

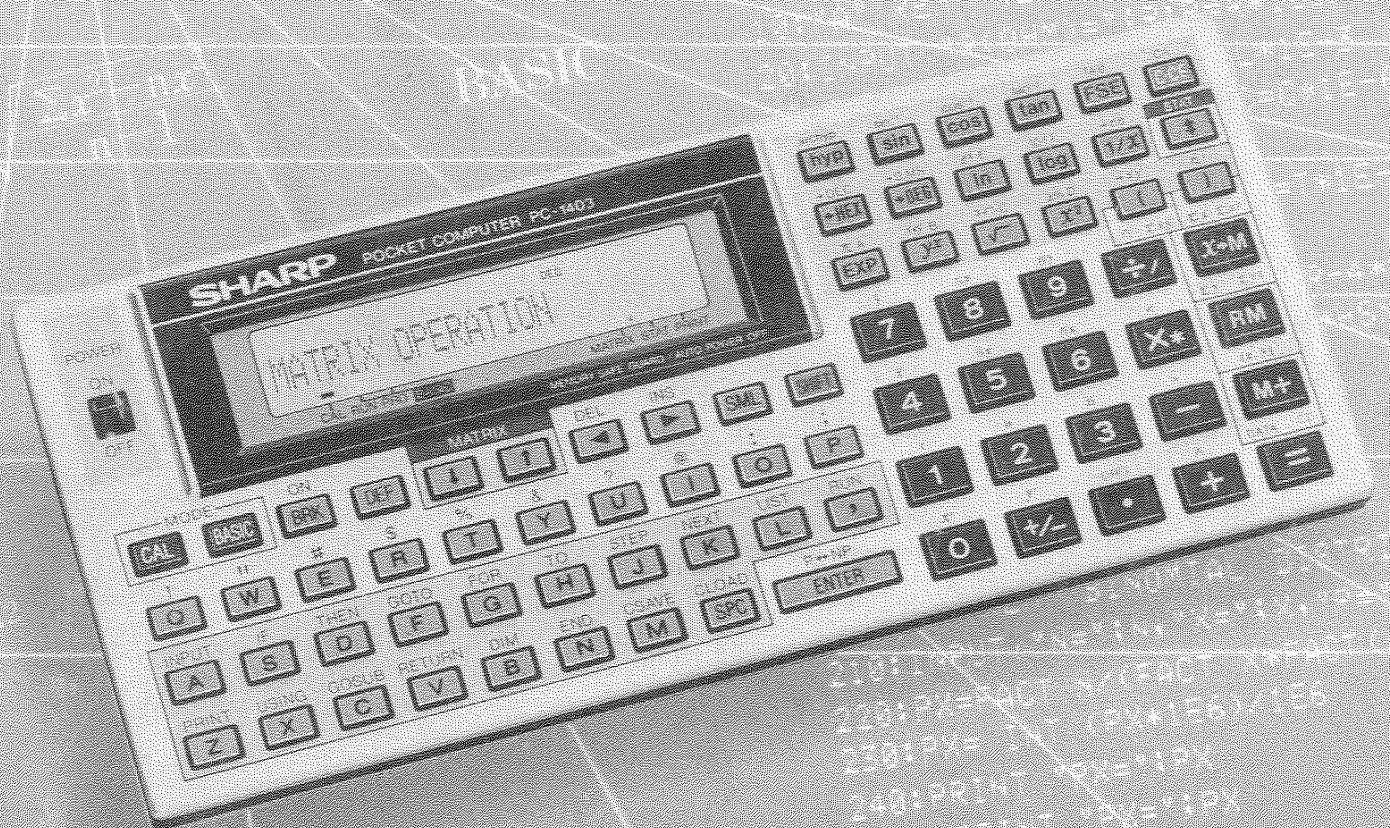
SHARP®

TASCHENCOMPUTER

MODELL

PC-1403

BEDIENUNGSANLEITUNG



$$c = \sqrt{a^2 + b^2 - 2ab \cos \theta}$$

```
210: GOTO 100  
220: PX=FACT(X)/FACT(X-RESULT)  
230: PX=2*(PX+IE6)/IE6  
240: PRINT PX=PX  
250: LPRINT PX=PX  
260: END
```

SHARP PC-1403

BEDIENUNGSANLEITUNG

Liebe 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

- 1 - MTZ
- 2 - VCC
- 3 - GND $\frac{\perp}{\equiv}$
- 4 - BUSY
- 5 - DOUT
- 6 - XIN
- 7 - XOUT
- 8 - DIN
- 9 - ACK
- 10 - IO2
- 11 - IO1

EINLEITUNG

I	Allgemeines	8
II	Betriebshinweise	9
III	Stromversorgung	11
IV	Tastaturabdeckung	15
V	Richtige Behandlung des Computers	16

TEIL I – THEORIE

1.1	GRUNDLAGEN SHARP PC-1403	18
1.1.1	Einschalten des Computers	18
1.1.2	Ausschalten des Computers	18
1.1.2.1	Automatisch.	18
1.1.2.2	Manuell	18
1.1.3	Bedienelemente.	19
1.1.4	Tastenfunktionen	20
1.1.5	Die Anzeige	22
1.1.6	Wahl der Betriebsart.	23
1.1.6.1	CAL-Mode	24
1.1.6.2	RUN- und PRO-Mode.	24
1.2	GRUNDLAGEN SHARP CE-126P	25
1.2.1	Allgemeines	25
1.2.2	Verwendung als Drucker.	26
1.2.2.1	Manueller Druckbetrieb (Protokollerstellung)	26
1.2.2.2	Programmgesteuerter Ausdruck	27
1.2.3	Verwendung des integrierten Kassetten-Interfaces	28
1.2.3.1	Auswahl des Kassettenrekorders.	28
1.2.3.2	Anschluß des Kassettenrekorders an das CE-126P	29
1.2.4	Datenspeicherung auf Magnetband	29
1.2.4.1	Allgemeines	29
1.2.4.2	Speichern von Daten und Programmen	30
1.2.4.3	Überprüfen der Abspeicherung.	31
1.2.4.4	Laden von Programmen oder Daten.	31

TEIL II – PRAXIS

2.1	SHARP PC-1403 – EINSATZ ALS WISSENSCHAFTLICHER RECHNER.	34
2.1.1	Betriebshinweise	34
2.1.1.1	Zweite Funktionsebene der Tastatur; Merkmale; Editier- und Korrekturmöglichkeiten.	34

2.1.1.2	Anzeige/Display-Format und Symbole.	41
2.1.1.3	Grundeinstellung.	42
2.1.1.4	Löschen der Eingabe oder einer Fehlermeldung.	43
2.1.1.5	Auf-/Abrundung bzw. Festlegung der Nachkommastellen	44
2.1.1.6	Wissenschaftliche Schreibweise.	45
2.1.1.7	Umwandlung des Winkels und der Zeit	46
2.1.1.8	Koordinatenumwandlung	47
2.1.1.9	Umwandlung "Hexadezimal – Dezimal" und Rechnen mit Zahlen in Hexadezimal-Schreibweise	47
2.1.2	Normale Berechnungen.	50
2.1.2.1	Addition und Subtraktion.	51
2.1.2.2	Multiplikation und Division.	51
2.1.2.3	Potenz- und Wurzelfunktion	52
2.1.2.4	Prozentrechnung.	53
2.1.2.5	Reziprok-Rechnung.	53
2.1.2.6	Speicherrechnung	53
2.1.2.7	Vorrangordnung und Verwendung der "Klammern"-Tasten.	54
2.1.3	Wissenschaftliche Berechnungen.	57
2.1.3.1	Trigonometrische und inverse trigonometrische Funktionen (Arcusfunktionen).	57
2.1.3.2	Hyperbel- und inverse Hyperbelfunktionen (Areafunktionen)	57
2.1.3.3	Logarithmische Funktionen.	58
2.1.3.4	Exponentialfunktionen.	58
2.1.3.5	Fakultät	58
2.1.4	Statistische Berechnungen.	59
2.1.5	Rechenbereich	65
2.1.6	Matrizen	69
2.1.6.1	Aufbau der Matrizen.	69
2.1.6.2	Eingabe der Matrixelemente	69
2.1.6.3	Matrizenrechnungen	73
2.1.6.4	Ausdruck der Matrizen	80
2.1.6.5	Fehlermeldung	81
2.2	SHARPPC-1403–EINSATZ ALS BASIC-RECHNER	82
2.2.1	Rechnen ohne programmunterstützung (RUN-Mode)	82
2.2.1.1	Grundrechnungsarten.	82
2.2.1.2	Rechengenauigkeit.	84
2.2.1.3	Wissenschaftliche Schreibweise.	85
2.2.1.4	Editier-/Korrekturmöglichkeit	86
2.2.1.5	Mathematische Funktionen/Klammerregeln	87
2.2.1.6	Hexadezimal-(Sedezimal-)Zahlen	88

2.2.1.7	Textausdrucke	89
2.2.1.8	Logische Vergleichsausdrücke	90
2.2.2	Sprachelemente.	91
2.2.2.1	Numerische Konstante	91
2.2.2.2	Textkonstante	92
2.2.2.3	Numerische Variable	92
2.2.2.4	Textvariable	93
2.2.2.5	Numerische Funktionen, Textfunktionen	93
2.2.2.6	Numerischer Ausdrücke	94
2.2.2.7	Textausdrücke	94
2.2.2.8	Logische Vergleichsausdrücke	95
2.2.3	Variablen.	100
2.2.3.1	Standardvariable	101
2.2.3.2	Einfache Variable	102
2.2.3.3	Indizierte Variable.	102
2.2.3.4	Besonderheiten der Variablen A.	103
2.2.3.5	Feldvariablen	105
2.2.3.6	Numerische Feldvariablen.	105
2.2.3.7	Textfeldvariablen	106
2.2.4	Arithmetische Funktionen	108
2.2.5	Textfunktionen.	109
2.2.6	Programmieren in BASIC	110
2.2.6.1	BASIC-Übersicht.	110
2.2.6.2	Programmerstellung.	111
2.2.6.3	Programmaufbau.	112
2.2.6.4	Eingabe eines Programms	112
2.2.6.5	Korrektur einer Zeile	114
2.2.6.6	Löschen und Kopieren einer Zeile.	115
2.2.6.7	Programmausführung.	115
2.2.6.8	Fehlermeldungen/Fehlersuche	116
2.3	BASIC REFERENZTEIL.	117
2.3.1	Kommandos.	119
2.3.2	Befehle	135
2.3.3	Funktionen	181
2.3.3.1	Pseudovvariable	181
2.3.3.2	Numerische Funktionen	184
2.3.3.3	String-Funktionen	191
2.4	FEHLERSUCHE.	194

ANHANG

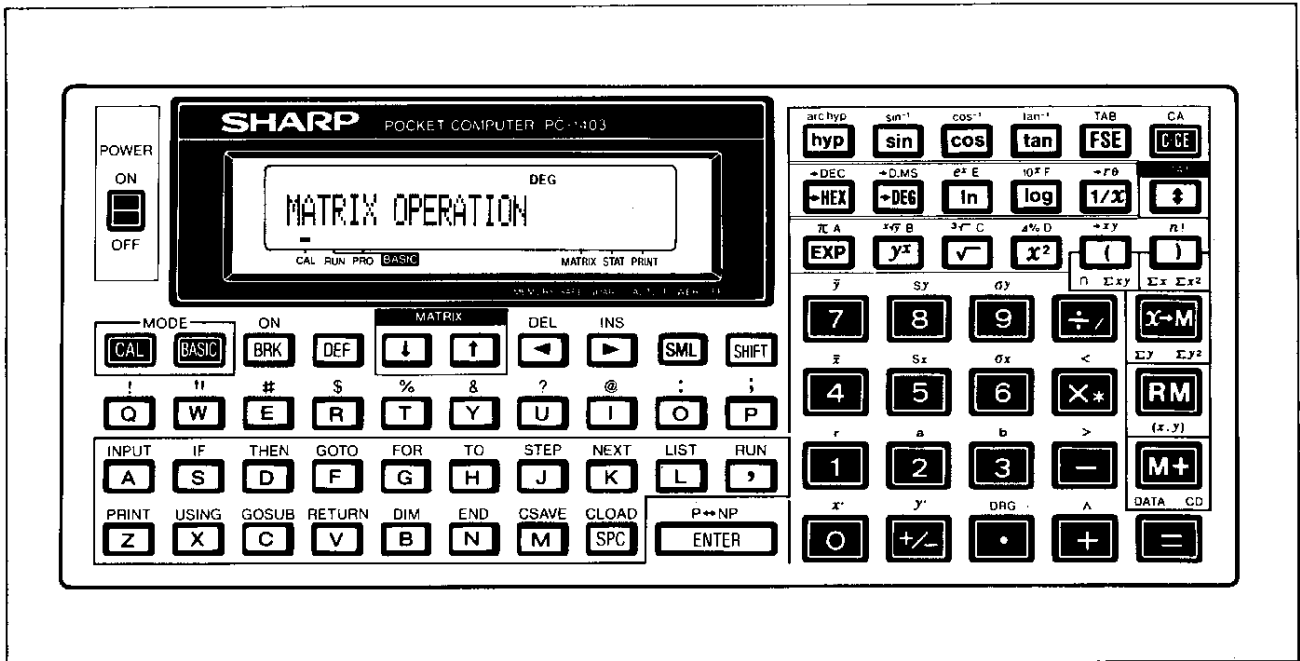
A.	Fehlermeldungen.	198
B.	ASCII-Code Tabelle	201
C.	Formatieren der Datenausgabe.	203
D.	Bewertung von Ausdrücken und Operatoren-Vorrang.	207
E.	Tastenfunktionen	209
F.	Technische Daten	214
G.	Die Benutzung von Programmen, die für andere PC-Modelle entwickelt wurden.	216
	PROGRAMMBEISPIELE	222
	INDEX	245

EINLEITUNG

EINLEITUNG

I ALLGEMEINES

Mit dem **PC-1403** stellt SHARP einen preisgünstigen und dennoch außerordentlich leistungsstarken BASIC-Taschencomputer mit integriertem wissenschaftlichem Rechner vor.



Frontansicht des **PC-1403**

Zusammen mit dem Thermodrucker mit integriertem Kassetten-Interface **CE-126P** wird der **PC-1403** zu einer kleinen, leistungstähigen 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.

Wir hoffen, Ihnen mit diesem Handbuch alle möglichen Hilfsmittel in die Hand zu geben, so daß Sie Ihren **PC-1403** für Ihre Belange optimal einsetzen können.

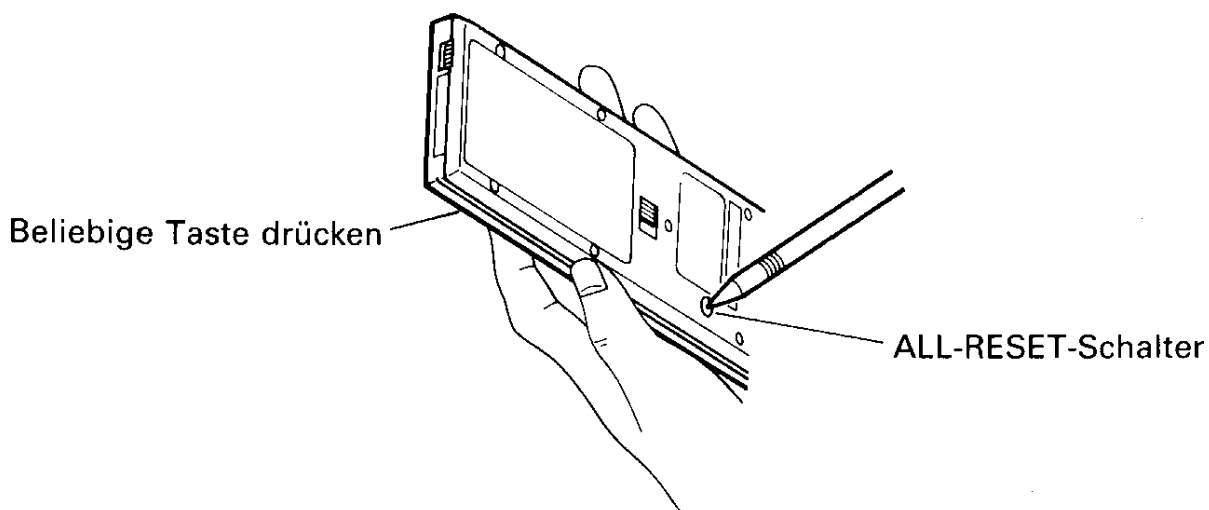
II BETRIEBSHINWEISE

1. Die Flüssigkristallanzeige des **PC-1403** 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 **C-CE** -Taste blockiert.
Sollte Ihnen dies passieren, müssen Sie den Computer **PC-1403** initialisieren. Dazu stehen zwei Möglichkeiten zur Auswahl:

FUNKTIONEN DES ALL RESET-SCHALTERS

ALL RESET: Dieser ALL-RESET-Schalter dient zum Rückstellen und Initialisieren des Computer, falls sich ein Fehler oder Problem mit der Clear- (**C-CE**) oder CA-Taste nicht lösen läßt.

Zum Rückstellen des Computers halten Sie eine beliebige Taste des Computers gedrückt und aktiviereb drücken den ALL-RESET-Schalter auf der Rückseite des Rechners mit einem spitzen Objekt.



ACHTUNG: Zum Aktivieren der Rückstellfunktion muß der ALL-RESET-Schalter (RESET) zumindest 2 - 3 Sekunden lang gedrückt werden. Kürzere Zeitspannen reichen unter Umständen nicht aus, um den Rechner zu initialisieren.

Den ALL-RESET-Schalter mit einem spitzen Objekt, z. B. einem Kugelschreiber, drücken. Vermeiden Sie Objekte, deren Spitzen abbrechen können (Bleistifte, Nadeln) oder breiter als der ALL-RESET-Schalter sind.

Falls nach dem obigen Rückstellvorgang die Tasten immer noch nicht ansprechen, drücken Sie den ALL-RESET-Schalter allein, wonach folgende Meldung im Display erscheint.

BUSY
MEMORY ALL CLEAR O.K.?

Drücken Sie nun **ENTER** , **Y** oder **=** .

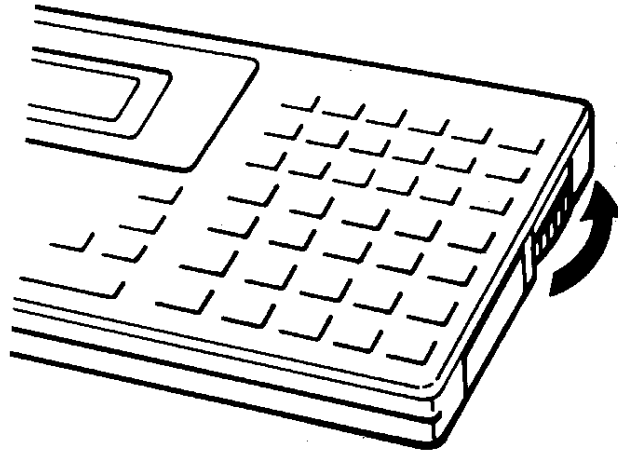
ACHTUNG: Falls innerhalb von ca. 2 Minuten keine dieser Tasten betätigt wird, schaltet sich der Computer automatisch aus.

Durch diesen Initialisierungsvorgang werden alle Inhalte des Speichers (Programm und Daten) und der RAM-Karte gelöscht. Daher sollten Sie den ALL-RESET-Schalter nur dann drücken, wenn eine grundlegende Initialisierung unumgänglich scheint.

Falls das Gerät immer noch nicht richtig funktioniert, entfernen Sie die Batterien für 10 Sekunden. Drücken Sie dann nach erneutem Einsetzen der Batterien den ALL-RESET-Schalter.

KONTRASTREGLER

Ihr Computer besitzt auf der rechten Seite einen Regler, der zum Einstellen des Display-Kontrasts dient. Stellen Sie den Regler auf optimalen Kontrast ein.



Kontrastregler: Durch Drehen des Kontrastreglers in Pfeilrichtung wird der Kontrast der Anzeige verstärkt, in entgegengesetzter Richtung verringert.

III. STROMVERSORGUNG

BATTERIEWECHSEL

Der **PC-1403** arbeitet ausschließlich mit Lithium-Batterien. Wenn er an die Option **CE-126P** angeschlossen ist, kann der Computer auch über diese versorgt werden, sofern deren momentane Betriebsspannung höher ist. Dies verringert den Stromverbrauch der Lithium-Zellen.

Beim Wechseln der Batterien beachten Sie bitte unbedingt die folgenden Hinweise:

- Wechseln Sie grundsätzlich beide Batterien gleichzeitig.
- Verwenden Sie niemals eine neue gemeinsam mit einer gebrauchten Batterie.
- Benutzen Sie nur Lithium-Zellen (Typ CR-2032, 2 Stück)

Austauschzeitpunkt

Falls die Anzeige auf dem Display trotz Einstellung auf maximalen Kontrast (Kontrastregler ganz nach links drehen) verblassen sollte, ist die Batteriespannung zu niedrig. In diesem Falls sind die Batterien umgehend zu erneuern.

ACHTUNG: Falls Sie das getrennt erhältliche CE-126 Cassetten-Interface oder CE-152 Datenrekorder- Interface einsetzen, können Sie Ihre Programme und Daten im Speicher vor dem Auswechseln der Batterien auf Cassette speichern.

Auswechseln der Batterien

- (1) Den Ein/Ausschalter in die OFF-Position schieben, um den Computer auszuschalten.
- (2) Die Schrauben an der Unterseite mit einem kleinen Kreuzschraubenzieher abnehmen (Abb. 1).

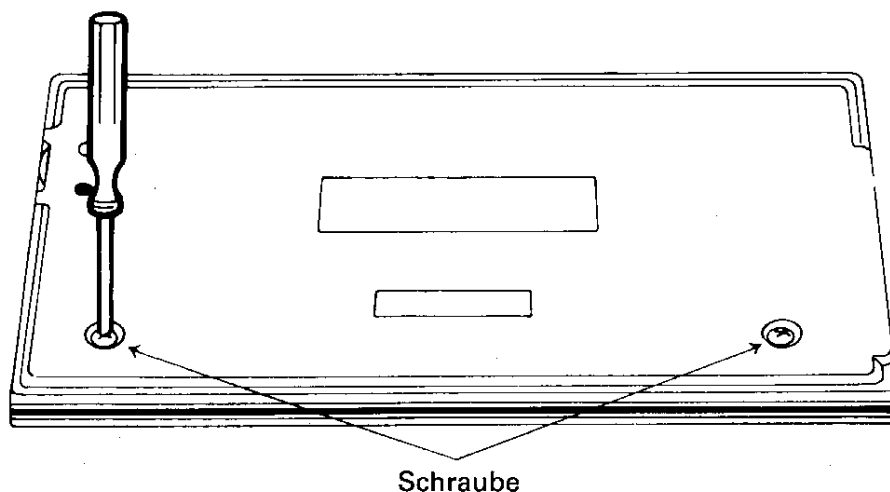


Abb. 1

- (3) Den Batteriefachdeckel entsprechend Abb. 2 in Pfeilrichtung schieben und abheben.

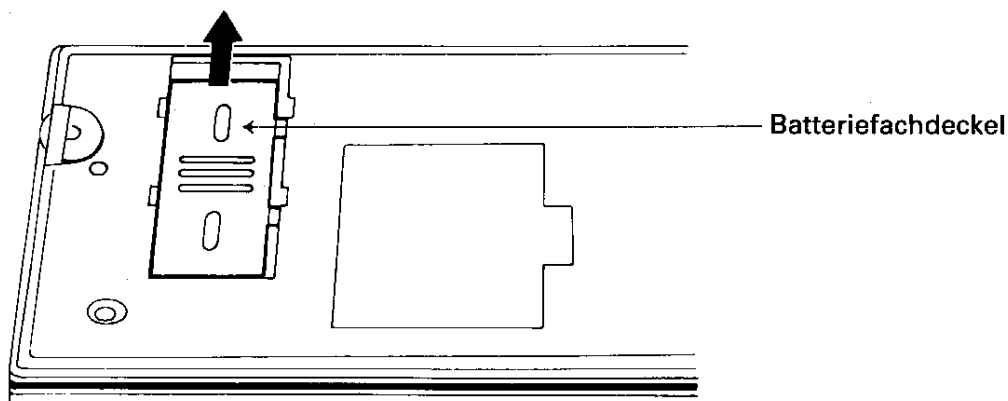


Abb. 2

(4) Die zwei Batterien auswechseln (Abb. 3)

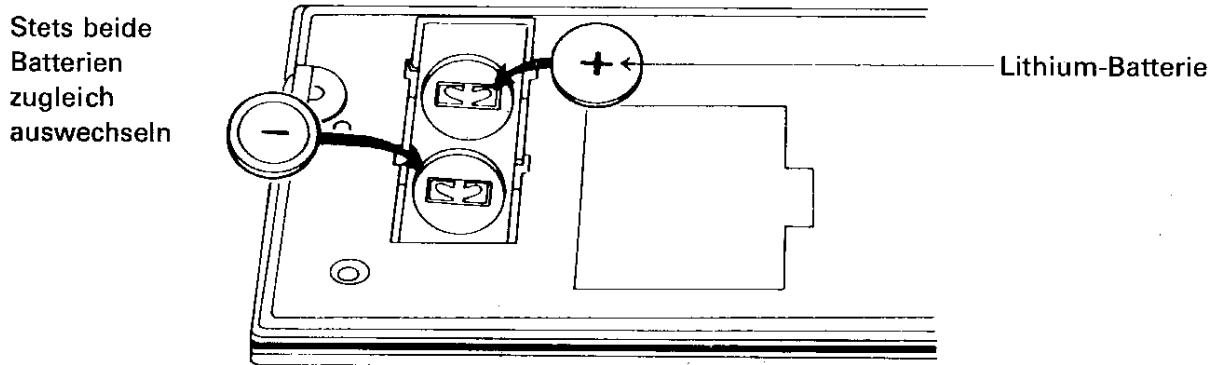


Abb. 3

- (5) Den Batteriefachdeckel gegen die in Abb. 2 gezeigte Pfeilrichtung aufschieben.
(6) Die Klauen der Bodenplatte in die Schlitze im Computer-Gehäuse einsetzen (Abb. 4).

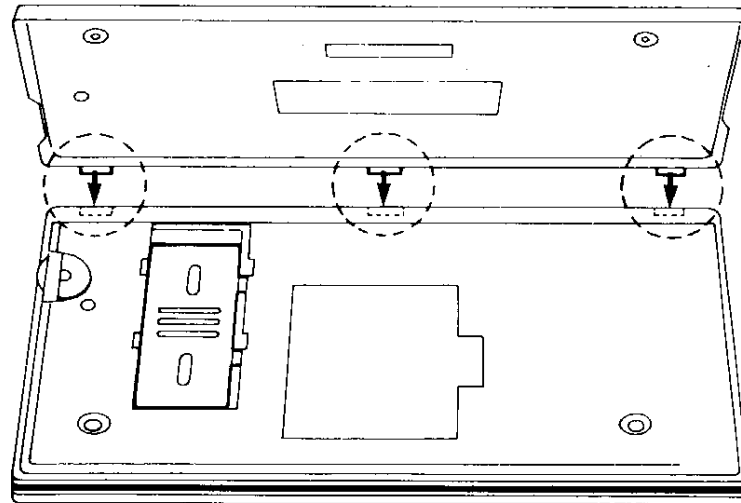
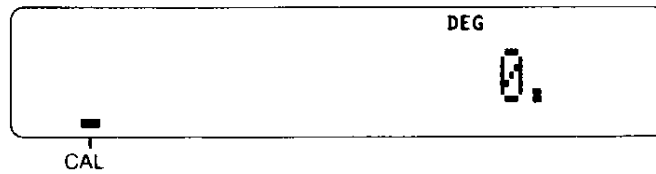


Abb. 4

- (7) Zum Anbringen der Schrauben die Bodenplatte etwas zurückschieber.

- (8) Den Ein/Ausschalter in die ON-Stellung schieben, um den Computer einzuschalten und dann den ALL-RESET-Schalter (RESET) betätigen. Danach ENTER drücken. Damit sollte folgendes Display auftauchen:



Falls das Display leer bleibt oder ein anderes Symbol als 0 anzeigt, sind die Batterien zu entfernen und erneut zu installieren. Dann das Display noch einmal überprüfen.

Achtung: Läßt man eine leere Batterie im Computer, kann er durch Säureaustritt aus der Batterie geschädigt werden. Leere Batterien umgehend erneuern!

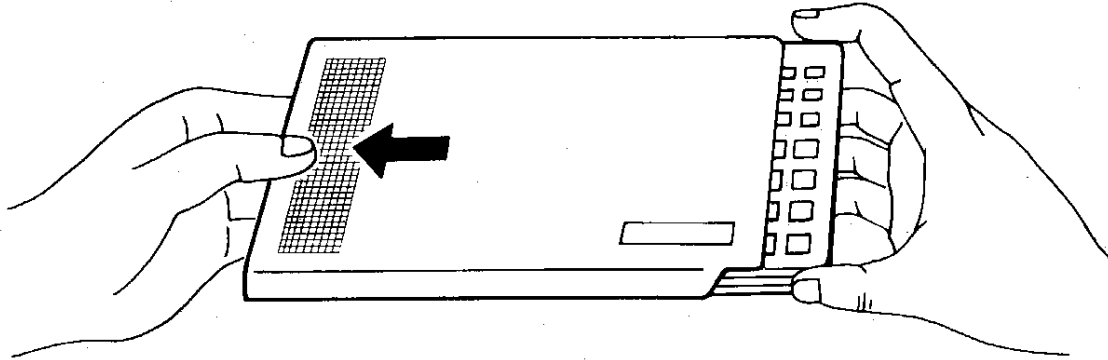
Vorsicht: Bewahren Sie die Batterien außerhalb der Reichweite von Kindern auf!

IV. TASTATURABDECKUNG

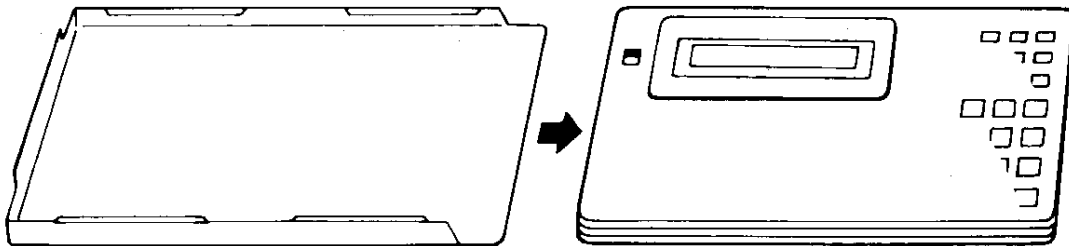
Zum Schutz der Tastatur und der Anzeige ist der **PC-1403** mit einer stabilen, doppelseitig aufschiebbarer Tastaturabdeckung ausgestattet.

Die Tastaturabdeckung des Computers wie dargestellt verwenden.

Bei Gebrauch:

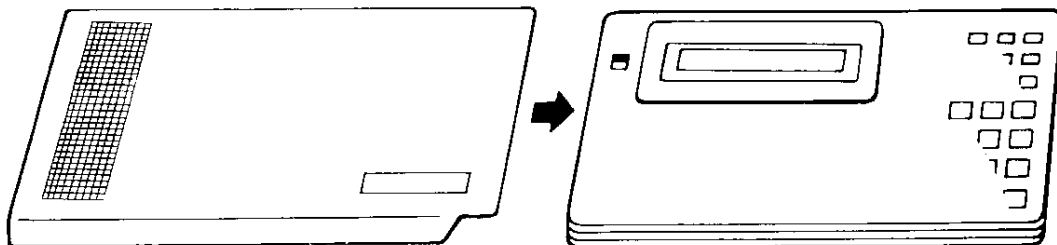


Die Tastaturabdeckung ist so gestaltet, daß sie während der Verwendung des Rechners auf dessen Unterseite geschoben werden kann.



Tastaturabdeckung bei Gebrauch des Rechners

Nach Gebrauch:



Tastaturabdeckung nach Gebrauch des Rechners

V. RICHTIGE BEHANDLUNG DES COMPUTERS

Um reibungsloses Funktionieren Ihres Computers zu gewährleisten, empfehlen wir die Beachtung der folgenden Punkte:

- Gehen Sie immer vorsichtig mit Ihrem Taschencomputer um, da die Flüssigkristallanzeige aus Glas gefertigt ist.
- Halten Sie den Computer von starken Temperaturschwankungen fern, ebenso von Feuchtigkeit oder Staub. Wenn Sie bei warmen Wetter Ihren PKW längere Zeit der direkten Sonne aussetzen und sich hohe Temperaturen aufbauen, kann das Ihren Computer beschädigen!
- Benutzen Sie zur Reinigung Ihres Computers ausschließlich ein trockenes, weiches Tuch. Verwenden Sie niemals Lösungsmittel, Wasser oder ein feuchtes Tuch!
- Um das Auslaufen der Batterien zu vermeiden, entfernen Sie diese, wenn Sie den Computer für längere Zeit nicht benutzen wollen.
- Wenn Sie die Hilfe einer Werkstatt benötigen, geben Sie Ihren Computer nur in ein autorisiertes SHARP Service Center oder wenden Sie sich an einen SHARP Fachhändler.
- Wenn der Computer starker statischer Aufladung oder auch starken Störungen ausgesetzt ist, kann er sich unter Umständen "aufhängen" (d. h. auf Tastendruck nicht mehr reagieren). Falls dies auftritt, drücken Sie den ALL-RESET-Schalter und halten dabei eine Taste fest (siehe Pannenhilfe).
- Bewahren Sie dieses Handbuch gut auf, falls Sie später einmal etwas nachschlagen wollen.

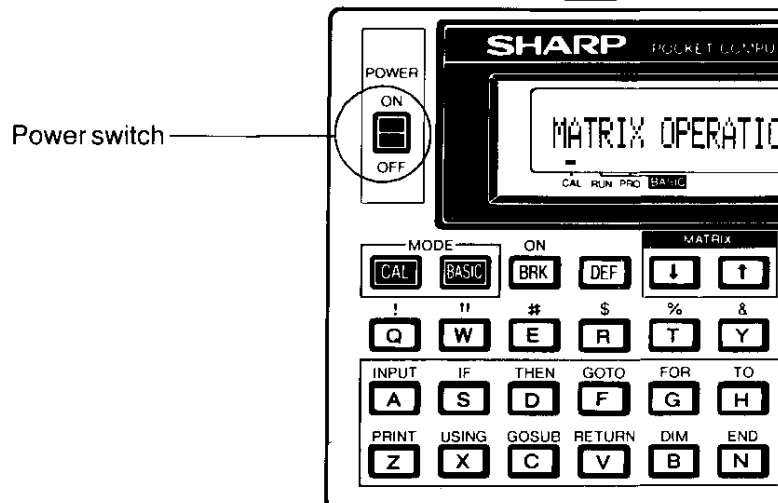
TEIL I
THEORIE

TEIL I THEORIE

1.1 GRUNDLAGEN SHARP PC-1403

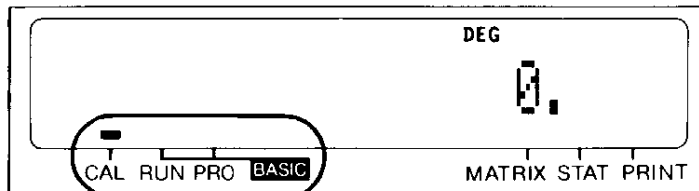
1.1.1 EINSCHALTEN DES COMPUTERS

Das Einschalten des Computers 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 Winkleinheit (DEG/GRAD/RAD) an.



Zeigt Betriebsart (CAL/RUN/PRO)

1.1.2 AUSSCHALTEN DES COMPUTERS

1.1.2.1 Automatisch

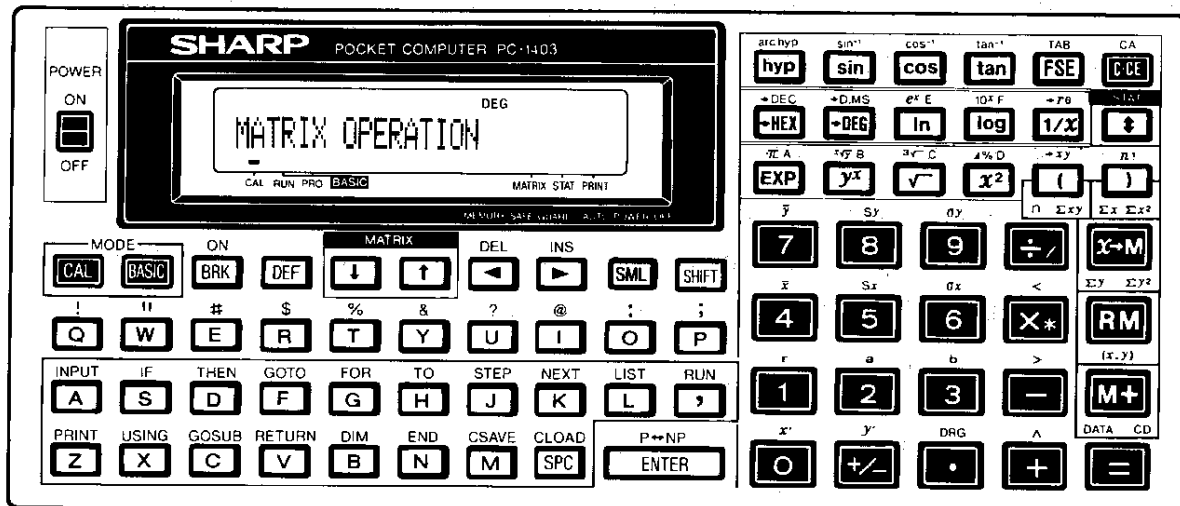
Der Computer schaltet sich nach ca. 11 Minuten nach der letzten Tastenbedienung automatisch ab. Dabei bleiben alle Programm- und Dateninformationen gespeichert.

Hat sich der Rechner automatisch ausgeschaltet, muß er über die Taste **ON BRK** wieder aktiviert werden. Nach dem Einschalten befindet sich der Rechner in der Betriebsart "CAL".

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 **PC-1403** besteht aus 77 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 bzw. 3 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 Doppelfunktions-taste **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, Hyperbelberechnungen, Polarkoordinatenberechnungen usw. durchführen. Dieser Teil der Tastatur ist mit bis zu drei Ebenen belegt.

Die **SHIFT** -, **BASIC** - und **hyp** -Tasten sind 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
- ON BRK** Unterbrechen des Programmablaufs. Am Display erscheint die Meldung **BREAK IN XXX** (XXX = Zeilennummer). Unterbricht auch Drucker- und Cassettenbetrieb.
- C-CE** 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 C-CE** 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
- 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**.
- !** ~ **®** Sonderzeichen
- SHIFT** **A** ~ **SHIFT** **SPC** Aufruf der wichtigsten **BASIC**-Anweisungen

SHIFT	P\leftrightarrowNP ENTER	Umschaltung für Druck/Nichtdruck beim Rechnen ohne Programmunterstützung im RUN/PRO-Modus
.		Dezimalpunkt
0	~ 9	numerische Zeichen
+	~ ÷ /	Grundrechnungsarten
hyp		Umschaltung Hyperbelfunktionen
SHIFT	archyp	Umschaltung Umkehrfunktionen der Hyperbelfunktionen
sin		Trigonometrische Funktionen
SHIFT	sin⁻¹	Umkehrfunktionen der trigonometrischen Funktionen
FSE		Umschaltung Fließkomma/Festkomma
SHIFT	TAB	Festlegung der Nachkommastellen
→HEX		Umrechnung Dezimalsystem – Hexadezimalsystem
SHIFT	→DEC	Umrechnung Hexadezimalsystem – Dezimalsystem
→DEG		Umrechnung Dezimalsystem – Sexagesimalsystem
SHIFT	→DMS	Umrechnung Sexagesimalsystem – Dezimalsystem
In		Natürlicher Logarithmus
SHIFT	e^x	Exponentialfunktion e ^x
log		Zehnerlogarithmus
SHIFT	10^x	Exponentialfunktion 10 ^x
1/x		Reziprokwert
SHIFT	→xy	Umwandlung Polarkoordinaten in rechtwinklige Koordinaten
SHIFT	→rθ	Umwandlung rechtwinklige Koordinaten in Polarkoordinaten
EXP		Exponentialdarstellung
SHIFT	π	Konstante PI
y^x		Potenzierfunktion
SHIFT	x√y	x-te Wurzel
√		Quadratwurzelfunktion
SHIFT	³√	Dritte Wurzel
x²		Quadratzahlfunktion

SHIFT n!	Fakultät
()	Klammern
X↔M ~ M+	Speicherrechnung
SHIFT DRG •	Umschaltung der Winkeleinheit
SHIFT > SHIFT <	Logische Vergleichsoperatoren
SHIFT STAT ↕	Statistikbetriebsart

1.1.5 DIE ANZEIGE

Der **PC-1403** verfügt über eine 24-stellige alphanumerische Flüssigkristallanzeige. Die einzelnen Zeichen werden in einer 5 x 7-Punktmatrix dargestellt.

Das Eingaberegister faßt 80 Zeichen. Alle Zeichen, die über die 24 möglichen darstellbaren Zeichen der Anzeige hinausgehen, müssen mit den Cursor-Tasten (**▶** und **◀**) 'gescrolled' werden.

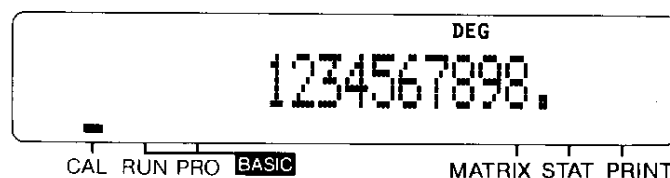
Der Cursor zeigt die Stelle an, an der die nächste Eingabe erfolgen kann. Er steht normalerweise auf der ersten freien Stelle und wird durch einen kurzen waagrechten Strich (**_**) dargestellt.

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.

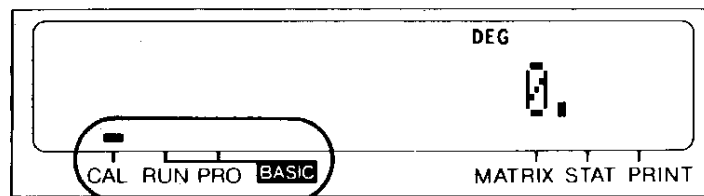


1.1.6 WAHL DER BETRIEBSART

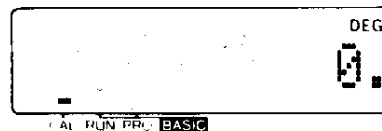
Beim **PC-1403** werden drei Betriebsarten unterschieden:

1. CAL-Mode: Der Computer wird als wissenschaftlicher Rechner verwendet.
2. RUN-Mode: Gespeicherte Programme können gestartet werden. Zusätzlich kann der Computer im RUN-Mode auch als gewöhnlicher Taschenrechner ohne Programmunterstützung verwendet werden.
3. PRO-Mode: BASIC-Programme können eingegeben, gelistet und verändert werden.

Die Wahl der Betriebsart erfolgt mit den grünen Tasten **CAL** und **BASIC**. Die gewählte Betriebsart wird am Display durch einen kleinen waagrechten Strich (—) angezeigt.



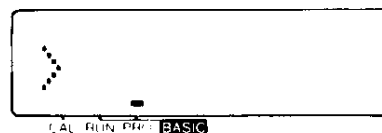
Wird der Computer eingeschaltet, so ist automatisch die Betriebsart "CAL" (Calculation) gewählt. Im Display wird das wie dargestellt angezeigt:



Wenn Sie nun die grüne Taste **BASIC** drücken, so wird die Betriebsart in den RUN-Mode umgeschaltet.



Durch nochmaliges Drücken der Taste **BASIC** wird in den PRO-Mode umgeschaltet.



Die **BASIC**-Taste ist ein Wechselschalter, d.h., durch einmaliges Drücken wird die Betriebsart umgeschaltet, durch nochmaliges Drücken der Urzustand wieder hergestellt.

Durch Drücken der **CAL**-Taste kann der Rechner wieder in den CAL-Mode umgeschaltet werden.

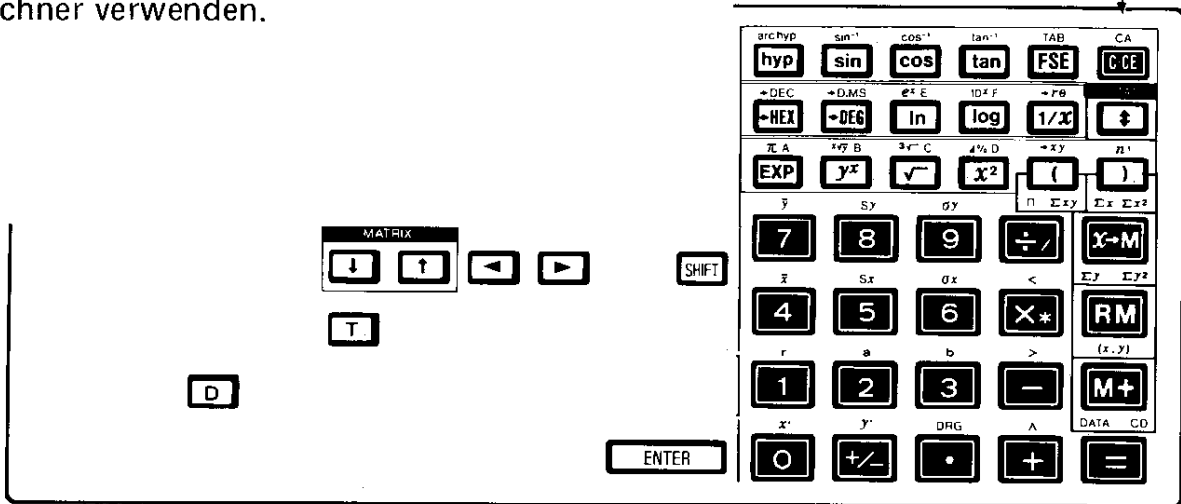
1.1.6.1 CAL-Mode

Schalten Sie Ihren Rechner durch Drücken der **CAL**-Taste in den CAL-Mode.

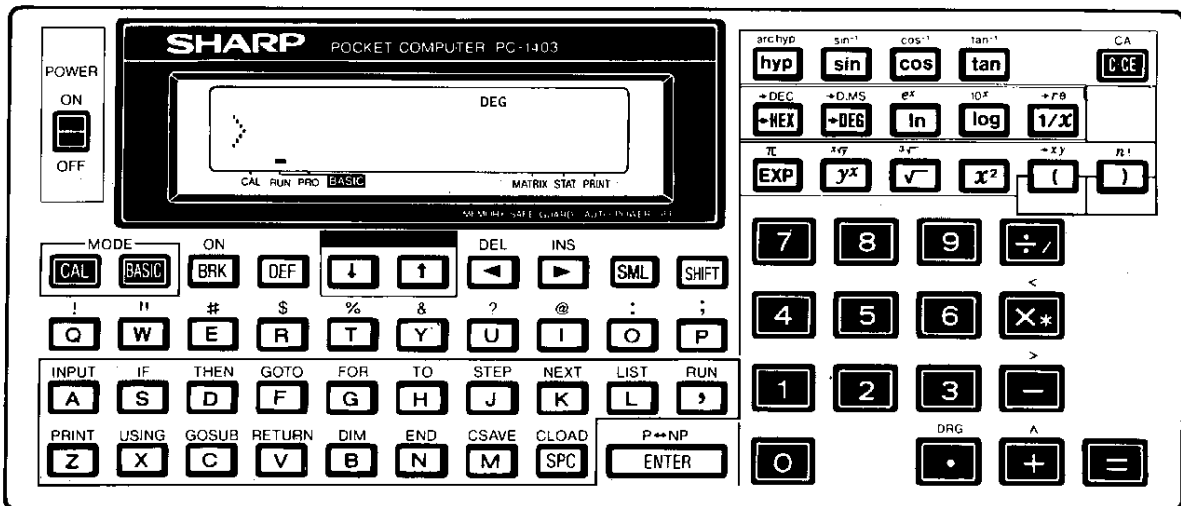
In diesem Mode können Sie mit den rechts abgebildeten Tasten den **PC-1403** als wissenschaftlichen Rechner verwenden.

Beispiel:

- C-CE** → 0.
- 1 2** → 12.
- +** → 12.
- 3** → 3.
- =** → 15. Red key



1.1.6.2 RUN- und PRO-Mode



Schalten Sie Ihren Rechner durch Drücken der Taste **BASIC** in den RUN- bzw. in den PRO-Mode. In dieser Betriebsart können die oben abgebildeten Tasten zur Eingabe verwendet werden.

