

# SHARP®

## POCKET COMPUTER

MODEL

# PC-E500

## OPERATION MANUAL



SHARP POCKET COMPUTER PC-E500

ENGINEER SOFTWARE  
SCIENTIFIC CONSTANTS AND FORMULAS

BUSY  
DEG CAPS  
Xmin=-500  
Xmax= 250  
Ymin=-0.004  
Ymax= 0.018

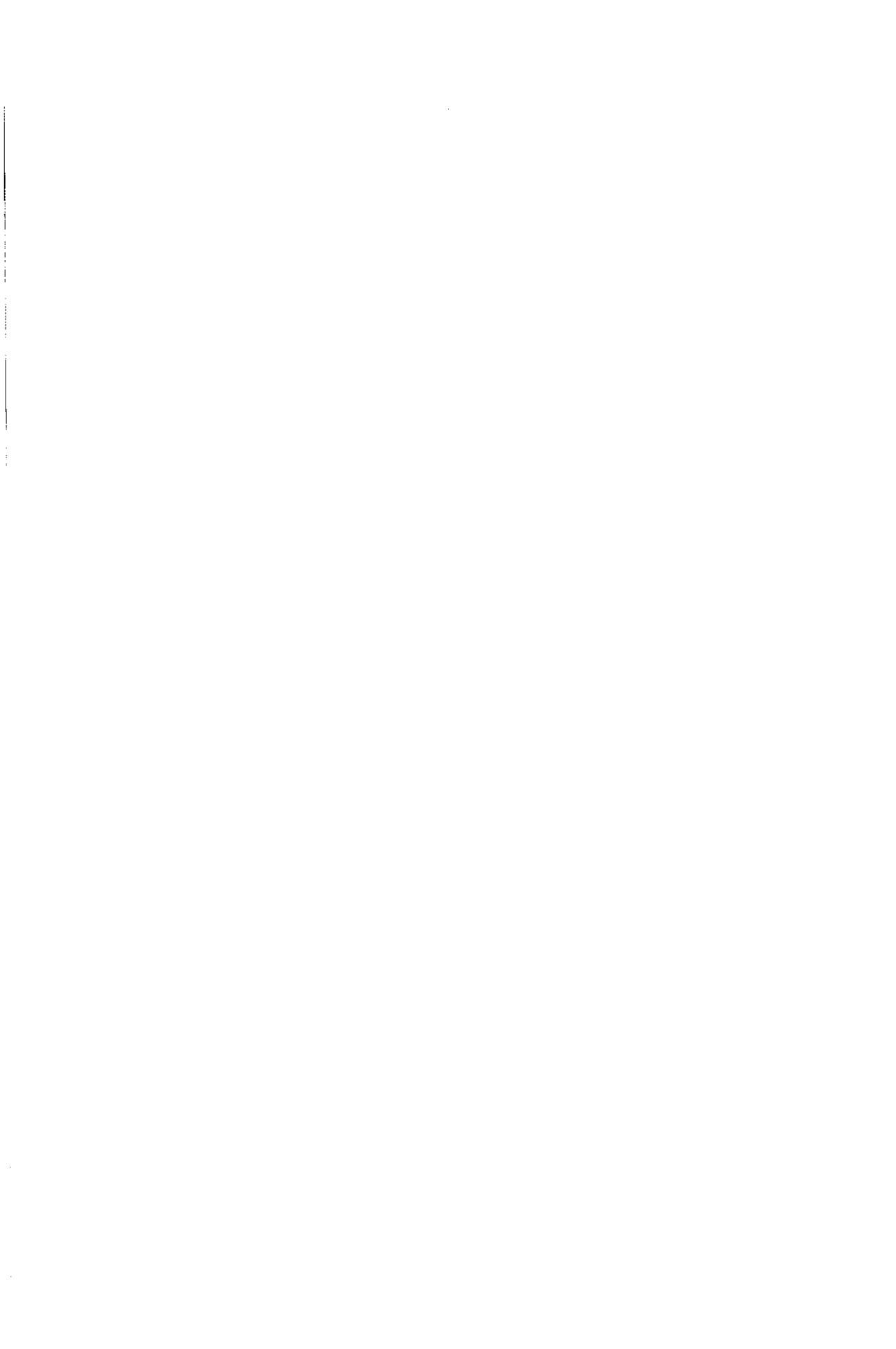
archyp	sin <sup>-1</sup>	cos <sup>-1</sup>	tan <sup>-1</sup>	TAB	CA
hyp	sin	cos	tan	FSE	OFF
-DEC	-DMS	°'	10	+r6	↑
-HEX	-DEG	ln	log	1/x	↓
T	y <sup>x</sup>	%	+	+xy	n!
EXP	√	x <sup>2</sup>	(	)	

7	8	9	÷/	DEL
4	5	6	X*	BS
1	2	3	-	M+
0	+/-	.	+	INS
				=

BASIC PF1  
CAL PF2  
MATRIX PF3  
STAT PF4  
ENG PF5

Q W E R T Y U I O P  
A S D F G H J K L  
Z X C V B N M  
SPACE

2ndF STO RCL P←NP  
PLAY BACK



# INTRODUCTORY NOTES

Welcome to the world of SHARP owners!

Few industries today can match the rapid growth and technological advances being made in the field of personal computing. Computers that just a short time ago would have filled a huge room, required a Ph.D. to program, and cost thousands of dollars, now fit in the palm of your hand, are easily programmed, and cost so little that they are within reach of nearly everyone.

Your new SHARP PC-E500 was designed to bring you state-of-the-art technology. It incorporates many advanced features:

- Programmable function keys simplify mode selection: You can choose the desired mode just by pressing the corresponding programmable function key on the Main Menu.
  - Function calculation: You can perform function calculations with the ease and efficiency of a function computer.
  - Statistical and Regression Calculations: You can perform single-variable statistical calculations or linear regression calculations. Stored data calculation lets you confirm or correct data entry.
  - Matrix Calculations: You can compute simultaneous linear equations (with  $n$  unknowns), inverse matrix, determinant, and transposed matrix calculations.
  - Engineer Software: The Engineer Software resident in the PC-E500 lets you recall mathematical, scientific, engineering and statistical constants, formulas, data, and functions and do calculations using those constants and formulas. The items you want can be easily recalled from menus. You can also include a program you create as part of the Engineer Software and access it later from a menu.
  - Algebraic Expression Reserve (AER) Memory: Formulas or constants you frequently use can be stored in memory and conveniently recalled for repeated use.
- High-precision calculations with 20 significant digits: Double-precision calculation capability makes the PC-E500 suitable for application software requiring high computation accuracy.
- Existing software resources are available: Software packages developed for our previous pocket and portable computers are compatible with the PC-E500.
- Easy-to-use editing features: Large, single-function edit keys (**DEL**, **BS**, **INS**) for data correction or deletion make data editing easier. In addition, the AUTO, RENUM, and DELETE commands simplify program editing.
- Optional pocket disk drive (Model CE-140F): Enables you to write or read a program or data quickly and easily, compared to tape recorders. You can also use the COPY command for simple data backup. The CE-140F is battery powered.
- RAM disks let you store programs and data just like a diskette:  
Part of the internal memory can be used as a RAM disk, which lets you save and load your programs and data just as you would using a diskette. An optional RAM card, if installed in the PC-E500, gives you additional space for RAM disks, data memory, and program memory (see page 18).

- A serial I/O interface: Allows direct computer-to-computer communication and also attachment of the versatile CE-515P color plotter/printer.

Congratulations on entering an exciting and enjoyable new world. We are sure that you will find this purchase one of the wisest you have ever made. The SHARP PC-E500 is a powerful tool, designed to meet your specific mathematical, scientific, engineering, business and personal computing needs. With the SHARP PC-E500 you can begin NOW to provide the solutions for tomorrow!

## **NOTICE**

- SHARP strongly recommends that separate permanent written records be kept of all important data. Data may be lost or altered in virtually any electronic memory product under certain circumstances. Therefore, Sharp assumes no responsibility for data lost or otherwise rendered unusable whether as a result of improper use, repairs, defect, battery replacement, use after the specified battery life has expired, or any other cause.
- SHARP assumes no responsibility, directly or indirectly, for financial losses or claims from third persons resulting from the use of this product and all of its functions, such as the loss of or alteration of stored data, etc.
- The information provided in this manual is subject to change without notice.

# TABLE of CONTENTS

INTRODUCTORY NOTES .....	i
USING THE PC-E500 FOR THE FIRST TIME .....	1
USING THIS MANUAL .....	4

## PART 1 HARDWARE

1. HARDWARE: OVERVIEW .....	8
2. HARDWARE: IN DETAIL .....	10
Turning On .....	10
Auto OFF .....	11
Key Notation in this Manual .....	11
The Display .....	12
Battery Replacement .....	14
3. RAM CARD .....	18
Number of Files .....	18
Installing or Replacing the RAM Card .....	18
RAM Card .....	20
How To Use the RAM Card .....	21
4. PERIPHERAL DEVICES .....	24
CE-140F Pocket Disk Drive .....	24
CE-126P Thermal Printer/Cassette Interface .....	24
CE-515P Color Plotter/Printer (Color Printer) .....	25
CE-124 Cassette Interface .....	25
Using the Cassette Interface .....	25
Using the CE-126P Printer/Cassette Interface .....	27
Using the CE-515P Color Plotter/Printer .....	28
Serial I/O Function .....	29

## PART 2 DIRECT INPUT OPERATION

5. OPERATION: A QUESTION OF MODES .....	32
Selecting Modes .....	32
6. CAL MODE .....	33
Calculations .....	33
Basic Operations .....	37
Scientific Calculations .....	39
Use of Parentheses .....	42
Decimal Places .....	42
Priority Levels .....	43
Hex Conversion and Calculations .....	45
7. ENGINEER SOFTWARE MODE .....	49
Engineer Software List .....	51
How To Create an Engineer Software Program .....	115

8. AER MODE .....	117
Programming and Recalling Expressions .....	117
Correcting and Deleting Expressions .....	119
Executing an Expression .....	121
Searching Expression Titles .....	122
Error Messages .....	122
9. STAT MODE .....	123
Calculating Statistics .....	123
Linear Regression Calculations .....	127
Variable Storage .....	136
Errors and Error Messages in Statistical and Regression Calculations ...	137
10. MATRIX MODE .....	138
Entry of Matrix Element .....	138
Matrix Operations .....	141
Scalar Operations .....	144
Example of Matrix Operations .....	145
Variable Storage .....	147
Printing a Matrix .....	148
Error Messages .....	148
11. RUN MODE .....	149
Selecting RUN Mode .....	149
Some Helpful Hints .....	149
Simple Calculations .....	150
Compound Calculations and Parentheses .....	151
Recalling Entries .....	151
Errors .....	153
Serial Calculations .....	154
Using Variables in Calculations .....	155
Single-Precision, Double-Precision .....	156
Last Answer Feature .....	156
Maximum Calculation Length .....	158
Scientific Calculations .....	158
Priority in Direct Input Calculations .....	162
Printing for Direct Input Calculations .....	162
Calculation Errors .....	163

## **PART 3 PROGRAM OPERATION**

12. CONCEPTS AND TERMS OF BASIC .....	166
String Constants .....	166
Hexadecimal Numbers .....	166
Variables .....	167
Program and Data Files .....	174
Filenames .....	174
Extension .....	174
Device Name .....	175
File Numbers .....	175

Data Files .....	176
Expressions .....	176
Numeric Operators .....	176
String Expressions .....	176
Relational Expressions .....	177
Logical Expressions .....	177
Parentheses and Operator Precedence .....	179
<b>13. PROGRAMMING .....</b>	<b>180</b>
Programs .....	180
BASIC Statements .....	180
Line Numbers .....	180
Labelled Programs .....	181
BASIC Commands .....	181
Direct Commands .....	182
Modes .....	182
Beginning to Program .....	182
Storing Programs in Memory .....	188
Data Files .....	189
Programmable Function Keys .....	191
<b>14. DEBUGGING .....</b>	<b>193</b>
Debugging Procedures .....	194

## **PART 4 BASIC REFERENCE**

<b>15. SCIENTIFIC &amp; MATHEMATICAL CALCULATIONS .....</b>	<b>198</b>
Calculation Ranges .....	207
<b>16. BASIC COMMAND DICTIONARY .....</b>	<b>209</b>

## **PART 5 APPENDICES**

<b>A ERROR MESSAGES .....</b>	<b>340</b>
<b>B CHARACTER CODE CHART .....</b>	<b>342</b>
<b>C KEY FUNCTIONS IN BASIC .....</b>	<b>344</b>
<b>D TROUBLESHOOTING .....</b>	<b>348</b>
<b>E SIGNALS USED IN THE SERIAL I/O INTERFACE .....</b>	<b>350</b>
<b>F SPECIFICATIONS .....</b>	<b>351</b>
<b>G USING PROGRAMS FROM OTHER SHARP COMPUTERS .....</b>	<b>353</b>
<b>H CARE OF THE PC-E500 .....</b>	<b>355</b>

**COMMAND INDEX**  
**INDEX**





# USING THE PC-E500 FOR THE FIRST TIME

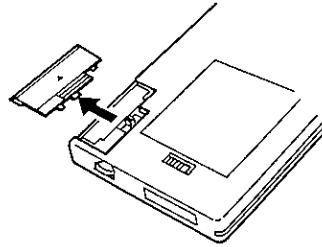
## Installing Batteries

The PC-E500 uses two types of batteries: four AAA dry batteries to power the computer itself, and a lithium battery for memory backup.

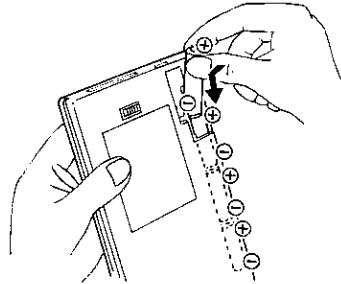
Install the four operating dry batteries and one memory backup lithium battery supplied as follows:

### Installing dry batteries

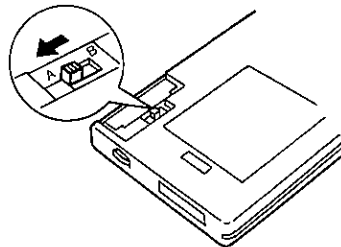
1. Remove the battery compartment lid on the back of the PC-E500.



2. Insert the batteries properly with the negative side first.



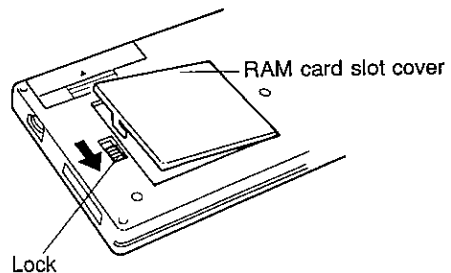
3. Be sure that the Memory Protect Switch is in position A.



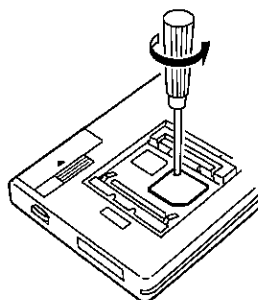
4. Replace the battery compartment lid.

### Installing the lithium battery

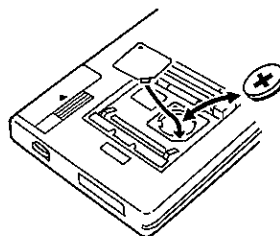
1. Slide the lock on the back to the side opposite to the LOCK position and remove the RAM card slot cover.



- Using a Philips screwdriver, remove the screw retaining the battery compartment lid (turn over the PC-E500, and the lid will drop off).



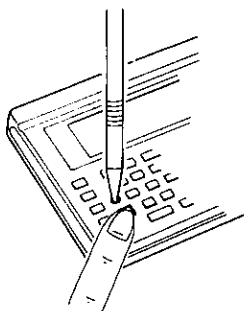
- Insert the supplied lithium battery, after wiping it with a dry cloth, with the positive side facing up. When installing, insert the edge of the battery under the claw.
- Replace the battery compartment lid and secure it with the screw.
- Replace the RAM card slot cover and slide the lock to the LOCK position.



## Resetting

Just after you have installed the batteries into the PC-E500, the internal status is indefinite. You have to initialize the PC-E500 to make it ready for use:

- While holding down the **ON** key, press the RESET button just beside the **BASIC** key with a ball-point pen or any other appropriate device. Release the RESET button before releasing the **ON** key.



Use only a ball-point pen or other similar device to press the RESET button. Do not use a mechanical pencil with its lead exposed or a device with a sharp point, such as a sewing needle.

Immediately after the RESET button has been pressed, the PC-E500 will display either of the following displays, A or B. If any other display appears, perform the above operation again.

The PC-E500 asks you to confirm that you wish to clear the memory.

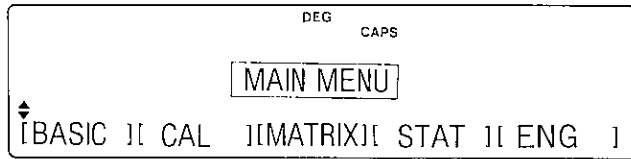
(Display A)

```
S1(MAIN):NEW CARD
PF1 --- INITIALIZE
```

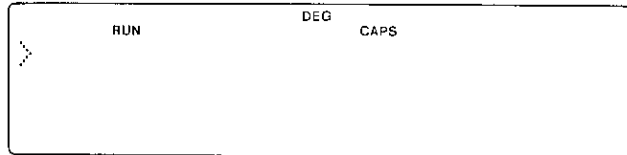
(Display B)

```
S1(MAIN):
ALL CLEAR OK? (Y/N)
```

2. If display A appears, press the **PF1** key. If display B appears, press the **Y** key. The following Main Menu will be displayed. This display also shows that initialization is complete and the entire memory within the PC-E500 is cleared:

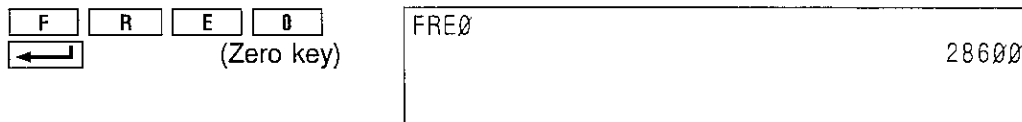


3. Press the **PF1** key, and the system will display the following BASIC mode display:



## Checking for Normal Computer Operation

To confirm that the PC-E500 is functioning normally, enter:



If the display appears as shown above, the PC-E500 is functioning normally and is in the Command Prompt mode.

The number "28600" indicates the memory capacity available for program or data.

### Note:

If the PC-E500 fails to display as indicated in any of the steps above, read the description of the relevant step again and retry correct operation for that step.

# USING THIS MANUAL

We designed this manual to introduce you to the capabilities and features of your PC-E500 and to serve as a reference tool. It has been divided into five parts, each of which introduces a particular aspect of the PC-E500. Although the manual is intended ultimately as a reference, we suggest that you carefully read the introductory chapters of Part 1 (hardware) and Part 2 (operation) before operation. The PC-E500 is a powerful tool and has many valuable and time-saving functions that even the seasoned computer buff will be pleased to discover!

## **Part 1: Hardware**

The first chapter of Part 1 provides an introduction to the features of the PC-E500. Chapter 2 describes the basic handling of the computer, the operation of keys, and the meanings of various display symbols. Both Chapters 1 and 2 are essential reading. Subsequent chapters deal with RAM card memory and peripheral devices (such as printers and disk drives).

## **Part 2: Direct Input Operation**

Part 2 is devoted to use of the PC-E500 as a calculator (scientific, statistic and matrix) or "direct input" computer. Direct input refers to the independent use of BASIC commands (that is, commands not within a BASIC program).

## **Part 3: Program Operation**

Part 3 introduces the BASIC programming language as implemented on the PC-E500. Even if you have programmed in BASIC before, we hope you will read this part thoroughly, since the BASIC language has many dialects. This part also contains time-saving information in the form of programming shortcuts and debugging techniques.

## **Part 4: Basic Reference**

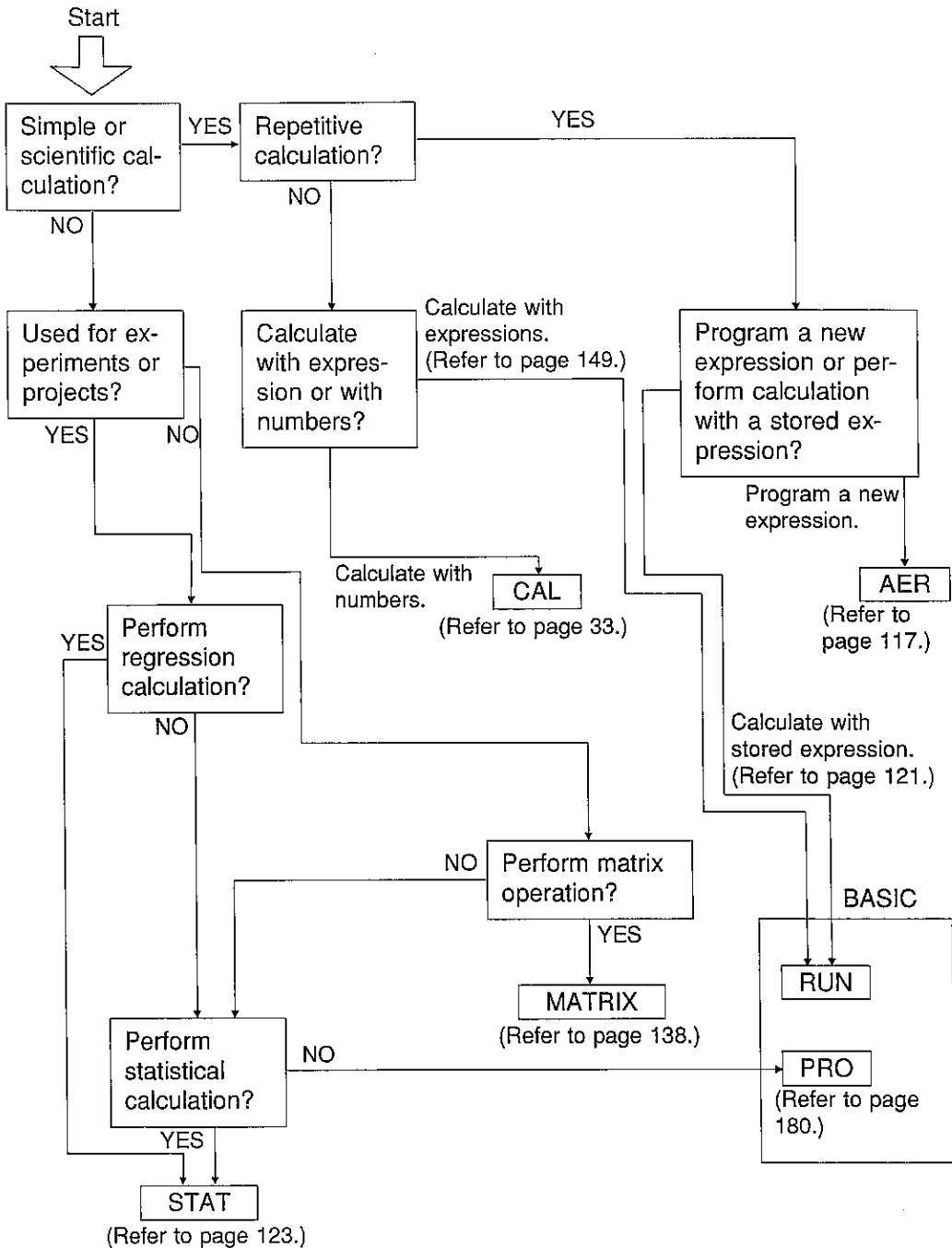
Part 4 is an alphabetic listing of the numeric functions and BASIC commands used in programming the PC-E500. Many of these commands can be used in the direct input modes of the PC-E500.

## **Part 5: Appendices**

Part 5 contains mainly reference material such as code tables, error messages and specifications. You will also find tips on how to keep your PC-E500 in good condition.

This manual is not intended to be a self-teaching course in BASIC, the complete description of which is beyond the scope of this manual. If you have never programmed in BASIC, you should buy a separate book or attend a class on the subject before trying to work through this manual.

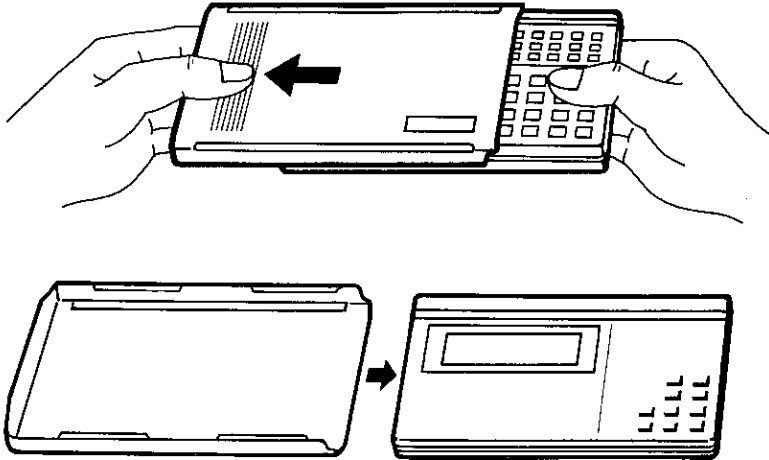
To use the Engineer software refer to page 49. Follow the flowchart to use the appropriate operation mode



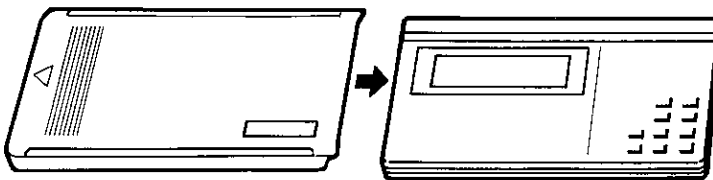
### The Protective Cover

Your computer is supplied with a cover to protect the operation panel when the computer is not being used.

- When the computer is to be used, remove the protective cover from the computer as shown below.



- When the computer is not being used, slide the protective cover over the operation panel, as shown below.



# **HARDWARE**

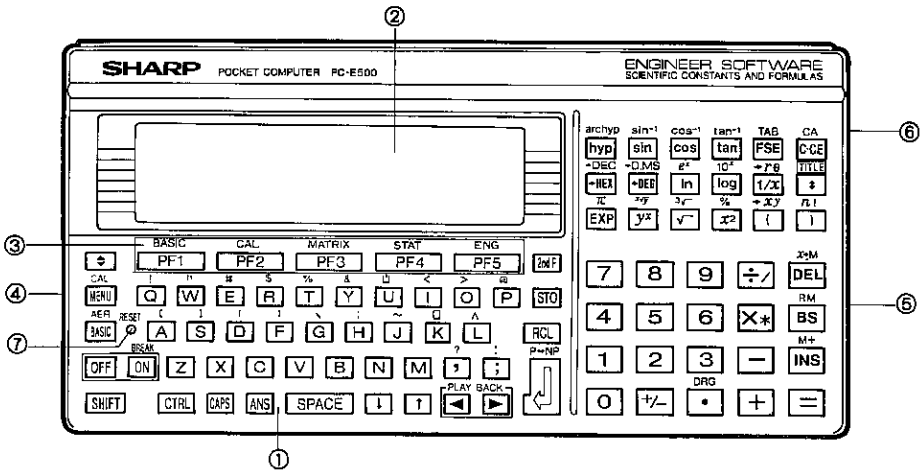
The first chapters of Part 1 introduce the features of the PC-E500\*, describe the handling of the computer, the operation of the keys, and the meanings of the various display symbols. Subsequent chapters describe the use of RAM cards and peripheral devices (such as the disk drive, the printer and the cassette tape recorder).

\* The PC-E500 is hereafter referred to as "the computer".

# 1. HARDWARE: OVERVIEW

The SHARP computer features a QWERTY-style keyboard layout that is similar to conventional typewriters, a liquid crystal display (LCD) with adjustable contrast, a RAM slot, and an 11-pin interface connector and a 15-pin serial I/O connector for attaching various optional equipment.

The following pages describe the individual parts of the computer to acquaint you with their location and functions.



- The **ON** and **OFF** keys have been recessed into the keyboard so that they are not pressed by mistake.
- Pressing the **O** key while holding the **CTRL** key will toggle the beep sound for key entry. Setting the beep while in the BASIC mode is effective in other operation modes.

## 1. Keyboard

With its typewriter-style layout and numeric keypad, the keyboard has 89 keys, including a number of special-purpose keys.

## 2. LCD Screen

The display has four lines of 40 characters each. The contrast of the extra-large characters is fully adjustable for comfortable viewing.

## 3. Programmable Function Keys

These keys are used to select operation modes, programs, BASIC statements, or other user-assigned functions.

## 4. 11-Pin Connector

Use the 11-pin connector to link the computer directly to optional peripheral equipment such as the disk drive (CE-140F), the cassette tape recorder and the printer (CE-126P), or to the optional interface unit (CE-124).



### 5. 15-pin Serial I/O Connector

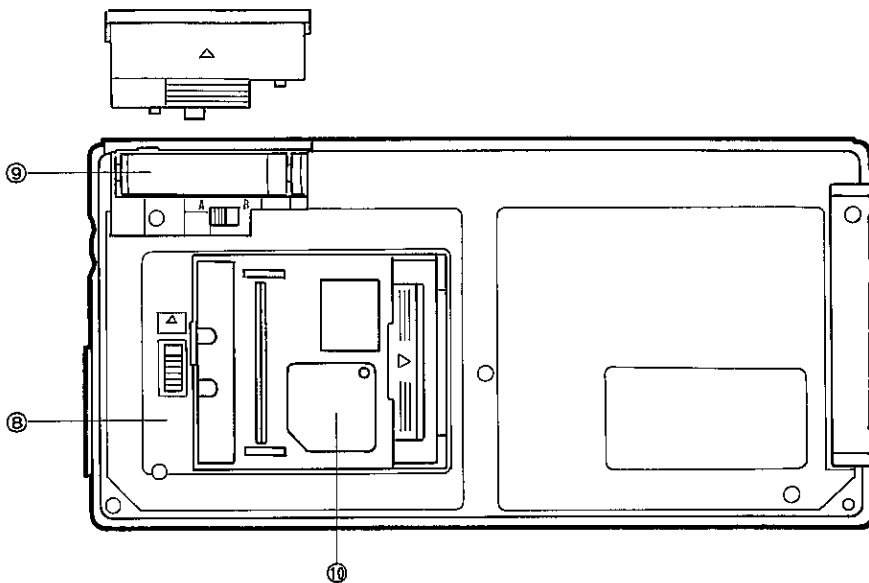
Use the Serial I/O connector to link the computer directly to optional peripheral equipment such as the color plotter/printer (CE-515P).

### 6. LCD Contrast Dial

Adjust this dial to lighten or darken the screen to suit the viewing conditions.

### 7. RESET Button

If the computer "hangs-up" for some reason during operation, the keyboard will be inoperative. It may be impossible to use the power switch or **BREAK** key to restart the computer. Press the RESET button to clear the memory and restore the computer to the condition at power-on. Use this switch with caution as you risk losing data and programs. See Appendix D for details on the use of this button.



### 8. RAM Card Slot

This slot houses an external RAM card which provides additional memory workspace and supports RAM disks. RAM cards of different capacities can be installed here. See Chapter 3 for details on how to use RAM cards.

### 9. Operating Battery Compartment

This compartment houses the four dry batteries required to power the computer. See Chapter 2 for details on replacing batteries.

### 10. Backup Battery Compartment

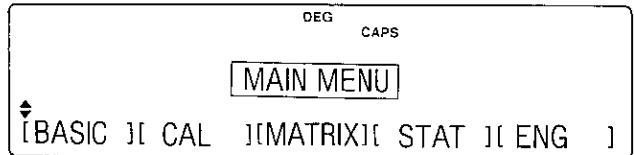
This compartment houses one lithium battery required to backup the internal memory.

## 2. HARDWARE: IN DETAIL

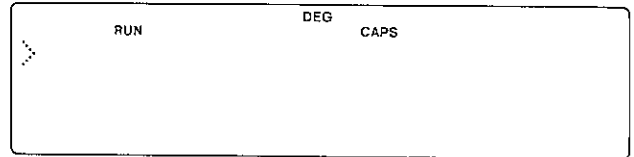
### Turning On

Press the **ON** key in the left-hand area of the computer keyboard. The computer will be turned on in one of the following three modes depending on the last mode selected:

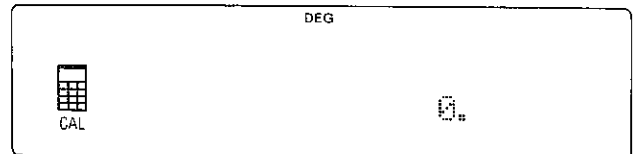
1. If the computer was last turned off in the Main Menu, MATRIX, STAT, or ENG mode, it enters the Main Menu when turned on.



2. If the computer was last turned off in the RUN, PRO, or AER mode, it enters the RUN mode when turned on.



3. If the computer was last turned off in the CAL mode, it returns to the CAL mode when turned on.



#### Note:

If the **BATT** warning indicator appears in the upper right corner of the display, it indicates that the battery has almost run out. Immediately save all stored program or data to a cassette tape or other permanent storage medium, then replace the batteries (see pages 14 and 25).

When the computer is turned on or off, you may see the display blacked out or irrelevant dots, line, or symbols appearing on it momentarily, but these are not unusual.

## Auto OFF

To preserve battery power, the computer automatically turns itself off if no key has been pressed for about 11 minutes. However, the Auto OFF feature takes effect in about one minute if the RESET message (see page 2) is currently being displayed.

Press the **ON** key to repower the computer if it has turned itself off.

The Auto OFF feature remains inactive when the computer is executing the INKEY\$ or INPUT\$ command. However, it is active when the computer is executing the INPUT command.

If the computer is left unused for a long period of time with the Auto OFF feature inactive, battery power will eventually be consumed, causing any stored programs or data to be lost.

## Key Notation in this Manual

**BREAK** **ON** : Turns the power on.  
**BREAK** : Interrupts the operation or calculation.

**CAL** **MENU** : Displays the Main Menu.  
**SHIFT** + **CAL** : Sets the computer in the CAL mode.

**sin<sup>-1</sup>** **sin** : sin key  
**2nd F** **sin<sup>-1</sup>** : sin<sup>-1</sup> key


When numerals, characters, or symbols printed on the keys or just above each key are referred to in this manual, only the pertinent letters will appear, with key boxes or the **SHIFT** key omitted, as shown in the following example:

**S** **H** **A** **R** **P** → SHARP  
**SHIFT** + **R** **4** **5** → \$45

### The Difference between the **SHIFT** and **2nd F** Keys

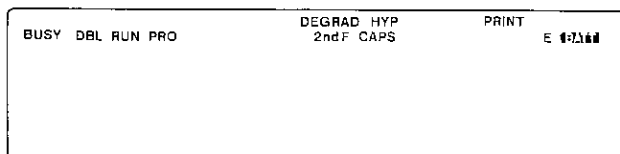
The **SHIFT** and **2nd F** keys are functionally identical. The only difference is that the **SHIFT** key must be pressed and held down when you press any other key, whereas the **2nd F** key allows you to press and then release it before you press any other key. In this manual, the **2nd F** key is used in the description of calculations in the CAL mode and calculations of functions, whereas the **SHIFT** key is used for all other descriptions.

- Where a space must be entered with the **SPACE** key, it is indicated by symbol **␣** in this manual, for example:  
"SHARP␣EL-865␣WN-104"

Keys may appear in their full boxed images in this manual, such as , whenever needed.

To discriminate the number zero from the capital letter "O," the zero appears as "0" on the PC-E500's display. In the descriptions in this manual, the number zero will also appear as "0" whenever it may be confusing.

## The Display






The computer has a four line 40-character dot-matrix liquid crystal display with a status line above. Each character occupies a 5 × 7 dot matrix. The display shows menu labels, key labels, and calculation processes.

In BASIC mode the display shows:

- > The Prompt. This symbol appears when the computer is waiting for input in the BASIC mode. As you type, the prompt disappears and is replaced by the cursor.
- └─█ The Cursor. This symbol tells you the location of the next character to be typed in. As you begin typing, the cursor replaces the prompt. The cursor can be positioned over particular characters when using the INSert and DELete functions. The block cursor changes to an underline cursor when not positioned over an existing character.

The status line consists of:

- BUSY** Displayed while the computer is executing a program or command.
- DBL** In BASIC modes, this tells you that double-precision calculation mode has been specified, and that up to 20 digits can be used.
- 2ndF** Displayed when the  key is pressed and disappears with subsequent key entry. Remember, the  key must be released before pressing any other key if that key's second function is to be used.
- CAPS** Indicates that the computer is in the CAPS Lock mode, in which all alphabetic characters are entered in uppercase. When this indicator is not on the display, all alphabetic characters are entered in lowercase. The  key lets you select or deselect the CAPS Lock mode.
- DEG** Indicates that the Degree mode is selected for angular functions.
- RAD** Indicates that the Radian mode is selected for angular functions.
- GRAD** Indicates that the Gradient mode is selected for angular functions.

- HYP**      Displayed when the **hyp** key is pressed, indicating that a hyperbolic function has been selected. If **2nd F** **hyp** are pressed, a phrase, 2ndF HYP, comes on to indicate that an inverse hyperbolic function has been selected.
- E**            Indicates that an ERROR has occurred.
- BATT**       Indicates that the operating batteries are low and should be replaced as soon as possible.
- RUN**        Indicates that the computer is in the RUN mode.
- PRO**        Indicates that the computer is in the PROgramming mode.
- PRINT**      Indicates the computer is ready to send data to the printer in the RUN mode. Press **SHIFT** + **PRINT** keys to toggle on and off. (Only available when the optional printer is connected.)

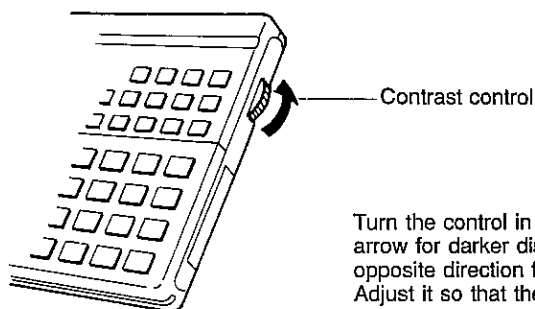
**Note:**

Japanes symbols “カナ” and “小文字” may appear faintly on the top right of the display. These symbols do not affect any function or operation of the computer.

**For Display Exceeding 4 Lines**

The display consists of 4 lines (40 character per line). Key entries or calculated results are displayed from the top line of the display. If the characters to be displayed exceed 4 lines, the displayed contents will be moved up by 1 line (the first displayed line will move off the top of the display).

**Contrast Control**



Turn the control in the direction of the arrow for darker display, and turn it the opposite direction for lighter display. Adjust it so that the display is easy to see.

# Battery Replacement

The computer uses two types of batteries - i.e. four Type-AAA dry batteries for powering the computer itself, and a lithium battery for memory backup.

If the operating dry batteries are discharged when the CE-126P printer is used with the computer, power will be supplied from the CE-126P, so the current drawn from the operating batteries will be reduced.

## Operating Battery Replacement Timing

If the **BATT** indicator appears in the upper right corner of the display, it indicates that the operating batteries are discharged. Immediately replace them with new ones. If you continue to use the computer with the **BATT** warning on the display, the computer will eventually turn off. After this, the computer cannot be turned on even if you press the **ON** key.

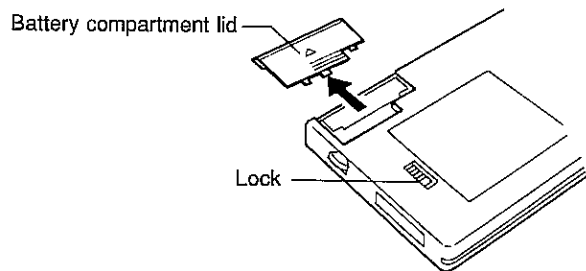
If an optional peripheral device is connected to the computer, the computer power may be supplied from the device. Note that, in this case, the **BATT** warning will remain off even if the computer's operating batteries are discharged. Before use, temporarily disconnect the peripheral device and check to make sure that the **BATT** warning does not appear on the display.

- While you are replacing the dry batteries, the memory is backed up by a lithium battery. However, if you have purchased an optional peripheral device, we recommend that you save the program and data from computer's internal memory to the storage device.

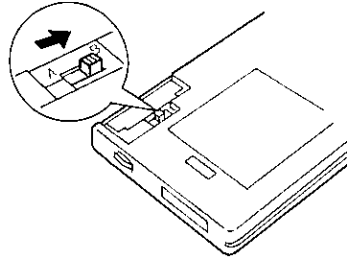
## Replacing Operating Batteries

If the dry batteries require replacement, follow the replacement procedure below. If the procedure is not followed properly, the computer may remain inoperative or the life of the memory backup battery may be shortened.

- Prepare the following types of batteries:  
Operating battery: Four Type-AAA dry batteries  
Memory backup battery: One lithium battery (CR2016)
1. Press **ON** then **OFF** to turn the computer off.
  2. If a RAM card is installed in the computer, remove it by referring to the descriptions on page 18. If you replace the batteries with the RAM card left installed in the computer, all the contents of the RAM card memory will be erased.
  3. Remove the battery compartment lid.



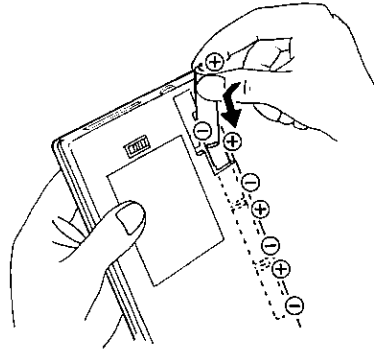
- Slide the Memory Protect Switch to position B.



**Caution:**

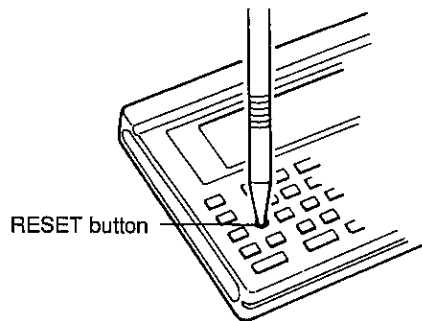
If you replace the batteries with the Memory Protect Switch left at position A, the contents of the memory will be erased.

- Remove the used batteries from the compartment and insert the new batteries into it, after wiping each battery with a dry cloth. Insert the negative side first.



Replace all four batteries at one time.

- Press the RESET button. Do not reinstall the RAM card at this point.



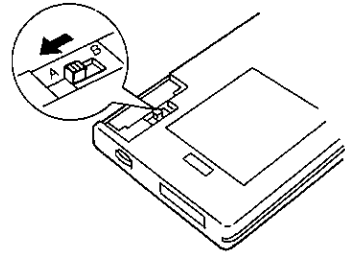
NO MEMORY

The message above will then appear and the computer will turn off in about 3 seconds. If no message or any other message appeared, press the RESET button again.

**Note:**

No message will appear if the lock at the back is not in the LOCK position.

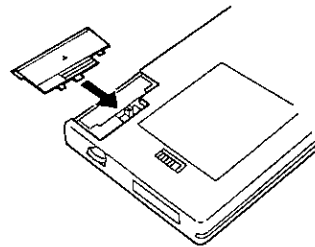
7. Install the RAM card by referring to the description on page 18. After installing it, be sure to slide the lock to the LOCK position. Otherwise, the computer will not operate.
8. Slide the Memory Protect Switch back to position A.



**Note:**

The computer will remain inoperative unless the Memory Protect Switch is set to position A.

9. Replace the battery compartment lid.



**Backup Battery Replacement Timing**

The memory backup battery has a backup capacity of about 2 years at room temperature (20°C/68°F). The contents of the memory remain intact for this duration even when the operating batteries are discharged or are being replaced. Note, however, that the backup battery will discharge quickly if the operating batteries are left discharged or are removed.

**Note:**

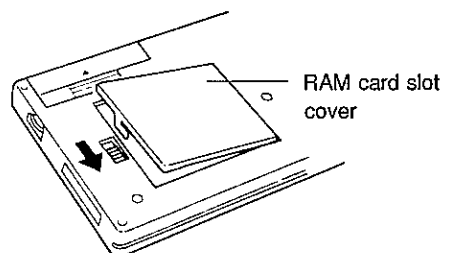
The battery life may be shortened under harsh operating environments with, for example, temperature extremes.

**Replacing the Memory Backup Battery**

When replacing the memory backup battery, check to make sure that the four dry batteries for the computer are not low (no **BATT** warning appears). If they are low, first replace the four dry batteries before replacing the backup battery. If you remove the backup battery when the dry batteries are low, the contents of the memory may be lost.

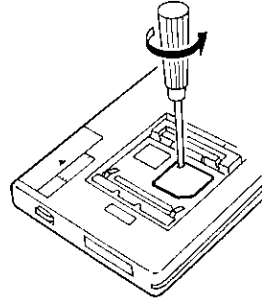
Follow the replacement procedure below:

1. Press **ON** then **OFF** to turn the computer off.
2. Slide the lock at the back of the computer to the side opposite to the LOCK position and remove the RAM card slot cover.

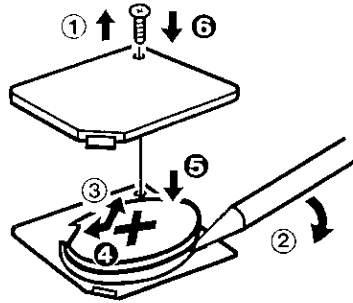




3. If a RAM card is installed in the computer, remove it.
4. Using a Philips screwdriver, remove the screw retaining the battery compartment lid and remove the lid (turn over the computer, and the lid will drop off).



5. Remove the used battery from the compartment, and insert the new lithium battery into it, after wiping it with a dry cloth. Insert the battery with the positive side facing up. When installing, insert the edge of the battery under the claw.



6. Replace the battery compartment lid and secure it with the screw.
7. Reinstall the RAM card if you have removed it.
8. Replace the RAM card slot cover and slide the lock to the LOCK position.

### Battery Handling Notes

Batteries, if misused, can explode or cause electrolyte leakage. Pay special attention to the following points:

1. Be sure to replace all four dry batteries at the same time.
2. Do not use new batteries with used batteries in the same unit.
3. Replacement batteries should be of the same type as those to be replaced.  
Some types of batteries are rechargeable, while other types are not. Carefully read the type description on the battery and choose the un rechargeable type.
4. Insert the batteries in the correct orientations as indicated in the battery compartments.

### Caution:

- **Keep batteries out of the reach of children.**
- Remove used batteries from the compartment. Otherwise, the computer may be damaged from electrolyte leakage.
- Do not throw batteries into a fire as this may result in an explosion.

## 3. RAM CARD

The computer has a standard 32K byte internal RAM for storing programs and data. This is large enough for most programs and calculations, but can be expanded by the addition of an optional RAM card in the slot on the underside of the computer. RAM cards are available with capacities of 8K bytes to 64K bytes. All contain a lithium battery to back up their contents even if removed from the computer, and so can be used as a fast alternative to a disk drive or cassette tape. Each card can be configured to store data or programs or both, and can be installed in or removed from the computer at will (see MEM\$ and FRE for configurations, and INIT for initialization of the card).

### Number of Files

RAM cards can be used to store either program or data files, or both. The number of files that can be stored on any one card is limited by the size of the files and the space automatically allotted on each card for its directory. The following list gives the maximum number of files (program and data) that can be stored on an initialized RAM disk F.

Disk size	No. of files
2K bytes	5
3 – 4K bytes	8
5 – 8K bytes	16
9 – 12K bytes	24
13 – 16K bytes	32
17 – 20K bytes	40
21 – 24K bytes	48
25 – 28K bytes	56
29K bytes or more	64

### Installing or Replacing the RAM Card

- The RAM card, once installed, should not be removed except when it is to be replaced with another RAM card or when its battery requires replacement.
- After you have installed a new RAM card or reinstalled a RAM card whose back-up battery has been replaced, be sure to carry out the following initialization:

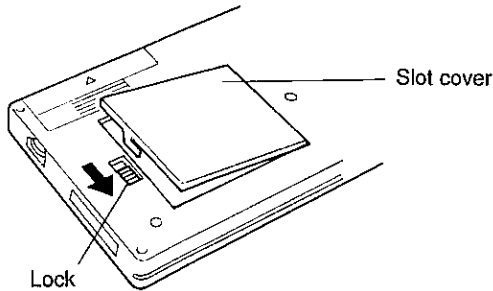
First press the **ON** key to turn the computer on, and the following messages will appear:

```
S2 (CARD) : NEW CARD
PF1 --- INITIALIZE
PF2 --- DO NOT INITIALIZE
```

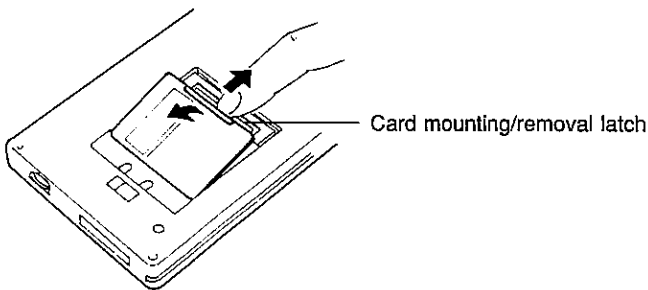
Press the **PF1** key to ready the RAM card for use in the computer.

The following describes how to install or replace the RAM card:

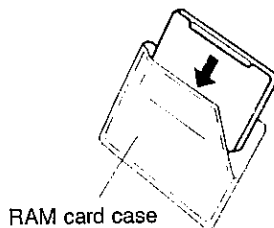
1. Press **ON** then **OFF** to switch the computer off.
2. Open the RAM card slot by sliding the lock and remove the slot cover. If no RAM card is in the slot, go to step 4.



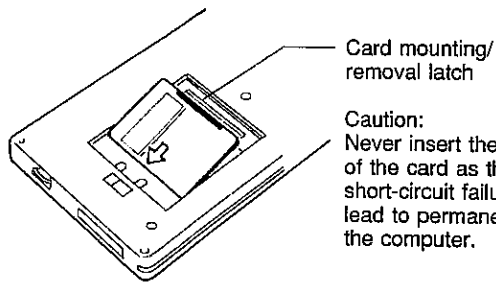
3. Slide the card mounting/removal latch in the direction of the arrow as shown in the figure. This releases the RAM card from the tab so that it can be removed.



4. Take the replacement RAM card out of its case and put the RAM card just removed from the slot into the case.

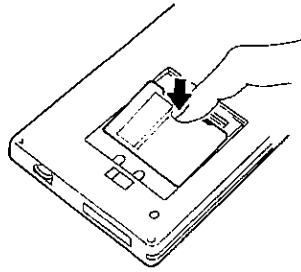


5. Insert the terminal end of the replacement RAM card as shown by the arrow.

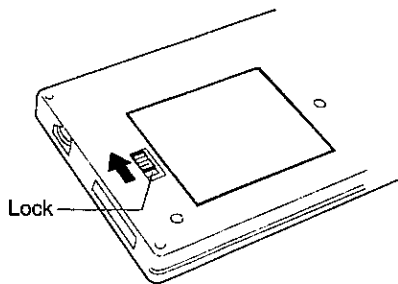


Caution:  
Never insert the opposite end  
of the card as this may cause  
short-circuit failure which can  
lead to permanent damage to  
the computer.

6. Gently press the RAM card and the card will snap into place.



7. Replace and lock the slot cover.



Caution  
The computer will not operate  
unless the cover is locked.  
If you forget this, lock the cover,  
switch the power off and then  
switch it on again.

## RAM Card

The three major purposes of the RAM card are:

1. To expand the computer's memory space.
2. To store programs for use as a program card (set up for RAM disk F).
3. To store programs and data for use as a program and data card (set up with MEM\$="S2").

The program or data stored on the RAM card can be kept intact with the backup battery even after the RAM card is removed from the computer.

### Note:

Carefully read the Operation Manual for the RAM card as well.

**Caution:**

The RAM chips on the RAM card are sensitive to static charges. Static charges build up in your body as you walk on a carpet. Before handling the RAM card, touch a door knob or any other metal structure to discharge yourself. Never touch the RAM card terminals under any circumstance.

The table below gives the capacities of each of the optional RAM cards and its approximate battery life:

RAM card	Capacity	MEM\$="S2"	Battery life	MEM\$="B"
CE-212M	8K bytes	7725 bytes	Approx. 15 months	36792 bytes
CE-2H16M	16K bytes	15917 bytes	Approx. 14 months	44984 bytes
CE-2H32M	32K bytes	32301 bytes	Approx. 14 months	61368 bytes
CE-2H64M	64K bytes	65069 bytes	Approx. 12 months	94136 bytes

**Note 1.** The capacities under "S2" in the table above indicate the capacities of the user (program and data) areas available with MEM\$="S2" configuration.

**Note 2.** Battery life indicates the time period over which a new battery can back up the contents of the RAM card (with the RAM card removed from the computer).


**Note 3.** The capacities under "B" in the table above indicate the capacities of the user (program and data) areas available with the computer's internal memory plus RAM card, with the configuration set to MEM\$="B".

If RAM disks E and/or F are used, the user areas are reduced accordingly.

## How To Use the RAM Card

### Leaving Installed RAM Card Unused


If you wish to use only the computer's internal memory, leaving the installed RAM card unused, use the following configuration:

MEM\$="S1" 

In this case, however, RAM disk F within the RAM card remains available.

### Using Only the RAM Card

If you wish to use only the RAM card, leaving the computer's internal memory unused, use the following configuration:

MEM\$="S2" 

In this case, RAM disk E and Algebraic Expression Reserve (AER) memory area within the computer memory remain available. You can use the RAM card in another PC-E500 only if you are not using RAM disk E or AER memory area within the computer's memory. Programs and variables (including fixed variables) are stored in the RAM card, whereas formulas and program function key designations are stored in computer's internal memory.

## Using the RAM Card as Additional Memory Space

If you wish to use the RAM card as memory space additional to the computer's internal memory, use the following configuration:

MEM\$="B"

In this case, RAM disk F becomes unavailable, while RAM disk E remains available. Note that, before changing the configuration to MEM\$="B," the RAM card must be cleared of all contents.

When you want more RAM disk capacity, it is recommended that you use RAM disk F with MEM\$="S1" configuration, rather than RAM disk E with MEM\$="B" configuration. MEM\$="S1" configuration allows you to use the RAM card in another PC-E500 or more than one RAM card just as you would use floppy disks.

When the RAM card is used as additional memory space, it is not usable in another PC-E500, as it must always be used in conjunction with the computer's internal memory.

If you wish to expand memory space using the RAM card without clearing the program or data currently stored in the computer's internal memory, follow the steps below to initialize only the RAM card ("S2" config.):

1. Turn the computer off, and install the RAM card.
2. While holding down the  key, press the RESET button, then release the RESET button before releasing the  key.
3. Either press the  key to clear data from the RAM card (S2) or press  to initialize the RAM card.
4. Press the  key to tell the computer not to clear the program or data from the computer's internal memory (S1)\*.
5. Enter: MEM\$="B"  in the RUN mode.

\* If you inadvertently press the  key in step 4 above, all program or data will be cleared from the computer's memory. It is strongly advised that you save all programs or data to tape or another medium before performing the above procedure.

## Switching the Memory Configuration

The memory configuration can be switched between MEM\$="S1" and MEM\$="S2" and between MEM\$="S1" and MEM\$="B."

(Memory configuration switching)

"S1" ↔ "S2"  
↓  
"B"

### Example:

To switch memory configuration from MEM\$="S1" into MEM\$="S2":

Enter: MEM\$="S2"

First specify the destination memory configuration, then press the  key. With "S1" configuration, only the computer's internal memory is used, whereas "S2" configuration lets the system use only the RAM card. Thus, you can use different programs and/or data stored in different areas by switching memory configuration between "S1" and "S2."

When switching from MEM\$="S2" to MEM\$="B," first select MEM\$="S1" configuration, press the RESET button to clear all data from the RAM card, and then select MEM\$="B" configuration.

### Supplement:

If you wish to replace the RAM card with one having a larger capacity when the memory is configured with MEM\$="B," you have to temporarily switch configuration from MEM\$="B" to MEM\$="S1" before you replace the RAM card. Note that, in this case, the contents stored with MEM\$="B" configuration should not exceed the capacity of the computer's internal memory. If this is exceeded, follow the steps below:

1. Temporarily save the program and/or data to a cassette tape or pocket disk.
2. Erase the saved program and/or data from memory.
3. Enter: MEM\$="S1"
4. Turn the computer off, then replace the RAM card with the larger capacity, then turn on again.
5. While holding down the  key, press the RESET button, then release the RESET button before releasing the  key.
6. Either press the  key to clear data from the RAM card (S2) or press  to initialize the RAM card.
7. Press the  key to tell the computer not to clear the program or data from the computer's internal memory (S1).
8. Enter: MEM\$="B"  in the RUN mode.
9. Reload the saved program and/or data from the cassette tape or pocket disk into the computer.

## 4. PERIPHERAL DEVICES

The computer can be used with the following optional SHARP peripherals:

<b>MODEL</b>	<b>DESCRIPTION</b>	<b>CONNECTOR</b>
CE-140F	Pocket Disk Drive	11-pin connector
CE-126P	Printer/Cassette Interface	11-pin connector
CE-124	Cassette Interface	11-pin connector
CE-515P	Color Plotter Printer	via CE-516L

A brief description of each unit is given below. For detailed information, refer to the individual operation manuals. See Chapter 3 for information on which RAM cards can be used with the computer.

### CE-140F Pocket Disk Drive

The CE-140F Pocket Disk Drive (2.5 inch micro-floppy disks), which connects directly to the 11-pin connector on the left side of the computer, is similar to the floppy disk drives that are generally available for personal computers. It enables you to write or read a program or data more quickly and easily than when using a cassette recorder. In addition, it enables you to access a data storage location more quickly and to use the COPY statement for simpler data backup. The CE-140F is battery-powered, making it truly portable.

Refer to the CE-140F Operation Manual supplied with the disk drive for instructions for connection to the computer.

### CE-126P Thermal Printer/Cassette Interface

The optional CE-126P Printer/Cassette Interface allows you to add a printer and to connect a cassette recorder to your computer.

The CE-126P features:

- 24-character wide thermal printing.
- Convenient paper feed and tear bar.
- Simultaneous printing of calculations, if desired.
- Easy control of printer output from BASIC programs.
- Built-in cassette interface with remote function.
- Manual and program control of recorder for storing programs and data.
- Dry battery operation for portability.

To connect the computer to the CE-126P, refer to the operation manual supplied with the CE-126P.



## CE-515P Color Plotter/Printer (Color Printer)

The computer can be connected to the optional CE-515P Color Printer.

The CE-515P features:

- Color graphic printing.
- Printing of geometric shapes such as circles and squares.
- Character size selection.
- Drawing line selection (e.g. solid line, dotted line.)

## CE-124 Cassette Interface

This device is required when interfacing the computer to a cassette recorder to save and load programs to tape (unless the built-in interface on the CE-126P printer is used). It plugs into the 11-pin connector on the computer and into the MICrophone and EARphone jack inputs of the recorder. Use a cassette recorder that meets the following specifications:


Item	Requirements
1. Recorder Type	Any standard cassette or micro-cassette recorder
2. Input Jack	"MIC" mini input jack (never the "AUX" jack)
3. Input Impedance	Low input jack impedance (200 – 1000 ohms)
4. Minimum Input Level	Below 3 mV or –50 dB
5. Output Jack	EXTernal, MONITOR, or EARphone mini output jack, or equivalent
6. Output Impedance	Below 10 ohms
7. Output Level	Above 1V (maximum practical output above 100 mW)
8. Distortion	Within 15% (range: 2 kHz to 4 kHz)
9. Wow and Flutter	0.3% maximum (WRMS)
10. Other	Stable speed recorder motor

## Using the Cassette Interface

### Recording (saving) onto magnetic tape


See Tape Notes.

1. Switch off the REMOTE switch on the CE-126P.
2. Enter a program into the computer.
3. Load the tape in the recorder. Advance the tape to the position where you want the program to be recorded, being careful to avoid the clear tape leader (non-magnetic mylar material) and any programs previously recorded.
4. Connect the red plug on the interface to the MIC jack on the tape recorder, and the black plug to the REM jack.
5. Switch on the REMOTE switch.

6. Simultaneously press the RECORD and PLAY buttons on the tape recorder.
7. Enter recording instructions (CSAVE command, SAVE command), and press the  key to start execution as follows.

First, set the computer to RUN or PRO mode. Next, press the following keys:

CSAVE "filename"   
 For example: CSAVE "AA" 

The tape begins to rotate when you press the  key, leaving about an 8-second non-signal blank (beep tone is recorded). The filename and disk contents are then recorded.

To output data, enter statements in the program as shown in the following example:


```
100 OPEN "CAS:DATA" FOR OUTPUT AS #2
110 PRINT #2, AB,C$,D(5)
120 CLOSE #2
```


8. When recording is complete, the PROMPT symbol (>) will be displayed and the tape recorder will automatically stop. Now you have your program on tape (it also remains in the computer).

Use the tape counter on the recorder to locate programs on tape.

### Verifying a Saved Program

After transferring a program to or from tape, you can verify that the program on tape and the program in the computer are identical (and thus be sure that data is correct before continuing programming or running programs) using the CLOAD? command.

1. Switch off the REMOTE switch.
2. Position the tape just before the file that you want to check.
3. Connect the gray plug to the EARphone and the black plug to the REMote jack sockets.
4. Switch on the REMOTE switch.
5. Press the PLAY button.
6. Enter a CLOAD? command and start execution with the  key. Do this as follows: Set the computer to RUN or PRO mode. Enter the following key sequence:

The filename you entered previously.  
 CLOAD? "AA" 

The computer will automatically search for the specified filename and compare the contents on tape with the contents in memory.

When the specified file name is found on the tape, an asterisk(\*) is automatically appended to the line typed in on the screen. This indicates that checking has started.


The PROMPT symbol (>) is displayed if the programs are identical.

If the programs differ, execution will be interrupted and an error message displayed. If this happens, try again.

## Loading from Magnetic Tape

See Tape Notes.

To load a program from magnetic tape into the computer, use the following procedure:


1. Switch off the REMOTE switch.
2. Load the tape in the tape recorder. Position the tape just before the portion to be read out.
3. Connect the gray plug to the EARphone and the black plug to the REMote jack sockets (if the recorder has no REMote socket, use the PAUSE button to control tape movement manually).
4. Switch on the REMOTE switch.
5. Press the PLAY button on the tape recorder (playback mode).  
Set the VOLUME control between middle and maximum.  
Set TONE to maximum treble.
6. Enter transfer instructions (CLOAD command, LOAD command), and press the  key in the following manner:  
Put the computer into RUN or PRO mode. Then press the following keys:

CLOAD "filename" 

For example: CLOAD "AA" 

The specified filename will be automatically searched for and its contents transferred to the computer.

Once the specified file is found on tape, loading begins. This is indicated by an asterisk(\*) that is automatically appended to the line typed in on the screen.



7. When the program has been transferred, the computer will automatically stop the tape and display the PROMPT (>) symbol.  
If an error occurs, try loading again from the beginning. If an error occurs again, repeat the process after adjusting the volume up or down a little. If no error code is displayed but the tape does not stop, something is wrong. Press the  key to stop the tape and try again.


## Tape Notes

1. Always use the same tape recorder for checking or loading that was used for saving the program. Using a different model may generate errors.
2. Use high quality cassette tapes only. Standard audio tapes should not be used.
3. Using an AC adaptor for the CE-126P interface can occasionally cause hum which affects the recording signal. If this happens, switch to battery use.
4. When reusing an old tape for recording programs, erase all old programs on the tape before recording.

## Using the CE-126P Printer/Cassette Interface


### Using the Printer

If you are using the computer for direct input calculations (in the RUN mode only), you may use the CE-126P to simultaneously print your calculations. Press the  key and then the  key (P<->NP) while in the RUN mode.


Press the  key at the end of a calculation. This prints the contents of the display on one line and the results on the next.

For example:

Input

300/50 

Paper






300/50  
6.

You may use the printer from within BASIC programs by using the LPRINT command (see the BASIC COMMAND DICTIONARY for details). Use LPRINT in the same form as the PRINT command.

You may also list your programs on the printer with the LLIST command (see the BASIC COMMAND DICTIONARY for details). If used without line numbers, LLIST will list all program lines currently in memory in their numerical order by line number. A line number range may also be given with LLIST to limit the number of lines that will be printed. When a program line is longer than 24 characters, two or more lines may be used to print one program line. The second and succeeding lines will be indented so that the line number will clearly identify each separate program line.

#### Notes:

- When the printer is exposed to strong external electrical noise, it may print numbers at random. If this happens, press the  key to stop the printing. Switch the CE-126P off, then on, and then press the  key. Pressing the  key will return the printer to its normal condition.
- When not in use, switch off the CE-126P to prolong battery life.

## Using the CE-515P Color Plotter/Printer

Connection of the optional CE-515P Color Printer to the computer allows you to obtain hard-copy outputs of programs and calculation results as well as graphic printouts in multiple colors. The CE-515P draws a chart or diagram with four different color pens. The CE-515P requires a cable (CE-516L) for connection to the serial I/O interface.

#### Color Printer Notes

The BASIC commands used in the computer include various printer commands applicable to a color dot printer or color plotter/printer, as well as the LLIST and LPRINT commands for printing programs and calculation results.

All the values of graphics-related commands except the ratio specification with CIRCLE are truncated to whole numbers when these commands are executed.

The CE-515P color plotter/printer can be used only when the serial input/output interface of the computer is in the "OPEN" state. Therefore, be sure to execute the OPEN command before executing a printer command to the plotter/printer.

## **Drawing Range and Coordinates of Graphics**

When printing data in graphic form with the CE-515P, you may use a printer function called scissoring, whereby that portion of a figure or diagram that falls on the printing paper is actually drawn and that portion outside the printing papers is imaginarily drawn. This function is very convenient when you wish to draw only a part of a figure, but it is easier for you to prepare a program for drawing the entire figure, or when you wish to draw a large diagram, by dividing it into parts according to the size of the printing paper.

However, if your program is incorrect, the figure that is intended to be drawn on the printing paper may be drawn in the imaginary area, so be sure to program correctly when you use this function.

The directions and positions required for drawing a line in graphics printing are expressed by X and Y coordinates. With the X coordinate for horizontal direction, “-” indicates the leftward direction and “+”, the rightward direction. With the Y coordinate for vertical direction, “+” indicates the upward direction and “-”, the downward direction.

Refer to the operation manual supplied with the printer for detailed information.

## **Serial I/O Function**

The computer is equipped with a serial I/O interface. This interface function can be used to connect the optional CE-515P color printer to the computer for graphic printing in multiple colors and to allow data transfer between the computer and a personal computer.

### **Note:**

Exercise care, since applying a voltage exceeding the allowable range of the computer to the I/O terminal may damage the internal parts.

### **Basic Information on Use of the Serial I/O Interface**

The circuit of the serial I/O interface is usually closed. If closed, data from the serial I/O terminal cannot be sent and received data cannot be read.

Therefore, you must open the circuit using the OPEN command. (An attempt to execute this command when already open will result in an error.)

The I/O signals must be the same for both the computer and the connected host. If the signals are different, the data cannot be read correctly and this will result in data errors. The OPEN command can be used to set or modify the I/O conditions. After the conditions for both sending and receiving are satisfied and the circuit is opened, the following commands are used to perform data or program I/O:

LPRINT, LLIST, SAVE, LOAD, CHAIN, MERGE, INPUT\$, PRINT#, INPUT#

To input or output a program (execute SAVE, LOAD, CHAIN or MERGE command), the computer automatically executes the OPEN and CLOSE commands.

At the end of the data (or program) transfer, the circuit of the serial I/O interface is closed. Although the CLOSE command is used to close the circuit, the circuit is also closed when the program ends (such as when the END command is executed) or when the RUN command is executed.

When writing a program which uses the serial I/O interface, the circuit must be opened, the I/O operation performed, and then the circuit must be closed as described above.

**Note:**

The computer is not equipped with a timer function which would interrupt communication with the connected equipment by measuring the waiting time for each I/O command to the serial I/O interface.

Therefore, if the connected equipment is not ready to communicate (such as when the power is off) while a command is being executed or if communication at the connected equipment is interrupted, the computer cannot complete command execution. If this is the case, press the **BREAK** key to stop command execution.

# DIRECT INPUT OPERATION

The PC-E500\* allows direct input calculations in the CAL (calculator), MATRIX, STAT (statistics), ENG (Engineer Software), AER and RUN modes (as opposed to calculations made as part of a program in PRO mode). PART 2 of this Operation Manual describes the use of these modes.

Since there is some overlapping of functions between modes, PART 2 starts with an overview of PC-E500 operation and mode selection. PRO mode is then discussed in detail in PART 3.

