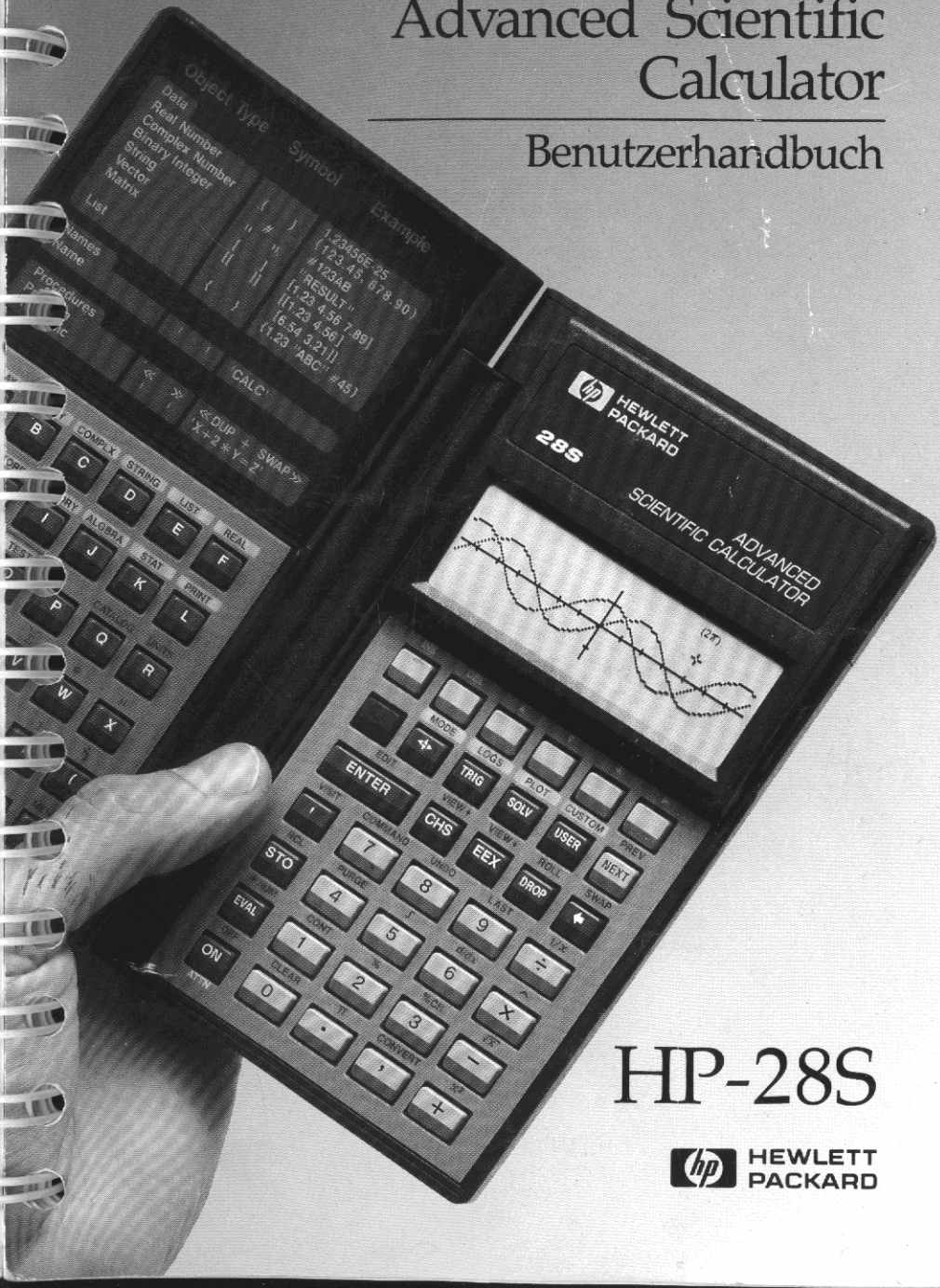
HEWLETT-PACKARD

Advanced Scientific Calculator

Benutzerhandbuch



HP-28S

HEWLETT PACKARD

HP-28S
Advanced Scientific Calculator

Benutzerhandbuch

HEWLETT PACKARD

2. Ausgabe September 1988
Bestellnummer 00028-90072

Hinweis

Änderungen der in dieser Dokumentation enthaltenen Informationen sind vorbehalten. Allgemeine Informationen über den Rechner und zur Gewährleistung finden Sie auf den Seiten 291 und 295.

Hewlett-Packard übernimmt weder ausdrücklich noch stillschweigend irgendwelche Haftung für die in diesem Handbuch dargestellten Programme und Beispiele—weder für deren Funktionsfähigkeit noch deren Eignung für irgendeine spezielle Anwendung. Hewlett-Packard haftet nicht für direkte oder indirekte Schäden im Zusammenhang mit oder als Folge der Lieferung, Benutzung oder Leistung der Programme. (Dies gilt nicht, soweit gesetzlich zwingend gehaftet wird.)

Hewlett-Packard übernimmt keine Verantwortung für den Gebrauch oder die Zuverlässigkeit von HP Software unter Verwendung von Geräten, welche nicht von Hewlett-Packard geliefert wurden.

Portable Computer Division
1000 N.E. Circle Blvd.
Corvallis, OR 97330, U.S.A.

Druckgeschichte

1. Ausgabe	Januar 1988	Fertigungsnr. 00028-90073
2. Ausgabe	September 1988	Fertigungsnr. 00028-90140

Vorwort

Mit Ihrem HP-28S können Sie auf einfache Weise die komplexesten Probleme lösen, einschließlich solcher Aufgabenstellungen, die Sie seither nicht mit einem Taschenrechner bearbeiten konnten. Der HP-28S kombiniert leistungsstarke numerische Rechenweise mit einer neuen Dimension—*symbolische Rechenweise*. Sie formulieren zuerst ein Problem symbolisch, lassen sich danach die symbolische Lösung anzeigen, aus welcher das globale Verhalten des Problems ersichtlich ist, und erhalten numerische Ergebnisse über die symbolische Lösung.

Der HP-28S enthält folgende Leistungsmerkmale:

- Algebraische Manipulation. Sie können in einem Ausdruck Terme erweitern, zusammenfassen oder neuordnen sowie eine Gleichung symbolisch nach einer Variablen auflösen.
- Höhere Mathematik (Calculus). Sie können Differentiale, bestimmte und unbestimmte Integrale berechnen.
- Numerische Lösungen. Unter Verwendung einer speziellen Lösungsroutine können Sie eine Gleichung oder einen Ausdruck nach jeder beliebigen Variablen auflösen. Ebenso ist die Lösung eines linearen Gleichungssystems möglich. Durch verschiedene Datentypen lassen sich komplexe Zahlen, Vektoren und Matrizen so einfach wie reelle Zahlen verwenden.
- Abbildungen. Ausdrücke, Gleichungen und statistische Daten lassen sich grafisch darstellen.
- Konvertierung von Einheiten. Sie können zwischen jeder äquivalenten Kombination der eingebauten 120 Einheiten eine Umrechnung vornehmen sowie Ihre eigenen Einheiten definieren.
- Statistik. Sie können statistische Operationen mit einer oder zwei Variablen sowie Wahrscheinlichkeitsberechnungen ausführen.
- Zahlensysteme. Es lassen sich Berechnungen im dualen, oktalen, dezimalen und hexadezimalen Zahlensystem durchführen; weiterhin sind Bitmanipulationen möglich.
- Direkte Eingabe für algebraische Ausdrücke und UPN-Logik für interaktive Berechnungen.

Das *HP-28S Benutzerhandbuch* ist in drei Teile aufgegliedert. In Teil 1, "Grundlagen", lernen Sie anhand einfacher Anwendungsbeispiele die Arbeitsweise des Rechners kennen. Teil 2, "Zusammenfassung der Rechnerfähigkeiten", baut auf Teil 1 auf und zeigt, wie Sie den Rechner zur Lösung Ihrer Problemstellungen einsetzen können. In Teil 3, "Programmierung", sind die Programmierungsfähigkeiten sowie deren Anwendung in einer Reihe von Beispielen beschrieben.

Das *HP-28C/S Referenzhandbuch* gibt Ihnen spezifische Informationen über Befehle und die Arbeitsweise des Rechners. Die ersten zwei Kapitel erläutern die Grundlagen sowie die gebräuchlichsten Operationen. Das dritte Kapitel stellt ein Verzeichnis der Menüs dar und beschreibt das Anwendungskonzept und die Befehle der einzelnen Menüs.

Es wird empfohlen, daß Sie sich zuerst durch die Beispiele im Teil 1 des Benutzerhandbuchs arbeiten, um mit dem Rechner und seinen Funktionen vertraut zu werden. Danach sollten Sie Teil 2 durcharbeiten, um ein breiteres Verständnis über die Rechneroperationen zu gewinnen. Wenn Sie näheres über einen bestimmten Befehl wissen möchten, können Sie dazu das Referenzhandbuch zu Hilfe nehmen. Den Einstieg in die Programmierung des Rechners finden Sie in Teil 3 des Benutzerhandbuchs.

Durch die erwähnten Handbücher erfahren Sie, wie Ihr HP-28S zur Lösung mathematischer Aufgabenstellungen eingesetzt werden kann; sie lehren jedoch nicht Mathematik. Es wird angenommen, daß Sie bereits mit den relevanten mathematischen Prinzipien vertraut sind. Um z.B. die Fähigkeiten des HP-28S im Bereich der höheren Mathematik effizient nutzen zu können, sollten Sie die elementaren Gesetze über Differential- und Integralrechnung kennen.

Auf der anderen Seite ist damit nicht gemeint, daß Sie über sämtliche Mathematikgebiete im HP-28S Bescheid wissen müssen, um die für Sie interessanten Teile anzuwenden. Wenn Sie z.B die Statistik-Operationen des HP-28S anwenden möchten, müssen Sie nicht die Integralrechnung verstanden haben.

Inhaltsverzeichnis

Teil 1: Grundlagen

Teil 3: Programmierung

Anhänge & Index

Verwenden dieses Handbuchs

Wenn Sie Zeit und Neigung haben, so lesen Sie dieses Handbuch von der ersten bis zur letzten Seite durch. Ansonsten wird folgende Vorgehensweise empfohlen, um die Anwendungsweise des Rechners kennenzulernen.

1. Lesen Sie zuerst die Kapitel 1 bis 5 im Teil 1, "Grundlagen", durch, um mit dem Rechner vertraut zu werden.
2. Wenn Sie an einem außerhalb von Teil 1 behandelten Thema interessiert sind, so können Sie die Beispiele in diesem Kapitel auch vorab versuchen.

Gliederung des Handbuchs

Teil 1, "Grundlagen", zeigt auf, wie einfache Problemstellungen bearbeitet werden können. Dabei lernen Sie auch die grundlegenden Bedienungsschritte für den Rechner kennen.

Teil 2, "Zusammenfassung der Rechnerfähigkeiten", baut auf Teil 1 auf. Sie erfahren hier Details über die Funktionsweise des Rechners, einschließlich Optionen und Fähigkeiten, welche in Teil 1 nicht behandelt wurden.

Teil 3, "Programmierung", beschreibt, wie Sie den Rechner programmieren können. Das letzte Kapitel, "Programmbeispiele", enthält eine Reihe von kurzen Programmen, die verschiedene Programmierungstechniken aufzeigen sollen.

Weitere Informationen

Beim Ausführen der in diesem Handbuch enthaltenen Beispiele können Fragen zu den demonstrierten bzw. erwähnten Fähigkeiten des Rechners auftreten. Beide Handbücher, dieses und das Referenzhandbuch, enthalten zusätzliche Informationen.

- Wenn Sie Probleme mit der Arbeitsweise des Rechners haben, können Sie unter "Antworten auf allgemeine Fragen" auf Seite 282 nachsehen.
- Eine kurze Beschreibung über die Funktionsweise jeder Taste finden Sie unter "Tastenverzeichnis" auf Seite 317.
- Eine kurze Beschreibung der Befehle innerhalb der einzelnen Menüs finden Sie in Anhang B, "Menütabellen", auf Seite 296.
- Detaillierte Informationen über ein Menü sind über das Referenzhandbuch erhältlich. Alle Menüs (sowie einige andere Themen) erscheinen in alphabetischer Reihenfolge. Der Inhalt des Schlüsselwortverzeichnisses ist auf der Rückseite des Referenzhandbuchs aufgelistet.
- Detaillierte Informationen über einen bestimmten Befehl finden Sie im "Verzeichnis der Operationen" am Ende des Referenzhandbuchs. Es gibt dort eine Referenz zu einem Eintrag im Schlüsselwortverzeichnis (gewöhnlich ein Menü) und eine Seitenangabe für den speziellen Befehl.

Teil 1

Grundlagen

1

Tastenfeld und Anzeige

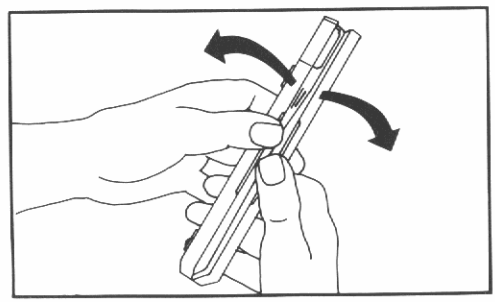
Dieses Kapitel beschreibt die Fähigkeiten des Rechners und demonstriert über ein einfaches Beispiel die prinzipielle Arbeitsweise und das Konzept des Rechners. Anschließend werden anhand einer Abbildung die wichtigsten Fähigkeiten des Tastenfelds und der Anzeige hervorgehoben. Am Ende lernen Sie den Befehlskatalog kennen, welcher eine hilfreiche Unterstützung bei der Anwendung von Befehlen bietet.

Bedienungsgrundlagen

Dieser Abschnitt enthält die Bedienungsgrundlagen für das Arbeiten mit dem Rechner.

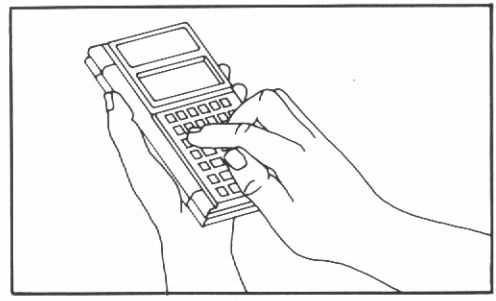
Öffnen und Schließen des Gehäuses

Das Gehäuse des Rechners läßt sich wie ein Buch aufklappen und schließen. Um es zu öffnen, halten Sie die Scharnierseite von sich weg und ziehen die zwei Seiten mit Hilfe der Daumen auseinander.



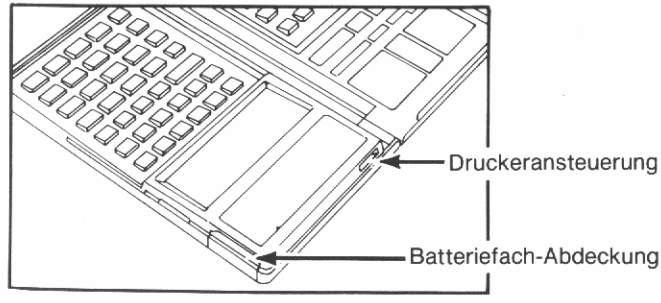
Um das Rechnergehäuse zu schließen, klappen Sie die zwei Seitenteile zusammen, bis Sie ein leises "Klicken" hören.

Sie können die linke Seite des Rechners vollständig umklappen, bis Sie an der Rückseite des rechten Tastenfelds anliegt. Dies ist sehr praktisch, wenn Sie unterwegs sind, den Rechner in einer Hand halten und mit der anderen Hand die erforderlichen Tasten drücken wollen, oder wenn Sie schlicht Platz auf Ihrem Schreibtisch einsparen möchten.



Aufsuchen des Batteriefachs und der Druckeransteuerung

Beachten Sie—bei aufgeklapptem Rechner—die Stelle des Batteriefachs und der Druckeransteuerung.



Der HP-28S wird mit 3 Alkali-Batterien (Typ N) betrieben. Sollten die beigefügten Batterien nicht bereits im Rechner eingesetzt sein, so folgen Sie den Anleitungen, die auf Seite 286 beginnen.

Wenn Sie den HP-28S zusammen mit einem Drucker betreiben, überträgt der Rechner die Information über ein Infrarotsignal an den Drucker. Druckoperationen sind in Kapitel 15 beschrieben.

Ein- und Ausschalten des Rechners

Drücken Sie [ON], um den Rechner einzuschalten. Da der HP-28S einen *Permanentspeicher* besitzt, bleiben die von Ihnen eingetippten Daten auch nach dem Ausschalten erhalten.

Wenn der Rechner eingeschaltet ist, wirkt die Taste [ON] als ATTN (*ATTentioN*). ATTN entspricht "Grundstellung", d.h. das Drücken bewirkt das Löschen von Text und das Abbrechen von Programmen.

Drücken Sie ■ [OFF], um den Rechner auszuschalten (zuerst Drücken der Umschalttaste ■, und danach Drücken der Taste, über welcher OFF steht).

Um den Batteriesatz zu schonen, schaltet sich der HP-28S etwa 10 Minuten nach dem letzten Tastendruck automatisch ab. Drücken Sie einfach [ON], um ihn wieder einzuschalten.

Löschen des gesamten Speicherbereichs

Sie können den Rechner in seinen ursprünglichen Status zurücksetzen, indem Sie ein "Memory Reset" durchführen, was das Löschen aller gespeicherten Daten zur Folge hat. Alle geänderten Modi (Dezimalzeichen, Winkelmodus, usw.) werden auf deren *Voreinstellung* zurückgesetzt.

Ausführen des Memory Resets:

1. Drücken Sie [ON] und halten Sie die Taste gedrückt.
2. Drücken Sie [INS] (in der oberen linken Ecke des rechten Tastenfelds) und halten Sie die Taste gedrückt.
3. Drücken Sie [▶] (in der oberen rechten Ecke des rechten Tastenfelds) und geben Sie die Taste wieder frei.
4. Geben Sie [INS] frei.
5. Geben Sie [ON] frei.

Der Rechner gibt ein Akustiksignal aus und in der Anzeige erscheint die Meldung `Memory Lost`. Diese Meldung wird durch das Drücken einer beliebigen Taste wieder gelöscht.

Wenn Sie den Löschvorgang begonnen haben, Ihre Absicht sich inzwischen jedoch geändert hat, dann *halten Sie weiterhin* [ON] *gedrückt*, während Sie [DEL] (neben der Taste [INS]) drücken; lassen Sie anschließend [ON] los. Das Drücken von [DEL] hebt die begonnene Operation wieder auf.

Einstellen des Anzeigekontrasts

Um die verschiedenen Lichtverhältnisse oder Blickwinkel individuell abstimmen zu können, läßt sich der Anzeigekontrast variieren.

Um den Kontrast einzustellen:

1. Drücken Sie [ON] und halten Sie die Taste gedrückt.
2. Drücken Sie [+] (evtl. mehrmals), um den Kontrast dunkler einzustellen, oder drücken Sie [−] (evtl. mehrmals), um den Kontrast heller einzustellen.
3. Geben Sie [ON] frei.

Berechnungen über das Tastenfeld

Versuchen Sie folgende Berechnung:

$$(15 + 23) \times \sin 30°$$

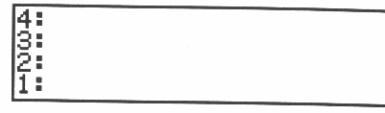
Die grundsätzlichen Schritte sind identisch mit der Verwendung von Bleistift und Papier. Zuerst wird die Summe von 15 + 23 berechnet, was zu einem Zwischenergebnis führt. Danach berechnen Sie den Sinus von 30°, was ein weiteres Zwischenergebnis erzeugt. Schließlich werden die Zwischenergebnisse zu einem Endergebnis zusammengefaßt.

Wenn Ihnen bei der Eingabe ein Fehler unterläuft, können Sie:

- [◄] drücken, um die zuletzt eingetippte Ziffer zu löschen.
- [ON] drücken, um alle eingetippten Ziffern zu löschen.

Beginnen Sie mit einem leeren Blatt Papier.

■ CLEAR



In der Anzeige wird der *Stack* angezeigt, welcher Ihren Arbeitsbereich darstellt. Momentan ist der Stack noch leer.

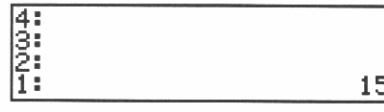
Drücken Sie 1 5, um 15 in die *Befehlszeile* zu schreiben.

15

Beachten Sie, daß der Stack nach oben geschoben wird, um Platz für die Befehlszeile zu erzeugen. Es werden also nur drei Stackebenen angezeigt.

Speichern Sie 15 im Stack.

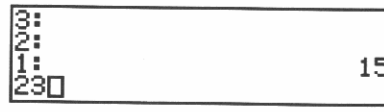
ENTER



Die Zahl wird in *Ebene 1* des Stacks übernommen, was durch 1: gekennzeichnet ist. Beachten Sie, daß die Befehlszeile verschwunden ist, so daß wieder 4 Stackebenen angezeigt werden.

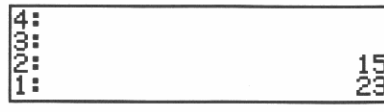
Drücken Sie 2 3, um 23 in die Befehlszeile einzutippen.

23



Speichern Sie 23 in Ebene 1.

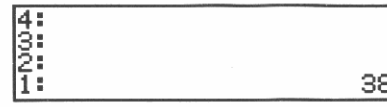
ENTER



Die Zahl 15, welche in Ebene 1 war, wird automatisch in Ebene 2 geschoben.

Addieren Sie 15 und 23.

[+]



Die Zahlen 15 und 23 werden vom Stack genommen und deren Summe, 38, wird nun in Ebene 1 zurückgegeben. Lassen Sie dieses Zwischenergebnis im Stack, während Sie das zweite Zwischenergebnis berechnen.

Um sin 30° zu berechnen, ist das TRIG Menü zu verwenden.

[TRIG]

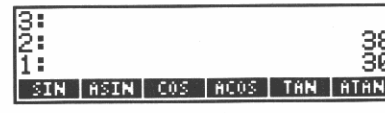
In der Grundzeile erscheinen nun sechs Befehle des TRIG Menüs. Die sechs *Menüfelder* (SIN bis ATAN) definieren die Funktion der *Menütasten* (die sechs unbeschrifteten Tasten unmittelbar unterhalb der Anzeige).

Drücken Sie [3] [0], um die Zahl 30 in die Befehlszeile einzugeben.

30

Speichern Sie 30 in Ebene 1.

[ENTER]



Das seitherige Ergebnis, 15 + 23 = 38, wird in Ebene 2 angehoben.

Berechnen Sie sin 30°.

SIN

Die Zahl 30 wird vom Stack genommen und dafür wird ihr Sinuswert, 0.5, in Ebene 1 zurückgegeben. Das erste Zwischenergebnis bleibt in Ebene 2.

Berechnen Sie 38 × 0.5.

[×]

Die Zahlen 38 und .5 werden dabei vom Stack genommen und deren Produkt, 19, wird in Ebene 1 des Stacks zurückgegeben.

Damit ist der gesamte Rechenvorgang abgeschlossen:

$$(15 + 23) \times \sin 30° = 19.$$

Als Zusammenfassung hier nun ein allgemeines Modell der soeben durchgeführten Berechnung:

1. Tippen Sie eine Zahl in die *Befehlszeile* ein.
2. Drücken Sie [ENTER], um die eingetippte Zahl in den *Stack* einzugeben.
3. Wenn sich die entsprechenden Zahlen im Stack befinden, drücken Sie die Taste zur Ausführung des gewünschten Befehls. (Wenn der Befehl nicht direkt auf oder über der Taste aufgedruckt ist, wählen Sie das *Menü*, das den Befehl enthält, und drücken anschließend die unterhalb dem *Menüfeld* liegende *Menütaste*.)

Das vorangehende Beispiel demonstriert, daß alle Berechnungen im Stack ausgeführt werden. Um diesen Vorgang hervorzuheben, hatten Sie jeweils [ENTER] gedrückt, um die Zahlen in den Stack einzugeben. *In der Praxis ist* [ENTER] *nur zur Trennung zweier nacheinander eingetippter Zahlen zu drücken.* Wiederholen Sie dieses Beispiel, ohne ein zweites und drittes Mal [ENTER] zu drücken.

Die oben demonstrierte Durchführung einer Berechnung (Eingabe der Zahlen in den Stack und anschließende Ausführung der mathematischen Operation) wird mit UPN (*Umgekehrte Polnische Notation*) oder Stack-Logik bezeichnet. Fast alle Befehle des HP-28S, nicht nur Rechenbefehle, verwenden Stack-Logik. Dieses System beruht auf zwei einfachen Regeln:

- Die Eingaben für eine Funktion—als *Argumente* der Funktion bezeichnet—müssen sich vor der Ausführung der Funktion im Stack befinden.
- Die Ergebnisse einer Funktion werden in den Stack zurückgegeben, wo sie als Argumente für die nächste Funktion verwendet werden können.

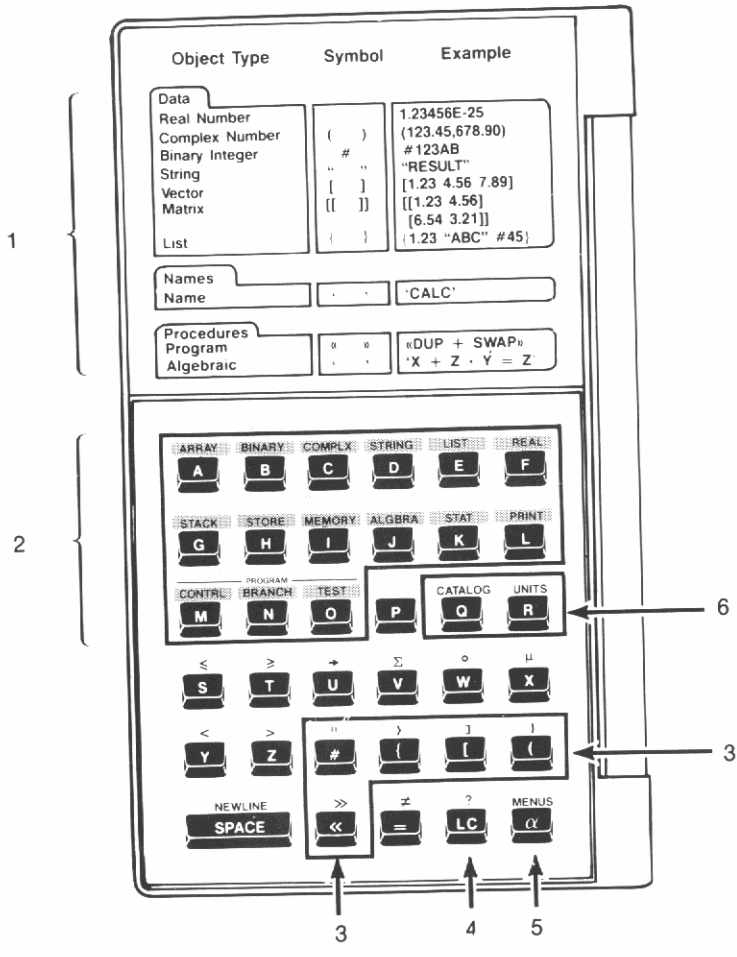
Sie können auch Berechnungen ausführen, indem Sie einen Ausdruck in algebraischer Form, wie er in konventioneller Schreibweise auf Papier erscheint, eingeben. Im nächsten Kapitel werden Sie die gleiche Berechnung unter Verwendung eines algebraischen Ausdrucks durchführen.

Übersicht des Rechners

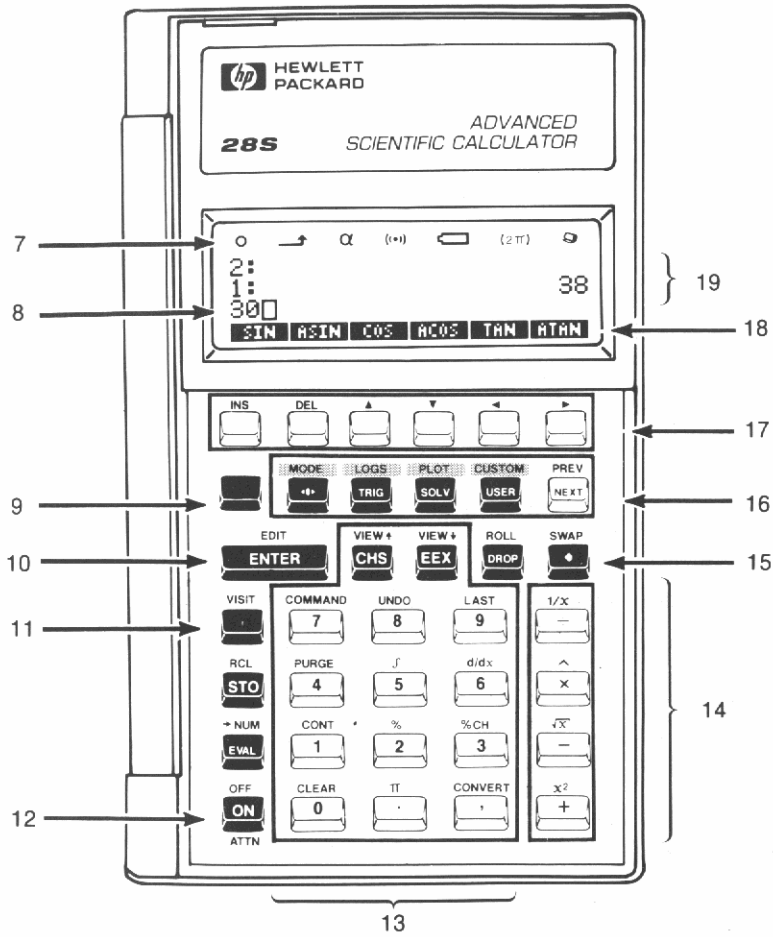
Dieser Abschnitt hebt einige wesentliche Leistungsmerkmale des Rechners hervor, einschließlich des Befehlskatalogs, welcher eine Beschreibung aller verfügbaren Befehle enthält.

Wichtigste Fähigkeiten und Konzepte

Die Abbildungen auf Seite 26 und 27 zeigen das Tastenfeld und die Anzeige, wobei die wichtigsten Fähigkeiten aufgeführt sind. Die Nummern in der nachfolgenden Beschreibung beziehen sich auf die Nummern in den Abbildungen.



1. Objekttypen und -Formate
2. Menüwahl (umgeschaltet)
3. Objekt-Begrenzungszeichen
4. Kleinschreibung
5. Eingabemodus
6. Befehls- und Einheiten-Katalog (umgeschaltet)



7. Indikatoren
8. Befehlszeile
9. Umschalttaste
10. Eingabe der Befehlszeile
11. Begrenzungszeichen für symbolische Objekte
12. Rechner ein/aus; Löschen der Befehlszeile; Programmabbruch
13. Eingabe von Zahlen
14. Arithmetische Funktionen
15. Rückschritt-Taste
16. Menüwahl, nächste Menüzeile
17. Menütasten
18. Menüfelder
19. Stackebenen

1. Objekttypen und -Formate. Diese Tabelle zeigt die erforderlichen Begrenzungszeichen hinsichtlich der 10 grundsätzlichen Arten von Objekten. Jedes der individuellen Elemente, mit welchen Sie auf Ihrem Rechner arbeiten, wird als "Objekt" bezeichnet. Bei den 10 Objekttypen handelt es sich um:

- Reelle Zahlen, wie z.B. 5 oder -4.3×10^{15}.
- Komplexe Zahlen, welche aus einem Datenpaar von reellen Zahlen bestehen und die eine komplexe Zahl der Form $x + iy$ oder den Punkt einer Ebene darstellen.
- Binärwerte, die aus vorzeichenlosen ganzen Zahlen bestehen und sich auf das duale, oktale, dezimale oder hexadezimale Zahlensystem beziehen.
- Strings, die beliebige Zeichensequenzen enthalten können.
- Vektoren, welche aus eindimensionalen Feldern bestehen und im Bereich der linearen Algebra verwendet werden.
- Matrizen, welche aus zweidimensionalen Feldern bestehen und im Bereich der linearen Algebra verwendet werden.
- Listen, die beliebige Sequenzen von Objekten enthalten können.
- Namen, die Ihnen erlauben, andere Objekte zu speichern; außerdem sind über diese symbolische Berechnungen möglich.
- Programme, welche das Erzeugen eigener Befehle ermöglichen.
- Algebraische Objekte, die Ausdrücke und Gleichungen darstellen.

2. Menüwahl (umgeschaltet). Zur Zuweisung der entsprechenden Befehle für Menütasten. Drücken Sie z.B. ■ ARRAY zur Auswahl des ARRAY Menüs. Sie können jedes beliebige andere Menü durch Drücken der jeweiligen Menüwahl-Taste aufrufen.

3. Objekt-Begrenzungszeichen. Diese Symbole identifizieren die unterschiedlichen Objekttypen. So kennzeichnet z.B. # Binärwerte, während « und ■ » zur Identifikation von Programmen dienen.

Reelle Zahlen erfordern keine Begrenzungszeichen. Symbolische Objekte (Namen und algebraische Objekte) erfordern das Zeichen ', welches sich auf dem rechten Tastenfeld befindet (siehe Element 11).

19. Kleinschreibung. Drücken Sie LC, um Ihre Alphazeichen in Kleinschreibung einzugeben. Dieser Modus bleibt erhalten, bis Sie entweder erneut LC drücken, ENTER zur Verarbeitung der Befehlszeile drücken, oder ON betätigen.

5. Eingabemodus. Die Befehlszeile hat drei Eingabemodi, jeder für einen spezifischen Objekttyp passend. Der Eingabemodus ändert sich automatisch in Abhängigkeit Ihrer Objekteingabe; manchmal ist eine manuelle Kontrolle jedoch wünschenswert. Die [α] Taste ermöglicht Ihnen die entsprechende Auswahl.

6. Befehls- und Einheiten-Katalog. Drücken Sie ■ [CATALOG], um eine Auflistung aller im HP-28S enthaltenen Befehle und deren benötigten Argumente (Seite 31) zu erhalten. Drücken Sie ■ [UNITS] zur Anzeige der für Konvertierungen zulässigen Einheiten (Seite 141).

7. Indikatoren. Diese weisen auf den jeweiligen Rechnerstatus hin. Sie haben dabei folgende Bedeutung:

Indikator	Bedeutung
O	Die Ausführung eines Programms wurde unterbrochen.
↗	Die Umschalttaste ■ wurde gedrückt.
α	Alpha-Eingabemodus wurde aktiviert.
((•))	Der HP-28S ist am Arbeiten—es ist keine Eingabe möglich.
▭	Schwache Batterien.
(2π)	Als Winkelmodus wurde Bogenmaß gewählt.
🖷	Der Rechner sendet Daten an den Drucker.

8. Befehlszeile. Die eingetippten Daten erscheinen zunächst in der Befehlszeile.

9. Umschalttaste. Drücken Sie die rote Taste ■, um den über der Taste aufgedruckten Befehl auszuführen.

10. Eingabe der Befehlszeile. Drücken Sie [ENTER], um den Inhalt der Befehlszeile zu verarbeiten.

11. Begrenzungszeichen für symbolische Objekte. Begrenzungszeichen sind spezielle Interpunktionszeichen zur Identifikation der unterschiedlichen Objekttypen; symbolische Objekte bestehen aus Namen und algebraischen Ausdrücken. Um ein symbolisches Objekt einzugeben, drücken Sie ['] zu Beginn der Eingabeoperation (und wenn erforderlich, auch am Ende des Objekts).

Reelle Zahlen erfordern keine Begrenzungszeichen. Eine Übersicht aller Begrenzungszeichen finden Sie auf dem linken Tastenfeld. (Sehen Sie dazu auch die Elemente 1 und 3.)

12. Rechner ein/aus, Löschen der Befehlszeile, Programmabbruch. Drücken Sie [ON], um den Rechner einzuschalten; das Ausschalten erfolgt durch die Tastenfolge ■[OFF]. (OFF steht oberhalb der Taste [ON]. "Drücken Sie ■[OFF]" bedeutet, Betätigen der Taste ■ und danach Drücken von [ON].)

Wenn der Rechner eingeschaltet ist, fungiert die Taste [ON] auch als ATTN (*ATTentioN*) Taste—eine Art "Grundstellung", wobei die Befehlszeile gelöscht und ein eventuell gerade laufendes Programm abgebrochen wird. (ATTN steht unterhalb der Taste [ON].)

13. Eingabe von Zahlen. Um Zahlenwerte einzutippen, sind die Zifferntasten [0] bis [9], [CHS] (*CHange Sign bzw. Vorzeichenwechsel*) und [EEX] (*Eingabe EXponent*) zu verwenden. In Abhängigkeit von Ihrer Wahl des Dezimalzeichens (siehe Seite 36) ist entweder [.] oder [,] zum Trennen von ganzzahligem Teil und gebrochenem Teil der Zahl zu verwenden. Die Eingabe von Zahlen ist auf Seite 39 erläutert.

14. Arithmetische Funktionen. Die arithmetischen Funktionen sind unter "Einwertige Funktionen" auf Seite 40 sowie unter "Zweiwertige Funktionen" auf Seite 41 beschrieben.

15. Rückschritt-Taste. Drücken Sie [◄], wenn das zuletzt eingetippte Zeichen gelöscht werden soll.

16. Menüwahl, nächste Menüzeile. Benutzen Sie die Menüwahl-Tasten, um den Menütasten entsprechende Befehle zuzuweisen. Wenn Sie z.B. das TRIG Menü aufrufen möchten, drücken Sie [TRIG]. Zum Aufrufen eines anderen Menüs ist die gewünschte Menüwahl-Taste zu drücken.

Sind keine Menüfelder angezeigt, so ist das *Cursor-Menü* aktiv. Die Operationen innerhalb des Cursor-Menüs ([INS] bis [▶]) sind in weisser Schrift über den Menütasten aufgedruckt. Erscheinen die Menüfelder in der Anzeige, so können Sie durch Drücken von [◄►] das Cursor-Menü aufrufen. Sie erhalten das zuvor angezeigte Menü wieder angezeigt, wenn Sie erneut [◄►] drücken.

Enthält ein Menü mehr als eine Zeile (mit je 6 Menüfeldern), so drücken Sie [NEXT] zur Anzeige der nächsten Zeile. Mit ■[PREV] erhalten Sie die vorherige Zeile angezeigt.

Es gibt weitere Menüwahl-Tasten auf dem linken Tastenfeld (siehe Element 2). Eine alphabetische Auflistung aller verfügbaren Menüs, sowie eine kurze Beschreibung deren Befehle finden Sie in Anhang B.

17. Menütasten. Die Funktion der Menütasten wird durch die zugehörigen Menüfelder definiert. Sind keine Menüfelder angezeigt, so führen diese Tasten die Operationen des Cursor-Menüs aus, die in weißer Beschriftung über den Tasten angebracht sind.

18. Menüfelder. Die Menüfelder zeigen die momentane Funktion der Menütasten an.

19. Stackebenen. Im Stack werden diejenigen Objekte angezeigt, mit welchen Sie momentan arbeiten. Jede der numerierten Stackebenen (Ebene 1, Ebene 2, usw.) enthält ein Objekt.

Der Befehlskatalog

Der HP-28S enthält eine alphabetische Auflistung aller Befehle. Für jeden Befehl wird dabei seine Anwendung—d.h. die vom Befehl benötigten Argumente—angezeigt. Eine vollständige Beschreibung eines Befehls finden Sie im "Verzeichnis der Operationen" am Ende des Referenzhandbuchs.

Rufen Sie den Katalog auf.

CATALOG

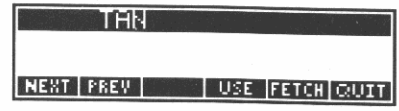
Als erster Befehl ist ABORT gelistet.

Während der Kataloganzeige wird die normale Arbeitsweise unterbrochen. Mit Hilfe der Menütasten NEXT und PREV können Sie den Katalog durchsehen. Mit der Operation USE läßt sich die Anwendungsweise, d.h welche Argumente von dem Befehl benötigt werden, anzeigen. Über die Operationen FETCH und QUIT wird die Anzeige des Katalogs abgeschlossen und zur normalen Arbeitsweise des Rechners zurückgekehrt.

Drücken Sie NEXT und PREV, um durch den Katalog zu "blättern". Sie können dabei die Tasten auch gedrückt halten.

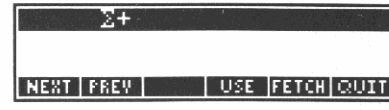
Sie können zum ersten Eintrag unter einem bestimmten Buchstaben durch Drücken der Buchstabentaste gelangen. Versuchen Sie z.B. "T".

T



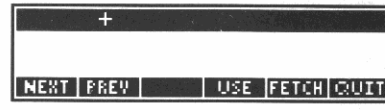
Der erste "T" Befehl ist die TAN Funktion. Wird statt einer Buchstabentaste eine andere Taste des linken Tastenfelds gedrückt, so wird der erste Eintrag des jeweiligen Symbols angezeigt. Versuchen Sie zum Beispiel "Σ".

■ Σ



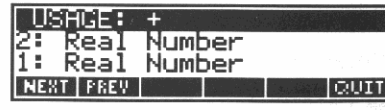
Der erste "Σ" Befehl ist Σ+. Kann unter der gedrückten Symboltaste kein Eintrag gefunden werden, so wird + angezeigt, der erste nichtalphabetische Befehl.

#



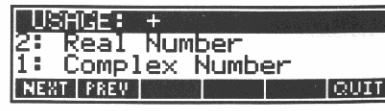
Überprüfen Sie die Anwendung von +.

USE



Dies zeigt, daß Sie zwei reelle Zahlen addieren können. Überprüfen Sie die nächste Kombination.

NEXT



Dies bedeutet, daß Sie eine reelle Zahl und eine komplexe Zahl addieren können. Überprüfen Sie die nächste Kombination.

NEXT

Hieraus ist ersichtlich, daß die Reihenfolge der Zahlen ohne Bedeutung ist.

Überprüfen Sie die 14 restlichen Kombinationen. Als letzte Kombination erhalten Sie:

Nachdem Sie sich die Anwendungsmöglichkeiten angesehen haben, kehren Sie zur Hauptanzeige des Katalogs zurück.

QUIT

Sie können nun einen anderen Katalogeintrag anzeigen lassen und dessen Kombinationen von Argumenten überprüfen. Wenn Sie mit der Durchsicht des Katalogs fertig sind, kehren Sie zur üblichen Anwendungsweise des Rechners zurück.

QUIT

Alternativ dazu können Sie den Katalog auch durch Drücken von FETCH abschließen, wodurch zusätzlich der Name des zuletzt angezeigten Befehls in die Befehlszeile geschrieben wird.

2

Arithmetische Operationen

Zur Durchführung arithmetischer Operationen stehen Ihnen auf dem HP-28S zwei Wege zur Verfügung. Sie können dazu entweder den Stack verwenden, wie Sie es im vorangehenden Beispiel getan haben, oder Sie können einen *Ausdruck* eingeben, der die zu lösende Aufgabenstellung enthält. Im vorigen Beispiel berechneten Sie die Aufgabe:

$$(15 + 23) \times \sin 30°$$

Hier ist beschrieben, wie Sie die gleiche Aufgabe unter Verwendung eines Ausdrucks lösen.

Löschen Sie den Stackinhalt und wählen Sie das TRIG Menü.

[▪] [CLEAR] [TRIG]

Beginn des Ausdrucks.

[']

Der Cursor ändert seine Form, um den *algebraischen Eingabemodus* anzudeuten.

Tippen Sie den ersten Teil des Ausdrucks ein.

[(] 15 [+] 23 [▪] [)]

Aufgrund des algebraischen Eingabemodus bewirkt das Drücken von [+] das Schreiben von + in die Befehlszeile, anstatt der Ausführung des Befehls.

Fahren Sie mit der Eingabe des Ausdrucks fort.

[×] SIN

Da Sie sich immer noch im algebraischen Eingabemodus befinden, erscheint nach Drücken von [×] das * Zeichen und durch Betätigen von SIN der Ausdruck SIN(in der Befehlszeile, ohne daß die Befehle ausgeführt wurden.

Übernehmen Sie den Ausdruck in den Stack.

30 [ENTER]

Die schließende Klammer) und das Begrenzungszeichen ' werden automatisch hinzugefügt.

Werten Sie den Ausdruck aus.

[EVAL]

Der Ausdruck wird im Stack gelöscht und als Ergebnis wird 19 in Stackebene 1 zurückgegeben.

Damit ist die Berechnung des Beispiels abgeschlossen.

$$(15 + 23) \times \sin 30° = 19.$$

Aufgabenstellungen, welche in Form eines algebraischen Ausdrucks vorliegen, lassen sich leichter durch direkte Eingabe und anschließende Auswertung lösen. Alternativ dazu liegen Vorteile bei der Berechnung über den Stack darin, daß Zwischenergebnisse angezeigt werden und eine fortlaufende Lösungsentwicklung möglich ist.

Die Beziehung zwischen Berechnungen im Stack und Ausdrücken wird in Kapitel 4, "Wiederholen einer Berechnung", erläutert. Kapitel 4 zeigt Ihnen, wie Sie Berechnungen im Stack durchführen und unter Verwendung von Variablen algebraische Ausdrücke erzeugen.

Eingeben und Anzeigen von Zahlen

Es gibt bestimmte Modi, die sich auf das Anzeigeformat von Zahlen auswirken. Um Ihnen die Auswahlmöglichkeiten zu demonstrieren, ist die Zahl ⅔ im Stack zu speichern.

Speichern Sie 2 im Stack.

2 ENTER

Dividieren Sie durch 3.

3 ÷

Das Ergebnis, ⅔, wird in Ebene 1 zurückgegeben. Der Wert ist die dezimale Näherung an ⅔, wie sie per Voreinstellung für Dezimalzeichen und Zahlen-Anzeigeformat angezeigt wird. Der nächste Abschnitt beschreibt weitere Auswahlmöglichkeiten.

Wählen des Dezimalzeichens

Sie können wählen, ob Sie einen Punkt oder ein Komma als *momentanes Dezimalzeichen* benutzen möchten. Diese Wahl ist entscheidend hinsichtlich der Eingabe von Zahlen und deren Erscheinungsform in der Anzeige.

Sie können das Dezimalzeichen wie folgt festlegen:

Wählen Sie das MODE Menü.

■ MODE

Es wird die erste Zeile des MODE Menüs angezeigt. Gehen Sie zur zweiten Zeile über.

NEXT

Wählen Sie das Komma als Dezimalzeichen.

RDX,

Der Dezimalpunkt wird durch ein Komma ersetzt und im Menüfeld für RDX, erscheint ein kleines Quadrat zur Kennzeichnung der getroffenen Auswahl.

Schalten Sie wieder auf den Dezimalpunkt um.

RDX,

Wählen des Zahlen-Anzeigeformats

Sie können wählen, wieviel Dezimalstellen angezeigt werden sollen.

Kehren Sie zur ersten Menüzeile des MODE Menüs zurück.

NEXT

Sie gelangen von der letzten Menüzeile zur ersten, indem Sie NEXT drücken. Da das MODE Menü nur zwei Zeilen hat, kommen Sie durch Drücken von NEXT zurück zur ersten Zeile.

Es gibt vier grundsätzliche Wahlmöglichkeiten für das Zahlen-Anzeigeformat. STD (*STandarD*), FIX (*FIXed*), SCI (*SCIentific/wissenschaftlich*) und ENG (*ENGineering/technisch*). Das Feld für STD enthält ein Quadrat zur Kennzeichnung der momentanen Einstellung. In diesem Format hängt die Anzahl der Dezimalstellen vom Betrag der Zahl ab. Bei einer ganzen Zahl erscheinen keine Nachkomma-

stellen; für das oben abgebildete Beispiel wird die maximale Anzahl von 12 Stellen verwendet.

Die anderen Anzeigeformate zeigen eine vorgegebene Anzahl von Dezimalstellen—von 1 bis 11—unabhängig vom momentan angezeigten Zahlenwert. Nachstehend werden alle anderen Formate mit zwei Nachkommastellen demonstriert. Dabei wird nur der Wert in der Anzeige gerundet—die interne Darstellung der Zahlen bleibt gleich.

Lassen Sie sich ⅔ anzeigen, auf zwei Dezimalstellen gerundet.

2 FIX

Zeigen Sie ⅔ in der Form von *Mantisse* und *Exponent* an, wobei die Mantisse auf zwei Stellen gerundet wird.

2 SCI

Der Betrag der Zahl ergibt sich als Produkt der Mantisse und der Basis 10, die mit dem angezeigten Exponenten potenziert wird. Die Mantisse liegt immer im Bereich zwischen 1 und 9.99999999999.

Zeigen Sie ⅔ als Mantisse und Exponent an, wobei der Exponent ein Mehrfaches von drei darstellt.

2 ENG

Kehren Sie zum normalen Zahlen-Anzeigeformat zurück.

STD

Zahleneingabe

Sie können Zahlen als Mantisse und Exponent eingeben, wobei sich der Wert der Zahl als Produkt von Mantisse und der Basis 10, potenziert mit dem jeweiligen Exponenten, ergibt. Beide (Mantisse und Exponent) können dabei ein negatives Vorzeichen annehmen.

Tippen Sie z.B. die Zahl -4.2×10^{-12} ein.

Tippen Sie zuerst die Ziffern für die Mantisse ein.

4.2

Wenn Ihnen ein Fehler unterlaufen ist, können Sie [◆] drücken, um die falschen Zeichen zu löschen und danach die richtigen einzutippen.

Als nächstes negieren Sie die Mantisse.

[CHS]

"CHS" (*CHange Sign*) bewirkt einen Vorzeichenwechsel—erneutes Drücken von [CHS] würde das Vorzeichen wieder umkehren.

Beginnen Sie nun mit dem Exponenten.

[EEX]

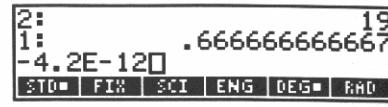
"EEX" (*Enter EXponent*) bedeutet "Eingabe Exponent". Es erscheint ein E in der Befehlszeile. Sollten Sie versehentlich [EEX] gedrückt haben, so können Sie E durch Drücken von [◆] wieder löschen.

Tippen Sie den Exponenten en.

12

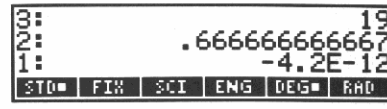
Negieren Sie den Exponenten.

[CHS]



Speichern Sie Zahl im Stack.

[ENTER]



Denken Sie daran, [CHS] zur Eingabe negativer Zahlen zu verwenden. Enthält dieses Handbuch z.B. die Tastenfolge -4 [×], so müssen Sie die Tasten [4] [CHS] [×] drücken.

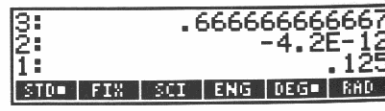
Einwertige Funktionen

Funktionen, welche sich auf eine einzelne Zahl bzw. Variable beziehen—z.B. das Negieren oder das Quadrieren einer Zahl—werden als einwertige Funktionen bezeichnet. Sie wirken immer auf die Zahl in Ebene 1. Es gibt vier einwertige Funktionen auf dem Tastenfeld:

- Drücken Sie [CHS] zum Vorzeichenwechsel einer Zahl.
- Drücken Sie [■] [1/x] zur Berechnung des Kehrwerts einer Zahl.
- Drücken Sie [■] [√], um die Quadratwurzel zu berechnen.
- Drücken Sie [■] [x²] zum Quadrieren einer Zahl.

Wenn Sie eine Zahl eintippen, muß vor der Ausführung der einwertigen Funktion nicht noch die Taste [ENTER] gedrückt werden—das Drücken der Funktionstaste bewirkt eine automatische Ausführung von ENTER. Zum Beispiel könnten Sie ⅛ wie folgt berechnen:

8 [■] [1/x]



Zweiwertige Funktionen

Funktionen, die sich auf zwei Zahlen beziehen—wie z.B. Addition oder Multiplikation—werden als zweiwertige Funktionen bezeichnet. Sie alle wirken auf die Zahlen in den Stackebenen 1 und 2.

Wenn Sie beide Argumente der Funktion eintippen (wie bei der Division von 2 durch 3 auf Seite 36), so müssen Sie [ENTER] zum Trennen der beiden Argumente drücken. Befinden sich die Argumente bereits im Stack, dann ist das Drücken von [ENTER] nicht erforderlich.

Addieren und Subtrahieren

Berechnen Sie 36 + 17.

36 [ENTER]
17 [+]

Das Ergebnis ist 53.

Bei der Addition spielt die Reihenfolge der Zahlen keine Rolle, da die Addition eine kommutative Funktion ist. Für die Subtraktion ist die Reihenfolge der Zahlen jedoch von Bedeutung. Berechnen Sie als nächstes 91 − 27.

91 [ENTER]
27 [−]

Als Ergebnis erhalten Sie 64.

Multiplizieren und Dividieren

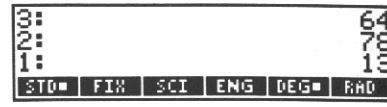
Berechnen Sie 13 × 6.

13 [ENTER]
6 [×]

Das Ergebnis ist 78.

Bei der Multiplikation ist die Reihenfolge der Zahlen ohne Bedeutung. Allerdings ist die Reihenfolge bei der Division zu beachten. Berechnen Sie nun 182/14.

182 ENTER
14 ÷



Als Ergebnis erhalten Sie 13.

Potenzieren und Radizieren

Die Reihenfolge der Zahlenwerte ist für beide Funktionen von Bedeutung. Berechnen Sie 5^3.

5 ENTER
3 ■ ^

Sie erhalten 125 als Ergebnis.
Um $\sqrt[4]{2401}$ zu berechnen, geben Sie zuerst 2401 in den Stack ein.

2401 ENTER

Potenzieren Sie nun 2401 mit ¼.

4 ■ 1/x ■ ^

Das Ergebnis ist 7.

Prozentrechnung

Berechnen Sie 40% von 85.

85 ENTER
40 ■ %

Als Ergebnis erhalten Sie 34.

Die Reihenfolge der Zahlenwerte ist bei der einfachen Prozentrechnung ohne Bedeutung; bei der Berechnung der prozentualen Änderung ist sie jedoch zu beachten. Berechnen Sie jetzt die prozentuale Änderung von 60 auf 75.

60 ENTER
75 ■ %CH

Das Ergebnis ist +25, d.h. 75 ist um 25% größer als 60.

Tauschen der Inhalte von Ebene 1 und 2

Für alle Funktionen, wo die Reihenfolge der Zahlen von Bedeutung ist—wie z.B. Subtraktion, Division, Potenzen, Wurzeln und prozentuale Änderung—können Sie die Reihenfolge der Zahlen durch Drücken von ■ SWAP ändern. Wenn Sie zum Beispiel momentan 25 in Ebene 1 gespeichert haben und Sie 30 − 25 berechnen möchten:

Tippen Sie 30 ein.

30

Tauschen Sie die Folge der Zahlen im Stack.

■ SWAP

Beachten Sie, daß SWAP automatisch ENTER ausführte.

Subtrahieren Sie 25 von 30.

[−]

Das Ergebnis ist 5.

Löschen von Objekten im Stack

Wenn Sie alle vorangehenden Beispiele durchgearbeitet haben, dann befindet sich inzwischen eine beträchtliche Anzahl von Zahlen-Objekten im Stack. Dieser wächst nämlich bei jeder Objekt-Eingabe, wobei die Objekte so lange im Stack verbleiben, bis sie durch eine Operation vom Stack genommen oder von Ihnen gelöscht werden.

Sie können Objekte einzeln oder alle gleichzeitig löschen.

Löschen Sie die Zahl in Ebene 1.

[DROP]

Objekte in höheren Ebenen werden jeweils um eine Ebene nach unten verschoben.

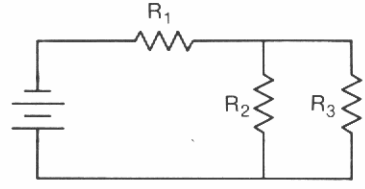
Löschen Sie alle Objekte im Stack.

[■] [CLEAR]

Es empfiehlt sich, vor der Lösung einer neuen Aufgabenstellung den Stackinhalt zu löschen. Sie wissen dann, daß die während der Bearbeitung der Aufgabe im Stack angehäuften Werte sich auf das momentane Problem beziehen und nicht von vorherigen Berechnungen übrig geblieben sind.

Kettenrechnungen

Bei der Ausführung von komplexeren Berechnungen dient der Stack zur temporären Speicherung von Zwischenresultaten. Die temporäre Speicherform ergibt sich dabei ganz automatisch. Nehmen Sie z.B. an, Sie haben einen Kurs in Elektrotechnik 1 begonnen und möchten den Gesamtwiderstand des nachfolgenden Schaltkreises berechnen:



Die Gleichung zur Lösung dieser Aufgabenstellung lautet:

$$R_{ges} = R_1 + \frac{1}{\frac{1}{R_2} + \frac{1}{R_3}}$$

Wenn R_1, R_2 und R_3 jeweils einen Widerstand von 8, 6 und 3 Ohm haben, ist folgender konkreter Ausdruck zu lösen:

$$R_{ges} = 8 + \frac{1}{\frac{1}{6} + \frac{1}{3}}$$

Lösen Sie das Problem wie folgt:

Speichern Sie 8 im Stack.

8 [ENTER]

Lassen Sie 8 bis zur endgültigen Addition mit dem Rest des Ausdrucks gespeichert.

Speichern Sie ⅙ im Stack.

6 ■ [1/x]

Speichern Sie ⅓ im Stack.

3 ■ [1/x]

Addieren Sie die Kehrwerte von 6 und 3.

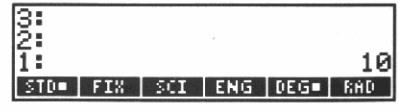
[+]

Ermitteln Sie den Reziprokwert der Summe.

■ [1/x]

Schließen Sie die Berechnung von R_{ges} ab.

[+]



Das Ergebnis ist 10 Ohm.

Wenn die falsche Funktion ausgeführt wurde

Der HP-28S enthält Rücksicherungsmöglichkeiten, welche Ihnen beim "Zurückgehen" nach dem versehentlichen Ausführen einer Funktion behilflich sind. Die nachfolgenden Schritte kehren die Auswirkung einer Funktion—ob ein- oder zweiwertig—wieder um.

1. Drücken Sie ■ UNDO, um den vorangehenden Stackinhalt (d.h. den vor der Ausführung der Funktion) zurückzuspeichern.
2. Wenn beim Auftreten des Fehlers eine Zahl in der Befehlszeile enthalten war, können Sie durch Drücken von ■ COMMAND den vorhergehenden Inhalt der Befehlszeile zurücksichern.
3. Setzen Sie nun Ihre Berechnung fort.

3

Verwenden von Variablen

Variablen erlauben Ihnen die Benennung eines Objekts. Sie erzeugen eine Variable, indem Sie ein Namen-Objekt mit einem beliebigen anderen Objekt assoziieren. Das Namen-Objekt definiert in diesem Fall den Variablennamen, und das zweite Objekt bestimmt den Variableninhalt. Über den Variablennamen können Sie dann auf deren Inhalt Bezug nehmen.

Variablen werden im *Benutzerspeicher*, ein vom Stack unabhängiger Speicherbereich des Rechners, gespeichert. Der primäre Zweck des Stacks liegt in der temporären Speicherung von Objekten (z.B. Zwischenergebnisse). Der Benutzerspeicher ist zur langfristigen Speicherung von Variablen gedacht, wie z.B. Zahlenwerte oder Ausdrücke, welche Sie wiederholt für Ihre Arbeit verwenden.

Dieses Kapitel beschreibt, wie sich numerische Variablen anwenden lassen, wobei jedoch Variablen für jeden Objekttyp verwendbar sind. So hat z.B. ein Programm keinen internen Namen. Sie benennen das Programm, indem Sie es in einer Variablen speichern; danach können Sie das Programm über den Variablennamen aufrufen und starten.

Die Schritte zum Erzeugen, Zurückrufen, Auswerten, Ändern oder Löschen einer Variablen sind für alle Variablen gleich, unabhängig von deren jeweiligem Inhalt. Diese Uniformität sorgt nicht nur für die einfache Handhabung des HP-28S, sondern auch für dessen Leistungsstärke, da nur wenige Ausnahmefälle vorkommen und eine größere Flexibilität gegeben ist.

Erste Operationen mit Variablen

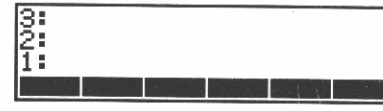
Numerische Variablen stellen die einfachsten Variablen dar. Nachstehend erfahren Sie, wie diese erzeugt, zurückgerufen und ausgewertet werden.

