingabe

A\$ = "POCKET"

ENTER

Ausgabe

POCKET

Eingabe

A\$ + " _ COMPUTER "

ENTER

Ausgabe

POCKET COMPUTER

Hinweis:

Auch das Leerzeichen (Space) ist innerhalb eines Textes ein gültiges Zeichen.

8.9 Vergleichsausdrücke

Der PC-1251 kennt folgende Vergleichsoperatoren:

< kleiner	<= kleiner oder gleich	= gleich	>= größer oder gleich	> größer	<> ungleich
--------------	------------------------------	-------------	-----------------------------	-------------	----------------

Mit diesen Operatoren können zwei numerische oder Textausdrücke miteinander verglichen werden. Ist die Vergleichsaussage richtig, liefert der Rechner das Ergebnis 1, ist sie falsch, ist das Ergebnis 0. Vergleichsausdrücke erhalten besondere Bedeutung im Zusammenhang mit der BASIC-Anweisung IF.

Beispiele:

a) Eingabe

CL

2 < 1

ENTER

Ausgabe

0.

b) Eingabe

CL

$1 \leq 2$

ENTER

Ausgabe

1.

c) Eingabe

CL

$\text{SIN } 1 < .7$

ENTER

Ausgabe

0.

(im RAD-Mode)

1.

(im DEG- und GRAD-Mode)

d) Eingabe

CL

"WILLI" > "ANTON"

ENTER

Ausgabe

1.

Die Textausdrücke WILLI und ANTON werden entsprechend dem ASCII-Code auf die lexikographische Reihenfolge hin überprüft.

Hinweis:

Da das Ergebnis eines logischen Ausdruckes eine Zahl ist, kann es in einer numerischen Variablen gespeichert werden.

9. Sprachelemente

9.1 Numerische Konstante

Eine numerische Konstante kann sein

- eine ganze Zahl
- eine gebrochene Zahl
- eine Zahl in wissenschaftlicher Darstellung
- eine Sedezimalzahl

Beispiele:

513

-2376

11.745

1.23456789E-12

&FFFF

&AA2B

&1

9.2 Textkonstante

Eine Textkonstante ist eine beliebige Zeichenfolge, die durch Anführungszeichen eingeschlossen wird.

Beispiel:

"SHARP"

" " (Textkonstante enthält Leerzeichen)

" " (Textkonstante der Länge 0)

9.3 Numerische Variable

Unter einer numerischen Variablen versteht man den Speicherplatz für einen Zahlenwert, auf den über den Variablennamen zugegriffen werden kann.

Der Variablenname besteht aus einem Buchstaben oder einem Buchstaben mit einem in Klammern folgenden Indexwert. Der Index kann ebenfalls durch eine Variable dargestellt werden.

Beispiele:

A
A(27)
B(1)
A(Z)

9.4 Textvariablen

Unter einer Textvariablen versteht man den Speicherplatz für eine Zeichenfolge, auf die über den Textvariablennamen zugegriffen wird. Der Name hat die gleiche Form wie der der numerischen Variablen mit einem zusätzlichen \$-Zeichen hinter dem Buchstaben.

Eine Textvariable kann maximal 7 Zeichen enthalten; benötigt man eine größere Zeichenkapazität, muß man über die DIM-Anweisung ein Textfeld vereinbaren (s. Kapitel: Variablen).

Beispiele:

A\$
B\$(Ø)

9.5 Numerische Funktion, Textfunktion

Eine Funktion ist ein Operator auf ein oder mehrere Parameter und hat einen Zahlenwert oder eine Zeichenfolge als Ergebnis.

Die Parameter werden hinter den Funktionsnamen gestellt und können je nach Funktion numerische oder Textausdrücke sein.

Sind mehrere Parameter erforderlich, so werden diese in Klammern gesetzt und durch Kommata getrennt.

Beispiel:

- LN 60 numerische Funktion; natürlicher Logarithmus des numerischen Parameters 60
- ASC "A" numerische Funktion; Ergebniswert ist die numerische Konstante 65, die den ASCII-Code der Textkonstante A darstellt.
- CHR\$ 65 Textfunktion; Ergebniswert ist die Textkonstante A
- MID\$ ("SHARP", 2, 2) Textfunktion; Ergebniswert ist die Textkonstante HA

Hinweis:

Funktionsoperatoren haben eine höhere Priorität als andere Operatoren.

Beispiel:

$$\text{SIN } A + B = (\text{SIN } A) + B$$

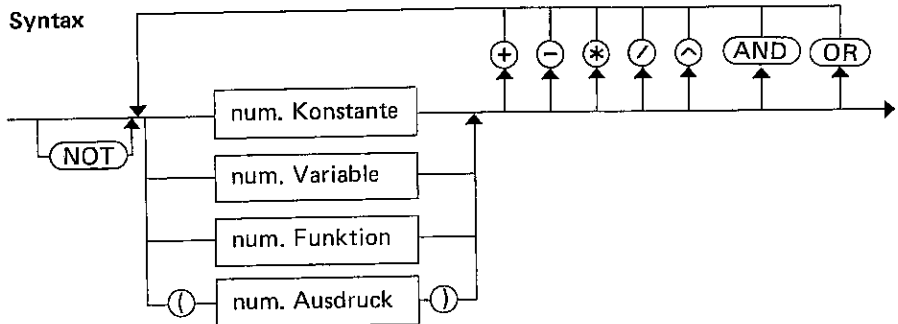
Soll der Sinus der Summe gebildet werden, muß geklammert werden:

$$\text{SIN } (A + B)$$

9.6 Numerischer Ausdruck

Ein numerischer Ausdruck besteht aus einer numerischen Konstante, Variablen oder numerischen Funktionen oder deren Verknüpfung durch die arithmetischen Operatoren + - * / ^ AND OR NOT und die Zusammenfassung durch Klammern.

Syntax



Beispiele:

15

(-8)

SIN 45

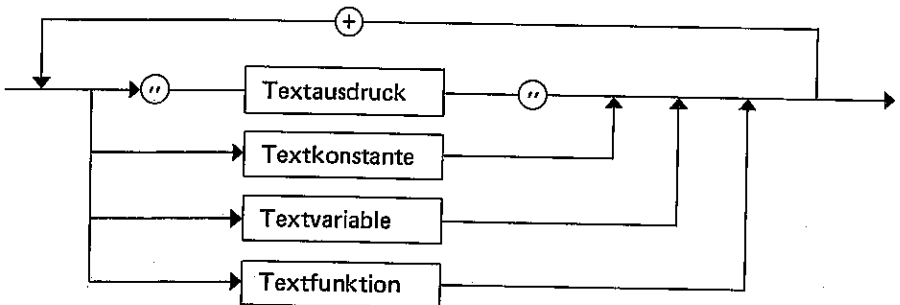
A + B/C

(C * (A * X + *B) * 5 / (D * C)) + 10

9.7 Textausdruck

Ein Textausdruck besteht aus einer Textkonstanten, Textvariablen oder einer Textfunktion und deren Verknüpfung durch das Zeichen +. Das Ergebnis eines Textausdrucks ist eine Zeichenfolge.

Syntax



Beispiele:

"A" + STR\$ A

A\$ + "PC"

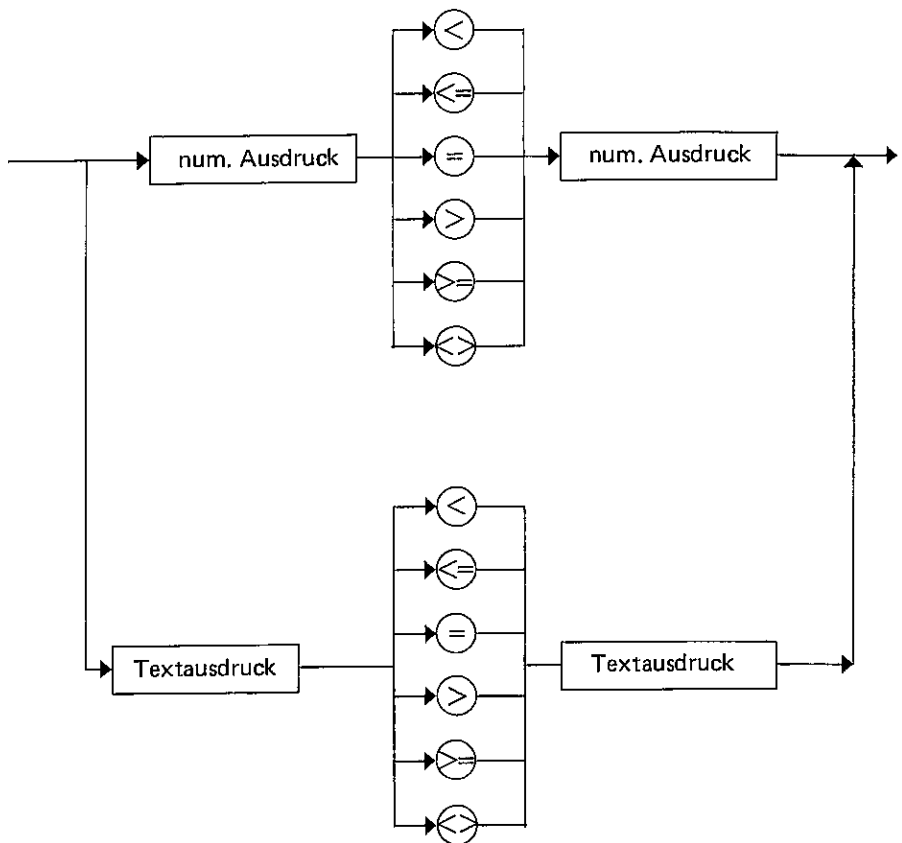
9.8 Logischer Ausdruck

Ein logischer Ausdruck ist der Vergleich zweier Ausdrücke durch die Operatoren

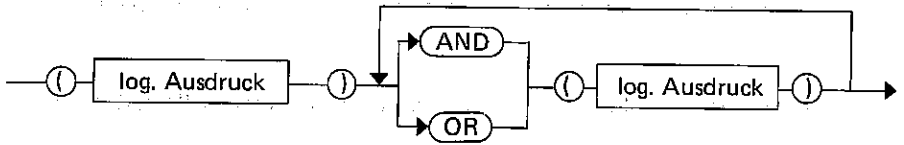
$<$, \leq , $=$, \geq , $>$, $<>$

oder die Verknüpfung solcher Vergleiche durch die Operatoren AND, OR, NOT.

Syntax



Logische Ausdrücke lassen sich wie folgt verknüpfen:



Ist die Vergleichsaussage richtig, ist das Ergebnis des logischen Ausdrucks eine 1, ist sie falsch ist das Ergebnis \emptyset .

Werden Textausdrücke miteinander verglichen, so wird auf die lexikographische Reihenfolge geprüft; diese richtet sich beim PC-1251 nach dem ASCII-Code (Anhang).

Haben zwei Textausdrücke eine unterschiedliche Länge, so wird bei einem Vergleich der kürzere Ausdruck mit dem ASCII-Zeichen Null aufgeführt.

Logische Ausdrücke werden in der BASIC-Anweisung IF verwendet.

Beispiel:

	Ergebnis
$1 < 2$	1
$1+2+3 < 2+3+4$	1
$1 > = 2$	\emptyset
$1 < 2 < 3$	1
"ANTON" < "WILLI"	1
"ANTON" = "ANTON 1"	\emptyset ("1" ist größer als ASCII-Null)
$(1 < 2) \text{ AND } (3 < 4)$	1
$(\text{"ANTON" < "WILLI"}) \text{ OR } (\text{"ANTON" = "ANTON 1"})$	1

10. Variablen

Variablen sind Bezeichner für Größen, deren Wert beim Schreiben eines Programms noch nicht bekannt ist. Sie werden beim Rechner durch Speicher realisiert, denen man unterschiedliche Daten durch Einlesen oder durch Auswerten eines bestimmten Ausdrucks zuordnen kann. Je nachdem, ob es sich um numerische oder Textwerte handelt, nennt man solche Variablen numerische Variablen oder Textvariablen.

Zur Unterscheidung zwischen numerischen und Textvariablen muß jeder Textvariablenname mit dem Zeichen \$ enden. Beide Typen können als einfache oder indizierte Variablen geschrieben werden. Zur leichteren Identifizierung erhalten die Variablennamen als symbolische Adresse eines Speicherplatzes. Mit Variablen lassen sich Gesetzmäßigkeiten unabhängig von ihrem jeweiligen Wert allgemein ausdrücken.

10.1 Variablen im PC-1251

Wie in der Graphik "Speicherorganisation" dargestellt, hat der **PC-1251** zwei verschiedene Speicherbereiche für Variablen:

- a) den Standardvariablenspeicher
- b) den Feldvariablenspeicher

Der Standardvariablenspeicher ist ein begrenzter Speicherbereich und dient ausschließlich zur Aufnahme von Werten der 26 Standardvariablen.

Der Feldvariablenspeicher ist Bestandteil des Hauptspeichers, der auch zur Aufnahme von Programmen dient. Die Kapazität des Hauptspeichers kann wahlweise für Programme und für Feldvariablen benutzt werden.

Folgende Variablentypen stehen im **PC-1251** zur Verfügung:

- 1) Standardvariablen
 - a) einfache numerische Variable
 - b) einfache Textvariable
 - c) indizierte numerische Variable
 - d) indizierte Textvariable
- 2) Feldvariablen
 - a) indizierte eindimensionale numerische Feldvariablen (Vektoren).
 - b) indizierte eindimensionale Textfeldvariablen (Vektoren).
 - c) indizierte zweidimensionale numerische Variablen (Matrizen)
 - d) indizierte zweidimensionale Textfeldvariablen (Matrizen)

10.2 Standardvariablen

26 Standardvariablen stehen aufgrund eines hierfür reservierten Speicherbereichs immer zur Verfügung. Sie können wahlweise als numerische oder als Textvariable aufgerufen werden.

Eine Standardvariable kann also zur Zeit entweder einen numerischen oder Textwert aufnehmen. Sie kann allerdings direkt überschrieben werden, d.h. wenn ein numerischer Wert zugeordnet war, kann trotzdem ein Textwert eingegeben werden. Dabei geht der ursprüngliche numerische Wert verloren.

Bei der Wertabfrage muß der Variablenname dem Inhalt dieser Variablen entsprechen. Wird sie als numerische Variable aufgerufen, ihr Inhalt ist aber ein Textwert, so wird ERROR 9 angezeigt und umgekehrt.

10.3 Einfache Variablen werden mit den Namen A bis Z als numerische Variable bzw. A\$ bis Z\$ als Textvariable aufgerufen.

Einer numerischen Einzelvariablen kann man eine vollständige Zahl, bestehend aus zehnstelliger Mantisse, zweistelligem Exponenten und zwei Vorzeichen zuweisen.

Zum Beispiel:

A = 123: A = SIN X : A = B + C: A = B * C

Einer Textvariablen kann man Texte mit 7 Zeichen, bestehend aus Buchstaben, Zahlen, Sonderzeichen und Leerstellen zuweisen.

Zum Beispiel:

B\$ = "TEXT": B\$ = "T E X T"

Bei der Wertzuweisung muß der Text in Anführungsstrichen stehen.

10.4 Indizierte Variablen werden aufgerufen mit den Namen A (1) bis A(26) als numerische Variable bzw.

A\$(1) bis A\$(26) als Textvariable.

Die indizierten Standardvariablen sind äquivalent zu den oben beschriebenen einfachen Standardvariablen.

So entspricht:

die einfache Variable A	der indizierten Variablen	A(1)	
"	B	"	A(2)
"	C	"	A(3)
"	Z	"	A(26)

Das gilt sowohl für numerische als auch für Textvariablen. Die indirekte Adressierung einer Variablen gestattet es, den "Namen", die "Adresse" der Variablen in Abhängigkeit vom Wert einer anderen Variablen, der ein Rechenergebnis sein kann, festzulegen; bzw. kann man den Index einer Variablen durch einen numerischen Ausdruck festlegen.

Beispiele:

a) LET A(A) = 1234

Es wird eine numerische Variable definiert, deren Index gleich dem ganzzahligen Anteil der im Speicher A stehenden Zahl ist.

Angenommen der Wert der Variablen A ist 7, so wird der Variablen A (7), also der Variablen G, der Wert 1234 zugewiesen.

b) LET A(B/5) = 789

Es wird eine numerische Variable definiert, deren Index gleich dem ganzzahligen Anteil des Quotienten aus B/5 ist.

Angenommen der Wert der Variablen B ist 134, so ist der Quotient aus $134/5 = 26,8$. A(26) entspricht der Variablen Z.

c) LET A\$(A) = "TEXT"

Der Wert der numerischen Variablen A definiert den Index für die Textvariable A\$.

Der Wert der Variablen A darf dabei nicht 0 sein, weil die Variable A\$(0) nicht existiert. Er darf aber auch nicht 1 sein, weil A\$(1) ja A\$ entspricht und A\$ und A dürfen, wie bereits erklärt, nicht nebeneinander definiert werden.

10.5 Besonderheiten der Variablen A

Die Variable A kann als:

- a) Standardvariable A bzw. A\$
- b) Indizierte Standardvariable A(n) bzw. A\$(n)
n = 1 bis 26
- c) Eindimensionale Feldvariable A(n) bzw. A\$(n)
n = 27 bis 255

definiert werden.

Dabei gelten folgende Grundsätze:

- 1) Den A-Variablen können nur numerische Werte mit zehnstelliger Mantisse, zweistelligen Exponenten und zwei Vorzeichen bzw. Textwerte mit max. 7 Zeichen zugewiesen werden. Sie können nur alternativ als numerische oder Textvariable definiert werden.

Sie können überschrieben werden durch z.B.

LET A(n) = 123 LET A\$(n) = "TEXT". n = 30

Durch die zweite Wertzuweisung wird der numerische Wert überschrieben und geht verloren.

- 2) Der Variablenname A(0) ist unzulässig.
- 3) Für indizierte Variable A(n), (n = 1 bis 26) = Standardvariablen A bis Z wird kein Speicherplatz im Hauptspeicher reserviert.
- 4) Für indizierte Variable A(n), (n = 27 bis 255) = eindimensionale Feldvariable wird ein Speicherplatz im Hauptspeicher reserviert, jedoch ist keine DIM-Anweisung notwendig (aber möglich). Der Platz wird automatisch durch den höchsten definierten Indexwert reserviert. Wird z.B. A(n), n = 50, aufgerufen, ist der A-Vektor mit den Elementen A(27) bis A(50) automatisch vereinbart.
- 5) Der A-Vektor wird bezüglich seiner Elementanzahl automatisch auf den Wert begrenzt, den er vor der Dimensionierung eines weiteren Vektors bzw. Matrix hatte.

Beispiel:

5: A (27) = 27
10: DIM B (10)
15: DIM A (40)
bzw. 15: A (40) = 40

Bei Zeile 15 wird ERROR 3 angezeigt. Der A-Vektor wurde durch die Dimensionierung des B-Vektors auf ein Element, also A(27) begrenzt. Durch den Aufruf von A(40) in Zeile 15 müßte ein zweiter A-Vektor definiert werden, was aber unzulässig ist.

- 6) Variablen A(n) bzw. A\$(n) (n = 1 bis 26) können nicht gelöscht werden. Durch das Kommando NEW oder CLEAR wird ihr Wert auf 0 (Null) bzw. auf das ASCII – Nullzeichen gesetzt. Variablen A(n) bzw. A\$(n) (n = 27 bis 255) werden durch das Kommando RUN gelöscht.

10.6 Feldvariable

Feldvariable sind im Gegensatz zu den Standardvariablen flexibel.

Sie können als eindimensionale Felder (Vektoren) oder als zweidimensionale Felder (Matrizen) definiert werden, die wiederum unterschiedlich viele Elemente haben können.

Feldvariablen werden im Hauptspeicher abgelegt. Wegen ihrer flexiblen Länge muß daher vor ihrem ersten Aufruf ein Speicherbereich reserviert werden. Einzelheiten siehe unter DIM.

Der Speicherplatzbedarf für eine Feldvariable setzt sich zusammen aus:

- 6 BYTES für den Variablennamen
- 8 BYTES für ein numerisches Element
- 16 BYTES für ein Textelement im Standardformat (16 Textzeichen)

10.7 Numerische Feldvariablen können sowohl als Vektoren als auch als Matrizen definiert werden. Jedem Element der Vektoren und Matrizen kann eine vollständige Zahl, bestehend aus einer zehnstelligen Mantisse, zweistelligen Exponenten und zwei Vorzeichen zugewiesen werden.

Vektoren werden mit Vektornamen und dem Index für ein Element aufgerufen.

Beispiel:

B (Ø) : C (3) : Z (255)

Matrizen werden mit dem Matrixnamen und den beiden Indizes für ein Element aufgerufen.

Beispiel:

D (2, 4) : E (1Ø, 1Ø) : Y (2, 1ØØ)

Insgesamt stehen für Vektoren und Matrizen folgende Feldnamen zur Verfügung:

- a) Vektoren B (n) bis Z (n) n = Ø bis 255
- b) Matrizen B (n, n) bis Z (n, n)

Der Index für die Vektoren- und Matrixelemente kann als numerische Konstante oder durch Variablen definiert werden.

Beispiel:

B (6) bzw. B (A)

C (3, 5) bzw. C (A, B)

Der Index wird durch den jeweiligen Wert der Variablen ausgedrückt.

Vektoren und Matrizen dürfen nicht mit dem gleichen Feldnamen definiert werden die Namen B (n) und B (n, n) z.B. sind also nicht zulässig; auch dürfen Vektornamen und Matrixnamen nicht zweimal definiert werden.

10.8 Textfeldvariable kann man sowohl als Vektoren als auch als Matrizen definieren. Jedem Element der Vektoren und Matrizen können im Standardformat Texte bestehend aus 16 Textzeichen zugewiesen werden. Durch eine entsprechende Dimensionierung kann die Größe der Elemente zwischen 1 und 80 Textzeichen definiert werden, z.B. mit DIM B\$ (10) * 4. In diesem Fall können die 11 Elemente des Vektors B jeweils nur 4 Textzeichen aufnehmen.

Insgesamt stehen für Vektoren und Matrizen folgende Feldnamen zur Verfügung:

- a) Vektoren B\$ (n) bis B\$ (n) n = 0 bis 255
- b) Matrizen B\$ (n, n) bis Z\$ (n, n)

Vektoren werden mit dem Vektorennamen und dem Index für ein Element aufgerufen.

Beispiel:

B\$ (0) : C\$ (3) : Z\$ (255)

Matrizen werden mit dem Matrixnamen und den beiden Indizes für ein Element aufgerufen.

Beispiel:

D\$ (2, 4) : E\$ (10, 10)

Die Indizes für die Vektor- und Matrixelemente können als numerische Konstante oder als numerische Variablen definiert sein. Z.B. B\$ (0) : C\$ (A) : D\$ (2, 4) und E\$ (E, F), wobei der Index durch den Wert der jeweiligen Variablen bezeichnet wird.

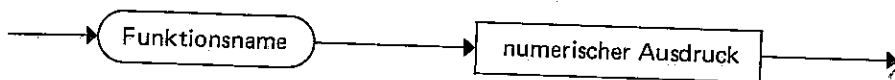
Vektoren und Matrizen dürfen nicht mit dem gleichen Feldnamen definiert werden, die Namen B\$ (0) und B\$ (A, B) sind also nicht zulässig, auch dürfen Vektorennamen und Matrixnamen nicht zweimal definiert werden.

11. Numerische Funktionen

Der PC-1251 bietet eine große Anzahl von mathematischen Standardfunktionen. Dazu gehören die trigonometrischen und ihre Umkehrfunktionen, natürlicher und dekadischer Logarithmus und Umkehrfunktionen etc.

Die Ergebnisse der Funktionen sind numerische Werte, als Parameter ist im allgemeinen ein numerischer Ausdruck zugelassen.

Syntax



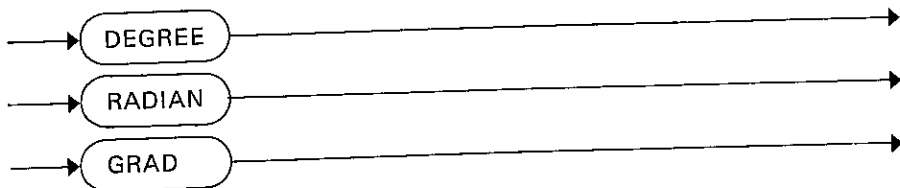
Die Funktionsnamen sind SIN, COS etc.; sie müssen über die Tastatur buchstabenweise eingegeben werden. Einfacher ist aber der Zugriff über die reservierbaren Tasten (s. Reserve-Mode)

Winkleinheitskommandos:

DEGREE/RADIAN/GRAD

Durch die Kommandos wird festgelegt, in welcher Winkleinheit die Parameter der trigonometrischen Funktionen verarbeitet werden.

Syntax



Die Winkel werden wie folgt angegeben:

DEGREE	($0 \sim 90^\circ$)
GRAD (GON)	($0 \sim 100^\text{g}$)
RADIAN	($0 \sim \text{PI}/2$)

(für jeweils einen Viertelkreis)

Die gewählte Winkleinheit erscheint oben in der Anzeige.

Hinweis:

Das Maß der Winkleinheit kann auch programmgesteuert verändert werden.

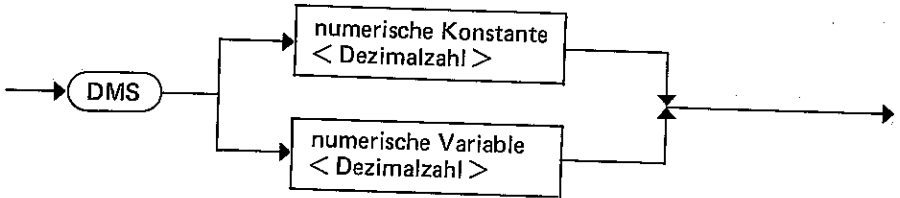
Beispiel:

- 1) 10 DEGREE : PRINT SIN 45
- 20 GRAD : PRINT SIN 50
- 30 Radian : PRINT SIN (PI/8)

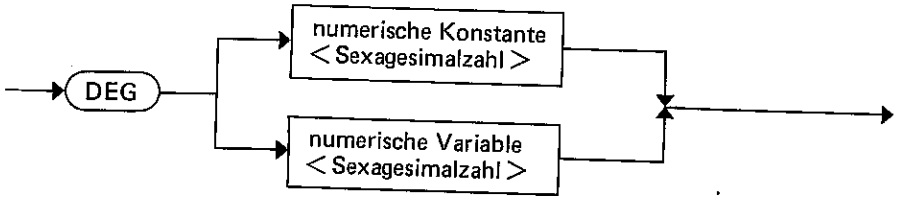
DMS/DEG

Die Funktionen ermöglichen die Umrechnung vom Sexagesimalsystem ins Dezimalsystem und umgekehrt.

Syntax

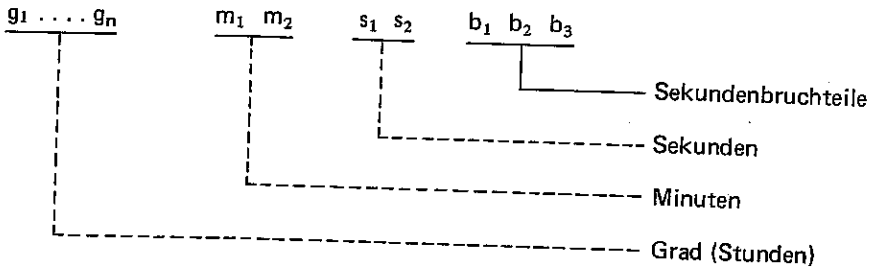


Das Funktionsergebnis ist eine Sexagesimalzahl.



Das Funktionsergebnis ist eine Dezimalzahl.

Sexagesimal geteilte Winkel oder Zeiten werden in folgendem Format verarbeitet:



Beispiel:

- a) Umrechnung von 15.4125 (Grad) in Grad/Minuten/Sekunden
Umrechnung von 15°24'45" in Dezimalgrad

```
10 A=DMS 15.4125
20 PRINT A
30 B=DEG 15.2445
40 PRINT B
50 END
```

Ausgabe bei Zeile 20

15.2445

ENTER

Das Ergebnis entspricht 15 Grad, 24 Minuten, 45 Sekunden.

Ausgabe bei Zeile 40

15.4125

Das Ergebnis entspricht 15.4125 Dezimalgrad.

Trigonometrie und inverse trigonometrische Funktionen: SIN, COS, TAN, ASN, ACS, ATN

Die Funktionen ermitteln den Sinus, Cosinus, Tangens und deren Umkehrwerte.

Beispiel:

```
a) 10 DEGREE : PRINT "X", "Y=SIN X"
    20 FOR X=0 TO 6
    30 LET Y=SIN X
    40 PRINT "X = "; X; "  "; "SIN X = "; Y
    50 NEXT X
    60 END
```

Ausgabe bei Zeile 40

X = 0. SIN X = 0.

ENTER

X = 1. SIN X = 1.745240

ENTER

bis

X = 6. SIN X = 1.045284

11.1 Mathematische Funktionen

LN	Natürlicher Logarithmus (Basis e)
LOG	Dekadischer Logarithmus (Basis 10)
EXP	Exponentiation zur Basis e
ABS	Absolutbetrag
INT	Ganzzahliger Anteil
SGN	Signum (Vorzeichen)
SQR ($\sqrt{\quad}$)	Quadratwurzel
PI (π)	Konstante PI

LN/LOG

Die Funktionen berechnen den natürlichen Logarithmus LN bzw. den Zehnerlogarithmus LOG.

Für beide Logarithmusfunktionen darf der Wert des logarithmischen Ausdrucks nicht negativ sein (ERROR 2).

Beispiel:

```
10 A = LN 100  
20 B = LOG 100  
30 PRINT A, B
```

Ausgabe bei Zeile 30

4.605170186

2.

EXP

Exponentialfunktion $e = \text{EXP } X$

Die Basiszahl e ist als $\text{EXP } 1 = 2.718281828$ gespeichert.

Beispiel:

```
10 A = LN 100  
20 B = EXP A  
30 PRINT B
```

Ausgabe bei Zeile 30

100.

ABS

Die ABS-Funktion ermittelt den Absolutbetrag eines numerischen Ausdrucks.

Beispiel:

```
a) 10 A = ABS (-6)
    20 B = ABS (6)
    30 C = ABS (-3 * 7)
    40 PRINT A;" ";B;" ";C
```

Ausgabe bei Zeile 40

```
6. 6. 21.
```

INT

Die Integer-Funktion ermittelt den ganzzahligen Anteil des numerischen Ausdrucks.

Die Funktion bewirkt für ein Argument mit positivem Wert das Abschneiden (d.h. der gebrochene Anteil einer Zahl wird unterdrückt). So erzeugt die Funktion eine Zahl, die kleiner oder gleich dem Argument ist. Hat das Argument jedoch einen negativen Wert, so wird eine negative Zahl erzeugt, deren Betrag größer oder gleich dem des Argumentes ist.

Beispiel:

```
a) 10 A = INT (PI)
    20 B = INT (3.9)
    30 C = INT (-3.14)
    40 PRINT A;" ";B;" ";C
```

Ausgabe bei Zeile 40

```
3. 3. -4.
```

SGN

Die SGN-Funktion ermittelt das Vorzeichen des numerischen Ausdrucks.

SGN X = 1 wenn $X > 0$

SGN X = 0 wenn $X = 0$

SGN X = -1 wenn $X < 0$

Beispiel:

```
1) 10 FOR A = 15 TO -15 STEP -15
    20 B = SGN A
    30 PRINT A, B
    40 NEXT A
```

Ausgabe bei Zeile 30

15.	1.	ENTER
-----	----	-------

0.	0.	ENTER
----	----	-------

-15.	-1.
------	-----

SQR ($\sqrt{\quad}$)

Die SQR-Funktion berechnet die Quadratwurzel des numerischen Ausdrucks; negative Werte sind nicht zulässig.

Beispiel:

```
a) 10 A = SQR 2
    20 B = SQR A
    30 PRINT A, B
```

Ausgabe bei Zeile 30

1.414213562	1.189207115
-------------	-------------

PI (π)

Die Konstante PI ist als = 3.141592654 gespeichert.

RND

Die Funktion liefert die nächste Zufallszahl einer Pseudozufallszahlenfolge.

Syntax



Der Wert des numerischen Ausdrucks bestimmt das Intervall, aus dem die Zufallszahlen gezogen werden, die dann das Ergebnis bilden.

Das Intervall ist wie folgt festgelegt:

RND X

$X < 0$ – die gleiche Folge von Zufallszahlen wird bei jedem Programm-
lauf generiert

$0 \leq X < 1$ – $0 \leq \text{RND } X < 1$

$X \geq 1$ – $1 \leq \text{RND } X \leq X$ (wenn X Ganzzahl)

$1 \leq \text{RND } X \leq (\text{INT } X) + 1$ (wenn X Dezimalbruch)

Die Genauigkeit der Zufallszahl beträgt 10 Stellen.

Hinweis:

1. Die durch RND erhaltenen Zahlen sind nicht echt zufällig, da sie auf Grund eines festgelegten Rechengangs ermittelt werden. Sie erscheinen jedoch als zufällig und haben auch dieselben statistischen Eigenschaften wie echte Zufallszahlen. Nach dem Einschalten erzeugt der **PC-1251** immer dieselbe Folge von Zufallszahlen. Sollen grundsätzlich neue Folgen von Zufallszahlen erzeugt werden, muß die RANDOM-Anweisung zusätzlich zur RND-Funktion verwendet werden.

Beispiel:

```
a) 5 CLEAR
    10 DIM B(5)
    (15 RANDOM)
    20 FOR A = 0 TO 5
    30 F = RND 49
    40 IF (B(0)=F)OR(B(1)=F)OR(B(2)=F)OR(B(3)=F)OR
        (B(4)=F)OR(B(5)=F) GOTO 30
    50 B(A) = F
    60 NEXT A
    70 PRINT B(0); B(1); B(2); B(3); B(4); B(5)
    80 END
```

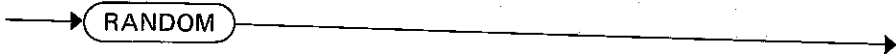
Ausgabe in Zeile 70

XX XX XX XX XX XX

RANDOM

Die Anweisung wählt einen neuen Startpunkt für den Zufallszahlengenerator.

Syntax



Die RANDOM-Anweisung muß vor dem ersten Aufruf der RND-Funktion i, Programm stehen. Sie bewirkt, daß bei jeder Programmausführung eine neue Zufallszahlenfolge initiiert wird.

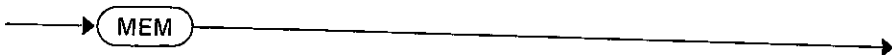
Beispiel:

Setzen Sie bei dem unter RND aufgeführten Beispiel auf Zeile 15 RANDOM.

MEM

Die MEM-Funktion gibt Auskunft über die Größe des freien Speicherbereichs (s. Speicherorganisation).

Syntax

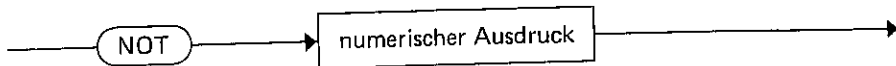


MEM gibt die Anzahl der unbelegten Bytes im Hauptspeicher an. Berücksichtigt werden dabei sowohl der Programm- als auch der Feldvariablenpeicher.

NOT-Funktion

Die NOT-Funktion negiert 16 duale Stellen eines numerischen Ausdrucks.

Syntax



Der Wert des numerischen Ausdrucks muß zwischen $-32768 = \&8000 = 2^{15}$ und $32767 = \&7FFF = 2^{15} - 1$ liegen.

Der numerische Ausdruck wird als 16-stellige Dualzahl, d.h. im Zweiersystem verschlüsselt. Die NOT-Funktion invertiert jede Ziffer der Dualzahlen und addiert EINS hinzu.

Negative Dualzahlen werden durch das "Zweierkomplement" dargestellt.

Das Zweierkomplement von $-X$ errechnet sich aus $(NOT X) + 1$.

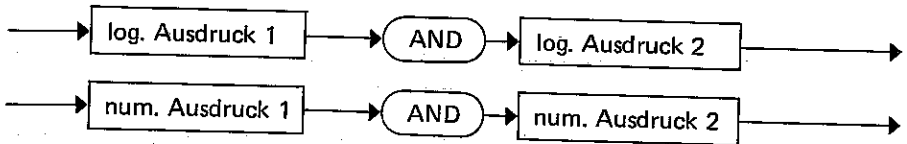
Beispiele:

X (dez.)	X (dual)	NOT X (dual)	NOT X (dez)
-15	1111111111110001	000000000001110	+14
-3	111111111111101	000000000000010	+2
-2	111111111111110	000000000000001	+1
-1	111111111111111	000000000000000	0
0	000000000000000	111111111111111	-1
+1	000000000000001	111111111111110	-2
+2	000000000000010	111111111111101	-3
+3	000000000000011	111111111111100	-4
+15	000000000001111	111111111110000	-16

AND-Operator

“UND”-Verknüpfung von zwei logischen Ausdrücken. Logisches “UND” der dualen Darstellung zweier numerischer Ausdrücke.

Syntax



Anwendung auf logische Ausdrücke:

Das Ergebnis ist wahr (1), wenn der Ausdruck 1 und der Ausdruck 2 wahr ist, sonst ist es falsch (0).

Beispiel:

```
10: INPUT "VARIABLE A= ?"  
   "A"  
20: IF (A>2) AND (A<8)  
   GOTO 50  
30: GOTO 10  
50: PRINT USING "##":A;  
   IST ZWISCHEN 2 UND 8
```

Die Sprunganweisung in Zeile 20 wird nur dann ausgeführt, wenn der logische Ausdruck $(A > 2)$ und der logische Ausdruck $(A < 8)$ wahr, also 1, ist.

Anwendung auf numerische Ausdrücke:

Die Werte der numerischen Ausdrücke müssen zwischen $-32768 = \&8000 = 2^{15}$ und $+32767 = \&7FFF = 2^{15} - 1$ liegen.

Das Ergebnis hat an jeder dualen Stelle eine 1, in der der Ausdruck 1 und der Ausdruck 2 eine 1 hat, sonst eine 0.

Beispiel:

X AND 1

Das Ergebnis ist 0, wenn X eine gerade Zahl ist
und 1, wenn X eine ungerade Zahl ist

34 AND 70 : Ergebnis ist 2

34 = 0100010

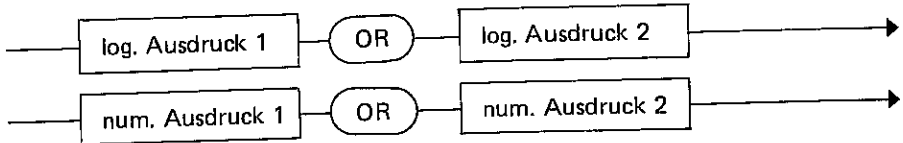
70 = 1000110

AND 0000010 = 2

OR-Operator

“ODER“-Verknüpfung von zwei logischen Ausdrücken. Logisches
“ODER“ der dualen Darstellung zweier numerischer Ausdrücke.

Syntax



Anwendung auf logische Ausdrücke:

Das Ergebnis ist wahr (1), wenn Ausdruck 1 oder Ausdruck 2 oder beide wahr sind, sonst ist es falsch (0):

Beispiel:

```
10: INPUT "VARIABLE A= ?"  
   " : A  
20: IF (A<=2) OR (A>=8)  
   GOTO 10  
50: PRINT USING "##"; A;  
   IST ZWISCHEN 2 UND 8
```

Die Sprunganweisung in Zeile 20 wird nur dann ausgeführt, wenn der logische Ausdruck ($A < 2$) oder der logische Ausdruck ($A > 8$) wahr, also 1, ist.

Anwendung auf numerische Ausdrücke:

Die Werte der Ausdrücke müssen zwischen $-32768 = \& 8000 = 2^{15}$ und $+32767 = \& 7FFF = 2^{15} - 1$ liegen.

Das Ergebnis hat in jeder dualen Stelle eine 1, in der im Ausdruck 1 oder im Ausdruck 2 oder in beiden eine 1 steht, sonst eine 0.

Beispiel:

17 OR 3 : Ergebnis ist 19

17 = 10001

3 = 00011

OR 10011 = 19

12. Programmieren in BASIC

Der PC-1251 verwendet die weit verbreitete Programmiersprache BASIC. Das Wort "BASIC" steht für "Beginner's All-Purpose Symbolic Instruction Code"; die Sprache wurde 1960 am Dartmouth College entwickelt und hat sich heute insbesondere für Mikrocomputer durchgesetzt.

BASIC ist im Gegensatz zu anderen Sprachen einfach und leicht verständlich; es gestattet den Dialogbetrieb mit dem Rechner und ist dabei so flexibel, daß ein bestehendes Programm ohne großen Aufwand geändert werden kann. Abgesehen von einigen Abweichungen von BASIC-Version zu BASIC-Version ist die Sprache maschinenunabhängig.

12.1 BASIC – Übersicht

Wie jede natürliche Sprache hat auch die (formale) Computersprache BASIC eine eigene Grammatik und einen zugehörigen Zeichensatz.

Letzterer setzt sich aus folgenden Buchstaben, Ziffern und Sonderzeichen zusammen:

A, ... , Z

0, 1, ... , 9

() . ; , : + - * / \$ & % # @ ! ? < > = ^

Zur Vermeidung von Verwechslungen unterscheidet man zwischen \emptyset (Null) und dem Buchstaben O.

Funktionstasten gibt es auf dem PC-1251 nicht; die Eingabe mathematischer Funktionen erfolgt über die Schreibmaschinentastatur. Es besteht allerdings die Möglichkeit, häufig benutzte Sprachelemente über die reservierbaren Tasten einfacher abzurufen (s. Reserve-Mode).

BASIC kennt folgende Sprachelemente:

- numerische Konstanten
- Textkonstanten
- numerische Variablen
- Textvariablen
- dimensionierte oder Feldvariablen
- BASIC-Schlüsselwörter für
 - Standardfunktionen
 - Zuweisungen
 - Eingaben
 - Ausgaben
 - Steuerungen
 - Kommandos
- Operationszeichen (+, -, *, /, <, >, =, <=, >=, <>)

Die Sprachelemente werden zu Programmsätzen zusammengefügt, die vom Rechner der angegebenen Reihenfolge nach abgearbeitet werden.

12.2 Programmerstellung

Die Programmerstellung gliedert sich im wesentlichen in drei Schritte.

1. Problemanalyse

Das gestellte Problem muß zunächst analysiert und dann so aufbereitet werden, daß sich die einzelnen Teilprogramme leicht und übersichtlich programmieren lassen.

.. Kodierung

Die in einzelne Schritte zerlegte Aufgabenstellung wird in eine Programmiersprache (hier BASIC) übertragen und in den Computer eingegeben.

3. Test

Dieser Schritt ist neben der Problemanalyse der schwierigste Teil. Nur sehr selten wird das kodierte Programm auf Anhieb fehlerfrei laufen. Eine Reihe von Fehlern kann dabei auftreten:

- Schreibfehler
- falsche Sprachelemente (Syntaxfehler)
- falscher logischer Aufbau
- falsche Programmstruktur (fehlerhafte Problemanalyse)



Der **PC-1251** bietet eine Reihe von Möglichkeiten, zumindest einem Teil der Fehler auf die Spur zu kommen (s. Fehlermeldung/Fehlersuche).

12.3 Programmaufbau

Bei der Übersetzung der Problemanalyse in BASIC muß eine Reihe von Regeln achtet werden.

Ein vollständiges BASIC-Programm besteht aus einer Folge von Anweisungen. Diese Anweisungen müssen in der Reihenfolge angeordnet sein, in der sie ausgeführt werden sollen; eine Ausnahme besteht, wenn durch eine Steueranweisung eine andere Reihenfolge programmiert wird.

