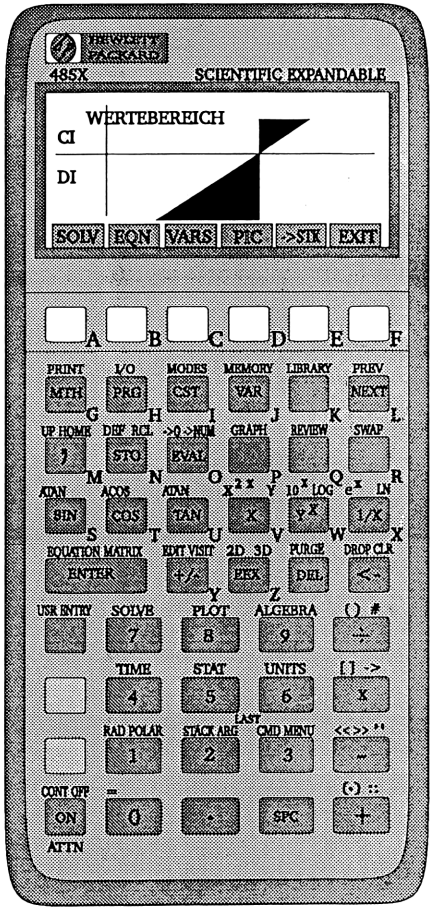
Programmierhandbuch
zum
Hewlett Packard HP-48SX
Pocket Computer



ISBN 3-89374-065-1

Robert Hübner

Fischel GmbH

Der neue **Taschencomputer HP-48SX** von Hewlett-Packard (HP) ist eine konsequente Weiterentwicklung und Synthese der großen Taschencomputermodelle HP-41 und HP-28. HP vereinigt damit in einzigartiger Weise Handlichkeit, Leistungsfähigkeit und Ausbaufähigkeit in einem Gerät. Mit diesem Taschencomputer dürfte sich Hewlett-Packard wieder einmal einen großen Vorsprung gegenüber seinen Konkurrenten gesichert haben.

Dieses Buch richtet sich sowohl an den Einsteiger als auch an den fortgeschrittenen Anwender des HP-48SX. Es ist mit seinen vielen Beispielen, Tips und wichtigen Hinweisen zur Programmierung eine Referenz für den professionellen Einsatz des Taschencomputers HP-48SX. Ich möchte Ihnen auch meine persönlichen Erfahrungen, welche sicher durch mein Informatikstudium geprägt sind, in diesem Buch mitteilen und hoffe ihnen damit einen leichteren Einstieg in die komplexe Programmierung des HP-48 zu geben.

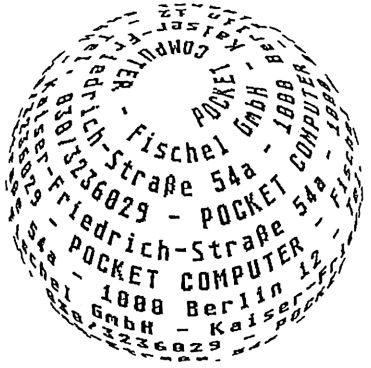
In diesem Zusammenhang möchte ich auch Thomas Hainzl für seine bereitwillige Hilfe und seiner zur Verfügung gestellten Zeit danken.

Nun wünsche ich Ihnen viel Erfolg beim Arbeiten mit diesem Buch und Ihrem Taschencomputer HP-48SX.

POCKET COMPUTER

FISCHEL GmbH
Zeitschrift für Taschencomputer

durch Information vorn
Kaiser-Friedrich-Straße 54a
1000 Berlin 12
Telefon (030)3236029
HRB 19396 Amtsgericht Charlottenburg



POCKET COMPUTER
Fischel GmbH
Kaiser-Friedrich-Straße 54a
1000 Berlin 12 - Tel. 030 / 323 60 29

Bankverbindung: Postgiroamt Berlin-West, Bankleitzahl 0010010, Kontonummer 461533-103

Öffnungszeiten: Montag-Freitag 10.00-18.00 Uhr, Samstag 10.00-14.00

Dieses Buch versucht Ihnen einen möglichst umfassenden Überblick über die Möglichkeiten des neuen Taschencomputers **Hewlett-Packard 48SX** zu geben.

Sie erlernen dabei von den Grundfunktionen bis zu komplexen Programmen aus verschiedenen Themenbereichen alles, was Ihnen eine gute Basis für den schnellen Umgang mit dem HP-48SX und für eine saubere Programmiertechnik gibt. Dabei wurde auch der Modularität und den Systemerweiterungen des HP-48SX genüge getan. Dies dürfte gerade im Hinblick auf die zukünftige Entwicklung im Personalcomputer-Bereich von entscheidender Bedeutung sein.
Differentiation und Integration numerischer und symbolischer Funktionen und Gleichungen, die Untersuchung eines Terms auf Nullstellen, Extremwerte, Schnittpunkte und seiner Achsenlage, die graphische Darstellung auch komplexer Funktionen und statistischer Werte und Tabellen und das vollständige Lösen von Differentialgleichungen sind nur einige Beispiele für die Leistungsfähigkeit dieser neuen Taschencomputer-Generation.
Außerdem sind dem Einsatz des HP-48SX durch seine Schnittstelle zum PC und APPLE-Macintosh und seine Steckplätze für ROM- und RAM-Karten fast keine Grenzen mehr gesetzt.

Das Buch gliedert sich in drei wichtige Themenbereiche.
Der erste Bereich umfaßt eine komplette Übersicht über die Befehle und Grundfunktionen des HP-48SX und gibt wichtige Hinweise über ihren optimalen Einsatz. Der zweite Teilbereich führt Sie in die Programmierung des HP-48SX ein und stellt Ihnen eine umfassende Programmbibliothek zur Verfügung. Mit dieser sind Sie in der Lage selbst schwierige und langwierige Probleme zu lösen. Außerdem ist es möglich die Programme als Units in eigene Anwendungsprogramme zu integrieren. Der dritte Themenbereich erklärt die Zusatzmodule des HP-48SX und Ihre Bedeutung. Dabei wird besonderes Augenmerk auf die Programmentwicklung und Programmarchivierung auf dem Personal-Computer gerichtet.
Die Basis für die im Buch enthaltenen Beispiele und Programme ist die Grundeinstellung des HP-48SX. Abweichungen werden jeweils abgegeben.

Der Benutzer dieses Buches sollte mit der Bedienung und der Programmierung des HP-48SX grundsätzlich vertraut sein. Die einzelnen Programmbefehle werden nur dort erklärt, wo es notwendig erscheint. Vergleichen Sie dazu das Benutzer- und das Referenzhandbuch zum HP-48SX.

Die Schreibweise der einzelnen Befehle erklärt sich wie folgt:

SOLV	Menüwahl des SOLV-Menüs.
[ON] oder ON	Tastatureingabe von ON.
EEX	Aufruf der Exponent-Funktion.
(HYP) oder *HYP*	Aufruf eines Menüpunktes über Softkey.
<-+	Umschalttaste links
+->	Umschalttaste rechts

Die Basis für die im Buch enthaltenen Programme und Beispiele ist die **Rechner-Grundeinstellung**. Abweichungen werden jeweils angegeben.

DER HP-48SX IM ÜBERBLICK

RECHNER GRUNDEINSTELLUNG.

Die normale Stackanzeige und die Reaktivierung des Tastenfeldes erreichen Sie durch das Drücken von ON.
Gegebenenfalls wiederholen Sie diese Prozedur.

TESTEN DES BETRIEBSSYSTEMS

SYSTEMSTOPP

Wenn Sie ein Programm anhalten möchten, das nicht auf das Drücken von **ON** reagiert, führen Sie einen Systemstopp durch.
Drücken und halten Sie **ON**, drücken Sie die dritte Softkey-Taste von links (C) und geben Sie **ON** wieder frei.

!ACHTUNG! Dieser Befehl LÖSCHT:
Alle unterbrochenen Programme
Alle lokalen Variablen
Das **CUSTOM**-Menü
Den Stack und alle rückgesicherten Elemente
Alle temporären Menüs
Es muß eine leere Stackanzeige erscheinen. Sollten unsinnige Zeichen abgebildet werden, so ist ein Memory-Reset durchzuführen.

MEMORY RESET

Um den gesamten Speicher zurückzusetzen und die Voreinstellungen des HP-48 SX

aufzurufen betätigen Sie ON, drücken Sie gleichzeitig noch die äußeren Softkey-Tasten (A,F).Lassen Sie alle drei Tasten los.

!ACHTUNG! Dieser Befehl führt einen Systemstopp durch!
Dieser Befehl LÖSCHT:

Alle Verzeichnisse und Benutzervariablen
Die benutzerdefinierte Tastatur
Alle geänderten Modi
Einen mit *MERGE* an eine RAM-Karte angebundenen Speicherbereich
Ein Tonsignal und die Anzeige "TRY TO RECOVER MEMORY?" (Speicherinhalt wieder herstellen?) werden ausgegeben.
Betätigen sie die Softkey YES, um den Speicherinhalt weitgehend zu rekonstruieren.

TASCHENCOMPUTER SELBSTTEST

Ist es nötig, den Taschencomputer auf Grund einer Fehlfunktion in den Urzustand zurückzusetzen und führt dies zu keinem Ergebnis, so kann ein Rechner-Selbsttest durchgeführt werden.
Drücken Sie ON und halten Sie die Taste gedrückt. Dann drücken Sie die dritte Softkey von links (C), und lassen Sie die Taste wieder los. Lassen Sie ON los.
Der Test ist fehlerfrei durchlaufen worden, wenn "IROM OK" und "IRAM OK "angezeigt werden.

TESTEN DES PORT-RAM

Durch den Test des Port-RAM können Sie feststellen, ob die Ports und die installierten Einsteckkarten fehlerfrei arbeiten.

Führen Sie folgende Schritte aus:

1. Überprüfen Sie ob die Einsteckkarten richtig in den Ports installiert sind.
2. Prüfen Sie , ob auf jeder Karte der Schalter in der Stellung "Schreiben/

Lesen'' steht.

3. Schalten Sie den Taschenrechner aus.
4. Drücken Sie **ON** und halten Sie die Taste fest. Drücken Sie die vierte Softkey von links
(D). Lassen Sie **ON** los. Eine vertikale Linie erscheint an beiden Seiten und in der Mitte des Anzeigefeldes.

Drücken Sie ▲ und lassen Sie die Taste wieder los.

Wenn der Test fehlerfrei durchlaufen wurde muß die Meldung "RAM1 OK" und/oder "RAM2 OK erscheinen.
Für jeden Port ohne Einsteckkarte bzw mit Karten-Lese- und Schreibschutz erscheint die Fehlermeldung ''RAMX 00002''.
Um zur normalen Rechnerfunktion zurückzukehren, muß ein Systemstopp durchgeführt werden.

SCHLEIFENTEST DES INFRAROTANSCHLUSSES.

Mit diesem Testprogramm werden die die Sende- und Empfangssensoren für die Infrarotübertragung geprüft.
Stellen Sie sicher, daß sich die Port-Abdeckung auf ihrem Platz befindet.

Um den Test zu starten, gehen Sie folgendermaßen vor:
Drücken und halten Sie **ON**.
Drücken Sie die vierte Softkey von links (D) und lassen Sie diese Taste wieder los.
Drücken Sie **EVAL.**

Es muß die Meldung ''IRLB OK'' erscheinen.
Um den Test zu beenden, muß ein Systemstopp durchgeführt werden.

SCHLEIFENTEST DES SERIELLEN ANSCHLUSSES.

Mit diesem Test wird die Funktion der Elektronik für die serielle Datenübertragung

des HP-48SX überprüft.
Dazu sind folgende Bedienungsschritte erforderlich:

1. Drücken Sie **ON** und halten Sie die Taste gedrückt.
2. Drücken Sie den vierten Softkey von links (**D**) und lassen Sie sie wieder los
3. Lassen Sie **ON** wieder los.
Ein vertikale Linie erscheint an beiden Seiten und in der Mitte des Anzeigefeldes
4. Verbinden Sie nun die beiden mittleren Stifte des vier-poligen seriellen Anschlusses (Kurzschluß).
Drücken Sie die Taste **PRG.**

In der Anzeige erscheint die Meldung "U_LB OK".
Erscheint die Meldung nicht ist der serielle Anschluß fehlerhaft.
Beenden Sie den Test mit einem Systemstop.

DIE EINGABEMODI DES HP-48SX

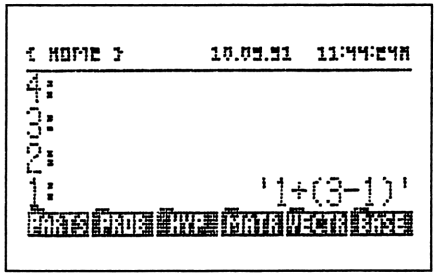
Ihr Taschencomputer stellt Ihnen sechs Eingabemodi zur Verfügung:

den unmittelbaren-,
den algebraischen-,
den Alpha-Eingabemodus.
den Eingabemodus für Matrizen
den Eingabemodus für Gleichungen und Funktionen
den Eingabemodus für Graphikobjekte

Der unmittelbare Eingabemodus dient zum Eingeben von Zahlen, Namen, Listen oder Feldern. Die Berechnung von 1+(3-1) wird in diesem Modus wie folgt durchgeführt:
1 ENTER 3 ENTER 1 ENTER - + Ergbnis: 3
Eine Rechenoperation wird nach der Eingabe der zugehörigen Objekte ausgeführt.
Zur Eingabe der Objekte kann auch ein Trennungszeichen verwendet werden.

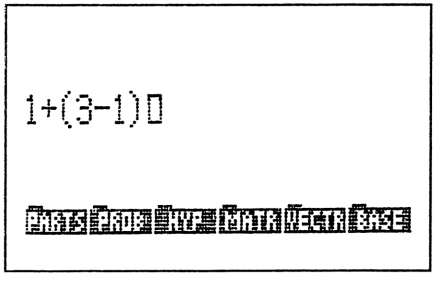
Beispiel:
1,3,1 -/+ oder 1 SPC 3 SPC 1 -/+

Der **algebraische Eingabemodus** dient zur Eingabe von Namen, Variablen und algebraischen Ausdrücken. Führen Sie die oben angegebene Berechnung wie folgt aus. Sie aktivieren den algebraischen Eingabemodus mit '.
' 1 + (3 - 1) ENTER
Das Ergebnis erhalten Sie durch eine Auswertung der Befehlszeile mit dem Befehl



EVAL oder **- > NUM.**

Der **algebraische Eingabemodus mit dem EquationWriter** ist zur Unmittelbaren graphischen Kontrolle einer Gleichung nützlich. Alle mathematischen Zeichen werden richtig auf dem Display dargestellt.
Der EquationWriter wird mit **<-┼ Equation** aufgerufen.
Die obige Rechenoperation sieht damit wie folgt aus:
<-┼ Equation 1 + <-┼ (3 - 1 > ENTER EVAL (-> NUM)

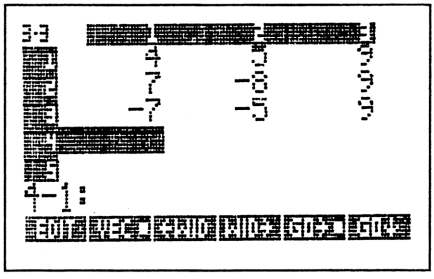


Sie können auch den Ausdruck als neue Gleichung definieren. Dies geschieht dadurch, daß Sie nach Eingabe der Gleichung in den EquationWriter **ENTER** und dann die Softkey *NEW* betätigen. Dann geben Sie den Namen der Gleichnung mit den Alpha-Tasten ein. Der HP 48 befindet sich dabei schon im Alpha-Eingabemodus. Betätigen Sie **ENTER** zum Übergeben der Gleichung in deren Variable.

Der **Alpha-Eingabemodus** dient zur Eingabe von Namen, Strings und Programmen. Dieser Modus wird durch die Tasten α, **┼->** **" "** oder **<-┼** **<< >>** aktiviert. Der Indikator α erscheint in der obersten Anzeigezeile. Zum Verlassen des Alpha-Modus betätigen Sie α ein zweites Mal. Sind Sie im String- bzw. im Programmeingabe-Modus verlassen Sie diesen bitte mit **ENTER**.

Der **Eingabemodus für Graphikobjekte** wird dazu verwendet, eine Pixelgraphik im Rechner zu erzeugen oder zu bearbeiten. Die Graphikumgebung wird mit dem Befehl: **<-┼ GRAPH** aufgerufen. Mit **STO** kann das fertige Grphikobjekt in die Befehlszeile kopiert werden.

Zum Schluß gibt es noch den **Eingabemodus für Matrizen mit Hilfe des MatrixWriters.**
Im MatrixWriter erhalten Sie dazu ein Arbeitsblatt, daß ähnlich aufgebaut ist wie in einer Tabellkalkulation. Es ist in nummerierte Zeilen und Spalten aufgeteilt. Ausgangspunkt ist Spalte 1, Zeile 1. Sie können Sich innerhalb des Arbeitsblattes mit Hilfe der Softkeys bewegen. Beim Verlassen des MatrixWriters mit **ENTER** nach der Dateneingabe erhalten Sie die eingegebenen Werte als Matrix im Stack.
Mit dieser Funktion können Sie ganz einfach große Matrizen erstellen und editieren. Danach können Sie alle notwendigen Rechenoperationen mit diesem Matrix durchführen.



Anmerkung: +-> **ENTRY** schaltet zwischen dem unmittelbaren, dem Programmeingabe- und dem algebraischen Eingabemodus um.

UNMITTELBAR------PROGRAMMEINGABE--+-> ENTRY--> ALGEBRAISCH
(Direkteingabe)

DER UMGANG MIT DEM STACK

Der dynamische Stack des HP-48SX hat den Vorteil, daß er keine feste Größe besitzt. Durch die Eingabe von Objekten werden immer neue Ebenen erzeugt; d.h., daß Sie alle eingegebenen Daten immer für Berechnungen zur Verfügung haben. Eine obere Grenze wird nur durch den freien Speicherplatz gesetzt.
Beachten Sie, daß Sie nicht mehr benötigte Objekte aus dem Stack löschen, um nicht einen beachtlichen Teil des Speichers damit zu belegen.

ANZEIGEN GROSSER OBJEKTE

Die Anzeige des HP-48SX kann 8 Zeilen und 22 Zeichen pro Zeile darstellen. Sollte ien Objekt mehr als 23 Zeichen pro Zeile beinhalten, werden die ersten Zeichen, bei Eingabe, nach links verschoben. Nach einer Übernahme des Objekts in den Stack erscheinen, falls der Multiline-Modus für Ebene 1 nicht aktiviert ist, nur eine Zeile und 3 Punkte am rechten Zeilen- bzw. Anzeigerand.
Zur Ansicht der gesamten Zeile in Ebene 1 aktivieren Sie den Multline-Modus oder betätigen Sie die Tasten <-+ **EDIT** oder +-> **VISIT**. Sie können das Objekt aber auch in den zu diesem Objekt gehörigen Anzeigemodus kopieren. D.h., daß Sie eine Matrix in den MatrixWriter und eine Gleichung oder Funktion in den EquationWriter übergeben können.
Dies geschieht mit der Taste ▼.
Sollte sich ein zu betrachtendes Objekt außerhab des Anzeigefensters befinden, können Sie dieses Fenster mit dem Befehl ^ nach oben verschieben. Mit diesem Befehl wird der Rollmodus (Scroll-Lock) für den Stack aktiviert. Die Tasten ▲,▼ werden zum Blättern verwendet.

Diese Funktion kann auch mit der Softkey *¡STK* aufgerufen werden.

EDITIEREN

Mit dem HP-48SX haben Sie die Möglichkeit gespeicherte oder im Stack abgelegte Objekte nachträglich in die Befehlszeile zu übernehmen und abzuändern. Ihr Taschencomputer stellt Ihnen dazu mehrere Editierfunktionen zur Verfügung.

<-┼ EDIT	Ebene 1 editieren
┼-> VISIT	Ebene 1 editieren
n ┼-> VISIT	Ebene n editieren
'DIFF' **┼-> VISIT**	Programm oder Variable 'DIFF' editieren
<-┼ Equation	Gleichung oder Funktion an den EquationWriter übergeben.
┼-> Matrix	Matrizen oder Vektoren an den MatrixWriter geben.
▼	Objekt in Ebene 1 in die entsprechende Umgebung übergeben.
-> STK	(Softkey) kopiert die gewählte Gleichung aus einer Stackebene nach Ebene 1.

EDIT∑
Kopiert Statistikdaten von *∑DAT* in den MatrixWriter.
! **ENTER** nach **EDIT** bzw. **VISIT** gibt das geänderte Objekt in die Variable oder in die Befehlszeile zurück. Soll der Editiervorgang abgebrochen werden, beträtigen Sie **ON**. Das Objekt wird unverändert übernommen.

DIE OBJEKTTYPEN DES HP-48SX

0	Reelle Zahl	z.B. 0 1 1/3 -8 ...
1	Komplexe Zahl	(a,jb)
2	Zeichenkette (String)	"Hallo"
3	Reeller Vektor oder Matrix	[[1 2 3 1/3 [-8 ...
4	Komplexer Vekt. oder Matrix	[[(1,2) (0.57,90) ...
5	Liste	{
6	Globaler Name	'DIFF'

7	Lokaler Name	nc
8	Programm	<< DUP 3 ROLL SWAP >>
9	Algebraisches Objekt	'2*y^2-A'
10	Binärwert	#1000100000001b
11	Grafik	GRAPHIC9x15
12	Markiertes Objekt	:tes:zähler1
13	Objekt mit Einheit	(58.2_kgm/s^2)/(2.5_m/s)
14	XLIB-Name	'MATHLIB'
15	Verzeichnis	'SYST'
16	Bibliothek	:1:268
17	Ausgelagertes Objekt	'O' 2 ...
18	Eingebaute Funktion	+/-
19	Eingebauter Befehl	(Softkey) LIBS

TIPS UND ÜBUNGEN FÜR ANFÄNGER UND FORTGESCHRITTENE

DIE WICHTIGEN MODULE DES HP-48SX

DER TERMINKALENDER

1. Einstellen und Anzeigen der Systemuhr

Die Systemuhr des HP-48SX können Sie im Menü <-┼ **Time** *Set* einstellen. Diese Einstellung beinhaltet Datum und Uhrzeit.
Legen Sie das Datums- und Uhrzeit-Anzeigeformat fest.
Mit der Softkey *12/24* kann zwischen 12 und 24 Stunden-Format umgeschaltet werden. Mit der Softkey **M/D** können Sie das Datumsformat von Monat/Tag/Jahr auf Tag.Monat.Jahr umschalten und umgekehrt.

Stellen Sie zuerst das Datum wie folgt ein:
<-┼ **TIME** *SET* 20.01.1991 -> **DAT**
Stellen Sie dann die Uhr ein:
5.07 -> *TIM*
Sie können hier noch mit *A/PM* zwischen AM und PM (vor- und nachmittags) wechseln.
Nun ist die Systemuhr eingestellt.
Sollten Sie die Uhrzeit einemal korrigieren müssen, so verwenden Sie das Menü
<-┼ **TIME** *ADJST.*

2. Eingabe eines Termins

Der HP-48SX kann zwei Arten von Terminen verwalten:
A. Einen Termin mit Ausgabe eines Alarmsignals und eines Begleittextes.
B. Einen Termin zur Steuerung des Taschencomputers.

Um einen Termin **nach dem Muster A** einzugeben, gehen Sie wie folgt vor:

Aktivieren Sie das Menü TIME mit <-┼ **TIME.**
Setzen Sie am 3.12.1991 um 8.30 Uhr einen Besprechungstermin.
Geben Sie den Termin ein:
<-┼ **TIME** *ALRM* 8.30 > *TIME* 03.121991 > *DATE* ┼-> ""Besprechung Herr Müller *EXEC.*
Legen Sie ein Terminerinnerungs-Intervall fest von 1 Woche fest:
RPT 1 **WEEK**
Bestätigen Sie den Termin mit *SET.* Der nächste aktive Termin wird nun angezeigt.

Wenn Sie einen Termin **nach Muster B** eingeben wollen, d.h. zum angegeben Zeitpunkt wird ein Objekt ausgewertet, führen sie folgende Schritte durch:

Stellen Sie Datum und Uhrzeit wie bei einem normalen Erinnerungs-Termin ein.
Geben Sie dann statt einen Text das auszuführende Objekt ein und drücken *EXEC.*
Drücken Sie *SET,* um den Steuerungstermin zu bestätigen.

Bestätigen und Speichern von Terminen.

Wird ein Erinnerungstermin fällig, ertönt für etwa 15 Sekunden ein Alarmsignal und der Termintext wird angezeigt. Diesen Termin können Sie bestätigen (löschen), indem Sie eine beliebige Taste drücken, solange das Alarmsignal hörbar ist.handelt es sich um einen Wiederholungstermin, wird er neu angesetzt.
Geschieht dies nicht bleibt die Terminerinnerung unbestätigt. D.h., der Termintext wird aus der Anzeige gelöscht, der Indikator ((˙)) bleibt erhalten und der Termin ist überfällig.

Sie können überfällige Termine folgendermaßen löschen:
Geben Sie <-┼ **TIME** ein, um den ältesten überfälligen Termin anzuzeigen.
Drücken Sie ACK, um den Termin aus der Terminliste zu löschen. Existieren mehrer überfällige Termine, so wird der nächste angezeigt und kann mit *ACK* gelöscht werden.
Mit dem Befehl *ACKA* können alle überfälligen Termine gelöscht werden.
Wollen Sie bestätigte, nicht zu wiederholende Terminerinnerungen vor dem Löschen sichern, so setzen Sie Flag -44.
Wollen Sie unbestätigte zu wiederholende Termine nach Ihrer Fälligkeit automatisch löschen lassen setzen Sie Flag -43.

Bei einem fälligen **Steuerungstermin**, wird eine Kopie des Terminindex (reelle Zahl = Rangordnung = Termin-Identifizierung) in Ebene 1 des Stacks gelegt und das ihm zugeordnete Objekt (Programm) ausgeführt.
Ein fälliger Steuerungstermin wird immer als bestätigt betrachtet, d.h. er bleibt immer in der Terminliste gespeichert.
Um das Akkustiksignal für Termine abzuschalten, muß Flag -57 gesetzt werden.

Alle Termine können mit <-┼ **TIME** *CAT* bzw. ┼-> **TIME** angezeigt werden.
Im Terminkatalog können Sie auf kein anderes Menü zugreifen und keine Stack-Operationen durchführen.

Einen Termin können Sie mit den **Cursortasten** anwählen, mit **VIEW** ansehen und mit **EDIT** bearbeiten.

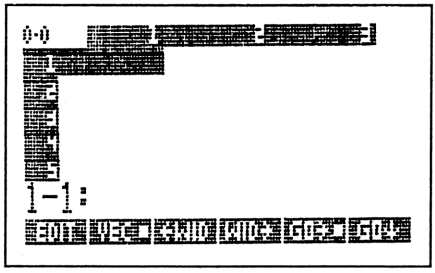
Verwendung von Terminen mit Programmen

Ein Termin wird in Programmen in folgender Form definiert **{Datum Uhrzeit Aktion Wiederholung}**. Aktion ist in diesem Fall das auszuführende Programm, Wiederholung ist das Wiederholungsintervall in Uhrzyklen (TICKS). Ein Uhrzyklus entspricht einer 1/8192 Sekunde.

Beispiel:
Ein Programm 'ADD' soll zu einem bestimmten Termin (03.12.1991 um 8.30) ausgeführt werden.
Setzen Sie den Termin, wie oben erklärt auf diesen Zeitpunkt.
Geben Sie als auszuführendes Objekt 'ADD' an und drücken Sie *EXEC* und *SET*.
Das Programm 'ADD' wird dann zu diesem Zeitpunkt ausgeführt.

DER MATRIXWRITER

Der MatrixWriter dient zur Eingabe und zum Bearbeiten von zweidimensionalen und dreidimensionalen Feldern (Vektoren oder Matrizen).
Der MatrixWriter wird mit +-> **MATRIX** gestartet.



Der Inhalt des MatrixWriters wird durch sogenannte Zellen definiert. Eine Zelle wird durch den Index bestimmt. Der Index wird im Format Zeile - Spalte angegeben.

Matrizen werden durch n Zeilen und n Spalten angegben.
Vektoren werden durch eine Zeile (Zeile 1) und n-Spalten definiert.

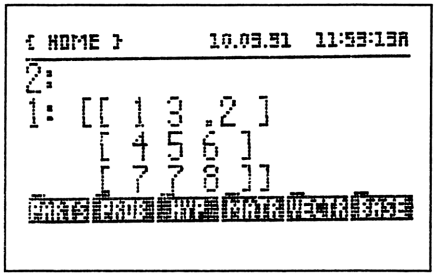
Matritzenelemente werden in der Befehlszeile eingegeben. Sie wird durch das Betätigen einer Zifferntaste automatisch aktiviert.
Mit **ENTER** nach der Dateneingabe wird das Feld in den Stack übertragen. Eine Matrix wird dann durch [[ ]] und ein Vektor durch [ ] gekennzeichnet.