Erzeugen einer numerischen Variablen

Nehmen Sie an, Sie würden wiederholt ein Volumen berechnen und dazu die Umrechnungszahl 133 verwenden. Erzeugen Sie deshalb eine Variable mit dem Namen VOL (für "Volumen") wie folgt:

Löschen Sie den Stackinhalt und wählen Sie das USER Menü.

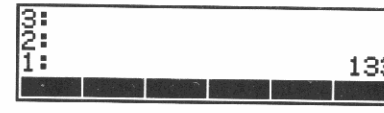
■ CLEAR USER



Das USER Menü zeigt die Benutzervariablen an. Es ist leer, da Sie noch keine Variablen erzeugt haben.

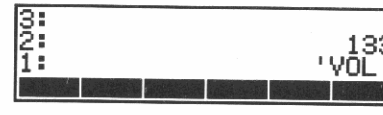
Speichern Sie die Zahl in Ebene 1.

133 ENTER



Speichern Sie den Namen `'VOL'` in Ebene 1.

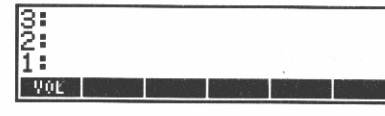
' VOL ENTER



Beachten Sie, daß ' automatisch am Ende hinzugefügt wurde. Die Zahl 133 wurde in Ebene 2 angehoben. (Das Drücken von ENTER ist nicht erforderlich, es wurde hier nur der Klarheit wegen angewendet.)

Erzeugen Sie die Variable VOL.

STO



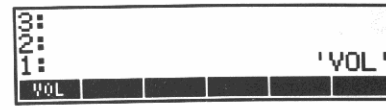
Die Zahl und der zugehörige Name wurden vom Stack genommen, wobei die Variable VOL mit einem Inhalt von 133 erzeugt wurde. Beachten Sie, daß VOL nun im USER Menü erscheint.

Zurückrufen einer numerischen Variablen

Nachdem Sie nun die Variable VOL erzeugt haben, rufen Sie deren Inhalt in den Stack zurück.

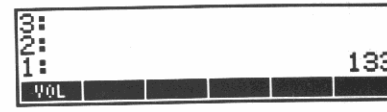
Geben Sie den Namen VOL in den Stack ein, wobei Sie vom USER Menü Gebrauch machen.

' VOL ENTER



Rufen Sie den Inhalt von VOL ab.

RCL



Sie erhalten die in VOL gespeicherte Zahl angezeigt.

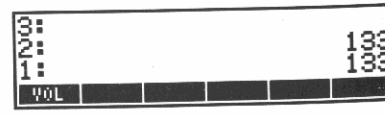
Bei einem Rechner mit Speicherregister ist das Zurückrufen von Registerinhalten ein vergleichbarer Vorgang. Im HP-28S werden Variableninhalte seltener abgerufen, dafür jedoch häufiger *ausgewertet*.

Auswerten einer numerischen Variablen

Bei numerischen Variablen bedeutet "auswerten" das selbe wie "zurückrufen"—Auswerten einer numerischen Variablen speichert deren Inhalt im Stack. Sie werden erkennen, daß Auswerten einfacher ist. (Sie werden später noch eine Programmvariable erzeugen und können daran erkennen, daß Auswerten und Zurückrufen unterschiedliche Ergebnisse zur Folge haben können.)

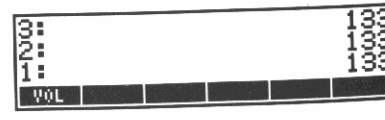
Speichern Sie den Inhalt von VOL im Stack, indem Sie VOL auswerten.

VOL



Sie können VOL auch dadurch auswerten, indem Sie den Namen ohne Anführungszeichen eintippen.

VOL ENTER



Ändern des Variableninhalts

Der Variableninhalt läßt sich durch die gleichen Schritte wie beim Erzeugen der Variablen ändern. Der neue Inhalt ersetzt dabei den alten.

Ändern Sie nun den Inhalt der Variablen VOL in 151.

Schreiben Sie den neuen Inhalt in die Befehlszeile.

151

Beachten Sie, daß der Cursor durch ein leeres Rechteck dargestellt ist. Seine Form wird sich im nächsten Schritt ändern.

Schreiben Sie den Variablennamen in die Befehlszeile.

[']

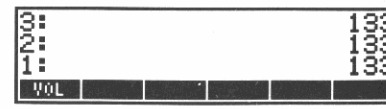
Der Cursor hat seine Form nach dem Drücken von ['] geändert, um den neuen *Eingabemodus* anzudeuten—die Interpretation der gedrückten Tasten.

Ursprünglich war die Befehlszeile im *unmittelbaren Ausführungsmodus*, geeignet für Berechnungen über das Tastenfeld. Das Drücken von ['], was einen Namen oder Ausdruck eingrenzt, verursachte eine Änderung in den *algebraischen Eingabemodus*:

- Drücken einer Funktionstaste, z.B. [+], verursacht hier das Schreiben des Zeichens + in die Befehlszeile, anstatt der Ausführung des Befehls.
- Drücken einer USER-Menütaste schreibt den Namen in die Befehlszeile, anstatt der Auswertung der Variablen.

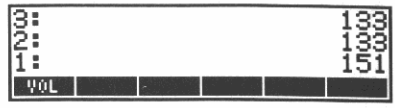
Speichern Sie nun den neuen Wert in der Variablen.

[STO]



Überprüfen Sie den neuen Wert.

VOL

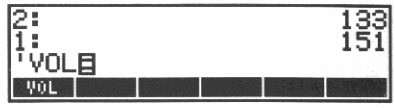


Löschen einer Variablen

Wenn Sie die Variable VOL nicht mehr benötigen, sollte sie im Benutzerspeicher gelöscht werden.

Schreiben Sie den Variablennamen in die Befehlszeile.

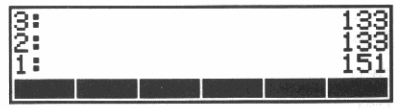
' VOL



(Das Anführungszeichen ' ist notwendig, um in diesem Fall das Auswerten der Variablen zu verhindern.)

Löschen Sie VOL im Benutzerspeicher.

PURGE



Beachten Sie, daß VOL nicht mehr im USER Menü angezeigt wird.

Ändern des Variablennamens

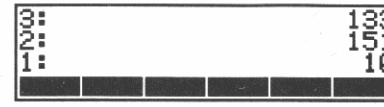
Sie können den Namen einer Variablen ändern, indem Sie eine neue Variable mit dem gleichen Inhalt erzeugen und die alte Variable löschen.

Im folgenden Abschnitt sind zuerst die Schritte zum Umbenennen einer Variablen durchzuführen. Anschließend werden Sie zum Schreiben eines Programms aufgefordert, welches die gleichen Schritte durchführt. Abschließend ist das Programm in einer Variablen zu speichern und über den Variablennamen aufzurufen bzw. auszuführen.

Als Vorbereitung ist eine Variable zu erzeugen—eine Variable A mit dem Inhalt 10, welche umbenannt werden soll.

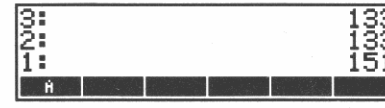
Speichern Sie den Wert 10 im Stack.

10 ENTER



Erzeugen Sie die Variable A.

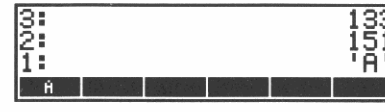
' A STO



Beachten Sie, daß A im USER Menü angezeigt wird.

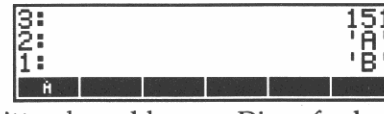
Nehmen Sie an, A soll in B umbenannt werden. Speichern Sie den alten Namen im Stack.

' A ENTER



Speichern Sie den neuen Namen im Stack.

' B ENTER



Damit sind die vorbereitenden Schritte abgeschlossen. Die erforderlichen Daten sind im Stack gespeichert. Alter und neuer Name dienen als *Argumente* für das Programm, wobei das Programm von der gegebenen Reihenfolge der Namen ausgeht. Die folgenden Schritte stellen das eigentliche Programm dar:

Drei allgemeine Stack-Manipulationen (OVER, ROT und SWAP). Sie werden deren Funktion während deren Ausführung kennenlernen.

Kopieren Sie den alten Namen in Ebene 1 (unter Verwendung des Befehls OVER im STACK Menü).

STACK OVER

Rufen Sie den Inhalt der Variable ab.

RCL

Stellen Sie den alten Namen in Ebene 1 (unter Verwendung des Befehls ROT, für "Rotieren").

ROT

Löschen Sie die alte Variable. Wenn Sie sie vor dem Erzeugen der neuen löschen, vermeiden Sie, daß eine Kopie des Inhalts erzeugt wird.

PURGE

Speichern Sie den Inhalt und den neuen Namen in der richtigen Reihenfolge.

SWAP

Erzeugen Sie die neue Variable.

STO

Sie können nun ein Programm erzeugen, welches diese Schritte automatisch ausführt.

Erzeugen einer Programmvariablen

Nachdem Sie die einzelnen Programmschritte eingetippt haben, sind diese in einer Variablen zu speichern.

Beginnen Sie das Programm mit dem entsprechenden Begrenzungszeichen.

«

Beachten Sie, daß sich die Darstellung des Cursors geändert hat und daß der α Indikator angezeigt wird, womit auf den *Alpha-Eingabemodus* hingewiesen wird. Das Drücken einer Taste für eine programmierbare Operation bewirkt das Schreiben des entsprechenden Namens in die Befehlszeile. Andere Operationen (nicht programmierbare) wie z.B. ◀ zum Löschen eines Zeichens, werden ausgeführt.

Tippen Sie nun die Schritte ein, welche Sie zuvor ausgeführt haben.

OVER RCL
ROT PURGE
SWAP STO
ENTER

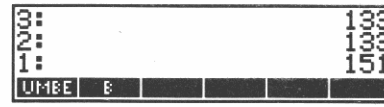
Beachten Sie, daß das schließende Begrenzungszeichen "»" automatisch hinzugefügt wurde.

Speichern Sie das Programm in der Variablen UMBENENNEN.

' UMBENENNEN STO

Überprüfen Sie das USER Menü.

USER



Beachten Sie, daß UMBENENNEN (in abgekürzter Form) im USER Menü erscheint.

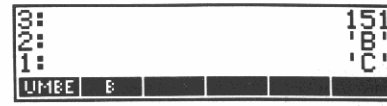
Nun können Sie UMBENENNEN ausführen. Tun Sie dies zuerst unter Verwendung von RCL, danach—auf die allemeinere Weise—über das USER Menü. Der Unterschied zwischen beiden Verfahren hebt die Besonderheiten zwischen Zurückrufen und Auswerten einer Programmvariablen hervor.

Zurückrufen einer Programmvariablen

In diesem Beispiel ist die Variable B in C umzubenennen.

Speichern Sie den alten und neuen Namen im Stack.

' B ENTER
' C ENTER



Rufen Sie das Programm UMBENENNEN auf.

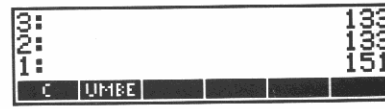
' UMBE RCL

RCL gibt für jede Variable einfach deren Inhalt in den Stack zurück.

Auswerten einer Programmvariablen

Um ein Programm im Stack auszuführen, muß es *explizit* ausgewertet werden.

EVAL

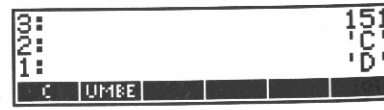


Das USER Menü zeigt, daß B in C umbenannt wurde.

Es war nicht notwendig, das Programm für dessen Ausführung in den Stack zurückzurufen, aber es wurde demonstriert, wie RCL sich für Programme anwenden läßt und wie EVAL die Ausführung bewirkt. Als nächstes lernen Sie einen einfachen Weg zur Ausführung von Programmen kennen.

Speichern Sie C und D als alten und neuen Namen im Stack.

' C ENTER
' D ENTER



Benennen Sie C in D um.

UMBE

Das USER Menü zeigt an, daß C in D umbenannt wurde. Das Programm wurde ausgeführt, indem einfach eine Taste im Benutzermenü gedrückt wurde.

Namen mit und ohne Anführungszeichen

In den obigen Beispielen haben Sie Variablennamen auf zwei verschiedene Weisen benutzt—mit Anführungszeichen und ohne. Von besonderer Bedeutung sind die Zeichen [']: Sie unterscheiden den *Namen* einer Variablen von ihrem *Inhalt*. Nachfolgend finden Sie eine Zusammenstellung der Unterschiede zwischen Namen mit und ohne Anführungszeichen:

- Ein in Anführungszeichen gesetzter Name stellt ein Objekt dar. Es wird in den Stack übernommen und kann als Argument für einen Befehl dienen. So haben Sie z.B. einen Namen in Anführungszeichen als Argument für [STO], [RCL] und ■[PURGE] benutzt, um die Variable VOL zu erzeugen, um ihren Inhalt zu verändern und um sie zu löschen.
- Ein nicht in Anführungszeichen auftretender Name ist wie ein Befehl, die Variable mit dem vorliegenden Namen auszuwerten. Der nicht in Anführungszeichen stehende Name kommt nicht in den Stack—statt dessen wird das in der Variable gespeicherte Objekt entsprechend dem Objekttyp ausgewertet: Numerische Variablen werden in den Stack übernommen und Programme werden ausgewertet. Was mit anderen Variablentypen geschieht, wird an späterer Stelle dieses Handbuchs erläutert.

Wenn Sie einen Namen ohne Anführungszeichen eintippen, welcher nicht mit einer Variablen in Verbindung steht, so wird dieser in Anführungszeichen gestellt und in den Stack übernommen.

4

Wiederholen von Berechnungen

Dieses Kapitel soll Ihnen aufzeigen, wie Sie einen Ausdruck erzeugen, der numerische Variablen enthält und wie Sie ein Leistungsmerkmal des Rechners—den "Löser"—zur Auswertung der Variablen einsetzen können.

Am Beginn von Kapitel 2 führten Sie eine Berechnung aus, indem Sie einen Ausdruck mit *Zahlen* eingegeben und anschließend ausgewertet haben. Sie werden über die nachstehenden Schritte einen Ausdruck im Stack erzeugen, indem für die Berechnung *Namen* als symbolische Argumente verwendet werden. Für die Zuordnung von Variablenwerten und zur Auswertung derselben wird der Löser benutzt. Bei jeder Ausführung wird der momentane Variableninhalt zur Berechnung verwendet. Ändert sich ein Wert Ihrer Variablen, so können Sie die Berechnung einfach wiederholen, um das neue Ergebnis zu ermitteln.

Weiterhin werden Sie in diesem Kapitel mit *Verzeichnissen* (ein Satz von zusammengehörenden Variablen) vertraut gemacht.

Erzeugen eines Ausdrucks

Es wird hier nochmals auf die Berechnung des Gesamtwiderstands (unter "Kettenrechnungen", Kapitel 2) Bezug genommen. Allerdings sollen hier nun anstatt der numerischen Werte Variablen verwendet werden.

$$R_{ges} = R_1 + \frac{1}{\frac{1}{R_2} + \frac{1}{R_3}}$$

Löschen Sie den Stack und wählen Sie das Cursor-Menü.

■ CLEAR

```
4:
3:
2:
1:
```

Wenn ein Menü angezeigt ist, so drücken Sie ◄► zur Anzeige des Cursor-Menüs.

Speichern Sie den Namen `'R1'` im Stack.

' R1 ENTER

```
4:
3:
2:
1:                              'R1'
```

Beachten Sie, daß das schließende ' automatisch hinzugefügt wird. Lassen Sie R1 im Stack, bis er zum Rest der Aufgabe addiert werden kann.

Speichern Sie den Kehrwert von R2 im Stack.

' R2 ENTER ■ 1/x

```
4:
3:
2:                              'R1'
1:                         'INV(R2)'
```

Speichern Sie den Kehrwert von R3 im Stack.

' R3 ENTER ■ 1/x

```
4:
3:                              'R1'
2:                         'INV(R2)'
1:                         'INV(R3)'
```

Addieren Sie die Kehrwerte von R2 und R3.

+

```
4:
3:
2:                              'R1'
1:              'INV(R2)+INV(R3)'
```

Bilden Sie die Kehrwerte der Summe.

■ 1/x

```
3:
2:                              'R1'
1:  'INV(INV(R2)+INV(R3)
    )'
```

Addieren Sie R1 zum Kehrwert der Summe.

+

```
3:
2:
1: 'R1+INV(INV(R2)+INV(
   R3))'
```

Hiermit haben Sie den Wert von R_{ges} ermittelt.

Sie können diesen Ausdruck auch direkt eintippen (mit Klammern, wo erforderlich). Jeder Ausdruck ist äquivalent zu einer Berechnung im Stack und Sie können daher wählen, welches Verfahren für die jeweilige Aufgabenstellung besser geeignet ist.

Erzeugen eines Verzeichnisses

Ein Verzeichnis besteht aus einem Satz von Variablen. Im Moment arbeiten Sie im HOME Verzeichnis—ein internes Verzeichnis, welches auch nach einem MEMORY RESET existiert. Sie werden in diesem Kapitel noch erfahren, wie Sie ein Unterverzeichnis innerhalb von HOME ("Haupt"- bzw. oberstes Verzeichnis) sowie ein weiteres Unterverzeichnis innerhalb von diesem erzeugen.

Hier einige Konzepte über Verzeichnisse, wie Sie in diesem Kapitel zur Anwendung kommen:

- Es kann immer nur ein Verzeichnis als *aktuelles Verzeichnis* dienen; nur dessen Variablen werden im USER Menü angezeigt.
- Enthält Verzeichnis A das Verzeichnis B, so wird A als *Oberverzeichnis* von B bezeichnet, wobei B das *Unterverzeichnis* von A darstellt.
- Wenn Sie im aktuellen Verzeichnis beginnen und zum zugehörigen Oberverzeichnis wechseln und danach zu dessen Oberverzeichnis, usw., so gelangen Sie immer zum HOME Verzeichnis. Die Reihenfolge der durchzulaufenden Verzeichnisse (von HOME zum jeweiligen Unterverzeichnis) wird als *aktueller Pfad* bezeichnet.

Sie können den aktuellen Pfad durch Ausführen von PATH überprüfen.

Wählen Sie das MEMORY Menü.

MEMORY

```
2:
1: 'R1+INV(INV(R2)+INV(
   R3))'
MEM MENU ORDER PATH HOME CRDIR
```

Überprüfen Sie den aktuellen Pfad.

PATH

Die von PATH zurückgegebene Liste beginnt immer mit HOME und endet mit dem aktuellen Verzeichnis. HOME ist immer der Ausgangspunkt für jeden Pfad und, da Sie seither noch kein anderes Verzeichnis erzeugt haben, hier auch das aktuelle Verzeichnis.

Um z.B. alle Ihre elektrotechnischen Aufgaben/Lösungen zusammenzufassen, erzeugen Sie das Unterverzeichnis "EE".

' EE CRDIR

Wechseln Sie zum EE Verzeichnis.

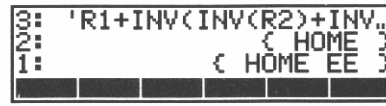
EE ENTER

Überprüfen Sie erneut den aktuellen Pfad.

PATH

EE wird nun als aktuelles Verzeichnis angezeigt. Um eine Auswirkung des Wechsels zum EE Verzeichnis anzudeuten, sollten Sie nun das USER Menü anzeigen.

USER



Beachten Sie, daß das Programm UMBENENNEN (aus dem letzten Kapitel) nicht mehr erscheint. Es werden nur die Variablen des *aktuellen Verzeichnisses* (EE) angezeigt; UMBENENNEN ist im HOME Verzeichnis.

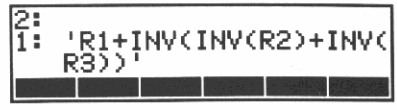
Allerdings können Sie immer noch UMBENENNEN ausführen, da jede Variable, deren Verzeichnis im *aktuellen Pfad* (HOME EE) vorkommt, durch den Variablennamen gefunden wird.

Dies ist einer der Vorteile von Verzeichnissen: *Wenn Sie allgemeine "Dienstprogramme" im HOME Verzeichnis speichern, dann können Sie diese immer ausführen, sie blähen jedoch nicht das USER Menü auf.*

Sie können nun im neuen Verzeichnis EE arbeiten.

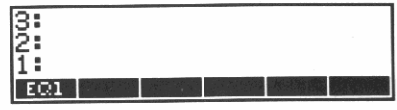
Löschen Sie die zwei Pfadlisten im Stack.

DROP DROP



Speichern Sie den Ausdruck in der Variablen EQ1 (*Gleichung 1*). Sie werden später den Grund dafür erkennen.

' EQ1 STO

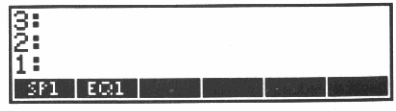


Die Variable EQ1 erscheint nun im USER Menü.

Nehmen Sie an, dieser Ausdruck wird für eine Vielzahl von Problemstellungen gebraucht, die Sie jeweils getrennt behandeln möchten. Sie können dazu die Werte für jedes Problem in einem besonderen Unterverzeichnis speichern.

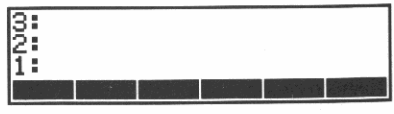
Erzeugen Sie das Unterverzeichnis SP1 (*Seriell-Parallel 1*) für das erste Problem.

' SP1 ENTER
CRDIR ENTER



Der Name des neuen Unterverzeichnisses erscheint im USER Menü. Drücken Sie die entsprechende Menütaste, um zu SP1 zu wechseln.

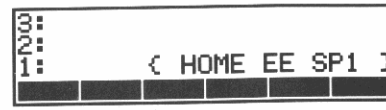
SP1



Das USER Menü ist erneut leer, da das aktuelle Verzeichnis (SP1) leer ist.

Überprüfen Sie den aktuellen Pfad.

PATH [ENTER]



Sie können jede Variable im HOME oder EE Verzeichnis über ihren Namen auffinden, da die Verzeichnisse im aktuellen Pfad liegen (HOME EE SP1); das USER Menü zeigt jedoch nur die Variablen des aktuellen Verzeichnisses (SP1) an.

Sie können nun im Löser den Ausdruck EQ1 verwenden.

Wiederholen einer Berechnung über den Löser

Prinzipiell sind drei Schritte bei der Anwendung des Lösers in Zusammenhang mit einem Ausdruck zu beachten.

1. Speichern Sie den Ausdruck (oder dessen Namen) in der Variablen "EQ" (*EQuation bzw. Gleichung*). Der Löser aktzeptiert nur diesen Namen.
2. Verwenden Sie das Löser-Menü, um den Variablen Zahlenwerte zuzuordnen.
3. Werten Sie über den Löser den Ausdruck aus.
4. Wiederholen Sie die Schritte 2 und 3 für andere Werte.

Nachfolgend die Schritte für das vorliegende Beispiel:

Schritt 1: Speichern Sie den Namen EQ1 in der Variablen EQ.

Dies mag Sie verwundern—warum den Namen in einer Variablen speichern? Warum nicht die Gleichung selbst in EQ speichern? Zum einen ist EQ1 leichter als der gesamte Ausdruck einzugeben, zum anderen werden Sie später feststellen, daß dadurch zwischen verschiedenen Gleichungen leichter gewechselt werdem kann.

Speichern Sie den Namen EQ1 im Stack.

['] EQ1 [ENTER]

Wenn Sie ['] vergessen haben, erhielten Sie den ganzen Ausdruck im Stack; drücken Sie in diesem Fall [DROP] und beginnen Sie erneut.

Wählen Sie das SOLVE Menü.

SOLV

Verwenden Sie STEQ (*STore Equation*), um den Namen EQ1 in der Variablen EQ zu speichern.

STEQ

Schritt 2: Weisen Sie den Variablen Werte zu.

Anzeigen des Löser-Menüs.

SOLVR

Die Variablen der momentanen Gleichung erscheinen im Löser-Menü. (Wenn die Gleichung mehr als sechs Variable enthält, kann mit NEXT die nächste Menüzeile mit Variablen angezeigt werden.)

Dieses Menü unterscheidet sich vom USER Menü: Das Löser-Menü *speichert Werte* in den Variablen, anstatt diese auszuwerten.

Sie können nun den Variablen R1, R2 und R3 Werte zuweisen. Speichern Sie zuerst die Zahl 8 in der Variablen R1.

8 R1

Drücken von R1 ist gleichwertig mit dem Speichern von `'R1'` im Stack und dem Drücken von STO. Beachten Sie, daß die oberste Anzeigezeile den Variablennamen und -inhalt enthält.

Speichern Sie 6 in der Variablen R2.

6 R2

Speichern Sie 3 in der Variablen R3.

3 R3

Schritt 3: Werten Sie den Ausdruck aus.

Das Menüfeld EXPR= bedeutet "Ausdruck gleich" (*EXPRession equals*)—Drücken der Taste wertet den Ausdruck aus.

EXPR=

Der Wert des Ausdrucks (10) wird in Ebene 1 zurückgegeben und erscheint in inverser Darstellung in der oberen Anzeigezeile.

Schritt 4: Wiederholen Sie die Schritte 2 und 3 für andere Variableninhalte. Was ergibt sich zum Beispiel, wenn R3 den Wert 12 annimmt?

Speichern Sie 12 in der Variablen R3.

12 R3

Werten Sie den Ausdruck für die momentanen Variableninhalte aus.

EXPR=

Der neue Wert (12) wird in Ebene 1 zurückgegeben und erscheint in inverser Darstellung in der oberen Anzeigezeile.

Verwenden von neuen Werten

Nehmen Sie an, Sie möchten eine Aufgabe bearbeiten (mit anderen Werten für R1, R2 und R3), und wollen später mit den momentanen Werten weiterarbeiten. Sie könnten die Werte jedesmal neu eingeben—es gibt aber auch einen einfacheren Weg:

1. Erzeugen Sie ein neues Verzeichnis für die neuen Werte.
2. Definieren Sie den gleichen Ausdruck als EQ.
3. Verwenden Sie den Löser wie zuvor, um neue Variablenwerte zuzuweisen, und werten Sie den Ausdruck aus.

Hier wird ein weiterer Vorteil von Verzeichnissen deutlich: *Innerhalb eines Verzeichnisses kann nur eine Variable mit einem bestimmten Namen vorkommen; der gleiche Variablenname kann jedoch in einer beliebigen Anzahl von Verzeichnissen vorkommen.*

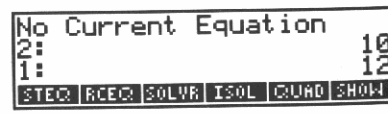
Schritt 1: Erzeugen Sie ein neues Verzeichnis.

Da das neue Verzeichnis alternativ zu SP1 ist, benennen Sie es mit SP2 und erzeugen es im gleichen Oberverzeichnis, EE. Dies stellt die erste Verzweigung innerhalb Ihrer Verzeichnisstruktur dar—zwei Unterverzeichnisse (SP1 und SP2 innerhalb des gleichen Oberverzeichnisses EE).

Um ein neues Unterverzeichnis in EE zu erzeugen, müssen Sie EE als aktuelles Verzeichnis wählen (sonst wäre es innerhalb von SP1).

Wechseln Sie zum Verzeichnis EE.

EE ENTER



Der Rechner gibt ein Tonsignal aus und zeigt `No Current Equation` sowie das Löser-Menü an. Die Meldung weist darauf hin, daß momentan keine Gleichung als 'EQ' im EE Verzeichnis definiert ist.

Erzeugen Sie das Verzeichnis SP2.

' SP2
MEMORY CRDIR

Wechseln Sie zum Verzeichnis SP2.

SP2 [ENTER]

Überprüfen Sie den aktuellen Pfad.

PATH

HOME und EE befinden sich im aktuellen Pfad (wie unter SP1 als aktuelles Verzeichnis), aber SP1 ist nicht enthalten. Als Resultat hieraus können Sie noch immer auf die Variablen in HOME (UMBENENNEN) und EE (EQ1) zugreifen—jedoch nicht auf die Variablen in SP1 (EQ, R1, R2 und R3); Sie können nun neue Variablen R1, R2 und R3 erzeugen.

Schritt 2: Definieren Sie den gleichen Ausdruck als EQ.

Wie zuvor ist STEQ zum Speichern von EQ1 in der Variablen EQ zu verwenden.

['] EQ1 [SOLV] STEQ

Schritt 3: Verwenden Sie den Löser wie zuvor, um Werte zuzuweisen und den Ausdruck auszuwerten. Verwenden Sie hier

$$R1 = 11, \quad R2 = 21, \quad R3 = 7$$

Wählen Sie das Löser-Menü.

SOLVR

Weisen Sie die neuen Variableninhalte zu.

11 [R1]
21 [R2]
7 [R3]

Werten Sie den Ausdruck aus.

EXPR=

Um zur vorherigen Aufgabe zurückzukehren, wäre EE auszuführen (um zum entsprechenden Verzeichnis zu kommen), SP1 auszuführen (zum Wechseln zum SP1 Verzeichnis), und SOLV SOLVR zu drücken (um das Löser-Menü zu aktivieren); die Variableninhalte in SP1 wären immer noch die alten.

Verwenden eines anderen Ausdrucks

Da Sie nun unterschiedliche Variableninhalte für die Anwendung von EQ1 haben, sollten Sie als zweiten Ausdruck EQ2 erzeugen, welchen Sie mit jedem der beiden Variablensätze verwenden können. Dies erfolgt über zwei grundsätzliche Schritte:

1. Wechseln Sie zum Verzeichnis EE, erzeugen Sie den neuen Ausdruck und speichern Sie diesen in der Variablen EQ2.
2. Wechseln Sie zum Verzeichnis SP1 oder SP2, ändern Sie den Inhalt von EQ von 'EQ1' nach 'EQ2', und verwenden Sie den Löser zur Auswertung des Ausdrucks.

Schritt 1: Wechsel zum Verzeichnis EE, Erzeugen des neuen Ausdrucks und Speichern desselben in der Variablen EQ2.

Wechseln Sie zum Verzeichnis EE.

EE ENTER

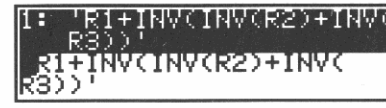
Erzeugen Sie den neuen Ausdruck. In diesem Beispiel wird EQ2 aus einer modifizierten Version des Ausdrucks EQ1 bestehen.

Rufen Sie den in EQ1 gespeicherten Ausdruck in den Stack zurück.

USER EQ1

Übernehmen Sie den Ausdruck in die Befehlszeile.

■ EDIT



Der Ausdruck in Ebene 1 erscheint in inverser Darstellung, um Sie zu warnen, daß er durch den Inhalt der Befehlszeile ersetzt wird. Außerdem wird der α Indikator als Hinweis für den Alpha-Eingabemodus angezeigt.

Edieren Sie den Ausdruck, um folgende Gleichung darzustellen:

$$R_{ges} = R_1 + \frac{1}{\frac{1}{R_2} + \frac{1}{R_3} + \frac{1}{R_3}}$$

Stellen Sie den Cursor in die unterste Anzeigezeile (die Operationen zum Verschieben des Cursors sind über das *Cursor-Menü*—die weißen Bezeichnungen oberhalb den Menütasten—zugänglich).

▼

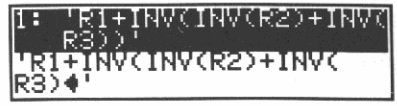
Das Cursor-Menü ist immer aktiv, wenn die Befehlszeile ohne ein anderes Menü angezeigt wird. Sie können das Cursor-Menü durch Drücken von ◄► ein- und ausschalten. ■ EDIT aktiviert automatisch das Cursor-Menü.

Stellen Sie den Cursor genau hinter den Term von R3.

▶ ▶ ▶

Wählen Sie den Einfügungsmodus.

INS



Die Form des Cursors wechselt in einen Pfeil, zur Kennzeichnung, daß Text links vom Cursor eingefügt wird. (Erneutes Drücken von INS schaltet wieder auf den Ersetzungsmodus um, unter welchem das alte Zeichen an der Cursorposition durch ein neues ersetzt wird.)

Tippen Sie den zweiten Term für R3 ein.

+ ■ 1/x (R3

Ersetzten Sie den Ausdruck in Ebene 1 durch die edierte Version in der Befehlszeile.

ENTER

Speichern Sie den neuen Ausdruck in einer Variablen EQ2.

' EQ2 STO

Schritt 2: Wechseln Sie zum Verzeichnis SP1 oder SP2 und ändern Sie die Definition für EQ von EQ1 nach EQ2; werten Sie anschließend über den Löser den Ausdruck aus.

In diesem Beispiel sollen die Werte in SP1 für den neuen Ausdruck verwendet werden.

Wechsel zum Verzeichnis SP1.

SP1

Ändern der Definition für EQ von EQ1 nach EQ2.

' EQ2 SOLV STEQ

Werten Sie den Ausdruck EQ2 mit den Variableninhalten von SP1 aus.

SOLVR EXPR=

Um EQ2 mit den Inhalten von SP2 auszuwerten, könnten Sie EE ausführen (zur Rückkehr zum Verzeichnis EE) und danach obigen Schritt 2 durchführen, unter Substitution von SP2 für SP1.

Rückkehr zum HOME Verzeichnis

Hätten Sie Ihre Aufgaben im EE Bereich nun abgeschlossen, so könnten Sie zum HOME Verzeichnis zurückkehren. Da HOME ein internes Verzeichnis darstellt, ist sein Name im MEMORY Menü enthalten.

Wechseln zum HOME Verzeichnis.

MEMORY HOME

Überprüfen Sie das USER Menü.

USER

Das Menüfeld EE ist das einzige Anzeichen für die von Ihnen in diesem Kapitel erzeugten Daten: EQ1, EQ2, die Unterverzeichnisse SP1 und SP2 und alle darin enthaltenen Variablen. Dies stellt einen weiteren wesentlichen Vorteil von Verzeichnissen dar: *Vom Oberverzeichnis aus betrachtet wird ein ganzes Verzeichnis—seine Variablen und die zugehörigen Unterverzeichnisse—schlicht durch den Verzeichnisnamen dargestellt.*

Zusammenfassung

Nachstehend die generelle Strategie, welcher Sie in diesem Kapitel gefolgt sind.

- Erzeugen Sie ein Verzeichnis für jeden Satz von Problemstellungen, welche in einer bestimmten Beziehung zueinander stehen.
- Speichern Sie jeden zur Lösung des Problems erforderlichen Ausdruck in einer Variablen.
- Erzeugen Sie ein Unterverzeichnis für die spezifischen Werte innerhalb jeder Aufgabenstellung.
- Verwenden Sie den Löser mit einer beliebigen Kombination von Ausdruck und Variableninhalten.

5

Funktionen für reelle Zahlen

In diesem Kapitel werden die Menüs TRIG, LOGS und REAL vorgestellt. Das TRIG Menü enthält trigonometrische Funktionen und Befehle, die sich auf Winkelmaße beziehen. Das LOGS Menü faßt logarithmische, exponentielle und hyperbolische Funktionen zusammen. Das REAL Menü enthält weitere Befehle für reelle Zahlen.

Alle Befehle in diesen Menüs sind kurz in Anhang B, "Menütabellen", beschrieben. Eine vollständige Beschreibung finden Sie unter "TRIG", "LOGS" oder "REAL" im Referenzhandbuch.

Trigonometrische Funktionen

Dieser Abschnitt zeigt Ihnen, wie Sie den Winkelmodus spezifizieren, Berechnungen mit π durchführen und wie Sie Winkeleinheiten konvertieren können.

Wählen des Winkelmodus

Der Rechner kann Argumente und Ergebnisse für Winkel in Grad ($1/360$ eines Kreises) oder im Bogenmaß (Radiant, $1/2\pi$ eines Kreises) interpretieren. Die Voreinstellung für den Winkelmodus beträgt Grad. Um die Beispiele in diesem Abschnitt durchzuführen, müssen Sie das Bogenmaß als Winkelmodus spezifizieren.

Löschen Sie den Stackinhalt und wählen Sie das MODE Menü.

■ CLEAR ■ MODE

Die zwei rechten Menüfelder, DEG (*DEGgrees*) und RAD (*RADians*), stellen die Auswahl für den Winkelmodus dar. Beachten Sie, daß DEG mit einem kleinen Quadrat angezeigt wird, um die momentane Einstellung für den Winkelmodus zu kennzeichnen.

Spezifizieren Sie das Bogenmaß als Winkelmodus.

RAD

Der Bogenmaß-Indikator (2π) wird angezeigt und die Belegung der Menüfelder ändert sich. (Die meisten Abbildungen in diesem Handbuch enthalten keine Indikatoren. Um den Indikator (2π) aufzufinden, sollten Sie auf Seite 27 nachsehen.)

Anzeige der ersten Menüzeile des TRIG Menüs.

TRIG

Bei diesen Funktionen handelt es sich um einwertige Funktionen, d.h. sie beziehen sich nur auf eine Zahl (in Ebene 1). Bei reellen Zahlen steuert der Winkelmodus, wie SIN, COS und TAN ihre Argumente interpretieren, und wie ASIN (*Arcussinus*), ACOS (*Arcuscosinus*) und ATAN (*Arcustangens*) ihre Ergebnisse ausdrücken.

Sie werden die SIN Funktion bei der nachfolgenden Diskussion von π verwenden.

Verwenden von π

Die transzendente Zahl π kann im HP-28S nicht exakt durch eine reelle Zahl dargestellt werden. Im allgemeinen erzielt man jedoch hinreichend genaue Ergebnisse durch die 12-stellige Näherung 3.14159265359.

Der HP-28S bietet außerdem eine symbolische Konstante π an, welche π exakt repräsentiert. Die Funktionen SIN, COS und TAN erkennen im Bogenmaß-Modus die symbolische Konstante π und liefern ein genaues Ergebnis. Die Funktionen SIN und COS erkennen auch den Ausdruck π/2.

Bei anderen Funktionen erzeugt die Konstante π einen Ausdruck, der π enthält. Wenn Sie ein numerisches Resultat erzwingen, verwendet der Rechner die 12-stellige Näherung.

Um den Unterschied zwischen 3.14159265359 und π zu demonstrieren, ist der Sinus von beiden zu berechnen.

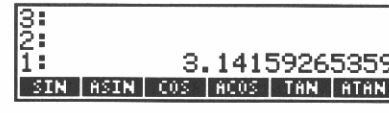
Speichern Sie '**π**' in Ebene 1.

[→] [π] [ENTER]

Obwohl dieses Objekt wie ein Name aussieht, ist es eigentlich ein Ausdruck mit einem einzigen Term, der symbolischen Konstante π.

Erzwingen Sie ein reellwertiges Ergebnis, indem Sie →NUM (*to NUMber*) verwenden.

[→] [→NUM]



Es wird die 12-stellige Näherung an π (3.14159265359) in Ebene 1 zurückgegeben.

Berechnen Sie den Sinus von der Näherung an π.

[SIN]

Das Ergebnis ($-2.06761537357 \times 10^{-13}$) ist nicht genau 0, da das Argument (3.14159265359) nicht genau π darstellte.

Berechnen Sie nun den Sinus von π.

[→] [π] [SIN]

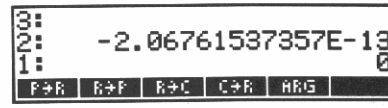
Die SIN Funktion erkennt die symbolische Konstante π und gibt das exakte Ergebnis (0) zurück.

Konvertieren zwischen Winkeleinheiten

Das TRIG Menü enthält Befehle, mit welchen die Umrechnung von Winkeln zwischen verschiedenen Einheiten durchgeführt werden kann. Die Befehle befinden sich in der dritten Zeile des TRIG Menüs. Beachten Sie kurz die Menüfelder der zweiten Zeile, bevor Sie zur dritten Menüzeile übergehen.

Anzeige der nächsten Zeile des TRIG Menüs.

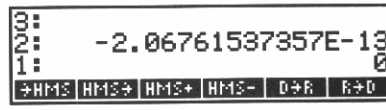
NEXT



Diese Befehle beschäftigen sich mit komplexen Zahlen und sind ebenfalls im COMPLEX Menü enthalten. Komplexe Zahlen werden im nächsten Kapitel behandelt.

Anzeige der dritten Zeile des TRIG Menüs.

NEXT



Über die Befehle HMS→ und D→R werden Sie eine Winkelgröße, die in Grad, Minuten und Sekunden ausgedrückt ist, in einen Winkel, der im Bogenmaß dargestellt ist, konvertieren.

Die vier HMS (*Hours-Minutes-Seconds*) Befehle erlauben Ihnen die Berechnungen mit Zahlen, deren dezimaler Anteil in Minuten und Sekunden ausgedrückt ist. Diese Zahlen müssen in einem speziellen Format—HMS-Format genannt—vorliegen:

h.MMSSs

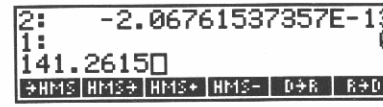
Hierbei stellt *h* (*Hours*) die Stunden (oder Grad) dar, *MM* repräsentiert die Minuten, *SS* stellt die Sekunden dar und *s* repräsentiert den dezimalen Anteil der Sekunden. *MM* und *SS* sind zweistellige Werte, während *h* und *s* jeweils für eine beliebige Stellenanzahl stehen.

Die Befehle →HMS (*dezimal-in-HMS*) und HMS→ (*HMS-in-dezimal*) konvertieren eine reelle Zahl zwischen dem normalen dezimalen Format in das spezielle HMS-Format. Die Befehle HMS+ (*HMS plus*) und HMS− (*HMS minus*) addieren bzw. subtrahieren Zahlen im HMS-Format, wobei das Ergebnis ebenso dargestellt wird.

Konvertieren Sie z.B. 141° 26′ 15″ in das dezimale Format.

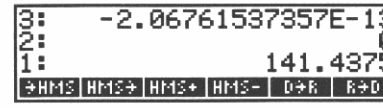
Geben Sie den Winkel im HMS Format ein.

141.2615



Konvertieren Sie den Winkel vom HMS Format in Dezimalgrad.

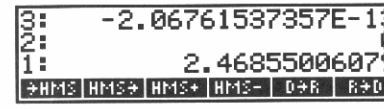
HMS→



Die anderen zwei Funktionen dieser Menüzeile, D→R (*Degrees-in-Radians*) und R→D (*Radians-in-Degrees*) konvertieren eine reelle Zahl von der Darstellung in Grad nach Bogenmaß oder umgekehrt.

Konvertieren Sie die Zahl in Ebene 1 von Grad in Bogenmaß-Darstellung.

D→R



Unter Zusammenfassung aller Schritte haben Sie nun berechnet:

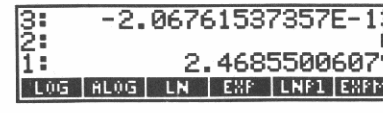
$$141° \; 26′ \; 15″ = 141.4375° = 2.46855006079 \text{ Radianten}$$

Logarithmische, exponentielle und hyperbolische Funktionen

Das LOGS Menü faßt logarithmische und exponentielle Funktionen (natürliche und dekadische) sowie hyperbolische Funktionen zusammen. Eine detaillierte Beschreibung dieser Funktionen finden Sie unter "LOGS" im Referenzhandbuch.

Anzeige der ersten Zeile im LOGS Menü.

■ LOGS



Die Funktionen LOG (*dekadischer LOGarithmus*) und ALOG (*dekadischer AntiLOGarithmus*) berechnen den Logarithmus und die Exponentialfunktion zur Basis 10. Die Funktionen LN (*Logarithmus Naturalis*) und EXP (*natürliche EXPonentialfunktion*) berechnen den Logarithmus und die Exponentialfunktion zur Basis e. (e ist eine transzendente Zahl mit dem Näherungswert 2.71828182846.)

Die Funktion LNP1 (*LN Plus 1*) berechnet für ein Argument x den Ausdruck ln $(x + 1)$, und die Funktion EXPM (*EXP Minus 1*) berechnet (exp x) $- 1$. Dies ist bei Argumenten nahe 0 von Bedeutung, da diese Funktionen eine bessere Genauigkeit liefern als die jeweilige Folge von Einzelfunktionen (ein Beispiel hierzu finden Sie unter "Finanzmathematische Berechnungen" auf Seite 103).

Anzeige der zweiten Menüzeile des LOGS Menüs.

NEXT

Hier sind die hyperbolischen Funktionen und deren Umkehrungen: SINH (*SINus Hyperbolicus*) und ASINH (*inverser SINus Hyperbolicus*), COSH (*COSinus Hyperbolicus*) und ACOSH (*inverser COSinus Hyperbolicus*), TANH (*TANgens Hyperbolicus*) und ATANH (*inverser TANgens Hyperbolicus*). Die Funktionen sind von der natürlichen Exponentialfunktion e^x abgeleitet und erwarten ihr Argument in Ebene 1.

Andere reellwertige Funktionen

Im REAL Menü sind Funktionen enthalten, die sich im wesentlichen auf reelle Zahlen beziehen.

Wählen Sie das REAL Menü.

■ REAL

Die Funktion NEG (*NEGieren*) gibt für x den Wert $-x$ zurück. FACT (*FACTorial*) gibt $n!$ für eine positive ganze Zahl n zurück; für eine gebrochene Zahl x wird die Gamma-Funktion $\Gamma(x + 1)$ angewendet. RAND (*RANDom number*) ergibt eine Zufallszahl, die sich über einen von RDZ (*RanDomiZe*) spezifizierten Startwert berechnet.

Die Funktionen MAXR (*MAXimum Real*) und MINR (*MINimum Real*) geben die symbolischen Konstanten für die größte und kleinste positive reelle Zahl, die im HP-28S darstellbar ist, zurück. (Hinweise zum Erhalten eines numerischen Werts für eine symbolische Konstante finden Sie unter "Verwenden von π" auf Seite 74.)

Der folgende Abschnitt zeigt Ihnen die Anwendung der Funktion NEG. Zur einfacheren Handhabung können Sie NEG durch Drücken von [CHS] (*CHange Sign*) ausführen, wenn keine Befehlszeile vorhanden ist. Um den Befehl NEG in die Befehlszeile einzugeben—z.B. beim Eintippen eines Programms—drücken Sie NEG oder [N] [E] [G].

Negieren Sie nun die Zahl in Ebene 1 zweimal, zuerst durch Drücken von [CHS] und anschließend durch Drücken von NEG.

Negieren der Zahl in Ebene 1.

[CHS]

Negieren Sie die Zahl erneut.

NEG

Definieren neuer Funktionen

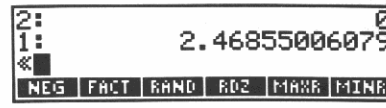
Sie können Programmvariablen erzeugen, welche sich wie interne Funktionen verwenden lassen—sogar in Ausdrücken. Solche Programmvariablen, als *Benutzerfunktionen* bezeichnet, müssen zwei Anforderungen erfüllen:

- Sie müssen explizit ihre Argumente definieren.
- Sie müssen genau ein Ergebnis zurückgeben.

Sie können z.B. eine Funktion COT für den Cotangens definieren, wobei $\cot x = 1/\tan x$.

Beginn des Programms.

«



Festlegen des Arguments.

■ → LC x LC

Der Rechtspfeil legt fest, daß der folgende Name eine *lokale Variable* darstellt, welche nur innerhalb dieses Programms vorkommt.

Es ist sinnvoll, zur Unterscheidung von lokalen und "globalen" Variablen eine bestimmte Namenskonvention zu beachten. In diesem Handbuch werden kleingeschriebene Buchstaben zur Kennzeichnung von lokalen Variablen benutzt. (Das Drücken von LC bewirkt das Umschalten auf Kleinschreibung; erneutes Drücken von LC schaltet wieder auf Großschreibung um.)

Definieren Sie die Funktion.

' ■ 1/x (TAN (LC x

Eingabe des Programms.

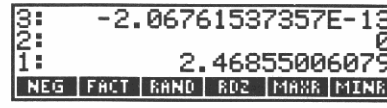
ENTER

Die schließende Klammer und das Begrenzungszeichen werden automatisch hinzugefügt.

Dieses Programm bedeutet: Nimm ein Argument vom Stack (in UPN Syntax) oder vom Ausdruck (in algebraischer Syntax) und benenne es als x; werte danach den Ausdruck $1/\tan x$ unter Verwendung der lokalen Definition von x aus.

Speichern Sie das Programm in einer Variablen COT.

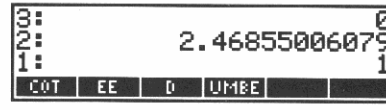
' COT STO



Sie können nun COT entweder in UPN oder in algebraischer Syntax benutzen, genau wie die internen trigonometrischen Funktionen.

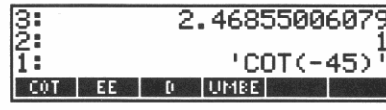
Berechnen Sie cot 45° unter Anwendung von UPN.

■ MODE DEG
45 USER COT



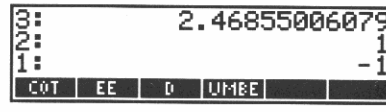
Berechnen Sie cot −45° unter Anwendung algebraischer Syntax.

' COT (−45 ENTER



Werten Sie den Ausdruck aus.

EVAL



6

Funktionen für komplexe Zahlen

Der HP-28S schließt einen Objekttyp mit ein, der komplexe Zahlen darstellt. So wird z.B. die komplexe Zahl $z = 3 + 4i$ durch das Objekt `(3,4)` repräsentiert. Da jede komplexe Zahl aus einem einzelnen Objekt besteht, können Sie mit komplexen Zahlen genauso einfach Berechnungen durchführen, wie mit reellen Zahlen.

Das Paar reeller Zahlen in einer komplexen Zahl können die Koordinaten eines Punktes in der Ebene darstellen. Der HP-28S verwendet z.B. komplexe Zahlen zur Darstellung von Abbildungskoordinaten. Im nachfolgenden Abschnitt werden zunächst zwei Koordinatensysteme beschrieben—Rechteckskoordinaten und Polarkoordinaten, und anschließend wird aufgezeigt, wie Sie Konvertierungen zwischen diesen Systemen ausführen können.

Verwenden von komplexen Zahlen

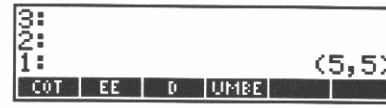
Die meisten reellwertigen Funktionen bearbeiten auch komplexe Zahlen. So können Sie z.B. arithmetische Berechnungen mit komplexen Zahlen genauso durchführen, wie Sie es von reellen Zahlen gewöhnt sind—geben Sie die Zahlen in den Stack ein und führen Sie die Funktion aus. Als Beispiel:

$$((9 + 2i) + (-4 + 3i)) \times (6 + i)$$

Löschen Sie den Stackinhalt und geben Sie $9 + 2i$ ein.

Addieren Sie $-4 + 3i$. (Drücken Sie [4] [CHS], um -4 einzugeben.)

[(] −4 [,] 3 [+]



Multiplizieren Sie mit $6 + i$.

[(] 6 [,] 1 [×]

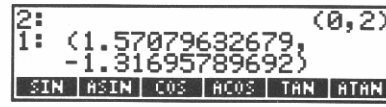
Manchmal kann ein reellwertiges Argument zu einem komplexen Ergebnis führen.

Berechnen Sie $\sqrt{-4}$.

−4 [■] [√x]

Berechnen Sie arcsin 2.

2 [TRIG] [ASIN]



Speziell auf komplexe Zahlen ausgerichtete Funktionen finden Sie im COMPLEX Menü.

Wählen Sie das COMPLEX Menü.

[■] [COMPLX]

Alle Befehle im COMPLEX-Menü sind kurz in Anhang B, "Menütabellen", dargestellt. Eine vollständige Beschreibung finden Sie unter "COMPLEX" im Referenzhandbuch.

- R→C (*Real-in-Complex*) konvertiert zwei reelle Zahlen x und y in eine komplexe Zahl (x, y).
- C→R (*Complex-in-Real*) konvertiert eine komplexe Zahl (x, y) in zwei reelle Zahlen x und y.
- RE (*REal part*) gibt den reellen Teil x von einem komplexen Argument (x, y) zurück.

- IM (*IMaginary part*) gibt den Imaginärteil y von einem komplexen Argument (x, y) zurück.
- CONJ (*CONJugate*) ergibt $(x, -y)$ für ein komplexes Argument (x, y).
- SIGN gibt den Einheitsvektor $(x/\sqrt{x^2 + y^2}, y/\sqrt{x^2 + y^2})$ für ein komplexes Argument (x, y) zurück.

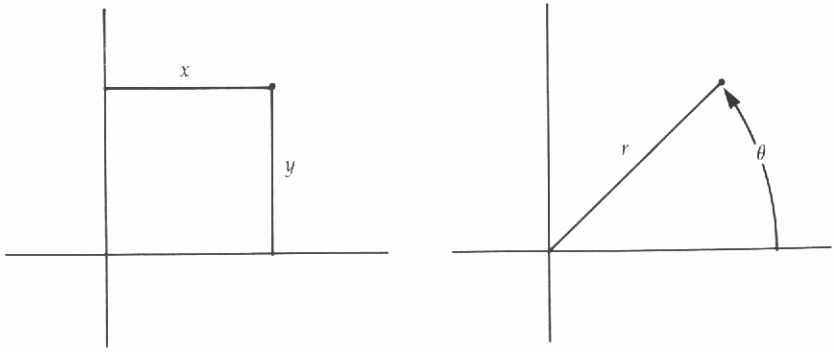
Anzeige der nächsten Zeile des COMPLEX Menüs.

NEXT

Diese Funktionen (außer NEG) beziehen sich auf komplexe Zahlen in Polarkoordinaten.

Verwenden von Polarkoordinaten

Ein Punkt in einer Ebene kann durch zwei verschiedene Koordinatensysteme bestimmt werden. Die nachstehende Abbildung zeigt einen Punkt, der in Rechtecksnotation (x, y) und in Polarnotation (r, θ) dargestellt ist.



- R→P (*Rechtecks-in-Polarnotation*) konvertiert eine komplexe Zahl in Rechtecksnotation (x, y) in Polarnotation (r, θ).
- P→R (*Polar-in-Rechtecksnotation*) konvertiert eine komplexe Zahl in Polarnotation (r, θ) in Rechtecksnotation (x, y).
- ABS (*Absolutwert*) berechnet r für ein komplexes Argument (x, y).
- NEG gibt $(-x, -y)$ für ein komplexes Argument (x, y) zurück.
- ARG gibt θ für ein komplexes Argument (x, y) zurück.

Beachten Sie, daß nur P→R eine komplexe Zahl als Polarkoordinaten interpretiert; alle anderen Funktionen—arithmetische, trigonometrische, logarithmische, usw.—interpretieren eine komplexe Zahl als Rechteckskoordinaten. Prägen Sie sich diese entscheidende Regel ein: *Jede komplexe Zahl in Polarnotation muß zuerst in Rechteckskoordinaten umgewandelt werden, bevor sie in einer Berechnung verwendet werden kann.*

Als Beispiel hierfür soll die resultierende Strecke und der Kurs berechnet werden, welcher sich nach dem Zurücklegen einer Strecke von 2 km bei einem Kurs von 36° und einer anschießenden Strecke von 3 km bei einem Kurs von 65° ergibt (auf 2 Dezimalstellen genau).

Wählen Sie Grad als Winkelmodus und FIX 2 Anzeigeformat.

[MODE] DEG 2 FIX

Geben Sie die ersten Koordinaten ein.

(2 , 36

Konvertieren Sie diese in Rechteckskoordinaten.

[COMPLX] [NEXT] P→R

Geben Sie das zweite Koordinatenpaar ein.

(3 , 65

Konvertierung in Rechteckskoordinaten.

P→R

Addieren Sie die Rechteckskoordinaten.

+

Zurückkonvertieren in Polarkoordinaten.

R→P

Die resultierende Strecke beträgt 4.85 km bei einem Kurs von 53.46°.

Benutzerfunktion für polare Addition

Nachstehend ein einfaches Programm PSUM (*Polare Summe*) zur Automatisierung der vorangehenden Schritte.

Beginn des Programms.

«

Festlegen der Argumente. (Verwenden Sie ein Leerzeichen zum Trennen der Argumente.)

■ → LC x SPACE y

Der Rechtspfeil zeigt an, daß die folgenden Namen *lokale Variablen* darstellen, welche nur in diesem Programm vorkommen.

Definieren Sie die Funktion.

' R→P (P→R (x ▮) +
P→R (y ENTER

Die schließende Klammer sowie Begrenzungszeichen werden automatisch für Sie hinzugefügt.

Dieses Programm bedeutet: Nimm zwei Argumente vom Stack (in UPN Syntax) oder aus einem Ausdruck (in algebraischer Syntax) und benenne diese x und y; berechne danach die Polarkoordinaten der Summe der Rechteckskoordinaten von x und y.

Speichern Sie das Programm in der Variablen PSUM.

' PSUM STO

Verwenden Sie nun PSUM zur Wiederholung der vorangehenden Berechnung, zuerst in UPN Syntax, danach in algebraischer Syntax.

Geben Sie die ersten Koordinaten ein.

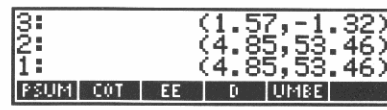
(2 , 36 ENTER

Eingabe der zweiten Koordinaten.

(3 , 65

Ausführen von PSUM.

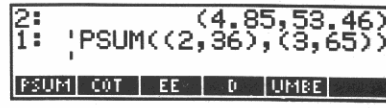
USER PSUM



Das Ergebnis stimmt mit dem vorhergehenden Ergebnis überein.

Versuchen Sie nun die Lösung über algebraische Syntax.

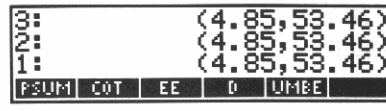
' PSUM ((2 , 36) , (3 , 65) ENTER



Die äußeren Klammern und das mittlere Komma definieren die Argumente für PSUM; das andere Klammernpaar und Komma sind Teil der Syntax für komplexe Zahlen. Denken Sie daran, daß Sie *zwei* Klammernpaare benötigen, wenn Sie eine komplexe Zahl als Argument bei algebraischer Syntax verwenden.

Werten Sie den Ausdruck aus.

EVAL



7

Grafische Darstellung

Dieses Kapitel macht Sie mit dem Abbilden eines Ausdrucks auf dem HP-28S vertraut. Die grafische Darstellung gibt Ihnen eine visuelle Vorstellung über das Verhalten eines Ausdrucks. Weiterhin wird Ihnen dadurch wesentlich erleichtert, erste Näherungen für die Nullstellen oder Maxima- und Minima-Werte zu schätzen. Im nächsten Kapitel, "Der Löser", erfahren Sie, wie Sie über diese Näherungen genaue Zahlenwerte erhalten können.

Das vorliegende Kapitel zeigt Ihnen, in welcher Weise einige der im PLOT Menü enthaltenen Befehle zu verwenden sind. Alle Befehle des PLOT Menüs sind kurz in Anhang B dargestellt. Eine vollständige Beschreibung finden Sie unter "PLOT" im Referenzhandbuch.

Im ersten Beispiel ist die Funktion sin x darzustellen (Bogenmaß als Winkelmodus). Zuerst sind jedoch einige vorbereitende Schritte durchzuführen.

Für die grafische Darstellung eines Ausdrucks wird die Variable PPAR zum Speichern einer Liste mit Abbildungsparametern verwendet. Löschen Sie eine eventuell bestehende Variable PPAR, um sicherzustellen, daß für die nächste Abbildung die Standardwerte zur Anwendung kommen.

Löschen Sie den Stack und wählen Sie das PLOT Menü.

CLEAR
PLOT

Anzeige der zweiten Zeile des PLOT Menüs.

NEXT

Löschen Sie eine evtl. bestehende Variable PPAR.

' PPAR ■ PURGE

Wählen Sie als Winkelmodus das Bogenmaß und STD als Zahlen-Anzeigeformat.

■ MODE RAD STD

Geben Sie nun den ersten Ausdruck ein.

' TRIG SIN X ENTER

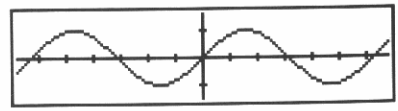
Speichern Sie den Ausdruck als *momentane Gleichung*—in einer normalen Variablen mit dem speziellen Namen EQ. (Dies entspricht der gleichen Konvention für den Löser, welcher Sie in Kapitel 4 gefolgt sind.)

■ PLOT STEQ

Drücken von STEQ ist gleichwertig mit dem Drücken von ' EQ STO.

Erzeugen Sie die grafische Darstellung des Ausdrucks.

DRAW



Warten Sie, bis der ((•)) Indikator am oberen Anzeigerand wieder erlöscht. Dies zeigt das Ende des Abbildungsvorgangs an.