Es gilt:

1. Die Programmierung erfolgt zeilenweise, wobei jede Programmzeile eine oder mehrere BASIC-Anweisungen enthalten kann. Innerhalb einer Zeile werden die Anweisungen durch einen Doppelpunkt getrennt (z.B. 10 INPUT Z : GOTO 200).
2. Die Anweisungen dürfen nicht länger als eine Zeile sein, d.h. sie können nicht in der nächsten Zeile fortgesetzt werden. Eine Zeile kann bis zu 79 Zeichen enthalten. 24 Zeichen sind jeweils auf der Anzeige sichtbar. Die restlichen eingegebenen Zeichen können mit Hilfe der Cursor-Tasten   sichtbar gemacht werden.
3. Jede Zeile muß mit einer positiven ganzen Zahl (der Zeilennummer) eingeleitet werden. Dieselbe Zeilennummer darf nur einmal im Programm verwendet werden.
Die Programmzeilen werden vom Rechner in aufsteigender Reihenfolge ausgeführt, wenn nicht durch Steueranweisungen eine andere Reihenfolge programmiert wurde.
Die Numerierung muß nicht lückenlos sein, es ist sogar zweckmäßig z.B. Zehnerschritte zu wählen, damit die Möglichkeit besteht, noch nachträglich Zeilen einzufügen.
Die Zeilennummern dürfen beim PC-1251 die Werte zwischen 1 und 999 annehmen.

12.4 Eingabe eines Programms

Die Eingabe eines Programms in den Rechner erfolgt im PRO-Modus.

Ein sich eventuell noch im Speicher befindendes Programm wird durch das Kommando NEW gelöscht. Der Programmspeicher ist dann leer, und die neuen Programmzeilen können eingetippt werden.



Beispiel:

- a) NEW
- ```
10 A = 15
20 B = 3
30 C = A + B
40 PRINT C
50 END
```

Jede Zeile muß mit **ENTER** abgeschlossen werden. Der Rechner fügt dann von sich aus einen Doppelpunkt zwischen Zeilennummer und der ersten Anweisung ein.

## 12.5 Überprüfen des Programms, Editieren und Auflisten

Ist das Programm eingegeben, kann man im PRO-Modus den Inhalt der einzelnen Zeilen noch einmal überprüfen.


Betätigt man die Tasten  und , erscheint die jeweils vor- bzw. nachstehende Programmzeile auf der Anzeige.

Durch das Kommando LIST kann man die Zeile mit der niedrigsten Zeilennummer in die Anzeige bringen, aber auch bestimmte genau spezifizierte Zeilen lassen sich durch LIST < Zeilennummer > aufrufen.

### Beispiel:





urch Eingabe von LIST 30 erscheint auf der Anzeige:

```
30: C = A+B
```

Die Zeile 40 kann jetzt durch Betätigen der -Taste abgerufen werden:

```
40: PRINT C
```

Will man die Eingabe verändern, korrigieren, erweitern oder löschen, geht das völlig problemlos. Editieren kann dabei nur im PRO-Modus erfolgen.

1. Man tippt eine neue Zeile mit der gleichen Zeilennummer ein. Die alte Zeile wird dadurch überschrieben.
2. Eine Zeile wird ersatzlos gelöscht, indem man < Zeilennummer >  eingibt.
3. Man holt sich wie oben beschrieben die zu ändernde Zeile auf die Anzeige. Mit den Tasten  und  schiebt man den Cursor auf die zu verändernde Eingabe.  
Der Doppelpunkt hinter der Zeilennummer verschwindet, und man kann die Zeile wunschgemäß verändern, so als hätte man sie gerade neu eingegeben.  
Die Änderung muß wieder mit  abgeschlossen werden.

### Beispiel:

In dem o.g. Programmbeispiel soll in Zeile 10 A = 17 gesetzt werden:

LIST 10

Die Zeile 10 erscheint auf der Anzeige.



Beginn der Änderung, der Doppelpunkt hinter der Zeilennummer verschwindet.



Cursor auf die 5.

7

Die 5 wird mit 7 überschrieben, der Doppelpunkt hinter der Zeilennummer erscheint wieder, die Zeile ist mit der Änderung gespeichert.

## 12.6 Programmausführung

Die Programmausführung kann nur im RUN-Modus stattfinden.

Der Programmstart kann durch verschiedene Kommandos erfolgen (s. RUN, GOTO, Definable Keys; Reserve-Modus).

Nach Eingabe des Startkommandos beginnt der Rechner mit der Bearbeitung des Programms. Während der Programmausführung ist auf der Anzeige das Wort "BUSY" sichtbar. Dies erlischt bei Programmende oder wenn eine Ausgabe auf dem Display erfolgt. Solange "BUSY" aufleuchtet, nimmt der Computer keine Eingaben an. Nach Programmende erscheint das Bereitschaftssymbol auf der Anzeige.

Will man die Programmausführung unterbrechen, betätigt man die Taste  . Auf der Anzeige erscheint die Meldung

BREAK IN <Zeilennummer>

z.B.

Die Ausführung kann durch das Kommando CONT fortgesetzt werden.



## 12.7 Fehlermeldung / Fehlersuche

Erkennt der **PC-1251** während der Programmausführung einen Fehler, bricht er die Bearbeitung ab, und auf der Anzeige erscheint eine Fehlermeldung der Form

ERROR < Fehlernummer > IN < Zeilennummer >

Die Liste der Fehlermeldungen finden Sie numerisch geordnet im Anhang. Gleichzeitig mit dem Fehlercode wird die Zeilennummer angegeben, in der sich der Fehler befindet.

Diese Zeile kann auch im RUN-Modus angezeigt werden: Zunächst löschen Sie die Fehlermeldung mit der Taste **CL**. Dann betätigen Sie die Taste **↑**. Die betreffende Zeile erscheint auf der Anzeige, und der Cursor blinkt auf dem fehlerhaften Sprachelement.

Will man die Zeile korrigieren, schaltet man in den PRO-Modus und verfährt wie weiter oben beschrieben.

### Beispiel:

```
1Ø A = 2
2Ø B = 1Ø
3Ø C = B/A
4Ø PRINT C,
5Ø END
```

Bei der Ausführung dieses Programms erscheint

E R R O R 1 I N 4 Ø

Die PRINT-Liste in Zeile 4Ø ist unvollständig.

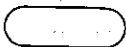
## 2.8 PC-1251 BASIC

Im folgenden Kapitel wird der BASIC-Sprachumfang des **PC-1251** ausführlich beschrieben. Dabei wird zum einen der genaue (mögliche) Aufbau der jeweiligen BASIC-Anweisung (Syntax) graphisch dargestellt und zum anderen die Wirkung der Anweisungen und Befehle erklärt.

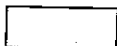
Folgt man beginnend von links nach rechts den Pfeilen der Graphen, erhält man die korrekte Syntaxvariante der einzelnen Anweisungen.



Die eingeschlossenen Zeichen sind Operatoren / Sonderzeichen bzw.



Schlüsselworte/Anweisungen. Sie müssen so in das Programm übernommen werden.



Der eingeschlossene Ausdruck beschreibt ein vom Benutzer zu definierendes Sprachelement und ist im Programm durch das Konkrete zu ersetzen.



In spitzen Klammern wird die Bedeutung des jeweiligen Sprachelements angegeben.

**ENTER** muß nach jeder Programmzeile eingegeben werden, obwohl es im Graphen fortgelassen wurde.

Die Syntax wird dann folgendermaßen dargestellt.

z.B:

GOTO

Schlüsselwort

2 \* (A + B)

numerischer Ausdruck  
< Zeilennummer >

oder

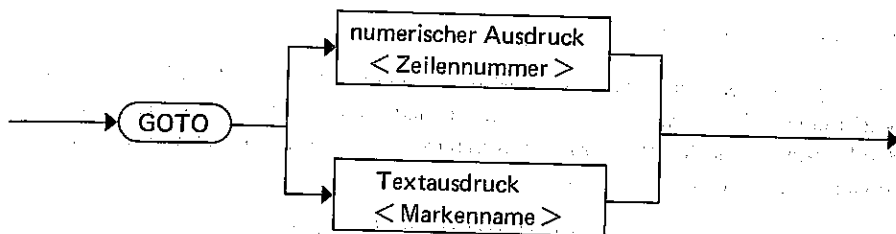
GOTO

Schlüsselwort

"SHARP"

Textausdruck  
< Markenname >

daraus folgt:



Die Richtung wird durch die Pfeile gekennzeichnet. Nach jeder Programmzeile muß **ENTER** eingegeben werden, obwohl dies in der Graphik fortgelassen wird.

Aus dem o. g. Beispiel kann man erkennen, daß das BASIC-Schlüsselwort GOTO sowohl in Verbindung mit einer numerischen Konstante, einer numerischen Variablen als auch einer Textkonstante und einer Textvariablen zulässig ist.

# NEW (PRO- und Reserve-Modus)

Das Kommando löscht den Haupt- und Reserve-Speicher.

## Syntax

NEW →

### A. PRO-Modus

Das gespeicherte Programm und alle Variablenwerte werden gelöscht bzw. auf Null gesetzt.

### B. Reserve-Modus

Alle Tastendefinitionen werden gelöscht.

## Hinweise:

1. Wird das Kommando im RUN-Modus eingegeben, erfolgt eine Fehlermeldung (ERROR 9).

# CLEAR

Das Kommando löscht alle Variablen und Felder aus dem Hauptspeicher und setzt die Standardvariablen auf  $\emptyset$  (Null).

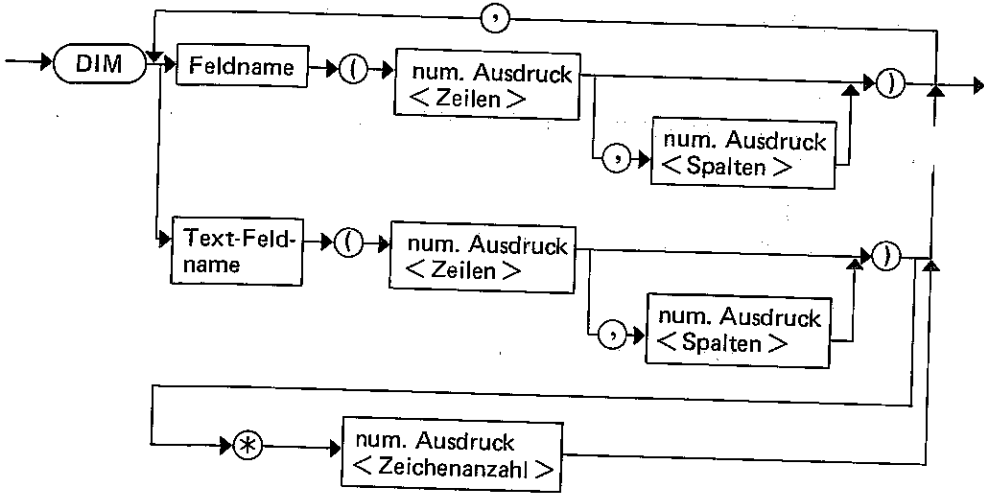
## Syntax

CLEAR →

# DIM

Mit dieser Anweisung werden Feldvariablen (Arrays) explizit vereinbart.

## Syntax



Die Feldvereinbarung legt für einen Vektor (eindimensional) oder eine Matrix (zweidimensional) die Gesamtlänge, die Dimension und die Indexbereiche fest. Das ist sinnvoll, da mit einfachen Variablen allein viele Algorithmen überhaupt nicht oder nur sehr umständlich programmiert werden können.

Deshalb vereinbart man mit den Feldvariablen auf einen Schlag mehrere systematisch geordnete sog. indizierte Variable.

Die DIM-Anweisung reserviert den für das Feld notwendigen Platz im Programm. speicherbereich.

Die Werte der numerischen Ausdrücke geben die Anzahl der Zeilen und Spalten des zweidimensionalen Feldes an; wird nur ein numerischer Ausdruck angegeben, erhält man einen eindimensionalen Vektor.

Textfelder können ebenfalls ein- oder zweidimensional vereinbart werden. Der letzte numerische Ausdruck gibt für diesen Fall die Länge der Zeichenketten an. Die Dimensionierung ist einerseits durch den Speicherplatz begrenzt, andererseits durch die Indizes, die aus dem Intervall  $0 \sim 255$  sein müssen.

Numerische und Textfelder dürfen den gleichen Namen haben; die Anweisung

DIM B(3, 3), B\$(2, 5)

ist zulässig.

Ein Textelement kann bis zu 80 Zeichen lang definiert werden; wird keine explizite Vereinbarung getroffen, beträgt die max. Länge 16 Zeichen.

Ein Feld darf im Programmablauf nur einmal vereinbart werden; andernfalls erfolgt eine Fehlermeldung (ERROR 3).

Während des Programmablaufs erfolgt der Zugriff auf ein Feldelement durch Angabe des Namens und der jeweiligen Indizes analog zur Form in der DIM-Vereinbarung. Eine LET-Anweisung sieht dann beispielsweise so aus:

LET B(1, 1) = 3.75

Für die Dimensionierung von zweidimensionalen Feldern stehen 50 Variablenamen zur Verfügung (B(X, X) – Z(X, X) und B\$(X, X) – Z\$(X, X)).

Für die Vereinbarung von eindimensionalen Feldern stehen 51 Variablenamen zur Verfügung (B(X) – Z(X) und B\$(X) – Z\$(X) und A(X) oder A\$(X)).

Die Variable A nimmt eine Sonderstellung ein. A(1) bis A(26) sind äquivalent zu den Standardvariablen A – Z; A(0) ist nicht definiert.

Die Variablen A(27) – A(255) brauchen nicht mit einer DIM-Anweisung vereinbart zu werden.

Die höchste aufgerufene Variable A(X) für  $X > 26$  vor einer ordentlichen Dimensionierungsanweisung (z.B. DIM B(5) etc.) vereinbart automatisch das Feld A auf A(X).

Z.B. sei die höchste aufgerufene A-Variable A(30).

Es folgt DIM B(5). Das Feld A ist damit dimensioniert auf A(30).

Der Aufruf einer Variablen A(X) für  $X > 30$  führt in diesem Fall zu einer Fehlermeldung (ERROR 3).

### Beispiel:

- a) 10 A(30) = 30  
20 DIM B(5)  
30 A(31) = 31

Die Anweisung in Zeile 30 führt zur Fehlermeldung ERROR 3.

```

b) 10 DIM B(5)
 20 A(30) = 30
 30 DIM C(10)
 40 FOR A(31) = 0 TO 10
 50 NEXT A(31)

```

Der Aufruf der Laufvariablen A(31) führt in diesem Fall zur Fehlermeldung ERROR 3 IN 40.

### Hinweise:

1. Wird ein Feld A (X) oder A\$(X) für  $X \leq 27$  mit der DIM-Anweisung vereinbart, wird **kein** Platz im Programmspeicherbereich reserviert (s. Standardvariable).
2. Es empfiehlt sich, der Übersicht halber alle Felder möglichst vor der ersten Anweisung im Programm explizit zu vereinbaren.
3. Weiterhin sollte man nach Möglichkeit alle Indexausdrücke als ganzzahlige Ausdrücke vorsehen.
4. Bei der Vereinbarung erhalten numerische Felder die Werte 0 und Textfelder das ASCII-Zeichen Null.
5. Da Null ein gültiger Wert für den Index ist, hat das Feld jeweils eine um 1 größere Anzahl von Zeilen bzw. Spalten als die Werte der Definition.

### Beispiele:

a) 10 DIM X(4)

Die Zeile 10 vereinbart ein numerisches eindimensionales Feld mit 5 Elementen, das folgendermaßen aussieht:

|      |      |      |      |      |
|------|------|------|------|------|
| X(0) | X(1) | X(2) | X(3) | X(4) |
|------|------|------|------|------|

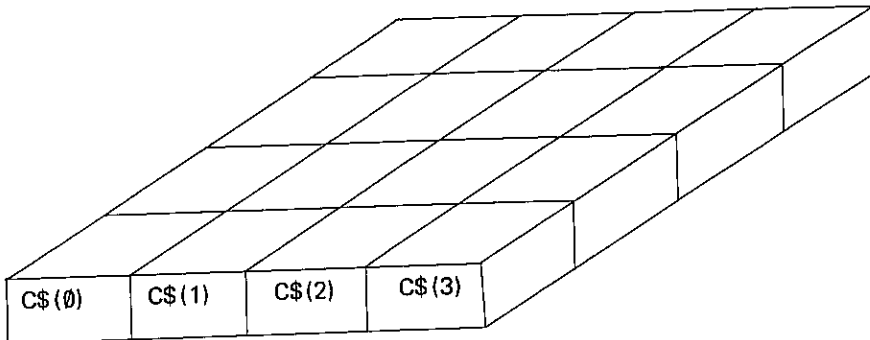
b) 20 DIM B(2, 3)

|         |         |         |         |
|---------|---------|---------|---------|
| B(0, 0) | B(0, 1) | B(0, 2) | B(0, 3) |
| B(1, 0) | B(1, 1) | B(1, 2) | B(1, 3) |
| B(2, 0) | B(2, 1) | B(2, 2) | B(2, 3) |

Die Zeile 20 vereinbart ein zweidimensionales Feld mit 12 Elementen.

c) 40 DIM C\$(3) \* 4

Die Zeile 40 vereinbart einen Textvektor mit 4 Elementen, von denen jedes eine Kapazität von 4 Zeichen hat.



d) 10 DIM V(3), P(2)

20 I=4 : J=5

30 DIM W\$(I \* J)

In Zeile 30 wird ein eindimensionaler Textvektor vereinbart, dessen Index sich aus den Variablen I und J errechnet (W\$(20)).

e) 10 DIM S(9, 1)

20 FOR I=0 TO 90 STEP 10

30 S(I/10, 1)=SIN I

40 S(I/10, 0)=I

50 NEXT I

60 FOR J=0 TO 9

70 PRINT S(J, 0), S(J, 1)

80 NEXT J

In Zeile 10 wird folgendes Feld S(9, 1) vereinbart:

| S(0, 0) | S(0, 1) |
|---------|---------|
|         |         |
|         |         |
|         |         |
|         |         |
|         |         |
|         |         |
|         |         |
|         |         |
|         |         |
|         |         |
| S(9, 0) | S(9, 1) |

In der Schleife von Zeile 20 bis 50 nimmt die Schleifenvariable I die Werte 0, 10, 20 . . . 90 an. In den Zeilen 30 und 40 werden die Indizes des Feldes S durch I/10 (also 1,2 etc.) bestimmt. Nach Durchlaufen der Schleife von Zeile

60 bis 80 enthält S in den Zeilen jeweils den Winkel I und den Sinus des Winkels SIN I, bzw. in der ersten Spalte alle Werte von I und in der zweiten Spalte alle zugehörigen Sinus-Werte.

Ausgabe bei Zeile 70

0. 0. ENTER

10. 1.73648 E-01 ENTER

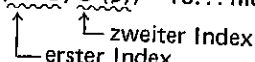
bis

80. 0.984807753 ENTER

90. 1.

### Hinweis:

Mit dem PC-1251 ist es möglich, einen Ausdruck als Index von zweidimensionalen Feldvariablen zu verwenden. Dabei jedoch die DIM-Anweisung nicht für den zweiten Index verwenden. Feldvariablen wie z.B. A(30) können jedoch dafür verwendet werden.

Beispiel 1:  $B(A * B, C(\emptyset)) = 10 \dots$  nicht verwendbar  


Beispiel 2:  $B(C(\emptyset), 5) = 10 \dots$  verwendbar

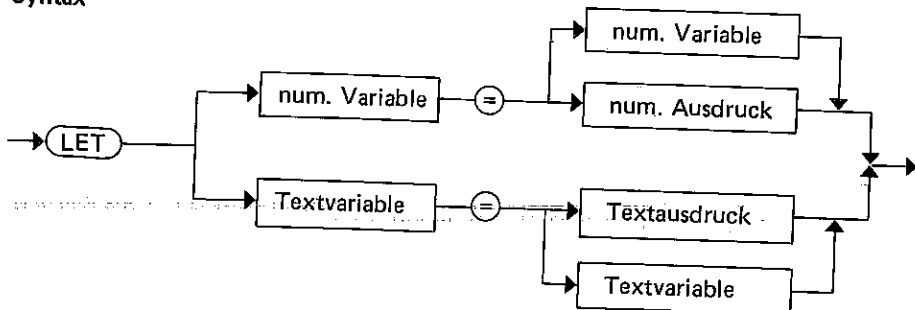
Beispiel 3:  $B(4, A(30)) = 10 \dots$  verwendbar

In Beispiel 1 kann daher C(∅) verwendet werden, wenn man es durch A(30) ersetzt oder, falls erforderlich, indem man  $A(30) = C(\emptyset)$  davor setzt.

## LET

Die Anweisung weist einer Variablen (einfach oder indiziert) einen Wert zu.

### Syntax





Ist die Variable indiziert, bewirkt die Wertzuweisung zunächst die Berechnung aller Indizes. Danach erfolgt die Auswertung des rechts stehenden Ausdrucks bzw. der Variablen; der entsprechende Ergebniswert wird dann anschließend der links stehenden Variablen zugewiesen.

### Hinweise:

1. Obwohl in einer Wertzuweisung ein Gleichheitszeichen auftritt, handelt es sich nicht um eine Gleichung im mathematischen Sinne.  
Die Programmzeile

10 LET A = A + 1

ist also durchaus sinnvoll; sie bewirkt, daß der Wert von A um 1 erhöht wird.

2. Der bisherige Wert der linksstehenden Variablen wird mit der Ausführung der Wertzuweisung überschrieben und geht damit verloren.
3. Das Schlüsselwort LET darf beim **PC-1251** weggelassen werden, sofern es nicht nach einer IF-Anweisung verwendet wird.
4. Einer numerischen Variablen kann nur ein numerischer Wert, einer Textvariablen nur ein Textausdruck zugewiesen werden.

### Beispiele:

a) 10 LET A=4.15

Die numerische Konstante 4.15 wird der Variablen A zugewiesen.

b) 10 LET Y=3 \* COS X

Der numerische Ausdruck  $3 * \cos X$  wird berechnet und der Variablen Y zugewiesen.

c) 10 LET B\$="PC-1251"

Die Textkonstante "PC-1251" wird der Textvariablen B\$ zugewiesen.

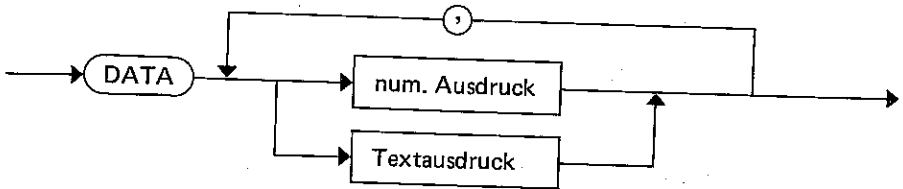
d) 10 B(I,J)=I^K

Die Indexwerte I/J der Variablen werden zunächst bestimmt, danach wird der numerische Ausdruck  $I^K$  berechnet und der Variablen B(I, J) zugewiesen. Die LET-Anweisung wurde hier ausgelassen. Beachten Sie, daß das Feld B(I, J) vorher dimensioniert werden muß (s. DIM).

# DATA

Die Vereinbarung schreibt Werte in eine Standarddatei ein, die jedem Programm zugeordnet ist.

## Syntax



Die DATA-Vereinbarung überträgt die nach DATA aufgelisteten Daten in der Reihenfolge von links nach rechts in die Komponenten der Standarddatei des Programms. Die Standarddatei ist dabei eine linear geordnete endliche Folge von Speichereinheiten, in die numerische oder Textkonstanten abgelegt werden können.

Treten in einem Programm mehrere DATA-Vereinbarungen auf, so werden vor Ausführung der ersten Programmanweisung alle zugehörigen Daten in der Reihenfolge ihres Auftretens in die Standarddatei geschrieben.

Auf diese Daten kann dann während der Programmausführung über die READ-Anweisung zugegriffen werden.

## Beispiel:

In einem Programm werden folgende Vereinbarungen vorgenommen:

```
100 DATA 4.15, 3.7E-1, -20
120 DATA A-B, "VON", 3E12
130 DATA "SHARP", "PC-1251", "ENDE"
```

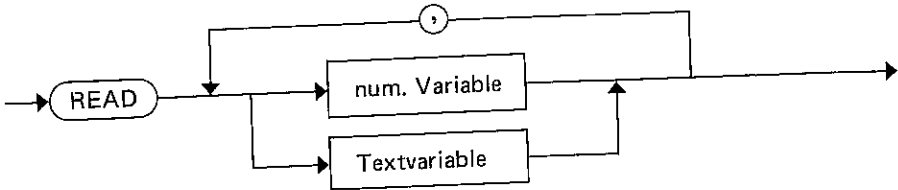
Die Übertragung in die Standarddatei sieht dann so aus:

|         |        |     |     |     |      |       |
|---------|--------|-----|-----|-----|------|-------|
| 4.15    | 3.7E-1 | -20 | A-B | VON | 3E12 | SHARP |
| PC-1251 | ENDE   |     |     |     |      |       |

# READ

Die Anweisung liest der Reihenfolge nach Werte aus der Standarddatei in die nach READ aufgeführten Variablen ein.

## Syntax



Mit der ersten READ-Anweisung wird der erste Wert der Standarddatei der ersten nach der READ-Anweisung aufgeführten Variablen zugewiesen.

Die zweite READ-Anweisung weist den zweiten Wert aus der Standarddatei der entsprechenden Variablen zu. Dabei können numerische Ausdrücke nur numerischen und Textausdrücke nur Textvariablen zugeordnet werden.

Folgt auf die READ-Anweisung mehr als eine Variable, so werden diesen der Reihenfolge nach die Elemente der Standarddatei zugeordnet.

Passen Variablentyp und Ausdruck nicht zusammen (z.B. READ A / DATA "SHARP"), wird automatisch der nächste passende Wert aus der Standarddatei eingelesen.

Ist in der DATA-Vereinbarung ein arithmetischer Ausdruck (z.B. DATA A+B), so wird zunächst dessen Wert berechnet, und dieser dann der entsprechenden Variablen zugewiesen.

### Hinweise:

1. Grundsätzlich müssen in der Standarddatei ausreichend Elemente gespeichert sein, damit allen auf READ folgenden Variablen ein Wert zugewiesen werden kann. Andernfalls erfolgt eine Fehlermeldung (ERROR 9).
2. Bei indizierten Variablen werden die Indizes zum Zeitpunkt der Ausführung der jeweiligen Wertzuordnung berechnet.
3. Bis zur Ausführung der nächsten READ- (oder RESTORE-) Anweisung bleibt das aktuelle Element der Standarddatei unverändert.
4. Nicht benutzte Daten aus der Standarddatei werden ignoriert.

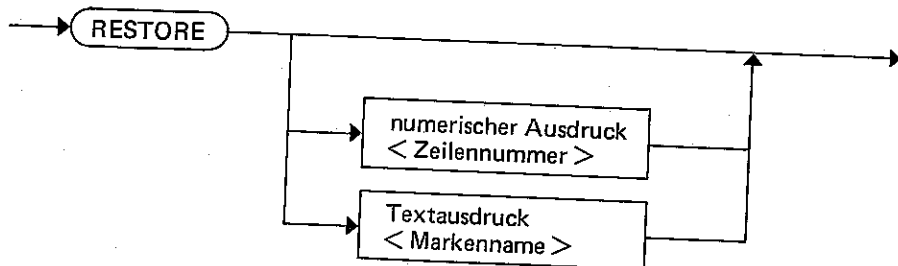
### Beispiele:

```
a) 10 READ A, B, C, D, E$, F, G$, H$, I$
 20 PAUSE H$: PAUSE E$: PAUSE G$: PAUSE I$
 30 DATA 4.15, 3.7E-1, -20
 40 DATA A-B, "VON", 3E12
 50 DATA "SHARP", "PC-1251", "ENDE"
```

## RESTORE

Die Anweisung ermöglicht das erneute Lesen der Standarddatei von Anfang an.

### Syntax



Die RESTORE-Anweisung bewirkt, daß ab der nächsten READ-Anweisung der Inhalt der Standarddatei von Anfang an gelesen wird.  
Der Wert des numerischen Ausdrucks gibt die Zeilennummer, der Textausdruck die Marke einer READ-Anweisung an.

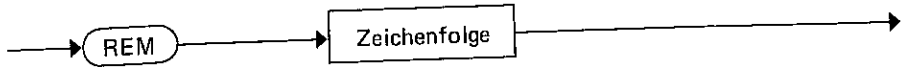
### Beispiele:

```
10 READ A, B, C
20 RESTORE
30 READ D, E, F, G, H, I
40 FOR Z = 1 TO 9
50 PAUSE A(Z)
60 NEXT Z
70 DATA 10, 20, 30, 40, 50, 60
```

# REM

Die Anweisung definiert eine Kommentarzeile im Programm.

## Syntax



Eine kommentierende Textzeile kann an jeder beliebigen Stelle eines Programms eingefügt werden. Die Zeile wird bei der Programmausführung ignoriert.

## Hinweise:

1. Die REM-Anweisung definiert die gesamte restliche Zeile als Kommentar.

10 REM INITIALISIERUNG: A=0

in diesem Fall wird auch die Wertzuweisung A=0 als Kommentar aufgefaßt und nie ausgeführt.

2. Um Programmlisten übersichtlicher und leichter verständlich zu machen, ist es empfehlenswert, vor größeren zusammenhängenden Programmabschnitten in Kommentarzeilen die Bedeutung des jeweiligen Programmteils zu erläutern.

## Beispiele:

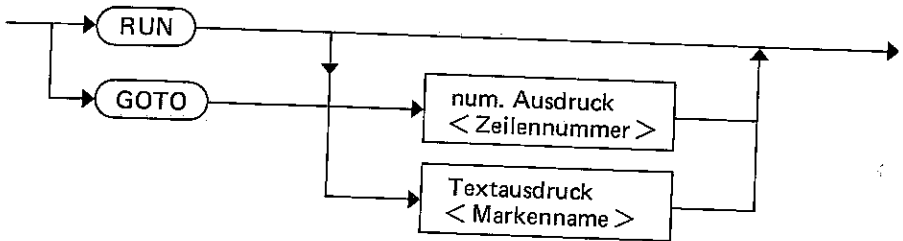
```
a) 10 REM *****
 20 REM *
 30 REM * TEXT *
 40 REM *
 50 REM *****
```

```
b) 10 READ P: REM PREIS PER STÜCK
 20 READ N: REM ANZAHL
 30 C=P*N
```

# RUN GOTO Definable Keys

Mit diesen Kommandos wird die Programmausführung gestartet.

## Syntax



1. Bei Eingabe des Kommandos RUN wird das Programm mit der niedrigsten Zeilennummer gestartet. Ist eine Zeilennummer oder ein Markenname explizit angegeben, wird die Ausführung mit der so spezifizierten Zeile begonnen.
2. Das Kommando GOTO ist nur in Verbindung mit einem numerischen (Zeilennummer) oder Textausdruck (Markenname) zulässig.
3. **Definable Keys**

Eine weitere Möglichkeit, ein Programm zu starten, sind die Definable Keys. Dazu muß die Taste **DEF** und eine der folgenden Tasten gedrückt werden:

**A S D F G H J K L = Z X C V B N**  
**M SPC**

Die Programmausführung beginnt dann mit der Zeile, die das entsprechende Zeichen als Marke enthält. Wird eine im Programm nicht enthaltene Marke angegeben, führt dies zu einer Fehlermeldung (ERROR 9).

## Hinweise:

1. Eine weitere Möglichkeit, die Programmausführung zu starten, besteht bei entsprechender Programmierung der reservierbaren Tasten (s. Reserve).
2. Wartet der Rechner nach einer INPUT-Anweisung auf eine Eingabe, kann man durch Betätigen von DEF und einer der Definable Keys zu einer anderen Programmzeile springen. Die INPUT-Anweisung wird dann nicht ausgeführt.

3. Bei Programmstart über RUN werden alle außer den Standardvariablen gelöscht.
4. Unterschiede zwischen RUN, GOTO und den Definable Keys  
 Beim Start des Programms werden bestimmte Grundzustände des Rechners gesetzt. Die drei Startkommandos unterscheiden sich hierbei wie aus der folgenden Tabelle ersichtlich:

| Zustandsänderung erfolgt bei                           | RUN | GOTO | Def. Keys |
|--------------------------------------------------------|-----|------|-----------|
| Die Anzeige wird gelöscht (s. AREAD)                   | ja  | ja   | ja        |
| Feldvariable werden gelöscht                           | ja  | nein | nein      |
| Die RESTORE-Anweisung wird ausgeführt (s. DATA / READ) | ja  | nein | nein      |
| Das USING-Format wird gelöscht                         | ja  | nein | nein      |
| Das WAIT-Intervall bleibt erhalten                     | ja  | nein | nein      |

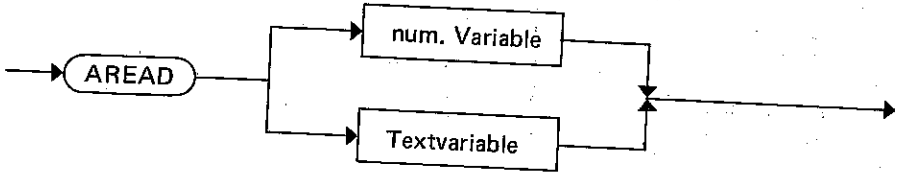
Bei allen drei Startbefehlen wird:

- die FOR-NEXT Information gelöscht
- die GOSUB-Rücksprungadresse gelöscht
- der Standardvariablenspeicher **nicht** gelöscht
- der TRACE-Zustand nicht geändert

# AREAD

Die Anweisung liest den vor Programmstart auf der Anzeige stehenden Wert in eine Variable ein.

## Syntax



AREAD muß die **erste** Anweisung im Programm sein. Der auf der Anzeige stehende numerische oder Textausdruck wird der auf AREAD folgenden Variablen zugewiesen. Das Programm muß über die Definable Keys gestartet werden.

## Beispiel:

- a) 10 "S" : AREAD X  
20 PRINT "SINH ("; X;") IST "; (EXP (X) - EXP (-X))/2

Geben Sie ein:

1.5

DEF S

Ausgabe bei Zeile 20

SINH (1.5) IST 2.1292794

Das Programm wird gestartet, und gleichzeitig wird der Wert 1.5 der Variablen X zugewiesen.



### Hinweise:

1. Wenn das Bereitschaftszeichen (" $>$ ") am Anfang der Programmausführung angezeigt wird, so ist die festgelegte Variable gelöscht.
2. Wenn der Inhalt zu Beginn der Programmausführung durch PRINT zur Anzeige gebracht wird, so wird folgendes gespeichert:

Beispiel: bei Ausführung des folgenden Programms:

```
10 "A" : PRINT "ABC", "DEFG"
```

```
20 "S" : AREAD A$: PRINT A$
```

RUN-Modus

```
DEF A → ABC DEFG
```

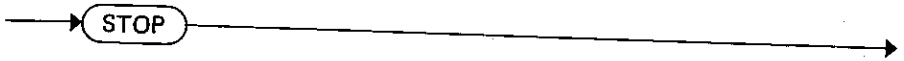
```
DEF S → DEFG
```

- Wenn auf der Anzeige PRINT numerischer Ausdruck, ein numerischer Ausdruck oder PRINT "String" erscheint, so wird der rechts befindliche Anzeigehalt gespeichert.
- Wenn auf der Anzeige PRINT numerischer Ausdruck, numerischer Ausdruck, numerischer Ausdruck... erscheint, so wird der erste Teil des Anzeigehalts (der ganz links befindliche) gespeichert.
- Wenn auf der Anzeige PRINT "String", "String", "String"... erscheint, so wird möglicherweise ein bedeutungsloser Inhalt gespeichert.

# STOP

Die Anweisung unterbricht die Programmausführung.

## Syntax



Durch die STOP-Anweisung wird die Ausführung eines Programms beendet. Auf der Anzeige erscheint

BREAK IN < Zeilennummer >

Nach Betätigen der **ENTER** -Taste erscheint das Bereitschaftssymbol auf der Anzeige. Es lassen sich jetzt z.B. der Inhalt der Variablen oder der gerade durchlaufene Programmabschnitt überprüfen.

Ein Programm kann mehrere STOP-Anweisungen enthalten. Die Programmausführung kann durch Eingabe der CONT-Anweisung fortgesetzt werden.

## Hinweise:

1. Bei längeren Programmen ist es sinnvoll, die STOP-Anweisung an das logische Ende eines Programmteiles zu setzen. Das erleichtert die Suche nach eventuellen Fehlern.
2. Drückt man die **↑** -Taste, erscheint die zuletzt ausgeführte Programmzeile in der Anzeige. Durch Betätigen der **↓** -Taste kann die Programmausführung zeilenweise fortgesetzt werden, wobei zunächst die Zeile abgearbeitet und dann die jeweilige Zeilennummer angezeigt wird. Betätigt man erneut die **↑** -Taste, erscheint wieder die zuletzt ausgeführte Programmzeile.

**Beispiel:**

- a) 10 PRINT 10
- 20 STOP
- 30 PRINT 30
- 40 PRINT 40
- 50 PRINT 50
- 60 PRINT 60
- 70 PRINT 70

START

ANZEIGE

RUN

10.

BREAK IN 20

30.

30:

30: PRINT 30

bis

70.

70: PRINT 70

# CONT

Das CONT-Kommando setzt die Programmausführung nach einer Unterbrechung durch STOP bzw. BREAK fort.

## Syntax

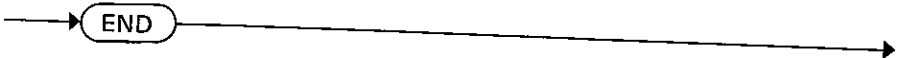


Wird durch CONT die Programmausführung nach einer Unterbrechung fortgesetzt, bleiben alle Zustände des Computers (FOR-NEXT, GOSUB, DIM etc.) erhalten. Auch kann man nach einer Programmunterbrechung die aktuellen Variablenwerte überprüfen und verändern.

# END

Die Anweisung beendet die Programmausführung.

## Syntax



Die Anweisung steht normalerweise am logischen Ende eines Programms und beendet die Ausführung.

### Hinweis:

Die END-Anweisung kann unter Umständen weggelassen werden, die Programmausführung wird automatisch nach Abarbeiten der letzten Programmzeile beendet. Hat man mehr als ein Programm im Speicher, wird bei fehlender END-Anweisung automatisch das nächste Programm gestartet.

**Beispiel:**

```
a) 10 PRINT "START PRO 1"
20 FOR I = 1 TO 20
30 PAUSE I
40 NEXT I
100 PRINT "START PRO 2"
110 PRINT "<END> IN PRO 1 VERGESSEN"
120 END
```

Ausgabe bei Zeile 10

START PRO 1

ENTER

Ausgabe bei Zeile 30

1.

bis

20.

Ausgabe bei Zeile 100

START PRO 2

ENTER

Ausgabe bei Zeile 120

<END> IN PRO 1 VERGESSEN

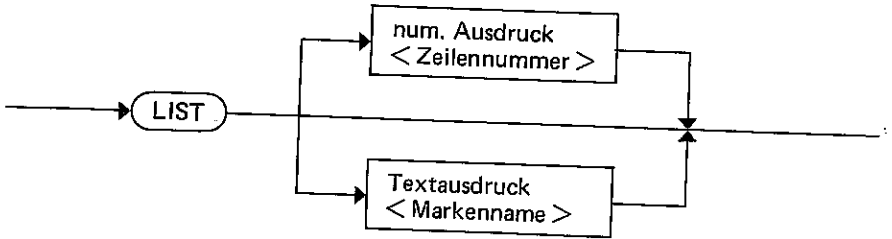
ENTER

Programmende

# LIST

Durch das Kommando LIST kann jeweils eine Programmzeile auf die Anzeige geholt werden.

## Syntax



Das Kommando kann nur im PRO-Modus ausgeführt werden. Im RUN- oder Reserve-Modus erscheint eine Fehlermeldung (ERROR 9).

LIST ohne zusätzlichen Ausdruck holt die Programmzeile mit der niedrigsten Zeilennummer auf die Anzeige.

Durch Angabe einer bestimmten Zeilennummer oder eines Markennamens lassen sich auch genau bestimmte Programmzeilen auf der Anzeige ausgeben.

Ist die angegebene Zeile nicht vorhanden, wird die nächsthöhere gelistet.

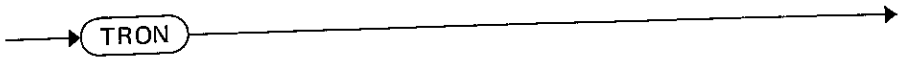
Durch Betätigen der Tasten  und  wird die folgende bzw. die vorherige Programmzeile angezeigt.

# TRACE

## A. TRON

Der Computer wird zum Suchen von Programmfehlern in den TRACE-Zustand geschaltet.

### Syntax



Nach Eingabe von TRON wird das Programm wie üblich gestartet. Die Programmausführung stoppt aber automatisch nach jeder Zeile, gleichzeitig wird die entsprechende Zeilennummer angezeigt.

Die Taste  läßt den Zeileninhalt auf der Anzeige erscheinen; die Taste  führt die Programmausführung fort.

## B. TROFF

Der TRACE-Zustand wird aufgehoben.

### Syntax



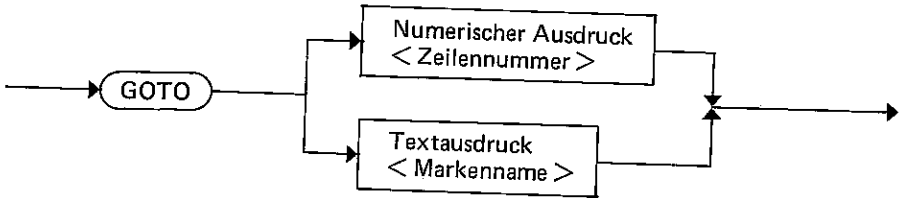
### Hinweise:

1. Die Kommandos TRON und TROFF können auch im Programm stehen. Dadurch ist es möglich, begrenzte Programmteile zu überprüfen.
2. Hält man die  Taste fest, erfolgt die zeilenweise Programmausführung schnell hintereinander.
3. Der TRACE-Zustand wird aufgehoben, wenn der Computer abgeschaltet wird, oder wenn man die Tasten   betätigt.

# GOTO

Die Anweisung lenkt die Programmausführung zu der im Sprungziel angegebenen Programmzeile.

## Syntax



Die Sprunganweisung GOTO lenkt den Programmlauf ohne jede Einschränkung zu einer bestimmten Programmzeile. Der numerische Ausdruck bezeichnet eine Zeilennummer, wobei jeweils nur der ganzzahlige Anteil berücksichtigt wird. Der Textausdruck gibt einen Markennamen als Sprungziel an; dieser kann aus max. 7 Zeichen bestehen.

## Hinweise:

1. Nicht jede Zeile ist als Sprungziel erreichbar. Ein Sprungziel innerhalb einer Schleife von außerhalb anzuspringen, ist nicht zulässig.
2. Ist das Sprungziel im Programm nicht vorhanden, erscheint eine Fehlermeldung (ERROR 4).

## Beispiele:

a) 10 LET A = 0  
20 INPUT X  
30 LET A = A + X  
:  
50 GOTO 20

Die Anweisungsfolge bewirkt die Summation der nacheinander für X eingegebenen Werte auf der Variablen A.



```
b) 10 A = 1
 20 "LABEL-1": LPRINT USING"#####";A;A^2;A^3
 30 A=A+1
 40 IF A > 10 THEN GOTO 60
 50 GOTO "LABEL-1"
 60 END
```

Das Programm berechnet die Quadrat- und Kubikzahlen zwischen 1 und 10.

Es liefert folgendes Ergebnis:

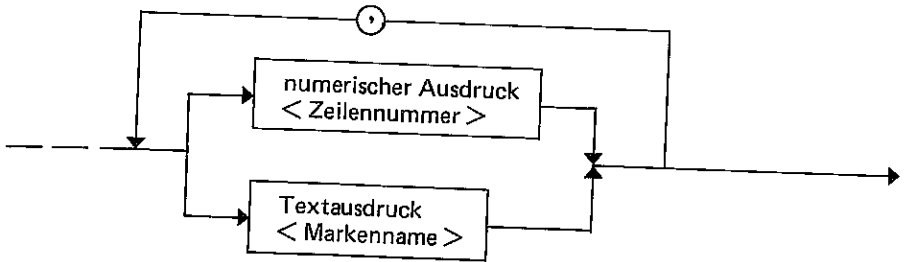
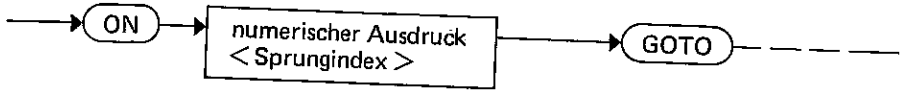
|    |     |      |
|----|-----|------|
| 1  | 1   | 1    |
| 2  | 4   | 8    |
| 3  | 9   | 27   |
| 4  | 16  | 64   |
| 5  | 25  | 125  |
| 6  | 36  | 216  |
| 7  | 49  | 343  |
| 8  | 64  | 512  |
| 9  | 81  | 729  |
| 10 | 100 | 1000 |

Die Ausgabe erfolgte über den Drucker CE-125 (Option). Deshalb wurde im Programm anstelle von PRINT die Ausgabeanweisung LPRINT verwendet.