Beispiele:

Matrix:

[[1 3 1/5]
[4 5 6]
[7 7 8]]

Eingabe:
+-> MATRIX
1 SPC 3 SPC 1 SPC 5 SPC / ENTER V
4 SPC 5 SPC 6 ENTER
7 SPC 7 SPC 8 ENTER ENTER

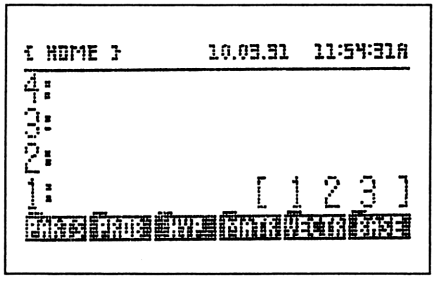


MatrixWriter nach der Dateneingabe

Die Matrix wird in den Stack übergeben.

Vektor: [1 2 3]

Eingabe:
+-> **MATRIX** *VEC* (VEC')
1 SPC 2 SPC 3 ENTER ENTER



Vektor im MatrixWriter

Der Vektor wird in den Stack übergeben.

Die einzelnen Zellen lassen sich mit < -*WID* verkleinern und mit *WID*-> vergrößern.

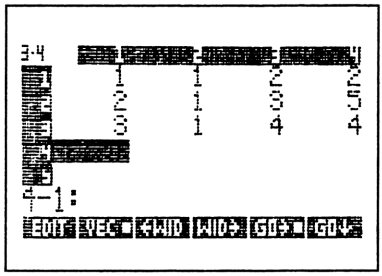
Elemente im MatrixWriter nachträglich ändern.

Übergeben Sie eine Matrix in den MatrixWriter
Beispiel: [[1 2 2]
[2 3 5]
[3 4 4]]

▼ Übergibt die Matrix aus dem Stack an den MatrixWriter.

Editieren Sie das Element 2-3 und ändern Sie es von 5 in 3:
▼ -> -> *EDIT* 3 ENTER

Fügen Sie zwischen den Spalten 1 und 2 eine neue mit den Werten 1 1 1 ein.
-> NXT +*COL*▲▲NXT *GO* 1 SPC 1 SPC 1 SPC ENTER GO-> ENTER



Geänderte Matrix im MatrixWriter

Komplexe Felder

Beispiel: [[1-2i 2]
[i 3+i]]

+-> MATRIX
<-+( ) 1 SPC 2 +/- ENTER
<-+( ) 2 ENTER _
<-+( ) 0 SPC 1 ENTER <-+( ) 3 SPC 1 ENTER

WID-> *WID->* verbreitert die Zellen, um die ganze Zahl sehen zu können.
ENTER übergibt die Matrix in den Stack.

Variablen mit definiertem Inhalt

Beispiel:
'A' = (3 4) = 3+4i
Matrix = [[1 i]
[2 A]]**+-> MATRIX**

1 SPC <-+ ( ) 0 SPC 1 ENTER ▼
2 SPC A ENTER

Die Variable A wird bei der Übergabe an die Matrix ausgewertet.
ENTER

DER EQUATIONWRITER

Der Taschencomputer HP-48SX besitzt eine sehr gute Funktion zum Bearbeiten und Darstellen von Funktionen und Gleichungen - den EquationWriter.
Dieses Modul gestattet es, eine Gleichung oder Funktion am Display wie auf dem *Papier* einzugeben und sichtbar zu machen. Darstellbar sind alle mathematischen Zeichen des HP-48SX.

Die Funktion

$$F(x)=\int_0^{x1} 1+\operatorname{asinh}(x)-\frac{1}{x}\,dx$$

läßt sich mit Hilfe des EquationWriters ganz einfach eingeben:

Aktivieren Sie den EquationWriter mit den Tasten **<-+ EQUATION.**
Geben Sie folgendes ein:
<-+ α F <-+ ( ) x ▷ <-+ = +-> ∫ 0 > <-+ α x 1 ▷ 1 + MTH HYP ASINH <-+ α x ▷ ▷ - 1 <-+ α α x ▷ ▷ x

Siehe Bild 8

ENTER
Die Funtion wird an den Stack übergeben. Sie hat nun die Form:
'F(x)=∫(0,x1,∫(1+ASINH(x))-1/x,x)'

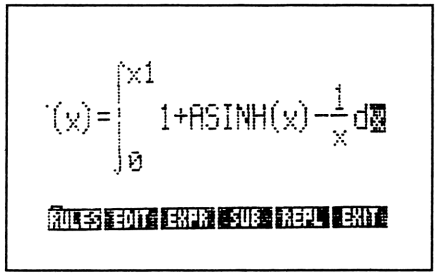


Bild 8 Gleichung im EquationWriter

Falls Sie eine falsche Eingabe gemacht haben, können Sie diese mit der <- (BACKSPACE) Taste korrigieren.
Auf die oben beschriebene Weise können Sie z.B. auch Summen, Brüche, Potenzen und Ableitungen (d/dx) im Display darstellen. Auch Funktionen oder Ausdrücke mit Einheiten sind möglich.

Z.B. 4.36 kgm/s²
Eingabe:
<-┼ EQUATION 4 . 3 6 _ <-┼ UNITS *MASS KG* *** <-┼ UNITS** *LENG M* **/ <-┼ UNITS** *TIME* **S Y^x 2 ▶ ▶ ENTER**

Wenn Sie algebraische Objekte und Objekte mit Einheiten im EquationWriter darstellen wollen, geben Sie das Objekt oder den Variablen-Namen in Ebene 1 ein und betätigen ∇ (+-> ∇ für eine Variable). Das Objekt wird damit in den EquationWriter übernommen und der Cursor an das Ende des Ausdrucks gesetzt.

Bei Ausdrücken, welche größer als das Display sind, drücken Sie **<-┼ GRAPH** und **◀**, damit der Ausdruck nach links (an das Ende) gerollt wird.

!Achtung! Der EquationWriter kann bei komplexen Ausdrücken einige Zeit für den Aufbau der Graphik benötigen.

WICHTIGE FUNKTIONEN DES EQUATIONWRITERS

<-┼ EDIT kopiert die Gleichung des EquationWriters zur schnelleren Bearbeitung in die Befehlszeile. Der Ausdruck kann bearbeitet werden und danach mit ENTER an den EquationWriter zurück gegeben werden

◁ aktiviert die Auswahlumgebung

Der Cursor hebt das letzte Objekt im Ausdruck hervor. Positionieren Sie den Cursor auf dem auszuwählenden Element und betätigen Sie *EXPR*. Verfahren Sie so mit allen Elementen die Sie in die Befehlszeile kopieren möchten. Mit *EDIT* wird der hervorgehobene Teilausdruck in die Befehlszeile kopiert und Sie können Ihn ändern. Mit **ENTER** wird dieser dann an den EquationWriter zurückgegeben.
Sollten Sie bei der Hervorhebung mit dem Auswahlcursor einen Fehler gemacht haben betätigen Sie *EXPR* ein zweites Mal, um die Hervorhebung zu löschen. Mit Exit kann die Auswahlumgebung ganz verlassen werden.

┼-> RCL fügt ein Objekt aus Ebene 1 in die aktuelle Gleichung und an der momentanen Cursorposition ein. Dabei werden alle Begrenzungszeichen automatisch entfernt.

REPL ersetzt den mit dem Auswahlmenü hervorgehobenen Teilausdruck durch ein Objekt in Ebene 1.

▶ beendet einen Teilausdruck.

SPC fügt das aktuelle Trennzeichen in den Ausdruck ein.

EVAL löst die Gleichung auf und verläßt den EquationWriter.

<-┼ GRAPH aktiviert den Rollmodus.

STO übergibt den Ausdruck als Graphikobjekt in den Stack.

+-> **CLR** löscht den aktuellen Ausdruck

+-> "" übergibt den Ausdruck als Zeichenkette an den Stack.

<-+ {} implizite Klammern Ein/Aus. Diese Funktion gestattet es z.B. Poten zen direkt einzugeben.

Bei implizite Klammern Aus (OFF) wird y^2 mit y y^x 2 eingetippt.

GRAFIK AUF DEM HP-48SX

DIE GRAFIK-UMGEBUNG DES HP-48SX

Die Grafik-Umgebung des HP-48SX erlaubt es auf relativ einfache Weise graphische Objekte (Bilder) auf dem HP-48 zu entwefen, zu ändern und abzuspeichern.
Die Grafik-Umgebung wird mit <-+ **GRAPH** aufgerufen.
Das Arbeiten in der Grafik-Umgebung erzeugt immer ein Grafikobjekt mit dem Namen PICT.
Aus der Grafik-Umgebung kann mit **STO** das Grafikobjekt an den STACK übergeben werden.
Ein fertiges Grafikobjekt kann mit -> *LCD* angezeigt und mit ▼ in die Grafik-Umgebung gebracht werden.

Die Organisation des Displays.

Die Anzeige des HP-48SX Taschencomputers hat eine Auflösung von 130 * 63 Pixel und ist aufgeteilt in 8 Zeilen mit 22 Zeichen. Ein Zeichen ist in der Regel 5 Pixel breit, der Zeichenabstand beträgt ein Pixel.
Zeichen werden durch die Angabe von Benutzerkoordinaten (Zeichenkoordinaten) am Display dargestellt. Das Format der Benutzerkoordinaten ist eine komplexe Zahl (Spalte Zeile). Diese Koordinaten werden durch die in PPAR definierten Parameter beeinflußt.

! Achtung! In der Grundeinstellung des Rechners ist der Mittelpunkt der Anzeige als Koordinate (0 0) definiert. Das Display hat also die Eckpunkte: (-6.5 3.2), (6.5 3.2), (-6.5 -3.1), (6.5 -3.1)!

Die Plotvariable **PPAR** enthält eine Liste mit folgenden Objekten:
{(Spaltemin Zeilemin) (Spaltemax Zeilemax) indep res axes ptype depend}.

(Spaltemin Zeilemin) Koordinaten der linken unteren Ecke des Displays.
(Spaltenmax Zeilenmax) Koordinaten der rechten oberen Ecke des Displays.
Diese beiden Koordinaten verändern automatisch den Achsen-Schnittpunkt.

indep	unabhängige Plotvariable (Variable oder Liste mit Namen und zwei reellen Zahlen für den Plotbereich).
res	Auflösung, eine reelle oder binäre Zahl, die den Pixelab stand des Graphen angibt.
axes	komplexe Zahl, welche die Koordinaten des Achsenschnitt punktes oder Liste mit Koordinaten und Achsenbeschrif tungen.
P-Typ	gibt den Plot-Typ an.
Depend	abhängige Plotvariable (Variable oder Liste mit Namen und zwei reellen Zahlen für den Plotbereich)

Die Plotparameterliste PPAR hat in der **Grundeinstellung** des Taschencomputers folgenden Inhalt:
{(-6.5 -3.1) (6.5 3.2) X 0 (0 0) FUNKTION Y }

!Achtung! Der Befehl RESET setzt den Inhalt von PPAR (außer dem Plottyp) auf die Standardwerte zurück.
RESET löscht auch PICT und stellt seine ursprüngliche Größe wieder her.

Pixel können mit der Angabe ihrer Pixelkoordinaten angesprochen werden. Pixelkoordinaten werden durch eine Liste mit zwei ganzzahligen Binärwerten {#Spalte #Zeile} angegeben.
Die **Standard-Pixelkoordinaten der Eckpunkte** sind:
{#0 #0} {#130 #0} {#0 #63} {#130 #63}.

Beispiel zur Verwendung der Graphik-Umgebung des HP-48SX zum Malen eines Bildes.

Nehmen wir an, Sie wollen ein Haus und die Sonne malen. Verwenden Sie dazu die Graphik-Umgebung des HP-48SX. Das Bild soll in etwa so aussehen:

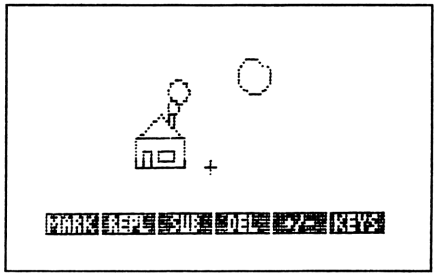


BILD 9 HAUS, SONNE in der Grafikumgebung

Gehen Sie dazu wie folgt vor:

1. Graphik-Umgebung mit **<-┼ GRAPH** aufrufen.
2. Zeichenmenü anwählen: **NXT**
3. Positionieren Sie den Cursor an den Anfangspunkt der Zeichnung. Z.B. 30 mal ◄ und 10 mal ▼
4. Beginnen Sie damit das Haus zu zeichnen:

BOX setzt die Zeichnemarke für einen Rahmen.
10 mal ▲ und 20 mal ► positioniert den Cursor auf die gegeüberliegende Ecke des Rahmens.
BOX zeichnet den Rahmen.
NXT *MARK* **NXT** 8 mal ▲ und neun mal ◄ *TLINE* setzt die erste Dachschräge.
NXT *MARK* 10 mal ◄ und 8 mal ▼ **NXT** *LINE* zeichnet die zweite Dachschräge.
10 mal ▼ und 3 mal ► **NXT** *MARK* **NXT NXT** *DOT+* 6 mal ▲ 3 mal ► und 6 mal ▼ *DOT+* zeichnet eine Tür.
3 mal ► 3 mal ▲ **NXT** *MARK* 3 mal ▲ 6 mal ► **NXT NXT** *BOX* zeichnet ein Fenster.
8 mal ▲ *DOT+* 4 mal ▲ 2 mal ◄ 3 mal ▼ *DOT+* zeichnet einen Schornstein.
6 mal ▲ 2 mal ► **NXT** *MARK* **<-┼ PREV** 2 mal ► *CIRCL* 6 mal ▲ **NXT** *MARK*

<-+ **PREV** 4 mal ► *CIRCL* erzeugt Rauchwolken.
5. Zeichnen Sie die Sonne:
5 mal ▲25 mal ► **NXT** * 6 mal ▲, <-+ **PREV** *CIRCL* malt die Sonne.
Sie können selbst noch beliebige Objekte im Display zeichnen.
Fehlgesetzte Punkte können mit *DOT*- oder *DEL* (Bereich Marke-Cursor) gelöscht werden.

6. Übergeben Sie die Graphik mit **STO** an den Stack. Betätigen Sie **ON** zum Verlassen der Grafik-Umgebung. Die Beschreibung des Graphikobjekts (Graphik 131 x 64) ist nun in Ebene 1 sichtbar und kann mit -> *LCD* angesehen bzw. mit der Angabe eines Namens und **STO** abgespeichert werden. Damit kann die Graphik zur späteren Weiterverwendung dienen.

Durch Überlagern mehrerer gesspeicherter Grafikobjekte können auch bewegte Grafiken erzeugt werden. Dies wird in einem späteren Kapitel erklärt.

BENUTZERANPASSUNG DES HP-48 TASCHENKOMPUTERS

BENUTZERMENÜS ERSTELLEN

Das CUSTOM-Menü

Das Menü **CST** (CUSTOM) beinhaltet vom Anwender selbst zu gestaltende Menüfelder. In der Rechner-Grundeinstellung ist dieses Menü leer.
Das Menü **CST** wird in einer Variablen mit dem Namen *CST* abgelegt.
Diese Variable ist eine reservierte Variable (fixiert auf nur diese eine Verwendung)

Die Variable *CST* hat folgenden Aufbau:

1. { ausführbare Variable }
2. { Befehl }
3. { "Menü-Feld" Objekt }

Objekte in **CST** besitzen den gleichen Funktionsumfang wie Objekte in eingebauten (internen) Menüs des HP-48 Taschencomputers. Objekte können z.B Programme, Listen, Bibliotheken, Strings usw. sein.

Das Menü **CST** wird nach der Eingabe der oben genannten Liste entweder mit +-> **MODES** *MENU* oder direkt durch die Speicherung der Liste in der Variablen 'CST' erzeugt.
Das Menü **CST** kann durch das Entfernen der Variablen CST (z.B. mit PURGE) gelöscht werden.

Gestalten des Menüs **CST**

Nehmen wir an, Sie wollen ein festes Benutzermenü mit folgender Struktur erzeugen:

CST *----SGL--PROG--TSM--ZE--PR1*

Dabei sind die Inhalte der Variablen wie folgt definiert:

SGL = Programm SEQ (Im VAR-Hauptverzeichnis speichern), speichert ein Objekt in eine Variable
SEQ: « CLLCD "Geben Sie den Namen ein:" PROMT -> n « n STO » »

PROG = Wechselt in das Menü VAR « 2 MENU »

TSM = Aufruf einer Einheit TSM: 1_TSM = 1 Standart-Monat '1_30*d'

ZE = Programm zum Aufruf der Grafikumgebung « GRAPH »

PR1 = Eingebauter Befehl zum Drucken der ersten Stackebene

Führen Sie folgende Schritte aus, um das Menü CST zu erzeugen:

Erzeugen Sie zuerst folgende Liste:
{ { "SGL" «2 MENU 'SEQ' EVAL» } { 'PROG" «2 MENU» } "TSM" { "ZE" «GRAPH» } PR1 }

Eingabe: **<-+ {} +-> " " α α S G L α ▶ SPC <-+ «» 2 +-> MODES** *MENU* **' α α S E Q α ▶ EVAL ▶ ▶ ▶ ▶ SPC <-+ {} +-> "" α α P R O G α ▶ SPC <-+ «» 2** *MENU* **▶ ▶ ▶ ▶ SPC +-> " " α α T S M α ▶ SPC <-+ {} +-> " " α α ZE α ▶ SPC <-+ «» <-+** *GRAPH* **▶ ▶ ▶ ▶ SPC <-+ PRINT** *PR1* **ENTER**

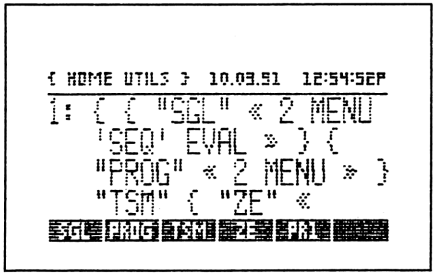


Bild 10 Erzeugtes Benutzermenü

Erzeugen Sie das Benutzermenü: **+-> MODES** *MENU*
Dabei wird die Variable CST im aktuellen Directory abgelegt.

Umschalttasten für ein Menüfeld definieren.

Dabei hat CST für ein Menüfeld folgenden Aufbau:
{ { "Menü-Feld" { Aktion normal Aktion links Aktion rechts } } }

Aktion kann z.B. ein Programm, eine Liste, ein String, usw. sein.
Aktion normal ist die Menüanwahl durch einfaches Betätigen der Menütaste.
Aktion links ist die Menüanwahl mit der linken Umschalttaste.
Aktion rechts ist die Menüanwahl mit der rechren Umschalttaste.

Erzeugen mehrerer Benutzermenüs.

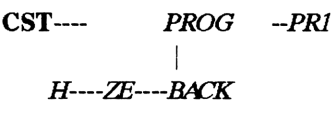
Sie können auf dem HP-48SX meherere unabhängige Benutzermenüs in einem oder mehreren Verzeichnissen erzeugen. Dazu schreiben Sie für jedes Menü eine Definitionsliste und speichern diese unter einem aussagekräftigen Namen ab (z.B. CST1, CST2, MEN1, MEN2, usw.).
Diese Variablen können entweder nur die für das entsprechende Menü spezifische Liste oder ein Programm zum Erzeugen eines Menüs aus dieser Liste beinhalten. Enthält die Variable nur die Definitionsliste muß man durch Betätigen der Variablen-Softkey die Liste in den Satck zurückgeben und mit dem Befehl **+-> MODES** *MENU* das Menü erzeugen. Mit einem Programm bleibt einem dieser Schritt erspart.

Ein typisches Programm zum Erzeugen eines Benutzermenüs sieht so aus:
« 'CST' PURGE {{ "STAT" ∑DAT }{ "LISTE" LS } GRAPH } MENU »

So können Sie in bestimmten Verzeichnissen (Directories) bestimmte Menüs ablegen. Theoretisch bedeutet dies, daß Sie Ihre Anwendungsprogramme oder Daten nicht in leicht einsehbaren Variablen abspeichern müssen, sondern diese in einer Menüstruktur definieren. Das ist zwar anfangs etwas umständlich, trägt aber auch zur Datensicherheit bei.

Erzeugen eines Untermenüs in CST

Erzeugen Sie folgende Menüstruktur:



PROG = Untermenü des Menüs CST
PR1 = Inhalt von Stackebene 1 drucken (Befehl)
H = String "Hallo"
ZE = Aufruf der Grafik-Umgebung
BACK = zurück zum Hauptmenü

Die spezifische Liste lautet wie folgt:
{ { "PROG" « { { "H" "HALLO" } { "ZE" «GRAPH» } { "BACK" «CST1 MENU» } } MENU » } PR1 }

Löschen Sie die alte Variable CST. Speichern Sie die obige Definitionsliste in der Variablen CST1. Rufen Sie mit CST1 die Liste in den Stack zurück. Betätigen Sie **+-> MODES** *MENU* zum Erzeugen des neuen Benutzermenüs.
Wie Sie erkennen können, erhalten Sie jetzt die beiden Menüfelder *PROG* und *PR1*. Wenn Sie jetzt die Softkey *PROG* betätigen erhalten Sie ein neues Menü mit den Feldern *H,ZE,BACK*. Mit *BACK* erhalten Sie wieder das ursprüngliche Menü zurück. Dies geschieht, indem Sie die in CST1 gespeicherte Definitionsliste in den Stack zurückgeben und mit *MENU* das Menü neu erzeugen.

TEMPORÄRE BENUTZERMENÜS

Temporäre Benutzermenüs erzeugen ein Menü, daß nur für eine begrenzte Zeit (z.B. Programmablauf) Gültigkeit besitzt und die Variable CST nicht überschreibt.
Temporäre Menüs werden mit dem Befehl *TMENU* erzeugt und sind im Aufbau mit "normalen" Benutzermenüs identisch.

DIE BENUTZERDEFINIERTE TASTATUR DES HP-48SX

Der HP-48SX besitzt nun, wie schon der HP-41 in Ansätzen, eine *völlig frei definierbare* Tastaturbelegung. Mit dieser ist es möglich sich seine eigene, programmspezifische und arbeitserleichternde Tastaurbelegung zu entwerfen. Dies ist besonders dann nützlich, wenn ein Anwender immer mit den gleichen speziellen Programmen arbeitet und er somit immer auf die gleichen Tastenkombination zurückgreifen muß. Man kann damit die ursprüngliche Tastatur des HP-48SX von unnützen Funktionen befreien und mit Hilfe einer Tastaturschablone (Overlay Kit, Option) die neuen Funktionen kenntlich machen.
Dabei ist es relativ unerheblich, ob es sich bei der Tastenfunktion um einen HP-internen Befehl oder um z.B. ein Anwenderprogramm handelt.

Um die benutzerdefinierte Tastatur zu aktivieren muß sich der Taschencomputer im sogenannten BENUTZERMODUS befinden.
Der Benutzermodus wird mit **<-┼ USR** aktiviert. Dabei stellt diese Tastenkombination einen 3-Wege-Umschalter dar.
Einfaches Drücken **<-┼ USR** setzt den Einmal-Benutzermodus (1USR).
Zweimaliges Drücken aktiviert den Benutzermodus (USER).
Dreimaliges Drücken schaltet den Benutzermodus ab.
Im Einmal-Benutzermodus wird die Tastaur nur für eine Operation umbelegt. Im Benutzermodus bleibt die umdefinierte Tastatur erhalten, bis der Modus wieder gelöscht wird.
Ein gesetztes Systemflag -61 desaktiviert den Einmal-Benutzermodus. Es läßt sich mit **<-┼ USR** nur noch der Benutzermodus ein und ausschalten.

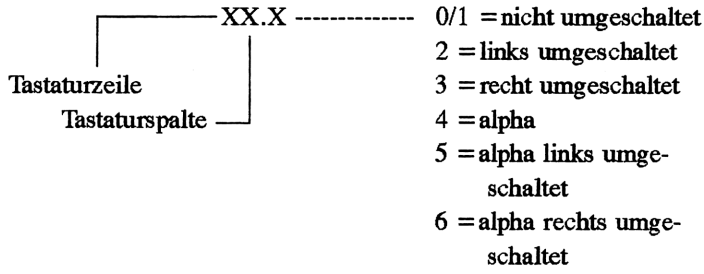
Umbelegen der Tastatur

Man unterscheidet Befehle zum Umbelgen einzelner Tasten und Befehle zum Umbelegen mehrerer Tasten.

Neudefinieren einzelner Tasten:

Einzelnen Tasten kann mit *ASN* eine neue Funktion zugewiesen werden.
Die dafür nötige Synatax lautet:

Ebene 2: Objekt (Tastendefinition)
Ebene 1: 3-stellige Positionsnummer der Taste



Beispiel: Der Taste ' soll der Stack-Befehl DEPTH zugeordnet werden.

Eingabe: **<-┼ «» PRG** *STK DEPTH* **ENTER 2 1 . 1 ENTER** *ASN*
Schalten Sie in den Benutzermodus um (siehe oben).
Die Taste ' löst nun bei einmaligem Druck den Befehl DEPTH aus.

Neudefinieren mehrerer Tasten:

Mit dem Besfehl *STOKEYS* lassen sich ganze Gruppen von Tasten umdefinieren. STOKEYS benötigt dazu eine Definitionsliste in folgender Form:

{ S Definition1 Position1 Definition2 Position2 ... }

S	= (nur bei Bedarf) erhält die Standardbelegung von nicht umdefinier ten Tasten. S ist nur dann erforderlich wenn DELKEYS vorher nicht aktiv war.
Definition	= beliebiges Objekt welches durch den Tastendruck ausgeführt wird.
Position	= 3-stellige Positionsnummer (siehe oben)

Beispiel:

Taste 7	= IF THEN END
Taste 8	= IF THEN ELSE END
Taste 9	= WHILE REPEAT END

Eingabe: { «IF THEN END» 52.1 «IF THEN ELSE END» 53.1 «WHILE REPEAT END» 54.1 } STOK

Aktivieren Sie die Benutzertastatur z.B. mit **<-┼ USR <-┼ USR.**
Wenn Sie jetzt die Taste **7** betätigen erhalten Sie z.B. in Programmen oder Strings die Befehlssequenz IF THEN END als Schreibhilfe.

Reaktivieren der Tastaturbelegung:

Löschen benutzerdefinierter Tasten

Mit dem Befehl *DELKEYS* kann eine oder mehrere Tastendefinitionen reaktiviert (Standardbelegung) werden.

Syntax:

1. Positionsnummer +-> **MODES** *DELK*
2. **S** *DELK*
   Die Standardtastenbelegung wird gelöscht.
3. **NULL** (0) *DELK*
   Alle Benutzerdefinierten Tastenbelegungen werden gelöscht.

Reaktivieren von einzelnen Standard-Tastenbelegungen:

Das Argument 'SKEY' löscht in Verbindung mit der 3-stelligen Tastenpositionsnummer und dem Befehl *ASN* eine einzelne Tastenfunktion.

Beispiel: 'SKEY' , **14.2** , *ASN* löscht für die Taste **VAR** die Benutzerdefinition.

Editieren von benutzerdefinierten Tasten:

Mit dem Befehl *RCLKEYS* (+-> **MODES**) kann eine Liste der aktuellen Benutzertasten in den Stack geholt werden. Diese Liste kann z.B. mit **EDIT** bearbeitet werden.

!Achtung! Jede gelöschte benutzerdefinierte Tastenbelegung hinterläßt im Speicher zwischen 2.5 und 15 Byte. Um diesen Speicherbereich zurückzugewinnen, sollte man von Zeit zu Zeit die Tastenbelegung reorganisieren. Dies geschieht mit der Befehlsfolge *RCLKEYS* **0** *DELKEYS* und *STOKEYS*.

BENUTZERDEFINIERTE FUNKTIONEN

Der HP-48SX bietet erstmals die Möglichkeit, den internen Befehlssatz weitere selbstgestaltete Funktionenen (externer Befehlssatz) erweitert werden. Für Benutzerdefinierte Funktionen gelten die gleichen Regeln wie für eingebaute Funktionen: Sie ist differenzierbar und akzeptiert symbolische oder algebraische Argumente.

Der Aufbau einer benutzerdefinierten Funktion

Eine benutzerdefinierte Funktion ist ein Programm mit folgenden Eigenschaften:
Es besteht nur aus einer Kombination von lokalen Variablen, deren Definition ein algebraischer Ausdruck ist.
Syntax: « **-> Name1 Name2 ... 'Ausdruck'** »
Es verwendet eine unbegrenzte zahl lokaler Variablen (Argumente), liefert jedoch nur **EIN** Ergebnis.
Lokale Variablen finden sie im Kapitel Programmierung.

Erzeugen einer Benutzerdefinierten Funktion

Syntax: **'Name(Argumente) = Ausdruck'**

Nehmen wir als Beispiel für eine benutzerdefinierte Funktion den Ausdruck:

F(x,a,b)=x^3-x^2+ax-b

Erzeugen Sie die benutzerdefinierte Funktion. Dazu geben Sie die Funktion in folgender Form ein:

'F(x,a,b)=x^3-x^2+a*x-b

Speichern Sie diese Funktion als benutzerdefinierte Funktion ab:
<-┼ DEF

Die Funktion bekommt den Namen F zuwiesen, da dieser links vor der Klammer der Argumente steht.
Nun können Sie durch den Aufruf des Menüs **VAR** diese Funktion anzeigen lassen.

