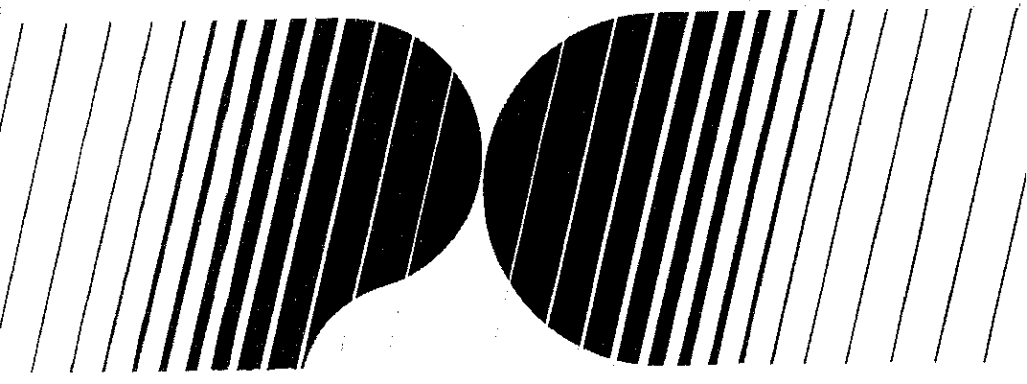
SHARP®

TASCHENCOMPUTER

MODELL

PC-1430

BEDIENUNGSANLEITUNG



SHARP PC-1430

BEDIENUNGSANLEITUNG

Lieber Leser !

Wir haben uns bemüht, dieses Handbuch ohne Fehler zu erstellen. Doch auch bei der sorgfältigsten Prüfung kann noch etwas übersehen werden. Wenn Sie daher Fehler finden und/oder den einen oder anderen Verbesserungsvorschlag zu diesem Handbuch haben, so teilen Sie uns dies bitte mit.

Vielen Dank im voraus

SHARP ELECTRONICS
HAMBURG

EINLEITUNG

I Allgemeines 7
II Betriebshinweise 8
III Stromversorgung 10
IV Tastaturabdeckung 12

TEIL I – THEORIE

1.1 GRUNDLAGEN 15
1.1.1 Einschalten des Computers 15
1.1.2 Ausschalten des Computers 15
1.1.2.1 Automatisch 15
1.1.2.2 Manuell 15
1.1.3 Bedienelemente 16
1.1.4 Tastenfunktionen 17
1.1.4.1 Eingabe 20
1.1.4.2 Notation 21
1.1.5 Die Anzeige 22
1.1.6 Wahl der Betriebsart 23

1.2 GRUNDLAGEN SHARP CE-126P 24
1.2.1 Allgemeines 24
1.2.2 Verwendung als Drucker 25
1.2.2.1 Manueller Druckbetrieb (Protokollerstellung) 25
1.2.2.2 Programmgesteuerter Ausdruck 26
1.2.3 Verwendung des integrierten Kassetten-Interfaces 27
1.2.3.1 Auswahl des Kassettenrekorders 27
1.2.3.2 Anschluß des Kassettenrekorders an das CE-126P 28
1.2.4 Datenspeicherung auf Magnetband 28
1.2.4.1 Allgemeines 28
1.2.4.2 Speichern von Daten und Programmen 29
1.2.4.3 Überprüfen der Abspeicherung 30
1.2.4.4 Laden von Programmen oder Daten 30

TEIL II – PRAXIS

2.1 EINSATZ ALS MANUELLER RECHNER 33
2.1.1 Betriebshinweise 33
2.1.1.1 Anzeige/Display-Format und Symbole 33
2.1.2 Normale Berechnungen 34
2.1.3 Eingabe-Korrektur 36

2.1.4 Statistische Berechnungen 38
2.1.5 Rechenbereich 42
2.1.6 Wissenschaftliche Berechnungen 44
2.1.7 Anzeigeformat und Rundung 47
2.1.7.1 Anzeigeformat 47
2.1.7.2 Rundungsfunkion 48

2.2 EINSATZ ALS BASIC-RECHNER 50

2.2.1 Rechnen ohne Programmunterstützung (RUN-Mode) 50
2.2.1.1 Grundrechnungsarten 50
2.2.1.3 Wissenschaftliche Schreibweise 52
2.2.1.4 Editier-/Korrekturmöglichkeit 53
2.2.1.5 Mathematische Funktionen/Klammerregeln 54
2.2.1.6 Textausdrücke 55
2.2.1.7 Logische Vergleichsausdrücke 56
2.2.2 Sprachelemente 58
2.2.2.1 Numerische Konstante 58
2.2.2.2 Textkonstante 58
2.2.2.3 Numerische Variable 58
2.2.2.4 Textvariable 59
2.2.2.5 Numerische Funktionen, Textfunktionen 59
2.2.2.6 Numerischer Ausdrücke 60
2.2.2.7 Textausdrücke 60
2.2.2.8 Logische Vergleichsausdrücke 61
2.2.3 Variable 63
2.2.3.1 Standartvariable 63
2.2.3.2 Einfache Variable 65
2.2.3.3 Indizierte Variable 65
2.2.3.4 Besonderheiten der Variablen A 66
2.2.3.5 Feldvariable 67
2.2.3.6 Numerische Feldvariable 68
2.2.3.7 Textfeldvariable 69
2.2.4 Arithmetische Funktionen 71
2.2.5 Textfunktionen 72
2.2.6 Programmieren in BASIC 73
2.2.6.1 BASIC-Übersicht 73
2.2.6.2 Programmerstellung 74
2.2.6.3 Programmaufbau 75
2.2.6.4 Eingabe eines Programms 75
2.2.6.5 Korrektur einer Zeile 76
2.2.6.6 Löschen einer Zeile 78
2.2.6.7 Programmausführung 78
2.2.6.8 Fehlermeldungen/Fehlersuche 79

2.2.7 BASIC-Befehlsvorrat 80
2.2.7.1 Fest vorprogrammierte Schlüsselwörter und arithmetische Funktionen 80

ANHANG

A. TASTENFUNKTIONEN DES PC-1430 170
B. LISTE DER FUNKTIONEN UND ANWEISUNGEN 172
C. BASIC — BEFEHLSÜBERSICHT (Kurzfassung) 173
D. STANDARDVARIABLE 182
E. OPERATOREN 183
E.1 Arithmetische Operatoren 183
E.2 Textfunktionsoperatoren 183
E.3 Logische Vergleichsoperatoren 183
E.4 Reihenfolge der Operatoren 183
F. RECHENBEREICH 184
G. ABGELEITETE MATHEMATISCHE FUNKTIONEN 185
H. PROGRAMM-HAUPTSPEICHER-BEGRENZUNG 186
I. FEHLERMELDUNGEN 187
J. REFERENZLISTE 188
K. TECHNISCHE DATEN PC-1430 189
L. TECHNISCHE DATEN CE-126P 190
PROGRAMMBEISPIELE 191
M. INDEX 219

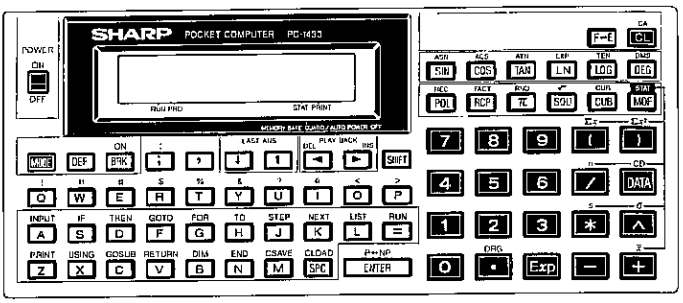
E I N L E I T U N G

E I N L E I T U N G

I ALLGEMEINES

Mit dem **PC-1430** stellt SHARP einen preisgünstigen und dennoch außerordentlich leistungsstarken BASIC-Taschencomputer mit integriertem wissenschaftlichem Rechner vor.

Erweitertes BASIC, etwa 17,4k-ROM-Betriebssystem und 2,0k-RAM-Bereich für Anwenderprogramme sowie 46 vorprogrammierte mathematisch wissenschaftliche Funktionen lassen diesen Rechner auch gehobenen Ansprüchen aus fast allen Bereichen gerecht werden.



Frontansicht des **PC-1430**

Zusammen mit dem Thermodrucker mit integriertem Kassetten-Interface **CE-126P** wird der **PC-1430** zu einer kleinen, leistungsfähigen Datenverarbeitungsanlage.

Bei der Gestaltung dieser Bedienungsanleitung wurde versucht, sowohl den Ansprüchen eines Anfängers als auch denen eines geübten Programmierers gerecht zu werden. In manchen Fällen war eine Kompromißlösung aber nicht zu umgehen, und wir verweisen daher bereits an dieser Stelle auf die umfangreiche BASIC-Literatur, die im Fachhandel erhältlich ist. Eine Referenzliste mit genauen Bestellangaben finden Sie im Anhang dieses Handbuchs.

Wir hoffen, Ihnen mit diesem Handbuch alle möglichen Hilfsmittel in die Hand zu geben, so daß Sie Ihren **PC-1430** für Ihre Belange optimal einsetzen können.

II BETRIEBSHINWEISE

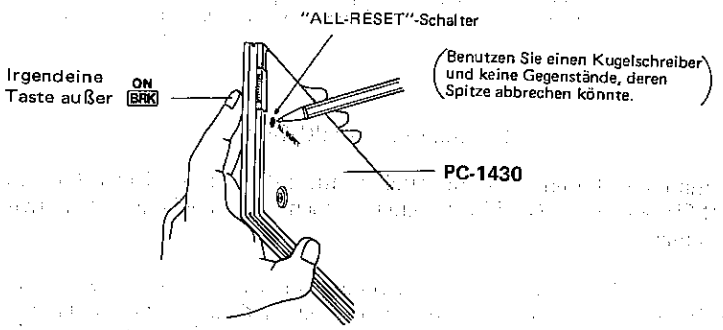
1. Die Flüssigkeitskristallanzeige des **Computers** ist in Spezialglas eingebettet und kann daher bei Gewalteinwirkung zerbrechen. Behandeln Sie den Computer deshalb mit Sorgfalt. Beim Transport immer die Tastaturabdeckung verwenden.

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-126P** eingeschaltet) kann der **Computer** "abstürzen". Dadurch werden alle Tastenfunktionen einschließlich der CL- oder CE-Taste blockiert.
   Sollte Ihnen dies passieren, müssen Sie den Computer initialisieren. Dazu stehen zwei Möglichkeiten zur Auswahl:

   1) Drücken Sie irgendeine Taste außer ON BRK und betätigen Sie den "ALL RESET"-Schalter an der Rückseite des Geräts. Bei dieser Art der Initialisierung werden die Programme, Variablen und der Reservespeicher nicht gelöscht.



Initialisieren des **Computers**

HINWEIS: Nicht die Taste ON BRK gedrückt halten und dann den ALL RESET Schalter drücken, weil dadurch Programme und Daten gelöscht werden könnten.

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

   3) Sollte dann das Problem immer noch vorhanden sein, entnehmen Sie die alten Batterien und setzen innerhalb von 10 Sekunden neue Batterien ein und wiederholen dann Schritt 2).

5. Wenn eine Reparatur des Computers erforderlich sein sollte, wenden Sie sich an das Geschäft, wo der Computer gekauft wurde.
   Reparaturarbeiten sollten ausschließlich von autorisierten SHARP-Servicestellen durchgeführt werden.

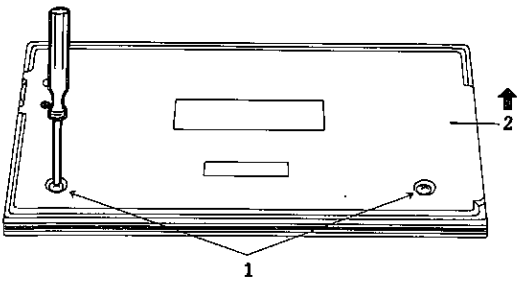
III. STROMVERSORGUNG

Der Taschencomputer **PC-1430** wird mit Lithiumbatterien betrieben. Die Batterien sind werksseitig bereits eingesetzt. Bei erschöpften Batterien muß ein Wechsel wie folgt vorgenommen werden:

1. Den Computer ausschalten.

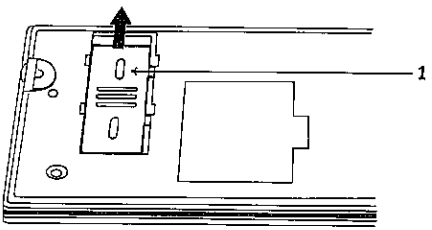
2. Die zwei Gehäuseschrauben (1) an der Rückseite des Geräts entfernen.

3. Den Gehäusedeckel (2) auf der Schraubenseite etwas anheben und in Richtung des Pfeils schieben und abheben.



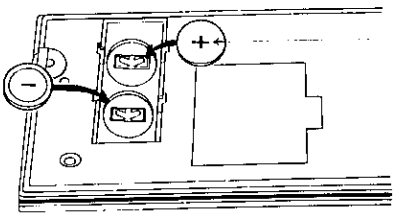
Entfernen des Gehäusedeckels

4. Die Batterieabdeckung (1) in Richtung des Pfeils schieben und dem Rechner entnehmen.



Abnehmen der Batterieabdeckung

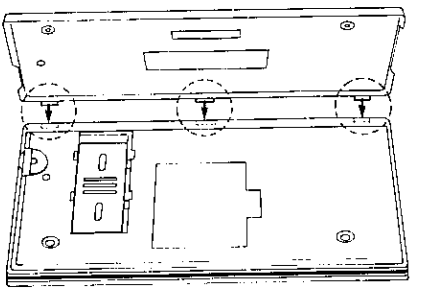
5. Die erschöpften Batterien (1) durch neue ersetzen.
   (Batterien nur paarweise wechseln)



Batteriewechsel

6. Die Batterieabdeckung wieder einsetzen und verriegeln.

7. Den Gehäusedeckel wie dargestellt auf den Rechner aufsetzen und verschrauben.



Aufsetzen des Gehäusedeckels

Folgende Punkte sind beim Batteriewechsel besonders zu beachten:

- Wechseln Sie die Batterien immer paarweise.
- Achten Sie darauf, daß Sie beim Batteriewechsel die alten Batterien nicht mit den neuen vertauschen.
- Verwenden Sie nur den angegebenen Batterietyp.
  Batterietyp: CR-2032 oder Ersatztype

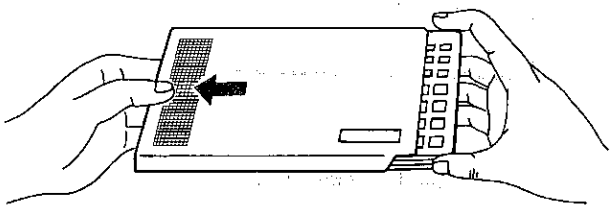
Die Batterien von Kindern fernhalten.

IV. TASTATURABDECKUNG

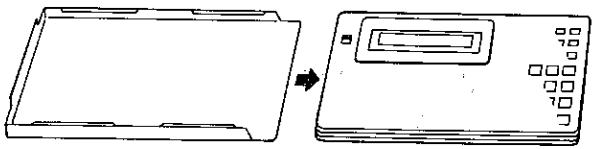
Zum Schutz der Tastatur und der Anzeige ist der **Computer** mit einer stabilen, doppelseitig aufschiebbaren Tastaturabdeckung ausgestattet.

Die Tastaturabdeckung des **Computers** wie dargestellt verwenden.

Bei Gebrauch:

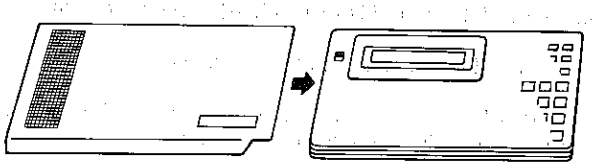


Die Tastaturabdeckung ist so gestaltet, daß sie während der Verwendung des Rechners auf dessen Unterseite geschoben werden kann.



Tastaturabdeckung bei Gebrauch des Rechners

Nach Gebrauch:



Tastaturabdeckung nach Gebrauch des Rechners

T E I L I

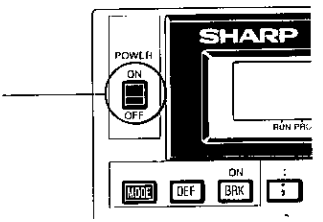
T H E O R I E

TEIL I THEORIE

1.1 GRUNDLAGEN

1.1.1 EINSCHALTEN DES COMPUTERS

Das Einschalten erfolgt zum einen über den Schiebeschalter links neben dem Display, zum anderen über die Taste ON BRK .



Nach dem Einschalten erscheinen auf der Anzeige folgende Symbole:

Zeigt die Winkeleinheit (DEG/GRAD/RAD) an.

Zeigt Betriebsart (RUN/PRO) an.

1.1.2 AUSSCHALTEN DES COMPUTERS

1.1.2.1 Automatisch

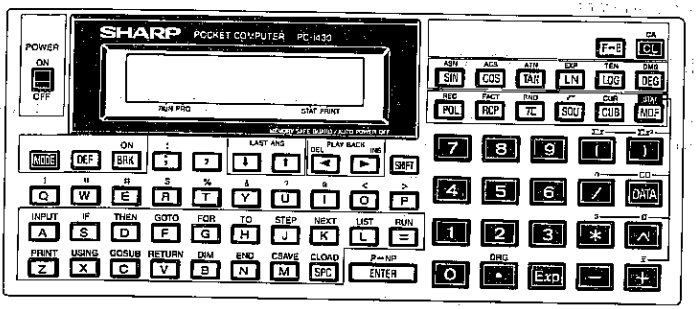
Der Computer schaltet sich nach ca. 11 Minuten nach der letzten Tastenbedienung automatisch ab. Dabei bleiben alle Programm- und Dateninformationen gespeichert.

Hat sich der Rechner automatisch ausgeschaltet, muß er über die Taste ON BRK wieder aktiviert werden. Auf der Anzeige erscheint die zuletzt bearbeitete oder angezeigte Zeile.

1.1.2.2 Manuell

Soll der Computer sofort nach Gebrauch ausgeschaltet werden, so muß der Schiebeschalter links neben dem Display in Stellung OFF gebracht werden. Die Programm- und Dateninformationen bleiben dabei gespeichert. Der Computer läßt sich auch während einer Programmausführung abschalten.

1.1.3 BEDIENELEMENTE



Das Bedienfeld des **Computers** besteht aus 73 Tasten, die zu 4 Blöcken zusammengefaßt sind:

- die Schreibmaschinen-Alphatastatur mit [ENTER]-Taste (2 Ebenen)
- die Tasten für die Sonderfunktionen
- die numerische Tastatur mit den Grundrechnungsarten (2 Ebenen)
- die Tasten für die wissenschaftlichen Funktionen (2 Ebenen)

Die Schreibmaschinen-Alphatastatur ist mit zwei Ebenen belegt. Bei einfachem Tastendruck gelten die auf den Tasten stehenden Bezeichnungen. Will man die über den Tasten stehenden Sonderzeichen (oberste Reihe der Tastatur) oder die vorprogrammierten BASIC-Anweisungen abrufen, muß vorher die Doppelfunktionstaste [SHIFT] gedrückt werden.

Mit der Taste [DEF] kann den unteren beiden Reihen noch eine dritte Ebene zugewiesen werden. Über die sogenannten 'Definable Keys' können verschiedene Programme sofort gestartet werden; man drückt dazu die Taste [DEF] und eine dem Programm zugewiesene Taste.

In der Reihe über der Alpha-Tastatur befinden sich die Sonderfunktionstasten zur Wahl der Betriebsart, zum Unterbrechen oder zum Auflisten sowie zum Editieren eines Programms.

Rechts neben der Alpha-Tastatur befindet sich die numerische Tastatur. Die zweite Ebene der numerischen Tastatur, die hauptsächlich für statistische Berechnungen dient, ist ebenfalls mit der [SHIFT]-Taste erreichbar.

Mit den Tasten oberhalb der numerischen Tastatur lassen sich die meisten wissenschaftlichen Berechnungen, wie trigonometrische Berechnungen, Polarkoordinatenberechnungen usw., durchführen. Dieser Teil der Tastatur ist mit zwei Ebenen belegt.

Die [SHIFT] -Taste ist ein Wechselschalter, d.h. durch einmaliges Drücken wird in den jeweiligen Mode umgeschaltet, durch nochmaliges Drücken der Urzustand wieder hergestellt.

1.1.4 TASTENFUNKTIONEN

[DEF] 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 Programms über einen Markennamen.

[SHIFT] Doppelfunktionstaste

[↓] [↑] Editiertasten

[◀] Cursor nach links; Editieraufruf

[▶] Cursor nach rechts; Editieraufruf

[BRK] Unterbrechen des Programmablaufs. Am Display erscheint die Meldung BREAK IN XXX (XXX = Zeilennummer) Unterbricht auch Drucker- und Cassettenbetrieb.

[CL] Löscht die Anzeige und die Fehlermeldungen.

[SHIFT] [INS ▶] Einfügen einzelner Zeichen (Platzhalter ␣ wird gesetzt)

[SHIFT] [DEL ◀] Löschen einzelner Zeichen

[ON BRK] Einschalten des Rechners nach automatischer Abschaltung

[CA CL] Löschen der Anzeige und
- Löschen des WAIT-Intervalls
- Löschen des USING-Formats
- Aufheben des TRACE-Betriebs
- Löschen der Fehlermeldungen

[A]~[Z] Alphatastatur A-Z; Variablennamen

[SPC] Leerzeichen
Leerzeichen, die nicht in Anführungsstriche (" ") eingeschlossen oder in einer REM-Anweisung enthalten sind, werden in Programmen und manuellen Berechnungen ignoriert.
Die [SPC] Taste kann auch als "defineable Key" verwendet werden.

[ENTER]	Beschließt die Eingabe einer Programmzeile. Beim Rechnen ohne Programmunterstützung im RUN-Modus ersetzt die Taste das "="-Zeichen. Programmstart in Verbindung mit RUN oder GOTO.
[MODE]	Dieser Wechselschalter dient zur Wahl der Betriebsarten RUN und PRO und.
! & [ ]~[ ]	Sonderzeichen
[SHIFT] [A]~ [SHIFT] [SPC]	Aufruf der wichtigsten BASIC-Anweisungen
P ↔ NP [SHIFT] [ENTER]	Umschaltung für Druck/Nichtdruck beim Rechnen ohne Programmunterstützung im RUN/PRO-Modus
[.]	Dezimalpunkt
[0] ~ [9]	numerische Zeichen
[+] ~ [/]	Grundrechnungsarten
[sin]	Trigonometrische Funktionen
[SHIFT] [ASN]	Umkehrfunktionen der trigonometrischen Funktionen
[F↔E]	Umschaltung Fließkomma/Festkomma
[SHIFT] [&]	Umrechnung Hexadezimalsystem - Dezimalsystem
[DEG]	Umrechnung Dezimalsystem - Sexagesimalsystem
[SHIFT] [DMS]	Umrechnung Sexagesimalsystem - Dezimalsystem
[LN]	Natürlicher Logarithmus
[SHIFT] [EXP]	Exponentialfunktion e^x
[LOG]	Zehnerlogarithmus
[SHIFT] [TEN]	Exponentialfunktion 10^x
[RCP]	Reziprokwert
[SHIFT] [REC]	Umwandlung Polarkoordinaten in rechtwinklige Koordinaten
[POL]	Umwandlung rechtwinklige Koordinaten in Polarkoordinaten
[EXP]	Exponentialdarstellung
[π]	Konstante PI

SHIFT RND	Zufallszahlen
MDF	Runden
SHIFT ^	Potenzierfunktion
SHIFT √	Quadratwurzelfunktion
SHIFT CUR	Dritte Wurzel
SQU	Quadratzahlfunktion
CUB	Kubikzahlfunktion
SHIFT FACT	Fakultät
( )	Klammern
SHIFT DRG	Umschaltung der Winkeleinheit
SHIFT > SHIFT <	Logische Vergleichsoperatoren
DATA	Dient zur Eingabe von Daten im STAT-Modus. In anderen Betriebsarten ist diese Taste nicht wirksam.
SHIFT STAT MDF	Die folgenden Funktionen stehen im STAT-Modus zur Verfügung.
Σx :	Summe der Daten
Σx^2 :	Summe der Quadrate der Daten
n :	Häufigkeit
CD :	Zur Korrektur von Daten, die mit DATA eingegeben wurde.
S :	Standardabweichung S
σ :	Standardabweichung
$\bar{x}$:	Mittelwert

1.1.4.1 Eingabe

Geben Sie die im folgenden Beispiel angeführten Buchstaben und Zeichen ein, die dann auf der Anzeige erscheinen müssen.

(Beispiel) [1] [2] [3] 123_

[S] [H] [A] [R] [P] 123SHARP_

Falls die Anzeige nicht "123SHARP" ist, haben Sie eine falsche Taste gedrückt. Drücken Sie die Taste CA zum Löschen der Anzeige und geben die Zeichenfolge erneut ein.

Geben Sie als nächstes "(?)" ein. Zur Eingabe dieser und der anderen hellbraunen Symbole und Funktionen, die oberhalb der Tasten angegeben sind, wird zuerst die Taste [SHIFT] und dann die gewünschte Taste gedrückt.

Geben Sie jetzt, unter Verwendung der Taste [O] [U] [P], die Zeichenfolge "123SHARP(?)" ein.

[SHIFT] [O] [SHIFT] [U] [SHIFT] [P] 123SHARP(?)_

Funktionen und BASIC-Kommandos werden auf die gleiche Weise eingegeben: Für die Kommandos, die in Hellbraun oberhalb der Tasten angegeben sind, wird zuerst die Taste [SHIFT] gedrückt.

[SIN] 123SHARP(?)SIN_

[SHIFT] [Z] RP(?)SINPRINT_

Bei der Eingabe von PRINT ist "123SHA" von der Anzeige verschwunden. Diese Zeichen können zurückgerufen werden, indem der Cursor mit der Taste [DEL ◄] nach links bewegt wird.

[DEL ◄] ARP(?)SINPRINT

[DEL ◄] 123SHARP(?)SINP

Wenn das Symbol □ die [S] erreicht, die Taste [DEL ◄] gedrückt halten. Diese Taste hat eine Wiederholfunktion und bewegt sich dann fortlaufend zur linken Seite der Zeichenfolge. Dabei blinkt das Symbol □.

123SHARP(?)SINP

Drücken Sie jetzt die Taste INS ▶. Das Symbol bewegt sich jetzt zur rechten Seite der Anzeige, wodurch bereits verschwundene Zeichen wieder auf die Anzeige zurückgeholt werden.

RP(?)SINPRINT_

Zusammengefaßt dienen die Tasten DEL ◀ und INS ▶ zum Zurückholen von Informationen auf die Anzeige. Die Symbole [ ] und _ werden als Cursor bezeichnet und geben die Position an, wo Daten eingegeben werden können. Wenn Sie versehentlich eine falsche Taste drücken, kann der Cursor mit der Taste DEL ◀ zurückbewegt und das richtige Zeichen eingegeben werden. (Weitere Einzelheiten siehe auf Seite 36.)

1.1.4.2 Notation

Ab jetzt werden in dieser Bedienungsanleitung die Tasten und Funktionen wie folgt dargestellt:

ASN / SIN → SHIFT ASN bedeutet ASN ($\sin^{-1}$).
ASN / SIN → SIN bedeutet SIN (sin).

DEL / ◀ → SHIFT DEL bedeutet DEL, die Löschfunktion.
DEL / ◀ → ◀ bedeutet die Funktion zur Versetzung des Cursors nach links.

Zahlen werden ohne Klammern dargestellt.

1 2 3 • 4 wird als 123.4 dargestellt.

Es kann gelegentlich vorkommen, daß diese Tasten auch später noch wie bis jetzt in dieser Bedienungsanleitung dargestellt werden.

** Σx, Σx^2, n, CD, s, σ sind Funktionen für statistische Berechnungen.

Wie allgemein üblich wird Null (0) von diesem Computer zur Unterscheidung vom großen Buchstaben "O" als Ø angezeigt. Diese Darstellung wird auch stellenweise in dieser Bedienungsanleitung angewendet, wenn dies aus Klarheitsgründen erforderlich ist.

1.1.5 DIE ANZEIGE

Der **Computer** verfügt über eine 16 stellige alphanumerische Flüssigkristallanzeige. Die einzelnen Zeichen werden in einer 5 x 7-Punktmatrix dargestellt.

Das Eingaberegister faßt 80 Zeichen. Alle Zeichen, die über die 16 möglichen darstellbaren Zeichen der Anzeige hinausgehen, müssen mit den Cursor-Tasten (▶ und ◀) 'gescrolled' werden.

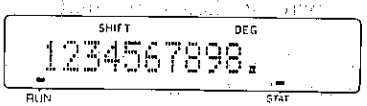
Der Cursor zeigt die Stelle an, an der die nächste Eingabe erfolgen kann. Er steht normalerweise auf der ersten freien Stelle und wird durch einen kurzen waagerechten Strich (_) dargestellt.

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

Stellt man den Cursor mit Hilfe der ▶- oder der ◀-Taste über ein bereits eingegebenes Zeichen, so erkennt man die Position am Blinken des Cursors an dieser Stelle in der vollen Punktmatrix.

Anhaltender Druck auf die Cursor-Tasten läßt den Cursor schnell an die gewünschte Stelle der Zeile bringen. Ein kurzer Druck verschiebt die Anzeige nur um ein Zeichen.

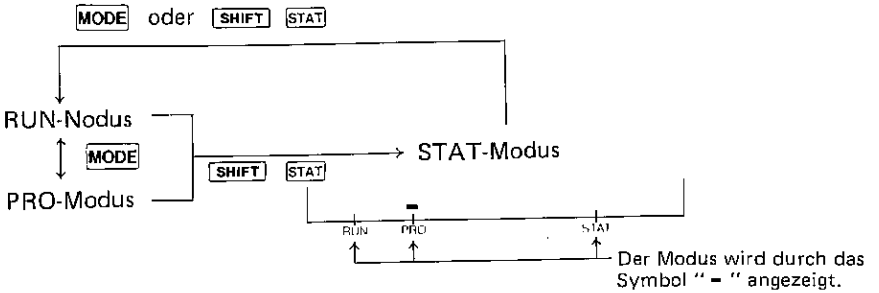
Versucht man mehr als 80 Zeichen einzulesen, werden diese vom Rechner nicht mehr angenommen. Jede zusätzliche Eingabe überschreibt das zuletzt eingegebene Zeichen. Der Cursor verändert hier sein Aussehen; er blinkt auf dem letzten Zeichen in der vollen Punktmatrix.



1.1.6 WAHL DER BETRIEBSART

Der Computer verfügt über drei Betriebsarten: RUN-Modus [RUN] für manuelle Berechnungen und Ausführen von Programmen, PRO-Modus [PRO] zum Speichern von Daten im Computer und Durchführen von Änderungen. Diese beiden Betriebsarten ergeben kombiniert den BASIC-Modus. Sie werden mit der Taste [MODE] gewählt.
Die dritte Betriebsart ist der STAT-Modus [STAT] zum Durchführen von statistischen Berechnungen. Diese Betriebsart wird durch Drücken von [SHIFT] [MDF] ein- und ausgeschaltet.

<Modus-Wahl>



1.2 GRUNDLAGEN SHARP CE-126P

1.2.1 ALLGEMEINES

Die Option **SHARP CE-126P** (Thermodrucker mit integriertem Kassetten-Interface) ermöglicht es Ihnen, Ihre mit dem **Computer** erstellten Programme oder Daten auszudrucken bzw. mit einem externen Kassettenrecorder zu speichern.

Eigenschaften des **CE-126P**:

- 24-Zeichen-Thermodrucker
- Manueller oder Programmgesteuerter Betrieb des Druckers
- 300-Baud-Kassetten-Interface
- Manueller oder Programmgesteuerter Betrieb des Kassetten-Interface
- Trockenbatterie-Betrieb (Transportfähigkeit)/Netzadapterbetrieb

Der Anschluß des **CE-126P** an Ihren **Computer** ist in dem, dem **CE-126P** beigelegten Handbuch beschrieben.

1.2.2 VERWENDUNG DES DRUCKERS

1.2.2.1 Manueller Druckbetrieb (Protokollerstellung)

Der manuelle Druckbetrieb ermöglicht es Ihnen, Ihre Rechnungen sofort auf Papier zu protokollieren. Dieser Betrieb wird durch Drücken der [SHIFT] - und der [ENTER]-Taste eingeschaltet. Am Display wird dies durch einen kleinen waagrechten Strich oberhalb von 'PRINT' angezeigt.

Auf dem Thermopapier wird bei der Protokollführung die gleiche Zeichenfolge in derselben Form, wie am Display angezeigt, ausgedruckt.

Nach Abschluß der Eingabe mit [ENTER] wird der Druck gestartet. Vorher können Sie Ihre Eingaben, wenn notwendig, noch korrigieren.