# ON GOTO

Die Anweisung führt die Programmausführung zu einem bestimmten Sprungziel in Abhängigkeit vom Wert eines numerischen Ausdrucks.

## Syntax



Durch den ganzzahligen positiven Wert des Sprungindex wird festgelegt, welches Element aus der Sprungzielliste auszuwählen ist.

Die Liste kann aus numerischen (Zeilennummern) und/oder Textausdrücken (Markennamen) bestehen.

Hat der Index den Wert 1, verzweigt das Programm zum ersten Element der Sprungzielliste, beim Wert 2 erfolgt der Sprung zum zweiten Element usw., beim Wert  $i$  zum  $i$ -ten Element.

Hat der Sprungindex einen Wert  $\leq 0$ , erfolgt kein Sprung; die Programmausführung wird mit der auf die ON – GOTO folgenden Anweisung fortgesetzt.

Die Sprungzielliste nach GOTO besteht in diesem Beispiel aus 4 Elementen, den Zeilennummern 50, 30, 60 und 70. Der Sprungindex  $I$  kann die Werte  $-2, -1, 0, 1, 2, 3, 4, 5$  annehmen.

Für die Werte  $< 0$  ( $-2, -1, 0$ ) erfolgt kein Sprung; es wird die Programmzeile 30 ausgeführt.

Nimmt  $I$  den Wert 1, 2, 3 oder 4 an, so wird je nach dem aktuellen Wert von  $I$  zum ersten, zweiten, dritten oder vierten Element der Sprungzielliste verzweigt.

Z.B. ist  $I=3$ , so wird das dritte Element der Sprungzielliste – in diesem Fall 60 – ausgewählt. Für  $I=5$  ist kein Sprungziel angegeben; die Programmausführung geht sofort auf Zeile 30.

### Beispiele:

```
a) 40 FOR I=0 TO 3
 50 ON I GOTO 100, 200, 300
 60 PRINT "KEINE VERZWEIGUNG"
 :
400 NEXT I
```

I kann die Werte 0, 1, 2, 3 annehmen; hat der Index I den Wert 1, verzweigt der Programmablauf zur Zeile 100, bei I=2 zur Zeile 200 usw. Ist I=0, wird sofort Zeile 60 ausgeführt.

```
5:USING "##":B$=" "
10:FOR I=-2 TO 5
20:ON I GOTO 50,30,60,70
30:LPRINT I;B$;"KEIN SPRUNG"
40:GOTO 200
50:LPRINT I;B$;"SPRUNG NACH 50"
55:GOTO 200
60:LPRINT I;B$;"SPRUNG NACH 60"
65:GOTO 200
70:LPRINT I;B$;"SPRUNG NACH 70"
200:NEXT I
```

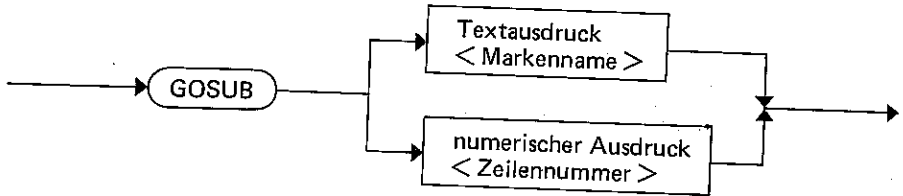
```
-2 KEIN SPRUNG
-1 KEIN SPRUNG
0 KEIN SPRUNG
1 SPRUNG NACH 50
2 KEIN SPRUNG
3 SPRUNG NACH 60
4 SPRUNG NACH 70
5 KEIN SPRUNG
```

Die Ausgabe erfolgte über den Drucker CE-125 (Option).

# GOSUB

Die Anweisung verzweigt den Programmablauf zu einem Unterprogramm.

## Syntax



Werden in einem Programm an verschiedenen Stellen die gleichen Befehlsfolgen benötigt, so schreibt man diese in der Regel aus Rationalisierungsgründen als Unterprogramm.

Die GOSUB-Anweisung lenkt die Programmausführung unbedingt zu der Programmzeile, in der das betreffende Unterprogramm beginnt. Analog zur GOTO-Anweisung wird das Sprungziel entweder als Zeilennummer oder als Markenname angegeben.

Ein Unterprogramm wird durch eine RETURN-Anweisung abgeschlossen; der Programmablauf wird dann mit der auf GOSUB folgenden Anweisung weitergeführt.

Die GOSUB-Anweisung kann an beliebiger Stelle im Programm stehen, auch können bis zu 10 Unterprogramme ineinandergeschachtelt werden.

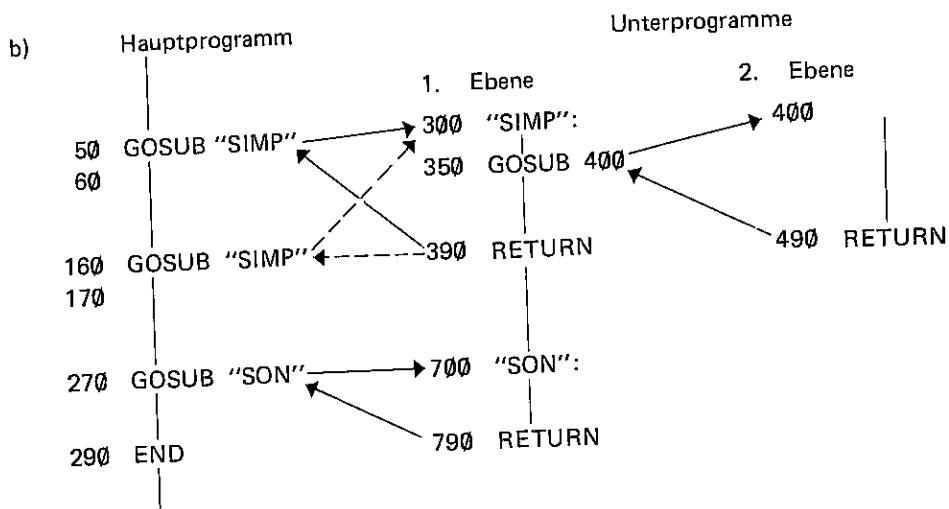
## Hinweise:

1. Geschachtelte Unterprogramme müssen streng hierarchisch geordnet sein; falls also ein Unterprogramm A ein Unterprogramm B aufruft, darf B nicht auch A aufrufen. Andererseits kann B sowohl vom Hauptteil des Programms als auch von A aufgerufen werden.
2. Ein Programm darf mehr als einen Aufruf desselben Unterprogramms enthalten.

### Beispiele:

a) 120 GOSUB 300  
130 PRINT "A="; A  
⋮  
180 GOSUB 300  
190 IF A < 10 THEN 250  
⋮  
300 LET X = C + B  
⋮  
380 RETURN

} Unterprogramm

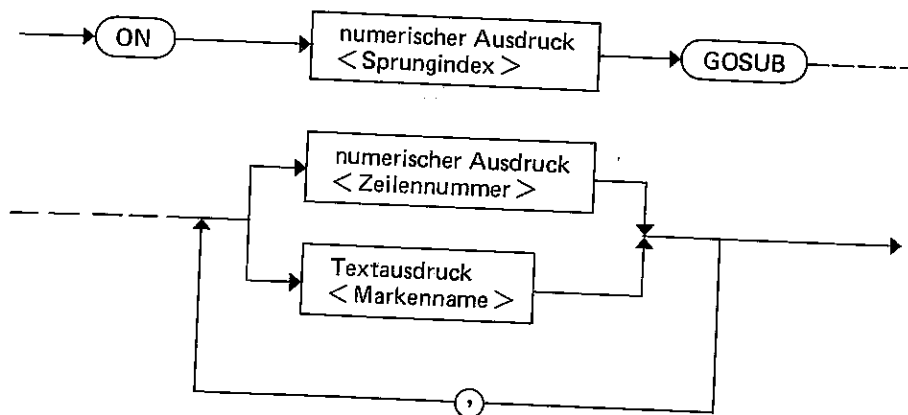


Hier werden mehrere Unterprogramme ineinander geschachtelt. In Zeile 50 und 160 wird das Unterprogramm "SIMP" aufgerufen. Dieses wiederum ruft in der nächsten Ebene das Unterprogramm mit der Zeilennummer 400 auf. Der Rücksprung erfolgt jeweils zu der Anweisung, die auf den betreffenden Unterprogrammaufruf folgt.

# ON ... GOSUB

Die Anweisung ruft ein Unterprogramm auf in Abhängigkeit vom Wert eines numerischen Ausdrucks.

## Syntax



Durch den ganzzahligen Wert des Sprungindex wird analog zur GOTO-Anweisung festgelegt, welches Element aus der Sprungzielliste auszuwählen ist. Die Liste kann dabei aus numerischen (Zeilennummern) oder Textausdrücken (Markennamen) bestehen.

Ist der Index negativ, 0 oder größer als die Anzahl der Elemente in der Zielliste, wird automatisch die nächste auf GOSUB folgende Anweisung ausgeführt (s. auch ON ... GOTO).

## Beispiel:

```
10 ON (SGN (D) + 2) GOSUB "MINUS", "NULL", "PLUS"
```

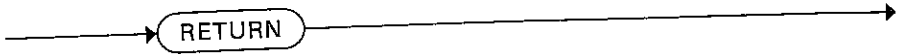
SGN (D) kann die Werte  $-1$ ,  $0$ ,  $1$  annehmen, jeweils für  $D < 0$ ,  $D = 0$ ,  $D > 0$ .

Damit erhält  $\text{SGN}(D) + 2$  die Werte 1, 2, 3 und entsprechend wird zum Unterprogramm verzweigt.

# RETURN

RETURN beendet ein Unterprogramm. Die Programmausführung springt zu der auf den entsprechenden Unterprogrammaufruf bzw. ON... GOSUB folgenden Anweisung zurück.

## Syntax



Die Wirkung der RETURN-Anweisung ist nur dann definiert, wenn mindestens eine GOSUB-Anweisung geöffnet ist.

Die RETURN-Anweisung bewirkt die Rückkehr zu der Anweisung, die auf die zuletzt geöffnete GOSUB-Anweisung als nächste folgt.

## Hinweise:

1. Ein Unterprogramm muß unbedingt mit RETURN abgeschlossen werden. GOTO ist nicht zulässig.
2. Eine RETURN-Anweisung im Hauptprogramm führt zu einer Fehlermeldung (ERROR 5).

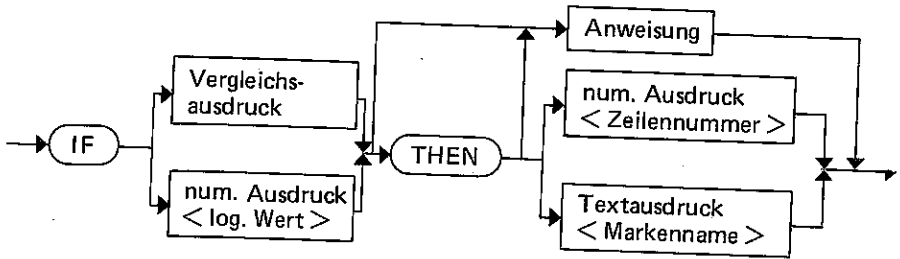
## Beispiele:

S. unter GOSUB

# IF ... THEN

Die Anweisung ermöglicht die Ausführung einer anderen Anweisung in Abhängigkeit vom logischen Wert einer Bedingung.

## Syntax



Der Vergleichs- bzw. der numerische Ausdruck stellt die Bedingung dar, die auf "wahr" oder "falsch" geprüft wird.

Ein Vergleich ist eine logische Aussage. Er wird als "wahr" angesehen, wenn der < logische Wert > größer ( $>$ )  $\emptyset$  und als "falsch" angesehen, wenn der < logische Wert > kleiner gleich ( $\leq$ )  $\emptyset$  ist.

Es gelten die aus der Mathematik bekannten Regeln für die Operatoren  $>$ ,  $<$ ,  $=$ ,  $\geq$ ,  $\leq$ ,  $\langle \rangle$ .

Die logischen Operatoren können auch auf Zeichenketten bzw. Textvariablen angewandt werden. Dabei wird die lexikographische lineare Ordnung des maschinenabhängig zugrundeliegenden Zeichensatzes geprüft.

Ergibt die geprüfte Bedingung den Wert "falsch", so ist die Ausführung der bedingten Anweisung beendet; ist die Bedingung "wahr", werden die auf THEN' (optional) folgenden Anweisungen ausgeführt. Die Angabe des numerischen bzw. Textausdrucks stellen dabei eine verkürzte Form der GOTO-Anweisung dar; der Anweisungsteil THEN muß in diesem Fall unbedingt gesetzt werden. THEN kann ausgelassen werden, wenn auf die Bedingung eine andere Anweisung folgt.

## Hinweis:

1. Anweisungen, die in der gleichen Programmzeile auf IF ... THEN folgen, werden nur dann ausgeführt, wenn die Bedingung nach IF "wahr" ist.
2. Beispiele für Vergleichsausdrücke bzw. < logische Werte > (s. Kapitel "Vergleichsausdrücke").



## Beispiel:

```
10:A=RND 100
20:INPUT "ZUFALLSZAHL I
 ST ? ";B
30:IF A<B GOTO 110
40:IF A>B GOTO 100
50:IF A=B PRINT "ZAHL G
 EFUNDEN <ENTER>":
 GOTO 200
100:PAUSE "GES. ZAHL IST
 GROESSER": GOTO 20
110:PAUSE "GES. ZAHL IST
 KLEINER": GOTO 20
200:INPUT "NOCHMAL ? JA/
 NEIN ";C$
210:IF C$="JA" THEN 10
220:IF C$="NEIN" THEN
 GOTO 230
230:END
```

### Zur Beachtung

1. Mit diesem Gerät können Rechnungen mit max. 12-stelliger Mantisse durchgeführt werden. Dabei wird jedoch die Mantisse zwecks höherer Genauigkeit bis zur 12. Stelle berechnet, für die Anzeige wird es jedoch zur 10. Stelle gerundet. Zum Beispiel werden  $5/9$  und  $5/9 * 9$  wie folgt berechnet:

$$5/9 \longrightarrow 5.55555555555E - 01$$

↑ Dieser Teil wird zur 10. Stelle gerundet.

$$\text{Anzeige} \longrightarrow 5.555555556E - 01$$

$$5/9 * 9 \longrightarrow 4.99999999999E 00$$

↑ Dieser Teil wird zur 10. Stelle gerundet.

$$\text{Anzeige} \longrightarrow 5$$

In dieser Weise werden Rechnungen für Mantissen bis zu 12 Stellen ausgeführt. Daher kann es zu Differenzen im Ergebnis von Rechnungen kommen, wenn man sie hintereinander oder getrennt ausführt.

(Beispiel 1)  $3^2 - 9 =$

Rechenfolge: 3   2  9  → - 9E - 11

Unabhängige Rechnung: 3   2  → 9

9  → 0

In der IF-Anweisung kann die Differenz sogar dazu führen, daß das Programm nicht läuft, wie für die jeweilige Reihenfolge geplant.

```
(Beispiel 2) 10 INPUT A
 20 IF A^2 >= 9 THEN 50
```

⋮

Bei  $A = 3$  führt  $3^2$  zum Ergebnis  $8.9999999999991E 00$ , so daß die IF-Anweisung nicht formuliert wird.

In diesem Falle programmiert man die Rechnung nochmals unter Verwendung von Variablen, so daß sie unabhängig wird, und zwar wie folgt:

```
10 INPUT A
15 B = A^2
20 IF B >= 9 THEN 50
```

Das Ergebnis von  $A^2$  wird anstelle einer Variablen eingesetzt, die zum Formulieren eines bedingten Ausdrucks verwendet wird.

⋮

(Potenzrechnungen basieren auf  $\log x$  und  $10^x$ , es besteht daher die Tendenz, daß Differenzen gegenüber den im Computer berechneten Ergebnissen verursacht werden.)

$$A^B \rightarrow 10^{B \log A}$$

2. Beim Potenzieren  $A^B$  wird im Falle eines negativen A-Wertes das Vorzeichen nicht korrekt angezeigt. Daher wird beim Potenzieren ( $A^B$ ) unter Verwendung der im Programm enthaltenen Variablen in der Zeile unmittelbar vor der Potenzrechnung eine B-Anweisung eingefügt, um zu verhindern, daß A negativ wird.

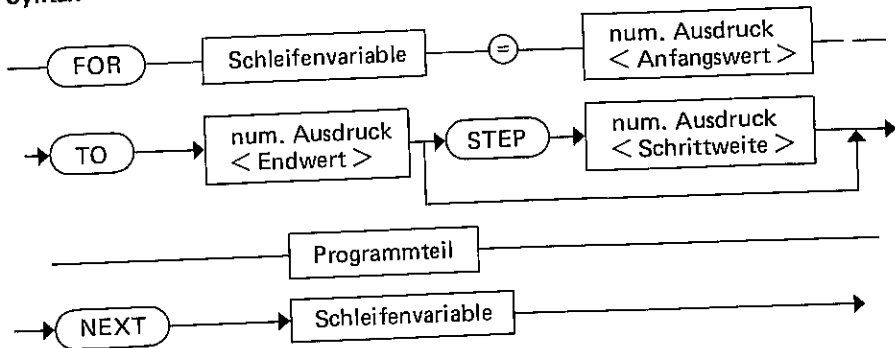
```
IF A < 0 PRINT "FEHLER" : END
```

↑ Variable mit dem Wert A in der Rechnung  $A^B$ .

# FOR ... TO ... STEP/NEXT

Die Anweisung dient zur wiederholten Ausführung des zwischen FOR ... TO und NEXT eingeschlossenen Programtteils.

## Syntax



Die Laufanweisung besteht aus der FOR-Anweisung und dem nachfolgenden Schleifenbereich. Dieser enthält die Anweisungen, die auf die FOR-Anweisung folgen. Dazu gehört auch die NEXT-Anweisung, die dieselbe Schleifenvariable wie die FOR-Anweisung enthalten muß.

Die Laufanweisung bewirkt, daß der zugehörige Schleifenbereich entsprechend dem Anfangs- und Endwert und der Schrittweite wiederholt durchlaufen wird.

Zunächst werden Anfangs- und Endwert und die Schrittweite berechnet. Fehlt letztere, so wird sie automatisch auf 1 gesetzt.

Der Schleifenvariablen wird daraufhin der Wert des Anfangsparameters zugewiesen.

Die Anweisungen des Schleifenbereichs werden dann bis zur zugehörigen NEXT-Anweisung ausgeführt. Durch die NEXT-Anweisung wird der Wert der Schleifenvariablen um die vereinbarte Schrittweite verändert, und die Schleife wird erneut durchlaufen.

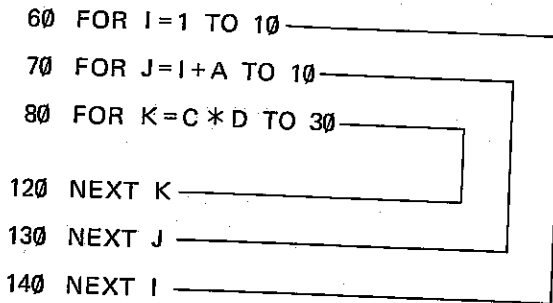
Die Ausführung der Laufanweisung ist beendet, wenn der Endwert der Schleifenvariablen größer gleich (bei positiver Schrittweite) bzw. kleiner gleich (bei negativer Schrittweite) dem Endparameter ist.

Die Programmausführung wird nach Beendigung des Schleifendurchlaufs mit der auf NEXT folgenden Anweisung weitergeführt.

**Hinweise:**

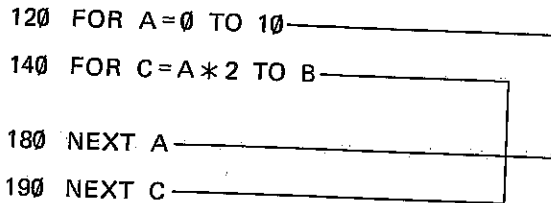
1. FOR . . . TO/NEXT Schleifen können in bis zu 5 Ebenen geschachtelt werden. Jede Schleife muß dabei ihre eigene Schleifenvariable mit eigenem Namen haben; die Schlei­fengrenzen dürfen sich nicht überlappen.

```
60 FOR I=1 TO 10
70 FOR J=I+A TO 10
80 FOR K=C*D TO 30
120 NEXT K
130 NEXT J
140 NEXT I
```



RICHTIGE  
SCHACHTELUNG

```
120 FOR A=0 TO 10
140 FOR C=A*2 TO B
180 NEXT A
190 NEXT C
```



FALSCH  
SCHACHTELUNG

2. Grundsätzlich kann eine Sprunganweisung im Schleifenbereich während der Ausführung der Laufanweisung in einen Programmteil außerhalb der Schleife führen.

Es darf allerdings nicht von außerhalb in einen Schleifenprogrammteil hineingesprungen werden (z.B. durch GOTO/GOSUB). Eine Ausnahme ist der Rücksprung in eine Schleife aus einem Unterprogramm. Wurde ein solches während eines Schleifendurchlaufs durch eine GOSUB-Anweisung aufgerufen, lenkt die Programmausführung nach Abarbeiten des Unterprogramms in die Schleife zurück.

Ein unzulässiger Sprung in eine Schleife führt bei der folgenden NEXT-Anweisung zu einer Fehlermeldung (ERROR 9).

```
1 GOTO 9
5 FOR I = 1 TO 10
9 PRINT "FALSCH"
10 NEXT I
```

falsche  
Verzweigung

```
100 FOR I = 1 TO 100
110 READ A$
120 IF A$ = "ENDE" GOTO 190
180 NEXT I
190 END
```

richtige  
Verzweigung

```
10 FOR I = 1 TO 100 STEP 5
20 READ A$
30 GOSUB "KASSETTE"
40
100 NEXT I
500 "KASSETTE": PRINT # "DATEN"; A$
600 RETURN
```

richtige Verzweigung in ein Unterprogramm

3. Der Wert der Schleifenvariablen darf ganzzahlige Werte zwischen  $-32768$  und  $32767$  annehmen; für die Schrittweite gilt das Gleiche mit Ausnahme des Wertes  $0$  (Endlosschleife). Letzteres führt zu einer Fehlermeldung (ERROR 3).
4. Nach Beendigung der Laufanweisung bleibt die Schleifenvariable definiert und behält den ihr zuletzt zugewiesenen Wert.
5. Die Werte der zugewiesenen Anfangs-, End- und Schrittweitenparameter bleiben fest, solange die Schleife durchlaufen wird. Das gilt auch dann, wenn der Wert einer zu ihrer Berechnung verwendeten Variablen innerhalb oder außerhalb des Schleifenbereichs neu bestimmt wird.

### Beispiele:

a)

```

100 FOR I=1 TO N
110 FOR J=1 TO N
120 READ B(I, J)
130 NEXT J
140 NEXT I

```

Einlesen der Zeilen einer Matrix  $B = B(I, J)$  über eine geschachtelte Schleifenanweisung.

```

b) 10 PAUSE "LISTE DER QUADRATZAHLEN"
20 PAUSE " VON 10 BIS 20"
30 FOR Z=10 TO 20
40 Q=Z * Z
50 PAUSE Q
60 NEXT Z
70 END

```

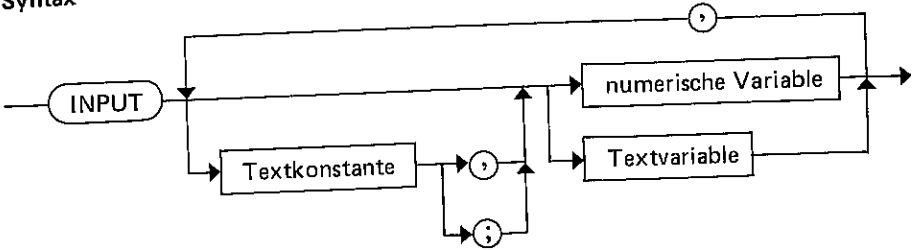
Beim ersten Durchlaufen der Schleife hat die Schleifenvariable  $Z$  den Wert  $10$ . In Zeile  $40$  wird daraus die Quadratzahl errechnet und der Variablen  $Q$  zugewiesen. In Zeile  $50$  wird der Wert von  $Q$  angezeigt; Zeile  $60$  führt die Programmausführung zurück zu Zeile  $30$  und die Schleifenvariable wird um  $1$  hochgesetzt. Die Schleife wird dann mit dem neuen Wert von  $Z$  erneut durchlaufen. Das letzte  $Q$ , das errechnet wird, ist  $Q=20 * 20$ .

Nach Ausführung der Zeile  $50$  wird automatisch die Schleife abgeschlossen und als nächste Anweisung die in Zeile  $70$  ausgeführt. Das Programm ist damit beendet.

# INPUT

Diese Anweisung weist Variablen Werte zu, die über die Tastatur eingegeben werden.

## Syntax



Die Ausführung des Programms wird unterbrochen. Es erscheint auf der Anzeige die Textkonstante oder ein "?", wenn kein Text spezifiziert ist als Hinweis darauf, daß die Werte für die Variablen über die Tastatur eingegeben werden können. Jede einzelne Eingabe wird mit **ENTER** abgeschlossen. Dies wird wiederholt, bis für alle Variablen die Werte eingegeben sind. Im Anschluß daran wird die Programmausführung fortgesetzt.

## Beispiele:

- a) 10 INPUT A  
20 IF A=1234 THEN 100  
30 GOTO 10  
100 PRINT A

Ausgabe bei Zeile 10

?

Eingabe: 1 2 3 4

1234

**ENTER**

Ausgabe bei Zeile 100

1234

Ist die eingegebene Zahlenfolge nicht 1234, so kehrt die Programmausführung zur Zeile 10 zurück.

b) 10 INPUT A\$  
20 IF A\$="START" THEN 100  
30 GOTO 10  
100 PRINT "PROG. ENDE"

Ausgabe bei Zeile 10

?

Eingabe: S T A R T

START

ENTER

Ausgabe bei Zeile 100

PROG. ENDE

c) 10 INPUT A, B  
20 IF A > B THEN 100  
30 GOTO 10  
100 PRINT "PROG. ENDE"

Ausgabe bei Zeile 10

?

Eingabe: 1 2 (Wertzuweisung für Variable A)

12

ENTER

?

Eingabe: 9 (Wertzuweisung für Variable B)

9

ENTER

Ausgabe bei Zeile 100

PROG. ENDE



- d) 10 INPUT "DATA-1", A  
20 IF A > 12 THEN 100  
30 GOTO 10  
100 PRINT "PROG. ENDE"

Ausgabe bei Zeile 10

DATA-1

Eingabe: 1 2 3 (Wertzuweisung Variable A)

123

ENTER

Ausgabe bei Zeile 100

PROG. ENDE

- e) 10 INPUT "DATA-1=" ; A , "DATA-2=" ; B  
20 IF A > B THEN 100  
30 GOTO 10  
100 PRINT "PROG. ENDE"

Ausgabe bei Zeile 10

DATA-1= \_

Eingabe: 7 5

DATA-1= 75

ENTER

DATA-2= \_

Eingabe: 7 1

DATA-2= 71

ENTER

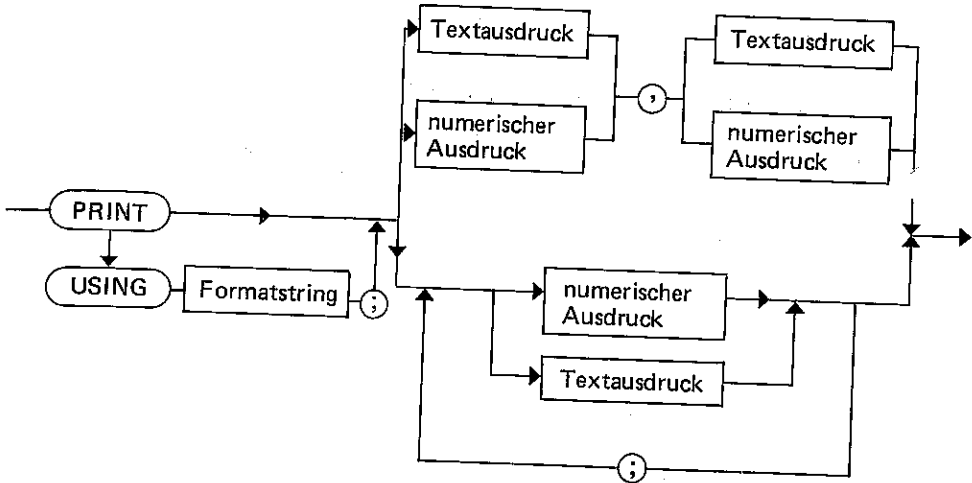
Ausgabe bei Zeile 100

PROG. ENDE

# PRINT

Mit dieser Anweisung werden numerische Werte und Zeichenfolgen auf der Anzeige ausgegeben.

## Syntax



Der USING-Teil der Anweisung spezifiziert das Ausgabeformat wie unter USING beschrieben. Der Formatstring steht für die dort erwähnte Textkonstante bzw. Textvariable.

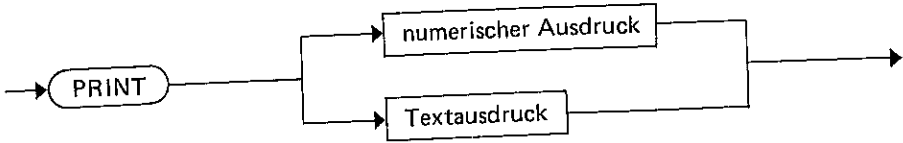
Über das Komma wird die Aufteilung der 24-spaltigen Anzeige in zwei gleich große Felder gesteuert.

Das Semikolon trennt die einzelnen Elemente einer Liste von Ausdrücken und steuert die Position des Cursors.

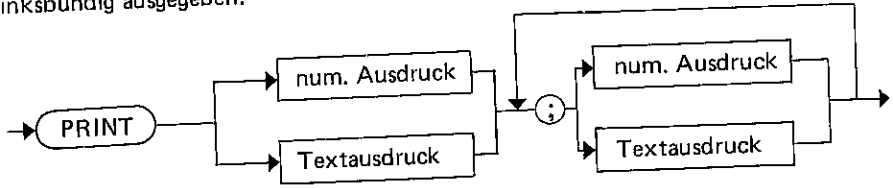
Wird die Cursor-Positionierung nicht explizit durch die Verwendung eines Semikolons gesteuert, werden die Werte einzelner numerischer Ausdrücke rechtsbündig und Textausdrücke linksbündig in das Feld geschrieben.

Nach dem Ausdrucken der PRINT-Liste (<Ausdrücke>) auf der Anzeige wird der Programmablauf angehalten, bis man ihn durch Drücken von **ENTER** wieder startet. Es sei denn, über die WAIT-Anweisung ist ein Zeitintervall definiert worden, nach dessen Ablauf die Ausführung automatisch wieder aufgenommen wird (s. WAIT).

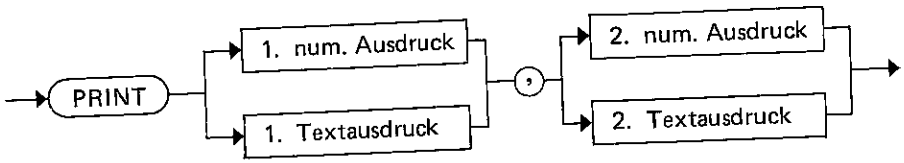
Die einzelnen Formate der PRINT-Anweisung haben folgende Wirkung:



Der in der PRINT-Liste spezifizierte Inhalt wird auf dem Display rechts- bzw. linksbündig ausgegeben.



Der Inhalt der PRINT-Liste wird angezeigt beginnend mit der momentanen Cursor-Position.



Die Anzeige wird in zwei Felder eingeteilt. Der erste Ausdruck erscheint in der linken, der zweite in der rechten Hälfte der Anzeige jeweils rechts- bzw. linksbündig.

**Hinweise:**

1. Die Anzeigen werden mit der nächsten Ausgabeanweisung gelöscht.
2. Mit der WAIT-Anweisung kann ein Zeitintervall definiert werden, nach dessen Ablauf die Programmausführung automatisch wieder gestartet wird.
3. Werden mehr Zeichen ausgegeben, als die Anzeige Schreibpositionen hat, so werden nur die ersten 24 Zeichen sichtbar.
4. Wird die PRINT-USING Anweisung benutzt, so ist das spezifizierte Format für alle PRINT-Anweisungen bis zur nächsten USING-Anweisung gültig.
5. Das USING-Format wird durch das Kommando RUN oder **[SHIFT] [CL]** gelöscht.

Beispiele:

- a) 10 A=3.1415  
20 B\$="SHARP"  
50 PRINT A  
60 PRINT B\$

Ausgabe bei Zeile 50

3.1415

ENTER

Ausgabe bei Zeile 60

SHARP

- b) 10 A=3.1415  
20 B\$="SHARP"  
30 C=7.89  
40 D\$="PC-1251"  
50 PRINT A;B\$;D\$;C  
60 PRINT B\$;" ";D\$;" ";A;" ";C

Ausgabe bei Zeile 50

3.1415SHARPPC-12517.89

ENTER

Ausgabe bei Zeile 60

SHARP PC-1251 3.1415 7.8

- c) 10 A=3.1415  
 20 B\$="SHARP"  
 30 C=7.89  
 40 D\$="PC-1251"  
 50 PRINT A, B\$  
 60 PRINT B\$, A  
 70 PRINT B\$, D\$  
 80 PRINT A, C

Ausgabe bei Zeile 50

**3.1415SHARP**

ENTER

Ausgabe bei Zeile 60

**SHARP 3.1415**

ENTER

Ausgabe bei Zeile 70

**SHARP PC-1251**

ENTER

Ausgabe bei Zeile 80

**3.1415 7.89**

- d) (5 WAIT 50)  
 10 A=3.1415  
 20 B\$="SHARP"  
 30 C=7.89  
 40 D\$="PC-1251"  
 50 E\$="&&###.#"  
 60 F\$="###.#&&"  
 70 PRINT USING E\$; B\$; A  
 80 PRINT USING F\$; B\$; A  
 90 PRINT USING; B\$; A  
 100 PRINT USING E\$; A, C  
 110 PRINT USING F\$; B\$, D\$  
 120 PRINT USING "###.####"; A, C  
 130 PRINT USING; A, C

Ausgabe bei Zeile 70

SH 3.1

ENTER

USING E\$ = &&###.# formatiert die Ausgabe mit zwei Textzeichen und zwei Vorkommastellen + eine Nachkommastelle für den numerischen Ausdruck.

Ausgabe bei Zeile 80

SH 3.1

ENTER

Die Ausgabe ändert sich nicht. Es ist also gleichgültig, ob zuerst der Textausdruck oder der numerische Ausdruck formatiert wird.

Ausgabe bei Zeile 90

SHARP3.1415

ENTER

USING ist nicht spezifiziert worden, daher wird die Ausgabe im Standardformat ausgeführt.

Ausgabe bei Zeile 100

3.1 7.8

ENTER

USING E\$ = &&###.#; die Textformatierung wird ignoriert.

Ausgabe bei Zeile 110

SH PC

ENTER

USING F\$ = ###.##&&; die Formatierung für den numerischen Ausdruck wird ignoriert.

Ausgabe bei Zeile 120

3.1415 7.8900

ENTER

USING ##.####; es werden 4 Nachkommastellen ausgegeben; fehlende Stellen werden mit Null aufgefüllt.

Ausgabe bei Zeile 130

3.1415

7.89

USING ohne Formatspezifizierung, die Anzeige erfolgt im Standardformat.

**Hinweis:**

Wird in Zeile 5 die WAIT-Anweisung programmiert, so startet die Programmausführung nach jeder Ausgabeanweisung automatisch nach Ablauf des Zeitintervalls ( $50 \times 1/64 = 0.78 \text{ sec.}$ ) (s. WAIT).

Wenn die Option CE-125 angeschlossen ist, kann die PRINT -Anweisung auch für die Druckwerk-Ausgabe verwendet werden. Die Ausgabe hat das gleiche Format wie es die Display-Ausgabe haben würde.

Dazu wird die Anweisung

PRINT=LPRINT eingegeben.

Diese Anweisung kann sowohl direkt als auch als Bestandteil des Programms eingegeben werden, z.B.

10: PRINT=LPRINT

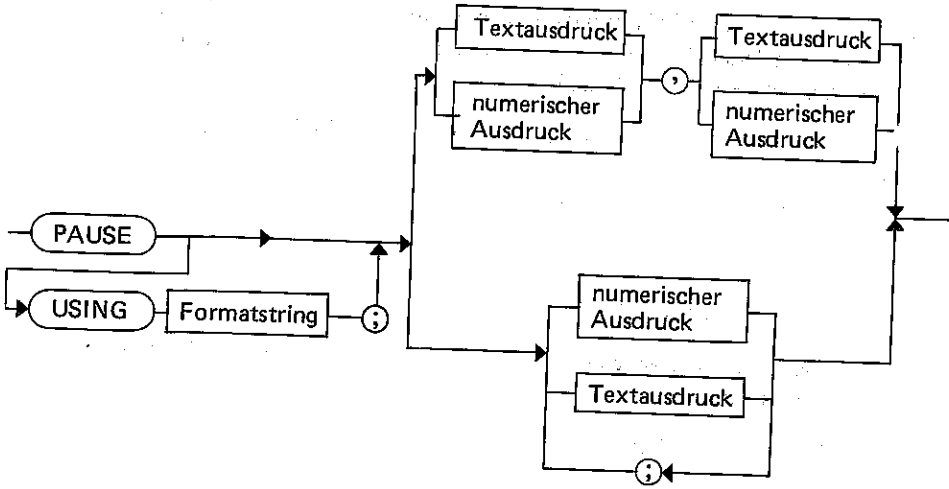
Die Anweisung wird gelöscht durch z.B.

100: PRINT=PRINT oder durch **SHIFT** **CL**

# PAUSE

Mit dieser Anweisung werden numerische Werte und/oder Zeichenfolgen auf der Anzeige ausgegeben und der Programmablauf automatisch fortgesetzt.

## Syntax



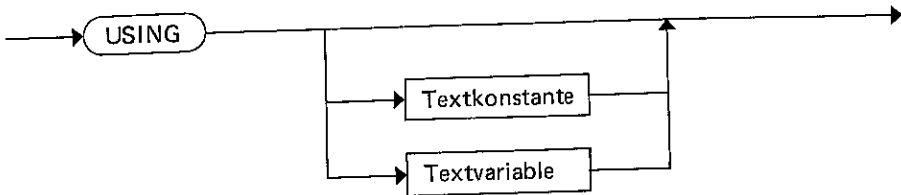
Die Syntax der PAUSE-Anweisung ist so aufgebaut wie bei der PRINT-Anweisung. Die Bedeutung der einzelnen Sprachelemente ist die gleiche. Im Unterschied zur PRINT-Anweisung wird der Programmablauf nach ca. 0,85 sec. wieder gestartet.



# USING

Diese Anweisung spezifiziert das Ausgabeformat von Texten und Werten numerischer Ausdrücke im Anschluß an PRINT- oder PAUSE-Anweisungen.

## Syntax



Wie bereits erwähnt, wählt der PC-1251 entsprechend dem auszugebenden Wert automatisch ein Ausgabeformat.

Mit der USING-Anweisung können Sie selbst bestimmen, in welchem Format die Ausgabe erfolgen soll.

Die Anweisung wirkt auf die Ausgabeanweisungen PRINT, PAUSE, LPRINT.

Die Formatspezifikation kann als Textkonstante oder als Wert einer Textvariablen eingegeben werden. Sie bleibt für alle nachfolgenden Ausgabeanweisungen bis zu ihrer Änderung gültig.

Wird die USING-Anweisung allein ohne Formatspezifikation gegeben, so wird damit die letzte Formatspezifikation unwirksam und auf die automatische Formatkonvertierung zurückgeschaltet. Das Gleiche bewirkt die Ausführung des RUN-Kommandos oder die **SHIFT** **CL** -Taste.

\*it der Textkonstanten bzw. Textvariablen wird das Format spezifiziert. Zur Festlegung des Formates dienen die Sonderzeichen

# . ^ &

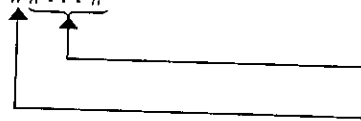
# bestimmt die Anzahl der Stellen im numerischen Ausgabefeld. Gibt man eine Folge des Zeichens # ein, so wird lediglich der ganzzahlige Anteil einer Zahl angezeigt, einschließlich Vorzeichen bei negativen Zahlen. Daher muß das Feld aus den ## . . . ## eine Stelle mehr besitzen als die größte auszugebende Zahl an Ziffern benötigt. Es können maximal 11 Stellen für Ganzzahlen bestimmt werden.

legt die Stelle des Dezimalpunktes fest. Die Folge der #-Zeichen nach dem Dezimalpunkt bestimmt die Anzahl der Nachkommastellen. In den Nachkommastellen werden fehlende Stellen immer mit Nullen aufgefüllt, überzählige Stellen werden ignoriert.

^ bestimmt die Anzeige im wissenschaftlichen Format.


& bestimmt die Anzahl der Zeichenstellen von Texten und Textausdrücken.

Die unterschiedlichen Formate lauten damit allgemein:

(1) "##...#"
 

Anzahl der Ganzzahlstellen

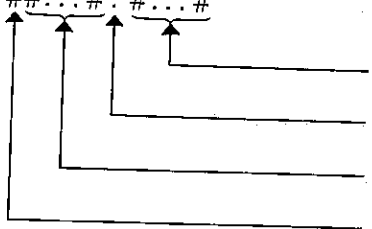
Vorzeichenstelle, das Vorzeichen wird nur bei negativen Zahlen ausgegeben

(2) "##.#..."
 

Dezimalpunkt wird ausgegeben

Anzahl der Vorkommastellen

Vorzeichenstelle

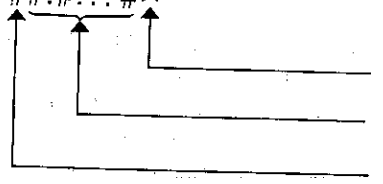
(3) "##.#...#.#..."
 

Anzahl der Nachkommastellen

Dezimalpunkt

Anzahl der Vorkommastellen

Vorzeichenstelle

(4) "##.#...#^"
 

wissenschaftliches Format

Mantisse

Vorzeichenstelle

Normierte Exponentialdarstellung zur Basis 10.

Das Format ist normiert auf eine Vorkommastelle, die Nachkommastellen und den Exponenten. Gibt man beim wissenschaftlichen Format mehr als eine Vorkommastelle oder mehrere  $\wedge$  ein, so hat das keinen Einfluß auf die Anzeige.

(5) `"&&...&&"`

↑ Anzahl der Zeichen eines Textfeldes

Die Zeichen eines Textausdrucks werden in das Feld geschrieben. Ist das Feld kleiner spezifiziert, werden nur die ersten Zeichen ausgegeben, ist es größer, wird der Rest mit Zwischenräumen aufgefüllt.

(6) Kombinierte Formatierung

a. `"##.#^&&&"`

↑ 3 Stellen für Text  
 ↑ wissenschaftliche Formatierung  
 ↑ Vorzeichenstelle

b. `"&&&&##.##"`

↑ eine Vorkommastelle, zwei Nachkommastellen, Vorzeichenstelle  
 ↑ vier Textstellen

FORMATSPEZIFIKATION

|   |                                                                                                                                                                                                       |
|---|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| # | Repräsentiert die Stellen des ganzzahligen und gebrochenen Anteils eines numerischen Feldes. Führende Nullen werden in Leerzeichen gewandelt, fehlende Nachkommastellen werden mit Nullen aufgefüllt. |
| . | Repräsentiert den Dezimalpunkt                                                                                                                                                                        |
| ^ | Repräsentiert die wissenschaftliche Formatausgabe.                                                                                                                                                    |
| & | Repräsentiert die Stellen zur Ausgabe von Zeichenfolgen.                                                                                                                                              |

### Hinweise:

1. Reicht das durch die USING-Anweisung festgelegte Ausgabeformat für die Darstellung einer Ganzzahl nicht aus, so wird ein Fehler 7 gemeldet.
2. Fehlerhafte Formate werden meistens erst bei der Ausgabeanweisung PRINT, PAUSE, LPRINT erkannt und angezeigt.

