

SHARP®

SCHENCOMPUTER

MODELL

PC-1280

BEDIENUNGSANLEITUNG



INHALTSVERZEICHNIS

EINLEITUNG

WIE MAN DIESES HANDBUCH BENUTZT

KAPITEL 1 HARDWARE

1	HARDWARE: ÜBERSICHT	2
2	HARDWARE: IM DETAIL	4
	Ein- und Ausschalten (ON/OFF)	4
	Automatische Abschaltung	5
	Die Tastatur	5
	Das Display	10
	Rücksetzen	12
	Einsetzen der Batterien	14
3	RAM-KARTEN	17
	Anzahl der Dateien	18
	Einlegen einer RAM-Karte	18
	Herausnahme einer RAM-Karte	19
	Anwendung der RAM-Karte	19
	Vorsichtsmaßnahmen bei der Verwendung der RAM-Karte	19
	Kopieren von RAM-Karten	20
	Verwendung von RAM-Karten aus anderen Computern	22
4	PERIPHERE GERÄTE	23
	Diskettenlaufwerk in Taschenformat CE-140F	24
	CE-126P Thermodrucker/Kassetten Interface	24
	CE-124 Kassetten Interface	25
	Benutzung des CE-126P Drucker/Kassetten Interface	26
	Benutzung des Druckers	26
	Anwendung des Kassetten Interface	27
	Überprüfen eines gesicherten Programms	28
	Laden von einem Magnetband	29
	Bandhinweise	30

KAPITEL 2 DIREKT-EINGABE BETRIEB

5	BEDIENUNG: EINE FRAGE DER EINSTELLUNG	34
	Das Ansteuern von Betriebsarten	35

6	RECHENBETRIEB	36
	Beginn des Rechenbetriebs	36
7	RUN-BETRIEB	40
	Einstellen des RUN-Betriebs	40
	Einige nützliche Hinweise	40
	Einfache Rechenoperationen	42
	Wiederabruf von Eingaben	43
	Fehler	48
	Verkettung von Rechenoperationen	49
	Negative Zahlen	51
	Komplexere Rechenoperationen und Klammersetzung	51
	Die Anwendung von Variablen in Rechenoperationen	52
	Mehrere Rechenschritte	54
	Wissenschaftliche Notation	56
	Einfache Genauigkeit, Doppelte Genauigkeit	59
	Einstellen des doppelten Genauigkeits-Modus	60
	Löschen des Modus der doppelten Genauigkeit	60
	Werte-Schablone	60
	Maximale Rechenlänge	62
	Wissenschaftliche Rechenoperationen	62
	Rangfolge bei manuellen Berechnungen	65
	Ausdrucken von Rechenergebnissen	66
	Rechenfehler	66

KAPITEL 3 PROGRAMMIER-BEDIENUNG

8	BEGRIFFE UND AUSDRÜCKE DES BASIC	70
	Zeichenfolgen-Konstanten	70
	Hexadezimalzahlen	71
	Variablen	71
	Vorgegebene Variablen	73
	Einfache Variablen	74
	Feld-Variablen	75
	Variablen der Form von A (.)	79
	Doppelt genaue Variable	82
	Eingabe von Werten mit doppelt genauen Variablen	83
	Gemischte einfache- und doppelt genaue Variablen	84
	Programme und Dateien	86
	Dateinamen	86

Erweiterung	87
Gerätename	87
Datei-Nummern	88
Arbeitskommandos für RAM- und Taschen-Disketten basierte Dateien	88
Daten-Dateien	89
Erstellen einer sequentiellen Datei	90
Das Lesen von Daten aus einer sequentiellen Datei	91
Hinzufügen von Daten in eine sequentiellen Datei	92
Ausdrücke	93
Numerische Operatoren	93
Zeichenfolge-Ausdrücke	94
Verhältnis-Ausdrücke	94
Logische Ausdrücke	95
Klammerung und Vorrang der Operatoren	97
RUN-Modus	98
Funktionen	98

9	PROGRAMMIEREN DES PC-1280	99
	Programme	99
	BASIC-Anweisungen	99
	Zeilennummern	99
	BASIC-Kommandos	100
	Direkt-Kommandos	101
	Modi	101
	Der Anfang des Programmierens mit dem PC-1280	102
	Das Speichern von Programmen im PC-1280	110
	Abkürzungen	111
	Die DEF -Taste und benannte Programme	111

10	RESERVE-Modus	113
----	---------------------	-----

11	FEHLERSUCHE	117
----	-------------------	-----

KAPITEL 4 COMPUTER-ANWEISUNGEN

12	WISSENSCHAFTLICHE UND MATHEMATISCHE BERECHNUNGEN	124
	Rechenbereiche	137

13	COMPUTER-ANWEISUNGEN	139
----	----------------------------	-----

KAPITEL 5 ANHANG

A	FEHLERMELDUNGEN	288
B	ZEICHENCODE-TABELLE	292
C	FORMATIERUNG DER DATENAUSGABE	294
	Numerische Masken	295
	Bestimmung der Anzahl von Stellen	295
	Bestimmung des Dezimalpunkts	296
	Bestimmung der wissenschaftliche Notation	297
	Bestimmung alphanumerischer Masken	298
	Gemischte Masken	298
D	AUSDRÜCKEN UND VORRANG VON OPERATIONEN	299
	Vorrang von Operatoren	299
	Beispiel für eine Bewertungsfolge	300
E	TASTENFUNKTIONEN IM BASIC	302
F	FEHLERSUCHE	308
G	TECHNISCHE DATEN	309
H	BENUTZUNG VON PROGRAMMEN ANDERER COMPUTER	311
	DIE VERWENDUNG VON RAM-KARTEN VON ANDEREN COMPUTERN IM PC-1280:	311
	DIE ANWENDUNG VON RAM-KARTEN AUS DEM PC-1280 IN ANDEREN COMPUTERN:	313
I	PFLEGE DES PC-1280	314
	PROGRAMMBEISPIELE	315
	INDEX	341

EINLEITUNG

Willkommen in der Welt der SHARP-Besitzer!

Nur wenige Industriezweige der Welt können heute Schritt halten mit dem schnellen Wachstum und dem technischen Fortschritt im Bereich der Personalcomputer. Computer, die noch vor kurzer Zeit einen ganzen Saal gefüllt hätten, deren Programmierung einen akademischen Abschluß erfordert und die Tausende von Dollars gekostet hätten, passen heute in Ihre Handfläche, sind einfach zu programmieren und kosten so wenig, daß fast jeder sie sich leisten kann.

Ihr neuer SHARP PC-1280 wurde entwickelt, um Ihnen den allerneuesten Stand dieser Computerrevolution zugänglich zu machen und verfügt über eine Vielzahl der fortschrittlichsten Fähigkeiten:

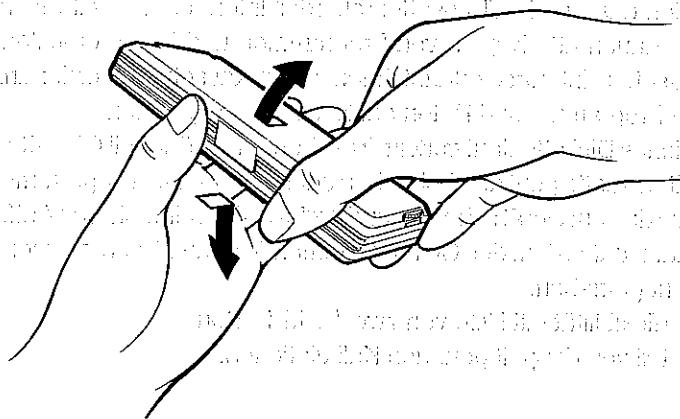
- **SPEICHERSCHUTZ** — der PC-1280 merkt sich gespeicherte Programme und Variable, selbst wenn Sie ihn abschalten.
- **Stromversorgung durch Batterien**, ideal für ein wirklich portables System.
- **AUTOMATISCHE ABSCHALTUNG**, die die Batterien schützt, in dem die Stromversorgung abgeschaltet wird, wenn innerhalb eines bestimmten Zeitraums keine Operation durchgeführt wird.
- Eine erweiterte Version des BASIC, die formatierte Datenausgabe ermöglicht, weiterhin zweidimensionale Felder, variable Zeichenfolge-Länge, Programmverkettung und viele andere fortschrittliche Arbeitsmöglichkeiten.
- Als Option das Drucker/Kassettenrekorder-Interface (Modell CE-126P), mit dem Sie dauerhafte Kopien von Programmen und Daten erstellen können. Das Kassetten-Interface erlaubt Ihnen, einen Kassettenrekorder anzuschließen, um Programme und Daten auf Kassette zu speichern.
- Zusätzliches Diskettenlaufwerk im Taschenformat (Modell CE-140F). Verglichen mit Kassette geht das Schreiben oder Lesen eines Programms oder von Daten mit diesem Gerät schneller und leichter. Zum einfacheren Auffinden von Daten kann außerdem das COPY-Kommando benutzt werden. Der CE-140F ist batteriegetrieben.
- Speicherflexibilität mit Hilfe von zwei RAM-Karten.
- Den Vorteil von doppelt genauen Kalkulationen.

Wir gratulieren Ihnen zu Ihrem Eintritt in eine aufregende und phaszinierende neue Welt. Wir sind sicher, daß Sie Ihre neue Errungenschaft schon bald als eine der klügsten ansehen werden. Der PC-1280 ist ein hilfreiches Werkzeug, das entwickelt wurde, um Ihren spezifischen mathematischen, wissenschaftlichen, technischen, kaufmännischen und persönlichen Computerbedürfnissen gerecht zu werden. Mit dem SHARP PC-1280 können Sie JETZT damit anfangen, die Lösungen zu finden, die Sie morgen brauchen werden!

Öffnung des Schutzgehäuses

Das Schutzgehäuse Ihres PC-1280 wurde exakt gearbeitet, um dessen LCD-Display und Tastatur zu schützen.

Falls neu, läßt sich das Schutzgehäuse unter Umständen nur unter Schwierigkeiten öffnen. Den PC-1280 in der linken Hand halten, dabei den Daumen auf die Vorderseite der unteren Gehäusehälfte setzen und die anderen vier Finger auf der Rückseite der oberen Gehäusehälfte positionieren. Sanft drücken und mit dem rechten Daumen öffnen.



WIE MAN DIESES HANDBUCH BENUTZT

Dieses Handbuch soll Sie einführen in die Fähigkeiten und Charakteristika Ihres PC-1280 und es soll Ihnen als wertvolles Nachschlagewerk dienen. Es wurde in sechs Abschnitte eingeteilt, jeder mit einer Einleitung versehen, der einen speziellen Aspekt des PC-1280 behandelt. So ist das Handbuch letztlich ein Nachschlagewerk, und wir empfehlen Ihnen, die einleitenden Abschnitte im Kapitel 1 (Hardware) und Kapitel 2 (Operation) sorgfältig durchzulesen: Der PC-1280 ist ein äußerst wirkungsvolles Werkzeug mit vielen wertvollen und zeitsparenden Funktionen, deren Entdeckung sogar alte Computerhasen erfreuen dürfte!

Kapitel 1: Hardware

Der erste Abschnitt von Kapitel 1 ist eine Einführung in die speziellen Eigenschaften des PC-1280. Abschnitt 2 beschreibt die Grundausrüstung des Computers, die Bedienung der Tasten und die Bedeutungen der verschiedenen Displaysymbole. Die Lektüre von Abschnitt 1 und 2 ist deshalb äußerst wichtig. Die folgenden Abschnitte behandeln den Zusatzspeicher (RAM-Karten) und periphere Geräte (wie Drucker und Kassettenrekorder).

Kapitel 2: Direkte Eingabe-Operation

Das 2. Kapitel ist dem Gebrauch des PC-1280 als Rechner oder "Direkteingabe" Computer gewidmet. Direkte Eingabe bezieht sich auf die unabhängige Verwendung von BASIC-Begriffen (d.h. nicht innerhalb eines BASIC-Programms).

Kapitel 3: Programm Operation

Kapitel 3 führt Sie in die BASIC-Programmiersprache zur Anwendung des PC-1280 ein. Auch wenn Sie bereits früher BASIC-Programme durchgeführt haben, so empfehlen wir Ihnen doch diesen Teil trotzdem durchzulesen, da sich alle BASIC-Dialekte unterscheiden. Dieses Kapitel enthält auch zeitsparende Information in der Form von Programmkurzschlüssen und Techniken zur Fehlerbeseitigung.

Kapitel 4: Basis Adressen

Kapitel 4 enthält eine alphabetische Auflistung der numerischen Funktionen und BASIC-Begriffe, die zur Programmierung des PC-1280 gebraucht werden. Viele dieser Begriffe können im direkten Eingabemodus des PC-1280 benutzt werden.

Kapitel 5: Anhang

Kapitel 5 enthält in der Hauptsache Nachschlagematerialien, wie Code-Tabellen, Fehlerhinweise und Zusammenstellungen von Kenn-
daten. Außerdem finden Sie hier Hinweise, wie Sie Ihren PC-1280 funktionstüchtig halten.

Dieses Handbuch ist nicht als Lehrbuch zum Selbsterlernen konzipiert. Eine vollständige Beschreibung ist nicht die Absicht dieses Handbuches. Wenn Sie bisher noch nicht in BASIC programmiert haben, empfehlen wir Ihnen, daß Sie sich hierfür ein separates Lehrbuch zulegen oder an einem BASIC-Kurs teilnehmen, bevor Sie dieses Handbuch durcharbeiten.

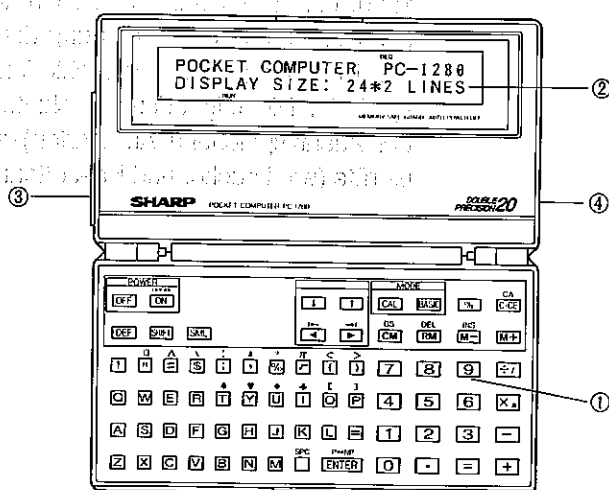
HARDWARE

Die erste Abschnitte von Kapitel 1 sind eine Einführung in die speziellen Eigenschaften des PC-1280, eine Beschreibung der Grundausrüstung des Computers, der Bedienung der Tasten und der Bedeutungen der verschiedenen Display-symbole. Die folgenden Abschnitte behandeln den Zusatzspeicher (RAM-Karten) und periphere Geräte (wie Drucker und Kassettenrekorder).

1. HARDWARE: ÜBERSICHT

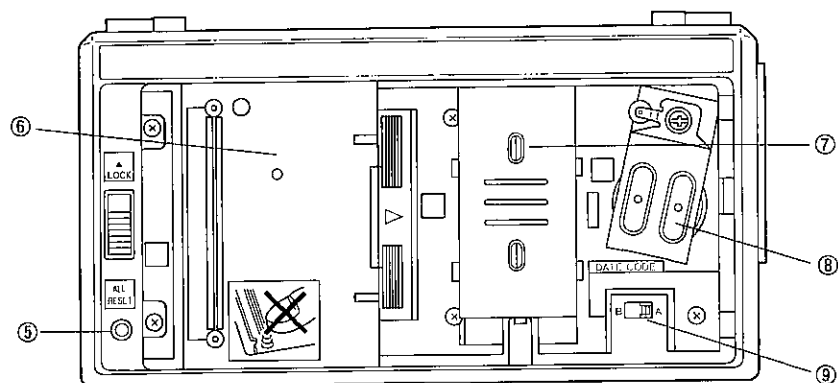
Die Ausstattung des SHARP PC-1280 besteht aus einer QWERTY-Stil-Tastatur, die der von herkömmlichen Schreibmaschinen ähnelt, einem Flüssig-Kristall-Display (LCD) mit einstellbarem Kontrast, einem Anschluß zur Speichererweiterung und aus einem 11-Stift-Interface-Stecker zum Anschluß verschiedener Zusatzgeräte. Der Computer mit seinem eingebauten 8K Bytes Speicher wird von einem robusten Doppelgehäuse geschützt, wenn er nicht in Gebrauch ist.

Auf den folgenden Seiten werden die einzelnen Teile des PC-1280 beschrieben, um Sie mit ihren Positionen und Funktionen bekannt zu machen.



Der PC-1280 von oben gesehen

- 1. Tastatur** Schreibmaschinentastatur und Zahlentastatur mit insgesamt 72 Tasten, einschließlich einer Anzahl Spezialtasten.
- 2. LCD Bildschirm** Der Bildschirm besteht aus zwei Zeilen mit je 24 Anschlägen. Der Kontrast der extra großen Zeichen ist zum besseren Sehen voll einstellbar.
- 3. 11-Stift-Stecker** Zum direkten Anschließen des PC-1280 an ein zusätzliches peripheres Gerät, wie etwa Plattenlaufwerk (CE-140F), Kassettenrekorder (CE-152) und Drucker (CE-126P) oder ein zusätzliches Interface-Gerät (CE-124).
- 4. LCD Kontrastregler** Hier kann der Bildschirm aufgehellert oder abgedunkelt werden und somit auf die passenden Sehbedingungen eingestellt werden.



Der PC-1280 von unten gesehen

- 5. Rücksetzschalter** Falls der PC-1280 aus irgend einem Grund während des Betriebs "bockt", arbeitet die Tastatur nicht. Auch kann es unmöglich sein, die **ON**- und **OFF**-Tasten zum erneuten Start des Computers zu betätigen. Drücken Sie den ALL RESET-Schalter zum Löschen des Speichers, damit der PC-1280 wieder in die Lage zum Stromeinschalten (Power ON) kommt. Benutzen Sie diesen Schalter mit Vorsicht, da Sie den Verlust von Daten und Programmen riskieren. Zur weiteren Information über den Gebrauch dieses Schalters lesen Sie Abschnitt 2.
- 6. RAM-Schlitz** In diesem Schlitz befinden sich die zusätzlichen RAM-Karten, die den Speicher-Arbeitsraum vergrößern. RAM-Karten verschiedenster Kapazität können hier eingeführt werden. Weitere Informationen, wie Sie diese RAM-Karten benutzen, finden Sie im Abschnitt 3.
- 7. Batteriegehäuse** In diesem Fach befinden sich die zwei Batteriezellen als Stromquelle für den PC-1280. Im Abschnitt 2 finden Sie Hinweise, wie Sie die Batterien erneuern.
- 8. Zellenfach für Sicherungsspeicher** In diesem Fach liegt eine Zelle zum Zurückgehen im eingebauten RAM-Speicher des PC-1280.
- 9. Speicherschutz-Schalter** (nur für eingebauten Speicher). Dieser Schalter schützt die Inhalte der einliegenden RAM-Karten beim Auswechseln der Batteriezellen. Bei Normalbetrieb steht der Schalter auf A. Wenn Sie die Batteriezellen auswechseln schalten Sie auf B.

2. HARDWARE: IM DETAIL

Ein- und Ausschalten (ON/OFF)

Die **ON**- und **OFF**-Tasten liegen auf der Tastatur oben links unter der POWER-Etikett. Wenn Sie zuerst die **ON**-Taste drücken oder immer wenn der Computer, nachdem Veränderungen vorgenommen wurden (wie die Erneuerung der Betriebszellen), eingeschaltet wird (ON), sieht das Display folgendermaßen aus:

MEM\$ = "2"
MEMORY ALL CLEAR O.K. ?

Zum Löschen des Arbeitsspeichers drücken Sie die **Y**-Taste. Der Cursor erscheint wie folgt auf dem Display:

> DEG
RUN

Beachten Sie die Symbole auf der Statuszeile oben und unten am Bildschirm. Diese zeigen die Betriebsart an, in der sich der Computer befindet und werden genauer etwas später in diesem Abschnitt beschrieben. Der Bildschirm hat das gleiche Aussehen, wenn Sie zum Ausschalten die **OFF**-Taste drücken, und die Symbole, die leuchten, geben den zur Zeit des Ausschaltens (OFF) befindlichen Status an.

Wurde der PC-1280 in der Kalkulator-Stellung ausgeschaltet (OFF), zeigt das Display beim Wiedereinschalten (ON) folgendes an:

DEG 0.
CAL

Schalten Sie den PC-1280 aus (OFF), wird das Display gelöscht, aber der Computer erinnert sich an alle Programme und Einstellungen der Reservetasten, wenn der Computer ausgeschaltet (OFF) wurde. Alle diese Einstellungen sind noch in Funktion, wenn Sie den PC-1280 wieder einschalten (ON).

Wenn Sie von Kassette mit dem CLOAD-Kommando ein Programm eingeben, drücken Sie die **ON**-Taste zum stoppen des Bandes und dann die **OFF**-Taste.

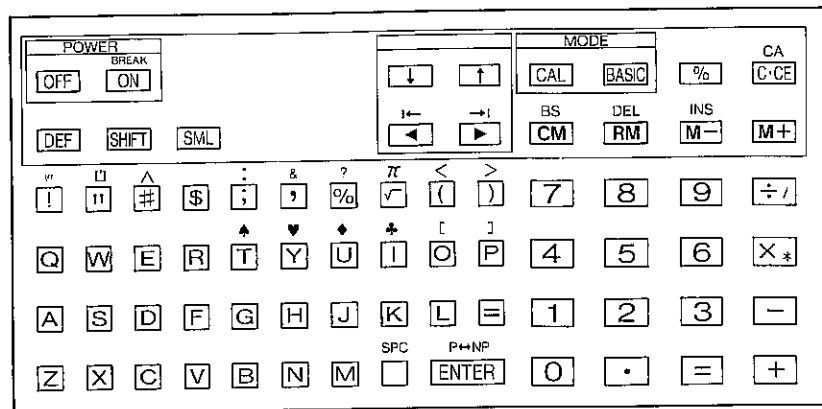
Automatische Abschaltung

Zum Schutz der Batterien schaltet sich der PC-1280 automatisch selbst aus, wenn ca. 11 Minuten lang keine Tasten betätigt werden. (Beachte, daß er sich nicht selbst ausschaltet (OFF), wenn ein Programm läuft.)

Zum erneuten Start des PC-1280 drücken Sie die **ON**-Taste, wenn sich der Computer selbst ausgeschaltet hat.

Die Tastatur

Der PC-1280 hat eine Tastatur mit der gleichen Anordnung wie gewöhnliche Schreibmaschinen (bekannt als QWERTY-Tastatur, auf Grund der Anordnung der Tasten). Daneben befindet sich eine Nummerntastatur in bequemer Rechneranordnung und verschiedene Tasten zu Sonderzwecken.



OFF

Zum Ausschalten des PC-1280 betätigen Sie die **OFF**-Taste zum Schutz der Batterien, wenn der Rechner nicht in Gebrauch ist.

BREAK Zum ersten Einschalten des PC-1280 drücken Sie die **ON**-Taste. **ON** ebenfalls zum Wiedereinschalten des PC-1280, wenn die AUTO OFF-Funktion (die Automatische Abschaltung) den Rechner ausgeschaltet hat. Zum Abbrechen eines Programms oder zum Stoppen der Eingabe eines Programms von der Kassette unter Verwendung des CLOAD-Kommandos drücken Sie diese Taste.

DEF Die **DEF**-Taste ist eine Spezialtaste, die zur Durchführung von BASIC-Programmen benutzt wird.

SHIFT Die **SHIFT**-Taste hat einige wichtige Funktionen. Erstens wird die **SHIFT**-Taste betätigt vor einer Taste mit einer oben in orange gedruckten Sekundärfunktion. Zweitens wird die **SHIFT**-Taste vor der **BASIC**-Taste gedrückt, um in den RSV-Betrieb (Reserve) zu kommen. Normalerweise die **BASIC**-Tastenkippschalter zwischen der RUN- und der PRO-Betriebsart (Programm) drücken; mit der **SHIFT**-Taste ist eine dritte Betriebsart möglich. Drittens drückt man die **SHIFT**-Taste und eine Taste unter der eine Zeichenfolge im RESERVE-Modus eingespeichert wurde, um diese Zeichenfolge abzurufen.

SML Diese Taste wird gedrückt, um in Kleinbuchstaben schreiben zu können. Ein einmaliger Druck schaltet den PC-1280 in den LOWER CASE-Modus (Kleinbuchstaben) (SML wird auf der Statuszeile im Display angezeigt). Ein erneuter Druck auf diese Taste bringt den PC-1280 wieder in den UPPER CASE-Modus (Großbuchstaben) (SML verschwindet).

A ~ Z Der SHARP PC-1280 hat die gleiche 26-Buchstaben Einteilung wie die meisten Schreibmaschinen. Normal werden Großbuchstaben geschrieben (im Gegensatz zu den meisten Schreibmaschinen), was bequem ist, da der PC-1280 Angaben und Kommandos nur in Großbuchstaben erkennt. Zur Eingabe von Kleinbuchstaben bringt man den PC-1280 durch Druck auf die **SML**-Taste in die LOWER CASE-Betriebsart (Kleinbuchstaben).

= Gleichheitszeichen. Beide, graue und schwarze **=**-Tasten können im Rechenbetrieb (CAL) zum Anzeigen des Endes einer Rechnung benutzt werden. In jeder anderen Betriebsart haben diese Tasten bei der BASIC-Programmierung eine Sonderfunktion. Im RUN- oder PRO-Betrieb drücken Sie die **ENTER**-Taste, und nicht **=**, um das Ende einer Rechnung anzuzeigen.

 Die Leertaste. Drücken Sie diese Taste, um den Cursor um einen Anschlag nach rechts zu bewegen. Befindet sich der Cursor über einem Zeichen, das gelöscht werden soll, drückt man die **SPC**-Taste zum Löschen des Zeichens.

 Diese Taste wird zur Eingabe des von Ihnen getippten Textes in den Computerspeicher betätigt. Die **ENTER**-Taste ähnelt der Schreibwerk-Rückführungs-Taste einer Schreibmaschine. **ENTER** muß gedrückt werden, bevor der PC-1280 auf die alphanumerische Eingabe von der Tastatur reagiert (ausgenommen der CAL-Modus (Rechenbetrieb)). Zum Kippen zwischen P (Drucker ON) und NP (Kein Drucker) drücken Sie **SHIFT** und **ENTER**.

 Diese Symbole befinden sich über der oberen Reihe der Alphatastatur. Sie werden in der RUN- und PRO-Betriebsart verwendet.

 Diese Tasten sind orange (die gleiche Farbe wie die **SHIFT**-Taste) und befinden sich auf den oben beschriebenen Tasten. Vor dem Tippen eines Symbols drücken Sie die **SHIFT**-Taste.

 Drücken Sie die **SHIFT**-Taste und dann die Alphabet-Taste (TYUIOP) unter dem Symbol.

 Pfeil nach unten. Diese Taste drücken Sie, um im Display die nächste Zeile sichtbar zu machen.

 Pfeil nach oben. Diese Taste drücken Sie, um im Display die vorausgehende Zeile sichtbar zu machen.

 Pfeil nach links. Diese Taste erlaubt Ihnen, den Cursor nach links zu bringen, ohne das getippte Zeichen zu löschen. Druck auf **SHIFT** und diese Taste bringt den Cursor um eine Tabulatoreinstellung (TAB) nach links.

 Pfeil nach rechts. Mit Hilfe dieser Taste können Sie den Cursor nach rechts bewegen, ohne dabei vorher eingegebene Zeichen zu löschen. Druck auf **SHIFT** und diese Taste bringt den Cursor um eine Tabulatoreinstellung nach rechts.

CAL Druck auf diese Taste bringt den PC-1280 in den Rechen-Modus. In dieser Stellung funktionieren nur die Nummern- und die Bedienungstasten (+, -, /, x, =, %, $\sqrt{\quad}$, M+, M-, RM, CM).

BASIC Nach dem Einschalten des PC-1280 sehen Sie auf der Statuszeile im Display die Betriebsart (CAL oder RUN, PRO oder RSV). Drücken Sie **BASIC**, um zwischen RUN und PROGRAM zu wählen und zu wechseln. Um in die RSV-Betriebsart zu kommen, drücken Sie **SHIFT** und dann **BASIC**. Zum Verlassen des RSV-Modus drücken Sie **BASIC** erneut. Alle Programme werden in der PRO-Betriebsart (Programm) geschrieben und redigiert, laufen aber im RUN-Modus. Der RSV-Betrieb wird zur Eingabe häufig verwendeter Funktionen auf einzelnen Tasten und Menüs verwendet, um zu helfen, diese Funktionen schnell zu identifizieren.

% Die Prozent-Taste wird zum Berechnen von Prozenten verwendet.

CA Die **C·CE**-Taste (Löschen und Eingabe Löschen) erlaubt Ihnen, das Display zu löschen. Zum Einstellen der CA-Funktion (vollständiges Löschen) wird die **SHIFT**- und dann die **C·CE**-Taste gedrückt. Der PC-1280 geht auf die Grundeinstellung zurück und das Display und der Speicherblock werden gelöscht.

HINWEIS: Die vollständige Löschfunktion löscht keine Programme, keine Reservespeicher und keine Variablen. Zum Löschen von Programmen tippt man im PROGRAM-Modus NEW ein und drückt **ENTER**. Zum Löschen des gesamten Reservespeichers tippt man im RESERVE-Modus NEW ein und drückt **ENTER**. Zum Löschen aller Variablen tippt man CLEAR ein und drückt **ENTER**, wodurch sie auf Null gestellt werden.

M+ Speicher-Plus-Taste (nur in der CAL-Betriebsart). Zum Hinzufügen des angezeigten Wertes in den Speicher wird diese Taste betätigt.

INS Speicher-Minus-Taste (nur in der CAL-Betriebsart). Zum Herausnehmen des angezeigten Wertes aus dem Speicher wird diese Taste gedrückt.

Drücken Sie diese Taste in der RUN-, PRO- und RSV-Betriebsart, wenn die Einschub-Funktion eingestellt werden soll und wenn Sie vor dem durch den Cursor angezeigten Zeichen eine freie Stelle schaffen wollen. Jetzt kann durch Drücken einer beliebigen Taste ein einzelnes Zeichen eingefügt werden.

DEL **RM** RM Speicher-Rückübertragungs-Taste (nur in der CAL-Betriebsart). Zum Anzeigen der Speicherinhalte auf dem Display wird diese Taste betätigt.

Drücken Sie diese Taste in der RUN-, PRO- und RSV-Betriebsart, wenn die DEL-Funktion eingestellt werden soll und das Zeichen unter dem Cursor gelöscht werden soll.

BS **CM** Speicherlöschtaste. Zum Löschen der Inhalte des Speichers drücken Sie diese Taste.

Zum Einstellen der Rückstell-Funktion drücken Sie diese Taste in der RUN-, PRO- und RSV-Betriebsart.

0 ~ **9** Zahlentasten. Die Anordnung dieser Tasten ähnelt denen eines normalen Rechners.

÷ / Divisionszeichen. Mit dieser Taste werden bei Berechnungen Divisionen durchgeführt.

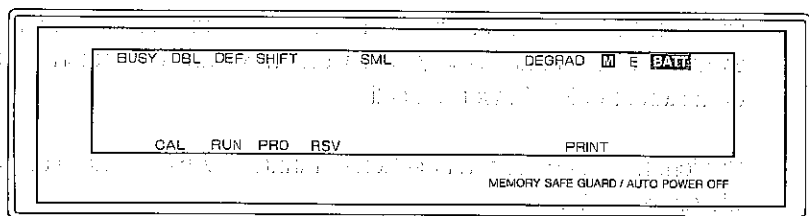
× * Multiplikationszeichen. Mit dieser Taste werden bei Berechnungen Multiplikationen durchgeführt.

− Minuszeichen. Mit dieser Taste werden Subtraktionen durchgeführt.

+ Additionszeichen. Mit dieser Taste werden Additionen durchgeführt.

= Gleichheitszeichen. Zum Anzeigen des Endes einer Rechnungseingabe im CAL-Modus.

Das Display



Das Display

Der PC-1280 hat ein 2 Zeilen Flüssigkristall-Display mit 24-Zeichen Punktmatrix. Jedes Zeichen besetzt eine 5x7 Punktmatrix.

Das Display besteht aus:

- > Bereitschaftssymbol. Dieses Symbol erscheint, wenn der Computer in der RUN-, PRO- und RSV-Betriebsart bereit ist, eine Eingabe aufzunehmen. Beim Tippen verschwindet das Bereitschaftssymbol und wird durch den Cursor ersetzt.
- Der Cursor. Dieses Symbol markiert die Stelle des nächsten einzugebenden Zeichens. Wenn man mit dem Tippen beginnt, ersetzt der Cursor das Bereitschaftssymbol. Als Markierungssymbol wird der Cursor auch im Zusammenhang mit den INSERT- und DELETE-Funktionen benutzt. Der viereckige Block-Cursor verwandelt sich, wenn er nicht über einem existierenden Zeichen in Position gebracht wird, wieder in einen Strich.
- BUSY Dieses Wort erscheint auf dem Display, wenn der PC-1280 ein Programm oder ein Kommando ausführt. Während dieser Zeit ist die Eingabe mit anderen Tasten nicht möglich.
- DEF Dieses Wort bezeichnet den Anweisungs-Modus. Dieses Symbol erscheint (ON) beim Druck auf die **DEF**-Taste und erlischt (OFF) mit dem folgenden Tastenkommando.

SHIFT	Die Anzeige geht an (ON), wenn die SHIFT -Taste gedrückt wurde und aus (OFF), mit dem folgenden Tastenkommando. Denken sie daran, daß die SHIFT -Taste vor dem Druck auf eine andere Taste freigegeben werden muß, wenn die zweite Funktion dieser Taste benutzt werden soll.
SML	Geht an (ON), wenn die SML -Taste betätigt wird. Buchstaben werden daraufhin als Kleinbuchstaben eingegeben. Zum Wiedereinstellen von Großbuchstaben wird SML erneut gedrückt.
DEG	Zeigt bei eingestellter Winkelfunktion den Grad-Modus an.
RAD	Zeigt bei eingestellter Winkelfunktion den Kreis-Modus an.
GRAD	Zeigt bei eingestellter Winkelfunktion den Neigungs-Modus an.
CAL	Hierdurch werden Sie darauf hingewiesen, daß sich der PC-1280 im Rechenbetrieb (CAL) befindet. Beachten Sie, daß in dieser Einstellung nur gewisse Tasten verwendet werden können.
RUN	Dieses Symbol zeigt den RUN-Modus für den PC-1280 an.
PRO	Dieses Symbol zeigt den PROgrammierungs-Modus für den PC-1280 an.
RSV	Dieses Symbol zeigt an, daß sich der PC-1280 im ReSeRve-Modus befindet.
PRINT	Dieses Symbol zeigt an, daß der PC-1280 bereit ist, im RUN- oder PRO-Modus Daten zum Drucker zu senden. Drücken Sie SHIFT und ENTER zum Ein- und Ausschalten (ON und OFF). (Nur möglich mit dem angeschlossenen Zusatzdrucker CE-126P.)
DBL	Dieses Symbol zeigt im BASIC-Modus an, daß Kalkulation mit doppelter Genauigkeit bestimmt wurde und das bis zu 20 Ziffern benutzt werden können.
M	Hierdurch wird angezeigt, daß im CAL-Modus ein anderer Wert als Null (0) gespeichert wurde.

E Hiermit wird angezeigt, daß Ihnen ein Fehler (ERROR) unterlaufen ist.

BATT Dieses Symbol weist Sie darauf hin, daß die Batterien schwach sind und so bald wie möglich ausgewechselt werden müssen.

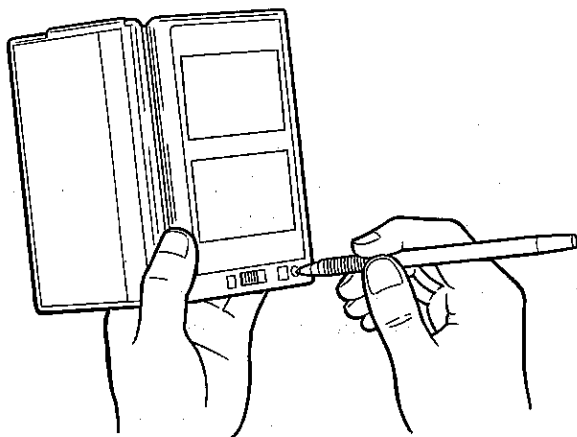
Zum Ausweiten der 2 Zeilen des Displays

Das Display des PC-1280 besteht aus 2 Zeilen (24 Zeichen pro Zeile). Tastatureingaben bzw. Rechenergebnisse werden von der ersten Zeile auf dem Display angezeigt. Werden die 2 Zeilen durch die eingegebenen Zeilen überschritten, so werden die angezeigten Inhalte 1 Zeile nach oben bewegt (die erste angezeigte Zeile verschwindet hinter der Oberkante des Bildschirms).

Rücksetzen

Bitte lesen Sie den gesamten Abschnitt durch, **BEVOR** Sie versuchen den PC-1280 rückzusetzen. Beachten Sie die folgenden Hinweise genau, so können Sie wertvolle Programme und Speicher schützen.

Es kommt vor, daß notwendig wird, den Computerspeicher zu löschen, seine Vor- und Grundeinstellungen in Ausgangsposition zu bringen und ihn erneut betriebsbereit zu machen. Zur Rücksetzung des PC-1280 stehen zwei Methoden zur Verfügung: alleiniger Druck auf den ALL RESET-Schalter, oder Druck auf den ALL RESET-Schalter bei gleichzeitigem Heruntergedrückthalten einer beliebigen Taste der Tastatur.



Drücken des ALL RESET-Schalters

Zum Drücken des ALL RESET-Schalters benutzen Sie einen spitzen Gegenstand, wie z.B. einen Kugelschreiber. Vermeiden Sie Gegenstände, deren Spitzen abbrechen können, wie Bleistifte oder Nadeln. Drücken Sie den Schalter und halten Sie ihn mindestens 3 Sekunden lang heruntergedrückt.

1. Schalten Sie den PC-1280 ein (ON), drücken Sie eine beliebige Taste auf der Tastatur (mit Ausnahme von der **Y**-Taste), halten Sie heruntergedrückt und drücken dann den ALL RESET-Schalter. Probieren Sie diese Methode, die alle Einstellungen und Programme im Speicher schützt, zuerst aus.
2. Drücken Sie den ALL RESET-Schalter bei eingeschaltetem Computer (ON) zum Löschen des gesamten Speichers, einschließlich der Programme, Variablen und Reservespeicher. Diese Rücksetzung des Computers ist vollständig und sollte nur dann angewendet werden, wenn Methode 1 den Computer nicht in seine Ausgangsstellung zur vollen Betriebsbereitschaft zurückbringt.

Auf dem Bildschirm erscheinen folgende Speicher-Löschinstruktionen:

MEM\$ = "Z"
MEMORY ALL CLEAR O.K. ?

Betätigen Sie die **Y**-Taste zum Löschen aller Speicherinhalte.

Drücken Sie die **N**-Taste zum Schutz der Speicherinhalte, wie in Schritt 1.

Diese Methode wird auch angewendet, den Computer nach Einsetzen oder Erneuern der Batteriezellen zu initialisieren.

Nach Betätigung des ALL RESET-Schalters kann das Display verlöschen oder es ist möglich, daß ein "verstümmeltes" Display erscheint. Drücken Sie den ALL RESET-Schalter erneut.

Falls nach Methode 1 das Gerät nicht normal arbeitet, entfernen Sie die Lithium-Batterien, reinigen die Kontakte und legen Sie die Batterien wieder ein. Drücken Sie erneut den ALL RESET-Schalter.

Einsetzen der Batterien

Der PC-1280 hat zwei Batteriesets: Betriebszellen, die das Display usw. mit Strom versorgen und eine Speicher-Sicherungszelle.

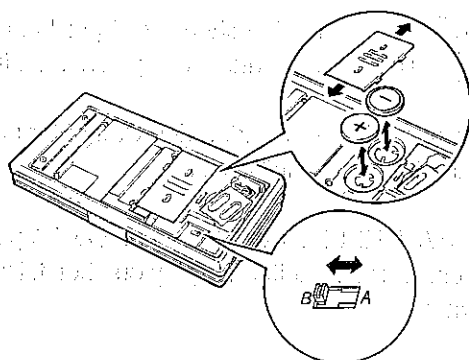
Die Betriebsbatteriezellen müssen so bald wie möglich erneuert werden, wenn das Display **BATT** anzeigt. Lassen Sie nie verbrauchte Zellen im Computer, da diese auslaufen und Beschädigungen am Computer verursachen können.

VORSICHT:

Bewahren Sie Batterien außerhalb der Reichweite von Kindern auf!

Erneuern der Betriebszellen

1. Schalten Sie das Gerät aus (OFF).
2. Sperren Sie den Computer und drehen Sie ihn so, daß die Unterseite nach oben weist. Zum Abnehmen der Abdeckung folgen Sie dem Diagramm auf dem Abdeckungsschlitz.
3. Entfernen Sie eine eventuell einliegende RAM-Karte.
4. Schalten Sie den Speicherschutz-Schalter auf B.
DAS IST WICHTIG und muß vor dem Herausnehmen der Batteriezellen geschehen, da sonst alle Daten und Programme im Speicher verloren gehen.



Einlegen der Betriebszellen

5. Drücken Sie den Tabulator und schieben den Deckel des Batteriegehäuses auf. Entfernen Sie die beiden Lithium-Batterien.
6. Legen Sie zwei neue Batteriezellen ein und achten Sie dabei auf die richtige, auf dem Diagramm angegebene, Polarität, wie im Diagramm gezeigt.

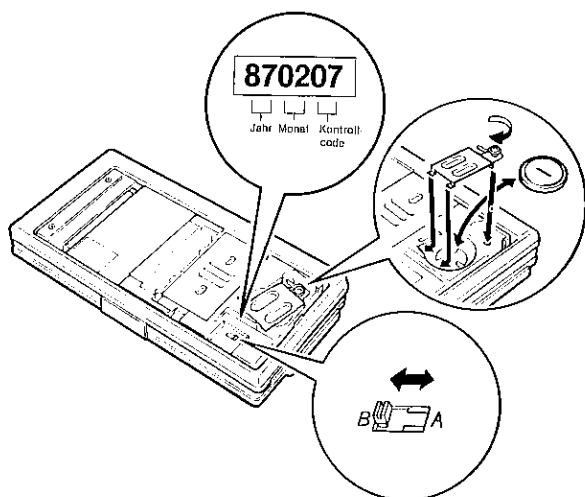
7. Schließen Sie das Batteriefach mit dem Deckel ④.
8. Betätigen Sie den ALL RESET-Schalter.
DIES IST WICHTIG zum Schützen der Programme und Daten.
9. Den Deckel schließen und den Schalter auf LOCK stellen.
10. Den Computer öffnen und einschalten und das Display beachten. Zeigt es das folgende mit einem *, die ALL RESET-Schalter drücken.



11. Schalten Sie den Computer aus, nehmen den Schlitzschutz ab und stellen den Speicherschutzschalter auf Position A.
12. Bei Verwendung, die RAM-Karten einführen.
13. Den Schlitzschutz aufsetzen und den Schalter auf LOCK stellen.

Erneuern einer Speicher-Sicherungszelle

Die Batterie zum Speichererhalt des PC-1280 hat unter normalen Betriebsbedingungen eine effektive Lebensdauer von ca. 5 Jahren. Dabei ist jedoch zu beachten, daß extreme Temperaturen diese Zeitspanne verkürzen und Speicherverlust verursachen. Beim werkseitigen Installieren der Batterie wurde am Speicherschutzschalter ein Etikett angebracht, daß Installationsdatum und Kontrollcode angibt.



Erneuern eine Speicher-Sicherungszelle

DIE SPEICHER-SICHERUNGSZELLE NICHT ENTNEHMEN, WENN DIESE SCHWACH ODER HERAUSGENOMMEN WERDEN, GEHEN ALLE SPEICHERINHALTE VERLOREN.

1. Schalten Sie den Computer ab (OFF). Sperren Sie den Computer und drehen Sie ihn so, daß die Unterseite nach oben weist.
2. Entfernen Sie die Schlitzabdeckung und die eventuell einliegenden RAM-Karte.
3. Kippen Sie den Speicherschutz-Schalter in die Position B.
4. Lösen Sie die Schraube, die den Deckel des Fachs für die Speicher-Sicherungszelle hält. Nehmen Sie den Deckel ab.
5. Legen Sie eine neue Zelle ein und achten dabei auf die richtige Polarität (Plus-pol weist nach unten).
6. Schalten Sie den Speicherschutz-Schalter wieder auf A.
7. Setzen Sie die Schlitzabdeckung wieder auf.

3. RAM-KARTEN

Der PC-1280 ist mit einem eingebauten 8K Bytes Memory zum Speichern von Programmen und Daten ausgestattet. Dies ist für die meisten Programme und Berechnungen ausreichend, kann aber durch Hinzufügen von einer zusätzlichen RAM-Karte in den Schlitz auf der Unterseite des Computers ausgeweitet werden.

RAM-Karten stehen in einer Kapazität von 2K Bytes bis 32K Bytes zur Verfügung. Alle enthalten eine Zelle, die ihre Inhalte sichert, auch wenn sie aus dem PC-1280 herausgenommen werden und auf diese Weise als handliche Alternative zu einer Kassette benutzt werden können. Jede Karte kann ausgerüstet werden, Daten oder Programme oder beides zu speichern und kann bei Bedarf in und aus dem PC-1280 ein- bzw. ausgewechselt werden (siehe dazu SET MEM, MEM und MEM\$ für Konfigurationen und INIT für Initialisierung der Karte).

Folgende Tabelle gibt die Kapazität jeder der zusätzlichen RAM-Karten und die ungefähre Lebensdauer ihrer Zelle an:

RAM-Karte	Programm & Datenbereich			RAM-Karte	
	MEM\$="1"	MEM\$="2"	MEM\$="B" MEM\$="D"	Kapazität	Lebensdauer
eingebaut	—	6558 Byt.	—	—	—
CE-210M	414 Byt.	6558 Byt.	8606 Byt.	2K Byt.	40 Monate
CE-211M	2462 Byt.	6558 Byt.	10654 Byt.	4K Byt.	24 Monate
CE-212M	6558 Byt.	6558 Byt.	14750 Byt.	8K Byt.	15 Monate
CE-2H16M	14750 Byt.	6558 Byt.	22942 Byt.	16K Byt.	14 Monate
CE-2H32M	31134 Byt.	6558 Byt.	39326 Byt.	32K Byt.	14 Monate

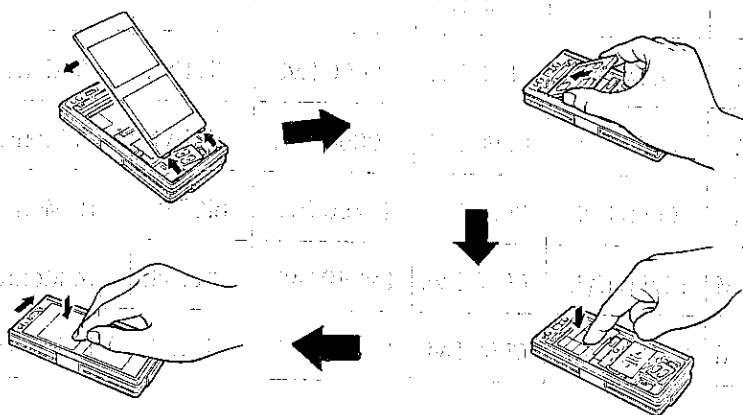
Anzahl der Dateien

Jede RAM-Karte (mit Ausnahme der CE-210M) kann entweder zum Einspeichern eines Programms oder einer Datei oder zu beidem benutzt werden. Die Anzahl der Dateien, die auf jeder einzelnen Karte gespeichert werden kann, ist durch die Größe der Dateien und durch den automatisch auf jeder Karte zugeordneten Platz für ihre charakteristischen Daten beschränkt. Die folgende Liste gibt die maximale Anzahl der Dateien (Programm und Daten) an, die auf jeder RAM-Karte gespeichert werden können.

CE-211M	4 Dateien
CE-212M	16 Dateien
CE-2H16M	32 Dateien
CE-2H32M	64 Dateien

Einlegen einer RAM-Karte

1. Überprüfen Sie, daß die RAM-Karte die einliegende Batterie enthält.
2. Schalten Sie den PC-1280 aus (OFF).
3. Schieben Sie den Verschuß auf der Unterseite des PC-1280 auf die Seite und entfernen Sie die rückseitige Abdeckung. Achten Sie darauf, daß Sie nicht einen der Steckeranschlüsse der RAM-Karte berühren.



Einlegen einer RAM-Karte

4. Legen Sie das Anschlußende der RAM-Karte dem angegebenen Pfeil entsprechend ein. Achten Sie darauf, die Karte nicht verkehrt herum einzulegen.

5. Drücken Sie die RAM-Karte leicht herunter, bis sie in ihren Platz einschnappt.
6. Schließen Sie mit der Abdeckung wieder den Schlitz. Beachten Sie, daß der PC-1280 nicht arbeitet, wenn die Abdeckung nicht korrekt verschlossen wurde.

Herausnahme einer RAM-Karte

1. Schalten Sie den PC-1280 aus (OFF).
2. Entfernen Sie, wie in der Einlegeanleitung beschrieben, die Schlitzabdeckung des PC-1280. Ziehen Sie am Hebel zum Einlegen bzw. zum Entfernen der Karte, wie in der Figur gezeigt, in Pfeilrichtung. Hierdurch wird die Karte vom Lötanschluß freigegeben, so daß sie herausgenommen werden kann. Nehmen Sie eine andere RAM-Karte aus ihrer Hülle heraus und legen die herausgenommene RAM-Karte in die RAM-Kartenhülle.
3. Setzen Sie die Schlitzabdeckung wieder auf den PC-1280 und schieben Sie den Deckel auf die "LOCK"-Position.

Anwendung der RAM-Karte

Auf die RAM-Karte geschriebene Programme bleiben erhalten (durch den Strom aus der Batterie in der RAM-Karte), auch wenn die Karte aus dem PC-1280 herausgenommen wird. Das gleiche Programm kann deshalb wieder verwendet werden, wenn die RAM-Karte erneut in den PC-1280 eingelegt wird.

RAM-Karten können auf verschiedene Weise benutzt werden. Mit dem SET MEM-Kommando im RUN- oder PRO-Modus wird der Gebrauch der RAM-Karte bezeichnet.

Vorsichtsmaßnahmen bei der Verwendung der RAM-Karte

Alle gespeicherten Programme und Daten werden gelöscht, wenn die Batterie in der RAM-Karte erneuert (entfernt) wird. Wir empfehlen Ihnen deshalb die wertvollen Programme auf der RAM-Karte zuerst auf Kassette oder auf eine andere RAM-Karte zu kopieren.

Die auf der RAM-Karte befindlichen Programme und Daten werden gelöscht, wenn der ALL RESET-Schalter betätigt wird. Im Abschnitt 2 finden Sie Hinweise zum Zurückstellen des PC-1280, während die Inhalte der RAM-Karten gesichert werden.

Kopieren von RAM-Karten

RAM-Karten, die zum Schutz von Programmen kein Codewort enthalten, können kopiert werden.

- Bei heruntergedrückt gehaltener **C**-Taste, schalten Sie den PC-1280 ein (ON).
- Auf die folgende Anzeige drücken Sie die **Y**-Taste:

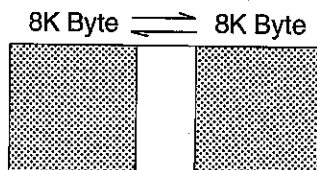
RAM CARD COPY ?

(Zum Abbrechen des RAM-Kopiervorgangs schalten Sie den PC-1280 aus.)
Das Display ändert sich auf:

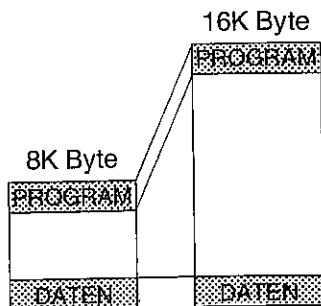
A: "1" — — — > "2"
B: "2" — — — > "1"

- Die **A**-Taste wird zum Kopieren von der RAM-Karte in den eingebauten Speicher gedrückt.
- Die **B**-Taste wird zum Kopieren von dem eingebauten Speicher auf die RAM-Karte gedrückt.

1. Alle RAM-Inhalte werden kopiert, wenn die RAM-Karte die gleiche Kapazität wie der eingebaute Speicher (8K Bytes) hat.



2. Wird eine RAM-Karte mit einer anderen Kapazität als 8K Bytes benutzt, so werden die Inhalte vom Beginn von RAM bis zum Ende des Textbereiches und vom Beginn der Datenzone bis zum Ende der RAM kopiert.



Hinweise:

In 2 werden die Inhalte nicht kopiert, wenn die Kapazität der DATENQUELLE größer ist als die des EMPFÄNGERS.

In 2 werden die Inhalte nicht kopiert, wenn die QUELLE nicht in der Stellung MEM\$="2" ist.

In sowohl 1 und 2 werden die Inhalte nicht kopiert, wenn die QUELLE von einem Code geschützt wurde.

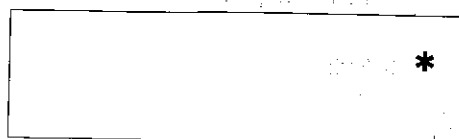
In sowohl 1 als auch 2 werden die Inhalte nicht kopiert, wenn der EMPFÄNGER geschützt geschrieben wurde.

Der Hinweis:

COPY COMPLETED.

erscheint auf dem Display, wenn der Kopiervorgang zufriedenstellend abgeschlossen ist.

Ist das Kopieren nicht befriedigend verlaufen, erscheint in der oberen rechten Ecke des Displays ein Stern:



Schalten Sie den PC-1280 aus (OFF) und versuchen Sie es erneut nach Überprüfung der relevanten Einstellungen.

Verwendung von RAM-Karten aus anderen Computern

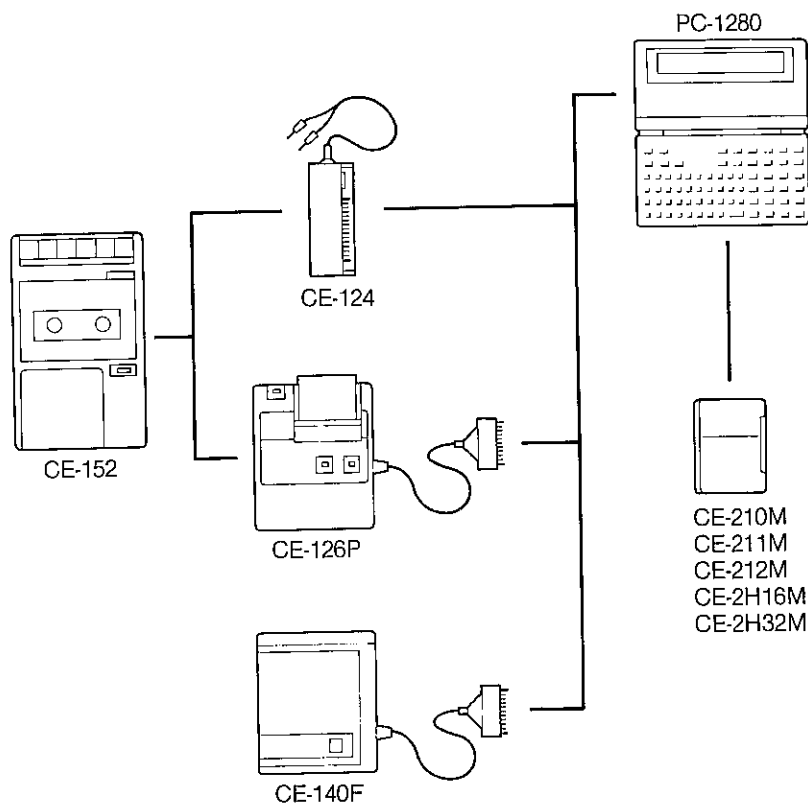
Die RAM-Karten des PC-1280 können mit anderen Computern der SHARP PC-Serie (PC-1450, PC-1460, PC-1425 und PC-1360) verwendet werden, und die RAM-Karten von diesen Computern können im PC-1280 benutzt werden.

Über die Benutzung von Programmen, die von anderen Computern der SHARP-Klasse importiert sind, finden Sie Details im Anhang H und im BASIC COMMAND-LEXIKON unter SET MEM-Kommando.

4. PERIPHERE GERÄTE

Der PC-1280 arbeitet zusammen mit den folgenden SHARP Zusatzgeräten:

MODELL	BESCHREIBUNG	ANSCHLUSS
CE-140F	Taschen-Disc-Einheit	11-Stift Stecker
CE-126P	Drucker/Kassette Interface	11-Stift Stecker
CE-124	Kassette Interface	11-Stift Stecker
CE-152	Kassettenrekorder	via CE-126P oder CE-124



Übersicht der Anlagenausrüstung

Eine kurze Beschreibung eines jeden Geräts folgt weiter unten. Zur genaueren Information lesen Sie bitte die entsprechenden Bedienungsanleitungen. Hinweise, wie RAM-Karten mit dem PC-1280 benutzt werden können, finden Sie im Abschnitt über RAM-Karten.

Diskettenlaufwerk in Taschenformat CE-140F

Das Diskettenlaufwerk im Taschenformat CE-140F (für 2,5-Zoll Taschen-Diskette), die direkt an den 11-Stift Stecker auf der linken Seite des PC-1280 angeschlossen werden kann, ähnelt die Taschen-Disketten, die normalerweise für Personalcomputer zur Verfügung stehen. Verglichen mit Kassettenrekordern erlauben sie Ihnen, ein Programm oder Daten schneller und leichter zu schreiben oder zu lesen. Dazu ermöglicht die Einheit Ihnen einen schnelleren Zugang zu einer Datenspeicherstelle und die Verwendung der COPY-Anweisung zur einfacheren Sicherung der Daten. Da der CE-140F batteriegetrieben ist, kann er leicht transportiert werden.

Lesen Sie die Bedienungsanleitung des CE-140F, die Hinweise zum Anschluß an den PC-1280 enthält.

CE-126P Thermodrucker/Kassetten Interface

Das zusätzliche CE-126P Drucker/Kassetten Interface erlaubt Ihnen, an Ihren SHARP PC-1280 einen Drucker und einen Kassettenrekorder anzuschließen.

Die Ausstattung des CE-126P:

- * 24-Zeichen Thermodrucker.
- * Bequemer Papiervorschub und Abreißschneide.
- * Auf Wunsch gleichzeitiger Ausdruck von Berechnungen.
- * Leichte Kontrolle des Displays oder des in BASIC Ausgedruckten.
- * Eingebautes Kassetten Interface Fernbedienungsfunktion.
- * Manuelle und Programmkontrolle des Rekorders zur Eingabe von Programmen und Daten.
- * Trockenbatteriebetrieb zum leichten Transportieren.

Zum Anschließen des PC-1280 an den CE-126P lesen Sie die Bedienungsanleitung, die dem CE-126P beiliegt.

CE-124 Kassetten Interface

Dieses Gerät ist erforderlich, wenn der Computer an einen Kassettenrekorder angeschlossen werden soll, um auf der Kassette Programme zu sichern oder zu laden (es sei denn, daß das eingebaute Interface des CE-126P benutzt wird). Mit dem 11-Stift Stecker wird es an den PC-1280 angeschlossen und an die Mikro-phon (MIC) und Kopfhörer (EAR) Eingangsbuchsen des Rekorders. Benutzen Sie entweder das SHARP CE-152 Modell, oder andere Kassettenrekorder mit den folgenden Spezifikationen:

Einheit	Anforderungen
1. Rekordertyp	Beliebiger Standardkassetten- oder Mikro-Kassettenrekorder
2. Eingangsbuchse	"MIC" Mini-Eingangsbuchse (nie die "AUX"-Buchse)
3. Eingangsimpedanz	Niedrige Eingangsbuchsenimpedanz (200—1000 Ohm)
4. Minimaler Eingangspegel	Unter 3 mV oder —50 dB
5. Ausgangsbuchse	EXTErn, MONITOR oder EARphone Mini-Ausgangsbuchse oder Entsprechendes
6. Ausgangsimpedanz	Unter 10 Ohm
7. Ausgangspegel	Über 1V (tatsächlicher Maximalausgang über 100 mW)
8. Verzerrung	Innerhalb 15% (Bereich: 2 kHz bis 4 kHz)
9. Gleichlaufschwankungen	0,3% maximal (WRMS)
10. Ferner	Rekordermotor mit stabiler Geschwindigkeit.

Spezifikationen des Kassettenrekorders

Benutzung des CE-126P Drucker/Kassetten Interface

Benutzung des Druckers

Benutzen Sie den PC-1280 zur direkten Rechnungseingabe (im RUN-Modus), so können Sie den CE-126P zum gleichzeitigen Ausdrucken Ihrer Berechnungen verwenden. Drücken Sie **SHIFT** und dann die **ENTER**-Taste (P↔NP) während des RUN-Modus.

Am Ende einer Rechnung drücken Sie die **ENTER**-Taste. Hierdurch werden die Inhalte einer Zeile des Displays ausgedruckt und die Ergebnisse auf der nächsten Zeile. Zum Beispiel:

Eingabe

Papier

300/50 **ENTER**

300/50

6.

Mit der LPRINT-Anweisung kann der Drucker aus BASIC-Programmen heraus ausdrucken (zur weiteren Information lesen Sie das BASIC KOMMANDO LEXIKON). Benutzen Sie LPRINT in der gleichen Weise wie die PRINT-Anweisung.

