

Owner's Handbook and Programming Guide



"The success and prosperity of our company will be assured only if we offer our customers superior products that fill real needs and provide lasting value, and that are supported by a wide variety of useful services, both before and after sale."

*Statement of Corporate Objectives.
Hewlett-Packard*

When Messrs. Hewlett and Packard founded our company in 1939, we offered one superior product, an audio oscillator. Today, we offer more than 3,000 quality products, designed and built for some of the world's most discerning customers.

Since we introduced our first scientific calculator in 1967, we've sold over a million world-wide, both pocket and desktop models. Their owners include Nobel laureates, astronauts, mountain climbers, businessmen, doctors, students, and housewives.

Each of our calculators is precision crafted and designed to solve the problems its owner can expect to encounter throughout a working lifetime.

HP calculators fill real needs. And they provide lasting value.



The HP-95C
Scientific Printing Calculator
with Partition Programming
Owner's Handbook
and
Programming Guide

January 1977

00095-90001

Contents

HP-95C Function and Key Index	7
The HP-95C Scientific Printing Calculator with Partition Programming	8
Function Key Index	9
Programming Key Index	10
Meet the HP-95C	13
Manual Problem Solving	14
Programmed Problem Solving	15
What Continuous Memory Means to You	16
Using This Handbook	17
Part One: Using Your HP-95C Calculator	19
Section 1: Getting Started	21
Display	21
Keyboard	21
Keying In Numbers	22
Negative Numbers	22
Clearing	22
Printer	22
Functions	23
One-Number Functions	24
Two-Number Functions	25
Chain Calculations	27
A Word About the HP-95C	31
Section 2: Printer and Display Control	33
Display Control Keys	33
Fixed Point Display	34
Scientific Notation Display	35
Engineering Notation Display	35
Format of Printed Numbers	37
Automatic Display Switching	39
Keying In Exponents of Ten	40
Calculator Overflow	42
Error Display	42
Low Power Indicator	43
Section 3: The Automatic Memory Stack	45
The Stack	45
Display	45
Manipulating Stack Contents	46
Reviewing the Stack	46
Exchanging x and y	47
Clearing the Stack	48
The ENTER Key	49
One-Number Functions and the Stack	50
Two-Number Functions and the Stack	51
Chain Arithmetic	52
Order of Execution	56
LAST X	57
Recovering From Mistakes	57
Recovering a Number for Calculation	58
Constant Arithmetic	58

Section 4: Storing and Recalling Numbers	61
Numbered Storage Registers	62
Storing Numbers	62
Recalling Numbers	62
The 1 Register	63
Printing the Storage Registers	64
Clearing Storage Registers	64
Storage Register Arithmetic	65
Storage Register Overflow	66

Section 5: Function Keys	69
Number Alteration Keys	69
Absolute Value	69
Integer Portion of a Number	69
Fractional Portion of a Number	70
Reciprocals	70
Factorials	71
Square Roots	71
Squaring	72
Using Pi	72
Percentages	73
Percent of Change	74
Trigonometric Functions	74
Trigonometric Modes	74
Functions	75
Hours, Minutes, Seconds/Decimal Hours Conversions	76
Adding and Subtracting Time and Angles	77
Polar/Rectangular Coordinate Conversions	80
Logarithmic and Exponential Functions	84
Logarithms	84
Raising Numbers to Powers	85
Statistical Functions	87
Accumulations	87
Printing Accumulations	89
Percent of Sum	90
Mean	91
Standard Deviation	93
Deleting and Correcting Data	95
Linear Regression	96
Linear Estimate	98
Coefficient of Determination	99
Vector Arithmetic	99

Part Two: Programming the HP-95C **103**

Section 6: Simple Programming	105
What Is a Program?	105
Printing a Program	106
Program Memory	106
Keycodes	108
Clearing a Program	109
Creating a Program	110
The Beginning of a Program	111
Loading a Program	111
Running a Program	112
Going to a Program Marker	112
Executing Instructions	113
Flowcharts	114
Problems	117

Section 7: The Printer and the Program	119
Printer Operation During a Running Program	119
Using the Printer for Creating Programs	119
Program Load Verification	121
Program Listing	122
Printing a Space	122
Problems	123
Section 8: Program Editing	125
Program Editing and Manipulation Functions	125
Nonrecordable Operations	125
Using GTO From the Keyboard	126
Pythagorean Theorem Program	127
Initializing a Program	128
Running the Program	128
Single-Step Execution of a Program	128
Modifying a Program	130
Single-Step Viewing Without Execution	131
Stepping Backward Through a Program	131
Verifying Program Changes	133
Running the Modified Program	134
Deleting an Instruction	134
Using the Printer for Editing	136
Problems	137
Section 9: Branching	141
Unconditional Branching and Looping	141
Using Labels	141
Transferring Execution Back to the Beginning of a Program	144
Conditionals and Conditional Branches	144
Problems	150
Section 10: Program Interruptions	157
Using R/S	157
Pausing to View Output	159
Keyboard Stops	161
Error Stops	162
Problems	162
Section 11: Subroutines	165
Routine-Subroutine Usage	169
Subroutine Limits	171
Selecting Subroutines From the Keyboard	171
Problems	172
Section 12: The I-Register and Indirect Control	175
Storing a Number in I	175
Incrementing and Decrementing the I-Register	175
Indirect Store and Recall	179
Problems	182
Section 13: Transferring Execution Between Programs	187
Branching to Another Program	187
Branching to a Label Within Another Program	187
Using a Program as a Subroutine	191
Calling a Subroutine From Another Program	192
Problem	192

Appendix A: Accessories, Service, and Maintenance	195
Your Hewlett-Packard Calculator	195
Standard Accessories	196
Optional Accessories	196
AC Line Operation	197
Battery Charging	198
Battery Operation	198
Using Continuous Memory	199
Battery Pack Replacement	199
Battery Care	200
Your HP-95C Printer	201
Paper for Your HP-95C	201
Replacing Paper	202
Printer Maintenance	203
Service	203
Low Power	203
Blank Display	204
Temperature Range	204
Warranty	204
Full One-Year Warranty	204
Out-of-Warranty	204
Warranty Transfer	204
Repair Policy	205
Repair Time	205
Shipping Instructions	205
Further Information	205
Appendix B: Error Displays	207
Appendix C: Stack Lift and LAST X	209

HP-95C
Function and Key Index

The HP-95C Scientific Printing Calculator with Partition Programming

Automatic Memory Stack

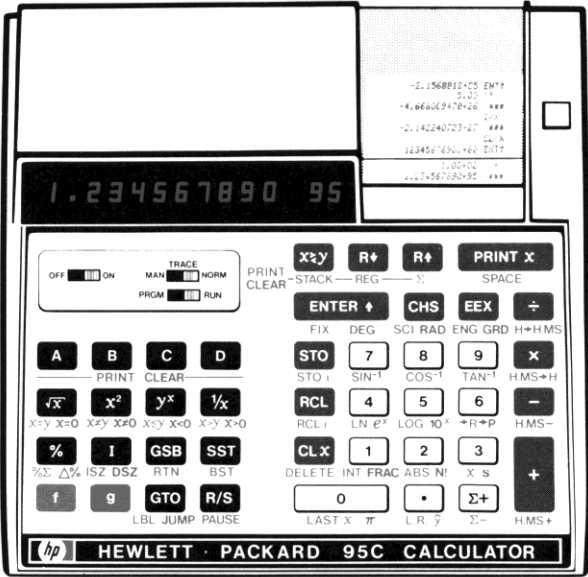
Registers

T	0.00000000 00
Z	0.00000000 00
Y	0.00000000 00

Displayed X

LAST X

0.00000000 00



Storage Registers

Manual Storage

R ₀	0.00000000 00
R ₁	0.00000000 00
R ₂	0.00000000 00
R ₃	0.00000000 00
R ₄	0.00000000 00
R ₅	0.00000000 00
R ₆	0.00000000 00
R ₇	0.00000000 00
R ₈	0.00000000 00
R ₉	0.00000000 00

I 0.00000000 00

Accumulation (Σ +) or Manual Storage

R ₀	0.00000000 00
R ₁	0.00000000 00
R ₂	0.00000000 00
R ₃	0.00000000 00
R ₄	0.00000000 00
R ₅	0.00000000 00

Program Memory

Program Partition Markers

A-000	lbl A
b-000	lbl b
C-000	lbl C
d-000	lbl d

200 Steps of Program Memory

Programming Key Index



PROGRAM Mode

PRGM-RUN switch set to
PRGM  RUN

All function keys except the ones below are loaded into program memory when pressed.

Active keys:

In PROGRAM mode only six operations are active. These operations are used to help record programs, and cannot themselves be recorded in program memory.

GTO Go to. Followed by  and **A**, **B**, **C** or **D**, sets calculator to top of designated program. Followed by  and number key (**0** through **9**), sets calculator to next designated label (if present) within current program (**page 126**).

f **PRINT** **A**, **B**, **C**, or **D**
Prints contents of designated program. Begins with current step if designated program is current program (**page 125**).

Automatic RUN Mode

PRGM-RUN switch set to RUN PRGM  RUN

Function keys may be executed as part of a recorded program or individually by pressing from the keyboard. Input numbers and answers are displayed by the calculator, except where indicated.

Pressed from the keyboard:

A **B** **C** **D** User-definable keys. Cause calculator to go directly to top of designated program and begin execution there (**page 113**).

GTO Go to. Followed by **A**, **B**, **C**, or **D**, sets calculator to top of designated program. Followed by number key (**0** through **9**), sets calculator to next designated label (if present) within current program. No instructions are executed (**page 126**).

GSB Go to subroutine. Followed by **A**, **B**, **C**, or **D**, causes calculator to go directly to top of designated label and begin execution there. Followed by number key (**0** through **9**), causes calculator to begin execution with next designated label (if present) within current program. Execution is as a subroutine (**page 171**).

f **PRINT** **A**, **B**, **C**, or **D**
Prints contents of designated program. Begins with current step if designated program is current program (**page 125**).

Executed as a recorded program instruction:

A **B** **C** **D** Transfer execution to designated program marker (top of designated program), where execution continues as a subroutine (**page 191**).

GTO Go to. Followed by **A**, **B**, **C**, or **D**, transfers execution to top of designated program, where execution resumes. Followed by number key (**0** through **9**), transfers execution to next designated label (if present) within current program, where execution resumes (**page 141**).

GSB Go to subroutine. Followed by number key (**0** through **9**), causes calculator to search downward through current program for next designated label and resume execution there. Execution is as a subroutine (**page 165**).

Mathematics

[\sqrt{x}] Computes square root of number in displayed X-register (page 71).

[x^2] Computes square of number in displayed X-register (page 72).

[$1/x$] Computes reciprocal of number in displayed X-register (page 70).

[π] Places value of pi (3.141592654) into displayed X-register (page 72).

[N!] Computes factorial of number in displayed X-register (page 71).

[+ - \times \div] Arithmetic operators (page 25).

Trigonometry

[DEG] Sets decimal degrees mode for trigonometric functions (page 74).

[RAD] Sets radians mode for trigonometric functions (page 74).

[GRD] Sets grads mode for trigonometric functions (page 74).

[SIN] [COS] [TAN] Compute sine, cosine, or tangent of value in displayed X-register (page 75).

[SIN⁻¹] [COS⁻¹] [TAN⁻¹] Compute arc sine, arc cosine, or arc tangent of number in displayed X-register (page 75).

[HMS+] Adds hours, minutes, seconds, or degrees, minutes, seconds in Y-register to those in X-register (page 77).

[HMS-] Subtracts hours, minutes, seconds or degrees, minutes, seconds in displayed X-register from those in Y-register (page 77).

[H \leftrightarrow HMS] Converts decimal hours or degrees to hours, minutes, seconds or degrees, minutes, seconds (page 76).

[HMS \leftrightarrow H] Converts hours, minutes, seconds or degrees, minutes, seconds to decimal hours or degrees (page 77).

Polar/Rectangular Conversion

[\rightarrow P] Converts x, y rectangular coordinates placed in X- and Y-registers to polar magnitude r and angle θ (page 80).

[\rightarrow R] Converts polar magnitude r and angle θ in X- and Y-registers to rectangular x and y coordinates (page 81).

Logarithmic and Exponential

[y^x] Raises number in Y-register to power of number in displayed X-register (page 85).

[10^x] Common antilogarithm. Raises 10 to power of number in displayed X-register (page 84).

[e^x] Natural antilogarithm. Raises e (2.718281828) to power of number in displayed X-register (page 84).

[LOG] Computes common logarithm (base 10) of number in displayed X-register (page 84).

[LN] Computes natural logarithm (base e , 2.718281828) of number in displayed X-register (page 84).

Statistics

[Σ] Accumulates numbers from X- and Y-registers into storage registers $R_{0.0}$ through $R_{9.9}$ (page 87).

[$\Sigma-$] Subtracts x and y values from storage registers $R_{0.0}$ through $R_{9.9}$ for correcting **[Σ]** accumulations (page 95).

CLEAR [C] Clears storage registers used for accumulations ($R_{0.0}$ through $R_{9.9}$) to zero (page 87).

[\bar{x}] Computes mean (average) of x and y values accumulated by **[Σ]** (page 91).

[S] Computes sample standard deviations of x and y values accumulated by **[Σ]** (page 93).

[L.R.] Linear regression. Computes y-intercept (A) and slope (B) for x and y data points accumulated using **[Σ]** (page 96).

[\hat{y}] Linear estimate. With set of x, y data points accumulated using **[Σ]**, computes estimated y for new x (page 98).

Percentage

[$\%$] Computes x% of y (page 73).

[$\Delta\%$] Computes percent of change from number in Y-register to number in displayed X-register (page 74).

[$\%$ Σ] Computes percent that x is of the number (Σx) in storage register $R_{0.0}$ (page 90).

Indirect Control

[I] Recalls number from I-register into displayed X-register. (To store number in I, use **[STO I]**.) (page 63).


[ISZ] Increment I, skip if zero. Adds 1 to contents of I. Skips one step if contents are then zero (page 175).

[DSZ] Decrement I, skip if zero. Subtracts 1 from contents of I. Skips one step if contents are then zero (page 175).


[STOI] Stores contents of X-register in storage register addressed by current number in I-register (page 179).

[RCL I] Recalls contents of storage register addressed by current number in I-register (page 179).


Function Key Index


Manual RUN Mode. PRGM-RUN switch PRGM  RUN set to RUN.


Function keys pressed from the keyboard execute individual functions as they are pressed. Input numbers and answers are displayed. All function keys listed below operate either from the keyboard or as recorded instructions in a program.

 Paper advance pushbutton. Press to advance paper without printing (page 23).


OFF  ON Power switch (page 21).

TRACE
MAN  NORM Print mode switch. Selects printing option (page 22).


 Pressed before function key, selects gold function printed below key (page 21).


 Pressed before function key, selects blue function printed below key (page 21).


Printing Functions

 Advances paper one space without printing (page 122).


PRINT  Prints contents of all numbered storage registers (page 64).


PRINT  Prints contents of automatic memory stack (page 45).


PRINT  Prints contents of displayed X-register (page 23).



PRINT  Causes printer to list contents of accumulation registers (storage registers R₀ through R₉) (page 89).


Digit Entry

ENTER  Enters a copy of number displayed in X-register into Y-register. Used to separate numbers (page 25).


 Changes sign of mantissa or exponent of 10 in displayed X-register (page 22).


 Enter exponent. After pressing, next numbers keyed in are exponents of 10 (page 40).


 through  Digit keys (page 22).


 Decimal point (page 22).


Number Manipulation

 Rolls up contents of stack for viewing in displayed X-register (page 47).


 Rolls down contents of stack for viewing in displayed X-register (page 46).


 Exchanges contents of X- and Y-registers of stack (page 47).


 Clears contents of displayed X-register to zero (page 22).

CLEAR  Clears contents of stack (X, Y, Z, and T) to zero (page 48).


Display Control


 Fixed point display. Followed by a number key, selects fixed point notation display (page 34).


 Scientific display. Followed by a number key, selects scientific notation display (page 35).

 Engineering display. Followed by a number key, selects engineering notation display (page 35).



Number Alteration


 Gives absolute value of number in displayed X-register (page 69).


 Leaves only integer portion of number in displayed X-register by truncating fractional portion (page 69).


 Leaves only fractional portion of number in displayed X-register by truncating integer portion (page 70).

Manual Storage

 Store. Followed by number key, decimal point and number key, or  stores displayed number in storage register specified (R₀ through R₉, R₋₀ through R₋₉, I). Also used to perform storage register arithmetic (page 62).

 Recall. Followed by number key, or decimal point and number key, recalls value from storage register specified (R₀ through R₉; R₋₀ through R₋₉) into the displayed X-register (page 62).

CLEAR  Clears contents of storage registers R₀ through R₉, and R₋₀ through R₋₉ (page 64).

 Recalls number displayed before the previous operation back into the displayed X-register (page 57).

PROGRAM Mode

Active keys:

9 CLEAR **A**, **B**, **C**, or **D** Clears all instructions from designated program and sets calculator to that program marker (**page 109**).

BST Back step. Moves calculator back one step of program memory in current program (**page 131**).

SST Single step. Moves calculator forward one step of program memory in current program (**page 125**).

Automatic RUN Mode

Pressed from the keyboard:

BST Back step. Sets calculator to and displays step number and keycode of previous step of current program when pressed; displays original contents of X-register when released. No instructions are executed (**page 131**).

SST Single step. Displays step number and keycode of current program memory step when pressed; executes instruction, displays result, and moves calculator to next step when released (**page 125**).

Executed as a recorded program instruction:

RTN Return. If executed as a result of a **GTO** instruction, stops execution and returns control to keyboard. If the result of an executed **A**, **B**, **C**, or **D** instruction or a **GSB** instruction, returns control to next step after that instruction (**page 165**).

LBL Label. Followed by number key (**0** through **9**), used as address by **GTO** and **GSB**—performs no operation (**page 141**).

PAUSE Stops program execution and displays contents of X-register for 1 second, then resumes program execution (**page 159**).

PROGRAM Mode

Active keys:

DELETE Deletes current instruction from program memory. All subsequent instructions in that program are moved up one step (page 134).

Automatic RUN Mode

Pressed from the keyboard:

DELETE After **I** prefix key, cancels that key. Does not disturb program memory or calculator status (page 134).

JUMP Followed by **A**, **B**, **C**, **D**, and in turn by a number key (**0** through **9**), moves calculator from current step to first designated label in selected program. No instructions are executed (page 187).

R/S Run/stop. Begins execution from current step of program memory. Stops execution if program is running (page 157).

Any key. Pressing any key on the keyboard stops execution of a running program.

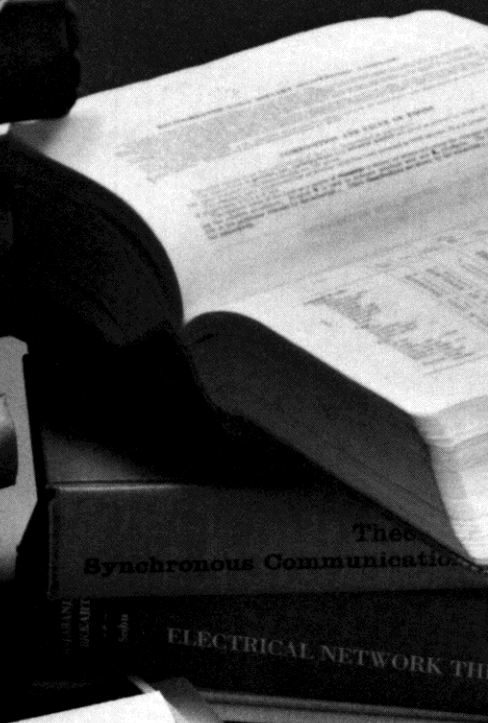
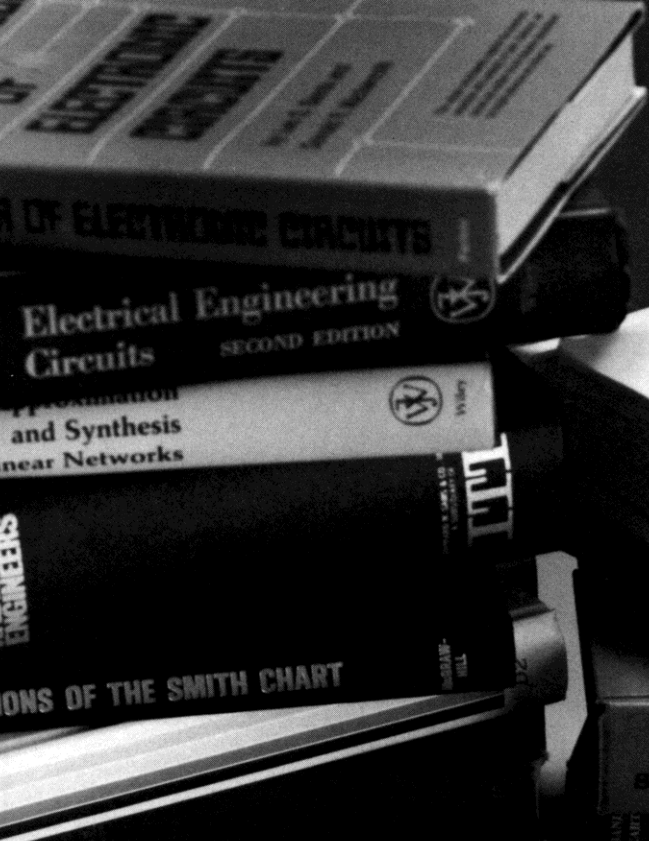
Executed as a recorded program instruction:

JUMP Followed by **A**, **B**, **C**, or **D**, and in turn by a number key (**0** through **9**), transfers execution from one program to the designated label in another program, where execution resumes (page 187).

X=Y **X≠Y** **X≤Y** **X>Y**
X=0 **X≠0** **X<0** **X>0**

Conditionals. Each tests value in X-register against value in Y-register or 0 as indicated. If true, calculator executes instruction in next step of program memory. If false, calculator skips one step before resuming execution (page 145).

R/S Run/stop. Stops program execution (page 157).



Meet The HP-95C

Congratulations!

With your purchase of the HP-95C Scientific Calculator with Partition Programming and Continuous Memory, you have acquired a truly versatile and unique calculating instrument. Using the Hewlett-Packard RPN logic system that slices with ease through the most difficult equations, the HP-95C is without parallel:

As a scientific calculator. As a scientific calculator, the HP-95C features a familiar adding-machine style keyboard for rapid data entry wedded to a powerful calculator with dozens of mathematical, statistical, and scientific functions, and a three-way printer for archival permanence to your answers.

As a problem-solving machine. Following the simple, step-by-step instructions in the *HP-95C Applications Book*, you can key in any of dozens of programs from the areas of mathematics, statistics, games, finance, surveying, and other fields and begin using your calculator. *Immediately.*

As a personal programmable calculator. The HP-95C is so easy to program and use that it requires no prior programming experience or knowledge of arcane programming languages. Yet even the most sophisticated computer experts marvel at the programming features of the HP-95C:

- Partition programming means that up to four programs can be remembered by the calculator—even when the power switch is OFF.
- 16 data storage registers.
- 200 steps of program memory.
- Fully merged prefix and function keys that mean more programming per step.
- Easy-to-use editing features for correcting and modifying programs.
- Powerful unconditional and conditional branching.
- Three levels of subroutines and 10 easily accessed labels—in each program.
- Indirect storage and recall.
- Printer to record results, list programs, or trace executing programs.


And in addition, the HP-95C can be operated from its rechargeable battery pack for *complete portability*, anywhere.


Now let's take a closer look at the HP-95C to see how easy it is to use, whether we solve a problem manually, or whether we use the programming power of the HP-95C to solve the problem at the press of a single key.























Manual Problem Solving

To get the feel of your HP-95C, try a few simple calculations. First, set the switches that are located in the upper left-hand corner of the keyboard as follows:

Set the OFF-ON switch OFF  ON to ON.

Set the PRGM-RUN switch PRGM  RUN to RUN.


Set the MAN-TRACE-NORM switch MAN  ^{TRACE} NORM to MAN.










To solve:	Press:	Display:
$5 + 6 = 11$	   	11.00
$8 \div 2 = 4$	   	4.00
$7 - 4 = 3$	   	3.00
$9 \times 8 = 72$	   	72.00
$\frac{1}{5} = 0.20$	 	0.20
Sine of $30^\circ = 0.50$	    <small>SIN⁻¹</small>	0.50

Now let's try something a little more involved. To calculate the surface area of a sphere, the formula $A = \pi d^2$ can be used, where:

- A is the surface area of the sphere,
- d is the diameter of the sphere,
- π is the value of pi, 3.141592654.

Ganymede, one of Jupiter's 12 moons, has a diameter of 3200 miles. You can use the HP-95C to manually compute the area of Ganymede. Merely press the following keys in order:

First, ensure that a paper roll has been properly installed in the calculator, and slide the MAN-TRACE-NORM switch MAN  ^{TRACE} NORM to NORM.

Press	Display	
   	3200.	Diameter of Ganymede.
	10240000.00	Square of the diameter.
 	3.14	The quantity π .
	32169908.78	Area of Ganymede in square miles.
	32169908.78	The answer printed.

You can see that the paper tape has preserved a record of your calculation. Save this tape—you are going to use it to write a program for your HP-95C.




Programmed Problem Solving

If you wanted the surface areas of each of Jupiter's 12 moons, you could repeat the above procedure 12 times. However, you might wish to write a *program* that would calculate area of a sphere from its diameter, instead of pressing all the keys for each moon.

To calculate the area of a sphere using a program, you should first *write* the program, then you must *record* the program into the calculator, and finally you *run* the program to calculate the answer.

Writing the Program: You have already written it! A program is nothing more than the series of keystrokes you would execute to solve the same problem manually.

Loading the Program: To load the keystrokes of the program into the calculator:



1. Slide the PRGM-RUN switch  RUN to PRGM (program).
2. Press  CLEAR  to clear program A.
3. Press the following keys in order. (When you are loading a program, the display gives you information that you will later find useful, but which you can ignore for now.)




These are the same keys you pressed to solve the problem manually.


The calculator will now remember this keystroke sequence.


Running the Program: To run the program to find the area of any sphere from its diameter:

1. Slide the PRGM-RUN switch  RUN back to RUN.
2. Key in the value of the diameter.
3. Press  to run the program.

When you press , the sequence of keystrokes you loaded is automatically executed by the calculator, giving you the same answer you would have obtained manually.



For example, to calculate the area of Ganymede, with a diameter of 3200 miles:





Press	Display
3200	3200.
	32169908.78 Square miles.

With the program you have loaded, you can now calculate the area of any of Jupiter's moons—in fact, of *any* sphere—using its diameter. You have only to leave the calculator in RUN mode and key in the diameter of each sphere for which you want the area, then press . For example, to compute the surface area of Jupiter's moon Io, with a diameter of 2310 miles:

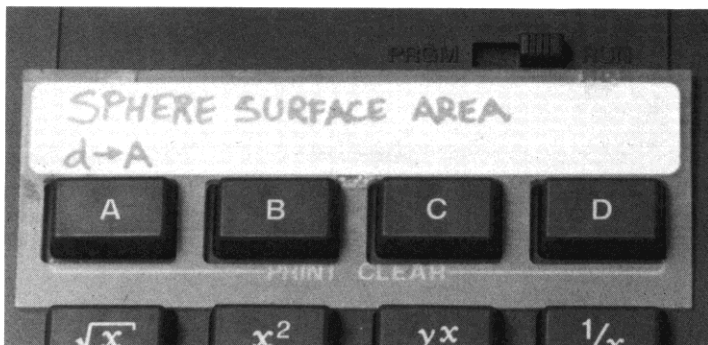
Press	Display
2310 	16763852.56 Square miles.

For the moons Europa, diameter 1950 miles, and Callisto, diameter 3220 miles:

Press	Display	
1950 	11945906.07	Area of Europa in square miles.
3220 	32573289.27	Area of Callisto in square miles.

Programming the HP-95C is *that* easy! The calculator remembers a series of keystrokes and then executes them at the press of a single key. In fact, the HP-95C can remember up to 200 separate operations (and many more keystrokes, since many operations require two, three, or even four keystrokes), partitioned among the four user-definable keys ( ,  ,  , ) in any way you choose. These user-definable keys permit you to preserve up to four programs in your calculator, and to run them at any time.

Labeling the Program: Now that you have a program loaded into the calculator and have assured yourself that it runs properly, you'll probably want to *label* the appropriate user-definable key with a symbol (or mnemonic) so that you will remember what program that key defines. Merely mark one of the program cards with an appropriate symbol, insert it in one of the card windows, and slip the card window over the user-definable keys, as shown here.



Using cards and card windows, you can label any of the programs you load into your HP-95C.

What Continuous Memory Means to You

Your HP-95C contains Continuous Memory—one of the newest, most advanced memory systems available in a scientific calculator. Continuous Memory means the program memory and storage registers stay “on” when your calculator is turned off. You can store your favorite program (or four) for days or weeks!

Continuous Memory is especially convenient when you want to retain data, save battery life, or customize your calculator (e.g., if you use 20% of your programs to solve 80% of your problems). You save considerable time because you don’t have to key in those common programs again and again—they are stored in your calculator. Continuous Memory reduces human entry errors, too; fewer keystrokes mean fewer chances of making inadvertent errors.

Perhaps the most important advantage of Continuous Memory is that it enables you to customize or personalize your calculator. The easiest way to customize your HP-95C is to make a list of the problems you encounter most frequently, rate them in order of priority, then write and save the specialized programs to solve those problems. Whenever you encounter a repetitive problem set, you just write the program once then use it at different times. You can even preserve one or two favorite programs in the calculator and use the other user-definable keys for other programs.

Besides saving programs, Continuous Memory lets you store data in the general-purpose storage registers. Constants, accumulations, and intermediate answers can be retrieved whenever you need them.

Continuous Memory also helps save battery life in many situations. If your HP-95C is left off, Continuous Memory can store your programs for 1½ months or longer. When you do use your calculator, keying in fewer programs means less time that the display is on—hence, less battery drain.

Using This Handbook

New to Hewlett-Packard calculators? Part One of this handbook has been designed to teach you to use your HP-95C as a powerful scientific calculator. By working through these sections of the handbook, you'll learn every function that you can use to calculate answers manually, and you'll come to appreciate the calculating efficiency of the Hewlett-Packard logic system with RPN. And since the programmability of the HP-95C stems from its ability to remember a series of manual keystrokes, Part One, Using Your HP-95C Calculator, is invaluable in laying the groundwork for Part Two, Programming the HP-95C.

Previous HP user? If you've already used Hewlett-Packard pocket or desktop calculators with RPN, you may want to familiarize yourself with the unique printer options of the HP-95C by reading pages 22 and 23, and then turn directly to Part Two, Programming the HP-95C. Later, though, you will undoubtedly wish to peruse Part One at your leisure in order to discover the many calculating advantages of the HP-95C.

Whether an old hand or a novice, you'll find the Function and Key Index on pages 7–11 invaluable as a quick reference guide, a programming guide, or even to illustrate the features of the HP-95C to your friends.

Part One
Using Your HP-95C
Calculator




Section 1

Getting Started


Your HP-95C is shipped fully equipped, including a battery pack. Although the calculator is completely portable, if you want to use your HP-95C on battery power alone, you should connect the ac adapter/recharger and charge the battery for 7–10 hours first. Whether you operate from battery power or from the ac adapter/recharger, *the battery pack must always be in the calculator*. The battery pack is never in danger of being overcharged.

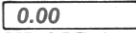



To begin: Slide the OFF-ON switch OFF  ON to ON.

Slide the Print Mode switch MAN  NORM to MAN.

Slide the PRGM-RUN switch PRGM  RUN to RUN.

Display

Numbers that you key into the calculator and intermediate and final answers are always seen in the bright red display. Should you see the display  the first time you turn the calculator ON, do not worry—it means that power to the calculator has been interrupted at some time. Merely press any key on the keyboard to clear the display, then continue.


The displays illustrating most examples in this handbook are shown to two decimal places, like this: . As you will see, numbers can be seen in a wide variety of formats with your HP-95C, but if you want the display on your calculator to look like the ones shown on the next few pages, press    now.


Keyboard

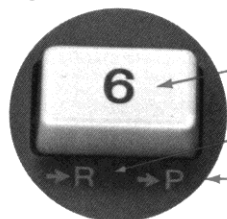
Each key on the keyboard can perform as many as three different functions. One function is indicated on the flat plane of the key face, while the second and third functions may be indicated by printed symbols in gold and blue, respectively, below the key.

There are two *prefix* keys,  and . By pressing one of these prefix keys before pressing a function key, you select the function printed in gold or blue below the function key.



To select the function printed on the face of the key, press the key.


To select the function printed in gold below the key, first press the gold prefix key , then press the function key.

To select the function printed in blue below the key, first press the blue prefix key , then press the function key.



To execute this function, press .

To execute this function, first press , then press .

To execute this function, first press , then press .

In this handbook, the selected key function will appear in the appropriate color outlined by a box, like this: **6**, **÷R**, **÷P**.

Keying In Numbers

Key in numbers by pressing the number keys in sequence, just as though you were writing on a piece of paper. The decimal point must be keyed in if it is part of the number (unless it is to the right of the last digit).

For example, to key in 148.84:

Press	Display
1 4 8 . 8 4	148.84

The resultant number 148.84 is seen in the display.

Negative Numbers

To key in a negative number, press the keys for the number, then press **CHS** (*change sign*). The number, preceded by a minus (–) sign, will appear in the display. For example, to change the sign of the number now in the display:

Press	Display
CHS	–148.84

You can change the sign of either a negative or a positive nonzero number in the display. For example, to change the sign of the –148.84 now in the display back to positive:

Press	Display
CHS	148.84

Notice that only negative numbers are given a sign in the display.

Clearing

You can clear any numbers that are in the display by pressing **CLX** (*clear x*). This key erases the number in the display and replaces it with **0.00**


Press	Display
CLX	0.00


If you make a mistake while keying in a number, clear the entire number string by pressing **CLX**. Then key in the correct number.


Printer

The printer has three modes of operation, which you control using the Print Mode switch

TRACE
MAN  NORM :

With the Print Mode switch **MAN**  **NORM** set to MAN (*manual*), the printer is idle and does not print unless you press the **PRINTX** key or one of the PRINT functions. This mode gives greatest economy of paper and battery power.

With the Print Mode switch  set to NORM (*normal*), the calculator records a history of the calculation sequence so that you can reconstruct your problem. In this mode you see digit entries and functions, but intermediate and final answers are not printed unless you press the **PRINTX** key.

With the Print Mode switch  set to TRACE, the calculator prints numbers, functions, and intermediate and final answers, just as they are seen in the display. The results of functions are printed with the symbol *** to the right of the number.

To advance the printer paper, press the paper advance pushbutton that is to the right of the paper output. Don't worry if the display blanks out while the paper advance is operating—this is normal. To advance the paper more than one space, simply hold the pushbutton down until the paper has advanced the desired amount. To replace the paper roll, refer to Using Your HP-95C Printer in appendix A of this handbook.

No matter what print mode you choose, you seldom have to worry about “overrunning” the printer when you are calculating. Your HP-95C contains a key buffer that “remembers” up to seven keystrokes—no matter how fast you press the keys.

Functions

The best way to see how simple functions operate on your HP-95C is with the Print Mode switch set to TRACE to give you a complete record of inputs, functions, and answers.

Slide the Print Mode switch  to TRACE now.

In spite of the dozens of functions available on the HP-95C keyboard, you will find the calculator functions simple to operate by using a single, all-encompassing rule: *When you press a function key, the calculator immediately executes the function written on the key.*


Pressing a function key causes the calculator to immediately perform that function.

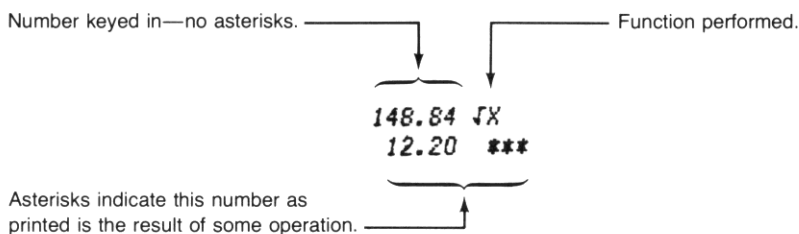
For example, to calculate the square root of 148.84, merely:

Press **Display**

148.84	148.84	148.84 \sqrt{x}
	12.20	12.20 ***

Let's look briefly at the printed copy of that problem to see the simple way that the HP-95C printer duplicates your calculations.

The paper tapes are printed just as you read, from left to right and top to bottom. The number, 148.84, is printed exactly as you keyed it in. A symbol for the function performed, , is printed next to it. The answer, 12.20, is printed with a three-asterisk label to its right, indicating that the HP-95C performed some operation in order to obtain the number as it is printed.



Now let's continue. To square the result of the previous calculation:

Press	Display
x^2	148.84
	x^2 148.84 ***

\sqrt{x} and x^2 are examples of one-number function keys; that is, keys that execute upon a single number. All function keys in the HP-95C operate upon either one number or two numbers at a time (except for statistics keys like $\Sigma+$ and \bar{S} —more about these later).

Function keys operate upon either one number or two numbers.

One-Number Functions

To use any one-number key:

1. Key in the number.
2. Press the function key (or press the prefix key, then the function key).

For example, to use the one-number function $1/x$ key, you first key in the number represented by x , then press the function key. To calculate $1/4$, key in 4 (the x -number) and press $1/x$.

Press	Display
4	4.
$1/x$	0.25
	4.00 $1/x$ 0.25 ***

Now try these other one-number function problems. Remember, *first key in the number, then press the function*:

$\frac{1}{25}$	=	0.04	
$\sqrt{2500}$	=	50.00	
10^5	=	100000.00	(Use the 10^x key.)
$\sqrt{3204100}$	=	1790.00	
$\log 12.58925411$	=	1.10	
71^2	=	5041.00	

Two-Number Functions

Two-number functions are functions that must have two numbers present in order for the operation to be performed. $+$, $-$, \times , and \div are examples of two-number function keys. You cannot add, subtract, multiply, or divide unless there are two numbers present in the calculator. Two-number functions work the same way as one-number functions—that is, the operation occurs when the function key is pressed. Therefore, *both numbers must be in the calculator before the function key is pressed*.

When more than one number must be keyed into the calculator before performing an operation, the **ENTER** key is used to separate the two numbers.

Use the **ENTER** key whenever more than one number must be keyed into the calculator before pressing a function.

If you key in only one number, you never need to press **ENTER**. To place two numbers into the calculator and perform an operation:

1. Key in the first number.
2. Press **ENTER** to separate the first number from the second.
3. Key in the second number.
4. Press the function key to perform the operation.

For example, to add 12 and 3:

Press		
12	The first number.	
ENTER	Separates the first number from the second.	12.00 ENT ↑
3	The second number.	3.00 +
+	The function.	15.00 ***

The answer, 15.00, is displayed and printed.

Other arithmetic functions are performed the same way:

To perform	Press	Display	
12 - 3	12 ENTER 3 -	9.00	12.00 ENT ↑ 3.00 - 9.00 ***
12 × 3	12 ENTER 3 ×	36.00	12.00 ENT ↑ 3.00 × 36.00 ***
12 ÷ 3	12 ENTER 3 ÷	4.00	12.00 ENT ↑ 3.00 ÷ 4.00 ***

The y^x key is also a two-number operation. It is used to raise numbers to powers, and you can use it in the same simple way that you use every other two-number function key:

1. Key in the first number.
2. Press **ENTER** to separate the first number from the second.
3. Key in the second number (power).
4. Perform the operation (press y^x).

When working with any function key (including y^x), you should remember that the displayed number is always designated by x on the function key symbols.

The number displayed is always x .

So \sqrt{x} means square root of the displayed number, $\sqrt[n]{x}$ means $\frac{1}{\text{displayed number}}$, etc.

Thus, to calculate 3^6 :

Press	Display	
3	3.	
ENTER	3.00	3.00 ENT↑
6	6.	6.00 y^x
		729.00 ***
y^x	729.00	The answer.

Now try the following problems using the y^x key, keeping in mind the simple rules for two-number functions:

16^4	(16 to the 4 th power) =	65536.00	
81^2	(81 squared) =	6561.00	(You could also have done this as a one-number function using x^2 .)
$225^{-.5}$	(Square root of 225) =	15.00	(You could also have done this as a one-number function using \sqrt{x} .)
2^{16}	(2 to the 16 th power) =	65536.00	
$16^{.25}$	(4 th root of 16) =	2.00	

Chain Calculations

The speed and simplicity of operation of the Hewlett-Packard logic system become most apparent during chain calculations. Even during the longest of calculations, you still perform only one operation at a time, and you see the results as you calculate—the Hewlett-Packard automatic memory stack stores up to four intermediate results inside the calculator until you need them, then inserts them into the calculation. This system makes the process of working through a problem as natural as it would be if you were working it out with pencil and paper, but the calculator takes care of the hard part.

For example, solve $(12 + 3) \times 7$.

If you were working the problem with a pencil and paper, you would first calculate the intermediate result of $(12 + 3)$...

$$\begin{array}{r} \cancel{(12 + 3)} \times 7 = \\ 15 \end{array}$$

...and then you would multiply the intermediate result by 7.

$$\begin{array}{r} \cancel{(12 + 3)} \times 7 = 105 \\ 15 \times 7 \end{array}$$

You work through the problem exactly the same way with the HP-95C, one operation at a time. You solve for the intermediate result first...

$$(12 + 3)$$


Press	Display	
12	12.	12.00 ENT↑
ENTER →	12.00	3.00 +
3	3.	15.00 ***
+	15.00	Intermediate result.

...and then solve for the final answer. You don't need to press **ENTER** → to store the intermediate result—the HP-95C automatically stores it inside the calculator when you key in the next number. To continue...

Press	Display	
7	7.	The intermediate result from the preceding operation is automatically stored inside the calculator when you key in this number.
×	105.00	Pressing the function key multiplies the new number and the intermediate result, giving you the final answer.
		7.00 ×
		105.00 ***

Because the HP-95C stores intermediate results automatically, you don't need to print them. You can slide the Print Mode switch to NORM to preserve a record of your calculations, and then press **PRINT X** to print the final answer.

For example, when you solved the above problem in TRACE mode, you preserved *all* intermediate and final results. To solve the same problem and preserve only a history of the calculation:

Slide the Print Mode switch  to NORM.

Press	Display	
12	12.	
ENTER	12.00	
3	3.	12.00 ENT↑
+	15.00	3.00 +
7	7.	7.00 ×
x	105.00	105.00 ***
PRINT X	105.00	

Preserves the final answer in your printed record.

Now try these problems. Notice that for each problem you only have to press **ENTER** to insert a pair of numbers into the calculator—each subsequent operation is performed using a new number and an automatically stored intermediate result.

To solve	Press	Display	
$\frac{(2 + 3)}{10}$	2		
	ENTER		2.00 ENT↑
	3		3.00 +
	+		10.00 ÷
	10		0.50 ***
	÷	0.50	
	PRINT X	0.50	
$3(16 - 4)$	16		
	ENTER		16.00 ENT↑
	4		4.00 -
	-		3.00 ×
	3		36.00 ***
	x	36.00	
	PRINT X	36.00	

To solve

Press

Display

$$\frac{14 + 7 + 3 - 2}{4}$$

14

ENTER

7

+

3

+

2

-

4

=

PRINT

5.50

5.50

14.00 ENT

7.00 +

3.00 +

2.00 -

4.00 ÷

5.50 ***

Problems that are even more complicated can be solved in the same simple manner, using the automatic storage of intermediate results. For example, to solve $(2 + 3) \times (4 + 5)$ with a pencil and paper, you would:

$$(2 + 3) \times (4 + 5)$$

First solve for the contents
of these parentheses...

...and then you would multiply the
two intermediate answers together.

...and then for these parentheses...

You work through the problem the same way with the HP-95C. First you solve for the intermediate result of $(2 + 3)$...

Press	Display
2	2.
ENTER	2.00
3	3
+	5.00
	Intermediate result.

2.00 ENT

3.00 +

Then add 4 and 5:

(Since you must now key in another *pair* of numbers before you can perform a function, you use the ENTER key again to separate the first number of the pair from the second.)

Procedure	Press	Display
$(2 + 3) \times (4 + 5)$ 5 9	4 ENTER 5 +	9.00
		4.00 ENT
		5.00 +

Then multiply the intermediate answers together for the final answer:

Procedure	Press	Display
$(2 + 3) \times (4 + 5)$ 5 9	X	45.00
	PRINT	45.00 ***

45.00 ***

Notice that you didn't need to write down or key in the intermediate answers from inside the parentheses before you multiplied—the HP-95C automatically stacked up the intermediate results inside the calculator for you and brought them out on a last-in, first-out basis when it was time to multiply.

No matter how complicated a problem may look, it can always be reduced to a series of one- and two-number operations. Just work through the problem in the same logical order you would use if you were working it with a pencil and paper.