\*The PC-E500 is hereafter referred to as "the computer".

# 5. OPERATION: A QUESTION OF MODES

Before starting to use your computer, you must decide which mode to use. The computer has six operational modes.

<b>BASIC mode:</b>	The BASIC mode is subdivided into RUN and PRO modes.
<b>PRO mode:</b>	Allows you to write or correct a BASIC program.
<b>RUN mode:</b>	Allows you to execute a BASIC program, BASIC commands or algebraic expressions (AER).
<b>CAL mode:</b>	Allows you to use the computer as you would an ordinary calculator.
<b>MATRIX mode:</b>	Allows you to perform operations using matrices.
<b>STAT mode:</b>	Allows you to do statistical and regression calculations.
<b>ENG mode:</b>	Allows you to use the Engineer Software.
<b>AER mode:</b>	Allows you to write or correct algebraic expressions.

## Selecting Modes

To display the Main Menu, press the **MENU** key. The Main Menu contains the BASIC, CAL, MATRIX, STAT, and ENG mode options. You can choose any of these modes by pressing the corresponding Programmable Function key (**PF1** - **PF5**). If you press the **↕** key on the Main Menu, an additional mode option, AER, appears, which you can select with the **PF1** key.

You can also change your mode selection after you have already chosen one, as follows:

- To enter the CAL mode, press **<sup>CAL</sup>MENU** while holding down **SHIFT**.
- To enter the RUN mode, press the **BASIC** key (RUN indicator appears).
- Pressing the **BASIC** key in the RUN mode switches the computer to the PRO mode (PRO indicator appears).
- To enter the AER mode, press the **<sup>AER</sup>BASIC** key while holding down **SHIFT**.

### Notes:

- To enter the MATRIX or STAT mode, return to the Main Menu and select the appropriate option.
- If you wish to change modes when executing the Engineer Software, press the **BREAK** key before making any of the mode change operations described above.



# 6. CAL MODE

You can use the computer as a 10-digit function calculator. To do this, you must first set the computer in the CAL mode. Either press the **SHIFT** + **CAL** or select the CAL mode with the **PF2** key on the Main Menu (recall with the **MENU** key).

### Notes:

- **SHIFT** + **CAL** means that you press the **CAL** key while holding down the **SHIFT** key. **2nd F** **CAL** also leads you to the same result.
- In the CAL mode, the results of calculations cannot be output to the printer.

### Calculations

Now try some simple calculations. Press the following keys while watching the display:

<u>Enter</u>	<u>Display</u>
123	123.
<b>+</b>	123.
654	654.
<b>=</b>	777.

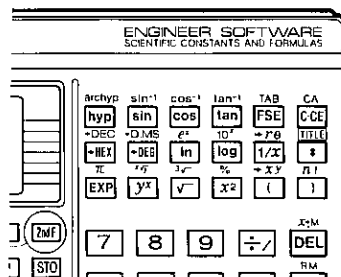
Did you get the correct answer? If you didn't, press the **C•CE** key, and try the same calculation again.

Now call up the value of pi ( $\pi$ ).

The symbol " $\pi$ " is printed in brown above the **EXP** key. The functions identified by brown letters can be used by first pressing the brown **2nd F** key, and then pressing the required function key.

Now press the **2nd F** **EXP** keys.

Yellow key-----



Enter

**2nd F** **7C**

Display

3.141592654

What you see in the display is the value of  $\pi$ .

Next, compute  $10^4$ . For this calculation, you should use the function  $10^x$ . This function is also identified in brown, so the **2nd F** key must be pressed.

Enter

4 **2nd F** **10<sup>x</sup>**

Display

10000.  
( $10^4 = 10000$ .)

An outline of some the major key functions:

\* **C•CE** (clear) (red key)

If this key is pressed immediately after numeric data is entered or the contents of the memory are recalled, that data will be cleared. In any other case, operation of the **C•CE** key will clear the operators and/or numeric data that have been entered. The contents of the memory are not cleared with the **C•CE** key operation.

Enter

123 **+** 456

Display

456.

**C•CE**

0.

789 **=**

912.  
( $123 + 789 = 912$ )

6 **X\*** 2 **+**

12.

**C•CE**

0.

6 **÷/** 2 **+**

3.

5 **=**

8.

The **C•CE** key may also be used to clear an error.

Enter	Display	Error symbol
5 <b>÷/√</b> 0 <b>=</b>		E
	0 .	
<b>C•CE</b>		
	0 .	

\* **FSE** (display mode switch)

This key is used to switch the display mode for the result of a calculation from the floating point decimal system (normal mode) to the fixed point decimal (FIX), scientific notation (SCI), or engineering notation (ENG) system, or vice versa.

Enter	Display	
23 <b>X*</b> 1000 <b>=</b>		
	23000 .	
		(Normal)
<b>FSE</b>		FIX
	23000 .000	
		(FIX)
<b>FSE</b>		SCI
	2 .300E 04	
		(SCI)
<b>FSE</b>		ENG
	23 .000E 03	
		(ENG)

\* **TAB** (specifies the number of decimal places)

This key is used to specify the number of decimal places when used in conjunction with a numeral key. Turn the power switch off and then on again. Press **FSE** key and the display will show "0. 000" (FIX mode).

1. Specify 2 decimal places.

<u>Enter</u>	<u>Display</u>
<b>2ndF</b> <b>TAB</b> <b>2</b>	FIX 0.00
5 <b>÷/</b> 8 <b>=</b>	FIX 0.63

2. Specify 5 decimal places.

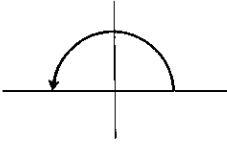
<b>2nd F</b> <b>TAB</b> <b>5</b>	FIX 0.62500
----------------------------------	----------------

\* **DRG** (specifies angular unit.)

This key is used to specify the angular units for numeric data used in trigonometric functions, inverse trigonometric functions, or coordinates conversion.

<u>Enter</u>	<u>Display</u>
	DEG (Degrees)
<b>2nd F</b> <b>DRG</b>	RAD (Radians)
<b>2nd F</b> <b>DRG</b>	GRAD (Grads)
<b>2nd F</b> <b>DRG</b>	DEG (Degrees)

$$180^\circ = \pi \text{ (rad)} = 200^\circ$$



DEG: Degree [°]  
 RAD: Radian [rad]  
 GRAD: Grad [g]

\*  to , , , and

: Used to enter a number in exponential form (the display shows "E" following the number entered).

Enter

Display

4  3

4.E 03  
( $4 \times 10^3$ )

4000.

-4000.

: Used to enter a negative number (or to reverse the sign from negative to positive).

Enter

Display

1.23

-1.23

5

-1.23E-05

( $-1.23 \times 10^{-5}$ )  
-0.0000123

0.0000123

## Basic Operations

### 1. Addition, Subtraction

Enter the following:

12  45.6  32.1  789  741  213

Answer: 286.5

## 2. Multiplication, Division

a. Enter the following: 841  $\boxed{\times}$  586  $\boxed{\div}$  12  $\boxed{=}$

Answer: 41068.83333

b. Enter the following: 427  $\boxed{+}$  54  $\boxed{\times}$  32  $\boxed{\div}$  7  $\boxed{-}$  39  $\boxed{\times}$  2  $\boxed{=}$

Answer: 595.8571429

Note that multiplication and division have priority over addition and subtraction. In other words, multiplication and division will occur before addition and subtraction.

Constant Multiplication: The first number entered is a constant.

Enter: 3  $\boxed{\times}$  5  $\boxed{=}$  Answer: 15

Enter: 10  $\boxed{=}$  Answer: 30

Constant Division: The number entered after the division sign is a constant

Enter: 15  $\boxed{\div}$  3  $\boxed{=}$  Answer: 5

Enter: 30  $\boxed{=}$  Answer: 10

### Note:

the computer places some calculations in pending status depending on their priority levels. Accordingly, in successive calculations the operator and numerical value of the calculation last performed in the computer are handled as a calculation instruction and a constant for the next calculation, respectively.

$a + b \times c =$   $+ bc$  (Constant addition)

$a + b \div c =$   $\div c$  (Constant division)

$a \div b \times c =$   $\frac{a}{b} \times$  (Constant multiplication)

$a \times b - c =$   $- c$  (Constant subtraction)

## 3. Memory Calculations

The independent memory can be accessed using the  $\boxed{\mathcal{X}\rightarrow\mathcal{M}}$ ,  $\boxed{\mathcal{R}\mathcal{M}}$ ,  $\boxed{\mathcal{M}+}$  keys .

Before starting a calculation, clear the memory by pressing  $\boxed{\mathcal{C}\rightarrow\mathcal{C}\mathcal{E}}$  and  $\boxed{\mathcal{X}\rightarrow\mathcal{M}}$  .

If a value other than 0 is stored into the memory, " $\mathcal{M}$ " will be displayed.

→ Enter: 12  $\boxed{+}$  5  $\boxed{\mathcal{M}+}$

Answer: 17

To subtract, key in: 2  $\boxed{+}$  5  $\boxed{=}$   $\boxed{+/-}$   $\boxed{\mathcal{M}+}$

Answer to this equation: -7

Enter  $\boxed{\mathcal{R}\mathcal{M}}$  to recall memory: 10 is displayed.

Enter: 12  $\boxed{\times}$  2  $\boxed{=}$   $\boxed{\mathcal{X}\rightarrow\mathcal{M}}$

Answer: 24 (Also takes place of 10 in memory)

Enter: 8  $\boxed{\div}$  2  $\boxed{\mathcal{M}+}$

Answer: 4  $\boxed{\mathcal{R}\mathcal{M}}$  : 28

↳ When subtracting a number from the memory, press the  $\boxed{+/-}$  and  $\boxed{\mathcal{M}+}$  keys.

In CAL mode, 26 memories, specified with  $\boxed{\mathcal{S}\mathcal{T}\mathcal{O}}$   $\boxed{\mathcal{A}}$  through  $\boxed{\mathcal{S}\mathcal{T}\mathcal{O}}$   $\boxed{\mathcal{Z}}$  , are available, as well as the memory specified with the  $\boxed{\mathcal{X}\rightarrow\mathcal{M}}$  key.

Pressing the  $\boxed{\mathcal{S}\mathcal{T}\mathcal{O}}$   $\boxed{\mathcal{A}}$  keys assigns data to BASIC numeric variable A.

To read this data, press the  $\boxed{\mathcal{R}\mathcal{C}\mathcal{L}}$   $\boxed{\mathcal{A}}$  keys.

Pressing the  $\boxed{\mathcal{A}}$   $\boxed{\leftarrow}$  keys in RUN mode is identical to pressing the  $\boxed{\mathcal{R}\mathcal{C}\mathcal{L}}$

$\boxed{\mathcal{A}}$  keys in CAL mode.

# Scientific Calculations

To perform trigonometric or inverse trigonometric functions, and coordinates conversion, designate the angular unit for the calculation. The angular unit DEG, RAD, or GRAD is specified using the **2nd F** **DRG** keys.

## 1. Trigonometric Functions

Set the angular unit to DEG.

Calculate:  $\sin 30^\circ + \cos 40^\circ =$

Enter the following: 30 **sin** + 40 **cos** **=**

Answer: 1.266044443

Calculate:  $\cos 0.25\pi$

Set the angular unit to RAD.

Enter: 0.25 **X\*** **2nd F**  **$\pi$**  **=** **cos**

Answer: 0.707106781

## 2. Inverse Trigonometric Functions

Calculate:  $\sin^{-1} 0.5$

Set the angular unit to DEG.

Enter: 0.5 **2nd F** **sin<sup>-1</sup>** Answer: 30

Calculate:  $\cos^{-1} -1$

Set the angular unit to RAD.

Enter: 1 **+/-** **2nd F** **cos<sup>-1</sup>** To enter a negative number, press the **+/-** key after the number.

Answer: 3.141592654 (value of  $\pi$ )

The calculation results of the respective inverse trigonometric functions will be displayed within the following limits:

$\theta = \sin^{-1} x$ ,  $\theta = \tan^{-1} x$

DEG:  $-90 \leq \theta \leq 90$  [ $^\circ$ ]

RAD:  $-\pi/2 \leq \theta \leq \pi/2$  [rad]

GRAD:  $-100 \leq \theta \leq 100$  [g]

$\theta = \cos^{-1} x$

DEG:  $0 \leq \theta \leq 180$  [ $^\circ$ ]

RAD:  $0 \leq \theta \leq \pi$  [rad]

GRAD:  $0 \leq \theta \leq 200$  [g]

## 3. Hyperbolic and Inverse Hyperbolic Functions

Calculate:  $\sinh 4$

Enter: 4 **hyp** **sin** Answer: 27.2899172

Calculate:  $\sinh^{-1} 9$

Enter: 9 **2nd F** **archyp** **sin** Answer: 2.893443986

## 4. Power Functions

Calculate:  $20^2$

Enter: 20  **$x^2$**  Answer: 400

Calculate:  $3^3$  and  $3^4$

Enter: 3  **$y^x$**  3 **=** Answer: 27

Enter: 3  **$y^x$**  4 **=** Answer: 81

## 5. Roots

Calculate:  $\sqrt{25}$

Enter: 25  $\sqrt{\quad}$

Answer: 5

Calculate: Cubic root of 27

Enter: 27  $\sqrt[2nd]{F}$   $\sqrt[3]{\quad}$

Answer: 3

Calculate: Fourth root of 81

Enter: 81  $\sqrt[2nd]{F}$   $\sqrt[x]{y}$  4  $=$

Answer: 3

## 6. Logarithmic Functions

Calculate:  $\ln 21$ ,  $\log 173$

Natural Logarithms:

Enter: 21  $\ln$

Answer: 3.044522438

Common Logarithms:

Enter: 173  $\log$

Answer: 2.238046103

## 7. Exponential Functions

Calculate:  $e^{3.0445}$

Enter: 3.0445  $\sqrt[2nd]{F}$   $e^x$

Answer: 20.99952881 (21 as in Natural Logarithms above)

Calculate:  $10^{2.238}$

Enter: 2.238  $\sqrt[2nd]{F}$   $10^x$

Answer: 172.9816359 (173 as in Common Logarithms above)

## 8. Reciprocals

Calculate:  $1/6 + 1/7$

Enter: 6  $1/x$   $+$  7  $1/x$   $=$

Answer: 0.309523809

## 9. Factorial

Calculate: 69!

Enter: 69  $\sqrt[2nd]{F}$   $n!$

Answer: 1.711224524E 98 (=  $1.711224524 \times 10^{98}$ )

Note that the section on Errors deals with the calculation limits of the computer.

Calculate:  ${}_8P_3 = \frac{8!}{(8-3)!}$

Enter: 8  $\sqrt[2nd]{F}$   $n!$   $\div$  ( ) 8  $-$  3 ( )  $\sqrt[2nd]{F}$   $n!$   $=$

Answer: 336

## 10. Percentage Calculations

Calculate: 45% of 2,780 ( $2,780 \times \frac{45}{100}$ )

Enter: 2780  $\times$  45  $\sqrt[2nd]{F}$   $\frac{x}{x^2}$   $=$

Answer: 1251

Calculate:  $200 - 200 \times \frac{30}{100}$

Enter: 200  $-$  30  $\sqrt[2nd]{F}$   $\frac{x}{x^2}$   $=$

Answer: 140



## 11. Angle/Time Conversions

To convert an angle given in the sexagesimal system (degrees/minutes/seconds) to its decimal equivalent, a value in degrees must be entered as an integer and values in minutes and seconds as decimal fractions, respectively.

Convert  $12^{\circ}47'52''$  to its decimal equivalent.

Enter: 12.4752  $\rightarrow$ DEG

Answer: 12.79777778

When converting an angle in decimal degrees to its sexagesimal equivalent (degrees/minutes/seconds), the answer is broken down: integer part = degrees; 1st and 2nd decimal digits = minutes; 3rd and 4th digits = seconds; and the 5th digit and up = fractions of seconds.

Convert 24.7256 to its sexagesimal equivalent (degrees/minutes/seconds)

Enter: 24.7256  $\rightarrow$ 2nd F  $\rightarrow$ DMS

Answer: 24.433216 or  $24^{\circ}43'32''$

A racehorse has the track times of 2 minutes 25 seconds, 2 minutes 38 seconds, and 2 minutes 22 seconds. What is the average running time of the horse?

Enter: 0.0225  $\rightarrow$ DEG  $\rightarrow$  + 0.0238  $\rightarrow$ DEG  $\rightarrow$  + 0.0222  $\rightarrow$ DEG  $\rightarrow$  =

Answer 1: 0.123611111

Enter:  $\rightarrow$   $\div$  3  $\rightarrow$  =

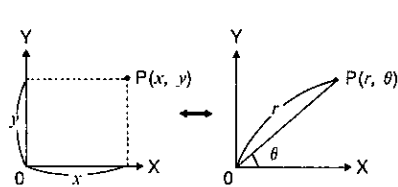
Answer 2: 0.041203703

Enter:  $\rightarrow$ 2nd F  $\rightarrow$ DMS

Answer 3: 0.022833333 or the average time is 2 minutes 28 seconds

## 12. Coordinate Conversion

Converting rectangular coordinates to polar ( $x, y \rightarrow r, \theta$ )



$$r = \sqrt{x^2 + y^2} \quad \text{DEG: } 0 \leq |\theta| \leq 180$$

$$\text{RAD: } 0 \leq |\theta| \leq \pi$$

$$\theta = \tan^{-1} \frac{y}{x} \quad \text{GRAD: } 0 \leq |\theta| \leq 200$$

Solve for  $x = 6$  and  $y = 4$

Angular unit: DEG

Enter: 6  $\rightarrow$   $\rightarrow$  4  $\rightarrow$ 2nd F  $\rightarrow$ r $\theta$

Answer: 7.211102551 (r)

Enter:  $\rightarrow$   $\rightarrow$

Answer: 33.69006753 ( $\theta$ )

Calculate the magnitude and direction (phase) of vector  $i = 12 + j9$

Enter: 12  $\rightarrow$   $\rightarrow$  9  $\rightarrow$ 2nd F  $\rightarrow$ r $\theta$

Answer: 15 (r)

Enter:  $\rightarrow$   $\rightarrow$

Answer: 36.86989765 ( $\theta$ )

Converting polar coordinates to rectangular ( $r, \theta \rightarrow x, y$ )

Solve for  $P(14, \pi/3)$ ,  $r = 14$ ,  $\theta = \pi/3$

Angular unit: RAD

Enter:  $\boxed{2\text{nd F}}$   $\boxed{\pi}$   $\boxed{\div}$   $\boxed{3}$   $\boxed{=}$   $\boxed{\blacklozenge}$   $\boxed{14}$   $\boxed{\blacklozenge}$   $\boxed{2\text{nd F}}$   $\boxed{\rightarrow xy}$   
 Answer: 7.000000002 (x)  
 Enter:  $\boxed{\blacklozenge}$   
 Answer: 12.12435565 (y)

In the above example,  $\theta = \pi/3$  is entered first and is replaced with  $r = 14$  by pressing the  $\boxed{\blacklozenge}$  key after  $r$  is entered.

## Use of Parentheses

The parentheses keys are needed to cluster together a series of operations when it is necessary to override the priority system of algebra. When parentheses are in use in the computer, "( )" will be displayed. Calculations in parentheses have priority over other calculations. Parentheses in the CAL mode can be used up to 15 times in succession. The calculation within the innermost set of parentheses will be performed first.

Calculate:  $12 + 42 \div (8 - 6)$   
 Enter:  $\boxed{12}$   $\boxed{+}$   $\boxed{42}$   $\boxed{\div}$   $\boxed{(}$   $\boxed{8}$   $\boxed{-}$   $\boxed{6}$   $\boxed{)}$   $\boxed{=}$   
 Answer: 33

Calculate:  $126 \div \{(3 + 4) \times (3 - 1)\}$   
 Enter:  $\boxed{126}$   $\boxed{\div}$   $\boxed{(}$   $\boxed{(}$   $\boxed{3}$   $\boxed{+}$   $\boxed{4}$   $\boxed{)}$   $\boxed{\times}$   $\boxed{(}$   $\boxed{3}$   $\boxed{-}$   $\boxed{1}$   $\boxed{)}$   $\boxed{)}$   $\boxed{=}$   
 Answer: 9

### Note:

The  $\boxed{(}$  key operation located just before the  $\boxed{=}$  or  $\boxed{M+}$  key operation can be omitted.

## Decimal Places

The  $\boxed{2\text{nd F}}$  and  $\boxed{\text{TAB}}$  keys are used to specify the number of decimal places in the calculation result. The number of decimal places after the decimal point is specified by a numeral key ( $\boxed{0}$  -  $\boxed{9}$ ) pressed after the  $\boxed{2\text{nd F}}$  and  $\boxed{\text{TAB}}$  keys. In this case, the display mode must be fixed decimal point (FIX), scientific notation (SCI), or engineer notation (ENG).

- $\boxed{2\text{nd F}}$   $\boxed{\text{TAB}}$   $\boxed{0}$  → Designates 0 decimal places.  
(The number is rounded to the nearest integer.)
- $\boxed{2\text{nd F}}$   $\boxed{\text{TAB}}$   $\boxed{1}$  → Designates 1 decimal place.  
(The number is rounded to 1 decimal place.)
- $\boxed{2\text{nd F}}$   $\boxed{\text{TAB}}$   $\boxed{9}$  → Designates 9 decimal place.  
(The number is rounded to 9 decimal places.)

To clear the TAB setting (designation of the decimal places), turn the power switch OFF then ON again. The display will now be in the normal display mode.

**Example:**

2nd F TAB 9  
0.5  $\div$  9 =

FSE

2nd F TAB 3

FSE

FSE

→ 0.055555556 (FIX mode)  
 (The number is rounded to 9 decimal places)

→ 5.555555556E-02 (SCI mode)  
 (The mantissa is rounded to 9 decimal places.)

→ 5.556E-02 (SCI mode)  
 (The mantissa is rounded to 3 decimal places.)

→ 55.556E-03 (ENG mode)

→ 0.055555555

This is held by the computer in the form of  $5.5555555555 \times 10^{-2}$ . Rounding the 11th digit of the mantissa results in  $5.555555556 \times 10^{-2}$ . When the display mode is changed to the floating decimal point mode, the rounded part may not be displayed as in this example.

This function can also be used for statistical or regression calculations and for matrix calculations.

### Priority Levels

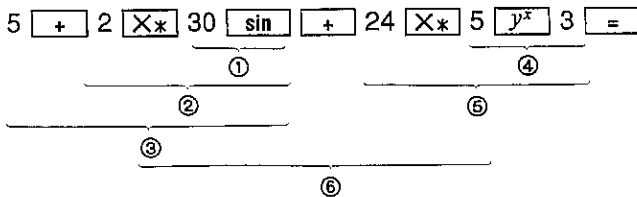
The computer is provided with a function that judges the priority levels of individual calculations, which permits keys to be operated according to a given mathematical formula. The following shows the priority levels of individual calculations.

#### Level Operations

1. Functions such as sin,  $x^2$ , %
2.  $y^x$ ,  $\sqrt[x]{y}$
3.  $\times$ ,  $\div$  (Calculations which are given the same priority level are executed in their sequence of input.)
4. +, -
5. =, M+

#### Example:

Key operation and sequence of calculations in  $5 + 2 \times \sin 30 + 24 \times 5^3 =$



The numbers ① – ⑥ indicate the sequence in which the calculations are carried out. When calculations are executed in sequence from the higher priority level function, a lower priority one must be set aside. The computer is provided with a memory area for up to 3 pending operations for calculations without parentheses. As the memory area can also be used for calculations including parentheses, calculations can be performed according to a given mathematical formula, unless the levels of parentheses and/or pending operations exceeds 8 in total.

- Single-variable functions ( $x^2$ ,  $1/x$ ,  $n!$ ,  $\rightarrow$ DEG,  $\rightarrow$ D.MS, etc.) are calculated immediately after key operation without being stored in memory.

### Calculation without Parentheses

#### Example:

$$1 \boxed{+} 2 \boxed{=} \quad \text{1 pending calculation}$$

①

$$1 \boxed{+} 2 \boxed{\times} 3 \boxed{=} \quad \text{2 pending calculations}$$

①      ②

$$1 \boxed{+} 2 \boxed{\times} 3 \boxed{y^x} 4 \boxed{=} \quad \text{3 pending calculations}$$

①      ②      ③

$$1 \boxed{+} 2 \boxed{\times} 3 \boxed{y^x} 4 \boxed{\div} 5$$

①      ②      ③

~~~~~  
②

1 pending calculation

2 pending calculations

3 pending calculations

After  $\boxed{y^x}$  is pressed, 3 calculations remain pending. Pressing the  $\boxed{\div}$  key executes the calculation of “ $y^x$ ” highest in priority level and “ $\times$ ” identical in priority level. After  $\boxed{\div}$  is pressed, the other 2 calculations remain pending.

### Calculation using Parentheses

#### Example:

i)  $1 \boxed{+} 2 \boxed{\times} 3 \boxed{y^x} \boxed{(}$  4 numerals and calculation instructions are left pending.

①      ②      ③

$$4 \boxed{\div} 5$$

④

ii)  $1 \boxed{+} 2 \boxed{\times} \boxed{(} 3 \boxed{-}$

①      ②      ③

$$4 \boxed{\div} 5 \boxed{)}$$

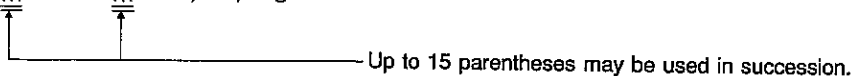
④

~~~~~

Pressing the  $\boxed{)}$  key executes the calculation of  $3 - 4 \div 5$  in the parentheses, leaving 2 calculations pending.

- Parentheses can be used if pending calculations do not exceed eight. However, a maximum of 15 parentheses can be used in succession in a calculation.

Ex.  $a \times (((b - c \times (((d + e) \times f) \div g) \dots\dots\dots$



# Hex Conversion and Calculations

(**→HEX**), (**→DEC**)

**→HEX**: Allows you to convert a decimal number into its hexadecimal equivalent and, at the same time, sets the computer to the HEX mode. (The display shows the symbol "HEX".)

**2nd F →DEC**: Allows you to convert a hexadecimal number into its decimal equivalent and, at the same time, releases the computer from the HEX mode. (Symbol "HEX" disappears from the display.)

Hexadecimal notation is one of the notation systems widely used in the computer field. The radix for hex notation is 16 and hex numbers consist of numerals 0 through 9 and uppercase letters A through F, which are used in place of numbers 10 through 15 of decimal notation.

(Hexadecimal)	(Decimal)
A	10
{	{
F	15

Hex numbers A through F can be entered by first placing your computer in the HEX mode (with **→HEX** key), then pressing the respective keys.

The symbol "HEX" indicates that the numeric data shown in the display is a hex number, and that you can perform any basic arithmetic operations on hex numbers. To clear the HEX mode, operate **2nd F →DEC**. You cannot clear it with the **C•CE** key.

## 1. Decimal to Hex Conversion

### Example:

Convert decimal number 30 into its hexadecimal equivalent:

<u>Enter</u>	<u>Display</u>
30 <b>→HEX</b>	1E. HEX

To perform a new conversion, temporarily clear the HEX mode with **2nd F →DEC**.

### Example:

Convert decimal number -2, into its hexadecimal equivalent:

Temporarily clear the HEX mode with **C•CE 2nd F →DEC**.

<u>Enter</u>	<u>Display</u>
2 <b>+/-</b> <b>→HEX</b>	FFFFFFFE. HEX

- If you attempt decimal-to-hex conversion on a negative decimal number, the computer internally performs “two’s complement” calculation and shows the result in 16’s complement.
- The  $\boxed{+/-}$  key may be used to reverse the sign of the numeric data now in the display. If the sign of a positive hex number is reversed, the complement of the positive number will be displayed.

**Example:**

Convert decimal number 123.4 into its hexadecimal equivalent:

<u>Enter</u>	<u>Display</u>
$\boxed{2nd\ F}\ \boxed{\rightarrow DEC}\ 123.4\ \boxed{\rightarrow HEX}$	7B. HEX

- If a decimal number having a fractional part is converted into a hex number, the fractional part of the decimal number is truncated and only its integer part is converted into a hex number.

**2. Hex to Decimal Conversion**

**Example:**

Convert hex number 2BC into its decimal equivalent:

<u>Enter</u>	<u>Display</u>
$\boxed{C\cdot CE}\ \boxed{\rightarrow HEX}\ 2\ B\ C\ \boxed{2nd\ F}\ \boxed{\rightarrow DEC}$	700.

**Example:**

Convert hex number FFFFFFFF12 into its decimal equivalent:

<u>Enter</u>	<u>Display</u>
$\boxed{C\cdot CE}\ \boxed{\rightarrow HEX}\ FFFFFFFF12$	FFFFFFFF12. HEX
$\boxed{2nd\ F}\ \boxed{\rightarrow DEC}$	-238.

- If any of hex numbers FFFFFFFF to FDABF41C01 is converted into its decimal equivalent, the corresponding decimal number will become negative.

**3. Hexadecimal Calculations**

Hexadecimal calculations can be done after your computer is set in the HEX mode. Press  $\boxed{C\cdot CE}\ \boxed{\rightarrow HEX}$  and the symbol HEX will be displayed.

**Example:**

A4 + BA =

Enter

A4  BA

Display

15E. HEX  
(350 in decimal)

**Example:**

8 × 3 =

Enter

8  3

Display

18. HEX  
(24 in decimal)

**Example:**

(12 + D) × B =

Enter

D   
 B

Display

155. HEX  
(341 in decimal)

**Example:**

43A - 3CB =  
+) A38 - 2FB =  
          
Total

Enter

Display

43A  3CB

6F. HEX

A38  2FB

73D. HEX

7AC. HEX

For hex calculations, you should note the following points:

- In hex calculations, the computer ignores all fractional parts. This means that the decimal point key,  , is meaningless during a hex calculation.
- If an intermediate result in successive hex calculations includes a fractional part, an error will result.

**Example:**

B  $\frac{\square}{\square}$  3  $\times$  ... Error (Symbol "E" is displayed.)

If a fractional part is in the result of the final calculation, it will be truncated and only the integer part of the result will be displayed.

**Example:**

B  $\frac{\square}{\square}$  3  $=$  ... 3. HEX

In the HEX mode, the  $\pm$  key may be used to obtain a complement for the hex number now on the display.

**Example:**

A B  $\pm$  → FFFFFFFF55. HEX  
 $\pm$  → AB. HEX

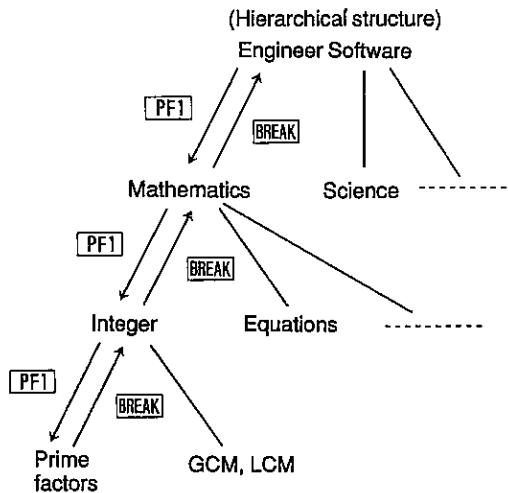
- In the HEX mode, the function keys on the computer are not usable.
- When the computer is in the STAT or MATRIX mode, neither conversion between decimal and hex numbers nor a hex calculation is possible.
- When you wish conversion between decimal and hex numbers in the BASIC mode, use the following assignment statements:  
From decimal to hex: A\$=HEX\$ N  
From hex to decimal: N=VAL("&H"+A\$) or N=&H4F3  
"&H" is the prefix for hex numbers.  
The allowable ranges of hex numbers are:  $0 \leq x \leq 2540BE3FF$  and  $FDABF41C01 \leq x \leq FFFFFFFF$ .



# 7. ENGINEER SOFTWARE MODE

The Engineer Software for the computer allows you to view or review the physical constants or formulas listed in the Engineer Software List (see page 51) and do calculations using those constants or formulas.

Individual programs in the Engineer Software are arranged in an hierarchical tree structure. Use the **programmable function keys** ( **PF1** – **PF5** ) to select a specific program. To use the secondary functions ( **PF6** – **PF10** ) of the programmable function keys, press the **⇩** key before pressing any of the **PF1** to **PF5** keys. (The secondary functions are not accessed with the **SHIFT** key.)




## Operations for Program Selection

If you wish to use, for example, the CUBIC EQUATION program when you are using the PRIME FACTORS program, first press the **BREAK** key once to return to the previous level in the tree structure, then use the appropriate **programmable function key** to select the CUBIC EQUATION program. Thus you can use the **BREAK** key to interrupt the program now being executed and select another program.







The descriptions in this section assume that you are already in the Engineer Software mode by pressing the **PF5** [ENG] key on the main menu. Otherwise, first enter the Engineer Software mode by pressing the **BREAK** key several times or **MENU** **PF5**. If the main menu is not displayed when the **MENU** key is pressed, press the **BREAK** key once before pressing the **MENU** key.

If you make a typing error during numeric data entry, delete the wrong entry with the **C•CE** key, then enter the correct data.




If you press only the  key without entering any data, the computer will respond in either of two ways:

- ① **Accepts the data now on the display.**
- ② **Returns to the first prompt for the current program.**

For more details, see the description for each program.

A value recalled from the physical constant table or periodic table may overflow the display. You can shift the overflowed portion of the value into the display with the  - , , , , or  key. For more details, see the description of the table.

### Caution:

When the computer is in the Engineer Software mode, you may find the  key ineffective in turning the power off. In such an event, press the  key before pressing the  key.

Once you execute any of the Engineer Software programs, all values assigned to BASIC variables will be cleared, with the exception of the array variable MD which shares statistical (STAT) data with the GRAPH (DATA) program.

Some Engineer Software programs automatically change the angular unit (DEG, RAD, GRAD) of input data into radians (RAD) in the course of execution of a calculation.

Any Engineer Software program, when selected, is read into a free space in the memory. If the free space is not sufficient for a particular program size, the computer will display an error message, "Out of memory in Program". If no space is available for variables, the computer will then display another error message, "Answer not found". In such a case, delete unnecessary program lines or variables from memory to make adequate free space. The "Answer not found" message will also appear if an overflow occurs during calculation or if an illegal operation is attempted.

For fractions or square roots ( $\sqrt{\quad}$ ) contained in formulas, the computer uses the following display notations:

(e.g.) •  $\tan 2\theta = \frac{2 \tan \theta}{1 - \tan^2 \theta} \rightarrow \tan 2\theta = 2 \tan \theta / (1 - \tan^2 \theta)$

•  $\sin \frac{\theta}{2} = \pm \sqrt{\frac{1 - \cos \theta}{2}} \rightarrow \sin(\theta/2) = \pm \sqrt{(1 - \cos \theta)/2}$




- Physical constants and solar/planetary constants appearing in this manual are subject to change.
- The Engineer Software may not yield as much data accuracy as the user expects. Carefully check the accuracy you wish for calculation results and that of the data you use for the calculation.

# Engineer Software List

Approximate size refers to the approximate number of bytes for a program. (Variables are excluded.)

		Program Name	Approximate Size (bytes)
<b>MATHEMATICS</b> PF1 [MATH]	INTEGER PF1 [INT]	PRIME FACTORS PF1 [PRIME]	2,300
		G.C.M. & L.C.M. PF2 [GCMLCM]	2,100
	EQUATIONS PF2 [EQU]	CUBIC EQUATION PF1 [CUBIC]	3,800
		NUMERICAL SOLUTION (NEWTON'S METHOD) PF2 [NEWTON]	2,400
		NUMERICAL SOLUTION (METHOD OF BISECTION) PF3 [BISECT]	2,500
	DIFFERENTIAL & INTEGRAL PF3 [DI&INT]	ROMBERG'S METHOD PF1 [INTGRL]	2,800
		RUNGE-KUTTA METHOD PF2 [DIFF]	2,600
		LAGRANGE'S METHOD PF3 [INTPLT]	2,200
	FORMULAS PF4 [FORM]	FACTORIZATION PF1 [FACT]	4,200
		TRIGONOMETRIC FUNCTION PF2 [TRIG]	5,400
		INTEGRATION PF3 [INTFRM]	4,900
		GREEK PF4 [GREEK]	2,700
	GRAPHICS PF5 [GRAPH]	GRAPH (FUNCTION) PF1 [FUNC]	3,400
		GRAPH (DATA) PF2 [DATA]	4,200
		AREA OF FIGURE PF3 [FIGURE]	9,900
SPECIAL FUNCTIONS PF1 [S-FUNC]*	GAMMA FUNCTION PF1 [GAMMA]	2,300	
<b>SCIENCE</b> PF2 [SCI]	PHYSICS PF1 [PHYS]	PHYSICAL CONSTANT PF1 [CONST]	5,200
		METRIC CONVERSION PF2 [M-CONV]	11,400
		EQUATION OF MOTION PF3 [MOTION]	3,200
	CHEMISTRY PF2 [CHEM]	PERIODIC TABLE PF1 [PERIOD]	9,400
		OUTER ELECTRON PF2 [ELECTR]	9,800
		STABLE ISOTOPE PF3 [ISOTOP]	16,700
	EARTH SCIENCE PF3 [EARTH]	METEOROLOGY PF1 [METEO]	5,100
		SOLAR/PLANETARY CONSTANTS PF2 [PLANET]	4,000
	BIOLOGY PF4 [BIO]	AMINO ACIDS FORMULAS PF1 [AMINO]	6,700

		Program Name	Approximate Size (bytes)
ENGINEERING PF3 [ENG]	ELECTRICAL ENGINEERING PF1 [ELEC]	COMPLEX NUMBER PF1 [COMPLX]	3,700
		ELECTRICAL FORMULAS PF2 [EE-FRM]	3,200
		ELECTRIC & MAGNETIC FIELDS PF3 [ELEMAG]	3,200
		LAPLACE TRANSFORMATION PF4 [LAPLAC]	5,000
	MECHANICAL ENGINEERING PF2 [MECH]	MECHANICAL FORMULAS PF1 [ME-FRM]	5,900
STATISTICS PF4 [STAT]	DISTRIBUTION & PROBABILITY PF1 [DISTR]	NORMAL DISTRIBUTION PF1 [NORMAL]	6,000
		† DISTRIBUTION PF2 [t]	6,200
		CHI-SQUARE DISTRIBUTION PF3 [CHI]	6,300
		F DISTRIBUTION PF4 [F]	6,400
EDIT PF5 [EDIT]	PF KEY LABEL EDITOR PF1 [EDITOR]		2,200

\* Press  if necessary. When [INT] or [EQU] is displayed, pressing  will display [S-FUNC]. Pressing  again will return to the previous display.

## \*111 PRIME FACTORS

### Description:

The PRIME FACTORS program factorizes an entered numerical value into its prime factors. The allowable input number is an integer in the range of:  $2 \leq \text{number} < 10^{10}$ .

### Operation:

**PF1** [MATH] **PF1** [INT]  
**PF1** [PRIME]

```
          * PRIME FACTORS *  
Base (2~9999999999) = ?
```

(Entry prompt)

### Example:

Factorize 2200 into its prime factors:

2200

```
          * PRIME FACTORS *  
2200 = 2^3*5^2*11
```

The answer is:  $2200 = 2^3 \times 5^2 \times 11$

To return to the entry prompt, press the  key after you get the answer.

### Note:

- If the input data is outside the allowable range, the computer will prompt for re-entry. If you press only the  key without entering any data, the computer will execute factorization on the data which was last entered.

## \*112 G.C.M. & L.C.M.

### Description:

The G.C.M. & L.C.M. program determines the greatest common measure (G.C.M.) and least common multiple (L.C.M.) of two integers, a and b. The allowable input numbers are integers in the range of:  $1 \leq \text{number} < 10^{10}$ .

The program uses the general Euclidian algorithm for computation.

### Operation:


**PF1** [MATH] **PF1** [INT]  
**PF2** [GCMLCM]

```
          * G, C, M, & L, C, M. *  
a (1~9999999999) = ?
```

(Entry prompt)

**Example:**

Determine the G.C.M. and L.C.M. of two integers,  $a = 6$  and  $b = 8$ :

6  8

```


          * G, C, M, & L, C, M, *
a (1~9999999999) = 6
b (1~9999999999) = 8_
  
```




```

          * G, C, M, & L, C, M, *
a= 6
G, C, M, = 2
L, C, M, = 24
          b= 8
  
```

The answer is: G.C.M. = 2 and L.C.M. = 24

Pressing the  key after you get the answer returns the prompt for number "a".

**Note:**

- If the entered number is outside the allowable range, the computer will prompt for re-entry. If you press only the  key without entering any number, the computer will automatically enter the number "1".

## \*121 CUBIC EQUATION

**Description:**

The CUBIC EQUATION program first prompts for entries of coefficients  $a$ ,  $b$ ,  $c$  and  $d$  for a cubic equation,  $ax^3 + bx^2 + cx + d = 0$ . It then determines the roots,  $\alpha$ ,  $\beta$ , and  $\gamma$ , that will let  $a(x - \alpha)(x - \beta)(x - \gamma) = 0$ , using Cardano's method.

If we assume  $x = y - \frac{a}{3}$  for the general form of the cubic equation,  $x^3 + ax^2 + bx + c = 0$ , we obtain  $y^3 + 3py + q = 0$ . Since  $p = -\frac{a^2}{9} + \frac{b}{3}$  and  $q = \frac{2}{27}a^3 - \frac{ab}{3} + c$ , we may solve the equation,  $x^3 + 3px + q = 0$ , which contains no term of  $x^2$ .

Substituting  $x = u + v$  for the equation above gives:

$$(u + v)^3 + 3p(u + v) + q = 0$$

$$u^3 + v^3 + 3(uv + p)(u + v) + q = 0$$

So we attempt to obtain values for  $u$  and  $v$  that will let  $u^3 + v^3 + q = 0$  and  $uv + p = 0$ .  $u^3$  and  $v^3$  are the two radicals of the quadratic equation,  $t^2 + qt - p^3 = 0$ :

$$u^3 = \frac{1}{2}(-q + \sqrt{q^2 + 4p^3}), \quad v^3 = \frac{1}{2}(-q - \sqrt{q^2 + 4p^3})$$

Therefore if we assume that  $q^2 + 4p^3 = \Delta$ , we obtain:

$$u = \sqrt[3]{\frac{1}{2}(-q + \sqrt{\Delta})}, \quad \omega \sqrt[3]{\frac{1}{2}(-q + \sqrt{\Delta})}, \quad \omega^2 \sqrt[3]{\frac{1}{2}(-q + \sqrt{\Delta})}$$

$$v = \sqrt[3]{\frac{1}{2}(-q - \sqrt{\Delta})}, \quad \omega \sqrt[3]{\frac{1}{2}(-q - \sqrt{\Delta})}, \quad \omega^2 \sqrt[3]{\frac{1}{2}(-q - \sqrt{\Delta})}$$

So if we consider  $uv = -p$ , the solutions are:

$$\sqrt[3]{\frac{1}{2}(-q + \sqrt{\Delta})} + \sqrt[3]{\frac{1}{2}(-q - \sqrt{\Delta})}$$

$$\omega \sqrt[3]{\frac{1}{2}(-q + \sqrt{\Delta})} + \omega^2 \sqrt[3]{\frac{1}{2}(-q - \sqrt{\Delta})}$$

$$\omega^2 \sqrt[3]{\frac{1}{2}(-q + \sqrt{\Delta})} + \omega \sqrt[3]{\frac{1}{2}(-q - \sqrt{\Delta})}$$

$$\text{where } \omega = \frac{-1 + \sqrt{3}i}{2}$$

$$\omega^2 = \frac{-1 - \sqrt{3}i}{2}$$

**Operation:**

PF1 [MATH] PF2 [EQU]

PF1 [CUBIC]

$$* \text{ CUBIC EQUATION } *$$

$$ax^3 + bx^2 + cx + d = 0$$

$$a = \text{?}$$

**Example:**Solve the cubic equation,  $x^3 - 8x^2 + 22x - 20 = 0$ :1  $\leftarrow$  (Coefficient  $a$  is entered.)

$$* \text{ CUBIC EQUATION } *$$

$$ax^3 + bx^2 + cx + d = 0$$

$$b = \text{?}$$

-8  $\leftarrow$  22  $\leftarrow$  -20  
 $\leftarrow$  (Coefficients  $b$ ,  $c$  and  $d$  are entered.)

$$a(x-\alpha)(x-\beta)(x-\gamma) = 0$$

$$\alpha = 2$$

$$\beta = 3 + i$$

$$\gamma = 3 - i$$

The roots of the equation are 2 and  $3 \pm i$ . $\leftarrow$  (Prompt for coefficient  $a$ )

$$* \text{ CUBIC EQUATION } *$$

$$ax^3 + bx^2 + cx + d = 0$$

$$a = \text{?}$$

**Notes:**

- Coefficient  $a$  should not equal zero.
- If only the  $\leftarrow$  key is pressed with no value entered for coefficient  $b$ ,  $c$  or  $d$ , a value of zero will be assumed in each case.
- Roots are displayed with six significant digits each (internal calculations are performed using double precision).

- A triple root appears as shown at right:

$$* \text{ CUBIC EQUATION } *$$

$$a(x-\alpha)^3 = 0$$

$$\alpha = 2$$

- A single real root and a repeated root appear as shown at right:

$$* \text{ CUBIC EQUATION } *$$

$$a(x-\alpha)(x-\beta)^2 = 0$$

$$\alpha = 2$$

$$\beta = 1$$

- The computer performs internal operations using double precision numbers. For an exponential root, it identifies the exponential part of the solution with the letter "D."  
Example:  $\alpha = -3.35D - 25$

## \*122 NUMERICAL SOLUTION (NEWTON'S METHOD)

The NUMERICAL SOLUTION (NEWTON'S METHOD) program determines the real value of  $x$  which causes the value of a function  $f(x)$  to equal zero ( $f(x) = 0$ ), using Newton's method.

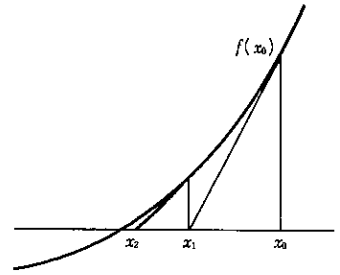
The computer first prompts for entry of the following parameters:

$x_0$ : **Initial value (temporary root, from which the real root  $x$  is obtained.)**

$h$ : **Interval on the  $x$  axis,  $f'(x) = \frac{f(x+h) - f(x)}{h}$**

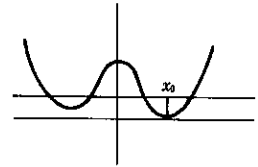
$\epsilon$ : **Convergence condition for solution; calculation is terminated if  $\epsilon \geq |x_{n+1} - x_n|$  is satisfied.**

**loop limit: Maximum repetition count**



### Notes:

- Use the letter "X" for the variable in the function  $f(x)$  you enter.
- If the given initial value is inappropriate, the solution may not converge.



If the tangential line at the initial value  $x_0$  is parallel with the  $x$  axis, the solution will not converge.

- A larger value of  $\epsilon$  will cause a larger error in the solution.
- Use a value of  $\epsilon$  greater than  $10^{-90}$ . Since the computer performs internal operations using 13-digit numbers, it may fail to obtain the solution if the given value of  $\epsilon$  is too small.
- The angular unit will be set to RAD automatically.

### Operation:

PF1 [MATH] PF2 [EQU]  
PF2 [NEWTON]

\* NUMERICAL SOLUTION (NEWTON'S METHOD) \*  
f(x) = ?


### Example:

With initial value  $x_0$  assumed to be 1, determine the root of  $f(x) = x^3 - 8x^2 + 22x - 20$ :


X 2nd F ^ 3 - 8 X\*  
X 2nd F ^ 2 + 22 X\*  
X - 20 ←

\* NUMERICAL SOLUTION (NEWTON'S METHOD) \*  
f(x) = X^3-8\*X^2+22\*X-20  
x0 = ?




1  (Set  $x_0 = 1.$ )


```
* NUMERICAL SOLUTION (NEWTON'S METHOD) *  
h= 0.00001 ?
```

 (Set  $h = 0.00001.$ )


```
* NUMERICAL SOLUTION (NEWTON'S METHOD) *  
h= 0.00001  
ε = 0.000001 ?
```

 (Set  $\epsilon = 0.000001.$ )


```
* NUMERICAL SOLUTION (NEWTON'S METHOD) *  
h= 0.00001  
ε = 0.000001  
loop limit= 30 ?
```

10  (Calculation loops up to 10 times.)





```
* NUMERICAL SOLUTION (NEWTON'S METHOD) *  
f(x)= X^3-8*X^2+22*X-20  
X= 2
```

 (Return to function entry prompt.)

```
* NUMERICAL SOLUTION (NEWTON'S METHOD) *  
f(x)= X^3-8*X^2+22*X-20 ?
```

If you wish to solve the same equation using different conditions, just press the  key in response to the prompt above. The computer will then prompt you to enter an initial value. In this way, you can determine other roots of the function, if any.

#### Notes:

- If you wish to use the expression or value you last entered, just press the  key for each prompt.
- If  $\epsilon \geq |x_{n+1} - x_n|$  is not satisfied or no solution was found after the calculation has been repeated the number of times specified by the loop count, the computer will terminate with "Answer not found" on the display. Pressing the  key at this point will display the executed loop count and the value of  $X_n$ . Pressing the  key a second time will return the function entry prompt.
- If the value given for  $h$  is too small, an overflow may occur in the course of internal computation, in which case the computer terminates with "Answer not found" shown on the display. If you press the  key at this point, the function prompt, is displayed, with no loop count or value of  $X_n$  displayed.

# \*123 NUMERICAL SOLUTION (METHOD OF BISECTION)

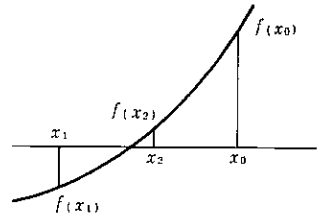
The NUMERICAL SOLUTION (METHOD OF BISECTION) program determines the real value of  $x$  that causes the value of a function  $f(x)$  to equal zero ( $f(x) = 0$ ), using the method of bisection.

The computer first prompts for entry of the following parameters:

$x_0, x_1$ : **Initial values (the root that exists between  $x_0$  and  $x_1$  is to be determined.)**

$\epsilon$ : **Convergence condition for solution; calculation is terminated if  $\epsilon \geq |x_{n+1} - x_n|$  is satisfied.**

**loop limit: Maximum repetition count**



### Notes:

- Use the uppercase letter "X" for the variable in the function  $f(x)$  you enter.
- If the interval between  $x_0$  and  $x_1$  is equally divided by point  $x_2$ , either  $f(x_0) \cdot f(x_2)$  or  $f(x_1) \cdot f(x_2)$  must be less than or equal to zero. Use initial values for  $x_0$  and  $x_1$  that will meet these conditions.
- A larger value of  $\epsilon$  will cause a larger error in the solution.
- Use a value of  $\epsilon$  greater than  $10^{-30}$ . Since the computer performs internal operations using 13-digit numbers, it may fail to obtain the solution if the given value of  $\epsilon$  is too small.
- The angular unit will be set to RAD automatically.

### Operation:

PF1 [MATH] PF2 [EQU]  
PF3 [BISECT]

\*NUMERICAL SOLUTION(METHOD OF BISECTION)  
f(x) = ?

### Example:


With the initial values of  $x_0$  and  $x_1$  assumed to be 5 and 0 respectively, determine the root of the function,  $f(x) = x^3 - 8x^2 + 22x - 20$ :

X [2nd F] [^] 3 [-] 8 [X\*]  
X [2nd F] [^] 2 [+] 22 [X\*]  
X [-] 20 [←]


\*NUMERICAL SOLUTION(METHOD OF BISECTION)  
f(x) = X^3-8\*X^2+22\*X-20  
x0 = ?

5 [←] 0 [←]  
(Set  $x_0 = 5$  and  $x_1 = 0$ .)

\*NUMERICAL SOLUTION(METHOD OF BISECTION)  
 $\epsilon = 0.000001$  ?


 (Set  $\epsilon = 0.000001$ .)

```
*NUMERICAL SOLUTION(METHOD OF BISECTION)
 $\epsilon = 0.000001$ 
loop limit= 40 ?
```


 (Calculation loops up to 40 times.)

```
*NUMERICAL SOLUTION(METHOD OF BISECTION)
f(x)=  $X^3-8*X^2+22*X-20$ 




X= 2
```

 (Return to function prompt.)

```
*NUMERICAL SOLUTION(METHOD OF BISECTION)
f(x)=  $X^3-8*X^2+22*X-20$  ?
```

If you wish to solve the same equation using different conditions, just press the  key in response to the function prompt above. The computer will then prompt you to enter new initial values. In this way, you can determine other roots of the equation, if any.

#### Notes:

- If you wish to use the expression or value you last entered, just press the  key for each prompt.
- If  $\epsilon \geq |x_{n+1} - x_n|$  is not satisfied after the calculation has been repeated the number of times specified by the loop count, the computer will terminate with "Answer not found" on the display. Pressing the  key at this point will display the executed loop count and the value of  $X_n$ . Pressing the  key a second time will return to the function prompt.

## \*131 ROMBERG'S METHOD

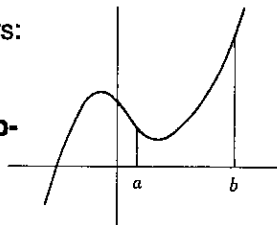
The ROMBERG'S METHOD program integrates a function  $f(x)$  over the interval  $[a, b]$ , using Romberg's method.

The computer first prompts for entry of the following parameters:

$a, b$ : **Integration interval** ( $\int_a^b \dots dx$ )

$\epsilon$ : **Convergence condition for solution; calculation is terminated if  $\epsilon \geq |A_{n+1} - A_n|$  is satisfied. ( $A_n$  is the  $n$ th approximation of the value of the integral.)**

**division limit: Maximum repetition count**



### Notes:

- Use the letter "X" for the variable in the function  $f(x)$  you enter.
- The program may yield incorrect results for some types of functions (see the next page).
- A larger value of  $\epsilon$  will cause a larger error in the result.
- Use a value of  $\epsilon$  greater than  $10^{-90}$ . Since the computer performs internal operations using 13-digit numbers, it may fail to obtain the solution if the given value of  $\epsilon$  is too small.
- The angular unit will be set to RAD automatically.

### Operation:

PF1 [MATH] PF3 [DI&INT]  
PF1 [INTGRL]

```

* ROMBERG'S METHOD *
f(x) = ?
    
```

### Example:

Integrate the function,  $f(x) = x^3 - 4x^2 - 2x + 20$ , over interval  $[0,8]$ :

X 2nd F  $\wedge$  3 - 4 X\*  
X 2nd F  $\wedge$  2 - 2 X\*  
X + 20  $\leftarrow$

```

* ROMBERG'S METHOD *
f(x) = X^3-4*X^2-2*X+20
a = ?
    
```

0  $\leftarrow$  8 (Set  $a = 0$  and  $b = 8$ .)


```

* ROMBERG'S METHOD *
f(x) = X^3-4*X^2-2*X+20
a = 0
b = 8_
    
```


$\leftarrow$

```


* ROMBERG'S METHOD *
ε = 0.000001 ?
    
```

 (Set  $\epsilon = 0.000001$ .)


```
* ROMBERG'S METHOD *  
 $\epsilon = 0.000001$   
division limit= 10 ?
```

 (Calculations loops up to 10 times.)




```
* ROMBERG'S METHOD *  
 $\int X^3 - 4X^2 - 2X + 20 \, dx$   
 $\int dx = 437.333333$ 
```

 (Return to function prompt.)

```
* ROMBERG'S METHOD *  
 $f(x) = X^3 - 4X^2 - 2X + 20$  ?
```

If you wish to integrate the same function over a different interval, just press the  key in response to the function prompt above. The computer will then prompt you to enter a new integration interval.

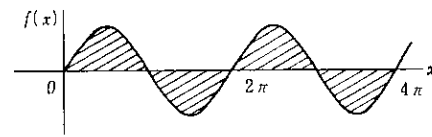
### Notes:

- If you wish to use the expression or value you last entered, just press the  key for each prompt.
- If  $\epsilon \geq |A_{n+1} - A_n|$  is not satisfied after the calculation has been repeated the number of times specified by the division count, the computer will terminate with "Answer not found" shown on the display. Pressing the  key at this point will display the executed loop count and the value of  $A_n$ . Pressing the  key a second time will return to the function prompt.

### Hints on Interval Setting

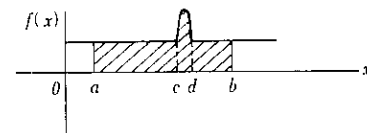
The program may yield an incorrect integration result depending on the type of function being integrated or on the integration interval used. For the functions shown below, use the conventions described:

#### 1) Periodic or symmetric functions



These types of functions should be integrated over each period or symmetric section, and the results for all the periods or sections involved should be added together later. For example, if you wish to integrate the function at left over the interval  $[0, 4\pi]$ , divide the interval into four sub-intervals of  $[0, \pi]$ ,  $[\pi, 2\pi]$ ,  $[2\pi, 3\pi]$ , and  $[3\pi, 4\pi]$ , integrate it over each of these sub-intervals, and finally total the results of each integration.

#### 2) Functions with peaks



If the integration interval in question includes a peak as shown, the computer may yield an incorrect integration result. For such a function, subdivide the interval at the leading and trailing edges of the peak, and sum up the integration results of all the intervals later.

## \*132 RUNGE-KUTTA METHOD

The RUNGE-KUTTA METHOD program solves a first ordinary differential equation,  $\frac{dy}{dx} = P(x)y + Q(x)$ , using the Runge-Kutta method.

The program sequentially determines the value of  $y$  for a given value of  $x$  which is the sum of  $x_0$  and increment  $h$ , with  $(x_0, y_0)$  being initial values specified by the user. Up to 50 values of  $y$  can be obtained.

The computer first prompts for entry of the following parameter values:

$x_0$ : **Initial value of  $x$**

$y_0$ : **Initial value of  $y$**

$h$ : **Increment**

### Notes:

Use letters "X" and "Y" for the variables of the equation you enter.

The angular unit will be set to RAD automatically.

### Operation:

PF1 [MATH] PF3 [DI&INT]  
PF2 [DIFF]

```

* RUNGE-KUTTA METHOD *
dy/dx= ?
    
```

### Example:

Solve the ordinary differential equation,  $\frac{dy}{dx} = -\frac{x}{y}$ , under the assumption that  $x_0 = 0$  and  $y_0 = 10$  (the solution is given in the form of  $x^2 + y^2 = C$ ):

$\left[ \leftarrow \right]$  X  $\left[ \leftarrow \right]$  Y  $\left[ \leftarrow \right]$

```

* RUNGE-KUTTA METHOD *
dy/dx= -X/Y
x0= ?
    
```

0  $\left[ \leftarrow \right]$  (Set  $x_0 = 0$ .)

```

* RUNGE-KUTTA METHOD *
dy/dx= -X/Y
x0= 0
y0= ?
    
```

10  $\left[ \leftarrow \right]$  (Set  $y_0 = 10$ .)

```

* RUNGE-KUTTA METHOD *
dy/dx= -X/Y
h= 1 ?
    
```

0.1  $\left[ \leftarrow \right]$  (Set  $h = 0.1$ .)

```

* RUNGE-KUTTA METHOD *
dy/dx= -X/Y
y(0) = 10
    
```

$y = 10$  for  $x = 0$




```
* RUNGE-KUTTA METHOD *  
dy/dx= -X/Y  
y(0.1) = 9.999499988
```

y = 9.9994 ... for x = 0.1





```
* RUNGE-KUTTA METHOD *  
dy/dx= -X/Y  
y(0.2) = 9.9979998
```

y = 9.9979 ... for x = 0.2

The computer will display values of  $y$  for each 0.1 increment of  $x$  each time you press the  key up to 50.

**Notes:**

- To recall the preceding value of  $y$ , press the  key.
- To return to the equation prompt, press the  key.

## \*133 LAGRANGE'S METHOD

The LAGRANGE'S METHOD program generates the  $n$ th order polynomial that passes ( $n + 1$ ) different points and determines the value of  $y$  for any value of  $x$ , using the Lagrange interpolation method. From 2 to 200 data points may be entered.

### Operation:

PF1 [MATH] PF3 [DI&INT]  
PF3 [INTPLT]

\* LAGRANGE'S METHOD \*

Number of data= 2 ?

(Initial prompt)

### Example:

Generate a polynomial that passes through the following four points and obtain the value of  $y$  for  $x = 2.3$ :

$P_1(-2, 13), P_2(1, 4), P_3(3, 18), P_4(4, 49)$

4  ←  
(The number of data is 4.)

\* LAGRANGE'S METHOD \*

Number of data= 4

x(1)= 0 ?

-2  ← 13  ←  
(Data for  $P_1$  is set.)

Number of data= 4

x(1)= 0 -2

y(1)= 0 13

x(2)= 0 ?

1  ← 4  ← 3  ←  
18  ← 4  ← 49  ←

x= ?

Get the value of  $y$  for  $x = 2.3$ :  
2.3  ←

x= 2.3

y= 7.367000004

x= ?

$y = 7.367 \dots$  for  $x = 2.3$

### Note:

- If you press only the  ← key without entering the value of  $x$  for interpolation, the initial prompt will be displayed.



## \*141 FACTORIZATION

The FACTORIZATION program displays any of the following 26 factorization formulas:

- (1)  $a^2 - b^2 = (a + b)(a - b)$
- (2)  $a^3 \pm b^3 = (a \pm b)(a^2 \mp ab + b^2)$
- (3)  $a^4 - b^4 = (a + b)(a - b)(a^2 + b^2)$
- (4)  $a^4 + b^4 = (a^2 + \sqrt{2}ab + b^2)(a^2 - \sqrt{2}ab + b^2)$
- (5)  $a^n - b^n = (a - b)(a^{n-1} + a^{n-2}b + a^{n-3}b^2 + \dots + b^{n-1})$
- (6)  $a^n + b^n = (a + b)(a^{n-1} - a^{n-2}b + a^{n-3}b^2 - \dots + b^{n-1})$  ( $n$ : an odd number)
- (7)  $a^2 \pm 2ab + b^2 = (a \pm b)^2$
- (8)  $a^3 \pm 3a^2b + 3ab^2 \pm b^3 = (a \pm b)^3$
- (9)  $(a \pm b)^2 \mp 4ab = (a \mp b)^2$
- (10)  $a^2 + b^2 + c^2 + 2bc + 2ca + 2ab = (a + b + c)^2$
- (11)  $a^4 + a^2b^2 + b^4 = (a^2 + ab + b^2)(a^2 - ab + b^2)$
- (12)  $a^3 + b^3 + c^3 - 3abc = (a + b + c)(a^2 + b^2 + c^2 - bc - ca - ab)$
- (13)  $(ac - bd)^2 + (ad + bc)^2 = (a^2 + b^2)(c^2 + d^2)$
- (14)  $(ac + bd)^2 + (ad - bc)^2 = (a^2 + b^2)(c^2 + d^2)$
- (15)  $(ac + bd)^2 - (ad + bc)^2 = (a^2 - b^2)(c^2 - d^2)$
- (16)  $(ac - bd)^2 - (ad - bc)^2 = (a^2 - b^2)(c^2 - d^2)$
- (17)  $a^2(b - c) + b^2(c - a) + c^2(a - b) = -(b - c)(c - a)(a - b)$
- (18)  $(b - c)^3 + (c - a)^3 + (a - b)^3 = 3(b - c)(c - a)(a - b)$
- (19)  $a^4 + b^4 + c^4 - 2b^2c^2 - 2c^2a^2 - 2a^2b^2 = (a + b + c)(b - c - a)(c - a - b)(a - b - c)$
- (20)  $x^2 + (a + b)x + ab = (x + a)(x + b)$
- (21)  $acx^2 + (ad + bc)x + bd = (ax + b)(cx + d)$
- (22)  $x^3 + (a + b + c)x^2 + (bc + ca + ab)x + abc = (x + a)(x + b)(x + c)$
- (23)  $a^2 - b^2 - c^2 - 2bc = (a + b + c)(a - b - c)$
- (24)  $(a + b + c)(bc + ca + ab) - abc = (b + c)(c + a)(a + b)$
- (25)  $(a + b + c)^3 - (a^3 + b^3 + c^3) = 3(b + c)(c + a)(a + b)$
- (26)  $a^3(b - c) + b^3(c - a) + c^3(a - b) = -(b - c)(c - a)(a - b)(a + b + c)$

**Operation:**

**PF1** [MATH] **PF4** [FORM]  
**PF1** [FACT]

* FACTORIZATION *				
1 : $a^2 - b^2 = (a+b)(a-b)$				
[ <b>↑</b> ]	[ <b>▲</b> ]	[CENTER]	[ <b>▼</b> ]	[ <b>↓</b> ]

**PF4** [ **▼** ]

* FACTORIZATION *				
6 : $a^n + b^n$ <span style="float: right;">(n=odd)</span>				
$= (a+b)(a^{n-1} - a^{n-2}b + a^{n-3}b^2 - \dots + b^{n-1})$				
[ <b>↑</b> ]	[ <b>▲</b> ]	[CENTER]	[ <b>▼</b> ]	[ <b>↓</b> ]

Use the following keys to recall a specific formula:

- PF1** [ **↑** ] : Recalls the first formula.
- PF2** [ **▲** ] : Recalls the formula which is five steps before the formula now displayed (formula with the current number minus 5).
- PF3** [CENTER] : Recalls the 13th (mid) formula.
- PF4** [ **▼** ] : Recalls the formula which is five steps beyond the formula now displayed (formula with the current number plus 5).
- PF5** [ **↓** ] : Recalls the 26th (last) formula.
- ↓** : Recalls the formula which immediately follows the formula now displayed.
- ↑** : Recalls the formula which immediately precedes the formula now displayed.

**Note:**

- The **PF2**, **PF4**, **↓**, and **↑** keys continuously recall formulas while they are pressed down.