Beispiele:

PRINT Z	USING X	GOSUB C	→ ZXC _		
r 1	a 2	DRG .	b 3	→ ZXC12.3 _	
	CA C-CE		→ >		
INPUT A	=	i 4	^ +	Sx 5	→ A = 4 + 5 _
				↑ Cursor	
CA C-CE	SHIFT	PRINT Z	→ PRINT _		
SHIFT	W	(√	→ PRINT "(√ _	

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 **PC-1403** 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 nur im RUN-Modus)

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*PI*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 126.

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 **C-CE** -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 **C-CE** .
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 SHARP-Taschencomputern erstellten Programme oder Daten entwickelt.

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

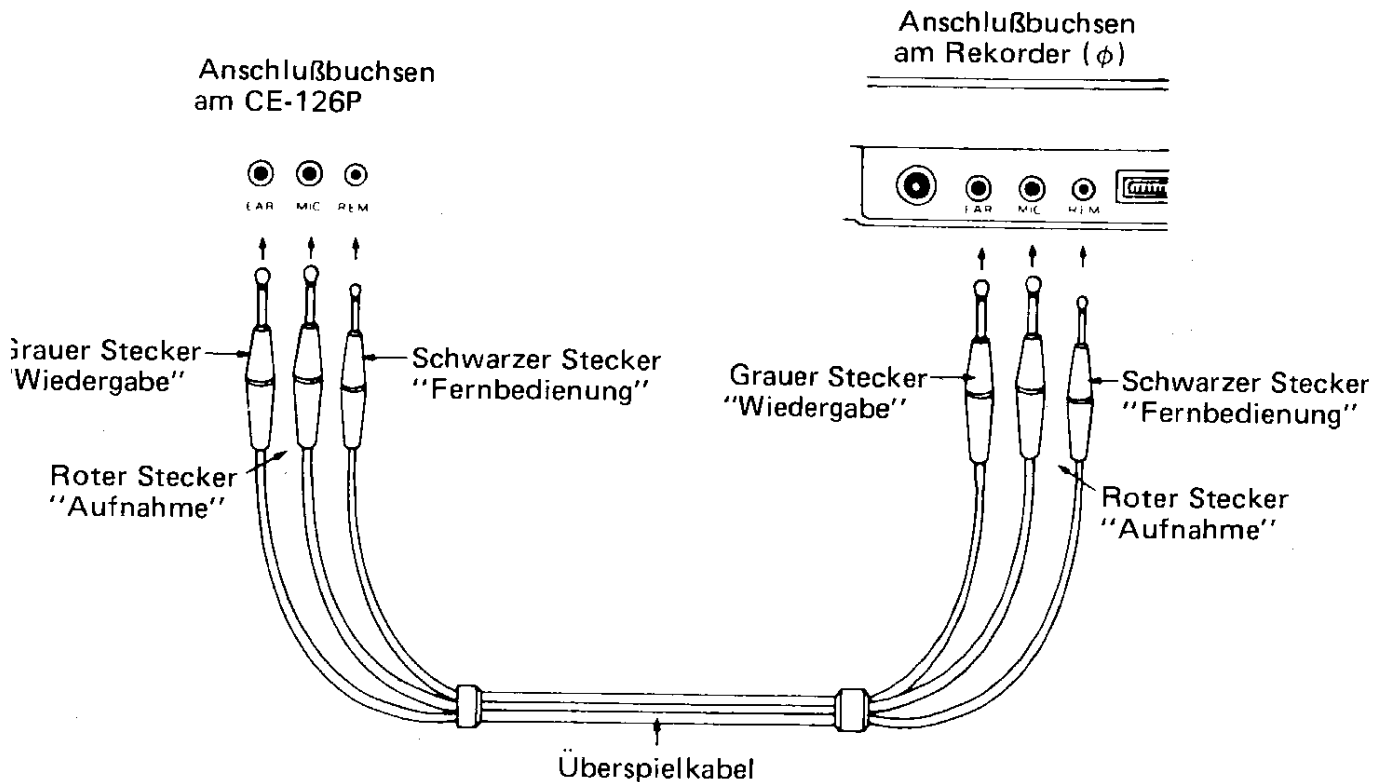
- Fernbedienung (REMOTE), Klinkenbuche 2,5 mm ϕ
- Bandzählwerk
- Mikrofoneingang (MIC), Klinkenbuchse 3,5 mm ϕ
- Ausgang (EAR), Klinkenbuchse 3,5 mm ϕ

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 _{SS} |
| – 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 vom Band zurückladen. Außerdem können Sie das Magnetband als externen Programmspeicher verwenden, um große Programme ablaufen zu lassen.

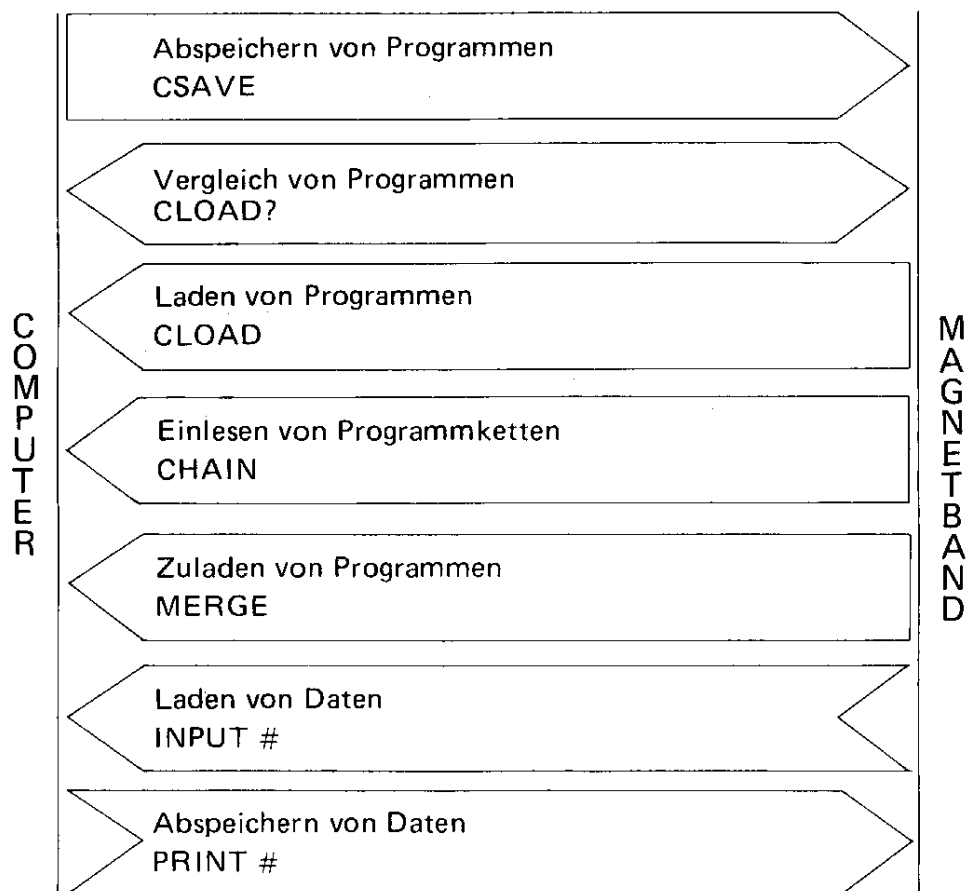
Die Informationen aus Daten und Programmen werden in Blöcken nach dem Zweitonverfahren auf das Magnetband geschrieben.

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 am 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 Überprüfen 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 den 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". Sollte der Kassettenrekorder bei Einstellung des Lautstärkereglers auf Maximum nicht ordnungsgemäß arbeiten, verringern Sie die Lautstärke und probieren erneut.
- 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. Sie hören die gleiche Signalfolge wie bei der Abspeicherung. 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.

**TEIL II
PRAXIS**

TEIL II PRAXIS

2.1 SHARP PC-1403 – EINSATZ ALS WISSENSCHAFTLICHER RECHNER

2.1.1 BETRIEBSHINWEISE

Die folgenden Absätze enthalten einige wichtige Hinweise, die sich auf die CAL-Betriebsart des Rechners beziehen. Die in TEIL I, Abschnitt 1.1 enthaltenen allgemeinen Betriebshinweise haben darüber hinaus Gültigkeit.

Das Umschalten des Rechners in die CAL-Betriebsart (**CAL** -Taste drücken) sowie das für diese Betriebsart relevante Tastenfeld wurden bereits in TEIL I, Kapitel 1.3 aufgezeigt, so daß an dieser Stelle auf eine nähere Erläuterung dieser Punkte verzichtet werden kann.

2.1.1.1 Zweite Funktionsebene der Tastatur; Merkmale; Editier- und Korrekturmöglichkeiten

Wie aus der nachfolgenden Abbildung ersichtlich, weisen nahezu alle Tasten des Tastenfelds der CAL-Betriebsart eine zweite und, wenn man die Tastenfunktionen für die statistischen Berechnungen (STAT) und die Umwandlung "Hexadezimal-Dezimal" hinzuzählt, eine dritte und vierte Funktionsebene auf.

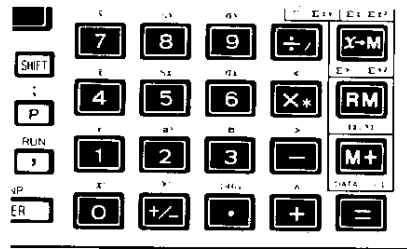
In den Erläuterungen und Rechenbeispielen ist die Auswahl der zweiten Tastenfunktion aus der Verwendung der **SHIFT** -Taste ersichtlich. Die Tastenfunktionen für die statistischen Funktionen und die Umwandlung "Hexadezimal-Dezimal" werden automatisch, d.h. durch das Drücken der **SHIFT** **STAT** -Taste bzw. **+HEX** -Taste vorgewählt.

Bei näherer Betrachtung des Tastenfelds der CAL-Betriebsart wird bereits ein wichtiges Merkmal des **SHARP PC-1403** deutlich. Es steht eine große Anzahl von Tastenfunktionen zur Verfügung, deren Verwendung zu einer wesentlichen Arbeits erleichterung und Arbeitsvereinfachung beiträgt.

Die Tastenfunktionen sind nachfolgend kurz aufgeführt und die entsprechenden Tasten angegeben. In den folgenden Abschnitten sind die einzelnen Funktionen zudem anhand von Berechnungsbeispielen erläutert. Die in Klammern stehende Seitenzahl gibt an, auf welcher Seite das Berechnungsbeispiel zu finden ist.

Grundrechenfunktionen (Seite 51):

Addition, Subtraktion, Multiplikation Division (Vorzeichenumkehr)



Speicherrechnung (Seite 53):

Übertragen in den Speicher



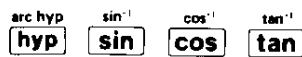
Anzeigen des Speicherinhalts



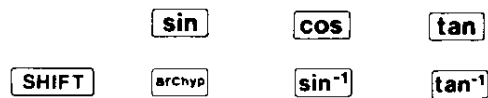
Addition und Subtraktion zum/vom Speicher



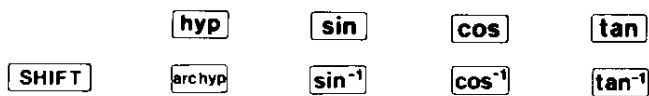
Trigonometrische und Hyperbel-Funktionen (Seite 57):



Trigonometrische und inverse trigonometrische Funktionen (Arcusfunktionen)



Hyperbelfunktion und inverse Hyperbelfunktion (Areafunktion)



Löschen der letzten Eingabe bzw. einer Fehlermeldung (Seite 43):



Löschen der vorangegangenen Eingabe bzw. des Displayinhalts durch Drücken der roten **C-CE** -Taste. Bei zweimaligem Drücken wird alles außer Speicher "M" gelöscht. Löschen von Speicher "M" durch Drücken von **C-CE** und **x→M** . Die Funktion "CA" ist in Betriebsart "CAL" unwirksam.

Umwandlung Dezimal-Hexadezimal und Hexadezimal-Dezimal (Seite 47):



Dezimal-Hexadezimal



Hexadezimal-Dezimal



Umwandlung von Winkel/Zeit (Seite 46):



Grad ° /Minuten ' /Sekunden " in Dezimalzahl (Sexagesimal-Dezimal)
(Std./Min./Sek. in Dezimalzahl)



Dezimalzahl in Grad ° /Minuten ' /Sekunden " (Dezimal-Sexagesimal)
(Dezimalzahl in Std./Min./Sek.)



Natürlicher Logarithmus und Exponentialfunktion der Zahl "e", Hexadezimal-eingabe:



Natürlicher Logarithmus (Seite 58)



Exponentialfunktion der Zahl "e" (Seite 58)



Hexadezimal "E"



Allgemeiner Logarithmus und Exponentialfunktion 10^x , Hexadezimal-eingabe:



Allgemeiner Logarithmus (Seite 58)



Exponentialfunktion 10^x (Seite 58)

SHIFT **10^x**

Hexadezimal "F"

←HEX **F**

Reziprok-Rechnen und Umwandlung von rechtwinkligen Koordinaten in Polarkoordinaten:

$\rightarrow r\theta$
 $1/x$

Reziprok-Rechnen (Seite 53)

$1/x$

Umwandlung von rechtwinkligen Koordinaten in Polarkoordinaten (Seite 47)

SHIFT **$\rightarrow r\theta$**

Austausch der im Rechner gespeicherten Zahl (Ersteingabe, Ergebnis) mit der angezeigten Zahl und Anwahl der Betriebsart "Statistische Berechnungen":

↕

Austausch

↕

Anwahl der Betriebsart "Statistische Berechnungen" (Seite 59)

SHIFT **STAT**

Exponenteingabe (wissenschaftliche Schreibweise) und Eingabe der Kreiszahl (Pi), Hexadezimal eingabe:

π A
EXP

Exponenteingabe (Seite 45)

EXP

Kreiszahl (Pi)

SHIFT **π**

Hexadezimal "A"

←HEX **A**

Potenzieren und Berechnung der x-ten Wurzel von y, Hexadezimal eingabe (Seite 52):

$x\sqrt[y]{B}$
 y^x

Potenzieren

y^x

Wurzelziehen

SHIFT **$x\sqrt{y}$**

Hexadezimal "B"

←HEX **B**

Berechnung von Quadrat- und Kubikwurzel, Hexadezimaleingabe (Seite 52):

		$\sqrt[3]{C}$
Quadratwurzel		$\sqrt{\quad}$
Kubikwurzel	SHIFT	$\sqrt[3]{\quad}$
Hexadezimal "C"	+HEX	C

Quadrierung und Prozentberechnung, Hexadezimaleingabe:

		$\Delta\% \text{ D}$
Quadrierung der angezeigten Zahl		x^2
Prozentberechnung (Prozentsatz, prozentuale Zunahme)	SHIFT	$\Delta\%$
Hexadezimal "D"	+HEX	D

Öffnen der Klammer und Umwandlung von Polarkoordinaten in rechtwinklige Koordinaten, statistische Berechnungen:

		$\rightarrow xy$
Öffnen der Klammer		(
Umwandlung von Polarkoordinaten in rechtwinklige Koordinaten (Seite 47)	SHIFT	$\rightarrow xy$

Statistische Berechnungen (Seite 59)

Datenanzahl	"STAT"	n
Datensumme von xy	"STAT" SHIFT	Σxy

Schließen der Klammer und Fakultätswert, statistische Berechnungen:

		$n!$
Schließen der Klammer)
Fakultätswert (Seite 58)	SHIFT	n!

Statistische Berechnungen (Seite 59)

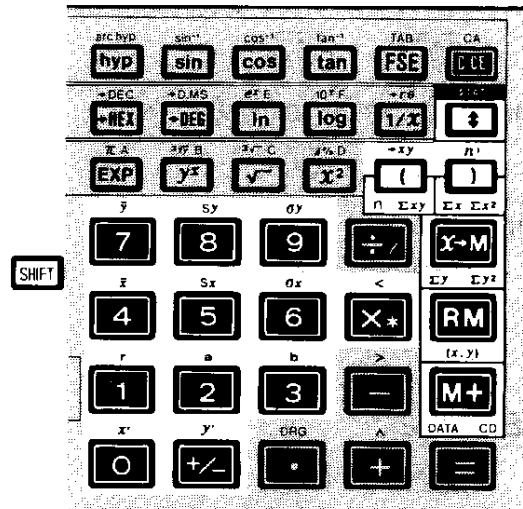
Datensumme von x

“STAT” Σx

Summe von x^2

“STAT” **SHIFT** Σx^2

Statistische Berechnungen (Seite 59):



Anwahl der Betriebsart "Statistische Berechnungen"

SHIFT **STAT**

Operandenanzahl

n

Datensumme von xy

SHIFT Σxy

Datensumme von x

Σx

Summe von x^2

SHIFT Σx^2

Mittelwert der Daten y

SHIFT \bar{y}

Standardabweichung der y-Werte mit Gesamtheitsparameter "n-1"

SHIFT s_y

Standardabweichung der y-Werte mit Gesamtheitsparameter "n"

SHIFT σ_y

Datensumme von y

Σy

Summe von y^2

SHIFT Σy^2

Mittelwert der Daten x

SHIFT \bar{x}

Standartabweichung der x-Werte mit Gesamtheitsparameter "n-1"

SHIFT Sx

Standartabweichung der x-Werte mit Gesamtheitsparameter "n"

SHIFT σ_x

Eingabe von x- und y-Daten

(x,y)

Korrelationskoeffizient

SHIFT r

Schnittpunkt der Regressionslinie mit der y-Achse

SHIFT a

Steilheit der Regressionslinie

SHIFT b

Dateneingabe

DATA

Korrektur einer falschen Eingabe

SHIFT CD

Näherungswert von x bei vorgegebenem y-Wert

Eingabe y-Wert SHIFT x'

Näherungswert von y bei vorgegebenem x-Wert

Eingabe x-Wert SHIFT y'

Anwahl der Winkeleinheit (Seite 42):

DEG (Grad; °, ', ")

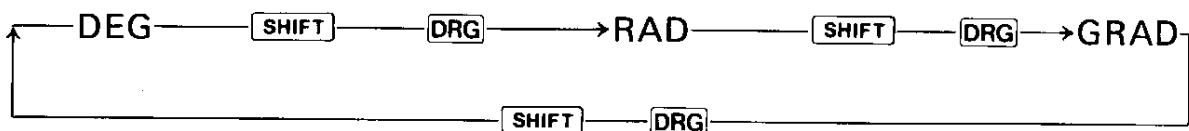
RAD (Radiant; rad)

SHIFT DRG

GRAD (Neugrad; gon)

SHIFT DRG

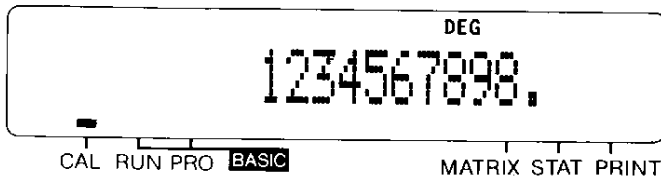
SHIFT DRG



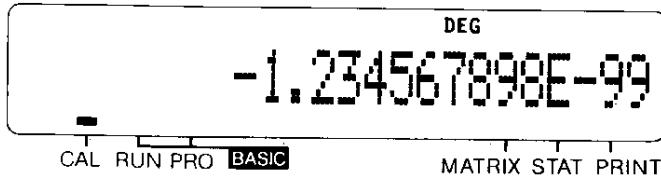
Anmerkung:

Die Funktionen (CA), < , > und ^ werden in der Betriebsart "CAL" nicht verwendet.

2.1.1.2 Anzeige/Display-Format und Symbole.



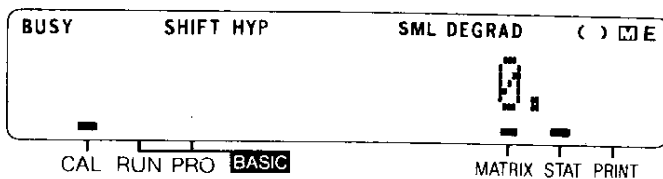
Normales Displayformat
Normal format)



Exponentielles Displayformat

Mantisse (12 Stellen) Exponent (4 Stellen)

Die Anzeige des **PC-1403** umfaßt 16 Stellen. In der Betriebsart "CAL" werden die Rechenergebnisse soweit möglich in Fließkommenschreibweise 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 4stellig (einschließlich Symbol und Vorzeichen) angezeigt.



Die Betriebsart "CAL" verwendet die oben abgebildeten Symbole und Markierungen, die nachfolgend aufgelistet sind und deren Bedeutung erläutert wird.

- CAL** Die Markierung (—) über der Beschriftung "CAL" gibt an, daß sich der Rechner in der Betriebsart "CAL" befindet.
- SHIFT** Das Wort "SHIFT" wird angezeigt, wenn die **SHIFT** -Taste gedrückt wurde und damit die zweite Funktionsebene der Tasten angewählt ist. Zum Ausschalten der Dauergroßschreibung SHIFT wird die **SHIFT** -Taste erneut gedrückt.
- HYP** Das Wort "HYP" wird angezeigt, wenn die **hyp** -Taste gedrückt wurde und damit eine Hyperbelfunktion angewählt ist. Werden die Tasten **SHIFT** und **hyp** gedrückt (Anzeige "SHIFT HYP"), so ist damit eine inverse Hyperbelfunktion (Areafunktion) angewählt.
- SML** Leuchtet durch Drücken der Taste **SML** . Dann werden Buchstaben als Kleinbuchstaben eingegeben.

DEG Diese drei Wörter können durch wiederholtes Betätigen der Tasten
RAD **SHIFT** und **DRG** angewählt werden. Die Wörter geben an, in welcher
GRAD Winkleinheit gerechnet wird.

DEG : (Grad; °, ', ")

RAD : (Radiant; rad)

GRAD : (Neugrad; gon)

Gestreckter Winkel = $180^\circ = \pi \text{ rad} = 200 \text{ gon}$

() Das Klammersymbol erscheint, sobald beim Eintippen einer Berechnungsformel die **()**-Taste verwendet wird.

M Dieses Symbol zeigt an, daß sich ein numerischer Wert, der ungleich Null ist, im Speicher befindet und der Speicher somit belegt ist.

E Das Symbol "E" in der rechten oberen Ecke des Anzeigefelds zeigt an, daß eine Bereichsüberschreitung aufgetreten ist oder eine falsche Anweisung gegeben wurde. Der Fehlerzustand kann durch das Betätigen der **C-CE**-Taste beseitigt werden.

STAT Die Markierung (–) über der Beschriftung "STAT" im rechten unteren Bereich des Anzeigefelds gibt an, daß sich der Rechner in der Betriebsart "Statistische Berechnungen" befindet. Die Betriebsart wird durch Drücken der Tasten **SHIFT** und **STAT** angewählt und durch nochmaliges Drücken wieder ausgeschaltet.

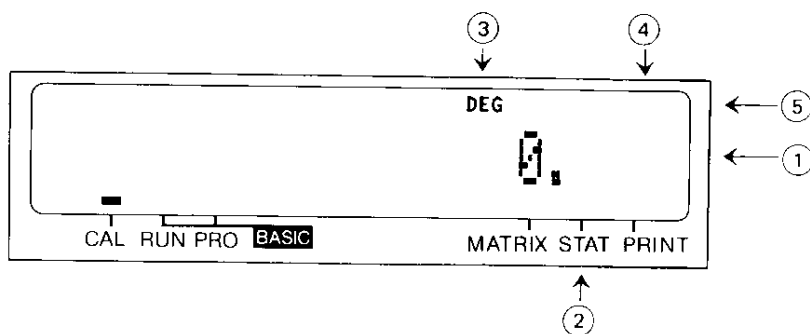
BUSY Diese Anzeige erscheint, während der Computer eine arithmetische Berechnung durchführt.

MATRIX Wenn im CAL-Modus **SHIFT** **↓** oder **SHIFT** **↑** gedrückt wird, erscheint über der MATRIX-Markierung unter rechts auf dem Display eine Klammer (–). Die MATRIX-Anzeige verweist darauf, daß der Computer für Matrix-Rechnungen bereit ist.

Zur Freigabe des Matrix-Modus, eine der obigen Tasten noch einmal drücken.

2.1.1.3 Grundeinstellung

Bevor mit den Berechnungen begonnen wird, sollte die Grundeinstellung des Rechners überprüft werden. Dazu den POWER-Schalter auf ON schalten oder, falls der Rechner automatisch abgeschaltet hat, die **ON BRK**-Taste drücken. Anschließend die **CAL**-Taste (grün) drücken, um so die Betriebsart "CAL" anzuwählen und zweimal die **C-CE**-Taste betätigen. Das Anzeigefeld sollte nun, entsprechend der nachfolgenden Abbildung, nur die Anfangsinformationen enthalten.



Falls dies nicht der Fall ist, die folgenden Kurzanweisungen lesen und die erforderlichen Maßnahmen durchführen.

- ① Es wird mehr als eine Null angezeigt (z.B. 0.00)
Zun Löschen der Tabulator-Einstellung (Stellen hinter dem Komma) wird der Computer aus- und dann wieder eingeschaltet. Danach ist das Display auf normalen Anzeige-Modus geschaltet.
- ② Über der Beschriftung "STAT" wird eine Markierung (—) angezeigt:
Die Betriebsart "Statistische Berechnungen" ist angewählt.
Durch Drücken der Tasten **SHIFT** **STAT** die Betriebsart aufheben.
- ③ Statt DEG wird RAD oder GRAD angezeigt:
Die Winkeleinheit ist in Radiant (rad) oder Neugrad (gon) vorgewählt.
Durch Drücken der Tasten **SHIFT** **DRG** Winkeleinheit auf DEG einstellen.
- ④ Das Symbol \square wird angezeigt:
Der Speicher ist mit einem numerischen Wert belegt. Löschen des Speichers durch Drücken der Tasten **C-CE** **X→M**.
- ⑤ Alle im oberen Bereich des Anzeigefelds dargestellten Symbole, mit Ausnahme der in Punkt ③ und ④ beschriebenen, können durch Drücken der **C-CE** -Taste gelöscht werden.

2.1.1.4 Löschen der Eingabe oder einer Fehlermeldung

Durch einmaliges Drücken der roten **C-CE** -Taste wird die letzte Eingabe bzw. der Displayinhalt gelöscht. Ein zweimaliges Drücken der **C-CE** -Taste löscht auch den zuerst eingegebenen Wert bzw. das Ergebnis der vorangegangenen Berechnung. Der Speicherinhalt \square wird dadurch nicht gelöscht.

123 **+** 456 4 5 6 .

C-CE 0 .

789 **=** 9 1 2 .

(123 + 789 = 912)

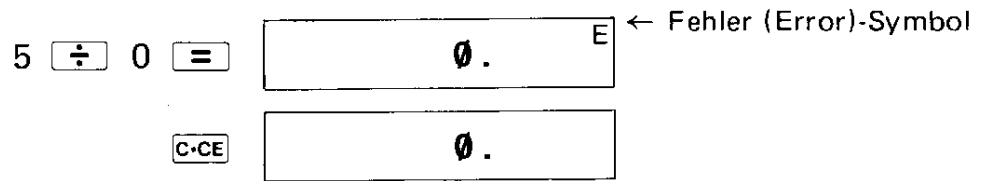
6 **x** 2 **+** 1 2 .

C-CE 0 .

6 **÷** 2 **+** 3 .

5 **=** 8 .

Die **C-CE** -Taste dient darüber hinaus zum Löschen der Fehlermeldung (E).



2.1.1.5 Auf-/Abrundung bzw. Festlegung der Nachkommastellen

Die Auf- oder Abrundung des Rechenergebnisses erfolgt in der Regel in der 11. Stelle. Da das Display 10 Stellen aufweist, ist die letzte Stelle des angezeigten Rechenergebnisses nicht auf- oder abgerundet. Bei Benützung der Tasten **=**, **M+** oder **Δ%** wird der angezeigte, nicht gerundete Wert im Falle weiterer Berechnungen verwendet.

Beispiel:

$$\begin{array}{l}
 1 \div 3 \times 3 = 1. \\
 1 \div 3 = 0.33333333 \\
 \times 3 = 0.99999999
 \end{array}$$

Die Anzahl der Nachkommastellen und damit die Rundung kann frei festgelegt werden. Dazu dient die **TAB** -Taste in Verbindung mit den Tasten (0) bis (9).

- SHIFT TAB 0** → Keine Nachkommastellen. Entsprechend der 1. Nachkommastelle wird auf- oder abgerundet.
- SHIFT TAB 1** → Eine Nachkommastelle. Entsprechend der 2. Nachkommastelle wird auf- oder abgerundet.
- SHIFT TAB 9** → Neun Nachkommastellen. Entsprechend der 10. Nachkommastelle wird auf- oder abgerundet.

Beispiel:

- SHIFT TAB 9** → **0.05555556** (FIX-Betriebsart)
Neun Nachkommastellen. Die 9. Nachkommastelle wurde aufgerundet.
- FSE** → **5.55555556E-02** (SCI-Betriebsart)
Neun Nachkommastellen. Die 9. Nachkommastelle der Mantisse wurde aufgerundet.

SHIFT **TAB** **3**

→ 5.556E-02 (SCI-Betriebsart)
Drei Nachkommastellen. Die 3. Nachkommastelle der Mantisse wurde aufgerundet.

FSE

→ 55.556E-03 (ENG-Betriebsart)
Drei Nachkommastellen. Die 3. Nachkommastelle wurde aufgerundet.

FSE

→ 0.055555555 (Normale Betriebsart)
Der Wert ist im Rechner in der Form von $5.5555555555 \times 10^{-2}$ gespeichert.

Die Rundung in der 9. Nachkommastelle der Mantisse ergibt $5.555555556 \times 10^{-2}$.

Da das Display 10 Stellen anzeigt, ist die Aufrundung in der Ergebnisanzeige nicht enthalten.

FSE

→ 0.056 (FIX-Betriebsart)

2.1.1.6 Wissenschaftliche Schreibweise

Die Displayanzeige erfolgt, solange die Anzeigekapazität nicht unter- oder überschritten wird, in Festkommenschreibweise. Durch Drücken der **FSE**-Taste kann jedoch auf einfache Weise zwischen "Festkomma-" und "Wissenschaftlicher Schreibweise" gewählt werden.

Soll oder muß ein Wert in wissenschaftlicher Schreibweise eingegeben werden, ist zur Eingabe des Exponenten die **EXP**-Taste zu verwenden. Das Vorzeichen des Exponenten läßt sich mit der **+/-**-Taste bestimmen.

Beispiel:

23 **X** 1000 **=**

23000.

FSE

23000.

FSE

2.E 04

C-CE 4 π **A** **EXP** 3

4.E 03

(4 x 10³)

=

4000.

1.23 **+/-**

-1.23

EXP 5 +/-	-1.23E-05 (-1.23 x 10 ⁻⁵)
=	-0.0000123

2.1.1.7 Umwandlung des Winkels und der Zeit

Zur Umwandlung von Winkel- oder Zeitangaben (° , ' , " bzw. Std., Min., Sek.) in den entsprechenden Dezimalwert muß der Wert als ganze Zahl (° ; Std.) plus Nachkommastellen (, ' , " ; Min., Sek.; jeweils zweistellig) eingegeben werden.

Aufgabe: Umwandlung von 12° 47' 52" in den entsprechenden Dezimalwert
 Tastenfolge: 12.4752 [+DEG]
 Ergebnis: 12.79777778

Bei der Umwandlung des dezimalen Winkel-/Zeitwerts in den sexagesimalen Wert (° , ' , " bzw. Std., Min., Sek.) ist das angezeigte Ergebnis wie folgt zuzuordnen:

Ganze Zahl	Nachkommastellen		
	1 und 2	3 und 4	5 bis 9
° (Std.)	' (Min.)	" (Sek.)	Zehntelsekunden

Aufgabe: Umwandlung des dezimalen Winkelwerts 24,7256 in den entsprechenden Sexagesimalwert
 Tastenfolge: 24.7256 [SHIFT] [+DMS]
 Ergebnis: 24.433216 24°43'32"

Beispiel:

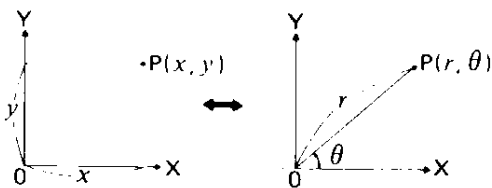
Ein Pferd läuft Rundenzeiten von 2 Min. 25 Sek., 2 Min. 38 Sek. und 2 Min. 22 Sek. Wie hoch ist die durchschnittliche Rundenzeit?

Tastenfolge: .0225 [+DEG] [+] .0238 [+DEG] [+] .0222 [+DEG] [=]
 Zwischenergebnis: 0.123611111
 Tastenfolge: [÷] 3 [=]
 Zwischenergebnis: 0.041203703
 Tastenfolge: [SHIFT] [+DMS]
 Ergebnis: 0.022833333

Die durchschnittliche Rundenzeit ist
 2 Minuten 28 Sekunden

2.1.1.8 Koordinatenumwandlung

Umwandlung von rechtwinkligen Koordinaten in Polarkoordinaten ($x, y \rightarrow r, \theta$)



$$r = \sqrt{x^2 + y^2}$$

$$\theta = \tan^{-1} \frac{y}{x}$$

DEG: $0 \leq |\theta| \leq 180^\circ$

RAD: $0 \leq |\theta| \leq \pi$

GRAD: $0 \leq |\theta| \leq 200^g$

Aufgabe: Umwandlung der rechtwinkligen Koordinaten $x = 6$ und $y = 4$ in Polarkoordinaten. Winkeleinheit DEG ($^\circ$).

Tastenfolge: 6 \downarrow 4 **SHIFT** $\rightarrow r\theta$

Ergebnis: 7.211102551 (r)

Tastenfolge: \downarrow

Ergebnis: 33.69006753 (θ)

Aufgabe: Umwandlung des Real- und Imaginärteils eines Vektors in Betrag und Richtung (Phase). Winkeleinheit DEG ($^\circ$).

Realteil $x = 12$

Imaginärteil $y = j9$

Tastenfolge: 12 \downarrow 9 **SHIFT** $\rightarrow r\theta$

Ergebnis: Betrag $r = 15$

Tastenfolge: \downarrow

Ergebnis: Richtung (Phase) $\theta = 36.86989765$

Umwandlung von Polarkoordinaten in rechtwinklige Koordinaten (Formel)

Aufgabe: Umwandlung der Polarkoordinaten $r = 14$ und $\theta = \pi/3$ in rechtwinklige Koordinaten. Winkeleinheit RAD (rad).

Tastenfolge: **SHIFT** **DRG**

SHIFT **π** \div 3 **=** \downarrow 14 \downarrow **SHIFT** $\rightarrow xy$

Ergebnis: 7.000000002 (x)

Tastenfolge: \downarrow

Ergebnis: 12.12435565 (y)

Anmerkung: Es wird zuerst $\pi/3$ errechnet. Nach der Eingabe von $r = 14$ werden die beiden Werte durch Drücken der \downarrow -Taste ausgetauscht und die richtige Reihenfolge so wieder hergestellt.

2.1.1.9 Umwandlung "Hexadezimal – Dezimal" und Rechnen mit Zahlen in Hexadezimal-Schreibweise

Die einfache Umwandlung von Zahlen zur Basis 10 (Dezimal) in Zahlen zur Basis 16 (Hexadezimal) und umgekehrt ist auf dem Feld der Datenverarbeitung und

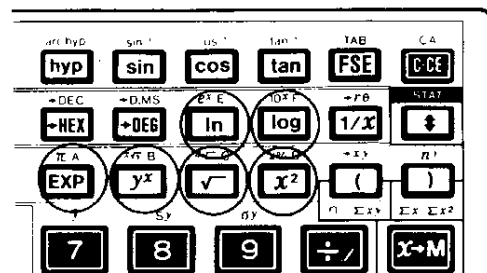
Computertechnik von großer Bedeutung, da in diesen Bereichen die Hexadezimal-Schreibweise häufig anzutreffen ist.

Die Hexadezimal-Schreibweise verwendet die Zahlen 0 bis 9 sowie die Buchstaben A bis F, wobei diese in Dezimal-Schreibweise den Werten 0 bis 15 entsprechen.

Dezimal	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hexadezimal	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Zur Eingabe einer Zahl in Hexadezimal-Schreibweise muß zuerst die **→HEX** -Taste gedrückt werden. Die Anzeige "HEX" weist darauf hin, daß es sich bei dem angezeigten Wert um eine Sedezimalzahl (Zahl in Hexadezimal-Schreibweise) handelt.

Die Buchstaben A bis F können nach Drücken der **→HEX** -Taste über die in der nebenstehenden Abbildung hervorgehobenen Tasten eingegeben werden.



Das Umschalten auf Dezimal-Schreibweise erfolgt durch Drücken der Tasten **SHIFT** **→DEC**.

Umwandlung "Dezimal-Hexadezimal"

Aufgabe: Umwandlung von Dezimal 30 in Hexadezimal

Tastenfolge: 30 **→HEX**

Anzeige: 1E. HEX

Anmerkung: Zur Eingabe einer weiteren Dezimalzahl durch Drücken der Tasten **SHIFT** **→DEC** wieder auf Dezimal-Schreibweise umschalten.

Aufgabe: Umwandlung von Dezimal -2 in Hexadezimal

Tastenfolge: **2** **+/-** **→HEX**

Anzeige: FFFFFFFE. HEX

Anmerkung: Die Umwandlung einer negativen Dezimalzahl erfolgt intern über das "Zweierkomplement". Das Resultat wird im "16er-Komplement" angezeigt.

Mit der **+/-** -Taste kann das Vorzeichen der angezeigten Zahl gewechselt werden. Bei Hexadezimal-Schreibweise erfolgt die Anzeige des jeweiligen 16er-Komplements.

Aufgabe: Umwandlung von Dezimal 123,4 in Hexadezimal

Tastenfolge: **SHIFT** **→DEC** 123.4 **→HEX**

Anzeige: 7B. HEX

Anmerkung: Es können nur ganze Zahlen in Hexadezimal umgewandelt werden. Die Nachkommastellen bleiben bei der Umwandlung unberücksichtigt.

Umwandlung "Hexadezimal-Dezimal"

Aufgabe: Umwandlung von Hexadezimal 2BC in Dezimal

Tastenfolge: **C-CE** **HEX** 2 B C **SHIFT** **DEC**

Anzeige: 700.

Aufgabe: Umwandlung von Hexadezimal FFFFFFFF12 in Dezimal

Tastenfolge: **C-CE** **HEX** FFFFFFFF 12 **SHIFT** **DEC**

Anzeige: FFFFFFFF12. HEX

-238.

Anmerkung: Die Umwandlung von Hexadezimal FDABF41C01 bis FFFFFFFF ergibt eine negative Dezimalzahl.

Rechnen mit Zahlen in Hexadezimal-Schreibweise

In der Betriebsart "HEX" können auch Rechenoperationen durchgeführt werden. Dabei sind folgende Hinweise zu beachten:

- Es kann nur mit ganzen Zahlen gerechnet werden, die **◦**-Taste ist daher ohne Funktion.
- Bei Divisionen wird immer eine ganze Zahl als Ergebnis angezeigt. Etwaige Nachkommastellen bleiben unberücksichtigt.
- Falls mit einem Zwischenergebnis weitergerechnet wird, das Nachkommastellen aufweist, erfolgt eine Fehlermeldung (Symbol "E" wird angezeigt).
- Bei Kettenrechnungen sind ggf. Klammern zu verwenden, da keine Beachtung der Vorrangordnung erfolgt.
- Durch Drücken der **+/-**-Taste kann auf einfache Weise das Komplement der angezeigten Sedezimalzahl ermittelt werden.

Beispiel: A B **+/-** → FFFFFFFF55. HEX
+/- → AB. HEX

Aufgabe: A4 + BA

Tastenfolge: A4 **+** B A **=**

Anzeige: 15E. HEX

Aufgabe: 8×3 Tastenfolge: 8 **X** 3 **=**

Anzeige:

18. HEX

Aufgabe: $(12 + D) \times B$ Tastenfolge: **C-CE** (12 **+** D **)** **X** B **=**

Anzeige:

155. HEX

Aufgabe: $43A - 3CB =$ $+) A38 - 2FB =$

(Total)

Tastenfolge: **C-CE** **x↔M**4 3 A **-** 3 C B **M+**

Anzeige:

6F. HEX

Tastenfolge: A 3 8 **-** 2 F B **M+**

Anzeige:

73D. HEX

Tastenfolge: **RM**

Anzeige:

7AC. HEX

2.1.2 NORMALE BERECHNUNGEN

Die nachfolgenden Beispiele zeigen die Bedienung des Rechners bzw. die Tastenfolge für die Grundrechenfunktionen.

Bevor damit begonnen wird, die Beispiele nachzuvollziehen, sollte, um eine gemeinsame Ausgangsbasis sicherzustellen, eine kurze Überprüfung der Grundeinstellung des Computers erfolgen.

POWER-Schalter auf ON schalten

(Falls der Rechner automatisch abgeschaltet hat, zum Einschalten die **ON BRK** - Taste drücken)

CAL -Taste (grün) drücken

C-CE -Taste (rot) zweimal drücken

Das Anzeigefeld sollte nun nur die Anfangsinformationen enthalten. Ist dies nicht der Fall, die Grundeinstellung des Computers gemäß 2.1.1.3 (Seite 42) durchführen.

2.1.2.1 Addition und Subtraktion

Aufgabe: Rechnen von $123 + 654$

Tastenfolge

$\overset{r}{\boxed{1}}$ $\overset{a}{\boxed{2}}$ $\overset{b}{\boxed{3}}$

$\overset{\wedge}{\boxed{+}}$

$\overset{\sigma x}{\boxed{6}}$ $\overset{Sx}{\boxed{5}}$ $\overset{\bar{x}}{\boxed{4}}$

$\boxed{=}$

Anzeige

123.

123.

654.

777.

(123 + 654 = 777)

Aufgabe: $12 + 45,6 - 32,1 + 789 - 741 + 213$

Tastenfolge: 12 $\boxed{+}$ 45.6 $\boxed{-}$ 32.1 $\boxed{+}$ 789 $\boxed{-}$ 741 $\boxed{+}$ 213 $\boxed{=}$

Ergebnis: 286.5

2.1.2.2 Multiplikation und Division

Aufgabe: $841 \times 586 / 0,12$

Tastenfolge: 841 $\boxed{\times}$ 586 $\boxed{\div}$.12 $\boxed{=}$

Ergebnis: 4106883.333

Aufgabe: $427 + 54 \times 32 / 7 - 39 \times 2$

Tastenfolge: 427 $\boxed{+}$ 54 $\boxed{\times}$ 32 $\boxed{\div}$ 7 $\boxed{-}$ 39 $\boxed{\times}$ 2 $\boxed{=}$

Ergebnis: 595.8571429

Anmerkung: Multiplikation und Division haben Vorrang vor Addition und Subtraktion. Die Vorrangordnung wird beachtet. Der Computer führte intern zuerst die Multiplikation und Division durch (siehe auch 2.1.2.7, Seite 54).

Multiplikationen mit einer Konstanten

Der zuerst eingegebene Wert kann als Konstante verwendet werden.

Aufgabe: $5 \times 3 = ?$, $10 \times 3 = ?$, usw.

Tastenfolge: 3 $\boxed{\times}$ 5 $\boxed{=}$

Ergebnis: 15

Tastenfolge: 10 $\boxed{=}$

Ergebnis: 30

Divisionen mit einer Konstanten

Der als zweites eingegebene Wert kann als Konstante verwendet werden.

Aufgabe: $15 / 3 = ?$, $30 / 3 = ?$, usw.

Tastenfolge: 15 $\boxed{\div}$ 3 $\boxed{=}$

Ergebnis: 5

Tastenfolge: 30 $\boxed{=}$

Ergebnis: 10

Hinweis: Bei Kettenrechnungen bleiben die entsprechend der Vorrangordnung letzte Recheninstruktion und der jeweilige numerische Wert erhalten und können damit für weitere Berechnungen bzw. als Konstanten verwendet werden.

Ausgeführter Rechenvorgang	Konstante	
$a + b \times c =$	+bc	(Konstanten-Addition)
$a \times b \div c =$	$\div c$	(Konstanten-Division)
$a \div b \times c =$	$\frac{a}{b} \times$	(Konstanten-Multiplikation)
$a \times b - c =$	-c	(Konstanten-Subtraktion)

2.1.2.3 Potenz- und Wurzelfunktion

Aufgabe: 20^2

Tastenfolge: 20 $\boxed{x^2}$

Ergebnis: 400

Aufgabe: 3^3

Tastenfolge: 3 $\boxed{y^x}$ 3 $\boxed{=}$

Ergebnis: 27

Aufgabe: 3^4

Tastenfolge: 3 $\boxed{y^x}$ 4 $\boxed{=}$

Ergebnis: 81

Aufgabe: $\sqrt{25}$

Tastenfolge: 25 $\boxed{\sqrt{\quad}}$

Ergebnis: 5

Aufgabe: 3. Wurzel von 27

Tastenfolge: 27 $\boxed{\text{SHIFT}}$ $\boxed{3\sqrt{\quad}}$

Ergebnis: 3

Aufgabe: 4. Wurzel von 81

Tastenfolge: 81 $\boxed{\text{SHIFT}}$ $\boxed{x\sqrt{y}}$ 4 $\boxed{=}$

Ergebnis: 3

2.1.2.4 Prozentrechnung

Aufgabe: 45% von 2780 ($2,780 \times \frac{45}{100}$)
 Tastenfolge: 2780 **X** 45 **SHIFT** **Δ%**
 Ergebnis: 1251

Prozentuale Zunahme/Abnahme

Der Ausgangswert (= 100%) muß als zweiter Wert, d.h. nach Drücken der **-**-Taste eingegeben werden.

Aufgabe: Anstieg von 473 auf 547, Prozentuale Zunahme?
 Tastenfolge: 547 **-** 473 **SHIFT** **Δ%**
 Ergebnis: 15.6448203

Aufgabe: Rückgang von 547 auf 473, Prozentuale Abnahme?
 Tastenfolge: 473 **-** 547 **SHIFT** **Δ%**
 Ergebnis: -13.52833638

2.1.2.5 Reziprok-Rechnung

Aufgabe: $1/6 + 1/7$
 Tastenfolge: 6 **1/x** **+** 7 **1/x** **=**
 Ergebnis: 0.309523809

2.1.2.6 Speicherrechnung

Der zur Verfügung stehende Speicher wird über die blau beschrifteten Tasten **x↔M** **RM** **M+** angesprochen. Bevor mit dem Speicher gearbeitet wird, muß dessen Inhalt ggf. erst gelöscht werden. Der Speicherinhalt bleibt auch bei ausgeschaltetem Rechner erhalten. Die Belegung des Speichers zeigt das Symbol **M** in der rechten oberen Ecke des Displays an. Zum Löschen des Speichers die Tasten **C-CE** und **x↔M** drücken.

Das Addieren eines Rechenergebnisses oder eines eingegebenen Werts zum Speicherinhalt erfolgt durch Drücken der **M+**-Taste. Das Drücken der **RM**-Taste bringt den Speicherinhalt am Display zur Anzeige.

Tastensequenz	Anzeige
12 + 5 M+	17
C-CE	0
RM	17
8 ÷ 2 M+	4
RM	21 (Speicherinhalt)

Soll ein Rechenergebnis oder ein eingegebener Wert vom Speicherinhalt subtrahiert werden, muß vor dem Betätigen der **M+** -Taste durch Drücken der **+/-** -Taste das Vorzeichen gewechselt werden.

Tastenfolge	Anzeige
2 + 5 = +/- M+	-7
C-CE	0
RM	14

Das Drücken der **x↔M** -Taste überträgt den im Display angezeigten Wert in den Speicher. Der Speicherinhalt wird überschrieben.

Tastenfolge	Anzeige
12 x 2 = x↔M	24
C-CE	0
RM	24

2.1.2.7 Vorrangordnung und Verwendung der "Klammern"-Tasten

Die Verwendung der Tasten **(** und **)** ist dann zwingend notwendig, wenn eine Serie von Einzelberechnungen zu einem Rechengvorgang zusammengefaßt werden soll und dabei die Vorrangordnung der Operatoren außer Kraft gesetzt werden muß.