Mit dem LLIST-Kommando (zur weiteren Information lesen Sie das BASIC KOMMANDO LEXIKON) können Sie Ihre Programme mit dem Drucker auch auflisten. Ohne Zeilennummern werden durch den LLIST-Befehl alle im Speicher befindlichen Programmzeilen in Reihenfolge Zeile für Zeile aufgelistet. Es kann mit dem LLIST-Kommando auch ein Zeilenbereich angegeben werden, um die ausdruckenden Zeilen zu beschränken. Wenn Programmzeilen mehr als 24 Zeichen enthalten, sollten zwei oder mehr Zeilen zum Ausdruck der Programmzeile benutzt werden. Die zweite und die folgenden Zeilen werden vier oder sechs Anschläge eingerückt, so daß die Zeilennummer jede separate Programmzeile klar identifiziert (Zeilennummern von 1 bis 999: vier; über 999: sechs).

ACHTUNG:

- * Wird auf Grund von falsch eingezogenem Papier ein Fehler (ERROR 8) angezeigt, so reißen Sie das Papier ab und ziehen das verbleibende Papier vollständig aus dem Drucker. Anschließend drücken Sie die **C·CE**-Taste, um die Fehleranzeige zu löschen.

- * Ist der Drucker von Außen starkem elektrischen Lärm ausgesetzt, so kann es vorkommen, daß er Zufallsziffern ausdruckt. Wenn dies passiert, drücken Sie zum Stoppen des Ausdrucks die **BREAK**-Taste. Schalten Sie den CE-126P aus (OFF) und wieder ein (ON) und drücken dann die **C-CE**-Taste. Der Druck auf diese Taste bringt den Drucker auf Normalbetrieb zurück.
- * Wird der CE-126P nicht angewendet, so schalten Sie ihn aus (OFF), um die Lebensdauer der Batteriezellen nicht zu beeinträchtigen.

Anwendung des Kassetten Interface

Aufnahme (Sichern) auf Magnetband

Lesen Sie die Kassetten-Hinweise.

1. Stellen Sie den REMOTE-Schalter des CE-126P auf OFF.
2. Geben Sie ein Programm oder Daten in den Computer ein.
3. Legen Sie eine Kassette in den Rekorder. Fahren Sie das Band auf die Position, wo das Programm aufgenommen werden soll und achten Sie darauf das Bandführungsstück (nicht magnetisches Mylar-Material) und alle vorher aufgenommenen Programme zu meiden.)
4. Schließen Sie den roten Stecker des Interface an die MIC-Buchse des Kassettenrekorders an und den schwarzen Stecker an die REM-Buchse.
5. Stellen Sie den REMOTE-Schalter auf ON.
6. Drücken Sie gleichzeitig die RECORD- und PLAY-Tasten des Kassettenrekorders (um ihn in Aufnahmestellung zu bringen).
7. Geben Sie die Aufnahmebefehle ein (CSAVE-Kommando, PRINT#-Kommando) und drücken Sie zum Start der Ausführung die **ENTER**-Taste. Zuerst setzen Sie das Gerät in den RUN- oder PRO-Modus. Danach drücken Sie folgende Tasten:

C S A V E "Dateiname" **ENTER**.

Zum Schreiben der Inhalte des Datenspeichers gehen Sie folgendermaßen vor:

P R I N T # Variable **ENTER**.

Zum Beispiel: C S A V E "A A" **ENTER**.

Das Band beginnt sich nach dem Druck auf die **ENTER**-Taste zu drehen und läßt eine Leerstelle von ca 8 Sekunden (Ein Piepzeichen wird aufgenommen). Der Dateiname und der Inhalt der Dateien werden dann aufgenommen.

8. Ist die Aufnahme beendet, erscheint im Display das BEREITSCHAFTSYMBOL (>), und der Kassettenrekorder geht automatisch in Stoppstellung. Jetzt haben Sie Ihr Programm auf Kassette (es verbleibt auch im PC-1280).

Zum automatischen Aufnehmen von Daten aus einem Programm heraus (PRINT#-Befehl, nicht direkter Eingabebetrieb), führen Sie die Schritte 1 bis 6 durch, bevor Sie das Programm ausführen.

Zur Lokalisierung von Programmen auf Band benutzen Sie das Bandlaufwerk.

Überprüfen eines gesicherten Programms

Nach dem Laden oder Überführen eines Programms auf oder von einem Band, können Sie überprüfen, ob das Programm auf Band und das Programm im PC-1280 identisch ist (und so sichergehen, daß alles in Ordnung ist, bevor Sie mit dem Programmieren fortfahren oder Programme laufen lassen).

1. Stellen Sie den REMOTE-Schalter auf OFF.
2. Bringen Sie das Band in die Position unmittelbar vor der Datei, die überprüft werden soll.
3. Schließen Sie den grauen Stecker an die Buchse EAR an und den schwarzen Stecker an die Buchsenfassung REM.
4. Stellen Sie den REMOTE-Schalter auf ON.
5. Drücken Sie die Taste PLAY.
6. Geben Sie ein CLOAD?-Kommando ein und starten Sie mit der **ENTER**-Taste die Ausführung. Machen Sie dies folgendermaßen: Bringen Sie den PC-1280 in den RUN- oder PRO-Modus. Tippen Sie folgendes ein:

Vorher eingegebener Dateiname



CLOAD? "AA" **ENTER**

Der PC-1280 sucht jetzt automatisch nach dem bezeichneten Dateinamen und vergleicht die Inhalte auf Band mit den Inhalten im Speicher.

Ist der bezeichnete Dateiname auf Band gefunden, so wird an die auf dem Bildschirm getippte Zeile automatisch ein Stern (*) angehängt. Hierdurch wird angezeigt, daß der Überprüfungsvorgang begonnen hat.

Das BEREITSCHAFTSSYMBOL (>) erscheint auf dem Display, wenn die Programme identisch sind.

Unterscheiden sich die Programme, so wird die Ausführung unterbrochen und auf dem Display wird der Fehlercode ERROR 8 angezeigt. Kommt das vor, versuchen Sie es erneut.

Laden von einem Magnetband

Siehe Bandhinweise.

Zum Laden, Überführen oder Lesen von Programmen und Daten aus einem Magnetband gehen Sie folgendermaßen vor:

1. Stellen Sie den REMOTE-Schalter auf OFF.
2. Legen Sie das Band in das Tonbandgerät ein. Bringen Sie das Band unmittelbar vor die Stelle, die herausgelesen werden soll.
3. Schließen Sie den grauen Stecker an die EAR-Buchse und den schwarzen Stecker an die REM-Buchsenfassung an (hat das Gerät keine REM-Buchsenfassung, so kontrollieren Sie den Bandlauf manuell mit der PAUSE-Taste).
4. Stellen Sie den REMOTE-Schalter auf ON.
5. Drücken Sie die PLAY-Taste des Tonbandgeräts (Wiedergabestellung).
Stellen Sie die LAUTSTÄRKE auf Mitte oder Maximum ein.
Stellen Sie den TON auf höchste Höhen ein.

- Geben Sie die Überführungsanweisungen ein (CLOAD-, INPUT#-Kommandos) und drücken Sie die **ENTER**-Taste.
Bringen Sie den PC-1280 in den RUN- oder PRO-Betrieb. Dann drücken Sie die folgenden Tasten:

C L O A D "Dateiname" ENTER

(Zum Laden der Inhalte aus dem Datenspeicher tippen Sie folgendes:
I N P U T # Variable ENTER)

Zum Beispiel: **C L O A D "A/A" ENTER**

Der bestimmte Dateiname wird jetzt automatisch gesucht und seine Inhalte werden in den PC-1280 überführt.

Ist die bestimmte Datei auf dem Band gefunden, beginnt das Laden. Dies wird durch einen Stern (*) angezeigt, der automatisch an die auf dem Bildschirm getippte Zeile angehängt wird.

- Wurde das Programm überführt, stoppt der Computer automatisch das Band, und das BEREITSCHAFTSSYMBOL (>) erscheint im Display.

- Zur Übertragung von Daten (INPUT#-Befehl) in Verlauf eines Programms führen Sie die Schritte bis 5 durch, ehe Sie das Programm starten.

Erscheint ERROR 8, so versuchen Sie erneut die Ladung von Anfang an durchzuführen. Tritt wieder der Fehlercode 8 auf, wiederholen Sie die Prozedur, nachdem Sie die Lautstärke ein wenig höher oder niedriger eingestellt haben. Tritt kein Fehlercode auf, das Band aber nicht stöppt, da ist etwas falsch gelaufen. Drücken Sie die **BREAK**-Taste zum Stoppen des Bandes und versuchen Sie es erneut.

Bandhinweise

- Verwenden Sie zum Überprüfen und Laden immer das gleiche Bandgerät, mit dem Sie das Programm gesichert haben. Die Verwendung eines anderen Geräts kann Fehler verursachen.

2. Verwenden Sie nur hochqualitative Kassettenbänder. Normalbänder sollten nicht benutzt werden.
3. Halten Sie die Tonköpfe und den Bandmechanismus sauber. Falls Fehler beim Lesen, Überprüfen oder Laden des Programms auftreten, kann das Problem ein verschmutzter Tonkopf sein. Versuchen Sie den Tonkopf zu reinigen und zu entmagnetisieren und wiederholen Sie den Prozess.
4. Stellen Sie die Lautstärke ein zwischen Mitte und Maximum. Eine inkorrekte Einstellung kann auch manchmal Fehler verursachen. Experimentieren Sie mit der Einstellung der Lautstärke, bis keine Fehler mehr auftreten.
5. Halten Sie alle Leitungen und Anschlüsse sauber und korrekt eingesteckt.
6. Bei Verwendung eines Wechselstromadapters für den CE-126P Interface kann gelegentlich ein Brummen die Aufnahme begleiten. Kommt das vor, schalten Sie auf Batteriebetrieb um.
7. Der Ton sollte immer auf höchste Töne eingestellt werden.
8. Wird zur Aufnahme von Programmen ein altes Band wiederverwendet, dann löschen Sie vor der Aufnahme alle alten Programme auf dem Band.

DIREKT-EINGABE BETRIEB

Der PC-1280 erlaubt Ihnen im CAL- (Rechen) und RUN-Modus Direkt-Eingabe-Rechnungen (im Gegensatz zu Rechnungen als Teil eines Programmes im PRO-Modus). KAPITEL 2 dieses Handbuches beschreibt den Gebrauch dieser zwei Betriebseinstellungen.

Da zwischen den Betriebseinstellungen einige Funktionsüberschneidungen vorkommen, beginnt KAPITEL 2 mit einer Übersicht über die Ansteuerung von Bedienung und Modus des PC-1280. Die Betriebseinstellungen PRO und RSV werden detailliert dann im KAPITEL 3 behandelt.

5. BEDIENUNG: EINE FRAGE DER EINSTELLUNG

Bevor Sie Ihren neuen PC-1280 in Gang setzen, müssen Sie eine Betriebsart bestimmen. Der PC-1280 hat vier Betriebsarten. Eine Jede ermöglicht Ihnen Funktionen, die keine der anderen Betriebsarten hat:

CAL Rechenbetrieb. Diese Betriebsart verwandelt den PC-1280 in eine Art Taschenrechner, mit dem Sie wahrscheinlich schon gearbeitet haben. Berechnungen werden durchgeführt und die Ergebnisse werden unmittelbar auf dem Display angezeigt. Die Tasteneingabe ist beschränkt auf die Speichertasten, die Prozent-Taste (**%**), das Wurzel-Zeichen ($\sqrt{\quad}$), die Zahlen-Tasten (**0—9**), dem Punkt (**.**), die Tasten der 4 Rechenarten (**+**, **-**, **x**, **÷**), das Gleichheitszeichen (**=**) und die **C/CE**-Taste.

RUN Die RUN-Betriebsart hat zwei Hauptfunktionen. Die erste erlaubt direkten Eingabe der meisten BASIC-Kommandos mit unmittelbarer Anzeige der Ergebnisse auf dem Display. Zeilennummern sind weder notwendig, noch können sie im RUN-Modus angewendet werden. Der RUN-Modus findet auch zur Ausführung (RUN) von BASIC-Programmen, die im PROgramm-Modus geschrieben worden sind.

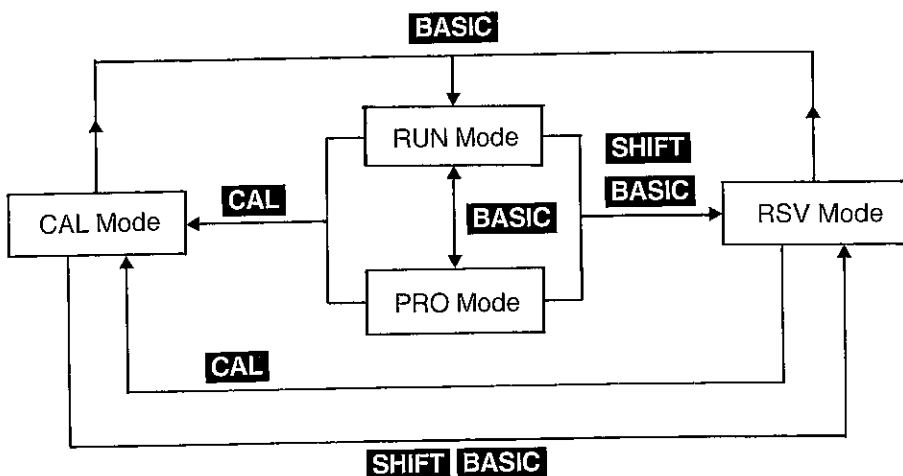
PRO In dieser Betriebsstellung werden BASIC Programme geschrieben, redigiert und gesichert oder gelöscht. Im PRO-Modus kann man keine Programme laufen lassen.

RSV Die ReSeRve-Betriebsstellung ermöglicht Ihnen die Bezeichnung und Eingabe vordefinierter Zeichenfolge- Variablen mit bestimmten Tasten.

Die Betriebsarten CAL, RUN und RSV werden in diesem Kapitel (KAPITEL 2) noch detailliert beschrieben. Der PRO-Modus wird in KAPITEL 3 erklärt.

Das Ansteuern von Betriebsarten

Die Betriebsarten werden angesteuert mit den Tasten **CAL**, **BASIC** und **SHIFT** + **BASIC**. Die Statuszeile auf dem Display zeigt die eingestellte Betriebsart an (CAL, RUN, PRO oder RSV). Die folgende Übersicht zeigt Ihnen, wie jede Betriebsart eingestellt wird.

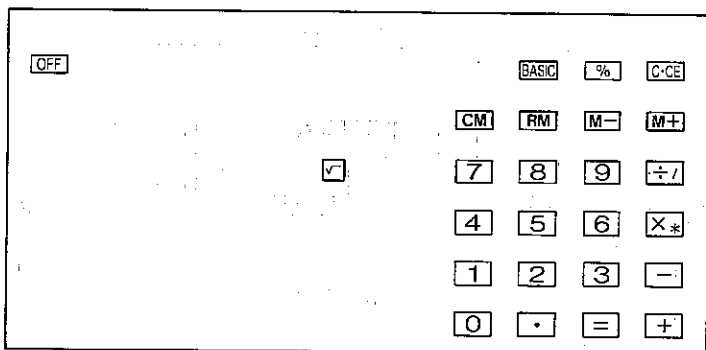


6. RECHENBETRIEB

Beginn des Rechenbetriebs

Mit der **CAL**-Taste stellt man den CAL-Modus (Rechenbetrieb) ein. Das CAL Symbol erscheint auf der Statuszeile im Display. Der PC-1280 verbleibt im Rechenbetrieb (CAL-Modus), egal wie oft Sie die **CAL**-Taste betätigen.

Die folgende Übersicht zeigt Ihnen, welche Tasten im CAL-Modus verwendet werden können.



Die Tastatur des Rechners

Im Rechenbetrieb (CAL-Modus) sind mit den Tasten der 4 Grundrechenarten, dem 4-Tasten-Memory, dem Quadratwurzelzeichen und der Prozent-Taste, 12-stellige Berechnungen möglich. Zur automatischen Punkte-Setzen in die eingegebenen Ziffern, können die Werte leichter abgelesen werden. (Zum Beispiel, im folgenden Beispiel, 1,234,567 multipliziert mit 8, werden die Kommata bei Druck auf die **X***-Taste automatisch angezeigt.

1 2 3 4 5 6 7

X*

8

1,234,567.

CAL

8.

CAL

Mit der **C·CE**-Taste wird die letzte Eingabe gelöscht (z.B. die nach dem Operator eingetippte Ziffer), oder ein Wert aus dem Speicher abgerufen.

1 2 3 + 4 5 6

456.
CAL

C·CE

123.
CAL

7 8 9 =

912.
CAL

ZWEIMALIGES Drücken der **C·CE**-Taste löscht alle vorausgegangenen Eingaben und Berechnungen. Die **C·CE**-Taste löscht ebenso den gesamten Speicher, falls, wie in obigem Beispiel, nicht nach einem Wert eingegeben.

6 × 2 +

12.
CAL

C·CE

0.
CAL

6 ÷ 2 +

3.
CAL

5 =

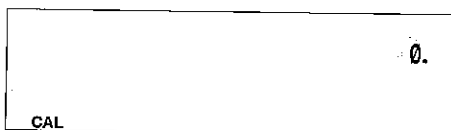
8.
CAL

Mit der **C·CE**-Taste löscht man auch einen Fehler (ERROR) (wenn das E-Zeichen auf der Statuszeile des Displays erscheint).

$$5 \div 0 =$$

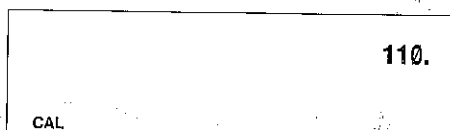


$$\text{C·CE}$$



Prozent-Rechnungen

$$100 + 10\%$$



$$100 - 10\%$$

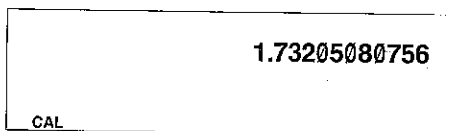


$$200 \div 10\%$$



Quadratwurzel-Rechnungen

$$3 \sqrt{}$$



Verwendung des Speichers für Berechnungen

Beispiel: $24 \times 13 =$

$+) 56 \div 7 =$

$-) 32 \times 4 =$

Summe ?

Eingabefolge:

C · CE CM

0.
CAL

 $2 \quad 4 \quad \times \quad 1 \quad 3 \quad M+$

312.

5 6 ÷ 7 M+

8.

3 2 X 4 M-

M
128.

RM

192.

7. RUN-BETRIEB

Jetzt, wo Sie sich mit der Anordnung und den Komponenten von SHARP's PC-1280 bekannt gemacht haben und im CAL-Modus einige einfache Berechnungen durchgeführt haben, kommen Sie dazu, einige aufregende Fähigkeiten Ihres neuen Computers zu entdecken.

Im RUN-Betrieb gibt Ihnen der PC-1280 die volle Breite von Rechenfunktionen PLUS die gesteigerte Leistung der BASIC Programmierfähigkeiten (nützlich für kompliziertere Berechnungen).

Einstellen des RUN-Betriebs

Drücken Sie die **ON**-Taste und überprüfen Sie die Statuszeile im Display des PC-1280. Wird CAL, PRO oder RSV angezeigt, so drücken Sie die **BASIC**-Taste, um in den RUN-Betrieb zu kommen. Überprüfen Sie, daß das Statusdisplay jetzt RUN anzeigt. Das Bereitschaftssymbol (>) zeigt an, daß der PC-1280 jetzt auf Ihre Eingabe wartet. Ihr Display sollte jetzt so aussehen:



Einige nützliche Hinweise

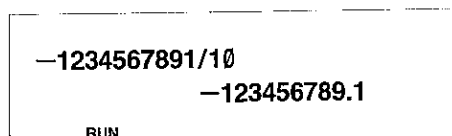
So lange Sie mit Ihrer neuen Maschine nicht völlige vertraut sind, werden ihnen bei der Dateneingabe sicher hier und da Fehler unterlaufen. Etwas später kommen wir dazu, Ihnen einige einfache Arten der Fehlerkorrektur zu erklären. Wenn Sie aber bereits jetzt die Fehlermeldung ERROR erhalten, so drücken Sie die **C·CE**-Taste und geben die Eingabe noch einmal ein. Wenn der Computer aber blockiert (er kann nicht auf alles antworten), betätigen Sie ALL RESET-Schalter (siehe dazu Abschnitt 2).

Das BEREITSCHAFTSSYMBOL (>) weist Sie darauf hin, daß der PC-1280 auf eine Eingabe wartet. Sobald Sie Daten eingeben, verschwindet das Bereitschaftssymbol und der CURSOR (___) geht an den rechten Rand des Displays und zeigt die nächste im Display zur Verfügung stehende Position an.

Der rechte  und der linke  Pfeil bewegen den Cursor.

Mit **ENTER** informieren Sie den PC-1280, daß Sie die Dateneingabe beendet haben, und daß der Computer die angegebenen Operationen durchführen kann. ENTER MUSS GEDRÜCKT WERDEN AM ENDE JEDER EINGABEZEILE ODER DER COMPUTER BEHANDELT IHRE RECHENEINGABEN NICHT.

Bei der Ausführung numerischer Berechnungen, erscheint die Eingabe links im Display und das Ergebnis rechts.



Wird die **SHIFT**-Taste in Verbindung mit einer anderen Taste benutzt (um z.B. in π zu kommen), drücken Sie **SHIFT**, geben **SHIFT** frei und drücken dann die andere Taste. **SHIFT** ist jedesmal nur für eine Taste aktiv.

Benutzen Sie bei der Eingabe von Berechnungen in den PC-1280 keine Dollarzeichen oder Kommata. Diese Zeichen haben in der BASIC Programmiersprache eine ganz besondere Bedeutung.

In diesem Handbuch benutzen wir zur Bezeichnung von Null das Zeichen $\emptyset$, so daß Sie die Nummer 0 und den Buchstaben O unterscheiden können.

Um Ihnen am Beginn zu einer korrekten Dateneingabe zu helfen, zeigen wir Ihnen in Rechnungsbeispielen jeden notwendigen Tastendruck. Wenn die **SHIFT**-Funktion angesprochen wird, werden wir das gewünschte Zeichen im folgenden Tastensymbol abbilden. Wenn z.B. Druck auf **SHIFT** und **%** das Fragezeichen (?) produziert, so schreiben wir **SHIFT %**.

Denken Sie daran, nach jeder Rechenoperation die **C·CE**-Taste zu betätigen (es sei denn, Sie möchten Kettenrechnungen durchführen). **C·CE** löscht die Anzeige und beseitigt eventuelle Blockade durch Fehler. Speicherinhalte werden nicht gelöscht.

Einfache Rechenoperationen

Der PC-1280 rechnet auf 12 Stellen genau. Schalten Sie ihren Computer ein und bringen Sie ihn in den RUN-Modus. Nun versuchen Sie, die folgenden einfachen Rechenbeispiele nachzuvollziehen.

5 0 + 5 0 ENTER

50 + 50

100.

RUN

1 0 0 - 5 0 ENTER

100 - 50

50.

RUN

6 0 * 1 0 ENTER

60 * 10

600.

RUN

Das Display des PC-1280 besteht aus zwei Zeilen (je 24 Zeichen pro Zeile). Alle Eingabe oder Rechenergebnisse werden jeweils von der obersten Zeile aus angegeben. Wenn die Daten die Kapazität von 2 Zeilen überschreiten, wird der Anzeigeeinhalt jeweils um eine Zeile nach oben gerollt (die erste angezeigte Zeile rollt nach oben und verschwindet dann aus dem Display).

3 0 0 / 5 ENTER

300 / 5

60.

RUN

1 0 SHIFT [^] 2 ENTER

10²

100.

RUN

2 * SHIFT $\sqrt{\pi}$ ENTER

2* π

6.283185307

RUN

$\sqrt{}$ 6 4 ENTER

$\sqrt{}$ 64

8.

RUN

4 E 3 ENTER

4E3

4000.

RUN

Wiederabruf von Eingaben

Auch nachdem der PC-1280 die Ergebnisse Ihrer Rechenoperation angezeigt hat, können Sie Ihre vorherige Eingabe aufbereiten lassen. Dazu benutzen Sie die Rechts-► und Links-◄ Pfeile.

Der Links-◄ Pfeil wird benutzt, um den Cursor an die Stelle hinter dem letzten Zeichen zu positionieren.

Der Rechts-► Pfeil wird benutzt, um den Cursor auf das erste Zeichen zu setzen.

Sie erinnern sich, daß die Rechts- und Links-Pfeile auch benutzt werden, um den Cursor innerhalb der Zeile zu bewegen. Die Pfeile sind sehr hilfreich beim Kontrollieren (oder Verändern) von Eingaben, ohne daß dabei der gesamte Ausdruck noch mal eingegeben werden muß.

Beim Durchlesen der folgenden Beispiele sollten Sie versuchen, diese gleichzeitig auf dem Computer auszuführen, um sich mit dem Gebrauch der Tasten vertraut zu machen.

Als Personalchef in einem großen Marketing-Unternehmen sind Sie für die Planung der jährlichen Verkaufstagung verantwortlich. Sie erwarten zu diesem dreitägigen Treffen 300 Personen. Für einen Teil der Zeit wird man sich in kleine Arbeitsgruppen aufteilen. Sie sind der Ansicht, daß Gruppen zu je 6 Personen eine gute Größe wären. Wieviele Gruppen müssen Sie einplanen?

300 / 6 ENTER

300/6
50.
RUN

Dann fällt Ihnen ein, daß Gruppen mit einer ungeraden Teilnehmerzahl möglicherweise effektiver arbeiten.

Rufen Sie Ihre letzte Eingabe mit Hilfe des  Pfeils wieder auf.



300/6
RUN

Um die neue Anzahl der Gruppen zu berechnen, müssen Sie 6 durch eine ungerade Zahl ersetzen. 5 Gruppenmitglieder scheinen Ihnen sinnvoller als 7.

Da Sie daran gedacht haben, den  Pfeil zu benutzen, steht der Cursor nun am Ende der Anzeige. Benutzen Sie noch einmal den  Pfeil, um den Cursor eine Stelle nach links zu bewegen.



300/6
RUN

Beachten Sie daß nach dieser Bewegung der Cursor sich in ein blinkendes Viereck verwandelt. Immer, wenn Sie den Cursor auf ein schon bestehendes Zeichen setzen, wird er in dieser Form dargestellt.

Geben Sie nun anstelle der 6 eine 5 ein. Achtung beim Einsetzen von Zeichen: Wenn Sie ein neues Zeichen über ein bereits bestehendes setzen, ist das ursprüngliche Zeichen endgültig gelöscht! Sie können keinen Ausdruck wieder abrufen, der überschrieben wurde.

5 ENTER

300/5	60.
RUN	

Sechzig scheint Ihnen eine vernünftige Gruppenanzahl zu sein. Sie beschließen deshalb, daß jede Kleingruppe aus 5 Teilnehmern bestehen wird.

Denken Sie daran, daß es sinnvoll ist, die letzte Eingabe zu kontrollieren, besonders dann, wenn Ihre Ergebnisse Ihnen zweifelhaft erscheinen. Nehmen Sie z.B. an, Sie hätten folgende Rechnung durchgeführt:

C CE 3 0 / 5 ENTER

30/5	6.
RUN	

Selbst ein müder, überarbeiteter Manager, wie Sie es sind, bemerkt, daß 6 kein glaubwürdiges Ergebnis ist, wenn man mit Hunderten von Personen operiert!

Rufen Sie Ihre Eingabe mit Hilfe des  Pfeils noch einmal ins Zeilendisplay.



30/5
RUN

Weil Sie den  Pfeil benutzt haben, steht der blinkende Cursor jetzt auf dem ersten Zeichen der Anzeige. Um die Eingabe zu korrigieren, möchten Sie jetzt eine zusätzliche Null einfügen. Mit Hilfe des  Pfeils bewegen Sie den Cursor, bis er über der Null steht. Wenn Sie nun mit der INS-Taste eine Einfügung machen wollen, setzen Sie den blinkenden Cursor auf denjenigen Buchstaben, vor dem das neue Zeichen eingefügt werden soll.



30/5

RUN

Mit der INS-Taste schaffen Sie Platz für das benötigte Zeichen.

INS

300/5

RUN

Das Betätigen von **INS** bewegt alle Zeichen um einen Schritt nach rechtss und fügt einen von Klammern umschlossenen Leerraum ein `[ ]`. Der blinkende Cursor steht nun auf diesem Leerraum und zeigt die Stelle der nächsten Eingabe an. Geben Sie die fehlende Null ein. Nachdem die Eingabe korrigiert ist, rufen Sie erneut Ihr Ergebnis ab.

0 ENTER

300/5

60.

RUN

Stellen Sie sich auf der anderen Seite vor, Sie hätten folgende Eingabe gemacht:

C CE 3 0 0 0 / 5 ENTER

3000/5

600.

RUN

Das Ergebnis scheint viel zu groß. Wenn Sie nur 300 Personen zu Ihrem Treffen erwarten, wie könnte es dann 600 Arbeitsgruppen geben?

Rufen Sie Ihre Eingabe mit Hilfe des  Pfeils ab.



3000/5

RUN

Der blinkende Cursor steht nun über dem ersten Zeichen der Anzeige. Um die Eingabe zu korrigieren, müssen Sie eine der Nullen löschen. Mit Hilfe des  Pfeils bewegen Sie den Cursor auf die erste (oder eine andere) Null. Wenn Sie ein Zeichen löschen wollen, setzen Sie den Cursor auf das zu löschende Zeichen.



3000/5
RUN

Um eine der Nullen zu löschen, betätigen Sie nun die DEL-Taste.

DEL

300/5
RUN

Durch Betätigen von **DEL** rutschen alle Zeichen um eine Stelle nach links. Das Zeichen, auf dem der Cursor steht, wird gelöscht, ebenso der Freiraum, den dieses Zeichen beansprucht. Der blinkende Cursor bleibt an der selben Stelle stehen und zeigt die nächste Stelle für eine Eingabe an. Da Sie keine weiteren Veränderungen vorzunehmen haben, beenden Sie die Rechnung.

ENTER

300/5	60.
RUN	

Achtung: Steht der Cursor auf einem Zeichen und Sie betätigen die SPC-Taste, wird das betreffende Zeichen gelöscht und durch einen Leerschritt ersetzt. Mit DEL löschen Sie das Zeichen und den von ihm eingenommenen Leerraum.

Fehler

Der Abruf der vorherigen Eingabe ist notwendig, wenn Sie eine ERROR-Meldung erhalten. Wir wollen uns vorstellen, daß Sie aus Versehen folgendes in den PC-1280 eingegeben haben:

C CE 3 0 0 / / 5 ENTER

300//5
ERROR 1

RUN

Natürlich sind Sie überrascht, wenn diese Meldung erscheint! ERROR 1 ist die einfache Art des Computers, Ihnen mitzuteilen: "Ich weiß nicht, was ich hier tun soll." Wenn Sie in diesem Fall den  Pfeil betätigen, erscheint der blinkende Cursor an der Stelle, wo der Fehler liegt.



300//5

RUN

Um diesen Fehler zu korrigieren, benutzen Sie die **DEL**-Taste.

DEL ENTER

300/5

60.

RUN

Zum Löschen von Fehlermeldungen kann die **BS**-Taste verwendet werden. Beachten Sie, daß **BS** den Cursor um eine Stelle zurückbewegt und das dortige Zeichen löscht. **DEL** dagegen löscht das Zeichen, auf dem der Cursor steht.

Wenn Sie beim Abruf Ihrer Eingabe aufgrund der Meldung ERROR 1 feststellen, daß Sie ein Zeichen vergessen haben, benutzen Sie die INS-Sequenz, um es zu korrigieren.

Wenn Sie den PC-1280 wie einen normalen Taschenrechner benutzen, werden Sie in der Hauptsache mit dem ERROR 1 (Syntaxfehler) konfrontiert werden. Eine komplette Aufstellung der Fehlermeldungen und ihrer Bedeutungen finden Sie im ANHANG A.

Verkettung von Rechenoperationen

Der PC-1280 erlaubt Ihnen, daß Sie das Ergebnis einer Rechnung in der folgenden Operation weiterverwenden.

Zurück zu unserem Beispiel: Ein Teil Ihrer Verantwortung bei der Planung dieser Konferenz besteht darin, einen detaillierten Budget-Vorschlag auszuarbeiten. Sie wissen, daß Ihnen für jeden Teilnehmer DM4000 zur Verfügung stehen. Berechnen Sie Ihr Gesamtbudget.

C CE 3 0 0 * 1 5 0
ENTER

300*150
45000.
RUN

Von dieser Summe beabsichtigen Sie, 15% für die Abschlußfeier aufzuwenden. Wie hoch ist Ihr Budget für die Feier?

Bei verketteten Rechenoperationen ist es nicht notwendig, Ihre vorherigen Ergebnisse nach einmal einzugeben, aber drücken Sie zwischen den Eingaben KEINESFALLS **C CE**.

*** . 1 5**

45000.*.15__
RUN

Beachten Sie, daß der Computer das Ergebnis Ihrer ersten Berechnung automatisch links im Display anzeigt und die neue Rechnung mit einbezieht, wenn Sie die Kommandos für die zweite Berechnung (***.15**) eingeben. In verketteten Rechenoperationen muß die Eingabe mit einem Operationssymbol beginnen. Beenden Sie Ihre Eingabe, wie immer, mit **ENTER**.

Achtung: Die **%**-Taste kann nicht in Rechenoperationen benutzt werden. Verwenden Sie die **%**-Taste ausschließlich als ein Zeichen.

Beispiel: **4 5 0 0 0 * 1 5 % ENTER** → ERROR 1

ENTER

45000.*.15

6750.

RUN

Fahren Sie fort, Ihr Budget zu berechnen. Das Hotel verlangt für die Verpflegung DM4000:

- 4 0 0 0 ENTER

6750.-4000

2750.

RUN

Die Dekoration wird DM1225 kosten:

- 1 2 2 5 ENTER

2750.-1225

1525.

RUN

Schließlich müssen Sie DM 2200 für den Sprecher und für das Rahmenprogramm einkalkulieren:

- 2 2 0 0 ENTER

1525.-2200

-675.

RUN

Offensichtlich werden sie entweder Ihre Pläne oder Ihre Kostenverteilung ändern müssen!

Negative Zahlen

Da die Abschlußfeier etwas ganz besonderes sein soll, beschließen Sie Ihre Pläne nicht zu ändern und zusätzliches Geld zu investieren. Trotzdem fragen Sie sich, wieviel Prozent des Gesamtbudgets wohl von diesem Posten in Anspruch genommen werden. Zuerst ändern Sie das Vorzeichen Ihres letzten Ergebnisses:

*** - 1
ENTER**

$-675. * -1$	675.
RUN	

Nun addieren Sie dieses Ergebnis zu Ihrem ursprünglichen Budget für die Feier hinzu:

+ 6 7 5 0 ENTER

$675. + 6750$	7425.
RUN	

Teilen Sie Ihr neues Ergebnis durch 45000 und Sie erfahren, wieviel Prozent des Gesamtbudgets diese neue Zahl ausmacht:

/ 4 5 0 0 0 ENTER

$7425. / 45000$	0.165
RUN	

Nun gut, Sie beschließen, 16,5% für die Abschlußfeier aufzuwenden.

Komplexere Rechenoperationen und Klammersetzung

Bei der Durchführung der obigen Rechenbeispiele hätten Sie einige dieser Operationen in einem Schritt zusammenfassen können. So hätten Sie etwa die beiden folgenden Berechnungen in eine Zeile schreiben können:

$$675 + 6750/45000$$

Solche komplexeren Rechnungen müssen jedoch sehr sorgfältig eingegeben werden:

675 + 6750/45000 könnte verstanden werden als

$$\frac{675 + 6750}{45000}$$

oder

$$675 + \frac{6750}{45000}$$

Bei der Durchführung komplexer mathematischer Operationen folgt der PC-1280 spezifischen Regeln der impliziten Klammerung und der Rangfolge von mathematischen Operationen (siehe ANHANG D). Um sicherzustellen, daß Ihre Eingaben in der von Ihnen gewünschten Reihenfolge bearbeitet werden, sollten Sie Klammern verwenden.

$$(675 + 6750)/45000 \quad \text{oder} \quad 675 + (6750/45000)$$

Wie unterschiedlich sich die Platzierung von Klammern auf das Ergebnis auswirkt, sehen Sie in den beiden folgenden Beispielen:

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

$$(675 + 6750)/45000$$

0.165

RUN

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

$$675 + (6750/45000)$$

675.15

RUN

Die Anwendung von Variablen in Rechenoperationen

Der PC-1280 kann bis zu 26 einstellige Variablen unter den alphabetischen Zeichen A — Z speichern. Wenn Ihnen der Begriff der Variablen nicht vertraut ist: Im Abschnitt 8 finden Sie ausführliche Erklärungen. Die Variablenzuweisung geschieht über einen Zuweisungsbefehl:

$$A = 5$$

$$B = -2$$

Sie können auch den Wert einer Variablen (rechts) einer anderen (links) zuweisen.

$$C = A + 3$$

$$D = E$$

Eine Variable kann in jeder Rechenoperation anstelle einer Zahl benutzt werden.

Nachdem Sie nun Ihre Abschlußfeier geplant haben, fehlen Ihnen noch die übrigen Berechnungen für Ihre Verkaufstagung. Sie möchten den Rest Ihres Budgets ebenfalls prozentual aufteilen. Zuerst müssen Sie herausfinden, wieviel Geld sie noch zur Verfügung haben. Weisen Sie eine Variable (R) dem verbleibenden Rest zu:

R = 4 5 0 0 0 - 7 4 2 5
ENTER

R = 45000-7425
37575.
 RUN

Wenn Sie nun **ENTER** drücken, führt der PC-1280 die Berechnung durch und zeigt den neuen Wert von R an. Sie können den Wert von jeder Variablen zur Anzeige bringen, indem Sie den Buchstaben eingeben, unter dem er gespeichert ist:

R ENTER

R
37575.
 RUN

Nun können Sie Rechenoperationen mit Ihren Variablen durchführen. Der Wert von R wird sich nicht verändern, bis Sie ihm einen neuen Wert zuweisen.

Sie beabsichtigen, 60% des verbleibenden Geldes auf die Unterbringung zu verwenden:

R * . 6 0 ENTER

R*.60

22545.

RUN

Weiterhin wollen Sie 25% des verbleibenden Budgets für die Durchführung von Management-Trainings-Seminaren ausgeben:

R * . 2 5 ENTER

R*.25

9393.75

RUN

Variablen behalten ihren zugewiesenen Wert auch wenn das Gerät ausgeschaltet (OFF) wird oder sich mit AUTO OFF automatisch ausschaltet. Variable gehen nur verloren, wenn:

- * Man ihnen einen neuen Wert zuweist.
- * Man INC CLEAR **ENTER** (nicht **C-CE**) eingibt.
- * Man das Gerät mit dem ALL RESET-Schalter löscht.
- * Die Batterien gewechselt werden (nur wenn die Speichersicherungs-Batterien gewechselt werden).

Es gibt gewisse Grenzen der Variablenzuweisung, sowie gewisse Programmabläufe, die Variablen verändern können. Im Kapitel 8 finden Sie detaillierte Informationen zur Variablenzuweisung, in Kapitel 9 wird der Gebrauch von Variablen bei der Programmierung erörtert.

Mehrere Rechenschritte

Mit dem PC-1280 ist es nicht nur möglich, verschiedene Rechenschritte zusammenzufassen, man kann auch einzelne Rechnungen nacheinander durchführen, ohne zwischendurch **ENTER** betätigen zu müssen. Die einzelnen Rechenoperationen werden durch Kommata getrennt. In der Anzeige erscheint nur das Ergebnis der letzten Rechenoperation. (Beachten Sie auch bitte, daß die maximale Zeilenlänge, die Ihr Computer annimmt, einschließlich **ENTER** 80 Zeichen beträgt.)

Sie fragen sich, wieviel Geld für die Unterbringung zur Verfügung stünde, wenn Sie, wie ursprünglich geplant, nur 15% für die Abschlußfeier aufwenden würden.

R = .85 * 45000, R
*** .60**

R = .85*45000,R*.60

RUN

Obwohl der Computer alle eingegebenen Rechenoperationen durchführt, zeigt er nur das letzte Ergebnis an:

ENTER

R = .85*45000,R*.60

22950.

RUN

Um herauszufinden, welchen Wert R in dieser Berechnung hat, geben Sie R ein:

R ENTER

R

38250.

RUN

Wissenschaftliche Notation

Wer mit sehr großen oder sehr kleinen Zahlenwerten zu arbeiten hat, benutzt oft ein spezielles Format, die sogenannte exponentielle oder wissenschaftliche Notation. In der wissenschaftlichen Notation wird eine Zahl in zwei Teile aufgespalten.

Der erste Teil besteht aus einer Dezimalzahl zwischen 1 und 10. Der zweite Teil gibt die Größe der Zahl in Zehnerpotenzen an. Wie Sie wissen, ist in einer gewöhnlichen Dezimalzahl die erste Stelle links von Komma die Stelle der Einer, an zweiter Stelle folgen die Zehner, an dritter die Hunderter, dann die Tausender usw. Dies sind einfach ansteigenden Potenzen von 10:

$$10^0 = 1, 10^1 = 10, 10^2 = 100, 10^3 = 1000, \text{ usw.}$$

Die wissenschaftliche Notation bricht also eine Dezimalzahl in zwei Teile auseinander. Ein Teil zeigt, um welche Ziffernfolge es sich handelt, der zweite, wieviele Stellen eine Zahl links oder rechts vom Komma aufweist. Zum Beispiel:

1234 wird 1,234 mal 10^3 (3 Stellen nach rechts)

654321 wird 6,54321 mal 10^5 (5 Stellen nach rechts)

.000125 wird 1,25 mal 10^{-4} (4 Stellen nach links)

Wissenschaftliche Notation ist sehr hilfreich, um Eingaben zu verkürzen. Um 1,0 mal 10^{87} einzugeben, müßte man eine 1 mit 87 Nullen schreiben! In wissenschaftlicher Notation aber würde diese Zahl folgendermaßen aussehen:

$$1.0 \times 10^{87} \text{ oder } 1.0\text{E}87$$

Der PC-1280 benutzt die wissenschaftliche Notation immer dann, wenn Zahlen in dezimaler Schreibweise die Anzeigekapazität übersteigen würden. Zu große Werte sind größer als 9999999999, zu kleine kleiner als -9999999999 im Modus der einfachen Genauigkeit, oder im Modus der doppelten Genauigkeit (DBL erscheint auf der Statuszeile im Display) größere Wert als 99999999999999999999 oder kleinere Werte als -99999999999999999999. Im Modus der einfachen Genauigkeit benutzt der PC-1280 den Großbuchstaben E, der "10 mal bis" bedeutet, und im Modus der doppelten Genauigkeit den Großbuchstaben D:

Betrachten Sie folgendes Beispiel in einfacher Genauigkeit:

```

-1234567891/10
-123456789.1
    
```

RUN

Die wissenschaftliche Notation wird hier nicht verwendet, weil die Werte die Kapazität nicht überschreiten.

Obwohl die Berechnung $0.1234567891 / 10000$ die Kapazität nicht überschreitet, und folgendermaßen im Display erscheint:

```

0.1234567891/10000
1.234567891E-05
    
```

RUN

wird angezeigt, daß das Ergebnis $1.234567891 \times 10^{-5}$ ist.

Im doppelten Genauigkeitsbetrieb (DBL erscheint auf der Statuszeile im Display) verwendet der PC-1280 den Großbuchstaben D, um anzuzeigen "10 mal bis":

```

DBL
-12345678912345678912/10
-1234567891234567891.2
    
```

RUN

Die wissenschaftliche Notation wird nicht benutzt, weil die Werte kleiner als die Grenzwerte sind.

Aber betrachten Sie das folgende Beispiel:

DBL
0.12345678912345/1000000
1.2345678912345D-07
RUN

Hier wird die Antwort $1.2345678912345 \times 10^{-7}$ durch ein D bezeichnet.

Wenn der Ausdruck mehr als 24 Anschläge hat, wird der Bildschirm bei der Eingabe von **ENTER** gelöscht und dann das Ergebnis angezeigt:

DBL
0.12345678912345678912/1
0000 _
RUN

Drücken Sie **ENTER**:

DBL
1.2345678912345678912
D-05
RUN

Sind Sie mit diesem Typ von Notation nicht vertraut, so nehmen Sie sich die Zeit einige sehr große und sehr kleine Werte einzugeben, um zu sehen, wie sie angezeigt werden.

Einfache Genauigkeit, Doppelte Genauigkeit

Die größte Zahl, mit der der PC-1280 operieren kann, besteht aus 10 Stellen plus 2 Stellen im Exponenten. Mit anderen Worten, die größte Zahl ist:

[illegible]

Die kleinste Zahl ist:

[illegible]

In doppelter Genauigkeit besteht die Zahl aus 20 Stellen, 9.9999999999999999 D 99 und 1. D—99 (das große E zeigt an, daß der Exponent im Modus der doppelten Genauigkeit durch ein großes D ersetzt wurde).

Einstellen des doppelten Genauigkeits-Modus

1. Gehen Sie in den RUN-Modus und drücken Sie zum Löschen des Displays **C CE**.
2. Tippen Sie DEFDBL und betätigen Sie **ENTER**.
DBL wird auf der Statuszeile im Display angezeigt, was bedeutet, daß sich der PC-1280 jetzt im Modus der doppelten Genauigkeit befindet.

Der Exponent wird in der doppelten Genauigkeit durch ein großes D angezeigt, wie im vorhergehenden Abschnitt beschrieben. Sie können jedoch bei der Eingabe eines Exponenten in einen Ausdruck entweder das konventionelle E oder das D verwenden.

Löschen des Modus der doppelten Genauigkeit

1. Zum löschen des Displays drücken Sie im RUN-Modus **C CE**.
2. Tippen Sie DEFSNG und drücken **ENTER**.
Das DBL auf der Statuszeile erlischt. Hierdurch wird angezeigt, daß sich der PC-1280 jetzt im Modus der einfachen Genauigkeit befindet.

Der doppelte Genauigkeitsmodus wird automatisch gelöscht, wenn:

1. Der PC-1280 ausgeschaltet wird (OFF) (oder durch die AUTO OFF-Funktion automatisch ausgeschaltet wird).
2. Das RUN-Kommandos ausgeführt wird.

Werte-Schablone

In einer einfachen Kettenrechnung kann das Ergebnis der vorherigen Berechnung lediglich als erste Zahl der gegenwärtigen Berechnung verwendet werden.

Zum Beispiel:

3 + 4 ENTER

3 + 4	7.
RUN	

*** 5 ENTER**

7.*5	35.
RUN	

Der PC-1280 hat aber eine Funktion, die Ihnen erlaubt, das Ergebnis der vorherigen Berechnung zurückzuholen und es irgendwo in der gegenwärtigen Berechnung zu benutzen. Dies nennt man eine Werte-Schablone und der vorige Wert kann beliebig oft zurückgerufen werden, indem Sie die **↑** - oder die **↓** -Taste drücken. Wenn Sie das letzte Beispiel eingegeben haben, so drücken Sie **C.CE** und dann entweder **↑** oder **↓** worauf die "35" angezeigt wird.

Schauen wir uns ein Beispiel an, wo ein vorheriges Ergebnis zweimal in der gegenwärtigen Berechnung eingesetzt wird.

Bitte beachten Sie, daß sich in diesem Beispiel die Werte-Schablone verändert und auf den neuesten Stand der gegenwärtigen Berechnung gebracht wird, jedesmal, wenn Sie **ENTER** drücken.

(Beispiel) Verwende das Ergebnis (6.25) der Operation 50 / 8, um folgendes zu berechnen: $12 \times 5 / 6.25 + 24 \times 3 / 6.25 =$

5 0 / 8 ENTER

50/8	6.25
RUN	

Letzte Antwort

1 2 * 5 / ↑ (oder ↓)

12*5/6.25__
RUN

Zuletzt abgerufene Antwort

+ 2 4 * 3 / ↑ (oder ↓)

12*5/6.25 + 24*3/6.25__
RUN

ENTER

12*5/6.25 + 24*3/6.25

21.12

RUN

C·CE ↑

21.12

RUN

Durch Druck auf **ENTER** wird die vorige "Letzte Antwort" durch das Ergebnis der vorangegangenen Rechnung ersetzt. Die letzte Antwort wird durch das Betätigen von **C·CE** oder einer anderen Taste jedoch nicht gelöscht.

Die letzte Antwort kann nicht angerufen werden, wenn sich der Computer im RUN-Modus befindet, die Programmausführung vorübergehend unterbrochen oder der Trace-Modus (TRON) gewählt ist.

Maximale Rechenlänge

Die Länge der Rechnung, die Sie eingeben können, ist auf 79 Tastenbedienungen begrenzt, bevor die **ENTER**-Taste gedrückt wird. Wenn Sie diese Grenze überschreiten, beginnt der Cursor zu blinken, um anzuzeigen, daß die folgende Eingabe ungültig ist. In diesem Fall muß die Rechnung in zwei oder mehrere Schritte aufgeteilt werden.

Wissenschaftliche Rechenoperationen

Der PC-1280 hat ein großes Ausmaß numerischer Funktionen für wissenschaftliche Rechenoperationen. KAPITEL 4 enthält eine alphabetische Auflistung dieser Funktionen (durch das Öffnen dieses Handbuchs an den gekennzeichneten Seiten schnell zu finden). Beachten Sie, daß die Notation der Funktionen in BASIC sich von den herkömmlichen mathematischen Notationen unterscheiden können.

Der PC-1280 ermöglicht auch die Bestimmung von Winkleinheiten, Bogenmaßen und Neugraden mit den Anweisungen DEGREE, RADIAN oder GRAD.

Winkeleinheit	Befehl	Erklärung
Alt-Grad	DEGREE	Rechter Winkel = $90[^\circ]$
Bogenmaß	RADIAN	Rechter Winkel = $\pi/2[\text{rad}]$.
Neugrad	GRAD	Rechter Winkel = $100[g]$.

Verwenden Sie zur Übung diese Hinweise, um in den folgenden Rechenbeispielen die Winkleinheiten zu bestimmen.

(Beispiel) $\sin 30 =$
 (Operation) DEGREE **ENTER** (spezifiziert "Alt-Grad" als Winkleinheit.)

SIN 30 ENTER

SIN 30
 0.5
 RUN

(Beispiel) $\tan \pi/4 =$
 (Operation) RADIAN **ENTER** (Spezifiziert "Bogenmaß" als Winkleinheit.)

TAN (PI / 4) ENTER

TAN (PI / 4)
 1.
 RUN

(Beispiel) $\cos^{-1}(-0.5) =$
 (Operation) DEGREE **ENTER** (Spezifiziert "Alt-Grad" als Winkleinheit.)

ACS - 0.5 ENTER

ACS -0.5
 120.
 RUN

(Beispiel) $\log 5 + \ln 5 =$

(Operation)

LOG 5 + LN 5
ENTER

LOG 5 + LN 5

2.308407917

RUN

(Beispiel) e^{2+3} (Operation)

EXP (2 + 3)
ENTER

EXP (2 + 3)

148.4131591

RUN

(Beispiel) $\sqrt{4^3 + 6^4}$

(Operation)

√ (4 SHIFT ^ 3 + 6
SHIFT ^ 4) ENTER

√ (4^3 + 6^4)

36.87817783

RUN

(Beispiel) Wandle 30 Grad 30 Minuten in DMS-Darstellung um in Dezimaldarstellung.

(Operation)

DEG 30 . 30 ENTER

DEG 30.30

30.5

RUN

(30.5 DEGREE)

(Beispiel) Wandle 30.755 Grad in Dezimaldarstellung um in DMS-Darstellung.

(Operation)

D M S 3 0 . 7 5 5
ENTER

DMS 30.755	30.4518
RUN	

(30 DEG. 45 MIN. 18 SEC.)

(Beispiel) Wandle CF8 in eine Zedimalzahl um.

(Operation)

SHIFT & C F 8 ENTER

&CF8	3320.
RUN	

Rangfolge bei manuellen Berechnungen

Sie können Ausdrücke in derselben Reihenfolge eingeben, in der sie geschrieben sind, einschließlich der Klammerung und der Funktionen. Die interne Reihenfolge bei der Berechnung und Behandlung von Zwischenergebnissen werden vom Computer selbst kontrolliert.

Die interne Rangfolge bei manuellen Berechnungen ist die folgende:

1. Abruf von Variablen oder PI
2. Funktionen (sin, cos usw.)
3. Exponentiation (^)
4. Vorzeichen (+, -)
5. Multiplikation oder Division (*, /)
6. Addition oder Subtraktion (+, -)
7. Größenvergleich (>, >=, <, <=, <>, =)
8. Logisches AND, OR, XOR

Anmerkungen:

- Werden in einem Ausdruck Klammern verwendet, hat die innerhalb der Klammer angegebene Operation höchste Priorität.

- Zusammengesetzte Funktionen werden von rechts nach links abgearbeitet ($\sin \cos^{-1} 0.6$).
- Mehrstufige Exponentiation (3^4^2 oder 3^4^2) wird von rechts nach links abgearbeitet.
- Von den obigen Punkten 3) und 4) hat die letzte Eingabe die höhere Priorität. (Z.B.) $-2^4 \rightarrow -(2^4)$
 $3^{-2} \rightarrow 3^{-2}$

Ausdrucken von Rechenergebnissen

Berechnungen und Ergebnisse können durch Betätigen von **SHIFT** **ENTER** ($P \leftrightarrow NP$) ausgedruckt werden, sofern der zusätzliche Drucker (CE-126P) angeschlossen und eingeschaltet ist. Beachten Sie, daß Berechnungen im CAL-Modus ausgeführt, nicht ausgedruckt werden können.

Wenn der Ausdruck nicht gewünscht wird, schalten Sie entweder den CE-126P aus, oder drücken Sie **SHIFT** und **ENTER** ($P \leftrightarrow NP$) erneut (NON PRINT-Modus).

Rechenfehler

Folgende Fehlertypen treten bei gewöhnlichen Rechnern, Taschen- und Personalcomputern auf:

Fehler, die mit der Anzahl der Kommastellen zusammenhängen

Normalerweise ist die maximale Anzahl der Dezimalstellen, mit denen ein Computer rechnen kann, festgelegt. Zum Beispiel $4/3$ ergibt $1.3333333333...$ Bei einem Computer mit einem Maximum von 8 Stellen werden diese 8 Ziffern als signifikant bezeichnet, darüberhinausgehende Ziffern werden entweder ignoriert oder gerundet.

(Beispiel) Computer mit 10 signifikanten Stellen:

10 signifikante Ziffern
 $4/3$ **ENTER** $\rightarrow 1.3333333333...$
 ignoriert, gerundet

Aus diesem Grund weicht das errechnete Ergebnis von dem tatsächlichen Ergebnis durch den ignorierten oder gerundeten Betrag ab. (Dieser Unterschied macht dann die Abweichung aus.)

Der PC-1280 ermittelt ein 12stelliges Resultat (im Modus der doppelten Genauigkeit ein 24stelliges). Das Ergebnis wird gerundet und so verarbeitet, daß die Fehlerquote so weit als möglich minimiert wird und das angezeigte Ergebnis dem tatsächlichen weitgehend nahekommt.

(Beispiel im Modus der eingachen Genauigkeit) $4/3 \times 3$

4/3*3	ENTER	→ 4.	wird sukzessive verarbeitet
4/3	ENTER	→ 1.333333333	wird unabhängig verarbeitet
*3	ENTER	→ 3.999999999	

Während der Rechner die Operation sukzessive verarbeitet, erhält er das Ergebnis mit 12 Kommastellen, die er als Resultat akzeptiert und anschließend rundet.

Wenn der Rechner unabhängig arbeitet, wird der angezeigte Wert (10 Kommastellen) für die Berechnung benutzt.

(Beispiel im Modus der doppelten Genauigkeit) $4/3 \times 3$

4/3*3	ENTER	→ 4.	wird sukzessive verarbeitet
4/3	ENTER	→ 1.3333333333333333	wird unabhängig
*3	ENTER	→ 3.9999999999999999	verarbeitet

Wenn der Rechner die Operation sukzessive verarbeitet, erhält er das Ergebnis von $4/3$ mit 24 Kommastellen, die er als Resultat akzeptiert und rundet.

Wenn der Rechner unabhängig arbeitet, wird der angezeigte Wert (20 Stellen) für die Berechnung benutzt.

Fehler, die bei funktionsbestimmten Algorithmen auftreten

Der Computer benutzt eine Reihe von Algorithmen, um die Werte von Funktionen zu berechnen, wie z.B. Potenz- und trigonometrische Funktionen. Wenn in Rechnungen solche Funktionen eingesetzt werden, summiert sich die Fehlerquelle. Dieser Fehlerfaktor nimmt zu, je öfter diese Funktionen in die Rechnung eingesetzt werden. Der eigentliche Fehler für jede einzelne Funktion hängt von den eingesetzten Werten ab und ist besonders schwerwiegend bei singulären Stellen und Wendepunkten (d.h., wenn ein Winkel sich 90 Grad annähert, nähert sich die Tangente der Unendlichkeit). Ein Beispiel des Algorithmus ist hier unten für Potenzrechnungen angegeben.

(Beispiel) $60^6 =$

$60 \wedge 6$ **ENTER** $\rightarrow 4.665599999E10$

Obwohl 60^6 ist 4.6656×10^{10} , geht der Rechner von einem Wert (y^x) aus, der dem folgenden Ausdruck entspricht:

$$y^x = 10^{x \times \log y}$$

Mit anderen Worten: 60^6 wird als $10^{6 \times \log 60}$ berechnet.

PROGRAMMIER-BEDIENUNG

Kapitel 3 ist der Anwendung der BASIC Programmiersprache und den entsprechenden Ausdrücken für den PC-1280 gewidmet. Wir beginnen mit einer generellen Diskussion von Programmierkonzepten und kommen dann zu einer mehr speziellen Anwendung dieser Konzepte im PC-1280. Kapitel 3 endet mit einigen Abkürzungsvorschlägen zum Programmieren und mit Hinweisen zum Auffinden von Fehlern in Ihren Programmen.

PRO- (Programm), RSV- (Reserve) und RUN-Modus (der detailliert im KAPITEL 2 beschrieben wurde) werden in diesem Abschnitt behandelt.

8. BEGRIFFE UND AUSDRÜCKE DES BASIC

In diesem Kapitel wollen wir einige Begriffe und Ausdrücke der Programmiersprache BASIC kennenlernen.

Zeichenfolgen-Konstanten

Der PC-1280 ist in der Lage, außer Zahlen auch Buchstaben und spezielle Symbole in vielfacher Weise zu verarbeiten. Diese Buchstaben, Zahlen und speziellen Symbole werden Zeichen genannt.

Folgende Zeichen stehen dem PC-1280 zur Verfügung:

1234567890

ABCDEFGHIJKLMNOPQRSTUVWXYZ

abcdefghijklmnopqrstuvwxyz

! " # \$ % & ' () * + , - . / = $\sqrt{\quad}$ π @ ^ _ \ : ; < > ♠ ♥ ♦ ♣ []

Im BASIC wird eine Folge von Zeichen als Zeichenfolge bezeichnet. Damit der PC-1280 den Unterschied zwischen einer Zeichenfolge und anderen Programnteilen, wie z.B. Befehlen oder Variablen-Bezeichnungen, erkennen kann, muß man die Zeichen, die zu einer Zeichenfolge gehören, in Anführungszeichen (") einschließen.

Es folgen einige Beispiele für Zeichenfolgen-Konstanten:

"HELLO"

"Goodbye"

"SHARP COMPUTER"

Die folgenden Beispiele werden als Zeichenfolgen-Konstanten nicht akzeptiert:

"COMPUTER Anführungszeichen am Ende fehlen.

"ISN"T" Anführungszeichen dürfen nicht innerhalb einer Zeichenfolge benutzt werden.

Hexadezimalzahlen

Das Dezimalsystem ist nur eines von verschiedenen Zahlensystemen. Ein anderes, dessen Bedeutung im Zusammenhang mit Computern stark angewachsen ist, ist das Hexadezimalsystem. Das Hexadezimalsystem basiert auf der Zahl 16 statt auf der Zahl 10. Um hexadezimale Ziffern zu schreiben, benutzt man die Ziffern 0 bis 9, sowie sechs weitere "Ziffern": A, B, C, D, E und F. Diese entsprechen den Zahlen 10, 11, 12, 13, 14 und 15. Wenn Ihr PC-1280 eine Zahl als hexadezimal auffassen soll, setzen Sie ein UND-Zeichen (&) vor den betreffenden Ausdruck.

&A = 10
 &10 = 16
 &100 = 256
 &FFFF = 65535

Variablen

Computer sind aufgebaut aus einer Vielzahl von kleinsten Speichereinheiten, genannt Bytes. Jedes Byte kann man sich als einzelnes Zeichen vorstellen. Das Wort "Byte" erfordert beispielsweise vier Speicherbytes, weil es vier Buchstaben enthält. Um herauszufinden, wieviele Bytes zum Arbeiten zur Verfügung stehen, geben Sie einfach MEM ein und drücken **ENTER**. Die angezeigte Zahl gibt an, wieviel Bytes zum Programmieren frei sind. Dieses Verfahren funktioniert prima für Worte, ist aber zum Speichern von Zahlen sehr unzulänglich. Aus diesem Grund werden Zahlen in codierter Form gespeichert. Aufgrund dieser Codierung ist der Computer in der Lage, auch lange Zahlen in nur 8 Bytes (oder 14 Bytes im Modus der doppelten Genauigkeit) zu speichern. Die größtmögliche Zahl, die gespeichert werden kann, ist +9.999999999E +99 (oder 9.999999999999999999 D +99 in doppelter Genauigkeit).