For example, to solve:

$$\frac{(9 + 8) \times (7 + 2)}{(4 \times 5)}$$

Press	Display		
9 ENTER 8 +	17.00	Intermediate result of (9 + 8).	9.00 ENT ↑
7 ENTER 2 +	9.00	Intermediate result of (7 + 2).	8.00 +
×	153.00	(9 + 8) multiplied by (7 + 2).	7.00 ENT ↑
4 ENTER 5 ×	20.00	Intermediate result of (4 × 5).	2.00 +
÷	7.65	The final answer.	×
PRINT x	7.65		4.00 ENT ↑
			5.00 ×
			÷
			7.65 ***

Now try these problems. Remember to work through them as you would with a pencil and paper, but don't worry about intermediate answers—they're handled automatically by the calculator.

$$(2 \times 3) + (4 \times 5) = 26.00$$

$$\frac{(14 + 12) \times (18 - 12)}{(9 - 7)} = 78.00$$

$$\frac{\sqrt{16.3805 \times 5}}{.05} = 181.00$$

$$4 \times (17 - 12) \div (10 - 5) = 4.00$$

$$\sqrt{(2 + 3) \times (4 + 5)} + \sqrt{(6 + 7) \times (8 + 9)} = 21.57$$

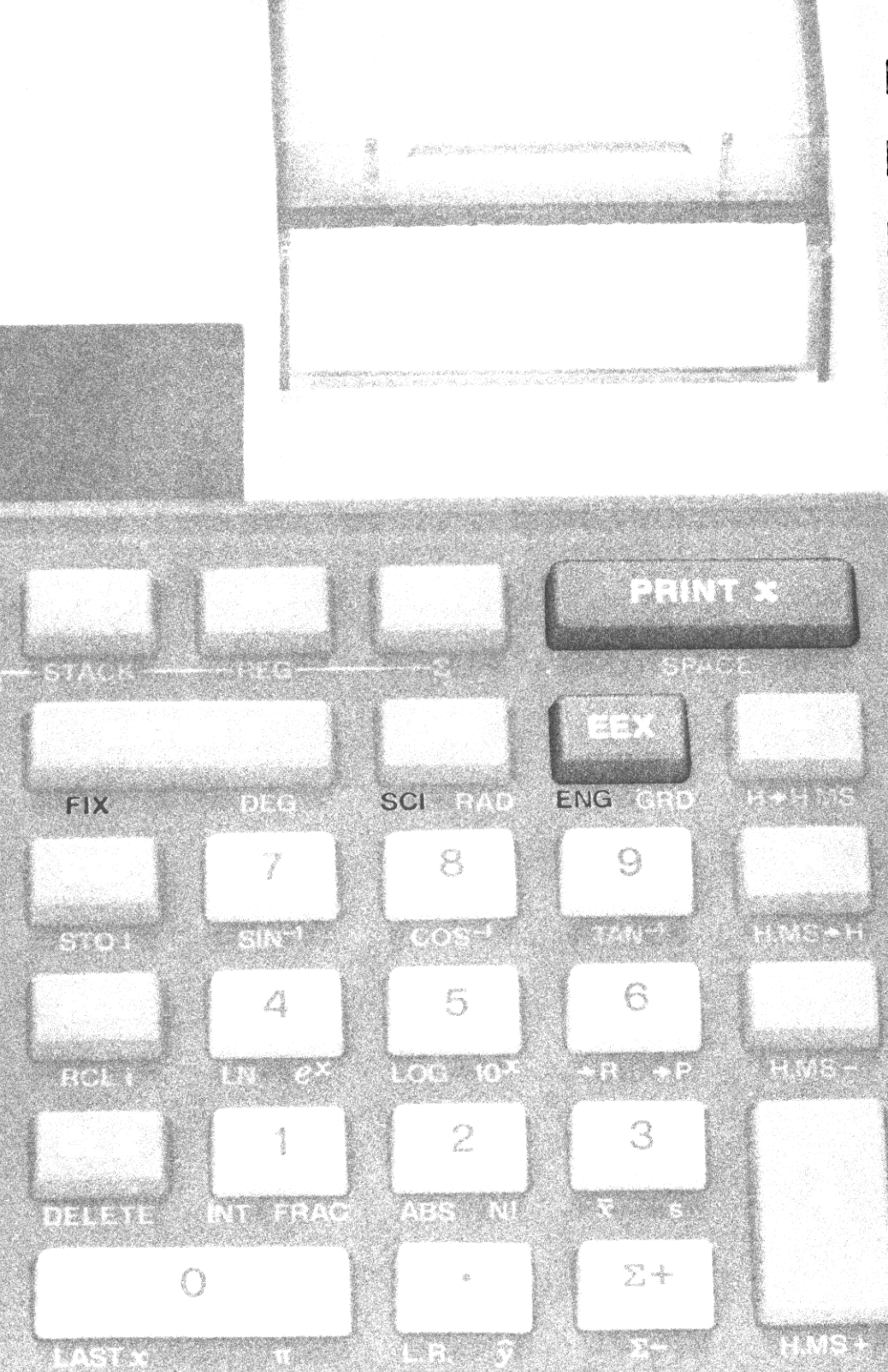
A Word About the HP-95C

Now that you've learned how to use the calculator, you can begin to fully appreciate the benefits of the Hewlett-Packard logic system. With this system, you enter numbers using a parenthesis-free, unambiguous method called RPN (Reverse Polish Notation).

It is this unique system that gives you all these calculating advantages:

- You never have to work with more than one function at a time. The HP-95C cuts problems down to size instead of making them more complex.
- Pressing a function key immediately executes the function. You work naturally through complicated problems, with fewer keystrokes and less time spent.
- Intermediate results appear as they are calculated. There are no "hidden" calculations, and you can check each step as you go.
- Intermediate results are automatically handled. You don't even have to print out long intermediate answers when you work a problem. (Of course, if you want intermediate answers, the HP-95C printer will record them in TRACE mode.)
- Intermediate answers are automatically inserted into the problem on a last-in, first-out basis. You don't have to remember where they are and then summon them.
- You can calculate in the same order that you do with pencil and paper. You don't have to think the problem through ahead of time.

The HP system takes a few minutes to learn. But you'll be amply rewarded by the ease with which the HP-95C solves the longest, most complex equations. With HP, the investment of a few moments of learning yields a lifetime of mathematical dividends.



PRINT x

STACK

FEG

S

SPACE

EEX

FIX

DEG

SCI RAD

ENG GRD

H→HMS

7

8

9

STO 1

SIN⁻¹

COS⁻¹

TAN⁻¹

HMS→H

4

5

6

RCL 1

LN e^x

LOG 10^x

→R →P

HMS-

1

2

3

DELETE

INT FRAC

ABS NI

Σ⁻ Σ⁺

0

.

Σ+

LAST x

π

LR ŷ

Σ-

HMS+

Printer and Display Control

In the HP-95C, you can select many different rounding options for display and printing of numbers. After a SEE 1 (power interrupt) display has been cleared (by pressing any key), for example, the calculator “wakes up” with numbers appearing rounded to two decimal places. Thus, the fixed constant π , which is actually in the calculator as 3.141592654, will *appear in the display* as 3.14 (unless you tell the calculator to display the number rounded to a greater or lesser number of decimal places).

Although a number may be shown to, say, only two decimal places, the HP-95C always computes internally using each number as a 10-digit mantissa and a two-digit exponent of 10. For example, when you compute 2×3 , you might *see* the answer to only two decimal places:

Press	Display
2 ENTER 3 ×	6.00

However, inside the calculator all numbers have 10-digit mantissas and two-digit exponents of 10. So the HP-95C *actually* calculates using full 10-digit numbers:

2.000000000 $\times 10^{00}$ ENTER 3.000000000 $\times 10^{00}$ ×

yields an answer that is actually carried to full 10 digits internally:

6.000000000 $\times 10^{00}$

You see only these digits... ...but these digits are also present.

Display Control Keys

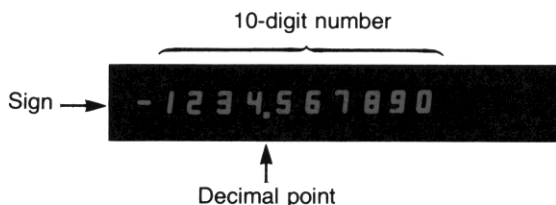
There are three functions, FIX, SCI, and ENG, that allow you to control the manner in which numbers appear in the display in the HP-95C.

FIX displays and prints numbers in fixed decimal point format, while SCI permits you to view numbers in a scientific notation format. ENG displays numbers in engineering notation, with exponents of 10 shown in multiples of three (e.g., 10^3 , 10^{-6} , 10^{12}). By pressing a digit key (0 through 9) after any of these display control functions, you specify the number of digits displayed.

No matter which format or how many displayed digits you choose, display control alters only the manner in which a number is displayed and printed in the HP-95C. The actual number itself is not altered by any of the print options or the display control keys. No matter what type of display you select, the HP-95C always calculates internally with a full 10-digit number, multiplied by 10 raised to a two-digit exponent.

Display mode is one of the items that is maintained by the Continuous Memory of the HP-95C, so even though you may turn the calculator OFF, when you turn it ON again you will see numbers displayed in the manner you selected earlier. If power to the calculator has been interrupted for some reason (resulting in a SEE 1 display), when you clear the power interrupt display by pressing any key you will see the calculator reset to a display with two decimal places. This is the display used to illustrate most of the examples in this handbook, but, as you will see, you can work problems with the display set to any mode you desire.

Fixed Point Display



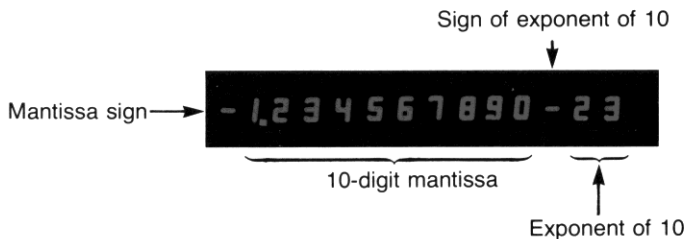
Using fixed point display, you can specify the number of places to be shown after the decimal point. It is selected by pressing FIX followed by a number key to specify the number of decimal places (0 through 9) to which the display is to be rounded. The displayed number begins at the left side of the display (or the right side of the printed tape) and includes trailing zeros within the setting selected.

For example:

Slide the Print Mode switch
MAN
NORM
 to MAN now so that you can concentrate on the display changes.

Press	Display	
f FIX 2	0.00	FIX 2 display mode used in this handbook.
123.4567	123.4567	
f FIX 0	123.	Display is rounded off to 0 decimal places. Internally, however, the number maintains its original value of 123.4567.
f FIX 4	123.4567	
f FIX 7	123.4567000	
f FIX 1	123.5	Notice that the display rounds if the first <i>hidden</i> digit is 5 or greater.
f FIX 2	123.46	Normal FIX 2 display.

Scientific Notation Display



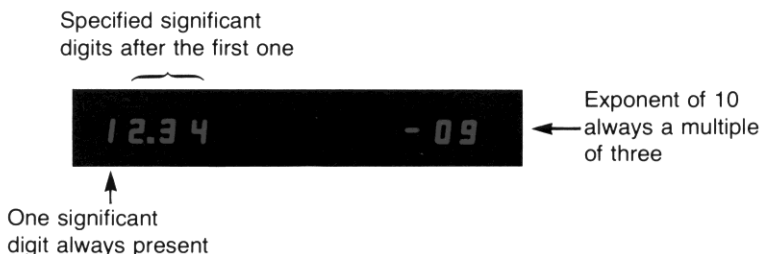
In scientific notation each number is displayed with a single digit to the left of the decimal point followed by a specified number of digits (up to nine) to the right of the decimal point and multiplied by a power of 10. Scientific notation is particularly useful when working with very large or small numbers.

Scientific notation is selected by pressing **f** **[SCI]** followed by a digit key to specify the number of decimal places to which the number is rounded. The display is left-justified and includes trailing zeros within the selected setting. The printed copy is right-justified, with a sign to identify the exponent of 10. For example:

Press	Display	
123.4567	123.4567	
f [SCI] 2	1.23	02 Indicates 1.23×10^2 .
f [SCI] 4	1.2346	02 Indicates 1.2346×10^2 . Notice that the display rounds if the first <i>hidden</i> mantissa digit is 5 or greater.
f [SCI] 7	1.2345670	02 Indicates 1.2345670×10^2 .
f [SCI] 9	1.234567000	02 Indicates 1.234567000×10^2 .

Note: You can easily key in numbers in scientific notation format by using the **[EEX]** (*enter exponent*) key—more about this later.

Engineering Notation Display



Engineering notation allows all numbers to be shown with exponents of 10 that are multiples of three (e.g., 10^3 , 10^{-6} , 10^{12}).

This is particularly useful in scientific and engineering calculations, where units of measure are often specified in multiples of three. Refer to the prefix chart below.

Multiplier	Prefix	Symbol
10^{12}	tera	T
10^9	giga	G
10^6	mega	M
10^3	kilo	k
10^{-3}	milli	m
10^{-6}	micro	μ
10^{-9}	nano	n
10^{-12}	pico	p
10^{-15}	femto	f
10^{-18}	atto	a

Engineering notation is selected by pressing **f** **ENG** followed by a number key. The first significant digit is always present in the display, and the number key specifies the number of additional significant digits to which the display is rounded. The decimal point always appears in the display. For example:

Press	Display	
.000012345	.000012345	
f ENG 1	12. -06	Engineering notation display. Number appears in the display rounded off to one significant digit after the omnipresent first one. Power of 10 is proper multiple of three.
f ENG 3	12.35 -06	Display is rounded off to third significant digit after the first one.
f ENG 9	12.34500000 -06	
f ENG 0	10. -06	Display rounded off to first significant digit.

Notice that rounding can occur to the *left* of the decimal point, as in the case of **ENG** 0 specified above.

When engineering notation has been selected, the decimal point shifts to show the mantissa as units, tens, or hundreds in order to maintain the exponent of 10 as a multiple of three. For example, multiplying the number now in the calculator by 10 causes the decimal point to shift to the right without altering the exponent of 10:

Press	Display	
2	12.3	-06
10	123.	-06

However, multiplying again by 10 causes the exponent to shift to another multiple of three and the decimal point to move to the units position. Since you specified 2 earlier, the HP-95C maintains two significant digits after the first one when you multiply by 10.

Press	Display	
10	1.23	-03

Decimal point shifts. Power of 10 shifts to 10^{-3} . Display maintains two significant digits after the first one.







Format of Printed Numbers

When using the printer, whether you are in MAN or NORM mode (where you must press to see answers) or in TRACE (where the HP-95C automatically prints answers as they are calculated), printed numbers can be shown in any display format—fixed point, scientific notation, or engineering notation. By selecting the display format, you also select the print format.

Results from your HP-95C are always displayed and printed in the format that you have chosen. The three-asterisk label that you see printed next to a result is a guarantee that it is in the chosen display format. Although numbers in the display are left-justified, printed numbers are right-justified.

Numbers that you key in—that is, numbers that are *not* the results of operations—are also printed by the HP-95C. When you key in a number with the Print Mode switch set to NORM or TRACE, the HP-95C does not print it until you change display format or press a function key. Then the number is printed *exactly as you keyed it in*. (One case is an exception to this rule—more about that later.) A number that you keyed in is not the result of an operation, and no asterisks are printed to its right. Subsequent *results*, of course, are printed in the selected format with a three-asterisk label. For example:

Slide the Print Mode switch **MAN**  **TRACE** NORM to NORM.

Press	Display		
.0000123456	.0000123456		
  3	1.235	-05	When you press any function, the number is first printed just as you keyed it in.
	1.235	-05	Results of functions, including display formatting, are printed in the selected format.
1234567890	1234567890.		
  6	1.234568	09	The number is printed as you keyed it in.
	1.234568	09	The three-asterisk label guarantees that the number is now in the selected format.

.0000123456 SCI3
1.235-05 ***
1234567890. ENG6
1.234568+09 ***

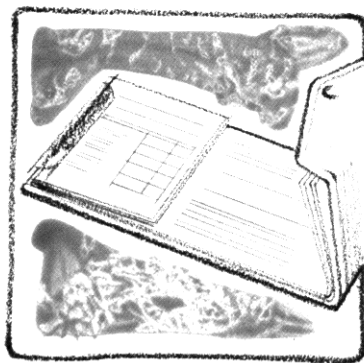
Notice that the HP-95C *prints* a + sign to show you positive exponents of 10.


Thus, whenever you key in a number, the HP-95C prints it just as you keyed it in; *then* the format is changed. It is easy for you to reconstruct your calculation because your exact inputs are identifiable from your printed copy.

When you have keyed in a number, there is one time that the HP-95C will change its format *before* printing. If you have specified fixed point notation the HP-95C will attempt to align the decimal points for easy readability on your printed copy. It will do this in fixed point notation by printing the number that you keyed in in the *specified* format (if the number can be printed without truncating), adding trailing zeros and exponent if necessary.

This feature permits you to key in numbers in fixed point notation and line up the decimal points in the printed record of your calculations.

Example: You begin the month with a balance of \$735.43 in your checking account. During the month, you write checks for \$235, \$79.95, \$5, \$1.44, \$17.83, \$50, and \$12.43. Calculate the closing balance for the account and preserve a printed record of your calculations.



First, ensure that the Print Mode switch **MAN**  **TRACE** NORM is set to NORM.

Press	Display	
f FIX 2	0.00	Normal FIX 2 display. (Display shown assumes no results remain from previous example.)
.05 ENTER +	0.05	
3 y^x PRINT x	1.250000000-04	Display automatically switched to SCI 9 to show answer.
		<div style="text-align: right;"> FIX2 .05 ENT↑ 3.00 y^x 1.250000000-04 *** </div>

After automatically switching from fixed to scientific, when a new number is keyed in or **CLx** is pressed, the display automatically reverts back to the fixed point display originally selected.

The HP-95C also switches to scientific notation if the answer is too large ($\geq 10^{10}$) for fixed point display. For example, the display will not switch from fixed if you solve 1582000×1842 :

Press	Display	
1582000	1582000.	
ENTER +	1582000.00	
1842 x	2914044000.	Fixed point format.
PRINT x	2914044000.	
		<div style="text-align: right;"> 1582000.00 ENT↑ 1842.00 x 2914044000. *** </div>

However, if you multiply the result by 10, the answer is too large for fixed point notation, and the calculator display switches automatically to scientific notation:

Press	Display	
10 x PRINT x	2.914044000 10	Scientific notation format.
		<div style="text-align: right;"> 10.00 x 2.914044000+10 *** </div>

Notice that automatic switching is between fixed and scientific notation display modes only—engineering notation display must be selected from the keyboard.

Keying In Exponents of Ten

You can key in numbers multiplied by powers of 10 by pressing **EEX** (*enter exponent of 10*) followed by number keys to specify the exponent of 10. For example, to key in 15.6 trillion (15.6×10^{12}), and multiply it by 25:

Press	Display	
15.6	15.6	
EEX	15.6 00	
12	15.6 12	This means 15.6×10^{12} .

Now Press Display

ENTER↑ 1.560000000 13
 25 **×** **PRINT** **x** 3.900000000 14

15.60+12 **ENT**↑
 25.00 **×**
 3.900000000+14 *******

You can save time when keying in exact powers of 10 by merely pressing **EEX** and then pressing the desired power of 10. For example, key in 1 million (10^6) and divide by 52.

Press Display

EEX 1. 00 You do not have
 to key in the num-
 ber 1 before
 pressing **EEX**
 when the number
 is an exact
 power of 10.

6 1. 06
ENTER↑ 1000000.00

1.00+06 **ENT**↑
 52.00 **÷**
 19230.77 *******

Since you have
 not specified
 scientific notation,
 the display reverts
 to fixed point
 notation when you
 press **ENTER**↑.

52 **÷** **PRINT** **x** 19230.77

To see your answer in scientific notation with six decimal places:

Press Display

f **SCI** 6 1.923077 04
PRINT **x** 1.923077 04

SCI6
 1.923077+04 *******

To key in negative exponents of 10, key in the number, press **EEX**, press **CHS** to make the exponent negative, then key in the power of 10. For example, key in Planck's constant (h)—roughly, 6.625×10^{-27} erg sec.—and multiply it by 50.

Press Display

CLx 0.000000 00
f **FIX** 2 0.00
 6.625 **EEX** 6.625 00
CHS 6.625 -00
 27 6.625 -27
ENTER↑ 6.625000000-27
 50 **×** **PRINT** **x** 3.312500000-25 Erg sec.

CL **x**
 FIX2
 6.625-27 **ENT**↑
 50.00 **×**
 3.312500000-25 *******

Calculator Overflow


When the number in the display would be greater than $9.99999999 \times 10^{99}$, the HP-95C displays all 9's to indicate that the problem has exceeded the calculator's range. For example, if you solve $(1 \times 10^{49}) \times (1 \times 10^{50})$, the HP-95C will display the answer:

Press	Display
CL X	0.00 CL X
EE X 49 ENTER +	1.000000000 49 1.00+49 ENT ↑
EE X 50 X	1.000000000 99 1.00+50 X
PRINT X	1.000000000 99 1.000000000+99 ***


But if you attempt to multiply the above result by 100, the HP-95C displays overflow by showing you all 9's:

Press	Display
100 X PRINT X	9.999999999 99 100.00 X Overflow indication. 9.999999999+99 ***

Error Display

If you happen to key in an improper operation, the word *See* will appear in the display, followed by a number code to indicate the cause of the error. In addition, if the Print Mode switch **MAN**  **NORM** is set to **NORM** or **TRACE**, the printer will print *See* followed by an error code number.


For example, if you attempt to calculate the square root of -4, the HP-95C will recognize it as an improper operation:

Ensure that the Print Mode switch **MAN**  **NORM** is set to **NORM**.

Press	Display
4 CHS	-4. -4.00 √X
√X	SEE 5

Error-causing conditions and error codes are listed in appendix B of this handbook. Pressing any key clears the error and is *not* executed. (Pressing the paper advance push-button clears the error and *is* executed.) The number that was in the display before the error-causing function is returned to the display so that you can see it.

Press	Display
CL X	-4.00

An error-causing operation during a running program stops the program at the step that caused the error, just as if you had pressed that key from the keyboard. You can then refer to the error code to see the type of error, and slide the PRGM-RUN switch **PRGM**  **RUN** to **PRGM** to see the step number of program memory that contains the error-causing instruction.

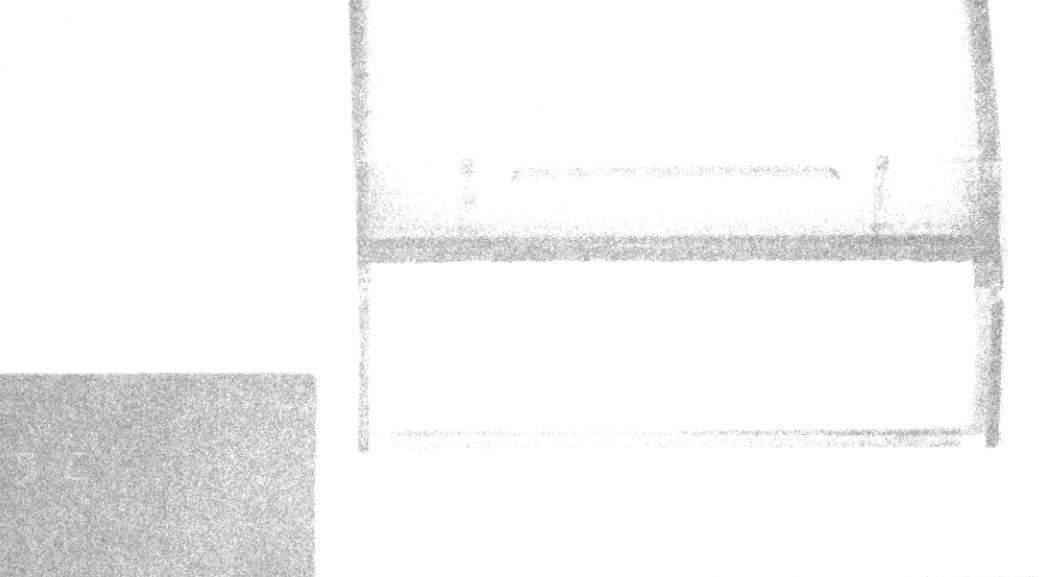
Low Power Indicator

When you are operating the HP-95C from battery power, a red lamp inside the display will glow to warn you that the battery is close to discharge.



Low Power Indicator

You must then connect the ac adapter/recharger to the calculator and operate from ac power, or you must substitute a fully charged battery pack for the one that is in the calculator. Refer to appendix A for a description of these operations.







The Automatic Memory Stack

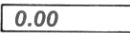
The Stack

Automatic storage of intermediate results is the reason that the HP-95C slides so easily through the most complex equations. And automatic storage is made possible by the Hewlett-Packard automatic memory stack.

Display

You can work through this section with the Print Mode switch at any setting you desire. However, the printed tapes that illustrate the examples in this section were created with the

Print Mode switch   NORM set to NORM. In addition, the display format used is obtained by pressing   2.

When you see decimal digits like  in the display, the number represents the contents of the “X-register” in the calculator.


Basically, numbers are stored and manipulated in the machine “registers.” Each number, no matter how few digits (e.g., 0, 1, or 5) or how many (e.g., 3.141592654, -23.28362, or $2.87148907 \times 10^{27}$), occupies one entire register.

The displayed X-register, which is the only visible register, is one of four registers inside the calculator that are positioned to form the automatic memory stack. We label these registers X, Y, Z, and T. They are “stacked” one on top of the other with the displayed X-register on the bottom. When the calculator is switched ON, these four registers are cleared to 0.00. (They are *not* maintained by the Continuous Memory.)

Switch the HP-95C OFF, then ON.

Name	Register
T	
Z	
Y	
X	

Always displayed.

You can view the contents of the entire stack at any time by printing them using the  (print stack) key.

Press	Display
 PRINT 	0.00

	PR	S
0.00	T	
0.00	Z	
0.00	Y	
0.00	X	

Notice that **f** PRINT **STACK**, like **PRINT X** and the other print functions, operates regardless of the position of the Print Mode switch.

Manipulating Stack Contents

The **R↓** (roll down), **R↑** (roll up), and **X↔Y** (*x exchange y*) keys allow you to review the stack contents or to shift data within the stack for computation at any time.

Reviewing the Stack

To see how the **R↓** key works, first load the stack with numbers 1 through 4 by pressing:

4 **ENTER↑** 3 **ENTER↑** 2 **ENTER↑** 1

4.00 ENT↑
3.00 ENT↑
2.00 ENT↑

The numbers that you keyed in are now loaded into the stack, and its contents look like this:

T	4.00
Z	3.00
Y	2.00
X	1.

Display

To see the contents of the stack now:

Press **f** PRINT **STACK** Display

f PRINT **STACK** 1.00

1.00 PR S
4.00 T
3.00 Z
2.00 Y
1.00 X

When you press the **R↓** key, the stack contents shift downward one register. So the last number that you have keyed in will be rotated around to the T-register when you press **R↓**. When you press **R↓** again, the stack contents again roll downward one register.

To see how the **R↓** key operates, press **f** PRINT **STACK** to list the stack contents after each press of the **R↓** key:

Press **R↓** Display

R↓

f PRINT **STACK** 2.00

R↓
PR S
1.00 T
4.00 Z
3.00 Y
2.00 X

Press Display

R↓
f PRINT **STACK** 3.00

R↓
PR S
 2.00 T
 1.00 Z
 4.00 Y
 3.00 X

R↓
f PRINT **STACK** 4.00

R↓
PR S
 3.00 T
 2.00 Z
 1.00 Y
 4.00 X

R↓
f PRINT **STACK** 1.00

R↓
PR S
 4.00 T
 3.00 Z
 2.00 Y
 1.00 X

Once again the number 1.00 is in the displayed X-register. Four presses of the **R↓** key roll the stack down four times, returning the contents of the stack to their original registers.

You can also manipulate the stack contents using the **R↑** (roll up) key. This key rolls the stack contents *up* instead of down, but it otherwise operates in the same manner as the **R↓** key.

Exchanging x and y

The **x↔y** (*x exchange y*) key exchanges the contents of the X- and the Y-registers without affecting the Z- and T-registers. If you press **x↔y** with data intact from the previous example, the numbers in the X- and Y-registers will be changed...

... from this ...

T	4.00
Z	3.00
Y	2.00
X	1.00

Display

... to this.

T	4.00
Z	3.00
Y	1.00
X	2.00

Display



You can verify this by first listing the stack contents and then pressing **x↔y**. To see the results, list the stack contents again:

Press	Display	
f PRINT STACK	1.00	PR S
x↔y	2.00	4.00 T
f PRINT STACK	2.00	3.00 Z
		2.00 Y
		1.00 X
		X↔Y
		PR S
		4.00 T
		3.00 Z
		1.00 Y
		2.00 X

Notice that whenever you move numbers in the stack using one of the data manipulation keys, the actual stack registers maintain their positions. Only the *contents* of the registers are shifted. The contents of the X-register are always displayed.

Clearing the Stack

When you press **CLx** (*clear x*), you clear only the displayed X-register. To clear the entire automatic memory stack, including the displayed X-register, use the **g** CLEAR **STACK** function. **g** CLEAR **STACK** replaces all numbers in the stack with zeros.

Although it may be comforting, *it is never necessary to clear the displayed X-register when starting a new calculation.* This will become obvious when you see how old results in the stack are automatically lifted by new entries.

Press **CLx** now, and the stack contents are changed...

... from this ...

T	4.00
Z	3.00
Y	1.00
X	2.00

Display

... to this.

T	4.00
Z	3.00
Y	1.00
X	0.00

Display

You can verify that only the X-register contents are affected by listing the stack contents after you have pressed **CLx**:

Press	Display	
f PRINT STACK	0.00	CL X
		PR S
		4.00 T
		3.00 Z
		1.00 Y
		0.00 X

Now press **9** **CLEAR** **STACK**. The contents of the stack are changed...

... from this ...

T	4.00
Z	3.00
Y	1.00
X	0.00

Display

... to this.

T	0.00
Z	0.00
Y	0.00
X	0.00

CL S

You can verify that the stack has been cleared completely and now contains all zeros by listing the stack contents:

Press **Display**

1 **PRINT** **STACK** 0.00

PR S
0.00 T
0.00 Z
0.00 Y
0.00 X

The **ENTER** Key

When you key a number into the calculator, its contents are written into the displayed X-register. For example, if you key in the number 314.32 now, you can see that the display contents are altered.

When you key in 314.32, the contents of the stack registers are changed...

... from this ...

T	0.00
Z	0.00
Y	0.00
X	0.00

... to this.

T	0.00
Z	0.00
Y	0.00
X	314.32

In order to key in another number at this point, you must first terminate digit entry—i.e., you must indicate to the calculator that you have completed keying in the first number and that any new digits you key in are part of a new number.

Use the **ENTER** key to separate the digits of the first number from the digits of the second.

When you press the **ENTER** key, the contents of the stack registers are changed...

... from this ...

T	0.00
Z	0.00
Y	0.00
X	314.32

Display

... to this.

T	0.00
Z	0.00
Y	314.32
X	314.32

Display

As you can see, the number in the displayed X-register is copied into Y. The numbers in Y and Z have also been transferred to Z and T, respectively, and the number in T has been lost off the top of the stack. But this will be more apparent when we have different numbers in all four registers.

Immediately after pressing **ENTER**, the X-register is prepared for a new number, and that new number writes over the number in X. For example, key in the number 543.28 and the contents of the stack registers change...

... from this ...

T	0.00
Z	0.00
Y	314.32
X	314.32

Display

... to this.

T	0.00
Z	0.00
Y	314.32
X	543.28

Display

CLX replaces any number in the display with zero. Any new number then writes over the zero in X.

For example, if you had meant to key in 689.4 instead of 543.28, you would press **CLX** now to change the stack...

... from this ...

T	0.00
Z	0.00
Y	314.32
X	543.28

Display

... to this.

T	0.00
Z	0.00
Y	314.32
X	0.00

Display

And then key in 689.4 to change the stack...

... from this ...

T	0.00
Z	0.00
Y	314.32
X	0.00

Display

... to this.

T	0.00
Z	0.00
Y	314.32
X	689.4

Display

Notice that numbers in the stack do not move when a number is keyed in immediately after you press **ENTER**, **CLX**, or one of the **PRINT** functions. However, numbers in the stack *do* lift upward when a new number is keyed in immediately after you press most other functions, including **R+**, **R+**, and **X²Y**. See appendix C, Stack Lift and LASTX, for a complete explanation and list of the operations that cause the stack to lift. (If you follow a regular function like **R+** or **X²** with a **PRINT** function, then key in a number, the stack will lift.)

One-Number Functions and the Stack

One-number functions execute upon the number in the X-register only, and the contents of the Y-, Z-, and T-registers are unaffected when a one-number function key is pressed.

For example, with numbers positioned in the stack as in the earlier example, pressing the \sqrt{x} key changes the stack contents...

... from this ...

T	0.00
Z	0.00
Y	314.32
X	689.4

Display

... to this.

T	0.00
Z	0.00
Y	314.32
X	26.26

Display

The one-number function executes upon only the number in the displayed X-register, and the answer writes over the number that was in the X-register. No other register is affected by a one-number function.

Two-Number Functions and the Stack

Hewlett-Packard calculators do arithmetic by positioning the numbers in the stack the same way you would on paper. For instance, if you wanted to add 34 and 21 you would write 34 on a piece of paper and then write 21 underneath it, like this:

$$\begin{array}{r} 34 \\ 21 \\ \hline \end{array}$$

and then you would add, like this:

$$\begin{array}{r} 34 \\ +21 \\ \hline 55 \end{array}$$

Numbers are positioned the same way in the HP-95C. Here's how it is done. (If you clear the stack first by pressing g CLEAR STACK , the numbers in the stack will correspond to those shown here in the example.)

Press	Display	
g CLEAR STACK	0.00	
34	34.	34 is keyed into X.
ENTER \uparrow	34.00	34 is copied into Y.
21	21.	21 writes over the 34 in X.
		CL S 34.00 ENT \uparrow

Use the f PRINT STACK function to see how 34 and 21 are sitting vertically in the stack as shown below:

Press	Display	
f PRINT STACK	21.00	21.00 PR S 0.00 T 0.00 Z 34.00 Y 21.00 X

Since the two numbers are now sitting vertically in the stack, we can add. Add the two numbers, then print the contents of the stack again to see how the two numbers combine and the answer is seen in the displayed X-register.

Press	Display	
+	55.00	The answer.
f PRINT STACK		
		<div><div>+</div><div>PR S</div><div>0.00 T</div><div>0.00 Z</div><div>0.00 Y</div><div>55.00 X</div></div>

The simple old-fashioned math notation helps explain how to use your calculator. Both numbers are always positioned in the stack in the natural order first, then the operation is executed when the function key is pressed. *There are no exceptions to this rule.* Subtraction, multiplication, and division work the same way. In each case, the data must be in the proper position before the operation can be performed.

Chain Arithmetic

You’ve already learned how to key numbers into the calculator and perform calculations with them. In each case you first needed to position the numbers in the stack manually using the **ENTER** key. However, the stack also performs many movements automatically. These automatic movements add to its computing efficiency and ease of use, and it is these movements that automatically store intermediate results. The stack automatically “lifts” every calculated number in the stack when a new number is keyed in because it knows that after it completes a calculation, any new digits you key in are a part of a new number. Also, the stack automatically “drops” when you perform a two-number operation.

To see how it works, let’s solve

$$16 + 30 + 11 + 17 = ?$$

If you press **g** CLEAR **STACK** first, you will begin with zeros in all the stack registers, as in the example below, but of course, you can also do the calculation without first clearing the stack.

Remember, too, that you can always monitor the contents of the stack at any time by using the **f** PRINT **STACK** function.

Press	Stack Contents	
16	<div><div>T</div><div>0.00</div><div>Z</div><div>0.00</div><div>Y</div><div>0.00</div><div>X</div><div>16.</div></div>	16 is keyed into the displayed X-register.

ENTER ↵

T	0.00
Z	0.00
Y	16.00
X	16.00

16 is copied into Y.

30

T	0.00
Z	0.00
Y	16.00
X	30.

30 writes over the 16 in X.

+

T	0.00
Z	0.00
Y	0.00
X	46.00

16 and 30 are added together.
The answer, 46, is displayed.

11

T	0.00
Z	0.00
Y	46.00
X	11.

11 is keyed into the
displayed X-register.
The 46 in the stack is
automatically raised.

```

16.00 ENT↑
30.00  +
11.00  +
17.00  +
74.00 ***

```

+

T	0.00
Z	0.00
Y	0.00
X	57.00

46 and 11 are added together.
The answer, 57, is displayed.

17

T	0.00
Z	0.00
Y	57.00
X	17.

17 is keyed into the X-register
57 is automatically entered
into Y.

+

PRINT X

T	0.00
Z	0.00
Y	0.00
X	74.00

57 and 17 are added together
for the final answer.

After any calculation or number manipulation, the stack automatically lifts when a new number is keyed in. Because operations are performed when the operations are pressed, the length of such chain problems is unlimited unless a number in one of the stack registers exceeds the range of the calculator ($up\ to\ 9.999999999 \times 10^{99}$).

In addition to the automatic stack lift after a calculation, the stack automatically drops during calculations involving both the X- and Y-registers. It happened in the above example, but let's do the problem differently to see this feature more clearly. For clarity, first press **CLX** to clear the X-register. Now, again solve $16 + 30 + 11 + 17 = ?$

Press

Stack Contents

16

T

Z

Y

X

0.00

0.00

0.00

16.

16 is keyed into the displayed X-register.

ENTER

T

Z

Y

X

0.00

0.00

16.00

16.00

16 is copied into Y.

30

T

Z

Y

X

0.00

0.00

16.00

30.

30 is written over the 16 in X.

ENTER

T

Z

Y

X

0.00

16.00

30.00

30.00

30 is entered into Y.
16 is lifted up to Z.

11

T

Z

Y

X

0.00

16.00

30.00

11.

11 is keyed into the displayed X-register.

ENTER↑

T	16.00
Z	30.00
Y	11.00
X	11.00

11 is copied into Y.
16 and 30 are lifted up to
T and Z respectively.

17

T	16.00
Z	30.00
Y	11.00
X	17.

17 is written over
the 11 in X.

+

T	16.00
Z	16.00
Y	30.00
X	28.00

17 and 11 are added
together and the rest of
the stack drops. 16
drops to Z and is also
duplicated in T. 30 and
28 are ready to be
added.

```

16.00 ENT↑
30.00 ENT↑
11.00 ENT↑
17.00  +
        +
        +
        +
74.00  ***

```

+

T	16.00
Z	16.00
Y	16.00
X	58.00

30 and 28 are added
together and the stack
drops again. Now 16
and 58 are ready to be
added.

+**PRINT X**

T	16.00
Z	16.00
Y	16.00
X	74.00

16 and 58 are added
together for the final
answer and the stack
continues to drop.

The same dropping action also occurs with **-**, **×** and **=**. The number in T is duplicated in T and drops to Z, the number in Z drops to Y, and the numbers in Y and X combine to give the answer, which is visible in the X-register.

This automatic lift and drop of the stack give you tremendous computing power, since you can retain and position intermediate results in long calculations without the necessity of reentering the numbers.

Order of Execution

When you see a problem like this one:

$$5 \times [(3 \div 4) - (5 \div 2) + (4 \times 3)] \div (3 \times .213)$$

you must decide where to begin before you ever press a key.

Experienced HP calculator users have determined that by starting every problem at its innermost number or parentheses and working outward, just as you would with paper and pencil, you maximize the efficiency and power of your HP calculator. Of course, with the HP-95C you have tremendous versatility in the order of execution.

For example, you could work the problem above by beginning at the left side of the equation and simply working through it in left-to-right order. All problems cannot be solved using left-to-right order, however, and the best order for solving any problem is to begin with the innermost parentheses and work outward. So, to solve the problem above:

Press	Display		
3	3.		
ENTER	3.00		
4	4.		
÷	0.75	Intermediate answer for (3 ÷ 4).	
5	5.		
ENTER	5.00		
2	2.		
÷	2.50	Intermediate answer for (5 ÷ 2).	3.00 ENT ↑
-	-1.75	Intermediate answer for (3 ÷ 4) - (5 ÷ 2).	4.00 ÷
4	4.		5.00 ENT ↑
ENTER	4.00		2.00 ÷
3	3.		-
×	12.00	Intermediate answer for (4 × 3).	4.00 ENT ↑
+	10.25	Intermediate answer for (3 ÷ 4) - (5 ÷ 2) + (4 × 3).	3.00 ×
3	3.		+
ENTER	3.00		3.00 ENT ↑
.213	.213		.213 ×
×	0.64	Intermediate answer for (3 × .213).	÷
÷	16.04		5.00 ×
5	5.	The first number is keyed in.	80.20 ***
×	80.20	The final answer.	
PRINT	80.20		

LAST X

In addition to the four stack registers that automatically store intermediate results, the HP-95C also contains a separate automatic register, the LAST X register. This register preserves the value that was last displayed in the X-register before the performance of a function. To place the contents of the LAST X register into the display again, press **f** **LAST X**.

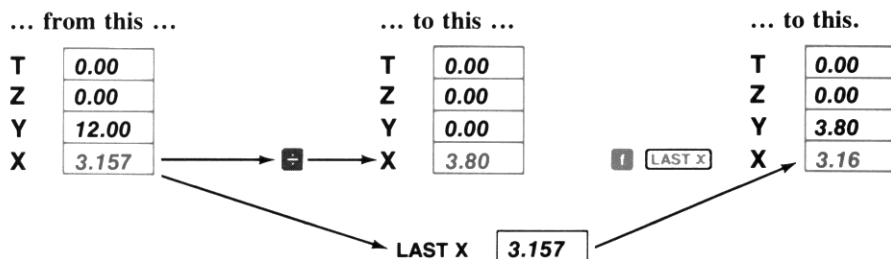
Recovering From Mistakes

LAST X makes it easy to recover from keystroke mistakes, such as pressing the wrong function key or keying in the wrong number.

Example: Divide 12 by 2.157 after you have mistakenly divided by 3.157.

Press	Display		
12	12.		
ENTER	12.00		
3.157 ÷	3.80	Oops! You made a mistake.	12.00 ENT ↑
f LAST X	3.16	Retrieves that last entry (3.157).	3.157 ÷
x	12.00	You're back at the beginning.	LSTX
2.157 ÷	5.56	The correct answer.	x
PRINT x	5.56		2.157 ÷
			5.56 ***

In the above example, when the first **÷** is pressed, followed by **f** **LAST X**, the contents of the stack and LAST X registers are changed...



This makes possible the correction illustrated in the example above.

Recovering a Number for Calculation

The LAST X register is useful in calculations where a number occurs more than once. By recovering a number using **LAST X**, you do not have to key that number into the calculator again.

Example: Calculate

$$\frac{7.32 + 3.650112331}{3.650112331}$$

Press	Display	
7.32	7.32	
ENTER	7.32	
3.650112331	3.650112331	
+	10.97	Intermediate answer.
f LAST X	3.65	Recalls 3.650112331 to X-register.
÷	3.01	The answer.
PRINT X	3.01	

```

7.32 ENT↑
3.650112331 +
LSTX
÷
3.01 ***

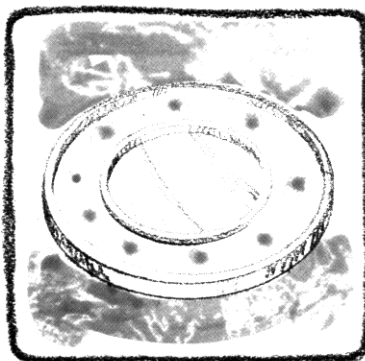
```


Constant Arithmetic

You may have noticed that whenever the stack drops because of a two-number operation (not because of **R↓**), the number in the T-register is reproduced there. This stack operation can be used to insert a constant into a problem.

Example: A bacteriologist tests a certain strain whose population typically increases by 15% each day. If he starts a sample culture of 1000, what will be the bacteria population at the end of each day for six consecutive days?

Method: Put the growth factor (1.15) in the Y-, Z-, and T-registers and put the original population (1000) in the X-register. Thereafter, you get the new population whenever you press **×**. Try working this problem with the Print Mode switch set to TRACE so that you'll have a record of all the answers without pressing **PRINT X** each time.



Slide the Print Mode switch **MAN**  **NORM** to **TRACE**.

Press	Display		
1.15	1.15	Growth factor.	
ENTER	1.15		1.15 ENT↑
ENTER	1.15		ENT↑
ENTER	1.15		ENT↑
		Growth factor now in T.	1000.00 x
1000	1000.	Starting population.	1150.00 ***
x	1150.00	Population after 1 st day.	x
x	1322.50	Population after 2 nd day.	1322.50 ***
x	1520.88	Population after 3 rd day.	x
x	1749.01	Population after 4 th day.	1520.88 ***
x	2011.36	Population after 5 th day.	x
x	2313.06	Population after 6 th day.	1749.01 ***
			2011.36 ***
			x
			2313.06 ***

When you press **x** the first time, you calculate 1.15×1000 . The result (1150.00) is displayed in the X-register and a new copy of the growth factor drops into the Y-register. Since a new copy of the growth factor is duplicated from the T-register each time the stack drops, you never have to reenter it.

Notice that performing a two-number operation such as **x** causes the number in the T-register to be duplicated there each time the stack is dropped. However, the **R↓** key, since it rotates the contents of the stack registers, does not rewrite any number, but merely shifts the numbers that are already in the stack.



PRINT
CLEAR

STACK

FIX

B

C

D

PRINT CLEAR

x^2

y^x

$1/x$

$x \leq y$ $x \geq 0$

$x \leq y$ $x < 0$

$x > y$ $x > 0$

I

GSB

SST

ISZ DSZ

RTN

BST

9

GTO

R/S

ROLL UP/DN

PAUSE

STO

STO 1

RCL

RCL 1

DELETE

LAST x

Section 4

Storing and Recalling Numbers

You have learned about the calculating power that exists in the four-register automatic memory stack and the LAST X register of your HP-95C calculator. In addition to the automatic storage of intermediate results that is provided by the stack, however, the HP-95C also contains 17 *addressable* data storage registers that are unaffected by operations within the stack. These registers allow you to manually store and recall constants or to set aside numbers for use in later calculations. Like all functions, you can use these storage registers either from the keyboard or as part of a program, and since *these* registers are part of the Continuous Memory of the HP-95C, they maintain their contents even though the calculator is turned OFF.