Sobald die **(** -Taste gedrückt wird, erscheint in der rechten oberen Ecke des Anzeigefelds das **()**-Symbol, welches so lange sichtbar bleibt, bis alle geöffneten Klammern wieder geschlossen wurden. Das Schließen der Klammer(n) über die **)** -Taste kann unterbleiben, wenn unmittelbar folgend die Tasten **=** oder **M+** zu drücken sind, da diese die gleiche Funktion erfüllen.

Die Berechnungen in der Klammer haben Priorität vor allen anderen Berechnungen. Die Klammerfunktion kann in einer Ebene bis zu fünfzehnmal verwendet werden. Als erstes werden die Berechnungen der innersten Klammer ausgeführt.

Aufgabe: $12 + 42 \div (8 - 6)$
 Tastenfolge: 12 **+** 42 **÷** **(** 8 **-** 6 **)** **=**
 Ergebnis: 33

Aufgabe: $126 \div [(3 + 4) \times (3 - 1)]$
 Tastenfolge: 126 **÷** **(** **(** 3 **+** 4 **)** **x** **(** 3 **-** 1 **)** **)** **)** **=**
 Ergebnis: 9

Anmerkung: Das Schließen der beiden Klammern unmittelbar vor der **=** -Taste (oder **M+** -Taste) kann unterbleiben.

Vorrangordnung

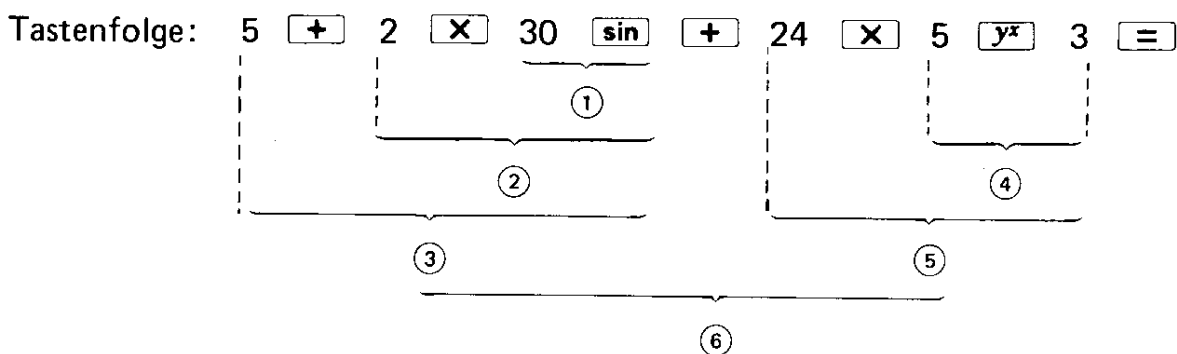
Da der Computer die Vorrangordnung der einzelnen Berechnungen beachtet, ist es möglich immer dann mathematische Formeln direkt (d.h. ohne daß Klammern eingefügt oder Formelumstellungen vorgenommen werden müssen) einzugeben, wenn es die Vorrangordnung nicht erfordert. Die Vorrangordnung bzw. die Reihenfolge, nach der die Berechnungen in einer längeren Rechenformel ausgeführt werden, ist wie folgt:

- (1) Funktionen wie sin oder x^2
- (2) y^x , $\sqrt[x]{y}$
- (3) x , \div (Berechnungen, die gleiche Priorität besitzen, werden der Reihe nach ausgeführt)
- (4) $+$, $-$
- (5) $=$, M+, $\Delta\%$

Beispiel: Tastenfolge und Reihenfolge der Berechnungen bei

$$5 + 2 \times \sin 30 + 24 \times 5^3 = 3006$$

Winkeleinheit DEG



Die Nummern ① ~ ⑥ geben die Reihenfolge der Berechnungen an.

Um die Berechnungen gemäß der Vorrangordnung bzw. der Verwendung von Klammern ausführen zu können, ist der Computer mit zusätzlichen Speichern ausgestattet. Diese ermöglichen es, daß mathematische Formeln, die bis zu acht Berechnungsebenen erforderlich machen, direkt eingegeben und berechnet werden können.

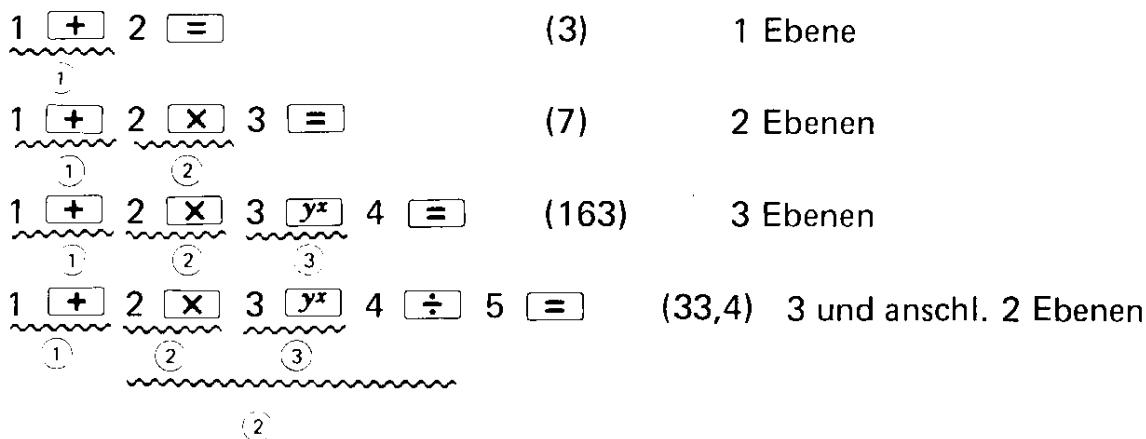
Die Klammerfunktion, welche ebenfalls eine Berechnungsebene ergibt, kann, solange keine weiteren Ebenen notwendig sind, bis zu fünfzehnmal verwendet werden.

$$a \times \underline{\underline{((b - c \times \underline{\underline{((d + e) \times f) \div g}})}} \dots\dots$$

↑ ↑
 Bis zu 15 Klammern sind möglich

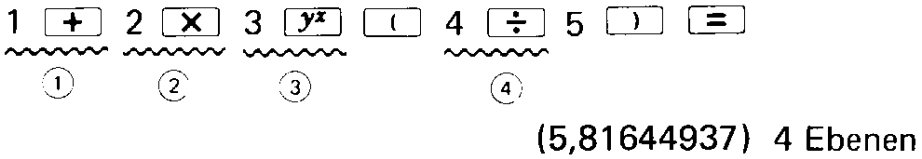
Funktionen mit nur einer Variablen wie (x^2 , $1/x$, $n!$, \rightarrow DEG, \rightarrow DMS, usw.) werden sofort nach dem Drücken der Taste berechnet und benötigen somit keine Ebene bzw. keinen Speicher.

Beispiel: Berechnungsebenen bzw. Zwischenspeicher
(ohne Klammerfunktion)

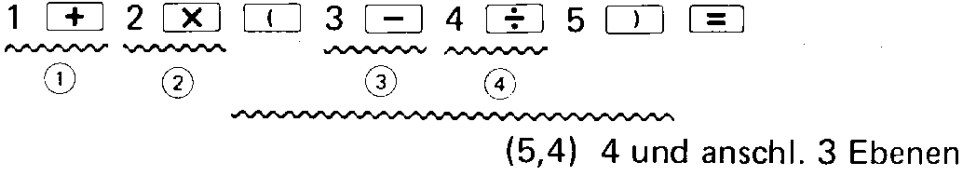


Mit Drücken der y^x -Taste sind 3 Ebenen erreicht. Nach Drücken der ÷ -Taste werden die Berechnungen "y^x" mit höherer Priorität und "x" mit gleicher Priorität als als "/" ausgeführt. Es bleiben somit nach Drücken der ÷ -Taste noch 2 Ebenen erhalten. Nach Drücken der = -Taste wird zuerst "/" (höhere Priorität) und anschließend "+" ausgeführt.

Beispiel: Berechnungsebenen bzw. Zwischenspeicher
(mit Klammerfunktion)



Die Berechnung in der Klammer wird zuerst ausgeführt. Anschließend folgt "y^x", dann "x" und zuletzt "+".



Nach Drücken der (-Taste wird zuerst "/", dann "-" ausgeführt. Auf das Drücken der = -Taste folgt "x" und zuletzt "+".

2.1.3 WISSENSCHAFTLICHE BERECHNUNGEN

Bei der Durchführung von Berechnungen, die Winkelwerte enthalten, muß auf die Winkeleinheit, die verwendet wird, geachtet werden. Das Vorwählen der Winkeleinheit erfolgt durch Drücken der Tasten **SHIFT** und **DRG**. Die gewählte Winkeleinheit wird im Display angezeigt.

2.1.3.1 Trigonometrische und inverse trigonometrische Funktionen (Arcusfunktionen)

Aufgabe: $\sin 30^\circ + \cos 40^\circ$
 Tastenfolge: Winkeleinheit auf DEG
 30 **sin** $+$ 40 **cos** $=$
 Ergebnis: 1,266044443

Aufgabe: $\cos 0,25\pi$
 Tastenfolge: Winkeleinheit auf RAD
 $.25$ **X** **SHIFT** **π** **=** **cos**
 Ergebnis: 0,707106781

Aufgabe: $\arcsin 0,5$
 Tastenfolge: Winkeleinheit auf DEG
 $.5$ **SHIFT** **sin⁻¹**
 Ergebnis: 30

Aufgabe: $\arccos -1$
 Tastenfolge: Winkeleinheit auf RAD
 1 **+/-** **SHIFT** **cos⁻¹**
 Ergebnis: 3,141592654 (Wert π)

Das Ergebnis einer inversen trigonometrischen Funktion kann nur in folgendem Bereich liegen:

$\theta = \sin^{-1} x, \theta = \tan^{-1} x$	$\theta = \cos^{-1} x$
DEG: $-90 \leq \theta \leq 90$ [°]	DEG: $0 \leq \theta \leq 180$ [°]
RAD: $-\frac{\pi}{2} \leq \theta \leq \frac{\pi}{2}$ [rad]	RAD: $0 \leq \theta \leq \pi$ [rad]
GRAD: $-100 \leq \theta \leq 100$ [g]	GRAD: $0 \leq \theta \leq 200$ [g]

2.1.3.2 Hyperbel- und inverse Hyperbelfunktionen (Areafunktion)

Die Anwahl der Hyperbelfunktion wird im Display durch das Symbol "HYP" bzw. bei inverser Hyperbelfunktion durch "SHIFT HYP" angezeigt.

Aufgabe: $\sinh 4$
 Tastenfolge: 4 **hyp** **sin**
 Ergebnis: 27,2899172

Aufgabe: $\operatorname{arsinh} 9$
 Tastenfolge: 9 **SHIFT** **archyp** **sin**
 Ergebnis: 2,893443986

2.1.3.3 Logarithmische Funktionen

Aufgabe: $\ln 21$, $\log 173$
 Tastenfolge: Natürlicher Logarithmus 21 **ln**
 Ergebnis: 3,044522438
 Allgemeiner Logarithmus 173 **log**
 Ergebnis: 2,238046103

2.1.3.4 Exponentialfunktionen

Aufgabe: $e^{3.0445}$ $10^{2.238}$
 Tastenfolge: Exponentialfunktion (e) 3.0445 **SHIFT** **e^x**
 Ergebnis: 20,99952881 (siehe $\ln 21$)
 Allgem. Exponentialfunktion 2.238 **SHIFT** **10^x**
 Ergebnis: 172,9816359 (siehe $\log 173$)

2.1.3.5 Fakultät

Aufgabe: $69!$ ($n! = 1 \times 2 \times 3 \times \dots \times n$)
 Tastenfolge: 69 **SHIFT** **n!**
 Ergebnis: 1,711224524E 98 ($1,711224524 \times 10^{98}$)

Anmerkung: Bei Berechnungen der Fakultät kann es sehr leicht zur Überschreitung der Rechenkapazität des Computers kommen, was die Fehlermeldung "E" zur Folge hat.

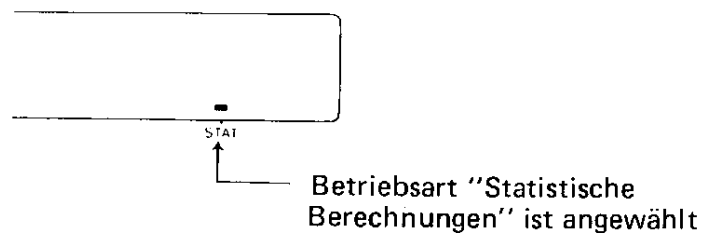
Aufgabe: ${}_8P_3 = \frac{8!}{(8-3)!}$
 Tastenfolge: 8 **SHIFT** **n!** **÷** (8 **-** 3 **)** **SHIFT** **n!** **=**
 Ergebnis: 336

2.1.4 STATISTISCHE BERECHNUNGEN

Die Betriebsart "Statistische Berechnungen" wird durch Drücken der Tasten **SHIFT** (gelb) und **STAT** (unter der roten **C-CE**-Taste) angewählt. Falls sich der Rechner in Betriebsart "RUN" oder "PRO" befindet, muß zuerst in die Betriebsart "CAL" umgeschaltet werden.

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 Tasten **SHIFT** und **STAT**.

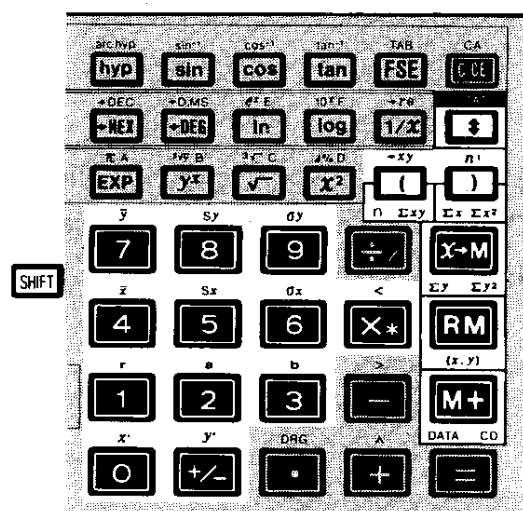


Wenn "STAT" angewählt ist, können die folgenden Berechnungen nicht durchgeführt werden:

- Speicherrechnungen
- Rechnungen, die Klammersausdrücke enthalten
- Koordinatenumwandlung
- Umwandlung "Hexadezimal-Dezimal"
- Rechnen mit Zahlen in hexadezimaler Schreibweise

In der Betriebsart "STAT" werden die in der folgenden Abbildung gekennzeichneten Tasten verwendet. Die Zweit-Funktion der verwendeten Tasten wird ebenfalls durch Drücken der **SHIFT**-Taste (gelb) angesprochen.

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	W	V	U
Statistik	n	Σx	Σx^2	Σxy	Σy	Σy^2

In diesem Zusammenhang ist jedoch folgendes zu beachten:

- Den Standardvariablen U bis Z wird ein Wert zugewiesen, d.h., ein gespeichertes Programm kann ggf. beeinträchtigt werden.
- Um vorher eingegebene bzw. berechnete Statistikwerte zu löschen, muß, bevor mit einer neuen Berechnung begonnen wird, die Betriebsart "STAT" rückgesetzt und anschließend wieder angewählt werden. Die Korrektur von falsch eingegebenen Daten ist über die Tasten **SHIFT** **CD** möglich.

Falls mit den errechneten Statistikwerten weitere Berechnungen durchzuführen sind, sollte dies zweckmäßigerweise in der Betriebsart "RUN" erfolgen. Aus dem folgenden Beispiel ist die grundsätzliche Vorgehensweise ersichtlich. Das Rechnen in Betriebsart "RUN" ist in Abschnitt 2.2.1 ausführlich erläutert.

Beispiel: Berechnung der Summe der Quadrate (S^2) der folgenden vier Werte 205, 221, 226 und 220

Die Berechnungsformel für S^2 lautet:

$$\begin{aligned} S^2 &= \Sigma(x - \bar{x})^2 \\ &= \Sigma x^2 - n\bar{x}^2 \\ &= \Sigma x^2 - \frac{1}{n} (\Sigma x)^2 \end{aligned}$$

Die Werte für Σx^2 (X), n (Z) und Σx (Y) werden in Betriebsart "STAT" ermittelt. Nach Umschaltung in "RUN" kann anschließend aus diesen Werten die Summe der Quadrate (S^2) errechnet werden.

Tastenfolge:

Umschalten auf "STAT"

CAL **SHIFT** **STAT**

Anzeige:

0.

Dateneingabe

205 221

226 220

Anzeige:

4.

Umschalten auf "RUN" und Berechnung von S^2

Anzeige:

>

X - Y * Y / Z _

246.

Statistische Berechnungen mit einer Variablen

In den statistischen Berechnungen finden folgende Benennungen Verwendung:

n Anzahl der eingegebenen Daten

$\sum x$ Gesamtsumme der eingegebenen Daten

$\sum x^2$ Quadratsumme der eingegebenen Daten

\bar{x} Mittelwert der eingegebenen Daten $\bar{x} = \frac{\sum x}{n}$

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

$$S_x = \sqrt{\frac{\sum x^2 - n\bar{x}^2}{n-1}}$$

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

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

$$\sigma_x = \sqrt{\frac{\sum x^2 - n\bar{x}^2}{n}}$$

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

Bei statistischen Berechnungen mit einer Variablen werden die Daten wie folgt eingegeben:

Wert oder, falls mehrmals der gleiche Wert einzugeben ist,

Wert Häufigkeit

Aufgabe: Berechnen der Standardabweichung S_x , des Mittelwerts \bar{x} und der Varianz $(S_x)^2$ der folgenden Daten:

Wert	35	45	55	65
Häufigkeit	1	1	5	2

Nach jeder Eingabe wird im Display die Anzahl der bisher eingegebenen Daten (n) angezeigt.

Tastenfolge:	Anzeige:
SHIFT STAT	0.
35 DATA	1.
45 DATA	2.
55 x 5 DATA	7.
65 x 2 DATA	9.

Mittelwert	SHIFT \bar{x}	53.88888889
Standardabweichung	SHIFT S_x	9.279607271
Varianz	x^2	84.11111111

Korrektur eingegebener Daten (CD): Die zuletzt eingegebenen Werte 65 x 2 sind falsch und müssen auf 60 x 2 berichtigt werden.

Tastenfolge:	Anzeige:
65 X 2 SHIFT CD	7.
60 X 2 DATA	9.

Mittelwert	SHIFT \bar{x}	52.78
Standardabweichung	SHIFT S_x	7.95
Varianz	x^2	63.19

Statistische Berechnungen mit zwei Variablen und Lineare Regression

Zusätzlich dazu, daß die statistischen Funktionen für zwei Variable (x und y) vorhanden sind und über die **Σxy** -Taste die Summe der xy-Werte errechnet werden kann, ist es in den statistischen Berechnungen für zwei Variable möglich, eine Beziehung (Korrelation) zwischen den beiden Datenfolgen herzustellen.

Jedes eingegebene Datenpaar besteht aus einem x- und y-Wert. Aus den Datenpaaren läßt sich eine Regressionslinie ableiten. Da die Beziehung zwischen den Datenpaaren über eine gerade Linie hergestellt wird, spricht man von einer "Linearen Regression".

Die Lineare Regression weist drei wichtige Werte auf: r , a und b . Die Gleichung der geraden Linie ist $y = a + bx$, wobei a den Punkt beschreibt, an dem die Linie die y -Achse schneidet, während b die Steilheit angibt.

Der Korrelationskoeffizient r ermöglicht eine Aussage darüber, mit welcher Genauigkeit ein Bezug zwischen den beiden Datenfolgen hergestellt werden kann. Eine perfekte Korrelation zwischen zwei Werten ist bei $r = 1$ bzw. $r = -1$ (negative Korrelation) gegeben, d.h., daß in diesem Fall, wenn der Wert der einen Variablen bekannt ist, der andere Wert mit einer Genauigkeit von 100% ermittelt werden kann. Je weiter der Korrelationskoeffizient r von 1 entfernt ist, um so ungenauer sind die ermittelten Werte.

Die nachfolgende Tabelle soll eine Bewertung des Korrelationskoeffizienten r ermöglichen.

	Korrelationskoeffizient (r)	Genauigkeit
Positive Korrelation	+ 0,8 ... + 1,0	sehr hoch
	+ 0,6 ... + 0,8	hoch
	+ 0,4 ... + 0,6	mittelmäßig
	+ 0,2 ... + 0,4	gering
Negative Korrelation	- 0,2 ... + 0,2	-----
	- 0,2 ... - 0,4	gering
	- 0,4 ... - 0,6	mittelmäßig
	- 0,6 ... - 0,8	hoch
	- 0,8 ... - 1,0	sehr hoch

Berechnungsformeln:

$$r = \frac{S_{xy}}{\sqrt{S_{xx} \cdot S_{yy}}}$$

$$\left(\begin{array}{l} S_{xx} = \Sigma x^2 - \frac{(\Sigma x)^2}{n} \\ S_{yy} = \Sigma y^2 - \frac{(\Sigma y)^2}{n} \\ S_{xy} = \Sigma xy - \frac{\Sigma x \cdot \Sigma y}{n} \end{array} \right)$$

$$\left. \begin{array}{l} a = \bar{y} - b\bar{x} \\ b = \frac{S_{xy}}{S_{xx}} \end{array} \right\} y = a + bx$$

Beispiel 1:

Ermittlung der voraussichtlichen Punktezahl in der Englischprüfung aus der in der Mathematikprüfung erreichten Punktezahl.

Die Grundlage der statistischen Berechnung bilden die erreichten Punktezahlen von sechs zufällig ausgewählten Prüfungskandidaten.

Prüfungskandidat Nr. n	Punkte in Math. x	Punkte in Englisch y
1	82	79
2	53	50
3	61	87
4	74	96
5	51	73
6	51	73

Tastenfolge:

82 (x,y) 79 DATA
 53 (x,y) 50 DATA
 61 (x,y) 87 DATA
 74 (x,y) 96 DATA
 51 (x,y) 73 X 2 DATA

Anzeige:

1.
2.
3.
4.
6.

Anmerkung: Mehrere identische Werte für x, y können über die X -Taste eingegeben werden.

Tastenfolge:

SHIFT r
 SHIFT a
 SHIFT b

Anzeige:

0.571587901
 34.26190476
 0.678571428

Der Korrelationskoeffizient $r = 0,57$ besagt, daß eine mittelmäßige Genauigkeit zu erwarten ist.

Die Gleichung der Geraden lautet für dieses Beispiel $y = 34,62 + 0,68x$.

Tastenfolge:

90 SHIFT y'

Anzeige:

95.33333333

Aufgrund dieser statistischen Berechnung kann ein Prüfungskandidat, der in der Mathematikprüfung 90 Punkte erreichte, damit rechnen, in der Englischprüfung 95 Punkte zu erreichen.

Beispiel 2:

Kann aus dem Gewicht, das ein Mann im Alter von 65 Jahren besitzt, auf die Gesamtlebensdauer geschlossen werden? Im Jahre 1950 wurden zehn Männer von annähernd gleicher Körpergröße (1,80 m) für ein Experiment ausgewählt, das Aufschluß darüber geben sollte, ob das Gewicht einen Einfluß auf die Gesamtlebensdauer hat.

Nummer	1	2	3	4	5	6	7	8	9	10
Lebensdauer	72	67	69	85	91	68	77	74	70	82
Gewicht mit 65	83	102	91	77	77	88	79	79	90	78

	Tasten folge:	Anzeige:
Tastenfolge:	[SHIFT] [STAT]	0.
	83 [(x,y)] 72 [DATA]	1.
	102 [(x,y)] 67 [DATA]	2.
(alle Daten eingeben)		⋮
	[SHIFT] [r]	-0.787476214

Der Korrelationsfaktor r gibt eine relative hohe negative Korrelation an. Ein höheres Gewicht bedeutet somit eine kürzere Lebensdauer.

Zum Zeichnen der Regressionslinie sind die Werte für a und b erforderlich.

[SHIFT] [a]	140.977975 (y-Achse)
[SHIFT] [b]	-0.775805391 (Steilheit)

Wie alt wird ein ca. 1,80 m großer Mann, der 86 kg wiegt?

86 [SHIFT] [y'] 74.25871137 Jahre

Um 90 Jahre alt zu werden, sollte ein 1,80 m großer Mann wie schwer sein?

90 [SHIFT] [x'] 65.70974576 kg

Wie schwer müßte man sein, um 150 Jahre alt zu werden? Aus dieser letzten Frage wird deutlich, daß die Lineare Regression nur in einem bestimmten eingegrenzten Bereich brauchbare Ergebnisse liefert.

2.1.5 RECHENBEREICH

Arithmetische Berechnungen:

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

Wissenschaftliche Berechnungen:

Funktion	Rechenbereich	Anmerkung
$\sin x$ $\cos x$ $\tan x$	DEG: $ x < 1 \times 10^{10}$ RAD: $ x < \frac{\pi}{180} \times 10^{10}$ GRAD: $ x < \frac{10}{9} \times 10^{10}$ Für $\tan x$ gelten folgende Einschränkungen DEG: $ x = 90 (2n - 1)$ RAD: $ x = \frac{\pi}{2} (2n - 1)$ $n = \text{Ganze Zahl}$ GRAD: $ x = 100 (2n - 1)$	
$\sin^{-1} x$ $\cos^{-1} x$	$-1 \leq x \leq 1$	
$\tan^{-1} x$	$ x < 1 \times 10^{100}$	
$\ln x$ $\log x$	$1 \times 10^{-99} \leq x < 1 \times 10^{100}$	$(\ln x = \log_e x)$
e^x	$-1 \times 10^{100} < x \leq 230.2585092$	$(e \approx 2.718281828)$
10^x	$-1 \times 10^{100} < x < 100$	
y^x	<ul style="list-style-type: none"> • $y > 0$: $-1 \times 10^{100} < x \log y < 100$ • $y = 0$: $x > 0$ • $y < 0$: x: ganzzahlig oder $1/x$: ungerade $-1 \times 10^{100} < x \log y < 100$ 	$y^x = 10^{x \cdot \log y}$
$\sqrt[x]{y}$	<ul style="list-style-type: none"> • $y > 0$: $-1 \times 10^{100} < \frac{1}{x} \log y < 100, x \neq 0$ • $y = 0$: $x > 0$ • $y < 0$: x oder $1/x$: ganzzahlig ($x \neq 0$) $-1 \times 10^{100} < \frac{1}{x} \log y < 100$ 	$\sqrt[x]{y} = 10^{\frac{1}{x} \cdot \log y}$
$\sqrt[3]{x}$	$ x < 1 \times 10^{100}$	
$\sinh x$ $\cosh x$ $\tanh x$	$-227.9559242 \leq x \leq 230.2585092$	
$\sinh^{-1} x$	$ x < 1 \times 10^{50}$	
$\cosh^{-1} x$	$1 \leq x < 1 \times 10^{50}$	
$\tanh^{-1} x$	$ x < 1$	
\sqrt{x}	$0 \leq x < 1 \times 10^{100}$	
x^2	$ x < 1 \times 10^{50}$	

Funktion		Rechenbereich	Anmerkung
$\frac{1}{x}$		$ x < 1 \times 10^{100}$ $x \neq 0$	
n!		$0 \leq n \leq 69$ (n = Ganze Zahl)	
→DEG		$ x < 1 \times 10^{100}$	
→D.MS		$ x < 1 \times 10^{100}$	
HEX → DEC		$0 \leq x \leq 2540BE3FF$ $FDABF41C01 \leq x \leq FFFFFFFF$	x ist in "HEX" eine ganze Zahl
DEC → HEX		$ x \leq 9999999999$	x = ganzzahlig
$x, y \rightarrow r, \theta$		$(x^2 + y^2) < 1 \times 10^{100}$ $\frac{y}{x} < 1 \times 10^{100}$	$r = \sqrt{x^2 + y^2}$ $\theta = \tan^{-1} \frac{y}{x}$
$r, \theta \rightarrow x, y$		$r < 1 \times 10^{100}$ $ r \sin \theta < 1 \times 10^{100}$ $ r \cos \theta < 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	$ x < 1 \times 10^{50}$ $ y < 1 \times 10^{50}$ $ \Sigma x < 1 \times 10^{100}$ $\Sigma x^2 < 1 \times 10^{100}$ $ \Sigma y < 1 \times 10^{100}$ $\Sigma y^2 < 1 \times 10^{100}$ $ \Sigma xy < 1 \times 10^{100}$ $ n < 1 \times 10^{100}$	
	\bar{x}	$n \neq 0$	
	Sx	$n \neq 1$ $0 \leq \frac{\Sigma x^2 - n\bar{x}^2}{n-1} < 1 \times 10^{100}$	
	σx	$n \neq 0$ $0 \leq \frac{\Sigma x^2 - n\bar{x}^2}{n} < 1 \times 10^{100}$	
	\bar{y}	$n \neq 0$	
	Sy	$n \neq 1$ $0 \leq \frac{\Sigma y^2 - n\bar{y}^2}{n-1} < 1 \times 10^{100}$	

Funktion	Rechenbereich	Anmerkung	
Statistische Berechnungen	$n \neq 0$ $0 \leq \frac{\Sigma y^2 - n\bar{y}^2}{n} < 1 \times 10^{100}$		
	$n \neq 0$ $0 < (\Sigma x^2 - n\bar{x}^2) \cdot (\Sigma y^2 - n\bar{y}^2) < 1 \times 10^{100}$ $\left \Sigma xy - \frac{\Sigma x \cdot \Sigma y}{n} \right < 1 \times 10^{100}$ $\left \frac{\Sigma xy - \frac{\Sigma x \cdot \Sigma y}{n}}{\sqrt{(\Sigma x^2 - n\bar{x}^2) \cdot (\Sigma y^2 - n\bar{y}^2)}} \right < 1 \times 10^{100}$		
	$n \neq 0$ $0 < \Sigma x^2 - n\bar{x}^2 < 1 \times 10^{100}$ $\left \Sigma xy - \frac{\Sigma x \cdot \Sigma y}{n} \right < 1 \times 10^{100}$ $\left \frac{\Sigma xy - \frac{\Sigma x \cdot \Sigma y}{n}}{\Sigma x^2 - n\bar{x}^2} \right < 1 \times 10^{100}$		
	a	a ist in derselben Bedingung wie b, und $ \bar{y} - b\bar{x} < 1 \times 10^{100}$	
	y'	$ a + bx < 1 \times 10^{100}$	
	x'	$\left \frac{y - a}{b} \right < 1 \times 10^{100}$	

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 Kumulierung 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 und $\sqrt[x]{y}$ auf.

2.1.6 MATRIZEN

In der Betriebsart CAL besitzt der COMPUTER eine Funktion zur Berechnung von Matrizen und ihrer Determinantenwerte.

Eine Matrix ist, wie nachfolgend gezeigt, eine rechteckige Anordnung a_{ik} ($i = 1, 2 \dots m$; $k = 1, 2 \dots n$) einer gegebenen Zahlenmenge ($m \times n$ Elemente).

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}$$

Bei diesem Computer kann eine solche Anordnung als Matrix X , Y oder M dargestellt werden. Die einzelnen Elemente, aus denen eine Matrix besteht, werden als Matrixelemente bezeichnet. Matrixelement a_{11} wird als $X(1,1)$, $Y(1,1)$ bzw. $M(1,1)$ dargestellt. Die Horizontalreihen der Matrixelemente werden als Zeilen, die Vertikalreihen als Spalten definiert.

2.1.6.1 Aufbau der Matrizen



Mit dem COMPUTER können drei Matrizen, X , Y und M , definiert werden. Jede Matrix kann innerhalb eines Bereichs von 1 bis 99 Vertikalreihen (d.h. Spalten) bzw. Horizontalreihen (d.h. Zeilen) dargestellt werden. Das Gesamtformat der Matrix ist jedoch von der Speicherkapazität des COMPUTERS abhängig.

Weiterhin werden die Matrizen X , Y und M im gleichen Speicherbereich gespeichert wie die BASIC-Datenfelder $X(*,*)$, $Y(*,*)$ und $M(*,*)$. Das heißt, daß die Werte der in der BASIC-Betriebsart eingegebenen Matrixelemente in der CAL-Betriebsart berechnet werden können.







Zur Eingabe der Werte der Matrixelemente in der BASIC-Betriebsart sind folgende Punkte zu beachten:



- (1) Die Matrixelemente $X(i,k)$ entsprechen den BASIC-Datenfeldelementen $X(i-1,k-1)$. So entspricht z.B. $X(1,2)$ dem Datenfeld $X(0,1)$.
- (2) Die eingespeicherten Werte der Matrixelemente werden bei der Ausführung der BASIC-Befehle RUN, CLEAR und NEW gelöscht.

2.1.6.2 Eingabe der Matrixelemente

Zur Aktivierung des MATRIX-Modus wird in der CAL-Betriebsart die Taste **SHIFT**  oder **SHIFT**  gedrückt. In diesem Modus können die Elemente einer Matrix zur Berechnung der Matrix bzw. zur Durchführung von Matrixoperationen eingegeben und angezeigt werden.

Nachfolgend werden die zur Eingabe und Darstellung der Matrixelemente benötigten Tasten und Funktionen beschrieben.

Tasten	Funktion
 	<ul style="list-style-type: none"> • Aktiviert den MATRIX-Modus. • Ermöglicht die Eingabe der Elemente für Matrix X, danach für Matrix Y, sowie die Berechnung der Matrizen. • Löschen des MATRIX-Modus durch erneutes Drücken beider Tasten.
 	<ul style="list-style-type: none"> • Aktiviert den MATRIX-Modus. • Ermöglicht die Durchführung von Matrizenrechnungen. • Löschen des MATRIX-Modus durch erneutes Drücken beider Tasten.
	<ul style="list-style-type: none"> • Zur Einspeicherung der Zeilen- bzw. Spaltenzahl (Bestimmung des Matrixformats) sowie der Daten für die einzelnen Matrixelemente. Danach ist der Computer bereit für die nächste Dateneingabe.
	<ul style="list-style-type: none"> • Cursorbewegung um eine Spalte nach rechts. (Wenn sich der Cursor am letzten Element einer Zeile befindet, springt er zum ersten Element.)
	<ul style="list-style-type: none"> • Cursorbewegung um eine Spalte nach links. (Wenn sich der Cursor am ersten Element einer Zeile befindet, springt er zum letzten Element.)
	<ul style="list-style-type: none"> • Cursorbewegung um eine Zeile nach oben (d.h. zum nächsthöheren Element in einer Spalte). • Rückführung zum vorangehenden Bedienungsschritt.
	<ul style="list-style-type: none"> • Cursorbewegung um eine Zeile nach unten (d.h. zum nächsttieferen Element in einer Spalte). • Versetzt den Computer in Wartestellung für den nächsten Bedienungsschritt.

Die Werte für die einzelnen Matrix-Elemente können während des Eingabe-Modus berechnet werden. Die Rechenoperationen müssen mit der  -Taste abgeschlossen werden, das Ergebnis wird mit der  -Taste dem aktuellen Element zugewiesen.

Beispiel 1: Eingabe der zwei folgenden Matrizen:

$$X = \begin{bmatrix} 10/3 & -5 & 2 \\ 8 & 2 & 23 \end{bmatrix}$$

$$Y = \begin{bmatrix} 5/3 & 3 & 2 \\ -1 & 0 & -8 \end{bmatrix}$$

Bedienung:

SHIFT ↓

MATRIX:X(0 _ , 0)

Da Matrix X nicht definiert ist, wird bei Aktivierung des MATRIX-Modus (0, 0) angezeigt.

2

MATRIX:X(2 _ , 0)

"2" wird zur Bestimmung der Zeilenzahl eingegeben.

ENTER

MATRIX:X(2, 0 _)

3 ENTER

X(1, 1) 0.

Danach wird durch Eingabe von "3" das Gesamtformat der Matrix (2, 3) bestimmt. Der Computer ist nun bereit für die Eingabe des Werts für das erste Matricelement $X(1, 1)$.

10 ÷/

X(1, 1) 10.

3 =

X(1, 1) 3.333333333

ENTER

X(1, 2) 0.

Nach der Einspeicherung des Elements $X(1, 1)$ steht der Computer bereit für die Eingabe des nächsten Elements $X(1, 2)$.

5 +/-

X(1, 2) -5

ENTER 2

X(1, 3) 2.

ENTER 8

X(2, 1) 8.

ENTER 2

X(2, 2) 2.

ENTER 23

X(2, 3) 23.

ENTER

MATRIX:Y(0 _ , 0)

Nach der Eingabe aller Daten für Matrix X muß das Format von Matrix Y definiert werden. Danach werden die Werte für die Elemente von Matrix Y wie für Matrix X beschrieben eingegeben.

2 ENTER

MATRIX:Y(2, 0 _)

3

MATRIX:Y(2, 3 _)

ENTER 5 \div 3 =

Y(1, 1) 1.666666667

ENTER 3

Y(1, 2) 3.

ENTER 2

Y(1, 3) 2.

ENTER 1 +/-

Y(2, 1) -1.

ENTER 0

Y(2, 2) 0.

ENTER 8 +/-

Y(2, 3) -8.

ENTER










MATRIX OPERATION

Nach der Eingabe aller Elemente für Matrix Y erscheint die Meldung "MATRIX OPERATION" auf der Anzeige und bestätigt, daß der Computer zur Durchführung von Rechenoperationen bereitsteht. Wenn nur eine Matrix (MATRIX X) berechnet werden soll, muß bei der Bestimmung des Formats von Matrix Y für die Zahl der Zeilen bzw. Spalten C-CE ENTER eingegeben werden.

2.1.6.3 Matrizenrechnungen

Solange die Meldung "MATRIX OPERATION" auf der Anzeige ausgewiesen wird, können die folgenden Tasten für Matrixoperationen verwendet werden:

Tasten	Funktion
$\boxed{+}$	<p>$X + Y \rightarrow X$: Addition.</p> <p>Das Ergebnis der Addition von Matrix X zu Matrix Y ist eine neue Matrix X.</p> <p>Diese Operation ist nur dann durchführbar, wenn das Format (d.h. Zahl der Zeilen und Spalten) beider Matrizen identisch ist.</p>
$\boxed{-}$	<p>$X - Y \rightarrow X$: Subtraktion.</p> <p>Das Ergebnis der Subtraktion der Matrix Y von der Matrix X ist eine neue Matrix X.</p> <p>Diese Operation ist nur dann durchführbar, wenn das Format (d.h. Zahl der Zeilen und Spalten) beider Matrizen identisch ist.</p> <p>Beispiel:</p> $\begin{bmatrix} 2 & 3 \\ 5 & 2 \end{bmatrix} - \begin{bmatrix} 1 & 6 \\ 3 & -1 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & -3 \\ 2 & 3 \end{bmatrix}$
$\boxed{X*}$	<p>$X \cdot Y \rightarrow X$: Multiplikation.</p> <p>Diese Operation ist nur durchführbar, wenn die Zahl der Spalten in Matrix X gleich der Zahl der Zeilen in Matrix Y ist.</p>
$\boxed{\div /}$	<p>$X \cdot Y^{-1} \rightarrow X$: Multiplikation mit der Matrix X und der inversen Matrix Y.</p> <p>Diese Operation ist nur durchführbar, wenn Zahl der Spalten in Matrix X gleich der Zahl der Spalten in Matrix Y^{-1} ist.</p>
$\boxed{1/x}$	<p>$X^{-1} \rightarrow X$: Berechnung der inversen Matrix von Matrix X.</p> <p>Das Ergebnis ist eine neue Matrix X.</p> <p>Diese Operation ist nur möglich, wenn das Format von Matrix X quadratisch ist (d.h. Zahl der Zeilen ist gleich der Zahl der Spalten).</p>
$n \boxed{+}$	<p>$n + X \rightarrow X$: Addition des Skalars n zu den Elementen der Matrix X.</p> <p>Bei dieser Operation wird der Skalar n zu den einzelnen Elementen der Matrix X addiert.</p> <p>ANMERKUNG: Die Addition von Skalaren ist eines der einzigartigen Merkmale des COMPUTER.</p>

Tasten	Funktion
n 	<p>$n - X \rightarrow X$: Subtraktion der einzelnen Elemente der Matrix X von einem Skalar n.</p> <p>Bei dieser Operation werden die einzelnen Elemente der Matrix X von dem Skalar n subtrahiert, wobei eine Matrix X erhalten wird, deren Elemente das Ergebnis der Subtraktion sind.</p> <p>ANMERKUNG: Die Subtraktion von Skalaren ist eines der einzigartigen Merkmale des COMPUTER.</p> <p>Beispiel:</p> $2 - \begin{bmatrix} 1 & 6 \\ 3 & -1 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & -4 \\ -1 & 3 \end{bmatrix}$
n 	<p>$n \cdot X \rightarrow X$: Multiplikation der Elemente von Matrix X mit einem Skalar n.</p>
n 	<p>$n \cdot X^{-1} \rightarrow X$: Multiplikation der Elemente der inversen Matrix X^{-1} mit einem Skalar n.</p> <p>Diese Operation ist nur durchführbar, wenn es sich bei Matrix X um eine quadratische Matrix handelt.</p>
	<p>$X \longleftrightarrow Y$: Austausch von Matrix X gegen Matrix Y.</p>
	<p>$X^t \rightarrow X$: Transposition der Matrix X. Das Ergebnis ist eine neue Matrix X.</p>
	<p>$X \rightarrow X$ (Anzeige): Gibt den Wert der Determinante für Matrix X an.</p> <p>Diese Operation ist nur durchführbar, wenn es sich bei Matrix X um eine quadratische Matrix handelt.</p>
	<p>$-X \rightarrow X$: Vorzeichenumkehr der einzelnen Elemente von Matrix X.</p>
	<p>$X \cdot X \rightarrow X$: Quadrierung der Matrix X.</p> <p>Diese Operation ist nur durchführbar, wenn es sich bei Matrix X um eine quadratische Matrix handelt.</p>
	<p>$X \rightarrow M$: Einspeicherung der Werte von Matrix X im Speicherbereich von Matrix M (mit gleichzeitiger Löschung des alten Speicherinhalts von Matrix M).</p> <p>Durch Drücken dieser Taste kann der Wert von Matrix X bis nach der Matrizenrechnung erhalten werden.</p>

Tasten	Funktion
RM	$M \rightarrow X$: Übertragung des Speicherinhalts (Matrix M) zu Matrix X , bei gleichzeitiger Löschung des alten Inhalts Elemente von Matrix X .
M+	$X + M \rightarrow M$: Kumulative Addition des Werts von Matrix X zum Speicherinhalt von Matrix M . Zum Durchführen dieser Rechenoperation müssen Matrix X und M jeweils die gleiche Anzahl von Reihen und Spalten aufweisen.

ANMERKUNGEN:

- * Die Ausführung einer Rechenoperation kann durch Drücken der Taste **BRK** unterbrochen werden. Hierbei bleiben die alten Werte der Matrizen X , Y und M erhalten.
- * Eine Division der Elemente von Matrix X durch einen Skalar n ist durch folgende Eingabe möglich: n **1/x** **x***.
- * Wenn die meisten Elemente einer einzugebenden Matrix den gleichen Wert haben, kann die Eingabe über die Operation 'Skalar $n + X$ ' (n **+**) vorgenommen werden. Danach haben alle Elemente den Wert 'n' und es brauchen nur noch die von 'n' abweichenden Elemente korrigiert werden. Es ist jedoch zu beachten, daß zuvor alle Elemente auf 'NULL' gesetzt sein müssen.

Nach jeder Rechenoperation erscheint wieder die Meldung "MATRIX OPERATION" auf der Anzeige, wobei der COMPUTER für die nächste Rechenoperation bereit ist. Nach Ermittlung des Determinantenwerts von Matrix X durch Drücken der Taste **D** kann die Meldung "MATRIX OPERATION" wieder abgerufen werden, indem man die Taste **↓**, **↑**, **◀**, **▶**, **C-CE** bzw. **ENTER** drückt.

Beispiel 2: Berechnung von $X + Y$ mit den Werten, die für die einzelnen Elemente der Matrizen X und Y in Beispiel 1 eingegeben wurden.

$$X = \begin{bmatrix} 10/3 & -5 & 2 \\ 8 & 2 & 23 \end{bmatrix} \quad Y = \begin{bmatrix} 5/3 & 3 & 2 \\ -1 & 0 & -8 \end{bmatrix}$$

Bedienung:

	MATRIX OPERATION
+	X+Y→X

(Während der Rechenausführung erscheint zur Bestätigung die Meldung "BUSY".)

MATRIX OPERATION

Das Ergebnis der Addition ist:

↓	MATRIX:X(2 _ , 3)
↓	X(1, 1) 5.
▶	X(1, 2) -2.
SHIFT ↓	 0.

Hiermit ist der MATRIX-Modus abgeschaltet.

Wenn nach Erscheinen der Meldung "MATRIX OPERATION" eine der Zahlentasten bzw. die Taste **•** gedrückt wird, sind Berechnungen mit Skalaren möglich.

Beispiel 3: Berechnung von $1/25 * X \rightarrow X$ mit dem Rechenergebnis für Matrix X in Beispiel 2

Bedienung :

SHIFT ↑

MATRIX OPERATION

2

SCALAR 2.

5

SCALAR 25.

1/x

SCALAR 0.04

X*

0.04*X → X

↓

MATRIX OPERATION

↓

MATRIX: X(2 , 3)

SHIFT ↑

X(1,1) 0.2

0.

Beispiel 4: Berechnung der folgenden linearen Gleichungen mit drei Unbekannten.

$$\begin{cases} 2x + 5y - z = -1 \\ x - y + 4z = 12 \\ 3x + 2y + z = 9 \end{cases}$$

TIP: Wie unten gezeigt, Matrizen X und Y eingeben und die Lösungen für x , y und z mit der Formel $X^{-1} \cdot Y$ errechnen.

$$X = \begin{bmatrix} 2 & 5 & -1 \\ 1 & -1 & 4 \\ 3 & 2 & 1 \end{bmatrix} \quad Y = \begin{bmatrix} -1 \\ 12 \\ 9 \end{bmatrix}$$

Bedienung:

Die Tasten **SHIFT** und **↓** drücken, um den MATRIX-Modus zu aktivieren. Danach die Werte der Elemente für Matrix X und Matrix Y wie in Beispiel 1 beschrieben eingeben.

	MATRIX OPERATION
1/x	invX→X
	MATRIX OPERATION
X*	X*Y→X
	MATRIX OPERATION
↓	MATRIX:X(3 _ , 1)
↓	X(1, 1) 3.
↓	X(2, 1) -1.
↓	X(3, 1) 2.

Es ergeben sich folgende Lösungen für x , y und z :

$$x = 3, y = -1, z = 2$$

ANMERKUNG:

Matrizenrechnung basiert auf dem gebräuchlichen Eliminationsverfahren. Da Computer Berechnungen jedoch strikt numerisch durchführen und überzählige Nachkommastellen abbrechen, können bei der Berechnung einer Determinante oder einer inversen Matrix Fehler auftreten.

Beispiel 5: Berechnung der inversen Matrix von $\begin{bmatrix} 3 & 1 \\ 1 & 1/3 \end{bmatrix}$

Bei dieser Matrix handelt es sich nicht um eine reguläre Matrix, weshalb theoretisch dazu auch keine inverse Matrix existiert. Computer speichern den Wert $1/3$ jedoch als "0,33 ...", weshalb die Darstellung der folgenden inversen Matrix möglich ist.

$$\begin{bmatrix} 3 & 1 \\ 1 & 0.33\dots3 \end{bmatrix}^{-1} = \begin{bmatrix} -33\dots3 & 1.E10 \\ 1.E10 & -3.E10 \end{bmatrix}$$

Aus diesem Grunde können die mit Computern erhaltenen Ergebnisse Fehler aufweisen. Es ist daher zu beachten, daß, abhängig von den gegebenen Anforderungen an die Rechengenauigkeit der Matrizenrechnung, gegebenenfalls eine Prüfung des Ergebnisses durch ein anderes Verfahren notwendig ist.

Wenn im obigen Beispiel der Wert der Determinante durch Multiplikation der ursprünglichen Matrix X mit 3 erhalten wird, bestätigt es sich, daß Matrix X keine reguläre Matrix ist, da das Ergebnis der Multiplikation $0 \left(\begin{vmatrix} 9 & 3 \\ 3 & 1 \end{vmatrix} = 0 \right)$ ist.

ANMERKUNG:

Da Matrizenrechnungen nicht durch eine einzelne Rechenoperation (d.h. einmalige Multiplikation) ausführbar sind, benötigt der **COMPUTER** einige Zeit zur Ausführung. Zur Berechnung der inversen Matrix einer aus 7 Zeilen und 7 Spalten bestehenden Matrix werden etwa 6 Sekunden benötigt. Die zur Berechnung benötigte Zeit hängt jedoch auch von den Werten der Matrizenelemente ab.

