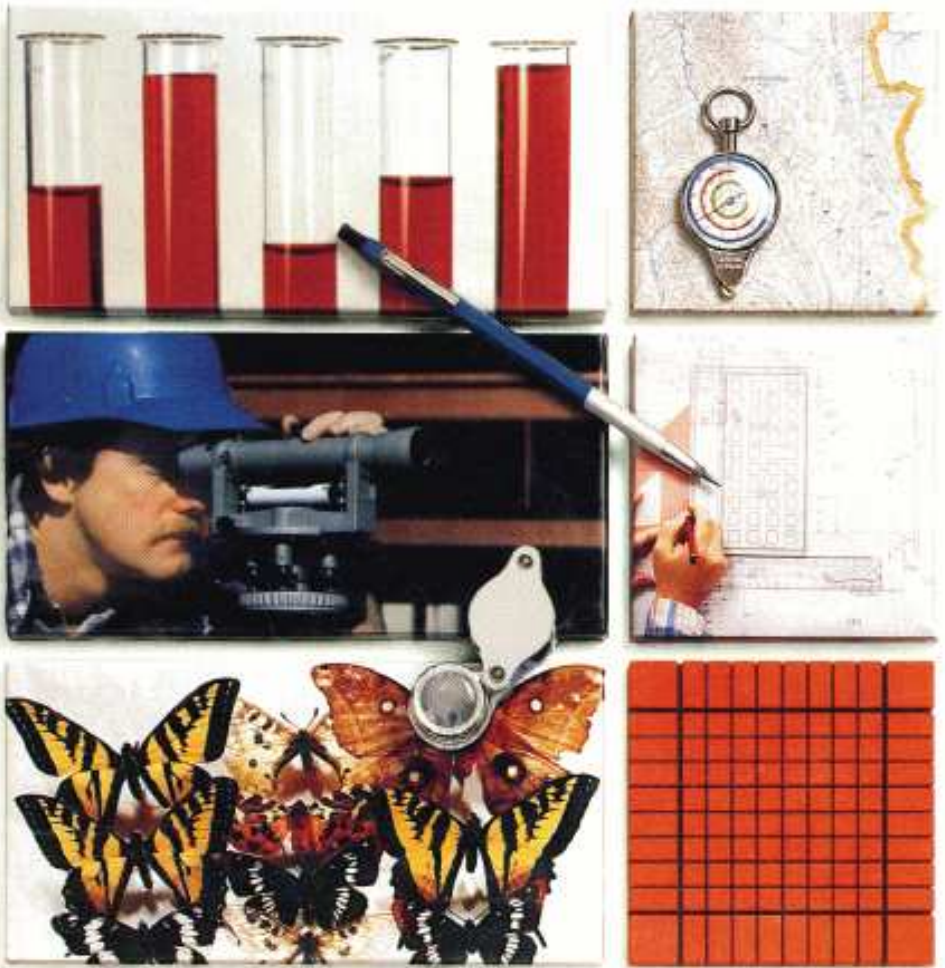


HEWLETT-PACKARD

# HP-10C

OWNERS HANDBOOK



Original scan copyright The Museum of HP Calculators  
[www.hpmuseum.org](http://www.hpmuseum.org)

Please support the HP Calculator Museum.

This version was translated to Microsoft Word and reformatted.

Version 1.

This document contains the following minor changes and corrections to the original 1982 manual:

- The description of sine/cos/tangent in the function index uses correct key symbols. Also added arcsine, arcos, arctan.
- Many index entries in the original manual appear to be incorrect. For example, the entry for “Looping” that refers to page 81 in the original manual. There is no mention of looping on page 81, but there is one at the bottom of page 80. I corrected these as I found them.
- The “Adding Instructions Within a Program” heading in section 7 is black instead of blue
- In the Table of Contents, section names are not followed by a colon.
- The index entries for keystrokes are slightly indented. This is because they contain a hidden character that is the first letter of the keystroke’s name. The hidden letter is necessary to get Word to put the index entries in the right place.
- The Programming and Function Key Indexes do not have blue boxes around the text.
- Incorrect keystroke images have been corrected (e.g., [LASTX] instead of [LSTX] and [GRD] instead of [GRAD]).

HEWLETT PACKARD

**HP-10C**

**Owner's Handbook**

February 1982

00010-90025

Printed in U.S.A.

Hewlett-Packard Company 1982

# Introduction

Congratulations! Your selection of an HP-10C calculator with Continuous Memory demonstrates your interest in quality, capability, and ease of use. You may be an experienced HP calculator user or you may be using an HP calculator for the first time. The primary goals of this handbook are to introduce you to all of the HP-10C features and to help you learn as quickly as possible how to use these features, regardless of your prior experience. Your anticipated use of the HP-10C and your level of prior experience with HP programmable calculators will determine how much time you need to devote to this handbook.

This handbook is divided into eight main sections. However, before reading these sections, gain some experience using your HP-10C by working through the introductory material entitled Your HP-10C: A Problem Solver, on page 7.

Section 1, Getting Started, covers the general operating information that both new and experienced HP calculator users should know. Sections 2 through 4 cover information that is more important to new users.

Sections 5 through 8 describe how to use the programming capabilities of the HP-10C.

The various appendices describe additional details of calculator operation, as well as warranty and service information.

The Function Key Index and Program Key Index at the back of the handbook can be used for quick reference to each function key and as a handy page reference to the comprehensive information inside the manual.

# Contents

|                          |          |
|--------------------------|----------|
| <b>Introduction.....</b> | <b>2</b> |
|--------------------------|----------|

|                      |          |
|----------------------|----------|
| <b>Contents.....</b> | <b>3</b> |
|----------------------|----------|

|   |          |
|---|----------|
| <b>Your HP-10C: A Problem Solver.....</b> | <b>7</b> |
|---|----------|

|                        |   |
|------------------------|---|
| Manual Solutions ..... | 8 |
|------------------------|---|

|                            |   |
|----------------------------|---|
| Programmed Solutions ..... | 9 |
|----------------------------|---|

|                                    |           |
|------------------------------------|-----------|
| <b>Part I HP-10-C Basics .....</b> | <b>11</b> |
|------------------------------------|-----------|

|  |           |
|--|-----------|
| <b>Section 1 Getting Started .....</b> | <b>12</b> |
|--|-----------|

|                        |    |
|------------------------|----|
| Power On and Off ..... | 12 |
|------------------------|----|

|                           |    |
|---------------------------|----|
| Low-Power Indication..... | 12 |
|---------------------------|----|

|                         |    |
|-------------------------|----|
| Keyboard Operation..... | 12 |
|-------------------------|----|

|                                      |    |
|--------------------------------------|----|
| Primary and Alternate Functions..... | 12 |
|--------------------------------------|----|

|                        |    |
|------------------------|----|
| Clearing Prefixes..... | 13 |
|------------------------|----|

|                       |    |
|-----------------------|----|
| Negative Numbers..... | 13 |
|-----------------------|----|

|                           |    |
|---------------------------|----|
| Keying in Exponents ..... | 13 |
|---------------------------|----|

|                        |    |
|------------------------|----|
| Display Clearing ..... | 14 |
|------------------------|----|

|                            |    |
|----------------------------|----|
| One-Number Functions ..... | 14 |
|----------------------------|----|

|   |    |
|---|----|
| Two-Number Functions and the <b>ENTER</b> Key ..... | 14 |
|---|----|

|                        |    |
|------------------------|----|
| Special Displays ..... | 17 |
|------------------------|----|

|                   |    |
|-------------------|----|
| Annunciators..... | 17 |
|-------------------|----|

|                                      |    |
|--------------------------------------|----|
| Radix Mark and Digit Separator ..... | 17 |
|--------------------------------------|----|

|                     |    |
|---------------------|----|
| Error Messages..... | 17 |
|---------------------|----|

|                              |    |
|------------------------------|----|
| Overflow and Underflow ..... | 17 |
|------------------------------|----|

|             |    |
|-------------|----|
| Memory..... | 18 |
|-------------|----|

|                         |    |
|-------------------------|----|
| Continuous Memory ..... | 18 |
|-------------------------|----|

|                        |    |
|------------------------|----|
| Resetting Memory ..... | 18 |
|------------------------|----|

|   |           |
|---|-----------|
| <b>Section 2 The Automatic Memory Stack, LAST X, and Data Storage .....</b> | <b>20</b> |
|---|-----------|

|  |    |
|--|----|
| The Automatic Memory Stack and Stack Manipulation..... | 20 |
|--|----|

|                                    |    |
|------------------------------------|----|
| Stack Manipulation Functions ..... | 21 |
|------------------------------------|----|

|   |    |
|---|----|
| Calculator Functions and the Stack..... | 22 |
|---|----|

|                           |    |
|---------------------------|----|
| Two-Number Functions..... | 23 |
|---------------------------|----|

|                          |    |
|--------------------------|----|
| Chain Calculations ..... | 24 |
|--------------------------|----|

|              |    |
|--------------|----|
| LAST X ..... | 25 |
|--------------|----|

|                           |    |
|---------------------------|----|
| Constant Arithmetic ..... | 26 |
|---------------------------|----|

|   |           |
|---|-----------|
| Storage Register Operations .....                   | 28        |
| Storing Numbers .....                               | 29        |
| Recalling Numbers .....                             | 29        |
| Clearing Data Storage Registers.....                | 30        |
| Storage Register Arithmetic.....                    | 30        |
| Storage Register Arithmetic Exercises .....         | 31        |
| Problems .....                                      | 31        |
| <b>Section 3 Numeric Functions .....</b>            | <b>33</b> |
| Pi .....  | 33        |
| Number Alteration Functions .....                   | 33        |
| One-Number Functions .....                          | 33        |
| General Functions .....                             | 34        |
| Trigonometric Operations .....                      | 34        |
| Trigonometric Functions.....                        | 35        |
| Time and Angle Conversions.....                     | 35        |
| Degrees/Radians Conversions .....                   | 36        |
| Logarithmic Functions .....                         | 36        |
| Two-Number Functions.....                           | 37        |
| Percentages .....                                   | 37        |
| The Power Function .....                            | 38        |
| Polar/Rectangular Coordinate Conversions .....      | 38        |
| Statistics Functions .....                          | 39        |
| Accumulating Statistics .....                       | 39        |
| Correcting Accumulated Statistics .....             | 42        |
| Mean.....   | 43        |
| Standard Deviation.....                             | 43        |
| Linear Regression.....                              | 45        |
| Linear Estimation and Correlation Coefficient ..... | 46        |
| <b>Section 4 Display Control.....</b>               | <b>49</b> |
| Display Mode Control .....                          | 49        |
| Fixed Decimal Display .....                         | 49        |
| Scientific Notation Display.....                    | 50        |
| Engineering Notation Display .....                  | 51        |
| Mantissa Display.....                               | 52        |
| Rounding at the Tenth Digit .....                   | 52        |
| <b>Part II Programming .....</b>                    | <b>55</b> |
| <b>Section 5 Programming Basics.....</b>            | <b>56</b> |
| Why Use Programs?.....                              | 56        |
| Creating a Program .....                            | 56        |
| Running a Program .....                             | 57        |
| Program Memory .....                                | 58        |
| Identifying Instructions in Program Lines .....     | 59        |

|   |           |
|---|-----------|
| Displaying Program Lines .....                                    | 60        |
| The <b>GTO</b> 00 instruction and Program line 00.....            | 61        |
| Expanding Program Memory; the <b>MEM</b> Key.....                 | 62        |
| Setting the Calculator to a Particular Program Line.....          | 64        |
| Executing a Program One Line at a Time .....                      | 65        |
| Interrupting Program Execution .....                              | 66        |
| Pausing During Program Execution .....                            | 66        |
| Stopping Program Execution Automatically.....                     | 67        |
| Stopping Program Execution Manually.....                          | 70        |
| Nonprogrammable Functions .....                                   | 70        |
| <b>Section 6 Branching and Looping .....</b>                      | <b>71</b> |
| Simple Branching .....  | 71        |
| Looping .....   | 71        |
| Conditional Branching .....                                       | 74        |
| <b>Section 7 Program Editing .....</b>                            | <b>83</b> |
| Changing an Instruction in a Program Line.....                    | 83        |
| Adding Instructions at the End of a Program .....                 | 83        |
| Adding Instructions Within a Program .....                        | 84        |
| Adding Instructions by Replacement .....                          | 84        |
| Adding Instructions by Branching .....                            | 85        |
| <b>Section 8 Multiple Programs.....</b>                           | <b>88</b> |
| Storing Another Program .....                                     | 88        |
| Running Another Program.....                                      | 90        |
| <b>Appendix A Stack Lift and LAST X .....</b>                     | <b>91</b> |
| Digit Entry Termination.....                                      | 91        |
| Stack Lift.....   | 91        |
| Disabling Operations.....   | 91        |
| Enabling Operations.....  | 91        |
| Neutral Operations .....  | 92        |
| LASTX.....  | 93        |
| <b>Appendix B Error Conditions.....</b>                           | <b>94</b> |
| Error 0: Improper Mathematics Operation.....                      | 94        |
| Error 1: Storage Register Overflow.....                           | 94        |
| Error 2: Improper Statistical Operation .....                     | 94        |
| Error 3: Statistical Register(s) Unavailable .....                | 95        |
| Error 4: Improper Line Number.....                                | 95        |
| Error 5: Improper Register Number .....                           | 95        |
| Error 9: Service.....   | 95        |
| Pr Error.....   | 95        |
| <b>Appendix C Battery, Warranty, and Service Information.....</b> | <b>96</b> |

|   |            |
|---|------------|
| Batteries.....  | 96         |
| Low-Power Indication .....  | 97         |
| Installing New Batteries .....  | 98         |
| Verifying Proper Operation (Self-Tests) .....                                     | 100        |
| Limited One-Year Warranty .....   | 101        |
| What We Will Do.....  | 101        |
| What Is Not Covered .....   | 101        |
| Warranty for Consumer Transactions in the United Kingdom.....                     | 102        |
| Obligation to Make Changes.....   | 102        |
| Warranty Information .....  | 102        |
| Service.....  | 103        |
| Obtaining Repair Service in the United States.....                                | 103        |
| Obtaining Repair Service in Europe .....  | 103        |
| International Service Information.....  | 105        |
| Service Repair Charge .....   | 105        |
| Service Warranty.....   | 105        |
| Shipping Instructions.....  | 105        |
| Further Information .....   | 106        |
| Programming and Applications Assistance .....                                     | 106        |
| Dealer and Product Information.....   | 106        |
| Temperature Specifications .....  | 107        |
| Federal Communications Commission Radio Frequency Interference<br>Statement ..... | 107        |
| <b>Function Key Index.....</b>  | <b>108</b> |
| Conversions .....   | 108        |
| Digit Entry .....   | 108        |
| Display Control .....   | 108        |
| Logarithmic and Exponential.....  | 108        |
| Mathematics.....  | 109        |
| Number Alteration.....  | 109        |
| Prefix Keys .....   | 109        |
| Stack Manipulation.....   | 109        |
| Statistics.....   | 109        |
| Storage.....  | 110        |
| Trigonometry .....  | 110        |
| <b>Programming Key Index.....</b>   | <b>111</b> |
| <b>Subject Index.....</b>   | <b>112</b> |
| <b>The HP-10C Keyboard and Continuous Memory .....</b>                            | <b>118</b> |



# Your HP-10C: A Problem Solver

Your HP-10C Programmable Scientific Calculator is a powerful problem solver that you can carry with you almost anywhere. It handles problems ranging from the simple to the complex, and can remember data. The HP-10C is so easy to program and use that it requires no prior programming experience or knowledge of programming languages.

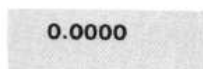
An important new feature of your HP-10C is its extremely low power consumption. This efficiency is responsible for the lightweight, compact model design, and eliminates the need for a cumbersome recharger. Power consumption in the HP-10C is so low that the average battery life in normal use is 6 to 12 months. In addition, the low-power indicator gives you plenty of warning before the calculator stops functioning.

The HP-10C also helps you to conserve power by automatically shutting its display off if it is left inactive for several minutes. But don't worry about losing data—any information you have keyed into your HP-10C is saved by Continuous Memory.

Your Hewlett-Packard calculator uses a unique operating logic, represented by the **[ENTER]** key, that differs from the logic in most other calculators. The power in HP calculator logic becomes obvious through use. Later we will cover the details of this logic, but right now let's get acquainted with **[ENTER]** in performing calculations.

For example, let's look at the arithmetic functions. First we have to get the numbers into the machine. To do this, key in the first number, press **[ENTER]** to separate the first number from the second, then key in the second number and press **[+]**, **[-]**, **[×]** or **[÷]**. Answers appear immediately after you press a numerical function key.

To get the feel of your new calculator, turn on the display by pressing the **[ON]** key. If any nonzero digits appear, you can press **[CLx]** to clear the display to 0.0000. If four digits are not displayed to the right of the decimal point, press **[f]** **[FIX]** 4 now so your display will match those in the following problems. (Displays illustrated in this handbook are set to the **[FIX]** 4 display setting unless otherwise specified.).



**Note:** An asterisk (\*) flashing in the lower

left corner of the display when the calculator is turned on signifies that the available battery power is running low. To install new batteries, refer to appendix C.

## Manual Solutions

It is not necessary to clear the calculator between problems. But if you make a digit entry mistake, press **[CLx]** and key in the correct number.\*

| To Solve:         | Keystrokes                                    | Display |
|-------------------|---|---------|
| $9 + 6 = 15$      | <b>[9]</b> <b>[ENTER]</b> <b>[6]</b> <b>+</b> | 15.0000 |
| $9 - 6 = 3$       | <b>[9]</b> <b>[ENTER]</b> <b>[6]</b> <b>-</b> | 3.0000  |
| $9 \times 6 = 54$ | <b>[9]</b> <b>[ENTER]</b> <b>[6]</b> <b>×</b> | 54.0000 |
| $9 \div 6 = 1.5$  | <b>[9]</b> <b>[ENTER]</b> <b>[6]</b> <b>÷</b> | 1.5000  |

Notice that in the four examples:

- Both numbers are in the calculator before you press **+**, **-**, **×**, or **÷**.
- **[ENTER]** is used only to separate two numbers that are keyed in one after the other.
- Pressing a numerical function key, in this case **+**, **-**, **×**, or **÷**, causes the function to execute immediately and the result to be displayed.

To see the close relationship between manual and programmed problem-solving, let's first calculate the solution to a problem manually, that is, from the keyboard. Then we'll use a program to calculate the solution to the same problem and others like it.

Most conventional home water heaters are cylindrical in shape. You can easily calculate the heat loss from such a tank using the formula  $q = h \times A \times T$ , where:

---

\* If you are new to HP calculators, you will notice most keys have two labels. For the main function—printed in white on top of the key—just press that key. For the function printed in gold, press the **[f]** key first.

$q$  is the heat loss from the water heater (Btu per hour).

$h$  is the heat-transfer coefficient.

$A$  is the total surface area of the cylinder.

$T$  is the temperature difference between the cylinder and the surrounding air.



**Example:** Assume you have a 52-gallon cylindrical water heater and you wish to determine how much energy is being lost because of poor insulation. In initial measurements, you found an average temperature difference between the heater surface and surrounding air of  $15^{\circ}\text{F}$ . The surface area of the tank is 30 square feet and the heat transfer coefficient is approximately 0.47. To calculate the heat loss of the water heater, simply press the following keys in order.

| Keystrokes                              | Display  |  |
|---|----------|--|
| 15 <input type="button" value="ENTER"/> | 15.0000  | Input temperature difference ( $T$ ) and area of water heater ( $A$ ). |
| 30                                      | 30.      |  |
| <input type="button" value="×"/>        | 450.0000 | Calculates $A \times T$ .  |
| .47                                     | 0.47     | Heat-transfer coefficient ( $h$ ).                                     |
| <input type="button" value="×"/>        | 211.5000 | Heat loss in Btu per hour ( $h \times AT$ ).                           |

## Programmed Solutions

The heat loss for the water heater in the preceding example was calculated for a  $15^{\circ}$  temperature difference. But suppose you want to calculate the heat loss for several temperature differences? You could perform each heat loss calculation manually. However, an easier and faster method is to write a program that will calculate the heat loss for any temperature difference.

**Writing the Program.** The program is the same series of keystrokes you executed to solve the problem manually.

**Loading the Program.** To load the instructions of the program into the HP-10C press the following keys in order. The calculator records (remembers) the instructions as you key them in. (The display gives you information you will find useful later, but which you can ignore for now.)

| Keystrokes              | Display  |   |
|-------------------------|----------|---|
| <b>[P/R]</b>            | 00-      | Places the HP-10C in Program mode. (Program annunciator appears.) |
| <b>[f] CLEAR [PRGM]</b> | 00-      | Clears program memory.  |
| <b>[ENTER]</b>          | 01-      | The same keys you pressed to solve the problem manually.          |
| <b>[3]</b>              | 02-      |   |
| <b>[0]</b>              | 03-      |   |
| <b>[x]</b>              | 04-      |   |
| <b>.</b>                | 05-      |   |
| <b>4</b>                | 06-      |   |
| <b>7</b>                | 07-      |   |
| <b>[x]</b>              | 08-      | Places the HP-10C in Run mode. (Program annunciator cleared)      |
| <b>[P/R]</b>            | 211.5000 |   |

**Running the Program.** Press the following keys to run the program.

| Keystrokes      | Display  |   |
|-----------------|----------|---|
| 15              | 15.      | The first temperature difference.                   |
| <b>[R/S]</b>    | 211.5000 | The Btu heat loss you calculated earlier by hand.   |
| 18 <b>[R/S]</b> | 253.8000 | The Btu heat loss for a new temperature difference. |

With the program you have loaded, you can now quickly calculate the Btu heat loss for many temperature differences. Simply key in the desired difference and press **[R/S]**. For example, complete the table at the right.

The answers you should see are 141.0000, 169.2000, 197.4000, 225.6000, 253.8000, and 282.0000.

| Temp. Diff. | Btu Heat Loss |
|-------------|---------------|
| 10          | ?             |
| 12          | ?             |
| 14          | ?             |
| 16          | ?             |
| 18          | ?             |
| 20          | ?             |


Programming is that easy! The calculator remembers a series of keystrokes and then executes them whenever you wish. Now that you have had some experience in using your HP-10C, let's take a look at some of the calculator's important operating details.

**Part I**  
**HP-10C**  
**Basics**

## Section 1

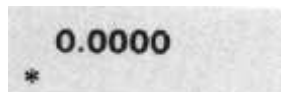
# Getting Started

### Power On and Off

The  key turns the HP-10C on and off.\* To conserve power, the HP-10C automatically turns itself off after several minutes of inactivity.

### Low-Power Indication

When a flashing asterisk, which indicates low battery power, appears in the lower left-hand side of the display, there is no reason to panic. You still have plenty of calculator time remaining: at least 15 minutes if you continuously run programs, and at least an hour if you manually perform operations. Refer to appendix C (page 96) for information on replacing the batteries.







### Keyboard Operation

#### Primary and Alternate Functions

Most keys on your HP-10C perform one primary and one alternate function. The primary function of any key is indicated by the character(s) on the upper face of the key. The alternate function is indicated by the character(s) printed in gold above the key.

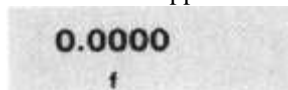


- To select the alternate function printed in gold above a key, first press the gold prefix key , then press the function key. For example:  .
- To select the primary function printed on the face of a key, press only that key. For example: .

---

\* Note this key is shorter than the others to prevent its being inadvertently pressed.

Notice that when you press the  $\boxed{f}$  prefix key, the **f** annunciator appears and remains in the display until a function key is pressed to complete the sequence.



## Clearing Prefixes

Certain function commands require two parts: a prefix and a number or another key. The prefixes are  $\boxed{f}$ ,  $\boxed{STO}$ ,  $\boxed{RCL}$ , and  $\boxed{GTO}$ ;  $\boxed{f}$   $\boxed{FIX}$   $\boxed{f}$   $\boxed{SCI}$ , and  $\boxed{f}$   $\boxed{ENG}$ . If you make a mistake while keying in a prefix for a function, press  $\boxed{f}$   $\boxed{CLEAR}$   $\boxed{PREFIX}$  to cancel the error. The  $\boxed{PREFIX}$  key is also used to show the mantissa of a displayed number, so all 10 digits of the number in the display will appear for a moment after the  $\boxed{PREFIX}$  key is pressed.

## Negative Numbers

To make a displayed number negative—either one that has just been keyed in or one that has resulted from a calculation—simply press  $\boxed{CHS}$  (*change sign*). When the display shows a negative number, pressing  $\boxed{CHS}$  removes the minus sign from the display, making the number positive.