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=2*π*F
                  314.1592654
P=60*X
                  18849.55592
P/32
                  589.0486225
```

1.2.2.2 Programmgesteuerter Ausdruck

Enthalten BASIC-Programme LPRINT-Anweisungen, so können die den Anweisungen folgenden Werte oder Textausdrücke über den angeschlossenen Thermodrucker (Option **CE-126P**) ausgedruckt werden.

Programme, die mit PRINT-Anweisungen geschrieben wurden, können ganz einfach umgewandelt werden, so daß die Ausgabe nicht über das Display, sondern über den angeschlossenen Drucker erfolgt. Dies wird mit dem Befehl PRINT = LPRINT erreicht, der in einer Programmzeile dem Programm vorangestellt oder im RUN-Mode direkt eingegeben werden kann.

Im letzteren Fall muß das Programm jedoch mit [DEF] – Taste oder GOTO <Zeilennummer> gestartet werden.

Mit dem Befehl PRINT = PRINT kann dieser Befehl wieder aufgehoben werden.

Die Umschaltung kann auch in Abhängigkeit von einem logischen Vergleichsausdruck innerhalb einer IF-Anweisung erfolgen.

Das im Hauptspeicher gespeicherte Programm kann mittels der LLIST-Anweisung am Drucker gelistet werden. Näheres hierzu siehe LLIST-Anweisung, Seite 154.

Ist die zu druckende Programmzeile länger als 24 Zeichen, so erfolgt die Ausgabe am Drucker zwei- bzw. mehrzeilig. Dabei wird der Druck ab der zweiten Zeile um vier bzw. um sechs Stellen eingerückt, um den Ausdruck übersichtlicher zu gestalten.

Hinweise:

1. Wenn während des Drucks ein Fehler auftritt, der im Zusammenhang mit nicht richtig transportiertem Papier steht, so reißen Sie das Papier an der Abrißkante ab und ziehen das verbleibende Papier aus dem Drucker.
   Löschen Sie die Fehlermeldung durch Drücken der [CL] -Taste und legen Sie die Papierrolle wieder richtig ein.

2. Wenn der Drucker durch irgendwelche externe elektrische Störfelder beeinflußt wird, kann es sein, daß der Ausdruck fehlerhaft ist. Drücken Sie die [BRK] -Taste, um den Druck zu unterbrechen. Schalten Sie den Thermodrucker **CE-126P** aus und nach einigen Sekunden wieder ein. Initialisieren Sie den Drucker durch Drücken von [CL] .

3. Schalten Sie den Drucker nur ein, wenn Sie ihn benützen (Schonung der Batterien).

1.2.3 Verwendung des integrierten Kassetten-Interfaces

Das integrierte Kassetten-Interface ermöglicht es Ihnen, Ihre Programme bzw. Daten auf Band zu speichern. Dazu benötigen Sie einen für Datenaufzeichnung geeigneten Kassettenrekorder,z.B. den CE-152.

Einmal gespeicherte Programme oder Daten können ganz leicht wieder in den Rechner geladen werden.

1.2.3.1 Auswahl des Kassettenrekorders

Wir empfehlen Ihnen, für Ihre Programmspeicherung den extra dafür vorgesehenen Kassettenrekorder CE-152 zu verwenden. Der CE-152 wurde eigens zur Speicherung der mit dem SHARP-Taschencomputer erstellten Programme oder Daten entwickelt.

Wenn Sie jedoch einen anderen Rekorder verwenden wollen, sollte dieser folgende Ausstattung haben:

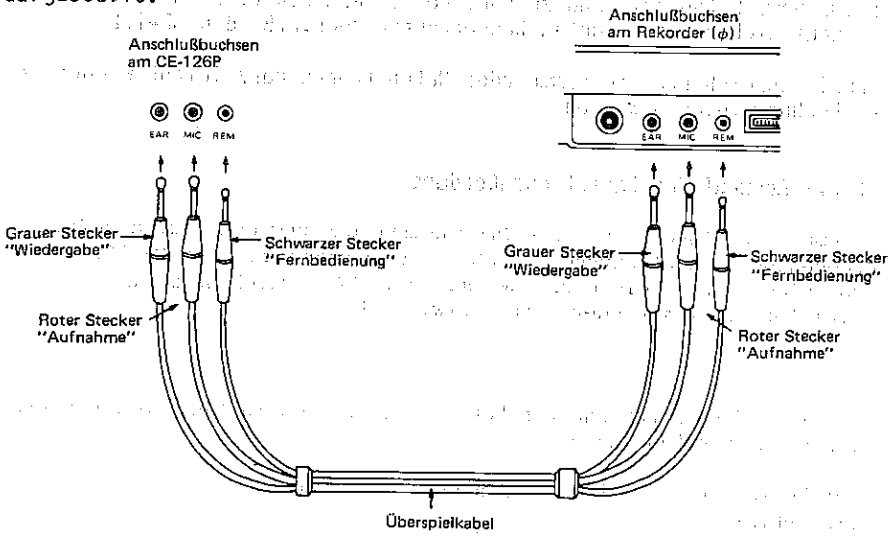
- Fernbedienung (REMOTE), Klinkenbuche 2,5 mm ø
- Bandzählwerk
- Mikrofoneingang (MIC), Klinkenbuchse 3,5 mm ø
- Ausgang (EAR), Klinkenbuchse 3,5 mm ø

Daneben sind folgende technische Daten gefordert:

- Eingangsimpedanz (MIC)	200...1000 Ohm
- Eingangsempfindlichkeit	Kleiner als 3 mV (-50 dB)
- Ausgangsimpedanz (EAR)	Kleiner als 10 Ohm
- Ausgangsleistung (EAR)	Größer als 1 V
- Klirrfaktor	Kleiner als 15 % im Bereich zwischen 2...4 kHz
- Gleichlaufschwankungen	0,3 % max. (W.R.M.S)

1.2.3.2 Anschluß des Kassettenrekorders an das CE-126P

Die herzustellenden Verbindungen sind in der folgenden Abbildung dargestellt.



1.2.4 DATENSPEICHERUNG AUF MAGNETBAND

1.2.4.1 Allgemeines

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

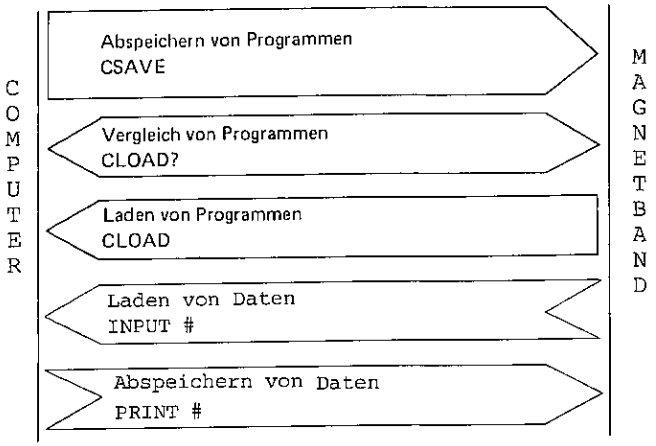
Die Informationen aus Daten und Programmen werden in Blöcken nach dem Zweitonverfahren auf das Magnetband geschrieben.

Auf einem Magnetband können abhängig vom Kassettentyp sehr viele Programme gespeichert werden. Damit Sie die Daten oder Programme wiederfinden und unterscheiden können, beginnt jeder Block aus einem Blocknamen (Programmnamen), der aus maximal 7 Zeichen besteht. Abgeschlossen wird der Speichervorgang, wenn die gesamte Information abgespeichert ist.

Um eine sichere Trennung der Blöcke zu gewährleisten, schreibt der Rechner vor dem Blocknamen automatisch etwa 5 bis 7 Sekunden lang einen konstanten Signalton. Ein kompletter Block hat damit folgende Form:

	Signalton	Blockname	Programme oder Daten	

Folgende Möglichkeiten werden bei der Datenspeicherung auf Magnetband geboten:



Hinweis:

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

1.2.4.2 Speichern von Daten und Programmen

Schließen Sie Ihren Kassettenrecorder wie in Abschnitt 1.2.3.2 beschrieben an Ihr CE-126P an. Nachdem Sie Ihr Programm eingegeben haben, können Sie den Rekorder für die Abspeicherung vorbereiten:

- Stellen Sie den REMOTE-Schalter des Interfaces auf OFF.
- Legen Sie eine Kassette in den Rekorder ein.
- Vergewissern Sie sich, daß die eingelegte Kassette zurückgespult ist.
- Suchen Sie eine freie Stelle auf dern Band.
  Merken Sie sich den Zählerstand.
- Stellen Sie den REMOTE-Schalter des Interfaces auf ON.
- Bereiten Sie die Aufnahme vor:
  "RECORD"- und "PLAY"-Taste drücken. Lautstärkeregler auf Mitte bis Maximum stellen, Tonhöhenregler auf "Höhen".
- Kommando Eingabe:
  Mit der CSAVE-Anweisung können Sie Ihr Programm gleichzeitig mit einem Namen, dem Blocknamen (Programmnamen) versehen.

Wenn Sie die [ENTER] -Taste betätigt haben, setzt sich das Magnetband in Bewegung. Ihr Programm wird jetzt übertragen und unter dem angegebenen Namen abgespeichert. Zu Beginn hören Sie 5 bis 7 Sekunden den konstanten Signalton, anschließend eine Folge von Tönen. Sobald der Computer am Ende des Programms angelangt ist, stoppt das Band. Das Bereitschaftssymbol wird auf der Anzeige des **Computers** angezeigt.

1.2.4.3 Oberprüfen der Abspeicherung

Nach der Abspeicherung Ihres Programms mit der CSAVE-Anweisung 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 der CLOAD?-Anweisung. Der Computer vergleicht das Programm in seinem Speicher mit dem auf dem Band gespeicherten Informationen. Ist die Aufzeichnung fehlerfrei, wird nach Abschluß der Überprüfung am Display des **Computers** wieder das Bereitschaftszeichen angezeigt.
Stellt der Computer beim Vergleich der Informationen einen Fehler fest, erfolgt die Fehlermeldung ERROR 8.
Löschen Sie das aufgezeichnete Programm und versuchen Sie es nochmals zu speichern. Gegebenenfalls verwenden Sie eine andere Bandstelle bzw. eine andere Kassette.

1.2.4.4 Laden von Programmen oder Daten

Beim Laden von Programmen gehen Sie wie folgt vor:

- Lautstärkeregler auf Mitte bis Maximum stellen; Tonhöhenregler auf "Höhen".
- Spulen Sie das Band an den Anfang zurück.
- Stellen Sie das Zählwerk auf Null.
- Suchen Sie mit Hilfe des Zählwerks die Bandstelle, an der Ihr Programm beginnt.
- Schalten Sie den REMOTE-Schalter des Kassetten-Interface auf ON.
- Drücken Sie die "PLAY"-Taste Ihres Kassettenrekorders.
- Geben Sie ein:

```
CLOAD "PROG. 1"
```

Wenn Sie die [ENTER]-Taste gedrückt haben, setzt sich das Magnetband in Bewegung.
Am Display wird während des Ladevorgangs ein Asterix (*) angezeigt.

Bei der Eingabe der CLOAD-Anweisung gibt es zwei Möglichkeiten:

CLOAD oder CLOAD "Programmname"

Dabei wird das Programm vom Magnetband in den Speicher des Rechners übertragen. Die beiden CLOAD-Anweisungen unterscheiden sich nur dadurch, daß bei der zweiten Anweisung der Blockname (Programmname) zusätzlich mit eingegeben wird. Damit wird erreicht, daß nur jenes Programm, dessen Blockname mit dem eingegebenen übereinstimmt, übertragen wird. Alle anderen auf Band gespeicherten Programme werden überlesen.

T E I L II

P R A X I S

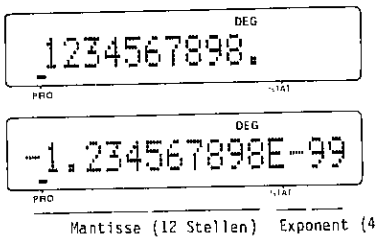
TEIL II P R A X I S

2.1 SHARP PC-1430 - EINSATZ ALS MANUELLER RECHNER

2.1.1 BETRIEBSHINWEISE

Die folgenden Absätze enthalten einige wichtige Hinweise, die sich auf die Verwendung des Rechners beziehen. Die in TEIL I, Abschnitt 1.1 enthaltenen allgemeinen Betriebshinweise haben darüber hinaus Gültigkeit.
Das Umschalten des Rechners in die RUN-Betriebsart (mit der MODE-Taste) wurde bereits in TEIL I, Kapitel 1.1.6 aufgezeigt, so daß an dieser Stelle auf eine nähere Erläuterung dieser Punkte verzichtet werden kann.

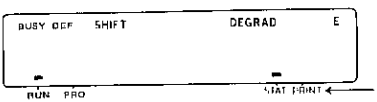
2.1.1.1 Anzeige/Display-Format und Symbole.



Fließkommaschreibweise (Normalformat)

Wissenschaftliche Schreibweise

Die Anzeige des **Compuers** umfaßt 16 Stellen. In der Betriebsart "RUN" werden die Rechenergebnisse soweit möglich in Fließkommaschreibweise dargestellt. Ist das Rechenergebnis kleiner 0,000000001 oder größer 9999999999 (größer −0,000000001 oder kleiner −9999999999), erfolgt die Anzeige in wissenschaftlicher Schreibweise. Dabei wird die Mantisse 12stellig (einschließlich Vorzeichen und Komma) und der Exponent 4 stellig (einschließlich Symbol und Vorzeichen) angezeigt.



Ausdruck über Drucker siehe 1.2 "GRUNDLAGEN SHARP CE-126P"

Der **Computer** verwendet die oben abgebildeten Symbole und Markierungen, die nachfolgend aufgelistet sind und deren Bedeutung erläuter wird.

BUSY: Dieses Symbol zeigt an, daß ein Programm abgearbeitet wird.

DEF: Nach Betätigung der Taste [DEF] wird dieses Symbol angezeigt. Dann wirken die alphabetischen Tasten als "defineable Keys", d.h. dann beginnt die Programmausführung mit der Zeile, die das entsprechende Zeichen als Marke enthält.

SHIFT: Dieses Symbol wird angezeigt, wenn die Taste [SHIFT] gedrückt wird. Durch Drücken von Tasten sind dann die Symbole und Funktionen zugänglich, die in hellbraun über den Tasten angegeben sind.

DEG, RAD, GRAD: Durch Drücken von [SHIFT] [DRG] werden die Winkeleinheiten Grad, Radiant und Neugrad für trigonometrische Funktionen, inverse trigonometrische Funktionen und Koordinatenumwandlung nacheinander gewählt.

RUN, PRO: Diese Betriebsarten werden mit der Taste [MODE] gewählt. Die jeweils gültige Betriebsart wird durch die Marke " ▬ " angezeigt.

STAT: Durch Drücken von [SHIFT] und [STAT MDF] wird die STAT-Betriebsart für statistische Berechnungen eingeschaltet. Dies wird durch einen Strich über STAT auf der Anzeige angezeigt.

PRINT: Die PRINT-Anzeige leuchtet nur, wenn der Drucker (Option) angeschlossen ist und [SHIFT] [P↔NP] [ENTER] eingegeben wird. Zum Ausschalten des Druckbetriebs die Tasten erneut drücken.
Wenn der Drucker angeschlossen ist, können in der PRINT-Betriebsart manuelle Berechnungen und Ergebnisse ausgedruckt werden.

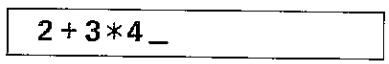
E: Das Symbol "E" in der rechten oberen Ecke des Anzeigefelds zeigt an, daß eine Bereichsüberschreitung aufgetreten ist oder eine falsche Anweisung gegeben wurde.

2.1.2 Normale Berechnungen

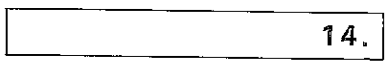
Für die Grundrechenarten verwendet der Computer die Symbole "+", "−", " * " (anstelle von "x") und "/" (anstelle von "÷"), sowohl bei manuellen Berechnungen als auch in BASIC-Programmen. Bei manuellen Berechnungen wird das Ergebnis mit der Taste [ENTER] und nicht mit dem Gleichheitszeichen (=) abgerufen.

Beispiel: 2 + 3 x 4 =

Eingabe: 2 [+] 3 [*] 4



[ENTER]



Beispiel: 5 x (−6) + 7 =

Eingabe: 5 [*] [−] 6 [+] 7

```
5*−6+7_
```

[ENTER]

```
−23.
```

- Wie im obigen Beispiel wird ein Minuszeichen (−), das auf ein anderes Rechensymbol folgt oder zwischen Klammerausdrücken steht, vom Computer als Kennzeichnung für negative Werte und nicht als Subtraktionszeichen verstanden.

Beispiel: (6 + 5) ÷ (4 − 1) =

Eingabe: [(] 6 [+] 5 [)] [/]
[(] 4 [−] 1 [)]

```
(6+5) / (4−1)_
```

[ENTER]

```
3.666666667
```

Beispiel: $-5 \times 10^{3} \div (4 \times 10^{-3})$ =

Eingabe: [CL] [−] 5 [Exp] 3 [/] 4
[Exp] [−] 3

```
−5E3/4E−3_
```

[ENTER]

```
−125ØØØ.
```

- Das Symbol [E], das Exponentialwerte kennzeichnet, kann mit der Taste [E] auf der Tastatur eingegeben werden. Jedoch wird die Verwendung der Taste [Exp] empfohlen, weil dies sowohl praktischer ist als auch zu weniger Rechenfehlern führt.

Beispiel: 520000 x 43200 =

Eingabe: 520000 [*] 43200 [ENTER]

```
2.2464E 1Ø
```

(2.2464×10^{10})

Formeln können in der gleichen Form eingegeben werden, wie sie normalerweise auf Papier geschrieben werden, das Ergebnis wird durch Drücken der Taste [ENTER] berechnet.
Für normale Berechnungen stehen damit die folgenden Operatoren zur Verfügung:

Addition	+	positiv, negativ	+ und −
Subtraktion	−	Exponenten	E [Exp]
Multiplikation	* (Sternchen)	Ergebnis	[ENTER]
Division	/ (Schrägstrich)		

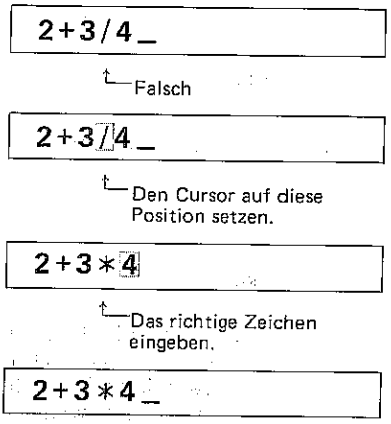
2.1.3 Eingabe-Korrektur

Bei der Eingabe von Berechnungen und Programmen werden oft die falschen Tasten gedrückt oder Zeichen vergessen. Diese Fehler können einfach und schnell korrigiert werden.

(1) Mehrere Fehler bei der Eingabe
Durch Drücken der Taste [CL] wird die gesamte Eingabe gelöscht.

(2) Drücken der falschen Taste
Den Cursor durch Drücken der Taste [◄] zum falschen Zeichen bringen und das richtige Zeichen eingeben.

Beispiel: 2 + 3 * 4 wurde eingegeben als
Eingabe: 2 [+] 3 [/] 4

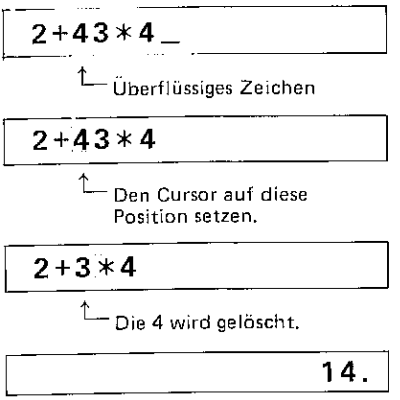


(3) Eingabe von nicht erforderlichen Zeichen

Den Cursor auf das überflüssige Zeichen setzen und die Tasten [SHIFT] und [DEL ◄] drücken.

Beispiel: 2 + 3 ∗ 4 wurde eingegeben als

Eingabe: 2 [+] 43 [∗] 4

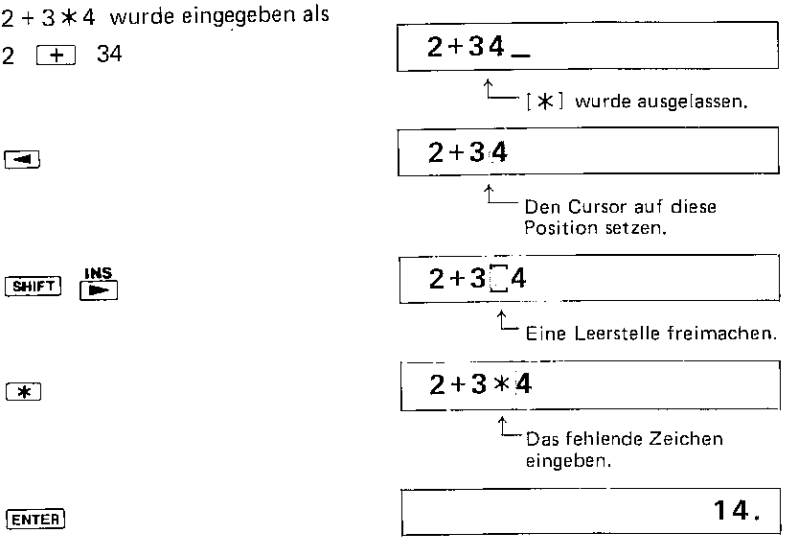


(4) Zeichen wurden bei der Eingabe ausgelassen

Den Cursor auf die Position setzen, wo das "vergessene" Zeichen eingefügt werden muß, und [SHIFT] [INS ►] drücken. Dann wird eine Leerstelle zum Einfügen eines Zeichens freigemacht.

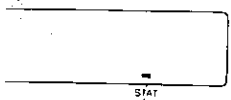
Beispiel: 2 + 3 ∗ 4 wurde eingegeben als

Eingabe: 2 [+] 34



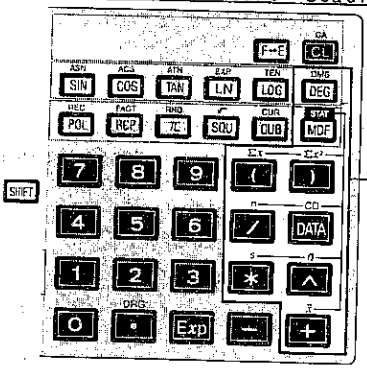
2.1.4 STATISTISCHE BERECHNUNGEN

Die Betriebsart "Statistische Berechnungen" wird durch Drücken der Tasten SHIFT und STAT angewählt.
Die Markierung (—) im rechten unteren Bereich des Anzeigefelds über der Beschriftung "STAT" gibt an, daß sich der Rechner in Betriebsart "Statistische Berechnungen" befindet.
Das Ausschalten der Betriebsart erfolgt durch Drücken der Taste MODE oder der Tasten SHIFT und STAT.



Betriebsart "Statistische Berechnungen" ist angewählt

Tastenfeld für die Betriebsart "Statistische Berechnungen"



Speicherung von Zwischen- und Endergebnissen
Bei der Durchführung von statistischen Berechnungen werden die nachfolgend aufgeführten Ergebnisse automatisch den angegebenen, in der Betriebsart "BASIC" verwendeten Standardvariablen zugeordnet. Die Werte bleiben bei der Umschaltung in die Betriebsart "BASIC" erhalten und können somit direkt für weitere Berechnungen verwendet oder in ein Programm übernommen werden.

Variable	Z	Y	X
Statistik	n	Σx	Σx^2

In diesem Zusammenhang ist jedoch folgendes zu beachten:

- Den Standardvariablen X bis Z wird ein Wert zugewiesen, d.h., ein gespeichertes Programm kann ggf. beeinträchtigt werden.

* Im STAT-Modus können keine BASIC-Programme eingesetzt werden. Kommandos wie MEM, LEN, VAL und andere, die numerische Werte zum Ergebnis haben, können jedoch in dieser Betriebsart verwendet werden.

Hinweis: Durch Einschalten des STAT-Modus werden die Speicher X, Y und Z gelöscht. Außerdem werden sie gelöscht, wenn sie für die Variablen X, Y und Z oder X$, Y$ und Z$ verwendet werden.

Statistische Berechnungen mit einer Variablen

In den statistischen Berechnungen finden folgende Benennungen Verwendung:

n	Anzahl der eingegebenen Daten
Σx	Gesamtsumme der eingegebenen Daten
Σx^2	Quadratsumme der eingegebenen Daten
$\bar{x}$	Mittelwert der eingegebenen Daten $\bar{x} = \frac{\Sigma x}{n}$

s Standardabweichung mit Gesamtheitsparameter " n-1 " (Stichproben-Standardabweichung)

$$s = \sqrt{\frac{\Sigma x^2 - n\bar{x}^2}{n-1}}$$

(Wird dann verwendet, wenn die eingegebenen Daten eine Auswahl bzw. Stichprobe aus der Gesamtheit darstellen.)

σ Standardabweichung mit Gesamtheitsparameter " n " (Standardabweichung der Gesamtheit)

$$\sigma = \sqrt{\frac{\Sigma x^2 - n\bar{x}^2}{n}}$$

(Wird dann verwendet, wenn die eingegebenen Daten als Gesamtheit aufzufassen sind bzw. die Stichproben die Gesamtheit ergeben.)

Die Daten werden nach dem folgenden Verfahren eingegeben:

(1) Zum Eingeben einzelner Werte die Taste [DATA] drücken.

(2) Um den gleichen Wert mehrfach einzugeben, den Wert, [*], die Häufigkeit und [DATA] eingeben.

Hinweis: Wenn der Wert oder die Häufigkeit in Form einer Gleichung eingegeben werden, müssen sie in Klammern gesetzt werden, weil sonst [*] vom Computer als arithmetischer Operand angesehen wird.

Beispiel: (127 – 100) [*] 3 [DATA] bewirkt eine dreimalige Eingabe von "27".
Durch 127 – 100 [*] 3 [DATA] wird "–173" einmal eingegeben.

(Bei statistischen Berechnungen wird mit "Häufigkeit" bezeichnet, wenn der gleiche Wert mehrfach vorkommt. Wenn beispielsweise ein Wert dreimal vorhanden ist, hat dieser Wert die Häufigkeit "3".)

Beispiel: Die Tabelle unten erhält die Ergebnisse eines Tests, den 35 zufällig ausgewählte Personen ausgeführt haben. Wir wollen jetzt mit diesen Daten den Mittelwert und die Standardabweichung der Testergebnisse berechnen.

Nr.	Punktzahl	Anzahl Personen	Nr.	Punktzahl	Anzahl Personen
1	30	1	5	70	8
2	40	1	6	80	9
3	50	4	7	90	5
4	60	5	8	100	2

Eingabe: [SHIFT] [STAT] (Zum Einschalten des STAT-Modus)

30 [DATA] 40 [DATA]

50 [*] 4 [DATA] 60 [*] 5 [DATA]

70 [*] 8 [DATA] 80 [*] 9 [DATA]

90 [*] 5 [DATA] 100 [*] 2 [DATA]

```
)
```

```
2.
```

(Standardabweichung, wobei der Stichprobenumfang als Grundgesamtheit angesehen wird.)