Die kleinste Zahl ist 1. E -99 (oder 1.D -99). So erhalten Sie einen recht großen Zahlenbereich, innerhalb dessen Sie arbeiten können. Wenn jedoch das Ergebnis der Rechnung diesen Rahmen übersteigt, wird der Computer Ihnen dies mitteilen, indem er eine Fehlermitteilung auf dem Display ausgibt. (Eine Erklärung dieser Fehlermeldung ERROR finden Sie im Anhang A). Geben Sie nun zur Probe ein:

9 E 9 9 * 9 ENTER

9E99*9
ERROR2

Damit der Computer wieder normal weiterarbeiten kann, brauchen Sie lediglich die **C-CE**-Taste zu drücken. Aber wie speichert man nun soviel Information? Das ist ganz einfach. Der Computer zieht es vor, verschiedenen Datengruppen Namen zu geben. Wir wollen die Zahl 556 abspeichern. Sie können, wenn Sie möchten, diese Zahl nennen, aber für dieses Beispiel möchten wir ihr den Namen R geben. Die Anweisung LET kann benutzt werden, um den Computer zu veranlassen, einer Variablen einen Wert zuzuweisen, allerdings nur in einem Programm-befehl. Der LET-Befehl ist aber nicht unbedingt erforderlich, darum werden wir ihn nicht sehr oft benutzen. Geben Sie nun $R = 556$ ein und drücken Sie **ENTER**. Der Computer hat nun den Wert 556 mit dem Buchstaben R in Verbindung gebracht. Diese Buchstaben, die man benutzt, um Informationen zu speichern, nennt man Variable. Zum Abfragen des Inhalts der Variablen R drücken Sie die **C-CE**-Taste, **R** und die **ENTER**-Taste. Der Computer antwortet, indem er den Wert 556 rechts im Display anzeigt. Diese Möglichkeit kann sehr nützlich sein, wenn man Programme oder Formeln eingibt.

Wir wollen als nächstes die Variable R in einer einfachen Formel verwenden. In dieser Formel steht R für den Radius eines Kreises, dessen Fläche wir berechnen wollen. Die Formel der Kreisfläche lautet: $A = \pi * R^2$. Geben Sie ein

R SHIFT ^ 2 * SHIFT
π ENTER $R^2 * \pi$

971179.3866

Das Ergebnis ist 971179.3866.

Dieses Verfahren, Variablen zu verwenden, wird eingehender erklärt, wenn wir uns mit dem Programmieren beschäftigen.

Bislang haben wir uns nur mit numerischen Zeichen befaßt. Wie speichert man nun alphabetische Zeichen? Grundsätzlich ist das Prinzip das gleiche, aber damit der Computer den Unterschied zwischen den beiden Variablentypen erkennen kann, muß nun ein \$ zum Namen der Variablen gesetzt werden. Wir wollen z.B. das Wort BYTE unter der Variablen B\$ speichern. Sehen Sie das \$-Zeichen hinter dem B.

Dies sagt dem Computer, daß der Inhalt der Variablen B alphanumerisch, bzw. eine Datenzeichenfolge ist. Damit dies klarer wird geben Sie ein:

B \$ = " B Y T
E " ENTER

BYTE

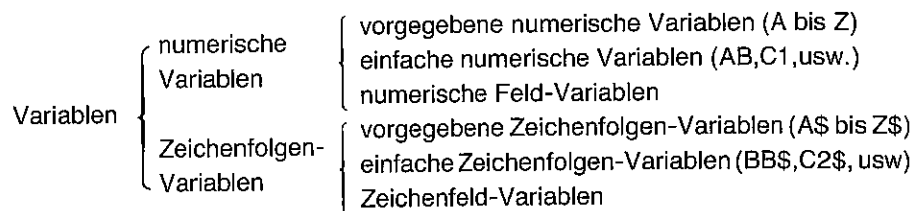
Der Wert BYTE ist nun unter der Variablen B\$ gespeichert. Um sich zu vergewissern, betätigen Sie **C CE** und tippen dann B\$ **ENTER**.

B \$ ENTER

BYTE

In der Anzeige steht BYTE diesmal auf der linken Seite der Anzeige, statt auf der rechten. Die Inhalte von Charakter-Zeichenfolgen oder Charakter-Variablen werden auf der ersten Zeile in der linken Ecke angezeigt.

Die Variablen, mit denen der SHARP PC-1280 arbeitet, sind folgendermaßen aufgegliedert:



Numerische Variable werden weiterhin eingeteilt in Variable mit einfacher Genauigkeit und doppelter Genauigkeit. Hierauf werden wir später noch genauer eingehen.

Vorgegebene Variablen

Der erste Bereich, die vorgegebene Variable, wird vom Computer grundsätzlich benutzt, um Daten zu speichern. Man kann ihn sich als reservierten Bereich denken. Mit anderen Worten, egal, wieviel Speicherplatz Ihr Programm in Anspruch nimmt, Ihnen stehen immer mindestens 26 Variable zur Speicherung von Daten offen. Diese Daten können entweder NUMERISCH oder in ZEICHENFOLGEN (Buchstaben)-Form gewählt werden. Vorgegebene Speicherplätze haben eine Kapazität von 8Byte und können immer nur einen Datetyp zur Zeit aufnehmen. Geben Sie zur Erklärung ein:

A = 1 2 3 ENTER**A = 123****123.**

Und dann:

A \$ ENTER**A\$
ERROR 9**

Sie erhalten die Fehlermeldung ERROR 9, was bedeutet, daß die numerischen Daten in den Speicherbereich mit dem Namen A gelegt haben und daß Sie dann dem Computer befohlen haben, diese Information als Zeichenfolge-Daten wieder abzurufen. Dies aber verwirrt den Computer, so daß er eine Fehlermeldung ausgibt. Drücken Sie die **C-CE**-Taste, um die Fehlermeldung zu beseitigen. Versuchen Sie nun das folgenden Beispiel:

**A \$ = " A B C
" ENTER****ABC**

Und dann:

A ENTER**A
ERROR 9**

Wieder ist der Computer verwirrt und meldet ERROR 9.

Einfache Variablen

Einfache Variablenbezeichnungen werden durch zwei (oder mehr) alphanumerische Zeichen charakterisiert, wie z.B. AA oder B1. Anders als feste Variablen haben die einfachen Variablen keinen fest im Speicher reservierten Bereich. Der Speicherbereich für einfache Variablen wird automatisch (im Programm oder im Datenbereich) bereitgestellt, sobald eine einfache Variable erstmalig benutzt wird.

Da für einfach numerische und einfache Zeichenfolge-Variablen verschiedene Sperrbereiche vorgesehen sind, können Variablen mit dem gleichen Namen (wie z.B. AB und AB\$) gleichzeitig benutzt werden.

Während alphanumerische Bezeichnungen für einfache Variablen-Namen verwendet werden können (bei alphabetischen Zeichen können nur Großbuchstaben verwendet werden), muß man als erstes Zeichen eines Variablen-Namens grundsätzlich einen Buchstaben wählen. Werden mehr als zwei Buchstaben zur Definition eines Variablen-Namens benutzt, sind nur die beiden ersten Zeichen von Bedeutung.

Achtung:

1. Die Namen von Funktionen oder BASIC-Befehlen dürfen für den PC-1280 nicht zur Bezeichnung von Variablen benutzt werden (z.B. PI, IF, TO, ON, SIN usw.).
2. Unter einer einzelnen Zeichenfolge-Variablen können bis zu 16 Zeichen oder Symbole gespeichert werden.

Feld-Variablen

In einigen Fällen ist es sinnvoll, Zahlen in organisierten Gruppen zu verarbeiten, z.B. eine Tabelle der Fußballergebnisse oder eine Steuertabelle. Im BASIC werden solche Gruppen Felder genannt. Ein Feld kann eindimensional sein, wie z.B. eine Liste, es kann aber auch zweidimensional sein, wie z.B. eine Tabelle.

Um ein Feld zu definieren, benutzt man den DIM-Befehl (Kürzel für Dimension). Felder müssen vor Gebrauch immer definiert werden. (Dies war nicht der Fall bei den Einwert-Variablen, die wir bislang benutzt haben.) Die Form für die Dimensionierung numerischer Felder ist:

DIM numerischer Variablen-Name (Größe)

Dabei bedeutet:

Numerischer Variablen-Name ist eine Bezeichnung der Variablen gemäß den oben besprochenen Benennungsregeln für numerische Variablen.

Größe ist eine Anzahl der Speicherplätze, die sich im Bereich zwischen 0 und 255 bewegen muß. Beachten Sie, daß wenn Sie eine Größenordnung eingeben, ein Speicherplatz mehr bereitgestellt wird.

Beispiele für mögliche DIMensionierungsbefehle sind:

DIM X(5)

DIM AA(24)

DIM Q5(0)

Der erste Befehl schafft ein Feld X mit 6 Speicherplätzen. Der zweite Befehl baut ein Feld AA mit 25 Speicherplätzen auf, der dritte ein Feld Q5 mit einem Speicherplatz, was im Grunde unsinnig ist (zumindestens für Zahlen), da man ebenso gut eine einwertige numerische Variable definieren könnte.

Es ist wichtig zu wissen, daß eine Feld-Variable X und eine Variable X sich unterscheiden und auch vom PC-1280 unterschieden werden. Das erste X bezeichnet eine Serie von numerischen Speicherplätzen, das zweite einen einzelnen und getrennte Speicherplatz.

Nachdem Sie nun wissen, wie man Felder aufbaut, mögen Sie sich fragen, wie man die einzelnen Speicherplätze anspricht. Da die gesamte Gruppe unter einem einzigen Namen abgelegt ist, sprechen wir einen einzelnen Speicherplatz ("Element" genannt) an, indem wir an den Namen der Gruppe eine Zahl in Klammern anschließen. Diese Zahl wird "Index" genannt. So müßte man z.B., um die Zahl 8 an fünfter Stelle in unserem (vorher definierten) Feld X unterzubringen, schreiben:

$$X(4) = 8$$

Wenn Sie die Zahl 4 verwirrt, bedenken Sie, daß die Nummerierung der Elemente in einem Feld mit Null beginnt und dann bis zu der in der DIM-Anweisung definierten Anzahl von Elementen fortläuft.

Der besondere Vorteil von Feldern liegt in der Möglichkeit, einen längeren Ausdruck oder eine Variable als Index zu benutzen.

Um ein Zeichenfeld zu DIMensionieren, muß man etwas anders vorgehen, als bei numerischen Feldern:

DIM Zeichenvariablen-Name (Größe)*Länge

Dabei bedeutet:

Zeichenvariablen-Name ist eine Bezeichnung der Variablen gemäß den oben besprochenen Benennungsregeln für normale Zeichenvariablen.

Größe ist die Anzahl der Speicherplätze (im Bereich zwischen 0 und 255). Beachten Sie, wenn Sie eine Größenordnung angeben, daß ein Speicherplatz mehr bereitgestellt wird.

***Länge** ist fakultativ. Wenn sie angegeben wird, spezifiziert sie die Länge der einzelnen im Feld enthaltenen Zeichenfolgen. Die Länge muß im Bereich von 1 bis 80 liegen. Wird die Länge nicht angegeben, erhalten die Zeichenfolgen automatisch eine Länge von 16 Zeichen.

Beispiele für mögliche DIMensionierungsbefehle für Zeichenfelder sind:

```
DIM XS(4)
DIM NMS(10)*10
DIM INS(1)*80
DIM RS(0)*26
```

Im ersten Fall wird ein Feld mit fünf Zeichenfolgen aufgebaut, in denen je 16 Zeichen gespeichert werden können. Der zweite DIM-Befehl baut ein Feld NM\$ auf mit elf Zeichenfolgen zu je 10 Zeichen.

Definiert man auch Zeichenfolgelängen, die kleiner als 16 Zeichen sind, spart man natürlich Speicherkapazität. Das dritte Beispiel zeigt ein Feld mit zwei Elementen zu je 80 Buchstaben an, und im letzten Fall wird eine einzige Zeichenfolge von 26 Zeichen aufgebaut.

Neben den einfachen Feldern, die wir gerade behandelt haben, läßt der PC-1280 auch "zweidimensionale" Felder zu. Ein eindimensionales Feld ist eine Liste von Daten, die in einer einzigen Spalte geordnet sind. Entsprechend ist ein zweidimensionales Feld eine Tabelle mit Zeilen und Spalten.

Zweidimensionale Felder werden folgendermaßen DIMensioniert:

DIM numerischer Variablen-Name (Zeilen, Spalten)

oder: **DIM** Zeichenvariablen-Name (Zeilen, Spalten)*Länge

DIM Zeichenvariablen-Name (Zeilen, Spalten)*Länge

Dabei bedeutet:

Zeilen gibt die Anzahl der Zeilen im Feld an. Dies muß eine Zahl zwischen 0 und 255 sein. Beachten Sie, daß, wenn Sie eine Größenordnung eingeben, eine Zeile mehr bereitgestellt wird.

Spalten gibt die Zahl der Spalten im Feld an. Dies muß eine Zahl zwischen 0 und 255 sein. Beachten Sie, daß, wenn Sie eine Größenordnung eingeben, eine Spalte mehr eingerichtet wird.

Die folgende Tabelle illustriert die Speicherplätze, die sich aus der Anweisung T(2,3) und den Indexen, die zu den jeweiligen Speicherplätzen gehören, ergeben:

	Spalte 0	Spalte 1	Spalte 2	Spalte 3
Zeile 0	T(0,0)	T(0,1)	T(0,2)	T(0,3)
Zeile 1	T(1,0)	T(1,1)	T(1,2)	T(1,3)
Zeile 2	T(2,0)	T(2,1)	T(2,2)	T(2,3)

Achtung: Zweidimensionale Felder nehmen viel Speicherplatz in Anspruch. Z.B. benötigt ein Feld mit 25 Zeilen und 35 Spalten 875 Speicherplätze!

Felder sind sehr nützliche Programmier-Hilfsmittel.

Die folgende Tabelle zeigt die Anzahl der Bytes, die zur Definierung jeder einzelnen Variable benötigt werden, sowie die Anzahl der für jeden einzelnen Programm-Befehl erforderlichen Nummer.

Variablen Typ	Nummer der verwendeten Bytes	
	Variablen-Name	Daten
Einfache Präzision Numerische Variable	7 Bytes	8 Bytes
Doppelte Präzision Numerische Variable	7 Bytes	14 Bytes
Zeichenfolge-Feld Variable	7 Bytes	*entsprechend der Feldgröße
Einfache Zeichenfolge- Variable (2 Zeichen)	7 Bytes	16 Bytes

- * Wenn z.B. DIM Z\$(2,3)*10 spezifiziert wird, werden 12 Variablen mit einer Speicherkapazität von je 10 Zeichen bereitgestellt. Dies erfordert 7 Bytes (Variablen-Name) + 10 Bytes (Anzahl der Zeichen) $\times$ 12 = 127 Bytes.

Element	Zeilennummer	Befehl & Funktion	ENTER & andere
Anzahl der verwendeten Bytes	3 Bytes	2 Bytes	1 Byte

Variablen der Form von A ()

Wird im Speicher des Computers ein Bereich für vorgesehene Variablen reserviert, kann dieser auch benutzt werden, um mit einem Index versehene Variablen, die die gleiche Form wie Feld-Variablen haben, zu definieren.

26 vorbestimmte Variablen-Namen stehen zur Verfügung, nämlich A bis Z (A\$ bis Z\$). Jeder dieser Namen kann als Index eine der Zahlen von 1 bis 26 tragen, wie z.B. A(1) — A(26) oder A\$(1) — A\$(26). Das bedeutet, daß die Variable A(1) anstelle der Variablen A benutzt werden kann, A(2) anstelle von B, A(3) anstelle von C usw.

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)

Wenn allerdings schon ein Feld mit dem Namen A oder A\$ durch einen DIM-Befehl definiert wurde, können Index-Variablen mit dem Namen A nicht zusätzlich definiert werden. Wurde beispielsweise durch DIM A(5) ein Feld A definiert, sind die Speicherplätze A(0) bis A(5) im Programm/Datenbereich reserviert. Wenn man dann eine Variable A(2) spezifiziert, bezieht sich diese nicht auf die vorbestimmte Variable B, sondern auf die Feld-Variable A(2) im Programm/Datenbereich. Spezifizieren Sie A(9), wird dies eine Fehlermeldung bewirken, da A(9) außerhalb des durch den DIM A(5)-Befehl bestimmten Bereiches liegt.

Wenn andererseits Index-Variablen der Form A() eingegeben worden sind, ist es möglich, gleichzeitig Felder A oder A\$ mit Hilfe eines DIM-Befehls einzurichten, es sei denn, die Definition der Index-Variablen wird mit dem CLEAR-Befehl gelöscht.

Wenn bei Verwendung des PC-1280 mit seiner totalen Speicherkapazität von mehr als 32KB durch MEM\$ die Größenstruktur "B" angezeigt wird, muß das eindimensionale Feld A\$(), bevor es benutzt werden kann, mit einer DIM-Anweisung bestimmt werden.

Sollen für Variablen der Form A() Indizes größer als 26 verwendet werden, werden die entsprechenden Speicherplätze im Programm/Datenbereich für diese A()-Variablen definiert, wenn ein Feld A nicht durch eine DIM-Anweisung definiert wurde. Wenn Sie z.B. A(35)=5 eingeben, werden die Speicherplätze A(27) bis A(35) im Programm/Datenbereich reserviert.

Da Variablen mit einem Index größer als 26 wie Feld-Variablen behandelt werden, unterliegen sie den folgenden besonderen Bestimmungen:

1. Speicherplätze für ein Feld mit demselben Namen müssen im Programm/Datenbereich nebeneinander liegen. Andernfalls erhalten Sie eine Fehlermeldung.

10: DIM B(2)

20: A(28)=5

30: BC=12

40: A(30)=9

Wenn dieses Programm eingegeben wird, befinden sich die Definitionen für das Feld "A" nicht in zwei aufeinanderfolgenden Segmenten des Programm/Datenbereichs und aus Zeile 40 wird sich ein Fehler ergeben.

[Programm/Datenbereich]

A(30)	}	Zeile 40
A(29)		
BC	}	Zeile 30
A(28)		
A(27)	}	Zeile 30
B(2)		
B(1)	}	Zeile 10
B(0)		

2. Numerische Feld-Variablen und Zeichenfolgen-Feld-Variablen mit demselben Index können nicht zur gleichen Zeit definiert werden. So können z.B. A(3) und A\$(3) nicht zur gleichen Zeit definiert werden, da sie im Programm-/Datenbereich den gleichen Platz einnehmen.
3. Zweidimensionale Felder können nicht definiert werden, ebensowenig ist es möglich, die Länge der Zeichenfolgen zu definieren, die in einer Feld-Variablen enthalten sein sollen. Die Länge der Buchstaben-Zeichenfolge, die von der Variablen A\$() erfaßt werden kann, ist beispielsweise auf sieben oder weniger Zeichen beschränkt.
4. Variablen mit dem Index (0) können nicht definiert werden. Wenn Sie A(0) oder A\$(0) definieren, werden Sie eine Fehlermeldung erhalten.
5. Wenn A(27) oder A\$(27) und größer als erste benutzt werden, werden 7 Bytes für den Variablen-Namen und 8 Bytes für jede Variable belegt.

Doppelt genaue Variable

Bestehende einfach genaue Variable können mit dem Nummernzeichen (#) in doppelt genaue Variable verwandelt werden. Zum Beispiel:

A#, AB#, X#(10), Y#(2,3) und X1#(5,6).

Doppelt genaue Variable bestehen aus 20 Stellen und einen zweistelligen Exponenten von -99 bis 99.

Achtung: Einfache Buchstaben-Variablen, an die das Nummernzeichen (#) angehängt wird (z.B. A#), sind keine vorgegebenen Variablen, sondern werden als doppelt genaue einfache numerische Variablen behandelt.

Die folgenden Variablenarten werden in getrennte Speicherbereiche eingegeben:

A	und A#
AB	und AB#
X(10)	und X#(10)

Variable können durch Anfügen eines Ausrufezeichens (!) als einfach genaue Variable (10 Stellen), oder durch Anhängen des Nummernzeichens (#) als doppelt genaue (20 Stellen) bezeichnet werden. Mit dem PC-1280 ist es jedoch möglich, jede numerische Variable mit den Anweisungen DEFSNG (einfach definiert) und DEFDBL (doppelt definiert) als einfach genaue oder doppelt genaue Variable zu behandeln. Dies ist besonders dann nützlich, wenn Ihr Programm zahlreiche doppelt genaue Variablen enthält.

Eingabe von Werten mit doppelt genauen Variablen

1. Benutzen Sie die Bestimmungszeichen (! und #)

$$AB! \text{ (oder } AB) = 1234567891234567891234 \rightarrow 1.234567891 \times 10^{21}$$

Der Wert wird mit 10 Stellen mit der einfach genauen Variablen AB! (oder AB) gespeichert.

$$AB\# = 1234567891234567891234 \rightarrow 1.2345678912345678912 \times 10^{21}$$

Der Wert wird mit 20 Stellen mit der doppelt genauen Variablen AB# gespeichert.

2. Verwenden Sie die Bestimmungszeichen und die Bestimmungs-Befehle (DEFSNG und DEFDBL)

10: DEFDBL B,D

20: DIM B(2)

25: D!= 123

30: DIM B!(2)

40: B(0)=9876543217876E5

50: B!(0)=B(0)

60: PRINT B(0): PRINT B!(0)

70: DEFSNG B,D

80: PRINT B#(0): PRINT B(0), D

[20] B wird in den Speicher als eine doppelt genaue Feld-Variable eingegeben (ohne DEFDBL-Anweisung in Zeile 10 würde Zeile 20 DIM B#(2) lesen).

[30] B! wird im Speicher als einfach genaue Variable eingegeben.

[50] Wenn die Inhalte der doppelt genauen Variable unter einer einfach genauen Variable eingegeben werden, ist die Anzahl der Stellen auf 10 beschränkt.

Bei der Ausführung des Programms, produziert Zeile 60 folgende Anzeige:

```
987654321787600000.
9.876543218E17
```

In Zeile 80 sieht das Display folgendermaßen aus:

```
987654321787600000.
9.87654E 17      123.
```

Gemischte einfache- und doppelt genaue Variablen

Enthält eine Berechnung doppelt genaue Variablen, so geht der PC-1280 bei Bedarf automatisch in den Modus der doppelten Genauigkeit.

Doppelte Genauigkeit wird in folgenden Fällen automatisch eingeschaltet:

Berechnungen mit 11 stelligen Werten und mehr:

z.B.: 1234567891234×5 .

Bei Verwendung des Buchstabens D zur Bezeichnung eines Exponenten:

z.B.: TAN 7.43D05

Zur Identifizierung von Werten mit dem Nummernzeichen (#):

z.B.: 4#/7

Bei der Verwendung doppelt genauer Variablen:

z.B.: AB# + BC.

Bei Verwendung des DEFDBL-Befehls:

z.B.: Tippen Sie im RUN-Modus DEFDBL ein. In der Statuszeile des Displays erscheint DBL.

5 / 9

5/9

D-01

z.B.:

$$A \ B \ \# \ = \ 5 \ \# \ / \ 9$$

AB# = 5# / 9_

ENTER

D-01

A B = A B # ENTER

$$AB = AB\#$$

5.555555556E-01

z.B.: DMS 1.99999999999999

85

Programme und Dateien

Die Anwendung von Programmen und Dateien steht im Mittelpunkt des Gebrauchs Ihres PC-1280. Dateien können im eingebauten Speicher, auf Diskette, auf Band oder als RAM-Dateien auf RAM-Karten gespeichert werden.

Mit dem PC-1280 stehen Ihnen zum Speichern von Programmen und Dateien eine große Auswahl von Medien zur Verfügung. Kassetten gehören dabei zu den billigsten und stehen in den meisten Fällen bereits zur Verfügung. Aber Ihre Verwendung ist zeitraubend und macht ein Bandgerät bzw. einen Rekorder erforderlich. SHARP's Taschen-Disketteneinheit (CE-140F) ist zum Band eine schnelle und bequeme Alternative. Der bedeutendste Vorteil liegt im schnellen Zugang zu den Dateien (in der gleichen Weise wie der Tonabnehmer eines Schallplattenspielers überall auf jede Rille einer Schallplatte gebracht werden kann). Aber am praktischsten von allem sind RAM-Karten. Diese erlauben die Speicherung von Programmen und Daten auf so kleinem Raum, daß mehrere RAM-Karten in einer Brusttasche untergebracht werden können. Außerdem arbeitet es sich mit ihnen zeitsparender, als mit jedem anderen Medium. Mit RAM-Karte wird Ihr PC-1280 zu einem der leistungsfähigsten Taschencomputer auf dem Markt.

Zur Benutzung von Dateien muß der BASIC-Programmierer dateibezogene Anweisungen kennen und verstehen, von denen es ungefähr 20 gibt. Auf dem PC-1280 werden eine Reihe in den Beispielen auf den folgenden Seiten verwendet. Da RAM-Dateien und Taschen-Diskette-basierte Dateien im Grunde in der gleichen Weise arbeiten, werden auch die Anweisungen in der gleichen Weise gegeben.

Dateinamen

Soll eine Datei in einem Medium, wie z.B. eine Taschen-Diskette oder eine RAM-Karte, gespeichert werden, so muß die Datei einen Namen bekommen (bei Sicherung auf Band mit dem CSAVE-Kommando ist das nicht notwendig). Dieser Name wird benutzt zum Laden der Programm-Dateien in den Computerspeicher oder zum Zugang zu Daten-Dateien in dem betreffenden Medium. Der Dateiname ist beliebig und kann aus bis zu 8 der folgenden Zeichen bestehen:

A—Z, a—z, 0—9, #, \$, %, &, ' , (,) , —, ^, @

Erweiterung

Die Erweiterung ist eine zusätzliche Art, Datei-Typen (wie z.B. BASIC-Programm-Dateien oder Text-Dateien) zu identifizieren. Die Erweiterung besteht aus drei Zeichen, die am Ende des Dateinamens und von ihr durch einen Punkt getrennt angehängt werden. Die Erweiterung wird spezifiziert, wenn eine Datei gesichert wird.

BASIC-Programme erhalten automatisch die Erweiterung .BAS, wenn sie mit dem SAVE-Kommando gesichert werden. Beim Zurückladen in den Speicher mit dem LOAD-Kommando muß die .BAS-Erweiterung nicht spezifiziert werden.

Wird mit den Anweisungen FILES oder LFILES ein Verzeichnis der in einem Gerät befindlichen Dateien aufgelistet, erscheinen die BASIC-Programme mit der .BAS-Erweiterung, unabhängig davon, ob irgendeine andere Erweiterung bei der Sicherung der Datei bestimmt wurde. Die .BAS-Erweiterung muß bei Verwendung des COPY-Kommandos immer bestimmt werden.

Gerätename

Da Dateien auf Band, Diskette oder RAM-Karten gespeichert werden können, muß auch das Gerät bestimmt werden, wenn der PC-1280 an mehr als ein Gerät angeschlossen wird. (Wenn kein Bestimmtes Gerät bezeichnet wurde, wird ein Bandgerät angenommen; ist kein Bandgerät oder keine RAM-Karte present, so wird eine Taschen-Diskette angenommen.) Der Dateiname muß von einem Doppelpunkt gefolgt sein (:). Es folgen die mit dem PC-1280 benutzen Gerätenamen.

F: RAM-Karte

X: Taschen-Disketteneinheit

Der komplette Datei-Deskriptor besteht also aus dem Gerätenamen, dem Dateinamen und der Erweiterung:

d:Dateiname.ext

In gewissen Fällen, wie oben beschrieben, kann der Gerätename und die Erweiterung ausgelassen werden.

Datei-Nummern

Datei-Nummern werden in Verbindung mit bestimmten Kommandos (z.B.: OPEN, INPUT#, PRINT#) zum Zugang in eine Daten-Datei benutzt. Die Nummern, die zusammen mit den Dateien benutzt werden, sind durch das Medium, auf das die Datei gespeichert ist, begrenzt:

Taschen-Diskette-basierte Dateien: #2 — #7

RAM-Disketten-basierte Dateien: #20 — #25

Arbeitskommandos für RAM- und Taschen-Disketten-basierte Dateien

CHAIN	Lädt und führt ein BASIC-Programm aus einem anderen Programm heraus aus.
CLOSE	Schließt eine Datei.
COPY	Kopiert eine Datei von Medium A auf Medium B, oder mit anderem Namen von Medium A auf Medium A.
DSKF	Zeigt die Anzahl der zur Speicherung auf dem bestimmten Medium zur Verfügung stehenden Bytes an.
EOF	Findet das Ende einer Datei.
FILES	Zeigt den Namen und die Attribute der auf dem bestimmten Medium gespeicherten Dateien an.
INPUT#	Liest von einer Daten-Datei die Variablen ein.
INIT	Formatiert und stellt das bezeichnete Medium auf Ausgangsposition.
KILL	Löscht auf dem bezeichneten Medium bestimmte Dateien.
LFILES	Druckt eine Liste von Dateien aus.

LOAD	Lädt ein Programm in den Speicher.
LOC	Zeigt die Position des Merkers innerhalb einer Datei an.
LOF	Zeigt die Größe einer bestimmten Datei an.
MERGE	Lädt ein Programm von einer Diskette ohne das (die) Programm(e) im Speicher zu löschen.
NAME	Ändert den registrierten Datei-Namen.
OPEN	Eröffnet den Zugang zu einer Datei.
PRINT #	Schreibt die Inhalte der bestimmten Variablen auf Diskette.
SAVE	Sichert ein BASIC-Programm auf Diskette.
SET	Weist zu oder nimmt Datei-Attribute weg.

Daten-Dateien

Es gibt zwei Typen von Daten-Dateien: aufeinanderfolgende Daten-Dateien und solche mit beliebigem Zugriff. Der PC-1280 unterstützt aufeinanderfolgende Daten-Dateien. Daten werden auf eine sequentielle Datei als eine Reihe von ASCII-Zeichen geschrieben, wobei eine Dateneinheit nach der anderen (sequentiell) in der Übertragungs-Ordnung gespeichert wird. Die Daten werden bei späterem Zugang aufeinanderfolgend wiedergegeben, eine Einheit nach der anderen.

Die folgenden Anweisungen und Funktionen werden für sequentielle Daten-Dateien in der Reihenfolge, in der sie hier auftreten, benutzt.

```

OPEN
PRINT #
INPUT #
EOF
LOC
LOF
CLOSE

```

Zur genaueren Information, wie diese Kommandos beim PC-1280 benutzt werden, lesen Sie das Kapitel COMPUTER-ANWEISUNGEN.

Erstellen einer sequentiellen Datei

Programm 1 ist ein kurzes Programm, das eine sequentielle Datei DATA erstellt, mit den Informationen, die Sie mittels Tastatur eingeben.

Programm 1 — Erstellung einer sequentiellen Datei

```

10: DIM DE$(1)*20
20: OPEN "F:DATA" FOR OUTPUT AS #20
30: CLS
40: INPUT "NAME: ";NA$
50: IF NA$="DONE" THEN 100
60: INPUT "DEPARTMENT: ";DE$(1)
70: INPUT "DATE HIRED: ";HI$
80: PRINT #20,NA$;" ";DE$(1);" ";HI$
90: GOTO 30
100: CLOSE #20
110: END

```

RUN

NAME: SAMUEL GOLDWYN

DEPARTMENT: AUDIO/VISUAL AIDS

DATE HIRED: 01/12/72

NAME: MARVIN HARRIS

DEPARTMENT: RESEARCH

DATE HIRED: 12/03/65

NAME: DEXTER HORTON

DEPARTMENT: ACCOUNTING

DATE HIRED: 04/27/81

NAME: DONE

Wie in Programm 1 gezeigt, sind folgende Schritte zum Erstellen einer sequentiellen Datei und zur Eingabe der Daten darin erforderlich:

1. Öffnen (OPEN) Sie die Datei zur Ausgabe (OUTPUT).
2. Mit der PRINT #-Anweisung schreiben Sie die Daten in die Datei.
3. Schließen (CLOSE) Sie die Datei und öffnen Sie sie erneut zum Lesen der Daten im INPUT-Modus.
4. Zum Lesen der Daten von der Datei in das Programm benutzen Sie den INPUT #-Befehl.

Das Lesen von Daten aus einer sequentiellen Datei

Schauen Sie sich jetzt Programm 2 an. Es geht in die Datei DATA hinein, die in Programm 1 erstellt wurde, und zeigt die Namen aller 1981 Angestellten an.

Programm 2 — Zugang in eine sequentielle Datei

```
10: DIM DE$(1)*20
20: OPEN "F:DATA" FOR INPUT AS #20
30: INPUT #20,NA$,DE$(1),HI$
40: IF RIGHT$(HI$,2)="81" THEN PRINT NA$
50: GOTO 30
```

RUN

```
DEXTER HORTON
ERROR 8 IN 30
```

Programm 2 liest, aufeinanderfolgend, jede Einheit der Datei und druckt die Namen aller 1981 Angestellten aus. Wenn alle Daten gelesen sind, verursacht Zeile 30 eine Fehlermeldung ERROR. Um diesen Fehler zu vermeiden, benutzen Sie die EOF-Funktion, die das Ende der Datei anfragt. Das überarbeitete Programm sieht folgendermaßen aus:

```

10: DIM DE$(1)*20
20: OPEN "F:DATA" FOR INPUT AS #21
25: IF EOF(21) THEN 60
30: INPUT #21,NA$,DE$(1),HI$
40: IF RIGHT$(HI$,2)="81" THEN PRINT NA$
50: GOTO 25
60: CLOSE #21
70: END

```

Hinzufügen von Daten in eine sequentielle Datei

Haben Sie eine Daten-Datei und wollen an ihr Ende weitere Daten hinzufügen, so können Sie die Datei nicht einfach im OUTPUT-Modus eröffnen und mit dem Schreiben der Daten beginnen. So bald Sie eine sequentielle Datei im Ausgabe-Modus öffnen, zerstören Sie alle darauf befindlichen Inhalte.

Benutzen Sie stattdessen den APPEND-Modus. Wenn die Datei nicht schon existiert, arbeitet der Öffnungsbefehl so als ob der OUTPUT-Modus bestimmt worden wäre.

Zum Hinzufügen von Daten in eine befindliche Datei mit Namen "FOLKS" gehen Sie auf folgende Weise vor:

Programm 3 — Hinzufügen von Daten in eine sequentielle Datei

```

110: DIM AD$(0)*40
120: OPEN "F:FOLKS" FOR APPEND AS #22
130: REM ADD NEW ENTRIES TO FILE
140: CLS
150: INPUT "NAME? ";NA$
160: IF NA$="00" THEN 230: REM 00 EXITS INPUT LOOP
170: INPUT "ADDRESS? ";AD$(0)
180: INPUT "BIRTHDAY? ";BI$
190: PRINT #22,NA$
200: PRINT #22,AD$(0)
210: PRINT #22,BI$
220: GOTO 140
230: CLOSE #22

```

Geben Sie als Antwort auf die Frage in Zeile 150 NAME? zweimal die Null (00) ein, damit das Programm dazu gebracht wird, aus der Eingabeschleife in Zeile 160 herauszuspringen. Mit der REM-Anweisung können Sie für sich private Programmhinweise notieren.

Ausdrücke

Ein Ausdruck ist eine Kombination von Variablen, Konstanten und Operatoren, die durch Rechenoperationen auf einen einzigen Wert zusammengefaßt werden kann. Die Rechenbeispiele, die Sie im Abschnitt 7 eingegeben haben, waren Beispiele für solche Ausdrücke. Ausdrücke sind ein wesentlicher Bestandteil von BASIC-programmen. Z.B. kann ein Ausdruck eine Formel sein, mit der das Ergebnis einer Gleichung errechnet wird, oder ein Test zur Bestimmung des Verhältnisses zwischen zwei Größen, ein Mittel um eine Reihe von Zeichenfolgen zu formatieren.

Numerische Operatoren

Der PC-1280 verfügt über fünf numerische Operatoren. Dies sind die arithmetischen Rechenzeichen, die Sie verwendet haben, als Sie die Benutzung des PC-1280 als Taschenrechner im Abschnitt 6 kennenlernten.

- + Addition
- Subtraktion
- * Multiplikation
- / Division
- ^ Exponentiation

Ein numerischer Ausdruck wird in der gleichen Weise konstruiert, wie Sie komplexe Rechenbefehle eingegeben haben. Numerische Ausdrücke können jede aussagefähige Kombination von numerischen Konstanten, numerischen Variablen und numerischen Operatoren beinhalten:

$(A*B) ^ 2$
 $A(2,3) + A(3,4) + 5.0 - C$
 $(A/B)* (C + D)$

Zeichenfolge-Ausdrücke

Zeichenfolge-Ausdrücke sind den numerischen Ausdrücken ähnlich, allerdings gibt es nur einen einzigen Zeichenfolgen-Operator: die Verkettung (+). Das benutzte Symbol ist dasselbe wie das Pluszeichen. Wird es mit einem Zeichenfolge-Paar benutzt, hüpft das + die zweite Zeichenfolge an das Ende der ersten an und schafft dadurch eine längere Zeichenfolge. Wenn Sie eine komplexere Zeichenfolgen-Verkettung und andere Zeichenfolgen-Operationen vornehmen, bedenken Sie bitte, daß der PC-1280 nur 80 Zeichen annimmt.

Achtung: Zeichenfolge-Einheiten und numerische Einheiten können nicht in demselben Ausdruck definiert werden, es sei denn, man benutzt eine der Funktionen, die Zeichenfolge-Werte in numerische Werte umwandeln oder umgekehrt:

"15" + 10 ist unzulässig

"15" + "10" ist "1510", nicht "25"

Verhältnis-Ausdrücke

Ein Verhältnis-Ausdruck vergleicht zwei Ausdrücke und gibt an, ob das festgestellte Verhältnis wahr oder unwahr ist. Die Verhältnis-Operatoren sind:

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

Die folgenden Ausdrücke werden als Verhältnis-Ausdrücke akzeptiert:

A < B

C(1,2) >= 5

D(3) <> 8

Wenn A gleich 10 wäre, B gleich 12, C(1,2) gleich 6 und D(3) gleich 9, wären alle diese Ausdrücke wahr.

Zeichenfolgen können ebenfalls mit Hilfe von Verhältnis-Ausdrücken verglichen werden. Die beiden Zeichenfolgen werden Zeichen für Zeichen gemäß dem Wert ihres ASCII-Codes verglichen (siehe Angang B: ASCII-Code-Tabelle). Ist eine Zeichenfolge kürzer als eine andere, wird 0 oder NULL in die freibleibenden Positionen eingesetzt. Alle folgenden Beispiele sind wahr:

"ABCDEF" = "ABCDEF"

"ABCDEF" <> "ABCDE"

"ABCDEF" > "ABCDE"

Verhältnis-Ausdrücke beurteilen nach wahr oder unwahr. Beim PC-1280 wird wahr durch 1 angegeben, unwahr durch eine 0. In jeder logischen Überprüfung wird ein Ausdruck, der 1 oder mehr ergibt, als wahr betrachtet, während jeder der 0 oder weniger ergibt, als unwahr verstanden wird. Gute Programmierpraktiken verlangen aber den Gebrauch eines expliziten Verhältnis-Ausdrucks statt sich auf diese Koinzidenz zu verlassen.

Aufgrund des internen Aufrundungsverfahrens des PC-1280 kann es bei Programmen mit einfach präzisen Zahlen von mehr als 10 Stellen bzw. doppeltpräzisen Zahlen von mehr als 20 Stellen auf beiden Seiten der Beziehung bei der Bewertung der Relativbeziehung zu Fehler kommen. Zur Lösung dieses Problems folgende Schritte befolgen:

1. Beliebigen numerischen, trigonometrischen oder anderen Ausdrücken der Relativbeziehung mit Hilfe der LET-Anweisung numerischen Variablen zuweisen.
2. Die zugeordneten numerischen Variablen in der Relativbeziehung wie gewohnt handhaben.

Beispiel 1:

10: IF SIN X >= 0 THEN 110

↓

10: LET Y = SIN X

15: IF Y >= 0 THEN 110

Beispiel 2:

20: IF A = B / 9 THEN 200

↓

20: LET C = B / 9

25: IF A = C THEN 200

Logische Ausdrücke

Logische Operationen benutzen zum Bau von Verbindungen zwischen Verhältnis-Ausdrücken die Funktionen der Boolischen Algebra AND (und), OR (oder), NOT (nicht) und XOR (X oder). Die logischen Operationen in einem einzigen Ausdruck werden nach arithmetischen- und Verhältnis-Operationen bewertet.

Auf diese Weise können logische Operatoren benutzt werden, die ihre Programmentscheidungen auf Grund mehrerer Bedingungen unter Verwendung der Anweisungen IF...THEN... treffen.

Beispiel:

```
IF A <= 32 AND B >= 90 THEN 150
```

Diese Anweisung bewirkt, daß die Ausführung auf Zeile 150 springt, wenn der Wert der numerischen Variable A kleiner oder gleich 32 ist, und wenn gleichzeitig der Wert der numerischen Variablen B größer oder gleich 90 ist.

```
IF X <> 13 OR Y = 0 THEN 50
```

Die Anweisung bewirkt, daß die Ausführung auf Zeile 50 springt, es sei denn die Variable X hat den Wert 13, oder wenn die Variable Y gleich 0 ist.

In einer logischen Operation, in der zwei Ziffern im Bereich +32768 bis -32767 involviert sind, werden die zwei Ziffern in 16Bit binäre ganze Zahlen umgewandelt (in die Komplementform der beiden) und die logische Verbindung wird dann für jedes entsprechende Bit-Paar in den zwei Ziffern bewertet.

Die Ergebnisse, angezeigt durch die logischen Operatoren für diese Bit-Bewertungen, werden im folgenden aufgelistet:

AND			OR			XOR			NOT	
X	Y	X AND Y	X	Y	Y OR X*	X	Y	X XOR Y	X	NOT X
1	1	1	1	1	1	1	1	0	1	0
1	0	0	1	0	1	1	0	1	0	1
0	1	0	0	1	1	0	1	1		
0	0	0	0	0	0	0	0	0		

*Y or X, nicht X OR Y, da der PC-1280 bei Ihrer Eingabe nicht zwischen der Eingabe X OR... und XOR unterscheiden kann.

Nachdem nach jedem Bit-Paar das entsprechende Ergebnis (eine 1 oder eine 0) entsprechend obiger Tabelle angezeigt wurde, wird die sich ergebende 16-Bit binäre Ziffer zurück in einen Dezimalwert verwandelt. Diese Ziffer ist das Ergebnis der logischen Operation.

Beispiel:

$$\begin{array}{rcl}
 41 \text{ AND } 27 \rightarrow & 41 = 101001 & \\
 \text{gleich} & 27 = 011011 & \text{AND} \\
 9 & \leftarrow 001001 &
 \end{array}$$

$$\begin{array}{rcl}
 41 \text{ OR } 27 \rightarrow & 41 = 101001 & \\
 \text{gleich} & 27 = 011011 & \text{OR} \\
 59 & \leftarrow 111011 &
 \end{array}$$

$$\begin{array}{rcl}
 41 \text{ XOR } 27 \rightarrow & 41 = 101001 & \\
 \text{gleich} & 27 = 011011 & \text{XOR} \\
 50 & \leftarrow 110010 &
 \end{array}$$

$$\begin{array}{rcl}
 \text{NOT } 3 \rightarrow & 3 = 0000000000000011 & \text{NOT} \\
 \text{gleich} & & \\
 -4 \text{ (Komplement} & \leftarrow 111111111111100 & \\
 \text{Form der beiden)} & &
 \end{array}$$

NOT X kann generell berechnet werden durch die Gleichung $\text{NOT } X = -(X + 1)$.

Klammerung und Vorrang der Operatoren

Bei der Bearbeitung komplexer Ausdrücke folgt der PC-1280 einer Reihe vordefinierter Prioritäten, die bestimmen, in welcher Reihenfolge die Operatoren bearbeitet werden. Das kann sehr wichtig sein:

$$\begin{array}{lcl}
 5 + 2 * 3 \text{ kann sein} & & \\
 5 + 2 = 7 \quad \text{oder} \quad 2 * 3 = 6 & & \\
 7 * 3 = 21 \quad \quad \quad 6 + 5 = 11 & &
 \end{array}$$

Die genauen Regeln des "Operatoren-Vorrangs" werden im Anhang D beschrieben.

Damit Sie sich nicht alle diese Regeln merken müssen, und damit Sie Ihre Programme eindeutiger gestalten, benutzen Sie immer Klammern, um die Reihenfolge der Bearbeitung von Operatoren vorzugeben. Das obige Beispiel wird eindeutig, wenn Sie schreiben:

$$(5 + 2) * 3 \quad \text{oder} \quad 5 + (2 * 3)$$

RUN-Modus

Im allgemeinen kann jeder der oben genannten Ausdrücke ebenso im RUN-Modus benutzt werden wie in der Programmierung einer BASIC-Anweisung. Im RUN-Modus wird ein Ausdruck sofort ausgewertet und angezeigt, z.B.:

(5 > 3) A N D
(2 < 6) E N T E R

(5>3)AND(2<6)

1.

Die 1 gibt an, daß der Ausdruck wahr ist.

Funktionen

Funktionen sind bestimmte Bestandteile der BASIC-Sprache, die einen Wert in einen anderen Wert transformieren. Funktionen arbeiten wie Variablen, deren Wert durch andere Variablen oder Ausdrücke festgelegt wurde. ABS ist eine Funktion, die den absoluten Wert der Zahl hervorbringt, auf die sie angewendet wird.

ABS(-5) ist 5

ABS(6) ist 6

LOG ist eine Funktion, die den dekadischen Logarithmus der Zahl berechnet, auf die LOG angewendet wird.

LOG(100) ist 2

LOG(1000) ist 3

Eine Funktion kann überall dort verwendet werden, wo auch Variablen benutzt werden können. Viele Funktionen erfordern keine Klammerung:

LOG 100 ist dasselbe wie LOG(100)

Sie müssen jedoch Klammern benutzen, sobald sich eine Funktion auf mehr als eine Zahl bezieht. In Kapitel 4, Numerische Funktionslisten, finden Sie die auf dem PC-1280 verfügbaren Funktionen.

9. PROGRAMMIEREN DES PC-1280

Im vergangenen Kapitel haben wir einige Begriffe und Ausdrücke der Programmiersprache BASIC kennengelernt. In diesem Kapitel wollen wir nun diese Elemente benutzen, um auf dem PC-1280 Programme zu schreiben. Wir möchten noch einmal darauf hinweisen, daß dieses Handbuch nicht als Einführung in die BASIC-Programmierung verstanden werden soll. Dieses Kapitel soll Sie lediglich in den besonderen Gebrauch des BASIC auf dem PC-1280 einführen.

Programme

Ein Programm besteht aus einer Reihe von an den Computer gerichteten Befehlen. Denken Sie daran, daß der PC-1280 nur eine Maschine ist, die genau die Operation durchführt, die sie angeben. Sie als Programmierer sind dafür verantwortlich, daß korrekte Befehle eingegeben werden.

BASIC-Anweisungen

Der PC-1280 setzt Programme entsprechend einem bestimmten Format um. Dieses Format wird Anweisung genannt. Sie geben die BASIC-Anweisungen immer nach einem bestimmten Muster ein. Anweisungen müssen mit einer Zeilennummer beginnen:

```
10: INPUT A  
20: PRINT A * A  
30: END
```

Zeilennummern

Jede Programmzeile muß eine eigene Nummer haben, und zwar muß diese eine ganze Zahl zwischen 1 und 65279 sein. Zeilennummern sind die Bezugspunkte des Computers. Sie geben dem PC-1280 an, in welcher Reihenfolge er ein Programm abarbeiten muß. Es ist nicht erforderlich, daß Sie die Programmzeilen folgerichtig eingegeben haben (obwohl das sicher wenig verwirrender sein dürfte, besonders wenn Sie noch Anfänger sind). Der Computer beginnt beim Durchar-

beiten eines Programms immer mit der niedrigsten Zeilennummer und arbeitet die folgende in aufsteigender Reihenfolge ab.

Mit dem AUTO-Kommando können Sie automatisch Zeilennummern einfügen. Jedes mal, wenn Sie die **ENTER**-Taste drücken, wird eine neue Zeilennummer, in der richtigen aufsteigenden Reihenfolge, automatisch eingefügt. Eine vollständige Beschreibung dieser nützlichen Funktion finden Sie im Abschnitt COMPUTER-ANWEISUNGEN.

Beim Programmieren ist es sinnvoll, genug Raum für spätere Einschübe zwischen den einzelnen Zeilen zu lassen (10, 20, 30, ... 10, 30, 50, usw.). Dies ermöglicht den Einschub von zusätzlichen Zeilen, falls notwendig.

VORSICHT: Benutzen Sie keine gleiche Zeilennummern in verschiedenen Programmen, die Sie später mischen wollen.

Wenn Sie gleiche Zeilennummern benutzen, wird die ältere Zeile gelöscht, wenn Sie eine neue mit derselben Nummer eingeben.

BASIC-Kommandos

Alle BASIC-Anweisungen müssen Befehle enthalten. Sie sagen dem Computer, welche Operation er durchführen soll. Ein Befehl ist ein Programmbestandteil, insofern wird die Operation nicht direkt darauf erfolgen.

Einige Anweisungen erfordern oder erlauben den Gebrauch eines Operanden:

```
10: DATA "HELLO"
20: READ B$
30: PRINT B$
40: END
```

Operanden informieren den Computer darüber, auf welche Daten sich die vom Befehl angeordnete Operation bezieht. Einige Befehle erfordern Operanden, bei anderen Befehlen sind sie fakultativ. Einige Befehle erlauben keine Operanden. (Im Abschnitt COMPUTER-ANWEISUNGEN finden Sie die BASIC-Befehle und ihre Anwendung auf dem PC-1280.)

Achtung: Befehle, Kommandos und Funktionen müssen in Großbuchstaben eingegeben werden.

Direkt-Kommandos

Direkt-Kommandos sind Anweisungen an den Computer, die außerhalb eines Programms eingegeben werden. Sie fordern den Computer auf, augenblickliche Handlungen mit dem Programm vorzunehmen oder einen bestimmten Modus zu setzen, der dann wieder die Art der Programmbearbeitung determiniert.

Direkt-Kommandos haben unmittelbare Wirkung — sobald Sie die Eingabe des Direkt-Kommandos beendet haben (durch Betätigen der **ENTER**-Taste), wird die Anweisung ausgeführt. Direkt-Kommandos gehen keine Zeilennummern voraus:

RUN
NEW
RADIAN

Einige Kommandos können sowohl im direkten- als auch im Programm-Modus verwendet werden. (Im Abschnitt COMPUTER-ANWEISUNGEN finden Sie die BASIC-Kommandos und ihren Gebrauch auf dem PC-1280.)

Modi

Sie erinnern sich sicherlich, daß Sie, als Sie den PC-1280 als Rechner benutzt haben, im CAL- oder im RUN-Modus gearbeitet haben.

Der RUN-Modus wird ebenfalls gebraucht, um die von Ihnen geschriebenen Programme abzuarbeiten.

Der PROgramm-Modus wird gewählt, wenn Sie Programme eingeben oder aufarbeiten wollen.

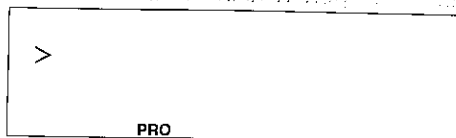
Der ReSeRve-Modus ermöglicht es, vordefinierte Zeilenfolgen-Variablen zu kennzeichnen und zu speichern. Weiterhin wird er für komplexere Programmierung gebraucht.

Der Anfang des Programmierens mit dem PC-1280

Nachdem Sie in vielen Übungen den PC-1280 als Taschenrechner benutzt haben, sind Sie sicherlich schon recht vertraut mit der Tastatur. Von nun an werden wir, wenn wir eine Eingabe zeigen, nicht mehr jede Tastenbedienung angeben. Denken Sie immer daran, **SHIFT** zu betätigen, um Zugang zu den Zeichen oberhalb der Tasten zu bekommen und vergessen Sie nicht am Ende jeder Zeile die Taste **ENTER** zu betätigen.

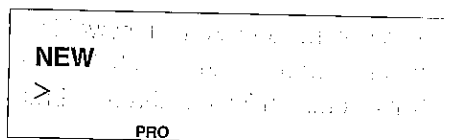
Nun können Sie mit dem Programmieren beginnen!

Um Programmieranweisungen in den Computer einzugeben, schalten Sie den Computer erst in den PROgramm-Modus, indem Sie die **BASIC**-Taste benutzen. Auf dem Display erscheint dann folgende Anzeige:



Geben Sie das NEW-Kommando ein.

N E W ENTER

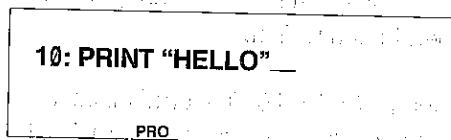


Das NEW-Kommando löscht aus dem Arbeitsspeicher des PC-1280 alle vorhandenen Programme und Daten. Das Bereitschaftssymbol erscheint, wenn Sie die **ENTER**-Taste gedrückt haben. Es zeigt an, daß der Computer nun Eingaben erwartet:

Beispiel 1 — Eingabe und Ablauf eines Programms

Vergewissern Sie sich, daß der PC-1280 sich im PRO-Modus befindet, und geben Sie das folgende Programm ein:

10 PRINT
"HELLO"



Beachten Sie, daß der PC-1280 bei Druck auf **ENTER** automatisch zwischen die Nummer und das Kommando einen Doppelpunkt setzt.

Überprüfen Sie, daß die Anweisung das richtige Format hat und ändern Sie dann durch Druck auf die **BASIC**-Taste den Modus auf den RUN-Modus.

C C E R U N ENTER

RUN
HELLO
RUN

Da dies die einzige Programmzeile ist, wird der Computer an dieser Stelle stoppen. Drücken Sie **ENTER**, um aus dem Programm herauszukommen und geben Sie RUN vor, wenn das Programm noch einmal ablaufen soll.

Beispiel 2 — Aufbereitung eines Programms

Nehmen wir an, Sie wollen den Inhalt dessen, was Ihr Programm anzeigt, verändern, d.h. Sie wollen Ihr Programm aufbereiten. Bei einem einzeiligen Programm können Sie die Eingabe schlicht wiederholen, aber wenn Sie komplexere Programme schreiben, wird die Aufbereitung zu einem wichtigen Bestandteil des Programmierens. Wir wollen das Programm, das Sie gerade geschrieben haben aufbereiten.

Sind Sie noch im RUN-Modus? Wenn ja, so schalten Sie zurück in den PROGRAMM-Modus.

Um Ihr Programm aufbereiten zu können, müssen Sie es nun wieder abrufen. Benutzen Sie dazu die **↑**-Taste. Sofern Ihr Programm vollständig durchgelaufen war, wird die **↑**-Taste lediglich die letzte Programmzeile abrufen. Falls ein Fehler im Programm vorlag oder Sie die Ausführung mit **BREAK** unterbrochen hatten, wird die **↑**-Taste die Zeile abrufen, in der der Fehler liegt, oder in der **BREAK** betätigt wurde. Um Veränderungen in Ihrem Programm vorzunehmen, bedienen Sie sich der **↓**-Taste, um sich nach unten zu bewegen (d.h. die jeweils nächste Zeile abzurufen). Wenn die **↑**- und die **↓**-Tasten festgehalten werden, rollt das Programm im Display nach oben bzw. unten, d.h. die einzelnen Zeilen werden nacheinander durch das Display laufen.

Sie erinnern sich sicherlich, daß Sie den Cursor innerhalb einer Programmzeile, die jeweils in der obersten Zeile des Displays angezeigt wird, mit dem  (rechter Pfeil) und dem  (linker Pfeil) bewegen können.

Mit Hilfe des -Pfeiles setzen Sie nun den Cursor auf das erste Zeichen, das Sie ändern möchten.

Achtung: Obwohl auf dem Display mehrere Zeilen angezeigt werden können, kann der Cursor nur in der obersten Zeile bewegt werden. Um eine andere Zeile aufzuarbeiten, muß die gewünschte Zeile mit der -Taste nach oben verschoben und dann aufbereitet werden.



```
10 PRINT "HELLO"
```



```
10 PRINT "HELLO"
```

Beachten Sie, daß der Cursor die Form eines blinkenden Rechtecks angenommen hat, um so anzuzeigen, daß er auf einem schon bestehenden Zeichen steht. Geben Sie folgendes ein:

```
G O O D B Y E " !
```

```
10 PRINT "GOODBYE"!_
```

Vergessen Sie nicht, am Zeilenende **ENTER** zu drücken. Schalten Sie um in den RUN-Modus.

```
R U N ENTER
```

```
RUN  
ERROR 1 IN 10
```

Dies ist ein neuer Typ von Fehlermeldung. Der Fehler ist nicht nur identifiziert (unser alter Freund der Syntax-Fehler), sondern auch die Nummer der Zeile in der er auftritt, wird angegeben.

Drücken Sie **C CE**, um die Fehlermeldung zu beseitigen.

Schalten Sie zurück in den PRO-Modus. Sie müssen unbedingt im PROgramm-Modus sein, um Änderungen im Programm vornehmen zu können. Mit Hilfe des  -Pfeils (oder ) rufen Sie die Zeile, in der sich der Fehler befindet, ab.

 (oder )

```
10 PRINT "GOODBYE!"
```

Der blinkende Cursor steht auf der kritischen Stelle. Im Abschnitt 8 haben Sie gelernt, daß Sie, wenn sie Zeichenfolgen-Konstanten in BASIC eingeben, alle dazugehörigen Zeichen in Anführungszeichen einschließen müssen. Benutzen Sie die DEL-Taste, um das "!" zu löschen.

DEL

```
10 PRINT "GOODBYE" _
```

Wir wollen nun das Ausrufezeichen (!) an die richtige Stelle setzen. Wenn man Programme aufbereitet, werden die DEL- und die INS-Funktionen genauso benutzt wie beim Korrigieren einfacher Rechenoperationen (siehe Abschnitt 7). Mit Hilfe des -Pfeiles bringen Sie den Cursor auf der der Einfügung folgende Zeichen.



```
10 PRINT "GOODBYE" _
```

Drücken Sie die INS-Taste. Ein  markiert die Stelle, an der ein neues Zeichen einzugeben ist:

INS

```
10 PRINT "GOODBYE"  "
```

Geben Sie das ! ein. Die Anzeige sieht nun folgendermaßen aus:



```
10 PRINT "GOODBYE!"
```

Denken Sie daran **ENTER** zu drücken, damit die Korrektur in das Programm aufgenommen wird.

- Achtung:**
1. Wenn Sie eine ganze Zeile aus Ihrem Programm löschen möchten, geben Sie nur die Zeilennummer ein und die ursprüngliche Zeile wird gelöscht. Mit dem DELETE-Kommando können Sie mehr als eine Zeile auf einmal löschen.
 2. Werden mit dem Programm-Modus Daten eingegeben, ohne daß der Cursor angezeigt ist, werden die entsprechenden Zeichen gewöhnlich in der linken Spalte der obersten Displayzeile angezeigt. Benutzen Sie dagegen die - oder -Taste, wenn das Bereitschaftssymbol angezeigt wird, werden die Inhalte Ihrer sukzessiven Eingaben anstelle des Bereitschaftssymbols angezeigt.

Beispiel 3 — Gebrauch von Variablen beim Programmieren

Wenn Ihnen der Gebrauch numerischen und Zeichenfolgen-Variablen nicht vertraut ist, lesen Sie noch einmal die entsprechenden Abschnitte in Kapitel 8.

Der Gebrauch von Variablen bei Programmieren erlaubt Ihnen eine sehr differenzierte Ausnutzung des PC-1280.

Erinnern Sie sich, daß Sie einfache numerische Variablen mit Buchstaben von A bis Z belegen:

A = 5

Um Zeichenfolgen-Variablen zu bestimmen, benutzen Sie ebenfalls einen Buchstaben, gefolgt von einem Dollarzeichen. Benutzen Sie niemals den gleichen Buchstaben, um eine numerische und eine Zeichenfolge-Variable zu bestimmen. Sie können nicht A und A\$ im selben Programm benutzen.

Erinnern Sie sich, daß einfache Zeichenfolge-Variable nicht mehr als 7 Zeichen umfassen können:

A\$ = "TOTAL"

Die einer Variablen zugewiesenen Werte können sich im Laufe eines Programms ändern, sie können jeweils den im Programm eingegebenen oder errechneten Wert annehmen. Eine Möglichkeit der Variablenzuweisung ist, den Befehl INPUT zu benutzen. Im folgenden Programmbeispiel ändert sich der Wert der Variablen A\$ aufgrund der Daten, die eingegeben werden, um auf die Frage "WORD?" zu beantworten.

Geben Sie dieses Programm ein:

```
10: INPUT"WORD?";A$
20: B = LEN(A$)
30: PRINT"WORD IS ";B;" LETTERS"
40: END
```

Da Zeile 30 des Programms länger als 24 Zeichen ist, wird der verbleibende Teil in der folgenden Zeile angezeigt.

Das zweite neue Element in diesem Programm ist der Gebrauch der END-Anweisung, um die Beendigung des Programms anzuzeigen. END läßt den Computer wissen, daß das Programm beendet ist. Es ist ein Vorteil beim Programmieren, immer eine END-Anweisung zu benutzen,

Wenn Ihre Programme komplexer werden, möchten Sie sie möglicherweise noch einmal durchsehen, bevor Sie sie arbeiten lassen. Zu diesem Zweck benutzen Sie das LIST-Kommando. LIST, das nur im PRO-Modus benutzt werden kann, bringt das gesamte Programm, angefangen bei der niedrigsten Zeilennummer, zur Anzeige.