## \*142 TRIGONOMETRIC FUNCTION

The TRIGONOMETRIC FUNCTION program displays any of the following 49 trigonometric function formulas:

$$(1) \sin^2\theta + \cos^2\theta = 1$$

$$(2) 1 + \tan^2\theta = \sec^2\theta$$

$$(3) 1 + \cot^2\theta = \operatorname{cosec}^2\theta$$

$$(4) \sin(\alpha \pm \beta) = \sin \alpha \cdot \cos \beta \pm \cos \alpha \cdot \sin \beta$$

$$(5) \cos(\alpha \pm \beta) = \cos \alpha \cdot \cos \beta \mp \sin \alpha \cdot \sin \beta$$

$$(6) \tan(\alpha \pm \beta) = \frac{\tan \alpha \pm \tan \beta}{1 \mp \tan \alpha \cdot \tan \beta}$$

$$(7) \cot(\alpha \pm \beta) = \frac{\cot \alpha \cdot \cot \beta \mp 1}{\cot \beta \pm \cot \alpha}$$

$$(8) \sin(\alpha + \beta) \cdot \sin(\alpha - \beta) = \sin^2\alpha - \sin^2\beta$$

$$(9) \sin(\alpha + \beta) \cdot \sin(\alpha - \beta) = \cos^2\beta - \cos^2\alpha$$

$$(10) \cos(\alpha + \beta) \cdot \cos(\alpha - \beta) = \cos^2\alpha - \sin^2\beta$$

$$(11) \cos(\alpha + \beta) \cdot \cos(\alpha - \beta) = \cos^2\beta - \sin^2\alpha$$

$$(12) \sin(\alpha \pm \beta) \cdot \cos(\alpha \mp \beta) = \sin \alpha \cdot \cos \alpha \pm \sin \beta \cdot \cos \beta$$

$$(13) \frac{\sin(\alpha + \beta)}{\sin(\alpha - \beta)} = \frac{\tan \alpha + \tan \beta}{\tan \alpha - \tan \beta}$$

$$(14) \frac{\cos(\alpha + \beta)}{\cos(\alpha - \beta)} = \frac{1 - \tan \alpha \cdot \tan \beta}{1 + \tan \alpha \cdot \tan \beta}$$

$$(15) \sin 2\theta = 2 \sin \theta \cdot \cos \theta$$

$$(16) \cos 2\theta = \cos^2\theta - \sin^2\theta$$

$$(17) \cos 2\theta = 1 - 2 \sin^2\theta$$

$$(18) \cos 2\theta = 2 \cos^2\theta - 1$$

$$(19) \tan 2\theta = \frac{2 \tan \theta}{1 - \tan^2\theta}$$

$$(20) \sin \frac{\theta}{2} = \pm \sqrt{\frac{1 - \cos \theta}{2}}$$

$$(21) \cos \frac{\theta}{2} = \pm \sqrt{\frac{1 + \cos \theta}{2}}$$

$$(22) \tan \frac{\theta}{2} = \pm \sqrt{\frac{1 - \cos \theta}{1 + \cos \theta}}$$

$$(23) \tan \frac{\theta}{2} = \frac{1 - \cos \theta}{\sin \theta}$$

$$(24) \tan \frac{\theta}{2} = \frac{\sin \theta}{1 + \cos \theta}$$

$$(25) \tan \frac{\theta}{2} = \operatorname{cosec} \theta - \cot \theta$$

$$(26) \cot \frac{\theta}{2} = \pm \sqrt{\frac{1 + \cos \theta}{1 - \cos \theta}}$$

$$(27) \cot \frac{\theta}{2} = \frac{\sin \theta}{1 - \cos \theta}$$

$$(28) \cot \frac{\theta}{2} = \frac{1 + \cos \theta}{\sin \theta}$$

$$(29) \cot \frac{\theta}{2} = \operatorname{cosec} \theta + \cot \theta$$

$$(30) \sin 3\theta = -4 \sin^3\theta + 3 \sin \theta$$

$$(31) \cos 3\theta = 4 \cos^3\theta - 3 \cos \theta$$

$$(32) \tan 3\theta = \frac{3 \tan \theta - \tan^3\theta}{1 - 3 \tan^2\theta}$$

$$(33) \sin \alpha \cdot \cos \beta = \frac{\sin(\alpha + \beta) + \sin(\alpha - \beta)}{2}$$

$$(34) \cos \alpha \cdot \sin \beta = \frac{\sin(\alpha + \beta) - \sin(\alpha - \beta)}{2}$$

$$(35) \cos \alpha \cdot \cos \beta = \frac{\cos(\alpha + \beta) + \cos(\alpha - \beta)}{2}$$

$$(36) \sin \alpha \cdot \sin \beta = \frac{\cos(\alpha + \beta) - \cos(\alpha - \beta)}{2}$$

$$(37) \sin \alpha + \sin \beta = 2 \sin\left(\frac{\alpha + \beta}{2}\right) \cdot \cos\left(\frac{\alpha - \beta}{2}\right)$$

$$(38) \sin \alpha - \sin \beta = 2 \cos\left(\frac{\alpha + \beta}{2}\right) \cdot \sin\left(\frac{\alpha - \beta}{2}\right)$$

$$(39) \cos \alpha + \cos \beta = 2 \cos\left(\frac{\alpha + \beta}{2}\right) \cdot \cos\left(\frac{\alpha - \beta}{2}\right)$$

$$(40) \cos \alpha - \cos \beta = -2 \sin\left(\frac{\alpha + \beta}{2}\right) \cdot \sin\left(\frac{\alpha - \beta}{2}\right)$$

$$(41) \tan \alpha \pm \tan \beta = \frac{\sin(\alpha \pm \beta)}{\cos \alpha \cdot \cos \beta}$$

$$(42) \cot \alpha \pm \cot \beta = \frac{\sin(\beta \pm \alpha)}{\sin \alpha \cdot \sin \beta}$$

$$(43) \cos \alpha \pm \sin \alpha = \sqrt{2} \sin(45^\circ \pm \alpha)$$

$$(44) \cos \alpha \pm \sin \alpha = \sqrt{2} \cos(45^\circ \mp \alpha)$$

$$(45) \tan\left(45^\circ \pm \frac{\theta}{2}\right) = \sec \theta \pm \tan \theta$$

$$(46) \tan\left(45^\circ \pm \frac{\theta}{2}\right) = \frac{1 \pm \sin \theta}{\cos \theta}$$

$$(47) \tan\left(45^\circ \pm \frac{\theta}{2}\right) = \cot\left(45^\circ \mp \frac{\theta}{2}\right)$$

$$(48) \tan(45^\circ + \theta) = \frac{1 + \tan \theta}{1 - \tan \theta}$$

$$(49) \cot(45^\circ - \theta) = \frac{1 + \cot \theta}{1 - \cot \theta}$$

**Operation:**

**PF1** [MATH] **PF4** [FORM]

**PF2** [TRIG]

```

* TRIGONOMETRIC FUNCTION *
1: sin² θ + cos² θ = 1
[  ⏮  ] [  ⏴  ] [CENTER] [  ⏶  ] [  ⏵  ]

```

**PF4** [↘]

```

* TRIGONOMETRIC FUNCTION *
6: tan(α±β) = (tan α ± tan β) / (1 ∓ tan α · tan β)
[  ⏮  ] [  ⏴  ] [CENTER] [  ⏶  ] [  ⏵  ]

```

Use the following keys to recall a specific formula:

**PF1** [⏮]

: Recalls the first formula.

**PF2** [⏴]

: Recalls the formula which is five steps before the formula now displayed (i.e. the formula with the current number minus 5).

**PF3** [CENTER]

: Recalls the 25th (mid) formula.

**PF4** [↘]

: Recalls the formula which is five steps beyond the formula now displayed (i.e. the formula with the current number plus 5).

**PF5** [⏵]

: Recalls the 49th (last) formula.

[⏶]

: Recalls the formula which immediately follows the formula now displayed.

[⏴]

: Recalls the formula which immediately precedes the formula now displayed.

**Note:**

- The **PF2**, **PF4**, [⏶], and [⏴] keys continuously recall formulas while they are pressed down.

## \*143 INTEGRATION

The INTEGRATION program displays any of the following 42 integration formulas:

- |  |  |
|--|--|
| <p>(1) <math>\int dx = x + C</math></p> <p>(2) <math>\int x^n dx = \frac{x^{n+1}}{n+1} + C \quad (n+1 \neq 0)</math></p> <p>(3) <math>\int \frac{1}{x} dx = \log x  + C</math></p> <p>(4) <math>\int \frac{1}{x \pm a} dx = \log x \pm a  + C</math></p> <p>(5) <math>\int e^x dx = e^x + C</math></p> <p>(6) <math>\int e^{ax} dx = \frac{1}{a} e^{ax} + C</math></p> <p>(7) <math>\int a^x dx = \frac{a^x}{\log a} + C \quad (a &gt; 0, a \neq 1)</math></p> <p>(8) <math>\int a^{nx} dx = \frac{a^{nx}}{n \cdot \log a} + C \quad (a &gt; 0, a \neq 1)</math></p> <p>(9) <math>\int \log x dx = x(\log x - 1) + C</math></p> <p>(10) <math>\int x^n \cdot \log x dx = \frac{x^{n+1}}{n+1} \left( \log x - \frac{1}{n+1} \right) + C \quad (n \neq -1)</math></p> <p>(11) <math>\int x e^{ax} dx = \frac{e^{ax}}{a^2} \cdot (nx - 1) + C</math></p> <p>(12) <math>\int \sin x dx = -\cos x + C</math></p> <p>(13) <math>\int \sin ax dx = -\frac{1}{a} \cdot \cos ax + C</math></p> <p>(14) <math>\int \cos x dx = \sin x + C</math></p> <p>(15) <math>\int \cos ax dx = \frac{1}{a} \cdot \sin ax + C</math></p> <p>(16) <math>\int \tan x dx = -\log \cos x  + C</math></p> <p>(17) <math>\int \tan ax dx = -\frac{1}{a} \log \cos ax  + C</math></p> <p>(18) <math>\int \cot x dx = \log \sin x  + C</math></p> <p>(19) <math>\int \cot ax dx = \frac{1}{a} \log \sin ax  + C</math></p> <p>(20) <math>\int \sec ax dx = \frac{1}{a} \log \tan \left( \frac{\pi}{4} + \frac{ax}{2} \right) + C</math></p> <p>(21) <math>\int \sec ax dx = \frac{1}{2a} \log \left( \frac{1 + \sin ax}{1 - \sin ax} \right) + C</math></p> | <p>(22) <math>\int \operatorname{cosec} ax dx = \frac{1}{a} \log \tan \frac{ax}{2} + C</math></p> <p>(23) <math>\int \operatorname{cosec} ax dx = -\frac{1}{2a} \log \left( \frac{1 + \cos ax}{1 - \cos ax} \right) + C</math></p> <p>(24) <math>\int \sin^2 x dx = \frac{1}{2} x - \frac{1}{4} \sin 2x + C</math></p> <p>(25) <math>\int \cos^2 x dx = \frac{1}{2} x + \frac{1}{4} \sin 2x + C</math></p> <p>(26) <math>\int \sec^2 ax dx = \frac{1}{a} \cdot \tan ax + C</math></p> <p>(27) <math>\int \operatorname{cosec}^2 ax dx = -\frac{1}{a} \cdot \cot ax + C</math></p> <p>(28) <math>\int \frac{1}{\sin x} dx = \log \tan \frac{x}{2} + C</math></p> <p>(29) <math>\int \frac{1}{\cos x} dx = \log \tan \left( \frac{\pi}{4} + \frac{x}{2} \right) + C</math></p> <p>(30) <math>\int e^{nx} \sin bx dx = \frac{1}{n^2 + b^2} e^{nx} (n \cdot \sin bx - b \cdot \cos bx) + C</math></p> <p>(31) <math>\int e^{nx} \cos bx dx = \frac{1}{n^2 + b^2} e^{nx} (n \cdot \cos bx + b \cdot \sin bx) + C</math></p> <p>(32) <math>\int \sin^{-1} x dx = x \sin^{-1} x + \sqrt{1-x^2} + C</math></p> <p>(33) <math>\int \cos^{-1} x dx = x \cos^{-1} x - \sqrt{1-x^2} + C</math></p> <p>(34) <math>\int \sinh x dx = \cosh x + C</math></p> <p>(35) <math>\int \cosh x dx = \sinh x + C</math></p> <p>(36) <math>\int \tanh x dx = \log \cosh x + C</math></p> <p>(37) <math>\int \frac{1}{\sqrt{a^2 - x^2}} dx = \sin^{-1} \frac{x}{a} + C \quad ( x  &lt; a)</math></p> <p>(38) <math>\int \frac{1}{a^2 - x^2} dx = \frac{1}{2a} \log \left  \frac{a+x}{a-x} \right  + C</math></p> <p>(39) <math>\int \frac{1}{a^2 + x^2} dx = \frac{1}{a} \tan^{-1} \frac{x}{a} + C</math></p> <p>(40) <math>\int \frac{1}{\sqrt{x^2 \pm a^2}} dx = \log(x + \sqrt{x^2 \pm a^2}) + C</math></p> <p>(41) <math>\int \sqrt{a^2 - x^2} dx = \frac{1}{2} \cdot \left( x \sqrt{a^2 - x^2} + a^2 \sin^{-1} \frac{x}{a} \right) + C</math></p> <p>(42) <math>\int \frac{1}{x^2 - a^2} dx = \frac{1}{2a} \log \left( \frac{x-a}{x+a} \right) + C \quad (x &gt; a)</math></p> |
|--|--|

- "log" means natural logarithm ( $\log_e$ ).
- All trigonometric functions are in radians.

**Operation:**

**PF1** [MATH] **PF4** [FORM]  
**PF3** [INTFRM]

\* INTEGRATION \*

1:  $\int dx = x + C$

[ **↑** ] [ **↕** ] [CENTER] [ **↓** ] [ **⇓** ]

**PF4** [ **⇓** ]

\* INTEGRATION \*

6:  $\int e^{n \cdot x} dx = \frac{1}{n} \cdot e^{n \cdot x} + C$

[ **↑** ] [ **↕** ] [CENTER] [ **↓** ] [ **⇓** ]

Use the following keys to recall a specific formula:

- PF1** [ **↑** ] : Recalls the first formula.
- PF2** [ **↕** ] : Recalls the formula which is five steps before the formula now displayed (i.e. the formula with the current number minus 5).
- PF3** [CENTER] : Recalls the 21st (mid) formula.
- PF4** [ **⇓** ] : Recalls the formula which is five steps beyond the formula now displayed (i.e. the formula with the current number plus 5).
- PF5** [ **⇓** ] : Recalls the 42nd (last) formula.
- [ **↓** ] : Recalls the formula which immediately follows the formula now displayed.
- [ **↑** ] : Recalls the formula which immediately precedes the formula now displayed.

**Note:**

- The **PF2**, **PF4**, [ **↓** ], and [ **↑** ] keys continuously recall formulas while they are pressed down.

## \*144 GREEK

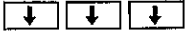
The GREEK program displays the following 24 Greek letters in both capital and small letters, their pronunciations, and corresponding Roman characters:

<i>A</i>	$\alpha$	Alpha	(a)	<i>N</i>	$\nu$	Nu	(n)
<i>B</i>	$\beta$	Beta	(b)	<i>E</i>	$\xi$	Xi	(x)
<i>G</i>	$\gamma$	Gamma	(g)	<i>O</i>	$o$	Omicron	(ö)
<i>D</i>	$\delta$	Delta	(d)	<i>P</i>	$\rho$	Rho	(r)
<i>E</i>	$\epsilon$	Epsilon	(ë)	<i>S</i>	$\sigma$	Sigma	(s)
<i>Z</i>	$\zeta$	Zeta	(z)	<i>T</i>	$\tau$	Tau	(t)
<i>H</i>	$\eta$	Eta	(ë)	<i>Y</i>	$\upsilon$	Upsilon	(u)
<i>Θ</i>	$\theta$	Theta	(th)	<i>Φ</i>	$\phi$	Phi	(ph)
<i>I</i>	$\iota$	Iota	(i)	<i>X</i>	$\chi$	Chi	(ch)
<i>K</i>	$\kappa$	Kappa	(k)	<i>Ψ</i>	$\psi$	Psi	(ps)
<i>A</i>	$\lambda$	Lambda	(l)	<i>Ω</i>	$\omega$	Omega	(ö)
<i>M</i>	$\mu$	Mu	(m)				

**Operation:**

**PF1** [MATH] **PF4** [FORM]  
**PF4** [GREEK]

		* GREEK *	
A	$\alpha$ Alpha	(a)	: B $\beta$ Beta (b)
$\Gamma$	$\gamma$ Gamma	(g)	: $\Delta$ $\delta$ Delta (d)
E	$\epsilon$ Epsilon	(e)	: Z $\zeta$ Zeta (z)



P	$\rho$ Rho	(r)	: $\Sigma$ $\sigma$ Sigma (s)
T	$\tau$ Tau	(t)	: $\Upsilon$ $\upsilon$ Upsilon (u)
$\Phi$	$\phi$ Phi	(ph)	: $\chi$ $\chi$ Chi (ch)
$\Psi$	$\psi$ Psi	(ps)	: $\Omega$ $\omega$ Omega (o)

**Note:**

- The **↓** and **↑** keys continuously recall Greek letters in the forward and backward directions while they are pressed down. Each operation of either key scrolls the screen 3 lines upward or downward at a time.

## \*151 GRAPH (FUNCTION)

The GRAPH (FUNCTION) program plots the graph of an entered function. Use the letter "X" for the variable in the function you enter.

**Operation:**

**PF1** [MATH] **PF5** [GRAPH]  
**PF1** [FUNC]

\* GRAPH (FUNCTION) \*  
f(x) = ?

**Example:**

Plot the function,  $1000 \times \sin X / X$ :

Make sure that DEG is selected as the angular unit. (To change the angular unit, enter the CAL or BASIC mode.)

1000 **×\*** SINX **÷/** X  
**←** (Enter function.)

\* GRAPH (FUNCTION) \*  
f(x) = 1000\*SINX/X  
Xmin = ?

90 **←** 1000  
(Plot the graph over the interval [90, 1000].)

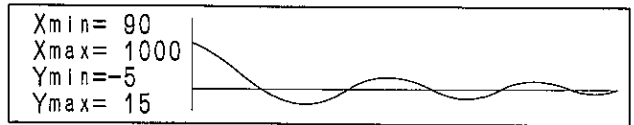
\* GRAPH (FUNCTION) \*  
f(x) = 1000\*SINX/X  
Xmin = 90  
Xmax = 1000\_




Xmin = 90  
Xmax = 1000  
Ymin = -3.789 ?  
Ymax = 11.1

The computer determines the maximum ( $Y_{\max}$ ) and minimum ( $Y_{\min}$ ) values of  $f(x)$  and displays them. To be sure that the entire graph within the specified interval is displayed, set  $Y_{\min}$  to, for example,  $-5$  and  $Y_{\max}$  to  $15$ :


$-5$    $15$  



(Graph is displayed.)

 (Function prompt returns.)

### Notes:

- If you wish to use the previous data or the expression now on the display without entering new data or a new expression, just press the  key.
- If you entered values such that  $X_{\min} \geq X_{\max}$  or  $Y_{\min} \geq Y_{\max}$ , the display will prompt for re-entry.
- If the entered values of  $X_{\min}$ ,  $X_{\max}$ ,  $Y_{\min}$ , and/or  $Y_{\max}$  exceed 5 digits including the decimal point, they will overlap on the graph when displayed.
- While a graph is being plotted, an asterisk (\*) appears to the left of " $Y_{\max}$ ". The asterisk disappears when plotting is complete. If the value of  $y$  is too large or too small, the curve will not be visible on the display. Change the range of  $X$  or  $Y$  until an adequate plotting range is obtained.
- To plot the graph, the interval between  $X_{\min}$  and  $X_{\max}$  is divided into small segments and the value of  $y$  for each segment is computed. The values of  $Y_{\min}$  and  $Y_{\max}$  are displayed only if the value of  $y$  falls into the range of  $10^{100}$  to  $10^{-3}$ . If either value exceeds this range, it will be set to zero. In such a case, enter the values of  $Y_{\min}$  and  $Y_{\max}$  that correspond to the range you wish to plot.
- If you accept the values of  $Y_{\min}$  and  $Y_{\max}$  that were computed from  $f(x)$ , a portion of the graph may be lost due to rounding error or some other cause.
- The plotted graph may appear as a series of dots, since the spaces between adjacent dots are not filled.



## \*152 GRAPH (DATA)

The GRAPH (DATA) program plots entered data. It is convenient for viewing regression curves of input data. Statistical data entered in the STAT mode can also be plotted using this program (at least two sets of data are required for plotting).

**Note:**

- If data is already stored in the STAT mode, no additional data can be entered. First delete the STAT data before you enter new data (the same applies to matrix *MD* in the MATRIX mode).

**Operation:**

[MATH]  [GRAPH]  
 [DATA]

```

* GRAPH (DATA) *
1. Clear all data  2. Input data
3. Plot
    
```

(Menu)

**1. Clear all data (for new data entry)**

- When plotting stored data:  
 If two or more sets of STAT data are already stored and you wish to plot them, this option need not be chosen. If you have inadvertently chosen this option, cancel it with  . The first and second sets of STAT data are assumed to be *x* and *y* data, respectively.
- When no data is stored or you wish to plot other data:  
 In this case, be sure to choose this option. After pressing  , specify the number of data pairs to be input.

**2. Input data (for confirmation)**

Choosing this option allows you to review or change stored data.

**3. Plot (for execution)**

Choosing this option plots the data.

**Example:**

Plot the following five pairs of data:

(8.8, 25), (5.3, 3.5), (3.2, 2.5), (1.8, 6.9), (7.2, 12.5)

("Clear all data" option is selected.)

```

* GRAPH (DATA) *
Clear ok (y/n)
    
```

(Stored data is cleared.)

```

* GRAPH (DATA) *
Number of data= ?
    
```

5  (The number of data sets to plot is 5.)

```
* GRAPH (DATA) *
1. Clear all data  2. Input data
3. Plot
```

2 (‘‘Input data’’ option is selected.)

```
* GRAPH (DATA) *
X(1)= 0 ?
```

8.8  25  5.3  3.5  3.2  2.5   
 1.8  6.9  7.2  12.5

```
* GRAPH (DATA) *
1. Clear all data  2. Input data
3. Plot
```

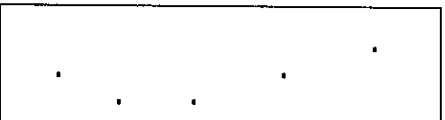
3 (‘‘Plot’’ option is selected.)

```
Xmin= 1.8 ?
Xmax= 8.8
Ymin= 2.5
Ymax= 25
```

The computer displays the maximum and minimum values of input data, which define the horizontal and vertical scale inputs. To make plotting easier to see, increase the limits, so that adequate marginal spaces are left above, below, and to each side of the graph. Here, set:  $X_{min} = 0$ ,  $X_{max} = 10$ ,  $Y_{min} = 0$ , and  $Y_{max} = 40$ .

0  10  0   
 40

```
Xmin= 0
Xmax= 10
Ymin= 0
Ymax= 40
```



(Returns to the menu.)

```
* GRAPH (DATA) *
1. Clear all data  2. Input data
3. Plot
```

### Changing Stored Data

In the following example, change the second point of data, (5.3, 3.5), to (5, 3):

2

```
* GRAPH (DATA) *
X(1)= 8.8 ?
```

```
* GRAPH (DATA) *
X(1)= 8.8
Y(1)= 25
X(2)= 5.3 ?
```

5

```
X(1)= 8.8  
Y(1)= 25  
X(2)= 5.3 5  
Y(2)= 3.5 ?
```

3

```
Y(1)= 25  
X(2)= 5.3 5  
Y(2)= 3.5 3  
X(3)= 3.2 ?
```

X\*

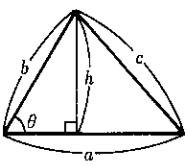
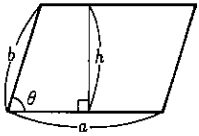
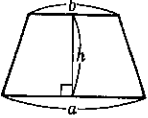
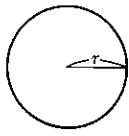
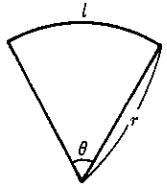
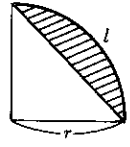
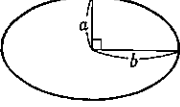
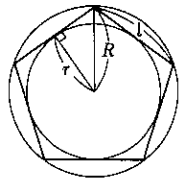
```
* GRAPH (DATA) *  
1. Clear all data 2. Input data  
3. Plot
```

**Notes:**

- If you wish to accept the data now on the display, just press the  key.
- When you no longer need to change data, you can return to the menu by pressing:  .
- The number of data points to plot, once specified, cannot be changed.
- If values are entered such that  $X_{\min} > X_{\max}$  or  $Y_{\min} > Y_{\max}$ , the computer will prompt for re-entry.
- While the data is being plotted, an asterisk (\*) appears to the left of " $Y_{\max}$ ". The asterisk disappears when plotting is complete.

## \*153 AREA OF FIGURE

The AREA OF FIGURE program calculates areas for triangles, parallelograms, trapezoids, circles, sectors, circular segments, ellipses and regular polygons. The angular unit will be set to DEG automatically.

Figure	Formula	
Triangle	(1) $S = \frac{ah}{2}$ (2) $S = \frac{1}{2} ab \sin \theta$ (3) $S = \sqrt{s(s-a)(s-b)(s-c)}$	where: $s = \frac{a+b+c}{2}$ 
Parallelogram	(4) $S = ah$ (5) $S = ab \sin \theta$	
Trapezoid	(6) $S = \frac{(a+b)}{2} h$	
Circle	(7) $S = \pi r^2$	
Sector	(8) $S = \frac{lr}{2}$ (9) $S = \pi r^2 \frac{\theta}{360}$	
Circular segment	(10) $S = \frac{1}{2} \left( lr - r^2 \sin \left( \frac{l}{r} \right) \right)$ This calculation is performed with the angular unit RAD. After the calculation, the angular unit setting will return to DEG.	
Ellipse	(11) $S = \pi ab$	
Regular polygon	(12) $S = f(n, r) = nr^2 \tan \frac{\pi}{n}$ (13) $S = f(n, R) = \frac{1}{2} nR^2 \sin \frac{2\pi}{n}$ (14) $S = f(n, l) = \frac{l^2}{4} n \cot \frac{\pi}{n}$	$n = 5$ for regular pentagon 

**Operation:**

**PF1** [MATH] **PF5** [GRAPH]  
**PF3** [FIGURE]

* AREA OF FIGURE *	
▶Triangle	$ah/2$
Triangle	$(ab \cdot \sin \theta)/2$
Triangle	$\sqrt{(S(S-a)(S-b)(S-c))}$

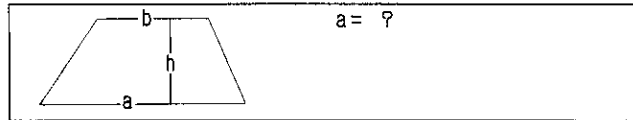
**Example 1:**

Determine the area of a trapezoid with upper length ( $b$ ), base ( $a$ ), and height ( $h$ ) given by:

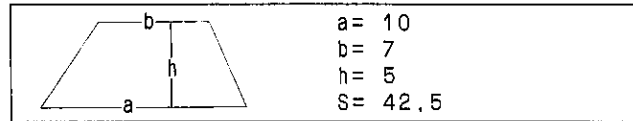
▶ ↓ ↓  
 (Specifies trapezoid.)

* AREA OF FIGURE *	
Parallelogram	$ah$
Parallelogram	$ab \cdot \sin \theta$
▶Trapezoid	$(a+b) \cdot h/2$

←



10 ← 7 ← 5 ←



The area of the trapezoid with  $a = 10$ ,  $b = 7$ , and  $h = 5$  is determined to be 42.5.

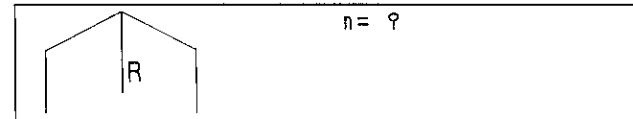
**Example 2:**

Determine the area of a regular polygon with given  $n$  and  $R$ :

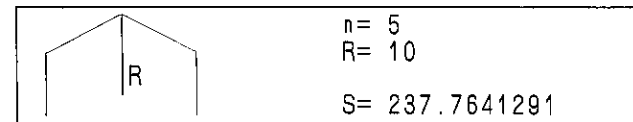
← ▶ ▶ ↓  
 (Specifies regular polygon.)

* AREA OF FIGURE *	
Ellipse	$\pi ab$
Regular polygon	$f(n, r)$
▶Regular polygon	$f(n, R)$

←



5 ← 10 ←



The area of a regular polygon with  $n = 5$  and  $R = 10$  is determined to be 237.7641291. The pointer ▶ on the menu points to the formula now selected. You can select any formula by shifting the pointer with the ↓ or ↑ key (or the ▶ or ◀ key).

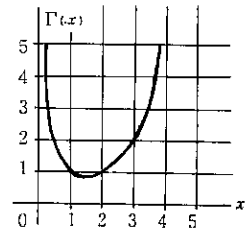
**Notes:**

- Each operation of the  $\downarrow$  key shifts the pointer down a single line, while the  $\uparrow$  key shifts it up a single line. Each operation of the  $\blacktriangleright$  key shifts the pointer down 3 lines, while the  $\blacktriangleleft$  key shifts it up 3 lines.
- If you press the  $\leftarrow$  key when the pointer is beside a specific formula, the system prompts you to enter parameters for that formula.
- Pressing the  $\leftarrow$  key after you get the answer returns the computer to the previous display.

**\*161 GAMMA FUNCTION**

The GAMMA FUNCTION program determines the value of the gamma function  $[\Gamma(x)]$  within the range of  $0 < x \leq 70$ . The result has 7 significant digits, with the 8th digit rounded off.

The gamma function appears as shown at right:



**Operation:**

- $\text{PF1}$  [MATH]
- $\blacktriangleleft$   $\text{PF1}$  [S-FUNC]
- $\text{PF1}$  [GAMMA]

```
          * GAMMA FUNCTION *
x (0<x≤70)= ?
```

(Entry prompt)

**Example:**

Determine the value of the gamma function for  $x = 3.2$ :

3.2  $\leftarrow$

```
          * GAMMA FUNCTION *
Γ(3.2) = 2.42397
```

**Note:**

- If you press only the  $\leftarrow$  key without entering the value of  $x$ , the entry prompt is displayed.

## \*211 PHYSICAL CONSTANT

The PHYSICAL CONSTANT program displays the following 47 physical constants:

Name and Symbol	Value	Unit
(1) Speed of light in vacuum $c$	$2.99792458 \times 10^8$	$\text{m} \cdot \text{s}^{-1}$
(2) Gravitational constant $G$	$6.67259 \times 10^{-11}$	$\text{N} \cdot \text{m}^2 \cdot \text{kg}^{-2}$
(3) Gravitational acceleration $g$	9.80665	$\text{m} \cdot \text{s}^{-2}$
(4) Electron rest mass $m_e$	$9.1093897 \times 10^{-31}$	kg
(5) Proton rest mass $m_p$	$1.6726231 \times 10^{-27}$	kg
(6) Neutron rest mass $m_n$	$1.6749286 \times 10^{-27}$	kg
(7) Muon rest mass $m_\mu$	$1.8835327 \times 10^{-28}$	kg
(8) Atomic mass unit $u$	$1.6605402 \times 10^{-27}$	kg
(9) Electric charge $e$	$1.60217733 \times 10^{-19}$	C
(10) Planck constant $h$	$6.6260755 \times 10^{-34}$	$\text{J} \cdot \text{s}$
(11) Boltzmann constant $k$	$1.380658 \times 10^{-23}$	$\text{J} \cdot \text{K}^{-1}$
(12) Magnetic permeability $\mu_0$	$12.566370614 \times 10^{-7}$	$\text{H} \cdot \text{m}^{-1}$
(13) Dielectric permittivity $\epsilon_0$	$8.854187817 \times 10^{-12}$	$\text{F} \cdot \text{m}^{-1}$
(14) Electron charge to mass ratio $e/m_e$	$1.75881962 \times 10^{11}$	$\text{C} \cdot \text{kg}^{-1}$
(15) Classical electron radius $r_e = e^2/4\pi\epsilon_0 m_e c^2$	$2.81794092 \times 10^{-15}$	m
(16) Fine structure constant $\alpha = e^2/4\pi\epsilon_0 \hbar c$	$7.29735308 \times 10^{-3}$	
(17) Quantum of circulation $h/2m_e$	$3.63694807 \times 10^{-4}$	$\text{J} \cdot \text{s} \cdot \text{kg}^{-1}$
(18) Bohr radius $a_0 = 4\pi\epsilon_0 \hbar^2/m_e e^2$	$5.29177249 \times 10^{-11}$	m
(19) Rydberg constant $R_\infty = e^2/16\pi^2\epsilon_0 \alpha_0 \hbar c$	$1.0973731534 \times 10^7$	$\text{m}^{-1}$
(20) Flux quantum $h/2e$	$2.06783461 \times 10^{-15}$	Wb
(21) Bohr magneton $\mu_B = e\hbar/2m_e$	$9.2740154 \times 10^{-24}$	$\text{J} \cdot \text{T}^{-1}$
(22) Electron magnetic moment $\mu_e$	$9.2847701 \times 10^{-24}$	$\text{J} \cdot \text{T}^{-1}$
(23) Free electron g-factor $2\mu_e/\mu_B$	2.002319304386	
(24) Nuclear magneton $\mu_N = e\hbar/2m_p$	$5.0507866 \times 10^{-27}$	$\text{J} \cdot \text{T}^{-1}$
(25) Proton magnetic moment $\mu_p$	$1.41060761 \times 10^{-26}$	$\text{J} \cdot \text{T}^{-1}$
(26) Proton g-factor $2\mu_p/\mu_N$	5.585694772	
(27) Gyromagnetic ratio of proton $\gamma_p$	$2.67522128 \times 10^8$	$\text{s}^{-1} \cdot \text{T}^{-1}$
(28) Neutron magnetic moment $\mu_n$	$9.6623707 \times 10^{-27}$	$\text{J} \cdot \text{T}^{-1}$
(29) Muon magnetic moment $\mu_\mu$	$4.4904514 \times 10^{-26}$	$\text{J} \cdot \text{T}^{-1}$
(30) Compton wavelength of the electron $\lambda_c = h/m_e c$	$2.42631058 \times 10^{-12}$	m

Name and Symbol	Value	Unit
(31) Compton wavelength of the proton $\lambda_{cP} = \hbar/m_P c$	$1.32141002 \times 10^{-15}$	m
(32) Stefan-Boltzmann constant $\sigma = \pi^2 k^4 / 60 \hbar^3 c^2$	$5.67051 \times 10^{-8}$	$\text{W} \cdot \text{m}^{-2} \cdot \text{K}^{-4}$
(33) Avogadro's constant $N_A$	$6.0221367 \times 10^{23}$	$\text{mol}^{-1}$
(34) Ideal gas at STP $V_0$	$2.241410 \times 10^{-2}$	$\text{m}^3 \cdot \text{mol}^{-1}$
(35) Gas constant $R = N_A k$	8.314510	$\text{J} \cdot \text{mol}^{-1} \cdot \text{K}^{-1}$
(36) Faraday constant $F = N_A e$	$9.6485309 \times 10^4$	$\text{C} \cdot \text{mol}^{-1}$
(37) Josephson frequency-voltage ratio $2e/h$	$4.8359767 \times 10^{14}$	$\text{Hz} \cdot \text{V}^{-1}$
(38) Quantum hole resistance $R_H$	25812.8056	$\Omega$
(39) Electron volt $eV$	$1.60217733 \times 10^{-19}$	J
(40) Astronomical unit $AU$	$1.49597870 \times 10^{11}$	m
(41) Parsec $pc$	$3.0856776 \times 10^{16}$	m
(42) Sea mile sea mile	1852	m
(43) Angstrom $\text{\AA}$	$1 \times 10^{-10}$	m
(44) Knot knot	1852/3600	$\text{m} \cdot \text{s}^{-1}$
(45) Torr Torr	101325/760	Pa
(46) Standard atmospheric pressure $atm$	101325	Pa
(47) Calorie cal	4.1868	J

### Operation:

**PF2** [SCI] **PF1** [PHYS]  
**PF1** [CONST]

```
* PHYSICAL CONSTANT *
1: c
= 2.99792458E+8 [m·s-1]
```

↓

```
* PHYSICAL CONSTANT *
2: G
= 6.67259E-11 [N·m2·kg-2]
```

These physical constants can be assigned to fixed variables (A – Z), so that they may be available in the CAL or BASIC mode.

Example:

Assign the gravitational constant (G) to variable A:

**STO**

```
* PHYSICAL CONSTANT *
2: G
= 6.67259E-11 [N·m2·kg-2]
STORE (A-Z) ?
```











A ←

```
* PHYSICAL CONSTANT *
2: G
   = 6.67259E-11      [N·m2·kg-2]
```

The value  $6.67259 \times 10^{-11}$  is now assigned to variable A.

**Notes:**

- The  key lets you scroll down through the chart, while the  key lets you scroll up through the chart. The  key skips two constants; the  key skips up two.
- The , , , and  keys continuously recall physical constants while they are pressed down.
- (44) knot and (45) Torr are given in the form of fractions in the chart. When they are assigned to variables, the decimal values of the fractions are assigned.
- If a constant with more than 10 digits is assigned to a variable, the 11th digit is rounded off before it is actually assigned.
- Since fixed variables with assigned constants are shared with BASIC programs, they will be cleared when the RUN or CLEAR command is executed. They are also cleared when another Engineer Software program is executed.

## \*212 METRIC CONVERSION

The METRIC CONVERSION program converts the units of length, area, volume, weight and energy.

Units of length I		m	in(inch)	ft(foot)	yd(yard)
	m	1	39.3701	3.28084	1.09361
	in	0.0254	1	0.0833333	0.0277778
	ft	0.3048	12	1	0.333333
	yd	0.9144	36	3	1
	mile	1609.344	63360	5280	1760
	cm	0.01	0.393701	0.0328084	0.0109361
	pc	$3.0856776 \times 10^{16}$	$1.214834357 \times 10^{18}$	$1.01236145 \times 10^{17}$	$3.37452788 \times 10^{16}$

Units of length II		mile	cm	Å	pc(parsec)
	m	0.000621371	100	$1 \times 10^{10}$	$3.24077927 \times 10^{-17}$
	in	$1.57828 \times 10^{-5}$	2.54	254000000	$8.231579346 \times 10^{-19}$
	ft	0.000189394	30.48	3048000000	$9.877895215 \times 10^{-18}$
	yd	0.000568182	91.44	9144000000	$2.963368564 \times 10^{-17}$
	mile	1	160934.4	$1.609344 \times 10^{13}$	$5.215528674 \times 10^{-14}$
	cm	$6.21371 \times 10^{-6}$	1	100000000	$3.24077927 \times 10^{-19}$
	pc	$1.917350576 \times 10^{13}$	$3.0856776 \times 10^{18}$	$3.0856776 \times 10^{26}$	1

### Reading the table:

To convert from "meter" to "inch", multiply by 39.3701.

Other conversions can be performed in a manner similar to the above operation.

Units of area		m <sup>2</sup>	a (are)	acre	mile <sup>2</sup>
	m <sup>2</sup>	1	0.01	0.000247105	$3.86102 \times 10^{-7}$
	a	100	1	0.0247105	$3.86102 \times 10^{-5}$
	acre	4046.86	40.4686	1	0.0015625
	mile <sup>2</sup>	2589990	25899.9	640	1

Units of volume		cm <sup>3</sup>	m <sup>3</sup>	in <sup>3</sup> (inch)	ℓ (litre)	ft <sup>3</sup> (foot)
	cm <sup>3</sup>	1	0.000001	0.0610237	0.001	$3.53147 \times 10^{-5}$
	m <sup>3</sup>	1000000	1	61023.7	1000	35.3147
	in <sup>3</sup>	16.3871	$1.63871 \times 10^{-5}$	1	0.0163871	0.000578704
	ℓ	1000	0.001	61.0237	1	0.0353147
	ft <sup>3</sup>	28316.8	0.0283168	1728	28.3168	1

Units of weight		g	kg	oz (ounce)	lb (pound)
	g	1	0.001	0.035274	0.00220462
	kg	1000	1	35.274	2.20462
	oz	28.3495	0.0283495	1	0.0625
	lb	453.59237	0.45359237	16	1

Units of energy I		eV	erg	cm <sup>-1</sup>	Hz
	eV	1	$1.60218 \times 10^{12}$	8065.54	$2.41799 \times 10^{14}$
	erg	$6.24151 \times 10^{11}$	1	$5.03411 \times 10^{15}$	$1.50919 \times 10^{26}$
	cm <sup>-1</sup>	0.000123984	$1.98645 \times 10^{16}$	1	$2.99792 \times 10^{10}$
	Hz	$4.13567 \times 10^{15}$	$6.62608 \times 10^{-27}$	$3.33564 \times 10^{-11}$	1
	K	$8.61738 \times 10^{-5}$	$1.38066 \times 10^{16}$	0.695039	$2.08367 \times 10^{10}$
	G	$5.78838 \times 10^9$	$9.27402 \times 10^{-21}$	$4.66864 \times 10^{-5}$	1399620
	J/mol	$1.03643 \times 10^{-5}$	$1.66054 \times 10^{-17}$	0.0835934	2506070000
	kcal/mol	0.0433854	$6.9511 \times 10^{-14}$	349.926	$1.04905 \times 10^{13}$

Units of energy II		K	G	J/mol	kcal/mol
	eV	11604.5	172760000	96485.3	23.0492
	erg	$7.24292 \times 10^{15}$	$1.07828 \times 10^{20}$	$6.02214 \times 10^{16}$	$1.43862 \times 10^{13}$
	cm <sup>-1</sup>	1.43877	21419.5	11.9627	0.00285774
	Hz	$4.79922 \times 10^{-11}$	$7.14478 \times 10^{-7}$	$3.99031 \times 10^{-10}$	$9.53241 \times 10^{-14}$
	K	1	14887.4	8.31451	0.00198624
	G	0.000067171	1	0.000558494	$1.33418 \times 10^{-7}$
	J/mol	0.120272	1790.53	1	0.000238889
	kcal/mol	503.463	7495250	4186.05	1

**Operation:**

**PF2** [SCI] **PF1** [PHYS]  
**PF2** [M-CONV]

```

* METRIC CONVERSION *

[LENGTH] [ AREA ] [VOLUME] [WEIGHT] [ENERGY]
```

(Menu)

Choose the item for metric conversion from the menu with the **PF1** - **PF5** key.

**Example 1:**

Convert 12 meters to inches:

**PF1** [LENGTH]

```

* METRIC CONVERSION *

(LENGTH) X= ?
```

Enter the data to convert.

12

```
          * METRIC CONVERSION *
<LENGTH> X= 12
from ?
[ m ] [ in ] [ ft ] [ yd ] [ mile ]
```

Enter the original unit for the data:

[m]

```
          * METRIC CONVERSION *
<LENGTH> X= 12
from [ m ] to ?
[ m ] [ in ] [ ft ] [ yd ] [ mile ]
```

Enter the target unit:

[in]

```
          * METRIC CONVERSION *
<LENGTH> X= 12
from [ m ] to [ in ]
          12          472.4412
```

The answer is 472.4412 inches.

Example 2:

Convert  $10^{-3}$  inch to angstroms (Å):

When the last result of metric conversion for length is still on the display, press the  key, and you can continue metric conversion for another length.

```
          * METRIC CONVERSION *
<LENGTH> X= ?
```

1  EXP -3   [in]

```
          * METRIC CONVERSION *
<LENGTH> X= 0.001
from [ in ] to ?
[ m ] [ in ] [ ft ] [ yd ] [ mile ]
```

Since angstrom (Å) is not in the menu on the display, first recall it into the display using the  key, then choose angstroms using :

[Å]

```
          * METRIC CONVERSION *
<LENGTH> X= 0.001
from [ in ] to [ Å ]
          0.001          254000
```

The answer is 254000 Å.

**Notes:**

- If the result of metric conversion exceeds the range of  $10^{-100} < X < 10^{100}$ , a message, "Answer not found" will appear. Press the  key, then enter appropriate data for conversion.
- If the  key is pressed when the prompt "X = ?" is on the display, the computer will return to the menu.

## \*213 EQUATION OF MOTION

The EQUATION OF MOTION program displays the following 21 equations of motion and energy formulas:

Name	Equation/Formula
(1) Linear motion with constant acceleration	$v = v_0 + at, a = \frac{\Delta v}{\Delta t}, s = v_0 t + \frac{1}{2} at^2$
(2) Newton's equation of motion	$F = ma$
(3) Circular motion (1)	$T = \frac{2\pi r}{v} = \frac{2\pi}{\omega} = \frac{1}{f}$
(4) Circular motion (2)	$\omega = \frac{2\pi}{T} = 2\pi f = \frac{v}{r}, F = mr\omega^2 = \frac{mv^2}{r}$
(5) Simple oscillation	$x = r \cdot \sin \omega t, v = r\omega \cdot \cos \omega t, a = -\omega^2 x$
(6) Hooke's law	$F = -kx$
(7) Spring oscillation	$a = \frac{F}{m} = -\frac{k}{m}x, T = 2\pi\sqrt{\frac{m}{k}}$
(8) Simple pendulum	$a = \frac{F}{m} = -\frac{g}{l}x, T = 2\pi\sqrt{\frac{l}{g}}$
(9) Coefficient of friction $\mu$	$F = \mu N$
(10) Work	$W = F \cdot s$
(11) Kepler's law (harmonic law)	$\frac{T^2}{r^3} = \text{Constant}$
(12) Universal gravitation	$F = G \cdot \frac{Mm}{r^2}, G = 6.67259 \times 10^{-11} [N \cdot m^2 / kg^2]$
(13) Moment of inertia I	$I = mr^2, E = \frac{1}{2} I \omega^2$
(14) Angular momentum	$J = I \omega$
(15) Law of conservation of momentum	$mv_1 + MV_1 = mv_2 + MV_2$
(16) Potential energy	$E_p = mgh$
(17) Potential energy (among planets)	$U_p = -G \frac{Mm}{r}$
(18) Kinetic energy	$E_k = \frac{1}{2} mv^2$
(19) Kinetic energy (among planets)	$E_k = \frac{1}{2} mr^2 \omega^2$
(20) Elastic energy	$E_e = \frac{1}{2} kx^2$
(21) Mass-energy relation	$E = mc^2$

**Operation:**

**PF2** [SCI] **PF1** [PHYS]  
**PF3** [MOTION]

\* EQUATION OF MOTION \*  
1:  $v=v_0+at$ ,  $a=\Delta v/\Delta t$ ,  $s=v_0t+1/2 \cdot at^2$   
[ **↵** ] [ **⬆** ] [CENTER] [ **⬇** ] [ **⬇** ]

**PF4** [ **⬇** ]

\* EQUATION OF MOTION \*  
6:  $F=-kx$   
[ **↵** ] [ **⬆** ] [CENTER] [ **⬇** ] [ **⬇** ]

Use the following keys to recall a specific equation or formula:

- PF1** [ **↵** ] : Recalls the first formula.  
**PF2** [ **⬆** ] : Recalls the formula which is five steps before the formula now displayed (i.e. the formula with current number minus 5).  
**PF3** [CENTER] : Recalls the 11th (mid) formula.  
**PF4** [ **⬇** ] : Recalls the formula which is five steps beyond the formula now displayed (i.e. the formula with current number plus 5).  
**PF5** [ **⬇** ] : Recalls the 21st (last) formula.  
[ **⬇** ] : Recalls the formula which just follows the formula now displayed.  
[ **⬆** ] : Recalls the formula which just precedes the formula now displayed.

**Note:**

- The **PF2**, **PF4**, [ **⬇** ], and [ **⬆** ] keys continuously recall formulas while they are pressed down.

## \*221 PERIODIC TABLE

The PERIODIC TABLE program displays the periodic table of the elements or the atomic weight by entering an atomic number:

### (1) Periodic Table of the Elements

The number refers to the atomic number.

Periods Groups	1A	2A	3A	4A	5A	6A	7A	8										1B	2B	3B	4B	5B	6B	7B	0
1	1 H																				2 He				
2	3 Li	4 Be											5 B	6 C	7 N	8 O	9 F	10 Ne							
3	11 Na	12 Mg											13 Al	14 Si	15 P	16 S	17 Cl	18 Ar							
4	19 K	20 Ca	21 Sc	22 Ti	23 V	24 Cr	25 Mn	26 Fe	27 Co	28 Ni	29 Cu	30 Zn	31 Ga	32 Ge	33 As	34 Se	35 Br	36 Kr							
5	37 Rb	38 Sr	39 Y	40 Zr	41 Nb	42 Mo	43 Tc	44 Ru	45 Rh	46 Pd	47 Ag	48 Cd	49 In	50 Sn	51 Sb	52 Te	53 I	54 Xe							
6	55 Cs	56 Ba	57~ 71	72 Hf	73 Ta	74 W	75 Re	76 Os	77 Ir	78 Pt	79 Au	80 Hg	81 Tl	82 Pb	83 Bi	84 Po	85 At	86 Rn							
7	87 Fr	88 Ra	89~ 103																						

*Lanthanides	57 La	58 Ce	59 Pr	60 Nd	61 Pm	62 Sm	63 Eu	64 Gd	65 Tb	66 Dy	67 Ho	68 Er	69 Tm	70 Yb	71 Lu
**Actinides	89 Ac	90 Th	91 Pa	92 U	93 Np	94 Pu	95 Am	96 Cm	97 Bk	98 Cf	99 Es	100 Fm	101 Md	102 No	103 Lr

#### Operation:

**PF2** [SCI] **PF2** [CHEM]  
**PF1** [PERIOD]

\* PERIODIC TABLE \*

Display table...1 Atomic weight...2

(Menu)

1

1a	2a	3a	4a	5a	6a	7a	8	8	8	1b	2b	3b
1: H												
2: Li	Be											B
3: Na	Mg											Al

The rest of the above table can be displayed by pressing the **▶**, **◀**, **↓**, and **↑** keys. Pressing the **←** key will return to the menu.

### (2) Atomic Weight

This periodic table lists the atomic weights which were recommended by the Atomic Weight Committee (1985) of IUPAC (International Union of Pure & Applied Chemistry) by referring to  $^{12}\text{C} = 12$ . It applies to natural substances of the earth. The accuracy is the last significant digit  $\pm 7$  for hydrogen, and the last significant digit  $\pm 3$  for other substances. Numbers given in brackets [ ] indicate the masses of the isotopes (with known longest half-lives) of the relevant substances.

• Atomic weight (1)

Atomic number	Element	Symbol	Atomic weight
1	Hydrogen	H	1.00794
2	Helium	He	4.002602
3	Lithium	Li	6.941
4	Beryllium	Be	9.012182
5	Boron	B	10.811
6	Carbon	C	12.011
7	Nitrogen	N	14.00674
8	Oxygen	O	15.9994
9	Fluorine	F	18.9984032
10	Neon	Ne	20.179
11	Sodium	Na	22.989768
12	Magnesium	Mg	24.3050
13	Aluminium	Al	26.981539
14	Silicon	Si	28.0855
15	Phosphorus	P	30.973762
16	Sulfur	S	32.066
17	Chlorine	Cl	35.4527
18	Argon	Ar	39.948
19	Potassium	K	39.0983
20	Calcium	Ca	40.078
21	Scandium	Sc	44.955910
22	Titanium	Ti	47.88
23	Vanadium	V	50.9415
24	Chromium	Cr	51.9961
25	Manganese	Mn	54.93805
26	Iron	Fe	55.847
27	Cobalt	Co	58.93320
28	Nickel	Ni	58.69
29	Copper	Cu	63.546
30	Zinc	Zn	65.39
31	Gallium	Ga	69.723
32	Germanium	Ge	72.61
33	Arsenic	As	74.92159
34	Selenium	Se	78.96
35	Bromine	Br	79.904
36	Krypton	Kr	83.80
37	Rubidium	Rb	85.4678
38	Strontium	Sr	87.62
39	Yttrium	Y	88.90585
40	Zirconium	Zr	91.224



• Atomic weight (2)

Atomic number	Element	Symbol	Atomic weight
41	Niobium	Nb	92.90638
42	Molybdenum	Mo	95.94
43	Technetium	Tc	[98]
44	Ruthenium	Ru	101.07
45	Rhodium	Rh	102.90550
46	Palladium	Pd	106.42
47	Silver	Ag	107.8682
48	Cadmium	Cd	112.411
49	Indium	In	114.82
50	Tin	Sn	118.710
51	Antimony	Sb	121.75
52	Tellurium	Te	127.60
53	Iodine	I	126.90447
54	Xenon	Xe	131.29
55	Caesium	Cs	132.90543
56	Barium	Ba	137.327
57	Lanthanum	La	138.9055
58	Cerium	Ce	140.115
59	Praseodymium	Pr	140.90765
60	Neodymium	Nd	144.24
61	Promethium	Pm	[145]
62	Samarium	Sm	150.36
63	Europium	Eu	151.965
64	Gadolinium	Gd	157.25
65	Terbium	Tb	158.92534
66	Dysprosium	Dy	162.50
67	Holmium	Ho	164.93032
68	Erbium	Er	167.26
69	Thulium	Tm	168.93421
70	Ytterbium	Yb	173.04
71	Lutetium	Lu	174.967
72	Hafnium	Hf	178.49
73	Tantalum	Ta	180.9479
74	Tungsten	W	183.85
75	Rhenium	Re	186.207
76	Osmium	Os	190.2
77	Iridium	Ir	192.22
78	Platinum	Pt	195.08
79	Gold	Au	196.96654
80	Mercury	Hg	200.59

• **Atomic weight (3)**

Atomic number	Element	Symbol	Atomic weight
81	Thallium	Tl	204.3833
82	Lead	Pb	207.2
83	Bismuth	Bi	208.98037
84	Polonium	Po	[209]
85	Astatine	At	[210]
86	Radon	Rn	[222]
87	Francium	Fr	[223]
88	Radium	Ra	[226]
89	Actinium	Ac	[227]
90	Thorium	Th	232.0381
91	Protactinium	Pa	231.03588
92	Uranium	U	238.0289
93	Neptunium	Np	[237]
94	Plutonium	Pu	[244]
95	Americium	Am	[243]
96	Curium	Cm	[247]
97	Berkelium	Bk	[247]
98	Californium	Cf	[251]
99	Einsteinium	Es	[252]
100	Fermium	Fm	[257]
101	Mendelevium	Md	[258]
102	Nobelium	No	[259]
103	Lawrencium	Lr	[260]

**Operation:**

```

* PERIODIC TABLE *
Display table...1 Atomic weight...2
    
```

(Menu)

2







```

* PERIODIC TABLE *
Atomic Symbol= ?
    
```

(Entry prompt)

Entering an atomic symbol will display the information on that element.

**Notes:**

- Pressing  only on the entry prompt will be displayed from the hydrogen item.
- Pressing  or  will scroll down or up one line.
- Pressing  or  will scroll down or up 10 lines.
- Pressing  will return to the menu.

## \*222 OUTER ELECTRON

The OUTER ELECTRON program displays the following items by entering an atomic symbol:

### ground state

**atomic:** Ground state of an atom

**ionic:** Ground state of an ion

### ionization voltage

**1st:** The 1st ionization potential (V)

**2nd:** The 2nd ionization potential (V)

The ionization potential is the energy per unit charge needed to remove an electron from a given kind of atom to an infinite distance and is usually expressed in volts.


### Operation:

**PF2** [SCI] **PF2** [CHEM]







**PF2** [ELECTR]

* OUTER ELECTRON *
Atomic Symbol = ?

(Entry prompt)

Entering an atomic symbol and then pressing  will display the above items for the atom.

### Notes:

- Pressing only  on the entry prompt will display the items from the hydrogen (H) atom.
- Pressing  or  will display the next or previous element.
- Pressing  or  will display the 10th element after or before the current element.
- Pressing  will return to the entry prompt.

## \*223 STABLE ISOTOPE

The STABLE ISOTOPE program displays the following items by entering an atomic symbol:


**isotope:** Isotope  
**mass:** Mass  
**abundance:** Abundance ratio

### Operation:







**PF2** [SCI] **PF2** [CHEM]  
**PF3** [ISOTOP]

```
      * STABLE ISOTOPE *  
  
Atomic Symbol= ?
```

(Entry prompt)

Entering the atomic symbol and then pressing  will display the above items for the atom sequentially.

### Notes:

- Pressing only  on the entry prompt will display the items from the hydrogen (H) atom.
- Pressing  or  will scroll down or up one line.
- Pressing  or  will scroll down or up 10 lines.
- Pressing  will return to the entry prompt.

## \*231 METEOROLOGY

The METEOROLOGY program converts the atmospheric pressure and wind speed.

Atmospheric pressure		atm	mb	mmHg
	atm	1	1013.25	760
	mb	$9.869232667 \times 10^{-4}$	1	0.75
	mmHg	$1.315789473 \times 10^{-3}$	1.33	1

### Reading the table:

To convert from "atm" to "mb", multiply by 1013.25.

Other conversions can be performed with the same operation.

Wind speed		m/s	knot	km/day
	m/s	1	1.944	86.4
	knot	$5.144032921 \times 10^{-1}$	1	44.44444444
	km/day	$1.157407407 \times 10^{-2}$	0.0225	1

The computer uses the Beaufort wind force table for conversion among m/s, knot, and force (wind velocity class). For conversion between force and km/day, it uses a table that correlates force with m/s.

### Operation:

PF2 [SCI] PF3 [EARTH]  
PF1 [METEO]

```

                * METEOROLOGY *
atmospheric pressure ...1
wind velocity       ...2
    
```

(Menu)

### Example:

Convert 960 mb to atm:

(selects atmospheric pressure.)

```

                * atmospheric pressure *
X= ?
    
```

Enter the source data (960):

960

```

                * atmospheric pressure *
X= 960
from ?
[ atm ] [ mb ] [ mmHg ]
    
```

Specify the atmospheric pressure of the source data:

[mb]

```

                * atmospheric pressure *
X= 960
from [ mb ] to ?
[ atm ] [ mb ] [ mmHg ]
    
```

Specify the units of converted data:

**PF1** [atm]

```

* atmospheric pressure *
X= 960
from [ mb ] to [ atm ]
      960      0.947446336
    
```

The answer is 0.947446336 atm.

**Notes:**

- If the result of metric conversion exceeds the range of  $10^{-100} < X < 10^{100}$ , a message, "Answer not found" will appear. Press the **←** key, then enter appropriate data for conversion.
- If the **←** key is pressed while the prompt "X = ?" is on the display, the menu will be displayed.

**\*232 SOLAR/PLANETARY CONSTANTS**

The SOLAR/PLANETARY CONSTANTS program displays the contents of the following table:

	Major distance from sun <i>a</i> (AU)	Eccentricity <i>e</i>	Equatorial radius km	Surface gravity (earth=1)	Volume (earth=1)	Mass (earth=1)	Rotation period (days)	Period of sidereal revolution P (tropical years)
Sun	—	—	696000	28.01	1304000	332946	25.38	—
Mercury	0.3871	0.2056	2439	0.38	0.056	0.055	58.65	0.2409
Venus	0.7233	0.0068	6052	0.91	0.857	0.815	243.01	0.6152
Earth	1.0000	0.0167	6378	1.00	1.000	1.000	0.9973	1.0000
Mars	1.5237	0.0934	3397	0.38	0.151	0.107	1.0260	1.8809
Jupiter	5.2026	0.0485	71398	2.37	1316	317.832	0.414	11.862
Saturn	9.5549	0.0556	60000	0.95	745	95.16	0.444	29.458
Uranus	19.2184	0.0463	25400	0.89	61	14.50	0.649	84.022
Neptune	30.1104	0.0090	24300	1.19	54	17.22	0.768	164.774
Pluto	39.5399	0.2490	2000?	0.02?	0.03?	0.0023?	6.387	248.534
Moon	—	—	1738	0.17	0.0203	0.012300	27.3217	—

**Operation:**

**PF2** [SCI] **PF3** [EARTH]  
**PF2** [PLANET]

```

* SOLAR/PLANETARY CONSTANTS *
0:Sun 1:Mercury 2:Venus 3:Earth
4:Mars 5:Jupiter 6:Saturn 7:Uranus
8:Neptune 9:Pluto M:Moon
    
```

(Initial menu)

Display the constants for Venus:

(Venus is selected.)

```
<< Venus >>a= 0.7233 [AU] e= 0.0068
radius= 6052 [km] gravity= 0.91
volume= 0.857 mass = 0.815
rotation= 243.01 [day] P= 0.6152 [year]
```

**Notes:**

- Pressing the  key will return to the initial menu.
- Pressing the  or  key will display the items of the next or previous planet.

## \*241 AMINO ACIDS FORMULAS

The AMINO ACIDS FORMULAS program displays the following major amino acid chemical structure formulas:

- 1...**Monoamino-monocarboxylic acid** (neutral amino acid)
- 2...**Monoamino-dicarboxylic acid** (acidic amino acid)
- 3...**Diamino-monocarboxylic acid** (basic amino acid)

Amino acid	Abbreviation		Amino acid	Abbreviation	
	3 letters	1 letter		3 letters	1 letter
<b>Neutral amino acid</b>	<b>Hydrophobic amino acid</b>		<b>Neutral amino acid</b>	<b>Hydropholic amino acid</b>	
glycine	Gly	G	serine	Ser	S
alanine	Ala	A	threonine	Thr	T
valine	Val	V	cysteine	Cys	C
leucine	Leu	L	tyrosine	Tyr	Y
isoleucine	Ile	I	asparagine	Asn	N
methionine	Met	M	glutamine	Gln	Q
proline	Pro	P			
phenylalanine	Phe	F			
tryptophane	Trp	W			
<b>Acidic amino acid</b>			<b>Basic amino acid</b>		
aspartic acid	Asp	D	lysine	Lys	K
glutamic acid	Glu	E	arginine	Arg	R
			histidine	His	H

**Operation:**

**PF2** [SCI] **PF4** [BIO]

**PF1** [AMINO]

```
* AMINO ACIDS FORMULAS *
1...Monoamino-monocarboxylic acid
2...Monoamino-dicarboxylic acid
3...Diamino-monocarboxylic acid
```






(Menu)

Select the type of amino acid:

**1** (Neutral amino acid is selected.)

```
G (Gly) glycine
      CH2 (NH2) COOH
A (Ala) alanine
      CH3 CH (NH2) COOH
```

**Notes:**

- Pressing the  key will return to the menu.
- Pressing the  or  key will scroll down or up four lines (one screen).
- Pressing the  or  key will scroll down or up 16 lines (four screens).



## \*311 COMPLEX NUMBER

The COMPLEX NUMBER program performs complex number calculations using the following key operations:

Key	Function
$\boxed{X}$	Stores the value into $X$ (in the order of the real and imaginary parts).
$\boxed{Y}$	Stores the value into $Y$ (in the order of the real and imaginary parts).
$\boxed{+}$	$X + Y \rightarrow X$ : Performs addition and stores the result into $X$ .
$\boxed{-}$	$X - Y \rightarrow X$ : Performs subtraction and stores the result into $X$ .
$\boxed{X*}$	$X*Y \rightarrow X$ : Performs multiplication and stores the result into $X$ .
$\boxed{\div/}$	$X/Y \rightarrow X$ : Performs division and stores the result into $X$ .
$\boxed{M+}$	$X + M \rightarrow M$ : Adds the value of $X$ to the memory.
$\boxed{RM}$	$M \rightarrow X$ : Recalls the value in memory and assigns it to $X$ .
$\boxed{X\rightarrow M}$	$X \rightarrow M$ : Stores the value of $X$ into the memory.
$\boxed{\leftrightarrow}$	$X \leftrightarrow Y$ : Exchanges the values of $X$ and $Y$ .
$\boxed{\sqrt{\quad}}$	$\sqrt{X} \rightarrow X$ : Calculates the square root of $X$ and stores the result into $X$ .
$\boxed{1/X}$	$1/X \rightarrow X$ : Calculates the reciprocal of $X$ and stores the result into $X$ .
$\boxed{X^2}$	$X*X \rightarrow X$ : Calculates the square of $X$ and stores the result into $X$ .
$\boxed{\begin{matrix} \rightarrow xy \\ ( \end{matrix}}$	<div style="display: flex; align-items: center;"> <div style="margin-right: 20px;">                     Absolute value of <math>X \rightarrow X</math>                      Argument of <math>X \rightarrow Y</math> </div> <div style="font-size: 2em;">}</div> <div>                     Stores the absolute value of <math>X</math> (<math>\sqrt{X_R^2 + X_I^2}</math>) into <math>X</math> and the argument of <math>X</math> (<math>\tan^{-1} \frac{X_I}{X_R}</math>) into <math>Y</math> where <math>X_R</math> is the real part of <math>X</math> and <math>X_I</math> the imaginary part of <math>X</math>.                 </div> </div>

### Operation:

$\boxed{PF3}$  [ENG]  $\boxed{PF1}$  [ELEC]  
 $\boxed{PF1}$  [COMPLX]

\* COMPLEX NUMBER \*

X = 0  
 Y = 0  
 (< X, Y, +-\*/ , M+, RM, X→M, ↓, √, 1/x, x<sup>2</sup>, →xy >)

### Example:

For  $X = 3 + 4i$  and  $Y = 6 + 9i$ , add  $Y$  to  $X$ , then square the result:

$\boxed{X}$  ( $X$  is selected from the menu.)

\* COMPLEX NUMBER \*

X(real) = 0 ?  
 (< X, Y, +-\*/ , M+, RM, X→M, ↓, √, 1/x, x<sup>2</sup>, →xy >)

3  4

```
* COMPLEX NUMBER *
X(real) = 3
X(image) = 0 4_
<< X,Y,+* /,M+,RM,x→M,↓,√,1/x,x²,→xy >>
```

```
* COMPLEX NUMBER *
X= 3 + 4i
Y= 0
<< X,Y,+* /,M+,RM,x→M,↓,√,1/x,x²,→xy >>
```

(Y is selected from the menu.)

```
* COMPLEX NUMBER *
Y(real) = 0 ?
<< X,Y,+* /,M+,RM,x→M,↓,√,1/x,x²,→xy >>
```

6  9

```
* COMPLEX NUMBER *
X= 3 + 4i
Y= 6 + 9i
<< X,Y,+* /,M+,RM,x→M,↓,√,1/x,x²,→xy >>
```

```
* COMPLEX NUMBER *
X= 9 + 13i
Y= 6 + 9i
<< X,Y,+* /,M+,RM,x→M,↓,√,1/x,x²,→xy >>
```

You get  $(3 + 4i) + (6 + 9i) = 9 + 13i$ . Now square the sum:

```
* COMPLEX NUMBER *
X= -88 + 234i
Y= 6 + 9i
<< X,Y,+* /,M+,RM,x→M,↓,√,1/x,x²,→xy >>
```

The final answer is:  $(9 + 13i)^2 = -88 + 234i$ .

## \*312 ELECTRICAL FORMULAS

The ELECTRICAL FORMULAS program displays the following 14 electrical formulas:

Name	Formula
(1) Ohm's law	$V = IR \quad (I = \frac{Q}{t}, R = \rho \cdot \frac{l}{S})$
(2) Direct-current power and Joules	$P = IV = I^2R, W = IVt = Pt$
(3) Conductance	$G = \frac{1}{R} = \frac{I}{V}$
(4) Kirchhoff's law	$\sum I_n = 0, \sum E_j - \sum R_i I_i = 0$
(5) Delta $\rightarrow$ Y transformation	$(\Delta \rightarrow Y) R = R_1 + R_2 + R_3$ $R_4 = \frac{R_1 R_2}{R}, R_5 = \frac{R_2 R_3}{R}, R_6 = \frac{R_3 R_1}{R}$
(6) Y $\rightarrow$ Delta transformation	$(Y \rightarrow \Delta) R = R_4 R_5 + R_5 R_6 + R_6 R_4$ $R_1 = \frac{R}{R_5}, R_2 = \frac{R}{R_6}, R_3 = \frac{R}{R_4}$
(7) Instantaneous value (alternating pressure and current)	$V = V_0 \sin \omega t, I = I_0 \sin \omega t$
(8) Effective value (alternating current)	$I = \frac{I_0}{\sqrt{2}}, V = \frac{V_0}{\sqrt{2}}$
(9) Electric power (alternating current)	$P = VI = \frac{1}{2} V_0 I_0$
(10) Power factor ( $\cos \phi$ )	$P = VI \cdot \cos \phi$
(11) Reactance	$X = \omega L = 2\pi f L, X = \frac{1}{\omega C} = \frac{1}{2\pi f C}$
(12) Impedance	$Z = \sqrt{R^2 + (\omega L - \frac{1}{\omega C})^2}, V_0 = Z I_0$
(13) Natural frequency	$f_0 = \frac{1}{2\pi \sqrt{LC}}$
(14) Electric resonance	$\frac{1}{2} \frac{Q^2}{C} + \frac{1}{2} L I^2 = \text{Constant}$

### Operation:

PF3 [ENG] PF1 [ELEC]  
 PF2 [EE-FRM]

\* ELECTRICAL FORMULAS \*

1: V=IR (I=Q/t, R=ρ·l/S)

[ ⏴ ] [ ⏵ ] [CENTER] [ ⏶ ] [ ⏷ ]







PF4 [⏴]

\* ELECTRICAL FORMULAS \*



6: (Y→Δ) R=R<sub>4</sub>R<sub>5</sub>+R<sub>5</sub>R<sub>6</sub>+R<sub>6</sub>R<sub>4</sub>, R<sub>1</sub>=R/R<sub>5</sub>,  
 R<sub>2</sub>=R/R<sub>6</sub>, R<sub>3</sub>=R/R<sub>4</sub>

[ ⏴ ] [ ⏵ ] [CENTER] [ ⏶ ] [ ⏷ ]

Use the following keys to recall a specific electrical formula:

- |  |  |
|--|--|
| <b>PF1</b>  | : Recalls the first formula.   |
| <b>PF2</b>  | : Recalls the formula which is five steps before the formula now displayed (i.e. the formula with current number minus 5). |
| <b>PF3</b> [CENTER]  | : Recalls the 7th (mid) formula.   |
| <b>PF4</b>  | : Recalls the formula which is five steps beyond the formula now displayed (i.e. the formula with current number plus 5).  |
| <b>PF5</b>  | : Recalls the 14th (last) formula.   |
|             | : Recalls the formula which just follows the formula now displayed.  |
|             | : Recalls the formula which just precedes the formula now displayed.   |

**Note:**

- The **PF2**, **PF4**, , and  keys continuously recall formulas while they are pressed down.