The diagram below shows the addressable storage registers. The addresses of the storage registers are indicated by the numbers 0 through 9, by .0 through .5, and by I.

Automatic Memory Stack

T	<input type="text"/>
Z	<input type="text"/>
Y	<input type="text"/>
X	<input type="text"/>

LAST X

Storage Registers

I	<input type="text"/>
R _{.5}	<input type="text"/>
R _{.4}	<input type="text"/>
R _{.3}	<input type="text"/>
R _{.2}	<input type="text"/>
R _{.1}	<input type="text"/>
R _{.0}	<input type="text"/>
R ₉	<input type="text"/>
R ₈	<input type="text"/>
R ₇	<input type="text"/>
R ₆	<input type="text"/>
R ₅	<input type="text"/>
R ₄	<input type="text"/>
R ₃	<input type="text"/>
R ₂	<input type="text"/>
R ₁	<input type="text"/>
R ₀	<input type="text"/>

Numbered Storage Registers

Storing Numbers

To store a displayed number in any of storage registers R_0 through R_9 .

1. Press **STO** (store).
2. Press the number key of the applicable register address (**0** through **9**).

For example, to store Avogadro's number (approximately 6.02×10^{23}) in register R_2 :

Slide the Print Mode switch to NORM  NORM if you want your printed tape to match the ones shown here.

Press	Display
6.02 EEX 23	6.02 23
STO 2	6.02000000 23
	6.02+23 S 2

Avogadro's number is now stored in register R_2 . Notice that when a number is stored, it is merely copied into the storage register, so 6.02×10^{23} also remains in the displayed X-register.

To store a displayed number in any of storage registers R_0 through R_5 .

1. Press **STO**.
2. Press the decimal point key **.**
3. Press the number key of the applicable register address (0 through 5).

For example, to store 16,495,000 (the number of persons carried daily by the Japanese National Railway) in register R_4 :

Press	Display
16495000	16495000.
STO . 4	16495000.00
	16495000.00 S.4

The number has been copied into storage register R_4 and also remains in the displayed X-register.

Recalling Numbers

Numbers are recalled from storage registers back into the displayed X-register in much the same way as they are stored. To recall a number from any of storage registers R_0 through R_9 :

1. Press **RCL** (recall).
2. Press the number key of the applicable register address (0 through 9).

For example, to recall Avogadro's number from register R_2 :

Press	Display	
RCL 2	6.02000000 23	R 2

To recall a number from any of registers $R_{.0}$ through $R_{.5}$:

1. Press **RCL**.
2. Press the decimal point key **□**.
3. Press the number key of the applicable register address (**0** through **5**).

For example, to recall the number of persons carried daily by the Japanese National Railway:

Press	Display	
RCL □ 4	16495000.00	R. 4

Recalling a number causes the stack to lift unless the preceding keystroke was **ENTER**, **CLX**, or **Σ+** (more about **Σ+** later).

The **I** Register

The **I** register has a number of special properties that make it useful in programming but these will be discussed later. When you are using the HP-95C manually, calculating from keyboard, the I-register is the most convenient storage register because you only need press **I** to recall its contents. You do not have to press **RCL** (although **RCL I** is a perfectly valid operation). To store a number in the I-register, you must press **STO I**.

Example: Three tanks have capacities in U.S. units of 2.0, 14.4, and 55.0 gallons, respectively. If 1 U.S. gallon is equivalent to 3.785 liters, what is the capacity in liters of each of the tanks?

Method: Place the conversion constant in the I-register and bring it out as required.

Press	Display		
3.785 STO I	3.79	Constant placed in I-register.	3.785 S I
2 ×	7.57	Capacity in liters of 1 st tank.	2.00 x
PRINT x	7.57		7.57 ***
14.4 I ×	54.50	Capacity in liters of 2 nd tank.	14.40 I
PRINT x	54.50		54.50 ***
55 I ×	208.18	Capacity in liters of 3 rd tank.	55.00 I
PRINT x	208.18		208.18 ***

Printing the Storage Registers

You can see the contents of all of the numbered storage registers at any time with the **PRINT** **[REG]** key. Simply press **[1]** **PRINT** **[REG]** to print a listing of the contents of all the numbered storage registers. For example, if you have worked through the examples as shown above, printing the contents of the storage registers should give you a listing like the one shown below.

Press

Display

[1] **PRINT** **[REG]** 208.18

```

PR R
0.00 ← 0
0.00 ← 1
6.02000000 ← 2
0.00 ← 3
0.00 ← 4
0.00 ← 5
0.00 ← 6
0.00 ← 7
0.00 ← 8
0.00 ← 9
0.00 ← .0
0.00 ← .1
0.00 ← .2
0.00 ← .3
16495000.00 ← .4
0.00 ← .5

```

If you want only a partial listing of storage registers, you can stop the printing of them at any time by pressing any key.

Clearing Storage Registers

Even though you have recalled the contents of a storage register into the displayed X-register, the number also remains in the storage register. You can clear primary storage registers in either of two ways:

- To replace a number in a single storage register, merely store another number there. To clear a storage register, replace the number in it with zero. For example, to clear storage register R_2 , press 0 **[STO]** 2.
- To clear *all* numbered storage registers back to zero at one time, press **[9]** **CLEAR** **[REG]**. This clears all numbered storage registers, while leaving the automatic memory stack and the I-register unchanged.

Remember that because of the Continuous Memory of the HP-95C the storage registers *retain* their contents even though the calculator is turned OFF. When you turn the calculator back ON again, you can summon and use the contents of the storage registers.

You can also clear storage registers $R_{.0}$ through $R_{.5}$ while leaving the remaining storage registers and the stack intact by using the CLEAR Σ function.

- Press **9** CLEAR Σ to clear storage registers $R_{.0}$ through $R_{.5}$ only.

Storage Register Arithmetic

Arithmetic can be performed *upon* the contents of storage registers R_0 through R_9 by pressing **STO** followed by the arithmetic function key followed in turn by the register address. For example:

Press	Result
STO + 1	Number in displayed X-register added to contents of storage register R_1 , and sum placed into R_1 ; $(r_1 + x \rightarrow R_1)$.
STO - 2	Number in displayed X-register subtracted from contents of storage register R_2 , and difference placed into R_2 ; $(r_2 - x \rightarrow R_2)$.
STO x 3	Number in displayed X-register multiplied by contents of storage register R_3 , and the product placed into R_3 ; $[(r_3)x \rightarrow R_3]$.
STO ÷ 4	Contents of storage register R_4 divided by number in displayed X-register, and quotient placed into register R_4 ; $(r_4 \div x \rightarrow R_4)$.

When storage register arithmetic operations are performed, the answer is written into the selected storage register, while the contents of the displayed X-register and the rest of the stack remain unchanged.

Notice that you can perform storage register arithmetic directly upon the contents of *only* storage registers R_0 through R_9 . You cannot perform storage register arithmetic upon registers $R_{.0}$ through $R_{.5}$ or the I-register.

Here is an example of storage register arithmetic.

Example: During harvest, farmer Flem Snopes trucks tomatoes to the cannery for three days. On Monday and Tuesday he hauls loads of 25 tons, 27 tons, 19 tons, and 23 tons, for which the cannery pays him \$55 per ton. On Wednesday the price rises to \$57.50 per ton, and Snopes ships loads of 26 tons and 28 tons. If the cannery deducts 2% of the price on Monday and Tuesday because of blight on the tomatoes, and 3% of the price on Wednesday, what is Snopes' total net income?



Method: Keep total amount in a storage register while using the stack to add tonnages and calculate amounts of loss.

Press	Display	
25 ENTER	25.00	
27 +	52.00	
19 + 23 +	94.00	Total of Monday's and Tuesday's tonnage.
55 x	5170.00	Gross amount for Monday and Tuesday.
STO 5	5170.00	Gross placed in storage register R ₅ .
2 %	103.40	Deductions for Monday and Tuesday.
STO - 5	103.40	Deductions subtracted from total in storage register R ₅ .
26 ENTER	26.00	
28 +	54.00	Wednesday's tonnage.
57.50 x	3105.00	Gross amount for Wednesday.
STO + 5	3105.00	Wednesday's gross amount added to total in storage register R ₅ .
3 %	93.15	Deduction for Wednesday.
STO - 5	93.15	Wednesday deduction subtracted from total in storage register R ₅ .
RCL 5	8078.45	Snopes' total net income from his tomatoes.
PRINT x	8078.45	

25.00 **ENT**↑

27.00 **+**

19.00 **+**

23.00 **+**

55.00 **x**

S 5

2.00 **%**

S-5

26.00 **ENT**↑

28.00 **+**

57.50 **x**

S+5

3.00 **%**


S-5

R 5

8078.45 *******

(You could also work this problem using the stack alone, but doing it as shown here illustrates how storage register arithmetic can be used to maintain and update different running totals.)

Storage Register Overflow

If you attempt a storage register arithmetic operation that would cause the magnitude of a number in any of the storage registers to exceed $9.99999999 \times 10^{99}$, the operation is performed and the HP-95C display immediately indicates SEE 6. In addition, if the Print Mode switch **MAN**  **NORM** is set to **NORM** or **TRACE**, the printer also registers the error message. When you then press any key, the error condition is cleared and the last value in the X-register before the error is again displayed. The storage register affected contains $9.99999999 \times 10^{99}$.

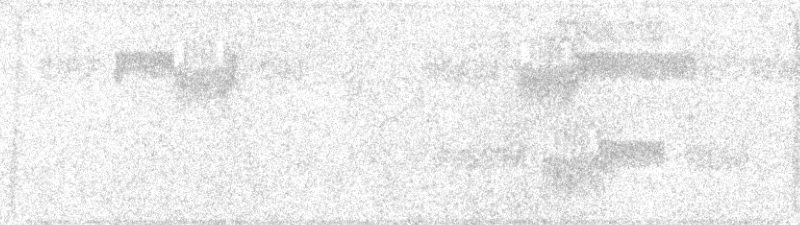
For example, if you store 7.33×10^{52} in primary register R_1 and attempt to use storage register arithmetic to multiply that value by 10^{50} , the HP-95C display will show SEE 6:

Press	Display
7.33	7.33
EE X 52	7.33 52
STO 1	7.330000000 52
EE X 50	1. 50
STO x 1	SEE 6

To clear the error and display the contents of the X-register, press any key. Storage register R_1 contains $9.999999999 \times 10^{99}$.

Press	Display
CL X	1.000000000 50 Contents of X-register.
RCL 1	9.999999999 99 Contents of storage register R_1 .

As with any error condition, pressing any key clears the error and is not executed. Pressing the paper advance pushbutton clears the error and *is* executed.



A B C D

PRINT CLEAR

\sqrt{x}

$x=y$ $x=0$

x^2

$x \neq y$ $x \neq 0$

y^x

$x \leq y$ $x < 0$

$1/x$

$x > y$ $x > 0$

%

%Σ Δ%

I

ISZ DSZ

GSB

RTN

SST

BST

f

g

GTO

LBL JUMP

R/B

PAUSE



Section 5

Function Keys





The HP-95C has dozens of internal functions that allow you to compute answers to problems quickly and accurately. Each function operates the same way, regardless of whether you press the function key manually or the function is executed as part of a program.

In this section, each function key is explained as it is used manually, with the Program Mode switch set to RUN. To save printing time and paper, you might wish to learn how to use the functions with the Print Mode switch set to MAN. Or you might wish to see every intermediate and final answer by setting the switch to TRACE. Except where indicated, however, all examples in this section are illustrated with the Print Mode switch set to NORM.



If you want your displays and printed copy to match the ones shown here, then:

TRACE
Set the Print Mode switch MAN  NORM to NORM.
Set the PRGM-RUN switch PRGM  RUN to RUN.

Number Alteration Keys

Besides , there are three keys provided for altering numbers in the HP-95C. These keys are , , and , and you will find them most useful when performing operations as part of a program.

Absolute Value


Some calculations require the absolute value, or magnitude, of a number. To obtain the absolute value of the number in the displayed X-register, press the  shift key followed by the  (*absolute value*) key. For example, to calculate the absolute value of -3 :

Press	Display
3 	$-3.$
 	$3.00 \quad -3 $
	-3.00 ABS

To see the absolute value of $+3$:

Press	Display
 	$3.00 \quad +3 $
	ABS

Integer Portion of a Number

To extract and display the integer portion of a number, press the  prefix key followed by

the **INT** (*integer*) key. For example, to display only the integers of the number 123.456:

Press	Display
123.456	123.456
f INT	123.00

Only the integer portion of the number remains.

123.456 INT

When **f** **INT** is pressed, the fractional portion of the number is lost. The entire number, of course, is preserved in the LAST X register.

Fractional Portion of a Number

To extract and display only the fractional portion of a number, press the **g** prefix key followed by the **FRAC** (*fraction*) key. For example, to see the fractional portion of the 123.456 used above:

Press	Display
f LAST X	123.46
g FRAC	0.46

Summons the original number back to the X-register.
Only the fractional portion of the number is displayed, rounded here to FIX 2 display.

LSTX
FRAC

When **g** **FRAC** is pressed, the integer portion of the number is lost. The entire number, of course, is preserved in the LAST X register.

Reciprocals

To calculate the reciprocal of a number in the displayed X-register, key in the number, then press **1/x**. For example, to calculate the reciprocal of 25:

Press	Display
25 1/x	0.04
PRINT X	0.04

25.00 1/x
0.04 ***

You can also calculate the reciprocal of a value in a previous calculation without reentering the number.

Example: In an electrical circuit, four resistors are connected in parallel. Their values are 220 ohms, 560 ohms, 1.2 kilohms, and 5 kilohms. What is the total resistance of the circuit?

$$R_T = \frac{1}{\frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_3} + \frac{1}{R_4}} = \frac{1}{\frac{1}{220} + \frac{1}{560} + \frac{1}{1200} + \frac{1}{5000}}$$

Press	Display
220 $\frac{1}{x}$	4.545454545-03
560 $\frac{1}{x}$	1.785714286-03
+	0.01
1200 $\frac{1}{x}$	8.333333333-03
+	0.01
5000 $\frac{1}{x}$	2.000000000-04
+	0.01
$\frac{1}{x}$	135.79

Sum of reciprocals.
The reciprocal of the
sum of the reciprocals
yields the answer in
ohms.

220.00 $1/x$
560.00 $1/x$
+
1200.00 $1/x$
+
5000.00 $1/x$
+
 $1/x$
135.79 ***

PRINT X

Factorials

The \boxed{NI} (factorial) key permits you to handle permutations and combinations with ease. To calculate the factorial of a positive integer in the displayed X-register, press $\boxed{9} \boxed{NI}$.

Example: Calculate the number of ways that six people can line up for a photograph.

Method: $P_6^6 = 6! = 6 \times 5 \times 4 \times 3 \times 2 \times 1$.

Press	Display
6	6.
$\boxed{9} \boxed{NI}$	720.00 The answer.
PRINT X	720.00

6.00 N!
720.00 ***

The calculator overflows for factorials of numbers greater than 69.

Square Roots

To calculate the square root of a number in the displayed X-register, press $\boxed{\sqrt{x}}$. For example, to find the square root of 16:

Press	Display
16 $\boxed{\sqrt{x}}$	4.00
PRINT X	4.00

16.00 \sqrt{x}
4.00 ***

To find the square root of the result:

Press	Display
$\boxed{\sqrt{x}}$	2.00
PRINT X	2.00

\sqrt{x}
2.00 ***

Squaring

To square a number in the displayed X-register, press **x²**. For example, to find the square of 45:

Press	Display
45 x²	2025.00
PRINT x	2025.00
	45.00 x²
	2025.00 ***

To find the square of the result:

Press	Display
x²	4100625.00
PRINT x	4100625.00
	4100625.00 x²
	4100625.00 ***

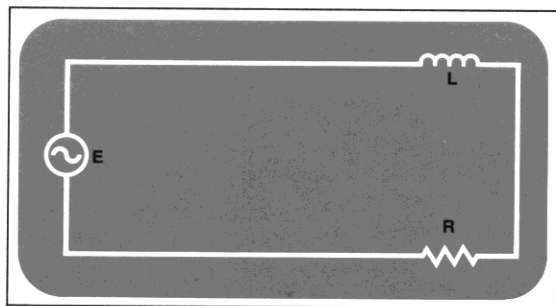
Using Pi

The value π accurate to 10 places (3.141592654) is provided as a fixed constant in the HP-95C. Merely press **9** **π** whenever you need it in a calculation. For example, to calculate 3π :

Press	Display
3 9 π x	9.42
PRINT x	9.42
	3.00 π
	9.42 ***

Example: In the schematic diagram below, X_L is 12 kilohms, R is 7 kilohms, E is 120 volts, and f is 60 Hz. Find the inductance of the coil L in henries according to the formula:

$$L = \frac{X_L}{2\pi f}.$$



$$L = \frac{X_L}{2\pi f} = \frac{12,000}{2 \times \pi \times 60}$$

Press	Display	
12 EE 3	12. 03	
ENTER \uparrow	12000.00	12.00+03 ENT \uparrow
2 \div	6000.00	2.00 \div
9 ST \div	1909.86	\div
60 \div	31.83	60.00 \div
PRINT X	31.83	31.83 ***

Henries.

Percentages

The **%** key is a two-number function which allows you to compute percentages. To find the percentage of a number:

1. Key in the base number.
2. Press **ENTER** \uparrow .
3. Key in the number representing percent rate.
4. Press **%**.

For example, to calculate a sales tax of 6.5% on a purchase of \$1500:

Press	Display	
1500 ENTER \uparrow	1500.00	Base number.
6.5	6.5	Percent rate.
%	97.50	The answer.
PRINT X	97.50	

6.5% of \$1500 is \$97.50.

In the above example, when the **%** key is pressed, the calculated answer writes over the percentage rate in the X-register, and the base number is preserved in the Y-register. The formula used is: $\frac{x \cdot y}{100} = \%$.

When you pressed **%**, the stack contents were changed...

	...from this...		...to this.
T	0.00	T	0.00
Z	0.00	Z	0.00
Y	1500.00	Y	1500.00
X	6.5	X	97.50

Since the purchase price is now in the Y-register and the amount of tax is in the X-register, the total amount can be obtained by simply adding:

Press	Display
+	1597.50 Total of price and sales tax combined.
PRINT X	1597.50
	1597.50 ***

Percent of Change

The $\Delta\%$ (percent of change) key is a two-number function that gives the percent increase or decrease from Y to X. To find the percent of change:

1. Key in the base number (usually, the number that happens first in time).
2. Press **ENTER**.
3. Key in the second number.
4. Press **9** $\Delta\%$.

The formula used is: $\frac{(x - y) 100}{y} = \Delta\%$.

Example: Find the percent of increase of your rent 10 years ago (\$70 per month) to today (\$240 per month).

Press	Display
70 ENTER	70.00
240 9 $\Delta\%$	242.86 Percent increase.
PRINT X	242.86
	70.00 ENT
	240.00 $\Delta\%$
	242.86 ***

Trigonometric Functions

Your HP-95C provides you with six trigonometric functions, which operate in decimal degrees, radians, or grads. You can convert angles between decimal degrees and *degrees minutes, seconds*, and you can add and subtract angles in any of these forms without converting them.

Trigonometric Modes

For trigonometric functions, angles can be assumed by the calculator to be in decimal degrees, radians, or grads. To select decimal degrees mode, press **9** **DEG** (*degrees*) before using a trigonometric function. To select radians mode, press **9** **RAD** (*radians*). Grads mode is selected with **9** **GRD** (*grads*).

Note: 360 degrees = 400 grads = 2π radians.

No matter what trigonometric mode you choose, the calculator will always remember it. The Continuous Memory of the HP-95C maintains the latest trigonometric mode that you have selected even though you may turn the calculator power switch OFF, then ON. Use one of the mode select functions if you wish to alter the trigonometric mode.

Functions

The six trigonometric functions provided by the calculator are:

- f **SIN** (*sine*)
- g **SIN⁻¹** (*arc sine*)
- f **COS** (*cosine*)
- g **COS⁻¹** (*arc cosine*)
- f **TAN** (*tangent*)
- g **TAN⁻¹** (*arc tangent*)

Each trigonometric function assumes that angles are in decimal degrees, radians, or grads, depending upon the trigonometric mode selected.

All trigonometric functions are one-number functions, so to use them, you key in the number, then press the function key(s).

Example 1: Find the cosine of 35°.

Press	Display	
g DEG	0.00	Degrees mode selected. (Display assumes no results remain from previous examples.)
		DEG
35	35.00	35.00 COS
f COS	0.82	0.82 ***
PRINT X	0.82	

Example 2: Find the arc sine in radians of .964.

Press	Display	
g RAD	0.82	Selects radians mode. (Results remain from previous example.)
		RAD
.964	.964	.964 SIN ⁻¹
g SIN⁻¹	1.30	1.30 ***
PRINT X	1.30	

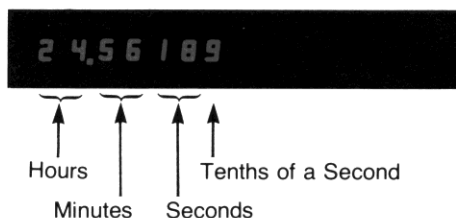
Example 3: Find the tangent of 43.66 grads.

Press	Display	
9 GRD	1.30	Selects grads mode. (Results remain from previous example.)
43.66	43.66	
f TAN	0.82	Grads.
PRINT x	0.82	

GRD
43.66 TAN
0.82 ***

Hours, Minutes, Seconds/Decimal Hours Conversions

Using the HP-95C, you can change time specified in decimal hours to *hours, minutes, seconds* format by using the **H→HMS** (*hours to hours, minutes, seconds*) function; you can also change from *hours, minutes, seconds* to decimal hours by using the **HMS→H** (*hours, minutes, seconds to hours*) function. When a time is displayed or printed in *hours, minutes, seconds* format, the digits specifying *hours* occur to the left of the decimal point, while the digits specifying *minutes, seconds, and fractions of seconds* occur to the right of the decimal point.



To convert from decimal hours to *hours, minutes, seconds*, simply key in the value for decimal hours and press **f** **H→HMS**. For example, to change 21.57 hours to *hours, minutes, seconds*:

Press	Display	
21.57	21.57	Key in the decimal time.
f FIX 4	21.5700	Reset display format.
f H→HMS	21.3412	This is 21 hours, 34 minutes, 12 seconds.
PRINT x	21.3412	

21.57 FIX4
→HMS
21.3412 ***

Notice that the display is *not* automatically switched to show you more than the normal two digits after the decimal point (**FIX** 2), so to see the digits for *seconds*, you had to reset the display format to **FIX** 4.

To convert from *hours, minutes, seconds* to decimal hours, simply key in the value for *hours, minutes, seconds* in that format and press **f** **HMS→H**. For example, to convert 132 hours, 43 minutes, and 29.33 seconds to its decimal degree equivalent:

Press	Display	
132.432933	132.432933	This is 132 hours, 43 minutes, 29.33 seconds.
f HMS→H	132.7248	This is 132.7248 hours.
PRINT x	132.7248	

Using the **H→HMS** and **HMS→H** operations, you can also convert angles specified in decimal degrees to *degrees, minutes, seconds*, and vice versa. The format for *degrees, minutes, seconds* is the same as for *hours, minutes, seconds*.

Example: Convert 42.57 decimal degrees to *degrees, minutes, seconds*.

Press	Display	
42.57	42.57	Key in the angle.
f H→HMS	42.3412	This means 42°34'12". (Display assumes FIX 4 notation remains specified from previous example.)
PRINT x	42.3412	

Example: Convert 38° 8'56.7" to its decimal equivalent.

Press	Display	
38.08567	38.08567	Key in the angle.
f HMS→H	38.1491	Answer in decimal degrees. (FIX 4 display specified from previous example.)
PRINT x	38.1491	

Adding and Subtracting Time and Angles

To add or subtract decimal hours, merely key in the numbers for the decimal hours and press **+** or **-**. To add or subtract *hours, minutes, seconds*, use the **HMS+** (*add hours, minutes, seconds*) and **HMS-** (*subtract hours, minutes, seconds*) keys.

Likewise, angles specified in *degrees, minutes, seconds* are added by pressing **f** **HMS+** and subtracted by pressing **f** **HMS-**.

Example: Find the sum of 45 hours, 10 minutes, 50.76 seconds and 24 hours, 49 minutes, 10.95 seconds.

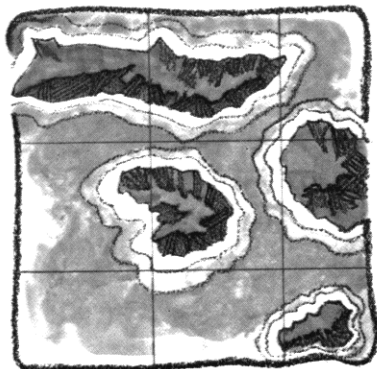
Press	Display		
45.105076 ENTER	45.105076 45.1051	FIX 4 notation from previous example.	45.105076 ENT↑
24.491095 f HMS+	24.491095 70.0002		24.491095 HMS+
f FIX 6 PRINT	70.000171 70.000171		FIX6 70.000171 ***

Example: Subtract 142.78° from $312^\circ 32' 17''$, with the answer in *degrees, minutes, seconds* format.

Press	Display		
312.3217 ENTER	312.3217 312.321700	FIX 6 from previous example.	312.321700 ENT↑
142.78 f H→HMS	142.78 142.464800	Decimal degrees. To degrees, minutes, seconds.	142.780000 →HMS
f HMS- PRINT	169.452900 169.452900	This is $169^\circ 45' 29''$.	HMS- 169.452900 ***
f FIX 2	169.45	Display mode reset.	FIX2

In the HP-95C, trigonometric functions assume angles in decimal degrees, decimal radians, or decimal grads, so if you want to compute any trigonometric functions of an angle given in *degrees, minutes, and seconds*, you must first convert the angle to decimal degrees.

Example: Lovesick sailor Oscar Odysseus dwells on the island of Tristan da Cunha ($37^\circ 03'S$, $12^\circ 18'W$), and his sweetheart, Penelope, lives on the nearest island. Unfortunately for the course of true love, however, Tristan da Cunha is the most isolated inhabited spot in the world. If Penelope lives on the island of St. Helena ($15^\circ 55'S$, $5^\circ 43'W$), use the following formula to calculate the great circle distance that Odysseus must sail in order to court her.



$$\text{Distance} = \cos^{-1} [\sin (\text{LAT}_s) \sin (\text{LAT}_d) + \cos (\text{LAT}_s) \cos (\text{LAT}_d) \\ \cos (\text{LNG}_d - \text{LNG}_s)] \times 60.$$

Where: LAT_s and LNG_s = latitude and longitude of the source (Tristan da Cunha).

LAT_d and LNG_d = latitude and longitude of the destination.

Solution: Convert all *degrees, minutes, seconds* entries into decimal degrees as you key them in. The equation for the great circle distance from Tristan da Cunha to the nearest inhabited land is:

$$\text{Distance} = \cos^{-1} [\sin (37^\circ 03') \sin (15^\circ 55') + \cos (37^\circ 03') \cos (15^\circ 55') \\ \cos (5^\circ 43' \text{W} - 12^\circ 18' \text{W})] \times 60$$

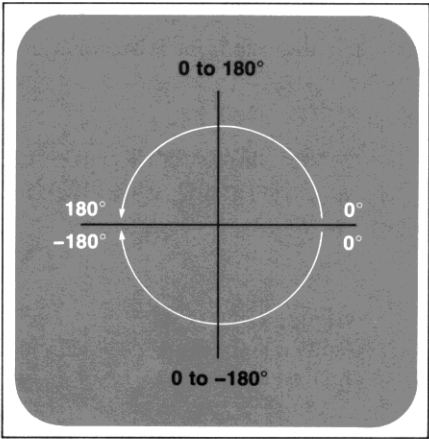
Press	Display	
g DEG	0.00	(Display assumes no results remain from previous examples.)
5.43	5.43	
f HMS→H	5.72	
12.18	12.18	
f HMS→H -	-6.58	
f COS	0.99	
15.55	15.55	
f HMS→H	15.92	
STO 1	15.92	
f COS	0.96	
x	0.96	
37.03 f HMS→H	37.05	
STO 0	37.05	
f COS	0.80	
x	0.76	
RCL 0 f SIN	0.60	
RCL 1 f SIN	0.27	
x	0.17	
+	0.93	
g COS⁻¹	21.92	
60 x PRINT x	1315.41	Distance in nautical miles that Odysseus must sail to visit Penelope.

DEG
5.43 →H
12.18 →H
-
COS
15.55 →H
S 1
COS
x
37.03 →H
S 0
COS
x
R 0
SIN
R 1
SIN
x
+
COS⁻¹
60.00 x
1315.41 ***

Polar/Rectangular Coordinate Conversions

Two functions, $\boxed{\rightarrow P}$ and $\boxed{\rightarrow R}$, are provided for polar/rectangular coordinate conversions. Polar angle θ is assumed in decimal degrees, radians, or grads, depending upon the trigonometric mode first selected by $\boxed{\text{DEG}}$, $\boxed{\text{RAD}}$, or $\boxed{\text{GRD}}$.

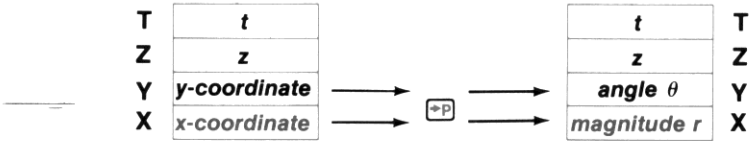
In the HP-95C, polar angle θ is represented in the following manner:



To convert from rectangular x, y coordinates to polar r, θ coordinates (magnitude and angle, respectively):

1. Key in the y -coordinate.
2. Press $\boxed{\text{ENTER}}$ to raise the y -coordinate value to the Y-register of the stack.
3. Key in the x -coordinate.
4. Press $\boxed{9} \boxed{\rightarrow P}$ (to polar). Magnitude r then appears in the X-register and angle θ is placed in the Y-register. (To display the value for θ , you press $\boxed{\text{X} \div \text{Y}}$.)

The following diagram shows how the stack contents change when you press $\boxed{\rightarrow P}$.



To convert from polar r , θ , coordinates to rectangular x , y , coordinates:

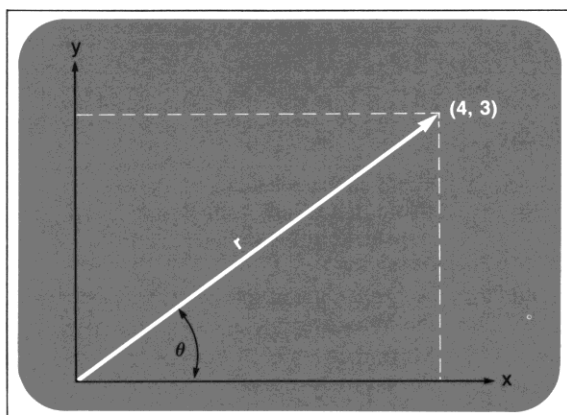
1. Key in the value for the angle θ .
2. Press **ENTER** to raise the value for θ to the Y-register of the stack.
3. Key in the value for magnitude r .
4. Press **f** **→R** (to rectangular). The x-coordinate then appears in the displayed X-register and the y-coordinate is placed in the Y-register. (To display the value for the y-coordinate, you can press **x↔y**.)

The following diagram shows how the stack contents change when you press **→R**.



After you press **→P** or **→R**, you can use the **x↔y** key to bring the calculated angle θ or the calculated y-coordinate into the X-register for viewing or further calculation.

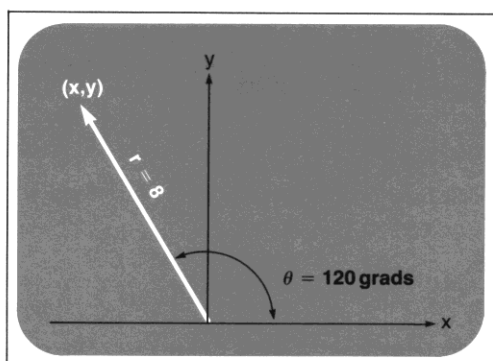
Example 1: Convert rectangular coordinates (4, 3) to polar form with the angle expressed in radians.



Press	Display	
9 RAD	0.00	Radians mode selected. (Display assumes no results remain from previous examples.)
3 ENTER	3.00	y-coordinate entered into the Y-register.
4	4.	x-coordinate keyed into the X-register.
9 →P	5.00	Magnitude r .
PRINT X	5.00	
X↔Y	0.64	Angle θ in radians.
PRINT X	0.64	

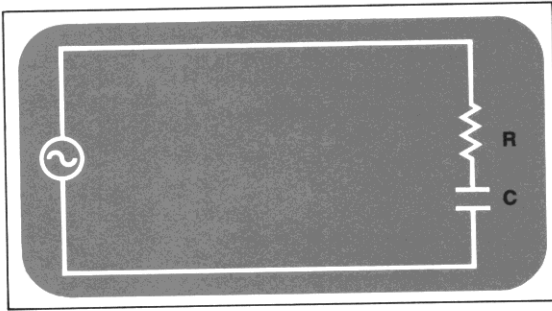
RAD
 3.00 ENT↑
 4.00 →P
 5.00 ***
 X↔Y
 0.64 ***

Example 2: Convert polar coordinates (8, 120 grads) to rectangular coordinates.

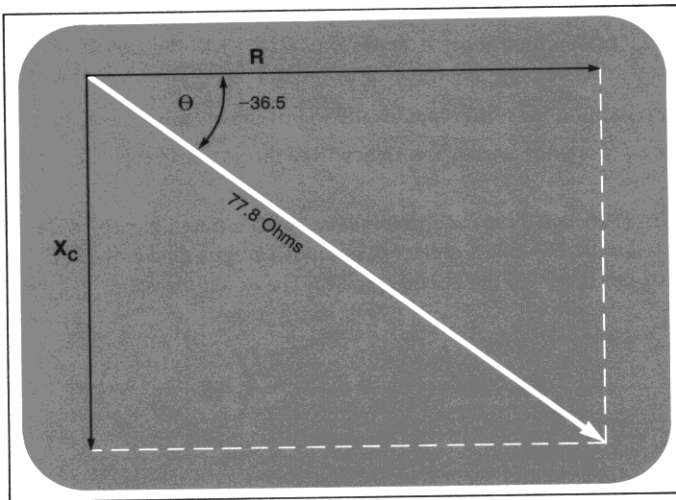


Press	Display	
9 GRD	0.64	Grads mode selected. (Note that results can remain from previous examples.)
120 ENTER	120.00	Angle θ entered into the Y-register.
8	8.	Magnitude r placed in displayed X-register.
f →R	-2.47	x-coordinate.
X↔Y	7.61	y-coordinate brought into displayed X-register for use, if desired.

GRD
 120.00 ENT↑
 8.00 →R
 X↔Y



Example 3: Engineer Tobias Slothrop has determined that in the RC circuit shown above, the total impedance is 77.8 ohms and voltage lags current by 36.5° . What are the values of resistance R and capacitive reactance X_c in the circuit?



Method: Draw a vector diagram using total impedance 77.8 ohms for polar magnitude r and -36.5° for angle θ . When the values are converted to rectangular coordinates, the x-coordinate value yields resistance R in ohms, and the y-coordinate value yields reactance X_c in ohms.

Solution:

Press	Display	
g DEG	7.61	Degrees mode selected. (Note that results can remain from previous examples.)
36.5 CHS	-36.5	
ENTER ↓	-36.50	
77.8	77.8	
f →R	62.54	Resistance R in ohms.
X_cY	-46.28	Reactance X _c , 46.28 ohms, available in displayed X-register.

DEG
-36.50 ENT↑
77.80 →R
X_cY

Logarithmic and Exponential Functions

Logarithms

The HP-95C computes both natural and common logarithms as well as their inverse functions (antilogarithms):

- f** **LN** is \log_e (natural log). It takes the log of the value in the X-register to base e (2.718...).
- g** **e^x** is antilog_e (natural antilog). It raises e (2.718...) to the power of the value in the X-register. (To display the value of e, press 1 **g** **e^x**.)
- f** **LOG** is \log_{10} (common log). It computes the log of the value in the X-register to base 10.
- g** **10^x** is antilog₁₀ (common antilog). It raises 10 to the power of the value in the X-register.

Example 1: The 1906 San Francisco earthquake, with a magnitude of 8.25 on the Richter Scale, is estimated to be 105 times greater than the Nicaragua quake of 1972. What would be the magnitude of the latter on the Richter Scale? The equation is:

$$R_1 = R_2 - \log \frac{M_2}{M_1} = 8.25 - \left(\log \frac{105}{1} \right)$$

Solution:

Press	Display	
8.25 ENTER ↓	8.25	
105 f LOG	2.02	
-	6.23	Rating on Richter Scale.
PRINT X	6.23	

8.25 ENT↑
105.00 LOG
-
6.23 ***

Example 2: Having lost most of his equipment in a blinding snowstorm, ace explorer Jason Quarmorte is using an ordinary barometer as an altimeter. After measuring the sea level pressure (30 inches of mercury) he climbs until the barometer indicates 9.4 inches of mercury. Although the exact relationship of pressure and altitude is a function of many factors, Quarmorte knows that an *approximation* is given by the formula:

$$\begin{aligned}\text{Altitude (feet)} &= 25,000 \ln \frac{30}{\text{Pressure}} \\ &= 25,000 \ln \frac{30}{9.4}\end{aligned}$$



Where is Jason Quarmorte?

Solution:

Press	Display	
30 ENTER \blacktriangledown	30.00	30.00 ENT \blacktriangle
9.4 =	3.19	9.40 \div
f LN	1.16	LN
25000	25000	25000.00 \times
x	29012.19 Altitude in feet.	29012.19 ***
PRINT x	29012.19	

Quarmorte is probably near the summit of Mount Everest (29,028 feet).

Raising Numbers to Powers

The **y^x** key is used to raise numbers to powers. Using **y^x** permits you to raise a positive real number to any real power—that is, the power may be positive or negative, and it may be an integer, a fraction, or a mixed number. **y^x** also permits you to raise any negative real number to the power of any integer (within the calculating range of the HP-95C, of course).

For example, to calculate 2^9 (that is, $2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2$):

Press	Display	
2 ENTER \blacktriangledown 9	9.	2.00 ENT \blacktriangle
y^x	512.00	9.00 y^x
PRINT x	512.00	512.00 ***

To calculate $8^{-1.2567}$:

Press	Display	
8 ENTER	8.00	8.00 ENT
1.2567 CHS	-1.2567	-1.2567 Y^x
Y^x	0.07	0.07 ***
PRINT X	0.07	

To calculate $(-2.5)^5$:

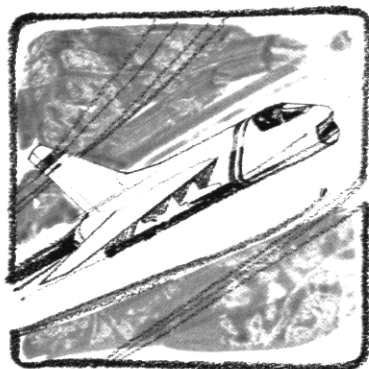
Press	Display	
2.5 CHS	-2.5	-2.50 ENT
ENTER	-2.50	5.00 Y^x
5 Y^x	-97.66	-97.66 ***
PRINT X	-97.66	

In conjunction with **1/x**, **Y^x** provides a simple way to extract roots. For example, find the cube root of 5. (This is equivalent to $5^{1/3}$.)

Press	Display	
5 ENTER	5.00	5.00 ENT
3 1/x	0.33	3.00 1/x
Y^x		Y^x
PRINT X	1.71	1.71 ***
		Cube root of 5.

Example: In a rather overoptimistic effort to break the speed of sound, highflying pilot Ike Daedalus cranks open the throttle on his surplus Hawker Siddeley Harrier aircraft. From his instruments he reads a pressure altitude (PALT) of 25,500 feet with a calibrated airspeed (CAS) of 350 knots. What is the flight mach number

$$M = \frac{\text{speed of aircraft}}{\text{speed of sound}}$$



if the following formula is applicable?

$$M = \sqrt{5 \left[\left(\left(\left(1 + 0.2 \left[\frac{350}{661.5} \right]^2 \right)^{3.5} - 1 \right) \left[1 - (6.875 \times 10^{-6}) 25,500 \right]^{-5.2656} \right) + 1 \right)^{0.286} - 1 \right]}$$

Method: The most efficient place to begin work on this problem is at the innermost set of brackets. So begin by solving for the quantity $\left[\frac{350}{661.5}\right]^2$ and proceed outward from there.

Press	Display		
350 ENTER	350.00		
661.5 ÷	0.53		
x²	0.28	Square of bracketed quantity.	350.00 ENT ↑
.2 x 1 +	1.06		661.50 ÷
3.5 y^x 1 -	0.21	Contents of left-hand set of brackets are in the stack.	x²
			.20 x
			1.00 +
			3.50 y^x
			1.00 -
1 ENTER	1.00		1.00 ENT ↑
6.875 EE^x	6.875 00		6.875-06 ENT ↑
CHS 6 ENTER	6.87500000-06		25500.00 x
25500 x -	0.82		-
5.2656 CHS	-5.2656	Contents of right-hand set of brackets are in the stack.	-5.2656 y^x
			x
y^x	2.76		1.00 +
x 1 +	1.58		.286 y^x
.286 y^x	1.14		1.00 -
1 -	0.14		5.00 x
5 x √x	0.84	Mach number of Daedalus' Harrier.	√x
			0.84 ***
PRINT x	0.84		

In working through complex equations like the one containing six levels of parentheses above, you really appreciate the value of the Hewlett-Packard logic system. Because you calculate one step at a time, you don't get "lost" within the problem. You see every intermediate result, and you emerge from the calculation confident of your final answer.

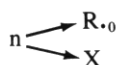
Statistical Functions

Accumulations

Pressing the **Σ+** key automatically gives you several different sums and products of the values in the X- and Y-registers at once. In order to make these values accessible for sophisticated statistics problems, they are automatically placed by the calculator into storage registers R₀ through R₅. The only time that information is automatically accumulated in the storage registers is when the **Σ+** (or **Σ-**) key is used. Before you begin any calculations using the **Σ+** key, you should first clear the storage registers used in accumulations by pressing **9** **CLEAR** **Σ**.

When you key a number into the display and press the $\Sigma+$ key, each of the following operations is performed:

1. The number that you keyed into the X-register is added to the contents of storage register $R_{.1}$. ($\Sigma x \rightarrow R_{.1}$)
2. The square of the number that you keyed into the X-register is added to the contents of storage register $R_{.2}$. ($\Sigma x^2 \rightarrow R_{.2}$)
3. The number in the Y-register of the stack is added to the contents of storage register $R_{.3}$. ($\Sigma y \rightarrow R_{.3}$)
4. The square of the number in the Y-register of the stack is added to the contents of storage register $R_{.4}$. ($\Sigma y^2 \rightarrow R_{.4}$)
5. The number that you keyed into the X-register is multiplied by the contents of the Y-register, and the product added to storage register $R_{.5}$. ($\Sigma xy \rightarrow R_{.5}$)
6. The number 1 is added to storage register $R_{.0}$, and the total number in $R_{.0}$ then writes over the number in the displayed X-register of the stack. The stack does not lift.



The number that you keyed into the X-register is preserved in the **LAST X** register, while the number in the stack Y-register remains in the Y-register.

Thus, when you press $\Sigma+$, the stack and storage register contents are changed...

...from this...

T	t		$R_{.5}$
Z	z		$R_{.4}$
Y	y		$R_{.3}$
X	x		$R_{.2}$
			$R_{.1}$
			$R_{.0}$

LAST X

...to this.

T	t	Σxy	$R_{.5}$
Z	z	Σy^2	$R_{.4}$
Y	y	Σy	$R_{.3}$
X	n	Σx^2	$R_{.2}$
		Σx	$R_{.1}$
		n	$R_{.0}$

LAST X

Just as in storage register arithmetic, if you attempt an operation which would cause the magnitude of a number in any of the storage registers used for $\Sigma+$ accumulations to be larger than $9.99999999 \times 10^{99}$, the operation is performed and the HP-95C display indicates **SEE 6** to indicate storage register overflow. Press any key to clear the error display.

To use *only* the Σx and Σy that you have accumulated in the storage registers, you can press **RCL** followed by $\Sigma+$. This brings Σx into the displayed X-register and Σy into the Y-register, overwriting the contents of those two stack registers. The stack does not lift. (This feature is particularly useful when performing vector arithmetic, like that illustrated on pages 99-101.)

To use *any* of the summations individually at any time, you can recall the contents of the desired storage register into the displayed X-register by pressing **RCL** followed by the number key of the storage register address. After you have pressed **Σ+**, recalling storage register contents or keying in another number writes over the number of entries (*n*) that is displayed. The stack does not lift.

Example: Find Σx , Σx^2 , Σy , Σy^2 , and Σxy for the paired values of *x* and *y* listed below.