Diese Funktion verwendet nun die Argumente:
x in Ebene3
a in Ebene2
b in Ebene1

Beispiel: x=2, a=5, b=3
Eingabe: **2 SPC 5 SPC 3 SPC F**
Ergebnis: 11

Benutzerdefinierte Funktionen können natürlich auch ineinander verschachtelt werden.

Nehmen wir an Sie benötigen noch eine benutzerdefinierte Funktion folgender Art:

$$HG(x,d,e) = \frac{F}{x*d*e}$$

HG sei die neue Funktion
F sei die oben genannte selbstdefinierte Funktion

Geben Sie die neue Funktion in folgender Form ein:

$$HG(x,d,e) = \frac{F(x,d,e)}{x*d*e}$$

Erzeugen Sie jetzt die benutzerdefinierte Funktion HG.
<-+ DEF

Die lokalen Variablen der ursprünglichen Funktion F werden in x, d und e umdefiniert. Diese Umdefinierung läßt aber die lokalen Variablen unberührt. Die gesamte Funktion F wird bei einem Funktionsaufruf von HG automatisch eingsetzt und mitberechnet.

FUNKTIONSANALYSE

Eine Funktionsanalyse wird am schnellsten mit Hilfe einer grafischen Darstellung der Funktion durchgeführt.

Wichtige Flags zur Funktionsanalyse:

Flag -30:	gesetzt:	Zeichnet (plottet) auch die linke Seite einer Gleichung (z=x^3). Das ist aber nur sinnvoll, wenn z einen sinnvollen Wert hat.
	rückgesetzt:	Unterdrückt den linken Ausdruck.

Sollte die linke Seite ein AUSDRUCK (tan(x)) sein, wird diese immer mit gezeichnet (unabhängig von -30).

Flag -31:	gesetzt:	(*CNCT: CNC'*) Der Funktionsgraph wird auch zwischen Unstetigkeitsstellen gezeichnet (Linie), das ist oft unerwünscht
	rückgesetzt:	Verbindet Unstetigkeitsstellen nicht.

Alle wichtigen Befehle zum Darstellen der Funktion finden Sie im Menü **PLOT**.

Alle wichtigen Befehle zur Funktionsanalyse befinden sich im Unterverzeichnis *FCN*.

Plotten (Zeichnen) von Funktionen

Eine Gleichung oder Funktion kann mit *DRAW*oder *AUTO* geplottet wrden, wenn sie in EQ gespeichert ist (*STEQ*, 'EQ' **STO**, *EQ+*).

Mehrere Gleichungen oder Funktionen können geplottet werden, wenn eine Liste mit ihren Namen an EQ übergeben wird.

Der **GLEICHUNGSKATALOG** nimmt Ihnen dabei die Arbeit zum Erstellen der Liste ab. Der Gleichungskatalog enthält alle momentan gespeicherten Funktionen und Gleichungen.
Wählen Sie den Gleichungskatalog z.B mit **<-+ PLOT**.

Setzen Sie den Cursor auf jede Gleichung, welche Sie gemeinsam mit anderen plotten wollen und betätigen Sie *EQ+*.
Bei einem Irrtum löscht *EQ-* den letzten Eintrag im Katalog.

Mit *PLOTR* werden die Gleichungen TEMPORÄR an EQ übergeben und kann mit *DRAW*oder *AUTO* dargestellt werden.

Um diesen temporären Eintrag in EQ dauerhaft zu speichern, müssen Sie für die Liste einen Namen vergeben.
Rufen Sie EQ mit *RCEQ* (**<-+ PLOT +->** *STEQ*) in den Stack zurück und drücken Sie *NEW*. Geben Sie einen Namen, welcher auf EQ endet ein und betätigen Sie **ENTER**.

Die Gleichung ist nun dauerhaft gespeichert.

Wollen Sie der Liste mit Gleichungen VOR dem Speichern in EQ einen Namen geben erzeugen Sie die Liste im Gleichungskatalog.
Betätigen Sie den Softkey -> *STK*, um die Liste in den Stack zu geben. Verlassen Sie den Gleichungskatalog mit **ATTN** und führen Sie *NEW* aus, um einen Namen zu vergeben.

Diskutiert werden soll die Funktion:
'x^3+x^2-x-3'

Es sollen folgende Größen ermittelt werden:

a) Alle Nullstellen der Funktion
b) Die Steigung an den Nullstellen
c) Der Achsenabschnitt (y-Wert an der Stelle x=0)
d) Die Koordinaten der Extremwerte

Verdeutlichen wir uns dazu folgendes Schaubild:

f(x) = 0

lokal monoton wachsend
für f'(x)$\geq$ 0, f"(x)$\leq$ 0

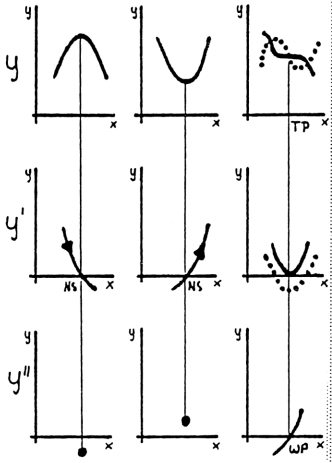
lokal monoton fallend
für f'(x)$\leq$ 0, f"(x)$\geq$ 0

Maximum
f'(x)= 0, f"(x) < 0

Minimum
f'(x)= 0, f"(x) > 0

f"(x)= 0, f"'(x)$\neq$ 0

Terassenpunkt
f'(x)= 0



Aus dem Schaubild erkennt man leicht die Zusammenhänge zwischen den Graphen der Ableitungen einer Funktion und Ihrer Funktionswerte.

Plotten Sie also obige Funktion:

Aktivieren Sie das Menü PLOT und geben Sie den Ausdruck in den Stack. Speichern Sie den Ausdruck mit *STEQ* in der Variablen EQ.
Aktivieren Sie das Unter-Menü *PLOTR* und setzen Sie alle Plotparameter mit *RESET* auf Ihre Standartwerte zurück. "Steigen" sie wieder eine Ebene "höher", in das Menü **PLOT**: **<-+ PREV**. Plotten Sie die Funktion mit *AUTO* (automatische Skalierung der Y-Achse).

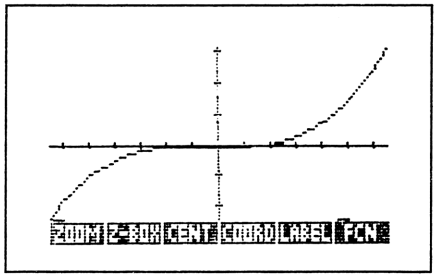


Bild 12 Funktion: x^3+x^2-x-3

Vergrößern Sie den interessanten mittleren Bereich:
Dazu bewegen Sie den Cursor mit den Cursortasten auf den linken, für den zu vergrößernden Bereich interessanten x-Wert. Setzen Sie eine Markierung mit *Z-BOX*, *ZOOM*, * oder *MARK*. Bewegen Sie den Cursor nun zum rechten gewünschten x-Wert. Vergrößern Sie jetzt das gewünschte Rechteck mit **<-+** *Z-BOX*, wobei die x-Achse AUTOMATISCH skaliert wird.

Sie können erkennen, daß sich in dem gewählten Bereich zwei Extrema und eine Nullstelle befinden.
Zoomen Sie gegebenfalls einen weiteren Bereich, indem Sie den Kursor an den ersten Eckpunkt eines imaginären Rahmens bewegen und eine Marke setzen. Gehen Sie dann mit dem Cursor an die gegenüberliegende Ecke des imaginären Rahmens und drücken Sie *Z-BOX* Der gewünschte Rahmen wird auf Displaygröße vergrößert.

Mit der Funktion *ZOOM* (*ZOOM* Variable Faktor **ENTER**: *ZOOM* **X 2 ENTER**) können Sie das Fenster um einen Faktor verkleinern.

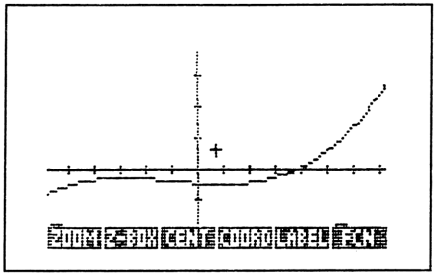


Bild 13 Vergrößerung des Graphen

Bewegen Sie den Cursor in die Nähe der Nullstelle und ermitteln Sie deren Wert mit *FCN ROOT*.

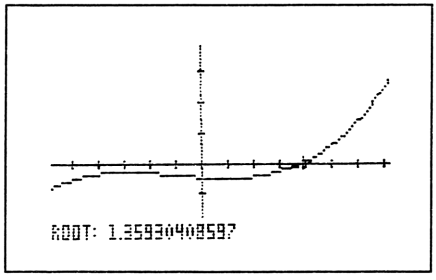


Bild 14 Ermittelte Nullstelle

Der Cursor steht nun auf der Nullstelle und die Koordinaten werden angezeigt.

Berechnen Sie die Steigung des Graphen an der Nullstelle.
Betätigen Sie eine beliebige Taste um das Menüfeld zu reaktivieren. Drücken Sie den Softkey *SLOPE*. Sie erhalten den Wert der Steigung angezeigt. Der Wert kann je nach Größe des gezoomten Rechtecks etwas schwanken.

Rufen Sie das Menü zurück und ermitteln Sie den Achsenabschnitt des Graphen (x=0). Dazu stellen Sie den Cursor auf die Y-Achse und betätigen *F(X)*. Der Wert wird angezeigt.

Stellen Sie nun den Wert des lokalen Minimums fest:
Bewegen Sie den Cursor in dessen Nähe und drücken Sie *EXTR*.
Der Wert des lokalen Minimums wird angezeigt.

Nun benötigen Sie noch den Wert des lokalen Maximums. Gehen Sie dabei wie oben vor.

Interesiert Sie noch die Ableitung der Funktion, können Sie diese mit *F'* darstellen. Die Schnittpunkte mit dem Funktionsgraphen erhalten Sie, wenn Sie den Cursor in deren Nähe bringen und *ISECT* ausführen.

AREA berechnet die Fläche unter einem Funktionsgraphen zwischen zwei x-Werten (Marke und Cursor).

Betätigen Sie **ON ON** zum Verlassen der Grafikumgebung. ALLE ermittelten Werte werden als markierte Objekte im Stack abgelegt.

!ACHTUNG! Sollten Sie beim Plotten einer Funktion oder Gleichung kein Achsenkreuz sehen betätigen Sie **<-+ REVIEW**. Sie erhalten die aktuellen Plotparameter zurück, welche darüber Aufschluß geben, wo Sie sich gerade befinden.
Zwei positive Skalierungs-Werte bedeuten z.B., daß Sie sich oberhalb der X-Achse befinden.
Auch *LABEL* kann Aufschluß über den Plotbereich geben, da der Befehl Achsenbeschriftungen erzeugt.
Mit *XAUTO* kann der Graph verkleinert und anschließend mit *Z-BOX* o.ä. wieder aufgezoomt werden. Auch mit *ZOOM* ist dies möglich.

Befehle zum Verändern der Plotparameter:

Diese Befehle beeinflussen gezielt die Argumente in der Plotparameter-Liste (in PPAR gespeichert)

INDEP Bestimmt die Variable in Ebene 1 als unabhängige Variable. Diese kann mit +-> *INDEP* gezeigt werden.
Beispiel: **X** *INDEP* oder { x -2 2} *INDEP*

DEPN(D) Erklärt die Variable in Ebene 1 zur unabhängigen Variablen. Diese kann mit +-> *DEPND* angezeigt werden.
Beispiel: **X** *DEPND* oder {X -2 2} *DEPND*

RES Stellt die Auflösung ein, +-> *RES* zeigt sie an.

AXES Legt den Koordinatenursprung neu fest, +-> *AXES* zeigt sie an.
Beispiel: (2,3) *AXES*

**H* Multipliziert den horizontalen Plotmaßstab oder vergrößert einen Ausschnitt bei n < 1.
Beispiel: **2** **H* oder **-3** **H*

**W* Multipliziert den vertikalen Plotmaßstab oder vergrössert einen Ausschnitt bei n < 1.
Beispiel: **2** **W*oder **-3** **W*

PTYPE Definiert den Plottyp

XRNG Legt den Anzeigebereich in X-Richtung fest.
Beispiel: **-4 6** *XRNG*

YRNG Legt den Anzeigebereich in Y-Richtung fest.
Beispiel: **-10 100** *YRNG*

CENT(R) Legt den Anzeige-Mittelpunkt fest.
Beispiel: (2,6) *CENTR*

<-+ **REVIEW** zeigt alle aktuellen Plotparameter

PLOTTYPEN:

Insgesamt kennt der HP-48SX fünf Plottypen zur Darstellung von Gleichungen und Funktionen und drei Plottypen zur Darstellung statistischer Werte.
Der Standart-Plottyp ist FUNCTION. Der Plottyp wird im Menü *PLOT PTYPE* oder *PLOTR PTYPE* bestimmt.

Typen:

FUNCTION Plottet Gleichungen, welche einen Wert f(x) zu jedem X-Wert liefert.

Definitionen:

FORM	Beispiel	gezeichnete Punkte
f(x)	x^3+x^2-x-3	(x,f(x))
y=f(x)	Y=x^2-x-3	Flag -30 = 0: (x,f(x))
		Flag -30 = 1: (x,f(x)) (x,y)
f(x)=g(x)	x^3=x^2-4	((x,f(x)) (x,g(x)))

CONIC Plottet Kegelschnitte

Hinweis: Kegelschnitte sollten nicht automatisch skaliert, sondern mit *CENT* und *SCALE* in Position gebracht werden. Eine Variable in *DEPND* sollte immer angegeben werden.

POLAR plottet Ausdrücke, welche den Radius für jeden Wert eines angegebenen polaren Winkels liefern (polare Kurven)

Definitionen:

FORM	Beispiel	gezeichnete Punkte
f(θ)	cos(θ)+sin(θ)	(f(θ),θ)
r=f(θ)	r=2cos(θ)	(f(θ),θ)
θ=konstant	θ=0,2 π	radiale Linie
f(θ)=g(θ)	4sin(θ)=r^2	(f(θ),θ) und (g(θ),θ)

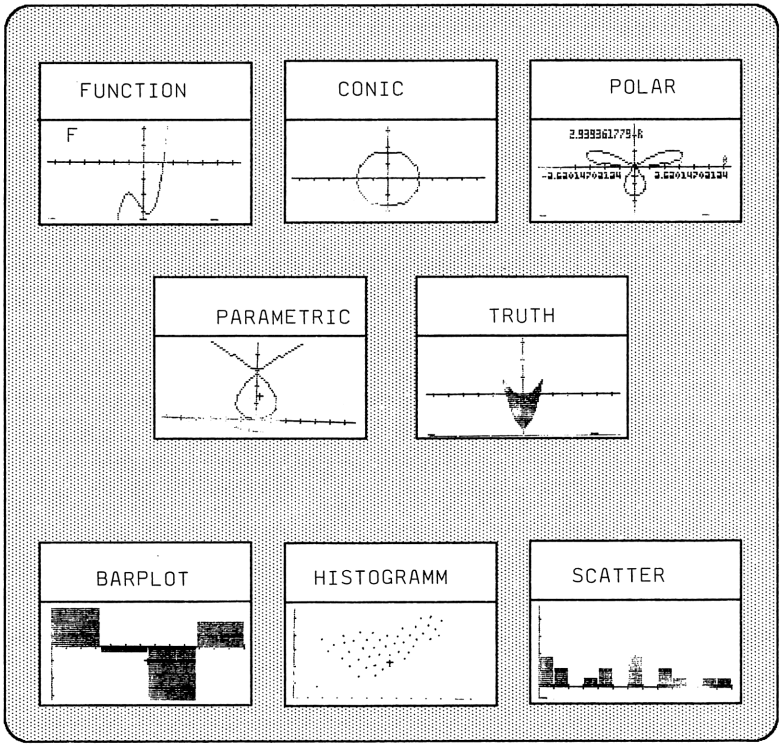
Für diese Funktion muß der DEG-Modus eingestellt werden.

PARAMETRIC Plottet Kurven in Parameterdarstellung

Bei diesem Funktionstyp muß der Ausdruck als algebraischer Ausdruck oder als ein Programm geschrieben werden, Ein komplexes Ergebnis muß zurückliefert werden.

TRUTH Plottet Ausdrücke, welche den Wert wahr oder falsch liefern, z.B. Gleichungen mit Vergleichsoperationen.
Jedes Pixel, für das der Ausdruck den Wert 1 ergibt wird eingeschaltet und jeder Ergebnis-Wert 0 läßt das Pixel unverän dert. Wird nichts anderes definiert, wird der Ausdruck für jedes Pixel ausgewertet.
In DEPND ist eine Variable anzugeben!

Die Eingabe von Wahrheitstabellen muß im algebraischen Eingabemodus geschehen.



DARSTELLUNG VON FUNKTIONEN MIT ANGABE IHRES WERTEBEREICHES:

Solche Funktion können mit Hilfe eines Programms dargestellt werden. Dabei sind aber einige Punkte zu beachten.

Darstellbar sind nur Programme, welche:

a) gleichbedeutend mit Funktionen des Typs FUNCTION oder des Types POLAR sind. Das Programm darf keine Werte vom Stack nehmen und nur einen unmarkierten Wert übergeben.

b) nur ein einziges komplexes Ergebnis vom Typ PARAMETRIC ausgeben.

c) Funktionen darstellen, die als Funktionen einer Variablen definiert sind.

Alle Operationen, die im Menü Graphics FCN verzeichnet sind, können nicht angewendet werden.

Beispiele:

zu a)

$$f(x) = \begin{cases} x^3-x^2+5 & \text{für } x>5 \\ x-3 & \text{für } x<=5 \end{cases}$$

Lösung: « IF 'X>5' THEN 'X^3-X^2+5' ELSE 'X-3' END »

zu b)

x= t-t^3 y=t^2

Lösung: « 't-t^3' -> NUM 't^2' -> NUM R>C »

zu c)
f(x)=COTH(X)

Lösung: 'COT(X)' *STEQ DRAW*

Weitere Beispiele:

FUNCTION: x=2x^2-3

Eingabe:
<-+ PLOT 'X=2*X^2-3' *STEQ PTYPE FUNC PLOTR* **NXT** *RESET NXT NXT AUTO*
Ausgabe:

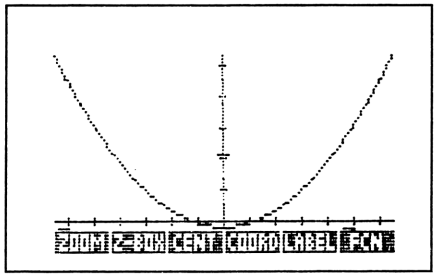


Bild 16 Plottyp FUNCTION

CONIC 2x^2+3y^2-9=0

Kegelschnitte:

x^2+y^2+c x+dy+e	=0	Kreis
ax^2+b y^2+c x+dy+e	=0	Ellipse (a+b)
ax^2-b y^2+c x+dy+e	=0	Hyperbel
x^2+c x+dy+e	=0	Parabel

Eingabe:
<-+ PLOT '2*X^2+3*y^2-9=0' *STEQ PTYPE CONIC PLOTR* **NXT** *RESET* 'Y *DEPN* **NXT NXT 'X** *INDEP AUTO*
Ausgabe:

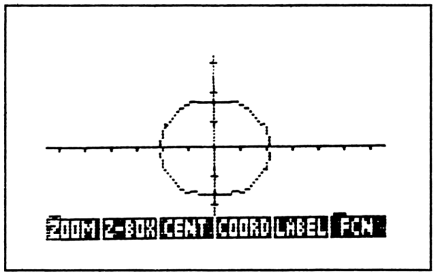


Bild 17 Typ CONIC

POLAR: $r=2\sin(3\theta)$

Eingabe:
<-+ RAD <-+ PLOT 'R=2*SIN(3*0) ENTER 'EQ STO PTYPE POLAR PLOTR '0 INDEP AUTO
Ausgabe:

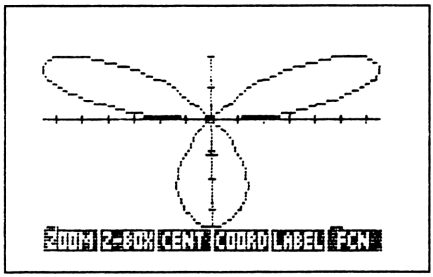


Bild 18 Plottyp POLAR

PARAMETRIC: x=t-t^3, y=t^2
Sinnvoll ist ein Zeichenbereich von x=[-3 3]

Eingabe:
<-+ **PLOT** 'T-T^3+i*(T^2)' *NEW* PK **ENTER** *PTYPE PARAPLOTR* **NXT** *RESET* **NXN NXT** {'T' -3 3} *ENTER INDEP AUTO*
Ausgabe:

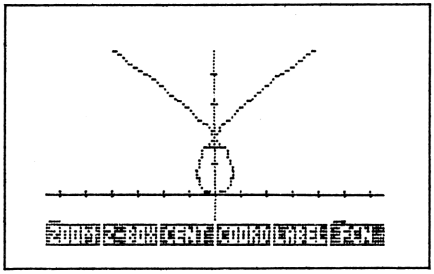
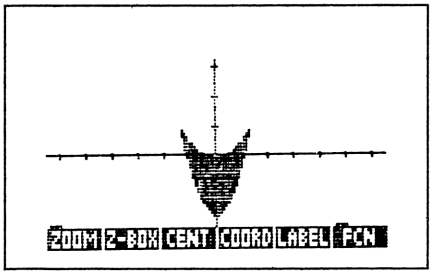


Bild 19 Typ PARAMETRIC

TRUTH: y<0.5x^2, y>1,5+x^2-2

!Achtung! Der HP-48SX braucht zum Aufbau der Grafik einige Minuten!

Eingabe:
<-+ **PLOT** 'Y<o.5*X^2 *AND* Y>1.5*X^2-2' *NEW* W_T **ENTER** *PTYPE TRUTH* **NXT** *RESET* 'Y *DEPND* **NXT NXT** *AUTO*
Ausgabe:



DARSTELLUNG STATISTISCHER DATEN

Der HP-48SX hat insgesamt drei Möglichkeiten statistische Daten optisch darzustellen:

BAR	Zeichnet ein Balkendiagramm mit den Daten einer angegebenen Spalte (*XCOL*) der Statistikmatrix ∑DAT.
HISTOGRAMM	Erstellt ein Verteilungsdiagramm mit den Daten aus einer angegebenen Spalte (*YCOL*) von ∑DAT.
SCATTER	Erzeugt ein Streudiagramm mittels Punkten aus ∑DAT.

Wichtige Befehle zum Beeinflussen der Datendarstellung:

Alle wichtigen Befehle zum Beeinflussen der Abbildung statistischer Daten, finden Sie im **STAT**-Menü.
Die Darstellung ist abhängig von einer Liste mit Plotparametern, welche in der Variablen ∑PAR abgelegt wird.
∑PAR hat folgendes Format:
{unabh.-Variable abh.-Variable Achsenabschnitt Steigung Modell}

Folgende Befehle haben entscheidenten Einfluß auf die erzeugte Grafik:

XCOL	Eine bestimmte Spalte (Zahl) wird zur unabhängigen Variablen. Erster Eintrag in ∑PAR, kann mit +->XCOL angezeigt werden.
YCOL	Eine angegebene Spalte (Zahl) wird zur abhängigen Variablen. Zweiter Eintrag in ∑PAR, kann mit +-> YCOL angezeigt werden.

LR Berechnet die lineare Regression für die gewählte abhängige und unabhängige Variable in ∑PAR und gibt den Schnittpunkt mit der Ordinate und die Steigung aus. Die Werte werden an dritter und vierter Stelle in ∑PAR gespeichert.

MODL Wählt das Grafikmodell für die Abbildung (Stichproben):
a) LIN = Linear (Gerade)
b) LOG = Logarithmus (LOG-Funktion)
c) EXP = Exponentialfunktion
d) PWR = Potenzfunktion
e) BEST= automatische Modellwahl

BARPL(OT) Gibt ein Balkendiagramm der Spalte X aus. Der Masstab wird automatisch gewählt.

HISTP(LOT) Zeichnet ein HäufigkeitsHistogramm der Spalte X. Der Maßstab wird automatisch gewählt.

SCATR(PLOT) Zeichnet einzelne Punkte (x,y) aus den angegebenen Spalten x und y auf Wunsch in der bestmöglichen Form. Das Modell wird automatisch gewählt.

∑LINE liefert einen Ausdruck, der die am Besten angepaßte Kurve darstellt.

BALKENDIAGRAMME

Mit BARPLOT läßt sich aus einer in ∑DAT gewählten Spalte (mit *XCOL*) ein Balkendiagramm erstellen. Wird keine Spalte angegben wählt der HP-48 automatisch die erste Spalte. Durch negative bzw. positive Werte entstehen Balken über oder der X-Achse.

Geben Sie alle benötigten Daten in den MatrixWriter ein und speichern Sie diese Matrix z.B. mit *NEW* in einer Variablen. Wählen Sie dann die für das Balkendiagramm wichtige Spalte und lassen Sie sich die Daten mit *BARPL* anzeigen.
Sobald die Graphik erstellt wird, wechseln Sie automatisch vom Statistik-Menü zum Plot-Menü. Mit **ON** (ATTN) wechseln Sie wieder in die Statistikumgebung.

Balkendiagramme sind die klassische Anzeigeform von relevanten Statistikdaten.

Beispiel:
Geben Sie folgende Matrix in den Stack ein:

```
[[ 4    -2     ]
 [ -1    4     ]
 [ -6    -3    ]
 [  2,5  1     ]]
```

Rufen Sie den MatrixWriter auf und schreiben Sie die Matrix in die entsprechenden Felder. Übergeben Sie die Matrix mit ENTER an den Stack. Speichern Sie die Matrix mit ∑+ in ∑DAT ab.

Stellen Sie die Daten graphisch dar:
<-┼ STAT NXT NXT 1 XCOL BARPL
Die Werte aus Spalte 1 werden als Balken dargestellt.

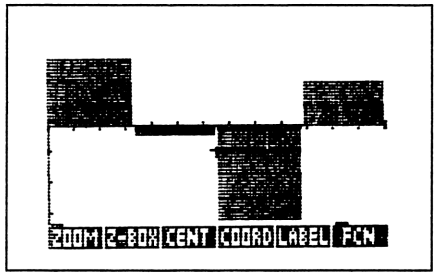


Bild 21 Plottyp BARPLOT

DIE SPEICHERORGANISATION DES HP-48SX

Der gesamte Speicher des HP-48 läßt sich in zwei Bereiche aufteilen.
Der erste Bereich des Speichers wird vom sogenannten ROM-Speicher (Read Only Memory) belegt. Der Inhalt des ROM-Speichers kann nicht verändert werden. Im ROM sind alle internen Befehle des HP-48SX gespeichert. Er hat eine Kapazität von 256 KByte. Dieser Speicherbereich läßt sich mit Hilfe von ROM-Einsteckkarten

erweitern. Auf diesen befinden sich sogennante **Befehlsbibliotheken**.

Der zweite Speicherbereich wird vom sogenannten RAM-Speicher (Random Access Memory) eingenommen. Der Inhalt des RAM-Speichers kann beliebig verändert werden. Der RAM Speicher des HP-48SX besitzt eine Größe von 32 KByte, der sich aber durch RAM-Einsteckkarten bis auf maximal 544 KByte erweitern läßt.

Der Speicher des HP-Taschencomputers kann über den normalen Hauptspeicher hinaus über sogenannte Ports angesprochen werden. Der HP-48SX verwendet 3 Portnummern (0, 1, 2) zur Verwaltung dieses speziellen Speicherbereiches.
Port 0 ist ein besonderer Bereich des Benutzerspeichers, der hauptsächlich für Sicherungsobjekte und Bibliotheken verwendet wird. Port 0 hat eine dynamische Größe, je nach Dimension seines Inhalts.

Die Verwendung von Port 0 hat den Vorteil, daß ich in der Lage bin Daten, welche innerhalb des Speicherbereiches, den ich über Port 0 anspreche, abgelegt sind geheimhalten kann. Der Benutzerspeicher (Hauptspeicher) des HP-48SX wird immer über diesen Port angesprochen. Das bedeutet, die Variablen in Port 0 erscheinen nicht im Menü VAR. Diese Variablen lassen sich nur über einen speziellen Befehl ansprechen.
Syntax:
: Portnummer : Variablenname EVAL oder
<-┼ LIBRARY Portnummer Variablenname
Der erste Befehl akzeptiert auch eine Liste mit mehreren Sicherungsobjekten und verarbeitet diese gemeinsam.
Diese Befehle gelten auch für Port 1 und Port 2.

Port 1 und Port 2 sind unabhängige Speicherbereiche, welche ebenfalls über ihre Adresse verwaltet werden. Dabei ist es gleichbedeutend, ob sich in den beiden Ports RAM- oder ROM-Speichererweiterungskarten befinden.

Der sogenannte unabhängige Speicher einer RAM-Karte in Port 1 oder Port 2 kann durch "Anbinden" an den Hauptspeicher des HP-48SX in einen sogenannten verbundenen bzw. abhängigen Speicher verwandelt werden.