## \*313 ELECTRIC & MAGNETIC FIELDS

The ELECTRIC & MAGNETIC FIELDS program displays the following 16 electric and magnetic field formulas:

Name	Formula
(1) Maxwell's equations	$\text{rot } \mathbf{H} = \mathbf{i}, \text{ rot } \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}, \text{ div } \mathbf{B} = 0, \text{ div } \mathbf{D} = \rho$
(2) Coulomb's law (electric field)	$F = \frac{Q_1 Q_2}{4\pi\epsilon_0 r^2} = \frac{k_0 Q_1 Q_2}{r^2}, k_0 = 9 \times 10^9$
(3) Electric field	$E = \frac{V}{d}, F = QE, W = QV$
(4) Electric capacitance C	$Q = CV, C = \epsilon_0 \cdot \frac{S}{d}$
(5) Electrostatic energy	$U = \frac{1}{2} QV = \frac{1}{2} CV^2$
(6) Electron in electric field	$a = \frac{QE}{m}, \frac{1}{2} mv^2 = eV$
(7) Coulomb's law (magnetic field)	$F = \frac{m_1 m_2}{4\pi\mu_0 r^2} = \frac{k_0 m_1 m_2}{r^2}, k_0 = \frac{10^7}{(4\pi)^2}$
(8) Magnetic field H	$H = \frac{I}{2\pi r}, H = \frac{IN}{2r}, H = In$
(9) Magnetic field	$F = \mu_0 I H l = I B l$
(10) Magnetic flux density	$B = \frac{m}{4\pi r^2} = \mu_0 H$
(11) Lorentz force	$F = QvB, r = \frac{mv}{QB}$
(12) Electron in magnetic field	$\frac{1}{2} mv^2 = \frac{Q^2 B^2 r^2}{2m}, \omega = \frac{v}{r} = \frac{QB}{m}$
(13) Faraday's law of electromagnetic induction	$V = -n \frac{\Delta\phi}{\Delta t}$
(14) Electromagnetic induction	$V = El = vBl, I = \frac{vBl}{R}$
(15) Mutual induction	$V_2 = -M \frac{\Delta I_1}{\Delta t}$
(16) Self induction	$V' = -L \frac{\Delta I}{\Delta t}, L = \frac{N\phi}{I}$

**Operation:**

**PF3** **[ENG]** **PF1** **[ELEC]**  
**PF3** **[ELEMAG]**

```
* ELECTRIC & MAGNETIC FIELDS *
1: rotH=I, rotE=-∂B/∂t, divB=0, divD=ρ
[ ↑ ] [ ← ] [CENTER] [ → ] [ ↓ ]
```

**PF4** **[↵]**

```
* ELECTRIC & MAGNETIC FIELDS *
6: a=QE/m, 1/2·mv2=eV
[ ↑ ] [ ← ] [CENTER] [ → ] [ ↓ ]
```

Use the following keys to recall a specific electric or magnetic formula:

- PF1** **[↑]** : Recalls the first formula.
- PF2** **[←]** : Recalls the formula which is five steps before the formula now displayed (i.e. the formula with current number minus 5).
- PF3** **[CENTER]** : Recalls the 8th (mid) formula.
- PF4** **[↵]** : Recalls the formula which is five steps beyond the formula now displayed (i.e. the formula with current number plus 5).
- PF5** **[↓]** : Recalls the 16th (last) formula.
- [↓]** : Recalls the formula which just follows the formula now displayed.
- [↑]** : Recalls the formula which just precedes the formula now displayed.

**Note:**

- The **PF2**, **PF4**, **[↓]**, and **[↑]** keys continuously recall formulas while they are pressed down.

## \*314 LAPLACE TRANSFORMATION

The LAPLACE TRANSFORMATION program displays the following 46 Laplace transform formulas:

$F(s)$	$f(t)$
(1) $\frac{1}{s}$	1
(2) $\frac{1}{s^2}$	$t$
(3) $\frac{1}{s^n}$	$\frac{t^{n-1}}{(n-1)!} \quad (n=1, 2, 3, \dots)$
(4) $\frac{n!}{s^{n+1}}$	$t^n$
(5) $\frac{1}{\sqrt{s}}$	$\frac{1}{\sqrt{\pi t}}$
(6) $\frac{\Gamma(n+1)}{s^{n+1}}$	$t^n \quad (n > -1)$
(7) 1	$\delta(t)$
(8) $s$	$\delta'(t)$
(9) $\frac{1}{s \pm m}$	$e^{\mp mt}$
(10) $\frac{1}{s(s+m)}$	$\frac{1}{m}(1 - e^{-mt})$
(11) $\frac{1}{s^2(s+m)}$	$\frac{1}{m^2}(e^{-mt} + mt - 1)$
(12) $\frac{a}{s^2+a^2}$	$\sin at$
(13) $\frac{s}{s^2+a^2}$	$\cos at$
(14) $\frac{1}{s^2+a^2}$	$\frac{1}{a}\sin at$
(15) $\frac{a}{s^2-a^2}$	$\sinh at$
(16) $\frac{s}{s^2-a^2}$	$\cosh at$
(17) $\frac{1}{s^2-a^2}$	$\frac{1}{a}\sinh at$
(18) $\frac{1}{s(s^2+a^2)}$	$\frac{1}{a^2}(1 - \cos at)$
(19) $\frac{1}{s^2(s^2+a^2)}$	$\frac{1}{a^3}(at - \sin at)$
(20) $\frac{1}{(s+m)(s+n)}$	$\frac{1}{n-m}(e^{-mt} - e^{-nt})$

	$F(s)$	$f(t)$
(21)	$\frac{s}{(s+m)(s+n)}$	$\frac{1}{m-n}(me^{-mt} - ne^{-nt})$
(22)	$\frac{1}{(s+m)^2}$	$te^{-mt}$
(23)	$\frac{1}{(s+m)^n}$	$\frac{1}{(n-1)!}t^{n-1}e^{-mt} \quad (n=1, 2, 3, \dots)$
(24)	$\frac{s}{(s+m)^2}$	$e^{-mt}(1-mt)$
(25)	$\frac{1}{s(s+m)^2}$	$\frac{1}{m^2}(1 - (1+mt)e^{-mt})$
(26)	$\frac{1}{s^2(s+m)^2}$	$\frac{t}{m^2} - \frac{2}{m^3} + \frac{2e^{-mt}}{m^3} + \frac{te^{-mt}}{m^2}$
(27)	$\frac{s+n}{(s+m)^2}$	$((n-m)t+1)e^{-mt}$
(28)	$\frac{\omega \cdot \cos \theta + s \cdot \sin \theta}{s^2 + \omega^2}$	$\sin(\omega t + \theta)$
(29)	$\frac{s \cdot \cos \theta - \omega \cdot \sin \theta}{s^2 + \omega^2}$	$\cos(\omega t + \theta)$
(30)	$\frac{1}{(s^2 + a^2)^2}$	$\frac{1}{2a^3}(\sin at - at \cdot \cos at)$
(31)	$\frac{s}{(s^2 + a^2)^2}$	$\frac{t}{2a} \sin at$
(32)	$\frac{s^2}{(s^2 + a^2)^2}$	$\frac{1}{2a}(\sin at + at \cdot \cos at)$
(33)	$\frac{s^2 - a^2}{(s^2 + a^2)^2}$	$t \cdot \cos at$
(34)	$\frac{1}{(s+m)^2 + n^2}$	$\frac{1}{n} e^{-mt} \sin nt$
(35)	$\frac{s+m}{(s+m)^2 + n^2}$	$e^{-mt} \cos nt$
(36)	$\frac{\Gamma(n)}{(s-a)^n}$	$t^{n-1} \cdot e^{at} \quad (n > 0)$
(37)	$\frac{s}{(s^2 + m^2)(s^2 + n^2)}$	$\frac{\cos mt - \cos nt}{n^2 - m^2} \quad (m^2 \neq n^2)$
(38)	$\frac{1}{\sqrt{s^2 + a^2}}$	$J_0(at)$
(39)	$\frac{1}{s^4 - a^4}$	$\frac{1}{2a^3}(\sinh at - \sin at)$
(40)	$\frac{s}{s^4 - a^4}$	$\frac{1}{2a^2}(\cosh at - \cos at)$



$F(s)$	$f(t)$
(41) $\frac{s^2}{s^4 - a^4}$	$\frac{1}{2a}(\sinh at + \sin at)$
(42) $\frac{s^3}{s^4 - a^4}$	$\frac{1}{2}(\cosh at + \cos at)$
(43) $\frac{s}{s^4 + 4a^4}$	$\frac{1}{2a^2} \cdot \sin at \cdot \sinh at$
(44) $\frac{4a^3}{s^4 + 4a^4}$	$\sin at \cdot \cosh at - \cos at \cdot \sinh at$
(45) $\frac{1}{s} \left( \frac{s-m}{s+m} \right)$	$-1 + 2e^{-mt}$
(46) $\frac{1}{s^2} \left( \frac{s-m}{s+m} \right)$	$\frac{2}{m} - t - \frac{2}{m} \cdot e^{-mt}$

**Operation:**

**PF3** [ENG] **PF1** [ELEC]  
**PF4** [LAPLAC]

```

* LAPLACE TRANSFORMATION *
1:1 / s
  1
[  F ] [  A ] [CENTER] [  V ] [  D ]

```

**PF4** [V]

```

* LAPLACE TRANSFORMATION *
6: Γ(n+1) / s^{n+1}
   t^n      (n > -1)
[  F ] [  A ] [CENTER] [  V ] [  D ]

```

Use the following keys to recall a specific transform formula:

- PF1** [F] : Recalls the first formula.
- PF2** [A] : Recalls the formula which is five steps before the formula now displayed (i.e. the formula with current number minus 5).
- PF3** [CENTER] : Recalls the 23rd (mid) formula.
- PF4** [V] : Recalls the formula which is five steps beyond the formula now displayed (i.e. the formula with current number plus 5).
- PF5** [D] : Recalls the 46th (last) formula.
- [D] : Recalls the formula which just follows the formula now displayed.
- [A] : Recalls the formula which just precedes the formula now displayed.

**Note:**

- The **PF2**, **PF4**, [D], and [A] keys continuously recall formulas while they are pressed down.

## \*321 MECHANICAL FORMULAS

The MECHANICAL FORMULAS program displays the following 12 mechanical formulas and performs calculations with the entered value.

	Name	Formula	Input value	Output value
<b>Gear ratio</b>	M : module	(1) $M = D_1/Z_1$	$D_1, Z_1$	$M$
	$D_1$ : driving gear pitch diameter	(2) $D_2 = D_1 Z_2 / Z_1$	$D_1, Z_2, Z_1$	$D_2$
	$D_2$ : driven gear pitch diameter $Z_1$ : number of driving gear teeth $Z_2$ : number of driven gear teeth $P$ : pitch	(3) $M = P/\pi$	$P$	$M$
<b>Hydrodynamic formulas</b>	Bernoulli's theorem $P$ : fluid pressure $\rho$ : mass density of the fluid $v$ : fluid velocity $z$ : vertical height $g$ : acceleration of gravity	(4) $v^2/2 + P/\rho + gz = \text{Constant}$	$P_1, P_2, \rho,$ $v_1, g, z_1, z_2$	$v_2$
	Equation of continuity $A$ : cross section of tube $\rho$ : mass density of the fluid $v$ : fluid velocity	(5) $A_1 v_1 = A_2 v_2 = \text{Constant}$	$A_1, v_1, \rho_1,$ $A_2, \rho_2$	$v_2$
	Reynolds number $L$ : bore of tube $U$ : fluid velocity $\nu$ : kinematic viscosity	(6) $R = LU/\nu$	$L, U, \nu$	$R$
<b>Thermodynamic</b>	Enthalpy $U$ : internal energy $p$ : fluid pressure $V$ : volume	(7) $H = U + pV$	$U, p, V$	$H$
	Carnot's cycle (heat quantity) $Q$ : heat quantity $\eta$ : thermal efficiency	(8) $\eta = (Q_1 - Q_2)/Q_1$	$Q_1, Q_2$	$\eta$
	Carnot's cycle (temperature) $T$ : temperature $\eta$ : thermal efficiency	(9) $\eta = (T_1 - T_2)/T_1$	$T_1, T_2$	$\eta$
<b>Materials Science</b>	Young's modulus $T$ : stress $E$ : Young's modulus $\epsilon$ : elongation per unit length	(10) $E = T/\epsilon$	$T, \epsilon$	$E$
	Shear stress $P$ : shear load $A$ : cross section	(11) $\tau = P/A$	$P, A$	$\tau$
	Shear stress $G$ : modulus of rigidity $\gamma$ : shear strain	(12) $\tau = G\gamma$	$G, \gamma$	$\tau$

### Operation:

PF3 [ENG] PF2 [MECH]  
PF1 [ME-FRM]

```

* MECHANICAL FORMULAS *
Module (M=D1/Z1=D2/Z2=P/π)
M=D1/Z1
[  ] [  ] [CENTER] [  ] [  ]

```

PF4 [ ]

```

* MECHANICAL FORMULAS *
Bernoulli's theorem
v2/2+P/ρ+gz=Constant
[  ] [  ] [CENTER] [  ] [  ]

```

If you press the [ ] key when a formula is displayed, the computer enables calculation using that formula:

[ ]

```

v2/2+P/ρ+gz=Constant
P1= ?

```

Pressing only the [ ] key at the data entry prompt will return to the previous display for the formula.

Use the following keys to recall a specific mechanical formula:

- PF1 [ ] : Recalls the first formula.
- PF2 [ ] : Recalls the formula which is three steps before the formula now displayed (i.e. the formula with current number minus 3).
- PF3 [CENTER] : Recalls the 6th (mid) formula.
- PF4 [ ] : Recalls the formula which is three steps beyond the formula now displayed (i.e. the formula with current number plus 3).
- PF5 [ ] : Recalls the 12th (last) formula.
- [ ] : Recalls the formula which just follows the formula now displayed.
- [ ] : Recalls the formula which just precedes the formula now displayed.

### Note:

- The [PF2], [PF4], [ ], and [ ] keys continuously recall formulas while they are pressed down.

### Reference:

- Details of Bernoulli's theorem:  
With Bernoulli's theorem, we can determine the velocity of fluid at an arbitrary point from the following formula if the pressures of the fluid at two points ( $P_1$ ,  $P_2$ ), its heights ( $z_1$ ,  $z_2$ ), mass density of the fluid ( $\rho$ ), acceleration of gravity ( $g$ ), and velocity of fluid at point  $P_1$  are given:

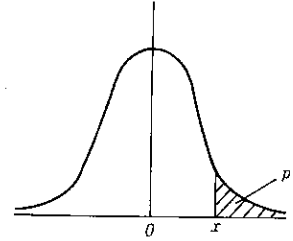
$$v_2 = \sqrt{\frac{2(P_1 - P_2)}{\rho} + v_1^2 + 2g(z_1 - z_2)} \quad \text{where } g = 9.80665 \text{ m/s}^2 \text{ on the ground.}$$

- Equation of continuity:  
We can determine the velocity of fluid at an arbitrary point from the following equation of continuity if the cross sectional areas ( $A_1$ ,  $A_2$ ) at two points, mass densities of fluid ( $\rho_1$ ,  $\rho_2$ ), and the velocity ( $v_1$ ) of fluid at another point are given:

$$v_2 = \frac{A_1 v_1 \rho_1}{A_2 \rho_2}$$

## \*411 NORMAL DISTRIBUTION

The NORMAL DISTRIBUTION program determines point  $x$  from a given probability  $p$  to the right of  $x$  or, in turn, the probability  $p$  to the right of  $x$  from a given point  $x$ . Both input data and result are given to four significant digits. The probability  $p$  must be given in the range of  $0 < p < 1$ .



**Operation:**

**PF4** [STAT] **PF1** [DISTR]  
**PF1** [NORMAL]

```

* NORMAL DISTRIBUTION *
N(p) ... 1           N(x) ... 2
    
```

(Menu)

**1** (N(p) is selected.)

```

p ( 0 < p < 1 ) = ?
    
```

0.05 **←**

```

p ( 0 < p < 1 ) = 0.05
x = 1.645
p ( 0 < p < 1 ) = ?
    
```

**←** **2** (N(x) is selected.)

```

x = ?
    
```

1.645 **←**

```

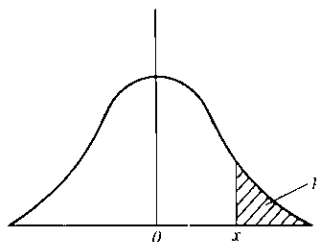
x = 1.645
p = 0.05
x = ?
    
```

**Note:**

- If you press only the **←** key at the prompt for  $x$  above, the display will return to the menu.

## \*412 t DISTRIBUTION

The t DISTRIBUTION program determines point  $x$  from a given degree of freedom  $n$  of the t distribution and a given probability  $p$  to the right of  $x$  or, in turn, the probability  $p$  to the right of  $x$  from a given degree of freedom  $n$  and a given point  $x$ . Both input data and result are given to four significant digits. The degree of freedom  $n$  must be an integer greater than 2; probability  $p$  must be given in the range of  $0 < p < 1$ .



### Operation:

**PF4** [STAT] **PF1** [DISTR]  
**PF2** [t]

```

* t DISTRIBUTION *
t(p) ... 1      t(x) ... 2
    
```

(Menu)

**1** (t(p) is selected.)

```

n ( 2 < n ) = ?
    
```

10 **←**

```

n ( 2 < n ) = 10
p ( 0 < p < 1 ) = ?
    
```

0.01 **←**

```

n ( 2 < n ) = 10
p ( 0 < p < 1 ) = 0.01
x = 2.764
n ( 2 < n ) = ?
    
```

**←** **2** **10** **←**  
(t(x) is selected.)

```

n = 10
x = ?
    
```

-2.764 **←**

```

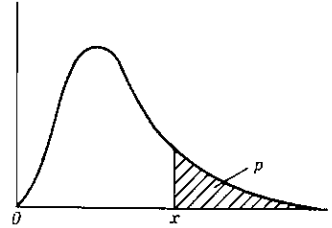
n = 10
x = -2.764
p = 0.99
n = ?
    
```

### Note:

- If you press only the **←** key at the prompt for  $n$  above, the display will return to the menu.

## \*413 CHI-SQUARE DISTRIBUTION

The CHI-SQUARE DISTRIBUTION program determines point  $x$  from a given degree of freedom  $n$  of  $\chi^2$  distribution and a given probability  $p$  to the right of  $x$  or, in turn, the probability  $p$  to the right of  $x$  from a given degree of freedom  $n$  and a given point  $x$ . Point  $x$  and the probability  $p$  to the right of  $x$  must have 3 and 4 significant digits, respectively. The degree of freedom  $n$  must be an integer greater than 2; probability  $p$  must be given in the range of  $0 < p \leq 1$ ; and  $x$  must be a non-negative number. The calculation may take some time depending on entry values.



**Operation:**

[STAT]    [DISTR]  
 [CHI]

\* CHI-SQUARE DISTRIBUTION \*

$\chi^2(p) \dots 1$                        $\chi^2(x) \dots 2$

(Menu)

( $\chi^2(p)$  is selected.)

$n = ?$

3

$n = 3$   
 $p (0 < p \leq 1) = ?$

0.95

$n = 3$   
 $p (0 < p \leq 1) = 0.95$   
 $x = 0.352$   
 $n = ?$

( $\chi^2(x)$  is selected.)

$n = 3$   
 $x = ?$

0.3518

$n = 3$   
 $x = 0.3518$   
 $p = 0.95$   
 $n = ?$

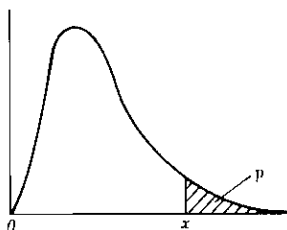
**Note:**

- If you press only the  key at the prompt for  $n$  above, the display will return to the menu.

## \*414 F DISTRIBUTION

The F DISTRIBUTION program lets you determine point  $x$  from given degrees of freedom,  $n_1$  and  $n_2$ , of F distribution and a given probability  $p$  to the right of  $x$  or, in turn, the probability  $p$  to the right of  $x$  from given degrees of freedom,  $n_1$  and  $n_2$ , and a given point  $x$ . Both input data and result are given to four significant digits. The degrees of freedom,  $n_1$  and  $n_2$ , must be positive integers; probability  $p$  must be given in the range of  $0 < p \leq 1$ ; and  $x$  must be a non-negative number.

The calculation may take some time depending on entered values.



### Operation:

[STAT]  [DISTR]  
 [F]

```

* F DISTRIBUTION *
F(p) ...1      F(x) ...2
    
```

(Menu)

(F(p) is selected.)

```

n1 = ?
    
```

5  10

```

n1 = 5
n2 = 10
p ( 0 < p ≤ 1 ) = ?
    
```

0.05

```

n2 = 10
p ( 0 < p ≤ 1 ) = 0.05
x = 3.326
n1 = ?
    
```

5  10   
(F(x) is selected.)

```

n1 = 5
n2 = 10
x = ?
    
```

3.326

```

n2 = 10
x = 3.326
p = 0.04999
n1 = ?
    
```

### Note:

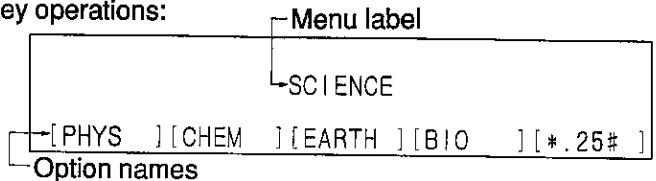
- If you press only the  key at the prompt for  $n_1$  above, the display will return to the menu.

## \*51 PF KEY LABEL EDITOR

The PF KEY LABEL EDITOR program changes the menu labels to be displayed when the **[PF1]** – **[PF5]** keys are pressed in the Engineer Software mode, or to designate a new menu option (file names) to an undefined PF key template (e.g. **[\*.25#]** shown below) and register it in a RAM disk.

For example, try the following key operations:

**[PF2]** **[SCI]**



If you wish to change an already defined option name, you have to create a file with the same file extension within the RAM disk (see page 115).

The following describes how to define a new menu label and/or option name. Before attempting the following operations, initialize RAM disk E by entering: **INIT "E: 10K"** **[←]** in the BASIC mode. Also, in the following example, suppose you have then saved two sports and hobby programs using: **SAVE "E: GOLF. 611"** and **SAVE "E: GAME. 62#"**.

### Designating a New Menu Label or Option Name to PF Keys

Operation:

**[PF5]** **[EDIT]** **[PF1]** **[EDITOR]**

```

*   PF KEY LABEL EDITOR   *
APPEND           ...1
MODIFY           ...2
    
```

(Menu)

Let's assign option name "HOBBY" to key template ".6##", and "SPORTS/GAME" to the menu label.

**[1]** (APPEND is selected.)

```

FILE NAME= ?
(EXAMPLE) E:KEY LABEL.6##  _
    
```

E: HOBBY. 6## **[←]**

```

MENU LABEL= ?
LINE1 =  _
    
```

(To enter a colon (: ) or sharp (#), press **[2nd F]** before pressing the corresponding symbol key.)



Next, enter up to three lines of menu label. You can enter up to 30 characters for each line.

SPORTS/GAME

```
MENU LABEL= ?
LINE1 = SPORTS/GAME
LINE2 = _
```

```
EDIT
[EDITOR][*.52# ][*.53# ][*.54# ][*.55# ]
```

(Press the  key only at LINE 2 and LINE 3 prompts since you have only a single line of menu label.)

Now verify what you've just registered:

**BREAK**

```
ENGINEER SOFTWARE
[HOBBY ][*.7## ][*.8## ][*.9## ][*.0## ]
```

You see that "HOBBY" is designated to the  key template.

Next, designate option name "SPORTS" to template ".61#."

**PF5** [EDIT]  
**PF1** [EDITOR]

```
* PF KEY LABEL EDITOR *
APPEND          ...1
MODIFY          ...2
```

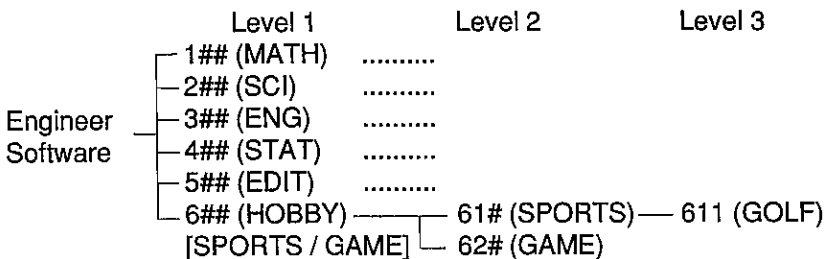
1 E:SPORTS.61#

```
MENU LABEL= ?
LINE1 = _
```

Since, this time, nothing needs to be designated to the menu label, press the  key for each prompt:

```
EDIT
[EDITOR][*.52# ][*.53# ][*.54# ][*.55# ]
```

The Engineer Software has now acquired the following hierarchical structure:



The title given in brackets [ ] indicates a menu label.

**Note:**

- While factorization and other calculation programs are registered at Level 3, they may be re-designated to Level 2 and then executed with key operations for Level 2, in much the same way as the EDITOR or GAME program. They may also be re-designated to Level 1 and then executed with key operations for Level 1.

**Changing a Registered Menu Label**

**Operation:**

Press the  key on the PF KEY LABEL EDITOR menu.

```

FILE NAME=?
(EXAMPLE) G:SCI.2## _

```

Now enter the name of the RAM file that contains the menu label you wish to change. Here let's append "Let's play" to the second line of the menu label "SPORTS/GAME" registered in RAM disk E:HOBBY.6##.

E:HOBBY.6##

```

MENU LABEL= SPORTS/GAME

LINE1 = _

```

(Press only  since no change is made to the first line.)

L  et   s  
 play

```

LINE1 =
LINE2 = Let's play
LINE3 = _

```

```

WHICH MEMORY ?
MAIN (E:) ...1      CARD (F:) ... 2

```

(Press only the  key since no change is made to the third line either.)

To rewrite the changed menu label to RAM disk E, press the  key:

\*

Now check to see if the menu label has been successfully changed:

[HOBBY]

```

SPORTS/GAME
Let's play

[SPORTS][GAME ][*.63# ][*.64# ][*.65# ]

```

\* If you wish to rewrite the menu label to RAM disk F by pressing the  key, the RAM card must be installed in the computer.

## Notes:

- If a program is saved with file extension "611", it is not executable if "6##" and "61#" are not defined. Register appropriate characters using the APPEND registration option.
- The contents of the Engineer Software are not rewritable as they reside on ROM (device name G). If you have changed the menu label for device G (specified with G: SCI. 2##), it is registered to the RAM disk (see the "WHICH MEMORY?" prompt).
- If the file name you input for registering a new menu label is not correct, the registered menu label is not valid.
- If the file name you input when changing a menu label is not correct, the computer will return to the MENU LABEL prompt with "FILE ERROR" message appearing on the display.
- If the file extension for the file name you enter is not adequate, the computer will prompt for re-entry.
- To delete a menu label from the RAM disk, use KILL, a BASIC command.


## How To Create an Engineer Software Program

Every Engineer Software program has a file extension of 000 to 999. User-created programs with file extension 000 to 999 can be included in the Engineer Software programs. The Engineer Software is supplied in ROM. If a program with the same file extension exists in RAM disk E, the program in RAM disk E will be executed before the one residing in ROM.

The order of priority is: RAM disk F > RAM disk E > ROM

By use of the order of priority, you can change the contents of an Engineer Software program or designate a program to a file with an undefined file extension.

First create a program with the PRO mode of BASIC, then do the following operations. These operations do not replace the contents of the ROM program. The ROM program can be restored if the replacement program written in a RAM disk is deleted.

**INIT "E: 10K"  (not needed if already done)**

**SAVE "E: SHARP.1##" **

**SAVE "E: PC-E.611" **

Now press the **MENU** **PF5** keys to enter the Engineer Software mode. The file name "MATH" will then be replaced with "SHARP". This means that the system will subsequently run the program "SHARP" (with file extension "1##") in RAM disk E instead of the program "MATH" (with the same file extension "1##") in ROM. To execute the program "SHARP", press the **PF1** key.

Now let's look for the program with file extension "611". You will not be able to find "PC-E" because "61#", the level just preceding "611", is not defined. Thus, you cannot execute a program if the level preceding that program is not defined. Note this point whenever writing a program.

**For the procedure to designate a program to "61#", see the description under "PF Key Label Editor".**

## Hints on Program Creation

The Engineer Software uses special instructions and functions that are not included in the BASIC instruction set:

- **A\$ = INKEY\$&1**


This statement returns the same code as the INKEY\$ whenever a key is pressed. However, if no key is pressed for approximately 11 minutes, the auto-power off feature is activated.





- **MERGE**

The MERGE instruction can be used in programs as well, provided that the program to be merged is written in intermediate code format and its line numbers are greater than those of any internal program.

- **P6 = EVAL "-X/Y + SIN X"**

This statement lets the system compute the expression enclosed in the double quotes using the values assigned to variables X and Y, and assigns the result to variable P6. Any Engineer Software program using input formula uses this statement format.

- The Engineer Software is supplied in ROM (device name G). The names of the program files stored in ROM can be listed by typing: **FI."G:"** , as with the case of RAM files. Also programs with a 3-digit file extension (excluding "#") can be read out of the ROM and then stored into a RAM disk (see the FILES command).
- When executing a statement such as **CHR\$&H83**, symbols different from the character code chart, such as Greek letters, may be displayed. Perform the following procedure:

1. Enter: **FI."G:"** 
2. Load the file named "GREEK".
3. Enter: **RUN** 
4. Press the **BREAK** key to interrupt the execution.
5. Enter: **CHR\$&H83**   
"T" will be displayed.
6. Press the **CTRL** + **+/-** keys.  
Enter: **CHR\$&H83**   
"â" will be displayed.  
Pressing the **CTRL** + **+/-** keys will display the previous character.

- If an Engineer Software program is manually loaded (LOAD) then run (RUN), the MERGE instruction, if any, in the program merges a common subroutine. If the execution of the program is temporarily suspended with the **BREAK** key and then resumed, the system will attempt to merge again the common subroutine already merged, which results in an error. In such a case, delete the MERGE instruction from the program before executing it.

### Reference:

The Engineer Software uses variables of two characters or more in length which begin with letter P, Q, or R. For calculations using input formulas, it uses variables X and Y. For data plotting, it also uses array variable MD.

## 8. AER MODE

The Algebraic Expression Reserve (AER) mode is convenient for repetitive calculations. Calculations using algebraic expressions and constants can be programmed in the AER mode to perform calculations on numbers or variables entered in the RUN Mode. Or, the algebraic expressions can be recalled from a BASIC program. (Refer to the AER command in the BASIC COMMAND DICTIONARY.) The stored expressions cannot be transferred to peripheral devices.

### Programming and Recalling Expressions

#### Programming Expressions

##### (1) Mode setting

Display the AER key label on the main menu (by pressing the **MENU** and **◀** keys) and press the **PF1** key or the **SHIFT** + **AER** keys.

```
01:TITLE? XXXXX bytes free          * AER *
```

- 01: Indicates the expression number (1 to 99). If 99 expressions are already stored, the number "99", the title and expression are displayed.
- TITLE?: Indicates the title of the expression. If the expression is not registered, "TITLE?" will be displayed.
- XXXXX bytes free: Indicates the number of bytes left in the storage area. Program and data areas are shared so that the number of bytes left in the storage area will decrease as additional programs are written. If the number exceeds 65535 bytes, "65535" will be displayed.

##### (2) Title entry

Up to 20 characters can be entered as the expression's title.

- Press the **←** key only to enter no title.

##### (3) Entry format



Expressions can be entered in the following formats:

① (parameter + expression):

F (parameter, parameter, ...) = expression

② (expression) : expression only

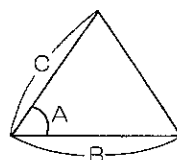
- Single precision numerical variables can be used. Double precision numerical variables, string variables, or array variables cannot be used. Also, double-precision calculations cannot be performed. (Refer to page 173.)
- Function keys such as **sin**, **cos** and **2nd F** **e<sup>x</sup>** can be used.
- A comma ( , ) is required to separate the variables.

- It is convenient to store constants in format 2 to be used in direct input operations with the last answer function.
  - The expression is stored by pressing the  key at the end.
  - When the memory is exceeded, "AER memory over" is displayed and the computer enters the AER edit mode.
  - Up to 80 characters or 80 bytes of an expression can be stored (8 bytes per value, 2 bytes per function and 1 byte per other character including the  key). If this is exceeded, "AER convert error" will be displayed, and the expression must be simplified.
- (4) After expression entry, the expression number will be incremented by one, and the computer displays the prompt for the next entry.

**Example:**

Enter the formula for the area of triangles.

$$\text{Area } S = \frac{B \cdot C \cdot \sin A}{2}$$



 +  AREA 

```
01:AREA                                     *AER*
:?
```

F (A, B, C) =

```
01:AREA                                     *AER*
:F(A,B,C)=_
```

B\*C\*SIN A/2

```
01:AREA                                     *AER*
:F(A,B,C)=B*C*SIN A/2_
```



```
02:TITLE? xxxxx bytes free                 *AER*
```

Entry of the formula for the area of triangles is complete, and the computer displays the prompt for the next entry.

**Example:**

Enter Planck's constant.

PLANCK 

```
02:PLANCK                                    *AER*
:?
```

6.626076E-34

02:PLANCK :6.626076E-34_	*AER*
-----------------------------	-------



03:TITLE? XXXXX bytes free	*AER*
----------------------------	-------

### Recalling Expressions

- (1) Pressing the **TITLE** key displays the currently stored titles and expressions sequentially.
- (2) Pressing the **SHIFT** + **TITLE** keys displays the currently stored titles and expressions in reverse order.
  - During expression entry, pressing the **TITLE** key cancels the current entry.

### Example:

Recall the currently stored expressions.

**SHIFT** + **AER**

03:TITLE? XXXXX bytes free	*AER*
----------------------------	-------

**TITLE**

02:PLANCK :6.626076E-34	*AER*
----------------------------	-------

**TITLE**

01:AREA :F(A,B,C)=B*C*SIN A/2	*AER*
----------------------------------	-------

## Correcting and Deleting Expressions

### Correcting Expressions

- (1) To correct the entered expression before pressing the key, position the cursor and correct with keys such as **DEL**, **INS**, **BS**.
- (2) To correct an expression already stored, set the operation mode to AER, recall the expression to be corrected with the **TITLE** key. Correct the entry as follows.
  - ① Title correction  
Press the key. The cursor moves to the first character of the expression title. Position the cursor and correct the expression title. After the correction is complete, press the key. If the title is not to be corrected, just press the key.
  - ② Expression correction  
Next, the expression is displayed with the cursor over the first character of the expression. Position the cursor and correct the expression. After the correction is complete, press the key. If the expression is not to be corrected, just press the key.

**Example:**

Correct the No.2 expression title "PLANCK" to "PLANCK CONSTANT".

Recall the title "PLANCK" by pressing the **TITLE** key in the AER mode.

```
02:PLANCK
:6.626076E-34 *AER*
```

**▶** ... **▶**\* (Press 7 times.)  
\* Or press the **SHIFT** + **▶** key to move the cursor to the column after the last character of the title.

```
02:PLANCK_ *AER*
```

**SPACE** CONSTANT **←**

```
02:PLANCK CONSTANT
:6.626076E-34 *AER*
```

**←**

```
03:TITLE? xxxxx bytes free *AER*
```

Now the correction is complete and the computer displays the next expression prompt.

**Deleting expressions**

Carry out the following procedure to delete an expression already stored.

- (1) Recall the expression to be deleted with the **TITLE** key.
- (2) Press the **SHIFT** + **C•CE** keys.  
"02:AER data clear OK? (Y/N)" will be displayed to confirm the deletion. "02" is the expression title number of the expression to be deleted.
- (3) Press the **Y** key to delete the expression or press the **N** key not to delete it. If the **N** key is pressed, the computer returns to the display from which the expression was recalled.  
(For now, press the **N** key. The expression stored in "02" will be used in future examples.)

**Note:**

When an expression is deleted, the deleted storage area is filled with the expression following the deleted expression, so some expression title numbers may change. In BASIC programs, it may be necessary to change the specified number in the AER command statement.



## Executing an Expression

Perform the execution in the RUN mode for the expression stored in the AER mode.

- (1) Press the **BASIC** key to set the RUN mode.
- (2) Press the **TITLE** key to recall expressions in forward order. Press the **SHIFT** + **TITLE** key to recall in reverse order.
- (3) When the expression to be executed is recalled, press the **←** key.
- (4) If the desired expression is in the format (parameter + expression), the computer displays the entry prompt for the parameter(s) specified in F ( ) sequentially. Press the **←** key after each numerical data entry. If the desired expression is in the format (expression), the calculated result or constant will be displayed.
  - If the **←** key is pressed without numerical data entry, the previously entered data value will be used for the parameter or 0 if no value was entered previously.
- (5) After parameter entry is completed, the calculated result will be displayed.

### Example:

Perform calculations using the stored expressions "PLANCK CONSTANT" and "AREA".

Set the RUN mode.

DEGREE **←**

<b>TITLE</b>	02:PLANCK CONSTANT :6.626076E-34
<b>←</b>	6.626076E-34
<b>TITLE</b> <b>TITLE</b>	01:AREA :F(A,B,C)=B*C*SIN A/2
<b>←</b>	A=?
Enter the value for angle A. 30	A=30_
<b>←</b>	A=30 B=?

Enter the value for side B.

100

```
A=30
B=100
C=?
```

Enter the value for side C.

20

```
A=30
B=100
C=20
500
```

The area for the triangle is 500.

Pressing the  key will execute the same expression again.

## Searching Expression Titles

In the RUN mode, expressions can be recalled by entering the whole or first few characters of the title and pressing the  key. After the recall, pressing the  key again will search for the same title starting from the one following the currently displayed expression.

- When the title cannot be found, "Title not found" is displayed.

### Example:

Recall the expression with the title "AREA" by searching for the characters "AR"

AR

```
01: AREA
: F(A,B,C)=B*C*SIN A/2
```

## Error Messages

### (1) AER format error

① The expression was entered in an incorrect format (parameter + expression).

② ")"=" is missing after "F("

- Press the  key to enter a new title and expression.
- Press the  or  key to display the expression, position the cursor, and correct it.

### (2) AER memory over

Expressions cannot be entered since memory is full.

- Press the  key to display the entry prompt.
- Press the  or  key to display the expression title.
- Delete unnecessary expressions.

### (3) Title not found

The title searched for with the specified characters cannot be found.

### (4) AER convert error

The size of the expression exceeds the capacity which can be stored (80 characters or 80 bytes)

- Press the  key to display the expression.
- Reduce the length of the expression.

# 9. STAT MODE

The computer can perform statistical and regression calculations on one or two variables. With statistical calculations you can obtain mean values, standard deviations, and other statistic quantities from sample data. Regression calculation determines the coefficients of linear regression formulas or estimated values from sample data. Sample data for statistical and regression calculations are stored to the single-precision array variable MD.

## Calculating Statistics

### Statistical Calculations

Single-variable statistical calculation and two-variable statistical calculation (linear regression) determines the following statistics:

(Single-variable statistical calculation determines the statistics ① to ⑥ only.)

- ① [n] Sample size of  $x$
- ② [ $\bar{x}$ ] Sample mean  $x$
- ③ [ $\Sigma x$ ] Sum of samples  $x$
- ④ [ $\Sigma x^2$ ] Sum of squares of samples  $x$
- ⑤ [s $x$ ] Sample standard deviation with population parameter taken to be  $n-1$ .  
 $s_x = \sqrt{(\Sigma x^2 - n\bar{x}^2)/(n-1)}$ 
  - This equation is used to estimate the standard deviation of a population from sample data ( $x$ ) extracted from that population.
- ⑥ [ $\sigma_x$ ] Population standard deviation with population parameter taken to be  $n$ .  
 $\sigma_x = \sqrt{(\Sigma x^2 - n\bar{x}^2)/n}$ 
  - This equation lets you assume the entire population as sample data ( $x$ ) or determine the standard deviation of the sample data which is taken to be a population.
- ⑦ [ $\bar{y}$ ] Sample mean  $y$
- ⑧ [ $\Sigma y$ ] Sum of sample  $y$
- ⑨ [ $\Sigma y^2$ ] Sum of squares of samples  $y$
- ⑩ [s $y$ ] Sample standard deviation with population parameter taken to be  $n-1$ .  
 $s_y = \sqrt{(\Sigma y^2 - n\bar{y}^2)/(n-1)}$
- ⑪ [ $\sigma_y$ ] Population standard deviation of samples ( $y$ ) with population parameter taken to be  $n$ .  
 $\sigma_y = \sqrt{(\Sigma y^2 - n\bar{y}^2)/n}$
- ⑫ [ $\Sigma xy$ ] Sum of products of samples  $x$  and  $y$

For exponential, logarithmic, power and inverse regression calculations, transform the respective regression calculation formula in the following table into a linear regression formula. Then each statistic can be determined using the transformed values for X and Y.

For two-variable statistical calculations, select the linear regression calculation.

Type	X	Y	Transformed formula
Linear	x	y	No transformation
Exponential	x	ln y	$Y = \ln a + bx$
Logarithmic	ln x	y	$y = a + bX$
Power	ln x	ln y	$Y = \ln a + bX$
Inverse	1/x	y	$y = a + bX$

### Example of Single-Variable Statistical Calculations

When there is more than one set of independent, single-variable data, first enter all of the data items, then calculate the statistics for each set of data.

#### Example:

Data sets A and B are composed of the following 20 data. From these data, determine the mean, standard deviations, and other statistics.

No.	data A	data B	No.	data A	data B
1	30	20	6	80	70
2	80	70	7	100	90
3	50	60	8	60	50
4	80	70	9	40	60
5	80	70	10	90	80

**MENU** (Main menu)

```

MAIN MENU
[BASIC] [CAL] [MATRIX] [STAT] [ENG]

```

Select the STAT mode.

**PF4** [STAT]

```

STATISTICAL VARIABLE
STAT:D(000,0...)
[CANCEL]

```

Enter the number of sets of data. Since there are two sets of data, enter "2".

2

```

001:d1=          ?
[INS] [DEL] [FREQ] [QUIT]

```

Enter data A and then data B.

30

```

001:d2=          ?
[INS] [DEL] [FREQ] [QUIT]

```

20

002:d1=	?
[ INS ] [ DEL ] [ FREQ ]	[ QUIT ]

80  70  50   
60

004:d1=	?
[ INS ] [ DEL ] [ FREQ ]	[ QUIT ]

- When you come across a repeated value, you can enter them all using the  [PF3] [FREQ] key. For example, you now encounter three consecutive "80's" in set A and three consecutive "70's" B. You can enter them by typing:

80  70  3

FREQUENCY ( 3 )
[CANCEL]

- If the incorrect value is entered, press the  [C-CE] key to clear. To cancel the [FREQ] operation after the [FREQ] key is pressed, press the  [PF5] [CANCEL] key. In this case, however, press the  key to continue data entry.

007:d1=	?
[ INS ] [ DEL ] [ FREQ ]	[ QUIT ]

Enter the remaining data in a similar manner:

100  90  60   
50  40  60   
90  80

011:d1=	?
[ INS ] [ DEL ] [ FREQ ]	[ QUIT ]

You have now completed all data entry. First determine statistics for data set A.

[PF5] [QUIT]

D (010, 2)
[ NEW ] [ DATA ] [ CALC ]

Next, press the  [PF3] [CALC] key to select the calculation.

[PF3]

STATISTIC/REGRESSION ANALYSIS
[ STAT ] [ a+bx ]

Press the **PF1** [STAT] key to determine the statistics for data set A.

**PF1**

```

STAT
                                x←d1_
                                [CANCEL]
    
```

"x ← d1\_" asks if you wish to set the data in the first column as  $x$ . Press the **←** key since data set A is in the first column:

**←**

```

                                0.
[ n ] [ x̄ ] [ Σx ] [ Σx² ] [ s x ]
    
```

Now, display the mean and sample standard deviation.

**PF2** [x̄]

```

                                69.
◄
[ n ] [ x̄ ] [ Σx ] [ Σx² ] [ s x ]
    
```

Press the **PF1** or **PF3** key to display the sample size  $n$  or sum of samples  $\Sigma x$ . The population standard deviation  $\sigma x$  is not displayed here. Press the **↕** key to display the key label [ $\sigma x$ ].

**↕**

```

                                69.
[ σ x ]
    
```

**PF1** [ $\sigma x$ ]

```

                                21.65640783
[ σ x ]
    
```

The symbol "◄" below **STAT** on the display indicates that information yet to be displayed exists and the **↕** key can be used.

To perform statistical computation on data set B, press the **BREAK** key to initialize the display and press the **PF3** [CALC] key.

**BREAK** **PF3**

```

STATISTIC/REGRESSION ANALYSIS
[ STAT ] [ a+bx ]
    
```

**PF1** [STAT]

```

STAT
                                x←d1_
                                [CANCEL]
    
```

Enter "2" since data set B is in the second column:

2 **←**

```

                                0.
[ n ] [ x̄ ] [ Σx ] [ Σx² ] [ s x ]
    
```

You can obtain statistics for data set B with the same keys used for data set A. To make calculations more readable, you can use the Notation function described on page 42.

To specify three decimal places for a calculation result, press the **FSE** **2nd F** **TAB** **3** keys before pressing the **PF1** [ $\sigma x$ ] key:

**PF1** [ $\sigma x$ ]

	18.000
[ $\sigma x$ ]	

## Linear Regression Calculations

For two-variable statistical calculations, you can determine the following data using linear regression calculations ( $y = a + bx$ ):

① Coefficient  $a = \bar{y} - b \bar{x}$

$$b = \frac{S_{xy}}{S_{xx}}$$

② Correlational coefficients

$$r = \frac{S_{xy}}{\sqrt{S_{xx} \cdot S_{yy}}}$$

where

$$S_{xx} = \sum x^2 - \frac{(\sum x)^2}{n}$$

$$S_{yy} = \sum y^2 - \frac{(\sum y)^2}{n}$$

$$S_{xy} = \sum xy - \frac{\sum x \cdot \sum y}{n}$$

③ Estimated value (x value estimated from y value)

$$x' = \frac{y - a}{b}$$

(y value estimated from x value)

$$y' = a + bx$$

④ Residual

$$\frac{\sum (y_i - (a + bx_i))^2}{n}$$

### Statistical and Regression Calculation Procedures

For statistical and regression calculations on the computer, you can enter data in table form, as shown below:

d1 column d2 column d3 column d4 column .....

1st row					
2nd row					
3rd row					
4th row					
⋮					

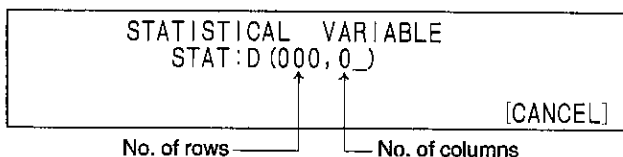
The number of columns can be specified from 1 to 9. The number of rows need not be specified; it is automatically incremented as you enter data. A section specified by a specific row and column is called a cell.

Statistical and regressional calculations are performed in the following sequence:

### ① Setting up

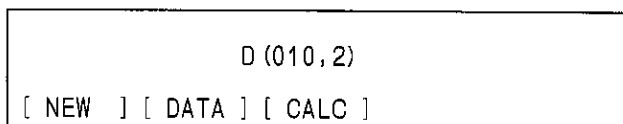
Press the **[PF4]** [STAT] key on the main menu.

**[MENU]** **[PF4]**



Now enter the number of columns from 1 to 9.

If a data table already exists in memory, the following kind of display will be shown.



If the **[PF1]** [NEW] key is pressed, the number of columns is entered and the **[←]** key is pressed, the current data will be cleared and the data table is ready to receive data.

### ② Entering data

After you specify the number of columns, enter the data.

Use the following keys to enter data or correct the entered data. Press the **[PF5]** [QUIT] key to complete data entry.

Key	Function
<b>[←]</b>	Stores the entered data into the cell currently on the display, and then displays the cell on the right. When the rightmost cell is displayed, this key displays the leftmost cell in the next row.
<b>[C•CE]</b>	Clears the data on the display. Returns the "?" prompt for new data entry.
<b>[C•CE]</b> <b>[←]</b>	Clears data from the displayed cell. The cell becomes unregistered.
<b>[▶]</b>	Displays the data of the cell on the right. If the rightmost cell is now on the display, this key displays the leftmost cell of the next row.
<b>[◀]</b>	Displays the data of the cell on the left. If the leftmost cell is now on the display, this key displays the rightmost cell of the previous row.
<b>[↑]</b>	Displays the data of the row above, in the same column.
<b>[↓]</b>	Displays the data of the row below, in the same column.
<b>[2nd F]</b> <b>[▶]</b>	Displays the rightmost cell of the current row.
<b>[2nd F]</b> <b>[◀]</b>	Displays the leftmost cell of the current row.
<b>[2nd F]</b> <b>[↑]</b>	Displays the top cell of the current column.
<b>[2nd F]</b> <b>[↓]</b>	Displays the bottom cell of the current column.
<b>[NEW]</b>	Enters the new data entry.
<b>[DATA]</b>	Re-enters the data entry display. Data correction, addition and deletion can be performed.
<b>[INS]</b>	Inserts the specified number of rows before the current row. Pressing the [INS] key will display the prompt for the number of rows to be inserted. Enter the desired number. If the capacity is exceeded, an error will occur and no further insertion will be allowed.



Key	Function
[DEL]	Deletes the specified number of rows from the displayed data. Pressing the [DEL] key will display the prompt for the number of rows to be deleted beginning with the current row. Enter the desired number. If the number exceeds the number of the rows left, all the rows from the displayed data onwards will be deleted. (An error will not occur.)
[FREQ]	If the same value is repeated consecutively in rows, you can enter the data by entering: (data for the rightmost cell) [PF3] [FREQ] number of rows with the same data [←].

- If you have made a typing error, recall the incorrect data with the [↑], [↓], [←], or [→] key, type the correct data, then press the [←] key.
- The maximum allowable size of the data table is 9 columns by 256 rows.
- The allowable range of entry data is  $0 \leq | \text{data} | < 1 \times 10^{99}$ . However, since the sum of squares is available as an operation, an overflow error may occur if data greater than  $10^{50}$  is included in entry data.
- Functions or formulas may also be entered as entry data such as:  $20 \times 3 + 12 =$  [←] and  $30 \sin$  [←]. However, the [+HEX] or [+DEC] key operation cannot be entered.  
If you wish to change the value of a term in the formula before pressing [←], press the [C•CE] key to display "?", then enter the replacement data.
- If there are some test data that cannot be entered because of measurement errors or other reasons, you may wish to exclude this data from the calculations. With this calculator, data items can be excluded from a calculation if you identify them as unregistered data. The row of data into which the unregistered data is placed will be ignored for subsequent calculations.
- To identify data as unregistered data items, press the [C•CE] [←] keys.
- All data will be cleared if the number of rows is changed.

③ Select the statistics or linear regression calculation.

④ Specify the desired statistic or result of linear regression calculation.

### Examples of Regression Calculation

Linear regression calculation

The following table lists the dates (April) on which migratory birds fly through a certain district, versus the average temperatures in March of the same district. From this table determine the coefficients,  $a$  and  $b$ , of the linear regression line,  $y = a + bx$ , and correlation coefficient  $r$ , and estimate the date of migration when the mean temperature in March is  $9.1^\circ\text{C}$ .

Year	1	2	3	4	5	6	7	8
Mean temp. ( $^\circ\text{C}$ )	6.2	7.0	6.8	8.7	7.9	6.5	6.1	8.2
Date of migration (y day)	13	9	11	5	7	12	15	7

**Operation:**

[MENU] [PF4] [STAT]

STATISTICAL VARIABLE  
STAT:D(000,0\_)  
[CANCEL]

Enter "2" for the number of columns.

2 [←]

001:d1= ?  
[INS] [DEL] [FREQ] [QUIT]

6.2 [←] 13 [←] 7 [←] 9 [←] 6.8 [←] 11 [←] 8.7 [←] 5 [←]  
7.9 [←] 7 [←] 6.5 [←] 12 [←] 6.1 [←] 15 [←] 8.2 [←] 7 [←]

009:d1= ?  
[INS] [DEL] [FREQ] [QUIT]

[PF5] [QUIT]

D(008,2)  
[NEW] [DATA] [CALC]

Next press the [PF3] [CALC] key to specify the STAT calculation submodule.

[PF3]

STATISTIC/REGRESSION ANALYSIS  
[STAT] [a+bx]

In this example, perform the linear regression calculation.

[PF2] [a + bx]

y=a+bx  
x←d1\_  
y←d2  
[CANCEL]

Since  $x$  and  $y$  are placed on columns d1 and d2, respectively, just press the [←] key twice.

[←] [←]

0.  
[r] [a] [b] [Se] [x']

Determine coefficients  $a$  and  $b$ , and correlation coefficient  $r$ .

[PF2] [a]

34.44951017  
[r] [a] [b] [Se] [x']

PF3 [b]

-3.425018839

[ r ] [ a ] [ b ] [ Se ] [ x' ]

PF1 [r]

-0.969106837

[ r ] [ a ] [ b ] [ Se ] [ x' ]

Next, estimate the value of  $y$  (date of migration) for  $x = 9.1$  (mean temperature). Press the  $\blacktriangleleft$  key to display the key label [y'] and press the desired PF key.

$\blacktriangleleft$  9.1 PF1 [y']

y = 3.281838734

[ y' ] [ n ] [ x ] [  $\Sigma x$  ] [  $\Sigma x^2$  ]

The following statistics can be determined:

residual ( $Se$ ), estimated value of  $x$  from value of  $y$  ( $x'$ ), sample size ( $n$ ), sample mean of  $x$  ( $\bar{x}$ ), sum of  $x$  ( $\Sigma x$ ), sum of squares of  $x$  ( $\Sigma x^2$ ), standard deviations of  $x$  ( $s_x$ ,  $\sigma_x$ ), sample mean of  $y$  ( $\bar{y}$ ), sum of  $y$  ( $\Sigma y$ ), sum of squares of  $y$  ( $\Sigma y^2$ ), standard deviations of  $y$  ( $s_y$ ,  $\sigma_y$ ), sum of products of  $x$  and  $y$  ( $\Sigma xy$ )

### Regression Calculation Using Data Table

The following table shows the results of an experiment attempted twice under different conditions. Perform regression calculation on this data.

No.	1st trial		2nd trial	
	$d_1(x)$	$d_2(y)$	$d_3(x)$	$d_4(y)$
1	1.0	15.5	1.5	24.1
2	2.5	28.0	3.0	38.0
3	5.0	43.2	7.0	69.5
4	10.0	90.0	8.0	71.3
5	20.0	158.0	10.0	90.7

### Operation:

MENU PF4 [STAT]

STATISTICAL VARIABLE  
STAT:D(000,0\_)

[CANCEL]

If the data is already registered, press the PF1 [NEW] key. The above display will appear.

Enter "4" for the number of columns.

4  $\blacktriangleleft$

001:d1= ?

[ INS ] [ DEL ] [ FREQ ] [ QUIT ]

1 [←] 15.5 [←] 1.5 [←] 24.1 [←]

2.5 [←] ...

... 90.7 [←]

006:d1=		?
[INS]	[DEL]	[QUIT]

[PF5] [QUIT]

D (005, 4)		
[NEW]	[DATA]	[CALC]

Next, press the [PF3] [CALC] key and perform the regression calculation on the data of the 1st trial.

[PF3] [PF2] [ $a + bx$ ]

$y = a + bx$	x←d1_
	y←d2_
	[CANCEL]

Press the [←] key twice since data  $x$  and  $y$  of the 1st trial are on columns d1 and d2, respectively.

[←] [←]

	0.			
[r]	[a]	[b]	[Se]	[x']

Determine coefficients  $a$  and  $b$ , and residual  $Se$ :

[PF2] [ $a$ ]

	8.673727735			
[r]	[a]	[b]	[Se]	[x']

[PF3] [ $b$ ]

	7.567048346			
[r]	[a]	[b]	[Se]	[x']

Determine the residual  $Se$  and store it in memory A:

[PF4] [ $Se$ ]  
[STO] [A]

	47.71196565			
[r]	[a]	[b]	[Se]	[x']

Perform the linear regression calculations on the data of the 2nd trial.

[BREAK] [PF3] [CALC]  
[PF2] [ $a + bx$ ]

$y = a + bx$	x←d1_
	y←d2_
	[CANCEL]

Specify columns d3 and d4 for data  $x$  and  $y$ , respectively:

3  4

$y = a + bx$		$x \leftarrow d3$
		$y \leftarrow d4$
[CANCEL]		

		0.	
[ r ]	[ a ]	[ b ]	[ Se ] [ x' ]

Determine coefficients  $a$  and  $b$ , and residual  $Se$ :

[a]

		13.84591633	
[ r ]	[ a ]	[ b ]	[ Se ] [ x' ]

[b]

		7.605776892	
[ r ]	[ a ]	[ b ]	[ Se ] [ x' ]

[Se]

		21.0863247	
[ r ]	[ a ]	[ b ]	[ Se ] [ x' ]

Recall the residual of the 1st trial and compare it with that of the 2nd trial:

		47.71196565	
[ r ]	[ a ]	[ b ]	[ Se ] [ x' ]

The residual is the sum of squares of the differences between the  $y$  values and the estimated values  $y'$  and used to determine the calculation accuracy.

**Note:**

Fixed variables J to Z are used for normal calculation. To store the result, use the  key and specify the fixed variables A to I, or  to store in memory.

**Example:**

Exponential regression calculation

From the data below, determine coefficients  $a$  and  $b$  of the equation  $y = a \cdot e^{bx}$ , and estimate the value of  $y$  when  $x = 12$ .

No.	1	2	3	4	5	6
$x$	2.0	7.0	9.2	4.3	5.1	8.0
$y$	0.6	4.2	8.3	1.2	2.7	5.1

- Calculation procedure  
 Take the natural logarithm of both sides of the equation,  $y = a \cdot e^{bx}$ :  
 $\ln y = \ln a + bx$  ..... (1)  
 Substituting Y for  $\ln y$  and A for  $\ln a$  gives:  
 $Y = A + bx$  ..... (2)  
 Use this equation for calculation.  
 When entering Y, use the natural logarithm of y (since  $Y = \ln y$ ).  
 To determine a, determine A and then calculate  $e^x$  since  $A = \ln a$  ( $a = e^A$ ).

**Operation:**

**MENU** **PF4** [STAT]

```

STATISTICAL VARIABLE
STAT:D (000,0_)
[CANCEL]
  
```

If the data is already registered, press the **PF1** [NEW] key. The above display will appear.

Enter "2" for the number of columns.

2 **←**

```

001:d1= ?
[INS] [DEL] [FREQ] [QUIT]
  
```

2 **←** 0.6 **ln** **←** 7 **←** 4.2 **ln** **←** 9.2 **←**  
 8.3 **ln** **←** 4.3 **←** 1.2 **ln** **←** 5.1 **←** 2.7 **ln** **←**  
 8 **←** 5.1 **ln** **←**

```

007:d1= ?
[INS] [DEL] [FREQ] [QUIT]
  
```

**PF5** [QUIT]

```

D (006,2)
[NEW] [DATA] [CALC]
  
```

Next, press the **PF3** [CALC] key to specify the calculation submodule.

**PF3**

```

STATISTIC/REGRESSION ANALYSIS
[STAT] [a+bx]
  
```

**PF2** [ $a + bx$ ]

```

y=a+bx
x←d1_
y←d2_
[CANCEL]
  
```

Press the  $\leftarrow$  key twice since data  $x$  and  $y$  are on columns d1 and d2, respectively:



0.				
[ r ]	[ a ]	[ b ]	[ Se ]	[ x' ]

Determine coefficients,  $a$  and  $b$ , and the correlation coefficient,  $r$ :

PF2 [a] 2nd F  $e^x$

0.306304729				
[ r ]	[ a ]	[ b ]	[ Se ]	[ x' ]

PF3 [b]

0.363606102				
[ r ]	[ a ]	[ b ]	[ Se ]	[ x' ]

PF1 [r]

0.982833978				
[ r ]	[ a ]	[ b ]	[ Se ]	[ x' ]

Next, estimate the value of  $y$  when  $x = 12$ . Press the  $\blacktriangle$  key to display the key label [y'] and press the desired PF key.

$\blacktriangle$  12 PF1 [y'] 2nd F  $e^x$

24.04911989				
[ y' ]	[ n ]	[ $\bar{x}$ ]	[ $\Sigma x$ ]	[ $\Sigma x^2$ ]

(Estimated value of  $y$ : approx. 24.05)

- As well as exponential regression, logarithmic regression ( $y = a + b \cdot \ln x$ ), power regression ( $y = ax^b$ ) and inverse regression ( $y = a + b/x$ ) can also be calculated in a similar manner. The following summarizes the data entry and operation procedures for these regressional calculations:

	Exponential regression	Logarithmic regression	Power regression	Inverse regression
Entry of sample data $x$	$x$ (as it is)	$\ln x$ ( $x > 0$ )	$\ln x$ ( $x > 0$ )	$1/x$ ( $x \neq 0$ )
Entry of sample data $y$	$\ln y$ ( $y > 0$ )	$y$ (as it is)	$\ln y$ ( $y > 0$ )	$y$ (as it is)
Coefficient $a$	PF2 [a] 2nd F $e^x$	PF2 [a]	PF2 [a] 2nd F $e^x$	PF2 [a]
Coefficient $b$	PF3 [b]	PF3 [b]	PF3 [b]	PF3 [b]
Estimated value $x'$	PF5 [x']	PF5 [x'] 2nd F $e^x$	PF5 [x'] 2nd F $e^x$	PF5 [x'] 1/X
Estimated value $y'$	PF1 [y'] * 2nd F $e^x$	PF1 [y'] *	PF1 [y'] * 2nd F $e^x$	1/X PF1 [y'] *

- Correlation coefficient and residual values corresponding to transformed  $X$  and  $Y$ .

\* Press the  $\blacktriangle$  key if necessary.

## Variable Storage

- Sample data will be stored in the same area as the single-precision array variable MD (\*, \*) in BASIC mode. The first subscript will increase as data is entered and the second subscript can be a value from 1 to 9 representing the number of columns of data.

The data for statistical or regression calculation are stored into array variable MD (n, m) as follows.

	$d_1$	$d_2$	$d_3$	.....	$d_{m+1}$
001	MD (0, 0)	MD (0, 1)	MD (0, 2)		MD (0, m)
002	MD (1, 0)	MD (1, 1)	MD (1, 2)		MD (1, m)
⋮					
n+1	MD (n, 0)	MD (n, 1)	MD (n, 2)		MD (n, m)

- The statistical or regression results are automatically stored into the fixed variables J to Z.

	Z	Y	X	W	V	U	T	S	R
Value	$n$	$\Sigma x$	$\Sigma x^2$	$\Sigma xy$	$\Sigma y$	$\Sigma y^2$	$\bar{x}$	$sx$	$\sigma x$

	Q	P	O	N	M	L	K	J
Value	$\bar{y}$	$sy$	$\sigma y$	$r$	$a$	$b$	0	0

If an error occurs, the variable contents do not change. The variable contents not calculated will become 0 (zero).

- The variable contents are cleared by the execution of a BASIC statement (i.e. RUN, CLEAR, NEW, ARUN, ERASE) or Engineer Software programs.
- Unregistered data ("?" on display) will be stored as  $-9.999999999E + 99$ .



## Errors and Error Messages in Statistical and Regression Calculations

In statistical and regression calculations, many operations may be required and this may cause some error due to truncation of numbers.

If an error occurs during statistical or regression calculation, the following error messages will be displayed, along with symbol "E". To clear the error message, press

**C•CE**.

<b>Error message</b>	<b>Description</b>
MEMORY OVER	<ul style="list-style-type: none"><li>• Memory overflow occurred during entry of a new table or addition of data.</li><li>• Result storage area has overflowed in the course of regression calculation.</li></ul>
CALCULATION ERROR	<ul style="list-style-type: none"><li>• An operation error has occurred during regression calculation.</li></ul>
IMPOSSIBLE CALCULATION	<ul style="list-style-type: none"><li>• Regression calculation was performed with 1 column of the matrix.</li></ul>
DATA ERROR	<ul style="list-style-type: none"><li>• Attempt has been made to enter a number greater than or equal to <math>10^{99}</math>.</li><li>• Data contains a value illegal for statistical or regression calculation.</li></ul>
O. <sup>E</sup>	<ul style="list-style-type: none"><li>• An operation error has occurred.</li></ul>

# 10. MATRIX MODE



The computer has a function to perform matrix operations or calculate determinant values. A matrix is a rectangular array  $a_{ik}$  ( $i = 1, 2, \dots, m; k = 1, 2, \dots, n$ ) of a given set of numbers ( $m \times n$  elements) as shown below.

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}$$

Such an array is expressed on this computer as matrix  $X$ ,  $Y$ ,  $M$ , or  $MA$  to  $MZ$ . Each of the numbers which form a matrix is called a matrix element. Matrix element  $a_{11}$  is expressed as  $X(1, 1)$ ,  $Y(1, 1)$ ,  $M(1, 1)$  or  $MA(1, 1)$  to  $MZ(1, 1)$ . The horizontal arrangement of matrix elements is called a row while the vertical arrangement is called a column. With the computer, 29 matrices  $X$ ,  $Y$ ,  $M$ , and  $MA$  to  $MZ$  can be defined. Each matrix can be defined within a range of 1 to 256 both vertically (i.e. columns) and horizontally (i.e. rows). However, the total matrix size is dependent upon the memory capacity of the computer. Matrix data can be stored to the single-precision array variables  $X$ ,  $Y$ ,  $M$ , or  $MA$  -  $MZ$ . Matrix  $MD$  is assigned to the same array variable as that to which sample data for statistical or regression calculations is assigned.

## Entry of Matrix Element

Press the **PF3** [MATRIX] key on the main menu to select the MATRIX mode. In this mode, you can enter the elements of a matrix for matrix operations, as well as have the computer perform operations on and display the matrix elements entered. The keys and their functions used to enter and display matrix elements are as described below:

Key	Function
<b>PF1</b> [X-DIM]	Stores the number of rows and the number of columns for matrix $X$ . Then the computer waits for element entry for matrix $X$ .
<b>PF2</b> [X-DATA]	The elements of matrix $X$ can be entered. The number of rows and the number of columns for matrix $X$ must be stored first.
<b>PF3</b> [Y-DIM]	Stores the number of rows and the number of columns for matrix $Y$ . Then the computer waits for the element entry for matrix $Y$ .
<b>PF4</b> [Y-DATA]	The elements of matrix $Y$ can be entered. The number of rows and the number of columns for matrix $Y$ must be stored first.
<b>PF5</b> [CALC]	Performs the matrix operation. The element entry for both matrices $X$ and $Y$ (or matrix $X$ only) must be completed first.
	Stores the matrix data and then the computer waits for the next data entry.
	Shifts the cursor to the right by one column. (When the cursor is at the rightmost column, the cursor moves to the leftmost element of the following row.)

Key	Function
	Shifts the cursor to the left by one column. (When the cursor is at the leftmost column, the cursor moves to the rightmost element of the preceding row.)
	Shifts the cursor up by one row (i.e. to the element immediately above the current element).
	Shifts the cursor down by one row (i.e., to the element immediately below the current element).
2nd F	Displays the rightmost element of the current row.
2nd F	Displays the leftmost element of the current row.
2nd F	Displays the top element of the current column.
2nd F	Displays the bottom element of the current column.

When you enter the respective elements of a matrix, you may use any of the keys that you use in the CAL mode for the four arithmetic operations and scientific calculations.

**Example:**

Enter the following two matrices:

$$X = \begin{bmatrix} 10/3 & -5 & 2 \\ 8 & 2 & 23 \end{bmatrix} \quad Y = \begin{bmatrix} 5/3 & 3 & 2 \\ -1 & 0 & -8 \end{bmatrix}$$

**Operation:**

[MATRIX]

In this manual, the display at right is called the matrix menu.

```

X ( 0 , 0 )      Y ( 0 , 0 )      M ( 0 , 0 )
[ X-DIM ] [ X-DATA ] [ Y-DIM ] [ Y-DATA ] [ CALC ]

```

Enter the number of rows and the number of columns for matrix X.

[X-DIM]

```

ENTER DIMENSION
X ( 0 , 0 )      Y ( 0 , 0 )      M ( 0 , 0 )
[CANCEL]

```

2 3

```

ENTER DIMENSION
X ( 2 , 3 )      Y ( 0 , 0 )      M ( 0 , 0 )
[CANCEL]

```

```

X ( 1 , 1 ) =      0 .
[ VIEW ] [ QUIT ]

```

After the size of matrix X is defined, the computer waits for the element data entry for X (1, 1).

10 3

```

X ( 1 , 1 ) =      3 . 333333333
[ VIEW ] [ QUIT ]

```

**[←]**

$X(1,2) =$	0.
[ VIEW ]	[ QUIT ]

After the element entry for  $X(1,1)$ , the computer waits for the next element entry for  $X(1,2)$ .

5 **[+/-]**

$X(1,2) =$	-5.
[ VIEW ]	[ QUIT ]

**[←]** 2

$X(1,3) =$	2.
[ VIEW ]	[ QUIT ]

Enter the rest of the elements of matrix  $X$  (to  $X(2,3)$ ).

**[←]** 8 **[←]**  
2 **[←]** 23 **[←]**

$X(2,3)$	$Y(0,0)$	$M(0,0)$
[X-DIM]	[X-DATA]	[Y-DIM]
[Y-DIM]	[Y-DATA]	[ CALC ]

The entry of all the element data for matrix  $X$  is complete. Display the value of each element.