<i>y</i>	7	5	9
<i>x</i>	5	3	8

Press	Display	
9 CLEAR Σ	0.00	Ensures that storage registers R ₀ through R ₅ are cleared to zero initially. Display assumes no results remain from previous example.
7 ENTER +	7.00	
5 Σ+	1.00	First pair is accumulated; <i>n</i> = 1.
5 ENTER +	5.00	
3 Σ+	2.00	Second pair is accumulated; <i>n</i> = 2.
9 ENTER +	9.00	
8 Σ+	3.00	Third pair is accumulated; <i>n</i> = 3.
RCL 1	16.00	Sum of <i>x</i> values from register R ₁ .
RCL 2	98.00	Sum of squares of <i>x</i> values from register R ₂ .
RCL 3	21.00	Sum of <i>y</i> values from register R ₃ .
RCL 4	155.00	Sum of squares of <i>y</i> values from register R ₄ .
RCL 5	122.00	Sum of products of <i>x</i> and <i>y</i> values from register R ₅ .
RCL 0	3.00	Number of entries (<i>n</i> = 3) from register R ₀ .

CL Σ
7.00 ENT↑
5.00 Σ+
5.00 ENT↑
3.00 Σ+
9.00 ENT↑
8.00 Σ+
R.1
R.2
R.3
R.4
R.5
R.0

Printing Accumulations

You can see *all* of the values accumulated by the **Σ+** key at any time. Simply press **1** **PRINT** **Σ**, and the printer will print out the contents of the storage registers used for summations along with a description for each summation.

For example, to list *all* of the accumulations that are now in the storage registers from the previous example:

Press **Display**

f PRINT **Σ** 3.00

```

PR Σ
3.00 N
16.00 ΣX
98.00 ΣX²
21.00 ΣY
155.00 ΣY²
122.00 ΣXY

```

Percent of Sum

The **%Σ** (*percent of sum*) function permits you to compute the percentage that several values are of a total, while leaving the total intact. **f** **%Σ** computes the percentage the number in the X-register is of the value in storage register R₁. The formula used is:

$$\frac{x}{\Sigma x} \times 100 = \% \Sigma$$

The computed value for **%Σ** writes over the number in the X-register, and the rest of the stack remains unchanged. (*x* is, of course, preserved in the LAST X register.)

You will probably want to accumulate the total value in register R₁ using the **Σ+** key before pressing **f** **%Σ**. (You could also accumulate a value in register R₁ manually, by simply storing the value there using the **STO** key.)

Example: A compound is made up of 5.4 grams of hydrogen (H), 172.8 grams of oxygen (O) and 866.7 grams of sulfur (S). What is the percentage by weight of each chemical in the compound, and what is the total weight of the compound?

Press **Display**

9 CLEAR **Σ** 0.00

Display assumes no results remain from previous example.

5.4 **STO** 1 **Σ+** 1.00

172.8 **STO** 2 2.00

Σ+ 2.00

866.7 **STO** 3 3.00

Σ+ 3.00

RCL 1 **f** **%Σ** 0.52

Percent H is of total weight.

```

CL Σ
5.40 S 1
Σ+
172.80 S 2
Σ+
866.70 S 3
Σ+
R 1
%Σ

```


PRINT Σ	0.52	
RCL 2 Σ	16.54	Percent O is of total weight.
PRINT Σ	16.54	
RCL 3 Σ	82.95	Percent S is of total weight.
PRINT Σ	82.95	
RCL Σ 1	1044.90	Total weight of the compound.
PRINT Σ	1044.90	

0.52 ***
R 2
 Σ
16.54 ***
R 3
 Σ
82.95 ***
R. 1
1044.90 ***

Mean

The Σ (mean) key is the key you use to calculate the mean (arithmetic average) of x and y accumulated in registers $R_{.1}$ and $R_{.3}$, respectively.

When you press Σ :

1. The mean (\bar{x}) of x is calculated using the data accumulated in register $R_{.1}$ (Σx) and $R_{.0}$ (n) according to the formula:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i \quad \left(\text{That is, } \frac{R_{.1}}{R_{.0}} = \bar{x} \right)$$

The resultant value for \bar{x} is seen in the displayed X-register.

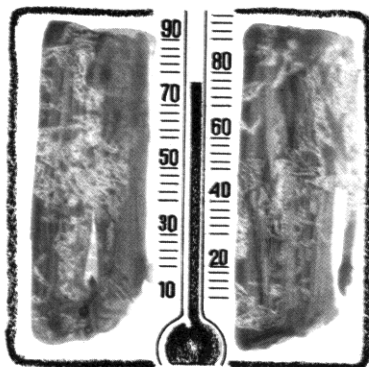
2. The mean (\bar{y}) of y is calculated using the data accumulated in register $R_{.3}$ (Σy) and register $R_{.0}$ (n) according to the formula:

$$\bar{y} = \frac{1}{n} \sum_{i=1}^n y_i \quad \left(\text{That is, } \frac{R_{.3}}{R_{.0}} = \bar{y} \right)$$

The resultant value for \bar{y} is available in the Y-register of the stack.

The easiest way to accumulate the required data in the applicable registers is through the use of the Σ key as described above.

Example: Below is a chart of daily high and low temperatures for a winter week in Fairbanks, Alaska. What are the *average* high and low temperatures for the week selected?

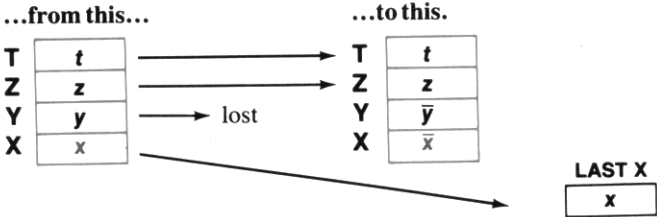


	Sun	Mon	Tues	Wed	Thurs	Fri	Sat
High	6	11	14	12	5	-2	-9
Low	-22	-17	-15	-9	-24	-29	-35

Press	Display		
9 CLEAR 2	0.00	Accumulation registers cleared. (Display assumes no results remain from previous calculations.)	
6 ENTER 22 CHS Σ+	1.00	Number of data pairs (n) is now 1.	CL Σ 6.00 ENT ↑ -22.00 Σ+
11 ENTER 17 CHS Σ+	2.00	Number of data pairs (n) is now 2.	11.00 ENT ↑ -17.00 Σ+
14 ENTER 15 CHS Σ+	3.00		14.00 ENT ↑ -15.00 Σ+
12 ENTER 9 CHS Σ+	4.00		12.00 ENT ↑ -9.00 Σ+
5 ENTER 24 CHS Σ+	5.00		5.00 ENT ↑ -24.00 Σ+
2 CHS ENTER 2	-2.00		-2.00 ENT ↑
29 CHS Σ+	6.00		-29.00 Σ+
9 CHS ENTER 9	-9.00		-9.00 ENT ↑
35 CHS Σ+	7.00	Number of data pairs (n) is now 7.	-35.00 Σ+
f Σ	-21.57	Average low temperature.	Σ -21.57 *** Σ → Y
PRINT X	-21.57		5.29 ***
Σ → Y	5.29	Average high temperature.	
PRINT X	5.29		

As shown, you can use the **PRINT X** and **Σ**→**Y** keys to print the values for \bar{x} and \bar{y} .

The illustrations below represent what happens in the stack when you press **f** **Σ**. Press **f** **Σ** and the contents of the stack registers are changed...



Standard Deviation

The \boxed{S} (*standard deviation*) key is the key you use to calculate the standard deviation (a measure of dispersion around the mean) of data accumulated in storage registers $R_{.0}$ through $R_{.5}$.

When you press $\boxed{g} \boxed{S}$:

1. Sample x standard deviation (s_x) is calculated using the data accumulated in storage register $R_{.2}$ (Σx^2), $R_{.1}$ (Σx), and $R_{.0}$ (n) according to the formula:

$$s_x = \sqrt{\frac{\Sigma x^2 - \frac{(\Sigma x)^2}{n}}{n - 1}}$$

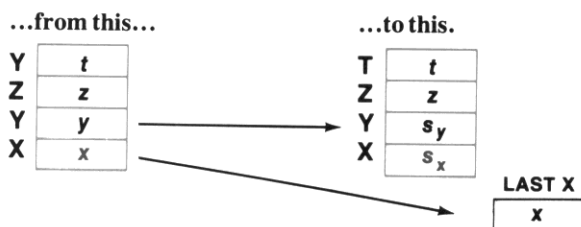
The resultant value for standard deviation of x (s_x) is seen in the displayed X-register.

2. Sample y standard deviation (s_y) is calculated using the data accumulated in storage registers $R_{.4}$ (Σy^2), $R_{.3}$ (Σy), and $R_{.0}$ (n) according to the formula:

$$s_y = \sqrt{\frac{\Sigma y^2 - \frac{(\Sigma y)^2}{n}}{n - 1}}$$

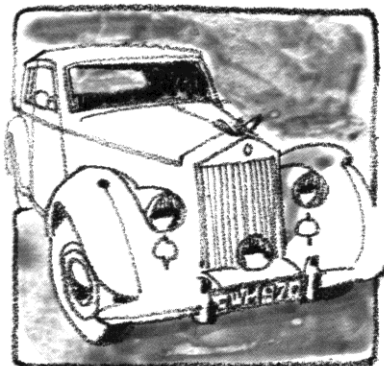
The resultant value for standard deviation of y (s_y) is available in the Y-register of the stack.

Thus, with data accumulated in registers $R_{.0}$ through $R_{.5}$, when you press $\boxed{g} \boxed{S}$, the contents of the stack registers are changed...



To use the value for standard deviation of y (s_y) simply use the $\boxed{x \div y}$ key to bring that value into the displayed X-register of the stack.

Example: In a recent survey to determine the age and net worth (in millions of dollars) of six of the 50 wealthiest persons in the United States, the following data were obtained (sampled). Calculate the average age and net worth of the sample, and calculate the standard deviations for these two sets of data.



Age	62	58	62	73	84	68
Net Worth	1200	1500	1450	1950	1000	1750

Press	Display		
9 CLEAR Σ	0.00	Clears storage registers used for Σ+ . (Display assumes no results remain from previous examples.)	
62 ENTER ↑	62.00		CL Σ
1200 Σ+	1.00	Number of data pairs (n) is 1.	62.00 ENT ↑
58 ENTER ↑	58.00		1200.00 Σ+
1500 Σ+	2.00		58.00 ENT ↑
62 ENTER ↑	62.00		1500.00 Σ+
1450 Σ+	3.00		62.00 ENT ↑
73 ENTER ↑	73.00		1450.00 Σ+
1950 Σ+	4.00		73.00 ENT ↑
84 ENTER ↑	84.00		1950.00 Σ+
1000 Σ+	5.00		84.00 ENT ↑
68 ENTER ↑	68.00		1000.00 Σ+
1750 Σ+	6.00	Number of data pairs (n) is 6.	68.00 ENT ↑
f Σ	1475.00	Average value of net worth.	1750.00 Σ+
x Σ y	67.83	Average age of the sample.	Σ
9 S	347.49	Standard deviation (s_x) of net worth of sample.	x Σ y
x Σ y	9.52	Standard deviation (s_y) of age of sample.	

If the six persons used in the sample were actually the *six wealthiest persons*, the data would have to be considered as a population rather than as a sample. The relationship between sample standard deviation (s) and the population standard deviation (σ) is illustrated by the following equation.

$$\sigma = s \sqrt{\frac{n - 1}{n}}$$

Since n is automatically accumulated in register R_0 when data is accumulated, it is a simple matter to convert the sample standard deviations that have already been calculated to population standard deviations.

If the accumulations are still intact from the previous example in registers R_0 through R_5 , you can calculate the population standard deviations this way:

Press	Display	
$\boxed{9} \boxed{S}$	347.49	Calculate s_x and s_y .
$\boxed{RCL} \boxed{0}$	6.00	Recall n .
$\boxed{1} \boxed{-}$	5.00	Calculate $n - 1$.
$\boxed{RCL} \boxed{0} \boxed{\div}$	0.83	Divide $n - 1$ by n .
$\boxed{\sqrt{x}} \boxed{x}$	317.21	Population standard deviation σ_x .
$\boxed{PRINT} \boxed{x}$	317.21	
$\boxed{x} \boxed{y}$	9.52	Brings s_y to the X-register.
$\boxed{f} \boxed{LAST X}$	0.91	Recall conversion factor.
\boxed{x}	8.69	Population standard deviation σ_y .
$\boxed{PRINT} \boxed{x}$	8.69	

S
 $R.0$
 1.00 $-$
 $R.0$
 \div
 JX
 x
 317.21 $***$
 $X \div Y$
 $LSTX$
 x
 8.69 $***$

Deleting and Correcting Data

If you key in an incorrect value and have not pressed $\boxed{\Sigma+}$, press \boxed{CLX} and key in the correct value.

If one of the values is changed, or if you discover after you have pressed the $\boxed{\Sigma+}$ key that one of the values is in error, you can correct the summations by using the $\boxed{\Sigma-}$ (summa-
minus) key as follows:

1. Key in the *incorrect* data pair into the X- and Y-registers. (You can use $\boxed{LAST X}$ to return a single incorrect data value to the displayed X-register.)
2. Press $\boxed{f} \boxed{\Sigma-}$ to delete the incorrect data.
3. Key in the correct values for x and y . (If one value of an x, y data pair is incorrect, both values must be deleted and reentered.)
4. Press $\boxed{\Sigma+}$.

The correct values for mean and standard deviation are now obtainable by pressing $\boxed{f} \boxed{x}$ and $\boxed{9} \boxed{S}$.

For example, suppose the 58-year old member of the *sample* as given above were to lose his position as one of the wealthiest persons because of a series of ill-advised investments in cocoa futures. To account for the change in data if he were replaced in the sample by

a 21-year old rock musician who is worth 1300 million dollars:

Press	Display		
58 ENTER ↑	58.00		
1500	1500.	Data to be replaced.	
f Σ-	5.00	Number of entries (n) is now five.	58.00 ENT ↑ 1500.00 Σ- 21.00 ENT ↑ 1300.00 Σ+
21 ENTER ↑	21.00		
1300	1300.		
Σ+	6.00	Number of entries (n) is now six again.	

The new data has been calculated into each of the summations present in the storage registers. To see the new mean and standard deviation:

Press	Display		
f Σ̄	1441.67	The new average (mean) worth.	
x ↔ y	61.00	The new average (mean) age available in X-register for use.	\bar{x} $x \leftrightarrow y$
g S	354.14	The new standard deviation for worth.	s $x \leftrightarrow y$
x ↔ y	21.55	The new standard deviation for age available in X-register for use.	

Linear Regression

Linear regression is a statistical method for finding a straight line that best fits a set of data points, thus providing a relationship between two variables. After a group of data points has been totaled in registers R₀ through R₅, you can calculate the coefficients of the linear equation $y = A + Bx$ using the least squares method by pressing **f** **L.R.**. (Naturally, at least two data points must be in the calculator before a least squares line can be fitted to them.)

To use the linear regression function on your HP-95C, first key in a series of data points using the **Σ+** key. Then press **f** **L.R.**.

When you press **f** **L.R.**, two values are calculated:

1. The y-intercept (A) of the least squares line of the data is calculated using the equation:

$$A = \frac{\sum y \sum x^2 - \sum x \sum xy}{n \sum x^2 - (\sum x)^2}$$

The y-intercept (A) appears in the displayed X-register of the stack.

2. The slope (B) of the least squares line of the data is calculated using the equation:

$$B = \frac{n \sum xy - \sum x \sum y}{n \sum x^2 - (\sum x)^2}$$

The slope (B) is available in the Y-register of the stack.

Thus, when you press **f** **[L.R.]**, the contents of the stack registers change...

...from this...

T	<i>t</i>
Z	<i>z</i>
Y	<i>y</i>
X	<i>x</i>

...to this.

T	<i>t</i>
Z	<i>z</i>
Y	B
X	A

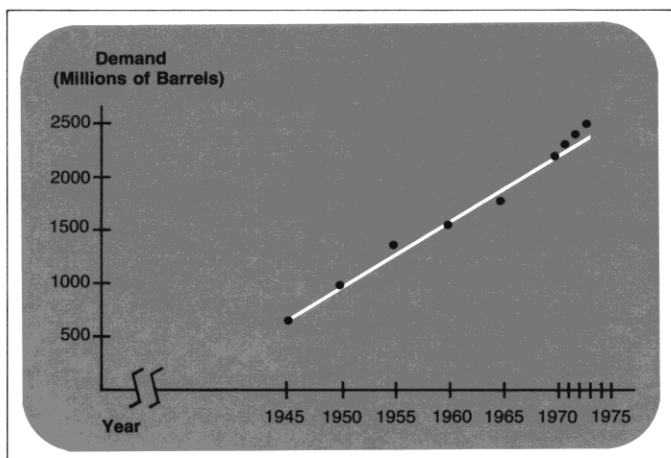
To use the value for B or to bring it into the displayed X-register, simply shift the stack contents with the **x↔y** key.

Example: Big Lyle Hephaestus, owner-operator of the Hephaestus Oil Company, wishes to know the slope and y-intercept of a least squares line for the consumption of motor fuel in the United States against time since 1945. He knows the data given in the table.

Motor Fuel Demand

(Millions of Barrels)	696	994	1330	1512	1750	2162	2243	2382	2484
Year	1945	1950	1955	1960	1965	1970	1971	1972	1973

Solution: Hephaestus *could* draw a plot of motor fuel demand against time like the one shown below.



However, with his HP-95C, Hephaestus has only to key the data into the calculator using the Σ^+ key, then press L.R. .

Press	Display	
g CLEAR Σ	0.00	Storage registers used for summations are cleared.
696 $\text{ENTER} \downarrow$	696.00	
1945 Σ^+	1.00	
994 $\text{ENTER} \downarrow$	994.00	
1950 Σ^+	2.00	
1330 $\text{ENTER} \downarrow$	1330.00	
1955 Σ^+	3.00	
1512 $\text{ENTER} \downarrow$	1512.00	
1960 Σ^+	4.00	
1750 $\text{ENTER} \downarrow$	1750.00	
1965 Σ^+	5.00	
2162 $\text{ENTER} \downarrow$	2162.00	
1970 Σ^+	6.00	
2243 $\text{ENTER} \downarrow$	2243.00	
1971 Σ^+	7.00	
2382 $\text{ENTER} \downarrow$	2382.00	
1972 Σ^+	8.00	
2484 $\text{ENTER} \downarrow$	2484.00	
1973 Σ^+	9.00	
		All data pairs have been keyed in.
f L.R.	-118290.63	The y-intercept of the line.
$\text{x} \div \text{y}$	61.16	Slope of the line.

CL Σ 696.00 ENT \uparrow 1945.00 Σ^+ 994.00 ENT \uparrow 1950.00 Σ^+ 1330.00 ENT \uparrow 1955.00 Σ^+ 1512.00 ENT \uparrow 1960.00 Σ^+ 1750.00 ENT \uparrow 1965.00 Σ^+ 2162.00 ENT \uparrow 1970.00 Σ^+ 2243.00 ENT \uparrow 1971.00 Σ^+ 2382.00 ENT \uparrow 1972.00 Σ^+ 2484.00 ENT \uparrow 1973.00 Σ^+

LR

 $\text{x} \div \text{y}$

Linear Estimate

With the data totaled in registers $R_{.0}$ through $R_{.5}$, a predicted y (that is, a \hat{y}) can be calculated by keying in a new x -value and pressing g \hat{y} .

For example, with data intact from the previous example in registers $R_{.0}$ through $R_{.5}$, if Hephaestus wishes to predict the demand for motor fuel for the years 1980 and 2000, he has only to key in the new x -values and press g \hat{y} .

Press	Display	
1980 g \hat{y}	2808.63	Predicted demand in millions of barrels for the year 1980.
2000 g \hat{y}	4031.86	Predicted demand in millions of barrels for the year 2000.

1980.00 \hat{y} 2000.00 \hat{y}

Coefficient of Determination

To establish how well the data fits the linear regression, you may want to calculate the coefficient of determination (r^2). The coefficient of determination is a value between 0 and 1. At $r = 0$ you have no fit, while at $r^2 = 1$ you have a perfect fit. The traditional equation for r^2 is:

$$r^2 = \frac{[\sum (x - \bar{x}) (y - \bar{y})]^2}{[\sum (x - \bar{x})^2] [\sum (y - \bar{y})^2]}$$

On your HP-95C, however, the most efficient way to calculate r^2 is to use this equivalent equation:

$$r^2 = \left[\frac{n \sum xy - \sum x \sum y}{n (n - 1) s_x s_y} \right]^2$$

Example: Calculate r^2 for the previously calculated linear regression.

Press	Display	
$\frac{1}{x}$ $\frac{1}{y}$	-118290.63	LR
ENTER	-118290.63	ENT↑
$\frac{1}{x}$ $\frac{1}{y}$	10.37	S
\div	61.59	\div
\div	0.99	\div
x^2 PRINT x	0.99	x^2
		0.99 ***

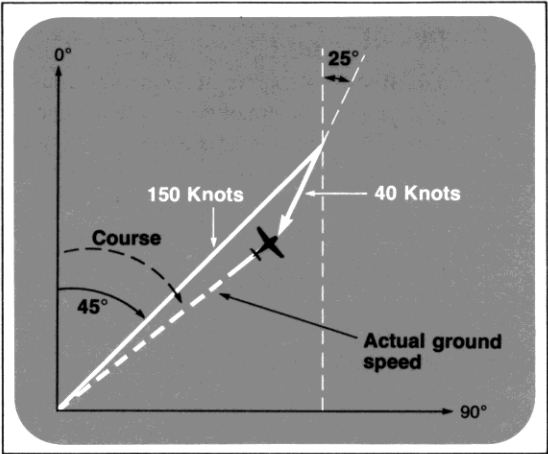
Since the correlation coefficient is 0.99, you can assume the fit of the line is excellent.

Vector Arithmetic

You can use your HP-95C to add or subtract vectors by combining the polar/rectangular conversion functions (the $\rightarrow P$ and $\rightarrow R$ keys) with the summation functions (the $\Sigma+$ and $\Sigma-$ keys).

Example: Grizzled bush pilot Apeneck Sweeney's converted Swordfish aircraft has a true air speed of 150 knots and an estimated heading of 45° . The Swordfish is also being buffeted by a headwind of 40 knots from a bearing of 25° . What is the actual ground speed and course of the Swordfish?

Method: The course and ground speed are equal to the difference of the vectors. (North becomes the x-coordinate so that the problem corresponds with navigational convention.)



Press	Display		
9 CLEAR ☐	0.00	Clears summation registers R ₀ through R ₅ . (Display assumes no results remain from previous examples.)	
9 DEG	0.00	Sets degrees mode.	
45 ENTER +	45.00	θ for 1 st vector is entered to Y-register.	
150	150.	r for 1 st vector is keyed in.	
f ☐	106.07	Converted to rectangular coordinates.	
☐ ☐	1.00	1 st vector coordinates accumulated in storage registers R ₁ and R ₃ .	
25 ENTER +	25.00	θ for 2 nd vector is entered to Y-register.	
40	40.	r for 2 nd vector is keyed in.	
f ☐	36.25	2 nd vector is converted to rectangular coordinates.	
f ☐	0.00	2 nd vector rectangular coordinates subtracted from those of 1 st vector.	
RCL ☐ ☐	69.81	Recalls both R ₁ and R ₃ .	

CL Σ

DEG

45.00 ENT \uparrow

150.00 $\rightarrow R$

$\Sigma +$

25.00 ENT \uparrow

40.00 $\rightarrow R$

$\Sigma -$

R Σ



PRINT X

X \leftrightarrow Y

PRINT X

113.24

Actual ground speed in
knots of the Swordfish.

113.24

51.94

Course in degrees of
the Swordfish.

51.94

 $\rightarrow P$

113.24 ***

X \leftrightarrow Y

51.94 ***

Part Two
Programming The HP-95C



Simple Programming

If you read *Meet the HP-95C* (pages 13–17), you have already seen that by using the programming capability of your HP-95C, you can increase the flexibility of the calculator a hundredfold or more, and you save hours of time in long computation. The Continuous Memory of the HP-95C permits you to key in up to four programs and save them permanently, even though the HP-95C power switch is turned OFF.

With your HP-95C Scientific Calculator with Partition Programming, Hewlett-Packard has provided you with the *HP-95C Applications Book*, containing dozens of programs. You can begin using the programming power of the HP-95C immediately by keying any of these programs into your HP-95C and then running it as often as you like.

However, in order to get the most from your calculator, you'll want to learn to program the HP-95C yourself, to solve your own problems. This part of the *HP-95C Owner's Handbook and Programming Guide* teaches you step by step to create simple programs that will solve complex problems, then introduces you to the many editing features of the HP-95C, and finally gives you a glimpse of just how sophisticated your programming can become with the HP-95C Calculator.

Programming your calculator is an extension of its use as a *manual* problem-solving machine, so if you haven't read Part One, *Using Your HP-95C Calculator*, you should go back and do so before you begin programming.

After most of the explanations and examples in this part, you will find problems to work using your HP-95C. These problems are not essential to your basic understanding of the calculator, and they can be skipped if you like. But we urge that you work them. They are rarely difficult, and they have been designed to increase your proficiency, both in the actual use of the features on your calculator and in creating programs to solve your *own* problems. If you have trouble with one of the problems, go back and review the explanations in the text, then tackle it again.

So that you can apply your own creative flair to the problems, no solutions are given for them. In programming, any solution that gives the correct outputs is the right one—there is no *one* correct program for any problem. In fact, when you have finished working through this part, and learned all the capabilities of the HP-95C, you may be able to create programs that will solve many of the problems faster, or in fewer steps, than we have shown in our illustrations.

Now let's start programming!

What Is a Program?

A *program* is nothing more than a series of calculator keystrokes that you would press to solve a problem manually. The calculator remembers these keystrokes when you key them in, then executes them in order at the press of a single key. If you want to execute the program again and again, you have only to press the single key each time.

If you worked through Meet the HP-95C (pages 13–17), you learned how to write, load, and run a simple program to solve for the area of a sphere. Now look at that program again.

Printing a Program

The printer on your HP-95C will give you a separate listing of any program contained in the calculator at any time. For example, to print a listing of the program that is now defined by the **A** key, first set the calculator switches as shown here:

ON-OFF switch OFF  ON to ON.

Print Mode switch MAN  ^{TRACE} NORM to MAN.

PRGM-RUN switch PRGM  RUN to RUN.

Now press **GTO** **A** followed by **f** **PRINT** **A**.

Since you still have the program for the area of a sphere remaining in **A** from when you worked through Meet the HP-95C, the calculator should generate a listing like the one shown here:

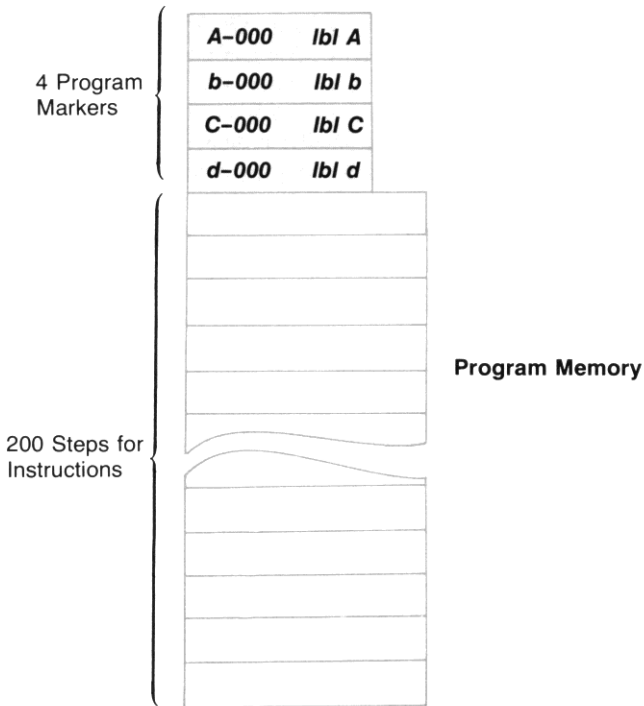
```
A-000 LBLA
A-001 X²
A-002 √
A-003 ×
A-004 PRTX
```

Note: If your program listing does not look like the one shown here, go back to Meet the HP-95C and work through the section entitled Programmed Problem Solving (pages 15–16). Then print the program listing again as illustrated above.

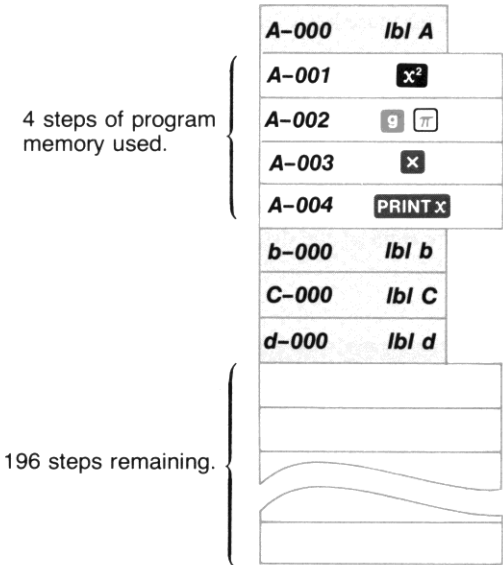
You can see that the calculator has printed a list of numbers and keys. This list is actually a complete list of the program that is now defined by **A** in the calculator. Each line of the print-out represents a single step of *program memory*.

Program Memory

In the HP-95C, keystrokes that make up programs are loaded and stored in a portion of the calculator called *program memory*. Program memory consists of 200 steps and four markers. Program memory is separate from the stack and storage registers.



When you loaded the program to calculate the area of a sphere into your HP-95C, the function keys you pressed were “remembered” in program memory after the A-000 program marker, so the contents of program memory now look like this:



Since you used four steps of program memory for the **A** program, there are still 196 unused steps of program memory for other programs. You can allocate the 200 available steps of program memory any way you choose. For example, you could have a 4-step program in partition **A**, a 150-step program in **B**, a 6-step program in **C**, and a 40-step program in **D**, if you so desired. Or you could have smaller programs utilizing all four markers and leave a portion of program memory unused. Or you could even use all 200 steps for one program if you wished!

Notice that nothing can be loaded into the four program markers. These markers act only as addresses within memory, as convenient “starting points” for programs. They cannot be deleted nor do they take up any actual space.


Refer to your printed copy again. The column on the left-hand side of the print-out indicates the *step number* of program memory. Notice that only the steps used in program **A** are printed. If you had a program stored in **B**, you would have to press **F** PRINT **B** to see *that* program printed.

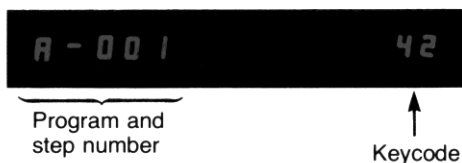
The right-hand column of the print-out shows you the symbol for the instruction that is contained in that program memory step. Each instruction (or function) is contained in a single step of program memory, no matter how many keystrokes it requires.

For example, if you examine the print-out of the program for calculating the area of a sphere, you will see that step A-001 contains the **x** instruction, a single keystroke. Step A-002 of program memory, however contains an instruction, **g** **π**, that required two keystrokes when you pressed it from the keyboard. Step A-003 contains the **x** instruction, and step A-004 contains the **PRINT x**. These are the keys that you pressed to load the program into the calculator.

Keycodes

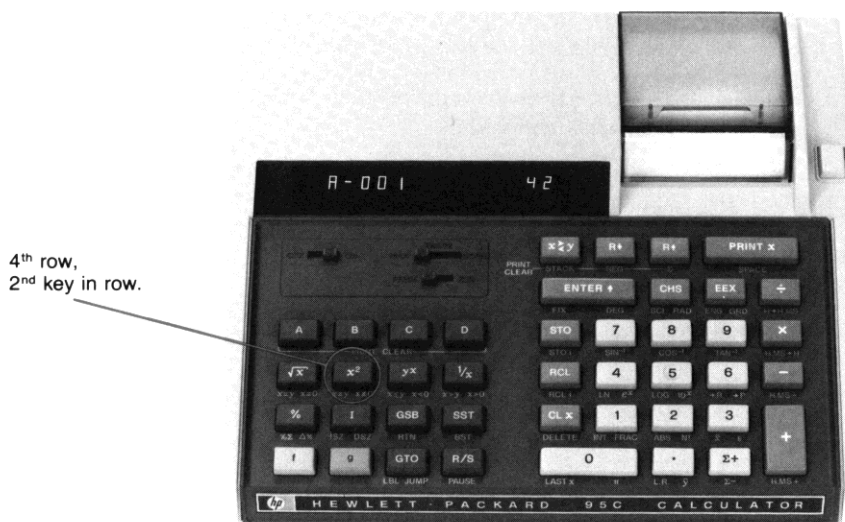
You have seen how you can examine a program that is in the calculator by printing it out on the paper tape. But you can also tell from the display what keystrokes are loaded into each step of a program by means of the *keycodes* for those keys. Let's look at some keycodes now.

1. First press **GTO A** to ensure that the calculator is set to the top of program A.
2. Slide the PRGM-RUN switch PRGM  RUN to PRGM. You should see the program marker for program A: **A-000** **Ibl A**.
3. Now press the **SST** (*single-step*) key. This moves the calculator down one step of program memory, and your display should look like the one below. The step number is on the left, and the keycode is on the right of the display.



Each key on the HP-95C has a keycode. For each keycode, the first digit denotes the *row* of the key on the keyboard and the second digit identifies the number of the key in that row. So keycode 42 identifies the 4th row on the keyboard, 2nd key in that row. Always count from the top down, and from left to right. Each key, no matter how large, counts as one.

The key in the 4th row, 2nd key in that row, is the x^2 key.



Now use the **SST** key again to examine the keycode in the next step of the program:

Press	Display
SST	A-002 62 0

The first keycode, 62, identifies the 6th row of the keyboard, 2nd key in that row, the **9** key. For convenience, digit keys are identified by single-digit keycodes of 0 through 9, so the second keycode identifies the **0** key. However, since the **0** key is preceded by the **9** key, the complete operation contained in step A-002 is **9** **π**.







Using this handy matrix system, you can easily identify any key by its keycode. (When seen in the display, as you know, references to **A**, **B**, **C**, and **D** appear as A, b, C, or d.) Each step of program memory can contain either a single digit or a complete operation, no matter how many keystrokes the operation requires.

Clearing a Program

The Continuous Memory of your HP-95C preserves any programs loaded into program memory even though the calculator is turned OFF.

To clear a complete program from the calculator, use the **9** prefix key in PRGM mode followed by the user-definable key (**A**, **B**, **C**, or **D**) of the program you want cleared.


Thus, to clear a selected program from the calculator:

1. Set the PRGM-RUN switch  to PRGM.
2. Press  followed by the key (CLEAR , CLEAR , CLEAR , CLEAR ) that selects the program you wish to clear.

Note: When clearing a long program, which can take several seconds, the display blinks all zeros to indicate that a program is being cleared.

Clearing a program also sets the calculator to the program marker (step 000) of that program, ready to key in a new program if you wish.







When you clear a program, only that program is affected. This permits you to maintain the stack, storage registers, and other programs in the calculator while clearing one program. All the steps of program memory formerly used by the cleared program are again available for storing program instructions for *any* program.

If power has been interrupted to the calculator (evidenced by a display of , all instructions in program memory are lost (that is, *all* programs are cleared) and the calculator resets to program marker A.

Now let's load another program into the HP-95C to see how some of its features work.

Creating a Program

If you wanted to use the HP-95C to manually calculate the area of a circle using the formula

$A = \pi \left(\frac{d}{2} \right)^2$, you could first key in the diameter d , then divide it by 2 by pressing  , and square that quantity by pressing . Next you would summon the quantity pi into the display by pressing  . Finally you would multiply by pressing .

Remember that a *program* to solve a problem is nothing more than the keystrokes you would press to solve the problem manually. Thus, in order to create a program for the HP-95C that will solve for the area of *any* circle, you use the same keys you pressed to solve the problem manually.

The keys that you used to solve for area of a circle according to the formula $A = \pi \left(\frac{d}{2} \right)^2$ are:




You will load these keystrokes into program to create a program.

The Beginning of a Program

To define the beginning of a program, you should use one of the program markers, A, B, C, or D. You can then have as many as four different programs loaded into the calculator at any time, and by selecting them with the user-definable keys **A**, **B**, **C**, and **D**, you can run them in the order you choose.


Later you will see how you can use the **LBL** function followed by a digit key (**0** through **9**) to define a routine—that is, a part of a program—and how routines may be selected from the keyboard, just like larger programs. However, for now we will concentrate on a simple program whose beginning is defined by one of the program markers.

Loading a Program


To load a program from the keyboard, simply slide the PRGM-RUN switch **PRGM**  **RUN** to **PRGM** (*program*). When the PRGM-RUN switch is in the PRGM position, the functions and operations that are normally executed when you press the keys are not executed. Instead, they are *stored* in program memory for later execution. *All operations on the keyboard except six can be loaded into program memory for later execution.* The six operations that cannot be loaded in as part of a program are:

f **PRINT** (**A** – **D**), **g** **CLEAR** (**A** – **D**), **f** **DELETE**, **SST**, **f** **BST**, **GTO**  (**A** – **D**, **0** – **9**).


These six operations are used to help you load, edit, and modify your programs in the calculator.

All other functions when pressed with the PRGM-RUN switch **PRGM**  **RUN** in **PRGM** mode are loaded into the calculator as program instructions to be executed later.

To load a complete program into the calculator:

1. Slide the PRGM-RUN switch **PRGM**  **RUN** to **PRGM**.
2. Press **g** followed by the appropriate user-definable key to ensure that no previous program is also defined by that program marker. This also sets the calculator to the program marker of the cleared program, ready for you to key in instructions for the new program.
3. Load the new program by pressing the function keys in the same order you would press them to solve a problem from the keyboard.

So to load the program for the area of a circle:

Slide the PRGM-RUN switch **PRGM**  **RUN** to **PRGM**.

Press	Display
g CLEAR C	C-000 lbl C

You can tell that the calculator is now set at program marker C because of the display **C-000** **lbl C** that is seen in the display.

Now that the calculator is set to program marker C and any previous instructions have been cleared from C, you can begin keying in the instructions for the program. Press the first function key of the program now:

Press	Display	
	C-001	2

As you can see, the first instruction has been loaded into the first step of program C, step C-001. The keycode, 2, indicates that the instruction is the digit key . Now load the second instruction of the program:

Press	Display	
	C-002	24

The second instruction of the program, , has been loaded into step C-002. The keycode 24 indicates that the instruction is found on the 2nd row of the keyboard, 4th key in that row. Now load the remainder of the program by pressing the keys. Observe the program memory step numbers and keycodes.

Press	Display	
	C-003	42
	C-004	62 0
	C-005	39

The program for solving for the area of a circle given its diameter is now loaded into the portion of program memory defined by .

Running a Program

To run a program, you have only to slide the PRGM-RUN switch to RUN, key in any “unknown” data that are required, and press the letter key (, , , or) that defines your program.

For example, to use the program now defined by to solve for circles with diameters of 3 inches, 6 meters, and 9 miles.

First, slide the PRGM-RUN switch PRGM RUN to RUN.

Press	Display	
3	7.07	Square inches.
6	28.27	Square meters.
9	63.62	Square miles.

Now let’s see how the HP-95C executed this program.

Going to a Program Marker

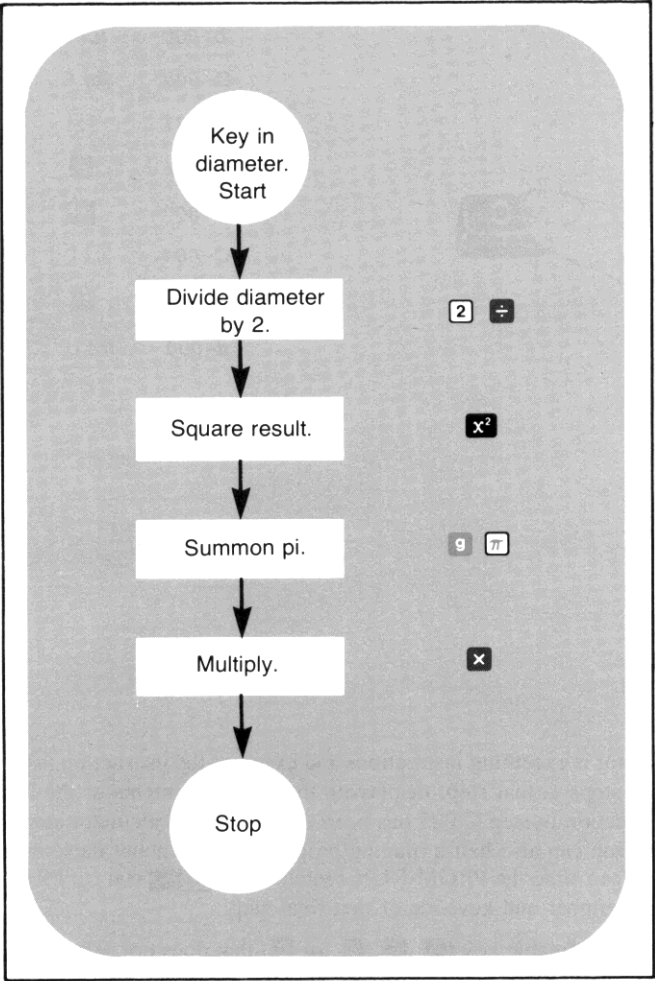
When you switched the PRGM-RUN switch PRGM RUN to RUN, the calculator was set at step C-005 of program memory, the last step you had filled with an instruction when you were loading the program. When you then pressed the user-definable key, the calculator went directly to the program marker, step C-000, and began executing instructions.

Flowcharts

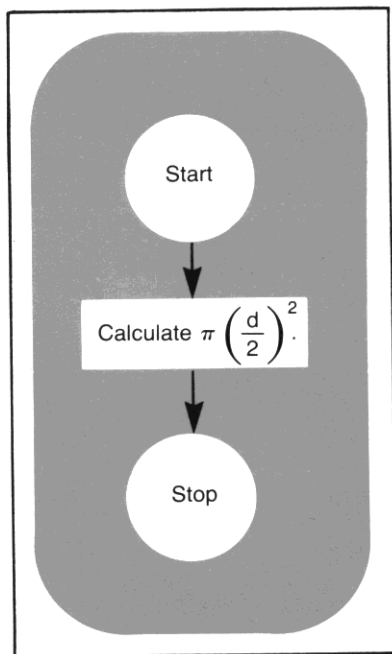
At this point, we digress for a moment from our discussion of the calculator itself to familiarize ourselves with a fundamental and extremely useful tool in programming—the flowchart.

A flowchart is an *outline* of the way a program solves a problem. With 200 possible instructions, it is quite easy to get “lost” while creating a long program, especially if you try to simply load the complete program from beginning to end with no breaks. A flowchart is a shorthand that can help you design your program by breaking it down into smaller groups of instructions. It is also very useful as documentation—a road map that summarizes the operation of a program.

A flowchart can be as simple as you like. Here is a flowchart that shows the operations you executed to calculate the area of a circle according to the formula $A = \pi \left(\frac{d}{2}\right)^2$. Compare the flowchart to the actual instructions for the program:



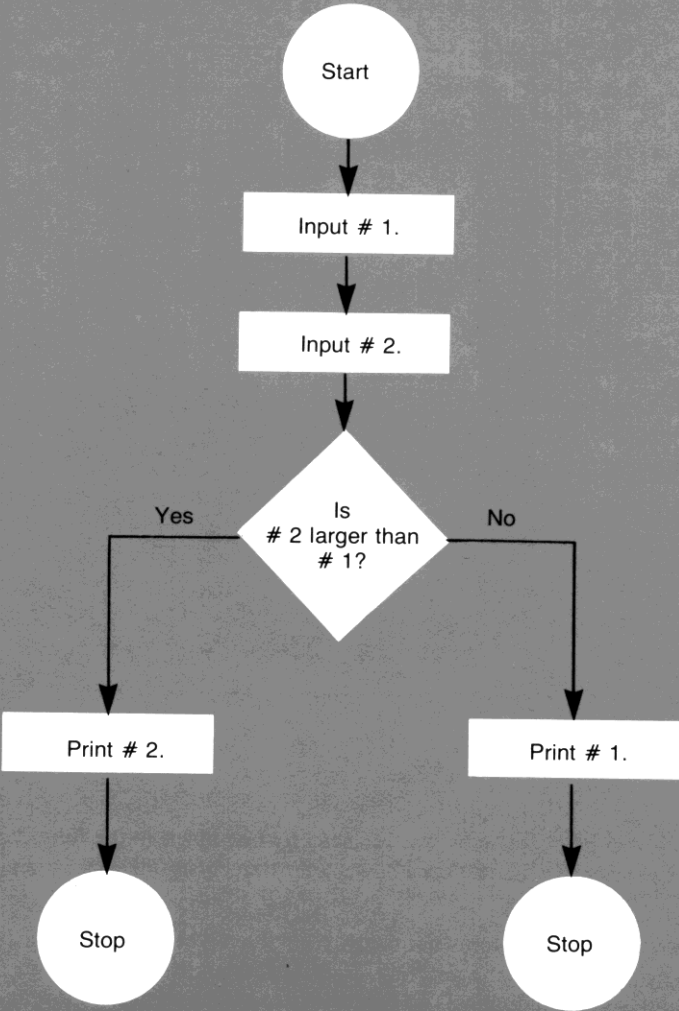
You can see the similarities. At times, a flowchart may duplicate the set of instructions exactly, as shown above. At other times, it may be more useful to have an entire group of instructions represented by a single block in the flowchart. For example, here is another flowchart for the program to calculate the area of a circle:



Here an entire group of instructions was replaced by one block in the flowchart. This is a common practice, and one which makes a flowchart extremely useful in visualizing a complete program.

You can see how a flowchart is drawn linearly, from the top of the page to the bottom. This represents the general flow of the program, from beginning to end. Although flowcharting symbols sometimes vary, throughout this handbook we have held to the convention of circles for the beginning and end of a program or routine, and rectangles to represent groups of functions that take an input, process it, and yield a single output. We have used a diamond to represent a *decision*, where a single input can yield either of two outputs.