Die horizontale Achse stellt die unabhängige Variable dar (x im vorliegenden Beispiel), und die vertikale Achse entspricht der abhängigen (Betrag des Ausdrucks *sin* x). Die Strichmarken auf beiden Achsen kennzeichnen Intervalle mit der Länge 1.

Drucken einer Grafik

Wenn Ihnen ein HP 82240A Drucker zur Verfügung steht, können Sie einen Ausdruck der soeben erstellten Grafik wie folgt erzeugen:

1. Positionieren Sie den Drucker entsprechend den Anweisungen im Handbuch des Druckers.
2. Halten Sie [ON] gedrückt.
3. Drücken Sie [L] (Taste, über welcher "PRINT" steht).
4. Geben Sie [ON] wieder frei.

Diese Tastenfolge entspricht dem Befehl PRLCD (*PRint LCD*, welcher sich in der ersten Zeile des PRINT Menüs befindet). Sie können diese Tastenfolge praktisch jederzeit zum Ausdrucken des Anzeigeinhalts benutzen, ohne dabei die Rechneroperationen zu unterbrechen.

Wenn Sie ein Programm zum Abbilden eines Ausdrucks und zum Drucken desselben entwickeln, ist folgende Befehlsfolge zu verwenden:

...CLLCD DRAW PRLCD...

Hinsichtlich des momentanen Beispiels ist jetzt die normale Stackanzeige zurückzurufen.

[ON]

Ändern des Abbildungsmaßstabs

Allgemein ist anzumerken, daß durch die grafische Darstellung eines Ausdrucks nicht unbedingt sofort die gewünschten Resultate erzielt werden. Wenn Sie einen unbekannten Ausdruck abbilden, kann eine neue Definiton des Abbildungsbereichs—durch die Abbildungsparameter festgelegt— erforderlich sein, um die relevanten Eigenschaften des Ausdrucks deutlicher zu machen.

Haben Sie bereits eine genaue Vorstellung des Abbildungsbereichs, welchen Sie darstellen möchten, so können Sie die Abbildungsparameter in PPAR direkt ändern. Manchmal müssen Sie aber auch etwas experimentieren, um den gewünschten Abbildungsbereich herauszufinden.

Als zweites Beispiel soll der Ausdruck $x^3 - x^2 - x + 3$ abgebildet werden.

Speichern Sie den Ausdruck in Ebene 1.

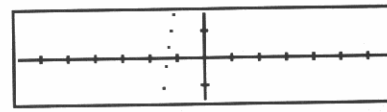
' X ■ ^ 3 − X ■ ^ 2 − X + 3
ENTER

Speichern Sie den Ausdruck als momentane Gleichung.

STEQ

Stellen Sie nun den Ausdruck grafisch dar.

DRAW



Die Abszisse entspricht den Werten der unabhängigen Variablen x, und die Ordinate stellt die Werte für $x^3 - x^2 - x + 3$ dar.

Die Abbildung zeigt Ihnen eine *Nullstelle* des Ausdrucks—ein Wert für X, an welchem der Wert des Ausdrucks gleich Null ist. Das nächste Kapitel zeigt, wie über den Löser ein genauer Zahlenwert für diese Nullstelle gefunden werden kann.

Um mehr von der Kurve abzubilden, ist die Skalierung der Ordinate zu erweitern.

Rufen Sie die normale Stackanzeige zurück.

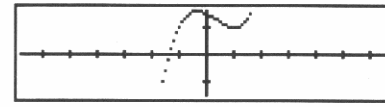
ON

Erweitern Sie die Höhe um den Faktor 2, indem Sie die Operation *H (*mal Höhe*) in der nächsten Menüzeile verwenden.

NEXT 2 *H

Stellen Sie die Kurve unter Verwendung der neuen Abbildungsparameter dar.

PREV DRAW



Die Strichmarken auf der Abszisse zeigen noch immer ein Intervall der Länge 1 an, die Markierungen auf der Ordinate entsprechen jedoch einem Intervall der Länge 2.

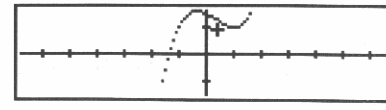
Verschieben Sie nun die Abbildung, indem Sie den interessanten Teil in das Zentrum der Anzeige verlegen.

Verschieben der Grafik

Nach jedem Abbildungsvorgang bleibt ein Fadenkreuz im Zentrum der Anzeige zurück (unsichtbar, wenn es mit Koordinatenursprung übereinstimmt). Mit Hilfe dieses kleinen Kreuzes können Sie jeden Punkt in der Anzeige *digitalisieren*, womit die Koordinaten des jeweiligen Punktes (in Abhängigkeit zum momentanen Abbildungsmaßstab) in den Stack zurückgegeben werden. Es soll nun der Punkt für den neuen Koordinatenursprung der nächsten Abbildung gewählt und zur Korrektur der Abbildungsparameter verwendet werden.

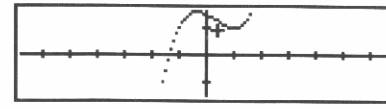
Bewegen Sie das Fadenkreuz an die angedeutete Position.

▶ (viermal drücken)
▲ (neunmal drücken)



Digitalisieren Sie diesen Punkt.

INS



Rufen Sie den Stack in die Anzeige zurück.

ON

Die Koordinaten des digitalisierten Punktes, durch eine komplexe Zahl dargestellt, erscheinen in Ebene 1.

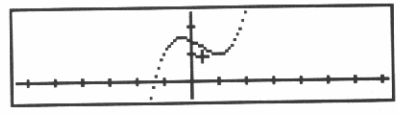
Redefinieren Sie den Koordinatenursprung unter Verwendung von CENTR in der nächsten Menüzeile.

NEXT CENTR

Die Koordinaten werden vom Stack genommen und zur Modifikation der Abbildungsparameter verwendet. Im Gegensatz zu *H ändert CENTR nicht die Skalierung.

Versuchen Sie eine erneute Abbildung des Ausdrucks.

PREV DRAW

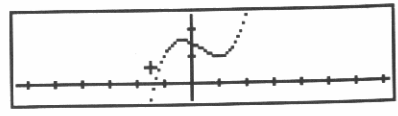


Vergrößern Sie nun einen interessanten Teil der Kurve. Sie können erneut *H verwenden, jetzt aber mit einem *gebrochenen* Skalierungsfaktor. (So würde z.B. .5 den vertikalen Maßstab wieder auf seinen ursprünglichen Wert vergrößern.) Es gibt aber auch ein flexibeleres Verfahren, um eine Abbildung zu vergrößern.

Neudefinieren des Abbildungsbereichs

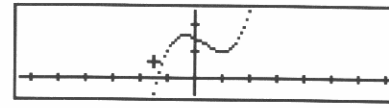
Zur vergößerten Darstellung der lokalen Extremwerte sollen zwei Punkte digitalisiert werden, einer für die untere linke Ecke sowie einer für die obere rechte Ecke der neuen Abbildung.

Bewegen Sie das Fadenkreuz in die gewünschte linke untere Ecke.

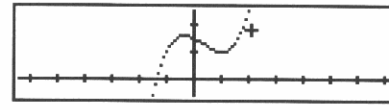


Digitalisieren Sie den Punkt.

INS

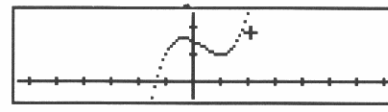


Bewegen Sie das Fadenkreuz in die gewünschte obere rechte Ecke.



Digitalisieren Sie den Punkt.

INS



Rufen Sie den Stack in die Anzeige zurück.

ON

Die Koordinaten des unteren linken Eckpunkts (in komplexer Darstellungsform) befinden sich in Ebene 2, die des oberen rechten Punkts in Ebene 1. (Die angezeigten Werte auf Ihrem HP-28S können leicht von den nachstehenden Werten abweichen.)

Redefinieren Sie unter Verwendung von PMAX (*Plot MAXima*) die obere rechte Ecke der Abbildung.

PMAX

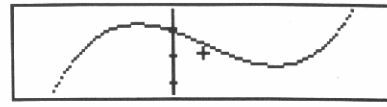
Die Koordinaten werden vom Stack genommen und als Abbildungsparameter für PMAX verwendet.

Redefinieren Sie unter Verwendung von PMIN (*Plot MINima*) die untere linke Ecke der Abbildung.

PMIN

Stellen Sie den Ausdruck erneut grafisch dar.

DRAW



Da Sie die Höhe und die Breite der Abbildung verändert haben, hat sich der Maßstab für Abszisse und Ordinate geändert.

Die Abbildung macht zwei *Extremstellen* der Kurve deutlich—ein lokales Maximum und Minimum. Im nächsten Kapitel ist die Ermittlung eines genauen Werts für das Minimum mit Hilfe des Lösers beschrieben. Um eine Wiederholung aller Schritte zum Erzeugen der momentanen Abbildungsparameter zu vermeiden, ist der momentane Inhalt von PPAR in einer anderen Variablen zu speichern. Soll die vorliegende Abbildung wieder erzeugt werden, so müssen Sie lediglich die alten Werte für PPAR zurückspeichern.

Rufen Sie den Stackinhalt in die Anzeige zurück.

ON

Speichern Sie den momentanen Inhalt von PPAR im Stack.

NEXT PPAR

Weitere Informationen über die Abbildungsparameter und Details über das Erzeugen einer grafischen Darstellung im allgemeinen finden Sie unter "PLOT" im Referenzhandbuch.

Erzeugen Sie die Variable PPAR1, in welcher die momentanen Abbildungsparameter gespeichert werden sollen.

' PPAR 1 STO

Sie können nun den Löser verwenden, um einen genauen Zahlenwert für eine Nullstelle oder einen lokalen Extremwert des Ausdrucks aufzufinden.

Grafische Darstellung von Gleichungen

Die in diesem Kapitel verwendeten Beispiele bezogen sich jeweils auf einen Ausdruck, es gelten jedoch die gleichen Bedingungen und Schritte zum Abbilden von Gleichungen. Enthält die Variable EQ eine Gleichung, so bildet DRAW *jede Seite* der Gleichung als Ausdruck ab. Sie können eine Lösung der Gleichung dadurch ermitteln, indem Sie den Schnittpunkt beider Kurven untersuchen. An dieser Stelle nehmen beide Seiten den gleichen Wert an.

8

Der Löser

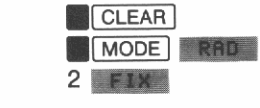
In diesem Kapitel ist das Auffinden einer Nullstelle und eines Extremwertes für den im vorangehenden Kapitel abgebildeten Ausdruck beschrieben. Sie werden dazu einige Resultate aus Kapitel 7 benötigen und sollten deshalb die relevanten Schritte in Kapitel 7 durchführen, falls Sie dies nicht bereits getan haben.

Eine vollständige Beschreibung des Lösers finden Sie unter "SOLVE" im Referenzhandbuch.

Auffinden einer Nullstelle für einen Ausdruck

Im nachfolgenden Beispiel wird angenommen, daß der Ausdruck $x^3 - x^2 - x + 3$ noch immer als momentane Gleichung in EQ gespeichert ist und daß die Variable PPAR1 erzeugt wurde. Sie werden die Abbildung nochmals erstellen und dabei eine Anfangsnäherung für eine Nullstelle digitalisieren, deren Koordinaten dann vom Löser zum Auffinden einer genauen Nullstelle verwendet werden.

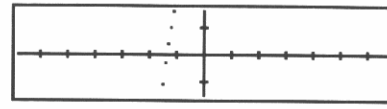
Löschen Sie zuerst den Stackinhalt; stellen Sie das Bogenmaß als Winkelmodus ein und wählen Sie FIX 2 für die Zahlenanzeige.



Löschen Sie PPAR, um die Standard-Abbildungsparameter zu verwenden.

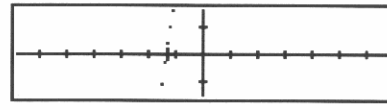
Stellen Sie nun den Ausdruck grafisch dar.

[■] [PREV] DRAW



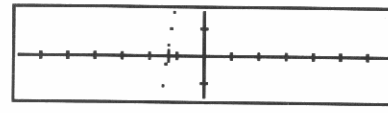
Diese Abbildung zeigt eine *Nullstelle* für den Ausdruck—ein Wert von X, für welchen der Ausdruck den Wert Null annimmt. Sie finden diesen Wert an der Schnittstelle der Kurve und der Abszisse.

Bewegen Sie das Fadenkreuz an die ungefähre Schnittstelle der Kurve mit der horizontalen Achse. (Benutzen Sie [▲], [▼], [◀] und [▶], um das Fadenkreuz zu verschieben.)



Digitalisieren Sie diese Anfangsnäherung für die Nullstelle.

[INS]



Dieser Punkt dient als Näherungswert zum Auffinden einer exakten Nullstelle für den Ausdruck. (Für den Fall, daß mehrere Nullstellen existieren, bestimmt die Näherung die von Ihnen gewünschte Nullstelle.)

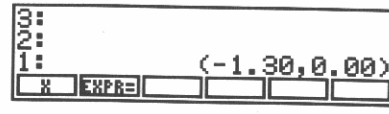
Rufen Sie den Stackinhalt in die Anzeige zurück.

[ON]

Die Koordinaten des digitalisierten Punktes erscheinen in komplexer Darstellung in Ebene 1. (Ihre Koordinaten können von den dargestellten Daten leicht abweichen.)

Wählen Sie das Löser-Menü.

[SOLV] SOLVR



Das Löser-Menü zeigt alle Variablen der momentanen Gleichung an (in diesem Beispiel nur X).

Speichern Sie die Anfangsnäherung in der Variablen X.

[X]

Obwohl der digitalisierte Punkt zwei Koordinatenwerte enthält, benutzt der Löser nur die erste Koordinate als Anfangsnäherung.)

Lösen des Ausdrucks nach X.

[■] [X]

Die Meldung `Sign Reversal` (*Vorzeichenwechsel*) deutet an, daß der Löser eine ungefähre Nullstelle (bis zu 12 Stellen genau) gefunden hat. Konnte der Löser eine exakte Lösung ermitteln, so wird die Meldung `Zero` angezeigt. Diese Hinweise werden als *qualifizierende Meldungen* bezeichnet und sind ausführlich unter "SOLVE" im Referenzhandbuch beschrieben.

Rufen Sie den Stackinhalt in die Anzeige zurück.

[ON]

Auffinden von Extremwerten

Um die Nullstelle eines Ausdrucks zu bestimmen, testet der Löser verschiedene Punkte der Kurve (beginnend mit Ihrer Anfangsnäherung), um Punkte näher zur X-Achse zu finden. Ist Ihre Näherung sehr nahe zu einem *positiven lokalen Minimum* oder zu einem *negativen lokalen Maximum*, so gibt es keine näheren Punkte zur X-Achse. Damit findet der Löser den gegebenen Extremwert (Minimum oder Maximum) anstatt einer Nullstelle. (Normalerweise bleibt der Löser nicht bei einem Extremwert "hängen", sofern Ihre Anfangsnäherung ihn nicht dazu zwingt.)

Sehen Sie sich die Kurve auf Seite 96 an. Aus ihr wird ersichtlich, daß der Ausdruck ein positives lokales Minimum und ein positives lokales Maximum bestitzt. Der Löser kann das Minimum bestimmen, da es sich in der Umgebung des *am nächsten* zur x-Achse liegenden Punktes befindet; das Maxium kann nicht ermittelt werden, da es sich in der Umgebung des Kreuzes befindet, welches *am weitesten* von der x-Achse entfernt ist.

Nachfolgend werden Sie den Ausdruck abbilden, indem Sie die in PPAR1 gespeicherten Abbildungsparameter verwenden. Danach sind drei Punkte als Näherung für ein Minimum zu digitalisieren, welche als Näherungswerte für Löser zur Bestimmung eines genauen Minimums dienen.

Rufen Sie die in PPAR1 gespeicherte Liste in den Stack zurück.

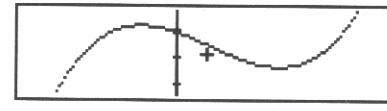
USER PPAR1

Speichern Sie die Variable PPAR zum Inhalt von PPAR1.

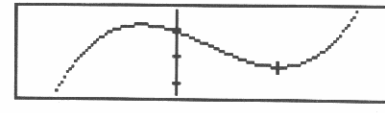
' PPAR STO

Erzeugen Sie die grafische Darstellung des Ausdrucks.

PLOT DRAW

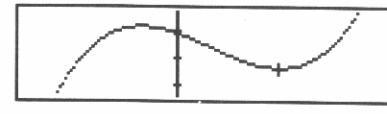


Stellen Sie das Fadenkreuz an das ungefähre Minimum.

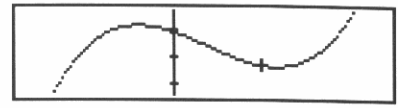


Digitalisieren Sie den Punkt.

INS

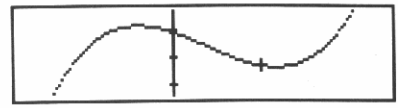


Bewegen Sie das Fadenkreuz etwas nach links vom Minimum.



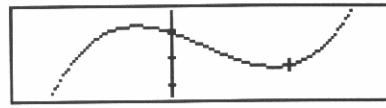
Digitalisieren Sie den Punkt.

INS



Bewegen Sie das Fadenkreuz etwas nach rechts vom Minimum und digitalisieren Sie den Punkt.

INS



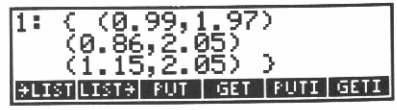
Rufen Sie den Stack in die Anzeige zurück.

ON

Die drei Punkte sind nun in Ebene 1, 2 und 3 gespeichert. (Ihre Werte können von den vorgegebenen leicht abweichen.)

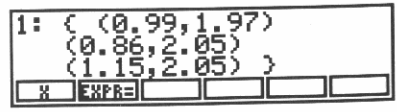
Kombinieren Sie die drei Näherungen nun in einer Liste. Dadurch können sie als ein einzelnes Objekt behandelt werden. Dies stellt eine typische Anwendung von Listen dar—Zusammenfassen mehrerer Objekte zu einem einzigen Objekt.

■ LIST 3 →LIST



Wählen Sie das Löser-Menü.

SOLV SOLVR



Das Löser-Menü zeigt alle Variablen der momentanen Gleichung (in diesem Beispiel nur X).

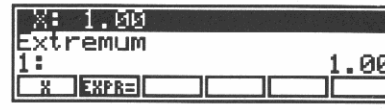
Speichern Sie die Liste der Datenpunkte in der Variablen X.

[X]

Die Datenpunkte der Liste werden vom Stack genommen und als Anfangsnäherungen in der Variablen X gespeichert.

Lösen Sie nach X.

[ ] [X]



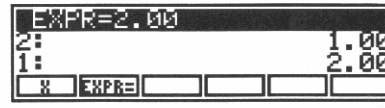
Die Meldung `Extremum` zeigt an, daß der Löser einen Extremwert für den Ausdruck gefunden hat.

Kehren Sie zur normalen Stackanzeige zurück.

[ON]

Berechnen Sie den Extremwert.

[EXPR=]



Das Mimimum ist 2.

Finanzmathematische Berechnungen

Dieser Abschnitt beschäftigt sich mit den Anwendungsmöglichkeiten des Lösers für finanzmathematische Aufgabenstellungen. Wenn n die Anzahl der Zahlungsperioden, i den Zinssatz pro Zahlungsperiode (in Prozent), *rate* den periodischen Zahlungsbetrag, *barw* den Barwert und *endw* den Endwert darstellt, ergibt sich nachfolgende mathematische Beziehung:

$$(1 - sppv) \times rate \times (100/i) + barw = -endw \times sppv$$

wobei

$$sppv\ (Barwert\ einer\ einzelnen\ Zahlung) = (1 + i/100)^{-n}$$

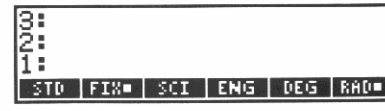
$$= \exp(-n \times \ln(1 + i/100)).$$

Diese Gleichung setzt die Rückzahlungen zum Ende jeder Periode voraus.

Hier die grundsätzlichen Schritte, die ausgeführt werden müssen:

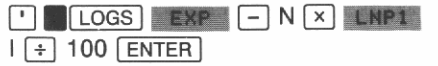
1. Tippen Sie den Ausdruck für *sppv* ein und speichern Sie diesen in der Variablen SPPV.
2. Tippen Sie die Gleichung ein und speichern Sie sie in der Variablen ANNU (*ANNUitäten*).
3. Bestimmen Sie ANNU als momentane Gleichung.
4. Verwenden Sie den Löser zur Berechnung von *n*, *i*, *rate*, *barw* oder *endw*, wobei außer der gesuchten Variablen die Werte für die restlichen Variablen vorgegeben sein müssen.

Löschen Sie, bevor Sie beginnen, den Stack und wählen Sie FIX 2 als Zahlen-Anzeigeformat.



Schritt 1: Tippen Sie den Ausdruck für *sppv* ein und speichern Sie ihn in der Variablen SPPV.

Tippen Sie den Ausdruck für *sppv* ein.



Dieser Ausdruck ist vorteilhafter, da LNP1 den Ausdruck $\ln(1 + i/100)$ mit einer höheren Genauigkeit berechnet.

Erzeugen Sie die Variable SPPV und prüfen Sie das USER Menü.

Schritt 2: Tippen Sie die Gleichung ein und speichern Sie sie in der Variablen ANNU.

Tippen Sie die Gleichung für ANNU ein.

' (1 − SPPV) × RATE ×
100 ÷ I + BARW = − ENDW ×
SPPV ENTER

Erzeugen Sie die Variable ANNU.

' ANNU STO

Das USER Menü zeigt ein neues Menüfeld (für ANNU).

Schritt 3: Bestimmen Sie ANNU zur momentanen Gleichung.

Tippen Sie den Namen "ANNU" ein.

' ANNU

Speichern Sie ANNU in der Variablen EQ.

SOLV STEQ

Schritt 4: Verwenden Sie den Löser zur Berechnung von *n*, *i*, *rate*, *barw* oder *endw*, wobei außer der Unbekannten die restlichen Werte vorgegeben sein müssen.

Wählen Sie das Löser-Menü.

SOLVR

Alle Variablen von ANNU und SPPV erscheinen nun im Menü. (Die Variablen in SPPV erscheinen deshalb, weil die momentane Gleichung, ANNU, SPPV enthält.)

Berechnen Sie nun BARW unter Verwendung der Werte N = 30 × 12, I = 11,5/12, RATE = −630 und ENDW = 0. Der Betrag für RATE hat ein negatives Vorzeichen, da es sich um einen abfließenden Betrag handelt, während zufließende Beträge mit einem positiven Vorzeichen versehen werden.

Weisen Sie zuerst einen Wert für N zu.

30 [ENTER] 12 [×] [N]

Weisen Sie den Wert für I zu.

11.5 [ENTER] 12 [÷] [I]

Weisen Sie den Wert für RATE zu.

−630 [RATE]

Weisen Sie den Wert für ENDW zu.

0 [ENDW]

Lösen Sie nun nach BARW.

[BARW]

Die Meldung `Zero` zeigt an, daß der berechnete Wert eine genaue Lösung für die momentane Gleichung darstellt.

9

Symbolische Lösungen

In diesem Kapitel werden zwei Verfahren zum Auffinden von symbolischen Lösungen erläutert. Ein einfacher Weg zur Lösung einer quadratischen Gleichung besteht in der Berechnung des linearen Ausdrucks, welcher beide Nullstellen darstellt. Daneben gibt es jedoch noch eine vielseitigere Methode, welche die symbolische Lösung für eine Variable in allgemeineren Gleichungen liefert.

Jedes Verfahren läßt sich auf algebraische Ausdrücke sowie auf Gleichungen anwenden. Die *Nullstelle* eines Ausdrucks $f(x)$ ist das selbe wie die *Lösung* der Gleichung $f(x) = 0$, und die *Lösung* der Gleichung $f(x) = g(x)$ ist äquivalent zur *Nullstelle* des Ausdrucks $f(x) - g(x)$.

Nullstellen eines quadratischen Ausdrucks

Sie können beide Nullstellen eines quadratischen Ausdrucks ermitteln, ohne daß die entsprechende Kurve abgebildet werden muß. Im folgenden Beispiel werden die Nullstellen für den Ausdruck $x^2 - 6x + 8$ berechnet.

Bevor Sie mit dem Beispiel beginnen, ist der Stackinhalt zu löschen und STD als Zahlen-Anzeigeformat zu spezifizieren.

[ ] CLEAR
[ ] MODE STD

Speichern Sie den Ausdruck im Stack.

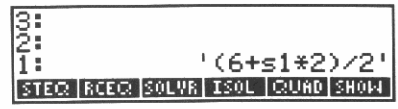
' X [ ] ^ 2 − 6 × X + 8 ENTER

Speichern Sie den Namen X im Stack, womit die zu lösende Variable spezifiziert wird.

' X ENTER

Berechnen Sie die Nullstellen unter Verwendung von QUAD (*QUADratic*) im Löser-Menü.

SOLV QUAD



Dieser Ausdruck stellt beide Nullstellen für den quadratischen Ausdruck dar. Die Variable s1 steht stellvertretend für ein beliebiges Vorzeichen, entweder +1 oder −1, und jeder Wert von s1 entspricht einer Nullstelle des Ausdrucks.

Speichern Sie den Ausdruck als momentane Gleichung.

STEQ

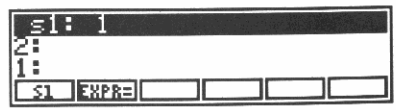
Anzeige des Löser-Menüs.

SOLVR

s1 ist als einzige Variable in der momentanen Gleichung enthalten.

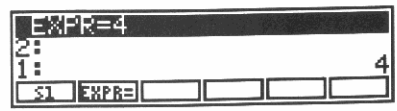
Spezifizieren Sie zuerst 1 für s1, um ein positives Vorzeichen zu erhalten.

1 S1



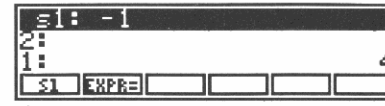
Geben Sie eine der Nullstellen in Ebene 1 zurück.

EXPR=



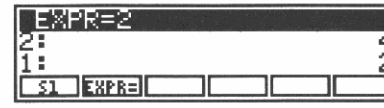
Spezifizieren Sie nun ein negatives Vorzeichen für s1.

−1 [S1]



Geben Sie die zweite Nullstelle in Ebene 1 zurück.

[EXPR=]



Die Nullstellen für $x^2 - 6x + 8$ sind $x = 4$ und $x = 2$.

Isolieren einer Variablen

Mit Hilfe Ihres HP-28S können Sie eine einzelne Variable in einer Gleichung isolieren, wodurch Sie einen Ausdruck erhalten, der die symbolische Lösung für die betreffende Gleichung darstellt. Das heißt, wenn z.B. x die zu lösende Variable der Gleichung ist und neben x noch die Variablen a, b und c in der Gleichung enthalten sind, dann wird durch das Isolieren von x ein Ausdruck mit a, b und c erzeugt, der die Gleichung erfüllt, wenn x den Wert des Ausdrucks annimmt.

Als erstes Beispiel soll x in der Gleichung

$$a(x + 3) - b = c$$

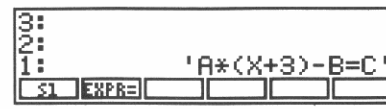
isoliert werden. Dies ist einfach, da x nur einmal vorkommt. In folgenden Beispielen wird aufgezeigt, wie eine Gleichung zu modifizieren ist, um x auf ein einzelnes Vorkommen zu reduzieren.

Löschen Sie den Stackinhalt.

[■] [CLEAR]

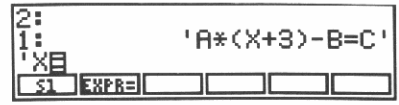
Speichern Sie die Gleichung im Stack.

['] A [×] [(] X [+] 3 [■] [)] [−] B [=] C
[ENTER]



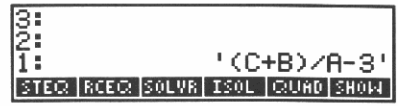
Spezifizieren Sie die zu isolierende Variable.

' X



Isolieren Sie x, indem Sie ISOL im Löser-Menü verwenden.

SOLV ISOL



Der erhaltene Ausdruck stellt eine symbolische Lösung der Gleichung für x dar—das heißt, die Gleichung

$$a(x + 3) - b = c$$

ist dann erfüllt, wenn $x = (c + b)/a - 3$.

Erweitern und Zusammenfassen

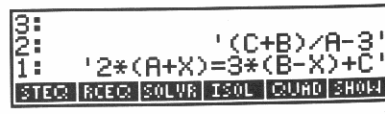
Wenn x mehrmals in der Gleichung vorkommt, so müssen Sie die Gleichung in der Weise modifizieren, daß Sie das Erscheinen von x auf ein einziges Vorkommen reduzieren. Im nächsten Beispiel wird gezeigt, wie Sie x in der Gleichung

$$2(a + x) = 3(b - x) + c$$

isolieren können. Der Lösungsweg dazu liegt im Erweitern der Gleichung, anschließender Subtraktion eines x-Terms von beiden Seiten und im Zusammenfassen der Gleichung, um den x-Term auf einer Seite zu löschen und auf der anderen Seite einen einzelnen x-Term zu schaffen.

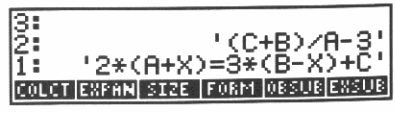
Speichern Sie die Gleichung im Stack.

' 2 × (A + X ▶) = 3 × (B
− X ▶) + C ENTER



Wählen Sie das ALGEBRA Menü.

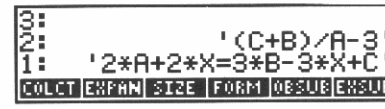
▶ ALGBRA



Dieses Beispiel verwendet die Befehle EXPAN (*EXPANd bzw. erweitern*) und COLCT (*COLleCT bzw. zusammenfassen*) zum Modifizieren der Gleichung. Anschließend wird die Gleichung bzw. deren Terme unter Verwendung von FORM (*algebraischen Ausdruck formen*) neu angeordnet. Alle Befehle im ALGEBRA-Menü sind kurz in Anhang B, "Menütabellen", beschrieben. Eine vollständige Beschreibung finden Sie unter "ALGEBRA" im Referenzhandbuch. Zusätzlich wird FORM, ein leistungsstarker algebraischer Editor, in "ALGEBRA (FORM)" des Referenzhandbuchs erläutert.

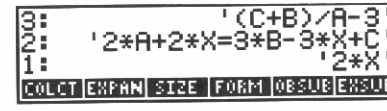
Erweitern Sie beide Seiten der Gleichung.

EXPAN



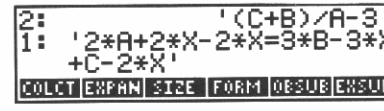
Um den x-Term ($2x$) der linken Seite von beiden Seiten der Gleichung zu subtrahieren, ist der linksseitige x-Term im Stack zu speichern.

' 2 × X ENTER



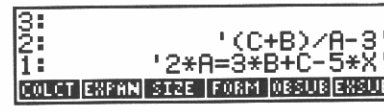
Subtrahieren Sie nun $2x$ von beiden Seiten.

−



Fassen Sie die Terme der Gleichung zusammen.

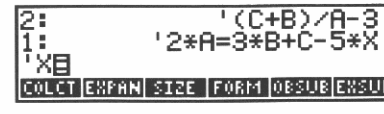
COLCT



Jede Seite wird getrennt bearbeitet, und die x-Terme auf der linken Seite heben sich gegenseitig auf.

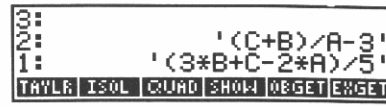
Nun können Sie x in der Gleichung isolieren. Spezifizieren Sie die zu isolierende Variable.

' X



Isolieren Sie x. Der Befehl ISOL erscheint in der zweiten Zeile des ALGEBRA Menüs sowie im Löser-Menü.

NEXT ISOL



Der erhaltene Ausdruck stellt eine symbolische Lösung der Gleichung für x dar—das bedeutet, die Gleichung

$$2(a + x) = 3(b - x) + c$$

ist dann erfüllt, wenn $x = (3b + c - 2a)/5$.

Anwenden von FORM

Wenn x an mehreren Stellen in der Gleichung vorkommt und ein x dabei in Verbindung mit einem symbolischen Koeffizienten steht, dann faßt der Befehl COLCT die Koeffizienten nicht zusammen. Im nächsten Beispiel isolieren Sie x in der Gleichung

$$a(x + b) + 2x = c$$

wobei x an mehreren Stellen vorkommt und mit einem symbolischen Koeffizienten a verwendet wird. Hierbei ist wie folgt vorzugehen: Erweitern Sie die Gleichung, verwenden Sie FORM zum Zusammenfassen der Koeffizienten von x, und isolieren Sie danach x.

Speichern Sie die Gleichung im Stack.

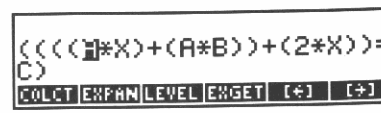
' A × (X + B) + 2 × X =
C ENTER

Erweitern Sie die Gleichung.

NEXT EXPAN

Verwenden Sie nun FORM zum Zusammenfassen der Koeffizienten von x.

FORM



Während der Ausführung von FORM ist die normale Betriebsweise des Rechners unterbrochen. Die FORM-Anzeige zeigt die Gleichung, wobei die Teilausdrücke durch Klammern voneinander getrennt sind. Mit Hilfe von FORM sind nun diese Teilausdrücke innerhalb der Gleichung zu modifizieren.

Das Ziel ist, die Terme `(A*X)` und `(2*X)` zu einem einzelnen Term `((A+2)*X)` zu kombinieren. Dazu sind die nachstehenden drei Schritte erforderlich; die momentane Form der Gleichung ist:

$$(ax + ab) + 2x = c$$

Im ersten Schritt werden ax und ab kommutiert, was zu der Form führt:

$$(ab + ax) + 2x = c$$

Im zweiten Schritt werden ax und $2x$ assoziiert, was zu der Form führt:

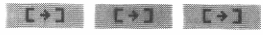
$$ab + (ax + 2x) = c$$

Im dritten Schritt werden ax und $2x$ zusammengeführt, was zu der Form führt:

$$ab + (a + 2)\,x = c$$

Schritt 1: Kommutieren Sie ax und ab.

Bewegen Sie den Cursor (invers dargestellte(s) Zeichen) über +.



Die Position des Cursors bestimmt den Teilausdruck, auf welchen die Operation angewendet werden soll. In diesem Beispiel ist auf den Teilausdruck `((A*X)+(A*B))` das Kommutativgesetz der Addition anzuwenden.

Nachdem Sie nun das Erscheinen von x auf eins reduziert haben, können Sie x relativ einfach isolieren.

Spezifizieren Sie die zu isolierende Variable.

' X

Isolieren Sie x.

NEXT ISOL

Der in Stackebene 1 angezeigte Ausdruck stellt eine symbolische Lösung der Gleichung für x dar—das bedeutet, die Gleichung

$$a(x + b) + 2x = c$$

ist dann erfüllt, wenn $x = (c - ab)/(a + 2)$.

10

Höhere Mathematik

Sie können jeden beliebigen Ausdruck, für welchen eine vernünftige Ableitung existiert, symbolisch differenzieren. Bei der Integration gibt es eine Einschränkung: Sie können das *bestimmte numerische Integral* eines beliebigen Ausdrucks berechnen, das *genaue symbolische Integral* jedoch nur für Polynome.

Dieses Kapitel enthält einfache Beispiele zum Auffinden von Ableitungen, unbestimmten und bestimmten Integralen. Weitere Informationen dazu finden Sie unter "Höhere Mathematik" im Referenzhandbuch.

Differenzieren von Ausdrücken

Sie können einen Ausdruck schrittweise differenzieren und dabei beobachten, wie der Rechner die verschiedenen Regeln zur Differentiation von Ausdrücken anwendet. Ein Ausdruck kann aber auch über einen einzigen Schritt differenziert werden. Das Endresultat ist bei beiden Vorgehensweisen identisch. Sie werden als nächstes einen Ausdruck zweimal differenzieren, zuerst schrittweise und anschliessend über eine einzige Operation.

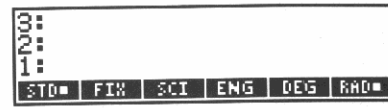
Schrittweises Differenzieren

Um die Differentiation in mehreren Schritten durchzuführen, tippen Sie die abgeleitete Funktion als Ausdruck ein. Differenzieren Sie z.B.

$$\frac{d}{dx} \tan (x^2 + 1)$$

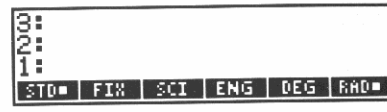
Bevor Sie mit dem Beispiel beginnen, ist der Stackinhalt zu löschen, Bogenmaß als Winkelmodus und STD als Zahlen-Anzeigeformat zu spezifizieren.

[■] [CLEAR]
[■] [MODE] RAD
STD



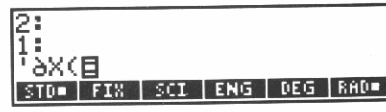
Löschen Sie die Variable X (sofern vorhanden).

['] X [■] [PURGE]



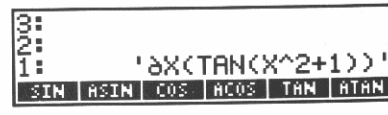
Beginnen Sie nun den Ausdruck für die Ableitung, indem Sie die Differentiationsvariable eintippen.

['] [■] [d/dx] X [(]



Als nächstes ist der zu differenzierende Ausdruck einzugeben.

[TRIG] TAN X [■] [^] 2 [+] 1 [ENTER]



Dieser Ausdruck stellt die Ableitung von x für tan $(x^2 + 1)$ dar.

Werten Sie diesen Ausdruck einmal aus.

[EVAL]

Aus dem Ergebnis ist die Anwendung der Kettenregel ersichtlich:

$$\frac{d}{dx}\tan(x^2+1) = \frac{d}{d(x^2+1)}\tan(x^2+1) \times \frac{d}{dx}(x^2+1)$$

Die Ableitung von tan wurde ausgewertet. Im nächsten Schritt ist die Ableitung von $x^2 + 1$ auszuwerten.

Werten Sie den Ausdruck ein zweites Mal aus.

EVAL

Aus dem Ergebnis ist die Ableitung einer Summe ersichtlich:

$$\frac{d}{dx}(x^2+1) = \frac{d}{dx}x^2 + \frac{d}{dx}1$$

Die Ableitung von 1 ist 0, womit dieser Term verschwindet. Im folgenden Schritt ist die Ableitung von x^2 auszuwerten.

Werten Sie den Ausdruck ein drittes Mal aus.

EVAL

Das Ergebnis reflektiert erneut die Anwendung der Kettenregel:

$$\frac{d}{dx}x^2 = \frac{d}{dx}(x)^2 \times \frac{d}{dx}x$$

Damit wurde die Ableitung von x^2 berechnet. Was noch verbleibt, ist die Ableitung von x selbst.

Werten Sie den Ausdruck zum vierten Mal aus.

EVAL

Hiermit haben Sie den Ausdruck vollständig abgeleitet.

Vollständiges Differenzieren

Um einen Ausdruck über einen einzigen Schritt zu differenzieren, ist die Differentiation als Stackoperation durchzuführen. Nehmen Sie an, Sie möchten nochmals folgende Ableitung bestimmen:

$$\frac{d}{dx} \tan (x^2 + 1)$$

Speichern Sie den zu differenzierenden Ausdruck im Stack.

■ CLEAR
' TAN X ■ ^ 2 + 1 ENTER

Spezifizieren Sie die Differentiationsvariable.

' X ENTER

Differenzieren Sie den Ausdruck.

■ d/dx

Die vollständige Ableitung wird in Ebene 1 zurückgegeben.

Integrieren eines Ausdrucks

Ihr HP-28S berechnet das unbestimmte Integral eines Ausdrucks unter Anwendung von *symbolischer Integration,* welche einen Ausdruck als Ergebnis hat. Hierdurch erhalten Sie jedoch nur für Polynome genaue Ergebnisse. (Für andere Ausdrücke wird eine Taylorreihe für den ursprünglichen Ausdruck verwendet. Details hierzu finden Sie unter "Höhere Mathematik" im Referenzhandbuch.) Das nachfolgende Beispiel zeigt die Anwendung der symbolischen Integration.

Im Gegensatz zum o.a. werden bestimmte Integrale durch *numerische Integration* berechnet, welche ein numerisches Resultat erzeugt. Dieses Verfahren läßt sich auf jeden Ausdruck anwenden, der sich "normal" verhält (siehe zweites Beispiel unten).

Symbolische Integration eines Polynoms

Dieses Beispiel zeigt die symbolische Integration des Polynoms

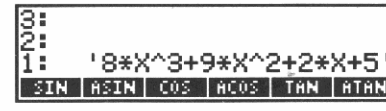
$$8x^3 + 9x^2 + 2x + 5.$$

Löschen Sie den Stackinhalt.

■ CLEAR

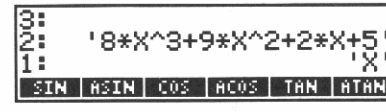
Speichern Sie das Polynom im Stack.

' 8 × X ■ ^ 3 + 9 × X ■ ^ 2 + 2 × X + 5 ENTER



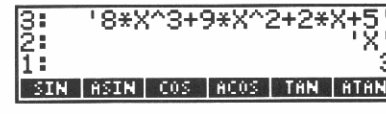
Spezifizieren Sie die Integrationsvariable.

' X ENTER



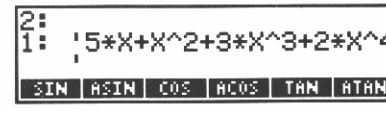
Spezifizieren Sie den Grad des Polynoms.

3 ENTER



Integrieren Sie das Polynom.

■ ∫



Warten Sie, bis der ((•)) Indikator wieder erlöscht, wodurch der Abschluß der Integration angezeigt wird. Das Integral wird in Ebene 1 zurückgegeben.

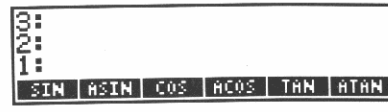
Numerische Integration eines Ausdrucks

In diesem Beispiel werden Sie einen numerischen Wert für das Integral berechnen.

$$\exp\,(x^3 + 2x^2 - x + 4)\,dx$$

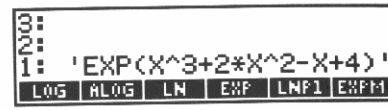
Löschen Sie den Stackinhalt.

▮ CLEAR



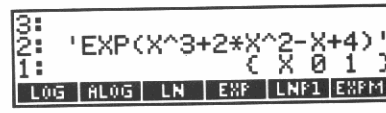
Speichern Sie den Ausdruck im Stack.

' ▮ LOGS EXP X ▮ ^ 3 + 2
× X ▮ ^ 2 − X + 4 ENTER



Tippen Sie die Variable und Integrationsgrenzen ein. Sie können dies in Form einer Liste durchführen. (Dies stellt einen typischen Anwendungsfall für eine Liste dar—das Zusammenfassen mehrerer Objekte, wodurch diese als ein einziges Objekt behandelt werden können.)

{ X SPACE 0 SPACE 1 ENTER



X ist die Integrationsvariable, 0 und 1 bilden die Integrationsgrenzen.

Tippen Sie nun die von Ihnen gewünschte Genauigkeit für die Durchführung der Integration ein.

Enthält der Ausdruck Konstanten, die aus empirischen Daten abgeleitet wurden, so spezifizieren Sie die Genauigkeit dieser Konstanten. Handelt es sich z.B. um Konstanten mit bis zu drei Dezimalstellen, so geben Sie .001 an.

Bei diesem Beispiel werden keine empirischen Konstanten verwendet, womit Sie eine 12-stellige Genauigkeit festlegen könnten. Da der iterative Rechenprozeß für eine numerische Integration mit zunehmender Genauigkeit eine längere Rechenzeit beansprucht, sollten Sie hier eine Genauigkeit von .00001 vorsehen.

1E−5 ENTER

Bestimmen Sie das Integral.

■∫

Der Näherungswert für das Integral wird in Ebene 2 zurückgegeben, während in Ebene 1 der zugehörige Fehlerbereich angezeigt wird.

Der Betrag für das Integral liegt bei 103.118 ± .001. Beachten Sie, daß der angegebene Fehlerbereich das angenäherte Produkt von dem Näherungswert für das Integral und der spezifizierten Genauigkeit darstellt.

11

Vektoren und Matrizen

Ihr HP-28S bearbeitet zwei Arten von Feldern: *Vektoren*, welche eindimensionale Felder darstellen, und *Matrizen*, die zweidimensionale Felder darstellen. Ihr HP-28S ermöglicht Ihnen die Eingabe von Vektoren und Matrizen als individuelle Objekte, die auch als *Feld-Objekte* bezeichnet werden. Sie können mit diesen Berechnungen genau so einfach wie mit gewöhnlichen Zahlen durchführen.

Dieses Kapitel zeigt die grundlegenden Anwendungen mit reellen Feldern—Vektoren und Matrizen, deren Elemente aus reellen Zahlen bestehen. Sie können aber genauso Berechnungen mit Feldern durchführen, deren Elemente aus komplexen Zahlen bestehen.

Alle Befehle im ARRAY Menü sind kurz in Anhang B, "Menütabellen", dargestellt. Eine vollständige Beschreibung finden Sie unter "ARRAY" im Referenzhandbuch.

Vektoren

Nachfolgend erfahren Sie, wie Sie auf dem HP-28S Vektorarithmetik anwenden sowie das Kreuz- und das Skalarprodukt bilden können.

Eintippen eines Vektors

Löschen Sie zuerst den Stackinhalt und wählen Sie STD als Zahlen-Anzeigeformat, bevor Sie mit der Bearbeitung des Beispiels beginnen.

CLEAR

MODE STD

Tippen Sie den Vektor [2 3 4] ein. Sie können als Trennzeichen zwischen den Koordinaten entweder Leerzeichen oder das alternative Dezimalzeichen verwenden.

[ ] 2,3,4 [ENTER]

Multiplizieren und Dividieren eines Vektors mit einer Zahl

Multiplizieren Sie den Vektor mit 15.

15 [×].

Die Reihenfolge der Argumente ist bei dieser Multiplikation ohne Bedeutung, genauso wie bei der Multiplikation zweier gewöhnlicher Zahlen. Bei der Division muß sich jedoch der Vektor in Ebene 2 und die Zahl in Ebene 1 befinden.

Dividieren Sie den Vektor durch 5.

5 [÷]

Addieren und Subtrahieren von Vektoren

Sie können Vektoren addieren und subtrahieren, wie Sie es von Zahlen her gewöhnt sind. Es wird lediglich vorausgesetzt, daß die Vektoren die gleiche Anzahl an Elementen besitzen. Bei der Subtraktion ist die Reihenfolge der Argumente entscheidend, genauso wie bei der Subtraktion zweier gewöhnlicher Zahlen.

Subtrahieren Sie den Vektor [−10 20 30].

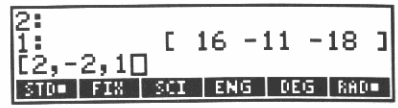
[ ] −10,20,30 [−]

Berechnen des Kreuzprodukts

Bilden Sie das Kreuzprodukt des Vektors aus Ebene 1 mit dem Vektor [2 −2 1]. (Das Kreuzprodukt ist nur für zwei- und dreielementige Vektoren definiert.)

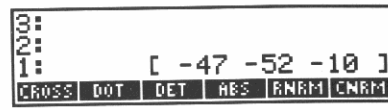
Tippen Sie den Vektor ein.

[ ] 2,−2,1



Berechnen Sie das Kreuzprodukt, indem Sie CROSS in der dritten Zeile des ARRAY Menüs drücken.

■ ARRAY NEXT NEXT CROSS

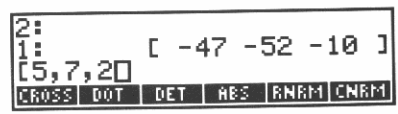


Berechnen des Skalarprodukts

Bilden Sie das Skalarprodukt von dem Vektor aus Ebene 1 (Ergebnis des vorherigen Beispiels) und dem Vektor [5 7 2]. (Beide Vektoren müssen dieselbe Anzahl Elemente besitzen.)

Tippen Sie den Vektor ein.

[ ] 5,7,2



Berechnen Sie das Skalarprodukt.

DOT

Matrizen

Dieser Abschnitt zeigt Ihnen, wie Sie eine Matrix invertieren und wie Sie die Determinante einer Matrix finden können. Beide Rechenalgorithmen beschränken sich auf eine *quadratische* Matrix—Anzahl der Zeilen und Spalten sind gleich.

Die Berechnungen, welche Sie mit Vektoren durchgeführt haben, lassen sich—mit Ausnahme des Kreuz- und Skalarprodukts—ebenso auf Matrizen anwenden. Sie können eine Matrix und eine Zahl auf die gleiche Weise multiplizieren, wie Sie den Vektor mit der Zahl multipliziert haben. Weiterhin ist die Addition und Subtraktion von Matrizen möglich, sofern beide Matrizen die gleiche Dimension haben.

Eintippen der Matrixelemente

Es soll folgende Matrix eingegeben werden:

$$\begin{bmatrix} 1 & 2 & 3 \\ 1 & 3 & 3 \\ 1 & 2 & 4 \end{bmatrix}$$

Tippen Sie das linke Begrenzungszeichen für Matrizen ein.

[[ ]

Geben Sie jede Zeile der Matrix wie einen einzelnen Vektor ein.

[[ ] 1,2,3
[[ ] 1,3,3
[[ ] 1,2,4 ENTER

Anzeigen einer großen Matrix

Wenn eine Matrix viele Elemente (oder gebrochene Elemente) besitzt, dann wird unter Umständen nur ein Teil der Matrix angezeigt. Um sämtliche Elemente anzuzeigen, müssen Sie ■ EDIT (wenn sich die Matrix in Ebene 1 befindet) oder ■ VISIT drücken, um die Matrix in die Befehlszeile zurückzurufen. Danach können Sie sich unter Verwendung des Cursor-Menüs jeden Teil der Matrix anzeigen lassen. Beziehen Sie sich für detaillierte Informationen dazu auf "Edieren gespeicherter Objekte" in Kapitel 18.

Invertieren einer Matrix

Da es sich bei der Matrix in Ebene 1 um eine quadratische Matrix handelt, können Sie die Inverse dieser Matrix bilden.

[1/x]

```
1: [[ 6 -2 -3 ]
    [ -1 1 0 ]
    [ -1 0 1 ]]
CROSS | DOT | DET | ABS | RNRM | CNRM
```

Bestimmen der Determinante

Da es sich bei der Matrix in Ebene 1 um eine quadratische Matrix handelt, können Sie die Determinante dieser Matrix bilden.

DET

```
3:
2:                    -619
1:                       1
CROSS | DOT | DET | ABS | RNRM | CNRM
```

Multiplizieren zweier Felder

Mit Hilfe der [×] Funktion können Sie auf einfache Weise zwei Matrizen oder eine Matrix mit einem Vektor multiplizieren. (Verwenden Sie CROSS oder DOT um zwei Vektoren, wie oben beschrieben, zu mulitplizieren.)

Multiplizieren zweier Matrizen

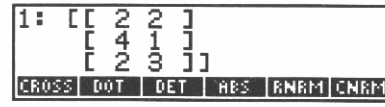
Bei der Multiplikation zweier Matrizen ist die Reihenfolge der Argumente von Bedeutung. Die Anzahl *Spalten* der Matrix in Ebene 2 muß mit der Anzahl *Zeilen* der Matrix in Ebene 1 übereinstimmen. So können Sie z.B. das Produkt der beiden folgenden Matrizen berechnen:

$$\begin{bmatrix} 2 & 2 \\ 4 & 1 \\ 2 & 3 \end{bmatrix} \begin{bmatrix} 2 & 2 & 1 & 4 \\ 3 & 4 & 2 & 1 \end{bmatrix}$$

Um dieses Matrizenprodukt zu berechnen:

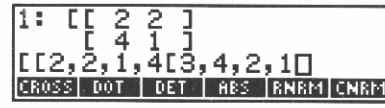
Geben Sie die erste Matrix ein.

[ ] [ ] 2,2
[ ] 4,1
[ ] 2,3 ENTER



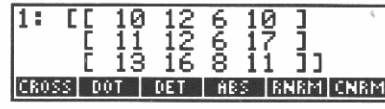
Tippen Sie die zweite Matrix ein.

[ ] [ ] 2,2,1,4
[ ] 3,4,2,1



Multiplizieren Sie die Matrizen.

×



Multiplizieren einer Matrix mit einem Vektor

Bei der Multiplikation einer Matrix mit einem Vektor ist die Reihenfolge der Argumente von Bedeutung. Die Matrix muß sich dabei in Ebene 2 und der Vektor in Ebene 1 befinden. Die Anzahl *Spalten* der Matrix muß der Anzahl *Elemente* des Vektors entsprechen.

Um die Matrix, welche sich momentan in Ebene 1 befindet, mit dem Vektor [3 1 1 2] zu multiplizieren:

Tippen Sie den Vektor ein.

[ ] 3,1,1,2

Multiplizieren Sie Matrix und Vektor.

×

Lösen eines linearen Gleichungssystems

Um ein System von n linearen Gleichungen mit n Variablen zu lösen, ist ein n-elementiger *Konstantenvektor*, eine $n \times n$ *Koeffizientenmatrix* und die Funktion [÷] anzuwenden. Der Konstantenvektor enthält die konstanten Werte der Gleichung, während die Koeffizientenmatrix die Koeffizienten der Variablen enthält.

Das nächste Beispiel zeigt, wie ein System von 3 linearen Gleichungen mit 3 unabhängigen Variablen gelöst werden kann. Nehmen Sie an, es würden diese Gleichungen vorliegen:

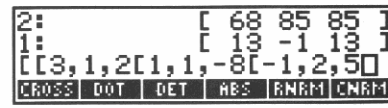
$$\begin{aligned} 3x + y + 2z &= 13 \\ x + y - 8z &= -1 \\ -x + 2y + 5z &= 13 \end{aligned}$$

Geben Sie den Konstantenvektor ein.

[⌊] 13,−1,13 [ENTER]

Tippen Sie die Koeffizientenmatrix ein.

[⌊] [⌊] 3,1,2
[⌊] 1,1,−8
[⌊] −1,2,5



Lösen Sie das Gleichungssystem.

[÷]

Die Elemente des Lösungsvektors stellen die Werte für die Variablen dar, welche den Gleichungen genügen. Dies ist der Fall für:

$$x = 2, \quad y = 5, \quad z = 1$$

Zum Lösen von Gleichungssystemen, in welchen die Anzahl der Variablen ungleich der Anzahl von linearen Gleichungen ist, beziehen Sie sich auf den Abschnitt "ARRAY" im Referenzhandbuch.

12

Statistische Funktionen

In diesem Kapitel wird zunächst erläutert, wie statistische Daten eingegeben werden. Anschließende Beispiele demonstrieren die Berechnungen mit einzelnen und gepaarten Stichprobenwerten unter Verwendung der Befehle im STAT Menü. Alle Befehle in diesem Menü sind kurz in Anhang B, "Menütabellen", dargestellt. Eine vollständige Beschreibung finden Sie unter "STAT" im Referenzhandbuch.

Die nachfolgende Tabelle enthält die Änderungen bezüglich des Verbraucherpreis-Index (VPI), des Erzeugerpreis-Index (EPI) und der Arbeitslosenrate (AL), wobei alle Werte in Prozent dargestellt sind und sich auf die USA für die Jahre 1975 bis 1979 beziehen. Geben Sie diese Daten ein, um anschließend statistische Auswertungen anstellen zu können.

Daten für Statistik-Beispiel

Jahr	VPI	EPI	AL
1975	9.1	9.2	8.5
1976	5.8	4.6	7.7
1977	6.5	6.1	7.0
1978	7.6	7.8	6.0
1979	11.5	19.3	5.8

Eingeben von Daten

Statistische Daten werden in einer Statistikmatrix mit dem Namen ΣDAT gespeichert—eine gewöhnliche Matrix mit einem speziellen Namen. Jede Zeile der Matrix enthält einen Datenpunkt, der im vorliegenden Beispiel die Jahreswerte von VPI, EPI und AL beinhaltet.

Löschen Sie zuerst den Stackinhalt und wählen Sie FIX 2 als Zahlen-Anzeigeformat, bevor Sie mit der Berechnung beginnen.

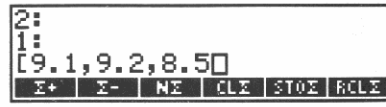
CLEAR
MODE 2 FIX

Löschen Sie evtl. zuvor gespeicherte Statistikdaten, indem Sie CLΣ drücken . (Eine existierende Variable ΣDAT wird dadurch gelöscht.)

STAT CLΣ

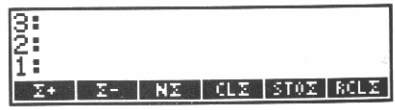
Tippen Sie die Daten für 1975 ein.

[9.1,9.2,8.5



Speichern Sie diese in ΣDAT.

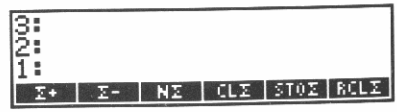
Σ+



Die Matrix ΣDAT wurde automatisch erzeugt. Die für 1975 gespeicherten Daten bilden die erste Zeile von ΣDAT.

Geben Sie die Daten für 1976 ein.

[5.8,4.6,7.7 Σ+



Diese Daten werden zu ΣDAT als zweite Zeile hinzugefügt.

Geben Sie die Daten für 1977 ein.

[ ] 6.5,6.1,7 Σ+

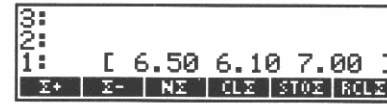
Die Daten für 1977 werden als dritte Zeile zur Statistikmatrix ΣDAT hinzugefügt.

Edieren von Daten

Wenn Ihnen beim Eintippen von Daten ein Fehler unterläuft und Sie dies noch vor dem Drücken von Σ+ bemerken, so können Sie einfach den Fehler noch in der Befehlszeile korrigieren. Wäre Ihnen der Fehler bereits bei den Daten für 1976 unterlaufen, dann können Sie den fehlerhaften Datenpunkt (d.h. die jeweilige Zeile der Statistikmatrix) in den Stack zurückrufen, die Werte richtig stellen und danach wieder in ΣDAT zurückspeichern.

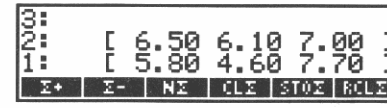
Entnehmen Sie den Datenpunkt für 1977 (letzte Zeile in ΣDAT) und geben Sie ihn in den Stack zurück.

Σ-



Entnehmen Sie den Datenpunkt für 1976 (letzte Zeile in ΣDAT) und geben Sie sie ihn den Stack zurück.

Σ-



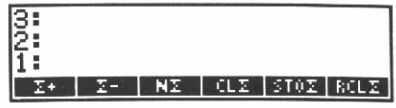
Wenn Ihnen für diese Punkte ein Fehler unterlaufen ist, so drücken Sie ■ EDIT, um den Punkt in die Befehlszeile zurückzugeben; Sie können die Werte nun ändern und durch Drücken von ENTER wieder im Stack speichern (siehe auch "Edieren gespeicherter Objekte" in Kapitel 18).

Speichern Sie den korrekten Datenpunkt für 1976 in ΣDAT zurück.

Σ+

Speichern Sie den Datenpunkt für 1977 in ΣDAT zurück.

Σ+



Geben Sie nun den Rest der Daten (für 1978 und 1979) ein.

[7.6,7.8,6 Σ+
[11.5,19.3,5.8 Σ+
NΣ

Statistische Funktionen für Einzelwerte

In diesem Abschnitt wird gezeigt, wie Sie für die Daten des vorangegangenen Beispiels den Mittelwert, die Standardabweichung und die Varianz berechnen können. Die Daten für VPI sind in der ersten Spalte von ΣDAT gespeichert, die Werte für EPI in der zweiten Spalte und die Daten für AL in der dritten Spalte.

Anzeige der zweiten Zeile des STAT Menüs.

NEXT

Diese Menüzeile enthält die Befehle zum Berechnen des Mittelwerts (*MEAN*), der Standardabweichung (*Standard DEViation*) und der Varianz.

Berechnen des Mittelwerts

Berechnen Sie den Mittelwert.

MEAN

Der Mittelwert für VPI beträgt 8.1, für EPI liegt er bei 9.4, und für AL beträgt er 7.

Berechnen der Standardabweichung

Berechnen Sie die Standardabweichung.