```
35.
```

SHIFT $\bar{x}$	(Mittelwert $\bar{x}$)	71.42857143
SHIFT s	(Standardabweichung der Stichprobe s)	16.47508942
SHIFT σ	(Standardabweichung der Grundgesamtheit σ)	16.23802542
	(Standardabweichung, wobei der Stichprobenumfang als Grundgesamtheit angesehen wird.)	
SHIFT n	(Stichprobenumfang n)	35.
SHIFT Σx	(Stichproben-Summe Σx)	2500.
SHIFT Σx^2	(Summe der Stichproben-Quadrate Σx^2)	187800.
SHIFT STAT	(Zum Ausschalten des STAT-Modus)	)

- Die Berechnungen können mit den bereits eingegebenen statistischen Daten fortgesetzt werden, wenn nach Berechnung des Mittelwerts und der Standardabweichung als Zwischenergebnisse neue Werte eingegeben werden. Die Reihenfolge, in der Daten eingegeben werden, ist nicht festgesetzt und kann nach Bedarf geändert werden.

Korrektur von falsch eingegebenen Daten

Zur Korrektur von falschen Daten wird SHIFT und CD gedrückt (CD bedeutet Datenkorrektur).

Beispiel: 35 soll 4 mal eingegeben werden, aber versehentlich wurde 25 eingegeben.

25 * 4 DATA
25 * 4 SHIFT CD
35 * 4 DATA

Wenn falsche Daten eingegeben wurden, den gleichen falschen Wert eingeben, SHIFT CD drücken und dann den richtigen Wert eingeben.

Wenn der Fehler sofort bemerkt wird:

25 * 4 DATA ◄ → 25 * 4 D A T A _
◄ → 25 * 4 D A T A
SHIFT CD →

Den Cursor zurücksetzen.

Beispiel: 40 soll 5 mal eingegeben werden, aber versehentlich wird 40 [*] 7 eingegeben:

40 [*] 7 DATA
40 * 2 SHIFT CD

Wenn ein Wert zu oft eingegeben wurde, den überschüssigen Teil abziehen.

2.1.5 RECHENBEREICH

Arithmetische Berechnungen:

Erster Operand
Zweiter Operand
Rechenergebniss
} $\pm 1 \times 10^{-99}$ bis $\pm 9{,}999999999 \times 10^{99}$ und 0

Wissenschaftliche Berechnungen:

Funktion	Rechenbereich	Anmerkung
$\sin x$ $\cos x$ $\tan x$	DEG: $\lvert x \rvert < 1 \times 10^{10}$ RAD: $\lvert x \rvert < \frac{\pi}{180} \times 10^{10}$ GRAD: $\lvert x \rvert < \frac{10}{9} \times 10^{10}$ Für tan x gelten folgende Einschränkungen DEG: $\lvert x \rvert = 90(2n-1)$ RAD: $\lvert x \rvert = \frac{\pi}{2}(2n-1)$ n = Ganze Zahl GRAD: $\lvert x \rvert = 100(2n-1)$	
$\sin^{-1} x$ $\cos^{-1} x$	$-1 \leqq x \leqq 1$	
$\tan^{-1} x$	$\lvert x \rvert < 1 \times 10^{100}$	
$\ln x$ $\log x$	$1 \times 10^{-99} \leqq x < 1 \times 10^{100}$	$(\ln x = \log_e x)$
e^x	$-1 \times 10^{100} < x \leqq 230{,}2585092$	$(e \fallingdotseq 2.718281828)$
10^x	$-1 \times 10^{100} < x < 100$	
y^x	• $y > 0$: $-1 \times 10^{100} < x \log y < 100$ • $y = 0$: $x > 0$ • $y < 0$: x: ganzzahlig oder 1/x: ungerade $-1 \times 10^{100} < x \log \lvert y \rvert < 100$	$y^x = 10^{x \cdot \log y}$
$\sqrt[3]{x}$	$\lvert x \rvert < 1 \times 10^{100}$	
MDF x	$\lvert x' \rvert < 1 \times 10^{100}$ x': gerundeter Wert	
$\sqrt{x}$	$0 \leqq x < 1 \times 10^{100}$	
x^2	$\lvert x \rvert < 1 \times 10^{50}$	
x^3	$\lvert x \rvert < 2{,}154434690 \times 10^{33}$	
$\frac{1}{x}$	$\lvert x \rvert < 1 \times 10^{100}$ $x \neq 0$	
n!	$0 \leqq n \leqq 69$ (n = Ganze Zahl)	
→DEG	$\lvert x \rvert < 1 \times 10^{100}$	
→D.MS	$\lvert x \rvert < 1 \times 10^{100}$	
HEX → DEC	$0 \leqq x \leqq$ 2540BE3FF FDABF41C01 $\leqq x \leqq$ FFFFFFFFFF	x ist in "HEX" eine ganze Zahl

Funktion		Rechenbereich	Anmerkung
$x, y \rightarrow r, \theta$		$(x^2 + y^2) < 1 \times 10^{100}$ $\frac{y}{x} < 1 \times 10^{100}$	$r = \sqrt{x^2 + y^2}$ $\theta = \tan^{-1} \frac{y}{x}$
$r, \theta \rightarrow x, y$		$r < 1 \times 10^{100}$, $\lvert r \sin\theta \rvert < 1 \times 10^{100}$ $\lvert r \cos\theta \rvert < 1 \times 10^{100}$	$x = r \cos\theta$ $y = r \sin\theta$ θ ist in derselben Bedingung wie x von sin x, cos x
Statistische Berechnungen	Data CD	$\lvert x \rvert < 1 \times 10^{50}$ $\lvert \Sigma x \rvert < 1 \times 10^{100}$ $\Sigma x^2 < 1 \times 10^{100}$ $\lvert n \rvert < 1 \times 10^{100}$	

Funktion		Rechenbereich	Anmerkung
	$\bar{x}$	$n \neq 0$	
	S	$n \neq 1$ $0 \leqq \frac{\Sigma x^2 - n\bar{x}^2}{n-1} < 1 \times 10^{100}$	
	σ	$n \neq 0$ $0 \leqq \frac{\Sigma x^2 - n\bar{x}^2}{n} < 1 \times 10^{100}$	

Die Genauigkeit beträgt in der Regel für die Fließkomma-Schreibweise ±1 in der 10. Stelle und für die wissenschaftliche Schreibweise ±1 in der 9. Nachkommastelle der Mantisse.
Die Berechnungsgenauigkeit sinkt jedoch in der Nähe des Singulär-

Punktes und des Umkehrpunktes der Funktion ab.
Ferner ist zu beachten, daß bei Kettenrechnungen eine Kummulierung des Fehlers erfolgt und damit mit jedem Rechengang eine Verschlechterung der Genauigkeit einhergeht. Der gleiche Effekt tritt rechnerintern bei der Durchführung von Funktionen wie y^x auf.

2.1.6 Wissenschaftliche Berechnungen

Von diesen Funktionen können INT, ABS und SGN und auch einige andere Funktionen mit den alphabetischen Tasten eingegeben werden. Beispielsweise kann "sin 30" als [sin] 30 oder als [S] [I] [N] 30 eingegeben werden. Für trigonometrische und inverse trigonometrische Funktionen und für Koordinatenumwandlung muß vorher das gewünschte Winkelmaß spezifiziert werden. Bei manuellen Berechnungen können die Winkelmaße mit [SHIFT] [DRG] oder mit den folgenden Anweisungen spezifiziert werden:

Winkelmaß	Kommando	Anzeige-Symbol	Wirkung
Grad	DEGREE	DEG	Ein rechter Winkel wird als 90° dargestellt.
Radiant	RADIAN	RAD	Ein rechter Winkel wird als $\pi/2$ dargestellt.
Neugrad	GRAD	GRAD	Ein rechter Winkel wird als 100^g dargestellt.

Diese Anweisungen dienen auch zum Spezifizieren der Winkeleinheit in Programmen. Verwenden Sie diese Anweisungen jetzt zur Ubung in den folgenden Rechenbeispielen:

Beispiel: $\sin 30^\circ =$

Eingabe: DEGREE [ENTER] (Spezifiziert Grad.)
SIN 30 [ENTER]

Beispiel: $\tan \frac{\pi}{4} =$

Eingabe: RADIAN [ENTER] (Spezifiziert Radiant.)
TAN (PI/4) [ENTER]

Beispiel: $\cos^{-1} (-0{,}5) =$

Eingabe: DEGREE [ENTER] (Spezifiziert Grad.)
ACS –0.5 [ENTER]

Beispiel: log 5 + ln 5 =
Eingabe: LOG 5 + LN 5 [ENTER]

Beispiel: e^{2+3} =
Eingabe: EXP (2 + 3) [ENTER]
Nicht die Taste [EXP] verwenden.

Beispiel: $\sqrt[3]{4^3 + 5^3}$ =
Eingabe: CUR (4 ^ 3 + 5 ^ 3) [ENTER]

Beispiel: 5! =
Eingabe: FACT 5 [ENTER]

Beispiel: Umwandlung der Hexadezimalzahl CF8 in eine Dezimalzahl.
Eingabe: &CF8 [ENTER]

Beispiel: Umwandlung der Sexagesimaldarstellung 30°30′ in eine Dezimalzahl.
Eingabe: DEG 30.30 [ENTER]

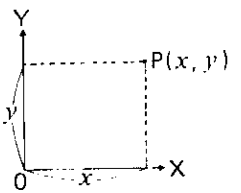
Beispiel: Umwandlung der Dezimalzahl 30,775 in Sexagesimaldarstellung.
Eingabe: DMS 30.755 [ENTER]

Koordinatenumwandlung: POL, REC

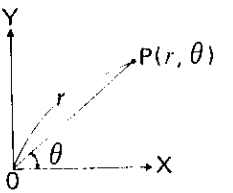
POL: Umwandlung von rechtwinkligen Koordinaten (x, y) in Polarkoordinaten (r, θ).

REC: Umwandlung von Polarkoordinaten (r, θ) in rechtwinklige Koordinaten (x, y).

Rechtwinklige Koordinaten



Polarkoordinaten



- Für θ gelten die folgenden Bereiche:
  DEG: $0 \leqq |\theta| \leqq 180$
  RAD: $0 \leqq |\theta| \leqq \pi$
  GRAD: $0 \leqq |\theta| \leqq 200$

Beispiel: Umwandlung von orthogonalen Koordinaten in Polarkoordinaten. Berechnung der Polarkoordinaten (r, θ) für den rechtwinkligen Koordinaten-Punkt (3, 8):

Eingabe: DEGREE [ENTER] (Spezifiziert "Grad".)

POL (3, 8) [ENTER] (r)

Z [ENTER] (θ)

* Der Wert von θ wird in der Variablen Z und der Wert von r in der Variablen Y gespeichert.

Beispiel: Umwandlung von Polarkoordinaten in rechtwinklige Koordinaten. Berechnung der rechtwinkligen Koordinaten (x, y) für den Polarkoordinaten-Punkt $(12, \frac{4}{5}\pi)$.

Eingabe: RADIAN [ENTER] (Spezifiziert "Radiant".)

REC (12, (4/5 ∗ PI))

[ENTER] (x)

Z [ENTER] (y)

* Die Werte von x und y werden in den Variablen Y bzw. Z gespeichert.

Hinweis: Bei der Koordinatenumwandlung werden die Ergebnisse in den Variablen Y und Z gespeichert. Dabei werden die vorigen Werte von Y und Z gelöscht.

2.1.7 Anzeigeformat und Rundung

2.1.7.1 Anzeigeformat

Zahlen werden mit einem 12-stelligen Festkomma-Teil und 2-stelligen Exponenten-Teil angezeigt. Wird beispielsweise 0.5 [/] 90 [ENTER] eingegeben, berechnet der Computer das folgende Ergebnis:

$5.55555555555 \times 10^{-3}$

aber rundet die elfte 5 auf und speichert das Ergebnis als:

$5.555555556 \times 10^{-3}$.

Wenn diese und ähnliche Zahlen normal angezeigt werden, fällt der Exponent weg. In Werten mit mehr als 10 Stellen fällt jedoch, wie im folgenden Beispiel

0.005555555556

die letzte Stelle weg, und die Anzeige sieht folgendermaßen aus:

0.005555555

Der tatsächliche Wert, der im Speicher des Computers enthalten ist, kann jedoch durch Drücken der Taste [F↔E] abgerufen und mit Exponent angezeigt werden.

Beispiel: 0.5 [/] 90 [ENTER] → Ø.ØØ5555555

[F↔E] → 5.555555556E−Ø3

Rechenergebnisse werden so weit wie möglich bis zu zehn Stellen angezeigt (der Festkomma-Teil besteht auch aus zehn Stellen, wenn ein Exponent enthalten ist), unnötige Nullen fallen jedoch weg. Es ist aber auch möglich, die Darstellungsart (Normal oder Exponentialdarstellung) und die Anzahl der angezeigten Stellen zu spezifizieren.

Mit dem USING-Kommando (siehe Seite 164) kann die Anzahl der angezeigten Stellen bei manuellen Berechnungen und die Verwendung von Exponentialdarstellung zur Anzeige der Ergebnisse jeder Berechnungsart spezifizieren werden.

Beispiel: USING "###.####" [ENTER] Spezifiziert die Anzeige von drei Stellen, einschließlich des Minuszeichens, vor dem Komma und vier Stellen nach dem Komma.

−0.5 / 90 [ENTER] → −Ø.ØØ55

9 ∗ 5 [ENTER] → 45.ØØØØ

USING "##.####^" [ENTER] Spezifiziert Exponentialdarstellung mit vier Stellen nach dem Komma.

−0.5 / 90 [ENTER] → −5.5555E−Ø3

9 ∗ 5 [ENTER] → 4.5ØØØE Ø1

Mit dieser Funktion kann erreicht werden, daß alle Ergebnisse mit der gleichen Anzahl Nachkommastellen oder in Exponentialdarstellung angezeigt werden.
Ein einmal eingegebenes USING-Kommando bleibt für alle folgenden Berechnungen gültig, bis ein neues USING-Kommando eingegeben oder der Computer auf die Anfangseinstellung zurückgestellt wird. Die Rückstellung auf die Anfangseinstellung erfolgt durch Eingabe von SHIFT CA CL oder USING ENTER .

Das USING-Kommando ist ausführlich auf Seite 164 beschrieben.

Hinweis: Das USING-Kommando kann in BASIC-Programmen zur Spezifizierung des Anzeigeformats von Zeichenstrings verwendet werden. Dies ist bei manuellen Berechnungen jedoch nicht möglich.

2.1.7.2 Rundungsfunktion (MDF)

Dieser Computer führt Berechnungen mit einem 12-stelligen Festkomma-Teil in Exponentialdarstellung aus. Das Anzeigeformat kann zwar geändert werden, die interne Speicherung der Ergebnisse erfolgt aber als 10-stelliger Festkomma-Teil in Exponentialdarstellung. Wenn das Ergebnis einer Gleichung in der folgenden Rechnung weiterverwendet wird, wird der intern gespeicherte Wert verwendet, um die Genauigkeit der Rechnung zu erhöhen.

Beispiel 1:

USING "###.###" ENTER		
0.5 / 9 ENTER	→	Ø.Ø55
*	→	.55555556E-Ø2*_
9 ENTER	→	Ø.5ØØ

Bei der Verarbeitung von experimentiellen Daten oder statistischen Werten wird normalerweise die Anzahl der Stellen, die in den Berechnungen verwendet werden soll, spezifiziert und, wenn erforderlich, die letzte Stelle gerundet. Wenn beispielsweise das Ergebnis von "0.5/9" im obigen Beispiel 1 auf die dritte Dezimalstelle gerundet und dann in der folgenden Berechnung verwendet würde, wäre das Ergebnis der nachfolgenden Berechnung anders als im Beispiel.
Mithilfe der Rundungsfunktion können derartige Berechnungen durchgeführt werden, ohne daß die Notwendigkeit besteht, den manuell gerundeten Wert erneut einzugeben.

Beispiel 2:

USING "###.###" ENTER		
0.5 / 9 ENTER	→	Ø.Ø55
MDF	→	Ø.Ø56
*	→	Ø.Ø56*_
9 ENTER	→	Ø.5Ø4

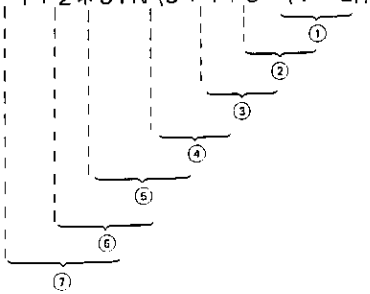
Bitte beachten Sie, daß die Rundungsfunktion MDF nur wirksam ist, wenn die Anzahl der Stellen im Ergebnis mit dem USING-Kommando spezifiziert wurde, und sonst ignoriert wird.

* MDF wird wie jede andere arithmetische Funktion eingegeben.

  Beispiel: USING "###.###" ENTER
  MDF (0.5/9)*9 ENTER → 0.504

Gleichungen können in normaler Reihenfolge in diesen Computer eingegeben werden. Da aber nicht alle Gleichungen in der gleichen Reihenfolge von links nach rechts, in der sie eingegeben werden, abgearbeitet werden, benötigt der Computer einen Bereich zum vorübergehenden Ablegen der Kommandos und Daten, die später bearbeitet werden. Dieser Speicherbereich wird "Puffer" genannt. Der PC-1430 verfügt über zwei Puffer, ein Funktionspuffer für 16 Schritte und ein Datenpuffer für acht Schritte.

Beispiel: 1+2*SIN(3+4*5^(4−2)) ENTER



Datenpuffer				Funktionspuffer				
1Schritt	2Schritt	3Schritt	...	1Schritt	2Schritt	3Schritt	4Schritt	...
4	5	4	...	−	(	^	*	...

Die Gleichung im obigen Beispiel wird beginnend mit Schritt (1) gelöst, aber die Schritte (2) bis (7) werden vorübergehend gespeichert, wie aus der Tabelle oben ersichtlich.
Dann werden die Funktionen und Daten beginnend mit Schritt (1) in den Rechenspeicher zurückgerufen. Klammern werden im Funktionspuffer gespeichert. Sie können bis zum fünfzehnten Schritt eingegeben werden, solange die Funktionen nicht die Kapazität des Funktionspuffers überschreiten.

2.2 EINSATZ ALS BASIC-RECHNER

2.2.1 RECHNEN OHNE PROGRAMMUNTERSTÜTZUNG (RUN-MODE)

Der **Computer** kann in der Programmiersprache BASIC programmiert werden. Dabei kann er auf zwei verschiedene Arten verwendet werden. Zum einen können Sie Ihre Berechnungen innerhalb eines Programms abarbeiten, zum anderen können die Anweisungen und Funktionen direkt eingegeben und ausfeführt werden (RUN-MODE).

Dabei kann man den **PC-1430** wie einen einfachen Taschenrechner benutzen. Um Das Ergebnis einer Rechnung zu erhalten muß man anstelle der '='-Taste die [ENTER] -Taste drücken.

Die allgemeine Form einer Rechnung im RUN-Mode ist:

[CL] numerischer Ausdruck [ENTER]

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

2.2.1.1 Grundrechnungsarten

a) Addition

[5] [0] [+] [5] [0] [ENTER]

```
100.
```

b) Subtraktion

[1] [0] [0] [−] [5] [0] [ENTER]

```
50.
```

c) Division

[3] [0] [0] [/] [5] [ENTER]

```
60.
```

d) Multiplikation

[6] [0] [*] [1] [0] [ENTER]

```
600.
```

e) Klammerrechnungen, Exponentation

[(] [6] [7] [5] [+] [6]
[7] [5] [0] [)] [/] [4]
[5] [0] [0] [0] [ENTER]

```
0.165
```

Hinweise:

1. Erst nach Drücken der [ENTER] Taste wird die Rechnung oder das Kommando ausgeführt und das Ergebnis angezeigt.

2. Das Ergebnis der Rechnung kann für weitere Berechnungen verwendet werden. Dazu die Zahl nicht mit der [CA/CL]-Taste löschen, sondern das nächste Operationszeichen der neuen Formel eingeben (+ - * / ^).

Beispiel:

[3] [0] [0] [*] [1] [5] [0] [ENTER]	45000.
[*] [.] [1] [5]	45000.*.15_
[ENTER]	6750.
[-] [4] [0] [0] [0]	6750.-4000_
[ENTER]	2750.
[-] [1] [2] [2] [5] [ENTER]	1525.
[-] [2] [2] [0] [0] [ENTER]	-675.

2.2.1.3 Wissenschaftliche Schreibweise

Man kann Zahlen auch im wissenschaftlichen Format eingeben. Damit ist es möglich, Zahlen bis zu einer Größenordung von ±9.999999999E ± 99 einzugeben und zu verarbeiten.

Wird dieser Wertbereich überschritten, erscheint die 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 zweistelligen Exponenten eingegeben, so werden nur die beiden zuletzt eingegebenen Ziffern bewertet.

Hinweis:

1. Wie im englischen Sprachraum üblich, wird nicht das Komma, sondern der Punkt zur Trennung von ganzem und gebrochenem Anteil von Zahlen verwendet.

2. Für den Operator "geteilt durch" wird ein Schrägstrich (/) anstelle des Doppelpunkts gesetzt.

3. Für "multipliziert mit" muß ein Stern (*) gesetzt werden.

4. Als Zeichen für die Exponentation wird ein Dach (^) verwendet. Ist die Basis negativ, muß sie in Klammern gesetzt werden. Als Exponenten sind bei negativen Grundzahlen nur ganze Zahlen zulässig. Ansonsten erscheint die 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.

2.2.1.4 Editier-/Korrekturmöglichkeit

Tippfehler können während der Eingabe oder bei der Überprüfung der Eingaben korrigiert werden.

Mit den Cursor-Tasten ◄ und ► können Sie den Cursor nach links oder rechts bewegen, ohne daß dabei die eingegebene Formel verändert wird. Zur Korrektur stellen Sie den Cursor über das zu korrigierende Zeichen und überschreiben dieses mit dem neuen Zeichen.

Zum Löschen eines Zeichens den Cursor über das zu löschende Zeichen stellen und die Tasten SHIFT und DEL drücken.

Zum Einfügen von Zeichen drücken Sie die Tasten SHIFT und INS . Es wird ein Platzhalter (␣) in die Formel eingefügt, der durch ein neues Zeichen überschrieben werden kann. Durch Drücken der ENTER-Taste werden alle überflüssigen Platzhalter aus der Formel gelöscht.

Ist das Ergebnis einer Berechnung schon ausgegeben oder erschien bei der Berechnung eine Fehlermeldung, so kann die Ausgangsformel mit den Tasten ↑ oder ↓ wieder zur Anzeige gebracht und wie oben beschrieben geändert werden.

Hinweis:

BASIC-Schlüsselwörter wie LET, SIN, COS, PRINT usw. werden nach Betätigung der ENTER-Taste intern abgekürzt. Obwohl das Schlüsselwort unverändert auf der Anzeige erscheint, wird es beim Editieren durch ein einziges neues Zeichen vollständig überschrieben. Will man sehr lange Programmzeilen eingeben, kommt man vorerst nur bis zum 80. Zeichen. Drücken Sie nun die ENTER-Taste und stellen Sie den Cursor an das Zeilenende. Nun können Sie noch zusätzlich einige Zeichen eingeben. Die Eingabe muß wieder mit ENTER abgeschlossen werden.

2.2.1.5 Mathematische Funktionen/Klammerregeln

Der **Computer** verfügt über alle gebräuchlichen mathematischen Funktionen. Diese können, im Gegensatz zu seinen Vorgängern, bei denen die Eingabe nur über die Buchstabentasten möglich war, über Funktionstasten eingegeben und im BASIC verarbeitet werden.

Beispiel:

SIN Ø ENTER

```
Ø.
```

Die Reihenfolge der Einzelschritte bei der Berechnung eines komplexen mathematischen Ausdrucks wird durch Klammern festgelegt.

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

Beispiel:

3 * 5 + 7 * 4

ist gleichbedeutend mit

(3 * 5) + (7 * 4)

Die Rangfolge der mathematischen Operatoren ist:

1. Klammern
2. Abruf von PI, MEM, Variablen
3. Funktionsoperationen in Bezug auf das folgende Argument
4. Exponentiation
5. Vorzeichen +, -
6. Multiplikation, Division
7. Addition, Subtraktion
8. Vergleichsoperationen

2.2.1.6 Textausdrücke

Textausdrücke sind Bestandteil der BASIC-Sprache. Sie können 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, wobei die Anführungszeichen nicht Bestandteil der Textkonstante sind.

Beispiel:

[CL] Eingabe

```
"SHARP"
```

[ENTER]

Ausgabe

```
SHARP
```

Textvariablen können bis zu sieben Zeichen enthalten. Ober die DIM-Anweisung lassen sich Textvariablen bis zu 80 Zeichen erzeugen.

Beispiel:

Eingabe

```
A$ = "SHARP"
```

[ENTER]

Ausgabe

```
SHARP
```

Texte können mit Hilfe des "+"-Zeichens aneinandergefügt werden.

Beispiel:

Eingabe

```
A$ = "POCKET"
```

[ENTER]

Ausgabe

```
POCKET
```

Eingabe

```
A$ + "␣COMPUTER"
```
ENTER

Ausgabe

```
POCKET COMPUTER
```

Hinweis:

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

2.2.1.7 Logische Vergleichsausdrücke

Der **PC-1430** kennt folgende Vergleichsausdrücke:

Mathematisches Symbol	BASIC	Deutsche Sprechweise
<	<	kleiner als
≦	<=	kleiner gleich
=	=	gleich
≧	>=	größer gleich
>	>	größer als
≠	<>	ungleich

Mit diesen Operatoren können sowohl zwei numerische als auch zwei Textausdrücke miteinander verglichen werden. Ist die Vergleichsaussage richtig, liefert der Rechner das Ergebnis "1", ist sie falsch, ist das Ergebnis "0". Im Zusammenhang mit der BASIC-Anweisung IF erhalten diese Vergleichsoperatoren eine besondere Bedeutung.

Beispiele:

Eingabe

CL
```
2 < 1
```
ENTER

Ausgabe

```
                    0.
```

b) Eingabe

[CL] `1 <= 2` ENTER

Ausgabe

```
1.
```

c) Eingabe

CL `SIN1<.7` ENTER

Ausgabe

```
Ø.
```
(im RAD-Mode)

```
1.
```
(im DEG- und GRAD-Mode)

d) Eingabe

CL `"WERNER" > "ANTON"` ENTER

Ausgabe

```
1.
```

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

Hinweis:

Da das Ergebnis eines logischen Ausdrucks eine Zahl ist, kann dieses als numerische Variable gespeichert und weiterverarbeitet werden.

2.2.2 SPRACHELEMENTE

2.2.2.1 Numerische Konstante

Eine numerische Konstante kann sein:

- eine ganze Zahl (positiv oder negativ)
- eine Dezimalzahl
- eine Zahl in wissenschaftlicher Schreibweise
- eine Sedezimal-(Hexadezimal-)Zahl

Beispiele:

```
513
-2376
11.745
1.23456788E-12
&AA2B
```

2.2.2.2 Textkonstante

Eine Textkonstante ist eine beliebige Zeichenfolge, die durch Anführungszeichen (") begrenzt wird.

Beispiele:

```
"SHARP"
"     "   (Textkonstante enthält Leerzeichen)
""        (Textkonstante der Länge 0)
```

2.2.2.3 Numerische Variable

Unter einer numerischen Variable versteht man den Speicherplatz, der zur Aufnahme des jeweiligen, zugehörigen Zahlenwertes bereitsteht.

Der Variablenname setzt sich aus einem Buchstaben oder einem Buchstaben mit einem in Klammern folgenden Indexwert zusammen. Dabei kann der Index ebenfalls durch eine Variable dargestellt werden.

Beispiele:

```
A
A(27)
B(1)
A(Z)
AB
A1
XY(2,4)
```

2.2.2.4 Textvariable

Unter einer Textvariable versteht man den Speicherplatz, der zur Aufnahme der jeweiligen, zugehörigen Zeichenfolge bereitsteht.

Der Name der Textvariable ist im Prinzip gleich aufgebaut wie der der numerischen Variable mit einem zusätzlichen "$"-Zeichen hinter dem Buchstaben.

Beispiele:

```
A$
B$(2)
C$(Z)
```

2.2.2.5 Numerische Funktionen, Textfunktionen

Eine numerische Funktion setzt sich aus einem Operator und einem oder mehreren Parametern zusammen. 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.

Das Ergebnis einer numerischen Funktion ist ein Zahlenwert, das einer Textfunktion ein Zeichen bzw. eine Zeichenfolge.

Beispiele:

LN60	numerische Funktion; natürlicher Logarithmus des numerischen Parameters 60
MID$("SHARP",2,2)	Textfunktion; Ergebnis ist die Textkonstante "HA".

Hinweis:

Funktionsoperatoren haben eine höhere Priorität als andere Operatoren.

Beispiel:

```
SIN A + B = (SIN A) + B
```

Soll der Sinus der Summe (A + B) gebildet werden, so müssen Klammern gesetzt werden:

```
SIN (A + B)
```

2.2.2.6 Numerische Ausdrücke

Ein numerischer Ausdruck setzt sich aus einer numerischen Konstanten, Variablen, numerischen Funktion oder deren Verknüpfung durch die arithmetischen Operatoren +, −, *, /, ^ und die Zusammenfassung durch Klammern zusammen.

Beispiele:

```
15
(-8)
SIN 45
A + B/C
(C*(A*X+B)*5/(D*C))+10
```

2.2.2.7 Textausdrücke

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

Beispiele:

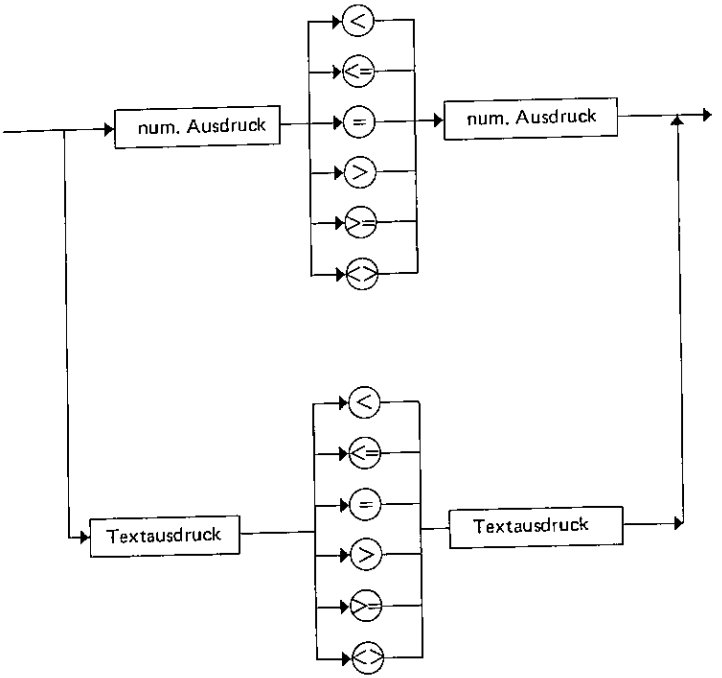
```
"A" + MID$ ("SHARP", 2, 2)
A$ + "PC"
```

2.2.2.8 Logische Vergleichsausdrücke

Ein logischer Vergleichsausdruck ist der Vergleich zweier Ausdrücke (numerisch oder Text) durch die Operatoren:

>, >=, <, <=, <>

Syntax:



Ist die Vergleichsaussage richtig, so ist das Ergebnis des logischen Ausdrucks "1", ist sie falsch, ist das Ergebnis "0".

Werden Textausdrücke miteinander verglichen, so werden die Textausdrücke auf lexikographische Reihenfolge geprüft.

Werden zwei Textausdrücke mit unterschiedlicher Länge verglichen, so werden beim Vergleich die fehlenden Stellen des kürzeren Ausdrucks mit dem ASCII-Zeichen Null aufgefüllt.

Logische Ausdrücke werden in der BASIC-Anweisung IF verwendet.

Beispiele:

	Ergebnis
1 < 2	1
1 + 2 + 3 < 2 + 3 + 4	1
1 >= 2	0
1 < 2 < 3	1
"ANTON" < "WERNER"	1
"ANTON" > "ANTON 1"	0 ("1" ist größer als ASCII-Null)

2.2.3 VARIABLE

Variablen sind Bezeichner für Größen, deren Wert erst im Laufe des Programmablaufs festgelegt werden. Sie werden beim Rechner durch Speicherplätze realisiert, denen man unterschiedliche Daten durch Einlesen oder durch Auswerten eines bestimmten Ausdrucks zuordnen kann. Je nachdem, ob es sich bei der Variablen um einen numerischen oder um einen Textausdruck handelt, nennt man solche Variablen numerische Variablen oder Textvariablen.

Zur Unterscheidung zwischen numerischen und Textvariablen muß jeder Name einer Textvariablen mit dem "$"-Zeichen enden. Beide Typen können als einfache oder indizierte Variable geschrieben werden. Zur leichteren Identifizierung erhalten sie einen Variablennamen als symbolische Adresse für den Speicherplatz. Mit Variablen lassen sich Gesetzmäßigkeiten unabhängig von ihrem jeweiligen Wert allgemein ausdrücken.

Im **PC-1430** stehen folgende zwei verschiedene Speicherbereiche für Variablen zur Verfügung (s. Abb. Anhang H, S. 230).

a) Standardvariablenspeicher
b) Feldvariablenspeicher

Der Standardvariablenspeicher ist ein begrenzter Speicherbereich und dient ausschließlich zur Aufnahme 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 zur Verfügung:

1. Standardvariablen
   a) einfache numerische Variable
   b) einfache Textvariable
   c) indizierte numerische Variable
   d) indizierte Textvariable

2. Feldvariablen
   a)indizierte eindimensionale numerische Feldvariable (Vektor)
   b)indizierte eindimensionale Textfeldvariable (Vektor)
   c)indizierte zweidimensionale numerische Feldvariable (Matrizen)
   d)indizierte zweidimensionale Textfeldvariable (Matrizen)

2.2.3.1 Standardvariable

Für die Standardvariablen stehen 26 reservierte Speicherbereiche zur Verfügung, die wahlweise als numerische oder als Textvariable belegt werden können.

Den Standardvariablen können also wahlweise numerische oder Textwerte zugeordnet werden. Bei der Wertabfrage der Standardvariablen muß der Variablenname dem Inhalt dieser Variablen entsprechen. Wird z.B. eine numerische Variable als Textvariable aufgerufen, und umgekehrt, so wird am Display die Fehlermeldung ERROR 9 angezeigt.

Standardvariable:

```
A = A$ = A(1)  = A$(1)
B = B$ = A(2)  = A$(2)
C = C$ = A(3)  = A$(3)
D = D$ = A(4)  = A$(4)
E = E$ = A(5)  = A$(5)
F = F$ = A(6)  = A$(6)
G = G$ = A(7)  = A$(7)
H = H$ = A(8)  = A$(8)
I = I$ = A(9)  = A$(9)
J = J$ = A(10) = A$(10)
K = K$ = A(11) = A$(11)
L = L$ = A(12) = A$(12)
M = M$ = A(13) = A$(13)
N = N$ = A(14) = A$(14)
O = O$ = A(15) = A$(15)
P = P$ = A(16) = A$(16)
Q = Q$ = A(17) = A$(17)
R = R$ = A(18) = A$(18)
S = S$ = A(19) = A$(19)
T = T$ = A(20) = A$(20)
U = U$ = A(21) = A$(21)
V = V$ = A(22) = A$(22)
W = W$ = A(23) = A$(23)
X = X$ = A(24) = A$(24)
Y = Y$ = A(25) = A$(25)
Z = Z$ = A(26) = A$(26)
```

2.2.3.2 Einfache Variable

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

Einer einfachen numerischen Variablen kann man eine Zahl mit maximal 10 Stellen, einen zweistelligen Exponenten und die Vorzeichen zuordnen.

Beispiele:

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

Einer Textvariablen kann man Zeichenfolgen bis zu sieben Zeichen, bestehend aus Buchstaben, Zahlen, Sonderzeichen und Leerstellen, zuweisen. Bei der Wertzuweisung muß der Text durch Anführungszeichen begrenzt werden.

Beispiele:

```
B$="TEXT"
B$="T E X T"
```

2.2.3.3 Indizierte Variable

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

Die indizierten Standardvariablen sind äquivalent zu den oben beschriebenen einfachen Standardvariablen. So entspricht z.B. die einfache Variable A der indizierten Variablen A(1), die einfache Variable B der indizierten Variablen A(2) usw.
Die genaue Zuordnung ist aus der Tabelle auf Seite 84 ersichtlich.

Die indirekte Adressierung (Indizierung) einer Variablen gestattet es, den Namen und die Adresse einer Variablen in Abhängigkeit vom Wert einer anderen Variablen, der z.B. ein Rechenergebnis sein kann, festzulegen. Außerdem kann der Index einer Variablen auch das Ergebnis eines numerischen Ausdrucks sein.

Beispiele:

```
LET A(A) = 1243
```

Hiermit 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.

```
LET A(B/5) = 789
```

Hierbei 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, und somit wird der Variablen A(26) = Z ein Wert von 789 zugeordnet.