## Keying in Exponents

$\boxed{EEX}$  (*enter exponent*) is used whenever an exponent is a part of a number you are keying in. To use  $\boxed{EEX}$ , first key in the mantissa, then press  $\boxed{EEX}$  and key in the exponent. For example, to key in Avogadro's number ( $6.0225 \times 10^{23}$ ):

| Keystrokes      | Display   |  |
|-----------------|-----------|--|
| 6.0225          | 6.0225    |  |
| $\boxed{EEX}$   | 6.0225 00 | The 00 prompts you to key in the exponent. |
| 2               | 6.0225 02 |  |
| 3               | 6.0225 23 | ( $6.0225 \times 10^{23}$ )                |
| $\boxed{ENTER}$ | 6.0225 23 | Enters number.                             |

To key in a number having a negative exponent of 10, first key in the number, press  $\boxed{EEX}$ , then key in the exponent, and then press  $\boxed{CHS}$  (*change sign*) to make the exponent negative.\* For example, key in Planck's constant ( $6.6262 \times 10^{-34}$  Joule-seconds) and multiply it by 50:

| Keystrokes           | Display   |
|----------------------|-----------|
| 6.6262 $\boxed{EEX}$ | 6.6262 00 |
| 34                   | 6.6262 34 |

---

\*  $\boxed{CHS}$  may also be pressed *before* the exponent, with the same result (unlike the mantissa, whose number entry must precede  $\boxed{CHS}$  ).

| Keystrokes     | Display                              |
|----------------|--------------------------------------|
| <b>[CHS]</b>   | 6.6262 -34                           |
| <b>[ENTER]</b> | 6.6262 -34                           |
| 50 <b>[X]</b>  | 3.3131 -32            Joule-seconds. |

**Note:** Decimal digits from the mantissa which spill into the exponent field will disappear from the display when you press **[EEX]**, but will be retained internally.

**[EEX]** will not operate with a number having more than seven digits to the left of the decimal point or radix, or with a mantissa whose absolute value is smaller than 0.000001. To key in such a number, use a form having a higher or lower exponent value, as appropriate. For example,  $123456789.8 \times 10^{23}$  can be keyed in as  $1234567.898 \times 10^{25}$ ;  $0.00000025 \times 10^{-15}$  can be keyed in as  $2.5 \times 10^{-22}$ .

Display Clearing

- In Run mode, pressing **[CLx]** (*clear X*) clears all digits in the display (X-register) to zero.
- In Program mode, **[CLx]** is programmable; it does *not* delete the currently displayed instruction. It is stored in the calculator as a programmed instruction.

One-Number Functions

A one-number function is any numeric function that performs an operation using only one number. To use any one-number function:

1. Key the number into the display (if it is not already there).
2. Press the function key(s).

| Keystrokes              | Display |
|-------------------------|---------|
| 45                      | 45.     |
| <b>[f]</b> <b>[LOG]</b> | 1.6532  |

Two-Number Functions and the **[ENTER]** Key

A two-number function must have two numbers present in the calculator before executing the function. **[+]**, **[-]**, **[X]** and **[÷]** are examples of two-number functions.

As in basic arithmetic, the two numbers should be keyed into the calculator in the order they would appear if the calculation were written down on paper from left to right.



**The  $\boxed{\text{ENTER}}$  Key.** If one of the numbers you need for a two-number function is already in the calculator as the result of a previous operation, you do not need to use the  $\boxed{\text{ENTER}}$  key. However, when you must key in two numbers before performing a function, use the  $\boxed{\text{ENTER}}$  key to separate the two numbers.

To place two numbers into the calculator and perform a two number function such as  $2 \div 3$ :

1. Key in the first number.
2. Press  $\boxed{\text{ENTER}}$  to separate the first number from the second.
3. Key in the second number.
4. Press the function key(s).

| Keystrokes             | Display |
|------------------------|---------|
| 2                      | 2.      |
| $\boxed{\text{ENTER}}$ | 2.0000  |
| 3                      | 3.      |
| $\boxed{\div}$         | 0.6667  |

Now try this problem. Notice that you have to press  $\boxed{\text{ENTER}}$  to separate numbers only when they are being keyed in one immediately after the other. A previously calculated result (intermediate result) will be automatically separated from a new number you key in.

To solve  $(9 + 17 - 4 + 23) \div 4$ :

| Keystrokes               | Display |                             |
|--------------------------|---------|-----------------------------|
| 9 $\boxed{\text{ENTER}}$ | 9.0000  |                             |
| 17 $\boxed{+}$           | 26.0000 | $(9 + 17).$                 |
| 4 $\boxed{-}$            | 22.0000 | $(9 + 17 - 4).$             |
| 23 $\boxed{+}$           | 45.0000 | $(9 + 17 - 4 + 23).$        |
| 4 $\boxed{\div}$         | 11.2500 | $(9 + 17 - 4 + 23) \div 4.$ |

Even more complicated problems are solved in the same simple manner—using automatic storage of intermediate results. (For problems with nested parentheses, refer to Chain Calculations, page 24.)

**Example:** Solve  $(6 + 7) \times (9 - 3)$ .

First solve for the intermediate result of  $(6 + 7)$ :

| Keystrokes             | Display |
|------------------------|---------|
| 6                      | 6.      |
| $\boxed{\text{ENTER}}$ | 6.0000  |
| 7                      | 7.      |

**Keystrokes****Display****[+]**

13.0000

 $(6 + 7).$ 

Now perform  $(9 - 3)$ . Since another pair of numbers must be keyed in, one immediately after the other, use the **[ENTER]** key again to separate the first number (9) from the second (3). (There is no need to press **[ENTER]** to separate the 9 from the previous intermediate result of 13 that is already in the calculator—the results of previous calculations are stored automatically.) To solve  $(9 - 3)$ :

**Keystrokes****Display**

9

9.

**[ENTER]**

9.0000

3

3.

**[-]**

6.0000

 $(9 - 3).$ 

Then multiply the intermediate results (13 and 6) together for the final answer.

**Keystrokes****Display****[×]**

78.0000

 $(6 + 7) \times (9 - 3) = 78.$ 

Notice that the HP-10C automatically stored the intermediate results for you and used them 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.

**Remember:**

- The **[ENTER]** key is used for separating the second number from the first in any operation requiring the sequential entry of two numbers.
- Any new digits keyed in following a calculation are automatically treated as a new number.
- Intermediate results are stored on a last-in, first-out basis.

Now try these problems. Work through them as you would with pencil and paper. Don't be concerned about intermediate answers—they are handled automatically by your HP-10C.

$$(16 \times 38) - (13 \times 11) = 465.0000$$

$$(27 + 63) \div (33 \times 9) = 0.3030$$

$$\sqrt{(16.38 \times 0.55)} \div 0.05 = 60.0300$$

$$4 \times (17 - 12) \div (10 - 5) = 4.0000$$

## Special Displays

### Annunciators

Your HP-10C display contains five annunciators that tell you the status of the calculator during certain operations. The annunciators are described, with the operations they refer to, in the appropriate sections of this handbook.

\* f RAD GRAD PRGM  
**Display Annunciator Set**

### Radix Mark and Digit Separator

A radix mark is the divider between the integer and fractional portions of a number. A digit separator distinguishes the groups of digits in a large number. In some countries the radix is a decimal point and the digit separator is a comma, while in other countries the reverse is true. To interchange the radix and digit separator conventions on your HP-10C, turn off the calculator, then hold down the  $\square \cdot$  key, turn the calculator back on, and release the  $\square \cdot$  key ( $\square \cdot / \square \text{ON}$ ).

$\square \cdot / \square \text{ON}$   12,345,678.91  
12.345.678,91

**Radix Mark/Digit Separator Interchange**

### Error Messages

If you attempt a calculation using an improper parameter, such as attempting to find the square root of a negative number, an error message will appear in the display. For a complete listing of error messages and their causes, refer to appendix B.

#### Keystrokes

4  $\square \text{CHS}$   $\square \sqrt{x}$

$\square \text{CLx}$

#### Display

**Error 0**

**-4.0000**

To clear any error message, press  $\square \text{CLx}$  (or any other key), then resume normal calculator operation.

### Overflow and Underflow

**Overflow.** When the result of a calculation in the display (X-register) is a number with a magnitude greater than  $9.99999999 \times 10^{99}$ , all 9's are displayed with the appropriate sign. When

(-) 9.999999 99

**Overflow Display**

overflow occurs in a running program, execution halts and the overflow display appears.

**Underflow.** If the result of a calculation is a number with a magnitude less than  $1.000000000 \times 10^{-99}$ , zero will be substituted for that number. Underflow will not halt the execution of a calculation or a running program.

## Memory

### Continuous Memory

The Continuous Memory feature in your HP-10C retains the following in the calculator, even when the display is turned off:



- All numeric data stored in the calculator.
- All programs stored in the calculator.
- Display mode and setting.
- Trigonometric mode (Degrees, Radians, or Grads).

When the HP-10C is turned on, it always “wakes up” in Run mode (**PRGM** annunciator cleared), even if it was in Program mode (**PRGM** annunciator displayed) when last turned off.

If the calculator is turned off, Continuous Memory is preserved for a few minutes when the batteries are removed. Data and programs are preserved longer than the calculator’s status is. Refer to appendix C for instructions on changing batteries.

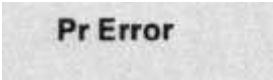
### Resetting Memory

If at any time you want to reset (entirely clear) the HP-10C Continuous Memory, do the following:

1. Turn the HP-10C off.
2. Hold down the  key, and press .



When you perform the memory reset operation, the error message shown to the right is displayed. Press **[CLx]** to clear the message.



**Pr Error**

**Note:** Continuous Memory can be inadvertently reset if the calculator is dropped or otherwise traumatized.

## Section 2

# The Automatic Memory Stack, LAST X, and Data Storage

## The Automatic Memory Stack and Stack Manipulation

Your HP-10C can take you through complex calculations easily because it automatically retains and returns intermediate results. These features are based on the automatic memory stack and the **ENTER** key.

The Automatic  
Memory Stack Registers

|   |        |                  |
|---|--------|------------------|
| T | 0.0000 |                  |
| Z | 0.0000 |                  |
| Y | 0.0000 |                  |
| X | 0.0000 | Always displayed |

When your HP-10C is in Run mode (that is, when the **PRGM** annunciator is not displayed) the number that appears in the display is the number in the X-register.

Any number keyed in or resulting from the execution of a numeric function is placed in the display (X-register). Executing a function or keying in a number will cause numbers already in the stack to lift, remain in the same register, or drop, depending upon the type of operation being performed. Numbers in the stack are available on a last-in, first-out basis. If, for example, the stack were loaded as shown on the left of each of the three illustrations shown below, pressing the indicated keys would result in the stack rearrangement shown on the right of each illustration.

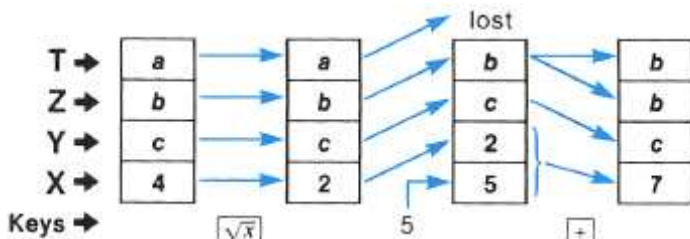






you do not need to use **[ENTER]**. Why? Executing most HP-10C functions has two results:

1. The specified function is executed.
2. The automatic memory stack is *enabled*; that is, the stack will lift automatically when the *next* number is keyed in or recalled.



There are four functions—**[ENTER]**, **[CLx]**, **[ $\Sigma+$ ]**, and **[ $\Sigma-$ ]**—which *disable* the stack. They do *not* provide for the lifting of the stack when the *next* number is keyed in or recalled. Following the execution of one of these functions, a new number will simply write over the currently displayed number instead of causing the stack to lift. (Although the stack lifts when **[ENTER]** is pressed, it will *not* lift when the *next* number is keyed in or recalled. The operation of **[ENTER]** illustrated on page 21 shows how **[ENTER]** disables the stack.) In most cases, the above effects will come so naturally that you won't even think about them.\*

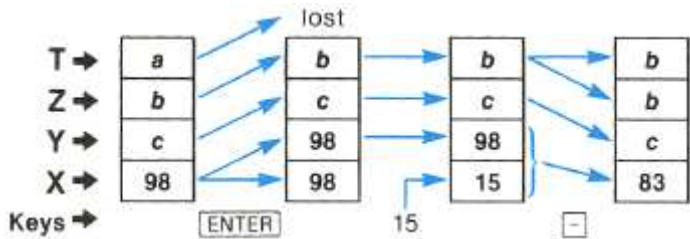
## Two-Number Functions

An important aspect of two-number functions is the positioning of the numbers in the stack. To execute an arithmetic function, the numbers should be positioned in the stack in the same way that you would vertically position them on paper. For example:

$$\begin{array}{r} 98 \\ -15 \\ \hline \end{array} \quad \begin{array}{r} 98 \\ +15 \\ \hline \end{array} \quad \begin{array}{r} 98 \\ \times 15 \\ \hline \end{array} \quad \begin{array}{r} 98 \\ 15 \\ \hline \end{array}$$

The numbers are positioned in the calculator in the same way, with the first (or top) number in the Y-register, and the second (or bottom) number in the X-register. When the arithmetic operation is performed, the stack drops, leaving the result in the X-register. Here is how an entire subtraction operation is executed:

\* For a further discussion of the stack, refer to appendix A.



The same number positioning would be used to add 15 to 98, multiply 98 by 15, or divide 98 by 15.

Chain Calculations

The simplicity and power of your HP-10C logic system are very apparent during chain calculations. The automatic stack lift and stack drop make it possible to do chain calculations without the necessity of keying in parentheses or storing intermediate results. An intermediate result in the X-register is automatically copied into the Y-register when a number is keyed in after a function key is pressed.\*

Virtually every chain calculation you are likely to encounter can be done using only the four stack registers. To avoid having to store an intermediate result in a storage register, you should begin every chain calculation at the innermost number or pair of parentheses and then work outward—just as you would if you were doing the calculation with pencil and paper. For example, consider the calculation of  $3 [4 + 5 (6 + 7)]$

| Keystrokes   | Display  |  |
|--------------|----------|--|
| 6[ENTER]7[+] | 13.0000  | Intermediate result of (6 + 7).        |
| 5[×]         | 65.0000  | Intermediate result of 5 (6 + 7).      |
| 4[+]         | 69.0000  | Intermediate result of [4 + 5(6 + 7)]. |
| 3[×]         | 207.0000 | Final result: 3 [4 + 5 (6 + 7)].       |

As you can see, we worked through the problem one operation at a time. The stack automatically dropped after each two-number calculation. And, after each calculation, the stack automatically lifted when a new number was keyed in. The next example illustrates this.

\* Except for [ENTER], [CLx], [Σ+], and [Σ-].

**Example:** Instead of the arrow diagrams we've used earlier, we'll use a table to follow stack operation as we solve the expression

$$(3 + 4) \times (6 - 4) \div 2:$$

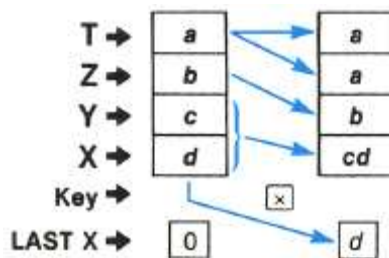
|        |   |       |   |   |   |       |
|--------|---|-------|---|---|---|-------|
|        | 1 | 2     | 3 | 4 | 5 | 6     |
| T →    | a | b     | b | b | b | c     |
| Z →    | b | c     | c | b | c | 7     |
| Y →    | c | 3     | 3 | c | 7 | 6     |
| X →    | 3 | 3     | 4 | 7 | 6 | 6     |
| Keys → | 3 | ENTER | 4 | + | 6 | ENTER |

|        |   |   |    |    |    |
|--------|---|---|----|----|----|
|        | 7 | 8 | 9  | 10 | 11 |
| T →    | c | c | c  | c  | c  |
| Z →    | 7 | c | c  | c  | c  |
| Y →    | 6 | 7 | c  | 14 | c  |
| X →    | 4 | 2 | 14 | 2  | 7  |
| Keys → | 4 | - | ×  | 2  | ÷  |

## LAST X

The HP-10C LAST X register, a separate data storage register, preserves the value that was last in the display before execution of a numeric function.\* This feature allows you to perform constant arithmetic (reusing a number without re-entering it). It can also assist you in error recovery.



**Example:** To multiply two separate values, such as 45.575 meters and 25.331 meters, by  $\pi$ :

---

\* The exceptions are the statistics functions  $\overline{x}$ ,  $s$ , and  $\overline{R}$ , which ignore the value in the display (X register) and calculate a result from data in the statistics storage registers ( $R_0$ - $R_5$ ).

|          | 1      | 2       | 3       | 4        |
|----------|--------|---------|---------|----------|
| T →      | a      | b       | b       | b        |
| Z →      | b      | c       | c       | b        |
| Y →      | c      | 45.5750 | 45.5750 | c        |
| X →      | 45.575 | 45.5750 | 3.1416  | 143.1781 |
| Keys →   | 45.575 | ENTER   | f π     | x        |
| LAST X → |        |         |         | 3.1416   |

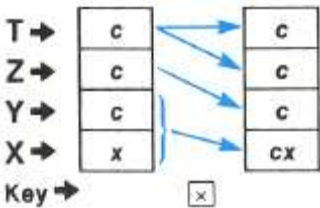
|          | 5        | 6        | 7        |
|----------|----------|----------|----------|
| T →      | b        | c        | c        |
| Z →      | c        | 143.1781 | c        |
| Y →      | 143.1781 | 25.3310  | 143.1781 |
| X →      | 25.331   | 3.1416   | 79.5797  |
| Keys →   | 25.331   | f LASTx  | x        |
| LAST X → | 3.1416   | 3.1416   | 3.1416   |

**LASTx** also makes it easy to recover from keystroke mistakes, such as executing the wrong function or keying in the wrong number. For example, to divide 287 by 13.9 after you have mistakenly divided by 12.9:

| Keystrokes | Display  |   |
|------------|----------|---|
| 287 ENTER  | 287.0000 |   |
| 12.9 ÷     | 22.2481  | Oops! The wrong divisor.  |
| f LASTx    | 12.9000  | Retrieves from LAST X the last entry to the X-register (the incorrect divisor) before ÷ was executed. |
| x          | 287.0000 | Performs the reverse of the function that produced the wrong answer.                                  |
| 13.9 ÷     | 20.6475  | The correct answer.   |

Constant Arithmetic

Using the LAST X register (described above) is one way to perform arithmetic with a constant. Another way is by filling the stack with the constant number. The method used



depends on the type of calculation (refer to examples above and below).

Because the number in the T-register remains there when the stack drops, this number can be used as a constant in arithmetic operations.

Load the stack with a constant by keying the constant into the X-register and pressing **ENTER** three times. Use the constant by keying in your initial factor and executing your planned series of arithmetic operations. Each time the stack drops, a copy of the constant will be made available for your next calculation and a new copy of the constant is reproduced in the T-register.

**Example:** A bacteriologist tests a certain strain of microorganisms whose population typically increases by 15% each day (a growth factor of 1.15). If he starts with a sample culture of 1000, what will be the bacteria population at the end of each day for five consecutive days?



**Method:** Use **ENTER** to put the constant growth factor (1.15) in the Y-, Z-, and T-registers and put the original population (1000) in the display (X-register). Thereafter, you get the new daily population whenever you press **×**. In **FIX** 2mode (press **f** **FIX** 2), your stacks would look like this:

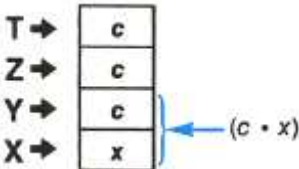
|        | 1 | 2    | 3            | 4            |
|--------|---|------|--------------|--------------|
| T →    | a | b    | c            | d            |
| Z →    | b | c    | d            | 1.15         |
| Y →    | c | d    | 1.15         | 1.15         |
| X →    | d | 1.15 | 1.15         | 1.15         |
| Keys → |   | 1.15 | <b>ENTER</b> | <b>ENTER</b> |

|        | 5            | 6      | 7        | 8        |
|--------|--------------|--------|----------|----------|
| T →    | 1.15         | 1.15   | 1.15     | 1.15     |
| Z →    | 1.15         | 1.15   | 1.15     | 1.15     |
| Y →    | 1.15         | 1.15   | 1.15     | 1.15     |
| X →    | 1.15         | 1,000. | 1,150.00 | 1,322.50 |
| Keys → | <b>ENTER</b> | 1000   | <b>×</b> | <b>×</b> |

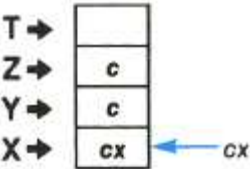
|        | 9                                | 10                               | 11                               |
|--------|----------------------------------|----------------------------------|----------------------------------|
| T →    | 1.15                             | 1.15                             | 1.15                             |
| Z →    | 1.15                             | 1.15                             | 1.15                             |
| Y →    | 1.15                             | 1.15                             | 1.15                             |
| X →    | 1,520.88                         | 1,749.01                         | 2,011.36                         |
| Keys → | <input type="button" value="×"/> | <input type="button" value="×"/> | <input type="button" value="×"/> |

When you press  the first time, you calculate  $1.15 \times 1000$ . The result (1,150.00) is displayed in the X-register, the stack drops, and a new copy of the constant is generated in the T-register. That is, given a constant  $c$  and a variable  $x$ , each time you press

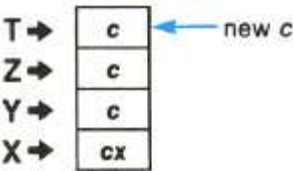
1. A new calculation involving the X- and Y-registers takes place.



2. The result of the calculation is placed in the display (X-register) and the contents of the rest of the stack drop.



3. A new copy of the number last in T (in this case, our constant) is generated in T.



Since a new copy of the growth factor is duplicated in the T-register each time the stack drops, you never have to re-enter it.

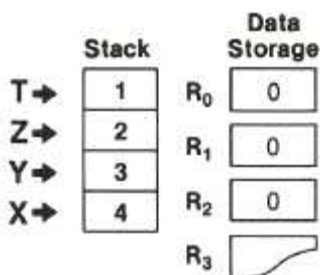
## Storage Register Operations

Storing and recalling numbers are operations involving the display (X-register) and the 10 data storage registers of the HP-10C. Data storage registers are entirely separate from the stack and LAST X registers.

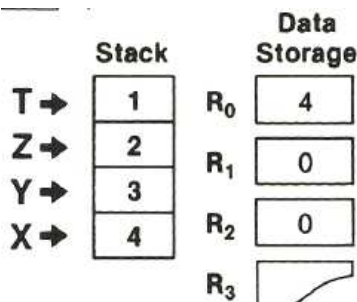
## Storing Numbers

**[STO]** (*store*). When followed by a storage register address (0 through 9), this key copies a number from the display (X-register) into the data storage register specified by the address number.

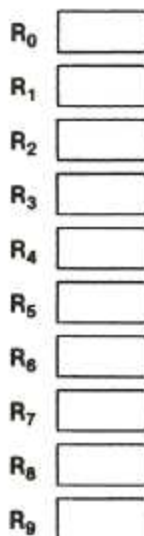
If ...



and you press **[STO]**0, then ...



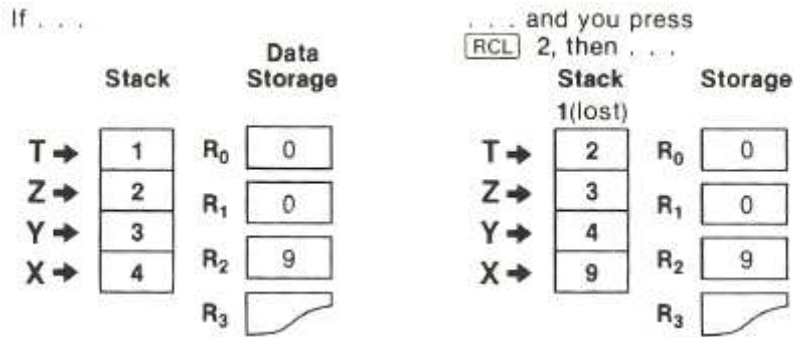
## Data Storage Registers



A copy of the stored number remains in the storage register until a new number is stored there or until the storage registers are cleared or Continuous Memory is reset.

## Recalling Numbers

**[RCL]** (*recall*). When followed by a storage register address (0 through 9), this key places a copy of the number in the specified data storage register into the display (X-register). If the stack is not disabled, executing a **[RCL]** operation causes the stack to lift.

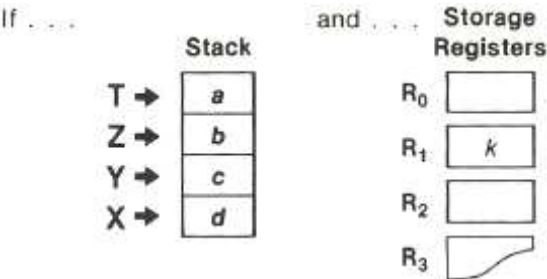


Clearing Data Storage Registers

Pressing **f** **CLEAR** **REG** (*clear registers*) clears the contents of all data storage registers, the stack, and the LAST X register to zero. To clear a single data storage register, store zero in that register.