SDEV

Die Standardabweichung (einer Stichprobe) für VPI beträgt 2.27, für EPI liegt sie bei 5.8, und für AL beträgt sie 1.14.

Berechnen der Varianz

Berechnen Sie die Varianz.

VAR

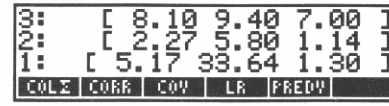
Die Varianz ergibt sich somit für VPI zu 5.17, für EPI zu 33.64, und für AL zu 1.3.

Statistische Funktionen für gepaarte Stichprobenwerte

Sie erfahren in diesem Abschnitt, wie Sie die Korrelation und die Kovarianz für die Werte aus dem vorangehendem Beispiel auffinden können. Anhand einer linearen Regression wird anschließend aufgezeigt, wie Werte für EPI durch Werte von VPI vorhergesagt werden können.

Anzeige der dritten Zeile des STAT Menüs.

NEXT



In diesem Menü finden Sie die Befehle für die Korrelation (*CORRelation*), Kovarianz (*COVariance*), lineare Regression und für einen Vorhersagewert (*PREDicted Value*).

Festlegen eines Spaltenpaares

Vor Berechnungen mit gepaarten Stichprobenwerten sind die Spalten der Matrix ΣDAT zu spezifizieren, welche die abhängigen und unabhängigen Daten enthalten. Im vorliegenden Beispiel sollen Werte für EPI (basierend auf den Werten von VPI) vorhergesagt werden, womit VPI als unabhängige und EPI als abhängige Spalte zu definieren ist.

Spezifizieren Sie Spalte 1 und 2 als unabhängige bzw. abhängige Daten.

1,2 COLΣ

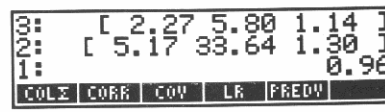
Die Zahlen 1 und 2 werden in der Liste ΣPAR (eine gewöhnliche Liste mit einem besonderen Namen) gespeichert. Befehle, welche statistische Berechnungen mit gepaarten Werten ausführen, beziehen sich auf die Variable ΣPAR.

Wenn Sie keine Spalten für die unabhängigen und abhängigen Daten spezifizieren, verwendet der Rechner die Standardwerte 1 und 2 als Parameterwerte für die Matrixspalten. In diesem Beispiel wäre die explizite Spezifikation der Spalten also nicht nötig gewesen; Sie sollten jedoch an die Ausführung von COLΣ denken, wenn Ihre relevanten Daten sich nicht in den Spalten 1 und 2 befinden.

Berechnen des Korrelationskoeffizienten

Berechnen Sie die Korrelation.

CORR



Die Korrelation zwischen VPI und EPI beträgt 0.96.

Berechnen der Kovarianz

Berechnen Sie die Kovarianz der Stichprobenwerte.

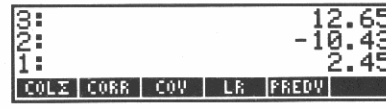
COV

Die Kovarianz von VPI und EPI ist 12.65.

Berechnen der linearen Regression

Berechnen Sie die Gerade, welche die beste Anpassung an die Daten von VPI und EPI darstellt.

LR



Der Schnittpunkt (−10.43) und die Steigung (2.45) werden ebenfalls in der Liste ΣPAR gespeichert.

Berechnen von Vorhersagewerten

Nehmen Sie an, Sie möchten die Werte für EPI vorausschätzen, wenn für VPI die Werte 6 und 7 vorliegen. Der Vorhersagewert kann über die in ΣPAR gespeicherte Steigung der Regressionsgeraden und den Ordinatenschnittpunkt berechnet werden.

Sagen Sie den Wert für EPI vorher, wenn VPI den Wert 6 annimmt.

6 PREDV

Der vorhergesagte Wert liegt bei 4.26.

Ermitteln Sie den Vorhersagewert für EPI, wenn der VPI bei 7 liegt.

7 PREDV

Der vorhergesagte Wert liegt bei 6.71.

13

Binäre Arithmetik

Sie erfahren in diesem Kapitel, wie auf dem HP-28S arithmetische Operationen auf Binärwerte angewendet werden können. Jeder Binärwert kann bis zu 64 Bits enthalten und stellt eine vorzeichenlose Binärzahl dar. Zur Vereinfachung bei der Eingabe und beim Ansehen der Ergebnisse besteht die Wahlmöglichkeit zwischen dezimaler, hexadezimaler, oktaler und dualer Zahlenbasis. Die von Ihnen getroffene Wahl beeinflußt allerdings nicht die interne Darstellung der Binärwerte, und Befehle wirken Bit-für-Bit auf die Binärwerte.

Alle Befehle im BINARY Menü sind kurz in Anhang B, "Menütabellen", beschrieben. Eine detailliertere Beschreibung finden Sie unter "BINARY" im Referenzhandbuch.

Wählen der Wortlänge

Die momentane Wortlänge beeinflußt die Länge von Binärwerten, die durch Befehle zurückgegeben werden, sowie die Anzeige von Binärwerten im Stack. Die Wortlänge liegt zwischen 1 und 64 Bits, wobei die Voreinstellung 64 Bits beträgt. Um z.B. eine Wortlänge von 16 Bits einzustellen, ist folgende Operation erforderlich:

Löschen Sie den Stackinhalt und rufen Sie das BINARY Menü auf, bevor Sie mit dem Beispiel beginnen.

■ CLEAR ■ BINARY

Spezifizieren Sie eine Wortlänge von 16 Bits.

16 STWS

Wenn Sie nun einen Binärwert eintippen, dessen Darstellung mehr als 16 Bits erfordert, so werden nur die 16 niederwertigsten Bits angezeigt.

Wählen des Zahlensystems

Das gewählte Zahlensystem beeinflußt die Anzeige der von Ihnen eingetippten Binärwerte im Stack. Sie haben die Wahlmöglichkeit zwischen dem dezimalen, hexadezimalen, oktalen und dualen Zahlensystem, wobei das Dezimalsystem als Voreinstellung benutzt wird.

Wählen Sie z.B. das hexadezimale Zahlensystem.

HEX

Das Menüfeld für HEX enthält nun ein kleines Quadrat zur Kennzeichnung, daß dieses Zahlensystem (Basis) gewählt wurde.

Eingeben von Binärwerten

Geben Sie die Adresse $24FF_{16}$ ein.

24FF ENTER

Das kleine "h" dient als *Basiskennzeichen*, um das momentan gewählte Zahlensystem anzudeuten. Die Angabe des Basiskennzeichens ist bei der Zahleneingabe nicht erforderlich, *außer die Zahl bezieht sich nicht auf die momentane Basis.*

Überprüfen Sie, wie dieser Wert in einem anderen Zahlensystem dargestellt wird. Sie müssen dazu lediglich die gewünschte Basis einstellen.

Wechsel zur dezimalen Basis.

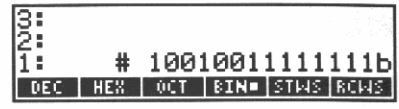
DEC

Wechsel zur oktalen Basis.

OCT

Wechsel zur binären Basis.

BIN



Rückkehr zur hexadezimalen Basis.

HEX

Rechnen mit Binärwerten

Berechnen Sie die Adresse, welche um $1F0_{16}$ kleiner als die vorgegebene Adresse ist.

[#] 1F0 [−]

Die Differenz wird in Stackebene 1 angezeigt.

Sie können Binärwerte und reelle Zahlen gleichzeitig in Ihren Berechnungen verwenden. Eine reelle ganze Zahl (ohne das Begrenzungszeichen # eingegeben) wird immer als Dezimalzahl interpretiert.

Berechnen Sie z.B. die um 27_{10} kleinere Adresse.

27 [−]

Die Differenz wird danach als Binärwert in Stackebene 1 angezeigt.

14

Konvertieren von Einheiten

Ziel dieses Kapitels ist die Veranschaulichung von Konvertierungen zwischen verschiedenen Einheiten. Anhand mehrerer Beispiele wird gezeigt, wie Sie den Zahlenwert einer physikalischen Größe von einer Einheit in eine äquivalente andere Einheit konvertieren können. Detaillierte Informationen hierzu finden Sie unter "UNITS" im Referenzhandbuch.

Der UNITS-Katalog

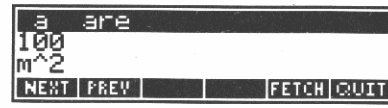
Sämtliche im HP-28S eingebaute Einheiten sind alphabetisch in einem Katalog mit dem Namen UNITS gelistet.

Löschen Sie zuerst den Stackinhalt und wählen Sie STD als Zahlen-Anzeigeformat.

■ CLEAR ■ MODE STD

Rufen Sie den UNITS-Katalog auf.

■ UNITS



Als erste Einheit wird "are", mit der Abkürzung "a", angezeigt. Diese Einheit wird für Flächenangaben verwendet und entspricht 100 m^2.

"Blättern" Sie den Katalog in beide Richtungen durch, indem Sie die Menütasten NEXT und PREV gedrückt halten.

Sie können den ersten Eintrag unter einem bestimmten Buchstaben aufrufen, indem Sie die gewünschte Buchstabentaste drücken.

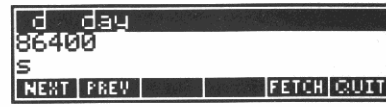
S

Aus der Eintragung wird ersichtlich, daß die richtige Abkürzung "s" lautet, der vollständige Name "second" heißt und daß der Wert eine Sekunde beträgt.

Stellen Sie immer sicher, daß Sie die korrekte Abkürzung—wie sie im UNITS-Katalog dargestellt ist—verwenden. So erkennt der HP-28S z.B. das kleingeschriebene "g" als Gramm, das großgeschriebene "G" jedoch wird als falsche Einheit zurückgewiesen.

Sehen Sie sich als nächstes den Eintrag für "day" an.

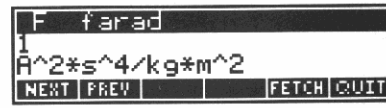
D



Sie erkennen, daß die korrekte Abkürzung "d" lautet, die vollständige Bezeichnung "day" ist und daß der Wert 86 400 Sekunden beträgt.

Rufen Sie als nächstes den Eintrag für "foot" auf.

F



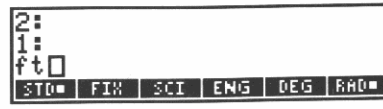
Der Katalog zeigt an dieser Stelle die Einheit "farad" an. Gehen Sie sieben Einträge weiter.

NEXT NEXT NEXT NEXT
NEXT NEXT NEXT

Sie erhalten den Eintrag für "international foot" angezeigt. Es sind zwei Versionen von "foot" im Katalog enthalten—der nächste Eintrag ist "survey foot".

Sie können auch die Abkürzung für "international foot" in die Befehlszeile übernehmen.

FETCH



Sie erhalten wieder die normale Anzeige und in der Befehlszeile wird die Abkürzung der Einheit angezeigt.

Die nachfolgenden Beispiele sollen verdeutlichen, wie Sie Einheiten direkt eintippen; falls Sie es vorziehen, können Sie aber auch ■ UNITS und FETCH verwenden.

Löschen Sie die Befehlszeile.

ON

Konvertieren von Einheiten

Es sollen zuerst 15 °C in die entsprechenden Grad Fahrenheit umgewandelt werden.

Speichern Sie den Zahlenwert im Stack.

15 ENTER

Geben Sie die Abkürzung für "Grad Celsius" ein.

■ ° C ENTER

Die Abkürzung der Einheit wird in einen Namen konvertiert.

Geben Sie die Abkürzung für "Grad Fahrenheit" ein.

■ ° F ENTER

Die Abkürzung der Einheit wird in einen Namen konvertiert.

Konvertieren Sie den Zahlenwert von der alten in die neue Einheit.

■ CONVERT

Das Ergebnis zeigt, daß 15 °C nach der Konvertierung 59 °F ergeben.

Im nächsten Beispiel sollen 40 Inches in Millimeter umgewandelt werden. Diesmal werden Sie durch Drücken von ■ CONVERT automatisch die ENTER Funktion für Sie ausführen lassen.

Löschen Sie den Stackinhalt und geben Sie den Zahlenwert ein.

■ CLEAR
40 ENTER

Geben Sie die Abkürzung für "Inches" ein.

LC in ENTER

Tippen Sie die Abkürzung für "Millimeter" ein und konvertieren Sie den Zahlenwert in die neue Einheit.

"Millimeter" ist nicht im UNITS-Katalog enthalten. Es handelt sich hierbei um eine Einheit mit *Vorsatz*—die Einheit `m` (für Meter), der ein `m` (für milli, oder ein Tausendstel) vorangestellt ist. Ähnlich verhält es sich mit `km` (für Kilometer) oder bei `ms`, was die Abkürzung für Millisekunden darstellt. Eine Liste mit allen zulässigen Vorsätzen finden Sie unter "UNITS" im Referenzhandbuch.

LC mm ■ CONVERT

Das Ergebnis zeigt, daß 40 Inches 1016 Millimeter entsprechen.

Konvertieren von Einheiten-Strings

Strings stellen Objekte dar, die eine von Ihnen spezifizierte Zeichenkette enthalten. Sie können *Einheiten-Strings* dazu verwenden, um komplexere Einheiten zu definieren.

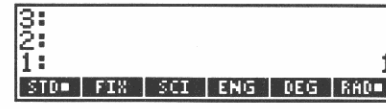
Ein Einheiten-String kann eine Einheit mit einem Exponenten enthalten (z.B. `"m^3"`), oder das Produkt zweier Einheiten (z.B. `"W*s"`), oder beliebige Kombinationen von Produkten und Potenzen.

Ein Einheiten-String kann ebenso einen Quotienten von Einheiten enthalten, wie z.B. "m/s". Allerdings darf das Symbol / nur einmal im String vorkommen. Stellen Sie sicher, daß Sie alle direkten Einheiten *vor* dem Divisionssymbol / und alle inversen Einheiten *nach* dem / gruppiert haben. So wird zum Beispiel "Meter pro Sekunde pro Sekunde" durch den Ausdruck "m/s^2"dargestellt.

Konvertieren Sie z.B. 1 Meile/Stunde in Meter/Sekunde.

Löschen Sie den Stackinhalt und geben Sie den Zahlenwert ein.

[■] [CLEAR]
1 [ENTER]



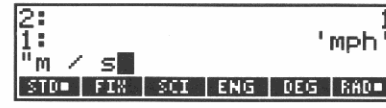
Eingabe der Abkürzung für "Meilen pro Stunde".

[LC] mph [ENTER]

Tippen Sie die Abkürzung für "Meter pro Sekunde" ein.

Es gibt keine interne Einheit für "Meter pro Sekunde", weshalb ein Einheiten-String zu verwenden ist.

[■] ["] [LC] m [÷] s



Es wurde der Alpha-Eingabemodus aktiviert, was durch die Form des Cursors angezeigt wird. In diesem Modus werden alle Befehle in die Befehlszeile geschrieben und Sie müssen daher [ENTER] drücken, um den String abzuschließen.

[ENTER]

Konvertieren Sie den Zahlenwert in die neue Einheit.

[■] [CONVERT]

Das Ergebnis zeigt, daß 1 Meile pro Stunde gleichwertig zu 0.44704 Meter pro Sekunde ist.

Im folgenden Beispiel sollen 10 Kubikmeter in US-Gallonen umgerechnet werden.

Löschen Sie den Stackinhalt und geben Sie den Zahlenwert ein.

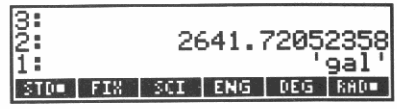
■ CLEAR
10 ENTER

Geben Sie den Einheiten-String für "Kubikmeter" ein.

■" LC m ■^ 3 ENTER

Tippen Sie die Abkürzung für "US-Gallon" ein und konvertieren Sie den Wert.

LC gal ■ CONVERT



Das Ergebnis zeigt, daß 10 Kubikmeter 2641.72052358 US-Gallonen entsprechen.

Überprüfen der verwendeten Einheiten

Die Verwendung von unkorrekten Einheiten kann zu unerwarteten numerischen Ergebnissen oder zu der Fehlermeldung `Inconsistent Units` führen. Sie lösen das Problem, indem Sie im UNITS-Katalog oder unter "UNITS" im Referenzhandbuch nach der richtigen Notation/Einheit nachsehen.

Unerwartete numerische Ergebnisse können dann auftreten, wenn Sie eine Einheit der richtigen Dimension verwenden, deren Zahlenwert jedoch unkorrekt ist. Wenn Sie z.B. Kubikmeter in "Gallonen" umrechnen, sollten Sie beachten, daß es neben "US-Gallonen" noch 2 ähnliche Einheiten gibt ("UK-Gallonen" und "Kanadische Gallonen").

Die Fehlermeldung `Inconsistent Units` wird dann angezeigt, wenn Sie zwei Einheiten spezifiziert haben, die in ihren Dimensionen nicht übereinstimmen. Dies kann z.B. vorkommen, wenn Sie die Einheit "S" (Siemens) als Zeiteinheit verwenden möchten. Die richtige Abkürzung für eine Zeiteinheit wäre "s" (Sekunden).

Benutzerfunktionen zum Konvertieren von Einheiten

Wenn Sie häufig bestimmte Konvertierungen ausführen, können Sie eine Funktion dafür definieren. Sie werden nachfolgend die Benutzerfunktionen O→G und G→O schreiben, welche zwischen Ounces und Gramm umrechnen; da es sich um Benutzerfunktionen handelt, können Sie sie in UPN oder algebraischer Syntax verwenden.

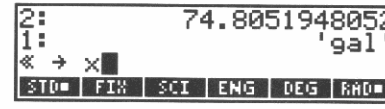
Beachten Sie, daß Benutzerfunktionen zwei Anforderungen erfüllen müssen:

- Sie müssen explizit ihr Argument festlegen.
- Sie müssen genau ein Ergebnis zurückgeben.

Schreiben Sie zuerst O→G.

Beginnen Sie das Programm und legen Sie das Argument fest.

« → LC x



Der Rechtspfeil zeigt an, daß die folgende Variable eine *lokale Variable* darstellt, die nur in diesem Programm vorkommt.

Definieren Sie die Konvertierung.

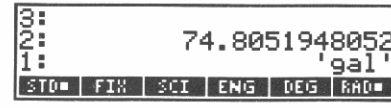
« x ' oz ' ' g ' CONVERT DROP ENTER

Die schließenden Begrenzungszeichen werden automatisch hinzugefügt.

Dieses Programm bedeutet: Nimm ein Argument vom Stack (in UPN Syntax) oder vom Ausdruck (in algebraischer Syntax) und benenne es als *x*; konvertiere *x* von Ounces in Gramm und lösche die Gramm-Einheit im Stack.

Speichern Sie das Programm in der Variablen O→G.

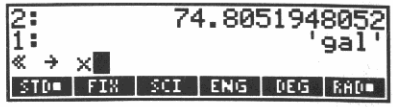
' O → G STO



Schreiben Sie nun G→O.

Beginnen Sie das Programm und legen Sie das Argument fest.

« ■→ LC x



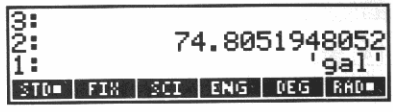
Definieren Sie die Konvertierung.

« x ' g ' ' oz ' ■ CONVERT DROP ENTER

Dieses Programm bedeutet: Nimm ein Argument vom Stack (in UPN Syntax) oder vom Ausdruck (in algebraischer Syntax) und benenne es als *x*; konvertiere *x* von Gramm in Ounces und lösche die Ounce-Einheit im Stack.

Speichern Sie das Programm in der Variablen G→O.

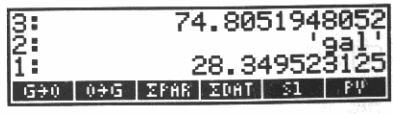
' G ■→ O STO



Um das Programm zu testen, rechnen Sie 1 Ounce in Gramm um und danach zurück in Ounces. Sie sollten wieder 1 erhalten.

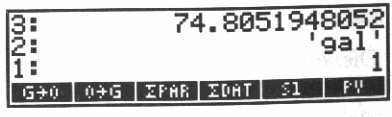
Konvertieren von 1 Ounce in Gramm.

1 USER O→G



1 Ounce enthält etwa 28 Gramm. Konvertieren Sie dies zurück in Ounces.

G→O



Die Konvertierungen sind—wie erwartet—invers.

15

Druckfunktionen

Sie lernen in diesem Kapitel einige Grundbefehle kennen, wie Sie Ihren HP-28S in Zusammenhang mit dem Drucker HP 82240A einsetzen können. Beziehen Sie sich zwecks Ausrichtung von Drucker und Rechner sowie zur Bedienung des Druckers auf die Drucker-Bedienungsanleitung.

Alle Befehle im PRINT Menü sind kurz in Anhang B, "Menütabellen", beschrieben. Eine vollständige Beschreibung finden Sie unter "PRINT" im Referenzhandbuch.

Drucken des Anzeigeinhalts

Sie können den Inhalt der Anzeige wie folgt ausdrucken:

1. Halten Sie [ON] gedrückt.
2. Drücken Sie [L] (Taste, über welcher "PRINT" steht).
3. Geben Sie [ON] wieder frei.

Diese Tastenfolge entspricht dem Befehl PRLCD (*PRint LCD*, in der ersten Zeile des PRINT Menüs). Sie können diese Tastenfolge praktisch jederzeit verwenden, ohne die Rechneroperationen zu unterbrechen.

Soll innerhalb eines Programms der Anzeigeinhalt gedruckt werden, so nehmen Sie einfach den Befehl PRLCD mit in das Programm auf.

Löschen Sie den Stackinhalt und rufen Sie das PRINT Menü auf.

[CLEAR]
[PRINT]

- PR1 (*PRint 1*) druckt die Objekte in Ebene 1.
- PRST (*PRint STack*) druckt alle im Stack gespeicherten Objekte.
- PRVAR (*PRint VARiable*) druckt Name und Inhalt von Variablen.
- PRLCD (*PRint LCD*) druckt den Inhalt der Anzeige.
- CR (*Carriage Right*) druckt eine Leerzeile.
- TRAC (*TRACe on/off*) schaltet den TRACE-Modus (für Protokolldruck) ein und aus.

Drucken eines Protokolls

Um eine Art "Protokoll" Ihrer fortlaufenden Berechnungen zu erstellen, können Sie einen entsprechenden Druckmodus spezifizieren.

TRAC

Ein kleines Quadrat erscheint im Menüfeld für TRAC, um den TRACE-Modus zu kennzeichnen.

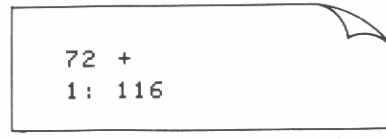
Beachten Sie nun, was nach der Addition zweier Zahlen—z.B. 44 und 72—erfolgt. Speichern Sie zuerst 44 im Stack.

44 ENTER

Ihre Eingabe sowie der Inhalt von Ebene 1 wird gedruckt.

Addieren Sie nun 72.

72 +



Erneut wird die Eingabe sowie der Inhalt von Ebene 1 gedruckt.

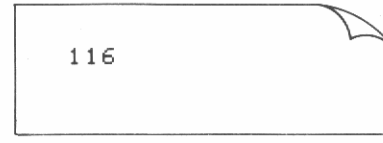
Schalten Sie den TRACE-Modus wieder aus.

TRAC

Drucken des Inhalts von Ebene 1

Anstatt alle Ergebnisse über den TRACE-Modus auszudrucken, können Sie auch selektiv über PR1 die Ergebnisse drucken.

PR1



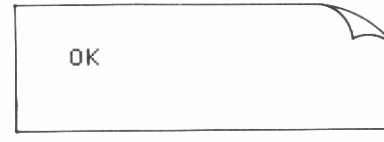
Das Ergebnis bleibt in Ebene 1—unverändert.

Sie können auch eine Nachricht bzw. einen Kommentar ausgeben, indem Sie den entsprechenden String in Ebene 1 eintippen und PR1 ausführen. Um z.B. die Meldung "OK" zu drucken, ist der entsprechende String zuerst im Stack zu speichern.

" OK ENTER

Drucken Sie nun die Meldung.

PR1



Es wird nur der Inhalt des Strings gedruckt, nicht die Begrenzungszeichen.

Drucken des Stackinhalts

Mit Hilfe des Befehls PRST lassen sich alle im Stack gespeicherten Objekte ausdrucken.

PRST

```
2: 116
1: "OK"
```

Der Stackinhalt bleibt dabei unverändert.

Drucken einer Variablen

Sie können den Inhalt einer Variablen ausdrucken, ohne daß Sie dazu die Variable selbst in den Stack zurückrufen. Um diese Möglichkeit zu veranschaulichen, speichern Sie den String "OK" in einer Variable mit dem Namen A und drucken Sie anschließend den Inhalt von A.

Erzeugen Sie die Variable A mit dem Inhalt "OK".

' A STO

```
3:
2:
1:                              116
PR1 PRST PRVAR PRLCD CR TRAC
```

Drucken Sie den Namen und den Inhalt der Variablen aus.

' A PRVAR

```
A
"OK"
```

Der Name der Variablen wird wieder vom Stack genommen, d.h. gelöscht.

Teil 2

Zusammenfassung der Rechnerfähigkeiten

16

Objekte

Teil 1 dieses Handbuchs enthält Anwendungsbeispiele für die 10 Basisobjekte im HP-28S. Objekte stellen die grundlegenden Elemente dieses Rechnertyps dar—sie dienen zur Formulierung der Problemstellung und zum Auffinden von Lösungen.

Der Zweck der meisten Objekttypen liegt in der Zeitersparnis durch die Bereitstellung spezieller Datentypen. Stellen Sie sich z.B. vor, reelle Zahlen für Felder zu verwenden, wobei die Kontrolle über die Feldelemente beizubehalten ist und Programme für arithmetische Berechnungen mit diesen Feldern zu schreiben sind. Es ist einfacher, die Zahlen in ein Feld-Objekt einzugeben, welches Sie als einzelne Größe behandeln können, und die Berechnungen über die normalen arithmetischen Funktionen auszuführen.

Der Grund für mehrere Objekttypen liegt tiefer als nur in der Verfügbarkeit verschiedener Datentypen. Die symbolischen und programmierbaren Fähigkeiten basieren auf symbolischen Objekten (Namen und algebraische Ausdrücke) und Programm-Objekten. Diese stellen nicht nur Daten dar—sie können ausgewertet werden und können ein Ergebnis erzeugen. (Die Auswertung von Objekten ist in Kapitel 23 behandelt.)

Unter Grundlage des einfachen Konzepts der Objekttypen reduziert der HP-28S die Regeln, an welche Sie sich erinnern müssen. Objekte werden in die Befehlszeile, den Stack oder eine Variable immer in der gleichen Weise eingetippt, unabhängig vom Objekttyp.

Dieses Kapitel faßt zusammen, was Sie über jeden Typ gelernt haben, gibt detailliertere Hinweise und schlägt zusätzliche Anwendungsmöglichkeiten vor.

Reelle Zahlen

Reelle Zahlen stellen Werte größer als -10^{500} und kleiner als 10^{500} dar. Sie werden intern als *Mantisse* zwischen 1 und 9.99999999999, einem Vorzeichen für die Mantisse, einem *Exponenten* zwischen 0 und 499 und dessem Vorzeichen gespeichert.

Im Stunden-Minuten-Sekunden Format. Sie können die korrespondierenden Befehle HMS+ und HMS− zum Addieren und Subtrahieren von Zahlen, welche in diesem Format dargestellt sind, verwenden. Für alle Berechnungen außer Addition und Subtraktion ist zuerst HMS→ zur Konvertierung vom HMS Format in Dezimalgrad-Format auszuführen. (Detaillierte Informationen finden Sie unter "TRIG" im Referenzhandbuch.)

Komplexe Zahlen

Komplexe-Zahlen-Objekte sind geordnete Paare von reelle Zahlen, welche den *Real-* und *Imaginärteil* einer komplexen Zahl oder die Koordinaten eines Punkts in der Ebene darstellen.

Rechtecks- und Polarkoordinaten. In Kapitel 7 und 8 haben Sie komplexe Zahlen für grafische Darstellungen verwendet; jede komplexe Zahl stellte *Rechteckskoordinaten* dar, d.h. Strecken in einem kartesischen Koordinatensystem.

Kapitel 6 erläuterte *Polarkoordinaten*—ein Radius und ein Winkel—und zeigte die Anwendung von R→P und P→R zum Konvertieren zwischen den unterschiedlichen Koordinatensystemen. Polarkoordinaten können zur Eingabe und Ergebnisdarstellung verwendet werden, sämtliche Berechnungen müssen jedoch über Rechteckskoordinaten stattfinden. Die Benutzerfunktion PSUM addiert Punkte in Polarnotation, indem zuerst konvertiert, dann addiert und zurückkonvertiert wird.

In algebraischen Objekten. Beim Eintippen einer komplexen Zahl in ein algebraisches Objekt können zwei Klammern erforderlich sein, wie z.B. `'SIN((0,1))'`. Das äußere Paar wird von der Funktion `SIN( )` benötigt, während das innere Paar Begrenzungszeichen für komplexe Zahlen darstellt.

Binärwerte

Binärwerte stellen eine Sequenz von Bits dar. Die Länge der Sequenz, von 1 bis 64, hängt von der momentanen Wortlänge ab. Die Anzeige von Binärwerten wird durch das spezifizierte *Zahlensystem* gesteuert, welche jedoch keinen Einfluß auf die interne Darstellung hat.

Große Binärwerte. Die Verwendung von Binärwerten im Dezimalsystem ermöglicht die genaue Darstellung einer 19-stelligen positiven ganzen Zahl, was 7 Stellen mehr entspricht als bei Verwendung reeller Zahlen.

Programmbeispiele: Die Programme in "Anzeigen von Binärwerten" auf Seite 257 erlauben die Anzeige eines Binärwerts in allen vier Zahlensystemen.

Sicherungsstatus. Der Befehl RCLF (*ReCalL Flags*) gibt einen Binärwert zurück, welcher den Status aller 64 Benutzerflags darstellt. STOF (*STOre Flags*) setzt Benutzerflags in Abhängigkeit zu einem binären Argument. Diese Befehle werden in "PRESERVE (Sichern und Wiederherstellen des vorherigen Status)", einem der Programme in "Anzeigen von Binärwerten", demonstriert.

Strings

Ein *String* enthält eine Sequenz von Zeichen. Teil 1 zeigte folgende Anwendungen für Strings.

- In Kapitel 14, "Konvertieren von Einheiten", haben Sie Strings zur Darstellung von Einheiten-Kombinationen und -Potenzen verwendet.
- In Kapitel 15, "Druckfunktionen", diente ein String zum Ausdrukken einer Meldung. Sie können mit DISP auch Meldungen anzeigen, wie in Kapitel 27, "Interaktive Programme", beschrieben.

Meistens enthält ein String Textinformationen; jedes Zeichen kann aber auch einen numerischen Wert zwischen 0 und 255 darstellen. Die Befehle CHR (*CHaraCter*) und NUM (*character NUMber*) konvertieren zwischen Zeichen und deren numerischen Werten.

Zeichen zusätzlich zum Tastenfeld. Sie können Zeichen anzeigen, welche nicht über das Tastenfeld zugänglich sind, indem Sie deren numerischen Wert eintippen und CHR ausführen. Außerdem gibt es nicht-anzeigbare Zeichen, welche gedruckt werden können. Eine Auflistung aller Zeichen finden Sie unter "STRING" im Referenzhandbuch.

Grafik-Strings. LCD→ (*LCD in String*) gibt einen Grafik-String zurück, der den momentanen Anzeigeinhalt darstellt; der Befehl →LCD (*String in LCD*) zeigt den Inhalt eines Grafik-Strings an.

String-Manipulationen. Die Programme "Anzeigen von Binärwerten", auf Seite 257, zeigen die Konvertierung eines Objekts in einen String, das Bestimmen der Stringgröße sowie das Zusammenfügen zweier Strings.

Felder

Felder können aus eindimensionalen (*Felder*) oder zweidimensionalen (*Matrizen*) Objekten bestehen und reelle oder komplexe Zahlen enthalten. Kapitel 11, "Vektoren und Matrizen", zeigt die grundlegenden Berechnungen mit Feldern. Teil 1 enthielt die folgenden zusätzlichen Anwendungen von Feldern:

- Kapitel 11 zeigt die Lösung eines Systems mit n linearen Gleichungen unter Verwendung eines n-elementigen Konstantenvektors und einer $n \times n$ Koeffizientenmatrix. Details zu diesem Prozeß finden Sie unter "ARRAY" im Referenzhandbuch.
- In Kapitel 12, "Statistische Funktionen", wurden die eingegebenen Statistikdaten in der Statistikmatrix ΣDAT gespeichert.

In algebraischer Syntax. Wenn ein Feld in einer Variablen gespeichert wird, können Sie sich auf die Elemente beziehen, indem Sie den Variablennamen als Funktion verwenden. Zum Beispiel kann die Summe des dritten und fünften Elements von V als `'V(3)+V(5)'` dargestellt werden.

Manipulation von Feldern. Die Programme "Summations-Statistik" auf Seite 262, und "Median von statistischen Daten" auf Seite 270, demonstrieren eine Vielzahl von Feld-Manipulationen.

Listen

Listen stellen eine Sequenz von Objekten dar; sie sind das allgemeinste Verfahren zum Zusammenfassen mehrerer Objekte zu einem. Teil 1 erläuterte folgende Anwendungsmöglichkeiten:

- In Kapitel 4, "Wiederholen einer Berechnung", gab der Befehl PATH eine Liste von Verzeichnisnamen zurück.
- In Kapitel 7, "Grafische Darstellung", enthielt die Listenvariable PPAR Parameter, welche von DRAW benutzt wurden.
- In Kapitel 8, "Der Löser", wurde eine Liste mit drei digitalisierten Näherungswerten bearbeitet.
- In Kapitel 10, "Höhere Mathematik" spezifizierten Sie die Integrationsvariable und Integrationsgrenzen unter deren Zusammenfassung in einer Liste.
- In Kapitel 12, "Statistische Funktionen", enthielt die Listenvariable ΣPAR Parameter für Datenpaare.

In algebraischer Syntax. Wenn eine Liste in einer Variablen gespeichert wird, können Sie sich auf die Elemente beziehen, indem Sie den Variablennamen als Funktion verwenden. Zum Beispiel kann die Summe des dritten und fünften Elements von L als `'L(3)+L(5)'` dargestellt werden.

Listen und Stack. Das Programm MEDIAN auf Seite 273 zeigt, wie Listenelemente in den Stack übernommen und Objekte im Stack in einer Liste kombiniert werden können.

Sortieren einer Liste. Das Programm SORT auf Seite 270 demonstriert das Sortieren von Listenelementen.

Entnehmen von Listenelementen. Das Programm LMED auf Seite 272 zeigt, wie Elemente aus einer Liste entnommen werden.

Namen

Namen stellen eine Reihe von Zeichen dar, um andere Objekte zu benennen. Sie können bis zu 127 Zeichen enthalten, obwohl aus praktischen Gründen ein Name nicht länger als fünf bis sechs Zeichen sein sollte.

Die zulässigen Zeichen auf dem Tastenfeld sind Buchstaben, Zahlen und die Sonderzeichen ? Σ π → µ ". Das erste Zeichen darf keine Zahl sein. Die folgenden Zeichen sind für Namen ausgeschlossen:

- Zeichen, welche Objekte trennen: Begrenzungszeichen, (# [ ] " ' « » ( ) ⟨ ⟩ ≪ ≫), Leerzeichen, Punkt oder Komma.
- Algebraische Operatoren: + − * / ^ √ = < > ≤ ≥ ≠ ∂ ∫

Der Rechner erkennt bei der Verarbeitung der Befehlszeile, ob es sich um einen globalen oder lokalen Namen handelt: wird er in einem Programm zur Erzeugung einer lokalen Variablen verwendet, so ist er lokal innerhalb des Programms, ansonsten ist es ein globaler Name.

Lokale Namen. In Teil 1 haben Sie Benutzerfunktionen geschrieben, welche lokale Variablen erzeugten. In diesem Handbuch sind lokale Namen kleingeschrieben, um sie von globalen Namen zu unterscheiden. Es soll daran erinnert werden, daß der Befehl "→" die lokale Verwendung bewirkt, nicht die Kleinbuchstaben. Wenn Sie eine lokale Variable `e` oder `i` benennen, überschreibt die lokale Definition die interne Definition.

Globale Namen. Alle anderen Namen in Teil 1 waren global, wie zum Beispiel:

- Namen für globale Variablen (Zahlenwerte für grafische Darstellungen oder für den Löser; alle Variablen im USER Menü).
- Namen für Verzeichnisse.
- Symbolisch verwendete Namen, ohne Bezug zu speziellen Werten.

Befehlsnamen, einschließlich `e`, `i` und π, können nicht als globale Namen verwendet werden. Zusätzlich sind die folgenden Namen für spezielle Zwecke reserviert:

- EQ bezieht sich auf die momentane Gleichung für den Löser und PLOT Befehle.
- ΣPAR bezieht sich auf eine Parameterliste für Statistikbefehle.
- PPAR bezieht sich auf eine Parameterliste für PLOT Befehle.
- ΣDAT bezieht sich auf die momentane Statistikmatrix.
- s1, s2, und so weiter, werden von ISOL und QUAD zur Darstellung beliebiger Vorzeichen erzeugt (in symbolischen Lösungen).
- n1, n2, und so weiter, werden von ISOL zur Darstellung beliebiger ganzer Zahlen erzeugt (in symbolischen Lösungen).
- Namen, die mit "der" beginnen, beziehen sich auf benutzerdefinierte Ableitungen.

Programme

Programme stellen Sequenzen von Objekten und Befehlen dar. Jedes Programm ist im wesentlichen eine in ein Objekt verwandelte Befehlszeile, deren Ausführung auf später verschoben werden soll.

Spezielle Programmbefehle sind unter den Menüs für PROGRAM BRANCH, PROGRAM CONTROL und PROGRAM TEST gelistet. Eine Beschreibung dieser Menüs finden Sie im Referenzhandbuch, zusammen mit dem allgemeinen Thema "Programme".

In Teil 1 entwickelten Sie fünf Programme:

- In Kapitel 3 haben Sie ein Programm zum Umbenennen von Variablen geschrieben und es in der Variablen UMBENENNEN gespeichert.
- In Kapitel 5 entwickelten Sie ein Programm für die Cotangens-Funktion und speicherten es in COT.
- In Kapitel 6 haben Sie ein Programm zum Addieren von Polarkoordinaten geschrieben und in PSUM gespeichert.
- In Kapitel 14 entwickelten Sie Programme zum Konvertieren zwischen Ounces und Gramm und speicherten sie in den Variablen O→G und G→O.

Benutzerfunktionen. Die Programme COT, PSUM, O→G und G→O sind Benutzerfunktionen—Sie beginnen mit dem Befehl →, gefolgt von einem oder mehreren Namen, welche zusammen eine oder mehrere lokale Variablen definieren und welchen ein Ausdruck oder ein Programm folgt. Wird die Benutzerfunktion in einer Variablen gespeichert, so können Sie die Variable in algebraischen Ausdrücken verwenden, wie Sie es von internen Funktionen gewohnt sind.

Programmstrukturen. Der Befehl →, gefolgt von Namen und einem Ausdruck oder Programm, wird als *lokale Variablenstruktur* bezeichnet. Es gibt neben diesen auch noch Programmstrukturen für Verzweigungen (wie IF ... THEN ... ELSE ... END) und Schleifen (wie DO ... UNTIL ... END). Beziehen Sie sich auf Kapitel 26, "Programmstrukturen", für eine nähere Beschreibung. In Kapitel 28, "Programmbeispiele", sind 20 Programme enthalten, welche die Anwendung jeder Programmstruktur aufzeigen.

Namenlose Programme. Programme müssen nicht unbedingt in Variablen gespeichert werden. Sehen Sie dazu z.B. "Vollständiges Erweitern und Zusammenführen", auf Seite 253, und "Anzeigen eines Binärwerts", auf Seite 257.

Algebraische Terme

Algebraische Terme umfassen eine oder mehrere Funktionen und deren Argumente, welche aus Zahlen, Namen oder Teilausdrücken bestehen. Algebraische Terme werden in algebraischer Syntax geschrieben und angezeigt (ähnlich zur mathematischen Notation). Es gibt zwei Arten von Termen: Ausdrücke und Gleichungen.

Ausdrücke

In Teile 1 wurden Ausdrücke auf drei verschiedene Wege benutzt: als Daten, als Funktionen und als implizite Gleichungen.

Ausdrücke als Daten. Wenn Sie Berechnungen mit Ausdrücken ausführen, wie z.B. Addieren, Quadrieren, usw., so erhalten Sie wieder einen Ausdruck als Ergebnis. Hierbei dienen die Ausdrücke als zu modifizierende Daten, unabhängig von jeglichen Werten, welche den Variablen zugewiesen wurden.

Ausdrücke als Funktionen. In Kapitel 4 hatten Sie den Ausdruck RGES erzeugt und mit Hilfe des Lösers den Variablen bestimmte Werte zugeordnet. Danach erfolgte die Auswertung von RGES, um das gewünschte Ergebnis zu erhalten. Hier diente der Ausdruck als Funktion, die bei vorgegebenen Eingabewerten zum Ergebnis führte.

Ausdrücke als implizite Gleichungen. In Kapitel 8 diente der Löser zur Bestimmung der *numerischen* Nullstelle eines Ausdrucks, d.h. der Zahlenwert der unabhängigen Variablen, für welchen der Ausdruck den Wert Null annimmt. In Kapitel 9 wurde QUAD zum Auffinden einer *symbolischen* Nullstelle verwendet—ein Ausdruck, der, nach Substitution für die Unabhängige, für den ursprünglichen Ausdruck den Wert 0 ergibt.

In beiden Fällen wirkt der Ausdruck $f(x)$ wie die Gleichung $f(x) = 0$, da die *Nullstelle* des Ausdrucks der *Lösung* der Gleichung entspricht.

Gleichungen

Gleichungen umfassen zwei Ausdrücke, die durch ein Gleichheitszeichen in Bezug zueinander gesetzt werden. Mathematisch gibt es zwei Verwendungsmöglichkeiten für "=":

- Um eine Aussage wie "$x^2 = 4$" oder "$x^2 + y^2 = 1$" anzuzeigen. Hier trifft die Gleichung nur für einige Variablenwerte zu.
- Um eine Identität oder Definition wie "$\sin 2x = 2 \sin x \cos x$" oder "$y = 3x^2 + 2x + 5$" anzuzeigen. Hier trifft die Gleichung für alle Werte der Variablen zu.

Im HP-28S werden Gleichungen nur für Aussagen benutzt; um eine Definition wie z.B. "$y = 3x^2 + 2x + 5$" vorzunehmen, wird der Ausdruck `'3*X^2+2*X+5'` *in einer Variablen mit dem Namen* `Y` *gespeichert.*

Im Abschnitt "Finanzmathematische Berechnungen" auf Seite 103 sind ANNU und SPPV mathematisch als Gleichungen ausgedrückt. Die Annuitätengleichung wurde als Gleichung eingegeben, während SPPV, deren Wert durch ihre Variableninhalte bestimmt wird, als Variable erzeugt wurde.

Gleichungen als Daten. Wenn Sie Berechnungen mit Gleichungen durchführen (z.B. Addieren zweier Gleichungen), so erhalten Sie als Ergebnis eine weitere Gleichung. Jede Seite der Gleichung wird gesondert behandelt—jede Seite ist ein Ausdruck, der als Datum behandelt wird. Die Gleichung behält im Prinzip ihre Aussagekraft, wobei sie nur für einige Werte ihrer Variablen erfüllt ist.

Lösen von Gleichungen. Wenn Sie eine Gleichung numerisch lösen, so finden Sie den Wert der unabhängigen Variablen, welcher beide Seiten der Gleichung erfüllt. Ähnlich verhält es sich, wenn Sie eine Gleichung symbolisch lösen (wie bei "Isolieren einer Variablen" auf Seite 109); Sie erhalten einen Ausdruck, welcher nach Substitution für die unabhängige Variable die Gleichung erfüllen würde.

Symbolische Konstanten

Algebraische Terme können die folgenden symbolischen Konstanten beinhalten. Diese sehen wie Namen aus, sind aber eigentlich Funktionen:

- MINR (*MINimum Reell*) stellt die kleinste positive reelle Zahl dar. Ihr Zahlenwert ist 1.00000000000E−499.
- MAXR (*MAXimum Reell*) stellt die größte positive reelle Zahl dar. Ihr Zahlenwert ist 9.99999999999E499.
- e stellt die Basis des natürlichen Logarithmus dar. Ihr numerischer Wert im HP-28S ist 2.71828182846.
- π entspricht dem Verhältnis von Umfang und Durchmesser eines Kreises. Ihr numerischer Wert im HP-28S ist 3.14159265359.
- i stellt die Imaginärzahl $\sqrt{-1}$ dar. Ihr Zahlenwert ist (0, 1).

Im numerischen Konstanten- oder Ergebnis-Modus ergibt die Auswertung von symbolischen Konstanten deren numerischer Wert; ansonsten ergibt sich deren symbolische Form. (Konstanten- und Ergebnis-Modus sind in Kapitel 24 behandelt.)

17

Operationen, Befehle und Funktionen

Jede interne Prozedur des HP-28S kann als Operation, Befehl, Funktion oder analytische Funktion klassifiziert werden.

- Eine *Operation* ist jede interne Prozedur des Rechners.
- Ein *Befehl* ist eine programmierbare Operation.
- Eine *Funktion* ist ein in algebraischen Ausdrücken zulässiger Befehl.
- Eine *ananlytische Funktion* ist eine Funktion, für welche der HP-28S eine Ableitung und die Inverse bilden kann.

Interne Prozeduren werden gewöhnlich durch ihre wichtigste Fähigkeit charakterisiert. Zum Beispiel stellt SWAP einen Befehl und eine Operation dar, während IP Funktion, Befehl und Operation darstellt. SWAP wird jedoch als Befehl und IP als Funktion charakterisiert. Die folgende Tabelle zeigt Beispiele für jeden Typ.

Operationen

Nicht programmierbare operationen	Befehle		
	RPN Befehle	Funktionen	
		nicht analytisch	analytisch
INS	SWAP	ABS	ASIN
NEXT	DROP	∂	EXP
■ EDIT	LAST	IP	INV
■ VIEW↑	RCL	MAX	LN
ENTER	PURGE	OR	NEG
EEX	∫	%CH	SIN
■ COMMAND	STO	R→D	SINH
■ UNDO	EVAL	R→P	SQ
■ CONT	CLEAR	XPON	+
ON	CONVERT	≠	=

Das Verzeichnis der Operationen am Ende des Referenzhandbuchs kennzeichnet jede interne Prozedur als Operation, Befehl, Funktion oder analytische Funktion. Nachstehende Erläuterungen gelten als grobe Faustregel für jeden Typ.

- Die meisten nicht-programmierbaren Operationen können nur durch einen Tastendruck ausgeführt werden. Es gibt jedoch programmierbare Äquivalente für einige Operationen: z.B. kann die [TRIG] Operation (zur Wahl des TRIG Menüs) durch ein Programm herbeigeführt werden, indem `21 MENU` ausgeführt wird, und die RAD Operation (zur Spezifikation von Radianten als Winkelmodus) kann durch Ausführung von `60 FS` herbeigeführt werden.
- Die meisten UPN Befehle beinhalten Stack-Manipulationen oder Änderungen im Benutzerspeicher anstatt von Berechnungen mathematischer Werte.
- Die meisten nicht-analytischen Funktionen sind mathematische Berechnungen ohne Inverse—d.h. sie berechnen einige Charakteristiken der Argumente, aber die Argumente können nicht über die Ergebnisse rekonstruiert werden. Beispiele beinhalten den gebrochenen oder den ganzzahligen Teile einer Zahl, Absolutwert oder die Vorzeichenfunktion.
- In der Mathematik stellt eine Funktion von komplexen Variablen eine *analytische Funktion* dar, wenn sie als Potenzreihe an jedem Punkt ihres Wertebereichs ausgedrückt werden kann; in diesem Fall hat sie eine Inverse und eine Ableitung. Der HP-28S macht einige Ausnahmen zu dieser Definition. So gibt es z.B. keine Ableitung für den Befehl `%`, obwohl eine möglich wäre; andererseits gibt es eine Ableitung für die Funktion `ABS`, obwohl sie im Punkt $0 + 0i$ nicht analytisch ist.

Jede interne Prozedur ist über eine Taste verfügbar, entweder auf dem Tastenfeld oder in einem Menü. Wenn Sie eine Taste drücken, hängt die Genauigkeit des Ergebnisses vom Prozedurtyp und vom *Eingabemodus* ab, was im nächsten Kapitel diskutiert wird.

18

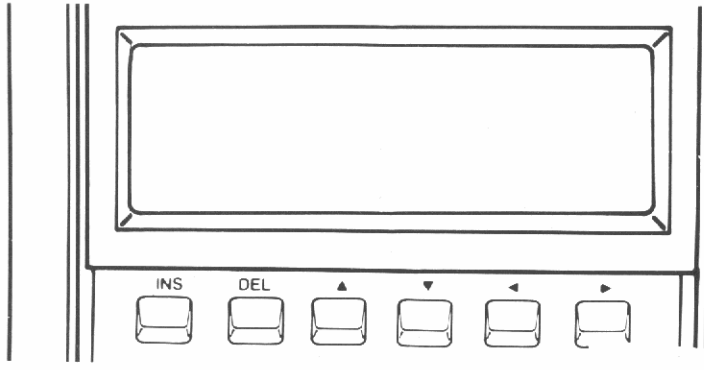
Die Befehlszeile

Die Befehlszeile kann eine beliebige Anzahl von Zeichen enthalten, welche Objekte in Textform darstellen. Sie erscheint am unteren Rand der Anzeige (über den Menüfeldern, falls diese vorhanden sind), wenn Sie mit der Eingabe eines Objekts beginnen oder wenn Sie ■ EDIT oder ■ VISIT zum Edieren eines existierenden Objekts drücken.

In der Befehlszeile können mehrere Textzeilen enthalten sein. Wenn Sie mehr als 23 Zeichen in eine Zeile eingeben, werden sie über den linken Anzeigerand hinaus verschoben. Eine Ellipse (...) erscheint in der linken Anzeigespalte, um die Existenz nicht dargestellter Zeichen anzudeuten. Sobald Sie versuchen, den Cursor über den linken Anzeigerand hinaus zu verschieben, werden die "unsichtbaren" Zeichen wieder sichtbar. Hierbei erscheint dann eine Ellipse am rechten Anzeigerand. Sind in der Befehlszeile mehrere Textzeilen enthalten, dann werden alle zusammen nach links oder rechts verschoben.

Das Cursor-Menü

Das Cursor-Menü ist ein spezielles Menü von Edieroperationen. Es ist immer aktiv, wenn die Befehlszeile vorhanden ist und keine Menüfelder angezeigt sind. Das Cursor-Menü enthält beides, umgeschaltete und normale Tasten. Letztere sind in weißer Beschriftung oberhalb der Menütasten angebracht, wie in der Abbildung dargestellt.



Wenn Sie eine normale Cursor-Menütaste gedrückt halten (außer INS), dann wird die entsprechende Operation so lange wiederholt, bis Sie die Taste wieder freigeben.

Taste	Bedeutung
INS	Schaltet zwischen Ersetzungs- und Einfügungsmodus (*INSert*) um. Im Ersetzungsmodus treten neue Zeichen an Stelle der ehemaligen Zeichen. Im Einfügungsmodus werden neue Zeichen zwischen die bereits bestehenden eingefügt.
DEL	Löscht (*DELete*) das Zeichen an der Cursorposition.
▲	Bewegt den Cursor um eine Zeile nach oben.
▼	Bewegt den Cursor um eine Zeile nach unten.
◀	Bewegt den Cursor um eine Position nach links.
▶	Bewegt den Cursor um eine Position nach rechts.

Die umgeschalteten Cursor-Menütasten (außer ■ INS) sind gleichwertig zum mehrfachen Drücken der einfachen Operationen.

Taste	Bedeutung
■ INS	Löscht alle Zeichen links des Cursors.
■ DEL	Löscht das Zeichen an der gegenwärtigen Cursorposition und alle Zeichen rechts davon.
■ ▲	Bewegt den Cursor zur obersten Reihe der Befehlszeile.
■ ▼	Bewegt den Cursor zur untertsten Reihe der Befehlszeile.
■ ◀	Bewegt den Cursor an das linke Ende der Befehlszeile.
■ ▶	Bewegt den Cursor an das rechte Ende der Befehlszeile.

Verschiedene Eingabetasten

Nachstehend beschriebene Tasten sind hilfreich beim Eingeben von Objekten in die Befehlszeile.

Taste	Bedeutung
◀▶	**Cursor-Menü ein/aus.** Bei desaktiviertem Cursor-Menü wird das Cursor-Menü aktiviert, im umgekehrten Fall wird es desaktiviert.
CHS	**Vorzeichenwechsel.** (*CHange Sign*). Steht der Cursor über einer Zahl, so wird deren Vorzeichen umgekehrt; steht er über einem anderen Zeichen, dann wird ein Minus-Zeichen eingefügt. (Falls keine Befehlszeile vorhanden ist, führen Sie NEG aus.)
EEX	**Eingabe Exponent.** Steht der Cursor bei einer Zahl ohne Exponent, so wird der Buchstabe E nach der Zahl eingefügt; steht der Cursor bei einer Zahl mit Exponent, so wird er hinter das E gestellt. Ist der Cursor nicht bei einer Zahl positioniert, dann wird 1E eingefügt.
◀	**Rückschritt-Taste.** Löscht das Zeichen links vom Cursor und bewegt diesen (sowie alle Zeichen rechts davon) eine Position nach links. Wenn Sie ◀ gedrückt halten, wird der Vorgang bis zum Freigeben der Taste wiederholt.
LC	**Kleinbuchstaben.** (*Lower Case*) Schaltet zwischen Groß- und Kleinschreibung um. Existiert die Befehlszeile, so wurde der Großschreibungsmodus aktiviert. Nach dem Drücken von LC werden die Buchstaben in der Form a bis z geschrieben.

■ MENUS **Menüsperre.** Schaltet die Menüsperre ein oder aus. Wurde die Menüsperre aktiviert, so werden die einfachen und umgeschalteten "Positionen" für die drei obersten Tastenreihen des linken Tastenfelds getauscht. Sie müssen z.B. für ARRAY bis UNITS nicht mehr zuerst ■ drücken; die Umschalttaste muß in dieser Situation für die Buchstaben A bis R gedrückt werden.

ON **Grundstellung.** Löscht die Befehlszeile.

Begrenzungs- und Trennzeichen für Objekte

Aufeinanderfolgende Objekte oder Befehle, welche in der Befehlszeile eingegeben werden, müssen durch folgende Zeichen voneinander getrennt werden:

- Ein Objekt-Begrenzungszeichen: « » [ ] { } # " ' « ».
- Ein Leerzeichen oder eine neue Zeile. Das Drücken von ■ NEWLINE fügt ein "Zeilenvorschub"-Zeichen an der momentanen Cursorposition ein. Bei der Ausführung der Befehlszeile sind diese Zeichen gleichwertig zu Leerzeichen.
- Ein Punkt oder ein Komma, je nachdem, welches Zeichen *nicht* als Dezimalzeichen verwendet wird.

Eingabemodi

Zur leichteren Eingabe verschiedener Objekte stehen drei Eingabemodi zur Verfügung—*unmittelbarer, algebraischer und Alpha-Eingabemodus*. Denken Sie an die Unterscheidungen, welche im vorherigen Kapitel angestellt wurden:

- *Operationen* sind nicht programmierbar.
- *Befehle* können in Programmen auftreten, aber nicht in algebraischen Ausdrücken.
- *Funktionen* und *Namen* können in Programmen und algebraischen Ausdrücken verwendet werden.

Der Rechner erkennt diese Unterschiede beim Eintippen der Objekte in die Befehlszeile. Das Drücken einer Operationstaste (wie z.B. ENTER) bewirkt immer die Ausführung der Operation. Der momentane Eingabemodus wirkt sich primär auf die Auswirkung nach dem Drücken einer Befehlstaste (wie z.B. STO), einer Funktionstaste (wie z.B. +) oder einer USER Menütaste aus.

Unmittelbarer Eingabemodus. Dieser dient zum Eingeben von Zahlen, Listen und Felder. Dieser Modus bedeutet:

- Eine Befehlstaste führt zuerst die Befehlszeile aus und anschliessend den Befehl.
- Eine Funktionstaste führ die Befehlszeile aus und anschließend die Funktion.
- Eine Variablentaste im USER Menü führt die Befehlszeile aus und wertet anschließend die Variable aus.

Algebraischer Eingabemodus. Dieser dient zum Eingeben von Namen und algebraischen Ausdrücken. Beginnen Sie die Befehlszeile mit ', so wird der algebraische Eingabemodus automatisch aktiviert. Dies bedeutet:

- Eine Befehlstaste führt die Befehlszeile aus und anschließend den Befehl.
- Eine Funktionstaste schreibt den Funktionsnamen in die Befehlszeile. Verlangt die Funktion die Angabe eines Arguments in Klammern, so wird die öffnende Klammer mit vorgegeben.
- Eine USER Menütaste schreibt den entsprechenden Namen in die Befehlszeile.

Alpha-Eingabemodus. Dieser dient zum Eingeben von Strings und Programmen. Das Drücken von ■" oder « aktiviert automatisch den Alpha-Eingabemodus und zeigt den α Indikator an. In diesem Modus bedeutet:

- Eine Befehlstaste schreibt den Befehl in die Befehlszeile.
- Eine Funktionstaste schreibt den Funktionsnamen in die Befehlszeile.
- Eine USER Menütaste schreibt den entsprechenden Variablennamen in die Befehlszeile.

Befindet sich der Cursor am Ende der Befehlszeile oder wurde der Einfügungsmodus aktiviert, so werden die erforderlichen Leerzeichen zur Trennung der Befehle mit eingefügt.

Ausnahmefälle

Um Ihnen die Modus-Wahl zu ermöglichen, während sich die Befehlszeile im unmittelbaren oder algebraischen Eingabemodus befindet, werden die folgenden Befehle ohne Auswirkung auf die Befehlszeile ausgeführt.

- STD, DEG und RAD im MODE Menü.
- DEC, HEX, OCT und BIN im BINARY Menü.

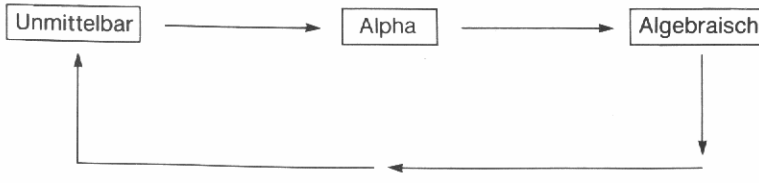
Da die folgenden Befehle nur in einem Programm sinnvoll sind, wird nach dem Drücken der jeweiligen Tasten immer der Befehlsname in die Befehlszeile geschrieben:

- HALT im PROGRAM CONTROL Menü.
- Alle Tasten im PROGRAM BRANCH Menü.

Um das versehentliche Löschen von Variablen zu vermeiden, wird nach dem Drücken von CLUSR (im MEMORY Menü) immer CLUSR in die Befehlszeile geschrieben. Sie müssen zur Ausführung des Befehls explizit [ENTER] drücken.

Manuelle Wahl des Eingabemodus

Der Rechner schaltet automatisch zwischen unmittelbarem und algebraischen Modus um, wenn ['] gedrückt wird. Er schaltet ebenso zum Alpha-Eingabemodus um, wenn Sie [■] ["] oder [«] drücken. Durch Drücken von [α] wird der Eingabemodus wie folgt umgeschaltet:



Manuelle Wahl des Eingabemodus

Sie können demzufolge durch ein- oder zweimaliges Drücken von [α] in jeden beliebigen Eingabemodus umschalten. Nachfolgend einige Beispiele zur Anwendung der Taste [α].

- Nehmen Sie an, Sie möchten ein Programm schreiben, welches jedoch nur ein- oder zweimal ausgeführt werden soll. Drücken Sie [α] zur Wahl des Alpha-Eingabemodus; tippen Sie das Programm ohne die entsprechenden Begrenzungszeichen ein und drücken Sie [ENTER] zur Ausführung des Programms; über ■ [COMMAND] können Sie das Programm in die Befehlszeile zurückrufen und erneut durch Drücken von [ENTER] ausführen.
- Es soll angenommen werden, daß mehrere Variablen über eine Operation zu löschen sind. Drücken Sie [{], um eine Liste zu beginnen. Drücken Sie [α] zur Wahl des Alpha-Eingabemodus und drücken Sie die Tasten der zu löschenden Variablen im USER Menü; durch Drücken von [ENTER] wird die Liste in den Stack übernommen—mit ■ [PURGE] lassen sich nun alle spezifizierten Variablen löschen.
- In einem Programm soll das Zeichen → in einem Namen verwendet werden. Da der Alpha-Eingabemodus aktiv ist, würde durch Drücken von ■ [→] der Befehl "→", umgeben von Leerzeichen, in die Befehlszeile übernommen werden. Drücken Sie deshalb [α], um den algebraischen Eingabemodus zu wählen, und drücken Sie nun ■ [→]; durch Drücken von [α] [α] kehren Sie zum Alpha-Eingabemodus zurück.