Für Matrizenrechnung erforderliche Speicherkapazität

Da für Matrizenrechnung und BASIC-Programme ein gemeinsamer Speicherbereich vorgegeben ist, muß die freie Speicherkapazität (d.h. die durch MEM **ENTER** in der BASIC-Betriebsart zu ermittelnde Kapazität) größer sein als die für Matrizenrechnung benötigte Kapazität, die mit folgender Formel ermittelt werden kann.

[(Zahl der Zeilen von Matrix X) \times (Zahl der Spalten von Matrix X) \times 8 + 7] Byte
 + [(Zahl der Zeilen von Matrix Y) \times (Zahl der Spalten von Matrix Y) \times 8 + 7] Byte
 + [(Zahl der Zeilen von Matrix M) \times (Zahl der Spalten von Matrix M) \times 8 + 7] Byte
 + [(Zahl der Zeilen der Resultantenmatrix) \times (Zahl der Spalten der Resultantenmatrix)
 \times 8 + 7] Byte

Wenn jedoch keine Matrix Y bzw. Matrix M verwendet werden soll, nimmt deren Wert (Zahl der Zeilen bzw. Spalten) in der vorangehenden Formel 0 an, und die benötigte Kapazität für Matrix Y bzw. M wird 0. Der Speicherraum für die Resultantenmatrix wird nur während der Ausführung einer Berechnung belegt und danach wieder freigegeben. Die Ausführung von $\boxed{x \rightarrow M}$ bzw. \boxed{RM} oder $\boxed{\downarrow}$ erfolgt ohne Resultantenmatrix. Zwei Resultantenmatrixen sind erforderlich für Matrixrechnungen mit $\boxed{\div /}$ und $n \boxed{\div /}$, weil hierbei zwei Rechnungen (d.h. Inversion und Multiplikation) durchgeführt werden.

- Wenn bei aktiviertem MATRIX-Modus die Meldung "MEMORY OVER" (Speicherkapazität erschöpft) auf der Anzeige erscheint, kann durch Löschen von BASIC-Variablen bzw. Programmen zusätzlicher Speicherraum für die Matrizenrechnung geschaffen werden.

Beispiel: Multiplikation der zwei folgenden Matrizen ($X \cdot Y \rightarrow X$)

$$X = \begin{bmatrix} 2 & 3 \\ 5 & 1 \end{bmatrix} \quad Y = \begin{bmatrix} 8 & 30 \\ 7 & 15 \end{bmatrix}$$

(Matrix M nicht definiert)

Die erforderliche Speicherkapazität errechnet sich folgendermaßen:

$$(2 \times 2 \times 8 + 7) + (2 \times 2 \times 8 + 7) + (2 \times 2 \times 8 + 7) = 117 \text{ Byte}$$

\uparrow
Matrix X

\uparrow
Matrix Y

\uparrow
Resultantenmatrix

2.1.6.4 Ausdruck der Matrizen

Zum Ausdruck der Daten (d.h. Werte der einzelnen Elemente) ist das folgende Programm vorzubereiten und auszuführen. Wenn zur Programmausführung jedoch "RUN" eingegeben und die Taste \boxed{ENTER} gedrückt wird, werden die Matrizenwerte aus dem Speicher gelöscht. Die Programmausführung muß daher immer mit der Taste \boxed{DEF} ausgelöst werden.

```

100 "M":INPUT "ROW";II
110 INPUT "COLUMN";JJ
120 FOR I=0 TO II-1
130 FOR J=0 TO JJ-1
140 LPRINT "X(";I+1;";";J+1;")=";X(I,J)
150 NEXT J:NEXT I:END
    
```

Zum Ausdruck der Werte von Matrix *Y* bzw. Matrix *M* müssen die beiden "X" in Zeile 140 durch "Y" bzw. "M" ersetzt werden.

Bedienung: Im RUN-Modus die Tasten **DEF** und **M** drücken. Die Werte der Matrix werden an den Drucker ausgegeben.

2.1.6.5 Fehlermeldung

Wenn während der Matrizenrechnung ein Fehler auftritt, erscheint eine der folgenden Fehlermeldungen zusammen mit einem "E" (Fehlerzeichen) auf der Anzeige. In diesem Fall muß die Taste **C-CE** gedrückt werden, um den Fehlerzustand aufzuheben, wonach die Fehlermeldung verschwindet und wieder "MATRIX OPERATION" angezeigt wird. Hierbei bleiben die vor der Ausführung der letzten Rechenoperation gegebenen Matrizenwerte unverändert erhalten.

Fehlermeldung	Ursache
IMPOSSIBLE CALCULATION	<ul style="list-style-type: none"> • Matrizen verschiedenen Formats. Die Matrizen stimmen nicht mit dem für Addition, Subtraktion bzw. Multiplikation der Matrizen erforderlichen Format überein. Dasselbe gilt für die versuchte Inversion bzw. Quadrierung einer nicht quadratischen Matrix.
MEMORY OVER	<ul style="list-style-type: none"> • Zu wenig Speicherraum. Bei der Operation $X \rightarrow M$ ist kein ausreichender Speicherraum für Matrix <i>M</i> vorhanden, oder der Speicherraum ist zu klein für die Durchführung arithmetischer Operationen.
DIVISION BY ZERO	<ul style="list-style-type: none"> • Versuchte Teilung durch Null. Teilung durch Null bei der Matrixinversion.
OVER FLOW	<ul style="list-style-type: none"> • Überlauf bei einer arithmetischen Rechenoperation.

2.2 SHARP PC-1403 – EINSATZ ALS BASIC-RECHNER

2.2.1 RECHNEN OHNE PROGRAMMUNTERSTÜTZUNG (RUN-MODE)

Der Computer kann auch 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 ausgeführt werden (RUN-Mode).

Dabei kann man den Computer 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:

C-CE 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) Klammern

**(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** **C-CE** -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.

2.2.1.2 Rechnengenauigkeit

Der Computer verarbeitet Zahlen mit einer Genauigkeit bis zu 10 Stellen. Dies führt bei einigen Rechnungen zu Rundungsfehlern. Die Genauigkeitsbeschränkung tritt auch beim Potenzieren oder beim Addieren von Zahlen sehr unterschiedlicher Größe auf.

Beispiele:

Eingabe

A = 199 * 199 * 199

ENTER

Ausgabe

7880599.

Eingabe

A - 199 ^ 3

ENTER

Ausgabe

0.00024

Eingabe

1 E 12 + 1 - 1 E 12

ENTER

Ausgabe

0.

2.2.1.3 Wissenschaftliche Schreibweise

Man kann Zahlen in den Computer auch im wissenschaftlichen Format eingeben. Damit ist es möglich, Zahlen bis zu einer Größenordnung von $\pm 9.999999999E \pm 99$ 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 das Potenzieren 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 auf das zu korrigierende Zeichen und überschreiben dieses mit dem neuen Zeichen.

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

Zum Einfügen von Zeichen drücken Sie die Tasten  und . Es wird ein Platzhalter () in die Formel eingefügt, der durch ein neues Zeichen überschrieben werden kann. Durch Drücken der  -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  -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  -Taste und stellen Sie den Cursor an das Zeilenende. Nun können Sie noch zusätzlich einige Zeichen eingeben. Die Eingabe muß wieder mit  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 **0** **ENTER**

0.

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.

Beispiele:

$$3 * 5 + 7 * 4$$

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. Potenzen
5. Vorzeichen +, –
6. Multiplikation, Division
7. Addition, Subtraktion
8. Vergleichsoperationen
9. Logische Operationen

2.2.1.6 Hexadezimal-(Sedezimal-)Zahlen

Natürliche Zahlen im Bereich zwischen 0 und 65535 können auch in sedezimaler Form eingegeben werden.

Der Sedezimalzahl wird dabei ein &-Zeichen vorangestellt. Zur hexadezimalen Darstellung einer Zahl dienen die Ziffern zwischen 0 bis 9 und die Buchstaben A bis F.

Beispiel:

Ausgabe

Eingabe

Ausgabe

Die Ausgabe von Zahlen in Sedezimalform erfolgt mit der Tastenfolge:

Dezimalzahl

Beispiel:

123

Hinweise:

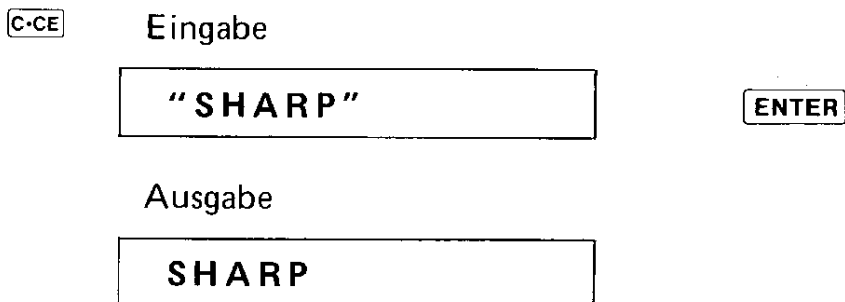
- In folgenden Fällen läßt sich eine Umwandlung nicht durchführen:
 - Wenn die Zahl eine Dezimalstelle hat, die bei Fließkommabetrieb nicht 0 ist.
 - Wenn eine Zahl in wissenschaftlicher Schreibweise dargestellt ist.
- Die HEX-Funktion dient nur zur Ausgabe/Anzeige einer Dezimalzahl in sedezimaler Form. Das Ergebnis der Umwandlung kann nicht mehr weiterverarbeitet werden.

2.2.1.7 Textausdrücke

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

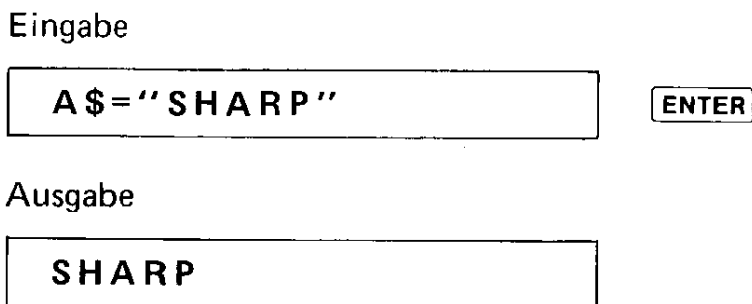
Man unterscheidet Textkonstanten, Textvariablen, Textfunktionen und zusammengesetzte Texte. Eine Textkonstante ist eine beliebige Zeichenfolge, die durch Anführungszeichen begrenzt ist, wobei die Anführungszeichen nicht Bestandteil der Textkonstante sind.

Beispiel:



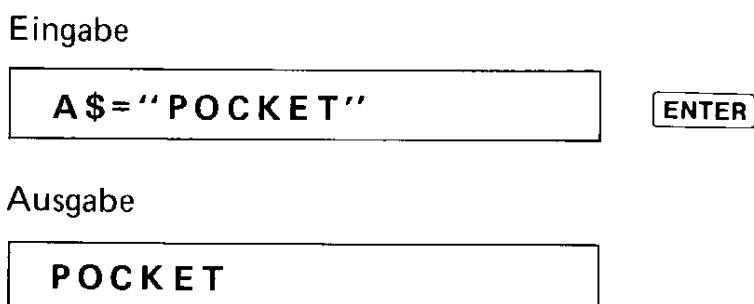
Textvariablen können bis zu sieben Zeichen enthalten. Über die DIM-Anweisung lassen sich Textvariablen bis zu 80 Zeichen erzeugen.

Beispiel:



Texte können mit Hilfe des '+'-Zeichens aneinandergesetzt werden.

Beispiel:



Eingabe

A\$+"_COMPUTER"

ENTER

Ausgabe

POCKET COMPUTER

Hinweis:

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

2.2.1.8 Logische Vergleichsausdrücke

Der Computer 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

C-CE

2 < 1

ENTER

Ausgabe

0.

b) Eingabe

CCE

 $1 \leq 2$

ENTER

Ausgabe

1.

c) Eingabe

CCE

 $\text{SIN } 1 < .7$

ENTER

Ausgabe

0.

(im RAD-Mode)

1.

(im DEG- und GRAD-Mode)

d) Eingabe

CCE

 $\text{"WERNER"} > \text{"ANTON"}$

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 bis zu 2 Zeichen zusammen, wobei das erste Zeichen ein Buchstabe sein muß und das zweite Zeichen eine Ziffer sein kann. Nicht erlaubt sind Sonderzeichen sowie Buchstabenkombinationen, die ein BASIC-Schlüsselwort oder eine Funktion ergeben, z.B. ON, TO, LN, PI etc.

Elemente von Vektoren oder Matrixen werden von ein bzw. zwei in Klammern gesetzte und durch ein Komma getrennte Indexwerte bezeichnet. Indexwerte können auch durch Variablen dargestellt werden.

Beispiele:

A
 A(15)
 A(1)
 A(Z)
 AB
 A1
 XY(2,4)
 XZ(A,B)

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

ASC"A" numerische Funktion; Ergebnis ist die numerische Konstante 65, die den ASCII-Code der Textkonstante "A" darstellt.

CHR\$65 Textfunktion; Ergebnis ist die Textkonstante "A".

MID\$("SHARP",2,2) Textfunktion; Ergebnis ist die Textkonstante "HA".

Hinweis:

Funktionsoperatoren haben eine höhere Priorität als andere Operatoren.

Beispiel:

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

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

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

2.2.2.6 Numerischer Ausdruck

Ein numerischer Ausdruck setzt sich aus einer numerischen Konstanten, Variablen, numerischen Funktion oder deren Verknüpfung durch die arithmetischen Operatoren +, −, *, /, ^, AND, OR und NOT 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 Textausdruck

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

Beispiele:

"A" + STR\$ A

A\$ + "PC"

2.2.2.8 Logische Vergleichsausdrücke

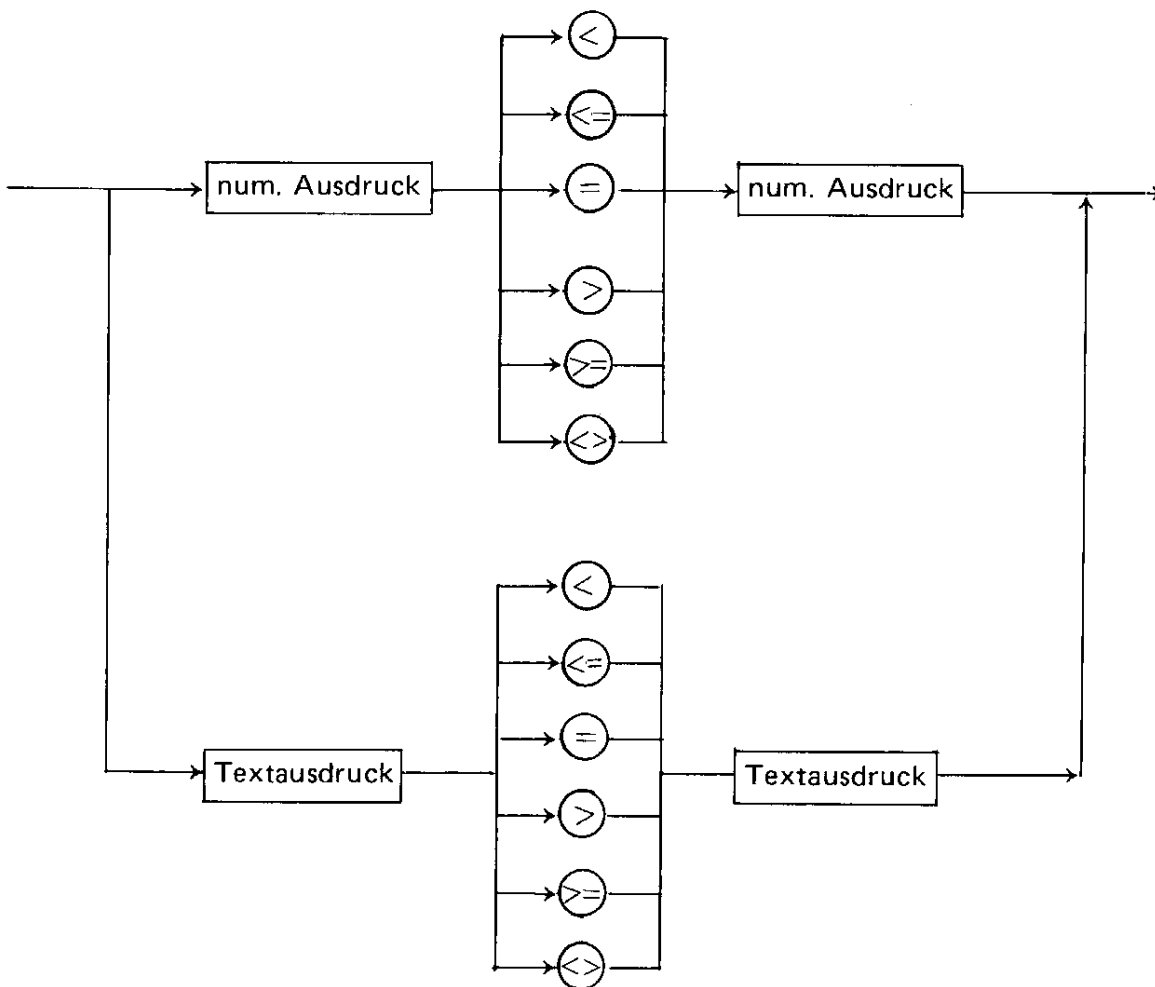
Ein logischer Vergleichsausdruck ist der Vergleich zweier Ausdrücke (numerisch oder Text) durch die Operatoren:

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

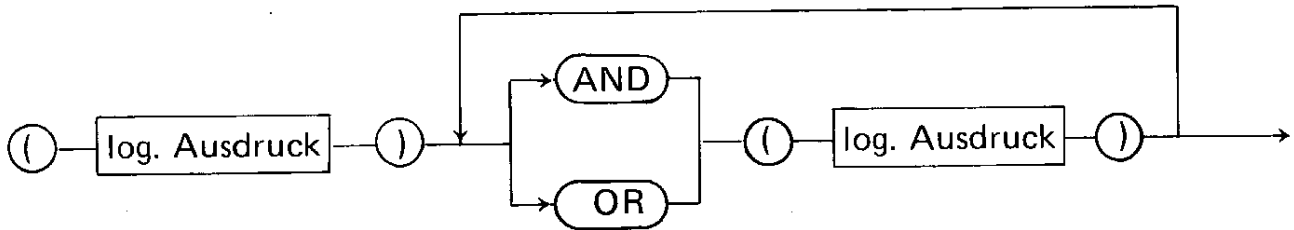
und die Verknüpfung solcher Vergleiche mit den Booleschen Operatoren AND, OR und NOT.

Operation	Mathematisches Symbol	BASIC
Negation	—	NOT
UND-Funktion	∧	AND
ODER-Funktion	∨	OR

Syntax:



Logische Vergleichsausdrücke lassen sich wie folgt verknüpfen:



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; diese richtet sich nach dem ASCII-Code (siehe Anhang).

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
$1 \geq 2$	0
$1 < 2 < 3$	1
"ANTON" < "WERNER"	1
"ANTON" > "ANTON 1"	0 ("1" ist größer als ASCII-Null)
$(1 < 2) \text{ AND } (3 < 4)$	1
$(\text{"ANTON"} < \text{"WERNER"}) \text{ OR } (\text{"ANTON"} = \text{"ANTON 1"})$	1

Logische Ausdrücke

Logische Ausdrücke sind Relationsausdrücke, die Operatoren AND, OR, XOR und NOT. AND, OR und XOR verwenden und dienen zur Verbindung von zwei Relationausdrücken; die Werte der kombinierten Ausdrücke sind in der folgenden Tabelle aufgeführt.

A AND B

		Wert von A	
		Wahr	Falsch
Wert von B	Wahr	Wahr	Falsch
	Falsch	Falsch	Falsch

A OR B

		Wert von A	
		Wahr	Falsch
Wert von B	Wahr	Wahr	Wahr
	Falsch	Wahr	Falsch

A XOR B

		Wert von A	
		Wahr	Falsch
Wert von B	Wahr	Falsch	Wahr
	Falsch	Wahr	Falsch

Die XOR Anweisung kann in einem Ausdruck nicht in Kombination mit der AND oder OR Anweisung verwendet werden. Um den Ausdruck $D=(A \text{ XOR } B) \text{ AND } C$ z. B. auszuführen, wird der Ausdruck für die Ausführung in zwei Abschnitte unterteilt: $D=A \text{ XOR } B$ und $D=D \text{ AND } C$.

Achtung:

Der Wert von A und B muß 0 (falsch) oder 1 (wahr) betragen.

Dezimalzahlen können wie folgt in Binärnotationen von 16 Bit aufgezeichnet werden.

Dezimalzahlen	16-Bit-Binärnotationen
32767	0111111111111111
:	:
3	0000000000000011
2	0000000000000010
1	0000000000000001
0	0000000000000000
-1	1111111111111111
-2	1111111111111110
-3	1111111111111101
:	:
-32768	1000000000000000

Die Negation (NOT) einer Binärnummer 0000000000000001 wird wie folgt vorgenommen:

NOT	0000000000000001
(Negation)–	1111111111111110

1 wird zu 0 invertiert und 0 zu 1. Dies nennt sich "Notation (NOT-Betrieb)". Wenn 1 und NOT 1 addiert werden, ergibt sich dann folgendes Resultat:

0000000000000001 (1)	
+) 1111111111111110 (NOT 1)	Einerkomplement
1111111111111111 (-1)	Zweierkomplement

Alle Bits werden deshalb 1. Gemäß der obigen Zahlenliste werden die Bits -1 in Dezimalnotationen, d.h.,

$$1 + \text{NOT } 1 = -1$$

Das Verhältnis zwischen dem numerischen Wert X und dessen negiertem (oder invertiertem) Wert NOT X ist:

$$X + \text{NOT } X = -1$$

Dieses Ergebnis ist eine Gleichung von NOT X = -X-1, bzw.

$$\text{NOT } X = -(X + 1)$$

Aus dieser Gleichung ergeben sich folgende Resultate:

$$\begin{aligned} \text{NOT } 0 &= -1 \\ \text{NOT } -1 &= 0 \\ \text{NOT } -2 &= 1 \end{aligned}$$

Mit diesen Operatoren können mehr als zwei relationale Ausdrücke kombiniert werden. Um den beabsichtigten Vergleich deutlich zu machen, empfiehlt sich die Verwendung von Klammern.

$$\begin{aligned} & (A < 9) \text{ AND } (B > 5) \\ & (A \geq 10) \text{ AND NOT } (A > 20) \\ & (C = 5) \text{ OR } (C = 46) \text{ OR } (C = 7) \\ & (X \geq 50) \text{ XOR } (X < 70) \end{aligned}$$

Der COMPUTER implementiert logische Operatoren als "bitweise" logische Funktionen an 16-Bit-Quantitäten. (Siehe Anmerkung bezüglich relationaler Ausdrücke und Wahr und Falsch.) Bei normalem Betrieb ist dies unwesentlich, da die einfachen 1 und 0 (Wahr und Falsch), die aus einem relationalen Ausdruck resultieren, nur mit einem Bit arbeiten. Bei Anwendung eines booleschen Operators auf einen anderen Wert als 0 oder 1 funktioniert dies unabhängig an jedem Bit. Wenn zum Beispiel A 17 und B 22 ist, dann ist (A OR B) 23:

$$\begin{array}{r} 17 \text{ OR } 22 \text{ ist } 10001 \text{ .. } 17 \\ \quad \quad \quad \underline{10110 \text{ .. } 22} \quad \text{OR Betrieb} \\ \quad \quad \quad 10111 \text{ .. } 23 \text{ in Dezimalzahl} \end{array}$$

17 und 22 werden zuerst in Binärzahlen umgewandelt. Für jede Ziffer wird, wenn jedes Bit 1 ist, der boolesche Operator 1 gelassen.

Andernfalls wird boolescher Operator 0 gelassen.

Wenn zum Beispiel A 41 und B 27 ist, dann ist (A XOR B) 50:

$$\begin{array}{r} 41 \text{ XOR } 27 \text{ ist } 101001 \text{ .. } 41 \\ \quad \quad \quad \underline{011011 \text{ .. } 27} \quad \text{XOR Betrieb} \\ \quad \quad \quad 110010 \text{ .. } 50 \text{ in Dezimalzahl} \end{array}$$

41 und 27 werden zuerst in Binärzahlen umgewandelt. Für jede Ziffer wird, wenn beide Bits 1 oder 0 sind, der boolesche Operator 0 gelassen.

Für einen geübten Programmierer wird sich dieser Betriebsablauf gelegentlich als sehr nützlich erweisen. Anfänger sollten sich an klare und einfache Wahr-oder Falsch-Vergleichsausdrücken halten.

2.2.3 VARIABLEN

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 Computer stehen folgende zwei verschiedene Speicherbereiche für Variablen zur Verfügung.

- 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 Variablen

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 Variablen

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 97 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 1243 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 Variable 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 Elementanzahl automatisch auf den Wert begrenzt, den er vor der Dimensionierung eines weiteren Vektors bzw. einer Matrix hatte.
6. Bei der ersten Verwendung von Indizes größer als A(27) oder A\$(27) werden 7 Bytes für den Variablennamen und 8 Bytes für jede Variable verwendet.

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.

7. Die Variablen A(n) bzw. A\$(n) (n = 1 bis 26) können nicht gelöscht werden. Mit den Kommandos NEW 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 Feldvariablen

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 2.3).

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 Feldvariablen

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 Vektorennamen 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, OR, 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, OR, 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 theoretisch im Intervall 0 – 255.

Tatsächlich ist das Intervall jedoch abhängig von der aktuell freien Speicherkapazität des Hauptspeichers.

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

Der Index für das Vektoren- und Matrizenelement 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 138) 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, OR, PI etc.

Die Indizes für die Elemente liegen theoretisch im Intervall 0 – 255.

Tatsächlich ist das Intervall jedoch abhängig von der aktuell freien Speicherkapazität des Hauptspeichers.

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

Beispiele:

B\$(0)

A5\$(0)

Z\$(80)

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

Beispiele:

D\$(2,4)

Z1\$(5,10)

Die Indizes für das Vektoren- und Matrizenelement 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 Textfeldvariablen '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

Hyperbelfunktionen

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 cos⁻¹	Arcuscosinus
AHC	SHIFT arChyp cos⁻¹	Areacosinus
AHS	SHIFT arChyp sin⁻¹	Areasinus
AHT	SHIFT arChyp tan⁻¹	Areatangens
ASN	SHIFT sin⁻¹	Arcussinus
ATN	SHIFT tan⁻¹	Arcustangens
COS	cos	Cosinus
CUR	SHIFT $\sqrt[3]{}$	Kubikwurzel
DEG	-DEG	Sexagesimalumrechnung
DEGREE		Winkleinheit Grad
DMS	SHIFT -DMS	Dezimal → Sexagesimal Umrechnung
EXP	SHIFT e^x	Exponentialumrechnung
FAC	SHIFT n!	Fakultät
GRAD		Winkleinheit Neugrad
HCN	hyp cos	Hyperbelcosinus
HEX/DEC		Hexadezimal ↔ Dezimal Umrechnung
HSN	hyp sin	Hyperbelsinus
HTN	hyp tan	Hyperbeltangens
INT		Ganzzahlfunktion
LN	ln	Natürlicher Logarithmus
LOG	log	Dekadischer Logarithmus
MDF		Aufrundung
MEM		Freier Speicherplatz
PI	SHIFT π	Kreiskonstante PI
POL	SHIFT →rθ	Umwandlung von rechtwinkligen Koordinaten in Polarkoordinaten
RADIAN		Winkleinheit
RANDOM		Zufallsgeneratoranfangswert

RCP	$1/x$	Reziprokwert
REC	SHIFT $\leftrightarrow xy$	Umwandlung von Polarkoordinaten in rechtwinklige Koordinaten
RND		Zufallszahl
ROT	SHIFT $x\sqrt{y}$	Wurzelberechnungen (x-te Wurzel)
SGN		Vorzeichenfunktion
SIN	sin	Sinus
SQU	x^2	Quadratzahl
SQR	$\sqrt{\quad}$	Quadratwurzel
TAN	tan	Tangens
TEN	SHIFT 10^x	Exponentialfunktion 10^x
^	SHIFT \wedge	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 zur Verfügung:

ASC	Ermittelt von einem ASCII-Zeichen den zugehörigen ASCII-Code.
CHR\$	Ermittelt von einem ASCII-Code das zugehörige ASCII-Zeichen.
LEFT\$	Entnimmt einer Zeichenfolge eine spezifizierte Zeichenanzahl von links.
RIGHT\$	Entnimmt einer Zeichenfolge eine spezifizierte Zeichenanzahl von rechts.
MID\$	Entnimmt einer Zeichenfolge eine spezifizierte Zeichenanzahl aus der Mitte.
STR\$	Wandelt den Wert eines numerischen Ausdrucks in eine Zeichenfolge um.
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 Anfang der 60er Jahre am Dartmouth College entwickelt und hat sich bis heute insbesondere bei den Mikrocomputern durchgesetzt.

BASIC ist im Gegensatz zu anderen Programmiersprachen einfach und leicht zu erlernen.

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 \emptyset (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 in der angegebenen Reihenfolge abgearbeitet 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 Teilprobleme 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 Übertragung 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-, GOSUB-, ON GOTO- und ON 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 24 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  -Taste intern abgekürzt. Bei der Eingabe einer sehr langen Programmzeile kann man vorerst nur 79 Zeichen eingeben. Nach Drücken der  -Taste werden die Schlüsselwörter abgekürzt, und man kann zusätzlich noch einige Zeichen eingeben. Die Eingabe muß wieder mit  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 Numerierung 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 eventuell noch im Speicher befindliches Programm kann mit der NEW-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
```

Überprüfen 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. Überschreiben 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   - oder der  -Taste Zeichen in der Zeile löschen oder einfügen. Auch ein Überschreiben 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  (Zeile 10 anzeigen)
```

```
 (Beginn Änderung)
```

```
   (Cursor auf die '5')
```

```
7  ('5' durch '7' ersetzen)
```

2.2.6.6 Löschen und Kopieren einer Zeile

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.

Kopieren einer Zeile

Beim Programmieren kommt es häufig vor, daß mehrere Programmzeilen fast den gleichen Inhalt haben oder daß Programmzeilen im ersten Programmteil auch in einem weiteren Programmteil benutzt werden können. Um sich die Arbeit der Eingaben zu vereinfachen, kann der Inhalt einer bestehenden Programmzeile auf eine oder weitere neue Zeilennummern kopiert werden. Die alte Zeilennummer und ihr Programm bleibt unverändert.

Das Kopieren einer Programmzeile erfolgt im PRO-Modus. Man bringt die zu kopierende Zeile mit "LIST" oder den Tasten **↑** : **↓** zur Anzeige. Mit den Cursor-Tasten wird der Cursor auf die Zeilennummer gebracht. Nun kann die Zeilennummer überschrieben werden, gegebenenfalls unter Zuhilfenahme der Tasten **SHIFT DEL** **◀** oder **SHIFT INS.** **▶**. Nach **ENTER** wird die kopierte Zeile mit der neuen Zeilennummer angezeigt und kann ggf. editiert werden.

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 erkannt und gemeldet. Die Fehlermeldung wird am Display in folgender Form angezeigt:

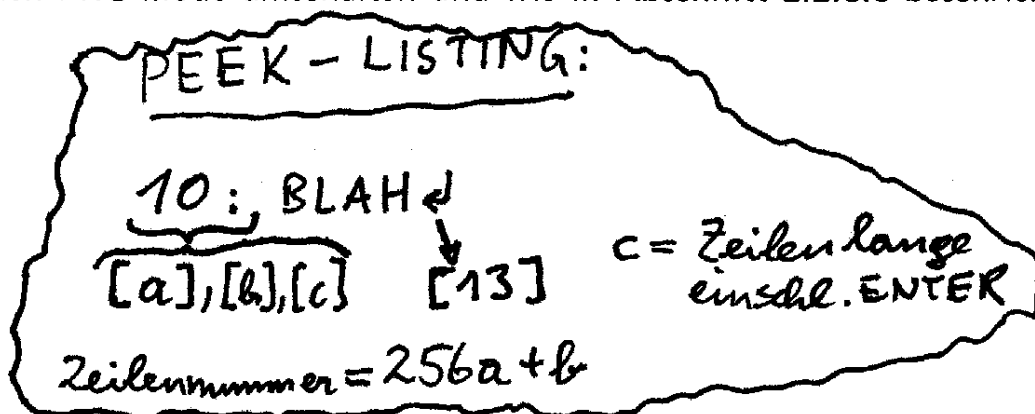
ERROR Fehlercode IN Zeilennummer

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

Gleichzeitig mit dem Fehlercode 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 **[C-CE]**-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 dieses Programms 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.

2.3 BASIC REFERENZTEIL

Das folgende Kapitel ist in drei Teile untergliedert:

2.3.1 KOMMANDOS Instruktionen an den Rechner, die außerhalb eines Programms gegeben werden, um Arbeitsbedingungen, Zusatzschaltungen und Programmkontrollen zu bestimmen.

2.3.2 BEFEHLE Kommando- und Befehlswörter, die gebraucht werden, um ein Programm zu erstellen.

2.3.3 FUNKTIONEN Spezielle BASIC-Operatoren, um Variable umzuwandeln.

Kommandos und verbale Befehle sind innerhalb jeder Kategorie in den betreffenden Abschnitten alphabetisch geordnet. Jeder Eintrag ist auf einer getrennten Seite, um das Auffinden zu erleichtern. Funktionen sind in drei Kategorien unterteilt und innerhalb jeder Kategorie alphabetisch geordnet. Der Inhalt von jedem Abschnitt ist auf den folgenden drei Seiten aufgeführt, damit Sie schnell die jeweilige Kategorie identifizieren und nachschlagen können.

(CHR\$) KOMMANDOS

Programmkontrolle

Variablenkontrolle

8 CONT
 9 DELETE ???
 18 GOTO*
 177 NEW
 176 RENUM
 176 RUN

207 CLEAR*
 203 DIM*
 175 MEM*

Cassettenkontrolle

Winkelmodus

3 CLOAD
 CLOAD?
 2 CSAVE
 7 OPEN
 9 SAVE
 Debug-Kontrolle
 INPUT#*
 184 MERGE
 PRINT#*
 188 CLOSE
 190 LOAD

193 DEGREE*
 195 GRAD*
 194 RADIAN*

Andere

180 LIST
 181 LLIST
 200 TROFF*
 199 TRON*

196 BEEP*
 128 MDF*
 179 PASS
 192 RANDOM*
 202 USING*
 197 WAIT*

* Diese Kommandos sind auch BASIC-Befehle. Ihre Wirkung als Kommando ist indentisch mit ihrer Wirkung als Befehl. Sie sind beschrieben in dem Abschnitt BEFEHLE.

(CHR\$) BEFEHLE

Kontrollen und Verzweigungen

229 CHAIN
 216 END 208
 213 FOR ... TO ... STEP 209
 224 GOSUB
 198 GOTO
 212 IF ... THEN 210
 217 NEXT
 211 ON ... GOSUB 224
 ON ... GOTO 198
 227 RETURN
 218 STOP

Eingabe und Ausgabe

225 AREAD 219 READ
 182 CSAVE 228 RESTORE
 220 DATA 202 USING
 223 INPUT 197 WAIT
 INPUT#
 226 LPRINT
 221 PAUSE
 222 PRINT
 PRINT#

Zuweisung und Deklarationen

201 CLEAR
 203 DIM
 214 LET

Andere

196 BEEP 194 RADIANT
 193 DEGREE 192 RANDOM
 195 GRAD 215 REM
 128 MDF 200 TROFF
 199 TRON

(CHR\$) FUNKTIONEN

Pseudovariablen

23 INKEY\$ 174 PI
 25 MEM

String-Funktionen

24 ASC 170 MID\$
 28 CHR\$ 172 RIGHT\$
 29 LEFT\$ 169 STR\$
 26 LEN 165 VAL
 27 PEEK
 05 POKE
 04 CALL

Numerische Funktionen

153 ABS 152 INT
 158 ACS 145 IN
 142 AHC 146 LOG
 141 AHS 130 POL
 143 AHT 135 RCP
 157 ASN 129 REC
 159 ATN 160 RND
 150 COS 131 ROT
 137 CUR 154 SGN
 155 DEG 149 SIN
 156 DMS 148 SQR
 140 EXP 136 SQU
 144 FACT 151 TAN
 139 HCS 154 TEN
 158 HSN 132 DECI
 140 HTN 133 HEX
 161 INT 163 NOT
 162 RND

2.3.1 KOMMANDOS

CLOAD

1. CLOAD
2. CLOAD "dateiname"

Abkürzung: CLO., CLOA.

Vergleiche: CLOAD?, CSAVE, MERGE, PASS

Wirkung:

Mit dem CLOAD-Kommando werden Programme von der Cassette in den Rechner geladen.

Anwendung:

Die erste Form des CLOAD-Kommandos löscht den Speicher des Rechners und lädt das erste Programm von der eingelegten Cassette.

Die zweite Form des Kommandos löscht den Speicher und lädt das mit "dateiname" angegebene Programm von der Cassette.

Ist der Computer im PROgramm- oder RUN-Modus, so wird das Programm von der Cassette in den Programmspeicher geladen.

Beispiele:

CLOAD Lädt das erste Programm von der Cassette.

CLOAD "PRO3" Sucht auf der eingelegten Cassette das Programm PRO3 und lädt es ein.

Bemerkungen:

1. Wird die angegebene Datei nicht gefunden, sucht der Rechner weiter nach der Datei, selbst wenn die Cassette abgelaufen ist. Unterbrechen Sie in diesem Fall die Suche mit der **BRK**-Taste. Das bezieht sich auch auf die Kommandos MERGE, CHAIN, CLOAD? und INPUT#, die später beschrieben werden.
 2. Tritt während der Ausführung der Kommandos CLOAD oder CHAIN (wird später beschrieben) ein Fehler auf, so ist das im Speicher vorhandene Programm defekt.
- * Während des Ladevorgangs wird an der rechten Seite das Symbol "*" angezeigt. Dieses Symbol verschwindet, wenn das Laden beendet ist. Wird ein Dateiname gesucht, erscheint dieses Symbol erst, wenn der Ladevorgang beginnt.

CLOAD?

1. CLOAD?
2. CLOAD? "dateiname"

Abkürzung: CLO.?, CLOA.?

Vergleiche: CLOAD, CSAVE, MERGE, PASS

Wirkung:

Mit dem Kommando CLOAD? vergleichen Sie das Programm im Speicher des Rechners mit einem auf Cassette gespeicherten Programm.

Anwendung:

Um zu prüfen, ob ein Programm richtig gespeichert wurde, spulen Sie die Cassette an den Anfang zurück und geben das Kommando CLOAD? ein.

Die erste Form des CLOAD?-Kommandos vergleicht den Speicherinhalt mit dem ersten auf der Cassette gefundenen Programm.

Die zweite Form des Kommandos CLOAD? sucht nach dem mit "dateiname" angegebenen Programm und vergleicht es dann mit dem Speicherinhalt.

Beispiele:

CLOAD? Vergleicht den Speicherinhalt mit dem ersten auf der Cassette gefundenen Programm.

CLOAD? "PRO3" Sucht auf der eingelegten Cassette nach dem Programm PRO3 und vergleicht es dann mit dem Speicherinhalt.

* Das Symbol "*" erscheint ganz rechts auf dem Display, während das Programm überprüft wird. Am Ende des Prüfungsvorgangs verschwindet das Sternchen, und es erscheint das Bereitschaftssymbol.

CONT

1. CONT

Abkürzung: C., CO., CON.

Vergleiche: RUN, STOP-Befehl

Wirkung:

Mit dem CONT-Kommando setzen Sie die Ausführung eines unterbrochenen Programms fort.

Anwendung:

Wurde die Ausführung eines Programms mit dem Befehl STOP oder mit der Taste **BRK** unterbrochen, so kann es mit dem Kommando CONT fortgesetzt werden.

CONT können Sie auch benutzen, wenn die Ausführung des Programms zeitweise beispielsweise durch Kommandos wie PRINT unterbrochen wurde.

Beispiel:

CONT Setzt die Ausführung eines unterbrochenen Programms fort.

CSAVE

1. CSAVE
2. CSAVE "dateiname"
3. CSAVE, "passwort"
4. CSAVE "dateiname", "passwort"

Abkürzung: CS., CSA., CSAV.

Vergleiche: CLOAD, CLOAD?, MERGE, PASS

Wirkung:

Mit diesem Kommando wird ein Programm aus dem Speicher des Rechners auf der Cassette abgespeichert.

Anwendung:

Die erste Form des Kommandos CSAVE speichert das Programm im Speicher des Rechners ohne speziellen Dateinamen auf der Cassette ab.

Die zweite Form des Kommandos weist dem Programm vor Abspeicherung den angegebenen Dateinamen zu und speichert es unter diesem Namen auf der Cassette ab.

Die dritte Form des Kommandos speichert das Programm ohne Dateinamen, aber mit einem Passwort auf der Cassette ab. Programme, die mit einem Passwort geschützt sind, können von jedem geladen und ausgeführt werden. Gelistet oder verändert werden kann es aber nur dann, wenn das korrekte Passwort eingegeben wird (näheres unter der Beschreibung des PASS-Kommandos).

Mit der vierten Form des CSAVE-Kommandos wird dem Programm sowohl ein Dateiname als auch ein Passwort zugewiesen.

Beispiel:

CSAVE "PRO3", "GEHEIM" Speichert das präsenste Programm unter dem Dateinamen PRO3 mit dem Passwort GEHEIM auf Cassette ab.

DELETE**1. DELETE [Nr. der ersten Zeile][, [Nr. der letzten Zeile]]**

Abkürzung: DEL., DELE., DELET.

Vergleiche: NEW, PASS

Wirkung:

Der DELETE-Kommand dient zum Löschen einer bzw. mehrerer Programmzeilen.

Dieser Befehl wird zur manuellen Operation in der Programmierbetriebsart verwendet.

Anwendung:

Zum Löschen mehrerer chronologisch aufeinanderfolgender Programmzeilen wird die Nr. der ersten und letzten zu löschenden Zeile angegeben.

Wenn eine der beiden Zeilennummern nicht auffindbar ist, tritt ein Fehler auf.

Wenn nur die Nr. der ersten Zeile angegeben wird, erfolgt Löschung dieser einen Zeile.

Wenn die Nr. der ersten Zeile mit nachfolgendem Komma (,) angegeben wird, wird diese Zeile zusammen mit allen nachfolgenden Zeilen gelöscht.

Wenn das Komma (,) mit nachfolgender Nr. der letzten Zeile angegeben wird, werden alle Zeilen ab Programmanfang einschließlich der angegebenen Zeilennummer gelöscht.

Wenn beide Zeilennummern ausgelassen werden, tritt ein Fehler auf.

Bei mit MERGE geladenen Programmen wirkt DELETE nur auf das zuletzt geladene Programm.

Wenn das Programm durch ein Kennwort geschützt ist, wird das DELETE-Kommando ignoriert.

Beispiel:

DELETE 100, Löscht Zeile 100 und alle nachfolgenden Zahlen.

GOTO

1. GOTO ausdruck

Abkürzung: G., GO., GOT.

Vergleiche: RUN

Wirkung:

Mit dem GOTO-Kommando wird die Ausführung eines Programms begonnen.

Anwendung:

Das GOTO-Kommando kann genauso wie das RUN-Kommando benutzt werden. Die Ausführung eines Programms wird von der durch ausdruck angegebenen Zeilennummer gestartet.

GOTO unterscheidet sich in fünf Punkten vom RUN-Kommando:

1. Der Wert des WAIT-Intervalles wird nicht zurückgesetzt.
2. Wurde die Anzeige durch einen USING-Befehl formatiert, wird sie nicht gelöscht.
3. Werte von Variablen und Feldern (Arrays) bleiben erhalten.
4. PRINT=LPRINT-Status wird nicht zurückgesetzt.
5. Der READ-Zeiger wird nicht zurückgesetzt.

Die Ausführung eines Programms mit GOTO entspricht der Ausführung mit der **DEF**-Taste.

Beispiel:

GOTO 100 Die Ausführung eines Programmes wird mit Zeile 100 gestartet.

LIST

1. LIST
2. LIST zeilennummer
3. LIST "label"

Abkürzung: L., LI., LIS.

Vergleiche: LLIST

Wirkung:

Mit dem LIST-Kommando wird ein Programm zur Anzeige gebracht.

Anwendung:

Das LIST-Kommando kann nur im PROgramm-Modus benutzt werden.

- * Mit LIST wird die erste Programmzeile angezeigt.
- * Mit LIST zeilennummer wird das Programm mit der angegebenen Zeilennummer angezeigt. Gibt es die angegebene Zeilennummer nicht, so wird das Programm mit der nächsthöheren Zeilennummer angezeigt.
- * Mit LIST "label" wird das Programm mit der mit diesem Etikett versehenen Zeile angezeigt.
- * Sind mit dem MERGE-Kommando mehrere Programme in den Speicher geladen, so wird mit dem LIST-Kommando die erste Zeilennummer des zuletzt geladenen Programms gelistet.
- * Wird das Kommando in der Form LIST "label" benutzt und das angegebene Label im zuletzt geladenenen Programm nicht gefunden, so sucht das Kommando das Label in den anderen Programmen im Speicher. Die dazugehörige Zeile wird dann gelistet.

Bei einem durch ein Passwort geschützten Programm wird das LIST-Kommando nicht ausgeführt.

Beispiel:

LIST 100 Die Zeile 100 wird angezeigt.

LLIST

1. LLIST
2. LLIST ausdruck
3. LLIST ausdruck 1, ausdrück 2
4. LLIST ausdruck,
5. LLIST, ausdruck
6. LLIST "label"

Abkürzung: LL., LLI., LLIS.

Vergleiche: LIST

Wirkung:

Mit dem LLIST-Kommando kann ein Programm auf dem Drucker ausgedruckt werden.

Anwendung:

Das LLIST-Kommando kann im PROgramm-Modus und im RUN-Modus benutzt werden.

Die erste Form druckt das gesamte Programm in Speicher aus. Die zweite Form druckt nur die mit ausdruck angegebene Zeile aus. Die dritte Form des Kommandos druckt das Programm von der durch den ausdruck1 angegebenen Zeile oder der nächstgrößeren bis zu der durch den ausdruck2 angegebenen Zeile oder der nächstgrößeren aus. Zwischen den beiden angegebenen Zeilen müssen mindestens zwei Zeilen liegen. Die vierte Form listet alle Programme ab der durch ausdruck eingegebenen Zeile bis zum Ende. Die fünfte Form des Kommandos LLIST druckt die Programmzeilen bis zur eingegebenen Zeile (inklusive) aus.

Wenn mit dem MERGE-Kommando mehrere Programme eingeladen wurde, so wirkt das LLIST-Kommando auf das zuletzt geladene Programm. Um ein zuvor geladenes Programm aufzurufen, benutzen Sie das Kommando LLIST "label".

Ein gesetztes Passwort unterdrückt das LLIST-Kommando.

Beispiel:

LLIST 100, 200

Die Programmzeilen zwischen 100 und 200 werden ausgedruckt.

MERGE

1. MERGE
2. MERGE "dateiname"

Abkürzung: MER., MERG.

Vergleiche: CLOAD

Wirkung:

Mit dem MERGE-Kommando können Programme von der Cassette an bereits im Speicher vorhandene angeknüpft werden.

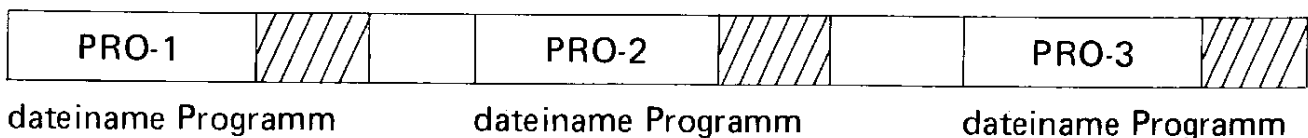
Anwendung:

Mit dem MERGE-Kommando wird das bereits im Speicher geladene Programm aufbewahrt und ein weiteres Programm von der Cassette in den Speicher des Rechners geladen. Auf diese Weise können mehrere Programme in den Computer geladen werden.

Beispiele:

Nehmen wir an, wir haben drei Programme mit den Dateinamen PRO1, PRO2 und PRO3 auf Cassette abgespeichert. Jetzt kann das Programm PRO1 mit dem Kommando CLOAD eingeladen werden, die Programme PRO2 und PRO3 mit dem MERGE-Kommando. Die Abbildung auf dieser Seite verdeutlicht, wie die Programme gespeichert werden.