**[PF2]** [X-DATA] **[PF1]** [VIEW]

3.333333333	-5	2
8	2	23
[ QUIT ]		

The matrix elements are displayed in matrix format (up to 3 rows and 3 columns of the matrix).

Return to the previous display.

**[PF5]** [QUIT]

$X(1,1) =$	3.333333333
[ VIEW ]	[ QUIT ]

Pressing the **[PF5]** [QUIT] key again returns to the matrix menu.

**[PF5]**

$X(2,3)$	$Y(0,0)$	$M(0,0)$
[X-DIM]	[X-DATA]	[Y-DIM]
[Y-DIM]	[Y-DATA]	[ CALC ]

Define the size of matrix  $Y$  in the same manner as you did for matrix  $X$ .

**[PF3]** [Y-DIM]

ENTER DIMENSION		
$X(2,3)$	$Y(0,0)$	$M(0,0)$
[CANCEL]		

2  $\leftarrow$  3  $\leftarrow$

Y ( 1 , 1 ) = 0 .

[ VIEW ] [ QUIT ]

After the size of matrix Y is defined, the computer waits for the element entry for Y ( 1 , 1 ). Enter the rest of the elements of matrix Y.

5  $\div$  3 =  $\leftarrow$   
 3  $\leftarrow$  2  $\leftarrow$   
 1 +/-  $\leftarrow$  0  $\leftarrow$   
 8 +/-  $\leftarrow$

X ( 2 , 3 )      Y ( 2 , 3 )      M ( 0 , 0 )

[ X-DIM ] [ X-DATA ] [ Y-DIM ] [ Y-DATA ] [ CALC ]

## Matrix Operations

Perform matrix operations. Press the **PF5** [CALC] key to display the matrix menu.

**PF5**

X ( 2 , 3 )      Y ( 2 , 3 )      M ( 0 , 0 )

[ X<sup>-1</sup> ] [ X<sup>t</sup> ] [ det X ] [ SCALAR ] [ See X ]

Determine the transpose matrix of X.

**PF2** [ X<sup>t</sup> ]

X ( 3 , 2 )      Y ( 2 , 3 )      M ( 0 , 0 )

[ X<sup>-1</sup> ] [ X<sup>t</sup> ] [ det X ] [ SCALAR ] [ See X ]

The symbol "◆" below **MAT** on the display indicates that information yet to be displayed exists and the  $\blacktriangleleft$  key can be used.

**PF5** [SeeX] **PF1** [VIEW]

3.33333333      8  
                   -5      2  
                   2      23  
                   .  
[ QUIT ]

Return to the previous display.

**PF5** **PF5**

X ( 3 , 2 )      Y ( 2 , 3 )      M ( 0 , 0 )

[ X<sup>-1</sup> ] [ X<sup>t</sup> ] [ det X ] [ SCALAR ] [ See X ]

Perform the transpose matrix operation to obtain the original matrix X.

**PF2** [ X<sup>t</sup> ]

X ( 2 , 3 )      Y ( 2 , 3 )      M ( 0 , 0 )

[ X<sup>-1</sup> ] [ X<sup>t</sup> ] [ det X ] [ SCALAR ] [ See X ]

Press the  key. The key label display will change.



$X(2, 3)$	$Y(2, 3)$	$M(0, 0)$
[ EQU ]	[ RESID ]	[ $X^2$ ]

The matrix operation performs these operations and arithmetic operations. The keys and their functions are described below.

Key	Function
[ $X^{-1}$ ]	$X^{-1} \rightarrow X$ : Performs the inverse matrix calculation of matrix $X$ . The result of this operation becomes new matrix $X$ . Note: To perform this operation, matrix $X$ must be a square matrix (having the same number of rows as the number of columns).
[ $X^t$ ]	$X^t \rightarrow X$ : Performs the transposition of matrix $X$ , giving the transposed matrix as the new matrix $X$ .
[det $X$ ]	$X$   $\rightarrow X$ (Display): Displays the value of the determinant of matrix $X$ . Note: To perform this operation, matrix $X$ must be a square matrix.
[SCALAR]	Performs the arithmetic operation with scalar.
[See $X$ ]	Displays the contents of matrix $X$ .
[EQU]	Solves simultaneous linear equations with $n$ unknowns.
[RESID]	Determines the residual.
[ $X^2$ ]	$X \cdot X \rightarrow X$ : Performs the squaring of matrix $X$ . Note: To perform this operation, matrix $X$ must be a square matrix.
[+]	$X + Y \rightarrow X$ : Performs addition. The result of adding matrix $X$ to matrix $Y$ becomes new matrix $X$ . Note: To perform addition, matrices $X$ and $Y$ must be equal to each other in both the number of rows and the number of columns.
[-]	$X - Y \rightarrow X$ : Performs subtraction. The result of subtracting elements of matrix $Y$ from the corresponding elements of matrix $X$ becomes the corresponding elements of new matrix $X$ . Note: To perform subtraction, matrices $X$ and $Y$ must be equal in both the number of rows and the number of columns. Example: $\begin{bmatrix} 2 & 3 \\ 5 & 2 \end{bmatrix} - \begin{bmatrix} 1 & 6 \\ 3 & -1 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & -3 \\ 2 & 3 \end{bmatrix}$
[ $X*$ ]	$X \cdot Y \rightarrow X$ : Performs the multiplication of matrices $X$ and $Y$ . Note: To perform multiplication, the number of columns in matrix $X$ must be equal to the number of rows in matrix $Y$ .

Key	Function
$\boxed{\div}$	$X \cdot Y^{-1} \rightarrow X$ : Performs the multiplication of matrix $X$ and inverse of matrix $Y$ . Note: To perform this operation, the number of columns in matrix $X$ must be equal to the number of rows in matrix $Y^{-1}$ . The matrix $Y$ must be a square matrix.
$\boxed{\leftrightarrow}$	$X \leftrightarrow Y$ : Exchanges matrix $X$ for matrix $Y$ .
$\boxed{+/-}$	$-X \rightarrow X$ : Reverses the positive or negative sign of each element of matrix $X$ .
$\boxed{X \rightarrow M}$	$X \rightarrow M$ : Stores the value of matrix $X$ in the memory location of matrix $M$ (while clearing the previous contents of matrix $M$ ). Used for retaining the value of matrix $X$ even after a matrix operation.
$\boxed{RM}$	$M \rightarrow X$ : Recalls the memory contents of matrix $M$ into matrix $X$ (while clearing the previous contents of matrix $X$ .)
$\boxed{M+}$	$X + M \rightarrow M$ : Adds the value of matrix $X$ cumulatively to the memory contents of matrix $M$ . Note: To perform this operation, matrices $X$ and $M$ must be equal to each other in both the number of rows and the number of columns.
$\boxed{STO} \quad \boxed{A}$ { $\boxed{STO} \quad \boxed{Z}$	$X \rightarrow MA$ : Stores matrix $X$ into matrix $MA$ { (previous contents of $MA$ are cleared). $X \rightarrow MZ$ : Stores matrix $X$ into matrix $MZ$ { (previous contents of $MZ$ are cleared).
$\boxed{RCL} \quad \boxed{A}$ { $\boxed{RCL} \quad \boxed{Z}$	$MA \rightarrow X$ : Recalls matrix $MA$ into matrix $X$ { (previous contents of matrix $X$ are cleared). $MZ \rightarrow X$ : Recalls matrix $MZ$ into matrix $X$ { (previous contents of matrix $X$ are cleared).

**Note:**

- Pressing the  $\boxed{BREAK}$  key during the execution of a matrix operation causes the calculation to be suspended. At this point, the values of matrices before the execution of the operation will be retained.

**Example:**

Perform  $X + Y$ , using the values of the respective elements of matrices  $X$  and  $Y$  entered (stored in memory) in the previous example.

$$X = \begin{bmatrix} 10/3 & -5 & 2 \\ 8 & 2 & 23 \end{bmatrix} \quad Y = \begin{bmatrix} 5/3 & 3 & 2 \\ -1 & 0 & -8 \end{bmatrix}$$

**Operation:**

Press the  $\boxed{BREAK}$  key to display the matrix menu if necessary.

```

X ( 2 , 3 )      Y ( 2 , 3 )      M ( 0 , 0 )
[X-DIM ] [X-DATA] [Y-DIM ] [Y-DATA] [ CALC ]

```

$\boxed{PF5}$  [CALC]

```

X ( 2 , 3 )      Y ( 2 , 3 )      M ( 0 , 0 )
[ X-1 ] [ Xt ] [ det X ] [SCALAR] [ See X ]

```

Execute the [CALC] function to perform the desired operation. When the [CALC] operation is executed, press the **BREAK** key if you wish to interrupt the operation.

**+**

X(2,3)	Y(2,3)	M(0,0)
[ X <sup>-1</sup> ]	[ X <sup>t</sup> ]	[ detX ] [SCALAR] [ SeeX ]

Recall the result.

**PF5** [SeeX]

X(1,1) =	5.
[ VIEW ]	[ QUIT ]

**PF1** [VIEW]

5	-2	4
7	2	15
		[ QUIT ]

## Scalar Operations

The computer can perform scalar operations.

### Example:

Perform  $X/25 \rightarrow X$ , using the calculation result of matrix  $X$  in the previous example.

$$X = \begin{bmatrix} 5 & -2 & 4 \\ 7 & 2 & 15 \end{bmatrix}$$

Display the previous display.

**PF5** **PF5**

X(2,3)	Y(2,3)	M(0,0)
[ X <sup>-1</sup> ]	[ X <sup>t</sup> ]	[ detX ] [SCALAR] [ SeeX ]

**PF4** [SCALAR]

SCALAR	0.
[ k*X ]	[ k*X <sup>-1</sup> ] [ X/k ] [ k+X ] [ k-X ]

25 **PF3** [X/k]

X(2,3)	Y(2,3)	M(0,0)
[ X <sup>-1</sup> ]	[ X <sup>t</sup> ]	[ detX ] [SCALAR] [ SeeX ]

The operation is completed. Display the result:

**PF5** **PF1**

0.2	-0.08	0.16
0.28	0.08	0.6
		[ QUIT ]



In scalar operations, the following can be performed: (k: scalar)

Key	Function
[k*X]	$k \cdot X \rightarrow X$ : Performs the multiplication of matrix $X$ elements by scalar $k$ .
[k*X <sup>-1</sup> ]	$k \cdot X^{-1} \rightarrow X$ : Performs the multiplication of inverse matrix $X^{-1}$ elements by scalar $k$ .
[X/k]	$X/k \rightarrow X$ : Performs the division of matrix $X$ elements by scalar $k$ .
[k + X]	$k + X \rightarrow X$ : Performs the addition of scalar $k$ to matrix $X$ elements. Mathematically, such an operation does not exist. The addition of scalars is one of the features unique to this computer.
[k - X]	$k - X \rightarrow X$ : Performs the subtraction of matrix $X$ elements from scalar $k$ . Mathematically, such an operation does not exist. The subtraction from scalars is another feature unique to this computer. Example: $2 - \begin{bmatrix} 1 & 6 \\ 3 & -1 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & -4 \\ -1 & 3 \end{bmatrix}$

**Note:**

- If most of the elements of a matrix have the same value, execute the [k + X] operation with all the matrix elements set to 0 and then correct only the value of each element having a value other than  $k$ . This will simplify the entry of the matrix elements.

## Examples of Matrix Operations

Solve the following simultaneous linear equations with three unknowns using matrix operations.

$$\begin{cases} 2x + 5y - z = -1 \\ x - y + 4z = 12 \\ 3x + 2y + z = 9 \end{cases}$$

To solve simultaneous linear equations with  $n$  unknowns,  $Ax = y$ , this computer calculates  $A$  as matrix  $X$  and  $y$  as matrix  $Y$ . The following operations are performed:  $X \rightarrow M$ ,  $X^{-1} \cdot Y \rightarrow X$ . The matrix  $X$  is stored into matrix  $M$ . (The previous contents of matrix  $M$  will be cleared.) The inverse matrix of  $X$  is multiplied by the matrix  $Y$ . The result will be stored in the matrix  $X$ . Coefficients of the equations remain in matrix  $M$ .

Enter the following matrices:

$$X = \begin{bmatrix} 2 & 5 & -1 \\ 1 & -1 & 4 \\ 3 & 2 & 1 \end{bmatrix} \quad Y = \begin{bmatrix} -1 \\ 12 \\ 9 \end{bmatrix}$$

Then perform the [EQU] function to solve the simultaneous linear equations.

Enter the matrix elements for  $X$  and  $Y$  according to the example above.

X (3, 3)	Y (3, 1)	M (0, 0)
[X-DIM ]	[X-DATA]	[Y-DIM ] [Y-DATA] [ CALC ]

[PF5] [CALC]

X (3, 3)	Y (3, 1)	M (0, 0)
[ X <sup>-1</sup> ]	[ X <sup>t</sup> ]	[ det X ] [SCALAR] [ See X ]



```

X (3,3)      Y (3,1)      M (0,0)
[ EQU ] [RESID] [ X2 ]

```

**PF1** [EQU]

```

X→M, X-1*Y→X
Push BREAK to Stop

```

```

X (3,1)      Y (3,1)      M (3,3)
[ X-1 ] [ Xt ] [ detX ] [SCALAR] [ SeeX ]

```

Display the results.

**PF5** [SeeX]

```

X (1,1) = 3.
[ VIEW ] [ QUIT ]

```

**PF1** [VIEW]

```

3
-1
2
[ QUIT ]

```

Thus, the x, y, and z values in the equations are as follows:

$$x = 3, y = -1, z = 2$$

**Note:**

Matrix operations are based on the method of elimination in common use. However, due to the nature of numerical calculations by any computer, an error may occur in the calculation of a determinant or an inverse matrix due to truncation.

The [RESID] function evaluates the error in the solution determined with the [EQU] function, stores to matrix X, the result of subtraction of matrix Y from the product of matrix M(matrix X before solving the simultaneous equation) and matrix X (solution obtained from simultaneous equation). The [RESID] function clears the obtained solution. To obtain the solution, save the solution in another matrix, MA to MZ, by pressing the **ST0** key if it is needed.

### Example:

To solve for the inverse matrix of  $\begin{bmatrix} 3 & 1 \\ 1 & 1/3 \end{bmatrix}$

This matrix is not a regular matrix and thus theoretically has no inverse matrix. With any computer, however, the value 1/3 is input as "0.33...3" and thus an inverse matrix exists, resulting in the following.

$$\begin{bmatrix} 3 & 1 \\ 1 & 0.33 \dots 3 \end{bmatrix}^{-1} = \begin{bmatrix} -33 \dots 3 & 1E + 10 \\ 1E + 10 & -3E + 10 \end{bmatrix}$$

Results obtained by the computer may have a finite error as shown by this example. Please note that verification by another method may be required depending on how the matrix operations will be applied.

In the above example, when you obtain the determinant value by multiplying the original matrix  $X$  by 3, you can confirm that the matrix  $X$  is not a regular matrix because the determinant is 0 ( $3 \cdot X = 0$ ).

### Note:

Because a matrix operation will not be completed by a single operation (e.g. one-time multiplication), it will take some time to complete the operation. It will take approximately 6 seconds to solve for the inverse matrix of a unit matrix consisting of 10 rows and 10 columns. This computation time varies depending on the values of matrix elements.


## Variable Storage




Matrices  $X, Y, M, MA - MZ$  are stored in the same locations as the single-precision array variables  $X (*, *)$ ,  $Y (*, *)$ ,  $M (*, *)$ , and  $MA (*, *) - MZ (*, *)$  in BASIC. Thus, it is possible to enter element values in BASIC and obtain the calculation result in the MATRIX mode. Matrix operations are performed in the single-precision mode only.

Note the following when entering element values in BASIC:

- ① The matrix element  $X(i, k)$  corresponds to BASIC array  $X(i - 1, k - 1)$  (i.e.  $X(1, 2)$  corresponds to array element  $X(0, 1)$ ).
- ② The matrix values can be cleared by the BASIC command RUN, CLEAR, NEW, ARUN, or ERASE.

### Memory Capacity for Matrix Operation

- Memory for matrix operation are shared with BASIC programs. Unused memory space (found by entering FRE0  in the BASIC mode) must be more than the sum of the following to allow for intermediate calculations:
- $[(\text{No. of columns in } X) \times (\text{No. of rows in } X) \times 7 + 10]$  bytes  
+  $[(\text{No. of columns in used matrix}) \times (\text{No. of rows in used matrix}) \times 7 + 10]$  bytes  
+  $[(\text{No. of columns in intermediate matrix}) \times (\text{No. of rows in intermediate matrix}) \times 7 + 10]$  bytes

The intermediate matrix only exists during execution and is cleared after execution. The intermediate matrix is not needed for the ,  or  keys. When an inverse matrix is calculated or used for simultaneous linear equations, two intermediate matrices are required as two calculations are performed.

- When "MEMORY OVER" is displayed, clear the variables or programs used in BASIC to provide more free memory space for calculations.  
For a matrix with many elements, install a RAM card with a larger memory capacity to provide more free memory space. (See 3. RAM CARD.)


**Example:**

$$X = \begin{bmatrix} 2 & 3 \\ 5 & 1 \end{bmatrix} \quad Y = \begin{bmatrix} 8 & 30 \\ 7 & 15 \end{bmatrix}$$


To perform  $X \cdot Y \rightarrow X$ , the number of bytes required is:  
 $(2 \times 2 \times 7 + 10) + (2 \times 2 \times 7 + 10) + (2 \times 2 \times 7 + 10) = 114$  bytes

matrix X
matrix Y
intermediate matrix


### Printing a Matrix

To print the result of matrices (i.e. element values), create the following program and run the program using the GOTO command (since the contents of matrices will be cleared after execution using RUN ).

```
100: *MAT: INPUT "ROW=";II
110: INPUT "COLUMN=";JJ
120: FOR I = 0 TO II-1
130: FOR J = 0 TO JJ-1
140: LPRINT "X(";I+1;"; ";J+1;")=";X(I, J)
150: NEXT J:NEXT I:END
```

To print a matrix *Y* or *MC*, change the variable *X* on line 140 to *Y* or *MC*.  
 To print the matrix, enter GOTO \*MAT  in the RUN mode.

### Error Messages

If an error occurs during a matrix operation, one of the following error messages will be displayed, along with symbol "E". Press the  key to release the computer from the error condition and the "E" symbol will go off and the display will return to the matrix menu.

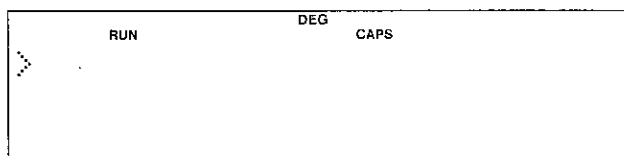
Error Message	Cause of Error
IMPOSSIBLE CALCULATION	<ul style="list-style-type: none"> <li>• Matrices do not match in size.</li> </ul> Matrices do not match in size during addition, subtraction, or multiplication, or an attempt was made to determine the inverse matrix, or to perform the squaring of a matrix which is not a square matrix.
MEMORY OVER	<ul style="list-style-type: none"> <li>• Insufficient memory</li> </ul> In an operation such as $X \rightarrow M$ or $X \rightarrow MA$ , memory space is insufficient to store matrix <i>M</i> or <i>MA</i> , or no work area is available for arithmetic operation.
DIVISION BY ZERO	<ul style="list-style-type: none"> <li>• 0 (zero) is used as divisor.</li> </ul> In an inverse matrix operation, an attempt was made to divide a number by zero.
OVERFLOW	<ul style="list-style-type: none"> <li>• Overflow has occurred during an operation.</li> </ul>

# 11. RUN MODE

The RUN mode is a very versatile operation mode, which allows calculations and operations available in the CAL, ENG, MATRIX, AER, and STAT modes, calculations using variables from those modes, and has the ability to run BASIC programs written in the PRO mode.

## Selecting RUN Mode

Select the BASIC mode by pressing the **MENU** **PF1** keys. If PRO is displayed, press the **BASIC** key to select the RUN mode. The prompt (>) tells you that the computer is awaiting entry. The display should now look like this:



## Some Helpful Hints

If you make an error during entry and get an error message, the simplest way to clear the error is to press the **C•CE** key and reenter. If the computer "hangs up" (you cannot get it to respond at all), press the RESET button while holding the **ON** key (see Appendix D).

The prompt (>) tells you that the computer is awaiting entry. As you enter data the prompt disappears and the cursor ( ) moves to the right, indicating the next available location in the display.

Pressing the right **▶**, left **◀**, up **↑** and down **↓** keys moves the cursor.

Press the **←** key to tell the computer that you have finished entering data and to signal the computer to perform the indicated operations. **You must press the ← key at the end of each line of entry or your calculations will not be acted upon by the computer.**

When performing numeric calculations, entries appear on the left of the display; the results appear on the right.



Do not use dollar signs or commas when entering calculations. These characters have special meanings in the BASIC programming language.

When using the **SHIFT** key to implement another key's second function, press and hold the **SHIFT** key and then press the other key. The **2nd F** key may also be used, as in the CAL mode.

When the **SHIFT** key is used, the character actually produced is represented in the following keystroke. For example pressing **SHIFT** + **Y** will produce the "&" character. This is written **SHIFT** + **&**.

Be sure to enter **C•CE** after each calculation (unless you are performing serial calculations). **C•CE** erases the display and resets any error condition. It does not erase anything stored in the computer memory.

- Recalling a formula after execution will cause a value such as "5E3" or "5E-3" to be displayed as "5000" or "0.005". A value such as "5.000" will be displayed as "5". If the value is outside the normal floating point display range, it will be displayed in scientific notation.
- Pressing the **0** key while holding the **CTRL** key will toggle the beep sound for key entry. Setting the beep while in the BASIC mode is effective for other operation modes.
- The computer has an 8 byte input buffer to hold up to 8 key entries while a computation is being performed.
- Pressing a number key while holding another number key will enter both values. For an example, the computer will recognize the entry as "12" if you press the **2** key while holding the **1** key.
- 8 bytes are used for a numeric value in single-precision mode while 13 bytes are used in double-precision mode. It is not always possible to store up to 254 characters in one line.

## Simple Calculations

The computer performs calculations in the RUN mode with 10-digit precision (unless set to the double-precision mode, which will be discussed later). Turn the power on and set it to the RUN mode. Now try these simple examples.

### Example:

$$2 + 3 \times 4 =$$

2 **+** 3 **X\*** 4 **←**



2+3*4
14

### Example:

$$5 \times (-6) + 7 =$$

5 **X\*** **-** 6 **+** 7 **←**

2+3*4
5*-6+7
14
-23

In the RUN mode, the  key must be pressed to obtain the calculated result instead of the  key.

The display of the computer consists of 4 lines (40 characters per line). Key entries and calculated results are displayed from the top line of the display. If the characters to be displayed exceed 4 lines, the displayed contents will be moved up by 1 line (the first line will move off the top of the display).

## Compound Calculations and Parentheses

You can combine several operations into one step as in the CAL mode. For example, you may enter:



$$675 + 6750/45000$$



Compound calculations, however, must be entered very carefully to avoid ambiguity.

When performing compound calculations, the computer has specific rules of expression evaluation and operator priority (see page 162). Use parentheses to clarify your expressions:

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

## Recalling Entries


Even after the computer has displayed the results of your calculation, you can verify that the entry was made correctly, and edit it if necessary. To edit, use the left arrow  and right arrow  keys.

Use the left arrow  key to position the cursor after the last character.  
Use the right arrow  key to position the cursor over the first character.

Remember that the left and right arrows are also used to position the cursor. The right and left arrows are very helpful in editing entries without having to retype the entire expression.



$$300 \div 6 \leftarrow$$

300/6	50
-------	----

Change this operation to 300/5.  
Recall your last entry using the  key.



300/6	50
300/6_	

Because you recalled the expression using , the cursor is positioned at the end of the display. Use  to move the cursor one space to the left.



300/6	50
300/6	

Notice that after you move the cursor, it becomes a flashing block. Whenever you position the cursor over an existing character, it will flash.

Enter a 5 to replace the 6. An important point in replacing characters — once you enter a new character over an existing character, the original is gone forever! You cannot recall an expression that has been erased.

5 



300/6	50
300/5	60

You can also insert or delete characters in an entry. Change the previous calculation to 3000/25.

Recall your entry using the  key.



300/5	50
300/5	60


Because you recalled using the  key, the flashing cursor is now over the first character. To make the correction you must insert a zero. Using the  key, move the cursor until it is over a zero. When making an INSert, position the flashing cursor over the character before which you wish to make the insertion.



300/5	50
300/5	60


**INS**

300/5	50
300/5	60

Pressing the **INS** key changes the shape of the flashing cursor () which points to the location where your entry will be inserted. Enter the zero. Move the cursor over the 5 and enter 2. Once the entry is corrected, display your new result.

0    2 

300/5	60
3000/25	120

Pressing the **INS** key enters the insert mode and pressing the **INS** key again or the  key exits the insert mode.



To DElete a character, use the **DEL** key. Change the previous calculation to 3/5. Recall your entry using **▶**.



3000/25	60
3000/25	120

The flashing cursor is now positioned over the first character in the display. To correct this entry, eliminate the zeros. Using **▶**, move the cursor to the first zero. To delete a character, always position the cursor over the character to be deleted.



3000/25	60
3000/25	120

Now use the DElete key to delete the zeros and the two.



3000/25	60
3/5	120

Pressing the **DEL** key deletes the character under the cursor and shifts all the following characters one space to the left. Since you have no other changes to make, complete the calculation by displaying the result.



3000/25	120
3/5	0.6

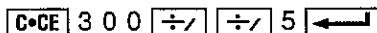
### Note:

Pressing the **SPACE** key when the cursor is positioned over a character replaces the character, leaving a blank space. DElete eliminates the character and the space it occupied.



You can also use the **BS** key to delete errors. Note that pressing the **BS** key moves the cursor back one position and deletes the character there, while pressing the **DEL** key deletes the character the cursor is positioned over.



## Errors

Recalling your last entry is essential if you get an error message. Let us imagine that, unintentionally, you typed this into the computer:






300//5
Syntax error

"Syntax error" is the computer's way of saying, "I don't know what you want me to do here". Press the  or  key to move the flashing cursor to where the error occurred.

 (or )

```
300//5
Syntax error
300//5
```

Use the  key to correct this error.

```
300//5
Syntax error
300/5
```

60

If, upon recalling your entry after a syntax error, you find that you have omitted a character, use the INSert sequence to insert it.




When using the computer as a calculator, the majority of errors you encounter will be syntax errors. For a complete listing of error messages, see Appendix A.

## Serial Calculations


The computer allows you to use the results of one calculation as part of the following calculation.


### Example:

What is 15% of 300 \* 150?


 3 0 0  1 5 0 

```
300*150
45000
```





In serial calculations it is not necessary to retype your previous results, but DO NOT press the  key between entries.

 . 1 5

```
300*150
45000
45000*.15_
45000
```

Notice that as you type in the second calculation (\*.15), the computer automatically displays the result of your first calculation at the left of the screen and includes it in the new calculation. In serial calculations the entry must begin with an operator. As always, you end the entry with the  key.

### Note:

The  and  keys cannot be used in percent calculations in RUN mode. The  key should be used as a character only and the  key is inoperative.

**Example:**

4 5 0 0 0  $\times$  15 SHIFT +  $\frac{x}{T}$   $\leftarrow$  → Syntax error

$\leftarrow$

300*150	45000
45000*.15	6750

To change the sign of the previous result, multiply by -1:

$\times$  - 1  $\leftarrow$

45000*.15	6750
6750*-1	-6750

## Using Variables in Calculations

The computer can store up to 26 simple numeric variables under the alphabetic characters A to Z. If you are unfamiliar with the concept of variables, they are more fully explained in Chapter 12. Variables are designated with an Assignment Statement:

A = 5  $\leftarrow$   
B = -2  $\leftarrow$

You can also assign the value of one variable (right) to another variable (left).

C = A + 3  $\leftarrow$   
D = C  $\leftarrow$

As you press  $\leftarrow$ , the computer performs the calculation and displays the new value of the variable. You can display the current value of any variable by entering the alphabetic character it is stored under:

$\text{C}\cdot\text{CE}$  C  $\leftarrow$

C	8
---	---

Variables will retain their assigned values even if the computer is switched OFF or undergoes an Auto OFF. Variables are lost only when:

- You assign a new value to the same variable.
- You enter CLEAR  $\leftarrow$  (not the  $\text{C}\cdot\text{CE}$  key).
- You clear the computer using the RESET button.

There are certain limitations on the assignment of variables, and certain programming procedures that cause them to be changed. See Chapter 12 for a discussion of assignment. See Chapter 12 for a discussion of the use of variables in programming.

## Single-Precision, Double-Precision

The largest number that the computer can handle in the single-precision mode is ten significant digits, with a two-digit exponent.

In the double-precision mode, it is increased to 20 significant digits, or 9.999999999999999999 D 99 and 9.999999999999999999 D -99 (the capital E, indicating the exponent, is replaced by a capital D in double-precision mode).

### Selecting Double-Precision Mode

1. Enter RUN mode and press the **C•CE** key to clear the display.
2. Enter DEFDBL and press the **←** key.

DBL is shown on the status line of the display, indicating that the computer is now in the double-precision mode.

The exponent is indicated by a capital letter D in the double-precision mode, as described in the previous section. However, you can still use either the E or D as a character when entering the exponent in an expression. See page 173 for more details.

### Canceling Double-Precision Mode

1. In the RUN mode, press the **C•CE** key to clear the display.
2. Enter DEFSNG and press the **←** key.

DBL is removed from the status line, indicating that the computer is now in the single-precision mode.

The double-precision mode is automatically canceled if:

1. The computer is switched OFF.
2. The RUN, NEW or CLEAR command is executed.

## Last Answer Feature

In a simple calculation, the result of the previous calculation can only be used in continuous calculations as the first number.

### Example:

3 **+** 4 **←**

3+4	7
-----	---

**X\*** 5 **←**

3+4	7
7*5	35

However, the computer has a feature that lets you recall the result of the previous calculation and use it in any location in the current calculation. This is called the last answer feature. It allows the previous answer to be recalled any number of times by pressing the **ANS** key. If you entered the last example, press **C•CE** then **ANS** and you will see "35" displayed.

Let's look at an example where a previous result is used twice in the current calculation. Note that in this example, the last answer changes and is updated with the current answer each time **←** is pressed.

**Example:**

Use the result (6.25) of the operation,  $50 \div 8$ , to compute  $12 \times 5/6.25 + 24 \times 3/6.25 =$

$50 \div 8$  **←**

$7 \times 5$	35
$50/8$	6.25
<small>Last answer</small>	

$12 \times 5$  **÷** **ANS**

$50/8$	35
$12 \times 5/6.25$	6.25
<small>Last answer recalled</small>	

**+**  $24 \times 3$  **÷** **ANS**

$50/8$	35
$12 \times 5/6.25 + 24 \times 3/6.25$	6.25

**←**

$50/8$	6.25
$12 \times 5/6.25 + 24 \times 3/6.25$	21.12


**C•CE** **ANS**

$21.12$	
---------	--

Pressing **←** causes the previous "last answer" to be replaced with the result of the latest calculation. The last answer is not, however, cleared by pressing the **C•CE** or **SHIFT** + **CA** keys but when the power is turned off.

The last answer cannot be recalled when the computer is not in the RUN mode. The last answer is replaced when a program is executed.

## Maximum Calculation Length

The length of the calculation that can be entered is limited to 254 key strokes before the  key is pressed. If you try to exceed this limit, the cursor will start flashing to show that further input is invalid. If this happens, break down the calculation into two or more steps.

Eight bytes are used for each numeric value in single-precision mode, so that it is not always possible to enter 254 keystrokes. (13 bytes per numeric value in double-precision mode.)

## Scientific Calculations

The computer has a wide range of numeric functions for use in scientific calculations. PART 4 contains an alphabetical listing of these functions. Note that the notation of the functions in BASIC may differ from conventional mathematical notations.

All scientific functions may be entered in the RUN mode either by pressing the appropriate function key or entering the BASIC command.

The computer also enables specification of angular units in degrees, radians or gradient using the DEGREE, RADIAN or GRAD commands.


Angular unit	Command	Description
Degrees	DEGREE	Represents a right angle as 90[°].
Radians	RADIAN	Represents a right angle as $\pi/2$ [rad].
Grads	GRAD	Represents a right angle as 100[g].

For practice, use these instructions to specify angular units when required in the following calculation examples:



**Example:**  $\sin 30 =$

**Operation:**

DEGREE  (Specifies "degree" for angular unit.)

S I N 3 0 


or

 3 0 

DEGREE	SIN30	0.5
--------	-------	-----

**Example:**  $\tan \pi/4 =$

**Operation:**

RADIAN  (Specifies "radian" for angular unit.)

T A N (  P I  4  )  


RADIAN	TAN(PI/4)	1
--------	-----------	---

**Example:**  $\cos^{-1}(-0.5) =$

**Operation:**

DEGREE  $\leftarrow$  (Specifies "degree" for angular unit.)

ACS  $\leftarrow$  0.5  $\leftarrow$

DEGREE	1
ACS-0.5	120

**Example:**  $\log 5 + \ln 5 =$

**Operation:**

LOG 5  $\leftarrow$  LN 5  $\leftarrow$

ASC-0.5	120
LOG5+LN5	2.308407917

**Example:**  $e^{2+3} =$

**Operation:**

EXP ( 2 + 3 )  $\leftarrow$

LOG5+LN5	2.308407917
EXP(2+3)	148.4131591

**Example:**  $\sqrt{4^3 + 6^4} =$

**Operation:**

$\leftarrow$  SQR ( 4  $\leftarrow$  3 )  $\leftarrow$  + 6  $\leftarrow$  4 )  $\leftarrow$

SQR(4^3+6^4)	36.87817783
--------------	-------------

**Example:**

Convert 30 deg. 30 min. in sexagesimal notation into decimal notation.

**Operation:**

DEG 30.30  $\leftarrow$

SQR(4^3+6^4)	36.87817783
DEG30.30	30.5

(30.5 DEGREE)

**Example:**

Convert 30.755 deg. in decimal notation to sexagesimal notation.

**Operation:**

DMS 30.755  $\leftarrow$

DEG30.30	30.5
DMS30.755	30.4518

(30 DEG. 45 MIN. 18 SEC.)

**Example:**

Conversion from rectangular into polar coordinates: Determine the polar coordinates (r,  $\theta$ ) for the point P(3, 8) in rectangular coordinates:

**Operation:**

DEGREE  $\leftarrow$  (Specifies "degrees" for angular unit.)

P O L ( ( 3 , 8 ) )  $\leftarrow$

	30.4518
DEGREE	
POL(3,8)	8.544003745

Z  $\leftarrow$

POL(3,8)	8.544003745
Z	69.44395478

- The value of  $\theta$  is stored in variable Z, and the value of r in variable Y.

**Example:**

Conversion from polar into rectangular coordinates: Determine the rectangular coordinates (x, y) for the point P(12,  $4\pi/5$ ) in polar coordinates.

**Operation:**

$\leftarrow$  RADIAN  $\leftarrow$  (Specifies "radians" for angular unit.)

R E C ( ( 1 2 , ( ( 4  
 $\div$  5  $\times$  P I ) ) ) )  
 $\leftarrow$

	-9.708203933
RADIAN	
REC(12,(4/5*PI))	

Z  $\leftarrow$

REC(12,(4/5*PI))	-9.708203933
Z	7.053423028

- The values of y and x are stored in variables Z and Y, respectively.

**Note:**

For coordinate conversion, the conversion results are stored in variables Z and Y. Therefore, the previous contents of Z and Y will be cleared.

**Example:**

Convert the hexadecimal number CF8 to its decimal equivalent.

**Operation:**

$\leftarrow$  2nd F & H C F 8  
 $\leftarrow$

&HCF8	3320
-------	------

"&H" represents a hexadecimal value.



— Reference —

Expressions composed of relational operators ( $=$ ,  $>$ ,  $<$ ,  $\geq$ ,  $\leq$ ,  $<>$ ) can take on the values listed in the following table:

$x$  and  $y$  represent numeric values.

$=$	-1 if $x = y$ 0 if $x \neq y$	$\geq$	-1 if $x \geq y$ 0 if $x < y$
$>$	-1 if $x > y$ 0 if $x \leq y$	$\leq$	-1 if $x \leq y$ 0 if $x > y$
$<$	-1 if $x < y$ 0 if $x \geq y$	$<>$	-1 if $x \neq y$ 0 if $x = y$ (“ $<>$ ” means “ $\neq$ ”)

- If, for example, “ $A = \text{numeric value}$ ” or “ $B = \text{formula}$ ” is used in a logical equation, the computer will not treat it as a logical equation but as an assignment statement for variables. When using an equal ( $=$ ) sign for logical equation, use it in the form of “ $\text{numeric value} = A$ ” or “ $\text{formula} = B$ ”, with the exception of conditional expressions used in IF statements.

**Direct Calculation Feature**

In the manual calculations described up to now, the  $\left[ \leftarrow \right]$  key has always been used to terminate a formula and obtain the calculation result of the formula. However, you can directly operate the functions of the computer with the desired function key (without operating the  $\left[ \leftarrow \right]$  key) when the objective numeric data is in the display.

**Example:**

Determine  $\sin 30^\circ$  and  $8!$ .

**Operation:**

DEGREE  $\left[ \leftarrow \right]$   
 $\left[ \text{C} \cdot \text{CE} \right]$  3 0  $\left[ \sin \right]$

SIN30	0.5
-------	-----

**Operation:**

$\left[ \text{C} \cdot \text{CE} \right]$  8  $\left[ 2\text{nd F} \right]$   $\left[ n! \right]$

FACT 8	40320
--------	-------

**Example:**

For  $\tan^{-1} \frac{5}{12}$ , first check the result of  $\frac{5}{12}$ , then determine  $\tan^{-1} \frac{5}{12}$ .

**Operation:**

DEGREE  $\left[ \leftarrow \right]$   
 5  $\left[ \div \right]$  12  $\left[ \leftarrow \right]$   $\left[ 2\text{nd F} \right]$   $\left[ \tan^{-1} \right]$

5/12	4.16666667E-01
ATN 4.16666667E-01	22.61986495

It should be noted, however, that this "direct" calculation mode is not available for functions requiring the entry of more than one numeric value (binominal functions) such as power, root, or coordinate conversion.

The direct calculation feature is not effective for formulas:

(e.g.)  $\boxed{C\cdot CE} 5 \boxed{X\cdot} 4 \rightarrow 5\cdot 4$   
 $\boxed{\log} \rightarrow 5\cdot 4\text{LOG}$

If no data is on the display, pressing a function key will display the corresponding BASIC command.

The direct calculation feature is effective only for numeric values. Therefore, if hexadecimal numbers A to F are entered for hex to decimal conversion, the direct calculation feature will remain inoperative. In such a case, perform an ordinary manual calculation using the  $\boxed{\leftarrow}$  key.

## Priority in Direct Input Calculations

You can enter formulas in the exact order in which they are written, including parentheses or functions. The order of priority in calculation and treatment of intermediate results will be taken care of by the computer.

The internal order of priority in manual calculation is as follows:

1. Recalling variables or PI
2. Function (sin, cos, etc.)
3. Power ( $\wedge$ ), root (ROT)
4. Sign (+, -)
5. Multiplication or division ( $\cdot$ , /)
6. Addition or subtraction (+, -)
7. Comparison of magnitude (>, >=, <, <=, <>, =)
8. Logical AND, OR, XOR

### Note:

- If parentheses are used in a formula, the operation given within the parentheses has the highest priority.
- Composite functions are operated from right to left ( $\sin \cos^{-1} 0.6$ ).
- Chained power ( $3^4$  or  $3 \wedge 4 \wedge 2$ ) is operated from right to left.
- For items 3) and 4) above, the last entry has higher priority.  
(e.g.)  $-2 \wedge 4 \rightarrow -(2^4)$   
 $3 \wedge -2 \rightarrow 3^{-2}$

## Printing for Direct Input Calculations

The calculation steps and results can be printed if the optional printer is connected and switched on, and the  $\boxed{\text{SHIFT}} + \boxed{\text{P}\cdot\text{NP}}$  keys are pressed (Print mode). Note that calculations made in the CAL mode cannot be printed out.

If a printout is not desired, either switch off the printer, or press  $\boxed{\text{SHIFT}} + \boxed{\text{P}\cdot\text{NP}}$  again (Non-Print mode).

# Calculation Errors

The following types of errors occur in ordinary calculators, pocket computers, and personal computers:

## Errors due to Least Significant Digit Processing

Usually, the maximum number of digits that can be calculated in a computer is fixed. For example,  $4/3$  results in  $1.3333333333\dots$ . In a computer with a maximum of 8 digits, the 8 digits are significant digits; other least significant digits are either truncated or rounded.

### Example:

Computer with 10 significant digits

$$4 \left[ \frac{\div}{/} \right] 3 \left[ \leftarrow \right] \rightarrow 1.3333333333\dots$$

Truncated, rounded

Therefore the calculated result differs from the true value by the amount truncated or rounded. (This difference is the error.)

In the computer, a 12-digit calculated result is obtained (or a 24-digit result in double-precision mode). This result is rounded and specially processed to minimize error in the displayed value.

Example in single-precision mode:  $4/3 \times 3$

$4 \left[ \frac{\div}{/} \right] 3 \left[ \times * \right] 3 \left[ \leftarrow \right] \rightarrow 4$	} Calculated in succession
$4 \left[ \frac{\div}{/} \right] 3 \left[ \leftarrow \right] \rightarrow 1.3333333333$	
$\left[ \times * \right] 3 \left[ \leftarrow \right] \rightarrow 3.999999999$	} Calculated independently

When calculated in succession, the result of  $4/3$  is obtained internally in 12 digits and is used for calculation and then rounded.

When calculated independently, the displayed value (10 digits) is used for the calculation.

Example in double-precision mode:  $4/3 \times 3$

$4 \left[ \frac{\div}{/} \right] 3 \left[ \times * \right] 3 \left[ \leftarrow \right] \rightarrow 4\#$	} Calculated in succession
$4 \left[ \frac{\div}{/} \right] 3 \left[ \leftarrow \right] \rightarrow 1.33333333333333333333\#$	
$\left[ \times * \right] 3 \left[ \leftarrow \right] \rightarrow 3.99999999999999999999\#$	} Calculated independently

When calculated in succession, the result of  $4/3$  is obtained internally in 24 digits and is used for calculation and then rounded.

When calculated independently, the displayed value (20 digits) is used for the calculation.

**Errors due to Function Determining Algorithms**

The computer uses a variety of algorithms to calculate the values of functions, such as power and trigonometric functions. When calculations use such functions, an additional source of error is introduced. This error factor increases the more functions are used in the calculation. The actual error for each function varies according to the values used and is greatest around singularities and inflection points (e.g., when an angle approaches 90 degrees, the tangent approaches infinity).

# PROGRAM OPERATION

Part 3 is devoted to the use of the BASIC programming language as implemented on the PC-E500\*. We begin with a discussion of programming concepts in general, and then move on to more specific application of those concepts to the PC-E500. Part 3 ends with some suggestions for shortcuts in programming, and for tracing bugs in your program.

PRO (program) mode and RUN mode (which was discussed in detail in PART 2) are described in this section.

\*The PC-E500 is hereafter referred to as "the computer".

This tells the computer that the contents of variable B\$ are alphabetic, or string data. To illustrate this, enter the following:

B **SHIFT** + **\$** = **SHIFT** +  
**"** B Y T E **SHIFT** + **"**  
**←**

```

R\2*PI
971179.3866
B$="BYTE"
>

```

The string BYTE is now stored in the variable B\$. To make sure of this, press the **C•CE** key and enter the following:

B **SHIFT** + **\$** **←**

```

B$
BYTE

```

The contents of character strings or character variables are displayed from the left edge of the next line.

Variables handled by the computer are divided into the following:

### Variables

Numeric variables:

Fixed numeric variables (A to Z)

Simple numeric variables (AB, C1, etc.)

Numeric array variables

String variables:

Simple string variables (A\$, B\$, C2\$, etc.)

String array variables

Numeric variables are further divided into single-precision and double-precision variables. These will be discussed later.

### Fixed Numeric Variables

The first type, fixed numeric variables, are always used by the computer for storing numerical data. They can be thought of as pre-allocated variables. In other words, no matter how much memory your program uses, you will always have at least 26 variables to choose from to store numerical data in. Fixed memory locations are eight bytes long.

### Simple Variables

Simple variable names are specified by alphanumeric characters, such as AB, B\$, C8\$. Unlike fixed variables, simple variables have no dedicated storage area in memory. The area for simple variables is automatically set aside (within the program and data area) when a simple variable is first used.

Since separate memory areas are defined for simple numeric variables and simple string variables even if they have the same name, variables such as AB, AB\$ and AB#, for example, may be used at the same time.

While alphanumeric characters are usable for simple variable names, the first character of a variable name must always be a letter. **Up to 40 characters** may be used to define a variable name.

**Notes:**

1. Variable names must not begin with a BASIC command (e.g. PRINTOUT, ONPRINT), but may contain BASIC commands if desired (e.g. APRINT, BONPRINT).
2. Each simple string variable can hold up to 254 characters or symbols.

**Array Variables**

Sometimes, it is useful to deal with numbers as an organized group, such as a list of scores or a tax table. In BASIC these groups are called arrays. Arrays can be one-dimensional, like a list, two-dimensional, like a table, or multi-dimensional up to 120 dimensions.

Use the DIM (short for dimension) statement to define an array. Arrays must always be declared before they are used (unlike the single-value variables we have been using). The form for the DIMension statement is:

**DIM** array-variable-name (size)

where:

**array-variable-name** is a variable that conforms to the normal rules for numeric or array variable names previously discussed.

**size** is the number of storage locations. Note that when you specify a number for the size, you get one more location than you specified.

Examples of legal numeric and string DIMension statements are:

```
DIM X(5)           → X (0), X (1), X (2), X (3), X (4), X (5)
DIM AA(24)
DIM QUITE5(0)
DIM X$(5)
DIM AA$(24)
DIM QUITE5$(0)
```

The first statement creates an array X with 6 storage locations. The second statement creates an array AA with 25 locations. The third statement creates an array with one location and is actually illogical since (for numbers at least) it is the same as declaring a single-value numeric variable.

It is important to know that an array-variable X and a variable X are separate and distinct to the computer. The former denotes a series of numeric storage locations, and the latter denotes a single and different location.

Now that you know how to create arrays, you might be wondering how we refer to each storage location. Since the entire group has only one name, the way in which we refer to a single location (called an “element”) is to follow the group name with a number in parentheses. This number is called a “subscript”. For example, to store the number 8 in the fifth element of our array X (declared previously) we would write:

$$X(4) = 8$$

If the use of 4 is puzzling, remember that the numbering of elements begins at zero and continues through to the number of elements declared in the DIM statement.

The real power of arrays lies in the ability to use an expression or a variable names as a subscript.

The n-dimensional array is declared by the statement:

**DIM** array-variable-name (size 1, size 2, ..., size n)

where:

**size n** specifies the number of elements in the nth dimension of the array. Note that when you specify the number of elements, you get one more element than indicated by the specification.

The following diagram illustrates the storage locations that result from the declaration DIM T(2, 3) and the subscripts (now composed of two numbers) that pertain to each location:

	column 0	column 1	column 2	column 3
row 0	T (0, 0)	T (0, 1)	T (0, 2)	T (0, 3)
row 1	T (1, 0)	T (1, 1)	T (1, 2)	T (1, 3)
row 2	T (2, 0)	T (2, 1)	T (2, 2)	T (2, 3)

**Note:**


Two-dimensional arrays can rapidly use up storage space. For example, an array with 25 rows and 35 columns uses 875 storage locations!

An n-dimensional array has n indices into the array. For example, DIM Z\$(3,3,3,4) generates a 4-dimensional array with 320 elements.

The following table shows the number of bytes used to define each variable and the number used by each program statement.



Variable type	Number of bytes used	
	Variable name	Data
Single-precision numeric variable	(Length of variable name + 4) bytes	7 bytes
Double-precision numeric variable	(Length of variable name + 4) bytes	12 bytes
String (array) variable	(Length of variable name + 8) bytes	The number of stored string (array) data (bytes)

Element	Line number	Statement & function	
Number of bytes used	3 bytes	2 bytes	1 byte

### Double-Precision Variables

Existing single-precision variables can be converted to double-precision variables by appending a sharp mark (#). For example:

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

Double-precision variables have 20 significant digits and a 2 digit exponent from -99 to 99.

#### Note:

Single-character variables to which the sharp mark is appended (e.g. A#) are not fixed numeric variables, but are treated as double-precision simple numeric variables.

The following types of variables are stored in separate memory areas:

A        and    A#  
 AB      and    AB#  
 X(10)   and    X#(10)

Variables can be specified as single-precision (10-significant-digit) variables by appending an exclamation mark (!), or as double-precision (20-significant-digit) variables by the sharp mark (#). However, the computer makes it possible to treat any numeric variable as a single- or double-precision variable by the DEFSNG (define single) and DEFDBL (define double) statements. This is especially useful if your program contains numerous double-precision variables.

## Storing Values in Double-Precision Variables

### 1. Using Declarative Signs (! and #)

AB! (or AB) = 1234567891234567891234 →  $1.234567891 \times 10^{21}$

The value is stored using 10 significant digits in the single-precision variable AB! (or AB).

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

The value is stored using 20 significant digits in the double-precision variable AB#.

### 2. Using Declarative Signs and Declarative Statements (DEFSNG and DEFDBL)

These are two BASIC commands used as variable definition statements. See Chapter 13 for a description of programming.

```
10: DEFDBL B,D
20: DIM B(2)
25: DI=123
30: DIM B!(2)
40: B(0)=9876543217876D5
50: B!(0)=B(0)
60: PRINT B(0): WAIT: PRINT B!(0),D
70: DEFSNG B,D
80: WAIT 0: PRINT B#(0): PRINT B(0), D
```

[20] B is stored in memory as a double-precision array variable (without the DEFDBL statement in line 10, line 20 should read DIM B#(2)).

[30] B! is stored in memory as a single-precision array variable.

[50] When the contents of a double-precision variable are stored under a single-precision variable, the number of significant digits is limited to 10.

If we run the program, line 60 will produce the following display:

RUN 

```
RUN
          987654321787600000
9.87654E+17
0
```

At line 80, the display will be:



```
          0
          987654321787600000
9.87654E+17      123
>
```

### Mixing Double- and Single-Precision Values

If a calculation includes double-precision variables, the computer will automatically select double-precision mode where necessary.

Double-precision mode is automatically selected in the following cases:

Calculations are executed on values using 11 or more significant digits:

Ex.  $1234567891234 \times 5$

The letter D is used in formulas to specify an exponent:

Ex.  $\text{TAN } 7.43\text{D}05$

Values are identified using the sharp mark (#):

Ex.  $4\#/7$

Double-precision variables are used:

Ex.  $\text{AB}\# + \text{BC}$

The DEFDBL statement is used:

Ex. In RUN mode, enter DEFDBL. DBL is shown on the status line of the display.

DEFDBL

DEFDBL  
>

5   $\div$  9

DEFDBL  
5/9  
  
5.5555555555555555555555556D-01

If the calculation formula contains a mixture of single- and double-precision values, each individual calculation within the formula is executed according to the degree of precision valid at that time.

#### Example:

$$\underbrace{6 \times 5}_{\text{Single-precision}} + \underbrace{4\#/7}_{\text{Double-precision}}$$

Double-precision calculation

If double-precision values are converted to single-precision variables, the double-precision value is rounded to 10 significant digits.

#### Example:

$\text{C}\cdot\text{CE}$    $\text{A B}$    $\text{SHIFT}$  +   $\#$    $=$

5   $\text{SHIFT}$  +   $\#$    $\div$  9

AB#=5#/9\_

AB#=5#/9  
  
5.5555555555555555555555556D-01

A B [=] A B [SHIFT] + [#]  
←

AB#=5#/9	5.5555555555555555555555556D-01
AB=AB#	5.5555555555555555555555556E-01

If a double-precision value is used in conjunction with a function of which the argument is single-precision, functional calculations are performed after the double-precision value is rounded off to a 10-digit value.

The following functions are performed in single-precision mode only: **DECI, HEX, POL, REC**

## Program and Data Files

Programs and data files are fundamental in the use of your computer. A wide choice of media for storing program and data files is available.

Part of the computer's internal memory can be used as RAM disk (E:), and is the most readily available storage device. External RAM cards may also be used as a RAM disk (F:), as you may use floppy disks in other computers. However, RAM cards are so compact that several can be carried in your shirt pocket. Using RAM cards, your computer becomes one of the most powerful pocket computers on the market.

SHARP's pocket disk drive (CE-140F) is another fast and convenient storage mechanism expanding the flexibility of your computer system. Standard 2.5-inch micro-floppy disks are used. Finally, a low-cost but relatively slow alternative is a cassette tape recorder.

## Filenames

When saved to a storage medium such as RAM disk E in the internal RAM, RAM disk F in the external RAM card, cassette tape, pocket disk, or serial I/O device, a file must be given a name. This name is used to load program files into computer memory, or to access data files on the medium. The filename may be any name up to 8 characters long and include the following characters:

A - Z, a - z, 0 - 9, #, \$, %, &, ', (, ), {, }, -, ^, \_, @, space

## Extension

A file extension is an additional way of identifying the type of file (e.g., BASIC program file or text file). The extension consists of three characters added to the end of the filename and separated from it by a period. The extension is specified when the file is saved.

BASIC programs are automatically given the extension .BAS when saved using the SAVE command. When reloaded into memory using the LOAD command, you do not need to specify the .BAS extension.

When the FILES or LFILES commands are used to list the files on a device, BASIC programs will appear with the .BAS extension unless some other extension has been specified by the user when the file was saved. The .BAS extension must always be specified when using the COPY command.

## Device Name

Since files can be stored on tape, disk, internal RAM, or external RAM card, the device must also be specified. The device name must be followed by a colon (:). The following are the device names used on the computer.

- X: Pocket disk drive
- E: RAM disk E (internal RAM)
- F: RAM disk F (external RAM card)
- CAS: Cassette tape
- COM: Serial I/O device

The complete file descriptor thus consists of the device name, filename, and extension:

d: filename.ext

## File Numbers

File numbers are used with certain commands (e.g., OPEN, INPUT# and PRINT#) to read data from or write data to files.

File numbers can be specified with #1 - #255. The number of file numbers which can be used simultaneously is limited as follows:

Device name	Max. number
Pocket disk drive	6
RAM disk E	} Total of 2
RAM disk F	
Cassette tape	1
Serial I/O	1

To save or load program files using the SAVE or LOAD command, the above number is decreased by 1 since one file number is used automatically for each device. Since file numbers are not used in the CSAVE or CLOAD command, program files can be saved to or loaded from tape using the CSAVE or CLOAD command even if another tape file is currently open.

All files are closed before executing the LOAD or MERGE command.

## Data Files

There are two types of data file; sequential data files and random access data files. The computer supports sequential data files only. Data is written to a sequential file as a series of ASCII characters stored one item after another (sequentially) in the order sent. The data is read back sequentially when later accessed.

## Expressions

An expression is some combination of variables, constants, and operators that can be evaluated to a single value. The calculations that you entered previously were examples of expressions. Expressions are an intrinsic part of BASIC programs. For example, an expression might be a formula that computes an answer to some equation, a test to determine the relationship between two quantities, or a means to format a set of strings.

## Numeric Operators

The computer has five numeric operators. These are the arithmetic operators that you used when exploring the use of the computer as a calculator in Chapter 6:

- + Addition
- Subtraction
- \* Multiplication
- / Division
- ^ Power

A numeric expression is constructed in the same way that you entered compound calculator operations. Numeric expressions can contain any meaningful combination of numeric constants, numeric variables, and the numeric operators:

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

## String Expressions

String expressions are similar to numeric expressions except that there is only one string operator — concatenation (+). This is the same symbol used for addition. When used with a pair of strings, the + attaches the second string to the end of the first string and makes one longer string. You should take care in making more complex string concatenations and other string operations because the work space available for string calculations is limited to 254 characters.

### Note:

String quantities and numeric quantities cannot be combined in the same expression unless one of the functions that convert a string value into a numeric value or vice versa is used:

"15" + 10 is illegal  
"15" + "10" is "1510", not "25"

## Relational Expressions

A relational expression compares two expressions and determines whether the stated relationship is true or false. The relational operators are:

> Greater than  
> = Greater than or Equal to  
= Equal to  
< > Not equal to  
< = Less than or Equal to  
< Less than

The following are valid relational expressions:

A < B  
C(1,2) > = 5  
D(3) < > 8

If A was equal to 10, B equal to 12, C(1,2) equal to 6, and D(3) equal to 9, all of these relational expressions would be true.

Character strings can also be compared in relational expressions. The two strings are compared character by character according to their ASCII value starting at the first character (see Appendix B). If one string is shorter than the other, a 0 or NULL will be used for any missing positions. All of the following relational expressions are true:

"ABCDEF" = "ABCDEF"  
"ABCDEF" < > "ABCDE"  
"ABCDEF" > "ABCDE"

Relational expressions evaluate to true or false. The computer represents true by a -1; false is represented by a 0.

## Logical Expressions

Logical operations use the Boolean algebra functions AND, OR, XOR and NOT to build connections between relational expressions. The logical operations in a single expression are evaluated after arithmetic and relational operations.

In this way, logical operators can be used to make program decisions based on multiple conditions using the IF ... THEN statement.

**Example:**

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

This statement causes execution to jump to line number 150 if the value of the numeric variable A is less than or equal to 32 and at the same time, the value of numeric variable B is greater than or equal to 90.

IF X <> 13 OR Y = 0 THEN 50

This statement causes execution to jump to line 50 unless variable X has the value 13, or if variable Y is equal to 0.

In a logical operation involving two numbers in the range -32768 to +32767, the two numbers are converted into 16-bit binary integers (in two's complement form) and the logical connection is then evaluated for each corresponding pair of bits in the two numbers.

The results returned by the logical operators for these bit evaluations are listed here:

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

After each bit pair has returned the corresponding result (a 1 or a 0) according to the above tables, the resulting 16-bit binary number is converted back to a decimal value. This number is the result of the logical operation.

**Example:**

41 AND 27 →  
equals  
9

41 = 101001  
27 = 011011 AND  
← 001001

41 OR 27 →  
equals  
59

41 = 101001  
27 = 011011 OR  
← 111011

41 XOR 27 →  
equals  
50

41 = 101001  
27 = 011011 XOR  
← 110010

NOT 3 →  
equals  
-4 (two's complement  
form)

3 = 0000000000000011 NOT  
← 1111111111111100



NOT X can generally be calculated by the equation  $\text{NOT } X = -(X+1)$ .

## Parentheses and Operator Precedence

When evaluating complex expressions, the computer follows a predefined set of priorities that determine the sequence in which operators are evaluated. This can be quite significant:

$5 + 2 * 3$  could be

$$\begin{array}{l} 5 + 2 = 7 \quad \text{or} \quad 2 * 3 = 6 \\ 7 * 3 = 21 \quad 6 + 5 = 11 \end{array}$$

The exact rules of "operator precedence" are given on page 162.

To avoid having to remember all these rules and to make your program more precise, always use parentheses to determine the sequence of evaluation. The above example is clarified by writing either:

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

# 13. PROGRAMMING

In the previous chapter, we examined some of the concepts and terms of the BASIC programming language. In this chapter, you will use these elements to create programs. Let us remind you, however, that this is not a manual on how to program in BASIC. What this chapter will do is familiarize you with the use of BASIC on your computer.

## Programs

A program consists of a set of instructions to the computer. Remember that the computer is only a machine. It will perform the exact operations that you specify. You, the programmer, are responsible for issuing the correct instructions.


## BASIC Statements

The computer interprets instructions according to a predetermined format. This format is called a statement. You must always enter BASIC statements in the same pattern. Statements must start with a line number:

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

## Line Numbers

Each line of a program must have a unique line number — any integer between 1 and 65279. Line numbers are the reference for the computer. They tell the computer the order in which to run the program, and can be used to tell the computer at which line to start. You need not enter lines in sequential order (although if you are a beginning programmer, it is probably less confusing for you to do so). The computer always begins execution with the lowest line number and moves sequentially through the lines of program in ascending order.

You can use the AUTO command to automatically insert line numbers for you. Each time you press the  key, a new line number, with the correct increment, will be automatically inserted. See the BASIC COMMAND DICTIONARY for a full description of this useful function.

It's wise to allow increments of several numbers in your line numbering (10, 20, 30, ... 10, 30, 50, etc.). This enables you to insert additional lines if necessary.

If you use the same line number, the old line with that number is deleted when you enter the new line.

## Labelled Programs

Often you will want to store several different programs in the memory at one time. (Remember that each must have unique line numbers). Normally, to start a program with a RUN or GOTO command, you need to remember the beginning line number of each program. However, there is an easier way. You can label each program with alphanumeric characters and run the program.

Label the first line of each program that you want to reference. The label consists of a letter and up to 19 alphanumeric characters with \* in front of it or in quotes, followed by a colon:

```
10: *A: PRINT "FIRST"  
20: END  
80: "B": PRINT "SECOND"  
90: END
```

Although both \*label and "label" forms may be used, \*label is recommended, since it executes more quickly and is more visible in the program listing.

## BASIC Commands

All BASIC statements must contain commands. They tell the computer what action to perform. A command is contained with a program, and as such is not acted upon immediately.

Some statements require or allow an operand:

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


Operands provide information to the computer telling it what data the command will act upon. Some commands require operands, while with other commands they are optional. Certain commands do not allow operands. (See the BASIC COMMAND DICTIONARY for BASIC commands and their uses.)

### Note:

Commands, functions and variables entered in lowercase characters will be converted to uppercase characters.

## Direct Commands

Direct commands are instructions to the computer that are entered outside of a program. They instruct the computer to perform some immediate action or set modes that affect how your programs are executed.

Direct commands have immediate effect — as soon as you complete entering direct commands (by pressing the  key), the command will be executed. Direct commands are not preceded by a line number.



```
RUN
NEW
RADIAN
```

## Modes

You will remember that you can set the computer in the CAL or RUN mode when using it as a calculator.

The RUN mode is also used to execute the programs you create. The PROgram mode is used to enter and edit your programs.

## Beginning to Program

After all your practice in using the computer as a calculator, you are probably quite at home with the keyboard. From now on, when we show an entry, we will not show every keystroke. Remember to use the  key to access characters above the keys and to end every line by pressing the  key.


Now you are ready to program!

To enter program statements into the computer, the computer must first be placed in the PRO (program) mode using the  key. The following display will appear:

```
PRO
>
```

Enter the NEW command.

```
NEW
>
```

The NEW command clears the memory of all existing programs and data. The prompt appears after you press the  key, indicating that the computer is awaiting input.

### Example 1 — Entering and Running a Program

Make sure the computer is in the PRO mode and enter the following program:

1 0 P R I N T

**SHIFT** + **"** H E L L O

**SHIFT** + **"**

```
NEW
10 PRINT "HELLO" _
```

Notice that the computer automatically inserts the colon between the number and the command when you press the **←** key.

Check that the statement is in the correct format and then change the mode to RUN by pressing the **BASIC** key.

**C-CE** R U N **←**

```
RUN
HELLO
>
```

Since this is the only line of the program, the computer will exit the program and return to the BASIC prompt ">".

### Example 2 — Editing a Program

Suppose you wanted to change the message that your program was displaying. That is, you wanted to edit your program. With a single line program you could just retype the entry, but as you develop more complex programs, editing becomes a very important component of your programming. Let's edit the program you have just written.


Are you still in the RUN mode? If so, switch back to the PRO mode.

You need to recall your program in order to edit it. Use the up arrow key **↑** to recall your program. If your program was completely executed, the **↑** key will recall the last line of the program. If there was an error in the program, or if you used the **BREAK** key to stop execution, the **↑** key will recall the line in which the error or break occurred. To make changes in your program, use the **↑** key to move up in your program (recall the previous line) and the **↓** key to move down in your program (display the next line). If held down, the **↑** or **↓** key will scroll vertically (up or down) through your program.

Remember that to move the cursor within the program line you use the **▶** (right arrow) and **◀** (left arrow) keys.

Pressing the **SHIFT** + **▶** keys will position the cursor after the rightmost character of a line.

Pressing the **SHIFT** + **◀** keys will position the cursor over the leftmost character of a line.

Using the  key, position the cursor over the first character you wish to change:



```
10:PRINT "HELLO"
```


      
  

```
10 PRINT "HELLO"
```

Notice that the cursor is now in the flashing block form, indicating that it is on top of an existing character. Enter:

GOODBYE  +   
 + 


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



Remember to press the  key at the end of the line. Change to the RUN mode.



RUN 

```
10:PRINT "GOODBYE"!
RUN
GOODBYE
Syntax error in 10
```

The error message indicates the type of error, and the line number in which the error occurred.

Press the  key to clear the error condition.

And return to the PRO mode. You must be in the PRO mode to make changes in a program. Using  (or ), recall the line in which the error occurred.


 (or )

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

The flashing cursor is positioned over the problem area. You learned, that when entering string constants in BASIC, all characters must be contained within quotation marks. Use the DELete key to eliminate the "!".



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

Now let's put the ! in the correct location. When editing programs, DELeTe and INSeRt are used in exactly the same way as they are in editing calculations. Using , position the cursor on top of the character that will be the first character following the insertion.



```
10 PRINT "GOODBYE"
```

Press the INSeRt key. A "◀" will indicate where the new data will be entered:


**INS**

```
10 PRINT "GOODBYE"
```



Enter the !. The display looks like this:

**SHIFT** + **I**

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

Remember to press the  key so the correction will be entered into the program.

#### Notes:

1. If you wish to DELeTe an entire line from your program, just enter the line number and the original line will be eliminated. The DELETE command can be used to delete more than one line at a time.
2. In the PRO mode, if keys are pressed when the cursor is not displayed, their corresponding characters are usually displayed from the leftmost column of the display. However, if the  or  key is pressed when the cursor is displayed, successive key entries are displayed starting from the cursor position.

#### Example 3 — Using Variables in Programming

If you are unfamiliar with the use of numeric and string variables in BASIC, reread the appropriate sections in Chapter 12.

Using variables in programming allows more sophisticated use of the computer's abilities.

Remember, you assign simple numeric variables using any letter from A to Z:

```
A = 5
```

To assign string variables you also use a letter, followed by a dollar sign.

```
A$ = "TOTAL"
```

The values assigned to a variable can change during the execution of a program, taking on the values entered or computed during the program. One way to assign a variable is to use the INPUT command. In the following program, the value of A\$ will change in response to the data typed in answer to the inquiry "WORD?".

Enter this program:

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

The second new element in this program is the use of the END statement to signal the completion of a program. END tells the computer that the program is completed. It is always good programming practice to use an END statement.

As your programs get more complex you may wish to review them before you begin execution. To look at your program, use the LIST command. LIST, which can only be used in the PRO mode, displays programs beginning with the lowest line number.

Try listing this program:

LIST

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

Use the  and  keys to move through your program until you have reviewed the entire program. After checking your program, change to the RUN mode and run it:

RUN

```
RUN
WORD?_
```



HELP

```
RUN
WORD?HELP_
```

```
WORD?HELP
THE WORD (HELP) HAS
4 LETTERS
>
```


This is the end of your program. Of course you may begin it again by entering RUN. However, this program would be a bit more entertaining if it presented more than one opportunity for input. We will now modify the program so it will keep running without entering RUN after each answer.



Return to the PRO mode and use the  or  key (or LIST) to reach line 50, or enter:


LIST 50 



50:END

You may enter 50 to delete the entire line or use the  key to position the cursor over the E in END. Change line 50 so that it reads:

50: GOTO 10

Now RUN the modified program.

The GOTO statement causes the program to loop (keep repeating the same operation). Since you put no limit on the loop it will keep going forever (an "infinite" loop). To stop this program press the  key.


When you have stopped a program using the  key, you can restart it using the CONT command. CONT stands for CONTINUE. With the CONT command the program will restart on the line that was being executed when the  key was pressed.

#### Example 4 — More Complex Programming

The following program computes N factorial (N!). The program begins with 1 and computes N! up to the limit that you enter. Enter this program:

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

Several new features are contained in this program. The WAIT command in 100 controls the time that displays are held before the program continues. The numbers and their factorials are displayed as they are computed. The time they appear on the display is set by the WAIT statement to approximately 2 seconds.

Notice that there are two statements in line 100 separated by a colon (:). You may put as many statements as you wish on one line (separating each by a colon) up to a maximum of 254 characters including the  key. Multiple-statement lines can make a program hard to read and modify, so it is good programming practice to use them only where the statements are very simple or there is some special reason to want the statements on one line.

In this program we have used the FOR command in line 120 and the NEXT command in line 150 to create a loop. In Example 3 you created an "infinite" loop that kept repeating the statements inside the loop until you pressed the **BREAK** key. With this FOR...NEXT loop, the computer adds 1 to N each time execution reaches the NEXT command. It then tests to see if N is larger than the limit L. If N is less than or equal to L, execution returns to the top of the loop and the statements are executed again. If N is greater than L, execution continues to line 160 and the program stops.