For example, if you had two numbers and wished to write a program that would print only the larger, you might design your program by first drawing a flowchart that looks like the one shown on page 116.



After drawing the flowchart, you would go back and substitute groups of instructions for each element of the flowchart. When the program was loaded into the calculator and run, if #2 was larger than #1, the answer to the question “Is #2 larger than #1?” would be YES, and the program would take the left-hand path, print #2, and stop. If the answer to the question was NO, the program would execute the right-hand path, and #1 would be printed. (You will see later the many decision-making instructions available on your HP-95C.)

As you work through this handbook, you will become more familiar with flowcharts. Use the flowcharts that illustrate the examples and problems to help you understand the many features of the calculator, and draw your own flowcharts to help you create, edit, eliminate errors in, and document your programs.

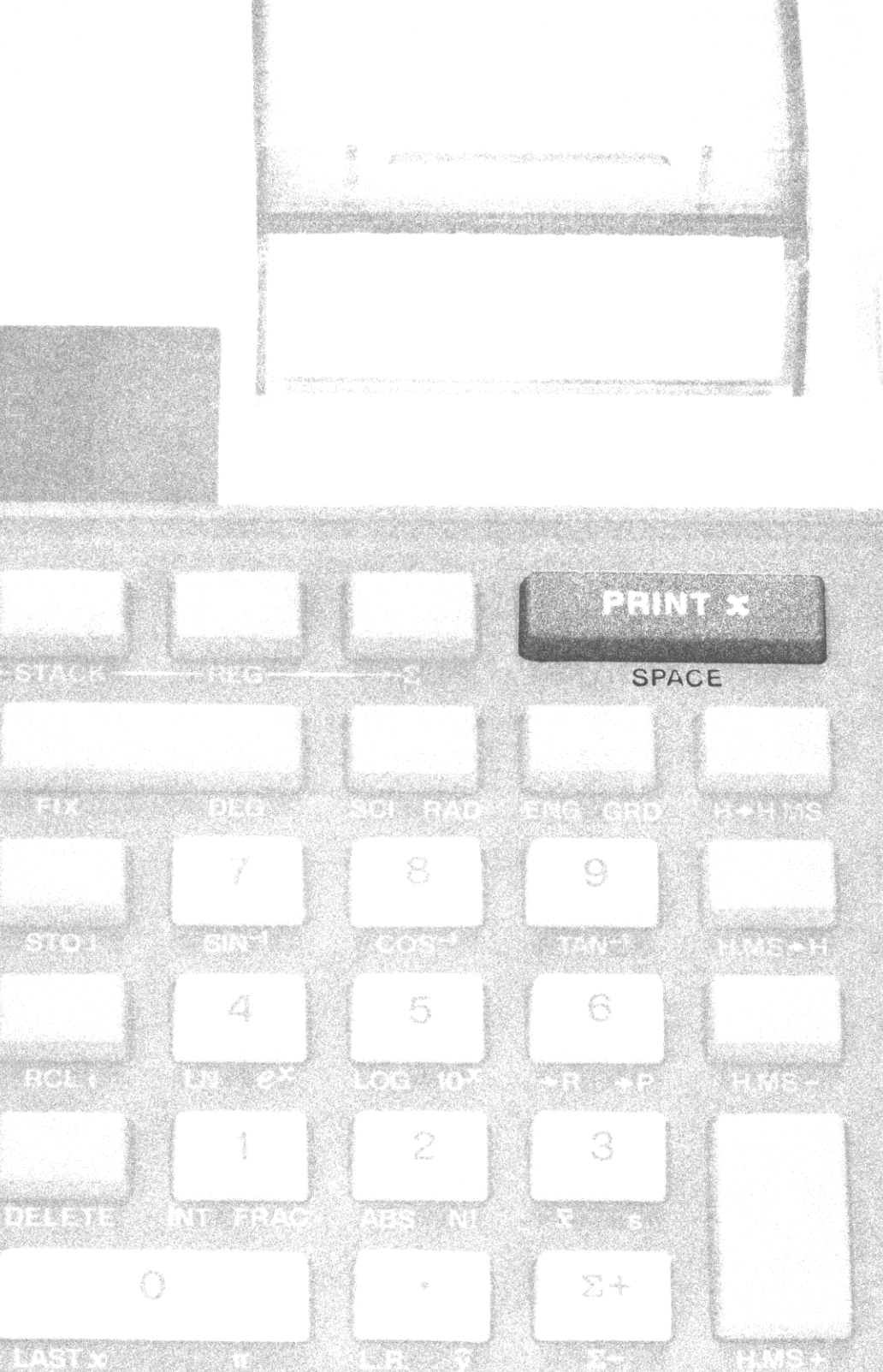
Problems

1. What would be the keycodes for the following operations: $\frac{1}{x}$, 9 GRD , $+$, f , HMS , $\text{STO} + 1$?
2. What operations are identified by the following keycodes: 21, 61 49, 62 12, 45 8, 35 59 0?
3. You have seen how to write, load, and run a program to calculate the area of a circle from its diameter. Now write and load a program that will calculate the radius r of a circle given its area A using the formula $r = \sqrt{A/\pi}$.

Load the program as follows:

- a. Set the PRGM-RUN switch to PRGM.
- b. Clear program A using the 9 CLEAR A operation.
- c. Key in the instructions for the program.
- d. Switch back to RUN mode and run the program to calculate the radii of circles with areas of 28.27 square inches, 113.10 square meters, and 254.47 square miles.

(Answers: $\boxed{3.00}$ inches, $\boxed{6.00}$ meters, $\boxed{9.00}$ miles.)




			PRINT x	
STACK	REG	S	SPACE	
FIX	DEG	SCI RAD	ENG GRD	H \leftrightarrow HMS
	7	8	9	
STO \downarrow	SIN $^{-1}$	COS $^{-1}$	TAN $^{-1}$	HMS \leftrightarrow H
	4	5	6	
RCL \uparrow	LN e^x	LOG 10^x	$\rightarrow R$ $\rightarrow P$	HMS \leftarrow
	1	2	3	
DELETE	INT FRAC	ABS NI	Σ^- Σ^+	
	0	.	Σ^+	
LAST x	π	LR \rightarrow	Σ^-	HMS \rightarrow

The Printer and the Program

All print functions on the HP-95C (except **PRINT A**, **PRINT B**, **PRINT C**, and **PRINT D**) can be recorded as instructions in program memory and later executed as part of a program. In addition, you can use the three modes of the HP-95C printer to aid you in programming.


Printer Operation During a Running Program

Like the other keys on your HP-95C, the printer operates in the same natural, normal manner during a running program as it does when you are using the calculator manually. Thus, if you have the Print Mode switch set to **TRACE** during a running program, the printer will preserve a record of each operation and intermediate result calculated during the program, just as it would if you were pressing each key yourself.

In most cases, you will probably want the HP-95C to run a program as quickly as possible, and you will not want the running program to have to “slow down” to engage the printer, as it does in the **TRACE** mode of the Print Mode switch. To ensure the fastest program execution, place the Print Mode switch **MAN**  **NORM** in the **MAN** or **NORM** position. If you want the HP-95C to print some results during or at the end of a running program, simply place **PRINT x** instructions in the program wherever printed results are required.

Using the Printer for Creating Programs

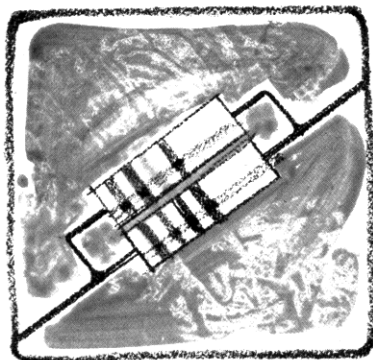
The printer on your HP-95C is a valuable and time-saving tool that you can use not only to record answers and program listings, but also to aid you in creating and editing programs.

For example, when creating a program you can use the printer to generate the list of keystrokes that will later form the bulk of your program. With the Print Mode switch **MAN**  **NORM** set to **NORM**, the printer records a history of the calculations necessary to solve a problem. Then you can simply go back and follow the listing produced by the printer when loading instructions into program memory.

Example: The formula to calculate the total resistance of two parallel resistances in an electrical circuit is:


$$R_t = \frac{R_1 \times R_2}{R_1 + R_2}$$







Write a program that will permit you to key in any two parallel resistances and calculate the total resistance.



Solution: Begin by selecting a pair of sample resistances; say, 1500 ohms and 2200 ohms. Then solve for the total resistance using the formula above with the Print Mode switch set to NORM. To solve for parallel resistances of 1500 ohms and 2200 ohms:

Set the PRGM-RUN switch PRGM  RUN to RUN.

Slide the Print Mode switch  NORM to NORM.

Press	Display
1500	1500.
 1	1500.00
2200	2200.
	3300000.00
 LAST X	2200.00
 1	1500.00
	3700.00
	891.89

As you can see, you have generated a list of keystrokes that solves the problem. Your list should look like this:

```

1500.00 S 1
2200.00 x
LSTX
R 1
+
=

```

Now all you have to do is to add to this list slightly so that it will work for *all* values of R1 and R2. With the Print Mode switch still set to NORM, solve the problem again so that the value for R2 will be stored in register R₁ and the value for R1 will be placed in the LAST X register when the multiplication operation is performed:

```

1500.00 ENT↑
2200.00 S 1
X↔Y
x
LSTX
R 1
+
=

```

Now assume that the values for R1 and R2 have been input to the Y- and X-registers, respectively. If the program were defined by **D**, it would look like this:


```

D-000 LBLD
D-001 S 1
D-002 X $\leftrightarrow$ Y
D-003  $\times$ 
D-004 LSTX
D-005 R 1
D-006 +
D-007  $\div$ 

```

Now load the program. The printer is a great aid here, too, as you will see.

Program Load Verification

In PRGM mode, with the Print Mode switch **MAN**  **NORM** set to TRACE, the printer records the keystrokes you press as you load a program. This gives a printed record of the program *as you key it in*, verifying that you are loading it correctly.

Switch to PRGM and TRACE modes now, and monitor the loading of the program as you press the keystrokes:

First, slide the Print Mode switch **MAN**  **NORM** to TRACE. Slide the PRGM-RUN switch **PRGM**  **RUN** to PRGM.

Press

Display

9 CLEAR **D**

d-000 lbl d

Clears previous program (if any) from **D** and sets calculator to program marker D.

Now follow the list of keystrokes to key in the rest of the program.

Press

Display

STO 1

d-001 35 1

D-001 S 1

X \leftrightarrow Y

d-002 11

D-002 X \leftrightarrow Y

\times

d-003 39

D-003 \times

f LAST X

d-004 61 0

D-004 LSTX

RCL 1

d-005 45 1

D-005 R 1

+

d-006 59

D-006 +


\div

d-007 24

D-007 \div

Now switch back to RUN mode and run the program to see that it yields the same answer for your test case:











Slide the PRGM-RUN switch  to RUN.

Slide the Print Mode switch  back to MAN (unless you want the printer to trace the operation of the program, too).


Press	Display
1500 	1500.00
2200	2200.
	891.89




As you can see, the printer is a great aid in the creation of programs.

Program Listing

Whether the PRGM-RUN switch is set to PRGM or RUN, you can list your program. When you press  PRINT , , , or , the printer lists the step number and operation for each step of the program, beginning with the current step and continuing until the last step is encountered. If you are in one program and press  PRINT followed by the user-definable key (, , , ) of another *program*, the printer prints all of the program selected.





You can stop the printing of a program at any time by simply pressing any key from the keyboard.

To list the program you currently have in :





Press	Display	
 	891.89	Sets calculator to top of program  .

 PRINT 	891.89
---	--------


D-000 LBLD
D-001 S 1
D-002 X $\frac{Z}{Y}$
D-003 x
D-004 LSTX
D-005 R 1
D-006 +
D-007 \div


Since the calculator stopped after the final step of program  when it completed running the program, you made use of the   operation to set it to the top of program  before printing.

Printing a Space


If you wish to insert a space between portions of your print-out or between answers, you can use the   (*space*) function. Like the paper advance pushbutton, this function advances the paper one space without printing. Digit entry is terminated by the paper advance pushbutton or by the   function.

For example:


Press	Display	
123	123.	
 SPACE	123.00	Paper advances one space.
456	456.	Digit entry was terminated by printing a space.



From the keyboard, of course, the paper advance pushbutton is the easiest way to advance the paper without printing. From a program, however, the use of  SPACE instructions allows you to place as many spaces as you desire in your printed results.

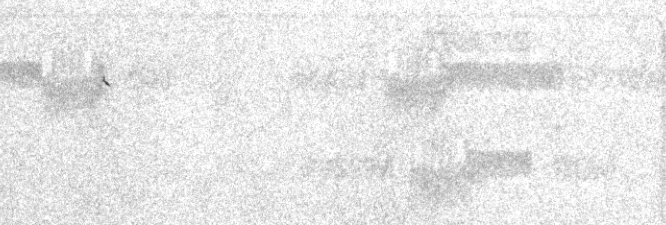
Problems

1. Use the printer to help you write and load a program that will convert temperature in degrees Celsius to Fahrenheit according to the formula $F = 1.8C + 32$. Define the program with the user-definable  key, and run it to convert Celsius temperatures of -40° , 0° , and $+72^{\circ}$ to Fahrenheit.

(Answers: -40.00 °F, 32.00 °F, 161.60 °F.)

2. Immediately after running the program in Problem #1, create and load a program that will convert temperature in degrees Fahrenheit back to Celsius according to the formula $C = (F - 32) 5/9$, selecting it with the user-definable  key. Run this new program to convert the temperatures in °F you obtained in Problem #1 back to °C.

If you wrote and loaded the programs as called for in Problems #1 and #2, you should be able to convert any temperature in Celsius to Fahrenheit by pressing , and any temperature in Fahrenheit to Celsius by pressing . You can see how you can have many different programs loaded into the HP-95C and select any one of them for running at any time.



PRINT
CLEAR

STACK

FIX

STO 1

RCL 1

DELETE

LAST x

B C D

PRINT CLEAR

x^2 y^x $1/x$

$x \neq y$ $x \geq 0$ $x \leq y$ $x < 0$ $x > y$ $y > 0$

1 GSB SST

ISZ DSZ

RTN

BST

9 GTO R/S

LBL JUMP

PAUSE

Program Editing

Often you may want to alter or add to a program that is loaded in the calculator. On your HP-95C keyboard, you will find several editing functions that permit you to easily change any step of a loaded program *without* reloading the entire program.



As you may recall, all functions and operations on the HP-95C keyboard can be recorded as instructions in program memory except six. Five nonrecordable operations, together with the **GTO** function used from the keyboard, are *program editing and manipulation functions*, and they can aid you in altering and correcting your programs.



Program Editing and Manipulation Functions


Nonrecordable Operations

9 **CLEAR** (**A**, **B**, **C**, or **D**) is one keyboard operation that cannot be recorded in program memory. As you know, when you press **9** in PRGM mode followed by one of the program markers (**A**, **B**, **C**, or **D**), all instructions are cleared from that program, and the portion of program memory formerly used by those instructions is again available for use in that or other programs. (In RUN mode, pressing **9** followed by one of the user-definable keys, **A**, **B**, **C**, or **D**, does nothing except cancel the **9** key.)

f **PRINT** (**A**, **B**, **C**, or **D**) is another operation that is not recordable in program memory. When you press **f** followed by a user-definable key, the HP-95C immediately begins printing the contents of that program beginning with the step of program memory to which the calculator is set. (If the calculator is set to another program, printing will begin with step 000 of the *selected* program.) This feature allows you to list a complete program at any time, regardless of the setting of the PRGM-RUN switch. The calculator prints the contents of the selected program until it reaches the end of the program or until you manually halt printing by pressing any key from the keyboard.

SST (*single step*) is another nonrecordable operation. When you press **SST** with the PRGM-RUN switch PRGM  RUN set to PRGM, the calculator moves to and displays the next step of program memory. When you press **SST** down with the PRGM-RUN switch PRGM  RUN set to RUN, the calculator displays the next step of program memory—when you release the **SST** key, the calculator executes the instruction loaded in that step. **SST** permits you to single step through a program, executing the program one step at a time or merely viewing each step without execution, as you choose.

f **BS** (*back step*) is a nonrecordable operation that displays the *previous* step of program memory. When you press **f** **BS** with the PRGM-RUN switch PRGM  RUN set to PRGM, the calculator moves to and displays the previous step of program memory. When you press and release **f** and then press down **BS** with the PRGM-RUN switch PRGM  RUN set to RUN, the calculator moves to and displays the contents of the previous step of program memory. When you then release **BS**, the original contents of the X-register are displayed. No instructions are executed.

The **DELETE** (*delete*) key is a nonrecordable operation that you can use to delete instructions from program memory. When the PRGM-RUN switch **PRGM**  **RUN** is set to PRGM and you press **f** **DELETE**, the instruction at the current step of program memory is erased, and all subsequent instructions in program memory move upward one step. The section of program memory shown below illustrates what would happen when you press **f** **DELETE** with the calculator set to step C-003.

With the calculator set to step C-003, when you press **f** **DELETE**, program memory is changed...

... from this ...


```
C-000 LBL C
C-001 X²
C-002 f
C-003 x
C-004 PRTX
```


... to this.

```
C-000 LBL C
C-001 X²
C-002 f
C-003 PRTX
```

In RUN mode, pressing **DELETE** merely clears (cancels) an **f** prefix key that you have pressed.

Using **GTO** From the Keyboard

GTO (*go to*), when used from the keyboard, is operated in two slightly different ways, depending upon the position of the PRGM-RUN switch. When pressed from the keyboard with the PRGM-RUN switch **PRGM**  **RUN** set to RUN, **GTO** followed by a user-definable key (**A**, **B**, **C**, or **D**) sets the calculator to step 000 of the selected program. No program instructions are executed.

When the PRGM-RUN switch **PRGM**  **RUN** is set to PRGM, press **GTO** followed by the decimal point key **.** followed in turn by a user-definable key to set the calculator to step 000 of the desired program. This is a nonrecordable operation, and no instructions are loaded into program memory by it.

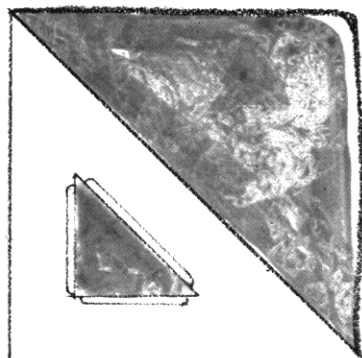
You *can* use **GTO** as a recorded instruction in a program—more about the use of **GTO** in a program in section 9, Branching.

Now let's load a program into the calculator and use these editing tools to check and modify it.

Pythagorean Theorem Program

The following program computes the hypotenuse of any right triangle, given the other two sides. The formula used is $c = \sqrt{a^2 + b^2}$.

Below are instructions for the program (basically, the same keys you would press to solve for c manually), assuming that values for sides a and b have been input to the X- and Y-registers of the stack.



To load the program:

First slide the PRGM-RUN switch PRGM  RUN to RUN. Then press **GTO A** to set the calculator to program marker A.

Press	Display	
GTO A	0.00	(Display assumes no results remain from previous examples.)

If you slide the PRGM-RUN switch to PRGM now, you can see that the calculator is set to program marker A. Any instructions in program A are still there at this point, but you can clear them with the **CLEAR A** operation:

Press	Display	
	A-000 Ibl A	Program marker A seen when you switch to PRGM mode.
g CLEAR A	A-000 Ibl A	Program A cleared.

Clearing program **A** would itself have set the calculator to program marker A, but the above example illustrates how **GTO** can be used to set the calculator to a program marker in RUN mode.

Now, load the Pythagorean Theorem program by pressing the keys shown below:


Press	Display	
x²	A-001	42
x↔y	A-002	11
x²	A-003	42
+	A-004	59
√x	A-005	41

With the program loaded into the HP-95C, you can run the program. For example, you can calculate the hypotenuse of a right triangle with side a of 22 meters and side b of 9 meters.

However, before you can run the program, you must *initialize* it.

Initializing a Program

Initialization of a program means nothing more than setting up the program (providing inputs, setting display mode, etc.) prior to the actual running of it. Some programs contain initialization routines that set up the data to run the program. In other programs, you may have to initialize manually from the keyboard before running. In the case of the program for calculating the hypotenuse of a triangle, to initialize the program you must place the values for sides a and b in stack registers X and Y. (Notice that the *order* does not matter in this case.) Thus, to initialize this program:


First, slide the PRGM-RUN switch PRGM  RUN to RUN.

Press	Display
22 	22.00
9	9.



The program for hypotenuse of a right triangle using the Pythagorean Theorem is now initialized for sides of 22 and 9 meters.

Running the Program

To run the program you have only to press the user-definable key that selects this program.




Press	Display	
	23.77	Length of side c in meters.

To compute the hypotenuse of a right triangle with a side a of 73 miles and a side b of 99 miles:

Press	Display	
73 	73.00	
99	99.	Program initialized for new set of data before running.
	123.00	Length of side c in miles.

Now let's see how we can use the nonrecordable editing features of the HP-95C to examine and alter this program.

Single-Step Execution of a Program

You can use the  (*single-step*) key to run a program one step at a time. As you know, the calculator executed program  sequentially, executing the instructions in the same order you keyed them in. When the calculator had executed the final step of program , it stopped, displaying the calculated value for the hypotenuse of the selected triangle.

First, set the calculator to the top of program A with the **GTO** operation:

Press	Display	
GTO A	123.00	Contents of X-register displayed. Calculator is set to top of program A.

With the Program Mode switch set to RUN, you can execute a recorded program one step at a time by pressing the **SST** (*single-step*) key.

To single-step through the Pythagorean Theorem program using a triangle with side *a* of 73 miles and side *b* of 99 miles:

Press	Display	
73 ENTER	73.00	
99	99.	Program initialized for this set of data before running.

Now press **SST** and hold it down to see the keycode for the next instruction. When you release the **SST** key, the next instruction is executed.

Press	Display	
SST	A-001 42	Keycode for x² seen when you hold SST down.
	9801.00	x² executed when you release SST .

The first instruction of the program was executed when you pressed and released **SST**. (Notice that you didn't have to press **A**—when you are executing a program one step at a time, pressing the **SST** key begins program execution with the instruction in the next step of program memory from wherever the calculator is set.)

Continue executing the program by pressing **SST** again. When you hold **SST** down, you see the keycode for the next instruction. When you release **SST**, that instruction is executed.

Press	Display	
SST	A-002 11	Keycode for x₁y .
	73.00	Executed.

When you press **SST** a third time in RUN mode, step A-003 of program memory is displayed. When you release the **SST** key, the instruction in that step, **x²**, is executed and the calculator halts.


Press	Display	
SST	A-003	Keycode for x² .
	5329.00	Executed.

Continue executing the program by means of the **SST** key. When you have executed the last instruction of the program, the **x²** instruction in step A-005, you have completed execution of the program, and the calculator displays the answer. This is just as if the calculator had executed the program automatically, instead of via the **SST** key.

Press	Display	
SST	A-004	59
	15130.00	
SST	A-005	41
	123.00	

You have seen how the **SST** key can be used in RUN mode to single-step through a program. Using the **SST** key in this manner can help you create and correct programs. Now let's see how you can use **SST** and **BST** in PRGM mode to help you modify a program.


Modifying a Program

Since you have completed execution of the above program, the HP-95C is set at the final step, step A-005. You can verify that the calculator is set at this step by briefly sliding the Program Mode switch **PRGM**  **RUN** to **PRGM** and observing the step number and keycode in the display.

Now let's modify this Pythagorean Theorem program so that it will *print* the values of sides *a* and *b* and the calculated value of the hypotenuse, side *c*. First, print the program so that you can see where instructions will have to be inserted to modify the program as required.

Press	Display	
GTO A	123.00	Sets calculator to top of program A.
f PRINT A	123.00	You want to insert PRINTX instructions after each of these instructions.


Printer lists program instructions until it has printed the final step of the program, then returns to step A-000 and stops.



- A-000 LBLA
- A-001 X²
- A-002 X↔Y
- A-003 X²
- A-004 +
- A-005 √X

Single-Step Viewing Without Execution

In section 6, Simple Programming, you used the **SST** key in PRGM mode to examine the keycodes in a program. This use of the **SST** key *without* executing the program can be used to step to any part of a program for editing.

When you slide the PRGM-RUN switch **PRGM**  **RUN** to PRGM, you should see that the calculator is reset to step A-000 of program memory. Since you will want to add a **PRINT x** instruction as the last step of the program, use the **SST** key now to single-step to the existing last step without executing instructions:

Slide the PRGM-RUN switch **PRGM**  **RUN** to PRGM.

Press	Display	
SST	A-001	42
SST	A-002	11
SST	A-003	42
SST	A-004	59
SST	A-005	41

Final step of the program.

You can see that the calculator is now set at step A-005 of program memory. If you press a recordable operation now, it will be loaded in the *next* step, step A-006. Thus, to load the **PRINT x** instruction so that the calculator will print the value of the hypotenuse when it has been calculated:

Press	Display	
PRINT x	A-006	14

Now we will use **BST** to step backward through the program and insert another **PRINT x** instruction.

Stepping Backward Through a Program

The **BST** (*back step*) function allows you to back step through a loaded program for editing whether the calculator is in RUN or PRGM mode. When you press **BST**, the calculator backs up one step in program memory. If the calculator is in RUN mode, the previous step is displayed as long as you hold down the **BST** key. When you release it, the original contents of the X-register are again displayed. In PRGM mode, of course, you can see the step number and keycode of the current instruction in the display at all times. No instructions are executed, whether the calculator is in RUN or PRGM mode.

You now have another **PRINT x** instruction to add to the Pythagorean Theorem program. The **PRINT x** instruction should be added after the **X₂Y** instruction, keycode 11, that is now loaded in step A-002 of program memory. If you have just completed loading a **PRINT x** instruction in step A-006 as described above, the calculator should still be set at step A-006. You can use the **BST** function to back the calculator up to step A-002, then insert the next **PRINT x** instruction in step A-003. To begin:

Ensure that the PRGM-RUN switch **PRGM**  **RUN** is set to PRGM.

Press	Display		
	A-006	14	Calculator initially set to step A-006.
f BST	A-005	41	Pressing BST once moves the calculator back one step in program memory.

When you press **f** **BST**, the calculator backs up one step in program memory. No instructions are executed when you use **BST**. Continue using **f** **BST** to move backward through program memory until the calculator displays step A-002.

Press	Display	
f BST	A-004	59
f BST	A-003	42
f BST	A-002	11

Since you wish to insert the **PRINTX** instruction *after* the **X₂Y** instruction now loaded in step A-002, you move the calculator to step A-002 first.

Program printing, searching, or execution always begin with the *current* step of program memory (that is, the step being displayed). Program loading, however, begins with the next step following the one being displayed. So when you key in an instruction with the calculator set to step A-002, that instruction is loaded in the next step, step A-003.

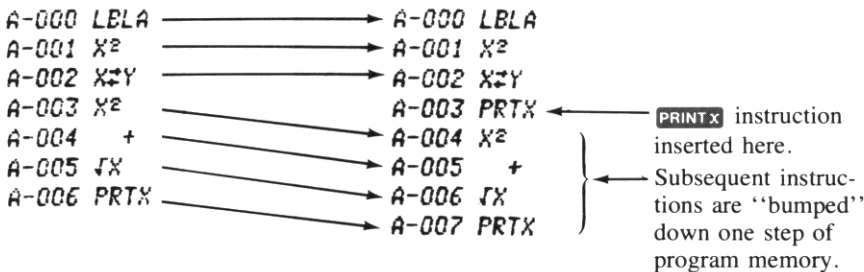
Press	Display	
PRINTX	A-003	14

You can see that a **PRINTX** instruction, keycode 14, has been loaded into step A-003. The instruction that was *formerly* in step A-003 is moved to step A-004, and the instructions in subsequent steps are similarly moved down one step. No instructions are lost—the **A** program has simply become one step longer.

When you added the **PRINTX** instruction after step A-002, program **A** was altered...

... from this ...

...to this.



You can continue adding instructions to any program until a total of 200 steps of program memory have been filled by instructions from all four programs. If you attempt to add another instruction when program memory is full, the calculator will immediately display **SEE 2**. No instructions will be lost from memory.

You now have one additional **PRINTX** instruction to insert, as the first step of the program. Use the **GTO** operation from the keyboard to set the calculator to the top of program A:

Press	Display
GTO ▢ A	A-000 <i>lbl A</i>

Since the calculator was in PRGM mode, you used the operation **GTO** **▢** followed by the desired user-definable key to set the calculator to the top of program A.

Notice that if you try to use **BST** operations to move the calculator past the beginning of the program, you cannot do so:

Press	Display
f BST	A-000 <i>lbl A</i>
f BST	A-000 <i>lbl A</i>

Now add the final **PRINTX** instruction as the first instruction of the program:

Press	Display
PRINTX	A-001 14

The **PRINTX** instruction, keycode 14, has been loaded into step A-001, and all subsequent instructions in program **A** have been moved down one step of program memory. Program **A** is now one step longer.

Verifying Program Changes

You have now finished modifying the Pythagorean Theorem program to print the values of all three sides when it is run. The altered program is shown below:

```

A-000 LBLA
A-001 PRTX
A-002 X²
A-003 X↔Y
A-004 PRTX
A-005 X²
A-006 +
A-007 √X
A-008 PRTX
    
```

You can verify that the program in your HP-95C is the same as the one shown by printing the program. To print the modified Pythagorean Theorem program now loaded in your calculator:

Press	Display	
GTO ▢ A	A-000	b A
f PRINT A	A-000	b A Prints contents of program A . Calcu- lator returns to step A-000 after printing final step of program A.

Verify that the program printed by your HP-95C duplicates the one shown above.

Running the Modified Program


To run the Pythagorean Theorem program, you have only to key in values for sides *a* and *b* and press **A**. The HP-95C will calculate the value of side *c*, and it should now print values for sides *a*, *b*, and *c*. For example, to compute the hypotenuse of a right triangle with sides *a* and *b* of 22 meters and 9 meters:

Slide the PRGM-RUN switch **PRGM**  **RUN** to **RUN**.


Press	Display	
22 ENTER	22.00	9.00 ***
9	9.	22.00 ***
A	23.77	23.77 ***
		Program initialized.
		The answer in meters.

Now run the program for a right triangle with sides *a* and *b* of 73 miles and 99 miles.
(Answer: 123 miles.)

Deleting an Instruction

Often in the modification of a program you may wish to delete an instruction from program memory. To delete the instruction to which the calculator is set, merely press the nonrecordable operation **f** **DELETE** (*delete*) with the HP-95C PRGM-RUN switch **PRGM**  **RUN** set to **PRGM**. (When the PRGM-RUN switch is set to **RUN**, pressing **DELETE** does nothing except cancel a pressed prefix key **f**.) When you delete an instruction from a program using the **DELETE** key, all subsequent instructions in that program are moved *up* one step, and the program is one step shorter. The calculator moves to the step *before* the deleted step and displays it.

For example, if you wanted to modify the Pythagorean Theorem program that is now loaded into the calculator so that *only* the value for side *c* was printed, you would have to delete the **PRINT X** instructions, keycode 14, that are presently loaded in steps A-001 and A-003 of program **A**. To delete these instructions, you must first set the calculator at these steps using **SST** or **BST**, then press **f** **DELETE**. To delete the **PRINT X** instruction now loaded in step A-004:

First, slide the PRGM-RUN switch PRGM  RUN to PRGM.

Press	Display		
	A-000	LBL A	Calculator set at top of program A .
SST	A-001	14	
SST	A-002	42	
SST	A-003	11	
SST	A-004	14	Calculator set to step A-004.
f DELETE	A-003	11	The instruction in step A-004 is deleted and the calculator moves to step A-003.

You can use the **SST** key to verify that the **PRINT X** instruction, keycode 14, has been deleted and subsequent instructions have been moved up one step.

Press	Display		
SST	A-004	42	The instruction formerly in step A-005 was moved up to step A-004, and all subsequent instructions were moved up one step, when you pressed f DELETE .

When you set the calculator to step A-004 of program memory and pressed **f** **DELETE**, program **A** was altered...

... from this ...

```
A-000 LBLA
A-001 PRTX
A-002 X²
A-003 XZ Y
A-004 PRTX
A-005 X²
A-006 +
A-007 JX
A-008 PRTX
```

... to this.

```
A-000 LBLA
A-001 PRTX
A-002 X²
A-003 XZ Y
A-004 X²
A-005 +
A-006 JX
A-007 PRTX
```

To delete the **PRINTx** instruction now loaded in step A-001, you can use the **BST** function to back step to that step number, and then delete the instruction with the **DELETE** operation.

Press	Display
f BST	A-003 11
f BST	A-002 42
f BST	A-001 14
f DELETE	A-000 lbl A

The **PRINTx** instruction, keycode 14, is deleted from step A-001 and the calculator displays step A-000. Subsequent instructions move up one step in program **A**.

If you have modified the program as described above, the HP-95C should now print only the computed value for the hypotenuse, side *c*, of a right triangle when sides *a* and *b* are input to the X- and Y-registers of the stack and the Pythagorean Theorem program is run.


Slide the PRGM-RUN switch **PRGM**  **RUN** to **RUN**, and run the program for right triangles with:

Sides <i>a</i> and <i>b</i> of 17 and 34 meters.	Answer for side <i>c</i> is 38.01 meters.	38.01 ***
Sides <i>a</i> and <i>b</i> of 5500 rods and 7395 rods.	Answer for side <i>c</i> is 9216.07 rods.	9216.07 ***

To replace any instruction with another, simply set the calculator to the desired step of program memory, press **f** **DELETE** to delete the first instruction, then press the keystrokes for the new instruction.

Using the Printer for Editing

In the TRACE position of the Print Mode switch, the printer preserves a record of every instruction executed in a running program, as well as all intermediate and final answers. This feature is a valuable aid in debugging and editing programs. To see how the printer reproduces the action of a program, slide the Print Mode switch to TRACE and run the Pythagorean Theorem program to solve for a triangle with sides *a* and *b* of 11282 kilometers and 65482.448 kilometers:

Slide the Print Mode switch **MAN**  **NORM** to **TRACE**.

Press	Display	
	11282.00	ENT↑
	65482.448	A
	A-001	X ²
	4287950996.	***
	A-002	X ² Y
	11282.00	***
	A-003	X ²
11282 ENTER↑	11282.00	
65482.448	65482.448	
A	66447.23	Kilometers.
	127283524.0	***
	A-004	+
	4415234520.	***
	A-005	JX
	66447.23	***
	A-006	PRTX
	66447.23	***

The printer shows every keystroke, every instruction executed, and, where calculated, every intermediate and final result.

When a program halts in the middle of execution because of an error or because of an overflow, you can slide the PRGM-RUN switch to PRGM to see the step number and keycode of the instruction that caused the error or overflow. It may be more helpful, however, to run the program with the Print Mode switch set to TRACE so that you chart the events, step-by-step, that led to the error. With the printer set to the TRACE mode, you can print the operation of the entire program, or, by first addressing the desired beginning step of the program with **SST** or **BST** you can print only a portion of the operation of the program if you desire.

You can use the printer in the TRACE mode in conjunction with **SST** to slow down execution even more. With the Print Mode switch set to TRACE, each time you press the **SST** key, one instruction is executed and the instruction and any results are also printed. With this feature, you can examine your programs step-by-step with a fine-toothed comb!

Problems

- You may have noticed that there is a single keyboard operation, **g** **⊕P**, that calculates the hypotenuse, side c , of a right triangle with sides a and b input to the X- and Y-registers. Replace the **x'**, **x²y**, **x'**, **+**, and **√x** instructions in the Pythagorean Theorem program with the single **⊕P** instruction, as follows:
 - Use the **f** PRINT **A** operation to verify that the Pythagorean Theorem program in your HP-95C contains the instructions shown below:

A-000	LBLA	
A-001	X ²	} Replace all of these instructions with a g ⊕P instruction.
A-002	X ² Y	
A-003	X ²	
A-004	+	
A-005	JX	
A-006	PRTX	

b. Use the **SST** keyboard operation to go to step A-005, the last instruction to be deleted in the program.

c. Use the **f** **DELETE** keyboard operation in PRGM mode to delete the instructions in steps A-005, A-004, A-003, A-002, and A-001.

Note: When modifying a program, you should always *delete* instructions before you *add* others, to ensure that program memory is not filled, resulting in an error message.

d. Load the **9** **→P** instruction into step A-001.

e. Verify that the modified program looks like the one below.

```
A-000 LBLA
A-001 →P
A-002 PRTX
```

f. Switch to RUN mode and run the program for a right triangle with sides a and b of 73 feet and 112 feet.

(Answer: 133.69 feet)

2. The following program is used by the manager of a savings and loan company to compute the future amounts of savings accounts according to the formula $FV = PV(1 + i)^n$, where FV is future value or amount, PV is present value, i is the periodic interest rate expressed as a decimal, and n is the number of periods. With PV entered into the Y-register, n keyed into the X-register, and an annual standard interest rate of 7.5%, the program is:

```
B-000 LBLB
B-001 1
B-002 ENT↑
B-003 .
B-004 0
B-005 7
B-006 5
B-007 +
B-008 X↔Y
B-009 Y^X
B-010 x
```

a. Load the program into the calculator.

b. Run the program to find the future amount of \$1,000 invested for 5 years.

(Answer: \$1,435.63)

Of \$2,300 invested for 4 years.

(Answer: \$3,071.58)

- c. Alter the program to account for a change of the annual interest rate from 7.5% to 8%, and add a **PRINT x** instruction so that the answer (future value) is printed.
- d. Run the program for the new interest rate to find the future value of \$500 invested for 4 years; of \$2,000 invested for 10 years.

(Answers: \$680.24; \$4,317.85)

3. The following program calculates the time it takes for an object to fall to the earth when dropped from a given height. (Friction from the air is not taken into account.) When the program is initialized by keying the height h in meters into the displayed X-register and **C** is pressed, the time t in seconds the object takes to fall to earth is computed according to the formula

$$t = \sqrt{\frac{2d}{9.8 \text{ meters/s}^2}}$$

- a. Clear all previously recorded programs from the calculator and load the program below.

```
C-000 LBL C
C-001 ENT↑
C-002 2
C-003 ×
C-004 9
C-005 .
C-006 8
C-007 ÷
C-008 √X
```

- b. Run the program to compute the time taken by a stone to fall from the top of the Eiffel Tower, 300.51 meters high. From a blimp stationed 1000 meters in the air.

(Answers: 7.83 seconds, 14.29 seconds)

- c. Alter the program to compute the time of descent when the height in *feet* is known, according to the formula

$$t = \sqrt{\frac{2d}{32.1740 \text{ feet/s}^2}}$$

- d. Run the altered program to compute the time taken by a stone to fall from the top of the Grand Coulee Dam, 550 feet high. From the 1350-foot height of the World Trade Center buildings in New York City.

(Answers: 5.85 seconds, 9.16 seconds)

PRINT CLEAR

\sqrt{x} x^y y^x $\frac{1}{x}$

$x=y$ $x=0$ $x\neq y$ $x\neq 0$ $x\leq y$ $x<0$ $x>y$ $x>0$

$\%$ $\frac{1}{\%}$ \sin \cos

$\Delta\%$ $\Delta\%$ \sin \cos

$\frac{1}{x}$ $\frac{1}{x}$ **GTO** $\frac{1}{x}$

IBI JUMP PAUSE

Branching

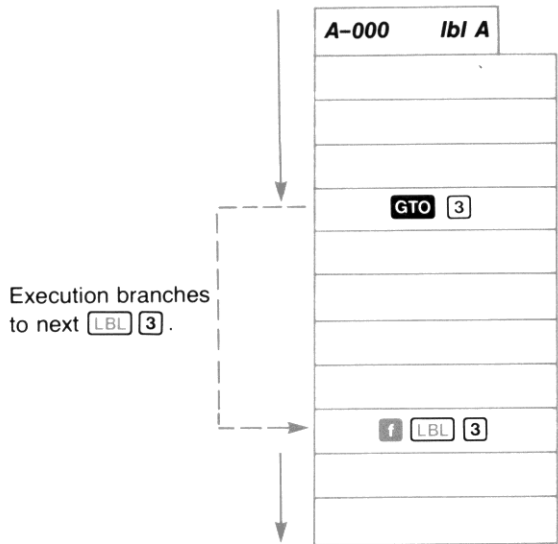
Unconditional Branching and Looping

You have seen how you can use the **GTO** (*go to*) operation from the keyboard to set the calculator to the beginning of any program. You can also use the **GTO** instruction as part of a program. When executed in a program, **GTO** can transfer execution back to the beginning of that program, or it can transfer execution to any *label* (0 through 9) in that program.

Using Labels

To define *routes*—that is, parts of programs that you may want to address—you can use the **f LBL** (*label*) function followed by a digit key (0 through 9). Thus, each program can contain as many as 10 different labels.

When the calculator is executing a program and encounters a **GTO** instruction followed by a label designator (0 through 9), it immediately halts execution and begins *searching* sequentially downward through that program for that label, then resumes execution. For example, if the calculator were executing a program and encountered the instruction **GTO 3**, it would immediately halt execution, search sequentially downward for **f LBL 3**, and resume execution.




A **GTO** instruction used this way is known as an *unconditional branch*. It always *branches* execution from the **GTO** instruction to the specified label. (Later, you will see how a conditional instruction can be used in conjunction with a **GTO** instruction to create a *conditional branch*—a branch that depends on the outcome of a test.)

Neither searching nor execution are transferred out of a given program by a **GTO n** instruction executed in a program. Thus, when the calculator is searching for a label and reaches the end of the program, it resumes searching at the top of *that* program. It does not search in another program, so you can use the complete set of labels 0 through 9 in program **A**, then use another set of labels 0 through 9 in program **B**, etc.

If the calculator attempts to execute a **GTO** **[n]** (or **GSB** **[n]**—more about **GSB** later) instruction for which there is no label address **[n]** in that program, execution halts and the calculator displays a **SEE 4** error message.


A common use of a branch is to create a “loop” in a program. For example, the following program calculates and prints the square roots of consecutive whole numbers beginning with the number 1. The HP-95C continues to compute the square root of the next consecutive whole number until you press **R/S** (or any key) to stop program execution (or until the calculator overflows).

To key in the program:

Slide the PRGM-RUN switch **PRGM**  **RUN** to **PRGM**.

Press Display

9 CLEAR D	<i>d-001</i> lbl d	Program D cleared of any previous programs, and calculator set to step d-000.
0	<i>d-001</i> 0	
STO 1	<i>d-002</i> 35 1	
f LBL 7	<i>d-003</i> 61 63 7	
1	<i>d-004</i> 1	
STO + 1	<i>d-005</i> 35 59 1	
RCL 1	<i>d-006</i> 45 1	
PRINT x	<i>d-007</i> 14	
√x	<i>d-008</i> 41	
PRINT x	<i>d-009</i> 14	
f SPACE	<i>d-010</i> 61 14	
GTO 7	<i>d-011</i> 63 7	

To run the program, slide the PRGM-RUN switch **PRGM**  **RUN** to **RUN** and press **D**. The calculator will begin printing a table of integers and their square roots, and will continue until you press any key from the keyboard to stop the program, or until the calculator overflows.

How it works: When you press **D**, the calculator begins executing instructions beginning with the beginning of program **D**. It executes each subsequent instruction in the program until it reaches step d-011, the **GTO** 7 instruction.

The **GTO** 7 instruction causes the calculator to *search* for the **LBL** 7 instruction in that program, program **D**. When the calculator encounters the **LBL** 7 instruction loaded in step d-003, execution begins again from that **LBL** 7 instruction. (Notice that the address after a **GTO** instruction in a program is a *label*, not a step number.)

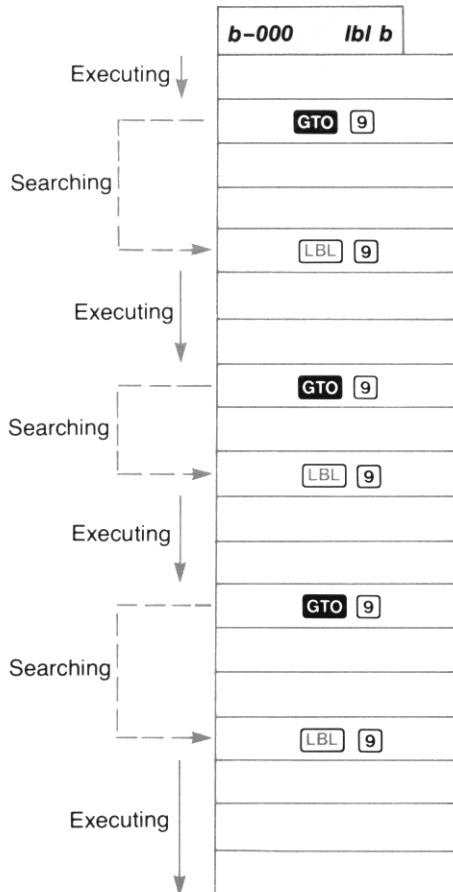
Since execution is transferred to the **LBL** 7 instruction in step d-003 each time the calculator executes the **GTO** 7 instruction in step d-011, the calculator will remain in this "loop," continually adding one to the number in storage register R_1 and printing the new number and its square root.