Cassette



CLOAD "PRO-1"

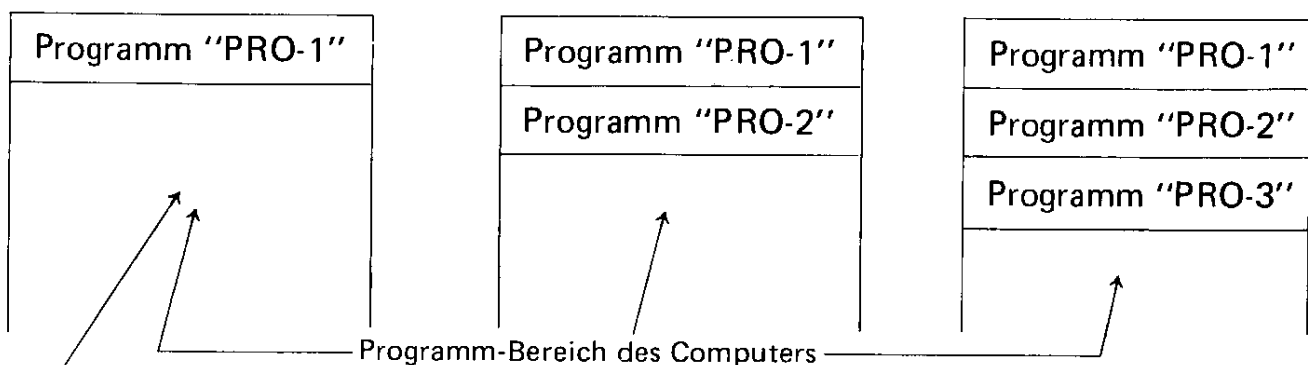
ENTER

MERGE "PRO-2"

ENTER

MERGE "PRO-3"

ENTER



Benutzen Sie das CLOAD-Kommando, um das erste Programm in den Rechner zu laden.

Programme, die mit dem MERGE-Kommando geladen wurden, werden wie im o. g. Beispiel gespeichert.

- * Ist die erste Zeilennummer des mit dem MERGE-Kommando geladenen Programms größer als die letzte Zeilennummer des vorher geladenen Programms, werden die beiden Programme im folgenden wie eins behandelt.
- * Ist die erste Zeilennummer des mit dem MERGE-Kommando geladenen Programms kleiner als die letzte Zeilennummer des vorher geladenen Programms, werden die beiden Programme als zwei verschiedene behandelt.
In dem obigen Beispiel sind die Zeilennummern für die ausgewählten Programme PRO1, PRO2 und PRO3 10–200, 50–150 und 160–300. Die Programme PRO1 und PRO2 werden als zwei verschiedene Programme behandelt. PRO2 und PRO3 werden bei den Zeilennummern 50–300 als ein Programm behandelt.
- * Wenn Sie Programme mit dem MERGE-Kommando laden, kann es zwei oder mehrere Programme mit den gleichen Zeilennummern geben. In diesem Fall beziehen sich die Kommandos RUN oder GOTO in der Form RUN ausdruck bzw. GOTO ausdruck auf das letzte, mit MERGE geladene Programm. Dann können die zuvor geladenen Programme nicht mehr ausgeführt werden.
Geben Sie deshalb immer ein Label am Beginn eines Programms an und lassen Sie es anschließend ausführen.
Beachten Sie, daß in jedem Fall nur das zuletzt mit MERGE geladene Programm editiert werden kann, wenn Sie das MERGE-Kommando geben. Alle zuvor mit MERGE geladenen Programme können nicht editiert werden. Geben Sie deshalb immer das Label eines Programms an, ehe Sie mit dem MERGE-Kommando das nächste Programm laden.

VERKNÜPFUNG PASSWORT-GESCHÜTZTER PROGRAMME

Wollen Sie ein Passwort-geschütztes Programm mit dem MERGE-Kommando laden, so müssen wir zwei Fälle unterscheiden:

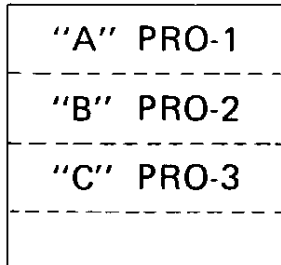
1. Das Programm im Speicher ist geschützt.
2. Das Programm im Speicher ist nicht geschützt.

Im ersten Fall kann kein Passwort-geschütztes Programm mit MERGE geladen werden. Im zweiten Fall werden alle Programme im Speicher durch das neu geladene, geschützte Programm ebenfalls zu Passwort-geschützten Programmen.

Sind im Speicher geschützte Programme und wird ein ungeschütztes Programm mit MERGE geladen, so wird dieses neu geladene Programm ebenfalls zu einem Passwort-geschützten Programm.

AUSFÜHRUNG MIT MERGE GELADENER PROGRAMME

Die Abbildung zeigt die Speicherorganisation, nachdem PRO1 geladen wurde. Anschließend wurde mit dem MERGE-Kommando PRO2 und PRO3 geladen. Wenn Sie RUN oder GOTO (RUN ausdrück, GOTO ausdrück) benutzen, um ein Programm zu starten, so wird immer PRO3 ausgeführt.



Benutzen Sie andererseits RUN "label" oder GOTO "label" oder eine andere definierte Taste, um den Ladevorgang zu starten, wird der Rechner nach dem angegebenen Label im Programm PRO3 suchen.

Wird das Label dort nicht gefunden, so setzt der Rechner seine Suche in dem Programm PRO1 fort. Ist die Suche auch dort nicht erfolgreich, wird PRO2 auf das gesuchte Label hin überprüft. Findet der Rechner das gewünschte Label, wird das Programm von der mit dem Label bezeichneten Zeile abgearbeitet.

Wenn der Rechner auf diese Weise ein Label sucht und das gewünschte Label in Programm PRO3 auftaucht, können die Programme PRO1 und PRO2 nicht ausgeführt werden.

NEW

1. NEW

Wirkung:

Das NEW-Kommando löscht Programme und Daten im Programmspeicher.

Anwendung:

Wird es im PROgramm-Modus benutzt, so werden alle Programme und Daten gelöscht. (Mit Passwort geschützte Programme können nicht gelöscht werden.)

Das NEW-Kommando ist im RUN-Modus nicht definiert. Sie erhalten nach Eingabe des NEW-Kommandos die Fehlermeldung ERROR 9.

Beispiel:

NEW Löscht den entsprechenden Speicherbereich.

PASS**1. PASS “zeichenkette”**

Abkürzung: PA., PAS.

Vergleiche: CSAVE, CLOAD, NEW

Wirkung:

Mit dem Kommando PASS können Passwörter gesetzt und gelöscht werden.

Anwendung:

Mit einem Passwort schützen Sie Ihr Programm vor Einsichtnahme oder Modifikation durch andere Anwender. Ein Passwort besteht aus einer Zeichenkette bis zu sieben Zeichen. Die erlaubten Zeichen für das Passwort sind alle Buchstaben des Alphabets und folgende Sonderzeichen:

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

sowie Pi.

Wurde das Kommando PASS gegeben, so ist das Programm im Speicher geschützt. Das bedeutet, daß das Programm nicht mehr editiert werden kann. Es kann nicht auf Cassette abgespeichert oder mit LIST oder LLIST ausgegeben werden. Weiterhin können weder Zeilen hinzugefügt noch gelöscht werden. Die einzige Möglichkeit, das Passwort auszuschalten, ist ein weiteres PASS-Kommando mit dem gleichen Passwort einzugeben.

Achtung: Wenn Sie ein Passwort mit mehr als 7 Zeichen eingeben, werden nur die ersten 7 Zeichen als gültig erkannt und dem Programm zugewiesen.

Geben Sie **ENTER** ein, nachdem Sie das Passwort bestimmt haben.

Sie erhalten eine Fehlermeldung, wenn Sie nach der Eingabe des Passwortes weitere Zeichen oder Symbole eingeben. In einem solchen Fall akzeptiert der Rechner das Passwort nicht. Es erfolgt eine Fehlermeldung nach Eingabe von PASS "ABCDEFGG": A = 123 **ENTER** .

Beispiel:

PASS "GEHEIM" Setzt für alle Programme im Speicher das Passwort "GEHEIM".

RUN

1. RUN
2. RUN zeilennummer

Abkürzung: R., RU.

Vergleiche: GOTO, MERGE

Wirkung:

Mit dem RUN-Kommando wird das Programm im Speicher gestartet.

Anwendung:

Die erste Form des Kommandos startet das Programm im Speicher mit der Zeile, die die niedrigste Zeilennummer hat.

Die zweite Form startet das Programm mit der angegebenen Zeilennummer. Die Kommandos RUN und GOTO unterscheiden sich in fünf Punkten.

- * Wenn mehrere Programme mit dem MERGE-Kommando geladen werden, wird das zuletzt geladene Programm mit dem Kommando RUN oder mit dem Kommando RUN ausdrück ausgeführt.

Für das Kommando RUN gelten folgende Merkmale:

1. Der Wert des WAIT-Intervalls wird zurückgesetzt.
2. Eine durch einen USING-Befehl erzeugte Formatierung der Anzeige wird gelöscht.
3. Variable und Felder (Arrays) werden gelöscht.
4. PRINT = PRINT-Status wird gesetzt.
5. Der READ-Zeiger wird auf den Anfang der ersten DATA-Zeile gesetzt.

Die Ausführung eines Programms mit GOTO und mit der **DEF**-Taste sind gleich. Alle drei Formen des Programmstarts löschen FOR/NEXT- und GOTO-Gruppen.

Beispiel:

RUN 100 Startet ein Programm mit der Zeile 100.

RENUM**1. RENUM [neue Zeilennummer][,alte Zeilennummer][,Inkrement]**

Abkürzung: REN., RENU.

Wirkung:

Der RENUM-Kommand dient zur Neunummerierung von Programmzeilen. Dieser Befehl ist wirksam für manuellen Betrieb in der PRO-(Programm-)Betriebsart.

Anwendung:

Dieser Befehl nummeriert in den festgelegten inkrementalen Stufen alte Zeilennummern neu. Dieser Befehl ist wirksam für manuellen Betrieb in der PRO-(Programm-)Betriebsart.

Bei Auslassung von Werten der neuen Zeilennummer und Inkrement wird für beide 10 angenommen. Wird die alte Zeilennummer ausgelassen, beginnt die Neunummerierung von der ersten Zeile des Programms. Kann die festgelegte Zeilennummer nicht gefunden werden, tritt eine Fehlermeldung ein.

Beispiel 1:

RENUM

Alle Programmzeilen in Inkrementen von 10 Stufen ab Zeile 10 neu nummerieren.

Beispiel 2

RENUM 100, 50, 10

Alte Zeilennummer 50 in neue Zeilennummer 100 ändern und nachfolgende Zeilennummern in Inkrementen von 10 Schritten neu nummerieren.

Der RENUM-Kommand ändert automatisch alle Zeilennummern-Bezugnahmen, die auf GOTO, GOSUB, IF THEN, ON GOTO, ON GOSUB, RESTORE usw. folgen, um die neuen Nummern wiederzugeben. In diesem Falle tritt jedoch ein Fehler ein, wenn eine der Zeilennummern im Programm eine Variable (z.B. GOTO A) oder ein Ausdruck (z.B. GOTO 2*50) ist. Tritt ein Fehler aufgrund einer solchen falschen Zeilennummer-Bezugnahme ein, kann eine Neunummerierung der betreffenden Zeilennummer nicht durch RENUM ausgeführt werden. In einem solchen Fall eine unkorrekte Zeilennummer als REM-Anweisung vorübergehend neu schreiben und nach Ablauf des RENUM-Befehls korrigieren (eventuell ändern in ON GOTO).

Der RENUM-Befehl kann nicht ausgeführt werden, wenn die Anzahl der neu zu nummerierenden Zeilen 65279 überschreitet oder wenn die Angabe eine Änderung in der Ablauffolge der Programmzeilen verlangt (zum Beispiel: Es wird der Versuch unternommen, RENUM 15, 30 bei drei bestehenden Programmzeilen 10, 20 und 30 auszuführen.)

Der Ablauf von RENUM nimmt bei einem großen Programm etwas Zeit in Anspruch. Wird, während ein Sternchen (*) am äußersten rechten Ende des Displays erscheint, die Taste **BRK** zur Unterbrechung des Programms gedrückt, kehrt das Programm vor beendetem Ablauf von RENUM in den Ausgangszustand zurück. Erscheinen jedoch zwei Sternchen (**) auf dem Display, wird die Unterbrechung durch die **BRK** -Taste ignoriert.

Der Arbeitsbereich (Anzahl der Programmzeilen x 4) von Bytes wird nur bei Ablauf des RENUM-Befehls benutzt. Durch Neunummerieren von Programmzeilen ändern sich die Zeilennummern-Bezugnahmen durch GOTO, GOSUB usw. ebenfalls. Das Originalprogramm kann deshalb eine erhöhte Anzahl an verwendeten Bytes aufweisen. Mit anderen Worten verwendet die neue Zeile GOTO 200 ein Byte mehr als die alte Zeile GOTO 20. Der RENUM-Befehl kann nicht ausgeführt werden, wenn die verbleibende Kapazität des Arbeitsbereichs aufgrund der höheren Byteanzahl zu klein wird. In einem solchen Fall sind Variablen durch den CLEAR-Befehl aus dem Speicher zu löschen, wonach eine Ausführung von RENUM eventuell möglich ist.

(Siehe Anhang A bezüglich der durch RENUM verursachten Fehlermeldungen.)

2.3.2 BEFEHLE

AREAD

1. AREAD variablenname

Abkürzung: A., AR., ARE., AREA.

Vergleiche: Befehl INPUT und die Beschreibung der **DEF**-Taste.

Anwendung:

Der AREAD-Befehl liest den vor dem Programmstart auf der Anzeige stehenden Wert in eine Variable ein.

Wirkung:

Wenn einem Programm ein Label (ein "Name") in Form eines Buchstabens zugewiesen wurde, so daß es mit der **DEF**-Taste gestartet werden kann, kann der AREAD-Befehl benutzt werden, um einen einzelnen Startwert einzugeben, ohne den INPUT-Befehl zu setzen. Der Befehl AREAD muß in der ersten Zeile nach der Zeile mit dem Label gesetzt sein. Taucht er an anderer Stelle im Programm auf, so wird er ignoriert. Es können sowohl Zeichenketten (Strings) als auch numerische Werte eingegeben werden, aber immer nur ein Wert pro Programm.

Erstellen Sie Ihr Programm im PROgramm-Modus. Der AREAD-Befehl kommt dann im RUN-Modus zum Tragen. Geben Sie erst den gewünschten Wert und dann drücken Sie erst **DEF**, gefolgt von dem Label, das Sie gesetzt haben. Wird ein String benutzt, so muß dieser nicht in Anführungsstriche gesetzt werden.

Beispiele:

```
10 "X": AREAD N      Erstellen Sie sich dieses Beispiel-Programm im
20 PRINT N ^ 2      PROgramm-Modus.
30 END
```

Die Eingabe

```
7 DEF X
```

gibt das Quadrat der Zahl "7", also "49" aus.

Wichtig:

1. Steht vor Beginn der Programmausführung das Bereitschaftssymbol (\triangleright) in der Anzeige, so wird die verwendete Variable auf \emptyset (Null) bzw. auf ASCII-Leerzeichen gesetzt.
2. Wird zu Beginn des Programms mit dem PRINT-Befehl etwas angezeigt, so wird folgendes gespeichert:

Folgendes Programm soll ausgeführt werden:

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

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

RUN Modus

```
DEF A → ABC DEFG
```

```
DEF S → DEFG
```

- * Wird PRINT numerischer ausdruck, numerischer ausdruck oder PRINT numerischer string, numerischer string ausgegeben, so wird der letzte angezeigte Ausdruck gespeichert.
- * Wird PRINT numerischer ausdruck; numerischer ausdruck; numerischer ausdruck; . . . ausgegeben, so wird der erste angezeigte Ausdruck gespeichert.
- * Wird PRINT string; string; string; . . . ausgegeben, so wird der letzte angezeigte Ausdruck gespeichert.

BEEP**1. BEEP ausdruck**

Abkürzung: B., BE., BEE.

Wirkung:

Mit dem BEEP-Befehl wird ein Ton erzeugt.

Anwendung:

Der Computer erzeugt auf den BEEP-Befehl hin einen oder mehrere 4kHz-Töne. Die Anzahl der Töne wird mit ausdruck angegeben (positive Zahlen kleiner als 9.999999999 E 99). Nur der ganzzahlige Anteil des angegebenen Ausdrucks wird im BEEP-Befehl verwertet.

BEEP kann auch mit numerischen Variablen arbeiten oder als Kommando benutzt werden. In diesem Falle werden die Töne direkt nach dem **ENTER** erzeugt.

Beispiele:

10	A = 5 : B\$ = "9"	
20	BEEP 3	Erzeugt 3 Töne.
30	BEEP A	Erzeugt 5 Töne.
40	BEEP (A+4)/2	Erzeugt 4 Töne.
50	BEEP B\$	Nicht erlaubt, ERROR 9 wird ausgegeben.
60	BEEP -4	Es wird kein Ton erzeugt, aber auch keine Fehlermeldung ausgegeben.

CHAIN

1. CHAIN
2. CHAIN ausdruck
3. CHAIN "dateiname"
4. CHAIN "dateiname", ausdruck

Abkürzung: CHA., CHAI.

Vergleiche: CLOAD, CSAVE, RUN

Wirkung:

Mit dem CHAIN-Befehl wird ein auf Cassette gespeichertes Programm ausgeführt. Er kann nur in Verbindung mit den Optionen CE-126P und CE-152 benutzt werden.

Anwendung:

Um den CHAIN-Befehl benutzen zu können, muß mindestens ein Programm auf Cassette abgespeichert sein. Dieses Programm wird dann mit dem CHAIN-Befehl geladen und ausgeführt.

Die erste Form des Befehls lädt das erste gefundene Programm von Cassette und beginnt die Ausführung mit der niedrigsten Zeile dieses Programms. Der Effekt ist der gleiche wie mit dem CLOAD- und dem RUN-Kommando im RUN-Modus.

Die zweite Form lädt das erste gefundene Programm und beginnt die Ausführung mit der durch ausdruck angegebenen Zeile.

Die dritte Form von CHAIN sucht das mit "dateiname" angegebene Programm, lädt es und führt es aus, beginnend mit der kleinsten Zeilennummer dieses Programms.

Die vierte Form des CHAIN-Befehls sucht das mit "dateiname" bezeichnete Programm, lädt es von Cassette und startet die Ausführung mit der in ausdruck angegebenen Zeile.

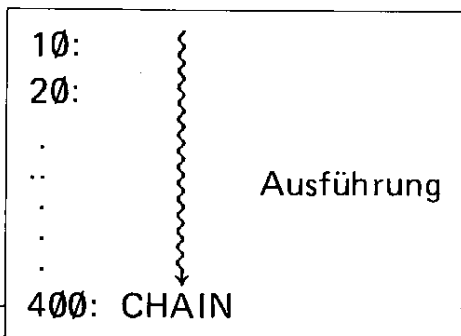
Beispiele:

- | | |
|----------------------|---|
| 10 CHAIN | Lädt das nächste Programm von Cassette und beginnt die Ausführung mit der niedrigsten Zeilennummer. |
| 20 CHAIN "PRO2", 480 | Sucht auf der Cassette das Programm PRO2, lädt es und startet es mit Zeile 480. |

Nehmen Sie z.B. einmal an, daß Sie drei Programme mit dem Namen PRO1, PRO2 und PRO3 auf Cassette gespeichert haben. Alle Programme enden mit einem CHAIN-Befehl.

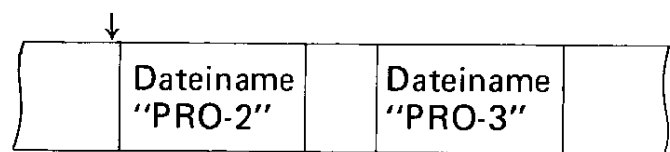
Während der Programmausführung wird der Rechner, wenn er auf einen CHAIN-Befehl trifft, die folgende Sektion in den Speicher rufen und sie ausführen. Auf diese Weise werden schließlich alle Sektionen abgearbeitet.

"PRO-1"

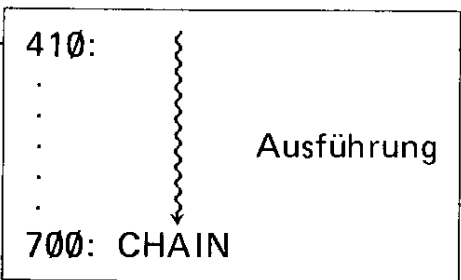


Cassettenrecorder
 ("↓" markiert die Position des
 Cassetten-Lese-Kopfes.)

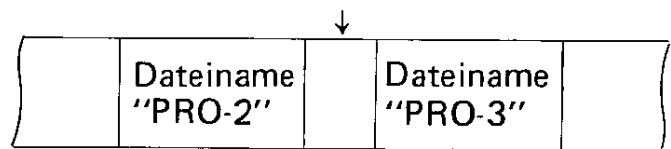
400: CHAIN "PRO-2", 410



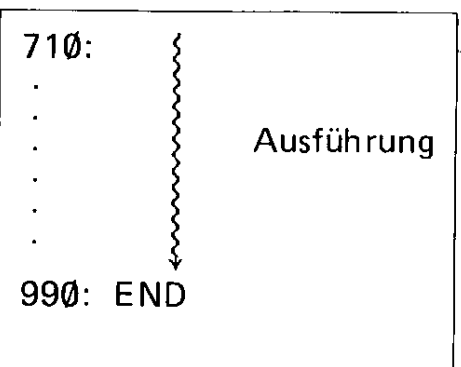
"PRO-2"



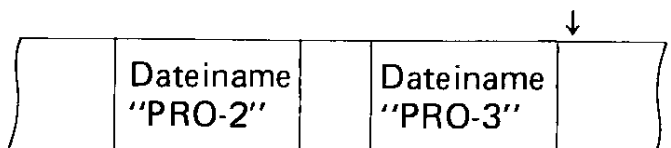
700: CHAIN "PRO-3", 710



"PRO-3"



990: END



Achtung: Programme, die mit MERGE geladen wurden, sollten in der Regel keine CHAIN-Anweisungen enthalten.

CLEAR

1. CLEAR

Abkürzung: CL., CLE., CLEA.

Vergleiche: DIM

Wirkung:

Mit dem CLEAR-Befehl werden alle einfachen und Feldvariablen gelöscht und alle Standardvariablen auf \emptyset (Null) gesetzt.

Anwendung:

Der CLEAR-Befehl setzt Speicherraum frei, der gebraucht wurde, um Werte von einfachen und Feldvariablen zu speichern. Dieser Befehl ist sehr hilfreich, wenn der Speicherbereich eingeschränkt ist und Variablen des ersten Teils eines Programms im zweiten Teil nicht mehr gebraucht werden. Wird CLEAR an den Anfang eines Programms gesetzt, so werden Speicherbereiche, die von den Variablen anderer Programme im Rechner belegt sind, frei gemacht.

Beispiele:

10 A = 5: DIM C(5)

20 CLEAR

Der Vektor C() wird gelöscht und die Standardvariable auf \emptyset (Null) gesetzt.

DATA1. DATA ausdruck, ausdruck, ausdruck, ausdruck . . .

Abkürzung: DA., DAT.

Vergleiche: READ, RESTORE

Wirkung:

Der DATA-Befehl stellt für den READ-Befehl die benötigten Daten bereit.

Anwendung:

Soll ein Feld (Array) mit Daten gefüllt werden, empfiehlt es sich, diese Daten in DATA-Befehlen bereit zu stellen und diese Daten dann mit einer FOR/NEXT-Schleife und einem READ-Befehl in das Feld einzulesen. Der erste READ-Befehl liest den ersten Wert aus der ersten DATA-Zeile, der zweite READ-Befehl den zweiten Wert und so fort. Mit jedem READ-Befehl wird ein Zeiger auf den nächsten Wert gesetzt. Sind die Daten einer DATA Zeile erschöpft, so wird der Zeiger an den Anfang der nächsten DATA-Zeile gesetzt.

DATA-Befehle haben keinen Einfluß auf den Ablauf des Programms, so daß sie an jeder Stelle des Programms eingefügt werden können. Viele Programmierer setzen DATA-Zeilen gerne hinter die Zeile, in der der dazugehörige READ-Befehl steht. Andere setzen alle DATA-Zeilen an den Anfang oder an das Ende eines Programmes. Wenn nötig, kann der Zeiger mit dem RESTORE-Befehl an den Anfang einer bestimmten DATA-Zeile gesetzt werden. So ist es auch möglich, eine DATA-Zeile mehrmals zu benutzen.

Beispiele:

10: DIM B(10)	Initialisierung eines Feldes
20: WAIT 128	
30: FOR I=1 TO 10	Lädt die Werte aus den DATA-Zeilen in das Feld
40: READ B(I)	B(). B(1) wird gleich 10, B(2) gleich 20, B(3)
50: PRINT B(I)	gleich 30 gesetzt, usw.
60: NEXT I	
70: DATA 10,20,30,40,50,60	
80: DATA 70,80,90,100	
90: END	

DEGREE

1. DEGREE

Abkürzung: DE., DEG., DEGR., DEGRE.

Vergleiche: GRAD, RADIAN

Wirkung:

Mit dem DEGREE-Befehl wird der Winkel in der Einheit ALT-GRAD interpretiert.

Anwendung:

Der Computer kann drei Formen der Winkeldarstellung verarbeiten:

1. ALT-GRAD
2. BOGENMASS
3. NEU-GRAD

Diese Formen werden für die Darstellung der Werte der Argumente von SINus-, COSinus- und TANgens-Funktionen und der Ergebnisse der Umkehrfunktionen ArcusSiNus, ArcusCoSinus und ArcusTaNgens benutzt. Mit der DMS- und der DEG-Funktion kann ein Dezimalwinkel in Grad, Minute und Sekunde (sexagesimal) umgeformt werden bzw. umgekehrt.

Beispiel:

1Ø: DEGREE

2Ø: X = ASN 1

3Ø: PRINT X

X hat jetzt den Wert 9Ø, dem Arcussinus von 1.

DIM1. DIM dim liste

<u>dim liste</u>	<u>dimensionierung</u> oder <u>dimensionierung, dim liste</u>
<u>dimensionierung</u>	<u>numerische dim</u> oder <u>string dim</u>
<u>numerische dim</u>	<u>numerischer name (größe)</u>
<u>string dim</u>	<u>string name (größe)</u> oder <u>string name (größe) * länge</u>
<u>numerischer name</u>	<u>erlaubte numerische variable</u>
<u>string name</u>	<u>erlaubte string variable</u>
<u>größe</u>	<u>größe</u> oder <u>größe, grÖße</u>
<u>größe</u>	<u>zahl der elemente</u>
<u>länge</u>	<u>länge eines strings im array</u>

Abkürzung: D., DI.

Wirkung:

Der DIM-Befehl wird benutzt, um Speicher für numerische oder String-Felder frei zu halten.

Anwendung:

Alle Feldvariablen, ausgenommen die Form A(), A\$() und einfache Variablen (AA, B1 etc.), müssen mit einem DIM-Befehl initialisiert werden, um den nötigen Speicher zu reservieren.

Ein Feld kann maximal 2 Dimensionen haben. Die maximale Länge einer Dimension ist 255. Zu der angegebenen Zahl addiert sich ein "nulltes" Element, so daß die Anzahl der Elemente einer Dimension immer um eins größer ist als angegeben, z.B. hat das Feld DIM B(3) die Elemente B(0), B(1), B(2) und B(3). In zweidimensionalen Feldern haben beide Dimensionen ein "Null"-Element.

In String-Feldern kann zusätzlich zur Anzahl der Elemente die Länge der einzelnen Strings vorgegeben werden, z.B. reserviert DIM B\$(3) * 12 Speicherraum für vier String-Elemente mit je 12 Zeichen Länge. Wird die Länge der Strings nicht vorgegeben, so wird eine maximale Länge von 16 Zeichen angenommen.

Bei der Initialisierung eines Feldes werden alle Werte

- a) eines numerischen Feldes auf \emptyset (Null)
- b) eines STRING-Feldes auf ASCII-Leerzeichen gesetzt

Informieren Sie sich hinsichtlich der Initialisierung und Dimensionierung der Felder A() und A\$() in dem Abschnitt, in dem die Variablen erklärt werden.

Beispiele:

- | | |
|-----------------------|---|
| 10: DIM B(10) | Reserviert Speicherraum für ein numerisches Feld mit 11 Elementen. |
| 20: DIM C\$(4,4) * 10 | Reserviert Speicherraum für ein zweidimensionales String-Feld mit 5 Spalten und 5 Reihen. Jeder String soll maximal 10 Zeichen lang sein. |

END

1. END

Abkürzung: E., EN.

Wirkung:

Der END-Befehl signalisiert das Ende eines Programms.

Anwendung:

Sind mehrere Programme geladen, so ist es nötig, den einzelnen Programmen Endmarken zu geben, damit bei der Ausführung eines Programms der Rechner nicht in ein anderes hineinläuft. Der Befehl END setzt diese Endmarke.

Beispiel:

```
10: PRINT "HALLO"  
20: END  
30: PRINT "TSCHUESS"  
40: END
```

Wenn Sie das Kommando RUN 10 geben, so gibt der Rechner das Wort HALLO aus, aber nicht das Wort TSCHUESS. Dieses wird auf das Kommando RUN 30 ausgegeben.

FOR . . . TO . . . STEP

1. FOR numerische variable = ausdruck1 TO ausdruck2
2. FOR numerische variable = ausdruck1 TO ausdruck2
STEP ausdruck3

Abkürzung: F., FO. ; STE.

Vergleiche: NEXT

Wirkung:

Der FOR-Befehl wird in Verbindung mit dem NEXT-Befehl gegeben, um eine bestimmte Operation mehrmals zu wiederholen.

Anwendung:

Der FOR- und der NEXT-Befehl schließen eine Gruppe von Befehlen ein, die wiederholt werden sollen. Wird diese Gruppe das erste Mal ausgeführt, so hat die Schleifenvariable den Wert von ausdruck1.

Erreicht das Programm nun den NEXT-Befehl, so wird die Schleifenvariable um den ausdruck3 erhöht und dann mit dem ausdruck2 verglichen. Ist der Wert der Schleifenvariablen kleiner oder gleich dem ausdruck2, so wird die eingeschlossene Gruppe ein weiteres Mal ausgeführt, beginnend mit dem Befehl hinter FOR. In der ersten Form des Befehls ist die Erhöhung (STEP) gleich 1. In der zweiten Form wird die Erhöhung durch den ausdruck3 vorgegeben. Ist der Wert der Schleifenvariablen größer geworden als ausdruck2, so wird das Programm mit dem Befehl hinter NEXT fortgesetzt. Da der Vergleich am Ende der FOR/NEXT-Schleife durchgeführt wird, werden die eingeschlossenen Befehle mindestens einmal ausgeführt.

Der Wert von ausdruck1, ausdruck2 und ausdruck3 muß zwischen $-9.999999999 \text{ E } 99$ und $9.999999999 \text{ E } 99$ liegen. Wird ausdruck3 gleich 0 gesetzt, so kann das Programm nicht fortgesetzt werden.

Die Schleifenvariable kann in der umschlossenen Gruppe von Befehlen benutzt werden. Mit ihr kann z. B. ein Feld-Index "hochgezählt" werden. Man sollte aber sehr vorsichtig sein, wenn der Wert der Schleifenvariablen verändert wird.

Programme dürfen niemals so geschrieben werden, daß von einem Befehl außerhalb einer FOR/NEXT-Schleife in eine solche hineingesprungen wird. Ein Programm sollte auch nicht aus einer FOR/NEXT-Schleife herauspringen. Sie sollten eine FOR/NEXT-Schleife immer über den NEXT-Befehl verlassen. Der Schleifenvariablenwert muß dazu nur größer als ausdruck2 gesetzt werden.

Eine Gruppe von Befehlen, die von einer FOR/NEXT-Schleife eingefasst werden, kann eine weitere FOR/NEXT-Schleife enthalten. Diese muß dann eine andere Schleifenvariable haben und vollständig in der ersten Schleife liegen. Wenn also in einer FOR/NEXT-Schleife ein FOR-Befehl auftaucht, so muß auch der dazugehörigen NEXT-Befehl in der Schleife liegen. Sie können bis zu fünf FOR/NEXT-Schleifen ineinanderlegen.

Beispiele:

<pre>10: FOR I=1 TO 5 20: PRINT I 30: NEXT I</pre>	<p>Diese Schleife gibt die Zahlen 1, 2, 3, 4 und 5 aus.</p>
<pre>40: FOR N=10 TO 0 STEP -1 50: PRINT N 60: NEXT N</pre>	<p>Diese Schleife zählt rückwärts von 10 nach 0 und gibt die Schleifenvariable aus.</p>
<pre>70: FOR N= 1 TO 10 80: X=1 90: FOR F=1 TO N 100: X=X * F 110: NEXT F 120: PRINT X 130: NEXT N</pre>	<p>Diese Schleife berechnet die Fakultäten der Zahlen von 1 bis 10 und gibt das Ergebnis auf der Anzeige wieder.</p>

Achtung: Programme sollten nicht aus einer FOR/NEXT-Schleife herausspringen. Es könnte hierbei passieren, daß Sie eine Fehlermeldung (ERROR 5) im Programmablauf (bei Programmen, die mehrere FOR/NEXT-Schleifen enthalten) angezeigt bekommen.

GOSUB**1. GOSUB ausdruck**

Abkürzung: GOS., GOSU.

Vergleiche: GOTO, ON ... GOSUB, ON ... GOTO, RETURN

Wirkung:

Mit dem GOSUB-Befehl verzweigt das Programm zu einer BASIC-Unterroutine.

Anwendung:

Wenn eine Gruppe von Befehlen in einem Programm an verschiedenen Stellen ausgeführt werden soll oder diese Befehlsgruppe in verschiedenen Programmen eingesetzt werden soll, so empfiehlt sich die Verwendung der Befehle GOSUB und RETURN.

Die Befehlsgruppe wird so untergebracht, daß das Programm normalerweise nicht in diese Befehle hineinlaufen kann. Eine Möglichkeit wäre, diese Gruppe hinter den END-Befehl zu stellen. An den Stellen des Hauptprogramms, an der diese Gruppe nun ausgeführt werden soll, fügen Sie den GOSUB-Befehl mit der Zeilennummer der Startzeile der Unterroutine als ausdruck ein. Die letzte Zeile der Unterroutine muß einen RETURN-Befehl enthalten. Wird der GOSUB-Befehl nun ausgeführt, so wird das Programm in der Unterroutine fortgesetzt. Erreicht der Computer den RETURN-Befehl so wird die Programmausführung mit dem auf den GOSUB folgenden Befehl im Hauptprogramm fortgesetzt.

In einer Unterroutine kann ein GOSUB vorkommen. Maximal können 10 Unterroutinen ineinandergelegt werden.

Der ausdruck im GOSUB-Befehl darf kein Komma enthalten, z.B. A(2,5) kann nicht benutzt werden. Um von einem bestimmten Punkt mehrere Unterroutinen anspringen zu können, wird der Befehl ON ... GOSUB benutzt. Der ausdruck kann nur eine erlaubte Zeilennummer sein, d. h. der Wert muß zwischen 1 und 65279 inklusive liegen oder Sie bekommen die Fehlermeldung ERROR 4.

Beispiel:

10: GOSUB 100

20: END

100: PRINT "HALLO"

110: RETURN

Das Programm springt von Zeile 10 in die Unterroutine in Zeile 100, gibt das Wort HALLO aus und springt dann in die Zeile 20 des Hauptprogramms zurück.

GOTO1. GOTO ausdruck

Abkürzung: G., GO., GOT.

Vergleiche: GOSUB, ON . . . GOSUB, ON . . . GOTO

Wirkung:

Mit dem GOTO-Befehl verzweigt sich das Programm.

Anwendung:

Der Befehl veranlaßt den Computer, von einer bestimmten Stelle in einem Programm an eine andere, durch den ausdruck bestimmte Stelle des Programms zu springen. Anders als im GOSUB-Befehl "erinnert" sich der Rechner nicht, von wo dieser Sprung ausgeführt wurde.

Der ausdruck des Befehls darf kein Komma, wie z. B. in A(1, 2) enthalten. Soll von einem Punkt die Möglichkeit gegeben werden, an eine von mehreren Stellen zu springen, so muß der ON . . . GOTO-Befehl benutzt werden. Der ausdruck muß eine Zeilennummer sein. Erlaubt sind die Werte von 1 bis 65279. Bei einem anderen Wert bekommen Sie die Fehlermeldung ERROR 4.

Gute Programme laufen von Anfang bis Ende ohne Sprünge, abgesehen von Verzweigungen zu Unterprogrammen. Der prinzipielle Gebrauch des GOTO-Befehls ist deshalb im IF . . . THEN-Befehl zu finden.

Beispiel:

10: INPUT A\$

20: IF A\$="J" THEN GOTO 50

30: PRINT "NEIN"

40: GOTO 60

50: PRINT "JA"

60: END

Wenn Sie ein "J" eingeben, gibt der Rechner das Wort JA aus, sonst das Wort NEIN.

GRAD

1. GRAD

Abkürzung: GR., GRA.

Vergleiche: DEGREE, RADIAN

Wirkung:

Mit dem GRAD-Befehl werden die Winkel in der Einheit NEU-GRAD interpretiert.

Anwendung:

Der Computer hat drei Formen der Winkeldarstellung:

1. ALT-GRAD
2. BOGENMASS
3. NEU-GRAD

Die Argumente der Funktionen SINus, COSinus und TANgens können in dieser Form eingegeben bzw. die Ergebnisse der Umkehrfunktionen ArcusSiNus, ArcusCoSinus und ArcusTaNgens in diesen Formen ausgegeben werden.

Beispiel:

10: GRAD	X hat jetzt den Wert 100, den Gradienten von Arcussinus
20: X=ASN 1	1.
30: PRINT X	

IF . . . THEN

1. IF bedingung THEN befehl
2. IF bedingung befehl

Abkürzung: keine für IF ; T., TH., THE.

Wirkung:

Das IF . . . THEN-Befehlspar wird benutzt, um einen bestimmten Befehl nur dann auszuführen, wenn eine Bedingung erfüllt ist.

Anwendung:

Im normalen Ablauf eines BASIC-Programms werden die Befehle in der Reihenfolge ihres Auftretens abgearbeitet. Das IF . . . THEN-Befehlspar erlaubt es, einen Befehl nur dann auszuführen, wenn eine bestimmte Bedingung erfüllt ist. Ist der Bedingungsteil des IF-Befehls wahr, so wird der Befehl ausgeführt, ist die Bedingung falsch, wird er übersprungen.

Der Bedingungsteil des IF-Befehls kann jeder vergleichende Ausdruck sein. Es ist auch möglich, einen numerischen Ausdruck als Bedingung zu setzen, obgleich der Sinn des Befehls dadurch wenig klar wird. Jeder Ausdruck, der kleiner oder gleich Null ist, setzt die Bedingung falsch. Jeder positive Wert setzt die Bedingung wahr.

Der Befehl, der auf den THEN-Teil des Befehls folgt, kann jeder BASIC-Befehl sein, auch ein weiterer IF . . . THEN-Befehl. Handelt es sich um eine Zuweisung, so muß der LET-Befehl mit dem Wort LET eingeleitet werden.

Die beiden Formen des Befehls sind in der Ausführung gleich, aber die erste Form ist wesentlich klarer.

Beispiel:

10: INPUT "WEITER?"; A\$	Dieses Programm wiederholt die Frage
20: IF A\$="JA" THEN GOTO 10	WEITER?, solange JA eingegeben wird.
30: IF A\$="NEIN" THEN GOTO 60	Es stoppt, wenn NEIN eingegeben wird.
40: PRINT "JA ODER NEIN, BITTE"	Andere Antworten werden nicht be-
50: GOTO 10	achtet.
60: END	

INPUT**1. INPUT eingabe liste****eingabe liste ist eingabe gruppe**

oder

eingabe gruppe, eingabe liste**eingabe gruppe ist var list**

oder

prompt, var liste

oder

prompt; var liste**var liste ist variable**

oder

variable, var liste**prompt ist jede String-Konstante**

Abkürzung: I., IN., INP., INPU.

Vergleiche: INPUT #, READ, PRINT

Wirkung:

Der INPUT-Befehl ermöglicht es, einen oder mehrere Werte über die Tastatur einzugeben.

Anwendung:

Wollen Sie in einem Programm die Grundwerte für jeden Programmlauf ändern, so empfiehlt sich der INPUT-Befehl. Die Basiswerte können dann mit der Tastatur eingegeben werden.

In der einfachsten Form fordert der Befehl Sie mit einem Fragezeichen in der linken Ecke der Anzeige auf, die nötige Eingabe zu machen. Sie geben dann den gewünschten Wert, gefolgt von **ENTER** ein. Dieser Wert wird dann der ersten Variablen der Variablenliste des INPUT-Befehls zugewiesen. Sind in dem Befehl mehrere Variable aufgelistet, so wird der Prozeß so oft wiederholt, bis alle Variablen belegt sind.

Sie können die Eingabe-Aufforderung (= Prompt), also das Fragezeichen, durch einen Prompt-String, das ist eine Bemerkung, die die nötige Eingabe erklären sollte, eigener Wahl ersetzen. Der Ablauf des Zuweisungsprozesses ändert sich nur insofern, als das Fragezeichen durch diesen String ersetzt wird.

Sollte der Befehl einen Prompt-String und die Variablenliste mehr als eine Variable enthalten, so wird der String nur bei der ersten Aufforderung gezeigt. Die folgenden werden wieder mit dem Fragezeichen gekennzeichnet. Beinhaltet der Befehl einen zweiten Prompt-String, so wird dieser vor der Zuweisung der direkt folgenden Variablen angezeigt.

Wenn Sie bei einem INPUT keinen Wert eingeben und direkt die **ENTER** -Taste drücken, so behält die entsprechende Variable ihren alten Wert.

Beispiele:

10: INPUT A

Löscht das Anzeigenfeld und setzt ein Fragezeichen an den Anfang der 1. Zeile.

20: INPUT "A="; A

Gibt "A=" aus und wartet auf eine Eingabe.

30: INPUT "A=", A

Gibt "A=" aus, wartet auf eine Eingabe und zeigt diese dann am Anfang der 1. Zeile an. Der Prompt-String wird also gelöscht.

40: INPUT "X=?"; X; "Y=?"; Y

Gibt erst das Prompt "X=?" aus, wartet auf eine Eingabe, löscht nach dem **ENTER** die Anzeige und gibt dann den Prompt "Y=?" aus.

INPUT #

1. INPUT # var liste
2. INPUT # “dateiname”; var liste

var liste ist variable oder
 variable, var liste

Abkürzung: I. #, IN. #, INP. #, INPU. #

Vergleiche: INPUT, PRINT #, READ

Wirkung:

Mit dem INPUT #-Befehl können Daten von einer Cassette geladen werden.

Anwendung:

Folgende Variable können im INPUT #-Befehl eingesetzt werden:

1. Standardvariable – A, B, C, A(7), D*, A(20)* usw.
2. Einfache Variable – AA, B3, CP\$ usw.
3. Feld-Variable – S(*), HP(*), K\$(*) usw.

1) Übergabe von Daten an Standardvariablen

Um Daten von der Cassette den Standardvariablen zuweisen zu können, müssen die Namen der Variablen im Befehl angegeben werden.

```
INPUT # "DATA1"; A, B, X, Y
```

Der Befehl weist die Daten der Datei DATA1 den Variablen A, B, X und Y in der gegebenen Reihenfolge zu.

Um eine Folge von Standardvariablen und falls definiert, erweiterte Variable (A(27) und weiter) mit Daten von Cassette zu belegen, muß der ersten Variablen ein Stern (*) angehängt werden.

```
INPUT # "D2"; D*
```

Mit diesem Befehl werden die Daten der Cassette den Variablen D bis Z und A(27) und höher zugewiesen.

```
INPUT # A(10)* (ohne DIMensionierung)
```

Dieser Befehl überträgt die Daten der ersten gefundenen Datei von Cassette auf die Variablen A(10) und höher (also J bis Z und A(27) und höher).

Achtung:

- a) Wurde bereits mit einem DIM-Befehl ein Feld namens "A" definiert, kann keine erweiterte Variable der Form A() definiert werden.

- b) Die Übertragung der Daten auf die Standardvariablen und Felder der Form A(), d.h. A(27) und höher, wird solange fortgesetzt, wie Daten gefunden werden, oder bis der Speicher des Rechners voll ist.

2) Datenübertragung auf einfache Variable

Werden im INPUT #-Befehl einfache Variablenamen eingegeben, so werden die Daten der Cassette auf diese Variablen übertragen.

```
INPUT # "DM!"; AB, Y1, XY$
```

Dieser Befehl überträgt die Daten von Cassette auf die Variablen AB, Y1 und XY\$.

Achtung:

- Numerische Daten müssen auf numerische Variable, String-Daten auf String-Variable übertragen werden. Andere Zuweisungen sind nicht möglich.
- Im Programm-Datenspeicherbereich muß Raum für die einfachen Variablen frei gehalten werden, bevor der INPUT #-Befehl ausgeführt werden kann, sonst erhalten Sie eine Fehlermeldung. Die "Platzreservierung" kann mit einer Zuweisung ausgeführt werden.

```
AA=0 ENTER
```

```
BI$="A" ENTER
```

```
INPUT # AA, BI$ ENTER
```

Für die Zuweisung können einfache numerische oder String-Werte benutzt werden.

3) Datenübertragung auf Feldvariable

Um Daten von der Cassette auf die Elemente eines Feldes zu übertragen, muß im Befehl der Name des Feldes in der Form feldname (*) angegeben werden.

```
50: DIM B(5)
```

```
60: INPUT # "DS4"; B( * )
```

Dieser Befehl überträgt die Daten der Datei DS4 auf die Variablen B(0) bis B(5) des Feldes B.

Achtung:

- Numerische Daten müssen auf numerische Feldvariablen gleicher Länge, String-Daten müssen auf String-Feldvariable gleicher Länge übertragen werden. Wird dies nicht beachtet, erhalten Sie eine Fehlermeldung.
- Sie müssen Raum im Programm-Datenspeicherbereich bereitstellen, bevor der INPUT #-Befehl ausgeführt wird oder Sie erhalten eine Fehlermeldung. Benutzen Sie dazu den DIM-Befehl.

WICHTIG

Stimmt die Anzahl der Variablen im INPUT #-Befehl nicht mit der Zahl der Daten auf der Cassette überein, geschieht folgendes:

- * Ist die Zahl der Daten in der Datei auf Cassette größer als die Zahl der Variablen des Befehls, so wird jeder Variablen ein Wert zugewiesen. Die überzähligen Daten werden ignoriert.
- * Ist die Zahl der Daten in der Datei auf Cassette kleiner als die Zahl der Variablen des Befehls, so werden alle Daten der Datei den Variablen zugewiesen und die überzähligen Variablen behalten ihren alten Wert.
Der Rechner wartet aber auf weitere Daten für diese Variablen. Unterbrechen Sie diesen Status mit der **BRK** -Taste.
- * Wird der INPUT #-Befehl ohne Variable eingegeben, so erhalten Sie bei dem Versuch der Ausführung die Fehlermeldung ERROR 1.

LET

1. **LET variable = ausdruck**
2. **variable = ausdruck**

Abkürzung: LE.

Wirkung:

Der LET-Befehl weist Variablen Werte zu.

Anwendung:

Mit dem LET-Befehl weisen Sie der angegebenen Variablen den Wert des Ausdrucks zu. Sie können einer numerischen Variablen nur numerische Werte und einer String-Variablen nur String-Werte zuweisen. Um Werte aus der einen in die andere Form zu übertragen, muß eine der beiden Funktionen STR\$ oder VAL eingesetzt werden.

Sie können das LET in einer Zuweisung weglassen, es sei denn, die Zuweisung folgt auf das THEN eines IF . . . THEN-Befehles. Dies ist der einzige Fall, wo das Wort LET im Befehl vorkommen muß.

Beispiele:

10: I = 10	Weist I den Wert 10 zu.
20: A = 5 * I	Weist A den Wert 50 zu.
30: X\$ = STR\$(A)	Weist X\$ den Wert "50" zu.
40: IF I >= 10 THEN LET Y\$ = X\$ + ".00"	Weist Y\$ den Wert "50.00" zu.