Anbinden des RAM-Speichers einer Erweiterungskarte an den Hauptspeicher.

Der Befehl *MERGE* koppelt den gesamten Speicher einer RAM-Erweiterungskarte an den vorhandenen Hauptspeicher an. Dazu muß sich der Schreibschutzschalter der RAM-Karte in Stellung Lesen/Schreiben befinden.

Syntax: **Portnummer <-+ MEMORY NXT NXT *MERG***

Geschieht dies, werden auf der RAM-Karte evtl. vorhandene Sicherungsobjekte (spezielle Variablen), zu Port 0 verschoben. Damit können Sie direkt im Hauptspeicher verwaltet werden.

!Achtung! Niemals eine RAM-Karte ausbauen, die ein Teil des Benutzerspeichers ist, da sonst die Daten im Hauptspeicher verlorengehen können. Vor dem Ausbau den Taschencomputer abschalten und den Speicherbereich auf der Karte mit FREE freigeben.

Sollten Sie trotzdem eine RAM-Karte mit eingebundenem Speicher entfernen erscheint die Meldung: "Replace RAM, Press ON", wenn der HP-48SX eine Operation ausführt. Nun können Sie versuchen, einen größeren Datenverlust zu verhindern, indem Sie bei eingeschaltetem Rechner die Karte wieder in den gleichen Port zurücksetzen und ON betätigen.

Sicherungsobjekte

Alle in einem Port gespeicherten Objekte (Variablen) nennt man Sicherungsobjekte. Sicherungsobjekte haben ein spezielles Format. Es enthält ein Objekt, seinen Namen und seine Prüfsumme.

Wichtige Befehle für den Zugriff auf RAM-Speicherkarten

Im Verzeichnis **<-+ MEMORY** sind alle wesentlichen Befehle verzeichnet.

BYTES Syntax: Objekt BYTES
Gibt die Prüfsumme und die Größe eines Objekts in Bytes an.

LIBS Syntax: LIBS
Zeigt eine Liste, welche die Namen, die Nummern und die Portadresse aller Sicherungsobjekte enthält, an.

MERG Syntax: Portnummer MERG
Koppelt einen RAM-Erweiterungsspeicher an den Hauptspeicher des HP-48SX an.

FREE Syntax: {Definitionen} Portnummer FREE
Listeninhalt:
Leer: Gesamten eingebundenen Speicher freigeben.
Sicherungsobjekte: Gesamten eingebundenen Speicher freigeben und die angegebenen Sicherungsobjekte von Port 0 nach Port 1 schieben.
Bibliotheken: Gesamten eingebundenen Speicher freigeben und Die angegebenen Bibliotheken an Port 1 zurückgeben.
Koppelt einen RAM-Erweiterungspeicher vom Hauptspeicher ab.

ARCHI(VE) Syntax: : Portnummer : Name ARCHIV
Legt eine Sicherungskopie des HOME-Verzeichnisses, der benutzerdefinierten Tastatur und des Terminkatalogs, an.

RESTO(RE) Syntax: : Portnummer : Name RESTOR
Stellt alle Daten wieder her, die mit dem Befehl ARCHIV gesichert wurden.

PVARS Syntax: Portnummer PVARS
gibt die in einem Port verwendete Speicherart in Ebene 1 und eine Liste der Sicherungsobjekte und Ihrer Identifizierungsnummer in Ebene 2 ab.

STO Syntax: : Portnummer : Name STO
Speichert das angegebene Objekt im entsprechenden Port ab.

PURGE Syntax: : Portnummer : Name : <-+ PURGE
Löscht das angegebene Objekt im entsprechenden Board.

MEM Syntax: <-+ MEMORY MEM
zeigt die Größe des verfügbaren Speichers.

RCL Syntax: : Portnummer : Name +-> RCL

VERWENDUNG VON ROM-ERWEITERUNGSKARTEN

ROM-Einsteckkarten unterscheiden gegenüber RAM-Karten nur dadurch, daß von von Ihnen nur gelesen werden kann. Die auf der ROM-Karte gespeicherten Objekte werden in einer sogenannten *Bibliothek* zusammengefaßt. Eine solche Bibliothek erweitert den internen Befehlssatz des HP-48SX. Der Inhalt dieser Bibliotheken kann weder dargestellt noch verändert werden, da eine Bibliothek ''maschinennahe'' Befehlsroutinen enthält. Bibliotheken können nicht mit dem HP-48SX erzeugt werden.

Eine Bibliothek ist definiert durch:

a) Ihr Identifizierungskennzeichen
Aufbau: : **Port# : Bibliothek (Library)#.**
Das Identifizierungskennzeichen dient als Argument für Befehle, welche die Objekte einer Bibliothek verarbeiten.
b) Den Bibliotheksnamen
Aufbau: **''Name : Objekt'' oder Menüname in Library**

Aktivieren einer ROM-Einsteckkarte

Das Aktivieren einer ROM-Karte geschieht über das sogenannte Anbinden der darauf gespeicherten Bibliothek an das Inhaltsverezichnis im Benutzerspeicher.
Das Anbinden geschieht entweder automatisch oder manuell.

Automatisches Anbinden:
Schalten Sie Ihren Rechner ab, stecken Sie die ROM-Karte richtig in einen Port, schließen Sie die Abdeckung und schalten Sie wieder ein.
Damit sind die auf der Karte gespeicherten Bibliotheken automatisch im Menü Library installiert.

Manuelles Anbinden:
Schalten Sie Ihren Rechner ab, stecken Sie die ROM-Karte richtig in einen Port und schließen Sie die Abdeckung.

Mit dem Befehl **ATTACH** installieren Sie bestimmte Bibliotheken in Ihrem HOME- oder Unterverzeichnis. An ein Unterverzeichnis kann jedoch nur EINE Bibliothek angebunden werden.
Die Syntax lautet: **Bibliotheks# ATTACH**

Zugreifen auf Bibliotheken

Mit dem Menü **LIBRARY** kann ich alle Bibliotheksobjekte im aktuellen Verzeichnispfad darstellen. Wollen Sie ein Programm aus einer Bibliothek starten, so betätigen Sie nur den Softkey mit dem Namen des Programmes.
Es kann sein, daß sich das Menü **LIBRARY** in Untermenüs verzweigt. Diese sind in Ihrer Struktur gleich mit den Verzeichnisstrukturen in **VAR**. Es gelten die gleichen Arbeitsregeln.

Mit *EQLIB* wird ein Menü aller Operationen der aktuellen Bibliothek angezeigt.

+-> **RCL** ruft eine Bibliothek auf den Stack'.

STO speichert eine Bibliothek in einem unabhängigen Speicher an einem angegebenen Port.

DIE PROGRAMMIERUNG DES HP-48SX

1. DER AUFBAU VON PROGRAMMEN

Ein Programm im HP-48 beginnt immer mit dem Zeichen «.
Ein Programm darf mehrere Unterprogramme enthalten, welche entweder auch mit dem Zeichen « beginnen oder in einem Namen enthalten sind. Die überwiegend objektorientierte Programmiersprache des HP-48 stellt ein nahezu ideales Werkzeug zum Bearbeiten komplexer mathematischer Aufgaben dar.

Allerdings verlangt die Gestaltung von Programmen etwas mehr Aufmerkasamkeit und Logik als z.B. bei der Programmierung bin BASIC, welche ja in sehr vielen anderen Taschencomputern zu finden ist. So gibt es in allen HP-Taschencomputern keine Befehle, welche ein ''wildes Herumspringen'' in Programmen erlauben, wie z.B. die BASIC-Befehle GOTO oder GOSUB. Statt dessen existiert eine Reihe von bedingten Verzweigungen, wie IF...THEN...ELSE...END oder CASE...END, welche nacheinander durchlaufen werden müssen. Außerdem hat jedes Programm im HP-48SX nur **EINEN** Ein- und Ausgang.

So können Programme logisch aufgebaut werden, sind relativ einfach beschreiben und zu testen. Wenn man sich an einfache programmtechnische Regeln hält, bleibt das gestaltete Programm klein, übersichtlich und kann leicht nachvollzogen werden.

Belohnt wird Ihre Mühe durch eine höhere Verarbeitungs- bzw. Rechengeschwindigkeit und durch eine größere Übersichtlichkeit von Programmen gegenüber einem Rechner, der mit Basic arbeitet.

GRUNDREGELN DER PROGRAMMIERUNG

1. Programmieren Sie strukturiert, d.h. gestalten Sie ein Programm so, daß Programmanweisungen nacheinander durchlaufen werden.

2. Vermeiden Sie Sprungstellen, wie z.B. geschachtelteVerzweigungen.

3. Stellen Sie an den Anfang des Programmes eine kurze Programmbeschrei bung und weisen Sie auf Unterprogramme hin.

4. Definieren Sie zu Beginn des Programmes alle benötigten Variablen, Unterprogramme und Prozeduren.

5. Schreiben Sie Unterprogramme nie in das Hauptprogramm, sondern extern in Variable.

6. Definieren Sie keine Endlosschleifen.

7. Achten Sie auf eine saubere und übersichtliche Programmgestaltung.

8. Gestalten Sie die Ein- und Ausgabemasken des Programmes so, daß die Bedeutung der einzelnen Funktionen sofort er- sichtlich wird.

9. Löschen Sie am Programmende alle nicht mehr benötigten Variablen und Funktionen.

10. Rufen Sie am Programmende alle Modi wieder auf, in denen sich der Rechner vor Beginn des Programms befand.

11. Wenn Sie Unterprogramme mit « innerhalb des Hauptprogrammes aufru fen und Variablen benutzen, die außerhalb dieses Unterprogrammes als lokale Variablen definiert wurden, können diese unter Umständen nicht erkannt werden. Verwenden Sie in diesm Fall nur normale Variablen.

Bemerkung: Nicht immer ist eine REKURSION in Programmen sinnvoll. Beachten Sie dabei vor allem das Laufzeitverhalten eines Programms!

PROGRAMMABLAUF

Sie haben zwei Möglichkeiten ein Programm in Ihrem Rechner ablaufen zu lassen:
1. Sie geben alle Werte in den Stack, welche das Programm benötigt und rufen das Programm auf. Das Programm holt sich alle Werte vom Stack und führt seine Berechnungen durch.
2. Sie rufen ein Programm auf, welches eine qualifizierte Meldung in den Stack gibt und dann die Eingabe der benötigten Werte verlangt. Daraufhin wird das Programm mit einer Taste oder automatisch fortgesetzt.

1. Übergabe von Objekten an Programme

Argumente bzw. Objekte werden normalerweise entweder an lokale Variablen (diese werden an anderer Stelle erklärt) oder globale Variablen übergeben.

Beispiel lokale Variablen:
Lokale Variablen existieren nur innerhalb eines Programms und werden nicht gespeichert.

« -> a b c « » »
Drei Werte werden vom Stack genommen und an die lokalen Variablen a, b, und c übergeben.

Beispiel globale Variablen:
Globale Variablen werden mit dem STO-Befehl erzeugt. Diese Variablen werden in einem Verzeichnis von VAR abgelegt und bleiben dort gespeichert. Sie müssen also am Programmende extra gelöscht werden.

« 'A' STO 'B' STO 'C' STO » oder « 3 -> LIST 'V' STO »
Beide Programme nehmen drei Argumente vom Stack. Der Unterschied ist nur, daß im ersten Fall die Objekte in drei globalen Variablen gespeichert werden und im zweiten Fall in einer Liste, in nur einer globalen Variablen.

Weitere Beispiele:
Diese schematischen Programme verlangen die Eingabe zweier Werte in den Stack, bevor sie gestartet werden. Diese Werte werden entweder in lokalen oder globalen Variablen gespeichert.

1:

```
« -> a b
        « Programm
        »
»
```

2:

```
«       'A' STO 'B' STO
        Programm
        « Unterprogramm
        »
        { A B } PURGE
»
```

3:
Dises Programm fragt, ob vorher eingegebene Werte a, b multipliziert oder dividiert werden sollen. Um eien Vorgang auszuwählen, müssen Sie die Taste M oder D betätigen.

```
«               DO CLLCD                Beginn äußere Schleife
"MULTIPIZIERN ODER"                     Auswahl-Meldung Teil 1 in Zeile 1
1 DISP "DIVIDIEREN?"                    Auswahl-Meldung Teil 2 in Zeile 2
2 DISP
"1 FÜR MULTIPLIKATION"                  Auswahl-Meldung Teil 3 in Zeile 3
3 DISP
"ODER 2 FÜR DIVISION"                   Auswahl-Meldung Teil 4 in Zeile 4
4 DISP
  DO                                    Beginn innere Schleife, solange wieder-
                                        holen bis eine Taste gedrückt wird.
  UNTIL KEY                             Falls Taste gedrückt gibt Key einen
  END                                   String und 1 zurück
UNTIL {"1" "2"}                         Beginn äußere Testanweisung: Taste
```

	gedrückt? Taste definiert? Und Liste
SWAP POS DUP	mit definierten Tasten. Vergleichen des Tasten-Strings mit den in der Liste definierten. POS gibt 1 für 1 und 2 für 2 und 0 für eine andere Taste zurück DUP erstellt eine Kopie der Position
IF 1 ==	Wenn gedrückt Taste = 1 Wahr-Flag
THEN 'a*b'	dann berechnen von a*b
EVAL	Ausdruck auswerten
ELSE 'a/b'	sonst berechnen von a/b
EVAL	Ausdruck auswerten
END SWAP	Ende IF-THEN-Struktur, falls Taste definiert, ist Wahr-Flag im Stack
END	Ende äußere Schleife falls die gedrückte Taste definiert war endet die Anzeige-Schleife.

»

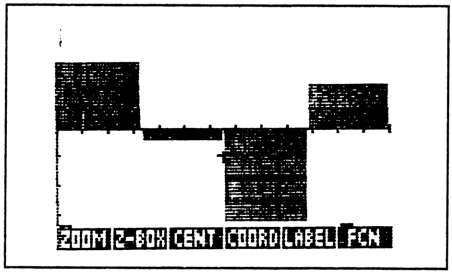


Bild 22 Übergabe von Daten über "Hotkey"

Variablennamen

Variablennamen können aus bis zu 127 Zeichen bestehen, sollten aber nicht länger als fünf Zeichen sein. Sonst können die Namen in den Menü-Feldern des HP-48SX nicht ganz dargestellt werden.

!ACHTUNG! Variablennamen dürfen keine Zahl als erstes Zeichen enthalten.
Namen von Befehlen können nicht als globale Variablen verwendet werden.
Die Konstanten e, i, s1 oder s2 können ebenfalls nicht als Variablen benannt werden.
Folgende Zeichen sind in Variablennamen ebenfalls auszuschließen:
() # [] ‘ “ {} < > «» * + - / & ∂ ∫ ° ≠ ^√ ∠ ≥ • ,
Namen, welche mit “der” beginnen, sind für benutzerdefinierte Ableitungen reserviert.

2. Programm-Dialoge unter Verwendung von Strings.

Sie sollten ihre Anwendungsprograme grundsätzlich mit **Ein- und Ausgabemasken** versehen. Ein- und Ausgabemasken sind graphische Programmelemente, welche Sie z.B. dazu auffordern bestimmte Daten einzugeben. Diese Aufforderung sollte einem Programmbenutzer kurz und deutlich zu erkennen geben, welche Daten er eingeben muß.
Die Ergebnisse eines Programms sollten villeicht kurz dokumentiert und optisch unmißverständlich über Ausgabemasken im Display dargestellt werden.

Es findet also ein Dialog zwischen Taschencomputer und Benutzer statt.

Dies ist besonders dann wichtig, wenn sich mit der Zeit eine größere Zahl von Programmen im Speicher des HP-48SX tummeln. Da weiß ein Anwender nach einiger Zeit warscheinlich nicht mehr genau, wie bestimmte Programme funktionieren. Mit sauber gestalteten Programm-Dialogen ist dieses Problem leicht zu umgehen.

Die Befehle zur Datendarstellung bzw -abfrage finden Sie im Menü **PRG** *CTRL* (Program Control) und **PRG** *DSPL* (Program Display).

Die wichtigsten Befehle sind:

HALT Unterbricht den Programmablauf.

INPUT Unterbricht den Programmablauf, um Daten einzugeben. Verhindert Stackoperationen!
Syntax:"Meldung" "Promt" *INPUT*

PROM(T) Stoppt den Programmablauf und zeigt eine Meldung in der Statuszeile des HP-48SX.
Syntax: "Meldung" *PROMT*

DISP Stellt ein Objekt in der angegebenen Zeile des Displays dar.
Syntax: "Meldung" **Zeileennummer** *DISP*

WAIT Unterbricht den Programmablauf für eine bestimmte Zeit
Syntax: **Sekunden** *WAIT* oder **0** *WAIT* (bis Tastendruck)

KEY Liefert ein Testergebnis nach Ebene 1 und die Position einer evtl. gedrückten Taste.
Syntax: **Taste** *KEY*

BEEP Erzeugt ein Akustiksignal
Syntax: Frequenz Dauer *BEEP*

CLLCD Löscht die Anzeige

FREEZE "Friert" bis zu drei Anzeigebereiche ein, bis eine Taste gedrückt wird.
Syntax: "Objekt" Bereichscode-Summe *FREEZE*
Bereichscode-Summe ist die Summe der Codes für einzelne Displaybereiche:
1 = Statusbereich, 2 = Stackbereich, 3 = Menübereich

PVIEW Stellt PICT mit den entsprechenden Koordinaten dar.
Syntax: Grafikobjekt (x,y) *PVIEW*

BLANK Erzeugt ein leeres Grafikobjekt.

GOR Überlagert Grafikobjekte
Syntax: G-Objekt1 (x,y) G-Objekt2 *GOR*

GXOR	Überlagert Grafikobjekte mittels Exclusiv-Oder Syntax: g-Objekt1 (x,y) G-Objekt2 *GXOR*
REPL	Ersetzt einen Teil eines Grafikobjekts oder einerZeichenkette Syntax: Objekt1 Position Objekt2 *REPL*
SUB	Erstellt einen Auszug aus einer Zeichenkette oder aus einem Grafikobjekt Syntax: Objekt Startpunkt Endpunkt *SUB*
-> LCD	Stellt ein Grafikobjekt im Display dar Syntax: Grafikobjekt *-> LCD*
LCD->	Wandelt eine Grafikdarstellung in ein Grafikobjekt um Syntax: Grafik *LCD->*
TEXT	Schaltet auf Stackanzeige um.

Beispiele für Programmdialoge (schematisch)

Beispiel 1:
Dieses Programm (schematisch) erzeugt nach Programmstart eine Meldung und fordert zur Dateneingabe auf. Das Programm läuft erst nach Betätigen von **CONT** weiter.
Siehe Bild 22 Seite 68

«		Programmanfang
	CLLCD	Löschen der Anzeige
	"Enter a und b" 2 DISP	Anzeige der Meldung in Zeile 2
	HALT	Programmunterbrechung bis Tastendruck
	-> a b	Argumente in lokale Variablen
	« Hauptprogramm	Programm
	»	
»		Programmende

Beispiel 2:
Dieses Programm erzeugt nach Programmstart eine Meldung zur Dateneingabe in der Statuszeile und hält das Programm an. Das Programm wird mit **CONT** fortgesetzt.
Siehe Bild 23 Seite 68

«	Programmanfang
"Geben Sie a und b ein"	Meldung
PROMT	Schreibt die Meldung in die Statuszeile, unterbricht das Programm
NW	führt das Programm NW aus
»	Programmende

Beispiel 3:
Nach Programmstart wird eine Meldung im Display angezeigt und solange "eingefroren" bis ein Tastatureingabe erfolgt. Mit **CONT** kann das Programm fortgesetzt werden.
Siehe Bild 24 Seite 68

«	Programmanfang
"Bitte■ a undb■ eingeben"	drei-stellige Meldung
CLCD	Bildschirm löschen
1 DISP	Meldung am Anfang von Zeile 1
3 FREEZE HALT	Meldung in Status- und Stackbereich und Programmunterbrechnung
NW	führt das Programm NW aus.
»	Programmende

Bemerkung: ■ ist das Bildschirmzeichen für Neue Zeile.

Beispiel 4:

Der INPUT-Befehl

Hier wird am Programmanfang eine Meldung in der ersten Zeile des Stackbereiches angezeigt, ein Promt in der letzten Zeile dargestellt und das Programm solange unterbrochen bis ENTER betätigt wird. Das Programm wird mit den in die Befehlszeile eingegebenen Werten fortgesetzt.
Sie auch Bild 25

« "Name von CON?	Meldung für erste Stackzeile
":CON:"	Prompt für Befehlszeile
INPUT	Meldungen ausgeben, Programm anhalten und auf ENTER warten
Programm	Programmablauf
»	Programmende

Bild 22 Dateneingabe mit Disp

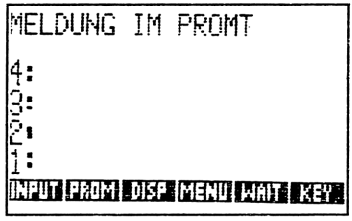


Bild 23 Dateneingabe mit Promt

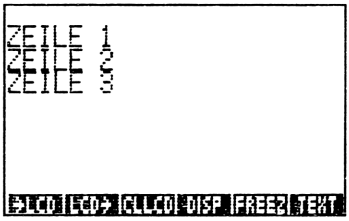


Bild 24 Dateneingabe mit FREEZE

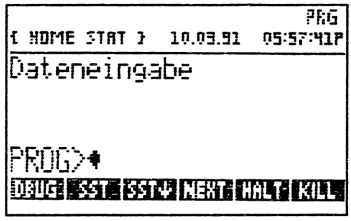


Bild 25 Dateneingabe mit INPUT

Die weiteren Möglichkeiten von INPUT

Allgemein ist das Argument von INPUT eine **Liste**, welche den Inhalt und die Interpretation der Befehlszeile bestimmt.

In dieser Liste kann eine Reihe von **Parametern** für INPUT angegeben werden:

- Ein Befehlszeilen-String, dessen Inhalt von INPUT in der Befehlszeile angezeigt wird.
- Eine reelle Zahl oder eine Liste mit 2 reellen Zahlen, welche die Anfangsposition des Befehlszeilen-Cursors angibt:
  - Eine reelle Zahl n gibt an, daß der Cursor auf das n-te Zeichen vom linken Befehlszeilenrand gesetzt wird.
    Eine positive Zahl initialisiert einen Einfügecursor.
    Eine negative Zahl initialisiert eien Überschreibcursor.
    0 wählt das Ende einer Zeichenkette in der Befehlszeile.
  - Eine Liste, welche die Zeile und Spalte der Anfangsposition des Cursors festlegt. Zahl legt eine Zeile innerhalb der Befehlszeile fest.
    Zeile: 1 = 1. Zeile der Befehlszeile
    Spalte: 0 = Ende der Zeichenkette in der angegebenen Zeile der Befehlszeile
    +n = Einfügecursor
    -n = Überschreibcursor
- Bis zu drei Parameter (ALG, α oder V), die ohne Anführungszeichen einzugeben sind.
  - ALG aktiviert den algebraischen Programmeingabemodus.
  - α (α +-> A) verriegelt die alphabetische Tastatur.
  - V überprüft ob der Inhalt der Ergebniszeichenkette " gültige Objekte sind. Sind diese Zeichen kein gültiges Objekt, so wird INVALID SYNTAX ausgegeben und ein weiteres Mal zu Dateneingabe aufgefordert.

Optionen für INPUT-Zeichenketten

" " INPUT:
Bildet eine leere Befehlszeile und stellt den Cursor an den Anfang.
Die von INPUT gelieferte Zeichenkette besteht nur aus den Eingabedaten.

":a:":b:"{-1 0} V:
Die Eingabezeichenkette besteht aus zwei markierten Objekten, welchen die Eingabedaten übergeben werden. Der Überschreibcursor wird direkt nach :a: in Zeile 1 gesetzt. Eine Syntax-Überprüfung wird durchgeführt.

"Telefonnummer?" { "( )/ " -1 } INPUT
Diese Sequenz erzeugt die Meldung "Telefonnummer?" in dem oberen Stackbereich, wählt den Überschreibcursor und stellt den String in der geschweiften Klammer in der Befehlszeile dar. Diese Eingabemaske kann jetzt ausgefüllt werden.

Aufbereitung der Ausgabedaten

Auch die Ausgabedaten sollten übersichtlich dargestellt werden.
Beispiel:
Folgendes Programm wandelt ein Ergebnis in einen String um und verknüpft diesen mit einer Meldung.

«	Programmanfang
Programm	Programmrumpf
-> STR	Ergebnis in String umwandeln
"Mittelwert = "	Voranstehende Meldung
SWAP +	Ergebnis anhängen
CLLCD 2 DISP 5 WAIT	Display löschen, Meldung in Zeile 2 anzeigen und 5 Sekunden Programm-Pause
»	Programmende

Temporäre Menüs zum Eingeben von Daten und Starten von Unterprogrammen.

Temporäre Menüs sind ein nützliches Hilfsmittel, wenn es darum geht:
a) Mehrmals verschiedene Daten einzugeben.
b) Auf zeitraubende graphische Bildaufbauten zu verzichten.
c) Programmabläufe vollständig interaktiv zu kontrollieren.

Hier nun ein Beispiel zur Kontrolle eines Programms über ein temporäres Benutzermenü:

```
«                                   Programmanfang
{                                   Aufbau der Menü-Definition

{                                   Definition des 1. Menüfeldes
 "SQX" «''Geben                     Menüname "SQX"
  Sie X ein" PROMPT                 Prompt anzeigen
  -> x «'x^2' EVAL»»}               Variable holen und x^2 berechnen
}                                   Ende 1. Menüfeld

{                                   Initialisieren 2. Menüfeld
 "DOX" «''Geben Sie                 Menüname "DOX"
  den X-Wert ein"                   Prompt anzeigen
  PROMPT -> x «'1/x'                Variable holen und 1/x berechnen
  EVAL »»                           Ausdruck auswerten, Ende Unterprogramm
}                                   Ende 2. Menüfeld

{                                   Initialisierung 3. Menüfeld
 "GR" {«'G' STO»                    normale Tastenfunktion: Menüname "GR",
                                    Objekt in G speichern
 «{ (-10 -10) (10 10) X             Taste links umgeschaltet:
   1 (0 0) FUNCTION Y}              Festlegen der Plotparameter
  'PPAR' STO G DRAW»                Speichern der Plotparameter, G zeichnen
 « G "Funk" -> TAG CLLCD            Taste rechts umgeschaltet: Funktion G
   1 DISP 1 FREEZE»}                Markieren und Anzeigen
}                                   Ende 3. Menüfeld
»                                   Programmende
```

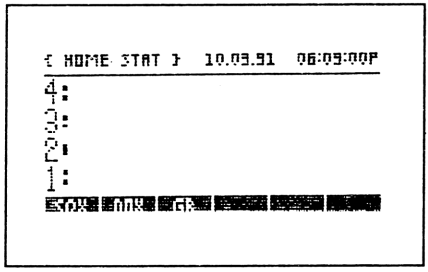


Bild 26 Temporäre Menüs

Das Programm baut ein Menü mit den Feldern SQX, DOX und GR auf. Mit dem Softkey *SQX* wird ein Wert x verlangt, gespeichert und quadriert.
Mit dem Softkey *DOX* wird ein x-Wert verlangt, gespeichert und 1/x berechnet.
Mit dem Softkey *GR* sind 3 Funktionen möglich:

Normales Drücken der Taste: Funktion in G speichern.

GR links umgeschaltet: Zeichnen des Graphen.

GR rechts umgeschaltet: Funktion markieren und der Variablen 'Funk' zuweisen. Dann das Objekt anzeigen.

PROGRAMMSCHLEIFEN UND BEDINGTE VERZWEIGUNGEN

Die Programmiersprache des HP-48SX bietet Ihnen mehrere Arten von Programmschleifen und bedingten Verzweigungen. Die dazu notwendigen Befehle sind im Menü PRG (PROGRAM) BRCH (BRANCH) abgelegt.

Viele dieser Programmstrukturen basieren auf Test-Anweisungen, wobei ein Flag vom Stack genommen und auf seinen Status überprüft wird. Hat ein solcher Flag den Status (Wert) 0, ist er "falsch" oder "gelöscht"- hat er einen anderen Wert, ist er "wahr" oder "gesetzt". Jede darauffolgende Programmverzweigung ist davon abhängig, ob ein Flag den Status 0 oder 1 besitzt. Gibt z.B. das Auswerten der Test-Anweisung einer IF Test-Anweisung THEN Wahr-Anweisung ELSE Falsch-Anweisung END einen "falsch"-Flag (0) zurück, so wird die Falschanweisung ausgeführt. In Test-Aneweisungen gibt es verschiedene Testbefehle. Der Befehl a+b prüft, ob das Argument a ungleich dem Argument b ist und liefert, falls dies zutrifft, einen "wahr"-Flag als Ergebnis.

Weitere Testbefehle sind:
>, <, ≤, ≥, ==, FS?, FC?, FS?C, FC?C, TYPE, SAME.
Diese Befehle sind im Benutzerhandbuch Bd. 2 auf Seite 556 ausführlich beschrieben.