Looping techniques like the one illustrated here are common and extraordinarily useful in programming. By using loops, you take advantage of one of the most powerful features of the HP-95C—the ability to update data and perform calculations automatically, quickly, and, if you so desire, endlessly.

```

D-000 LBLD
D-001 0
D-002 S 1
D-003 LBL7
D-004 1
D-005 S+1
D-006 R 1
D-007 PRTX
D-008 JX
D-009 PRTX
D-010 SPC
D-011 GTO7
    
```

You can use unconditional branches to create a loop, as shown above, or in any part of a program where you wish to transfer execution to another label. When the calculator executes a **GTO** instruction, it searches sequentially downward through program memory and begins execution again at the first specified label it encounters.

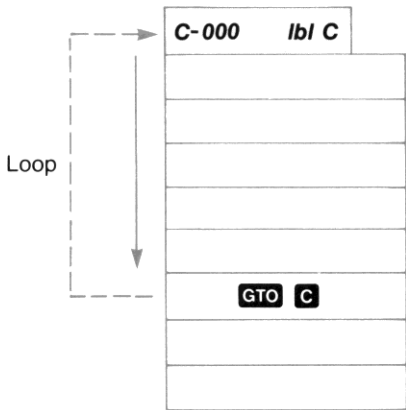


Since the calculator always searches sequentially downward through program memory for the next designated label, you could use the same label several times in a single program, as illustrated here.

When you pressed **B** from the keyboard, the calculator would begin executing program **B** until it executed the first **GTO 9** instruction, whereupon it would search downward through the program for the next **LBL 9** instruction before resuming execution. When the calculator then executed the second **GTO 9** instruction, it would search downward again until it encountered another **LBL 9** address, then would resume executing instructions. This would be repeated for the third **GTO 9** and **LBL 9** instructions.

Transferring Execution Back to the Beginning of a Program

In addition to the 10 labels to which **GTO** can transfer execution in a program, you can also use the instruction **GTO** followed by the appropriate program designator (**A**, **B**, **C**, or **D**) to transfer execution back to the beginning of a running program. Execution then *continues* from the top of the program—it does not halt. Thus, you could set up a loop in, say, program **C**, like the one shown here:

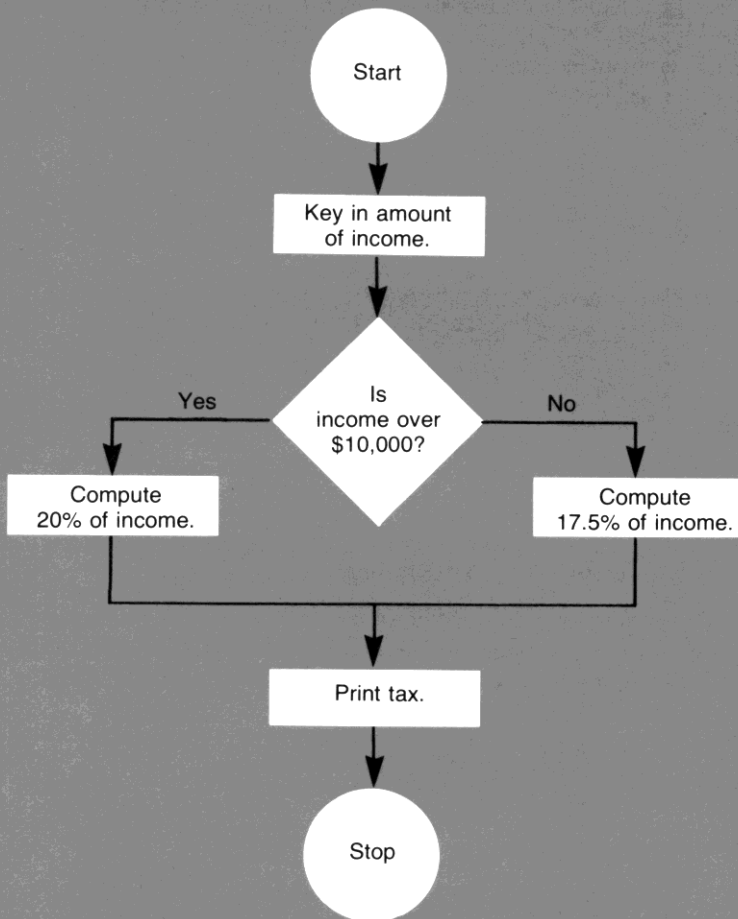


Note: When executed as an instruction in a program, **GTO** can actually be followed by *any* program designator (**A**, **B**, **C**, or **D**). More about using **GTO** to transfer execution from one program to another in section 13, Transferring Execution Between Programs.






As you will remember, you can press **GTO** followed by **A**, **B**, **C**, or **D** *from the keyboard* in RUN mode to set the calculator to the top of the designated program without executing any instructions or running the program.







Conditionals and Conditional Branches

Often there are times when you want a program to make a decision. For example, suppose an accountant wishes to write a program that will calculate and print the amount of tax to be paid by a number of persons. For those with incomes of \$10,000 per year or under, the amount of tax is 17.5%. For those with incomes of over \$10,000, the tax is 20%. A flowchart for the program might look like this:

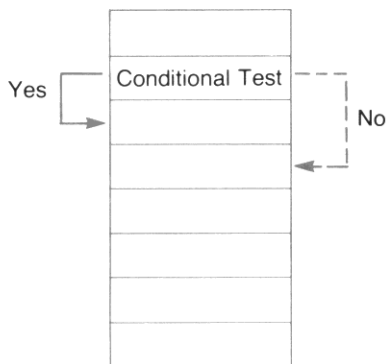


The *conditional* operations on your HP-95C keyboard are useful as program instructions to allow your calculator to make decisions like the one shown above. The eight conditionals that are available on your HP-95C are:

-  $X \neq Y$ tests to see if the value in the X-register is unequal to the value in the Y-register.
-  $X = Y$ tests to see if the value in the X-register is equal to the value in the Y-register.
-  $X > Y$ tests to see if the value in the X-register is greater than the value in the Y-register.
-  $X \leq Y$ tests to see if the value in the X-register is less than or equal to the value in the Y-register.
-  $X \neq 0$ tests to see if the value in the X-register is unequal to zero.

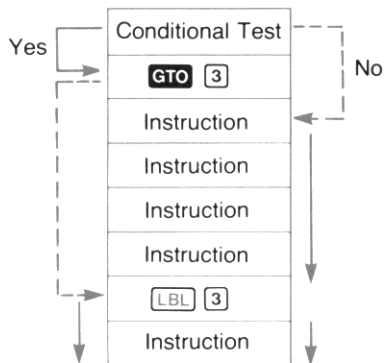
-   tests to see if the value in the X-register is equal to zero.
-   tests to see if the value in the X-register is greater than zero.
-   tests to see if the value in the X-register is less than zero.

Each conditional essentially asks a question when it is encountered as an instruction in a program. If the answer is YES, program execution continues sequentially downward with the next instruction in program memory. If the answer is NO, the calculator branches *around* the next instruction. For example:



You can see that after it has made the conditional test, the calculator will *do* the next instruction if the test is *true*. This is the “DO if TRUE” rule.

The step immediately following the conditional test can contain any instruction. The most commonly used instruction, of course, will be a **GTO** instruction. This will branch program execution to another section of program memory if the conditional test is true.



Now let's look at that accountant's problem again. For persons with incomes of more than \$10,000 he wants to compute a tax of 20%. For persons with incomes of \$10,000 or less, the tax is 17.5%. The following program will test the amount in the X-register and compute and print the correct percentage of tax.

To key in the program:

Slide the PRGM-RUN switch  RUN to PRGM.

Press	Display		
9 CLEAR A	A-000	lbl A	Program A cleared of previous instructions, and calculator set to top of that program.
PRINT x	A-001	14	Prints amount of income.
EEX	A-002	23	} Amount of \$10,000 placed in Y-register.
4	A-003	4	
X↔Y	A-004	11	
f X>Y	A-005	61 44	} If amount of income is greater than \$10,000, go to portion of program defined by label 1 .
GTO 1	A-006	63 1	
1	A-007	1	} Tax percentage for this portion of the program is 17.5.
7	A-008	7	
.	A-009	66	
5	A-010	5	
GTO 2	A-011	63 2	} Tax percentage for this portion of the program is 20.
f LBL 1	A-012	61 63 1	
2	A-013	2	
0	A-014	0	
f LBL 2	A-015	61 63 2	
%	A-016	51	
PRINT x	A-017	14	

To run the program to compute taxes on incomes of \$15,000 and \$7,500:

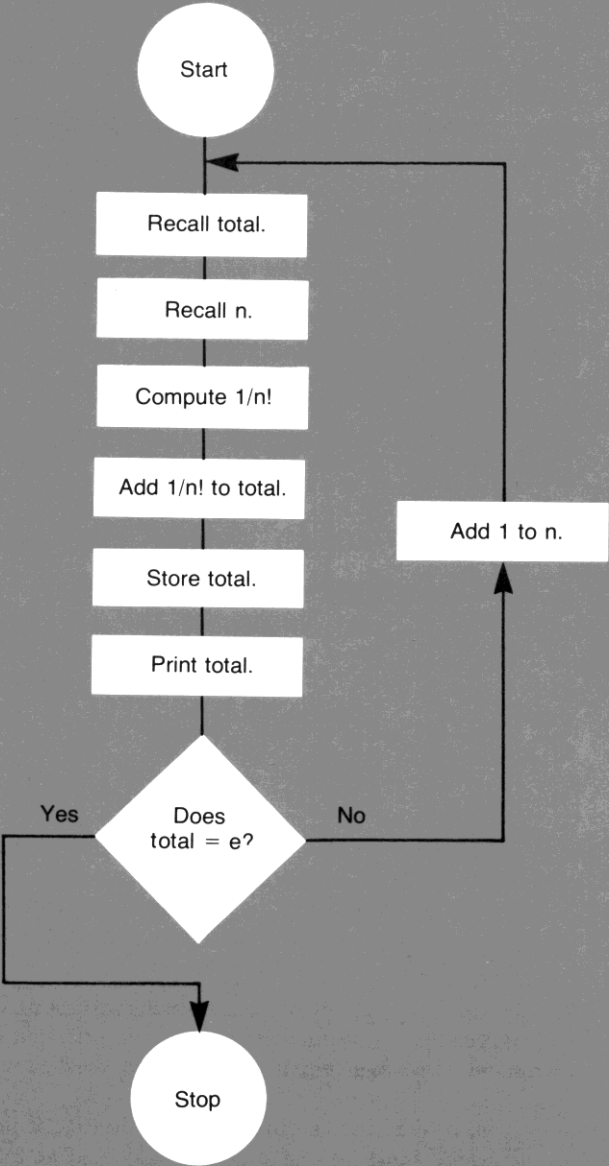
Slide the PRGM-RUN switch  RUN to RUN.

Press	Display	
		15000.00 ***
15000 A	3000.00	3000.00 ***
7500 A	1312.50	7500.00 ***
		1312.50 ***


Another place where you often want a program to make a decision is within a loop. The loops that you have seen have to this point been *infinite* loops—that is, once the calculator begins executing a loop, it remains locked in that loop, executing the same set of instructions over and over again, forever (or, more practically, until the calculator overflows or you halt the running program by pressing **R/S** or any other key).




You can use the decision-making power of the conditional instructions to shift program execution out of a loop. A conditional instruction can shift execution out of a loop after a specified number of iterations or when a certain value has been reached within the loop.
























Example: As you know, your HP-95C contains the value for e , the base of natural logarithms. (You can display the calculator's value for e by pressing **1** **9** **e^x** .) The following program uses the series $e = 1/0! + 1/1! + 1/2! + \dots + 1/n!$ to *approximate* the value for e . After each iteration through the loop, the latest approximation is printed and compared to the calculator's value for e . When the two values are equal, execution is transferred out of the loop to stop the program.






To load the program into the calculator:


Slide the PRGM-RUN switch PRGM  RUN to PRGM.

Press	Display	
 CLEAR 	b-000	<i>lbl b</i> Clears previous instructions from program  and sets calculator to top of that program.

 1	b-001	45 1
 0	b-002	45 0
 	b-003	62 2
	b-004	44
	b-005	59
  9	b-006	61 21 9
 1	b-007	35 1
	b-008	14
1	b-009	1
 	b-010	62 4
 	b-011	61 41
 	b-012	63 7
1	b-013	1
  0	b-014	35 59 0
 	b-015	63 b
  	b-016	61 63 7

Notice that a   instruction is used to transfer execution back to the top of the program and *continue* execution if the total does not equal *e*. When the total *does* equal *e*, execution transfers to the end of the program where execution stops.

To initialize the program ensure that the storage registers are cleared. Then press  to run the program.

First, slide the PRGM-RUN switch PRGM  RUN to RUN.

Press	Display
-------	---------

 CLEAR 	0.00
---	------

Ensures that storage registers are cleared to zero initially. (Display assumes no results remain from previous examples.)

	2.718281828
---	-------------

1.000000000	***
2.000000000	***
2.500000000	***
2.666666667	***
2.708333334	***
2.716666667	***
2.718055556	***
2.718253969	***
2.718278771	***
2.718281527	***
2.718281803	***
2.718281828	***

You can see that execution continues within the loop until the approximation for e equals the calculator's value for e . When the instruction $\boxed{X=Y}$ in step b-011 is finally true, execution is transferred out of the loop by the subsequent $\boxed{GTO} \boxed{7}$ instruction to the last step of the program, $\boxed{LBL} \boxed{7}$. After executing the last step of the program, execution halts.

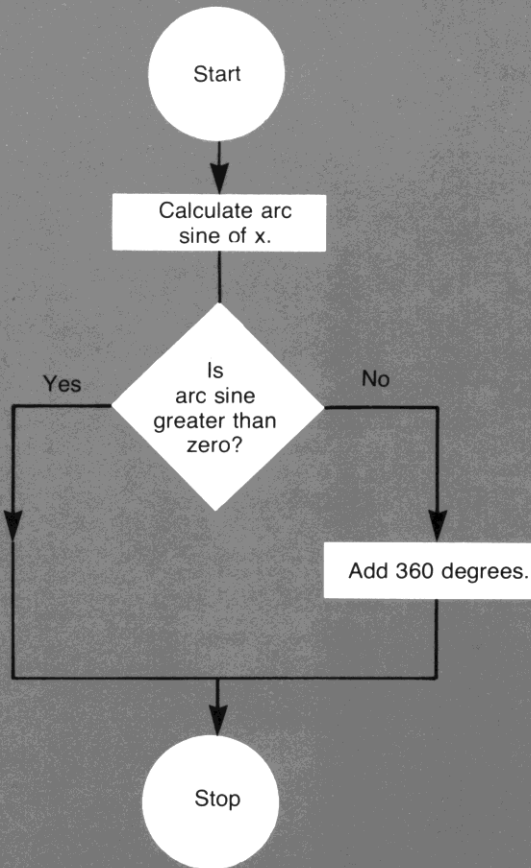
Problems

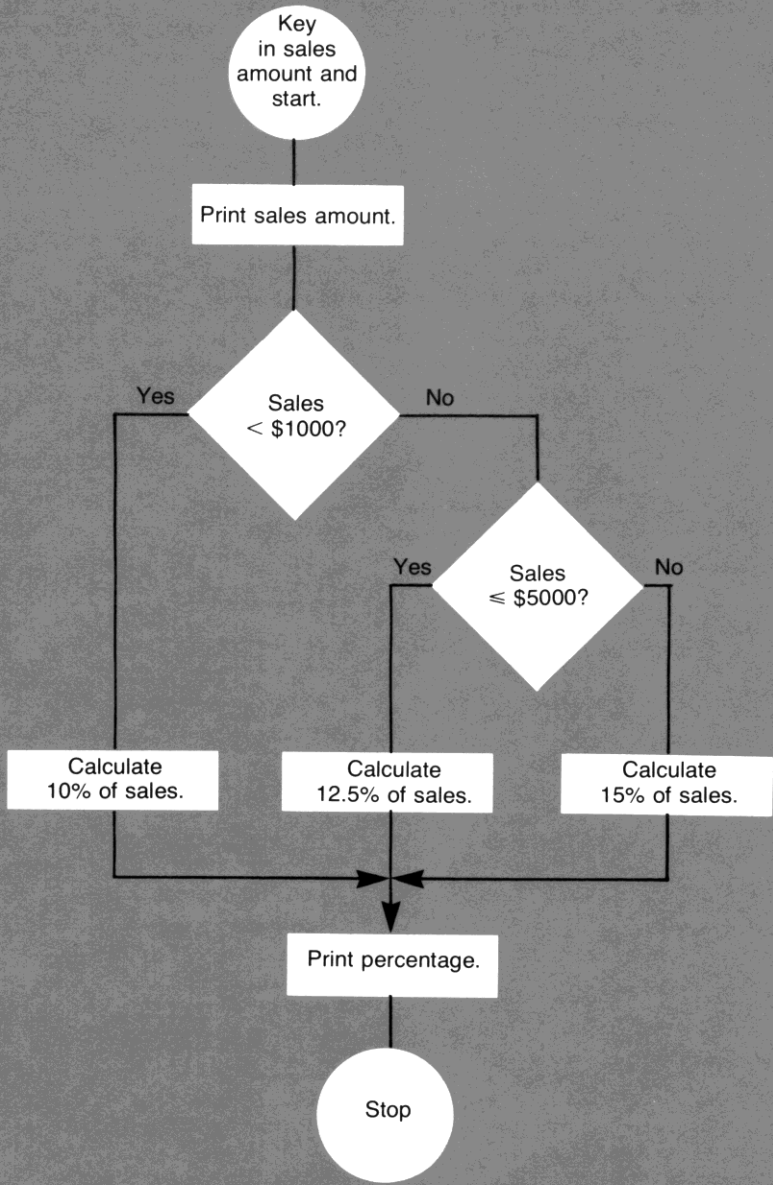
1. The following program calculates and prints the square of the number 1 each time it is run. Key the program in with the PRGM-RUN switch $\boxed{\text{PRGM}} \boxed{\text{RUN}}$ set to PRGM, then switch to RUN and run the program a few times to see how it works. Finally, modify the program by inserting a $\boxed{LBL} \boxed{3}$ instruction after the $\boxed{STO} \boxed{1}$ instruction in step C-002, and a $\boxed{GTO} \boxed{3}$ instruction after the $\boxed{\text{PRINT}} \boxed{\text{SPACE}}$ instruction. This should create a loop that will continually print a new number and its square, then increment the number by 1, print it, and compute and print *its* square, etc. To load the original program, before modification:

Slide the PRGM-RUN switch $\boxed{\text{PRGM}} \boxed{\text{RUN}}$ to PRGM.

Press	Display	
$\boxed{9} \boxed{\text{CLEAR}} \boxed{\text{C}}$	C-000	<i>lbl C</i> Clears previous instructions from program $\boxed{\text{C}}$.
0	C-001	0
$\boxed{\text{STO}} \boxed{1}$	C-002	35 1
1	C-003	1
$\boxed{\text{STO}} \boxed{+} \boxed{1}$	C-004	35 59 1
$\boxed{\text{RCL}} \boxed{1}$	C-005	45 1
$\boxed{\text{PRINT}} \boxed{x}$	C-006	14
$\boxed{x^2}$	C-007	42
$\boxed{\text{PRINT}} \boxed{x}$	C-008	14
$\boxed{\text{f}} \boxed{\text{SPACE}}$	C-009	61 14

2. Write a program that will calculate the arc sine (that is, \sin^{-1}) of a value that has been keyed into the displayed X-register. Test the resulting angle with a conditional, and if it is negative or zero, add 360 degrees to it to make the angle positive. Use the flowchart below to help you write the program:





3. Use the flowchart on page 152 to help you write a program that will allow a salesman to compute his commissions at the rates of 10% of sales of up to \$1000, 12.5% for sales of \$1000 to \$5000, and 15% for sales of over \$5000. The program should print the amount of sales and the amount of commission.

Load the program and run it for sales amounts of \$500, \$1000, \$1500, \$5000, and \$6000.

(Answers: \$50.00, \$125.00, \$187.50, \$625.00, \$900.00)

4. Use the flowchart on page 154 to create a program that computes and prints the future value (FV) of a compound interest savings account in increments of one year, according to the formula:

$$FV = PV(1 + i)^n$$

where FV = future value of the savings account.



PV = present value (or principal) of the account.

i = interest rate (expressed as a decimal fraction; e.g., 6% is expressed as 0.06).

n = number of compounding periods (usually, years).

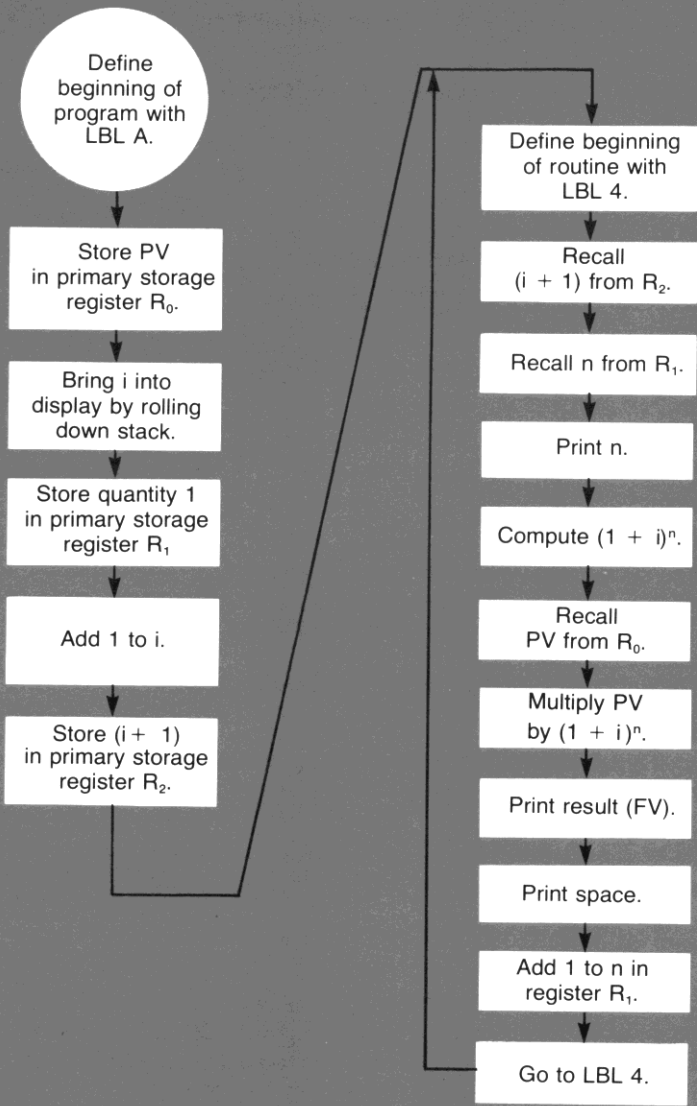
Assume that program execution will begin with i entered into the Y-register of the stack and with PV keyed into the displayed X-register.

After you have written and loaded the program, run it for an initial interest rate i of 6% (keyed in as .06) and an initial deposit (or present value, PV) of \$1000.

(Answer: 1st year, \$1060; 2nd year, \$1123.60; 3rd year, \$1191.02, etc.)

The program will continue running until you press **RS** (or any key) or the HP-95C overflows. You can see how your savings would grow from year to year. Try the program for different interest rates i and values of PV .

When you are satisfied that the program is running correctly, modify it using a conditional instruction so that you can store a limit for n in a storage register and have the program stop execution after that limit is reached.



5. Create and load into the calculator a program that will print a table of consecutive integers and their common logarithms. The program should begin with the number 1, and continue printing each number and its common logarithm until an upper limit is reached, at which time the program should stop. An example of the printed table with an upper limit of 5 is shown below.

1.000000000	***
0.000000000	***
2.000000000	***
0.301029996	***
3.000000000	***
0.477121255	***
4.000000000	***
0.602059991	***
5.000000000	***
0.698970004	***

Run the program with a few upper limits to see that it works correctly.



PRINT
CLEAR

A B C D
PRINT CLEAR

\sqrt{x} x^2 y^x $1/x$
 $x=y$ $x=0$ $x\neq y$ $x\geq 0$ $x\leq y$ $x<0$ $x>y$ $x\geq 0$

% $\frac{1}{x}$ GDB SST
%2 $\Delta\%$ ISZ DSZ RTN BST

f 9 GTO R/S

LBL JUMP PAUSE

Program Interruptions

In your programs, there may often be occasions when you want a program to halt during execution so that you can key in data, or to pause so that you can quickly view results before the program automatically resumes running. Besides illustrating the two functions, **R/S** and **PAUSE**, that are used for program interruptions, this section also shows you how the keyboard can be used to halt program execution, and how an error will halt a running program.

Using

The **R/S** (*run/stop*) function can be used either as an instruction in a program or pressed from the keyboard.

When pressed from the keyboard:

1. If a program is running, **R/S** stops the program.
2. If a program is stopped or not running, and the calculator is in RUN mode, **R/S** starts the program running beginning with the current location in program memory.

When executed as an instruction during a running program, **R/S** stops program execution after its step of program memory. If **R/S** is then pressed from the keyboard, execution begins with the current step of program memory. (When **R/S** is pressed, it displays the step number and keycode of that current step—when released, execution begins with that step.)

You can use these features of the **R/S** instruction to stop a running program at points where you want to key in data. After the data has been keyed in, restart the program using the **R/S** key from the keyboard.

Example: The following program lets you key in a percentage discount and calculates the cumulative cost of various quantities of differently priced items from which the discount has been subtracted. **R/S** instructions are inserted in the program to allow you to key in data at various points.

To key in the program:

Slide the PRGM-RUN switch  RUN to PRGM.

Press	Display	
9 CLEAR D	d-000	lbl d Clears any previous instructions from program D .
9 CLEAR REG	d-001	62 12
STO 0	d-002	35 0 Store discount percentage in R ₀ .
f LBL 9	d-003	61 63 9
R/S	d-004	64 Stop to key in quantity.
ENTER ↑	d-005	21
R/S	d-006	64 Stop to key in price.
x	d-007	39
RCL 0	d-008	45 0
%	d-009	51
-	d-010	49
STO + 1	d-011	35 59 1 Add to running total in R ₁ .
RCL 1	d-012	45 1 Recall running total for display.
GTO 9	d-013	63 9

Now run the program to calculate the cumulative total of the following purchases at a discount of 15%:

Quantity	Price of Each
5	\$ 7.35
7	\$12.99
14	\$14.95

Then run the program to calculate the cumulative total of the following purchases at a discount of 25%:















Quantity	Price of Each
7	\$ 4.99
12	\$ 1.88
37	\$ 8.50



In order to calculate the cumulative total for each percentage of discount, merely key in the percentage value and press **D**. When the calculator stops executing the first time, key in the quantity of an item and press **R/S** from the keyboard to resume execution.

When the program stops a second time, key in the price of that item and again press **R/S** to resume program execution from that point.





To run the program:





Slide the PRGM-RUN switch  RUN to RUN.



Press	Display	
15	15.	Key in percentage of discount.
	15.00	
5 	5.00	The first quantity.
7.35 	31.24	Running total.
7 	7.00	
12.99 	108.53	Running total
14 	14.00	
14.95 	286.43	Cumulative cost for items at 15% discount.
25	25.	Percentage of discount.
	25.00	
7 	7.00	The first quantity.
4.99 	26.20	Running total.
12 	12.00	
1.88 	43.12	Running total.
37 	37.00	
8.50 	278.99	Cumulative cost for items at 25% discount.

If you have a number of halts for data entries like the ones shown here, it may be helpful to “identify” each step by recording a familiar number into the program immediately before each  instruction. When the calculator then stops execution because of the  instruction, you can look at the displayed X-register to see the “identification number” for the required data input at that point. For example, if your program contained eight stops for data inputs, it might be helpful to have the numbers 1 through 8 appear so that you would know which input was required each time. (Don’t forget that the “identification number” will be pushed up into the Y-register of the stack when you key in a new number.)

Pausing to View Output

You now know two instructions that will slow or halt a running program for data output— and .  can print the value contained in the X-register at any point in the program, while  stops a running program and allows you to view results in the display.

Another instruction that can be used to slow a running program to view output is the  instruction. An   instruction executed in a program momentarily interrupts program execution. The length of the pause is about one second, although more  instructions in subsequent steps of program memory can be used to lengthen viewing time, if desired.

You can use   in a program to monitor the operation of the program without printing every result.

Example: Named after a 13th-century mathematician, the Fibonacci series is a series of numbers that expresses many relationships found in mathematics, architecture, and nature. (For example, in many plants, the proliferation of branches follows a series of Fibonacci numbers.) The series is of the form 0, 1, 1, 2, 3, 5, 8, 13..., where each element is the sum of the two preceding elements.




The following program contains a loop that generates the next Fibonacci element, pauses to display it, then generates still another Fibonacci element and pauses to display *that*, etc. The loop is an infinite one, and the program will run, continually displaying the next Fibonacci element, until you stop the program by pressing **R/S** (or any key) from the keyboard.

To key in the program:

Slide the PRGM-RUN switch PRGM  RUN to PRGM.

Press	Display		
g CLEAR B	<i>b-001</i>	<i>lbl b</i>	Clears previous instructions from program B .
0	<i>b-001</i>	0	
STO 0	<i>b-002</i>	35 0	
1	<i>b-003</i>	1	
STO 1	<i>b-004</i>	35 1	
f PAUSE	<i>b-005</i>	61 64	
f LBL 0	<i>b-006</i>	61 63 0	} Infinite loop.
RCL 0	<i>b-007</i>	45 0	
RCL 1	<i>b-008</i>	45 1	
+	<i>b-009</i>	59	
f PAUSE	<i>b-010</i>	61 64	
STO 0	<i>b-011</i>	35 0	
RCL 0	<i>b-012</i>	45 0	
RCL 1	<i>b-013</i>	45 1	
+	<i>b-014</i>	59	
f PAUSE	<i>b-015</i>	61 64	
STO 1	<i>b-016</i>	35 1	
GTO 0	<i>b-017</i>	63 0	

Now switch to RUN mode and run the program. Press **R/S** (or any key) to stop the program after you have seen how quickly the Fibonacci series increases. To run the program: Slide the PRGM-RUN switch **PRGM**  **RUN** to RUN.

Press	Display
B	1.00
	1.00
	2.00
	3.00
	5.00
	8.00
	13.00
	21.00
	34.00
	55.00
	89.00
R/S	144.00

The **PAUSE** instruction permits you to see that each element in the Fibonacci series is the sum of the previous two elements in the series.

Keyboard Stops

As you know, pressing any key from the keyboard during a running program halts that program. The program may halt after any step—if you slide the PRGM-RUN switch to PRGM after a program is halted, you will see the step number and keycode of the last step that was executed.

The HP-95C has been designed so that program execution will *not* halt in the middle of a digit entry sequence. If you press any key while a number is being placed in the X-register by a running program, the entire number will be “written” and the following step executed by the program before the program halts. For example, in the section of program memory shown below, if you pressed any key from the keyboard while a running program was executing step d-015, execution would continue through step d-019 before execution was halted. If you switched the PRGM-RUN switch to PRGM, you would see the step number and keycode for the last step executed, step d-019.

d-012	X↔Y
d-013	PRINT X
d-014	1
d-015	7
d-016	.
d-017	2
d-018	4
d-019	STO 3
d-020	RCL 2

When a program is halted, you can resume execution by pressing **R/S** from the keyboard in RUN mode. When you press **R/S**, the program resumes execution with the next step as though it had never stopped at all. For example, you can resume execution of the Fibonacci series program now:

Press	Display	
R/S	233.00	The program resumes execution.
	377.00	
R/S	610.00	The program halts again.

Error Stops

If the calculator executes any error-causing operation during a running program, execution immediately halts and the calculator displays the word **SEE** followed by the number of the appropriate error message. To see the step number and keycode of the error-causing instruction, you can briefly slide the PRGM-RUN switch to PRGM.

Sliding the PRGM-RUN switch to PRGM clears the error, as does pressing any key from the keyboard. (The key function is not executed.) You can then resume program execution, if you wish, by pressing **R/S** from the keyboard in RUN mode.

Problems

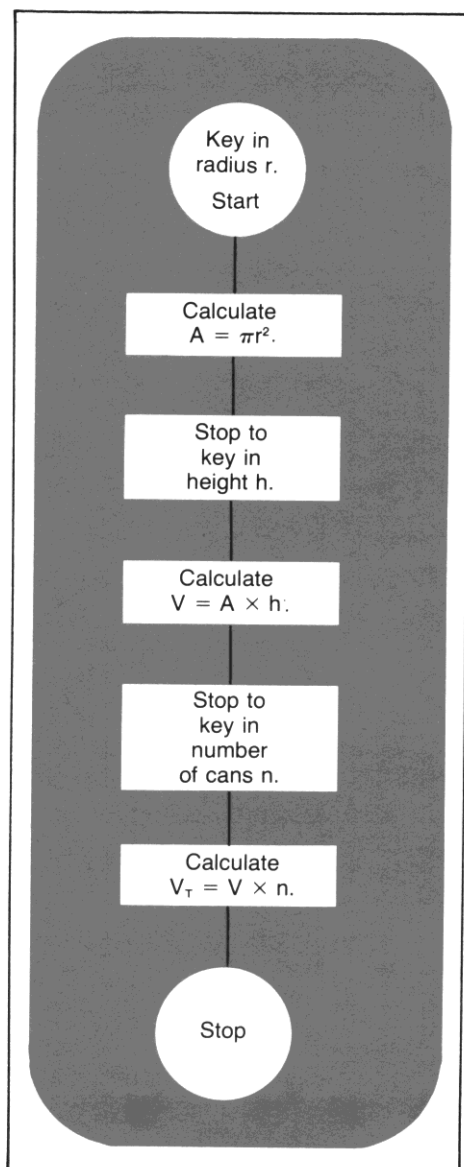
1. Write a program that will use the factorial function **(N!)** to calculate and pause to display the factorials of successive integers beginning with the number 1. (Hint: Place 1 in a storage register, recall it, then use storage register arithmetic to increment the number in the storage register, etc.) The program should contain an infinite loop that you can stop by pressing any key from the keyboard.

Load the program and run it to see that it works correctly.

2. For several different sizes of cans, the foreman at a canning company knows the radius r of the base of the can, the height h of the can, and n , the number of cans of that size. Write a program that will permit the foreman to key in the value for the radius and stop to display the area A of the can. The foreman then keys in the value for the height h , presses **R/S**, and the program calculates the volume V according to the formula $V = A \times h$, stopping to display the value for V . Finally the foreman keys in the number of cans n , presses **R/S** again, and the total volume, V_T , is calculated and displayed.

Use the following flowchart to help you write and load the program. Then run the program for 20,000 cans with heights of 25 centimeters and radii of 10 centimeters; for 7500 cans with heights of 8 centimeters and base radii of 4.5 centimeters.

$$\begin{aligned}\text{Answers: } A &= 314.16 \text{ cm}^2, V = 7853.98 \text{ cm}^3, V_T = 157079632.7 \text{ cm}^3. \\ A &= 63.62 \text{ cm}^2, V = 508.94 \text{ cm}^3, \\ V_T &= 3817035.07 \text{ cm}^3.\end{aligned}$$





A B C D
PRINT CLEAR

\sqrt{x} x^2 y^x $1/x$
 $x=y$ $x=0$ $x \neq y$ $x \neq 0$ $x \leq y$ $x < 0$ $x > y$ $y > 0$

% $\frac{1}{x}$ GSB SST
 $\% \Delta \%$ ISZ DSZ RTN BST

f g STO R/S
LBL JUMP PAUSE

Subroutines

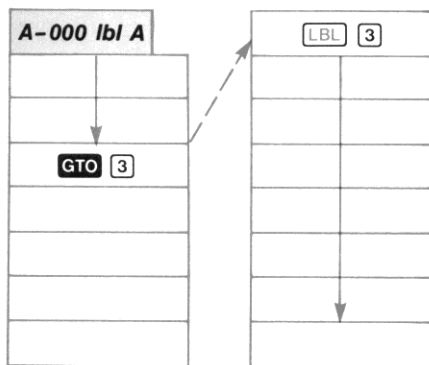
Often, a program contains a certain series of instructions that may be executed several times throughout the program. When the same set of instructions occurs more than once in a program, it can be executed as a subroutine. A subroutine is *selected* by a **GSB** (*go to subroutine*) operation, followed by a label address (**0** through **9**). You can also use an entire program as a subroutine by using the **GSB** instruction followed by one of the program markers (**A**, **B**, **C**, or **D**)—more about the use of entire programs as subroutines later.

A subroutine is defined by a **LBL** instruction at its beginning, just like a routine in a program. The end of a subroutine is defined by a **RTN** (*return*) instruction.

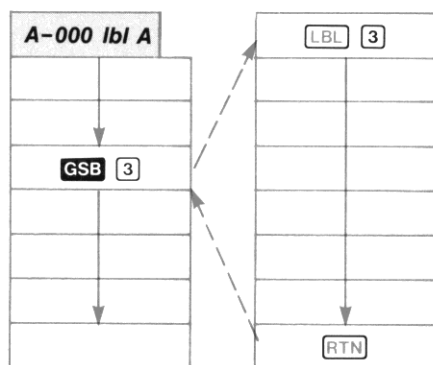
If executed as the result of pressing a user-definable key (**A**, **B**, **C**, or **D**) or as the result of a **GTO** instruction in a program, a **RTN** instruction causes program execution to stop. This is exactly the same as when a program executes the last step of the program and stops.

However, when it is executed as the result of a **GSB** instruction, a **RTN** transfers execution back to the next instruction after the **GSB**. Execution then continues sequentially downward through program memory from the **GSB**. The illustration below should make the distinction between a branch and a subroutine more clear.

Branch



Subroutine



After a **GSB** instruction, the **RTN** returns execution to the step after the **GSB**, where execution continues.

In the illustration on the left, if you pressed **A** from the keyboard, the program would execute instructions sequentially downward through program memory. If it encountered a **GTO 3** instruction, it would then search for the next **LBL 3** and continue execution from there, until it encountered the end of the program (or a **RTN**). Execution would then stop.

However, if the running program encounters a **GSB 3** (*go to subroutine 3*) instruction, as shown on the right, it searches downward for the next **LBL 3** and resumes execution. When it encounters a **RTN** (*return*), program execution is once again transferred, this time back to the point of origin of the subroutine, and execution *resumes* with the next instruction after the **GSB 3**.

When the calculator has executed the instruction in the final step of a program, execution then proceeds just as if it had then executed a **RTN**. If the end of the program was executed as the result of pressing a user-definable key or executing a **GTO** instruction, execution stops.

If the last step of the program was executed as the result of a **GSB** instruction, execution returns to the next step in the program *after* the **GSB** instruction and *continues*.

As you can see, the only difference between a subroutine and normal branch is the transfer of execution *after* the **RTN**. After a **GTO**, the end of a program or the next **RTN** halts a running program; after a **GSB**, the next **RTN** returns execution back to the main program, where it continues until the end of the program (or a **RTN** or **R/S**) is encountered. The same routine may be executed by **GTO** and **GSB** any number of times in a program.

Example: A quadratic equation is of the form $ax^2 + bx + c$. Its two roots may be found by the formulas $r_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$ and $r_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$. Notice the similarity between the solutions for r_1 and r_2 .

Calculate r_1

```
A-000 LBLA
A-001 S 3
A-002 R↓
A-003 S 2
A-004 R↓
A-005 S 1
A-006 R 2
A-007 CHS
A-008 R 2
A-009 X²
A-010 R 1
A-011 R 3
A-012 ×
A-013 4
A-014 ×
A-015 -
A-016 JX
A-017 +
A-018 R 1
A-019 2
A-020 ×
A-021 ÷
A-022 PRTX
```

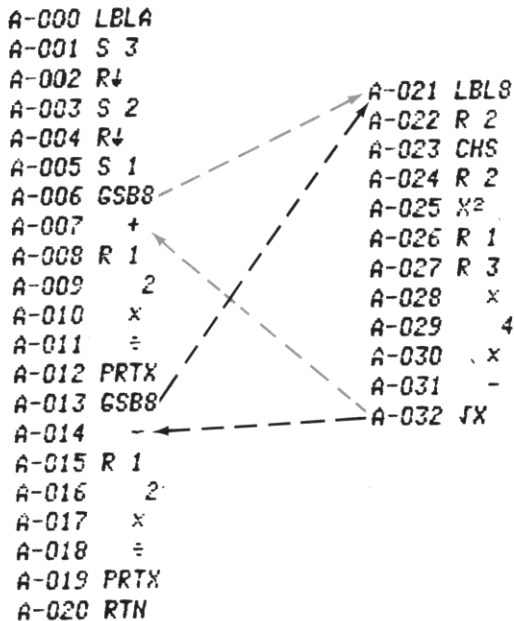
These sections
of program
memory are
identical.

Calculate r_2

```
A-023 R 2
A-024 CHS
A-025 R 2
A-026 X²
A-027 R 1
A-028 R 3
A-029 ×
A-030 4
A-031 ×
A-032 -
A-033 JX
A-034 -
A-035 R 1
A-036 2
A-037 ×
A-038 ÷
A-039 PRTX
```

The program here permits you to enter the values for a , b , and c into the Z-, Y-, and X-registers of the stack. When you press the user-definable **A** key, the resultant roots r_1 and r_2 are calculated and printed by the program.

Notice that the routine for calculating r_1 contains a large section of program memory that is identical to a large section in the routine for calculating r_2 . Since these two sections are identical, you can simply create a *subroutine* that will execute this section of instructions. The subroutine is then called up and executed in both the solution r_1 and the solution for r_2 . This illustrates how the subroutine is used in the program.



With the modified program, you first initialize it by entering the values for a , b , and c to the Z-, Y-, and X-registers of the stack. When you then press **A**, a , b , and c are stored in registers R_1 , R_2 , and R_3 , respectively.

When the **GSB 8** instruction in step A-006 is encountered, execution is transferred to **LBL 8** in step A-021 and computes the quantities $-b$ and $\sqrt{b^2 - 4ac}$, placing them in the X- and Y-registers of the stack, ready for addition or subtraction. When the last step in the program has been executed, execution returns back to the next step after the **GSB** instruction (just as if a **RTN** had been executed), and execution continues with the **+** instruction in step A-007. Thus, the root r_1 is computed and printed.

After printing r_1 , execution continues with the **GSB 8** instruction in step A-013, transfers out to execute the **LBL 8** subroutine, and returns. This time $\sqrt{b^2 - 4ac}$ is subtracted from $-b$, and root r_2 is computed and printed. When the calculator executes the **RTN** in step A-020, execution returns to the top of **A** and halts. By using a subroutine in this program, seven steps of program memory are saved!

To key in the program containing the subroutine:

Slide the PRGM-RUN switch  RUN to PRGM.

Press	Display		
9 CLEAR A	A-000	lbl A	Clears any previous instructions from program A .
STO 3	A-001	35 3	Stores c in R_3 .
R+	A-002	12	
STO 2	A-003	35 2	Stores b in R_2 .
R+	A-004	12	
STO 1	A-005	35 1	Stores a in R_1 .
GSB 8	A-006	53 8	$\left. \begin{array}{l} \text{Calculates } r_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \end{array} \right\}$
+	A-007	59	
RCL 1	A-008	45 1	
2	A-009	2	
x	A-010	39	
÷	A-011	24	
PRINT x	A-012	14	
GSB 8	A-013	53 8	$\left. \begin{array}{l} \text{Calculates } r_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a} \end{array} \right\}$
-	A-014	49	
RCL 1	A-015	45 1	
2	A-016	2	
x	A-017	39	
÷	A-018	29	
PRINT x	A-019	14	
f RTN	A-020	61 53	
f LBL 8	A-021	61 63 8	$\left. \begin{array}{l} \text{Subroutine places } -b \text{ in Y-register} \\ \text{and } \sqrt{b^2 - 4ac} \text{ in X-register, ready} \\ \text{for addition or subtraction.} \end{array} \right\}$
RCL 2	A-022	45 2	
CHS	A-023	22	
RCL 2	A-024	45 2	
x²	A-025	42	
RCL 1	A-026	45 1	
RCL 3	A-027	45 3	
x	A-028	39	
4	A-029	4	
x	A-030	39	
-	A-031	49	
√x	A-032	41	

To initialize the program, you enter a to the Z-register of the stack, b to the Y-register, and key in c to the X-register. Then press **A** to calculate and print the two roots.

Run the program now to find the roots of the equation $x^2 + x - 6 = 0$; of $3x^2 + 2x - 1 = 0$.

To run the program:

Slide the PRGM-RUN switch PRGM  RUN to RUN.

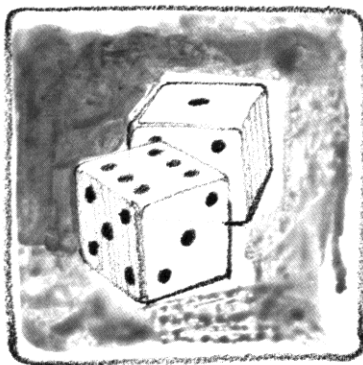
Press	Display	
1 ENTER	1.00	
1 ENTER	1.00	
6 CHS	-6.	2.00 ***
A	-3.00	-3.00 ***
		Roots r_1 and r_2 are calculated and printed.
3 ENTER	3.00	
2 ENTER	2.00	0.33 ***
1 CHS	-1.	-1.00 ***
A	-1.00	r_1 and r_2 are printed.

If the quantity $b^2 - 4ac$ is a negative number, the calculator will display SEE 5 and the running program will stop. For a more efficient and accurate method of finding the roots of a quadratic equation, see the Quadratic Equation program in your *HP-95C Applications Book*.


Routine-Subroutine Usage

Subroutines give you extreme versatility in programming. A subroutine can contain a loop, or it can be executed as part of a loop. Another common and space-saving trick is to use the same routine both as a subroutine and as part of the main program.