Für Drucker CE-126P

LPRINT

1. LPRINT druckausdr
2. LPRINT druckausdr, druckausdr
3. LPRINT druckliste; druckausdr; . . . ; druckausdr

Abkürzung: LP., LPR., LPRI., LPRIN.

Vergleiche: PRINT, USING

Wirkung:

Mit dem LPRINT-Befehl werden Informationen auf der Option CE-126P ausgegeben.

Anwendung:

- * Im ersten Format des Befehls werden numerische Werte rechtsbündig und alphabetische Zeichen vom linken Rand des Papiers ausgedruckt. Wenn eine Zeile mehr als 24 Zeichen enthält, wird automatisch ein Zeilenvorschub-Befehl ausgeführt.
- * Im zweiten Format werden die 24 Zeichen einer Zeile in zwei Gruppen zu je 12 Zeichen aufgeteilt. Die Daten werden symmetrisch vom Komma ausgedruckt.

Werte, die mehr als 12 Zeichen lang sind, werden auf 12 Zeichen gekürzt. Numerische werden dabei in Dezimalbruch und Zehnerpotenz ausgegeben. Strings werden nur bis zum 12. Zeichen ausgedruckt.

Format 3 druckt die Werte links beginnend aus. Werden mehr als 24 Zeichen eingegeben, so wird der Druck in der nächsten Zeile fortgesetzt. Maximal können 96 Zeichen ausgedruckt werden.

Achtung:

Ein String-Ausdruck darf keinen Basic-Befehl beinhalten.

Beispiele:

```

10: A=10: B=20: X$="ABCD": Y$="XYZ"
20: LPRINT A
30: LPRINT X$
40: LPRINT A, B
50: LPRINT X$, A; B
60: LPRINT X$, Y$
70: LPRINT A * B

```

MDF**1. MDF Ausdruck**

Abkürzung: MD.

Vergleiche: USING

Wirkung:

Die MDF-Funktion dient zur Aufrundung des für einen Ausdruck erhaltenen Werts.

Anwendung:

Mit der MDF-Funktion wird der Wert eines Ausdrucks an der Dezimalstelle aufgerundet, die mit dem USING-Kommand festgelegt wurde.

Diese Funktion arbeitet nur nach der Festlegung der Nachkommastellenzahl durch den USING-Kommand.

Beispiel:

	Anzeige
USING "##.##" MDF (0.5/9)	0.056
10 USING "##.##" 20 A=MDF (5/9) 30 PRINT A	0.556
40 USING 50 PRINT A, 5/9 60 END	0.556 5.55555E-01

NEXT

1. NEXT numerische variable

Abkürzung: N., NE., NEX.

Vergleiche: FOR

Wirkung:

Der NEXT-Befehl kennzeichnet das Ende einer Gruppe von Befehlen, die in einer FOR/NEXT-Schleife wiederholt werden sollen.

Anwendung:

Der NEXT-Befehl tritt immer in Verbindung mit einem FOR-Befehl auf und wird deshalb dort ausführlich erklärt. Die Variable des NEXT muß mit der des dazugehörigen FOR übereinstimmen.

Beispiel:

```
10: FOR I=1 TO 10  
20: PRINT I  
30: NEXT I
```

ON . . . GOSUB1. ON ausdruck GOSUB ausdr liste

ausdr liste ausdruck
 oder
 ausdruck, ausdr liste

Abkürzung: O. ; GOS., GOSU.

Vergleiche: GOSUB, GOTO, ON . . . GOTO

Wirkung:

Der ON . . . GOSUB-Befehl wird benutzt, um eine der gegebenen Unterroutinen abhängig von einem Wert auszuführen.

Anwendung:

Bei der Ausführung des ON . . . GOSUB-Befehls wird der ausdruck zwischen ON und GOSUB auf einen ganzzahligen Wert reduziert. Ist der Wert dieses Ausdrucks 1, so wird die erste der durch ausdr liste angegebenen Unterroutinen nach dem GOSUB ausgeführt. Ist der Wert 2, so wird die zweite ausgeführt usw. Nach dem RETURN aus der Unterroutine wird der Befehl der direkt auf den ON . . . GOSUB-Befehl folgt, ausgeführt.

Wenn der Wert des Ausdrucks kleiner 1 oder größer der Zahl der gegebenen Unterroutinen ist, wird das Programm mit dem auf den ON . . . GOSUB-Befehl folgenden Befehl fortgesetzt.

Achtung: In den Ausdrücken nach dem GOSUB dürfen keine Kommata vorkommen. Der Computer kann Kommata in Ausdrücken nicht von Kommata zwischen Ausdrücken unterscheiden.

Beispiel:

<pre> 10: INPUT A 20: ON A GOSUB 100, 200, 300 30: END 100: PRINT "ERSTE" 110: RETURN 200: PRINT "ZWEITE" 210: RETURN 300: PRINT "DRITTE" 310: RETURN </pre>	<p>Wird in Zeile 10 "1" eingegeben, so gibt der Rechner "ERSTE" aus. Auf "2" gibt er "ZWEITE" und auf "3" "DRITTE" aus. Auf andere Eingaben wird nichts ausgegeben.</p>
--	---

ON . . . GOTO**1. ON ausdruck GOTO ausdr liste**

ausdr liste ausdruck
 oder
 ausdruck, ausdr liste

Abkürzung: O.; G., GO., GOT.

Vergleiche: GOSUB, GOTO, ON . . . GOSUB

Wirkung:

Mit dem Befehl ON . . . GOTO wird es möglich, das Programm abhängig von einem Wert an einer bestimmten Stelle fortzusetzen.

Anwendung:

Bei der Ausführung des Befehls ON . . . GOTO wird der ausdruck zwischen ON und GOTO auf einen ganzzahligen Wert reduziert. Ist der Wert dieser Zahl 1, so wird das Programm an der ersten angegebenen Stelle fortgesetzt. Ist er 2, an der zweiten usw.

Ist der Wert des Ausdrucks kleiner 1 oder größer der Zahl der gegebenen Sprungziele, so wird der auf den ON . . . GOTO-Befehl folgende Befehl ausgeführt.

Achtung: In den Ausdrücken, die auf das GOTO folgen, dürfen keine Kommata vorkommen. Der Computer ist nicht in der Lage, Kommata, die in Ausdrücken auftreten, von Kommata, die zwischen Ausdrücken stehen, zu unterscheiden.

Beispiel:

```

10: INPUT A
20: ON A GOTO 100,200,300
30: GOTO 900
100: PRINT "ERSTE"
110: GOTO 900
200: PRINT "ZWEITE"
210: GOTO 900
300: PRINT "DRITTE"
900: END

```

Die Eingabe '1' läßt den Rechner 'ERSTE', '2' 'ZWEITE' und '3' 'DRITTE' ausgeben.

PAUSE

1. PAUSE druck ausdr
2. PAUSE druck ausdr, druck ausdr
3. PAUSE druck liste; druck liste; . . . ; druck liste

druck liste druck ausdr
 oder
 druck ausdr; druck liste

druck ausdr ausdruck
 oder
 USING format; ausdruck

Das USING format wird unter USING beschrieben.

Abkürzung: PAU., PAUS.

Vergleiche: LPRINT, PRINT, USING, WAIT

Wirkung:

Mit dem PAUSE-Befehl wird eine Ausgabe auf dem Display zeitlich begrenzt wiedergegeben.

Anwendung:

Der PAUSE-Befehl wird benutzt, um kurze Texte, Ergebnisse usw. anzuzeigen. Die Ausführung des Befehls ist die gleiche wie die des PRINT-Befehls. Mit einer Ausnahme: Nach der Ausgabe des Textes oder des Wertes macht der Computer eine Pause von ca. 0,85 Sekunden, bevor der nächste Befehl ausgeführt wird. Ein WAIT-Intervall oder **ENTER** wird nicht gebraucht.

Die erste Form des PAUSE-Befehls gibt einen einzelnen Wert aus. Numerische Werte werden ganz rechts, String-Werte ganz links beginnend ausgegeben.

In Form 2 wird das Anzeigenfeld in Gruppen mit je 12 Zeichen unterteilt. Die Werte werden vom ersten spezifizierten Wert an in aufsteigender Reihenfolge angegeben. Auch hier werden innerhalb des Blocks numerische Werte ganz links und String-Werte ganz rechts beginnend ausgegeben.

- * Die Anzahl der Werte in Form 2 darf höchstens 2 betragen.
- * Ist ein Wert mehr als 12 Spalten (Zeichen) lang, wird
 - a) ein numerischer Wert als Dezimalbruch mit Exponent wiedergegeben,
 - b) ein Stringwert auf seine ersten 12 Zeichen gekürzt.

Form 3 gibt die Werte in direkter Folge von der linken Seite des Bildschirms beginnend wieder.

Achtung: Wenn der eingegebene Wert von Form 3 das Maximum von 24 Spalten überschreitet, wird der überzählige Teil nicht angezeigt.

Beispiele:

10: A=10 : B=20 : X\$="ABCDEF" : Y\$="XYZ"

	ANZEIGE	
20: PAUSE A		10.
30: PAUSE X\$	ABCDEF	
40: PAUSE X\$, B	ABCDEF	20.
50: PAUSE Y\$; X\$	XYZABCDEF	
60: PAUSE A * B		200

PRINT

1. PRINT druck ausdr
2. PRINT druck ausdr, druck ausdr
3. PRINT druck liste
4. PRINT = LPRINT
5. PRINT = PRINT

druck liste druck ausdr
 oder
 druck ausdr; druck liste

druck ausdr ausdruck
 oder
 USING format; ausdruck

Das USING format wird bei USING beschrieben.

Abkürzung: P., PR., PRI., PRIN.

Vergleiche: LPRINT, USING, WAIT

Wirkung:

Mit dem PRINT-Befehl können Ausgaben auf der Anzeige des Computers oder einen Drucker gemacht werden.

Anwendung:

Mit dem PRINT-Befehl werden Text und Ergebnisse ausgegeben. Die erste Form des Befehls gibt einen einzelnen Wert aus. Numerische Werte werden ganz rechts, String-Werte ganz links beginnend wiedergegeben.

In diesen Fall werden die Werte (im Bereich der 12 Spalten) in einem Ausdruck von der rechten Seite des Displays her und die Zeichen von der linken Seite her angezeigt.

- * Die Anzahl der Werte in Form 2 darf höchstens 2 betragen.
- * Ist ein Wert länger als 12 Spalten (Zeichen), so wird
 - a) ein numerischer Wert als Dezimalbruch mit Exponent wiedergegeben,
 - b) ein String-Wert auf die ersten 12 Zeichen gekürzt ausgegeben.

In Form 3 des Befehls werden die Werte direkt hintereinander linksbündig angezeigt.

Beispiele:

10: A=123 : B=456 : X\$="ABCDEF" : Y\$="VWXYZ"

ANZEIGE

20: PRINT X\$, B

ABCDEF

456.

30: PRINT A;B

123.456.

40: PRINT X\$;A

ABCDEF123.

50: PRINT Y\$;B

VWXYZ456.

PRINT #

1. PRINT # var liste
2. PRINT # "dateiname"; var liste

var liste variable
 oder
 variable, var liste

Abkürzung: P.#, PR.#, PRI.#, PRIN.#

Vergleiche: INPUT#, PRINT, READ

Wirkung:

Mit dem PRINT #-Befehl werden Werte auf Cassette abgespeichert.

Anwendung:

Folgende Variablentypen können benutzt werden:

1. Standardvariablen – A, B, X, A(26), C*, A(10)* usw.
2. Einfache Variable – AA, B2, XY\$ usw.
3. Feld-Variable – B(*), CD(*), N\$(*) usw.

1) Abspeichern von Standardvariablen

Um die Werte von Standardvariablen auf Cassette abzuspeichern, müssen die Namen der Variablen im PRINT #-Befehl angegeben werden.

```
PRINT# "DATEI1"; A, B, X, Y
```

Dieser Befehl speichert die Inhalte der Variablen A, B, X und Y in der Datei "DATEI1" auf Cassette ab.

Wenn Sie den Inhalt aller Variablen ab einer bestimmten abspeichern wollen, so geben Sie die Startvariable gefolgt von einem Stern (*) ein.

```
PRINT# "D2"; D*
```

Dieser Befehl speichert die Inhalte der Variablen D bis Z (und A(26) und höher, wenn definiert) in der Datei mit dem Dateinamen "D2" ab.

```
PRINT# E, X$, A(30)*
```

Dieser Befehl speichert die Inhalte der Standardvariablen E und X\$ sowie die Inhalte der erweiterten Variablen A(30) und höher auf Cassette ab.

Achtung: A() kann nicht zur Definition nachfolgender Variablen benutzt werden, wenn A bereits durch einen DIM-Befehl definiert wurde. Ansonsten können die Variablen A(1) bis A(26) im PRINT #-Befehl genauso angegeben werden, wie die Variablen A bis Z bzw. A\$ bis Z\$.

2) Abspeicherung der Inhalte von einfachen Variablen

Die Inhalte von einfachen Variablen (Zwei-Zeichen-Variablen) werden abgespeichert, indem die Namen der Variablen im PRINT #-Befehl angegeben werden.

```
PRINT # "DM1"; AB, Y1, XY$
```

Dieser Befehl speichert die Inhalte der einfachen Variablen AB, Y1 und XY\$ in der Datei mit dem Dateinamen "DM1" ab.

3) Abspeicherung der Inhalte von Feldvariablen

Die Inhalte aller Variablen eines Feldes können auf Cassette abgespeichert werden, indem der Name des Feldes im Befehl angegeben und die Variablen selbst mit einem Stern (*) umschrieben werden.

```
PRINT # "DS2"; X(*), Y(*)
```

Dieser Befehl speichert die Inhalte der Elemente der Felder X (X(0), X(1), . . .) und Y (Y(0), Y(1) . . .) in der Datei mit dem Dateinamen "DS2" ab.

Achtung: Es ist nicht möglich, den Inhalt eines oder einiger Elemente eines Feldes gesondert abzuspeichern.

* Wird der PRINT #-Befehl ohne Variable eingegeben, erhalten Sie die Fehlermeldung ERROR 1.

WICHTIG

Die Lokation von erweiterten (A(26) und höher), einfachen und/oder Feldvariablen muß vor Ausführung des PRINT #-Befehls im Programm/Datenspeicherbereich bereitgestellt werden. Sonst ist die Ausführung des Befehls nicht möglich und Sie erhalten eine Fehlermeldung.

RADIAN

1. RADIAN

Abkürzung: RAD., RADl., RADIA.

Vergleiche: DEGREE, GRAD

Wirkung:

Mit dem RADIAN-Befehl wird der Winkel in der Einheit BOGENMASS interpretiert.

Anwendung:

Der Computer hat drei Möglichkeiten der Darstellung von Winkelwerten:

- ALT-GRAD
- BOGENMASS
- NEU-GRAD

Sie werden für die Argumente der Funktionen SINus, COSinus und TANgens sowie für die Ergebnisse der Umkehrfunktionen ArcusSiNus, ArcusCoSinus und Arcus-TaNgens gebraucht.

Der RADIAN-Befehl wechselt die Darstellungsform auf Radialwert. Die Radialform gibt den Winkel als Bogenmaß A in Abhängigkeit vom Radius wieder. 360 Grad sind z. B. $2 * \text{PI}$, da der Einheitskreis den Umfang 2 mal PI mal Radius hat.

Beispiel:

10: RADIAN

20: X=ASN 1

30: PRINT X

X hat den Wert 1.570796327 (=PI/2), Arcsin von 1.

RANDOM

1. RANDOM

Abkürzung: RA., RAN., RAND., RANDO.

Wirkung:

Der RANDOM-Befehl setzt eine neue Startzahl für den Zufallszahlengenerator.

Anwendung:

Wenn mit der RND-Funktion Zufallszahlen erzeugt werden, so startet der Computer bei einer vorgegebenen Zahl. Der RANDOM-Befehl setzt die Startzahl auf einen neuen, zufällig gewählten Wert.

Die Startzahl für den Zufallszahlengenerator ist jedes Mal, wenn der Computer eingeschaltet wird, die gleiche. Auch die Sequenz der Zufallszahlen wiederholt sich damit, es sei denn, die Basis (oder Start-)zahl wird mit dem RANDOM-Befehl gewechselt. Diese Eigenschaft ist sehr wichtig, ist es doch möglich, den korrekten Ablauf eines Programms zu ermitteln, da auch dann, wenn im Programm RND-Befehle auftreten, das Verhalten des Rechners immer gleich sein muß. Mit dem RANDOM-Befehl verbessern Sie also das Zufallsverhalten Ihres Programms, da die Basiszahl des Zufallsgenerators auch zufällig ermittelt wird.

Beispiel:

```
10: RANDOM
20: X=RND 10
30: GOTO 20
```

Bei Start in Zeile 20 wird die Standardbasis des Zufallszahlengenerators eingesetzt. Bei Start in Zeile 10 wird eine neue Basis ermittelt.

READ**1. READ var liste**

var liste variable
 oder
 variable, var liste

Abkürzung: REA.

Vergleiche: DATA, RESTORE

Wirkung:

Der READ-Befehl wird gebraucht, um Daten aus den DATA-Zeilen herauszulesen und Variablen zuzuweisen.

Anwendung:

Wenn einem Feld Basiswerte zugewiesen werden sollen, so ist es sinnvoll, diese Daten in DATA-Zeilen unterzubringen und sie von dort mit einem READ-Befehl in einer FOR/NEXT-Schleife in das Feld zu übertragen. Wird der erste READ-Befehl ausgeführt, so wird der erste Wert aus den DATA-Zeilen zugewiesen, mit dem zweiten READ-Befehl der zweite Wert usw. Die Variablen der READ-Anweisung und die Daten der DATA-Anweisung müssen in Bezug auf ihre Anzahl und ihren Typ zueinander passen, d. h. die Anzahl der Variablen muß gleich der Anzahl der Daten sein, einer numerischen Variablen kann nur ein numerischer Ausdruck und einer Textvariablen nur ein Textausdruck zugewiesen werden.

Wenn nötig, können dieselben Daten ein zweites Mal gelesen werden. Hierzu wird der RESTORE-Befehl gebraucht.

Beispiel:

10: DIM B(10)	Dimensionierung eines Feldes
20: WAIT 32	
30: FOR I=1 TO 10	Zuweisung der Werte aus den DATA-Zeilen in die
40: READ B(I)	Feldvariablen B(1) bis B(10); B(1) ist 10, B(2) ist
50: PRINT B(I) * 2	20 usw.
60: NEXT I	
70: DATA 10,20,30,40,50,60	
80: DATA 70,80,90,100	
90: END	

REM

1. REM kommentar

Wirkung:

Mit dem REM-Befehl können Kommentare in ein Programm eingefügt werden.

Anwendung:

Es ist sinnvoll, in ein Programm erläuternde Kommentare einzufügen. Es kann sich hierbei um Titel, Autorennamen, Daten der letzten Änderung, Anwendungshinweise usw. handeln.

Solche Kommentare können mit einem REM-Befehl eingefügt werden. Der REM-Befehl hat keinen Einfluß auf den Programmablauf. Er kann überall im Programm eingefügt werden.

Beispiel:

1Ø: REM DIESE ZEILE HAT KEINEN EFFEKT

RESTORE

1. RESTORE
2. RESTORE ausdruck

Abkürzung: RES., REST., RESTO., RESTOR.

Vergleiche: DATA, READ

Wirkung:

Der RESTORE-Befehl ermöglicht es, eine DATA-Zeile mehrmals zu lesen oder die Reihenfolge, in der die DATA-Zeilen gelesen werden sollen, zu verändern.

Anwendung:

Normalerweise werden mit dem READ-Befehl die Daten aus den DATA-Zeilen eine nach dem anderen herausgelesen, in der Reihenfolge, in der sie stehen. Ist der letzte Wert aus einer DATA-Zeile herausgelesen, so wird, wenn vorhanden und abgefragt, der erste Wert aus der nächsten DATA-Zeile genommen.

Die erste Form des RESTORE-Befehls setzt den Zeiger an den Anfang der ersten DATA-Zeile zurück.

Die zweite Form setzt den Zeiger an den Anfang der DATA-Zeile, deren Zeilennummer gleich der im ausdruck angegebenen Zeilennummer ist.

Beispiel:

10: DIM B(10)	Dimensionierung eines Feldes
20: WAIT 32	
30: FOR I=1 TO 10	Setzt den Zeiger an den Anfang der 1. DATA-Zeile.
40: RESTORE	Den Elementen von B werden die Werte zugewiesen.
50: READ B(I)	Alle Elemente haben den Wert 20.
60: PRINT B(I) * I	
70: NEXT I	
80: DATA 20	
90: END	

RETURN

1. RETURN

Abkürzung: RE., RET., RETU., RETUR.

Vergleiche: GOSUB, ON . . . GOSUB

Wirkung:

Mit dem RETURN-Befehl wird eine Unteroutine geschlossen, der Programmablauf wird mit dem Befehl, der auf den GOSUB-Befehl folgt, fortgesetzt.

Anwendung:

Eine Unteroutine kann mehr als ein RETURN enthalten. Erreicht das Programm einen dieser RETURN-Befehle, wird die Unteroutine geschlossen und das Programm mit dem Befehl hinter dem GOSUB bzw. ON . . . GOSUB, von dem der Rechner in die Unteroutine gesprungen ist, fortgesetzt. Wird ein RETURN ohne GOSUB ausgeführt, erhalten Sie die Fehlermeldung ERROR 5.

Beispiel:

10: GOSUB 100

20: END

100: PRINT "HALLO"

110: RETURN

Das Programm druckt "HALLO" aus und springt zurück in Zeile 20.

STOP

1. STOP

Abkürzung: S., ST., STO.

Vergleiche: END; CONT-Kommando

Wirkung:

Mit dem STOP-Befehl wird die Ausführung eines Programms gestoppt.

Anwendung:

Erreicht der Computer in einem Programm einen STOP-Befehl, so wird die Ausführung des Programms abgebrochen und Sie erhalten eine Meldung wie z.B. "BREAK IN 200", wenn das Programm in Zeile 200 gestoppt wurde. Sie können jetzt die Variablen des Programms überprüfen und so die Ausführung desselben kontrollieren. Die Ausführung des Programms kann mit dem CONT-Befehl fortgesetzt werden.

Beispiel:

10: STOP BREAK IN 10 erscheint auf der Anzeige.

TROFF

1. TROFF

Abkürzung: TROF.

Vergleiche: TRON

Wirkung:

Mit dem TROFF-Befehl wird der Trace-Modus ausgeschaltet.

Anwendung:

Die Ausführung des Befehls TROFF hat zu Folge, daß ein Programm wieder auf "normale" Weise ausgeführt wird.

Beispiel:

10: TRON	Bei Ausführung des Programms werden die Zeilen-
20: FOR I=1 TO 3	nummern ausgegeben:
30: NEXT I	10, 20, 30, 30 und 40
40: TROFF	

TRON

1. TRON


Abkürzung: TR., TRO.




Vergleiche: TROFF

Wirkung:

Der TRON-Befehl schaltet den Trace-Modus ein.

Anwendung:

Der Trace-Modus unterstützt die Fehlersuche in Programmen. Wird das Programm gestartet, wird die erste Zeile des Programms abgearbeitet und deren Zeilennummer angezeigt. Durch Drücken der  -Taste wird die nächste Programmzeile abgearbeitet und die Zeilennummer angezeigt usw.

Das Programm der jeweils angezeigten Zeilennummer kann durch Drücken der  -Taste angezeigt werden. Der Trace-Modus bleibt so lange eingeschaltet, bis er mit dem TROFF-Befehl wieder ausgeschaltet wird oder bis Sie die Tastenfolge   eingeben.

Beispiel:

1Ø: TRON

2Ø: FOR I=1 TO 3

3Ø: NEXT I

4Ø: TROFF

Bei Ausführung des Programms werden die Zeilennummern ausgegeben:
1Ø, 2Ø, 3Ø, 3Ø, 3Ø und 4Ø

USING

1. USING
2. USING “spezifikation”

Abkürzung: U., US., USI., USIN.

Vergleiche: LPRINT, PAUSE, PRINT

Weitere Erklärung zu USING finden Sie im Anhang C.

Wirkung:

Mit dem USING-Befehl kann das Anzeigenfeld oder der Druck formatiert werden.

Anwendung:

Der USING-Befehl kann einzeln oder als Erweiterung der Befehle LPRINT, PAUSE oder PRINT eingesetzt werden. Der USING-Befehl erstellt eine Ausgabeformatierung, die für alle folgenden Ausgabebefehle Gültigkeit hat, bis die Formatierung durch einen neuen USING-Befehl geändert wird.

Die Spezifikationen des Befehls bestehen aus einer in Anführungsstrichen (") gesetzten Zeichenkette. Die gültigen Zeichen dafür sind:

- # Anzahl der Stellen für einen numerischen Ausdruck
- Dezimalpunkt
- ^ Wissenschaftliches Format
- & Anzahl der Stellen für einen Textausdruck

Die Spezifikation "#####" formatiert die Ausgabe eines numerischen Ausdrucks mit 3 Ziffern, kein Dezimalpunkt und eine Vorzeichenstelle. Für numerische Ausdrücke muß immer eine Position für das Vorzeichen vorgesehen sein, auch, wenn nur positive Zahlen ausgegeben werden sollen.

Eine Formatspezifikation kann beide Formate enthalten, eins für numerische und eins für Textausdrücke.

Die Spezifikation "###.##&&&" formatiert eine Zahl mit zwei Vorkomma-, zwei Nachkommastellen und eine Stelle für das Vorzeichen sowie eine Zeichenkette mit 3 Zeichen.

Wird wie in Form 1 des Befehls, keine Spezifikation gegeben, so werden alle gesetzten Formate gelöscht und die Standardformatierung tritt wieder in Kraft.

Beispiele:

10: A=125 : X\$="ABCDEF"

ANZEIGE

20: PRINT USING "##.##^"; A

1.25E02

30: PRINT USING "&&&&&&"; X\$

ABCDEF

40: PRINT USING "####&&"; A; X\$

125ABC

(Siehe Anhang C für zusätzliche Informationen über die Verwendung von USING.)

WAIT

1. WAIT
2. WAIT ausdruck

Abkürzung: W., WA., WAI.

Vergleiche: PRINT

Wirkung:

Der WAIT-Befehl spezifiziert die Anzeigedauer einer Ausgabe.

Anwendung:

Bei normaler Ausführung eines Programms wartet der Computer nach einem PRINT-Befehl, bis die **ENTER** -Taste gedrückt wird. Der WAIT-Befehl weist den Computer nun an, eine Ausgabe für eine vorgegebene Zeit anzuzeigen und dann die Ausführung des Programms fortzusetzen, genau wie mit dem PAUSE-Befehl. Der **ausdruck** im WAIT-Befehl setzt die Länge des Intervalls fest. Er kann jeden Wert von 0 bis 65535 haben. Dabei bedeutet die Erhöhung um 1 eine Verlängerung des Intervalls um 1/59 Sekunden. WAIT 0 ist zu schnell, als daß die Ausgabe gelesen werden könnte; WAIT 65535 setzt das Intervall auf ca. 19 Minuten. WAIT ohne Ausdruck setzt den Computer auf seine normale Arbeitsweise zurück.

Beispiel:

10: WAIT 59

Der Rechner wartet nach PRINT etwa 1 Sekunde.

2.3.3 FUNKTIONEN

2.3.3.1 PSEUDOVARIABLE

Pseudovariablen sind eine Gruppe von Funktionen, die ohne Argument wie einfache Variablen behandelt werden.

INKEY\$

1. INKEY\$

INKEY\$ ist eine String-Pseudovariablen, die den Wert der letzten, auf der Tastatur gedrückten Taste zugewiesen bekommt. Die Tasten **ENTER**, **C-CE**, **SHIFT**, **DEF**, **SML**, **↑**-Pfeil, **↓**-Pfeil, **▶**, **▶**-Pfeil, **CAL**, **BASIC** und wissenschaftliche Funktionstasten haben den Wert Null (= leer). Mit INKEY\$ können Eingaben über die Tastatur ohne ein abschließendes **ENTER** gemacht werden.

```

5: WAIT 50
10: A$= INKEY$
20: B=ASC A$
30: IF B=0 THEN GOTO 10
40: IF B=...
```

Die Zeilen ab Zeile 40 können eine Tastatur-Testroutine enthalten (z. B. 40 PRINT A\$). Wenn das Programm gestartet wird, ist der Wert von INKEY\$ = NULL, da die letzte gedrückte Taste **ENTER** war.

- Wenn sich am Anfang des Programms ein INKEY\$ Befehl befindet, kann es vorkommen, daß beim Starten des Programms die Starttaste vom INKEY\$ Befehl gelesen wird. Beispielsweise kann es im folgenden Programm
10 "Z" : Z\$ = INKEY&
vorkommen, daß die Taste **Z** gelesen wird, wenn das Programm durch Drücken der Tasten **DEF Z** gestartet wird.

MEM

1. MEM

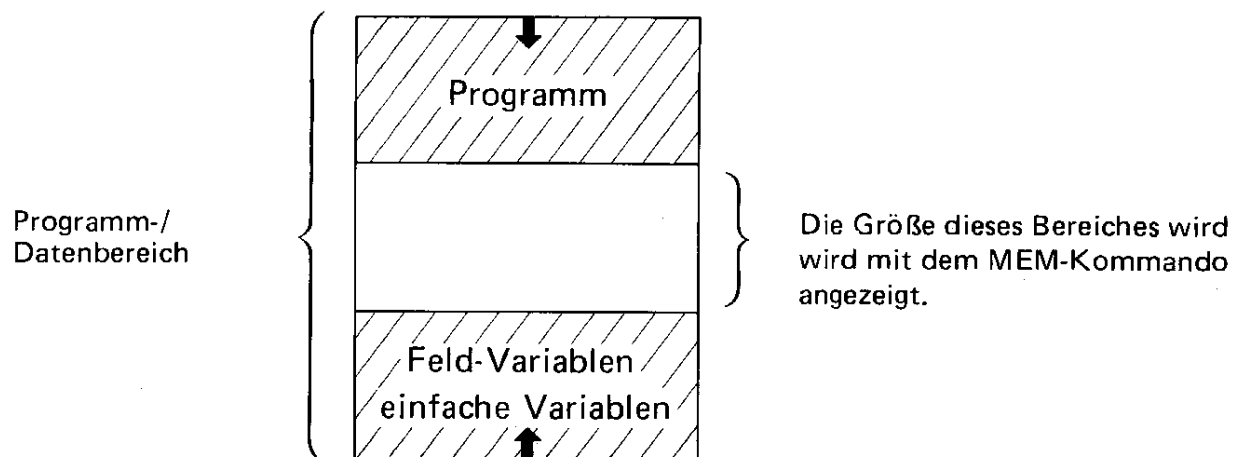
Abkürzung: M.

Wirkung:

Ermittelt die Anzahl der freien Bytes im Programmspeicher.

Anwendung:

Liefert die Anzahl der freien Bytes (der nicht durch ein Programm, Feldvariable oder einfache Variable belegte Bereich) im Programm/Daten-Bereich.



Die Programmgröße (in Bytes) kann durch folgende Eingabe angezeigt werden:

Beispiel:

RUN-Modus

CLEAR

(löscht die einfachen Variablen, die Feld-Variablen usw.)

6878 – MEM (Es wird die Programmgröße
– in Bytes – angezeigt.)

PI**1. PI**

PI ist eine numerische Pseudovariablen, die den Wert π hat. Sie hat die gleiche Funktion wie die spezielle Taste π auf der Tastatur. Wie alle anderen Zahlen wird auch der Wert von PI mit einer Genauigkeit von 10 Stellen wiedergegeben (3.141592654).

2.3.3.2 NUMERISCHE FUNKTIONEN

Numerische Funktionen sind eine Gruppe von mathematischen Funktionen, die aus einem numerischen Argument ein numerisches Ergebnis errechnen. Die Gruppe enthält trigonometrische und logarithmische Funktionen, sowie solche, die ganze Zahlen und Beträge errechnen bzw. Vorzeichen ermitteln. Viele BASIC-Dialekte fordern, daß das Argument der Funktion in Klammern steht. Dieser Computer braucht Klammern nur dann, wenn in einer komplexen Berechnung das Argument der Funktion genau abgegrenzt werden muß.

$\text{LOG } 100 + 100$ wird als

$(\text{LOG } 100) + 100$ und nicht als $\text{LOG } (100 + 100)$

interpretiert.

ABS

1. ABS numerischer ausdrück

Errechnet den Absolut-Betrag des angegebenen Arguments.

$\text{ABS } -10$ ist gleich 10

ACS

1. ACS numerischer ausdrück

Errechnet den Arcuscosinus des Arguments. Arcuscosinus ist die Umkehrfunktion des Cosinus. Das Ergebnis ist also der Winkel zu dem angegebenen Wert. Die Darstellung des Ergebnisses ist abhängig vom Winkelmodus des Computers (DEG, RAD, GRAD).

$\text{ACS } .5$ ist im DEG-Modus gleiche 60

AHC

1. AHC numerischer ausdrück

Errechnet den Areacosinus des angegebenen Arguments.

AHC5 ist gleich 2.29243167 .

AHS1. AHS numerischer ausdruck

Errechnet den Arcasinus des angegebenen Arguments.

AHS6 ist gleich 2.491779853.

AHT1. AHT numerischer ausdruck

Errechnet den Areatangens des angegebenen Arguments.

ASN1. ASN numerischer ausdruck

Errechnet den Arcussinus des angegebenen Argumens. Arcussinus ist die Umkehrfunktion des Sinus. Die Darstellung des Ergebnisses ist abhängig vom Winkelmodus des Computers (DEG, RAD, GRAD).

ASN .5 ist im DEG-Modus gleich 30

ATN1. ATN numerischer ausdruck

Das Ergebnis ist der Arcustangens des Arguments. Arcustangens ist die Umkehrfunktion des Tangens. Die Darstellung des Ergebnisses ist abhängig vom Winkelmodus des Computers (DEG, RAD, GRAD).

ATN 1 ist im DEG-Modus gleich 45.

COS1. COS numerischer ausdruck

Errechnet den Cosinus eines Winkels. Die Darstellung des Ergebnisses ist abhängig vom Winkelmodus des Computers (DEG, RAD, GRAD).

COS 60 ist im DEG-Modus gleich .5

CUR1. CUR numerischer ausdrück

Errechnet die Kubikwurzel des gegebenen Arguments.

CUR8 ist gleich 2.

DEG1. DEG numerischer ausdrück

Das Ergebnis ist die Umwandlung des Winkelarguments (Alt-Grad) als DMS-Darstellung (Grad, Minute, Sekunde) in die Dezimaldarstellung DEG. Im DMS-Format stellt der ganzzahlige Anteil der Zahl die Grandzahl, die ersten beiden Dezimalstellen die Minuten und die dritte und vierte Dezimalstelle die Sekunden dar. Die weiteren Stellen geben die Dezimalsekunden wieder. 55 Grad 10' 44,5'' werden also z.B. als 55.10445 dargestellt. Im DEG-Format ist der ganzzahlige Anteil wieder Grad, aber die Dezimalstellen sind Dezimalgradbruchteile.

DEG 55.10445 ist also gleich 55.17902778.

DMS1. DMS numerischer ausdrück

DMS ist die Umkehrfunktion zur DEG-Funktion (vergleiche DEG).

DMS 55.17902778 ist gleich 55.10445.

EXP1. EXP numerischer ausdrück

Errechnet die Potenzen der Zahl e (= 2.718181828, die Basis des Logarithmus naturalis).

EXP 1 ist gleich 2.718291829

FACT1. FACT numerischer ausdrück

Errechnet die Fakultät des gegebenen Arguments.

FACT5 ist gleich 120.

HCS1. HCS numerischer ausdruck

Errechnet den Hyperbelcosinus des gegebenen Arguments.

HCS5 ist gleich 74.20994852.

HSN1. HSN numerischer ausdruck

Errechnet den Hyperbelsinus des gegebenen Arguments.

HSN4 ist gleich 27.2899172

HTN1. HTN numerischer ausdruck

Errechnet den Hyperbeltangens des gegebenen Arguments.

HTN1 ist gleich 0.761594156.

INT1. INT numerischer ausdruck

Das Ergebnis ist der ganzzahlige Anteil des Arguments.

INT PI ist gleich 3.

LN1. LN numerischer ausdruck

Errechnet den Logarithmus zur Basis e (2.718281828) des Arguments.

LN 100 ist gleich 4.605170186

LOG1. LOG numerischer ausdruck

Errechnet den Logarithmus zur Basis 10 des Arguments.

LOG 100 ist gleich 2.

POL**1. POL (numerischer ausdrück, numerischer ausdrück)**

Bewirkt die Umwandlung von rechtwinkligen Koordinaten in Polarkoordinaten. Der erste numerische Ausdruck ist die Entfernung von der Y-Achse und der zweite die Entfernung von der X-Achse. Die berechneten Werte für die Entfernung und den Winkel der Polarkoordinaten werden den festen Variablen Y bzw. Z zugeordnet. Der Wert des Winkels ist dabei von der Winkeleinheit (DEGREE, GRAD, RADIAN) abhängig.

POL (3, 4) ist gleich (5,53.13010235) in DEGREE.

RCP**1. RCP numerischer ausdrück**

Errechnet den Reziprokwert des gegebenen Arguments.

RCP5 ist gleich 0.2.

REC**1. REC (numerischer ausdrück, numerischer ausdrück)**

Bewirkt die Umwandlung von Polarkoordinaten in rechtwinklige Koordinaten. Der erste numerische Ausdruck ist die Entfernung und der zweite der Winkel, der von der Winkeleinheit (DEGREE, GRAD, RADIAN) abhängig ist. Die berechneten Werte, die Entfernungen von der Y-Achse und von der X-Achse, werden den festen Variablen Y bzw. Z zugeordnet.

REC (7, 50) in DEGREE ist gleich (4.499513268, 5.362311102).

RND**1. RND numerischer ausdrück**

Erzeugt Zufallszahlen. Ist das Argument kleiner 1, aber größer 0, so ist das Ergebnis kleiner 1 und größer oder gleich 0. Ist das Argument eine ganze Zahl und größer oder gleich 1, so ist das Ergebnis kleiner oder gleich dem Argument und größer oder gleich 1. Ist das Argument keine ganze Zahl und größer 1, so ist das Ergebnis kleiner oder gleich der nächstgrößeren ganzen Zahl des Arguments und größer oder gleich 1.

Argument	----- Ergebnis -----	
	Untergrenze	Obergrenze
.5	$0 <$	< 1
2	1	2
2.5	1	3

Die Sequenz ist jedes Mal die gleiche, wenn der Computer eingeschaltet wird, da die Basiszahl immer die gleiche ist. Um diese Basis-Zahl zufällig zu ändern, brauchen Sie den RANDOM-Befehl.

ROT

1. numerischer ausdrück ROT numerischer ausdrück

Errechnet die Wurzel des angegebenen Arguments.

125ROT3 ist gleich 5.

(Beispiel: $\sqrt[3]{125}$ muß als 125 ROT 3 eingegeben werden.)

SGN

1. SGN numerischer ausdrück

Das Ergebnis ist abhängig vom Vorzeichen des Arguments. Ist das Argument positiv, ist das Ergebnis gleich 1; ist das Argument negativ, so ist das Ergebnis gleich -1 ; wenn das Argument gleich 0 ist, ist auch das Ergebnis gleich 0 .

SGN -5 ist gleich -1 .

SIN

1. SIN numerischer ausdrück

Errechnet den Sinus des Winkelarguments. Die Darstellung ist abhängig von dem Winkelmodus des Computers (DEG, RAD, GRAD).

SIN 30 ist im DEG-Modus gleich .5.

SQR1. SQR numerischer ausdruck

Errechnet die Quadratwurzel des Arguments. Das Ergebnis ist identisch mit der Anwendung der speziellen $\sqrt{\square}$ -Taste der Tastatur.

SQR 4 ist gleich 2.

SQU1. SQU numerischer ausdruck

Errechnet das Quadrat des gegebenen Arguments.

SQU3 ist gleich 9.

TAN1. TAN numerischer ausdruck

Errechnet den Tangens des Winkelarguments. Die Darstellung ist abhängig vom Winkelmodus des Computer (DEG, RAD, GRAD).

TAN 45 ist im DEG-Modus gleich 1.

TEN1. TEN numerischer ausdruck

Exponentialfunktion mit Basis 10 und numerischer Ausdruck als Exponent.

TEN3 ist gleich 1000.

2.3.3.3 STRING-FUNKTIONEN

Ein String ist eine Kette aus alphanumerischen und Sonderzeichen. String-Funktionen sind eine Gruppe von Operationen, die Strings manipulieren. Sie nehmen einen String und errechnen einen numerischen Wert dazu, oder Sie nehmen einen numerischen Wert und errechnen den String dazu, oder Sie bearbeiten den String als String. In vielen BASIC-Dialekten muß das Argument der Funktion in Klammern gesetzt werden. Nicht so bei diesem Computer. Klammern müssen nur dann gesetzt werden, wenn es nötig ist, das Argument von anderen Werten deutlich zu trennen. String-Ausdrücke mit zwei oder drei Ausdrücken benötigen immer eine Klammerung.

ASC

1. ASC "string ausdruck"

Errechnet den Zeichen-Code des ersten Zeichens des Ausdrucks. Die Zeichen-Code-Tabelle finden Sie in Anhang B.

ASC "A" ist gleich 65.

Der Computer benutzt ASCII-Codes und ihre Zeichen.

CHR\$

1. CHR\$ numerischer ausdruck

Errechnet das Zeichen des gegebenen numerischen Wertes. CHR\$ ist die Umkehrfunktion der ASC-Funktion.

CHR\$ 65 ist gleich "A".

Achtung: Wenn der Zeichencode von 13 angegeben wird, während Sie manuell das CHR\$-Kommando ausführen, wird der jeweils folgende Inhalt nicht angezeigt.

Beispiel:

CHR\$70+CHR\$71+CHR\$13+CHR\$75+CHR\$76

ENTER → FG

Die Zeichen K und L für die Codes 75 und 76 werden nicht angezeigt.

LEFT\$1. LEFT\$ (“string ausdruck”, numerischer ausdruck)

Diese Funktion erstellt einen Teil-String des durch “string ausdruck” angegebenen Strings. Von links beginnend wird die mit numerischer ausdruck gegebene Anzahl von Zeichen herausgenommen.

LEFT\$ (“ABCDEF”, 2) ist gleich “AB”.

LEN1. LEN “string ausdruck”

Errechnet die Länge eines Strings.

LEN “ABCDE” ist gleich 5.

MID\$1. MID\$ (“string ausdruck”, num asdr1, num asdr2)

Diese Funktion erstellt einen Teil-String aus dem durch ausdruck angegebenen string. Durch num asdr1 wird das erste Zeichen bestimmt, num asdr2 bestimmt die Anzahl der herauszunehmenden Zeichen.

MID\$ (“ABCDEF”, 2, 3) ist gleich “BCD”.

RIGHT\$1. RIGHT\$ (“string ausdruck”, numerischer ausdruck)

Diese Funktion erstellt einen Teil-String des durch “string ausdruck” angegebenen Strings. Von rechts beginnend wird die Anzahl der von numerischer ausdruck gegebenen Zeichen herausgenommen.

RIGHT\$ (“ABCDEF”, 3) ist gleich “DEF”.

STR\$1. STR\$ numerischer ausdruck

STR\$ wandelt einen numerischen Wert in einen String um.

STR\$ 1.59 ist gleich “1.59”.

VAL**1. VAL “string ausdruck”**

VAL ist die Umkehrfunktion zu STR\$. VAL wandelt einen String in einen numerischen Ausdruck um.

VAL “1.59” ist gleich 1.59.

Achtung: VAL kann nur numerische Zeichen (0 bis 9), Vorzeichen (+ und –) und das “E” für Exponenten umwandeln! Sind andere Zeichen in dem string ausdruck, so werden alle Zeichen rechts des letzten erlaubten Zeichens ignoriert. Spaces werden wie nicht-existent behandelt.

2.4 FEHLERSUCHE

In diesem Kapitel sollen Sie einige Hinweise erhalten, was Sie unternehmen können, wenn Ihr **SHARP PC-1403** nicht tut, was Sie von ihm erwarten. Es ist in zwei Teile gegliedert – der erste beschäftigt sich mit allgemeiner Bedienung des Gerätes, und der zweite mit der BASIC-Programmierung. Für jedes Problem schlagen wir eine Reihe von Lösungen vor. Sie sollten jede von ihnen versuchen – aber nur eine zur Zeit –, bis Sie das Problem gelöst haben.

BEDIENUNG DES GERÄTS

Wenn:

Sie das Gerät einschalten, aber nichts in der Anzeige erscheint.

Die Anzeige zwar funktioniert, aber keine Reaktion auf Tastendruck erfolgt.

Sie eine Rechnung oder Antwort eingeben, aber keine Reaktion erfolgt.

Sie ein BASIC-Programm abarbeiten lassen, etwas angezeigt wird und das Programm dann anhält.

Dann sollten Sie:

- 1) Überprüfen, ob Sie das Gerät eingeschaltet haben (ON-Position).
 - 2) Die **BRK** -Taste betätigen, um festzustellen, ob sich das Gerät automatisch abgeschaltet hatte.
 - 3) Die Batterien wechseln.
 - 4) Den Kontrast einstellen.
 - 5) Überprüfen, ob sich der Schiebeschalter in der LOCK-Position befindet.
-
- 1) **C-CE** drücken, um die Anzeige zu löschen.
 - 2) **CA** (**SHIFT** **C-CE**) drücken, um die Anzeige zu löschen.
 - 3) Das Gerät aus- und wieder einschalten.
 - 4) Irgendeine Taste drücken, festhalten und den ALL-RESET-Schalter drücken.
 - 5) Den ALL-RESET-Schalter ohne zusätzlichen Tastendruck betätigen.
-
- 1) Drücken Sie **ENTER** .
-
- 1) Drücken Sie **ENTER** .

Sie eine Rechnung eingeben und diese in der Form eines BASIC-Statements angezeigt wird (Doppelpunkt nach der ersten Zahl).

1) Schalten Sie vom PROgramm- in den RUN-Modus um.

Sie keinerlei Reaktion auf Tastenbetätigung erhalten.



1) Halten Sie irgendeine Taste fest und drücken Sie den ALL-RESET-Schalter.

2) Wenn Sie dann noch immer keine Reaktion erhalten, drücken Sie den ALL-RESET-Schalter, ohne dabei eine andere Taste festzuhalten. Dabei werden jedoch alle Daten, Programme und Speicherinhalte gelöscht.



FEHLERSUCHE IM BASIC

Wenn Sie ein neues BASIC-Programm eingeben, wird dieses in der Regel beim ersten Startversuch nicht laufen. Selbst, wenn Sie nur ein Programm abtippen, von dem Sie wissen, daß es korrekt ist, wie z. B. die in diesem Handbuch vorgestellten, dürfte Ihnen normalerweise mindestens ein Tippfehler unterlaufen. Handelt es sich um ein längeres Programm, wird es oft auch mindestens einen logischen Fehler enthalten. Es folgen einige grundsätzliche Hinweise, wie Sie solche Fehler finden und korrigieren.

Sie starten Ihr Programm und erhalten eine Fehlermeldung:

1. Schalten Sie zurück in den PROgramm-Modus und benutzen Sie die  oder  -Taste, um die fehlerhafte Zeile ins Display zu rufen. Der Cursor wird sich an der Stelle befinden, wo Ihr Computer verwirrt wurde.
2. Wenn Sie aus der Art, wie die Programmzeile geschrieben ist, keinen offensichtlichen Fehler entnehmen können, kann das Problem auch an den von Ihnen verwendeten Werten liegen. So wird beispielsweise CHR\$(A) eine Fehlermeldung bewirken, wenn A den Wert 1 hat, weil CHR\$(1) ein unerlaubtes Zeichen ist. Überprüfen Sie die Werte der von Ihnen verwendeten Variablen, indem Sie entweder im RUN- oder im PROgramm-Modus die einzelnen Variablennamen gefolgt von **ENTER** eingeben.

Sie starten das Programm mit RUN und erhalten keine Fehlermeldung, doch das Programm tut nicht, was Sie von ihm erwarten:

3. Überprüfen Sie das Programm Zeile für Zeile unter Verwendung von LIST und den  - und  -Tasten, um herauszufinden, ob Sie das Programm korrekt

eingegeben haben. Es ist erstaunlich, wieviele Fehler beim bloßen erneuten Durchsehen eines Programms gefunden werden können!