Versuchen Sie dieses Programm aufzulisten:

LIST ENTER

```
10: INPUT "WORD?";A$
20: B= LEN (A$)
```

Benutzen Sie dazu die Pfeile  und , um sich in Ihren Programmen zu bewegen, bis Sie das gesamte Programm durchgesehen haben. Um eine Zeile noch einmal anzusehen, die mehr als 24 Zeichen enthält, bringen Sie den Cursor an die

äußerste rechte Stelle des Displays und die weiteren Zeichen werden in der Anzeige erscheinen. Nachdem Sie Ihr Programm kontrolliert haben, lassen Sie es ablaufen:

C C E R U N ENTER

**RUN
WORD?__**

H E L P

**RUN
WORD?HELP__**

ENTER

**WORD?HELP
WORD IS 4. LETTERS**

Dies ist das Ende Ihres Programms. Natürlich können Sie es noch einmal ablaufen lassen, indem Sie RUN eingeben. Auf jeden Fall wäre das Programm unterhaltsamer, wenn es mehr als eine Eingabemöglichkeit enthielte. Wir wollen das Programm nun so modifizieren, daß es weiterläuft, ohne daß man nach jeder Antwort RUN eingeben muß.

Gehen Sie zurück in den PRO-Modus und benutzen Sie den Auf- oder Abwärts-Pfeil (oder LIST), um in Zeile 40 zu kommen. Betätigen Sie den Auf- oder Abwärts-Pfeil, bis sich die Zeile 40 am oberen Rand des Bildschirms befindet oder Sie tippen:

L I S T 4 0 ENTER

40: END

Sie können nun 40 eingeben, um so die gesamte Zeile zu löschen. Sie können aber auch den Cursor  auf das E in END setzen. Ändern Sie die Zeile 40 folgendermaßen:

40: GOTO 10

Nun lassen Sie das modifizierte Programm ablaufen (RUN):

Die GOTO-Anweisung schafft eine Programm-Schleife (d.h. eine bestimmte Operation wird immer wieder durchgeführt). Da Sie der Schleife kein Ende gesetzt haben, wird sie sich nicht unterbrechen (eine "endlose" Schleife). Um dieses Programm zu unterbrechen, drücken Sie die **BREAK**-Taste.

Wenn Sie ein Programm mit der **BREAK**-Taste unterbrochen haben, können Sie es mit dem CONT-Kommando neu starten. CONT steht für CONTINUE (fortsetzen). Bei Eingabe des CONT-Kommandos läuft das Programm von da aus weiter, wo es mit **BREAK** unterbrochen wurde.

Beispiel 4 — Komplexere Programmierung

Das folgende Programm berechnet die N-Fakultät (N!). Das Programm beginnt mit 1 und berechnet N! bis zu der Grenze, die Sie eingeben. Geben Sie das Programm ein:

```
100: F = 1: WAIT 118
110: INPUT "LIMIT?";L
120: FOR N = 1 TO L
130: F = F*N
140: PRINT N,F
150: NEXT N
160: END
```

In diesem Programm sind einige neue Erscheinungen enthalten. Der Befehl WAIT in Zeile 100 kontrolliert den Zeitraum, über den eine Zeile im Display bleibt, ehe das Programm fortgesetzt wird. Die Zahlen und ihre Fakultäten werden angezeigt, wenn sie berechnet werden. Durch die WAIT-Anweisung bleiben sie etwa 2 Sekunden in der Anzeige, statt darauf zu warten, daß Sie **ENTER** drücken.

Ebenfalls in Zeile 100 tauchen zwei Anweisungen auf, die durch einen Doppelpunkt (:) getrennt werden. Sie können so viele Anweisungen in eine Zeile schreiben, wie Sie möchten, solange Sie die einzelnen Anweisungen durch Doppelpunkte voneinander trennen und die maximale Anzahl von 80 Zeichen pro Zeile, einschließlich **ENTER**, nicht überschreiten. Solche mehrfach-Anweisungen machen allerdings das Lesen und Schreiben eines Programms schwieriger. Es ist deshalb besser, sie nur zu benutzen, wenn die Einzel-Anweisungen sehr kurz und einfach sind oder wenn es einen besonderen Grund dafür gibt, Mehrfach-Anweisungen zu verwenden.

In unserem Programm haben wir in Zeile 120 den Befehl FOR und in Zeile 150 den Befehl NEXT benutzt, um eine Schleife aufzubauen. In Beispiel 3 hatten Sie eine "endlose" Schleife geschaffen, die sich kontinuierlich wiederholte. Mit dieser FOR/NEXT-Schleife addiert der PC-1280 jedesmal 1 zu N hinzu, wenn der Programmablauf den NEXT-Befehl erreicht. Hierbei wird geprüft, ob N die anfangs gesetzte Grenze überschreitet. Ist N kleiner oder gleich der Grenze, beginnt der Programmdurchlauf wieder am Anfang der Schleife. Ist N größer als die Grenze, geht der Programmablauf in Zeile 160 weiter und wird dort beendet.

In einer FOR/NEXT-Schleife können Sie jede numerische Variable benutzen. Sie müssen auch nicht mit Schritt 1 anfangen, und Sie können bei jedem Programmschritt jeden beliebigen Betrag dazu addieren. Näheres dazu finden Sie unter den COMPUTER-ANWEISUNGEN.

Wir haben dieses Programm mit Zeilennummern von 100 aufwärts belegt. Eine solche Belegung mit unterschiedlichen Zeilennummern macht es möglich, daß man mehrere Programme zur gleichen Zeit im Arbeitsspeicher hat. Um dieses Programm anstelle des Programms der Zeile 10 durchlaufen zu lassen, geben Sie folgendes ein:

C·CE
R U N 1 0 0

Außer der Möglichkeit Programme durchlaufen zu lassen, indem Sie die erste Zeilennummer eingeben, können Sie den Programmen auch einen Buchstaben als Namen geben und sie mit der **DEF**-Taste starten.

Das Speichern von Programmen im PC-1280

Sie erinnern sich sicherlich, daß Einstellungen, ReSeRve-Tasten und Funktionen im Computer erhalten bleiben, auch wenn Sie ihn abschalten. Programme bleiben ebenfalls im Speicher erhalten, wenn Sie den PC-1280 abschalten oder er das automatisch tut (mit AUTO OFF). Selbst wenn Sie die Tasten **BREAK**, **C·CE** oder **CA** betätigen, bleiben die Programme erhalten.

Programme gehen nur dann aus dem Speicher verloren, wenn Sie einen der folgenden Abläufe ausführen:

- NEW eingeben, ehe Sie in der PRO-Betriebsart mit dem Programmieren beginnen.
- Den Computer mit dem ALL RESET-Schalter initialisieren.
- Ein neues Programm mit DENSELBEN ZEILENNUMMERN schreiben, die Sie bereits an ein im Speicher vorhandenes gegeben haben.

Diese kurze Einführung in das Programmieren mit dem PC-1280 sollte dazu dienen, die aufregenden Möglichkeiten des Programmierens mit Ihrem neuen Computer näher zu beschreiben. Für weitere Übungen, siehe Programmbeispiele.

Abkürzungen

Der PC-1280 bietet Ihnen verschiedene Möglichkeiten, die Programmierung zu vereinfachen, indem Sie die Anzahl der Tastenbedienungen und anderer sich wiederholender Abläufe reduzieren.

Eine dieser Möglichkeiten besteht darin, Befehle und Kommandos abzukürzen oder die automatische Zeilennummerierung mit dem Kommando AUTO (siehe dazu COMPUTER-ANWEISUNGEN).

In diesem Abschnitt sollen vor allem zwei Möglichkeiten erörtert werden, mit deren Hilfe Sie unnötige Tastenbedienung vermeiden können, nämlich die Benutzung der **DEF**-Taste und die des Reserve-Modus.

Die **DEF**-Taste und benannte Programme

Es wird häufiger vorkommen, daß Sie mehrere verschiedene Programme zur gleichen Zeit im Speicher des PC-1280 abspeichern wollen. (Vergessen Sie nicht verschiedene Zeilennummern zu vergeben.) Um ein Programm mit einem RUN- oder einem GOTO-Befehl zu starten, müssen Sie sich normalerweise an die erste Zeilennummer eines jeden Programms erinnern. Aber es gibt eine einfachere Möglichkeit! Sie können jedes Programm mit einem einzelnen Buchstaben benennen und das Programm dann mit nur zwei Tastenbedienungen mit Hilfe von **DEF** starten.

Setzen Sie den Kennbuchstaben in die erste Zeile des Programms, auf das Sie sich später beziehen wollen. Der Kennbuchstabe muß in Anführungszeichen eingeschlossen und von einem Doppelpunkt gefolgt sein.

```
10: "A": PRINT "FIRST"
```

```
20: END
```

```
80: "B": PRINT "SECOND"
```

```
90: END
```

Jedes der folgenden Zeichen kann als Kennbuchstabe benutzt werden: A, S, D, F, G, H, J, K, L, =, Z, X, C, V, B, N, M und SPC. Wie Sie sehen, sind dies die Tasten der beiden unteren Reihen der Tastatur.

Um das Programm ablaufen zu lassen, brauchen Sie nicht RUN 80 oder GOTO 10 einzugeben, sondern Sie drücken nur die **DEF**-Taste und dann den Kennbuchstaben. Im obigen Beispiel würde, wenn Sie **DEF** und dann **B** drücken würden, ein "SECOND" in der Anzeige erscheinen.

Wenn Sie **DEF** benutzen, um ein Programm zu starten, werden Variablen und Modus-Einstellungen ebenso verarbeitet, als wenn Sie GOTO benutzen würden. Für weitere Einzelheiten siehe COMPUTER-ANWEISUNGEN.

10. RESERVE-Modus

Ein weiterer Weg, den PC-1280 zeitsparend zu bedienen, ist der RSV-Modus.

Innerhalb des Speichers des PC-1280 sind 144 Zeichen für den "Reserve-Speicher" vorgesehen. Sie können diesen Speicherteil benutzen, um häufig gebrauchte Ausdrücke zu speichern. Um diese wieder abzurufen, brauchen Sie nur zwei Tasten zu bedienen.

Versuchen Sie das folgende Beispiel für das Speichern und Wiederabrufen einer reservierten Zeichenfolge nachzuvollziehen.

Schalten Sie den PC-1280 auf den RSV-Modus, indem Sie die **SHIFT**- und **BASIC**-Tasten drücken. Beachten Sie, daß die Anzeige "RUN" und "PRO" verschwindet und die Nachricht RSV auf dem Bildschirm erscheint.

Geben Sie NEW ein und drücken Sie dann die **ENTER**-Taste. Dadurch löschen Sie alle vorher gespeicherten Daten so, wie Sie mit NEW im PRO-Modus alle vorher gespeicherten Programme löschen.

Geben Sie **SHIFT** gefolgt von **A** ein:

SHIFT A

A: _

RSV

Beachten Sie, daß A links in der Anzeige erscheint und von einem Doppelpunkt gefolgt wird.

Geben Sie das Wort PRINT ein und betätigen Sie **ENTER**:

P R I N T ENTER

A: PRINT

RSV

Ein Leerschritt erscheint nach dem Doppelpunkt und zeigt an, daß PRINT jetzt in Reserve-Speicher unter dem Buchstaben A gespeichert worden ist.

Schalten Sie den PC-1280 jetzt auf PROGramm-Modus um. Geben Sie NEW gefolgt von **ENTER** ein, um den Programmspeicher zu löschen. Geben Sie 10 als Zeilennummer ein und betätigen Sie dann die **SHIFT**- und die **A**-Taste:

1 0 SHIFT A

10 PRINT

ENTER

10: PRINT

Sofort wird hinter der Zeilennummer das Wort "PRINT" in der Anzeige erscheinen.

Im Reserve-Speicher kann jede beliebige Zeichenfolge gespeichert werden. Die gespeicherten Zeichenfolgen können jederzeit im PROGramm- oder im RUN-Modus wieder abgerufen werden, indem Sie **SHIFT** drücken und anschließend den Buchstaben, unter dem Sie die Zeichenfolge gespeichert haben. Die dazu zur Verfügung stehenden Tasten sind dieselben, wie im Zusammenhang mit **DEF** erwähnt, also alle Tasten in den unteren zwei Reihen der Tastatur.

Die maximale Länge einer Reservezeichenfolge für eine Taste ist 78 Bytes, einschließlich des Tastennamens.

Um eine gespeicherte Zeichensequenz aufzuarbeiten, schalten Sie den RSV-Modus ein und drücken Sie **SHIFT** gefolgt vom Buchstaben, unter dem die Sequenz gespeichert ist. Mit Hilfe der **◀**-, **◀**-, **DEL**- und **INS**-Tasten können Sie nun genauso aufbereiten, wie in anderen Modi.

Ist das letzte Zeichen in einer abgespeicherten Sequenz "@", so wird es beim Abrufen der Sequenz als **ENTER** interpretiert. Speichern Sie beispielsweise die Zeichenfolge "GOTO 100@" unter der **G**-Taste, so wird bei Betätigung von

SHIFT und **G** im RUN-Modus augenblicklich das Programm in Zeile 100 beginnen. Ohne das "@" Zeichen muß nach Betätigung von **SHIFT** und **G** **ENTER** gedrückt werden, um das Programm zu starten.

Das Löschen von Reserveprogrammen

1. Wie Sie wissen, löscht die Tastenfolge **NEW** und **ENTER** den gesamten Reservespeicher. Denken Sie daran, daß die oben beschriebene Operation im RSV-Modus durchgeführt werden muß.
2. Um einen Reservespeicher zu löschen, benutzen Sie die **SPC**- oder **DEL**-Taste, wie unten beschrieben:

Beispiel: Löschen Sie **A*A**, was unter Zeilennummer **S** gespeichert ist.

SHIFT S

S: _

RSV

A * A ENTER

S:A*A

RSV

C CE

>

RSV

SHIFT S

S:A*A

RSV

◀ oder ▶

S:A*A

RSV

DEL DEL DEL

S: _____

RSV

ENTER

S: _____

RSV

11. FEHLERSUCHE

Wenn Sie ein neues BASIC-Programm eingeben, wird dieses in der Regel beim ersten Startversuch nicht laufen. Selbst wenn Sie ein Programm nur 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 den aufwärts (↑) oder abwärts (↓) Pfeil, um die fehlerhafte Zeile ins Display zu rufen. Der Cursor wird sich an der Stelle befinden, wo der PC-1280 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) einen Leerraum erzeugen, wenn A den Wert 1 hat. 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:

1. Überprüfen Sie das Programm Zeile für Zeile unter Verwendung von LIST und der ↑ 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!
2. Versuchen Sie jede einzelne Zeile beim Durchlesen 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.

3. Fügen Sie eine oder mehrere zusätzliche PRINT-Anweisungen in das 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 eines Programms schon ausgearbeitet wurden. Sie können den Programmablauf auch an kritischen Stellen vorübergehend mit STOP unterbrechen, um dann einzelne Variable zu überprüfen.
4. Verwenden Sie TRON (Trace ON) und TROFF (Trace OFF) als Direkt-Kommandos innerhalb des Programms, 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.

Zum Fortsetzen des Programms drücken Sie einmal die **↑**-Taste. Daraufhin wird die nächste Programmzeile ausgeführt und ihre Zeilennummer angezeigt. Um die Programmzeile zu überprüfen, können Sie dann die **↓**-Taste betätigen. Sie können auch die Inhalte der einzelnen Variablen überprüfen, indem Sie die Variablennamen eingeben und **ENTER** drücken:

A ENTER (wenn A = 4 vor der Operation **A ENTER** eingegeben wird)

A ENTER

A	4.
---	----

Zur Abarbeitung jeder einzelnen Programmzeile ist es notwendig, die **↓**-Taste zu drücken. Wenn Sie den normalen Ablauf von Zeile zu Zeile unterbrechen möchten, drücken Sie die **ENTER**-Taste. Wenn Sie es sich anders überlegt haben, und das Programm fortführen möchten, geben Sie das CONT-Kommando.

Ein Beispiel unseres Hypothenuse-Programms finden Sie auf der nächsten Seite.

Input	Display
	>
T R O N	TRON__
ENTER	>
R U N	RUN__
ENTER	?
3	3. _
ENTER	?
4	4. _
ENTER	10:
↑	10 INPUT A, B
↓	20:
↑	20 A = A * A : B = B * B
A	A. _
ENTER	9.
B	B. _
ENTER	16.
↓	30:
H	H. _
ENTER	5.
↓	HYPOTENUSE = 5.
↑	40 PRINT "HYPOTENUSE = "; H
↓	40:
↓	>

Unabhängig davon, wie vorsichtig Sie auch Programmieren, wird es vorkommen, daß Sie ein Programm schreiben, das nicht das macht, was Sie von ihm erwarten. Dann geht es darum, den Fehler zu finden. BASIC hat deshalb eine spezielle Programmablauf-Routine, den "Trace"-Modus, entwickelt. Im Trace-Modus zeigt der PC-1280 die Zeilennummer jeder Programmzeile an und stoppt nach der Ausführung jeder einzelnen Zeile. Dadurch haben Sie die Möglichkeit, die Reihenfolge der Instruktionen bei Ihrer Ausführung zu verfolgen. Wenn das Programm nach der Ausführung einer Zeile anhält, können Sie die Werte der Variablen überprüfen oder ändern.

TRON (Trace ON) startet den Trace-Modus. Die TRON-Anweisung kann als Direkt-Kommando (im RUN-Modus) oder auch innerhalb eines Programms als Befehl verwendet werden. Wird TRON als Direkt-Kommando verwendet, informiert es den PC-1280, daß eine Ablaufverfolgung während der Ausführung aller nachfolgenden Programme benötigt wird. Die zu verfolgenden Programme werden dann ganz normal gestartet, mit Hilfe des GOTO- oder RUN-Kommandos.

Wird TRON innerhalb eines Programms verwendet, so wird der Trace-Modus erst beim Erreichen der entsprechenden Zeile eingeschaltet. Wenn aus irgendeinem Grund die Zeile nie erreicht wird, bleibt der Trace-Modus ineffektiv.

Nach dem Einschalten bleibt der Trace-Modus effektiv, bis er von einem TROFF-Befehl ausgeschaltet wird. TROFF kann in einem Programm auch als Direkt-Kommando benutzt werden. Den Trace-Modus kann man außerdem durch die folgende Tastenfolge ausschalten:

CA
SHIFT C CE

Um ein Beispiel für die Benutzung des Trace-Modus zu haben, geben Sie bitte das folgende Programm ein, das die Länge der Hypotenuse in einem rechtwinkligen Dreieck bei gegebener Länge der beiden anderen Seiten bestimmt:

Programmliste:

```
10: INPUT A,B
20: A = A * A: B = B * B
30: H =  $\sqrt{A + B}$ 
40: PRINT "HYPOTENUSE = ";H
```

Im RUN-Modus geben Sie das TRON-Kommando gefolgt vom RUN-Kommando. Beachten Sie, daß der INPUT-Befehl wie gewohnt abgearbeitet wird, indem ein Fragezeichen für jeden eingegebenen Wert angezeigt wird. Sobald Sie zwei Werte eingegeben haben, erscheint die Zeilennummer der INPUT-Anweisung.

10:

Wenn Sie die **↑**-Taste drücken und festhalten, können Sie die ganze Zeile überprüfen:

10 INPUT A,B

COMPUTER-ANWEISUNGEN

Kapitel 4 enthält eine alphabetische Liste aller BASIC-Anweisungen für den PC-1280. Dieser Abschnitt dient zum Nachschlagen und ist als solcher zum bequemeren Gebrauch besonders markiert.

Der erste Teil der markierten Seiten enthält eine alphabetische Liste der numerischen Funktionen und Pseudovariablen.

Der zweite Teil dieser Seiten besteht aus einer alphabetischen Liste aller anderen BASIC-Anweisungen.

12. WISSENSCHAFTLICHE UND MATHEMATISCHE BERECHNUNGEN

Der PC-1280 ist mit zahlreichen eingebauten Funktionen für wissenschaftliche, mathematische und statistische Berechnungen ausgestattet. Eine alphabetische Liste folgt weiter unten. All diese Funktionen können bei der Anwendung des PC-1280 im RUN-Betrieb (NICHT im CAL-Modus) als Teil von Berechnungen benutzt werden, bzw. auch als BASIC-Anweisungen in einem Programm.

Für trigonometrische Funktionen können Alt-Grad, Minuten oder als entsprechender Neu-Grad-Wert eingegeben werden:

ALT-GRAD: Stellen Sie den PC-1280 auf den Alt-Grad-Eingabe-Modus durch Tippen von DEGREE (auf der Statuszeile im Display erscheint DEG); dies ist die Grundeinstellung.

BOGENMASS: Stellen Sie den PC-1280 in die Bogenmaß-Eingabe-Modus durch Tippen von RADIAN (auf der Statuszeile im Display erscheint RAD).

NEU-GRAD: Stellen Sie den PC-1280 in den Neu-Grad-Eingabe-Modus durch Tippen von GRAD (auf der Statuszeile im Display erscheint GRAD).

Diese drei Modi (DEG, RAD und GRAD) können auch innerhalb eines Programms eingestellt werden. Ist ein Modus gesetzt, müssen alle trigonometrischen Funktionen der Einstellung des Computers angepaßt werden (Alt-Grad-, Bogenmaß- oder Neu-Grad-Werte), bis der Modus von Hand oder innerhalb eines Programmes geändert wird. Die Modus-Einstellung wird selbst während dem Ausschaltzustand gespeichert. Die folgenden Beispiele gelten für Direkt-Eingabe der in Alt-Grad eingegebenen Funktionen.

Mit D markierte Funktionen können der Betriebsart mit doppelter und einfacher Stellengenauigkeit verwendet werden. Die mit S gekennzeichneten Funktionen können nur mit einfacher Stellengenauigkeit verwendet werden, obwohl die internen Berechnungen mit doppelter Stellengenauigkeit durchgeführt werden.

ABS**|x|****Funktion:** Absoluter Wert**D**

Anmerkungen: Zeigt den absoluten Wert des numerischen Arguments an. Der absolute Wert ist der Wert einer Ziffer unabhängig von seinem Vorzeichen. $ABS-10$ ist 10.

Beispiel: $ABS-10$ **ENTER**

10

ACS **$\cos^{-1}x$** **Funktion:** Reziprok oder Arcuscosinus**D**

Anmerkungen: Zeigt den Arcuscosinus des numerischen Arguments an. Der Arcuscosinus ist der Winkel, dessen Cosinus gleich dem Ausdruck ist. Der Wert ist abhängig von der Betriebsart (Dezimalgrad, Bogenmaß oder Neu-Grad).

Beispiel: $DEGREE$ **ENTER**
 $ACS-0.5$ **ENTER**

120.

AHC **$\cosh^{-1}x$** **Funktion:** Reziproker hyperbolischer Cosinus**D**

Anmerkungen: Zeigt den Arc-hyperbolischen Cosinus des numerischen Arguments an.

Beispiel: $AHC10$ **ENTER**

2.9932228456

AHS $\sinh^{-1}x$ **Funktion:** Reziproker hyperbolischer Sinus D**Anmerkungen:** Zeigt den Arcus-hyperbolischen Sinus des numerischen Arguments an.**Beispiel:** AHS27.3 **ENTER** 4.000369154**AHT** $\tanh^{-1}x$ **Funktion:** Reziproker hyperbolischer Tangens D**Anmerkungen:** Zeigt den Arcus-hyperbolischen Tangens des numerischen Arguments an.**Beispiel:** AHT0.7 **ENTER** 8.673005277E-01**ASN** $\sin^{-1}x$ **Funktion:** Reziprok oder Arcussinus D**Anmerkungen:** Zeigt den Arcussinus des numerischen Arguments an. Der Arcussinus ist der Winkel zwischen dem der Sinus gleich dem Ausdruck ist. Der angezeigte Wert ist abhängig vom eingestellten Modus des Geräts (Alt-Grad, Bogenmaß oder Neu-Grad).**Beispiel:** DEGREE **ENTER**
ASN0.5 **ENTER** 30.

ATN $\tan^{-1}x$ **Funktion:** Reziprok oder Arcustangens**D****Anmerkungen:** Zeigt den Arcustangens des numerischen Arguments an. Der angezeigte Wert ist abhängig vom eingestellten Modus des Geräts (Alt-Grad, Bogenmaß oder Neu-Grad).**Beispiel:** DEGREE **ENTER**
ATN1 **ENTER**

45.

COS $\cos x$ **Funktion:** Cosinus**D****Anmerkungen:** Zeigt den Cosinus des Winkelarguments an. Der angezeigte Wert ist abhängig vom eingestellten Modus des Geräts (Alt-Grad, Bogenmaß oder Neu-Grad).**Beispiel:** DEGREE **ENTER**
COS120 **ENTER**

-0.5

CUB x^3 **Funktion:** Kubik**D****Anmerkungen:** Zeigt die Kubikzahl des Arguments an.**Beispiel:** CUB3 **ENTER**

27.

CUR

$\sqrt[3]{x}$

Funktion: Kubikwurzel S

Anmerkungen: Zeigt die Kubikwurzel des Arguments an:

Beispiel: CUR125 **ENTER** 5.

DECI

Funktion: Umformung von hexadezimal auf dezimal S

Anmerkungen: Wandelt einen hexadezimalen Wert in einen dezimalen um.

Beispiel: DECIF82 **ENTER** 3970.

DEG

dd°mm'ss"

Funktion: Umwandlung von Alt-Grad/Minuten/Sekunden in eine dezimale Form S

Anmerkungen: Wandelt das Argument eines Winkels in der DMS-Form (Alt-Grad, Minuten, Sekunden) in die DEG-Form (Dezimal-Grad) um. In der DMS-Form repräsentiert die ganze Zahl die Gradzahl, die erste und zweite Stelle die Minuten und die dritte und vierte Stelle hinter dem Komma repräsentieren die Sekunden. Jede weitere Stelle gibt Dezimalsekunden an.

Beispiel: DEG30.5230 **ENTER** 30.875

DMS

ddd.ddd°

Funktion: Umwandlung Dezimal-Grad in Grad/Minuten/Sekunden **S**

Anmerkungen: Wandelt das Winkelargument in DEG auf DMS um. (siehe DEG)

Beispiel: DMS124.8055 **ENTER** 124.48198 (124°48'19"8)

EXP e^x

Funktion: Exponentialfunktion **D**

Anmerkungen: Zeigt den Wert von e (2.718281828 — die Basis des natürlichen Logarithmus) erhoben zum Wert des numerischen Arguments an.

Beispiel: EXP1.2 **ENTER** 3.320116923

FACT $n!$

Funktion: Faktor n **D**

Anmerkungen: Zeigt den Faktor des eigenen Arguments an.

Beispiel: FACT7 **ENTER** 5040.

HCS $\cosh x$

Funktion: Hyperbolischer Cosinus **D**

Anmerkungen: Zeigt den hyperbolischen Cosinus des numerischen Arguments an.

Beispiel: HCS3 **ENTER** 10.067662

HEX

Funktion: Umwandlung von dezimal auf hexadezimal **D**

Anmerkungen: Wandelt einen dezimalen Wert in eine hexadezimale Form um.

Beispiel: HEX7820 **ENTER** 1E8C.

HSN**sinh x**

Funktion: Hyperbolischer Sinus **D**

Anmerkungen: Zeigt den hyperbolischen Sinus des numerischen Arguments an.

Beispiel: HSN4 **ENTER** 27.2899172

HTN**tanh x**

Funktion: Hyperbolischer Tangens **D**

Anmerkungen: Zeigt den hyperbolischen Tangens des numerischen Arguments an.

Beispiel: HTN0.9 **ENTER** 7.162978702E-01

INT

Funktion: Ganze Zahl **D**

Anmerkungen: Zeigt die ganze Zahl des eigenen Arguments an. Die ganze Zahl von PI ist 3.

Beispiel: INT-1.9 **ENTER** -2.

LN $\log_e x$ **D****Funktion:** Natürlicher Logarithmus**Anmerkungen:** Zeigt den Logarithmus der Basis e (2.718281828...) des eigenen numerischen Arguments an.**Beispiel:** LN2 **ENTER** 6.931471806E-01**LOG** $\log_{10} x$ **D****Funktion:** Logarithmus mit der Basis 10**Anmerkungen:** Zeigt den Logarithmus der Basis 10 seines eigenen numerischen Arguments an.**Beispiel:** LOG1000 **ENTER** 3.**NCR** ${}_n C_r = n! / r!(n-r)!$ **S****Funktion:** Kombination**Anmerkungen:** Gibt die Werte als $NCR(n,r)$ ein**Beispiel:** NCR(6,3) **ENTER** 20.**NPR** ${}_n P_r = n! / (n-r)!$ **S****Funktion:** Permutation**Anmerkungen:** Gibt den Wert als $NPR(n,r)$ ein**Beispiel:** NPR(6,3) **ENTER** 120.

PI π **Funktion:** PI

Anmerkungen: PI ist eine numerische Pseudovariablen mit dem Wert PI. Die Anwendung von PI ist identisch mit dem Gebrauch der π -Taste. Der Wert von PI ist in der einfachen Genauigkeit (DEFSNG-Modus) auf 10 Stellen genau, im Modus der doppelten Genauigkeit (DEFDBL) 20 Stellen.

Beispiel:DEFDBL **ENTER**PI **ENTER**

3.1415926535897932385

POL $(x,y) \rightarrow (r,\theta)$

Funktion: Umwandlung einer rechtwinkligen in eine polare Koordinate **S**

Anmerkungen: Wandelt die numerischen Argumente in einem rechrwinkligen Koordinaten-Format in ein polares Koordinaten-Format um.

Das erste Argument bezeichnet die Entfernung von der y-Achse und die Sekunden von der x-Achse. Die Werte werden umgekehrt, Die Entfernung und der Winkel in den polaren Koordinaten werden den festen Variablen Y und Z zugeordnet. Der umgewandelte Winkel hängt ab von der Einstellung des PC-1280 (Alt-Grad, Bogenmaß oder Neu-Grad).

Beispiel:DEGREE **ENTER**POL(8,6) **ENTER**Z **ENTER**

10. (r=10)

36.86989765

(θ≈36.9°)

$\wedge$	y^x
----------	-------

Funktion: x-ste POWER

D

Anmerkungen: Zeigt die Potenz eines numerischen Werts an.

Beispiel: $4^{2.5}$ **ENTER**

32.

RCP	$1/x$
------------	-------

Funktion: Reziprok

D

Anmerkungen: Zeigt die Reziproke ihres numerischen Arguments an.

Beispiel: RCP4 **ENTER**

0.25

REC	$(r,\theta) \rightarrow (x,y)$
------------	--------------------------------

Funktion: Umwandlung polarer in rechtwinklige Koordinaten

S

Anmerkungen: Wandelt numerische Argumente im polaren Koordinaten-Format in ein rechtwinkliges Koordinaten-Format um.

Das erste Argument bezeichnet das Entfernungs- und das Sekunden-Argument des Winkels. Die Winkelangabe hängt ab von der Einstellung des PC-1280 (DEG, RAD oder GRAD). Die umgekehrten Werte, die Entfernungen von der x-Achse und der y-Achse, werden den festen Variablen x und y zugewiesen.

Beispiel: DEGREE **ENTER**
 REC(12,30) **ENTER**
 Z **ENTER**

10.39230485 (X = 10.4)
 6. (y=6)

RND**Funktion:** Beliebige Ziffer**Anmerkungen:** Siehe RND und RANDOM in den COMPUTER-ANWEISUNGEN**ROT** $\sqrt[x]{y}$ **Funktion:** x-ste Wurzel**Anmerkungen:** Zeigt die x-ste Wurzel des eigenen Arguments an. Geben Sie sie als yROTx ein.**Beispiel:** 7776ROT5 **ENTER**

6.

SGN**Funktion:** Zeichen des Arguments

Anmerkungen: Zeigt den auf das Zeichen des Arguments basierenden Wert an.

Wenn $x > 0$, wird 1 angezeigt.

Wenn $x < 0$, wird -1 angezeigt.

Wenn $x = y$, wird 0 angezeigt.

SIN $\sin x$ **Funktion:** Sinus**Anmerkungen:** Zeigt den Sinus des Winkelarguments an. Der angezeigte Wert hängt von der Einstellung des PC-1280 ab (Alt-Grad, Bogenmaß oder Neu-Grad).

Beispiel: DEGREE **ENTER**

SIN30 **ENTER**

0.5

SQR $\sqrt{x}$ **Funktion:** Quadratwurzel**D**

Anmerkungen: Zeigt die Quadratwurzel des Arguments an. Benutzen Sie entweder das SQR-Kommando oder Das Wurzelzeichen ($\sqrt{\quad}$) auf der Tastatur.

Beispiel: SQR3 **ENTER** 1.732050808

SQU x^2 **Funktion:** Quadrat**2**

Anmerkungen: Zeigt das Quadrat des Arguments an.

Beispiel: SQR4 **ENTER** 16

TAN $\tan x$ **Funktion:** Tangens**D**

Anmerkungen: Zeigt den Tangens des Winkelarguments an. Der angezeigte Wert hängt von der Einstellung des PC-1280 ab (Alt-Grad, Bogenmaß oder Neu-Grad).

Beispiel: DEGREE **ENTER**
TAN45 **ENTER** 1.

TEN

 10^x **Funktion:** Antilogarithmus

D

Anmerkungen: Zeigt den Wert von 10 (die Basis des normalen Logarithmus) erhoben auf den Wert des eigenen numerischen Arguments.**Beispiel:** TEN3 **ENTER**

1000.

Rechenbereiche

Numerische Berechnungen:

Für Rechenoperationen mit x muß x in einem der zwei unten aufgeführten Bereiche liegen.

$$-1 \times 10^{100} < x \leq -1 \times 10^{-99} \text{ für negatives } x$$

$$10^{-99} \leq x < 10^{100} \text{ für positives } x$$

Der angezeigte Wert für x wird von der Anzahl der möglichen Stellen auf dem Display begrenzt.

Funktionen:

Funktion	Bereich von x
sin x	DEG: $ x < 1 \times 10^{10}$ Einfachgenauigkeit
cos x	$ x < 1 \times 10^{20}$ Doppelgenauigkeit
tan x	RAD: $ x < \frac{\pi}{160} \times 10^{10}$ Einfachgenauigkeit
	$ x < \frac{\pi}{160} \times 10^{20}$ Doppelgenauigkeit
	GRAD: $ x < \frac{10}{9} \times 10^{10}$ Einfachgenauigkeit
	$ x < \frac{10}{9} \times 10^{20}$ Doppelgenauigkeit
	Ebenso nur für tan x :
	DEG: $ x \neq 90 (2n-1)$
	RAD: $ x \neq \frac{\pi}{2} (2n-1) \quad (n=\text{ganzzahlig})$
	GRAD: $ x \neq 100 (2n-1)$
$\sin^{-1}x$	$-1 \leq x \leq 1$
$\cos^{-1}x$	
$\tan^{-1}x$	$ x < 1 \times 10^{100}$
$\sinh x$	$-227.9559242 \leq x \leq 230.2585092$ Einfachgenauigkeit
$\cosh x$	$-227.95592420641052271 \leq x \leq 230.25850929940456840$ Doppelgenauigkeit
$\tanh x$	
$\sinh^{-1}x$	$ x < 1 \times 10^{50}$
$\cosh^{-1}x$	$1 \leq x < 10^{50}$
$\tanh^{-1}x$	$ x < 1$
$\ln x$	$1 \times 10^{-99} \leq x < 1 \times 10^{100}$
$\log x$	
e^x	$-1 \times 10^{100} < x \leq 230.2585092$ Einfachgenauigkeit
	$-1 \times 10^{100} < x \leq 230.25850929940456840$ Doppelgenauigkeit
10^x	$-1 \times 10^{100} < x < 100$
$\sqrt[n]{x}$	$ x < 1 \times 10^{100}$
$\frac{1}{x}$	$ x < 1 \times 10^{100}; x \neq 0$

Funktion	Bereich von x
x^2	$ x < 1 \times 10^{50}$
$\sqrt{x}$	$0 \leq x < 1 \times 10^{100}$
$n!$	$0 \leq n \leq 69$ (n=ganzzahlig)
DMS→DEG DEG→DMS	$ x < 1 \times 10^{100}$
y^x ($y^x = 10^{x \cdot \log y}$)	wenn $y > 0$, $-1 \times 10^{100} < x \log y < 100$ wenn $y = 0$, $x > 0$ wenn $y < 0$, $x = \text{ganzzahlig}$ oder wenn $\frac{1}{x}$ ist ungeradzahlig ($x \neq 0$) und $-1 \times 10^{100} < 100 < x \log y < 100$
$\sqrt[y]{y}$ ($\sqrt[y]{y} = 10^{\frac{1}{x} \log y}$)	wenn $y > 0$, $-1 \times 10^{100} < \frac{1}{x} \log y < 100$; $x \neq 0$ wenn $y = 0$, $x > 0$ wenn $y < 0$, x oder $\frac{1}{x}$ muß ganzzahlig und nicht Null sein, und $-1 \times 10^{100} < \frac{1}{x} \log y < 100$
DECI→HEX	$ x \leq 9999999999$; $x = \text{ganzzahlig}$
HEX→DECI	$0 \leq x \leq 2540\text{BE3FF}$ (x in hexadezimal) FDABF41C01 $\leq x \leq \text{FFFFFFFF}$
$x, y = r, 0$	$(x^2 + y^2) < 1 \times 10^{100}$ $r = \sqrt{x^2 + y^2}$ $\frac{y}{x} < 1 \times 10^{100}$ $0 = \tan^{-1} \frac{y}{x}$
$r, 0 \rightarrow x, y$	$r < 1 \times 10^{100}$ $x = r \cos \theta$ $ r \sin \theta < 1 \times 10^{100}$ $y = r \sin \theta$ $ r \cos \theta < 1 \times 10^{100}$
nPr	$0 \leq r \leq n < 10^{100}$ n, r ganzzahlig
nCr	$0 \leq r \leq n < 10^{100}$ n, r ganzzahlig wenn $n - r < r$, $n - r \leq 69$ wenn $n - r \leq r$, $r \leq 69$

- Die folgenden Funktionen sind nur Funktionen mit einfacher Stellengenauigkeit. Vor der Verwendung doppeltgenaue Werte in einfachgenau umwandeln:
DECI, DEG, DMS, HEX, NCR, NPR, POL, REC

13. COMPUTER-ANWEISUNGEN

Die folgenden Seiten enthalten eine Auflistung aller BASIC-Kommandos, die Sie auf dem PC-1280 benutzen können.

Der Einfachheit halber wurde zu Ihrer Bequemlichkeit für diesen Nachschalteteil folgende Symbole hinzugefügt:

- [] Die eckigen Klammern sind Ihnen freigestellt. Sie sind selbst nicht Teil der Kommando-Eingabe.
- () Sie werden benutzt um geklammerte Werte in gewissen Kommandos einzuschließen. Sie sollten als Teil des Kommandos mit eingegeben werden.
- " " Sie werden zum einschließen von Parametern in gewissen Kommandos benutzt. Sie sollten als Teil des Kommandos mit eingegeben werden.

Bei der Eingabe irgendeines der Kommandos auf den folgenden Seiten ist es NICHT notwendig zwischen den Parametern eine Leerstelle zu lassen. Sie wurden nur der leichten Verständlichkeit wegen angegeben. Zum Beispiel: SAVE "d:Dateiname" kann abgekürzt und eingegeben werden als SAVE"d:Dateiname" oder SA."d:Dateiname".

Die einzelnen Modi, in denen die Befehle verwendet werden können, sind auf den Seiten gegenüber dem Befehlsnamen oben angegeben. Die folgende Terminologie findet Verwendung:

- Tape** Cassettenbandbefehl. Ein Datenrekorder muß über das CE-126P oder CE-124 Interface angeschlossen sein.
- Disk** RAM- oder Taschen-Diskettenbefehl. Das CE-140F Taschenlaufwerk muß angeschlossen oder die RAM-Karte im Schacht geladen sein.
- PRO** Kann im PRO-Modus direkt eingegeben werden.

Die Befehlszeile ist in der Regel mit einem Cursor (blinkernde vertikale Linie) versehen, der den aktuellen Eingabepunkt markiert.

RUN Kann im RUN-Modus direkt eingegeben werden.

Program Kann als Programmzeile eingegeben werden.

Die Befehlszeile ist in der Regel mit einem Cursor (blinkernde vertikale Linie) versehen, der den aktuellen Eingabepunkt markiert.

Die Befehlszeile ist in der Regel mit einem Cursor (blinkernde vertikale Linie) versehen, der den aktuellen Eingabepunkt markiert.

Die Befehlszeile ist in der Regel mit einem Cursor (blinkernde vertikale Linie) versehen, der den aktuellen Eingabepunkt markiert.

Die Befehlszeile ist in der Regel mit einem Cursor (blinkernde vertikale Linie) versehen, der den aktuellen Eingabepunkt markiert.

Die Befehlszeile ist in der Regel mit einem Cursor (blinkernde vertikale Linie) versehen, der den aktuellen Eingabepunkt markiert.

Die Befehlszeile ist in der Regel mit einem Cursor (blinkernde vertikale Linie) versehen, der den aktuellen Eingabepunkt markiert.

Die Befehlszeile ist in der Regel mit einem Cursor (blinkernde vertikale Linie) versehen, der den aktuellen Eingabepunkt markiert.

Die Befehlszeile ist in der Regel mit einem Cursor (blinkernde vertikale Linie) versehen, der den aktuellen Eingabepunkt markiert.

Die Befehlszeile ist in der Regel mit einem Cursor (blinkernde vertikale Linie) versehen, der den aktuellen Eingabepunkt markiert.

Programm**AREAD****FORMAT:** 1. AREAD variable name**Abkürzung:** AR.**Vergleiche:** INPUT, und die Beschreibung der **DEF**-Taste**WIRKUNG:**

Der AREAD-Befehl wird benutzt, um eine einzige Variable in ein Programm einzulesen, das mit der **DEF**-Taste gestartet wurde.

ANMERKUNGEN:

AREAD wird zur Eingabe eines einzelnen Startwert benutzt, ohne den INPUT Befehl zu setzen, so daß das Programm unter Verwendung eines Buchstabens als Etikett gestartet werden kann (siehe dazu die Beschreibung der **DEF**-Taste). AREAD muß in der ersten Zeile nach der Zeile mit dem Etikett erscheinen. Taucht AREAD an einer anderen Stelle im Programm auf, so wird der Befehl ignoriert. Es können sowohl numerische als auch Zeichenfolgen-Variablen benutzt werden, aber immer nur ein Wert pro Programm.

Bei Benutzung des AREAD-Befehls wird der gewünschte Wert im RUN-Modus eingegeben, die **DEF**-Taste gedrückt und der Buchstabe, der das Programm kennzeichnet. Wird eine Zeichenfolge-Variable benutzt, so muß diese nicht in Anführungszeichen gesetzt werden.

Die bezeichnete Variable wird beim Erscheinen von PROMPT (">") auf dem Bildschirm beim Start des Programms gelöscht.

Beachten Sie, wie im folgenden Beispiel unter Verwendung von PRINT am Beginn des Programms Text gespeichert wird.

```
10 "A":PRINT "ABC", "DEFG"
20 "S":AREAD A$:PRINT A$
```

RUN mode

DEF A → ABC

DEFG

DEF S → DEFG

Die zuletzt angezeigten Inhalte werden gespeichert, wenn das Display PRINT Numerischer Ausdruck, Numerischer Ausdruck, Numerischer Ausdruck... oder PRINT "Zeichenfolge", "Zeichenfolge"... anzeigt.

Der erste (ganz links) angezeigte Inhalt wird gespeichert, wenn das Display PRINT Numerischer Ausdruck; Numerischer Ausdruck; Numerischer Ausdruck... anzeigt.

Der zuletzt (ganz rechts) angezeigte Inhalt wird gespeichert, wenn das Display PRINT "Zeichenfolge"; "Zeichenfolge"; "Zeichenfolge"... anzeigt.

BEISPIELE:

10: "X":AREAD N

20: PRINT N^2

30: END

Die Eingabe von 7 DEF X läßt auf dem Display "49" erscheinen.

ARUN

Programm

FORMAT: 1. ARUN [Ausdruck]
2. ARUN ["Etikett"]

Abkürzung: ARU.

Vergleiche: AUTOGOTO, RUN

WIRKUNG:

Stellt den Computer so ein, daß ein Programm automatisch beim Einschalten und im bezeichneter RUN-Stellung gestartet wird.

ANMERKUNGEN:

Ist ARUN als erste Programmanweisung (in der ersten Zeile des Programms) enthalten, startet das Programm, sobald der Computer eingeschaltet ist. Denselben Effekt erzielt man, wenn über die Tastatur ein RUN-Kommando eingegeben wurde.

"Ausdruck" muß auf eine Zeilennummer bezogen sein. Zur Bezeichnung der Zeilennummer, auf welche das Programm springt, wenn der Computer eingeschaltet wird, muß deshalb eine Numerische Variable, eine Gleichung oder Variablen eingegeben werden.

"Etikett" muß die erste Anweisung einer Zeile innerhalb eines Programmes sein. Es muß aus Alphanumerischen Zeichen oder Symbolen bestehen.

ARUN ist AUTOGOTO ähnlich, mit Ausnahme, daß alle Variablen und Datenfelder vor der Programmausführung gelöscht werden.

Damit das ARUN-Kommando angenommen wird, muß der PC-1280 in der RUN-, PRO- oder RSV-Stellung ausgeschaltet werden.

BEISPIELE:

```

5: ARUN
10: CLS:WAIT 50
20: PRINT "WELCOME TO THE WORLD OF"
30: PRINT "THE COMPUTER"
40: PRINT "YOU HAVE"; MEM;" BYTES FREE"
50: END

```

Das Programm läuft automatisch bei POWER ON.

ASC

RUN PRO Programm

FORMAT: 1. ASC Zeichenfolge-Variable
2. ASC "Zeichenfolge"

Abkürzung: A.

Vergleiche: CHR\$

WIRKUNG:

Berechnet den ASCII-Codewert für das erste Zeichen in der vorgegebenen Zeichenfolge.

ANMERKUNGEN:

Die Zeichenfolge kann als Inhalt einer Zeichenfolge-Variablen in der Form X\$ oder als wirkliche Zeichenfolge in Anführungszeichen, wie z.B. "XXXX", angegeben werden. Nur der Wert des ersten Zeichens der Zeichenfolge wird berechnet. Die Zeichencode-Tabelle finden Sie im Anhang B.

BEISPIELE:

```
10: INPUT "ENTER A CHARACTER ";A$
20: N= ASC (A$)
30: PRINT "THE ASCII CODE IS ";N
40: GOTO 10
```

- [10] Der Benutzer drückt eine Taste, um irgendein Zeichen einzugeben.
- [20] ASC findet die Codenummer für dieses Zeichen.
- [30] Druckt die Antwort aus.
- [40] Wiederholt sich, bis der Benutzer das Programm durch betätigen der **BREAK**-Taste stoppt.

AUTO

PRO

FORMAT: 1. AUTO [Start der Zeilennummer][,Ergänzungswert]

Abkürzung:

Vergleiche:

WIRKUNG:

Ermöglicht in der PRO-Stellung die automatische Eingabe der Programm-Zeilenummer. Die Nummer der Startzeile und die Ergänzungswert-Variable werden definiert.

ANMERKUNGEN:

Falls sie nicht bestimmt wurde, setzt der Computer die Nummer der ersten Zeile automatisch auf 10 und den Ergänzungswert ebenfalls auf 10. Wurde jedoch vorher die Anweisung AUTO auf andere Werte eingestellt, so werden diese Werte verwendet.

ERROR 4 erscheint, wenn die Zeilennummer 65279 überschritten wird oder wenn eine Zeilennummer eingegeben wird, die bereits im Programm existiert.

Die AUTO-Stellung wird aufgehoben, (1) durch Ausschalten des Computers, (2) durch Druck auf die **CAL**-Taste, zum Übergehen in den Rechnungsbetrieb, (3) durch Herausgehen aus der PRO-Stellung, (4) durch die Anweisung NEW, (5) durch Druck auf die Tasten **C CE** und (6) durch Druck auf die **BREAK**-Taste.

BEISPIELE:

AUTO 10 Start an der Zeilennummer 10, gefolgt von 20, 30,...

AUTO 200,20 Start an der Zeilennummer 200, gefolgt von 220, 240,...

AUTOGOTO

Programm

FORMAT: 1. AUTOGOTO Ausdruck
2. AUTOGOTO "Etikett"

Abkürzung: AU.

Vergleiche: ARUN, GOTO

WIRKUNG:

Stellt den Computer so ein, daß der Programmablauf automatisch beim Einschalten erfolgt.

ANMERKUNGEN:

Wenn AUTOGOTO in der ersten Zeile eines Programmes (die erste Anweisung) enthalten ist, startet das Programm beim Einschalten des Computers. Er arbeitet so, als ob eine GOTO-Anweisung über die Tastatur eingegeben wurde. AUTOGOTO ähnelt ARUN, löscht aber nicht alle Variablen und Datenfelder bevor das Programm anläuft.

"Ausdruck" muß einer Zeilennummer entsprechen. Zur Bezeichnung der Zeilennummer, auf welche das Programm springt, wenn der Computer eingeschaltet wird, muß deshalb eine Numerische Variable, eine Gleichung oder Variablen eingegeben werden.

"Etikett" muß die erste Anweisung einer Zeile innerhalb eines Programmes sein. Es muß aus Alphanumerischen Zeichen oder Symbolen bestehen.

BEISPIELE:

10: AUTOGOTO 60

:
:
:
:

Bei POWER ON beginnt die Ausführung des Programms ab Zeile 60.

BEEP

RUN PRO Programm

FORMAT: 1. BEEP Nummer
2. BEEP Nummer [,Ton[,Dauer]]

Abkürzung: B.

Vergleiche:

WIRKUNG:

Erzeugt Töne der vorgegebenen Höhe und Dauer über den internen Lautsprecher des Computers.

ANMERKUNGEN:

Die eingebene Zahl ist die Anzahl der ertönenden Signaltöne. Einen positiven Wert oder Ausdruck bis zu 9.999999999E99 spezifizieren.

Die Wahl der Tonhöhe spezifiziert die steigende Tonhöhe (Frequenz) des Pieptons im Bereich von 255 bis 0. 255 entspricht einer Frequenz von ungefähr 230 Hz, während 0 einer Frequenz von ungefähr 2,8 kHz entspricht.

Die Wahl der Tondauer spezifiziert die Dauer des Pieptons im Bereich von 0 bis 65535. Die Einstellung der Piepdauer ändert sich mit dem Tonparameter. Der gleiche Wert der Dauer erscheint in niedrigen Frequenzen etwas länger.

Zum Stoppen des Pieptons drücken Sie die **BREAK**-Taste.

Die folgende Tabelle gibt Ihnen Einzelheiten der Parameter-Kombinationen an:

Ton-Art	Nr.	Ideale Frequenz	Reale Frequenz	Toleranz (%)	Ton	Ton-art	Nr.	Ideale Frequenz	Reale Frequenz	Toleranz (%)	Ton
do	3	261,6	261,8	0,08	222	do	27	1046,5	1040,7	-0,56	39
do#	4	277,2	277,7	0,18	208	do#	28	1108,7	1113,0	0,39	35
re	5	293,7	294,3	0,20	195	re	29	1174,7	1174,3	-0,03	32
re#	6	311,1	311,4	0,10	183	re#	30	1244,5	1242,7	-0,14	29
mi	7	329,6	329,0	-0,18	172	mi	31	1318,5	1319,6	0,08	26
fa	8	349,2	348,8	-0,11	161	fa	32	1396,9	1406,6	0,69	23
fa#	9	370,0	371,0	0,27	150	fa#	33	1480,0	1471,2	-0,60	21
so	10	392,0	391,4	-0,20	141	so	34	1568,0	1580,2	0,77	18
so#	11	415,3	414,2	-0,27	132	so#	35	1661,2	1662,3	0,07	16
la	12	440,0	439,9	-0,02	123	la	36	1760,0	1753,4	-0,38	14
la#	13	466,2	465,5	-0,15	115	la#	37	1864,7	1855,1	-0,52	12
ti	14	493,9	494,2	0,06	107	ti	38	1975,5	1969,2	-0,32	10
do	15	523,3	522,4	-0,17	100	do	39	2093,0	2098,4	0,26	8
do#	16	554,4	554,1	-0,05	93						
re	17	587,3	589,9	0,44	86						
re#	18	622,3	624,4	0,34	80						
mi	19	659,3	656,4	-0,44	75						
fa	20	698,5	699,4	0,13	69						
fa#	21	740,0	739,9	-0,01	64						
so	22	784,0	785,3	0,17	59						
so#	23	830,6	825,8	-0,58	55						
la	24	880,0	882,8	0,32	50						
la#	25	932,3	934,3	0,21	46						
ti	26	987,8	992,2	0,44	42						

MIN. 230,6 Hz (255)

$$T = (90 + 4n) \times 3,9 \text{ (}\mu\text{s)}$$

MAX. 2844,4 Hz (0)

$$f = \frac{1}{T} = \frac{1}{(90 + 4n) \times 3,9} \text{ (MHz)}$$

BEISPIELE:

```
10: FOR I=1 TO 3
20: FOR J=5 TO 25 STEP 5
30: BEEP I,J,150
40: NEXT J
50: NEXT I
60: END
```

- [10] Diese äußere Schleife wird benutzt, um die Anzahl der Pieptöne von 1 auf 3 zu ändern.
 [20] Der Zähler der inneren Schleife wird benutzt, um die Tonhöhe zu verändern.
 [30] Die BEEP-Anweisung wird 5 mal ausgeführt.

CHAIN

Band Disketten Programm

- FORMAT:**
1. CHAIN
 2. CHAIN Ausdruck
 3. CHAIN "[d:]Dateiname"
 4. CHAIN "[d:]Dateiname",Ausdruck

Abkürzung: CHA.

Vergleiche: CLOAD, CSAVE, LOAD, SAVE, RUN

WIRKUNG:

CHAIN lädt und startet aus einem Programm heraus ein andere Programm, das auf Band, RAM- oder Taschen-Diskette gespeichert wurde.

ANMERKUNGEN:

Formate 1 und 2 benötigen keinen Peripherienamen, sie dienen nur zur Band-speicherung. Für Formate 3 und 4 folgende Peripherienamen eingeben:

F: RAM Diskette

X: Diskettenlaufwerk-Fach

Auf dem entsprechenden Speichermedium müssen bei Verwendung von CHAIN ein oder mehrere Programme vorhanden sein. Das gerade laufende Programm wird an dem Punkt gelöscht, an dem ein CHAIN-Befehl eingegeben wird und das bezeichnete Programm bereitgestellt und gestartet wird.

Alle Variablen (außer den fixierten) werden gelöscht, wenn das Programm als Resultat eines mit der CHAIN-Anweisung geladenen Programm zu groß für die Programm-Speicherzone wird.

Format 1 lädt das erste gespeicherte Programm vom Band und startet es von der niedrigsten Zeilennummer im Programm. Der Effekt ist der gleiche nach Eingabe CLOAD (oder LOAD) und RUN in der RUN-Stellung.

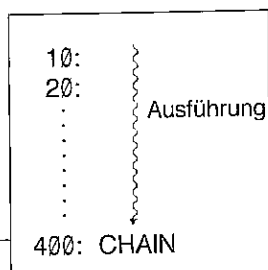
Format 2 lädt das erste gespeicherte Programm vom Band und startet es an der mit "Ausdruck" bezeichneten Zeile.

Format 3 sucht das mit einem "Dateinamen" bezeichnete Programm auf der Diskette, lädt es und beginnt bei der niedrigsten Zeilennummer.

Format 4 sucht das mit einem "Dateinamen" bezeichnete Programm, lädt es und beginnt mit der durch "Ausdruck" bezeichneten Zeilennummer.

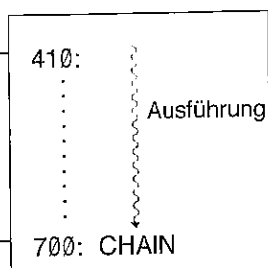
Nehmen wir z.B. an, daß Sie drei Programmsegmente mit den Namen PRO-1, PRO-2 und PRO-3 haben. Jedes Segment endet mit einem CHAIN-Befehl.

"PRO-1"



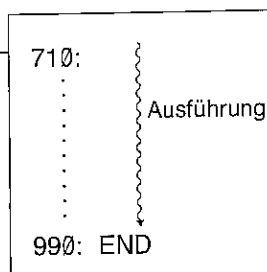
400: CHAIN "PRO-2", 410

"PRO-2"

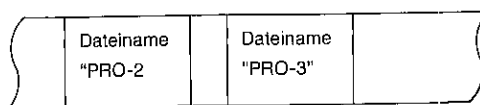
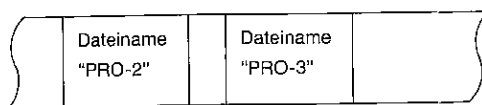
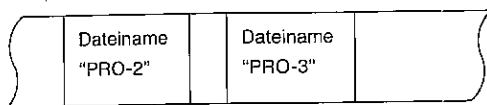


700: CHAIN "PRO-3", 710

"PRO-3"



Magnetband
(↑ markiert die Position des
Kassetten-Lesekopfes.)



Bei laufendem Programm wird der nächste Abschnitt in den Speicher abgerufen und startet, wenn das CHAIN-Kommando eingegeben wird. In dieser Weise werden in etwa alle weiteren Abschnitte verarbeitet.

Hinweis:

Wird zum Laden eines Programmes, das eine CHAIN-Anweisung enthält, der MERGE-Befehl verwendet, überprüfen Sie, daß das CHAIN-Kommando mit Absicht durchgeführt wird.

Die .BAS-Erweiterung wird angenommen, wenn sie im Dateinamen ausgelassen wurde.

Dateien, die mit dem CHAIN-Kommando eröffnet wurden, werden nicht automatisch geschlossen.

Mit CHAIN geladene Dateien werden automatisch Passwort-geschützt, wenn die Dateien im eingebauten Speicher des PC-1280 Passwort-geschützt wurden.

Zeichenfolge-Variablen können nicht als Dateinamen benutzt werden.

Die Fehlermeldung ERROR 8 erscheint, wenn (1) bei nicht in den PC-1280 eingelegerter RAM-Karte das Gerät F: angegeben wurde, wenn (2) die angegebene Datei nicht existiert und (3) wenn die angegebene Datei bereits offen ist.

BEISPIELE:

10: CHAIN

20: CHAIN "PRO-2", 480

[10] Lädt das erste Programm (vom Band) und beginnt die Ausführung mit der niedrigsten Zeilennummer.

[20] Sucht (auf dem Band) das Programm PRO-2, lädt und startet es mit der Zeile 480.

CHR\$

RUN PRO Programm

FORMAT: 1. CHR\$ numerische Ausdruck

Abkürzung: CH.

Vergleiche: ASC

WIRKUNG:

Springt auf das Zeichen zurück, das dem numerischen Zeichencode seines Arguments entspricht.

ANMERKUNGEN:

Sie finden die Tabell der Zeichencodes und ihres Verhältnisses zu Zeichen in Anhang B. CHR\$ 65 ist gleich "A".

Ein Wert größer als 256 bewirkt einen Fehler.

BEISPIELE:

```
10: WAIT 0:AA$=""  
20: INPUT "CODE=" ;A:CLS  
30: AA$=AA$+CHR$A  
40: CURSOR 7,1: PRINT AA$  
50: GOTO 20
```

Zeigt die Zeichen an, die durch die Codeeingabe in Zeile 20 representiert sind.

CLEAR

RUN PRO Programm

FORMAT: 1. CLEAR
2. CLEAR Variable 1, Variable 2,..., Variable n

Abkürzung: CL.

Vergleiche: DIM, ERASE

WIRKUNG:

Mit diesem Befehl werden alle in einem Programm benutzten Variablen gelöscht und alle gesetzten Variablen auf Null bzw. leer gesetzt.

ANMERKUNGEN:

Der CLEAR-Befehl setzt Speicherraum frei, der zum Speichern von einzelnen oder doppelten Präzisionsvariablen gebraucht wurde. Er ordnet die gesicherten Variablen unter Verwendung der DIM-Anweisung. Verwenden Sie CLEAR, wenn die im ersten Teil eines Programms verwendeten Variablen nicht gebraucht werden und der zur Verfügung stehende Platz begrenzt ist. CLEAR kann ebenfalls am Beginn eines Programmes verwendet werden. Damit wird von vorher zur Ausführung eines Programms eingegebenen Variablen beanspruchter Platz geschaffen, wenn mehrere Programme gespeichert sind.

CLEAR läuft erneut die von doppelten Präzisions-Variablen besetzte Speicherbereiche an und löscht die doppelte Präzisionsstellung.

CLEAR setzt die Speicherbereiche, die durch die Variablen A-Z, A\$ bis Z\$ oder A(1) bis A(26) (ohne DIM Deklaration) benutzt wurden, nicht frei, wenn diese fest vorgegeben wurden (siehe dazu Kapitel 4). Format 1 löscht alle Variablen und setzt die permanenten Variablen auf Null oder leer und läßt Zwei-Zeichen-Variablen undefiniert. Format 2 erlaubt daß spezifische Variablen gelöscht werden. Im Format 2 können keine Feldnamen angegeben werden. Zum Löschen bestimmter Felder siehe ERASE-Kommando.

BEISPIELE:

```
10: A=5: DIM C(5)
20: CLEAR
```

[20] Setzt den C. zugewiesenen Bereich frei und setzt A gleich Null.

Wenn doppelte Präzisions-Variablen, die unter Verwendung des DEFDBL-Kommandos bestimmt wurden, im CLEAR-Befehl enthalten sind, werden die Inhalte der doppelten Präzisions-Variablen gelöscht, die Variable selbst aber bleibt erhalten.