Example: The program below simulates the throwing of a pair of dice, printing first the value of one die (an integer from one to six) and then that of the second die (another integer from one to six). The "heart" of the program is a random number generator (actually a pseudo random number generator) that is executed first as a subroutine and then as part of the main program. When you key in a first number (called a "seed"), and press **B**, the digit for the first die is generated and printed using the **4** routine as a subroutine. Then the digit for the second die is generated using the same routine as part of the main program.



To key in the program:

Slide the PRGM-RUN switch PRGM  RUN to PRGM.

Press

Display

g CLEAR B	<i>b-000</i>	<i>lbl b</i>
STO 0	<i>b-001</i>	35 0
f SPACE	<i>b-002</i>	61 14
GSB 4	<i>b-003</i>	53 4
f LBL 4	<i>b-004</i>	61 63 4
RCL 0	<i>b-005</i>	45 0
9	<i>b-006</i>	9
9	<i>b-007</i>	9
7	<i>b-008</i>	7
x	<i>b-009</i>	39
g FRAC	<i>b-010</i>	62 1
STO 0	<i>b-011</i>	35 0
6	<i>b-012</i>	6
x	<i>b-013</i>	39
1	<i>b-014</i>	1
+	<i>b-015</i>	59
f INT	<i>b-016</i>	61 1
f FIX 0	<i>b-017</i>	61 21 0
PRINT x	<i>b-018</i>	14

Previous instructions cleared from program **B**.

LBL 4 executed first as a subroutine.

LBL 4 then executed as a routine.

Now slide the PRGM-RUN switch to RUN and “roll” the dice with your HP-95C. To roll the dice, key in a decimal “seed” (that is, $0 < n < 1$). Then press **B**. The calculator will print the number rolled by the first die, then the number rolled by the second. To make another roll, key in a new seed and press **B** again.

You can play a game with your friends using the “dice.” If your first “roll” is 7 or 11, you win. If it is another number, that number becomes your “point.” You then keep “rolling” (keying in seeds and pressing **B**) until the dice again total your point (you win) or you roll a 7 or an 11 (you lose). To run the program:

Slide the PRGM-RUN switch PRGM  RUN to RUN.

Press

.2315478 B	Your point is 10. \longrightarrow	$\left\{ \begin{array}{ll} 6. & *** \\ 4. & *** \end{array} \right.$
.3335897 B	You missed your point. \longrightarrow	$\left\{ \begin{array}{ll} 4. & *** \\ 1. & *** \end{array} \right.$
.9987562 B	Missed it again. \longrightarrow	$\left\{ \begin{array}{ll} 5. & *** \\ 4. & *** \end{array} \right.$
.9987563 B	Congratulations! You win. \longrightarrow	$\left\{ \begin{array}{ll} 5. & *** \\ 5. & *** \end{array} \right.$

Now try it again.

Press

.21387963 **B**

Your point is 4. → { 2. ***
2. ***

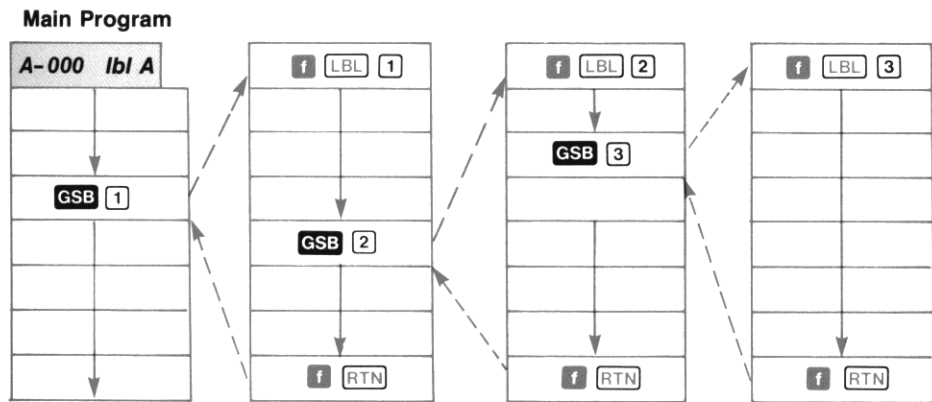
.6658975 **B**

Whoops! You lose. → { 6. ***
1. ***

Subroutine Limits

A subroutine can call up another subroutine, and that subroutine can call up yet another. Subroutine branching is limited only by the number of *returns* that can be held pending by the HP-95C. Three subroutine returns can be held pending at any one time in the HP-95C. The diagram below should make this more clear.

Three returns can be pending.



The calculator can return back to the main program from subroutines that are three deep, as shown. However, if you attempt to call up subroutines that are four deep, the calculator will display the error message **SEE 3**. As always, an error halts a running program at the step that caused the error, and you can clear an error display by pressing any key from the keyboard.

Naturally, the calculator can execute the **RTN** instruction as the result of a **GTO** any number of times without causing an error.

Selecting Subroutines From the Keyboard

Just as you can select any program from the keyboard by pressing the appropriate user-definable key (**A**, **B**, **C**, or **D**), so you can also select any subroutine from the keyboard by pressing **GSB** followed by the digit key (**0** through **9**) of the desired subroutine.

For example, if you pressed **GSB 5** with the calculator set to step A-000 initially, the calculator would search downward through program **A** until it encountered the first **f LBL 5** instruction in that program. Execution would then proceed like a normal subroutine, halting when a **RTN** or the end of the program was executed.

When a running program executes a **R/S** instruction and halts, or when a running program has halted because a key has been pressed from the keyboard, all pending **RTN** instructions are remembered by the calculator. You can then press **R/S** from the keyboard to resume program execution.

However, if you press **A** through **D** or **GSB** **0** through **GSB** **9** from the keyboard, any **RTN** instructions still pending are forgotten by the calculator.

If you are executing a program one step at a time with the **SST** key and encounter a **GSB** instruction, the calculator will execute the entire subroutine before continuing to the next step. If subroutines are nested (that is, a subroutine within a subroutine, etc.), all subroutines will be executed before the calculator moves to the next step.

Problems

- Look closely at the program for finding roots r_1 and r_2 of a quadratic equation (page 167). Can you see other instructions that could be replaced by a subroutine? (Hint: Look at steps A-008 through A-012 and steps A-015 through A-019.) Modify the program by using another subroutine and run it to find the roots of $x^2 + x - 6 = 0$; of $3x^2 + 2x - 1 = 0$.

(Answers: 2, -3; 0.33, -1.)

How many more steps of program memory did you save?

- Create, load, and run a program that will print all permutations of three integers that you have stored in registers R_1 , R_2 , and R_3 . For example, all permutations of the integers 1, 2, and 3 might be printed as:

123
132
213
231
312
321

The following subroutine will cause the digits you recall from R_1 , R_2 , and R_3 to be printed as a permutation in the order you have recalled them. Use this subroutine and the flowchart on the following page to help you create and load the program.

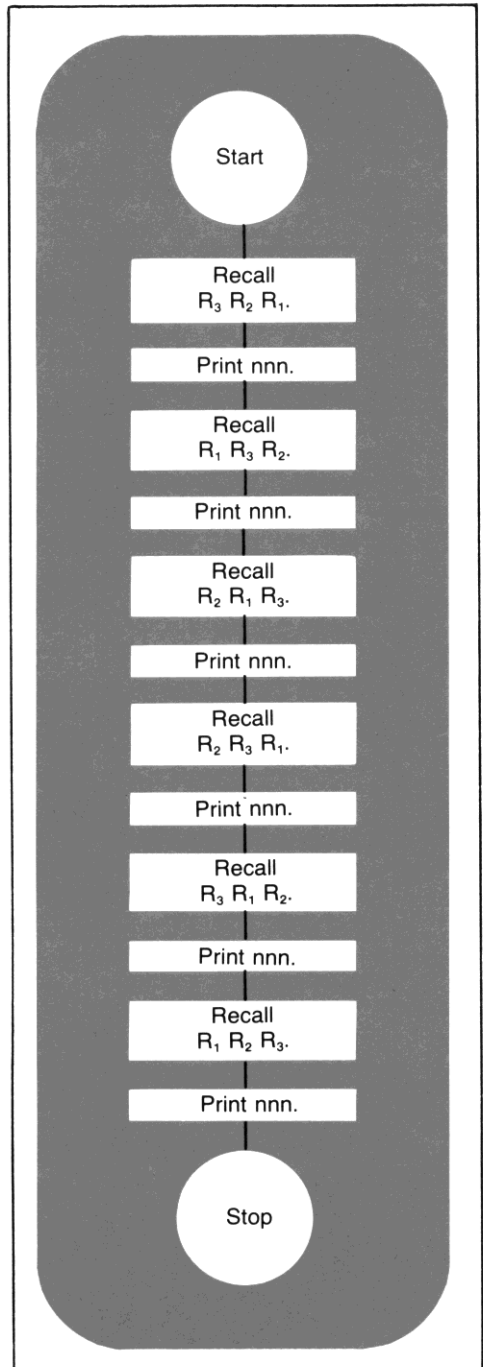
```
A-001 LBL 1
A-002 1
A-003 0
A-004 0
A-005 X
A-006 X↔Y
A-007 1
A-008 0
A-009 X
A-010 R↑
A-011 +
A-012 +
A-013 PRTX
A-014 RTN
```

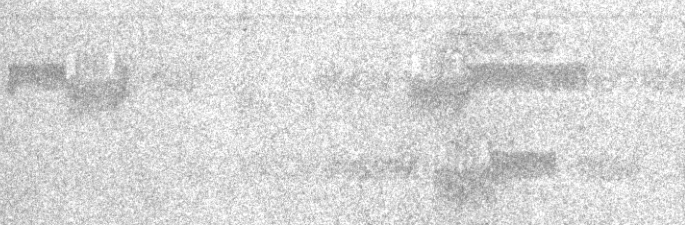
This subroutine prints numbers recalled into the Z-, Y-, and X-registers of the stack as nnn.

The program should recall the contents of storage registers R_1 , R_2 , and R_3 into the Z-, Y-, and X-registers of the stack and then use the "print nnn" subroutine to print them in the order that they are recalled.

When you have created and loaded the program, store the digits 5, 7, and 9 into storage registers R_1 , R_2 , and R_3 , respectively. Then run the program to show all the permutations of these three numbers.

Answer: 579
795
957
597
759
975





PRINT
CLEAR

STACK

FIX

STO

STO i

RCL i

DELETE

LAST x

B

C

D

PRINT CLEAR

x^2

yx

$1/x$

$=0$ $x \neq y$ $x \neq 0$ $x \leq y$ $x < 0$ $x > y$ $x > 0$

I

GSB

SST

ISZ DSZ

RTN

BST

9

GTO

R/S

LBL JUMP PAUSE

The I-Register and Indirect Control

The I-register is one of the most powerful programming tools available to you on your HP-95C. In a preceding section, Storing and Recalling Numbers, you learned about the use of the I-register as a simple storage register, similar to registers R_0 through R_9 and R_{10} through R_{15} . And of course, you can always use the I-register this way, as another storage register, whether you are using it as an instruction in a program or operating manually from the keyboard.

But the I-register is much more powerful than a mere storage register. Using the instructions **ISZ** and **DSZ**, you can increment (add 1 to) or decrement (subtract 1 from) the current value in the I-register, a feature that you will find extremely useful in controlling loops. And the **STO I** and **RCL I** instructions allow you to use the current value in the register as an address, selecting the addressed storage register in the calculator for storing or recalling of numbers.

Storing a Number in I

To store a number in the I-register, you can use the **STO I** operation. For example, to store the number 7 in the I-register:

Ensure that the PRGM-RUN switch **PRGM**  **RUN** is set to RUN.

Press	Display
-------	---------

7 STO I	7.00
----------------	------

To recall a number from the I-register into the displayed X-register, you do not have to use the **RCL** operation—you merely press **I**.

Press	Display
-------	---------

CLX	0.00
------------	------

I	7.00	The number stored in I is recalled.
----------	------	--


Incrementing and Decrementing the I-Register

Another way of altering the contents of the I-register, and one that is most useful during a program, is by means of the **ISZ** (*increment I, skip if zero*) and **DSZ** (*decrement I, skip if zero*) instructions. These instructions either add the number 1 to (increment) or subtract the number 1 from (decrement) the I-register each time they are executed. In a running program, if the number in the I-register has become zero, program execution *skips* the next step after the **ISZ** or **DSZ** instruction and continues execution (just like a false conditional instruction).

The **I** **ISZ** and **I** **DSZ** instructions always increment or decrement first; *then* the test for zero is made. For test purposes, numbers between but not including -1 and $+1$ are the same as zero.

Example: Here is a program that illustrates how **f ISZ** works. It contains a loop that pauses to display the current value in the I-register, then uses the **f ISZ** instruction to increment that value. The program will continue to run, continually adding one to and displaying the contents of the I-register, until you press **R/S** (or any key) from the keyboard.

To key in the program:

Slide the PRGM-RUN switch **PRGM**  **RUN** to **PRGM**.

Press	Display	
9 CLEAR A	A-000	<i>lbl A</i> Clears program A .
I	A-001	52 Recalls I-register contents.
f PAUSE	A-002	61 64 Pauses to display contents.
f ISZ	A-003	61 52 Adds 1 to I-register.
GTO A	A-004	63 A If contents of I-register are not zero, execution trans-
		fers back to top of program A .
1	A-005	1 If contents of I-register are zero, 1 is placed in
		I-register.
STO I	A-006	35 52
GTO A	A-007	63 A

Now run the program beginning with a value of 0 in the I-register. Stop the program after five iterations or so by pressing **R/S**.

Slide the PRGM-RUN switch **PRGM**  **RUN** to **RUN**.

Press	Display	
0 STO I	0.00	Zero stored in
		I-register.
A	0.00	
	1.00	
	2.00	
	3.00	
	4.00	
R/S	5.00	

Although the **ISZ** and **DSZ** instructions increment and decrement the I-register by 1, the value of the I-register need not be a whole number. For example:


Press	Display
5.28 CHS	-5.28
STO I	-5.28
A	-5.28
	-4.28
	-3.28
	-2.28
	-1.28
R/S	1.00

In practice, you will find that you will usually use **ISZ** and **DSZ** with numbers that are integers, since these instructions are most useful as counters—that is, to control the number of iterations of a loop—and to select storage registers. (More about using the I-register as a selection register later.)

The **DSZ** (*decrement I, skip if zero*) instruction operates in the same manner as the increment instruction, except that it subtracts, rather than adds, one each time it is used. When a running program executes a **9 DSZ** instruction, for example, it subtracts 1 from the contents of the I-register, then test to see if the I-register is 0. (A number between +1 and -1 tests as zero.) If the number in the I-register is greater than zero, execution continues with the next step of program memory. If the number in the I-register is zero, the calculator skips one step of program memory before resuming execution.

Example: The island of Manhattan was sold in the year 1624 for \$24.00. The program below shows how the amount would have grown each year if the original amount had been placed in a bank account drawing 5% interest compounded annually. The number of years for which you want to see the amount is stored in the I-register, then the **DSZ** instruction is used to keep track of the number of iterations through the loop.

To load the program:

Slide the PRGM-RUN switch **PRGM**  **RUN** to **PRGM**.

Press

Display

9 CLEAR B	<i>b-000</i>	<i>lbl b</i>	Program B cleared.
STO I	<i>b-001</i>	35 52	
1	<i>b-002</i>	1	
6	<i>b-003</i>	6	
2	<i>b-004</i>	2	
4	<i>b-005</i>	4	
STO 1	<i>b-006</i>	35 1	Initialization routine.
2	<i>b-007</i>	2	
4	<i>b-008</i>	4	
STO 2	<i>b-009</i>	35 2	
1 LBL 5	<i>b-010</i>	61 63 5	
RCL 2	<i>b-011</i>	45 2	
5	<i>b-012</i>	5	
%	<i>b-013</i>	51	
STO + 2	<i>b-014</i>	35 59 2	Counting loop, controlled by I-register and DSZ .
1	<i>b-015</i>	1	
STO + 1	<i>b-016</i>	35 59 1	
9 DSZ	<i>b-017</i>	62 52	
GTO 5	<i>b-018</i>	63 5	



Press	Display	
I SPACE	<i>b-019</i>	61 14 When value in I becomes zero, execution skips to here and year and amount are printed.
RCL 1	<i>b-020</i>	45 1
I FIX 0	<i>b-021</i>	61 21 0
PRINT X	<i>b-022</i>	14
RCL 2	<i>b-023</i>	45 2
I FIX 2	<i>b-024</i>	61 21 2
PRINT X	<i>b-025</i>	14

To run the program, key in the number of years for which you want to see the amount and press **B**.

For example, to run the program to find the amount of the account after 5 years; after 15 years:

Slide the PRGM-RUN switch **PRGM**  **RUN** to **RUN**.

Press	Display	
5	5.	
B	30.63	After five years, in 1629, the account would have been worth \$30.63.
		1629. *** 30.63 ***
15	15.	
B	49.89	After 15 years, in 1639, the account would have been worth \$49.89.
		1639. *** 49.89 ***

How it works: When you key in the number of years and press **B**, the number of years is stored in the I-register by the **STO I** instruction, the beginning year (1624) is stored in storage register R_1 , and the beginning amount (\$24.00) is stored in register R_2 .

After this initialization procedure, calculation begins. Each time through the loop, 5% of the amount is computed and added to the amount in R_2 , and one (1) year is added to the year in R_1 . The **DSZ** instruction subtracts one from the I-register; if the value in I is not then zero, execution is transferred back to **LBL 5** and the loop is executed again.

The loop continues to be executed until the value in the I-register becomes zero. Then execution skips to the **I SPACE** instruction in program memory step B-019. Execution continues sequentially downward from step B-019, recalling the current year from R_1 and formatting and printing it, then recalling the current amount from R_2 and formatting and printing that underneath the year.

To see what the amount in the account will be in 1977, you can key in the number of years from 1624 to 1977 (the number is 353) and run the program. (This will take 4-5 minutes to run, plenty of time to go get a cup of coffee.)

Indirect Store and Recall

You have seen how the value in the I-register can be altered using the **STO I**, **ISZ**, and **DSZ** operations. The value in the I-register can, in turn, be used to control the storage and recall of numbers with the instructions **STO I** and **RCL I**. **STO I** and **RCL I** use the current number stored in the I-register as an *address*.

When you press **I** **STO I**, the value that is in the display is stored in the storage register addressed by the number in the I-register. Similarly, **I** **RCL I** recalls the contents of the storage register addressed by the current number in the I-register.

When using **STO I** or **RCL I**, the I-register can contain positive or negative numbers from 0 through 15. The numbers 0 through 9 address storage registers R_0 through R_9 , while numbers 10 through 15 address registers R_{-0} through R_{-5} . The diagram below should illustrate these addresses more clearly:

When the number
in the I-register
is:

15
14
13
12
11
10

9
8
7
6
5
4
3
2
1
0


The storage register
addressed by **STO I**
or **RCL I** is:

R_{-5}
 R_{-4}
 R_{-3}
 R_{-2}
 R_{-1}
 R_{-0}

R_9
 R_8
 R_7
 R_6
 R_5
 R_4
 R_3
 R_2
 R_1
 R_0

As with **ISZ** and **DSZ**, both **STO I** and **RCL I** look at the absolute value of the integer portion of the current number stored in the I-register when using it for an address. If **STO I** or **RCL I** is executed (whether from the keyboard or in a running program) when the absolute value of the integer portion of the number in I is not in the range of 0-15, the calculator will immediately register an error by a display of **SEE 7**

By using the calculator manually, you can easily see how **STO I** and **RCL I** are used in conjunction with the I-register to address the different storage registers:

First, ensure that the PRGM-RUN switch **PRGM**  **RUN** is set to RUN.

Press	Display	
CL X	0.00	
9 CLEAR REG	0.00	Clears all storage registers (except the I-register) to zero.
5 STO I	5.00	Stores the number 5 in the I-register.
1.23 f STO I	1.23	Stores the number 1.23 in the storage register addressed by the number in I—that is, storage register R ₅ .
14 STO I	14.00	The number 14 is stored in the I-register.
85083 f STO I	85083.00	This number is stored in the storage register (R ₄) addressed by the current number in I.

As you know, to recall the actual *contents* of the I-register, you can simply press **I**:

Press	Display	
I	14.00	Current contents of the I-register are recalled.

To recall the contents of the storage register addressed by the number in I, press **f** **RCL I**:

Press	Display	
RCL 5	1.23	Contents of storage register R ₅ recalled.
f RCL I	85083.00	Since the I-register still contains the number 14, this operation recalls the contents of the storage register (R ₄) addressed by the number 14.


By changing the number in the I-register, you change the address specified by **STO I** or **RCL I**. For example:

Press	Display
5 I	5.00
I RCL I	1.23

Naturally, the most effective use of the I-register as an address for **STO I** and **RCL I** is in a program.

Example: The following program uses a loop to place the number representing its address in storage registers R_0 through R_9 and R_{10} through R_{15} . During each iteration through the loop, program execution pauses to show the current value of I. When I reaches zero, execution finally is transferred out of the loop by the **DSZ** instruction and the program stops.

To key in the program:

Slide the PRGM-RUN switch **PRGM**  **RUN** to **PRGM**.

Press	Display	
9 CLEAR C	C-000	<i>lbl C</i> Clears program C .
9 CLEAR REG	C-001	62 12
1	C-002	1
5	C-003	5
STO I	C-004	35 52
I LBL 1	C-005	61 63 1
I	C-006	52
I STO I	C-007	61 35
I PAUSE	C-008	61 64
9 DSZ	C-009	62 52
GTO 1	C-010	63 1
I PRINT REG	C-011	61 12

Clears program **C**.

Program initialized.

Current value in I stored in storage register addressed by **STO I**.

Pause to display current value of I.

Subtract 1 from value in I-register.

If $I \neq 0$, execute loop again.

Otherwise, print the contents of all the storage registers.

When the program is run, it begins by clearing the storage registers and placing 15 in the I-register. Then execution begins, recalling the current value in the I-register and storing that number in the corresponding address—for example, when the I-register contains the number 12, that number is recalled and stored in the storage register (R_{12}) that is addressed by the number 12. Each time through the loop, the number in the I-register is decremented, and the result is used both as data and as an address by the **STO I** instruction. When the number in the I-register reaches zero, execution transfers out of the loop and the contents of all storage registers are printed.

To run the program:

Slide the PRGM-RUN switch PRGM  RUN to RUN.

Press **Display**

C 1.00

0.00 ← 0
1.00 ← 1
2.00 ← 2
3.00 ← 3
4.00 ← 4
5.00 ← 5
6.00 ← 6
7.00 ← 7
8.00 ← 8
9.00 ← 9
10.00 ← 0
11.00 ← 1
12.00 ← 2
13.00 ← 3
14.00 ← 4
15.00 ← 5

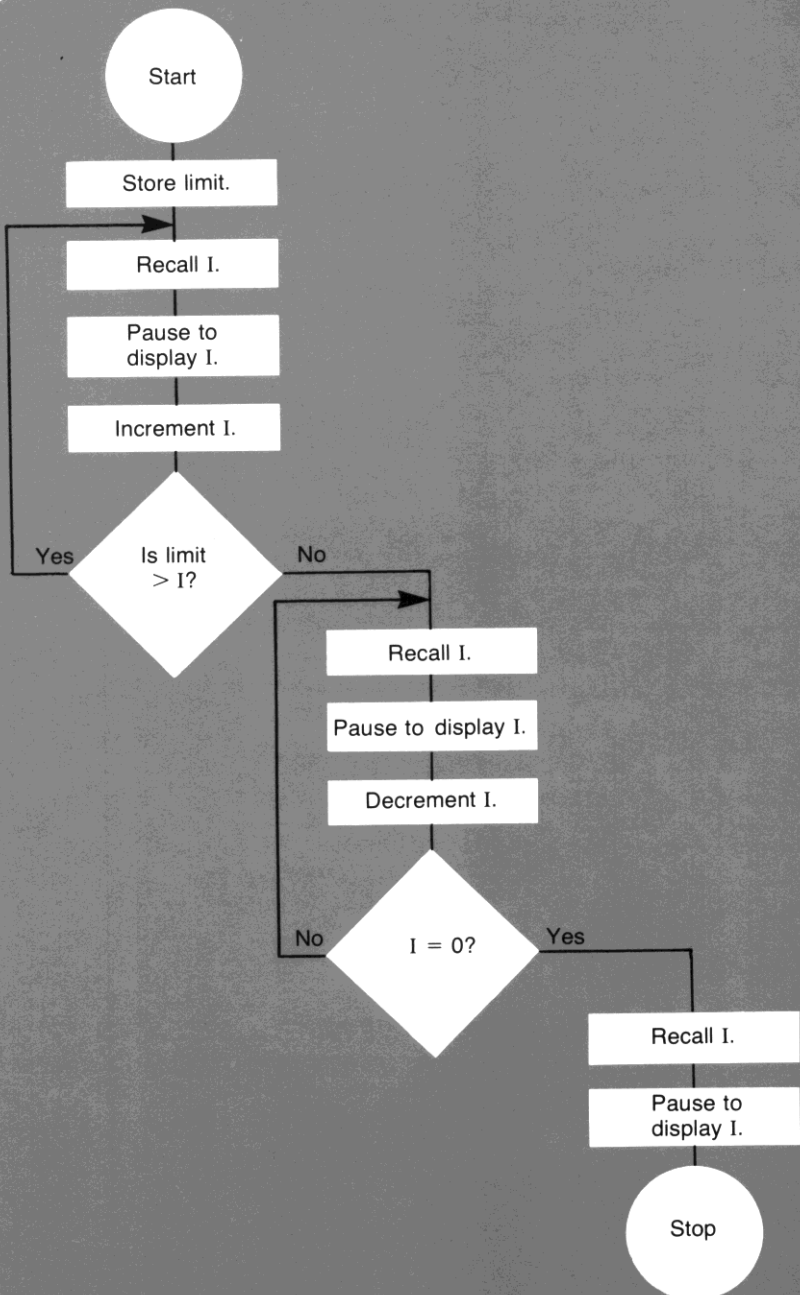
The contents of the I-register have been decremented to zero.

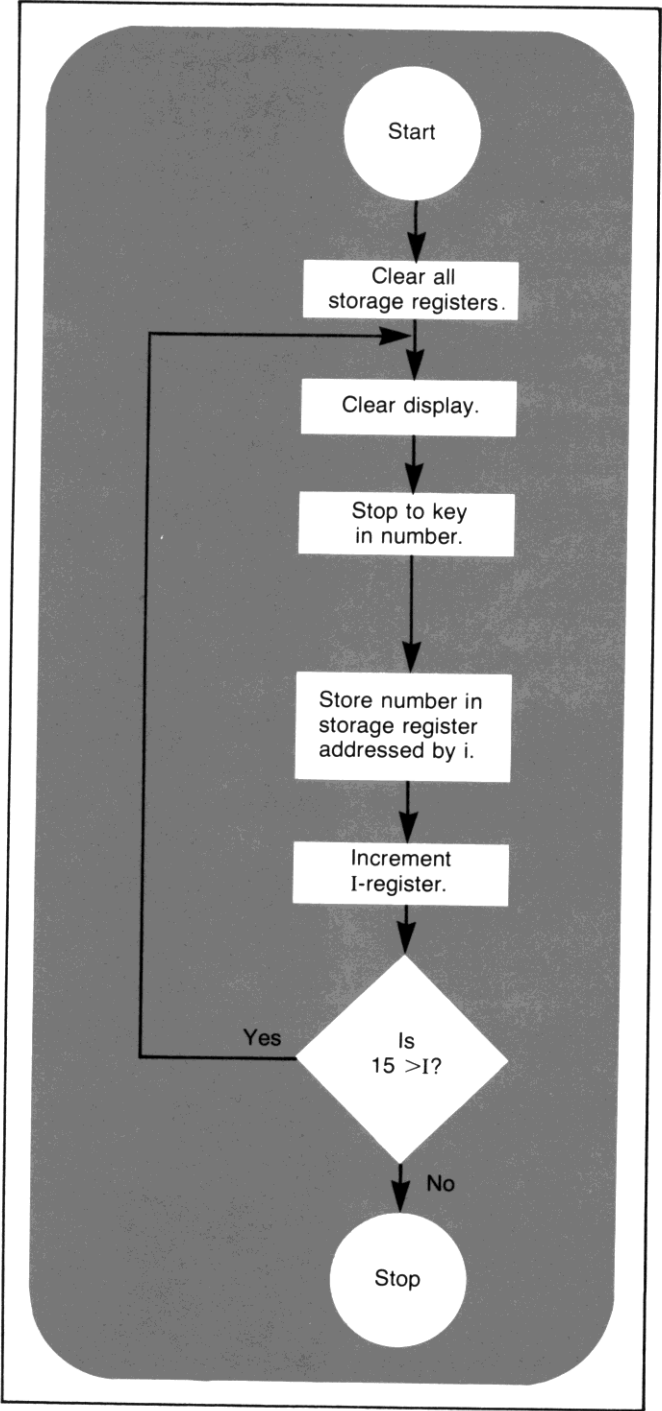
The **STO I** and **RCL I** instructions, in conjunction with the I-register and **ISZ** and **DSZ**, are extremely useful in programs requiring sequential storage or recall.

Problems

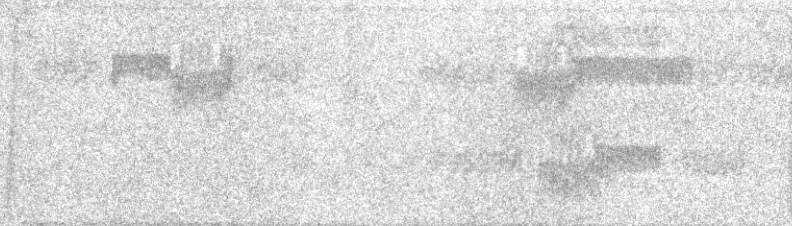
- When you press **A**, the program here stores in primary storage register R_9 a number that you have keyed in, then decrements the value in R_9 using storage register arithmetic. Each time through the loop, the program pauses to show the current value in R_9 . When the value in R_9 reaches zero, the program stops. Write, load, and run a program that uses the I-register and **9 DSZ** (instead of R_9 and **9 X≠0**) to give the same results.


```
A-000 LBLA
A-001 S 9
A-002 LBL1
A-003 PSE
A-004 1
A-005 S-9
A-006 R 9
A-007 X≠0
A-008 GT01
```
- Write and load a program using **ISZ** to illustrate how an initial deposit of \$1000 would grow year-by-year at a yearly compound interest rate of 5.5%. The program should print the current year and subsequent years, together with the value of the account for each year. The program should contain an infinite loop that you can stop by pressing **R/S** from the keyboard whenever you wish. Run the program to print the year and amount for at least 5 years.
- Write, load, and run a program that will count from zero *up* to a limit using the **ISZ** instruction, and then count back down to zero using the **DSZ** instruction. The program can contain two loops, and it can contain a conditional instruction besides the **ISZ** and **DSZ** instructions. Use the flowchart on the opposite page to help you.





4. a. Create and load a program using **DSZ**, **STO I**, and **R/S** that permits you to key in a series of values during successive stops. The values should be stored in storage registers R_0 through R_9 and R_{10} through R_{15} in the order you key them in. Use the flowchart on page 184 to help you.
- b. Now create and load a program immediately after the first one that will recall and print the contents of each storage register in reverse order (that is, print R_{15} first, then R_{14} , etc.). The program should stop running after it has printed the contents of R_0 .
- Run the program you loaded for problem 4a, keying in a series of 15 different values. Then run the program you loaded for 4b. All 15 values should be printed, but the last one you keyed in should be the first printed, etc.



PRINT
CLEAR

A

B

C

D

PRINT CLEAR

\sqrt{x}

x^2

y^x

$1/x$

$x=y$ $x=0$

$x \neq y$ $x \neq 0$

$x < y$ $x < 0$

$x > y$ $x > 0$

%

I

GSB

SST

%Σ Δ%

ISZ DSZ

RTN

BST

f

g

GTO

R/S

LBL JUMP

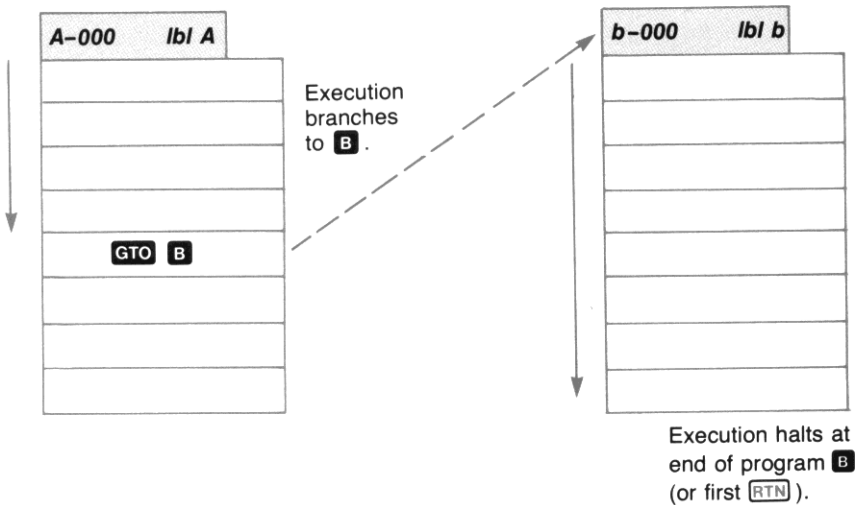
PAUSE

Transferring Execution Between Programs

As you know, you can use **GTO** or **GSB** to transfer execution from one point to another within a program. However, there may also be occasions when you want a running program to transfer execution *out* of that program to another program. This section shows how to use the instructions **GTO**, **GSB**, and **JUMP** to transfer execution from one program to another.

Branching to Another Program

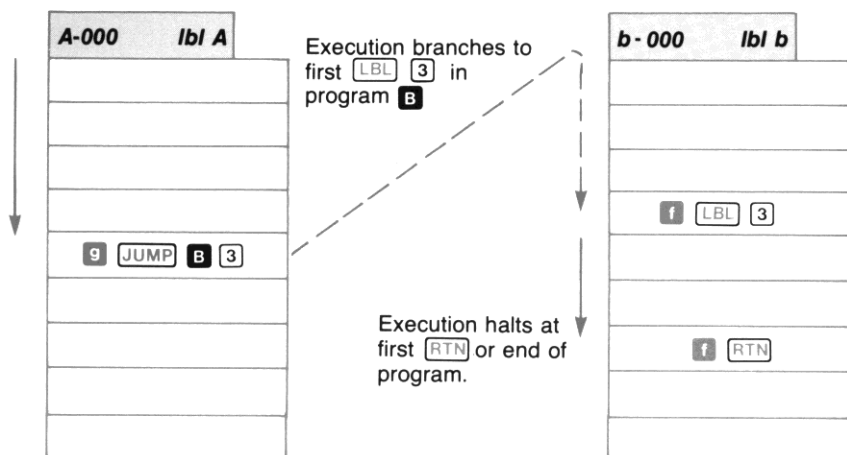
To branch execution from one program to another, use the instruction **GTO** followed by the program marker (**A**, **B**, **C**, or **D**) of the program to which execution is to be branched.



In the illustration shown above, when you pressed **A**, execution would continue through program **A** until the **GTO B** instruction was executed, whereupon execution would transfer to the top of program **B** and continue. When a **RTN** (or the end of program **B**) was executed, execution would then stop.

Branching to a Label Within Another Program

As you know, when a running program executes a **GTO** **3** instruction, the calculator searches in that program for the next **LBL** **3** before resuming execution. However, using the instruction **9 JUMP** followed by the program designator (**A**, **B**, **C**, or **D**), followed in turn by the desired label (**0** through **9**) of the selected program, you can transfer execution from one program to a routine within another program.

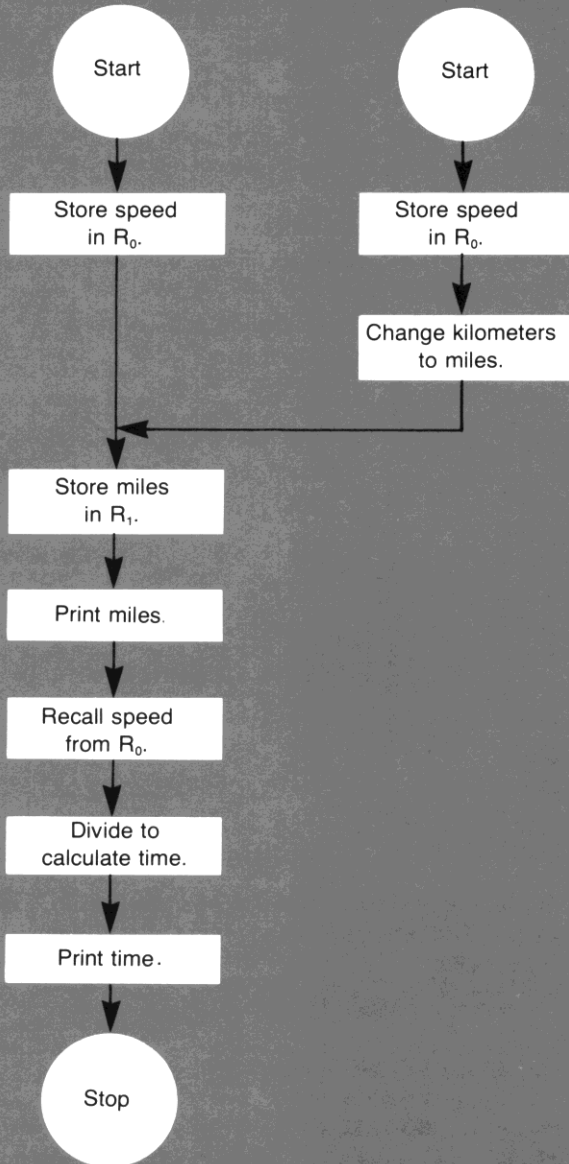


In the illustration shown above, when you pressed **A**, execution would begin, continuing until the **g JUMP B 3** instruction caused the calculator to jump to program **B** and search for the first **f LBL 3**. Execution would resume with the **LBL 3** and continue until a **RTN** or the end of program **B** were executed, whereupon execution would stop there.

Notice that you use the **JUMP** instruction to transfer execution from one program to a routine within *another* program. If you are keying in instructions in program **A** and try to key in, say **g JUMP A 2**, the instruction that is actually keyed in will be **GTO 2** (keycode 63 2), since that is actually the instruction you are trying to load.

Example: Having had quite enough of the frigid winters of Alaska, rugged frontiersman Natty Bumpo is driving his 1959 Edsel the length of North and Central America to the relatively sunny climate of Brazil. In the course of his journey, Bumpo wishes to be able to read distance markers and calculate how long it will take him to reach the next town. He can read his speed in miles per hour from the Edsel's speedometer, but he knows that some of the distance markers will be in miles and some in kilometers.

The flowchart on page 189 shows how Bumpo uses his HP-95C to calculate the time it will take him to drive between points. When he keys in the number of miles, presses **ENTER**, then keys in the speed in miles per hour and presses **A**, the HP-95C calculates the time in hours and prints the number of miles and the time in hours. When he keys in the number of kilometers, presses **ENTER**, then keys in the speed in miles per hour and presses **B**, the program changes kilometers to miles, then transfers to program **A** to calculate the time in hours and print the number of miles and the time.

**Miles
Program A****Kilometers
Program B**

To key in the first program:

Slide the PRGM-RUN switch PRGM  RUN to PRGM.

Press	Display	
9 CLEAR A	A-000	lbl A Clears previous instructions from A .
STO 0	A-001 35 0	Stores speed in R ₀ .
R+	A-002 12	
f LBL 9	A-003 61 63 9	
STO 1	A-004 35 1	Stores number of miles in R ₁ .
f FIX 0	A-005 61 21 0	
PRINT X	A-006 14	Prints number of miles.
RCL 0	A-007 45 0	
=	A-008 24	Calculates time in decimal hours.
f H→H:MM:SS	A-009 61 24	Changes time to <i>hours, minutes, seconds</i> .
f FIX 2	A-010 61 21 2	
PRINT X	A-011 14	Prints time as <i>h.mm</i> .

Now key in the second program:

Press	Display	
9 CLEAR B	b-000	lbl b Clears previous instructions from program B and sets calculator to top of B .
STO 0	b-001 35 0	Stores speed in R ₀ .
R+	b-002 12	
1	b-003 1	} Changes kilometers to miles.
.	b-004 66	
6	b-005 6	
0	b-006 0	
9	b-007 9	
3	b-008 3	
4	b-009 4	
4	b-010 4	
=	b-011 24	
9 JUMP A 9	b-012 62 63 A9	Transfers execution to LBL 9 in program A .

As he leaves Boise, Idaho, Bumpo sees a sign that says “Dallas, 1637 miles.” To run the program to find how long it will take the Edsel to reach Dallas if Bumpo keeps the speedometer set on 55 miles per hour:

Slide the PRGM-RUN switch PRGM  RUN to RUN.

Press	Display	
1637 ENTER	1637.00	Number of miles keyed in.
55	55.	Speed keyed in.
A	29.45	The time in 29 hours, 45 minutes. Both number of miles and time are printed.
	1637. ***	
	29.45 ***	

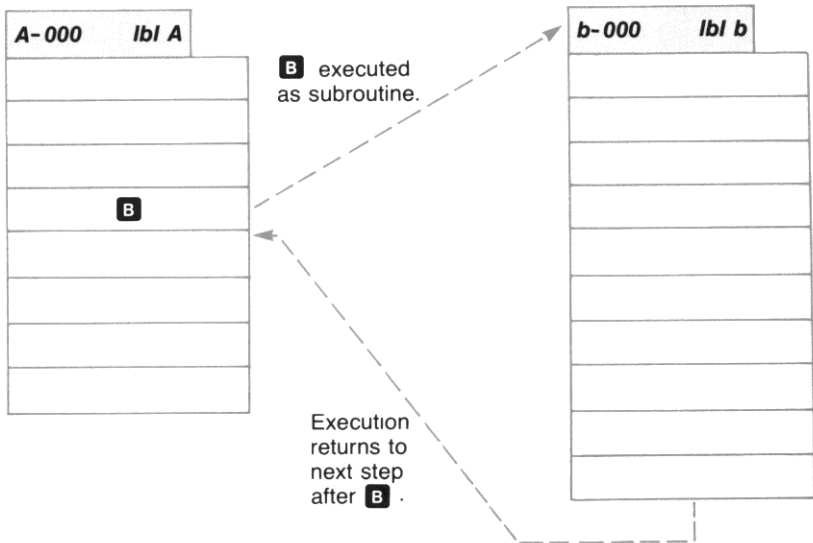
As he rockets through the Sonoran desert at 85 miles per hour, Bumpo glimpses a sign giving the distance to Mexico City as 1900 kilometers. What is the distance in miles to Mexico City and how long will it take the Edsel at this speed?

Press	Display	
1900 ENTER ▶	1900.00	Number of kilometers keyed in.
		1181. ***
85	85.	Speed keyed in.
		13.53 ***
B	13.53	The time is 13 hours, 53 minutes. Both number of miles and time are printed.

The above example illustrates how the HP-95C can easily branch from one program to another.

Using a Program as a Subroutine

With your HP-95C, you can not only branch from one program to another, you can also call a second program as a *subroutine*. Just use one of the program designators (**A**, **B**, **C**, or **D**) in the first program to call up the designated program as a subroutine.



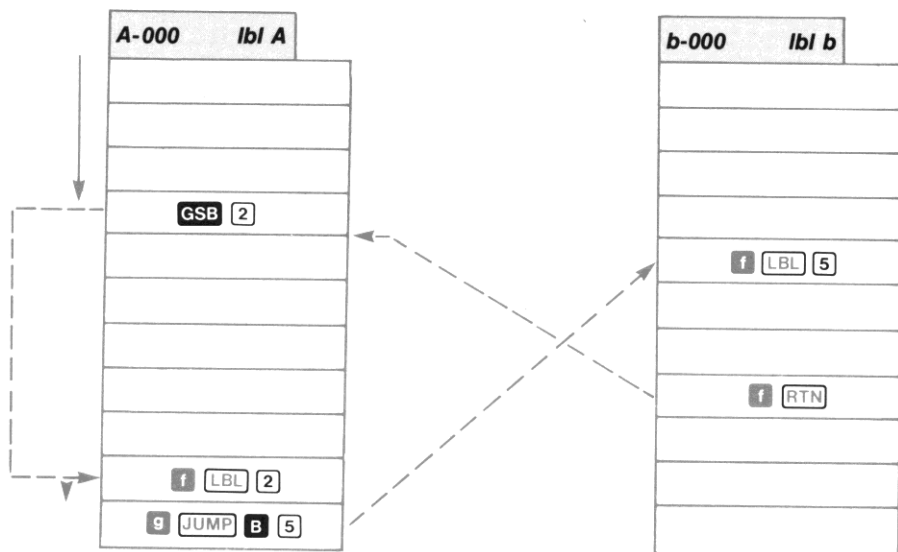
In the illustration shown here, when program **A** executed the **B** instruction, execution would transfer to the top of program **B** and resume. When the end of program **B** (or a **RTN**) was executed, execution would then transfer back to program **A**, resuming with instruction in the next step of program memory after the **B** instruction.

Notice that execution is exactly as though program **A** had executed the instruction **GSB B** (and, in fact, if you attempt to key in the instruction **GSB B** into program **A**, **C**, or **D**, you will see only the program memory step number and the symbol for program **B**, like this: **A-004 b**).

Naturally, the three-level limit for subroutines remains the same, whether a routine or an entire program is called up as a subroutine. Remember that the end of a program acts exactly as a **RTN**.