Die wichtigsten Merkmale der Programmschleifen und bedingten Verzweigungen des HP-48SX

If Test-Anweisung THEN Wahr-Anweisung END
Schreibhilfe: **<-┼ IF**
Der Befehl THEN prüft den Flag (Ergebnis Test-Anweisung) auf seinen Status und führt die Wahr-Anweisung aus, falls der Flag den Wert 1 hat.
Beispiel: If x 0 ≠ THEN DUP END
Diese Prozedur dupliziert das Objekt in Ebene 1, wenn x ungleich 0 ist.

IF Test-Anweisung THEN Wahr-Anweisung ELSE Falsch-Anweisung END
Schreibhilfe: **┼-> IF**
Beispiel: IF a 0 < THEN NEG ELSE SWAP END
Diese Prozedur negiert das Objekt in Ebene 1, wenn a kleiner 0 ist. Sonst wird Ebene 2 mit Ebene 1 vertauscht.

CASE **Test-Anweisung1 THEN Wahr-Anweisung1 END**
Test-Anweisung2 THEN Wahr-Anweisung2 END
...
Test-Anweisungn THEN Wahr-Anweisungn END
Standard-Anweisung (Falsch-Anweisung, bei Bedarf)
END
Schreibhilfe:

<-┼ CASE
┼-> CASE (THEN...END)

Diese Programmstruktur wertet nacheinander die Test-Anweisungen aus. Ist eine Testanweisung wahr, folgt die Ausführung der zugehörigen Wahr-Anweisung und ein Sprung zu END. Ist eien Testanweisung falsch, wird die nächste Test-Anweisung ausgewertet. Sind alle Test-Anweisungen falsch wird die Standard-Anweisung ausgeführt.

Beispiel:
CASE
x 1 > THEN DUP END
x 1 < THEN DROP END
"Falscher Wert" 2 DISP
END

Falls x größer 1 ist, liefert die Test-Anweisung das Ergebnis wahr. Danach wird das Objekt in Ebene 1 dupliziert und die Programmsequenz beendet. Ist x nicht größer als 1, wird zum nächsten Test-Ausdruck (x < 1) gewechselt. Dort wird der gleiche Ablauf vollzogen.
Ist x weder kleiner noch größer 1 wird die Meldung Falscher Wert in Zeile 2 des Displays angezeigt. Danach wird die Sequenz beendet.

Test-Anweisung Wahr-Anweisung IFT (Wenn-dann-Ende)
IFT ist die abgekürzte Form von IF...THEN...END.
IFT nimmt einen Flag von Ebene 2 und ein Objekt (Wahranweisung) von Ebene 1. Das Objekt in Ebene 1 wird ausgewertet, wenn der Flag wahr ist. Ist der Flag falsch, wird das Objekt verworfen.
Beispiel: «0 < "nicht negativ" IFT»
Ist ein Objekt, daß von Ebene 1 genommen wird größer Null, wird der String "nicht negativ" in den Stack geschrieben.

'IFTE(Test-Anweisung Wahr-Anweisung Falsch-Anweisung)'
IFTE wertet die Test-Anweisung aus und führt die Wahr-Anweisung durch, falls der zurückgegebene Flag den Wert 1 hat. Sonst wird die Falsch-Anweisung ausgeführt.
Beispiel: « -> n 'IFTE(n< 10,n^3,1)»
Dieses kleine Programm nimmt eine Variable n vom Stack und prüft in der Test-Anweisung, ob n kleiner 10 ist. Wenn ja wird ein Flag 1 zurückgeliefert und die Wahr-Anweisung n^3 ausgeführt. Wenn nicht wird eine 1 in den Stack geschrieben.

Test-Anweisung Wahr-Anweisung Falsch-Anweisung IFTE
Beispeiel: « 0 > "positiv" "negativ" IFTE»
Ein Wert wird aus Ebene 1 genommen und "positiv" wird angezeigt, wenn der Wert größer Null ist. Sonst wird "negativ" angezeigt.

IFERR Prüf-Anweisung THEN Fehler-Anweisung END
Schreibhilfe: **<-+ IFERR**
Sollte beim Auswerten der Prüf-Anweisung ein Fehler auftreten, so wird die Fehler-Anweisung ausgeführt.
Beispiel:
IFERR '1/x' EVAL THEN ''Unendliches Ergebnis'' 2 DISP END
Tritt beim Auswerten von 1/x Ein Fehler auf, wird der String ''Unendliches Ergebnis'' angezeigt.

IFERR Prüf-Anweisung THEN Fehler-Anweisung ELSE Richtig-Anweisung END
Der Unterschied zur normalen IFERR-Anweisung ist der, daß diese Prozedur die Richtig-Anweisung ausführt, wenn die Prüf-Anweisung wahr ist.
Beispiel:
IFERR '(X^2)/X' EVAL THEN ''Unendliches Ergebnis'' ELSE ''OK'' END
Ergibt sich bei der Auswertung von (X^2)/X kein Fehler, wird ''OK'' in das Display geschrieben.

Meldung Zahl (reell/binär) DOERR END
Mit dem Befehl DOERR kann absichtlich ein Fehler erzeugt werden. Der Befehl kann z.B in der IFERR-Prozedur verwendet werden.
Zahl ist eine Anweisung an DOERR, entweder die internen Systemmeldungen (wird von ERRN geliefert), eine selbst definierte Meldung oder keine Meldung anzuzeigen.

Nichts	=selbst definierte Meldung
0	= keine Meldung
Zahl	= systeminterne Meldung (wird z.B. von ERRM) geliefert.

Beispiel:
x IF 1 > THEN ''x größer 1'' 0 DOERR END
Diese Prozedur gibt die Meldung ''x größer 1'' aus, falls x größer 1 sein sollte.

Anfangswert Endwert START Schleifen-Anweisung NEXT
Die Schleifen-Anweisung wird solange ausgeführt, bis der Endwert erreicht ist.
Schreibhilfe: **<-+ START**

Beispiel: 1 6 START SQ(x) NEXT
x wird sechsmal quadriert und die Schleife beendet.

Anfangswert Endwert START Schleifenanweisung Schrittweite STEP
Schreibhilfe: +-> **START**
Die Schleife wird (Endwert-Anfangswert)/Schrittweite mal widerholt.
Beispiel: 1 6 START + 2 STEP
Der Befehl + wird dreimal angewendet, um drei Stackobjekte zu addieren.

Anfangswert Endwert FOR Name Schleifen-Anweisung END
Schreibhilfe: <-+ **FOR**
In der lokalen Variablen Name werden nacheinander die Werte vom Anfangswert bis zum Endwert gespeichert und in der Schleifenanweisung berechnet.

Beispiel: 1 6 FOR n 'n/x' EVAL END
In der Variablen n werden die Zahlen 1 bis 6 gespeichert und n/x berechnet.

Anfangswert Endwert FOR Name Schleifen-Anweisung Schrittweite END
Schreibhilfe: +-> **FOR**
Die Schleife wird (Endwert-Anfangswert)/Schrittweite mal widerholt.
Beispiel: 1 6 FOR x 'a*x' EVAL 2 STEP
a*x wird drei mal berechnet.

DO Schleifen-Anweisung UNTIL Test-Anweisung END
Die unbestimmte Schleife wird solange widerholt, bis die Test-Anweisung den Wert wahr liefert.
Schreibhilfe: <-+ **DO**
Beispiel: DO SPEC UNTIL x 0 == END
Die Variable SPEC wird solange ausgewertet bis x den Wert Null hat.

WHILE Test-Anweisung REPEAT Schleifen-Anweisung END
Schreibhilfe: <-+ **WHILE**
Die Schleifen-Anweisung wird solange befolgt, wie die Test-Anweisung den Status wahr hat.

Beispiel: WHILE z 0 ≠ REPEAT '1/z' EVAL END
1/z wird solange berechnet, bis z gleich Null ist.

Schleifenzähler (Inhalt lokale Variable) setzen

Name *INCR*
INCR erhöht die Zahl, welche in der Variablen Name gespeichert ist um 1.

Name *DECR*
DECR verringert die Zahl in der Variablen um 1.

SYMBOLISCHE ARGUMENTE IN PROGRAMMEN

Der HP-48SX bietet die Möglichkeit Programme zu entwickeln, welche auf einfache Art symbolische Argumente (Variablen) verarbeiten. Das ist wohl der größte Vorteil des HP-48SX Taschencomputers gegenüber der Konkurrenz.
Algebraische Berechnungen mit symbolischen Argumenten sind im **ALGEBRA**-Menü möglich. Es gibt auch eine Reihe von Befehlen außerhalb dieses Menüs, um symbolische Argumente zu verarbeiten, das sind z.B. ∫ oder ∂ .

Wollen Sie aber normale Funktionen und Befehle auf symbolische Argumente anwenden, müssen Sie ein Programm erstellen.

Dazu übergeben Sie Ihre symbolischen Argumente (Variablen mit oder ohne Inhalt) **immer** in eine **Liste**.
Eine Liste hat den Vorteil, daß in ihr eine Reihe willkürlicher Objekte stehen kann. Mit den Befehlen *LIST->* , *PUT, GET, PUTI, GETI, SUB, OBJ->* , *NUM und SIZE*, die im Menü **PRG** *OBJ* zu finden sind, kann völlig frei auf jedes einzelne Objekt in einer Liste zugegriffen werden.

Mit dem Befehl: **ZahlElemente -> *LIST*** kann eine bestimmte Anzahl von Objekten in eine Liste übernommen werden.

Beispiel:

Ebene 5: « SWAP»
Ebene 4: 8
Ebene 3: a
Ebene 2: 3
Ebene 1: -> LIST
übernimmt die drei Elemente von Ebene 5: bis 3: (definiert durch die 3 in Ebene 2:) in eine Liste.

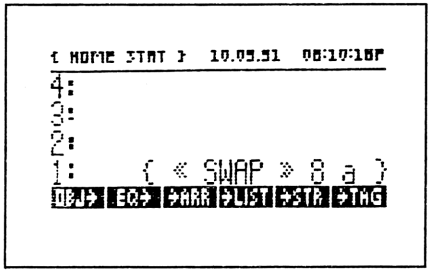


Bild 27 Listeneingabe

Vergleichen Sie bitte das Programm DIV (Divergenz) in der Programmbibliothek. In diesem Programm werden zwei Vektoren |x y z| =a(r) in {x y z} und |x y z| =r in {x y z} in zwei Listen übernommen.
Die Objekte werden dann aus der ersten Liste (a) entnommen und in Variablen (D,C,B) gespeichert. Anschließend werden diese dann einzeln differenziert. Die Ergebnisse werden addiert und in einer neuen Variablen (di) abgelegt. Dann werden die Objekte der zweiten Liste (r) entnommen und in den Variablen gespeichert, welche in di vorkommen. Das Objekt di wird abschließend ausgewertet und als Ergebnis in den Stack geschrieben.
Sie sehen daran, daß Listen eine elementare Bedeutung bei der Verarbeitung von Variablen besitzen.

UNTERPROGRAMME

Unterprogramme sind notwendig und sinnvoll, wenn innerhalb mehrerer verschiedener Programme, immer wieder auf die gleichen Unterroutinen zugegriffen werden muß. Eine solche, immer wiederkehrende Programmsequenz, in alle Programme zu übernehmen würde den RAM-Speicher des Systems nur unnötig belasten. Deshalb werden solche Routinen in Variablen gespeichert, auf die dann alle Programme zurückgreifen können.
Ein weiterer Grund Unterprogramme zu verwenden ist der, alle Programme "modular" aufzubauen.

MODULARE PROGRAMMIERUNG bedeutet, daß ein Programm in viele, sinnvoll kleine, unabhängige Teile (Module) aufgesplittet wird, welche aber nur **EINEN** Ein- und Ausgang besitzen sollten.
Aus solchen "vorgefertigten" Modulen kann dann ganz schnell ein neues Programm entstehen.
Diese Module sind, ihrer Übersichtlichkeit wegen, leicht und schnell auf Ihre Funktion hin zu überprüfen.

Aufruf von Unterprogrammen

Ein Unterprogramm wird einfach durch seinen Namen (Variable ohne Begrenzungszeichen) aufgerufen.

Einschränkungen:

a) Der Name darf kein HP-48SX-Befehl sein.

b) Der Name (=Variable mit Unterprogramm) darf nicht in einem niedrigeren bzw. in einem parallelen Verzeichnis abgelegt sein.

Vergleichen Sie dazu das Kapitel über Verzeichnisse im HP-48SX.

Auch Bibliotheken kann man als Unterprogramme bezeichnen, sie werden nur etwas anders aufgerufen (s.o.).

Beispiel: « *SWAP DUP* NNI **2** *DISP* »
Dieses Programm vertauscht Ebene 1 mit Ebene 2 und dupliziert das in Ebene 1 stehende Objekt. Dann wird das Unterprogramm NNI aufgerufen, welches z.B. Argumente vom Stack nimmt und dann eine Meldung an das Hauptprogramm zurückliefert, welches dann die Meldung in der zweiten Displayzeile darstellt.
Unterprogramme können natürlich auch direkt in das Hauptprogramm geschrieben werden. Dies ist jedoch selten sinnvoll (siehe oben).

Beispiel: « *SWAP DUP* « + * -> *STR* » **2** *DISP* »
Das Unterprogramm in der Mitte des Hauptprogramms wird automatisch als solche erkannt, da es ebenfalls in *Programm-Begranzungszeichen (« »)* steht und automatisch ausgeführt.
Nehmen wir an, das dieses Unterprogramm unser oben beschriebenes Unterprogramm NNI ist. Es nimmt hier drei Argumente vom Stack, addiert Objekt 1 und 2 und multipliziert dieses mit Objekt 3. Dann wird das Ergebnis in einen String umgewandelt und an das Hauptprogramm zurückgegeben. Im Hauptprogramm erfolgt dann die Ausführung des Befehls 2 DISP.

PROGRAMM-PROZEDUREN

Unter einer Programmprozedur auf dem HP-48SX versteht man eine Programmsequenz, welche nur **EINEN Ein und Ausgang** besitzen sollte und in einer Variablen gespeichert ist. Eine Prozedur kann auch als Unterprogramm bezeichnet werden.

Definition von Variablen und Prozeduren in Programmen

Definieren Sie, der Übersichtlichkeit wegen, alle im Programm verwendeten Variablen am Anfang des Programms. So können evtl. auftretende Programmfehler leichter aufgespürt und behoben werden, da der Inhalt der Variablen im Programm leicht abgefragt werden kann.
Verzeichnen Sie ebenfalls alle Prozeduren, welche innerhalb eines Programms aufgerufen werden. So kann leicht überprüft werden, warum ein bestimmter Programmteil nicht funktioniert.

Sie haben mehrere Möglichkeiten, Variablen zu Beginn des Programms einen Variableninhalt zuzuweisen z.B.:

a) Mit einem **Markieren** des Variableninhalts
Syntax: **:Name (Markierung): Objekt (Variableninhalt)**
Mit diesem Befehl weisen Sie der Variablen Name einen Variableninhalt (Objekt) zu.
Beispiel: :v: 1
Der Wert 1 wird in der Variablen v gespeichert.

b) Durch ein direktes Abspeichern des Objektes mit STO
Sytax: **Objekt (Variableninhalt) 'Name' STO**
Beispiel: 1 'V' STO
Speichert den Wert 1 in der Variablen V.

c) Durch das direkte Erzeugen einer lokalen Variablen
Syntax: **Objekt -> lokale Variable**
Damit läßt sich ein Objekt direkt an eine lokale Variable übergeben.
Beispiel: 5 -> n
Der Wert 5 wird an eine lokale Variable übergeben.

LOKALE VARIABLEN

Lokale Variablen sind ein nützliches Hilfsmittel, wenn es darum geht, bestimmten Variablen für eine BEGRENZTE Zeit einem folgendem Programm zur Verfügung zu stellen. Das bedeutet, daß diese Variablen nur für die "Laufzeit" eines Programms existieren.
Eine lokale Variable wird nicht in einem Verzeichnis gespeichert, sondern nur temporär *im Haupspeicher* verwaltet.
Nach einem Beenden des aufrufenden Programmes wird die Variable gelöscht.
Lokale Variablen können aber innerhalb der Programm-Laufzeit durch einfache Eingabe des Namens abgerufen werden.

Durch die Verwendung lokaler Variablen erspart man sich eine Überprüfung und das Löschen der in einem Verzeichnis "übriggebliebenen", nicht mehr benötigten ‚Programm-Variablen.

!Achtung!Bedenken Sie , daß lokale Variablen nur im aufrufenden Programm gültig sind also **nicht** in vorhandenen Unterprogrammen aufgerufen werden können!

Lokale Variablen werden in einem Programm für ein darauffolgendes Unterprogramm bereitgestellt.
Beispiel:
« -> n x « Programm zum Verarbeiten von n und x » »

DER RICHTIGE AUFBAU EINES PROGRAMMS

Richten Sie sich bei der Konstruktion eines Programms nach den Grundregeln der Programmierung.
Lassen sich alle geforderten Punkte erfüllen, müßte ein Anwenderprogramm so aussehen:

Hauptprogramm

```
«       "Copyright"
        "Programmbeschreibung"
        "Verwendete Unterprogramme"
        "Verwendete Prozeduren"
        "Verwendete Variablen"
        Variablen-Definition (falls möglich)
        Prozeduren definieren
        Bereitstellen der für die verwendeten Unterprogramme
        und Prozeduren erforderlichen Argumente. Evtl. Auf-
        forderung zur Dateneingabe über Bildschirmmasken.

        Aufruf der Unterprogramme und Prozeduren

        Aufbereitung der Ergebnisse und Darstellung am Display.
»
```

Unterprogramme

```
«       "Programmbeschreibung"
        Variablen-Definition (falls nötig)
        Prozeduren und villeicht vorhandene Unter-Unter-
        Programme definieren

        Prozeduren und Unterprogramme abarbeiten

        Ergebnisse an das Hauptprogramm übergeben
»
```

Beispiel:

Hauptprogramm: WURZ

```
«       "Programm WURZ, Robert Hübner 1990"
        "Programm zum Multiplizieren von Wurzeln"
        Unterprogramm: NW"
        "Eingabe: x, n, z"
        "Ausgabe: n*((∑der 2., 3., 4. Wurzel von x)*z)"
        "Verwendetes Unterprogramm: NW"
        "Verwendete Prozedur: START-STEP-Schleife für die
        Anzahl der Berechnungen"
        "Variablen: x, a, z"
        "Laufvariable: n"
        "Variablenzuweisung:"

        :a:1

        CLLCD
        "Berechnet n*((2.+3.+4-te√x)*z)"
        2 DISP
        "Eingabe : x n z und CONT"
        4 DISP HALT
        ->x n z

        «   x 'X' STO
            a n START NW z * 1 STEP
            'X' PURGE
        »

        CLLCD
        "Ergebnis= " SWAP ->STR + 2 DISP
»
```

Unterprogramm: NW

```
«       "Programm zum Berechnen von ∑x+x^(1/3)+x^(1/4)"
        "Laufvariable: e"
        "Verwendete Prozedur: FOR-NEXT-Schleife"

        2 4 FOR e 'X^(1/e)' EVAL NEXT + +
»
```

HP-48SX PROGRAMMBIBLIOTHEK

Diese Kapitel stellt Ihnen einige Programme zur Verfügung, welche Ihnen die Arbeit mit Ihrem Taschencomputer erleichtern sollen.

Sie finden auf den folgenden Seiten Programme zu den wichtigen Themenbereichen Mathematik, Statistik, Elektrotechnik und Graphik.

Die Programme in der Bibliothek können meist auch in anderen Anwender-Programmen Ihren Dienst versehen und als sogenannte Programmodule verwendet werden. Mit diesen Modulen sind Sie in der Lage mit einfachen Mitteln neue, eigene Anwenderprogramme aufzubauen.

Alle Programme sind nur da ausführlich erklärt, wo es notwendig erscheint. Die Programme setzen Grundkenntnisse in der Funktion des HP-48SX vorraus.

Für Schäden, welche durch unsachgemäßen Gebrauch dieser Programme oder durch nicht sachgerechte Benutztung des HP-48SX entstehen, kann keine Haftung übernommen werden. Für evtl. vorhandendene Druckfehler übernehme ich keine Verantwortung.

Bei Fragen wenden Sie sich bitte an den Verlag.

ÜBERSICHT ÜBER DIE PROGRAMMBIBLIOTHEK 12/1990

Mathematik-Bibliothek

Utilities-Bibliothek

Grafik-Bibliothek

BIBLIOTHEK MATHEMATIK

SQRI

EinProgramm zum Berechnen der n-ten Wurzel aus einer reellen, oder imaginä ren Zahl

Verwendet: keine weiteren Programmmodule
Eingabe: SQRI Zahl n CONT
Ausgabe: n berechnete Wurzeln

```
« CLLCD 2 FIX ''ENTER (a,bj),n'' 2 DISP
HALT -> z n
« DEG z n n z OVER
INV ^ SWAP DUP 1 - 1 SWAP
START
OVER R->P C->R 3 PICK 360 SWAP / +
R->C R->P SWAP NEXT DROP
»
»
```

Beispiel: (cos 120° + i sin 120 °)^(1/4) =z

Eingabe : SQRI 120 COS 120 SIN PRG OBJ NXT R->C 4 ENTER CONT

Ergebnis: (-0.5,0.87), (0.87,0.5), (-0.87,-0.5), (0.5,-0.87)

HORS

Das Programm HORS simuliert das Horner-Schema für reelle bzw. komplexe Zahlen und Parameter.

Verwndet : Unterprogramm Repeat
Eingabe: HORS { Koeffizientenliste }, x0-Wert (Entwicklungsstelle) CONT
Ausgabe: vollständige Entwicklung nach Horner in umgekehrter Reihenfolge.

Listing:

```
« CLLCD
"ENTER {},NS" PROMT -> a b
« 2 'c' STO a b a SIZE 1 SWAP a LIST-> 0 SWAP 1 + -> LIST 'a'
STO
START a 1 GET DUP 2 a SIZE
FOR i b * a i GET + « EXPAN » Repeat COLCT DUP NEXT
DROP DROP a SIZE 1 - -> LIST DUP 'a' STO c 1 + DUP 'c' STO
ROLLD NEXT
DROP 'c' PURGE
»
»
```

Beispiel: z^4-8z^3+22z^2-24z+12 soll für x0=2 entwickelt werden
Eingabe : HORS { 1 -8 22 -24 12 } ENTER 2 ENTER CONT
Ausgabe :

```
{                          1.00 }
{                     1.00  0.00 }
{                1.00 -2.00 -2.00 }
{           1.00 -4.00  2.00  0.00 }
{  1.00 -6.00 10.00 -4.00  4.00 }
{  1.00 -8.00 22.00 -24.00 12.00
```

GAUSS

GAUSS ist ein Programm zur Simulation des GAUß-Algorithmus auf dem HP-48SX. Es können reelle oder imaginäre Zahlen und Variable verarbeitet werden.

Verwendet: Unterprogramm rolle und ugaus
Eingabe : GAUSS {Koeffizientenmatrix} CONT
Ausgabe : Gauß-Jordan-Form der Matrix

Listing:

```
Hauptprogramm: GAUSS
«CLLCD
"Homogene Form AX=B" 1 DISP
"Enter {x11 x12 x13 ...}" 2 DISP
"Enter {x21 x22 x23 ...}" 3 DISP
"Enter {x31 x32 x33 ...}" 4 DISP
HALT DEG 0 FIX {w v} PURGE
1 5 START
  DUP SIZE 1 - 'w' STO 0 'q' STO
  DUP DUP SIZE 'z1' STO LIST->
  'z' STO rolle
  1 z START
    IFERR DUP IF 0 == THEN
            DROP 1 'q' STO+ ELSE
          END
    THEN DROP
    END
  NEXT z1 q - ->LIST SIZE
  IF 2 > THEN w ROLL
    w 1 - 1 SWAP START
      w ROLL ugaus
```

```
    NEXT w ROLLD CLLCD 250 .1 BEEP
    "Weiter? = CONT" 2 DISP 3 WAIT
    HALT ELSE
  END {w q z1 z} PURGE
NEXT
»
```

Unterprogramm: rolle

```
«z1 2
FOR i i ROLLD -1 STEP
»
```

Unterprogramm: ugaus

```
« -> a b
  « b 1 GET a 1 GET / NEG 'v' STO a
    SIZE 2 SWAP
    FOR i a i GET v * b GET + EXPAN
      COLCT
    NEXT a SIZE 1 - -> LIST a 'v'
    PURGE
    »
»
```

Beispiel : x1- x2- x3=-2
2x1+ ux3=-4
-x1+2x2+ x3=v

Eingabe : GAUS, {1 -1 -1 -2}, {2 0 u -4}, {-1 2 1 v}, CONT

ROT

Dieses Programm berechnet die ROTATION rot u eines Vektorfeldes und gibt an, ob in einem Punkt, lokal betrachtet, eine Rotationsbewe gung stattfindet. Es wird eine Determinante mit der Regel von SARRUS berechnet. Es können auch symbolische Werte eingegeben werden.

Verwendet: Unterprogramm Repeat
Eingabe : ROT {Determinaten-Werte} CONT
Ausgabe : Rotation {a(r)} und Lösungsvector {r}

```
« CLLCD
  "Enter {a(r)}, {r}" 2 DISP
  HALT -> a r
  « a LIST-> DROP 'D' STO 'C' STO 'B' STO
    '∂Y(D)' « EVAL » Repeat
    '∂Z(C)' « EVAL » Repeat - COLCT
    '∂Z(B)' « EVAL » Repeat
    '∂X(D)' « EVAL » Repeat - COLCT
    '∂X(C)' « EVAL » Repeat
    '∂Y(B)' « EVAL » Repeat - COLCT
    3 DUPN 3 ->LIST 4 ROLLD 'W' STO
    'V' STO 'U' STO r LIST-> DROP 'Z' STO
    'Y' STO 'X' STO
    U EVAL COLCT V EVAL COLCT W EVAL COLCT
    3 ->LIST
  » { a r B C D U V W X Y Z } PURGE
»
```

Beispiel: a(r)= y^2+z^2 , r= 1

$$\begin{matrix} 0 & 2 \\ 0 & 2 \end{matrix}$$

Eingabe : ROT {'y^2+z^2 0 0}, {1 2 2}, CONT
Ausgabe : {0.2*z -2*y} {0 4 -4}

DIVG

Das Programm DIVG ermittelt die Divergenz eines Vektorfeldes. Die dafür zugrunde liegende Formel lautet:

$$\text{div } v = \frac{dvx}{dx} + \frac{dvy}{dy} + \frac{dvz}{dz}$$

Das Programm verwendet diese Formel zur Berechnung des Vektorfeldes. Symbolische Werte können verwendet werden.
Verwendet: Unterprogramm Repeat
Eingabe : DIVG {a(r)}{r} CONT
Ausgabe : Divergenz und Lösungsvektor

Listing:

```
« CLLCD
  "Enter {a(r)}, {r}" 2 DISP
  HALT -> a r
  « a LIST-> DROP 'D' STO 'C'
    STO 'B' STO
    '∂X(B)' « EVAL » Repeat
    '∂Y(C)' « EVAL » Repeat +
    '∂Z(D)' « EVAL » Repeat +
    DUP 'di' STO r LIST-> DROP
    'Z' STO 'Y' STO 'X' STO di
    EVAL COLCT {X Y Z B C D di}
    PURGE
  »
»
```

Beispiel:

$$a(r)=\begin{vmatrix} 2xy\hat{}2 \\ 2x\hat{}2y \\ 4z\hat{}3 \end{vmatrix} \quad r=\begin{vmatrix} 1 \\ 2 \\ 2 \end{vmatrix}$$

Eingabe: DIVG {2xy^2 2x^2y 4z^3} {1 2 2} CONT
Ausgabe: 2*y^2+2*x^2+4*(3*z^2), 58

root

Dieses Programm sucht selbsttätig die Nullstellen einer beliebigen Funktion f(x) im Intervall [-5 5].
Dazu muß die Funktion vorher in EQ gespeichert sein.

Verwendet: Keine weiteren Module
Eingabe : f(x) in EQ, root
Ausgabe : Alle Nullstellen im Bereich [-5 5]

Listing:

```
« 1 'z' STO
  -5 4 FOR n EQ 'X' n n 1 +
      2 -> LIST ROOT
      NEXT 8 FIX
  0 8 START RND SWAP RND ==
     LASTARG 3 ROLL
     IF 1 == THEN DROP ELSE
     SWAP "NS" z 1 + 'z' STO
     z -> STR + STR-> STO
     END
  NEXT "NS" z 1 + -> STR +
  STR-> STO 'z' PURGE
»
```

Fourier-Reihen

Es soll eine Fourierreihe folgender Form entwickelt werden:

$$f(x) = \frac{4h}{\pi}\left(\sin x + \frac{1}{3}\sin 3x + \frac{1}{5}\sin 5x \ldots\right)$$

Das ist eine **Rechteckkurve**. Nehmen wir an, Sie wollen die ersten 5 Terme dieser Reihe graphisch darstellen. Dazu können Sie ein kleines Programm schreiben, welches Ihnen die einzelnen Terme berechnet, sie addiert und in EQ speichert, um sie dann zu zeichnen. Die einzelnen Terme können Sie etwas umschreiben in:

$$\frac{4h}{\pi} * \frac{\sin(i*X)}{i}$$

Nun müssen Sie nur noch sichergehen, daß i von 1 ab in Dreierschritten anwächst. Dies geht mit einer kleinen Schleife.
1 2*n FOR i ... STEP
Dann werden die Terme addiert, in EQ gespeichert und das Ergebnis ausgedruckt. Die Null am Anfang des Programms ist der erste Summand. Benennen Sie das Programm z.B mit FOUR.