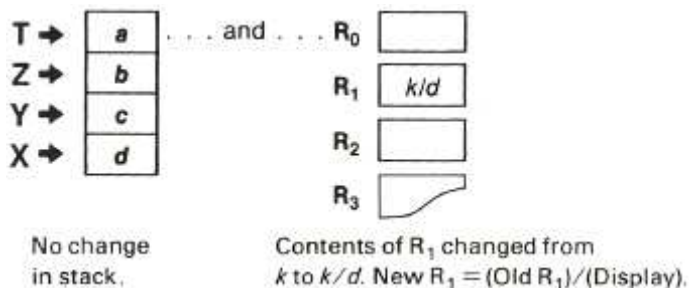
Storage Register Arithmetic

**STO** (**+**, **-**, **×**, **÷**) *n* (*storage register arithmetic*). This sequence uses the number in the display (X-register) to perform arithmetic upon the contents of a specified storage register *n*. The key sequence is **STO** followed by an arithmetic function key, followed in turn by the register address (0 through 9). The result of any storage register arithmetic operation is placed in the specified data storage register.



represent the current status of the memories, then executing **STO** **÷** 1 results in:





## Storage Register Arithmetic Exercises

### Keystrokes

### Display

[f] [FIX] 4

Display shows result of previous calculation.

18 [STO] 0

18.0000

Stores 18 in *R*<sub>0</sub>.

3 [STO] [÷] 0

3.0000

Divides number in *R*<sub>0</sub> (18) by 3.

[RCL] 0

6.0000

Recalls copy of new number in *R*<sub>0</sub>.

4 [STO] [×] 0

4.0000

Multiplies new number in *R*<sub>0</sub> (6.0000) by 4.

[RCL] 0

24.0000

Recalls copy of new number in *R*<sub>0</sub>.

[STO] [+] 0

24.0000

Adds what is in display (24) to number in *R*<sub>0</sub>.

[RCL] 0

48.0000

Recalls copy of new number in *R*<sub>0</sub>.

40 [STO] [-] 0

40.0000

Subtracts 40 from number in *R*<sub>0</sub>.

[RCL] 0

8.0000

Recalls copy of new number in *R*<sub>0</sub>.

## Problems

- Calculate the value of *x* in the following equation

$$x = \frac{8.33(4 - 5.2) + ((8.33 - 7.46) \times 0.32)}{4.3(3.15 - 2.75) - (1.71 \times 2.01)}$$

Answer: 20.9104.

A possible keystroke solution is:

4[ENTER]5.2[-]  
 8.33[x] [f] [LASTx] 7.46  
 [-].32[x] [÷]  
 3.15[ENTER]2.75[-]  
 4.3[x] 1.71[ENTER]  
 2.01[x] [-] [÷]

2. Use constant arithmetic to calculate the remaining balance of a \$1000 loan after six payments of \$100 each and an interest rate of 1% (0.01) per payment period.

Procedure: Load the stack with  $(1 + i)$ , where  $i$  = interest rate, and key in the initial loan balance. Use the following formula to find the new balance after each payment:

$$\text{New Balance} = (\text{Old Balance}) \times (1 + i) - \text{Payment}.$$

The initial key sequence would be:

1.01 [ENTER] [ENTER] [ENTER] 1000

For each payment, continue with:

[x] 100 [-]

Balance after six payments: \$446.32.

3. Store 100 in  $R_5$ . Then:
1. Divide the contents of  $R_5$  by 25.
  2. Subtract 2 from the contents of  $R_5$ .
  3. Multiply the contents of  $R_5$  by 0.75.
  4. Add 1.75 to the contents of  $R_5$ .
  5. Recall the contents of  $R_5$ .

Answer: 3.2500.

## Section 3

# Numeric Functions

Your HP-10C numeric function set enables you to perform a wide range of operations involving number alteration, math, and statistics. Each function is used in the same way for both keyboard and program execution.

## Pi

Pressing  $\boxed{f} \boxed{\pi}$  places the first 10 digits of  $\pi$  (3.141592654) in the display (X-register). If the stack is not disabled, pressing  $\boxed{f} \boxed{\pi}$  causes the stack to lift.

## Number Alteration Functions

In addition to  $\boxed{CHS}$  (*change sign*, refer to page 13) your HP-10C has two functions for altering numbers:  $\boxed{INT}$  and  $\boxed{FRAC}$ .

**Integer Portion.** Pressing  $\boxed{f} \boxed{INT}$  replaces the number in the display (X-register) with its integer portion, that is, replaces all digits to the right of the decimal with zeros.

**Fractional Portion.** Pressing  $\boxed{f} \boxed{FRAC}$  replaces the number in the display (X-register) with its decimal portion, that is, any digits to the left of the decimal are replaced with zeros.

| To Calculate       | Keystroke Example                    | Display              |
|--------------------|--------------------------------------|----------------------|
| Integer portion    | 123.4567<br>$\boxed{f} \boxed{INT}$  | 123.4567<br>123.0000 |
| Fractional portion | 123.4567<br>$\boxed{f} \boxed{FRAC}$ | 123.4567<br>0.4567   |

## One-Number Functions

The one-number math functions on the HP-10C have the following characteristics:

34    Section 3: Numeric Functions

- Use the number in the display (X-register) as the argument for the function.
- Replace the number in the display (X-register) with the result of executing the function.
- Do not affect numbers in the Y-, Z-, and T-registers.

General Functions

**Reciprocal.** Pressing  $\frac{1}{x}$  calculates the reciprocal of the number in the display (X-register), that is, divides 1 by the number in the display.

**Factorial.** Pressing  $f$   $n!$  calculates the factorial value as follows: when n, a positive integer  $\leq 69$  is in the display (X-register),  $n!$  calculates the product of the integers from 1 to n. Also,  $0! = 1$ .

**Square Root.** Pressing  $\sqrt{x}$  calculates the square root of the number in the display (X-register).

**Squaring.** Pressing  $f$   $x^2$  calculates the square of the number in the display (X-register).

| To Calculate | Keystroke Example   | Display           |
|--------------|---------------------|-------------------|
| Reciprocal   | 25<br>$\frac{1}{x}$ | 25.<br>0.0400     |
| Factorial    | 8<br>$f$ $n!$       | 8.<br>40,320.0000 |
| Square Root  | 3.9<br>$\sqrt{x}$   | 3.9<br>1.9748     |
| Square       | 12.3<br>$f$ $x^2$   | 12.3<br>151.2900  |

Trigonometric Operations

The six basic trigonometric functions operate in the trigonometric mode you select.

**Trigonometric Modes.** Selecting a specific trigonometric mode does not convert any number already in the calculator to that mode; it merely tells the calculator what unit of measure (degrees, radians, or grads) to assume a number is expressed in for a trigonometric function.

**Degrees Mode.** Pressing  $f$   $DEG$  selects the Degrees trigonometric mode. No annunciator appears in the display.

**Radians Mode.** Pressing  $\boxed{f} \boxed{RAD}$  selects the Radians trigonometric mode. While Radians mode is set, the **RAD** annunciator appears in the display.

0.0000  
RAD

**Grads Mode.** Pressing  $\boxed{f} \boxed{GRD}$  selects the Grads trigonometric mode. When Grads mode is set, the **GRAD** annunciator appears in the display.

0.0000  
GRAD

The calculator is always set to one of the three trigonometric modes. Continuous Memory maintains the last mode selected, even when the calculator is turned off and on again. If Continuous Memory is reset (refer to page 18), the calculator will automatically reset to Degrees mode.

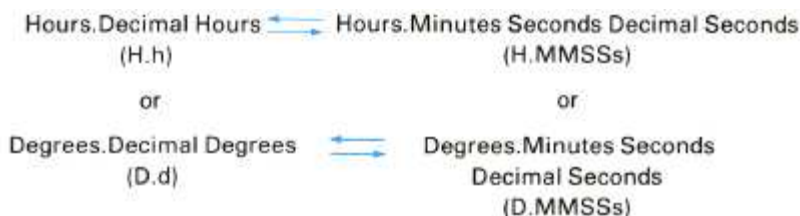
## Trigonometric Functions

| Pressing                       | Calculates      |
|--------------------------------|-----------------|
| $x \boxed{f} \boxed{SIN}$      | sine $x$        |
| $x \boxed{f} \boxed{SIN^{-1}}$ | arc sine $x$    |
| $x \boxed{f} \boxed{COS}$      | cosine $x$      |
| $x \boxed{f} \boxed{COS^{-1}}$ | arc cosine $x$  |
| $x \boxed{f} \boxed{TAN}$      | tangent $x$     |
| $x \boxed{f} \boxed{TAN^{-1}}$ | arc tangent $x$ |

To use any of the trigonometric functions, ensure that the calculator is set to the desired trigonometric mode (Degree, Radian, or Grad), then execute the desired function.

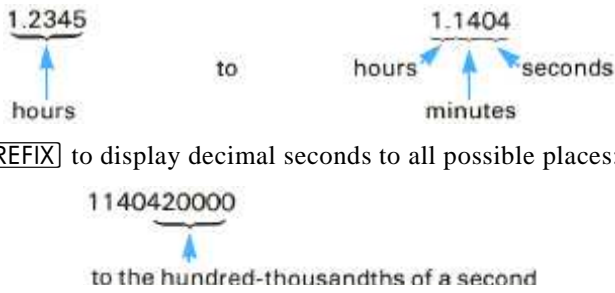
## Time and Angle Conversions

Numbers representing time or angles are interpreted by the HP-10C in a decimal or minutes-seconds format, depending upon the conversion being executed:



**Hours (or Degrees)-Minutes-Seconds Conversion.** Pressing  $\boxed{f} \boxed{\rightarrow H.MS}$  converts the number in the display (X-register) from a decimal hours (or decimal degrees) format to an hours (or degrees)-minutes-seconds-decimal seconds format.

For example, press  $\boxed{f} \boxed{\rightarrow H.MS}$  to convert



Press **[f] [PREFIX]** to display decimal seconds to all possible places:



**Decimal Hours (or Degrees) Conversion.** Pressing **[f] [↔H]** converts the number in the display (X-register) from an hours (or degrees)-minutes-seconds-decimal seconds format to a decimal hour (or degrees) format.

Degrees/Radians Conversions

The **[→DEG]** and **[→RAD]** functions are used to convert angles between decimal degrees and radians (D.d → R.r and R.r → D.d).

| To Convert                 | Example Keystrokes        | Display        |
|----------------------------|---------------------------|----------------|
| Decimal Degrees to Radians | 40.5<br><b>[f] [→RAD]</b> | 40.5<br>0.7069 |
| Radians to Decimal Degrees | <b>[f] [→DEG]</b>         | 40.5000        |

Logarithmic Functions

**Natural Logarithm.** Pressing **[f] [LN]** calculates the natural logarithm of the number in the display (X-register), that is, the logarithm to the base e (2.718281828) of the number in the X-register.

**Natural Antilogarithm.** Pressing **[e<sup>x</sup>]** calculates the natural antilogarithm of the number in the display (X-register), that is, raises e (2.718281828) to the power of the number in the X-register.

**Common Logarithm.** Pressing **[f] [LOG]** calculates the common logarithm of the number in the display (X-register), that is, the logarithm to the base 10.

**Common Antilogarithm.** Pressing **[10<sup>x</sup>]** calculates the common antilogarithm of the number in the display (X-register), that is, raises 10 to the power of that number.

| To Calculate    | Example Keystrokes           | Display              |
|-----------------|------------------------------|----------------------|
| Natural Log     | 45<br>[f] [LN]               | 45.<br>3.8067        |
| Natural Antilog | 3.4012<br>[e <sup>x</sup> ]  | 3.4012<br>30.0001    |
| Common Log      | 12.4578<br>[f] [LOG]         | 12.4578<br>1.0954    |
| Common Antilog  | 3.1354<br>[10 <sup>x</sup> ] | 3.1354<br>1,365.8405 |

## Two-Number Functions

Your HP-10C two-number math functions use the values in the display (X-register) and in the Y-register to calculate a result. To use any of these functions, key in the Y-register value first, press [ENTER] to lift the value into the Y-register, key in the displayed X-register value, then execute the function.

### Percentages

To find a specified percentage of a number

1. Key in the base number.
2. Press [ENTER].
3. Key in the percent rate.
4. Press [%].
- (5. To add that percentage to the base number, press [+].)

For example, to find the sales tax (at 3%) and total cost of a \$15 item:

| Keystrokes | Display |                                   |
|------------|---------|-----------------------------------|
| 15[ENTER]  | 15.0000 | Enters the base number (price).   |
| 3[%]       | 0.4500  | Calculates 3% of \$15 (45 cents). |
| [+]        | 15.4500 | Total cost of item is \$15.45.    |

The Power Function

Pressing  $y^x$  calculates a power of a number—that is,  $y^x$ . The y-value is keyed in before the x-value.

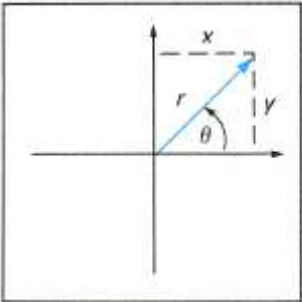
- 1. Key in the base number, which is designated by the y on the key.
- 2. Press  $\text{ENTER}$  to separate the second number (the exponent) from the first number (the base).
- 3. Key in the exponent, which is designated by the x on the key.
- 4. Press  $y^x$  to calculate the power.

| To Calculate               | Keystrokes                              | Display |
|----------------------------|---|---------|
| $2^{1.4}$                  | 2 $\text{ENTER}$ 1.4 $y^x$              | 2.6390  |
| $2^{-1.4}$                 | 2 $\text{ENTER}$ 1.4 $\text{CHS}$ $y^x$ | 0.3789  |
| $(-2)^3$                   | 2 $\text{CHS}$ $\text{ENTER}$ 3 $y^x$   | -8.0000 |
| $\sqrt[3]{2}$ or $2^{1/3}$ | 2 $\text{ENTER}$ 3 $1/x$ $y^x$          | 1.2599  |

Polar/Rectangular Coordinate Conversions

Two functions ( $\text{P}$ ,  $\text{R}$ ) are provided in your HP-10C for polar/ rectangular coordinate conversions.

The angle  $\theta$  is assumed to be in decimal degrees, radians, or grads, depending upon which trigonometric mode (Degrees, Radians, or Grads) the calculator is set to. Angle  $\theta$  is measured as shown in the illustration to the right. The answer returned for  $\theta$  is between  $180^\circ$  and  $-180^\circ$ .



**Polar Conversion.** Pressing  $f \text{P}$  (*polar*) converts values in the X- and Y-registers representing rectangular coordinates ( $x$ ,  $y$ ) to polar coordinates (magnitude  $r$ , angle  $\theta$ ). First displayed is  $r$ ; press  $x \leftrightarrow y$  (*x exchange y*) to display  $\theta$ . (For a discussion of memory registers, refer to section 2.)

**Rectangular Conversion.** Pressing  $f \text{R}$  (*rectangular*) converts values in the X- and Y-registers representing polar coordinates (magnitude  $r$ , angle  $\theta$ ) to rectangular coordinates ( $x$ ,  $y$ ). First displayed is  $x$ ; press  $x \leftrightarrow y$  (*x exchange y*) to display  $y$ .



| To Convert                        | Example Keystrokes         | Display |
|-----------------------------------|----------------------------|---------|
| Rectangular coordinates to polar: |                            |         |
| $y$                               | 5 <b>ENTER</b>             | 5.0000  |
| $x$                               | 10                         | 10.     |
| $r$                               | <b>f</b> <b>→P</b>         | 11.1803 |
| $\theta$                          | <b>x<math>\geq</math>y</b> | 26.5651 |
| Polar coordinates to rectangular: |                            |         |
| $\theta$                          | 30 <b>ENTER</b>            | 30.0000 |
| $r$                               | 12                         | 12.     |
| $x$                               | <b>f</b> <b>→R</b>         | 10.3923 |
| $y$                               | <b>x<math>\geq</math>y</b> | 6.0000  |

## Statistics Functions

### Accumulating Statistics

The HP-10C can perform one- or two-variable statistical calculations. The data are entered into the calculator using the  **$\Sigma+$**  key, which automatically calculates and stores statistics of the data in storage registers  $R_0$  through  $R_5$ . (These registers are therefore referred to as the *statistics registers*.)

Before beginning to accumulate statistics for a new set of data, you should clear the statistics registers by pressing **f** **CLEAR** **REG**.\*

In one-variable statistical calculations, to enter each data point—referred to as an  $x$ -value—key the  $x$ -value into the display, then press  **$\Sigma+$** .

In two-variable statistical calculations, to enter each data pair—referred to as the  $x$ - and  $y$ -values;

1. Key the  $y$ -value into the display first.
2. Press **ENTER**.
3. Key the  $x$ -value into the display.
4. Press  **$\Sigma+$** .

---

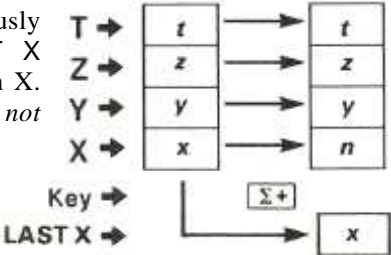
\* This also clears  $R_6$  through  $R_9$ , the stack registers, and the display. However, it will not clear any of registers  $R_0$  through  $R_9$  that are currently converted to program lines. If the statistics registers have been converted to program lines, **Error 3** will be displayed when you try to use  **$\Sigma+$** . The **MEM** function (page 62) can be used to determine memory status.

The data are compiled as follows:

| Register       | Contents     |   |
|----------------|--------------|---|
| R <sub>0</sub> | $n$          | Number of data points (pairs) accumulated ( $n$ also appears in the display). |
| R <sub>1</sub> | $\Sigma x$   | Summation of $x$ -values.   |
| R <sub>2</sub> | $\Sigma x^2$ | Summation of squares of $x$ -values.  |
| R <sub>3</sub> | $\Sigma y$   | Summation of $y$ -values.   |
| R <sub>4</sub> | $\Sigma y^2$ | Summation of squares of $y$ -values.  |
| R <sub>5</sub> | $\Sigma xy$  | Summation of products of $x$ - and $y$ -values.                               |

When you execute  $\Sigma+$ , the number previously in the X-register is placed in the LAST X register and the updated  $n$ -value is placed in X. *The number previously in the Y-register is not changed.*

You can recall any of the accumulated statistics to the display (X-register) by pressing  $\text{RCL}$  and the number of the data storage register containing the desired statistic.



**Example.** Electrical energy researcher Helen I. Voltz suspects a possible relationship between the rise in worldwide coal production in the years 1972 through 1976 and a similar rise in worldwide electricity output for the same period. To assist in a study of the data, Voltz will use her HP-10C to accumulate the coal production and electrical output statistics. Find  $\Sigma x$ ,  $\Sigma x^2$ ,  $\Sigma y$ ,  $\Sigma y^2$ , and  $\Sigma xy$  for the paired  $x$ - and  $y$ -values of Voltz’s data.\*



**\*Note:** Some sets of data points consist of a series of  $x$ -values (or  $y$ -values) that differ from each other by a comparatively small amount. You can maximize the precision of any statistical calculation involving such data by keying in only the differences between each value and a number approximating the average of the values. This number must be added to the result of calculating  $\bar{x}$ ,  $\hat{y}$ ,  $\hat{x}$  or the  $y$ -intercept of  $\text{[L.R.]}$ . For example, if your  $x$ -values consist of 665999, 666000, and 666001, you should enter the data as  $-1$ ,  $0$ , and  $1$ . If afterwards you calculate  $\bar{x}$ , add 666000 to the answer. In some cases the calculator cannot compute  $s$ ,  $r$ ,  $\text{[L.R.]}$ ,  $\hat{y}$ , or  $\hat{x}$  with data values that are too close to each other; and if you attempt to do so, the calculator will display **Error 2**. This will not happen, however, if you normalize the data as described above.

| Year   | 1972  | 1973  | 1974  | 1975  | 1976  |
|--|-------|-------|-------|-------|-------|
| Coal Production (y)<br>(billions of metric tons)       | 1.761 | 1.775 | 1.792 | 1.884 | 1.943 |
| Electricity Output (x)<br>(billions of megawatt-hours) | 5.552 | 5.963 | 6.135 | 6.313 | 6.713 |

| Keystrokes                      | Display |  |
|---------------------------------|---------|--|
| $\boxed{f}$ CLEAR $\boxed{REG}$ | 0.0000  | Clears data storage registers ( $R_0$ through $R_9$ and stack).                |
| $\boxed{f}$ $\boxed{FIX}$ 3     | 0.000   | Limits display to correspond to the significant figures of data.               |
| 1.761 $\boxed{ENTER}$           | 1.761   |  |
| 5.552 $\boxed{\Sigma+}$         | 1.000   | 1972 data.   |
| 1.775 $\boxed{ENTER}$           | 1.775   |  |
| 5.963 $\boxed{\Sigma+}$         | 2.000   | 1973 data.   |
| 1.792 $\boxed{ENTER}$           | 1.792   |  |
| 6.135 $\boxed{\Sigma+}$         | 3.000   | 1974 data.   |
| 1.884 $\boxed{ENTER}$           | 1.884   |  |
| 6.313 $\boxed{\Sigma+}$         | 3.000   | 1975 data.   |
| 1.943 $\boxed{ENTER}$           | 1.943   |  |
| 6.713 $\boxed{\Sigma+}$         | 4.000   | 1976 data.   |
| $\boxed{RCL}$ 1                 | 30.676  | Sum of $x$ -values ( $\Sigma x$ ) from register $R_1$ .                        |
| $\boxed{RCL}$ 2                 | 188.939 | Sum of squares of $x$ -values ( $\Sigma x^2$ ) from register $R_2$ .           |
| $\boxed{RCL}$ 3                 | 9.155   | Sum of $y$ -values ( $\Sigma y$ ) from register $R_3$ .                        |
| $\boxed{RCL}$ 4                 | 16.788  | Sum of squares of $y$ -values ( $\Sigma y^2$ ) from register $R_4$ .           |
| $\boxed{RCL}$ 5                 | 56.292  | Sum of products of $x$ - and $y$ -values ( $\Sigma xy$ ) from register $R_5$ . |

**Note:** Unlike storage register arithmetic, the  $\boxed{\Sigma+}$  and  $\boxed{\Sigma-}$  operations allow overflow to occur in storage registers  $R_0$  through  $R_5$  without indicating **Error 1** in the display.

## Correcting Accumulated Statistics

If you discover that you have entered data incorrectly, the accumulated statistics can be easily corrected. If one value of an  $(x, y)$  data pair is incorrect, you must delete and re-enter both values.

1. Key the incorrect data pair into the X- and Y-registers.
2. Press  $\boxed{f} \boxed{\Sigma-}$  to delete the incorrect data.\*
3. Key in the correct values for  $x$  and  $y$ .
4. Press  $\boxed{\Sigma+}$ .

Alternatively, if the incorrect data point or pair is the most recent one entered and  $\boxed{\Sigma+}$  has been pressed, you can press  $\boxed{f} \boxed{\text{LAST}x} \boxed{f} \boxed{\Sigma-}$  to remove the incorrect data.

**Example:** After keying in the preceding data, Voltz found new information indicating that the coal output for the last data pair should have been 1.946 instead of 1.943. Use  $\boxed{\Sigma-}$  to remove the statistical data that were accumulated as a result of using the older, incorrect data pair; then key in the correct data pair.

| Keystrokes                   | Display |   |
|------------------------------|---------|---|
| 1.943 $\boxed{\text{ENTER}}$ | 1.943   | Keys in the data pair we want to replace and deletes the pair's unwanted statistics. Number of pair entries then drops to four. |
| 6.713 $\boxed{\Sigma-}$      | 4.000   |   |
| 1.946 $\boxed{\text{ENTER}}$ | 1.946   | Keys in and accumulates the replacement data pair. Number of pairs accumulated is again five.                                   |
| 6.713 $\boxed{\Sigma+}$      | 5.000   |   |

Retain these statistics in your calculator for use in the following examples.

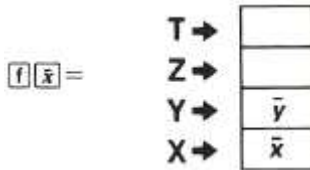
---

\*Note: Although  $\boxed{f} \boxed{\Sigma-}$  can be used to delete an erroneous  $(x, y)$  pair, it will not delete any rounding errors that may have occurred when the statistics of that pair were added into registers  $R_1$  through  $R_5$ . Consequently, subsequent results may be different than they would have been if the erroneous pair had not been entered with  $\boxed{\Sigma+}$ . However, the difference will not be serious unless the erroneous pair has a magnitude that is enormous compared with the correct pair; and in such a case it may be wise to start over again and re-enter the data carefully.