### Beispiele:

```
a) 10 A = 123.456789
 20 USING "####.#"
 30 PRINT A
 40 USING
 50 PRINT A
```

Ausgabe bei Zeile 30

|       |
|-------|
| 123.4 |
|-------|

ENTER

In Zeile 40 wird auf die automatische Formatierung zurückgeschaltet.

Ausgabe bei Zeile 50

|            |
|------------|
| 123.456789 |
|------------|

- b) Die Ausgabe bei unterschiedlicher Formatierung in Zeile 20

| Format             | Ausgabe       |
|--------------------|---------------|
| < Standardformat > | 123.456789    |
| "###.##"           | Fehlermeldung |
| "####"             | 123           |
| "####."            | 123.          |
| "####.##"          | 123.45        |
| "#.###^"           | 1.234E 02     |
| "#.#####^"         | 1.23456E 02   |

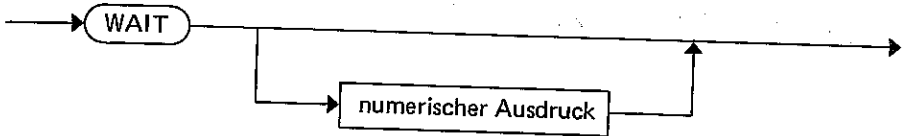
```
c) 10 X = SQR 3
 20 A$ = "AUSGABE"
 30 F$ = "&&&&###"
 40 USING F$: PRINT A$, -X
 50 USING
 60 PRINT A$: X
```



# WAIT

Diese Anweisung legt das Zeitintervall fest, das vergehen soll, bevor nach einer PRINT-Anweisung der Programmablauf wieder startet.

## Syntax



Normalerweise wird der Programmablauf nach einer PRINT-Anweisung erst mit der Eingabe von **ENTER** fortgesetzt. Die WAIT-Anweisung gestattet es, ein Zeitintervall vorzugeben, nach dessen Ablauf die Ausführung automatisch wieder gestartet wird.

Der numerische Ausdruck legt das Intervall fest. Gemessen wird in Einheiten von ca. 1/64 sec. Die Werte des Ausdrucks dürfen variieren von 0 bis 65535. Die Wartezeit ist damit  $n * 1/64$  und liegt zwischen 0 sec und 1023,9 sec.

Die WAIT-Anweisung wirkt auf alle folgenden PRINT-Anweisungen.

Gibt man keinen Ausdruck an, so wird die Automatik abgeschaltet und die Ausführung muß wieder mit **ENTER** fortgesetzt werden.

## Beispiel:

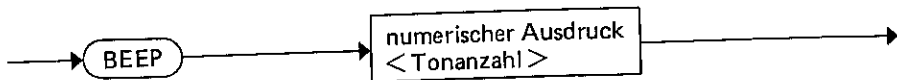
- a) 10 WAIT 54.4  
100 PRINT

Diese WAIT-Anweisung hat die gleiche Wirkung wie die PAUSE-Anweisung (s. PAUSE).

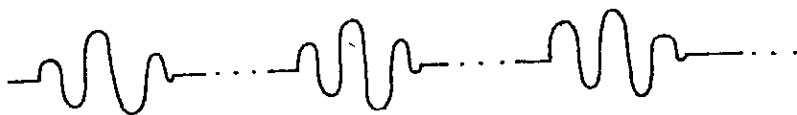
# BEEP

Mit dieser Anweisung wird der akustische Signalgeber gesteuert.

## Syntax



Das Signal besteht aus einer von kurzen Pausen unterbrochenen Folge von Tönen gleicher Frequenz.



## Beispiel:

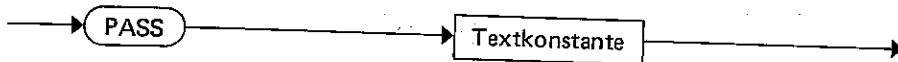
```
a) 10 A = 10
 20 B = 20
 30 C = SQR (AA+BB)
 40 BEEP 5
 50 PRINT C
 60 END
```

Die Ausgabe des Ergebnisses in Zeile 50 wird durch 5 Signaltöne angekündigt.

# PASS (Losungswort)

Diese Anweisung schützt ein Programm vor dem Zugriff unbefugter Personen.

## Syntax



Zum Schutz der einzelnen Programme vor unberechtigter Benutzung, dem Auflisten oder Verändern können diese mit einem Passwort versehen werden.

Bei der Verwendung eines Passworts werden die Kommandos LIST und LLIST nicht ausgeführt, alle Editierfunktionen werden abgeschaltet, Zeilennummern können nicht gelöscht oder eingefügt werden.

Das Passwort kann aus bis zu 7 Zeichen bestehen, überzählige Zeichen werden ignoriert.

Das Passwort schützt den gesamten Programmspeicher. Auch wenn mehrere Programme gespeichert sind, kann nur ein Passwort zugeordnet werden.

Wird nach der Programmeingabe ein Passwort eingegeben, ist der Programmspeicher für weitere Programmeingaben gesperrt.

Das Passwort wird gelöscht, indem es nochmals eingegeben wird.

Ein Passwort kann nicht überschrieben werden. Ist ein Passwort definiert, führen weitere Passwort-Eingaben zur Fehlermeldung 9.

Ist das Passwort verloren gegangen, kann es nur durch das Kommando NEW gelöscht werden. Dabei geht auch das Programm verloren.

Folgende Regeln sind bei der Verwendung des Passworts zu beachten:

1. Das Passwort kann nur direkt in PRO oder RUN eingegeben werden. Es darf also nicht in einer Programmzeile stehen.
2. Ein Passwort wird gelöscht, indem es in der gleichen Zeichenfolge nochmals eingegeben wird bzw. durch das Kommando NEW, wodurch auch das Programm gelöscht wird.



3. Ist ein Programm durch ein Passwort geschützt, kann es **nicht** auf die Kasette geladen werden.
4. Ist ein Programm nicht geschützt, kann es als geschütztes Programm auf die Kasette geladen werden (s. CSAVE).
5. Beim Rückladen von Programmen von der Kasette in den Computer ergeben sich folgende Abhängigkeiten:

Es bedeuten –

- $\emptyset$  = kein Programm im Computer
- 1 = geschütztes Programm im Computer
- 2 = ungeschütztes Programm im Computer
- A = geschütztes Programm auf der Kasette
- B = ungeschütztes Programm auf der Kasette

| Operation           | Hauptspeicherinhalt               |
|---------------------|-----------------------------------|
| $\emptyset$ CLOAD A | A mit Passwort von A              |
| $\emptyset$ CLOAD B | B ohne Passwort                   |
| 1 CLOAD A           | A mit Passwort von A              |
| 1 CLOAD B           | B ohne Passwort                   |
| 2 CLOAD A           | A mit Passwort von A              |
| 2 CLOAD B           | B ohne Passwort                   |
| $\emptyset$ MERGE A | unzulässig (A mit Passwort von A) |
| $\emptyset$ MERGE B | unzulässig (B ohne Passwort)      |
| 1 MERGE A           | Fehlermeldung 9                   |
| 1 MERGE B           | 1 und B mit Passwort von 1        |
| 2 MERGE A           | 2 und A mit Passwort von A        |
| 2 MERGE B           | 2 und B ohne Passwort             |

### Beispiel:

Nach Erprobung des Programms wird es geschützt durch Eingabe von

z.B. PASS "GEHEIM"

Nach der Eingabe kann das Programm nicht mehr gelistet, aber mit RUN gestartet werden.

Soll auch das unterbunden werden, kann durch ein entsprechendes Zusatzprogramm auch der Programmstart blockiert werden. Nur wenn das bestimmte Kennwort eingegeben wird, wird das Programm ausgeführt.

Wird an den Anfang eines Programms z.B.

```
5 INPUT "PRO-1 KENNWORT ? "; A$
10 IF A$ <> "GEHEIM" GOTO 5
```

gesetzt, und am Ende wieder ein Passwort eingegeben, wird das Programm ohne Kenntnis des Kennworts nur bis Zeile 10 abgearbeitet.

### Hinweis:

Das Passwort und das Kennwort können unterschiedlich sein, weil sie programmtechnisch in keiner Beziehung stehen.

## 12.9 TEXTFUNKTIONEN

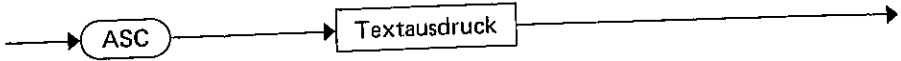
Mit dem PC-1251 können Sie nicht nur rein numerische Aufgaben lösen, sondern auch Texte verarbeiten. Die im folgenden Kapitel beschriebenen Funktionen sollen Ihnen diese Arbeit erleichtern. So können Sie beispielsweise Zeichenfolgen aus Texten heraustrennen und zu neuen zusammensetzen.

Die Syntax erläutert die Struktur der Parameterliste. Die Ergebnisse der Funktionen sind entweder Zeichenketten, die Textvariablen zugewiesen oder in Textatdrücken weiterverarbeitet werden können, oder numerische Werte, die sich in numerischen Ausdrücken verarbeiten oder numerischen Variablen zuweisen lassen.

# ASC

Die ASC-Funktion bestimmt den ASCII-Code von Zeichen.

## Syntax



Ergebnis: numerischer Wert

Der ASCII-Code des **ersten Zeichens** vom Textausdruck ist der Ergebniswert. Die Umkehrfunktion zu ASC ist die Textfunktion CHR\$.

## Beispiele:

- a) 10 A = ASC "A"  
20 PRINT A

Ausgabe bei Zeile 20

65.

65 ist der ASCII-Code des Zeichens A.

- b) 10 A\$ = "SHARP"  
20 A = ASC A\$  
30 PRINT A

Ausgabe bei Zeile 30

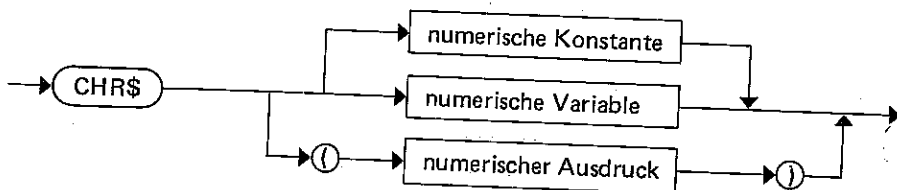
83.

83 ist der ASCII-Code des Zeichens S, dem ersten Zeichen in der Textvariablen A\$ (s. Tabelle im Anhang).

# CHR\$

Die Textfunktion CHR\$ ermittelt aus dem ASCII-Code das zugehörige Zeichen.

## Syntax



Ergebnis: Textzeichen

Der Wert des numerischen Ausdrucks im Intervall

$$32 \leq \langle \text{Ausdruck} \rangle \leq 96$$

definiert den ASCII-Code eines Zeichens. Das Funktionsergebnis ist das zugehörige Zeichen.

Die Funktion CHR\$ ist die Umkehrfunktion zur ASC-Funktion.

## Beispiele:

a) Aufstellung einer Code-Tabelle

```
10 FOR A=33 TO 96
20 PAUSE A;"-----";CHR$ A
30 NEXT A
```

Ausgabe bei Zeile 20

33.-----!

Durch die PAUSE-Anweisung in Zeile 20 wird das Programm nach 0.85 sec automatisch wieder gestartet.

34. --- = --- "

bis Schleifenende = 96

96. --- = --- E

**Hinweis:**

Nicht alle Codes entsprechen darstellbaren Zeichen, wie Sie aus der ASCII-Tabelle im Anhang ansehen können.

```
b) 5 DIM C$(0): C$(0) = " "
10 FOR B=65 TO 80
20 C$(0) = C$(0) + CHR$ B
30 NEXT B
40 PRINT C$(0)
```

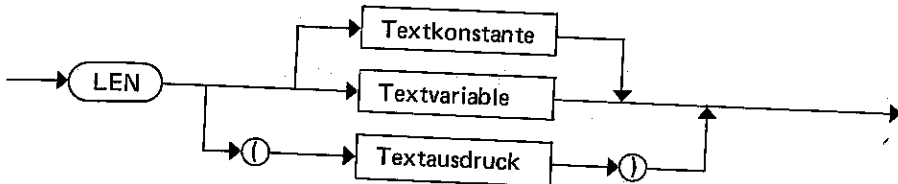
Ausgabe bei Zeile 40

ABCDEFGHIJKLMNOP

# LEN

Die LEN-Funktion berechnet die Anzahl der Zeichen eines Textausdruckes.

## Syntax



Ergebnis: numerischer Wert

Die LEN-Funktion berechnet die Zeichen eines Textausdruckes; das Ergebnis ist die Zeichenanzahl.

## Beispiel:

- a) 1Ø A = LEN "SHARP"  
2Ø B\$ = "HAMBURG"  
3Ø C = LEN B\$  
4Ø PRINT A;C

Ausgabe bei Zeile 4Ø

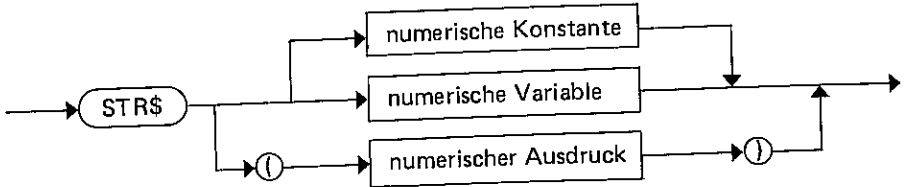
5.7.

5 ist die Anzahl der Zeichen von SHARP, 7 die von HAMBURG.

# STR\$

Die Textfunktion STR\$ wandelt den Wert eines numerischen Ausdrucks in die dezimale Zeichenfolge um.

## Syntax



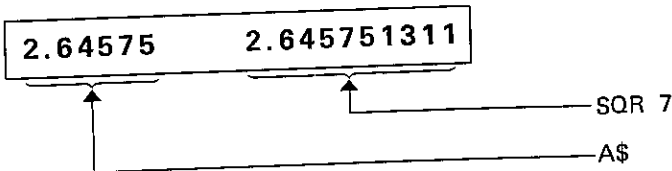
Ergebnis: Text

STR\$ ist die Umkehrfunktion von VAL.

## Beispiel:

- a) 10 A\$ = STR\$ SQR 7  
20 PRINT A\$, SQR 7

Ausgabe bei Zeile 20

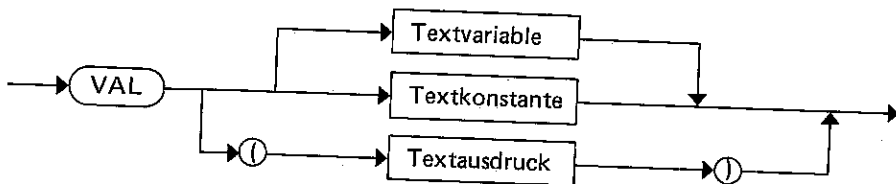


In Zeile 10 wird der Wert von  $\text{SQR } 7 = 2.645751311$  in eine Zeichenfolge umgewandelt und der Variablen A\$ zugewiesen, wobei A\$ maximal 7 Zeichen aufnehmen kann.

# VAL

Die VAL-Funktion berechnet den Wert von einer als Zeichenfolge angegebenen Zahl.

## Syntax



Ergebnis: numerischer Wert

VAL ist die Umkehrfunktion von STR\$.

Die Konvertierung beginnt mit dem ersten Zeichen des Textes und wird abgebrochen, wenn ein Zeichen auftritt, das nicht in der Liste (0 . . . 9, . und E) enthalten ist. Die Zeichen +, - sind nur als Vorzeichen zulässig, Zwischenräume werden überlesen.

## Beispiele:

a) 10 A = VAL "12.3E12"  
20 PRINT A

Ausgabe bei Zeile 20

1.23E 13

b) 10 A = VAL "1 2 3"  
20 PRINT A

Ausgabe bei Zeile 20

123.



c) 10 A\$="4 7 1 1"  
20 B=VAL A\$  
30 PRINT B

Ausgabe bei Zeile 30

4711.

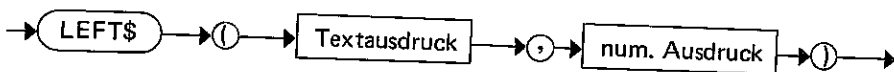
**Hinweis:**

einem Zeichen-String enthaltene Leerstellen werden normalerweise als nicht existent angesehen. Wenn jedoch eine Leerstelle im dem Exponenten-Teil entsprechenden Abschnitt (rechts von **E**) enthalten ist, so werden alle Zeichen-Strings rechts von der Leerstelle ignoriert.

# LEFT\$

Die Textfunktion LEFT\$ entnimmt aus einer Zeichenfolge die ersten Zeichen von links.

## Syntax



Ergebnis: Text

Der Wert des numerischen Ausdrucks bestimmt die Anzahl der Zeichen, die von links aus der Zeichenkette des Textausdrucks entnommen werden, und die die Ergebniszeichenfolge bilden.

## Beispiel:

- a) 10 DIM B\$(1)
- 20 B\$(0) = "SHARP HAMBURG"
- 30 B\$(1) = LEFT\$(B\$(0), 5)
- 40 PRINT B\$(1)

Ausgabe bei Zeile 40

SHARP

Aus der Zeichenkette von B\$(0) werden in der Zeile 30 fünf Zeichen entnommen und der Variablen B\$(1) zugewiesen.

- b) 10 A\$ = "2000": B\$ = "HAMBURG"
- 20 C\$ = LEFT\$(B\$, 4)
- 30 PRINT A\$; " "; C\$

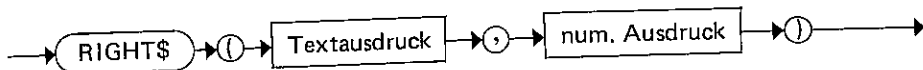
Ausgabe bei Zeile 30

2000 HAMB

# RIGHT\$

Die Textfunktion RIGHT\$ entnimmt aus einer Zeichenfolge die Zeichen von rechts.

## Syntax



Ergebnis: Text

Wie bei der LEFT\$-Textfunktion bestimmt der Wert des numerischen Ausdrucks die Anzahl der Zeichen, die aus der Zeichenkette des Textausdruckes von rechts entnommen werden, und die die Ergebniszeichenfolge bilden.

## Beispiel:

- a) 1Ø DIM B\$(1)
- 2Ø B\$(Ø) = "2ØØØ HAMBURG"
- 3Ø B\$(1) = RIGHT\$(B\$(Ø), 7)
- 4Ø PRINT "ORT "; B\$(1)

Ausgabe bei Zeile 4Ø

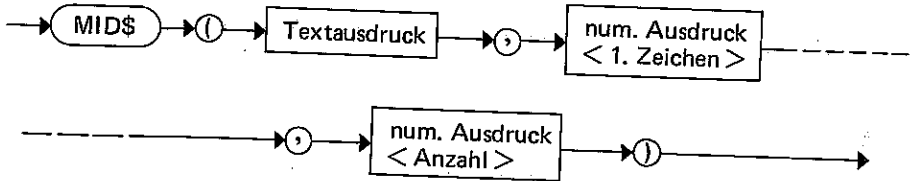
**ORT HAMBURG**

In Zeile 3Ø werden aus B\$(Ø) die letzten 7 Zeichen entnommen und der Variablen B\$(1) zugewiesen.

# MID\$

Die Textfunktion MID\$ entnimmt einer Zeichenkette den mittleren Teil.

## Syntax



Ergebnis: Text

Aus der Zeichenkette des Textausdruckes wird der mittlere Teil entnommen. Der Wert vom numerischen Ausdruck < 1. Zeichen > legt die Position innerhalb der Zeichenketten fest, von der ab die Zeichen entnommen werden sollen. Der Wert vom Ausdruck < Anzahl > bestimmt die Anzahl der Zeichen, die entnommen werden.

## Beispiel:

- a) 10 DIM B\$(1) \* 20  
20 B\$(0) = "TEL. 049-40-23775-1"  
30 B\$(1) = MID\$(B\$(0), 10, 8)  
40 PRINT B\$(1); " DURCHW. 279"

Ausgabe bei Zeile 40

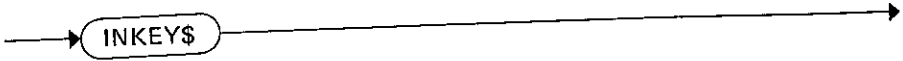
**40-23775 DURCHW. 279**

In Zeile 30 werden beginnend mit dem 10. Zeichen 8 Zeichen entnommen und der Variablen B\$(1) zugewiesen.

# INKEY\$

Die Zeichenfunktion INKEY\$ liefert das Zeichen einer Taste als Funktionswert.

## Syntax



gebnis: Text

Trifft der Programmablauf auf die INKEY\$-Zeichenfunktion, so wird die Ausführung nicht unterbrochen. Das im Moment eingetastete Zeichen wird dem INKEY\$ zugewiesen. Wird keine Taste betätigt, so erhält INKEY\$ das ASCII-Code Zeichen "NULL".

## Beispiel:

```
a) 10 A$=" "
20 A$=INKEY$
30 IF A$="E" THEN 900
40 IF A$="S" THEN 60
50 GOTO 20
60 PRINT A$; "START <PROGRAMM>"

700 GOTO 20

900 PRINT "ENDE"
910 END
```

Ist bei der Abarbeitung der Programmzeile 20 die Taste **E** oder **S** gedrückt, verzweigt sich das Programm gemäß den Anweisungen in Zeile 30 und 40. Ist keine oder eine andere Taste als **E** und **S** gedrückt, springt die Programmausführung zurück zu Zeile 20.

Mit der Taste **S** wird ein Programmdurchlauf gestartet, mit **E** wird das Programm beendet. Nach jedem einzelnen Durchlauf muß es durch **S** wieder neu gestartet werden.

# 13. Der ReSeRve-Mode

Der Reserve-Speicher des PC-1251 umfaßt 48 Bytes. In diesen Speicher kann man häufig benutzte Ausdrücke (BASIC-Anweisungen, Funktionen) eingeben. Der Zugriff erfolgt über die 18 Reserve-Tasten A, S, D, F, G, H, J, K, L, =, Z, X, C, V, B, N, M, SPC.

Zur Belegung der Tasten muß die Betriebsart RSV mit dem Schiebeschalter rechts der Anzeige eingestellt werden.

Über NEW **ENTER** kann man den gesamten Reserve-Speicher löschen.

Der Aufruf der einzelnen Tasten geschieht wie folgt:

## Beispiel:

Eingabe: **SHIFT** **A**

Ausgabe

A: \_

Jetzt kann die Programmierung erfolgen. Jede Eingabe muß mit **ENTER** abgeschlossen werden.

## Beispiel:

Eingabe: PRINT **ENTER**

Ausgabe:

A: PRINT

Hinter dem Doppelpunkt erscheint nach Beendigung der Eingabe ein Leerzeichen; dieses zeigt an, daß die Eingabe vom Computer angenommen wurde.

Die Tastenbelegung kann geändert werden, indem man über **SHIFT** < Reserve-Taste > im Reserve-Mode die zu ändernde Programmierung in die Anzeige bringt. Jetzt schiebt man den Cursor auf die zu ändernde Stelle und editiert wie gewohnt. Gelöscht werden kann die Belegung der einzelnen Tasten durch Überschreiben mit Leerräumen. Jede Eingabe muß wieder mit **ENTER** abgeschlossen werden.

Im Reserve-Mode kommt dem Zeichen @ eine besondere Bedeutung zu. Es dient bei einer Programmierung der reservierbaren Tasten als Platzhalter für **ENTER**. Das gestattet es, Programme auch über die reservierbaren Tasten zu starten.

### Beispiel:

Eingabe: SHIFT A

Ausgabe:

A: \_

Gewünschte Programmierung: RUN < Zeilennummer > ENTER

Eingabe

A: RUN 100 @

ENTER

Ein Programm beginnend mit der Zeile 100 kann jetzt im RUN-Mode über folgende Tasten gestartet werden:

SHIFT A

Hat man die Taste nur mit dem BASIC-Befehl RUN < Zeilennummer > programmiert, ist es erforderlich zum Starten des Programms

SHIFT

< Reserve-Taste >

ENTER

einzugeben.

### **Schablonen**

Zum Lieferumfang des **PC-1251** gehören zwei Schablonen, die über die Definable Keys/Reserve-Tasten gelegt werden können. Man kann hierauf die Belegung der Reservetasten bzw. die Art der Programme, die durch den entsprechenden Definable Key gestartet werden, vermerken.

### Hinweis:

Wenn der reservierte Inhalt einer Zeichenkette (String) einschließlich "TO" (außer bei "GOTO") angezeigt wird, (ohne Cursor) darf man die ENTER-Taste nicht drücken.

# 14. CE-125 (OPTION)

## DRUCKER UND MIKROKASSETTENREKORDER MIT INTEGRIERTEM INTERFACE FÜR EXTERNE KASSETTENREKORDER

Die Kommandos für den Drucker stehen nur mit dem als Option erhältlichen Drucker/Mikrocassettenrecorder CE-125 zur Verfügung. Da der Computer nicht über diese Kommandos verfügt, ist eine Programmierung damit nur möglich, wenn der CE-125 angeschlossen ist.

Desgleichen wird ein Fehler verursacht (ERROR-Code 8), wenn man das Drucken ausführt, während der Druckerschalter man den CE-125 auf OFF steht. In diesem Falle stellt man den Druckerschalter auf ON und drückt die CL-Taste. Dann d Druckvorgang noch einmal ausführen.

### 14.1 Einleitung

Der CE-125 ist ein Zusatzgerät zum PC-1251 und dient

1. Als weitere Datenausgabeeinheit zum Ausdrucken von:
  - a) Programmlisten
  - b) Datenlisten
  - c) Ergebnisausgabe bei Programmausführung
  - d) Ergebnisausgabe bei manuellem Rechnen
2. Als externer Programm- und Datenspeicher auf Magnetband.

Der Kassettenbetrieb ermöglicht folgende Betriebsarten:

- a) Manuelles Abspeichern von Programmen
- b) Manuelles und programmkontrolliertes Rückladen von Programmen
- c) Manuelles und programmkontrolliertes Abspeichern von Daten
- d) Manuelles und programmkontrolliertes Rückladen von Daten
- e) Manuelles Rückladen von 2 oder mehr Programmen
- f) Vergleichen von Programmen
- g) Abspeichern von Reserveausdrücken
- h) Manuelles Rückladen von Reserveausdrücken

und Rückladen von Programmen und Daten von einem externen Bandgerät.



## 14.2 Anschluß des CE-125 an den Computer PC-1251

Beim Anschluß des **CE-125** verfahren Sie unbedingt wie nachstehend beschrieben:

1. **PC-1251** ausschalten.

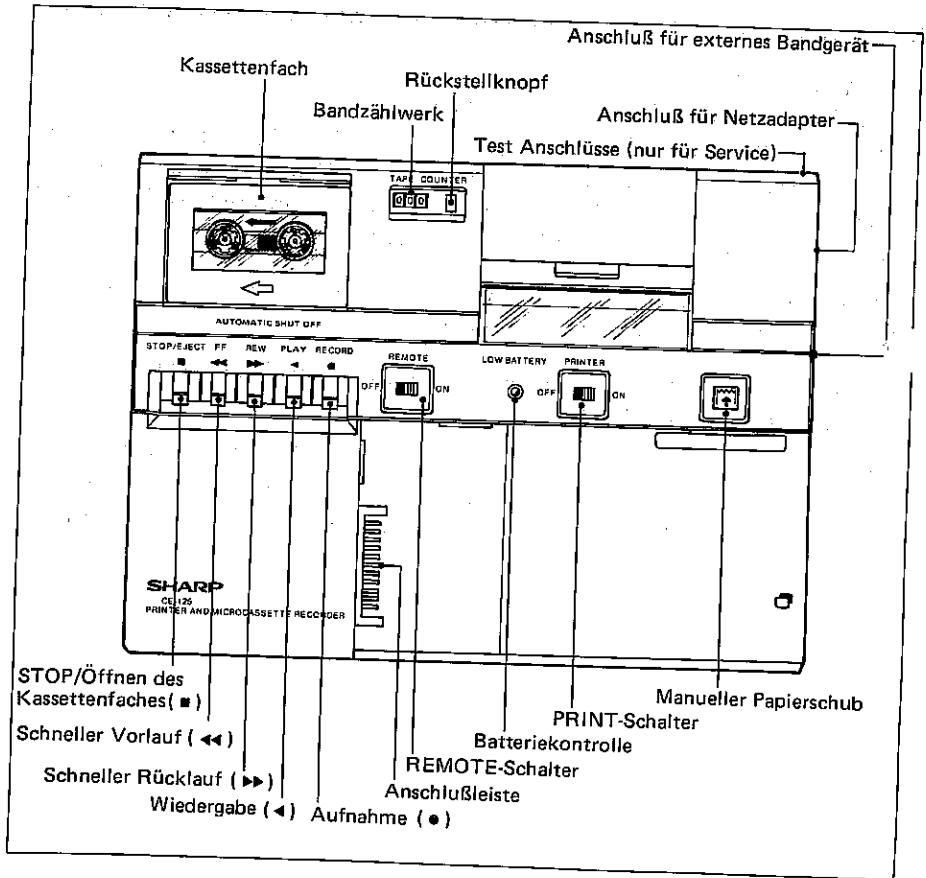
PRINT-Schalter des **CE-125** auf OFF schalten.

**Achtung:** Wird das nicht beachtet, kann der **PC-1251** "abstürzen". Dadurch werden alle Bedienungstasten blockiert, und der Computer kann nicht mehr bedient werden. Der **PC-1251** muß dann durch Betätigen des "ALL-RESET"-Schalters auf der Unterseite des Gerätes initialisiert werden.

Außerdem kann die manuelle Druckfunktion nicht angewählt werden. In diesem Falle muß die **[CL]** -Taste betätigt werden.

2. Sie entfernen am **PC-1251** die Abdeckkappe der Buchsenleiste und rasten sie auf der Unterseite des **CE-125** ein.
3. Sie legen den Computer so auf den **CE-125**, so daß die linke Seite des Computers abschließt, die Führungsstege in die Nut am **PC-1251** eingreifen und der Computer auf dem **CE-125** aufliegt. Dann schieben Sie den Computer mit leichtem Druck auf die Steckerleiste.

## 14.3 Bedienungs- und Kontrolleinrichtungen



## 14.4 Betriebsvorbereitung

Der CE-125 hat keinen Ein/Aus-Schalter. Für den Druckbetrieb muß lediglich der PRINT-Schalter auf ON gestellt werden, für den Kassettenbetrieb wird die gewünschte Kassettenfunktionstaste gedrückt. Steht der REMOTE-Schalter auf ON wird die Funktionsausführung vom PC-1251 gesteuert.

## 14.5 Auswechseln der Papierrolle

- Sie schalten den PRINT-Schalter auf ON
- Sie öffnen die Papierabdeckung
- Sie begradigen den Anfang der Papierrolle mit einer Schere und führen ihn gerade in den Papierschlitz ein.
- Sie drücken die Papiervorschubtaste bis das Papier an der Abreißschiene sichtbar wird.
- Sie legen die Rolle in die Papierwanne und schließen die Abdeckung.

**Achtung:** Ist der Papieranfang nicht begradigt oder wird das Papier schief eingeführt etc., kann ein Papierstau eintreten. Zur Vermeidung von Druckwerkschäden unterbricht eine Schutzschaltung die Stromversorgung des Motors. Die Papiervorschubtaste ist dann blockiert bzw. bei Druckerangaben wird ERROR 8 angezeigt. Sie ziehen in diesem Fall das Papier vorsichtig zurück (es geht ziemlich schwer), schalten den PRINT-Schalter auf OFF und wieder ON und drücken die **[CL]**-Taste und fahren bei Punkt c) fort.

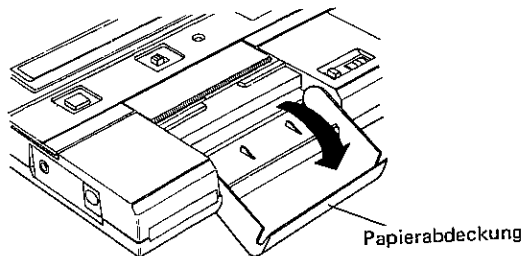


Abb. 1

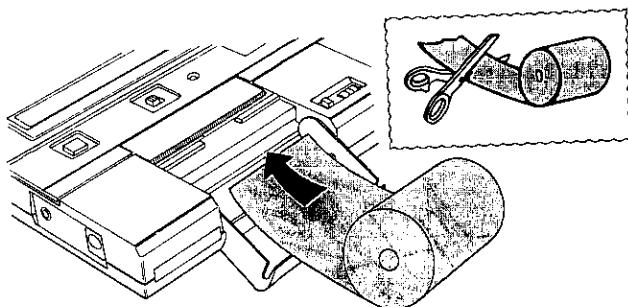


Abb. 2

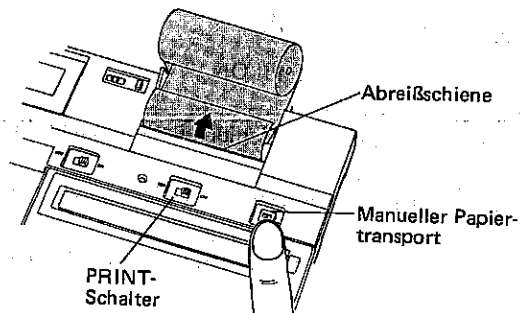


Abb. 3

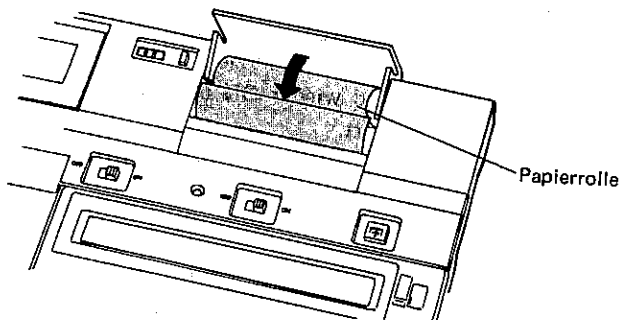
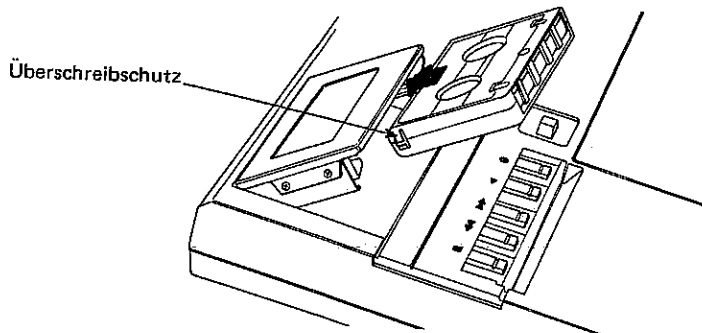


Abb. 4

## 14.6 Einsetzen der Mikrokassette

- a) Sie drücken die STOP/EJECT-Taste.
- b) Sie schieben die Kassette auf die Führungsstege der geöffneten Kassettenfachabdeckung, das Tonband zeigt nach vorn (siehe Abb.)
- c) Sie schließen die Kassettenfachabdeckung durch leichten Druck.



### Hinweis:

Die Mikrokassetten sind mit einem Überschreibschutz ausgestattet. Er befindet sich für die A – und B – Seite jeweils an der Seitenfläche der Kassette in Form einer kleinen Plastikzunge; eine Marke bezeichnet die Zuordnung.

Soll eine Programm-Kassette gegen ungewolltes Überschreiben geschützt werden, kann das durch Herausbrechen einer oder beider Plastikzungen geschehen.

Mit einer so präparierten Kassette läßt sich die RECORD-Taste nicht mehr drücken; die Aufnahmefunktion (CSAVE und PRINT #) ist gesperrt.

Sollen auf der Kassette zu einem späteren Zeitpunkt weitere Programme abge-ichert werden, können die herausgebrochenen Zungen durch einen schmalen Klebestreifen ersetzt werden. Der Überschreibschutz ist damit aufgehoben.

## **14.7 Stromversorgung und Fehlermeldung 8**

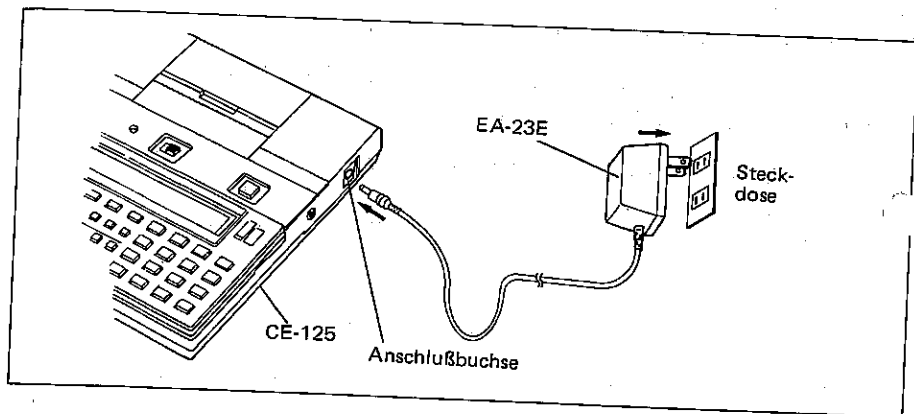
Der **CE-125** wird mit wiederaufladbaren Batterien (Ni-Cd) betrieben. Mit einer Vollladung können ca. 2000 Zeilen gedruckt bzw. ca. 4 Stunden Kassettenbetrieb durchgeführt werden. Sind die Batterien erschöpft, leuchtet die Ladezustandskontrolle "LOW BATTERY" auf, bei Druckbetrieb wird gleichzeitig ERROR 8 gemeldet. Sie löschen die Fehlermeldung mit der **[CL]** -Taste und schalten den PRINT-Schalter auf OFF/ON. Dabei erlischt die "LOW BATTERY"-Anzeige. Schalten Sie den **PC-1251** aus und schließen Sie das Ladegerät EA-23E wie unten abgebildet an den **CE-125** an.

**Verbinden Sie nun zuerst das Ladegerät mit dem CE-125, erst dann stecken Sie den Stecker des Ladegerätes in die Steckdose.**

Danach kann der Druck- bzw. Kassettenbetrieb wieder aufgenommen werden. Die Aufladezeit der Batterien beträgt ca. 15 Stunden.

### Hinweis:

Zur Entlastung der in dem **PC-1251** eingebauten Lithiumbatterie übernimmt der **CE-125** die Stromversorgung für den **PC-1251** dann, wenn die Batteriespannung des **CE-125** höher ist als die des **PC-1251**. Das ist speziell dann der Fall, wenn das Ladegerät **EA-23E** angeschlossen ist.



## 14.8 Druckerausgaben

Nachdem der **PC-1251** und **CE-125** gekoppelt wurden, wird der Drucker initialisiert. Dazu schalten Sie den **PC-1251** ein und stellen den **PRINT**-Schalter auf **ON** und drücken die **CL**-Taste.

## 14.9 Manuelles Drucken

Der manuelle Druckbetrieb ermöglicht es, Rechnungen auf dem Papierstreifen zu protokollieren. Er wird angewählt durch **SHIFT** **ENTER**. In der Anzeige des **PC-1251** erscheint der Buchstabe **P**.

Auf dem Papierstreifen wird bei der Protokollführung die gleiche Zeichenfolge in der gleichen Form und analog positioniert wie auf der Anzeige ausgedruckt.

Der Druck beginnt, wenn die **ENTER** -Taste betätigt wird. Vorher kann die Eingabe wie normal korrigiert werden. Das folgende Bild zeigt eine Protokollführung als Beispiel:

```
12000*.065
 780.
780+25.6
 805.6
SIN 45
 7.071067812E-01
F=50
 50.
X=2TF
 314.1592654
P=60X
 18849.55592
P/32
 589.0486225
```

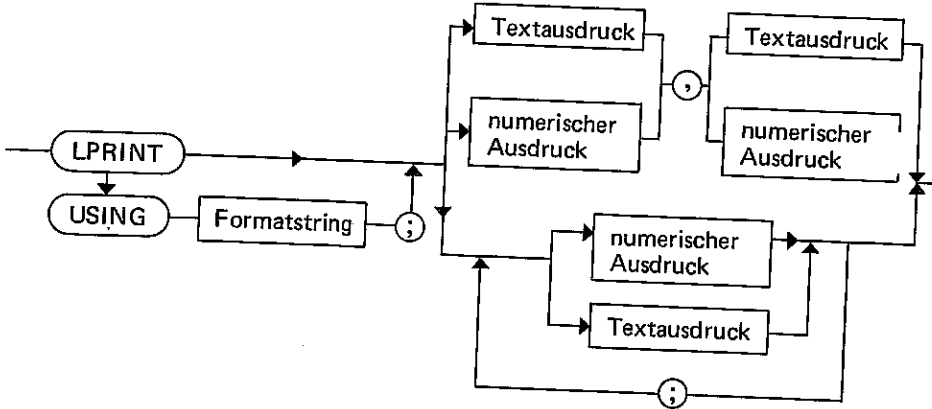
#### Hinweis:

- Bei der Protokollführung ist die manuelle Papiertransport-Taste blockiert. Sie wird erst nach Betätigen der **CL** -Taste wieder freigegeben.
- Im Falle, daß ein Papierstau einen Fehler (ERROR-Code 8) verursacht, reißt man den Papierstreifen ab und zieht den im Drucker verbliebenen Rest des Papierstreifens ganz heraus. Um den Fehlerzustand aufzuheben, drückt man dann die CL-Taste.
- Wenn der Drucker/Recorder starken äußeren elektrischen Störeinflüssen ausgesetzt ist, werden möglicherweise bedeutungslose Zahlen ausgedruckt. In diesem Falle stoppt man durch Drücken der **BRK** -Taste den Druckbetrieb und betätigt dann die CL-Taste.  
Durch Drücken der CL-Taste wird der normale Betriebszustand des Druckers wieder hergestellt.  
(Wenn ein Papierstau auftritt, oder das Gerät während des Druckens starken elektrischen Störeinflüssen von außen ausgesetzt ist, funktioniert der Drucker möglicherweise nicht ordnungsgemäß und nur das Symbol "BUSY" wird angezeigt. In diesem Falle unterbricht man den Druckvorgang durch Betätigung der **BRK** -Taste. (Nun den Papierstau beseitigen). Dann druckt man die CL-Taste.
- Sogar beim Drucken mit dem LPRINT-Kommando kann die Eingabe durch die Anweisung INPUT, INKEY\$ oder PRINT ausgeführt werden.  
In diesem Falle wird der Druckbetrieb jedoch durch Betätigung der CL-Taste gestoppt. Daher die CL-Taste auf jeden Fall nach Ausführung des Druckvorgangs betätigen.

## 14.10 LPRINT

Mit dieser Anweisung werden numerische Werte und Zeichenfolgen auf dem Drucker ausgegeben.

### Syntax



Der USING-Teil der Anweisung spezifiziert das Ausgabeformat, wie unter USING beschrieben. Der Formatstring steht für die dort erwähnte Textkonstante bzw. Textvariable.

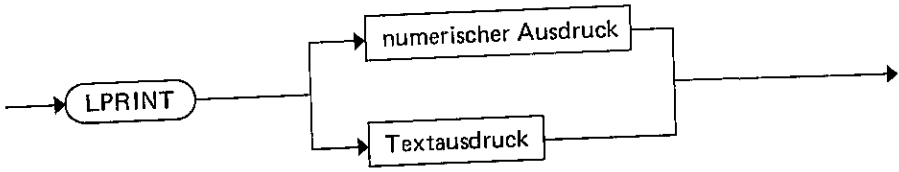
Über das Komma wird die Aufteilung der 24-spaltigen Druckzeile in zwei gleich große Felder gesteuert. Das Semikolon trennt die einzelnen Elemente einer Liste von Ausdrücken und steuert die Position des Cursors.

Wird die Cursor-Positionierung nicht explizit durch die Verwendung eines Semikolons gesteuert, werden die Werte einzelner numerischer Ausdrücke rechtsbündig und Textausdrücke linksbündig in das Feld geschrieben.