BEISPIELE:

```
10: DEFDBL A
20: AB=12345678912345678:AB!=123456
30: PRINT AB, AB!
40: CLEAR AB
50: PRINT AB, AB!
60: END
```

Das Display in Zeile 30:

```
1 2 3 4 5 6 7 8 9 1 2 3 4 5 6 7 8.
1 2 3 4 5 6.
```

Das Display in Zeile 50:

```
0.
1 2 3 4 5 6.
```

CLOAD

Band RUN PRO

FORMAT: 1. CLOAD
2. CLOAD "Dateiname"

Abkürzung: CLO.

Vergleiche: CLOAD?, CSAVE, MERGE, PASS

WIRKUNG:

Lädt ein auf Band gesichertes Programm in den Rechner.

ANMERKUNGEN:

Die erste Form des CLOAD-Kommandos löscht die existierenden Programme aus dem Speicher und läßt das erste Programm auf dem Band, angefangen bei der aktuellen Position.

Die zweite Form des CLOAD-Kommandos löscht den Speicher, sucht das Band nach dem Programm ab, dessen Name mit "Dateiname" angegeben ist und ladet das Programm.

Ist der PC-1280 in der PROgramm- oder RUN-Stellung, so wird das Programm vom Band in den Programmspeicher geladen. Ist der PC-1280 in der Reservestellung, werden die Daten in den Reservespeicher geladen. Sie müssen darauf achten, daß Sie keine Programme in den Reservespeicher und keine Reservedaten in den Programmspeicher laden.

Der Computer kann nicht unterscheiden zwischen einem Programm und einem Reserveprogramm. Wenn ein Programm oder ein Reserveprogramm in den falschen Speicher des Computers geladen wird, kann es sein, daß das System nicht mehr funktionsfähig ist. In diesem Fall drücken Sie die ALL RESET-Taste an der Rückseite des Computers, um den Zustand zu löschen.

Unterbrechen Sie die Ausführung des Kommandos mit Hilfe der **BREAK**-Taste, falls die Suche erfolglos war.

Treten während der Ausführung Fehler auf, wird das geladene Programm nicht zugelassen.

Lesen Sie den Bandbetrieb betreffend den Abschnitt über "periphere Geräte".

BEISPIELE:

CLOAD*

CLOAD"PRO3"***

- * Lädt das erste Programm vom Band.
- ** Sucht auf dem Band den Dateinamen "PRO3" und lädt es ein.

CLOAD?

Band RUN PRO

FORMAT: 1. CLOAD?
2. CLOAD?"Dateiname"

Abkürzung: CLO.?

Vergleiche: CLOAD, CSAVE, MERGE, PASS

WIRKUNG:

Mit dem Kommando CLOAD? wird ein auf Band gespeichertes Programm mit einem im Speicher enthaltenen Programm verglichen.

ANMERKUNGEN:

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

Die erste Form des CLOAD?-Kommandos vergleicht das im Speicher enthaltene Programm mit dem ersten Programm auf dem Band, angefangen bei der aktuellen Position.

Die zweite Form des CLOAD?-Kommandos sucht nach dem mit "Dateinamen" bezeichneten Programm und vergleicht es dann mit dem Speicherinhalt.

Lesen Sie den Bandbetrieb betreffend den Abschnitt über "periphere Geräte".

BEISPIELE:

CLOAD?*

CLOAD?"PRO3"***

* Vergleicht den Speicherinhalt mit dem ersten auf dem band gefundenen Programm.

** Sucht auf dem Band den Dateinamen "PRO3" und vergleicht ihn dann mit dem Speicherinhalt.

CLOSE**Disketten RUN PRO Programm****FORMAT:** 1. CLOSE [#Dateinummer, #Dateinummer ...]**Abkürzung:** CLOS.**Vergleiche:** OPEN**WIRKUNG:**

Schließt eine oder mehrere Dateien in dem gerade befindlichen Zusatzgerät ab.

ANMERKUNGEN:

Das CLOSE-Kommando schließt eine oder mehrere Dateien mit der (den) bestimmten Dateinummer(n) ab. Die Nummern müssen #2 bis #7 (für Taschen-Diskette) oder #20 bis #25 (für RAM-diskette) sein. Ist keine Dateinummer bestimmt, werden alle Dateien abgeschlossen. Die Dateinummer ist dann zum Gebrauch für andere Dateien freigegeben.

Alle Dateien werden in den folgenden Fällen abgeschlossen (CLOSE):

- Nach Durchführung eines END-, NEW- oder RUN-Kommandos.
- Das (binäre) Programm wurde mit dem SAVE-Kommando auf die Ram- oder Taschen-Diskette gesichert oder mit dem LOAD- oder dem MERGE-Kommando in den Speicher eingelesen.
- Der Strom wurde abgeschaltet (Automatisches Stromabschalten).
- Der PC-1280 wurde in der CAL-Stellung geschaltet.

BEISPIELE:

CLOSE #2, #5

Schließt die Dateien #2 und #5.

CLS

Programm

FORMAT: 1. CLS**Abkürzung:****Vergleiche:** CURSOR**WIRKUNG:**

Das CLS-Kommando löscht das Display.

ANMERKUNGEN:

Das CLS-Kommando löscht das Display und versetzt es in die Startposition 0.

BEISPIELE:

```
10: WAIT 3
20: INPUT A$
30: FOR B=0 TO 41
40: CLS
50: CURSOR B
60: PRINT A$
70: NEXT B
80: CLS
90: END
```

Dieses Programm zeigt die Eingabe an und bewegt sie von der linken zur rechten Seite (von der oberen Zeile zur unteren Zeile). In jedem Durchgang der FOR-NEXT-Schleife der Zeilen 30-70 wird die Anzeige mit dem CLS-Befehl gelöscht, die Startposition des Displays mit dem CURSOR-Befehl verschoben und der Inhalt von A\$ mit Hilfe des PRINT-Befehls angezeigt. Wenn Sie auf diese Weise das Display beschreiben und löschen, kann bewirkt werden, daß der Bildschirminhalt rollt. (Löschen Sie Zeile 40 und führen Sie das Programm aus. Beachten Sie den Unterschied.)

RUN

CONT**FORMAT:** 1. CONT**Abkürzung:** C.**Vergleiche:** RUN, STOP**WIRKUNG:**

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

ANMERKUNGEN:

Wenn die Ausführung eines Programms mit dem Befehl STOP unterbrochen wurde, so kann es mit der Anweisung CONT fortgesetzt werden.

Wurde ein Programm mit der **BREAK**-Taste unterbrochen, so kann es nach Eingabe des CONT-Befehls am Aufforderungszeichen fortgesetzt werden.

CONT kann auch benutzt werden, um ein Programm fortzusetzen, das mit PRINT oder GPRINT unterbrochen wurde.

BEISPIELE:

CONT

Setzt die Ausführung eines unterbrochenen Programms fort.

COPY

Disketten RUN PRO Programm

FORMAT: 1. COPY "d: Dateiname 1" TO "d: Dateiname 2"**Abkürzung:** COP.**Vergleiche:** SET**WIRKUNG:**

Kopiert die Inhalte einer Ram- oder Taschen-Diskette auf eine andere Diskette oder bei Taschen-Disketten auf eine andere Zone der gleichen Diskette.

ANMERKUNGEN:

Bestimme die periphere Einheit als RAM-Diskette (F:) oder Taschen-Diskette (X: oder Y:).

Bestimme X: und Y:, um die Inhalte einer Diskette auf eine andere zu kopieren, auch wenn nur ein Laufwerk vorhanden ist. Setzen Sie die Diskette nach Erscheinen der Hinweise ("INSERT DISC INTO X:" oder "INSERT DISC INTO Y:") wieder ein und drücken dann **ENTER**. Fahren Sie mit diesem Prozess fort, bis das Anforderungszeichen (>) erscheint.

COPY "X:*.*" to "Y:*.*" kopiert alle Dateien einer Taschen-Diskette auf eine andere Taschen-Diskette.

Wildcards (* oder ?) können nicht verwendet werden, wenn von einer RAM-Karte auf eine Taschen-Diskette oder umgekehrt kopiert werden soll.

Eine Fehlermeldung ERROR erscheint wenn der Dateiname 2 bereits existiert und die Datei Passwort-geschützt ist oder wenn die RAM-Karte Passwort-geschützt ist.

Eine Fehlermeldung ERROR erscheint, wenn das Empfänger-Gerät nicht aufgestellt und angeschlossen wurde.

Der Dateiname 1 muß definiert sein.

Gibt es bereits einen Dateinamen 2, wird dieser gelöscht.

Die Erweiterung muß definiert werden, wenn der Dateiname die Ziffer 1 enthält.

BEISPIELE:

COPY "F:RICH.BAS" TO "X:RICH.BAS"

Kopiert die Datei RICH.BAS auf einer RAM-Diskette auf RICH.BAS auf einer Taschen-Diskette.



CSAVE

Band RUN PRO Programm

- FORMAT:**
1. CSAVE
 2. CSAVE "Dateiname"
 3. CSAVE, "Passwort"
 4. CSAVE "Dateiname", "Passwort"

Abkürzung: CS.

Vergleiche: CLOAD, CLOAD?, MERGE, PASS

WIRKUNG:

Mit dem Kommando CSAVE wird ein Programm auf dem Band gesichert.

ANMERKUNGEN:

Die erste Form des Kommandos CSAVE speichert alle Programme des Rechners ohne spezielle Dateinamen auf dem Band.

Die zweite Form des Kommandos CSAVE speichert alle Programme im Speicher des Rechners auf dem Band und weist den entsprechenden Dateinamen zu.

Die dritte Form des Kommandos CSAVE schreibt alle Programme im Speicher des Rechners ohne speziellen Dateinamen auf das Band ab und weist das angegebene Passwort zu. Programme, die mit einem Passwort geschützt sind, können von jedermann geladen, aber nur von jemandem gelistet oder verändert werden, der das korrekte Passwort kennt. (Siehe PASS-Kommando.)

Die vierte Form des Kommandos CSAVE speichert alle Programme im Speicher des Rechners auf Band und weist sowohl den angegebenen Dateinamen als auch das angegebene Passwort zu.

Ist der PC-1280 in PROgramm- oder RUN-Stellung, wird der Programmspeicher auf Band gesichert. In der Reserve-Stellung wird der Reservespeicher gesichert.

Lesen Sie den Bandbetrieb betreffend den Abschnitt über "periphere Geräte".

BEISPIELE:

CSAVE "PRO3", "GEHEIM"

Speichert die jetzt gesicherten Programme unter dem Namen "PRO3" mit dem Passwort "GEHEIM" auf Band ab.



CURSOR

Programm

- FORMAT:**
1. CURSOR Ausdruck 1, Ausdruck 2
 2. CURSOR Ausdruck
 3. CURSOR

Abkürzung: CU.

Vergleiche: CLS, INPUT, PRINT, PAUSE, GPRINT

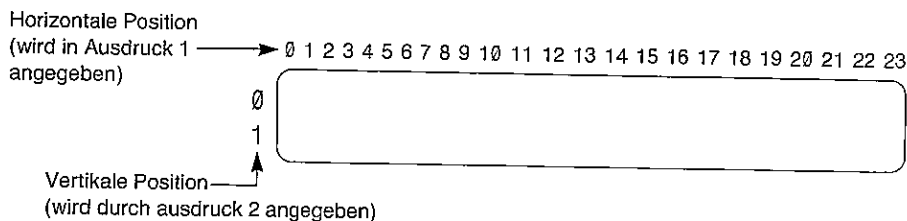
WIRKUNG:

Dieser Befehl markiert den Startpunkt des Displays in Spalteneinheiten.

ANMERKUNGEN:

Format 1 und Format 2 geben den Startpunkt des Displays in Einheiten einer Zeichenposition für die durch das PRINT-, PAUSE-Kommando etc. angezeigten Inhalte.

Die Displayposition wird folgendermaßen mit Format 1 spezifiziert.



Jede Position auf dem Display ist durch einen horizontalen Wert und einen vertikalen Wert bestimmt. Die Werte von Ausdruck 1 und Ausdruck 2 geben die horizontale bzw. die vertikale Position an. Der Bereich von Ausdruck 1 ist 0 bis 23 und der Bereich von Ausdruck 2 ist 0 bis 1.

Das Format 3 löscht den Startpunkt des Displays.

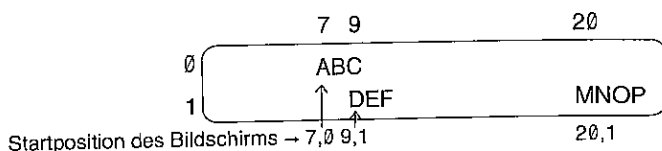
Mit dem CURSOR-Kommando können Sie den Text an einer beliebigen Stelle auf den Bildschirm schreiben, ohne den bestehenden Bildschirmtext zu beeinflussen, außer da, wo Zeichen direkt überschrieben werden. Benutzen Sie das CLS-Kommando, um den gesamten Bildschirm zu löschen.

Falls die angezeigten Zeichen die Bildschirmkapazität überschreiten, wird der Bildschirm nach oben gerollt, um alle Zeichen wiedergeben zu können, auch wenn die Startposition mit dem CURSOR-Kommando spezifiziert wurde.

BEISPIELE:

```
5: CLS
10: CURSOR 7,0: PRINT "ABC"
20: CURSOR 9,1: PRINT "DEF"
30: CURSOR 20,1: PRINT "MNOP"
```

Bei Ausführung des Programms erhalten Sie folgendes Bild:



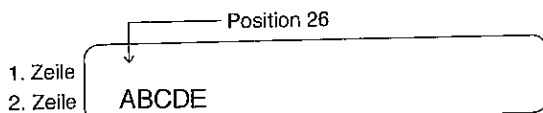
Die Startposition des Bildschirms wird folgendermaßen mit dem Format 2 spezifiziert.

1. Zeile	0 1 2 3 22 23
2. Zeile	24 25 26 46 47

Den Positionen auf dem Bildschirm werden Zahlen von 0 bis 47 zugewiesen, angefangen in der Ecke oben links des Bildschirms bis zur Ecke unten rechts. Der Wert des Ausdrucks in Format 2 bezeichnet die Zahl der Position für die Startposition des Bildschirms. Deshalb müssen Sie sichergehen, daß der Wert des Ausdrucks in Format 2 innerhalb des Bereiches von 0 bis 47 liegt. Wenn dieser Bereich überschritten wird, erhalten Sie eine Fehlermeldung (ERROR 3).

BEISPIELE:

```
5: CLS
10: CURSOR 26: PRINT "ABCDE"
```



CURSOR kann auch benutzt werden, um den Cursor auf dem Bildschirm zu positionieren und um ein INPUT-Kommando einzugeben. Nachdem das INPUT-Kommando ausgeführt ist, bleibt aber der Cursor unverändert an der durch das CURSOR-Kommando angegebenen Stelle erhalten, so daß ein PRINT-Kommando den INPUT-Text überschreibt.

BEISPIELE:

```
10: CLS
20: CURSOR 0,1
30: INPUT "DATA = ";A
40: PRINT A/2
```

[30 UND 40] Die in Zeile 20 definierte Position ist sowohl für Zeile 30 als auch für Zeile 40 gültig.

```
10: CLS
20: CURSOR 0,1
30: INPUT "DATA = ";A
35: CURSOR
40: PRINT A/2
```

[35] Die in 20 definierte Position wird gelöscht.

DATA

Programm

FORMAT: 1. DATA Liste der Konstanten

Abkürzung: DA.

Vergleiche: READ, RESTORE

WIRKUNG:

DATA wird benutzt, um Daten für READ bereitzustellen.

ANMERKUNGEN:

Soll ein Feld mit Anfangswerten geladen werden, ist es empfehlenswert, die Werte in einem DATA-Befehl aufzulisten und sie dann mit einem READ-Kommando in einer FOR...NEXT-Schleife in das Feld einzulesen. Wird der erste READ-Befehl ausgeführt, wird der erste Wert im DATA-Kommando abgerufen. Mit jedem der darauf folgenden READ-Befehle werden die nachfolgenden Werte in der Reihenfolge gelesen, in der sie im Programm erscheinen, ohne Rücksicht zu nehmen auf die Anzahl der in jedem DATA-Befehl definierten Werte oder auf die Anzahl der benutzten DATA-Anweisungen.

Ein DATA-Befehl kann aus beliebigen numerischen oder Zeichenfolge Konstanten, getrennt durch Konstanten, zusammengesetzt sein. Die Zeichenfolge-Konstanten müssen in Anführungszeichen gesetzt werden.

DATA-Anweisungen haben keinen Einfluß, wenn sie im Laufe der normalen Ausführung des Programmes angetroffen werden, so daß sie an jeder Stelle des Programms eingefügt werden können. Viele Programmierer setzen sie gerne hinter den dazugehörigen READ-Befehl. Wenn nötig können die Werte der DATA-Anweisung mit der RESTORE-Anweisung ein zweites Mal gelesen werden.

BEISPIELE:

```

10: DIM B(10)
20: WAIT 128
30: FOR I=1 TO 10
40: READ B(I)
50: PRINT B(I)
60: NEXT I
70: DATA 10,20,30,40,50,60
80: DATA 70,80,90,100
90: END

```

[10] Initialisierung eines Feldes.

[40] Lädt die Werte aus der DATA-Anweisung so, daß B(1) wird 10, B(2) wird 20, B(3) wird 30 usw.

RUN PRO Programm

DEFDBL

FORMAT: 1. DEFDBL Zeichenbereich
2. DEFDBL

Abkürzung: DEF.

Vergleiche: DEFSNG

WIRKUNG:

Format 1 definiert eine oder mehrere Variable, wie ehemalige genaue und jetzt doppelte genaue haben und Format 2 bestimmt "doppelte Genauigkeit" Stellung Berechnungen.

ANMERKUNGEN:

In Format 1 werden die Variablen im "Zeichenbereich" als doppelt genaue bestimmt. Der "Zeichenbereich" kann wie folgt spezifiziert werden:

- DEFDBL C—F
wobei die Variablen C, D, E und F als doppelt genaue bestimmt sind oder
- DEFDBL E,F,Z,H—J
wobei die Variablen E, F, Z, H, I und J als doppelt genaue bestimmt sind.

Variable Namen, die Deklarations-Zeichen (!) mit einfacher Genauigkeit vorausgehen, haben Typenpriorität über die durch die DEFDBL-Anweisung deklarierten Variablennamen. Zum Beispiel E und F in der Anweisung

DEFDBL E,F,Z

werden als doppelt genaue Variablen behandelt, aber E! und F! werden als Variablen mit einfacher Genauigkeit behandelt.

Variablen, die weder als doppelt noch als einfach genau deklariert sind, werden als Variable mit einfacher Genauigkeit behandelt.

In Format 2 werden alle nachfolgenden Berechnungen in doppelter Genauigkeit ausgeführt. In dieser Stellung erscheint auf dem Bildschirm der Hinweis DBL. Die Stellung der doppelten Genauigkeit wird durch die folgende Kommandos gelöscht.

- DEFSNG-Anweisung
- Ausschalten des PC-1280
- RUN-Kommandos

Bei Verwendung des DIM-Befehls zum Festsetzen der Anzahl der Datenelemente, die in einem numerischen Feld erlaubt sind, muß die DEFDBL-Anweisung zuerst benutzt werden, wenn solche Datenelemente als Variablen mit doppelter Genauigkeit behandelt werden sollen:

```
DEFDBL A
DIM A(3,2)
```

Im folgenden Beispiel werden die Variablen als Variablen mit einfacher Genauigkeit behandelt:

```
DIM A(3,2)
DEFDBL A
```

RUN PRO Programm

DEFSNG

FORMAT: 1. DEFSNG Zeichenbereich
2. DEFSNG

Abkürzung: DEFS.

Vergleiche: DEFDBL

WIRKUNG:

Format 1 definiert eine oder mehrere Variable, wie ehemalige genaue und jetzt doppelte genaue haben und Format 2 bewirkt, daß die Ausführung auf die Zeile mit dem gegebenen Etikett springt und dort startet.

ANMERKUNGEN:

In Format 1 werden die Variablen im "Zeichenbereich" als ehemalige genaue bestimmt. Der "Zeichenbereich" kann wie folgt spezifiziert werden:

- DEFSNG C—F
wobei die Variablen C, D, E und F als einfach genaue bestimmt sind oder
- DEFSNG E,F,Z,H—J
wobei die Variablen E, F, Z, H, I und J als einfach genaue bestimmt sind.

Variable Namen, die Deklarations-Zeichen (#) mit doppelter Genauigkeit vorgehen, haben Typenpriorität über die durch die DEFSNG-Anweisung deklarierten Variablennamen. Zum Beispiel E und F in der Anweisung

DEFSNG E,F,Z

werden als einfach-genau Variablen behandelt, aber E# und F# werden als Variablen mit doppelter Genauigkeit behandelt.

Variablen, die weder als doppelt noch als einfach genau deklariert sind, werden als Variable mit einfacher Genauigkeit behandelt.

In Format 2 werden alle nachfolgenden Berechnungen in einfacher Genauigkeit ausgeführt. Der Hinweis DBL auf dem Bildschirm des PC-1280 wird in dieser Stellung gelöscht.

DEGREE

RUN PRO Programm

FORMAT: 1. DEGREE

Abkürzung: DE.

Vergleiche: GRAD, RADIAN

WIRKUNG:

DEGREE wird benutzt, um Winkelwerte auf Alt-Grad umzuschalten.

ANMERKUNGEN:

Der PC-1280 kann drei Formen der Winkeldarstellungen verarbeiten: Alt-Grad (dezimal), Radialwerte und Neu-Grad. Diese Formen werden für die Darstellung der Werte und Argumente von den SINus-, COSinus- und TANGens- Funktionen und der Ergebnisse der Umkehrfunktionen Arcussinus (ASN), Arcuscosinus (ACS) und Arcustangens (ATN) benutzt.

Die DEGREE-Funktion schaltet die Form für alle Winkelwerte auf Alt-Grad um, bis Sie GRAD und RADIAN einsetzen. Die DMS- und DEG-Funktionen werden benutzt, um Winkelwerte von Alt-Grad in Grade, Minuten, Sekunden und umgekehrt umzuformen.

BEISPIELE:

```
10: DEGREE
20: X= ASN 1
30: PRINT X
```

[20] X hat jetzt den Wert 90, d.h. 90 Grad, dem Arcussinus von 1.

PRO

DELETE

FORMAT:

1. DELETE Zeile #
2. DELETE Zeile #,
3. DELETE Zeile #, Zeile #
4. DELETE ,Zeile #

Abkürzung: DEL.

Vergleiche: NEW

WIRKUNG:

Löscht die angegebenen Programmzeilen im Speicher.

ANMERKUNGEN:

Format 1 löscht nur die angegebene Programmzeile. Format 2 löscht die Programmzeilen von der angegebenen Zeilennummer bis zur höchsten Programmzeile im Speicher. Format 3 löscht alle Programmzeilen zwischen der ersten angegebenen Zeilennummer (niedrigster Wert) und der zweiten angegebenen Zeilennummer (höherer Wert). Format 4 löscht die Programmzeilen von der niedrigsten Zeilennummer im Speicher bis zur angegebenen Zeilennummer.

Wenn Sie DELETE im RUN-Modus benutzen, erhalten Sie die Fehlermeldung ERROR 9. Falls Sie ein Passwort eingesetzt haben, passiert nichts und das Aufforderungszeichen erscheint. Wenn der Speicher mehr als ein Programm enthält, weil das MERGE-Kommando eingesetzt wird, arbeitet DELETE auf das zuletzt gemischte Programm. Als Zeilennummern sind nur die Ziffern 0 bis 9 gültig. Wenn Sie eine Zeile angeben, die nicht existiert, erhalten Sie eine Fehlermeldung. Wenn Sie eine Startzeilennummer angeben, die größer ist als die Endzeilennummer, erhalten Sie ebenfalls eine Fehlermeldung.

Um das ganze Programm zu löschen, benutzen Sie das NEW-Kommando.

BEISPIELE:

DELETE 150*

DELETE 200,**

DELETE 50,150***

DELETE ,35****

* Löscht nur die Zeile 150.

** Löscht alles von Zeile 200 bis zur höchsten Zeilennummer.

*** Löscht alle Zeilen zwischen der Zeile 50 und der Zeile 150.

**** Löscht alle von der niedrigsten Zeilennummer bis zur Zeile 35.

DIM

RUN PRO Programm

- FORMAT:**
1. DIM Numerischer Variablenname (Größe)
 2. DIM Zeichenfolge Variablenname (Größe)*Länge
 3. DIM Numerischer Feldname (Reihen,Spalten)
 4. DIM Zeichenfolge Feldname (Reihen,Spalten)*Länge

Abkürzung: D.

Vergleiche: ERASE, CLEAR

WIRKUNG:

DIM wird benutzt, um Speicher für numerische und Zeichenfolgen-Feldvariablen frei zu halten.

ANMERKUNGEN:

Außer für Felder der Form A() und A\$() muß jedes andere Feld mit einem DIM-Befehl initialisiert werden, um den nötigen Speicherraum bereit zu halten. Die Größe eines Feldes wird durch die Anzahl der Elemente in diesem Feld bestimmt.

Ein Feld kann maximal zwei Dimensionen haben; die maximale Größe einer Dimension ist 255. Zu der angegebenen Zahl von Elementen, die im DIM-Befehl definiert ist, addiert sich ein "nulltes" Element. So hat z.B. das Feld DIM B(3) die Elemente B(0), B(1), B(2) und B(3). In zweidimensionalen Feldern hat es eine extra "nullte" Reihe und Spalte.

In Zeichenfolgen-Feldern kann zusätzlich zur Anzahl der Elemente die Größe der einzelnen Zeichenfolge-Elemente vorgegeben werden. So reserviert z.B. DIM B\$(3)*12 Speicherraum für 4 Zeichenfolgen mit je 12 Zeichen Länge. Wird die Länge der Zeichenfolgen nicht vorgegeben, kann jede Zeichenfolge maximal 16 Zeichen enthalten.

Bei der Initialisierung eines Feldes werden alle Werte eines numerischen Feldes gleich Null und eines Zeichenfolgen-Feldes gleich leer gesetzt.

Was die DIMensionierung der Felder A und A\$ betrifft, informieren Sie sich im Abschnitt, in dem die Variablen erklärt werden.

Feld-Variablen können mit dem CLEAR- oder ERASE-Kommando gelöscht (oder undefiniert) gesetzt werden.

Wenn das Programm mit dem RUN-Kommando gestartet wird, werden die Feld-Variablen automatisch gelöscht.

Eine Variable kann nur einmal angegeben werden und jeder Versuch einer erneuten Angabe innerhalb eines Programms hat Fehlermeldungen zur Folge, wenn nicht zuerst das CLEAR- oder ERASE-Kommando gegeben wurde. Seien Sie also vorsichtig, wenn Sie ein Programm mit Hilfe eines GOTO-Kommandos oder der **DEF**-Taste ausführen, daß Sie nicht die gleiche DIM-Anweisung noch einmal ausführen, außer wenn Sie zuerst **CLEAR** betätigt haben.

BEISPIELE:

10: DIM B(10)

20: DIM C\$(4,4)*10

30: DIM F\$(12)

40: DIM H\$(4,6)

[10] Schafft Platz für ein numerisches Feld mit 11 Elementen.

[20] Schafft Platz für ein zweidimensionales Zeichenfolgen-Feld mit 5 Reihen und 5 Spalten; jede Zeichenfolge hat ein maximum von 10 Zeichen.

[30] Schafft Platz für eine Zeichenfolge- Variable F\$ mit 13 Elementen.

[40] Schafft Platz für ein zweidimensionales Zeichenfolge-Feld mit 5 Reihen und 7 Spalten; das sind 35 Elemente.

DSKF

Disketten RUN PRO Programm

FORMAT: 1. DSKF (1)
2. DSKF (3)

Abkürzung: DS.

Vergleiche:

WIRKUNG:

Dieses Kommando stellt auf einer RAM- oder Taschen-Diskette wieder freien Platz her.

ANMERKUNGEN:

DSKF stellt in Bit den Umfang einer freien Diskettenzone her.

Die Taschen-Diskette wird in Einheiten von 512 Bytes benutzt; ein 200 Byte Programm besetzt deshalb einen 512 Byte Bereich auf einer Taschen-Diskette. Die RAM-Diskette wird in Einheiten von 256 Bytes benutzt, und das gleiche Programm besetzt deshalb auf der RAM-Diskette einen Bereich von 256 Bytes.

Format 1 gibt den auf der Taschen-Diskette zur Verfügung stehenden Platz an.

Format 2 gibt den auf einer RAM-Diskette zur Verfügung stehenden Platz an.

Eine Fehlermeldung ERROR erscheint, wenn in Format 2 keine RAM-Karte vorhanden ist.

BEISPIELE:

>DSKF(1)

Zeigt den auf der Taschen-Diskette zur Verfügung stehenden Platz an.

END

Programm

FORMAT: 1. END**Abkürzung:** E.**Vergleiche:****WIRKUNG:**

Ein END-Befehl signalisiert das Ende eines Programms.

ANMERKUNGEN:

Sind mehrere Programme in den Speicher eingegeben, so ist es nötig, den einzelnen Programmen Endmarkierungen zu geben, damit bei der Ausführung eines Programms der Rechner nicht in ein anderes Programm hineinläuft. Dies wird mit dem END-Befehl als letzte Anweisung im Programm erreicht.

BEISPIELE:

```
10: PRINT "HELLO"  
20: END  
30: PRINT "GOODBYE"  
40: END
```

Mit diesen Programmen im Speicher druckt ein Kommando RUN 10 das Wort HELLO, aber nicht das Wort GOODBYE. Diese wird auf das Kommando RUN 30 ausgegeben.

EOF

Disketten Programm

FORMAT: 1. EOF (Dateinummer)

Abkürzung: EO.

Vergleiche:

WIRKUNG:

Bestimmt, wenn das Ende einer folgenden Datei erreicht wurde.

ANMERKUNGEN:

Die EOF-Funktion überprüft, ob alle Daten in einer folgenden Datei (mit der bestimmten Dateinummer) gelesen wurden. Verwenden Sie eine Dateinummer von #2 bis #7 für Taschen-Disketten basierte Dateien, #20 bis #25 für RAM-Disketten basierte Dateien.

Wurden alle Daten gelesen, kehrt EOF als Funktionswert 1 (richtig) zurück. Wenn nicht, erscheint es mit 0 (falsch).

Ist eine Datei mit einer bestimmten Nummer nicht offen, tritt ein Fehler auf.

Eine Fehlermeldung ERROR erscheint, wenn die Dateinummer innerhalb #20 bis #25 bestimmt wurde, aber keine RAM-Karte vorhanden ist.

BEISPIELE:

```
IF EOF(2) THEN CLOSE #2
```

Datei #2 wird geschlossen, wenn alle Daten in dieser Datei gelesen wurden.

BEISPIELE:

```
10: OPEN "X:A" FOR OUTPUT AS #2
20: PRINT #2, 123,456,789
30: CLOSE
40: OPEN "X:A" FOR INPUT AS #2
50: INPUT #2,A,B
60: X=EOF (2)
70: INPUT #2,C
80: Y=EOF (2)
90: CLOSE:END
```

[60] In dieser Zeile wurden nicht alle Daten gelesen, X=0.

[70] Alle Daten wurden gelesen, Y=1.

RUN PRO Programm

ERASE

FORMAT: 1. ERASE Feld 1, Feld 2, ... Feld n

Abkürzung: ER.

Vergleiche: CLEAR

WIRKUNG:

Löscht die angegebenen Felder.

ANMERKUNGEN:

ERASE löscht die angegebenen Felder. Der fixierte Feldbereich A(26) kann nicht gelöscht werden. Einzelne Feldelemente können nicht gelöscht werden. Es wird nur das ganze Feld gelöscht und sein Speicherbereich wird freigemacht. Sollen alle Variablen im Programm gelöscht werden, benutzen Sie CLEAR. Zum Neudefinieren einer Feldgröße, müssen Sie zuerst mit ERASE löschen und dann mit einer DIM-Anweisung neu definieren.

BEISPIELE:

```
10: DIM AA(10)
:
:
200: ERASE AA
```

ERL

RUN PRO Programm

FORMAT: 1. ERL

Abkürzung:**Vergleiche:** ERN, ON ERROR GOTO

WIRKUNG:

Gibt die Zeilennummer an, auf der sich bei der Programmausführung ein Fehler ereignete.

ANMERKUNGEN:

Die ERL-Funktion wird zusammen mit der ERN-Funktion und der ON ERROR GOTO-Anweisung bei Fehler-Verarbeitungs-Routinen verwendet, um zu kontrollieren, ob beim Programmablauf ein Fehler auftritt. Eine Zeilennummer wird nur bei ERL gesetzt, wenn während der Programmausführung ein Fehler auftritt.

BEISPIELE:

Siehe ERN

ERN

RUN PRO Programm

FORMAT: 1. ERN

Abkürzung:

Vergleiche: ERL, ON ERROR GOTO

WIRKUNG:

Gibt die Fehler-Codenummer des letzten Fehlers an.

ANMERKUNGEN:

Die ERL-Funktion wird zusammen mit der ERN-Funktion und der ON ERROR GOTO-Anweisung bei Fehler-Verarbeitungs-Routinen verwendet, um zu kontrollieren, ob beim Programmablauf ein Fehler auftritt. Eine Zeilennummer wird nur bei ERL gesetzt, wenn während der Programmausführung ein Fehler auftritt.

BEISPIELE:

```
10: ON ERROR GOTO 100
20: FOR N=1 TO 20
30: READ A
40: PRINT A
50: NEXT N
60: END
100: IF ERL =30 AND ERN =5 THEN PRINT "YOU HAVEN'T GOT A DATA LINE"
110: STOP
```

FILES

... RUN PRO Diskette

- FORMAT:**
1. FILES "d:"
 2. FILES "d:Dateiname"
 3. FILES "d:mehrdeutiger Dateiname"

Abkürzung: FI.

Vergleiche: LFILES, SET

WIRKUNG:

Zeigt die Namen und Ergänzungen einer oder mehrerer Dateien einer Taschen-Diskette oder RAM-Diskette an.

ANMERKUNGEN:

FILES zeigt den Dateinamen an, die Dateierweiterung (.BAS oder andere Erweiterung) und P (Schreibsperre) Ergänzung (siehe dazu SET-Kommando).

Bestimmen Sie das Gerät (d:) durch F: (RAM-Diskette) oder X: (Taschen-Diskette).

Wurde kein Dateiname bestimmt, werden alle Dateien des bestimmten Geräts auf dem Bildschirm bestimmt, jedes mal eine Angabe. Zum Blättern zur nächsten Eingabe drücken Sie die Tasten **↓** oder **ENTER**. Zur Anzeige der vorausgegangenen Eingabe drücken Sie **↑**. Zum Abbruch des Auflistens drücken Sie **C-CE** oder **BREAK**.

Zur Verzeichnis-Information einer einzelnen Datei bestimmen Sie einen einzelnen Dateinamen und eine Dateierweiterung.

Bestimmen Sie einen mehrdeutigen Dateinamen zur Auflistung einer Verzeichnis-Information über Dateigruppen mit gewöhnlichen Namensformen. Zu diesem Zweck stehen zwei "Joker" zur Verfügung. Der Stern "*" steht für jegliche Anzahl von Zeichen (einschließlich Ø) im Dateinamen. Das Fragezeichen "?" steht für ein einzelnes Zeichen im Dateinamen. Im folgenden finden Sie Beispiele für die Verwendung von "Jokerzeichen":

Dateispezifikation

TEST?

T??T

S?MPLE

A????

R*

aufgegriffene Dateien

TEST, TESTS, TEST1, TESTA

TEST, TEXT, TORT, TXYT

SIMPLE, SAMPLE, S2MPLE

ABCDEF, APPEND, A12345

RATES, R1, RETURNS, RAND2, R

FILES hat keinen Effekt, wenn die bestimmte Datei nicht im angegebenen Medium existiert.

Eine Fehlermeldung ERROR erscheint, wenn das bestimmte Gerät nicht aufgestellt und angeschlossen ist.

BEISPIELE:

>FILES"X:"

Listet alle Dateien im Taschen-Disketten-Antrieb X: auf den Bildschirm.

>FILES"F:DATA"

Zeigt Information der Datei DATA auf der RAM-Diskette an.

>FILES"F:???1"

Listet alle Dateien auf, die sich auf einer RAM-Diskette befinden und die aus 4 Zeichen bestehen und mit 1 enden.

FOR...NEXT

Programm

FORMAT: 1. FOR {Festvariable
Einfachgen., einstellige Variable} = Ausdruck 1 TO Ausdruck 2
2. FOR {Festvariable
Einfachgen., einstellige Variable} = Ausdruck 1 TO Ausdruck 2
STEP Ausdruck 3

Abkürzung: F. N. STE.

Vergleiche: NEXT

WIRKUNG:

FOR wird in Verbindung mit NEXT benutzt, um eine Reihe von Operationen mehrmals zu wiederholen.

ANMERKUNGEN:

FOR und NEXT schließen eine Gruppe von Anweisungen ein, die wiederholt werden sollen. Wird diese Gruppe das erste Mal ausgeführt, so hat die Schleifenvariable (die Variable, die unmittelbar nach FOR genannt wird) ihren Anfangswert (Ausdruck 1).

Erreicht die Ausführung nun den NEXT-Befehl, so wird die Schleifenvariable durch den Ergänzungswert erhöht und dann mit dem Schlußwert (Ausdruck 2) verglichen. Ist der Wert der Schleifenvariablen weniger oder gleich dem Ausdruck 2, so wird die eingeschlossene Gruppe ein weiteres Mal ausgeführt, beginnend mit dem Befehl hinter FOR. Der vorgegebene Ergänzungswert ist 1; in der zweiten Form wird die Erhöhung durch den Ausdruck 3 vorgegeben. Ist der Wert der Schleifenvariablen größer als Ausdruck 2, so wird die Ausführung unmittelbar hinter NEXT fortgesetzt. Da der Vergleich am Ende durchgeführt wird, werden die Anweisungen innerhalb eines FOR...NEXT Paares mindestens einmal ausgeführt.

Anfangs-, Schluß und Ergänzungswert muß im Bereich von $-9.999999999E99$ bis $9.999999999E99$ liegen. Wird der Ergänzungswert 0 gesetzt, wird FOR...NEXT in einer unendlichen Schleife ausgeführt.

Die Schleifenvariable kann in der umschlossenen Gruppe von Anweisungen benutzt werden, z.B. als ein Feld-Index. Man sollte aber vorsichtig sein, wenn der Wert der Schleifenvariablen verändert wird.

Die als Schleifenvariable verwendete numerische Variable kann eine numerische Festvariable oder eine einfachgenaue einstellige Zahlenvariable darstellen.

Programme sollen so geschrieben werden, daß die Ausführung nicht aus dem FOR...NEXT Paar herausspringt bevor der Zähler den Schlußwert erreicht. Um aus einer Schleife herauszukommen bevor sie die bestimmte Anzahl von Malen wiederholt wurde, setzen Sie die Schleifenvariable höher als den Schlußwert.

Die Gruppe von Anweisungen, die von den FOR...NEXT Paar eingefaßt werden, kann ein weiteres Paar von FOR..NEXT-Anweisungen enthalten, das eine andere Schleifenvariable haben muß, solange das eingeschlossene Paar vollständig eingeschlossen ist. Wenn also eine FOR-Anweisung in der Gruppe auftaucht, so muß auch der dazugehörige NEXT-Befehl eingeschlossen sein. So können bis zu fünf FOR...NEXT Paare ineinander "geschachtelt" sein. Ein illegales Herausspringen aus einer inneren Schleife hat die Falschmeldung ERROR 5 (Schachtelungsfehler) zur Folge.

Ein ERROR entsteht, wenn eine doppelt genaue Variable als numerische Variable bestimmt wird. Doppelt genaue Anfangswerte und Ergänzungswerte werden als einfach genaue Werte behandelt.

Innerhalb einer FOR...NEXT Schleife dürfen keinesfalls die Kommandos CLEAR oder ERASE verwendet werden.

BEISPIELE:

```

10: FOR I=1 TO 5
20: PRINT I
30: NEXT I
40: FOR N=10 TO 0 STEP -1
50: PRINT N
60: NEXT N
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

```

[10-30] Diese Gruppe von Anweisungen druckt die Zahlen 1,2,3,4,5 aus.

[40-60] Diese Gruppe von Anweisungen zählt rückwärts 10,9,8,7,6,5,4,3,2,1,0.

[70-130] Diese Gruppe von Anweisungen berechnet und druckt die Fakultäten N der Zahlen von 1 bis 10 aus.

GOSUB

Programm

FORMAT: 1. GOSUB Ausdruck
2. GOSUB "Etikett"

Abkürzung: GOS.

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

WIRKUNG:

Lenkt die Programmausführung auf eine BASIC-Subroutine ab.

ANMERKUNGEN:

Wenn die gleiche Gruppe von Anweisungen in einem Programm mehrere Male ausgeführt werden soll oder wenn diese Gruppe von Anweisungen in verschiedene Programme eingesetzt werden soll, so empfiehlt sich die Verwendung der BASIC-Subroutinen mit Hilfe von GOSUB und RETURN.

Die Gruppe von Anweisungen wird im Programm an der gleichen Stelle eingeschlossen, an der sie normalerweise nicht erreicht werden kann. Eine häufige Stelle folgt der END-Anweisung, die das Ende des Hauptprogramms angibt. An diesen Stellen des Hauptprogramms, an denen die Subroutinen ausgeführt werden sollen, fügen Sie die GOSUB-Anweisung mit der Zeilennummer der Startzeile der Subroutine als Ausdruck ein. Die letzte Zeile der Subroutine muß einen RETURN-Befehl enthalten. Wird der GOSUB-Befehl nun ausgeführt, so springt der PC-1280 an die angegebene Zeilennummer und verarbeitet die Anweisungen, bis ein RETURN-Befehl erreicht wird. Der Rechner springt dann in die Anweisung hinter dem GOSUB-Befehl zurück.

In einer Subroutine kann ein GOSUB vorkommen. Maximal können 10 Subroutinen ineinander "geschachtelt" werden.

Der Ausdruck in einer GOSUB-Anweisung darf kein Komma enthalten, z.B. 'A (1,2)' kann nicht benutzt werden. Da es zur Wahl verschiedener Subroutinen an gegebenen Stellen im Programm eine ON...GOSUB-Struktur gibt, besteht normalerweise der Ausdruck nur aus der gewünschten Zeilennummer. Wird ein

numerischer Wert benutzt, so muß er als eine gültige Zeilennummer bewertet werden, d.h. zwischen 1 und 65279, oder Sie erhalten eine Fehlermeldung ERROR 4.

Format 2 bewirkt, daß die Ausführung auf die Zeile mit dem gegebenen Etikett springt und dort startet.

BEISPIELE:

```
10: GOSUB 100  
20: END  
100: PRINT "HELLO"  
110: RETURN
```

In Ausführung druckt dieses Programm HELLO einmal.

GOTO

ALS DIREKTES EINGABEKOMMANDO

RUN

FORMAT: 1. GOTO Ausdruck
2. GOTO "Etikett"

Abkürzung: G.

Vergleiche: RUN

WIRKUNG:

Mit dem Kommando GOTO wird die Ausführung eines Programms begonnen.

ANMERKUNGEN:

Das Kommando GOTO kann anstelle des Kommandos RUN benutzt werden, um die Ausführung eines Programms an der durch Ausdruck bestimmten Zeilennummer zu starten.

GOTO unterscheidet sich in sechs Punkten von RUN:

- 1) Der Wert des WAIT-Intervalles wird nicht zurückgesetzt.
- 2) Wurde die Anzeige durch USING-Anweisung formiert, wird sie nicht gelöscht.
- 3) Variablen und Felder bleiben erhalten.
- 4) Der PRINT=LPRINT-Status wird nicht zurückgesetzt.
- 5) Der READ-Zeiger wird nicht zurückgesetzt.
- 6) Die Cursor-Spezifikation bleibt erhalten.

Die Ausführung eines Programms mit dem Kommando GOTO ist die gleiche wie die Ausführung mit der **DEF**-Taste.

Wird mehr als ein Programm gleichzeitig in den Speicher eingegeben, versehen Sie jedes Programm auf der ersten Zeile mit einem Etikett. Jedes beliebige Programm kann dann einfach durch Eingabe des GOTO-"Etiketts" ausgeführt werden.

BEISPIELE:**GOTO 100**

Die Ausführung des Programms wird bei Zeile 100 gestartet.

GOTO "A"

Die Ausführung wird bei Etikett A gestartet.

GOTO

BEI VERWENDUNG IN EINEM PROGRAMM Programm

FORMAT: 1. GOTO Ausdruck
2. GOTO "Etikett"

Abkürzung: G.

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

WIRKUNG:

GOTO veranlaßt den Rechner, von einer bestimmten Stelle in einem BASIC-Programm an eine andere Stelle zu springen. Anders als beim GOSUB "erinnert" sich GOTO nicht, von wo dieser Sprung ausgeführt wurde.

ANMERKUNGEN:

Der Ausdruck in einer GOTO-Anweisung darf kein Komma, wie z.B. in "A(1,2)" enthalten. Da eine ON...GOTO-Struktur vorhanden ist, um verschiedene Punkte an bestimmten Stellen im Programm anzuwählen, besteht normalerweise der Ausdruck nur aus der gewünschten Zeilennummer. Wird ein numerischer Wert benutzt, so muß er als eine gültige Zeilennummer bewertet werden, d.h. zwischen 1 und 65279, oder Sie bekommen die Fehlermeldung ERROR 4.

Gute Programme laufen normalerweise von Anfang bis Ende ohne Sprünge, abgesehen von Subroutinen, die während des Programms ausgeführt werden. Der prinzipielle Gebrauch des GOTO-Befehls ist deshalb in der Anweisung IF...THEN zu finden.

Format 2 bewirkt, daß die Ausführung auf die Zeile mit dem gegebenen Etikett springt und dort startet.

BEISPIELE:

```
10: INPUT A$
20: IF A$="Y" THEN GOTO 50
30: PRINT "NO"
40: GOTO 60
50: PRINT "YES"
60: END
```

Dieses Programm druckt das Wort "YES" aus, wenn Sie "Y" eingeben, und druckt das Wort "NO", wenn Sie etwas anderes eingeben.

GPRINT

RUN PRO Programm

FORMAT: 1. GPRINT "Zeichenfolge"
2. GPRINT Ausdruck[;Ausdruck;Ausdruck;...]
3. GPRINT

Abkürzung: GP.

Vergleiche: CURSOR, PRINT, WAIT

WIRKUNG:

Zeigt das angegebene Punktmuster an.

ANMERKUNGEN:

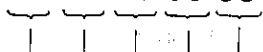
Das GPRINT-Kommando gibt das angegebene Punktmuster aus. Jedes Spalte besteht aus sieben vertikalen Punkten.

In einer GPRINT-Zeichenfolge können bis zu 5 Spalten von Bitbilddaten spezifiziert werden. Dies entspricht der Displaybreite. Werden mehr als 5 Spalten eingegeben, werden die übrigen Spalten ignoriert. Bei weniger als 5 Spalten werden die restlichen als "Null"-Spalten eingefügt, um 5 Spalten zu erhalten.

In Format 1 wird das 7-Punktmuster in 3 untere und 4 obere Punkte geteilt. Jede Punktgruppe wird dann durch Hexadezimalzahlen repräsentiert. (0 bis F für die oberen 4 Punkte. Das unterste Bit in der unteren Einheit wird ignoriert, so daß 0 bis 7 für die unteren 4 Verwendung findet und 8 bis F tatsächlich mit 0 bis 7 identisch ist.) Die Zeichenfolge ist umschlossen von " ".

Hex.zahl	Punktmuster	Hex.zahl	Punktmuster	Hex.zahl	Punktmuster	Hex.zahl	Punktmuster
0		4		8		C	
1		5		9		D	
2		6		A		E	
3		7		B		F	

GPRINT "00 00 00 00 00"



Jedes Nummernpaar repräsentiert ein vertikales Punktmuster (von 7 Punkten). Die erste Nummer repräsentiert die unteren 3 Punkte, die zweite Nummer die oberen 4 Punkte.

BEISPIELE:

GPRINT "01 03 0C 10 70"

— Repäsentiert das obere 4-Punktmuster

— Repäsentiert das untere 3-Punktmuster



GPRINT 1;3;12;16;112 produziert das gleiche Punktmuster.

Setzen Sie einen Strichpunkt (;) ans Ende der Zeichenfolge, damit der Cursor automatisch an die nächste Position gebracht wird.

In Format 2 wird ein senkrechtes 7-Punktmuster als Hexadezimal- oder Dezimalwert wiedergegeben. Ein "Gewicht" wird jedem Punkt in den senkrechten 7-Punktmuster wie unten angegeben zugewiesen.

□	←	1
□	←	2
□	←	4
□	←	8
□	←	10
□	←	20
□	←	40

Gewicht von jedem Punkt
(hexadezimal)

□	←	1
□	←	2
□	←	4
□	←	8
□	←	16
□	←	32
□	←	64

Gewicht von jedem Punkt
(dezimal)

Geben Sie das Punktmuster mit einem numerischen Wert an, mit der Summe der Gewichte der Punkte, die auf den Bildschirm zu setzen sind. Der Wert kann dabei zwischen 0 und 255 liegen.

In Format 3 wird das vorher bestimmte und angezeigte Muster unverändert dargestellt.

ACHTUNG:

Wird das GPRINT-Kommando mit einem ";" abgeschlossen, hat das folgende Kommando von der unmittelbar folgenden Cursorposition aus Wirkung (das ";" verkettet die Kommandos). Wird das GPRINT-Kommando mit einem "." beendet oder durch Druck auf **ENTER**, geht die Spezifikation für die vertikale Koordinate (X-Koordinate) auf 0.

Wird nach dem Positionieren des Cursors auf der gleichen Zeile der mit GPRINT angezeigten Grafiken die PRINT-Anweisung gegeben, löscht der Druck die Grafiken rechts der Cursor-Position.

KOMMANDOS

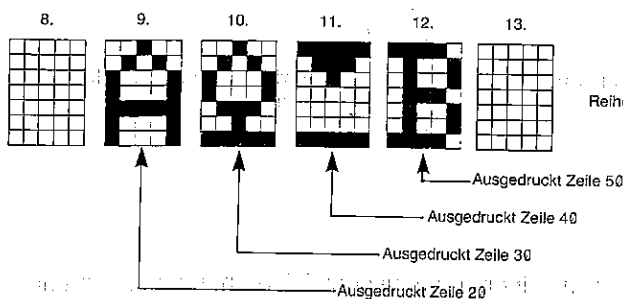
BEISPIELE:

```

5: WAIT 0:CLS
10: CURSOR 9,0
20: PRINT "A";
30: GPRINT "4C5271524C";
40: GPRINT &41;&43;&47;&43;&41;
50: PRINT "B"
60: WAIT :GPRINT

```

Zeichenposition des Displays von links



RUN PRO Programm

GRAD**FORMAT:** 1. GRAD**Abkürzung:** GR.**Vergleiche:** DEGREE, RADIANT

WIRKUNG:

GRAD wird benutzt, um die Form von Winkelwerten auf Neu-Grad umzuschalten.

ANMERKUNGEN:

Der PC-1280 kann 3 Formen der Winkeldarstellung verarbeiten: Alt-Grad (dezimal), Radial-Werte und Neu-Grad. Diese Formen werden für die Darstellung der Werte der Argumente von den SINus-, COSinus- und TANGens-Funktionen und der Ergebnisse der Umkehrfunktionen ArcusSiNus, ArcusCoSinus und ArcusTaNgens benutzt.

Die GRAD-Funktion wird benutzt, um Winkelwerte in Gradienten umzuformen, bis DEGREE oder RADIANT benutzt wird. Die Neu-Grad-Form stellt die Winkel-messung in Form von Prozentgradient dar, d.h. ein Winkel von 45° ist ein Gradient von 50%.

BEISPIELE:

10: GRAD

20: X= ASN 1

30: PRINT X

X hat jetzt den Wert 100, d.h. 100 Grad, dem Arcussinus von 1.

HEX\$

RUN PRO Programm

FORMAT: 1. HEX\$ (Ausdruck)

Abkürzung: H.\$

Vergleiche:

WIRKUNG:

Kehrt Dezimalzahlen in eine hexadezimale Zeichenfolge um:

ANMERKUNGEN:

Der Wert des Ausdrucks muß im Bereich -9999999999 bis 9999999999 liegen. Die entstehende Hexadezimal-Zeichenfolge hat bis zu 10 Stellen.

BEISPIELE:

C\$=HEX\$(12)+HEX\$(15)

C\$ erscheint erneut als Zeichenfolge "CF".

Programm**IF ... THEN****FORMAT:** 1. IF Bedingung [THEN] Anweisung**Abkürzung:** IF . . . T.**Vergleiche:****WIRKUNG:**

IF...THEN wird benutzt, um eine Anweisung auszuführen oder nicht auszuführen, wenn bestimmte Bedingungen bei der Programmausführung erfüllt sind.

ANMERKUNGEN:

Im normalen Ablauf von BASIC-Programmen werden die Anweisungen in der Reihenfolge ihres Auftretens abgearbeitet. Das IF...THEN-Befehlspaar erlaubt es, eine bestimmte Anweisung nur dann auszuführen, wenn eine bestimmte Bedingung erfüllt ist. Ist der Bedingungsteil der IF-Anweisung wahr, so wird die Anweisung ausgeführt, ist die Bedingung falsch, wird sie übersprungen.

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

Die Anweisung die auf den THEN-Teil folgt, kann jede BASIC-Anweisung sein, auch eine weitere IF...THEN-Anweisung. Handelt es sich um eine LET-Anweisung, so muß der LET-Befehl angegeben werden.

Zur Anwendung von IF...THEN-Anweisungen mit Relativbeziehungen, siehe Seite 95.

BEISPIELE:

```

10: INPUT "CONTINUE? ";A$
20: IF A$="YES" THEN GOTO 10
30: IF A$="NO" THEN GOTO 60
40: PRINT "YES OR NO, PLEASE"
50: GOTO 10
60: END

```

Dieses Programm fragt andauernd "CONTINUE?", solange "YES" eingegeben wird; es stoppt wenn "NO" eingegeben wird, und beschwert sich andernfalls.

INIT

Diskette RUN PRO

FORMAT: 1. INIT "d:"

Abkürzung: INI.

Vergleiche:

WIRKUNG:

Initialisiert eine Diskette (RAM- oder Taschen-Diskette).

ANMERKUNGEN:

Initialisiert alle neuen Taschen-Disketten (X:) und RAM-Disketten (F:) mit dem INIT-Kommando.

Beim Initialisieren dieser Disketten werden alle Informationen auf der Diskette gelöscht.

Initialisieren Sie mit diesem Kommando beide Seiten einer neuen Taschen-Diskette. Tippen Sie INIT (X:) zum Initialisieren der Seite A; nehmen Sie die Diskette heraus, wenden Sie sie und führen Sie wieder ein, bevor Sie das Kommando wiederholen.

RAM-Karten können nur initialisiert werden, wenn MEM\$="2".

Eine Fehlermeldung ERROR erscheint, beim Versuch, eine RAM-Karte zu initialisieren wenn der PC-1280 nicht in die MEM\$="2"-Betriebsart geschaltet ist.

Eine Fehlermeldung ERROR erscheint, beim Versuch, eine RAM-Karte zu initialisieren, deren Inhalte Passwort-geschützt wurden (oder wenn die RAM-Karte selbst Passwort-geschützt wurde).

Eine Fehlermeldung ERROR erscheint, beim Versuch eine RAM-Karte zu initialisieren, die nicht im PC-1280 einlegt, oder eine 2KB RAM-Karte zu initialisieren.

Programm**INKEY\$**

FORMAT: 1. INKEY\$**Abkürzung:** INK.**Vergleiche:**

WIRKUNG:

Weist der bestimmten Variablen den Wert der betätigten Taste zu, während die INKEY\$-Funktion ausgeführt wird.

ANMERKUNGEN:

INKEY\$ wird benutzt, um auf die Betätigung einzelner Tasten zu antworten, ohne auf das Ende der Eingabe durch die **ENTER**-Taste zu warten.

Auf der nächsten Seite finden Sie eine Liste anwendbarer Tasten und Zeichen, die zurückgegeben werden.

BEISPIELE:

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

Die Zeilen 40 und dahinter enthalten Test für die Taste und die nötigen Bedienungsmaßnahmen.
(Zum Beispiel: 60: PRINT B)

INKEY\$ Zeichen Codetabelle

	HOCH	0	1	2	3	4	5	...	F
TIEF									
0		SHIFT	SPC	0		P			
1			!	1	A	Q			
2	C-CE	DEF	"	2	B	R			
3			#	3	C	S			
4	↑	SML	\$	4	D	T			
5	↓		%	5	E	U			
6				6	F	V			
7		BS		7	G	W			
8	BASIC		(	8	H	X			
9	CAL	M+	)	9	I	Y			
A	OFF			*		J	Z		
B	INS		+	;	K				
C	DEL		,		L				√
D	ENTER		-	=	M				
E	►		.		N				
F	◄		/		O				

Programm

INPUT

FORMAT: 1. INPUT ["Aufforderungszeichen" $\left\{ \begin{smallmatrix} \{ \} \end{smallmatrix} \right\}$] Liste der Variablen
 2. INPUT "Aufforderungszeichen" $\left\{ \begin{smallmatrix} \{ \} \end{smallmatrix} \right\}$ Variable,
 "Aufforderungszeichen" $\left\{ \begin{smallmatrix} \{ \} \end{smallmatrix} \right\}$ Variable

Abkürzung: 1.

Vergleiche: INPUT #, READ, CURSOR, PRINT

WIRKUNG:

INPUT wird benutzt um einen oder mehrere Werte über die Tastatur einzugeben.

ANMERKUNGEN:

Wollen Sie verschiedene Werte für jeden Programmablauf eingeben, benutzen Sie INPUT, um diese Werte über die Tastatur einzugeben.

In der einfachsten Form hat die INPUT-Anweisung keine Aufforderungs-Zeichenfolge, sondern ein Fragezeichen erscheint in der linken oberen Ecke des Displays. Sie geben dann einen Wert, gefolgt von **ENTER**, ein. Dieser Wert wird dann der ersten Variablen der Variablenliste zugewiesen. Sind mehrere Variablen in der gleichen INPUT-Anweisung enthalten, so wird dieser Prozess so oft wiederholt, bis die Liste erschöpft ist. Trennen Sie die Variablen durch Kommata.

Wenn ein Aufforderungszeichen in der INPUT-Anweisung enthalten ist, so ist der Prozess genau derselbe, nur daß anstelle eine Fragezeichens, ein Aufforderungszeichen in der linken oberen Ecke des Displays erscheint. Wenn dem Aufforderungszeichen ein Semikolon folgt, erscheint der Cursor nach ausgeführter Anweisung unmittelbar nach dem Aufforderungszeichen. Wenn dem Aufforderungszeichen ein Komma folgt, erscheint nach ausgeführtem Befehl das Aufforderungszeichen, und wenn Daten eingegeben wurden, erscheinen die eingegebenen Daten auf dem Bildschirm von der Stelle des ersten Zeichens im Aufforderungszeichen, so daß das Aufforderungszeichen überschrieben wird.

Wenn ein Aufforderungszeichen angegeben wird und die darauf folgende Variablenliste mehr als eine Variable enthält, so werden die zweite und die folgenden Variablen mit einem Fragezeichen gekennzeichnet. Beinhaltet die Liste ein zweites Aufforderungszeichen, so wird es für die unmittelbar darauf folgende variable angezeigt.

Wurde die Startposition des Bildschirms mit Hilfe des **CURSOR**-Kommandos vor der Ausführung des **INPUT**-Kommandos angegeben, so erscheint ein Aufforderungszeichen oder ein "?" an dieser Stelle.

Wenn Sie beim betätigen der **ENTER**-Taste keinen Wert eingeben, so behält die Variable den Wert, den sie vor der **INPUT**-Anweisung hatte.

Wenn Sie bei der Eingabe Ihres Wertes einen Fehler machen, drücken Sie die **C.CE**-Taste. Dies löscht die Eingabe und bringt den Cursor wieder auf die Startposition. Sie können auch die **◀** und **▶**-Tasten zur Positionierung des Cursors benutzen, aber diese löschen nicht zuerst die falsche Eingabe, was ihren Gebrauch schwieriger macht.

BEISPIELE:

```
10: INPUT A
20: INPUT "A=";A
30: INPUT "A=",A
40: INPUT "X=?";X,"Y=?";Y
```

[10] Setzt ein Fragezeichen an den linken Rand.

[20] Gibt "A=" aus und wartet auf Eingabe.

[30] Gibt "A=" aus. Bei der Eingabe verschwindet "A=" und der eingegebene Wert wird am linken Rand angezeigt.

[40] Gibt "X=?" aus und wartet auf die erste Eingabe. Nach dem **ENTER** wird "Y=?" am linken Rand angezeigt.

Diskette RUN PRO Programm

INPUT#

FORMAT: 1. INPUT# Dateinummer, Variable [,Variable,...,Variable]

Abkürzung: I.#

Vergleiche: OPEN, PRINT#

WIRKUNG:

Liest Dateieinheiten der folgenden Dateien auf Diskette (RAM- oder Taschen-Diskette).

ANMERKUNGEN:

Die Dateinummer ist die der Datei gegebene Nummer zur Eingabeöffnung mit dem OPEN-Befehl. Falls versucht wird, Dateieinheiten einer Datei einzugeben, die nicht eröffnet wurde, kommt eine Fehlermeldung.

Die Dateinummer muß eine im OPEN-Kommando bestimmte Nummer sein (für Taschen-Disketten basierte Dateien #2 bis #7 und für RAM-Disketten basierte Dateien #20 bis #25).

Bestimmen Sie die Variablen wie folgt:

- Fixierte Variable (A, X, B\$, etc.)
- Einfache Variable (CD, EF\$ etc.)
- Matricelemente (B(10), C\$(5,5) etc.)
- Ganze Felder (B(*), C\$(*) etc.)