## Mean

The  $\bar{x}$  function computes the arithmetic mean (*average*) of the  $x$ - and  $y$ -values using the accumulated statistics in registers  $R_1$  and  $R_3$ , respectively. When you press  $\boxed{f} \boxed{\bar{x}}$ :

- The contents of the stack registers lift in the same way as if you keyed in two numbers in sequence. (The mean of  $x$  is simultaneously copied into the X-register as the mean of  $y$  is copied into the Y-register.)



- The mean of the  $x$ -values ( $\bar{x}$ ) is calculated using the statistics accumulated in  $R_1$  ( $\sum x$ ) and  $R_0$  ( $n$ ). The mean of the  $y$ -values ( $\bar{y}$ ) is calculated using the data accumulated in registers  $R_3$  ( $\sum y$ ) and  $R_0$  ( $n$ ). The formulas used are shown below:

$$\bar{x} = \frac{\sum x}{n}$$

$$\bar{y} = \frac{\sum y}{n}$$

**Example:** From the 5-year statistical data you accumulated (and corrected) in the preceding examples, calculate the average coal production and electrical output for the entire period.

### Keystrokes

### Display

$\boxed{f} \boxed{\bar{x}}$

6.135

Average electrical output (average of X-register inputs) for the 5-year period.

$\boxed{x} \boxed{\bar{y}}$

1.832

Average coal production (average Y-register inputs) for the 5-year period.

Retain these statistics in the calculator for use in the next example.

## Standard Deviation

Pressing  $\boxed{f} \boxed{s}$  computes the *standard deviation* (a measure of dispersion around the mean) of the accumulated statistical data. The formulas used by the

HP-10C to compute  $s_x$ , the standard deviation of the accumulated  $x$ -values, and  $s_y$ , the standard deviation of the accumulated  $y$ -values are:

$$s_x = \sqrt{\frac{n\sum x^2 - (\sum x)^2}{n(n-1)}} \qquad s_y = \sqrt{\frac{n\sum y^2 - (\sum y)^2}{n(n-1)}}$$

These formulas give the best estimates of the population standard deviations from the sample data. Consequently, the standard deviation given by these formulas is termed by convention the *sample* standard deviation. When you press f S:

1. The contents of the stack registers are lifted as they are when you press f ⌈x̄, as described on page 43.
2. The standard deviation of the  $x$ -values ( $s_x$ ) is calculated using the data accumulated in registers  $R_2$  ( $\sum x^2$ ),  $R_1$  ( $\sum x$ ), and  $R_0$  ( $n$ ) according to the formula shown above. The resultant value for  $s_x$  is placed in the X-register.
3. The standard deviation of the  $y$ -values ( $s_y$ ) is calculated using the statistical data accumulated in registers  $R_4$  ( $\sum y^2$ ),  $R_3$  ( $\sum y$ ), and  $R_0$  ( $n$ ) according to the formula shown above. The resultant value for  $s_y$  is available in the Y-register.

**Example:** Calculate the standard deviation for the corrected coal production and for the electrical output accumulations used in the preceding examples.

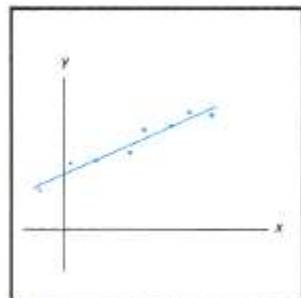
| Keystrokes                       | Display |   |
|----------------------------------|---------|---|
| <span>f</span> <span>S</span>    | 0.429   | Standard deviation of electrical output (X-register inputs) for the five-year period. |
| <span>⌈x̄</span> <span>⌈y</span> | 0.080   | Standard deviation of coal production (Y-register inputs) for the five-year period.   |

Retain the preceding statistics in your HP-10C for use in the next example.

When your data constitute not just a sample of a population but rather *all* of the population, the standard deviation of the data is the *true* population standard deviation (denoted  $\sigma$ ). The formula for the true population standard deviation differs by a factor of  $\sqrt{(n-1)n}$  from the formula used for the S function. For large  $n$ , the difference between the values is small, and for most applications can be ignored. Nevertheless, if you want to calculate the exact value of the population standard deviation for an entire population, you can easily do so: simply add, using the Σ+ key, the mean ( $\bar{x}$ ) of the data to the data and press f S. The result will be the true population standard deviation of the original data.

## Linear Regression

Linear regression is a statistical method for finding a straight line that best fits a set of two or more data pairs, thus providing a relationship between two variables. After the statistics of a group of data pairs have been accumulated in registers  $R_0$  through  $R_5$ , you can calculate the coefficients in the linear equation  $y = Ax + B$  using the least squares method by pressing  $\boxed{f} \boxed{L.R.}$ .



To use the linear regression function on your HP-10C, use the  $\boxed{\Sigma+}$  key to accumulate the statistics of a series of two or more data pairs. Then execute  $\boxed{L.R.}$ . When you press  $\boxed{f} \boxed{L.R.}$ :

1. The contents of the stack registers are lifted as they are when you press  $\boxed{f} \boxed{\bar{x}}$ , as described on page 43.
2. The slope ( $A$ ) and the  $y$ -intercept ( $B$ ) of the least squares line of the data are calculated using the equations:

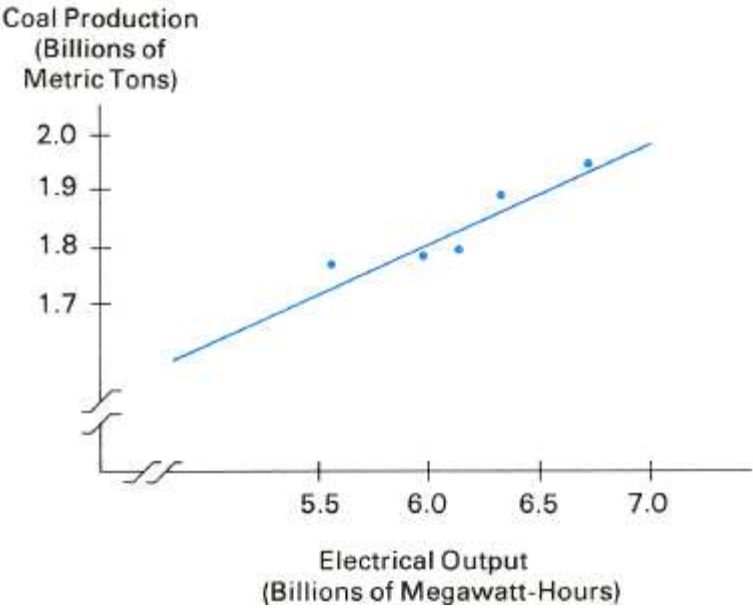
$$A = \frac{n\Sigma xy - \Sigma x \Sigma y}{n\Sigma x^2 - (\Sigma x)^2} \quad B = \frac{\Sigma y \Sigma x^2 - \Sigma x \Sigma xy}{n\Sigma x^2 - (\Sigma x)^2}$$

The slope  $A$  is placed in the  $Y$ -register; the  $y$ -intercept,  $B$ , is placed in display ( $X$ -register).

|                 |     |                |
|-----------------|-----|----------------|
| $T \rightarrow$ | $t$ |                |
| $Z \rightarrow$ | $z$ |                |
| $Y \rightarrow$ | $A$ | slope          |
| $X \rightarrow$ | $B$ | $y$ -intercept |

**Example:** Calculate the  $y$ -intercept and slope of Voltz's corrected data.

Solution: Voltz *could* draw a plot of coal production against electrical output like the one shown below. However, with her HP-10C, Voltz has only to accumulate the statistics (as we have already done) using the  $\boxed{\Sigma+}$  key, then press  $\boxed{f} \boxed{L.R.}$ .



| Keystrokes                                   | Display |                          |
|--|---------|--------------------------|
| <code>f</code> <code>L.R.</code>             | 0.777   | y-intercept of the line. |
| <code>x</code> <code>↔</code> <code>y</code> | 0.172   | Slope of the line.       |

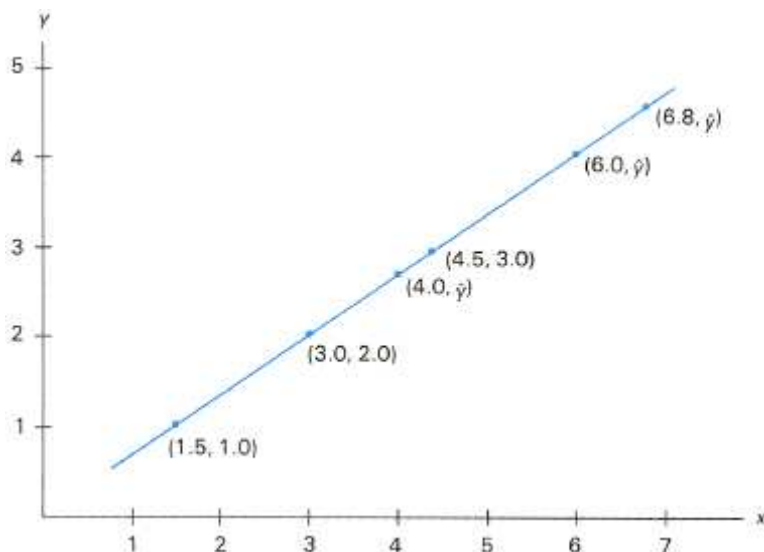
Retain these statistics in your calculator for use in the next example.

Linear Estimation and Correlation Coefficient

When you execute the `ŷ,r` or `ẋ,r` function, the *linear estimate* ( $\hat{y}$  or  $\hat{x}$ ) is placed in the X-register, and the *correlation coefficient* ( $r$ ) is placed in the Y-register.

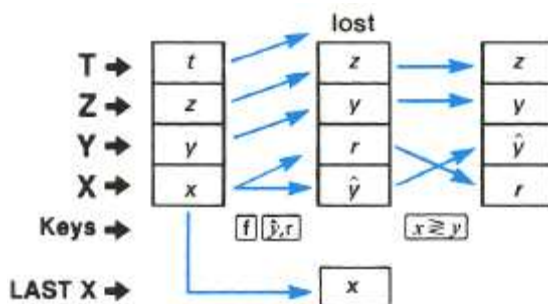
**Linear Estimation.** With statistics accumulated in registers  $R_0$  through  $R_5$ , a predicted value for  $y$  (denoted  $\hat{y}$ ) can be calculated by keying in a known value for  $x$  and pressing `f` `ŷ,r`. Similarly, a predicted value for  $x$  (denoted  $\hat{x}$ ) can be calculated by keying in a known value for  $y$  and pressing `f` `ẋ,r`.





**Correlation Coefficient.** Both linear regression and linear estimation presume that the relationship between the  $x$ - and  $y$ -data values can be approximated, to some degree, by a linear function (that is, a straight line). The correlation coefficient ( $r$ ) is a determination of how closely your data fit a straight line. The correlation coefficient can range from  $r = +1$  to  $r = -1$ . At  $r = +1$ , the data fall exactly onto a straight line with positive slope. At  $r = -1$ , the data fall exactly onto a straight line with negative slope. At  $r = 0$ , the data cannot be approximated at all by a straight line. With statistics accumulated in registers  $R_0$  through  $R_5$ , the correlation coefficient  $r$  is calculated by pressing  $\boxed{f} \boxed{\hat{y}, r}$  (or  $\boxed{f} \boxed{\hat{x}, r}$ ).

The number that appears in the display will be a  $\hat{y}$ -value (or  $\hat{x}$ -value) (which is meaningless if you did not key in a specific  $x$ -value (or  $y$ -value), as described above). To view the correlation coefficient value ( $r$ ), exchange the contents of the X- and Y-registers by pressing  $\boxed{x \leftrightarrow y}$ .



**Example.** Using the statistics saved from the previous example, if Voltz wishes to predict coal production ( $y$ ) for 1977, she keys in an estimate of electrical

production (a “known”  $x$ -value) for 1977 and presses  $\boxed{f}\boxed{\hat{y},r}$ . Because the correlation coefficient for Voltz's data is automatically included in the calculation, she can view how closely her data it a straight line by simply pressing  $\boxed{x\lessgtr y}$  after the  $\hat{y}$  prediction appears in the display.

| Keystrokes                   | Display |  |
|------------------------------|---------|--|
| 7.142                        | 7.142   | Voltz’s estimate of 1977 electrical output.            |
| $\boxed{f}\boxed{\hat{y},r}$ | 2.005   | Predicted coal production for 1977.                    |
| $\boxed{x\lessgtr y}$        | 0.921   | The original data closely approximate a straight line. |

Conversely, suppose Voltz wishes to estimate the electrical output should there be a future drop in coal production to 1.800 billion metric tons.

| Keystrokes                   | Display |   |
|------------------------------|---------|---|
| 1.8                          | 1.8     | Hypothetical future coal production.                      |
| $\boxed{f}\boxed{\hat{x},r}$ | 5.951   | Predicted electrical output for that year.                |
| $\boxed{x\lessgtr y}$        | 0.921   | Again, the correlation coefficient for the original data. |

## Section 4

# Display Control

Owing to Continuous Memory, when you turn on your HP-10C, the display setting will be the same as it was before you last turned off the calculator. Regardless of the display options in effect, the HP-10C always internally represents each number as a 10-digit mantissa and a two-digit exponent of 10. Thus when the calculator is set to display only four digits past the decimal point, the fixed constant  $\pi$  is always represented internally as  $3.141592654 \times 10^{00}$ .



## Display Mode Control

Your HP-10C has three display modes—**FIX**, **SCI**, and **ENG**—that use a specified constant (0 through 9) to specify display setting. The illustration below shows how the number 123,456 would be displayed by a four-digit setting in each of the three types of modes.

| Keystrokes            | Display      |
|-----------------------|--------------|
| <b>f</b> <b>FIX</b> 4 | 123.456.0000 |
| <b>f</b> <b>SCI</b> 4 | 1.2346 05    |
| <b>f</b> <b>ENG</b> 4 | 123.46 03    |
| <b>f</b> <b>FIX</b> 4 | 123,456.0000 |

## Fixed Decimal Display

**FIX** (*fixed decimal*) displays numbers using a fixed decimal mode without exponents. In any **FIX** display setting, the calculator will automatically switch to **SCI** mode to allow viewing of a displayed number that is too large or too small to be viewed in the current **FIX** mode.



Fixed decimal display is selected or modified by pressing **[f] [FIX]** followed by the appropriate number key to specify the number of decimal places (0 to 9) you want the display rounded to.

| Keystrokes                  | Display    |   |
|-----------------------------|------------|---|
| 123.4567895 $\square$ ENTER | 123.4568   | Display is rounded to four decimal places. However, internally the number is maintained in its original value to 10 digits. |
| $\square$ f $\square$ FIX 6 | 123.456790 | The display is rounded upward if the first undisplayed digit is 5 or greater.   |
| $\square$ f $\square$ FIX 0 | 123.       |   |
| $\square$ f $\square$ FIX 4 | 123.4568   | Usual $\square$ FIX 4 display.  |

Scientific Notation Display

**[SCI]** (*scientific*) displays numbers in scientific notation mode. To select or modify a **[SCI]** mode, press **[f] [SCI]** followed by the number key (0 through 6) that specifies the number of decimal places you want the display rounded to.



| Keystrokes                  | Display        |  |
|-----------------------------|----------------|--|
| 123.4567895 $\square$ ENTER | 123.4568       | Display is rounded to four decimal places.   |
| $\square$ f $\square$ SCI 2 | 1.23      02   | $1.23 \times 10^2$ ; display rounded down.   |
| $\square$ f $\square$ SCI 4 | 1.2346    02   | $1.2346 \times 10^2$ ; display rounded up.   |
| $\square$ f $\square$ SCI 6 | 1.234568    02 | $1.234568 \times 10^2$ ; display rounded up. |

As indicated in the above examples, display rounding occurs on numbers with more decimal places than the number specified by your mode setting. In **[SCI]** mode, specifying seven or more digits to the right of the decimal point (**[SCI]**7, 8, or 9) will move rounding to the *internally held* decimal place that cannot be displayed.\*

| Keystrokes                | Display     |   |
|---------------------------|-------------|---|
| <b>[f]</b> <b>[SCI]</b> 7 | 1.234567 02 | Rounding occurs at seventh decimal digit; display cannot show seventh decimal digit in <b>[SCI]</b> mode, so no rounding occurs in the display. |
| <b>[f]</b> <b>[SCI]</b> 8 | 1.234567 02 | Rounds to eighth decimal digit. No change in displayed decimal digits.  |
| <b>[f]</b> <b>[SCI]</b> 9 | 1.234567 02 | Rounds to ninth decimal digit. No change in displayed decimal digits.   |

## Engineering Notation Display

**[ENG]** (*engineering*) displays numbers in an engineering notation format which operates the same as **[SCI]** notation format except:

- In engineering notation, the first significant digit is always present in the display. The number key you press after **[f]** **[ENG]** specifies the number of *additional* digits to which you want to round the display.
- Engineering notation shows all exponents in multiples of three.

| Keystrokes | Display  |
|------------|----------|
| .012345    | 0.012345 |

---

\*If one or more trailing 9's exist internally following the last digit allowed in the display setting, rounding may be propagated in the displayed digits for **[SCI]**7 and 8 display settings. For example, 1.00000094 in **[SCI]**7 will not cause rounding in the displayed version of the number, but 1.00000095 (...95 to...99) in **[SCI]**7 will cause rounding in the displayed digits.

| Keystrokes            | Display  |     |
|-----------------------|--|-----|
| <b>f</b> <b>ENG</b> 1 | 12.  | -03 |
|                       | Engineering notation display. Display is rounded to one significant digit after the leading digit. Power of 10 is multiple of three. |     |
| <b>f</b> <b>ENG</b> 3 | 12.35  | -03 |
|                       | Display is rounded to third significant digit after the leading digit.   |     |
| <b>f</b> <b>ENG</b> 2 | 12.3   | -03 |
|                       | Display changed to <b>ENG</b> 2 format.  |     |
| 10 <b>x</b>           | 123  | -03 |
|                       | Decimal shifts to maintain multiple of three in exponent.  |     |
| <b>f</b> <b>FIX</b> 4 | 0.1235   |     |
|                       | Usual <b>FIX</b> 4 format.   |     |

## Mantissa Display

All numbers held in the calculator’s stack and data storage registers are represented internally as 10-digit mantissas with two-digit exponents. When you want to view the full 10-digit mantissa of a number held in the X-register, press **f** **CLEAR** **PREFIX** and hold the **PREFIX** key. The mantissa of the currently displayed number will appear and remain in the display until you release the **PREFIX** key.

| Keystrokes                                    | Display    |
|---|------------|
| <b>f</b> $\pi$                                | 3.1416     |
| <b>f</b> <b>CLEAR</b> <b>PREFIX</b><br>(hold) | 3141592654 |

## Rounding at the Tenth Digit

As mentioned earlier, your HP-10C holds every value to 10 digits internally, regardless of the number of places specified in the current **FIX**, **SCI**, or **ENG** display setting. The final result of every calculation or series of calculations is rounded to the tenth digit. For example,  $\pi$  and  $2/3$  have nonterminating decimal representations (3.1415926535 ... and 0.6666666666 ...). Because the HP-10C can provide only a finite approximation of such numbers (10 digits), a small error due to rounding can occur in the tenth digit. This error can be increased through lengthy calculations, but in the majority of

cases it does not enter the range of significant digits. To accurately assess the effects of rounding error for a given calculation requires the use of numerical analysis methods that are beyond the scope of this handbook.





The image shows the front cover of a book. The cover is a solid, vibrant blue color. It has rounded corners at the top and bottom. The text is centered on the cover in a white, serif font. The text is arranged in two lines: "Part II" on the top line and "Programming" on the bottom line. The entire cover is framed by a thin white border.

## **Part II**

# **Programming**

## Section 5

# Programming Basics

## Why Use Programs?

A program is simply a sequence of keystrokes that is stored in the calculator. Whenever you have to calculate with the same sequence of keystrokes several times, you can save a great deal of time by incorporating these keystrokes into a program. Instead of pressing all the keys each time, you press just one key to start the program: the calculator does the rest automatically! No prior programming experience is necessary to learn HP-10C programming.

## Creating a Program

Creating a program consists simply of *writing* the program, then *storing* it:

1. Write down the sequence of keystrokes that you would use to calculate the quantity or quantities desired.
2. Press **[P/R]** to set the calculator to *Program mode*. When the calculator is in Program mode, functions are not executed when the keys are pressed. Instead, the keystrokes are stored inside the calculator. The **PRGM** status indicator in the display is lit when the calculator is in Program mode.
3. Press **[f] CLEAR [PRGM]** to erase any previous programs that may be stored inside the calculator. If you want to create a new program without erasing a program already stored, skip this step (refer to section 8, Multiple Programs).
4. Key in the sequence of keystrokes that you wrote down in step 1. Skip the beginning keystrokes that enter data which would differ each time the program is used.

**Example:** Lerner Student has noticed that all the physiology literature refers to temperature in degrees Celsius. This is confusing to Lerner who has grown up using degrees Fahrenheit in everyday life. Write a program that converts degrees Celsius to degrees Fahrenheit for Lerner.

The formula is:  $^{\circ}\text{F} = (^{\circ}\text{C} \times 1.8) + 32$



First we'll manually calculate the degrees Fahrenheit of a 24°C room:

| Keystrokes   | Display |  |
|--------------|---------|--|
| 24           | 24.     | Keys in room temperature in degrees Celsius.           |
| <b>ENTER</b> | 24.0000 | Separates temperature from factor to be keyed in next. |
| 1.8          | 1.8     |  |
| <b>×</b>     | 43.2000 |  |
| 32           | 32.     |  |
| <b>+</b>     | 75.2000 | Room temperature in degrees Fahrenheit.                |

Next, set the calculator to Program mode and erase any program(s) already stored:

| Keystrokes                        | Display |                                  |
|-----------------------------------|---------|----------------------------------|
| <b>P/R</b>                        | 00-     | Sets calculator to Program mode. |
| <b>f</b> <b>CLEAR</b> <b>PRGM</b> | 00-     | Clears program(s).               |

Finally, press the keys that we used above to solve the problem manually. Do not key in 24; this number will vary each time the program is used. Don't be concerned right now about what appears in the display as you press the keys; we'll discuss that later in this section.

| Keystrokes   | Display |    |
|--------------|---------|----|
| <b>ENTER</b> | 01-     | 36 |
| 1            | 02-     | 1  |
| <b>.</b>     | 03-     | 48 |
| 8            | 04-     | 8  |
| <b>×</b>     | 05-     | 20 |
| 3            | 06-     | 3  |
| 2            | 07-     | 2  |
| <b>+</b>     | 08-     | 40 |

## Running a Program

To run (or “execute”) a program:

- 1. Press **[P/R]** to set the calculator back to Run mode. This also sets the program back to line 00. If the calculator is already in Run mode (that is, the **PRGM** status indicator in the display is not lit), skip this step.
- 2. Key any required data into the calculator, just as if you were calculating manually. When a program is run, it uses the data already keyed into the display and the registers inside the calculator.
- 3. Press **[R/S]** (*run/stop*) to begin program execution. During execution, **running** will flash in the display.

**Example:** Run the program created above to calculate the temperature in degrees Fahrenheit of a water bath at 35.7°C and a refrigerator at 4.3°C.

| Keystrokes   | Display |  |
|--------------|---------|--|
| <b>[P/R]</b> |         | Sets calculator to Run mode. Display shows number previously calculated. |
| 35.7         | 35.7    | Water bath temperature in degrees Celsius.                               |
| <b>[R/S]</b> | 96.2600 | Temperature in degrees Fahrenheit.                                       |
| 4.3          | 4.3     | Refrigerator temperature in degrees Celsius.                             |
| <b>[R/S]</b> | 39.7400 | Temperature in degrees Fahrenheit.                                       |

That's all there is to creating and running simple programs! But if you want to use programs frequently, you'll want to know more about programming—such as how to check what keystrokes are stored in program memory, how *many* keystrokes can be stored in program memory, how to correct or otherwise modify programs, how to skip keystrokes when running a program, and so on. Before you can understand these aspects of programming, we need to briefly discuss how keystrokes are treated by the calculator when they are stored in Program mode and when they are executed in Run mode.

## Program Memory

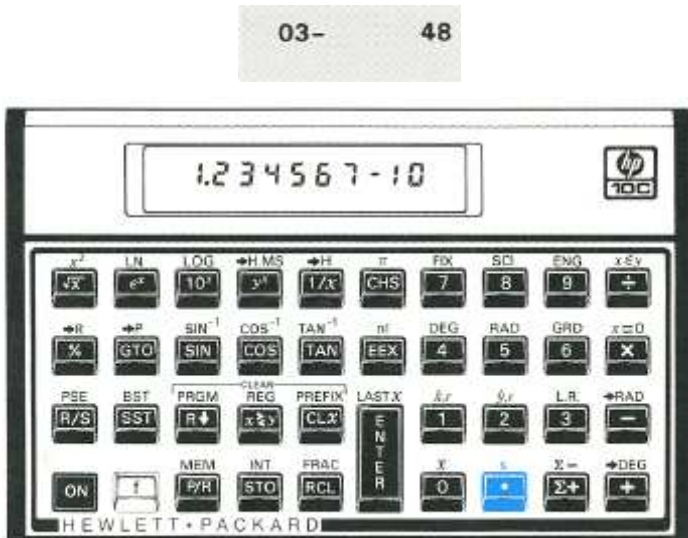
Keystrokes entered into the calculator in Program mode are stored in *program memory*. Each digit, decimal point, or function key is called an instruction and is stored in one line of program memory—usually referred to simply as a *program line*. Keystroke sequences beginning with the **[f]**, **[STO]**, and **[GTO]** prefix keys are considered to comprise a *complete instruction* and are stored in only one program line.