Wie der Cursor verschiedene Modi darstellt

Die Form des Cursors kennzeichnet den gegenwärtigen Eingabemodus und die Wahl für Einfügungs- oder Ersetzungsmodus. Die folgende Tabelle zeigt die sechs möglichen Kombinationen der verschiedenen Modi an.

	Einfügungsmodus	Ersetzungsmodus
Unmittelbar	▷	□
Algebraisch	◁	▤
Alpha	◆	■

Ausführen der Befehlszeile

Nach Drücken von [ENTER] (oder einer Taste, die ENTER im momentanen Modus bewirkt) führt der Rechner folgende Schritte aus:

1. Der "Busy" Indikator ((•)) wird eingeschaltet.
2. Wenn UNDO aktiv ist, wird eine Kopie des momentanen Stackinhalts gesichert.
3. Der Textstring in der Befehlszeile wird auf Begrenzungs- und Trennzeichen durchsucht und danach in entsprechende Teilstrings zerlegt.
4. Jeder Teilstring wird auf Syntax geprüft, um die korrespondierenden Objekttypen zu identifizieren.
5. Wenn COMMAND aktiv ist, wird eine Kopie der Befehlszeile im Befehlsstack gesichert.
6. Die Befehlszeile wird ausgeführt.
7. Der "Busy" Indikator ((•)) wird wieder ausgeschaltet.

Endet die Syntaxprüfung in Schritt 4 fehlerhaft, so werden die Schritte 5 und 6 nicht ausgeführt. Statt dessen wird `Syntax Error` angezeigt, wobei der unkorrekte Text invers dargestellt wird, gefolgt vom Cursor. Resultierte der Fehler aus einer unvollständigen Syntax, dann wird der Cursor am Ende der Zeile positioniert.

Edieren gespeicherter Objekte

Sie können ein gespeichertes Objekt in die Befehlszeile zurückrufen, dort ansehen oder edieren, und, falls gewünscht, das gespeicherte Objekt durch das modifizierte ersetzen.

Taste	Bedeutung
■ [EDIT]	**Ediere Ebene 1:** Gibt ein Objekt aus Ebene 1 in die Befehlszeile zurück.
n ■ [VISIT]	**Ediere Ebene *n*:** Gibt ein Objekt aus Ebene *n* in die Befehlszeile zurück.
'Name' ■ [VISIT]	**Ediere eine Variable:** Gibt den Inhalt der spezifizierten Variablen in die Befehlszeile zurück.

Es wird das Cursor-Menü und der Alpha-Eingabemodus aktiviert. Das ursprüngliche Objekt, sofern sichtbar, wird hervorgehoben, um Sie zu erinnern, daß nur eine Kopie modifiziert wird und das Original noch erhalten ist.

Nach dem Abschluß des Ediervorgangs bzw. des Anzeigens haben Sie folgende Möglichkeiten:

- Drücken Sie [ON], um den Ediervorgang aufzuheben, die Befehlszeile zu löschen und das ursprüngliche Objekt unverändert zu lassen.
- Drücken Sie [ENTER] (oder eine Taste, welche ENTER ausführt), um das ursprüngliche Objekt zu ersetzen.

Ist das Cursor-Menü am Abschluß des Ediervorgangs immer noch aktiv, so wird das vorangehende Menü zurückgespeichert.

Rücksichern der Befehlszeile

Der HP-28S sichert den Inhalt der vier zuletzt ausgeführten Befehlszeilen. Einmaliges Drücken von ■[COMMAND] gibt die als letzte ausgeführte Befehlszeile zurück (wobei die momentane, falls vorhanden, ersetzt wird); ein weiteres Drücken von ■[COMMAND] bringt die als zweitletzte ausgeführte Befehlszeile zurück, und so weiter. Drücken Sie mehr als viermal ■[COMMAND], so beginnt die Reihenfolge erneut mit der zuletzt ausgeführten Befehlszeile.

Einige Anwendungen von ■[COMMAND] sind unter "Wenn die falsche Funktion ausgeführt wurde" auf Seite 47 und "Manuelle Wahl des Eingabemodus" auf Seite 171 beschrieben.

Sie können diese Rücksicherungsmöglichkeit desaktivieren, indem Sie CMD im MODE Menü drücken. Das Quadrat wird danach aus dem Menüfeld genommen, um den inaktiven Status anzudeuten. Das erneute Drücken von CMD reaktiviert diese Art der Rücksicherung.

Die Befehlszeile als String

Der in die Befehlszeile eingetippte Text ist äquivalent zum Inhalt eines String-Objekts—d.h. einer Zeichensequenz. Sie können eine Befehlszeile auch dadurch ausführen, indem Sie den entsprechenden Text/Befehl in einen String eingeben und anschließend STR→ (*STRing-in-Objekte*) ausführen. Dieses Verfahren ist sehr hilfreich, wenn Sie Programme in Textform speichern möchten, was wesentlich kompakter als in Objektform ist. Außerdem werden alle lokalen Namen, welche bei der Ausführung von STR→ existieren, in der Befehlszeile erkannt.

19

Der Stack

Dieses Kapitel wiederholt einige Dinge, welche Sie zuvor über den Stack erfahren haben, und beschreibt Befehle zur Manipulation von Objekten im Stack.

Näheres zum Stack-Konzept

Der Stack ist eine Folge von numerierten *Ebenen*, von welchen jede ein Objekt enthält. Die über die Befehlszeile eingetippten Objekte werden nach der Ausführung von ENTER in den Stack übernommen, wobei das erste Objekt in der Befehlszeile als erstes im Stack gespeichert wird. Jedes Objekt kommt zuerst in Ebene 1 und schiebt dabei alle anderen eine Ebene nach oben. Außer der Speicherbereichsgrenze gibt es keine Einschränkung für die Anzahl von Stackebenen.

Im allgemeinen nimmt ein Befehl Eingabe-Objekte (*Argumente* genannt) vom Stack und gibt Ausgabe-Objekte (*Ergebnisse* genannt) in den Stack zurück. So nimmt z.B. die Funktion + zwei Argumente vom Stack und gibt deren Summe in Ebene 1 zurück.

Diese Art Logik, bei welcher der Befehl auf die Argumente folgt, wird *Stacklogik, postfix logic* oder *UPN* (*Umgekehrte Polnische Notation*) genannt, nach dem polnischen Mathematiker Jan Lukasiewicz (1878-1956).

Die Ergebnisse eines Befehls sind wieder als Argumente für den nächsten Befehl verwendbar. Falls Sie diese nicht gleich verwenden, so lassen Sie sie einfach im Stack—Sie können sie später von dort zurückrufen.

Objekte verlassen den Stack über Ebene 1, wobei die verbleibenden Objekte um eine Ebene nach unten geschoben werden. Aus Gründen der Übersichtlichkeit empfiehlt es sich, daß der Stackinhalt gelöscht wird, bevor die Bearbeitung einer neuen Aufgabe begonnen wird.

Ansehen des Stackinhalts

Normalerweise ist der Inhalt der ersten zwei Stackebenen in der Anzeige sichtbar. Stellt das Objekt in Ebene 1 ein größeres Objekt dar, so wird nur dessen erster Teil angezeigt. Die Operationen ■ VIEW↑ und ■ VIEW↓ ermöglichen Ihnen das Ansehen des ersten Teils aller Objekte im Stack sowie des vollständigen Objekts in Ebene 1.

Diese Operationen bewegen das "Anzeigefenster", durch welches Sie den Stackinhalt ansehen können. Die Größe des Fensters variiert von einer bis zu vier Anzeigezeilen, in Abhängigkeit von der Existenz eines Menüs, der Befehlszeile oder beiden.

Taste	Bedeutung
■ VIEW↑	Bewegt das Fenster nach oben (zu höheren Stackebenen).
■ VIEW↓	Bewegt das Fenster nach unten (zu niedrigeren Stackebenen).

Das Ansehen des Stackinhalts hat keine Auswirkung auf dessen Inhalt, die Befehlszeile oder die Wirkungsweise von Befehlen.

Manipulieren des Stacks

In Teil 1 haben Sie bereits einige Befehle zur Manipulation des Stacks benutzt: CLEAR (zum Löschen des Inhalts), DROP (zum Löschen des Objekts in Ebene 1) und SWAP (zum Tauschen der Inhalte von Ebene 1 und 2). Dieser Abschnitt enthält eine kurze Beschreibung aller Lösch-, Verschiebe- und Kopier-Befehle für Stackobjekte; Einzelheiten erfahren Sie unter "STACK" im Referenzhandbuch.

Verschieben von Stackobjekten. Diese Befehle ordnen die Objekte im Stack neu an (die Anzahl bleibt unverändert). Ist den Befehlen ein *"n"* vorangestellt, so benötigen sie ein reelles Argument.

Befehl	Bedeutung
SWAP	Tauscht das Objekt aus Ebene 2 mit dem in Ebene 1.
ROT	Verschiebt das Objekt aus Ebene 3 in Ebene 1.
n ROLL	Verschiebt das Objekt aus Ebene *n* in Ebene 1.
n ROLLD	Verschiebt das Objekt aus Ebene 1 in Ebene *n*.

Die Befehle ROT (*ROTieren*), ROLL und ROLLD (*ROLL Down*) beschreiben die Auswirkung ihrer Anwendung: ROT verschiebt das Objekt aus Ebene 3 in Ebene 1, wobei ein Block von drei Objekten rotiert wird; ROLL und ROLLD "rollen" einen Block von *n* Objekten.

Kopieren von Stackobjekten. Diese Befehle geben eine Kopie eines oder mehrerer Stackobjekte zurück. Das Kopieren von nur einem Objekt gibt dessen Kopie in Ebene 1 zurück und verschiebt die anderen je eine Ebene nach oben. Werden mehrere Objekte kopiert, so erfolgt dies in einem Block auf ähnliche Weise. Ist den Befehlen ein *"n"* vorangestellt, so benötigen sie ein reelles Argument.

Befehl	Bedeutung
DUP	Kopiert das Objekt aus in Ebene 1. (Wenn keine Befehlszeile existiert, können Sie DUP durch Drücken von ENTER ausführen.)
OVER	Kopiert das Objekt in Ebene 2.
n PICK	Kopiert das Objekt in Ebene *n*.
DUP2	Kopiert die Objekte in Ebene 1 und 2.
n DUPN	Kopiert die Objekte in Ebene 1 bis *n*.

Löschen von Stackobjekten. Diese Befehle löschen ein oder mehrere Objekte im Stack, wobei die verbleibenden Objekte auf die niedrigeren Ebenen verschoben werden. Ist den Befehlen ein *"n"* vorangestellt, so benötigen sie ein reelles Argument.

Befehl	Bedeutung
DROP	Löscht das Objekt in Ebene 1.
DROP2	Löscht die Objekte in Ebene 1 und 2.
n DROPN	Löscht die Objekte in Ebene 1 bis *n*.
CLEAR	Löscht alle Objekte.

Lokale Variable

In Teil 1 haben Sie einige Benutzerfunktionen geschrieben—Programme, welche lokale Variable definieren und diese in einem einzelnen Ausdruck oder Programm verwenden. Benutzerfunktionen können in algebraischen Ausdrücken auf gleiche Weise wie in internen Funktionen verwendet werden.

Der Gebrauch von lokalen Variablen vermindert den Bedarf für Stackmanipulationen. Beim Erzeugen einer lokalen Variablen wird deren Inhalt vom Stack genommen und Sie können dann auf diesen über den Variablennamen Bezug nehmen.

Lokale Variable finden neben der Verwendung in Benutzerfunktionen noch andere Anwendungsmöglichkeiten. Fast alle Beispiele in Kapitel 28 verwenden lokale Variable. Von besonderem Interesse dürfte "Kubische Funktionen" auf Seite 241, "MULTI (Mehrfache Ausführung)" auf Seite 253, "PRESERVE (Sichern und Wiederherstellen des vorherigen Status)" auf Seite 258 und "SORT" auf Seite 270 sein.

Rücksicherung der letzten Argumente

Der HP-28S speichert die Argumente der zuletzt ausgeführten Befehle. In Abhängigkeit vom Befehl können bis zu drei Argumente gesichert werden. (Erfordert ein Befehl keine Argumente, so bleiben die zuletzt gesicherten Argumente erhalten.) LAST gibt die gesicherten Argumente in die von diesen zuvor belegten Stackebenen zurück.

Wenn Sie genau die gleichen Argumente für zwei oder mehr aufeinander folgende Befehle benötigen, können Sie LAST ausführen, um für den nächsten Befehl Kopien der Argumente im Stack bereitzuhalten. Sind nicht genau die gleichen Argumente erforderlich oder treten die Befehle nicht nacheinander auf, so ist es einfacher, lokale Variable zu verwenden.

Sie können LAST (das Sichern von Argumenten) desaktivieren, indem Sie LAST im MODE Menü drücken. Das Quadrat wird aus dem Menüfeld genommen, wodurch der inaktive Status dieser Funktion angezeigt wird. Dieser Schritt wird nicht allgemein empfohlen, da der Rechner die letzten Argumente beim Eintreten des Fehlerfalls verwendet. Allerdings können Sie durch Desaktivieren von LAST versuchen, die Programm- oder Befehlsausführung fortzusetzen, falls diese aufgrund einer unzureichenden Speichergröße abgebrochen wurde. Denken Sie jedoch daran, durch Drücken von LAST die Funktion wieder zu aktivieren.

Rücksicherung des Stacks

Bei jedem Drücken von ENTER (oder einer Taste, welche ENTER ausführt) speichert der HP-28S zuerst eine Kopie des Stacks und führt danach die spezifizierten Anweisungen aus. Wenn die Ergebnisse nicht wie erwartet ausfallen, können Sie den gesicherten Stackinhalt durch Drücken von ■ UNDO wieder zurückspeichern. Beachten Sie, daß UNDO sich nur auf den Stack bezieht—es ändert keine Benutzerflags oder Benutzervariablen. Ein Beispiel zur Anwendung von ■ UNDO finden Sie auf Seite 47.

Sie können diese Sicherungsmöglichkeit desaktivieren, indem Sie UNDO im MODE Menü drücken. Danach wird das kleine Quadrat aus dem Menüfeld genommen, was den inaktiven Status der Funktion kennzeichnet. Um UNDO zu reaktivieren, ist UNDO ein zweites Mal zu drücken.

Der Stack als Liste

Der Inhalt des Stacks ist gleichwertig zum Inhalt einer Liste—d.h. einer Sequenz von Objekten. Sie können alle im Stack gespeicherten Objekte in einer einzelnen Liste speichern, indem Sie die Befehlsfolge DEPTH →LIST ausführen. DEPTH gibt die Anzahl gespeicherter Stack-Objekte zurück und →LIST (*Stack in Liste*) übernimmt die entsprechende Anzahl von Objekten in eine Liste.

Öfters wird auch eine Liste durch den Befehl LIST→ (*Liste in Stack*) in den Stack "entleert". Nachdem die Elemente im Stack manipuliert wurden, können Sie durch den Befehl →LIST wieder in eine Liste übernommen werden. Beispiele hierzu finden Sie unter "MEDIAN (Median von statistischen Daten)" auf Seite 273.

20

Der Speicherbereich

Der HP-28S verwendet den vorhandenen Speicherbereich für verschiedene Anwendungszwecke, wobei Befehlszeile, Stack, Benutzerspeicher, Rücksicherungsoptionen und Betriebssystem eingeschlossen sind. Die Befehlszeile und der Stack sind in Kapitel 18 und 19 beschrieben. Dieses Kapitel diskutiert primär den Benutzerspeicher, einschließlich der Verzeichnisse; außerdem werden Situationen erläutert, in welchen der verfügbare Speicherbereich sich als zu klein erwiesen hat und welche Auswirkungen sich daraus ergeben.

Benutzerspeicher

Im Benutzerspeicher sind Variable und Verzeichnisse gespeichert.

Globale Variable

Eine Variable ist die Kombination eines Namens und jedes anderen Objekts. Der Name stellt den Variablennamen dar, während das weitere Objekt den Wert bzw. den Inhalt der Variablen repräsentiert.

Globale Variablen sind die im Benutzerspeicher enthaltenen Variablen. Weiterhin sind hier auch *lokale Variablen* gespeichert, welche von Programmen erzeugt werden und nur während deren Programmausführung existieren. Sie stellen eine Ersatzmöglichkeit für Stackmanipulationen dar und sind in Kapitel 19, "Der Stack", beschrieben. Im vorliegenden Kapitel bezieht sich der Term "Variable" auf globale Variable.

Sie verwendeten bereits die folgenden Befehle zum Erzeugen, Zurückrufen und Löschen von Variablen, wobei die Befehle jede Variable gleich—unabhängig von deren Argumenten—behandelt haben.

Befehl	Bedeutung
STO	Erzeugt eine Variable mit dem spezifizierten Inhalt und Namen.
RCL	Ruft den Inhalt der spezifizierten Variablen zurück.
PURGE	Löscht eine oder mehrere spezifizierte Variablen.

Verzeichnisse

In Kapitel 4, "Wiederholen von Berechnungen", haben Sie den Löser zur Berechnung des Gesamtwiderstands zweier Schaltkreise verwendet. Nachfolgend ein Rückblick auf die hierbei von Ihnen erlernten Konzepte.

Es gibt zwei primäre Gründe zum Erzeugen von Verzeichnissen:

- Das *Gruppieren* von Variablen für eine bestimmte Anwendung. Sie haben das Verzeichnis EE für das Arbeiten mit Problemstellungen aus der Elektrotechnik erzeugt und sich dabei ausschießlich auf die darin enthaltenen Variablen konzentriert. Analog verläuft es bei der Bearbeitung anderer Aufgaben (in anderen Verzeichnissen), wobei die unrelevanten Variablen nicht in Erscheinung treten.
- Das *Separieren* von verschiedenen Variablensätzen, welche die gleichen Namen beinhalten. Sie erzeugten die Verzeichnisse SP1 und SP2 (*Seriell-Parallel-1* und *Seriell-Parallel-2*) innerhalb von EE, um unterschiedliche Werte für R1, R2 und R3 zu speichern. Sie wechseln von einem Variablensatz zum nächsten, indem Sie einfach das Verzeichnis wechseln.

Erzeugen eines Verzeichnisses. Um ein Verzeichnis zu erzeugen, geben Sie den gewünschten Namen ein und führen den Befehl CRDIR (*CReate DIRectory bzw. Erzeuge Verzeichnis*) aus. Der Name des Verzeichnisses erscheint danach im USER Menü. Das neue Verzeichnis wird als *Unterverzeichnis* bezeichnet, während das zugehörige höhere Verzeichnis als *Oberverzeichnis* bezeichnet wird.

Das aktuelle Verzeichnis. Ursprünglich existierte nur das interne Verzeichnis HOME (oberstes Verzeichnis aller Verzeichnisse). Nachdem ein weiteres Verzeichnis erzeugt wurde, könnnen Sie das *aktuelle Verzeichnis* wählen—das Verzeichnis, dessen Variablen im USER Menü angezeigt werden.

Sie wählen das aktuelle Verzeichnis, indem Sie dessen Namen auswerten. Wenn Sie z.B. gerade ein Verzeichnis erzeugt hätten, so würden Sie dieses durch Drücken der entsprechenden Taste im USER Menü als aktuelles Verzeichnis bestimmen.

Fast alle Befehle, die Variable verwenden, beziehen sich immer auf das aktuelle Verzeichnis (da die Verzeichnisse die Verwendung bestimmter Variablen steuern sollen). Sie können eine Variable nur dann ändern, wenn Sie sich im aktuellen Verzeichnis befindet.

Nachstehende Befehle des MEMORY Menüs wirken auf das aktuelle Verzeichnis:

Befehl	Bedeutung
VARS	Gibt eine Liste mit allen Variablennamen und Verzeichnissen im aktuellen Verzeichnis zurück.
ORDER	Ordnet Variable und Verzeichnisse im aktuellen Verzeichnis entsprechend der Reihenfolge in einer Liste.
CLUSR	Löscht alle Variablen und leere Verzeichnisse im aktuellen Verzeichnis.

Der aktuelle Pfad. Über den Befehl PATH können Sie überprüfen, an welcher Stelle der Verzeichnisstruktur Sie sich momentan befinden. Sie erhalten eine Liste, welche die Folge der Verzeichnisse—ausgehend von HOME bis zum aktuellen Verzeichnis—spezifiziert.

In einigen Fällen durchsucht der Rechner nicht nur das aktuelle Verzeichnis, sondern den ganzen aktuellen Pfad. Die Suche beginnt im aktuellen Verzeichnis; wurde die Variable hier nicht gefunden, so wird die Suche im Oberverzeichnis fortgesetzt, usw., bis zum HOME Verzeichnis.

Dies erfolgt bei der Auswertung von Namen—schließlich ist eine Rückkehr zu einem Oberverzeichnis nur möglich, wenn dessen Name erfolgreich ausgewertet werden kann. Die Auswertung von Namen tritt dann auf, wenn Sie einen Namen ohne Anführungszeichen eingeben, den Löser anwenden, algebraische Ausdrücke im Stack auswerten, und so weiter.

Weitere Befehle, die den aktuellen Pfad durchsuchen, sind RCL und PRVAR (*PRint VARiables*). Beachten Sie, daß keine dieser Anweisungen den Inhalt der Variablen verändern kann.

Da das HOME Verzeichnis immer im aktuellen Pfad enthalten ist, kann der Rechner immer die Variablen des HOME Verzeichnisses auffinden. Vielleicht richten Sie es sich so ein, daß Sie den Inhalt von HOME auf Unterverzeichnisse und die Variablen beschränken, welche Sie ständig zur Verfügung haben möchten.

Verzeichnisstruktur. Die nachstehenden Diagramme zeigen die von Ihnen erzeugte Verzeichnisstruktur in Kapitel 4. Im ersten Diagramm stellt HOME das aktuelle Verzeichnis dar, im zweiten ist es EE und im dritten SP2. Jedes Diagramm verwendet die folgenden Symbole:

In Verzeichnisdiagrammen benutzte Symbole

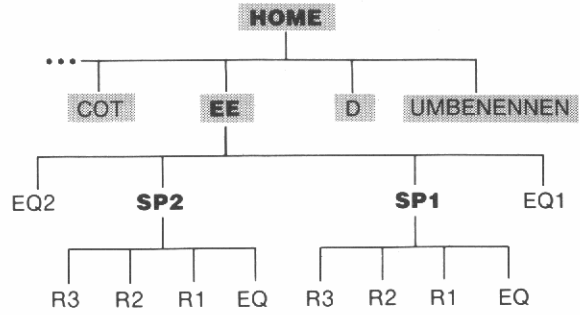
Name Name eines Verzeichnisses.

Name Name des aktuellen Verzeichnisses.

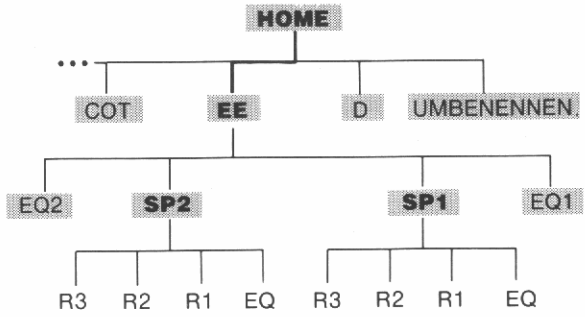
Name Ein Name im aktuellen Verzeichnis. Diese Namen erscheinen im USER Menü. Die korrespondierenden Variablen können geändert werden.

—— Der aktuelle Pfad.

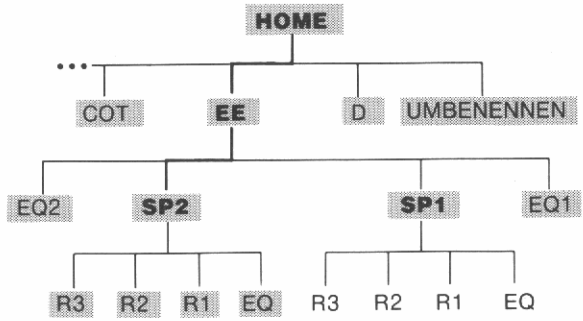
Name Ein Name im aktuellen Pfad. Diese Namen können nur durch deren Auswertung, RCL oder PRVARS gefunden werden. Die korrespondierenden Variablen können nicht geändert werden.



Aktuelles Verzeichnis ist HOME



Aktuelles Verzeichnis ist EE



Aktuelles Verzeichnis ist SP2

Löschen eines Verzeichnisses. Sie können ein *leeres* Verzeichnis genauso wie eine Variable löschen: Wechseln Sie zum Verzeichnis, in welchem das zu löschende Verzeichnis enthalten ist und geben Sie dessen Namen in den Stack ein; das Ausführen von PURGE löscht dann das Verzeichnis.

Enthält das zu löschende Verzeichnis Variable oder Unterverzeichnisse, so müssen Sie diese zuerst löschen, um ein leeres Verzeichnis zu erhalten. Hier eine allgemeine Vorgehensweise:

1. Wählen Sie das zu löschende Verzeichnis.
2. Führen Sie CLUSR zum Löschen des Verzeichnisinhalts aus.
3. Wechseln Sie zum zugehörigen Oberverzeichnis.
4. Löschen Sie das gewünschte Verzeichnis.

Wenn in Schritt 2 ein `Non-Empty Directory` Fehler auftritt, so enthält das Verzeichnis ein Unterverzeichnis, welches nicht leer ist. In diesem Fall müssen Sie die Schritte 1, 2 und 3 zum Löschen des Unterverzeichnisinhalts ausführen. Danach können Sie mit Schritt 2 bis 4 fortfahren, um das Verzeichnis zu löschen.

Wechseln innerhalb der Verzeichnisstruktur. Kapitel 28 enthält Programme zum Wechseln innerhalb der Verzeichnisstruktur. Näheres dazu finden Sie unter "Wechsel von Verzeichnissen" auf Seite 275.

Rücksicherungsmöglichkeiten

Der HP-28S speichert automatisch Kopien von Befehlszeilen, Argumenten und dem Stack. Diese Kopien ermöglichen es Ihnen, im Fehlerfall zum vorherigen Zustand zurückzukehren. Sie können dann eine Berechnung wiederholen, ohne von vorne beginnen zu müssen. Die Kopien von Befehlszeilen und Argumenten eignen sich auch bestens zur Wiederholung von Berechnungen.

Da diese Kopien einen beträchtlichen Teil des Speichers belegen können, läßt sich jede dieser Rücksicherungsmöglichkeiten—Befehlszeile, Argumente, Stack—aktivieren und desaktivieren. Die Operationen hierzu erscheinen im MODE Menü.

Im allgemeinen empfiehlt es sich, diese Fähigkeiten aktiviert zu lassen. Ist nur noch ein kleiner Speicherbereich verfügbar und es wurden größere Objekte durch eine dieser Rücksicherungen gespeichert, so können Sie freien Speicher zurückgewinnen, indem Sie gezielt eine der Sicherungsmöglichkeiten desaktivieren und reaktivieren, wobei der entsprechende Speicherplatz freigegeben wird.

Zu kleiner Speicherbereich

Der HP-28S enthält 32 KBytes als Benutzerspeicher, von welchen etwa 400 für Systemzwecke reserviert sind. Praktisch benötigt jede Operation im HP-28S etwas an Speicherplatz—selbst das Interpretieren der Befehlszeile. Einige Befehle (COLCT, EXPAN, TAYLR) haben einen beträchtlichen Speicherbedarf, vor allem, wenn ihre Argumente komplizierter werden. Reservieren Sie wenigstens einige 1000 Bytes für den dynamischen Gebrauch durch das Betriebssystem.

Sie können die Größe des freien Speicherplatzes durch den Befehl MEM (im MEMORY Menü) abfragen.

Da sich das Betriebssystem des HP-28S Teile des Speichers mit den Benutzer-Objekten teilt, kann der Speicher so voller Benutzer-Objekte sein, daß normale Operationen des Rechners schwierig oder undurchführbar werden. Der HP-28S besitzt eine Reihe von Warnungen für knapp werdenden freien Speicherplatz, welche nachstehend in der Reihenfolge zunehmender Dringlichkeit—d.h. abnehmender Speicherplatzreserven—gelistet sind:

Unzureichender Speicherplatz. Wenn nicht genügend Speicherplatz zur Ausführung eines Befehls vorhanden ist, wird die Ausführung gestoppt und es wird `Insufficient Memory` angezeigt. Ist LAST aktiv, so werden die ursprünglichen Argumente zurückgespeichert, andernfalls gehen sie verloren.

Kein Platz für UNDO. Ist nicht genügend Speicherplatz zum Kopieren des Stackinhalts vorhanden, so erscheint die Meldung `No Room for UNDO` und UNDO wird automatisch desaktiviert.

Kein Platz für ENTER. Wenn nicht genügend Platz zur Verarbeitung der Befehlszeile vorhanden ist, löscht der Rechner die Befehlszeile und zeigt `No Room to ENTER` an. Eine Kopie der gelöschten Befehlszeile wird im Befehls-Stack gesichert, sofern er aktiviert ist.

Wenn Sie ein existierendes Objekt mit EDIT oder VISIT edieren möchten und eine Kopie der unverarbeiteten Befehlszeile im Befehls-Stack gespeichert wurde, so löschen Sie zunächst die ursprüngliche Form des Objekts. Drücken Sie danach ■ COMMAND zur Rücksicherung der Befehlszeile, die das modifizierte Objekt enthält; durch Drücken von ENTER können Sie dann die neue Version eingeben.

Zu kleiner Speicherbereich. Sind weniger als 128 Bytes an freiem Speicherplatz vorhanden, so blinkt die Meldung `Low Memory!` einmal in der obersten Anzeigezeile. Sie blinkt von nun an nach jeder Tasteneingabe, bis zusätzlicher Speicherplatz verfügbar ist. Löschen Sie unnötige Objekte aus dem Speicher, bevor Sie Ihre Berechnungen fortsetzen.

Kein Platz für Stackanzeige. Es kann vorkommen, daß der Rechner alle anstehenden Operationen abschließt und dann nicht genügend Speicherplatz für die normale Stackanzeige übrig hat. In diesen Fällen zeigt der Rechner `No Room to Show Stack` in der obersten Anzeigezeile an. Die zur Anzeige des Stackinhalts vorgesehenen Zeilen enthalten nun nur noch die Objekttypen wie zum Beispiel `Real Number`, `Algebraic`, und so weiter.

Der zur Anzeige eines Stack-Objekts erforderliche Platz hängt vom Objekttyp ab—algebraische Objekte haben im allgemeinen den geringsten Speicherbedarf. Löschen Sie ein oder mehrere Objekte aus dem Speicher oder speichern Sie es als Variable, so daß es nicht angezeigt werden muß.

Kein freier Speicherplatz. Der Extremfall von geringem Speicherplatz liegt dann vor, wenn nicht mehr genügend Speicher vorhanden ist, um irgendeine Funktion auszuführen—Anzeigen des Stacks, usw.. In dieser Situation *müssen* Sie Teile des Speichers löschen, bevor Sie fortfahren können. Es wird eine spezielle `Out of Memory` Prozedur aktiviert, welche die folgende Anzeige hervorruft:

Der Rechner frägt Sie der Reihe nach, ob einer der folgenden Bereiche gelöscht werden soll:

1. Der COMMAND Stack (sofern aktiviert).
2. Der UNDO Stack (sofern aktiviert).
3. Die Argumente von LAST (sofern aktiviert).
4. Das CUSTOM Menü (sofern vorhanden).
5. Der Stack.
6. Jede Variable im HOME Verzeichnis.

Für jeden zu löschenden Bereich ist die Menütaste YES zu drücken; für die beizubehaltenden Bereiche drücken Sie NO.

Nachdem Sie YES wenigstens einmal gedrückt haben, können Sie versuchen, die `Out of Memory` Prozedur durch Drücken von ON abzubrechen. Ist nun genügend Speicherplatz vorhanden, so kehrt der Rechner zur normalen Anzeige zurück, andernfalls ertönt ein Akustiksignal und die Löschoperation wird fortgesetzt. Nach einem Durchlauf aller fünf Optionen versucht die `Out of Memory` Prozedur zur normalen Betriebsweise zurückzukehren. Ist immer noch nicht genügend Speicherplatz frei, dann beginnt die Prozedur wieder von vorne.

Wenn Sie für ein leeres Verzeichnis YES drücken, dann wird es gelöscht. Enthält das Verzeichnis noch Variable, so werden diese angezeigt.

Optimieren der Rechnerleistung

Von Zeit zu Zeit reorganisiert der Rechner den Speicherbereich, um eine optimalere Ausnutzung zu erhalten. Normalerweise ist dieser Vorgang nur als kurze Pause, z.B. während der Abbildung einer Grafik, bemerkbar. Bei einer hohen Speicherbelegung kann der Rechner jedoch selbst bei einfachen Operationen (z.B. Menüwahl) langsamer arbeiten.

Dieser Abschnitt enthält Hinweise zur Erhöhung der Verarbeitungsgeschwindigkeit (durch Reduzierung der erforderlichen Speicherreorganisationen) und Maximierung des verfügbaren Speichers (durch Erhöhung der Effektivität der Reorganisationen).

Erhöhung der Geschwindigkeit:

- Speichern Sie nur einige hundert Objekte im Stack.
- Lassen Sie keine großen Listen (mehr als zweihundert Objekte) im Stack—speichern Sie sie im Benutzerspeicher.

Maximieren des verfügbaren Speicherbereichs:

Hinweis

Die folgende Prozedur löscht den Stack, gesicherte Daten (COMMAND, UNDO, LAST), das momentane CUSTOM Menü, und jedes unterbrochene Programm.

---

1. Löschen Sie nicht mehr benötigte Variablen und Verzeichnisse im Benutzerspeicher.
2. Speichern Sie im Benutzerspeicher die Daten des Stacks, welche erhalten bleiben sollen.
3. Führen Sie einen Systemstopp durch (ON ▲).

Sie befinden sich nun in HOME (aktuelles Verzeichnis).

Minimieren der Speicherbelegung bei der Berechnung von Feldern: Speichern Sie Felder in Variablen und nehmen Sie über den Namen Bezug; vermeiden Sie deren Gebrauch im Stack. Nachstehende Hinweise sollen bei der Einhaltung dieser Strategie behilflich sein:

1. Planen Sie im voraus, wieviel Variable benötigt werden, Zwischenergebnisse eingeschlossen.
2. Erzeugen Sie *kleine* Felder des jeweiligen Typs (reell/komplex, Vektor/Matrix) und speichern Sie diese in Variablen; verwenden Sie dann RDM zur Korrektur deren Größe.
3. Führen Sie Berechnungen unter Verwendung der Speicherarithmetik-Befehle im STORE Menü durch.
4. Um individuelle Elemente zu bearbeiten, ist GET, GETI, PUT, PUTI *mit dem Variablennamen* oder algebraische Syntax (wie z.B. `'A(5,6)' EVAL` und `'B(3)' STO`) zu verwenden; übernehmen Sie nicht das ganze Feld in den Stack.

21

Menüs

Alle Operationen, Befehle und Funktionen im HP-28S sind über das Tastenfeld oder ein Menü verfügbar. Wenn Sie ein Menü aufrufen bzw. wählen, erscheinen sechs *Menüfelder* in der unteren Anzeigezeile. Diese Felder stellen eine *Menüzeile* dar, welche die momentane Definition der sechs *Menütasten* (direkt unterhalb der Anzeige) kennzeichnet. (Das Cursor-Menü ist eine Ausnahme, da dessen Definition in weißer Beschriftung über den Tasten angebracht ist.)

Zusätzlich zu den Tasten, die spezifische Menüs auswählen (z.B. ■ ARRAY oder TRIG), steuern die folgenden Tasten ebenfalls Menüoperationen:

Taste	Bedeutung
◆▶	**Cursor-Menü ein/aus.** Ist das Cursor-Menü inaktiv, so wird es aktiviert; bei aktivem Status wird es desaktiviert.
■ CUSTOM	**Letzes CUSTOM Menü.** Zeigt das CUSTOM Menü an, welches als letztes durch den MENU Befehl erzeugt wurde.
NEXT	**Nächste Menüzeile.** Zeigt die nächste Zeile der Menütasten an. Wurde die letzte Zeile angezeigt, so erscheint wieder die erste Zeile.
■ PREV	**Vorherige Menüzeile.** Zeigt die vorherige Zeile der Menütasten an. Wurde die erste Zeile angezeigt, so erscheint die letzte Zeile.
■ MENUS	**Menüsperre.** Schaltet die Menüsperre ein oder aus. Bei aktivierter Sperre sind die umgeschalteten und normalen "Positionen" für die ersten drei Tastenreihen auf dem linken Tastenfeld umgekehrt. Das Drücken von A wählt in diesem Fall das ARRAY Menü und Drücken von ■ A bewirkt das Schreiben von "A".

Befehlsmenüs

Die folgenden Menüs enthalten Tasten für interne Operationen, von denen die meisten programmierbar sind. Eine kurze Beschreibung der Befehle finden Sie in Anhang B, "Menütabellen"; das Referenzhandbuch behandelt die Menüs und deren Befehle in ausführlicher Form.

Die Wirkung der Tasten in diesen Menüs hängt vom jeweiligen Eingabemodus ab (auf Seite 169 beschrieben).

Menü	Bedeutung
ALGEBRA	Algebra-Befehle.
ARRAY	Vektor- und Matrix-Befehle.
BINARY	Arithmetik ganzer Zahlen, Basisumwandlungen, Bitmanipulationen.
COMPLEX	Befehle für komplexe Zahlen.
LIST	Listen-Befehle.
LOGS	Logarithmische, exponentielle und hypberbolische Funktionen.
MEMORY	Benutzerspeicher, Verzeichnisse.
MODE	Wahl des Anzeige-, Rücksicherungs- und Winkelmodus sowie des Dezimalzeichens.
PLOT	Grafik-Befehle.
PRINT	Druckbefehle.
PROGRAM BRANCH	Programmverzweigungsstrukturen.
PROGRAM CONTROL	Programmsteuerung, -stopp und Einzelschritt-Ausführung.
PROGRAM TEST	Flags, logische Vergleiche.
REAL	Befehle für reelle Zahlen.
SOLVE	Befehle für numerische und symbolische Lösungen, der Löser.

Menü	Bedeutung
STACK	Befehle zur Stackmanipulation.
STAT	Befehle zur Statistik und Wahrscheinlichkeitsrechnung.
STORE	Speicherarithmetik-Befehle.
STRING	Befehle für Zeichenketten bzw. Strings.
TRIG	Trigonometrische Funktionen, Umwandlung zwischen Koordinatensystemen und Winkelmaßen.

Menüs von Operationen

Die folgenden Menüs enthalten nicht-programmierbare Operationen.

Menü	Bedeutung
Cursor	Zum Edieren der Befehlszeile (in Kapitel 18 beschrieben).
CATALOG	Befehlskatalog, einschließlich des USAGE Untermenüs. In Kapitel 22 beschrieben.
UNITS	Für Konvertierungen verfügbare Einheiten. In Kapitel 14 beschrieben.

Menüs von Variablen

Menü	Bedeutung
Löser	Speichert Werte und löst nach Variablen der momentanen Gleichung. Die unterschiedliche Anzeigeform (dunkle Zeichen auf hellem Hintergrund) kennzeichnet seine unterschiedliche Wirkungsweise.
USER	Zeigt Variable und Unterverzeichnisse im aktuellen Verzeichnis an. Die Wirkung der Tasten hängt vom gewählten Eingabemodus ab.

CUSTOM Menüs

Über den Befehl MENU kann aus einer Liste von Namen und Befehlen ein passendes Menü erzeugt werden. Dieses CUSTOM Menü kann einem Löser-Menü oder einem USER Menü sehr ähnlich sein.

- Ist der Befehl STO das erste Element der Liste, wobei eine Reihe von Namen folgt, dann erzeugt MENU ein *CUSTOM Eingabemenü*. Dieses Menü sieht wie ein Löser-Menü aus und wirkt wie ein solches: Das Drücken einer Menütaste nimmt einen Wert vom Stack und speichert ihn in der entsprechenden Variablen. Beispiele hierzu finden Sie im Kapitel 27, "Interaktive Programme".
- Enthält die Liste eine Reihe von Namen und Befehlen (STO ist nicht als erstes Element gespeichert), so erzeugt MENU ein *CUSTOM Benutzermenü*. Dieses Menü wirkt wie eine Verbindung des USER Menüs und eines Befehlsmenüs. Ein Beispiel dazu finden Sie unter "Wechseln von Verzeichnissen" auf Seite 275.

22

Der Befehlskatalog

In Kapitel 1 haben Sie den Befehlskatalog zum Überprüfen der korrekten Schreibweise verschiedener Befehle und zum Ansehen der möglichen Argumente für die Funktion "+" verwendet. Dieses Kapitel wiederholt die im Katalog verfügbaren Operationen, einschließlich des USAGE Menüs, welches die zulässigen Kombinationen von Argumenten auflistet.

Durch Drücken von ■ CATALOG wird der Befehl ABORT, der den ersten alphabetisch gelisteten Befehl darstellt, sowie das CATALOG Menü angezeigt.

Taste	Bedeutung
NEXT	Zeigt den nächsten Katalogeintrag an. Wenn Sie diese Taste gedrückt halten, wird der Katalog "durchgeblättert", bis Sie die Taste wieder freigeben.
PREV	Zeigt den vorherigen Katalogeintrag an. Wenn Sie diese Taste gedrückt halten, wird der Katalog rückwärts "durchgeblättert", bis Sie die Taste wieder freigeben.
USE	Aktiviert die Anzeige des USAGE Menüs (siehe unten), welches die Stackargumente des jeweiligen Befehls anzeigt.
FETCH	Verläßt den Befehlskatalog und schreibt den zuletzt angezeigten Befehl in die Befehlszeile.
QUIT	Verläßt den Befehlskatalog, ohne die Befehlszeile zu verändern.

Sie können den Katalog verlassen und die momentane Befehlszeile löschen, indem Sie die Taste ON drücken.

Auffinden eines Befehls

Sie können die Tasten des linken Tastenfelds zum Aufsuchen eines bestimmten Anfangszeichens verwenden.

- Das Drücken einer Buchstabentaste zeigt den ersten Befehl an, welcher mit dem zugehörigen Buchstaben beginnt. Existiert kein entsprechender Befehl, so wird der letzte Befehl angezeigt, der mit dem vorherhenden Buchstaben beginnt.
- Das Drücken eines Sonderzeichens (z.B. ■ Σ) zeigt den ersten Befehl an, welcher mit dem zugehörigen Zeichen beginnt. Existiert kein entsprechender Befehl, so wird der letzte Befehl angezeigt, der mit dem vorhergehenden Sonderzeichen beginnt.
- Das Drücken von ■ MENUS führt zur Anzeige von →STR, der letzte Eintrag im Befehlskatalog.

Überprüfen der Befehlssyntax

Sie können die zulässigen Stackargumente für den momentan angezeigten Befehl auf einfache Weise überprüfen. Drücken Sie dazu USE, was eine zweite Ebene des Befehlskatalogs—als USAGE Menü bezeichnet—aktiviert. Hier werden alle Kombinationen von Argumenten für den Befehl aufgelistet. Ist mehr als eine Kombination zulässig, so erscheinen zusätzlich nachstehend aufgeführte Menüfelder:

Taste	Bedeutung
NEXT	Zeigt die nächste Kombination von Argumenten.
PREV	Zeigt die vorherige Kombination von Argumenten.
QUIT	Bewirkt die Rückkehr zur Hauptanzeige des Befehlskatalogs. Sie können dann zu anderen Befehlen wechseln oder den Befehlskatalog verlassen, indem Sie erneut QUIT drücken.

Durch Drücken von ON können Sie das USAGE Menü sowie den Befehlskatalog verlassen, wobei gleichzeitig die momentane Befehlszeile gelöscht wird.

23

Auswertungen

Alle Rechneroperationen, von einfachen Berechnungen über das Tastenfeld bis zu komplexen Programmen, beinhalten eine oder mehrere Auswertungen. Als Beispiel:

- Wenn Sie ein oder mehrere Objekte in die Befehlszeile eintippen und [ENTER] drücken, wird die Befehlszeile in ein Programm übersetzt, welches anschließend ausgewertet wird.
- Wenn Sie im unmittelbaren Eingabemodus eine Taste im USER Menü drücken, wird der korrespondierende Name ausgewertet.
- Führen Sie eine Differentiation schrittweise aus, so drücken Sie [EVAL] zur Auswertung des Ausdrucks in Ebene 1.

Am einfachsten ist die Arbeitsweise des Rechners zu verstehen, wenn man an das *Verzögern von Auswertungen* und *Verursachen von Auswertungen* denkt. Obwohl der Term "Verzögern von Auswertungen" neu ist, so ist der Vorgang bereits bekannt: Bei jeder Eingabe eines Namens/algebraischen Ausdrucks in Anführungszeichen kennzeichnen die Begrenzungszeichen des Objekts, daß die Auswertung verzögert werden und das Objekt statt dessen in den Stack übernommen werden soll.

Die verzögerte Auswertung ist die Basis für jedes Programm. Der HP-28S erweitert dieses Konzept in einer einheitlichen Weise, um symbolische Operationen zu ermöglichen—Sie können Namen und algebraische Ausdrücke als Daten für symbolische Berechnungen verwenden. So wählen Sie z.B., wann (falls überhaupt) Sie einen Ausdruck auswerten möchten. Sie können ihn differenzieren, symbolisch lösen, Substitutionen für Variable vornehmen, und so weiter.

Dieses Kapitel beschreibt die Auswirkungen beim Auswerten verschiedener Objekte. Als allgemeine Einführung sollten Sie die folgenden *Objektklassen* betrachten:

- *Daten-Objekte*. Diese Klasse umfaßt reelle und komplexe Zahlen, Binärwerte, Strings, Felder und Listen. Der "Wert" eines Daten-Objekts entspricht genau seinem Inhalt.
- *Namen-Objekte*. Diese Klasse umfaßt globale und lokale Namen. Beim "Wert" eines Namens handelt es sich im allgemeinen um den Inhalt einer Variablen.
- *Prozedur-Objekte*. Diese Klasse umfaßt algebraische Ausdrücke und Programme. Der "Wert" einer Prozedur ist das Ergebnis des in ihr definierten Prozesses.

Diese Klassen definieren im groben, was beim Auswerten eines Objekts geschieht: Es gibt sich selbst zurück, den Inhalt einer Variablen oder das Ergebnis eines Prozesses. Es ist nicht ganz einfach, aber Sie erfahren nachstehend weitere Einzelheiten zu jeder Objektklasse.

Daten-Objekte

Diese stellen die einfachsten Objekte dar. Das Auswerten eines Daten-Objekts führt zum gleichen Objekt.

Beachten Sie, daß Listen mehrere Daten-Objekte darstellen, da Sie jeden Objekttyp enthalten können. Wenn Sie eine Liste mit Namen betrachten, so sind die Namen vor der Auswertung geschützt und können erst nach dem Entfernen aus der Liste ausgewertet werden.

Namen-Objekte

Im allgemeinen entpricht der "Wert" eines Namens dem Inhalt einer Variablen. Die Auswertung von lokalen Namen ist einfach und wird zuerst beschrieben, gefolgt von der Auswertung globaler Namen.

Auswerten von lokalen Namen

Wie in Kapitel 19 beschrieben, dient der Gebrauch lokaler Variablen der Vereinfachung von Stackmanipulationen. Der Zweck von lokalen Variablen liegt (1) in der Beseitigung des Variableninhalts vom Stack und (2) in der Rückgabe des Inhalts, wann immer dieser benötigt wird. Folglich wird bei der Auswertung eines lokalen Namens immer der Inhalt der korrespondierenden Variablen in den Stack zurückgegeben.

Auswerten von globalen Namen

Im allgemeinen verursacht die Auswertung eines globalen Namens die Auswertung des Inhalts der korrespondierenden globalen Variablen; oder mit anderen Worten: Das Auswerten eines globalen Namens hat den gleichen Effekt wie die Auswertung des in ihm gespeicherten Objekts.

Es gibt zwei Ausnahmen zu dieser Regel:

- Existiert keine Variable mit dem spezifizierten Namen, so wird der Name in den Stack zurückgegeben. Ein als Variable verwendeter undefinierter Name wird als *formale Variable* bezeichnet.
- Wenn der Inhalt der spezifizierten Variablen ein algebraischer Ausdruck ist, dann wird dieser *nicht ausgewertet*. Dadurch soll Ihnen die Fortsetzung symbolischer Berechnungen ermöglicht werden. Möchten Sie die Auswertung erzwingen, so verwenden Sie EVAL, während der Ausdruck in Ebene 1 gespeichert ist. (Soll ein algebraischer Ausdruck wiederholt ausgewertet werden, um ein numerisches Ergebnis zu erzielen, so führen Sie →NUM aus.)

Enthält die Variable ein Daten-Objekt, so entspricht die Auswertung des Variablennamens dem Zurückrufen des Variableninhalts. Allerdings kann die Auswertung eines Variablennamens zu einer langen Kette von Auswertungen führen. Ist z.B. in einer Variablen ein Name gespeichert, der eine weitere Variable definiert, die ebenfalls einen Variablennamen enthält, so führt die Auswertung des ersten Variablennamens letztlich zur Auswertung des Inhalts der dritten Variablen.

Hinweis

Erzeugen Sie keine Variable mit ihrem Namen als Inhalt (z.B. eine Variable X mit dem Inhalt 'X'). Das Auswerten einer solchen Variablen würde eine Endlosschleife zur Folge haben. Um eine Endlosschleife abzubrechen, müssen Sie einen Systemstopp (ON ▲) durchführen, welcher auch den Stackinhalt löscht).

Gleiches gilt für Variablen, welche sich zyklisch auf andere Variablen beziehen. Das Auswerten einer Variablen, die in einer ringförmigen Definition enthalten ist, führt ebenfalls zu einer Endlosschleife.

Prozedur-Objekte

Im allgemeinen besteht der "Wert" einer Prozedur aus dem Ergebnis des in ihr definierten Prozesses. Programme stellen die globaleren Prozedur-Objekte dar und sind deshalb zuerst beschrieben; anschließend erfolgt eine Beschreibung der algebraischen Ausdrücke.

Auswerten von Programmen

Ein Programm besteht aus einer Folge von Objekten und Befehlen. In diesem Handbuch wird der Term "Auswerten eines Programms" und "Ausführen eines Programms" abwechselnd benutzt. Bei der Auswertung eines Programms wird, allgemein ausgedrückt, der jeweilige Inhalt schrittweise ausgeführt, indem jedes Objekt in den Stack übernommen und jeder Befehl ausgewertet wird. Es sollten zwei zusätzliche Punkte beachtet werden:

- Nicht in Anführungszeichen stehende Namen werden ausgewertet, während Namen in Anführungszeichen im Stack gespeichert werden. Namen werden aus Gründen einer beabsichtigten Verzögerung ihrer Auswertung in Anführungszeichen gesetzt, wie auf Seite 57 besprochen.
- Programmstrukturen werden entsprechend ihrer eigenen Gesetze ausgewertet. In Teil 1 haben Sie mehrere Benutzerfunktionen geschrieben, welche eine *lokale Variablenstruktur* enthalten. Programmstrukturen sind in Kapitel 26 beschrieben.

Die Gesetze zur Auswertung von Namen und Programmen führen zu einer der fundamentalen Ideen beim Programmieren des HP-28S. In der folgenden Diskussion bedeutet "Programm" das in einer Variablen gespeicherte Programm, während mit "Name eines Programms" der Name der Variable, in welcher das Programm gespeichert ist, gemeint ist.

Die grundlegende Idee liegt in der *strukturierten Programmierung*. Dies bedeutet, daß eine umfangreichere Problemstellung in mehrere Teilaufgaben aufgegliedert und für jede dieser Teilaufgaben ein *Unterprogramm* geschrieben wird. Das Hauptprogramm kann nun relativ einfach aussehen und die prinzipielle Logik der Lösung darstellen. Es kann jede Teilaufgabe lösen, indem einfach der Name (ohne Anführungszeichen) des Unterprogramms mit aufgenommen wird. Soll die Ausführung öfters erfolgen, dann ist lediglich dessen Name an der entsprechenden Stelle aufzunehmen. Wenn andere Hauptprogramme das gleiche Unterprogramm verwenden, können diese das Unterprogramm in der gleichen Weise ausführen.

Stukturierte Programmierung ist unter "Vollständiges Erweitern und Zusammenfassen" auf Seite 253, "Anzeigen von Binärwerten" auf Seite 257 und "Median von statistischen Daten" auf Seite 270 beschrieben.

Auswerten von algebraischen Ausdrücken

Jeder Ausdruck ist gleichwertig mit einem Programm, welches ausschließlich nicht in Anführungszeichen stehende Namen und Funktionen enthält. Auswerten eines algebraischen Ausdrucks führt zum gleichen Ergebnis wie die Auswertung des korrespondierenden Programms: Nicht in Anführungszeichen eingeschlossene Namen werden ausgewertet und Funktionen werden ausgeführt.

Das Ergebnis der Auswertung eines Namens hängt von der Existenz einer Variablen mit dem entsprechenden Namen ab, wie oben unter "Auswerten von globalen Namen" beschrieben. Einige Beispiele:

- Verweist ein Name auf eine Benutzerfunktion, so können Sie den Namen der Benutzerfunktion wie eine interne Funktion verwenden. Die Auswertung des Ausdrucks verursacht die Ausführung der Benutzerfunktion. Die Argumente für die Benutzerfunktion, welche in Klammern eingeschlossen sind und dem Namen der Benutzerfunktion folgen, sind Teil des algebraischen Ausdrucks.

- Verweist ein Name auf ein Programm, welches keine Argumente vom Stack nimmt und genau ein Argument in den Stack zurückgibt, dann können Sie den Programmnamen zum (indirekten) Bezug auf das Ergebnis verwenden. Die Auswertung des algebraischen Ausdrucks bewirkt die Ausführung des Programmstruktur, wodurch tatsächlich der Programmname durch das Ergebnis ersetzt wird. Beispiele hierzu finden Sie unter "Summations-Statistik" auf Seite 262.
- Bezieht sich ein Name auf einen weiteren Ausdruck, so verursacht die Auswertung des ersten Ausdrucks *nicht* die Auswertung des zweiten Ausdrucks. Statt dessen wird der Name des zweiten Ausdrucks durch dessen Inhalt ersetzt.

Ein Sonderfall unter den Funktionen bildet die Funktion "=", welche Gleichungen von Ausdrücken unterscheidet. In Abhängigkeit des verwendeten Modus (symbolisch oder numerisch) ergibt die Ausführung von = eine Gleichung oder ein numerisches Ergebnis.

- Im symbolischen Ergebnismodus erzeugt die Auswertung einer Gleichung eine neue Gleichung. Der neue linke Ausdruck ist das Ergebnis der Auswertung des ursprünglichen linken Ausdrucks; gleiches gilt auch für den rechten Ausdruck.
- Im numerischen Ergebnismodus erzeugt das Auswerten einer Gleichung die numerische Differenz zwischen dem ursprünglichen linken Ausdruck und dem neuen Ausdruck.

Der nächste Abschnitt beschreibt die unterschiedlichen Ergebnismodi etwas detaillierter.

Auswerten von Funktionen

Die Aktionen bei der Auswertung einer Funktion sind vom eingestellten Ergebnismodus, welcher symbolisch oder numerisch sein kann, abhängig. Diese Modi sind ebenfalls im nächsten Kapitel, "Betriebsmodi", beschrieben.

Symbolischer Ergebnismodus. Dieser Modus dient als Standard-Einstellung; in ihm akzeptiert eine Funktion symbolische Argumente und gibt symbolische Argumente zurück. Die Wirkung von Funktionen im symbolischen Modus ist offensichtlich, wenn Sie Berechnungen mit Namen und Ausdrücken zum Erzeugen größerer Ausdrücke anstellen.

Numerischer Ergebnismodus. Dieser alternative Modus wird bei der Darstellung von Grafiken und vom Löser verwendet. Sein Zweck liegt in der Sicherstellung eines numerischen Ergebisses bei der Auswertung einer Funktion. In diesem Modus wiederholen Funktionen die Auswertung von symbolischen Argumenten, wobei nur numerische Argumente akzeptiert und numerische Ergebnisse zurückgegeben werden.

Sie können durch den Befehl →NUM erzwingen, daß ein Objekt so oft ausgewertet wird, bis ein numerisches Ergebnis vorliegt; in Kapitel 5 diente diese Anwendung zur Anzeige eines numerischen Werts für π.

Hinweis Sie sollten im numerischen Ergebnismodus keine Variable auswerten, die unter anderem ihren eigenen Namen als Inhalt hat (wie z.B. eine Variable X mit dem Ausdruck 'X+Y'). Die Auswertung solch einer Variablen verursacht eine Endlosschleife. Um eine Endlosschleife anzuhalten, müssen Sie einen Systemstopp ausführen (ON ▲), wodurch gleichzeitig der Stackinhalt gelöscht wird.

Gleiches gilt für Variablen, welche sich zyklisch auf andere Variablen beziehen. Das Auswerten einer Variablen, die in einer ringförmigen Definition enthalten ist, führt ebenfalls zu einer Endlosschleife.

24

Betriebsmodi

Sie können die Ergebnisse vieler Operationen durch die entsprechende Wahl eines Modus beeinflussen. Einige Modi (z.B. Winkelmodus) werden über eine Menütaste spezifiziert, was durch ein kleines Quadrat im zugehörigen Menüfeld gekennzeichnet wird. Wählen Sie z.B. das Bogenmaß als Winkelmodus, so erscheint das Menüfeld als RAD▪ .

Die meisten Modi (wie z.B. Akustiksignal ein oder aus) werden durch das Setzen oder Löschen von Benutzerflags—über die Befehle SF (*Set Flag*) und CF (*Clear Flag*)—spezifiziert. Zum Beispiel steuert Flag 51 das Akustiksignal und läßt sich über den Befehl 51 SF ausschalten.

Dieses Kapitel beschreibt die Einflüsse der verschiedenen Modi auf die Funktionsweise des Rechners und listet die zugehörigen Menüfelder und Flags auf. Außerdem werden die Indikatoren gezeigt, welche bei der Wahl des jeweiligen Modus erscheinen. Für jeden Modus ist zuerst die Voreinstellung aufgeführt, welche nach einem Speicherverlust aktiviert wird.

Allgemeine Modi

Diese Modi beziehen sich auf Berechnungen und den Tonsignalgeber.

Winkelmodus

Dieser Modus legt fest, ob die reelle Zahl in Grad oder im Bogenmaß interpretiert werden soll. Dies bezieht sich auf die Argumente von trigonometrischen Funktionen sowie die Ergebnisse derer Inversen.

Grad-Modus (DEG , Flag 60 gelöscht). Reelle Zahlen stellen ein Winkelmaß in Grad dar.

Radiant-Modus (RAD , Flag 60 gesetzt, (2π)). Reelle Zahlen stellen ein Winkelmaß im Bogenmaß dar.

Akustiksignal-Modus

Dieser Modus steuert die Ausgabe eines Akustiksignals im Fall einer fehlerhaften Eingabe/Verarbeitung.

Akustiksignal ein (Flag 51 gelöscht). Der Rechner gibt einen Ton aus.

Akustiksignal aus (Flag 51 gesetzt). Der Rechner bleibt "stumm".

Hauptwert

Eine von ISOL oder QUAD ermittelte Lösung erfordert im allgemeinen beliebige Vorzeichen (+1 oder −1) und ganzzahlige Werte (0, 1, 2, ...) zur Darstellung aller möglichen Lösungen. Dieser Modus legt fest, ob beliebige Vorzeichen und ganzzahlige Vielfache in den Lösungen enthalten sind.

Hauptwert aus (Flag 34 gelöscht). Lösungen von ISOL und QUAD schließen die Variablen s1, s2, ... für beliebige Vorzeichen und n1, n2, ... für ganzzahlige Vielfache ein.

Hauptwert ein (Flag 34 gesetzt). ISOL und QUAD betrachten beliebige Vorzeichen als +1 und ganzzahlige Vielfache als 0.

Konstanten-Modus

Dieser Modus wirkt sich auf die Auswertung von symbolischen Kontstanten (e, i, MINR, MAXR oder π) aus. Im numerischen Ergebnismodus (Flag 36 gelöscht) ergibt die Auswertung einer symbolischen Konstante deren numerischen Wert, unabhängig vom Konstanten-Modus.