Ein Fehler tritt auf, wenn eine Datei weniger Daten, als die Anzahl der bestimmten Variablen enthält.

Die Daten und Variablen müssen von der gleichen Art sein (z.B. müssen numerische Werte numerischen Variablen zugeordnet sein, Werte mit einfacher Genauigkeit Variablen mit einfacher Genauigkeit, Werte mit doppelter Genauigkeit Variablen mit doppelter Genauigkeit usw.)

Die Variablen müssen in der gleichen Reihenfolge wie in der Datei aufgelistet sein.

Als ein Abgrenzungszeichen verwenden Sie ein Komma (,), Leerzeichen (&20) oder CR (&0D) +LF (&0A), wenn Daten in numerischen Variablen gelesen werden. Leerstellen, die den Daten vorausgehen, werden dabei ignoriert.

Als ein Abgrenzungszeichen verwenden Sie ein Komma (,), CR (&0D) +LF (&0A), wenn Daten in Zeichen-Variablen gelesen werden. Leerstellen, die den Daten vorausgehen, werden dabei ignoriert. Erscheint ein doppeltes Anführungszeichen am Datenbeginn, so werden die Daten bis zum nächsten doppelten Anführungszeichen gelesen. Ein Komma in einer Zeichenfolge umgeben von doppelten Anführungszeichen ist kein Abgrenzungszeichen.

BEISPIELE:

```

10: A$ = "AB" + CHR$ 34 + "CDE" + CHR$ 34
20: B$ = CHR$ 34 + "CD,EF" + CHR$ 34
30: PRINT A$
40: PRINT B$
50: OPEN "X:ABC.DAT" FOR OUTPUT AS #2
60: PRINT #2, A$; ", "; B$
70: CLOSE
80: OPEN "X:ABC.DAT" FOR INPUT AS #2
90: INPUT #2, C$, D$
100: PRINT C$
110: PRINT D$
120: CLOSE : END

```

<Ausführung>

```

RUN  ENTER  AB"CDE"
      ENTER  "CD,EF"
      ENTER  AB"CDE"
      ENTER  CD, EF

```

Band RUN PRO Programm

INPUT#

FORMAT: 1. INPUT# ["Dateiname";] Variable [, Variable, ..., Variable]

Abkürzung: I.#

Vergleiche: INPUT, PRINT#, READ

WIRKUNG:

INPUT# wird benutzt, um Daten von einer Kassette oder Band einzugeben.

ANMERKUNGEN:

Folgende Variablentypen können von der INPUT#-Anweisung eingesetzt werden:

- (1) Festpunktvariablen—A, B C, A(7), D*, A(20)*, usw.,
- (2) Einfache Variablen—AA, B3, CP\$, usw.,
- (3) Feldvariablen—S(*), K\$(*), usw.

1) Übergabe von Daten auf Festpunktvariablen

Um Daten von der Kassette auf Festpunktvariablen zu übertragen, müssen die Variablennamen in der INPUT#-Anweisung eingegeben werden.

INPUT# "DATA 1" ;A,B,X,Y

Diese Anweisung weist die Daten der Kassetten-Datei namens "DATA 1" den Variablen A, B, X, Y in der gegebenen Reihenfolge zu.

Um alle verfügbaren Festpunktvariablen und, falls definiert, erweiterte Variablen (A(27) und weiter) mit Daten von der Kassette zu belegen, muß der ersten Variablen ein Stern (*) angehängt werden.

INPUT# "D-2";D*

Mit dieser Anweisung wird der Inhalt der Banddatei "D-2" den Variablen D bis Z und A(27) und höher zugewiesen.

INPUT# A(10)*(ohne DIMensionierung)

Diese Anweisung überträgt die Daten der ersten gefundenen Datei nach dem Bandanfang von der Kassette auf die Variablen A(10) und weiter (also J bis Z und A(27) und weiter).

Wurde bereits mit einem DIM-Befehl ein Feld namens A definiert, können keine indizierten Festpunktvariablen der Form A() definiert werden.

Die Übertragung von Daten auf Festpunktvariablen und erweiterte Variablen (A(27) und weiter) wird so lange fortgesetzt, bis das Ende der Ursprungsdatei auf dem Band gefunden wird, aber wenn der Speicher des Computers voll ist, erhalten Sie eine Fehlermeldung ERROR 6.

2) Datenübertragung auf einfache Variablen

Daten einer Kassettendatei können auf einfache Daten übertragen werden, indem die gewünschten einfachen Variablen in die INPUT#-Anweisung eingegeben werden.

INPUT# "DM-1";AB,Y1,XY\$

Diese Anweisung überträgt die Daten von der Banddatei namens "DM-1" auf die einfachen Variablen AB, Y1 und XY\$.

Numerische Daten müssen auf numerische einfache Variablen übertragen werden, und Zeichendaten müssen einfache Zeichenvariablen sein. Andere Übertragungen sind nicht möglich.

Im Programm-Datenbereich muß Raum für die einfachen Variablen freigehalten werden, bevor die INPUT#-Anweisung ausgeführt werden kann. Sonst erhalten Sie eine Fehlermeldung. Verwenden Sie Platzreservierungs-Anweisungen, um Plätze für die einfachen Variablen zu reservieren.

AA=0 ENTER

B1\$="A" ENTER

INPUT# AA, B1\$ ENTER

Verwenden Sie numerische Werte oder Zeichen, in den Reservierungs-Anweisungen, um Plätze für die Variablen zu reservieren.

3) Datenübertragung auf Feldvariablen

Um Daten von einer Banddatei auf Feldvariablen zu übertragen, muß der Namen des Feldes in der INPUT #-Anweisung in der Form Feldname (*) angegeben werden.

```
50 DIM B(5)
60 INPUT# "DS-4";B(*)
```

Diese Anweisung überträgt die Daten von der Banddatei namens "DS-4" auf die Variablen (B(0) bis B(5)) im Feld B.

Numerische Daten müssen auf numerische Feldvariablen gleicher Länge wie die der Daten; Zeichenvariablen müssen auf Zeichenfeldvariablen gleicher Länge wie die Daten übertragen werden. Wird dies nicht beachtet, erhalten Sie eine Fehlermeldung.

Sie müssen Raum im Programm- Datenspeicherbereich für die Feldvariablen freihalten, bevor die INPUT #-Anweisung ausgeführt wird. Sonst erhalten Sie eine Fehlermeldung. Benutzen Sie die DIM-Anweisung, um das Feld im voraus zu definieren.

—ACHTUNG—

Stimmt die Anzahl der Variablen in der INPUT #-Anweisung nicht mit der Zahl der Daten auf der Kassette überein, geschieht folgendes:

- Ist die Zahl der Daten in der Banddatei (die zu übertragen sind) größer als die Zahl der Variablen, so wird die Datenübertragung bis zur letzten Variable durchgeführt, und die restlichen Daten werden ignoriert.
- Ist die Zahl der Daten in der Banddatei (die zu übertragen sind) kleiner als die Zahl der Variablen, so werden alle Daten der Datei den Variablen bis an das Ende der Datei zugewiesen und die überzähligen Variablen behalten ihren alten Inhalt. In diesem Falle aber wartet der Computer auf die Datenübertragung von der Kassette. Um diesen Zustand zu stoppen, betätigen Sie die **BREAK**-Taste.
- Wird die INPUT #-Anweisung ohne Variable eingegeben, erhalten Sie die Fehlermeldung ERROR 1.

KILL

Diskette RUN PRO

FORMAT: 1. KILL "d:Dateiname.ext"

Abkürzung: K.

Vergleiche: SAVE, SET

WIRKUNG:

Löscht eine Datei auf einer Diskette (RAM- oder Taschen-Diskette).

ANMERKUNGEN:

Bestimmen Sie das Gerät (d:) und den Dateinamen. Auch die Erweiterung muß bestimmt werden.

Es tritt ein Fehler auf, wenn die bestimmte Datei nicht existiert oder eröffnet ist.

Ein Fehler tritt auf, wenn das bestimmte Gerät nicht aufgestellt oder angeschlossen ist.

Ein Fehler tritt auf, wenn das Dateiaattribut P ist oder eine RAM-Karte selbst geschützt worden ist. Ändern Sie das Attribut mit dem SET-Kommando, um eine Datei mit dem P Attribut oder eine Datei auf einer mit einem Codewort geschützten RAM-Karte zu löschen.

Kommt im Dateinamen eine Feldweiterung vor, so schließen Sie die Felderweiterung mit ein.

Verwenden Sie keine Wildcards (* oder ?) mit dem KILL-Kommando.

Verwenden Sie als Dateinamen keine Zeichenfolge-Variablen.

BEISPIELE:

KILL "X:PR01.BAS"

Löscht die Datei PR01.BAS auf einer Taschen-Diskette.

LEFT\$

RUN PRO Programm

FORMAT: 1. LEFT\$(X\$,N)
2. LEFT\$("Zeichenfolge",N)

Abkürzung: LEF.

Vergleiche: MID\$, RIGHT\$

WIRKUNG:

Nimmt die Zeichen N vom linken Ende einer Zeichenfolge X\$ heraus.

ANMERKUNGEN:

Der Wert N muß zwischen 0 und 80 liegen. Bruchteile werden abgerundet (abgeschnitten). Ist N kleiner als 1, wird eine Nullzeichenfolge herausgenommen. Ist N größer als die Anzahl der Zeichen in X\$, wird die ganze Zeichenfolge herausgenommen.

BEISPIELE:

```
10: X$="SHARP"  
20: FOR N=1 TO 5  
30: LET S$=LEFT$(X$,N)  
40: PRINT S$  
50: NEXT N
```

```
RUN  
S  
SH  
SHA  
SHAR  
SHARP
```

LEN

RUN PRO Programm

FORMAT: 1. LEN(X\$)
2. LEN("Zeichenfolge")

Abkürzung:

Vergleiche:

WIRKUNG:

Errechnet die Anzahl der Zeichen in der Zeichenfolge X\$.

ANMERKUNGEN:

Die Anzahl der Zeichen in der Zeichenfolge schließt alle Leerzeichen oder Druckzeichen wie Kontrollcodes oder Wagenrückläufe ein.

BEISPIELE:

```
10: INPUT "ENTER A WORD ";A$
20: N= LEN A$
30: PRINT "THE WORD LENGTH IS ";N
40: END
```

RUN

```
ENTER A WORD CHERRY
THE WORD LENGTH IS 6.
```

[10] Gibt ein Wort ein. In diesem Beispiel gibt der Anwender "CHERRY" ein.
[20] Ermittelt die Länge eines Worts.
[30] Druckt die Antwort aus.

LET**RUN PRO Programm****FORMAT:** 1. [LET] Variable=Ausdruck**Abkürzung:** LE.**Vergleiche:** IF...THEN**WIRKUNG:**

LET wird benutzt, um einer Variablen einen Wert zuzuweisen.

ANMERKUNGEN:

Mit LET weisen Sie der angegebenen Variablen den Wert des Ausdrucks zu. Die Form des Ausdrucks muß mit der Variablen übereinstimmen, d.h. nur numerische Ausdrücke können numerischen Variablen und nur Zeichefolge-Ausdrücke können Zeichenfolgen-Variablen zugewiesen werden. Um Werte aus der einen Form in die andere zu übertragen, muß eine der beiden expliziten Umsetzungsfunktionen (STR\$ oder VAL) eingesetzt werden.

Der LET-Befehl kann in allen LET-Anweisungen weggelassen werden, es sei denn, die Anweisungen folgen auf das THEN einer IF...THEN-Anweisung.

BEISPIELE:

```
10: I=10
20: A=5*I
30: X$=STR$(A)
40: IF I>=10 THEN LET Y$=X$+"00"
```

```
[10] Weist I den Wert 10 zu.
[20] Weist A den Wert 50 zu.
[30] Weist X$ den Wert 50 zu.
[40] Weist Y$ den Wert 50.00 zu.
```

LFILES

Diskette RUN PRO

FORMAT: 1. LFILES ["d:[Dateiname]"]

Abkürzung: LF.

Vergleiche: FILES

WIRKUNG:

Druckt die Namen und Attribute der bestimmten auf Diskette gespeicherten Datei(en).

ANMERKUNGEN:

Bestimmen Sie einen besonderen Dateinamen, um den Namen und die Attribute nur von dieser Datei auszudrucken.

Wildcards (* und ?) können zur Bestimmung von Dateien verwendet werden.

Ein FEHLER tritt auf, wenn das angegebene Gerät nicht aufgestellt und angeschlossen ist.

Ein FEHLER tritt auf, wenn die bestimmte Datei nicht vorhanden ist.

PRO**LIST**

FORMAT: 1. LIST
2. LIST Zeilennummer
3. LIST "Etikett"

Abkürzung: L.
Vergleiche: LLIST

WIRKUNG:

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

ANMERKUNGEN:

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

Mit Format 1 wird das Programm von der ersten Zeile angezeigt, bis das Display voll ist. Mit Format 2 wird das Programm von der angegebenen Zeilennummer angezeigt, bis das Display voll ist. Mit der  -Taste kommen Sie in die nächste Zeile der Liste.

Gibt es die angegebene Zeilennummer nicht, so wird das Programm ab der Zeile angezeigt, deren Zeilennummer der angegebenen am nächsten kommt.

Mit Format 3 wird das Programm ab der mit diesem Etikett versehenen Zeile angezeigt, bis das Display voll ist.

Sind mit dem MERGE-Kommando mehrere Programme in den Speicher geladen, so wird mit dem LIST-Kommando das letzte eingeladene Programm gelistet.

Wird aber das im Format 3 angegebene Etikett im zuletzt geladenen Programm nicht gefunden, so sucht das Kommando das Etikett vom ersten Programm an ab. Wird das angegebene Etikett gefunden, so wird die dazugehörige Zeile angezeigt. Falls ein Passwort angegeben wurde, wird das LIST-Kommando ignoriert.

BEISPIELE:

LIST 100

Zeigt die Zeilennummer 100 an.

LLIST

RUN PRO

- FORMAT:**
1. LLIST
 2. LLIST Zeile#
 3. LLIST Zeile#, Zeile#
 4. LLIST Zeile#,
 5. LLIST, Zeile#

Abkürzung: LL.

Vergleiche: LIST

WIRKUNG:

Druckt auf dem zusätzlichen Drucker CE-126P ein Programm aus.

ANMERKUNGEN:

Das LLIST-Kommando kann im PRO- oder RUN-Modus verwendet werden.

Format 1 druckt alle im Speicher befindlichen Programme aus.

Format 2 druckt nur die Programmzeilen, deren Nummer bestimmt wurde.

Format 3 druckt die Anweisungen der ersten Zeilennummer bis zur zweiten. Es müssen zwischen den Ziffern mindestens zwei Zeilen liegen.

Format 4 druckt alle Programmzeilen aus, die mit der bestimmten Zeilennummer beginnen.

Format 5 druckt alle Zeilen bis und einschließlich der mit der angegebenen Zeilennummer.

Wenn mit dem MERGE-Kommando mehrere Programme eingeladen wurden, so arbeitet das LLIST-Kommando mit dem zuletzt geladenen Programm. Um ein zuvor geladenes Programm aufzurufen, benutzen Sie das Kommando:

LLIST "Etikett",

Falls ein Passwort eingegeben wurde, wird das LLIST-Kommando ignoriert.

BEISPIELE:**LLIST 100,200**

Listet die Anweisungen zwischen den Zeilennummern 100 und 200 auf.

LOAD

Diskette RUN PRO

FORMAT: 1. LOAD "d:Dateiname"[;R]**Abkürzung:** LOA.**Vergleiche:** CLOAD, CSAVE, SAVE und CHAIN**WIRKUNG:**

Lädt ein BASIC-Programm von der bestimmten Diskette.

ANMERKUNGEN:

LOAD lädt ein Programm mit dem bestimmten Dateinamen. Die Inhalte aller Variablen (mit Ausnahme von der fixierten Variablen) werden gelöscht, wenn es einen ungenügenden Speicher gibt, die Programme darin zu laden, ohne es so zu machen. LOAD schließt alle Dateien. Lesen Sie die Beschreibung des CLOSE-Kommandos.

Ist die Ausdehnung in ".BAS" kann sie weggelassen werden.

Programme werden nicht geladen, wenn beim Lesen ein Fehler auftritt.

Ein nicht geschütztes Programm (d.h. ein nicht durch ein Passwort geschütztes Programm) kann geschützt werden, wenn sich der Computer beim Laden in der geschützten Betriebsart befindet. Zum Initialisieren des Computerspeichers betätigen Sie den RESET-Schalter und laden dann das Programm.

Sobald das Programm geladen ist, läßt man es mit R laufen.

Ein FEHLER tritt auf, wenn das angegebene Gerät nicht aufgestellt und angeschlossen ist oder wenn die angegebene Datei nicht existiert.

ASCII-Format Dateien können nicht auf der RAM-Diskette geladen werden.

Diskette RUN PRO Programm**LOC****FORMAT:** 1. LOC Dateinummer**Abkürzung:****Vergleiche:** OPEN**WIRKUNG:**

Meldet die befindliche Anzeigenposition (boolesch) in einer Datei.

ANMERKUNGEN:

Das LOC-Kommando gibt die Anzahl der Aufnahmen an, die, seit die angegebene Nummer eröffnet wurde, gelesen und geschrieben wurden. Eine Aufzeichnung hat eine Kapazität von 256 Bytes.

Ein Fehler tritt auf, wenn eine Datei mit der bezeichneten Nummer nicht geöffnet wurde.

Das LOC-Kommando kann nur mit Dateien auf Diskette (RAM- oder Taschen-Diskette) verwendet werden.

Taschen-Disketten basierte Dateien haben Nummern #2 bis #7. RAM-Disketten basierte Dateien haben Nummern #20 bis #25.

Ein FEHLER tritt auf, wenn das bestimmte Gerät nicht aufgestellt und angeschlossen ist.

BEISPIELE:

```

10: OPEN "X:FILE01" FOR INPUT AS #2
20: IF EOF(2) THEN 50
30: INPUT #2,N
40: GOTO 20
50: M=LOC(2)
60: PRINT "THE FILE HAS ";M;" RECORDS"
70: CLOSE #2
80: END

```

LOF

Diskette RUN PRO Programm

FORMAT: 1. LOF Dateinummer**Abkürzung:****Vergleiche:** OPEN**WIRKUNG:**

Gibt die Größe der bestimmten Datei an.

ERKUNGEN:

Das LOF-Kommando gibt die Größe einer Datei mit der bestimmten Dateinummer an. Die tatsächliche Größe der Datei erscheint in Bytes auf dem Display.

Taschen-Disketten werden in Einheiten von 512 Bytes und RAM-Disketten in Einheiten von 256 Bytes verwendet. Die Gesamtgröße aller Diskettendateien ist deshalb nicht gleich der Größe der benutzten Diskettenzone (Anzahl der Bytes).

Ein FEHLER tritt auf, wenn das bestimmte Gerät nicht aufgestellt und angeschlossen ist oder wenn eine Datei mit der angegebenen Nummer nicht gefunden wird (Taschen-Disketten basierte Dateien #2 bis #7 und RAM-Disketten basierte Dateien #20 bis #25).

BEISPIELE:

```
10: OPEN "X:FILE01" FOR INPUT AS #2
20: N= LOF (2)
30: PRINT "FILE01 FILE SIZE IN BYTES IS ";N
40: CLOSE #2
50: END
```

[10] Öffnet die Datei FILE01 zur Eingabe.

[20-30] Findet die Größe der Datei und druckt den Wert aus.

[40] Schließt die Datei.

KOMMANDOS

RUN PRO Programm

LPRINT

FORMAT: 1. LPRINT $\left\{ \begin{smallmatrix} \text{Zeich.} \\ \text{Ausdr.} \end{smallmatrix} \right\} [, \left\{ \begin{smallmatrix} \text{Zeich.} \\ \text{Ausdr.} \end{smallmatrix} \right\}]$
 2. LPRINT $\left\{ \begin{smallmatrix} \text{Zeich.} \\ \text{Ausdr.} \end{smallmatrix} \right\} [; \left\{ \begin{smallmatrix} \text{Zeich.} \\ \text{Ausdr.} \end{smallmatrix} \right\} , \dots , \left\{ \begin{smallmatrix} \text{Zeich.} \\ \text{Ausdr.} \end{smallmatrix} \right\}] [;]$

Abkürzung: LP.

Vergleiche: PRINT, USING

WIRKUNG:

LPRINT gibt Daten an den Drucker aus.

ANMERKUNGEN:

Wird mit Format 1 ein Datenfeld bestimmt, so werden numerische Werte vom rechten Papierrand aus ausgedruckt; Zeichenfolgen vom linken Rand.

Ergebnisse von Berechnungen mit doppelter Genauigkeit, die länger als 24 Anschläge sind, oder Zeichenfolgen mit mehr als 24 Anschlägen, werden automatisch beim Ausdruck auf der nächsten Zeile fortgesetzt.

Trennen Sie Ausdrücke oder Zeichenfolgen durch ein Komma (,), um die 24 Anschläge jeder Druckzeile in zwei 12 Anschlag-Felder zu teilen. Numerische Werte mit einfacher Genauigkeit oder Zeichenfolgen werden in jedes dieser Felder gedruckt; numerische Berechnungen werden vom rechten Rand jedes 12 Anschlag-Feldes ausgedruckt, Zeichenfolgen vom linken Rand. Numerische Werte mit mehr als 12 Anschlägen werden vom rechten Rand des folgenden 12 Anschlag-Feldes auf Papier gedruckt. Zeichenfolgen mit mehr als 12 Anschlägen werden vom linken Rand des ersten Feldes ausgedruckt und gehen in das zweite Feld über.

Zum Beispiel, wenn A=123, B=456, AB#=12345678901234
LPRINT AB#,A,B wird folgendes ausgedruckt:

```
12345678901234.
123.           456.
```

LPRINT A,AB#,B dagegen bewirkt folgenden Ausdruck:

```
123.           123
45678901234.   456.
```

Mit Format 2 werden die Daten in Folge vom linken Rand aus ausgedruckt. Werte oder Zeichenfolgen mit mehr als 24 Zeichen (Maximum=96) werden automatisch auf der nächsten Zeile fortgesetzt.

Der Ausdruck kann ein Wert mit doppelter Genauigkeit sein.

RUN PRO Programm

LSET

FORMAT: 1. LSET Zeichenfolge Variable=Zeichenfolge

Abkürzung: LS.

Vergleiche: RSET, LET

WIRKUNG:

Speichert Daten, links beginnend, in einem Zeichenfolge Variable Feld.

ANMERKUNGEN:

Das LSET-Kommando (Links SET) speichert Datenzeichen in dem Zeichenfolge Variable Feld. Die eingegebenen Daten beginnen im Feld von links.

Unmittelbar nach dem LSET-Kommando geben Sie eine bestimmte Anweisung ein.

Geben Sie einen Ausdruck ein, wie z.B. B\$="ABC", mit der Zeichenfolge Variable links und dem Zeichenfolge Ausdruck rechts. In diesem Fall bedeutet das "=" Zeichen kein Gleichheitszeichen, sondern daß die Inhalte rechts vom Zeichen zur Zeichenfolge Variablen links gehören.

Das LSET-Kommando hat die gleiche Funktion, als wenn ein Zeichenfolge Ausdruck mit dem LET-Kommando als eine Zeichenfolge Variable bestimmt wird.

LSET B\$="ABC" speichert die Zeichenfolge "ABC" links vom Feld B\$:

B\$ =

A	B	C				
---	---	---	--	--	--	--

Wenn die der Variablen zugeordnete Zeichenfolge kürzer als die Variable ist, wird der verbleibende Platz mit Leerstellen besetzt. (Die verbleibenden Zwischenräume werden bei Verwendung des LET-Kommandos nicht mit Leerstellen besetzt.)

MDF

RUN PRO Programm

FORMAT: 1. MDF(Ausdruck)
2. MDF(Ausdruck, Schwellwert-Nummer)

Abkürzung: MD.

Vergleiche: USING

WIRKUNG:

Der MDF-Befehl wird benutzt, um den Wert eines Ausdrucks aufzurunden.

ANMERKUNGEN:

MDF ist eine Funktion, die benutzt wird, um den Wert eines Ausdrucks auf die durch das USING-Kommandos angegebene Anzahl von Dezimalstellen aufzurunden.

Dieser Befehl ist nur effektiv, wenn die Anzahl der Dezimalstellen mit Hilfe des USING-Kommandos für einen Wert definiert wurde.

Format 1 benutzt die normale Schwellwert-Nummer 4. Dies bedeutet, daß wenn die erste Stelle des abgerundeten Teils größer ist als 4, eine Stelle zum nicht abgerundeten Teil übertragen wird. Diese Schwellwert-Nummer kann im Format 2 definiert werden:

BEISPIELE:

USING "###.###"

MDF(0.5/9)

0.056

USING "###.###"

MDF(0.5/9,5)

0.055

10: USING "###.###"

20: A= MDF(5/9)

30: PRINT A

40: USING

50: PRINT A,5/9

60: END

R U N ENTER

0.556

ENTER

0.556 5.55555E-01

KOMMANDOS

MEM

RUN PRO Programm

FORMAT: 1. MEM**Abkürzung:** M.**Vergleiche:** MEM\$, SET MEM

WIRKUNG:

Gibt den zur Eingabe von Daten und Programmen im Speicher zur Verfügung stehende Platz an.

ANMERKUNGEN:

MEM gibt den freien Programmierbereich im Speicher an. Dieser Wert hängt von dem Gebrauch der RAM-Karten im Schlitz ab (siehe MEM\$ und SET MEM).

Bitte beachten Sie, daß wenn Sie "D" mit Hilfe von SET MEM einstellen, der ganze Bereich, der für die RAM-Karte angegeben ist, nicht vollständig frei ist für das Programm und die Daten, und zwar weil die Programm- und Variablenspeicherbereiche auf die entsprechenden RAM-Bereiche begrenzt sind (der eingebaute Speicher und die Karte im Schlitz auf der Rückseite des PC-1280). Wenn irgendein Speicher voll besetzt ist, kann der restliche Freibereich im anderen Speicher (der eingebaute oder die RAM-Karte) nicht voll benutzt werden.

MEM\$

RUN PRO Programm

FORMAT: 1. MEM\$

Abkürzung: M.\$

Vergleiche: MEM, SET MEM

WIRKUNG:

MEM\$ gibt die RAM Konfiguration an.

ANMERKUNGEN:

MEM\$ wird ohne Parameter eingegeben und gibt den Wert von "1", "2", "B" oder "D" an.

"1" bedeutet, daß nur die RAM-Karte im Schlitz verwendet wird.

"2" bedeutet, daß nur der eingebaute Speicher verwendet wird.

"B" und "D" bedeutet, daß sowohl die RAM-Karte als auch der eingebaute Speicher verwendet werden.

"B" und "D" geben verschiedene Möglichkeiten an, wie diese beiden Speicherbereiche verwendet werden (siehe SET MEM).

MERGE

Diskette Band RUN PRO

FORMAT: 1. MERGE
2. MERGE "[d:]Dateiname"

Abkürzung: MER.

Vergleiche: CLOAD, CHAIN

KOMMANDOS

WIRKUNG:

Mit dem MERGE-Kommando kann man ein auf Band oder Diskette gespeichertes Programm laden und mit einem bereits im Speicher vorhandenen Programm mischen.

ANMERKUNGEN:

Wenn kein Gerät bestimmt wurde, wird ein Bandgerät angenommen. Bestimmen Sie andere Geräte wie folgt:

F: RAM Diskette

X: Taschen-Disketten-Betrieb

Mit dem MERGE-Kommando wird das bereits im Speicher geladene Programm gesichert und ein weiteres Programm vom bezeichneten Gerät in den Rechner geladen. Auf diese Weise können verschiedene Programme gleichzeitig in den Speicher des PC-1280 geladen werden. Gemischte Programme können die gleichen Zeilennummern haben.

Wenn 2 oder mehrere Programme im Speicher enthalten sind, beziehen sich die RUN und GOTO-Kommandos immer auf das letzte zu mischende Programm, außer wenn Sie Programmetiketten verwenden. Mit dem RUN "Etikett" oder mit dem GOTO "Etikett" starten Sie jedes andere Programm, außer das zuletzt in den Speicher gemischte. Etikette müssen zugeordnet werden, bevor Programme gemischt werden, da Programme nicht redigiert werden können, wenn nach ihnen andere eingemischt wurden. Haben zwei oder mehrere Programme das gleiche Etikett, wird das letzte zu mischende Programm ausgeführt.

Alle Dateien werden geschlossen, wenn ein binäres Programm geladen wird.

Verknüpfung Passwort-geschützter Programme

Passwort-geschützte Programme können mit dem MERGE-Befehl nicht geladen werden, wenn die Programme bereits im Speicher geschützt sind.

Alle Programme im Speicher werden zu Passwort-geschützten, wenn mit MERGE ein Passwort-geschütztes Programm geladen wird.

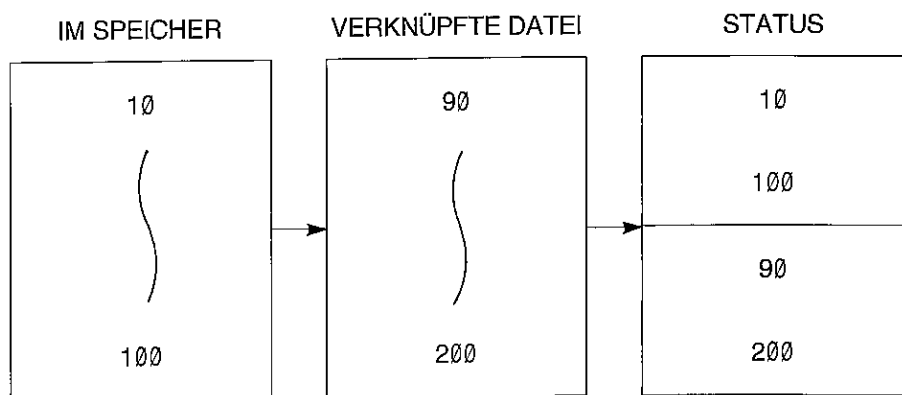
Alle Programme im Computer-Speicher werden zu Passwort-geschützten, wenn mit MERGE ein Passwort-geschütztes Programm geladen wird.

Verwenden Sie als Dateinamen keine Zeichenfolgen-Variablen.

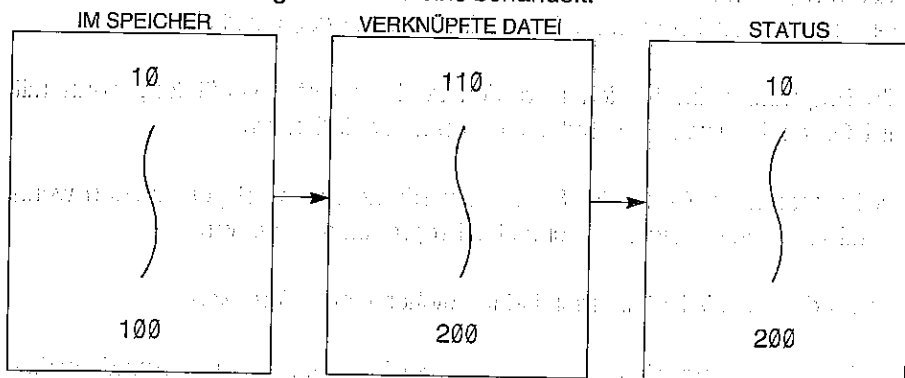
Die .BAS-Erweiterung kann ausgelassen werden, wenn die Programmdatei diese Erweiterung aufweist.

Eine Fehler-Meldung erscheint, wenn das bestimmte Gerät nicht aufgestellt und angeschlossen ist oder die bestimmte Datei nicht existiert.

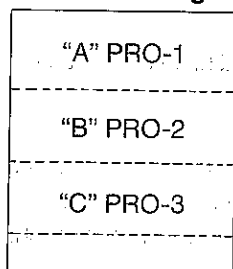
Ist die Zeilennummer des mit dem MERGE-Kommando geladenen Programms kleiner als die letzte Zeilennummer des vorher geladenen Programms, werden die beiden Programme, wie in der folgenden Übersicht gezeigt, als zwei verschiedene behandelt.



Ist die erste Zeilennummer des mit dem MERGE-Kommando geladenen Programms größer als die letzte Zeilennummer des vorher geladenen Programms, werden die beiden Programme wie eins behandelt.



Ausführen eines gemischten Programms



Die Abbildung zeigt die Speicherorganisation, wenn PRO-1 geladen wird, nachdem PRO-2 und PRO-3 mit dem MERGE-Kommando geladen wurden. Wird das Programm mit RUN oder GOTO gestartet, wird PRO-3 gestartet. Wird RUN "Etikett", GOTO "Etikett" oder eine bestimmte Taste verwendet, sucht PRO-3 nach dem bestimmten Etikett, von dem aus das Programm startet.

Wird das bestimmte Etikett nicht in PRO-3 gefunden, wird die Suche im nächsten zuletzt gemischten Programm, PRO-2, fortgesetzt.

Verwendung von MERGE mit RAM-Disketten

Es ist nicht möglich, mit Hilfe des MERGE-Befehls ein Programm auf RAM-Diskette mit einem Programm im Speicher zu einem durchgehenden Programm zu kombinieren. Der Versuch resultiert in einer Fehlermeldung.

Falls der MERGE-Befehl zum Laden des Programmes von RAM-Diskette verwendet und das Programm separat vom Programm im Speicher angelegt werden soll, zuerst eine Scheinzeile am Ende des Programms im Speicher anfügen. Dann mit dem Programm auf RAM-Diskette kombinieren. Für die Scheinzeile die Nummer 65279 verwenden.

Bsp.: 65279: END

BEISPIELE:

>MERGE "F:PROFIT"

Mischt die PROFIT genannte Datei auf der RAM-Diskette in den Speicher.

RUN PRO Programm**MID\$**

FORMAT: 1. MID\$(X\$,N,M)
2. MID\$("Zeichenfolge",N,M)

Abkürzung: MI.

Vergleiche: LEFT\$, RIGHT\$

WIRKUNG:

Erstellt eine Zeichenfolge von M Zeichen in der Zeichenfolge X\$, beginnend beim N-ten Zeichen in der Zeichenfolge X\$.

ANMERKUNGEN:

Ist N kleiner als 1 oder größer als die Anzahl von Zeichen in X\$, wird eine Null-Zeichenfolge erstellt. M muß zwischen 0 und 80 liegen und N muß zwischen 1 und 80 liegen. Bruchteile werden abgerundet.

BEISPIELE:

```
10: Z$="ABCDEFGG"  
20: LET Y$= MID$(Z$,3,4)  
30: PRINT Y$
```

```
RUN  
CDEF
```

NAME

Diskette RUN PRO

FORMAT: 1. NAME "d:alter Dateiname" AS "d:neuer Dateiname"

Abkürzung: NA.

Vergleiche:

WIRKUNG:

Benennt die Dateien auf einer Diskette um.

ANMERKUNGEN:

Das NAME-Kommando benennt die Diskettendatei "alter Dateiname" in "neuer Dateiname" um, und zwar auf RAM- (F:) oder Taschen-Disketten (X:).

Ein Fehler tritt auf, wenn der "alte Dateiname" nicht oder der "neue Dateiname" schon existiert.

Ein Fehler tritt auf, wenn der "alte Dateiname" mit der P-Funktion (Schreib-Schutz) gesichert wurde.

Ein Fehler tritt auf, wenn entweder der "alte Dateiname" oder der "neue Dateiname" geöffnet wurde.

Ein Fehler tritt auf, wenn das bestimmte Gerät nicht aufgestellt und angeschlossen ist.

Die Erweiterung muß eingeschlossen sein, wenn der Dateiname eine enthält.

Wird das Gerät (d:) weggelassen, so wird eine Taschen-Diskette angenommen. Eine RAM-Diskette muß mit F angegeben werden.

Verwenden Sie als Dateinamen keine Zeichenfolgen-Variablen.

Verwenden Sie keine Wildcards (* oder ?).

BEISPIELE:

>NAME "F:OLDNAM" AS "F:NEWNAM"

Benennt die Datei OLDNAM auf der Diskette als NEWNAM.

PRO**NEW**

FORMAT: 1. NEW**Abkürzung:****Vergleiche:**

WIRKUNG:

NEW wird benutzt, um bestehende Programme oder den Reservespeicher zu löschen.

ANMERKUNGEN:

Wird es im PROgramm-Modus benutzt, löscht das NEW-Kommando alle Programme und Daten, die sich momentan im Speicher befinden. (Die mit Passwort geschützten Programme können nicht gelöscht werden.)

Wird das NEW-Kommando im Reserve-Modus benutzt, löscht es den bestehenden Reservespeicher.

Das NEW-Kommando ist im RUN-Modus nicht definiert und hat die Fehlermeldung ERROR 9 zur Folge.

BEISPIELE:

NEW

Löscht das Programm oder den Reservespeicher.

NEXT

Programm

FORMAT: 1. NEXT {Festvariable
Einfachgen., einstellige Variable}

Abkürzung: N.

Vergleiche: FOR

WIRKUNG:

NEXT wird benutzt, um das Ende einer Gruppe von Anweisungen zu kennzeichnen, die in einer FOR/NEXT-Schleife wiederholt werden sollen.

ANMERKUNGEN:

Der Gebrauch von NEXT ist unter FOR erklärt. Die numerische Variable in einer NEXT-Anweisung muß mit der numerischen Variablen des dazugehörigen FOR übereinstimmen.

BEISPIELE:

10: FOR I=1 TO 10

20: PRINT I

30: NEXT I

Druckt die Zahlen von 1 bis 10 aus, jedesmal wenn **ENTER** betätigt wird.

Programm

ON ERROR GOTO

FORMAT: 1. ON ERROR GOTO Zeilennummer
2. ON ERROR GOTO "Etikett"

Abkürzung: O.ERR.G.

Vergleiche: ERN, ERL

WIRKUNG:

Erstellt eine Fehler-Abfang-Routine.

ANMERKUNGEN:

Wurde eine Fehler-Abfang-Funktion ermöglicht, wird, wenn ein Fehler entdeckt wird, die Kontrolle der Fehler-Routine übertragen. Es erscheint keine Fehlermeldung.

Die Fehler-Abfang-Funktion wird durch das Kommando ON ERROR GOTO 0 außer Kraft gesetzt.

BEISPIELE:

See ERN

Programm

Abkürzungen: O. GOS.

Vergleiche: GOSUB, GOTO, ON...GOTO

ON...GOSUB wird benutzt, um eine der gegebenen Subroutinen abhängig vom Wert eines Kontrollausdrucks auszuführen.

Bei der Ausführung von ON...GOSUB wird der Ausdruck zwischen ON und GOSUB ausgewertet und auf einen ganzzahligen Wert reduziert. Ist der ganzzahlige Wert 1, so springt die Kontrolle auf die erste Zeilennummer der Liste, wie in einem normalen GOSUB. Ist der Wert 2 so springt die Kontrolle auf die zweite Zeilennummer in der Liste.

Wenn der Wert des Ausdrucks Null, negativ oder größer als die Zahl der in der Liste gegebenen Subroutinen ist, wird keine Subroutine ausgeführt und die Programmausführung führt weiter mit der nächsten Zeile.

Trennen Sie Zeilennummern oder Ausdrücke in der Liste durch Kommata.

In den Ausdrücken nach dem GOSUB dürfen keine Kommata vorkommen. Der PC-1280 kann nicht zwischen Kommata in Ausdrücken und Kommata zwischen Ausdrücken unterscheiden.

BEISPIELE:

```
10: INPUT A
20: ON A GOSUB 100,200,300
30: END
100: PRINT "FIRST"
110: RETURN
200: PRINT "SECOND"
210: RETURN
300: PRINT "THIRD"
310: RETURN
```

Die Eingabe einer 1 druckt 'FIRST' aus, eine 2 druckt 'SECOND' aus, eine 3 druckt 'THIRD' aus. Auf jede andere Eingabe wird nichts ausgedruckt.

ON...GOTO

Programm

FORMAT: 1. ON Ausdruck GOTO {Ausdruck} {"Etikett"} ; {Ausdruck} {"Etikett"} ; ...

Abkürzungen: O. G

Vergleiche: GOSUB, GOTO, ON...GOSUB

WIRKUNG: ON...GOTO wird benutzt, um eine der gegebenen Subroutinen abhängig vom Wert eines Kontrollausdrucks auszuführen.

ANMERKUNGEN:

Bei der Ausführung von ON...GOTO wird der Ausdruck zwischen ON und GOTO ausgewertet und auf einen ganzzahligen Wert reduziert. Ist der ganzzahlige Wert 1, so springt die Kontrolle auf die erste Position der Liste. Ist der Wert 2 so springt die Kontrolle auf die zweite Position in der Liste; usw.

Wenn der Wert des Ausdrucks Null, negativ oder größer als die Zahl der in der gegebenen Liste ist, geht die Programmausführung weiter mit der nächsten Zeile.

Trennen Sie Zeilennummern oder Ausdrücke in der Liste durch Kommata.

In den Ausdrücken nach dem GOTO dürfen keine Kommata vorkommen. Der PC-1280 kann nicht zwischen Kommata in Ausdrücken und Kommata zwischen Ausdrücken unterscheiden.

BEISPIELE:

```
10: INPUT A
20: ON A GOTO 100,200,300
30: GOTO 900
100: PRINT "FIRST"
110: GOTO 900
200: PRINT "SECOND"
210: GOTO 900
300: PRINT "THIRD"
310: GOTO 900
900: END
```

Die Eingabe einer 1 druckt 'FIRST' aus, eine 2 druckt 'SECOND' aus, eine 3 druckt 'THIRD' aus. Auf jede andere Eingabe wird nichts ausgedruckt.

OPEN

Disketten RUN PRO Programm

FORMAT: 1. OPEN "d:Dateiname" FOR Modus AS # Zeilennummer

Abkürzung: OP.

Vergleiche: CLOSE

WIRKUNG:

Öffnet eine Datei, bestimmt mit dem "d:Dateinamen" zur Anwendung der "# Dateinummer". Folgende Ein- bzw. Ausgaben in die Datei werden begleitet von der Beziehung zu der "# Dateinummer".

ANMERKUNGEN:

Die Dateinummer muß zwischen #2 bis #7 für Taschen-Disketten basierte Dateien liegen und von #20 bis #25 für RAM basierte Dateien.

Wenn kein Gerät bestimmt wurde, wird ein Bandgerät angenommen. RAM-Dateien werden mit F: und Taschen-Disketten basierte mit X: bezeichnet.

"Mode" bestimmt die Zugangsmethode in die Datei wie folgt:

INPUT bestimmt die folgende Eingabe von einer existierenden Datei aus.

OUTPUT bestimmt die folgende Ausgabe an ein Gerät oder Datei.

APPEND bestimmt Anfügungen an eine folgende Datei.

Wird mit einem bestehenden Dateinamen OUTPUT bestimmt, wird die Datei gelöscht, bevor die neue geschaffen wurde.

Ein Fehler tritt auf, wenn im APPEND- oder INPUT-Modus die bestimmte Datei nicht existiert.

Ein Fehler tritt auf, wenn die bestimmte Datei im APPEND- oder INPUT-Modus das Attribut P hat (Schreib-Schutz: siehe SET-Kommando).

Ein Fehler tritt bei dem Versuch auf, eine bereits eröffnete Datei mit der Dateinummer zu öffnen.

Eine einzelne Datei kann nicht gleichzeitig mit verschiedenen Dateinummern eröffnet werden.

Wildcards (* und ?) können nicht verwendet werden.

Ein FEHLER tritt auf, wenn das bestimmte Gerät nicht aufgestellt und angeschlossen ist.

Hat der Dateiname eine Erweiterung, so muß sie auch dabei sein.

BEISPIELE:

```

10: OPEN "X:DATA"FOR OUTPUT AS #2
20: FOR J=1 TO 5
30: PRINT #2,J
40: NEXT J
50: CLOSE #2
60: OPEN "X:DATA" FOR INPUT AS #2
70: IF EOF (2) THEN 110
80: INPUT #2,J
90: PRINT J
100: GOTO 70
110: REM THE END OF THE DATA FILE HAS BEEN REACHED
120: CLOSE #2
130: END

```

[10] Bringt eine neue Datei auf die Diskette
 [30] Schreibt die Ziffern 1 bis 5 in Folge in die Datei.
 [60] Eröffnet erneut die Datei.
 [70] Prüft die Datei auf das Ende hin.
 [80] Liest die Nummern 1 bis 5 ein und druckt sie aus.

PASS

RUN PRO

FORMAT: 1. PASS "Zeichenfolge"**Abkürzung:** PA.**Vergleiche:** CSAVE, SAVE, CLOAD, LOAD, NEW**WIRKUNG:**

Mit dem PASS-Kommando können Passwörter gesetzt und gelöscht werden.

ANMERKUNGEN:

Passwörter werden benutzt, um Programme vor Einsichtnahme oder Modifikation durch andere zu schützen. Die sieben Zeichen müssen Buchstaben des Alphabets oder folgende Sonderzeichen sein: ! # \$ % & () * + - / , . : ; < = > ? @ $\sqrt{\quad}$ π ^

Wurde das PASS-Kommando gegeben, so sind die Programme im Speicher geschützt. Ein Passwort-geschütztes Programm kann im Speicher nicht überprüft oder modifiziert werden. Es kann nicht auf Band oder Diskette abgespeichert oder mit LIST oder LLIST ausgegeben werden, weiterhin können weder Programmzeilen hinzugefügt noch gelöscht werden. Falls mehrere Programme im Speicher enthalten sind und das PASS-Kommando gegeben wird, sind alle im Speicher enthaltenen Programme geschützt. Die einzige Möglichkeit, diesen Schutz auszuschalten, ist ein weiteres PASS-Kommando mit dem gleiche Passwort einzugeben.

Wenn Sie ein Passwort mit mehr als sieben Zeichen eingeben, werden nur die ersten sieben Zeichen als gültig erkannt und zum Setzen und Aufheben des Schutzes benutzt.

Geben Sie **ENTER** gleich nach dem Passwort ein.

Sie erhalten eine Fehlermeldung, wenn Sie nach der Eingabe des Passwortes weitere Zeichen oder Symbole eingeben. In diesem Fall kann das Passwort nicht annulliert werden.

(Beispiel) PASS"ABCDEFGH":A = 123 **ENTER** → ERROR 1

PASS und RAM-Karten

Bei MEM\$="1" können Passwörter mit RAM-Karten verwendet werden.

Ist eine RAM-Karte eingelegt und das Passwort bestimmt, und wird die RAM-Karte dann herausgenommen, verbleiben die Inhalte der RAM-Karte Passwort-geschützt.

Sind Programme im eingebauten Speicher Passwort-geschützt und es wird eine neue RAM-Karte eingelegt, werden die Inhalte der RAM-Karte nicht aufgefunden (sind auf der eingeführten RAM-Karte Passwort-geschützte Programme, so ist dieses Passwort gültig).

BEISPIELE:

PASS "SECRET" **ENTER**

Hierdurch wird das Passwort 'SECRET' für alle Programme im Speicher geschaffen.

PAUSE

1. PRINT ... Programm

- FORMAT:**
1. PAUSE Ausdruck[:]
 2. PAUSE Ausdruck, Ausdruck, ..., Ausdruck
 3. PAUSE Ausdruck; Ausdruck; ...; Ausdruck[:]
 4. PAUSE USING Format Zeichenfolge; Ausdruckliste

Abkürzung: PAU.

Vergleiche: LPRINT, PRINT, CURSOR, USING, WAIT

WIRKUNG:

PAUSE wird benutzt, um eine Information auf dem Display kurz auszudrucken.

ANMERKUNGEN:

PAUSE wird benutzt, um kurze Information, Ergebnisse von Berechnungen usw. anzuzeigen. Die Ausführung von PAUSE ist die gleiche wie die von PRINT, mit der Ausnahme, daß bei PAUSE der PC-1280 eine Pause von etwa 0,85 Sekunden macht, bevor er mit der Programmausführung fortfährt, ohne auf die **ENTER**-Taste oder den WAIT-Intervall zu warten.

Zur Beschreibung der Formate 1 bis 3 lesen Sie die Beschreibung des PRINT-Kommandos.

Die USING-Klausel in Format 4 wird unter USING beschrieben.

BEISPIELE:

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

	<u>Display</u>	
20: PAUSE A		10.
30: PAUSE X\$	ABCDEF	
40: PAUSE X\$,Y\$,A,B	ABCDEF XYZ	
		10. 20.
50: PAUSE Y\$;X\$;	XYZABCDEF	
60: PAUSE A*B	XYZABCDEF200.	

PRINT

Programm

FORMAT:

1. PRINT Ausdruck[;]
2. PRINT Ausdruck, Ausdruck, ..., Ausdruck
3. PRINT Ausdruck; Ausdruck; ...; Ausdruck[;]
4. PRINT

Abkürzung: P.

Vergleiche: LPRINT, PAUSE, CURSOR, USING, WAIT

WIRKUNG:

PRINT weist Information am Bildschirm aus.

ANMERKUNGEN:

Mit PRINT werden Texte, Ergebnisse usw. auf dem Display ausgewiesen.

Format 1 gibt einen einzelnen Wert aus. Wenn es sich dabei um einen numerischen Ausdruck handelt, so wird der Wert von der rechten Seite des Bildschirms angezeigt. Handelt es sich um einen Zeichenfolgen-Ausdruck, beginnt die Anzeige ganz links.

Zum Beispiel:

10: PRINT "ABCD"		ABCD	
20: PRINT 123			123.

Beachten Sie aber, daß wenn das CURSOR-Kommando benutzt wurde, beginnt die Anzeige an der bezeichneten Stelle.

In Format 2 bestimmen Sie zur Anzeige zwischen zwei und 4 Elementen. Das Display ist aufgeteilt in vier Gruppen mit 12 Spalten (oder 2x24 im Modus mit doppelter Genauigkeit). Die Werte werden in Folge vom ersten bestimmten Wert aus angezeigt. Auch hier werden innerhalb eines Blocks von 12 Spalten (oder 24 im Modus mit doppelter Genauigkeit) numerische Werte vom rechten und Zeichenfolgen vom linken Rand wie folgt angezeigt:

```

10: A=123: B#=5#/9: C$="ABCD"
20: CLS: PRINT "A=",A
30: CLS: PRINT A,C$,B#
40: CLS: PRINT A,B#,C$

```

RUN **ENTER**

ENTER

ENTER

12er SPALTE

12er SPALTE

A=

123.

123. ABCD

5.555555555555555555D-01

123. 5.555555555

55555555D-01 ABCD

Ist im Format 2 ein einzelner Wert mehr als 12 Stellen lang (wenn der Dezimalbruch in Exponentialanzeige 7 Stellen oder mehr hat) werden die niedrigstwertigen Stellen abgeschnitten.

Wenn die Zeichenfolge über die 12 Spalten hinausgeht, werden nur die ersten 12 Zeichen (von links) angezeigt.

Wenn im Format 2 ein Wert mit doppelter Genauigkeit 24 Stellen überschreitet (wenn der Dezimalbruch in der Exponentialanzeige 19 oder mehr Anschläge hat), werden die niedrigstwertigen Stellen abgeschnitten.

Ein Fehler tritt auf, wenn die Gesamtanzahl der Anschläge oder Zeichen die Bildschirmgrenze von 48 Spalten überschreitet:

PRINT A,B#,C\$,"EFG"

In Format 3 wird der bestimmte Wert fortlaufend von der linken Displayseite angezeigt.

```

10: A=123: B#=5#/9: C$="ABCD"
20: PRINT "A=";A
30: PRINT "EFGH";B#;C$;A

```

>RUN **ENTER**

A=123.

ENTER

EFGH 5.5555555555555555555555555555
6D-01ABCD123.

KOMMANDOS

Wurde aber die Startposition des Displays mit dem Format 3 (mit Strichpunkt am Zeilenende) oder dem CURSOR-Kommando vorgegeben, beginnt die Anzeige an dieser Stelle.

Wenn der eingegebene Wert in Format 3 das Maximum von 48 Spalten überschreitet, wird der überschüssige Teil nicht angezeigt. Hat der anzuzeigende Ausdruck mehr als 48 Spalten, kommt eine Fehlermeldung (ERROR 6), wenn die 48. Spalte mitten in einem numerischen Wert liegt.

Endet die Druckzeile mit einem Komma, wird der angegebene Wert von der linken Displayseite aus angezeigt. Die Spalte, die dem Ende dieses angezeigten Wertes folgt, wird bestimmt als die Anzeige-Startposition für Display-Anweisungen, wie solche für das nächste PRINT-Kommando

```

10: A=123: B=45
20: PRINT "123*45=";
30: C= A*B
40: PRINT C

```

RUN **ENTER**

123*45=

ENTER

123*45=5535.

Wird Format 4 auf eine Zeile verwendet, auf der mittels GPRINT Grafiken angezeigt werden, werden die Grafiken gelöscht.

BEISPIELE:

```
10: WAIT 20
20: FOR A=0 TO 9
30: PRINT A;
40: NEXT A
50: WAIT
60: PRINT
```

Zeigt das Ergebnis von Zeile 30 ohne Stop an.

Band RUN PRO Programm**PRINT#**

FORMAT: 1. PRINT# Variable
 2. PRINT# Variable, Variable, ..., Variable
 3. PRINT# "Dateiname"; Variable [, Variable, ... Variable]

Abkürzung: P.#

Vergleiche: INPUT#

WIRKUNG:

Schreibt Werte in eine sequentielle Datei auf Band oder Diskette.

ANMERKUNGEN:

Folgende Variablentypen können für Variablennamen benutzt werden.

- (1) Festpunktvariablen: A, B, X, A(26), C*, A(10)*, etc.
- (2) Einfache Variablen: AA, B2, XY\$, etc.
- (3) Feldvariablen: B(*), CD(*), N\$(*), etc.

1) Sichern des Inhalts fester Variablen

Der Inhalt fester Variablen kann durch Angabe des Namens der entsprechenden Variablen (getrennt durch Kommata) in der PRINT-Anweisung gesichert werden.

```
PRINT# "DATA 1";A,B,X,Y
```

Diese Anweisung speichert die Inhalte der Variablen A,B,X und Y in einer Datei mit Namen "DATA 1".

Wenn Sie den Inhalt der spezifizierten Festpunktvariable und aller darauf folgenden speichern wollen, so geben Sie diesen Variablennamen gefolgt von einem Stern (*) ein.

```
PRINT# "D-2";D*
```

Diese Anweisung speichert die Inhalte der Variablen D bis Z (und der erweiterten Variablen A (27) und weiter, falls definiert) in die Datei namens "D-2".

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

Diese Anweisung speichert die Inhalte der Festpunktvariablen E und X\$ und der erweiterten Variablen A(30) und aller restlichen Variablen ohne Dateiname. Die indizierten Festpunktvariablen-Namen A(1) bis A(26) können in der PRINT#-Anweisung in gleicher Weise angegeben werden wie A bis Z (oder A\$ bis Z\$). Wenn aber das Feld A bereits durch eine DIM-Anweisung definiert wurde, kann A() nicht für die Definierung indizierter Festpunktvariablen benutzt werden.

2) Sichern der Inhalte von einfachen (Zwei-Zeichen Variable) Variablen

Die Inhalte von einfachen Variablen können gespeichert werden, indem die entsprechenden Variablennamen angegeben werden.

PRINT# "DM-1"; AB,Y1,XY\$

Diese Anweisung speichert die Inhalte der einfachen Variablen AB, Y1 und XY\$ in einer Datei namens "DM-1".

3) Speichern der Inhalte von Feldvariablen

Die Inhalte aller Variablen eines bestimmte Feldes können gesichert werden, indem der mit einem umklammerten Stern (*) indizierte Name des Feldes angegeben wird.

PRINT# "DS-2";X(*),Y\$(*)

Diese Anweisung speichert die Inhalte aller Elemente (X(0),X(1),...) des Feldes X und aller Elemente (Y\$(0),Y\$(1),...) des Feldes Y\$ in der Datei namens "DS-2".

Es ist nicht möglich, die Inhalte eines oder einiger Elemente eines Feldes gesondert zu speichern. Während Festpunktvariablen oder indizierte Festpunktvariablen zulassen, daß nur ein spezifizierter Teil von ihnen gespeichert wird, ist dies mit einem durch die DIM-Anweisung definierten Feld (wie A) nicht möglich.

Wird die PRINT#-Anweisung ohne Variablennamen eingegeben, erhalten Sie eine Fehlermeldung (ERROR 3).

ACHTUNG

Die Plätze für erweiterte Variable wie A(27) und weiter, für einfache Variablen und/oder Feldvariablen müssen vor Ausführung der PRINT#-Anweisung im Programm/Datenspeicherbereich bereitgestellt werden. Sonst ist die Ausführung der PRINT#-Anweisung für nicht definierte Variablen nicht möglich und Sie erhalten eine Fehlermeldung.

PRINT#

Diskette RUN PRO Programm

FORMAT: PRINT #Dateinummer, { Ausdruck }
 { Zeichenfolge }
 [{ ; } { Ausdruck } { Zeichenfolge } ... { ; } [{ Ausdruck } { Zeichenfolge }]]

Abkürzung: P.#

Vergleiche: OPEN, INPUT#

WIRKUNG:

Schreibt die Inhalte der bestimmten Variable auf die Diskette.

ANMERKUNGEN:

Dateinummer #2 bis #7 für Taschen-Disketten basierte Dateien und #20 bis #25 für RAM-Disketten basierte Dateien.

- Die PRINT#-Anweisung ist nur bei Dateien wirksam, für die OUTPUT oder APPEND in dem OPEN-Befehl angegeben wurden.
- Die Filenummer muß die beim OPEN-Befehl gegebene sein.
- Wenn eine ein- oder zweidimensionale Feldvariable in Form eines "Feldnamens (*)" bestimmt wurde, so wird das gesamte Feld auf die Diskette geschrieben. Wenn die betreffenden Elemente des Feldes bestimmt sind, so müssen Sie in der Form von "B(7)", "C\$(5,6)" usw. angegeben werden.
- Ist das gesamte Feld bestimmt, so werden seine Elemente in der Reihenfolge C\$(0,0), C\$(0,1), C\$(0,2),...C\$(1,0),...C\$(5,5) geschrieben.
- Wenn ein Zeichen oder eine Zeichenvariable verwendet wird, die sich vom Gesamtfeld unterscheidet, so muß sie, wie im folgenden Beispiel gezeigt, mit einem Komma (,) oder einem Strichpunkt (;) verbunden werden.
 PRINT #2,"ABC"
 PRINT #2,A\$

Wird PRINT #2,"ABC",A\$ ausgeführt, so wird kein Daten-Abgrenzungs-Code geschrieben und "ABC" und A\$ können nicht getrennt werden.

- Die maximale Anzahl der Dateien (einschließlich der Programmdateien), die auf einer Taschen-Diskette untergebracht werden können, ist 48.
- Die maximale Anzahl von Dateien, die auf eine RAM-Karte geschrieben werden können, liegt je nach Kapazität der RAM-Karte zwischen 4 und 64 Dateien.

- Ein numerischer Wert wird in der Form aufgenommen, daß das Vorzeichen (Leerschritt, wenn es positiv ist), die numerische Zeichenfolge und der Freiraum in dieser Reihenfolge erscheinen.

- Das Aufnahmeformat ist das folgende:

- ① CR(&0D) und LF(&0A) erscheint, wenn den Daten kein Komma oder Strichpunkt folgt

BEISPIELE:

PRINT #2, -1.2 | - | 1 | . | 2 | | CR | LF |

PRINT #2, "ABC" | A | B | C | CR | LF |

- ② Folgt ein Komma den Daten, so werden 20 Bytes besetzt. Ein numerischer Wert ist rechtsbündig und eine Zeichenfolge linksbündig.

BEISPIELE:

PRINT #2, - 1.2,3

| | | ~ | - | 1 | . | 2 | | | 3 | . | | CR | LF |

← 20 Bytes →

PRINT #2, "ABC", "DEF"

A	B	C	~	D	E	F	CR	LF
---	---	---	---	---	---	---	----	----

← 20 Bytes →

Ist eine Zeichenfolge länger als 20 Bytes, so wird der darüber hinausgehende Teil in den Bereich für die nächsten 20 Bytes geschrieben. Die absolute Grenzgröße ist 80 Bytes.

- ③ Folgt ein Strichpunkt den Daten, so wird ohne Freiräume gespeichert.

BEISPIELE:

PRINT #2, - 1.2;3

| - | 1 | . | 2 | | 3 | . | | CR | LF |

PRINT #2, "ABC"; "DEF"

A	B	C	D	E	F	CR	LF
---	---	---	---	---	---	----	----

In diesem Fall können die Zeichenfolgen "ABC" und "DEF" nicht getrennt ausgegeben werden.

RADIAN

RUN PRO Programm

FORMAT: 1. RADIAN

Abkürzung: RAD

Vergleiche: DEGREE, GRAD

WIRKUNG:

RADIAN wird benutzt, um die Form von Winkelwerten auf Neu-Grad umzuschalten.

ANMERKUNGEN:

Der PC-1280 kann 3 Formen der Winkeldarstellung verarbeiten: Alt-Grad (dezi-), Radial-Werte und Neu-Grad. Diese Formen werden für die Darstellung der Werte der Argumente von den SINus-, COSinus- und TANGens-Funktionen und der Ergebnisse der Umkehrfunktionen ArcusSiNus, ArcusCoSinus und ArcusTaNgens benutzt.

Die RADIAN-Funktion wird benutzt, um Winkelwerte auf Radialwerte umzuformen, bis DEGREE oder GRAD benutzt wird. Die Radialform gibt die Winkel als Bogenmaß in abhängigkeit vom Radius wieder, der z.B. 360° sind 2π , da der Einheitskreis der Umfang 2 mal π mal Radius hat.

BEISPIELE:

10: RADIAN

20: X= ASN 1

30: PRINT X

X hat den Wert 1,570796327 oder $\pi/2$, der Arcsinus von 1.