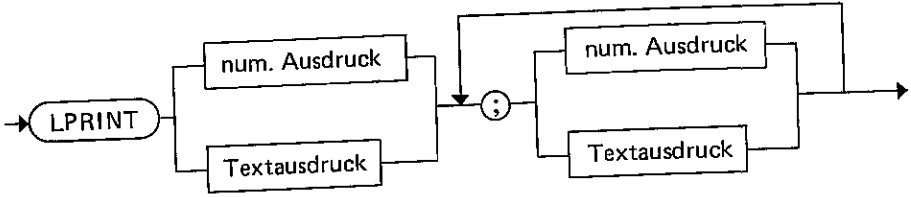
Nach dem Ausdrücken der LPRINT-Liste (< Ausdrücke >) auf dem Papierstreifen wird der Programmlauf nicht angehalten.



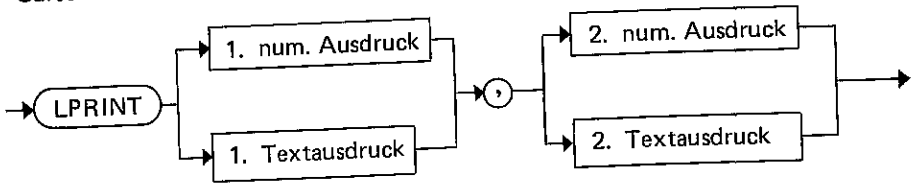
Die einzelnen Formate der LPRINT-Anweisung haben folgende Wirkung:



Der in der LPRINT-Liste spezifizierte Inhalt wird auf dem Drucker rechts- bzw. linksbündig ausgegeben.



Der Inhalt der PRINT-Liste wird ausgedruckt, beginnend mit der momentanen Cursor-Position.



Die Zeile wird in zwei Felder geteilt.

Der erste Ausdruck erscheint in der linken, der zweite in der rechten Hälfte jeweils rechts- bzw. linksbündig.

### ninweise:

1. Werden mehr als 24 Zeichen ausgegeben, wird zwei- oder mehrzeilig gedruckt. Für den Fall, daß die Ausdrücke der LPRINT-Liste durch ein Komma verbunden sind, werden die Textausdrücke auf 12 Zeichen begrenzt.
2. Wird die LPRINT-USING Anweisung benutzt, so ist das spezifizierte Format für alle LPRINT-Anweisungen bis zur nächsten USING-Anweisung gültig.
3. Das USING-Format wird durch das Kommando RUN oder **SHIFT** **CL** gelöscht.

Beispiele:

a)

```
10:A=3.1415
20:B$="SHARP"
50:LPRINT A
60:LPRINT B$
```

SHARP

3.1415

b)

```
10:A=3.1415
20:B$="SHARP"
30:C=7.89
40:D$="PC-1251"
50:LPRINT A;B$;D$;C
60:LPRINT B$;" ";D$;" "
 ;A;" ";C
```

3.1415SHARPPC-12517.89  
SHARP PC-1251 3.1415 7.8  
9

c)

```
10:A=3.1415
20:B$="SHARP"
30:C=7.89
40:D$="PC-1251"
50:LPRINT A;B$
60:LPRINT B$;A
70:LPRINT B$;D$
80:LPRINT A;C
```

3.1415SHARP  
SHARP 3.1415  
SHARP PC-1251  
3.1415 7.89

d)

```

10:A=3.1415
20:B$='SHARP'
30:C=7.89
40:D$='PC-1251'
50:E$='&&###.#'
60:F$='###.##&&'
70:LPRINT USING E$;B$;A
80:LPRINT USING F$;B$;A
90:LPRINT USING ;B$;A
100:LPRINT USING E$;A;C
110:LPRINT USING F$;B$;D
$
120:LPRINT USING "###.###
#";A;C
130:LPRINT USING ;A;C

```

```

SH 3.1
SH 3.1
SHARP3.1415
 3.1 7.8
SH PC
 3.1415 7.8900
 3.1415 7.89

```

**Hinweis:**

Ist der PRINT-Schalter auf OFF geschaltet und die Programmausführung trifft auf eine LPRINT-Anweisung, leuchtet zunächst die LOW BATTERY-Anzeige auf, etwas später wird ERROR 8 angezeigt.

Sie schalten auf PRINT ON und löschen die Fehlermeldung mit **CL** .

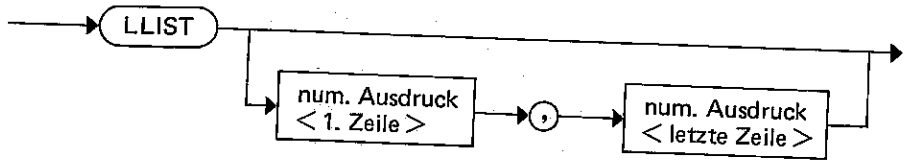
Automatischer Papiervorschub.

Neben dem manuellen Papiervorschub kann auch ein automatischer Zeilenvorschub durchgeführt werden.

Der automatische Zeilenvorschub ist nur unter Programmkontrolle möglich; dazu geben Sie z.B. die Anweisung 10 LPRINT "" .

# LLIST

Mit dieser Anweisung wird das Programm auf dem Drucker gelistet.



Die LLIST-Anweisung hat die gleiche Funktion wie die LIST-Anweisung. Das Programm wird jedoch auf dem Drucker und nicht auf der Anzeige aufgelistet.

Der 1. Ausdruck spezifiziert die erste, der 2. Ausdruck spezifiziert die letzte auszudruckende Zeile.

Sind die spezifizierten Zeilennummern nicht existent, wird das Listing mit der nächsthöheren existierenden Zeilennummer begonnen bzw. beendet.

Wenn keine Zeilennummern spezifiziert sind, werden alle Programmzeilen ausgedruckt.

# 15. Datenspeicherung auf Magnetband

Der im CE-125 integrierte Microkassettenrecorder ermöglicht die in der Einleitung aufgezählten Betriebsarten.

Die Rekorderfunktionen werden gesteuert durch die Tasten

|            |                         |
|------------|-------------------------|
| STOP/EJECT | = STOP/Kassettenauswurf |
| FF         | = Schneller Vorlauf     |
| REW        | = Schneller Rücklauf    |
| PLAY       | = Wiedergabe            |
| RECORD     | = Aufnahme              |

Mit dem REMOTE-Schalter wird der computergesteuerte bzw. der manuelle Kassettenbetrieb gewählt. Steht der REMOTE-Schalter auf ON, werden alle oben genannten Rekorderlauffunktionen für den manuellen Betrieb gesperrt. Steht der REMOTE-Schalter auf OFF, können alle Lauffunktionen auch manuell ausgeführt werden.

Auf dem Magnetband können Sie Daten, Programme und Reserveausdrücke abspeichern oder von dort zurückladen. Außerdem können Sie das Magnetband als externen Programmspeicher verwenden, um große Programme ablaufen zu lassen.

Die Informationen aus Daten und Programmen werden in Blöcken nach dem Zweitonverfahren auf das Magnetband geschrieben.

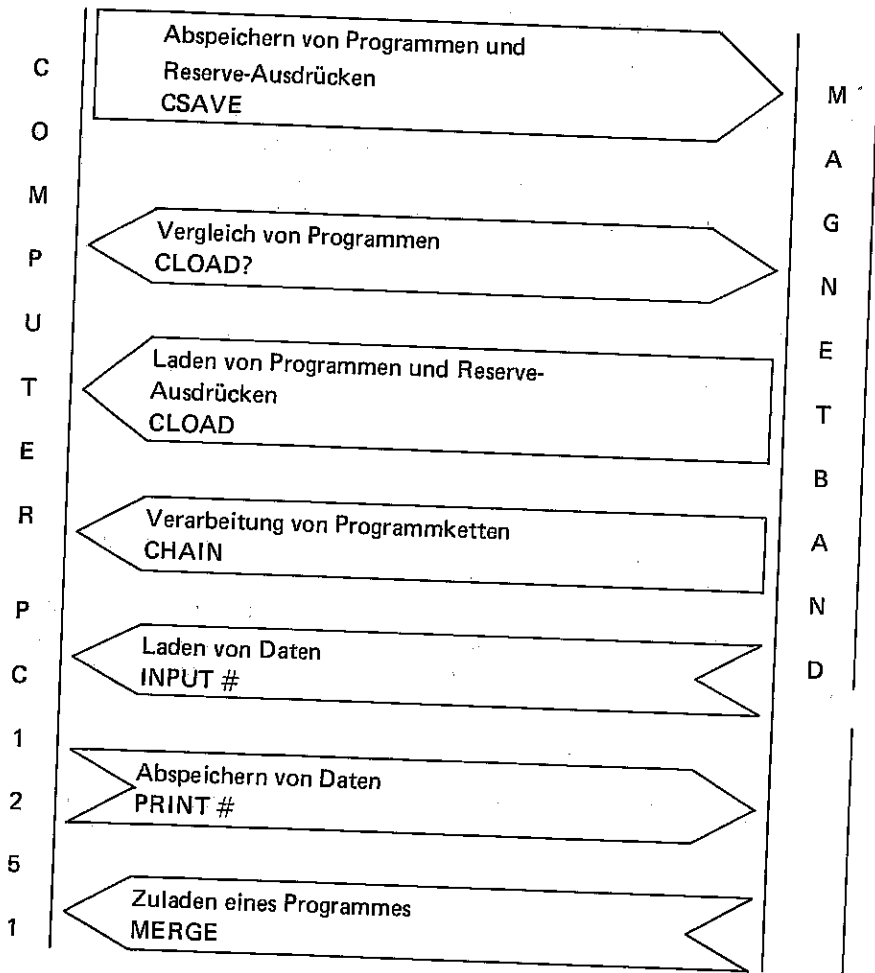
Auf einem Magnetband können sehr viele Informationen gespeichert werden. Damit Sie die Daten oder Programme wiederfinden und unterscheiden können, beginnt jeder Block mit einem Blocknamen, der aus max. 7 Zeichen besteht. Abgeschlossen wird der Block, wenn die gesamte Information abgespeichert ist. Um eine sichere Trennung der Blöcke zu gewährleisten schreibt der Computer vor dem Blocknamen automatisch etwa 5 bis 8 Sekunden einen konstanten Signalton. Ein kompletter Block hat damit folgende Form:

|  |           |           |                      |  |
|--|-----------|-----------|----------------------|--|
|  | Signalton | Blockname | Programme oder Daten |  |
|--|-----------|-----------|----------------------|--|

**Hinweis:**

Auf die Informationen auf dem Band können Sie nur sequentiell zugreifen. Damit Sie Ihre Daten und Programme schneller wieder auffinden, sollten Sie sich vor dem Abspeichern von Informationen den Zählerstand des Kassettenrekorders und den Blocknamen der Daten aufschreiben. Sie werden sehr rasch eine umfangreiche Bibliothek von Daten erhalten.

Folgende Möglichkeiten werden beim Bandbetrieb geboten:



## 15.1 Benutzung des Magnetbandes

Nachdem Sie Ihr Programm so wie normalerweise auch eingegeben haben, können Sie den Rekorder für die Abspeicherung vorbereiten. Sie müssen dafür folgendes tun:

- a) Stellen Sie den REMOTE-Schalter des CE-125 auf OFF.
- b) Legen Sie eine Kassette in den Rekorder ein und vergewissern Sie sich, daß das Band zurückgespult ist.
- c) Stellen Sie das Bandzählwerk auf Null.
- d) Suchen Sie sich eine geeignete Stelle auf dem Band und merken Sie sich den Zählerstand (ist das Band leer, zu Beginn des Bandes)
- e) Stellen Sie den REMOTE-Schalter auf ON.
- f) Bereiten Sie die Aufnahme vor durch Drücken der RECORD-Taste.
- g) Kommando-Eingabe:  
Mit dem Abspeicherkommando CSAVE können Sie Ihr Programm gleichzeitig mit einem Namen, dem Blocknamen versehen.  
Geben Sie ein: CSAVE "PROG"

Wenn Sie die **ENTER** -Taste betätigt haben, setzt sich das Magnetband in Bewegung. Ihr Programm wird jetzt übertragen und unter dem Namen "PROG" abgespeichert. Zu Beginn hören Sie 5 bis 8 Sekunden den konstanten Signalton, anschließend eine Folge von Tönen. Sobald der Computer am Ende des Programmes angelangt ist, hört das Band auf zu laufen; das Bereitschaftssymbol erscheint wieder auf der Anzeige.

### Hinweise:

Um ein Programm auf einer bespielten Cassette zu speichern, löscht man den zur Aufnahme erforderlichen Teil (Ablauf von ca. 5 Zählwerknummern), bevor man das Kommando für Aufnahme eingibt und ausführt.  
(Auf jeden Fall sicherstellen, daß das vorherige Programm gänzlich und restlos gelöscht ist.)

## 15.2 Überprüfen der Abspeicherung und Zurückgewinnen der Information.

Nach der Abspeicherung Ihres Programmes mit dem CSAVE-Kommando und bevor Sie das Programm im Computer löschen, ist es empfehlenswert, zu überprüfen, ob es fehlerfrei übertragen wurde. Das Band könnte beispielsweise eine schadhafte Stelle haben.

Die Überprüfung ist ganz einfach und erfolgt mit dem CLOAD?-Kommando. Der Computer vergleicht das Programm in seinem Speicher mit dem auf dem Band. Wenn alles geklappt hat, wird anschließend das Bereitschaftszeichen ausgegeben. Wenn der Computer einen Unterschied feststellt, wird er einen Fehler anzeigen (ERROR 8). Löschen Sie dann diesen Teil des Bandes und laden Sie mit Hilfe c Netzadapters die Batterien auf. Gegebenenfalls nehmen Sie eine andere Bandstelle bzw. eine andere Kassette.



## 15.2.1 Vorbereitung für das CLOAD?-Kommando

- h) Schalten Sie den Rekorder auf STOP.
- i) Stellen Sie den REMOTE-Schalter auf OFF.
- j) Die Cassette, die die zum Vergleichen gewünschte Bandaufzeichnung enthält, in den Cassettenrecorder einlegen. Unter Verwendung des Bandzählwerks spult man das Band bis in die Nähe der Stelle vor, an der der zu vergleichende Inhalt aufgenommen ist.
- k) Durch Drücken der Wiedergabetaste PLAY (◀) aktiviert man die Wiedergabe-Funktion.
- l) Nach dem Piepton im signallosen Teil, bevor die Aufzeichnung des Programms bzw. der Daten ertönt, stellt man den REMOTE-Schalter auf "ON".
- m) Geben Sie ein: CLOAD?

Wenn Sie die **ENTER** -Taste gedrückt haben, setzt sich das Band in Bewegung. Sie hören die gleiche Signalfolge wie bei der Abspeicherung.

Die Programme werden verglichen und anschließend erscheint das Bereitschaftszeichen auf der Anzeige. Ist ein Fehler aufgetreten, versuchen Sie die Abspeicherung noch einmal.

Jetzt können Sie sicher sein, zu einem späteren Zeitpunkt das Programm wieder vom Magnetband zurückzugewinnen zu können.

Die Handgriffe hierzu sind die gleichen: Anstatt CLOAD? geben Sie jetzt unter Punkt m

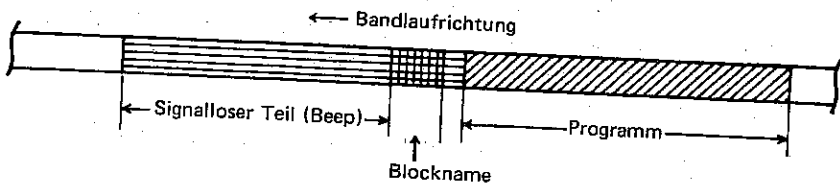
CLOAD oder CLOAD "PROG" ein.

Es wird das Programm vom Magnetband in den Speicher des Rechners übertragen. Die beiden CLOAD-Kommandoformen unterscheiden sich nur dadurch, daß bei dem zweiten Kommando der Blockname zusätzlich mit der aufgeführten Zeichenfolge verglichen wird. Nur wenn diese identisch sind, findet die Übertragung statt.

Stimmt der eingegebene Programmname und der gespeicherte Blockname nicht überein, so werden alle auf dem Band folgenden Blöcke nach diesem Programmnamen abgesucht. Wird er gefunden, findet die Übertragung des Programmes in den Computer statt.

## Vorsichtsmaßnahmen beim Mischen und Übertragen

Das Programm wird wie nachstehend dargestellt auf Band aufgezeichnet:



Bei der Wiedergabe des aufgenommenen Programms produziert der signallose Abschnitt einen Dauerton (Beep), wohingegen der Blockname und das aufgezeichnete Programm unterbrochene Signaltöne produzieren.

Für das Vergleichen oder Übertragen von Programmen spult man das Band bis zum signallosen Abschnitt (Dauerton hörbar) vor bzw. zurück und führt dann erst das Kommando (CLOAD?, CLOAD, MERGE, CHAIN oder INPUT #) aus.

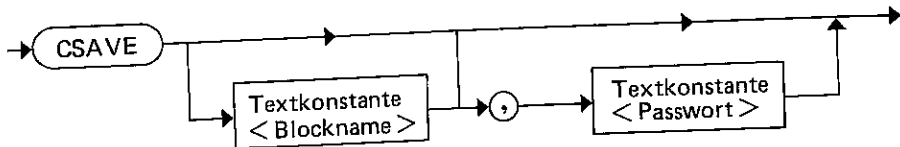
Wenn die Ausführung des Kommandos vor dem signallosen Teil beginnt, ist das Vergleichen bzw. die Übertragung gegebenenfalls nicht möglich.

In diesem Falle erscheint das "BUSY"-Symbol nicht und der Bandlauf wird nicht gestoppt. Zum Stoppen des Recorderbetriebs betätigt man in diesem Falle die **BRK**-Taste. Daraufhin versucht man die Ausführung noch einmal von Anfang an.

# CSAVE

Mit diesem Kommando werden Programme und RESERVE-Ausdrücke auf das Magnetband geschrieben.

## Syntax



Mit dem Kommando CSAVE werden Programme und Reserveausdrücke auf das Magnetband geschrieben. Die Textkonstante legt den Blocknamen bzw. das Schlüsselwort zum Programmschutz auf dem Band fest. Dabei werden nur die ersten 7 Zeichen berücksichtigt. Unter dem Blocknamen können Sie später das Programm auf der Kassette wiederfinden.

Die Überspielung dauert abhängig von der Blocklänge einige Minuten.

Es werden in PRO- und RUN-Mode das gesamte Programm, im RESERVE-Mode alle Reserveausdrücke abgespeichert.

Im Anschluß an das Speichern des Blockes sollte man mit dem Kommando CLOAD? überprüfen, ob nicht etwa bedingt durch Bandfehler Informationen verlorengegangen sind.

Soll das Programm als geschütztes Programm abgespeichert werden, muß hinter dem Blocknamen noch ein PASS-Wort benannt werden.

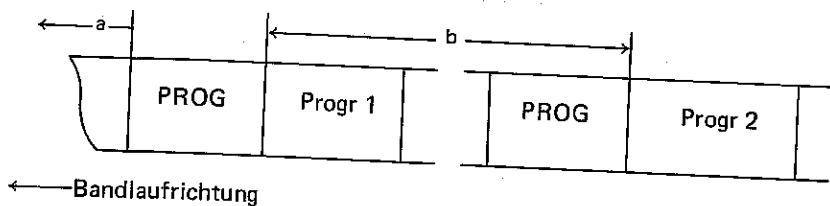
Das ist aber nur möglich, wenn das abzuspeichernde Programm selbst kein PASS-Wort hat (siehe PASS).

### Beispiel:

```
CSAVE "PRO-1"
CSAVE "PRO-1", "GEHEIM"
CSAVE, "GEHEIM"
CSAVE
```

### Hinweise:

1. Wird ein neues Programm so auf dem Band gespeichert, daß es einen anderen Block auch nur teilweise überlappt, wird die ursprüngliche Information teilweise gelöscht. Dies führt zu einem Fehler beim Laden dieses Blockes.
2. Haben zwei Blöcke den gleichen Namen, etwa "PROG", so lädt der Computer das zuerst aufgefundene Programm. Befindet sich der Tonkopf in der Darstellung im Bereich a, so wird Programm 1 geladen. Befindet er sich jedoch im Bereich b, so wird Programm 2 geladen.

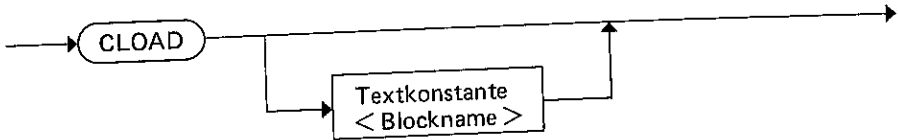


3. Zweckmäßigerweise legt man sich für jedes Band der Programmbibliothek ein Karteiblatt an mit Namen, Hinweis auf Informationstyp (Progr., Reserv., Data), Zählerstand und kurzen Programmbeschreibung. Hat man den Namen und Zählerstand vergessen, so kann es sehr mühselig und aufwendig sein, sich diese Information wieder zu besorgen. Kennen Sie den Zählerstand, so können Sie mit dem schnellen Vorlauf den Tonkopf schon richtig positionieren und gewinnen Zeit.
4. Reserveausdrücke werden nicht mit einem PASS-Wort abgespeichert.
5. Für alle Programme, die nicht mit dem END oder dem RETURN-Kommando, sondern z.B. mit GOTO "A" enden, gibt man als letzte Zeile ein END-Kommando ein. Wenn andernfalls ein Programm, das nicht mit END oder RETURN endet aufgenommen wird, so kann das im Anschluß daran aufgenommene Programm, gegebenenfalls nicht übertragen, bzw. gemischt werden.

# CLOAD

Mit diesem Kommando werden Programme und Reserveausdrücke vom Band geladen.

## Syntax



Mit dem Ladekommando CLOAD werden Programme und Reserveausdrücke vom Computer gesucht und in den Programmspeicher geladen. Das Laden ist nur manuell als direktes Kommando möglich. Programme oder Reserveausdrücke, die sich im Speicher befinden, werden vor der Ladeoperation gelöscht. Die Textkonstante bezeichnet den Blocknamen. Nach diesem Blocknamen wird auf dem Band automatisch gesucht. Ist der Block gefunden, wird er in den Speicher geladen und der Computer gibt das Bereitschaftssymbol aus. Wird keine Textkonstante angegeben, so wird der folgende Block unabhängig vom Blocknamen geladen.

Abhängig von der gewählten Betriebsart wird der Block als Programm (RUN- bzw. PRO-Mode) oder als RESERVE-Ausdruck (RESERVE-MODE) aufgefaßt.

## Hinweise:

1. Befindet sich der gesuchte Blockname nicht auf dem eingelegten Band, sucht der Computer auch dann nach dem fehlenden Namen weiter, wenn das Band abgelaufen ist. In diesem Falle muß man mit **BRK** diese Operation abbrechen.

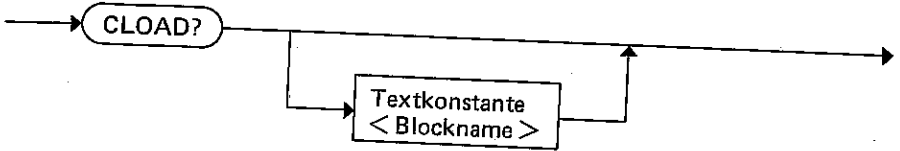
Dies gilt für die nachstehend beschriebenen Kommandos MERGE, CHAIN, CLOAD? und INPUT.

2. Wurde die falsche Betriebsart gewählt, Laden von Reserveausdrücken im PRO-MODE und umgekehrt, so wird dieser Block überlesen.
3. Wenn während der Ausführung von CLOAD oder CHAIN (Beschreibung folgt später) ein Fehler auftritt, so wird das im Computer gespeicherte Programm unbrauchbar.

# CLOAD?

Mit diesem Kommando werden Band- und Computer-Informationen verglichen.

## Syntax



Mit dem Kommando CLOAD? vergleicht man die auf dem Band gespeicherte Information des spezifizierten Blocknamens (Textkonstante) mit dem im Computer gespeicherten Programm (RUN- bzw. PRO-Mode) oder den RESERVE-Ausdrücken. Der Vergleich kann nur manuell abgerufen werden.

Der Ablauf des CLOAD?-Kommandos ist identisch mit dem CLOAD-Kommando. Werden bei dem Vergleich Fehler festgestellt, so erscheint eine Fehlermeldung (ERROR 8); sonst das Bereitschaftssymbol.

Ein evtl. benanntes PASS-Wort wird nicht geprüft.

## Beispiel:

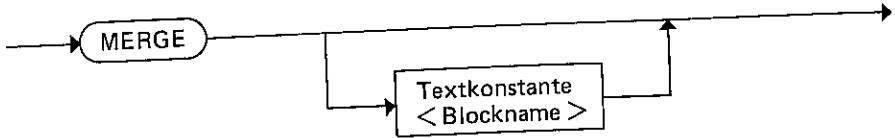
CLOAD?

CLOAD? "PROG-1"

# MERGE

Mit diesem Kommando werden Programme vom Band zu den im Computer gespeicherten hinzugeladen.

## Syntax



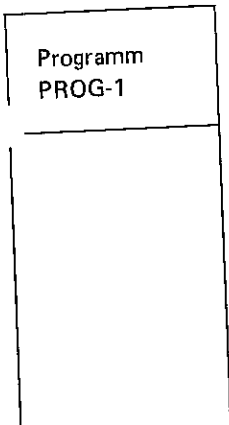
Das MERGE-Kommando hat die gleiche Funktion wie das CLOAD-Kommando, allerdings wird das alte Programm nicht gelöscht, sondern das neue hinzugefügt. Auf diese Weise wird es Ihnen ermöglicht, zwei oder mehrere Programme im Computer zu speichern.

Die Anweisungsnummern der eingelesenen Programme werden nicht verändert. Es kann also passieren, daß die gleichen Nummern mehrmals vorhanden sind.

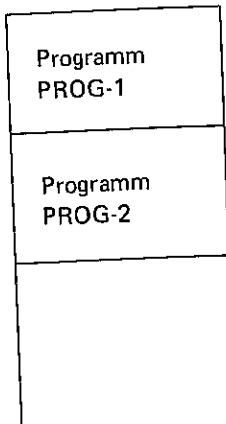
## Beispiel:

Ein Auszug aus dem Programmspeicher

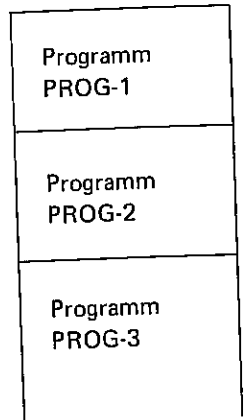
CLOAD "PROG-1"



MERGE "PROG-2"



MERGE "PROG-3"



Wie schon erwähnt, können die mit CLOAD und MERGE eingelesene Programme die gleichen oder teilweise die gleichen Zeilennummern haben (siehe Beispiel a).

In diesem Fall werden die Programme jeweils als Einzelprogramm behandelt; wird das Programmende erreicht, so wird automatisch der Programmablauf beendet. Dabei gelten folgende Regeln:

- a) Nur das zuerst eingelesene Programm (CLOAD) kann mit RUN, alle weitere Programme können nur mit RUN/GOTO < Markenname > oder DEF < Definable Key > gestartet werden.
- b) Nur das zuletzt eingelesene Programm (MERGE) kann editiert werden.
- c) Sprünge mit GOTO < Zeilennummer > werden nur im selben Programm ausgeführt.
- d) Anweisungen mit GOTO < Markenname > erlauben Sprünge in andere Programme.

Sind die mit CLOAD und MERGE eingelesenen Programme mit sequentiell hochlaufenden Zeilennummern versehen, z.B. "PROG 1" von 10 bis 200, "PROG 2" von 200 bis 300 und "PROG 3" von 310 bis 500 (Beispiel b) und in der genannten Folge eingelesen, werden alle drei wie ein Programmblock behandelt. Wird im Programmablauf eine END-Anweisung erreicht, wird er beendet.

Dabei gelten folgende Regeln:

- a) Das jeweils erste Programm kann mit RUN, die folgenden können mit RUN-< Zeilennummer > gestartet werden.
- b) Alle Programme können editiert werden.
- c) Sprünge mit GOTO < Zeilennummer > zu anderen Programmen sind möglich wie auch Sprünge mit GOTO < Markenname >.

Ist ein im Hauptspeicher befindliches Programm mit einem PASS-Wort versehen, kann kein weiteres geschütztes Programm mit MERGE eingelesen werden.

Ist ein im Hauptspeicher befindliches Programm ungeschützt und ein geschütztes Programm wird mit MERGE hinzugeladen, ist das erste Programm ebenfalls geschützt.



## Beispiel:

a)

```
10:"PROG1":P$="PROG1"
20:LPRINT "START VON "
P$
30:A=0
40:IF A>=2 GOTO 400
300:LPRINT "< PROGRAMM A
USFUEHRUNG >"
310:A=A+1: GOTO 40
400:LPRINT "ENDE VON ";P
$
410:END
10:"PROG2":P$="PROG2"
20:LPRINT "START VON "
P$
30:A=0
40:IF A>=2 GOTO 400
300:LPRINT "< PROGRAMM A
USFUEHRUNG >"
310:A=A+1: GOTO 40
400:LPRINT "ENDE VON ";P
$
410:END
10:"PROG3":P$="PROG3"
20:LPRINT "START VON "
P$
30:A=0
40:IF A>=2 GOTO 400
300:LPRINT "< PROGRAMM A
USFUEHRUNG >"
310:A=A+1: GOTO 40
400:LPRINT "ENDE VON ";P
$
410:END
415:"A" INPUT "START <PR
OG.NR.> ? " ;B
420:IF B=1 GOTO "PROG1"
430:IF B=2 GOTO "PROG2"
440:IF B=3 GOTO "PROG3"
450:END
```

PROG1,  
eingelesen mit CLOAD

---

PROG2,  
eingelesen mit MERGE

---

PROG3,  
eingelesen mit MERGE

---

Nachträglich manuell  
eingegebenes Programm

b)

```
10:"PROG1":P$="PROG1"
20:LPRINT "START VON ";
P$
30:A=0
40:IF A>=2 GOTO 190
50:LPRINT "<PROGRAMM AU
SFUEHRUNG>"
180:A=A+1: GOTO 40
190:LPRINT "ENDE VON ";P
$
200:END
210:"PROG2":P$="PROG2"
220:LPRINT "START VON ";
P$
230:A=0
240:IF A>=2 GOTO 290
250:LPRINT "<PROGRAMM AU
SFUEHRUNG>"
280:A=A+1: GOTO 240
290:LPRINT "ENDE VON ";P
$
300:END
310:"PROG3":P$="PROG3"
320:LPRINT "START VON ";
P$
330:A=0
340:IF A>=2 GOTO 490
350:LPRINT "<PROGRAMM AU
SFUEHRUNG>"
480:A=A+1: GOTO 340
490:LPRINT "ENDE VON ";P
$
500:END
```

PROG1

eingelassen mit CLOAD

-----  
PROG 2

eingelassen mit MERGE

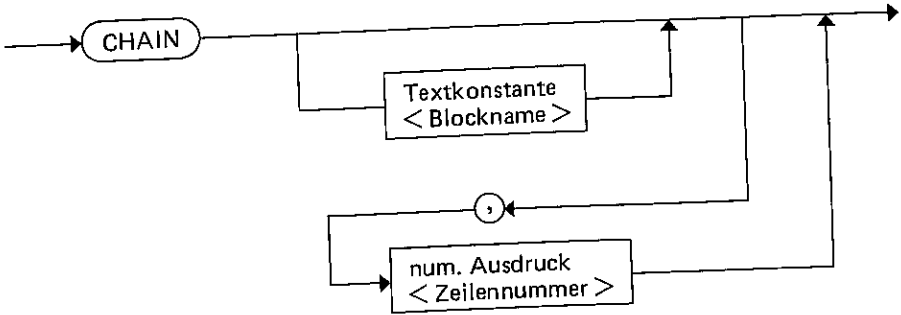
-----  
PROG 3

eingelassen mit MERGE

# CHAIN (Programmanweisung)

Mit dieser Anweisung wird eine Folge von auf dem Magnetband gespeicherten Programmen ausgeführt.

## Syntax

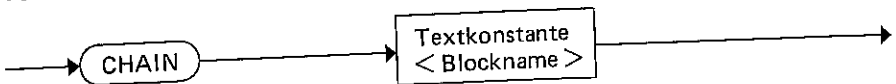


CHAIN ist eine Programmanweisung und kann nicht als manuelles Kommando verwendet werden. Die CHAIN-Anweisung ermöglicht es Ihnen, ein Programm laufen zu lassen, das so umfangreich ist, daß es nicht auf einmal in den Programmspeicher zu lassen. Solche Programme müssen unterteilt und mit dem CSAVE-Kommando einzeln auf das Band übertragen werden. Am Ende eines jeden Abschnitts stehen die CHAIN-Anweisungen. Sie legen fest, welcher Block als nächster Programmteil ausgeführt werden soll. Die Programmteile müssen somit in sequentieller Reihenfolge auf dem Band abgespeichert sein. Denken Sie vor dem Programmieren des nächsten Programmblocks daran, den alten mit NEW zu löschen.

Immerhin lauten die CHAIN-Anweisungen:



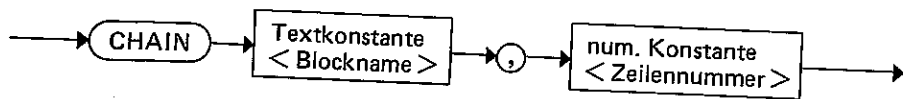
Es wird der nächste Programmblock eingeladen.



Es wird das Programm mit dem spezifizierten Blocknamen eingelesen.



Es wird der nächste Programmblock eingeladen, der Programmablauf startet mit der spezifizierten Zeilennummer.



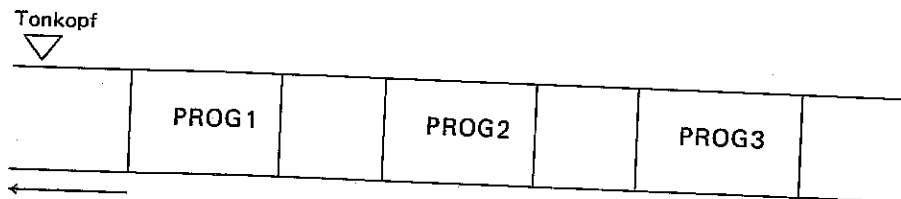
Es wird das Programm mit dem spezifizierten Blocknamen eingelesen, der Programmablauf startet mit der spezifizierten Zeilennummer.

Vor dem Laden des Programmblocks wird der Programmspeicher gelöscht, Variablen bleiben jedoch erhalten. Sprünge von Block zu Block sind somit nicht möglich.

### Beispiel:

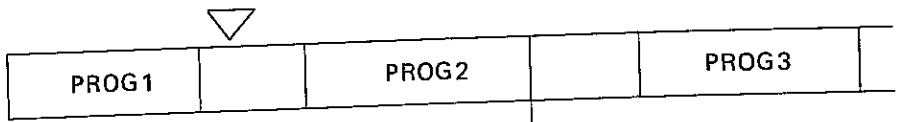
Die folgenden Diagramme veranschaulichen das Zusammenfügen der Programmblöcke:

Auf dem Band befinden sich die drei Programmteile mit den Blocknamen: PROG 1, PROG 2, PROG 3



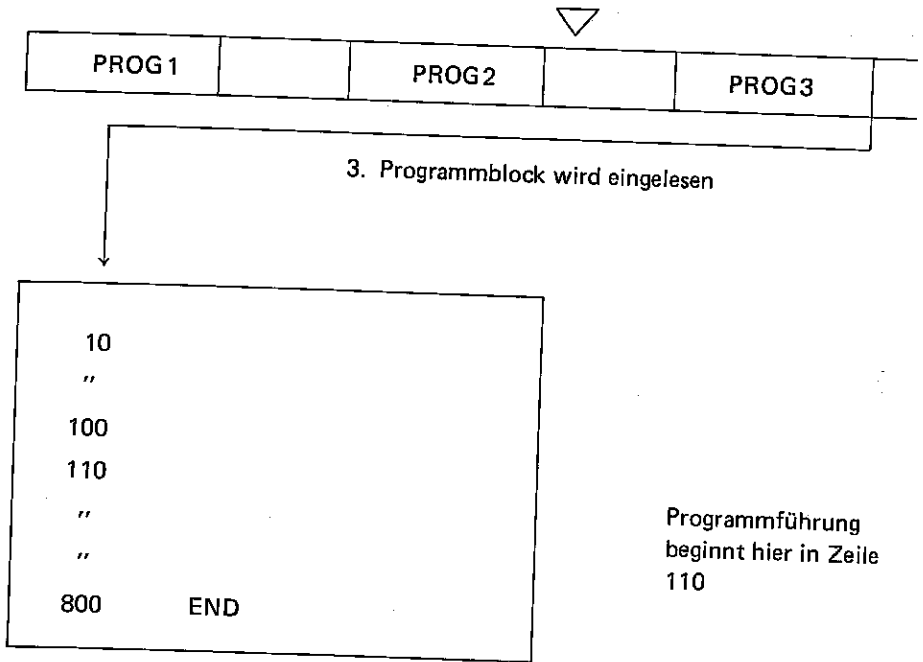
Mit CLOAD "PROG1" wird der erste Programmteil geladen.

```
10 INPUT A, B
"
"
"
"
300 CHAIN "PROG2"
```



2. Programmblock wird eingelesen

```
10 :
"
"
500 : CHAIN "PROG3", 110
```



Die Programmteile werden automatisch miteinander verkettet und ausgeführt.

▽ kennzeichnet jeweils die Position des Tonkopfes vor der Ausführung der CHAIN-Instruktion.

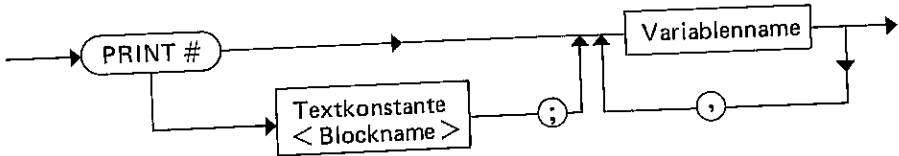
**Hinweise:**

1. Man sieht, daß die Programmblöcke in der richtigen Reihenfolge auf dem Band abgespeichert sein müssen: Sonst ist es erforderlich, daß Sie manuell eingreifen und dies auch bei der Programmierung berücksichtigen.
2. Außerdem ist es notwendig, den Tonkopf vor den ersten Programmteil zu bringen. Alles weitere steuert der Computer selbsttätig.
3. Die einzelnen Programmblöcke können mit einem PASS-Wort geschützt sein.

# PRINT #

Mit dieser Anweisung werden die Werte von Variablen auf dem Band abgespeichert.

## Syntax



Der **PC-1251** unterscheidet zwischen dem Programm und dem Datenspeicher. Mit der **PRINT #**-Anweisung werden die Werte einer Variablen oder einer Gruppe von Variablen auf dem Band abgespeichert. Die Textkonstante **< Blockname >** hat die gleiche Bedeutung wie bei dem **CSAVE**-Kommando und legt den Namen des Datensatzes auf dem Band fest.

Die Speicherung kann nur in der Betriebsart **RUN** sowohl manuell als auch programmgesteuert erfolgen.

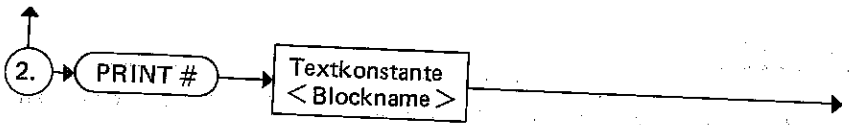
Der Rechner kann Datenblöcke von Programmblöcken unterscheiden. Es ist daher möglich, die zu einem Programm gehörenden Daten mit dem gleichen Namen zu versehen.

Geben Sie einen einzelnen Variablennamen oder eine Folge von Variablennamen an, werden die Inhalte dieser aufgeführten Variablen abgespeichert.

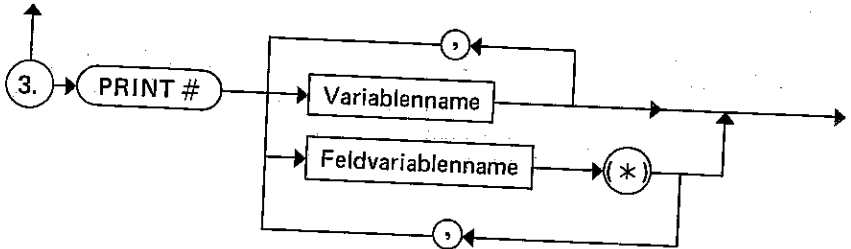
3 einzelnen Formate der **PRINT #**-Anweisung haben folgende Wirkung:



Es werden alle Werte der **A**-Variablen (Standardvariablen von **A** bis **Z** und alle Elemente des **A**-Vektors) auf dem Magnetband abgespeichert (siehe **VARIABLEN**).



Diese Anweisung hat die gleiche Wirkung wie unter (1), jedoch werden die Daten unter einem Blocknamen abgespeichert.



Es werden die Werte der definierten Variablen abgespeichert.

Wird eine A-Variable definiert, so werden beginnend mit dieser alle nachfolgenden A-Variablen, einschließlich des A-Vektors abgespeichert.

Lautet die Anweisung z.B. PRINT# C, so werden die Werte der Variablen C bis Z = A(3) bis A(26) und alle Elemente des A-Vektors abgespeichert.

Wird eine Feldvariable definiert, so werden alle Elemente dieser Variablen abgespeichert.

Aufgerufen wird eine Feldvariable mit z.B. B(\*).

Lautet die Anweisung z.B. PRINT# J, B\$(\*), C(\*) werden die Werte folgender Variablen auf der Kassette gespeichert:

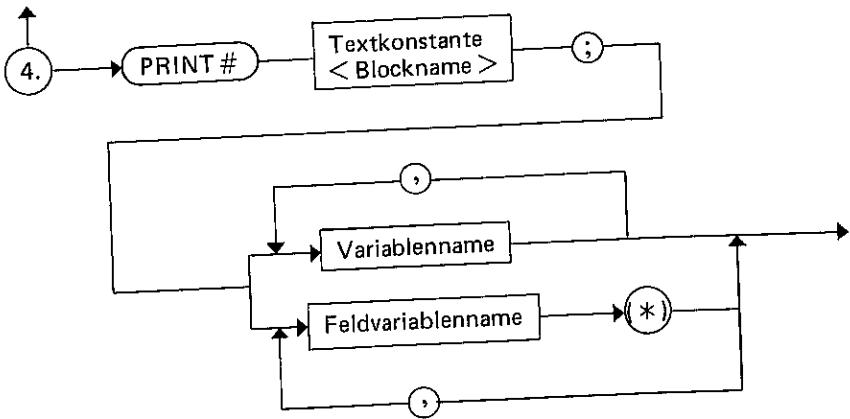
Standardvariable J bis Z = A(10) bis A(26).

Alle Elemente des A-Vektors.

Alle Elemente der Textfeldvariablen B\$.

Alle Elemente der numerischen Feldvariablen C.





Diese Anweisung hat die gleiche Wirkung wie unter (3), jedoch werden die Daten unter einem Blocknamen abgespeichert.

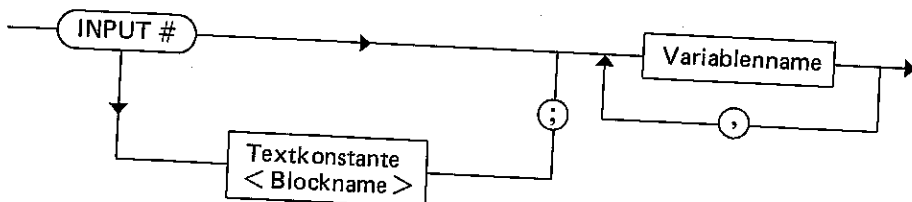
#### Hinweise:

1. Denken Sie bitte rechtzeitig daran, daß das Magnetband für die Aufnahme vorbereitet sein muß.
2. Einzelne Elemente eines Feldes können nicht direkt abgespeichert werden.
3. Vor Ausführung des Kommandos PRINT# muß eine Variable über A (27) oder eine dimensionale Variable im Programm-/Datenspeicherbereich gesichert werden. Wenn man die Variable nicht vor dem PRINT#-Kommando bestimmt, so führt dies zu einem Fehler (ERROR 3).
4. Wenn eine Variable über A (27) bestimmt und das PRINT#-Kommando ausgeführt wird, tritt ERROR 3 auf, da die bezeichnete Variable nicht im Programm-/Datenspeicherbereich sicher gespeichert ist. Wenn diese auftritt, kann die Variable über A (27) anschließend nicht mehr verwendet werden. Wenn keine Variable über A (27) verwendet werden soll, löscht man alle Variablen durch Eingabe von CLEAR und Drücken der **ENTER**-Taste.