When a program is run, each instruction in program memory is executed—that is, the keystroke in that program line is performed, just as if you were pressing the key manually—beginning with the current line in program memory and proceeding sequentially with the higher numbered program lines.

Whenever the calculator is in Program mode, the display shows information about the program line to which the calculator is currently set. At the left of the display is the number of the program line within program memory. The remaining digits in the display comprise a code that indicates what instruction has been stored in that program line. No code is shown for program line 00, since no instruction is stored there.

## Identifying Instructions in Program Lines

Each key on the HP-10C keyboard—except for the digit keys 0 through 9—is identified by a two-digit “keycode” that corresponds to the key's position on the keyboard. The first digit in the keycode is the number of the key row, counting from row 1 at the top; the second digit is the number of the key in that row, counting from left to right: 1 for the first key through 9 for the ninth key and 0 for the tenth key in the row. The keycode for each digit key is simply the digit on the key. Thus, when you keyed the instruction  $\boxed{\square}$  into program memory, the calculator displayed:



Fourth row, eighth key

This indicates that the key for the instruction in program line 03 is in the fourth row on the keyboard and is the eighth key in that row: the  $\boxed{\square}$  key. When you keyed the instruction  $\boxed{+}$  into program memory, the calculator displayed:



This indicates that the key for the instruction in program line 08 is in the fourth row on the keyboard and is the tenth key in that row: the **[+]** key. When you keyed the digit 3 into program memory, the keycode displayed was only the digit 3.

Since keystroke sequences beginning with **[f]**, **[STO]**, **[RCL]**, and **[GTO]** are stored in only one program line, the display of that line would show the keycodes for all the keys in the keystroke sequence.

| Instruction               | Keycode    |
|---------------------------|------------|
| <b>[f]</b> LOG            | nn- 42 13  |
| <b>[STO]</b> <b>[+]</b> 1 | nn-44 40 1 |
| <b>[GTO]</b> 00           | nn- 22 00  |

Displaying Program Lines

Pressing **[P/R]** to set the calculator from Run mode to Program mode displays the line number and keycode for the program line to which the calculator is currently set.

Occasionally you'll want to check several or all of the instructions stored in program memory. The HP-10C enables you to review program instructions either forward or backward through program memory:

- Pressing **[SST]** (*single step*) while the calculator is in Program mode advances the calculator to the next line in program memory, then displays that line number and the keycode of the instruction stored there.
- Pressing **[f]** **[BST]** (*back step*) while the calculator is in Program mode sets the calculator back to the previous line in program memory, then displays that line number and the keycode of the instruction stored there.

For example, to display the first two lines of the program now stored in program memory, set the calculator to Program mode and press **[SST]** twice:

| Keystrokes   | Display |    |  |
|--------------|---------|----|--|
| <b>[P/R]</b> | 00-     |    | Sets calculator to Program mode and displays current line of program memory. |
| <b>[SST]</b> | 01-     | 36 | Program line 01: <b>[ENTER]</b> .  |
| <b>[SST]</b> | 02-     | 1  | Program line 02: digit 1.  |

Pressing **[f]** **[BST]** does the reverse:

**Keystrokes****Display**

|                  |            |           |                  |
|------------------|------------|-----------|------------------|
| <b>[f] [BST]</b> | <b>01-</b> | <b>36</b> | Program line 01. |
| <b>[f] [BST]</b> | <b>00-</b> |           | Program line 00. |

If either the **[SST]** key or the **[BST]** key is held down, the calculator displays *all* of the lines in program memory. Press **[SST]** again now, but this time hold it down until program line 08 is displayed.

**Keystrokes****Display**

|                        |            |           |                  |
|------------------------|------------|-----------|------------------|
| <b>[SST]</b>           | <b>01-</b> | <b>36</b> | Program line 01. |
|                        | <b>:</b>   | <b>:</b>  |                  |
| <b>(Release [SST])</b> | <b>00-</b> | <b>40</b> | Program line 08. |

Program line 08 contains the last instruction you *keyed into* program memory. However, if you press **[SST]** again, you'll see that this is *not* the last line *stored* in program memory:

**Keystrokes****Display**

|              |            |              |                  |
|--------------|------------|--------------|------------------|
| <b>[SST]</b> | <b>09-</b> | <b>22 00</b> | Program line 09. |
|--------------|------------|--------------|------------------|

As you should now be able to tell from the keycodes displayed, the instruction in program line 09 is **[GTO]00**.

**The **[GTO] 00** instruction and Program line 00**

Whenever you run the program now stored in program memory, the calculator executes the instruction in line 09 after executing the eighth instruction you keyed in. This **[GTO]00** instruction—as its name implies—tells the calculator to “go to” program line 00 and execute the instruction in that line. Although line 00 does not contain a regular instruction, it does contain a “hidden” instruction that tells the calculator to halt program execution. Thus, after each time the program is run, the calculator automatically goes to program line 00 and halts, ready for you to key in new data and run the program again. (The calculator is also automatically set to program line 00 when you press **[P/R]** to set the calculator from Program mode to Run mode.)

The **[GTO] 00** instruction was already stored in line 09—in fact, in *all* program lines—*before* you keyed in the program. If no instructions have been keyed into program memory, if Continuous Memory is reset, or if **[f] CLEAR [PRGM]** is pressed (in Program mode), the instruction **[GTO]00** is automatically stored in program lines 01 through 09. As you key each instruction into program memory, it replaces the **[GTO]00** instruction in that program line.

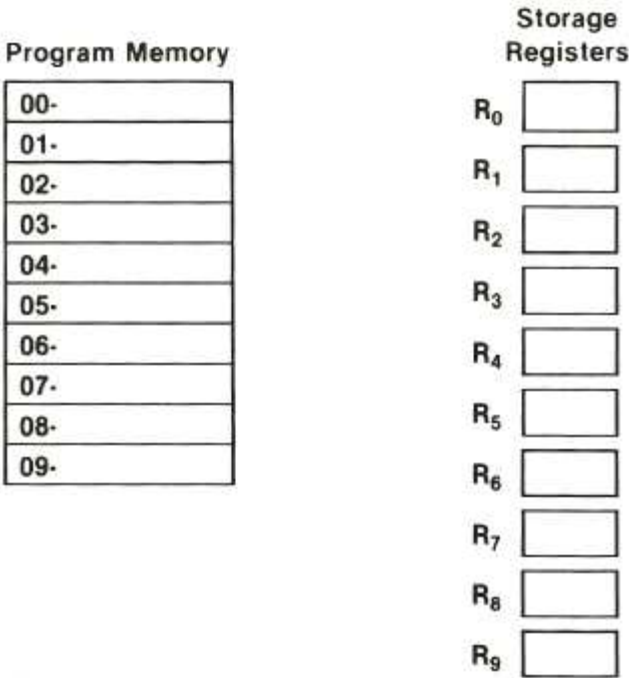
If your program should consist of exactly nine instructions, there would be no **[GTO]00** instructions remaining at the end of program memory. Nevertheless, after such a program is executed the calculator automatically returns to program

line 00 and halts, just as if there were a **[GTO]00** instruction following the program.

If you key in more than nine instructions, program memory automatically expands to accommodate the additional instructions.

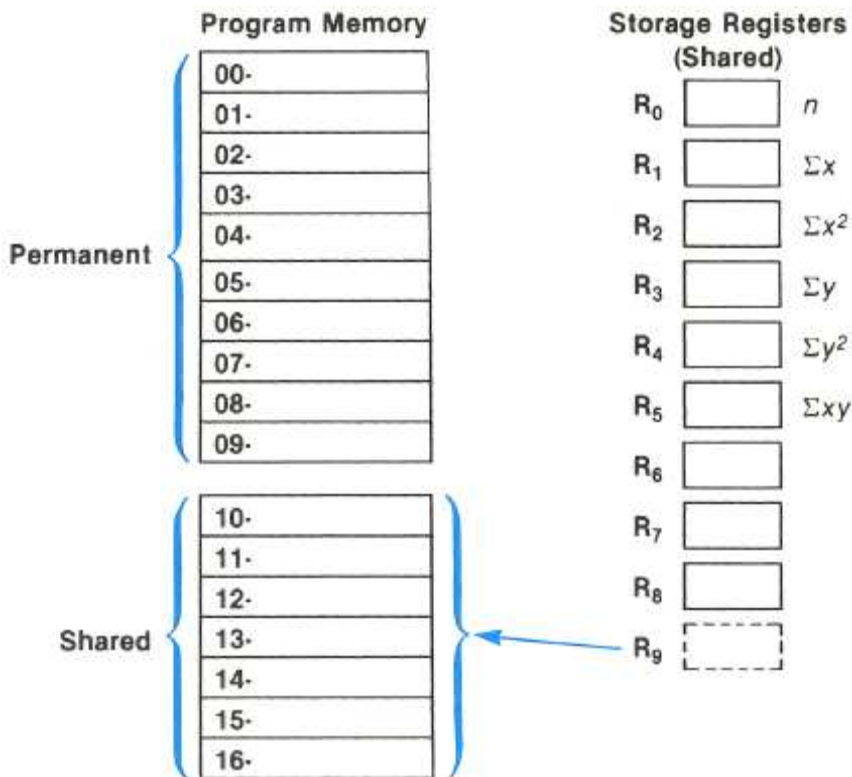
**Expanding Program Memory; the **[MEM]** Key**

If no instructions have been keyed into program memory, if Continuous Memory has been reset, or if **[f]CLEAR[PRGM]** has been pressed (in Program mode), program memory consists of nine program lines, and there are 10 storage registers available for storage of data.



As you key in a 10th instruction, storage register R<sub>9</sub> is automatically converted into seven new lines of program memory. The instruction you key in is stored in program line 10, and the **[GTO]00** instruction is automatically stored in program lines 11 through 16.





Program Memory Allocation:

| Register       | Line Number |
|----------------|-------------|
| R <sub>9</sub> | 10-16       |
| R <sub>8</sub> | 17-23       |
| R <sub>7</sub> | 24-30       |
| R <sub>6</sub> | 31-37       |
| R <sub>5</sub> | 38-44       |
| R <sub>4</sub> | 45-51       |
| R <sub>3</sub> | 52-58       |
| R <sub>2</sub> | 59-65       |
| R <sub>1</sub> | 66-72       |
| R <sub>0</sub> | 73-79       |

Program memory is automatically expanded like this whenever another seven instructions have been keyed into program memory—that is, when you key an instruction into program line 17, 24, 31, etc. In each case, the additional program lines made available are converted, seven lines at a time, from the last available data storage register (whether or not a number has been stored in that

register; if it has, it will be lost). Furthermore, the six new program lines (following the 17th, 24th, etc.) will each contain the instruction  $\boxed{\text{GTO}}00$ .

To determine at any time (whether in Program or Run mode) how many program lines {including those containing  $\boxed{\text{GTO}}00$ } are currently in memory and how many storage registers are currently available for conversion to program lines or for data storage, press and hold  $\boxed{\text{f}}\boxed{\text{MEM}}$  (*memory*). The calculator will respond with a display like the following:



Up to 79 instructions can be stored in program memory. Doing so would require the conversion of 10 data storage registers ( $79 = 9 + [10 \times 7]$ ), leaving no storage registers available for data storage.

**Note:** Your HP-10C converts storage registers to program lines in reverse numerical order, from  $R_9$  to  $R_0$ . For this reason it is good practice to perform  $\boxed{\text{STO}}$  and  $\boxed{\text{RCL}}$  operations using data registers in the opposite order; that is, beginning with register  $R_0$ . This procedure helps avoid accidentally trying to use  $\boxed{\text{STO}}$  and  $\boxed{\text{RCL}}$  for data registers which have been converted to lines of program memory. Remember also that the calculator does not retain data previously stored in registers that are later converted to lines of program memory.

## Setting the Calculator to a Particular Program Line

There will be occasions when you'll want to set the calculator directly to a particular program line—such as when you're storing a second program in program memory or when you're modifying an existing program. Although you can set the calculator to any program line by using  $\boxed{\text{SST}}$  as described above, you can do so more quickly as follows:

- With the calculator in Program mode, pressing  $\boxed{\text{GTO}}\boxed{\cdot}$  followed by two digit keys sets the calculator to the program line specified by the digit keys, and then displays that line number and the keycode of the instruction stored there.
- With the calculator in Run mode, pressing  $\boxed{\text{GTO}}$  followed by two digit keys sets the calculator to the program line specified by the digit keys. Since the calculator is not in Program mode, the line number and keycode are not displayed.

The decimal point is not necessary if the calculator is in Run mode, but it is necessary if the calculator is in Program mode.

For example, assuming the calculator is still in Program mode, you can set it to program line 00 as follows:

| Keystrokes                              | Display |                  |
|---|---------|------------------|
| $\boxed{\text{GTO}} \boxed{\bullet} 00$ | 00-     | Program line 00. |

## Executing a Program One Line at a Time

Pressing  $\boxed{\text{SST}}$  repeatedly with the calculator in Program mode (as described earlier) enables you to verify that the program you have *stored* is identical to the program you *wrote*—that is, to verify that you have keyed the instructions in correctly. However, this does not ensure that the program you wrote *calculates* the desired results correctly: even programs created by the most experienced programmers often do not work correctly when they are first written.

To help you verify that your program works correctly, you can execute the program one line at a time, using the  $\boxed{\text{SST}}$  key. Pressing  $\boxed{\text{SST}}$  while the calculator is in Run mode advances the calculator to the next line in program memory, then displays that line's number and the keycode of the instruction stored there, just as in Program mode. In *Run* mode, however, when the  $\boxed{\text{SST}}$  key is released the instruction in the program line just displayed is executed and the display then shows the result of executing that line. (You can retain the program line information by holding the  $\boxed{\text{SST}}$  key down.)

For example, to execute the program stored in the calculator one line at a time:

| Keystrokes           | Display     |   |
|----------------------|-------------|---|
| $\boxed{\text{P/R}}$ | 39.7400     | Sets calculator to Run mode and to line 00 in program memory. (Display shown assumes results remain from previous calculation.) |
| 35.7                 | 35.7        | Keys in temperature of water bath (degrees Celsius).  |
| $\boxed{\text{SST}}$ | 01-      36 | Program line 01: $\boxed{\text{ENTER}}$ .   |
|                      | 35.7000     | Result of executing program line 01.  |
| $\boxed{\text{SST}}$ | 02-      1  | Program line 02: 1.   |
|                      | 1.          | Result of executing program line 02.  |

| Keystrokes | Display        |    |  |
|------------|----------------|----|--|
| <b>SST</b> | 03-<br>1.      | 48 | Program line 03: <b>•</b> .<br>Result of executing<br>program line 03.                                   |
| <b>SST</b> | 04-<br>1.8     | 8  | Program line 04: 8.<br>Result of executing<br>program line 04.   |
| <b>SST</b> | 05-<br>64.2600 | 20 | Program line 05: <b>×</b> .<br>Result of executing<br>program line 05.                                   |
| <b>SST</b> | 06-<br>3.      | 3  | Program line 06: 3.<br>Result of executing<br>program line 06.   |
| <b>SST</b> | 07-<br>32.     | 2  | Program line 07: 2.<br>Result of executing<br>program line 07.   |
| <b>SST</b> | 08-<br>96.2600 | 40 | Program line 08: <b>+</b> .<br>Result of executing<br>program line 08 (the last<br>line of the program). |

Pressing **f** **BST** while the calculator is in Run mode sets the calculator to the previous line in program memory, then displays that line's number and the keycode of the instruction stored there, just as in Program mode. In *Run* mode, however, when the **BST** key is released the display again shows the same number as it did before **f** **BST** was pressed: *no* instruction in program memory is executed.

## Interrupting Program Execution

Occasionally you'll want a program to stop executing so that you can see an intermediate result or enter new data. The HP-10C provides two functions for doing so: **PSE** (*pause*) and **R/S** (*run/stop*).

### Pausing During Program Execution

When a running program executes a **PSE** instruction, program execution halts for about 1 second, then resumes. During the pause, the calculator displays the last result calculated before the **PSE** instruction was executed.

If you press any key during a pause, program execution is halted indefinitely. To resume program execution at the program line following that containing the **PSE** instruction, press **R/S**.

## Stopping Program Execution Automatically

Program execution is automatically halted when the program executes a  $\boxed{\text{R/S}}$  instruction. To resume executing the program from the program line at which execution was halted, press  $\boxed{\text{R/S}}$ .

**Example:** Mother's Kitchen, a canning company, wants to package a ready-to-eat spaghetti mix containing three cans: one of spaghetti sauce, one of grated cheese, and one of meatballs. Mother's needs to calculate the base areas, total surface areas, and volumes of the three cylindrically shaped cans. It would also like to know, per package, the total can base area, surface area, and volume.



The program to calculate this information uses these formulas and data:

$$\text{base area} = \pi r^2$$

$$\text{volume} = \text{base area} \times \text{height} = \pi r^2 h$$

$$\text{surface area} = 2 \times \text{base area} + \text{side area} = 2\pi r^2 + 2\pi rh$$

| Radius, $r$   | Height, $h$ | Base Area | Volume | Surface Area |
|---------------|-------------|-----------|--------|--------------|
| 2.5 cm        | 8.0 cm      | ?         | ?      | ?            |
| 4.0           | 10.5        | ?         | ?      | ?            |
| 4.5           | 4.0         | ?         | ?      | ?            |
| <b>TOTALS</b> |             | ?         | ?      | ?            |

The keystroke sequence will use storage register arithmetic (described on page 30) in registers  $R_1$ ,  $R_2$ , and  $R_3$  to calculate area and volume sums. We will clear the registers before starting to ensure that the column sums are initialized to zero.

Since  $r$  must be used twice in the course of the calculations, we'll store  $r$  in  $R_0$ . We will do this *prior to running* the program, since  $r$  will vary with each run. We will then recall  $r$  *in the program* with a  $\boxed{\text{RCL}}$  instruction. We will enter  $h$  directly, right after the base area calculation is made. We will provide for the program to stop at this point, so that the data can be entered. This is accomplished with an  $\boxed{\text{R/S}}$  instruction in the program. We can then manually restart the program by pressing  $\boxed{\text{R/S}}$ . (Alternatively, we could store and recall  $h$ , as for  $r$ , though in this program it would require more total keystrokes to do so.) This illustrates the two modes of data entry in a program:

1. Prior entry. Store (with  $\boxed{\text{STO}}$ ) the data in a storage register prior to running the program, and then recall it (with  $\boxed{\text{RCL}}$ ) within the program.

2. Direct entry. Enter the data at the necessary point in the program, then start or re-start (with  $\boxed{R/S}$ ) the program. If this is not at the beginning of the program, a programmed stop instruction  $\boxed{R/S}$  is necessary to allow data entry.

Now we'll enter the program into program memory. Do *not* key in the radius and height of each can; these values will vary, and so will be entered *prior* to each program run.

We'll also include  $\boxed{PSE}$  (or  $\boxed{R/S}$ ) instructions so that the intermediate results for BASE AREA, VOLUME, and SURFACE AREA are automatically displayed when the program is executed.

| Keystrokes                             | Display          |  |
|--|------------------|--|
| $\boxed{P/R}$                          | 00-              | Sets calculator to Program mode.                         |
| $\boxed{f} \boxed{CLEAR} \boxed{PRGM}$ | 00-              | Clears program memory.                                   |
| $\boxed{f} \boxed{x^2}$                | 01-      42   11 |  |
| $\boxed{f} \boxed{\pi}$                | 02-      42   16 |  |
| $\boxed{\times}$                       | 03-           20 |  |
| $\boxed{STO} 4$                        | 04-      44   4  |  |
| $\boxed{STO} \boxed{+} 1$              | 05-44   40   1   |  |
| $\boxed{R/S}$                          | 06-           31 | Stops to display BASE AREA and allow entry of $h$ value. |
| $\boxed{\times}$                       | 07-           20 |  |
| $\boxed{f} \boxed{PSE}$                | 08-      42   31 | Pauses to display VOLUME.                                |
| $\boxed{STO} \boxed{+} 2$              | 09-44   40   2   |  |
| $\boxed{RCL} 0$                        | 10-      45   0  | Recalls $r$ , which will be stored in $R_0$ .            |
| $\boxed{\div}$                         | 11-           10 |  |
| 2                                      | 12-           2  |  |
| $\boxed{\times}$                       | 13-           20 |  |
| $\boxed{RCL} 4$                        | 14-      45   4  |  |
| 2                                      | 15-           2  |  |
| $\boxed{\times}$                       | 16-           20 |  |
| $\boxed{+}$                            | 17-           40 |  |
| $\boxed{STO} \boxed{+} 3$              | 18-44   40   3   |  |

Now, to run the program:

| Keystrokes         | Display  |  |
|--------------------|----------|--|
| <b>P/R</b>         |          | Sets calculator to Run mode. (Display left from previous calculation.) |
| <b>f CLEAR REG</b> | 0.0000   | Clears registers $R_0$ through $R_9$ .                                 |
| 2.5 <b>STO</b> 0   | 2.5000   | Enters and stores $r$ of first can in $R_0$ .                          |
| <b>R/S</b>         | 19.6350  | BASE AREA of first can.  |
| 8                  | 8.       | Enters $h$ of first can.   |
| <b>R/S</b>         | 157.0796 | VOLUME of first can.   |
|                    | 164.9336 | SURFACE AREA of first can.   |
| 4 <b>STO</b> 0     | 4.0000   | Enters and stores $r$ of second can in $R_0$ .                         |
| <b>R/S</b>         | 50.2655  | BASE AREA of second can.   |
| 10.5               | 10.5     | Enters $h$ of second can.  |
| <b>R/S</b>         | 527.7876 | VOLUME of second can.  |
|                    | 364.4247 | SURFACE AREA of second can.  |
| 4.5 <b>STO</b> 0   | 4.5000   | Enters and stores $r$ of third can in $R_0$ .                          |
| <b>R/S</b>         | 63.6173  | BASE AREA of third can.  |
| 4                  | 4.       | Enters $h$ of third can.   |
| <b>R/S</b>         | 254.4690 | VOLUME of third can.   |
|                    | 240.3318 | SURFACE AREA of third can.   |
| <b>RCL</b> 1       | 133.5177 | Sum of BASE AREAS.   |
| <b>RCL</b> 2       | 939.3362 | Sum of VOLUMES.  |
| <b>RCL</b> 3       | 769.6902 | Sum of SURFACE AREAS.  |

If the duration of the pause is not long enough to write down the number displayed, you can prolong it by using more than one **PSE** instruction, or you can have the program stop automatically by programming a **R/S** instruction.

Program execution is also automatically halted when the calculator overflows (refer to page 17) or attempts an improper operation that results in an **Error**

display. Either of these conditions signifies that the program itself probably contains an error.

To determine at which program line execution has halted (in order to locate the error), press any key to clear the **Error** display, then press **[P/R]** to set the calculator to Program mode and display that program line.

You may also want to display the current program line (by pressing **[P/R]**) if your program has halted at one of several **[R/S]** instructions in your program and you want to determine which one that is. To continue executing the program afterward:

1. Press **[P/R]** to set the calculator back to Run mode. This also sets the program back to line 00.
2. If you want to resume execution from the program line at which execution halted rather than from line 00, press **[GTO]** followed by the two-digit program line number desired.
3. Press **[R/S]** to resume execution.

## Stopping Program Execution Manually

Pressing any key while a program is running halts program execution. You may want to do this if the calculated results displayed by a running program appear to be incorrect (indicating that the program itself is incorrect).

To halt program execution during a pause in a running program (that is, when **[PSE]** is executed), press any key.

After stopping program execution manually, you can determine at which program line execution has halted and/or resume program execution as described above.

## Nonprogrammable Functions

When the calculator is in Program mode (**PRGM** annunciator displayed), almost every function on the keyboard can be recorded as an instruction in program memory. The following functions cannot be stored as instructions in program memory:

**[f] CLEAR [PRGM]**  
**[f] CLEAR [PREFIX]**  
**[GTO] [•] nn**

**[P/R]**  
**[f] [MEM]**  
**[•] [/ON]**

**[SST]**  
**[f] [BST]**



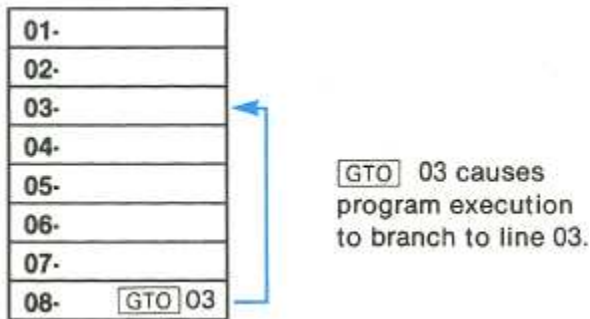
## Section 6

# Branching and Looping

Although the instructions in a program normally are executed in order of their program line numbers, in some situations it is desirable to have program execution transfer or “branch” to a program line that is not the next line in program memory. Branching also makes it possible to automatically execute portions of a program more than once—a process called “looping.”