Symbolische Konstanten (Flag 35 gesetzt). Die Auswertung einer symbolischen Konstanten ergibt ihre symbolische Form.

Numerische Konstanten (Flag 35 gelöscht). Die Auswertung einer symbolischen Konstante ergibt ihren numerischen Wert.

Ergebnismodus

Der aktuelle Ergebnismodus beeinflußt das Ergebnis bei der Auswertung einer Funktion mit symbolischen Argumenten.

Symbolische Ergebnisse (Flag 36 gesetzt). Sind symbolische Argumente vorgegeben, so ergeben Funktionen symbolische Ergebniss.

Numerische Ergebnisse (Flag 36 gelöscht). Funktionen ergeben immer numerische Ergebnisse. Die Auswertung einer symbolischen Konstante ergibt immer deren numerischer Wert, unabhängig vom Konstanten-Modus.

Eingabe- und Anzeigemodi

Diese Modi beeinflussen die Eingabe und Anzeige von Objekten.

Eingabemodus

Der momentane Eingabemodus beeinflußt die durch das Drücken einer Befehls-, Funktions- oder USER-Menütaste erzielten Ergebnisse. Er ändert sich automatisch, wenn Sie ['], ["] oder [«] drücken, wobei eine manuelle Änderung durch Drücken von [α] möglich ist. Außerdem ändert sich die Form des Cursors, um den jeweiligen Modus zu kennzeichnen.

Unmittelbarer Eingabemodus (Offener Cursor). Die Befehlszeile wird beim Drücken einer Befehls-, Funktions- oder USER-Menütaste ausgeführt.

Algebraischer Eingabemodus (Teilweise ausgefüllter Cursor). Die Befehlszeile wird nach dem Drücken einer Befehlstaste ausgeführt.

Alpha-Eingabemodus (Ausgefüllter Cursor, α). Die Befehlszeile wird nur nach Drücken von [ENTER] ausgeführt.

Ersetzungs- oder Einfügungsmodus

Das Drücken von [INS] im Cursor Menü wechselt zwischen Ersetzungs- und Einfügungsmodus. Der jeweilige Modus wird durch die Form des Cursors gekennzeichnet.

Ersetzungsmodus (Rechtecks-Cursor). Die neuen Zeichen ersetzen die vorhandenen Zeichen.

Einfügungsmodus (Pfeil-Cursor). Neue Zeichen werden zwischen den vorhandenen Zeichen eingefügt.

Groß- oder Kleinschreibung

Das Drücken von [LC] schaltet zwischen Groß- und Kleinschreibung um.

Großschreibung. Durch Drücken einer Buchstabentaste wird das Zeichen in Großschreibung dargestellt.

Kleinschreibung. Durch Drücken einer Buchstabentaste wird das Zeichen in Kleinschreibung dargestellt.

Anzeige in Ebene 1

Viele Objekte sind zu groß, um in einer Zeile angezeigt werden zu können. Sie können spezifizieren, daß—falls erforderlich—mehrere Zeilen zur Anzeige von Objekten in Ebene 1 verwendet werden, oder daß generell nur eine Zeile verwendet wird.

ML ein (ML• , Flag 45 gesetzt). Objekte in Ebene 1 werden gegebenenfalls in mehr als einer Zeile angezeigt.

ML aus (ML , Flag 45 gelöscht). Objekte in Ebene 1 werden nur in einer Zeile dargestellt.

Dezimalzeichen

Sie können das Komma oder den Punkt als Dezimalzeichen und als Trennzeichen zwischen Objekten (in der Befehlszeile) spezifizieren, wobei das Leerzeichen immer als Trennzeichen verwendet werden kann.

RDX, aus (RDX. , Flag 48 gelöscht). Als Dezimalzeichen wird der Punkt verwendet und das Komma dient als Trennzeichen zwischen Objekten.

RDX, ein (RDX, , Flag 48 gesetzt). Als Dezimalzeichen wird das Komma verwendet und der Punkt dient als Trennzeichen zwischen Objekten.

Zahlenformat

Diese Modi legen die Anzahl angezeigter Dezimalstellen fest. Die Befehle FIX, SCI und ENG erfordern ein reellwertiges Argument *n*. Das momentane Zahlen-Anzeigeformat beeinflußt auch den Befehl RND (*RuNDen*).

STD Format (STD). Reelle Zahlen werden mit einem Dezimalpunkt oder, falls notwendig, mit einem Exponenten dargestellt.

FIX Format (FIX). Reelle Zahlen werden mit *n* Dezimalstellen dargestellt. Ein Exponent wird nur bei Bedarf angezeigt.

SCI Format (SCI). Reelle Zahlen werden als Mantisse (kleiner als 10 und mit *n* Dezimalstellen) und Exponent dargestellt.

ENG Format (ENG). Reelle Zahle werden als Mantisse, welche *n* + 1 Ziffern enthält, und einem Exponenten, der ein Mehrfaches von 3 ist, dargestellt.

Zahlenbasis

Für die Eingabe und Anzeige von Binärwerten haben Sie die Auswahl zwischen mehreren Zahlensystemen. Die Wahl der Basis hat keinen Einfluß auf die interne Struktur von Binärwerten, welche immer als Bitfolge betrachtet werden.

DEC Basis (DEC•). Ohne Basiskennzeichen eingegebene Binärwerte werden immer als Dezimalzahl interpretiert. Hier werden alle Binärwerte als Dezimalzahl, gefolgt von einem "d", angezeigt.

HEX Basis (HEX•). Ohne Basiskennzeichen eingegebene Binärwerte werden immer als Hexadezimalzahl interpretiert. Hier werden alle Binärwerte als Hexadezimalzahl, gefolgt von einem "h", angezeigt.

OCT Basis (OCT•). Ohne Basiskennzeichen eingegebene Binärwerte werden immer als Oktalzahl interpretiert. Hier werden alle Binärwerte als Oktalzahl, gefolgt von einem "o", angezeigt.

BIN Basis (BIN•). Ohne Basiskennzeichen eingegebene Binärwerte werden immer als Dualzahl interpretiert. Hier werden alle Binärwerte als Dualzahl, gefolgt von einem "b", angezeigt.

Wortlänge für Binärwerte

Die momentane Wortlänge kann von 1 bis 64 Bits variieren. Sie steuert die Anzeige von Binärwerten; außerdem werden Binärwerte auf die momentane Wortlänge abgekürzt, wenn sie als Argumente verwendet oder als Ergebnisse zurückgegeben werden. Die Wortlänge kann über den Befehl *n* `STWS` (*STore WordSize*) auf *n* Bits gesetzt werden.

Rücksicherungsmodi

Diese Modi bestimmen, ob Kopien von Befehlszeilen, Stack und Argumenten für Befehle erzeugt werden. Sie haben dadurch im Fehlerfall die Möglichkeit, die vorherigen Daten zurückzusichern.

CMD Modus

Dieser Modus bestimmt, ob beim Drücken von [ENTER] (oder einer Taste, die ENTER ausführt) eine Kopie der Befehlszeile erstellt wird.

CMD ein (CMD•). Befehle werden für eine Rücksicherung durch ■[COMMAND] gesichert.

CMD aus (CMD **).** Befehlszeilen werden nicht gesichert.

UNDO Modus

Dieser Modus bestimmt, ob beim Drücken von ENTER (oder einer Taste, welche ENTER ausführt) eine Kopie des Stacks erstellt wird.

UNDO ein (UNDO* **).** Der Stack wird für eine Rücksicherung durch ■ UNDO gesichert.

UNDO aus (UNDO **).** Der Stackinhalt wird nicht gesichert.

LAST Modus

Dieser Modus legt fest, ob Kopien der Argumente bei der Ausführung von Befehlen erzeugt werden.

LAST ein (Flag 31 gesetzt, LAST* **).** Argumente werden für eine Rücksicherung durch LAST oder im Fehlerfall gesichert.

LAST aus (Flag 31 gelöscht, LAST **).** Argumente werden nicht gesichert. Tritt ein Fehler ein, so werden die Argumente des letzten Befehls nicht in den Stack zurückgegeben.

Mathematische Ausnahmen

Bestimmte Fehler, welche während gewöhnlicher Berechnungen mit reellen Zahlen auftreten können, werden als *mathematische Ausnahmen* bezeichnet. Eine Ausnahme kann als gewöhnlicher Fehler wirken und die Berechnung anhalten, oder ein Standard-Ergebnis vorgeben und die Fortsetzung der Berechnung ermöglichen.

Unendliches-Ergebnis-Behandlung

Diese Ausnahme tritt auf, wenn das Ergebnis einer Berechnung "unendlich" wäre (z.B. `'LN(0)'`, `'TAN(90)'` (in Grad), oder `'X/0'`).

Unendliches-Ergebnis-Fehler ein (Flag 59 gesetzt). Ein "unendliches Ergebnis" wird als Fehler interpretiert.

Unendliches-Ergebnis-Fehler aus (Flag 59 gelöscht). Ein unendliches Ergebnis gibt den Wert ±9,99999999999E499 zurück und setzt den zugehörigen Indikator (Flag 64).

Überlauf-Behandlung

Eine Überlauf-Ausnahme tritt ein, wenn eine Berechnung zu einem endlichen Ergebnis führen würde, dessen Absolutbetrag größer als die größte im Rechner darstellbare Zahl ist (z.B. die Auswertung von `'9E499+9E499'`, `'EXP(5000)'` oder `'FACT(2000)'`).

Überlauf-Fehler aus (Flag 58 gelöscht). Überlauf-Ausnahmen geben ±9,99999999999E499 zurück und setzen den Überlauf-Indikator (Flag 63).

Überlauf-Fehler ein (Flag 58 gesetzt). Überlauf-Ausnahmen werden als Fehler interpretiert.

Unterlauf-Behandlung

Eine Unterlauf-Ausnahme tritt ein, wenn eine Berechnung zu einem endlichen Ergebnis führen würde, dessen Absolutbetrag kleiner als die kleinste im Rechner darstellbare Zahl ist (z.B. die Auswertung von `'1E-499/2'` oder `'EXP(-5000)'`).

Unterlauf-Fehler aus (Flag 57 gelöscht). Unterlauf-Ausnahmen geben 0 als Standardergebnis zurück und setzen den Unterlauf-Indikator + (Flag 62) oder den Unterlauf-Indikator −, in Abhängigkeit vom tatsächlichen Vorzeichen.

Unterlauf-Fehler ein (Flag 57 gesetzt). Unterlauf-Ausnahmen werden als Fehler interpretiert und geben die Meldung `Negative Underflow` oder `Positive Underflow` zurück, abhängig vom Vorzeichen des tatsächlichen Ergebnisses.

Druckmodi

Die folgenden Modi geben Ihnen eine größere Flexibilität bei Druckoperationen.

Drucken eines Protokolls

Sie können automatisch ein Protokoll Ihrer Berechnungen erzeugen.

Protokolldruck aus (TRAC , Flag 32 gelöscht). Es erfolgt kein automatisches Drucken Ihrer Berechnungen.

Protokolldruck ein (TRAC• , Flag 32 gesetzt). Bei jeder Ausführung der Befehlszeile druckt der Rechner den Inhalt der Befehlszeile, die die Ausführung verursachende Operation und das Ergebnis in Ebene 1.

Automatischer CR Modus

Gewöhnlich möchten Sie Daten an den Drucker übertragen und diese mit einem einzigen Befehl ausdrucken. In anderen Fällen, wie z.B. beim Drucken von Grafiken, sollen die Daten im Druckpuffer gesammelt werden, ohne sofort ausgedruckt zu werden. Dieser Modus legt fest, ob die Druckbefehle automatisch das Drucken veranlassen.

Auto CR (Flag 33 gelöscht). Druckbefehle senden am Ende der Zeichenübertragung ein *Carriage Right* Zeichen, was den Ausdruck der Daten verursacht.

Ansammeln der Druckdaten (Flag 33 gesetzt). Druckbefehle enthalten kein CR-Zeichen, was zum Ansammeln der Daten im Druckpuffer führt.

Druckgeschwindigkeit

Der Rechner kann nicht feststellen, ob der Drucker zum Empfang weiterer Daten bereit ist oder nicht, und berechnet daher eine sichere Übertragungsrate. Dieser Modus bestimmt, ob die Berechnung für einen mit Batterien oder über ein Netzteil betriebenen Drucker erfolgen soll.

Normale Druckgeschwindigkeit (Flag 52 gelöscht). Die Datenübertragungsrate ist auf einen mit Batterien betriebenen Drucker ausgerichtet.

Erhöhte Druckgeschwindigkeit (Flag 52 gesetzt). Die Datenübertragungsrate ist auf einen über Adapter betriebenen Drucker ausgerichtet.

Zeilenabstand

Dieser Modus bestimmt, ob Leerzeilen automatisch gedruckt werden.

Einfacher Zeilenabstand (Flag 47 gelöscht). Es werden nicht automatisch Leerzeilen gedruckt.

Doppelter Zeilenabstand (Flag 47 gesetzt). Es wird automatisch zwischen den Textzeilen jeweils eine Leerzeile gedruckt.

25

Systemoperationen

Dieses Kapitel beschreibt spezielle Tastenkombinationen, die gewöhnliche Rechneroperationen unterbrechen. Diese Systemoperationen schließen das Drucken des Anzeigeinhalts, Einstellen des Anzeigekontrasts, Anhalten von Programmen, Zurücksetzen des Speicherbereichs und das Ausführen von Diagnosetests mit ein.

Alle Systemoperationen beginnen mit dem Drücken der Taste [ON]. Sie können jede Operation wieder aufheben, indem Sie [DEL] drücken, bevor Sie [ON] wieder freigeben.

Die nachstehende Tabelle listet die Tastenfolge der Systemoperationen, worauf eine kurze Beschreibung jeder Operation folgt.

Systemoperationen

Name	Tastenfolge
Anzeigeinhalt drucken	[ON] [L]
Kontrasteinstellung	[ON] [+] oder [ON] [−]
Grundstellung	[ON]
Systemstopp	[ON] [▲]
Speicher zurücksetzen	[ON] [INS] [▶]
Elektronik-Test	[ON] [◀]
Tastenfeld-Test	[ON] [NEXT]
Systemoperation aufheben	[ON] [DEL]

Drucken des Anzeigeinhalts

Um den Inhalt der momentanen Anzeige zu drucken:

1. Halten Sie [ON] gedrückt.
2. Drücken Sie [L] (die Taste, über der "PRINT" steht).
3. Geben Sie [ON] frei.

Kontraststeuerung

Um den Kontrast der Anzeige zu ändern:

1. Halten Sie [ON] gedrückt.
2. Drücken Sie [+], um den Kontrast zu erhöhen oder [−], um den Kontrast zu verringern. Sie können, solange Sie [ON] gedrückt halten, [+] oder [−] wiederholt drücken oder ständig gedrückt halten, um so die beste Kontrasteinstellung zu finden.
3. Geben Sie [ON] frei.

Löschoperationen

Es gibt drei Löschoperationen, die nachfolgend in der Reihenfolge mit zunehmender Wirkung gelistet sind.

Grundstellung

Um zur normalen Stackanzeige zurückzukehren, führen Sie eine Grundstellung durch Drücken von [ON] aus. In manchen Fällen müssen Sie [ON] zweimal drücken. Grundstellung hat dabei folgende Wirkung:

- Löscht die Befehlszeile.
- Hebt alle Ausführungen von Befehlen oder Prozeduren auf.
- Verläßt besondere Operationen wie FORM und PLOT, und schließt die Anzeige eines Katalogs ab.
- Startet erneut die normale Wirkungsweise des Tastenfelds.

Systemstopp

Um ein Programm anzuhalten, das nicht auf [ON] reagiert, führen Sie einen Systemstopp folgendermaßen aus:

1. Halten Sie [ON] gedrückt.
2. Drücken Sie [▲].
3. Geben Sie [ON] frei.

Ein Systemstopp hat folgende Wirkung:

- Alle Auswirkungen von Grundstellung.
- Löscht alle Programme mit unterbrochener Ausführung sowie alle lokalen Variablen.
- Löscht den Stack.
- Löscht zur Rücksicherung gespeicherte Elemente (CMD, UNDO, LAST).
- Löscht das CUSTOM Menü.
- Wählt HOME als aktuelles Verzeichnis.
- Aktiviert das Cursor-Menü.

Zurücksetzen des Speichers (Memory Reset)

Um den ganzen Speicherbereich zurückzusetzen:

1. Halten Sie [ON] gedrückt.
2. Halten Sie [INS] und [▶] gedrückt.
3. Geben Sie [INS] und [▶] frei.
4. Geben Sie [ON] frei.

Das Zurücksetzen des Speichers hat folgende Wirkung:

- Alle Auswirkungen von Grundstellung und Systemstopp.
- Löscht alle Benutzervariablen und Verzeichnisse.
- Setzt alle Benutzerflags auf ihre Standardwerte zurück.
- Gibt ein Tonsignal aus und zeigt in Zeile 1 `Memory Lost` an.

Testoperationen

Es gibt zwei Systemoperationen zum Testen der Funktionsweise des Rechners. Die erste testet wiederholt den elektronischen Teil. Dieser Test verläuft selbständig. Die zweite ist ein Tastenfeld-Test, wozu Sie alle Tasten in einer bestimmten Reihenfolge drücken müssen. Beide Tests führen einen Systemstopp durch.

Elektronik-Test

Um den sich wiederholenden Elektronik-Test auszuführen:

1. Starten Sie den Test.
   a. Halten Sie [ON] gedrückt.
   b. Drücken Sie [◀].
   c. Geben Sie [ON] frei.
2. In der Anzeige erscheinen horizontale und vertikale Linien, danach eine leere Anzeige, anschließend zufällige Zeichenmuster und dann, bevor sich der Test automatisch wiederholt, kurz das Testergebnis.
   - Die Meldung `OK-28S` bedeutet, daß der Rechner den Test bestanden hat.
   - Eine Meldung wie z.B. `1 FAIL` bedeutet, daß der Rechner den Test nicht bestanden hat. Die Zahl gibt die Fehlerart an.

     Wenn Sie den Test durch Drücken einer Taste abbrechen, wird eine Fehlermeldung angezeigt (der Test erwartet keinen Tastendruck). *Diese Art von Fehlermeldung bedeutet nicht, daß der Rechner funktionsgestört ist.*
3. Verlassen Sie den Test, indem Sie einen Systemstopp durchführen.
   a. Halten Sie [ON] gedrückt.
   b. Drücken Sie [▲].
   c. Geben Sie [ON] frei.

Tastenfeld-Test

Um einen Tastenfeld-Test durchzuführen:

1. Starten Sie den Test.
   a. Halten Sie [ON] gedrückt.
   b. Drücken Sie [NEXT].
   c. Geben Sie [ON] frei.
2. Der Rechner zeigt `KEYBOARD TEST` an.
   a. Testen Sie zuerst die oberste Reihe des linken Tastenfelds, indem Sie nacheinander [A] [B] [C] [D] [E] [F] drücken.
   b. Testen Sie entsprechend die zweite bis zur sechsten Reihe.
   c. Testen Sie die erste Reihe des rechten Tastenfelds, indem Sie nacheinander [INS] [DEL] [▲] [▼] [◀] [▶] drücken.
   d. Testen Sie entsprechend die zweite bis zur siebten Reihe. (Drücken Sie die [ON] Taste in der richtigen Reihenfolge—der Test wird dadurch nicht unterbrochen.)
3. Wenn Sie alle Tasten in der richtigen Reihenfolge gedrückt haben und der Rechner fehlerfrei funktioniert, wird `OK-285` angezeigt. Eine Meldung wie z.B. `1 FAIL` bedeutet, daß Sie entweder nicht die richtige Reihenfolge eingehalten haben oder der Rechner den Test nicht bestanden hat. Die angezeigte Zahl gibt die Fehlerart an.
4. Drücken Sie [ON].

Teil 3

Programmierung

26

Programmstrukturen

Viele Programme entsprechen einer Folge von unmittelbar ausgeführten Tastenoperationen. Die Objekte werden in den Stack übernommen und Befehle werden ausgeführt, um das gewünschte Ergebnis zu berechnen. Diese Programme sind eigentlich nur eine Aufzeichnung der Objekte und Befehle in derselben Reihenfolge, wie sie über das Tastenfeld ausgeführt würden. Jedoch stehen Ihnen in Programmen weit mehr Möglichkeiten zur Verfügung als über einfache Tastenfolgen.

Beispielsweise haben Sie in Teil 1 ein Programm zum Erzeugen von lokalen Variablen geschrieben. Der spezielle Befehl →, gefolgt von einem oder mehreren Namen und von einer Prozedur, wird als *Struktur von lokalen Variablen* bezeichnet. Sie können den Befehl → nicht über die Tastatur ausführen; er muß im gleichen Programm wie die Namen und die Prozedur erscheinen, welche die gesamte Programmstruktur bilden.

In diesem Kapitel wird zuerst noch einmal die Struktur von lokalen Variablen behandelt. Anschließend werden weitere Programmstrukturen beschrieben, die Tests durchführen und ggf. die Programmausführung ändern. Die Befehle dafür erscheinen im PROGRAM BRANCH Menü. Lesen Sie unbedingt das erste Beispiel des Abschnitts "Bedingte Strukturen" auf Seite 223, das Sie in das im restlichen Teil dieses Kapitels verwendete Konzept einführt.

Strukturen von lokalen Variablen

In Teil 1 haben Sie mehrere Benutzerfunktionen geschrieben, welche die wichtigsten Anwendungen der Strukturen von lokalen Variablen darstellen. Es gibt zwei Forderungen an Benutzerfunktionen:

- Direkte Angabe der Argumente einer Funktion.
- Rückgabe von genau einem Ergebnis.

Zum Beispiel wurde die Benutzerfunktion COT (aus Kapitel 5) folgendermaßen geschrieben:

```
« → x 'INV(TAN(x))' »
```

Hier wird über die Struktur einer lokalen Variablen ein Argument in x abgespeichert (erfüllt die erste Forderung) und der Ausdruck `'INV(TAN(x))'` ausgewertet (erfüllt die zweite Forderung). Die Benutzerfunktion O→G (aus Kapitel 14) beinhaltet anstatt eines Ausdrucks ein Programm, welches jedoch genau ein Ergebnis zurückgibt. Dadurch erfüllt O→G ebenfalls die zweite Forderung.

Die beiden Forderungen gelten nur für Benutzerfunktionen. Im allgemeinen werden lokale Variablen als Ersatz für Stack-Operationen verwendet. Im folgenden Beispiel wird die Summe und die Differenz zweier Zahlen zurückgegeben. Da zwei Ergebnisse zurückgegeben werden, kann dies keine Benutzerfunktion sein.

```
« → x y « x y + x y - » »
```

Weitere Beispiele finden Sie in Kapitel 28. Dort werden Strukturen von lokalen Variablen vorwiegend zur Vermeidung von Stack-Operationen anstatt zur Erzeugung von Benutzerfunktionen verwendet.

Bedingte Strukturen

Bedingte Strukturen ermöglichen einem Programm, eine spezifizierte Bedingung abzufragen und, basierend auf dem Ergebnis, eine Entscheidung zu treffen. Dieser Abschnitt beginnt zuerst mit einem Beispiel, anhand dessen Programmstrukturen im allgemeinen und anschließend weitere Formen von bedingten Strukturen behandelt werden.

Angenommen, Sie schreiben ein Programm mit der Variablen x zur Berechnung von $(\sin x)/x$. Hier entsteht ein Problem, da der Quotient für den Fall $x = 0$ undefiniert ist. Im folgenden Beispiel wird für den Fall $x \neq 0$ $(\sin x)/x$ und für $x = 0$ der Wert 1 zurückgegeben.

```
IF X 0 ≠ THEN X SIN X / ELSE 1 END
```

Bei der Programmausführung ergibt sich dann folgender Ablauf:

1. Der IF Befehl markiert nur den Beginn der Struktur. Er kann überall vor dem THEN Befehl stehen.
2. X wird ausgewertet.
3. Die Zahl 0 wird in den Stack übernommen.
4. Die Funktion ≠ hat den Wert von X und 0 als Argumente.
   - Falls die Argumente "ungleich" sind, gibt ≠ "1" zurück.
   - Falls die Argumente *nicht* "ungleich" sind, gibt ≠ "0" zurück.
5. Der Befehl THEN hat 1 bzw. 0 als Argument.
   - Falls das Argument 1 ist, wertet THEN das Programm bis ELSE (d.h. `X SIN X /`) aus.
   - Falls das Argument 0 ist, wertet THEN das Programm von ELSE bis END (d.h. `1`) aus.
6. Die Programmausführung wird nach END fortgesetzt.

Nachfolgend werden zuerst einige allgemeine Informationen über Programmstrukturen gegeben.

Programmstruktur-Befehle. Die Befehle IF, THEN, ELSE und END sind Beispiele für *Programmstruktur-Befehle*. Die Reihenfolge und Bedeutung dieser Befehle ist ähnlich ihrer sprachlichen Anwendung. Programmstruktur-Befehle können nicht so flexibel wie andere Befehle verwendet werden; ihre Anwendung beschränkt sich auf die in diesem Kapitel beschriebenen Kombinationen.

Testfunktionen und Testbefehle. Die Funktion ≠ wird als *Testfunktion* bezeichnet. Ausgehend von zwei Zahlenwerten gibt ≠ entweder 1 oder 0 zurück, um anzuzeigen, ob das Ergebnis wahr oder falsch ist. Weitere Testfunktionen sind <, ≤, ≥,> und ==. (Beachten Sie, daß = für Gleichungen und nicht zum Testen der Gleichheit verwendet wird.) Wenn Testfunktionen symbolische Argumente beinhalten, so ist das Ergebnis ebenfalls symbolisch.

Desweiteren gibt es Testbefehle, die als Ergebnis immer 1 oder 0 zurückgeben. Beispielsweise sind der Testbefehl SAME und die Testfunktion == ähnlich, mit dem Unterschied, daß dieser Befehl überprüft, ob zwei Objekte identisch sind. Außerdem gibt es Testbefehle für den Gebrauch von Flags. Lesen Sie für zusätzliche Informationen über Testfunktionen und Testbefehle unter "PROGRAM TEST" im Referenzhandbuch nach.

Flags. Die Zahlen 1 und 0, die Testbefehle zurückgeben, werden als Stack*flags* bzeichnet. Da sie kennzeichnen, ob das Testergebnis wahr oder falsch ist, wird 1 als *Wahr-Flag* und 0 als *Falsch-Flag* bezeichnet.

Die Bezeichnung "Flag" bezieht sich ebenso auf die internen *Benutzerflags*. Diese sind von 1 bis 64 durchnumeriert, wobei die Flags 31 bis 64 für den Rechner eine besondere Bedeutung haben. Die Flags 1 bis 30 können dagegen zu beliebigen Wahr/Falsch-Unterscheidungen verwendet werden. Desweiteren kann ein Stackflag in einem Benutzerflag gespeichert werden, da beide einen Wahrheitswert darstellen. Beispielsweise wird durch die Sequenz

```
IF A B < THEN 12 SF ELSE 12 CF END
```

Flag 12 gesetzt, falls A $<$ B, oder Flag 12 gelöscht, falls A $\geqslant$ B. Später kann dann mit der Folge

```
IF 12 FS? THEN ...
```

geprüft werden, ob Flag 12 gesetzt ist. Diese Sequenz gibt denselben Wahrheitswert wie der ursprüngliche Test `A B <` zurück. Der Vorteil dieser Methode liegt darin, daß der Wahrheitswert des ursprünglichen Tests erhalten bleibt, selbst wenn die Werte für A und B verändert werden. Die Befehle zum Ändern und Testen von Benutzerflags finden Sie unter "PROGRAM TEST" im Referenzhandbuch. Im restlichen Teil diese Kapitels wird mit "Flag" ein Stackflag bezeichnet.

Anweisungen. Die Objekte und Befehle zwischen zwei Programmstruktur-Befehlen werden als *Anweisung* bezeichnet. Die Programmstruktur behandelt jede Anweisung einzeln. Eine Anweisung wird durch die logische Bedeutung oder den vorangehenden Befehl bezeichnet. Daraus folgt für das erste Beispiel:

- Die Anweisung zwischen IF und THEN (`X 0 ≠`) wird als *Test-Anweisung* bzw. *IF Anweisung* bezeichnet.
- Die Anweisung zwischen THEN und ELSE (`X SIN X /`) wird als *Wahr-Anweisung* bzw. *THEN Anweisung* bezeichnet.
- Die Anweisung zwischen ELSE und END (`1`) wird als *Falsch-Anweisung* bzw. *ELSE Anweisung* bezeichnet

Die Anweisungen in diesem Beispiel stellen einfache numerische Berechnungen dar. Im allgemeinen können Anweisungen aus einer beliebigen Sequenz von Objekten und Befehlen bestehen. Tatsächlich verhält sich eine Anweisung wie ein Unterprogramm innerhalb eines Programms. Wenn Sie für die Anweisung ein getrenntes Programm schreiben und dieses Programm in einer Variablen speichern, können Sie stellvertretend für die gesamte Anweisung den Variablennamen verwenden. In diesem Fall kann eine einfache Struktur wie

```
IF A THEN B ELSE C END
```

einen komplizierten Abfrageprozess mit zwei möglichen und komplizierten Ergebnissen darstellen, die von A, B und C abhängig sind.

IF ... THEN ... ELSE ... END

Unter Verwendung der hier definierten Terminologie, kann die Auswertung dieser bedingten Struktur folgendermaßen beschrieben werden: Die IF Anweisung wird ausgewertet und gibt ein Flag zurück. Falls das Flag wahr ist, wird die THEN Anweisung ausgewertet; falls das Flag falsch ist, wird dagegen die ELSE Anweisung ausgewertet.

Ein weiteres Beispiel zu diesem Strukturtyp finden Sie in dem Beispiel "FIB2 (Fibonacci-Reihe, Schleifenversion)" auf Seite 248.

IFTE (If-Then-Else-End Funktion)

Das erste Beispiel in diesem Kapitel kann mit Hilfe der Funktion IFTE in algebraischer Syntax geschrieben werden:

```
'IFTE(X≠0,SIN(X)/X,1)'
```

Diese Form ist besonders für symbolische Berechnungen praktisch. Wenn die Programmstruktur mit undefiniertem X ausgeführt wird, ergibt sich als Ergebnis diese algebraische Form. Die Argumente von IFTE müssen jedoch in algebraischer Syntax darstellbar sein; um UPN Befehle in der Bedingung aufnehmen zu können, muß die Programmstruktur-Form verwendet werden.

Die IFTE Funktion wird in "FIB1 (Fibonacci-Reihe, rekursive Version)" auf Seite 247 verwendet.

IF ... THEN ... END

Falls eine ELSE Anweisung nicht benötigt wird—d.h., die beiden Möglichkeiten lauten: etwas ausführen oder nichts ausführen—so kann in der Programmstruktur die ELSE Anweisung weggelassen werden. Im folgenden Beispiel wird sichergestellt, daß das Objekt in Ebene 1 größer als das Objekt in Ebene 2 ist, indem die Objekte notfalls vertauscht werden.

```
IF DUP2 ≤ THEN SWAP END
```

Beachten Sie die Verwendung von DUP2 zur Erstellung von Kopien der Objekte. Die Kopien werden dann durch den Vergleich ≤ zerstört. Ein weiteres Beispiel zu IF ... THEN ... END finden Sie unter "SORT (Sortieren einer Liste)" auf Seite 270.

IFT (If-Then-End Befehl)

Das vorherige Beispiel kann anstelle der Programmstruktur mit dem Befehl IFT geschrieben werden:

```
DUP2 ≤ « SWAP » IFT
```

Die Sequenz `DUP2 ≤` legt ein Flag im Stack ab, das Programm `« SWAP »` wird in den Stack übernommen, und der Befehl `IFT` übernimmt das Flag und das Programm als Argumente. Falls das Flag wahr ist, wird das Programm ausgewertet, ansonsten übersprungen. Das Ergebnis ist mit dem der Programmstruktur-Form identisch.

Fehlerbehandlung

In manchen Fällen können Sie vorhersagen, ob während des Programmablaufs ein Fehler auftreten kann. In der Regel wird durch einen Fehler die Programmausführung unterbrochen. Wenn Sie jedoch eine *Fehlerbehandlung* für den fehlerverursachenden Befehl vorsehen, dann kann das Programm auch nach Auftreten eines Fehlers weiter ablaufen.

Beachten Sie, daß $(\sin x)/x$ für $x = 0$ ein undefiniertes Ergebnis liefert. Eine andere Methode, $(\sin 0)/0 = 1$ zu definieren, wäre:

```
IFERR X SIN X / THEN DROP2 1 END
```

Dies bedeutet: Zuerst die IFERR Anweisung auswerten (`X SIN X /`). Falls ein Fehler auftritt, die THEN Anweisung (`DROP2 1`) auswerten.

Dieses Beispiel enthält den Befehl DROP2, um die beiden Nullen, die den Fehler verursacht haben, zu löschen. Beachten Sie, daß dabei angenommen wird, daß LAST aktiv ist. Falls LAST nicht aktiv ist, sind die beiden Nullen nicht vorhanden, und der Befehl DROP2 ist in diesem Fall ungeeignet. Beachten Sie immer den Zustand von LAST, wenn Sie Fehlerbehandlungen verwenden.

Ein weiteres Beispiel zu `IFERR ... THEN ... END` ist in "BDISP (Binäre Anzeige)" auf Seite 259 gegeben. Desweiteren können Sie eine ELSE Anweisung mitverwenden, die nur ausgewertet wird, falls kein Fehler auftritt. Dies hat folgende Form:

```
IFERR ... THEN ... ELSE ... END
```

Bestimmte Schleifenstrukturen

Schleifenstrukturen enthalten eine *Schleifenanweisung*, die wiederholt ausgewertet wird. Dabei spezifiziert das Programm im voraus, wie oft die Schleifenanweisung ausgewertet werden soll. Eine alternative Programmstruktur ist die *unbestimmte Schleifenstruktur*, bei der eine Test-Anweisung darüber entscheidet, ob die Auswertung einer Schleifenanweisung wiederholt werden soll. In diesem Abschnitt werden die bestimmten Schleifenstrukturen beschrieben; unbestimmte Schleifenstrukturen werden auf Seite 231 behandelt.

START... NEXT

Das folgende Beispiel gibt viermal ein Akustiksignal aus.

```
1 4 START 440 .1 BEEP NEXT
```

Diese Struktur läuft folgendermaßen ab:

1. Der Befehl START nimmt die Werte 1 und 4 vom Stack und erzeugt einen *Zähler*. Mit diesem Zähler wird dann die Anzahl der wiederholten Schleifendurchläufe festgehalten. Der Wert 1 gibt den *Startwert* und der Wert 4 den *Endwert* des Zählers an.

2. Die Schleifenanweisung `440 ,1 BEEP` wird ausgeführt.
3. Der Befehl NEXT erhöht den Zähler um 1.
4. Der momentane Zählerstand wird mit dem Endwert des Zählers verglichen.
   - Falls der momentane Zählerstand nicht größer als der Endwert des Zählers ist, werden die Schritte (2), (3) und (4) wiederholt.
   - Falls der momentane Zählerstand den Endwert des Zählers übertrifft, ist die bestimmte Schleifenstruktur beendet und die Programmausführung wird nach dem NEXT Befehl fortgesetzt.

In diesem Beispiel werden die Schritte (2), (3) und (4) viermal wiederholt. Der Schleifenzähler wird dabei zuerst von 1 auf 2, dann auf 3, dann auf 4 und schließlich auf 5 erhöht. An dieser Stelle ist der Zählerstand höher als 4. Die bestimmte Schleifenstruktur wird daher beendet. Beachten Sie, daß Schritt (1) vor jeglichem Test ausgeführt wird. Die Schleifenanweisung wird folglich immer zumindest einmal ausgewertet. Ein weiteres Beispiel zu START ... NEXT wird in "FIB2 (Fibonacci Reihe, Schleifenversion)" auf Seite 248 gegeben.

FOR *Zähler*... NEXT

In vielen Fällen ist es praktisch, den momentanen Wert des Zählers als Variable in der Schleifenanweisung zu verwenden. In diesem Fall ersetzen Sie START durch FOR *Name*. Der Zähler wird zu einer lokalen Variablen mit einem spezifizierten Namen. Wie schon zuvor angemerkt, sind in diesem Handbuch lokale Variablen zur Unterscheidung von globalen Variablen durch Kleinbuchstaben gekennzeichnet. Im folgenden Beispiel werden die ersten fünf Glieder einer quadratischen Reihe (von ganzen Zahlen) im Stack abgelegt.

```
1 5 FOR x x SQ NEXT
```

Die Sequenz `FOR x` wird nur einmal ausgeführt. Die Sequenz `x SQ` stellt die Schleifenanweisung dar, die wiederholt ausgeführt wird.

In den bisherigen Beispielen wurde als Startwert immer der Wert 1 spezifiziert. Im allgemeinen ist jedoch jede beliebige ganze Zahl zulässig. Im folgenden Beispiel werden die Quadrate der ganzen Zahlen von 3 bis 9 im Stack abgelegt.

```
3 9 FOR × × SQ NEXT
```

Ein weiteres Beispiel finden Sie unter "BDISP (Binäre Anzeige)" auf Seite 259.

... Schrittweite STEP

Der Befehl NEXT erhöht den Zähler stets um eins. Um eine andere *Schrittweite* zu spezifizieren, ersetzen Sie NEXT durch *n* STEP, wobei *n* die entsprechende Schrittweite darstellt. STEP wird wie in den vorherigen Beispielen gewöhnlich nach FOR *Zähler* benutzt, kann jedoch ebenso START folgen. Im folgenden Beispiel werden die Quadrate aller *ungeraden* ganzen Zahlen von 1^2 bis 5^2 im Stack abgelegt.

```
1 5 FOR × × SQ 2 STEP
```

Dabei wird die Schleifenanweisung `× SQ` 2 dreimal ausgeführt. Der Befehl `STEP` erhöht den Zähler zuerst von 1 auf 3, dann auf 5 und schließlich auf 7. An dieser Stelle ist der Zählerstand höher als der Endwert 5. Die bestimmte Schleifenstruktur wird daher beendet.

In den bisherigen Beispielen wurde immer ein *zunehmender* Zähler verwendet. Analog kann für *abnehmende* Werte des Zählers eine negative Schrittweite spezifiziert werden. Im folgenden Beispiel werden die Quadrate der ungeraden ganzen Zahlen von 5^2 bis 1^2 im Stack abgelegt.

```
5 1 FOR × × SQ -2 STEP
```

Die Sequenz `-2 STEP` vermindert den Zähler zuerst von 5 auf 3, dann auf 1 und schließlich auf −1. An dieser Stelle ist der Zählerstand niedriger als der Endwert 1. Die bestimmte Schleifenstruktur wird daher beendet.

Das Programm "SORT (Sortieren einer Liste)" auf Seite 270 verwendet `-1 STEP`, um den Zähler jeweils um eins zu vermindern. In diesem Fall ändert STEP wie NEXT den Wert des Zählers um eins, mit dem Unterschied, daß der Zähler vermindert wird, anstatt erhöht.

Unbestimmte Schleifenstrukturen

Anstatt die Anzahl der Schleifendurchläufe im voraus zu spezifizieren, können Sie ebenfalls eine *unbestimmte Schleifenstruktur* verwenden, die sowohl eine Schleifenanweisung wie auch eine Testanweisung enthält. Dabei werden die Anweisungen abwechselnd ausgeführt, wobei das Ergebnis der Testanweisung über die Fortsetzung entscheidet.

In diesem Abschnitt werden zwei Arten von unbestimmten Schleifenstrukturen behandelt. Bei der ersten, DO... UNTIL... END, wird die Schleifenanweisung vor der Testanweisung ausgeführt. Folglich wird die Schleifenanweisung stets mindestens einmal ausgeführt. Bei der zweiten Art, WHILE... REPEAT... END, wird dagegen zuerst die Testanweisung ausgeführt. Folglich wird in bestimmten Fällen die Schleifenanweisung nie ausgeführt.

DO...UNTIL...END

Im folgenden Beispiel wird ein Objekt wiederholt solange ausgewertet, bis das Objekt durch die Auswertung nicht mehr geändert wird. Da die Auswertung mindestens einmal stattfinden muß, bevor ein Test erfolgen kann, wird in diesem Fall die Schleifenstruktur DO... UNTIL... END verwendet.

```
DO DUP EVAL UNTIL DUP ROT SAME END
```

Der Ablauf dieser Struktur erfolgt folgendermaßen:

1. Die Schleifenanweisung `DUP EVAL` wird ausgeführt, wobei das Objekt und das Ergebnis im Stack abgelegt werden.
2. Die Testanweisung `DUP ROT SAME` wird ausgewertet, wobei das Ergebnis und ein Flag im Stack abgelgt werden. Das Flag zeigt an, ob das Objekt und das Auswertungsergebnis identisch sind.
3. Das Flag wird vom Stack genommen. Der Wert des Flags entscheidet, ob die Schleifenstruktur wiederholt wird.
   - Falls das Flag falsch ist, werden die Schritte (1), (2) und (3) wiederholt.
   - Falls das Flag wahr ist, wird die Schleifenstruktur beendet.

Angenommen, Sie möchten 'A+B' vollständig auswerten, wobei A 'P+Q' und P den Wert 2 enthält. Die erste Auswertung der Schleifenanweisung gibt 'A+B' und 'P+Q+B' zurück. Diese beiden Ausdrücke sind nicht identisch. Die Schleifenanweisung wird daher ein zweites Mal ausgewertet, wobei 'P+Q+B' und '2+Q+B' zurückgegeben werden. Die beiden Ausdrücke sind nicht identisch. Die Schleifenanweisung wird daher ein drittes Mal mit dem Ergebnis '2+Q+B' und '2+Q+B' ausgewertet. Die beiden Ausdrücke sind identisch, und die Schleifenstruktur wird folglich beendet.

Dieses Beispiel hat eine ähnliche Auswirkung wie →NUM, mit der Ausnahme, daß →NUM einen Fehler verursacht, wenn ein Name undefiniert ist. Eine vielseitigere Version dieses Beispiels finden Sie unter "MULTI (Mehrfache Ausführung)" auf Seite 253.

WHILE ... REPEAT ... END

Im folgenden Beispiel wird eine beliebige Anzahl von Vektoren vom Stack genommen und der Statistikmatrix hinzugefügt. Da *vor* dem Versuch, einen Vektor der Statistikmatrix hinzuzufügen, zuerst geprüft werden muß, ob das Objekt in Ebene 1 ein Vektor ist, wird hier die Struktur WHILE ... REPEAT ... END verwendet.

```
WHILE DUP TYPE 3 == REPEAT Σ+ END
```

Der Ablauf dieser Struktur lautet:

1. Die Testanweisung `DUP TYPE 3 ==` wird ausgewertet, wobei ein Flag im Stack abgelegt wird. Dieses Flag kennzeichnet, ob das Objekt in Ebene 2 ein reeller Vektor ist.
2. Das Flag wird vom Stack genommen. Der Wert des Flags bestimmt, ob die Schleifenanweisung ausgeführt wird.
   - Falls das Flag wahr ist, wird die Schleifenanweisung `Σ+` ausgeführt, d.h. der Vektor wird der momentanen Statistikmatrix hinzugefügt und die Schritte (1) und (2) werden ausgeführt.
   - Falls das Flag falsch ist, wird die Schleifenstruktur beendet.

Beachten Sie, daß WHILE ... REPEAT ... END beendet wird, wenn das Flag falsch ist, DO ... UNTIL ... END jedoch, wenn das Flag wahr ist. Wenn Sie den Wahrheitswert einer Testanweisung ändern möchten, fügen Sie als letzten Befehl NOT hinzu: WHILE ... NOT REPEAT oder UNTIL ... NOT END.

Ein weiteres Beispiel zu WHILE ... REPEAT ... END finden Sie unter "PAD (Auffüllen mit führenden Leerzeichen)" auf Seite 257.

Geschachtelte Programmstrukturen

Da eine Anweisung innerhalb einer Programmstruktur sich wie ein Unterprogramm verhält, kann die Anweisung selbst eine Programmstruktur beinhalten. Die Struktur innerhalb der Anweisung wird dann als *innere* Struktur und die Struktur, die die Anweisung enthält, als *äußere* Struktur bezeichnet. Das Programm "SORT (Sortieren einer Liste)" auf Seite 270 zeigt geschachtelte bestimmte Schleifen.

Außer Ihrer eigenen Übersicht gibt es keine Begrenzung der verschiedenen Ebenen von geschachtelten Strukturen. In einigen Fällen ist es einfacher, die innere Strukturen in Programmen abzuspeichern und deren Namen als Anweisungen in den äußeren Strukturen zu verwenden.

27

Interaktive Programme

In manchen Fällen benötigen Programme während der Programmausführung Angaben vom Benutzer. Beispielsweise kann es vorkommen, daß Werte eingegeben werden sollen oder zwischen mehreren Möglichkeiten der Programmfortsetzung gewählt werden soll.

In diesem Kapitel wird gezeigt, wie die folgenden Befehle aus dem PROGRAM CONTROL Menü den Benutzer zur Eingabe auffordern oder zum Treffen einer Wahl veranlassen können.

Befehl	**Bedeutung**
HALT	Unterbrechung der Programmausführung.
s WAIT	Unterbrechung des Programms für *s* Sekunden.
KEY	Rückgabe eines Tasten-Strings, wenn eine Taste gedrückt wurde.
f s BEEP	Ausgabe eines Akustiksignals für *s* Sekunden mit der Frequenz *f*.
CLLCD	Löschen der Anzeige.
n DISP	Anzeige eines Objekts in Zeile *n* der Anzeige.
CLMF	Wiederherstellung der normalen Anzeige nach Beendigung der Programmausführung.

Aufforderung zur Eingabe

Die folgende Sequenz erzeugt ein CUSTOM Eingabemenü für die Variablen A, B und C, gibt ein Akustiksignal aus und unterbricht das Programm zur Eingabeaufforderung.

```
... « STO A B C » MENU 440 ,1 BEEP HALT ...
```

Das angezeigte Menü enthält, ähnlich wie im Lösermenü, die Felder [A], [B] und [C]. Nach der Eingabe eines Werts in den Stack kann der Benutzer durch Drücken einer dieser Tasten den Wert in der entsprechenden Variablen speichern. Um das Programm nach der Eingabe der Variablenwerte fortzusetzen, ist ■[CONT] zu drücken.

Treffen einer Auswahl

Für komplexe Aufgaben ist es einfacher, eine Reihe von kleinen Programmen zu schreiben, wobei jedes Programm eine der Aufgaben bearbeitet. Angenommen, eine Aufgabe ist bearbeitet und der Benutzer muß zur Ausführung der nächsten Aufgabe zwischen den Programmen HOP, SKIP und JUMP wählen. Die folgende Sequenz erzeugt ein CUSTOM Benutzermenü für die Programme HOP, SKIP und JUMP, gibt ein Akustiksignal aus und beendet die Programmausführung.

```
...« HOP SKIP JUMP » MENU 440 ,1 BEEP »
```

Das angezeigte Menü enthält die Felder HOP, SKIP und JUMP. Wenn der Benutzer eine der Menütasten drückt, wird die nächste Aufgabe bearbeitet. Diese und die nachfolgenden Aufgaben können dann bis zum Ende der komplexen Aufgabe mit einer ähnlichen Sequenz enden, durch die der Benutzer jeweils eine weiteres Programm-Menü zur Auswahl hat.

Ein umfangreicheres Beispiel

Im folgenden Beispiel wird eine Meldung angezeigt, danach gewartet, bis eine Taste gedrückt wird und anschließend geprüft, ob die Taste definiert ist (d.h., eine gültige Auswahlmöglichkeit ist). Falls die Taste definiert ist, wird die entsprechende Aufgabe bearbeitet; falls die Taste nicht definiert ist, wird eine Fehlermeldung angezeigt und der Prozeß neu gestartet.

In diesem Beispiel gibt es eine "äußere" DO ... UNTIL ... END Struktur, die solange wiederholt wird, bis der Benutzer eine definierte Taste drückt. Die äußere DO Anweisung enthält eine "innere" DO ... UNTIL ... END Struktur, die solange wiederholt wird, bis der Benutzer eine Taste drückt. Die äußere UNTIL Anweisung enthält eine Bedingung, die eine Fehlermeldung anzeigt, falls die Taste nicht definiert ist.

In der nachfolgenden Liste markieren die Einrückungen die äußere Struktur, die Anweisungen, die innere Struktur sowie deren Anweisungen.

Sequenz	Bemerkung
`{ "Alfred" "Birgit" "Connie" }`	Diese Liste enthält die möglichen Ergebnisse. Sie bleibt im Stack, bis die folgende DO...UNTIL...END Struktur zur Kennzeichnung der Wahl des Benutzers 1, 2 oder 3 zurückgibt.
`DO`	Beginn der äußeren Schleifenanweisung. Diese Anweisung zeigt Auswahlmeldungen an, damit Sie die verschiedenen Möglichkeiten erkennen und eine Wahl treffen können.
`CLLCD`	Löschen der Anzeige.
`"Drücken von"`	Auswahlmeldung für Zeile 1.
`1 DISP`	Anzeige der Meldung.
`" [A] für Alfred"`	Auswahlmeldung für Zeile 2.
`2 DISP`	Anzeige der Meldung.
`" [B] für Birgit"`	Auswahlmeldung für Zeile 3.
`3 DISP`	Anzeige der Meldung.
`" [C] für Connie"`	Auswahlmeldung für Zeile 4.
`4 DISP`	Anzeige der Meldung.
`DO UNTIL KEY END`	Diese innere unbestimmte Schleife wird solange wiederholt, bis eine Taste gedrückt wird. Der Befehl KEY gibt dann `0` zurück, falls keine Taste gedrückt wurde oder einen String (für die Taste) und `1`, falls eine Taste gedrückt wurde.
`UNTIL`	Beginn der äußeren Testanweisung. Die Anweisung prüft, ob die gedrückte Taste definiert ist.

```
{ "A" "B" "C" }
```

Diese Liste enthält die definierten Tasten. Die Beziehung zwischen den definierten Tasten und den möglichen Ergebnissen ist eindeutig.

```
SWAP POS
```

Vergleichen des Tasten-Strings mit den in der Liste definierten Tasten. POS (*Position*) gibt 1 zurück, falls der Tasten-String "A", 2, falls der String "B", 3, falls der String "C" ist und 0, falls keine Übereinstimmung erzielt wird.

```
 IF DUP
```

Erstellen einer Kopie der Position, um sie als Flag zu benutzen. Falls die Position 1, 2 oder 3 ist, wird die THEN-Anweisung ausgeführt. Falls die Position 0 ist, wird die ELSE-Anweisung ausgeführt.

```
 THEN 1
```

Die Taste ist definiert, und deshalb wird ein Wahr-Flag im Stack abgelegt.

```
 ELSE
```

Die Taste wurde nicht definiert, und daher wird eine Fehlermeldung angezeigt und ein Akustiksignal ausgegeben.

```
  CLLCD "Falsche Taste"
  1 DISP
```

Anzeige einer Fehlermeldung.

```
  440 ,1 BEEP
```

Ausgabe des Akustiksignals.

```
  1 WAIT
```

Eine Sekunde lang warten.

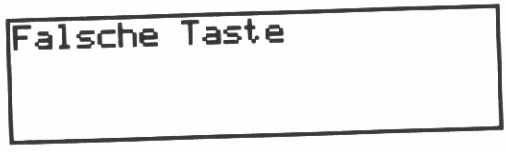
```
 END
```

Beenden der IF ... THEN ... ELSE ... END Struktur. Falls die Taste definiert ist, ist die Position und ein Wahr-Flag im Stack abgelegt. Falls die Taste nicht definiert war, ist nur die Position (die dann auch das Falsch-Flag darstellt) im Stack.

END	Beenden der äußeren unbestimmten Schleife. Falls die Taste definiert war, endet die Schleife mit der Position im Stack. Falls die Taste nicht definiert war, wird die Schleifenanweisung wiederholt.
GET	Abrufen des entsprechenden Ergebnisses, wenn die Liste der möglichen Ergebnisse und eine Position gegeben sind.
EVAL	Auswerten des Ergebnisses. In diesem Beispiel hat EVAL keinen Einfluß, da das Ergebnis ein String ist. In einem realistischeren Beispiel kann das Ergebnis z.B. ein Programm (möglicherweise in einer Variablen gespeichert) sein. In diesem Fall würde EVAL benötigt.
CLMF	Aktivieren der normalen Stackanzeige.

Wenn diese Sequenz ausgeführt wird, werden die Auswahlmeldungen angezeigt.

Falls eine von [A], [B] oder [C] verschiedene Taste gedrückt wird, ertönt ein Akustiksignal und die Fehlermeldung wird eine Sekunde lang angezeigt.



Dann werden wieder die Auswahlmeldungen angezeigt. Beim Drücken von [A], [B] oder [C] wird der String `"Alfred"`, `"Birgit"` bzw. `"Connie"` in die Ebene 1 zurückgegeben.

Durch Ändern der Liste der möglichen Ergebnisse, der Auswahlmeldungen und der Liste der definierten Tasten können Sie diese Sequenz effektiver als das Speichern eines Strings im Stack machen. Noch allgemeiner kann folgendes Programm erstellt werden, indem diese Sequenz in eine Struktur von lokalen Variablen eingebaut wird.

```
« → Tasten m1 m2 m3 m4
 « DO CLLCD
    m1 1 DISP
    m2 2 DISP
    m3 3 DISP
    m4 4 DISP
    DO UNTIL KEY END
   UNTIL Tasten SWAP POS
    IF DUP
    THEN 1
    ELSE CLLCD "Falsche Taste" 1 DISP
     440 ,1 BEEP 1 WAIT
    END
   END
 »
 GET EVAL CLMF
»
```

Wenn Sie dieses Programm in einer Variablen mit dem Namen TASTE? speichern, können Sie obiges Beispiel folgendermaßen ausführen:

```
{ "Alfred" "Birgit" "Connie" }
{ "A" "B" "C" }
"Drücken von"
" [A] für Alfred"
" [B] für Birgit"
" [C] für Connie"
TASTE?
```

28

Programmbeispiele

Dieses Kapitel enthält 20 Programme für Ihren HP-28S. Diese Programme sind nützlich und, weit wichtiger, zeigen eine Vielfalt von Programmiermethoden. Für jedes Programm finden Sie folgende Informationen vor:

- Stack-Diagramm. Ein Stack-Diagramm ist eine zweispaltige Tabelle für "Argumente" und "Ergebnisse". "Argumente" zeigt, was sich im Stack vor der Programmausführung befinden muß; "Ergebnisse" zeigt, was nach der Programmausführung im Stack abgelegt ist.

  Das Stack-Diagramm läßt nicht alle Vorgänge erkennen; z.B. könnte ein Programm, das den Inhalt des Benutzerspeichers ändert oder Objekte anzeigt, auf den Stack keinen Einfluß haben.
- Programmiermethode. Dies ist der interessanteste Teil. Wenn Sie verstehen, wie eine bestimmte Methode in diesem Kapitel benutzt wird, können Sie sie später in Ihren eigenen Programmen verwenden.
- Benötigte Programme. Manche Programme rufen andere Programme als Unterprogramme auf. Sie können die benötigten Programme und das aufrufende Programm in beliebiger Reihenfolge eingeben, müssen jedoch alle Programme vor der Ausführung des aufrufenden Programms eingegeben haben.
- Programm und Bemerkung. In diesem Kapitel wird für die Programmlisten ein bestimmtes Format verwendet, um die Struktur des Programms und des Prozesses zu verdeutlichen. Sie müssen bei der Eingabe eines Programms jedoch nicht unbedingt diesem Format folgen. Geben Sie allerdings Leerzeichen an den Stellen ein, wo sie in der Liste erscheinen oder zwischen Objekten, die in getrennten Zeilen enthalten sind.

  Sie können ein Programm zeichenweise oder mit Hilfe von Menüs befehlsweise eingeben. Beide Eingabemethoden sind gleichwertig, solange die beiden erzeugten Programmlisten übereinstimmen.

Wenn Sie ein Programm eintippen, können Sie alle schließenden Klammern sowie Begrenzungszeichen *ganz am Ende des Programms* weglassen—diese werden beim Drücken von ENTER automatisch hinzugefügt.

- Beispiel: Die folgenden Beispiele gehen vom STD Anzeigeformat aus. Das STD Format können Sie über das MODE Menü oder durch Drücken von STD ENTER wählen.

Die wichtigste in diesem Kapitel gezeigte Programmiermethode ist die *strukturierte Programmierung*: Kleine Programme werden zum Aufbau anderer Programme verwendet. Folgende Programme werden in anderen Programmen benutzt.

- BOXO wird in BOXV verwendet.
- MULTI wird in EXCO verwendet.
- PAD und PRESERVE werden in BDISP verwendet.
- ΣGET wird in ΣX2, ΣY2 und ΣXY verwendet.
- SORT und LMED werden in MEDIAN verwendet.

Kubische Funktionen (Box)

Dieser Abschnitt enthält zwei Programme:

- BOXO berechnet die gesamte Oberfläche eines Quaders.
- BOXV verwendet BOXO zur Berechnung des Verhältnisses von Oberfläche und Volumen für einen Quader.

BOXO (Oberfläche eines Quaders)

Gegeben ist die Länge, Breite und Höhe eines Quaders. Berechnen Sie die gesamte Oberfläche der sechs Seitenflächen.

Argumente	Ergebnisse
3: *Länge*	3:
2: *Breite*	2:
1: *Höhe*	1: *Fläche*

Programmiermethode:

- Struktur von lokalen Variablen. Lokale Variablen ermöglichen Ihnen, den Argumenten Namen zuzuordnen, die nicht im Widerspruch zu globalen Variablen stehen. Ähnlich wie bei globalen Variablen ist die Verwendung lokaler Variablen hilfreich, da dadurch Argumente beliebig oft und ohne aufwendige Stackverwaltung verwendet werden können. Jedoch gehen die lokalen Variablen nach Beenden der Programmstruktur verloren, von welcher sie erzeugt wurden.

  Eine Struktur von lokalen Variablen besteht aus drei Teilen.

  1. Einem als "→ " bezeichneten Befehl. Beachten Sie bei der Eingabe dieses Befehls, vor und nach ihm jeweils ein Leerzeichen einzufügen. (Wie jeder andere Befehl kann → nur erkannt werden, wenn dieser Befehl durch Leerzeichen abgetrennt ist. Verwechseln Sie jedoch diesen Ein-Zeichen-Befehl nicht mit Begrenzungszeichen wie # oder «.)
  2. Einem oder mehreren Namen.
  3. Einer Prozedur (Ausdruck, Gleichung oder Programm), die die Namen enthält. Diese Prozedur wird als *definierende* Prozedur bezeichnet.

  Während der Auswertung einer Struktur mit lokalen Variablen wird für jeden Namen eine lokale Variable erzeugt und die entsprechende Werte aus dem Stack entnommen. Die definierende Prozedur wird dann ausgewertet und ersetzt die Werte der lokalen Variablen.

  Um die Vorteile der lokalen Variablen besser zu schätzen, vergleichen Sie die nachfolgend aufgeführte Version von BOXO mit der Version auf Seite 244.

- Benutzerfunktion. Dieser Programmtyp funktioniert sowohl mit UPN als auch mit algebraischer Syntax. Eine Benutzerfunktion ist ein Programm, das ausschließlich aus einer Struktur von lokalen Variablen besteht und exakt ein Ergebnis zurückgibt.

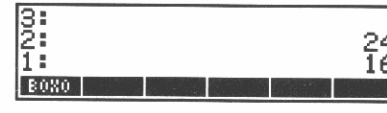
Programm	Bemerkung
«	Beginnen des Programms.
→ l b h	Erzeugen von lokalen Variablen für Länge, Breite und Höhe. Entsprechend der Konvention werden Kleinbuchstaben verwendet. Die Werte werden entweder vom Stack (in UPN) oder von den Argumenten der Benutzerfunktion (in algebraischer Syntax) entnommen.

Programm	Bemerkung
`'2*(b*h+l*h+l*b)'`	Der definierende Ausdruck für die Oberfläche. Wenn die Benutzerfunktion ausgeführt wird, wird dieser Ausdruck ausgewertet und die Oberfläche im Stack abgelegt.
»	Beenden des Programms.
ENTER	Speichern des Programms im Stack.
' BOXO STO	Speichern des Programms unter BOXO.

Beispiel: Einer der Vorteile von Benutzerfunktionen ist, daß sie sowohl in UPN als auch in algebraischer Syntax funktionieren. Berechnen Sie die Oberfläche eines Quaders, der 24 cm lang, 16 cm breit und 12 cm hoch ist. Verwenden Sie zuerst UPN und anschließend algebraische Syntax.