# INPUT #

Mit dieser Anweisung werden Daten vom Band gelesen und Variablen zugewiesen.

## Syntax



Mit der INPUT #-Anweisung kann man die Daten, die mit der PRINT #-Anweisung auf das Band geschrieben wurden, wieder lesen und sie Variablen zuweisen.

Die Sprachelemente haben die gleiche Bedeutung wie in der PRINT #-Anweisung. Die Textkonstante bezeichnet den Namen des zu lesenden Datensatzes. Der einzelnen Variablen oder der Folge von Variablen und Feldern werden die Werte zugewiesen, die bei der PRINT #-Anweisung abgespeichert wurden. Felder sind so wie bei der PRINT #-Anweisung anzugeben.

Wenn sich die Anzahl der Variablen zwischen der INPUT # und entsprechenden PRINT #-Anweisungen unterscheiden, tritt folgende Aktion ein:

Die Anzahl der INPUT #-Variablen ist größer:

Die restlichen Variablen erhalten den Wert 0 bzw. leeren Text.

Die Anzahl der PRINT #-Variablen ist größer:

Die restlichen Variablen auf dem Band werden ignoriert.

## Beispiel:

10 INPUT # J, B\$(\*), C(\*)

10 INPUT # "DATA-1"; J, B\$(\*), C(\*)

### 15.3 Rückladen von Programmen und Daten von einem externen Bandgerät.

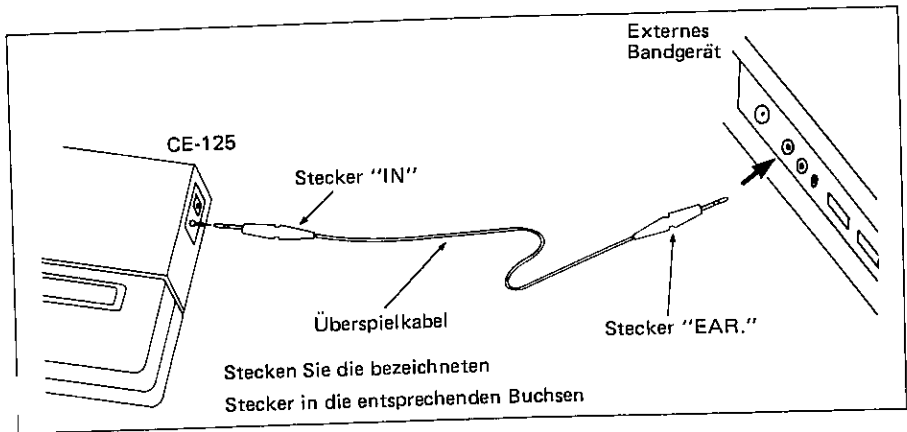
Das in den CE-125 integrierte Kassetteninterface gestattet es, Programme und Daten von einem zweiten Bandgerät einzulesen. Dabei ist gedacht an Programme und Daten, die mit dem SHARP PC-1210/11/12-Taschencomputer auf einer Kassette abgespeichert wurden.

Es können nur Programme und Daten eingelesen werden; deshalb sind für diese Betriebsart nur CLOAD, CLOAD?, MERGE, CHAIN und INPUT # anzuwenden.

Der zweite Rekorder wird nicht vom PC-1251/CE-125 gesteuert. Deshalb müssen vor dem Einlesen die Programm und Datenblöcke manuell vor den Tonkopf eingestellt werden. Wird das nicht beachtet, können Lesefehler eintreten.

Wird das Überspielkabel in die Buchse "IN" gesteckt, sind die obengenannten Kassettenanweisungen nicht mehr für den eingebauten Rekorder verfügbar.

### ANSCHLUSS EINES EXTERNEN BANDGERÄTES



# 16. Programmbeispiele

Wir haben an dieser Stelle einige Programmbeispiele aufgeführt, die die Programmiermöglichkeiten des PC-1251 veranschaulichen sollen.

Jedes Programm wird kurz in seiner Logik und Struktur erklärt.

## Darlehensrückzahlung

Über dieses Programm können die Höhe einer Darlehensrückzahlung und die Gesamtkosten eines Darlehens berechnet werden.

Es wird zunächst der Kreditbetrag, die Zinsen und die Laufzeit in Monaten abgefragt. Die Rückzahlung wird über die folgende Formel berechnet:

$$A = \frac{P * (1 + I)^N * I}{(1 + I)^N - 1}$$

A = Annuität

P = Kreditbetrag

I = Zinssatz/Monat als Dezimalbruch dargestellt (z.B. 1% = 0.01)

N = Laufzeit in Monaten

Das Programm berechnet das Endkapital über die gesamte Laufzeit und den Gesamtzinsbetrag.

Es werden die Zinsen/Jahr abgefragt, weil dies die übliche Berechnungsbasis darstellt.

```

10: INPUT "BETRAG?";P
20: INPUT "ZINSEN/JAHR";I
30: I=I/12

40: I=I/100

50: INPUT "MONATE?";N
60: T=(P * ((1+I) ^ N)) * I

70: B=((1+I) ^ N)-1

80: A=T/B
90: A = (INT ((A * 100) + .5))/100
100: PRINT "MTL.RATE=";A

110: Z=A*N

120: PRINT "GESAMTKOSTEN = ";Z
130: X=Z-P

140: PRINT "GESAMT ZINSEN=";X
150: INPUT " NEUE BERECHNUNG";Q$
160: IF (LEFT$(Q$, 1)="J") THEN GOTO 10

170: END

```

```

Eingabe des Kreditbetrages
Eingabe des Zinssatzes/Jahr
Division durch 12 zur Berechnung
der monatlichen Zinsen
Division durch 100, um Prozent in
Dezimalbruch umzuwandeln
Eingabe Laufzeit in Monaten
Berechnung des Zählers der
Formel
Berechnung des Nenners der
Formel
Bilden des Quotienten
Runden auf 2 Stellen
Anzeige der monatlichen Rück-
zahlungsrate
Produkt Anzahl der Monate mal
monatlicher Betrag zur
Ermittlung der Gesamtkosten
Anzeige Gesamtkosten
Differenz Gesamtkosten-Darlehen
zur Ermittlung der Zinsen
Anzeige des Zinsbetrages
Soll das Programm neu gestartet
werden?
Rücksprung zu Zeile 10, wenn in
Zeile 150 "JA" eingegeben
wird
Sonst Ende

```

### Hinweis:

Die Berechnung in Zeile 90 ist ein kleiner Programmiertrick, mit dem man Zahlen beliebig runden kann. Die Multiplikation mit 100 verschiebt die ersten beiden Dezimalstellen auf die linke Seite des Dezimalpunktes. Durch Anwendung der 'T-Funktion auf den so entstandenen Dezimalbruch wird der gebrochene Anteil abgeschnitten.

Es wird nun 0.5 addiert, so daß aufgerundet wird, wenn der abgeschnittene Teil größer als 0.5 ist. Der Betrag wird dann durch 100 geteilt, um die ursprüngliche Dezimalreihe mit 2 Stellen nach dem Dezimalpunkt wieder herzustellen.

## Simulation eines Glücksspielautomaten

Dieses Programm simuliert den Betrieb eines Glücksspielautomaten. Es wird dabei ausgegangen von 3 Rädern auf denen verschiedene Bilder sind. Jeweils ein Motiv von jedem Rad wird gleichzeitig angezeigt, wobei 3 Motive auf den Rädern wechseln.

Bei jedem Spiel werden die Räder eine zeitlang schnell gedreht. Sie werden dann langsamer und kommen nach und nach zum Halten. Der Haltepunkt jedes Rades wird durch einen Zufallszahlengenerator bestimmt.

Wird am Ende des Spiels dreimal dasselbe Motiv angezeigt, haben Sie gewonnen.

Sie können in dem Programm leicht die Anzahl der Motive, die Spielgeschwindigkeit und die Kombinationen, die einen Gewinn herbeiführen, verändern.

Zunächst wird die Kontrolle der Drehgeschwindigkeit über Variable gesteuert (d.h. die Zeit, die vergeht, bevor ein neues Motiv in der Anzeige erscheint).

Eine Variable kontrolliert die Umdrehung während der ersten Phase, eine andere die während der zweiten Auslaufphase.

Eine dritte Variable kontrolliert die Länge anderer Anzeigen.

Die Zahl der möglichen Motive wird ebenfalls mit Hilfe einer Variablen festgelegt. Obwohl in diesem Beispiel nur drei Motive möglich sind, können Sie dies leicht ändern, indem Sie die DATA-Liste erweitern.

Der Gewinnbetrag ist so kalkuliert, daß die erwartete Auszahlung etwas unter den erwarteten Kosten liegt. Man kann auch diese Routine durch komplexere Gewinnregeln mit unterschiedlichen Auszahlungen umschreiben.

Variablen kontrollieren ebenfalls die Länge der ersten und zweiten Drehphase.

10: S = 0: F = 10: G = 90: H = 200

20: T = 3: Z = INT(((T^3)-1)/T): C=3 \* T: B=6 \* T

30: DIM R(3), D(3), P\$(T) \* 8

40: RANDOM

50: FOR I = 1 TO 3  
60: D(I) = RND(T)  
70: NEXT I

80: FOR I = 1 TO T  
90: READ P\$(I)  
100: NEXT I

110: DATA "ORANGE", "APFEL", "BIRNE"

120: WAIT H

130: PRINT "\$ SLOT MACHINE"  
140: WAIT  
150: PRINT "ENTER DRUECKEN"

200: REM \*\*\* START SPIEL \*\*\*  
`0: M = 0  
\_20: FOR I = 1 TO 3  
230: R(I) = B + RND(C)  
240: IF (R(I) > M) THEN LET M = R(I)  
250: NEXT I

260: WAIT F

270: E = (G - F) / (M - C)

Variablen-Initialisierung S = Gewinn/Verlustspeicher, F = Wert von PRINT WAIT für die erste Drehphase, G = längstes PRINT WAIT beim Drehen, H = PRINT WAIT für Informationszeigen  
T = Anzahl der Motive, Z = Gewinnquote, C = Länge des schnellen Drehens, B = maximale Länge der langsamen Drehphase

Feldvereinbarung: R ( ) = wie oft jedes Rad sich dreht, D ( ) = Anzahl der Motive, P\$ ( ) = Namen der Objekte  
Generieren einer jeweils neuen Zufallszahlenfolge durch RND ( )

Initialisieren jedes Motivs in jedem "Fenster" durch eine Zufallszahl

Einlesen der Motivnamen  
Leerräume werden eingefügt.

WAIT-Intervall für die Start-Anzeige  
Start-Anzeige  
Aufheben des WAIT-Intervalls  
Eingabe von ENTER, um das Spiel zu starten

Initialisieren von M

Bestimmen der Haltezeit  
M = längste Haltezeit

Setzen des WAIT-Intervalls für die erste Drehphase  
Berechnen des Werts, um das Intervall bei jeder Drehung während der Auslaufphase zu verkürzen

```

280: FOR A=1 TO M
290: IF (A>C) THEN WAIT F+((A-C) * E)
300: FOR I=1 TO 3
310: IF (R(I)>0) THEN LET D(I)=D(I)+1
320: IF (D(I)>T) THEN LET D(I)=1
330: R(I)=R(I)-1
340: NEXT I
360: NEXT A
370: WAIT H

380: PRINT P$(D(1)); P$(D(2)); P$(D(3))
390: W=-1
400: IF (D(1)=D(2)) AND (D(2)=D(3)) THEN LET W=Z
410: IF W<0 THEN PRINT "VERLOREN"
420: IF W>0 THEN PRINT "SIE GEWINNEN"; Z;
 "DOLLARS"
430: S=S+W

440: ON (2+SGN(S)) GOTO 450, 470, 490

450: PRINT "SIE HABEN"; ABS(S); "$ VERLOREN"
460: GOTO 500
470: PRINT "AUSGLEICH"
480: GOTO 500
490: PRINT "SIE HABEN"; S; "$ GEWINNEN"
500: INPUT "NEUES SPIEL"; Q$

510: IF (LEFT$(Q$, 1)="J") THEN GOTO 200
520: END

```

Schleife von 1 zur längsten  
 Haltezeit M  
 Falls 2 Phase, Drehzeit um jeweils  
 1 verringern  
 Schleife Fenster 1 bis 3  
 Hält die Drehung an, weitergehen  
 zum nächsten Motiv  
 Zurück zumersten Motiv wenn  
 $D(I) > 3$   
 Zähler der verbleibenden  
 Drehungen verringern

Setzen des längsten WAIT-  
 Intervalls  
 Erneute Anzeige des Spielendes  
 Verlust von 1\$ wird zugewiesen  
 Gewinnquote wird zugewiesen  
 Falls verloren, Anzeige

Falls gewonnen, Anzeige  
 Addieren der Gewinne bzw.  
 Verluste

Sprung zur Anzeige entsprechend  
 dem Spielergebnis  
 Anzeige bei  $S < 0$   
 GOTO-Abfrage, ob Spielende  
 Anzeige bei  $S = 0$   
 GOTO-Abfrage, ob Spielende  
 Anzeige bei  $S > 0$   
 Abfrage, ob neues Spiel  
 gewünscht  
 Prüfen des ersten Buchstaben der  
 Antwort



## Minenfeldspiel

Das Spiel wird auf einem Brett mit 10 x 10 Feldern gespielt. Oben ist Norden, unten Süden, links Westen, rechts Osten. Die Spalten werden horizontal von links nach rechts von 1 – 10 durchnummeriert. Desgleichen werden die Reihen vertikal von unten nach oben von 1 – 10 durchnummeriert. Man beginnt in der Südwest-Ecke, Quadrat (1, 1). Ziel ist es die Nordost-Ecke, Quadrat (10, 10) zu erreichen.

Durch Eingabe von 1 – 9 kann man sich in gewünschter Richtung bewegen. Die Richtung wird durch folgenden Schlüssel bestimmt:

|          |      |         |
|----------|------|---------|
| Nordwest | Nord | Nordost |
| 7        | 8    | 9       |
| West     |      | Ost     |
| 4        | 5    | 6       |
| Südwest  | Süd  | Südost  |
| 1        | 2    | 3       |

In durch einen Zufallszahlengenerator bestimmten Quadraten liegen Minen. Betreten Sie ein solches Feld, haben Sie verloren. Um das zu verhindern, gibt der **PC-1251** ein akustisches Signal ab, wenn Sie ein Quadrat betreten; und zwar für jedes Quadrat in der Nähe, einer Mine. Allerdings wird die Richtung nicht angegeben. Das Programm sorgt dafür, daß in den Ecken keine Minen so liegen, ok, daß es unmöglich ist, zu starten oder anzukommen.

Die Anzahl der Minen bestimmen Sie selbst durch Eingabe auf die Frage "SCHWIERIGKEITSGRAD?".

Achten Sie in diesem Programm den Gebrauch von Unterprogrammen; dadurch wird das Hauptprogramm kurz und übersichtlich, während alle speziellen Berechnungen in verschiedenen Unterprogrammen stattfinden.

10: S = 10

20: DIM F(S+1, S+1)

30: WAIT 128

40: INPUT "SCHWIERIGKEITSGRAD"; B

50: PRINT "EINE MINUTE WARTEN"

60: FOR I=1 TO S

70: FOR J=1 TO S

80: F(S, S) = 0

90: NEXT J

100: NEXT I

110: FOR I=1 TO B

120: X = RND(S)

130: Y = RND(S)

140: IF ((X+Y)>5) THEN GOTO 120

150: IF (((1+S-X)+(1+S-Y))<5) THEN GOTO 120

160: F(X, Y) = 1

170: NEXT I

180: F(S, S) = 9

190: X = 1: Y = 1

200: PRINT "SIE SIND BEI ("; X; ", "; Y; ")"

210: GOSUB 400

220: BEEP C

230: INPUT "WOHIN JETZT"; A\$

240: D = VAL(A\$)

250: IF ((D<1) OR (D>9)) THEN PRINT  
"NUR 1 BIS 9 BITTE" : GOTO 230

260: IF (D>6) THEN GOSUB 500

270: IF (D<4) THEN GOSUB 550

Festlegen der Feldgröße 10 x 10  
Feldvereinbarung, Index um je 1  
größer in beiden Richtungen;  
das erleichtert den Sprung zu  
Zeile 220

WAIT-Intervall 2 sec.

Eingabe Anzahl der Minen

Anzeige der Wartezeit vor

Spielbeginn

Schleife, in der alle Feldelemente  
auf Null gesetzt werden; not-  
wendig um etwaige Minen aus  
vorherigen Spielen zu löschen

Schleife und Zähler: Anzahl der  
Minen

X Koordinate über Zufallszahlen-  
generator

Y Koordinate über Zufallszahlen-  
generator

Prüfen, ob Mine zu nahe beim  
Startpunkt liegt

Prüfen, ob Mine zu nahe beim  
Zielpunkt liegt

Legen der Mine

Ziele markieren

Start setzen bei (1, 1)

Spielstart; Positionsanzeige

Unterprogramm zum Zählen der  
Minen in der Nähe

Signalton zum Anzeigen der  
Minen, die in der Nähe liegen

Richtungs-Eingabe

Umwandlung in numerischen  
Ausdruck

Bei unzulässiger Eingabe neue  
Abfrage

Wenn Eingabe 7, 8 oder 9 GOTO  
Unterprogramm Nord

Bei Eingabe von 1, 2 oder 3  
GOSUB Süd

280: IF ((D=3) OR (D=6) OR (D=9)) THEN GOSUB 600  
290: IF ((D=1) OR (D=6) OR (D=7)) THEN GOSUB 650

300: IF (F(X, Y)=1) THEN GOTO 700

310: IF (F(X, Y)=9) THEN GOTO 750

320: GOTO 200

400: C = 0

0: FOR I = X-1 TO X+1

420: FOR J = Y-1 TO Y+1

430: IF (F(I, J) = 1) THEN LET C = X+1

440: NEXT J

450: NEXT I

460: RETURN

500: IF (Y=S) THEN PRINT

"SIE SIND IN ECKE NORD": GOTO 520

510: Y = Y+1

520: RETURN

550: IF (Y=1) THEN PRINT

"SIE SIND IN ECKE SUED": GOTO 570

560: Y = Y-1

570: RETURN

600: IF (X=S) THEN PRINT

"SIE SIND IN ECKE SUED": GOTO 620

610: X = X+1

2: RETURN

650: IF (X=1) THEN PRINT

"SIE SIND IN ECKE WEST": GOTO 670

660: X = X-1

670: RETURN

700: PRINT "BOOM!!!!!!!! VERLOREN"

710: GOTO 800

750: PRINT "PRIMA, GEWONNEN"

800: INPUT "NEUES SPIEL": A\$

810: IF (LEFT\$(A\$, 1) = "J") THEN GOTO 40

820: END

Für D=3, 6 oder 9 GOSUB Ost  
Für D=1, 4 oder 7 GOSUB West

Wenn das erreichte Feld eine  
Mine enthält GOTO Anzeige  
VERLOREN

Wenn das erreichte Feld das  
Ziel ist GOTO Anzeige  
GEWONNEN

Sonst Sprung

Unterprogramm zum Zählen der  
Minen

Zähler setzen für die Minen  
Schleife durch benachbarte  
Felder

Wenn eine Mine da ist, Zähler um  
1 heraufsetzen

Unterprogramm Nord

Wenn an Nord-Ecke, nichts tun  
1 Feld nördlich gehen  
RETURN

Wenn an Süd-Ecke, nichts tun  
1 Feld südlich gehen  
RETURN

Unterprogramm Ost

Wenn an Ost-Ecke, nichts tun  
1 Feld östlich gehen  
RETURN

Unterprogramm West

Wenn an West-Ecke, nichts tun  
1 Feld nördlich gehen  
RETURN

Anzeige "VERLOREN"  
GOTO END

Anzeige "GEWONNEN"

Abfrage, ob neues Spiel  
gewünscht

Zurück zum Anfang, wenn  
erster Eingabebuchstabe J ist  
ENDE

## Sortierprogramm

In diesem Programm wird eine der einfachen Sortiertechniken angewandt, nämlich das "Sortieren durch Suche". Im Programmablauf wird ein Feld unsortierter Daten auf das jeweils größte Element hin durchsucht. Dieses wird dann an erste Stelle gesetzt und aus den verbleibenden Elementen wird das nächstgrößte ausgewählt etc. Jedes ausgewählte Element wird mit einer sehr kleinen Zahl überschrieben, so daß es bei der nächsten Suche unberücksichtigt bleibt.

```
10: INPUT "WIEVIELE ARTIKEL?";N
20: DIM O(N), S(N)

30: FOR I=1 TO N
40: INPUT O (I)
50: NEXT I
60: PAUSE "SORTIEREN"
70: FOR I=1 TO N

80: T=1

90: FOR J=1 TO N
100: IF (O(J)>O(T)) THEN LET T=J

110: NEXT J
120: S (I) = O (T)

130: O (T) =-9.999999999E99

140: NEXT I
150: BEEP 2
```

Eingabe der Anzahl der zu sortierenden Werte  
Feldvereinbarung zur Aufnahme der Daten O ( ) für die unsortierten Daten, S ( ) für sortierte Daten  
Schleife zum Einlesen der Daten  
Eingabe der Daten

Anzeige Beginn des Sortierens  
Die äußere Schleife setzt die Indizes von S ( ), in die die jeweils größten Werte gelesen werden  
Willkürlich gesetzter Zähler für den größten Wert im ersten Element von O ( )  
Schleife zum Prüfen der unsortierten Daten  
Wird ein größerer Wert gefunden, wird der Wert T so geändert, daß er auf alle bis dahin größten Werte hinweist

Dieser Wert wird dem nächsten Element des Feldes mit den sortierten Daten zugewiesen  
In das entsprechende Element aus dem Feld unsortierter Daten wird die kleinstmögliche Zahl eingelesen, so daß der Wert nicht noch einmal als größter Wert bestimmt werden kann  
Ende der Schleife

Signal, daß das Sortieren beendet ist

```
160: PAUSE "SORTIEREN BEENDET"
170: INPUT "ANZEIGE ODER DRUCK?"; A$
180: IF (LEFT$(A$,1)="D") THEN PRINT=LPRINT

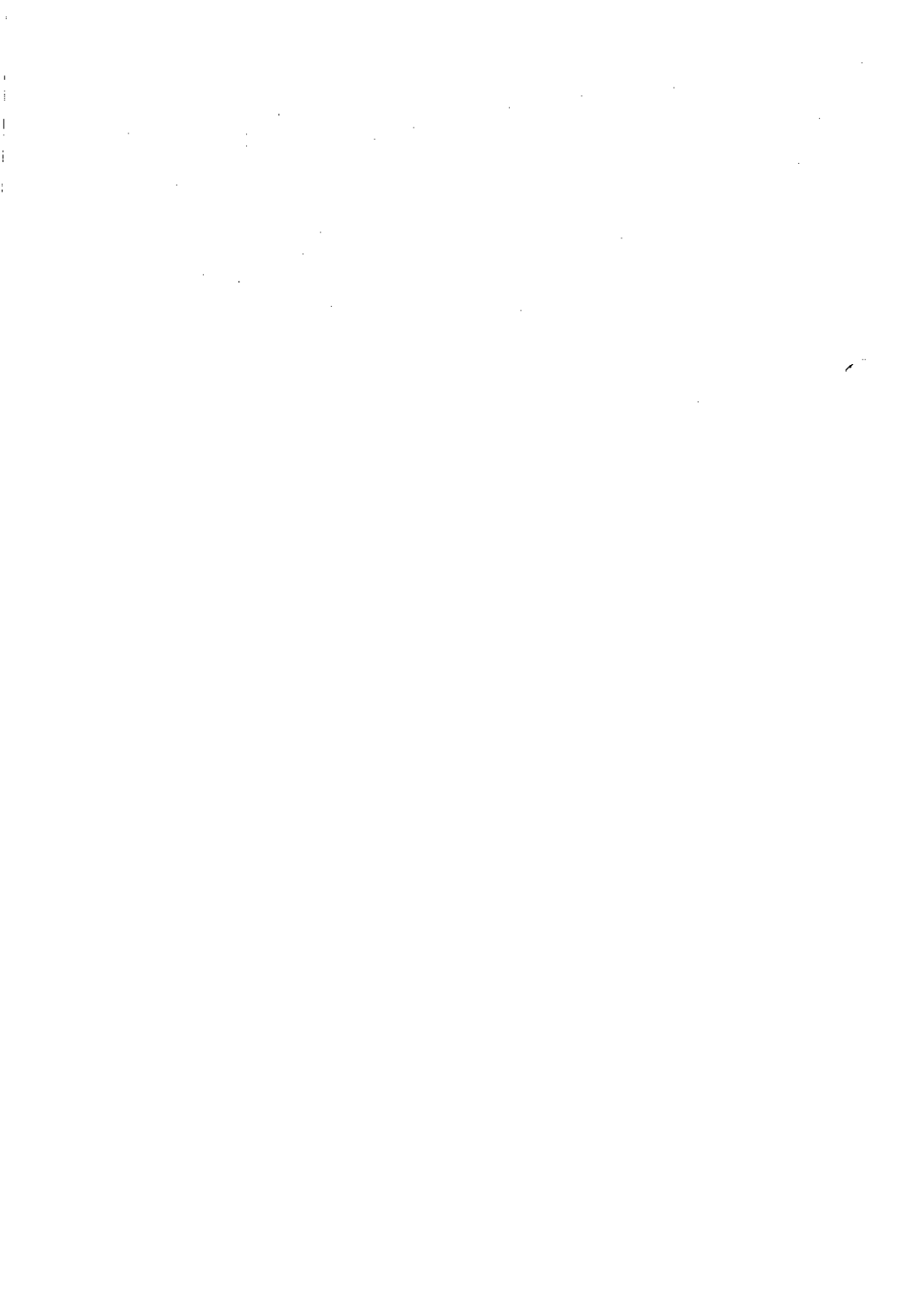
190: FOR I = 1 TO N
200: PRINT S(I)

210: NEXT I
220: PRINT = PRINT
230: END
```

Abfrage, ob Ergebnis angezeigt  
oder ausgedruckt werden soll  
Wird der Drucker gewählt, wird  
die Ausgabe über den Drucker  
angewiesen

Schleife durch das Feld der  
sortierten Daten  
Ausgabe der Elemente, der Reihe  
nach vom größten zum  
kleinsten

Wiederherstellen der Ausgangs-  
ausgabe



# ANHANG

# A. ASCII-Code Tabelle

| ASCII Zeichen | Entsprechender Code |         |             |
|---------------|---------------------|---------|-------------|
|               | Dual                | Dezimal | Hexadezimal |
| Leerraum      | 00100000            | 32      | 20          |
| !             | 00100001            | 33      | 21          |
| "             | 00100010            | 34      | 22          |
| #             | 00100011            | 35      | 23          |
| \$            | 00100100            | 36      | 24          |
| %             | 00100101            | 37      | 25          |
| &             | 00100110            | 38      | 26          |
| □             | 00100111            | 39      | 27          |
| (             | 00101000            | 40      | 28          |
| )             | 00101001            | 41      | 29          |
| *             | 00101010            | 42      | 2A          |
| +             | 00101011            | 43      | 2B          |
| ,             | 00101100            | 44      | 2C          |
| -             | 00101101            | 45      | 2D          |
| .             | 00101110            | 46      | 2E          |
| /             | 00101111            | 47      | 2F          |
| 0             | 00110000            | 48      | 30          |
| 1             | 00110001            | 49      | 31          |
| 2             | 00110010            | 50      | 32          |
| 3             | 00110011            | 51      | 33          |
| 4             | 00110100            | 52      | 34          |
| 5             | 00110101            | 53      | 35          |
| 6             | 00110110            | 54      | 36          |
| 7             | 00110111            | 55      | 37          |
| 8             | 00111000            | 56      | 38          |
| 9             | 00111001            | 57      | 39          |
| :             | 00111010            | 58      | 3A          |
| ;             | 00111011            | 59      | 3B          |
| <             | 00111100            | 60      | 3C          |
| =             | 00111101            | 61      | 3D          |
| >             | 00111110            | 62      | 3E          |
| ?             | 00111111            | 63      | 3F          |



| ASCII Zeichen | Entsprechender Code |         |             |
|---------------|---------------------|---------|-------------|
|               | Dual                | Dezimal | Hexadezimal |
| @             | 01000000            | 64      | 40          |
| A             | 01000001            | 65      | 41          |
| B             | 01000010            | 66      | 42          |
| C             | 01000011            | 67      | 43          |
| D             | 01000100            | 68      | 44          |
| E             | 01000101            | 69      | 45          |
| F             | 01000110            | 70      | 46          |
| G             | 01000111            | 71      | 47          |
| H             | 01001000            | 72      | 48          |
| I             | 01001001            | 73      | 49          |
| J             | 01001010            | 74      | 4A          |
| K             | 01001011            | 75      | 4B          |
| L             | 01001100            | 76      | 4C          |
| M             | 01001101            | 77      | 4D          |
| N             | 01001110            | 78      | 4E          |
| O             | 01001111            | 79      | 4F          |
| P             | 01010000            | 80      | 50          |
| Q             | 01010001            | 81      | 51          |
| R             | 01010010            | 82      | 52          |
| S             | 01010011            | 83      | 53          |
| T             | 01010100            | 84      | 54          |
| U             | 01010101            | 85      | 55          |
| V             | 01010110            | 86      | 56          |
| W             | 01010111            | 87      | 57          |
| X             | 01011000            | 88      | 58          |
| Y             | 01011001            | 89      | 59          |
| Z             | 01011010            | 90      | 5A          |
| √             | 01011011            | 91      | 5B          |
| ¥             | 01011100            | 92      | 5C          |
| π             | 01011101            | 93      | 5D          |
| ^             | 01011110            | 94      | 5E          |
| _             | 01011111            | 95      | 5F          |
| E             | 01100000            | 96      | 60          |

## B. Rechenbereich

| Funktionen                                     | Rechenbereich                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| $y^x$ ( $y^x$ )                                | $-1 \times 10^{100} < x \log y < 100$<br>$y=0, x \leq 0$ : ERROR 2 Beisp.) $0^0$ [ENTER] $\rightarrow$ ERROR 2<br>$y=0, x > 0$ : $0$ $0^0$ [ENTER] $\rightarrow$ $0$<br>$(-4)^{05}$ [ENTER] $\rightarrow$ ERROR 2<br>Um das Auftreten falscher Vorzeichen zu vermeiden, können negative Zahlen nicht mit dem $\wedge$ -Operator in die Potenz erhoben werden.<br>Beim Potenzieren $Y^X$ wird im Falle eines negativen Y-Wertes das Vorzeichen nicht korrekt angezeigt. |
| SIN $x$<br>COS $x$<br>TAN $x$                  | Die folgenden Werte sind für TAN $x$ nicht zulässig<br>DEG: $ x  < 1 \times 10^{10}$<br>RAD: $ x  < \frac{\pi}{180} \times 10^{10}$<br>GRAD: $ x  < \frac{10}{9} \times 10^{10}$<br>DEG: $ x  = 90(2n-1)$<br>RAD: $ x  = \frac{\pi}{2}(2n-1)$<br>GRAD: $ x  = 100(2n-1)$<br>(n: Ganzzahl)                                                                                                                                                                              |
| SIN <sup>-1</sup> $x$<br>COS <sup>-1</sup> $x$ | $-1 \leq x \leq 1$                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| TAN <sup>-1</sup> $x$                          | $ x  < 1 \times 10^{100}$                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| LN $x$<br>LOG $x$                              | $1 \times 10^{-99} \leq x < 1 \times 10^{100}$                                                                                                                                                                                                                                                                                                                                                                                                                         |
| EXP $x$                                        | $-1 \times 10^{100} < x \leq 230.2585092$                                                                                                                                                                                                                                                                                                                                                                                                                              |
| $\sqrt{x}$                                     | $0 \leq x < 1 \times 10^{100}$                                                                                                                                                                                                                                                                                                                                                                                                                                         |

Andere Funktionen können nur berechnet werden, wenn der Wert für  $x$  im angegebenen Bereich bleibt:

$$1 \times 10^{-99} \leq |x| < 1 \times 10^{100} \text{ and } 0$$

Innerhalb des obengenannten Rechenbereiches ist der Fehler an der letzten angezeigten Stelle in der Regel kleiner als  $\pm 1$ .

Bei der wissenschaftlichen Darstellung gilt dies für die letzte Stelle der Mantisse.

# C. Fehlermeldungen

## 1. Syntax Fehler.

z.B. 10: PRINTT "SHARP"

Die Anweisung PRINT wurde falsch eingegeben.

## 2. Unzulässige Berechnungen.

z.B. der Rechenbereich des **PC-1251** von  $9.9E99 * 10$  ist überschritten, oder es wurde versucht, eine Division durch 0 durchzuführen.

## 3. Fehler in der Speicherbelegung.

Es wurde z.B. versucht, eine Operation auszuführen, bei der ein Konflikt entstanden ist zwischen der Wirkung der Operation und der Speicherorganisation des **PC-1251**.

z.B. 10 FOR I = 1 TO 35000

Der Wert der Laufvariablen ist höher als zugelassen.

## 4. Fehler in der Zeilennummer.

Sie haben eine unzulässige Zeilennummer verwendet.

## 5. Schachtelungsfehler.

Zu einer NEXT- bzw. RETURN-Anweisung existiert keine entsprechende FOR- bzw. GOSUB-Anweisung.

z.B. 10: FOR A = 1 TO 10

100 NEXT B

## 6. Speicherüberlauf.

Die Speicherkapazität des **PC-1251** wurde überschritten z.B: DIM B (100, 100).

## 7. Formatfehler.

z.B. 10: USING "####"

20: A=123\*20

30: PRINT A

## 8. Fehler bezüglich einer Option.

Die Informationsübertragung zwischen **PC-1251** und einer Option Drucker oder Rekorder **CE-125** funktioniert nicht. Batteriespannung im **CE-125** überprüfen. Verbindung **PC-1251/CE-125** überprüfen.

## 9. Sonstige.

Ein anderer Fehler als bisher spezifiziert ist aufgetreten.

z.B. CHR \$(1)

1 ist hier ein unzulässiger Wert

oder

10: A = 5 : PRINT A\$

Es kann nur entweder A oder A\$ als Variable verwendet werden.

# D. Liste der Funktionen und Anweisungen

## Funktionen

| Funktion | Abkürzung               | Bemerkungen                                                                                       | Seite |
|----------|-------------------------|---------------------------------------------------------------------------------------------------|-------|
| ABS      | AB.                     | Absolutbetrag                                                                                     | 50    |
| ACS      | AC.                     | Arcuscosinus ( $\cos^{-1}$ )                                                                      | 48    |
| AND      | AN.                     | Logischer Operator "UND"                                                                          | 56    |
| ASC      |                         | Umwandlung von Zeichen in ASCII-Code<br>(Umkehrfunktion von CHR\$)                                | 121   |
| ASN      | AS.                     | Arcussinus ( $\sin^{-1}$ )                                                                        | 48    |
| ATN      | AT.                     | Arcustangens ( $\tan^{-1}$ )                                                                      | 48    |
| CHR\$    | CHR.                    | Wandelt ASCII-Code in Zeichen um.<br>(Umkehrfunktion von ASC)                                     | 122   |
| COS      |                         | Cosinus X                                                                                         | 48    |
| DEG      |                         | Umrechnungsfunktion<br>Dezimalsystem $\Rightarrow$ Sexagesimalsystem                              | 46    |
| DMS      | DM.                     | Umrechnungsfunktion<br>Sexagesimalsystem<br>$\Rightarrow$ Dezimalsystem                           | 46    |
| EXP      | EX.                     | $e^x$ ; Exponentiation zur Basis e                                                                | 49    |
| INKEY\$  | INK.<br>INKE.<br>INKEY. | Zeichenfunktion; das Zeichen einer ge-<br>drückten Taste ergibt den Funktionswert.                | 131   |
| INT      |                         | Integerfunktion; zur Ermittlung des ganz-<br>zähligen Anteils eines numerischen Aus-<br>drucks.   | 50    |
| LEFT\$   | LEF.<br>LEFT.           | Textfunktion; entnimmt aus einer Zeichen-<br>folge eine spezifizierte Zeichenanzahl von<br>links. | 128   |
| LEN      |                         | Ermittlung der Zeichenanzahl eines Text-<br>ausdruckes.                                           | 124   |

| Funktion           | Abkürzung                      | Bemerkungen                                                                                    | Seite |
|--------------------|--------------------------------|------------------------------------------------------------------------------------------------|-------|
| LOG                | LO.                            | $\log_{10}x$ ; allgemeiner Logarithmus zur Basis 10.                                           | 49    |
| LN                 |                                | $\log_e x$ ; natürlicher Logarithmus zur Basis e.                                              | 49    |
| MEM                | M.<br>ME.                      | Ermittlung der Anzahl der freien Bytes im Hauptspeicher                                        | 54    |
| MID\$              | MI.<br>MID.                    | Textfunktion; entnimmt eine spezifizierte Anzahl von Zeichen aus der Mitte einer Zeichenkette. | 130   |
| NOT                | NO.                            | logische Negation                                                                              | 55    |
| OR                 |                                | logischer Operator "ODER"                                                                      | 57    |
| $\pi/PI$           |                                | Abruf der Kreiszahl<br>(= 3.141592654)                                                         | 52    |
| RIGHT\$            | RI.<br>RIG.<br>RIGH.<br>RIGHT. | Textfunktion; entnimmt aus einer Zeichenfolge die Zeichen vom rechten Ende.                    | 129   |
| RND                | RN.                            | Erzeugung einer Zufallszahl                                                                    | 52    |
| SGN                | SG.                            | Ermittlung des Vorzeichens eines numerischen Ausdrucks.                                        | 51    |
| SIN                | SI.                            | Sinus; $\sin x$                                                                                | 48    |
| $\sqrt{\quad}/SQR$ | SQ.                            | Quadratwurzel                                                                                  | 51    |
| STR\$              | STR.                           | Textfunktion; wandelt den Wert eines numerischen Ausdrucks in die dezimale Zeichenfolge um.    | 125   |
| TAN                | TA.                            | Tangens; $\tan x$                                                                              | 48    |
| VAL                | V.<br>VA.                      | Textfunktion; wandelt eine als Zeichenfolge eingegebene Zahl in ihren numerischen Wert um.     | 126   |
| $\wedge$           |                                | allgemeine Exponentialfunktion.                                                                | 22    |

## B. Anweisungen

| Anweisung                 | Abkürzung                                      | Bemerkungen                                                                                           | Seite |
|---------------------------|------------------------------------------------|-------------------------------------------------------------------------------------------------------|-------|
| AREAD                     | A.<br>AR.<br>ARE.<br>AREA.                     | Automatische Wertzuweisung zu einer spezifizierten Variablen nach Programmstart über Definable Keys.  | 78    |
| BEEP                      | B.<br>BE.<br>BEE.                              | Akustisches Signal in der Länge veränderbar.                                                          | 117   |
| CLEAR                     | CL.<br>CLE.<br>CLEA.                           | Löschen aller Variablen und Felder aus dem Hauptspeicher. Die Standardvariablen werden auf 0 gesetzt. | 65    |
| CONT                      | C.<br>CO.<br>CON.                              | Fortsetzung der Programmausführung nach STOP oder BREAK (nur im RUN-Mode).                            | 82    |
| DATA                      | DA.<br>DAT.                                    | Definition von Datenfeldern in Verbindung mit READ.                                                   | 72    |
| DEGREE                    | DE.<br>DEG.<br>DEGR.<br>DEGRE.                 | Setzen der Winkleinheit GRAD.                                                                         | 45    |
| DIM                       | D.<br>DI.                                      | Deklaration von Feldvariablen (Arrays)                                                                | 66    |
| END                       | E.<br>EN.                                      | Abschlußkommando eines Programms                                                                      | 82    |
| FOR<br>TO<br>STEP<br>NEXT | F.<br>FO.<br><br>STE.<br><br>N.<br>NE.<br>NEX. | Einrichtung von Programmschleifen                                                                     | 97    |
| GOSUB                     | GOS.<br>GOSU.                                  | Aufruf eines Unterprogramms. Rücksprung mit RETURN.                                                   | 90    |

| Anweisung   | Abkürzung                  | Bemerkungen                                                                                                              | Seite |
|-------------|----------------------------|--------------------------------------------------------------------------------------------------------------------------|-------|
| GOTO        | G.<br>GO.<br>GOT.          | Verzweigt das Programm zu einer spezifizierten Zeile.<br>Im RUN-Mode: Start des Programms an einer spezifizierten Zeile. | 86    |
| GRAD        | GR.<br>GRA.                | Setzen der Winkeleinheit<br>NEUGRAD                                                                                      | 45    |
| IF          |                            | Prüft nachfolgende Bedingung auf "wahr" oder "unwahr"<br>(wird in Verbindung mit THEN benutzt)                           | 94    |
| INPUT       | I.<br>IN.<br>INP.<br>INPU. | Weist Variablen Werte zu, die über die Tastatur eingegeben werden.                                                       | 101   |
| LET         | LE.                        | Wertzuweisung für Variablen                                                                                              | 70    |
| LIST        | L.<br>LI.<br>LIS.          | Auflisten von Programmzeilen in der Anzeige (nur im PRO-Mode)                                                            | 84    |
| NEXT        | N.<br>NE.<br>NEX.          | S. FOR...TO...STEP<br>NEXT                                                                                               | 97    |
| NEW         |                            | Im PRO-Mode: löscht den Inhalt des Hauptspeichers.<br>Im RESERVE-Mode: löscht den Inhalt des RESERVE-Speichers.          | 65    |
| ON<br>GOSUB | O.<br>GOS.<br>GOSU.        | Ruft in Abhängigkeit vom ON-Argument ein spezifiziertes Unterprogramm auf.                                               | 92    |
| ON<br>GOTO  | O.<br>G.<br>GO.<br>GOT.    | Verzweigt den Programmablauf in Abhängigkeit vom ON-Argument zu einer spezifizierten Programmzeile.                      | 88    |
| PASS        | PASS                       | Diese Anweisung schützt ein Programm vor dem Zugriff unbefugter Personen.                                                | 118   |
| PAUSE       | PAU.<br>PAUS.              | Ausgabanweisung; nach Ablauf von ca. 0,85 Sek. wird das Programm fortgesetzt.                                            | 110   |

| Anweisung | Abkürzung                          | Bemerkungen                                                                             | Seite |
|-----------|------------------------------------|-----------------------------------------------------------------------------------------|-------|
| PRINT     | P.<br>PR.<br>PRI.<br>PRIN.         | Ausgabeanweisung.                                                                       | 104   |
| RADIAN    | RAD.<br>RADI.<br>RADIA.            | Setzen der Winkleinheit RADIAN                                                          | 45    |
| RANDOM    | RA.<br>RAN.<br>RAND.<br>RANDO.     | Initiiert eine neue Folge von Zufallszahlen (in Verbindung mit RND)                     | 54    |
| READ      | REA.                               | Weist Variablen Werte aus einem Datenfeld zu (in Verbindung mit DATA).                  | 73    |
| REM       |                                    | Definiert den Rest der Zeile als Kommentar.                                             | 75    |
| RESTORE   | RES.<br>REST.<br>RESTO.<br>RESTOR. | Definieren der nächsten zu bearbeitenden DATA Anweisung.                                | 74    |
| RETURN    | RE.<br>RET.<br>RETU.<br>RETUR.     | Rücksprung in die auf den Unterprogrammaufruf folgende Zeile (in Verbindung mit GOSUB). | 93    |
| RUN       | R.<br>RU.                          | Start eines Programms (nur im RUN-Mode).                                                | 76    |
| STOP      | S.<br>ST.<br>STO.                  | Unterbrechung der Programmausführung                                                    | 80    |
| THEN      | T.<br>TH.<br>THE.                  | S. IF-Instruktion                                                                       | 94    |
| TRON      | TR.<br>TRO.                        | Automatischer Stopbefehl nach Abarbeiten jeder Zeile (Debugging, Trace)                 | 85    |
| TROFF     | TROF.                              | Aufhebung von TRON                                                                      | 85    |



| Anweisung | Abkürzung                  | Bemerkungen                                                                                               | Seite |
|-----------|----------------------------|-----------------------------------------------------------------------------------------------------------|-------|
| USING     | U.<br>US.<br>USI.<br>USIN. | Bestimmung des Ausgabeformats.                                                                            | 111   |
| WAIT      | W.<br>WA.<br>WAI.          | Festlegen einer Zeitspanne, nach der nach einer PRINT-Instruktion der Programmablauf automatisch startet. | 116   |

### . Anweisung für Druckerausgabe

| Anweisung | Abkürzung                      | Bemerkungen                         | Seite |
|-----------|--------------------------------|-------------------------------------|-------|
| LLIST     | LL.<br>LLI.<br>LLIS.           | Drucken von Programmlisten.         | 146   |
| LPRINT    | LP.<br>LPR.<br>LPRI.<br>LPRIN. | Ausgabeinstruktion für den Drucker. | 142   |

#### 4. Anweisungen für den Kassettenbetrieb

| Anweisung | Abkürzung                          | Bemerkungen                                                                                   | Seite |
|-----------|------------------------------------|-----------------------------------------------------------------------------------------------|-------|
| CHAIN     | CH.<br>CHA.<br>CHAI.               | Programmgesteuertes Rückladen von spezifizierten Programmblöcken.                             | 161   |
| CLOAD     | CLO.<br>CLOA.                      | Rückladen von spezifizierten Programmen.                                                      | 155   |
| CLOAD?    | CLO.?<br>CLOA.?                    | Vergleich eines im PC-1251 befindlichen Programms mit dem gleichen Kassettenprogramm.         | 156   |
| CSAVE     | CS.<br>CSA.<br>CSAV.               | Abspeichern eines Programms auf Kassette.                                                     | 153   |
| INPUT #   | I. #<br>IN. #<br>INP. #<br>INPU. # | Rückladen von Daten.                                                                          | 168   |
| MERGE     | MER.<br>MERG.                      | Rückladen von Programmen, wobei die im Hauptspeicher befindlichen Programme erhalten bleiben. | 157   |
| PRINT #   | P. #<br>PR. #<br>PRI. #<br>PRIN. # | Abspeichern von Daten.                                                                        | 165   |

# E. Anwendung der Kommandos, Anweisungen und Funktionen in den verschiedenen Betriebsarten

O = Ja X = Nein

|                 | Programmunterstützung |     |         | Programm |
|-----------------|-----------------------|-----|---------|----------|
|                 | RUN                   | PRO | RESERVE |          |
| NEW             | X                     | O   | O       | X        |
| LIST            | X                     | O   | X       | X        |
| RUN             | O                     | X   | X       | X        |
| CONT            | O                     | X   | X       | X        |
| TRON            | O                     | O   | O       | O        |
| TROFF           | O                     | O   | O       | O        |
| PRINT           | X                     | X   | X       | O        |
| INPUT           | X                     | X   | X       | O        |
| PAUSE           | X                     | X   | X       | O        |
| USING           | O                     | O   | O       | O        |
| WAIT            | O                     | O   | O       | O        |
| IF ~ (THEN)     | X                     | X   | X       | O        |
| GOTO            | O                     | X   | X       | O        |
| ON ~ GOTO       | X                     | X   | X       | O        |
| GOSUB           | X                     | X   | X       | O        |
| ON ~ GOSUB      | X                     | X   | X       | O        |
| RETURN          | X                     | X   | X       | O        |
| FOR ~ TO ~ STEP | X                     | X   | X       | O        |
| NEXT            | X                     | X   | X       | O        |
| STOP            | X                     | X   | X       | O        |
| END             | X                     | X   | X       | O        |
| DIM             | O                     | O   | O       | O        |
| CLEAR           | O                     | O   | O       | O        |
| LET             | X                     | X   | X       | O        |
| REM             | X                     | X   | X       | O        |
| DATA            | X                     | X   | X       | O        |
| READ            | X                     | X   | X       | O        |
| RESTORE         | X                     | X   | X       | O        |
| BEEP            | O                     | O   | O       | O        |
| DEGREE          | O                     | O   | O       | O        |
| GRAD            | O                     | O   | O       | O        |
| RADIAN          | O                     | O   | O       | O        |
| AREAD           | X                     | X   | X       | O        |
| RANDOM          | O                     | O   | O       | O        |
| LPRINT          | X                     | X   | X       | O        |
| LLIST           | O                     | O   | X       | X        |
| CSAVE           | O                     | O   | O       | O        |
| CLOAD           | O                     | O   | O       | X        |
| MERGE           | O                     | O   | X       | X        |
| CHAIN           | X                     | X   | X       | O        |
| CLOAD?          | O                     | O   | O       | X        |
| PRINT #         | O                     | O   | O       | O        |
| INPUT #         | O                     | O   | O       | O        |
| PASS            | O                     | O   | O       | X        |

Pseudovariablen und Schlüsselwörter

|              |   |
|--------------|---|
| PI ( $\pi$ ) | 0 |
| MEM          | 0 |
| INKEY\$      | 0 |

Numerische Funktionen

|     |   |
|-----|---|
| SIN | 0 |
| COS | 0 |
| TAN | 0 |
| ASN | 0 |
| ACS | 0 |
| ATN | 0 |
| EXP | 0 |
| LN  | 0 |
| LOG | 0 |
| INT | 0 |
| ABS | 0 |
| SGN | 0 |
| DEG | 0 |
| DMS | 0 |
| RND | 0 |
| SQR | 0 |
| NOT | 0 |

Textfunktionen

|         |   |
|---------|---|
| ASC     | 0 |
| VAL     | 0 |
| LEN     | 0 |
| CHR\$   | 0 |
| STR\$   | 0 |
| LEFT\$  | 0 |
| RIGHT\$ | 0 |
| MID\$   | 0 |

Operatoren

|                                             |   |
|---------------------------------------------|---|
| ^ Exponentiation                            | 0 |
| *, /, +, -<br>arithmetische Operatoren      | 0 |
| +, - (Vorzeichen)                           | 0 |
| >, >=, <, <=, <>,<br>= Vergleichsoperatoren | 0 |
| AND, OR<br>logische Operatoren              | 0 |
| & Hexadezimalzahlen                         | 0 |

## F. Referenzliste

Gottfried, Byron S.