Eingabe: FOUR, h, n, CONT
Ausgabe: Graph

Listing: FOUR

```
« RAD CLLCD
  Enter h,n'' 2 DISP
  HALT -> h n
  « 0 1 n 2 * FOR v
    '4*h/π*SIN(v*X)/v'
    EVAL + 2 STEP
    CLLCD STEQ DRAW
  »
»
```

UTILITIES-BIBLIOTHEK

Repeat

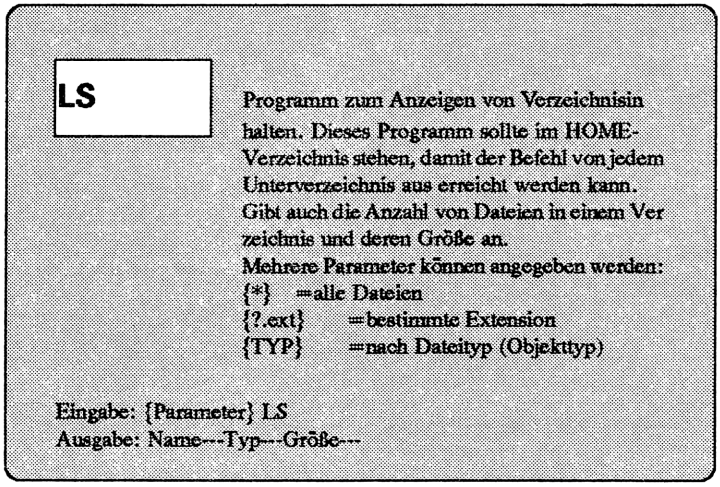
Repeat widerholt ein vorangehendes Programm so lange, bis sich an dem mit dem Programm be arbeiteten Ausdruck nichts mehr ändert

```
<< ->p
    << DO DUP p EVAL
        UNTIL DUP 3 ROLL
SAME
END
>>
>>
```

FASST

Fasst Terme vollständig zusammen

```
<<  << EXCO >> Repeat
    << COLCT >> Repeat
>>
```



Listing: 1

```
<<
 IF DEPTH 0 ≠
 THEN 1 1 1 1 0 1
1 -> opt obj siz
vsiz obl bit t d
   << opt DUP TYPE
    IF 5 ==
    THEN DUP SIZE
     IF 1 ==
     THEN 1 GET
TYPE 't' STO
       IF t 0 ==
       THEN opt
1 GET TVARS
       ELSE
        IF t 6
==
```

Listing 2

```
        THEN
opt 1 GET -> STR DUP
DUP2 2 2 SUB
        IF
"?" ==
        THEN
IF "." POS 0 ≠
THEN SIZE 3 SWAP
SUB 'opt' STO VARS
SIZE 'vsiz' STO { }
1 vsiz
 FOR v VARS v GET
-> STR DUP 'obj' STO
"." POS DUP 'd' STO
  IF 0 ≠
  THEN obj DUP
SIZE 'siz' STO d
siz SUB
    IF opt ==
    THEN VARS v
GET +
    ELSE
    END
  ELSE
  END
 NEXT
ELSE DROP2 VARS
SIZE 'vsiz' STO { }
1 vsiz
 FOR v VARS v GET
-> STR "." POS
  IF 0 ==
  THEN VARS v GET
+
  ELSE
```

Listing 3

```
  END
 NEXT
END
        ELSE
CLLCD
"Falscher Parameter"
3 DISP 0 WAIT
        END
       ELSE
        IF t
18 ==
        THEN
opt 1 GET
IF { * } 1 GET ==
THEN VARS
ELSE 400 .5 BEEP
CLLCD
"Falscher Parameter"
3 DISP 0 WAIT DROP
END
        END
       END
      END DUP
SIZE DUP
      IF 0 ≠
      THEN
'siz' STO 'obl' STO
1 siz
       FOR v
obl v GET DUP -> STR
       IF
"'LS'" ≠
       THEN
DUP DUP BYTES SWAP
DROP DUP 'bit' STO+
```

Listing 4

```
SWAP RCL TYPE
        ELSE
DUP   BYTES   SWAP
DROP
DUP 'bit' STO+ 8
        END 3
ROLL DUP ->STR SIZE
2 - DUP
        IF 8
  ≤
        THEN
8 SWAP - 1
START "-" + -1
STEP "-" +
        ELSE
DROP
        END
SWAP DUP ->STR SIZE
DUP
        IF 2
  ≤
        THEN
2 SWAP - 1
START "-" + -1
STEP "-" +
        ELSE
DROP
        END +
SWAP DUP ->STR SIZE
DUP
        IF 8
  ≤
        THEN
8 SWAP - 1
START "-" + -1
```

Listing 5

```
STEP "-" +
        ELSE
DROP
        END +
DUP SIZE
        IF 17
  ≤
        THEN
17 SWAP - 1
START 32 CHR + -1
STEP
        END
       NEXT
siz ->STR " " +
"Datei(en)" + DUP
SIZE DUP
        IF 17   ≤
        THEN 17
SWAP - 1
         START
32 CHR + -1
         STEP
        END
"mit" " " + bit
->STR + " " + "Byte"
+ DUP SIZE DUP
        IF 17   ≤
        THEN 17
SWAP - 1
         START
32 CHR + -1
         STEP
        END
      ELSE
DROP2 400 .5 BEEP
```

Listing 6

```
CLLCD
"Keine Datei gefunden"
3 DISP 0 WAIT DROP
        END
      ELSE DROP
CLLCD 400 .5 BEEP
"Bitte Option angeben"
3 DISP
"{Zahl} = Typ" 5
DISP
"{?.Zahl} = Extension"
6 DISP
"{ * } = alle" 7
DISP 0 WAIT DROP
      END
    ELSE 400 1
BEEP CLLCD
"Syntaxfehler" 3
DISP "Syntax=" 5
DISP
"{*,?.Zahl,Typ} LS"
6 DISP 0 WAIT DROP
    END
  >>
 ELSE 400 1 BEEP
CLEAR CLLCD
"Syntaxfehler" 3
DISP "SYNTAX=" 4
DISP
"{*;?:ext,TYP} LS"
6 DISP = WAIT DROP
  END
```

GRAFIK

Demo

Als Anregung diente hier das Programm Walk, welches im Handbuch des HP-48SX zu finden ist. Es zeigt eine Gestalt, welche über den Bildschirm wandert.

Eingabe : WALK
Ausgabe: DEMO

```
<<
GROB                    9                    15
E300140015001C001400E3008000C110AA009400900041002200141028 00
-> man
  << ERASE { # 0h
# 0h } PVIEW { # 0h
# 19h } PICT SWAP
man GXOR .5 WAIT 0 MAXR
   FOR i PICT {
# 0h # 0h } # 83h
# 40h BLANK REPL i
131 MOD R->B # 19h 2
->LIST PICT SWAP man
GOR .05 WAIT PICT {
# 0h # 0h } # 83h
# 40h BLANK REPL
PICT i .1 + 131 MOD
R->B # 0h 2 ->LIST
MEN GOR .1 WAIT 4 STEP
  >>
>>
```

GRAFIK FRAKTAL

JULIA

Diese fraktale Grafik erzeugt JuliaJulia -Mengen. Die Berechnung muß im RAD-Modus erfolgen und kann etwas Zeit in Anspruch nehmen.

Eingabe : x, y, cx, cy
Ausgabe: Gradfik
PPAR = (-2,-2) (2,2)
Eingabevorschlag: 0 0 0 1 JULIA oder 0 0 -1 0 JULIA

```
<< -> x y cx cy
  << ERASE 1 5000
   FOR i x cx - y
cy - -> wx wy
     <<
       CASE 'wx > 0'
        THEN '
ATAN(wy/wx)'
        END 'wx < 0
'
        THEN 'pi+
ATAN(wy/wx)'
        END 'wx = =
0'
        THEN 'pi/2
'
        END
       END -> NUM 2
/-> t
       << ' √√(wx*wx
+wy*wy)' -> NUM
```

```
        IF RAND
.5 > =
        THEN NEG
        END -> r
        < < t COS r
* 'x' STO t SIN r *
'y' STO x y R -> C
C -> PX PIXON
        > >
       > >
     > >
   NEXT
  > >
> >
```

MARTIN

Das Programm Martin erzeugt eine Girlande nähnliche Strucktur.
ACHTUNG: Dieses Programm muß mit ATTN abgebrochen werden.

Eingabe : a, b, c
Ausgabe : Grafik
PPAR: : (-160, -190) (55,25)
Eingabevorschlag: -137 17 -4 MARTIN

MARTIN

```
   < < 0 0 -> a b c x
y
    < < ERASE
     WHILE '1 = = 1
'
     REPEAT x y
R-> C C-> PX PIXON a x
```

```
y x b * c - ABS √x
SIGN * - 'x' STO -
'y' STO
      END
    > >
  > >
```

PPAR {
(-160,-190) (55,25)
X 0 (0,0) CONIC Y }

CONETT

Der Bildungsalgorithmus dieser Grafik beruht auf einem repetitiven Algorithmus zum Berechnen jedes einzelnen Bildpunktes.. Dieses Programm erzeugt nach einigen Minuten ein Interferenzmuster.

Verwendet: Unterprogramm CP (GROB-File)
Eingabe : l, r, o, u
Ausgabe : Grafik
PPAR: (0,0) (130,63)
Eingabevorschlag: -0.25 2 -1.5 1.5

CP

GROB 131 64

```
EFFFFFFFFFFFFFF1000008FFFF700087 0EFFFFFFFFFFFFFFF100000CFFFF100060E30
000FFFFFFFFFFF100000EFFF70000000000000CFFFFFFFF100000FFFF300000000
0000000CFFFFFFF00000CFFFF0000000000000000FFFFFFF700000EFFF30000000
00000000000EFFFFF300008FFF70000000000000000EFFFFFF10000EFFF1000000
000000000000EFFFFF700008FFF70000000000000000FFFFFF30000EFFF000000
0000000000008FFFFFF0000CFFF30CFFFFF700000000000CFFFFF30000FFF70EFFFF
FFFF0000000000EFFFF70000EFF70EFFFFFFFFF3000000008FFFFF10008FF70EFFFFFF
FFFF700000000EFFFF70000FF70EFFFFFFFFFF700000008FFFFF0000EF70EFFFFFFFF
FFF30000000EFFFF1000CF70EFFFFFFFFFFFF1000000CFFFF70008F70EFFFFFFFFFF
FF70000000FFFFF0000F70EFFFFFFFFFFFF1000000EFFFF1000E70EFFFFFFFFFFFF
7000000CFFFF3000C70EFFFFFFFFFFFFF1000008FFFF700087 0EFFFFFFFFFFFFFF3
000000FFFFF000070EFFFFFFFFFFFFFF000000EFFFF000070EFFFFFFFFFFFFFF100
```

000CFFFF100060EFFFFFFFFFFFFFFF3000008FFFF100060EFFFFFFFFFFFFFFF700000
8FFFF300040EFFFFFFFFFFFFFFF7000000FFFF300040EFFFFFFFFFFFFFFFF000000FF
FF700040EFFFFFFFFFFFFFFF000000FFFF700040EFFFFFFFFFFFFFFF100000EFFF7
00040EFFFFFFFFFFFFFFFF100000EFFF700040EFFFFFFFFFFFFFFF100000EFFF7000
40EFFFFFFFFFFFFFFF100000EFFF700040EFFFFFFFFFFFFFFF000000FFFF700040E
FFFFFFFFFFFFFFF000000FFFF700040EFFFFFFFFFFFFFFF7000000FFFF300040EFFF
FFFFFFFFFFF7000008FFFF300040EFFFFFFFFFFFFFFF3000008FFFF100060EFFFFFF
FFFFFFFFF100000CFFFF100060EFFFFFFFFFFFFFFF000000EFFFF000070EFFFFFFFF
FFFFF3000000FFFFF000070EFFFFFFFFFFFFFF1000008FFFF700087 0EFFFFFFFFFFF
FF7000000CFFFF3000C70EFFFFFFFFFFFFF1000000EFFFF1000E70EFFFFFFFFFFFF7
0000000FFFFF0000F70EFFFFFFFFFFFF1000000CFFFF70008F70EFFFFFFFFFFF3000
0000EFFFF1000CF70EFFFFFFFFFF700000008FFFFF0000EF70EFFFFFFFFF70000000
0EFFFF70000FF70EFFFFFFFF3000000008FFFFF10008FF70EFFFFFFF0000000000EF
FFF70000EFF70CFFFFF700000000000CFFFFF30000FFF70000000000000000008FFF
FFF0000CFFF300000000000000000FFFFF30000EFFF000000000000000000EFFFF
F700008FFF70000000000000000EFFFFFF10000EFFF10000000000000000EFFFFFF
300008FFF70000000000000000FFFFFFF700000EFFF300000000000000CFFFFFFF0
0000CFFFF000000000000CFFFFFFFF100000FFFF30000E30000FFFFFFFFFFF1000
00EFFF700000EFFFFFFFFFFFFFFF100000CFFFF10006000000000000000000000000
00000000000

CONETT

```
   < < 130 63 -> 1 r                PPAR { (0,0)
u o x y                             (130,63) X 0 (0,0)
     < < ERASE 1 x                  CONIC Y }
       FOR i 1 y                    END
         FOR j 1 r
1 - i * x / + 2 ^ u
o u - j * y / + 2 ^
+ IP 2 MOD 0
           IF SAME
           THEN i
j R-> C C-> PX PIXON
           END
         NEXT
       NEXT
     > > GRAPH
   > >
```

ERGÄNZUNGEN

DER HP-DEBUGGER

Bei der Übergabe von fertig eingegebenen Anwenderprogrammen an den Speicher des HP-48SX wird eine Syntax-Überprüfung des Übergebenen Programmes durchgeführt. Tritt bei der Prüfung ein Fehler auf, so wird die Übergabe abgebrochen, der Cursor auf die Fehlerstelle gesetzt und diese markiert. Zusätzlich wird eine qualifizierende Meldung angezeigt. Der Fehler kann behoben und das Programm erneut an den Speicher weitergereicht werden. Nur Programme, welche keine Syntaxfehler enthalten werden gespeichert.
Es gibt allerdings Fehler, welche nicht durch einen Syntaxfehler sondern z.B. durch einen logischen Fehler hervorgerufen werden.

Der HP-48SX hat ein eingebautes Programm, welches es erlaubt auf komfortable Art und Weise selbst-geschriebene oder andere Anwender-Programme auf solche Fehler zu untersuchen. Dieses Programm nennt sich **Debugger**.
Das Programm wird mit dem Befehl *DBUG* im Menü **PRG** *CTRL* aufgerufen.
Syntax: **'Name' 'DBUG'**

Wenn Sie den Namen eines Anwenderprogrammes in den Stack geben und den Debugger aufrufen, startet das Programm und die Programmausführung wird **VOR** dem 1. Objekt unterbrochen. Jetzt kann mit dem Befehl *SST* (PRG CTRL) das Programm Schritt für Schritt durchgeprüft werden.
Verfolgen Sie schrittweise den Programmablauf, wobei jeder Befehl in der obersten Anzeigezeile angezeigt wird.
Tritt ein Fehler bei der Verarbeitung eines solchen Einzelbefehls auf, wird eine Fehlermeldung ausgegeben und kann in Verbindung mit der Stack-Konfiguration auf den Fehler hinweisen. Daraufhin kann der Fehler behoben werden.

Auch der Programmbefehl *HALT* oder ein gleichwertiger Befehl können den Rechner in den Debug-Modus versetzen, wenn die Taste *SST* zur Programmfortsetzung benutzt wird.
Dazu schreiben Sie diesen Befehl einfach an die Verdächtige Programmstelle und starten das Programm erneut.

FEHLERERKENNUNG UND FEHLERBEHEBUNG

Es kann nach Eingabe eien Programmes vorkommen, daß das Programm nach seinem Starten plötzlich abgebrochen oder unterbrochen wird und eine Fehlermeldung in der Anzeige erscheint.

Gehen Sie dann bitte so vor:

1. Die Meldung des HP-48 gibt Auskunft über die Art des Fehlers. Lesen Sie dazu bitte die entsprechenden Kapitel über Fehlermeldungen.

2. Betätigen Sie eine beliebige Taste zum Anzeigen des normalen Stacks.

3. Die Anordung des Stacks in Verbindung mit der Fehlermeldung kann Aufschluß über die nicht durchführbare Operation geben. Suchen Sie in diesem Falle die Stelle im eingegebenen Programm:

4a) Dies könne Sie, indem Sie das entsprechende Programm mit **'Name' ENTER +-> VISIT** aufrufen. Sie befinden sich jetzt im Editiermodus. Stellen Sie den Cursor in die entsprechende Programmzeile. Bedienen Sie sich dazu der Cursortasten.

5a) Vergleichen Sie das eingegebene Programm an dieser Stelle mit dem vorgegbenen Listing.

6a) Haben sie einen Eingabefehler entdeckt, verbessern Sie Ihn z.B. mit *DEL*, INS oder durch Überschreiben aus und betätigen **ENTER**. Das geänderte Programm wird wieder in der Variablen gespeichert.

7a) Starten Sie das Programm erneut. Tritt wieder ein Fehler auf, wiederholen Sie die obere Prozedur.

Sollte Ihre Eingabe richtig sein oder wissen Sie mit der ausgegebenen Meldung nichts anzufangen, können Sie das Programm mit dem HP-Debugger schrittweise überprüfen.

4b) Editieren Sie das Programm mit **+-> VISIT**.

5b) Fügen Sie in das entprechende Programm oder Unterprogramm am Anfang oder in der Nähe der verdächtigen Fehlerstelle den Befehl *HALT* ein. Beachten Sie, daß der Befehl nur in dem Programm oder Untrprogramm gilt, in dem er steht. Ist in diesem Programm ein Unterprogramm enthalten, so wird das Unterprogramm nicht schrittweise durchlaufen, wenn kein HALT-Befehl darin vorkommt.

6b) Geben Sie das Programm mit **ENTER** zurück in seinen Namen.

7b. Starten sie das Programm neu. Es wird an der "Haltestelle" automatisch unterbrochen.

8b) Wählen Sie den Befehl *SST* (Single Step) zum schrittweisen Abarbeiten (debuggen) des Programmes. So könne Sie die Fehlerstelle genau lokalisieren und an Hand der Stack Konfiguration und der Fehlermeldung auf den Programmfehler schließen.

9b) Korrigieren Sie Ihn und starten Sie das Programm nach Betätigen von **ENTER** erneut.

Sind alle Fehler verbessert, entfernen Sie den Befehlt *HALT* wieder aus dem entsprechenden Programm.

MENÜORGANISATION

Es ist sinnvoll die einzelnen Programme in typischen Verzeichnissen zusammenzufassen und diese Verzeichnis-Namen im VAR-HOME-Verzeichnis zu speichern. Mir dieser Methode haben Sie einen viel schnelleren Zugriff auf die entsprechenden Programme, da Sie das Hauptmenü (HOME-Verzeichnis) nicht ständig durchblättern müssen.

Dfinieren sie schon vor der Eingabe von Programmen den entsprechenden Verzeichnisnamen des Verzeichnisses in dem Sie Ihre Programme abspeichern wollen. Natürlich könne sie diese Organisation auch nachträglich durchführen.

Beispiel:
Sie wollen alle Programme, welche statistiche Auswertungen durchführen in dem Verzeichnis STATS speichern.

Geben Sie den Namen STATS in den Stack und erzeugen Sie mit *CRDIR* das Verzeichnis. Der Name STATS sollte nun im Hauptverzeichnis Ihres Rechners zu finden sein.

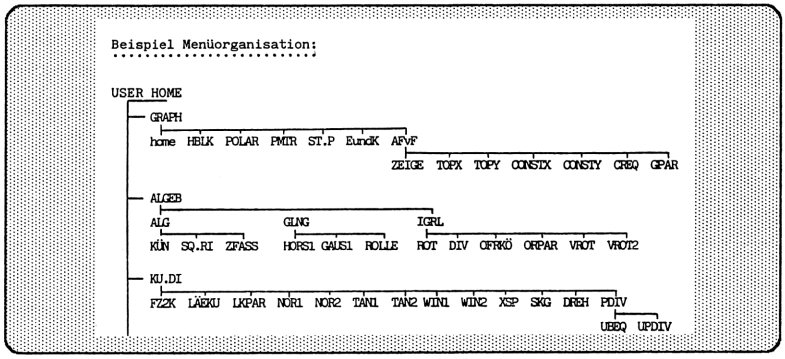
Um in das Verzeichnis zu wechseln betätigen Sie im VAR-Verzeichnis den Softkey *STATS*. Es erscheint ein leeres Menü, das Sie sich nun im Menü HOME\STATS befinden.

Nun können Sie alle Daten und Programme in dieses Menü speichern. Wollen Sie in das STATS-Menü keine Programme mehr aufnehmen, so kehren Sie mit dem Befehl **HOME** oder **<-+ UP** in das Hauptverzeichnis zurück.

Definieren Sie das nächste Unterverzeichnis und gehen Sie so vor wie oben beschrieben.

Befinden sie sich in einem Untermenü, wie z.B. in HOME\STATS können Sie natürlich darin wieder ein Untermenü erzeugen. So behalten Sie immer die Übersicht über Ihre eingegebenen Daten.

Folgendes Schaubild zeigt Ihnen einen Menüvorschlag für Ihre Programme.



Nachträgliches Ordenen von Daten

Haben Sie schon alle Programme gespeichert und wollen Sie sie nun nachträglich umorganisieren, gehen Sie wie folgt vor:

Daten verschieben:

Rufen Sie die entsprechende Variable durch Eingabe Ihres Namens und **RCL** (Recall) in den Stack zurück. Erzeugen Sie, wenn nötig, ein neues Zielverzeichnis. Wechseln sie in das entsprechende Verzeichnis und speichern Sie das Programm dort neu ab. Bewegen Sie sich wieder zurück ins ursprüngliche Menü und löschen Sie dort den alten Variableneintrag.

Sie können auch ganze Gruppen von Variablen in einer Liste zusammenfassen und dann verschieben.

Verzeichnisse verschieben oder umbenennen

Wollen Sie ein Verzeichnis verschieben, so rufen Sie es durch die Eingabe seines Namens und **RCL** (Recall) in den Stack zurück. Der HP-48 erkennt selbsttätig, daß es sich bei dem Variableninhalt um ein Verzeichnis handelt. Wechseln Sie in ein anderes Verzeichnis und speichern Sie die Verzeichnisvariable dort ab. Löschen Sie das alte Verzeichnis.

Soll ein Verzeichnis umbenannt werden, rufen Sie es ebenfalls mit RCL in den Stack zurück, vergeben einen neuen Namen und speichern es wieder. Löschen Sie den alten Variablennamen.

Gehen Sie bei allen Programmen oder Verzeichnissen so vor.

DIE WICHTIGSTEN ZUSATZMODULE IM HP-48SX SCHNITTSTELLENPAKET

DAS DATENÜBERTRAGUNGSPROGRAMM KERMIT

Als Option für den HP-48SX ist ein Schnittstellenpaket erhältlich, das es gestattet zwischen dem Taschencomputer und einem Personal Computer bzw. einem Apple Macintosh zu kommunizieren.
Hauptbestandteil des Schnittstellenpaketes ist ein sogenanntes Übertragungsprotokoll KERMIT. Dieses Programm ist in der Lage den Zeichensatz des HP-48SX in einen Computerzeichensatz umzuwandeln.
Dazu wird zwischen dem HP-48SX und dem Computer ein Schnittstellenkabel angeschlossen. Dieses Kabel verbindet den vierpoligen Stecker des HP-48SX mit der seriellen Schnittstelle Ihres IBM-compatiblen Computers bzw. der Apple Schnittstelle. Dazu sind auch Adapterstecker im Paket enthalten.
Das Schnittstellenpaket für IBM-compatible Computer hat die HP-Best.-Nr. 82208A.
Das Schnittstellenpaket für Apple Macintosh hat die Best.-Nr. 82209A.

Schließen Sie das Schnittstellenkabel an Ihren Computer an und starten Sie KERMIT auf Ihrem Computer.
Sie sehen nun am Bildschirm die Kermit-Befehlszeile. In ihr können Kermit-Befehle eingegeben werden.
Configurieren Sie KERMIT auf folgende Weise:

Definieren Sie den Port (COM1 oder COM2) bei IBM-compatiblen PCs. Stellen Sie dann die Baudrate auf den für den Computer maximal zulässigen Wert (siehe Computerhandbuch).

Nun ist KERMIT betriebsbereit.
Im Anhang finden Sie eine Kermit-Befehlsliste für IBM-compatible PCs.

Gehen Sie nun auf Ihrem Taschencomputer ins Menü **<-+ I/O** und rufen Sie mit *SETUP* die I/O-Betriebsparameter Ihres HP-48SX auf.
Stellen Sie den SETUP auf die gleichen Werte, wie auf Ihrem Computer. Die einzelnen Einstellungen können durch einen Druck auf die entsprechnde Menütaste geändert werden.

Das sind in der Regel:

IR/wire:	wire
ASCII/binary:	ASCII
baud:	9600
parity:	none 0
checksum-type:	3
translate code	1

Alle Gundeinstellungen des HP-48-KERMIT werden innerhalb einer Liste in der Variablen **IOPAR** gespeichert.
Struktur:
{Baudrate Parität Empfangstakt Sendetakt Prüfsumme Übersetzungcode}
{BAUD PARIT 0 0 CKSM TRAN}

Die zwei Übertragungsmöglichkeiten von KERMIT:

Zur Datenübertragung stehen zwei Betriebsmodi zu Verfügung:
Starten Sie vor der Datenübertragung Ihr Kermit-Programm.
Alle Befehle des HP-48-KERMIT finden Sie im Menü **<-+ I/O**.

Betriebsmodus LOKAL/LOKAL:
Wenn Sie eine Variable zwischen Taschencomputer und Computer übertragen wollen können Sie in diesem Modus den den HP-48SX-Befehl RECV verwenden. Die übertragene Variable wird dann unter dem Namen den der Sender mitteilt gespeichert.

Wollen Sie den Namen bei der Übertragung ändern, geben Sie einen neuen Namen ein und betätigen Sie *RECN*. Das Objekt wird unter dem angegebenen Namen übertragen. Wenn Sie eine oder mehrere Variablen (auch Verzeichnisse) übertragen wollen, geben Sie den Namen oder eine Liste mit Namen ein und geben den Befehl *SEND*.

Betriebsmodus LOKAL SERVER:
Ein Server ist ein Gerät, das von einem anderen angeschlossenen Gerät fernbedient wird. Soll ein Gerät als Server arbeiten, bringen Sie es in den Server-Modus. Am HP geschieht dies mit dem Befehl *SERVE* oder **+-> I/O.**
Wollen Sie eine Datei zum Server übertragen, geben Sie den Namen der Variablen ein und betätigen *SEND*.
Wollen Sie eine Datei von Server holen, geben Sie den Namen und den Befehl *KGET* ein.
Wollen Sie diesen Namen bei der Übertragung ändern, geben Sie eine Namensliste mit einer sogenannten Unterliste in den Stack.
Syntax:
{{alter Name neuer Name}{alter Name neuer Name } Name3 Name4 }
Name3 und Name4 werden unverändert übertragen.

Der Server-Modus wird mit *FINIS* beendet.

Einschränkungen von Kermit:

Dateinamen

Variablen werden auf dem HP-48SX und auf einem Computer unterschiedlich benannt. Das kann bei der Übertragung vom Computer zum HP-48SX zu Problemen führen.
Dies geschieht wenn:

a) Der Dateiname Zeichen enthält, die nicht der Syntax für Variablennamen entsprechen - z.B. # {} : usw.. Der HP-48 zeigt eine Fehlermeldung an und bricht die Übertragung ab.

b) Der Dateiname ist gleich einem internen HP-Befehl. In diesem Fall hängt der Taschencomputer eine 1 als Erweiterung (Extension) an - z.B. TAN.1.

c) Der Dateiname entspricht einem schon vorhandenen Variablennamen. Hier wird in der Regel ebenfalls eine Zahl als Erweiterung angehängt. Ist Flag -36 gesetzt wird die Variable jedoch überschrieben.

Bei der Datenübertragung vom HP-48SX an den Computer ergibt sich ein Übertragungsfehler, wenn die vom HP-48 gesendete Variable nicht den Dateidefinitionen des Computers entspricht. Stellen Sie dies in der Übertragung sicher.

Tritt ein Übertragungfehler auf, kann mit *KERR* der Text des letzten empfangenen Datenpaketes (Fehlerpaket) angezeigt werden.

Senden von Befehlen an einen Server.

Mit sogenannten PKT-Befehlen läßt sich ein Server, welcher sich im Kermit-Modus befindet, fernsteuern.
Syntax:
"Kermitbefehl" "Pakettyp" PKT

Kermit-Befehle sind z.B. D (DIRECTORY), E (ERASE) usw..
Pakettyp ist bei Befehlen in der Regel G.