```
LET A$(A) ="TEXT"
```

Der Wert der numerischen Variablen A definiert den Index der Textvariablen A$. Dabei muß der Wert der Variablen größer als 1 sein, da die Variable A$(0) nicht existiert und die Variable A$(1) der Variablen A$ entspricht und die beiden, wie bereits erklärt, nicht gleichzeitig definiert werden können.

2.2.3.4 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 = 0 bis 255)
d) Zweidimensionale Feldvariable A(n,n) bzw. A$(n,n)
   (n = 0 bis 255)

definiert werden.

Dabei gelten folgende Grundsätze:

1) Die Variable A ohne Dimensionierung

1. Diesen A-Variablen können nur numerische Werte mit zehnstelliger Mantisse, zweistelligem Exponenten und einem Vorzeichen bzw. Textwerte mit max. 7 Zeichen zugewiesen werden. Sie können nur alternativ als numerische oder Textvariablen definiert werden.

2. Der Variablenname A(0) ist unzulässig.

3. Für die indizierte Variablen A(n), (n = 1 bis 26) = Standardvariablen A bis Z, wird kein Speicherplatz im Hauptspeicher reserviert.

4. Für die indizierte Variable A(n), (n = 27 bis 255) = eindimensionaler Feldvariable, wird ein Speicherplatz im Hauptspeicher reserviert. 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 Elementeanzahl automatisch auf den Wert begrenzt, den er vor der Dimensionierung eines weiteren Vektors bzw. einer Matrix hatte.

Beispiel:

```
 5: A(27) = 27
10: DIM B (10)
15: DIM A (40)        oder 15: A(40) = 40
```

Nach dem Starten des Programms wird die Fehlermeldung ERROR 3 angezeigt. Der A-Vektor wurde durch die Dimensionierung des B-Vektors auf ein Element, nämlich A(27), begrenzt.
Durch den Anruf von A(40) in Zeile 15 müßte ein zweiter A-Vektor definiert werden, was aber unzulässig ist.

6. Die Variablen A(n) bzw. A$(n) (n = 1 bis 26) können nicht gelöscht werden. Mit den Kommandos NEW/NEWØ bzw. CLEAR wird ihr Wert auf 0 (Null) bzw. auf das ASCII-Zeichen Null gesetzt.
Variablen A(n) bzw. A$(n) werden durch das Kommando RUN gelöscht.

2) Die Variable A mit Dimensionierung

1. Die dimensionierte Variable A wird behandelt wie in den Kapiteln 2.2.3.5. - 2.2.3.7. beschrieben ist.

2. Nach einer Dimensionierung der Variablen A kann auf die Standardvariablen nur noch direkt zugegriffen werden.
Nach z. B. DIM A(26) entspricht A(1) nicht mehr der Standardvariablen A und A(26) nicht mehr der Standardvariablen Z.
Das Element A(1) und die Standardvariable A haben nun verschiedene Speicherplätze.

3. Wird nach dem Aufruf einer undimensionierten Standardvariablen, z. B. A(2)=B eine Dimensionierung, z. B. DIM A(30) durchgeführt, wird ERROR 3 angezeigt.

2.2.3.5 Feldvariable

Feldvariablen können als eindimensionale Felder (Vektoren) oder als zweidimensionale Felder (Matrizen) definiert werden, die wiederum unterschiedlich viele Elemente haben können.
Die Feldvariablen werden im Hauptspeicher abgelegt und müssen vor ihrem Aufruf wegen ihrer flexiblen Länge dimensioniert werden. Weitere Einzelheiten siehe unter dem BASIC-Befehl DIM (Abschnitt 2.5.7).

Der Speicherplatzbedarf für eine Feldvariable setzt sich zusammen aus:

7 BYTES für den Variablennamen
8 BYTES für ein numerisches Element
16 BYTES für ein Textelement im Standardformat

2.2.3.6 Numerische Feldvariable

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

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

Der Vektorenname kann aus einem oder aus zwei Zeichen bestehen. Das erste Zeichen muß ein Buchstabe sein, das zweite Zeichen kann ein Buchstabe oder eine Ziffer sein.

Buchstabenkombinationen, die ein BASIC-Schlüsselwort oder eine Funktion ergeben, sind als Variablenname nicht erlaubt, z. B. LN, ON, PI etc.

Beispiel:

```
B(0)
C1(3)
DF(80)
```

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

Der Matrizenname kann aus einem oder aus zwei Zeichen bestehen. Das erste Zeichen muß ein Buchstabe sein, das zweite Zeichen kann ein Buchstabe oder eine Ziffer sein.

Buchstabenkombinationen, die ein BASIC-Schlüsselwort oder eine Funktion ergeben, sind als Variablenname nicht erlaubt, z. B. LN, ON, PI etc.

Beispiel:

D(2,4)
E1(1,1)
YZ(0,0)

Insgesamt stehen für Vektoren und Matrizen folgende Namen zur Verfügung:

Alle Einzelbuchstaben oder alle Zweierkombinationen aus den Buchstaben A - Z und den Ziffern 0 - 9, wobei das erste Zeichen ein Buchstabe sein muß.

Die Indizes für die Elemente liegen im Intervall 0 - 255.

Buchstabenkombinationen, die ein BASIC-Schlüsselwort oder eine Funktion ergeben, sind als Variablenname nicht erlaubt, z. B. LN, ON, PI etc.

Der Index für die Vektoren- und Matrizenelemente kann als numerische Konstante oder durch eine Variable definiert werden.

Beispiele:

B(6) bzw. B(A)
H1(3,5) bzw. H1(E,B)

Der Index wird durch den jeweiligen Wert der Variablen ausgedrückt.

Vektoren und Matrizen dürfen nicht mit dem gleichen Feldnamen und nicht zweimal definiert werden. Die Namen B(n) und B(n,n) sind also nicht gleichzeitig zulässig.

Hinweis:

Bei Verwendung der Feldvariablen 'A' beachten Sie das Kapitel 2.2.3.4.

2.2.3.7 Textfeldvariablen

Textfeldvariablen können wie die numerischen Feldvariablen als Vektor oder als Matrix definiert werden. Jedem Element der Vektoren oder Matrizen können im Standardformat Texte mit maximal 16 Zeichen zugeordnet werden. Durch eine entsprechende Dimensionierung (siehe DIM-Anweisung Seite 97) kann die Größe der Elemente zwischen 1 und 80 Zeichen frei festgelegt werden.

Beispiel: DIM B$(10)*4

In diesem Beispiel können für die Elemente des B$- Vektors (B$(0) bis B$(10)) jeweils nur 4 Zeichen eingegeben werden.

Insgesamt stehen für die Vektoren und Matrizen folgende Feldnamen zur Verfügung:

Alle Einzelbuchstaben oder alle Zweierkombinationen aus den Buchstaben A - Z und aus den Ziffern 0 – 9, wobei das erste Zeichen ein Buchstabe sein muß.

Buchstabenkombinationen, die ein BASIC-Schlüsselwort oder eine Funktion ergeben, sind als Variablenname nicht erlaubt, z. B. LN, ON, PI etc.

Die Indizes für die Elemente liegen im Intervall 0 - 255.

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

Beispiele:

```
B$(0)
A5$(0)*20
Z$(80)*1
```

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

Beispiele:

```
D$(2,4)
Z1$(5,10)*12
```

Die Indizes für die Vektoren- und Matrizenelemente können als numerische Konstante oder als numerische Variable definiert sein, wobei der Index durch den Wert der jeweiligen Variablen bezeichnet wird.

Beispiele:

```
C$(A)
AE$(E,F)
```

Vektoren und Matrizen dürfen nicht mit dem gleichen Feldnamen und nicht zweimal definiert werden. Die Namen B$(n) und B$(n,n) sind also nicht gleichzeitig zulässig.

Hinweis:

Bei Verwendung der Texfeldvariablen 'A' beachten Sie das Kapitel 2.2.3.4.

2.2.4 ARITHMETISCHE FUNKTIONEN

Der **Computer** bietet eine große Anzahl von arithmetischen Standard-Funktionen. Hierzu gehören:

Trigonometrische und ihre Umkehrfunktionen

Natürlicher und dekadischer Logarithmus und Umkehrfunktionen
Polarkoordinatenberechnungen

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

Fast alle Funktionsnamen können über die Funktionstasten des Rechnerteils eingegeben werden:

Funktionsname		Funktion
ABS		Absolutwert
ACS	SHIFT ACS	Arcuscosinus
ASN	SHIFT ASN	Arcussinus
ATN	SHIFT ATN	Arcustangens
COS	cos	Cosinus
CUB	CUB	Kubikzahl
CUR	SHIFT CUR	Kubikwurzel
DEG	DEG	Sexagesimalumrechnung
DEGREE		Winkeleinheit Grad
DMS	SHIFT DMS	Sexagesimalumrechnung
EXP	SHIFT EXP	Exponentialumrechnung
FACT	SHIFT FACT	Fakultät
GRAD		Winkeleinheit Neugrad
HEX→DEC	SHIFT &	Sedezimalumrechnung
INT		Ganzzahlfunktion
LN	LN	Natürlicher Logarithmus
LOG	LOG	Dekadischer Logarithmus
MEM		Freier Speicherplatz
PI	π	Kreiskonstante PI
POL	SHIFT POL	Polarkoordinatenumrechnung
RADIAN		Winkeleinheit
RANDOM		Zufallsgeneratoranfangswert
RCP	RCP	Reziprokwert
REC	SHIFT REC	Polarkoordinatenumrechnung
RND		Zufallszahl
SGN		Vorzeichen
SIN	SIN	Sinus
SQU	SQU	Quadratzahl
SQR	SQR	Quadratwurzel

SQR	SHIFT √	Wurzelfunktion
TAN	TAN	Tangens
TEN	SHIFT TEN	Exponentialfunktion 10^x
^	SHIFT ^	Potenzfunktion

2.2.5 TEXTFUNKTIONEN

Mit dem **Computer** 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 Ergebnisse der Textfunktionen sind entweder Zeichenketten, die Textvariablen zugewiesen oder in Textausdrücken weiterverarbeitet werden können, oder numerische Werte, die sich in numerischen Ausdrücken verarbeiten oder numerischen Variablen zuordnen lassen.

Folgende Textfunktionen stehen Ihnen mit dem **Computer** zur Verfügung:

MID$	Entnimmt einer Zeichenfolge eine spezifizierte Zeichenanzahl aus der Mitte.
VAL	Wandelt eine als Zeichenfolge eingegebene Zahl in ihren numerischen Wert um.
LEN	Berechnet die Anzahl der Zeichen eines Textausdruckes.

2.2.6 PROGRAMMIEREN IN BASIC

Der **Computer** verwendet die weit verbreitete Programmiersprache BASIC (Beginner's All-Purpose Symbolic Instruction Code). Diese Programmiersprache wurde 1960 in Dartmouth College entwickelt und hat sich bis heute insbesondere bei den Mikrocomputern durchgesetzt.

BASIC ist im Gegensatz zu den anderen Programmiersprachen einfach und leicht verständlich.

BASIC 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 diese Programmiersprache maschinenunabhängig.

2.2.6.1 BASIC - Übersicht

Wie jede natürliche Sprache hat auch die formale Computersprache BASIC eine Grammatik und einen zugehörigen Zeichensatz.
Letzterer setzt sich aus folgenden Zeichen zusammen:

Buchstaben:	A, B, ... , Z
Zahlen:	0, 1, ... , 9
Sonderzeichen:	( ) . ; , : + - * / $ & % # @ ! ? < > = ^

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

BASIC kennt folgende Sprachelemente:

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

Diese Sprachelemente werden zu Programmsätzen zusammengefügt, die vom Rechner der angegebenen Reihenfolge nach abgeleitet werden.

2.2.6.2 Programmerstellung

Die Programmerstellung gliedert sich im wesentlichen in drei Schritte:

PROBLEMANALYSE

UMSETZUNG IN DIE PROGRAMMIERSPRACHE
UND EINGABE IN DEN COMPUTER

TESTEN DES PROGRAMMS

Problemanalyse:

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

Umsetzung in die Programmiersprache und Eingabe in den Computer:

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

Testen des Programms:

Nur sehr selten wird das umgesetzte und eingegebene Programm auf Anhieb fehlerfrei arbeiten. Dabei können die verschiedenartigsten Fehler auftreten:

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

Der **Computer** bietet einige Möglichkeiten auftretende Fehler zu finden und zu beheben (siehe 2.2.6.8 Fehlermeldungen/Fehlersuche).

2.2.6.3 Programmaufbau

Bei der Obertragung der Problemanalyse in ein BASIC-Programm sind die im folgenden aufgeführten Regeln zu beachten:

- Ein vollständiges BASIC-Programm besteht aus einer Reihe von Anweisungen, die in der Reihenfolge angeordnet sein müssen, in der sie später ausgeführt werden sollen. Eine Ausnahme hierfür sind die GOTO- und GOSUB-Anweisungen, die das Programm an eine festgelegte Stelle verzweigen können.

- Die Programmierung erfolgt zeilenweise, wobei jede Programmzeile eine oder mehrere durch Doppelpunkt getrennte Programmanweisungen enthalten kann.

- 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, wovon jeweils 16 Zeichen auf dem Display angezeigt werden. Die restlichen Zeichen einer Zeile können mit den (Cursor-)Tasten ► oder ◄ sichtbar gemacht werden.

  Hinweis: BASIC-Schlüsselwörter wie LET, SIN, COS, PRINT usw. werden nach Betätigung der ENTER-Taste intern abgekürzt. Bei der Eingabe einer sehr langen Programmzeile kann man vorerst nur 79 Zeichen eingeben. Nach Drücken der ENTER-Taste werden die Schlüsselwörter abgekürzt, und man kann zusätzlich noch einige Zeichen eingeben. Die Eingabe muß wieder mit ENTER abgeschlossen werden.

- Jede Zeile muß mit einer positiven ganzen Zahl (der Zeilennummer) beginnen. Eine Zeilennummer darf nur einmal im Programm verwendet werden. Die Programmzeilen werden vom Rechner in aufsteigender Reihenfolge ausgeführt, wenn nicht durch Steueranweisungen (z.B.GOTO) eine andere Reihenfolge festgelegt wird.
  Die Nummerierung muß nicht lückenlos sein, es ist sogar zweckmäßig, z.B. Zehnerschritte zu wählen, da damit die Möglichkeit geboten ist, nachträglich noch Programmzeilen einzufügen.
  Die Zeilennummern dürfen im Bereich von 1 bis 65279 gewählt werden.

2.2.6.4 Eingabe eines Programms

Die Programmeingabe erfolgt im PRO-Mode. Ein sich eventuell noch im Speicher befindliches Programm kann mit der NEW- oder NEWØ-Anweisung gelöscht werden.

Beispiel einer Programmeingabe:

```
NEW

10 A = 15
20 B = 3
30 C = A + B
40 PRINT C
50 END
```

Jede Programmzeile muß mit [ENTER] abgeschlossen werden. Der Rechner fügt dann einen Doppelpunkt zwischen die Zeilennummer und der ersten Anweisung ein.

Beispiel:

10 A = 15 [ENTER]

Anzeige:

10:A = 15

Oberprüfen des Programms, Editieren und Auflisten

Ist die Programmeingabe abgeschlossen, so kann man im PRO-Mode den Inhalt der einzelnen Zeilen noch einmal überprüfen.

Betätigt man die [↑]-Taste, wird die vorhergehende Programmzeile angezeigt.
Betätigt man die [↓]-Taste, wird die folgende Programmzeile angezeigt.

Mit der LIST-Anweisung kann man genau spezifizierte Zeilen zur Anzeige bringen.

Beispiele:

```
LIST          Die erste Zeile eines Programms wird angezeigt.
LIST 100      Die Programmzeile 100 wird angezeigt.
```

Mit der LLIST-Anweisung kann das Programm über den Drucker (Option CE-126P) auf Papier 'gelistet' werden.

Beispiele:

```
LLIST         Das gesamte Programm wird ausgedruckt.
LLIST 100     Die Programmzeile 100 wird ausgedruckt.
LLIST 100,200 Die Programmzeilen von 100 bis 200 werden ausgedruckt.
LLIST 100,    Das Programm wird ab Programmzeile 100 ausgedruckt.
LLIST ,100    Das Programm wird bis Programmzeile 100 ausgedruckt.
```

Beispiel:

```
10 A = 15
20 B = 3
30 C = A + B
40 PRINT C
50 END
```

Durch Eingabe von LIST 30 (nur im PRO-Mode möglich) wird am Display folgendes angezeigt:

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

Die Zeile 40 kann jetzt durch Drücken der [↓] -Taste angezeigt werden.

```
40:PRINT C
```

2.2.6.5 Korrektur einer Zeile

Die Korrektur einer Zeile (nur im PRO-Mode) ist sehr einfach und kann auf zwei verschiedene Arten erfolgen:

1. Oberschreiben der Zeile:
   Man gibt die neue Zeile mit derselben Zeilennummer ein. Die alte Version der Programmzeile wird dadurch überschrieben.

2. Editieren:
   Man bringt die zu editierende Zeile mit LIST zur Anzeige und drückt eine der beiden Cursor-Tasten ([←] oder [→]). Damit wird der Doppelpunkt zwischen der Zeilennummer und der ersten Programmanweisung unterdrückt und die Zeile zur Korrektur freigegeben. Jetzt können Sie mit Hilfe der [SHIFT] [DEL ←] -oder der [INS →] -Taste Zeichen in der Zeile löschen oder einfügen. Auch ein Oberschreiben einzelner Zeichen ist möglich.

Beispiel:

In dem oben angeführten Beispiel soll die Zahl 15 durch die Zahl 17 ersetzt werden. Hierzu geben Sie ein:

LIST 10 [ENTER] (Zeile 10 anzeigen)

[→] (Beginn Änderung)

[→] [→] [→] (Cursor auf die '5')

7 [ENTER] ('5' durch '7' ersetzen)

2.2.6.6 Löschen einer Zeile

Das Löschen einer Programmzeile erfolgt im PRO-Mode. Eine Zeile wird ersatzlos gelöscht, wenn man nur die Zeilennummer eingibt und die [ENTER]-Taste drückt.

Beispiel:

```
10 A = 15
20 B = 3
30 C = A + B
40 PRINT C
50 END
```

Eingabe:

```
10 [ENTER]
20 [ENTER]
```

Listet man nun das Programm, so sieht man, daß die Zeilen 10 und 20 gelöscht wurden.

2.2.6.7 Programmausführung

Die Programmausführung kann nur im RUN-Mode erfolgen. Der Programmstart kann durch drei verschiedene Anweisungen erfolgen (siehe RUN/GOTO/Definable Keys).

Nach Eingabe des Startkommandos beginnt der **Computer** mit der Abarbeitung der einzelnen Programmzeilen. Während der Programmausführung ist auf der Anzeige das Wort "BUSY" sichtbar. Dies erlischt entweder am Programmende oder wenn eine Anzeige am Display erfolgt. Solange 'BUSY' aufleuchtet, nimmt der Computer keine Eingaben von der Tastatur an. Nach Beendigung des Programms wird am Display das 'Bereitschaftssymbol' (>) angezeigt.

Zur Unterbrechung einer Programmausführung dient die [BRK] -Taste. Am Display wird folgendes angezeigt:

BREAK IN XX wobei XX die Zeilennummer ist, die vor der Unterbrechung bearbeitet wurde.

Will man nach einer Unterbrechung das Programm fortsetzen, so genügt es, die CONT(inue)-Anweisung einzugeben:

```
CONT
```

Die Programmausführung wird fortgesetzt.

2.2.6.8 Fehlermeldung/Fehlersuche

Tritt während der Programmausführung ein Fehler auf, so wird dieser vom **Computer** erkannt und gemeldet. Die Fehlermeldung wird am Display in folgender Form angezeigt:

ERROR Fehlerkode IN Zeilennummer

Die Liste der Fehlermeldungen und ihre Erklärung finden Sie numerisch geordnet im Anhang.

Gleichzeitig mit dem Fehlerkode wird die Nummer der Zeile, in der der Fehler auftrat, angezeigt. Diese Zeile kann nun auch im RUN-Mode am Display zur Anzeige gebracht werden. Löschen Sie die Fehlermeldung mit der [CL]-Taste und drücken Sie die Taste [↑] . Die Zeile erscheint auf der Anzeige, und der Cursor blinkt auf dem fehlerhaften Sprachelement. Will man die Zeile korrigieren, muß man in den PRO-Mode umschalten und wie in Abschnitt 2.2.6.5 beschrieben vorgehen.

Beispiel:

```
10 A = 2
20 B = 10
30 C = B/A
40 PRINT C,
50 END
```

Bei der Ausführung diese Programm erscheint:

ERROR 1 IN 40

Die PRINT-Liste in Zeile 40 ist unvollständig. Schreiben Sie hinter das Komma ein A.

```
40 PRINT C,A
```

Das Programm ist jetzt fehlerfrei.

2.2.7 BASIC – BEFEHLSVORRAT

Im folgenden Kapitel wird der BASIC-Sprachumfang des **Computers** ausführlich beschrieben. Dabei sind für jeden Befehl kurz die Funktion, die Syntax, eventuelle Bemerkungen und ein Beispiel angegeben.

Hinweis zur Syntax:

Die Bedeutung des jeweiligen Sprachelements wird in spitzen Klammern angegeben.

2.2.7.1 Fest vorprogrammierte Schlüsselwörter und arithmetische Funktionen

Die am häufigsten verwendeten BASIC-Anweisungen können durch die Tastenkombination SHIFT und einen Buchstaben der beiden unteren Tastenreihen abgerufen werden.

Außerdem lassen sich fast alle arithmetischen Funktionen direkt über die Funktionstasten auf der rechten Seite des Rechners direkt ins BASIC übernehmen.

Die Zuordnung der Anweisungen ist aus der folgenden Tabelle zu entnehmen:

Tastenbetätigung		BASIC-Anweisung
SHIFT	A	INPUT
SHIFT	S	IF
SHIFT	D	THEN
SHIFT	F	GOTO
SHIFT	G	FOR
SHIFT	H	TO
SHIFT	J	STEP
SHIFT	K	NEXT
SHIFT	L	LIST
SHIFT	=	RUN
SHIFT	Z	PRINT
SHIFT	X	USING
SHIFT	C	GOSUB
SHIFT	V	RETURN
SHIFT	B	DIM
SHIFT	N	END
SHIFT	M	CSAVE
SHIFT	SPC	CLOAD

Hinweise:

1. Da diese Begriffe im Rechner kodiert als Schlüsselwörter für BASIC-Anweisungen gespeichert werden, ist es z.B. nicht möglich,

   aus [L] [SHIFT] [L]

   ein LLIST zusammenzusetzen.

2. Eine weitere Möglichkeit zur Reduzierung der Eingabearbeit ist durch die Verwendung von Abkürzungen gegeben. Eine Liste der Abkürzungen finden Sie im Anhang.

Die Zuordnung der arithmetischen Befehle ist aus der in Abschnitt 2.2.4 dargestellten Tabelle zu entnehmen.

ABS

Funktion:

Ermittelt den Absolutbetrag eines numerischen Ausdrucks.

Syntax:

ABS numerischer Ausdruck

Bemerkungen:

Der Absolutwert ist der Wert eines numerischen Ausdrucks ohne Berücksichtigung seines Vorzeichens.

Beispiel:

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

Ausgabe nach RUN:

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

ACS

Funktion:

Berechnet den Arcuscosinus eines numerischen Ausdrucks in der angegebenen Winkeleinheit.

Syntax:

ACS numerischer Ausdruck

Bemerkungen:

Die Berechnung der Arcuscosinusfunktion kann in folgenden drei Winkeleinheiten erfolgen:
DEGREE (0 bis 90°)
GRAD (0 bis 100^g)
RADIAN (0 bis PI/2)
Die gewählte Winkeleinheit wird am Display angezeigt.

Beispiel:

```
10 DEGREE : INPUT X
20 LET Y = ACS X
30 PRINT"X=";X;" ";"ACS X=";Y
```

Ausgabe nach RUN:

```
?                          1   ENTER

X=1. ACS X=0.
```

ASN

Funktion:

Berechnet den Arcussinus eines numerischen Ausdrucks in der angegebenen Winkeleinheit.

Syntax:

ASN numerischer Ausdruck

Bemerkungen:

Die Berechnung der Arcussinusfunktion kann in folgenden drei Winkeleinheiten erfolgen:
DEGREE (0 bis 90°)
GRAD (0 bis 100^{g})
RADIAN (0 bis PI/2)
Die gewählte Winkeleinheit wird am Display angezeigt.

Beispiel:

```
10 DEGREE : INPUT X
20 LET Y = ASN X
30 PRINT"X=";X;" ";"ASN X=";Y
```

Ausgabe nach RUN:

```
?                          1   ENTER

X=1. ASN X=90.
```

ATN

Funktion:

Berechnet den Arcustangens eines numerischen Ausdrucks in der angegebenen Winkeleinheit.

Syntax:

ATN numerischer Ausdruck

Bemerkungen:

Die Berechnung der Arcustangensfunktion kann in folgenden drei Winkeleinheiten erfolgen:
DEGREE (0 bis 90°)
GRAD (0 bis 100^{g})
RADIAN (0 bis PI/2)
Die gewählte Winkeleinheit wird am Display angezeigt.

Beispiel:

```
10 DEGREE : INPUT X
20 LET Y = ATN X
30 PRINT"X=";X;" ";"ATN X=";Y
```

Ausgabe nach RUN:

```
?                          1   ENTER

X=1. ATN X=45.
```

CLEAR

Funktion:

Löscht alle Variablen und Felder aus dem Hauptspeicher und setzt die Standardvariablen auf 0 (Null).

Syntax:

CLEAR

Bemerkungen:

Das Programm bleibt im Gegensatz zum NEW-Befehl erhalten.

Beispiel:

```
10 LET X=17
20 WAIT 59: PRINT "X="; X
30 CLEAR
40 WAIT: PRINT "X="; X
50 END
```

Ausgabe nach RUN:

```
X=17.
X=0.
```

CLOAD (nur mit Option CE-126P)

Funktion:

Laden der Programme von Band.

Syntax:

CLOAD
CLOAD Textausdruck

Bemerkungen:

Mit der CLOAD-Anweisung werden auf Band gespeicherte Programme gesucht und geladen. Die CLOAD-Anweisung ist nur als direktes Kommando möglich. Befindet sich ein Programm im Hauptspeicher, so ist dieses vor einer CLOAD-Anweisung zu sichern, da die Anweisung den gesamten Hauptspeicher löscht, bevor ein neues Programm geladen wird.

Der der CLOAD-Anweisung folgende Textausdruck bezeichnet den Programmnamen. Wird dieser beim Ladevorgang gefunden, so wird das Programm in den Hauptspeicher geladen.

Erfolgt die Eingabe von CLOAD ohne einen Textausdruck (Programmnamen), so wird das nächste auf Band gespeicherte Programm geladen.

Befindet sich das Programm nicht auf der im Recorder befindlichen Cassette, so sucht der Computer auch dann noch nach dem Programmnamen, wenn das Band schon abgelaufen ist. In diesem Fall muß der Ladevorgang mit der BRK -Taste abgebrochen werden.

Tritt während des Ladevorgangs ein Fehler auf, so ist das im Computer befindliche Programm nicht verwendbar. Der Ladevorgang muß wiederholt werden.

Beispiele:

```
CLOAD

CLOAD"PROG.1"
```

CLOAD? (nur mit Option CE-126P)

Funktion:

Vergleich des im Hauptspeicher gespeicherten Programms mit dem auf Band gespeicherten Programm.

Syntax:

CLOAD?
CLOAD? Textausdruck

Bemerkungen:

Mit der CLOAD?-Anweisung wird das im Hauptspeicher gespeicherte Programm mit dem auf Band gespeicherten Programm verglichen. Tritt dabei ein Fehler auf, so wird am Display die Fehlermeldung ERROR 8 angezeigt.

Der Ablauf der CLOAD?-Anweisung ist identisch mit dem der CLOAD-Anweisung.

Ein eventuell verwendetes PASS-Wort wird mit CLOAD? nicht geprüft.

Beispiele:

```
CLOAD?

CLOAD? "PROG.1"
```

CONT

Funktion:

Die Programmausführung wird nach einer Unterbrechung durch STOP oder durch Drücken der BREAK-Taste fortgesetzt.

Syntax:

CONT

Bemerkungen:

Alle Zustände des Rechners (FOR...NEXT, GOSUB, DIM usw.) bleiben erhalten. Dadurch ist es möglich, einen Programmablauf zu unterbrechen (STOP oder BREAK), um die aktuellen Werte von Variablen zu überprüfen und gegebenenfalls zu verändern. Die Programmfortsetzung erfolgt mit der CONT-Anweisung.

Beispiel:

```
 5 A=0
10 FOR I=1 TO 20
20 A=A+1
30 WAIT 59: PRINT A
40 NEXT I
50 END
```

Ausgabe nach RUN:

```
1.
2.
3.

BREAK-Taste drücken

BREAK IN XX          XX...Zeilennummer, in der der Programmablauf
                          unterbrochen wurde.

CONT
4.
5.
6.
.
.
.
20.
```

COS

Funktion:

Berechnet den Cosinus eines numerischen Ausdrucks in der angegebenen Winkeleinheit.

Syntax:

COS numerischer Ausdruck

Bemerkungen:

Die Berechnung der Cosinusfunktion kann in folgenden drei Winkeleinheiten erfolgen:
DEGREE (0 bis 90°)
GRAD (0 bis 100^{g})
RADIAN (0 bis PI/2)
Die gewahlte Winkeleinheit wird am Display angezeigt.

Beispiel:

```
10 DEGREE : INPUT X
20 LET Y = COS X
30 PRINT Y
```

Ausgabe nach RUN:

```
?                    1  ENTER

9.998476952E-01
```

CSAVE (nur mit Option CE-126P)

Funktion:

Speichern eines Programms auf Band.

Syntax:

CSAVE
CSAVE "Programmname"
CSAVE ,"PASS-Wort"
CSAVE "Programmname","PASS-Wort"

Bemerkungen:

Mit der CSAVE-Anweisung können Programme auf Band gespeichert werden. Bei der CSAVE-Anweisung gibt es folgende vier Eingabemöglichkeiten:

CSAVE Alle im Computer gespeicherten Programme werden auf Band gespeichert.

CSAVE "Programmname" Alle im Computer gespeicherten Programme werden unter dem angegebenen Programmnamen auf Band gespeichert.

CSAVE ,"PASS-Wort" Alle im Computer gespeicherten Programme werden mit dem PASS-Wort auf Band gespeichert. Programme, die mit einem PASS-Wort auf Band gespeichert wurden und wieder geladen werden, können nur nach Eingabe des zugehörigen PASS-Wortes wieder gelistet oder bearbeitet werden.

CSAVE "Programmname","PASS-Wort" Alle im Computer gespeicherten Programme werden unter dem angegebenen Programmnamen mit dem PASS-Wort auf Band gespeichert.

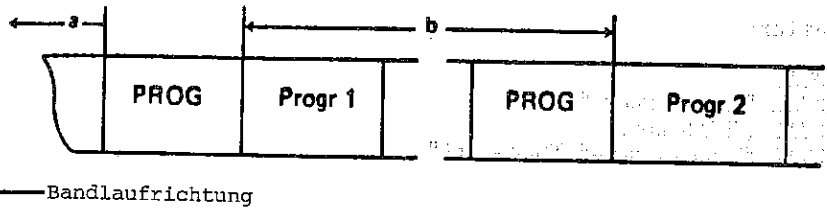
Die CSAVE-Anweisung kann im PRO- und im RUN-Mode eingegeben werden. Je nach Programmlänge dauert der Speichervorgang einige Minuten.

Im Anschluß an das Speichern des Programms sollte man mit der CLOAD?-Anweisung überprüfen, ob nicht, bedingt durch einen Fehler beim Speichervorgang oder durch einen Bandfehler, Informationen verlorengegangen sind.

Hinweise:

1. Wird ein neues Programm so auf dem Band gespeichert, daß es ein anderes Programm auch nur teilweise überlappt, wird die ursprüngliche Information gelöscht. Dies führt zu einem Fehler beim Laden des ursprünglichen Programms.

2. Haben zwei Programme denselben Programmnamen, z.B. "PROG", so lädt der Computer das zuerst aufgefundene Programm in den Speicher. Befindet sich der Tonkopf in bezug auf das Band in der Abbildung im Bereich a, so wird das erste Programm mit dem Namen "PROG" geladen. Befindet er sich jedoch im Bereich b, so wird das zweite Programm geladen.



3. Zweckmäßigerweise sollte für jede Cassette der Programmbibliothek ein Karteiblatt angelegt werden, aus dem die Programmnamen, Hinweise auf Informationstyp (Programm, Daten), Zählerstand und kurze Programmbeschreibung hervorgehen. Damit läßt es sich vermeiden, daß Programme oder Daten nur mühselig und mit sehr großem Aufwand wiedergefunden werden können.

Beispiele:

```
CSAVE

CSAVE "PROG 1"

CSAVE ,"GEHEIM"

CSAVE "PROG 1","GEHEIM"
```

CUR

Funktion:

Berechnet die Kubikwurzel eines numerischen Ausdrucks.

Syntax:

CUR numerischer Ausdruck

Bemerkungen:

-

Beispiel:

```
10 X=27
20 Y=CUR X
30 PRINT"X=";X;" CUR X=";Y
40 END
```

Ausgabe nach RUN:

```
X=27. CUR X=3.
```

DATA

Funktion:

Definiert Datenfelder für die READ-Anweisung.

Syntax:

DATA numerischer Ausdruck
DATA Textausdruck
DATA Liste numerischer Ausdrücke
DATA Liste von Textausdrücken
DATA Liste von numerischen Ausdrücken und Textausdrücken

Bemerkungen:

Die DATA-Anweisung gewinnt ihre Bedeutung erst im Zusammenhang mit der READ-Anweisung.
Mit der DATA-Anweisung werden Daten innerhalb eines Programms gespeichert und zum Abruf mit der READ-Anweisung bereitgestellt. DATA-Anweisungen können in beliebiger Anzahl und an beliebiger Stelle im Programm auftreten und werden während des Programmablaufs übersprungen. Die DATA-Anweisung kann eine beliebige Anzahl von Konstanten enthalten (bis zur maximalen Zeilenlänge), die jeweils durch ein Komma getrennt werden. DATA-Anweisungen können wiederholt von Anfang an gelesen werden, wenn man die RESTORE-Anweisung benutzt.

Hinweis:
Wichtig ist es, daß die Daten-Typen in einer DATA-Anweisung mit den entsprechenden Variablen-Typen in der READ-Anweisung übereinstimmen.

Hinweis:
DATA-Anweisungen in der ersten Programmzeile werden ignoriert. Daher dürfen DATA-Anweisungen erst ab der zweiten Programmzeile vorkommen.

Beispiele:

```
10 DATA 12,13,14
20 DATA "TEXTE","KOELN"
30 DATA 1,"BEI","SPIEL"
40 DATA A + B, X/Y
```

DEGREE/GRAD/RADIAN

Funktion:

Festlegung der Winkeleinheit.

Syntax:

DEGREE
GRAD
RADIAN

Bemerkungen:

Die Winkel werden wie folgt angegeben:

Winkeleinheit	Anzeige	Viertelkreis	Einheit
DEGREE (Grad)	DEG	(0 bis 90°)	Grad
GRAD (Neugrad)	GRAD	(0 bis 100^g)	Gon
RADIAN (Radiant)	RAD	(0 bis PI/2)	rad

Die gewählte Winkeleinheit wird im Display angezeigt.
Die Winkeleinheit kann auch programmgesteuert verändert werden.

Beispiel:

```
10 WAIT 200
20 DEGREE : PRINT SIN 45
30 GRAD : PRINT SIN 50
40 RADIAN : PRINT SIN (PI/4)
```

Ausgabe nach RUN:

```
7.071067812E-01
7.071067812E-01
7.071067812E-01
```

DEG

Funktion:

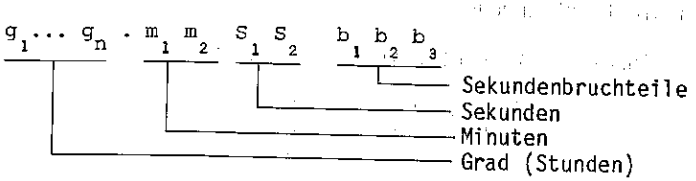
Umrechnung vom Sexagesimalsystem ins Dezimalsystem.

Syntax:

DEG numerische Konstante
DEG numerische Variable

Bemerkungen:

Sexagesimal geteilte Winkel oder Zeiten werden in folgendem Format verarbeitet:



Beispiel:

Umrechnung von 15°24'45'' in Dezimalgrad

```
10 A=DEG 15.2445
20 PRINT A
30 END
```

Ausgabe nach RUN:

```
        15.4125
```

DIM

Funktion:

Dimensionierung von ein- bzw. zweidimensionalen Feldern (Arrays).

Syntax:

DIM Feldname (num. Ausdruck)
DIM Feldname (num. Ausdruck, num. Ausdruck)

DIM Text-Feldname (num. Ausdruck)
DIM Text-Feldname (num. Ausdruck) * num. Ausdruck
DIM Text-Feldname (num. Ausdruck, num. Ausdruck)
DIM Text-Feldname (num. Ausdruck, num. Ausdruck) * num. Ausdruck

Bemerkungen:

Die DIM-Anweisung legt für einen Vektor (eindimensional) oder für eine Matrix (zweidimensional) die Gesamtlänge, die Dimension und die Indexbereiche fest.

Die DIM-Anweisung reserviert für das Feld (Text-Feld) den notwendigen Speicherplatz im Hauptspeicher.

Der Feldname (Text-Feldname) bezeichnet das gesamte Feld.

Die Werte der numerischen Ausdrücke spezifizieren die Anzahl der Zeilen eines eindimensionalen Feldes bzw. die Anzahl der Zeilen und Spalten eines zweidimensionalen Feldes.

Bei Textfeldern kann mit einem weiteren numerischen Ausdruck (nach dem Asterix-Zeichen '*') die Textfeldlänge (Zeichenanzahl) festgelegt werden.

Der Zugriff auf ein Feldelement während des Programmlaufs erfolgt durch die Angabe des Feldnamens (Text-Feldnamens) und ein oder zwei Indizes (numerische Ausdrücke) je nach DIM-Vereinbarung.

Im Unterschied zu einer normalen Variablen sieht die LET-Anweisung beispielsweise wie folgt aus:

```
LET AX (5) = 7
LET AR (1,1) = 3.24
```

Die Dimensionierung ist erstens durch den zur Verfügung stehenden Speicherplatz und zweitens durch die Indizes, die im Bereich zwischen 0 und 255 liegen müssen, begrenzt.

Bei Textfeldern darf die Textfeldlänge eines Elements 80 Zeichen nicht überschreiten. Wird die Textfeldlänge bei der Dimensionierung nicht festgelegt, so wird dem Textfeld automatisch eine Länge von 16 Zeichen zugeordnet.

Bei der Dimensionierung wird den numerischen Feldern der Wert Null und den Textfeldern der ASCII-Wert Null zugeordnet.
Ein Feld darf innerhalb eines Programms nur einmal definiert werden, ansonsten wird die Fehlermeldung ERROR 5 angezeigt.

Da Null (0) ein legaler Wert für einen Index ist, hat das Feld jeweils eine um 1 größere Anzahl von Zeilen bzw. Spalten als die Werte der DIM-Anweisung.

Numerische und Textfelder dürfen den gleichen Namen haben; die DIM-Anweisung:

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

ist zulässig.
Für die Dimensionierung von Vektoren und Matrizen können Feldnamen, bestehend aus einem oder zwei Zeichen verwendet werden. Dabei muß das erste Zeichen ein Buchstabe sein, das zweite Zeichen kann aus einem Buchstaben oder einer Ziffer bestehen.

Buchstabenkombinationen, die ein BASIC-Schlüsselwort oder eine Funktion ergeben, sind als Variablenname nicht erlaubt, z. B. LN, ON, PI etc.

Die Indizes liegen im Intervall 0 – 255

Hinweise:

1. Die Variable A nimmt eine Sonderstellung ein.
   Einzelheiten siehe Abschnitt 2.2.3.4.

2. Es empfiehlt sich, der Übersicht halber alle Felder möglichst vor der ersten Anweisung des Hauptprogramms zu dimensionieren.

3. Nach Fertigstellung des Programms sollten die DIM-Anweisungen anhand des noch verfügbaren Speicherplatzes überprüft werden. Dabei gilt:

   Anzahl der dimensionierten Elemente INT ((MEM -7)/8)
   Handelt es sich um eine Textfeldvariable, muß anstelle der 8 eine 16 bzw. die gewünschte Textfeldvariable eingegeben werden.

4. Mit dem PC-1430 ist es möglich, eine indizierte Variable als Index einer zweidimensionalen Feldvariablen zu verwenden.