RUN PRO Programm

RANDOM

FORMAT: 1. RANDOM

Abkürzung: RA.

Vergleiche: RND

WIRKUNG:

RANDOM wird benutzt, um eine neue Startzahl für den Zufallsgenerator zu setzen.

ANMERKUNGEN:

Wenn mit der RND-Funktion Zufallszahlen erzeugt werden, so startet der PC-1280 bei einer vorgegebenen Startzahl. Der RANDOM-Befehl setzt diese Startzahl auf einen neuen, zufällig gewählten Wert.

Die Startzahl ist jedes mal die gleiche, wenn der PC-1280 eingeschaltet wird, so daß sich die Sequenz der Zufallszahlen mit RND jedesmal wiederholt, es sei denn, die Startzahl wird gewechselt. Diese Eigenschaft ist bei der Entwicklung eines Programms sehr wichtig, denn dies bedeutet, daß das Verhalten des Programms immer gleich sein muß, jedesmal, wenn es abläuft, obwohl es eine RND-Funktion enthält. Wenn Sie wirklich zufällige Zahlen haben wollen, kann mit der RANDOM-Anweisung die Startzahl selbst zufällig ermittelt werden.

BEISPIELE:

10: RANDOM

20: X=RND 10

Bei Start in Zeile 20 beruht der Wert von X auf einer Standard-Startzahl. Bei Start in Zeile 10 wird eine neue Basis ermittelt.

READ

Programm

FORMAT: 1. READ Variable
2. READ Variable, Variable, ..., Variable

Abkürzung: REA.

Vergleiche: DATA, RESTORE

WIRKUNG:

READ wird gebraucht, um Werte aus der DATA-Anweisung herauszulesen und sie Variablen zuzuweisen.

ANMERKUNGEN:

Wenn einem Feld Basiswerte zugewiesen werden sollen, so ist es sinnvoll, diese Werte in einer DATA-Anweisung unterzubringen und sie von dort mit einer READ-Anweisung in einer FOR...NEXT-Schleife in das Feld zu übertragen. Die folgenden READ-Befehle benutzen die folgenden Werte in der Reihenfolge ihrer Erscheinung im Programm, unabhängig davon, wieviele Werte in jeder DATA-Anweisung eingesetzt werden.

Wenn nötig, können dieselben Werte in einer DATA-Anweisung ein zweites mal mit Hilfe der RESTORE-Anweisung gelesen werden.

BEISPIELE:

```
10: DIM B(10)
20: WAIT 32
30: FOR I=1 TO 10
40: READ B(I)
50: PRINT B(I)*2;
60: NEXT I
70: DATA 10,20,30,40,50,60
80: DATA 70,80,90,100
90: END
```

[10] Dimensionierung eines Feldes.

[40] Lädt die Werte aus der DATA-Anweisung in B() ein; B(1) ist 10, B(2) ist 20, B(3) ist 30 usw.

REM

Programm

FORMAT: 1. REM Kommentar

Abkürzung:

Vergleiche:

WIRKUNG:

REM wird benutzt, um Kommentare in ein Programm einzufügen.

ANMERKUNGEN:

Öfters ist es sinnvoll, in ein Programm erläuternde Kommentare einzufügen. Es kann sich hierbei um Titel, Autorennamen, Daten der letzten Änderungen, Anwendungshinweise, Erinnerungshinweise für Algorithmen usw. handeln. Solche Kommentare können mit Hilfe der REM-Anweisung eingefügt werden.

Die REM-Anweisung hat keinen Einfluß auf den Programmablauf und kann überall im Programm eingefügt werden. Alles, was in dieser Zeile nach dem REM-Befehl kommt, wird als Kommentar behandelt.

BEISPIELE:

10: REM THIS LINE HAS NO EFFECT

RENUM

PRO

FORMAT: 1. RENUM [neue Zeilennummer][,[alte Zeilennummer],[Inkrement]]

Abkürzung: REN.

Vergleiche: DELETE, LIST

WIRKUNG:

Führt die Neunummerierung der Zeilen eines Programms aus.

ANMERKUNGEN:

Die Zeilennummern werden mit dem vorgegebenen Inkrement von "alte Zeile" auf "neue Zeile" geändert. Wird "neue Zeile" nicht angegeben, werden die Zeilen neu nummeriert, beginnend mit 10 in Inkrementen mit 10. RENUM aktualisiert die Bezugszeilennummern in den Anweisungen GOTO, ON...GOTO, GOSUB, ON...GOSUB, RESTORE und (IF)...THEN.

Ist eine Zeilennummer größer als 65279, oder existiert eine angegebene Zeilennummer nicht, erhalten Sie die Fehlermeldung ERROR 4. Wenn Sie die Reihenfolge der Ausführung ändern, erhalten Sie die Fehlermeldung ERROR 1. RENUM wird nur in den BASIC- und PRO-Modi verwendet (andere Modi: Fehlermeldung ERROR 9). Wenn Sie ein Passwort benutzt haben, wird das Kommando ignoriert und das Aufforderungszeichen erscheint wieder.

Wenn mehrere Programme mit MERGE im Speicher gemischt wurden, so führt RENUM nur die Neunummerierung des letzten gemischten Programms aus. Falls die niedrigste Zeilennummer des neu nummerierten Programms größer ist als die höchste Zeilennummer des zuletzt gemischten Programms, werden die beiden Programme von da an als ein einzelnes Programm angesehen.

Enthält eine Zeilennummer in einer GOTO-Anweisung nicht numerische Zeichen, erscheint die Fehlermeldung ERROR 9.

Die Fehlermeldung ERROR 3 erscheint, wenn die Neunummerierung die Zeilenlänge auf über 80 Bytes erweitert.

Wird in einer Zeile mit mehreren Anweisungen RESTORE, gefolgt von einem Doppelpunkt, verwendet (z.B. 20 RESTORE: READ A), so bewirkt jeder Versuch, dieses Programm mit RENUM neu zu nummerieren, die Fehlermeldung ERROR 9.

Wenn auf der Anzeige das Zeichen "*" erscheint, so drücken Sie die **BREAK**-Taste, um die Neunummerierung zu unterbrechen. Wenn "**" erscheint, bedeutet das, daß die Unterbrechung nicht möglich ist. Die Fehlererzeugung oder der Gebrauch der **BREAK**-Taste läßt das Programm unverändert.

RENUM-FEHLERLISTE

Fehlermeldung	Beschreibung
ERROR 1	Der RENUM-Befehl enthält einen Syntaxfehler.
ERROR 1 IN Zeilennummer	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.
ERROR 6	Die Speicherkapazität reicht nicht zur Durchführung des RENUM-Befehls oder wird während der Neunummerierung reduziert.
ERROR 9	Es wurde 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 niedriger 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).

BEISPIELE:

```

10: INPUT "CONTINUE";A$
20: IF A$="YES" THEN 10
30: IF A$="NO" THEN 60
40: PRINT "ENTER YES OR NO PLEASE! "
50: GOTO 10
60: END

```

RENUM 100,10,5

```

100: INPUT "CONTINUE";A$
105: IF A$="YES" THEN 100
110: IF A$="NO" THEN 125
115: PRINT "ENTER YES OR NO PLEASE! "
120: GOTO 100
125: END

```

RESTORE

Programm

FORMAT: 1. RESTORE
2. RESTORE Ausdruck

Abkürzung: RES.

Vergleiche: DATA, READ

WIRKUNG:

RESTORE wird benutzt, um Werte in einer DATA-Anweisung mehrmals zu lesen oder die Reihenfolge, in der diese Werte gelesen werden sollen, zu ändern.

ANMERKUNGEN:

Beim normalen Gebrauch von READ beginnt der PC-1280 das Lesen beim ersten Wert in einer DATA-Anweisung und geht sequentiell durch alle übrigen Werte. Die erste Form der RESTORE-Anweisung setzt den Zeiger auf den ersten Wert der DATA-Anweisung zurück, damit er noch einmal gelesen werden kann. Die zweite Form der RESTORE-Anweisung setzt den Zeiger auf den ersten Wert der ersten DATA-Anweisung zurück, deren Zeilennummer größer als der Wert des Ausdrucks ist.

Wird RESTORE, gefolgt von einem Doppelpunkt (z.B. 20 RESTORE: READ A) in einer Zeile mit mehreren Anweisungen benutzt, so bewirkt jeder Versuch, das Programm mit RENUM neu zu nummerieren, die Fehlermeldung ERROR 9. Vergewissern Sie sich, daß alle RESTORE-Anweisungen auf separate Zeilen geschrieben werden, oder in Format 2; gefolgt von einem Ausdruck (z.B. RESTORE X*Y).

BEISPIELE:

```
10: DIM B(10)
20: WAIT 32
30: FOR I=1 TO 10
40: RESTORE
50: READ B(1)
60: PRINT B(I)*I;
70: NEXT I
80: DATA 20
90: END
```

[10] Dimensionierung eines Feldes

[50] Weist jedem Element von B() den Wert von 20 zu.

RETURN

Programm

FORMAT: 1. RETURN

Abkürzung: RE.

Vergleiche: GOSUB, ON...GOSUB

WIRKUNG:

RETURN wird am Ende einer Subroutine benutzt, um zur Anweisung nach dem ursprünglichen GOSUB-Befehl zurückzuspringen.

ANMERKUNGEN:

Eine Subroutine kann mehr als eine RETURN-Anweisung enthalten, aber die erste, die ausgeführt wird, beendet die Ausführung der Subroutine. Die nächste Anweisung, die ausgeführt wird, ist dann die Anweisung nach dem GOSUB oder ON...GOSUB-Befehl, die die Subroutine aufruft. Wird ein RETURN ohne GOSUB ausgeführt, erhalten Sie die Fehlermeldung ERROR 5.

BEISPIELE:

```
10: GOSUB 100
20: END
100: PRINT "HELLO"
200: RETURN
```

Das Programm druckt bei der Ausführung das Wort HELLO einmal aus.

RIGHT\$

RUN PRO Programm

FORMAT: 1. RIGHT\$(X\$,N)
2. RIGHT\$("Zeichenfolge",N)

Abkürzung: RI.

Vergleiche: LEFT\$, MID\$

WIRKUNG:

Erstellt eine Zeichenfolge von N Zeichen in der Zeichenfolge X\$, beginnend bei rechten Ende der Zeichenfolge.

ANMERKUNGEN:

Der Wert N muß zwischen 0 und 80 liegen. Bruchteile werden abgerundet (abgeschnitten). Ist N kleiner als 1, wird eine Null-Zeichenfolge herausgenommen. Ist N größer als die Anzahl der Zeichen in X\$, wird die ganze Zeichenfolge herausgenommen.

BEISPIELE:

```
5: WAIT 32
10: XX$="SHARP COMPUTER"
20: FOR N=1 TO 14
30: LET SS$=RIGHT$(XX$,N)
40: PRINT SS$
50: NEXT N
```

```
RUN
R
ER
TER
UTER
PUTER
MPUTER
OMPUTER
COMPUTER
  COMPUTER
P COMPUTER
RP COMPUTER
ARP COMPUTER
HARP COMPUTER
SHARP COMPUTER
>
```

RUN PRO Programm**RND****FORMAT:** 1. RND numerischer Ausdruck**Abkürzung:** RN.**Vergleiche:** RANDOM**WIRKUNG:**

RND erzeugt Zufallszahlen.

ANMERKUNGEN:

Ist der Wert des Arguments kleiner als 1, aber größer als Null, so ist die Zufallszahl kleiner als 1 und größer oder gleich Null. Ist das Argument eine ganze Zahl, die größer oder gleich 1 ist, ist das Ergebnis eine Zufallszahl, die größer oder gleich 1 oder kleiner oder gleich dem Argument ist. Ist das Argument größer oder gleich 1 und keine ganze Zahl, ist das Ergebnis eine Zufallszahl, die größer oder gleich 1 und kleiner oder gleich der kleinsten ganzen Zahl ist, die wiederum größer als das Argument ist (in diesem Fall ändert sich die Erzeugung der Zufallszahl, je nach dem Wert des Dezimalteils des Arguments):

<u>Argument</u>	----- Ergebnis -----	
	<u>Untere Grenze</u>	<u>Obere Grenze</u>
.5	0<	<1
2	1	2
2.5	1	2

Normalerweise wird die gleiche Sequenz von Zufallszahlen erzeugt, da jedesmal die gleiche Basiszahl benutzt wird, wenn der PC-1280 eingeschaltet wird. Um diese Basiszahl zufällig zu ändern, brauchen Sie das RANDOM-Kommando.

RUN PRO Programm

Abkürzung: RS.

Vergleiche: LSET, LET

Speichert Daten, rechts beginnend, in einem Zeichenfolge-Variablen-Bereich.

Das RSET-Kommando (Rechts SET) speichert Zeichendaten in dem Zeichenfolge-Variablen-Bereich. Die eingegebenen Daten beginnen an der rechten Seite des Felds.

RSET B\$="ABC" speichert die Zeichenfolge "ABC" rechts im Feld B\$:

BS =

				A	B	C
--	--	--	--	---	---	---

Wenn die der Variablen zugeordnete Zeichenfolge länger ist als durch diese Variable eingegeben werden kann, werden nur diese Zeichen (beginnend links von der Zeichenfolge) gespeichert:

```
RSET B$="ABCDEFGHIJ"
```

wird gespeichert als:

B\$ $\rightarrow$

A	B	C	D	E	F	G
---	---	---	---	---	---	---

RUN**RUN**

FORMAT: 1. RUN
 2. RUN {Zeilennummer}
 "Etikett"

Abkürzung: R.

Vergleiche: GOTO, MERGE, ARUN

WIRKUNG:

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

ANMERKUNGEN:

Die erste Form des RUN-Kommandos startet ein Programm mit der Zeile, die die niedrigste Zeilennummer hat.

Die zweite Form des RUN-Kommandos startet ein Programm mit der angegebenen Zeilennummer.

Wenn mehrere Programm im Speicher enthalten sind, weil Programme mit MERGE gemischt wurden, führt das RUN-Kommando das zuletzt gemischte Programm aus, außer wenn im Programm und im Kommando ein Programmetikett benutzt wird.

Das RUN-Kommando unterscheidet sich von GOTO in 6 Punkten:

- 1) Der Wert des WAIT-Intervalles wird zurückgesetzt.
- 2) Wurde die Anzeige durch USING-Anweisungen formatiert, so wird sie gelöscht.
- 3) Variablen und Felder, mit Ausnahme der festen Variablen, werden gelöscht.
- 4) Der PRINT = LPRINT-Status wird zurückgestellt.
- 5) Der READ-Zeiger wird zurückgesetzt an die erste DATA-Anweisung.
- 6) Die Cursor-Spezifikationen werden gelöscht.

Die Ausführung eines Programms mit dem GOTO-Kommando ist die gleiche, wie die Ausführung mit der **DEF**-Taste. Alle drei Formen des Programmstarts löschen FOR/NEXT- und GOSUB-Gruppen.

BEISPIELE:

RUN 100

Beginnt die Ausführung des Programms ab Zeile 100.

Diskette RUN PRO Programm

SAVE

FORMAT: 1. SAVE "d:Dateiname" [,A]

Abkürzung: SA.

Vergleiche: LOAD

WIRKUNG:

Sendet das BASIC-Programm im Speicher des Rechners auf einer Taschen-Diskette oder RAM-Diskette.

ANMERKUNGEN:

Die SAVE-Anweisung benennt ein BASIC-Programm im eingebauten Speicher des PC-1280 und schreibt es dann auf Diskette (RAM- oder Taschen-Diskette).

Alle Dateien werden geschlossen, bevor die Datei gesichert wird.

Wird keine Erweiterung bestimmt, so wird .BAS angenommen. Die Erweiterung kann aus bis zu drei Zeichen bestehen.

Wurde eine A Option bestimmt, wird die Datei im ASCII-Format gesichert, sonst wird sie nämlich im komprimierten Binärformat gesichert. Beachten Sie, daß die Dateien im ASCII-Format nicht auf RAM-Disketten gesichert werden können.

Eine bestehende Datei wird gelöscht, wenn der gleiche Dateiname bestimmt wird, aber eine Fehlermeldung wird angezeigt, wenn die existierende Datei mit P (Schreibschutz) versehen ist, oder eine angesprochene RAM-Karte.

Der SAVE-Befehl wird ignoriert, wenn ein Programm im Rechner Passwort-geschützt wurde, oder wenn kein Programm im Speicher ist.

Die zuletzt gemischte Programmzeile wird gesichert, wenn mehrere Dateien mit dem MERGE-Kommando geladen wurden.

Verwenden Sie keine Zeichenfolgen-Variablen als Dateinamen.

Ein Fehler (ERROR) tritt auf, wenn das bestimmte Gerät nicht aufgestellt und angeschlossen ist.

BEISPIELE:

SAVE "X:PRO1", A

Sichert das Programm PRO1 als eine ASCII Datei auf der Taschen-Diskette.

SET**Diskette RUN PRO**

FORMAT: 1. SET "{X:}" Dateiname", {"P"}
 2. SET "F:", {"P"}
 {" "}

Abkürzung: SE.

Vergleiche: SAVE

WIRKUNG:

Weist zu oder entfernt Dateischutz.

ANMERKUNGEN:

Die Inhalte der Datei können bei angegebenem P weder versehentlich gelöscht noch erneuert werden. Format 2 ermöglicht, daß alle Dateien auf der RAM-Diskette gesichert werden. Hierdurch wird irrtümliches Lösches der Inhalte von der RAM-Diskette ausgeschlossen, wenn mit dem SET MEM-Kommando die Speicherkonfiguration geändert wird.

Wildcards (* oder ?) können zur Bestimmung von Dateinamen benutzt werden. Aber die Erweiterung darf nicht vergessen werden. Zum Beispiel muß die .BAS-Erweiterung bestimmt werden.

Zum Löschen des Schutzes, bestimmen Sie eine Leerstelle für das Attribut. Ist der Schutz erst mal entfernt, kann die Datei oder die RAM-Karte frei gelöscht werden.

Verwenden Sie als Dateinamen keine Zeichenfolge-Variablen.

Eine Fehlermeldung kommt, wenn das SET-Kommando bei einer geöffneten Datei benutzt wird.

Ein Fehler tritt auf, wenn das bestimmte Gerät nicht aufgestellt und angeschlossen ist.

Ein Fehler tritt auf, wenn die bestimmte Datei bereits mit dem P-Attribut geschützt ist.

BEISPIELE:**>SET "XPAYRUN.BAS", "P"**

Schützt das Programm PAYRUN.BAS auf der Taschen-Diskette vor Zuschreibung, Löschung oder Neubenennung.

SET MEM

PRO RUN

- FORMAT:**
1. SET MEM "2"
 2. SET MEM "1"
 3. SET MEM "B"
 4. SET MEM "D"

Abkürzung: SE.M.

Vergleiche: MEM, MEM\$, SET, INIT

WIRKUNG:

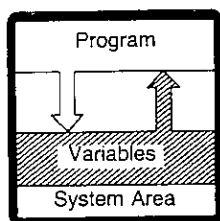
Strukturiert den eingebauten Speicher und die RAM-Karte.

ANMERKUNGEN:

Siehe Abschnitt über den Einsatz von RAM-Karte.

1. **SET MEM "2"** strukturiert NUR den eingebauten Speicher des PC-1280. Das zur Verfügung stehende Feld ist zur Benutzung durch Programme und vorge-speicherte Variablen aufgeteilt.

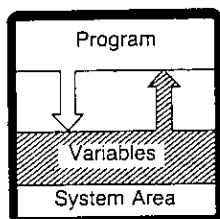
EINLIEGENDER SPEICHER



} MEM setzt die Bytesanzahl für diesen freien Speicher-bereich

2. **MEMSET "1"** strukturiert RAM-Karten im Schlitz auf der Rückseite des PC-1280.

RAM-KARTE



} MEM setzt die Bytesanzahl für diesen freien Speicher-bereich im RAM-Karten-Speicher.

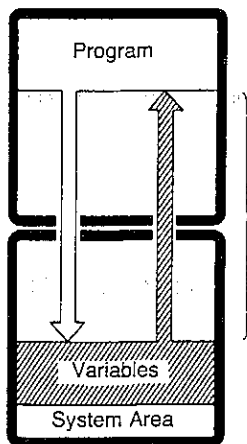
Verwenden Sie SET MEM "1" zur Benutzung der RAM-Karte unabhängig vom eingebauten Speicher (der mit SET MEM "2" strukturiert werden muß).

Führen Sie eine RAM-Karte die nicht mit dem INIT-Kommando initialisiert wurde, in den Schlitz ein und tippen Sie SET MEM "1" **ENTER**. Der PC-1280 antwortet folgendermaßen:

MEM\$ = "1"
RAM CARD CLEAR O.K. ?

Tippen Sie **Y** zum Löschen aller Inhalte auf der RAM-Karte, machen Sie sie zur Benutzung als ein Speichermodul bereit. Drücken Sie **N** oder schalten Sie den PC-1280 aus (OFF), um den Prozess abzubrechen.

3. SET MEM "B" strukturiert den eingebauten Speicher und den RAM- Kartenspeicher als ein zusammenhängender Speicherbereich. Dies ist bei besonders großen Programmen von Vorteil, oder wenn eine große Anzahl von Daten behandelt werden soll.



RAM CARD

MEM setzt die Bytesanzahl für diesen freien Speicherbereich im eingebauten und im RAM-Kartenspeicher.

EINLIEGENDER SPEICHER

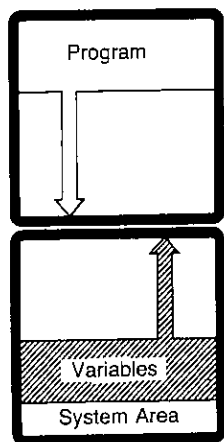
Wenn strukturiert mit SET MEM "B", werden der eingebaute und der RAM-Kartenspeicher beide verwendet als unentbehrlich für Programme und für die Eingabe von Variablen benutzt. Die RAM-Karte kann nicht herausgenommen werden.

Wird SET MEM "1" nach der Konfiguration des Speichers durch SET MEM "B" getippt, so werden Programme und Variable auf die RAM-Karte gebracht, aber nur, wenn zu ihrer Aufnahme genügend Platz auf der RAM-Karte zur Verfügung steht.

Das Tippen von SET MEM "2" ergibt keinen Effekt.

Beachten Sie, daß SET MEM "B" nur gilt, wenn MEM\$="1".

4. SET MEM "D" strukturiert den eingebauten Speicher für Variable und die RAM-Karte für Programme.



RAM CARD

MEM setzt die Bytesanzahl für diesen freien Speicherbereich im eingebauten und im RAM-Kartenspeicher.

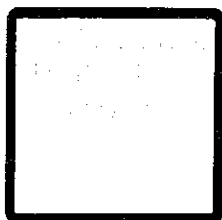
EINLIEGENDER SPEICHER

Die Programme dürfen nicht die zur Verfügung stehende Kapazität der RAM-Karte überschreiten, und die Variablen dürfen nicht die Kapazität des eingebauten Speichers überschreiten.

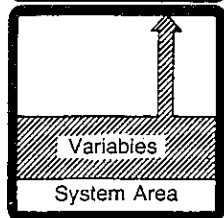
Zum Trennen der Programme von den Variablen benutzen Sie SET MEM "D".

SET MEM "D" ist gültig, wenn MEM\$="1".

5. **SET MEM "2"** gefolgt von dem Kommando **INIT "F:"** bewirkt, daß die RAM-Karte als RAM-Datei benutzt werden kann.



RAM CARD



EINGEBAUTER SPEICHER

Um die Inhalte der RAM-Karten vor unbeabsichtigtem Löschen zu schützen, verwenden Sie das SET-Kommando (SET "F:", "P"). Erst einmal mit dem SET-Kommando geschützt, tritt bei Versuch SET MEM "1" zu setzen eine Fehlermeldung (ERROR) auf.

ÄNDERUNG DER SPEICHERKONFIGURATION

		Neue Konfiguration			
		RAM-Karte MEM\$="1"	Einliegende RAM MEM\$="2"	RAM-Karte und einliegende RAM MEM\$="B"	RAM-Karte und einliegende RAM MEM\$="D"
Gegenwärtige Konfiguration	RAM-Karte MEM\$="1"		Tippen Sie SET MEM "2" ein	Tippen Sie SET MEM "B" ein	Tippen Sie SET MEM "D" ein
	Einliegende RAM MEM\$="2"	Tippen Sie SET MEM "1" ein		Nicht möglich, ohne zuvor die einliegende RAM-Karte zu löschen und sie erneut vom Anfang zu konfigurieren. Beim Versuch tritt eine Fehlermeldung auf.	
	RAM-Karte und einliegende RAM MEM\$="B"	Tippen Sie SET MEM "1" ein. Wird der gesamte Bereich für Programme benutzt, sind Variablen und Systeme Größer als der zur Verfügung stehende Speicher der	Tippen Sie SET MEM "2" ein. Es ist nicht möglich, präsen- te Speicher-In- halte zurückzu- halten. Bei Ver- such tritt die Feh- lermeldung ERROR auf.		Tippen Sie SET MEM "D" ein. Geht ein bestehen- der Programmbe- reich über die Ka- pazität der RAM- Karte hinaus, oder wenn ein bestehen- der Variablenbe- reich die Kapazität des internen Spei- chers übersteigt, tritt eine Fehlermel- dung ERROR auf.
	RAM-Karte und einliegende RAM MEM\$="D"	RAM-Karte, tritt eine Fehlermel- dung auf		Tippen Sie SET MEM "B" ein	

KOMMANDOS

STOP

Programm

FORMAT: 1. STOP**Abkürzung:** S.**Vergleiche:** END, CONT**WIRKUNG:**

STOP wird benutzt, um die Ausführung eines Programms für Diagnosezwecke zu stoppen.

ANMERKUNGEN:

Wenn STOP während des Programmablaufs angetroffen wird, stoppt der PC-1280 die Ausführung und es erscheint eine Meldung, wie "BREAK IN 200", wobei 200 die Nummer der Zeile ist, die STOP enthält. STOP wird während der Entwicklung eines Programms benutzt, um den Ablauf des Programms zu kontrollieren oder den Zustand der Variablen zu überprüfen. Sie können die Ausführung mit dem CONT-Kommando fortsetzen.

BEISPIELE:

10: STOP

Läßt die Meldung "BREAK IN 10" auf der Anzeige erscheinen.

STR\$

RUN PRO Programm

FORMAT: 1. STR\$

Abkürzung: STR.

Vergleiche: VAL

WIRKUNG:

Wandelt numerische Daten in Zeichenfolgedaten um.

ANMERKUNGEN:

Die STR\$-Funktion wandelt eine ganze Zahl in eine Zeichenfolge um. Die Zeichenfolge besteht aus den gleichen Ziffern, wie die ursprüngliche Zahl, wird aber als eine Zeichenfolge von SHARP ASCII-Zeichen in der weiteren Verarbeitung behandelt. Die STR\$-Funktion hat den entgegengesetzten Effekt von der VAL-Funktion.

Sind die numerischen Daten negativ, so steht vor der Zeichenfolge ein Minuszeichen (-). Ist der numerische Wert zu groß, um in einer Zeichenfolge-Variablen enthalten zu sein, wird er in Gleichpunktnotation angegeben.

BEISPIELE:

```
110: N=N*3
120: A$= STR$(N)
130: B$= LEFT$(A$,1)
140: M= VAL(B$)
```

- [110] Das Programm führt einige Berechnungen auf der numerischen Variable N aus.
- [120] Die numerische Variable N wird in die Zeichenfolge-Variable A\$ umgewandelt. Zeichenfolgen-Variablen können viel leichter manipuliert werden als numerische Zeichen. In diesem Beispiel nehmen wir an, daß die erste Ziffer der Zahl benötigt wird. Vielleicht ist sie ein Code für einen weiteren Prozess. Da wir die Zahl in eine Zeichenfolge umgewandelt haben, können wir irgendeines der Zeichenfolgen-Manipulationskommandos wie LEFT\$, RIGHT\$ und MID\$ benutzen.
- [130] Speichert nur die erste Ziffer der Zahl oder das Zeichen, wie es jetzt vom Programm behandelt wird, in der Zeichenfolgen-Variablen B\$.
- [140] Die Einzelziffer wird wieder in eine numerische Variable umgewandelt, so daß sie vom Programm als eine Zahl behandelt werden kann.

TROFF

RUN PRO Programm

FORMAT: 1. TROFF**Abkürzung:** TROF.**Vergleiche:** TRON**WIRKUNG:**

TROFF wird benutzt, um den Trace-Modus (TRON) zu löschen.

ANMERKUNGEN:

Die Ausführung von TROFF hat zur Folge, daß das Programm wieder auf normale Weise ausgeführt wird.

BEISPIELE:

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

Dieses Programm gibt bei seiner Ausführung die Zeilennummern 10, 20, 30, 30, 30 und 40 an, wenn Sie die -Taste drücken.

RUN PRO Programm**TRON****FORMAT:** 1. TRON**Abkürzung:** TR.**Vergleiche:** TROFF**WIRKUNG:**

TRON wird benutzt, um den Trace-Modus einzuschalten.

ANMERKUNGEN:

Der Trace-Modus unterstützt die Fehlersuche in Programmen. Ist der Trace-Modus eingeschaltet, so wird die Zeilennummer der gerade ausgeführten Zeile ausgegeben. Der PC-1280 wartet dann auf die Betätigung des Abwärtspfeils (**↓**), um die nächste Zeile auszuführen. Mit dem Aufwärtspfeil (**↑**) kann die gerade ausgeführte Zeile zur Anzeige gebracht werden. Der Trace-Modus bleibt so lange eingeschaltet, bis er mit einem TROFF-Befehl wieder ausgeschaltet wird oder bis Sie die Tastenfolge **SHIFT** und **CCE** eingeben. Nachdem ein Ergebnis im Trace-Modus an der im CURSOR-Kommando bestimmten Stelle im Display angezeigt wurde, erscheint in der darauffolgenden Zeile die nächste Zeilennummer.

Wenn im Trace-Modus die Startposition des Displays im Programm durch eine CURSOR-Anweisung bestimmt wurde, um eine Variable abzurufen oder eine manuelle Berechnung durchzuführen, wartet der PC-1280 auf Betätigung des Abwärtspfeils, der die CURSOR-Stellung löscht, wenn die Ausführung wieder aufgenommen wird.

BEISPIELE:

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

Dieses Programm gibt bei seiner Ausführung die Zeilennummern 10, 20, 30, 30, 30 und 40 an, wenn Sie die **↓**-Taste drücken.

USING

RUN PRO Programm

- FORMAT:**
1. USING
 2. USING Format Zeichenfolge
 3. USING Zeichen Variable

Abkürzung: U.

Vergleiche: LPRINT, MDF, PAUSE, PRINT

Weitere Informationen über dieses Kommando finden Sie im Anhang C.

WIRKUNG:

USING wird gebraucht, um die angezeigte oder gedruckte Ausgabe zu formatieren.

ANMERKUNGEN:

USING kann einzeln oder als Erweiterung eines LPRINT-, PAUSE- oder PRINT-Befehls eingesetzt werden. USING erstellt eine Ausgabeformatierung, die für alle folgenden Ausgaben Gültigkeit hat, bis die Formatierung durch ein neues USING geändert wird.

Format Zeichenfolge ist aus einer Kombination folgender Zeichen zusammengesetzt:

- # Rechtbündiges Feld für numerische Zeichen.
- . Dezimalpunkt
- ^ Gibt an, daß die Zahlen in Exponentialschreibweise wiedergegeben werden.
- & Linksbündiges Feld für alphanumerische Zeichen.
- , Wird als Markierung der Tausender in numerischen Feldern benutzt.

Die Spezifikation "####" erstellt z.B. ein rechtsbündiges Feld mit Platz für 3 Ziffern und das Vorzeichen. In numerischen Feldern muß immer eine Position für das Vorzeichen vorgesehen sein, auch wenn nur positive Zahlen ausgegeben werden sollen.

Die Format Zeichenfolgen können mehr als ein Feld beschreiben. "####&&&&" setzt z.B. ein numerisches Feld und ein Zeichenfeld direkt nebeneinander.

Wird keine Format Zeichenfolge gegeben, wie in Format 1, wird die Spezialformatierung abgeschaltet.

BEISPIELE:

DISPLAY

```
10: A=125:X$="ABCDEF"
20: PRINT USING "##.##^",A
30: PRINT USING "#####";X$
40: PRINT USING "#####";A;X$
```

```
ABCDEF
125ABC
```

1.25E 02



VAL

RUN PRO Programm

FORMAT: 1. VAL(X\$)
2. VAL("Zeichenfolge")

Abkürzung: V.

Vergleiche: STR\$

WIRKUNG:

Wandelt eine Zeichenfolge von numerischen Zeichen in einen Dezimalwert um.

ANMERKUNGEN:

VAL ist die umgekehrte Funktion der STR\$-Funktion. Sie ändert eine Zeichenfolge, die aus numerischen Zeichen besteht, in einen numerischen Wert, der als Zahl in der weiteren Verarbeitung benutzt werden kann.

Ist die Zeichenfolge dezimal, muß sie aus den Zeichen 0 bis 9 bestehen und einen wahlweisen Dezimalpunkt und ein Vorzeichen haben. In dieser Form ist VAL das Gegenteil der STR\$-Funktion.

Werden unzulässige Zeichen eingegeben, wird die Umwandlung bis zum ersten unzulässigen Zeichen durchgeführt.

BEISPIELE:

```
10: INPUT "CYCLE FREQUENCY ":A$
15: IF ASC(A$)<48 OR ASC(A$)>57 THEN 100
20: F= VAL(A$)
30: PRINT F
40: STOP
:
:
100: PRINT "MUST BE A NUMBER": GOTO 10
```

[10] Gibt eine Zeichenfolge ein, die in einen numerischen Wert umgewandelt werden soll.

[20] Die Zeichenfolge wird in ihr numerisches Äquivalent umgewandelt. Die Eingabe einer Zahl als Zeichenfolge gibt dem Programmierer die Möglichkeit, die Eingabe zu überprüfen, um sicher zu gehen, daß der Typ richtig ist und die Eingabe in der richtigen Reihenfolge im Programm selbst vorgenommen wurde.

RUN PRO Programm**WAIT**

FORMAT: 1. WAIT Ausdruck
2. WAIT

Abkürzung: W.

Vergleiche: PAUSE, PRINT, GPRINT

WIRKUNG:

WAIT wird benutzt, um die Zeit festzusetzen, die die Ausgabe einer Information auf der Anzeige während des Programmablaufs wiedergegeben werden soll.

ANMERKUNGEN:

Bei normaler Programmausführung wartet der PC-1280 nach einem PRINT-Kommando, bis die **ENTER**-Taste gedrückt wird. WAIT weist den PC-1280 an, eine Ausgabe für eine vorgegebene Zeit anzuzeigen und dann die Ausführung des Programms automatisch fortzusetzen (ähnlich wie bei PAUSE). Der Ausdruck nach WAIT setzt die Länge des Intervalls fest. Der Intervall kann jeden Wert von 0 bis 65535 haben. Dabei bedeutet jede Erhöhung um 1 eine Verlängerung um 1/59 Sekunde. WAIT 0 ist zu schnell, als daß die Ausgabe gut gelesen werden könnte. WAIT 65535 ist etwa 19 Minuten. WAIT ohne Ausdruck dahinter setzt den PC-1280 in seinen ursprünglichen Wartezustand, bis die **ENTER**-Taste gedrückt wird.

BEISPIELE:

10: WAIT 59

Der Rechner wartet nach PRINT etwa 1 Sekunde.

ANHANG

Fehlermeldungen	A
Zeichencode-Tabelle	B
Formatierung der Datenausgabe	C
Bewertung von Ausdrücken und Vorrang der Operatoren	D
Tastenfunktionen im BASIC	E
Fehlersuche	F
Technische Daten	G
Benutzung von Programmen anderer Computer	H
Pflege des PC-1280	I

Anhang A

FEHLERMELDUNGEN

Es gibt neuen verschiedene Fehler-Codes, die im PC-1280 eingebaut sind. Die folgende Tabell erläutert diese Codes.

Fehler-

Nummer	Bedeutung
1	1. Bedeutung
2	2. Bedeutung
3	3. Bedeutung
4	4. Bedeutung
5	5. Bedeutung
6	6. Bedeutung
7	7. Bedeutung
8	8. Bedeutung
9	9. Bedeutung
10	10. Bedeutung
11	11. Bedeutung
12	12. Bedeutung
13	13. Bedeutung
14	14. Bedeutung
15	15. Bedeutung
16	16. Bedeutung
17	17. Bedeutung
18	18. Bedeutung
19	19. Bedeutung
20	20. Bedeutung
21	21. Bedeutung
22	22. Bedeutung
23	23. Bedeutung
24	24. Bedeutung
25	25. Bedeutung
26	26. Bedeutung
27	27. Bedeutung
28	28. Bedeutung
29	29. Bedeutung
30	30. Bedeutung
31	31. Bedeutung
32	32. Bedeutung
33	33. Bedeutung
34	34. Bedeutung
35	35. Bedeutung
36	36. Bedeutung
37	37. Bedeutung
38	38. Bedeutung
39	39. Bedeutung
40	40. Bedeutung
41	41. Bedeutung
42	42. Bedeutung
43	43. Bedeutung
44	44. Bedeutung
45	45. Bedeutung
46	46. Bedeutung
47	47. Bedeutung
48	48. Bedeutung
49	49. Bedeutung
50	50. Bedeutung
51	51. Bedeutung
52	52. Bedeutung
53	53. Bedeutung
54	54. Bedeutung
55	55. Bedeutung
56	56. Bedeutung
57	57. Bedeutung
58	58. Bedeutung
59	59. Bedeutung
60	60. Bedeutung
61	61. Bedeutung
62	62. Bedeutung
63	63. Bedeutung
64	64. Bedeutung
65	65. Bedeutung
66	66. Bedeutung
67	67. Bedeutung
68	68. Bedeutung
69	69. Bedeutung
70	70. Bedeutung
71	71. Bedeutung
72	72. Bedeutung
73	73. Bedeutung
74	74. Bedeutung
75	75. Bedeutung
76	76. Bedeutung
77	77. Bedeutung
78	78. Bedeutung
79	79. Bedeutung
80	80. Bedeutung
81	81. Bedeutung
82	82. Bedeutung
83	83. Bedeutung
84	84. Bedeutung
85	85. Bedeutung
86	86. Bedeutung
87	87. Bedeutung
88	88. Bedeutung
89	89. Bedeutung
90	90. Bedeutung
91	91. Bedeutung
92	92. Bedeutung
93	93. Bedeutung
94	94. Bedeutung
95	95. Bedeutung
96	96. Bedeutung
97	97. Bedeutung
98	98. Bedeutung
99	99. Bedeutung
100	100. Bedeutung

- Syntax-Fehler
Der PC-1280 kann nicht verstehen, was Sie eingegeben haben.
Prüfen Sie die Eingabe auf Dinge, wie Semikolon am Ende einer PRINT-Anweisung, falsch geschriebene Wörter und fehlerhafte Anwendungen.
z.B. 3 * / 2
- Rechenfehler

Hier haben Sie vermutlich eines der drei folgenden Dinge getan:
 - Versucht, eine zu große Zahl zu benutzen. Das Rechenergebnis ist größer 9.999999999999999D 99.
 - Versucht durch 0 zu teilen.
z.B. 5/0
 - Versucht, eine unlogische Rechnung auszuführen.
z.B. LN-30 oder ASN 1.5
- Unzulässige Funktion (DIMensionierungsfehler/Argumentfehler)

Die Feldvariable existiert bereits.
Das Feld wurde spezifiziert, aber nicht dimensioniert.
Der Index des Feldes übersteigt die in der DIM-Anweisung vorgegebene Größe.
z.B. DIM B(256)

Unzulässiges Funktionsargument. Das heißt Sie haben versucht, den Computer etwas tun zu lassen, wozu ernicht imstande ist.
Das Intervall ist größer als 65535.
z.B.: WAIT 66000

4 Zu hohe Zeilennummer

Hier haben Sie vermutlich eines der folgenden zwei Dinge gemacht:

1. Versucht mit GOTO, GOSUB, RUN, LIST oder THEN usw. eine nicht vorhandene Zeile anzusprechen.
2. Versucht eine zu große Zeilennummer zu verwenden. Die höchstmögliche Zeilennummer ist 65279.

5 NEXT ohne FOR

Das Einrichten einer Subroutine übersteigt 10 Stufen.

Die Einrichtung einer FOR-Schleife übersteigt 5 Stufen.

Die Befehle RETURN ohne GOSUB, NEXT ohne FOR oder READ ohne DATA.

Fasungsvermögen 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. Sie kann aber auch auftreten, wenn ein Programm zu lang ist, oder wenn der Reserve-Speicher mehr als 144-Byte enthält.

7 PRINT USING Fehler

Das bedeutet, daß in einer USING-Anweisung eine unzulässige Formatspezifikation enthalten ist.

8 E/A-Anschluß- oder Disk-Fehler

Dieser Fehler kann auftreten, wenn Sie einen zusätzlichen Drucker und/oder Kassettenrekorder an den PC-1280 angeschlossen haben. Es bedeutet, daß Sie ein Kommunikationsproblem zwischen dem Gerät und dem PC-1280 haben. Dieser Fehler erscheint bei Floppy- oder RAM-Diskettenfehlern.

D.h. es wurde versucht, nicht existente Dateien zu öffnen.

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 (z.B. A\$) anzusprechen, während die Daten der Variablen unter einem anderen Namen (z.B. A) abgespeichert worden waren.

Eingabefehler

Wenn Sie ein Programm ausführen, kann eine Fehlermeldung aufgrund eines Eingabefehlers auftreten. Beachten Sie in diesem Fall die folgenden Anmerkungen:

z.B.: KPRINT wird anstelle von LPRINT eingegeben.

Im folgenden wird K in normaler Weise in ein L korrigiert, indem der Cursor über das K gebracht wird und man dann L tippt.

```

10:K PRINT A$
10 K PRINT A$
10:L PRINT
    ↑
    | Leerschritt

```

L ENTER

Wenn Sie auf diese Weise die Korrektur durchführen, versteht sie der Computer nicht als Befehl. In diesem Beispiel müssen Sie KPRINT löschen und dann erneut LPRINT eingeben.

```

10:K PRINT A$
10 K PRINT A$
10 A$
10 [ ] [ ] [ ] [ ] [ ] A$
10:LPRINT A$
    ↑
    | Kein Leerraum, da als Befehl erkannt.

```

DEL DEL
INS...INS
L P R I N T ENTER

Mit der Cursortaste können Sie den Befehl auf korrekte Eingabe überprüfen.

Wurde das Wort als ein Kommando eingegeben, so springt der Cursor von ersten Zeichen im Kommando auf den nächsten Leerraum. Wurde das Wort nicht richtig als Kommando eingegeben, so kann der Cursor auf jedes Zeichen im Wort gebracht werden, wie hier gezeigt:

(Richtige Eingabe)

10:RADIAN
 10 RADIAN
 10 RADIAN__

(Falsche Eingabe)

10:RADAN
 10 RADAN
 10 RADAN
 ,10 RADAN

ANHANG B**ZEICHENCODIERTABELLE**

Die folgende Tabelle zeigt die Umwandlungswerte für den Einsatz von CHR\$ und ASC. Die Spalte zeigt das erste hexadezimale Zeichen oder die ersten 4 Binärbits, die Reihe zeigt das zweite hexadezimale Zeichen oder die zweiten vier Binärbits. Die obere linke Ecke jedes Kastens enthält die Dezimalzahl für das Zeichen. Die untere rechte Ecke zeigt das Zeichen. Wird kein Zeichen angegeben, bedeutet es, daß das Zeichen unzulässig ist auf dem PC-1280.

Zum Beispiel das Zeichen A ist dezimal 65 oder hexadezimal 41 oder binär 01000001. Das Zeichen $\sqrt{\quad}$ ist dezimal 252 oder hexadezimal FC oder binär 11111100.

Die Zeichencodes werden folgendermaßen angezeigt:

Beispiele:

Code für *

Hexadezimal &2A

Dezimal 42 (32 + 10)

Code für $\sqrt{\quad}$

Hexadezimal &FC

Dezimal 252 (240 + 12)

Anzeige des Zeichens mit Hilfe des CHR\$-Kommandos:

Das Zeichen für den Code 0 (&00) in der Tabelle ist leer. Es wird nichts angezeigt. Die Vierecke in der Tabelle, wo kein Zeichen angegeben ist, bedeuten Leerzeichen.

Drucken eines Zeichens auf dem CE-126P mit Hilfe des CHR\$-Kommandos:

- Benutzen Sie nicht den Code 0 (&00).
- Die Vierecke in der Tabelle, wo keine Zeichen angegeben sind, bedeuten Leerzeichen.
- Die Codes 249 (&F9) und 250 (&FA) sind Leerzeichen.
- Das Zeichen für den Code 92 (&5C) ist "■".

Ersten 4 Bits

Z
w
e
i
t
e
n
4
B
i
t
s

Hex	0	1	2	3	4	5	6	7	8	E	F
Binary	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	r	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	132	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

FORMATIERUNG DER DATENAUSGABE

Es ist manchmal wichtig und nützlich, neben dem Inhalt ausgegebener Daten auch das Format zu kontrollieren. Der PC-1280 bedient sich hierzu des Befehls USING. Mit Hilfe dieses Befehls können Sie spezifizieren:

- Die Anzahl der Stellen
- Die Position des Dezimalpunkts
- Wissenschaftliche Notation
- Anzahl der Zeichen in einer Zeichenfolge

Diese verschiedenen Formate bestimmen Sie mit einer "Ausgabe-Maske", die aus einer Zeichenfolgen-Konstanten oder auch aus einer Zeichenfolgen-Variablen bestehen kann.

10: USING "# # # #"

20: M\$ = "&&&&&"

30: USING M\$

Wird der Befehl USING ohne Maske benutzt, werden alle Spezialformate aufgehoben.

40: USING

USING kann auch in einer PRINT-Anweisung 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 Masken

Eine numerische Maske darf nur benutzt werden, um numerische Werte, d.h. numerische Konstanten oder Variablen, zu kontrollieren. Wird eine Zeichenfolge-Konstante oder -Variable angezeigt, während eine numerische USING-Maske wirksam ist, wird die Maske hierauf nicht angewendet werden. Ein ausgegebener Wert muß immer in den von der Maske vorgegebenen Freiraum passen. Es muß auch Platz 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.

Sie können die MDF-Funktion zusammen mit USING benutzen, um Dezimalstellen vor dem Ausdrucken zu runden. Mit MDF können Werte in den Computer auf die gleiche Anzahl von Dezimalstellen, wie in der USING-Anweisung bestimmt, beibehalten werden. Normalerweise werden Werte bis zur nächsten ganzen Zahl aufgerundet, sofern der angeschnittene Teil 5 oder größer ist. MDF erlaubt Ihnen, den Schwellen-Aufrundungswert zu verändern.

Bestimmung der Anzahl von Stellen

Die gewünschte Zahl von Stellen wird mit Hilfe des '#'-Zeichens festgelegt. Die Anzeige oder der Ausdruck behalten immer so viele Zeichen, wie in der Maske vorgesehen sind. Dabei erscheint die Zahl rechts in diesem Bereich, die restlichen Stellen werden durch Leerschritte aufgefüllt. Bei positiven Zeichen wird sich daher auf der linken Seite der Anzeige immer mindestens ein Leerschritt befinden. Die größte verwendete numerische Maske sollte nicht mehr als 21 '#'-Zeichen enthalten. Wenn die Gesamtstellenzahl des ganzzahligen Teils 22 Stellen übersteigt, wird dieser ganzzahlige Teil als 21-stellig aufgefaßt.

Achtung: In allen Beispielen dieses Anhangs werden Sie am Anfang und am Ende des Anzeigenfeldes ein "!"- Zeichen finden, mit denen die Größe des Feldes anschaulich gemacht werden soll.

Anweisung

Display

10: USING "# # # #"

(Bringen Sie den PC- 1280 in den RUN-Modus, geben Sie RUN ein und drücken Sie **ENTER**.)

20: PRINT 25

25

30: PRINT -350

-350

40: PRINT 1000

ERROR 7 IN 40

Beachten Sie, daß die letzte Eingabe eine Fehlermeldung hervorruft, weil 5 Stelle (4 Ziffern und eine Stelle für das Vorzeichen) benötigt werden, die Maske aber nur 4 vorsieht.

Ein Komma in einer numerischen Maske wird ausgedruckt wie es ist und kann als Markierung der Tausendereinheiten verwendet werden.

Anweisung

Display

10: USING "# # #, # # #"

20: PRINT 123456

123,456

Bestimmung des Dezimalpunkts

Das Zeichen für den Dezimalpunkt "?" kann in einer numerischen Maske enthalten sein, um die diewünschte 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 ausgefüllt. Enthält der anzuzeigende Wert mehr Stellen als die Maske vorsieht, werden diese abgeschnitten (nicht gerundet).

Anweisung	Display
10: USING “#####”	
20: PRINT 25	25.00
30: PRINT -350.5	-350.50
40: PRINT 2.547	2.54

Bestimmung der wissenschaftliche Notation

Das Zeichen '^' kann in einer Maske enthalten sein, um anzuzeigen, daß die Zahl in wissenschaftlicher Notation angegeben werden soll. Die Zeichen '#' und '.' werden in der Maske benutzt, um den "charakteristischen" Teil der Zahl, d.h. der Teil, der links vom E steht (oder vom D im Modus der doppelten Genauigkeit) zu formatieren. Links vom Dezimalpunkt sollten immer zwei '#'-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 19 '#'-Zeichen erscheinen. Nach dem charakteristischen Teil wird das Exponentialzeichen E (oder D) angezeigt, gefolgt von der Stelle für das Vorzeichen und dem 2-stelligen Exponenten. Das kleinst von der Maske vorgegebene Feld für wissenschaftliche Notation wäre demnach '# # ^', damit werden Zahlen in der Form "2E 99" dargestellt. Das größtmögliche Feld für wissenschaftliche Notation wäre '# # . # ^', das Zahlen wie "—1.1234567891234567890D—12" darstellen kann.

Anweisung	Display
10: USING "# # #.# # ^"	
20: PRINT 2	2.00E 00
30: PRINT -365.278	-3.65E 02

Bestimmung alphanumerischer Masken

Zeichenfolgen-Konstante und Variablen werden mit Hilfe des "&"-Zeichens ausgegeben. Jedes "&" gibt eine im vorgesehenen Feld anzuzeigende Stelle ein. Die Zeichenfolge wird auf der linken Seite dieses Feldes angezeigt. Ist die Zeichenfolge kürzer als der dafür eingeräumte Raum, werden die rechts verbleibenden Stellen mit Leerschritten aufgefüllt. Ist die Zeichenfolge länger als das Feld, wird sie abgeschnitten.

Anweisung

Display

10: USING "&&&&&&"

20: PRINT "ABC" **ABC**

30: PRINT "ABCDEFGHI" **ABCDEF**

Gemischte Masken

In den meisten Anwendungsfällen, wird eine USING-Maske entweder alle numerischen Zeichen oder alle Zeichen zur Zeichenfolgen-Formatierung beinhalten. Für bestimmte Zwecke können aber auch beide gemeinsam in einer USING-Maske enthalten sein. In solchen Fällen markiert jede Umschaltung von Zeichenfolgen 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.

Anweisung

Display

10: PRINT USING "# # # # #.## &&";25;"CR" **25.00CR**
—5.78DB

20: PRINT —5.789;"DB"

Vergessen Sie nicht: Wurde ein USING-Format einmal spezifiziert, wirkt es sich auf alle nachfolgenden Daten aus, bis es aufgehoben oder durch einen anderen USING-Befehl ersetzt wird.

Anhang D

AUSDRÜCKEN UND VORRANG VON OPERATIONEN

Wenn in den PC-1280 ein komplexer Ausdruck eingegeben wird, bewertet er Teile dieses Ausdrucks in einer Reihenfolge, die durch die Vorrangstellung der einzelnen Teile bestimmt wird. Geben Sie folgenden Ausdruck ein:

$$100/5 + 45$$

$$\frac{100}{5 + 45} = 2 \quad \text{oder} \quad \frac{100}{5} + 45 = 65$$

Da der PC-1280 eine Möglichkeit haben muß zwischen diesen beiden Operationen zu unterscheiden, bedient er sich seiner Regeln des Operatorenvorrangs. 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 werden Operatoren vom SHARP's PC-1280 gemäß den folgenden Prioritätsregeln verarbeitet, angefangen mit der höchsten Priorität:

Stufe	Operationen
1	Klammern
2	Variablen und Pseudovariablen
3	Funktionen
4	Exponentiation (^)
5	Einheitliches Minuszeichen, negatives Vorzeichen (—)
6	Multiplikation und Division (*, /)
7	Addition und Subtraktion (+, —)
8	Verhältnis-Operatoren (<, <=, =, >, >=, >)
9	Logische Operatoren (AND, OR, 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 äußere zuletzt.

Für die Prioritätsstufen 4 und 5 gilt, daß die letzte Eingabe die höchste Priorität hat.

Zum Beispiel: $-2^4 \rightarrow -(2^4)$

$$3^{-2} \rightarrow 3^{-2}$$

Beispiel für eine Bewertungsfolge

Wir gehen aus von dem Ausdruck:

$$((3 + 5 - 2) * 6 + 2) / 10^{\text{LOG } 100}$$

Der PC-1280 würde nun zuerst die innersten Klammern verarbeitet. Da “+” und “-” auf derselben Stufe stehen, wird von links nach rechts gerechnet, also die Addition zuerst ausgeführt:

$$((8 - 2) * 6 + 2) / 10^{\text{LOG } 100}$$

Dann würde die Subtraktion durchgeführt:

$$((6) * 6 + 2) / 10^{\text{LOG } 100}$$

oder:

$$(6 * 6 + 2) / 10^{\text{LOG } 100}$$

In der nächsten Klammer würde zuerst die Multiplikation durchgeführt werden:

$$(36 + 2) / 10^{\text{LOG } 100}$$

Und dann die Addition:

$$(38) / 10^{\text{LOG } 100}$$

oder:

$$38 / 10^{\text{LOG } 100}$$

Nachdem nun die Klammern ausgelöscht sind, hat die LOG-Funktion die höchste Priorität und wird somit als nächstes ausgeführt:

$$38 / 10^2$$

Als nächstes folgt die Exponentiation:

$$38 / 100$$

Und zuletzt wird die Division ausgeführt:

$$0.38$$

Das ist der Wert des Ausdrucks.

Anhang E

TASTENFUNKTIONEN IM BASIC

BREAK ON

Wird benutzt, um den PC-1280 einzuschalten.

- Das Betätigen dieser Taste während eines Programmlaufs bewirkt, eine Unterbrechung der Programmausführung.
- Bei manuellen Operationen, Ein-/Ausgabe- Befehlen wie BEEP, CLOAD usw. wird mit Betätigung dieser Taste die Befehlsausführung unterbrochen.

SHIFT

- Die gelbe Taste mit der Aufschrift "SHIFT" muß benutzt werden, um Doppelfunktionen anzusprechen (die auf der Taste oben stehenden Zeichen).

z.B.: wird mit **SHIFT** und **&** das &-Zeichen eingegeben.

- Wenn etwas im ReSeRve-Modus gespeichert oder abgerufen wird, muß diese Taste vor der benannten Taste gedrückt werden.

CA C·CE

- Wird benutzt, um Eingabe und Anzeige zu löschen.
- Wird benutzt, um Blockaden nach einem Fehler aufzuheben.

SHIFT CA C·CE

- Löscht nicht nur den Anzeigeteil, sondern initialisiert darüber hinaus den Computer.
—Initialisierung—
- Hebt den WAIT-Timer auf.
- Löscht das Anzeige-Format (USING- Format).
- Hebt den TRON-Zustand (TROFF) auf.
- Hebt PRINT = LPRINT auf.
- Hebt Blockade durch Fehler auf.

BASIC

- Mit dieser Taste können Sie den Betriebsmodus von RUN auf PROgramm oder von PROgramm auf RUN umschalten.

- CAL** ● Mit dieser Taste können Sie vom BASIC-Modus in den CAL-Modus umschalten.
- SHIFT BASIC** ● Wird benutzt, um in den ReSeRve-Modus zu kommen.
- 0 — 9** ● Zifferntasten
- .** ● Dezimalpunkt
 ● Wird benutzt, um eine Abkürzung eines Kommandos/Befehls/Funktion einzugeben.
 ● Gibt in der Bestimmung eines USING- Formats die Stellung des Dezimalpunkts an.
- E** ● Wird benutzt, um in wissenschaftlicher Notation den Exponenten zu bestimmen.
- D** ● Zur Eingabe des Exponenten im Modus der doppelten Genauigkeit.
- /** ● Divisionstaste
- *** ● Multiplikationstaste
 ● Wird benutzt, um eine Feld-Variable in INPUT#, PRINT# usw. zu bestimmen.
- +** ● Additionstaste
- ● Subtraktionstaste
- SHIFT $\hat{\#}$** ● Wird benutzt, um Zahlen zu potenzieren.
 ● Wird benutzt, um das Exponenten- Anzeigesystem für numerische Daten in USING-Anweisungen zu spezifizieren.
- SHIFT ($<$** ● Werden benutzt, wenn logische Operationen in IF-SHIFT Sätzen 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 S P a C e) nach Betätigen der **DEF**-Taste gedrückt wird, wird das Programm von der Programmzeile an gestartet, die das gleiche Etikett hat wie der Tastencode.

A — Z

- Buchstabentasten. Diese Tasten sind Ihnen wahrscheinlich von einer gewöhnlichen Schreibmaschine vertraut.
- Wenn diese Taste gedrückt wird, werden es Großbuchstaben in der Anzeige angegeben. Wenn nach Betätigen von der **SML**-Taste eine Alphabetische Taste gedrückt wird, werden es Kleinbuchstaben in der Anzeige angegeben.

SPC

- Wird zum Eingeben von Zeichen oder Programmen benutzt, um einen Leerschritt zu produzieren.
- Wenn diese Taste gedrückt wird, werden es Großbuchstaben in der Anzeige angegeben. Wenn nach Betätigen von der **SML**-Taste eine Alphabetische Taste gedrückt wird, werden es Kleinbuchstaben in der Anzeige angegeben.

=

- In Zuweisungsanweisungen wird diese Taste benutzt, um der links vom "=" stehenden Variablen die rechts stehenden Inhalte (Zeichen oder Zahlen) zuzuweisen.
- Wird benutzt, wenn in IF-Sätzen logische Operatoren eingegeben werden.
- Im CAL-Modus zum Angeben des Endes einer Berechnung.

:

- Zum Aktivieren des Multi-Display (zwei oder mehr Werte werden gleichzeitig angezeigt).

SHIFT

- Zum Trennen von Zwei oder mehrerer Befehlen auf einer Zeile.

,

- Zum Aktivieren des Multi-Display (zwei oder mehr Werte werden gleichzeitig angezeigt).

▶

- Bewegt den Cursor nach rechts (auf einmaligen Tastendruck Bewegung um eine Stelle; wird die Taste festgehalten, Dauerfunktion).
- Zur Ausführung von Playback-Anweisungen.

- Ruft den Cursor, wenn er bei der Anzeige von Inhalten nicht angezeigt ist.
- Löscht bei manueller Rechnung Fehlermeldungen.



- Bewegt den Cursor nach links (auf einmaligen Tastendruck Bewegung um eine Stelle; wird die Taste festgehalten, Dauerfunktion).
- Sonst wie unter der -Taste beschrieben.

INS

M

- Fügt einen Leerschritt ein (Das Zeichen erscheint dabei) zwischen der durch den Cursor angegebenen und der vorhergehenden Position.

DEL

RM

- Löscht den Inhalt der Stelle, auf der der Cursor steht.

BS

CM

- Löscht die Buchstabe links von dem Cursor.

 π

SHIFT



- Zur Eingabe von Pi (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 ein INPUT- oder PRINT-Kommando unterbrochen wurde.

P ↔ NP

SHIFT

ENTER

- Stellt den druckenden oder den SHIFT ENTER nichtdruckenden Modus her, wenn ein zusätzlicher Drucker angeschlossen ist.

SML

- Zum Anwählen und Aufheben des SML 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 angegeben. Durch erneutes Drücken von **SML** schalten Sie den Kleinschriftmodus wieder aus und es werden wieder Großbuchstaben in der Anzeige angegeben.

- !** • Zur Bestimmung von Variablen in einfacher Genauigkeit.
- 37** • Zum Bestimmen und Löschen von Zeichen und zur Spezifizierung von Zeichenfolgen.
- #** • Zusammen mit der USING-Anweisung zum Bestimmen eines numerischen Felds.
Zur Bestimmung von Variablen in doppelter Genauigkeit.
- \$** • Zur Zuweisung von Zeichenvariablen.
- SHIFT &** • Zusammen mit der USING-Anweisung zum Bestimmen von Zeichenfolgen-Feldern.
- SHIFT @** • Zur Eingabe von **ENTER** in eine gespeicherte Zeichenfolge-Sequenz im RESERVE-Modus, z.B. GOTO 100 @.
- %** • Als ein in Anführungszeichen eingeschlossenes Zeichen verwenden.