You may use any fixed numeric variable or single-precision simple numeric variable in a FOR...NEXT loop, you do not have to start counting at 1 and you can increment any amount at each step. See the BASIC COMMAND DICTIONARY for details.

We have labeled this program with line numbers starting with 100. Labeling programs with different line numbers allows you to have several programs in memory at one time. To RUN this program instead of the one at line 10, change to the RUN mode and enter:

```
C•CE  
R U N 1 0 0
```

You could also give the program a name using a label and start the program with RUN \*label.

Notes on the PRINT command:

If more than four lines must be displayed, the first lines will scroll up off the display, and cannot be recalled. Use the PAUSE or WAIT command in the program to display data more slowly, or use the printer. (Refer to the PAUSE, WAIT or LPRINT command.)

The WAIT command applies to every PRINT command. Break long PRINT commands into a number of shorter commands if the display scrolls too quickly.

**Example:**

```
100 PRINT A, B, ..., P  
    ↓  
100 PRINT A, B, ..., H: PRINT I, J, ..., P
```

Since the WAIT command is not supported by many personal computers, a wait loop such as FOR J=1 TO 500:NEXT J can also be used to extend the display time.

## Storing Programs in Memory

You will remember that settings and functions remain in the computer even after it is turned off. Programs also remain in memory when you turn off the computer, or it undergoes an Auto OFF. Even if you use the **BREAK**, **C•CE** or **SHIFT** + **CA** keys, the programs will remain.

Programs are lost from memory only when you:

- Enter NEW before beginning programming in the PRO mode.
- Initialize the computer using the RESET button.
- Create a new program using the SAME LINE NUMBERS as a program already in memory.

## Data Files


Following are some programming examples which use data file storage.

### Creating a Sequential File

Program 1 is a short program that creates a sequential file, DATA, from information you input on the keyboard.

#### Program 1 — Creating a Sequential File

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

Before execution, enter: INIT "E:10K"  to allocate storage space in the RAM disk E.

```
RUN
NAME: SAMUEL GOLDWYN
DEPARTMENT: AUDIO/VISUAL AIDS
DATE HIRED: 01/12/72
NAME: MARVIN HARRIS
DEPARTMENT: RESEARCH
DATE HIRED: 12/03/65
NAME: DEXTER HORTON
DEPARTMENT: ACCOUNTING
DATE HIRED: 04/27/81
NAME: DONE
```

## Reading Data from a Sequential File

Now look at Program 2. It accesses the file DATA that was created in Program 1 and displays the name of everyone hired in 1981.

### Program 2 — Accessing a Sequential File

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

RUN

```
DEXTER HORTON
Input past end in 30
```

Program 2 reads, sequentially, every item in the file, and prints the names of employees hired in 1981. When all the data has been read, line 30 causes an ERROR. To avoid this error, use the EOF function, which tests for the end-of-file. The revised program looks like this:

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

As shown by these programs, the following steps are required to create a sequential file and access the data in it:

1. OPEN the file for OUTPUT.
2. Write data to the file using the PRINT# statement.
3. CLOSE the file and reopen it in INPUT mode to read the data.
4. Use the INPUT# statement to read data from the file into the program.

### Adding Data to a Sequential File

If you have an existing data file and want to add more data to the end of it, you cannot simply open the file in the OUTPUT mode and start writing data. As soon as you open a sequential file in the OUTPUT mode, you destroy its current contents.

Instead, use the APPEND mode. If the file does not already exist, the OPEN statement will work exactly as it would if the OUTPUT mode had been specified.

The following procedure can be used to add data to an existing file called "FOLKS".

### Program 3 — Adding Data to a Sequential File

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

Input 00 in answer to the question NAME? in line 150 to cause the program to jump out of the input loop in line 160. The REM statement can be used to write programming notes to yourself.

This brief introduction to programming should serve to illustrate the exciting programming possibilities of your new computer.

### Programmable Function Keys

Programmable function keys ( **PF1** – **PF5** ) can also be used to write or run programs as well as to specify the operation mode on the Main Menu and for statistical or matrix operations.

The following commands are allocated for each PF key when the RESET button is pressed.

	<b>PF1</b>	<b>PF2</b>	<b>PF3</b>	<b>PF4</b>	<b>PF5</b>
<b>Primary</b>	RUN	LIST	CONT	SAVE	LOAD
<b>Secondary</b>	INPUT	PRINT	GOTO	GOSUB	RETURN

The assigned command key labels for the PF key will alternately appear or disappear at the bottom line of the display by pressing the **CTRL** + **↕** keys.

Pressing the **↕** key will display the primary and secondary command key labels alternately. Pressing a PF key only while the secondary command key labels are displayed will implement the secondary command. When the command key labels are not displayed, pressing a PF key will implement the primary command. Press the PF key while holding the **SHIFT** key to implement the secondary command.

## Program Execution

More than one program can be stored in this computer if the memory capacity is not exceeded. Execute the second or subsequent program using one of the following:

1. the RUN command: RUN line number
2. the GOTO command: GOTO line number

Execution begins from the specified line number. If a label such as \*AB is entered in the program, the program can be executed by entering RUN\*AB .

The following lists the differences between the variables and status when a program is executed using the GOTO and RUN commands.

Execution using RUN	Execution using GOTO
<ul style="list-style-type: none"><li>• Clears the WAIT setting.</li><li>• Clears the USING format.</li><li>• Clears array and simple variables.</li><li>• Clears double-precision operation mode and variable specifications.</li><li>• Initializes the DATA statement for the READ statement.</li><li>• Clears the PRINT=LPRINT setting.</li><li>• Closes all the files</li><li>• Clears the position set by LOCATE or GCURSOR.</li><li>• Clears the error trap function.</li><li>• Clears ERN and ERL variables.</li></ul>	<ul style="list-style-type: none"><li>• Retains the WAIT setting.</li><li>• Retains the USING format.</li><li>• Retains array and simple variables.</li><li>• Retains double-precision operation mode and variable specifications.</li><li>• Does not initialize the DATA statement for the READ statement.</li><li>• Retains the PRINT=LPRINT setting.</li><li>• Does not close all the files.</li><li>• Retains the position set by LOCATE or GCURSOR.</li><li>• Retains the error trap function.</li><li>• Retains ERN and ERL variables.</li></ul>

### Note:




When the program is executed using the RUN command, the variables for data are cleared. (Fixed variables are retained.) To retain the data, execute using the GOTO command.

Since strings such as RUN and INPUT can be assigned to the programmable function key, it is possible to assign a statement such as RUN\*AB  to a PF key. A program with the label \*AB can then easily be executed by pressing a PF key. Refer to the KEY command for PF key assignment.



# 14. DEBUGGING

After entering a new BASIC program, it often does not work the first time. Even if you are simply entering a program that you know is correct, such as those provided in this manual, it is common to make at least one typing error. It may also contain at least one logic error as well.

Following are some general hints on how to find and correct your errors. You run your program and get an error message:

1. Go back to the PRO mode and use the  or  key to recall the line with the error. The cursor will be positioned at the place in the line where the computer became confused.
2. If you cannot find an obvious syntax error, the problem may lie with the values that are being used. For example, CHR\$(A) will produce a space if A has a value of 1. Check the values of the variables in either the RUN or PRO mode by entering the name of the variable and pressing the  key.

You run the program and don't get an error message, but the program doesn't do what you expect.

1. Check through the program line by line using LIST and the  and  keys to see if you have entered the program correctly. It is surprising how many errors can be corrected by just taking another look at the program.
2. Think about each line as you go through the program as if you were the computer. Take sample values and try to apply the operation in each line to see if you get the result that you expected.
3. Insert one or more extra PRINT statements in your program to display key values and key locations. Use these to isolate the parts of the program that are working correctly and the location of the error. This approach is also useful for determining which parts of a program have been executed. You can also use STOP to temporarily halt execution at critical points so that several variables can be examined.
4. Use TRON (Trace ON) and TROFF (Trace OFF), either as direct commands or within the program to trace the flow of the program through individual lines. Stop to examine the contents of critical variables at crucial points. This is a very slow way to find a problem, but it is sometimes the only way.

No matter how careful you are, eventually you will create a program that does not do quite what you expect it to. To isolate the problem, BASIC has a special method of executing programs known as the "Trace" mode.

TRON (Trace ON) starts Trace mode. The TRON instruction may be issued as a direct command (in RUN mode) or it may be embedded within a program. Used as a direct command, TRON informs the computer that tracing is required during the execution of all subsequent programs. The programs to be traced are then started in a normal manner, with a GOTO or RUN command.

If TRON is used as a statement, it will initiate the Trace mode only when the line containing it is executed. If, for some reason, that line is never reached, Trace mode will remain inactive.

## Debugging Procedures

1. Set the computer to RUN mode.
2. Enter TRON  to specify the trace mode.
3. Enter RUN  to execute the program. The line number will be displayed at above right of the display for about 0.5 second after each line is executed.
4. Press the **BREAK** key when the desired line number is displayed. The break message is displayed and execution is interrupted. Press the  key to display the last statement executed. To resume execution, press the **SHIFT** +  keys or enter CONT . However, if execution is interrupted during data entry using the INPUT command, just press the  key as for usual program continuation.
5. Press the  key to move to the line to be checked. Holding the  key will execute the program step by step. Releasing the key will stop program execution.
6. Continue the trace procedure and check if the program is executing properly by confirming program execution order and variable contents after each line is executed. If the program is not executing properly, correct the logic.
7. After debugging, enter TROFF  to exit the trace mode.

### Example:

```

10 INPUT "A=";A,"B=";B
20 C=A*2
30 D=B*3
40 PRINT "C=";C;" D=";D
50 END

```

### Run the program.

RUN mode			
TRON <input type="text"/>	>		
RUN <input type="text"/>	A=		} Execute INPUT command
8 <input type="text"/> (Data entry)	B=		
9 <input type="text"/> (Data entry)	C=16	D=27	} Execute PRINT command End of execution (prompt is displayed)
	>		



The executed line number will be displayed at the right top for about 0.5 second.

When execution is interrupted with the **BREAK** key, recall the variables manually and check that the values are as expected. Pressing the **↓** key will execute one statement at a time and entering **CONT** **←** will execute the statements continuously.

**Note:**

The trace mode will remain in effect unless **TROFF** **←** is entered, the **SHIFT** + **CA** keys are pressed, or the power is turned off.

To debug by interrupting program execution:

Perform one of the following:

- Press the **BREAK** key during program execution.
- Press the **BREAK** key in the trace mode.
- Enter the **STOP** command at the location to be stopped.

The Break message will be displayed and execution will be interrupted.

Then

1. Check the variable contents manually.
  2. Press the **↓** key to execute subsequent statements line by line.  
Press the **SHIFT** + **↓** keys or enter **CONT** **←** to return to previous operation.
- A program interrupted by the **BREAK** key or the **STOP** command can be executed line by line by pressing the **↓** key. Trace messages will be displayed at each step.



# BASIC REFERENCE

Part 4 contains alphabetical listings of all the BASIC commands supported by the PC-E500\* and can be used as a ready reference.

The first section contains an alphabetical listing of numeric functions and pseudovariables.

The second section is an alphabetical listing of all other BASIC commands.

\* The PC-E500 is hereafter referred to as "the computer".

## 15. SCIENTIFIC & MATHEMATICAL CALCULATIONS

The computer has a wide range of built-in functions for scientific, mathematical and statistical calculations. They are listed below alphabetically. All the functions listed below can be used as part of calculations when using the computer in RUN mode. They may also be used as BASIC commands within programs.

For trigonometric functions, entries can be made in degrees, radians or as a gradient value, as appropriate:

**DEGREE:** Set the computer to degree entry mode by typing DEGREE (the status line on the display shows DEG), this is the default mode.

**RADIAN:** Set the computer to radian entry mode by typing RADIAN (the status line on the display shows RAD).

**GRADIENT:** Set the computer to gradient entry mode by typing in GRAD (the status line on the display shows GRAD).

These three modes (DEG, RAD, and GRAD) can also be set from within a program. Once one mode is set, all entries for trigonometric functions must be in the units set (degrees, radians, or gradient values) until the mode is changed either manually or from within a program. The mode setting is preserved even when the power is turned off. The examples given below are all for direct entry of the functions entered in degrees.

Functions marked **D** can be used in both double-precision and single-precision modes. Functions marked **S** can be used only in single-precision mode, where double-precision arguments are first converted to a single-precision value and then used in calculations.

Most functions can also be implemented by pressing the corresponding function key. Functions marked (\*) have no corresponding key and must be entered through the keyboard.

**\*ABS** $|x|$ **Function:** Absolute value**D****Remarks:** Returns the absolute value of the numeric argument. The absolute value is the magnitude of the number irrespective of its sign. ABS-10 is 10.**Example:** ABS -10 

10

**ACS** $\cos^{-1}x$ **Function:** Inverse or arc cosine**D****Remarks:** Returns the arc cosine of the numeric argument. The arc cosine is the angle whose cosine is equal to the argument. The value returned depends on the mode (DEG, RAD or GRAD).**Example:** DEGREE   
ACS -0.5 

120

**AHC** $\cosh^{-1}x$ **Function:** Inverse hyperbolic cosine**D****Remarks:** Returns the inverse hyperbolic cosine of the numeric argument.**Example:** AHC 10 

2.993222846

**AHS** $\sinh^{-1}x$ **Function:** Inverse hyperbolic sine**D****Remarks:** Returns the inverse hyperbolic sine of the numeric argument.**Example:** AHS 27.3 

4.000369154

**AHT** $\tanh^{-1}x$ **Function:** Inverse hyperbolic tangent**D****Remarks:** Returns the inverse hyperbolic tangent of the numeric argument.**Example:** AHT 0.7 

8.673005277E-01

<b>ASN</b>	$\sin^{-1}x$
------------	--------------

**Function:** Inverse or arc sine **D**

**Remarks:** Returns the arc sine of the numeric argument. The arc sine is the angle whose sine is equal to the argument. The value returned depends on the mode (DEG, RAD or GRAD).

**Example:** DEGREE   
ASN 0.5  30

<b>ATN</b>	$\tan^{-1}x$
------------	--------------

**Function:** Inverse or arc tangent **D**

**Remarks:** Returns the arc tangent of the numeric argument. The value returned depends on the mode (DEG, RAD or GRAD).

**Example:** DEGREE   
ATN 1  45

<b>COS</b>	<b>cosx</b>
------------	-------------

**Function:** Cosine **D**

**Remarks:** Returns the cosine of the angle argument. The value returned depends on the mode (DEG, RAD or GRAD).

**Example:** DEGREE   
COS 120  -0.5

<b>*CUB</b>	$x^3$
-------------	-------

**Function:** Cube **D**

**Remarks:** Returns the cube of the argument.

**Example:** CUB 3  27

<b>CUR</b>	$\sqrt[3]{x}$
------------	---------------

**Function:** Cube root **D**

**Remarks:** Returns the cube root of the argument.

**Example:** CUR 125  5

## DECI

**Function:** Hexadecimal to decimal conversion

**S**

**Remarks:** Converts a hexadecimal value to its decimal equivalent.

**Example:** DECI F82

3970

## DEG

$dd^{\circ}mm'ss'' \rightarrow ddd.dddd^{\circ}$

**Function:** Deg/min/sec to decimal conversion

**D**

**Remarks:** Converts an angle argument in DMS (Degrees, Minutes, Seconds) format to DEG (Decimal Degrees) format. In DMS format the integer portion of the number represents degrees, the first and second digits after the decimal point represent minutes, the third and fourth digits after the decimal point represent seconds, and any further digits represent fractional seconds.

**Example:** DEG 30.5230  (30°52'30")

30.875

## DMS

$ddd.dddd^{\circ} \rightarrow dd^{\circ}mm'ss''$

**Function:** Decimal to deg/min/sec conversion

**D**

**Remarks:** Converts an angle argument in DEG format to DMS format (see DEG).

**Example:** DMS 124.8055

124.48198 (124°48'19"8)

## EXP

$e^x$

**Function:** Exponential function

**D**

**Remarks:** Returns the value of e (2.718281828... the base of natural logarithms) raised to the value of the numeric argument.  
The corresponding function key is .

**Example:** EXP 1.2

3.320116923

## FACT

$n!$

**Function:** Factorial n

**D**

**Remarks:** Returns the factorial of the argument.

**Example:** FACT 7

5040

## HCS

cosh x

**Function:** Hyperbolic cosine

D

**Remarks:** Returns the hyperbolic cosine of the numeric argument.

**Example:** HCS 3 

10.067662

## HEX

**Function:** Decimal to hexadecimal conversion

S

**Remarks:** Converts a decimal value to its hexadecimal equivalent.

**Example:** HEX 7820 

1E8C

## HSN

sinh x

**Function:** Hyperbolic sine

D

**Remarks:** Returns the hyperbolic sine of the numeric argument.

**Example:** HSN 4 

27.2899172

## HTN

tanh x

**Function:** Hyperbolic tangent

D

**Remarks:** Returns the hyperbolic tangent of the numeric argument.

**Example:** HTN 0.9 

7.162978702E-01

## \*INT

**Function:** Integer

D

**Remarks:** Returns the integer portion of the argument. The integer portion of PI is 3.

**Example:** INT -1.9 

-2



**LN** $\log_e x$ **Function:** Natural or Napierian logarithm**D****Remarks:** Returns the logarithm to the base e (2.718281828 ...) of the numeric argument.**Example:** LN 2 

6.931471806E-01

**LOG** $\log_{10} x$ **Function:** Common logarithm**D****Remarks:** Returns the logarithm to the base 10 of the numeric argument.**Example:** LOG 1000 

3

**\*NCR** ${}_n C_r = n! / r!(n-r)!$ **Function:** Combination**D****Remarks:** Enter the values as NCR(n,r).**Example:** NCR (6,3) 

20

**\*NPR** ${}_n P_r = n! / (n-r)!$ **Function:** Permutation**D****Remarks:** Enter the values as NPR(n,r).**Example:** NPR (6,3) 

120

**PI** $\pi$ **Function:** PI**D****Remarks:** PI is a numeric pseudovalue that has the value of  $\pi$ . Use of PI is identical to use of the  key. The value of PI has 10-digit accuracy in single-precision (DEFSNG) mode, and 20-digit accuracy in double-precision (DEFDBL) mode.**Example:** DEFDBL   
PI 

3.1415926535897932385#

**POL** $(x,y) \rightarrow (r,\theta)$ **Function:** Rectangular to Polar coordinate conversion**S****Remarks:** Converts numeric arguments of rectangular coordinates to their polar coordinate equivalents.

The first argument indicates the distance from the y-axis and the second the distance from the x-axis. The values converted indicate the distance from the origin and the angle in the polar coordinates, and are assigned to the fixed variables Y and Z respectively. The angle depends on the mode (DEG, RAD, or GRAD).

**Example:** DEGREE

POL (8,6)

Z

10 (r = 10)  
36.86989765  
( $\theta \approx 36.9^\circ$ )

**^** $y^x$ **Function:**  $x^{\text{th}}$  power**D****Remarks:** Returns the  $x^{\text{th}}$  power of the numeric argument. Enter as  $y \wedge x$ .**Example:**  $4 \wedge 2.5$  

32

**RCP** $1/x$ **Function:** Reciprocal**D****Remarks:** Returns the reciprocal of the numeric argument.**Function:** RCP 4 

0.25

**REC** $(r,\theta) \rightarrow (x,y)$ **Function:** Polar to rectangular coordinate conversion**S****Remarks:** Converts numeric arguments of polar coordinates to their rectangular coordinate equivalents.

The first argument indicates the distance from the origin and the second argument the angle. The angle depends on the mode (DEG, RAD or GRAD). The converted values indicate the distances from the y-axis and the x-axis, and are assigned to the fixed variables Y and Z, respectively.

**Example:** DEGREE

REC (12,30)

Z

10.39230485 ( $x \approx 10.4$ )  
6 (y = 6)

## \*RND

**Function:** Random number

**D**

**Remarks:** See RND and RANDOMIZE in the BASIC COMMAND DICTIONARY.

## ROT

$x^{\sqrt{y}}$

**Function:**  $x^{\text{th}}$  root

**D**

**Remarks:** Returns the  $x^{\text{th}}$  root of the argument  $y$ . Enter as  $y\text{ROT}x$ .

**Example:** 7776 ROT 5

6

## \*SGN

**Function:** Sign of argument

**D**

**Remarks:** Returns a value based on the sign of the argument.

If  $x > 0$ , the function returns 1.

If  $x < 0$ , the function returns  $-1$ .

If  $x = 0$ , the function returns 0.

## SIN

$\sin x$

**Function:** Sine

**D**

**Remarks:** Returns the sine of the angle argument. The value returned depends on the mode (DEG, RAD or GRAD).

**Example:** DEGREE

SIN 30

0.5

## SQR

$\sqrt{x}$

**Function:** Square root

**D**

**Remarks:** Returns the square root of the argument.

**Example:** SQR 3

1.732050808

<b>SQU</b>	$x^2$
------------	-------

**Function:** Square **D**

**Remarks:** Returns the square of the argument.

**Example:** SQU 4  16

<b>TAN</b>	$\tan x$
------------	----------

**Function:** Tangent **D**

**Remarks:** Returns the tangent of the angle argument. The value returned depends on the mode (DEG, RAD or GRAD).

**Example:** DEGREE   
TAN 45  1

<b>TEN</b>	$10^x$
------------	--------

**Function:** Antilogarithm **D**

**Remarks:** Returns the value of 10 (the base of the common log) raised to the value of the numeric argument.

**Example:** TEN 3  1000

# Calculation Ranges

## Numerical Calculations:

For a calculation involving  $x$ , the number  $x$  must be within one of the ranges below:

$$-1 \times 10^{100} < x \leq -1 \times 10^{-99} \text{ for negative } x$$

$$10^{-99} \leq x < 10^{100} \text{ for positive } x$$

$$x = 0$$

The value of  $x$  displayed is limited by the number of digits which fit on the display screen.

## Functions:

Function	Range of $x$
sin $x$ cos $x$ tan $x$	DEG: $ x  < 1 \times 10^{10}$ Single-precision $ x  < 1 \times 10^{20}$ Double-precision RAD: $ x  < \frac{\pi}{180} \times 10^{10}$ Single-precision $ x  < \frac{\pi}{180} \times 10^{20}$ Double-precision GRAD: $ x  < \frac{10}{9} \times 10^{10}$ Single-precision $ x  < \frac{10}{9} \times 10^{20}$ Double-precision Also, for tan $x$ only: ( $n = \text{integer}$ ) DEG: $ x  \neq 90(2n-1)$ RAD: $ x  \neq \frac{\pi}{2}(2n-1)$ GRAD: $ x  \neq 100(2n-1)$
$\sin^{-1} x$ $\cos^{-1} x$	$-1 \leq x \leq 1$
$\tan^{-1} x$	$ x  < 1 \times 10^{100}$
sinh $x$ cosh $x$ tanh $x$	$-227.9559242 \leq x \leq 230.2585092$ Single-precision $-227.95592420641052271 \leq x \leq 230.25850929940456840$ Double-precision
$\sinh^{-1} x$	$ x  < 1 \times 10^{50}$
$\cosh^{-1} x$	$1 \leq x < 1 \times 10^{50}$
$\tanh^{-1} x$	$ x  < 1$
ln $x$ log $x$	$1 \times 10^{-99} \leq x < 1 \times 10^{100}$
$e^x$	$-1 \times 10^{100} < x \leq 230.2585092$ Single-precision $-1 \times 10^{100} < x \leq 230.25850929940456840$ Double-precision
$10^x$	$-1 \times 10^{100} < x < 100$
$\sqrt[x]{x}$	$ x  < 1 \times 10^{100}$
$\frac{1}{x}$	$ x  < 1 \times 10^{100}, x \neq 0$
$x^2$	$ x  < 1 \times 10^{50}$
$\sqrt{x}$	$0 \leq x < 1 \times 10^{100}$
$n!$	$0 \leq n \leq 69$ ( $n = \text{integer}$ )
DMS $\rightarrow$ DEG DEG $\rightarrow$ DMS	$ x  < 1 \times 10^{100}$

Function	Range of x	
$y^x$ $(y^x = 10^{x \cdot \log y})$	when $y > 0$ , $-1 \times 10^{100} < x \log y < 100$ when $y = 0$ , $x > 0$ when $y < 0$ , $\left\{ \begin{array}{l} x = \text{integer or } \frac{1}{x} = \text{odd integer (} x \neq 0) \\ \text{and } -1 \times 10^{100} < x \log  y  < 100 \end{array} \right.$	
$\sqrt[x]{y}$ $(\sqrt[x]{y} = 10^{\frac{1}{x} \log y})$	when $y > 0$ , $-1 \times 10^{100} < \frac{1}{x} \log y < 100$ , $x \neq 0$ when $y = 0$ , $x > 0$ when $y < 0$ , $\left\{ \begin{array}{l} x \text{ or } \frac{1}{x} \text{ must be non-zero integer,} \\ \text{and } -1 \times 10^{100} < \frac{1}{x} \log  y  < 100 \end{array} \right.$	
DECI → HEX	$ x  \leq 9999999999$ , $x = \text{integer}$	
HEX → DECI	$0 \leq x \leq 2540BE3FF$ ( $x$ in hexadecimal) $FDABF41C01 \leq x \leq FFFFFFFF$	
$x, y \rightarrow r, \theta$	$(x^2 + y^2) < 1 \times 10^{100}$ $\frac{y}{x} < 1 \times 10^{100}$	$\left\{ \begin{array}{l} r = \sqrt{x^2 + y^2} \\ \theta = \tan^{-1} \frac{y}{x} \end{array} \right.$
$r, \theta \rightarrow x, y$	$r < 1 \times 10^{100}$ $ r \sin \theta  < 1 \times 10^{100}$ $ r \cos \theta  < 1 \times 10^{100}$	$x = r \cos \theta$ $y = r \sin \theta$
nPr	$0 \leq r \leq n < 10^{100}$ $n, r$ integers	
nCr	$0 \leq r \leq n < 10^{100}$ $n, r$ integers when $n - r < r$ , $n - r \leq 69$ when $n - r \geq r$ , $r \leq 69$	

- The following functions are single-precision functions only. Before using them, double-precision values are converted to single-precision values:  
DECI, HEX, POL, REC

# 16. BASIC COMMAND DICTIONARY

The following pages contain an alphabetic listing of the BASIC commands that you can use on the computer.

For simplicity, the following conventions have been adopted in compiling this dictionary:

- expression** Indicates a numeric value, numerical variable or a formula including numeric values and numerical variables.
- variable** Indicates a numerical variable or string variable including array variables.
- “string”** Indicates a character string enclosed in quotation marks.
- string variable** Indicates a string variable or string array variable.
- \*label** Indicates \*label.  
(Although both \*label and “label” forms may be used with this computer, \*label is recommended. \*AB and “AB” are different labels.)
- d:** Indicates a device name.  
The following are device names used on the computer:  
X: Pocket disk drive  
E: RAM disk E (internal RAM)  
F: RAM disk F (external RAM card)  
CAS: Cassette tape  
COM: Serial I/O device
- [ ]** The parameter in square brackets is optional. The brackets themselves are not part of the command entry.
- ( )** Used to enclose parameter values in certain commands. They should be entered as part of the command.
- “ ”** Used to enclose string parameter values in certain commands.
- {A}**  
**{B}** A or B can be selected.
- P** Program execution is possible.
- D** Direct input operation is possible.

**Abbreviation**

Most of the commands can be abbreviated.  
The shortest abbreviation allowed is given in this manual.  
Example:

Abbreviation: P. (for PRINT)

The following abbreviations are also valid:

PR. PRI. PRIN.



---

# AER

---

P  
D

**FORMAT:** AER number [(expression, expression, ...)]

**Abbreviation:** AE.

**See Also:**

---

**PURPOSE:**

Executes the algebraic expression(s) that were stored in the AER mode.

**REMARKS:**

AER transfers the algebraic expressions stored in AER mode into a BASIC program.

Specify the number designating the algebraic expression, and the computer will recall the expression and run it. An error occurs if an undesignated number is specified.

The number of parameters specified in the command depends on the stored expression.

An error occurs if the number of parameters does not match the number of variables used in the stored expression.

If an expression,  $F(X)=\sin X + \cos X$ , is designated by the number 2 in AER mode, the statement `AER 2(90)` will make the computer execute the expression  $\sin 90 + \cos 90$ .

If a constant is designated by the number 3, the statement `Z=AER 3` will cause the computer to read the constant and assign it to variable Z. In this case no parameter need be specified.

If an expression,  $F(X,Y)=X * \log Y$ , is designated by number 5, the statement `Z=AER 5(10,100) + 3` causes the computer to execute  $10 * \log 100+3$  and assign 23 to Z.

Defined variables may be used as parameters.

*Example:* `A=10:AAZ=100:Z=AER 5(A,AAZ)+3`

**Notes:**

1. If an expression stored in AER mode is deleted, the number given in the AER statement may have to be changed.
2. Only single-precision numeric variables can be used in the expressions stored in AER mode. String variables or double-precision numeric variables cannot be used.
3. When using single-precision numeric variables other than fixed variables in the expression to be stored in the AER mode, for example, when  $F(AA)=\sin AA+8$  is designated by number 4, store an arbitrary value to AA (e.g. `AA=0:B=AER 4(2)`) and then execute the AER statement.

---

# ARUN

---

P

**FORMAT:** ARUN  $\left[ \begin{array}{l} \{ \text{line number} \} \\ \{ * \text{label} \} \end{array} \right]$

**Abbreviation:** AR.

**See Also:** AUTOGOTO, RUN

---

**PURPOSE:**

Sets the computer to start a program automatically when turned on and RUN mode is specified.

**REMARKS:**

Include ARUN as the first program statement (in the first line of the program) to start the program as soon as the computer is turned on. This has the same effect as if a RUN command had been entered from the keyboard.

(If the power is turned off in the RUN, PRO or AER mode, the computer will be in the RUN mode when the power is turned on again.)

Pressing the **BASIC** key in the Main Menu, CAL or AER mode to specify the RUN mode will execute the program automatically.

\*label must be the first statement of a line within the program, and must consist of alphanumeric characters or symbols.

Turning the power on after it has been turned off by the Auto OFF function will not execute the program automatically.

ARUN is similar to AUTOGOTO except that all variables and arrays other than fixed variables are cleared before program execution.

**EXAMPLE:**

```
5: ARUN
10: CLS:WAIT 100
20: PRINT "WELCOME TO THE WORLD OF"
30: PRINT "THE COMPUTER"
40: PRINT "YOU HAVE";FRE0;" BYTES FREE"
50: END
```

The program runs automatically when the power is turned on.

---

# ASC

---

P  
D

**FORMAT:** ASC { "string"  
                  string variable }

**Abbreviation:** A.

**See Also:** CHR\$

---

**PURPOSE:**

Returns the character code for the first character in the specified string.

**REMARKS:**

Specify the string as the contents of a string variable in the form X\$ or as an actual string enclosed in quotes, "XXXX". Only the character code of the first character in the string is returned. See Appendix B for character code tables.

**EXAMPLE:**

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

[10] The user presses a key to enter a character.

[20] ASC finds the code number for the character.

[30] The answer is displayed.

[40] Repeats until the user halts the program by pressing the **BREAK** key.

---

# AUTO

---

D

**FORMAT:** AUTO [[starting line number][,increment]] 

**Abbreviation:**

**See Also:**

---

**PURPOSE:**

Provides automatic insertion of program line numbers in the PRO mode.


**REMARKS:**



Valid only as direct input in the PRO mode.

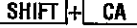
Starting line number and incremental value may be specified.

If not specified, the computer automatically sets the first line number to 10 and the increment to 10. However, if the AUTO command has been previously set to other values, those values are used.

An error is generated if the starting line number exceeds 65279.

When the mode is changed from PRO to RUN and then back to PRO mode, entering AUTO  assumes the previously set increment and resumes line numbering from the most recently generated line number.

Pressing the  or  key while a line number is displayed will have the same effect.

Pressing the  keys, turning the power off and then on, or entering an operation mode other than PRO or RUN will exit the AUTO mode.

**EXAMPLE:**

AUTO 10 Starting line number is 10, followed by 20, 30, ...

AUTO 200,20 Starting line number is 200, followed by 220, 240, ...

---

# AUTOGOTO

---

P

**FORMAT:** AUTOGOTO {line number}  
                          {\*label}

**Abbreviation:** AU.

**See Also:** ARUN, GOTO

---

**PURPOSE:**

Starts a program automatically when the computer is turned on and in the RUN mode.

**REMARKS:**

Include AUTOGOTO in the first line of a program (the first statement) to start the program when the computer is turned on and is in the RUN mode. This functions as if a GOTO command had been entered from the keyboard. AUTOGOTO is similar to ARUN but does not clear all variables and arrays before starting the program. (If the power is turned off in the RUN, PRO or AER mode, the computer will be in the RUN mode when the power is turned on again.)

Pressing the **BASIC** key in the Main Menu, CAL or AER mode to specify the RUN mode will execute the program automatically.

Turning the power on after it has been turned off by the Auto OFF function will not execute the program automatically.

\*label must be the first statement of a line within the program, and must consist of alphanumeric characters or symbols.

**EXAMPLE:**

```
10: AUTOGOTO 60
:
:
:
:
```

The program starts at line 60 when the computer is turned on.

---

# BASIC

---

D

**FORMAT:** BASIC 

**Abbreviation:** BA.

**See Also:** TEXT

---

**PURPOSE:**


Clears the text mode.

**REMARKS:**

Valid only as direct input in the PRO mode.

Executing this command clears the Text mode and returns the mode to BASIC. As the mode returns to BASIC, the prompt symbol changes from "<" to ">". Changing from the Text mode to the BASIC mode usually changes the text held in the computer memory to a program (internal code).

All lowercase letters other than those in character strings enclosed in quotation marks (" ") or following a REM ( ' ) statement are automatically converted to uppercase letters.

Abbreviations such as "P." and "I." are not converted to their respective commands. (To do so, move the cursor to the line and press the  key.) Because of the characteristics of the BASIC function, commands and formats included in the text but not found in the computer will not be executed.

During program conversion, "✖" is displayed at the bottom right of the display. Approximately 600 bytes are required for work area to convert a program.

If a converted line is too long, an error will occur.

If a password has been set, executing the BASIC command results in an error.

---

# BEEP

---

P  
D

**FORMAT:** BEEP number [ ,[tone][,duration] ]

**Abbreviation:** B.

**See Also:**

---

**PURPOSE:**

Generates beeps of the specified tone and duration through the computer's internal speaker.

**REMARKS:**

Number specifies the number of times the beep will sound. Specify a positive value or expression up to 65535.

Tone specifies the frequency of the beep in the range of 255 to 0. As the value of the tone parameter is increased, the frequency is reduced.

Duration specifies the duration of the beep in the range of 0 to 65535. The beep duration setting varies with the tone parameter. A given duration value will give a relatively longer beep at lower frequencies.

If the duration is omitted, a default value of 583 is set.

If the tone is omitted, a default value of 12 is set.

If the duration and tone are omitted, the frequency of the beep is set to approximately 4 KHz.

Press the **BREAK** key to stop a beep.

The chart below gives details on combinations of parameters:

Note	No.	Ideal Frequency	Actual Frequency	Tolerance (%)	Tone	Note	No.	Ideal Frequency	Actual Frequency	Tolerance (%)	Tone
do	3	261.6	261.8	0.08	222	do	27	1046.5	1040.7	-0.56	39
do#	4	277.2	277.7	0.18	208	do#	28	1108.7	1113.0	0.39	35
re	5	293.7	294.3	0.20	195	re	29	1174.7	1174.3	-0.03	32
re#	6	311.1	311.4	0.10	183	re#	30	1244.5	1242.7	-0.14	29
mi	7	329.6	329.0	-0.18	172	mi	31	1318.5	1319.6	0.08	26
fa	8	349.2	348.8	-0.11	161	fa	32	1396.9	1406.6	0.69	23
fa#	9	370.0	371.0	0.27	150	fa#	33	1480.0	1471.2	-0.60	21
so	10	392.0	391.4	-0.20	141	so	34	1568.0	1580.2	0.77	18
so#	11	415.3	414.2	-0.27	132	so#	35	1661.2	1662.3	0.07	16
la	12	440.0	439.9	-0.02	123	la	36	1760.0	1753.4	-0.38	14
la#	13	466.2	465.5	-0.15	115	la#	37	1864.7	1855.1	-0.52	12
ti	14	493.9	494.2	0.06	107	ti	38	1975.5	1969.2	-0.32	10
do	15	523.3	522.4	-0.17	100	do	39	2093.0	2098.4	0.26	8
do#	16	554.4	554.1	-0.05	93						
re	17	587.3	589.9	0.44	86						
re#	18	622.3	624.4	0.34	80						
mi	19	659.3	656.4	-0.44	75						
fa	20	698.5	699.4	0.13	69						
fa#	21	740.0	739.9	-0.01	64						
so	22	784.0	785.3	0.17	59						
so#	23	830.6	825.8	-0.58	55						
la	24	880.0	882.8	0.32	50						
la#	25	932.3	934.3	0.21	46						
ti	26	987.8	992.2	0.44	42						

MIN. 230.6 Hz (255)

MAX. 2844.4 Hz (0)

frequency (Hz) =  $256000/(90+4n)$

beep duration ( $\mu$ s) = specified duration/(frequency  $\times 10^6$ )

where n: tone parameter

**EXAMPLE:**

```

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

```

[10] The outer loop changes the number of beeps from 1 to 3.

[20] The inner loop changes the tone.

[30] The BEEP statement is executed 15 times.



---

# CHAIN

---

P

**FORMAT:** 1. CHAIN  
2. CHAIN "d: filename"[, {line number  
                                  \*label}]  
          d: X, E, F, CAS, COM

**Abbreviation:** CHA.

**See Also:** LOAD, MERGE

---

**PURPOSE:**

Loads and starts, from within one program, another program that has been stored on the specified device.

**REMARKS:**

To use CHAIN, the specified program must be present on the specified device. The currently running program is cleared from memory at the point where a CHAIN command is encountered and the specified program is loaded and started. Entering the CHAIN statement at the end of each program will continue to load programs automatically. An error will occur if the program becomes too large for the program area as a result of programs loaded by the CHAIN command.

Format 1 is for use only with tape and loads the first stored program from tape and starts it from the lowest line number in the program. The effect is the same as having entered CLOAD (or LOAD) and RUN in the RUN mode.

Format 2 searches the specified device for the program indicated by "filename", loads it, and starts it at the specified line number or \*label, or the lowest line number if omitted.

A program stored to tape with the CSAVE command cannot be loaded with the CHAIN command. Store the program with the SAVE command.

**Note:**

A file is automatically opened when executing the CHAIN command. If the device name is specified as CAS or COM, files for its device must be closed.

**EXAMPLE:**

CHAIN "X:PRO1", 100

Searches the disk for a program named PRO1, loads it and begins execution with line number 100.

---

# CHR\$

---

P  
D

**FORMAT:** CHR\$ expression

**Abbreviation:** CH.

**See Also:** ASC

---

**PURPOSE:**

Returns the character that corresponds to the numeric character code of the parameter.

**REMARKS:**

See Appendix B for a chart of character codes and their relationship to characters, e.g., CHR\$ 65 is "A".

A hexadecimal number can be specified with "&H" in front of the character code (eg. A\$ = CHR\$ &H5A)

A value greater than 255 generates an error.

**EXAMPLE:**

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

Displays the characters represented by the codes entered in line 20.

---

# CIRCLE

---

P  
D

**FORMAT:** CIRCLE (expression 1, expression 2), expression 3  
[,expression 4][,expression 5][,expression 6]  
[,expression 7][,expression 8][,expression 9]

**Abbreviation:** Cl.

**See Also:** PAINT, COLOR, GRAPH

---

**PURPOSE:**

Draws a circle.

**REMARKS:**

This command is effective only in the Graphics mode and is used to draw a circle, arc, sector, or ellipse with a solid line.

Expressions 1 and 2 are used to specify coordinates X and Y, respectively, at the center of a circle. The values of expressions 1 and 2 must be within the range of -999 to 999.

Expression 3 is used to specify the radius of a circle. The value of expression 3 must be within the range of 1 to 499 ( $1 \leq \text{expression 3} \leq 499$ ).

Expression 4 is used to specify the color of the line. The value of expression 4 may be specified within the range of 0 to 3. (See COLOR for colors specified by the respective values.) If expression 4 is omitted, the previous value (i.e., the color previously specified) is assumed.

Expressions 5 and 6 are used to specify the starting angle and ending angle, respectively, of an arc or sector. The respective values of expressions 5 and 6 must be within the range of -2047 to 2047. If the value is 0, the positive x-axis is used as the starting angle. If a positive value is given, the counterclockwise direction is specified. If a negative value is given, the clockwise direction is specified. The default value of expression 5 is 0 degrees, and that of expression 6 is 360 degrees.

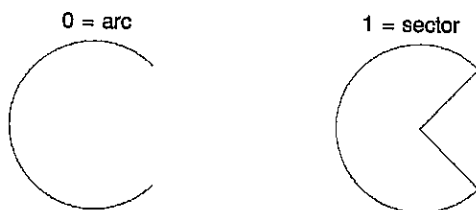
Expression 7 is used to specify the following ratio:

$$\text{Ratio} = \frac{r_y (\text{radius in Y-axis direction})}{r_x (\text{radius in X-axis direction})}$$

If the value of expression 7 is 1, a circle is drawn. If the given value is other than 1, an ellipse is drawn. The default value of expression 7 is 1.

Expression 8 is used to specify a pitch angle. The printer starts drawing a circle (arc or sector) or ellipse by dividing it in units of pitch angle from the starting angle to the ending angle. The value of expression 8 must be within the range of 1 to 2047. The default value of expression 8 is 1.

Expression 9 is used to specify an arc or a sector. If the value of expression 9 is 0, an arc is drawn. If the value given is 1, a sector is drawn. The default value of expression 9 is 0.



Note:

The respective values of expressions 5, 6, and 8 are given units of degrees irrespective of the specified angular mode.

**EXAMPLES:**

```
5: OPEN
10: GRAPH
20: CIRCLE (240, -100),100,0,0,360,1/2,10,0
30: LTEXT
40: LPRINT
50: END
```

Ratio = 0.5

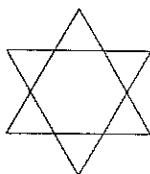
These two commands return the printer to the Text mode and move the print head back to its leftmost position.



```
5: OPEN
10: GRAPH
20: CIRCLE (240,-100),100,0,90,450,1,120,0
30: CIRCLE (240,-100),100,3,-90,270,1,120,0
40: LTEXT
50: LPRINT
60: END
```

Pitch angle = 120°

Pitch angle = 120°



The figure is actually printed in color.

---

# CLEAR

---

P  
D

**FORMAT:** CLEAR [variable 1, variable 2, ..., variable n]

**Abbreviation:** CL.

**See Also:** DIM, ERASE

---

**PURPOSE:**

Erases variables that have been used in the program and resets all preallocated variables to zero or null.

**REMARKS:**

CLEAR recovers memory space used to store simple numeric variables, and array variables secured using the DIM statement. Also use CLEAR at the beginning of a program to clear space occupied by variables from previously run programs if several programs are in memory.

Do not use the CLEAR command within a FOR...NEXT loop.

Use the ERASE command to clear specific array variables.

**EXAMPLE:**

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

[20] Frees up the spaces assigned to C( ) and resets A to zero.

---

# CLOAD

D

---

**FORMAT:** 1. CLOAD "filename"   
2. CLOAD 

**Abbreviation:** CLO.

**See Also:** CLOAD?, CSAVE

---

**PURPOSE:**

Loads a program saved on tape.

**REMARKS:**

Valid only as direct input in the PRO or RUN mode.

Format 1 clears the memory of an existing program, searches the tape for the program indicated by "filename", and loads the program.

Format 2 clears the memory and loads the first program stored on tape, starting at the current position.

During execution, "\*" is displayed at the bottom right after the file name is displayed. After execution, "\*" disappears and the prompt (>) is displayed. When searching for a file name, "\*" is not displayed. The same applies to the CLOAD? command.

If the specified file name is not found, the computer continues to search for the file name even after the end of the tape has been reached. Press the **BREAK** key to stop searching for the file name. The same applies to the CLOAD?, CHAIN, MERGE, and INPUT commands.

If an error occurs during execution of the CLOAD command, the program being loaded is invalid.

Only programs stored using the CSAVE command can be loaded using the CLOAD command.

See tape operation in the section describing peripherals.

**EXAMPLE:**

CLOAD\*  
CLOAD "PRO3"\*\*\*

\* Loads the first program found on the tape.



\*\* Searches the tape for the program "PRO3" and loads it.

---

# CLOAD?

D

---

**FORMAT:** 1. CLOAD? "filename"   
2. CLOAD? 

**Abbreviation:** CLO.?

**See Also:** CLOAD, CSAVE

---

**PURPOSE:**

Compares a program saved on tape with one stored in memory.

**REMARKS:**

Valid only as direct input in the PRO or RUN mode

To verify that a program was saved correctly, rewind the tape to the beginning of the program and use the CLOAD? command.

Format 1 searches the tape for the program indicated by "filename" and then compares it to the program stored in memory.

Format 2 compares the program stored in memory with the first program stored on the tape, starting at the current tape position.

When the tape program does not match with the one stored in memory, an error will occur. During execution, "\*" is displayed at the bottom right after the file name is displayed. After execution, "\*" disappears and the prompt (>) is displayed.

**Note:**

When loading a program created with another computer and stored on tape, the program is converted to PC-E500 code. In such a case, the CLOAD? command cannot be executed.

See tape operation in the section describing peripherals.

**EXAMPLE:**

CLOAD?\*  
CLOAD?"PRO3"\*

\* Compares the first program found on the tape with the one in memory.

\*\* Searches the tape for the program "PRO3" and compares it to the one in memory.

---

# CLOSE

---

P  
D

**FORMAT:** CLOSE [# file number, # file number, ...]

**Abbreviation:** CLOS.

**See Also:** OPEN

---

**PURPOSE:**

Closes a file or files on the currently accessed device.

**REMARKS:**

This command closes files with the specified file numbers. If no file number is specified, all files are closed. The file numbers are then released for use with other files.

All files are closed in the following cases:

- An END or RUN command is executed.
- The power is turned off.
- The computer is changed to an operation mode other than PRO or RUN.
- The program is written or read (by the LOAD, CLOAD, or MERGE command).

**EXAMPLE:**

CLOSE #2, #5, #21

Closes files #2, #5, and #21.



---

# CLS

---

P  
D

**FORMAT:** CLS

**Abbreviation:**

**See Also:** LOCATE

---

**PURPOSE:**

Clears the display.

**REMARKS:**

Clears the display and resets the display start position to (0,0).

**EXAMPLE:**

```
10: WAIT 20
20: INPUT A$
30: FOR B = 0 TO 39
40: CLS
50: LOCATE B, 1
60: PRINT A$
70: NEXT B
80: CLS
90: END
```

This program displays the entry while moving it from left to right on the display. Each time the FOR...NEXT loop of lines 30-70 is executed, the display is cleared with the CLS command, the display start position is shifted with the LOCATE command, and the contents of A\$ are displayed with the PRINT command. By writing and clearing in this manner, the display can be made to appear to move. (Delete line 40 and run the program. Note the difference.)

---

# COLOR

---

P  
D

**FORMAT:** COLOR expression

**Abbreviation:** COL.

**See Also:** LLINE, RLINE, CIRCLE, PAINT

---

**PURPOSE:**

Specifies the color of characters or lines to be printed by the CE-515P.

**REMARKS:**

Four different colors can be specified by giving 0 to 3 as the value of the expression.

- 0 ... Black
  - 1 ... Blue
  - 2 ... Green
  - 3 ... Red
- 

---

# CONSOLE

---

P  
D

**FORMAT:** CONSOLE [expression]

**Abbreviation:** CONS.

**See Also:** OPEN, LPRINT, LLIST

---

**PURPOSE:**

Sets the number of columns per line for sending data.

**REMARKS:**

This command sets the number of columns per line for data sent from the serial I/O interface (terminal) using the LPRINT or LLIST command.

The computer sends an end code (CR, LF, or CR+LF) after sending the preset line of data.

Valid values of the expression are integers in the range of 1 to 160.

For expression values from 1 to 7 inclusive, the number of columns is set to 7. If the value of the expression exceeds 160, 160 columns per line will be set. An error results if the value is 0 or negative.

If an expression is not specified, the command is ignored and the number of columns previously set is retained.


The number of columns is set to 39 when the batteries are replaced or when the RESET button is pressed while holding the  ON key.

---

# CONT

---

D

**FORMAT:** CONT 

**Abbreviation:** C.

**See Also:** STOP


---

**PURPOSE:**

Continues a program that has been temporarily halted.

**REMARKS:**

Valid only as direct input in the RUN mode.

Enter CONT to continue running a program that has been stopped with the STOP command. Enter CONT at the prompt to continue a program that has been halted using the  key. CONT can also be used to continue a program interrupted by PRINT or GPRINT. (See WAIT command.)


**EXAMPLE:**

CONT

Continues an interrupted program.

# COPY

D

**FORMAT:** COPY "d1:filename 1" TO "[d2:]filename 2" [,A] 

d1: X, E, F, CAS, COM

d2: X, Y, E, F, CAS, COM

**Abbreviation:** COP.

**See Also:**

## PURPOSE:

Copies the contents of a file from one storage device (disk, etc.) to another storage device.

## REMARKS:

Copies the contents of the file with filename 1 within the device with d1 to another file with filename 2 within the device with d2. The following shows a cross reference for device names usable for devices 1 and 2:

		d2					
		X	E	F	CAS	COM	Y
d1	X	○	○	○	×	×	○
	E	○	○	○	○	○	
	F	○	○	○	○	○	
	CAS	×	○	○	×	○	
	COM	×	○	○	○	×	

○: Usable    ×: Not usable

An error occurs if the same name is given for d1 and d2 and filenames 1 and 2.

If "A" is specified, the system regards filename 1 as an ASCII file and code "1AH" as an EOF (end of file) code.

If CAS or COM is given for d2, the destination file becomes an ASCII file. If CAS or COM is given for d1, the source file must be an ASCII file. If d2 is omitted, the same device as d1 is assumed.


An error occurs if filename 1 is not given. If filename 2 already exists, it will be overwritten.

COPY "F:\*.\*)" TO "X:\*.\*)" copies all files in RAM disk F to a disk. COPY "X:\*.\*)" TO "F:\*.\*)" copies all files on either surface (A or B) of a disk to RAM disk F.

## Notes:

1. Whenever making a copy, note the differences in the capacities among pocket disks and RAM disks.
2. You cannot copy the contents of RAM disk F to another RAM disk F.

If X is given for d1 and Y for d2, you can copy files from one disk to another using a single disk drive.

For example, COPY "X:\*.\*" TO "Y:\*.\*" copies all files from one disk to another disk in the same drive. The system will ask you "INSERT DESTINATION DISK OK?" and "INSERT SOURCE DISK OK?". After inserting the requested disk in response to each prompt, press the  key. Repeat these operations until the system returns to the BASIC prompt, ">".

The extension can be omitted if it is blank.

Notes:

1. An error occurs if the device with d2 or the file with filename 2 is write-protected (see SET command).
2. The wildcard cannot be used for device name "CAS" or "COM".

---

# CROTATE

---

P  
D

**FORMAT:** CROTATE expression

**Abbreviation:** CR.

**See Also:**

---

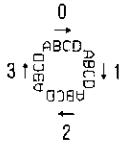
**PURPOSE:**

Specifies the orientation and printing direction of characters to be printed by the CE-515P.

**REMARKS:**

This command is effective only when the printer is in the graphics mode. By changing the value of the expression, you may change the printing direction and orientation of characters to be printed.

If you give a value in the range of 0 to 3 to the expression, one of the following four orientations and printing directions of characters is specified, causing the printer to print in the direction indicated by the arrow.



When the LTEXT command is executed, the orientation and printing direction of characters to be printed automatically returns to normal, as it does when the value 0 is given.

**EXAMPLE:**

```
5: OPEN
10: GRAPH
20: GLCURSOR (200, -30)
30: FOR Z = 0 TO 3
40: CROTATE Z
50: LPRINT "PABCD"
60: NEXT Z
70: LTEXT
80: LPRINT
90: END
```

---

# CSAVE

---

P  
D

**FORMAT:** 1. CSAVE "filename"  
2. CSAVE  
3. CSAVE "filename", "password"  
4. CSAVE, "password"

**Abbreviation:** CS.

**See Also:** CLOAD, CLOAD?, PASS

---

**PURPOSE:**

Saves a program to tape.

**REMARKS:**

Format 1 writes all program lines in memory to the tape and assigns the indicated filename.

Format 2 writes all program lines in memory to the tape with no filename.

Format 3 writes all program lines in memory to the tape and assigns them the indicated filename and password.

Format 4 writes all program lines in memory to the tape without a filename and assigns the indicated password.

Programs saved with a password may be loaded by anyone, but only someone who knows the password can list or modify them. (See PASS command.)

If a program in memory is write-protected, the CSAVE command is ignored.

Avoid writing different programs with the same file name onto the same side (side A or B) of a tape. This may cause the wrong program to be read. It is recommended that the number on the tape counter is recorded when writing a program onto tape. Wild cards (\*,?) and the period for the file extension cannot be used.

See tape operation in the section describing peripherals.

**EXAMPLE:**

CSAVE "PRO3", "SECRET"

Saves all programs in memory to tape under the name "PRO3", protected with the word "SECRET".

---

# CSIZE

---

P  
D

**FORMAT:** CSIZE expression

**Abbreviation:** CSI.

**See Also:**

---

**PURPOSE:**

Specifies the size of characters to be printed.

**REMARKS:**

The size of characters to be printed can be specified by giving a value within the range 1 to 15.

The value "1" is the minimum size of characters; the size of characters specified by 2, 3, 4, ... will be two, three, four, ... times the minimum character size, as will be the character pitch and line spacing.

Note:

When the printer power is turned on, the printer is set to print characters at CSIZE 2.

**EXAMPLE:**

CSIZE 1

Character size: 0.8 mm (W) × 1.2 mm (H) (4 × 6 steps)

Character pitch: 1.2 mm (6 steps)

Line spacing: 2.4 mm (12 steps)



---

# DATA

---

P

**FORMAT:** DATA list of values

**Abbreviation:** DA.

**See Also:** READ, RESTORE

---

**PURPOSE:**

Provides values for use by READ.

**REMARKS:**

When assigning initial values to an array, it is convenient to list the values in a DATA statement and use a READ statement in a FOR...NEXT loop to load the values into the array. When the first READ is executed, the first value in the first DATA statement is returned. Succeeding READs use succeeding values in the sequential order in which they appear in the program, regardless of how many values are listed in each DATA statement or how many DATA statements are used.

A DATA statement may contain any numeric or string values, separated by commas. Enclose string values in quotes. Spaces at the beginning or end of a string should be included in the quotes.

DATA statements have no effect if encountered in the course of regular execution of the program, so they can be inserted wherever appropriate. Many programmers include them after the READ that uses them. If desired, the values in a DATA statement can be read a second time using the RESTORE statement.

**EXAMPLE:**

```
10: DIM B(10)
20: WAIT 128
30: FOR I = 1 TO 10
40: READ B(I)
50: PRINT B(I)
60: NEXT I
70: DATA 10,20,30,40,50,60
80: DATA 70,80,90,100
90: END
```

[10] Sets up an array.

[40] Loads the values from the DATA statement so that B(1) will be 10,B(2) will be 20,B(3) will be 30, etc.

---

# DEFDBL

---

P  
D

**FORMAT:** 1. DEFDBL character range  
2. DEFDBL

**Abbreviation:** DEF.  
**See Also:** DEFSNG

---

**PURPOSE:**

Defines variable(s) with single-character names as having double precision accuracy or specifies "double-precision" mode calculations.

**REMARKS:**

In format 1, the variables in the "character range" are designated as double precision. "character range" can be specified as follows:

- DEFDBL C-F  
where variables, C, D, E and F are designated as double precision, or
- DEFDBL E,F,Z,H-J  
where variables E, F, Z, H, I and J are designated as double precision.

Variable names followed by the single-precision type declaration character (!) are given type priority over variable names declared by the DEFDBL statement. For example, variables E and F in the statement

```
DEFDBL E,F
```

will be treated as double-precision variables, but E! and F! will be treated as single-precision variables. Variables not declared as double- or single-precision will be treated as single-precision variables.

In format 2, all subsequent calculations are carried out to double precision. The DBL mark is shown on the screen in this mode. The double-precision mode is canceled by the following:

- The DEFSNG statement is executed
- The computer is turned off
- The RUN or NEW command is executed
- A program is loaded (except by the CHAIN command)

When using the DIM statement to establish the number of elements allowed in a numeric array, the DEFDBL statement must be used first if those elements are to be treated as double-precision variables:

```
DEFDBL A  
DIM A(3,2)
```

In the example below, the elements will be treated as single-precision variables:

```
DIM A(3,2)
DEFDBL A
```

---

## DEFSNG

---

P  
D

**FORMAT:** 1. DEFSNG character range  
2. DEFSNG

**Abbreviation:** DEFS.

**See Also:** DEFDBL

---

**PURPOSE:**

Defines variable(s) with single-character names as having single precision accuracy or cancels double-precision mode specified by DEFDBL.

**REMARKS:**

In format 1, the variables in the "character range" are designated as single-precision. "Character range" can be specified as follows:

- DEFSNG C–F  
where variables C, D, E and F are designated as single-precision, or
- DEFSNG E,F,Z,H–J  
where variables E, F, Z, H, I and J are designated as single-precision.

Variable names followed by double-precision type declaration characters (#) are given type priority over variable names declared by the DEFSNG statement. For example, variables E and F in the statement

```
DEFSNG E,F
```

will be treated as single-precision variables, but variables E# and F# as double-precision variables. Variables not declared as double- or single-precision are treated as single-precision variables.

In format 2, all subsequent calculations are carried out with single precision. The DBL mark on the screen is canceled in this mode.

---

# DEGREE

---

P  
D

**FORMAT:** DEGREE

**Abbreviation:** DE.

**See Also:** GRAD, RADIAN

---

**PURPOSE:**

Changes the form of angular values to decimal degrees.

**REMARKS:**

The computer has three forms for representing angular values — decimal degrees, radians and gradient. These forms are used in specifying the arguments to the SIN, COS, and TAN functions and in returning the results from the ASN, ACS, and ATN functions.

The DEGREE function changes the form of all angular values to decimal degree form until GRAD or RADIAN is used. The DMS and DEG functions can be used to convert angles from decimal degree form to degree, minute, second form and vice versa.

**EXAMPLE:**





```
10: DEGREE
20: X = ASN 1
30: PRINT X
```

[20] X now has a value of 90; i.e., 90 degrees, the arc sine of 1.

---

# DELETE

D

- FORMAT:**
1. DELETE line number 
  2. DELETE line number { } 
  3. DELETE line number { } line number 
  4. DELETE { } line number 

**Abbreviation:** DEL.

**See Also:** NEW, PASS

---

**PURPOSE:**

Deletes specified program lines in memory.

**REMARKS:**

Valid only as direct input in the PRO mode.

Format 1 deletes only the specified program line. Format 2 deletes program lines from the line number specified up to the highest program line in memory. Format 3 deletes all program lines between the first specified line number (lower value) and the second specified line number (higher value). Format 4 deletes program lines from the lowest line number in memory up to the specified line number.

Using DELETE in the RUN mode generates an error. If a password has been used, the command is not executed and the prompt is displayed. Only the digits 0-9 can be in the line numbers. Specifying a line that does not exist generates an error. Specifying a start line number that is greater than the end line number also generates an error.

If the first and second line numbers are omitted, an error will occur.

To delete the whole program, use the NEW command.

**EXAMPLE:**

```
DELETE 150*
DELETE 200,**
DELETE 50-150***
DELETE ,35****
```

- \* Deletes line 150 only.
- \*\* Deletes from line 200 to the highest line number.
- \*\*\* Deletes all lines between and including line 50 and line 150.
- \*\*\*\* Deletes from the lowest line number up to line 35.

---

# DIM

---

P  
D

**FORMAT:** DIM variable name 1 (size 1[, size 2, size 3, ...] )  
[, variable name 2 (size 1 [, size 2, size 3, ...])]

**Abbreviation:** D.

**See Also:** ERASE, CLEAR, RUN

---

**PURPOSE:**

Reserves space for numeric and string array variables.

**REMARKS:**

DIM must be used to reserve space for an array variable. The size of an array is the number of elements in that array.

The variable name consists of a letter and up to 39 alphanumeric characters. For string variables, "\$" is attached to the end of the variable name. Numeric variables may be either single-precision or double-precision variables. Even though two variables may have the same name (one single-, one double-precision), they will be recognized as two different variables.

Size 1, size 2 are called the "subscripts", and specify the number of elements in the nth dimension of the array. An array with one subscript is called a one-dimensional array, with two subscripts, it is called a two-dimensional array and an array with n subscripts is called an n-dimensional array.

*Example:*

DIM B(3):

one-dimensional array B( ) reserves 4 array elements B(0), B(1), B(2) and B(3)

DIM XA\$(2,3):

two-dimensional string array XA\$( ) reserves 12 array elements XA\$(0,0), XA\$(0,1), ..., XA\$(2,2), XA\$(2,3)

Integers 0–65534 can be used as subscripts, but an error may occur if a variable with the specified size cannot be reserved because of limits in the memory size and conditions of use.

An error will occur if a string array variable exceeds 64K bytes.

If the subscripts include a decimal point, only the integer part will be recognized (and the fractional part will be ignored).

*Example:*

X(2.3) recognized as X(2)

Y(0.25) recognized as Y(0)

The subscript may be declared by a numeric variable or expression:

10: INPUT A,B

20: DIM X(A), Y#(B-1,A\*B)

More than one array can be declared using one DIM statement.

*Example:*

```
DIM V(5), K$(4,3), XB$(5)
```

If an array has been defined, it cannot be defined again. For example, both DIM X(5) and DIM X(3,4) cannot be defined since the variable names are the same. However, DIM X!(5) and DIM X#(3,4) can be defined since one is a single-precision variable and the other is a double-precision variable.

When a program is executed using the RUN or ARUN command, allocated array variables are cleared, but they are not cleared using the GOTO statement. Thus, when a program is to be executed again using the GOTO statement, an error will occur if a DIM statement attempts to reallocate space for an existing array variable. Either GOTO a line following the DIM statement, or add an ERASE statement and redefine the array.

*Example:*

```
50: ERASE X: DIM X(3,4)
```

Numerical array and string array variables are recognized as different arrays; thus, the arrays Z( ) and Z\$( ) can be defined simultaneously.

The DIM statement cannot be used within a FOR...NEXT loop.

---

# DSKF

---

P  
D

**FORMAT:** 1. DSKF (d:)  
          d: X, E, F  
          2. DSKF (n)  
          n: 1 for disk, 3 for RAM disk E, 4 for RAM disk F

**Abbreviation:** DS.

**See Also:**

---

**PURPOSE:**

Returns the amount of free space on a pocket disk or RAM disk E or F.

**REMARKS:**

DSKF returns the size of the free disk area in bytes.

The pocket disk is used in blocks of 512 bytes; the RAM disk is used in blocks of 256 bytes.

To store a program formatted in intermediate code, an additional 20 bytes are required for control area. Thus, a 500 byte program will require 1,024 bytes on a pocket disk, or 768 bytes on RAM disk E or F.

**EXAMPLE:**

DSKF(1)

Returns the free space on the pocket disk.



---

# END

---

P

**FORMAT:** END

**Abbreviation:** E.

**See Also:**

---

**PURPOSE:**

Signals the end of a program.

**REMARKS:**

The program will be terminated when the END statement is executed. Statements after the END statement in the same line cannot be executed. All opened files are closed.

**EXAMPLE:**

```
10: PRINT "HELLO"  
20: END  
30: PRINT "GOODBYE"  
40: END
```

With these programs in memory, RUN 10 prints HELLO, but not GOODBYE. RUN 30 prints GOODBYE.

---

# EOF

---

P  
D

**FORMAT:** EOF (file number)

**Abbreviation:** EO.

**See Also:**

---

**PURPOSE:**

Determines if the end of a sequential file has been reached.

**REMARKS:**

The EOF function checks if all data in a sequential file (with the specified file number) has been read.

If all data has been read, EOF returns -1 (true) as its function value. If not, EOF returns 0 (false). For the device name COM, EOF returns -1 (true) if the 10-character buffer is empty and 0 (false) if not.

An error occurs if a file with the specified number has not been opened for input.

*Example:*

```
IF EOF(21) THEN CLOSE #21
```

File #21 is closed if all data in that file has been read.

**EXAMPLE:**

```
10: OPEN "X:A" FOR OUTPUT AS #2
20: PRINT #2, 123,456,789
30: CLOSE
40: OPEN "X:A" FOR INPUT AS #2
50: INPUT #2,A,B
60: X = EOF (2)
70: INPUT #2,C
80: Y = EOF (2)
90: CLOSE:END
```

[60] Not all data has been read in this line. X = 0.

[70] All data has been read. Y = -1.

---

# ERASE

---

P  
D

**FORMAT:** ERASE array 1 [, array 2, ... array n]

**Abbreviation:** ER.

**See Also:** CLEAR, DIM

---

**PURPOSE:**

Erases specified arrays.

**REMARKS:**

Array elements cannot be erased individually; the whole array is cleared and its memory area is freed. To re-define an array size, first ERASE it and then re-specify it in a DIM statement.

Double-precision array variables can be specified. The contents of double-precision variables specified using the DEFDBL command are erased, but the DEFDBL mode is not.

Do not use the ERASE command within a FOR...NEXT loop.

**EXAMPLE:**

```
10: DIM AA(10)
   :
   :
200: ERASE AA
```

---

# ERL

---

P  
D

**FORMAT:** ERL

**Abbreviation:**

**See Also:** ERN, ON ERROR GOTO

---

**PURPOSE:**

Returns the line number at which an error occurred during program execution.

**REMARKS:**

The ERL function is used with the ERN function and the ON ERROR GOTO statement in error processing routines to control program flow when an error occurs. A line number is only set in ERL if the error occurred during program execution.

ERL will be cleared when

- (1) a RUN statement is executed
- (2) the power is turned off
- (3) a program is loaded.

**EXAMPLE:**

See ERN.

---

# ERN

---

P  
D

**FORMAT:** ERN

**Abbreviation:**

**See Also:** ERL, ON ERROR GOTO

---

**PURPOSE:**

Returns the error code number of the execution error.

**REMARKS:**

The ERN function is used with the ERL function and the ON ERROR GOTO statement in error processing routines to control program flow when an error occurs.

See Appendix A for a list of error messages.

ERN will be cleared when

- (1) a RUN statement is executed
- (2) the power is turned off
- (3) a program is loaded.

**EXAMPLE:**


```
10: ON ERROR GOTO 100:WAIT
20: FOR N = 1 TO 20
30: READ A
40: PRINT A
50: NEXT N
60: END
100: IF ERL = 30 AND ERN = 53 THEN PRINT "YOU HAVE NO DATA"
```

---

# FILES

---

D

**FORMAT:** FILES [{"d:}[filename]"   
d: X, E, F

**Abbreviation:** Fl.

**See Also:** LFILES, SET

---

**PURPOSE:**



Displays names and attributes of specified file(s) on RAM disk or pocket disk.















**REMARKS:**

FILES displays the filename, the filename extension (.BAS or other extension), and "P" (write-protection) attribute (see SET command).

If no device name is specified, the last device name used will be assumed. If no filename is specified, all files on the specified device will be displayed. If neither device nor filename is specified, all files on the last device used will be displayed. To display a series of filenames, use an ambiguous filename. (See below.) To display a single filename, specify only that filename and its extension.

For RAM files, the number of bytes used is also displayed. (See the SAVE and SET commands.)

A maximum of four filenames will be displayed at one time, and an ⇄ mark will appear to the left of the filenames. Scroll through the files by pressing the  and  keys to move the ⇄ mark up or down, respectively.

Press  +  to move to the bottom of the previous page, and  +  to move to the top of the next page. Pressing   or   when the ⇄ mark is next to a desired filename allows the file to be loaded into memory. Pressing   or   kills (deletes) the file where the ⇄ mark is located. Once a file has been deleted, it cannot be recovered, so use this option with care. To avoid loading or killing a file, press any key other than  or  when the OK? prompt appears.

Specify an ambiguous filename to list directory information on groups of files with common name forms. There are two wildcard characters available for this purpose. The asterisk "\*" stands for any number of characters (including none) in the filename. The question mark "?" stands for a single character in a filename. The following are examples of the use of wildcard characters:

File specification

TEST?  
T??T  
S?MPLE  
A?????  
R\*

Files listed

TEST, TESTS, TEST1, TESTA  
TEST, TEXT, TORT, TXYT  
SIMPLE, SAMPLE, S2MPLE  
ABCDEF, APPEND, A12345  
RATES, R1, RETURNS, RAND2, R

If the **GOCE**, **BREAK**, or **SHIFT** + **CA** keys are pressed, or if the last filename is being displayed and the **←** key is pressed, the entry prompt ">" is displayed, and the computer waits for the next command.