## Simple Branching

The **GTO** (*go to*) instruction is used in a program to transfer execution to any program line. The program line desired is specified by keying its two-digit line number into the program line containing the **GTO** instruction. When the **GTO** instruction is executed, program execution branches or “goes to” the program line specified and then continues sequentially as usual.



You have already seen a common use of branching: the **GTO** 00 instruction (that is stored in program memory after the program you key in) transfers execution to program line 00. A **GTO** instruction can be used to branch forward as well as backward in program memory. Backward branching is typically done to create loops (as described next); forward branching is typically done in conjunction with an  $x \leq y$  or  $x=0$  instruction for conditional branching (as described afterward).

## Looping

If a **GTO** instruction specifies a lower numbered line in program memory, the instructions in the program lines between the specified line and the **GTO** instruction will be executed repeatedly. As can be seen in the illustration above

under Simple Branching, once the program begins executing the “loop”, it will execute it again and again.


If you want to terminate the execution of a loop, you can include an  $[X \leq Y]$  or  $[X = 0]$  instruction (described below) or an  $[R/S]$  instruction within the loop. You can also terminate execution by pressing any key while the loop is being executed.

**Example:** Your friendly neighborhood Radiobiology Lab wants to predict the diminishing radioactivity of a test amount of  $^{131}\text{I}$ , a radioisotope. The following program automatically figures the number of milliCuries of isotope theoretically remaining at four-day intervals of decay. (The half-life of  $^{131}\text{I}$  is 8 days.) The initial batch ( $N_0$ ) of isotope for this example is 100 milliCuries.

The formula for  $N_t$ , the amount of radioisotope left after  $t$  days, is:

$$N_t = N_0 e^{-kt}$$

where  $k = \frac{\ln 2}{8 \text{ days}} = 0.087 \text{ days}^{-1}$

 half-life of  $^{131}\text{I}$

We'll store the  $N_0$  value (100 milliCuries) in register  $R_1$  before running the program, and enter the first  $t$  value (4 days) just prior to executing the program. The program will automatically increase the value of  $t$  by 4 days for each successive  $N_t$  calculation (each loop).

| Keystrokes                  | Display           |  |
|-----------------------------|-------------------|--|
| $[P/R]$                     | 00-               | Sets calculator to Program mode.   |
| $[f] \text{ CLEAR } [PRGM]$ | 00-               | Clears program memory.   |
| $[STO] 0$                   | 01-    44    0    | Stores number from display in $R_0$ . This number will be $t$ , the number of days the $^{131}\text{I}$ batch has existed.             |
| $[RCL] 0$                   | 02-    45    0    | Recalls $t$ . This program line is the one to which program execution will later branch, $t$ will change each time the program is run. |
| $[f] \text{ PSE}$           | 03-    42    31   | Pauses to display $t$ .  |
| 2                           | 04-            2  |  |
| $[f] \text{ LN}$            | 05-    42    12   |  |
| 8                           | 06-            8  |  |
| $[\div]$                    | 07-            10 |  |
| $[x]$                       | 08-            20 | $kt$ .   |

| Keystrokes             | Display    |  |
|------------------------|------------|--|
| <b>[CHS]</b>           | 09- 16     | $-kt$ .  |
| <b>[e<sup>x</sup>]</b> | 10- 12     | $e^{-kt}$ .  |
| <b>[RCL] 1</b>         | 11- 45 1   | Recalls $N_0$ , the initial number of milliCuries, from $R_1$ .  |
| <b>[x]</b>             | 12- 20     | $N_t$ , the milliCuries of $^{131}\text{I}$ remaining after $t$ days.  |
| <b>[f] [PSE]</b>       | 13- 42 31  | Pauses to display $N_t$ .  |
| 4                      | 14- 4      |  |
| <b>[STO] [+]</b> 0     | 15-44 40 0 | Adds 4 days to $t$ .   |
| <b>[GTO]</b> 02        | 16- 22 02  | Transfers program execution to line 02, so the new $t$ can be recalled to display before beginning new calculations. |

Now we can run the program. It will pause at the beginning and end of each loop to display each  $t$  and  $N_t$  value. Program execution will continue indefinitely—until we stop it. (You can run the program one step at a time—using **[SST]**—if you want to follow the execution of the loop.)

| Keystrokes             | Display |  |
|------------------------|---------|--|
| <b>[P/R]</b>           | 0.0000  | Sets calculator to Run mode. (Display shown assumes no results remain from previous calculations.) |
| <b>[f] CLEAR [REG]</b> | 0.0000  | Clears registers $R_0$ through $R_9$ .   |
| <b>[f] [FIX] 3</b>     | 0.000   | Sets display format to three decimal places.   |
| 100 <b>[STO]</b> 1     | 100.000 | Stores $N_0$ in $R_1$ .  |
| 4                      | 4.      | Keys in $t$ .  |
| <b>[R/S]</b>           | 4.000   | $t = 4$ days.  |
|                        | 70.711  | $N_4$ (milliCuries of $^{131}\text{I}$ remaining after 4 days).                                    |
|                        | 8.000   | $t = 8$ days (the half-life of $^{131}\text{I}$ ).   |
|                        | 50.000  | $N_8$ .  |
|                        | 12.000  | $t = 12$ days.   |

| Keystrokes                    | Display  |                          |
|-------------------------------|----------|--------------------------|
|                               | 35 . 355 | N <sub>12</sub> .        |
| <span>R/S</span> (or any key) | 35 . 355 | Halts program execution. |

## Conditional Branching

Often there are situations when it is desirable for a program to be able to branch to different lines in program memory, depending on certain conditions. For example, a program used to calculate sales commissions might need to branch to different program lines depending on the commission rate for the particular sales level. Also, conditional branching can be used to end the execution of a loop such as that in the preceding example. A conditional instruction can shift the program out of a loop either after a specified number of loop executions, or when a certain value within the loop has been reached.

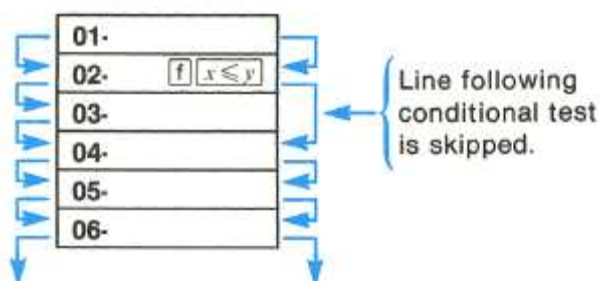
The HP-10C provides two *conditional test* instructions that are used in programs for conditional branching:

- $x \leq y$  tests whether the number in the X-register (represented by the *x* in the key symbol) is less than or equal to the number in the Y-register (represented by the *y* in the key symbol).
- $x = 0$  tests whether the number in the X-register is equal to zero.

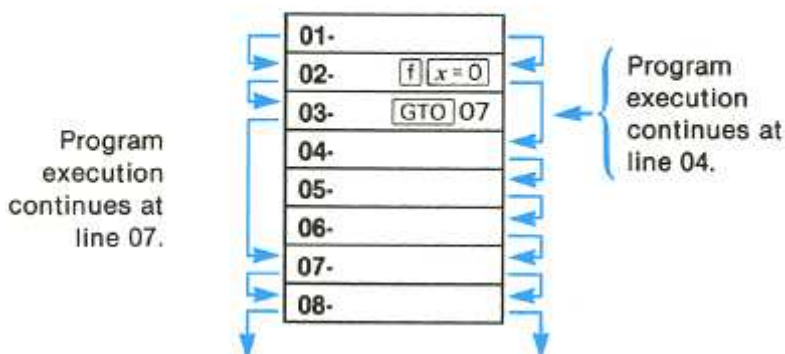
The possible results of executing either of these instructions are:

- If the condition tested for is true when the instruction is executed, program execution continues sequentially with the instruction in the next line of program memory.
- If the condition tested for is false when the instruction is executed, program execution skips the instruction in the next line of program memory and continues with the instruction in the following line.

These rules can be summarized as “DO if TRUE.”

Program Execution  
If TrueProgram Execution  
If False

The program line immediately following that containing the conditional test instruction can contain any instruction; however, the most commonly used instruction there is `GTO`. If a `GTO` instruction follows a conditional test instruction, program execution branches elsewhere in program memory if the condition is true and continues with the next line in program memory if the condition is false.

Program Execution  
If TrueProgram Execution  
If False

**Example:** The radioisotope decay program (page 72) can be made to stop automatically after a specified number of loops by using the `x=0` condition.

Suppose  $N$  at  $t = 12$  days is the last value you need. You can stop execution automatically at this point by storing the number of loops to be run (3), subtracting 1 from this number each time the loop is run, and instructing the program to stop (with a `GTO 00` instruction) when this number equals zero.

The revised program would then look like this

**Keystrokes****Display**

P/R

00-

f CLEAR PRGM

00-

| Keystrokes                | Display    |   |
|---------------------------|------------|---|
| <b>[STO]</b> 0            | 01- 44 0   |   |
| 3                         | 02- 3      | Loop counter. Initializes the total number of loops to be run.          |
| <b>[STO]</b> 2            | 03- 44 2   | Stores loop counter in R <sub>2</sub> .*                                |
| <b>[RCL]</b> 0            | 04- 45 0   | Original lines.   |
| <b>[f]</b> <b>[PSE]</b>   | 05- 42 31  |   |
| 2                         | 06- 2      |   |
| <b>[f]</b> <b>[LN]</b>    | 07- 42 12  |   |
| 8                         | 08- 8      |   |
| <b>[÷]</b>                | 09- 10     |   |
| <b>[x]</b>                | 10- 20     |   |
| <b>[CHS]</b>              | 11- 16     |   |
| <b>[e<sup>x</sup>]</b>    | 12- 12     |   |
| <b>[RCL]</b> 1            | 13- 45 1   |   |
| <b>[x]</b>                | 14- 20     |   |
| <b>[f]</b> <b>[PSE]</b>   | 15- 42 31  |   |
| 4                         | 16- 4      |   |
| <b>[STO]</b> <b>[+]</b> 0 | 17-44 40 0 |   |
| 1                         | 18- 1      | Loop increment number.  |
| <b>[STO]</b> <b>[-]</b> 2 | 19-44 30 2 | Subtracts 1 from initialized loop counter (line 02) in R <sub>2</sub> . |
| <b>[RCL]</b> 2            | 20- 45 2   | Recalls loop counter to display.  |
| <b>[f]</b> <b>[x=0]</b>   | 21- 42 20  | Tests whether number in X-register (loop number) equals zero.           |
| <b>[GTO]</b> 00           | 22- 22 00  | If condition is true, branches to beginning of program and stops.       |

---

\* These lines can be entered manually *prior* to running the program instead of writing them into the program if you wish to: 1) conserve program lines; or 2) retain the flexibility of periodically changing the number of loops to be executed.

**Keystrokes****Display**

GTO 04

23- 22 04

If condition is false, branches to line 04 to continue program by restarting loop. (Note that the line number specified has been changed.)

To run this revised program, we proceed as follows:

**Keystrokes****Display**

P/R

Sets calculator to Run mode.

f CLEAR REG

0.000

100 STO 1

100.000

Stores  $N_0$  in  $R_1$ .

4

4.

Keys in  $t$ .

R/S

4.000

$t$ .

70.711

$N_4$ .

8.000

50.000

$N_8$ .

12.000

35.355

$N_{12}$ .

0.000

Loop counter equals 0.

**Exercise:** Write a program that will enable a salesperson to compute a commission at the rates of 10% for sales up through \$1,000 and 12.5% for sales over \$1,000. The test value (1,000) and the commission rates can be stored for recall or included in the program. Below, they are stored in registers  $R_0$  through  $R_2$  for later recall by the program.

**Note:** If a program requires that certain numbers be in the X- and Y-registers when instructions such as  $[X \leq Y]$  are executed, it is helpful when writing the program to show the quantities in each register after each instruction is executed, as in the following diagram.



|        |      |                      |                      |                      |                       |
|--------|------|----------------------|----------------------|----------------------|-----------------------|
|        | 1    | 2                    | 3                    | 4                    | 5                     |
| Y →    | 0    | sale                 | 1,000                | 1,000                | 1,000                 |
| X →    | sale | 1,000                | sale                 | sale                 | sale                  |
| Keys → | sale | <code>[RCL] 0</code> | <code>[X ≥ Y]</code> | <code>[X ≤ Y]</code> | <code>[GTO] 07</code> |
| Line → |      | 01                   | 02                   | 03                   | 04                    |

|        |                      |                       |                      |                  |
|--------|----------------------|-----------------------|----------------------|------------------|
|        | 6                    | 7                     | 8                    | 9                |
| Y →    | sale                 | sale                  | sale                 | sale             |
| X →    | 12.50                | 12.50                 | 10.00                | commis.          |
| Keys → | <code>[RCL] 2</code> | <code>[GTO] 08</code> | <code>[RCL] 1</code> | <code>[%]</code> |
| Line → | 05                   | 06                    | 07                   | 08               |

We'll key the amount of the sale into the display before running the program so that it will be in the X-register before the `[RCL] 0` instruction in program line 01 is executed. This instruction will place the test value (1,000) in the X-register (the display) and move the sale amount into the Y-register. The `[X ≤ Y]` instruction in program line 02 will exchange the numbers in the X- and Y-registers: that is, it will place the sales figure back into the X-register and place the test value into the Y-register. This is necessary because when either the `[RCL] 2` instruction in line 05 or the `[RCL] 1` instruction in line 07 is executed, the number in the X-register is moved into the Y-register; if the `[X ≤ Y]` instruction were not included, the test value (1,000) rather than the sale amount would be in the Y-register when the `[%]` instruction in line 08 is executed.

| Keystrokes                    | Display          |   |
|-------------------------------|------------------|---|
| <code>[P/R]</code>            | 00-              |   |
| <code>[f] CLEAR [PRGM]</code> | 00-              |   |
| <code>[RCL] 0</code>          | 01-     45   0   | Recalls test value (1,000) into X-register.   |
| <code>[X ≤ Y]</code>          | 02-           34 | Places sale amount (which you will key in prior to running the program) into X-register and test value into Y-register. |
| <code>[f] [X ≤ Y]</code>      | 03-     42   10  | Tests whether number in X-register (sale) is less than or equal to number in Y-register (1,000).                        |



| Keystrokes      | Display   |   |
|-----------------|-----------|---|
| <b>[GTO]</b> 07 | 04- 22 07 | If condition is true, branches to program line 07.                            |
| <b>[RCL]</b> 2  | 05- 45 2  | If condition is false, recalls commission rate of 12.5% from R <sub>2</sub> . |
| <b>[GTO]</b> 08 | 06- 22 08 | Branches program to line 08.  |
| <b>[RCL]</b> 1  | 07- 45 1  | Recalls commission rate of 10% from R <sub>1</sub> .                          |
| <b>[%]</b>      | 08- 21    | Calculates commission.  |

Now we'll store the required numbers in registers R<sub>0</sub>, R<sub>1</sub>, and R<sub>2</sub>, and then run the program, using **[SST]** so that we can check that the branching occurs properly. It's good practice with programs containing conditional test instructions to check that the program branches correctly for all possible conditions: in this case, if the sale amounts are less than, equal to, or greater than the test value.

| Keystrokes   | Display |   |
|--------------|---------|---|
| <b>[P/R]</b> | 0.000   | Sets calculator to Run mode. (Display shows result of previous calculations.) |

(Initializing)

|                                      |          |  |
|--------------------------------------|----------|--|
| <b>[f]</b> <b>CLEAR</b> <b>[REG]</b> | 0.000    |  |
| <b>[f]</b> <b>[FIX]</b> 2            | 0.00     | Sets display format to two decimal places.       |
| 1000 <b>[STO]</b> 0                  | 1,000.00 | Stores test value in R <sub>0</sub> .            |
| 10 <b>[STO]</b> 1                    | 10.00    | Stores 10% commission rate in R <sub>1</sub> .   |
| 12.5 <b>[STO]</b> 2                  | 12.50    | Stores 12.5% commission rate in R <sub>2</sub> . |

(First Data Test)

| Keystrokes       | Display              |  |
|------------------|----------------------|--|
| 500              | 500.                 | Keys sale amount less than test value into display (X-register).   |
| <code>SST</code> | 01- 45 0<br>1,000.00 | Program line 1: <code>RCL</code> 0.<br>Test value has been recalled to X-register, moving sale amount to Y-register.           |
| <code>SST</code> | 02- 34<br>500.00     | Program line 02: <code>X≤Y</code> .<br>Sale amount has been placed in X-register and test value has been placed in Y-register. |
| <code>SST</code> | 03- 42 10<br>500.00  | Program line 03:<br><code>f</code> <code>X≤Y</code> .  |
| <code>SST</code> | 04- 22 07<br>500.00  | Condition tested by <code>X≤Y</code> was true, so program execution continued with line 04:<br><code>GTO</code> 07.            |
| <code>SST</code> | 07- 45 1<br>10.00    | Program line 07: <code>RCL</code> 1.<br>10% commission rate has been recalled to X-register, moving sale amount to Y-register. |
| <code>SST</code> | 08- 21<br>50.00      | Program line 08: <code>%</code> .<br>10% of 500.00 = 50.00   |
| <code>SST</code> | 09- 22 00<br>50.00   | Program line 09: sets calculator back to line 00.<br>Commission.   |

(Second Data Test)

|      |        |  |
|------|--------|--|
| 1000 | 1,000. | Keys sale equal to test value into display (X-register). |
|------|--------|--|

## Keystrokes

## Display

|            |                       |   |
|------------|-----------------------|---|
| <b>SST</b> | 01- 45 0<br>1,000.00  | Program line 1: <b>RCL</b> 0.<br>Test value has been recalled to X-register, moving sale amount to Y-register.              |
| <b>SST</b> | 02- 34<br>1,000.00    | Program line 02: <b>X&lt;Y</b> .<br>Sale amount has been placed in X-register and test value has been placed in Y-register. |
| <b>SST</b> | 03- 42 10<br>1,000.00 | Program line 03:<br><b>f</b> <b>X&lt;Y</b> .  |
| <b>SST</b> | 04- 22 07             | Condition tested by <b>X&lt;Y</b> was true, so program execution continued with line 04: <b>GTO</b> 07.                     |
| <b>SST</b> | 07- 45 1<br>10.00     | Program line 07: <b>RCL</b> 1.<br>10% commission rate has been recalled to X-register, moving sale amount to Y-register.    |
| <b>SST</b> | 08- 21<br>100.00      | Program line 08: <b>%</b> .<br>10% of 1,000.00 = 100.00.  |
| <b>SST</b> | 09- 22 00<br>100.00   | Program line 09: sets calculator back to line 00.<br>Commission.  |

(Third Data Entry)

|      |        |  |
|------|--------|--|
| 1500 | 1,500. | Keys sale greater than test value into display (X-register). |
|------|--------|--|

| Keystrokes       | Display                         |   |
|------------------|---------------------------------|---|
| <code>SST</code> | 01-     45   0<br>1,000.00      | Program line 1: <code>RCL</code> 0.<br>Test value has been recalled to X-register, moving sale amount to Y-register.  |
| <code>SST</code> | 02-           34<br>1,500.00    | Program line 02: <code>X&lt;Y</code> .<br>Sale amount has been placed in X-register and test value has been placed in Y-register.   |
| <code>SST</code> | 03-     42 10<br><br>1,500.00   | Program line 03: <code>f</code> <code>X&lt;Y</code> .   |
| <code>SST</code> | 05-     45   2<br><br><br>12.50 | Condition tested by <code>X&lt;Y</code> was false, so program execution skipped the next line and continued at line 05: <code>RCL</code> 2.<br>12.5% commission rate has been recalled to X-register, moving sale amount to Y-register. |
| <code>SST</code> | 06-     22 08<br><br>12.50      | Program line 06: <code>GTO</code> 08.   |
| <code>SST</code> | 08-           21<br>187.50      | Program line 08: <code>%</code> .<br>12.5% of 1,500.00 = 187.50.  |
| <code>SST</code> | 09-     22 00<br><br>187.50     | Program line 09: sets calculator back to line 00.<br>Commission.  |

## Section 7

# Program Editing

There are various reasons why you might want to modify a program you have stored in program memory, such as to correct a program that turns out to have errors, or to insert new instructions (like **[STO]**, **[PSE]**, or **[R/S]**).

Using program editing, you can do such modifications without rekeying in an entire program.

## Changing an Instruction in a Program Line

To change a single instruction in program memory:

1. In Program mode, use **[SST]**, **[BST]**, or **[GTO] [•]** to set the calculator to the program line *preceding* the line containing the instruction to be changed.
2. Key in the new instruction.
3. Return to Run mode.

For example, to change an instruction stored in program line 05, press **[GTO] [•] 04**, then key in the new instruction that is to be stored in program line 05. The instruction previously stored in line 05 will be replaced; it is *not* “bumped” into line 06.

## Adding Instructions at the End of a Program

To add one or more instructions at the end of the last program stored in program memory:

1. In Program mode, set the calculator to the last (highest numbered) line keyed into program memory.
2. Key in the new instruction(s).
3. Return to Run mode.

## Adding Instructions Within a Program

If instructions are to be added within a program—or at the end of a program which is not the last one in memory—simply keying them in will replace the instructions previously stored in those program lines. The contents of all higher numbered program lines will remain unchanged. Then, the replaced (written-over) instructions need to be added back. This can be done by re-keying in these and all remaining instructions. (Refer to Adding Instructions by Replacement.) Alternatively, new instructions can be added by branching outside the program(s) to add lines. (Refer to Adding Instructions by Branching.)

### Adding Instructions by Replacement

1. In Program mode, set the calculator to the last program line to be executed before the added instruction(s).
2. Key in the new instruction(s).
3. Re-key in the original instructions, starting with the first one written over by a new instruction. (Remember to alter **GTO** line numbers as necessary.)

**Example:** With the sales commission program, test value (in  $R_0$ ), and commission rates (in  $R_1$  and  $R_2$ ), from the preceding section still stored in the calculator, let's insert a new instruction. Add a **R/S** instruction before the **%** instruction so that the program will display the commission rate before displaying the amount of commission. Since there is only one instruction (**%**) to be keyed back in, it is simplest to add the **R/S** instruction by replacement, as follows:

| Keystrokes             | Display           |   |
|------------------------|-------------------|---|
| <b>P/R</b>             | 00-               | Sets calculator to Program mode.  |
| <b>GTO</b> <b>▢</b> 07 | 07-    45    1    | Sets calculator to last program line to be executed, which contains the <b>RCL</b> 1 instruction. |
| <b>R/S</b>             | 08-            31 | Keys in new instruction.  |
| <b>%</b>               | 09-            21 | Keys in original instruction which was replaced by new instruction.                               |
| <b>P/R</b>             |                   | Sets calculator back to Run mode. (Display will show results remaining from last calculation.)    |

| Keystrokes             | Display |   |
|------------------------|---------|---|
| 500 <span>[R/S]</span> | 10.00   | Percent commission rate for a \$500 sale. |
| <span>[R/S]</span>     | 50.00   | Commission for a \$500 sale.              |

## Adding Instructions by Branching

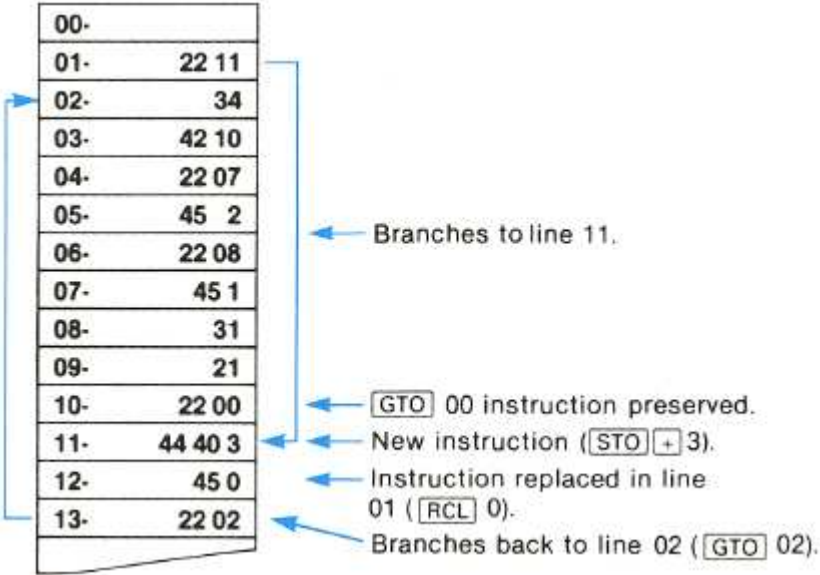
With branching, you can use a [GTO] instruction to move program execution to a new line sequence *after* the current end of your program. A second [GTO] instruction then returns execution to the main body of the program.