Die Abwärtspfeil  und Aufwärtspfeil  Tasten haben je nach dem angewählten Modus und Status die in der folgenden Tabelle aufgelisteten Funktionen:

Modus	Status		
RUN	Programmdurchführung	Keine Funktion	
	Programm vorübergehend unterbrochen	Um die nächste Zeile durchführen zu lassen	Diese Taste festhalten, um bereits abgearbeitete Zeilen zur Anzeige zu bringen.
	PRINT, GPRINT-Anweisung wird ausgeführt		
	PRINT-Anweisung wird momentan ausgeführt		
	Unterbrechung		
	INPUT-Anweisung wird momentan ausgeführt	Keine Funktion	
	Fehlermeldung während der Programmausführung	Keine Funktion	Um fehlerhafte Zeile zur Anzeige zu Bringen
TRON-Status	Zur Fehlersuche	Diese Taste festhalten um bereits abgearbeitete Zeilen zur Anzeige zu bringen.	
PRO	(Wenn der Modus von RUN nach PRO geändert wird und die Programmzeile nicht angezeigt wird)		
	Programm vorübergehend unterbrochen	Um die unterbrochene Zeile anzuzeigen.	Wie links
	Fehlermeldung	Um die fehlerhafte Zeile anzuzeigen	Wie links
	Anderer Zustand	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.	
RSV	Keine Funktion		

ACHTUNG:

- In der Anzeige ist die **ENTER**-Taste das gleiche wie ein Leerschritt.
- Wenn etwa 11 Minuten lang keine Taste betätigt wird, schaltet sich der Computer automatisch ab (automatische Abschaltung).

Anhang F

FEHLERSUCHE

Hier erhalten Sie einige Tips, was Sie machen können, wenn Ihr SHARP PC-1280 nicht tut, was Sie von ihm erwarten. Für jedes einzelne Problem schlagen wir Ihnen eine Reihe von Lösungen vor. Sie sollten jede von Ihnen versuchen, aber immer nur eine auf einmal, bis Sie das Problem gelöst haben.

Wenn:

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 eine Antwort eingeben, aber keine Reaktion erfolgt.

Sie ein BASIC-Programm abarbeiten lassen, etwas angezeigt wird und das Programm dann anhält.

Sie eine Rechnung eingeben und diese in der Form einer BASIC-Anweisung angezeigt wird (Doppelpunkt nach der ersten Zahl).

Sie keinerlei Reaktion auf Tastenbetätigung erhalten:

Dann sollten Sie:

1. Die **ON**-Taste betätigen, um festzustellen, ob sich das Gerät automatisch abgeschaltet hat.
2. Die Batteriezellen wechseln.
3. Den Kontrast einstellen.
4. Überprüfen, ob den Schalter an der Rückseite in der LOCK-Stellung ist.
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 dann 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**.
1. Schalten Sie von PROGRAMM in den RUN-Modus um.
1. Halten Sie irgendeine Taste fest und drücken Sie den ALL RESET-Schalter.
2. Wenn Sie dann immer noch keine Reaktion erhalten, drücken Sie den ALL RESET-Schalter, ohne dabei eine andere Taste festzuhalten. Dabei werden jedoch alle Daten, Programme und Speicherinhalte gelöscht.

Anhang G:**TECHNISCHE DATEN**

Modell:	PC-1280 Taschencomputer	
Prozessor:	8 Bit CMOS CPU	
Programmiersprache:	BASIC	
System ROM:	136 K Bytes	
Speicherkapazität:	System intern ca. 1.6 K Bytes Datenbereich 208 Bytes Programm/Daten-Bereich 6558 Bytes Reservierter Bereich 144 Bytes erweiterbar auf 40 KB mit RAM-Karten	
Stack:	Subroutine: 10 Stacks FOR-NEXT: 5 Stacks	Funktion: 16 Stacks Daten: 8 Stacks
Operatoren:	Addition, Subtraktion, Multiplikation, Division, trigonometrische Funktionen und Arcusfunktionen, logarithmische und Exponentialfunktionen, Winkelumkehrfunktionen, Quadrat und Quadratwurzel, Vorzeichen, Absolut, Integral, Verhältnis-Operatoren, logische Operationen usw.	
Numerische Genauigkeit:	10 Stellen (im Modus der einfachen Genauigkeit)/ 20 Stellen (im Modus der doppelten Genauigkeit) (Mantisse) + 2 Stellen Exponent	
Aufbereitungs- Möglichkeiten:	Cursor rechts und links, Zeile auf- und abwärts, Zeicheneinfügung, Zeichenlöschung	
Speicherschutz:	CMOS, batteriegeschützt.	

- Interface-Anschluß:** 11-poliger Anschluß (für Kassetten Interface, Diskette, Drucker usw.)
- RAM-Karten-Schlitze:** Jede Karte 2KB, 4KB, 8KB, 16KB oder 32KB.
- Display:** 2 Zeilen, 24 Stellen Flüssigkristallanzeige mit 5 × 7 Punktzeichen.
- Tastatur:** 72 Tasten. Alphabetisch, numerisch, Sonderzeichen und Funktionen. Numerische Schablone. Vom Benutzer wählbare Tasten.
- Sicherungsbatterie:** Eingebaute Speichersicherungszelle Typ: CR-2032
- Stromversorgung:** 6,0 V DC @ 0,03 W
Unter normalen Umständen 120 Stunden ununterbrochener Betrieb (bei angenommenen 10 Minuten Bedienung und 50 Minuten Anzeigezeit per Stunde bei einer Temperatur von 20°C) möglich. Die Betriebszeit kann je nach Verwendung und Batterietyp leicht schwanken.
- Betriebstemperatur:** 0 — 40°C
- Abmessungen:** 135 (B) × 70.5 (T) × 19.2 (H) mm (geschlossen)
135 (B) × 141 (T) × 9.6 (H) mm (offen)
- Gewicht:** ca. 180 g (mit Batteriezellen)
- Zubehör:** Lithiumzellen (eingebaut) und ein Handbuch.
- Zusatzgeräte:** RAM-Einsteckkarten 2kB (CE-210M), 4kB (CE-211M), 8kB (CE-212M), 16kB (CE-2H16M), 32kB (CE-2H32M), Kassettenrekorder (CE-152/CE-127R), Drucker/Kassetten-Interface (CE-126P), Diskettenlaufwerk (CE-140F).

Anhang H:

BENUTZUNG VON PROGRAMMEN ANDERER COMPUTER

RAM-Karten des PC-1280 können in anderen Computern der SHARP Serie (PC-1450, PC-1460, PC-1425 und PC-1360) verwendet werden und umgekehrt. Folgen Sie den kommendenaufgeführten Hinweisen.

DIE VERWENDUNG VON RAM-KARTEN VON ANDEREN COMPUTERN IM PC-1280:

1. Überprüfen Sie, daß MEM\$ auf dem PC-1280 den Wert "2" anzeigt. Wenn nicht, führen Sie SET MEM "2" **ENTER** aus.
2. Schalten Sie den PC-1280 aus (OFF), legen Sie eine RAM-Karte von einem der anderen Computermodelle in den Schlitz des PC-1280 (siehe Abschnitt über das Einlegen von RAM-Karten).
3. Schalten Sie den PC-1280 ein und tippen:

SET MEM "1" **ENTER**

Programme der eingeführten RAM-Karte werden automatisch zum Gebrauch auf dem PC-1280 umgeformt. Beachten Sie dabei aber folgendes:

- Falls kein ausreichender Systembereich (ca. 1.6 K Bytes auf der RAM-Karte) gewährleistet ist, erscheint folgender Hinweis:

MEM\$ = "1"
RAM CARD CLEAR O.K.?

Druck auf die **Y**-Taste löscht den RAM-KARTE- SPEICHER einschließlich aller dort gespeicherten Programme. Zum Schutz aller Programme im Speicher der RAM-Karte, schalten Sie den PC-1280 aus und entnehmen Sie die Karte.

Der PC-1280 wird sich selbst automatisch ausschalten, wenn ca. 1 Minute lang keine Taste betätigt wird.

- Wenn irgendeine Zeile des eingeführten Programms länger als 80 Bytes ist, wird der überschüssige Teil abgeschnitten, und folgende Mitteilung erscheint nach der Umformung des Programms:

WARNING:
SOME LINE EXCEEDED 79B

Zum Löschen der Mitteilung drücken Sie irgendeine Taste.

- Das Bereitschaftssymbol (>) erscheint, wenn die Umformung abgeschlossen ist:

>

Achten Sie darauf, daß alle Variablen gelöscht wurden, und daß Programme, die nicht vom PC-1280 gestützte Kommandos enthalten, neugeschrieben werden müssen.

Einige auf dem PC-1360 geschriebenen Programme werden aufgrund von Speicherfehlern nicht automatisch umgeformt, wenn sie vom Band in den PC-1280 eingelesen werden. Sichern Sie solche Programme mit dem CSAVE@-Kommando auf dem PC-1360, bevor Sie versuchen, sie in den PC-1280 einzuladen.

CSAVE@ [Dateiname][, "PASSWORD"] **ENTER**

DIE ANWENDUNG VON RAM-KARTEN AUS DEM PC-1280 IN ANDEREN COMPUTERN:

RAM-Karten des PC-1280 können in jedem der oben angeführten Computern durch Stellen des PC-1280 Speichers auf "2" benutzt werden.

- Bei Anwendung im PC-1460 und PC-1425

Mit dem PC-1280 im RUN-Betrieb tippen Sie CONVERT 1 ein und drücken

ENTER.

Programme werden automatisch für den PC-1460 und den PC-1425 umgeformt.

- Bei Anwendung im PC-1360

Mit dem PC-1280 im RUN-Betrieb tippen Sie CONVERT 2 ein und drücken

ENTER.

Programme werden automatisch für den PC-1360 umgeformt.

- Bei Anwendung in PC-1450

Mit dem PC-1280 im RUN-Betrieb tippen Sie CONVERT 3 ein und drücken

ENTER.

Programme werden automatisch für den PC-1450 umgeformt.

Nach der Umformung schaltet sich der PC-1280 selbst automatisch aus (OFF).

Drücken Sie während der Umwandlung NICHT die **BREAK**-Taste. Bei unabsichtlichem Druck, startet der Umformungsvorgang von neuem.

Alle Programme, die nicht vom Empfängercomputer gestützte Programme enthalten, müssen neu geschrieben werden.

Daten können nicht umgekehrt werden. Ein Versuch, die Daten umzukehren, bewirkt ihre Löschung.

Anhang I

PFLEGE DES PC-1280

Um reibungsloses Funktionieren des PC-1280 zu gewährleisten, empfehlen wir die Beachtung folgender Punkte:

- Gehen Sie immer vorsichtig mit dem Computer um, da die Flüssigkristallanzeige aus Glas gefertigt ist.
- Halten Sie den Computer von starken Temperaturschwankungen fern, ebenso von Feuchtigkeit oder Staub. Wenn Sie bei warmem Wetter Ihren PKW längere Zeit in der prallen Sonne stehen lassen, und sich im Wageninneren hohe Temperaturen aufbauen, kann das Ihren Computer beschädigen.
- Benutzen Sie zur Reinigung des Computers nur 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 der Computer starker elektrischer Aufladung oder auch starkem Lärm 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 andere Taste heruntergedrückt (siehe Fehlersuche).
- Bewahren Sie dieses Handbuch gut auf, falls Sie später einmal etwas nachschlagen wollen.

PROGRAMMBEISPIELE

Nach dem Durchlesen der vorangehenden Beschreibungen der verschiedenen Funktionen sollten Sie eine Anzahl von Programmbefehlen kennen. Um jedoch Anwendungsprogramme in BASIC zu schreiben, sollten Sie neben den in dieser Anleitung aufgeführten Beispielen Ihre eigenen Anwendungsprogramme schreiben und ausführen.

Wie bei allen erlernten Fähigkeiten macht Übung erst den Meister und dies trifft auf das Programmieren von Computern ebenso wie das Autofahren oder Tennisspielen zu.

Es ist außerdem wichtig, Bezug auf Programme zu nehmen, die von anderen geschrieben worden sind und anhand dieser Beispiele zu lernen. Der Zweck dieses Kapitels ist daher die Einführung in Programmbeispiele, die mit einer Reihe von verschiedenen BASIC-Befehlen arbeiten.

In diesen Programmbeispielen auf den folgenden Seiten werden gewisse Konventionen benutzt, um sie leichter verständlich zu gestalten.

1. PROGRAMMLISTE

Alle hier in den Programmbeispielen aufgeführten Programmlisten sind Ausdrücke mit demn CE-126P Drucker, die aus drucktechnischen Gründen verkleinert präsentiert werden.

2. PROGRAMMUMFANG

Der Programmumfang wird durch die Bytes am Ende der einzelnen Programmlisten angegeben.

3. AUSDRUCKEN

Für alle Programme, die einen Druck auf Papier ermöglichen, werden mit dem CE-126P Drucker produzierte Ausdrücke aufgeführt. Diese werden wie die Programmlisten hier verkleinert dargestellt.

4. SPEICHERINHALTE

Für jedes Programmbeispiel wird eine Tabelle mit den diversen Speicherinhalten gezeigt. Variable mit vorbestimmter Verwendung werden durch deren spezifischen Zweck angegeben, während die nicht vorbestimmten Variablen, wie z.B. jene, die in einem Arbeitsbereich gespeichert werden, um Zwischenergebnisse festzuhalten durch einen Haken (✓) gekennzeichnet werden.

INHALT

1. Fakultätsberechnungen mit doppelter Stellengenauigkeit	317
2. Telefonliste	320
3. Klassifizierte Gesamtbeträge	327
4. Maulwurfklopfen (beliebtes japanisches Spiel)	338

Die Firma SHARP CORPORATION und/oder deren Tochterfirmen übernehmen keine Verantwortung für etwaige finanzielle Verluste oder Schäden, die durch die Ausführung der in dieser Anleitung aufgeführten Programmbeispiele erwachsen. Beim Einsetzen dieser Programme sollten Sie daher stets bedenken, daß diese Programme eventuell Ihre Anforderungen und Zwecke nicht ganz erfüllen und manche Programme nicht die für Sie notwendige Präzision liefern. Daher sollten Sie die Daten in den einzelnen von Ihnen verwendeten Programmbeispielen sorgfältig analysieren und prüfen, ob sie Ihren Anforderungen entsprechen. Falls nicht, sollten Sie diese Programme Ihren Anforderungen gemäß modifizieren.

1. Programmtitel: FAKULTÄTSBERECHNUNGEN MIT DOPPELTER STELLENGENAUIGKEIT

■ ÜBERSICHT

Ein praktisches Programm zur Fakultätsberechnung, dessen Genauigkeit selbst bei hohen Werten erhalten bleibt.

Gewöhnlich werden Werte abgerundet, jedoch ist dies in manchen Fällen nicht akzeptabel. In solch einem Fall ist das vorliegende Programm sehr praktisch.

Nach der Eingabe des Programms in den PC-1280 das Programm starten und den Wert eingeben, dessen Fakultät zu berechnen ist. Das Ergebnis wird praktischerweise in 20 Stellen unterteilt angegeben.

■ ANLEITUNG

1. Das Programm durch Eingabe von **DEF A** starten.
2. Den Wert eintippen, dessen Faktoren berechnet werden sollen.
3. Das Ergebnis wird bei Drücken von **ENTER** in 20 Stelleneinheiten dargestellt.

Das Ausgabeformat ist wie folgt:

Beispiel: Für einen Wert von 30!

40. Stelle	21. Stelle	
↓	↓	
0 0 0 0 0 0 0 2 6 5 2 5 2 8 5 9 8 1 2 1	2	}
9 1 0 5 8 6 3 6 3 0 8 4 8 0 0 0 0 0 0 0	1	
↑	↑	
20. Stelle	1. Stelle	

Die Zahlen geben die Position der 20-stelligen Sequenzen an. Nummer 1 steht für die ersten 20 Stellen usw.

4. Nach der Ausgabe der ersten Stelle **ENTER** drücken, um das Programm zu beenden.

■ BEZUG

- Was ist die Fakultät von n (ganze Zahl)?

$$n! = n (n - 1) (n - 2) \dots 2 \cdot 1$$

$$9! = 9 \times 8 \times 7 \times 6 \times 5 \times 4 \times 3 \times 2 \times 1 = 362880$$
- Der höchste in dieses Programm eingebare Wert beträgt 553!

■ BEISPIEL

Herausfinden der Fakultät von 50! mit doppelter Genauigkeit.

■ TASTENOPERATION

1. DEF A

Value=

Computer wartet auf Eingabe des Wertes zur Fakultätsberechnung.

2. 50 **ENTER**

Calculating!

Am Rechnen

0000000000000000030414 4
09320171337804361260 3

Zeigt die 8. bis 41. Stelle

3. **ENTER**

81660647688443776415 2
6896051200000000000000 1

Zeigt 40 bis 1. Stelle

4. **ENTER**

 $\angle$

Programm beenden

PROGRAMMLISTE

```

10:"A" CLEAR :CLS :
   WAIT 0: DEFDBL
20:DIM AN$(255):G$(1)*2
   4
30:AN$(0)=1
40:INPUT "Value=";K
50:CLS :PRINT "Calculations!"
60:FOR Z=2 TO K
70:FOR N=U TO M
80:P#=AN$(N)*Z+C
90:C=INT (P#/1010)
100:AN$(N)=P#-C*1010
110:NEXT N
120:IF AN$(U)=0 LET U=U+
   1
130:IF C=0 THEN 170
140:M=M+1
150:AN$(N)=C
160:C=0
170:NEXT Z
180:IF M=INT (M/2)*2=0
   LET M=M+1
190:FOR N=M TO 0 STEP -4
200:CLS : WAIT 0
210:G$(0)= RIGHT$( "0000
   00000"+ STR$ AN$(N),
   10)+ RIGHT$( "0000000
   000"+ STR$ AN$(N-1),
   10)+" "
220:G$(0)=G$(0)+ STR$ ((
   N+1)/2)
230:IF N=1 LET G$(1)="
   ": GOTO 260
240:G$(1)= RIGHT$( "0000
   00000"+ STR$ AN$(N-2
   ),10)+ RIGHT$( "00000
   00000"+ STR$ AN$(N-3
   ),10)+" "
250:G$(1)=G$(1)+ STR$ ((
   N+1)/2-1)
260:PRINT G$(0)
270:WAIT
280:PRINT G$(1)
290:NEXT N
300:CLS : END

```

599 Bytes

■ SPEICHERINHALTE

C	Übertragung
K	Wert für die zu berechnende Fakultät
M	√
N	Schleifenzähler
P#	Zur Berechnung
U	√
Z	Schleifenzähler
AN# (255)	Antwortkette
G\$(1)*24	Zur Anzeige

■ ZEILENPLAN

Zeilen	Inhalt
10~30	Grundstellung
40	Eingabe des Werts für die zu berechnende Fakultät
50~170	20-stellige Berechnungen
180~290	Anzeige in Einheiten von 20 Stellen
300	Operationsende

2. Programmtitel: TELEFONLISTE

Benötigtes Peripheriegerät: CE-140F

■ ÜBERSICHT

Telefonlisten sind ein wichtiges Werkzeug des Geschäftsmanns. Aber Telefonlisten werden beim Nachschlagen und Aufstocken schmutzig und oft zerfetzt. Dieses Programm erleichtert die Neueingabe von Namen und Telefonnummern, das Löschen von alten Daten und das schnelle Auffinden einer bestimmten Nummer. Die Daten können ebenso auf Diskette gespeichert werden.

■ ANLEITUNG

1. Das Programm durch Eingabe von **DEF A** beginnen.

Das Programm durch Eingabe von RUN und Drücken und **ENTER** starten, falls es bereits eingegeben und geladen ist oder registrierte Daten gelöscht werden sollen.

2. Ein Menü taucht auf. Die Operation durch Wählen zwischen 1 — 6 spezifizieren.

1. Auffinden

- (1) Den Namen der Person- oder Firma, deren Nummer gefunden werden soll, eintippen (max. 15 Zeichen).
- (2) Der Name wird angezeigt, wenn die Eingabe gefunden wurde. Falls eine nicht anwendbare Eingabe gefunden wird, taucht die Meldung "Not Found!" und danach wieder das Menü auf.
- (3) **ENTER** drücken, um zum Submenü überzugehen und zwischen 1 oder 2 wählen.

1. Nächste Wahl

Suchen der nächsten Nummer. Der Vorgang ist der gleiche, wie bei (2) oben.

2. Menü

Zum Wurzelmenü zurückkehren

2. Hinzufügen

- (1) Namen eintippen (max. 15 Zeichen)

- (2) Telefonnr. eingeben (max. 12 Zeichen einschl. Bindestriche)
- (3) Zum Speichern der Eingabe **Y**, zum Widerrufen **N** drücken.
- (4) Zur Eingabe eines weiteren Namens und dessen Nummer **Y** drücken und den Prozess (1) wiederholen.
N drücken, um zum Wurzelmenü zurückzukehren.
- (5) Die Meldung "Over 100 registered" taucht auf, wenn versucht wird, mehr als 100 Namen und Telefonnummern einzugeben und der Computer schaltet auf das Wurzelmenü zurück.

3. Löschen

- (1) Die Operationen (1) und (2) unter FIND wiederholen und die zu löschenden Namen und Nummern suchen.
- (2) Zum Wählen des Submenüs **ENTER** drücken und zwischen Operation 1, 2 oder 3 wählen.

1. Löschen

Der gewählte Name und die Nummer werden gelöscht und die Operation kehrt zum Wurzelmenü zurück.

2. Nächste Wahl

Der nächste Name und die nächste Nummer werden herausgesucht.

3. Menü

Schaltet auf Wurzelmenü zurück.

4. Überprüfen

Alle eingespeicherten Namen und Telefonnummern werden angezeigt. Die **ENTER**-Taste drücken, um das Display zu verschieben. **N** drücken, um zum Wurzelmenü zurückzukehren. Nach dem Anzeigen aller Daten schaltet der Computer automatisch zum Wurzelmenü zurück.

5. Diskette

- (1) Zwischen 1 und 2 dieses Submenüs wählen.
 - 1. Auf Diskette speichern
Die Daten werden auf Diskette gespeichert.
 - 2. Daten laden
Die Daten auf Diskette werden in den Speicher eingelesen.
- (2) Den Dateinamen eingeben (bis zu 8 alphanumerische Zeichen). Nach der Eingabe des Dateinamens **ENTER** drücken, um gemäß (1) zu speichern oder zu laden. Danach wird auf das Wurzelmenü umgeschaltet.

6. END

Das Programm beenden.

■ BEZUG

- Es können normalerweise bis zu 100 Eingaben gehandhabt werden. Durch Einsatz einer RAM-Karten kann die Kapazität auf bis 255 Eintragungen erweitert werden.
Den Wert MX=100 in Zeile 10 zu MX=255 ändern.

■ BEISPIEL

Die folgenden Daten eingeben.

Name	Nummer
ADAM Carl	01-345-6789
BARTLETT Mary	818-201-0118
JAMES Malcom	020-86609487
SMITH Paul	078-331-4999
SMITH Peter	09-12309876

- Versuchen Sie SMITH zu finden.
- Durch Eingabe von "SM" SMITH suchen und "delete" (löschen) auswählen, um "SMITH Paul" zu löschen.
- Das Register unter dem Dateinamen FRIEND auf Diskette speichern.
Hinweis: Alle Namen und Nummern dieses Beispiels haben keinen Bezug zu wirklichen Personen usw.

■ TASTENOPERATION

1. DEF A

```
1:Find    2:Add    3:Delete
4:Verify 5:Disk  6:End ?
```

Menü

2. 2

```
Add
Name=
```

Computer wartet auf Eingabe des Namens

3. ADAM Carl ENTER

```
Add
TEL=
```

Computer wartet auf Eingabe der Telefonnummer

4. 01-345-6789 ENTER

```
Add OK? (Y/N)
```

SAVE (Speichern) wählen oder nicht.

5. Y

```

Add Continue ? (Y/N)

```

Continue (Fortsetzen) wählen oder nicht.

6. Y

```

Add
Name=_

```

Computer wartet auf Eingabe des Namens

```

:
Vorgang wiederholen.
:

```

7. Y

```

Add Continue ? (Y/N)

```

Wie Schritt 5.

8. N

```

1:Find    2:Add    3:Delete
4:Verify  5:Disk   6:End ?

```

Zeigt Wurzelmenü

9. 1

```

Find
Name=_

```

Computer wartet auf Eingabe des zu suchenden Namens

10. SMITH **ENTER**

```

Name: SMITH Paul
TEL : 078-331-4999

```

Zeigt die erste Wahl

11. **ENTER**

```

1:Next Choice 2:Menu

```

Aktion wählen

12. 1

```

Name: SMITH Peter
TEL : 09-123-9876

```

Zeigt nächste Wahl

13. **ENTER**

```

1:Next Choice 2:Menu

```

Aktion wählen

14. 2

```

1:Find    2:Add    3:Delete
4:Verify  5:Disk   6:End ?

```

Zeigt das Wurzelmenü

15. 3

```
Delete
Name=_
```

Computer wartet auf Eingabe des zu löschenden Namens

16. SM **ENTER**

```
Name: SMITH Paul
TEL : 098-331-4999
```

Zeigt die erste Wahl

17. **ENTER**

```
Name: SMITH Paul
1:Delete 2:Next 3:Menu
```

Aktion wählen

18. 1

```
1:Find 2:Add 3>Delete
4:Verify 5:Disk 6:End ?
```

Zeigt das Wurzelmenü

19. 5

```
Disk
1.Data save 2.Data load
```

Wählt Aktion.

20. 1

```
File name=_
```

Computer wartet auf Eingabe des zu speichernden Dateinamens.

21. FRIEND **ENTER**

```
Press ENTER when disk is
ready !
```

Computer wartet auf Drücken der **ENTER**-Taste.

22. **ENTER**

```
Wait !
```

Speichert Daten auf Diskette.

```
1:Find 2:Add 3>Delete
4:Verify 5:Disk 6:End ?
```

Zeigt das Wurzelmenü

23. 6

```
>
```

Programmende

PROGRAMMLISTE

```

10:CLER:Y#:=100: DON Y
   F(N#)+15:TF(N#)+12
20:FF=1
30:"A" CLS : WAIT 0
40:ON ERROR GOTO 1200
50:IF FF=0 THEN 1010
60:PRINT "1:Find 2:Ad
   3:Delete"
70:PRINT "4:Verify 5:Di
   st 6:End ?"
80:Z#:= INKEY$: IF Z#="
   " THEN 80
90:IF Z#="2" GOSUB 170:
   GOTO 160
100:IF Z#="5" GOSUB 810:
   GOTO 160
110:IF Z#="6" CLS : END
120:IF ( VAL Z#<1)+(
   VAL Z#>6)+( VAL Z#:=1
   AND D=0)+( VAL Z#:=3
   AND D=0)+( VAL Z#:=4
   AND D=0) BEEP 1:
   GOTO 80
130:IF Z#="3" LET Z#="2"
   : GOTO 150
140:IF Z#="4" LET Z#="3"
150:ON VAL Z# GOSUB 550,
   420:680
160:BEEP 1: GOTO 30
170:CLS : WAIT 0:0=1
180:W#="Add"
190:GOSUB 1160
200:IF MM=-1 THEN 390
210:BN#="":BT#=""
220:PRINT W#
230:INPUT "Name=":BN#
240:IF ( LEN BN#>15)+(
   LEN BN#<1) BEEP 1:
   CLS : GOTO 220
250:CLS
260:PRINT W#
270:INPUT "TEL=":BT#
280:IF ( LEN BT#>12)+(
   LEN BT#<1) BEEP 1:
   CLS : GOTO 260
290:CLS : CURSOR 5:
   PRINT "Add OK ? (Y/N
   )"
300:Z#:= INKEY$: IF Z#="
   " THEN 300
310:IF Z#="N" THEN 340
320:IF Z#<>"Y" BEEP 1:
   GOTO 300
330:N#(MM)=BN#:T#(MM)=BT
   #
340:CLS : CURSOR 2:
   PRINT "Add Continue
   ? (Y/N)"
350:Z#:= INKEY$: IF Z#="
   " THEN 350
360:IF Z#="N" RETURN
370:IF Z#="Y" THEN 170
380:BEEP 1: GOTO 350
390:WAIT 100: BEEP 3

400:CLS : CURSOR 4:
   PRINT STR$ "MX+1:"
   "eistered 1"
410:WAIT 0: RETURN
420:CLS : WAIT 0:S=0
430:W#="Delete"
440:PRINT W#
450:BN#="": INPUT "Name=
   " :BN#
460:KL:= LEN BN#
470:IF (KL>15)+(KL<1)
   BEEP 1: CLS : GOTO 4
   40
480:GOSUB 1070: IF SC=-1
   RETURN
490:WAIT 0: CURSOR 2:
   PRINT " 1:Delete 2:H
   elp 3:Menu"
500:Z#:= INKEY$: IF Z#="
   " THEN 500
510:IF Z#="1" LET T#(SC)
   ="" :N#(SC)="" :
   RETURN
520:IF Z#="2" LET S=SC+1
   : GOTO 480
530:IF Z#="3" RETURN
540:BEEP 1: GOTO 500
550:CLS : WAIT 0:S=0
560:W#="Find"
570:PRINT W#
580:BN#="": INPUT "Name=
   " :BN#
590:KL:= LEN BN#
600:IF (KL>15)+(KL<1)
   BEEP 1: CLS : GOTO 5
   20
610:GOSUB 1070: IF SC=-1
   RETURN
620:CLS
630:WAIT 0: CURSOR 2:
   PRINT "1:Next choice
   2:Menu"
640:Z#:= INKEY$: IF Z#="
   " THEN 640
650:IF Z#="1" LET S=SC+1
   : GOTO 610
660:IF Z#="2" RETURN
670:BEEP 1: GOTO 640
680:CLS
690:FOR I=0 TO MX
700:IF N#(I)="" THEN 790
710:CLS
720:WAIT 0: PRINT "Name:
   " :N#(I)
730:WAIT : PRINT "TEL : "
   :T#(I)
740:WAIT 0
750:CLS : CURSOR 4:
   PRINT "Continue ? (Y
   /N)"
760:Z#:= INKEY$: IF Z#="
   " THEN 760
770:IF Z#="N" LET I:=I+1:
   GOTO 790

780:IF Z#<>"Y" BEEP 1:
   GOTO 760
790:NEXT I
800:RETURN
810:WAIT 0
820:CLS : PRINT "Disk"
830:PRINT "1.Data some
   2.Data load"
840:Z#:= INKEY$: IF Z#="
   " THEN 840
850:IF Z#="1" OR Z#="2"
   THEN 970
860:BEEP 1: GOTO 840
870:CLOSE #2: OPEN "2:"+
   F# FOR OUTPUT AS #2
880:PRINT #2:D,MM
890:PRINT #2,N#(*),T#(*)
900:CLOSE #2
910:RETURN
920:CLOSE #2: OPEN "N:"+
   F# FOR INPUT AS #2
930:INPUT #2,D,MM
940:INPUT #2,N#(*),T#(*)
950:CLOSE #2
960:RETURN
970:CLS
980:F#="": INPUT "File
   name=":F#
990:IF ( LEN F#>8)+(
   LEN F#<1) BEEP 1:
   GOTO 970
1000:CLS : PRINT "Press
   ENTER when disk i
   s"
1010:CURSOR 32: PRINT "
   ready !"
1020:W#:= INKEY$: IF W#
   ="" THEN 1020
1030:IF W#:= CHR$( 80)
   THEN 1050
1040:BEEP 1: GOTO 1020
1050:CLS : CURSOR 9:
   PRINT "Wait !"
1060:ON VAL Z# GOTO 870
   ,920
1070:SC=-1: CLS :
   PRINT W#
1080:KL:= LEN BN#
1090:FOR I=S TO MX
1100:IF LEFT$( N#(I),KL
   )=BN# LET SC:=I:I=M
   X
1110:NEXT I
1120:IF SC=-1 WAIT 200:
   CLS : CURSOR 6:
   PRINT "Not Found
   !": WAIT 0:
   RETURN
1130:CLS
1140:WAIT 0: PRINT "Nam
   e=":N#(SC)
1150:WAIT : PRINT "TEL
   " :T#(SC): RETURN

```

PROGRAMMBEISPIELE

```

1160:MM=-1
1170:FOR I=0 TO MX
1180:IF N$(I)="" LET MM
      =I:I=MX
1190:NEXT I
1200:RETURN
1210:CLS : WAIT 200:
      PRINT " Start usin
      " RUN ENTER!":
      WAIT 0: GOTO 1230
1220:CLS : BEEP 5:
      WAIT 200: CURSOR 0
      : PRINT "ERROR !"
      : WAIT 0
1230:CLS : END
  
```

2,408 Bytes

■ SPEICHERINHALTE

D	Datenflagge
I	✓
S	✓
W\$	✓
Z\$	Für Eingabe
FF	✓
KL	Länge der Zeichenkette
MM	✓
MX	Max. Kettenanzahl
SC	Auffindungsflagge
BN\$	Zur Namenseingabe
BT\$	Zur Telefonnummerneingabe
FI\$	Dateiname
N\$(MX)*15	Name
T\$(MX)*12	Telefonnummer

■ ZEILENPLAN

Zeilen	Inhalt
10~ 20	Grundstellung
30~ 150	Menü
160~ 400	Hinzufügen
410~ 530	Löschen
540~ 660	Suchen
670~ 790	Überprüfen
800~1060	Diskettenzugriff
1070~1150	Subroutine finden
1160~1200	Zu speichernde Leeradressen
1210~1230	Fehlermeldungen

3. Programmtitel: KLASSIFIZIERTE GESAMTBETRÄGE

Benötigte Peripheriegeräte: CE-126P und CE-140F

■ ÜBERSICHT

Durch simples Eintippen der Codes und Preise von bereits aufgenommenen Objekten können die Gesamtbeträge selbst für eingegebene Zeitspannen blitzschnell errechnet werden. Daten können auf Diskette gespeichert werden.

■ ANLEITUNG

1. **DEF B** Registrieren von Objekten

- (1) **DEF B** eintippen, um das Programm zu starten, Einstellungen zu initialisieren und Objekte zu registrieren. Durch das Initialisieren werden zuvor aufgenommene Daten gelöscht.
- (2) Nach der Initialisierung die Namen der Objekte mit jeweils max. 15 Zeichen eingeben. Das Programm kann bis 50 Objekte aufnehmen. Die Eingabeprozedur ist für jedes Objekt zu wiederholen. **ENTER** ohne Eingabe drücken, um aus der Eingabeschleife zu gehen. Das Programm springt automatisch aus der Eingabeschleife, nachdem 50 Objekte registriert sind.

2. **DEF C** Objekte anfügen

- (1) **DEF C** drücken, um das Programm zu starten und neue Objekte anzufügen. Das Programm startet mit (2), siehe oben.

3. **DEF D** Objekte editieren

- (1) **DEF D** eintippen, um das Programm zu starten und Eingaben zu verändern. Zuerst den CODE für das zu bearbeitende Objekt eintippen. Dann **ENTER** ohne weitere Eingabe drücken, um aus der Editierschleife herauszugehen.
- (2) Den angezeigten alten Namen mit dem neuen Objektamen (bis zu 15 Zeichen) überschreiben. Danach kehrt die Operation zu (1) zurück.

4. **DEF A** Dateneingabe

- (1) **DEF A** eintippen, um das Programm zu starten und Daten einzugeben. Zuerst das Datum in 4 Stellen eingeben. Das Format ist Monat, Tag (MM.TT.). Z.B. **0 5 2 9** für den 29. Mai eintippen und dann **ENTER** drücken.
- (2) Als nächstes den Objektcode eingeben. Nach der Codeeingabe stehen verschiedene Optionen zur Wahl:

Tastenoperation	Programmoperation
C ENTER	Die unmittelbar zuvor eingegebenen Daten werden gelöscht. Operation kehrt zur Code-Eingabe zurück.
? ENTER	Dies ermöglicht die Eingabe des Objektnamens ohne Code. Das Programm geht zu (2) oben, wenn der Objektname gefunden wurde. Falls nicht, erscheint die Meldung NOT REGISTERED (nicht registriert), wonach der Code eingegeben wird. Zum Abschluß der Eingabe ENTER drücken.

(3) Anschließend wird der Betrag mit bis zu acht Zahlen eingegeben. Es gibt zwei Möglichkeiten: entweder den Gesamtpreis oder den Preis $\times$ Anzahl der Objekte. Als Beispiel für die erste Methode **1 0 0 0** für 1000\$ eintippen. Bei der zweiten Methode wird **5 0 * 2 0** für 20 Objekte a 50\$ eingegeben. Nach der Preiseingabe kehrt das Programm zu (2) zurück.

5. **DEF Z** Drucken der Gesamtbeträge

Zum Starten des Programms und Drucken der Gesamtbeträge **DEF Z** eintippen. Falls der Drucker bereit ist, die **S**-Taste drücken, um das Drucken zu starten. Zum Abbrechen des Druckvorgangs und Herausgehen aus dem Programm eine beliebige andere Taste drücken. Falls **S** betätigt wird, stoppt das Programm nach Drucken des Gesamtbetrags.

6. **DEF G** Gesamtbeträge löschen.

DEF G eintippen, um das Programm zu starten und Gesamtbeträge zu löschen. Das Programm stoppt nach der Löschung. Objektnamen werden damit jedoch nicht aus dem Register gelöscht.

7. **DEF F** Speichern/Laden von Daten

(1) **DEF F** eintippen, um das Programm zu starten und auf die Diskette zugreifen zu können. Zum Wählen von DATA SAVE (Datenspeicherung) 1, zum Wählen von DATA LOAD (Daten einlesen) 2 drücken.

(2) Den Dateinamen mit bis zu 4 Zeichen eingeben.

(3) Wenn die Diskette bereit ist, **S** drücken. Zum Abbrechen und Herausgehen aus dem Programm eine beliebige andere Taste drücken. Falls **S** betätigt wird, stoppt das Programm nach dem Speichern/Laden der Daten. Wie in (1) gewählt.

8. **DEF X** Gesamtbeträge für bestimmte Zeitspannen

- (1) **DEF X** eintippen, um das Programm zu starten und die Gesamtsummen für eine bestimmte Zeitspanne zu berechnen. Die Gesamtsummenberechnung für bestimmte Zeitspannen bezieht sich auf Laden der innerhalb einer bestimmten Zeitspanne eingegebenen Daten von Diskette und deren Addierung.
- (2) Das Start- und das Enddatum mit jeweils vier Zahlen eingeben (MM.TT). Niemals zwei aufeinanderfolgende Tage eingeben.
- (3) Die Daten zwischen den zwei Tagen werden geladen und addiert, wonach das Programm stoppt.

■ **BEZUG**

(1) Max. Objektanzahl

Die max. Anz. von Objekten ist in Programmzeile 30 auf 50 beschränkt. Diese Anzahl kann jedoch durch Verwendung der CE-210M RAM-Karte auf 150, durch Verwenden der CE-211M auf 210 und durch Einsatz der CE-212M RAM-Karte auf 255 erhöht werden.

Standard	CE-210M	CE-211M	CE-212M
90	150	210	255

- (2) Dieses Programm kalkuliert 28 Tage für Februar. Für Schaltjahre ist die zweite Klammer auf Zeile 1690 des Programms von (F>28) zu (F>29) verändern.

■ **BEISPIEL**

Das Bäckerduzend (Gesamtumsatz einer Bäckerei)

- (1) Folgende acht Objekte und ihre Codes registrieren.

Code	Objekt	Code	Objekt
1	Ring donut	5	Sliced bread6
2	Iced cake	6	Sliced bread8
3	Jam donut	7	Croissant
4	Rock cake	8	Dinner roll

Code 8 mittels der Zufügensfunktion des Programms eingeben.

■ **AUSDRUCK 1**

```

1.12
No. 1 Ring donut
Price $      1.500
Cost %      9.16
-----
No. 2 Iced cake
Price $      1.600
Cost %      9.77
-----
No. 3 Jam donut
Price $      1.400
Cost %      8.55
-----
No. 4 Rock cake
Price $      3.240
Cost %     19.78
-----
No. 5 Sliced bread6
Price $      4.320
Cost %     26.37
-----
No. 6 Sliced bread8
Price $      960
Cost %      5.86
-----
No. 7 Croissant
Price $      1.680
Cost %     10.26
-----
No. 8 Dinner roll
Price $      1.680
Cost %     10.26
-----

```

(2) Die Verkaufssummen für 3 Tage eingeben und dann die Daten speichern:

Code			12. Januar	13. Januar	14. Januar
	Einheit	Anzahl	Umsatz	Umsatz	Umsatz
1	—	—	1500	1200	650
2	50	32	—	1250	800
3	50	28	—	1350	1400
4	—	—	3240	450	500
5	120	12	—	1440	1920
6	120	8	—	1680	2640
7	70	24	—	1330	1120
8	40	42	—	1480	1640
5	120	24	—		

(3) Als nächstes die Gesamtbeträge der drei Tage berechnen.

AUSDRUCK 2

1.13
No. 1 Rins donut
Price \$ 1,200
Cost % 11.79

No. 2 Iced cake
Price \$ 1,250
Cost % 12.28

No. 3 Jam donut
Price \$ 1,350
Cost % 13.26

No. 4 Rock cake
Price \$ 450
Cost % 4.42

No. 5 Sliced bread6
Price \$ 1,440
Cost % 14.15

No. 6 Sliced bread8
Price \$ 1,680
Cost % 16.50

No. 7 Croissant
Price \$ 1,330
Cost % 13.06

No. 8 Dinner roll
Price \$ 1,480
Cost % 14.54

1.14
No. 1 Rins donut
Price \$ 650
Cost % 6.09

No. 2 Iced cake
Price \$ 800
Cost % 7.50

No. 3 Jam donut
Price \$ 1,400
Cost % 13.12

No. 4 Rock cake
Price \$ 500
Cost % 4.69

No. 5 Sliced bread6
Price \$ 1,920
Cost % 17.99

No. 6 Sliced bread8
Price \$ 2,640
Cost % 24.74

No. 7 Croissant
Price \$ 1,120
Cost % 10.50

No. 8 Dinner roll
Price \$ 1,640
Cost % 15.37

1.12- 1.14
No. 1 Rins donut
Price \$ 3,550
Cost % 9.00

No. 2 Iced cake
Price \$ 3,650
Cost % 9.80

No. 3 Jam donut
Price \$ 4,150
Cost % 11.10

No. 4 Rock cake
Price \$ 4,190
Cost % 11.20

No. 5 Sliced bread6
Price \$ 7,680
Cost % 20.60

No. 6 Sliced bread8
Price \$ 5,280
Cost % 14.10

No. 7 Croissant
Price \$ 4,130
Cost % 11.00

No. 8 Dinner roll
Price \$ 4,800
Cost % 12.80

■ TASTENOPERATION

<Registrieren von Objekten>

1. **DEF B**

```
Code No.1
Item Name=.
```

Computer wartet auf Eingabe des Objektnamens für Codenr. 1.

2. Ring donut **ENTER**

```
Code No.2
Item Name=_
```

Computer wartet auf Eingabe des Objektnamens für Codenr. 2.

:

Namenseingabe fortsetzen

3. Croissant **ENTER**

```
Code No.8
Item Name=_
```

Computer wartet auf Eingabe des Objektnamens für Codenr. 8.

4. **ENTER**

```
>
```

Programmende

<Zusätzliche Registrierung von Objektnamen>

1. **DEF C**

```
Code No.8
Item Name=
```

Computer wartet auf Eingabe des Objektnamens für Codenr. 8.

2. Dinner roll **ENTER**

```
Code No.9
Item Name=_
```

Computer wartet auf Eingabe des Objektnamens für Codenr. 9.

3. **ENTER**

```
>
```

Programmende

<Editieren von Programmnamen>

1. **DEF D**

```
Code No.1
Item Name=
```

Computer wartet auf Codenummerneingabe für den zu editierenden Objektnamen.

2. 4 **ENTER**

Rock Cake
Item for editing

Computer wartet auf Codenummerneingabe für den zu editierenden Objektnamen.

3. Rock cake **ENTER**

Code No. 4
Rock Bun

Code No. = _

Computer wartet auf Codenummerneingabe für den zu editierenden Objektnamen.

4. **ENTER**

>

Programmende

<Dateneingabe>

1. **DEF A**

DATA : mm . dd
?

Computer wartet auf Dateneingabe.

2. 0112 **ENTER**

DATE: 1.12
Code No. 1 = _

Computer wartet auf Codenr.

3. 1 **ENTER**

Ring Donut
Price = _

Computer wartet auf Preiseingabe.

4. 1500 **ENTER**

DATE: 1.12
Code No. 1 = _

Computer wartet auf Codeeingabe.

Preiseingabe fortsetzen.

5. 120*24 **ENTER**

DATE: 1.12
Code No. 1 = _

Computer wartet auf Codenr.

6. **ENTER**

>

Programmende

<Drucken der Gesamtbetrags tabellen>

1. **DEF Z**

```
* Totals Printout *
Start using Skey!
```

Computer wartet auf Drücken der S-Taste.

2. S

```
* Totals Printout *
Wait !
```

Drucken

```
>
```

Programmende

<Speichern/Laden von Daten>

1. **DEF F**

```
* Disk Access *
1.Save 2.Load
```

Wählt die Aktion

2. 1

```
* Data Save *
File Name =
```

Computer wartet auf Eingabe des Dateinamens

3. 0112 **ENTER**

```
* Data Save *
Start using Skey!
```

Computer wartet auf Drücken der S-Taste.

4. S

```
* Data Save *
Wait !
```

Speichert

```
>
```

Programmende

<Löschen von Gesamtbeträgen>

1. **DEF G**

```
Data Deletion Complete !
```

Ende der Datenlöschung

```
>
```

Ende der Gesamtbetragslöschung

* Das Programm wiederholt sich von den "Tagen der Dateneingabe" bis zur "Löschung der Gesamtbeträge."

<Gesamtbeträge für Zeitspannen>

1. DEF X

```

* Period Data *
Starting Date=_

```

Computer wartet auf Eingabe des
Anfangstages.

2. 0112 ENTER

```

* Period Data *
Ending Date=_

```

Computer wartet auf Eingabe des
Endtages.

3. 0114 ENTER

```

* Period Data *
1.12 Loading!

```

Einlesen der Daten

4. S

```

* Period Data *
1.13 Loading!

```

Einlesen der Daten

```

* Period Data *
1.14 Loading!

```

Einlesen der Daten

```

>

```

Programmende

PROGRAMMLISTE

```

10:"B" WAIT 0
20:ON ERROR GOTO 1710
30:CLEAR :A=50: GOSUB 1
60
40:B=B+1: IF B=A THEN
60
50:CLS : CURSOR 6:
PRINT "Registered !"
: BEEP 2: CLS : END
60:CLS : PRINT "Code No
": STR$ B
70:INPUT "Item Name=":I
$(0): GOTO 90
80:N=B-1: CLS : END
90:IF LEN I$(0)>15
GOSUB 510: GOTO 60
100:S$(B)=I$(0): GOTO 40
110:"D" WAIT 0
120:ON ERROR GOTO 1710
130:CLS : INPUT "Code No
":I0: GOTO 150
140:CLS : END
150:C=INT C: IF C>N OR
C<=0 BEEP 1: GOTO 13
0
160:CLS
170:PRINT S$(C): INPUT "
Item for editing=":I1
$(0): GOTO 190
180:GOTO 210
190:IF LEN I$(0)>15
GOSUB 510: GOTO 160
200:S$(C)=I$(0)
210:CLS : PRINT "Code No
": STR$ C: PAUSE S$
(C): GOTO 130
220:"C" WAIT 0:B=N
230:ON ERROR GOTO 1710
240:GOTO 40
250:"A" WAIT 0
260:ON ERROR GOTO 1710
270:IF N=0 GOSUB 500:
CLS : END
280:CLS : PRINT "DATE:mm
,dd"
290:CURSOR 29: INPUT P$
300:GOSUB 1630: IF 0=1
BEEP 1: GOTO 280
310:E$=RIGHT$( " "+
STR$ E,2)
320:F$=RIGHT$( " "+
STR$ F,2)
330:CLS : PRINT "DATE:";
E$;",";F$
340:INPUT "Code No.=":C$
: GOTO 370
350:Z=0
360:CLS : END
370:IF C$="" THEN 440
380:IF C$="C" GOSUB 520:
GOTO 330
390:D=INT VAL C$: IF D=
0 OR D>N GOSUB 500:
GOTO 330
400:CLS : PRINT S$(D)
410:K=0: INPUT "Price=";
K
420:IF K>=1E8 CLS :
CURSOR 7: PAUSE "Too
Big !": BEEP 2:
GOTO 400
430:K(D)=K(D)+K(K)=K(0
)+K: GOTO 330
440:INPUT "Item Name=":I
$(0): GOTO 460
450:GOTO 330
460:FOR I=1 TO N
470:IF S$(I)=I$(0) LET D
=I:I=N
480:NEXT I: IF D=0
GOSUB 500: GOTO 330
490:GOTO 400
500:CLS : CURSOR 4:
PAUSE "Not Registere
d !": BEEP 2:
RETURN
510:CLS : CURSOR 7:
PAUSE "Too Long !":
BEEP 1: RETURN
520:IF K=0 RETURN
530:K(D)=K(D)-K(K)=K(0
)-K
540:CLS : PAUSE " ALL D
ata Canceled !":K=0
: BEEP 2: RETURN
550:"G" WAIT 0
560:ON ERROR GOTO 1710
570:ERASE K
580:DIM K(A)
590:CLS : PAUSE "Data De
letion Complete !"
600:BEEP 2
610:CLS : END
620:"Z" WAIT 0
630:ON ERROR GOTO 1710
640:IF N=0 GOSUB 500:
CLS : END
650:IF K(0)=0 CLS :
PAUSE " Not Fesis
tered !": BEEP 2:
CLS : END
660:CLS : PRINT " * Tot
als Printout *":
GOSUB 790: IF ASC L$
<>83 CLS : END
670:IF Z=1 LPRINT X$;","
;Y$;",";I$(0);",";I0$;
GOTO 690
680:LPRINT E$;",";F$
690:FOR I=1 TO N
700:LPRINT "No.:";
RIGHT$( " "+STR$ I
,3);",";I$(I)
710:LPRINT "Price $ ";
USING "#####";K(I)
720:USING
730:LPRINT "Cost %
"; USING "####.
";MDF (K(I)/K(0)*
100)
740:USING
750:LPRINT "-----
-----"
760:NEXT I
770:LPRINT "": LPRINT "
780:CLS : END
790:CURSOR 24: PRINT "
Start using S key!"
800:L$=INKEY$: IF L$="
" THEN 800
810:IF L$<>"S" RETURN
820:CURSOR 24: PRINT "
Wait ! "
830:RETURN
840:"F" WAIT 0: CLS
850:ON ERROR GOTO 1710
860:CURSOR 4: PRINT " * D
isk Access *"
870:CURSOR 29: PRINT "1.
Save 2.Load"
880:L$=INKEY$: IF L$="
" THEN 880
890:ON VAL L$ GOTO 910,1
040
900:BEEP 1: GOTO 880
910:CLS
920:CURSOR 5: PRINT " * D
ata Save *"
930:INPUT "Date=":F$
940:GOSUB 1630: IF 0=1
BEEP 1: GOTO 910
950:GOSUB 790: IF ASC L$
<>83 CLS : END
960:E$=RIGHT$( " "+
STR$ E,2)
970:F$=RIGHT$( " "+
STR$ F,2)
980:F1$="X:";E$;F$
990:CLOSE #2: OPEN F1$
FOR OUTPUT AS #2
1000:PRINT #2;A;N
1010:PRINT #2;S$(*);K(
)
1020:CLOSE #2
1030:CLS : END
1040:CLS
1050:CURSOR 5: PRINT " *
Data Load *"
1060:INPUT "Date=":F$
1070:GOSUB 1630: IF 0=1
BEEP 1: GOTO 1040
1080:GOSUB 790: IF ASC
L$<>83 CLS : END
1090:E$=RIGHT$( " "+
STR$ E,2)
1100:F$=RIGHT$( " "+
STR$ F,2)
1110:F1$="X:";E$;F$
1120:CLOSE #2: OPEN F1$
FOR INPUT AS #2

```

PROGRAMMBEISPIELE

```

1130:INPUT #2,A,N
1140:ERASE K,I$,S$
1150:GOSUB 1620
1160:INPUT #2,S$(*),K(*)
)
1170:CLOSE #2
1180:Z=0
1190:CLS : END
1200:"X" WAIT 0
1210:ON ERROR GOTO 1710
1220:Z=0
1230:DP$="* Period Dat
a * "
1240:CLS
1250:CURSOR 4: PRINT DP
$
1260:INPUT "Starting Date=";I$
1270:GOSUB 1630: IF Q=1
BEEP 1: GOTO 1240
1280:V$= RIGHT$( " " +
STR$ E;2)
1290:W$= RIGHT$( " " +
STR$ F;2)
1300:X$=V$:Y$=W$
1310:CLS
1320:CURSOR 4: PRINT DP
$
1330:INPUT "Ending Date=";I$
1340:GOSUB 1630: IF Q=1
BEEP 1: GOTO 1310
1350:M$= RIGHT$( " " +
STR$ E;2)
1360:O$= RIGHT$( " " +
STR$ F;2)
1370:FI$="X":Y$=W$
1380:GOSUB 1490
1390:IF V$=M$ AND W$=O$
CLOSE #2: ERASE KK
: CLS : END
1400:E= VAL V$:F= VAL W
$
1410:IF E=2 LET G=1:
GOTO 1440
1420:G=3
1430:IF (E/7)*(E- INT (
E/2)*2=0)+(E>8)*(E
- INT (E/2)*2=1)
LET G=2
1440:V$= STR$ (E+(F=28)
*(G=1)+(F=30)*(G=2
)+(F=31)*(G=3))
1450:W$= STR$ (F+1-(F=2
8)*(G=1)*28-(F=30)
*(G=2)*30-(F=31)*(
G=3)*31)
1460:V$= RIGHT$( " " +V$
;2)
1470:W$= RIGHT$( " " +W$
;2)
1480:GOTO 1370
1490:CLS
1500:CURSOR 4: PRINT DP
$

```

```

1510:CURSOR 29: PRINT V
$I","I$;" Loading
!"
1520:CLOSE #2: OPEN FI$
FOR INPUT AS #2
1530:INPUT #2,A,N
1540:IF Z=1 THEN 1580
1550:ERASE K,S$,KK
1560:DIM K(A),S$(A),KK(
A)
1570:Z=1
1580:INPUT #2,S$(*),KK(
*)
1590:CLOSE #2
1600:FOR I=0 TO A:K(I)=
K(I)+KK(I): NEXT I
1610:RETURN
1620:DIM K(A),I$(0)*20:
S$(A)*15:B=N:
RETURN
1630:Q=0: IF LEN P<>4
LET Q=1: RETURN
1640:E= VAL LEFT$( P;2
):F= VAL RIGHT$( P
;2)
1650:IF E<1 OR E>12
LET Q=1: RETURN
1660:IF E=2 LET G=1:
GOTO 1690
1670:G=3
1680:IF (E/7)*(E- INT (
E/2)*2=0)+(E>8)*(E
- INT (E/2)*2=1)
LET G=2
1690:IF (F(1)+(F=28)*(G
=1)+(F=30)*(G=2)+(
F=31)*(G=3) LET Q=
1
1700:RETURN
1710:CLS : BEEP 5:
WAIT 200: CURSOR 8
: PRINT "ERROR !"
: WAIT 0
1720:CLS : END

```

■ **SPEICHERINHALTE**

A	Max. Objektanzahl
B	Objektanzahlzähler
C, C\$	Für Codenr.-Eingabe
D	Codenr.
E, E\$	Datumseingabe (Monat mm)
F, F\$	Datumseingabe (Tag tt)
G, G\$	Jahreseingabe (yy)
H, H\$	✓
I	✓
J	✓
K	Preis
L\$	✓
N	Anzahl der registrierten Objekte
P\$	✓
Q	✓
T\$	Dateiname
U\$	Dateiname
K(A)	Für Gesamtpreise
IS(Ø)*20	Für Eingabe von Objektnamen
SS(5)*15	Objektnamen
KK (A)	Für Gesamtbertäge innerhalb einer Zeitspanne

≈ **ZEILENPLAN**

Zeilen	Inhalte
10~ 30	Initialisierung
40~ 100	Objektnameneingabe
110~ 210	Objektnameneditierung
220~ 240	Objektnamenänderung
250~ 430	Dateneingabe
440~ 490	Objektnamensfindung
500~ 510	Fehlerdisplay
520~ 540	Löschung von Eingaben
550~ 610	Löschen von Gesamtbeträgen
620~ 780	Drucken von Gesamtbeträgen
790~ 830	Eingabe über S-Taste
840~1190	Diskettenzugriff
1200~1610	Verarbeiten der Daten einer Zeitspanne
1620	Initialisierung der Subroutine
1630~1700	Subroutine zur Datenprüfung
1710~1720	Fehler-Verarbeitung

4. Programmtitel: MAULWURFKLOPFEN (beliebtes japanisches Spiel)

■ ÜBERBLICK

Ein Spiel, um Ihre Intuition und reaktion zu testen.

■ SPIEL

1. **DEF A** zum Starten des Programms eintippen.
2. Nach Erscheinen des Programmtitels eine beliebige Taste zum Beginnen des Spiels drücken. Zahlen blinken auf dem Display auf: versuchen Sie die gleichen Zahlen auf der Tastatur zu drücken.
3. Nach 20 Versuchen wird Ihr Ergebnis angezeigt.
Das Ergebnis verbessert sich jedesmal um einen Punkt, wenn es Ihnen gelang, die gleiche Nummer wie die angezeigt zu treffen; jeder Fehlschlag resultiert in einem Punktabzug. Wenn keine Taste angeschlagen wird ändert sich das Ergebnis nicht.

■ BEZUG

Reduzieren des Betrages von W in Zeile 20 ($W=10$, falls das Programm entsprechend der Angabe eingetippt wurde) reduziert die Anzahl der notwendigen Treffer. Versuchen Sie es.

■ TASTENOPERATION

1. **DEF A**

M o l e - B a s h i n g

Programmstart

3. 6

4

2. **ENTER**

6

Spielstart

4. 4

9

5. 9

3

Das Spiel wird auf gleiche Weise fortgesetzt

6. 5

Score=14

Das Endergebnis wird angezeigt.

7. **ENTER**

>

Programmende

■ SPEICHERINHALTE

A\$	
B, B\$	Angezeigte Zahl
C	Anzeigeposition (Zahl)
D\$	Zahleneingabe
F	
I	Schleifenzähler
J	Schleifenzähler
M	
T	Endergebnis
W	

■ ZEILENPLAN

Zeilen	Inhalt
10~ 20	Initialisierung
30~ 40	Titelanzeige
50~100	Zahlenanzeige
110~170	Verarbeiten der Tasteneingabe
180~200	Anzeige des Ergebnisses

■ PROGRAMMLISTE

```

10:"AY" CLS : CLEAR :
    WAIT 0: RANDOM
20:M=20:W=10
30:PRINT "Mole-Poshins"
40:A$= INKEY$ : IF A$="
    " THEN 40
50:FOR I=1 TO M
60:CLS
70:B= RND 10-1
80:C= RND 48-1
90:B$= STR$ B
100:CURSOR C: PRINT B$
110:F=0
120:FOR J=1 TO N
130:D$="":D$= INKEY$
140:IF F=0 AND B$=D$
    LET T=T+1:F=1:J=W:
    GOTO 160
150:IF F=0 AND D$(<>)B$
    AND D$(<>)" LET T=T-1
    :F=1
160:NEXT J
170:NEXT I
180:CLS : WAIT
190:PRINT "Score="T
200:CLS : END

```


INDEX

A

- Abkürzung 111
- Arbeitskommandos 88—89
- Ausdrücke 93, 94—97
- Automatische Abschaltung 5

B

BASIC

- Anweisungen 99
- Begriffe und Ausdrücke 70—98
- Kommandos 100—101
- Programme 99
- BATT-Symbol 12
- Batterien 3, 12, 14—16
- Betriebsarten 35
- Binäre Zahlen 96
- BUSY-Symbol 10
- Byte 71

C

- CAL-Symbol 11

D

- Dateinamen 86
- DBL-Symbol 11
- DEF-Symbol 10
- DEG-Symbol 11
- Display 10—12

E

- Eingabefehler 290—291
- Eingebaute Speicher 273—276
- E-Symbol 12

F

- Fehlermeldung 117—121, 288—291
- Fehlersuche 308
- Feld-Variablen 75—82
 - Zweidimensional 79—81
- Funktionen 98

G

- GRAD-Symbol 11
- Genauigkeit (Einfache/Doppelte) 59—62

K

- Kassetten Interface 25—28
- Kommandos (Basic) 100—101
- Kommandos (Direkte) 101
- Kompatibilität 311—313
- Kontrast 2
- Kursor 10

M

- M-Symbol 11
- Magnetband 29—30

N

Numerische Masken 295

P

Periphere Geräte (Übersicht) 23

Pflege 314

PRINT-Symbol 11

PROgramm-Modus 117

Programme und Dateien 86

Programmieren (Fehlersuche) 117—121

PRO-Symbol 11

R

RAD-Symbol 11

RAM-Karte 18—22

Rechenbetrieb 36—39

Rechenfehler 48, 66—68, 290—291

Rechenlänge 62

Rechenoperationen

Komplexere 51—52

Mehrere Rechenschritte 54

Variablen 52—54

Verkettung 49—50

Wissenschaftliche 62

Reinigung 314

RUN-Symbol 11

Rücksetzschalter 3, 12—13

S

SHIFT-Symbol 11

Sicherungszelle 15

SML-Symbol 11

Speicherkapazität 285

Speicherkonfiguration 277

T

Tastatur 5—9

Tastenfunktionen (im BASIC)

302—306

Tastenfunktion Tabelle 307

Technische Daten 309—310

U

Umwandlungswerte CHR\$/ASCII

292—293

V

Variablen 71—85

Doppelte 82—85

Einfache 74—75

Feld-Variablen 75—79

Vorgegebene 73

W

Wechselstromadapter 27

Wissenschaftliche Notation

56—58

Z

Zeichencode-Tabelle 292—293

Zeichenfolgen-Konstanten 70

Zeilennummern 99—100

SHARP CORPORATION

OSAKA, JAPAN

©1987 SHARP CORPORATION

PRINTED IN JAPAN/IMPRIMÉ AU JAPON

7M 0.8-I(TINSG1123ECZZ)③