ÜBERTRAGEN VON DATEN OHNE KERMIT-PROTOKOLL

Der Befehl *XMIT* sendet die Zeichenkette, die in der Stackebene 1 steht, ohne KERMIT. Bei erfolgreicher Übertragung wird eine 1 im Stack abgelegt. Bei einem Übertragungsfehler erscheint eine 0 und der nicht übertragene Teil des Textes wird in Ebene 2 dargestellt.
Der Befehl *SBRK* sendet einen *BREAK* an den seriellen Anschluß.

!Achtung! Der Befehl *XMIT* überprüft nicht die Vollständigkeit der übertragenen Daten! Fügen Sie an das Datenende der zu sendenden Daten eine Prüfsumme ein und vergleichen Sie sie mit der Prüfsumme der empfangenen Daten.
Ist die Uhr des HP-48SX zu sehen, sollten Sie NICHT eine Übertragungsrate von 9600 Baud wählen, da dies zu Übertragungsfehlern führen kann.

ASCII- und Binärmodus

ASCII

Empfangene Daten, welche mit einem Computer angezeigt, bearbeitet oder gedruckt werden sollen, müssen im ASCII-Modus übertragen werden.
Bei der Übertragung wird die Zeichenfolge **%%HP: Modi ;** an den Anfang der gesendeten Datei geschrieben. Bei einer Übertragung an den HP-48 werden die Betriebsmodi des Taschencomputers an Hand dieser Zeichenfolge für die Dauer der Übertragung abgeändert.
Modi: T (Umsetzungscode), A (Winkelmodus) und F (Dezimalzeichen)

Bemerkung:
Übertragung HP-48 -> Computer
Bei Umsetzungcode 1, 2 oder 3 werden alle Zeilenvorschub-Zeichen (Line Feed) in Druckkopfrücklauf - und Zeilenvorschub-Befehl (Carriage Return - Line Feed) umgewandelt.
Übertragung Computer -> HP-48
Bei Umsetzungcode 1, 2 oder 3 werden alle CR-LF Zeichen in LF-Zeichen umgewandelt.

Zeichen-Konvertierung (TRANSIO)

Bestimmte HP-Zeichen werden von vielen Computerprogrammen nicht dargestellt:

a) HP-Zeichen mit Zeichennummern von 128 - 159 können ohne besondere Programme nicht dargestellt werden.

b) HP-Zeichen mit den Nummern von 160 - 255 können nur mit Computern die den ISO 8859-Zeichensatz besitzen verarbeitet werden.

Siehe Tabelle Sonderzeichen im Benutzerhandbuch Nr. 2

Der Umsetzungscode (Translation Code) kann mit dem Befehl TRANSIO bestimmt werden. Dabei werden einige HP-Zeichen ab Nr. 127 konvertiert.
Mit dieser Konvertierung ist es möglich, diese Zeichen in Ihren Computer einzugeben bzw. darzustellen.

translate code: 1
Keine Übersetzung der HP-Zeichen
translate code: 2/3
Daten HP-48 -> Computer: Jedes Zeichen \ wird durch \\ ersetzt.
Daten Computer -> HP-48: Zeichenfolgen, die mit \ beginnen, bleiben unverändert.

Ausnahmen:

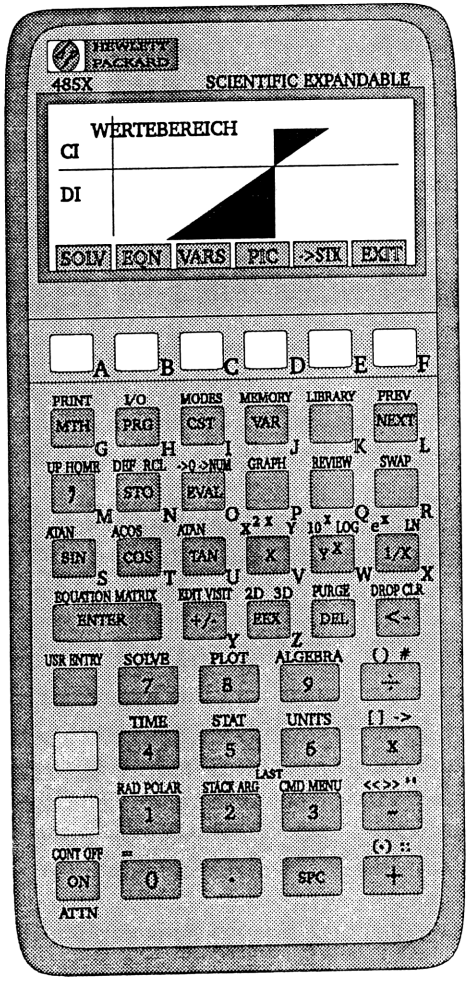
a) Sie sind identisch mit einer Sequenz in der Tabelle.

b) Beim Umsetzungscode 2 folgt auf dem \ eine 3-stellige Ziffer im Bereich von 000 bis 159.

c) Beim Umsetzungscode 3 folgt dem \ eine 3-stellige Ziffer im Bereich von 000 bis 255.

Sollte bei der Übertragung von Daten zwischen HP-48SX und einem Computer ein Fehler auftreten der sich nicht beheben läßt, so können Sie versuchen den gesamten Inhalt der zu übertragenden Variablen in einen String (" ") zu setzen. Achten sie darauf,
daß Sie danach alle noch im Variableninhalt (Programm) verbliebenen String-Begrenzungszeichen entfernen. Eine Datenübertragung mit Sub-Strings kann zu weiteren Problemen führen. Nach der Übertragung können Sie die geänderten Zeichen wieder austauschen.

!Achtung! Unter Umständen stellt Ihr Computerprogramm (Editor) bestimmte Zeichen, welche nicht im Zeichensatz des Editors bzw. Computers enthalten sind oder eine andere Position in Ihrer Zeichensatztabelle besitzen, am Bildschirm anders dar als auf dem HP-48.
In einem solchen Fall können Sie beruigt sein, da bei der Rücksendung der Daten mit dem entsprechenden TRANSIO-Code die Originalzeichen wieder erscheinen.
Verwenden Sie immer den gleichen Übersetzungscode bem Sichern und Rücksichern der Daten! Entscheiden Sie sich dauerhaft für einen Übersetzungscode, um Mißverständnisse auszuräumen!


HEWLETT PACKARD
48SX
SCIENTIFIC EXPANDABLE
WERTEBEREICH
CI
DI
SOLV
EQN
VARS
PIC
->STK
EXIT
A
B
C
D
E
F
PRINT
MTH
G
I/O
PRG
H
MODES
CST
I
MEMORY
VAR
J
LIBRARY
K
PREV
NEXT
L
UP HOME
M
DEF RCL
STO
N
->Q ->NUM
EVAL
O
GRAPH
P
REVIEW
Q
SWAP
R
ATAN
SIN
S
ACOS
COS
T
ATAN
TAN
U
x²
X
V
10^x LOG
y^x
W
e^x LN
1/X
X
EQUATION MATRIX
ENTER
EDIT VISIT
+/-
Y
2D 3D
EEX
Z
PURGE
DEL
DROP CLR
<-
USR ENTRY
SOLVE
7
PLOT
8
ALGEBRA
9
( ) #
÷
TIME
4
STAT
5
UNITS
6
[ ] ->
x
LAST
RAD POLAR
1
STACK ARG
2
CMD MENU
3
<<>> ''
~
CONT OFF
ON
ATTN
=
0
.
SPC
( ) ::
+

DAS ÜBERSETZUNGSPROGRAMM GROB2TIF

Auf der **Schnittstellendiskette** befindet sich ein Programm mit dem Namen GROB2TIF.EXE. Dieses Programm übersetzt das Grafikformat einer Variablen, welche ein Graphikobjekt enthält (GROB-File) in das TIF (TAG IMAGE FILE) - Format.
Das TIF-Format (TIFF) ist eine Bit-Muster Graphik (Rastergraphik), welche als eine Folge von Bildpunkten (Pixel) beschrieben ist. Extensions für Bitmuster-Graphiken sind z.B. .TIF, .TGA, .PCX, usw..

Syntax: GROB2TIF < GROBDATEI> TIFFDATEI

!Achtung!. Dieser Befehl gilt für IBM-kompatible PCs und muß genau wie oben eingegben werden (Leerstellen beachten).

Vorgehensweise:
Erzeugen Sie auf Ihrem HP-48SX ein Graphikobjekt (GROBFILE).
Dies kann mit *LCD->* oder mit **STO** aus der Graphikumgebung heraus geschehen.
Dann speichern Sie dieses Objekt in einer Variablen.
Übertragen Sie mit KERMIT diese Datei auf Ihren Computer.
Wandeln Sie dann das Graphikobjekt in eine TIFF-Datei um.
Diese Datei kann von vielen Computerprogrammen als Graphik eingelesen und bearbeitet werden.

DAS PROGRAMM INPRINT

Das Programm INPRINT Ihres Schnittstellen-Paketes ist ein Programm, welches Druckbefehle von anderen HP-Taschencomputern über die Infrarotschnittstelle empfängt.
Damit ist es möglich z.B. von HP-28 bzw. HP-41 Daten an den HP-48SX zu senden. Dabei wird z.B. der Zeichensatz ROMAN 8, welcher auf dem Drucker HP 82240 ausgedruckt werden soll, in den ISO 8859 LATIN 1 Zeichensatz des HP-48SX übersetzt.

Vorgehensweise:
Übertragen Sie die Datei INPRT.EXE auf Ihren HP-48SX. Die Übertragung der Datei **muß** im Binärmodus geschehen.
Stellen Sie auf dem zu sendenden Taschencomputer den einfachen Zeilenabstand ein. Stellen Sie dann die Taschencomputer (Sender und HP-48SX) mit Ihrer Stirnseite gegenüber. **Die Infrarot-Sendediode muß sich gegenüber dem Pfeil am oberen Rand des HP-48 befinden und sollte nicht weiter als einen Zentimeter vom HP-48SX entfernt sein.**
Starten Sie das Programm *INPRT* auf Ihrem HP-48 und geben Sie auf Ihrem sendenden Taschencomputer den Druckbefehl. Nach dem Start von INPRT erscheint auf dem HP-48 Display eine leere Anzeige. Nach erfolgreichem Empfang der Druckdaten erscheint in Ebene 1 eine Statusmeldung (0 oder 1) und in Ebene 2 werden die empfangenen Daten dargestellt.
Die Statusmeldung 0 bedeutet, daß bei der Übertragung Fehler festgestellt wurden. Wiederholen sie die Übertragung bzw. überprüfen Sie die angekommenen Daten.

!Achtung! Der Zeichensatz und Befehlssatz des HP-48SX ist **NICHT** hundertprozentig kompatibel zu den anderen Taschencomputermodellen. Programme, welche auf anderen Rechnern erstellt wurden, dürfen erst **nach Anpassung der Befehle** verwendet werden. Es kann sonst zu einer Rechnerfehlfunktion kommen, wenn ein solches Programm ohne Überarbeitung gestartet wird.

WEITERE SCHNITTSTELLEN PROGRAMME

EPSPRINT.LIB ist eine Bibliothek, mit der Sie Graphikobjekt eauf einem Epsondrucker wie z.B. dem FX-80 oder dem FX-85 ausdrucken können.
Dabei läßt sich der Vergrößerungsfaktor und die Zeichendichte der Graphik angeben.

PCLPRINT.LIB ist eine Bibliothek zum Ausdrucken eines Graphikobjektes auf einem PCL-Drucker. PCL-Drucker sind z.B. der HP ThinkJet, der HP DeskJet oder der HP LaserJet.
Die Auflösung und der Vergrößerungsfaktor der gedruckten Graphik lassen sich ebenfalls einstellen.

DER BEFEHL WSLOG

Der Befehl WSLOG ist ein sogenannter unkommentierter Befehl des HP-48SX. Der Befehl führt ein Protokoll über jeden erfolgten fehlerhaften Warmstart des HP-Taschencomputers.
Ein Warmstart wird z.B. durch das Einschalten des Rechners mit **ON** ausgelöst. WSLOG zeichnet automatisch das Datum, die Zeit und die Ursache jedes Warmstarts auf.
Dabei wird durch Löschen des Flag -42 auf das US-Datums-Format umgeschaltet. Somit ist WSLOG ein gutes Werkzeug, die Ursache einer Rechner-Fehlfunktion festzustellen.

Syntax: **WSLOG**
Ausgabe: **"Fehlercode-Datum-Zeit" jedes Fehler-Warmstarts**

Fehlercodes:

0	Das Warmstart-Protokoll wurde durch drücken von **ON-SPC** gelöscht und der Rechner mit **ON** reaktiviert. ON SPC versetzt den HP-48 in einen *"Koma-Modus."* Dabei werden die Rechnerfunktionen "eingefroren" (die Systemuhr bleibt stehen) Mit der Taste **ON** wird der HP-48SX reaktiviert und das Fehlerprotokoll gelöscht.
1	Das Interrupt-System verzeichnete an den Batteriekontakten sehr schlechten Batteriezustand (Das ist nicht mit einer geringen Systemspannung gleichzusetzen) und versetzte den Rechner in einen *"Tiefschlaf-Modus"* (die Systemuhr läuft weiter). Wenn, nach einem Wiederherstellen der Batterie, **ON** betätigt wird, wird das System *"warmgestartet"* und mit dem Fehlercode 1 festgehalten.
2	Hardwarefehler wärend der IR-Übertragung (Timeout)
3	Adresse 0 überlesen

4 Systemzeit-Modul zusammengebrochen

5 Bei einer Reaktivierung aus dem Tiefschlafmodus wurden keine Veränderungen am PORT-Status, aber einige Veränderungen der Daten auf einer oder beider Karten

6 Frei

7 Ein fünf-stelliges CMOS-TEST-Wort im RAM konnte nicht gelesen werden (Checksum-Error). Dieses Wort wird bei jedem Interrupt getestet aber es wird nur als Anzeiger für mögliche RAM-Fehler verwendet.

8 Bei der Konfigurierung des Systems über Device-Treiber wurden Unstimmigkeiten festgestellt:

a) Das Interrupt-System stellte fest, daß einer der fünf Device-Treiber nicht arbeitet.

b) Unerwartete Device-Kennungen wurden, während versucht wurde 3 von 5 Devices zu konfigurieren (Port 1, Port 2 und Xtra), festgestellt.

c) Wie b) aber Fehler während Tiefschlaf-Reaktivierung

9 Fehlerhafte Termin-Alarm-Liste

A Frei

B Erweiterungskarte herausgezogen oder Karte locker

C Hardware-Reset (z.B. elektrostatische Entladung oder Benutzer Reset)

D Eine erwartete Fehlerbehandlungs-Routine wurde im Datenstrom nicht gefunden.

E Fehlerhafte Konfigurationstabelle (Checksum-Error bei den Konfigurationsdaten)

F System RAM-Karte entfernt

ANHANG

Flag	Beschreibung	Vorgabe
-01	Nur Hauptwert anzeigen ein	0
-02	Symb. Darstellung von Konstanten aus	0
-03	Num. Darstellung von Ergebnissen aus	0
-04	--	
-05--10	Binärzahlenlänge 64 Bit	1
-11--12	Andere Zahlenbasis als DEC	0
-13--14	--	
-15--16	Anderes Koordinatensystem als Rechteck	0
-17--18	Anderer Winkelmodus als GRAD	0
-19	Komplexmodus ein	0
-20	Bereichsunterschreitung liefert Fehler	0
-21	Überlauf liefert Fehler	0
-22	Unendliches Ergebnis liefert 9.99E499	0
-23	Anzeige negative Bereichsunterschreitung	0
-24	Anzeige positive Bereichsunterschreitung	0
-25	Anzeige Überlauf	0
-26	Anzeige unendliches Ergebnis	0
-27--29	--	
-30	Grafische Darstellung links u. rechts	0
-31	Verbinden von Punkten zu einer Kurve	0
-32	Grafikcursor invers	0
-33	E/A-Geräte-Schnittstelle IR	0
-34	Druck-Schnittstelle seriell	0
-35	E/A-Datenformat BIN	0
-36	Datei-Überschreibschtz aus	0
-37	Zweizeiliger Druck	0
-38	Zeilenvorschub aus	0
-39	E/A-Meldungen unterdrücken	0
-40	Uhr immer anzeigen	0
-41	Zeitformat 24 Stnden	0
-42	Datumsformat Tag.Monat.Jahr	0
-43	Wiederholungstermine nicht neu ansetzen	0
-44	Bestätigte Alarme bleiben gespeichert	0
-45--48	Anzahl Dezimalstellen einstellen	0
-49--50	Zahlenformat FIX-Modus aus	0
-51	Dezimaltrennzeichen: ,	0
-52	Einzeilige Anzeige	0
-53	Klammern in Rechenoperationen ein	0
-54	--	
-55	Letztes Argument nicht sichern	0
-56	Akustisches Fehlersignal aus	0
-57	Akustisches Alarmsignal aus	0
-58	Ausführliche Meldungen aus	0
-59	Schnelle Kataloganzeige aus	0
-60	Verriegelung Alpha-Modus durch 1xALPHA	0
-61	Verriegelung Benutzer-Mod mit 1x USR	0
-62	Benutzermodus ein	0
-63	Benutzerdefiniertes Enter ein	0

Standardmenüs des HP-48 und zugehörige Menünummern
Übersicht:

Menü#	Menüname
0	LAST MENU
1	CST
2	VAR
3	MTH
4	MTH PARTS
5	MTH PROB
6	MTH HYP
7	MTH MATR
8	MTH VECTR
9	MTH BASE
10	PRG
11	PRG STK
12	PRG OBJ
13	PRG DISP
14	PRG CTRL
15	PRG BRCH
16	PRG TEST
17	PRINT
18	I/O
19	I/O SETUP
20	MODES
21	MODES Customization
22	MEMORY
23	MEMORY Arithmetic
24	LIBRARY
25	PORT 0
26	PORT 1
27	PORT 2
28	EDIT
29	SOLVE
30	SOLVE SOLVR
31	PLOT
32	PLOT PTYPE
33	PLOT PLOTR
34	ALGEBRA
35	TIME
36	TIME ADJST
37	TIME ALARM

HP 48 SX

#(hex)	Bedeutung
001	Insufficient Memory
002	Directory Recursion
003	Undefined Lokal Name
004	Undefined XLIB Name
005	Memory Clear
006	Power Lost
008	Invalid Card Data
009	Objekt in use
00A	Port Not available
00B	No Room in Port
00C	Objekt Not in Port
101	No Room to Safe Stack
102	Can't Edit Null Char.
103	Invalid User Funktion
104	No Current Equation
106	Invalid Syntax
124	LAST CMD Disabled
126	HALT not allowed
128	Wrong Argument Count
129	Circular Reference
12A	Directory Not Allowed
12B	Non-Emty Directory
12C	Invalid Definition
12E	Invalid PPAR
12F	Non-Real Result
130	Unable to Isolate
131	No Room to Show Stack
135	Out of Memory
13C	Name Conflict
201	Too Few Arguments

202	Bad Argument Type
203	Bad Argument Value
204	Undefined Name
205	LSTARG Disabled
206	Incomplete Subexpression
207	Implizit () off
208	Implizit () on
301	Positive Underflow
302	Negative Underflow
303	Overflow
304	Undefined Result
305	Infinite result
501	Invalid Dimension
502	Invalid Array Element
503	Deleting Row
504	Deleting Column
505	Inserting Row
506	Inserting Column
601	Invalid $\sum$ Data
602	Nonexistend $\sum$DAT
603	Insufficient $\sum$ Data
604	Invalid $\sum$PAR
605	Invalid $\sum$ Data LN(Neg)
606	Invalid $\sum$ Data LN(0)
607	Invalid EQ
608	Current equation:
609	No current equation
60A	Enter eqn, press NEW
60B	Name the equation, press ENTER
60C	Select plot type

60D	Emty catalog
60F	No Statistics data to plot
610	Autoscaling
614	Select a model
619	Acknowledged
61A	Enter Alarm, press SET
61B	Select repeat intervall
61C	I/O setup menu
61D	Plot type:
61E	""
61F	(OFF SCREEN)
620	Invalid PTYPE
621	Name the stat data, press ENTER
622	Enter value (zoom out if > 1), press ENTER
623	Copied to stack
624	x axis zoom w/AUTO.
625	x axis zoom.
626	y axis zoom
627	x and y-axis zoom
A01	Bad Guess(es)
A02	Constant?
A03	Interrupted
A04	Zero
A05	Sign reversal
A06	Extremum
B01	Invalid UNIT
B02	Inconsist Units
C01	Bad Packet Block Check
C02	Timeout

C03	Receive Error
C04	Receive Buffer Overrun
C05	Parity Error
C06	Tranfer Failed
C07	Protocol Error
C08	Invalid Server Cmd
C09	Port Closed
C0A	Connecting
C0B	Retry #
C0C	Awaiting Server Cmd.
C0E	Receiving
C0F	Objekt Discarded
C10	Packet #
C11	Processing Command
C12	Invalid IOPAR
C13	Invalid PRTPAR
C14	I/O: Batt Too Low
C15	Emty Stack
C17	Invalid Name
D01	Invalid Date
D02	Invalid Time
D03	Invalid Repeat
D04	Nonexistent Alarm

Meldung	Bedeutung	#hex
Acknowledged	Alarm bestätigt	619
Autoscaling	x-/y-Achsen-Skalierung	610
Awaiting Server Cmd.	Server-Modus aktiv	C0C
Bad Argument Type	Operation mit nicht erlaubten Typen	202
Bad Argument Value	Parameterwert außerhalb des Gültigkeitsbereiches	203
Bad Guess(es)	Schätzwert für SOLVE bzw ROOT liegt außerhalb des Wertebereichs	A01
Bad Packet Block Check	Die berechneten Prüfsummen eines Paketes sind ungleich	C01
Can't Edit Null Char.	Versuch einen String mit dem Zeichen char#0 zu editieren	102
Circular Reference	Versuch einen Variablennamen. in sich selbst zu speichern	129
Connecting	Prüfung der IR/seriellen Schnittstelle	C0A
Constant?	SOLVE oder ROOT haben an jedem Funktionspunkt den gleichen Wert	A02

Copied to stack	-> STK hat die ausgewählte Gleichung in den Stack kopiert	623
Current equation	Aktuelle Gleichung	608
Deleting Column	Der MatrixWriter löscht gerade eine Spalte	504
Deleting Row	Der MatrixWriter löscht gerade eine Zeile	503
Directory Not Allowed	Ein Verzeichnisname wird als Argument verwendet	12A
Directory Recursion	Versuch ein Verzeichnis in sich selbst zu speichern	002
Emty catalog	Keine Daten im aktuellen Katalog	60D
Enter alarm, press SET	Aufforderung zur Alarmeingabe	61A
Enter eqn, press NEW	Neue Gleichung in EQ speichern	60A
Enter value (zoomout if >1), press ENTER	Wert für Zoom-Funktion aufnehmen und zoomen	622
Extremum	Von SOLVE oder ROOT gefundener Extremwert	A06
HALT Not Allowed	Im Programm wurde HALT gefunden während der MatrixWriter, DRAW oder SOLVE aktiv war	126
I/O setup menu	Menü zur Einstellung der Setup-Parameter	61C
Implicit () off	Implizite Klammern aus	207
Implicit () on	Implizite Klammern ein	208

Incomplete Subexpression	▸,▾, oder ENTER gedrückt, bevor alle Parameter einer Funktion angegeben sind	206
Inconsistent Units	Versuch einer Umwandlung nicht passender Einheiten	B02
Infinite Result	Mathematischer Fehler: unendliches Ergebnis	305
Inserting Column	Der MatrixWriter fügt eine Spalte ein	504
Inserting Row	Der MatrixWriter fügt eine Zeile ein	503
Insufficient Memory	Nicht genügend freier Speicherplatz	001
Insufficient ∑Data	∑DAT enthielt nicht genügend Statistikpunkte zur Berechnung	603
Interrupted	SOLVE/ROOT wurden durch ATTN unterbrochen	A03
Invalid Array	ENTER hat Daten des Falschen Typs an ein Feld übergeben	502
Invalid Card Data	Der HP 48 erkennt Daten aus der Steckkarte nicht	008
Invalid Date	Ein Argument ist keine reelle Zahl im richtigen Format oder außerhalb des Wertebereichs	D01
Invalid Definition	Falscher Aufbau einer Gleichung für Define	12C
Invalid Dimension	Ein Feldargument hatte falsche Dimensionen	501
Invalid EQ	EQ enthält keinen algebraischen Ausdruck für die Funktionen des Menüs GRAPHICS FCN oder für DRAW im Modus CONIC	607
Invalid IOPAR	IOPART enthält falsche oder fehlende Elemente	C12
Invalid Name	Unzulässiger Dateiname	C17

Invalid PPAR	PPAR enthält falsche oder fehlende Elemente	12E
Invalid PRTPAR	PRTPAR enthält falsche oder fehlende Elemente	C13
Invalid PTYPE	Plottyp für die aktuelle Gleichung ist ungültig	620
Invalid Repeat	Wiederholungsintervall für Terminerinnerung liegt außerhalb des zulässigen Bereichs	D03
Invalid Server	Empfang eines ungültigen Befehls im Server-Modus	C08
Invalid Syntax	ENTER oder STR-> wegen Syntaxfehler unmöglich	106
Invalid Time	Der Zeitparameter hat ein falsches Format oder ist außerhalb des zulässigen Bereichs	D02
Invalid Unit	Operation mit einer ungültigen Benutzereinheit	B01
Invalid User Function	Art oder Aufbau des Objekts für eine benutzerdefinierte Funktion war falsch	103
Invalid DATA	∑DAT enthält ungültiges Objekt	601
Invalid DATA LN(Neg)	Versuch eine nichtlineare Kurve zu ermitteln, wobei DAT ein negatives Element enthielt	605
Invalid ∑ DATA	Versuch eine nichtlineare Kurve zu ermitteln, wobei DAT ein 0-Element enthielt	606
Invalid ∑PAR	PAR enthält falsche oder fehlende Parameter	604
LAST CMD disabled	LAST CMD wurde gedrückt, obwohl diese Funktion gesperrt war	125
LAST STACK	LAST STACK wurde trotz Sperre gedrückt	124
LASTARG disabled	LASTARG wurde trotz Sperre gedrückt	205

Low Battery	Die Batteriespannung ist zu gering, um fehlerfrei zu senden	C14
Memory Clear	Der Speicher wurde gelöscht	005
Name Conflict	Versuch, | (wobei) auszuführen um einer Integrationsvariablen oder einem Summenindex einen Wert zuzuweisen	13C
Name the equation, press ENTER	Der Gleichung einen Namen zuweisen und in EQ speichern	60B
Name the statdata,press ENTER	Den Statistikdaten einen Namen zuweisen und in _DAT speichern	621
Negative Underflow	Mathematischer Fehler: Berechnung lieferte negatives Ergebnis ≠ 0 >-MINR	302
No Current Equation	SOLVR, DRAW, oder RCEQ wurden ohne EQ benutzt	104
No current equation	Statusmeldung von PLOT/SOLVE	609
No Room in Port	Unzureichender RAM-Port-Speicherplatz	00B
No Room to Save Stack	Nicht genügend Speicher, um den Stack zu sichern. Last Stack wird gesperrt	101
No Room to Show	Stackobjekte werden mangels Speicherplatz nur durch ihren Typ dargestellt	131
No stat data to plot	Keine Daten in ∑DAT gespeichert	60F
Non-Empty Directory	Versuch ein nicht leeres Verzeichnis zu löschen	12B
Non-Real Result	HP_SOLVE, ROOT, DRAW oder ∫ lieferte ein nicht reelles Ergebnis als Zahl oder Einheit	12F

Nonexistent Alarm	Ein eingegebener Termin ist in der Terminliste nichtenthalten	D04
Nonexistent ∑DAT	Statistikbefehl ohne Existenz von ∑DAT	602
Object Discarded	Der Sender hat ein EOF-(Z)-Paket mit einem ''D'' im Datenfeld übertragen	C0F
Object in Use	Versuch, PURGE oder STO auf ein Sicherungsobjekt anzuwenden, als der zugehörige Datenblock bearbeitet wurde	009
Object not in Port	Versuch auf ein nichtvorhandenes Sicherungsobjekt oder eine Bibliothek zuzugreifen	00C
(OFF SCREEN)	Funktionswert, Nullstellen, Extremwerte oder Schnittpunkte nicht im Bereich der Anzeige	61F
Out of Memory	Unzureichender Speicherplatz. Eines oder mehrere Objekte müssen gelöscht werden um den Betrieb fortzusetzen	135
Overflow	Mathematischer Fehler: Ergebniswert (absolut) größer als MAXR	303
Packet #	Nummer eines empfangenen/gesendeten Daten-Pakets	C10
Parity Error	Das Paritätsbit der empfangenen Bytes entspricht nicht der Einstellung	C05
Port Closed	Mögliche Gerätestörung bei Schnittstelle. Selbsttest starten	C09
Port Not Available	Es wurde ein Portbefehl auf ein leeres Port ausgeführt	00A
Positive Underflow	Mathematischer Fehler: MINR > Ergebniswert >0	301