4. Versuchen Sie, jede einzelne Zeile beim Durchsehen so zu interpretieren, als wären Sie ein Computer. Nehmen Sie einfache Werte und realisieren Sie die Operationen der einzelnen Zeilen, um herauszufinden, ob Sie die gewünschten Ergebnisse erhalten.
5. Fügen Sie eines oder mehrere zusätzliche PRINT-Statements in Ihr Programm ein, um die einzelnen Werte und Tastenbelegungen zur Anzeige zu bringen. Benutzen Sie diese, um die korrekten Teile des Programms von den möglicherweise fehlerhaften zu isolieren. Diese Vorgehensweise ist auch nützlich um zu bestimmen, welche Teile des Programms schon abgearbeitet wurden. Sie können den Programmablauf auch an kritischen Stellen vorübergehend mit STOP unterbrechen, um dann einzelne Variable zu überprüfen.
6. Verwenden Sie TRON und TROFF, entweder als Befehle oder als Programmbestandteile, um den Programmablauf durch die einzelnen Zeilen hindurch verfolgen zu können. Halten Sie das Programm an kritischen Punkten an, um den Inhalt von wichtigen Variablen zu überprüfen. Dies ist zwar eine sehr langsame Methode, Fehler aufzuspüren – aber es ist manchmal die einzige.

ANHANG

ANHANG A

FEHLERMELDUNGEN

Fehlernummer	Bedeutung
1	<p>Syntax-Fehler</p> <p>Der Computer kann nicht verstehen, was Sie eingegeben haben. Prüfen Sie die Eingabe auf Dinge wie Semikolon am Ende eines PRINT-Statements, falsch geschriebene Befehle und fehlerhaften Gebrauch von Zeichen.</p> <p>(z.B.: 3 * /2)</p>
2	<p>Rechenfehler</p> <p>Hier haben Sie vermutlich eines der drei folgenden Dinge getan:</p> <ol style="list-style-type: none"> 1. Versucht, eine zu große Zahl zu benutzen; das Rechenergebnis ist größer als 9.999999999 E 99. 2. Versucht, durch Null zu teilen. (z.B.: 5 / Ø) 3. Versucht, eine unlogische Rechnung aus führen. (z.B.: LN – 3Ø oder ASN 1.5)
3	<p>Unzulässige Funktion (DIMensionierungs-Fehler, Argument-Fehler)</p> <p>Die Feld-Variable existiert bereits. Sie haben versucht, ein Feld zu spezifizieren, ohne es zuvor zu dimensionieren.</p> <p>Der Index des Feldes übersteigt die im DIM-Statement vorgegebene Größe. (z.B.: DIM B(256))</p> <p>Unzulässiges Funktionsargument, d.h. Sie haben versucht, Ihren Computer etwas tun zu lassen, wozu er nicht imstande ist. Wenn das Zeitintervall des WAIT-Befehls größer als 65535 ist. (z.B.: WAIT 66ØØØ)</p>

- 4 Zu hohe Zeilennummer
- Hier haben Sie vermutlich einen der folgenden Fehler gemacht:
1. Versucht, mit GOTO, GOSUB, RUN, LIST, THEN etc. eine nicht vorhandene Zeile anzusprechen.
 2. Versucht, eine zu große Zeilennummer zu verwenden. Die höchstmögliche Zeilennummer ist 65279.
- 5 Verschachtelungsfehler
- Das Einrichten einer Subroutine übersteigt 10 Ebenen.
- Das Einrichten einer FOR-Schleife übersteigt 5 Ebenen.
- Die Befehle RETURN ohne GOSUB, NEXT ohne FOR, READ ohne DATA wurden benutzt.
- Fassungsvermögen des Puffers überschritten.
- 6 Speicherkapazität überschritten
- Diese Fehlermeldung erhalten Sie normalerweise, wenn Sie versuchen, ein Feld zu DIMensionieren, das zu groß für die Speicherkapazität ist, oder wenn ein Programm zu lang ist.
- 7 PRINT USING Fehler
- Das bedeutet, daß in einem USING-Statement eine unzulässige Format-Spezifikation enthalten ist.
- 8 E/A-Anschluß-Fehler
- Dieser Fehler kann nur auftreten, wenn Sie den zusätzlichen Drucker und/oder Cassettenrecorder an den Computer angeschlossen haben.
- Dieser Fehler kann auch auftreten, wenn Sie den seriellen E/A-Anschluß benutzen.
- Die Meldung informiert Sie über ein Übermittlungsproblem zwischen dem Drucker und/oder Cassettenrecorder und dem Computer.

Andere Fehler

Dieser Fehlercode wird angezeigt, wenn der Computer ein Problem hat, das mit den anderen Fehlercodes nicht zu erfassen ist. Eine der häufigsten Ursachen für das Auftreten dieser Meldung ist der Versuch, Daten einer Variablen unter einem bestimmten Namen anzusprechen (z.B. A\$), während die Daten der Variablen unter dem anderen Namen (d.h. A) abgespeichert worden waren.

RENUM BETREFFENDE FEHLER


Fehlermeldung	Beschreibung
ERROR 1	Der RENUM-Befehl enthält einen Syntaxfehler.
ERROR 1 IN-Zeilen- nummer	Im Befehl für das Sprungziel fehlt eine Zeilennummer (z.B. GOTO, GOSUB usw.).
ERROR 3 IN-Zeilennummer	Während der Ausführung eines RENUM-Befehls wird eine Zeilennummer größer als 65279 angetroffen. Die Länge einer Programmzeile überschreitet 79 Bytes.
ERROR 4	Die angegebene alte Zeilennummer ist nicht im Programm vorhanden.
ERROR 4 IN-Zeilennummer	Die als Sprungziel angegebene Zeilennummer in der Programmdatei nicht vorhanden. Zeilennummer
ERROR 6	Die Speicherkapazität reicht nicht zur Durchführung des RENUM-Befehls oder wird während der Neunummerierung reduziert.
ERROR 9	Es würde versucht, RENUM in einer anderen Betriebsart als PRO (Programm) durchzuführen. Es wurde versucht, die Ablauffolge von Programmzeilen zu ändern, indem für die neue Zeilennummer ein niedrigerer Wert vorgegeben wurde als die Zeilennummer unmittelbar vor der alten Zeilennummer.
ERROR 9 IN-Zeilennummer	Die als Sprungziel angegebene Zeilennummer ist ungeeignet, weil sie eine Variable, einen Ausdruck oder eine Funktion verwendet (z.B. unkorrekte Zeilennummern-Bezugnahme).

ANHANG B

ASCII-CODE TABELLE

Die folgende Tabelle enthält die Umwandlungswerte für CHR\$ und ASC. Die Spalten geben das erste hexadezimale Zeichen bzw. die ersten vier Bits, die Zeilen das zweite hexadezimale Zeichen bzw. die letzten vier Bits. Oben links in jedem Kästchen ist der Dezimalcode für das jeweilige Zeichen angegeben. Das Zeichen selbst befindet sich unten rechts. Wenn kein Zeichen angegeben ist, ist dieses Zeichen nicht verfügbar. Beispielsweise ist das Zeichen "A" dezimal 65, hexadezimal 41 und binär 01000001. Das Zeichen "√" ist dezimal 252, hexadezimal FC und binär 11111100.

Achtung:

- * Die Zeichen für die Zeichencodes 92 (&5C), 249 (&F9) und 250 (&FA) unterscheiden sich in der Anzeige auf dem Display vom Ausdruck mit den Druckern CE-126P.
- * Das vom CE-126P Drucker mit dem Zeichensatzcode 92 (&5C) bezeichnete Zeichen ist  und ist in der nachfolgenden Liste aufgeführt.
- * Beim Einsatz des CE-126P Druckers wird Code 0(&00) nicht verwendet.
- * Andere Codes als 0(&00) bis 31 (&1F), denen keine Zeichen zugewiesen sind, werden als Leerstellen verstanden.
- * Die Codes 249 (&F9) und 250 (&FA) sind Leerstellen.

Erste 4 Bits

Z
w
e
i
t
e
4
B
i
t
s

hex binär	0	1	2	3	4	5	6	7	8	E	F
	0000	0001	0010	0011	0100	0101	0110	0111	1000	1110	1111
0	0	16	32	48	64	80	96	112	128	224	240
0000	NUL		SPACE	0	@	P	q	p			
1	1	17	33	49	65	81	97	113	129	225	241
0001			!	1	A	Q	a	q			
2	2	18	34	50	66	82	98	114	130	226	242
0010			"	2	B	R	b	r			
3	3	19	35	51	67	83	99	115	131	227	243
0011			#	3	C	S	c	s			
4	4	20	36	52	68	84	100	116	132	228	244
0100			\$	4	D	T	d	t			
5	5	21	37	53	69	85	101	117	133	229	245
0101			%	5	E	U	e	u			♠
6	6	22	38	54	70	86	102	118	134	230	246
0110			&	6	F	V	f	v			♥
7	7	23	39	55	71	87	103	119	135	231	247
0111			'	7	G	W	g	w			♦
8	8	24	40	56	72	88	104	120	136	232	248
1000			(8	H	X	h	x			♣
9	9	25	41	57	73	89	105	121	137	233	249
1001)	9	I	Y	i	y			■
A	10	26	42	58	74	90	106	122	138	234	250
1010			*	:	J	Z	j	z			□
B	11	27	43	59	75	91	107	123	139	235	251
1011			+	;	K	[k	{			π
C	12	28	44	60	76	92	108	124	140	236	252
1100			,	<	L	\	l				√
D	13	29	45	61	77	93	109	125	141	237	253
1101			-	=	M]	m	}			
E	14	30	46	62	78	94	110	126	142	238	254
1110			.	>	N	^	n	~			
F	15	31	47	63	79	95	111	127	143	239	255
1111			/	?	O	_	o				

ANHANG C

FORMATIEREN DER DATENAUSGABE

Es ist manchmal wichtig oder nützlich, neben dem Inhalt ausgegebener Daten auch das Format zu kontrollieren. Der Computer bedient sich hierzu des Befehls **USING**. Mit Hilfe dieses Befehls können Sie spezifizieren:

- die Anzahl der Stellen
- die Position des Dezimalpunktes
- wissenschaftliche Notation
- Anzahl der Zeichen in einem String

Diese verschiedene Formate bestimmen Sie mit einer "Ausgabe Maske", die aus einer String-Konstanten oder auch einer String-Variablen bestehen kann.

10: USING "####"

20: M\$ = "&&&&"

30: USING M\$

Wird der Befehl **USING** ohne Maske benutzt, werden damit alle Spezialformate aufgehoben.

40: USING

Der Befehl **USING** kann auch in einem **PRINT**-Statement benutzt werden.

50: PRINT USING M\$; N

Wann immer **USING** benutzt wird, bewirkt es die Kontrolle aller ausgegebenen Daten, bis ein neuer **USING**-Befehl auftritt.

NUMERISCHE DATEN

Eine numerische **USING**-Maske darf nur benutzt werden, um numerische Werte, d.h. numerische Konstante oder numerische Variable zu kontrollieren. Wird eine String-Konstante oder -Variable angezeigt, während eine numerische **USING**-Maske wirksam ist, wird die Maske hierauf nicht angewendet werden.

Ein auszugebender Wert muß immer in den von der Maske vorgesehenen Freiraum passen. Es muß auch Raum für das Vorzeichen vorgesehen sein, selbst, wenn die Zahl immer positiv sein wird. Eine Maske, die vier Stellen zuläßt, kann also nur für dreistellige Zahlen verwendet werden.

NUMERISCHE MASKEN

Die gewünschte Anzahl von Stellen wird mit Hilfe des "#"-Zeichens festgelegt. Die Anzeige oder der Ausdruck beinhalten immer soviele Zeichen, wie in der Maske vorgesehen sind. Dabei erscheint die Zahl rechts in diesem Bereich, die restlichen Stellen werden durch Leerschritte aufgefüllt. Bei positiven Zahlen wird sich daher auf der linken Seite der Anzeige immer mindestens ein Leerschritt befinden. Da der Computer nicht mehr als 10 Stellen erfassen kann, sollte die größte verwendete numerische Maske maximal 11 "#"-Zeichen enthalten.

Wenn die Gesamtstellenzahl des ganzzahligen Teils 11 Stellen übersteigt, wird dieser ganzzahlige Teil als 11 Stellen aufgefaßt.

Anmerkung: In allen Beispielen dieses Anhangs werden Sie am Anfang und am Ende eines Anzeigefeldes "I"-Zeichen finden, mit denen die Größe des Feldes anschaulich gemacht werden soll.

EINGABE

10: USING "####"

20: PRINT 25

30: PRINT -350

40: PRINT 1000

ANZEIGE

(Bringen Sie den Computer in den RUN-Mode, geben Sie RUN ein und drücken Sie die **ENTER** -Taste.)

| 25 |

| -350 |

ERROR 7 in 40

Beachten Sie, daß die letzte Eingabe eine Fehlermeldung hervorruft, weil 5 Stellen (4 Zahlen und eine Stelle für das Vorzeichen) benötigt werden, die Maske aber nur 4 vorsieht.

NACHKOMMASTELLEN

Das Zeichen für den Dezimalpunkt (".") kann in einer Maske enthalten sein, um die gewünschte Position des Dezimalpunktes festzulegen. Wenn die Maske mehr Stellen bereitstellt, als für den anzuzeigenden Wert benötigt werden, werden die auf der rechten Seite übrigbleibenden Stellen mit Nullen aufgefüllt. Enthält der anzuzeigende Wert mehr Stellen, als die Maske vorsieht, werden diese abgeschnitten (nicht gerundet).

EINGABE

10: USING "#####.##"

20: PRINT 25

30: PRINT -350.5

40: PRINT 2.547

ANZEIGE

| 25.00 |

| -350.50 |

| 2.54 |

WISSENSCHAFTLICHE NOTATION

Das “^”-Zeichen kann in einer Maske enthalten sein, um so anzuzeigen, daß die Zahl in wissenschaftlicher Notation ausgegeben werden soll. Die Zeichen “#” und “.” werden in der Maske benutzt, um den “charakteristischen” Teil der Zahl (d.h. den Teil, der links vom “E” steht) zu formatieren. Links vom Dezimalpunkt sollten immer 2 “#”-Zeichen stehen, um genug Raum für eine ganze Zahl und das Vorzeichen zu schaffen. Der Dezimalpunkt kann – muß aber nicht – vorgegeben werden.

Rechts vom Dezimalpunkt können bis zu 9 Stellen eingerichtet werden (mit Hilfe von “#”-Zeichen). Nach dem “charakteristischen” Teil wird das Exponentiationszeichen (E) angezeigt, gefolgt von der Stelle für das Vorzeichen und dem Exponenten (2stellig). Das kleinste von einer Maske vorzugebende Feld für wissenschaftliche Notation wäre demnach “##^”, damit werden Zahlen in der Form “2 E 99” dargestellt. Das größtmögliche Feld für wissenschaftliche Notation wäre “##.#####^”, das Zahlen wie “–1.234567890 E –12” darstellen kann.

EINGABE

10: USING “###.##^”

20: PRINT 2

30: PRINT –365.278

ANZEIGE

| 2.00E 00 |

| –3.65E 02 |

ALPHANUMERISCHE MASKEN

String-Konstanten und -Variable werden mit Hilfe des “&”-Zeichens ausgegeben. Jedes “&” gibt eine im vorgesehenen Feld anzuzeigende Stelle an. Der String wird auf der linken Seite dieses Feldes angezeigt. Ist der String kürzer als der dafür eingeräumte Raum, werden die rechts verbleibenden Stellen mit Leerschritten aufgefüllt. Ist der String länger als das Feld, wird er abgeschnitten:

EINGABE

10: USING “&&&&&”

20: PRINT “ABC”

30: PRINT “ABCDEFGHI”

ANZEIGE

| ABC |

| ABCDEF |

GEMISCHTE MASKEN

In den meisten Anwendungsfallen wird eine USING-Maske entweder alle notwendigen Zeichen zur Formatierung eines numerischen Ausdrucks oder zur Formatierung eines Strings beinhalten. Für bestimmte Zwecke können aber auch beide gemeinsam in einer USING-Maske enthalten sein. In solchen Fällen markiert jede Umschaltung von String-formatierenden Zeichen auf Zahlen-formatierende

Zeichen (und umgekehrt) die Grenze für einen bestimmten Wert. So können mit einer Maske der Form “#####&&&” zwei verschiedene Werte formatiert werden – ein numerischer Ausdruck, für den 5 Stellen vorgesehen sind, und ein alphanumerischer Ausdruck, für den 4 Stellen bereitgestellt wurden.

EINGABEANZEIGE

10: PRINT USING “###.##&&”; 25; “CR”	25.00CR
20: PRINT –5.789; “DB”	–5.78DB

Achtung: Wurde ein USING-Statement einmal spezifiziert, wirkt es sich auf alle nachfolgenden Daten aus, bis es aufgehoben oder durch einen anderen USING-Befehl ersetzt wird.

ANHANG D

BEWERTUNG VON AUSDRÜCKEN UND OPERATOREN-VORRANG

Wenn in den Computer ein komplexer Ausdruck eingegeben wird, bewertet er Teile dieses Ausdrucks in einer Reihenfolge, die durch die Vorrangstellung der einzelnen Teile bestimmt wird. Geben Sie den Ausdruck

$$100/5+45$$

ein, entweder als Rechenoperation oder als Teil eines Programms, so weiß der Computer nicht, ob Sie meinen:

$$\frac{100}{5+45} = 2 \quad \text{oder} \quad \frac{100}{5} + 45 = 65$$

Da der Computer eine Möglichkeit haben muß, zwischen diesen beiden Operationen zu entscheiden, bedient er sich seiner Regeln des Operatoren-Vorrangs. Da die Division eine höhere "Priorität" hat als die Addition (siehe unten), wird er entscheiden, daß zuerst die Division durchgeführt wird und anschließend die Addition, d.h. die zweite Möglichkeit wird ausgeführt und als Ergebnis 65 ausgegeben.

VORRANG VON OPERATOREN

Im BASIC-Modus werden Operatoren gemäß den folgenden Prioritätsregeln verarbeitet (angefangen mit der höchsten Priorität):

1. Klammern
2. Variable und Pseudovvariable
3. Funktionen
4. Exponentiation (^)
5. Vorzeichen (+, -)
6. Multiplikation und Division (*, /)
7. Addition und Subtraktion (+, -)
8. Verhältnis-Operatoren (<, <=, =, <>, >=, >)
9. Logische Operatoren (AND, OR, NOT, XOR)

Treten in einem Ausdruck zwei oder mehr Operatoren derselben Prioritätsstufe auf, wird der Ausdruck von links nach rechts verarbeitet. (Exponentiation wird von rechts nach links verarbeitet!) Beachten Sie, daß bei einer Operation $A+B-C$ das Ergebnis dasselbe ist, ob Sie nun die Addition oder die Subtraktion zuerst ausführen.

Enthält ein Ausdruck ineinanderliegende Klammern, so wird die innerste Klammer zuerst bearbeitet, die äußerste zuletzt.

Für die Prioritätsstufen 3 und 4 gilt, daß die letzte Eingabe die höchste Priorität hat.

z. B.:

$$-2 \wedge 4 \longrightarrow -(2^4)$$

$$3 \wedge -2 \longrightarrow 3^{-2}$$

BEISPIEL FÜR EINE BEWERTUNGSFOLGE

Wir gehen aus von dem Ausdruck:

$$((3+5-2) * 6+2)/10 \wedge \text{LOG } 100$$

Der Computer verarbeitet nun zuerst die innersten Klammern. Da "+" und "-" auf derselben Stufe stehen, wird von links nach rechts gerechnet, also die Addition zuerst ausgeführt.

$$((8-2) * 6+2)/10 \wedge \text{LOG } 100$$

Dann wird die Subtraktion durchgeführt:

$$((6) * 6+2)/10 \wedge \text{LOG } 100$$

Oder:

$$(6 * 6+2)/10 \wedge \text{LOG } 100$$

In der nächsten Klammer wird zuerst die Multiplikation durchgeführt:

$$(36+2)/10 \wedge \text{LOG } 100$$

Dann folgt die Addition:

$$(38)/10 \wedge \text{LOG } 100$$

Oder:

$$38/10 \wedge \text{LOG } 100$$

Nachdem nun die Klammern aufgelöst sind, hat die LOG-Funktion höchste Priorität.

$$38/10 \wedge 2$$

Als nächstes folgt die Exponentiation:

$$38/100$$

Und zuletzt wird die Division ausgeführt:

$$0.38$$

Dies ist der Wert des Ausdrucks.

ANHANG E

TASTENFUNKTIONEN

ON
BRK

(ON) wird benutzt, um den Computer anzuschalten, wenn er sich automatisch abgeschaltet hatte.

(BREAK)

Das Betätigen dieser Taste während eines Programmablaufs bewirkt eine Unterbrechung der Programmausführung.

Bei manuellen Operationen, Ein-/Ausgabe-Befehlen wie BEEP, CLOAD etc. wird mit Betätigung dieser Taste die Befehlsausführung unterbrochen.

SHIFT

Die gelbe Taste mit der Aufschrift "SHIFT" muß benutzt werden, um Doppelfunktionen anzusprechen (die jeweils in brauner Schrift über den entsprechenden Tasten stehenden Zeichen). Z. B. wird mit **SHIFT** **U** das "?" angesprochen.

C-CE

Wird benutzt, um Eingabe und Anzeige zu löschen, weiterhin hebt diese Taste eventuelle Blockaden durch Fehler auf.

SHIFT **CA**

Löscht nicht nur den Anzeigehalt, sondern initialisiert darüber hinaus den Computer, d. h.

- hebt den WAIT-Timer auf;
- löscht das Anzeigeformat (USING-Format);
- hebt den TRON-Befehl auf (TROFF);
- hebt PRINT=LPRINT auf;
- hebt Blockade durch Fehler auf.

0 bis **9** Zifferntasten**.**

Dezimalpunkt

- wird benutzt, um eine Abkürzung eines Befehls, eines Kommandos oder einer Funktion einzugeben;
- gibt in der Bestimmung eines USING-Formats die Stellung des Dezimalpunktes an.

E

Wird benutzt, um in wissenschaftlicher Notation den Exponenten zu bestimmen (mit der Buchstabentaste "E").

/

Divisionszeichen.

Multiplikationszeichen



- wird benutzt, um eine Feld-Variable in INPUT#, PRINT# etc. zu bestimmen.



- +** Additionszeichen
- Subtraktionszeichen
- SHIFT** **^** Wird benutzt, um Zahlen zu potenzieren und um das Exponenten-Anzeige-System für numerische Daten in USING-Anweisungen zu spezifizieren.
- SHIFT** **<** Werden benutzt, wenn logische Operatoren in einer IF-Sequenz eingegeben werden sollen.
- SHIFT** **>**
- DEF** Wenn eines der folgenden 18 Zeichen (A, S, D, F, G, H, J, K, L, , , Z, X, C, V, B, N, M, SPaCe) nach Betätigen der **DEF** -Taste gedrückt wird, wird das entsprechend gekennzeichnete Programm gestartet.
- A** bis **Z** Buchstabentasten
- Diese Tasten sind Ihnen wahrscheinlich von einer gewöhnlichen Schreibmaschine her vertraut.
- Auf einfachen Tastendruck erscheinen Großbuchstaben in der Anzeige. In den Kleinschriftmodus schalten Sie um, indem Sie **SML** drücken.
- SPC** Wird beim Eingeben von Zeichen oder Programmen benutzt, um einen Leerschritt zu produzieren.
- =**
- In Zuweisungs-Statements: weist der links vom “=” stehenden Variablen die rechts stehenden Zeichen (Buchstabe oder Zahl) zu.
 - Wird weiterhin benutzt, wenn in einer IF-Sequenz logische Operatoren eingegeben werden.
- SHIFT** **!** Nur Textzeichen
- "** Kennzeichnet Textkonstante
- #** Bestimmt im USING “Format” die Anzahl der Ziffern
- \$** Kennzeichnet Textvariablen
- %** Nur Textzeichen
- &** Bestimmt im USING “Format” die Anzahl der Textzeichen; kennzeichnet eine Hexadezimalzahl
- ?** Zur Eingabe von CLOAD?
- SHIFT** **e** Nur Textzeichen

SHIFT :	Trennzeichen zwischen mehreren Befehlen in einer Programmzeile
SHIFT ;	Interpunktion in Ein/Ausgabe-Anweisungen etc.
,	Interpunktion in Ein/Ausgabe-Anweisungen etc.
()	Eingabe von Klammern
▶	<ul style="list-style-type: none"> – Bewegt den Cursor nach rechts (auf einmaligen Tastendruck Bewegung um eine Stelle; wird die Taste festgehalten, Dauerfunktion). – Zur Ausführung von Playback-Anweisungen. – Löscht bei manueller Rechnung Fehlermeldungen.
◀	<ul style="list-style-type: none"> – Bewegt den Cursor nach links (auf einmaligen Tastendruck Bewegung um eine Stelle; wird die Taste festgehalten, Dauerfunktion). – Sonst wie unter der ▶-Taste beschrieben.
SHIFT INS	Fügt einen Leerschritt ein, dabei erscheint das Zeichen "□". Dieser Leerschritt wird vor das Zeichen gesetzt, auf dem der Cursor steht.
SHIFT DEL	Löscht den Inhalt der Stelle, auf der der Cursor steht.
SHIFT π	Zur Eingabe von Pi (π)
SHIFT √	Zur Eingabe der Quadratwurzel
SHIFT INPUT	BASIC-Befehlstasten: Durch Drücken von SHIFT und einer Buchstabetaste (einschließlich Komma, Leerstelle (SPC) und ENTER) wird der über der Taste angegebene Befehl in den Computer eingegeben.
SHIFT CLOAD	
CAL	Zur Einstellung des CAL-Modus.
BASIC	Zur Einstellung des RUN-Modus, wenn der CAL-Modus eingestellt ist.
	Zur Einstellung des PRO-Modus, wenn der RUN-Modus eingestellt ist.
	Mit jedem Drücken der Taste BASIC wird abwechselnd RUN und PRO eingestellt.
SHIFT DRG	Zur Einstellung der Winkereinheit.
hyp	Zur Eingabe einer hyperbolischen Funktion.
SHIFT archyp	Zur Eingabe einer inversen hyperbolischen Funktion.
sin ~ x²	Zur Eingabe der angegebenen Funktionen.
SHIFT sin⁻¹	
SHIFT π!	

- ENTER**
- Zur Eingabe einer Programmzeile in den Computer.
 - Wird beim Schreiben von Programmen benötigt.
 - Bedingt manuelle Rechnung oder direkte Ausführung eines Befehls durch den Computer.
 - Zum Neustart eines Programms, das zeitweise durch einen INPUT- oder PRINT-Befehl unterbrochen wurde.
- SHIFT** **P↔NP** Stellt die Verbindung zum Drucker her bzw. unterbindet sie (sofern ein zusätzlicher Drucker an den Computer angeschlossen ist).
- SML**
- Zum Anwählen und Aufheben des Kleinschriftmodus (schaltet die Anzeige SML an bzw. aus).
 - Die SML-Anzeige erscheint, wenn **SML** gedrückt wird. Wenn Sie nun die Tasten **A**, **B** und **C** betätigen, werden a, b und c in der Anzeige ausgegeben. Durch erneutes Drücken von **SML** schalten Sie den Kleinschriftmodus wieder aus, die Anzeige SML verschwindet und es werden wieder Großbuchstaben in der Anzeige ausgegeben.


Tasten für Matrix-Operationen sind auf Seite 69 aufgeführt.

Die  - und  -Tasten haben die folgenden Funktionen, je nach angewähltem Modus und Status des Computers:

Modus	Status		
RUN	Programmdurchführung		
	Programm vorübergehend unterbrochen, INPUT-, PRINT-Statement wird ausgeführt	um die nächste Zeile durchführen zu lassen.	Diese Taste festhalten, um bereits abgearbeitete Zielen zur Anzeige zu bringen.
	Unterbrechung		
	Fehlermeldung während der Programmausführung		
	TRON Status	zur Fehlersuche	um die fehlerhafte Zeile zur Anzeige zu bringen
			Diese Taste festhalten, um bereits abgearbeitete Zeilen zur Anzeige zu bringen.

Wenn der Modus von RUN nach PRO geändert wird, wird dies nicht angezeigt!

PRO	Programm vorübergehend unterbrochen	um die unterbrochene Zeile anzuzeigen	wie links
	Fehler	um die fehlerhafte Zeile anzuzeigen.	wie links
	Anderer Status	um die erste Zeile anzuzeigen.	um die letzte Zeile anzuzeigen.
(Wenn die Programmzeile angezeigt wird:)		um die nächste Programmzeile anzuzeigen.	um die vorherige Programmzeile anzuzeigen.

- In der Anzeige ist die  -Taste das gleiche wie ein Leerschritt.
- Wenn 12 Minuten lang keine Taste betätigt wird, schaltet sich der Computer automatisch ab.

ANHANG F

TECHNISCHE DATEN

Modell:	PC-1403 Taschencomputer
Prozessor:	8 Bit CMOS CPU
Programmiersprache:	BASIC
Speicherkapazität:	System-ROM 72K Bytes RAM: System: ca. 1,1K Bytes Anwender: Fester Speicherbereich 208 Bytes (A-Z, A\$-Z\$) Programm-/Datenbereich 6878 Bytes
Stack:	Unterroutine: 10 Stacks Funktion: 16 Stacks FOR-NEXT: 5 Stacks Daten: 8 Stacks
Operatoren:	Addition, Subtraktion, Multiplikation, Division, trigonometrische und inverstrigonometrische Funktionen, logarithmische und Exponentialfunktionen, Winkelumkehrfunktionen, Quadrat und Quadratwurzel, Vorzeichen, Absolut, Integral, Verhältnis-Operatoren, logi-Operatoren, Matrix Operatoren.
Rechengenauigkeit:	10 Stellen (Mantisse) + 2 Stellen (Exponent)
Editiermöglichkeit:	Cursor rechts und links, Zeile auf- und abwärts, Zeicheneinfügung, Zeichenlöschung
Speicherschutz:	CMOS, batteriegestützt

Anzeige:	24-Stellen-Flüssigkristallanzeige mit 5 x 7-Punkt-Zeichen
Tastatur:	77 Tasten, alphabetisch, numerisch, Sonderzeichen, Funktionen, numerische Schablone, definierbare Tastenfunktionen
Stromversorgung:	6,0V DC-Lithium-Zellen Typ: CR-2032 x 2
Stromverbrauch:	6,0V DC ca. 0,03 W Ca. 120 Stunden bei Dauerbetrieb unter normalen Bedingungen. (Gilt bei einer Betriebstemperatur von 20°C, wenn sich eine Betriebsstunde aus 10 Minuten für Bedienung und Programmausführung und 50 Minuten Anzeigzeit zusammensetzt.)
	* Abhängig von Arbeitsmethoden und verwendetem Batterietyp schwankt die Betriebszeit leicht.
Betriebstemperatur:	0 Grad Celsius bis 40 Grad Celsius
Abmessungen:	170 (B) x 72 (T) x 9.5 (H) mm
Gewicht:	ca. 150 g (mit Batterien)
Zubehör:	Deckel, 2 eingebaute Lithium-Zellen, Tastaturschablone und Handbuch
Zusatzgeräte:	Cassetenrecorder (CE-152) Drucker/Cassetten-Interface (CE-126P)

ANHANG G

DIE BENUTZUNG VON PROGRAMMEN, DIE FÜR
ANDERE PC-MODELLE ENTWICKELT WURDEN

Programme, die für die folgenden Computer geschrieben wurden, können mit geringfügigen Veränderungen auf dem **PC-1403** verwendet werden.

PC-1210 Serie: PC-1210/11

PC-1245 Serie: PC-1245/46/47

PC-1250 Serie: PC-1250/51

PC-1260 Serie: PC-1260/61

PC-1350 Serie: PC-1350

PC-1401 Serie: PC-1401/02

PC-1450 Serie: PC-1450

PC-1460 Serie: PC-1460

PC-2500 Serie: PC-2500

Obwohl sich die Funktionen der obigen Modelle geringfügig unterscheiden, können die mit diesen Modellen erstellten Programme auf dem **PC-1403** verwendet werden, wenn die nachstehend beschriebenen Modifikationen durchgeführt werden.

Hinweis 1: Der **PC-1403** kann Programme von Cassette lesen, die mit Computern der Serien **PC-1210**, **PC-1245** und **PC-1250** auf Cassette geschrieben wurden, aber umgekehrt können Programme, die mit dem **PC-1403** erstellt und auf Cassette geschrieben wurden, nicht mit den Computern der drei obengenannten Serien gelesen werden.

Hinweis 2: Programm-Cassetten für die Serien **PC-1245** und **PC-1210**, auf die mehrere Programme mit dem **MERGE**-Befehl aufgenommen wurden, können nicht mit dem **PC-1403** verwendet werden. Vielmehr müssen die Programme einzeln mit dem **MERGE**-Befehl in den **PC-1403** geladen werden.

Hinweis 3: Programme, die mit der Serie **PC-1250** erstellt werden und die Befehle **POKE** oder **CALL** enthalten, können nicht auf dem **PC-1403** ausgeführt werden. Wird die Ausführung auf dem **PC-1403** versucht, kann es vorkommen, daß der Computer keine Eingabe von den Tasten annimmt.

MODIFIKATIONEN VON PROGRAMMEN DER PC-1245 SERIE (PC-1250 SERIE)

Wenn Sie die Programme, die für die **PC-1245** Serie entwickelt wurden, auf dem **PC-1403** benutzen wollen, müssen Sie folgende Änderungen vornehmen:

1. Multiplikation ohne das " * "-Zeichen

Bei der **PC-1245** Serie kann der Operator für die Multiplikation (*) fortgelassen werden (z.B. **AB** anstatt **A * B**, **DC** statt **D * C**). Dies ist auf dem **PC-1403** nicht

möglich, da dieser zwei aufeinanderfolgende Zeichen als einfache Variable behandelt. Benutzen Sie daher die Schreibweise rechts im Beispiel:

(z. B.) $A = \text{SIN } \underline{BC} \rightarrow A = \text{SIN } \underline{(B * C)}$

2. Definition von Index-Variablen mit dem DIM-Befehl

Wenn auf den Computern der PC-1245 Serie der Befehl DIM A(30) ausgeführt wird, werden Speicherplätze von A(27) bis A(30) als Erweiterung eines Definitionsbereiches einer festen Variablen reserviert.

Auf dem PC-1403 wird jedoch bei der Ausführung von DIM A(30) ein getrennter Speicherbereich für die Feld-Variablen A(0) bis A(30) und das Feld mit dem Namen A reserviert.

Wenn Sie Index-Variable als Erweiterung von Festvariablen definieren wollen, benutzen Sie bitte die rechts im Beispiel angegebene Schreibweise:

(z. B.) DIM A(30) \rightarrow A(30) = 0

3. Datenein-/ausgabebefehl für Dateien auf Cassette:

Bei der PC-1245 Serie wird mit dem Befehl PRINT #C der Inhalt der Variablen C und aller folgenden abgespeichert.

Beim PC-1403 jedoch wird mit diesem Befehl lediglich der Inhalt der Variablen C auf Band gespeichert. Um den Inhalt einer bestimmten Variablen und aller ihr folgenden auf Band abzuspeichern, benutzen Sie bitte die Schreibweise rechts im Beispiel:

(z. B.) PRINT # A \rightarrow PRINT # A*
 INPUT # C \rightarrow INPUT # C*

4. Wert einer Schleifenvariablen nach Beendigung einer FOR-NEXT-Schleife

Der Wert einer Schleifenvariablen, den Sie nach Ausführung einer FOR-NEXT-Schleife erhalten, unterscheidet sich beim PC-1403 von dem bei der PC-1245 Serie. Wird der Wert einer Schleifenvariablen in einem Bedingungsausdruck benutzt, müssen Sie ihn beim PC-1403 gegenüber der PC-1245 Serie um 1 erhöhen.

(z. B.) 10 FOR I=0 TO 10
 50 NEXT I
 60 IF I=10 THEN 100

Ändern Sie den Wert in Zeile 60 folgendermaßen:

60 IF I=11 THEN 110

(Auf dem PC-1403 müssen Sie den Wert einer Schleifenvariablen um eins erhöhen. Die Anzahl der Schleifendurchläufe bleibt jedoch gleich.)

5. Umdefinition von =

Die Gleichheitstaste = funktioniert beim PC-1403 nicht als definable Key. Daher muß in Programmen, in denen die Gleichheitstaste definiert ist, eine andere Taste verwendet werden.

(z.B.) 100 "=" : ... → 100 "N" : ...

6. Exponentialzeichen "IE"

Der PC-1403 benützt den Großbuchstaben "E" für die Darstellung der Exponentiation. Die folgenden Änderungen werden dadurch erforderlich:

A = 1.234 IE 5 → A = 1.234E5
 B = IE 6 → B = 1E6

Wird ein Programm der PC-1245 Serie von Band in den PC-1403 eingelesen, wird diese Veränderung automatisch vom PC-1403 durchgeführt.)

7. Der Zeichen-Code der PC-1245 Serie unterscheidet sich geringfügig von dem des PC-1403. Wenn die folgenden Codes durch die CHR\$-Funktion zugewiesen werden, ändern Sie die Codes:

ASCII	PC-1245	PC-1403
39 (&27)	⌈	,
91 (&5B)	√	[
92 (&5C)	¥	\
93 (&5D)	π]
96 (&60)	IE	.
250 (&FA)	– (Error)	⌈
251 (&FB)	– (Error)	⌊
252 (&FC)	– (Error)	π
		√

Anmerkung: Wie oben gezeigt, verfügt der PC-1403 nicht über das Zeichen IE.

ZUSÄTZLICHE MODIFIKATIONEN

1. Die Modelle der PC-1245 Serie verfügen über eine Programmzeilenspanne von 1 – 999. Der PC-1403 hat eine erweiterte Spanne von 1 – 65279. Aus diesem Grund beansprucht die Zeilennummer 3 Bytes im RAM (bei der PC-1245 Serie nur 2 Bytes.). Die Änderung wird automatisch ausgeführt, wenn Sie Programme von Cassette in den Computer einladen. Es besteht die Möglichkeit eines Speicherüberlaufs, wenn Sie zu lange Programme laden oder ausführen lassen (ERROR 6). Wenn eine Zeile etwa 80 Bytes umfaßt, kann durch diese Veränderung das Zeilenende gelöscht werden.

2. Wenn Sie ein Programm der PC-1245 Serie in den PC-1403 einladen, wird aufgrund der Änderung in der Zeilennummerierung (von 2 auf 3 Bytes) die BUSY-Anzeige noch für 1 oder 2 Sekunden aufleuchten, nachdem das Band zu laufen aufgehört hat. Während dieser Zeit wird das "*" -Zeichen in der rechten unteren Ecke des Zeilendisplays sichtbar sein.

ÄNDERUNGEN BEZÜGLICH DER PC-1210 SERIE

Um Programm der PC-1210 Serie auf dem PC-1403 zu benutzen, müssen diese in gleicher Weise geändert werden, wie Programme der PC-1245 Serie (außer den Punkten 2 und 7). Zusätzlich sind folgende Änderungen notwendig:

1. IF-Statement

Beispiel:

Die folgende Programmzeile eines Programms der PC-1210 Serie

```
50 IF A > L PRINT "A"
```

wird vom PC-1450 interpretiert als

```
50 IF A > LPRINT "A"
```

und erzeugt eine Fehlermeldung.

Der Fehler taucht auf, weil ein Befehl (LPRINT) nicht in der PC-1210 Serie verwendet wird, wohl aber beim PC-1403.

Um dieses Problem zu lösen, sollten Sie in das IF-Statement einen THEN-Befehl einfügen:

```
50 IF A > L THEN PRINT "A"
```

2. USING-Format

Der USING-Befehl unterscheidet sich bei der PC-1210 Serie und dem PC-1403. in folgender Weise:

Beispiel:

```
10 A = -123.456
20 PAUSE USING "####.##"; A
30 PAUSE A, USING "####"; A
```

Bei der Ausführung dieses Programms wird folgendes angezeigt:

* PC-1210	-123	-123.45 -123
* PC-1403	-123.45	-123.45 -123

In der PC-1210 Serie wird die Anzeige auf der linken Seite im gleichen Format angezeigt wie auf der rechten Seite. Beim PC-1403 gestaltet sich die Anzeige entsprechend dem vorangegangenen Format-Befehl.

Dies bezieht sich nicht nur auf den PAUSE-Befehl, sondern gilt auch für PRINT- und LPRINT-Befehle.

3. Auslassen von ")"

In der PC-1210 Serie kann das ")" weggelassen werden, sofern es direkt vor ENTER oder ":" vorkommt. Beim PC-1403 kann das ")"-Zeichen nicht ausgelassen werden.

Deshalb müssen Sie es einfügen, wenn Sie Programme übernehmen wollen.

4. PRINT-Befehl

Der PC-1403 hat einen PRINT-Befehl für die Anzeige und einen LPRINT-Befehl für die Ausgabe auf dem Drucker. Alle PRINT-Befehle können umgeändert werden in LPRINT-Befehle durch ein PRINT=LPRINT-Statement.

Die PC-1210 Serie hat keinen LPRINT-Befehl. Um bei Übernahme von Programmen auf dem PC-1403 Ausgaben auf dem Drucker zu erhalten, müssen Sie ein PRINT=LPRINT-Statement in das Programm einfügen.

5. Variablen

Wird ein Programm mit RUN auf der PC-1210 Serie gestartet, werden alle Variablen beibehalten. Beim PC-1403 werden alle Variablen von A(27) an gelöscht.

Deshalb ist es nötig, falls die Variablen beibehalten werden sollen, das Programm auf dem PC-1403 mittels GOTO oder der **DEF** -Taste zu starten.

MODIFIKATIONEN FÜR DIE PC-1260 SERIE

(1) Zeichencode-Modifikation

Der Zeichencode 96 (&60) ist in der PC-1260 Serie ein Leerzeichen, aber im PC-1403 ein Hochkomma ('). Wenn daher in einem Programm der CHR\$-Befehl mit dem Zeichencode 96 zur Spezifizierung einer Leerstelle vorkommt, muß dieser Zeichencode in 32 (&20) verändert werden.

(2) Befehle CLS, CURSOR

Der PC-1403 verfügt nicht über die Befehle CLS und CURSOR. Daher müssen diese Befehle in Programmen, die auf dem PC-1403 laufen sollen, gelöscht und verändert werden.

MODIFIKATIONEN FÜR DIE PC-1401, PC-1450 UND PC-1460 SERIE

Programme, die auf Computern der Serie PC-1401, PC-1450 und PC-1460 erstellt wurden, können ohne Veränderungen auf dem PC-1403 laufen.

MODIFIKATIONEN FÜR DIE SERIEN PC-1350 UND PC-2500

Der PC-1403 verfügt nicht über die folgenden Befehle, so daß diese verändert werden müssen, wenn sie in einem Programm vorkommen.

CLS, CURSOR, MEM\$, GCURSOR, GPRINT, LINE, POINT, PRESET, PSET, (TEST)

Programmierungsbeispiele

Durch das Lesen der Beschreibung zu den verschiedenen Funktionen im vorangehenden Kapitel haben Sie sich mit einer Anzahl von Programmbefehlen vertraut gemacht. Zur Zusammenstellung von Befehlen für Applikationsprogramme in BASIC, ist es jedoch unbedingt notwendig, zusätzlich zu den in dieser Anleitung erläuterten Programmen auch eigene praktische Applikationsprogramme zu schreiben.

Genauso wie sich Autofahren nur durch die Handhabung des Lenkrads oder Tennis spielen durch Training erlernen läßt, kann auch effizientes Programmieren nur durch Üben möglichst vieler Programme gemeistert werden, ganz gleich wie gut Sie jede der Übungen beherrschen.

Es ist somit äußerst wichtig für Sie, zunächst Programme zu üben, die Sie nicht selbst geschrieben haben. In diesem Kapitel werden einige Programmierbeispiele mit Anwendung verschiedener Befehle in BASIC als Übungs- und Bezugsmaterial vorgestellt.

Zum besseren Verständnis der Programmbeispiele in diesem Kapitel werden nachfolgend die in diesen Beispielen verwendeten Bezeichnungen erläutert:

(1) PROGRAMMLISTUNG

Alle Programmlisten in den Programmierbeispielen werden mit dem Drucker CE-126P in tatsächlicher Größe ausgedruckt.

(2) PROGRAMMKAPAZITÄT

Am Ende einer jeden Programmliste wird die Kapazität des Programms selbst in der Anzahl von Bytes angegeben.

(3) AUSDRUCKEN

Bei auszudruckenden Programmen wird die Ausgabe für das ausgeführte Programm mit dem Drucker CE-126P in tatsächlicher Größe durchgeführt.

(4) SPEICHERINHALT

In der Speicherinhalttabelle jedes Programmbeispiels werden verschiedene Variablen mit vorherbestimmter Anwendung (z.B. solche, die zur Erhaltung von sofortigen Kalkulationszwischenergebnissen in den Arbeitsbereich einzuspeichern sind usw.) mit dem Prüfhäkchen "✓" angezeigt.

SHARP CORPORATION und/oder ihre Zweigniederlassungen tragen keine Verantwortung oder Haftung für etwaige finanzielle Verluste oder Schäden, die durch die Anwendung irgendeines der in dieser Anleitung beschriebenen Programmbeispiele entstehen. Bei der Anwendung dieser Programme ist daran zu denken, daß diese Programme ihren Anforderungen mitunter nicht vollständig gerecht werden oder manche Programm nicht so präzise sind, wie Sie es eventuell erwarten. Die Daten eines jeden Programmbeispiels sind deshalb sorgfältig zu überprüfen, um zu sehen, ob diese den Ansprüchen genügen. Sollte dies nicht der Fall sein, können diese durch Änderungen den Applikationszwecken angepaßt werden.

INHALT

Programmtitel	Seite
• Umwandlungen zwischen orthogonalen Koordinaten und Polarkoordinaten	225
• Berechnung des Flächeninhalts eines Vielecks	230
• Anschmiegen eines Kreises an zwei Kreise	237
• Zahlenspiel	241

PROGRAMMTITEL: Umwandlungen zwischen orthogonalen Koordinaten und Polarkoordinaten

Dies ist ein sehr nützliches Programm zur Durchführung von Umwandlungen zwischen orthogonalen Koordinaten und Polarkoordinaten (sphärische Koordinaten) in drei Dimensionen. Nach Eingabe aller Daten für die Umwandlung, wird das Resultat der Umwandlung entsprechend der zu dieser Zeit gültigen Winkeldefinition erzielt.

■ BEDIENUNG

1. **DEF** **A** drücken (zur Umwandlung von orthogonalen Koordinaten in Polarkoordinaten).

Wird der Wert jedes orthogonalen Koordinaten x , y und z entsprechend der Abfrage auf dem Display eingegeben, erscheint der Wert jedes Polarkoordinaten r , θ und φ in der angegebenen Reihenfolge, und das Programm ist beendet.

2. **DEF** **B** drücken (zur Umwandlung von Polarkoordinaten in orthogonale Koordinaten).

Wird der Wert jedes Polarkoordinaten R , θ und φ entsprechend der Abfrage auf dem Display eingegeben, erscheint der Wert jedes Polarkoordinaten x , y und z in der angegebenen Reihenfolge und das Programm ist beendet.

Achtung: Ist die Einheit der Winkeldefinition DEG, ergibt dies ein Umwandlungsergebnis in Gradeinheiten. Ebenso ergibt sich ein Umwandlungsergebnis in Radianten-Einheiten, wenn die Winkeldefinition als RAD angegeben wird.