The new line sequence must begin at the *second* line after the end of the original program in order to preserve the automatically recorded [GTO]00 instruction (which tells the calculator to stop and return to the beginning of the program when the program is done).

1. In Program mode, set the calculator to the last program line to be executed before the added instruction(s).
2. Key in a [GTO] instruction that specifies the *second* line after the last line of your program(s).
3. Set the calculator to the last line of your program(s) and key in a [GTO]00 instruction.
4. Key in the instruction(s) being added.
5. Key in the instruction that was replaced by the [GTO] instruction keyed in at step 2.
6. Key in a [GTO] instruction to return to the first line (in the original program) to be executed after the new instruction(s).

**Example:** Suppose you now wanted to have the program in the preceding example sum the sale amounts for all the times the program is run. Let's use storage register arithmetic in register  $R_3$  to do this. This means we need to add the instruction [STO] [+] 3 before the current program line 01. Since the addition of this one line by simple replacement would require that lines 01 through 10 be re-keyed in, it is quicker to add the instructions by branching (since branching only requires four extra lines—not counting the new instructions, which are identical in each case).

The following illustration shows how branching occurs in the edited program.



| Keystrokes                        | Display       |   |
|-----------------------------------|---------------|---|
| <code>P/R</code>                  | 00-           | Sets calculator to Program mode.  |
| <code>GTO</code> <code>00</code>  | 00-           | Last program line to be executed before added instruction. In <i>this</i> case, this step is not necessary since we were at line 00 already.                      |
| <code>GTO</code> 11               | 01-    22 11  | Programs a branch to line 11, the second line after the last line of the program.   |
| <code>GTO</code> <code>09</code>  | 09-        21 | Sets calculator to the last line of the program so that the <code>GTO</code> 00 instruction keyed in next will be stored in the first line following the program. |
| <code>GTO</code> 00               | 10-    22 00  | Ensures that the <code>GTO</code> 00 instruction follows the program.   |
| <code>STO</code> <code>+</code> 3 | 11-44 40 3    | Keys in instruction being added.  |



| Keystrokes        | Display   |   |
|-------------------|-----------|---|
| <b>[RCL] 0</b>    | 12- 45 0  | Keys in instruction replaced in line 01 by <b>[GTO] 11</b> instruction. |
| <b>[GTO] 02</b>   | 13- 22 02 | Branches back to first line to be executed after added instruction.     |
| <b>[P/R]</b>      |           | Sets calculator back to Run mode.                                       |
| 0 <b>[STO] 3</b>  | 0.00      | Clears register R <sub>3</sub> .  |
| 1000 <b>[R/S]</b> | 10.00     | Commission rate for a \$1,000 sale.                                     |
| <b>[R/S]</b>      | 100.00    | Commission for a \$1,000 sale.  |
| 1500 <b>[R/S]</b> | 12.50     | Commission rate for a \$1,500 sale.                                     |
| <b>[R/S]</b>      | 187.50    | Commission for a \$1,500 sale.  |
| <b>[RCL] 3</b>    | 2,500.00  | Total sales.  |

## Section 8

# Multiple Programs

You can store multiple programs in program memory, provided that you separate them by instructions that will halt program execution after each program is run and return to the beginning of the program. You can run programs after the first one stored in program memory by setting the calculator to the first line of the program before pressing  $\boxed{R/S}$ .

## Storing Another Program

To store a program after another program is already stored in program memory:

1. In Program mode, go to the last line of the last program. Do *not* clear program memory.
2. If there is *only one* program already in program memory, add a  $\boxed{GTO}00$  instruction so that program execution returns to line 00 after this first program is run.
3. Key the program into program memory. (Be sure that any  $\boxed{GTO}$  instructions specify the correct line numbers.)
4. Press  $\boxed{R/S}$ . This will halt program execution at the end of the program.
5. If the new program does not end with a loop, key in a  $\boxed{GTO}$  instruction to the first line of the new program. This transfers program execution back to the beginning of the program if  $\boxed{R/S}$  is pressed to rerun the program.

**Example:** Assuming that program memory still contains the last program from the preceding section (which consisted of 13 program lines), store after that program the temperature conversion program from section 5 (page 56). Since this will be the second program stored in program memory, we'll insert a  $\boxed{GTO}00$  instruction to separate it from the first program (step 2 above). This program does not end with a loop, so we'll do steps 4 and 5, too.

### Keystrokes

$\boxed{P/R}$

### Display

00-

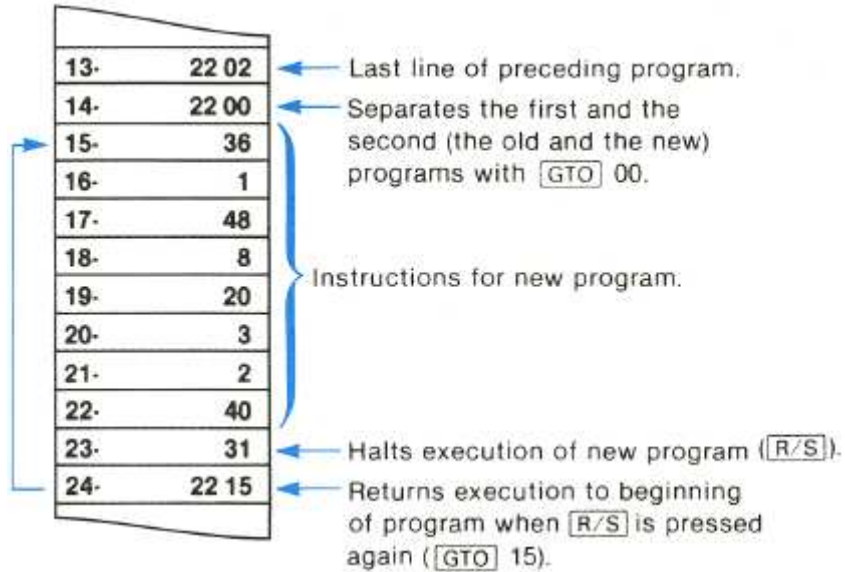
Sets calculator to  
Program mode.

**Keystrokes**

**Display**

|                     |     |       |  |
|---------------------|-----|-------|--|
| <b>[GTO] [•] 13</b> | 13- | 22 02 | Sets calculator to last line keyed into program memory.                  |
| <b>[GTO] 00</b>     | 14- | 22 00 | Ensures that second program is separated from first by <b>[GTO] 00</b> . |
| <b>[ENTER]</b>      | 15- | 36    | } Keys in program.   |
| <b>1</b>            | 16- | 1     |  |
| <b>[•]</b>          | 17- | 48    |  |
| <b>8</b>            | 18- | 8     |  |
| <b>[x]</b>          | 19- | 20    |  |
| <b>3</b>            | 20- | 3     |  |
| <b>2</b>            | 21- | 2     | } Halts program execution.   |
| <b>[+]</b>          | 22- | 40    |  |
| <b>[R/S]</b>        | 23- | 31    | Branches to beginning of program.  |
| <b>[GTO] 15</b>     | 24- | 22 15 | Sets calculator back to Run mode. (Display shows previous results.)      |
| <b>[P/R]</b>        |     |       |  |

The following illustration shows how the second program looks in program memory:



## Running Another Program

To run a program that does not begin with program line 01:

1. In Run mode, set the calculator to the first line of the program desired.
2. Press R/S.

**Example:** Run the temperature conversion program, now stored in the calculator beginning at program line 15, for a 37.5°C water bath.

| Keystrokes   | Display |  |
|--|---------|--|
| <span style="border: 1px solid black; padding: 0 2px;">GTO</span> 15   |         | Sets calculator to first line of program to be executed. (Run mode.) |
| 37.5 <span style="border: 1px solid black; padding: 0 2px;">R/S</span> | 96.26   | Temperature of water in degrees Fahrenheit.                          |

## Appendix A

# Stack Lift and LAST X

Your HP-10C calculator has been designed to operate in a natural 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—especially as you program the HP-10C—when you wish to know the effect of a particular operation upon the stack. The following explanation 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 of these operations are part of a new number. The  $\boxed{\text{CHS}}$ ,  $\boxed{\cdot}$ , and  $\boxed{\text{EEX}}$  operations do *not* terminate digit entry.

## 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 four *stack-disabling* operations on the calculator. These operations disable the stack lift, so that a number keyed in *after* one of these operations writes over the current number in the displayed X-register and the stack does not lift. These special disabling operations are:

 $\boxed{\text{ENTER}}$  $\boxed{\text{CLx}}$  $\boxed{\Sigma+}$  $\boxed{\Sigma-}$ 

### Enabling Operations

Most of the operations on the keyboard, including one- and two-number mathematical functions like  $\boxed{x^2}$  and  $\boxed{\times}$ , are *stack-enabling* operations. This

means that a number keyed in *after* one of these operations will lift the stack (because the stack has been “enabled” to lift).

|        |    |                 |                |
|--------|----|-----------------|----------------|
| T →    |    |                 |                |
| Z →    |    |                 |                |
| Y →    |    | 4.0000          | 4.0000         |
| X →    | 4. | 4.0000          | 3.             |
| Keys → | 4  | ENTER           | 3              |
|        |    | Stack disabled. | No stack lift. |

|        |                |                 |                |
|--------|----------------|-----------------|----------------|
| T →    |                |                 |                |
| Z →    |                |                 |                |
| Y →    | 53.1301        | 53.1301         | 53.1301        |
| X →    | 5.0000         | 0.0000          | 7.             |
| Keys → | f →P           | CLx             | 7              |
|        | Stack enabled. | Stack disabled. | No stack lift. |

Neutral Operations

Some operations, like **FIX**, are neutral; that is, they do not alter the previous status of the stack lift. Thus, if you disable the stack lift by pressing **ENTER**, then press **f** **FIX** *n* 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, **x<sup>2</sup>** then execute a **FIX** instruction followed by a digit entry sequence, the stack will lift.

The following operations are neutral on the HP-10C:

|            |                               |                            |            |
|------------|-------------------------------|----------------------------|------------|
| <b>FIX</b> | <b>GRD</b>                    | <b>MEM</b>                 | <b>PSE</b> |
| <b>SCI</b> | <b>GTO</b> <i>nn</i>          | <b>CLEAR</b> <b>PREFIX</b> | <b>P/R</b> |
| <b>ENG</b> | <b>GTO</b> <b>.</b> <i>nn</i> | <b>CLEAR</b> <b>PRGM</b>   | <b>x=0</b> |
| <b>DEG</b> | <b>BST</b>                    | <b>CHS</b> <sup>*</sup>    | <b>x≤y</b> |

<sup>\*</sup> **CHS** is neutral during entry of a number from the keyboard, as in 123**CHS** to enter −123, or 123 **EEX** 6**CHS** to enter 123 ×10<sup>−6</sup>. But otherwise, **CHS** enables the stack, as you would expect.

RAD

SST

R/S

ON

## LASTX

The following operations save  $x$  in the LAST X register:

-

 $\Sigma+$ 

LN

TAN

+

 $\Sigma-$  $e^x$  $\text{TAN}^{-1}$  $\times$ 

%

LOG

 $\sqrt{x}$  $\div$  $\hat{y}, r$  $10^x$  $x^2$  $\rightarrow \text{H.MS}$  $\hat{x}, r$ 

SIN

 $1/x$  $\rightarrow \text{H}$  $n!$  $\text{SIN}^{-1}$  $y^x$  $\rightarrow \text{DEG}$ 

FRAC

COS

 $\rightarrow \text{R}$  $\rightarrow \text{RAD}$ 

INT

 $\text{COS}^{-1}$  $\rightarrow \text{P}$

## Appendix B

# Error Conditions

If you attempt a calculation containing an improper operation—say, division by zero—the display will show **Error** and a number. To clear an error message, press any key.

The following operations will display **Error** plus a number:

### Error 0: Improper Mathematics Operation

Illegal argument to math routine:

$\boxed{\div}$ , where  $x = 0$ .

$\boxed{y^x}$ , where  $y = 0$  and  $x \leq 0$ , or  $y < 0$  and  $x$  is noninteger.

$\boxed{\sqrt{x}}$ , where  $x < 0$ .

$\boxed{1/x}$ , where  $x = 0$ .

$\boxed{\text{LOG}}$ , where  $x \leq 0$ .

$\boxed{\text{LN}}$ , where  $x \leq 0$ .

$\boxed{\text{SIN}^{-1}}$ , where  $|x| > 1$ .

$\boxed{\text{COS}^{-1}}$ , where  $|x| > 1$ .

$\boxed{\text{STO}}$   $\boxed{\div}$ , where  $x = 0$ .

$\boxed{n!}$ , where  $x$  is noninteger, or  $x < 0$ .

### Error 1: Storage Register Overflow

Storage register overflow (except  $\boxed{\Sigma+}$ ,  $\boxed{\Sigma-}$ ). Magnitude of number in storage register would be larger than  $9.999999999 \times 10^{99}$ .

### Error 2: Improper Statistical Operation

$\boxed{\bar{x}}$   $n = 0$

$\boxed{s}$   $n \leq 1$

$\boxed{\hat{y},r}$   $n \leq 1$

$\boxed{\hat{x},r}$   $n \leq 1$

$\boxed{\text{L.R.}}$   $n \leq 1$



**Note:** Error 2 is also displayed if division by zero or the square root of a negative number would be required during computation with any of the following formulas:

$$s_x = \sqrt{\frac{M}{n(n-1)}} \quad s_y = \sqrt{\frac{N}{n(n-1)}} \quad r = \frac{P}{\sqrt{M \cdot N}}$$

$$A = \frac{P}{M} \quad B = \frac{M \sum y - P \sum x}{n \cdot M} \quad (A \text{ and } B \text{ are the values returned by the operation } \boxed{\text{L.R.}}, \text{ where } y = Ax + B.)$$

$$\hat{y} = \frac{M \sum y + P(n \cdot x - \sum x)}{n \cdot M} \quad \hat{x} = \frac{P \sum x + M(n \cdot y - \sum y)}{n \cdot P}$$

where:

$$M = n \sum x^2 - (\sum x)^2$$

$$N = n \sum y^2 - (\sum y)^2$$

$$P = n \sum xy - \sum x \sum y$$

### Error 3: Statistical Register(s) Unavailable

Registers R<sub>0</sub> through R<sub>5</sub> unavailable for statistical computations because currently converted to program memory.

### Error 4: Improper Line Number

Line number called for is currently unoccupied, or nonexistent (>79), or you have attempted to load more than 79 lines of program memory.

### Error 5: Improper Register Number

Storage register named is currently converted to program memory, or is a nonexistent storage register.

### Error 9: Service

Self-test discovered circuitry problem, or wrong key pressed during key test. Refer to appendix C.

### Pr Error

Continuous Memory interrupted and reset because of power failure.

## Appendix C

# Battery, Warranty, and Service Information

## Batteries

The HP-10C is powered by three batteries. In “typical” use, the HP-10C has been designed to operate 6 months or more on a set of alkaline batteries. The batteries supplied with the calculator are alkaline, but silver-oxide batteries (which should last twice as long) can also be used.

A set of three fresh alkaline batteries will provide at least 80 hours of *continuous* program running (the most power-consuming kind of calculator use\*). A set of three fresh silver-oxide batteries will provide at least 180 hours of *continuous* program running. If the calculator is being used to perform operations other than running programs, it uses much less power. When only the display is on—that is, if you are not pressing keys or running programs—very little power is consumed.

If the calculator remains turned off, a set of fresh batteries will preserve the contents of Continuous Memory for as long as the batteries would last outside of the calculator—at least 1½ years for alkaline batteries or at least 2 years for silver-oxide batteries.

The actual lifetime of the batteries depends on how often you use the calculator, whether you use it more for running programs or more for manual calculations, and which functions you use.\*

The batteries supplied with the calculator, as well as the batteries listed below for replacement, are *not* rechargeable.

---

\* Power consumption in the HP-10C depends on the mode of calculator use: off (with Continuous Memory preserved); idle (with only the display on); or “operating” (running a program, performing a calculation, or having a key pressed). While the calculator is turned on, typical calculator use is a mixture of idle time and “operating” time. Therefore, the actual lifetime of the batteries depends on how much time the calculator spends in each of the three modes.

**WARNING**

Do not attempt to recharge the batteries; do not store batteries near a source of high heat; do not dispose of batteries in fire. Doing so may cause the batteries to leak or explode.

The following batteries are recommended for replacement in your HP-10C (not all batteries are available in all countries):

**Alkaline**

Eveready A76\*  
 UCAR A76  
 RAY-O-VAC RW82  
 National or Panasonic LR44  
 Varta 4276

**Silver-Oxide**

Eveready 357\*  
 UCAR 357  
 RAY-O-VAC RS76 or RW42  
 Duracell MS76  
 Duracell 10L14  
 Varta 541

**Low-Power Indication**

An asterisk (\*) flashing in the lower left corner of the display when the calculator is on signifies that the available battery power is running low.

With alkaline batteries installed:

- The calculator can be used for at least 2 hours of continuous program running after the asterisk first appears.<sup>†</sup>
- If the calculator remains turned off, the contents of its Continuous Memory will be preserved for at least 1 month after the asterisk first appears.

With silver-oxide batteries installed:

- The calculator can be used for at least 15 minutes of continuous program running after the asterisk first appears.<sup>†</sup>
- If the calculator remains turned off, the contents of its Continuous Memory will be preserved for at least 1 week after the asterisk first appears.

---

\* Not available in the United Kingdom or Republic of Ireland.

<sup>†</sup> Note that this time is the minimum available for *continuous program running*—that is, while continuously “operating” (as described in the footnote on the previous page). If you are using the calculator for manual calculations—a mixture of the idle and “operating” modes—the calculator can be used for a much longer time after the asterisk first appears.

## Installing New Batteries

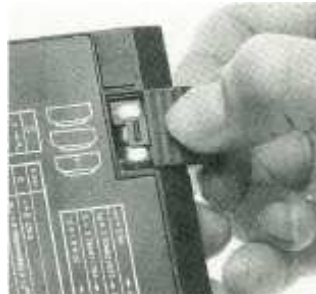
The contents of the calculator's Continuous Memory are preserved for a short time while the batteries are out of the calculator (provided that you turn off the calculator before removing the batteries). This allows you ample time to replace the batteries without losing data or programs. If the batteries are left out of the calculator for an extended period, the contents of Continuous Memory may be lost.

To install new batteries, use the following procedure

1. Be sure that the calculator is off.
2. Holding the calculator as shown, press outward on the battery compartment door until it opens slightly.



3. Grasp the outer edge of the battery compartment door, then tilt it up and out of the calculator.



### CAUTION

In the next two steps, be careful not to press any keys while batteries are out of the calculator. If you do so, the contents of Continuous Memory may be lost and keyboard control may be lost (that is, no response to keystrokes).

4. Turn the calculator over and gently shake, allowing the batteries to fall into the palm of your hand.



### CAUTION

In the next step, replace all three batteries with fresh ones. If you leave an old battery inside, it may leak. Furthermore, be careful not to insert the batteries backwards. If you do so, the contents of Continuous Memory may be lost.

5. Holding open the two plastic flags shielding the battery compartment, insert three new batteries. The batteries should be positioned with their flat sides (the sides marked +) facing *toward* the nearby rubber foot, as shown in the illustration on the calculator case.



6. Insert the tab of the battery compartment door into the slot in the calculator case.
7. Lower the battery compartment door until it is flush with the case, then push the door inward until it is tightly shut.
8. Turn the calculator on. If for any reason Continuous Memory has been reset (that is, if its contents have been lost), the display will show **Pr Error**. Pressing any key will clear this message from the display.



## Verifying Proper Operation (Self-Tests)

If it appears that the calculator will not turn on or otherwise is not operating properly, review the following steps.

For a calculator that does *not* respond to keystrokes:

1. Press the  $\boxed{y^x}$  and  $\boxed{ON}$  keys simultaneously, then release them. This will alter the contents of the X-register, so clear the X-register afterward.
2. If the calculator still does not respond to keystrokes, remove and reinsert the batteries. Make sure the batteries are properly positioned in the compartment.
3. If the calculator still does not respond to keystrokes, leave the batteries in the compartment and short both battery terminals together. (Fold back the plastic flaps to expose the terminals, which are the metal strips on either side of the battery compartment.) *Only momentary contact is required.* After you do this, the contents of the Continuous Memory will be lost, and you may need to press the  $\boxed{ON}$  key more than once to turn the calculator back on.
4. If the calculator still does not turn on, install fresh batteries. If there is still no response, the calculator requires service.

For a calculator that *does* respond to keystrokes:

1. With the calculator off, hold down the  $\boxed{ON}$  key and press  $\boxed{X}$ .
2. Release the  $\boxed{ON}$  key, then release the  $\boxed{X}$  key. This initiates a complete test of the calculator's electronic circuitry. If everything is working correctly, within about 15 seconds (during which the word **running** flashes) the display should show **-8,8,8,8,8,8,8,8,8**, and all of the status indicators (except the \* low-power indicator) should turn on.\* If the display shows **Error 9**, goes blank, or otherwise does not show the proper result, the calculator requires service.†

**Note:** Tests of the calculator's electronics are also performed if the  $\boxed{+}$  key or the  $\boxed{\div}$  key is held down when  $\boxed{ON}$  is released.†‡ These tests are included in the calculator to be used in verifying that it is operating properly during manufacture and service.

---

\* The status indicators turned on at the end of this test include some that normally are not displayed on the HP-10C.

† If the calculator displays **Error 9** as a result of the  $\boxed{ON}/\boxed{X}$  test or the  $\boxed{ON}/\boxed{+}$  test but you wish to continue using your calculator, you should reset Continuous Memory as described on page 18.

‡ The  $\boxed{ON}/\boxed{+}$  combination initiates a test that is similar to that described above, but continues indefinitely. The test can be terminated by pressing any key, which will halt

If you had suspected that the calculator was not working properly but the proper display was obtained in step 2, it is likely that you made an error in operating the calculator. We suggest you reread the section in this handbook applicable to your calculation. If you still experience difficulty, write or telephone Hewlett-Packard at an address or phone number listed under Service (page 103).

## Limited One-Year Warranty

### What We Will Do





The HP-10C is warranted by Hewlett-Packard against defects in materials and workmanship for one year from the date of original purchase. If you sell your unit or give it as a gift, the warranty is automatically transferred to the new owner and remains in effect for the original one-year period. During the warranty period, we will repair or, at our option, replace at no charge a product that proves to be defective, provided you return the product, shipping prepaid, to a Hewlett-Packard service center.

### What Is Not Covered

This warranty does not apply if the product has been damaged by accident or misuse or as the result of service or modification by other than an authorized Hewlett-Packard service center.

No other express warranty is given. The repair or replacement of a product is your exclusive remedy. **ANY OTHER IMPLIED WARRANTY OF MERCHANTABILITY OR FITNESS IS LIMITED TO THE ONE-YEAR DURATION OF THIS WRITTEN WARRANTY.** Some states, provinces, or countries do not allow limitations on how long an implied warranty lasts, so the above limitation may not apply to you. **IN NO EVENT SHALL HEWLETT-PACKARD COMPANY BE LIABLE FOR CONSEQUENTIAL DAMAGES.** Some states, provinces, or countries do not allow the exclusion or

---

the test within 15 seconds. The / combination initiates a test of the keyboard and the display. When the  key is released, certain segments in the display will be lit. To run the test, the keys are pressed in order from left to right along each row, from the top row to the bottom row. As each key is pressed, different segments in the display are lit. If the calculator is operating properly *and all keys are pressed in the proper order*, the calculator will display **10** after the last key is pressed. (The  key should be pressed both with the third-row keys and with the fourth-row keys.) If the calculator is not working properly, *or if a key is pressed out of order*, the calculator will display **Error 9**. *Note that if this error display results from an incorrect key being pressed, this does not indicate that your calculator requires service.* This test can be terminated by pressing any key out of order (which will, of course, result in the **Error 9** display). Both the **Error 9** display and the **10** display can be cleared by pressing any key.

## 102 Appendix C: Battery, Warranty and Service Information

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, province to province, or country to country.

### Warranty for Consumer Transactions in the United Kingdom

This warranty shall not apply to consumer transactions and shall not affect the statutory rights of a consumer. In relation to such transactions, the rights and obligations of Seller and Buyer shall be determined by statute.

### Obligation to Make Changes

Products are sold on the basis of specifications applicable at the time of manufacture. Hewlett-Packard shall have no obligation to modify or update products once sold.