Calling a Subroutine From Another Program

By using the instructions **GSB**, **LBL**, and **JUMP**, you can use the HP-95C to call up a routine in one program as a subroutine from another program.



In the illustration shown here, if you pressed **A** from the keyboard in RUN mode, the calculator would execute instructions until the **GSB 2** instruction was executed and execution resumed with the next **LBL 2** in that program. The first instruction executed after the **LBL 2** would be **JUMP B 5**, causing the calculator to jump to **LBL 5** in program **B** and resume execution there. Since execution of the routine is as a subroutine, the first **RTN** encountered would cause execution to return to the next step *after* the **GSB** instruction—even though the next step is in program **A**.

Problem

The area of a sphere can be calculated according to the equation $A = 4\pi r^2$, where r is the radius. The formula for finding the volume of a sphere is $V = \frac{4\pi r^3}{3}$. This may also be expressed as $V = \frac{r \times A}{3}$.

Write and load a program to calculate the area A of a sphere given its radius r . Load this program so that it is selected by user-definable key **A**.

Then write and load a second program to calculate the volume V of a sphere, using the equation $V = \frac{r \times A}{3}$. Load this program so that it is selected by user-definable key **B**, and include the instruction **GSB A** to use program **A** as a subroutine calculating area.

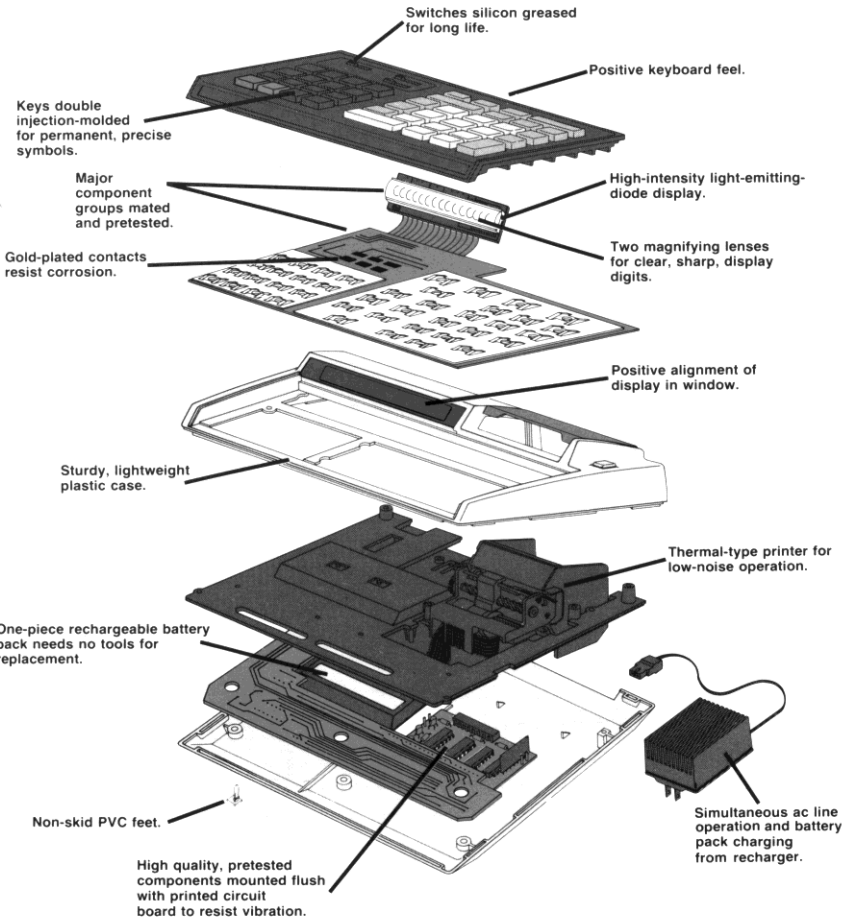
Run the two programs to find the area and volume of the planet Earth, a sphere with a radius of about 3963 miles. Of the Earth's moon, a sphere with a radius of about 1080 miles.

Answers: Earth area = 197359487.5 square miles
 Earth volume = 2.6071188×10^{11} cubic miles
 Moon area = 14657414.69 square miles
 Moon volume = 5276669290 cubic miles.

Accessories, Service, and Maintenance

Your Hewlett-Packard Calculator

Your HP-95C is another example of the award-winning design, superior quality, and attention to detail in engineering and construction that have marked Hewlett-Packard electronic instruments for more than 30 years. Each Hewlett-Packard calculator is precision crafted by people who are dedicated to giving you the best possible product at any price.



After construction, every calculator is thoroughly inspected for electrical or mechanical flaws, and each function is checked for proper operation.

When you purchase a Hewlett-Packard calculator, you deal with a company that stands behind its products. Besides an instrument of unmatched professional quality, you have at your disposal many extras, including a host of accessories to make your calculator more usable and service that is available worldwide.

Standard Accessories

Your HP-95C comes complete with the following standard accessories:

Accessory	HP Number
Battery Pack (installed in calculator before packaging)	82033A
<i>HP-95C Owner's Handbook and Programming Guide</i>	00095-90001
<i>HP-95C Applications Book</i>	00095-90003
AC Adapter/Recharger (one of the following)	
U.S. (90-127 Vac, 50-60 Hz)	82040A
European (200-254 Vac, 50-60 Hz)	82031A
Australian (200-254 Vac, 50-60 Hz)	82039A
U.K. (Desktop, 200-254 Vac, 50-60 Hz)	82032A
Two Rolls of Paper (available in six-roll packs)	
Carrying Case	82035A

You can purchase additional standard accessories from your nearest dealer or by mail from Hewlett-Packard. See Optional Accessories below for information on how to order.

Optional Accessories

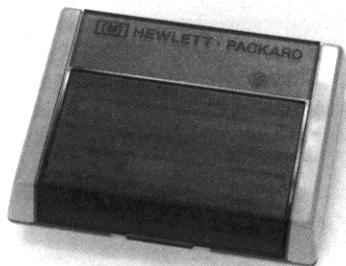
Security Cable 82044A

A tough six-foot long steel cable that prevents unauthorized borrowing or pilferage of your calculator by locking it to a desk or work surface. The cable is plastic-covered to eliminate scarring of furniture, and you have full access to all features of your HP-95C at all times. Comes complete with lock.

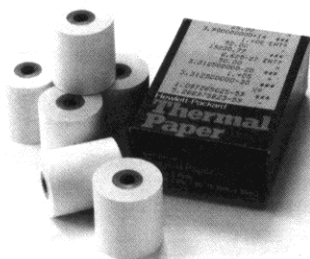


Reserve Power Pack**82037A**

The reserve power pack attached to the calculator's ac adapter/recharger to keep an extra battery pack freshly charged and ready for use. Comes complete with extra battery pack.

**Paper Rolls****82045A**

Each pack gives you six rolls of special Hewlett-Packard thermal paper for your HP-95C printer.



To order additional standard or optional accessories for your HP-95C see your nearest dealer or fill out an Accessory Order Form and return it with check or money order to:

HEWLETT-PACKARD
Corvallis Division
P.O. Box 999
Corvallis, Oregon 97330

If you are outside the U.S., please contact the Hewlett-Packard Sales Office nearest you. Availability of all accessories, standard or optional, is subject to change without notice.

AC Line Operation

Your calculator contains a rechargeable battery pack that is made up of nickel-cadmium batteries. When you receive your calculator, the battery pack inside may be discharged, but you can operate the calculator immediately by using the ac adapter/recharger. Even though you are using the ac adapter/recharger, the batteries must remain in the calculator whenever the calculator is used.

Note: Attempting to operate the HP-95C from the ac line with the battery pack removed may result in wrong or improper displays.

The procedure for using the ac adapter/recharger is as follows:

1. You need not turn the HP-95C OFF.
2. Insert the female ac adapter/recharger plug into the rear connector of the HP-95C.
3. Insert the power plug into a live ac power outlet.

CAUTION

Always use an HP recharger like the one supplied with your calculator when operating from an ac outlet. Failure to do so may result in damage to your calculator.

Battery Charging

The rechargeable batteries in the battery pack are being charged when you are operating the calculator from the ac adapter/recharger. With the batteries in the calculator and the recharger connected, the batteries will charge with the calculator OFF or ON. Normal charging times from fully discharged battery pack to full charge are:

Calculator OFF: 7–10 hours

Calculator ON: 17 hours



Shorter charging periods will reduce the operating time you can expect from a single battery charge. Whether the calculator is OFF or ON, the HP-95C battery pack is never in danger of becoming overcharged.

Note: It is normal for the ac adapter/recharger to be warm to the touch when it is plugged into an ac outlet.

Battery Operation

To operate the HP-95C from battery power alone, simply disconnect the female recharger plug from the rear of the calculator. (Even when not connected to the calculator, the ac adapter/recharger may be left plugged into the ac outlet.)

Using the HP-95C on battery power gives the calculator full portability, allowing you to carry it nearly anywhere. A fully charged battery pack provides approximately 3 to 7 hours of continuous operation. By turning the power OFF when the calculator is not in use, the charge on the HP-95C battery pack should easily last throughout a normal working day.

The printer is the most power-consuming part of your HP-95C, and you can maximize battery operating time by leaving the calculator in **MANUAL**  **TRACE**  **NORM** printing mode when printing is not necessary.

Using Continuous Memory

When you turn OFF your HP-95C, the following information is retained:

- All programs that are loaded into the calculator.
- Contents of the 16 addressable storage registers and the I-register.
- Trigonometric mode status (DEG, RAD, or GRAD).
- Display status (FIX, SCI, or ENG, and number of displayed digits).

Regardless of where you stopped in a program, the HP-95C returns to step A-000 (top of program **A**) when you turn it ON again.

Numbers in the stack are not saved when you turn the calculator OFF; however you can use the addressable storage registers to retain data in the calculator.

Continuous Memory requires that the *batteries be kept in the calculator*. If the low power indicator appears in the display, turn your HP-95C OFF immediately, and connect it to an ac outlet or insert a new battery pack. If you allow the battery to discharge completely, the information in memory will be lost.

If you drop or traumatize your HP-95C, or if power to the Continuous Memory is interrupted whether the calculator is OFF or ON, the contents of program memory and the data storage registers may be lost. If this occurs, when the calculator is then turned ON, the display will show SEE 1. To restore the display, ensure that the battery is charged, or connect the ac adapter/recharger, and press any key.

Battery Pack Replacement

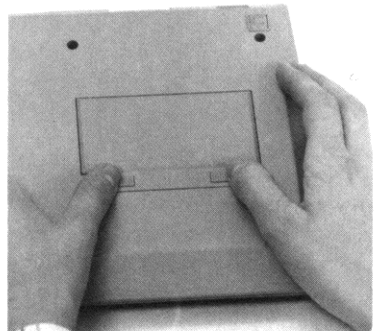
If it becomes necessary to replace the battery pack, use only another Hewlett-Packard battery pack like the one shipped with your calculator.

CAUTION

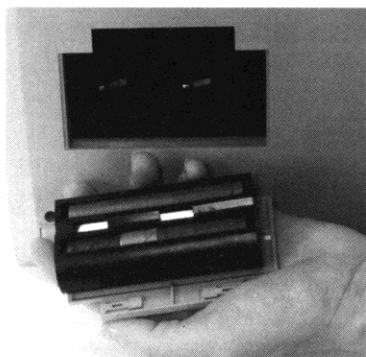
Use of any batteries other than the Hewlett-Packard battery pack may result in damage to your calculator.

To retain the memory while changing batteries, first turn your calculator OFF, then follow the instructions below. A small capacitor provides temporary standby power to the memory for approximately 10 seconds to 2 minutes while you change batteries. To change the battery pack:

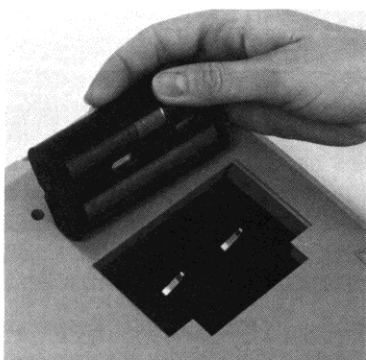
1. Turn the ON-OFF switch to OFF and disconnect the ac adapter/recharger from the calculator.
2. Slide the two battery door latches inward.



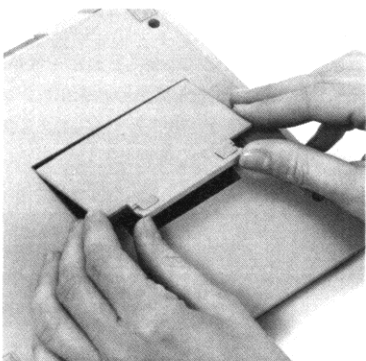
3. Let the battery door and battery pack fall into the palm of your hand.
4. If the battery connector springs have been flattened inward, bend them slightly outward again.



5. Insert the new battery pack so that its contacts face the calculator and line up with the connector springs.



6. Insert the end of the battery door that is opposite the latches behind the retaining groove and close the door.
7. Secure the battery door by pressing it gently while sliding the two battery door latches outward.



Battery Care

When not being used, the batteries in your HP-95C have a self-discharge rate of approximately 1% of available charge per day. After 30 days, a battery pack could have only 50 to 75% of its charge remaining, and the calculator might not even turn on. If a calculator fails to turn on, you should substitute a charged battery pack, if available, for the one in the calculator. The discharged battery pack should be charged for at least 14 hours.

If a battery pack will not hold a charge and seems to discharge very quickly in use, it may be defective. The battery pack is warranted for one year, and if the warranty is in effect, return the defective pack to Hewlett-Packard according to the shipping instructions. (If you are in doubt about the cause of the problem, return the complete HP-95C along with its battery pack and adapter/recharger.) If the battery pack is out of warranty, see your nearest dealer or use the Accessory Order Form provided with your HP-95C to order a replacement.

WARNING

Do not attempt to incinerate or mutilate your HP-95C battery pack—the pack may burst or release toxic materials.

Do not connect together or otherwise short circuit the battery terminals—the pack may melt or cause serious burns.

To maximize the life you get from battery pack, keep printing to a minimum and display only the fewest number of digits necessary during portable operation.

Your HP-95C Printer

The printing device in your HP-95C is a thermal printer that uses a moving print head to print upon a special heat-sensitive paper. When the print head is energized, it heats the paper beneath it. The heat causes a chemical change in the paper, which then changes color. The printer in your HP-95C prints answers quickly and quietly, and has been expressly designed to give you a permanent record of your computations in a portable scientific calculator.

Paper for Your HP-95C

Because the printer in your HP-95C is a thermal printer, it requires special heat-sensitive paper. You should use only the Hewlett-Packard thermal paper available in 80-foot rolls from your nearest HP distributor or sales office, or by mail from:

HEWLETT-PACKARD
Corvallis Division
P.O. Box 999
Corvallis, Oregon 97330

Because of the special heat-sensitive requirements of the paper, standard adding machine paper will *not* work in the HP-95C. Also, since different types of thermal paper vary in their sensitivities, the use of thermal paper other than that available from Hewlett-Packard may result in poor print quality or even in damage to your calculator.

CAUTION

Use only Hewlett-Packard paper in your HP-95C.

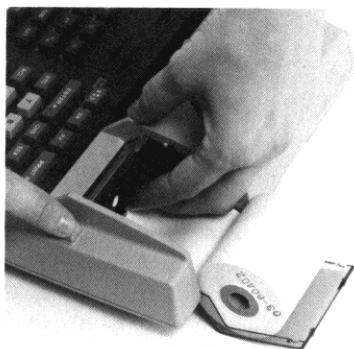
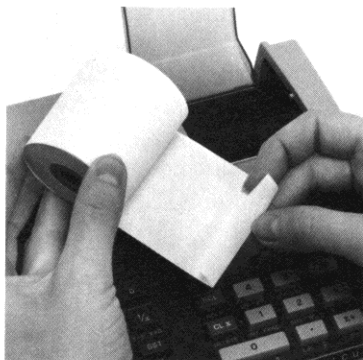
The heat-sensitive paper used in your HP-95C should be stored in a cool, dark place. Discoloration of paper may occur if it is exposed to direct sunlight for long periods of time, if storage temperatures rise above 50°C (122°F), or if the paper is exposed to excessive humidity or to acetone, ammonia, or other organic compounds. (Exposure to gasoline or oil fumes will not harm your HP-95C paper supply.)

Printed tapes from your HP-95C will last 30 days or more without fading under fluorescent light, but to ensure the permanence of your records, you should store printed tapes at room temperature in a dark place away from direct sunlight, heat, or fumes from organic compounds. (For added permanence, you can copy tapes with a suitable office copier.)

Replacing Paper

To replace the paper roll in your HP-95C proceed as follows:

1. Open the paper roll cover and remove the empty core from the paper well.
2. Before inserting the new roll of paper into the calculator, discard the first 2/3 turn to ensure that no glue, tape, or other foreign matter is on the paper.
3. Fold the leading edge of the paper and crease the fold with your fingernail.
4. Temporarily place the paper roll into the paper roll cover and insert the leading edge of paper into the slot near the bottom of the paper well.



5. Turn the calculator ON-OFF switch to ON and press the paper advance pushbutton several times until the leading edge of paper becomes visible beneath the clear plastic tear bar. You can remove the tear bar for accessibility, if desired.
6. Drop the roll of paper into the paper well and close the paper roll cover.



When there is no paper in the calculator, the paper advance pushbutton operates, but the printer does not.

Printer Maintenance

The printer in your HP-95C, like the rest of the calculator, is crafted for engineering excellence and is designed to give trouble-free operation with a minimum of maintenance. All moving parts in the printer mechanism contain self-lubricating compound, and no lubrication, cleaning, or servicing of the mechanism is ever required. You may want to occasionally remove the clear plastic tear bar and clean it with mild soap and water solution. (Do not use acetone or alcohol to clean the tear bar.)

You should *never* attempt to insert a tool, such as a screwdriver, pencil, or other hard object, into the printer or its mechanism. If the paper tape should become jammed and fail to feed properly, clear it by grasping the tape and pulling it forward or backward through the printer mechanism. (You can remove the plastic tear bar for accessibility.)

If the paper is feeding properly through the printer mechanism, but no printing appears on the tape, the paper roll is probably inserted backwards. (The paper is chemically treated, and will print on only one side.) Tear off the leading edge of paper, open the paper roll cover and grasp the paper roll, and pull it backward to remove the paper tape that is in the print mechanism. Reverse the paper roll and feed it back into the printing mechanism as described earlier under Replacing Paper.

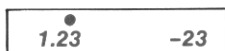
If, after reversing, there is still no printing on the tape when you press **PRINT** or other print functions, remove the paper roll and insert a roll of Hewlett-Packard thermal paper.

Note: Printer operation may be affected if the printer is in close proximity to a strong magnetic field. Normal operation can be restored by removing the calculator from the vicinity of the magnetic field. No permanent damage will result.

Service

Low Power

When you are operating from battery power, a bright red lamp inside the display will glow to warn you that the battery is close to discharge.



Low Power Indicator

You must then either connect the ac adapter/recharger to the calculator as described under AC Line Operation, or you must substitute a fully charged battery pack for the one in the calculator.

Blank Display

If the display blanks out, turn the HP-95C OFF, then ON. If a display of zeros does not appear in the display in RUN mode, check the following:

- 1. If the ac adapter/recharger is attached to the HP-95C, make sure it is plugged into an ac outlet.
- 2. Examine the battery pack to see if the contacts are dirty.
- 3. Substitute a fully charged battery pack, if available, for the one that was in the calculator.
- 4. If the display is still blank, try operating the HP-95C using the recharger (with the batteries in the calculator).
- 5. If, after step 4, the display is still blank, service is required. (Refer to Warranty paragraphs.)

Temperature Range

Temperature ranges from the calculator are:

Operating	0° to 45°C	32° to 113°F
Charging	15° to 40°C	59° to 104°F
Storage	-40° to + 55°C	-40° to +131°F

Warranty

Full One-Year Warranty

The HP-95C and its accessories are warranted against defects in materials and workmanship for one (1) year from the date of delivery. During the warranty period, Hewlett-Packard will repair or, at its option, replace at no charge components that prove to be defective, provided the calculator or accessory is returned, shipping prepaid, to Hewlett-Packard's Repair Center. (Refer to Shipping Instructions).

This warranty does not apply if the calculator or accessory has been damaged by accident or misuse, or as a result of service or modification by other than an authorized Hewlett-Packard Repair Center. No other expressed warranty is given by Hewlett-Packard. **HEWLETT-PACKARD SHALL NOT BE LIABLE FOR CONSEQUENTIAL DAMAGES.**

Some states do not allow the exclusion or limitation of incidental or consequential damages, so the above limitation or exclusion may not apply to you.

This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.

Out-of-Warranty

After the one-year warranty period, calculators will be repaired for a moderate charge. All repair work performed beyond the warranty period is warranted for a 90-day period.

Warranty Transfer

If you sell your calculator or give it as a gift, the warranty is transferable and remains in effect for the new owner until the original one-year expiration date. It is not necessary for the owner to notify Hewlett-Packard of the transfer.

Obligation to Make Changes

Products are sold on the basis of specifications applicable at the time of sale. Hewlett-Packard shall have no obligation to modify or update products once sold.

Repair Policy

Hewlett-Packard maintains Repair Centers in most major countries throughout the world. You may have your calculator repaired at a Hewlett-Packard Repair Center any time it needs service, whether the unit is under warranty or not.

The Hewlett-Packard United States Repair Center (for handheld and portable printing calculators) is located at Corvallis, Oregon.

The mailing address is: Hewlett-Packard Company
Corvallis Division, Service Dept.
P.O. Box 999
Corvallis, Oregon 97330

Repair Time

Hewlett-Packard calculators are normally repaired and reshipped within five (5) working days of receipt at any Repair Center. This is an average time and could possibly vary depending upon time of year and work load at the Repair Center.

Shipping Instructions

The calculator should be returned, along with completed Service Card, in its shipping case (or other protective package) to avoid in-transit damage. Such damage is not covered by warranty, and Hewlett-Packard suggests that the customer insure shipments to the Repair Center. A calculator returned for repair should include the ac adapter/recharger and the battery pack. Send these items to the address shown on the Service Card.

Whether the unit is under warranty or not, it is your responsibility to pay shipping charges to the Hewlett-Packard Repair Center.

After warranty repairs are completed, the Repair Center returns the unit with postage prepaid. On out-of-warranty repairs, the unit is returned C.O.D. (covering shipping costs and the service charge).


Further Information

Service contracts are not available. Calculator circuitry and design are proprietary to Hewlett-Packard, and Service Manuals are not available to customers.

Should other problems or questions arise regarding repairs, please call your nearest Hewlett-Packard Sales Office or Repair Center.




















Appendix B

Error Displays


Certain improper operations or conditions will cause the calculator display to show the word **SEE** followed by a number indicating the type of error. In addition, if the Print Mode switch **MAN**  **NORM** is set to **NORM** or **TRACE**, the error message will be printed (unless the calculator is out of paper.)

The error displays and their causes are shown below.

- SEE 1** Power interrupt to the memory. All memory is cleared. When power is restored and the calculator turned ON again, it "wakes up" in DEG (degree) trigonometric mode, FIX 2 display mode, and set to step A-000.
- SEE 2** Attempting to key in another instruction when all 200 steps of program memory are filled.
- SEE 3** More than three subroutine levels.
- SEE 4** No **LBL** or program marker as target of **GTO** or **GSB**.
- SEE 5** Any of the following numerical errors:

	where $x = 0$
	where $y = 0$ and $x \leq 0$
	where $y < 0$ and x is non-integer
	where $x < 0$
	where $x = 0$
	where $x \leq 0$
	where $x \leq 0$
	where $ x $ is > 1
	where $ x $ is > 1
	where $x = 0$
	where $n = 0$
	where $n \leq 1$
 or 	where $n \sum x^2 - (\sum x)^2 = 0$
 or 	where $n = 0$
	where $y = 0$
	where $\sum x = 0$
	where $x < 0$ or x is non-integer

SEE 6 Overflow of any storage register.

SEE 7  where $\text{ABS}(\text{INT } I) > 15$

 where $\text{ABS}(\text{INT } I) > 15$.

Stack Lift And LAST X

Your HP-95C calculator has been designed to operate in a natural, normal manner. As you have seen as you worked through this handbook, you are seldom required to think about the operation of the automatic memory stack—you merely work through calculations in the same way you would with a pencil and paper, performing one operation at a time.

There may be occasions, however, particularly as you program the HP-95C, when you wish to know the effect of a particular operation upon the stack. The following explanation and table should help you.

Digit Entry Termination

Most operations on the calculator, whether executed as instructions in a program or pressed from the keyboard, terminate digit entry. This means that the calculator knows that any digits you key in after any of these operations are part of a new number.

Stack Lift

There are three types of operations on the calculator, depending upon how they affect the stack lift. These are stack *disabling* operations, stack *enabling* operations, and *neutral* operations.

Disabling Operations

There are only four stack disabling operations on the calculator. These operations disable the stack lift, so that a number keyed in after one of these disabling operations writes over the current number in the displayed X-register and the stack does not lift. These special disabling operations are:

ENTER↓ **CLX** **Σ+** **Σ-**

Enabling Operations

The bulk of the operations on the keyboard, including one- and two-number mathematical functions like **x²** and **x³**, are stack *enabling* operations. These operations enable the stack lift, so that a number keyed in after one of the enabling operations lifts the stack.

Neutral Operations

Some operations, like **PRINTX** and **GTO 3**, are neutral; that is, they do not alter the previous status of the stack lift. Thus, if you have previously disabled the stack lift by pressing **ENTER↓**, then press **PRINTX** and key in a new number, that number will write over the number in the X-register and the stack will not lift. Similarly, if you have previously enabled the stack lift by executing, say, **x²**, then execute a **GTO 3** instruction followed by a digit entry sequence, the stack will lift.

The table below lists all legal operations on the HP-95C. Enabling operations are designated by a code of "E," disabling operations by "D," and neutral operations by "N." The table also indicates those operations that save the number from the X-register in the LAST X register.

Printed Symbol	Keystrokes	Enabling, Disabling, or Neutral	Saves x in LAST X
A	A	N	
ABS	f ABS	E	Yes
B	B	N	
	f BST	N	
C	C	N	
CHS	CHS *	E	
	g CLEAR A , B , C , D	N	
CL R	g CLEAR REG	N	
CL Σ	g CLEAR Σ	N	
CL S	g CLEAR STACK	N	
CL X	CLX	D	
COS	f COS	E	Yes
COS ⁻¹	g COS⁻¹	E	Yes
DEG	g DEG	N	
	f DELETE	N	
=	=	E	
DSZ	g DSZ	N	Yes
EEX	EEX *		
ENG0 ENG9	f ENG 0 through 9	N	
ENT↑	ENTER↑	D	
e ^x	g e^x	E	Yes
FRAC	g FRAC	E	Yes
FIX0 FIX9	f FIX 0 through 9	N	
GRD	g GRD	N	
GSB0 GSBA	GSB 0 through 9 , A through D	N	
GTO0 GTOD	GTO 0 through 9 , A through D	N	
→HMS	f →HMS	E	Yes
→H	f HMS→H	E	Yes
HMS-	f HMS-	E	Yes
HMS+	f HMS+	E	Yes
I	I	E	
INT	f INT	E	Yes
ISZ	f ISZ	N	
JPA0 JPD9	g JUMP A through D , followed by 0 through 9	N	
LSTX	f LAST X	E	
LBLO LBL9	f LBL 0 through 9	N	
LN	f LN	E	Yes
LOG	f LOG	E	Yes
LR	f LR	E	Yes
-	-	E	Yes
N!	g NI	E	Yes

Printed Symbol	Keystrokes	Enabling, Disabling, or Neutral	Saves x in LAST X
$\rightarrow P$	$\boxed{g} \rightarrow \boxed{P}$	E	Yes
$\%$	$\boxed{\%}$	E	Yes
$\Delta\%$	$\boxed{g} \Delta \boxed{\%}$	E	Yes
$\Sigma\%$	$\boxed{f} \boxed{\%} \boxed{\Sigma}$	E	Yes
π	$\boxed{g} \boxed{\pi}$	E	
$+$	$\boxed{+}$	E	Yes
PSE	$\boxed{f} \boxed{\text{PAUSE}}$	N	
PR R	$\boxed{f} \boxed{\text{PRINT}} \boxed{A}, \boxed{B}, \boxed{C}, \boxed{D}$	N	
PR Σ	$\boxed{f} \boxed{\text{PRINT}} \boxed{\text{REG}}$	N	
SPC	$\boxed{f} \boxed{\text{PRINT}} \boxed{\text{SPACE}}$	N	
PR S	$\boxed{f} \boxed{\text{PRINT}} \boxed{\text{STACK}}$	N	
PRTX	$\boxed{\text{PRINT}} \boxed{x}$	N	
$\rightarrow R$	$\boxed{f} \rightarrow \boxed{R}$	E	Yes
$R \downarrow$	$\boxed{R} \downarrow$	E	
$R \uparrow$	$\boxed{R} \uparrow$	E	
RAD	$\boxed{g} \boxed{\text{RAD}}$	N	
R 0 R 9	$\boxed{\text{RCL}} \boxed{0} \text{ through } \boxed{9}$	E	
R.0 R.5	$\boxed{\text{RCL}} \boxed{\cdot} \boxed{0} \text{ through } \boxed{\cdot} \boxed{5}$	E	
R i	$\boxed{f} \boxed{\text{RCL}} \boxed{i}$	E	
R Σ	$\boxed{\text{RCL}} \boxed{\Sigma}$	E	Yes
R/S	$\boxed{R/S}$	N	
RTN	$\boxed{f} \boxed{\text{RTN}} **$	N	
S	$\boxed{g} \boxed{S}$	E	Yes
SCIO SCI9	$\boxed{f} \boxed{\text{SCI}} \boxed{0} \text{ through } \boxed{9}$	N	
$\Sigma+$	$\boxed{\Sigma+}$	D	Yes
$\Sigma-$	$\boxed{\Sigma-}$	D	Yes
SIN	$\boxed{f} \boxed{\text{SIN}}$	E	Yes
SIN^{-1}	$\boxed{g} \boxed{\text{SIN}^{-1}}$	E	Yes
\sqrt{x}	$\boxed{\sqrt{x}}$	E	Yes
	$\boxed{\text{SST}}$	N	
S=0 S=9	$\boxed{\text{STO}} \boxed{=} \boxed{0} \text{ through } \boxed{9}$	E	
S-0 S-9	$\boxed{\text{STO}} \boxed{-} \boxed{0} \text{ through } \boxed{9}$	E	
S+0 S+9	$\boxed{\text{STO}} \boxed{+} \boxed{0} \text{ through } \boxed{9}$	E	
Sx0 Sx9	$\boxed{\text{STO}} \boxed{\times} \boxed{0} \text{ through } \boxed{9}$	E	
S 0 S 9	$\boxed{\text{STO}} \boxed{0} \text{ through } \boxed{9}$	E	
S.0 S.5	$\boxed{\text{STO}} \boxed{\cdot} \boxed{0} \text{ through } \boxed{\cdot} \boxed{5}$	E	
S I	$\boxed{\text{STO}} \boxed{I}$	E	
S i	$\boxed{f} \boxed{\text{STO}} \boxed{i}$	E	
TAN	$\boxed{f} \boxed{\text{TAN}}$	E	Yes
TAN^{-1}	$\boxed{g} \boxed{\text{TAN}^{-1}}$	E	Yes
x	\boxed{x}	E	Yes

Printed Symbol	Keystrokes	Enabling, Disabling, or Neutral	Saves x in LAST X
$X=0$		N	
$X \neq 0$		N	
$X < 0$		N	
$X > 0$		N	
$X=Y$		N	
$X \neq Y$		N	
$X \leq Y$		N	
$X > Y$		N	
\bar{x}		E	Yes
x^2		E	Yes
$1/x$		E	Yes
$X \leftrightarrow Y$		E	
\hat{y}		E	Yes
Y^x		E	Yes
10^x		E	Yes

* **CHS**, **EE**, , and the digits through are normally used as part of a digit entry sequence. However, if you press **CHS** after digit entry has been terminated by another operation, the stack lift will be enabled.

**The end of a program acts exactly like a operation.

Index

A

- Absolute value, **69**
- Accessories, optional, **196-197**
- Accessories, standard, **196**
- Accumulations, **87-89**
- Accumulations, printing, **89**
- AC Line operation, **197-198**
- Adding and subtracting time and angles, **77**
- Alteration, number, **69-70**
- Angles, adding and subtracting, **77**
- Arc sine, arc cosine, arc tangent, **75**
- Arithmetic, **25**
- Arithmetic average, **91**
- Arithmetic, chain, **52**
- Arithmetic, constant, **58**
- Arithmetic, storage register, **65**
- Arithmetic, vector, **99**
- Automatic display switching, **39**
- Automatic memory stack, **45-56**
- Average, arithmetic, **91**

B

- Back step, **125, 131-133**
- Battery care, **200-201**
- Battery charging, **21, 198**
- Battery operation, **198**
- Battery replacement, **199-200**
- Beginning of a program, **111**
- Blank display, **204**
- Blue key, **21**
- Branching, conditional, **144-150**
- Branching to another program, **187**
- Branching, unconditional, **141-144**

C

- Cable, security, **197**
- Calculator overflow, **42**
- Card window, **16**
- Chain calculations, **27, 52**
- Change, percent of, **74**

Charging, battery, **21, 198**
 Chart, prefix, **36**
 Clearing a program, **109-110, 125**
 Clearing storage registers, **64**
 Clearing the display, **22**
 Clearing the stack, **48**
 Coefficient of determination, **99**
 Conditionals and conditional branches, **144-150**
 Constant arithmetic, **58**
 Continuous memory, **16-17, 34, 45, 109, 199**
 Control, display, **33-37**
 Conversions, hours, minutes, seconds/decimal hours, **76**
 Conversions, polar/rectangular coordinate, **80-84**
 Correcting data, **95**
 Cosine, **75**

D

Decision-making, **144-150**
 Decrement, skip if zero, **175, 177-178**
 Deleting and correcting data, **95**
 Deleting instructions, **126, 134-136**
 Determination, coefficient of, **99**
 Deviation, standard, **93**
 Display, **21, 45**
 Display, blank, **204**
 Display, clearing the, **22**
 Display control, **33-37**
 Display, engineering notation, **35-37**
 Display, error, **42**
 Display, fixed point, **34**
 Display, low power, **43, 203**
 Display, scientific notation, **35**
 Display, SEE, **42**
 Display switching, automatic, **39**
 DO if TRUE rule, **146**

E

Editing, program, **125**
 Editing, using the printer for, **136-137**
 End of a program, **165**
 Engineering notation display, **35-37**
ENTER key, **25, 49**
 Error display, **42, 207**
 Error stops, **162**
 Estimate, linear, **98**
 Exchanging x and y, **47**
 Executing instructions, **113**

Execution, order of, **56**
Exponential functions, **85-87**
Exponents of 10, keying in, **40**

F

Factorials, **71**
Fixed point display, **34**
Flowcharts, **114-117**
Format of printed numbers, **37**
Fractional portion of a number, **70**
Functions, **23**
 one-number, **24, 50**
 two-number, **25, 51**
Function and key index, **8-11**

G

Going to a program marker, **126, 187**
Gold key, **21**

H

Hours, minutes, seconds/hours conversions, **76**

I

Increment, skip if zero, **175-177**
Indicator, low power, **43, 203**
Indirect store and recall, **179-182**
Initializing a program, **128**
Instructions, deleting, **134-136**
Instructions, executing, **113**
Integer portion of a number, **69**
Interruptions, program, **157-162**
I-register, **63, 175-185**

J

Jump, **187-191**

K

Keyboard, **8, 21**
Keyboard stops, **161-162**
Keycodes, **108-109**
Keying in exponents of 10, **40**
Keying in numbers, **22**
Keys, prefix, **21**

L

Labeling a program, **16**
Labels, **141, 187**
LAST X, **57-58, 209-212**
Limits, subroutine, **171**
Linear estimate, **98**
Linear regression, **96**
Loading a program, **15, 111-112**
Load verification with printer, **121**
Logarithms, **84-85**
Loops, **142**
Low power indicator, **43, 203-204**

M

Manipulating stack contents, **46**
Manual problem solving, **14**
Marker, program, going to, **112**
Markers, program, **106-108**
Mean, **91**
Memory, continuous, **34, 45**
Memory, program, **106-108**
Mistakes, recovering from, **57**
Modes, trigonometric, **74**
Modifying a program, **130**

N

Negative numbers, **22**
Nonrecordable operations, **125-126**
Number alteration, **69-70**
Number, fractional portion of, **70**
Number, integer portion of, **69**
Number, recovering a, **58**
Numbers, format of printed, **37**
Numbers, keying in, **22**
Numbers, negative, **22**
Numbers, raising to powers, **85**
Numbers, recalling, **62**
Numbers, storing, **62**

O

Operation, battery, **198**
Order of execution, **56**
Overflow, calculator, **42**
Overflow, storage register, **66**

P

- Paper, **197, 201**
- Paper advance, **23, 123**
- Paper, replacing, **202-203**
- Pause, **159-161**
- Percentages, **73**
- Percent of change, **74**
- Percent of sum, **90**
- Pi, **72**
- Polar/rectangular coordinate conversions, **81-84**
- Power pack, reserve, **197**
- Prefix chart, **36**
- Prefix keys, **21**
- Printed numbers, format of, **37**
- Printer, **22-23, 119-123**
- Printer maintenance, **203**
- Printer operation during a program, **119**
- Printer, using to create programs, **119-121**
- Printing accumulations, **89**
- Printing a program, **106, 122**
- Printing a space, **122-123**
- Printing storage registers, **64**
- Printing the stack, **45**
- Print mode switch, **22-23**
- Problems, **24, 30, 117, 123, 137, 150, 162, 172, 182, 192**
- Program, **105**
- Program, beginning of, **111**
- Program changes, verifying, **133-134**
- Program, initializing a, **128**
- Program, labeling a, **16**
- Program, loading a, **15, 111-112**
- Program marker, going to, **112**
- Program markers, **106-108**
- Program memory, **106-108**
- Program, modifying a, **130**
- Program, running a, **15, 112, 128**
- Program, single step execution of a, **125, 128-130**
- Program, stopping a, **113**
- Program, using as a subroutine, **191-192**
- Pythagorean Theorem program, **127**

R

- Raising numbers to powers, **85**
- Recalling numbers, **62, 175**
- Reciprocals, **70**
- Recovering a number, **58**
- Recovering from mistakes, **57**

Rectangular/polar coordinate conversions, **80-84**
 Register, **1**, **63**, **175-185**
 Registers, **45**
 Registers, storage, **61**
 Regression, linear, **96**
 Repairs, **205**
 Replacement, battery pack, **199-200**
 Replacing paper, **202-203**
 Reserve power pack, **197**
 Return, **165**
 Reviewing the stack, **46**
 Rolls, paper, **197**
 Routines, **141**
 RPN, **31**
 Running a program, **15**, **112**
 Run/stop, **157-159**

S

Scientific notation display, **35**
 Searching for a label, **142**, **143-144**
 Security cable, **197**
 SEE display, **42**, **162**, **207**
 Service, **203-204**
 Shipping instructions, **205**
 Sine, **75**
 Single-step execution, **125**, **128-130**
 Single-step viewing without execution, **131**
 Space, printing a, **122-123**
 Square roots, **71**
 Squaring, **72**
 Stack, **45-56**, **209-212**
 Stack, clearing the, **48**
 Stack contents, manipulating, **46**
 Stack, one-number functions and, **50**
 Stack, printing the, **45**
 Stack, reviewing the, **46**
 Stack, two-number functions and, **51**
 Standard deviation, **93**
 Statistical functions, **87-99**
 Stepping backward through a program, **131-133**
 Steps, program memory, **106**
 Stopping a program, **113**, **157-159**, **161-162**
 Storage registers, **61**
 Storage register arithmetic, **65**
 Storage registers, clearing the, **64**
 Storage register overflow, **66**
 Storage registers, printing the, **64**

Storing numbers, **62, 175**
Subroutines, **165-172**
Subroutine, calling from another program, **192**
Subroutine limits, **171**
Subroutine, using a program as, **191-192**
Sum, percent of, **90**

T

Tangent, **75**
Temperature range, **204**
Time, adding and subtracting, **77**
Transferring execution, **144**
Trigonometric functions, **74-76**
Trigonometric modes, **74**

U

User-definable keys, **111**

V

Vector arithmetic, **99**
Verification with printer, **121**
Verifying program changes, **133-134**
Viewing, single-step, **131**

W

Warranty, **204**

X

x and y, exchanging, **47**

Service Information

Must be **completed** and **returned** with your calculator, charger and batteries

Owner's Name

Date Purchased

Home Phone

Work Phone

Ship-to address for returning repaired calculator

Street Address

City

State

Zip

What Is The Problem Area?

- ☐ Intermittent Problem
- ☐ Printer Problem
- ☐ Keyboard Problem
- ☐ Display Problem

- ☐ Programming Problem
- ☐ HP Program
- ☐ User Supplied Program
- ☐ Continuous Memory Problem

Describe Problem:

Model No.

Serial No.

Preferred method of payment for out-of-warranty repairs. If not specified, unit will be returned C.O.D. ☐ BankAmericard ☐ Master Charge

Card No.

Expiration Date

Name appearing on credit card

- ☐ Purchase Order, companies with established Hewlett-Packard credit only. (Include copy of Purchase Order with shipment.)

Authorized Signature

P.O. Number

HEWLETT  PACKARD

HP-95C Registration Card

Please fill in and return this postage-paid card. This will enable us to send future product information to you.

Name _____

First Initial Last

Title _____

Company _____
(Include Bldg., Division, Room No., etc.)

Street/Box/Route _____

City _____ State _____ Zip _____

Date Product Received Month / / Day / Year /

- 1 Where was your calculator purchased?

- 101 ☐ Directly from an HP sales office or factory
- 102 ☐ By mail from HP
- 103 ☐ From any retail store

- 2 Check the one category best describing your job function.**

- 401 ☐ Top Management
402 ☐ Middle Management/Supervisory
403 ☐ Professional/Technical
404 ☐ Student
405 ☐ Other (Specify) _____

- 3 Rank two categories of applications for which your HP-95C will be used (1 for most important, 2 for second).

- 301 ☐ Engineering
302 ☐ Physical Science
303 ☐ Natural Science
304 ☐ Computer Science/Data Processing
305 ☐ Aviation/Marine Navigation

- 306 ☐ Statistics/Mathematics
307 ☐ Financial Analysis
308 ☐ Real Estate/Lending
309 ☐ Budgeting/Forecasting

If you are outside the United States:

- Return this card in the enclosed warranty envelope.
- If no envelope, please mail this card to the nearest Hewlett-Packard sales and service office.

Calculator Catalog and Buying Guide Request Card

Thank you for your order. Your friend or associate might also like to know about Hewlett-Packard calculators. If you would like us to send him/her the new **HEWLETT-PACKARD PERSONAL CALCULATOR DIGEST** (*the HP Magazine and Product Catalog*), please write his/her name and address on this postage-paid Request Card.

Primary Interest: ☐ Scientific Calculators ☐ Business Calculators
☐ Fully Programmable Calculators ☐ All

Name _____

Title _____

Company _____

Street _____

City _____ State _____ Zip _____

Valid in U.S. only

430J

Useful Conversion Factors

The following factors are provided to 10 digits of accuracy where possible. Exact values are marked with an asterisk. For more complete information on conversion factors, refer to *Metric Practice Guide E380-74* by the American Society for Testing and Materials (ASTM).

Length

1 inch	= 25.4 millimeters*
1 foot	= 0.304 8 meter*
1 mile (statute)†	= 1.609 344 kilometers*
1 mile (nautical)†	= 1.852 kilometers*
1 mile (nautical)†	= 1.150 779 448 miles (statute)†

Area

1 square inch	= 6.451 6 square centimeters*
1 square foot	= 0.092 903 04 square meter*
1 acre	= 43 560 square feet
1 square mile†	= 640 acres

Volume

1 cubic inch	= 16.387 064 cubic centimeters*
1 cubic foot	= 0.028 316 847 cubic meter
1 ounce (fluid)†	= 29.573 529 56 cubic centimeters
1 ounce (fluid)†	= 0.029 573 530 liter
1 gallon (fluid)†	= 3.785 411 784 liters*

Mass

1 ounce (mass)	= 28.349 523 12 grams
1 pound (mass)	= 0.453 592 37 kilogram*
1 ton (short)	= 0.907 184 74 metric ton*

Energy

1 British thermal unit	= 1 055.055 853 joules
1 kilocalorie (mean)	= 4 190.02 joules
1 watt-hour	= 3 600 joules*

Force

1 ounce (force)	= 0.278 013 85 newton
1 pound (force)	= 4.448 221 615 newtons

Power

1 horsepower (electric)	= 746 watts*
-------------------------	--------------

Pressure

1 atmosphere	= 760 mm Hg at sea level
1 atmosphere	= 14.7 pounds per square inch
1 atmosphere	= 101 325 pascals

Temperature

Fahrenheit	= 1.8 Celsius + 32
Celsius	= 5/9 (Fahrenheit - 32)
kelvin	= Celsius + 273.15
kelvin	= 5/9 (Fahrenheit + 459.67)
kelvin	= 5/9 Rankine

† U.S. values chosen. * Exact values.



1000 N.E. Circle Blvd., Corvallis, OR. 97330

For additional Sales and Service Information contact your local Hewlett-Packard Sales Office or call 800/648-4711. (In Nevada call collect 702/323-2704.)

00095-90001

Printed in U.S.A.