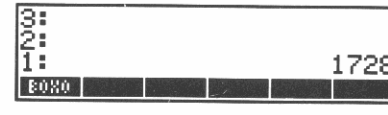
Geben Sie für die UPN Version zuerst die Länge und Breite ein.

USER
24 ENTER
16 ENTER



Geben Sie dann die Höhe ein und führen Sie BOXO aus.

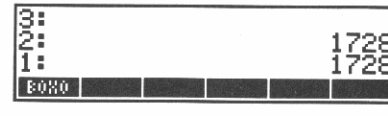
12 BOXO



Die Oberfläche beträgt 1728 Quadratzentimeter.

Verwenden Sie nun die algebraische Version.

' BOXO (24,16,12 EVAL



Die Oberfläche beträgt auch hier 1728 Quadratzentimeter.

BOXO ohne lokale Variablen

Das folgende Programm verwendet zur Berechnung der Oberfläche eines Quaders nur Stack-Operationen. Vergleichen Sie dieses Programm mit BOXO.

Argumente	Ergebnisse
3: *Länge*	3:
2: *Breite*	2:
1: *Höhe*	1: *Fläche*

Programm	Bemerkung
«	Beginnen des Programms.
DUP2 *	Berechnen von bh.
ROT	Verschieben von b in Ebene 1.
4 PICK	Kopieren von l in Ebene 1.
*	Berechnen von bl.
+	Berechnen von $bh + bl$.
ROT ROT	Verschieben von l und h in Ebene 2 und 1.
*	Berechnen von lh.
+	Berechnen von $bh + bl + lh$.
2 *	Berechnen von $2(bh + bl + lh)$.
»	Beenden des Programms.

Da diese Version von BOXO keine Benutzerfunktion ist, kann sie nicht in algebraischer Syntax benutzt werden.

BOXV (Verhältnis von Oberfläche und Volumen eines Quaders)

Gegeben ist die Länge, Breite und Höhe eines Quaders. Berechnen Sie das Verhältnis von Oberfläche und Volumen des Quaders.

Argumente	Ergebnisse
3: *Länge*	3:
2: *Breite*	2:
1: *Höhe*	1: *Fläche/Volumen*

Programmiermethode:

- Geschachtelte Benutzerfunktionen. BOXV ist eine Benutzerfunktion, deren definierender Ausdruck BOXO bei Berechnungen verwendet. Desweiteren könnte BOXV zur Definition anderer Benutzerfunktion verwendet werden.

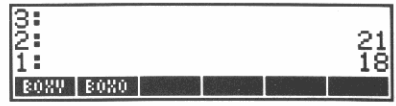
  Zur Erinnerung: BOXO wurde mit den lokalen Variablen *l*, *b* und *h* definiert. Beachten Sie, daß nun in der Definition für BOXV das Programm BOXO die Argumente *x*, *y* und *z* verwendet. Da jeder Satz von lokalen Variablen voneinander unabhängig ist, spielt es keine Rolle, ob die lokalen Variablen in den beiden Definitionen übereinstimmen oder nicht. Es ist jedoch unbedingt erforderlich, daß lokale Variablen innerhalb einer Definition einheitlich sind.

Programm	Bemerkung
`«`	Beginnen des Programms.
`→ x y z`	Erzeugen von lokalen Variablen für Länge, Breite und Höhe. Dieses Programm verwendet *x*, *y* und *z* an Stelle von *l*,*b* und *h*.
`'BOXO(x,y,z)`	Beginnen des definierenden Ausdrucks mit der Benutzerfunktion BOXO.
`/(x*y*z)'`	Dividieren durch das Volumen des Quaders.
`»`	Beenden des Programms.
ENTER	Eingeben des Programms in den Stack.
' BOXV STO	Speichern des Programms unter BOXV.

Beispiel: Berechnen Sie das Verhältnis von Oberfläche und Volumen eines 21 cm langen, 18 cm breiten und 9 cm hohen Quaders; führen Sie die Berechnung zuerst in UPN und dann in algebraischer Syntax aus.

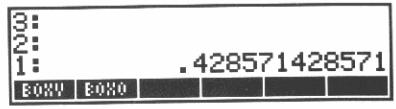
Geben Sie für die UPN Version zuerst die Länge und Breite ein.

USER
21 ENTER
18 ENTER



Geben Sie dann die Höhe ein und führen Sie BOXV aus.

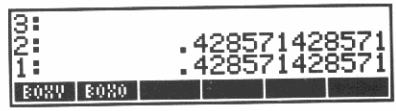
9 BOXV



Das Verhältnis beträgt 0.428571428571.

Verwenden Sie nun die algebraische Version.

' BOXV (21,18,9 EVAL



Das Verhältnis beträgt auch hier 0.428571428571.

Fibonacci-Reihen

Gegeben ist eine ganze Zahl n. Berechnen Sie das n-te Glied der Fibonacci-Reihe F_n, wobei

$$F_0 = 0, \quad F_1 = 1, \quad F_n = F_{n-1} + F_{n-2}$$

Dieser Abschnitt enthält zur Lösung dieser Aufgabe zwei verschiedene Programme.

- FIB1 ist eine Benutzerfunktion die *rekursiv* definiert ist—der definierende Ausdruck enthält seinen eigenen Namen. FIB1 ist kurz, leicht zu verstehen und in algebraischen Objekten verwendbar.
- FIB2 ist ein Programm mit einer bestimmten Schleife. Es kann nicht in algebraischen Objekten verwendet werden, ist länger und komplizierter als FIB1, ist jedoch schneller als FIB1.

FIB1 (Fibonacci-Reihe, rekursive Version)

Argumente	Ergebnisse
1: n	1: F_n

Programmiermethode:

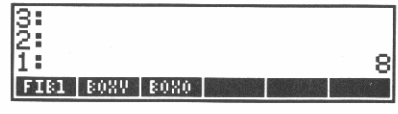
- IFTE (If-Then-Else Funktion). Der FIB1 definierende Ausdruck enthält die bedingte Funktion IFTE, die sowohl in UPN als auch in algebraischer Syntax verwendet werden kann. (FIB2 verwendet die Programmstruktur IF ... THEN ... ELSE ... END.)
- Rekursion. Der FIB1 definierende Ausdruck ist, genau wie F_n durch F_{n-1} und F_{n-2} definiert ist, durch FIB1 ausgedrückt.

Programm	Bemerkung
«	Beginnen des Programms.
→ n	Definieren einer lokalen Variable.
'	Anfang des definierenden Ausdrucks.
IFTE(n≤1,	Falls $n \leq 1$,
n,	dann $F_n = n$;
FIB1(n-1)+FIB1(n-2))	sonst $F_n = F_{n-1} + F_{n-2}$.
'	Ende des definierenden Ausdrucks.
»	Beenden des Programms.
ENTER	Speichern des Programms im Stack.
' FIB1 STO	Speichern des Programms in FIB1.

Beispiel: Berechnen Sie F_6 unter Verwendung der UPN Syntax, und F_{10} unter Verwendung der algebraischen Syntax.

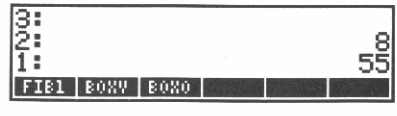
Berechnen Sie mit UPN zuerst F_6.

USER
6 FIB1



Berechnen Sie anschließend F_{10} über algebraische Syntax.

' FIB1 (10 EVAL



FIB2 (Fibonacci-Reihe, Schleifenversion)

Argumente	Ergebnisse
1: n	1: F_n

Programmiermethode:

- IF ... THEN ... ELSE ... END. FIB2 verwendet für die Bedingung die Programmstruktur-Form. (FIB1 verwendet IFTE.)
- START ... NEXT (bestimmte Schleife). Zur Berechnung von F_n beginnt FIB2 mit F_0 und F_1 und berechnet F_i in einer Schleife.

Programm	Bemerkung
«	Beginnen des Programms.
→ n	Erzeugen einer lokalen Variablen.
«	Beginnen des definierenden Programms.
IF n 1 ≤	Falls n ≤ 1,
THEN n	dann $F_n = n$;
ELSE	Beginnen der ELSE-Anweisung.
0 1	Ablegen von F_0 und F_1 im Stack.
2 n	Von 2 bis n,
START	Ausführen folgender Schleife:
DUP	Anlegen einer Kopie vom letzten F (beginnt mit F_1).
ROT	Verschieben von vorherigem F (beginnt mit F_0) in Ebene 1.
+	Nächstes F berechnen (beginnt mit F_2).
NEXT	Wiederholen der Schleife.
SWAP DROP	F_{n-1} nach unten verschieben.
END	Beenden der ELSE-Anweisung.
»	Beenden des definierenden Programms.
»	Beenden des Programms.
ENTER	Eingeben des Programms in den Stack.
' FIB2 STO	Speichern des Programms unter FIB2.

Beispiel: Berechnen Sie F_6 und F_{10}. Beachten Sie, daß FIB2 schneller als FIB1 abläuft.

Berechnen Sie F_6.

USER
6 FIB2

Berechnen Sie F_{10}.

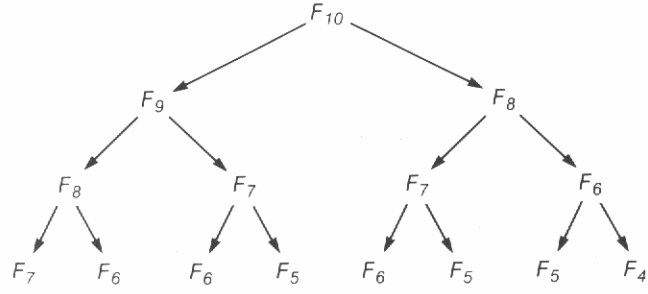
10 FIB2

Vergleich von FIB1 und FIB2

FIB1 berechnet die Zwischenwerte F_i mehr als einmal, während FIB2 jeden Zwischenwert F_i nur einmal berechnet. FIB2 ist folglich schneller.

Der Unterschied in der Ausführungsgeschwindigkeit erhöht sich mit zunehmendem n, da die Ausführungsgeschwindigkeit für FIB1 mit n exponentiell, für FIB2 dagegen nur linear ansteigt.

Das unten abgebildete Diagramm zeigt die ersten Schritte von FIB1 zur Berechnung von F_{10}. Beachten Sie die Anzahl der Zwischenwerte: 1 in der ersten Zeile, 2 in der zweiten Zeile, 4 in der dritten Zeile und 8 in der vierten Zeile.



Einzelschritt-Ausführung

Sie können die Funktionsweise eines Programms leichter verstehen, wenn Sie es schrittweise ausführen und dabei dessen Einfluß auf den Stack beobachten. Sie können dabei Ihre eigenen Programme besser "debuggen" (d.h. Fehler erkennen und beseitigen) oder Programme, die andere geschrieben haben, leichter verstehen.

In diesem Abschnitt wird gezeigt, wie Sie FIB2 schrittweise ausführen können. Sie können diese Vorgehensweise jedoch auf jedes beliebige Programm anwenden. Die allgemeine Vorgehensweise lautet:

1. Verwenden Sie VISIT, um den Befehl HALT in das Programm an der Stelle einzufügen, wo Sie mit der Einzelschritt-Ausführung beginnen möchten. (Sie werden sehen, wie durch die Position von HALT in FIB2 die Programmausführung beeinflußt wird.)
2. Führen Sie das Programm aus. Nachdem der HALT Befehl ausgeführt wurde, hält das Programm an (was durch das Symbol des STOP-Zeichens angezeigt wird).
3. Wählen Sie das PROGRAM CONTROL Menü.
4. Drücken Sie SST einmal, damit der nächste Programmschritt angezeigt und dann ausgeführt wird.

   Sie können nun:

   - Wiederholt SST drücken, um nacheinanderfolgende Schritte anzuzeigen und auszuführen.
   - ■ CONT drücken, um die Programmausführung fortzusetzen.
   - KILL drücken, um das Programm abzubrechen.
5. Wenn Sie das Programm wieder ohne Unterbrechungen ablaufen lassen möchten, entfernen Sie mit Hilfe von VISIT den Befehl HALT aus dem Programm.

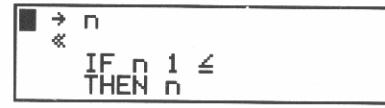
Geben Sie als erstes Beispiel den Befehl HALT am Anfang von FIB2 ein.

Löschen Sie den Stack und wählen Sie das USER Menü.

[■] [CLEAR]
[USER]

Verwenden Sie VISIT, um FIB2 in die Befehlszeile zurückzugeben.

['] FIB2 [■] [VISIT]



Geben Sie den Befehl HALT ein.

[▶] [INS] [■] [CONTRL] HALT

Speichern Sie die geänderte Version von FIB2.

[ENTER]

Berechnen Sie F_1. Zuerst geschieht nichts, außer, daß der ○ Indikator angezeigt wird.

[USER]
1 FIB2

Wählen Sie das PROGRAM CONTROL Menü und führen Sie SST (*Single-STep*) aus. Beobachten Sie die oberste Zeile der Anzeige, um den ersten Schritt vor seiner Ausführung angezeigt zu erhalten.

[■] [CONTRL]
SST

Beachten Sie, daß → n einen Schritt darstellt; "Schritt" ist nicht das nächste Objekt des Programms, sondern eine logische Einheit.

Vergleichen Sie die zu Beginn dieses Abschnitts gegebene allgemeine Vorgehensweise. Sie haben die ersten vier Schritte ausgeführt und können nun für den fünften eine der drei Möglichkeiten wählen.

Drücken Sie für dieses Beispiel wiederholt SST, bis der ○ Indikator erlischt. Dies zeigt an, daß FIB2 beendet ist.

Zur Berechnung von F_1 wird nur die THEN Anweisung in FIB2 ausgeführt. Führen Sie als zweites Beispiel `3 FIB2` aus und gehen Sie schrittweise durch die Berechnung von F_3. Hier wird die ELSE Anweisung einschließlich der START ... NEXT Schleife ausgeführt. Für $n = 3$ wird die START ... NEXT Schleife zweimal durchlaufen.

Nehmen Sie als drittes Beispiel an, daß Sie im Einzelschritt-Modus die START ... NEXT Schleife in einem Schritt ausführen möchten—Sie möchten den Stackinhalt vor jeder Schleifeniteration ansehen, jedoch nicht alle Schritte in FIB2 bzw. in der Schleife im Einzelschritt-Modus ausführen. Verschieben Sie dafür HALT in die Schleife. FIB2 hält nun erst an, wenn die Schleife erreicht wird. Sie können dann mit ■ CONT (*CONTinue*) die Schleife jedesmal einmal ausführen.

Verwenden Sie VISIT, um FIB2 in die Befehlszeile zurückzugeben.

USER
' FIB2 ■ VISIT

```
■ HALT → n
  «
    IF n 1 ≤
    THEN n
```

Verwenden Sie die Cursor-Tasten, um HALT zu löschen. Fügen Sie nun HALT (nach dem START Befehl) folgendermaßen ein:

```
    IF n 1 ≤
    THEN n
    ELSE 0 1 2 n
      START HALT◆DUP R...
```

Speichern Sie die geänderte Version von FIB2.

ENTER

```
3:
2:
1:
FIB2 FIB1 BOXV BOXO
```

Starten Sie die Berechnung von F_3. FIB2 hält dann vor der Ausführung der Schleife an.

3 FIB2

```
3:
2:                    0
1:                    1
FIB2 FIB1 BOXV BOXO
```

Setzen Sie die Ausführung der Schleife fort. FIB2 hält vor der zweiten Ausführung der Schleife an.

CONT

Setzen Sie die Ausführung der Schleife fort. Da dies der letzte Schleifendurchlauf ist, wird FIB2 bis zum Programmende ausgeführt.

CONT

Vergessen Sie nicht, den HALT Befehl zum Schluß mit VISIT zu entfernen.

Vollständiges Erweitern und Zusammenfassen

Dieser Abschnitt enthält zwei Programme:

- MULTI wiederholt die Ausführung eines Programms solange, bis die Ausführung keine Änderung mehr verursacht.
- EXCO verwendet MULTI zum vollständigen Erweitern und Zusammenfassen.

MULTI (Mehrfache Ausführung)

Führen Sie ein Programm, das auf ein gegebenes Objekt wirkt, solange aus, bis das Objekt durch das Programm nicht mehr geändert wird.

Argumente	Ergebnisse
2: *Objekt* 1: « *Programm* »	2: 1: *resultierendes Objekt*

Programmiermethode:

- DO ... UNTIL ... END (unbestimmte Schleife). Die DO Anweisung enthält die zu wiederholenden Schritte; die UNTIL Anweisung enthält den Test, der entscheidet, ob beide Anweisungen wiederholt werden (falls falsch) oder die Schleife verlassen wird (falls wahr).
- Programme als Argumente. Obwohl Programme in der Regel benannt und dann durch Aufrufen des Namens ausgeführt werden, können sie auch im Stack abgelegt und dann als Argumente für andere Programme verwendet werden.
- Auswertung der lokalen Variablen. Das Programmargument, das wiederholt ausgeführt werden soll, ist in einer lokalen Variablen gespeichert. Es ist hilfreich, Objekte in lokalen Variablen zu speichern, wenn Sie nicht im voraus wissen, wieviele Kopien benötigt werden.

  MULTI läßt einen der Unterschiede zwischen globalen und lokalen Variablen erkennen: Wenn eine globale Variable einen Namen oder ein Programm enthält, so wird bei der Auswertung des Namens der Inhalt der Variablen ausgewertet. Der Inhalt einer lokalen Variablen wird dagegen immer nur zurückgerufen. Folglich verwendet MULTI den lokalen Namen, um das Programmargument im Stack abzulegen, und führt dann zur Auswertung des Programms einen expliziten EVAL Befehl aus.

Programm	**Bemerkung**
`«`	Beginnen des Programms.
`→ p`	Erzeugen der lokalen Variablen *p*, die das Programmargument enthält.
`«`	Beginnen des definierenden Programms.
`DO`	Beginnen der DO Anweisung.
`DUP`	Anlegen einer Kopie des Objekts.
`p EVAL`	Anwenden des Programms auf das Objekt, Zurückgeben einer neuen Version. (Der EVAL Befehl ist zur Ausführung des Programms notwendig, da lokale Variable ihren Inhalt immer unausgewertet in den Stack zurückgeben.)
`UNTIL`	Beginnen der UNTIL Anweisung.
`DUP`	Anlegen einer Kopie der neuen Version des Objekts.
`ROT`	Verschieben der alten Version in Ebene 1.
`SAME`	Prüfen, ob alte und neue Version identisch sind.

Programm	Bemerkung
END	Beenden der UNTIL Anweisung.
»	Beenden des definierenden Programms.
»	Beenden des Programms.
ENTER	Eingeben des Programms in den Stack.
' MULTI STO	Speichern des Programms unter MULTI.

Beispiel. Die Funktion von MULTI wird im nächsten Programm gezeigt.

EXCO (Vollständiges Erweitern und Zusammenfassen)

Führen Sie für ein gegebenes Objekt EXPAN wiederholt durch, bis sich der algebraische Ausdruck nicht mehr ändert. Führen Sie dann COLCT solange durch, bis sich der algebraische Ausdruck nicht mehr ändert. In einigen Fällen ist das Ergebnis eine Zahl.

Argumente	Ergebnisse
1: *'algebraischer Ausdruck'*	1: *'algebraischer Ausdruck'*
1: *'algebraischer Ausdruck'*	1: *z*

Programmiermethode:

- Strukturierte Programmierung. EXCO ruft das Programm MULTI zweimal auf. Selbst wenn MULTI sonst nicht mehr aufgerufen wird, ist das Programm mit MULTI als separatem Programm und Programmaufruf trotzdem effizienter als wenn die Befehle für MULTI zweimal geschrieben würden.

Benötigte Programme:

- MULTI (auf Seite 253) führt wiederholt die Programme aus, die EXCO als Argumente bereitstellt.

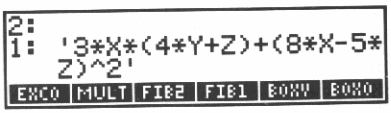
Programm	Bemerkung
«	Beginnen des Programms.
« EXPAN »	Speichern von EXPAN im Stack.
MULTI	Ausführen von EXPAN, bis sich das algebraische Objekt nicht mehr ändert.
« COLCT »	Speichern von COLCT im Stack.
MULTI	Ausführen von COLCT, bis sich das algebraische Objekt nicht mehr ändert.
»	Beenden des Programms.
ENTER	Eingeben des Programms in den Stack.
' EXCO STO	Speichern des Programms unter EXCO.

Beispiel: Vollständiges Erweitern und Zusammenfassen des Ausdrucks:

$$3x\ (4y + z) + (8x - 5z)^2$$

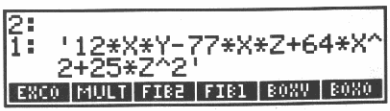
Geben Sie den Ausdruck ein.

USER
' 3 × X ×
(4 × Y + Z ▪) +
(8 × X − 5 × Z ▪) ▪ ^ 2
ENTER



Erweitern Sie und fassen Sie vollständig zusammen.

EXCO



Zur vollständigen Erweiterung von Ausdrücken mit vielen Produkten von Summen oder mit Potenzen können viele Iterationen von EXPAN notwendig sein, was zu einer langen Ausführungszeit für EXCO führen kann.

Anzeigen von Binärwerten

Dieser Abschnitt enthält drei Programme:

- PAD ist ein Dienstprogramm, das ein Objekt in einen String konvertiert, um es rechtsbündig anzuzeigen.
- PRESERVE ist ein Dienstprogramm zur Verwendung in Programmen, die den Rechnerstatus ändern (Winkelmodus, binäres Zahlensystem, usw.).
- BDISP zeigt einen Binärwert im Zahlensystem HEX, DEC, OCT oder BIN an. Es ruft PAD zur rechtsbündigen Anzeige auf, und ruft PRESERVE auf, um die binäre Darstellung zu erhalten.

PAD (Auffüllen mit führenden Leerzeichen)

Konvertieren Sie ein Objekt in einen String und fügen Sie am Anfang Leerzeichen hinzu, falls der String weniger als 23 Zeichen enthält.

Wenn ein kurzer String mit DISP angezeigt wird, so erscheint der String *linksbündig*. Die Position des letzten Zeichens ist durch die Länge des Strings bestimmt.

PAD verschiebt die Position des letzten Zeichens nach rechts, indem es zu Beginn des Strings Leerzeichen hinzufügt. Wenn der String 23 Zeichen lang ist, erscheint er in der Anzeige *rechtsbündig*: das letzte Zeichen erscheint am rechten Rand der Anzeige.

PAD hat keinen Einfluß auf Strings, die länger als 22 Zeichen sind.

Argumente	Ergebnisse
1: *Objekt*	1: " *Objekt*"

Programmiermethode:

- WHILE ... REPEAT ... END (unbestimmte Schleife). Die WHILE Anweisung enthält einen Test, der darüber entscheidet, ob die REPEAT Anweisung und der Test noch einmal ausgeführt (falls wahr) oder die REPEAT Anweisung übersprungen und die Schleife verlassen werden soll (falls falsch).

- String-Operationen. PAD demonstriert das Konvertieren eines Objekts in die String-Form, das Bestimmen der Anzahl der Zeichen und das Zusammenfügen zweier Strings.

Programm	Bemerkung
«	Beginnen des Programms.
→STR	Sicherstellen, daß das Objekt in String-Form vorliegt. (Strings werden durch diesen Befehl nicht beeinflußt.)
WHILE	Beginnen der WHILE Anweisung.
DUP SIZE 23 <	Enthält der String weniger als 23 Zeichen?
REPEAT	Beginnen der REPEAT Anweisung.
" " SWAP +	Hinzufügen eines vorangestellten Leerzeichens.
END	Beenden der REPEAT Anweisung.
»	Beenden des Programms.
ENTER	Eingabe des Programms in den Stack.
' PAD STO	Speichern des Programms unter PAD.

Beispiel: Die Funktionsweise von PAD wird im Programm BDISP gezeigt.

PRESERVE (Sichern und Wiederherstellen des vorherigen Status)

Sichern Sie, ausgehend von einem im Stack abgelegten Programm, den momentanen Status, führen Sie das Programm aus und stellen Sie den vorherigen Status wieder her.

Argumente	Ergebnisse
1: « *Programm* »	1: *(Ergebnis des Programms)*

Programmiermethode:

- RCLF und STOF. PRESERVE verwendet RCLF (*ReCaLl Flags*), um den momentanen Rechnerstatus in einem Binärwert zu speichern, und STOF (*STOre Flags*), um den Status über diesen Binärwert wieder herzustellen.

- Struktur einer lokalen Variablen. PRESERVE erzeugt kurzzeitig eine lokale Variable, die nur zum Entfernen des Objekts aus dem Stack dient. Sein definierendes Programm wertet nur das im Stack abgelegte Programmargument aus.

Programm	Bemerkung
«	Beginnen des Programms.
RCLF	Zurückrufen eines 64-Bit Binärwerts, der dem Status aller 64 Benutzerflags entspricht.
→ f	Speichern des Binärwerts in der lokalen Variablen *f*.
«	Beginnen des definierenden Programms.
EVAL	Auswerten des Programmarguments.
f STOF	Wiederherstellen des Status von allen 64 Benutzerflags.
»	Beenden des definierenden Programms.
»	Beenden des Programms.
ENTER	Eingeben des Programms in den Stack.
' PRESERVE STO	Speichern des Programms unter PRESERVE.

Beispiel: Die Funktionsweise von PRESERVE wird im Programm BDISP erläutert.

BDISP (Binäre Anzeige)

Zeigen Sie eine Zahl mit HEX, DEC, OCT und BIN als Zahlensystem an.

Argumente	Ergebnisse
1: # *n*	1: # *n*
1: *n*	1: *n*

Programmiermethode:

- IFERR ... THEN ... END (Fehlerbehandlung). Zur Behandlung von reellen Zahlen enthält BDISP den Befehl R→B (*Reell-in-Binär*). Dieser Befehl verursacht jedoch einen Fehler, wenn das Argument *bereits* ein Binärwert ist.

Um die Programmausführung beim Auftritt eines Fehlers fortzusetzen, ist der R→B Befehl in eine IFERR Anweisung eingebaut. Da beim Auftreten eines Fehlers keine Ausführung nötig ist, enthält die THEN Anweisung keinen Befehl.

- Aktivieren von LAST. Falls ein Fehler auftritt, muß LAST aktiviert sein, um das Argument in den Stack zurückgeben zu können. BDISP setzt Flag 31, um über das Programm die Rücksicherungsmöglichkeit mit Hilfe von LAST zu aktivieren.
- FOR ... NEXT Schleife (bestimmte Schleife mit Zähler). BDISP führt eine Schleife von 1 bis 4 aus und zeigt dabei jedesmal *n* in einem anderen Zahlensystem und in einer anderen Zeile an.

  Der Schleifenzähler (in disem Programm mit *j* bezeichnet) ist eine lokale Variable. Sie wird durch die FOR ... NEXT Programmstruktur (und nicht durch einen → Befehl) erzeugt und wird durch NEXT automatisch erhöht.
- Unterprogramme. BDISP zeigt drei verschiedene Arten der Verwendung von Unterprogrammen.
  1. BDISP enthält ein Steuerungs-Unterprogramm und einen Aufruf von PRESERVE. Das Steuerungs-Unterprogramm wird in den Stack übernommen und durch PRESERVE ausgewertet.
  2. Erzeugt BDISP für *n* eine lokale Variable, so ist das definierende Programm ein Unterprogramm.
  3. Es gibt vier Unterprogramme, die die Wirkung der Schleife "anpassen". Jedes Unterprogramm enthält einen Befehl zum Ändern des Zahlensystems, und in jedem Schleifendurchlauf wird eines der Unterprogramme ausgeführt.

Benötigte Programme:

- PAD (auf Seite 257) erweitert einen String auf 23 Zeichen, damit DISP den String rechtsbündig anzeigt.
- PRESERVE (auf Seite 258) speichert den momentanen Status, führt das Steuerungs-Unterprogramm aus und stellt den alten Status wieder her.

Programm	Bemerkung
«	Beginnen des Programms.
«	Beginnen des Steuerungs-Unterprogramms.
DUP	Anlegen einer Kopie von *n*.
31 SF	Setzen von Flag 31 zum Aktivieren von LAST.
IFERR	Beginnen der Fehlerbehandlung.
R→B	Konvertieren von *n* in einen Binärwert.
THEN	Falls ein Fehler auftrat,
END	Keine Aktion (keine Befehle in der THEN Anweisung).
→ n	Erzeugen der lokalen Variablen *n*.
«	Beginnen des definierenden Programms.
CLLCD	Löschen der Anzeige.
« BIN »	Unterprogramm für BIN.
« OCT »	Unterprogramm für OCT.
« DEC »	Unterprogramm für DEC.
« HEX »	Unterprogramm für HEX.
1 4	Start- und Endwert des Zählers.
FOR j	Beginnen der Schleife mit Zähler *j*.
EVAL	Auswerten eines der Zahlensystem-Unterprogramme (als erstes HEX).
n →STR	Erzeugen eines Strings zum Anzeigen von *n* im momentanen Zahlensystem.
PAD	Auffüllen des Strings auf 23 Zeichen.
j DISP	Anzeigen des Strings in der *j*-ten Zeile.
NEXT	Erhöhen von *j* und Wiederholen der Schleife.
»	Beenden des definierenden Programms.
»	Beenden des Steuerungs-Unterprogramms.
PRESERVE	Speichern des momentanen Status, Ausführen des Steuerungs-Unterprogramms und Wiederherherstellen des alten Status.
»	Beenden des Programms.
ENTER	Eingeben des Programms in den Stack.
' BDISP STO	Speichern des Programms unter BDISP.

Beispiel: Schalten Sie auf das Dezimalsystem (DEC) um, zeigen Sie # 100 in allen Zahlensystemen an und überprüfen Sie, ob nach der Ausführung von BDISP das Dezimalsystem wieder eingestellt ist.

Löschen Sie den Stack und wählen Sie das BINARY Menü.

CLEAR
BINARY

Stellen Sie sicher, daß momentan das Dezimalsystem eingestellt ist und geben Sie # 100 ein.

DEC
100 ENTER

Führen Sie BDISP aus. (Wechseln Sie nicht Menüs, da im nächsten Schritt das Binärsystem angezeigt werden soll.)

BDISP ENTER

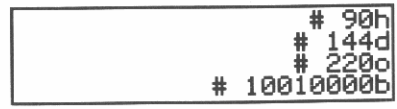
Kehren Sie zur normalen Stackanzeige zurück und überprüfen Sie das momentan eingestellte Zahlensystem.

ON

Obwohl das Steuerungs-Unterprogramm den Rechner im Binärsystem läßt, wird durch PRESERVE das Dezimalsystem wieder hergestellt.

Konvertieren Sie 144, um BDISP auch auf reelle Zahlen anzuwenden.

USER
144 BDISP



Summations-Statistik

Für die Statistik mit Stichproben-Paaren ist es oft nützlich, Quadratsummen (Σx^2 und Σy^2) und die Summe von Produkten (Σxy) zweier Variablen zu berechnen. Nachfolgend fünf Programme:

- SUMS erzeugt die Variable ΣCOV, die die Kovarianzmatrix für die momentane Statistikmatrix ΣDAT enthält.
- ΣGET entnimmt eine Zahl aus einer spezifizierten Position in ΣCOV.
- ΣX2 verwendet ΣGET, um Σx^2 aus ΣCOV zu entnehmen.
- ΣY2 verwendet ΣGET, um Σy^2 aus ΣCOV zu entnehmen.
- ΣXY verwendet ΣGET, um Σxy aus ΣCOV zu entnehmen.

Wenn ΣDAT n Spalten enthält, ist ΣCOV eine $n \times n$ Matrix. Die Programme ΣX2, ΣY2 und ΣXY beziehen sich auf ΣPAR (*Statistik-PARameter*), um die Spalten der x- und y-Werte (mit C_1 bzw. C_2 bezeichnet) zu bestimmen.

Programmiermethode:

- Matrixoperationen. Diese Programme zeigen, wie eine Matrix transponiert und wie ein Element aus einer Matrix entnommen werden kann.
- In algebraischen Objekten verwendbare Programme. Da ΣX2, ΣY2 und ΣXY der algebraischen Syntax entsprechen (keine Argumente vom Stack nehmen, ein Ergebnis in den Stack zurückgeben), können deren Namen in Ausdrücken und Gleichungen wie gewöhnliche Variablen verwendet werden.
- ΣPAR Vereinbarung. Mehrere Befehle zur Statistik mit Stichproben-Paaren verwenden zum Spezifizieren eines Spaltenpaares in ΣDAT eine Variable mit dem Namen ΣPAR. ΣPAR enthält eine Liste mit vier Zahlen, wobei die beiden ersten die Spalten spezifizieren. (Die beiden anderen geben die Steigung und den Abszissenabschnitt der Regressionsgeraden an.)

  SUMS stellt durch Ausführen von 0 PREDV DROP sicher, daß ΣPAR existiert. Falls ΣPAR nicht bereits existiert, erzeugt der Befehl PREDV (*PREDicted Value*) die Variable ΣPAR und weist ihr Standardwerte zu. DROP entfernt den für 0 berechneten Vorhersagewert.

  ΣX2, ΣY2 und ΣXY verwenden zur Bestimmung, welche Elemente aus ΣCOV zu entnehmen sind, die in ΣPAR gespeicherten Werte.

SUMS (Summations-Statistikmatrix)

Erzeugen Sie die Variable ΣCOV, die die Kovarianzmatrix der Statistikmatrix ΣDAT enthält.

Wenn ΣDAT z.B. die $n \times 2$ Matrix

$$\begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \vdots & \vdots \\ x_n & y_n \end{bmatrix}$$

ist, so enthält ΣCOV die Kovarianzmatrix

$$\begin{bmatrix} \Sigma x^2 & \Sigma xy \\ \Sigma xy & \Sigma y^2 \end{bmatrix}.$$

Argumente	Ergebnisse
1:	1:

Programm	Bemerkung
«	Beginnen des Programms.
RCLΣ	Zurückrufen des Inhalts der $n \times m$ Statistikmatrix ΣDAT.
DUP	Erstellen einer Kopie.
TRN	Transponieren der Matrix. Das Ergebnis ist eine $m \times n$ Matrix.
SWAP *	Multiplizieren der beiden Matrizen, um die $m \times m$ Kovarianzmatrix zu erstellen. (Wenn die beiden Matrizen nicht vertauscht würden, wäre das Ergebnis eine $n \times n$ Matrix.)
'ΣCOV' STO	Speichern der Kovarianzmatrix in der Variablen ΣCOV.
0 PREDV DROP	Sicherstellen, daß ΣPAR existiert.
»	Beenden des Programms.
ENTER	Eingeben des Programms in den Stack.
' SUMS STO	Speichern des Programms unter SUMS.

ΣGET (Abrufen eines Elements von ΣCOV)

Entnehmen Sie für gegebenes p und q, die entweder die erste oder die zweite *Position* in ΣPAR kennzeichnen, das Element *rs* von ΣCOV, wobei r und s die entsprechenden ersten oder zweiten *Elemente* in ΣPAR sind.

ΣGET wird von ΣX2, ΣY2 und ΣXY mit folgenden Argumenten aufgerufen.

- Für ΣX2 ist $p = 1$ und $q = 1$.
- Für ΣY2 ist $p = 2$ und $q = 2$.
- Für ΣXY ist $p = 1$ und $q = 2$.

Argumente	Ergebnisse
2: *1 oder 2* 1: *1 oder 2*	2: 1: *Element rs von ΣCOV*

Programm	Bemerkung
«	Beginnen des Programms.
ΣCOV	Speichern der Kovarianzmatrix im Stack.
ΣPAR	Speichern der Liste mit den Statistikparametern im Stack.
DUP	Erstellen einer Kopie.
5 ROLL	Verschieben von p in Ebene 1.
GET	Abrufen von r, das p-te Element in ΣPAR.
SWAP	Verschieben von ΣPAR in Ebene 1.
4 ROLL	Verschieben von q in Ebene 1.
GET	Abrufen von s, das q-te Element in ΣPAR.
2 →LIST	Speichern von { r, s } im Stack.
GET	Abrufen des Elements *rs* von ΣCOV.
»	Beenden des Programms.
ENTER	Eingeben des Programms in den Stack.
' ΣGET STO	Speichern des Programms unter ΣGET.

ΣX2 (Summe der Quadrate von x)

Berechnen Sie Σx^2, wobei die x-Werte die Elemente von C_1 sind (die Spalte, die durch den ersten Parameter in ΣPAR spezifiziert ist).

Argumente	Ergebnisse
1 :	1 : Σx^2

Programm

```
«
 1 1

 ΣGET
»
```
ENTER

' ΣX2 STO

Bemerkung

Beginnen des Programms.
Zweimaliges Spezifizieren von C_1.
Entnehmen von Σx^2.
Beenden des Programms.
Eingeben des Programms in den Stack.
Speichern des Programms unter ΣX2.

ΣY2 (Summe der Quadrate von y)

Berechnen Sie Σy^2, wobei die y-Werte die Elemente von C_2 sind (die Spalte, die durch den zweiten Parameter in ΣPAR spezifiziert ist).

Argumente	Ergebnisse
1 :	1 : Σy^2

Programm

```
«
 2 2

 ΣGET
»
```
ENTER

' ΣY2 STO

Bemerkung

Beginnen des Programms.
Zweimaliges Spezifizieren von C_2.
Entnehmen von Σy^2.
Beenden des Programms.
Eingeben des Programms in den Stack.
Speichern des Programms unter ΣY2.

ΣXY (Summe der Produkte von x und y)

Berechnen Sie Σxy, wobei die x- und y-Werte die entsprechenden Elemente von C_1 und C_2 sind (die Spalten, die durch den ersten und zweiten Parameter in ΣPAR spezifiziert sind).

Argumente	Ergebnisse
1:	1: Σxy

Programm	Bemerkung
«	Beginnen des Programms.
1 2	Spezifizieren von C_1 und C_2.
ΣGET	Entnehmen von Σxy.
»	Beenden des Programms.
ENTER	Eingeben des Programms in den Stack.
' ΣXY STO	Speichern des Programms unter ΣXY.

Beispiel: Berechnen Sie ΣX2, ΣY2 und ΣXY für folgende statistische Daten:

$$\begin{bmatrix} 18 & 12 \\ 4 & 7 \\ 3 & 2 \\ 11 & 1 \\ 31 & 48 \\ 20 & 17 \end{bmatrix}$$

Die allgemeine Vorgehensweise lautet folgendermaßen:

1. Eingeben der statistischen Daten.
2. Ausführen von SUMS und Erzeugen der Kovarianzmatrix ΣCOV.
3. Ausführen von ΣX2, ΣY2 und ΣXY.
4. Falls ΣDAT mehr als zwei Spalten enthält (d.h., falls jeder Datenpunkt mehr als zwei Variablen enthält):
   a. Ausführen von COLΣ, um neue Werte für C_1 und C_2 zu spezifizieren. Diese Werte werden in ΣPAR gespeichert.
   b. Ausführen von ΣX2, ΣY2 und ΣXY.

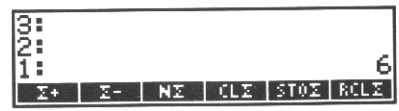
Versuchen Sie nun, das oben gegebene Beispiel auszuführen.

Löschen Sie den Stack, wählen Sie das STAT Menü und löschen Sie ΣDAT.

CLEAR
STAT
CLΣ

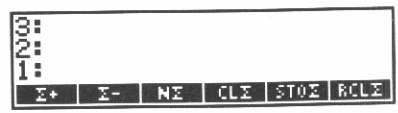
Geben Sie die Daten ein und überprüfen Sie dann, ob Sie alle sechs Datenpunkte eingegeben haben.

[ ] 18,12 Σ+
[ ] 4,7 Σ+
[ ] 3,2 Σ+
[ ] 11,1 Σ+
[ ] 31,48 Σ+
[ ] 20,17 Σ+
NΣ



Löschen Sie die Anzahl der Datenpunkte.

DROP



Erzeugen Sie die Kovarianzmatrix ΣCOV.

USER
SUMS

Berechnen Sie Σx^2.

ΣX2

Berechnen Sie Σy^2.

ΣY2

Berechnen Sie Σxy.

ΣXY

Wenn die Statistikmatrix mehr als zwei Spalten aufweisen würde, könnten Sie neue Werte für C_1 und C_2 spezifizieren. Geben Sie zur Übung $C_1 = 1$ und $C_2 = 2$ an (die momentanen Werte).

Der Befehl COLΣ ist zwar im STAT Menü verfügbar, es ist hier jedoch einfacher, im USER Menü zu bleiben und den Befehl auszuschreiben.

1 ENTER
2 COLΣ ENTER

Sie könnten nun ΣX2, ΣY2 und ΣXY für das neue Spaltenpaar C_1 und C_2 ausführen.

Vergessen Sie nicht, jedesmal SUMS auszuführen, wenn Sie Daten der Statistikmatrix ΣDAT hinzufügen oder aus ihr löschen.

Median von statistischen Daten

Dieser Abschnitt enthält drei Programme:

- SORT sortiert die Elemente einer Liste.
- LMED berechnet den Median für eine sortierte Liste.
- MEDIAN verwendet SORT und LMED, um den Median der momentanen statistischen Daten zu bestimmen.

SORT (Sortieren einer Liste)

Sortieren Sie eine Liste in aufsteigender Reihenfolge.

Argumente	Ergebnisse
1 : ⟨ *Liste* ⟩	1 : ⟨ *sortierte Liste* ⟩

Programmiermethode:

- Sortieren. SORT vergleicht, beginnend mit der ersten und zweiten Zahl in einer Liste, benachbarte Zahlen und verschiebt die größere in Richtung Listenende. Dieser Vorgang wird das erste Mal ausgeführt, um die größte Zahl an das Listenende zu verschieben, das zweite Mal, um die zweitgrößte Zahl an die zweitletzte Position der Liste zu verschieben, usw. für die restlichen Listenelemente.
- Geschachtelte bestimmte Schleifen. Die äußere Schleife steuert für jede Ausführung des Vorgangs die Halt-Position, die innere Schleife geht dabei jedesmal von 1 bis zur Halt-Position.
- Geschachtelte Strukturen von lokalen Variablen. SORT enthält zwei Strukturen von lokalen Variablen, wobei sich die zweite innerhalb des definierenden Programms der ersten Struktur befindet. Durch diese Schachtelung ist es einfacher, die erste lokale Variable zu erzeugen, sobald ihr Wert berechnet wird. Dadurch wird der Wert der Variablen vom Stack entfernt, und es werden nicht beide Werte gleichzeitig berechnet und beide lokalen Variablen gleichzeitig erzeugt.

- FOR ... STEP und FOR ... NEXT (bestimmte Schleifen). SORT verwendet zwei Zähler: — 1 STEP vermindert den Zähler für die äußere Schleife bei jedem Durchlauf um eins; NEXT erhöht den Zähler für die innere Schleife bei jedem Durchlauf um eins.

Programm	**Bemerkung**
`«`	Beginnen des Programms.
`DUP SIZE 1 - 1`	Von der zweitletzten bis zur ersten Position,
`FOR j`	Beginnen der äußeren Schleife mit Zähler *j*.
`1 j`	Von der ersten bis zur *j*-ten Position,
`FOR k`	Beginnen der inneren Schleife mit Zähler *k*.
`k GETI → n1`	Abrufen der *k*-ten Zahl in der Liste und Speichern in der lokalen Variablen n_1.
`«`	Beginnen des äußeren definierenden Programms.
`GETI → n2`	Abrufen der nächsten Zahl in der Liste und Speichern in der lokalen Variablen n_2.
`«`	Beginnen des inneren definierenden Programms.
`DROP`	Löschen des Zählers.
`IF n1 n2 >`	Falls die beiden Zahlen in der falschen Reihenfolge stehen,
`THEN`	folgendes ausführen:
`k n2 PUTI`	Zurückspeichern der zweiten Zahl in die *k*-te Position.
`n1 PUT`	Zurückspeichern der *k*-ten Zahl in die nächste Position.
`END`	Beenden der THEN Anweisung.
`»`	Beenden des inneren definierenden Programms.
`»`	Beenden des äußeren definierenden Programms.
`NEXT`	Erhöhen von *k* und Wiederholen der inneren Schleife.
`-1 STEP`	Vermindern von *j* und Wiederholen der äußeren Schleife.
`»`	Beenden des Programms.
ENTER	Eingeben des Programms in den Stack.
' SORT STO	Speichern des Programms unter SORT.

Beispiel:

Sortieren Sie die Liste { 8, 3, 1, 2, 5 }.

USER
{ } 8.3,1.2,5 SORT

LMED (Median einer Liste)

Berechnen Sie den Median für eine gegebene sortierte Liste. Wenn die Liste eine ungerade Anzahl von Elementen enthält, ist der Median das Element in der Mitte der Liste. Wenn die Liste eine gerade Anzahl von Elementen enthält, ergibt sich der Median als Mittelwert der beiden zur Mitte benachbarten Elemente der Liste.

Argumente	Ergebnisse
1: { *sortierte Liste* }	1: *Median der sortierten Liste*

Programmiermethode:

- FLOOR und CEIL. Wenn das Argument eine ganze Zahl ist, gibt sowohl FLOOR als auch CEIL diese ganze Zahl zurück; wenn das Argument keine ganze Zahl ist, gibt FLOOR die nächst kleinere ganze Zahl und CEIL die nächst größere ganze Zahl zurück.

Programm	Bemerkung
«	Beginnen des Programms.
DUP SIZE	Die Größe der Liste.
1 + 2 /	Die mittlere Position der Liste (gebrochene Zahl, falls Listengröße geradzahlig ist).
→ p	Speichern der mittleren Listenposition in der lokalen Variablen *p*.
«	Beginnen des definierenden Programms.
DUP	Erstellen einer Kopie der Liste.
p FLOOR GET	Abrufen der Zahl in oder direkt unterhalb der mittleren Position.
SWAP	Verschieben der Liste in Ebene 1.
p CEIL GET	Abrufen der Zahl in oder direkt oberhalb der mittleren Position.

Programm	Bemerkung
+ 2 /	Der Mittelwert der beiden Zahlen in der bzw. um die mittlere Position.
»	Beenden des definierenden Programms.
»	Beenden des Programms.
ENTER	Eingeben des Programms in den Stack.
' LMED STO	Speichern des Programms unter LMED.

Beispiel:

Berechnen Sie den Median der Liste, die Sie mit SORT sortiert haben.

USER
LMED

LMED wird von MEDIAN aufgerufen.

MEDIAN (Median von statistischen Daten)

Geben Sie einen Vektor zurück, der aus den Medianwerten der Spalten der statistischen Daten besteht.

Argumente	Ergebnisse
1:	1: [x_1 x_2 ... x_m]

Programmiermethode:

- Felder, Listen und Stackelemente. MEDIAN entnimmt eine Spalte mit Daten aus ΣDAT in Vektorform. Um den Vektor in eine Liste zu konvertieren, legt MEDIAN die Vektorelemente im Stack ab und fügt sie zu einer Liste zusammen. Aus dieser Liste wird dann unter Verwendung von SORT und LMED der Median berechnet.

  Der Median für die m-te Spalte wird zuerst und der Median für die erste Spalte zuletzt berechnet, wobei der jeweils berechnete Median in die nächst höhere Stackebene verschoben wird.

Nachdem alle Medianwerte berechnet und in der richtigen Reihenfolge im Stack abgelgt sind, werden sie zu einem Vektor zusammengefaßt.

- FOR ... NEXT (bestimmte Schleife mit Zähler). MEDIAN verwendet zum Berechnen des Medians für jede Spalte eine Schleife. Da die Medianwerte in umgekehrter Reihenfolge (letzte Spalte zuerst) berechnet werden, wird der Zähler zur Umkehrung der Reihenfolge der Medianwerte benutzt.

Benötigte Programme:

- SORT (auf Seite 270) legt eine Liste in aufsteigender Reihenfolge an.
- LMED (auf Seite 272) berechnet den Median einer sortierten Liste.

Programm	Bemerkung
`«`	Beginnen des Programms.
`RCLΣ`	Speichern einer Kopie der momentanen Statistikmatrix ΣDAT im Stack (zur Sicherung).
`DUP SIZE`	Speichern der Liste { *n m* } im Stack, wobei *n* die Anzahl der Zeilen und *m* die Anzahl der Spalten in ΣDAT ist.
`LIST→ DROP`	Speichern von *n* und *m* im Stack. Wert für Listengröße löschen.
`→ n m`	Erzeugen von lokalen Variablen für *n* und *m*.
`«`	Beginnen des definierenden Programms.
`'ΣDAT' TRN`	Transponieren von ΣDAT. Nun ist *n* die Anzahl der Spalten und *m* die Anzahl der Zeilen in ΣDAT.
`1 m`	Die erste und letzte Zeile.
`FOR j`	Für jede Zeile:
`Σ-`	Entnehmen der letzten Zeile in ΣDAT. Es wird mit der *m*-ten Zeile begonnen, die der *m*-ten Spalte der ursprünglichen ΣDAT entspricht.
`ARRY→ DROP`	Speichern der Zeilenelemente im Stack. Löschen der Indexliste { *n* }, da *n* bereits in einer lokalen Variablen gespeichert ist.
`n →LIST`	Anlegen einer Liste mit *n* Elementen.
`SORT`	Sortieren der Liste.
`LMED`	Berechnen des Medians der Liste.

Programm	Bemerkung
`j ROLLD`	Verschieben des Medians in entsprechende Stackebene.
`NEXT`	Erhöhen von *j* und Wiederholen der Schleife.
`m 1 →LIST`	Anlegen der Liste { *m* }.
`→ARRY`	Zusammenfügen aller Medianwerte in einem Vektor mit *m* Elementen.
`»`	Beenden des definierenden Programms.
`SWAP`	Verschieben des ursprünglichen ΣDAT in Ebene 1.
`STOΣ`	Zurückrufen von ΣDAT mit früheren Werten.
`»`	Beenden des Programms.
ENTER	Eingeben des Programms in den Stack.
' MEDIAN STO	Speichern des Programms unter MEDIAN.

Beispiel: Berechnen Sie den Median der Daten auf Seite 268. (Es wird vorausgesetzt, daß Sie die Daten eingetippt haben.) Es gibt zwei Spalten mit Daten. MEDIAN gibt folglich einen Vektor mit zwei Elementen zurück.

Berechnen Sie den Median.

USER
MEDI

Der Median beträgt für die erste Spalte 14.4 und für die zweite 9.5.

Wechseln von Verzeichnissen

Dieser Abschnitt enthält zwei Programme:

- UP erzeugt ein Menü aus Oberverzeichnissen.
- DOWN erzeugt ein Menü aus Unterverzeichnissen.

Diese Programme haben für diejenigen, die immer ihre ganze Verzeichnisstruktur kennen und immer wissen, wo sie sich gerade befinden, keine Verwendung. Für diejenigen, die gelegentlich verwirrt werden, sind diese Programme hilfreich.

UP (Wechsel zu einem Oberverzeichnis)

Erzeugen Sie ein Menü, das die Namen seines Oberverzeichnisses und dessen Oberverzeichnisses usw. bis zum HOME Verzeichnis (d.h. aller übergeordneten Verzeichnisse) enthält.

Argumente	Ergebnisse
1:	1:

Programmiermethode:

- Liste der Oberverzeichnisse. UP verwendet PATH, um den Namen des aktuellen Verzeichnisses sowie aller übergeordneten Verzeichnisse (Oberverzeichnisse) zurückzugeben.
- Untermenge einer Liste. UP verwendet SUB, um den Namen des aktuellen Verzeichnisses aus der PATH Liste zu entfernen.
- CUSTOM Menü. UP verwendet MENU, um aus der modifizierten PATH Liste ein CUSTOM Menü mit Oberverzeichnissen zu erzeugen.

Programm	Bemerkung
«	Beginnen des Programms.
PATH	Speichern der PATH Liste im Stack.
1	Speichern von 1 im Stack.
OVER SIZE 1 -	Speichern von *Größe* − 1 im Stack.
SUB	Erzeugen einer Untermenge aus der PATH Liste, die alle Namen außer dem letzten (dem aktuellen Verzeichnis) enthält.
MENU	Erzeugen eines Menüs mit Oberverzeichnissen.
»	Beenden des Programms.
ENTER	Eingeben des Programms in den Stack.
' UP STO	Speichern des Programms in UP.

Beispiel: Erzeugen Sie ausgehend vom HOME Verzeichnis eine Hierarchie von Unterverzeichnissen D1, D2 und D3; verwenden Sie dann UP, um von D3 nach D1 zu wechseln.

Löschen Sie den Stack und wechseln Sie zum HOME Verzeichnis.

CLEAR
MEMORY HOME

Erzeugen Sie das Unterverzeichnis D1 und wechseln Sie zu ihm.

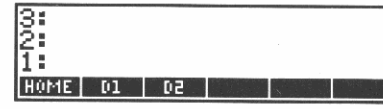
' D1 CRDIR
D1 ENTER

Wiederholen Sie den Vorgang für die Unterverzeichnisse D2 und D3.

' D2 CRDIR
D2 ENTER
' D3 CRDIR
D3 ENTER

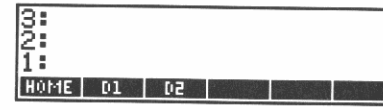
Zeigen Sie das Menü von Oberverzeichnissen an.

UP ENTER



Wechseln Sie zum Verzeichnis D1.

D1



DOWN (Wechseln zu einem Unterverzeichnis)

Erzeugen Sie eine Menü, das die Namen aller Unterverzeichnisse des aktuellen Verzeichnisses enthält.

Argumente	Ergebnisse
1:	1:

Programmiermethode:

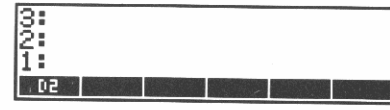
- Variablenliste. DOWN verwendet VARS, um die Variablenliste und die Liste mit Unterverzeichnissen des aktuellen Verzeichnisses zurückzugeben.
- Fehlerbehandlung. Zur Prüfung, ob ein Name in der VARS Liste ein Verzeichnis darstellt, verwendet DOWN den Namen als Argument von RCL; da Verzeichnisse nicht in den Stack zurückgegeben werden können, tritt ein Fehler auf, wenn der Name ein Verzeichnis ist. Der Name wird dann der Liste mit Namen von Verzeichnissen hinzugefügt.

Programm	**Bemerkung**
`«`	Beginnen des Programms.
`VARS`	Speichern einer Liste der Namen aller Variablen und Unterverzeichnisse im Stack.
`→ v`	Speichern der VARS Liste in der lokalen Variable *v*.
`«`	Beginnen des definierenden Programms.
`{ }`	Speichern der Liste mit Verzeichnisnamen im Stack (zu Beginn leer).
`1 v SIZE`	Speichern von 1 und *Größe von v* im Stack.
`FOR j`	Für jeden Namen in *v*:
`v j GET`	Abrufen des Namens.
`IFERR RCL DROP`	Versuchen, den Inhalt einer Variablen mit dem Namen abzurufen; falls erfolgreich, Inhalt löschen.
`THEN +`	Falls RCL einen Fehler verursacht, muß der Name ein Verzeichnisname sein. Folglich Name der Liste mit Verzeichnisnamen hinzufügen.
`END`	Beenden der THEN Anweisung und der Programmstruktur.
`NEXT`	Für nächsten Namen in *v* wiederholen.
`MENU`	Erzeugen eines CUSTOM Menüs für die Verzeichnisnamen.
`»`	Beenden des definierenden Programms.
`»`	Beenden des Programms.
ENTER	Eingeben des Programms in den Stack.
' DOWN STO	Speichern des Programms unter DOWN.

Beispiel: Im vorherigen Beispiel (auf Seite 277) haben Sie eine Hierarchie der Unterverzeichnisse D1, D2 und D3 angelegt und das Beispiel mit D1 als aktuellem Verzeichnis beendet. Wechseln Sie für dieses Beispiel von D2 nach D3.

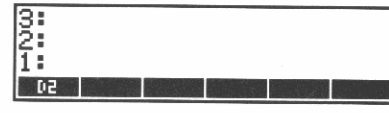
Zeigen Sie das Menü von Unterverzeichnissen an.

DOWN ENTER



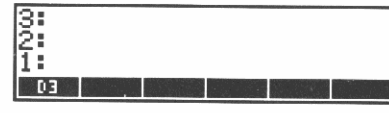
Wechseln Sie nach D2.

D2



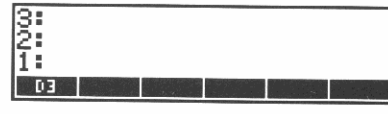
Zeigen Sie das Menü von Unterverzeichnissen an.

DOWN ENTER



Wechseln Sie nach D3.

D3



Anhänge & Index

A

Kundenunterstützung, Batterien und Service

Über diesen Anhang erhalten Sie Informationen, die Ihnen beim Auftreten von Problemen mit Ihrem Rechner weiterhelfen sollen. Wenn Sie Schwierigkeiten bei der Anwendung des Rechners haben und Sie im Inhaltsverzeichnis (Seite 5) kein entsprechendes Thema finden, beziehen Sie sich zur Lösung Ihres Problems auf den folgenden Abschnitt. Sollten Sie dort nicht die erwarteten Antworten auf Ihre Fragen finden, können Sie sich über die Adresse auf der Innenseite des Rückumschlags mit Hewlett-Packard in Verbindung setzen.

Wenn Sie die Batterien ersetzen möchten, so beziehen Sie sich auf Seite 286. Wenn Sie vermuten, daß Ihr HP-28S nicht zuverlässig arbeitet, lesen Sie bitte den Abschnitt "Feststellen der Reparaturbedürftigkeit" auf Seite 289 durch. Sollte es sich herausstellen, daß der Rechner zur Reparatur eingeschickt werden muß, so beziehen Sie sich auf die Hinweise unter "Einjährige Gewährleistungsfrist" auf Seite 292 und "Im Reparaturfall" auf Seite 293.

Antworten auf allgemeine Fragen

F: *Warum läßt sich der Rechner nicht einschalten, wenn* [ON] *gedrückt wird?*
A: Dies kann an einem relativ einfach zu lösenden Problem liegen, oder es ist eine Reparatur erforderlich. Beziehen Sie sich auf "Feststellen der Reparaturbedürftigkeit" auf Seite 289.

F: *Wie kann überprüft werden, ob der Rechner einwandfrei funktioniert?*
A: Führen Sie den auf Seite 290 beschriebenen Test durch.

F: *Welche Möglichkeit gibt es zum Löschen des Speicherinhalts?*
A: Halten Sie die drei Tasten [ON] [INS] [▶] gedrückt, und geben Sie diese in der Reihenfolge frei, wie es unter "Löschen des gesamten Speicherbereichs (Memory Reset)" auf Seite 20 beschrieben ist.

F: *Was bedeuten die drei Punkte (...) am Ende der Anzeigezeile?*
A: Diese Punkte, auch als *Ellipse* bezeichnet, deuten auf weitere (momentan unsichtbare) Zeichen des angezeigten Objekts hin.

F: *Wie erhält man die vollständige Anzeige eines Objekts?*
A: Verwenden Sie ■ EDIT oder ■ VISIT zur Anzeige des Objekts in der Befehlszeile. Sie können danach mit Hilfe der Cursortasten jeden Teil des Objekts anzeigen. Durch Drücken von ON können Sie die Edierfunktion wieder aufheben.

F: *Was ist mit "Objekt" gemeint?*
A: "Objekt" ist ein allgemeiner Begriff für fast alles, was von Ihnen bearbeitet wird. Zahlen, algebraische Ausdrücke, Felder, Programme, usw. werden als Objekte bezeichnet. Beziehen Sie sich auf den Abschnitt "Wichtigste Fähigkeiten und Konzepte" auf Seite 25, wenn Sie eine kurze Beschreibung der Objekttypen suchen.

F: *Der Rechner gibt ein Akustiksignal und die Fehlermeldung* `Bad Argument Type` *aus. Was hat dies zu bedeuten?*
A: Die Objekte im Stack sind für den gewünschten Befehl nicht zulässig. Diese Fehlersituation kann z.B. dann auftreten, wenn Sie STO drücken, ohne daß sich in Ebene 1 ein Objekt befindet. Verwenden Sie CATALOG, um die erforderlichen Argumente für einen Befehl nachzusehen (siehe "Der Befehlskatalog" auf Seite 31).

F: *Der Rechner gibt ein Akustiksignal aus und zeigt die Meldung* `Too Few Arguments` *an. Was hat dies zu bedeuten?*
A: Es waren zu wenig Argumente im Stack gespeichert, um den von Ihnen benutzten Befehl auszuführen (es ist z.B. nur ein Argument im Stack und es sollte + ausgeführt werden). Verwenden Sie CATALOG, um die korrekte Anzahl von Argumenten für den Befehl nachzusehen (siehe "Der Befehlskatalog" auf Seite 31).