```
B(A*B,C(0))=10
B(C(0),5)=10
B(4,A(30))=10
```

Beispiele:

(1) 10 DIM X(4)

Die Programmzeile 10 vereinbart ein numerisches eindimensionales Feld mit fünf Elementen, das wie folgt aussieht:

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

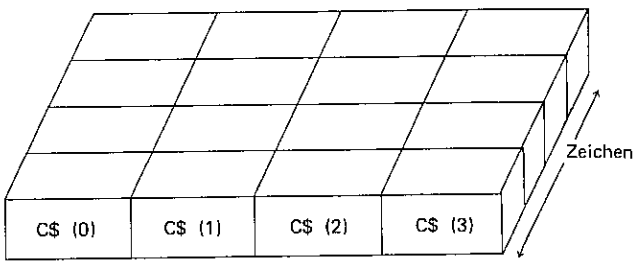
(2) 20 DIM B(2,3)

Die Programmzeile 20 vereinbart ein zweidimensionales Feld mit 12 Elementen.

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)

(3) 30 DIM C$(3) * 4

Die Programmzeile 30 vereinbart ein Textfeld mit vier Elementen mit einer Textfeldlänge von je vier Zeichen.



(4)
```
10 DIM V(3), P(2)
20 I = 4 : J = 5
30 DIM W$ (I * J)
```

In Programmzeile 30 wird ein Textvektor dimensioniert, dessen Index sich aus dem Produkt der Variablen I und J errechnet (W$(20)).

(5)
```
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 vereinbart:

S(Ø, Ø)	S(Ø, 1)
S(9, Ø)	S(9, 1)

In der Programmschleife von Zeile 20 bis Zeile 50 nimmt die Schleifenvariable I die Werte 0, 10, 20 bis 90 an. In den Zeilen 30 und 40 werden die Indizes des Feldes S bestimmt.
Nach Durchlaufen der Schleife enthält S den winkel und den Sinus des Winkels.

Ausgabe nach RUN:

```
 0.       0.    ENTER
10.  1.7E-01    ENTER
20.  3.4E-01    ENTER
30.       0.5   ENTER
40.  6.4E-01    ENTER
50.  7.6E-01    ENTER
60.  8.6E-01    ENTER
70.  9.3E-01    ENTER
80.  9.8E-01    ENTER
90.       1.
```

DMS

Funktion:

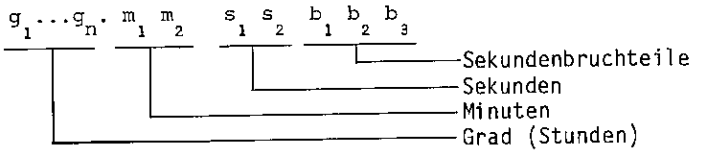
Umrechnung eines numerischen Ausdrucks vom Dezimalsystem ins Sexagesimalsystem.

Syntax:

DMS numerischer Ausdruck

Bemerkungen:

Sexagesimal geteilte Winkel oder Zeiten werden in folgendem Format verarbeitet:



Beispiel:

Umrechnung von 15.4125 (Grad) in Grad/Minuten/Sekunden

```
10 A=DMS 15.4125
20 PRINT A
30 END
```

Ausgabe nach RUN:

```
         15.2445
```

END

Funktion:

Beendet die Programmausführung.

Syntax:

END

Bemerkungen:

Die END-Anweisung steht normalerweise am Ende eines Programmes und beendet die Ausführung.

Unter Umständen kann die END-Anweisung auch weggelassen werden, da die Programmausführung automatisch nach Abarbeiten der letzten Programmzeile beendet wird. Sind mehrere Programme im Hauptspeicher gespeichert, wird bei fehlender END-Anweisung das nächste Programm sofort gestartet.

Beispiel:

```
10 PRINT "START PRO 1"
20 FOR I = 1 TO 20
30 WAIT 59 : PRINT I
40 NEXT I
100 PRINT "START PRO 2"
110 PRINT "PRO 1 KEIN END"
120 END
```

Ausgabe nach RUN:

```
START PRO 1
```

ENTER

```
1.
2.
.
.
.
20.

START PRO 2
```

ENTER

```
PRO 1 KEIN END
```

EXP

Funktion:

Exponentialfunktion e^X.

Syntax:

EXP numerischer Ausdruck

Bemerkungen:

Die Basiszahl e ist als EXP 1 = 2.718281828 gespeichert.

Beispiel:

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

Ausgabe nach RUN:

```
             100.
```

FACT

Funktion:

Berechnet die Fakultät eines numerischen Ausdrucks.

Syntax:

FACT numerischer Ausdruck

Bemerkungen:

Die Fakultät eines numerischen Ausdrucks berechnet sich nach der folgenden Formel:

```
Fakultät von X = 1*2*3*....*X
```

Beispiel:

```
10 A = 5
20 B = FACT A
30 PRINT B
40 END
```

Ausgabe nach RUN:

```
             120.

1 * 2 * 3 * 4 * 5 = 120
```

FOR...TO...STEP - NEXT

Funktion:

Wiederholung von Anweisungen in einer Schleife.

Syntax:

FOR num. Variable = num Ausdruck TO num. Audruck
FOR num. Variable = num. Ausdruck TO num. Audruck STEP num. Ausdruck

Bemerkungen:

Eine Programmschleife besteht aus einer FOR-Anweisung, den nachfolgenden Programmanweisungen und der NEXT-Anweisung.

Die FOR-Anweisung bewirkt, daß der von FOR und NEXT eingeschlossene Programmteil in Abhängigkeit der numerischen Ausdrücke (Anfangswert, Endwert, Schrittweite) wiederholt ausgeführt wird.

Wird die Schrittweite (STEP) nicht definiert, so ist sie automatisch auf den Wert 1 festgesetzt.

Ist der Endwert der Schleife erreicht, so wird die Programmausführung nach der NEXT-Anweisung fortgesetzt.

Programmschleifen können bis zu fünf Ebenen ineinander geschachtelt sein. Es ist nur darauf zu achten, daß immer zuerst die innere Schleife mit NEXT abgeschlossen werden muß.

Beispiel:

```
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
130 FOR C = A * 2 TO B

180 NEXT A
200 NEXT C
```

FALSCHE SCHACHTELUNG

Eine Sprunganweisung (GOTO, GOSUB) im Schleifenbereich kann während der Ausführung der Schleife in einen Programmteil außerhalb der Schleife verzweigen.
Es darf jedoch nicht von außerhalb in eine Programmschleife verzweigt werden. Eine Ausnahme ist der Rücksprung in eine Schleife aus einem Unterprogramm. Wurde ein solches während des Schleifendurchlaufs durch GOSUB aufgerufen, lenkt die Programmausführung nach Abarbeiten des Unterprogramms in die Schleife zurück.

Ein unzulässiger Sprung in eine FOR...NEXT-Schleife führt bei der folgenden NEXT-Anweisung zu der Fehlermeldung ERROR 5.

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

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

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

Vor dem ersten Durchlauf wird der Schleifenvariablen der Wert vom ersten numerischen Ausdruck (Anfangswert) der FOR-Anweisung zugewiesen.
Erreicht der Programmablauf die NEXT-Anweisung, so wird die Schrittweite (dritter numerischer Ausdruck) zum Wert der Schleifenvariablen addiert und der Programmteil mit diesem Wert erneut durchlaufen. Dies wird solange fortgesetzt, so lange der Wert innerhalb der im folgenden angegebenen Bereiche liegt:

(1) Bei positiver Schrittweite:

Anfangswert $<$ Schleifenvariable $<$ Endwert + Schrittweite

(2) oder bei negativer Schrittweite:

Schrittweite + Endwert $<$ Schleifenvariable $<$ Anfangswert

Der Wert der Schleifenvariablen einer Sprunganweisung darf im Bereich zwischen -9.999999999E99 und 9.999999999E99 liegen. Für die Schrittweite gilt der gleiche Bereich; wird der Wert 0 gewählt, ist die Schleife unendlich.

Nach Beendigung der Schleifenanweisung bleibt die Schleifenvariable definiert und behält den ihr zuletzt zugewiesenen Wert.

Die Werte der zugewiesenen Anfangs-, End- und Schrittweitenparameter bleiben fest, solange die Schleife durchlaufen wird. Das gilt auch, wenn der Wert einer zu ihrer Berechnung verwendeten Variablen innerhalb oder außerhalb der Schleife geändert wird.

Beispiele:

zu (1) Einlesen der Zeilen einer Matrix B=B(1,J) über eine geschachtelte Schleifenanweisung.

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

(2)
```
10 WAIT 59 : PRINT"LISTE DER QUADRATZAHLEN"
20 WAIT 59 : PRINT"VON 10 BIS 20"
30 FOR Z = 10 TO 20
40 Q = SQU Z
50 WAIT 59 : PRINT Q
60 NEXT Z
70 END
```

GOSUB

Funktion:

Verzweigung zu einem Unterprogramm.

Syntax:

GOSUB numerischer Ausdruck
GOSUB Textausdruck

Bemerkungen:

Werden in einem Programm an verschiedenen Stellen gleiche Befehlsfolgen verwendet, so werden diese als Unterprogramm (Subroutine) geschrieben.

Die GOSUB-Anweisung verzweigt den Programmablauf zu der Programmzeile, in der das entsprechende Unterprogramm beginnt. Analog zur GOTO-Anweisung wird das Sprungziel entweder als Zeilennummer oder als Markenname angegeben.

Ein Unterprogramm wird mit einer RETURN-Anweisung abgeschlossen; der Programmablauf wird dann mit der der GOSUB-Anweisung folgenden Anweisung fortgesetzt.

Die GOSUB-Anweisung kann an beliebiger Stelle im Programm stehen.

Bis zu zehn GOSUB-Anweisungen können ineinander geschachtelt werden. Dabei darf jedoch z.B. ein Unterprogramm A, das ein Unterprogramm B aufruft, nicht wiederum von B aufgerufen werden. Anderseits kann B sowohl vom Hauptteil des Programms als auch von A aus aufgerufen werden.

Beispiel:

(1)
```
120 GOSUB 300
130 PRINT "A="; A
 .
 .
180 GOSUB 300
190 IF A = 10 THEN 250
 .
 .
300 LET X = C + B        )
 .                       ) Unterprogramm
 .                       )
380 RETURN               )
```

(2)

Hauptprogramm	Unterprogramme	
	1. Ebene	2. Ebene
50 GOSUB "SIMP"	400 "SIMP":	500 LET X = A * B
60	450 GOSUB 500	.
.	.	.
.	.	.
.	.	590 RETURN
160 GOSUB "SIMP"	490 RETURN	
170		
.		
.		
.		
270 GOSUB "SON"	700 "SON":	
280	.	
.	.	
.	.	
.	790 RETURN	
390 END		

In diesem Beispiel werden mehrere Unterprogramme ineinander verschachtelt. In Zeile 50 und 160 wird das Unterprogramm "SIMP" aufgerufen. Dieses wiederum ruft in Zeile 450 das Unterprogramm der zweiten Ebene in Zeile 500 auf. Der Rücksprung erfolgt jeweils zu der Anweisung, die auf den entsprechenden GOSUB-Aufruf folgt.

GOTO

Funktion:

Verzweigt die Programmausführung.

Syntax:

```
GOTO num. Ausdruck < Zeilennummer >
GOTO Textausdruck < Markenname >
```

Bemerkungen:

Die GOTO-Anweisung verzweigt den Programmablauf zu einer bestimmten Programmzeile, die entweder als numerischer Ausdruck (Zeilennummer) oder als Textausdruck (Markenname) angegeben sein kann.

Ist die angegebene Sprungadresse im Programm nicht vorhanden, so erfolgt die Fehlermeldung ERROR 4.

Beispiel:

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

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

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
```

IF...THEN

Funktion:

Programmverzweigung in Abhängigkeit von einem logischen Ausdruck.

Syntax:

IF Vergleichsausdruck **THEN** Anweisung
IF Vergleichsausdruck **THEN** num. Ausdruck
IF Vergleichsausdruck **THEN** Textausdruck

IF num. Ausdruck **THEN** Anweisung
IF num. Ausdruck **THEN** num. Ausdruck
IF num. Ausdruck **THEN** Textausdruck

Bemerkungen:

Der Vergleichsausdruck bzw. der numerische Ausdruck stellt die Bedingung dar, die auf 'wahr' oder 'falsch' geprüft wird.

Im Fall des numerischen Ausdrucks wird die Bedingung als 'wahr' angesehen, wenn der Wert größer (>) 0, und als 'falsch' wenn er kleiner gleich (≤) 0 ist.

Der Vergleichsausdruck wird nach den mathematischen Regeln der Operatoren < ,<=,=,>=,> ,<>bestimmt und liefert die Werte "1" (wahr) oder "0" (falsch).

Bei der Anwendung der Operatoren auf Textausdrücke wird auf lexikalische Reihenfolge geprüft. Dabei werden nur die ersten 16 Zeichen des Textausdruckes berücksichtigt.

Ist die Bedingung einer IF-Abfrage 'wahr', so werden die auf THEN folgenden Anweisungen ausgeführt.

Wird nach einer THEN-Anweisung nur eine Zeilennummer oder ein Textausdruck angegeben, so stellt dies die verkürzte Schreibweise für GOTO dar.

Beispiel: IF A < 3 THEN 100 = IF A < 3 THEN GOTO 100

Anweisungen, die in der gleichen Zeile stehen wie die IF...THEN-Anweisung, werden nur ausgeführt, wenn das Ergebnis der Abfrage 'wahr' ist.

Mit dem **PC-1430** können Rechnungen mit bis zu 12stelliger Mantisse durchgeführt werden. Die Mantisse wird intern bis zur 12. Stelle berechnet, für die Anzeige wird jedoch ab der 10. Stelle gerundet.

Beispiel:

	internes Ergebnis	Anzeige
5/9 ---→	5.55555555555E-01	5.555555556E-01
5/9∗9---→	4.99999999999E-00	5.

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

Beispiel:

		Ergebnis
a.)	1/6 ∗ 3	0,5
b.)	1/6 [ENTER] ∗3	5.000000001E-01

Obgleich die Ausgangspunkte und die Berechnungen gleich sind, werden zwei unterschiedliche Ergebnisse errechnet, weil bei der Berechnung von

a.) einmal, und bei der Berechnung von b.) zweimal gerundet wird.

In einer IF-Anweisung kann diese Differenz dazu führen, daß das Programm nicht mehr fehlerfrei arbeitet.

```
300:  "A":INPUT A
310:  X=3
320:  Y=1/A
330:  Z=Y∗X
340:  IF Z=1/A∗X THEN 370
350:  PRINT "KEINE VERZWEI
      G.".
360:  GOTO 300
370:  PRINT "VERZWEIGUNG"
380:  GOTO 300
```

Während bei der Eingabe von z. B. "3" die Verzweigung im nebenstehenden Programm ordnungsgemäß ausgeführt wird, führt die Eingabe von z. B. "6" zu einer falschen Abzweigung

Dieser Fehler kann vermieden werden, wenn in der Zeile 340 der Vergleichsausdruck IF Z = 1/A ∗ X geändert wird in

```
IF Z = Y ∗ X
```

Beispiele:

(1) 10 IF A = 10 THEN PRINT "A = 10"

Wenn der Wert der Variablen A = 10 ist, wird am Display "A = 10" angezeigt.

(2) 10 IF A$ = "JA" THEN GOTO 200

(3) 10 IF A$ = "JA" GOTO 200

(4) 10 IF A$ = "JA" THEN 200

Die Beispiele 2,3 und 4 haben dieselbe Wirkung. Die Programmzeile 10 enthält die Abfrage für die Textvariable A$. Ist A$ ="JA", so verzweigt das Programm zu Zeile 200.

(5)
```
10 X = 100
 .
 .
 .
50 IF X THEN 200
 .
 .
 .
200 PRINT "SHARP"
```

Der Programmverlauf wird in Zeile 50 zu Zeile 200 verzweigt, da die Abfrage IF X > 0 ergibt und daher "wahr" ist.

(6)

```
10 A = RND 500
20 INPUT "ZUF.ZAHL ?";B
30 IF A < B GOTO 110
40 IF A > B GOTO 100
50 IF A = B PRINT "RICHTIG (ENTER)": GOTO 200
100 WAIT 150 : PRINT "ZAHL I. GROESSER" : GOTO 20
110 WAIT 150 : PRINT "ZAHL I. KLEINER" : GOTO 20
200 INPUT "NOCHMAL-J/N"; C$
210 IF C$ = "J" THEN 10
220 IF C$ <> "N" THEN 200
230 END
```

INKEY$

Funktion:

Tastaturabfrage

Syntax:

Textvariable = **INKEY$**

Bemerkungen:

INKEY$ fragt die Tastatur ab und ergibt, je nach gedrückter Taste einen Textausdruck von einem Zeichen Länge oder einen leeren Textausdruck (ASCII-Null), wenn keine Taste gedrückt wurde.

Beispiel:

```
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 (PROGRAMM)"
910 END
```

Wird bei der Abarbeitung der Programmzeile 20 die Taste [S] oder [E] gedrückt, verzweigt sich das Programm gemäß den Anweisungen in Zeile 30 und 40. Wird keine oder eine andere Taste gedrückt, springt die Programmausführung in Zeile 50 zurück zu Zeile 20.
Mit der Taste [S] wird das Programm gestartet, mit der Taste [E] beendet. Nach jedem Durchlauf muß es mit [S] wieder gestartet werden.

([Exp] ist gleich wie [E] .)
Das Programm kann mit der Taste [BRK] unterbrochen werden.
Mit dem INKEY$ Kommando können keine Kommandos oder Symbole eingelesen werden, die die Taste [SHIFT] erfordern.
Wenn das Kommando INKEY$ am Beginn eines Programms eingegeben wird, kann beim Starten des Programms automatisch die Starttaste gelesen werden.

INPUT

Funktion:

Weist Variablen Werte zu, die über die Tastatur eingegeben werden.

Syntax:

INPUT numerische Variable
INPUT Textvariable
INPUT"Textausdruck", numerische Variable
INPUT"Textausdruck"; numerische Variable
INPUT"Textausdruck", Textvariable
INPUT"Textausdruck"; Textvariable

Bemerkungen:

Die Ausführung des Programms wird unterbrochen, und auf der Anzeige erscheint der eingegebenen Textausdruck oder ein ? (Fragezeichen). Das Fragezeichen dient als Hinweis darauf, daß die Werte für die Variablen über die Tastatur eingegeben werden können. Jede einzelne Eingabe wird mit [ENTER] abgeschlossen. Dies wird so lange wiederholt, bis für alle Variablen die Werte eingegeben sind. Im Anschluß daran wird die Programmausführung fortgesetzt.

Beispiel:

(1)
```
10 INPUT A
20 IF A = 1234 THEN 100
30 GOTO 10
100 PRINT A
```

Ausgabe nach RUN:

?

Eingabe: 1 2 3 4

1234 [ENTER]

Anzeige nach Abarbeitung der Programmzeile 100:

1234.

Ist die eingegebene Zahlenreihe nicht 1234, so springt die Programmausführung von Zeile 30 zu Zeile 10 zurück.

(2)
```
10 INPUT A$
20 IF A$ = "START" THEN 100
30 GOTO 10
100 PRINT "PROG.ENDE"
```

Ausgabe nach RUN:

?

Eingabe: START

START ENTER

Anzeige nach Abarbeitung der Programmzeile 100:

PROG.ENDE

(3)
```
10 INPUT A,B
20 IF A > B THEN 100
30 GOTO 10
100 PRINT "PROG.ENDE"
```

Ausgabe nach RUN:

?

Eingabe: 1 und 2 (Wertzuweisung für Variable A)

12 ENTER

?

Eingabe: 9 (Wertzuweisung für Variable B)

9 ENTER

Anzeige nach Abarbeitung der Programmzeile 100:

PROG.ENDE

(4)
```
10 INPUT"DATA-1",A
20 IF A > 12 THEN 100
30 GOTO 10
100 PRINT "PROG.ENDE"
```

Ausgabe nach RUN:

DATA-1

Eingabe: 123 (Wertzuweisung für Variable A)

123 ENTER

Anzeige nach Abarbeitung der Programmzeile 100:

PROG.ENDE

(5)
```
10 INPUT "DATA-1=";A,"DATA-2=";B
20 IF A > B THEN 100
30 GOTO 10
100 PRINT"PROG.ENDE"
```

Ausgabe nach RUN:

DATA-1=_

Eingabe: 75

DATA-1=75_ ENTER

DATA-2=_

Eingabe: 71

DATA-2=71_ ENTER

Anzeige nach Abarbeitung der Programmzeile 100:

PROG.ENDE

INPUT# (nur mit Option CE-126P)

Funktion:

Daten werden vom Band gelesen und Variablen zugewiesen.

Syntax:

INPUT# Variable
INPUT# Textkonstante; Variable

Bemerkungen:

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

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

Wenn die Anzahl der einzulesenden Variablen nicht mit der Anzahl der gespeicherten Variablen übereinstimmt, so ergeben sich folgende Zustände:

1. Anzahl der einzulesenden Variablen ist größer als die Anzahl der auf Band gespeicherten Variablen:

   Die Übertragung wird nicht beendet und muß mit BREAK abgebrochen werden oder es wird die Fehlermeldung ERROR 8 angezeigt.

2. Die Anzahl der einzulesenden Variablen ist kleiner als die Anzahl der auf Band gespeicherten Variablen:

   Die überzählige Variable wird ignoriert.

Beispiel:

```
INPUT# A, K*, A1$(*)
INPUT#"DATEI"; A*, X1(*)
```

Hinweis:

Bei korrektem Einlesen der Daten wird in der rechten Stelle der Anzeige ein Sternsymbol angezeigt.

INT

Funktion:

Die INTeger-Funktion ermittelt den ganzzahligen Anteil eines numerischen Ausdrucks.

Syntax:

INT numerischer Ausdruck

Bemerkungen:

Ist der Wert des numerischen Ausdrucks positiv, so ergibt die INT-Funktion einen Wert, der gleich oder kleiner dem Wert des numerischen Ausdrucks ist. Ist der Wert jedoch negativ, so ergibt sich ein Wert, dessen Betrag gleich oder größer dem Wert des numerischen Ausdrucks ist.

Beispiel:

```
10 A = INT (PI)
20 B = INT (3.99)
30 C = INT (-3.14)
40 PRINT A;" ";B;" ";C
50 END
```

Ausgabe nach RUN:

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

LEN

Funktion:

Berechnet die Anzahl der Zeichen eines Textausdrucks.

Syntax:

LEN Textausdruck

Bemerkungen:

Leerstellen werden bei der Berechnung der Zeichen mitgerechnet.

Beispiel:

```
10 A = LEN "SHARP"
20 B$ = "HAMBURG"
30 C = LEN B$
40 PRINT A;" ";C
50 END
```

Ausgabe nach RUN:

5. 7.

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

LET

Funktion:

Weist einer Variablen (einfach oder indiziert) einen Wert zu.

Syntax:

```
LET numerische Variable = numerische Variable
LET numerische Variable = numerischer Ausdruck
LET Textvariable = Textvariable
LET Textvariable = Textausdruck
```

Bemerkungen:

Die LET-Anweisung weist einer Variablen (numerisch oder Text) einen Wert zu. Dabei muß der Variablentyp dem zugewiesenen Wert entsprechen, d.h., einer numerischen Variablen darf nur ein numerischer Wert zugewiesen werden, einer Textvariablen nur ein Textausdruck.
Zur Umwandlung einer Textvariablen in eine numerische Variable dient die Anweisung VAL.

Beispiel:

```
(1) LET A = 5

(2) LET X = Y

(3) LET B$ = "SHARP"

(4) LET C$ = D$
```

LIST

Funktion:

Auflisten der Programmzeilen.

Syntax:

```
LIST
LIST num. Ausdruck < Zeilennummer >
LIST num. Ausdruck, num. Ausdruck < Bereich >
```

Bemerkungen:

Die Ausführung des LIST-Kommandos kann nur im PRO-Mode erfolgen. Im RUN-Mode wird die Fehlermeldung ERROR 9 angezeigt.

LIST ohne zusätzliche Angabe listet die Programmzeile mit der niedrigsten Zeilennummer auf. Durch Angabe einer bestimmten Zeilennummer oder eines Markennamens lassen sich bestimmte Programmzeilen zur Anzeige bringen. Ist die angegebene Programmzeile nicht vorhanden, so wird die nächsthöhere gelistet.

Durch Betätigung der Tasten [↑] oder [↓] wird die folgende bzw. die vorangehende Programmzeile angezeigt.

Beispiel:

(1) LIST 100

Die Programmzeile 100 wird zur Anzeige gebracht.

(2) LIST

(3) LIST "S"

LLIST (nur mit Option CE-126P)

Funktion:

Auflisten der Programmzeilen auf dem Drucker.

Syntax:

LLIST
LLIST numerischer Ausdruck < Zeilennummer >
LLIST numerischer Ausdruck , numerischer Ausdruck < Bereich >
LLIST numerischer Ausdruck , 1. zu druckende Zeile
LLIST numerischer Ausdruck

Bemerkungen:

Die LLIST-Anweisung hat die gleiche Funktion wie die LIST-Anweisung. Das Programm wird jedoch auf dem Drucker und nicht auf der Anzeige gelistet.
Ohne Angaben von Programmzeilen wird das gesamte Programm ausgedruckt.
Existieren die angegebenen Zeilennummern nicht, so wird das Listing bei der nächsthöheren Zeilennummer begonnen bzw. beendet.

Beispiel:

LLIST

Das im Hauptspeicher gespeicherte Programm wird am Drucker gelistet.

LLIST 10,100

Die Programmzeilen, beginned mit der Zeilennummer 10 bis zur Zeilennummer 100, werden am Drucker gelistet.

LLIST 100,

Das im Hauptspeicher gespeicherte Programm wird ab Zeile 100 am Drucker gelistet.

LLIST, 100

Das im Hauptspeicher gespeicherte Programm wird bis Zeile 100 am Drucker gelistet.

LPRINT (nur mit Option **CE-126P**)

Funktion:

Numerische Werte und Textausdrücke am Drucker ausgeben.

Syntax:

LPRINT numerischer Ausdruck
LPRINT Textausdruck
LPRINT numerischer Ausdruck,(;) numerischer Ausdruck
LPRINT Textausdruck,(;) Textausdruck
LPRINT numerischer Ausdruck,(;) Textausdruck
LPRINT Textausdruck,(;) numerischer Ausdruck
LPRINT USING Format; numerischer Ausdruck; ...

Bemerkungen:

Der USING-Teil der LPRINT-Anweisung spezifiziert das Ausgabeformat, wie unter USING beschrieben.

Mit dem Komma wird die Aufteilung der 24spaltigen 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 vereinbart, so werden numerische Ausdrücke rechtsbündig und Textausdrücke linksbündig gedruckt.

Nach der Ausführung einer LPRINT-Anweisung wird der Programmablauf, im Gegensatz zur PRINT-Anweisung, automatisch fortgesetzt.

Hinweise:

1. Werden mehr als 24 Zeichen ausgegeben, wird zwei- oder mehrzeilig gedruckt. Für den Fall, daß die Ausdrücke der LPRINT-Liste mit einem Komma verknüpft sind, werden Textausdrücke auf 12 Zeichen begrenzt.

2. Wird in einer LPRINT-Anweisung die USING-Anweisung verwendet, so ist das festgelegte Format für alle weiteren LPRINT-Anweisungen bis zur nächsten USING-Anweisung gültig.

3. Das USING-Format wird durch die RUN-Anweisung oder durch Drücken der Tasten [SHIFT] und [CL] gelöscht.

Die einzelnen Formate der LPRINT-Anweisung haben folgende Wirkung:

LPRINT numerischer Ausdruck
LPRINT Textausdruck

Der in der LPRINT-Anweisung spezifizierte Inhalt wird rechts- bzw. linksbündig ausgedruckt.

LPRINT numerischer Ausdruck, numerischer Ausdruck
LPRINT Textausdruck, Textausdruck

Die Zeile wird in zwei Felder geteilt. Der erste numerische Ausdruck (bzw. Textausdruck) wird in der linken, der zweite in der rechten Hälfte jeweils rechts- bzw. linksbündig gedruckt.

LPRINT numerischer Ausdruck ; numerischer Ausdruck
LPRINT Textausdruck ; Textausdruck

Der Inhalt der LPRINT-Liste wird, beginnend mit der jeweiligen Cursor-Position, ausgedruckt.

Beispiele:

(1)
```
10 A = 3.1415
20 B$ = "SHARP"
30 LPRINT A
40 LPRINT B$
```

Ausgabe nach RUN:

```
                3.1415
SHARP
```

(2)
```
10 A = 3.1415
20 B$ = "SHARP"
30 C = 7.89
40 D$ = "PC-1430"
50 LPRINT A;B$
60 LPRINT B$,A
70 LPRINT B$,D$
80 LPRINT A,C
```

Ausgabe nach RUN:

```
3.1415SHARP
SHARP             3.1415
SHARP       PC-1430
      3.1415        7.89
```

(3)
```
10 A = 3.1415
20 B$ = "SHARP"
30 C = 7.89
40 D$ = "PC-1430"
50 LPRINT A;B$;D$;C
60 LPRINT A;" ";B$;" ";D$;" ";C
```

Ausgabe nach RUN:

```
3.1415SHARPPC-14017.89
3.1415 SHARP  PC-1430 7.8
9
```

(4)
```
10 A = 3.1415
20 B$ = "SHARP"
30 C = 7.89
40 D$ = "PC-1430"
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
```

Ausgabe nach RUN:

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

LN

Funktion:

Berechnet den natürlichen Logarithmus eines numerischen Ausdrucks.

Syntax:

LN numerischer Ausdruck

Bemerkungen:

Der Wert des numerischen Ausdrucks darf nicht negativ sein.

Beispiel:

```
10 A = LN 100
20 PRINT A
30 END
```

Ausgabe nach RUN:

```
4.605170186
```

LOG

Funktion:

Berechnet den Zehnerlogarithmus eines numerischen Ausdrucks.

Syntax:

LOG numerischer Ausdruck

Bemerkungen:

Der Wert des numerischen Ausdrucks darf nicht negativ sein.

Beispiel:

```
10 A = LOG 100
20 PRINT A
30 END
```

Ausgabe nach RUN:

```
 2.
```

MEM

Funktion:

Ermittelt den freien Speicherbereich des Hauptspeichers.

Syntax:

MEM

Bemerkungen:

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

Beispiel:

MEM

Anzeige am Display:

1254

Ist das Programm durch ein PASS-Wort geschützt, so kann es durch die NEWØ-Anweisung gelöscht werden, die Standardvariablen werden auf Null gesetzt (s. PASS).

MID$

Funktion:

Ergibt den mittleren Teil eines Textausdrucks mit der Zeichenzahl n ab dem p-ten Zeichen.

Syntax:

MID$ (Textausdruck, numerischer Ausdruck 1, numerischer Ausdruck 2)

Bemerkungen:

Der Wert des ersten numerischen Ausdrucks legt die Position innerhalb des Textausdrucks fest, von der ab die Zeichen entnommen werden. Der Wert des zweiten numerischen Ausdrucks bestimmt die Zeichenanzahl, die entnommen werden soll.

Beispiel:

```
10 DIM A$(0)*20
20 A$(0)="SHARP HAMBURG"
30 B$=MID$(A$(0),7,4)
40 PRINT"2000 ";B$
50 END
```

Ausgabe nach RUN:

```
2000 HAMB
```

Dem Textausdruck "SHARP HAMBURG" werden, beginnend mit dem siebten Zeichen, vier Zeichen entnommen und der Textvariablen B$ zugewiesen.

NEW

Funktion:

Löscht den Hauptspeicher.

Syntax:

NEW

Bemerkungen:

Der Hauptspeicher wird gelöscht. Alle Variablen werden auf Null gesetzt.

Die NEW-Anweisung kann nur im PRO-Mode eingegeben werden, ansonsten wird die Fehlermeldung ERROR 9 angezeigt.

Beispiel:

NEW

Hinweis: Dieses Kommando ist nicht wirksam, wenn ein Passwort gesetzt wurde.

NEWØ

Funktion:

Initialisiert den Computer (löscht den Hauptspeicher).

Syntax:

NEWØ

Bemerkungen:

Alle Programme, Feldvariable, einfache Variable und andere Speicherinhalte werden gelöscht.

* Dieses Kommando ist selbst dann wirksam, wenn ein Passwort gesetzt wurde.

PASS

Funktion:

Schützt ein Programm vor dem Zugriff durch unbefugte Personen.

Syntax:

PASS Textkonstante

Bemerkungen:

Zum Schutz der Programme vor unberechtigtem Auflisten oder Verändern können die Programme mit einem PASS-Wort versehen werden.
Bei der Verwendung eines PASS-Wortes werden die LIST- und die LLIST-Anweisung nicht ausgeführt. Die Editierfunktionen werden abgeschaltet.
Es können keine Programmzeilen ergänzt oder gelöscht werden.

Das PASS-Wort kann maximal aus sieben Zeichen bestehen. Wird ein längeres PASS-Wort eingegeben, so werden die überzähligen Zeichen ignoriert.

Das PASS-Wort schützt den gesamten Programmspeicher, d.h. sind mehrere Programme gespeichert, läßt sich nicht jedem Programm ein PASS-Wort zugeordnen.

Wird nach der Eingabe eines Programms ein PASS-Wort eingegeben, so ist der Programmspeicher für weitere Programmeingaben gesperrt.

Das PASS-Wort wird durch nochmalige Eingabe wieder aufgehoben.

Ein PASS-Wort kann nicht überschrieben werden, d.h., ist bereits ein PASS-Wort definiert, so führt die Eingabe eines weiteren PASS-Wortes zu einer Fehlermeldung (ERROR 9).

Wenn kein Programm im Computer gespeichert ist und ein PASS-Wort eingegeben wird, wird dieses nicht gesetzt, sondern eine Fehlermeldung (ERROR 1) wird angezeigt.

Folgende Punkte sind bei der Verwendung eines PASS-Wortes zu beachten:

1. Das PASS-Wort kann nur direkt im RUN- oder im PRO-Mode eingegeben werden, darf also nicht in einer Programmzeile stehen.
2. Ein PASS-Wort wird gelöscht, indem es nochmals eingegeben wird.
3. Ein durch ein PASS-Wort geschütztes Programm kann mit der NEWØ-Anweisung gelöscht werden.

4. Ist ein Programm durch ein PASS-Wort geschützt, so kann es nicht auf Band gespeichert werden.
5. Ist ein Programm nicht geschützt, so kann es beim Abspeichern auf Band geschützt werden (siehe CSAVE).
6. Beim Laden von geschützten Programmen ergeben sich folgende Abhängigkeiten:

Operation	Hauptspeicherinhalt
0 CLOAD A	A mit PASS-Wort von A
0 CLOAD B	B ohne PASS-Wort
1 CLOAD A	A mit PASS-Wort von A
1 CLOAD B	B ohne PASS-Wort
2 CLOAD A	A mit PASS-Wort von A
2 CLOAD B	B ohne PASS-Wort

Es bedeuten - 0 = kein Programm im Computer
1 = geschütztes Programm im Computer
2 = ungeschütztes Programm im Computer
A = geschütztes Programm auf Kassette
B = ungeschütztes Programm auf Kassette

Beispiel:

PASS"SHARP"

Das Programm im Hauptspeicher wird durch das PASS-Wort "SHARP" geschützt.

Eingabe:

LIST

Das Programm wird durch das PASS-Wort geschützt und kann nicht gelistet werden.

Nochmalige Eingabe von:

PASS"SHARP"

hebt den Programmschutz wieder auf.

PI

Funktion:

Die Konstante Pi (π) ist als 3.141592654 gespeichert.

Syntax:

PI oder [π]

Bemerkungen:

Die Konstante PI kann über die Funktionstasten aufgerufen werden. Dabei wird sie am Display nicht als PI, sondern als π dargestellt.

Beispiel:

```
10 INPUT"RADIUS:";R
20 U=PI*R
30 PRINT"UMFANG=";U
40 END
```

Ausgabe nach RUN:

```
RADIUS:_
```

Eingabe
2.5 [ENTER]

```
UMFANG=7.8539816
```

POL

Funktion:

Umwandlung von rechtwinkligen Koordinaten in Polarkoordinaten.

Syntax:

POL (numerischer Ausdruck, numerischer Ausdruck)

Bemerkungen:

Der erste numerische Ausdruck gibt die Entfernung von der y-Achse an, der zweite die Entfernung von der x-Achse. Die berechneten Werte für die Entfernung und den Winkel in den Polarkoordinaten werden den Standardvariablen Y (Entfernung) und Z (Winkel) zugeordnet. Der Wert des Winkels ist dabei von der Winkeleinheit (DEGREE, GRAD, RADIAN) abhängig.
Im BASIC-Programm muß die POL-Funktion einer Variablen zugewiesen werden. Dabei darf die Variable Z nicht verwendet werden. Um Speicherplatz zu sparen, kann die Variable Y als Zuordnungsvariable gewählt werden.

Beispiel:

```
110:  DEGREE
120:  INPUT "STRECKE Y=";
      A
130:  INPUT "STRECKE X=";
      B
140:  Y=POL (A,B)
150:  PRINT "RADIUS R=";Y
160:  PRINT "WINKEL=";Z
170:  END
```

PRINT

Funktion:

Ausgabe von numerischen Werten und Zeichenfolgen am Display.

Syntax:

PRINT numerischer Ausdruck
PRINT Textausdruck
PRINT numerischer Ausdruck ,(;) numerischer Ausdruck
PRINT Textausdruck ,(;) Textausdruck
PRINT numerischer Ausdruck ,(;) Textausdruck
PRINT Textausdruck ,(;) numerischer Ausdruck
PRINT USING Format ; numerischer Ausdruck
PRINT USING Format ; Textausdruck

Bemerkungen:

Der USING-Teil der PRINT-Anweisung spezifiziert das Ausgabeformat, wie unter USING beschrieben.

Mit dem Komma wird die Aufteilung der 16spaltigen Displayzeile 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 vereinbart, so werden numerische Ausdrücke rechtsbündig und Textausdrücke linksbündig angezeigt.

Nach der Ausführung einer PRINT-Anweisung wird der Programmablauf angehalten, bis man ihn durch Drücken der ENTER-Taste wieder startet. Mit der WAIT-Anweisung läßt sich ein Zeitintervall festlegen, nach dessen Ablauf die Programmausführung auch nach einer PRINT-Anweisung automatisch fortgesetzt wird (siehe WAIT).

Die einzelnen Formate der PRINT-Anweisung haben folgende Wirkung:

PRINT numerischer Ausdruck
PRINT Textausdruck

Der in der PRINT-Anweisung spezifizierte Inhalt wird auf der Anzeige ausgegeben; ein Textausdruck wird linksbündig mit maximal 16 Zeichen, ein numerischer Ausdruck rechtsbündig mit maximal 10 Stellen und 2 Exponentialstellen angezeigt.

PRINT numerischer Ausdruck, numerischer Ausdruck (oder Textausdruck)
PRINT Textausdruck, Textausdruck (oder numerischer Ausdruck)

Die Anzeige wird in zwei Felder geteilt. Der erste numerische Ausdruck (bzw. Textausdruck) wird in der linken, der zweite in der rechten Hälfte angezeigt.
Dabei werden Textausdrücke jeweils linksbündig mit maximal 8 Zeichen und numerische Ausdrücke rechtsbündig mit maximal 6 Stellen angezeigt. Bei größeren Zahlen wird automatisch auf die wissenschaftliche Schreibweise umgeschaltet.

PRINT numerischer Ausdruck ; numerischer Ausdruck (oder Textausdruck)
PRINT Textausdruck ; Textausdruck (oder numerischer Ausdruck)

Der Inhalt der PRINT-Liste wird, beginnend mit der jeweiligen Cursor-Position, mit maximal 16 Stellen angezeigt.

Beispiele:

(1)
```
10 A = 3.1415
20 B$ = "SHARP"
30 PRINT A
40 PRINT B$
```

Ausgabe nach RUN:

```
        3.1415
                    ENTER
SHARP
```

(2)
```
10 A = 1430
20 B$ = "SHARP"
30 C = 100000 * A
40 PRINT A,B$
50 PRINT B$,A
60 PRINT A,A
70 PRINT C,A
```

Ausgabe nach RUN:

```
 1430. SHARP
                    ENTER
SHARP       1430.
                    ENTER
 1430.      1430.
                    ENTER
1.430E 08 1430.
```

(3)
```
10 A = 1430
20 B$ = "SHARP"
30 C$ = "PC"
40 D$ = " "
50 E$ = "VON"
60 PRINT E$;D$;B$;D$;C$;A
70 PRINT C$;A;E$;D$;B$
80 PRINT B$;D$;C$;D$;A
90 END
```

Ausgabe nach RUN:

```
VON SHARP PC 1430.
                    ENTER
PC1430. VON SHARP
                    ENTER
SHARP PC 1430.
```

(4)
```
5 WAIT 50
10 A = 1.234
20 B$ = "SHARP"
30 C = 5.67
40 D$ = "PC-1430"
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$;D$
120 PRINT USING "##.####";A,C
130 PRINT USING; A,C
```

Ausgabe nach RUN:

SH 1.2

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

SH 1.2

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

SHARP1.234

Das Format der Anzeige wurde nicht festgelegt. Die Ausgabe der Werte erfolgt daher im Standardformat.

1.2 5.6

USING E$ = &&####.#; die Textformatierung wird bei der Ausgabe von zwei numerischen Werten unterdrückt.

SHPC

USING F$ = ###.#&&; die Formatierung für den numerischen Ausdruck wird bei der Ausgabe von zwei Textausdrücken unterdrückt.

1.2340 5.6700

USING ##.####; es werden vier Nachkommastellen angezeigt; fehlende Stellen werden mit Null (0) aufgefüllt.

1.234 5.67

USING ohne Formatangaben; die Anzeige erfolgt im Standardformat.

Hinweise:

1. Die Anzeige wird nach Drücken der [CL]-Taste bzw. nach Fortsetzung des Programms gelöscht.

2. Mit der WAIT-Anweisung kann ein Zeitintervall definiert werden, nach dessen Ablauf die Programmausführung automatisch fortgesetzt wird.

3. Werden mehr Zeichen ausgegeben, als die Anzeige Schreibpositionen hat, so werden nur die ersten 16 Zeichen angezeigt.

4. Wird die PRINT USING-Anweisung verwendet, so bleibt das festgelegte Format so lange erhalten, bis eine weitere USING-Anweisung folgt.

5. Das USING-Format kann durch die RUN-Anweisung oder durch Drücken der Tasten [SHIFT] und [CL] gelöscht werden.

6. Ist die Option CE-126P an den Rechner angeschlossen, so kann die PRINT-Anweisung auch zur Ausgabe am Drucker verwendet werden. Mit der Anweisung PRINT = LPRINT erfolgt die Umschaltung der Ausgabe vom Display auf den Drucker. Diese Anweisung kann entweder direkt oder in einer Programmanweisung eingegeben werden.
   Die Anweisung wird gelöscht:
   a) Aus-/Einschalten des Computers
   b) RUN
   c) [SHIFT] [CL]
   d) (Zeilennummer) PRINT = PRINT

Beispiel:

```
10 PRINT = LPRINT
20 A$ = "PC-1430"
30 B$ = "SHARP"
40 PRINT A$,B$
50 PRINT = PRINT
60 PRINT A$,B$
70 END
```

Ausgabe nach RUN:

auf dem Drucker:

PC-1430 SHARP

auf dem Display:

PC-1430 SHARP

PRINT#

Funktion:

Werte von Variablen auf Band speichern.

Syntax:

```
PRINT# Variable
PRINT# Variable*
PRINT# Feldvariable (*)
PRINT# Variablenliste
PRINT# Textkonstante; Variablenliste
```

Bemerkungen:

Beim **PC-1430** unterschiedet man zwischen Programm- und Datenspeicher. Mit der **PRINT #**-Anweisung werden die Werte einer Variablen oder einer Gruppe von Variablen auf Band gespeichert.

Die Textkonstante in der PRINT#-Anweisung hat die gleiche Bedeutung wie bei der CSAVE-Anweisung. Sie legt den Namen fest, mit dem die Daten auf Band gespeichert werden. Der Name kann aus bis zu 7 Zeichen bestehen.

Die Speicherung kann nur im RUN-Mode sowohl manuell als auch programmgesteuert erfolgen.

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

Geben Sie einen einzelnen Variablennamen oder eine Folge von Variablennamen an, so werden die Inhalte der Variablen auf Band gespeichert.

Die einzelnen Formate der PRINT#-Anweisung haben folgende Wirkung:

PRINT# VARIABLE

Der Wert der definierten Standardvariablen wird abgespeichert

```
z. B. PRINT# A
      PRINT# K
      PRINT# Z1$
```

PRINT# VARIABLE*

Die Werte der Standardvariablen, beginnend mit der definierten Standardvariablen, werden abgespeichert. Sind A-Vektor-Elemente definiert, so werden auch diese abgespeichert.

z. B. PRINT A*
Es werden alle Standardvariablen von A - Z abgespeichert

PRINT K*
Es werden die Standardvariablen von K - Z abgespeichert

PRINT# FELDVARIABLE (*)
Alle Elemente der definierten Feldvariablen werden abgespeichert.
Einzelne Elemente können nicht auf Band gespeichert werden.

z. B. PRINT B(*)
PRINT AI (*)
PRINT YZ$ (*)

PRINT # VARIABLENLISTE

Die Werte der Variablen werden in der Reihenfolge ihres Aufrufs abgespeichert

z. B. PRINT # A*, B$(*), AZ(*)

PRINT# TEXTKONSTANTE; VARIABLEN (LISTE)

Diese Anweisung verhält sich wie die vorhergenannten, die Textkonstante spezifiziert einen Dateinamen, unter dem die Variablen abgespeichert werden.

z. B. PRINT# "DATEI"; Y*, A5(*)

RANDOM

Funktion:

Setzt einen Startpunkt für den Zufallszahlengenerator.

Syntax:

RANDOM

Bemerkungen:

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

Beispiel:

```
10 RANDOM
20 FOR I = 1 TO 10
30 X=RND(5)
40 WAIT 59:PRINT X
50 NEXT I
60 END
```

RCP

Funktion:

Ergibt den Reziprokwert eines numerischen Ausdrucks.

Syntax:

RCP numerischer Ausdruck

Beispiel:

```
10 A=100
20 B=RCP A
30 PRINT"X=";A;"RCP X=";B
40 END
```

Ausgabe nach RUN:

```
X=100.RCP X=0.01
```

READ

Funktion:

Weist den angegebenen Variablen Werte aus den DATA-Zeilen zu.

Syntax:

READ numerische Variable
READ Textvariable
READ Variablenliste

Bemerkungen:

Mit der ersten READ-Anweisung wird der erste Wert der DATA-Zeilen der ersten nach der READ-Anweisung aufgeführten Variablen zugeordnet. Die zweite READ-Anweisung weist den zweiten Wert der DATA-Zeilen der entsprechenden Variablen zu. Dabei können numerische Ausdrücke nur numerischen Variablen und Textausdrücke nur Textvariablen zugeordnet werden.

Folgt der READ-Anweisung eine Variablenliste (Reihe von numerischen Variablen und/oder Textvariablen), so werden diesen der Reihenfolge nach die Elemente der DATA-Zeilen zugewiesen.

Sind Variablentyp und Ausdruck nicht identisch (z.B. READ A / DATA "SHARP"), erscheint Fehlermeldung 9.

Ist in der DATA-Anweisung ein arithmetischer Ausdruck (z.B. DATA A+B) enthalten, so wird dieser zuerst berechnet und dann der entsprechenden Variablen zugewiesen.

Hinweise:

1. Grundsätzlich müssen in den DATA-Zeilen 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 der DATA-Zeiger auf dem zuletzt eingelesenen Element.

4. DATA—Anweisungen in der ersten Programmzeile werden ignoriert. Daher dürfen DATA-Anweisungen erst ab der zweiten Programmzeile vorkommen.

REC

Funktion:

Umwandlung von Polarkoordinaten in rechtwinklige Koordinaten.

Syntax:

REC (numerischer Ausdruck, numerischer Ausdruck)

Bemerkungen:

Der erste numerische Ausdruck gibt den Radius an, der zweite den Winkel.
Die berechneten Werte für den x- und den y-Wert werden den Standardvariablen Z (x-Wert) und Y (y-Wert) zugeordnet.
Der Wert des eingegebenen Winkels ist von der Winkeleinheit (DEGREE, GRAD, RADIAN) abhängig.
Im BASIC-Programm muß die REC-Funktion einer Variablen zugewiesen werden. Dabei darf die Variable Z nicht verwendet werden. Um Speicherplatz zu sparen, kann die Variable Y als Zuordnungsvariable gewählt werden.

Beispiel:

```
10:  DEGREE
20:  INPUT "RADIUS R= "; A
30:  INPUT "WINKEL= ";B
40:  Y=REC (A,B)
50:  PRINT "X-WERT= ";Z
60:  PRINT "Y-WERT= ";Y
70:  END
```

REM

Funktion:

Definiert eine Kommentarzeile im Programm.

Syntax:

REM Textausdruck < Kommentar >

Bemerkungen:

An jeder beliebigen Stelle des Programms kann zur Erläuterung eine kommentierende Textzeile eingefügt werden. Die Zeile wird bei der Programmausführung übersprungen.

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

Beispiel: 10 X=45 REM:INITIALISIERUNG: A = 0

In diesem Beispiel wird die Wertzuweisung, die hinter der REM-Anweisung steht, als Kommentar aufgefaßt und nie ausgeführt.

Beispiele:

```
10 REM ************************
20 REM *                      *
30 REM *       T E X T        *
40 REM *                      *
50 REM ************************
```

```
10 READ P: REM PREIS PER STOCK
20 READ N: REM ANZAHL
30 C=P*N:REM GESAMTPREIS
40 END
```

RESTORE

Funktion:

Ermöglicht das erneute Lesen der DATA-Anweisungen von Beginn an.

Syntax:

RESTORE
RESTORE numerischer Ausdruck (Zeilennummer)
RESTORE Textausdruck (Markenname)

Bemerkungen:

Die RESTORE-Anweisung setzt den Zeiger für die DATA-Anweisungen zur ersten DATA-Anweisung zurück, so daß die DATA-Anweisungen wieder von Beginn an gelesen werden können.
Der Wert des numerischen Ausdrucks gibt die Zeilennummer, der Textausdruck den Markennamen der READ-Anweisung an.

Beispiel:

```
10 READ A,B,C
20 RESTORE
30 READ D,E,F,G,H,I
40 FOR Z = 1 TO 9
50 WAIT 59 : PRINT A(Z)
60 NEXT Z
70 DATA 10,20,30,40,50,60
```

Ausgabe nach RUN:

```
 10.
 20.
 30.
 10.
 20.
 30.
 40.
 50.
 60.
```

RETURN

Funktion:

Beendet einen Unterprogrammaufruf (GOSUB).

Syntax:

RETURN

Bemerkungen:

Die RETURN-Anweisung in einem Unterprogramm (Subroutine) beendet den Unterprogrammaufruf. Die Programmausführung wird mit der Anweisung, die dem Unterprogrammaufruf (GOSUB) folgt, fortgesetzt. Unterprogramme können an jeder beliebigen Stelle im Programm stehen, es ist aber empfehlenswert, sie deutlich vom Hauptprogramm abzuheben.

Ein Unterprogramm muß unbedingt mit RETURN abgeschlossen werden. Die GOTO-Anweisung ist nicht zulässig.

Eine RETURN-Anweisung ohne GOSUB-Anweisung führt zu der Fehlermeldung ERROR 5.

Beispiel:

```
 5 WAIT 200
10 PRINT "HAUPTPROGRAMM"
20 GOSUB 100
30 PRINT "HAUPTPROGRAMM"
40 END
100 PRINT"UNTERPROGRAMM"
110 A$ = INKEY$
120 IF A$ = "" THEN 110
130 RETURN
```

Ausgabe nach RUN:

```
HAUPTPROGRAMM

UNTERPROGRAMM

                    < Alphataste >

HAUPTPROGRAMM
```

RND

Funktion:

Erzeugt eine Pseudozufallszahl.

Syntax:

RND numerischer Ausdruck

Bemerkungen:

Die Ganzzahlenstelle 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
                Programmlauf generiert
0 < = X < 1 --- 0 < = RND X < 1
X > = 1     --- 1 < = RND X < = X (wenn X Ganzzahl)
                1 < = RND X < = (INT X) + 1 (wenn X Dezimalbruch)
```

Die Genauigkeit der Zufallszahl beträgt 10 Stellen.

Die mit der RND-Anweisung erhaltenen Zahlen sind nicht echt zufällig, da sie aufgrund eines festgelegten Rechenganges ermittelt werden. Sie erscheinen jedoch als zufällig und haben auch dieselben statistischen Eigenschaften wie echte Zufallszahlen.

Nach dem Einschalten erzeugt der **Computer** immer dieselbe Folge von Zufallszahlen.

Um eine neue Zufallsreihenfolge zu initiieren, muß die RANDOM-Anweisung zusätzlich zur RND-Anweisung eingegeben werden.

Beispiel:

```
10 WAIT 200
20 PRINT"ZUFALLSZAHL:";RND 6
30 GOTO 20
```

Ausgabe nach RUN:

```
ZUFALLSZAHL:X
```

nach ca. 2 Sek wird die nächste Zufallszahl angezeigt.

RUN/GOTO/Definable Keys

Funktion:

Starten der Programmausführung

Syntax:

RUN
RUN numerischer Ausdruck
RUN Textausdruck
GOTO numerischer Ausdruck
GOTO Textausdruck

Bemerkungen:

Nach der Eingabe der RUN-Anweisung und Drücken der [ENTER] Taste wird das Programm mit der niedrigsten Zeilennummer gestartet. Ist eine Zeilennummer (numerischer Ausdruck) oder ein Markenname (Textausdruck) explizit angegeben, wird die Programmausführung mit der spezifizierten Zeile gestartet.

Die RUN-Anweisung kann nur im RUN-Mode eingegeben werden. Ansonsten wird die Fehlermeldung ERROR 9 angezeigt.

Eine weitere Möglichkeit, ein Programm zu starten, bietet die GOTO-Anweisung. Sie ist nur in Verbindung mit einer Zeilennummer (numerischer Ausdruck) oder einem Markennamen (Textausdruck) zulässig.

Die dritte Möglichkeit, ein Programm zu starten, sind die 'Definable Keys' (definierbare Tasten).
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 (z.B. "A") enthält. Wird eine im Programm nicht enthaltene Marke angegeben, führt dies zur Fehlermeldung ERROR 9.
Wartet der Rechner nach einer INPUT-Anweisung auf eine Eingabe, so kann durch Drücken der Taste [DEF] und einer der 'Definable Keys' zu einer anderen Programmzeile verzweigt werden. Die INPUT-Anweisung wird dann nicht ausgeführt.

Unterschiede zwischen RUN, GOTO und den 'Definable Keys'

Beim Start des Programms werden bestimmte Grundzustände des Rechners gesetzt. Die drei Anweisungen zum Starten eines Programms unterscheiden sich hierbei, wie aus der folgenden Tabelle ersichtlich:

Zustandsänderung	RUN	GOTO	Def. Keys
Die Anzeige wird gelöscht	ja	ja	ja
Die Feldvariablen werden gelöscht	ja	nein	nein
Die RESTORE-Anweisung wird ausgeführt	ja	nein	nein
Das USING-Format wird gelöscht	ja	nein	nein
Das WAIT-Intervall bleibt erhalten	ja	nein	nein
PRINT = LPRINT wird gelöscht	ja	nein	nein

Bei allen drei Startbefehlen wird:

- die FOR-NEXT-Information gelöscht
- die GOSUB-Rücksprungadresse gelöscht
- der Standardvariablen-Speicher **nicht** gelöscht
- der TRACE-Zustand nicht geändert.

Beispiele:

```
RUN

RUN 100

RUN "PROG.1"

GOTO 100

GOTO "PROG.1"

DEF A
```

SGN

Funktion:

Ermittelt das Vorzeichen eines numerischen Ausdrucks.

Syntax:

SGN numerischer Ausdruck

Bemerkungen:

```
SGN X =  1 wenn X > 0
SGN X =  0 wenn X = 0
SGN X = -1 wenn X < 0
```

Beispiel:

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

Ausgabe nach RUN:

```
 15.     1.          ENTER
  0.     0.          ENTER
-15.    -1.
```

SIN

Funktion:

Berechnet den Sinus eines numerischen Ausdrucks in der angegebenen Winkeleinheit.

Syntax:

SIN numerischer Ausdruck

Bemerkungen:

Die Berechnung der Sinusfunktion kann in folgenden drei Winkeleinheiten erfolgen:
DEGREE (0 bis 90°)
GRAD (0 bis 100ᵍ)
RADIAN (0 bis PI/2)
Die gewählte Winkeleinheit wird am Display angezeigt.

Beispiel:

```
200:  DEGREE
210:  INPUT X
220:  LET Y=SIN X
225:  A$="#.##^^"
230:  PRINT "SIN" ;X;"=";
      USING A$;Y
```

Ausgabe nach RUN:

```
?            45 [ENTER]

SIN45. = 7.07E-01
```

SQR

Funktion:

Berechnet die Quadratwurzel eines numerischen Ausdrucks.

Syntax:

SQR numerischer Ausdruck

Bemerkungen:

Der Radikand (numerischer Ausdruck unter der Wurzel) darf nicht negativ sein.

Beispiel:

```
10 A = SQR 2
20 B = SQR A
30 PRINT A,B
40 END
```

Ausgabe nach RUN:

```
1.41421 1.18920
```

SQU

Funktion:

Berechnet das Quadrat eines numerischen Ausdrucks.

Syntax:

SQU numerischer Ausdruck

Beispiele:

(1)
```
10 A = SQU 3
20 PRINT A
30 END
```

Ausgabe nach RUN:

```
 9.
```

(2)
```
10 X = 25
20 Y = SQU X
30 PRINT X,Y
40 END
```

Ausgabe nach RUN:

```
 25.    625.
```

STOP

Funktion:

Unterbricht die Programmausführung.

Syntax:

STOP

Bemerkungen:

Durch die STOP-Anweisung wird die Programmausführung unterbrochen. Auf der Anzeige erscheint:

BREAK IN XXX (XXX ... aktuelle Zeilennummer)

Alle Zustände des Rechners (FOR...NEXT, GOSUB, DIM usw.) bleiben erhalten. Dadurch ist es möglich, einen Programmablauf zu unterbrechen (STOP oder BREAK), um die aktuellen Werte von Variablen zu überprüfen und gegebenenfalls zu verändern.

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

Bei längeren Programmen ist es ratsam, die STOP-Anweisung an das Ende von logischen Programmblöcken zu setzen, so daß die Suche nach eventuellen Programmfehlern erheblich erleichtert wird.

Durch Drücken der [↑] -Taste wird die zuletzt ausgeführte Programmzeile zur Anzeige gebracht.

Durch Betätigen der [↓] -Taste wird die Programmausführung zeilenweise fortgesetzt, wobei als erstes die angezeigte Zeile abgearbeitet und dann die Zeilennummer angezeigt wird.

Betätigt man erneut die [↑] -Taste, so wird wieder die zuletzt ausgeführte Programmzeile angezeigt.

Beispiel:

```
10 PRINT 10
20 STOP
30 PRINT 30
40 PRINT 40
50 PRINT 50
60 PRINT 60
70 PRINT 70
```

Ausgabe nach RUN:

```
              10.  ENTER
BREAK IN 20
                   ↑
20: STOP
                   ↓
              30.
                   ↑
30: PRINT 30
                   ↓
30:
  :                ↑
  :
70: PRINT 70
```

TAN

Funktion:

Berechnet den Tangens eines numerischen Wertes in der angegebenen Winkeleinheit.

Syntax:

TAN numerischer Ausdruck

Bemerkungen:

Die Berechnung der Tangensfunktion kann in folgenden drei Winkeleinheiten erfolgen:
DEGREE (0 bis 90°)
GRAD (0 bis 100^g)
RADIAN (0 bis PI/2)
Die gewählte Winkeleinheit wird am Display angezeigt.

Beispiel:

```
10 DEGREE : INPUT X
20 LET Y = TAN X
30 USING "##.###"
40 PRINT"TAN X=";Y
```

Ausgabe nach RUN:

```
?                          1 ENTER

TAN X= 0.017
```

TEN

Funktion:

Exponentialfunktion 10^X.

Syntax:

TEN numerischer Ausdruck

Beispiel:

```
10 A = LOG 100
20 B = TEN A
30 PRINT B
40 END
```

Ausgabe nach RUN:

```
 100.
```

TRON

Funktion:

Einschalten des TRACE-Betriebs.

Syntax:

TRON

Bemerkungen:

Nach Eingabe der TRON-Anweisung wird das Programm wie üblich mit RUN gestartet. Die Programmausführung wird aber nach jeder Zeile automatisch gestoppt, und die entsprechende Zeilennummer wird angezeigt.

Mit der [↑] -Taste kann der jeweilige Zeileninhalt zur Anzeige gebracht, mit der [↓] -Taste die Programmausführung fortgesetzt werden.

Hält man die [↓] -Taste fest, so erfolgt die Programmausführung schnell hintereinander.

Die TRON- und TROFF-Anweisungen können auch im Programm enthalten sein, wodurch begrenzte Programmteile überprüft werden können.

Der TRACE-Betrieb kann durch Ausschalten des Computers, durch die TROFF-Anweisung oder durch Drücken der Tasten [SHIFT] und [CA CL] wieder ausgeschaltet werden.

Beispiel:

```
10 TRON
20 FOR I = 1 TO 3
30 NEXT I
40 TROFF
```

TROFF

Funktion:

Ausschalten des TRACE-Betriebs.

Syntax:

TROFF

Bemerkungen:

Der TRACE-Betrieb (Programm-Ablaufverfolgung) wird ausgeschaltet.

Beispiel:

```
10 TRON
20 FOR I = 1 TO 3
30 NEXT I
40 TROFF
```

USING

Funktion:

Legt das Ausgabeformat von Textausdrücken und Werten numerischer Ausdrücke im Anschluß an eine PRINT- oder LPRINT-Anweisung fest.

Syntax:

USING ;
USING Format;

Bemerkungen:

Der **Computer** wählt entsprechend dem auszugebenden Wert automatisch ein Ausgabeformat.

Mit der USING-Anweisung kann diese automatische Auswahl abgeschaltet werden, und Sie können selbst bestimmen, in welchem Format die Ausgabe erfolgen soll.

Die USING-Anweisung ist für alle PRINT- oder LPRINT-Anweisungen wirksam.

Das Format kann als Textkonstante oder als Textvariable eingegeben werden. Es bleibt für alle nachfolgenden Ausgabe-Anweisungen erhalten, bis ein neues Format festgelegt wird.

Wird die USING-Anweisung ohne Format eingegeben, so wird die letzte Formatspezifizierung unwirksam und die automatische Formatierung wieder eingeschaltet.

Dasselbe bewirkt die Ausführung der RUN-Anweisung oder das Drücken der Tasten SHIFT und CA CL .

Zur Festlegung des Formats dienen folgende Sonderzeichen:

(#)... bestimmt die Anzahl der Stellen im numerischen Ausgabefeld.

(.)... legt die Stelle des Dezimalpunktes fest. Die Folge der #-Zeichen nach dem Dezimalpunkt legt die Anzahl der Nachkommastellen fest. In den Nachkommastellen werden fehlende Stellen immer mit Null aufgefüllt. Überzählige Stellen des auszugebenden Wertes werden ignoriert.

(^)... bestimmt die Ausgabe im wissenschaftlichen Format. Dabei können maximal neun Nachkommastellen bestimmt werden.

(&)... bestimmt die Anzahl der Zeichenstellen von Texten. Ist das spezifizierte Format zu klein, so werden nur die formatierten Stellen des Textausdrucks angezeigt. Ist das spezifizierte Format zu groß, so werden die überzähligen Stellen mit Leerstellen aufgefüllt.

Im folgenden sind die unterschiedlichen Formatfestlegungen der USING-Anweisung aufgeführt:

(1) "##...#"

- Anzahl der Ganzzahlenstellen
- Vorzeichenstelle (wird nur bei negativen Zahlen angezeigt)

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

- Dezimalpunktausgabe
- Anzahl der Vorkommastellen
- Vorzeichenstelle

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

- Anzahl der Nachkommastellen
- Dezimalpunkt
- Anzahl der Vorkommastellen
- Vorzeichenstelle

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

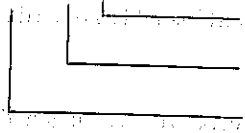
- wissenschaftliches Format
- Mantisse
- Vorzeichenstelle

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

- Anzahl der Textstellen

(6) Kombinierte Formatierung

a) "##.#^&&&"

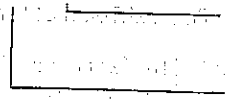


drei Textstellen

wissenschaftliches Format

Vorzeichenstelle

b) "&&&&##.##"



eine Vorkommastelle, zwei Nachkommastellen, Vorzeichenstelle

vier Textstellen

Hinweise:

1. Reicht das mit der USING-Anweisung festgelegte Ausgabeformat für die Darstellung einer Ganzzahl nicht aus, so wird die Fehlermeldung ERROR 7 angezeigt.

2. Fehlerhafte Formate werden meist erst bei den Ausgabeanweisungen (PRINT oder LPRINT) erkannt und angezeigt.

Beispiele:

(1)
```
10 A = 123.456789
20 USING "####.#"
30 PRINT A
40 USING
50 PRINT A
```

Ausgabe nach RUN:

```
 123.4
```

ENTER

```
123.456789
```

In der Programmzeile 40 wird auf automatische Formatierung umgeschaltet.

VAL

Funktion:

Berechnet den Wert einer als Zeichenfolge angegebenen Zahl.