Programmieren mit BASIC  
Düsseldorf: McGraw-Hill  
Book Co., 1978  
ISBN 0-07-092022-2

Hasse, V./W. Stucky

BASIC. Programmieren für Anfänger  
Mannheim: Bibliographisches Institut, 1977  
BI-Hochschultaschenbuch 744  
ISBN 3-411-00744-3

Kahlig, Peter

Graphische Darstellung mit dem Taschencomputer PC-1211 (SHARP)  
aus der Reihe  
Anwendung programmierbarer Taschenrechner Nr. 14  
Braunschweig: Vieweg-Verlag, 1982  
ISBN 3-528-04203-6

Kaucher, E. et al.,

Programmiersprachen im Griff  
Band 3: BASIC  
Mannheim: Bibliographisches Institut, 1981  
BI-Hochschultaschenbuch 797  
ISBN 3-411-00797-4

Kreth, Horst

Lehr- und Übungsbuch für die Rechner SHARP PC-1210 und PC-1211  
aus der Reihe  
programmieren von Taschenrechnern Nr. 7  
Braunschweig: Vieweg-Verlag, 1982  
ISBN 3-528-04212-5

Pol, Bernd

Wie man die BASIC programmiert  
Einführung – Techniken – Fallstudien  
aus der Reihe  
CHIP-Wissen Software  
Würzburg: Vogel-Verlag, 1981  
ISBN 3-8023-0637-6

Dwyer, Thomas & Margot Critchfield

BASIC and the Personal Computer  
Reading: Addison-Wesley Publishing Company, 1978  
ISBN 0-201-01589-7

# G. PC-1251 Technische Daten

|                     |                                                                                                                            |
|---------------------|----------------------------------------------------------------------------------------------------------------------------|
| Modell:             | PC-1251 Taschencomputer                                                                                                    |
| Rechenstellen:      | 10 Stellen Mantisse, 2 Stellen Exponent                                                                                    |
| Rechensystem:       | AEL mit Hierarchie                                                                                                         |
| Anzeige:            | Flüssigkristallanzeige, 24 Stellen in 5 x 7-Matrix                                                                         |
| Tastatur:           | 52 Tasten<br>Alphatastatur (Schreibmaschine)<br>Zifferntastatur (10-er Block)<br>Funktionstasten<br>programmierbare Tasten |
| Programmiersprache: | BASIC                                                                                                                      |
| CPU:                | CMOS 8-bit                                                                                                                 |
| Betriebssystem:     | ROM 24 kByte                                                                                                               |
| Speicherkapazität:  | RAM:<br>500 Byte System<br>3486 Byte BASIC-Programm<br>208 Byte Standardvariable<br>48 Byte Reserve-Speicher               |
| Speicherschutz:     | Programm-Daten-Reserve-Speicher sind beim Ausschalten geschützt.                                                           |
| Stromversorgung:    | 6 Volt, 2 Stück Lithium-Batterien (z.B. VARTA CR2032)                                                                      |
| Stromverbrauch:     | 0,03W bei 6 Volt                                                                                                           |
| Betriebszeit:       | ca. 300 Stunden mit einem Batteriesatz                                                                                     |
| Betriebstemperatur: | 0° ~ 40°C                                                                                                                  |
| messungen:          | 135 x 70 x 9,5 (B, T, H) in mm                                                                                             |
| Gewicht:            | ca. 115 g                                                                                                                  |
| Zubehör:            | 1 Schutzhülle, 2 Batterien (eingebaut), 2 Tastaturscha-blone, 1 Betriebsanleitung                                          |
| Option:             | CE-125: Drucker/Mikrokassettenrekorder mit integrierter Schnittstelle für ein externes Bandgerät                           |

# H. CE-125 (Option) Technische Daten

## 1. Drucker

|                       |                                   |
|-----------------------|-----------------------------------|
| Zeichen/Zeile:        | 24                                |
| Druckgeschwindigkeit: | 0,8 Zeilen/sec.                   |
| Druckprinzip:         | Thermodruck in 5 x 7-Matrix       |
| Papiertransport:      | manuell oder programmkontrolliert |
| Papierabmessung:      | 58 x 18 (mm) Thermopaper EA-1250P |

## 2. Mikrokassettenrekorder

|                      |                     |
|----------------------|---------------------|
| Kassette:            | Mikrokassette MC-60 |
| Bandgeschwindigkeit: | 2,4 cm/sec.         |
| Fernbedienung:       | Abschaltbar         |

---

|                     |                                                                                                                                 |
|---------------------|---------------------------------------------------------------------------------------------------------------------------------|
| Stromversorgung:    | 4,8 Volt (4 x NiCd-Batterien)                                                                                                   |
| Ladegerät:          | EA-23E                                                                                                                          |
| Betriebszeit:       | ca. 2000 Zeilen Druckbetrieb oder<br>ca. 4 Std. Kassettenbetrieb                                                                |
| Betriebstemperatur: | 5° ~ 35°C                                                                                                                       |
| Abmessungen:        | 205 x 149 x 23 (B, T, H in mm)                                                                                                  |
| Gewicht:            | ca. 550 g                                                                                                                       |
| Zubehör:            | 1 Transportbehälter<br>1 Ladegerät EA-23E<br>1 Programm-Kassette<br>3 Papierrollen<br>1 Überspielkabel<br>1 Bedienungsanleitung |



## Programmbeispiele

Durch Studium der Funktionsbeschreibungen bis zu dieser Seite hat der Anwender sich wahrscheinlich einiges Wissen über eine ganze Reihe von Programmkommandos verschafft. Um mit BASIC frei zu programmieren, ist es jedoch erforderlich, über die in der Bedienungsanleitung aufgeführten Programme hinaus selbst Programme zu schreiben. Wie beim Autofahren oder Tennisspielen kann man auch beim Programmieren nur dann echte Fortschritte machen, wenn man möglichst viele Programme selbst erstellt. Auch die Bezugnahme auf von anderen geschriebene Programme ist sehr wichtig.

Zur Bezugnahme enthalten die folgenden Seiten eine Reihe von Programmen unter Verwendung von BASIC-Kommandos.

**Der Sharp Konzern und/oder seine Niederlassungen sind in keiner Weise für eventuelle Verluste oder Beschädigungen aufgrund der in dieser Bedienungsanleitung und der zugehörigen Software-Cassette verwendeten Programme verantwortlich oder haftbar zu machen.**

Hinweis: Die in diesen Beispielen enthaltenen Programme sind auch auf der zum Zubehör des CE-125 gehörenden Microcassette gespeichert.

**Die Programmcassette ist in Englisch abgefaßt.**

# INHALT

| (Programmname)                                              | (Seite) |
|-------------------------------------------------------------|---------|
| ● WURZEL EINER GLEICHUNG .....                              | 201     |
| ● MITTELWERT, VARIANZ UND STANDARDABWEICHUNG .....          | 205     |
| ● SCHNITT ZWISCHEN KREISEN UND GERADEN .....                | 210     |
| ● BERECHNUNG DER ANZAHL VON TAGEN ZWISCHEN ZWEI DATEN ..... | 214     |
| ● SCHREIBÜBUNGEN .....                                      | 218     |
| ● MONDLANDUNG .....                                         | 222     |
| ● GEDÄCHTNISTEST .....                                      | 226     |
| ● KÄFERJAGD .....                                           | 227     |
| ● DOPPELTE ROTATION .....                                   | 236     |

## Zur Anzeige der im Programm selbst verbrauchten Bytes.

Am Ende des Programmlistings wird angezeigt, wieviel Bytes im jeweiligen Programm verbraucht werden.

Beim Programm MONDLANDUNG sind es z.B. 3486 Bytes. Um die Anzahl zu ermitteln, geht man folgendermaßen vor:

In der RUN-Stellung

1) CLEAR

2) 3486  MEM  zeigt die Anzahl der Bytes an.

### ÜBERBLICK

Die Wurzel einer Gleichung zu ermitteln ist im allgemeinen sehr zeitraubend. Nachstehend ist ein Verfahren zur Wurzelberechnung nach Newton aufgeführt. Bei dem Verfahren nach Newton variiert der Anfangspunkt automatisch gemäß dem festgelegten Intervall.

### INHALT (Formel)

$$X_{n+1} = X_n - \frac{f(X_n)}{f'(X_n)}$$

Beträgt der absolute Wert in der Differenz zwischen  $X_n$  und  $X_{n+1}$  weniger als  $10^{-8}$ , wird  $X_n$  als Wurzel angezeigt.

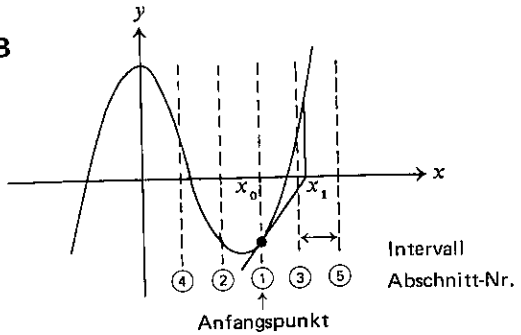
Der Differentialwert  $f'(X)$  wird wie folgt definiert:

$$f'(X) = \frac{f(X+h) - f(X)}{h} \quad (h : \text{Minutenwert})$$

Um  $10^{-8}$  abzuwandeln, verändert man E-8 in Zeile 340.

### ANLEITUNG ZUM BETRIEB

Eingabe: Anfangswert  
 Minutenwert  
 Intervall



Ausgabe: Wurzelwert (die **ENTER** -Taste drücken, um die Wurzel im nächsten Intervall zu ermitteln).

### BEISPIEL

$x^3 - 2x^2 - x + 2 = 0$  (Wurzel = -1, 1, 2)  
 Anfangspunkt = 0  
 Minutenwert =  $10^{-4}$   
 Intervall = 0,5

Die Funktion wird als Unterprogramm ab Zeile 500 geschrieben.

Dazu wird im PRO-Modus folgende Unterroutine eingegeben:

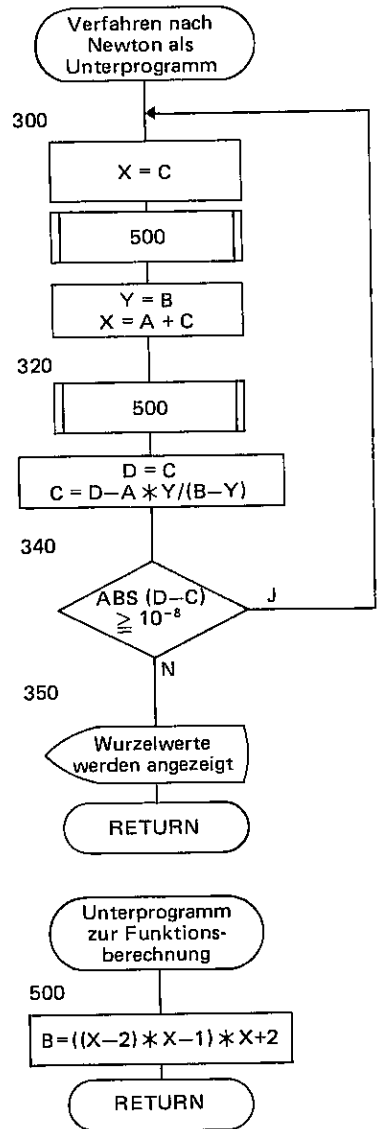
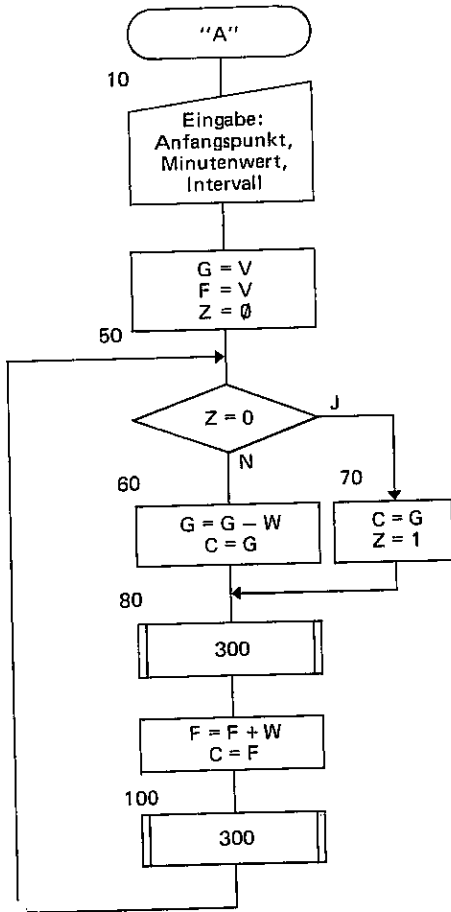
500 B = ((X - 2) \* X - 1) \* X + 2 **ENTER**

510 RETURN **ENTER**

## TASTENBETÄTIGUNG

| Schritt-Nr. | Eingabe             | Anzeige         | Bemerkungen                                                               |
|-------------|---------------------|-----------------|---------------------------------------------------------------------------|
| 1           | <b>DEF</b> <b>A</b> | ANFANGSWERT = _ |                                                                           |
| 2           | 0 <b>ENTER</b>      | MINUTE = _      |                                                                           |
| 3           | 0.0001 <b>ENTER</b> | INTERVALL = _   |                                                                           |
| 4           | 0.5 <b>ENTER</b>    | ERGEBNIS = _ 2. |                                                                           |
| 5           | <b>ENTER</b>        | ERGEBNIS = 1.   | Zum Auffinden der nächsten Wurzel nochmal die <b>ENTER</b> -Taste drücken |
| 6           | <b>ENTER</b>        | ERGEBNIS = -1.  |                                                                           |
| 7           | <b>ENTER</b>        | ERGEBNIS = 1.   |                                                                           |
| 8           | <b>ENTER</b>        | ERGEBNIS = -1.  |                                                                           |
| 9           | <b>ENTER</b>        | ERGEBNIS = -1.  |                                                                           |
| 10          | <b>ENTER</b>        | ERGEBNIS = -1.  |                                                                           |
| 11          | <b>ENTER</b>        | ERGEBNIS = 2.   |                                                                           |

# FLUSSDIAGRAMM



# PROGRAMMLISTING

```

10:"A": INPUT "ANFANGSW
 ERT=";I;V
20:INPUT "MINUTE=";A
30:INPUT "INTERVALL=";W
40:G=V:F=V:Z=0
50:IF Z=0 GOTO 70
60:G=G-W:C=G: GOTO 80
70:C=G:Z=1
80:GOSUB 300
90:F=F+W:C=F
100:GOSUB 300
110:GOTO 50
120:END
300:X=C: GOSUB 500
310:Y=B:X=A+C
320:GOSUB 500
330:D=C:C=D-A*Y/(B-Y)
340:IF ABS (D-C)>=E-8
 GOTO 300
350:BECP 3: PRINT "ERGES
 NIS=";C
360:RETURN
500:B=((X-2)*X-1)*X+2
510:RETURN
520:END

```

263 Bytes

# SPEICHERINHALT

|   |                   |
|---|-------------------|
| A | Minutenwert       |
| B | f(x)              |
| C | $X_0$             |
| D | f(x+h)            |
| E |                   |
| F | ✓                 |
| G | ✓                 |
| H |                   |
| I |                   |
| J |                   |
| K |                   |
| L |                   |
| M |                   |
| N |                   |
| O |                   |
| P |                   |
| Q |                   |
| R |                   |
| S |                   |
| T |                   |
| U |                   |
| V | Anfangswert       |
| W | Intervall         |
| X | X                 |
| Y | f(x)              |
| Z | Anfangsmarkierung |

**Programmname: MITTELWERT, VARIANZ UND STANDARDABWEICHUNG**

**ÜBERBLICK**

Nach Eingabe der Daten wird die Gesamtsumme, der Mittelwert, die Varianz und die Standardabweichung berechnet.

Korrektur der Eingabedaten als auch der Daten mit Gewichtung ist möglich.

**INHALT (Formeln)**

Gesamtsumme  $\Sigma x_i \cdot f_i$                       Standardabweichung  $\sigma = \sqrt{\sigma^2}$

Mittelwert  $\bar{x} = \frac{\Sigma x_i \cdot f_i}{\Sigma f_i}$

Varianz  $\sigma^2 = \frac{\Sigma (x_i - \bar{x}) f_i}{\Sigma f_i - 1}$

**ANLEITUNG ZUM BETRIEB**

- 1. **DEF** **A** : Wahl, ob Daten mit oder ohne Gewichtung verarbeitet werden sollen, Eingabe der Daten.
- 2. **DEF** **B** : Auffinden von Korrekturpositionen in den Daten.  
**DEF** **C** : Korrektur der Daten.
- 3. **DEF** **D** : Berechnung von Gesamtsumme, Mittelwert, Varianz und Standardabweichung.

**BEISPIEL**

|       |      |      |      |      |      |
|-------|------|------|------|------|------|
| $x_i$ | 14.1 | 14.2 | 14.3 | 14.4 | 14.5 |
| $f_i$ | 8    | 19   | 23   | 15   | 10   |

(Daten mit Gewichtung)

# TASTENBETÄTIGUNG

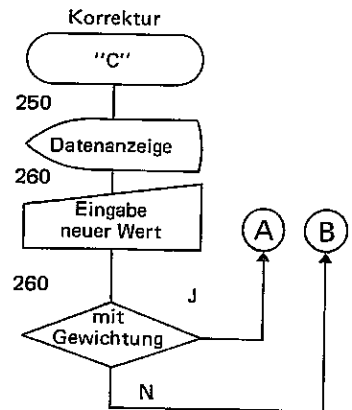
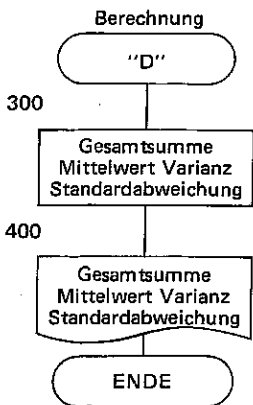
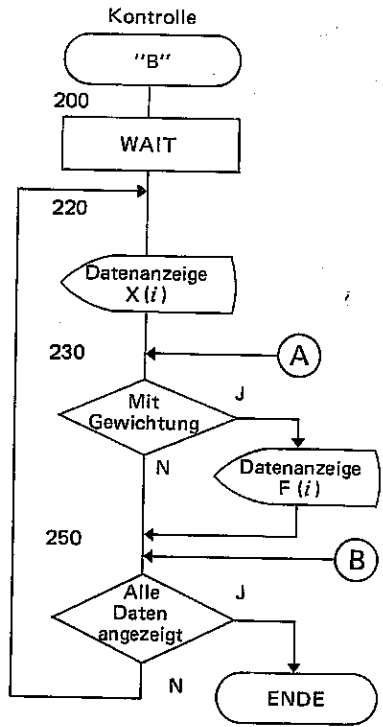
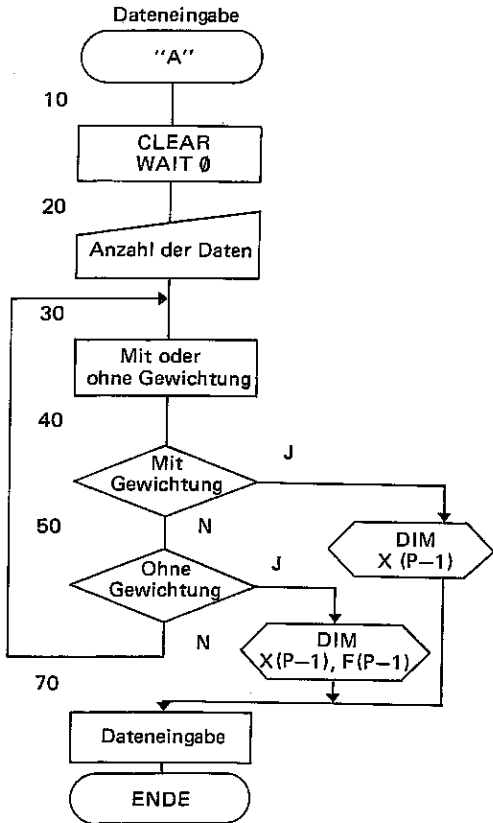
| Schritt-Nr. | Eingabe    | Anzeige               | Bemerkungen                            |
|-------------|------------|-----------------------|----------------------------------------|
| 1           | DEF A      | ANZAHL DER DATEN = _  | Eingabe der Anzahl der Daten           |
| 2           | 5 ENTER    | GEWICHTUNG=1/KEINE=2? | Wahl ob Daten mit oder ohne Gewichtung |
| 3           | 1 ENTER    | X (1) =               |                                        |
|             |            | ?                     |                                        |
| 4           | 14.1 ENTER | F (1) =               |                                        |
|             |            | ?                     |                                        |
| 5           | 8 ENTER    | X (2) =               |                                        |
|             |            | ?                     |                                        |
|             | ⋮          | ⋮                     |                                        |
| 12          | 14.5 ENTER | F (5) =               |                                        |
|             |            | ?                     |                                        |
| 13          | 10 ENTER   | >                     | Ende des Programmteils                 |



# TASTENBETÄTIGUNG

| Schritt-Nr. | Eingabe    | Anzeige                 | Bemerkungen                                          |
|-------------|------------|-------------------------|------------------------------------------------------|
| 1           | DEF B      | X (1) = 14.1            |                                                      |
| 2           | ENTER      | F (1) = 8               |                                                      |
| 3           | ENTER      | X (2) = 14.1            | DEF C starten, um fehlerhafte Eingabe zu korrigieren |
| 4           | DEF C      | X (2) =                 |                                                      |
|             |            | NEUER WERT = ? _        | Eingabe des neuen Wertes                             |
| 5           | 14.2 ENTER | F (2) = 19              |                                                      |
|             | ENTER      | ⋮                       |                                                      |
|             | ⋮          | ⋮                       |                                                      |
|             |            |                         |                                                      |
| 1           | DEF D      | GESAMTSUMME = 1072.5    | Anzeige der Gesamtsumme                              |
| 2           | ENTER      | MITTELWERT = 14.3       | Anzeige des Mittelwertes                             |
| 3           | ENTER      | VARIANZ=1.432432432E-02 | Anzeige der Varianz                                  |
| 4           | ENTER      | STANDARDABWEICHUNG =    | Anzeige der Standardabweichung                       |
| 5           | ENTER      | 1.196842693E-01         |                                                      |
| 6           | ENTER      | >                       | Programmende                                         |

# FLUSSDIAGRAMM



## PROGRAMMLISTING

```

10:"A": CLEAR : WAIT 0
20:INPUT "ANZAHL DER DA
TEN=";P
30:INPUT "GEWICHTUNG=1/
KEINE=2?";A
40:IF A=2 DIM X(P-1):
GOTO 70
50:IF A=1 DIM X(P-1),F(
P-1): GOTO 70
60:GOTO 30
70:FOR I=0 TO P-1
80:B$="X(" + STR$(I+1)+
")="
85:PAUSE B$: INPUT X(I)
: GOTO 100
90:GOTO 85
100:IF A=2 GOTO 150
120:B$="F(" + STR$(I+1)+
")="
130:PAUSE B$: INPUT F(I)
: GOTO 150
140:GOTO 130
150:NEXT I: END
200:"B": WAIT :I=0
210:B$="X(" + STR$(I+1)+
")=":J=1: PRINT B$;X
(I)
230:IF A=1 LET B$="F(" +
STR$(I+1)+")=":
PRINT B$;F(I):J=2
240:I=I+1
250:IF I=P END
255:GOTO 210
260:"C": PAUSE B$: IF
LEFT$(B$,1)="X"
INPUT "NEUER WERT=";
X(I): GOTO 290
270:IF LEFT$(B$,1)="F"
INPUT "NEUER WERT=";
F(I): GOTO 290
280:GOTO 250
290:IF J=1 GOTO 230
291:GOTO 210
300:"D":N=0:T=0:S=0: FOR
I=0 TO P-1:X=X(I)
305:F=1: IF A=1,LET F=F(
I)

```

```

310:N=N+F:T=T+F*X:S=S+F*
X*X: NEXT I
400:WAIT :X=T/N:Q=(S-N*X
*X)/(N-1):S=SQ:
PRINT "GESAMTSUMME="
;T: PRINT "MITTELWER
T=";X
410:PRINT "VARIANZ=";Q:
PRINT "STANDARDABWEI
CHUNG=": PRINT S
420:END

```

650 Bytes

## SPICHERINHALT

|        |                    |
|--------|--------------------|
| A      |                    |
| B      |                    |
| C      |                    |
| D      |                    |
| E      |                    |
| F      |                    |
| G      |                    |
| H      |                    |
| I      |                    |
| J      | Marke              |
| K      |                    |
| L      |                    |
| M      |                    |
| N      |                    |
| O      |                    |
| P      | Anzahl der Daten   |
| Q      | Varianz            |
| R      |                    |
| S      | Standardabweichung |
| T      | Gesamtsumme        |
| U      |                    |
| V      |                    |
| W      |                    |
| X      | Mittelwert         |
| Y      |                    |
| Z      |                    |
| X(P-1) | Daten              |
| F(P-1) | Daten              |

# Programmname: SCHNITT ZWISCHEN KREISEN UND GERADEN

## ÜBERBLICK

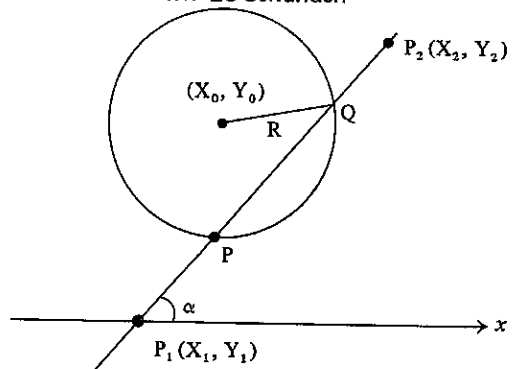
Es werden die Schnittpunkte zwischen Kreisen und Geraden in der  $X - Y$  Ebene bestimmt.

## INHALT (Formeln)

Die Schnittpunkte zwischen einem Kreis und einer Geraden seien  $P$  und  $Q$ .

Hinweis: Die Winkel müssen in Grad, Minuten und Sekunden folgendermaßen eingegeben werden.

$$123.1423 = 123 \text{ Grad } 14 \text{ Minuten } 23 \text{ Sekunden}$$



## ANLEITUNG ZUM BETRIEB

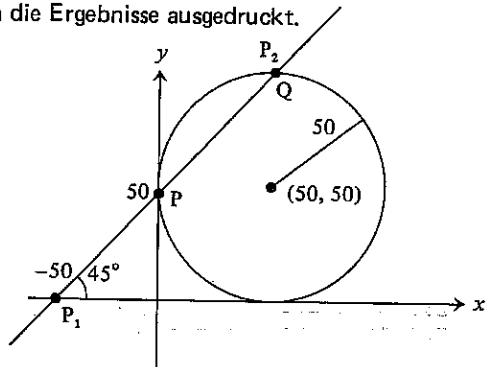
1. Wird die Gerade über 2 Punkte bestimmt, so wird das Programm über **DEF** **A** gestartet.

Wird die Gerade durch 1 Punkt und 1 Winkel bestimmt, starten Sie die Programmausführung über **DEF** **B**.

2. Nach der Dateneingabe werden die Ergebnisse ausgedruckt.

## BEISPIEL

$$\begin{aligned} X_1 &= -50 \\ Y_1 &= 0 \\ X_2 &= 50 & X_P &= 0 \\ Y_2 &= 100 & Y_P &= 50 \\ X_0 &= 50 & X_Q &= 50 \\ Y_0 &= 50 & Y_Q &= 100 \\ R &= 50 \\ \alpha &= 45^\circ \end{aligned}$$



Hinweis: Die Koordinatenwerte sind bis auf 5 Dezimalstellen genau.

# TASTENBETÄTIGUNG

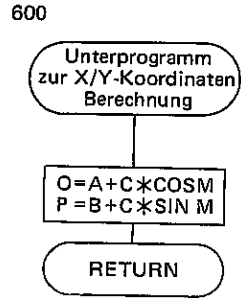
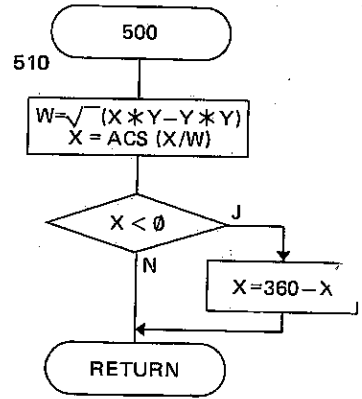
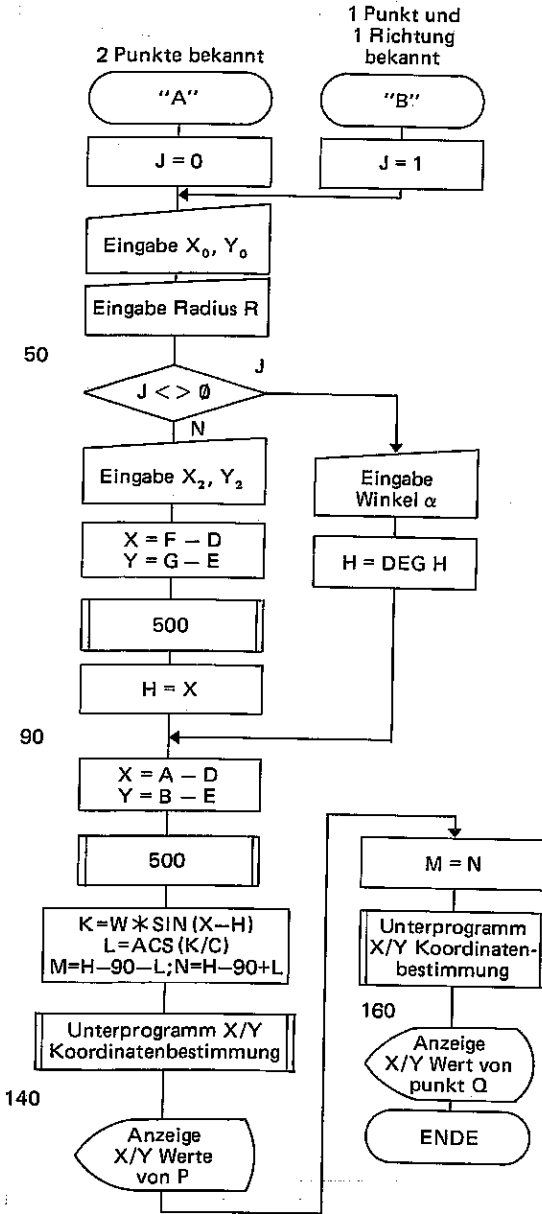
(2 Punkte der Geraden sind bekannt)

| Schritt-Nr. | Eingabe   | Anzeige             | Bemerkungen  |
|-------------|-----------|---------------------|--------------|
| 1           | DEF A     | $X_0 = \_$          |              |
| 2           | 50 ENTER  | $Y_0 = \_$          |              |
| 3           | 50 ENTER  | $R = \_$            |              |
| 4           | 50 ENTER  | $X_1 = \_$          |              |
| 5           | -50 ENTER | $Y_1 = \_$          |              |
| 6           | 0 ENTER   | $X_2 = \_$          |              |
| 7           | 50 ENTER  | $Y_2 = \_$          |              |
| 8           | 100 ENTER | P: 0.0000 49.9999   | $(x_p, y_p)$ |
| 9           | ENTER     | Q: 50.0000 100.0000 | $(x_q, y_q)$ |

(1 Punkt und 1 Winkel sind bekannt)

| Schritt-Nr. | Eingabe   | Anzeige             | Bemerkungen  |
|-------------|-----------|---------------------|--------------|
| 1           | DEF B     | $X_0 = \_$          |              |
| 2           | 50 ENTER  | $Y_0 = \_$          |              |
| 3           | 50 ENTER  | $R = \_$            |              |
| 4           | 50 ENTER  | $X_1 = \_$          |              |
| 5           | -50 ENTER | $Y_1 = \_$          |              |
| 6           | 0 ENTER   | $A = \_$            |              |
| 7           | 45 ENTER  | P: 0.0000 49.9999   | $(x_p, y_p)$ |
| 8           | ENTER     | Q: 50.0000 100.0000 | $(x_q, y_q)$ |
| 9           | ENTER     | >                   |              |

# FLUSSDIAGRAMM



## PROGRAMMLISTING

```

10:"A":J=0: GOTO 30
20:"B":J=1
30:DEGREE : INPUT "X0="
 "A,"Y0=" :B,"R=" :C
40:INPUT "X1=" :D,"Y1="
 "E
50:IF J<>0 INPUT "A=" :
 H:H= DEG H: GOTO 90
60:INPUT "X2=" :F,"Y2="
 "G
70:X=F-D:Y=G-E: GOSUB 5
 00
80:H=X
90:X=A-D:Y=B-E: GOSUB 5
 00
100:K=W* SIN (X-H)
110:L= ACS (K/C)
120:M=H-90-L:N=H-90+L
130:GOSUB 600
140:PRINT USING "#####.
 #####":P:" :O:P
150:M=N: GOSUB 600
160:PRINT "Q:" :O:P
170:END
500:W=J(X*X+Y*Y)
510:X= ACS (X/W): IF Y<0
 LET X=360-X
520:RETURN
600:O=A+C* COS M:P=B+C*
 SIN M: RETURN
610:END

```

339 Bytes

## SPEICHERINHALT

|   |                    |
|---|--------------------|
| A | $X_0$              |
| B | $Y_0$              |
| C | R                  |
| D | $X_1$              |
| E | $Y_1$              |
| F | $X_2$              |
| G | $Y_2$              |
| H | ✓                  |
| I |                    |
| J | ✓                  |
| K | $h$                |
| L | $\alpha$           |
| M | $Q_P$              |
| N | $Q_Q$              |
| O | $X_P, X_Q$         |
| P | $Y_P, Y_Q$         |
| Q |                    |
| R |                    |
| S |                    |
| T |                    |
| U |                    |
| V |                    |
| W | L                  |
| X | $\Delta X, \theta$ |
| Y | $\Delta Y$         |
| Z |                    |

# Programmname: BERECHNUNG DER ANZAHL VON TAGEN ZWISCHEN ZWEI DATEN

## ÜBERBLICK

Wieviele Tage sind seit Ihrer Geburt vergangen? Dieses Programm beantwortet solche Fragen. Nach Eingabe eines bestimmten Datums gibt das Programm die Anzahl der seitdem verstrichenen Tage an.

## ANLEITUNG ZUM BETRIEB

Das Programm wird durch Drücken von   initiiert.

### [Eingabe]

AUSGANGSJAHR

MONAT

TAG

ZIELJAHR

MONAT

TAG

Das Programm wird durch Eingabe von   anstelle der Jahreszahl beendet.

### [Beispiel]

von 1976 Jahr 10 Monat 5 Tag

bis 1982 Jahr 6 Monat 4 Tag: 2068 Tage

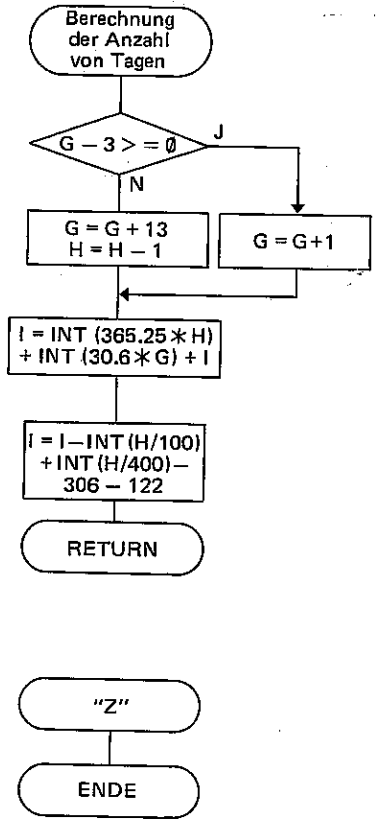
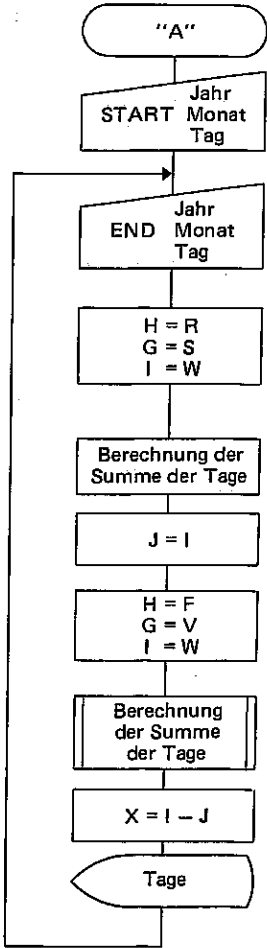
bis 1985 Jahr 1 Monat 1 Tag: 3010 Tage



# TASTENBETÄTIGUNG

| Schritt-Nr. | Eingabe    | Anzeige         | Bemerkungen                           |
|-------------|------------|-----------------|---------------------------------------|
| 1           | DEF A      | START JAHR = -- |                                       |
| 2           | 1976 ENTER | MONAT = --      | Eingabe des Ausgangsdatums: 5.10.1976 |
| 3           | 10 ENTER   | TAG = --        |                                       |
| 4           | 5 ENTER    | END JAHR = --   |                                       |
| 5           | 1982 ENTER | MONAT = --      | Eingabe des Zieldatums: 4.6.1982      |
| 6           | 6 ENTER    | TAG = --        |                                       |
| 7           | 4 ENTER    | TAGE = 2068     |                                       |
| 8           | ENTER      | END JAHR = --   |                                       |
| 9           | 1985 ENTER | MONAT = --      | Eingabe des Zieldatums: 1.1.1985      |
| 10          | 1 ENTER    | TAG = --        |                                       |
| 11          | 1 ENTER    | TAGE = 3010     |                                       |
| 12          | ENTER      | END JAHR = --   |                                       |
| 13          | DEF Z      | >               | Programmende                          |

# FLUSSDIAGRAMM



## PROGRAMMLISTING

```

10: "A"
20: INPUT "START JAHR=";
 R, "MONAT="; S, "TAG=";
 T
30: INPUT "END JAHR="; F,
 "MONAT="; V, "TAG="; W
50: H=R
60: G=S: I=T
70: GOSUB 500
80: J=I
100: H=F
110: G=V: I=W
120: GOSUB 500
130: X=I-J
140: WAIT : USING : PRINT
 "TAGE=", X
150: GOTO 30
500: IF G-3 >= 0 LET G=G+1:
 GOTO 520
510: G=G+13: H=H-1
520: I= INT (365.25*H)+
 INT (30.6*G)+I
530: I=I- INT (H/100)+
 INT (H/400)-306-122:
 RETURN
600: "Z": END

```

270 Bytes

## SPEICHERINHALT

|   |                        |
|---|------------------------|
| A |                        |
| B |                        |
| C |                        |
| D |                        |
| E |                        |
| F | Jahr (nach Berechnung) |
| G | ✓                      |
| H | ✓                      |
| I | ✓                      |
| J | ✓                      |
| K |                        |
| L |                        |
| M |                        |
| N |                        |
| O |                        |
| P |                        |
| Q |                        |
| R | Startjahr              |
| S | Startmonat             |
| T | Starttag               |
| U |                        |
| V | Zielmonat              |
| W | Zieltag                |
| X | Gewünschter Tag        |
| Y |                        |
| Z |                        |

## **ÜBERBLICK**

Schnelle Tastenbetätigung bringt erhebliche Zeitersparnis. Können Sie die Tastatur schnell und fehlerfrei bedienen? Dieses Programm ist eine Übung, mit der Sie Ihr Tempo beim Eintasten erhöhen. Das Ergebnis ist eine schnelle, effektivere Programmeingabe ins Gerät.