Power Lost	Der Taschenrechner wurde nach einem Spannungsausfall wieder eingeschaltet. Evtl Speicherinhalt verfälscht	006
Prozessing Command	Befehlspaket des Leitrechners wird verarbeitet	C11
Protocol Error	Empfang eines Packetes, dessen Länge kleiner als ein Nullpacket war	C07
Receive Buffer	Kermit: Es wurden mehr als 255 wiederholte Bytes gesendet, bevor ein neues Packet empfangen wurde SRECV: Die ankommenden Daten führten zum Überlauf	C04
Receive Error	UART-Datenverlust oder Synchronisationsfehler	C03
Receiving	Name des Objekts empfangen	C0E
Retry #	Zeigt die Anzahl der Versuche während der Wiederholung des Packetaustausches an	C0B
Select a model	Statistikmodell zur Kurvenermittlung wählen	614
Select plot type	Darstellungsart wählen	60C
Select repeat	Wiederholungsintervall für Terminerinnerung wählen	61B
Sending	Erkennt den Namen des Objekts während des Sendens	C0D
Sign Reversal	HP-SOLVE oder ROOT konnten keine Nullstelle, sonder nur zwei benachbarte Punkte, welche das Vorzeichen wechseln, finden	A05
Timeout	Drucken seriell: Empfang von XOFF, Unterbrechung, es wird auf XON gewartet	C02

To Few Arguments	Der Befehl erfordert mehr Argumente, als im Stack vorhanden	201
Transfer Failed	10 aufeinander folgende Versuche, ein Packet zu empfangen, sind fehlgeschlagen	C06
Unable to Isolate	ISOL konnte nicht ausgeführt werden, weil der Variablenname fehlt oder im Argument einer Funktion enthalten ist, die keine Umkehrung besitzt	130
Undefined Local Name	Hat einen lokalen Namen aufgerufen, für den keine entsprechende Variable existiert	003
Undefined Name	Hat einen globalen Namen aufgerufen, für den keine entsprechende variable existiert	204
Undefined Result	Mathemathischer Fehler: Undefiniertes Ergebnis (z.B: 0/0)	304
Undefined XLIB Name	Versuch, einen XLIB-Namen auszuführen, wobei die angegebene Bibliothek nicht existierte	004
Wrong Argument	anwenderdefinierte Funktion mit einer falschen Klammerzahl ausgeführt	128
x and Y-axis zoom.	x- und y-Achse zoomen	627
x axis zoom	x-Achse zoomen	625
x axis zoom w/AUTO.	x-Achse automatisch zoomen	624
y axis zoom.	y-Achse zoomen	626
ZERO	Ergebnis von HP-SOLVE oder ROOT ist eine Nullstelle	A04
`\'/'	Keine Ausführung bei gedrücktem EXECS	61E

```
Ask                                (get console input to variable)
Bye                                (logout remote server)
C or Connect                       (become a terminal)
Clear                              (clear serial port buffer)
Close                              (logging file)
Comment                            (text is ignored)
CWD or CD                          (change dir &/or disk)
Define/Assign                      (a command macro)
Delete                             (a file)
Directory
Disable                            (selected server commands)
Do                                 (a macro)
Echo text                          (show line on screen)
Enable                             (selected server commands)
EXIT                               (leave Kermit)
Finish                             (to remote server)
Get                                (remote file opt new name)
Goto                               (label, Take file or Macro)
Hangup                             (drop DTR, hang up phone)
HELP                               for an Introduction, use
''?''                              within commands for specific help.
I or Input [timeout] text          (scripts)
If [not] <condition> <command>
Log                                (Packet, Session, Transaction)
Logout                             (remote server)
Mail                               (file to host Mailer)
Output text                        (for scripts)
Pause [seconds]                    (for scripts)
Pop                                (exit current Take file or macro)
Push                               (go to DOS, keep Kermit)
Quit                               (leave Kermit)
R or Receive                       (opt local filename)
Reinput                            (script Input, reread buffer)
Remote                             (prefix for commands)
Run                                (a program)
S or Send                          (local file    new name)
Server [timeout]                   (become a server)
Set                                (most things)
Show                               (most things)
Space                              (free on current disk)
Status                             (show main conditions)
Stay                               (in Kermit after startup)
Stop                               (exit all Take files & macros)
Take                               (do a command file)
Transmit filespec [prompt          (raw upload)
Type                               (a file)
Version                            (show Kermit''s id
Wait [timeout] on modem \cd \cts \dsr
```

A	Ampere, elektrischer Strom (1 A)
a	Ar, Fläche (100m^2)
Acre	Acre, Fläche (4046,87260987m^2)
arcmin	Bogenminute, Winkel in der Ebene (4,62962962963*10^-5)
arcs	Bogensekunde, Winkel in der Ebene (7,71604938272*10^-7)
atm	Athmosphäre, Druck (101325kg/m*s^2)
AU	Astronomische Einheit, Länge (1,495979*10^11m)
bar	Bar, Druck (1*10^5kg/m*s^2)
b	Barn, Fläche (1*10^-28m^2)
bbl	Barrel, Volumen (0,158987294928m^3)
Bq	Becquerel, Aktivität (1 1/s)
Btu	Internationales Btu, Energie (1055,05585262kgm^2/s^2)
bu	Bushel, Volumen (0,03523907m^3)
cal	Kalorie, Energie (4,186kgm^2/s^2)
C	Coulomb, elektrische Ladung (1As)
cd	Candela, Lichtstärke (1cd)
chai	Chain, Länge (20,1168402337m)
Ci	Curie, Aktivität (3,7*10^10 1/s)
c	Lichtgeschwindigkeit, Geschwindigkeit (299792458m/s)
cm/s	Zentimeter pro Sekunde, Geschwindigkeit (0,01m/s)
cm	Zentimeter, Länge (0,01m)
cm^2	Quadratzentimeter, Fläche (1*10^-4 m^2)

cm^3	Kubikzentimeter, Volumen (1*10^-6m^3)
ct	Karat, Masse (0,002kg)
cu	US cup, Volumen (2,365882365*10-4m^3)
d	Tag,, Zeit (86400s)
dyn	dyn, Kraft (0,00001kgm/s^2)
erg	erg, Energie (0,0000001kgm^2/s^2)
eV	Elektronenvolt, Energie (1,60219*10^-19kgm^2/s^2)
fath	Fathom, Länge (1,82880365761m)
fbm	Board foot, Volumen (0,002359737216m^3)
fc	Footcandle, Beleuchtungsstärke (0,856564774909cd/m^2)
Fdy	Farad, elektrische Ladung (96487As)
fermi	Fermi, Länge (1*10^-15m)
F	Farad, Kapazität (1A^2s^4/kgm^2)
flam	Foot lambert, Leuchtdichte (3,42625909964cd/m^2)
ft*lbf	Foot-poundf, Energie (1,35581794833kgm^2/s^2)
ft/s	Feet/second, Geschwindigkeit (0,3048m/s)
ft	International Foot, Länge (0,3048m)
ftUS	Survey foot, Länge (0,304800609601m)
ft^2	square foot, Fläche (0,09290304m^2)
ft^3	cubic foot, Volumen (0,028316846592m^3)
galC	Canadian gallon, Volumen (0,004546092m^3)
galUK	UK gallon, Volumen (o,004546092m^3)
ga	Normalfallbeschleunigung, Geschwindigkeit (9,80665m/s^2)

gf	Gramm, Kraft (0,00980665kgm/s^2)
g	Gramm, Masse (0,001kg)
grad	Grad, Winkel in der Ebene (0,0025)
grain	Grain, Masse (0,00006479891kg)
Gy	Grey, absorbierte Dosis (1m^2/s^2)
ha	Hektar, Fläche (10000m^2)
H	Henry, Induktivität (1kgm^2/A^2s^2)
hp	Horse Power, Leistung (745,699871582kgm^2/s^3)
h	Stunde, Zeit (3600s)
Hz	Hertz, Frequnz (1/s)
inH2O	Inches of water, Druck (24884kg/m*s^2)
inHg	Inch Quecksilbersäule, Druck (3386,38815789kg/m*s^2)
in	Inch, Länge (0,0254m)
in^2	square inch, Fläche (0,00064516m^2)
in^3	cubic inch, Volumen (0,000016387064m^3)
J	Joule, Energie (1kgm^2/s^2)
kcal	Kilokalorie, Energie (4186kgm^2/s^2)
kg	Kilogramm, Masse (1kg)
kip	Kilopund, Kraft (4448,22161526kgm/s^2)
K	Kelvin, Temperatur (1K)
km	Kilometer, Länge (1km)
km^2	Quadratkilometer, Fläche (1km^2)

knot	Nautische Meilen pro Stunde, Geschwindigkeit (0,514444444444m/s)
kph	Kilometer pro Stunde, Geschwindigkeit (0,277777777778m/s)
lam	Lmabert, Leuchtdichte (3183,09886184cd/m^2)
lb	Avoirdupois Pound, Masse (0,45359237kg)
lbf	Gewichtskraft, Kraft (4,44822161526kgm/s^2)
lbt	Troy Pound, Masse (0,3732417kg)
L	Liter, Volumen (0,001m^3)
lm	Lumen, Lichtstrom (7,95774715459*10^-2cd)
lx	Lux, Beleuchtungstärke (7,95774715459*10^-2cd/m^2)
lyr	Light Year, Länge (9,46052840488*10^15m)
m/s	Meter pro Sekunde, Geschwindigkeit (1m/s)
MeV	Megaelektronenvolt, Energie (1,60219*10^-13kgm^2/s^2)
mho	Siemens, elektrischer Leitwert (1A^2s^3/kgm^2)
mi	Internationale Meile, Länge (1609,344m)
mil	Tausendstel Zoll, Länge (0,0000254m)
min	Minute, Zeit (60s) , Geschwindigkeit (m/s)(
miUS	US-Meile, Länge (1609,34721869m)
miUS^2	US-Quadratmeile, Fläche (258998,47032m^2),
mi^2	Internationale Quadratmeile, Fläche (2589988m^2)
ml	Milliliter, Volumen (1*10^-6m^3)
m	Meter, Länge (1m)
mmHg	Millimeter Quecksilbersäule, Druck (133,322368421kg/m*s^2)

mm	Millimeter, Länge (0,0001m)
mol	Mol, Masse (1mol)
mp/h	Meilen pro Stunde, Geschwindigkeit (0,44704m/s)
Mpc	Megaparsec, Länge (3,08567818585*10^22m)
m^2	Quadratmeter, Fläche (1m^2)
m^3	Kubikmeter, Volumen (1m^3)
nmi	Nautische Meile, Länge (1852m)
N	Newton, Kraft (1kgm/s^2)
ozFL	Us Flüssigkeitsunze, Volumen (2,95735295625*10-5m^3)
ozt	Unze (tr), Masse (0,031103475kg)
ozUK	UK Flüssigkeitsunze, Volumen (2,8413075*10^-5m^3)
oz	Unze, Masse (0,0283495231kg)
Pa	Pascal, Druck (kg/m*s^2)
pc	Parsec, Länge (3,08567818585*10^16m)
pdl	Poundal, Kraft (0,138254954376kgm/s^2)
ph	Phot, Beleuchtungsstärke (795,774715459cd/m^2)
pk	Peck, Volumen (0,0088097675m^3)
P	Poise, dynamische Viskosität (0,1kg/ms)
psi	Pfund pro Quadratinch, Druck (6894,75729317kg/m*s^2)
pt	pint, Volumen (0,000473176473m^3)
qt	quart, Volumen (0,0009463529946m^3)
rad	Rad, Energiedosis (0,01m^2/s^2)

rd	rod, Länge (5,0292100584m)
rem	rem, Dosisäquivalent (0,01m^2/s^2)
r	Radiant Winkel in der Ebene (0,1591549343092)
R	Roentgen, Strahlenbelastung (0,000258 As/kg)
sb	stilb, Leuchtdichte (10000cd/m^2)
slug	slug, Masse (14,5939029372kg)
sr	Steradiant, Raumwinkel (7,95774715459*10^-2)
s	Sekunde, Zeit (1s)
S	Siemens, elektrischer Leitwert (1A^2s^3/kgm^3)
st	Stere, Volumen (1m^3)
St	stokes, chinematische Viskosität (0,0001m^2/s)
Sv	Sievert, Dosisäquivalent (0,01m^2/s^2)
tbsp	tablespoon, Volumen (1,47867647813*10^-5m^3)
therm	Wärmeinheit EEC, Energie (105506000kgm^2/s^2)
t	Metrische Tonne, Masse (1000kg)
ton	short ton, Masse (907,18474kg)
tonUK	long ton, Masse (1016,0469088kg)
torr	Torr (mmHg), Druck (133,322368421kg/m*s^2)
tsp	teaspoon, Volumen (4,92892159375*10^-6m^3)
T	Tesla, magnetischer Fluß (1kg/As^2)
u	Atommasseneinheit, Masse (1,66057*10^-27kg)
V	Volt, elektrische Potentialdifferenz (1kgm^2/As^3)

Wb	Weber, magnetischer Fluß (1kgm^2/As^2)
W	Watt, elektrische Leistung (1kgm^2/s^3)
W	Watt, Leistung (1kgm^2/s^3)
yd	Yard, Länge (0,9144m)
yd^2	square yard, Fläche (0,83612736m^2)
yd^3	cubic yard, Volumen (0,764554857984m^3)
yr	Jahr, Zeit (31556925,9747s)
Å	Angstrom, Länge (1*10^-10m)
μ	Micron, Länge (1*10^-6m)
Ω	Ohm, elektrischer Widerstand (1kgm^2/a^2s^3)
°C	Grad Celsius, Temperatur
°F	Grad Fahrenheit, Temperatur
°	Grad, Winkel in der Ebene (2,77777777778*10^-3)
°R	Grad Rankine, Temperatur

UNITS	LENGTH
m	Meter, Länge (1m)
cm	Zentimeter, Länge (0,01m)
mm	Millimeter, Länge (0,0001m)
yd	Yard, Länge (0,9144m)
ft	International Foot, Länge (0,3048m)
in	Inch, Länge (0,0254m)
Mpc	Megaparsec, Länge (3,08567818585*10^22m)
pc	Parsec, Länge (3,08567818585*10^16m)
lyr	Light Year, Länge (9,46052840488*10^15m)
AU	Astronomische Einheit, Länge (1,495979*10^11m)
km	Kilometer, Länge (1km)
mi	Internationale Meile, Länge (1609,344m)
nmi	Nautische Meile, Länge (1852m)
miUS	US-Meile, Länge (1609,34721869m)
chai	Chain, Länge (20,1168402337m)
rd	rod, Länge (5,0292100584m)
fath	Fathom, Länge (1,82880365761m)
ftUS	Survey foot, Länge (0,304800609601m)
mil	Tausendstel Zoll, Länge (0,0000254m)
μ	Micron, Länge (1*10^-6m)
Å	Angstrom, Länge (1*10^-10m
fermi	Fermi, Länge (1*10^-15m)

Units Area

m^2	Quadratmeter, Fläche (1m^2)
cm^2	Quadratzentimeter, Fläche (1*10^-4 m^2)
b	Barn, Fläche (1*10^-28m^2)
yd^2	square yard, Fläche (0,83612736m^2)
ft^2	square foot, Fläche (0,09290304m^2)
in^2	square inch, Fläche (0,00064516m^2)
km^2	Quadratkilometer, Fläche (1km^2)
ha	Hektar, Fläche (10000m^2)
a	Ar, Fläche (100m^2)
mi^2	Internationale Quadratmeile, Fläche (2589988m^2)
miUS^2	US-Quadratmeile, Fläche (258998,47032m^2),
Acre	Acre, Fläche (4046,87260987m^2)

UNITS VOL

m^3	Kubikmeter, Volumen (1m^3)
st	Stere, Volumen (1m^3)
cm^3	Kubikzentimeter, Volumen (1*10^-6m^3)
yd^3	cubic yard, Volumen (0,764554857984m^3)
ft^3	cubic foot, Volumen (0,028316846592m^3)
in^3	cubic inch, Volumen (0,000016387064m^3)
L	Liter, Volumen (0,001m^3)
galUK	UK gallon, Volumen (o,004546092m^3)

galC	Canadian gallon, Volumen (0,004546092m^3)
qt	quart, Volumen (0,0009463529946m^3)
pt	pint, Volumen (0,000473176473m^3)
ml	Milliliter, Volumen (1*10^-6m^3)
cu	US cup, Volumen (2,365882365*10-4m^3)
ozFL	Us Flüssigkeitsunze, Volumen (2,95735295625*10-5m^3)
ozUK	UK Flüssigkeitsunze, Volumen (2,8413075*10^-5m^3)
tbsp	tablespoon, Volumen (1,47867647813*10^-5m^3)
tsp	teaspoon, Volumen (4,92892159375*10^-6m^3)
bbl	Barrel, Volumen (0,158987294928m^3)
bu	Bushel, Volumen (0,03523907m^3)
pk	Peck, Volumen (0,0088097675m^3)
fbm	Board foot, Volumen (0,002359737216m^3)

TIME

yr	Jahr, Zeit (31556925,9747s)
d	Tag,, Zeit (86400s)
h	Stunde, Zeit (3600s)
min	Minute, Zeit (60s)
s	Sekunde, Zeit (1s)
Hz	Hertz, Frequnz (1/s)

UNITS SPEED

m/s	Meter pro Sekunde, Geschwindigkeit (1m/s)

cm/s	Zentimeter pro Sekunde, Geschwindigkeit (0,01m/s)
ft/s	Feet/second, Geschwindigkeit (0,3048m/s)
kph	Kilometer pro Stunde, Geschwindigkeit (0,277777777778m/s)
mp/h	Meilen pro Stunde, Geschwindigkeit (0,44704m/s)
knot	Nautische Meilen pro Stunde, Geschwindigkeit (0,514444444444m/s)
c	Lichtgeschwindigkeit, Geschwindigkeit (299792458m/s)
ga	Normalfallbeschleunigung, Geschwindigkeit (9,80665m/s^2)

UNITS MASS

kg	Kilogramm, Masse (1kg)
g	Gramm, Masse (0,001kg)
lb	Avoirdupois Pound, Masse (0,45359237kg)
oz	Unze, Masse (0,0283495231kg)
slug	slug, Masse (14,5939029372kg)
lbt	Troy Pound, Masse (0,3732417kg)
ton	short ton, Masse (907,18474kg)
tonUK	long ton, Masse (1016,0469088kg)
t	Metrische Tonne, Masse (1000kg)
ozt	Unze (tr), Masse (0,031103475kg)
ct	Karat, Masse (0,002kg)
grain	Grain, Masse (0,00006479891kg)
u	Atommasseneinheit, Masse (1,66057*10^-27kg)

mol	Mol, Masse (1mol)

UNITS FORCE

N	Newton, Kraft (1kgm/s^2)
dyn	dyn, Kraft (0,00001kgm/s^2)
gf	Gramm, Kraft (0,00980665kgm/s^2)
kip	Kilopund, Kraft (4448,22161526kgm/s^2)
lbf	Gewichtskraft, Kraft (4,44822161526kgm/s^2)
pdl	Poundal, Kraft (0,138254954376kgm/s^2)

UNITS ENRG

J	Joule, Energie (1kgm^2/s^2)
erg	erg, Energie (0,0000001kgm^2/s^2)
kcal	Kilokalorie, Energie (4186kgm^2/s^2)
cal	Kalorie, Energie (4,186kgm^2/s^2)
Btu	Internationales Btu, Energie (1055,05585262kgm^2/s^2)
ft*lbf	Foot-poundf, Energie (1,35581794833kgm^2/s^2)
therm	Wärmeinheit EEC, Energie (105506000kgm^2/s^2)
MeV	Megaelektronenvolt, Energie (1,60219*10^-13kgm^2/s^2)
eV	Elektronenvolt, Energie (1,60219*10^-19kgm^2/s^2)

UNITS POWER

W	Watt, Leistung (1kgm^2/s^3)
hp	Horse Power, Leistung (745,699871582kgm^2/s^3)

UNITS PRESS

Pa	Pascal, Druck (kg/m*s^2)
atm	Athmosphäre, Druck (101325kg/m*s^2)
bar	Bar, Druck (1*10^5kg/m*s^2)
psi	Pfund pro Quadratinch, Druck (6894,75729317kg/m*s^2)
torr	Torr (mmHg), Druck (133,322368421kg/m*s^2)
mmHg	Millimeter Quecksilbersäule, Druck (133,322368421kg/m*s^2)
inHg	Inch Quecksilbersäule, Druck (3386,38815789kg/m*s^2)
inH2O	Inches of water, Druck (24884kg/m*s^2)

UNITS TEMP

°C	Grad Celsius, Temperatur
°F	Grad Fahrenheit, Temperatur
K	Kelvin, Temperatur (1K)
°R	Grad Rankine, Temperatur

UNITS ELEC

V	Volt, elektrische Potentialdifferenz (1kgm^2/As^3)
A	Ampere, elektrischer Strom (1 A)
C	Coulomb, elektrische Ladung (1As)
Ω	Ohm, elektrischer Widerstand (1kgm^2/a^2s^3)
F	Farad, Kapazität (1A^2s^4/kgm^2)
W	Watt, elektrische Leistung (1kgm^2/s^3)
Fdy	Farad, elektrische Ladung (96487As)

H	Henry, Induktivität (1kgm^2/A^2s^2)
mho	Siemens, elektrischer Leitwert (1A^2s^3/kgm^2)
S	Siemens, elektrischer Leitwert (1A^2s^3/kgm^3)
T	Tesla, magnetischer Fluß (1kg/As^2)
Wb	Weber, magnetischer Fluß (1kgm^2/As^2)

UNITS ANGL

°	Grad, Winkel in der Ebene (2,77777777778*10^-3)
r	Radiant Winkel in der Ebene (0,1591549343092)
grad	Grad, Winkel in der Ebene (0,0025)
arcmin	Bogenminute, Winkel in der Ebene (4,62962962963*10^-5)
arcs	Bogensekunde, Winkel in der Ebene (7,71604938272*10^-7)
sr	Steradiant, Raumwinkel (7,95774715459*10^-2)

UNITS LIGHT

fc	Footcandle, Beleuchtungsstärke (0,856564774909cd/m^2)
flam	Foot lambert, Leuchtdichte (3,42625909964cd/m^2)
lx	Lux, Beleuchtungstärke (7,95774715459*10^-2cd/m^2)
ph	Phot, Beleuchtungsstärke (795,774715459cd/m^2)
sb	stilb, Leuchtdichte (10000cd/m^2)
lm	Lumen, Lichtstrom (7,95774715459*10^-2cd)
cd	Candela, Lichtstärke (1cd)
lam	Lmabert, Leuchtdichte (3183,09886184cd/m^2)

UNIT RAD

Gy	Grey, absorbierte Dosis (1m^2/s^2)
rad	Rad, Energiedosis (0,01m^2/s^2)
rem	rem, Dosisäquivalent (0,01m^2/s^2)
Sv	Sievert, Dosisäquivalent (0,01m^2/s^2)
Bq	Becquerel, Aktivität (1 1/s)
Ci	Curie, Aktivität (3,7*10^10 1/s)
R	Roentgen, Strahlenbelastung (0,000258 As/kg)

UNITS VISK

P	Poise, dynamische Viskosität (0,1kg/ms)
St	stokes, chinematische Viskosität (0,0001m^2/s)

LITERATURVERZEICHNIS

Lehr- und Übungsbuch Mathematik 1-3, Harry Deutsch Verlag, Leipzig 1983.

w.P. Minorski: Aufgabensammlung der höheren Mathematik, Vieweg, Braunschweig 1986.

Kusch: Mathematik 1-4, Girardet-Verlag, Essen 1978

Bronstein-Semendjajew: Taschenbuch der Mathematik, Nauka Verlag, Moskau 1987

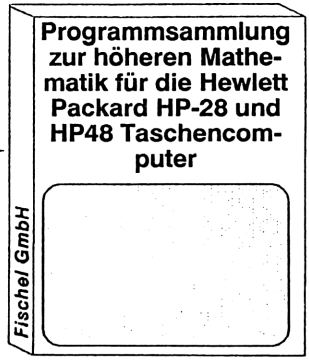
Bartsch: Taschenbuch mathematischer Formeln, Harry Deutsch Verlag, Leipzig 1986

Stoer-Bulirsch: Einführung in die numerische Mathematik 1 und 2, Berlin 1980.

D. Ciesinger/ S. Ebeling: Programmsammlung zur höheren Mathematik für die HP-28 und HP-48 Taschencomputer, ISBN 3-89374-078-3, Preis: 49,- DM

Inhaltsverzeichnis

Sofort lieferbar! Ciesinger/Ebeling



ISBN 3-89374-078-3
Preis: 49,-- DM
(inkl. 7% MwSt.)

Aufruf!
An alle Autoren!

ISBN 3-89374-076-7
Preis: 49,-- DM
(inkl. 7% MwSt.)

Für dieses neue Buchprojekt suchen wir noch Beiträge. Wir würden uns daher freuen, wenn Sie uns Ihre Programme für den HP-48SX, ihre Ratschläge zur besseren Ausnutzung des Gerätes, Tricks zur Handhabung usw. zuschicken würden. Ihre Einsendung wird selbstverständlich gratifiziert. Eine Veröffentlichung kann allerdings erst stattfinden, wenn Sie die Überlassungserklärung am Ende dieses Heftes unterschrieben an uns einschicken:
Fischel GmbH,
Kaiser-Friedrich-Str. 54a,
1000 Berlin 12

POCKET + LAPTOP COMPUTER

Zeitschrift für mobile Datensysteme

Einmaliges Sonderangebot!

Nur für kurze Zeit:
Abonnieren Sie jetzt! Es lohnt sich!

Für nur 72,- DM für 12 Hefte (Ausland 84,- DM) erhalten Sie regelmäßig alle Informationen aus dem Markt für tragbare Computersysteme. Und für alle Leser dieses Heftes haben wir noch eine besondere Überraschung auf Lager: **Sichern Sie sich jetzt Ihr Pocket + Laptop Abonnement! Wenn Sie zusätzlich ein Buch (siehe Bestellschein) bestellen, erhalten Sie aus unserem Sortiment ein Buch Ihrer Wahl gratis dazu!**

Dieses einmalige Angebot gilt nur, wenn Sie mit diesem Formular abonnieren! Schicken Sie es, zusammen mit dem ausgefüllten Bücher - Bestellschein an:

Fischel GmbH, Kaiser-Friedrich-Str. 54a, 1000 Berlin 12, Fax: 030/ 324 09 28.

Selbstverständlich gilt dieses Angebot auch für Leser, die Ihr Abonnement verlängern wollen, aber **nur,** wenn die Verlängerung **mit diesem Bestellschein** erfolgt!

Neuer Service! Ab jetzt gibt es alle Bücher auch per BTX: wählen Sie: * IKS #

1 + 1 = 3 !!

Sie abonnieren + bestellen ein Buch = Sie erhalten ein Buch Ihrer Wahl gratis dazu!

Ja, ich mache von Ihrem Angebot Gebrauch und abonniere die Zeitschrift "Pocket + Laptop Computer"

ab Heft Nr: _____

Name: _____

Anschrift: _____

Plz, Ort: _____

Ich habe folgenden Rechnertyp: _____

Den ausgefüllten Bücher - Bestellschein habe ich beigelegt.* Als Gratis-Buch schicken Sie mir bitte:

_____

☐ Ich habe einen Scheck über DM _____ beigelegt.

☐ Ich habe den Betrag von DM _____ auf Iher Konto 461533-103, Postgiroamt Berlin (West), BLZ 10010010 überwiesen.

Bearbeitung erfolgt nur, wenn die Zahlungsart angegeben ist!

* RÜCKSEITE

Programmierhandbuch

zum Hewlett-Packard

HP-48 SX

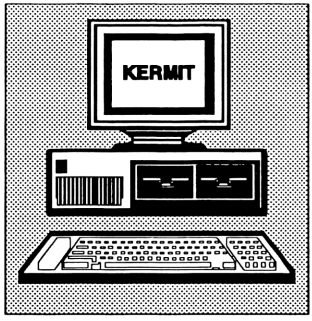
Pocketcomputer

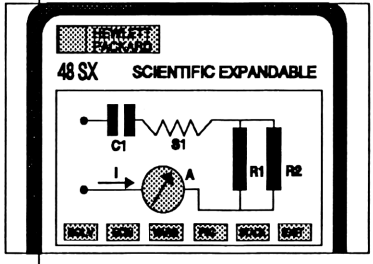
DATENÜBERTRAGUNG

IN 28

```
<< HOME CLLCD
" <IN 28> PROGRAMM ZUR
DATENÜBERTRAGUNG VOM
HP 28 S/C ZUM HP 48 SX
VERSION 1.1 VON
ROBERT HüBNER 1990"
CLLCD
"Datenübertragung" 2
DISP "beginnt" 3 DISP
400 .2 BEEP 1 WAIT INPRT
OBJ-> PATH DUP SIZE 1 1 1
-> l p pa si zp zl f

<< -56 FC?C 'f' STO p
SIZE 'zp' STO 1 'z' STO
2 zp
FOR j p j GET ->STR
EPOBJ
IF { 7 14 15 16 17 18 19 }
obp STR->
POS 0 ==
THEN
IF VARS obp STR->
POS 0 ==
THEN
```





Robert Hübner

ISBN 3-89374-065-1

Fischel GmbH