Syntax:

VAL Textausdruck

Bemerkungen:

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

Beispiele:

(1)
```
10 A = VAL "12.3E12"
20 PRINT A
30 END
```

Ausgabe nach RUN:

```
 1.23E 13
```

(2)
```
10 A = VAL "1 2 3"
20 PRINT A
30 END
```

Ausgabe nach RUN:

```
  123.
```

(3)
```
10 A$ = "4 7 1 1"
20 B = VAL A$
30 PRINT B
40 END
```

Ausgabe nach RUN:

```
  4711.
```

WAIT

Funktion:

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

Syntax:

WAIT num. Ausdruck < Zeitintervall >

Bemerkungen:

Normalerweise wird der Programmablauf nach einer PRINT-Anweisung mit der Eingabe von ENTER fortgesetzt. Mit der WAIT-Anweisung kann ein Zeitintervall festgelegt werden, nach dessen Ablauf die Ausführung automatisch wieder gestartet wird.

Der numerische Ausdruck legt das Intervall fest. Dabei gilt:

1 Einheit = 1/59 Sekunde.

Der Wert des numerischen Ausdrucks darf im Bereich zwischen 0 und 65535 liegen. Damit ergeben sich Zeitintervalle im Bereich von 0 Sekunden bis zu ca. 19 Minuten (= 65535 * 1/59 = 1110,8 Sekunden).

Die WAIT-Anweisung gilt für alle folgenden PRINT-Anweisungen.

Folgt der WAIT-Anweisung kein numerischer Ausdruck, so wird die Automatik abgeschaltet, und die Ausführung muß wieder mit fortgesetzt werden.

Beispiel:

```
10 WAIT 590
20 PRINT"SHARP"
30 PRINT"HAMBURG"
40 END
```

Ausgabe nach RUN:

SHARP

und nach 10 Sekunden:

HAMBURG

ANHANG

ANHANG A TASTENFUNKTIONEN DES PC-1430

ARITHMETISCHE FUNKTIONEN

Funktionsname		Funktion
ABS		Absolutwert
ACS	SHIFT ACS	Arcuscosinus
ASN	SHIFT ASN	Arcussinus
ATN	SHIFT ATN	Arcustangens
COS	COS	Cosinus
CUB	CUB	Kubikzahl
CUR	SHIFT CUR	Kubikwurzel
DEG	DEG	Sexagesimalumrechnung
DEGREE		Winkeleinheit Grad
DMS	SHIFT DMS	Sexagesimalumrechnung
EXP	SHIFT EXP	Exponentialumrechnung
FACT	SHIFT FACT	Fakultät
GRAD		Winkeleinheit Neugrad
HEX→DEC	SHIFT &	Sedezimalumrechnung
INT		Ganzzahlfunktion
LN	LN	Natürlicher Logarithmus
LOG	LOG	Dekadischer Logarithmus
MEM		Freier Speicherplatz
PI	π	Kreiskonstante PI
POL	SHIFT POL	Polarkoordinatenumrechnung
RADIAN		Winkeleinheit Radiant
RANDOM		Zufallsgeneratoranfangswert
RCP	RCP	Reziprokwert
REC	SHIFT REC	Polarkoordinatenumrechnung
RND		Zufallszahl
SGN		Vorzeichen
SIN	SIN	Sinus
SQU	SQU	Quadratzahl
SQR	SHIFT √	Quadratwurzel
DEC	SHIFT DEG	Sedezimalumwandlung
TAN	TAN	Tangens
TEN	SHIFT TEN	Exponentiation zur Basis 10

Tastenbetätigung		BASIC-Anweisung
SHIFT	A	INPUT
SHIFT	S	IF
SHIFT	D	THEN
SHIFT	F	GOTO
SHIFT	G	FOR
SHIFT	H	TO
SHIFT	J	STEP
SHIFT	K	NEXT
SHIFT	L	LIST
SHIFT	=	RUN
SHIFT	Z	PRINT
SHIFT	X	USING
SHIFT	C	GOSUB
SHIFT	V	RETURN
SHIFT	B	DIM
SHIFT	N	END
SHIFT	M	CSAVE
SHIFT	SPC	CLOAD

ANHANG B LISTE DER FUNKTIONEN UND ANWEISUNGEN

ARITHMETHISCHE FUNKTIONEN

ABS
ACS
ASN
ATN
COS
CUB
CUR
DEGREE/GRAD/RADIAN
DEG
DMS
EXP
FACT
INT
LN
LOG
MDF
MEM
PI
POL
RANDOM
RCP
REC
RND
SGN
SIN
SQR
SQU
TEN
TAN

TEXTFUNKTIONEN

INKEY$
LEN
MID$
VAL

PROGRAMM-ANWEISUNGEN

CLEAR
CONT
DIM
Definable Keys
END
FOR...TO...STEP/NEXT
GOSUB
GOTO
IF...THEN
LET
NEW
NEW Ø
REM
RETURN
RUN/GOTO
STOP
TRON
TROFF

EINGABE-/AUSGABE-ANWEISUNGEN

DATA
INPUT
LIST
PRINT
READ
RESTORE
USING
WAIT

EIN-/AUSGABEANWEISUNGEN MIT DER OPTION CE-126P

INPUT#
PRINT#

DRUCKERFUNKTIONEN (OPTION CE-126P)

LLIST
LPRINT

PROGRAMMSPEICHERUNG AUF MAGNETBAND (OPTION CE-126P)

CLOAD
CLOAD?
CSAVE

PROGRAMMSCHUTZ

PASS

ANHANG C BASIC BEFEHLSOBERSICHT (Kurzfassung)

ARITHMETHISCHE FUNKTIONEN

Funktion	Wirkung	Beispiele
ABS (Ausdruck)	Ergibt den Absolutbetrag eines numerischen Ausdrucks.	ABS(L*0.7) ABS(SIN(X))
ATN (Ausdruck)	Ergibt den Arcustangens eines numerischen Ausdrucks.	ATN(3.5) ATN(A*0.8)
DEGREE/GRAD/RADIAN	Es wird die Winkeleinheit festgelegt, mit der die trigonometrischen Funktionen verarbeitet werden. Die gewählte Winkeleinheit wird angezeigt.	DEGREE RADIAN GRAD
DMS (Ausdruck)	Ermöglicht die Umrechnung vom Dezimalsystem ins Sexagesimalsystem.	DMS 15.2445
DEG (Ausdruck)	Ermöglicht die Umrechnung vom Sexagesimalsystem ins Dezimalsystem.	DEG 15.4125
SIN (Ausdruck)	Ergibt den Sinus eines numerischen Ausdrucks.	SIN 0.755 SIN (A/B)
COS (Ausdruck)	Ergibt den Cosinus eines numerischen Ausdrucks.	COS 0.755 COS (A/B)
TAN (Ausdruck)	Ergibt den Tangens eines numerischen Ausdrucks.	TAN X TAN (X*0.145)
ACS (Ausdruck)	Ergibt den Arcuscosinus eines numerischen Ausdrucks.	ACS 0.8 ACS (A*B)
ASN (Ausdruck)	Ergibt den Arcussinus eines numerischen Ausdrucks.	ASN 0.8 ASN (A*B)
ATN (Ausdruck)	Ergibt den Arcustangens eines numerischen Ausdrucks.	ATN 0.8 ATN (A*B)

Funktion	Wirkung	Beispiele
LN (Ausdruck)	Ergibt den natürlichen Logarithmus (zur Basis e) eines numerischen Ausdrucks (Ausdruck muß positiv sein).	LN 60 LN (A+B)
LOG (Ausdruck)	Ergibt den Zehnerlogarithmus (zur Basis 10) eines numerischen Ausdrucks (Ausdruck muß positiv sein).	LOG 100 LOG (A+B)
INT (Ausdruck)	Ergibt den abgerundeten ganzzahligen Wert eines numerischen Ausdrucks.	INT (A*B*C)
SGN (Ausdruck)	Ergibt -1 für einen negativen Ausdruck; 0 für einen Null-Ausdruck und +1 für einen positiven Ausdruck.	SGN (-12) SGN (A*B-3)
SQR (Ausdruck)	Ergibt die Quadratwurzel des numerischen Ausdrucks. (Ausdruck muß positiv sein)	SQR (A*B-C)
PI	Die Konstante PI ist als 3.141592654 gespeichert.	
RND (Ausdruck)	Ergibt eine Pseudo-Zufallszahl zwischen 1 und der Ganzzahlenstelle des Ausdrucks. Grenzen: $1 \leq$ Ausdruck ≤ 32768	RND 80 RND (X+Y)
RANDOM	Setzt den Zufallsgenerator auf einen neuen Anfangswert.	RANDOM
MEM	Ergibt die Anzahl der ungenutzten Bytes im Hauptspeicher.	MEM
RCP	Ergibt den Reiprokwert eines numerischen Ausdrucks.	RCP 7
POL	Rechnet Polarkoordinaten in rechtwinklige Koordinaten um.	Y=POL(3,5)
REC	Rechnet rechtwinklige Koordinaten in Polarkoordinaten um.	REC (2,45)

TEXTFUNKTIONEN

Funktion	Wirkung	Beispiele
LEN Textausdruck	Ermittelt die Länge (Anzahl der Zeichen) eines Textausdrucks.	LEN A$
VAL Textausdruck	Ergibt einen numerischen Wert entsprechend der im Textausdruck enthaltenen Zahl.	VAL "1 2 3" VAL A$+B$
MID$(Textausdruck,p,n)	Ergibt den mittleren Teil eines Textausdrucks mit der Zeichenzahl n ab dem p-ten Zeichen.	MID$(A$,5,1) MID$(M$+B$,P,4)
INKEY$	Diese Funktion fragt die Tastatur ab und ergibt je nach gedrückter Taste einen Textausdruck von einem Zeichen Länge bzw. einen leeren Textausdruck, wenn keine Taste gedrückt wurde.	A$ = INKEY$

EINGABE-/AUSGABE-ANWEISUNGEN

Funktion	Wirkung	Beispiele
PRINT Ausdruck	Mit dieser Anweisung werden numerische oder Textausdrücke auf der Anzeige ausgegeben.	PRINT A PRINT A$ PRINT "SHARP"
,	Über das Komma wird die Aufteilung der 16-spaltigen Anzeige in zwei gleich große Felder gesteuert.	
;	Das Semikolon trennt die einzelnen Elemente einer Liste von Ausdrücken und steuert die Position des Cursors.	

Funktion	Wirkung	Beispiele
INPUT "Meldung";(,) Variable	Bringt die "Meldung" (falls angegeben) zur Anzeige und weist der Variable Werte zu die über die Tastatur eingegeben werden.	INPUT A INPUT C,A$ INPUT "WERT";B INPUT "NAME";B
USING Formatzeichen	Diese Anweisung definiert das Ausgabeformat von Texten und Werten numerischer Ausdrücke im Anschluß an eine PRINT-Anweisung. Zur Festlegung des Formats dienen die folgenden Zeichen:	
	# bestimmt die Anzahl der Stellen im numerischen Ausgabefeld. Grenzen: 11 Stellen für Ganzahlen (inkl. Vorzeichen) 12 Stellen für Dezimalzahlen	USING"####"
	. legt die Stelle des Dezimalpunktes fest.	USING"###.##"
	bestimmt die Ausgabe im wissenschaftlichen Format. Es sind maximal 9 Nachkommastellen zulässig.	USING"##.# "
	& bestimmt die Anzahl der Zeichenstellen von Texten und Textausdrücken.	USING"&&&&&&&"
	^ bestimmt die Ausgabe nach dem wissenschaftlichen Format. Dabei können maximal neun Nachkommastellen bestimmt werden.	USING"##.#^"
WAIT Zeitintervall	Diese Anweisung legt das Zeitintervall fest, das vergehen soll bevor nach einer PRINT-Anweisung der Programmablauf wieder startet. Hinweis: Die WAIT-Anweisung wirkt auf alle folgenden PRINT-Anweisungen.	WAIT 54.4
DATA Konstantenliste	Diese Anweisung stellt Daten zur Eingabe mit der READ-Anweisung zur Verfügung.	DATA 4.15,7.45 DATA "SHARP", "ENDE"

Funktion	Wirkung	Beispiele
READ Variablenliste	Diese Anweisung weist den angegebenen Variablen Werte aus den DATA-Zeilen zu.	READ A,B,C READ A$,B$,C$
RESTORE	Setzt den DATA-Zeiger auf das erste Element der ersten DATA-Anweisung zurück.	RESTORE RESTORE 20 RESTORE "xxx"

EIN-/AUSGABEANWEISUNGEN MIT DER OPTION CE-126P

Funktion	Wirkung	Beispiele
PRINT# Variable	Mit dieser Anweisung werden die Werte der Variablen auf Band gespeichert.	PRINT# C PRINT# J,B$(*)
INPUT# Variable	Mit dieser Anweisung werden die auf Band gespeicherten Werte gelesen und den Variablen zugeordnet.	INPUT# C INPUT# J,B$(*)
LPRINT Variable	Mit dieser Anweisung werden numerische Werte oder Textausdrucke auf dem Drucker ausgegeben.	LPRINT A LPRINT "SHARP" LPRINT A$;B$;C

PROGRAMM-ANWEISUNGEN

Funktion	Wirkung	Beispiele
CLEAR	Diese Anweisung löscht alle Variablen und Felder im Hauptspeicher und setzt die Standardvariablen auf 0 (Null)	CLEAR
NEW	Diese Anweisung löscht den Hauptspeicher außer Programme, die durch PASS-Wort geschützt sind.	NEW
NEWØ	Wie NEW, löscht aber Programme, die durch PASS-Wort geschützt sind.	
RUN/GOTO	Mit diesen Anweisungen wird die Programmausführung gestartet.	RUN RUN 200 RUN "TEST" GOTO 200 GOTO "TEST"

Funktion	Wirkung	Beispiele
Definable Keys (Definierbare Tasten)	Eine weitere Möglichkeit, ein Programm zu starten, sind die sogenannten 'Definable Keys'. Dazu muß die DEF -Taste und die dem jeweiligen Programm zugeordnete Taste gedrückt werden.	DEF A
STOP	Diese Anweisung unterbricht die Programmausführung	STOP
CONT	Diese Anweisung setzt die Programmausführung nach einer Unterbrechung durch eine STOP-Anweisung oder durch BREAK fort.	CONT
END	Diese Anweisung beendet die Programmausführung	END
LIST Zeilennummer "Markenname"	Mit dem Kommando LIST kann jeweils eine Programmzeile zur Anzeige gebracht ('gelistet') werden.	LIST LIST 100 LIST "LABEL 1"
TRON	Trace (Ablaufverfolgung) einschalten. ↑ -Taste: Zeileninhalt anzeigen ↓ -Taste: Programmausführung fortsetzten	TRON
TROFF	Trace (Ablaufverfolgung) ausschalten.	TROFF
GOTO Zeilennummer "Markenname"	Diese Anweisung verzweigt zur angegebenen Zeilennummer oder zum angegebenen Markennamen	GOTO 100 GOTO "LABEL 1"

Funktion	Wirkung	Beispiele
GOSUB Zeilennummer Markenname	Diese Anweisung verzweigt zu dem Unterprogramm mit der angegeben Zeilennummer oder dem angegebenen Markennamen.	GOSUB 200 GOSUB "LABEL 1"
RETURN	Diese Anweisung beendet ein Unterprogramm. Die Programmausführung springt zur Adresse des letzten Unterprogrammaufrufs zurück.	RETURN
IF...THEN	Die Anweisung ermöglicht eine Verzeigung des Programms in Abhängigkeit vom Ergebnis eines logischen Vergleichsausdrucks.	IF A = 3 THEN 10 IF A$ = "X" THEN 100
FOR...TO...STEP/NEXT	Diese Anweisung dient zur wiederholten Ausführung des zwischen FOR...TO und NEXT eingeschlossenen Programmteils. Mit Hilfe der STEP-Anweisung kann die Schrittweite des FOR...TO-Ausdrucks bestimmt werden. Entfällt diese, so ist die Schrittweite mit 1 festgelegt.	FOR I=1TO10 NEXT I FOR A = 1TO10 STEP 10 NEXT A
DIM	Mit dieser Anweisung werden Feldvariable (Arrays) explizit dimensioniert.	DIM B(25) DIM B(25),B$(1) DIM Z(10,8) DIM Z$(12,5)
LET	Diese Anweisung weist einer Variablen (einfach oder indiziert) einen Wert zu.	LET A = 7 LET A$="SHARP"

Funktion	Wirkung	Beispiele
REM	Diese Anweisung definiert eine Kommentarzeile im Programm	REM TESTPRO
RANDOM	Diese Anweisung setzt den Pseudozufallsgenerator auf einen neuen Ausgangswert.	RANDOM
LLIST	Mit dieser Anweisung wird das Programm mit dem Drucker (Option CE-126P) gelistet.	LLIST

PROGRAMMSPEICHERUNG AUF MAGNETBAND

Funktion	Wirkung	Beispiele
CSAVE Programmname	Mit dieser Anweisung werden Programme auf Band gespeichert.	CSAVE CSAVE "PRG A"
	Soll das Programm als geschütztes Programm gespeichert werden, so muß nach dem Programmnamen noch ein PASS-Wort eingegeben werden.	CSAVE,"GEHEIM" CSAVE "PRG A", "GEHEIM"
CLOAD Programmname	Mit dieser Anweisung werden Programme vom Band geladen.	CLOAD CLOAD "PRG A"
CLOAD?	Mit dieser Anweisung wird das im Hauptspeicher gespeicherte Programm mit dem am Band gespeicherten Programm verglichen.	CLOAD? CLOAD? "PRG A"

PROGRAMMSCHUTZ

Funktion	Wirkung	Beispiele
PASS "Textausdruck"	Diese Anweisung schützt ein Programm vor dem Zugriff unbefugter Personen. Das PASS-Wort kann durch nochmalige Eingabe wieder gelöscht werden. Ein geschütztes Programm kann nicht auf Band gespeichert werden. Ist ein Programm nicht geschützt, so kann es als geschütztes Programm auf Band gespeichert werden (siehe CSAVE).	PASS "GEHEIM"

ANHANG D STANDARDVARIABLE

```
A = A$ = A(1)  = A$(1)
B = B$ = A(2)  = A$(2)
C = C$ = A(3)  = A$(3)
D = D$ = A(4)  = A$(4)
E = E$ = A(5)  = A$(5)
F = F$ = A(6)  = A$(6)
G = G$ = A(7)  = A$(7)
H = H$ = A(8)  = A$(8)
I = I$ = A(9)  = A$(9)
J = J$ = A(10) = A$(10)
K = K$ = A(11) = A$(11)
L = L$ = A(12) = A$(12)
M = M$ = A(13) = A$(13)
N = N$ = A(14) = A$(14)
O = O$ = A(15) = A$(15)
P = P$ = A(16) = A$(16)
Q = Q$ = A(17) = A$(17)
R = R$ = A(18) = A$(18)
S = S$ = A(19) = A$(19)
T = T$ = A(20) = A$(20)
U = U$ = A(21) = A$(21)
V = V$ = A(22) = A$(22)
W = W$ = A(23) = A$(23)
X = X$ = A(24) = A$(24)
Y = Y$ = A(25) = A$(25)
Z = Z$ = A(26) = A$(26)
```

ANHANG E OPERATOREN

E.1 Arithmetische Operatoren

+	Addition
-	Subtraktion
*	Multiplikation
/	Division
^	Exponentation

E.2 Textfunktionsoperatoren

+	Verkettung

E.3 Logische Vergleichsoperatoren

	Bedeutung in numerischen Ausdrücken	Bedeutung in Textausdrücken
<	kleiner als	ASCII-Kode kleiner
>	größer als	ASCII-Kode größer
<=	kleiner gleich	ASCII-Kode kleiner oder gleich
>=	größer gleich	ASCII-Kode größer oder gleich
<>	ungleich	ASCII-Kode ungleich

E.4 Reihenfolge der Operatoren

1. Klammern
2. Abruf von PI, MEM, Variablen
3. Funktionsoperationen in Bezug auf das folgende Argument
4. Exponentation
5. Vorzeichen + , -
6. Multiplikation, Division
7. Addition, Subtraktion
8. Vergleichsoperationen

ANHANG F RECHENBEREICH

Funktion		Rechenbereich	Anmerkung
$\sin x$ $\cos x$ $\tan x$		DEG: $\lvert x \rvert < 1 \times 10^{10}$ RAD: $\lvert x \rvert < \frac{\pi}{180} \times 10^{10}$ GRAD: $\lvert x \rvert < \frac{10}{9} \times 10^{10}$ Für tan x gelten folgende Einschränkungen DEG: $\lvert x \rvert = 90(2n-1)$ RAD: $\lvert x \rvert = \frac{\pi}{2}(2n-1)$ n = Ganze Zahl GRAD: $\lvert x \rvert = 100(2n-1)$	
$\sin^{-1} x$ $\cos^{-1} x$		$-1 \leqq x \leqq 1$	
$\tan^{-1} x$		$\lvert x \rvert < 1 \times 10^{100}$	
$\ln x$ $\log x$		$1 \times 10^{-99} \leqq x < 1 \times 10^{100}$	($\ln x = \log_e x$)
e^x		$-1 \times 10^{100} < x \leqq 230{,}2585092$	(e = 2,718281828)
10^x		$-1 \times 10^{100} < x < 100$	
$\sqrt[3]{x}$		$\lvert x \rvert < 1 \times 10^{100}$	
MDF x		$\lvert x' \rvert < 1 \times 10^{100}$ x': gerundeter Wert	
x^3		$\lvert x \rvert < 2{,}154434690 \times 10^{33}$	
$\sqrt{x}$		$0 \leqq x < 1 \times 10^{100}$	
x^2		$\lvert x \rvert < 1 \times 10^{50}$	
$\frac{1}{x}$		$\lvert x \rvert < 1 \times 10^{100}$ $x \neq 0$	
n!		$0 \leqq n \leqq 69$ (n = Ganze Zahl)	
→DEG		$\lvert x \rvert < 1 \times 10^{100}$	
→DMS		$\lvert x \rvert < 1 \times 10^{100}$	
HEX→DEC $x, y \rightarrow r, \theta$		$0 \leqq x \leqq$ 2540BE3FF FDABF41C01 $\leqq x \leqq$ FFFFFFFFFF $(x^2 + y^2) < 1 \times 10^{100}$ $\frac{y}{x} < 1 \times 10^{100}$	x ist in "HEX" eine ganze Zahl $r = \sqrt{x^2 + y^2}$ $\theta = \tan^{-1} \frac{y}{x}$
$r, \theta \rightarrow x, y$		$r < 1 \times 10^{100}$, $\lvert r \sin\theta \rvert < 1 \times 10^{100}$ $\lvert r \cos\theta \rvert < 1 \times 10^{100}$	$x = r\cos\theta$ $y = r\sin\theta$ θ ist in derselben Bedingung wie x von sin x, cos x
Statistische Berechnungen	Data CD	$\lvert x \rvert < 1 \times 10^{50}$ $\lvert \Sigma x \rvert < 1 \times 10^{100}$ $\Sigma x^2 < 1 \times 10^{100}$ $\lvert n \rvert < 1 \times 10^{100}$	
	$\bar{x}$	$n \neq 0$	
	S	$n \neq 1$ $0 \leqq \frac{\Sigma x^2 - n\bar{x}^2}{n-1} < 1 \times 10^{100}$	
	σ	$n \neq 0$ $0 \leqq \frac{\Sigma x^2 - n\bar{x}^2}{n} < 1 \times 10^{100}$	

ANHANG G ABGELEITETE MATHEMATISCHE FUNKTIONEN

Funktion	
Sekans	SEC(X) = 1/COS(X)
Cosekans	CSC(X) = 1/SIN(X)
Cotangens	COT(X) = 1/TAN(X)
Arcussinus	ARCSIN(X) = ATN(X/SQR(-X*X+1))
Arcuscosinus	ARCCOS(X) = -ATN(X/SQR(-X*X+1))+1.5708
Arcussekans	ARCSEC(X) = ATN(SQR(X*X-1))+(SGN(X)-1)*1.5708
Arcuscosekans	ARCCSC(X) = ATN(1/SQR(X*X-1))+(SGN(X)-1)*1.5708
Arcuscotangens	ARCCOT(X) = -ATN(X)+1.5708
Hyperbelsinus	SINH(X) = (EXP(X)-EXP(-X))/2
Hyperbelcosinus	COSH(X) = (EXP(X)+EXP(-X))/2
Hyperbeltangens	TANH(X) = -EXP(-X)/(EXP(X)+EXP(-X))*2+1
Hyperbelsekans	SECH(X) = 2/(EXP(X)+EXP(-X))
Hyperbelcosekans	CSCH(X) = 2/(EXP(X)-EXP(-X))
Hyperbelcotangens	COTH(X) = EXP(-X)/(EXP(X)-EXP(-X))*2+1
Hyperbel-Areasinus	ARSINH(X) = LOG(X+SQR(X*X+1))
Hyperbel-Areacosinus	ARCOSH(X) = LOG(X+SQR(X*X-1))
Hyperbel-Areatangens	ARTANH(X) = LOG((1+X)/(1-X))/2
Hyperbel-Areasekans	ARSECH(X) = LOG((SQR(-X*X+1)+1)/X)
Hyperbel-Areacosekans	ARCSCH(X) = LOG((SGN(X)*SQR(X*X+1)+1)/X)
Hyperbel-Areacotangens	ARCOTH(X) = LOG((X+1)/(X-1))/2

ANHANG H PROGRAMM-HAUPTSPEICHER-BEGRENZUNG

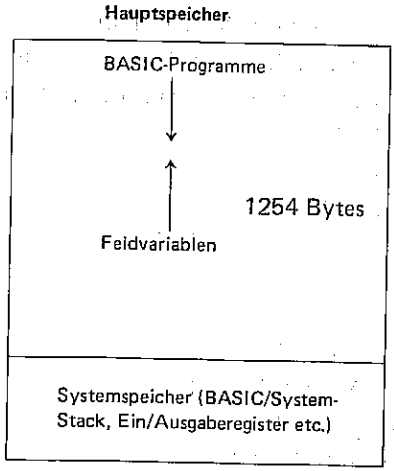
Zulässige Bereiche:

Zeichenketten: bis zu 80 Zeichen

zulässige Zeilennummern: 1 bis 65279 (einschließlich)

Länge der Programmzeile: bis zu 80 Zeichen

Speicherorganisation:



Standardvariablenspeicher (208 Bytes)

A ~ Z	=	A(1)	~	A(26) bzw.
A$ ~ Z$	=	A$(1)	~	A$(26)

ANHANG I FEHLERMELDUNGEN

ERROR Erklärung der Fehlermeldung

1 **Syntax Fehler:**
z.B. 10: PRINTT "SHARP"
Die PRINT-Anweisung wurde falsch eingegeben.

2 **Unzulässige Berechnungen:**
z.B. der Rechenbereich des **Computers** wurde überschritten, oder es wurde versucht, eine Division durch 0 einzugeben.

3 **Fehler in der Speicherbelegung:**
Es wurde z. B. versucht, eine Operation auszuführen, bei der ein Konflikt endstanden ist zwischen der Wirkung der Operation und der Speicherorganisation.
z.B. 10 FOR I = 1 TO 35000
Der Wert der Schleifenvariable ist höher als zugelassen.

4 **Fehler in der Zeilennummer:**
Sie haben eine unzulässige Zeilennummer verwendet.

5 **Schachtelungsfehler:**
Zu einer NEXT- bzw. RETURN-Anweisung exestiert keine entsprechende FOR- bzw. GOSUB-Anweisung.
z.B. 10: FOR A = 1 TO 10
20: NEXT B

6 **Speicherüberlauf:**
Die Speicherkapazität des **Computers** wurde überschritten.

7 **Formatfehler:**
z.B. 10: USING"####"
20: A=123*20
30: PRINT A

8 **Fehler bezüglich einer Option:**
Die Informationsübertragung zwischen dem **PC-1430** und der Option **CE-126P** funktioniert nicht. Batteriespannung des **CE-126P** überprüfen. Verbindung **Computer** überprüfen. Verbindung zum externen Kassettenrekorder überprüfen.

9 **Sonstige:**
Ein anderer Fehler als bisher spezifiziert ist aufgetreten.
z.B.
10: A = 5 : PRINT A$
Es kann nur entweder A oder A$ als Variable verwendet werden.

ANHANG J REFERENZLISTE

Gottfried, Byron S.

Programmieren mit BASIC
Düsseldorf: McGraw-Hill
Book Co., 1978
ISBN 0-07-092022-2

Haase, 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

ANHANG K TECHNISCHE DATEN PC-1430

Modell:	**PC-1430** Taschencomputer
Rechenstellen:	10 Stellen Mantisse, 2 Stellen Exponent
Rechensystem:	AEL mit Hierarchie
Anzeige:	Flüssigkristallanzeige, 16 Stellen in 5x7-Matrix
Tastatur:	73 Tasten Alphatastatur (Schreibmaschine) Zifferntastatur Funktionstasten
Programmiersprache:	BASIC
CPU:	CMOS 4-Bit
Betriebssystem:	ROM Ca. 17,4-kByte
Speicherkapazität:	RAM: Ca. 500 Byte System 1254 Byte BASIC-Programm 208 Byte Standardvariablen
Speicherschutz:	Programm-Daten-Speicher sind beim Ausschalten geschützt.
Stromversorgung:	6 Volt, 2 Stück Lithium-Batterien (z.B. VARTA CR2032, UCAR CR2032)
Stromverbrauch:	0,07 W bei 6 Volt
Betriebszeit:	ca. 75 Stunden mit einem Batteriesatz
Betriebstemperatur:	0°C ... 40°C
Abmessungen:	170 x 72 x 9,5 (BxHxT) in mm
Gewicht:	ca. 125g
Zubehör:	1 Tastaturabdeckung, 2 Batterien (eingebaut) 1 Tastaturschablone, 1 Bedienungsanleitung
Option:	CE-126P: Thermodrucker mit integriertem Kassetten-Interface für ein externes Bandgerät CE-152: Kassettenrecorder

ANHANG L TECHNISCHE DATEN CE-126P

Modell: CE-126 Thermodrucker mit integriertem Kassetten-Interface

1. Drucker

Zeichen/Zeile: 24

Druckgeschwindigkeit: 0,8 Zeilen/sek.

Druckprinzip: Thermodruck in 5x7-Matrix

Papiertransport: manuell oder programmgesteuert

Papierabmessungen: 58 x 18 mm Thermopapier (EA-1250P)

2. Integriertes Kassetten-Interface

An das integrierte Kassetten-Interface kann jeder handelsübliche Kassettenrecorder angeschlossen werden, der folgende Bedingungen erfüllt:

- Fernbedienung (REMOTE), Klinkenbuche 2,5 mm ø
- Bandzählwerk
- Mikrofoneingang (MIC), Klinkenbuchse 3,5 mm ø
- Ausgang (EAR), Klinkenbuchse 3,5 mm ø

Daneben sind folgende technische Daten gefordert:

- Eingangsimpedanz (MIC)	200...1000 Ohm
- Eingangsempfindlichkeit	Kleiner als 3 mV (-50 dB)
- Ausgangsimpedanz (EAR)	Kleiner als 10 Ohm
- Ausgangsleistung (EAR)	Größer als 1 V
- Klirrfaktor	Kleiner als 15 % im Bereich zwischen 2...4 kHz
- Gleichlaufschwankungen	0,3 % max. (W.R.M.S)

PROGRAMMBEISPIELE

Nach dem Durcharbeiten der Bedienungsanleitung sind Sie jetzt mit den wichtigsten Programm-Kommandos vertraut. Die beste Methode, Programmieren zu lernen, ist, selber Programme zu schreiben. Auch hier gilt, wie in allen anderen Gebieten, daß Übung den Meister macht. Aber es ist auch sehr hilfreich, Programme nachzuvollziehen, die von anderen geschrieben wurden. Aus diesem Grund haben wir einige Programmbeispiele in diese Bedienungsanleitung aufgenommen, die Ihnen hoffentlich beim Erlernen der Programmiersprache BASIC nützlich sein werden.

Hinweis: Sharp Corporation und ihre Tochtergesellschaften übernehmen keine Haftung für Verluste oder Schäden, die aus der Anwendung der Programme in dieser Bedienungsanleitung entstehen könnten.

Die Verwendung von langen Gleichungen und hohen Zahlen kann aufgrund der begrenzten Speicherkapazität des Computers zu Fehlern führen. Daher müssen Berechnungen und Daten überprüft und gegebenenfalls geändert werden.

INHALT

(Programm name) Seite

- R–L–C SCHALTKREIS IMPEDANZ 193
- SCHNITT ZWISCHEN KREISEN UND GERADEN 198
- KLOTHOIDE 202
- WURZEL EINER GLEICHUNG 205
- TRANSFORMATION VON RECHTWINKLIGEN KOORDINATEN UND POLARKOORDINATEN 208
- BERECHNUNG DER ANZAHL VON TAGEN ZWISCHEN ZWEI DATEN 211
- TABELLENPROGRAMM 214

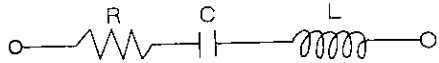
- Am Ende der einzelnen Programmlisten ist jewils die Anzahl Bytes angegeben, die zur Speicherung der Programme erforderlich sind.

Programmname: R-L-C SCHALTKREIS IMPEDANZ

Dieses Programm berechnet die Impedanz einer parallelen oder seriellen Schaltung von R-L-C-Elementen bei einer Frequenz f.

AUFGABE

1. Serielle R-L-C Schaltung



$\omega = 2\pi f$

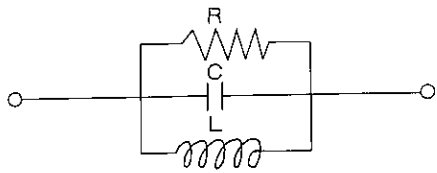
$$Z = R + j(\omega L - \frac{1}{\omega C}) = x + jy$$

Impedanz: $$|Z| = \sqrt{R^2 + (\omega L - \frac{1}{\omega C})^2}$$

Phase: $$\theta = \tan^{-1}(\frac{\omega L - \frac{1}{\omega C}}{R})$$

Dann werden $|Z|$ und θ durch eine $(x, y) \xrightarrow{\text{POL}} (r, \theta)$ Umwandlung erhalten.