### Warranty Information

If you have any questions concerning this warranty, please contact an authorized Hewlett-Packard dealer or a Hewlett-Packard sales and service office. Should you be unable to contact them, please contact:

- In the United States:

**Hewlett-Packard**  
Corvallis Division 1000 N.E. Circle Blvd.  
Corvallis, OR 97330  
Telephone: (503) 758-1010  
Toll-Free Number: (800) 547-3400 (except in  
Oregon, Hawaii, and Alaska)

- In Europe:

**Hewlett-Packard S.A.**  
7, rue du Bois-du-Lan  
P.O. Box  
CH-1217 Meyrin 2  
Geneva  
Switzerland

Telephone: (022) 83 81 11

**Note:** Do **not** send calculators to this address for repair.



- In other countries:

**Hewlett-Packard Intercontinental**

3495 Deer Creek Rd.  
Palo Alto, California 94304  
U.S.A.

Telephone: (415) 857-1501

**Note:** Do *not* send calculators to this address for repair.

## Service

Hewlett-Packard maintains service centers in most major countries throughout the world. You may have your unit repaired at a Hewlett-Packard service center any time it needs service, whether the unit is under warranty or not. There is a charge for repairs after the one-year warranty period.

Hewlett-Packard calculator products normally are repaired and reshipped within five (5) working days of receipt at any service center. This is an average time and could possibly vary depending upon the time of year and work load at the service center. The total time you are without your unit will depend largely on the shipping time.

## Obtaining Repair Service in the United States

The Hewlett-Packard United States Service Center for handheld and portable calculator products is located in Corvallis, Oregon:

**Hewlett-Packard Company**

Corvallis Division Service Department  
P.O. Box 999/1000 N.E. Circle Blvd  
Corvallis, Oregon 97330, U.S.A.

Telephone: (503) 757-2000

## Obtaining Repair Service in Europe

Service centers are maintained at the following locations. For countries not listed, contact the dealer where you purchased your calculator.

**AUSTRIA**

HEWLETT-PACKARD GmbH  
Kleinrechner-Service  
Wagramerstr –Lieblgasse  
A 1220 VIENNA  
Telephone (222)23 65 11

**BELGIUM**

HEWLETT PACKARD  
BELGIUM SA/NV  
Boulevard de la Woluwe 100  
Woluwelaan  
B 1200 BRUSSELS  
Telephone (2) 762 32 00

**DENMARK**

HEWLETT-PACKARD A/S  
Datavej 52  
DK 3460 BIRKEROD  
(Copenhagen)  
Telephone (02)81 66 40

**EASTERN EUROPE**

Refer to the address listed  
under Austria

**FINLAND**

HEWLETT-PACKARD OY  
Revontulentie 7  
SF 02100 ESPOO I0(Helsmki)  
Telephone (90)455 02 11

**FRANCE**

HEWLETT PACKARD FRANCE  
S.A.V. Calculateurs de Poche  
Division Informatique  
Personnelle  
F 91947 LesUlis Cedex  
Telephone (6)907 78 25

**GERMANY**

HEWLETT PACKARD GmbH  
Kleinrechner-Service  
Vertnebszentrale  
Berner Strasse 117  
Postfach560 140  
D6000 FRANKFURT 56  
Telephone (611)50041

**ITALY**

HEWLETT-PACKARD  
ITALIANA SPA  
Casella postale 3645 (Milano)  
Via G Di Vittono, 9  
20063 CERNUSCO SUL  
NAVIGLIO (Milan)  
Telephone (2)90 36 91

**NETHERLANDS**

HEWLETT PACKARD  
NEDERLAND B V  
Van Heuven Goedhartlaan 121  
NL 1181 KK AMSTELVEEN  
(Amsterdam)  
PO Box 667  
Telephone (020)472021

**NORWAY**

HEWLETT-PACKARD NORGE  
A/S  
P O Box 34  
Oesterndalen 18  
N 1345 OESTERAAS (Oslo)  
Telephone (2) 17 11 80

**SPAIN**

HEWLETT-PACKARD  
ESPANOLA S. A.  
Calle Jerez 3  
E MADRID 16  
Telephone (1) 458 2600

**SWEDEN**

HEWLETT-PACKARD  
SVERIGE AB  
Enighetsvagen 3  
Box 205 02  
S 161 BROMMA 20 (Stockholm)  
Telephone (8)730 05 50

**SWITZERLAND**

HEWLETT-PACKARD  
(SCHWEIZ)AG  
Kleinrechner-Service  
Allmend 2  
CH 8967 WIDEN  
Telephone (057)50111

**UNITED KINGDOM**  
HEWLETT-PACKARD Ltd  
King Street Lane  
Winnersh. Wokingham

GB BERKSHIRE RG11 5AR  
Telephone 734)784774

## **International Service Information**

Not all Hewlett-Packard service centers offer service for all models of HP calculator products. However, if you bought your product from an authorized Hewlett-Packard dealer, you can be sure that service is available in the country where you bought it.

If you happen to be outside of the country where you bought your unit, you can contact the local Hewlett-Packard service center to see if service is available for it. If service is unavailable, please ship the unit to the address listed above under Obtaining Repair Service in the United States. A list of service centers for other countries can be obtained by writing to that address.

All shipping, reimportation arrangements, and customs costs are your responsibility.

## **Service Repair Charge**

There is a standard repair charge for out-of-warranty repairs. The repair charges include all labor and materials. In the United States, the full charge is subject to the customer's local sales tax. In European countries, the full charge is subject to Value Added Tax (VAT) and similar taxes wherever applicable. All such taxes will appear as separate items on invoiced amounts.

Calculator products damaged by accident or misuse are not covered by the fixed repair charges. In these situations, repair charges will be individually determined based on time and material.

## **Service Warranty**

Any out-of-warranty repairs are warranted against defects in materials and workmanship for a period of 90 days from date of service.

## **Shipping Instructions**

Should your unit require service, return it with the following items:

- A completed Service Card, including a description of the problem.
- A sales receipt or other documentary proof of purchase date if the one-year warranty has not expired.

The product, the Service Card, a brief description of the problem, and (if required) the proof of purchase date should be packaged in the original shipping

case or other adequate protective packaging to prevent in-transit damage. Such damage is not covered by the one-year limited warranty; Hewlett-Packard suggests that you insure the shipment to the service center. The packaged unit should be shipped to the nearest Hewlett-Packard designated collection point or service center. Contact your dealer for assistance. (If you are not in the country where you originally purchased the unit, refer to International Service Information, above.)

Whether the unit is under warranty or not, it is your responsibility to pay shipping charges for delivery to the Hewlett-Packard service center.

After warranty repairs are completed, the service center returns the unit with postage prepaid. On out-of-warranty repairs in the United States and some other countries, the unit is returned C.O.D. (covering shipping costs and the service charge).

### **Further Information**

Service contracts are not available. Calculator product 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 service center.

## **Programming and Applications Assistance**

Should you need technical assistance concerning programming, applications, etc., call Hewlett-Packard Customer Support at (503) 757-2000. This is not a toll-free number, and we regret that we cannot accept collect calls. As an alternative, you may write to:

**Hewlett-Packard  
Corvallis Division Customer Support  
1000 N.E. Circle Blvd.  
Corvallis, OR 97330**

## **Dealer and Product Information**

For dealer information, locations, product information prices, please call (800) 547-3400. In Oregon, Alaska, or Hawaii, call (503) 758-1010.

## Temperature Specifications

- Operating: 0° to 55° C (32° to 131° F)
- Storage: -40° to 65° C (-40° to 149° F)

## Federal Communications Commission Radio Frequency Interference Statement

The HP-10C generates and uses radio frequency energy and if not installed and used properly, that is, in strict accordance with the manufacturer's instructions, may cause interference to radio and television reception. It has been type tested and found to comply with the limits for a Class B computing device in accordance with the specifications in Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference in a residential installation. However, there is no guarantee that interference will not occur in a particular installation. If your HP-10C does cause interference to radio or television reception, which can be determined by turning the calculator off and on, you are encouraged to try to correct the interference by one or more of the following measures:

- Reorient the receiving antenna.
- Relocate the calculator with respect to the receiver.
- Move the calculator away from the receiver.

If necessary, you should consult your dealer or an experienced radio/television technician for additional suggestions. You may find the following booklet prepared by the Federal Communications Commission helpful: *How to Identify and Resolve Radio-TV Interference Problems*. This booklet is available from the U.S. Government Printing Office, Washington, D.C. 20402, Stock No. 004-000-00345-4.

# Function Key Index

**ON** Turns the calculator's display on and off.

## Conversions

**→R** Converts polar magnitude  $r$  and angle  $\theta$  in X- and Y-registers respectively to rectangular  $x$ - and  $y$ -coordinates (**Page 38**).

**→P** Converts  $x$ -,  $y$ -rectangular coordinates placed in X- and Y-registers respectively to polar magnitude  $r$  and angle  $\theta$  (**Page 38**).

**→H.MS** Converts decimal hours (or degrees) to hours, minutes, seconds (or degrees, minutes, seconds) (**Page 35**).

**→H** Converts hours, minutes, seconds (or degrees, minutes, seconds) to decimal hours (or degrees) (**Page 36**).

**→RAD** Converts degrees to radians (**Page 36**).

**→DEG** Converts radians to degrees (**Page 36**).

## Digit Entry

**ENTER** Enters a copy of a number in display (X-register) into Y-register; used to separate multiple number entries (**Page 14**).

**CHS** Changes sign of number or exponent of 10 in display (X-register) (**Page 13**).

**EEX** Enter exponent; next digits keyed in are exponents of 10 (**Page 13**).

**0** through **9** Digit keys.

**.** Decimal point.

## Display Control

**FIX**  $n$  Selects fixed point display mode (**Page 49**).

**SCI**  $n$  Selects scientific notation display mode (**Page 50**).

**ENG**  $n$  Selects engineering notation display mode (**Page 51**).

## Mantissa. Pressing

**f** **CLEAR** **PREFIX** displays all 10 digits of the number in the X-register as long as the **PREFIX** key is held down (**Page 52**). It also clears any partial key sequences (refer to Clearing Prefixes, **Page 13**).

## Logarithmic and Exponential

**LN** Computes natural logarithm of number in display (X-register) (**Page 36**).

**e<sup>x</sup>** Natural antilogarithm. Raises  $e$  to power of number in display (X-register) (**Page 36**).

**LOG** Computes common logarithm (base 10) of number in display (X-register) (**Page 36**).

**10<sup>x</sup>** Common antilogarithm. Raises 10 to power of

number in display (X-register) (**Page 36**).

Raises number in Y-register to power of number in display (X-register) enter y, then x) (**Page 38**).

## Mathematics

Arithmetic operators (**Page 7**).

Computes square root of number in display (X-register) (**Page 34**).

Computes square of number in display (X-register) (**Page 34**).

Calculates  $n$  factorial ( $n!$ ) (**Page 34**).

Computes reciprocal of number in display (X-register) (**Page 34**).

Places value of  $\pi$  (3.141592654) in display (X-register) (**Page 33**).

Percent. Computes  $x\%$  of value in the Y-register (**Page 37**).

## Number Alteration

Leaves only integer portion of number in display (X-register) by truncating fractional portion (**Page 33**).

Leaves only fractional portion of number in display (X-register) by truncating integer portion (**Page 33**).

## Prefix Keys

Pressed before a function key, selects gold function printed above that key (**Page 12**).

**CLEAR** Cancels the prefix keystroke and partially entered instructions such as: . Also displays 10-digit mantissa of number in display (X-register) (**Pages 13 and 52**).

For other prefix keys, refer to listings under Display Control, Storage, and (in Programming Key Index).

## Stack Manipulation

Exchanges contents of X- and Y-

stack registers (**Page 22**).

Rolls down contents of stack (**Page 22**).

Clears contents of display (X-register) to zero (**Page 14**).

## Statistics

Accumulates statistics of numbers from X- and Y-registers in storage registers  $R_0$  through  $R_5$  (**Page 39**).

Subtracts statistics of numbers in X- and Y-registers from storage registers  $R_0$  through  $R_5$  for correcting accumulations (**Page 42**).

Computes mean (average) of x- and y-values accumulated by (**Page 43**).

Computes sample standard deviations of x- and y-values accumulated by . (**Page 43**).

Linear estimate and correlation coefficient. Computes estimated value of  $y$  ( $\hat{y}$ ) for a given value of  $x$ , or estimated value of  $x$  ( $\hat{x}$ ) for a given value of  $y$ , by least squares

method and places result in display (X-register). Computes the correlation coefficient ( $r$ ) of the linear estimate data by measuring how closely the data pairs would, if plotted on a graph, represent a straight line, and places result in Y-register (**Page 46**).

**L.R.** Linear regression. Computes  $y$ -intercept ( $B$ ) and slope ( $A$ ) for linear function  $y = Ax + B$  that best approximates  $x$ - and  $y$ -values accumulated using  **$\Sigma+$** . The value of the  $y$ -intercept is placed in the X-register; the value of the slope is placed in the Y-register (**Page 45**).

## Storage

**STO** Store. Followed by register address (0 through 9), stores displayed number in the storage register specified. Also used to perform storage register arithmetic (**Page 29**).

**RCL** Recall. Followed by address (0 through 9), recalls number from storage register specified into

the display (X-register) (**Page 29**).

**CLEAR REG** Clears contents of the stack and all storage registers to zero (**Page 30**).

**LASTx** Recalls number displayed before the current operation back into the display (X-register) (**Page 22**).

## Trigonometry

**DEG** Sets decimal degree mode for trigonometric functions—indicated by absence of **GRAD** or **RAD** annunciator (**Page 34**).

**RAD** Sets Radians mode for trigonometric functions—indicated by **RAD** annunciator (**Page 35**).

**GRD** Sets Grads mode for trigonometric functions—indicated by **GRAD** annunciator (**Page 35**).

**SIN COS TAN** Compute sine, cosine, or tangent, respectively, of number in display (**Page 35**).

**SIN<sup>-1</sup> COS<sup>-1</sup> TAN<sup>-1</sup>** Compute arcsine,

arccosine, or arctangent, respectively, of number in display (**Page 35**).




# Programming Key Index

**P/R** Program/Run mode. Sets the calculator to Program mode—**PRGM** annunciator on—or Run mode—**PRGM** annunciator cleared. Automatically sets program to line 00 when returning to Run mode (**Page 56**).

**MEM** Displays current status of program memory/storage register allocation (number of allocated program lines and number of available data storage registers) (**Page 62**).

**GTO** *nn* Go to. Used with 00 through 79. In Run mode: causes calculator to search downward in program memory for designated line number and halt. In Program mode: becomes an instruction within the program (**Page 65**).

**GTO**  *nn* Go to line number. Positions calculator to the existing line number specified by *nn* (**Page 65**).

**BST** Back step. Moves calculator back one line in program memory. Displays line number and contents of previous program line (**Page 60**).

**SST** Single step. In Program mode: moves calculator forward one or more lines in program memory. In Run mode: displays line number and contents of next program line, and executes the step (**Page 60**).

**CLEAR** **PRGM** In Program mode, clears all instructions from program memory and resets calculator to line 00. In Run mode, only resets calculator to line 00 (**Page 56**).

**PSE** Pause, Halts program execution for about 1 second to display contents of X-register, then resumes execution (**Page 66**).

**R/S** Run/Stop. Begins program execution from current line number in

program memory. Stops execution if program is running (**Page 58**).

**$x \leq y$**   **$x=0$**  Conditionals. Tests value in X-register against value in Y-register or zero as indicated. If true, calculator executes instruction in next line of program memory. If false, calculator skips one line in program memory before resuming execution (**Page 74**).

# Subject Index

Page numbers in **bold** type indicate primary references; page numbers in regular type indicate secondary references.

## A

---

Adding Program Instructions · **83–85**

Angle conversions · **35**, 38–39

Annunciators · **17**, 34, 35, 56, 58

Antilogarithms · **36**

Arithmetic

calculations, chain · **25**

calculations, simple · **14–16**

calculations, with constants · **21**

storage register · **30–31**, 67, 85

Assistance, technical · **106**

## B

---

**[BST]** · 60, 66

Backstep · **60–61**

Batteries, installing · 18, **98**

Battery life · 12, **96**, 97

Branching

adding instructions by · **85**

conditional · 71, **74–75**

simple · **71**, 86

## C

---

**[CLx]** · **7**, **14**

Calculator

service · **103–6**

shipping · **105**

temperature specifications · **107**

verifying operation · **100–101**

Can size example program · **67**

Chain Calculations · **24–25**

Change sign · **12**, 13, 33

Changing program instructions · **83**

Clearing

display · **7**, **14**

error display · **17**, 19, 70

prefixes · **13**, 52  
 programs · **56**, 61  
 statistics registers · **39**  
 storage registers · **30**  
 Conditional tests · **74–75**  
 Constant arithmetic · **21**  
 Correlation coefficient · 46, **47**, 48

## D

---

Data entry, programming · **67**, 76  
 Data entry, statistical · 39–40  
 Degrees mode · **34**  
 Degrees/Radians Conversions · **36**  
 Digit entry · 17, **91**  
 Digit separator · **17**  
 Display  
   clearing · **7**, **14**  
   error · **17**, 19, 40  
   format · 49–52  
   mantissa · 13, **52**  
   overflow and underflow · **17**, 69  
   program lines · **60–61**, 70  
   rounding · 51, **52**  
   running · **58**  
 Do if True rule · **74**

## E

---

$\boxed{\text{EEX}}$  · **13**  
 Electrical energy example program · **40**  
 Engineering notation ( $\boxed{\text{ENG}}$ ) · **51**  
 Error messages · **17**, 19, 40, 70, **94–95**  
 $\boxed{\text{ENTER}}$  · 15, 21  
 Exponents · **13**, 38, 50–52

## F

---

Factorial · **34**  
 Fixed decimal display ( $\boxed{\text{FIX}}$ ) · 7, 41, **49–50**, 79  
 Fractional portion · **33**  
 Functions  
   logarithmic · **36**  
   nonprogrammable · **70**  
   one-number · 14, **33–37**  
   primary and alternate · **12**  
   trigonometric · **35**  
   two-number · **37–39**, **37–39**

## G

---

GRAD annunciator · **35**

Grads Mode · **35**

**[GTO]** · **64, 71**, 75, 84, 85, 88

**[GTO]00** · **61–62**, 71, 75, 86, 88

## I

---

Identifying program lines · 64–65, 70

Integer portion · **33**

Internal digit representation · **49**, 51

Interrupting programs · **66**, 68–70, **70**

## K

---

Keycodes · **59–60**

## L

---

LAST X register · **25–26**, 40, 42, 47, 93

Linear estimates · **46–47**, 48

Linear regression · **45–46**

**[LASTx]** · 22

Logarithmic Functions · **36**

Looping · **71**, 74, 75–77

Low-power indicator · **8**, **12**, **97**, 100

## M

---

Mantissa · 13, 49, 50, **52**

Mean · **43**

Memory

allocation · **62–64**

continuous · **18**, 49

program · **58**, **62–64**

resetting · **18**

stack registers · **20**

**[MEM]** · 39, **62–64**

Microorganism population example problem · **27–28**

**[MEM]** · 39, **62–64**

## N

---

Negative sign entry · **13**

Nonprogrammable functions · **70**

## O

---

**ON** · 7, **12**, 17, 18, 100  
 One-number functions · **33–37**  
 Order of entry · **14**  
 Overflow · **17**, 69, 94

## P

---

**P/R** · **56**, 58, 60  
 Pausing program execution · **66**, 68  
 Percentage · **37**  
 Pi · **33**, 52  
 Polar-rectangular coordinate conversion · 38–39  
 Power  
     consumption · **96**  
     function · **38**  
     low · **8**, **12**, **97**, 100  
**Pr Error** · 19, 95, **99**  
 Prefix clearing · **13**  
 Prefix keys · **12**  
 PRGM annunciator · 18, **56**, 58  
 Primary and alternate functions · **12**  
 Product information · **106**  
 Program  
     clearing · **56**, 61  
     entry · **56**  
     instructions · **58**, 59, **83–85**  
     interrupting execution · **66**, 68–70  
     lines · 58, **64–66**  
     memory · **58**, **62–64**  
     mode · **56**, 58, 64  
     resuming execution · **66**  
     running · **57**, **90**  
     storing, multiple · **88**, **89**  
     verifying · **65–66**

## R

---

**R/S** · **58**, 74  
 RAD annunciator · **35**  
 Radians Mode · **35**  
 Radians/degrees conversion · **36**  
 Radio frequency interference statement · **107**  
 Radioisotope decay example program · **72**  
 Radix mark · **17**  
 Recalling numbers · **29**, 67  
 Reciprocal · **34**

Rectangular-polar coordinate conversion · **38–39**  
Registers, statistics · 39, **40**, 43, 44, 46  
Registers, storage · **28**, 62, 67, 85  
Repair Service · **103**  
Replacement, adding instructions by · **84–85**  
     $\boxed{\text{RCL}}$  · **64**, 78  
Roll down · **22**  
Rounding · 42, 50, **52**  
Run mode · **58**, 65  
running · **58**

## S

---

$\boxed{\text{SST}}$  · **60**, 79–82  
Sales commission example program · **77**  
Scientific notation ( $\boxed{\text{SCI}}$ ) · **50–51**  
Self-test · **100**  
Service · **103–6**  
Sign · **13**, 50  
Single step · **60**, 65  
Slope · **45**, 47  
     $\boxed{\text{STO}}$  · **64**  
Square root · **34**  
Squaring · **34**  
Stack  
    disabling · **23**, **91**  
    drop · **20**, 23, 24, 28  
    enabling · **23**, **91**  
    lift · 21, 24, 33, 43, 45, 91  
    neutral · **92**  
Standard deviation, sample · **44**  
Standard deviation, true population · **44**  
Statistics  
    accumulation · **39**  
    correction of accumulation · 42  
    error messages · **39**, 40, **94–95**  
    formulas · 43, 44, 45, **95**  
    precision · **40**  
    registers · 39, **40**, 43, 44, 45, 46  
    registers, clearing · **39**  
    Rounding error · **42**  
     $\boxed{\Sigma-}$  · 42  
     $\boxed{\Sigma+}$  · 39  
Stopping program execution automatically · **67**, 68–69  
Stopping program execution manually · **70**  
Storage register arithmetic · 30–31, 67, 85  
Storage registers · **28–32**, 62–64, 67, 85  
Storing numbers · **29**, 67

## T

---

Temperature conversion example program · **56–57**

Time conversions · **35**

T-register · 20, 28

Trigonometric Functions · **35**

Trigonometric modes · 18, **34–35**

## U

---

Underflow · **18**, 69

## W

---

Warranty · **101–3**

Water heater example program · **8–10**

## X

---

X exchange Y · 43, 46, 47

$\bar{x}$  · **46–47**

X-register · **20**, 28, 37, 40, 45, 77, 92

## Y

---

$\bar{y}$  · **46–47**

y-intercept · **45**

Y-register · 20, 37, 40, 45, 77, 92

## Z

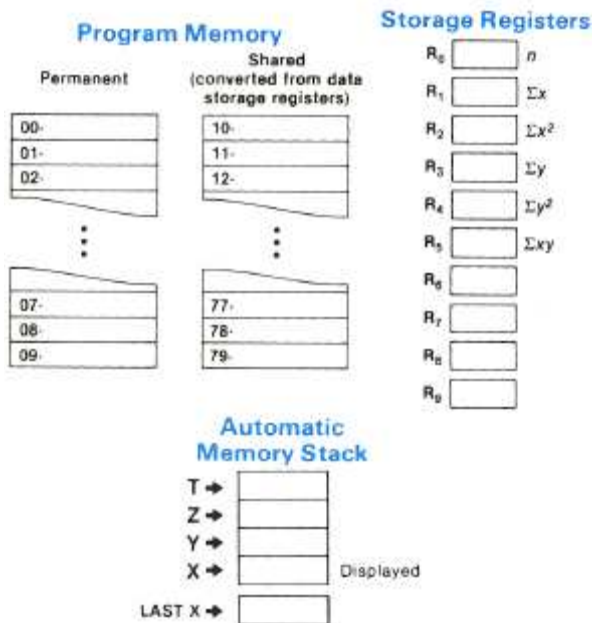
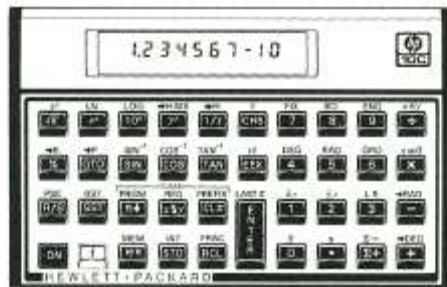
---

Z-register · 20





# The HP-10C Keyboard and Continuous Memory



The basic program memory and storage register allocation is nine lines of programming and 10 data storage registers. The calculator automatically converts one data storage register into seven lines of program memory, one register at a time, as you need them. Conversion begins with  $R_9$  and ends with  $R_0$ , giving you a maximum of 79 program lines.  $R_0$  through  $R_5$  are also used as statistics registers.



Corvallis Division

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

00010-90025

Printed in U.S.A