## **INHALT**

Die Anzahl der Buchstaben (3 bis 6) wird durch eine zufallszahlengenerierende Funktion festgelegt.

Mit der gleichen Funktion wird der Zeichen-String (A bis Z) ausgewählt.

## **ANLEITUNG ZUM BETRIEB**

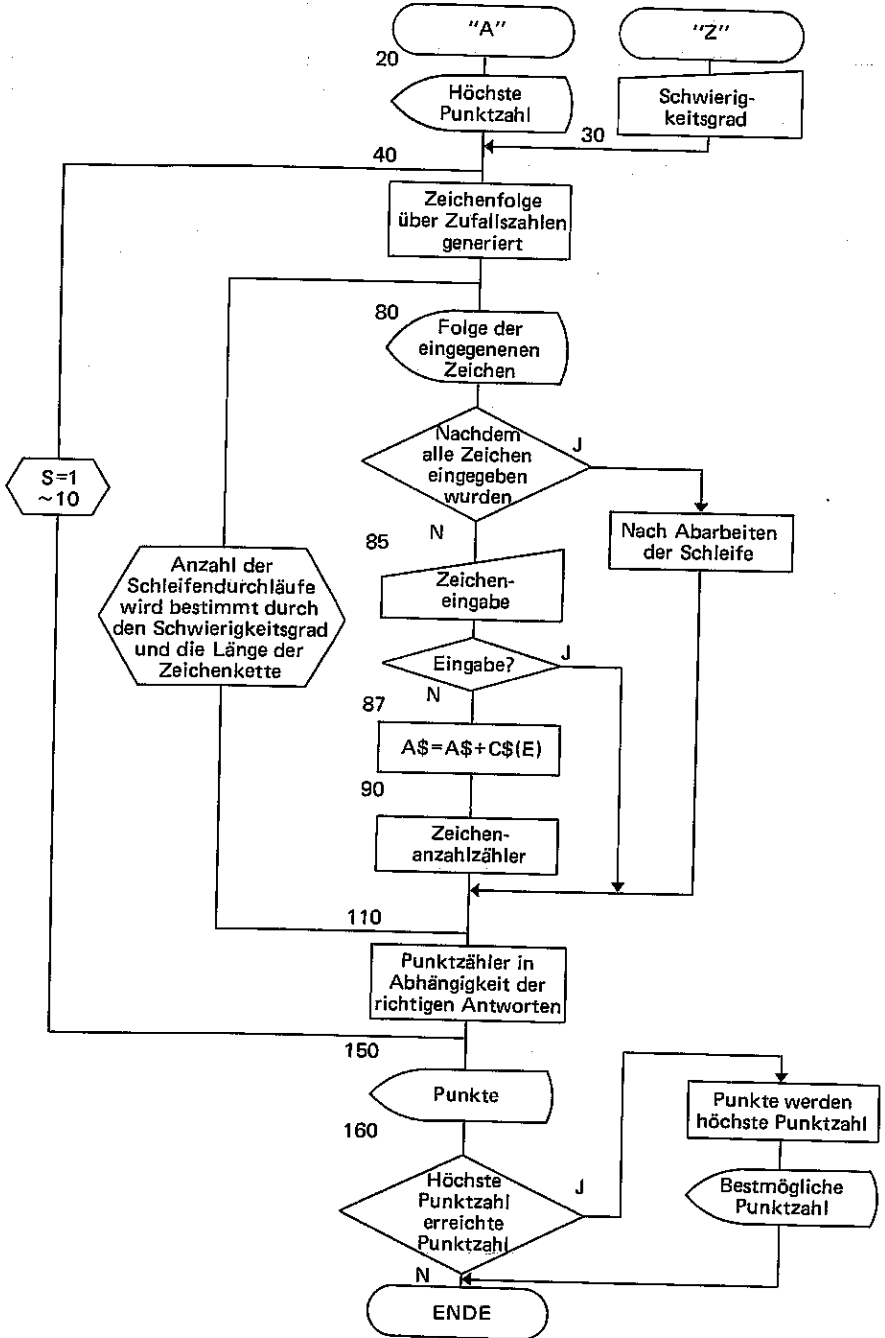
Wenn der Summer ertönt, wird eine Schreibübung von 3 bis 6 Buchstaben angezeigt. Mit der Tastatur gibt man die gleichen Buchstaben innerhalb der vorgegebenen Zeit ein. Bei korrekter Eingabe erhält man 10 Punkte, und 5 Punkte, wenn mehr als 50% korrekt ist. Falls die vorgegebene Zeit überschritten wird, erscheint eine weitere Übung in der Anzeige.

Die festgelegte Zeit richtet sich nach der Zahl der angezeigten Buchstaben und nach den Schwierigkeitsgraden (1, 2, 3). Bei Grad 1 ist die Zeit am kürzesten und bei Grad 3 am längsten. Für jeden Grad gibt es 10 Schreibübungen. Es gilt, möglichst 100 Punkte zu erreichen.

# TASTENBETÄTIGUNG

| Schritt-Nr. | Eingabe | Anzeige                 | Bemerkungen                                                   |
|-------------|---------|-------------------------|---------------------------------------------------------------|
| 1           | DEF Z   | GRAND (1, 2, 3)?_       | Eingabe des Schwierigkeitsgrades                              |
| 2           | 1 ENTER | A Z B D C               |                                                               |
| 3           | A       | A Z B D C A             |                                                               |
| 4           | Z       | A Z B D C A Z           |                                                               |
| ⋮           | ⋮       | ⋮                       |                                                               |
|             |         | IHRE PUNKTZAHL = 80     | Nach 10 Spieldurchläufen wird die Punktzahl angezeigt         |
|             |         | BESTMÖGLICHE PUNKTZAHL  | Erscheint nur, wenn man die höchste Punktzahl erreicht hat    |
|             |         | >                       |                                                               |
| 1           | DEF A   | HOECHSTE PUNKTZAHL = 80 | Erscheint, wenn man im selben Schwierigkeitsgrad weiter macht |
|             |         | B W V S                 |                                                               |
| 2           | B       | BWVS B                  |                                                               |
| ⋮           | ⋮       | ⋮                       |                                                               |
|             |         | IHRE PUNKTZAHL = 60     |                                                               |
|             |         | >                       |                                                               |

# FLUSSDIAGRAMM



## PROGRAMMLISTING

```

10:"Z": CLEAR : DIM B$(
5):C$(5): RANDOM
15:INPUT "GRAD(1,2,3)?"
:L: WAIT 0
17:IF (L=1)+(L=2)+(L=3)
<>1 THEN 15
18:GOTO 30
20:"A": WAIT 0:P=0:
PAUSE "HOECHSTE PUNK
TZAHL=":X
30:FOR S=1 TO 10
40:B= RND 4+2:Y$="":R=
INT (B/2)
50:FOR C=0 TO B-1:C$(C)
=" "
60:D= RND 26:B$(C)=
CHR$(D+&40):Y$=Y$+
CHR$(D+&40): NEXT C
:A$=""
70:BEEP 3:E=0: WAIT 30:
USING "#####"
80:FOR W=1 TO B*10/L:
PRINT Y$;" ":A$:
IF E=B LET W=B*20/L:
GOTO 100
85:C$(E)= INKEY$: IF C
$(E)=" " THEN 100
87:A$=A$+C$(E)
90:E=E+1
100:NEXT W:Q=0
110:FOR W=0 TO B-1: IF B
$(W)=C$(W) LET Q=Q+1
120:NEXT W: IF Q<=R THEN
150
130:IF Q=B LET P=P+10:
GOTO 150
140:P=P+5
150:NEXT S: USING : BEEP
3: PAUSE "IHRE PUNKT
ZAHL=":P
160:IF P>X LET X=P: WAIT
100: PRINT "BESTMOEG
LICHE PUNKTZAHL"
170:END

```

## SPEICHERINHALT

|        |                    |
|--------|--------------------|
| AS     | ✓                  |
| B      | ✓                  |
| C      | Schleifenvariable  |
| D      | ✓                  |
| E      | ✓                  |
| F      |                    |
| G      |                    |
| H      |                    |
| I      |                    |
| J      |                    |
| K      |                    |
| L      | Schwierigkeitsgrad |
| M      |                    |
| N      |                    |
| O      |                    |
| P      | Punktzahl          |
| Q      | ✓                  |
| R      | ✓                  |
| S      | Schleifenvariable  |
| T      |                    |
| U      |                    |
| V      |                    |
| W      | Schleifenvariable  |
| X      | Höchste Punktzahl  |
| Y\$    | ✓                  |
| Z      |                    |
| B\$(5) | ✓                  |
| C\$(5) | ✓                  |

491 Bytes

## ÜBERBLICK

In diesem Spiel geht es darum, eine Rakete, die nur einen begrenzten Treibstoffvorrat hat, möglichst sanft zu landen. Die Rakete ist im freien Fall. Die Maschine wird benutzt, um den freien Fall zu bremsen. Findet die Zündung zu früh statt oder wird zu viel Treibstoff verbraucht, wird die Rakete zurück ins All geworfen und verglüht.

Ist der Treibstoff verbraucht, schlägt sie auf dem Planeten auf und explodiert. Das Ziel ist, die Rakete so sanft wie möglich zu landen, indem man die Maschine kontrolliert und den Treibstoffverbrauch beobachtet.

## INHALT (Formeln)

Die Schwerkraft bei 5 m/(Zeiteinheit)<sup>2</sup>.

Werden 5 Treibstoffeinheiten pro Zeiteinheit verbrannt, wird die Schwerkraft ausgeglichen.

$$H = H_0 + V_0 t + \frac{1}{2} a t^2$$

$$V = V_0 + a t$$

$$V^2 = V_0^2 + 2 a H$$

$$H_0 = 500, V_0 = -50, F_0 = 200$$

H : Höhe

V : Geschwindigkeit

a : Beschleunigung

t : Zeit

V<sub>0</sub> : Anfangsgeschwindigkeit

H<sub>0</sub> : Anfangshöhe

V<sub>0</sub> : Anfangsgeschwindigkeit

F<sub>0</sub> : Treibstoffmenge

F : Verbrauchter Treibstoff

Die Anfangshöhe, Treibstoffmenge und Zeit sind in Zeile 30 durch die DATA-Anweisung gespeichert. Die Mengen können durch Umschreiben der Programmzeile geändert werden.

## ANLEITUNG ZUM BETRIEB

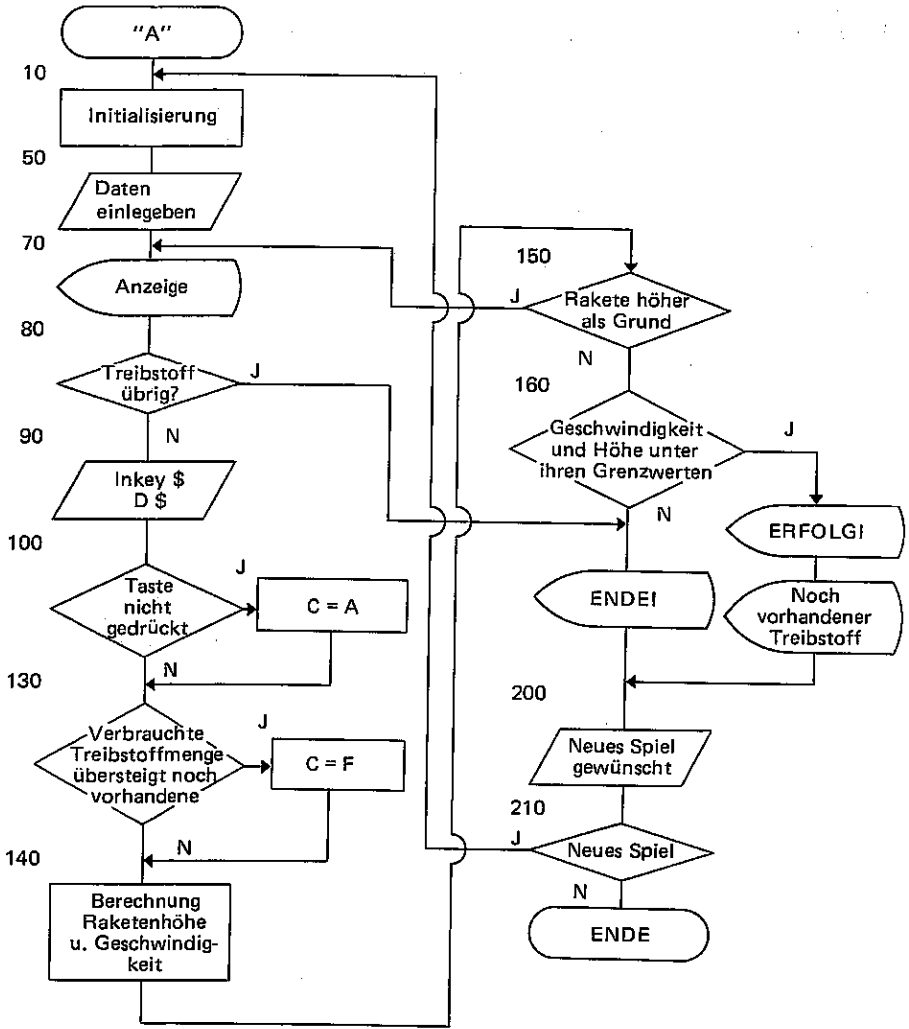
1. Das Programm wird über **DEF** **A** gestartet. Über **Ø** ~ **9** wird die vor Landung der Rakete benötigte Treibstoffmenge ausgeglichen.



# TASTENBETÄTIGUNG

| Schritt-Nr. | Eingabe           | Anzeige                  | Bemerkungen                                       |
|-------------|-------------------|--------------------------|---------------------------------------------------|
| 1           | DEF A             | *** START ***            |                                                   |
|             |                   |                          | Tasten 0 ~ 9 geben Treibstoffmenge/Zeiteinheit an |
|             |                   | H: 500 S: -50 F: 200 C=0 |                                                   |
| 2           | 9                 | H: 452 S: -46 F: 191 C=9 |                                                   |
| 3           | 9                 | Wiederholung             |                                                   |
| ⋮           | ⋮                 | ⋮                        |                                                   |
|             | falls erfolgreich | ERFOLG !!                |                                                   |
|             |                   | VERBLEIB: F = 15         |                                                   |
|             | falls erfolglos   | ENDE !!                  |                                                   |
|             |                   | NEUES SPIEL (J/N)?       | Eingabe, ob ein neues Spiel gewünscht wird        |
| }           | J                 |                          | Neues Spiel                                       |
|             | N                 | >                        | Ende                                              |

# FLUSSDIAGRAMM



## PROGRAMMLISTING

```

10:"A": WAIT 50: CLEAR
 : USING :S=-50:A=0:D
 $=""
20:BEEP 3: PRINT " ***
 START ***"
30:DATA "ZEIT=",50,"TRE
 IBSTOFF=",200,"HOEHE
 =",500
40:RESTORE
50:READ B$,W,B$,F,B$,H
60:WAIT W
70:PRINT USING "####";"
 H:";H;" S:";S;" F:";
 F;" C:"; STR$ C
80:IF F<=0 GOTO 170
90:BEEP 1:D$= INKEY$
100:IF D$="" LET C=A:
 GOTO 130
110:C= VAL D$
120:A=C
130:IF C>F LET C=F
140:F=F-C:X=C-5:H=H+S+X/
 2:S=S+X
150:IF H>0 GOTO 70
160:IF (ABS H<5)+(ABS
 S<5)=2 BEEP 5: PRINT
 "ERFOLG !!": GOTO 18
 0
170:BEEP 3: PRINT "ENDE
 !!": GOTO 190
180:WAIT 150: PRINT
 USING "####";"VERBLE
 IB :F=";F
190:WAIT 50: PRINT "NEUE
 S SPIEL (J/N) ?":Z$=
 INKEY$
200:IF (Z$="J")+(Z$="N")
 <>1 GOTO 190
210:IF Z$="J" GOTO 10
220:END

```

444 Bytes

## SPEICHERINHALT

|     |                                             |
|-----|---------------------------------------------|
| A   |                                             |
| B   |                                             |
| C   | Verbraucher Treibstoff                      |
| D\$ | Verbraucher Treibstoff                      |
| E   |                                             |
| F   | Anfangs Treibstoffmenge/<br>Treibstoffmenge |
| G   |                                             |
| H   | Anfangshöhe Höhe                            |
| I   |                                             |
| J   |                                             |
| K   |                                             |
| L   |                                             |
| M   |                                             |
| N   |                                             |
| O   |                                             |
| P   |                                             |
| Q   |                                             |
| R   |                                             |
| S   | Geschwindigkeit                             |
| T   |                                             |
| U   |                                             |
| V   |                                             |
| W   | Zeit                                        |
| X   |                                             |
| Y   |                                             |
| Z   |                                             |

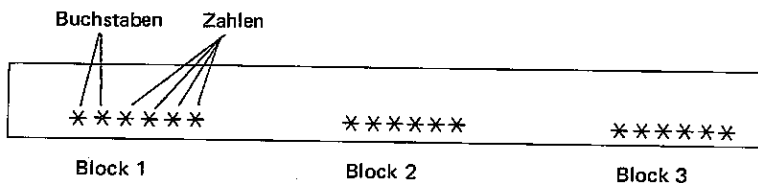
## ÜBERBLICK

3 Zeilen mit 18 Zahlen werden ca. 5 sec. lang angezeigt.

Sie müssen nachdem die Anzeige verschwunden ist, die gerade gesehenen Zeichen eingeben.

## INHALT (Formeln)

Z.B. wird eine solche Zeichenfolge ca. 5 sec. lang angezeigt. Es handelt sich um 2 Buchstaben und 4 Zahlen.



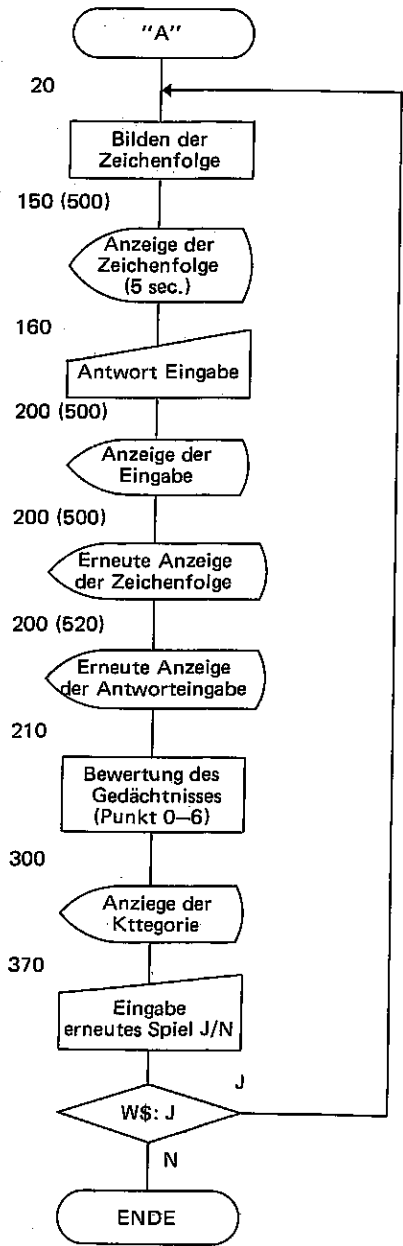
- \* Die 3 Blocks müssen im Gedächtnis behalten und dann als Antwort eingegeben werden.
- \* Der Computer analysiert Ihre Antwort und wirft eine der 7 möglichen Auswertungen aus.  
Jeder Block ist in 2 Teile geteilt à 3 Zeichen. Sind alle Antworten richtig ergibt es eine Punktzahl von 6.

| Punkte | Ergebnis    |
|--------|-------------|
| 0      | Idiot       |
| 1      | Schlecht    |
| 2      | Mittelmäßig |
| 3      | ok          |
| 4      | gut         |
| 5      | intelligent |
| 6      | genial      |

# TASTENBETÄTIGUNG

| Schritt-Nr. | Eingabe                                                                                                        | Anzeige                                             | Bemerkungen                                                                                          |
|-------------|----------------------------------------------------------------------------------------------------------------|-----------------------------------------------------|------------------------------------------------------------------------------------------------------|
| 1           | <input type="button" value="DEF"/> <input type="button" value="A"/>                                            | GEDAECHT                                            | Titel                                                                                                |
| 2           |                                                                                                                | AB1234 PK4398 VC7216<br>1            2            3 | Anzeige Zeichenfolge (5 sec.)<br><input type="radio"/> Buchstaben<br><input type="checkbox"/> Zahlen |
| 3           |                                                                                                                | ANTW. = _                                           | Eingabe 1                                                                                            |
| 4           | AB1234 <input type="button" value="ENTER"/>                                                                    | ANTW. = _                                           | Eingabe 2                                                                                            |
| 5           | PK4398 <input type="button" value="ENTER"/>                                                                    | ANTW. = _                                           | Eingabe 3                                                                                            |
| 6           | VC7216 <input type="button" value="ENTER"/>                                                                    | AB1234 PK4398 VC7216                                | Anzeige Zeichenfolge (1,5 sec.)                                                                      |
|             |                                                                                                                | AB1234 PK4398 VC7216                                | Anzeige Antwort                                                                                      |
|             |                                                                                                                | IDIOT                                               | Anzeige der Kategorie                                                                                |
|             |                                                                                                                | SCHLECHT                                            |                                                                                                      |
|             |                                                                                                                | MITTELMAESSIG                                       |                                                                                                      |
|             |                                                                                                                | OK                                                  |                                                                                                      |
|             |                                                                                                                | GUT!                                                |                                                                                                      |
|             |                                                                                                                | * INTELLIGENT *                                     |                                                                                                      |
|             |                                                                                                                | ** GENIAL **                                        |                                                                                                      |
|             |                                                                                                                | NEUES SPIEL (J/N) ? _                               | Neues Spiel?                                                                                         |
| 7           | <input type="button" value="J"/> oder <input type="button" value="N"/><br><input type="button" value="ENTER"/> |                                                     | Wenn J, GOTO Schritt 2                                                                               |
|             |                                                                                                                | >                                                   | Wenn N, END                                                                                          |

# FLUSSDIAGRAMM



# PROGRAMMLISTING

```

10:"A": USING : WAIT 20
 : PRINT "GEDAECHTNI
 STEST": CLEAR :
 RANDOM
20: DIM G$(6)*1,N$(10)*1
 ,V$(3)*3,X$(3)*6,Z$(
 3)*3,Y$(3)*6
30: FOR I=1 TO 9:N$(I)=
 STR$ I: NEXT I:N$(10
)="0"
50: FOR I=1 TO 6
60: J= RND 26:J=J+64
70: G$(I)= CHR$(J):
 NEXT I
80: FOR I=1 TO 3
90: Y$(I)=" "
100: FOR J=1 TO 3:K= RND
 9
110: Y$(I)=Y$(I)+N$(K):
 NEXT J
120: L= RND 9:J=(I-1)*2+1
130: A$(I)=G$(J)+G$(J+1)+
 N$(L)
140: H$=Y$(I):A$(I+3)=
 RIGHT$(H$,3): NEXT
 I
150: GOSUB 500
160: FOR I=1 TO 3
170: INPUT " ANTW. = ";X$(
 I):X$(I)= LEFT$(X$(
 I),6)
180: Z$(I)= LEFT$(X$(I),
 3)
190: V$(I)= RIGHT$(X$(I)
 ,3): NEXT I
200: GOSUB 520: GOSUB 500
 : GOSUB 520
210: N=0
220: FOR I=1 TO 3
230: IF A$(I)=Z$(I) LET N
 =N+1
240: IF A$(I+3)=V$(I) LET
 N=N+1
250: NEXT I
260: N=N+1
270: WAIT 150: ON N G
 300,310,320,330,:
 350,360
300: BEEP 1: PRINT "
 OT": GOTO 370
310: BEEP 1: PRINT "
 LECHT": GOTO 370
320: BEEP 2: PRINT "
 TELMAESSIG": GOTT
 0
330: BEEP 2: PRINT "
 ": GOTO 370
340: BEEP 3: PRINT "
 T I": GOTO 370
350: BEEP 4: PRINT "*
 ELLIGENT *": GOTO
 0
360: BEEP 5: PRINT "**
 IAL**"
370: W$="": BEEP 1: IN
 "NEUES SPIEL (J/N
 ";W$
380: IF W$="N" THEN 600
390: IF W$="J" THEN 50
395: GOTO 370
400: GOTO 370
500: WAIT 300: BEEP 2:
 PRINT A$(1);A$(4):
 ";A$(2);A$(5):"
 ";A$(3);A$(6)
510: RETURN
520: WAIT 80: BEEP 1:
 PRINT USING "#####
 ;X$(1); USING ;"
 ; USING "#####";X
 2); USING ;" ";
 USING "#####";X$(
525: USING
530: RETURN
600: END

```

908 Bytes

# SPEICHERINHALT

|        |                                 |
|--------|---------------------------------|
| A\$    |                                 |
| B\$    |                                 |
| C\$    | 3 Spalten Buchstaben DIM A\$(6) |
| D\$    |                                 |
| E\$    |                                 |
| F\$    |                                 |
| G      |                                 |
| H\$    | ✓                               |
| I      | Index                           |
| J      | Zufallszahlengenerator          |
| K      | Zufallszahlengenerator          |
| L      | Zufallszahlengenerator          |
| M      |                                 |
| N      | Zähler                          |
| O      |                                 |
| P      |                                 |
| Q      |                                 |
| R      |                                 |
| S      |                                 |
| T      |                                 |
| U      |                                 |
| V      |                                 |
| W\$    | Eingabe neues Spiel             |
| X      |                                 |
| Y      |                                 |
| Z      |                                 |
| G\$(6) | Buchstaben (1 ~ 6)              |
| N\$(6) | Arbeit (1 ~ 6)                  |
| V\$(3) | 3 Spalten nach der Antwort      |
| X\$(3) | Arbeit (1 ~ 3)                  |
| Y\$(3) | Arbeit (1 ~ 3)                  |
| Z\$(3) | 3 Spalten vor der Arbeit        |



## ÜBERBLICK

In diesem Spiel muß ein Mann einen Käfer jagen.

## INHALT

Der Käfer bewegt sich durch einen Zufallszahlengenerator. Der Mann jagt den Käfer und tötet ihn.

Der Mann bewegt sich über die Tasten 

|         |
|---------|
| 8       |
| 4 ← ↔ 6 |
| 2       |

 (INKEY\$).

Jedesmal, wenn der Mann sich um einen Schritt bewegt, tut der Käfer das auch (manchmal bleibt der Käfer auch stehen).

Die Ausgangsposition des Mannes ist (0, 0).

Die Ausgangsposition des Käfers wird über einen Zufallszahlengenerator gewählt. Hinweise werden angezeigt.

Der Abstand wird über die Gleichung  $ABS(X-A) + ABS(Y-B)$  berechnet und angezeigt. Die Ausgangsenergie ist 100. Sie nimmt bei jedem Schritt um 1 ab.

Jedesmal, wenn ein Käfer getötet wird, steigt die Energie um 5,10 oder 15 (der Betrag wird über einen Zufallszahlengenerator gewählt).

Die Punktzahl wird nach der Zahl der getöteten Käfer bei Energie = 0 berechnet.

Das Programm kann über RUN 

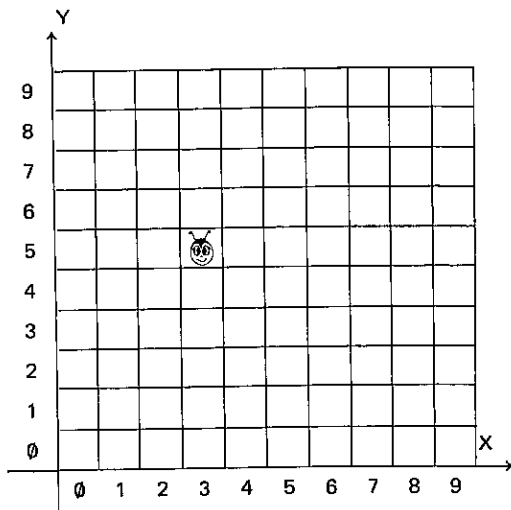
|       |
|-------|
| ENTER |
|-------|

 oder 

|     |
|-----|
| DEF |
|-----|

|   |
|---|
| A |
|---|

 gestartet werden.



Position des Mannes (X, Y)

Position des Käfers (a, b)

Zur Anzeige: Kleinbuchstaben geben aktuelle Werte an.

|        |                     |       |
|--------|---------------------|-------|
| (x, y) | ABSTAND = $\varrho$ | E = e |
|--------|---------------------|-------|

Ausgangsposition  
(X-Koordinate/Y-Koordinate)

Hinweis  
(Abstand)

Verbleibende Energie

- Jedesmal, wenn der Mann sich bewegt, ändert sich die Anzeige.

Der Käfer ist gefangen.

|       |                    |
|-------|--------------------|
|       | TREFFER!           |
| Pause | BANG! BANG!        |
|       | PUNKTE t ENERGIE e |

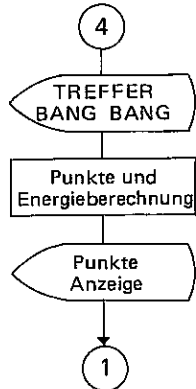
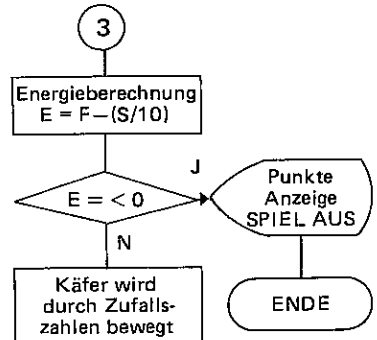
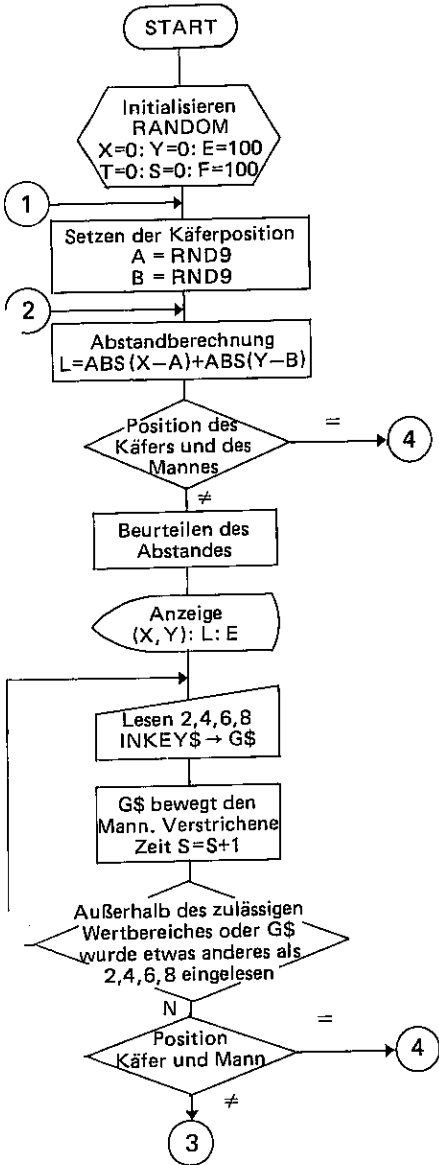
**BEEP:**

- ★ Hinweis: Wenn der Abstand 1 ist, ertönt ein akustisches Signal 3 mal; ist der Abstand 2, 2 mal und ist der Abstand 3, 1 mal.
- ★ Ist der Abstand größer als 3, ertönt kein Signal.
- ★ Ist der Käfer gefangen, gibt der Computer 5 Signale.

## TASTENBETÄTIGUNG

| Schritt-Nr. | Eingabe | Anzeige               | Bemerkungen |
|-------------|---------|-----------------------|-------------|
| 1           | DEF A   | (0,0) ABSTAND=5 E=100 |             |
|             | 8       | (0,1) ABSTAND=4 E=99  |             |
|             | 6       | (1,1) ABSTAND=2 E=98  | BEEP 2      |
|             | 8       | TREFFER!              | BEEP 5      |
|             |         | BANG! BANG!           |             |
|             |         | PUNKTE 1 ENERGIE 108  |             |

# FLUSSDIAGRAMM



# PROGRAMMLISTING

```

10:"A": RANDOM : WAIT 2
50: PRINT "*** KAEFER
 JAGD ***": BEEP 3
20:X=0:Y=0:E=100:F=100:
 T=0:S=0
30:A= RND 9:B= RND 9
40:L= ABS (X-A)+ ABS (Y
 -B)
50:IF X=A AND Y=B GOTO
 400
100:IF L=1 BEEP 3
110:IF L=2 BEEP 2
120:IF L=3 BEEP 1
130:WAIT 50: PRINT "(";
 STR$(X);", "; STR$(
 Y);") ABSTAND=";
 STR$(L);" E="; STR$(
 E)
150:S=S+1:E=F- INT (S/2)
153:IF E<=0 THEN 500
155:G$= INKEY$: IF G$="
 " GOTO 130
157:BEEP 1
160:IF G$="2" LET Y=Y-1:
 GOTO 210
170:IF G$="4" LET X=X-1:
 GOTO 210
180:IF G$="6" LET X=X+1:
 GOTO 210
190:IF G$="8" LET Y=Y+1:
 GOTO 210
200:GOTO 150
210:IF X<0 LET X=0: GOTO
 150
220:IF Y<0 LET Y=0: GOTO
 150
230:IF X>9 LET X=9: GOTO
 150
240:IF Y>9 LET Y=9: GOTO
 150
250:IF X=A AND Y=B GOTO
 400
260:E=F- INT (S/2)
270:IF E<=0 GOTO 500
280:R= RND 5
290:IF R=1 LET B=B-1:
 GOTO 340
300:IF R=2 LET A=A-1:
 GOTO 340
310:IF R=3 LET A=A+1:
 GOTO 340
320:IF R=4 LET B=B+1:
 GOTO 340
340:IF A<0 OR A>9 GOTO 3
 70
350:IF B<0 OR B>9 GOTO 3
 70
360:GOTO 40
370:BEEP 4: PAUSE "*** V
 ORSICHT ***": GOTO 3
 0
400:PAUSE "TREFFER !"
410:BEEP 5
420:PAUSE "BANG! BANG!"
430:T=T+1:C= RND 3*5:F=F
 +C
435:E=F- INT (S/2)
440:WAIT 100: PRINT "PUN
 KTE ";T;" ENERGIE ";
 E
450:GOTO 30
500:WAIT : PRINT "PUNKTE
 "; STR$(T);" *SPIE
 L AUS!!"
510:END

```

735 Bytes

# SPEICHERINHALT

|     |                               |
|-----|-------------------------------|
| A   | X-Koordinate                  |
| B   | Y-Koordinate                  |
| C   | Energiezähler                 |
| D   |                               |
| E   | Verbleibende Energie          |
| F   | Energie                       |
| G\$ | Tasteneingabe 2, 4, 6, 8      |
| H   |                               |
| I   |                               |
| J   |                               |
| K   |                               |
| L   | Abstand Käfer/Mann            |
| M   |                               |
| N   |                               |
| O   |                               |
| P   |                               |
| Q   |                               |
| R   | Größe der Bewegung des Käfers |
| S   | Verstrichene Zeit             |
| T   | Punkte                        |
| U   |                               |
| V   |                               |
| W   |                               |
| X   | Position Mann X-Koordinate    |
| Y   | Position Mann Y-Koordinate    |
| Z   |                               |

## ÜBERBLICK

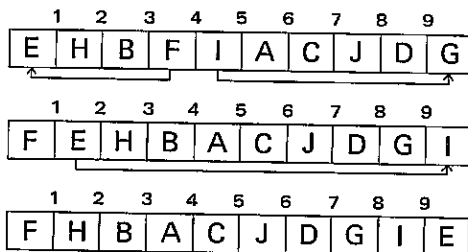
Ein Denkspiel, wobei willkürlich ausgelegte Buchstaben (A bis J) geordnet werden müssen. Schaffen Sie's beim erstenmal? Vielleicht nicht. Versuchen Sie es trotzdem.

## ANLEITUNG UND BETRIEB

**DEF** **A** "Doppelte Rotation" wird angezeigt. Daraufhin erscheinen die Buchstaben A, B, C . . . J in willkürlicher Reihenfolge in der Anzeige. Mit der Eingabe von Fixpunkten (1 bis 9) werden die angezeigten Buchstaben "rotiert". Ihre Punktzahl wird durch die Häufigkeit der Tastenbetätigungen bestimmt. Je weniger es sind, desto besser.

## BEISPIEL

Wenn z.B. der Fixpunkt 4 in diesen Zeichen-String eingegeben wird, werden die Buchstaben wie links im Bild dargestellt rotiert. Die nächste Darstellung erhält man durch Eintasten von Fixpunkt 1.

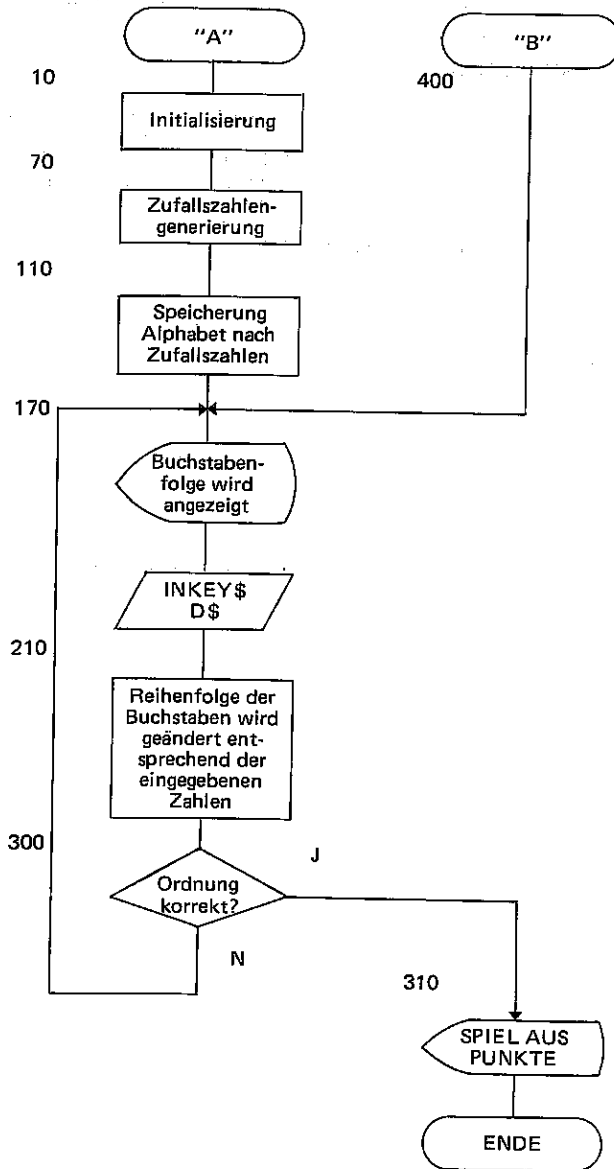


Versuchen Sie einmal auf diese Weise, das Ordnen möglichst effektiv zu gestalten.

# TASTENBETÄTIGUNG

| Schritt-Nr. | Eingabe                                                             | Anzeige                                                                 | Bemerkungen                                                      |
|-------------|---------------------------------------------------------------------|-------------------------------------------------------------------------|------------------------------------------------------------------|
| 1           | <input type="button" value="DEF"/> <input type="button" value="A"/> | DOPPELTE ROTATION                                                       |                                                                  |
|             |                                                                     | A ~ J                                                                   | Anzeige Zufallszeichenfolge                                      |
| 2           | <input type="button" value="1"/> ~ <input type="button" value="9"/> |                                                                         | Zahlen zwischen 1 und 9 werden gewählt und eingegeben.           |
|             |                                                                     | Eingabe Wiederholung                                                    |                                                                  |
|             |                                                                     |                                                                         |                                                                  |
|             |                                                                     | ABCDEFGHIJ                                                              |                                                                  |
|             |                                                                     | SPIEL AUS                                                               |                                                                  |
|             |                                                                     | PUNKTE 35                                                               |                                                                  |
|             |                                                                     | >                                                                       |                                                                  |
| 1           | <input type="button" value="DEF"/> <input type="button" value="B"/> | A ~ J                                                                   | Soll das neue Spiel mit der selben Zufallszeichenfolge beginnen? |
|             |                                                                     | Wie <input type="button" value="DEF"/> <input type="button" value="A"/> |                                                                  |

# FLUSSDIAGRAMM





## PROGRAMMLISTING

```

10:"A": CLEAR : WAIT 50
 : RANDOM : DIM B$(4)
20:PAUSE "DOPPELTE ROTA
 TIO~"
30:B$(0)="ABCDEFGHJIJ"
40:B$(1)=""
50:A=0
60:FOR I=1 TO 10
70:R= RND 10
80:S=2^(R-1)
85:B=S AND A
90:IF B<>0 GOTO 70
100:A=A OR S
110:B$(1)=B$(1)+ MID$(B
 $(0),R,1): NEXT I
120:B$(2)=B$(1)
130:N=0
150:BEEP 1
170:D$="": PRINT B$(2):D
 $= INKEY$
180:C= VAL D$
190:IF C=0 GOTO 170
210:B$(3)= LEFT$(B$(2),
 C)
220:B$(4)= RIGHT$(B$(2)
 ,10-C)
240:IF C=1 GOTO 260
250:B$(3)= RIGHT$(B$(3)
 ,1)+ LEFT$(B$(3),C-
 1)
260:IF C=9 GOTO 280
270:B$(4)= RIGHT$(B$(4)
 ,9-C)+ LEFT$(B$(4),
 1)
280:B$(2)=B$(3)+B$(4)
290:N=N+1
300:IF B$(2)<>B$(0) GOTO
 150
310:BEEP 5: PAUSE "SPIEL
 AUS"
320:WAIT 200: PRINT
 USING "####";"PUNKTE
 ";N
330:END
400:"B": WAIT 50: GOTO 1
 20
410:END

```

475 Bytes

## SPEICHERINHALT

|        |                |
|--------|----------------|
| A      | ✓              |
| B      | ✓              |
| C      | ✓              |
| D\$    | Eingabe Taste  |
| E      |                |
| F      |                |
| G      |                |
| H      |                |
| I      | ✓              |
| J      |                |
| K      |                |
| L      |                |
| M      |                |
| N      | Punkte         |
| O      |                |
| P      |                |
| Q      |                |
| R      | Zufallszahlen  |
| S      | ✓              |
| T      |                |
| U      |                |
| V      |                |
| W      |                |
| X      |                |
| Y      |                |
| Z      |                |
| B\$(4) | Alphabet Folge |





**SHARP CORPORATION**

**OSAKA, JAPAN**

1982 © SHARP CORPORATION  
Printed in Japan/Imprimé au Japon  
2M5T(T1NSG3742CCZZ)②