FILES has no effect if the specified files do not exist on the specified device.

**EXAMPLE:**

FILES"X:"

Lists all files on pocket disk drive X: on the screen.

FILES"E:DATA"

Displays information about the file DATA on RAM disk E.

FILES"F:????1"

Lists all files on RAM disk F that have 4-letter names ending in 1.

---

# FOR...NEXT

---

P

**FORMAT:** FOR { fixed numeric variable  
single-precision simple numeric variable } =  
expression 1 TO expression 2 [STEP expression 3]  
{  
NEXT [ { fixed numeric variable  
single-precision simple numeric variable } ]

**Abbreviation:** F. N. STE.

**See Also:**

---

**PURPOSE:**

In combination with NEXT, repeats a series of operations a specified number of times.

**REMARKS:**

FOR and NEXT are used in pairs to enclose a group of statements that are to be repeated. If the variable following NEXT is omitted, the variable following FOR is assumed. The first time this group of statements is executed the loop variable (the variable named immediately following FOR) is assigned its initial value (expression 1).

When execution reaches the NEXT statement, the loop variable is increased by the STEP value (expression 3) and then this value is tested against the final value (expression 2). If the value of the loop variable is less than or equal to the final value, the enclosed group of statements is executed again, starting with the statement following FOR. If expression 3 for step size is omitted, the increment becomes 1. If the value of the loop variable is greater than the final value, execution continues with the statement that immediately follows NEXT. Because the comparison is made at the end, the statements within a FOR...NEXT pair are always executed at least once.

When the increment is zero, FOR...NEXT will continue in an infinite loop.

The loop variable may be used within the group of statements, for example as an index to an array, but care should be taken in changing the value of the loop variable.

Write programs so that the program flow does not jump out of a FOR...NEXT loop before the counter reaches the final value. To exit a loop before it has been repeated the specified number of times, set the loop variable higher than the final value.



The group of statements enclosed by a FOR...NEXT pair can include another pair of FOR...NEXT statements that use a different loop variable as long as the enclosed pair is completely enclosed; i.e., if a FOR statement is included in the group, the matching NEXT must also be included. FOR...NEXT pairs may be "nested" up to six levels deep. Illegally jumping out of an inner loop will generate an error, a nesting error. See Appendix F.

An error results if a double-precision variable is specified as the numeric variable. Double-precision initial values, final values, and increments are treated as single-precision values.

Do not use the CLEAR, DIM or ERASE command within a FOR...NEXT loop.

---

## FRE

---

P  
D

**FORMAT:** 1. FRE 0  
2. FRE 1

**Abbreviation:** FR.

**See Also:**

---

**PURPOSE:**

Returns the free space available in the program and data area in bytes.

**REMARKS:**

FRE returns the byte count of the free space (not occupied by program, array variables, or simple variables) in the program and data area of memory.

To speed up execution, the computer reserves a certain fixed number of bytes for each string variable even though a shorter string may be assigned to the variable. Thus, the size of the free space is not affected by the lengths of strings assigned to string variables. It is, however, possible to eliminate idle space in each variable to increase free space.

Format 1 returns the free space by eliminating idle space in each string variable, so its execution may take a little more time.

Format 2 returns the free space without eliminating idle space in string variables. It is useful for determining the approximate amount of free space.

The value of free space returned by format 1 may not match that returned by format 2.

---

# GCURSOR

---

P

**FORMAT:** GCURSOR (expression 1, expression 2).

**Abbreviation:** GC.

**See Also:** LOCATE, GPRINT

---

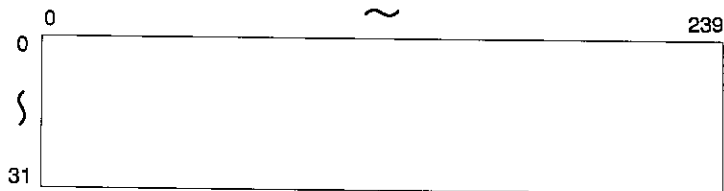
**PURPOSE:**

Specifies the starting point of dot graphics display.

**REMARKS:**

GCURSOR specifies the display starting point for the dot pattern to be displayed by the GPRINT command.

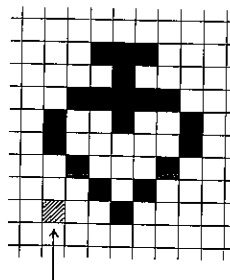
The screen consists of 240 columns of 32 rows of dots, which can be addressed by column numbers 0 to 239 and row numbers 0 to 31. Any dot on the screen can therefore be addressed as a starting point by specifying the column number with expression 1 and the row number with expression 2.



**EXAMPLE:**

```
5: CLS
10: GCURSOR (115,20)
20: GPRINT "1824458F452418"
```

This program prints the following dot pattern near the center of the screen (the display starting point indicated by the shaded box is not displayed):



Display starting point (115,20)

**Note:**

The values of expressions 1 and 2 may range from -32768 to 32767. It should be noted, however, that if the value of expression 1 is outside the range of 0 to 239 or that of expression 2 is outside the range of 0 to 31, the display starting point will become a virtual point which is outside the screen boundaries.

Location (0,7) is automatically assumed as the display starting point if the RUN command is executed or **SHIFT**+**CA** is pressed.

If a program is started with GOTO, only the row number specified by this command is maintained, with the column number automatically reset to zero.

---

# GLCURSOR

---

P  
D

**FORMAT:** GLCURSOR (expression 1, expression 2)

**Abbreviation:** GL.

**See Also:**

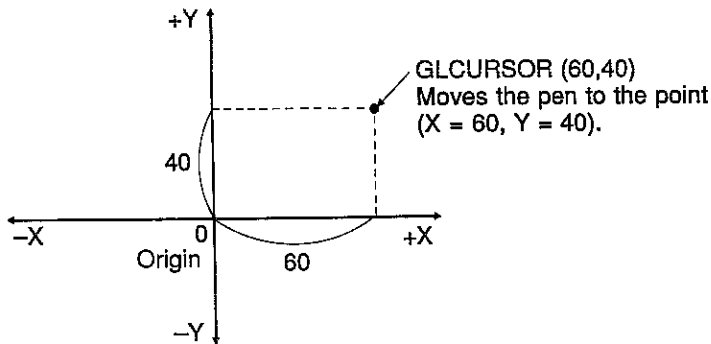
---

**PURPOSE:**

Moves the pen.

**REMARKS:**

GLCURSOR is effective only in the Graphics mode and is used to move the pen in the X or Y direction from the origin. The pen moves to the coordinates specified by expression 1 (X coordinate) and expression 2 (Y coordinate). The value of each expression must be within the range of -999 to 999. The minimum amount of movement (a value of 1) is 0.2 mm in either direction.



**Note:**

Moving the pen in the +Y direction means that the paper must be fed in the reverse direction.

Refer to the operation manual of the printer for the scissoring area.

**EXAMPLE:**

10: GRAPH

20: GLCURSOR (60,40)

Moves the pen to the point  
(X = 60, Y = 40).

---

# GOSUB...RETURN

---

P

**FORMAT:** GOSUB {line number  
                  \*label  
                  }  
                  {  
                  RETURN

**Abbreviation:** GOS. RE.

**See Also:** GOTO, ON...GOSUB

---

**PURPOSE:**

Diverts program execution to a BASIC subroutine.

**REMARKS:**

When you wish to execute the same group of statements several times in the course of a program, it is convenient to use the BASIC capability for subroutines using GOSUB and RETURN.

The group of statements is included in the program at some location where they are not reached in the normal sequence of execution. A common location is following the END statement that marks the end of the main program.

At each location in the main body of the program where a subroutine is to be executed, include a GOSUB statement with a line number or \*label that indicates the starting line number of the subroutine. The last line of each subroutine must be a RETURN.

When GOSUB is executed, the computer transfers control to the indicated line number or \*label and processes the statements until a RETURN is reached. Control is then transferred back to the statement following the GOSUB.

Subroutines may be "nested" up to 36 levels deep. (See Appendix F.)

Since there is an ON...GOSUB structure for choosing different subroutines at given locations in the program, the expression in a GOSUB statement usually consists of just the desired line number or \*label.

**EXAMPLE:**

```
10: GOSUB 100
20: END
100: PRINT "HELLO"
110: RETURN
```

When run, this program prints HELLO once.

---

# GOTO

---

P  
D

**FORMAT:** GOTO {line number}  
                  {\*label}

**Abbreviation:** G.

**See Also:** GOSUB, ON...GOTO, RUN

---

**PURPOSE:**

Transfers program control to a specified line number or \*label.

**REMARKS:**

GOTO transfers control from one location in a BASIC program to another location. Unlike GOSUB, GOTO does not "remember" the location from which the transfer occurred.

Usually, a program is executed sequentially from the smallest line number. However, execution can be transferred to the line with the given line number or \*label. Program execution can be started from the specified line by specifying a GOTO statement as direct input in the RUN mode. The transfer destination is specified by entering the line number or \*label after the GOTO command.

*Example:*

```
GOTO 40      Jumps to line 40
GOTO *AB     Jumps to the line with label *AB
```

If the specified line number or \*label does not exist, an error occurs.

If two or more identical \*labels are included in a program, program execution transfers to the line with the lower line number.

**EXAMPLE:**

```
10: INPUT A$
20: IF A$ = "Y" GOTO 50
30: PRINT "NO"
40: GOTO 60
50: PRINT "YES"
60: END
```

This program prints "YES" if a "Y" is entered and prints "NO" if anything else is entered.

# GPRINT

P  
D

- FORMAT:**
1. GPRINT "string"
  2. GPRINT expression [; expression; expression; ...]
  3. GPRINT

**Abbreviation:** GP.

**See Also:** GCURSOR, PRINT

**PURPOSE:**

Displays the specified dot pattern.

**REMARKS:**

The GPRINT command displays the specified dot pattern. Each column of bit image data is represented by 8 vertical dots.

In format 1, the 8-dot pattern is divided into a lower group of 4 dots and an upper group of 4 dots. Each group of dots is then represented by a hexadecimal number. The string is enclosed by " ".

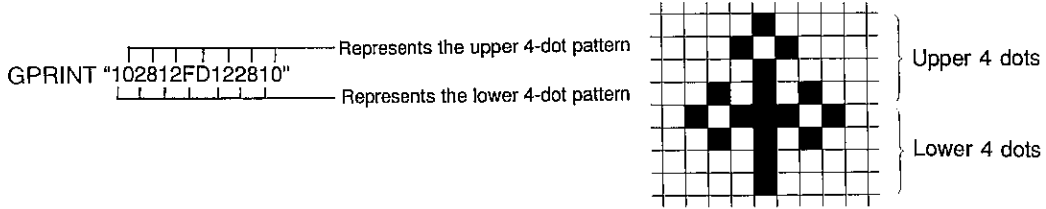
Hexadecimal number	Dot pattern	Hexadecimal number	Dot pattern	Hexadecimal number	Dot pattern	Hexadecimal number	Dot pattern
0		4		8		C	
1		5		9		D	
2		6		A		E	
3		7		B		F	

GPRINT "00 00 00 00"



Each pair of hexadecimal numbers represents one vertical dot pattern (of 8 dots). The first number represents the lower 4 dots, the second number represents the upper 4 dots.

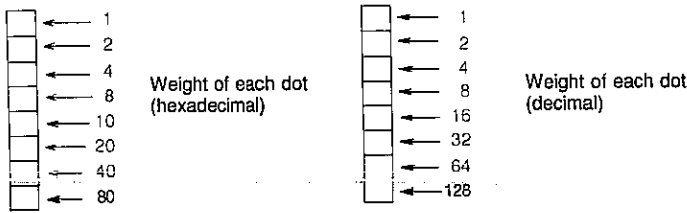
Example:



GPRINT 16;40;18;253;18;40;16 produces the same dot pattern.

Specify a semicolon (;) at the end of the string to automatically move the cursor to the next position.

In format 2, the vertical 8-dot pattern is specified using a hexadecimal or decimal value. A "weight" is assigned to each dot in the vertical 8-dot pattern, as shown below.




Specify the dot pattern with a numeric value equal to the sum of the "weights" of the dots to be displayed. The value may be any number between 0 and 255.

In format 3, the previously specified and displayed pattern is displayed without modification.

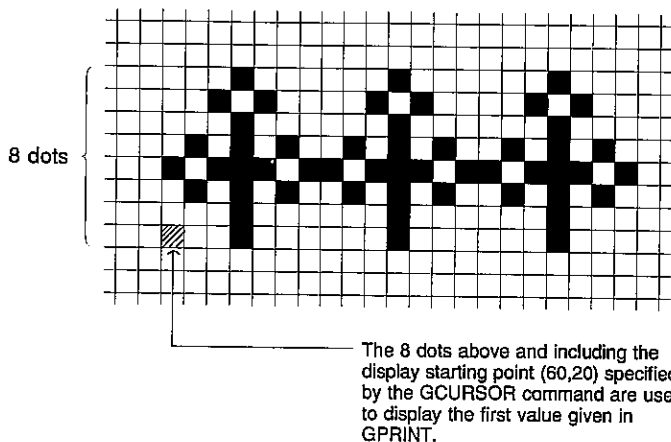
The dot pattern specified in the GPRINT command will be displayed beginning with the 8 dots on and above the display starting point specified by the G\_CURSOR command.

Note:

If the GPRINT command is terminated with ";", a subsequent GPRINT command takes effect from the next cursor position (the ";" concatenates the commands). If the GPRINT command is terminated with ":" or by pressing , the horizontal position returns to 0.

**EXAMPLE:**

```
10: AA$ = "102812FD122810"  
20: GCURSOR (60,20)  
30: GPRINT AA$;AA$;AA$
```



---

# GRAD

---

P  
D

**FORMAT:** GRAD

**Abbreviation:** GR.

**See Also:** DEGREE, RADIAN

---

**PURPOSE:**

Changes the form of angular values to gradient form.

**REMARKS:**

The computer has three forms for representing angular values: decimal degrees, radians, and gradient. These forms are used in specifying the arguments to the SIN, COS, and TAN functions and in returning the results from the ASN, ACS, and ATN functions.

The GRAD function changes the form for all angular values to gradient form until DEGREE or RADIAN is used. Gradient form represents angular measurement in terms of percent gradient, i.e., a 45° angle is a 50 percent gradient.

**EXAMPLE:**

```
10: GRAD  
20: X = ASN 1  
30: PRINT X
```

X now has a value of 100, i.e., a 100 gradient, the arc sine of 1.



---

# GRAPH

---

P  
D

**FORMAT:** GRAPH

**Abbreviation:** GRAP.

**See Also:** LTEXT

---

**PURPOSE:**

Sets the printer in the Graphics mode.

**REMARKS:**




GRAPH switches the printer from the Text mode to the Graphics mode for drawing a graph.

The printer is automatically released from the Graphics mode and returns to the Text mode after the execution of the LLIST command.

To print characters in the Graphics mode, LPRINT is used in either of the following two formats:

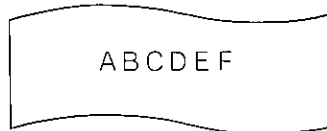
- (1) LPRINT "P character string"
- (2) LPRINT "P"  $\left. \begin{array}{l} + \\ ; \end{array} \right\}$  string variable

Notes:

1. If you interrupt printer operation by pressing the **BREAK** key while the printer is drawing a figure in the Graphics mode, be sure to enter:  
LPRINT   
If you fail to do this, subsequent commands to the printer may be executed incorrectly.
2. To return the print head to its leftmost position, enter:  
LTEXT   
LPRINT   
In this case the printer will be released from the Graphics mode.
3. When any of the printer-related commands effective only in the Graphics mode (CIRCLE, CROTATE, GLCURSOR, etc.) are used in the Text mode, the printer will print characters according to the character code.

**EXAMPLE:**

```
5: OPEN
10: GRAPH
20: GLCURSOR (200,-30)
30: LPRINT "PABCDEF"
40: LTEXT
50: LPRINT
60: END
```



---

# HEX\$

---

P  
D

**FORMAT:** HEX\$ expression

**Abbreviation:** H.

**See Also:**

---

**PURPOSE:**

Converts a decimal number into its hexadecimal character string equivalent.

**REMARKS:**

The value of the expression must be in the range of -9999999999 to 9999999999. The resulting hexadecimal character string will be up to 10 digits long.

**EXAMPLE:**

C\$ = HEX\$12 + HEX\$15

C\$ is assigned the character string "CF".

# IF...THEN...ELSE

**FORMAT:** IF condition THEN  $\left. \begin{array}{l} \text{\{line number\}} \\ \text{\{*label\}} \\ \text{\{statement\}} \end{array} \right\} [\text{ELSE } \left. \begin{array}{l} \text{\{line number\}} \\ \text{\{*label\}} \\ \text{\{statement\}} \end{array} \right\}]$

**Abbreviation:** IF T. EL.

**See Also:** AND, OR, NOT, XOR

## PURPOSE:

Conditionally executes a statement at the time the program is run.

## REMARKS:

When the condition of the IF statement is true, the statement following THEN is executed; if it is false, the statement following ELSE is executed. When the ELSE statement is omitted and the condition is false, the statement following THEN is skipped.

If THEN or ELSE is followed by a GOTO statement, either THEN or GOTO may be omitted (ELSE statement must be included).

### Example 1:

```
IF A<5 THEN C=A*B:GOTO 50
```

If A is smaller than 5, then assign the product, A\*B, to C and go to line 50.

### Example 2:

```
IF B=C+1 GOTO 60 ELSE 100
```

or

```
IF B=C+1 THEN 60 ELSE 100
```

If B equals C+1, then go to line 60; otherwise go to line 100.

The condition (e.g. A<5) of the IF statement can be any relational expression as listed below:

Relational expression	Description
○○ = ××	Equal to
○○ > ××	Greater than
○○ >= ××	No less than
○○ < ××	Less than
○○ <= ××	No greater than
○○ <> ××	Not equal to

Note: ○○ and ×× represent expressions (5\*4, A, 8, etc.).

More than one relational expression can be linked with the logical operators “\*” or “+”. For example:

IF (A>5)\*(B>1) THEN ....

If A is greater than 5 and B is greater than 1, the statement following THEN is executed. Logical operator “AND” may be used in place of “\*”.

IF (A>5)+(B>1) THEN ....

If A is greater than 5 or B is greater than 1, the statement following THEN is executed. Logical operator “OR” may be used in place of “+”.

### Using Character Strings in Relational Expressions

The magnitudes of character strings can be compared when used in a relational expression of an IF...THEN...ELSE statement. The magnitudes of character codes are compared. For example, characters A, B, and C have codes 65, 66, and 67, respectively. So A is smaller than B, and B is smaller than C.

#### **EXAMPLE:**

```
10: INPUT"CONTINUE?";A$
20: IF A$="YES" THEN 10
30: IF A$="NO" GOTO 60
40: PRINT "YES OR NO, PLEASE"
50: GOTO 10
60:
```

#### Note:

Whenever a variable name is to be followed by a statement, be sure to insert a space between them, for example:

```
100 IF A=B THEN 200
```

↑ A space is needed.



Pay special attention to this when you use the IF, FOR, ON...GOTO, or ON...GOSUB command.

---

# INIT

D

---

**FORMAT:** 1. INIT "d:?K"    
d: E, F  
2. INIT "d:"    
d: X, E, F

**Abbreviation:** INI.

**See Also:**

---

## **PURPOSE:**

Initializes a disk (RAM or pocket disk).

## **REMARKS:**

Format 1 specifies the data file areas on RAM disks E and F. The data file area lets you use programs and data as if they were stored in external storage.

Specify the size (?) of the data file area in kilobytes. The allowable size for RAM disk E is 2 to 27. For RAM disk F, the allowable size is 2 to (RAM card capacity minus 1).


For example, an area size of 2 to 31 can be specified for a 32K-byte RAM card. Note, however, that these nominal sizes may not be available depending on the size of system area or variable area used.

When this command is executed, the system asks if you are sure you want to delete the existing contents of the RAM disk. If yes, press  Y  . If no, press  N  . When  N  is pressed, the contents of the RAM disk file area are not affected.

When initializing RAM disk F, the RAM card must be installed and formatted (see 3. RAM CARD).

If "OK" is specified in format 1, the command clears the file area and cancels area definition.

If RAM disk E or F is specified as the device name in format 2, the same size as the current RAM disk area is initialized.

The pocket disks are double-sided. To initialize the reverse side, turn it over, insert it into the drive, and type INIT "X:"   again.

## **Notes:**

1. When you wish to use a RAM card as RAM disk F, the card must first be formatted before executing the INIT command. (See 3. RAM CARD)
2. When using a new pocket disk for the first time, initialize it with the INIT command.
3. When the size of the current disk area is to be changed, its contents must be cleared.

---

# INKEY\$

---

P

**FORMAT:** INKEY\$

**Abbreviation:** INK.


**See Also:**

---



**PURPOSE:**

Gives the specified variable the value of the key pressed while the INKEY\$ function is executed.

**REMARKS:**

INKEY\$ is used to respond to the pressing of individual keys without waiting for the  key to end the entry.

See the following table for a list of applicable keys and the characters that are returned.

The INKEY\$ command reads the  or  key if it is pressed. Thus, it is unable to read the function or symbol key that is pressed following either of these keys.


**EXAMPLE:**

```
5: CLS: WAIT 60
10: IF INKEY$ < > " " THEN 10
15: A$=INKEY$
20: B=ASC A$
30: IF B=0 THEN 10
40: IF B ...
```

Lines 40 and beyond contain tests for the key and the actions to be taken (for example: 60: PRINT B).

# INKEY\$ Character Code Table

High Low	0	1	2	3	4	5	...	8	9	...	F
0		SHIFT	SPACE	0		P					PF1
1	2ndF	hyp		1	A	Q			ln		PF2
2	C•CE			2	B	R			log		PF3
3	↓	FSE		3	C	S					PF4
4	↑	ANS		4	D	T					PF5
5	↓	CAPS		5	E	U		→HEX	sin		
6	RCL	CTRL		6	F	V			cos		
7	STO	BS		7	G	W		1/x	tan		
8	BASIC		(	8	H	X		x <sup>2</sup>			
9	MENU		)	9	I	Y					
A	OFF	+/-	x*		J	Z					
B	INS		+	;	K				→DEG		
C	DEL		,	L						√	
D	←		-	=	M						
E	▶	◊	.		N	y <sup>x</sup>					
F	◀	EXP	+/		O						

- The  key functions as a Break key.

---

# INPUT

---

P

- FORMAT:**
1. INPUT variable [,variable]
  2. INPUT "prompt string", variable [ [,"prompt string"], variable]
  3. INPUT "prompt string"; variable [ [,"prompt string"]; variable]

**Abbreviation:** I.

**See Also:** INPUT#, INKEY\$, READ, LOCATE


---

**PURPOSE:**

Allows entry of one or more values from the keyboard.

**REMARKS:**

When you want to enter different values each time a program is run, use INPUT to enter these values from the keyboard.

Format 1 displays symbol "?" to prompt data entry. If data is entered and the  key is pressed at this prompt, the system assigns the data to the variable and resumes program execution.

If more than one variable is specified, the data prompt is repeated the corresponding number of times.

During data prompt, format 2 displays the character string enclosed by double quotes (" ") as entry guidance. The guidance disappears when data is entered.

Format 3 also displays entry guidance during data prompt, but the entered data appears following the entry guidance, which does not disappear.

Formats 1, 2, and 3 may be concurrently used in one INPUT statement:

```
INPUT "A=";A,B,"C=?",C
```

The type of the variables given in the INPUT statement must match the type of input data. Assign string data to string variables, and numerical data to numerical variables. If "ABC" is entered in response to a numerical entry prompt, the value assigned to variable ABC is assigned. This allows you to enter such value as SIN30.

If the start position is specified using the LOCATE statement before executing the INPUT statement, the prompt string or ? will be displayed at the specified location.


**EXAMPLE:**

```
10: INPUT A
20: INPUT "A=";A
30: INPUT "A=",A
40: INPUT "X=?";X,"Y=?";Y
```

[10] Puts a question mark at the left margin.

[20] Displays "A=" and waits for data to be entered.

[30] Displays "A=". When data is entered, "A=" disappears and the data is displayed starting at the left margin.

[40] Displays "X=?" and waits for the first entry. After  is pressed, "Y=?" is displayed at the left margin.



---

# INPUT\$

---

P  
D

**FORMAT:** 1. INPUT\$ (character count)  
2. INPUT\$ (character count, # file number)

**Abbreviation:** I.\$

**See Also:** INPUT#, OPEN, PRINT#


---

**PURPOSE:**

Allows input of a character string with the specified number of characters from the keyboard or a file.

**REMARKS:**

Format 1 is used to enter a character string with the given number of characters from the keyboard. Execution automatically proceeds to the next statement after a string has been entered.

The  (carriage return) key is also counted as one character.

When entering a string from the keyboard, the OPEN command need not be executed.

Format 2 reads a character string with the given number of characters from the file with the given file number. An error occurs if the specified file has not been opened. The INPUT\$ command is valid only for a file which has been opened in the INPUT mode.

Numerals are treated as characters when read.

Since the INPUT\$ command reads the specified number of characters, the data stored in the file must have the proper format readable by the INPUT\$ command.

Note: The INPUT\$ command cannot be used for pocket disk files (X:).

**EXAMPLE:**

100: A\$=INPUT\$ (5, #5)

110: AB\$=INPUT\$(20, #5)

This program reads 5 characters into variable A\$ and then 20 characters into variable AB\$ via buffer No.5.

Values returned by INPUT\$ (from keyboard):

Byte 1

High Low	0	1	2	3	4	5	6	7
0	*	CTRL P	SPACE	Ø	@	P	SHIFT M (.)	p
1	CTRL A	CTRL Q	!	1	A	Q	a	q
2	CTRL B	CTRL R	INS	"	2	B	R	b
3	CTRL C	CTRL S	#	3	C	S	c	s
4	CTRL D	CTRL T	\$	4	D	T	d	t
5	CTRL E	CTRL U	%	5	E	U	e	u
6	CTRL F	CTRL V	&	6	F	V	f	v
7	CTRL G	CTRL W	,	7	G	W	g	w
8	CTRL H	CTRL X	(	8	H	X	h	x
9	CTRL I	CTRL Y	)	9	I	Y	i	y
A	CTRL J	CTRL Z	*	:	J	Z	j	z
B	CTRL K	CTRL =	+	;	K	[	k	(
C	CTRL L C-CE	▶	,	<	L	\	l	
D	CTRL M J	◀	-	=	M	]	m	}
E	CTRL N	↑	.	>	N	^	n	~
F	CTRL O	↓	/	?	O	_	o	DEL

\* 00H indicates the beginning of a 2-byte code.

The key operations listed in the following table cause INPUT\$ to return the following codes in the 2nd byte following 00H:

Byte 2

High Low	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	CTRL HYD						CTRL Ø	CTRL SPACE		RCL		SHIFT RCL		CTRL RCL		
1							CTRL 1	CTRL :	sin	STO	sin <sup>-1</sup>	SHIFT STO	CTRL sin	CTRL STO		
2				SHIFT INS		CTRL INS	CTRL 2	CTRL (	cos		cos <sup>-1</sup>	→xy	CTRL COS			
3						CTRL PF1	CTRL 3	CTRL )	tan		tan <sup>-1</sup>	n!	CTRL tan			
4	OFF	SHIFT OFF				CTRL PF2	CTRL 4		FSE	sinh	TAB	sinh <sup>-1</sup>	CTRL FSE	CTRL E		
5	BASIC	AER			CTRL BASIC	CTRL PF3	CTRL 5		→HEX	cosh	→DEC	cosh <sup>-1</sup>	CTRL HEX	CTRL FSE		
6	MENU	CAL			CTRL MENU	CTRL PF4	CTRL 6		→DEG	tanh	→DMS	tanh <sup>-1</sup>	CTRL DEG			
7						CTRL PF5	CTRL 7		ln		e <sup>x</sup>		CTRL ln			
8		SHIFT BS			CTRL BS		CTRL 8		log		10 <sup>x</sup>		CTRL log			
9	CAPS	SHIFT CAPS			CTRL CAPS		CTRL 9		1/x		→yθ		CTRL 1/x			
A							CTRL *		↑		SHIFT ↑		CTRL :			
B		SHIFT MENU			CTRL DEL		CTRL +		EXP		π		CTRL EXP			
C		CA	SHIFT ▶	SHIFT ▶	CTRL C-CE	▶	CTRL ,		y <sup>x</sup>		x√y		CTRL y <sup>x</sup>			
D		SHIFT ◀	SHIFT ◀	SHIFT ◀	CTRL J	◀	CTRL ,		√		³√		CTRL √			
E	◄	DRG	SHIFT ◄	SHIFT ◄	CTRL ◄	◄	CTRL ,		x <sup>2</sup>		%		CTRL x <sup>2</sup>			
F	CTRL OFF			SHIFT ↓			CTRL ↓		+/-		SHIFT +/-		CTRL +/-			

- Press the key given in the lower row in each box while holding down the **SHIFT** or **CTRL** key.
- The **ANS** key returns the same code as the **RCL** key.  
( **SHIFT** + **ANS** returns the same code as **SHIFT** + **RCL** . )

Note:

Key operations that return 2-byte codes return 2 bytes no matter whether INPUT\$(1) or INPUT\$(2) is specified.

**EXAMPLE:**

```
10: Z$=INPUT$(1)
20: PRINT HEX$ ASC Z$
30: GOTO 10
```

---

# INPUT#

---

P  
D

**FORMAT:** INPUT# file number, variable, variable, ..., variable

**Abbreviation:** I.#

**See Also:** OPEN, PRINT#

---

**PURPOSE:**

Reads items from sequential files on disk (RAM or pocket disk).

**REMARKS:**

The file number is the number given to the file when opened for input with the OPEN statement. The file number must be a number specified in an OPEN statement. If the device name is specified as COM in the OPEN statement, it is not necessary to open for input with the OPEN statement.

Specify variables as follows:

- Fixed variables (A, X, etc.)
- Simple variables (CD, EF\$, A#, B\$, etc.)
- Array elements (B(10), C\$(5,5), etc.)

Note:

Variable names with # at the end are double-precision variables.

An error occurs if the file contains less data than the number of specified variables. If the file contains more data, the rest of the data is ignored.

The data and variables must be of the same type (e.g., numeric values must be assigned to numeric variables, single-precision values to single-precision variables, double-precision values to double-precision variables, etc.)

Use a comma (,), space (&H20), CR (&H0D), LF (&H0A) or CR + LF as a delimiter when data are read into numeric variables. Spaces preceding data are ignored.

Use a comma (,), CR, LF or CR + LF as a delimiter to read data into character variables. Spaces preceding data are ignored. If a double quotation mark appears at the beginning of data, data is read up to the next double quotation mark. A comma in a character string enclosed by double quotation marks is assumed not to be a delimiter.

**EXAMPLE:**

```
10: A$ = "AB" + CHR$ 34 + "CDE" + CHR$ 34
20: B$ = CHR$ 34 + "CD,EF" + CHR$ 34
30: PRINT A$
40: PRINT B$
50: OPEN "X:ABC.DAT" FOR OUTPUT AS #2
60: PRINT #2,A$;" ";B$
70: CLOSE
80: OPEN "X:ABC.DAT" FOR INPUT AS #2
90: INPUT #2, C$, D$
100: PRINT C$
110: PRINT D$
120: CLOSE:END
```

**Execution**

```
RUN  AB"CDE"
      "CD,EF"
      AB"CDE"
      CD,EF
```

---

# KEY

---

P  
D

**FORMAT:** KEY key number, character string

**Abbreviation:** KE.

**See Also:** KEY\$

---

**PURPOSE:**

Designates a character string to a programmable function key.

**REMARKS:**

KEY assigns a command or control string to the function key with the given key number (KEY6-KEY10 are selected by pressing **PF1** – **PF5** while holding down the **SHIFT** key).

Key numbers must be in the range of 1 to 10 for **PF1** to **PF10**, respectively. Up to 31 characters or control characters can be used in the character string.

Use the CHR\$ command for any characters that cannot be entered from the keyboard.

Characters CHR\$&H80 to CHR\$&H9F and &HE0 to &HFF should not be assigned to function keys.

**EXAMPLE:**

KEY1,"R.100"+CHR\$13

This statement assigns R.100 **←** (RUN 100 **←**) to the **PF1** key.

KEY6,"G.\*AB"+CHR\$13

This statement assigns G.\*AB **←** (GOTO \*AB **←**) to the **PF6** key. Thus, you can execute a statement just by pressing the function key to which the statement string is assigned, if you include the **←** key in the string.

---

# KEY\$

---

P  
D

**FORMAT:** KEY\$ key number

**Abbreviation:** KE.\$

**See Also:** KEY

---

**PURPOSE:**

Returns the character string assigned to a programmable function key.

**REMARKS:**

KEY\$ returns the character string assigned to the key with the given key number (KEY6-KEY10 are selected by pressing  PF1  -  PF5 while holding down the  SHIFT key). Strings can also be returned to string variables.

**EXAMPLE:**


If string "ABC" is designated to KEY 3, KEY\$ 3  displays ABC. A\$=KEY\$ 3 assigns string "ABC" to string variable A\$.

---

# KILL

D

---

**FORMAT:** KILL "d:filename"   
d: X, E, F

**Abbreviation:** K.

**See Also:** SAVE, SET





---

**PURPOSE:**

Deletes a file on disk (RAM or pocket disk).

**REMARKS:**

Specify the device (d:) and filename. The extension may be omitted if it is blank.

When file names are displayed using the FILES command, files can be deleted by specifying the file with the  or  key and pressing the  and  keys.

An error occurs if the specified file does not exist, or is open.

An error occurs if the file attribute is "P".

Change the attribute to "—" using the SET command to delete a file.

**EXAMPLE:**

KILL "X:PRO1.BAS"

Deletes the file PRO1.BAS on pocket disk.

KILL "E:PRO1.BAS"

Deletes the file PRO1.BAS on RAM disk E.



---

# LEFT\$

---

P  
D

**FORMAT:** LEFT\$("string",expression)

**Abbreviation:** LEF.

**See Also:** MID\$, RIGHT\$

---

**PURPOSE:**

Returns the specified number of characters from the left end of a given string.

**REMARKS:**

LEFT\$ returns the number of characters specified by the expression from the left end of the given string.

For example, if A\$="ABCD", LEFT\$(A\$,3) returns the leftmost 3 characters, "ABC".

**EXAMPLE:**

```
10: X$="SHARP"  
20: FOR N=1 TO 5  
30: LET S$=LEFT$(X$,N)  
40: PRINT S$  
50: NEXT N
```

```
RUN  
S  
SH  
SHA  
SHAR  
SHARP
```

---

# LEN

---

P  
D

**FORMAT:** LEN "string"

**Abbreviation:**

**See Also:**

---

**PURPOSE:**

Returns the number of characters in a string.

**REMARKS:**

The number of characters in the string includes any blanks or non-printable characters such as control codes or carriage returns.

**EXAMPLE:**

```
10: INPUT "ENTER A WORD";A$
20: N=LEN A$
30: PRINT "THE WORD LENGTH IS";N
40: END
```

```
RUN
ENTER A WORD CHERRY
THE WORD LENGTH IS 6
```

[10] Prompts for a word. In this example, the user enters "CHERRY".

[20] Finds the length of the word.

[30] Prints out the answer.

---

# LET

---

P  
D

**FORMAT:** 1. LET numeric variable = expression  
2. LET string variable = string

**Abbreviation:** LE.

**See Also:**

---

**PURPOSE:**

Used to assign a value to a variable.

**REMARKS:**

LET assigns the value of the expression to the designated variable. The type of the expression must match that of the variable; i.e. only numeric expressions can be assigned to numeric variables and only string expressions can be assigned to string variables.

The LET command may be omitted in all LET statements.

**EXAMPLE:**

```
10: I=10
20: A=5*I
30: X$=STR$ A
40: IF I >=10 THEN LET Y$=X$+".00"
```

[10] Assigns the value 10 to I.  
[20] Assigns the value 50 to A.  
[30] Assigns the value 50 to X\$.  
[40] Assigns the value 50.00 to Y\$.

---

# LF

---

P  
D

**FORMAT:** 1. LF  
2. LF expression

**Abbreviation:**

**See Also:**

---

**PURPOSE:**

This command is used to advance the printing paper (CE-515P).

**REMARKS:**

LF is effective only in the Text mode.

In format 1, the printer advances the paper by one line.

In format 2, the printer advances the paper by the specified number of lines.

The value of the expression must be within the range of -999 to 999.

If the value of the expression is a positive value, the paper is advanced in the forward direction. If a negative value is given, the paper is advanced in the reverse direction.

Note:

The line spacing when LF is executed will be the same as that specified by CSIZE.

**EXAMPLE:**

LF10


Advances the paper 10 lines.

---

# LFILES

D

---

**FORMAT:** LFILES ["[d:] [filename]"]   
d: X, E, F

**Abbreviation:** LF.

**See Also:** FILES

---

**PURPOSE:**

Prints out the names and attributes of the specified file(s) stored on disk.

**REMARKS:**

When the device name is omitted, the one given in the last file statement is assumed. For example, if the last file statement is FILES"E:", RAM disk E is assumed as the device.

Specify a particular filename to print out the name and attributes of that file only. Do not include a filename if you want to print out the names and attributes of all files on disk.

A filename must always be followed by a file extension. Printout appears as "filename.ext attribute." (See the SAVE and SET commands.)

Wildcards (\* and ?) can be used to specify filenames.

**EXAMPLE:**

LFILES  Prints all the files within the device given in the last file command.

---

# LINE

---

P  
D

**FORMAT:** LINE [(expression 1, expression 2)] – (expression 3, expression 4)  
                  [, { $\begin{matrix} S \\ R \\ X \end{matrix}$ }] [,expression 5] [, { $\begin{matrix} B \\ BF \end{matrix}$ }]

**Abbreviation:** LIN.

**See Also:** GCURSOR, PSET

---

**PURPOSE:**

Used to draw a line between two specified points.

**REMARKS:**

LINE is used to draw a line from the coordinates specified by (expression 1, expression 2) to the coordinates specified by (expression 3, expression 4).

*Example:*

LINE(0,0) – (239,31)

This statement draws a diagonal line from the upper left corner to lower right corner of the screen.

The values of expressions 1 through 4 should be within the range of –32768 to 32767. To specify points within the screen, use the following range of values:

Expressions 1 and 3: 0 to 239

Expressions 2 and 4: 0 to 31

No error occurs if coordinates outside the screen are specified, provided that the values of the expressions are within the range of –32768 to 32767. In this case only the portion of the line within the range of the screen will appear.

(Expression 1, expression 2) may be omitted. If omitted, a line is drawn from the origin (0, 0) or from the point specified by (expression 3, expression 4) used in a previous LINE statement.

*Example:*

5: CLS

10: LINE (10,0) – (239,16)

20: WAIT:LINE – (100,31)

**Note:**

Since the screen is made up of a matrix of dots, a diagonal line may appear as a staircase, and curves may not appear as complete curves.

Options S, R, and X are used to set, reset, or reverse the specified line on the screen.

S: Draws a line while activating the corresponding dots on the screen (set).

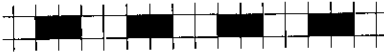
R: Draws a line while deactivating the corresponding dots on the screen (reset).

This option is useful to draw a line in reverse video or to erase an existing line.

X: Draws a line, activating the corresponding dots if they are inactive, or deactivating the corresponding dots if they are already active (reverse).

The default parameter is S.

Expression 5 is used to specify the type of line by a value from 0 to 65535. For example, if the value of expression 5 is 26214 (&H6666), the following line is drawn:



16 dots

The same dot pattern appears repeatedly to draw a dotted line.

Binary representation of 26214 (&H6666) is:

0110011001100110

If you compare this bit pattern with the dot pattern above, you will notice that the dots corresponding to "one" bits are activated while those corresponding to "zero" bits are not. Thus, the type of line is determined by the binary 16-bit pattern of the value of expression 5. If the value is zero, no line appears; and if it is 65535 (&HFFFF), a solid line is drawn. A solid line is also drawn if expression 5 is omitted.

If option R is specified, the dots corresponding to "one" bits are deactivated on the screen; if option X is specified, the status of the dots corresponding to "one" bits are reversed.

Options B and BF are used to draw a rectangle whose opposite corners are specified by (expression 1, expression 2) and (expression 3, expression 4).

Option B is used to draw a line rectangle.

Option BF is used to draw a solid rectangle.

**EXAMPLE:**

```
10: CLS : WAIT 0
20: AA$ = "102812FD122810"
30: GCURSOR (64,20)
40: GPRINT AA$;AA$;AA$
50: LINE(24,0)-(124,31),&HF18F,B
60: LINE(34,3)-(114,28),X,BF
70: GOTO 60
```

---

# LIST

---

D

- FORMAT:**
1. LIST
  2. LIST line number
  3. LIST \*label

**Abbreviation:** L.

**See Also:** LLIST, PASS

---

**PURPOSE:**

Displays a program.

**REMARKS:**

Valid only as direct input in the PRO mode.

In format 1, the program is displayed from its first line until the display is full. In format 2, the program is displayed from the specified line number until the display is full. Use the  key to advance to the next line in the list. If the line for the specified number does not exist, the program will be displayed from the line with the next largest number that does exist.

In format 3, the program is displayed from the line with the specified label until the display is full.

If a password has been set, the LIST command is ignored.

An error will occur if a \*label is specified which does not exist in the program or a line number is specified which is past the last line number in program.

**EXAMPLE:**

LIST 100

Displays line number 100.



---

# LLINE

---

P  
D

**FORMAT:** LLINE [(expression 1, expression 2)] – (expression 3, expression 4)  
[,expression 5] [,expression 6] [,B]

**Abbreviation:** LLIN.

**See Also:** RLINE, PAINT, COLOR

---

**PURPOSE:**

Used to draw a line between two specified points (CE-515P).

**REMARKS:**

LLINE is effective only in the Graphics mode and is used to draw a line from the coordinates specified by (expression 1, expression 2) to the coordinates specified by (expression 3, expression 4).

(Expression 1, expression 2) may be omitted. If omitted, a line is drawn from the current pen position to the coordinates specified by (expression 3, expression 4).

Expression 5 is used to specify one of the following types of lines by giving a value (0 to 15) to the expression. The default value of expression 5 is 0.

0	_____
1	.....
2	.....
3	.....
4	.....
5	.....
6	.....
7	.....
8	.....
9	.....
10	.....
11	.....
12	.....
13	.....
14	.....
15	_____

Expression 6 is used to specify the color of the line to be drawn. The value of expression 6 may be within the range of 0 to 3. (Refer to the COLOR command for the color specified by each value.)

Expression 6 may be omitted. If omitted, the previous value is assumed.

If B is specified at the end of the LLINE command, a rectangle is drawn using coordinates specified by (expression 1, expression 2) and (expression 3, expression 4) as the end points its diagonal.

**Example:**

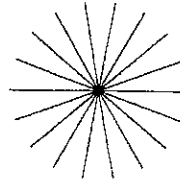
LLINE (10,20)–(200,–20),2,0,B

**Note:**

(Expression 1, expression 2) cannot be omitted when drawing a rectangle.

**EXAMPLE:**

```
5: OPEN
10: GRAPH:RANDOMIZE
20: GLCURSOR (240, -120)
25: SORGN
30: FOR J=0 TO 340 STEP 20
40: A=107 *COS J
50: B=107 *SIN J
60: R=RND 4-1
70: LLINE (0,0)-(A,B),0,R
80: NEXT J
90: LTEXT
100: LPRINT
110: END
```



This figure is actually printed in color.

[20] Moves the pen to near the center of the paper and designates it as the origin of coordinates for drawing figure.

[60] Random number 0 to 3 is assigned to R.

[70] Color is specified by the value of R.

[90-100] The printer returns to the Text mode and moves the print head to its leftmost position.

**Specification When Drawing Lines Continuously**

LLINE in the following format allows the printer to draw lines continuously.

```
LLINE (expression 1, expression 2) - (expression 3, expression 4)
      - (expression 5, expression 6) - (expression 7, expression 8)
      - (expression 9, expression 10) - (expression 11, expression 12),
      expression 13, expression 14
```

In the above format, a line connecting the point specified by (expression 1, expression 2) to the point specified by (expression 3, expression 4), and a line connecting the point specified by (expression 3, expression 4) to the point specified by (expression 5, expression 6), etc. can be drawn continuously. A maximum of six coordinate pairs can be specified.

---

# LLIST

---

D

- FORMAT:**
1. LLIST
  2. LLIST  $\left. \begin{array}{l} \text{line number} \\ *label \end{array} \right\}$
  3. LLIST [line number 1],[line number 2]

**Abbreviation:** LL.

**See Also:** LIST, PASS, OPEN, CONSOLE

---

**PURPOSE:**

Prints out a program on the optional printer.

**REMARKS:**

Valid only as direct input in the PRO or RUN mode.

When the serial I/O interface has been opened using the OPEN command, the LLIST command outputs the program to the serial I/O interface terminal. To return the program print command to the printer (CE-126P), execute the CLOSE command.

Format 1 prints or sends all of the programs in memory.

Format 2 prints or sends only the program line for which the number or label is specified.

Format 3 prints or sends the statements from line number 1 through line number 2. There must be at least two lines between the numbers.


Either line number 1 or line number 2 may be omitted. If line number 1 is omitted, the program listing is printed from its first line through line number 2. If line number 2 is omitted, the program listing is printed from line number 1 through the end of the program.

If the line with the line number given in format 2 does not exist or the lines with line numbers 1 and 2 given in format 3 do not exist, the nearest larger numbers are assumed.


If a password has been set, the LLIST command is ignored.

The number of print columns per line is set by the CONSOLE command. If set to 23 columns or less, executing the LLIST command results in an error. Use the OPEN command to specify delimiter codes.


**EXAMPLE:**

LLIST 100, 200 

Prints program listing between line numbers 100 and 200.

LLIST ,200 

Prints program listing from the first line through line 200.

LLIST 100, 


Prints program listing from line 100 through the last line.

---

# LOAD

D

---

**FORMAT:** LOAD ["d:filename",R]   
d: X, E, F, CAS, COM

**Abbreviation:** LOA.

**See Also:** SAVE, CHAIN, MERGE, LOAD?





---

**PURPOSE:**

Loads a BASIC program.

**REMARKS:**

LOAD loads a program with the specified filename. An error occurs if the program area is exceeded as a result of loading of a program. In such a case, clear unnecessary variables from the data area.

If program filenames have been displayed with the FILES command, the desired program can be simply loaded by first choosing it with the  or  key, then entering  .

If all options are omitted, the program is loaded through the serial I/O port.

The file extension may be omitted only if it is ".BAS".


When the device name (d:) is CAS or COM, only ASCII code data is valid.

If a load error occurs, a program written in intermediate format will not be loaded at all, but a program written in ASCII format is loaded to the line just before the line where the error occurred.

Specify ",R" to run the program as soon as it is loaded.

While a program is being loaded, all files are closed, and all values or variables specified for the USING, ON ERROR GOTO, WAIT, LOCATE, ERL, and ERN commands are cleared.

Up to 256 bytes of program code can be loaded at a time. An error occurs if no delimiter is encountered before loaded data exceeds 256 bytes.

When a serial I/O device is used for loading, delimiters and the end-of-file code may be set with the OPEN command. (If no end-of-file code was received, use the  key to stop loading.)


When a delimiter is received, the computer translates the loaded data into a intermediate format. An error occurs if a single line exceeds 256 bytes or the first character of a line is not a numerical character (line number).

---

# LOAD?

---

D

**FORMAT:** LOAD? ["d:filename"]   
d: X, E, F, CAS, COM

**Abbreviation:** LOA.?

**See Also:** LOAD, CLOAD?

---

**PURPOSE:**

Compares a program saved on a device with one stored in memory.

**REMARKS:**

LOAD? compares the program with the given filename with the one in memory. An error occurs if a mismatch was found during comparison.

If all options are omitted, LOAD? compares the program read from the serial I/O port with the one in memory.

The file extension may be omitted only if it is ".BAS".

When the device name (d:) is CAS or COM, only ASCII format data is valid.

Up to 256 bytes of program code can be loaded at a time. An error occurs if no delimiter is encountered before loaded data exceeds 256 bytes.

When a serial I/O device is used for loading, delimiters and the end-of-file code may be set with the OPEN command. (If no end-of-life code was received, use the

**BREAK** key to stop loading.)

---

# LOC

---

P  
D

**FORMAT:** LOC file number

**Abbreviation:**

**See Also:** OPEN

---

**PURPOSE:**

Returns the current pointer position (logical) in a file.

**REMARKS:**

The LOC command returns the number of records read or written since the file with the specified number was opened. One record is 256 bytes long. If the device name is COM (communication), LOC returns the number of data bytes stored in the 10-character (20-byte) buffer.

An error occurs if a file with the specified number has not been opened.

**EXAMPLE:**

```
10: OPEN "X:FILE01" FOR INPUT AS #2
20: IF EOF(2) THEN 50
30: INPUT #2,N
40: GOTO 20
50: M=LOC(2)
60: PRINT "THE FILE HAS";M;"RECORDS"
70: CLOSE #2
80: END
```

---

# LOCATE

---

P  
D

**FORMAT:** 1. LOCATE [expression 1] [,expression 2]  
2. LOCATE

**Abbreviation:** LO.

**See Also:** CLS, INPUT, PRINT, PAUSE, GCURSOR

---

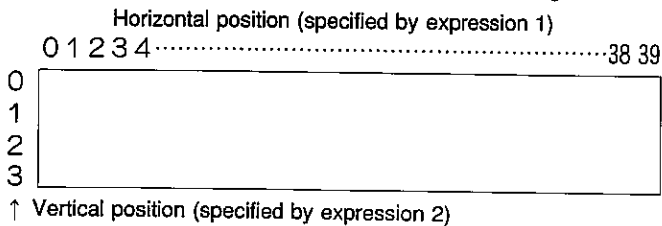
**PURPOSE:**

Specifies the display start position in column units.

**REMARKS:**

Specifies the display start position in units of a character position for the contents displayed by the PRINT command, PAUSE command, etc.

The display position is specified as follows using format 1.



A position on the display is specified by its horizontal and vertical positions. Expression 1 specifies the horizontal position, and expression 2 specifies the vertical position. The range of expression 1 is 0 to 39, and the range of expression 2 is 0 to 3. An error occurs if the expressions are not specified within these ranges.

When expression 1 or expression 2 is omitted, the current position is assumed. When a comma (,) or semi-colon (;) is not placed at the end of the statement (e.g. PRINT "ABC"), the start position for the next display command moves to the horizontal position of the next line.

**Example:**

```
10: CLS
20: LOCATE 2,1: PRINT "ABCDE"
30: LOCATE ,2: PRINT "123"
```



Format 2 clears the display start position.

Using the LOCATE command allows text to be written to any part of the display without affecting existing text except where characters are directly overwritten. Use the CLS command to clear the whole display.

If the number of characters exceed the limits of the display, the display is scrolled to show all the characters, even if the display start position was specified with the LOCATE command.



---

# LOF

---

P  
D

**FORMAT:** LOF file number

**Abbreviation:**

**See Also:** OPEN

---

**PURPOSE:**

Returns the size of the specified file.

**REMARKS:**

The LOF command returns the size of a file with the specified file number. The actual size of the file is displayed in bytes.

If the device name is COM (communication), this command returns the byte count remaining in the 10-character (20-byte) buffer.

An error will occur if the specified file is not open.

Pocket disks are used in units of 512 bytes, RAM disks E and F in units of 256 bytes; the total size of all files will not be equal to the total used pocket or RAM disk area (number of bytes).

**EXAMPLE:**

```
10: OPEN "X:FILE01" FOR INPUT AS #2
20: N=LOF(2)
30: PRINT "FILE01 FILE SIZE IN BYTES IS ";N
40: CLOSE #2
50: END
```

[10] Opens the file FILE01 for input.

[20-30] Finds the size of the file and prints out the value.

[40] Closes the file.

---

# LPRINT

---

P  
D

- FORMAT:**
1. LPRINT {expression  
string} [ , {expression  
string} ] [.]
  2. LPRINT {expression  
string} [ ; {expression  
string} ] [.]
  3. LPRINT USING "format"; {expression  
string} [ [ ; ] {expression  
string} ] [ [ ; ] ]
  4. LPRINT

**Abbreviation:** LP.

**See Also:** PRINT, USING, OPEN, CONSOLE

---

## **PURPOSE:**

Outputs given data to the printer.

## **REMARKS:**

When the serial I/O interface is opened with the OPEN command, the LPRINT command outputs the program to the serial I/O interface terminal. To return the printing command to the CE-126P Printer, execute the CLOSE command.

When printing characters in Graphics mode, see the GRAPH command.

When a comma (,) or semicolon (;) is placed at the end of the statement, the next LPRINT command in the program will print its data directly after the data printed by the first LPRINT.

If a single item is specified using format 1 for the CE-126P Printer, numerical values are printed from the right margin of the paper; strings from the left.

For output to the CE-515P Printer or the serial I/O terminal, both numerical values and strings are printed from the left margin.

Results of double-precision calculations longer than 24 digits or strings exceeding 24 characters or those exceeding the number of columns specified by the CONSOLE command are automatically wrapped round to the next line when printed.

Separate expressions or strings using commas (,) to divide the 24 columns of each printer line into 12-column areas. Single-precision numerical values or strings are printed in each of these areas; numerical calculations are printed from the right margin of each 12-column area, strings from the left. However, in Double-Precision mode, single-precision values are printed using the same number of columns as double-precision values. Strings longer than 12 characters are printed from the left margin of the first area and extend into the second area; for numerical values longer than 12 digits, the extra part of the mantissa is truncated when printed.

Format 2 prints out data in succession from the left margin.

When a semicolon (;) is placed at the end of the statement, the next LPRINT command in the program will print its data in succession to the data printed by the first LPRINT. When a serial I/O device is used, a delimiter code is transferred each time

data with the number of columns specified by the `CONSOLE` command is output.

Format 3 prints out data in the exact format specified in the statement. Either a comma (,) or semicolon (;) may be used as a separator. For the format for `USING`, see the `USING` command.

Format 4 prints only delimiter codes. If the preceding `LPRINT` statement is terminated with a semicolon (;) with unprinted data left in the buffer, format 4 prints that data.

**EXAMPLE:**

```
LPRINT AB#,A,B  
LPRINT AB#;A;B;Z$
```

---

# LTEXT

---

P  
D

**FORMAT:** LTEXT

**Abbreviation:** LT.  
**See Also:** GRAPH

---

**PURPOSE:**

Sets the Text mode (CE-515P).

**REMARKS:**

LTEXT is used to select the printer to the Text mode for printing alphabetic and numeric characters.

**Note:**

The printer is automatically put in the Text mode when `LLIST` is executed. Be sure to set the operation mode with either `GRAPH` or `LTEXT` after an Auto OFF, and each time the power switch of either the computer or the printer is turned on.

**EXAMPLE:**

```
LTEXT Sets the Text mode.
```

---

# MDF

---

P  
D

**FORMAT:** 1. MDF (expression)  
2. MDF (expression, threshold number)

**Abbreviation:** MD.

**See Also:** USING

---

**PURPOSE:**

Rounds up the value of an expression.

**REMARKS:**

The MDF function rounds the value of an expression to the number of decimal places specified by the USING command.

MDF is effective only when the number of decimal places is specified for a value by the USING command.

Format 1 uses the standard default threshold number of 4. This means that if the first digit of the truncated part is more than 4, one is carried to the non-truncated part. This threshold number can be specified using format 2.

**EXAMPLE:**

```
10: USING "###.###"  
20: A = MDF (5/9)  
30: PRINT A  
40: USING  
50: PRINT A, 5/9  
60: END
```

Display

RUN

0.556

0.556 5.55555E-01

---

# MEM\$

---

P  
D

**FORMAT:** MEM\$

**Abbreviation:** ME.

**See Also**

---

**PURPOSE:**

Indicates RAM card configuration.

**REMARKS:**

The returned character indicates the memory access condition. "S1" means that the external RAM card is not being used. "S2" means that only the external RAM card is being used. "B" means that the RAM card is being used as an extension of the internal memory space.


Refer to "3. RAM CARD" for more details.

---

# MERGE

---

D

**FORMAT:** MERGE "[d:] filename"   
d: X, E, F, CAS, COM

**Abbreviation:** MER.

**See Also:** LOAD, CHAIN

---

**PURPOSE:**

Loads a program file from the specified device and merges it with a program in memory.

**REMARKS:**

MERGE retains the program in memory and then loads the specified program. The program file to be merged must have ASCII format (see the SAVE command).

Note:

When a program in intermediate format is loaded, an abnormal condition may occur.

If the line numbers overlap, the lines of the specified file replace the lines of the program in memory.

If the device name is omitted, the device name used in the last file command is assumed. For example, if the last file command is FILES"E:", RAM disk E is assumed as the device name.

An error occurs if the program overflows the program area as a result of a MERGE command. Clear unnecessary variables, then try MERGE again.

**EXAMPLE:**

Program in memory

```
10: INPUT A, B
15: PRINT A, B
20: C = SQR (A * A + B * B)
25: PRINT C
30: END
```

Program to be merged (ASCII format)

```
10: INPUT A, B, C
20: S = (A + B + C)/2
30: AREA = SQR (S * (S - A) * (S - B) * (S - C))
40: PRINT AREA
50: END
```

Program after merge

```
10: INPUT A, B, C
15: PRINT A, B
20: S = (A + B + C)/2
25: PRINT C
30: AREA = SQR (S * (S - A) * (S - B) * (S - C))
40: PRINT AREA
50: END
```

The contents of lines 10, 20, and 30 are replaced with those of the same lines of the loaded program.

---

# MID\$

---

P  
D

**FORMAT:** MID\$(string,N,M)

**Abbreviation:** M.

**See Also:** LEFT\$, RIGHT\$

---

**PURPOSE:**

Returns a string of M characters from inside a string starting from the Nth character in the string.

**REMARKS:**

If N is greater than the number of characters in the string, a null string is returned. If N is less than 1, an error occurs. M must be in the range of 0 to 254 and N in the range of 1 to 254. Fractions will be rounded down.

**EXAMPLE:**

```
10: Z$="ABCDEFGH"  
20: LET Y$= MID$(Z$,3,4)  
30: PRINT Y$
```

Display

RUN 


CDEF

---

# NAME

---

D

**FORMAT:** NAME "d:old filename" AS "new filename"   
d: X, E, F

**Abbreviation:** NA.

**See Also:**

---

**PURPOSE:**

Renames files on RAM or pocket disks.

**REMARKS:**

The NAME command renames the disk file "old filename" as "new filename" on the RAM disk (E: or F:) or pocket (X:) disk.

An error occurs if "old filename" does not exist, or a file with "new filename" already exists. An error occurs if "old filename" is protected with the "P" (write-protection) function. An error occurs if "old filename" is open.

The extension can be omitted if it is blank.

**EXAMPLE:**

NAME "E:OLDNAM" AS "E:NEWNAM"

Names file OLDNAM on the RAM disk E as NEWNAM.



---

# NEW

---

D

**FORMAT:** NEW

**Abbreviation:**

**See Also:** CLEAR, PASS

---

**PURPOSE:**

Clears existing programs and data.

**REMARKS:**

The NEW command clears all programs and data that are currently in memory.  
(Programs with passwords cannot be cleared.)

The contents of the AER memories are retained.

**EXAMPLE:**

NEW

---

# ON ERROR GOTO

---

P

**FORMAT:**

1. ON ERROR GOTO {line number} {\*label}
2. ON ERROR GOTO 0

**Abbreviation:** O. ERR. G.

**See Also:** ERN, ERL, RESUME

---

**PURPOSE:**

Sets up an error trapping routine.

**REMARKS:**

When the error trap function is enabled, control is transferred to the error routine if an error is detected. An error message is not displayed.

Within an error routine, control can be branched depending on the value of ERN and ERL. The routine is terminated by the RESUME command.

Format 2 clears declaration of error trapping. Declaration of error trapping is also cleared in any of the following cases:

- (1) RUN command is executed.
- (2) Computer is turned off.
- (3) A program is loaded.
- (4) **SHIFT** + **CA** is pressed.

The program stops executing if an error occurs within an error routine.

**EXAMPLE:**

See ERN.

---

# ON...GOSUB

---

**FORMAT:** ON expression GOSUB {line number 1  
\*label 1}, {line number 2  
\*label 2}, ...

**Abbreviation:** O. GOS.

**See Also:** GOSUB, GOTO, ON...GOTO

---

**PURPOSE:**

Executes one of a set of subroutines, depending on the value of a control expression.

**REMARKS:**

When ON...GOSUB is executed, the expression between ON and GOSUB is evaluated and reduced to an integer. If the value of the integer is 1, control is transferred to line number 1 or \*label 1 in the list, as in a normal GOSUB. If the expression is 2, control is transferred to line number 2 or \*label 2 in the list, and so forth.

**Note:**

Be sure to place a space just before the GOSUB command. Otherwise it may be regarded as a variable.

If the expression is zero, negative, or larger than the number of line numbers provided in the list, no subroutine is executed and execution proceeds with the next statement or line of the program.

An error occurs if the value of the expression is -32769 or less or 32768 or more.

Use commas (,) to separate line numbers or \*labels in the list.

**EXAMPLE:**

```
10: INPUT A
20: ON A GOSUB 100,200,300
30: END
100: PRINT "FIRST"
110: RETURN
200: PRINT "SECOND"
210: RETURN
300: PRINT "THIRD"
310: RETURN
```

An entry of 1 displays "FIRST"; 2 displays "SECOND"; 3 displays "THIRD". Any other entry does not produce any display.

---

# ON...GOTO

---

P

**FORMAT:** ON expression GOTO  $\left\{ \begin{array}{l} \text{line number 1} \\ \text{*label 1} \end{array} \right\}, \left\{ \begin{array}{l} \text{line number 2} \\ \text{*label 2} \end{array} \right\}, \dots$

**Abbreviation:** O. G.

**See Also:** GOSUB, GOTO, ON...GOSUB

---

**PURPOSE:**

Transfers control to one of a set of locations, depending on the value of a control expression.

**REMARKS:**

When ON...GOTO is executed the expression between ON and GOTO is evaluated and reduced to an integer. If the value of the integer is 1, control is transferred to line number 1 or \*label 1 in the list. If the expression is 2, control is transferred to line number 2 or \*label 2 in the list, and so forth.

**Note:**

Be sure to place a space just before the GOTO command. Otherwise it may be regarded as a variable.

If the expression is zero, negative, or larger than the number of line numbers provided in the list, execution proceeds with the next statement or line of the program.

An error occurs if the value of the expression is -32769 or less or 32768 or more.

Use commas (,) to separate line numbers or \*labels in the list.

**EXAMPLE:**

```
10: INPUT A
20: ON A GOTO 100,200,300
30: GOTO 900
100: PRINT "FIRST"
110: GOTO 900
200: PRINT "SECOND"
210: GOTO 900
300: PRINT "THIRD"
310: GOTO 900
900: END
```

An entry of 1 displays "FIRST"; 2 displays "SECOND"; 3 displays "THIRD". Any other entry does not produce any display.

---

# OPEN

---

P  
D

- FORMAT:**
1. OPEN "d:filename" FOR mode AS # file number  
d: X, E, F, CAS
  2. OPEN "baud rate, parity, word length, stop bit, type of code, delimiter, end-of-file code, XON, shift code " AS # file number
  3. OPEN

**Abbreviation:** OP.

**See Also:** CLOSE, OPEN\$

---

**PURPOSE:**

Format 1 opens the file specified by "d:filename" for use with the specified file number. Subsequent input/output to the file is accomplished by referring to the file number.

Formats 2 and 3 allow data to be transferred through the serial I/O interface (COM).

**REMARKS:**

The file number must be from 1 to 255.

A total of 6 file number control areas for a disk and a total of 2 for other devices are allocated. Up to 6 disk files and up to 2 files for other devices can be opened simultaneously. However, only one file can be opened for devices CAS and COM.

**Note:**

When executing the SAVE, LOAD, CHAIN or MERGE statement, the computer opens one file automatically.

In format 1, "mode" specifies the method of access to the file, as follows:

INPUT	Specifies sequential input from an existing file.
OUTPUT	Specifies sequential output to a device or file.
APPEND	Specifies addition to a sequential file.

If OUTPUT is specified using an existing filename, that file is erased before the new one is created.

An error occurs when using the APPEND or INPUT mode if the specified file does not exist.

An error occurs when using the APPEND or OUTPUT mode if the specified file has the "P" attribute set (write-protection: see SET command).

An error occurs if an attempt is made to open a file which has already been opened, or allocate a file number which has already been allocated.

*Example:*

For pocket disks

OPEN "X:PRO1" FOR OUTPUT AS #3

Creates a new file on the disk named PRO1 with file number 3.

For RAM disk E

OPEN "E:PRO1" FOR OUTPUT AS #21

Creates a new file on the RAM disk E named PRO1 with file number 21.

In format 2, the following parameters can be selected:

Baud Rate: 300, 600, 1200, 2400, 4800, 9600

Specifies the modulation rate (transfer rate).

(1 baud = 1 bit/sec)

Parity: N, E, O

Specifies the type of parity.

N: No parity bit is transmitted nor received.

E: Specifies even parity.

O: Specifies odd parity.

Word Length: 7, 8

Specifies how many bits to be transmitted or received per character.

Number of Stop Bits: 1, 2

Type of Code: A

Always specify A, since the computer can send/receive only ASCII codes.

Delimiter: C, F, L

Specifies the type of delimiter to indicate the end of data, end of a program line, etc.

C: Specifies the CR (carriage return) code.

F: Specifies the LF (line feed) code.

L: Specifies the CR code + LF code.

End-of-file Code: &H00-&HFF

Specifies the end-of-file code used to indicate the end of the program, etc.

(May be required when using the SAVE or LOAD commands.)

XON: N, X

Specifies communication control using XON and XOFF.

N: Specifies no control using XON and XOFF.

X: Specifies control using XON and XOFF.

Shift code: S, N

When the word length is specified as 7 characters, the SO, SI switching enables transfer of characters whose codes are 128 or greater.

S: Specifies to switch SO and SI

N: Specifies not to switch SO and SI

*Example:*

```

OPEN "COM:1200,N,8,1,A,C,&H1A,N,S" AS #22
  1200 ..... Baud rate (1200 baud)
  N..... Parity (none)
  8 ..... Word length (8 bits)
  1 ..... Number of stop bits (1 bit)
  A..... Type of code (ASCII)
  C..... Delimiter (CR code)
  &H1A.... End-of-file code (&H1A)
  N..... XON (none)
  S..... Shift code (Yes)

```

The device name "COM:" can be omitted.

The conditions in the example above are set after the batteries have been replaced or after the RESET button is pressed.

Any condition specified in the OPEN command can be omitted. If omitted, the current condition remains unchanged.

```

OPEN ",,2"

```

Only the number of stop bits is changed.

In format 3, all conditions set previously are retained. This format enables data to be transferred through the I/O interface. File number 1 is always used.

---

# OPEN\$

---

P  
D

**FORMAT:** OPEN\$

**Abbreviation:** OP.\$  
**See Also:** OPEN

---

**PURPOSE:**  
Obtains the currently set I/O conditions.

**REMARKS:**  
The currently set I/O conditions are obtained as a character string.

**EXAMPLE:**  
 OPEN\$   
 1200, N, 8, 1, A, C, &H1A, N, S

---

# PAINT

---

P  
D

**FORMAT:** PAINT expression 1 [, expression 2]

**Abbreviation:** PAI.

**See Also:** LLINE, RLINE, COLOR

---

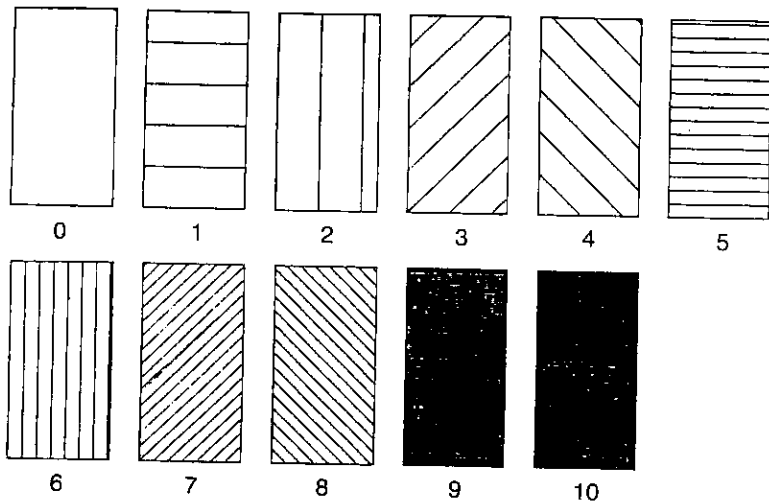
**PURPOSE:**

Used to hatch the inside of a rectangle.