■ BEISPIEL

1. Umwandlungen von orthogonalen Koordinaten in Polarkoordinaten $r=0$ und θ unbegrenzt, wenn $r=y=0$

$$r = \sqrt{x^2 + y^2 + z^2}$$

$$\theta = \sin^{-1}(z/r)$$

$$\varphi = \tan^{-1}(y/x) \text{ wenn } x > 0$$

$$\varphi = 90^\circ \text{ wenn } x=0 \text{ und } y \geq 0$$

$$\varphi = -90^\circ \text{ wenn } x=0 \text{ und } y < 0$$

$$\varphi = \tan^{-1}(y/x) + 180^\circ \text{ wenn } x < 0 \text{ und } y \geq 0$$

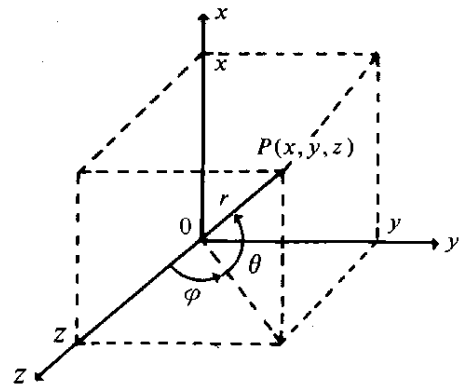
$$\varphi = \tan^{-1}(y/x) - 180^\circ \text{ wenn } x < 0 \text{ und } y < 0$$

2. Umwandlungen von Polarkoordinaten in orthogonale Koordinaten

$$x = r \cos \theta \cdot \cos \varphi$$

$$y = r \cos \theta \cdot \sin \varphi$$

$$z = r \sin \theta$$



■ BEISPIELE

1. Wandeln Sie orthogonale Koordinaten in Polarkoordinaten um.

$$x = -1$$

$$y = 2$$

$$z = -3$$

Winkeldefinitionsangabe: DEG

2. Wandeln Sie Polarkoordinaten in orthogonale Koordinaten um.

$$r = 3.741657387$$

$$\theta = -53.3077479$$

$$\varphi = 116.5650512$$

Winkeldefinitionsangabe: DEG

■ TASTENBETÄTIGUNG

< Orthogonale Koordinaten → Polarkoordinaten >

1. **DEF** **A**
 $x = _$
2. -1 **ENTER**
 $y = _$
3. 2 **ENTER**
 $z = _$
4. -3 **ENTER**
 r
5. **ENTER**
3.741657387
6. **ENTER**
THETA
7. **ENTER**
-53.30077479
8. **ENTER**
PHI
9. **ENTER**
116.5650512
10. **ENTER**
>

< Polarkoordinaten → Orthogonale Koordinaten >

1. **DEF** **B**
 $r = _$
2. 3.741657387 **ENTER**
THETA = _
3. -53.30077479 **ENTER**
PHI = _
4. 116.5650512 **ENTER**
 x
5. **ENTER**
-1.000000001
6. **ENTER**
 y
7. **ENTER**
2.
8. **ENTER**
 z
9. **ENTER**
-3.
10. **ENTER**
>

■ PROGRAMMLISTING

```

10:"A":INPUT "x=";X
20:INPUT "y=";Y
30:INPUT "z=";Z
40:R=SQR (X*X+Y*Y+Z*Z):
   IF R=0 PAUSE "r=0 Un
   defined":END
50:C=ASN (Z/R):IF X>0
   LET F=ATN (Y/X):GOTO
   80
60:A=(Y=0)+SGN Y:IF X=0
   LET F=A*ACS 0:GOTO 8
   0
70:F=ATN (Y/X)+A*ACS -1
80:WAIT :PRINT "r":
   PRINT R:PRINT "THETA
   ":PRINT C:PRINT "PHI
   ":PRINT F:END
90:"B":INPUT "r=";R
100:INPUT "THETA=";C
110:INPUT "PHI=";F
120:Y=R*COS C:X=Y*COS F:
   Y=Y*SIN F:Z=R*SIN C
130:WAIT :PRINT "x":
   PRINT X:PRINT "y":
   PRINT Y:PRINT "z":
   PRINT Z:END

```

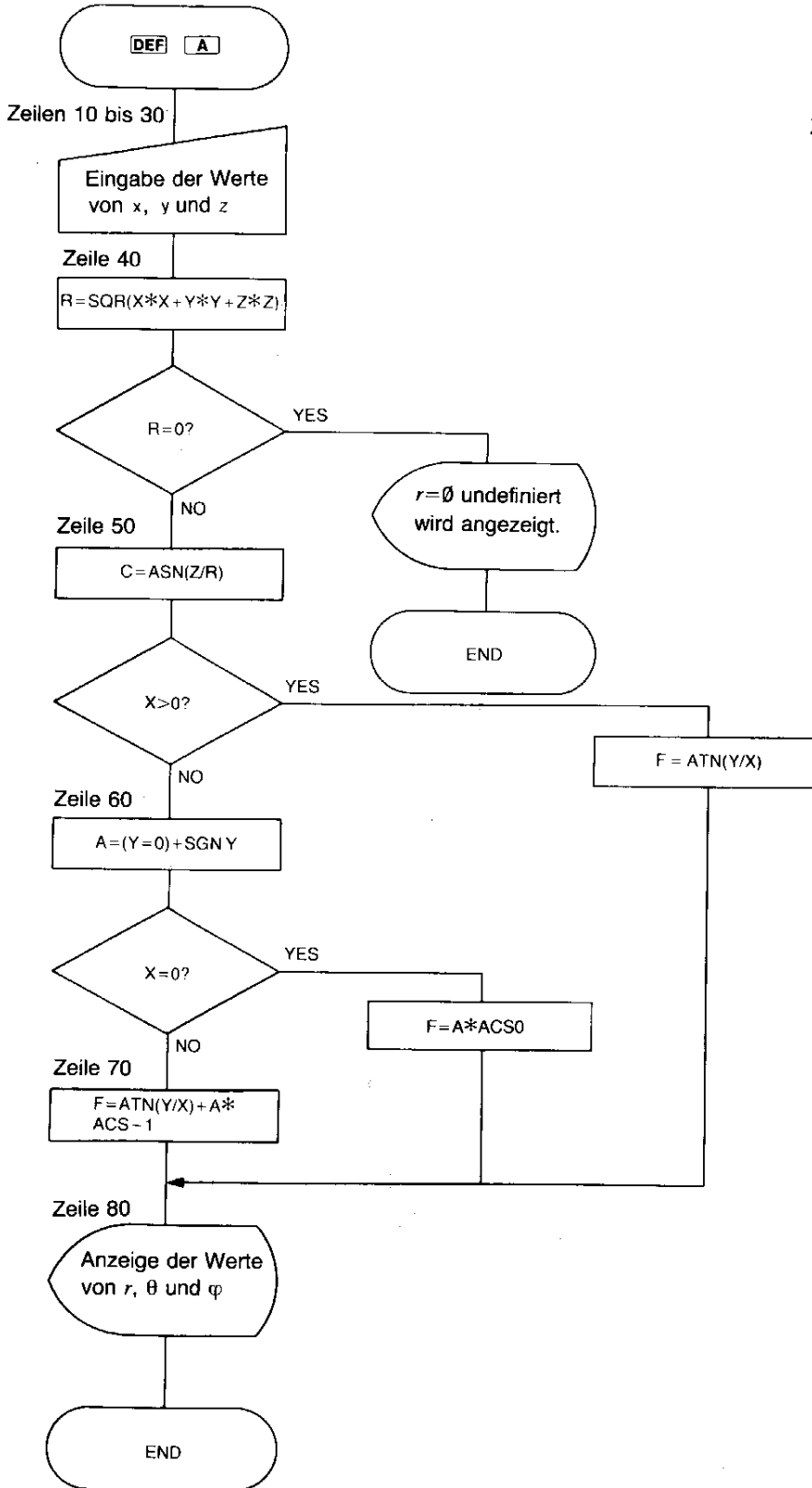
300 Bytes

■ SPEICHERINHALT

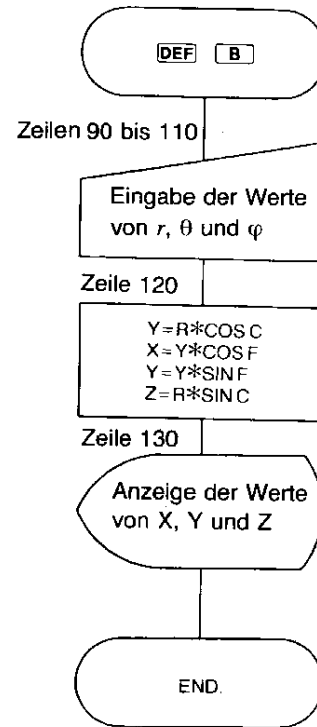
A	√
C	θ
F	φ
R	r
X	x-Koordinate
Y	y-Koordinate
Z	z-Koordinate

■ ABLAUFDIAGRAMM

(Orthogonale Koordinaten → Polarkoordinaten)



(Polarkoordinaten → Orthogonale Koordinaten)



Jedes Vieleck ist theoretisch eine Zusammensetzung aus Dreiecken. Lassen Sie uns den Flächeninhalt eines Vielecks berechnen. Dieses Programm errechnet den Flächeninhalt eines Vielecks durch Unterteilung des Vielecks in Dreiecke, berechnet dann den Flächeninhalt jedes Dreiecks und schließlich die Gesamtsumme der Fläche aller vorhandenen Dreiecke.

■ **BEDIENUNG**

1. **DEF** **A** drücken. (Programmstart)

Entsprechend der Abfrage auf dem Display, die Anzahl der Eckpunkte und die Koordinaten x und y jedes Eckpunktes eingeben.

2. Anschließend **DEF** **B** drücken.

Auf dem Display werden die Nummern von drei Eckpunkten abgefragt, die einzugeben sind, wonach die Fläche des Dreiecks ausgedruckt wird. Wird vor der Eingabe der Eckpunktnummer **ENTER** gedrückt, wird die Gesamtfläche aller Dreiecke auf dem Drucker ausgegeben.

3. Durch Betätigen von **DEF** **C** wird die Gesamtfläche aller Dreiecke gelöscht, und das Programm ist beendet.

Achtung: Für die Anzahl der Eckpunkte gelten folgende Grenzen:

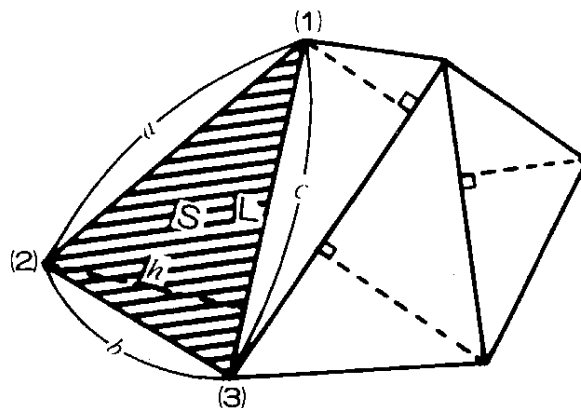
255 Eckpunkte

■ **BEISPIEL**

$$S = \sqrt{s(s-a)(s-b)(s-c)}$$

$$s = \frac{a+b+c}{2}$$

Fläche des Dreiecks $S' = \frac{1}{2}hL$



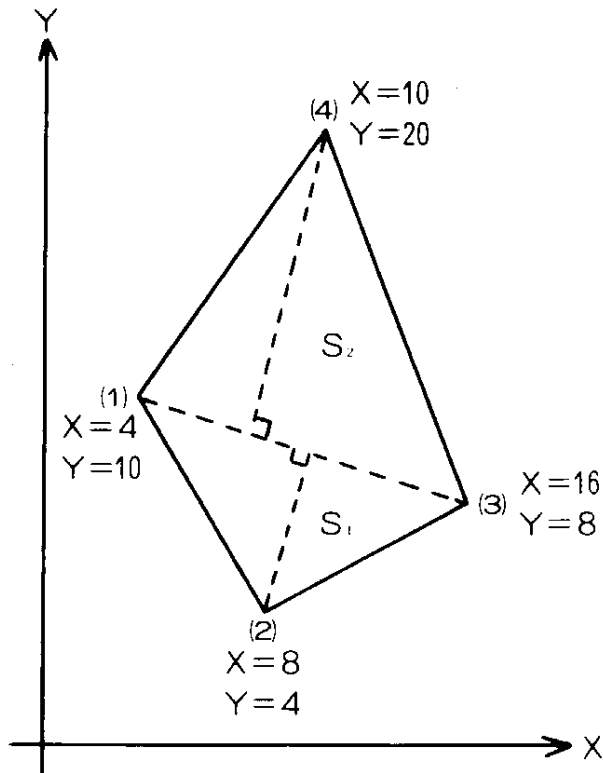
mit

L: Basislänge (längste Seite unter a, b und c)

h: Höhe (Die Werte können mit bis zu vier Nachkommastellen eingegeben werden.)

■ BEISPIEL

Berechnen Sie die Fläche des folgenden 4seitigen Vielecks.



■ AUSDRUCK

Point 1x= 4
 Point 1y= 10
 Point 2x= 8
 Point 2y= 4
 Point 3x= 16
 Point 3y= 8
 Point 4x= 10
 Point 4y= 20

1-2-3

A= 7.211
 B= 8.944
 C= 12.166
 H= 5.260
 S= 31.996580

1-3-4

A= 12.166
 B= 13.416
 C= 11.662
 H= 9.838
 S= 65.993304

TS= 97.989884

■ TASTENBETÄTIGUNG

<Eingabe der Koordinaten X und Y jedes Eckpunkts>

1. DEF A

Numbers=_

2. 4 ENTER

Point 1x=

?

3. 4 ENTER

Point 1y=

?

⋮

Gleiche Eingabe wie oben.

⋮

4. 20 ENTER

>

<Eingabe der Eckpunktnummern jedes Dreiecks>

1. DEF B

Point=_

2. 1 ENTER

Point=_

3. 2 ENTER

Point=_

4. 3 ENTER

Point=_

5. 1 ENTER

Point=_

6. 3 ENTER

Point=_

7. 4 ENTER

Point=_

8. ENTER

>

<Löschen der Gesamtsumme der Dreieckflächen>

1. DEF C

** TS CLEAR **

>

■ PROGRAMMLISTING

```

10:"A":USING :CLEAR :
  WAIT 0
20:INPUT "Numbers=";N
30:IF N<1 BEEP 2:GOTO 2
  0
40:DIM X(N-1),Y(N-1),B$
  (0)
50:FOR I=0 TO N-1
60:B$(0)="x=":GOSUB 360
70:INPUT X(I):B$(0)="x="
  "+STR$ X(I):GOSUB 3
  70:GOTO 90
80:N=I:END
90:B$(0)="y=":GOSUB 360
  :INPUT Y(I)
100:B$(0)="y=" +STR$ Y(I
  ):GOSUB 370:NEXT I
110:BEEP 1:END
120:"B":USING :INPUT "Po
  int=";0,"Point=";P,"
  Point=";Q:GOTO 140
130:GOTO 310
140:IF (0<1)+(0>N)+(P<1)
  +(P>N)+(Q<1)+(Q>N)<>
  0 GOTO 120
150:C=X(0-1):D=Y(0-1):E=
  X(P-1):F=Y(P-1):G=X(
  Q-1):H=Y(Q-1)
160:X=E-C:Y=F-D:GOSUB 33
  0
170:A=X:X=G-E:Y=H-F:
  GOSUB 330
180:B=X:X=C-G:Y=D-H:
  GOSUB 330
190:C=X:IF A>X LET X=A
200:IF B>X LET X=B
210:I=(A+B+C)/2:S=SQR (I
  *(I-A)*(I-B)*(I-C))
220:J= INT ((2*S/X)*10^3
  )/10^3:L=X:GOSUB 340
230:X=L:S=X*J/2:K=K+S:L=
  A:GOSUB 340
240:A=L:L=B:GOSUB 340
250:B=L:L=C:GOSUB 340
260:C=L:L=S:GOSUB 350
270:S=L:L=K:GOSUB 350
280:K=L:LPRINT " ";STR$
  0+"-"+STR$ P+"-"+
  STR$ Q
290:LPRINT "A=" ; USING
  "#####.###";A
300:LPRINT "B=" ;B:
  LPRINT "C=" ;C:
  LPRINT "H=" ;J:
  LPRINT USING "#####
  #####.#####";"S="
  ;S:GOTO 120
310:LPRINT "":LPRINT "*T
  S*="; USING "#####
  ##.#####";K:END
320:"C":K=0:USING :PAUSE
  " ** TS CLEAR **":
  END
330:X=SQR (X*X+Y*Y):
  RETURN
340:L= INT (L*1000+.5)/1
  000:RETURN
350:L= INT (L*1000000)/1
  000000:RETURN
360:PAUSE "Point ";STR$
  (I+1);B$(0):RETURN
370:LPRINT "Point ";STR$
  (I+1);B$(0):RETURN

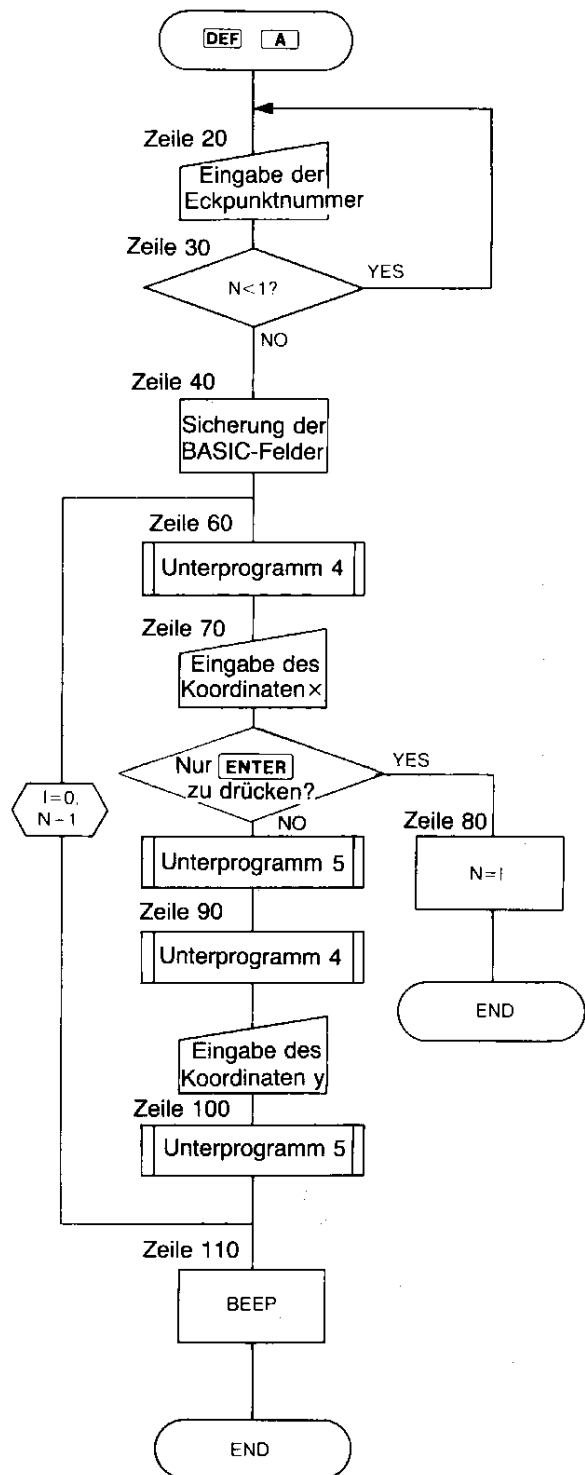
```

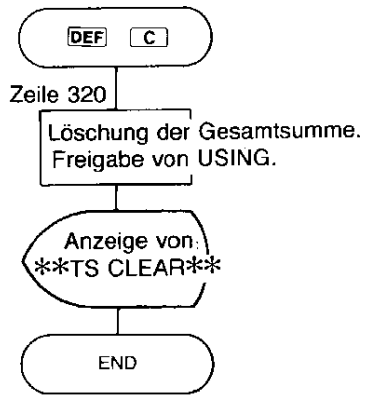
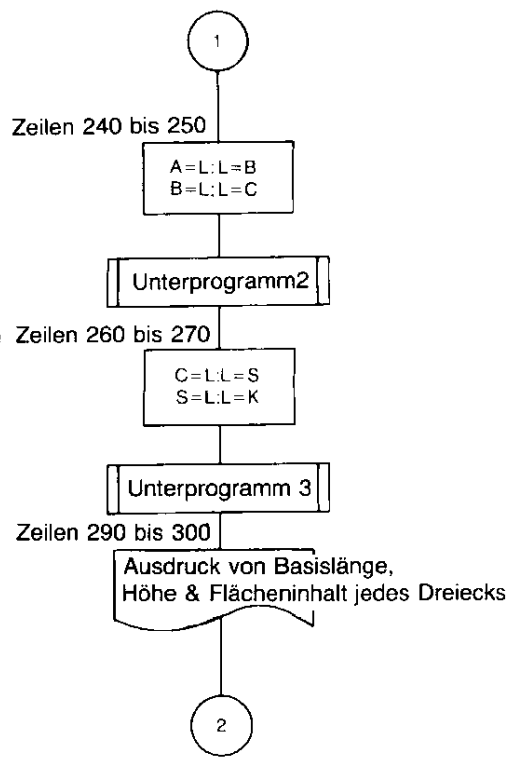
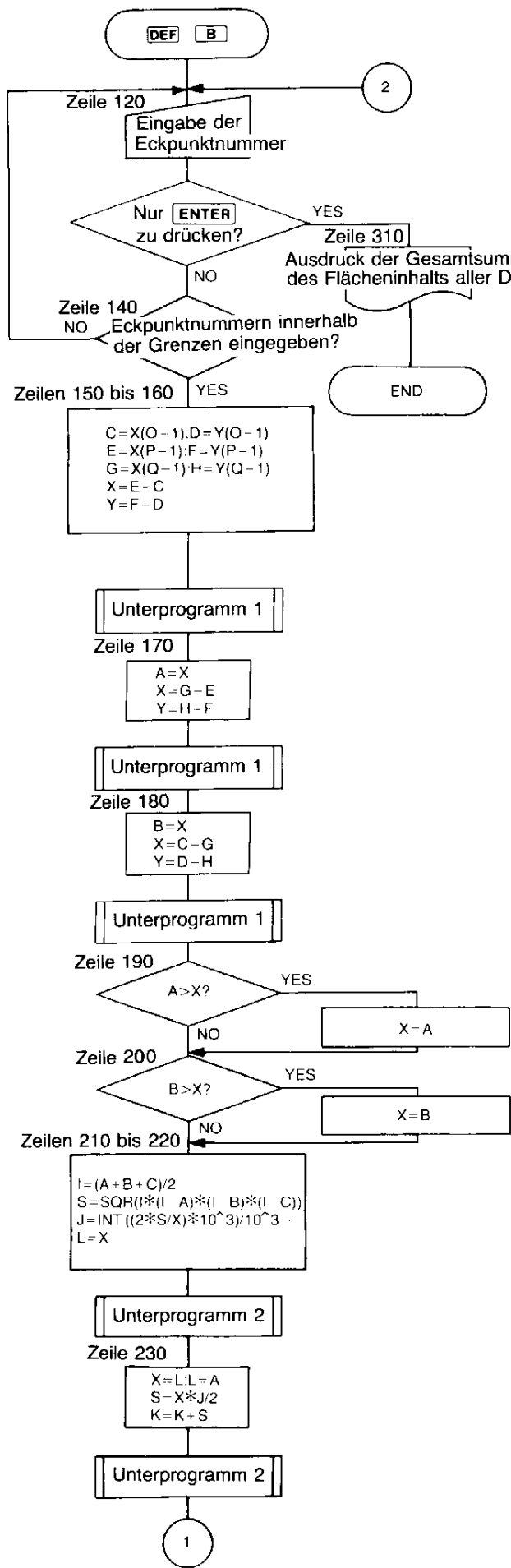
962 Bytes

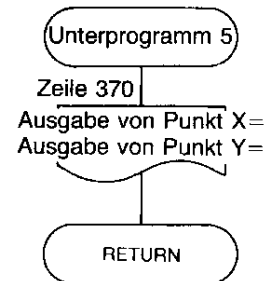
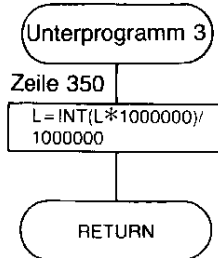
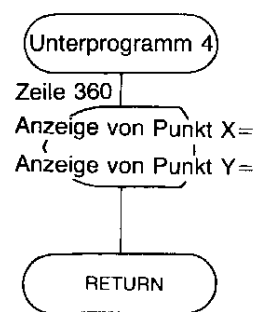
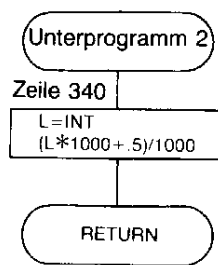
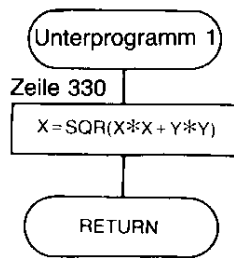
■ SPEICHERINHALT

A	<i>a</i>
B	<i>b</i>
C	x_1, \checkmark
D	y_1
E	x_2
F	y_2
G	x_3
H	y_3
I	S
J	<i>h</i>
K	Σs
L	\checkmark
N	Anzahl der Eckpunkte
O	\checkmark
P	\checkmark
Q	\checkmark
S	S
X	\checkmark
Y	\checkmark
X(N-1)	<i>x</i> -Koordinaten
Y(N-1)	<i>y</i> -Koordinaten
B\$(0)	\checkmark

■ ABLAUFDIAGRAMM





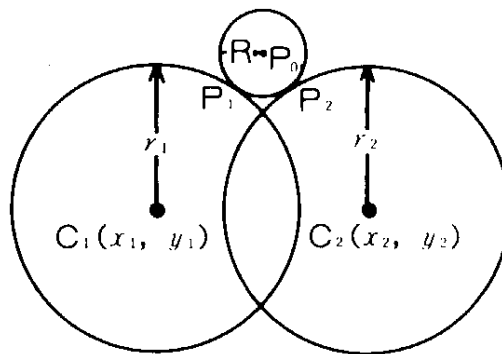


PROGRAMMTITEL: Anschmiegen eines Kreises an zwei Kreises

Es sind zwei sich einander berührende Kreise vorhanden und ein dritter, der sich an diese beiden anschmiegt. Wird hieraus enge Zuneigung entstehen? Eine solche Anschauung lockert das Studium der Geometrie etwas auf. Dieses Programm berechnet die Mittelpunktkoordinaten eines Kreises, der sich an zwei Kreise anschmiegt und die Koordinaten der Tangentpunkte der beiden Kreise.

■ BEDIENUNG

1. **DEF** **A** drücken. (Programmstart)
2. Entsprechend der Abfrage auf dem Display die Mittelpunktkoordinaten x_1 und y_1 des Kreises C_1 und Radius r_1 , die Mittelpunktkoordinaten x_2 und y_2 von Kreis C_2 und Radius r_2 und den Radius R des Kreises, der sich an die Kreise C_1 und C_2 anschmiegt eingeben.

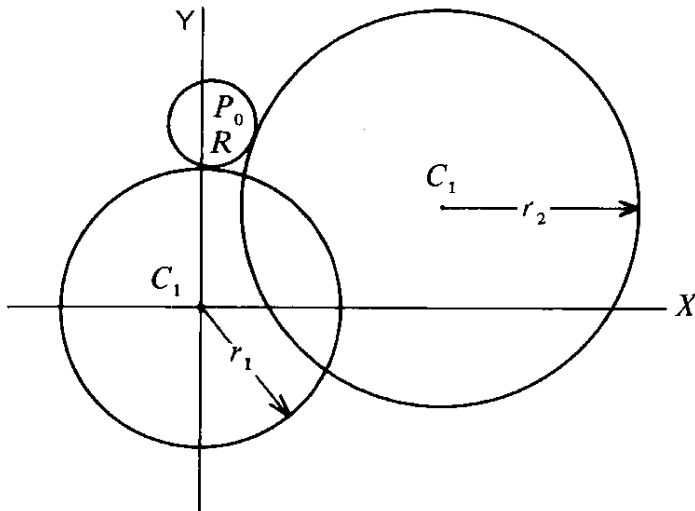


3. Anschließend die Daten für die Unterscheidungskriterien eingeben:

(1)	Wenn der Kreis extern Tangente zu Kreis C_1 ist.	1
	Wenn der Kreis intern Tangente zu Kreis C_1 ist.	-1
(2)	Wenn der Kreis extern Tangente zu Kreis C_2 ist.	1
	Wenn der Kreis intern Tangente zu Kreis C_2 ist.	-1
(3)	Wenn sich der Kreis an der linken Seite befindet, gesehen von der Mitte von Kreis C_1 auf Kreis C_2 .	1
	Wenn sich der Kreis an der rechten Seite befindet, gesehen von der Mitte von Kreis C_1 auf Kreis C_2 .	-1

Nach Eingabe der Daten werden die Mittelpunktkoordinaten des Kreises, der sich an C_1 und C_2 anschmiegt, und die Koordinaten der Tangentpunkte P_1 und P_2 auf dem Display angezeigt. Das Programm ist damit beendet.

■ BEISPIEL



$$C_1: x_1 = 0, y_1 = 0, r_1 = 30$$

$$C_2: x_2 = 50, y_2 = 20, r_2 = 40$$

$$R = 10$$

Unterscheidungskriterien:

- (1) 1 (Tangente extern)
- (2) 1 (Tangente extern)
- (3) 1 (linke Seite)

■ TASTENBETÄTIGUNG

1. **DEF** **A**

C1 $x = _$

2. **0** **ENTER**

C1 $y = _$

3. **0** **ENTER**

C1 $r = _$

⋮

Gleiche Eingabe wie oben.

⋮

4. **10** **ENTER**

C1 . OUT : 1 IN : -1 = _

5. **1** **ENTER**

C2 . OUT : 1 IN : -1 = _

6. **1** **ENTER**

LEFT : 1 RIGHT : 1 - 1 = _

7. **1** **ENTER**

P0 $x =$ **4.08**

8. **ENTER**

P0 $y =$ **39.79**

9. **ENTER**

P1 $x =$ **3.06**

10. **ENTER**

P1 $y =$ **29.84**

11. **ENTER**

P2 $x =$ **13.27**

12. **ENTER**

P2 $y =$ **35.83**

13. **ENTER**

>

■ PROGRAMMLISTING

```

10:"A":DEGREE :INPUT "C
  1 x=";A
20:INPUT "C1 y=";B
30:INPUT "C1 r=";D
40:INPUT "C2 x=";D
50:INPUT "C2 y=";E
60:INPUT "C2 r=";P
70:INPUT "R=";S
80:INPUT "C1.OUT:1 IN:-
  1=";Q
90:INPUT "C2.OUT:1 IN:-
  1=";R
100:INPUT "LEFT:1 RIGHT:
  -1=";G
110:F=P+R*S:C=O+Q*S:H=D-
  A:I=E-B:J=SQR (H*H+I
  *I):K=ACS (H/J):IF 0
  >I LET K=-K
120:L=ACS ((C*C+J*J-F*F)
  /2/C/J)
130:N=K+G*L:M=A+C*COS N:
  N=B+C*SIN N:X=Q*(A-M
  ):Y=Q*(B-N):GOSUB 24
  0
140:IF ((Q=-1)*(S>0))=1
  LET W=W+180
150:H=M+S*COS W:I=N+S*
  SIN W:X=R*(D-M):Y=R*
  (E-N):GOSUB 240
160:IF ((R=-1)*(S>P))=1
  LET W=W+180
170:J=M+S*COS W:K=N+S*
  SIN W
180:M=M+SGN M*.005:N=N+
  SGN N*.005
190:H=H+SGN H*.005:I=I+
  SGN I*.005
200:J=J+SGN J*.005:K=K+
  SGN K*.005
210:PRINT "P0 x="; USING
  "#####.##";M:
  PRINT "P0 y=";N

```

```

220:PRINT "P1 x=";H:
  PRINT "P1 y=";I
230:PRINT "P2 x=";J:
  PRINT "P2 y=";K:END
240:W=ACS (X/SQR (X*X+Y*
  Y)):IF 0>Y LET W=-W
250:RETURN

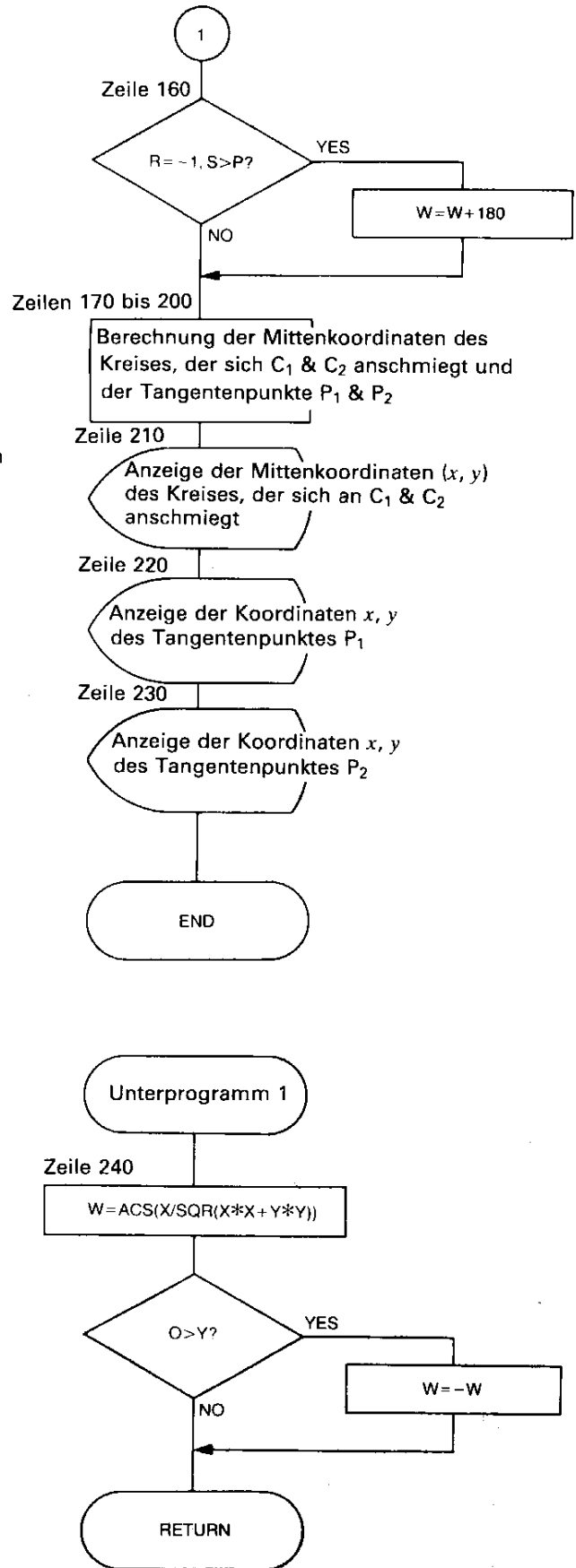
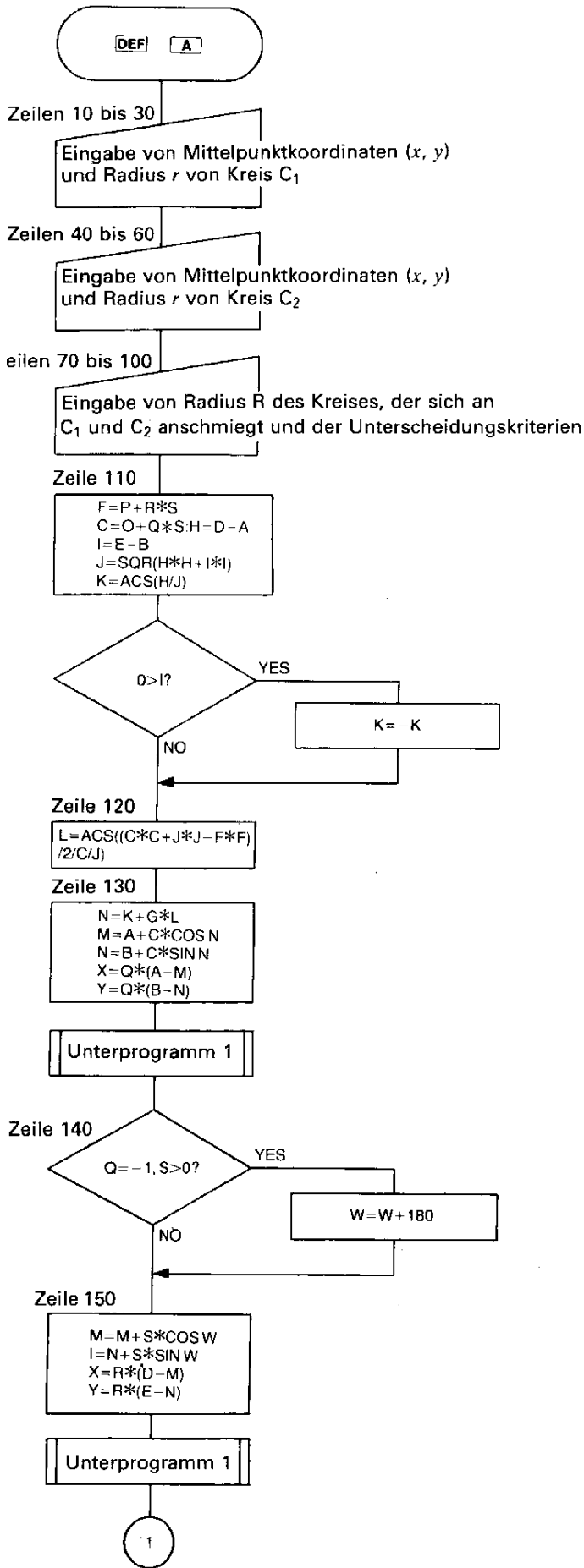
```

653 Bytes

■ SPEICHERINHALT

A	x_1
B	y_1
C	✓
D	x_2
E	y_2
F	✓
G	Unterscheidungskriterium 3
H	$P_1 x$
I	$P_1 y$
J	$P_2 x$
K	$P_2 y$
L	✓
M	$P_0 x$
N	$P_0 y$
O	r_1
P	r_2
Q	Unterscheidungskriterium 1
R	Unterscheidungskriterium 2
S	R
W	✓
X	✓
Y	✓

■ ABLAUFDIAGRAMM



PROGRAMMTITEL: Zahlenspiel

In diesem Programm werden dreistellige Zufallszahlen erzeugt, die erraten werden müssen. Statt anstrengender Programme und Studien für die nächste Prüfung versuchen Sie es zur Abwechslung einmal hiermit. Es geht darum, mit möglichst wenigen Versuchen einen Treffer zu erzielen.

■ BEDIENUNG

1. **DEF** **A** (Programmstart)
2. Auf dem Display wird "X=" angezeigt. Geben Sie jetzt eine dreistellige Zahl ein, von der Sie meinen, daß sie der Computer erzeugt haben könnte. Auf dem Display wird dann die Anzahl der bisherigen Versuche und die von Ihnen eingegebene dreistellige Zahl zusammen mit einem Kommentar angezeigt (etwa 1 Sekunde später).

Beispiel:

- Kommentar: 1 1

Erscheint obige Anzeige, bedeutet die erste Stelle (1) nach dem Kommentar, daß eine Stelle der von Ihnen eingegebenen Zahl der erzeugten Zufallszahl hinsichtlich von Position und Wert entspricht. Die zweite Stelle (1) zeigt, daß eine weitere Stelle der eingegebenen Zahl hinsichtlich des Werts richtig ist, aber die Position dieser Stelle falsch ist.

- Kommentar: 3 0

Diese Anzeige bedeutet, daß die von Ihnen eingegebene Zahl hinsichtlich der Stellen als auch der Position der Stellen richtig ist. Nach einem Treffer wie diesem erscheint auf dem Display die Anzeige "VERY GOOD!" sowie die Anzahl der bisherigen Versuche.

Achtung: Es können nur dreistellige Zahlen eingegeben werden.

■ TASTENBETÄTIGUNG

1. DEF A

X=_

2. 123 ENTER

1 123

Comment:0 1

X=_

3. 145 ENTER

2 145

Comment:1 0

X=_

:

Gleiche Eingabe wie oben.

:

4. 305 ENTER

6 305

Comment:3 0

VERY GOOD! 6

>

■ PROGRAMMLISTING

```

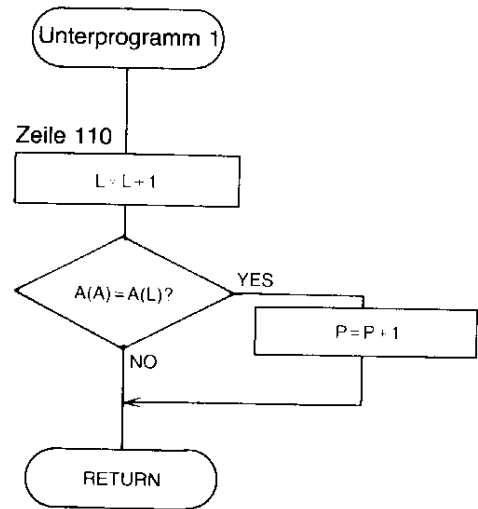
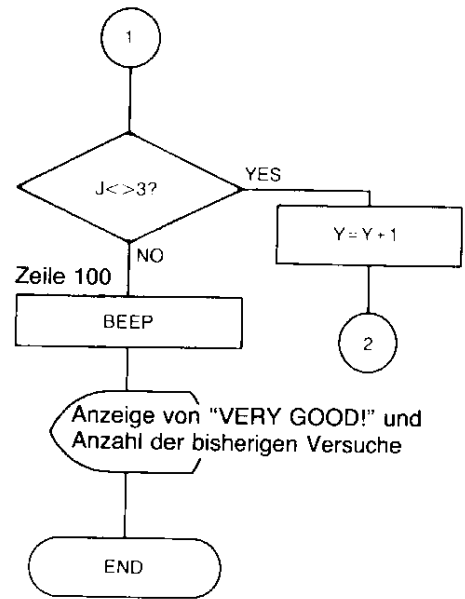
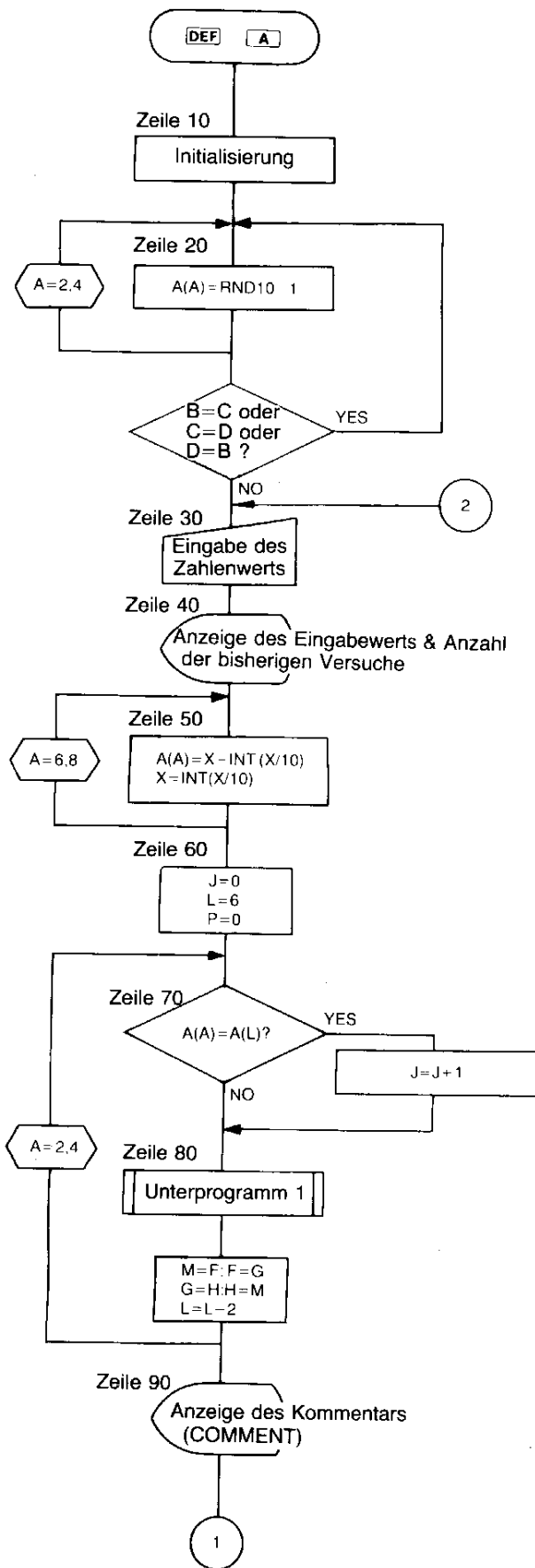
10:"A":CLEAR :RANDOM :Y
   =1
20:FOR A=2 TO 4:A(A)=
   RND 10-1:NEXT A:IF (
   B=C)+(C=D)+(D=B)<>0
   THEN 20
30:BEEP 1:INPUT "X=";X
40:USING :PAUSE Y,X
50:FOR A=6 TO 8:A(A)=X-
   INT (X/10)*10:X= INT
   (X/10):NEXT A
60:J=0:L=6:P=0
70:FOR A=2 TO 4:IF A(A)
   =A(L) LET J=J+1
80:GOSUB 110:GOSUB 110:
   M=F:F=G:G=H:H=M:L=L-
   2:NEXT A
90:PAUSE "Comment:";
   USING "####";J;P:IF
   J<>3 LET Y=Y+1:GOTO
   30
100:BEEP 2:PRINT "VERY G
   OOD ! ";Y:END
110:L=L+1:IF A(A)=A(L)
   LET P=P+1
120:RETURN
  
```

309 Bytes

■ SPEICHERINHALT

A	✓
B	✓
C	3-stellige Zahl
D	✓
F	✓
G	✓
H	✓
J	Kommentar
L	✓
M	✓
P	Kommentar
X	Eingabewert
Y	Anzahl der Versuche

■ ABLAUFDIAGRAMM



ANHANG I. INDEX

ABS	184
ACS	184
Addition	51
AHC	184
AHS	185
AHT	185
AND	97
AREAD	135
Anzeige	22
ASC	191
ASN	185
ATN	185
Auf-/Abrundung	44
Ausschalten	18
BEEP	137
Betriebstart	23
Betriebshinweise	9
C.CE Key	43
CHAIN	138
CHR\$	191
CLEAR	140
CLOAD	119
CLOAD?	120
CONT	121
COS	185
CSAVE	122
CUR	186
DATA	141
Datenspeicherung	29
DEGREE/GRAD/RADIAN	142, 150, 169
DEG	186
DIM	143
DELETE	123
DMS	186
Definable Keys	19, 210
Division	51
Drucker	25
END	145
Editierfunktionen	86
Einfache Variable	102
Einschalten	18

EXP	186
Exponentialfunktionen	58
FACT	186
Fakultät	58
Fehlermeldung/Fehlersuche	116
Feldvariable	105
FOR...TO...STEP	146
GOSUB	148
GOTO	124, 149
HCS	187
Hexadezimalumwandlung	47
Hexadezimalzahlen	47
HSN	187
HTN	187
Hyperbel- und inverse Hyperbelfunktionen.....	57
IF...THEN	151
Indizierte Variable	102
INKEY\$	181
INPUT	152
INPUT #	154
INT	187
Kassetten-Interface	28
Koordinatenumwandlung	47
LEFT\$	192
LEN	192
LET	157
LIST	125
LLIST	126
LN	187
LOG	187
LPRINT	158
Logarithmische Funktionen	58
Löschen einer Zeile	115
Löschen der Eingabe oder einer Fehlermeldung.....	43
Matrizen	69
MDF	159
MEM	182
MERGE	127
MID\$	192
Multiplikation	51
NEW	130
NEXT	160
Nachkommastellen	44

NOT	98
Numerische Feldvariable	105
ON...GOSUB	161
ON...GOTO	162
OR	97
PASS	131
PAUSE	163
PI	183
PRINT	165
PRINT #	167
Programmaufbau	112
Programmausführung	115
Programmeingabe	112
Programmerstellung	111
POL	188
Potenzfunktion	52
RADIAN.....	169
RANDOM	170
RCP	188
READ	171
REC	188
Rechenbereich	65
Rechengenauigkeit	84
REM	172
RENUM	133
RESTORE	173
RETURN	174
Reziprok-Rechnung	53
RIGHT\$.....	192
RND	188
ROT	189
RUN/GOTO	132
Schlüsselwörter, vorprogrammierte	86
SGN	189
SIN	189
SQR	190
SQU	190
Speicherrechnung	53
Statistische Berechnungen	59
STOP	175
STR\$	192
Stromversorgung	11
Subtraktion	51

TAN	190
Tastaturabdeckung	15
Tastenfunktionen	209
TEN	190
Textausdruck	94
Textfeldvariable	106
Trigonometrische Funktionen und Umkehrfunktion	57
TROFF	176
TRON	177
USING	178
Vergleichsausdrücke, logische	95
VAL	193
Variable A	103
Variablen	100
WAIT	180
Winklumwandlung	46
Wissenschaftliche Schreibweise	45, 85
Wurzelfunktion	52

SHARP CORPORATION

OSAKA, JAPAN

1986 © SHARP CORPORATION
PRINTED IN JAPAN/IMPRIMÉ AU JAPON
0B2T(TINSG1067ECZZ)⑤