F: *Der Rechner gibt ein Akustiksignal aus und zeigt eine zu den oberen Fehlermeldungen unterschiedliche Meldung an. Wie kann herausgefunden werden, was die Meldung verursacht?*
A: Beziehen Sie sich auf Anhang A im Referenzhandbuch.

F: *Wie kann das Akustiksignal ausgeschaltet werden?*
A: Tippen Sie `51 SF` ENTER. Dadurch wird Flag 51 gesetzt, welches den Warnton abstellt.

F: *Wie kann eine Kopie des Anzeigeinhalts gedruckt werden?*
A: Halten Sie ON gedrückt, während Sie L drücken, und geben Sie danach ON wieder frei.

F: *Die Tasten* [A] *bis* [R] *funktionieren nicht. Weshalb?*
A: Sie haben wahrscheinlich die Menüsperre aktiviert, wonach durch die Tasten [A] bis [R] ein Menü aufgerufen wird, sofern nicht zuerst ■ gedrückt wird. Um die Menüsperre auszuschalten, ist ■ [MENUS] zu drücken.

F: *Einige zuvor benutzte Variablen sind anscheinend verschwunden. Wo können diese gespeichert sein?*
A: Sie haben die Variablen vielleicht in einem anderen Verzeichnis verwendet. Wenn Sie den Verzeichnisnamen nicht mehr kennen, so müssen Sie alle Verzeichnisse überprüfen.

F: *Wie kann die Größe des freien Speicherbereichs festgestellt werden?*
A: Führen Sie MEM [ENTER] aus, um die Anzahl der verfügbaren freien Bytes angezeigt zu erhalten.

F: *Warum hat sich das Aussehen des Cursors geändert?*
A: Der Cursor gibt Auskunft über den momentanen Eingabemodus. Es gibt den unmittelbaren (leerer Cursor), algebraischen (teilweise ausgefüllter Cursor) und Alpha-Eingabemodus (ausgefüllter Cursor). Die Form des Cursors kennzeichnet den Ersetzungsmodus (Rechtecks-Cursor) oder den Einfügungsmodus (Pfeil-Cursor). Näheres finden Sie auf Seite 172 unter "Wie der Cursor verschiedene Modi darstellt".

F: *Es wurde ein Name eingetippt (oder eine USER-Menütaste gedrückt), ohne daß der Name in den Stack übernommen wurde. Warum?*
A: Sie haben den Namen *ohne Anführungszeichen* eingegeben, wodurch auf den *Inhalt* einer Variablen Bezug genommen wird. Um einen Namen in den Stack einzugeben, drücken Sie zuerst ['].

F: *Die Berechnung der dritten Wurzel von −27 ergibt nicht −3. Warum?*
A: Jede Zahl hat drei 3. Wurzeln, von welchen zwei komplexe Zahlen sind. Der HP-28S gibt eine der drei Wurzeln—als Hauptwert bezeichnet—zurück. Bei positiven reellen Argumenten stellt der Hauptwert die reelle Wurzel dar; bei negativen reellen Argumenten ist der Hauptwert eine der komplexen Wurzeln. Sie können die *b*-te Wurzel einer reellen Zahl *a* über folgendes Programm berechnen:

```
« → a b 'SIGN(a)*ABS(a)^INV(b)' »
```

Drücken Sie ['] RWURZEL [STO], um das Programm in der Variablen RWURZEL zu speichern. Sie können nun die reelle 3. Wurzel von −27 durch Eintippen von 27 [CHS] [ENTER] 3 [ENTER] RWURZEL [ENTER] berechnen.

F: *Der Rechner arbeitet langsamer als sonst, außerdem blinkt der* 💾 *Indikator. Was ist geschehen?*
A: Der Rechner befindet sich im Protokoll-Druckmodus. Drücken Sie ■ PRINT TRAC, um diesen Druckmodus auszuschalten.

F: *Der Drucker gibt mehrere Zeilen relativ schnell aus und verlangsamt danach die Druckgeschwindigkeit. Warum?*
A: Der Rechner überträgt eine bestimmte Datenmenge an den Drucker und verlangsamt danach die Übertragungsgeschwindigkeit, um sicherzustellen, daß der Drucker mit der Druckausgabe Schritt halten kann.

F: *Der Drucker "verliert" Zeichen oder druckt gelegentlich das Zeichen ■. Woran kann das liegen?*
A: Die Entfernung oder der Übertragungswinkel zwischen dem Rechner und dem Drucker kann zu groß sein. Angaben über Einschränkungen finden Sie im Benutzerhandbuch des Druckers.

F: *Wie kann die Druckgeschwindigkeit erhöht werden?*
A: Wird der Drucker über einen Adapter betrieben, so kann der Rechner die Daten mit einer höheren Rate übertragen, indem Sie `52 SF` ENTER tippen (dies setzt Flag 62 zur Steuerung der Druckgeschwindigkeit).

F: *Was ist der Unterschied zwischen STO und STORE?*
A: Der STO Befehl weist einer Variablen einen bestimmten Wert zu. Im STORE Menü sind Befehle zur Speicherarithmetik enthalten; diese Befehle verwenden den Inhalt einer Variablen als Argument und weisen das Ergebnis wieder der Variablen zu.

F: *Anstatt eines symbolischen Ergebnisses hat der Rechner ein numerisches Ergebnis ausgegeben. Weshalb?*
A: Für eine oder mehrere Variablen wurden bereits numerische Werte zugewiesen. Löschen Sie den Inhalt dieser Variablen. (Beziehen Sie sich dazu auf "Löschen einer Variablen" auf Seite 53.)

F: *Nach Drücken von* DRAW *wird die Anzeige gelöscht und der* ((•)) *Indikator beginnt zu blinken—es wird jedoch keine Grafik angezeigt.*
A: Die berechneten Werte liegen außerhalb des Anzeigebereichs (siehe "Ändern des Abbildungsmaßstabs" auf Seite 91).

F: *Nach der Auswertung einer Variablen (oder eines Ausdrucks) reagiert der Rechner nicht mehr auf das Drücken einer Taste. Was ist geschehen?*
A: Sie haben wahrscheinlich eine Variable definiert, welche auf sich selbst Bezug nimmt. Dies bewirkt die Ausführung einer "Endlosschleife". Um diese Schleife abzubrechen, ist ein Systemstopp, wie nachstehend beschrieben, auszuführen:

1. Halten Sie [ON] gedrückt.
2. Drücken Sie [▲].
3. Geben Sie [ON] wieder frei.

Definieren Sie nun die Variable um, damit die Endlosschleife beseitigt wird.

Wenn diese Antworten nicht zu der Lösung Ihres Problems geführt haben, können Sie sich über die Adresse auf der Innenseite des Rückumschlags mit Hewlett-Packard in Verbindung setzen.

Batterien

Der HP-28S wird mit drei Alkali-Batterien betrieben. Bei "typischem" Einsatz kann der Rechner mit einem neuen Batteriesatz 6 bis 12 Monate arbeiten. Die tatsächliche Zeit hängt allerdings von der spezifischen Anwendungsart des Rechners ab.

Verwenden Sie nur neue Batterien (Typ N), keine aufladbaren.

"Schwache Batterie" Indikator

Wenn der Rechner eine abfallende Batteriespannung erkennt (der Indikator  erscheint im oberen Teil der Anzeige), kann der HP-28S mindestens noch für 10 Stunden weiter betrieben werden. Im ausgeschalteten Zustand bleibt der Inhalt des Permanentspeichers noch für etwa einen Monat erhalten.

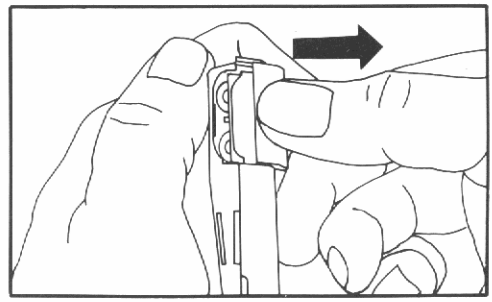
Einsetzen der Batterien

Wenn Sie den HP-28S gerade gekauft haben und die Batterien zum ersten Mal einsetzen, können Sie sich dazu beliebig viel Zeit lassen.

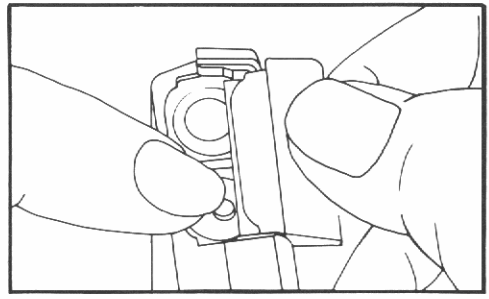
Wird jedoch ein verbrauchter Batteriesatz ersetzt, sollten Sie daran denken, daß zum Erhalten der im Permanentspeicher abgelegten Daten nur eine begrenzte Zeit zur Verfügung steht. Ist das Batteriefach geöffnet worden, so muß innerhalb einer Minute der neue Batteriesatz eingesetzt und das Batteriefach wieder geschlossen werden, wenn kein Datenverlust erfolgen soll. Die neuen Batterien sollten deshalb direkt greifbar sein. Außerdem muß der Rechner während des gesamten Vorgangs ausgeschaltet sein.

Um die Batterien einzusetzen:

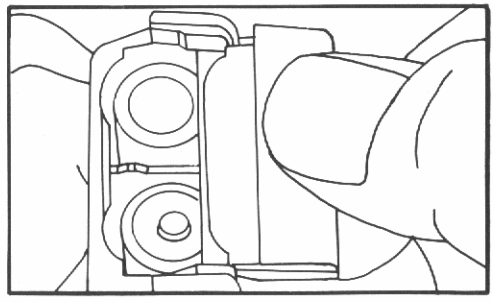
1. Halten Sie die drei neuen Batterien griffbereit.
2. Öffnen Sie den Rechner, um die Batteriefach-Abdeckung entnehmen zu können. Stellen Sie hierbei sicher, daß der Rechner ausgeschaltet ist. *Drücken Sie nicht* [ON], *bevor das Austauschen der Batterien abgeschlossen ist. Wird der Rechner vorher eingeschaltet, so kann dies die Löschung des Permanentspeichers zur Folge haben.*
3. Halten Sie den Rechner so, daß die Öffnung des Batteriefachs nach oben zeigt. Um die Abdeckung des Batteriefachs abzunehmen, schieben Sie diese in Richtung der Rechnerrückseite.



4. Drehen Sie den Rechner um, damit Sie die Batterien entnehmen können.
5. Setzen Sie die drei neuen Batterien ein. Achten Sie dabei auf deren richtige Polarität; diese ist auf der Rückseite des Rechners abgebildet. Stellen Sie die korrekte Polarität sicher, bevor Sie die Abdeckung wieder einsetzen.
6. Drücken Sie die Batterien in das Batteriefach, indem Sie dazu die Abdeckung teilweise in die vorgesehenen Führungsnuten einschieben.



7. Nehmen Sie Ihre Finger zu Hilfe, um die Batterien vollständig in das Batteriefach zu drücken und schieben Sie dann die Abdeckung über die Batterien, bis das Fach wieder komplett verschlossen ist.



Pflege des Rechners

Um die Anzeige des Rechners zu reinigen, sollten Sie ein leicht angefeuchtetes Tuch benutzen. Vermeiden Sie jede direkte Nässeeinwirkung auf den Rechner.

Das Scharnier ist wartungsfrei und benötigt daher keine Pflege.

Umgebungsbedingungen

Im Hinblick auf die Produktzuverlässigkeit sollten Sie folgende Temperatur- und Luftfeuchtigkeitsgrenzen für den HP-28S einhalten:

- Betriebstemperatur: 0° bis 45°C
- Lagerungstemperatur: −20° bis 65°C
- Luftfeuchtigkeit für Betrieb und Lagerung: 90% relative Luftfeuchtigkeit bei max. 40°C

Feststellen der Reparaturbedürftigkeit

Benutzen Sie nachstehende Richtlinien, um die zuverlässige Funktionsweise des Rechners zu überprüfen. Wenn der Rechner repariert werden muß, beachten Sie bitte die Abschnitte "Einjährige Gewährleistungsfrist" auf Seite 292 und "Im Reparaturfall" auf Seite 293.

Wenn nach dem Drücken von [ON] nichts angezeigt wird:

1. Überprüfen Sie die Kontrasteinstellung.
   - **a.** Halten Sie [ON] gedrückt.
   - **b.** Drücken Sie mehrmals [+].
   - **c.** Geben Sie [ON] wieder frei.
   - **d.** Ist die Anzeige noch immer leer, drücken Sie [ON] und wiederholen die Schritte a, b und c.

2. Tauschen Sie die Batterien aus (auf Seite 287 beschrieben).
3. Wenn die Schritte 1 und 2 keine Abhilfe herbeiführen, ist eine Reparatur des Rechners erforderlich. Beziehen Sie sich auf die Abschnitte "Einjährige Gewährleistungsfrist" auf Seite 292 und "Im Reparaturfall" auf Seite 293.

Wenn Daten in der Anzeige erscheinen, das Drücken von Tasten jedoch keinen Einfluß auf den Rechner hat:

1. Führen Sie einen Systemstopp durch.
   a. Halten Sie [ON] gedrückt.
   b. Drücken Sie [▲].
   c. Geben Sie [ON] wieder frei.
2. Führen Sie ein "Memory Reset" durch.
   a. Halten Sie [ON] gedrückt.
   b. Halten Sie [INS] und [▶] gedrückt.
   c. Geben Sie [INS] und [▶] wieder frei.
   d. Geben Sie [ON] wieder frei.
3. Wenn die Schritte 1 und 2 keine Abhilfe herbeiführen, ist eine Reparatur des Rechners erforderlich. Beziehen Sie sich auf die Abschnitte "Einjährige Gewährleistungsfrist" auf Seite 292 und "Im Reparaturfall" auf Seite 293.

Der Elektronik-Test

Wenn Sie sich nicht sicher sind, ob der Rechner einwandfrei arbeitet:

1. Wenn Sie über einen Drucker verfügen, so schalten Sie diesen ein.
2. Starten Sie den Elektronik-Test.
   a. Halten Sie [ON] gedrückt.
   b. Drücken Sie [◀].
   c. Geben Sie [ON] wieder frei.

Der Test wird automatisch durchlaufen und wiederholt sich fortlaufend. (Ist dies nicht der Fall, so haben Sie wahrscheinlich aus Versehen die Tasten [ON] [▼] gedrückt. Dies bewirkt den Start eines weiteren Test, welcher werksseitig benutzt wird und Eingaben über das Tastenfeld erwartet. Beenden Sie diesen Test, indem Sie einen Systemstopp, wie unter Schritt 4 beschrieben, durchführen. Starten Sie danach den Elektronik-Test.)

3. Beachten Sie die Meldung dieses Testprogramms. Zuerst erscheinen horizontale und vertikale Linien, danach eine leere Anzeige, ein beliebiges Punktmuster und am Schluß das Ergebnis des Tests.

- Die Meldung `0K-28S` bedeutet, daß der Rechner den Test bestanden hat.
- Eine Meldung wie `1 FAIL` zeigt an, daß der Test nicht fehlerfrei durchlaufen wurde. Die Ziffer gibt über die Natur der Fehlerursache Auskunft und sollte bei der Einsendung des Rechners zur Reparatur mit angegeben werden.

  Wenn Sie den Elektronik-Test durch das Drücken einer Taste unterbrochen haben, *bedeutet die Anzeige einer Fehlermeldung nicht, daß der Rechner fehlerhaft arbeitet.*

4. Beenden Sie den Test durch Ausführen eines Systemstopps.

a. Halten Sie [ON] gedrückt.

b. Drücken Sie [▲].

c. Geben Sie [ON] wieder frei.

5. Wenn der Rechner den Test mit einer Fehlermeldung abschließt, *wobei diese Meldung nicht durch eine Unterbrechung des Tests Ihrerseits verursacht wurde,* dann ist eine Reparatur des Rechners erforderlich. Sehen Sie dazu auch "Einjährige Gewährleistungsfrist" und "Im Reparaturfall" auf Seite 293.

Einjährige Gewährleistungsfrist

Hewlett-Packard gewährleistet, daß der Rechner frei von Material- und Verarbeitungsfehlern ist. Die Garantiezeit beginnt ab dem Kaufdatum und beträgt ein Jahr. Während dieser Zeit verpflichtet sich Hewlett-Packard, etwaige fehlerhafte Teile kostenlos instandzusetzen oder auszutauschen, wenn der Rechner direkt oder über einen autorisierten Vertragshändler an Hewlett-Packard eingeschickt wird. Versandkosten bis zur Auslieferung bei einem Hewlett-Packard Service-Zentrum gehen zu Ihren Lasten, unabhängig davon, ob sich das Gerät noch in der Garantiezeit befindet oder nicht. Wenn Sie den Rechner verkaufen oder verschenken, so wird die Gewährleistung automatisch auf den neuen Eigentümer übertragen und bezieht sich weiterhin auf das ursprüngliche Kaufdatum.

Weitere Hinweise

Batterien sowie durch Batterien verursachte Schäden sind von der Gewährleistung durch Hewlett-Packard nicht erfaßt. Setzen Sie sich mit dem Hersteller der Batterien zwecks einer diesbezüglichen Gewährleistung in Verbindung.

Die von Hewlett-Packard angebotene Gewährleistung gilt nicht für Schäden, die durch unsachgemäße Betriebsweise entstanden sind. Der Ausschluß gilt ebenso, wenn Modifikationen oder Servicearbeiten durch nicht von Hewlett-Packard autorisierten Reparaturzentren durchgeführt wurden.

Es gibt keinen weiteren Gewährleistungsumfang. Die Einleitung der erforderlichen Reparatur- oder Ersatzleistungen ist ausschließlich dem Kunden überlassen. **Weitergehende Ansprüche, insbesondere auf Ersatz von Folgeschäden, können nicht geltend gemacht werden.** Dies gilt nicht, soweit gesetzlich zwingend gehaftet wird.

Änderungsverpflichtung

Geräte werden in der zum Zeitpunkt der Herstellung gültigen Version gebaut. Hewlett-Packard übernimmt keine Verpflichtung, früher verkaufte Geräte zu verändern oder an eine neue Version anzupassen.

Im Reparaturfall

Hewlett-Packard unterhält in den meisten Ländern der Welt Reparaturzentren. Diese Zentren reparieren oder ersetzen Ihren Rechner durch ein gleich- oder höherwertigeres Modell, unabhängig vom Garantiefall. Nach der Garantiezeit von einem Jahr werden Reparaturkosten berechnet. Der Service wird normalerweise innerhalb von 5 Arbeitstagen ausgeführt.

Service-Adressen

- **In Europa:** Sofern Sie sich in der BRD aufhalten, können Sie sich auf die Anschriften auf der Innenseite des Rückumschlags beziehen. Die Anschrift der europäischen Zentrale finden Sie nachstehend. *Nehmen Sie zuerst Kontakt mit Hewlett-Packard auf, bevor Sie Ihren Rechner zur Reparatur einschicken.*

  Hewlett-Packard S.A.
  150, route du Nant-d'Avril
  1217 Meyrin 2
  Schweiz
  Tel: (022) 82 81 11

- **In den USA:**

  Hewlett-Packard
  Calculator Service Center
  1030 N.E. Circle Blvd
  Corvallis, OR 97330, USA
  Tel: (503) 757 2002

- **In anderen Ländern:** Nehmen Sie Kontakt mit der nächstgelegenen Hewlett-Packard-Geschäftsstelle auf, um die korrekte Anschrift eines Reparaturzentrums zu erfahren.

  Alle Versand- und Reimportformalitäten sowie eventuell anfallende Zollaufwendungen unterliegen der Verantwortlichkeit des Kunden.

Reparaturkosten

Für Reparaturen nach der Garantiezeit wird eine Reparaturkostenpauschale erhoben. Diese schließt sämtliche Arbeits- und Materialkosten mit ein. In der BRD unterliegt die Pauschale der Mehrwertsteuer. Sämtliche Steuern werden auf der Rechnung getrennt ausgewiesen.

Die Reparaturkostenpauschale deckt nicht die Reparatur von Rechnern, welche durch Gewalteinwirkung oder Fehlbedienung zerstört wurden. In diesem Fall werden die Reparaturkosten individuell nach Arbeits- und Materialaufwand festgesetzt.

Versandanweisungen

Wenn Ihr Rechner repariert werden muß, senden Sie ihn bitte mit folgenden Unterlagen ein:

- Vollständige Absenderangabe und eine Beschreibung des Fehlers. Wenn der Verpackung Ihres Rechners eine Servicekarte beigelegt war, können Sie diese für die Angabe der entsprechenden Informationen verwenden.
- Rechnung oder anderer Kaufbeleg, wenn die einjährige Garantiezeit noch nicht abgelaufen ist.

Der Rechner und die erforderlichen Begleitinformationen sollten in der Originalverpackung oder einer adäquaten Schutzverpackung versandt werden, um Transportschäden zu vermeiden. Solche Transportschäden werden durch die einjährige Garantiezeit nicht abgedeckt; der Versand zum Reparaturzentrum erfolgt auf Ihre Gefahr, wobei Hewlett-Packard Ihnen zu einer Transportversicherung rät.

Gewährleistung bei Reparaturen

Für Reparaturen außerhalb der Garantiezeit leistet Hewlett-Packard eine Garantie von 90 Tagen ab Reparaturdatum bezüglich Material- und Bearbeitungsfehlern.

Servicevereinbarungen

Für Ihren Rechner gibt es eine Vereinbarung über Serviceunterstützung. Beziehen Sie sich auf die Dokumentation, welche der Versandpackung beigelegt ist. Für zusätzliche Informationen sollten Sie sich mit Ihrem HP Vertragshändler oder einer Hewlett-Packard-Geschäftsstelle in Verbindung setzen.

Sicherheitsbestimmungen

Funkschutz

Der HP-28S wurde zusammen mit dem zugehörigen Taschendrucker von Hewlett-Packard geprüft und entspricht den Bestimmungen der Allgemeinen Verfügung FTZ 1046/84.

Wird der Rechner zusammen mit Geräten betrieben, welche nicht von Hewlett-Packard hergestellt oder empfohlen sind, dann ist sicherzustellen, daß die gesamte Konfiguration der oben genannten Verfügung entspricht.

B

Menütabellen

In diesem Anhang sind alle Befehle der HP-28S-Menüs gelistet. Die Menüs sind in alphabetischer Reihenfolge aufgeführt, von ALGEBRA bis TRIG. Detaillierte Informationen über ein Menü erhalten Sie im Kapitel 3, "Schlüsselwortverzeichnis", des Referenzhandbuchs. Dort sind alle Menüs im einzelnen beschrieben. Wenn Sie nähere Informationen über einen bestimmten Befehl erhalten möchten, beziehen Sie sich auf das Verzeichnis der Operationen am Ende des Referenzhandbuchs. Sie finden dort einen Verweis auf eine Seite, auf welcher der Befehl ausführlich beschrieben ist.

Dieser Anhang enthält nicht die Menüs der interaktiven Operationen, welche durch CATALOG, FORM, den Löser und UNITS zur Verfügung gestellt werden.

- CATALOG ist in Kapitel 22 beschrieben und auf Seite 31 anhand eines Beispiels erläutert.
- FORM ist auf Seite 112 beschrieben. Details finden Sie unter "ALGEBRA (FORM)" im Referenzhandbuch.
- Der Löser wird in Kapitel 8 beschrieben. Details finden Sie unter "SOLVE" im Referenzhandbuch.
- UNITS ist unter "Der UNITS-Katalog" auf Seite 141 beschrieben. Details finden Sie unter "UNITS" im Referenzhandbuch.

Die Befehle innerhalb dieses Anhangs erscheinen als Gruppe, wie sie in einer Menüzeile angezeigt werden. Durch Drücken von NEXT erhalten Sie die nächste Zeile angezeigt, mit ■ PREV erhalten Sie die vorangehende Zeile.

Die Spalte "Befehl" enthält den Namen in der Anzeige. Unter "Beschreibung" finden Sie eine kurze Beschreibung des Befehls oder des vollständigen Namens. Die Seitenangabe verweist auf ein Beispiel oder eine Erwähnung des Befehls. Ist keine Seite angegeben, so beziehen Sie sich auf das Verzeichnis der Operationen im Referenzhandbuch.

ALGEBRA

	Befehl	Beschreibung	Seite
Zeile 1	COLCT	Terme zusammenfassen	111
	EXPAN	Ausdruck erweitern	111
	SIZE	Bestimmen der Dimension eines Objekts	
	FORM	Algebraischen Ausdruck formen	112
	OBSUB	Objekt substituieren	
	EXSUB	Ausdruck substituieren	
NEXT			
Zeile 2	TAYLR	Taylorreihe	
	ISOL	Isolieren	112
	QUAD	Quadratische Form	
	SHOW	Variable anzeigen	
	OBGET	Objekt aus Ausdruck holen	
	EXGET	Teilausdruck aus Ausdruck holen	

ARRAY

	Befehl	Beschreibung	Seite
Zeile 1	→ARRY	Zahlen in einem Feld zusammenfassen	275
	ARRY→	Feldelemente in separate Stack-Objekte übertragen	274
	PUT	Element in Feld oder Liste übernehmen	
	GET	Element aus Feld oder Liste holen	
	PUTI	Element übernehmen und Index erhöhen	
	GETI	Element holen und Index erhöhen	
NEXT			
Zeile 2	SIZE	Dimension bestimmen	274
	RDM	Redimensionieren	
	TRN	Matrix transponieren	264
	CON	Konstantenfeld	
	IDN	Einheitsmatrix erzeugen	
	RSD	Residuum	
NEXT			
Zeile 3	CROSS	Kreuzprodukt	126
	DOT	Skalarprodukt	126
	DET	Determinante	128
	ABS	Absolutbetrag	
	RNRM	Zeilensummennorm	
	CNRM	Spaltensummennorm	
NEXT			
Zeile 4	R→C	Reell-in-komplex	
	C→R	Komplex-in-reell	
	RE	Reeller Anteil	
	IM	Imaginärer Anteil	
	CONJ	Konjugieren	
	NEG	Negieren	

BINARY

	Befehl	Beschreibung	Seite
Zeile 1	DEC	Dezimales Zahlensystem	140
	HEX	Hexadezimales Zahlensystem	139
	OCT	Oktales Zahlensystem	140
	BIN	Binäres Zahlensystem	140
	STWS	Wortlänge speichern	139
	RCWS	Wortlänge zurückrufen	
NEXT			
Zeile 2	RL	Nach links rotieren	
	RR	Nach rechts rotieren	
	RLB	Linkes Byte rotieren	
	RRB	Rechtes Byte rotieren	
	R→B	Reell-in-Binärwert	261
	B→R	Binärwert-in-reell	
NEXT			
Zeile 3	SL	Verschiebung nach links	
	SR	Verschiebung nach rechts	
	SLB	Verschieben des linken Bytes	
	SRB	Verschieben des rechten Bytes	
	ASR	Arithmetische Verschiebung nach rechts	
NEXT			
Zeile 4	AND	Logisches UND	
	OR	Logisches ODER	
	XOR	Exklusives ODER	
	NOT	Logisches NICHT	

COMPLEX

	Befehl	Beschreibung	Seite
Zeile 1	R→C	Reell-in-komplex	83
	C→R	Komplex-in-reell	83
	RE	Reeller Anteil	83
	IM	Imaginärer Anteil	84
	CONJ	Konjugieren	84
	SIGN	Ausrichtung	84
NEXT			
Zeile 2	R→P	Rechtecks-in-Polarnotation	86
	P→R	Polar-in-Rechtecksnotation	85
	ABS	Absolutbetrag	85
	NEG	Negieren	85
	ARG	Argument	85

LIST

	Befehl	Beschreibung	Seite
Zeile 1	→LIST	Stack-in-Liste	181
	LIST→	Liste-in-Stack	181
	PUT	Element übernehmen	271
	GET	Element holen	237
	PUTI	Element übernehmen und Index erhöhen	271
	GETI	Element holen und Index erhöhen	271
NEXT			
Zeile 2	POS	Position	237
	SUB	Substring	276
	SIZE	Dimension bestimmen	271

LOGS

	Befehl	Beschreibung	Seite
Zeile 1	LOG	Dekadischer Logarithmus	78
	ALOG	Dekadischer Antilogarithmus	78
	LN	Natürlicher Logarithmus	78
	EXP	Exponentialfunktion	78
	LNP1	Natürlicher Logarithmus von 1 + x	78
	EXPM	Exponentialfunktion minus 1	78
NEXT			
Zeile 2	SINH	Sinus hyperbolicus	78
	ASINH	Arcussinus hyperbolicus	78
	COSH	Cosinus hyperbolicus	78
	ACOSH	Arcuscosinus hyperbolicus	78
	TANH	Tangens hyperbolicus	78
	ATANH	Arcustangens hyperbolicus	78

MEMORY

	Befehl	Beschreibung	Seite
Zeile 1	MEM	Verfügbarer Speicherbereich	188
	MENU	CUSTOM Menü erzeugen	195
	ORDER	Variablen in spezifizierter Folge anordnen	184
	PATH	Aktueller Pfad	67
	HOME	HOME Verzeichnis wählen	71
	CRDIR	Verzeichnis erzeugen	66
NEXT			
Zeile 2	VARS	Variablen in aktuellem Verzeichnis	184
	CLUSR	Löschen des aktuellen Verzeichnisses (Inhalt)	184

MODE

	Befehl	Beschreibung	Seite
Zeile 1	STD	Standard-Anzeigeformat	38
	FIX	Feste Anzahl Nachkommastellen	38
	SCI	Wissenschaftliches Anzeigeformat	38
	ENG	Technisches Anzeigeformat	38
	DEG	Winkelmodus Grad	74
	RAD	Winkelmodus Radiant	74
NEXT			
Zeile 2	CMD	COMMAND aktivieren oder desaktivieren	210
	UNDO	UNDO aktivieren oder desaktivieren	211
	LAST	LAST aktivieren oder desaktivieren	211
	ML	Mehrzeilen-Format aktivieren oder desaktivieren	208
	RDX,	Komma als Dezimalzeichen aktivieren oder desaktivieren	37
	PRMD	Modi drucken und anzeigen	

PLOT

	Befehl	Beschreibung	Seite
Zeile 1	STEQ	Gleichung speichern	90
	RCEQ	Gleichung zurückrufen	
	PMIN	Minimum abbilden	95
	PMAX	Maximum abbilden	95
	INDEP	Unabhängige Variable wählen	
	DRAW	Abbildung erstellen	90
NEXT			
Zeile 2	PPAR	Abbildungsparameter zurückrufen	90
	RES	Auflösung der Abbildung spezifizieren	
	AXES	Koordinatenursprung festlegen	
	CENTR	Zentrum einer Abbildung festlegen	94
	*W	Breite multiplizieren	
	*H	Höhe multiplizieren	93
NEXT			
Zeile 3	STOΣ	Statistikmatrix speichern	
	RCLΣ	Statistikmatrix zurückrufen	
	COLΣ	Spalte der Statistikmatrix festlegen	
	SCLΣ	Statistikmatrix skalieren	
	DRWΣ	Statistikmatrix abbilden	
NEXT			
Zeile 4	CLLCD	Anzeige löschen	
	DGTIZ	Abbildungspunkt digitalisieren	
	PIXEL	Pixel	
	DRAX	Achsen zeichnen	
	CLMF	Meldungsflag löschen	
	PRLCD	Anzeigeinhalt drucken	

PRINT

	Befehl	Beschreibung	Seite
Zeile 1	PR1	Ebene 1 drucken	151
	PRST	Stackinhalt drucken	152
	PRVAR	Variable drucken	152
	PRLCD	Anzeigeinhalt drucken	149
	CR	Druckkopf rechts positionieren	
	TRAC	Protokoll-Druckmodus aktivieren/desaktivien	150
NEXT			
Zeile 2	PRSTC	Stack drucken (kompakt)	
	PRUSR	Benutzervariable drucken	
	PRMD	Modi drucken	

PROGRAM BRANCH

	Befehl	Beschreibung	Seite
Zeile 1	IF	Anfang der IF Struktur	226
	IFERR	Anfang der IF ERROR Struktur	227
	THEN	Anfang der THEN Struktur	226
	ELSE	Anfang der ELSE Struktur	226
	END	Ende der Programmstruktur	226
NEXT			
Zeile 2	START	Anfang einer bestimmten Schleife	228
	FOR	Anfang einer bestimmten Schleife	229
	NEXT	Ende einer bestimmten Schleife	228
	STEP	Ende einer bestimmten Schleife	230
	IFT	If-Then Befehl	227
	IFTE	If-Then-Else Funktion	226
NEXT			
Zeile 3	DO	Definiert unbestimmte Schleife	231
	UNTI	Definiert unbestimmte Schleife	231
	END	Ende der Programmstruktur	231
	WHIL	Definiert unbestimmte Schleife	232
	REPEA	Definiert unbestimmte Schleife	232
	END	Ende der Programmstruktur	232

PROGRAM CONTROL

	Befehl	Beschreibung	Seite
Zeile 1	SST	Einzelschritt	250
	HALT	Programm unterbrechen	234
	ABORT	Programm abbrechen	
	KILL	Unterbrochenes Programm abbrechen	250
	WAIT	Programmausführung unterbrechen	234
	KEY	String für Tastencode zurückgeben	234
NEXT			
Zeile 2	BEEP	Akustiksignal ausgeben	234
	CLLCD	Anzeige löschen	234
	DISP	Anzeige	234
	CLMF	Meldungsflag löschen	234
	ERRN	Fehlernummer zurückgeben	
	ERRM	Fehlermeldung zurückgeben	

PROGRAM TEST

	Befehl	Beschreibung	Seite
Zeile 1	SF	Flag setzen	205
	CF	Flag löschen	205
	FS?	Flag gesetzt?	225
	FC?	Flag gelöscht?	
	FS?C	Flag gesetzt? Löschen	
	FC?C	Flag gelöscht? Löschen	
NEXT			
Zeile 2	AND	Logisches UND	
	OR	Logisches ODER	
	XOR	Exklusives ODER	
	NOT	Logisches NICHT	232
	SAME	Test auf Übereinstimmung	231
	==	Gleich	222
NEXT			
Zeile 3	STOF	Flags speichern	156
	RCLF	Flags zurückrufen	156
	TYPE	Eintippen	232

REAL

	Befehl	Beschreibung	Seite
Zeile 1	NEG	Negieren	78
	FACT	Fakultät (Gammafunktion)	78
	RAND	Zufallszahl	78
	RDZ	Anfangswert für Zufallsgenerator	78
	MAXR	Reelles Maximum	79
	MINR	Reelles Minimum	79
NEXT			
Zeile 2	ABS	Absolutbetrag	
	SIGN	Vorzeichen	
	MANT	Mantisse	
	XPON	Exponent	
NEXT			
Zeile 3	IP	Ganzzahliger Anteil	
	FP	Gebrochener Anteil	
	FLOOR	Nächste kleinere ganze Zahl	272
	CEIL	Nächste größere ganze Zahl	272
	RND	Runden	
NEXT			
Zeile 4	MAX	Maximum	
	MIN	Minimum	
	MOD	Modulo	
	%T	Prozent von Total	

SOLVE

	Befehl	Beschreibung	Seite
Zeile 1	STEQ	Gleichung speichern	64
	RCEQ	Gleichung zurückrufen	
	SOLVR	Menü für Löser-Variablen	102
	ISOL	Isolieren	110
	QUAD	Quadratische Form	108
	SHOW	Variable anzeigen	
NEXT			
Zeile 2	ROOT	Algorithmus zum Lösen einer Gleichung	

STACK

	Befehl	Beschreibung	Seite
Zeile 1	DUP	Duplizieren	178
	OVER	Objekt in Ebene 2 duplizieren	178
	DUP2	Zwei Objekte duplizieren	178
	DROP2	Zwei Objekte löschen	179
	ROT	Rotieren	178
	LIST→	Liste-in-Stack	181
NEXT			
Zeile 2	ROLLD	Stackinhalt nach unten verschieben	178
	PICK	Duplizieren des *n*-ten Objekts	178
	DUPN	*n* Objekte duplizieren	178
	DROPN	*n* Objekte löschen	179
	DEPTH	Objektanzahl im Stack ermitteln	181
	→LIST	Stack-in-Liste	181

STAT

	Befehl	Beschreibung	Seite
Zeile 1	Σ+	Datenpunkt der Statistikmatrix hinzufügen	132
	Σ-	Letzten Datenpunkt der Statistikmatrix löschen	133
	NΣ	Anzahl der Datenpunkte von Statistikmatrix	134
	CLΣ	Löschen der Statistikmatrix	132
	STOΣ	Speichern der Statistikmatrix	275
	RCLΣ	Zurückrufen der Statistikmatrix	264
NEXT			
Zeile 2	TOT	Total	
	MEAN	Mittelwert	134
	SDEV	Standardabweichung	135
	VAR	Varianz	135
	MAXΣ	Maximum in Statistikmatrix	
	MINΣ	Minimum in Statistikmatrix	
NEXT			
Zeile 3	COLΣ	Spalten der Statistikmatrix wählen	136
	CORR	Korrelation	136
	COV	Kovarianz	136
	LR	Lineare Regression	137
	PREDV	Vorhersagewert	137
NEXT			
Zeile 4	UTPC	Rechtsseitige Chi-Quadrat Verteilung	
	UTPF	Rechtsseitige F-Verteilung	
	UTPN	Rechtsseitige Normalverteilung	
	UTPT	Rechtsseitige t-Verteilung	
	COMB	Kombinationen	
	PERM	Permutationen	

STORE

	Befehl	Beschreibung	Seite
Zeile 1	STO+	Speicherarithmetik +	
	STO-	Speicherarithmetik −	
	STO*	Speicherarithmetik ×	
	STO/	Speicherarithmetik ÷	
	SNEG	Speicherarithmetik-Negation	
	SINV	Speicherarithmetik-Inversion	
NEXT			
Zeile 2	SCONJ	Speicherarithmetik-Konjugation	

STRING

	Befehl	Beschreibung	Seite
Zeile 1	→STR	Objekt-in-String	258
	STR→	String-in-Objekt	175
	CHR	Zeichen	156
	NUM	Rückgabe des Zeichencodes	156
	→LCD	String-in-Anzeige	157
	LCD→	Anzeige-in-String	157
NEXT			
Zeile 2	POS	Position	
	SUB	Substring	
	SIZE	Dimension des Strings	258
	DISP	Anzeigen	156

TRIG

	Befehl	Beschreibung	Seite
Zeile 1	SIN	Sinus	74
	ASIN	Arcussinus	74
	COS	Cosinus	74
	ACOS	Arcuscosinus	74
	TAN	Tangens	74
	ATAN	Arcustangens	74
NEXT			
Zeile 2	P→R	Polar-in-Rechtecksnotation	76
	R→P	Rechtecks-in-Polarnotation	76
	R→C	Reell-in-komplex	76
	C→R	Komplex-in-reell	76
	ARG	Argument	76
NEXT			
Zeile 3	→HMS	Dezimal in Stunden-Minuten-Sekunden	76
	HMS→	Stunden-Minuten-Sekunden in dezimal	76
	HMS+	Stunden-Minuten-Sekunden Addition	76
	HMS−	Stunden-Minuten-Sekunden Subtraktion	76
	D→R	Grad-in-Radiant	77
	R→D	Radiant-in-Grad	77

C

Hinweise zu Leistungsmerkmalen des HP-28S

Das dem HP-28S beigelegte Referenzhandbuch wurde zum Gebrauch mit dem HP-28C und HP-28S Taschenrechner entwickelt. Ihr HP-28S enthält jedoch neben einem größeren Speicherbereich etliche Erweiterungen gegenüber dem HP-28C, auf welche in diesem Anhang besonders eingegangen wird.

Speicherorganisation und -Verwaltung

Speicherbereich (MEMORY)

Dieses Menü enthält folgende Einträge:

MEM	MENU	ORDER	PATH	HOME	CRDIR
VARS	CLUSR				

Verzeichnis-Befehle

Die nachstehend beschriebenen Befehle lassen sich für Verzeichnisse anwenden. Alle Befehle, außer PURGE, erscheinen im MEMORY Menü. Jeder Befehl ist dabei als "modifiziert" oder als "neu" (im Vergleich zum HP-28C) klassifiziert.

CLUSR	**Clear User.** Löscht alle Variablen und leere Unterverzeichnisse im momentanen Verzeichnis. Sind in einem Unterverzeichnis noch Variablen enthalten, so machen Sie dieses Verzeichnis zum momentanen Verzeichnis, führen CLUSR aus, bestimmen danach das Oberverzeichnis zum momentanen Verzeichnis und führen nun erneut CLUSR aus. (Modifiziert).
CRDIR	**Create Directory.** Nimmt einen Namen vom Stack und erzeugt ein Verzeichnis mit diesem Namen. Das neue Verzeichnis ist ein Unterverzeichnis des momentanen Verzeichnisses. (Neu).
HOME	Bestimmt das HOME Verzeichnis zum momentanen Verzeichnis. (Neu).
ORDER	Ordnet die Variablen und Unterverzeichnisse im momentanen Verzeichnis. (Modifiziert).
PATH	Gibt eine Auflistung aller Verzeichnisnamen im momentanen Pfad aus. Der erste Name lautet immer HOME. (Neu).
PURGE	Löscht ein leeres Unterverzeichnis. Sind in einem Unterverzeichnis noch Variablen enthalten, so bestimmen Sie dieses Verzeichnis zum momentanen Verzeichnis, führen CLUSR aus, bestimmen danach das Oberverzeichnis zum momentanen Verzeichnis und führen nun PURGE aus. (Modifiziert).
VARS	Gibt eine Auflistung aller Variablen und Unterverzeichnisse, die im momentanen Verzeichnis enthalten sind, aus. (Neu)

CUSTOM Menüs

Der HP-28S enthält den Befehl MENU, um CUSTOM Menüs zu erzeugen oder um innerhalb eines Programms ein internes Menü aufzurufen. Weiterhin kann über die Taste [CUSTOM] das zuletzt verwendete CUSTOM Menü angezeigt werden.

Interne Menüs. Sie können ein internes Menü über ein Programm auswählen, indem Sie eine Menünummer (in Ebene 1) spezifizieren und anschließend MENU ausführen. Nachstehend sind alle Menünummern gelistet:

Nummer	Menü	Nummer	Menü
1	ARRAY	13	CONTROL
2	BINARY	14	BRANCH
3	COMPLEX	15	TEST
4	STRING	16	MODE
5	LIST	17	LOGS
6	REAL	18	PLOT
7	STACK	19	CUSTOM
8	STORE	20	Cursor
9	MEMORY	21	TRIG
10	ALGEBRA	22	SOLVE
11	STAT	23	USER
12	PRINT	24	Löser

CUSTOM Benutzermenüs. Sie können beides, Variablennamen und Befehle, in einem CUSTOM Benutzermenü anzeigen lassen. Beispielsweise wird durch

```
« A B C SIN COS TAN » MENU
```

ein CUSTOM Benutzermenü erzeugt, welches die Variablen A, B und C sowie die Befehle SIN, COS und TAN einschließt. Die Tasten A ,

B und C wirken wie USER Menütasten, während SIN, COS und TAN wie TRIG Menütasten wirken.

CUSTOM Eingabemenüs. Sie können Werte in Variablen speichern, indem Sie ein CUSTOM Eingabemenü erzeugen. So wird z.B. nach dem Ausführen von

```
« STO A B C » MENU
```

ein löser-ähnliches Eingabemenü für die Variablen A, B und C erzeugt. Das Drücken einer Menütaste speichert das Objekt aus Ebene 1 in der entsprechenden Variablen.

Letztes CUSTOM Menü. Durch Drücken von ■ CUSTOM wird das zuletzt erzeugte CUSTOM Menü angezeigt.

Grafikfähigkeiten

Der HP-28S enthält mehrere Befehle zum Arbeiten mit Abbildungen und zum Digitalisieren über ein Programm.

Grafik-Strings

Der Anzeigeinhalt kann durch einen 548-Zeichen-String dargestellt werden, wobei Intervalle mit je 137 Zeichen den Inhalt einer Zeile repräsentieren. Der numerische Wert jedes Zeichens — in 8 Bit Darstellung — codiert eine Spalte mit acht Pixel; das niederwertigste Bit repräsentiert den obersten Pixel der Spalte.

Die nachfolgenden Befehle, welche im STRING Menü erscheinen, dienen zur Konvertierung von Anzeigeinhalten und Grafik-Strings.

LCD→	Gibt einen Grafik-String zurück, welcher den Inhalt der Anzeige darstellt.
→LCD	Nimmt einen Grafik-String vom Stack und ersetzt den momentanen Anzeigeinhalt durch den im String codierten Inhalt.

Die logischen Befehle AND, OR, XOR und NOT akzeptieren Strings als Argumente. AND, OR und XOR verbinden zwei Strings der gleichen Länge und erzeugen einen String, welcher die binäre Kombination der ursprünglichen Strings — als binäre Sequenz betrachtet — darstellt. NOT gibt die binäre Negation des ursprünglichen Strings zurück.

Im Prinzip können Sie einen Grafik-String in der Befehlszeile edieren. Allerdings enthalten die meisten Strings ein 0-wertiges Zeichen (zur Darstellung einer Spalte ohne Pixel), welches nicht in die Befehlszeile zurückgegeben werden kann.

Digitalisieren

Die Digitalisierungsfähigkeit von DRAW und DRWΣ des HP-28C ist im HP-28S in dem Befehl DGTIZ enthalten. Nach der Ausführung von DGTIZ werden die PLOT Menütasten aktiviert und es wird das Fadenkreuz angezeigt. Außerdem:

- Die zweite Menütaste ([DEL]) wird redefiniert, um LCD→ auszuführen.
- Wird das Fadenkreuz angezeigt, so bewirkt das Niederhalten der Cursor-Menütaste die Anzeige der Cursor-Koordinaten in Ebene 4.

Listen

Die LIST Fähigkeiten des HP-28S schließen Verkettung, automatische Auswertung und das Auffinden eines Listenelements mit ein.

Verkettung

Stellt eines der Objekte in Ebene 1 oder 2 eine Liste dar und ist das andere Objekt keine Liste, so wird nach Ausführen von + das zweite Objekt in eine Liste konvertiert und die beiden Listen werden verkettet.

Automatische Auswertung

Ist als Argument für einen Befehl eine Liste mit einem oder mehreren Zahlenwerten erforderlich, so können die Zahlen in der Liste durch Objekte ersetzt werden, die in numerische Werte ausgewertet werden. Dies gilt für Listen mit einem oder zwei Elementen, die als Felder und Listenindizes oder deren letzten zwei Elemente als Argument für $\int$ verwendet werden.

Position

Ist eine Liste (in Ebene 2) und ein Objekt (in Ebene 1) gegeben, so wird nach der Ausführung von POS die Position des Objekts innerhalb der Liste zurückgegeben (0, falls das Objekt nicht gefunden wurde).

Kombinationen und Permutationen

Die Befehle COMB und PERM sind zum STAT Menü hinzugefügt worden. Jeder Befehl erfordert zwei reellwertige Argumente (n in Ebene 2, m in Ebene 1) und gibt ein reellwertiges Ergebnis zurück.

COMB	Gibt die Anzahl der möglichen Kombinationen von m Elementen, die aus einer Grundgesamtheit von n Elementen gewählt werden, zurück: $n!/m!(n - m)!$
PERM	Gibt die Anzahl der möglichen Permutationen zurück: $n!/(n - m)!$

Drucken

Der HP-28S enthält eine Systemoperation gleichwertig zu PRLCD und eine Option für das Drucken mit doppeltem Zeilenabstand.

Drucken des Anzeigeinhalts

Um den Inhalt der Anzeige ohne dessen Veränderung auszudrucken:

1. Halten Sie [ON] gedrückt.
2. Drücken Sie [L] (Taste, oberhalb welcher PRINT aufgedruckt ist) und geben Sie die Taste wieder frei.
3. Geben Sie [ON] wieder frei.

Drucken mit doppeltem Zeilenabstand

Das Setzen von Flag 47 (durch Ausführen von 47 SF) bewirkt das Drucken mit doppeltem Zeilenabstand. Durch Löschen von Flag 47 (47 CF) führt wieder zur normalen Druckweise.

Dimensionslose Winkeleinheiten

Raum- und Flächenwinkel werden als *dimensionslos* bezeichnet, da sie keine physikalische Dimension beinhalten. Sie können die folgenden *dimensionslosen Einheiten* als Konstanten in einem Einheiten-String verwenden; der Rechner kann jedoch dimensionslose Einheiten nicht auf deren dimensionsmäßige Konsistenz überprüfen.

Dimensionslose Einheit	Abkürzung	Wert
Arcmin	arcmin	1/21600 Einheitskreis
Arcsec	arcs	1/1296000 Einheitskreis
Degree	°	1/360 Einheitskreis
Grade	grad	1/400 Einheitskreis
Radian	r	$1/2\pi$ Einheitskreis
Steradian	sr	$1/4\pi$ Einheitskugel

Einige photometrische Einheiten sind in Steradiant definiert. Diese Einheiten beinhalten den Faktor $1/4\pi$ in ihren numerischen Werten. Da

dieser Faktor dimensionslos ist, kann der Rechner nicht das Vorhandensein bzw. das Fehlen dieses Faktors überprüfen. Sie sollten deshalb bei der Umrechnung zwischen photometrischen Einheiten, welche diesen Faktor enthalten, und Einheiten, bei welchen der Faktor fehlt, die dimensionslose Einheit "sr" mit einschließen. Die nachstehende Tabelle listet photometrische Einheiten in Abhängigkeit davon, ob ihre Definition Steradiant mit einschließt.

Photometrische Einheiten

Enthält Steradiant	Ohne Steradiant
Lumen (lm)	Candela (cd)
Lux (lx)	Footlambert (flam)
Phot (ph)	Lambert (lam)
Footcandle (fc)	Stilb (sb)

Um zwischen photometrischen Einheiten der gleichen Spalte umzurechnen, ist kein "sr" Faktor notwendig. Um jedoch zwischen Einheiten in unterschiedlichen Spalten umzurechnen, ist entweder die linke Einheit durch "sr" zu dividieren oder die rechte Einheit mit "sr" zu dividieren. *Vergessen Sie dies nicht, denn der Rechner kann nicht überprüfen, ob Ihre Dimensionen konsistent sind.* Einige Beispiele für konsistente photometrische Einheiten:

"lm"	ist konsistent mit	"cd*sr"
"fc/sr"	ist konsistent mit	"flam"
"lm/sr*m^2"	ist konsistent mit	"lam"

Zusätzliche Fehlermeldungen

Alphabetisch gelistete Meldungen

Meldung	Fehlernummer Hex.	Dez.	Bedeutung
Directory Not Allowed	12A	299	Der Name eines existierenden Verzeichnisses wurde als Argument verwendet.
Non-Empty Directory	12B	300	Es wurde versucht, ein nicht leeres Verzeichnis zu löschen.
Power Lost	6	6	Eine zu niedere Batteriespannung kann zum Verlust der gespeicherten Daten geführt haben.

Nach Fehlernummer gelistete Meldungen

Hex.	Dez.	Meldung
Von allgemeinen Operationen resultierende Fehler		
006	006	Power Lost
12A	299	Directory Not Allowed
12B	300	Non-Empty Directory

Tastenverzeichnis

Dieses Verzeichnis beschreibt die Wirkungsweise der Tasten auf dem linken und rechten Tastenfeld des Rechners. Zuerst ist ein alphabetisches Verzeichnis der Tasten auf dem linken Tastenfeld aufgeführt, dem ein Verzeichnis für das rechte Tastenfeld folgt. Am Schluß finden Sie ein Verzeichnis für die Tasten des Cursor-Menüs (die weißen Bezeichnungen über den Menütasten).

Dieses Verzeichnis schließt umgeschaltete Tasten wie z.B. ■ [ARRAY] und ■ [OFF] mit ein. Es enthält nicht die Alpha-Tasten, wie [A] bis [Z] und [0] bis [9], welche immer das jeweilige Zeichen in die Befehlszeile schreiben. (Andere Alpha-Tasten schließen Begrenzungszeichen wie [(], Operatoren wie z.B. [=] und symbolische Konstanten wie ■ [π] mit ein. Diese Zeichen haben eine spezielle Bedeutung für den Rechner, aber ihre Tasten sind einfache Alpha-Tasten.) Wenn Sie eine Taste finden, welche in diesem Verzeichnis nicht gelistet ist, so handelt es sich um eine Alpha-Taste.

Für jede Taste gibt es eine kurze Beschreibung ihrer Wirkungsweise und einen Seitenverweis. Ist die Taste in diesem Handbuch nicht erwähnt, dann beziehen Sie sich für Informationen über eine solche Taste (oder eine beliebige andere Taste) auf das Verzeichnis der Operationen am Ende des Referenzhandbuchs.

Linkes Tastenfeld

Taste	Beschreibung	Seite
■ ALGBRA	Wählt das ALGEBRA Menü.	110
■ ARRAY	Wählt das ARRAY Menü.	124
■ BINARY	Wählt das BINARY Menü.	138
■ BRANCH	Wählt das PROGRAM BRANCH Menü.	222
■ CATALOG	Ruft den Befehlskatalog auf.	196
■ COMPLX	Wählt das COMPLEX Menü.	83
■ CONTRL	Wählt das PROGRAM CONTROL Menü.	234
LC	Schaltet zwischen Groß- und Kleinschreibung um.	168
■ LIST	Wählt das LIST Menü.	102
■ MENUS	Schaltet Menüsperre ein und aus.	192
■ MEMORY	Wählt das MEMORY Menü.	182
■ PRINT	Wählt das PRINT Menü.	149
■ REAL	Wählt das REAL Menü.	78
■ STACK	Wählt das STACK Menü.	176
■ STAT	Wählt das STAT Menü.	131
■ STORE	Wählt das STORE Menü.	191
■ STRING	Wählt das STRING Menü.	156
■ TEST	Wählt das PROGRAM TEST Menü.	225
■ UNITS	Ruft den UNITS-Katalog auf.	141
α	Wechselt den Eingabemodus.	171

Rechtes Tastenfeld

Taste	Beschreibung	Seite
ATTN (ON)	Bricht die Programmausführung ab; löscht die Befehlszeile; beendet die Anzeige für Kataloge, FORM und Abbildungen.	216
CHS	Ändert das Vorzeichen einer Zahl in der Befehlszeile oder führt den Befehl NEG aus.	168
■ CLEAR	Löscht den Stackinhalt.	179
■ COMMAND	Schiebt einen Eintrag vom Befehlsstack in die Befehlszeile.	174
■ CONT	Setzt ein unterbrochenes Programm fort.	235
■ CONVERT	Führt eine Einheitenkonvertierung durch.	143
■ CUSTOM	Wählt das zuletzt angezeigte CUSTOM Menü.	192
■ d/dx	Differentiation.	117
DROP	Löscht ein Objekt im Stack.	179
■ EDIT	Kopiert ein Objekt aus Ebene 1 zum Edieren in die Befehlszeile.	173
EEX	Eingabe des Exponenten in Befehlszeile.	168
ENTER	Analysiert und wertet die Befehlszeile aus.	173
EVAL	Wertet ein Objekt aus.	118
■ LAST	Gibt die zuletzt verwendeten Argumente zurück.	179
■ LOGS	Wählt das LOGS Menü.	77
■ MODE	Wählt das MODE Menü.	36
NEXT	Zeigt die nächste Zeile von Menüfeldern an.	192
ON (ATTN)	Schaltet den Rechner ein; bricht Programme ab; löscht den Inhalt der Befehlszeile; beendet die Anzeige von Katalogen, FORM und Abbildungen.	216
■ OFF	Schaltet den Rechner aus.	20
■ PLOT	Wählt das PLOT Menü.	89
■ PREV	Zeigt die vorangehende Zeile von Menüfeldern an.	192

Taste	Beschreibung	Seite
■ PURGE	Löscht eine oder mehrere Variablen.	183
■ RCL	Ruft den Inhalt einer Variablen (unausgewertet) zurück.	183
■ ROLL	Bewegt das Objekt in Ebene $n+1$ in Ebene 1.	178
SOLV	Wählt das SOLVE Menü.	99
STO	Speichert ein Objekt in einer Variablen.	183
■ SWAP	Tauscht den Inhalt der Objekte in Ebene 1 und 2 aus.	178
TRIG	Wählt das TRIG Menü.	74
■ UNDO	Ersetzt den Inhalt des Stacks.	180
USER	Wählt das USER Menü.	49
■ VIEW↑	Bewegt das Anzeigefenster eine Zeile nach oben.	177
■ VIEW↓	Bewegt das Anzeigefenster eine Zeile nach unten.	177
■ VISIT	Kopiert ein Objekt zum Edieren in die Befehlszeile.	173
■ x^2	Quadriert eine Matrix oder eine Zahl.	40
■ 1/x	Kehrwert.	40
+	Addiert zwei Objekte.	41
−	Subtrahiert zwei Objekte.	41
×	Multipliziert zwei Objekte.	41
÷	Dividiert zwei Objekte.	42
■ %	Prozent.	43
■ %CH	Änderung in Prozent.	43
■ ^	Potenziert eine Zahl.	42
■ $\sqrt{x}$	Berechnet die Quadratwurzel.	40
■ $\int$	Bestimmtes oder unbestimmtes Integral.	120
■	Umschalttaste.	29
◀✥▶	Wählt das Cursor-Menü oder ruft das letzte Menü zurück.	168
◀	Rückschritt-Taste.	168
■ →NUM	Erzeugt ein numerisches Ergebnis.	75

Das Cursor-Menü

Sie finden die Bezeichnungen für das Cursor-Menü in weißer Beschriftung direkt über den Menütasten (oberste Tastenreihe auf dem rechten Tastenfeld). Das Cursor-Menü ist aktiv, wenn die Befehlszeile angezeigt wird und keine Menüfelder in der Anzeige erscheinen. Um bei angezeigten Menüfeldern das Cursor-Menü aufzurufen, drücken Sie [◆▶]. Das zuvor angezeigte Menü kann wieder aufgerufen werden, indem Sie erneut [◆▶] drücken.

Taste	Beschreibung	Seite
[INS]	Schaltet zwischen Einfügungs- und Ersetzungsmodus um.	167
■[INS]	Löscht alle Zeichen links des Cursors.	168
[DEL]	Löscht das Zeichen an der Cursorposition.	167
■[DEL]	Löscht das Zeichen an der Cursorposition und alle Zeichen rechts davon.	168
[▲]	Bewegt den Cursor nach oben.	167
■[▲]	Bewegt den Cursor so weit als möglich nach oben.	168
[▼]	Bewegt den Cursor nach unten.	167
■[▼]	Bewegt den Cursor so weit als möglich nach unten.	168
[◀]	Bewegt den Cursor nach links.	167
■[◀]	Bewegt den Cursor so weit als möglich nach links.	168
[▶]	Bewegt den Cursor nach rechts.	167
■[▶]	Bewegt den Cursor soweit als möglich nach rechts.	168

Index

Fettgedruckte Seitenangaben kennzeichnen einen primären Eintrag, Seitenangaben in normaler Druckweise kennzeichnen einen sekundären Eintrag.

H

I

J, K

L

M

N

O

P

Q

R

S

T

U

V

W

Z

Unterstützung von Hewlett-Packard

Bezüglich Antworten auf die Anwendungsweise des Rechners: Wenn Sie Fragen zur Anwendung des Rechners haben, sollten Sie sich zuerst auf das Inhaltsverzeichnis, den Sachindex und den Abschnitt "Antworten auf allgemeine Fragen" in Anhang A beziehen. Sollten Sie in diesem Handbuch keine ausreichende Auskunft für Ihre Problemstellung finden, so können Sie sich über die nachstehende Adresse mit Hewlett-Packard in Verbindung setzen:

Hewlett-Packard GmbH
Support Zentrum Ratingen
Berliner Straße 111
D-4030 Ratingen

Telefon: (02102) 494-500

Im Fall einer erforderlichen Reparatur: Falls die Hinweise in Anhang A auf eine notwendige Reparatur hindeuten, dann können Sie den Rechner an das nachstehende Reparaturzentrum schicken:

Hewlett-Packard GmbH
Reparaturzentrum Frankfurt
Berner Straße 117
D-6000 Frankfurt 56

Telefon: (069) 500001-0

Informationen über Hewlett-Packard Fachhändler, Produkte und Preise: Setzen Sie sich diesbezüglich mit der Hewlett-Packard Vertriebszentrale in Verbindung:

Hewlett-Packard Vertriebszentrale
Hewlett-Packard-Straße
D-6380 Bad Homburg

Telefon: (06172) 400-0

Inhaltsverzeichnis

HEWLETT PACKARD

Bestellnummer
00028-90072

00028-90140 German
Printed in West Germany 9/88