**REMARKS:**

PAINT is effective only in the Graphics mode.

For rectangles which are drawn using the LLINE or RLINE command with option B, values 0 to 10 may be specified for expression 1.

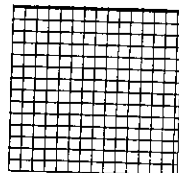


Expression 2 is used to specify the color of hatching. The value of expression 2 must be within the range 0 to 3. (Refer to the COLOR command for the color specified by each value.)

Repeated use of the PAINT command together with the LLINE command allows the printer to draw a special pattern as shown in the example.

**EXAMPLE:**

```
LLINE(0,0)-(200,-200),0,0,B  
PAINT 5,0  
PAINT 6,0
```





---

# PASS

D

---

**FORMAT:** PASS "character string" 

**Abbreviation:** PA.

**See Also:** CSAVE, SAVE, CLOAD, LOAD, DELETE, LIST, NEW, RENUM

---

**PURPOSE:**

Sets and cancels passwords.

**REMARKS:**

Passwords are used to protect programs from listing or editing by unauthorized users. A password consists of a character string that is no more than 8 characters long. The 8 characters must be alphanumeric characters or symbols. The character string cannot be a null string.

Once a PASS command has been given, the programs in memory are protected. A program protected by a password cannot be examined or modified in memory. It cannot be saved to tape or disk, or listed with LIST or LLIST. Nor is it possible to add or delete program lines. The only way to remove this protection is to execute another PASS command with the same password.

If a password is set in the program to be loaded, that password is also set within the computer. If not, no password is set within the computer.

If PASS is executed when no program is in the computer, an error occurs and no password is set.

A password-protected program is protected against the NEW or DELETE command as well.

Press  immediately after the password.

Writing characters or symbols after the password results in an error and the password cannot be canceled.

**Example:**

PASS "ABCDEFGH":A = 123  → An error occurs.

**EXAMPLE:**

PASS "SECRET" 

Establishes the password "SECRET" for the program in memory.

---

# PAUSE

---

P  
D

- FORMAT:**
1. PAUSE  $\left. \begin{array}{l} \text{expression} \\ \text{string} \end{array} \right\} \left[ \begin{array}{l} \left. \begin{array}{l} \text{expression} \\ \text{string} \end{array} \right\} \right] [;]$
  2. PAUSE  $\left. \begin{array}{l} \text{expression} \\ \text{string} \end{array} \right\} \left[ \begin{array}{l} \left. \begin{array}{l} \text{expression} \\ \text{string} \end{array} \right\} \right] [;]$
  3. PAUSE USING "format";  $\left. \begin{array}{l} \text{expression} \\ \text{string} \end{array} \right\} \left[ \left[ \begin{array}{l} (, \\ ; \end{array} \right] \left. \begin{array}{l} \text{expression} \\ \text{string} \end{array} \right\} \right] \left[ \begin{array}{l} (, \\ ; \end{array} \right]$
  4. PAUSE

**Abbreviation:** PAU.

**See Also:** PRINT

---

**PURPOSE:**

Briefly shows information on the display.

**REMARKS:**

PAUSE is used to display prompt information, results of calculations, etc. The operation of PAUSE is identical to PRINT, except that after PAUSE the computer waits for a preset interval of about .85 seconds and then continues execution of the program.

This command is provided to ensure compatibility with other PC models. It is recommended that you replace it with the PRINT command wherever possible.

For the format with the USING command, see the USING command.

---

# POINT

---

P  
D

**FORMAT:** POINT (expression 1, expression 2)

**Abbreviation:** POI.

**See Also:** GCURSOR, PSET, PRESET

---

**PURPOSE:**

Returns the status of a specified dot.

**REMARKS:**

POINT returns 1 if the dot specified by coordinates (expression 1, expression 2) is set, and returns zero if it is cleared. If the specified dot is outside the display boundaries, the command returns -1.

The values of expressions 1 and 2 may be within the range of -32768 to 32767. A dot within the display boundaries is addressed only if the value of expression 1 is 0 to 239 and that of expression 2 is 0 to 31.

**EXAMPLE:**

10: CLS : WAIT 5:A = 75	
20: LINE (50, 0) - (50, 31)	Draws two vertical lines.
30: LINE (100, 0) - (100, 31)	
40: PSET (A, 16)	Sets a dot between the two lines.
50: B = POINT (A + 1,16)	Tests whether the dot on the right side of the active dot is active or not.
60: IF B THEN 150	If it is set, go to line 150.
70: PSET (A + 1,16)	If it is cleared, set it.
80: PRESET (A, 16)	Inactivates the dot which was first set.
90: A = A + 1	Increments the coordinate to address the next dot position.
100: GOTO 50	Returns to line 50.
150: B = POINT (A - 1,16)	Tests whether the dot on the left side of the active dot is active or not.
160: IF B THEN 50	If it is set, go to line 50.
170: PSET (A - 1, 16)	If it is cleared, set it.
180: PRESET (A,16)	Clears the dot which was first set.
190: A = A - 1	Decrements the coordinate to address the preceding dot position.
200: GOTO 150	Go to line 150.

Executing this program causes a dot to move back and forth between the two vertical lines.

---

# PRESET

---

P  
D

**FORMAT:** PRESET (expression 1, expression 2)

**Abbreviation:** PRE.

**See Also:** PSET, GCURSOR, POINT

---

**PURPOSE:**

Clears (resets) a dot at the specified coordinates on the display.

**REMARKS:**

PRESET clears the dot at the specified (expression 1, expression 2).

The values of expressions 1 and 2 may be within the range of  $-32768$  to  $32767$ . A dot within the display boundaries is addressed only if the value of expression 1 is 0 to 239 and that of expression 2 is 0 to 31.

**EXAMPLE:**

```
10: CLS
20: LINE (20, 0) - (130, 31), BF
30: FOR X = -25 TO 25 STEP 0.5
40: Y = -1 * SQR ABS (25 * 25 - X * X)
50: PRESET (X + 75, Y + 31)
60: NEXT X
70: WAIT : GPRINT
```

Executing this program draws a half circle within a solid rectangle.

---

# PRINT

---

P  
D

- FORMAT:**
1. PRINT  $\left\{ \begin{array}{l} \text{expression} \\ \text{string} \end{array} \right\} \left[ \left[ \begin{array}{l} \text{expression} \\ \text{string} \end{array} \right] \right] [,]$
  2. PRINT  $\left\{ \begin{array}{l} \text{expression} \\ \text{string} \end{array} \right\} \left[ \left[ \begin{array}{l} \text{expression} \\ \text{string} \end{array} \right] \right] [;]$
  3. PRINT USING "format";  $\left\{ \begin{array}{l} \text{expression} \\ \text{string} \end{array} \right\} \left[ \left[ \begin{array}{l} \text{expression} \\ \text{string} \end{array} \right] \right] \left[ \left[ \begin{array}{l} \text{expression} \\ \text{string} \end{array} \right] \right] \left[ \left[ \begin{array}{l} \text{expression} \\ \text{string} \end{array} \right] \right]$
  4. PRINT

**Abbreviation:** P.

**See Also:** LPRINT, PAUSE, USING, WAIT, LOCATE

---

**PURPOSE:**

Displays information.

**REMARKS:**

PRINT displays prompt information, results of calculations, etc.

If the start position is specified by the LOCATE statement, the data will be displayed from the specified location.

If a comma (,) or semicolon (;) is at the end of the statement, the contents will be displayed continuously.

Format 1 displays as follows:

- 1) For a single item to be displayed:

If the expression is numeric, the value is shown from the right margin of the display. If it is a string, it is shown from the left margin of the display.

- 10: PRINT 1234  
20: PRINT "ABCD"

```
          |                1234 |
          | ABCD                |
```

- 2) For two or more items to be displayed (specified with commas):

Single-precision values are displayed in groups of 13 columns, while double-precision values are displayed in 40 columns. Numeric values are displayed from the right margin, and strings from the left margin.

In double-precision mode, one value is displayed on each line, single-precision numeric values are displayed in the same format as the double-precision numeric values.

```

10: A = 1234: B# = 5#/9: C$ = "ABCDE":WAIT 200
20: CLS: PRINT "A=",A
30: CLS: PRINT A,C$,B#
40: CLS: PRINT A,B#,C$

```

```

RUN [←] | A=                1234                |
        |                                     |
        |          1234ABCDE                 |
        | 5.55555555555555555555556D-01    |
        |          1234                5.5555555555 |
        | 555555556D-01ABCDE                 |

```

If a single-precision value exceeds 12 digits (when the decimal fraction in the exponential display is 7 digits or more), the least significant digits are truncated. When a character string exceeds 12 columns, only the first 12 characters (from the left) are displayed.

Format 2 displays the data continuously from the left margin of the display.

```

10: A = 1234:B# = 5#/9:C$ = "ABCDE"
20: PRINT "A=";A
30: PRINT "EFGH";B#;C$;A

```

```

RUN [←] | A= 1234                |
        | EFGH 5.55555555555555555555556D-01 ABCDE 123 |
        | 4                                                                |
        | >                                                                |

```

Format 3 displays the data by following the specified format.

Refer to the USING command for USING format. Commas (,) and semicolons (;) will be treated as usual. A USING statement can be used only once in one PRINT statement.

*Example:* PRINT USING "&&&&&&&";"ANSWER=";;PRINT USING "####.##";5/9

Format 4 displays the previously displayed value as is. (Usually, it is used together with the WAIT command to retain the current display.)

```

10: CLS
20: FOR A=0 TO 159
30: PRINT CHR$(A+32);
40: NEXT A
50: WAIT: PRINT

```

The characters displayed between lines 20 and 40 will remain on the display at line 50. (An infinite interval is set.)

### PRINT → LPRINT setting

The computer can switch all PRINT commands to function as LPRINT commands. Connect the printer before executing the following statement:

Setting: PRINT=LPRINT

Resetting: PRINT=PRINT

Resetting can also be performed by:

- 1) executing the RUN command
- 2) pressing the **SHIFT** + **CA** keys
- 3) turning the power off and then on.

Since the RUN command resets the setting, run the program using the GOTO command.

---

# PRINT#

---

P  
D

**FORMAT:** PRINT# file number, {expression  
string } [ [ , ] {expression  
string } ] [ [ , ] ]

**Abbreviation:** P.#

**See Also:** OPEN, INPUT#

---

**PURPOSE:**

Writes values of specified variables into a specified file.

**REMARKS:**

PRINT# is valid only for a file opened for OUTPUT or APPEND with the OPEN command. The file number is the number given to the file when opened. The mode does not need to be specified when the device name is COM (communication).

When an array variable (one or two dimensions) has been specified in the form of "array name(\*)", the entire array is written to the file. Its elements are written in the order of, for example, C\$(0,0), C\$(0,1), C\$(0,2)...C\$(1,0).....C\$(5,5). It is recommended that the FOR...NEXT statement be used for writing array variable data.

When the respective elements of the array are specified, they must be specified in the form of "B(7)", "C\$(5,6)", etc.

When a character or string element is used, it must not be specified using a comma (,) or semicolon (;):

PRINT#2,"ABC"

PRINT#2,A\$

If PRINT#2,"ABC",A\$ is executed, no data delimiter is written and "ABC" and A\$ cannot be distinguished.

A numeric value is recorded in such a form that the sign (space when it is positive), numeric character string, and space appear in that form. The recording format is shown below:

- (1) When a comma or semicolon does not follow the data, CR(&H0D) and LF(&H0A) are provided.

*Example:*

PRINT #2, -1.2

-	1	.	2		CR	LF
---	---	---	---	--	----	----

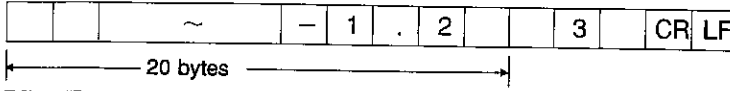
PRINT #2, "ABC"

A	B	C	CR	LF
---	---	---	----	----

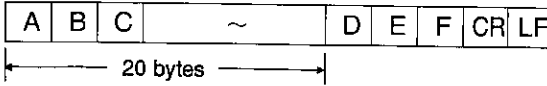
(2) When a comma follows the data, 20 bytes are occupied. A numeric value is right justified and a character string is left justified.

*Example:*

PRINT #2, - 1.2,3



PRINT #2, "ABC", "DEF"



When the character string exceeds 20 bytes, the excess part is written to the next 20-byte area. The maximum size is 254 bytes. However, it is 80 bytes when the device is a pocket disk.

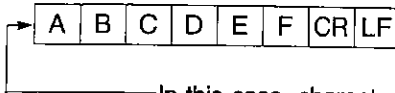
(3) When a semicolon follows the data, it is stored without spaces.

*Example:*

PRINT #2, - 1.2;3



PRINT #2, "ABC";"DEF"



In this case, character strings "ABC" and "DEF" are not read separately.

When character strings are recorded with commas or semicolons, they must be read with the INPUT\$ command by specifying the exact format in which they were recorded. However, the INPUT\$ command is not usable for disks.

**EXAMPLE:**

```

10: OPEN "E:DATA" FOR OUTPUT AS #2
20: FOR J=0 TO N
30: FOR K=0 TO M
40: PRINT #2,C$(J, K)
50: NEXT K:NEXT J:CLOSE
  
```



---

# PSET

---

P  
D

**FORMAT:** 1. PSET (expression 1, expression 2)  
2. PSET (expression 1, expression 2) ,X

**Abbreviation:** PS.

**See Also:** PRESET, GCURSOR, POINT

---

**PURPOSE:**

Sets or clears a dot at the specified coordinates on the display.

**REMARKS:**

Format 1 sets the dot at the coordinates (expression 1, expression 2).

Format 2 clears the specified dot if it is set, and sets it if it is cleared.

The values of expressions 1 and 2 may be within the range of -32768 to 32767. A dot on the display is addressed only if the value of expression 1 is 0 to 239 and that of expression 2 is 0 to 31.

**EXAMPLE:**

```
10: CLS :DEGREE
20: FOR A=0 TO 600 STEP 3
30: B= -1 * SIN A
40: Y=INT (B *16)+16
50: X=INT (A/4)
60: PSET (X,Y)
70: NEXT A
80: WAIT:GPRINT
```

---

# RADIAN

---

P  
D

**FORMAT:** RADIAN

**Abbreviation:** RAD.

**See Also:** DEGREE, GRAD

---

**PURPOSE:**

Changes the form of angular values to radians.

**REMARKS:**

The computer has three forms for representing angular values — degrees, radians, and gradient. These forms are used in specifying the arguments to the SIN, COS and TAN functions and in returning the results from the ASN, ACS, and ATN functions.

The RADIAN function changes the form of all angular values to radian form until DEGREE or GRAD is used. Radian form represents angles in terms of the length of the arc with respect to the radius, i.e.,  $360^\circ$  is  $2\pi$  radians, since the circumference of a circle is  $2\pi$  times the radius.

**EXAMPLE:**

```
10: RADIAN
20: X = ASN 1
30: PRINT X
```

X now has a value of 1.570796327 or  $\pi/2$ , the arc sine of 1.

---

# RANDOMIZE

---

P  
D

**FORMAT:** RANDOMIZE

**Abbreviation:** RA.

**See Also:** RND

---

**PURPOSE:**

Resets the seed for random number generation.

**REMARKS:**

When random numbers are generated using the RND function, the computer begins with a predetermined "seed" or starting number. RANDOMIZE resets this seed to a new randomly determined value.

The starting seed will be the same each time the computer is turned on, so the sequence of random numbers generated with RND is the same each time, unless the seed is changed. This is very convenient during the development of a program because it means that the behavior of the program should be the same each time it is run, even though it includes a RND function. When you want the numbers to be truly random, the RANDOMIZE statement can be used to make the seed itself random.

**EXAMPLE:**

```
10: RANDOMIZE  
20: X = RND 10
```

When run from line 20, the value of X is based on the standard seed. When run from line 10, a new seed is used.

---

# READ

---

P

**FORMAT:** READ variable, variable, ..., variable

**Abbreviation:** REA.

**See Also:** DATA, RESTORE

---

**PURPOSE:**

Reads values from a DATA statement and assigns them to variables.

**REMARKS:**

When assigning initial values to an array, it is convenient to list the values in a DATA statement and use a READ statement in a FOR...NEXT loop to load the values into the array. When the first READ is executed, the first value in the first DATA statement is returned. Succeeding READs use succeeding values in the order in which they appear in the program, regardless of how many values are listed in each DATA statement or how many DATA statements are used.

If desired, the values in a DATA statement can be read a second time using the RESTORE statement.

**Note:**

The type of data must match the type of variables (numerical or string) to which it is to be assigned.

**EXAMPLE:**

```
10: DIM B(10)
20: WAIT 60
30: FOR I = 1 TO 10
40: READ B (I)
50: PRINT B(I)*2;
60: NEXT I
70: DATA 10, 20, 30, 40, 50, 60
80: DATA 70, 80, 90, 100
90: READ C, D, E$, F$
100: PRINT C,D,E$,F$
110: DATA 3,5,G=,H=
120: END
```

[10] Set up an array.

[40] Loads the values from the first DATA statements into B( ). B(1) is 10, B(2) is 20, B(3) is 30, etc.

[90] Loads the values from the final DATA statement. C is 3, D is 5, E\$ is G=, F\$ is H=.

---

# REM(')

---

P

**FORMAT:** REM remark or ' remark

**Abbreviation:**

**See Also:**

---

**PURPOSE:**

Includes comments in a program.

**REMARKS:**

It is often useful to include explanatory comments in a program. These can provide titles, names of authors, dates of last modification, usage notes, reminders about algorithms, etc. These comments are included using the REM (or apostrophe (')) statement.

The REM (') statement has no effect on program execution and can be included anywhere in the program. Everything following REM (' in that line is treated as a comment.

**EXAMPLE:**


```
10: ' THIS LINE HAS NO EFFECT
100: REM THIS LINE HAS NO EFFECT EITHER.
```

---

# RENUM

D

---

**FORMAT:** RENUM [new line number] [, [old line number] [,increment]] 

**Abbreviation:** REN.

**See Also:** DELETE, LIST

---


**PURPOSE:**

Renumbers the lines of a program.



**REMARKS:**

Valid only as direct input in the PRO mode.

The line numbers are changed from old line numbers to new line numbers with the specified increment. If the new line number is not specified, the lines are renumbered starting with line 10. If the increment is not specified, the lines are renumbered with an increment of 10. RENUM updates referenced line numbers in GOTO, ON...GOTO, GOSUB, ON...GOSUB, RESTORE, and (IF)...THEN statements.


An error occurs if line numbers are given as a variable (GOTO A) or numerical expression (GOTO 2  50). If an error occurs, renumbering is not carried out. If a line number is given by a variable or expression, temporarily make it a remark (REM), and correct it after executing the RENUM command. It is recommended that you replace such commands with ON...GOTO commands, etc.

If a line number exceeds 65279, an error is generated. If a specified old line number does not exist, an error is generated. Changing the execution order generates an error. If a password has been used, an error occurs.

If the display shows "\*", pressing the  key will interrupt renumbering. A display of "\*\*" indicates that renumbering cannot be interrupted. Error generation or use of the  key leaves the program unchanged.

**EXAMPLE:**

```
10: INPUT "CONTINUE";A$
20: IF A$ = "YES" THEN 10
30: IF A$ = "NO" THEN 60
40: PRINT "ENTER YES OR NO PLEASE!"
50: GOTO 10
60: END
```

RENUM 100, 10, 5 

```
100: INPUT "CONTINUE";A$
105: IF A$ = "YES" THEN 100
110: IF A$ = "NO" THEN 125
115: PRINT "ENTER YES OR NO PLEASE!"
120: GOTO 100
125: END
```

---

# RESTORE

---

P

**FORMAT:** 1. RESTORE {line number  
                  \*label }  
          2. RESTORE

**Abbreviation:** RES.

**See Also:** DATA, READ

---

**PURPOSE:**

Rereads values in a DATA statement or changes the order in which these values are read.

**REMARKS:**

In the regular use of READ the computer begins reading with the first value in a DATA statement and proceeds sequentially through the remaining values. Format 1 resets the pointer to the first value of the DATA statement whose line number is equal to the specified line number or \*label. Format 2 resets the pointer to the first value of the first DATA statement, so that it can be read again.

**EXAMPLE:**

```
10: DIM B(10)
20: WAIT 32
30: FOR I = 1 TO 10
40: RESTORE
50: READ B(I)
60: PRINT B(I)*I;
70: NEXT I
80: DATA 20
90: END
```

[10] Sets up an array.

[50] Assigns the value 20 to each of the elements of B( ).

---

# RESUME

---

P

**FORMAT:** 1. RESUME  
2. RESUME NEXT  
3. RESUME {line number}  
          {\*label}

**Abbreviation:** RESU.

**See Also:** ON ERROR GOTO

---

**PURPOSE:**

Resumes program execution at the end of an error handling routine.

**REMARKS:**

RESUME resumes program execution after completing an error handling routine to which control was passed by the ON ERROR GOTO command. This command validates the ON ERROR GOTO command again. If control is returned to the main program by any other command (GOTO, etc.), execution will be aborted if an error subsequently occurs.

The error handling routine lets you take the necessary action to prevent recurrence of the same error.

Control is returned depending on the format:

- (1) Format 1 returns control to the statement which caused the error. If an error occurs again in the same statement, the error handling routine is executed again.
- (2) Format 2 returns control to the statement following the error statement.
- (3) Format 3 returns control to the specified line.

**EXAMPLE:**

```
10: ON ERROR GOTO 100
20: INPUT A, B
30: PRINT A/B
   :
   :
100: RESUME 20
    :
```

If zero is assigned to variable B or an overflow occurs from A/B, control returns to the input routine on line 20 and prompts for correct data entry.



---

# RIGHT\$

---

P  
D

**FORMAT:** RIGHT\$(string,N)

**Abbreviation:** RI.

**See Also:** LEFT\$, MID\$

---

**PURPOSE:**

Returns N characters from the right end of a string.

**REMARKS:**

Fractions will be truncated. If N is less than 1, a null string is returned. If N is greater than the number of characters in the string, the whole string is returned.

**EXAMPLE:**

```
5: WAIT 60
10: XX$ = "SHARP COMPUTER"
20: FOR N = 1 TO 14
30: SS$ = RIGHT$(XX$,N)
40: PRINT SS$
50: NEXT N
```

---

# RLINE

---

P  
D

**FORMAT:** RLINE (expression 1, expression 2)-(expression 3, expression 4)  
[, expression 5][, expression 6][, B]

**Abbreviation:** RL.

**See Also:** LLINE, PAINT, COLOR

---

**PURPOSE:**

Used to draw a line between the two points specified by relative coordinates.

**REMARKS:**

Valid only in the Graphics mode.

RLINE differs from LLINE in that LLINE takes the origin of coordinates specified by SORGN as a reference and specifies the location of each point by coordinates relative to that origin, whereas RLINE takes the current position of the pen as the origin of coordinates and specifies the location of the next point relative to that origin.

The format is the same as LLINE, except that (expression 1,expression 2) cannot be omitted. (See LLINE command.)

**EXAMPLE:**

```
5: OPEN
10: GRAPH
20: FOR A = 1 TO 5
30: RLINE (0,0) - (40,-30)
40: RLINE (0,0) - (40,30)
50: NEXT A
60: LTEXT
70: LPRINT
80: END
```



---

# RND

---

P  
D

**FORMAT:** RND numeric expression

**Abbreviation:** RN.

**See Also:** RANDOMIZE

---

**PURPOSE:**

Generates a random number.

**REMARKS:**

If the value of the expression is less than 1 but greater than or equal to zero, the random number is less than 1 and greater than zero. If the expression is an integer greater than or equal to 1, the result is a random number greater than or equal to 1 and less than or equal to the expression. If the expression is greater than or equal to 1 and not an integer, the result is a random number greater than or equal to 1 and less than or equal to the smallest integer that is larger than the expression. (In this case, the generation of the random number changes depending on the value of the decimal portion of the argument.) If the expression is negative, the previously set numeric expression is used to generate the random number.

<u>Argument</u>	<u>Result</u>	
	<u>Lower Bound</u>	<u>Upper Bound</u>
.5	0 <	< 1
2	1	2
2.5	1	3



The same sequence of random numbers is normally generated because the same "seed" is used each time the computer is turned on. To randomize the seed, see the RANDOMIZE command.

---

# RUN

---

D

**FORMAT:** 1. RUN   
2. RUN { line number }  
          { \*label } 

**Abbreviation:** R.

**See Also:** GOTO, ARUN

---

**PURPOSE:**

Executes a program in memory.

**REMARKS:**

Format 1 executes a program beginning with the lowest numbered statement in memory.

Format 2 executes a program beginning with the specified line number.

An error occurs if the specified line number or \*label was not found.

If two or more identical labels exist in a program, the one with a smaller line number is executed.

**EXAMPLE:**

RUN 100

Executes the program starting from line 100.

---

# SAVE

---

P  
D

**FORMAT:** SAVE ["d:filename" [,A]]  
d: X, E, F, CAS, COM

**Abbreviation:** SA.

**See Also:** LOAD, DSKF, MERGE, CSAVE, FILES

---

**PURPOSE:**

Saves the basic program to the specified device.

**REMARKS:**

The SAVE statement names a BASIC program in memory and then writes it to the specified file.

A file name is the name given to a program or a set of data. The desired file can be readily retrieved by the computer if it is given a file name.

A filename may consist of up to eight alphanumeric characters or symbols.

If all options are omitted, COM (communication) is assumed for the device name.

If the A option is specified, the file is saved in ASCII format, otherwise it is saved in intermediate code format. When the device name is CAS or COM, the file is saved in ASCII format even if the A option is not specified.

If no extension is specified, .BAS is assumed. The extension can consist of up to three characters.

An existing file will be erased if the same filename is specified, but an error occurs if the existing file has the "P" (write-protect) attribute set (see SET command). An error occurs if the program in memory is made secret.

For the number of recordable files (including data files), see "3. RAM CARD". For the remaining capacity of a disk or RAM disk E or F, see the DSKF command.

**EXAMPLE:**

SAVE "E:PRO1", A

Saves the program as an ASCII file with the name "PRO1" on RAM disk E.

---

# SET

D

---

**FORMAT**      SET "d:filename", { "P" }  
                       " " }

                    d: X, E, F

**Abbreviation:** SE.

**See Also:** SAVE

---

**PURPOSE:**

Assigns or removes file protection.

**REMARKS:**

The contents of a file cannot be inadvertently deleted or rewritten if "P" is specified. To clear the protection, specify a space (" "). Once protection has been removed, the file can be deleted or written to freely.

Wildcards (\* or ?) can be used for filename specification, but the extension must not be omitted. For example, the .BAS extension must be specified.

An error occurs if the SET command is used with a file that is open.

Note:

A disk can be write-protected by sliding its write-protect tab (see the Operation Manual for the CE-140F).

**EXAMPLE:**

SET "X:PAYRUN.BAS", "P"

Protects the program "PAYRUN.BAS" on pocket disk from being written to, erased, or renamed.

---

# SORGN

---

P  
D

**FORMAT:** SORGN

**Abbreviation:** SO.

**See Also:** GLCURSOR, LLINE, RLINE

---

**PURPOSE:**

Changes the origin of coordinates for drawing with the pen.

**REMARKS:**

SORGN is effective only in the Graphics mode and is used to specify the current position of the pen as the new origin of coordinates.

When drawing a figure, it may not be easy for the printer to do so if the origin of coordinates is located at the left side of the paper. Move the pen to the desired position on the paper by executing GLCURSOR and then specify that position as the origin of coordinates with SORGN.

This will enable the figure to be drawn with the current position of the pen taken as a reference point.

**EXAMPLE:**

```
10: GRAPH
20: GLCURSOR (60, 40)
30: SORGN
```

Specifies the current position (X=60, Y=40) of the pen as the new origin of coordinates.

---

# STOP

---

P

**FORMAT:** STOP

**Abbreviation:** S.

**See Also:** CONT

---

**PURPOSE:**

Halts execution of a program for diagnostic purposes.

**REMARKS:**

When STOP is encountered in program execution, execution halts and a message such as "Break in 200" is displayed where 200 is the number of the line containing the STOP. STOP is used during the development of a program to check the flow of the program or to examine the state of variables. Execution may be restarted with the CONT command or **SHIFT** + **↓** keys. Pressing the **↓** key executes the program statement-by-statement.

**EXAMPLE:**

10: STOP

Causes "Break in 10" to appear on the display.



---

# STR\$

---

P  
D

**FORMAT:** STR\$ expression

**Abbreviation:** STR.

**See Also:** VAL

---

**PURPOSE:**

Converts numeric data into string data.

**REMARKS:**

The STR\$ function changes numeric data to a string. The string will be composed of the same digits as the original number. The STR\$ function has the opposite effect of the VAL function.

If the numeric data is negative, the string will be preceded by a minus (-) sign.

When a numerical value is converted into a character string using the STR\$ command, the first character is the sign (space for +) e.g. B\$=" 12.34084".

**EXAMPLE:**

```
:  
:  
:  
110: N=N*3  
120: A$=STR$ N  
130: B$=LEFT$ (A$,3)  
140: M=VAL B$  
:
```

- [110] Program performs calculations on numeric variable N.
- [120] The numeric variable N is converted to the string variable A\$. String variables are easier to manipulate than numerics. In this example, suppose that the first 3 digits of the number are required. Having converted the number to a string, we can use any of the string manipulation commands: LEFT\$, RIGHT\$, MID\$.
- [130] Stores only the first 3 digits (characters) of the number into string variable B\$.
- [140] The first 3 digits are reconverted into a numeric variable for processing by the program as a number.

---

# TEXT

---

D

**FORMAT:** TEXT 

**Abbreviation:** TE.

**See Also:** BASIC

---


**PURPOSE:**

Sets the Text mode.

**REMARKS:**

Valid only as direct input in the PRO mode.

The text function is used when entering a program written for a higher-level personal computer. The program entered using the computer can be sent to the host through the serial I/O interface.

Executing the TEXT command sets the Text mode. In the Text mode, a number corresponding to the line number, and then information corresponding to program commands or data is entered. Press the  key to write the entries to the program/data area.

The written contents are not converted to commands (internal codes), as they are in the BASIC mode. The text is stored as it is (as characters and/or numbers) in character codes. The text is arranged in the order of the numbers corresponding to the line number at the beginning of each line. (Line number editing function.)

The text written in the Text mode is stored as it is. Therefore, command abbreviations in BASIC (such as I. for INPUT) are displayed and stored as such.

If a program is stored in the internal code of the computer with the text mode set, it is converted to character code.

During program conversion, "✕" is displayed at the right end of the display unit. The prompt symbol is "<" in the Text mode. (It is usually ">".)

If a password has been set, an error occurs when the TEXT command is executed.

---

# TROFF

---

P  
D

**FORMAT:** TROFF

**Abbreviation:** TROF.

**See Also:** TRON

---

**PURPOSE:**

Cancels trace (TRON) mode.

**REMARKS:**

Execution of TROFF restores normal execution of the program.

**EXAMPLE:**

See TRON.

---

---

# TRON

---

P  
D

**FORMAT:** TRON

**Abbreviation:** TR.

**See Also:** TROFF

---

**PURPOSE:**

Starts the trace mode.

**REMARKS:**

The trace mode provides assistance in debugging programs. When the trace mode is on, the line number of each statement is displayed after each statement is executed. To stop trace execution, press the **BREAK** key or execute the STOP command. After trace execution is stopped, the computer waits for the down arrow key to be pressed before moving on to the next statement. The trace mode continues until TROFF is executed, or the **SHIFT** + **CA** keys are pressed.

**EXAMPLE:**

```
10: TRON
20: FOR I = 1 TO 3
30: NEXT I
40: TROFF
```

When run, this program displays the line numbers 10, 20, 30, 30, and 30.

---

# USING

---

P  
D

**FORMAT:** 1. USING format string  
2. USING

**Abbreviation:** U.

**See Also:** LPRINT, PAUSE, PRINT

---

**PURPOSE:**

Controls the format of displayed or printed output.

**REMARKS:**

USING can be used by itself or as a clause within a PRINT, LPRINT, or PAUSE statement. When the USING command is used in a PRINT, LPRINT, or PAUSE statement, it is valid only for the values or strings output by that statement. If it is used independently (on an independent line), it is valid for all the subsequent PRINT or LPRINT commands. USING establishes a specified format for output that is used for all output that follows until changed by another USING.

#: Right justified numeric field character.

Length of integer field: 2 to 21 (including sign)

If a value is shorter than the specified numeric field, the extra portion of the field is filled with spaces.

If a numeric field with a length of 22 or more digits is specified, it is regarded to be 21 digits long.

Length of decimal field: 0 to 20 (0 to 19 for exponential numbers)

If a value is shorter than the specified field, zeros appear in the extra portion of the field. If the former is longer than the latter, the extra digits are truncated.

.: Decimal point (delimiter for integer and decimal parts)

,: Used as a 3-digit separator in numeric fields.

To separate every 3 digits of integer field with commas (,), place a comma in or at the end of the integer field.

^: Used to indicate that numbers should be displayed in scientific notation.

With this notation, the length of the mantissa field is always 2 (1 digit and the sign), without regard to the specified length of the integer field. If the given length of the decimal field is 19 or more digits, the length of the decimal field of the mantissa is also 19 digits.

&: Left justified alphanumeric field

If a string is shorter than the specified field, spaces appear in the extra portion of the field. If the former is longer than the latter, the extra characters are dropped.

(1) USING"###"

Prints the sign and 2 integer digits.

(2) USING"###."

Prints the sign, 2 integer digits, and a decimal point.

- (3) USING"###.##"  
Prints the sign, 2 integer digits, a decimal point, and 2 decimal places.
- (4) USING"###,###."  
Prints the sign, 4 integer digits, a 3-digit separator (,) and a decimal point.  
For numerical data, 3-digit separator (,) is counted as a digit. So if you want to print a number "-1,234,567.", you have to use ten field characters (#), such as USING"#####,###."
- (5) USING"##.##^"  
Prints numerical data in exponential form with up to 2 decimal places.  
Spaces for 1 integer digit and the sign are automatically reserved for the mantissa, and for 2 integer digits, the capital E or D, and the sign for the exponent.  
Note: ^ and comma (,) may not be used concurrently.
- (6) USING"&&&&&&"  
Prints a string of 6 characters.
- (7) USING"###&&&&"  
Prints a string adjacent to a numeric value.
- (8) USING  
Format 2 clears formatting.

Formatting is also cleared by executing the RUN command, pressing **SHIFT** + **CA**, or turning the computer off and then on.

**EXAMPLE:**

10: B=-10:C=10.7703

20: PRINT USING "&&&###" ; "B=" ; B ; "\_C=" ; PRINT USING "###.###"; C

Note:

The USING command is not valid for manual calculations. It must always be used in the PRINT or LPRINT statement.

**Supplement:**

A program which simultaneously outputs numerical and string characters written for other computers should be modified as follows:

PRINT USING "#####.##" ; H ; "(m)"



PRINT USING "#####.##" ; H ; PRINT "(m)"

---

# VAL

---

P  
D

**FORMAT:** VAL string

**Abbreviation:** V.

**See Also:** STR\$

---

**PURPOSE:**

Converts a string of numeric characters into a decimal value.

**REMARKS:**

The VAL function converts a character string, which may include the hex number designator (&H), numbers (0–9), a sign (+, –), and exponential symbols (E or D), into a numeric value.

If the string is in decimal notation, it must be composed of the characters 0 to 9, with an optional decimal point and sign. In this form, VAL is the opposite of the STR\$ function.

If illegal characters are included, conversion is performed up to the first occurrence of an illegal character.

Control codes (&H00 to &H1F) cannot be used.

**EXAMPLE:**

A=VAL"-120"    Assigns -120 to variable A.  
B=VAL"3.2\*4="    Assigns 3.2 to variable B.  
C=VAL"&H64"    Assigns 100 to variable C.

---

# WAIT

---

P  
D

**FORMAT:** 1. WAIT expression  
2. WAIT

**Abbreviation:** W.

**See Also:** PRINT, GPRINT

---

**PURPOSE:**

Controls the length of time that displayed information is shown before program execution continues.

**REMARKS:**

Format 1 specifies the time in which execution of the PRINT command halts. The program temporarily halts for the specified time interval, then automatically restarts.

The value of the expression may be set to any value from 0 to 65535. A value of 1 as the expression corresponds to an interval of approx. 1/59 sec. The power on default for the value of the expression is zero.

The WAIT command is valid for all the PRINT or GPRINT commands used in the program. To set an infinite interval, use format 2.

**Note:**

The WAIT command is not available on personal computers in general. On PCs, the FOR...NEXT statement is used for wait time control as follows:

```
50: FOR J=1 TO 500:NEXT J
```

**EXAMPLE:**

```
10: WAIT 59
```

Causes PRINT to wait about 1 second.





## **APPENDICES**

Error Messages	A
Character Code Chart	B
Key Functions in BASIC	C
Troubleshooting	D
Signals Used in the Serial I/O Interface	E
Specifications	F
Using Programs from Other SHARP Computers	G
Care of the PC-E500	H

# APPENDIX A

## ERROR MESSAGES

When an error occurs, one of the error messages listed below will be displayed. For errors which occur during program execution, the error message is followed by the line number in which the error occurred. The error number and the line number are stored into the variables ERN and ERL, respectively. For errors which occur during direct input operation, the values of the variables ERN and ERL do not change.

Error message	Error No.	Meaning
Syntax error	10	Invalid expressions or statements have been used.
Direct command error	11	An attempt was made to execute a command which is illegal in direct input operation. An attempt was made to execute a command which is illegal in program execution.
Mode error	12	The mode for PRO or RUN was selected incorrectly. The mode selection in the OPEN statement was incorrect.
Can't continue	13	The CONT statement was executed illegally.
Program not exist	14	An attempt was made to designate a password to a program which does not exist.
Overflow	20	The calculated result exceeds the calculation range.
Division by Zero	21	An attempt was made to divide by zero.
Illegal function call	22	Illegal operation was attempted.
Duplicate Definition	30	An attempt was made to declare an array variable name which is already declared.
Array specified without DIM	31	The array variable name was specified without the DIM statement.
Subscript out of range	32	Array was addressed illegally (array subscript exceeds the size of the array specified in the DIM statement)
Data out of range	33	The specified value exceeds the allowable range.
Undefined line	40	The specified line number or label does not exist.
Illegal line number	41	The line number was specified illegally.
Bad line number	44	The ending line number was specified with a number less than the starting line number in a statement such as LLIST or DELETE.
GOSUB or FOR nesting exceeded	50	The levels of nesting in the GOSUB or FOR statement exceeds the allowable range.
RETURN without GOSUB	51	An attempt was made to execute the RETURN statement without calling the subroutine.
NEXT without FOR	52	The FOR statement is missing for the NEXT statement.

Error message	Error No.	Meaning
Out of data	53	The DATA statement is missing for the READ statement.
Buffer space exceeded	54	The size of the BASIC interpreter exceeds the available work area.
String too long	55	The length of the entered string exceeds 254 bytes.
Line buffer overflow	56	The line exceeds 254 bytes.
RESUME without error	57	An attempt was made to execute the RESUME statement during non-error processing.
Out of memory	60	The size of program or variable exceeds the memory capacity.
Can't print in specified format	70	Characters cannot be printed in the format specified in the USING statement.
USING format error	71	The format specified in the USING statement is illegal.
I/O error	72	I/O device error.
Too many files open	73	The number of files to be opened exceeds the limit.
NAME error	74	The file name specified in the NAME statement is illegal.
Bad drive name	75	The specified drive name is illegal.
File write protected	76	The file is write protected.
Disk full	77	No further memory storage available on the disk, or the number of files exceeds the limit.
Tape read error	80	An error occurred while reading data from the cassette tape recorder.
Verify error	82	Error in data verification.
Printer error	84	Printer error (The printer was turned off while printing, etc.)
File not open	85	The file has not been opened.
File already open	86	The file has already been opened.
Input past end	87	An attempt was made to read data past the end of file.
Type mismatch	90	The type of the specified data does not match.
Password mismatch	92	An invalid password was entered.
Invisible program	93	An attempt was made to write to a protected program.
File not found	94	The specified file does not exist.
Bad file name	95	The specified file name is illegal.

## APPENDIX B

### CHARACTER CODE CHART

The character code chart shows the characters and their character codes used by the CHR\$ and ASC commands. Each character code consists of 2 hex characters (or 8 binary bits). The most significant hex character (4 bits) is shown along the top of the chart and the least significant hex character (4 bits) is shown down the left side of the chart. If no character is shown, it is an illegal character on the computer. This character set is similar to the IBM PC character set.

For example, the character "A" is hex 41 or decimal 65 or binary 01000001. The character "P" is decimal 80 or hex 50 or binary 01010000.

The character codes are represented as follows:

**Examples:**

Code for \*

Hexadecimal &H2A

Decimal 42 (32 + 10)

Code for P

Hexadecimal &H50

Decimal 80

**Notes:**

- To print characters on the CE-515P, refer to the Character Code Table in the CE-515P Operation Manual and specify the character code.
- The characters for codes &H00-&H1F can be displayed only when assigned to the programmable function keys using the KEY command. (Do not assign the special characters for codes &H80-&H9F or &HE0-&HFF to the programmable function keys.)
- The characters for codes &H07-&H0D and &H1C-&H1F can be displayed only when the PRINT CHR\$ statement is executed.
- When printing on the CE-126P, the codes &H00-&H1F and &H7F-&HFF are spaces.

IBM is a registered trademark of International Business Machines Corporation.

Most Significant 4 Bits

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NL	DE	space	0	@	P	'	p	Ç	É	á	☐	☐	☐	α	≡
1	SH	D1	!	1	A	Q	a	q	ü	æ	í	☐	☐	☐	β	±
2	SX	D2	"	2	B	R	b	r	é	Æ	ó	☐	☐	☐	Γ	≥
3	EX	D3	#	3	C	S	c	s	â	ô	ú	☐	☐	☐	π	≤
4	ET	D4	\$	4	D	T	d	t	ä	ö	ñ	☐	☐	☐	Σ	∫
5	EQ	NK	%	5	E	U	e	u	à	ò	Ñ	☐	☐	☐	σ	∫
6	AK	SN	&	6	F	V	f	v	á	û	á	☐	☐	☐	μ	÷
7	BL	EB	'	7	G	W	g	w	ç	ù	ó	☐	☐	☐	τ	≈
8	BS	CN	(	8	H	X	h	x	ê	ÿ	¿	☐	☐	☐	φ	°
9	HT	EM	)	9	I	Y	i	y	ë	Ö	—	☐	☐	☐	θ	•
A	LF	SB	*	:	J	Z	j	z	è	Ü	—	☐	☐	☐	Ω	•
B	HM	EC	+	;	K	I	k	{	ï	ç	½	☐	☐	☐	δ	√
C	CL	→	,	<	L	\	l	:	î	£	¼	☐	☐	☐	∞	n
D	CR	←	—	=	M		m	}	ï	¥	ı	☐	☐	☐	φ	²
E	SO	↑	.	>	N	^	n	~	Ä	Pt	«	☐	☐	☐	€	▪
F	SI	↓	/	?	O	_	o	△	À	ƒ	»	☐	☐	☐	∩	

Least Significant 4 Bits

## APPENDIX C

### KEY FUNCTIONS IN BASIC

**BREAK**  
**ON**

- Use to turn the power on when the power has been turned off by the Auto OFF function.
- Pressing this key during program execution functions as a **BREAK** key and causes program execution to be interrupted.
- When pressed during direct input operation, execution of BEEP or CLOAD is interrupted.

**SHIFT**  
**2nd F**

- The yellow keys marked "SHIFT" and "2nd F" must be pressed (and held for "SHIFT") to use a key's second function (indicated immediately above the key).

e.g.: **SHIFT** +  $\frac{f}{Y}$  → & is entered.  
( **2nd F**  $\frac{f}{Y}$  )

**C•CE**

- Use to clear the contents of the entry and the display.
- Use to reset after an error.

**SHIFT** + **CA**

- Not only clears the display contents, but resets the computer to its initial state.
  - Resets the WAIT timer.
  - Resets the display format. (USING format)
  - Resets the TRON state (TROFF).
  - Resets PRINT=LPRINT.
  - Resets the error condition.

**MENU**

- Use to display the Main Menu.

**BASIC**

- Use to change the operational submenu selection from RUN to PRO or from PRO to RUN.

**SHIFT** + **AER**

- Use to select the Algebraic Expression Reserve (AER) mode.

**SHIFT** + **CAL**

- Use to select the CAL mode.

**TITLE**

- Use to recall the stored algebraic expressions.

**0** -- **9**

- Numeric keys

**CTRL** + **0**

- Use to toggle the beep sound for key entry.

**CTRL** + **◆**

- Use to toggle the PF key label display.

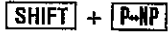
**CTRL** + **E**

- Use to delete the text from the current cursor position to the end of the line.

- CTRL** + **U**
- Use to delete the whole line on which the current cursor is located.
  - For control combinations, press and hold the **CTRL** key, and then the second key.
- SHIFT** + **^**
- Use for power calculations.
  - Use to specify scientific notation for numerical data in USING statements.
- SHIFT** + **<**  
**SHIFT** + **>**
- Use when entering logical operations in IF statements.
  - Use to specify relational expressions.
- SHIFT** + **?**
- Use to enter a question mark.
  - Use to specify the CLOAD? command.
- ;**
- Use to provide multi-display (two or more values displayed at a time).
  - Use to separate commands and variables.
- SHIFT** + **:**
- Use to separate more than one statement defined on a single line.
- ,**
- Use to provide multi-dipslay (two or more values displayed at a time).
  - Use as 3-digit separator in the USING statement.
- ▶**
- Shifts the cursor to the right.
  - Executes playback instructions.
  - Call the cursor if not displayed while the contents are displayed.
  - Clears an error in direct input operation.
- ◀**
- Shifts the cursor to the left.
  - Otherwise the same as the **▶** key.
- SHIFT** + **▶**
- Shifts the cursor to the end of the current row or entry data if the cursor is on the screen.
- SHIFT** + **◀**
- Shifts the cursor to the beginning of the current row or of the entry data if the cursor is on the screen.
- INS**
- Toggles the insert mode on and off.
- DEL**
- Deletes the character at the cursor position.
- BS**
- Deletes the character to the left of the cursor.



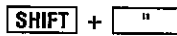
- Enters a program line into the computer.
- Use when writing programs.
- Requests manual calculation or direct execution of a command statement by the computer.
- Use to restart a program temporarily stopped by an INPUT command.



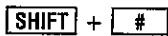
- Use to set print and non-print mode when an optional printer is connected.



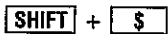
- Toggles the uppercase mode and the CAPS symbol in the display.
- When the CAPS symbol is displayed, uppercase letters are entered. If CAPS is pressed, the CAPS symbol disappears and lowercase characters are entered.



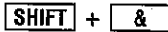
- Use to specify the beginning and end of strings and labels.





- Use with USING statement to define a numeric field.
- Use to specify the PRINT# or INPUT# statement.



- Use when assigning character variables.



- Use with the USING statement, to define character string fields.
- Use to indicate a hexadecimal value.

The  and  keys have the following functions, depending on the mode, as well as the state of the computer, as listed in the following table:



Mode	State	↓	↑
RUN	Program being executed	Not functional	
	Interrupted by the STOP command or the BREAK key	Execute the next line and stop. Press SHIFT + ↓ to execute subsequent lines continuously.	Hold down to display program line being executed or already executed.
	Error condition during program execution	Not functional	Hold down to display error-producing line
PRO	(When the mode is changed to PRO mode and program lines are not displayed)		
	Program is temporarily interrupted	Display the line interrupted.	Same as left.
	Error condition	To display the line with error.	Same as left
	Other condition	To display the first line.	To display the last line.
	(When the program lines are displayed)		
	To display the next program line.	To display the preceding program line.	

**Note:**

- If no key is entered in the key entry request mode for approximately 11 minutes, the power is automatically turned off (Auto OFF function.)

## APPENDIX D

### TROUBLESHOOTING

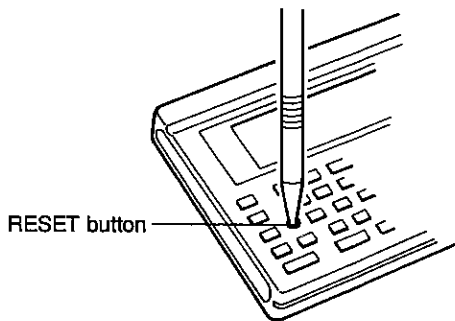
This appendix provides you with some hints on what to do when your computer does not do what you expect it to. You should try each of the following suggestions, one at a time, until you have corrected the problem.

1. If the display is too light or too dark,
  - adjust the contrast control.
2. If the power does not come on (nothing is displayed),
  - the batteries may be exhausted. Replace the batteries.
  - the memory protect switch may be set to position B. Check that the switch is set to position A.
  - the RAM slot cover lock might not be set to the LOCK position. Check that the lock is set to the LOCK position.
3. If the power does not turn off,
  - the computer is running a program using a command which takes a long time, such as BEEP, CSAVE or CLOAD. Press the **BREAK** key to interrupt program execution and then the **OFF** key.If the power is still not turned off, perform the following operation 4.
4. If the computer does not operate properly,
  - a peripheral device may have been connected or disconnected while the power was on, or there may have been an error during program execution, or the computer may have been subjected to strong electrical noise or shocks during use.

Perform one of the following operations.

- (1) **Reset** (retaining the memory contents)

Press the RESET button with a ball-point pen or any other appropriate device and then the **OFF** **ON** keys. If the computer still operates improperly after clearing the error condition with this operation, there may be an error in programs or data entered. Follow (2) below, pressing the **Y** key to clear the memory contents.



Use only a ball-point pen or similar device to press the RESET button. Do not use a mechanical pencil with its lead exposed or a device with a sharp point, such as a sewing needle.

- (2) **All Reset** (clearing all the memory contents)

Press the RESET button while holding the **ON** key.

Release the RESET button before releasing the **ON** key.

- When using the computer only:

```
S1(MAIN):  
ALL CLEAR OK? (Y/N)
```

If the  Y key is pressed, all the memory is cleared.

If the  N key is pressed, the memory contents remain intact.

- When the RAM card is installed:

```
S2(CARD):  
ALL CLEAR OK? (Y/N)
```

The display prompts for RAM card initialization. Press the  Y key to initialize the RAM card. Then the display prompts for computer initialization. Press the  Y key to initialize the computer.

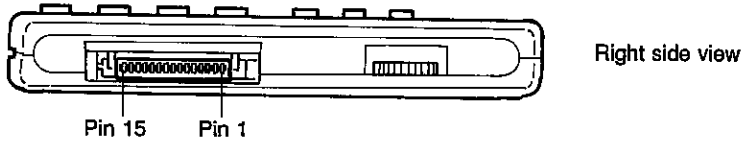
If the RAM card is installed and MEM\$="B" is used, the message using the computer only will be displayed. Press the  Y key to initialize the computer and the RAM card. Press the  N key not to initialize them.

The Auto OFF function will take effect in approximately one minute for the above displays.

## APPENDIX E

### SIGNALS USED IN THE SERIAL I/O INTERFACE

The computer is equipped with a 15-pin connector for the serial I/O interface. The pins used and their signals are described below.



#### Pin Connections Used

Pin	Name	Symbol	I/O	Function
1	Frame Ground	FG	—	Protective chassis ground
2	Send Data	SD	Out	Outputs a DC data signal
3	Receive Data	RD	In	Inputs a DC data signal
4	Request to Send	RS	Out	HIGH: Sends carrier
5	Clear to Send	CS	In	HIGH: Transmission enabled
7	Signal Ground	SG	—	Reference 0 voltage for all signals
8	Data Carrier Detect	CD	In	HIGH: Carrier signal received
10		VC	—	Power supply
11	Receive Ready	RR	Out	HIGH: Receive enabled
13		VC	—	Power supply
14	Data Terminal Ready	ER	Out	HIGH: Local terminal ready

#### Notes:

1. HIGH: VC voltage level; LOW: SG voltage level
2. The computer uses CMOS components. Application of voltages exceeding the allowable range, i.e., voltage level between SG and VC, may damage the computer.

# APPENDIX F

## SPECIFICATIONS

Model:	PC-E500 Pocket Computer	
Processor:	8-bit CMOS CPU	
Programming language:	BASIC	
System ROM:	256 K bytes	
Memory capacity:	System internal	3.8K bytes approx.
	Fixed variable area	312 bytes
	Program/data area	28600 bytes
Stack:	Total:	145 bytes
	( Subroutine: 4 bytes/stack )	
	( FOR-NEXT: 21 bytes/stack )	
Operators:	Addition, subtraction, multiplication, division, trigonometric and inverse trigonometric functions, logarithmic and exponential functions, angle conversion, square and square root, power, sign, absolute, integer, coordinate conversion, pi, etc.	
Numeric precision:	10 digits (mantissa) + 2 digits (exponent) single-precision mode 20 digits (mantissa) + 2 digits (exponent) double-precision mode In the CAL, MATRIX and STAT modes, only the single-precision mode can be used.	
Editing features:	Cursor left and right, line up and down, character insert, character delete	
Memory protection:	Battery backup	
Interface capability:	11 pin (for cassette interface, disk, printer, etc.)	
Serial input/output features:		
Standards:	Start-stop transmission (asynchronous) system Half/full duplex	
Baud rates:	300, 600, 1200, 2400, 4800, 9600 baud (bps)	
Parity bits:	Even, odd, or no parity	
Data bits:	7 or 8 bits	
Stop bit:	1 or 2 bits	
Connectors used:	15-pin connector (for external equipment)	
Output signal level:	CMOS level (4 to 6 volts)	
Interfacing signals:	Inputs: RD, CS, CD Outputs: SD, RS, RR, ER Others: SG, FG, VC	

Display:	4-line, 40-column liquid crystal display with 5 × 7 dot matrix.
Keys:	89 keys Alphabetic, numeric, special symbols, and functions Numeric keypad Programmable function keys
Power supply:	For computer operation: 6.0 Vdc Type-AAA dry cell battery (R03) × 4 For memory backup: 3.0 Vdc Lithium battery (CR2016) × 1
Power consumption:	0.07 W at 6.0 Vdc Approximately 70 hours of continuous operation under normal conditions (based on 10 minutes of operation or program execution and 50 minutes of display per hour at a temperature of 20°C/68°F). The operating time may vary slightly depending on usage and the type of battery used.
Operating temperature:	0° – 40°C (32° – 104°F)
Dimensions:	200(W) × 100(D) × 14(H) mm 7-7/8"(W) × 3-15/16"(D) × 9/16"(H)
Weight:	250 g (0.55 lb.) (with batteries)
Accessories:	Hard cover, four dry batteries, one lithium battery, and Operation Manual.
Options:	Plug-in RAM cards 8KB (CE-212M), 16KB (CE-2H16M), 32KB(CE-2H32M), 64KB (CE-2H64M) Printer/Cassette Interface (CE-126P) Printer (CE-515P) Pocket Disk Drive (CE-140F) Others

## APPENDIX G

### USING PROGRAMS FROM OTHER SHARP COMPUTERS

Programs written for the following SHARP PC series computers can be run on the PC-E500 computer with slight modifications:

PC-1475, PC-1401, PC-1402, PC-1403(H), PC-1450, PC-1460, PC-1425, PC-1360

The PC-E500's WAIT command has an initial default value of zero. This means that the display will not be temporarily frozen when the PRINT command is executed. If you wish to display more than three lines at a time, insert a WAIT command in the transplanted program to temporarily freeze the display (see "WAIT Command"). Since the PC-E500 has 40 display columns, the transplanted program may require modifications to fit the PC-E500's display width. Programs containing commands or characters not defined on the PC-E500 also require modification.

- **Memory Capacity**

A program written for another computer requires a different amount of memory on the PC-E500.

- **Variable Names**

The PC-E500 allows you to use up to 40 characters for a variable name, whereas existing models allow only up to 2 characters.

- **User-Defined Keys**

The PC-E500 has no user-definable keys for program execution. Use programmable function keys or enter GOTO\*label [←]. Use \*label rather than "label" in the GOTO statement.

- **Line Numbers**

Variables or expressions cannot be used for line numbers specified in the GOTO, GOSUB, RESTORE, or THEN statement.

- **Tape Recorder**

When loading another computer's program into the PC-E500, use the following command:

CLOAD@"filename" [←] or CLOAD@[←]

If the filename is omitted, the program first encountered after the tape is started is loaded. The PC-E500 automatically translates loaded program codes into PC-E500 program codes as it reads each line. An error occurs if a translated program line exceeds 255 bytes. An error also occurs if the size of the loaded program is too large for the PC-E500's program area. If this happens, clear unnecessary variables from the program area, then reload the same program. For code translation, a work area of approx. 600 bytes is required in addition to the program space.

When inputting/outputting data from/to another computer, the OPEN command must be executed.

**Example:**

```
20 PRINT #F,G → 10 OPEN "CAS:" FOR OUTPUT AS #1
                20 PRINT #1,F,G
                30 CLOSE #1
```

- **Array Variables**

1. If array A( ) is used in another computer's program without being declared with a DIM statement, declaration of the array is required at the beginning of the program. Fixed variable space is not shared with array variable A( ).
2. If the number of characters is declared for a string array, delete that declaration.

**Example:**

```
DIM AB$(30)*40 → DIM AB$(30)
The PC-E500 stores up to 254 characters.
```

- **Miscellaneous**

1. For logical operations, the PC-E500 returns value -1 for true, and 0 for false.
2. It is not allowed to use expressions in the DATA statement, such as DATA SIN 30+2, CHR\$66. Use the statement in the form of, for example, DATA 2.5, B.



## **APPENDIX H**

### **CARE OF THE PC-E500**

To ensure trouble-free operation of your computer, note the following:

- Always handle the computer carefully, as the liquid crystal display is made of glass.
- Keep the computer away from extreme temperature changes, moisture, and dust. During warm weather, vehicles left in direct sunlight are subject to high temperature buildup. Prolonged exposure to high temperature may damage your computer.
- Use only a soft, dry cloth to clean the computer. Do not use solvents, water, or wet cloths.
- To avoid battery leakage, remove the batteries when the computer will not be in use for an extended period of time.
- If the computer is subjected to strong static electricity or external noise, it may "hang up" (all keys become inoperative). If this happens, press the RESET button. (See TROUBLESHOOTING.)
- Keep this manual for future reference.

## COMMAND INDEX

### General Commands

AER .....	211
ARUN .....	212
ASC .....	213
AUTO .....	214
AUTOGOTO .....	215
BASIC .....	216
BEEP .....	217
CHR\$ .....	220
CLEAR .....	223
CLS .....	227
CONT .....	229
DATA .....	235
DEFDBL .....	236
DEFSNG .....	237
DEGREE .....	238
DELETE .....	239
DIM .....	240
END .....	243
ERASE .....	245
ERL .....	246
ERN .....	247
FOR...NEXT .....	250
FRE .....	251
GCURSOR .....	252
GOSUB...RETURN .....	254
GOTO .....	255
GPRINT .....	256
GRAD .....	258
HEX\$ .....	260
IF...THEN...ELSE .....	261
INKEY\$ .....	264
INPUT .....	266
INPUT\$ .....	267
KEY .....	272
KEY\$ .....	273
LEFT\$ .....	275
LEN .....	276
LET .....	277
LINE .....	280
LIST .....	282
LOCATE .....	290
MDF .....	294
MEM\$ .....	295
MID\$ .....	297
NEW .....	299
ON ERROR GOTO .....	300

ON...GOSUB .....	301
ON...GOTO .....	302
PASS .....	307
PAUSE .....	308
POINT .....	309
PRESET .....	310
PRINT .....	311
PSET .....	315
RADIAN .....	316
RANDOMIZE .....	317
READ .....	318
REM ( ' ) .....	319
RENUM .....	320
RESTORE .....	321
RESUME .....	322
RIGHT\$ .....	323
RND .....	325
RUN .....	326
STOP .....	330
STR\$ .....	331
TEXT .....	332
TROFF .....	333
TRON .....	333
USING .....	334
VAL .....	336
WAIT .....	337

### Printer Commands

CIRCLE .....	221
CLOSE .....	226
COLOR .....	228
CONSOLE .....	228
CROTATE .....	232
CSIZE .....	234
GLCURSOR .....	253
GRAPH .....	259
LF .....	278
LFILES .....	279
LLINE .....	283
LLIST .....	285
LPRINT .....	292
LTEXT .....	293
OPEN .....	303
PAINT .....	306
RLINE .....	324
SORGN .....	329

## Disk Commands

CHAIN .....	219
CLOSE .....	226
COPY .....	230
DSKF .....	242
EOF .....	244
FILES .....	248
INIT .....	263
INPUT\$ (Not for pocket disk) .....	267
INPUT# .....	270
KILL .....	274
LFILES .....	279
LOAD .....	287
LOAD? .....	288
LOC .....	289
LOF .....	291
MERGE .....	296
NAME .....	298
OPEN .....	303
PRINT# .....	313
SAVE .....	327
SET .....	328

MERGE .....	296
OPEN .....	303
OPEN\$ .....	305
PRINT# .....	313
SAVE .....	327

## Cassette Tape Recorder Commands

CHAIN .....	219
CLOAD .....	224
CLOAD? .....	225
CLOSE .....	226
COPY .....	230
CSAVE .....	233
EOF .....	244
INPUT\$ .....	267
INPUT# .....	270
LOAD .....	287
LOAD? .....	288
MERGE .....	296
OPEN .....	303
PRINT# .....	313
SAVE .....	327

## Serial I/O Commands

CHAIN .....	219
CLOSE .....	226
COPY .....	230
EOF .....	244
INPUT\$ .....	267
INPUT# .....	270
LOAD .....	287
LOAD? .....	288
LOC .....	289
LOF .....	291

# INDEX

## A

AER mode 32, 117  
All reset 2, 348  
Amino acids formulas 95  
Area of figure 76  
Array variables 169  
Atmospheric pressure 93  
Auto OFF 11

## B

BASIC  
  commands 181  
  concepts and terms 166  
  mode 32  
  program operation 165  
  statements 180  
Basic operations 37  
BATT indicator 13  
Batteries  
  handling 17  
  operating 9  
  memory backup 9  
  replacement 14  
BUSY indicator 12

## C

CAL mode 10, 32, 33  
Calculations  
  double-precision 156  
  errors 163  
  hexadecimal 46  
  length 158  
  matrix 138  
  ranges 207  
  regression 127  
  scientific 39, 158  
  serial 154  
  single-precision 156  
  statistical 123  
CAPS indicator 12  
Care of computer 355  
Cassette recorder  
  interface 24, 25  
  specifications 25  
  use 25

Character codes 342  
Chi-square distribution 110  
Color plotter/printer 25  
  use 28  
Complex numbers 97  
Configuration of memory 21  
Connector  
  11-pin 8  
  15-pin 9  
Constants  
  physical 79  
  planetary 94  
  string 166  
  solar 94  
Contrast 13  
Conversions  
  angle/time 41, 159  
  hexadecimal/decimal 45  
  metric 82  
  polar/rectangular 41, 160  
Cubic equation 54  
Cursor 12

## D

Data files 174, 189  
Data table 131  
DBL indicator 12  
Debugging 193  
Decimal places 35, 42  
DEFDBL 171  
DEFSNG 171  
DEG indicator 12  
Degree 158  
Device name 175  
Devices, peripheral 24  
Differential equations 62  
Digits, number of 35  
Direct command 182  
Direct input 31  
Direct calculation 161  
Display 8, 12  
Display mode 35  
Distribution  
  chi-square 110  
  F 111

- normal 108
- t 109
- Double-precision 156
- Double-precision mode 156
- Double-precision variables 171

## E

- E indicator 13
- Editor, PF key label 112
- Electric & magnetic fields 101
- Electrical formulas 99
- Electron, outer 91
- Elements, periodic table 87
- ENG mode 32
- Engineer software 32, 49
  - list 51
  - program creation 115
- Equation of motion 85
- Error messages 122, 137, 153, 340
- Errors 153, 163
- Expressions 176
  - AER mode 117
  - deleting 119
  - entry 117
  - logical 177
  - recalling 119, 121
  - relational 177
  - searching 122
  - string 176
  - title 117
- Extensions, of file names 174

## F

- F distribution 111
- Factorization 65
- Features i
- Figure, area 76
- Filenames 174
- File numbers 175
- Files
  - data 176, 189
  - number of 175
  - program 174
- First-order differential equations 62
- Fixed variables 168
- Formulas
  - amino acids 95
  - electric & magnetic fields 101
  - electrical 99

- integration 69
- Laplace transforms 103
  - mechanical 106
  - trigonometric 67
- Free memory (FRE) 3

## G

- Gamma function 78
- GRAD indicator 12
- Gradient 158
- Graph
  - data 73
  - function 71
- Greatest common measure 53
- Greek alphabet 70

## H

- Hardware 8
- Hexadecimal numbers 166
  - conversion 45
  - calculations 46
- HYP indicator 13

## I

- I/O interface signals 350
- Initializing
  - memory 2
  - RAM cards 19
- Integration 60
- Integration formulas 69
- Isotope, stable 92

## K

- Key functions 344
- Key label editor 112
- Keyboard 8

## L

- Labels 181
- Lagrange's method 64
- Last answer recall 156
- Laplace transformation 103
- LCD screen 8, 12
  - contrast dial 9, 13
- Least common multiple 53
- Line numbers 180
- Linear equation solution 145
- Logical operators 177

## M

Magnetic fields 101  
Main menu 10  
MATRIX mode 32, 138  
Matrix  
  calculations 141  
  entry 138  
  errors 148  
  printing 148  
  scalar operations 144  
Mechanical formulas 106  
Memory  
  available 3  
  calculations using 38  
  configuration 22  
Menu labels 112  
Meteorology 93  
Method of bisection 58  
Metric conversion 82  
Motion, equations of 85

## N

Names  
  device 175  
  file 174  
Newton's method 56  
Normal distribution 108  
Numeric operators 37  
Numerical integration 60  
Numerical solution  
  Bisection 58  
  Newton's method 56

## O

Operation modes 32  
  selection 32  
Operator precedence 43, 162, 179  
Operators  
  logical 177  
  matrix 142  
  numeric 37  
  relational 161, 177  
Outer electron 91

## P

Parentheses 42, 44, 179  
Periodic table 87  
Peripheral devices 24  
Physical constants 79

Planetary constants 94

## Plot

  data 73  
  function 71  
Plotter/printer 25  
  use 28  
Pocket disk drive 24  
Polynomial, Lagrange 64  
Precedence 43, 162, 179  
Prime factors 53  
PRINT indicator 13  
Printer/cassette interface 24  
  use 27  
Printing, direct input 162  
Priority levels 43, 162, 179  
PRO mode 32  
  indicator 13  
Program  
  entering 182  
  execution 192  
  files 174  
  from other computers 353  
  listing 183, 186  
  storing 188  
Programmable function keys 8, 49, 191  
  label editor 112  
Programming 180, 182  
Prompt 12  
Protective cover 6

## R

RAD indicator 12  
Radians 158  
RAM card 18  
  capacities 21  
  configuration 22  
  installing 18  
  number of files 18  
  slot 9  
  use 21  
RAM disk files 174  
Relational operators 161, 177  
Resetting  
  All reset 2, 348  
  button 9  
  memory reset 2  
Romberg's method 60  
RUN mode 10, 32, 149  
  calculations in 150

indicator 13  
selection 149  
Runge-Kutta method 62

simple string 168  
storage 136, 147  
string array 168, 169  
using in calculations 155

## S

Scientific calculations 39, 158  
Scientific notation 35  
Second function key 11  
Sequential file 189  
Serial calculations 154  
Serial I/O 29  
Shift key 11  
Simultaneous linear equations 145  
Single-precision 156  
Single-precision mode 156  
Single-precision variables 171  
Single variable statistics 124  
Solar constants 94  
Specifications 351  
Stable isotope 92  
STAT mode 32, 123  
Statistics  
    calculations 123  
    data entry 124, 128  
    errors 137  
    FREQ key 125  
    regression 127, 131  
    single-variable 124  
Status line 12  
Storing programs 188

## T

t distribution 109  
Tape  
    recorder interface 25  
    recording 25  
Trace mode 193  
Trigonometric calculations 39, 158  
Trigonometric formulas 67  
Troubleshooting 348  
Turning ON 10

## V

Variables 167  
    double-precision 171  
    fixed numeric 168  
    numeric array 168, 169  
    in programs 185  
    single-precision 171

## W

Wind speed 93

== MEMO ==





# SHARP CORPORATION

OSAKA, JAPAN

© 1989 SHARP CORPORATION  
PRINTED IN JAPAN/IMPRIMÉ AU JAPON

9G1KS(TINSE1189ECZZ)©