2. Parallele R-L-C Schaltung



$\omega = 2\pi f$

$$Z = \frac{1}{\frac{1}{R} + j(\omega C - \frac{1}{\omega L})} = x + jy$$

Impedanz: $$|Z| = \frac{1}{\sqrt{\frac{1}{R^2} + (\omega C - \frac{1}{\omega L})^2}}$$

Phase: $$\theta = \tan^{-1}[R(\frac{1}{\omega L} - \omega C)]$$

Einsetzen: $\frac{1}{R} + j\left(\omega C - \frac{1}{\omega L}\right) = x' + jy'$

und wie folgt berechnen:

$$(x',\ y') \xrightarrow{POL} (r,\ \theta) \longrightarrow \left(\frac{1}{r},\ -\theta\right) \xrightarrow{REC} (x,\ y)$$

$$\downarrow \quad \downarrow$$
$$|Z| \quad \theta$$

ANLEITUNG

1. DEF A (Impedanz einer seriellen Schaltung)
   Die Werte für Winderstand R (Ω), Kapazität C (μF), Induktivität L (mH) und Frequenz f (Hz) nacheinander eingeben. Das Ergebnis der Berechnung für x, y, $|Z|$ und θ (Grad) wird angezeigt.
2. DEF B (Impedanz einer parallelen Schaltung)
   Die Bedienung ist die gleiche wie für DEF A .

BEISPIEL

Serielle Schaltung

R = 5 [Ω]
C = 10 [μF]
L = 25 [mH]
f = 50 [Hz]

Parallele Schaltung

R = 8 [Ω]
C = 0.5 [μF]
L = 40 [mH]
f = 60 [Hz]

TASTENBETÄTIGUNG

< Serielle Schaltung >

Schritt-Nr.	Tasteneingabe	Anzeige	Bemerkungen
1	DEF [A]	SERIAL CIRCUIT	Serielle Schaltung
		R = _	Widerstand (Ω)
2	5 ENTER	C = _	Kapazität (μF)
3	10 ENTER	L = _	Induktivität (mH)
4	25 ENTER	F = _	Frequenz (Hz)
5	50 ENTER	X	Anzeige für x
6	ENTER	5.	
7	ENTER	Y	Anzeige für y
8	ENTER	−310.4559045	
9	ENTER	IMPEDANCE Z	Anzeige für Impedanz $\|Z\|$
10	ENTER	310.4961653	
11	ENTER	THETA(DEG)	Anzeige für Phase θ
12	ENTER	−89.07731137	
13	ENTER	>	ENDE

< Parallele Schaltung >

Schritt-Nr.	Tasteneingabe	Anzeige	Bemerkungen
1	DEF B	PARALLEL CIRCUIT	Parallele Schaltung
		R = _	Widerstand (Ω)
2	8	C = _	Kapazität (μF)
3	0.5 ENTER	L = _	Induktivität (mH)
4	40 ENTER	F = _	Frequenz (Hz)
5	60 ENTER	X	Anzeige für x
6	ENTER	6.250732478	
7	ENTER	Y	Anzeige für y
8	ENTER	3.306690691	
9	ENTER	IMPEDANCE Z	Anzeige für Impedanz $\|Z\|$
10	ENTER	7.071482153	
11	ENTER	THETA (DEG)	Anzeige für Phase θ
12	ENTER	27.87922142	
13	ENTER	>	ENDE

PROGRAMMLISTING

```
 10:"A":A=0:WAIT 100:
    PRINT "SERIAL CIRCUI
    T":GOTO 30
 20:"B":A=1:WAIT 100:
    PRINT "PARALLEL CIRC
    UIT"
 30:WAIT
 35:USING
 40:DEGREE
 50:INPUT "R=";R
 60:INPUT "C=";C:C=C*1E-
    6
 70:INPUT "L=";L:L=L*1E-
    3
 80:INPUT "F=";F
 90:W=2*PI *F
100:IF A=1 GOTO 200
110:REM SERIAL
120:U=R:V=W*L-RCP (W*C)
130:Y=POL (U,V)
140:REM PRINT
150:PRINT "X":PRINT U
160:PRINT "Y":PRINT V
170:PRINT "IMPEDANCE Z":
    PRINT Y
180:PRINT "THETA(DEG)":
    PRINT Z
190:END
200:REM PARALLEL
210:Y=RCP R:Z=W*C-RCP (W
    *L)
220:Y=POL (Y,Z):U=RCP Y
230:V=-Z
240:Y=REC (U,V)
250:REM PRINT
260:PRINT "X":PRINT Y
270:PRINT "Y":PRINT Z
280:PRINT "IMPEDANCE Z":
    PRINT U
290:PRINT "THETA(DEG)":
    PRINT V
300:END
```

444 bytes

SPEICHERINHALT

A	Kennbuchstabe	
C	Kapazität	
F	Frequenz	
L	Induktivität	
R	Widerstand	
U	x	$\|Z\|$
V	y	θ
W	$2\pi f\,(\omega)$	$2\pi f\,(\omega)$
Y	$\|Z\|$	x', x
Z	θ	$y' y$

Programmname: SCHNITT ZWISCHEN KREISEN UND GERADEN

ÜBERBLICK

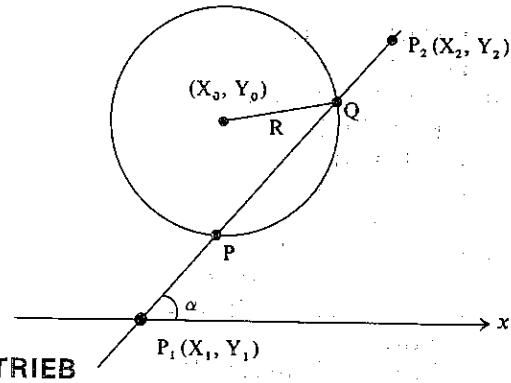
Es werden die Schnittpunkte zwischen Kreisen und Geraden in der X – Y Ebene bestimmt.

INHALT (Formeln)

Die Schnittpunkte zwischen einem Kreis und einer Geraden seien P un Q.

Hinweis: Die Winkel müssen in Grad, Minuten und Sekunden folgendermaßen eingegeben werden.

123.1423 = 123 Grad 14 Minuten 23 Sekunden

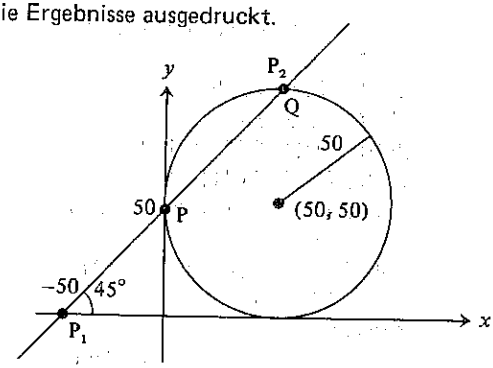


ANLEITUNG ZUM BETRIEB

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

$X_1 = -50$
$Y_1 = 0$
$X_2 = 50$ $X_P = 0$
$Y_2 = 100$ $Y_P = 50$
$X_0 = 50$ $X_Q = 50$
$Y_0 = 50$ $Y_Q = 100$
$R = 50$
$\alpha = 45°$



Hinweis: Die Koordinatenwerte sind bis auf 5 Dezimalstellen genau.

TASTENBETÄTIGUNG
(2 Punkte der Geraden sind bekannt)

Schritt-Nr.	Eingabe	Anzeige	Bemerkungen
1	DEF A	XØ = _	
2	50 ENTER	YØ = _	
3	50 ENTER	R = _	
4	50 ENTER	X1 = _	
5	−50 ENTER	Y1 = _	
6	0 ENTER	X2 = _	
7	50 ENTER	Y2 = _	
8	100 ENTER	X−P ØØ.ØØØØ	(x_P, y_P)
9	ENTER	Y−P 49.9999	
10	ENTER	X−Q 5ØØ.ØØØØ	(x_Q, y_Q)
11	ENTER	Y−Q 1ØØ.ØØØØ	
12	ENTER	>	ENDE

(1 Punkt und 1 Winkel sind bekannt)

Schritt-Nr.	Eingabe	Anzeige	Bemerkungen
1	DEF B	XØ = _	
2	50 ENTER	YØ = _	
3	50 ENTER	R = _	
4	50 ENTER	X1 = _	
5	−50 ENTER	Y1 = _	
6	0 ENTER	A = _	
7	45 ENTER	X−P Ø.ØØØØ	(x_P, y_P)
8	ENTER	Y−P 49.9999	
9	ENTER	X−Q 5Ø.ØØØØ	(x_Q, y_Q)
10	ENTER	Y−Q 1ØØ.ØØØØ	
11	ENTER	>	ENDE

PROGRAMMLISTING

```
10:"A":J=0:GOTO 30
20:"B":J=1
30:WAIT
35:USING
40:DEGREE :INPUT "X0=";
   A,"Y0=";B,"R=";C
50:INPUT "X1=";D,"Y1=";
   E
60:IF J<>0 INPUT "A=";H
   :H=DEG H:GOTO 100
70:INPUT "X2=";F,"Y2=";
   G
80:X=F-D:Y=G-E:GOSUB 50
   0
90:H=X
100:X=A-D:Y=B-E:GOSUB 50
    0
110:K=W*SIN (X-H)
120:L=ACS (K/C)
130:M=H-90-L:N=H-90+L
140:GOSUB 600
150:PRINT USING "######.
    ####";"X-P";O:PRINT
    "Y-P";P
160:M=N:GOSUB 600
170:PRINT "X-Q";O:PRINT
    "Y-Q";P
180:END
500:W=SQR (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
```

376 bytes

SPEICHERINHALT

A	X_0
B	Y_0
C	R
D	X_1
F	X_2
G	Y_2
H	$\sqrt{}$
J	$\sqrt{}$
K	h
L	α
M	Q_P
N	Q_Q
O	X_P, X_Q
P	Y_P, Y_Q
W	L
X	ΔX, θ
Y	ΔY

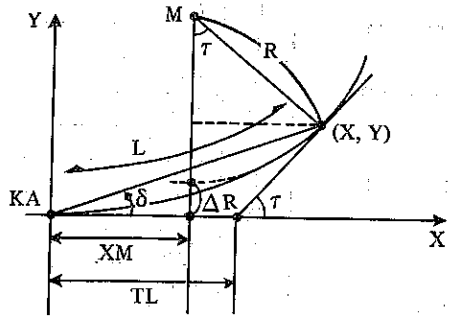
Programmname: KLOTHOIDE	Option Thermaldrucker/Kassetten-Interface CE-126P erforderlich

ÜBERBLICK

Dieses Programm dient zum Ermitteln der verschiedenen in einer Klothoide verwendeten Elemente, z.B. Übergangsbogen für den Straßenbau.

INHALT

Für die Klothoide werden A, Rbzw. A, L eingegeben, um X, Y zu ermitteln.



$$L = \frac{A^2}{R}$$

$$l = \frac{A}{R}$$

$$X \fallingdotseq Al\left(1 - \frac{l^4}{40} + \frac{l^8}{3456} - \frac{l^{12}}{599040}\right)$$

$$Y \fallingdotseq \frac{Al^3}{6}\left(1 - \frac{l^4}{56} + \frac{l^8}{7040} - \frac{l^{12}}{1612800}\right)$$

Winkel der Tagente $\tau = \frac{L}{2R} \times \frac{180}{\pi}$

$XM = X - R \sin \tau$ $\quad \delta = \tan^{-1}\frac{Y}{X}$

$TL = X - Y \cot \tau$

$TK = Y \cdot \operatorname{cosec} \tau$ $\quad SC = Y \cdot \operatorname{cosec} \delta$

Betrag der Verschiebung

$\Delta R = Y + R \cos \tau - R$

BEDIENUNGSSCHRITTE

1. DEF A : Wenn die Parameter und der Radius der Klothoide bekannt sind,
   DEF B : wenn die Parameter der Klothoide und die Länge der Kurve bekannt sind.
2. Durch Eingabe der Variablen werden die Elemente der Klothoide über den Drucker ausgegeben.

BEISPIEL

A = 100	X = 49.9219	XM = 24.9869 m
R = 200 m	Y = 2.081	TL = 33.3606 m
L = 50	t = 7°09'43"	ΔR = 0.5205 m
	SC = 49.9652	δ = 2°23'13"
		TK = 16.6915

(Hinweis) Die Entfernung wird auf die fünfte Stelle gerundet.

AUSDRUCK

```
A=100.
R=200.
L=          50.0000
X=          49.9219
Y=           2.0810
T=           7.0943
XM=         24.9869
TL=         33.3606
TK=         16.6915
DR=          0.5205
D=           2.2313
SC=         49.9652
```

```
A=100.
L=50.
R=         200.0000
X=          49.9219
Y=           2.0810
T=           7.0943
XM=         24.9869
TL=         33.3606
TK=         16.6915
DR=          0.5205
D=           2.2313
SC=         49.9652
```

TASTENBETÄTIGUNG

Schritt-Nr.	Eingabe	Anzeige	Bemerkungen
1	DEF A	A = _	Bereit zur Eingabe der Parameter
2	100 ENTER	R = _	Ausdruck der Parameter und Bereitschaft zur Eingabe des Radius am Ende der Klothoide
3	200 ENTER	>	Ausdruck des Radius und Bereitschaft zum Ausdrucken der Klothoiden-Elemente
1	DEF B	A = _	Bereit zur Eingabe der Parameter
2	100 ENTER	L = _	Ausdruck der Parameter und Bereitschaft zur Eingabe der Kurvenlänge
3	50 ENTER	>	Ausdruck der Kurvenlänge und Ausdruck der Klothoiden-Elemente

PROGRAMMLISTING

```
 10:"A":L=1:GOTO 30
 20:"B":L=0
 30:WAIT
 35:USING
 40:DEGREE :INPUT "A=";A
    :LPRINT "A=";A
 50:IF L=0 INPUT "L=";D:
    LPRINT "L=";D:B=A*A/
    D:GOTO 70
 60:INPUT "R=";B:LPRINT
    "R=";B
 70:C=A/B:D=A*A/B
 80:USING "########.####
    "
 90:IF L=0 LPRINT "R= ";
    B:GOTO 110
100:LPRINT "L= ";D
110:X=A*C*(1-C^4/40+C^8/
    3456-C^12/599040)
120:Y=A*C^3/6*(1-C^4/56+
    C^8/7040-C^12/161200
    )
130:LPRINT "X= ";X:
    LPRINT "Y= ";Y
140:C=D*90/B/PI :K=C:
    GOSUB 300
150:LPRINT "T= ";K
160:E=X-B*SIN C:LPRINT "
    XM=";E
170:G=X-Y/TAN C:LPRINT "
    TL=";G
180:F=Y/SIN C:LPRINT "TK
    =";F
190:H=Y+B*COS C-B:LPRINT
    "DR=";H
200:I=ATN (Y/X):K=I:
    GOSUB 300
210:LPRINT "D= ";K
220:J=Y/SIN I:LPRINT "SC
    =";J
230:END
300:K=DMS (K+.00014):
    RETURN
```

483 bytes

SPEICHERINHALT

A	A
B	R
C	l, τ
D	L
E	XM
F	TK
G	TL
H	ΔR
I	δ
J	SC
K	$\theta°$
L	Unterscheiden
X	X
Y	Y

Programmname: WURZEL EINER GLEICHUNG

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.

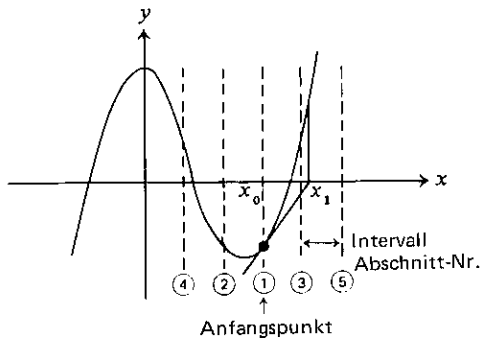
Der Differentialwert $f'(X)$ wird wie folgt definiert:

$$f'(X) = \frac{f(X+h)-f(X)}{h} \qquad (h: \text{Minutenwert})$$

Um 10^{-8} abzuwandeln, verändert man IE−8 in Zeile 240.

ANLEITUNG ZUM BETRIEB

Eingabe: Anfangswert
Minutenwert
Intervall



Ausgabe: Wurzelwert (die ENTER -Taste drücken, um die Wurzel im nächsten Intervall zu ermitteln).

Dieses Programm verwendet den Grundalgorithmus der Newton-Methode. Während mehrere Wurzeln erhalten werden können, führt die begrenzte Speicherkapazität und Anzeigekapazität dazu, daß Teile der Wurzeln eventuell nicht angezeigt werden.

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:

300 B = ((X − 2) * X − 1) * X + 2 [ENTER]

310 RETURN [ENTER]

TASTENBETÄTIGUNG

Schritt-Nr.	Eingabe	Anzeige	Bemerkungen
1	[DEF] [A]	STARTING POINT = _	ANFANGSWERT
2	0 [ENTER]	MINUTE INTV. = _	MINUTE
3	0.0001 [ENTER]	INTERVAL = _	INTERVALL
4	0.5 [ENTER]	2.	
5	[ENTER]	1.	Zum Auffinden der nächsten Wurzel nochmal die [ENTER] -Taste drücken
6	[ENTER]	−1.	
7	[ENTER]	1.	
8	[ENTER]	−1.	
9	[ENTER]	−1.	
10	[ENTER]	−1.	
11	[ENTER]	2.	

PROGRAMMLISTING

```
 10:"A":WAIT
 15:USING
 20:INPUT "STARTING POIN
    T=";V
 30:INPUT "MINUTE INTV.=
    ";A
 40:INPUT "INTERVAL=";W
 50:G=V:F=V:Z=0
 60:IF Z=0 GOTO 80
 70:G=G-W:C=G:GOTO 90
 80:C=G:Z=1
 90:GOSUB 200
100:F=F+W:C=F
110:GOSUB 200
120:GOTO 60
130:END
200:X=C:GOSUB 300
210:Y=B:X=A+C
220:GOSUB 300
230:D=C:C=D-A*Y/(B-Y)
240:IF ABS (D-C)>=1E-8
    GOTO 200
250:PRINT C
260:RETURN
300:B=((X-2)*X-1)*X+2
310:RETURN
```

285 bytes

SPEICHERINHALT

A	Minutenwert
B	f (x)
C	X_0
D	f (x + h)
F	$\sqrt{}$
G	$\sqrt{}$
V	Anfangswert
W	Intervall
X	x
Y	f (x)
Z	Anfangsmarkierung

Programmname: **TRANSFORMATION VON RECHTWINKLIGEN KOORDINATEN UND POLARKOORDINATEN**

ÜBERSICHT

In diesem Programm wird die Transformation in zwei oder in drei Dimensionen durchgeführt. Die Gradeinheit von Eingaben und Ausgaben entspricht der Voreinstellung.

ANLEITUNG ZUM BETRIEB

Das Programm enthält die vier nachstehend aufgeführten Funktionen:

- zwei Dimensionen
  - Rechtwinklig nach Polar
  - Polar nach Rechtwinklig
- drei Dimensionen
  - Rechtwinklig nach Polar
  - Polar nach Rechtwinklig

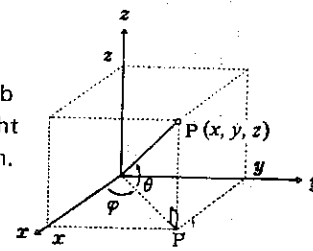
2. Drei Dimensionen

a) Rechtwinklig → Polar

$r = \sqrt{x^2 + y^2 + z^2}$

$\theta = \mathrm{Sin}^{-1}\ (z/r)$

Wenn x = y = z = 0, dann ist r = 0 deshalb können θ und φ nicht mehr definiert werden.



Wenn $x > 0$, dann $\varphi = \mathrm{Tan}^{-1}\ (y/x)$
Wenn $x = 0$ und $y \geq 0$, dann $\varphi = 90°$
Wenn $x = 0$ und $y < 0$, dann $\varphi = -90°$
Wenn $x < 0$ und $y \geq 0$, dann $\varphi = \mathrm{Tan}^{-1}\ (y/x) + 180°$
Wenn $x < 0$ und $y < 0$, dann $\varphi = \mathrm{Tan}^{-1}\ (y/x) - 180°$

b) Polar → Rechtwinklig

$x = r\ \mathrm{Cos}\ \theta \cdot \mathrm{Cos}\ \varphi$
$y = r\ \mathrm{Cos}\ \theta \cdot \mathrm{Sin}\ \varphi$
$z = r\ \mathrm{Sin}\ \theta$

DEF A ; zweidimensional Rechtwinklig in Polar
DEF B ; zweidimensional Polar in Rechtwinklig

BEISPIEL

a) Rechtwinklig → Polar

X = −1
Y = 2
Z = −3

b) Polar → Rechtwinklig

R = 3.741657387
θ = −53.30077479°
φ = 116.5650512

TASTENBETÄTIGUNG

* Zunächst gewünschte Gradeinheit wählen.

Schritt-Nr.	Eingabe	Anzeige	Bemerkungen
1	DEF B	R = _	
2	3.741657387 ENTER	THETA = _	
3	−53.30077479 ENTER	PHI = _	
4	116.5650512 ENTER		
5	ENTER		
6	ENTER		
1	DEF A	X = _	
2	−1 ENTER	Y = _	
3	2 ENTER	Z = _	
4	−3 ENTER		
5	ENTER		
6	ENTER		

PROGRAMMLISTING

```
 10:"A":CLEAR
 20:USING
 30:GOSUB 500:INPUT "Z="
    ;Z
 40:R=SQR (X*X+Y*Y+Z*Z)
 50:IF R=0 WAIT :PRINT "
    R=0 ANG. UNFIXED":
    END
 60:C=ASN (Z/R)
 70:IF X>0 LET F=ATN (Y/
    X):GOTO 120
 80:GOSUB 700
 90:IF X=0 LET F=A*ACS 0
    :GOTO 120
100:F=ATN (Y/X)+A*ACS -1
110:WAIT
120:PRINT "R":PRINT R
130:PRINT "THETA":PRINT
    C
140:PRINT "PHI":PRINT F
150:END
160:"B":CLEAR
170:WAIT
180:USING
190:GOSUB 600
200:GOSUB 620
210:INPUT "PHI=";F
220:X=X*COS F:Y=Y*SIN F:
    Z=R*SIN C
230:PRINT "X":PRINT X
240:PRINT "Y":PRINT Y
250:PRINT "Z":PRINT Z
260:END
500:INPUT "X=";X,"Y=";Y
510:RETURN
600:INPUT "R=";R,"THETA=
    ";C
610:RETURN
620:X=R*COS C:Y=R*COS C:
    RETURN
700:A=(Y=0)+SGN Y:RETURN
```

405 bytes

SPEICHERINHALT

A	√
C	θ
F	φ
R	r
X	X
Y	Y
Z	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 DEF A initiiert.

[EINGABE]

DEF A

AUSGANGSJAHR	ENTER
MONAT	ENTER
TAG	ENTER
ZIELJAHR	ENTER
MONAT	ENTER
TAG	ENTER

Das Programm wird durch Eingabe von DEF Z anstelle der Jahreszahl beendet.

Hinweis: Die Anzahl der von diesem Programm berechneten Tage schließt den Anfangstag nicht ein. Wenn der Anfangstag enthalten sein soll, verwenden Sie die folgende Zeile im Programm:
250: PRINT "DAYS=", X+1

[BEISPIEL]

von 1976 Jhar 10 Monat 5 Tag
bis 1982 Jhar 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 YEAR = _	START JAHR Eingabe des Ausgangsdatums: 5.10.1976
2	1976 ENTER	MONTH = _	MONAT
3	10 ENTER	DAY = _	TAG
4	5 ENTER	END YEAR = _	END JAHR Eingabe des Zieldatums: 4.6.1982
5	1982 ENTER	MONTH = _	MONAT
6	6 ENTER	DAY = _	TAG
7	4 ENTER	DAYS = 2068.	TAGE
8	ENTER	END YEAR = _	Eingabe des Zieldatums: 1.1.1985
9	1985 ENTER	MONTH = _	
10	1 ENTER	DAY = _	
11	1 ENTER	DAYS = 3010.	
12	ENTER	END YEAR = _	
13	DEF Z	>	Programmende

PROGRAMMLISTING

```
10:"A":CLEAR
20:WAIT
25:USING
30:INPUT "START YEAR=";
   A:GOTO 50
40:GOTO 30
50:INPUT "MONTH=";S:
   GOTO 70
60:GOTO 50
70:INPUT "DAY=";T:GOTO
   90
80:GOTO 70
90:R=A
100:INPUT "END YEAR=";A:
    GOTO 120
110:GOTO 100
120:INPUT "MONTH=";V:
    GOTO 140
130:GOTO 120
140:INPUT "DAY=";W:GOTO
    160
150:GOTO 140
160:H=R
170:G=S:I=T
180:GOSUB 300
190:J=I
200:H=A
210:G=V:I=W
220:GOSUB 300
230:X=I-J
250:PRINT "DAYS=",X
260:GOTO 100
300:IF G-3>=0 LET G=G+1:
    GOTO 320
310:G=G+13:H=H-1
320:I= INT (365.25*H)+
    INT (30.6*G)+I
330:I=I- INT (H/100)+
    INT (H/400)-306-122:
    RETURN
340:"Z":END
```

393 bytes

SPEICHERINHALT

A	Jahr (nach Berechnung)
G	√
H	√
I	√
J	√
R	Startjahr
S	Startmonat
T	Starttag
V	Zielmonat
W	Zieltag
X	Gewünschter Tag

Programmname: TABELLENPROGRAMM	CE-126P (Thermaldrucker/ Kassetteninterface) erforderlich

Dies ist ein sehr praktisches und leicht zu benutzendes Summierungsprogramm. Code-Nummern oder Postennamen und die dazugehörigen Daten können in beliebiger Reihenfolge eingegeben werden. Die summe und der Prozentanteil jedes Codes (Postens) und am Ende die Gesamtsumme werden ausgegeben.

ANLEITUNG

1) DEF Z : (Registrierung der Postennamen)
Die Titel in der Reihenfolge ihrer Codes eingeben.

2) DEF A : (Eingabe der Daten)
Durch Eingabe der Code-Nummern werden die Namen der Posten angezeigt, dann können die Daten eingegeben werden.

3) DEF S : (Ausdruck der Summen)
Die Einzelsummen der Codes (Posten) werden addiert und die Gesamtsumme und der Prozentsatz werden angezeigt. Zuerst muß eingegeben werden, ob der Prozentsatz ausgedruckt werden soll.

BEISPIEL

Eingabedaten

Code	Posten
1	PC-1245
2	PC-1251
3	PA-7050
4	EL-331
5	EL-332
6	WN-106
7	CT-660
8	EL-550
9	CS-2302
10	EA-11E
11	EA-150
12	EA-850C

Code-Tabelle

Code	Preis	Anzahl	Summe
4	5800	3	------
3	----	--	39,800
8	14,800	15	------
9	19,800	20	------
11	3,300	40	------
12	800	18	------
8	14,800	20	------
7	------	--	178,000
8	------	--	74,000
3	------	--	597,000
12	800	99	-------
2	29,800	4	-------

Anmerkung: Die angegebenen Preise sind keine tatsächlichen Preise.

AUSDRUCK

Beispiel mit Prozentsatz

1	PC-1245	0	0.00%
2	PC-1251	119200	5.51%
3	PA-7050	636800	29.41%
4	EL-331	17400	0.80%
5	EL-332	0	0.00%
6	WN-106	0	0.00%
7	CT-660	178000	8.22%
8	EL-550	592000	27.34%
9	CS-2302	396000	18.29%
10	EA-11E	0	0.00%
11	EA-150	132000	6.10%
12	EA-850C	93600	4.32%
TOTAL		2165000	

Beispiel ohne Prozentsatz

1	PC-1245	0
2	PC-1251	119200
3	PA-7050	636800
4	EL-331	17400
5	EL-332	0
6	WN-106	0
7	CT-660	178000
8	EL-550	592000
9	CS-2302	396000
10	EA-11E	0
11	EA-150	132000
12	EA-850C	93600
TOTAL		2165000

TASTENBETÄTIGUNG

< Eingabe der Postennamen >

Schritt-Nr.	Tasteneingabe	Anzeige	Bemerkungen
1	DEF Z	1.	
		ITEM NAME = _	Wartet auf die Eingabe von Code Nr. 1
2	PC-1245 ENTER	2.	
		ITEM NAME = _	Wartet auf die Eingabe von Code Nr. 2
3	PC-1251 ENTER	3.	
	⋮	⋮ Auf die gleiche Weise eingeben	
13	EA-850C ENTER	13.	
		ITEM NAME = _	
14	ENTER	>	Ende durch Drücken nur der Taste ENTER .

< Eingabe der Daten >

Schritt-Nr.	Tasteneingabe	Anzeige	Bemerkungen
1	DEF A	SUMMATION	Anzeige des Titels
		CODE = _	
2	4 ENTER	EL-331	
		DATA = _	
3	5800 * 3 ENTER	CODE = _	
4	3 ENTER	PA-7050	
		DATA = _	
5	39800 ENTER	CODE = _	
6	8 ENTER	EL-550	
		DATA = _	
⋮		: Auf die gleiche Weise eingeben	
25	29800 * 4 ENTER	CODE = _	
26	ENTER	>	Ende durch Drücken nur der Taste ENTER .

< Ausdruck der Ergebnisse >

Schritt-Nr.	Tasteneingabe	Anzeige	Bemekungen
1	DEF S	SUMMATION LIST	Anzeige des Titels
		PERCENT. (Y/N) _	
2	Y ENTER		Ausgabe der Summen mit Prozentsätze
	N ENTER		Ausgabe der Summen ohne Prozentsätze

PROGRAMMLISTING

```
 10:"Z":CLEAR :DIM C$(36
    )*10,D(36)
 20:I=0:WAIT
 25:USING
 30:I=I+1:WAIT 100:PRINT
    I:BEEP 1:INPUT "ITEM
     NAME=";C$(I):GOTO 3
    0
 40:N=I-1
 50:END
 60:"A":WAIT 100:PRINT "
    SUMMATION"
 70:INPUT "CODE=";C$(0):
    GOTO 90
 80:END
 90:I=VAL C$(0):IF I<0
    GOTO 150
100:IF I>N GOTO 150
110:PRINT C$(I)
120:D=0:INPUT "DATA=";D
130:D(I)=D(I)+D:E=E+D
140:GOTO 70
150:WAIT 100:PRINT "NO R
    EGISTRATION":GOTO 70
160:"S":WAIT 100:PRINT "
    SUMMATION LIST"
170:INPUT "PERCENT.(Y/N)
    ";C$
180:IF (C$="Y")+(C$="N")
    <>1 GOTO 170
190:FOR I=1 TO N
200:LPRINT USING "###";I
    ;"   ";C$(I)
210:IF C$="N" LPRINT "
              "; USING "#
    ##########";D(I):
    GOTO 240
220:H=D(I)/E*100:H= INT
    (H*100+.5)/100
```

SPEICHERINHALT

C$	√
D	Preis (eingegebener Wert)
E	Summe
H	Prozentsatz
I	√
N	Anzahl
C$(36)*10	Postenname
D(36)	Einzelpreis
C$(0)	Code-Nr.

```
230:LPRINT "   "; USING
    "############";D(I);"
     "; USING "###.##";H
    ;"%"
240:NEXT I
250:LPRINT "":LPRINT "TO
    TAL      "; USING "#
    ##########";E
260:END
```

571 bytes

INDEX

ABS 82
ACS 83
Anzeige 22
Arithmetische Funktionen 71
ATN 85
Ausschalten 15
Betriebsart 23
CLEAR 86
CLOAD 87
CLOAD? 88
CONT 89
COS 90
CSAVE 91
CUR 93
DATA 94
Datenspeicherung 28
DEGREE/GRAD/RADIAN 95
DEG 96
DIM 97
DMS 101
Definable Keys 152
Drucker 25
END 102
Einfache Variable 65
Einschalten 16
EXP 103
FACT 104
Fehlermeldung/Fehlersuche 79
Feldvariable 67
FOR... TO ... STEP/NEXT 105
GOSUB 108
GOTO 110
IF ... THEN 111
Indizierte Variable 65
INKEY$ 114
INPUT 115
INPUT # 118
INT 119
Kassetten-Interface 27
LEN 120
LET 121
LIST 122

LLIST . . . 123
LPRINT . . . 124
LN . . . 127
LOG . . . 128
Löschen einer Zeile . . . 78
MDF . . . 48
MEM . . . 129
MID$ . . . 130
NEW . . . 131
NEW Ø . . . 132
Numerische Feldvariable . . . 68
PASS . . . 133
PI . . . 135
POL . . . 136
PRINT . . . 137
PRINT # . . . 142
Programmaufbau . . . 75
Programmausführung . . . 78
Programmeingabe . . . 76
Programmerstellung . . . 74
RANDOM . . . 144
RCP . . . 145
READ . . . 146
REC . . . 147
Rechenbereich . . . 42
REM . . . 148
RESTORE . . . 149
RETURN . . . 150
RND . . . 151
RUN/GOTO . . . 152
SGN . . . 154
SIN . . . 155
SQR . . . 156
SQU . . . 157
Statistische Berechnungen . . . 38
STOP . . . 158
Stromversorgung . . . 10
TAN . . . 160
Tastaturabdeckung . . . 13
Tastenfunktionen . . . 17
TEN . . . 161
Textausdrücke . . . 55
Textfunktionen . . . 72
Textfeldvariable . . . 69
TRON . . . 162

TROFF 163
USING 164
Vergleichsausdrücke, logische 61
VAL 167
Variable A 66
Variablen 63
WAIT 168
Wissenschaftliche Schreibweise 52

SHARP CORPORATION

OSAKA, JAPAN

Printed in Japan/Imprimé au Japc
5B12.1T(TINSG4416CCZ: