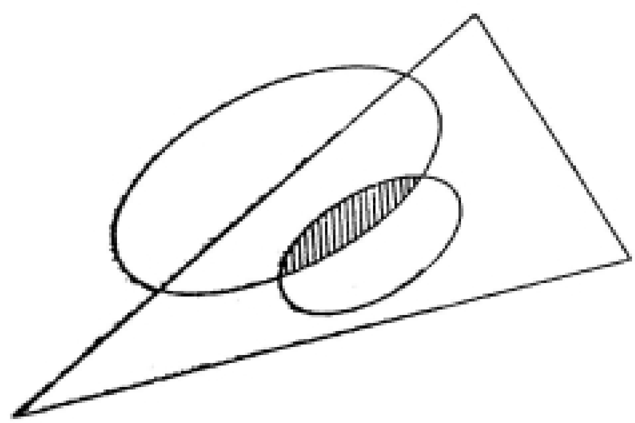
CASIO

PERSONAL COMPUTER

SCIENTIFIC LIBARY 16

FX-850P/FX-880P



OWNER'S MANUAL

FOREWORD

Thank you very much for purchasing the CASIO FX-850P Personal Computer. This manual introduces and explains the scientific calculation function and BASIC programming language used with this computer. It is suggested that everyone from BASIC novices to veterans become familiar with the name and function of each part of the computer before attempting operation. Even when BASIC programming is not employed, a Formula Storage Function provides simplified formula calculations and ratio calculations, a built-in Data Bank Function allows memo handling and searches.
Besides this, this computer also features a Built-in Scientific Utility which provides a total of 116 software utilities for statistical, mathematical and scientific applications.

PRECAUTIONS

This computer is a product of CASIO's high level of electronics engineering, testing, and quality control. The following points should be carefully noted to allow this unit to provide the years of trouble free operation for which it is designed.

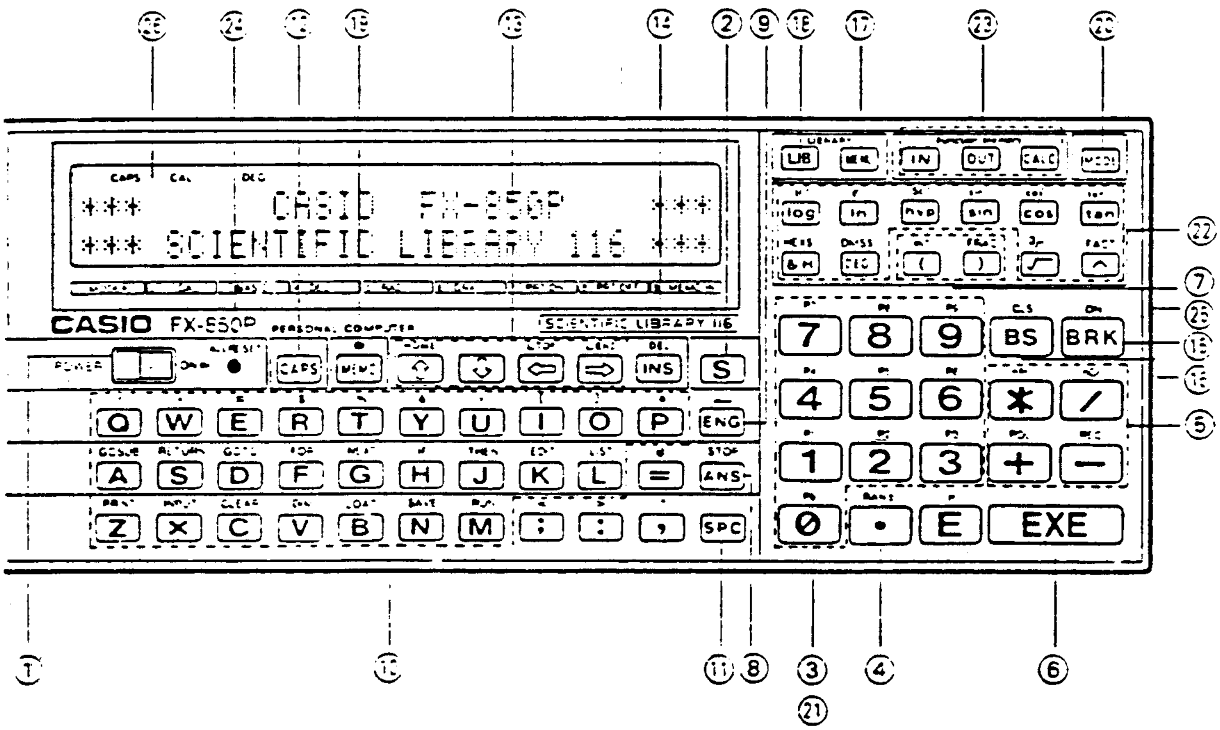
- This unit is constructed of precision electronic components and should never be disassembled, dropped, or otherwise subjected to strong impact. Strong shocks can cause termination of program execution or alteration of the unit's memory.
- Do not use or store this unit in areas subjected to high temperatures, humidity or dust.
- Display response may become slow or fail completely at extremely low temperatures. Normal operation should resume after the unit reaches normal temperature.
- The connectors of this unit are designed exclusively for connection of the specified FA-6 expansion units only.
- The display may become dim when the buzzer sounds, but this does not indicate malfunction and is no cause for worry.
- Batteries should be replaced as soon as possible after weakened batteries are indicated by a dim display during normal operation.
- Replace batteries at least once every two years even if the unit is not used during this period. Dead batteries left in the unit may cause serious damage due to fluid leakage and should be removed as soon as possible.
- Keep the connector of the unit covered with the connector cap whenever the unit is not connected to an expansion unit, and avoid touching the connector.
- Strong static electrical charge may cause alteration of memory contents or key operation failure. If this situation should occur, remove the batteries and load them again.
- Always ensure that the power supply of this unit is switched OFF before connecting peripheral devices.
- Never use thinner, benzine, or other volatile agents for cleaning the exterior of the unit. Use a soft cloth dipped into a mild solution of water and a neutral detergent, and wring the cloth out completely.
- Do not switch the power of the unit OFF during program execution or during calculations.
- When a malfunction occurs, contact the store where the computer was purchased or a nearby dealer.
- Before seeking service, please read this manual again, check the power supply, check the program for logic errors, etc.

CONTENTS

PART 1

UNIT CONFIGURATION

1-1 GENERAL GUIDE



1. Power Switch
2. Shift Key
3. Numeric Keys
4. Decimal Key
5. Arithmetic Operator Keys
6. Execute Key
7. Parentheses Keys
8. Answer Key
9. Engineering Key
10. Alphabet Keys
11. Space Key
12. CAPS Key
13. Cursor Keys
14. Insert/Delete Key
15. Break Key
16. Backspace/Clear Screen Key
17. Menu Key
18. LIB Key
19. Memo Key
20. Mode Key
21. Program Area Keys
22. Function Keys
23. Formula Storage Key
24. ALL RESET Button
25. P Button
26. Screen

1-2 OPERATIONAL FUNCTIONS

① **Power Switch (—·▯▮··)**
Slides to the right to switch power ON and to the left to switch power OFF.

② **Shift Key ([S])**
Used to enter BASIC commands and symbols noted in red on the key panel. Each press of this key causes the symbol " [S] " to switch ON and OFF on the display.
* Throughout this manual, this key is represented by [SHIFT] in order to distinguish it from the alphabetic [S] key.

③ **Numeric Keys ([0] ~ [9])**
Enter the numeric values noted on each key.

④ **Decimal Key ([·])**
Enters a decimal point.

⑤ **Arithmetic Operator Keys ([+], [−], [*], [/])**
Enter the arithmetic operators noted on the keys.
[+] : Addition
[−] : Subtraction
[*] : Multiplication
[/] : Division

⑥ **Execute Key ([EXE])**
Finalizes entry of a calculation and produces the result. The function of this key is equivalent to a "=" key on a standard calculator.
This key is also used to enter lines of a program and for actual execution of programs.

⑦ **Parentheses Keys ([(] [)])**
Enter parentheses in such parenthetical calculations as: 5 × (10 + 20).

⑧ **Answer Key ([ANS])**
Recalls the result of the most recently performed manual or program calculation. Pressing this key during program execution causes the execution to be suspended until the [EXE] key is pressed (STOP displayed).

⑨ **Engineering Key ([ENG] / ⌓)**
Converts a calculation result to an exponential display.

⑩ **Alphabet Keys**
Enter the alphabetic characters noted on each key.

⑪ **Space Key ([SPC])**
Enters a space.

⑫ **CAPS Key ([CAPS])**
Switches the alphabet keys between upper case and lower case characters. The upper case mode is indicated by the "CAPS" symbol on the display.

⑬ **Cursor Keys ([⇦], [⇨], [⇧], [⇩])**
Move the cursor on the screen. Each press moves the cursor in the direction noted on the keys pressed, while holding down the keys causes continuous, high speed movement. Each cursor key also takes on a different function when pressed in combination with the [SHIFT] key.

KEY	FUNCTION	[SHIFT] +
[⇦]	Cursor left	Moves to beginning of logical line
[⇨]	Cursor right	Moves to end of logical line
[⇧]	Cursor up	Scrolls screen up without cursor movement
[⇩]	Cursor down	Scrolls screen down without cursor movement

⑭ **Insert/Delete Key ([INS] / [DEL])**
Inserts a space at the current cursor position by shifting everything from the cursor position right one space to the right. In combination with the [SHIFT] key, deletes the character at the current cursor position and automatically fills in the space created by shifting everything to the right of the cursor one space to the left. Holding down this key for either function causes continuous high speed operation of the respective function.

⑮ **Break Key ([BRK])**
Terminates manual operations, program execution, printer output, and LIST output. Also reactivates the power supply when it has been interrupted by the Auto Power OFF function (see page 10).

⑯ **Backspace/Clear Screen Key ([BS] / [CLS])**
Deletes the character located immediately to the left of the cursor and automatically fills in the space created by shifting everything from the cursor position right one space to the left. In combination with the [SHIFT] key, clears the contents of the screen and locates the cursor at the upper left corner of the screen.

⑰ **Menu Key ([MENU])**
Only operable in the CAL mode, this key is used to display a menu of built-in scientific library. See PART 11 SCIENTIFIC LIBRARY for details.

⑱ **LIB Key ([LIB])**
Only operable in the CAL mode, this key executes the operation corresponding to an entered library number. See PART 11 SCIENTIFIC LIBRARY for details.

⑲ **Memo Key ([MEMO])**
Used to display and search for DATA BANK data. See PART 5 DATA BANK FUNCTION for details.

⑳ **Mode Key ([MODE])**
Used in combination with numeric keys to specify operational modes.
[MODE] [0] .... CAL mode (selected when power is switched ON)
[MODE] [1] .... BASIC mode (program writing/editing)
[MODE] [4] .... DEG mode (angle unit = degrees)
[MODE] [5] .... RAD mode (angle unit = radians)
[MODE] [6] .... GRA mode (angle unit = grads)
[MODE] [7] .... Print ON
[MODE] [8] .... Print OFF
[MODE] [9] .... MEMO IN mode (DATA BANK function)

㉑ **Program Area Keys ([SHIFT] [P0] – [P9])**
Executes the program in the corresponding program area in the CAL mode. Specifies a program area for writing or editing in the BASIC mode.

㉒ **Function Keys ([log] , [ln] , [sin] , etc.)**
Allow one-touch entry of often used functions.

• Direct input functions
[log] , [ln] , [exp] , [sin] , [cos] , [tan] , [&H], [DEG] , [√] , [^]

• [SHIFT] functions
[10^x] , [e^x] , [SET] , [sin⁻¹] , [cos⁻¹] , [tan⁻¹] , [HEX] , [DMS] , [INT] , [FRAC] , [∛] , [FACT] , [nPr] , [nCr] , [POL] , [REC] , [RAN#] , [π]

㉓ **Formula Storage Keys ([IN] , [OUT] , [CALC])**
Used when working with the formula storage function. See PART 4 FORMULA STORAGE FUNCTION for details.

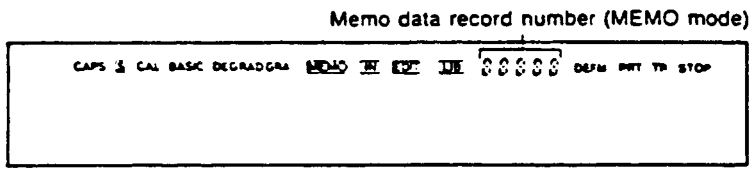
㉔ **ALL RESET Button (ALL RESET •)**
Clears all memory contents and enters the CAL mode. All important data should be saved elsewhere before pressing this button. If pressing this button does not clear memory contents, first press the P button and then press the ALL RESET button again.

(25) **P Button ($\overset{\bullet}{\text{P}}$) (rear panel)**
Hardware reset button to halt misoperation caused by static electricity. Though execution is interrupted, memory contents are retained. The ALL RESET button should be used when the misoperation damages memory contents. Note that power switches OFF and then ON again when the P button is pressed.

(26) **Screen**
A 32-column × 2-line liquid crystal display upon which 5 × 7-dot characters appear.

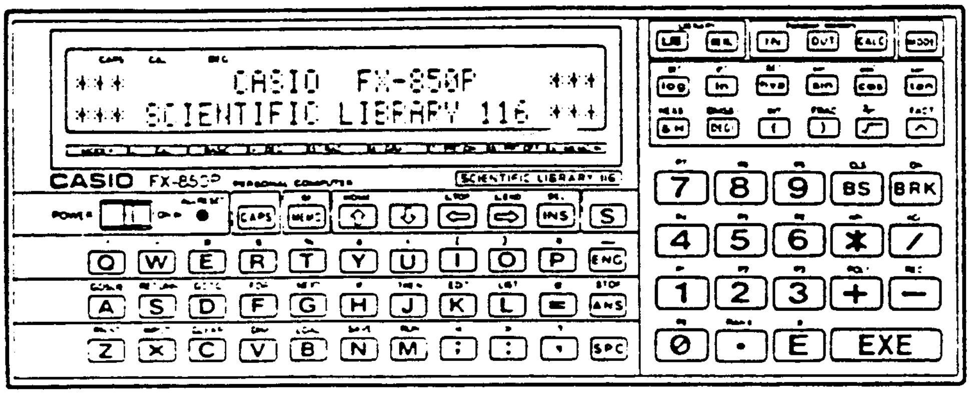
1-3 SYMBOL DISPLAY

The symbols noted on the display illustrated below appear to show the current status of a calculation.



CAPS	:	Upper case alphabetic characters (lower case when not displayed)
[S]	:	Shift mode (commands/functions marked above keys can be input)
CAL	:	CAL mode (basic arithmetic calculations or function calculations)
BASIC	:	BASIC mode (BASIC program input, editing, execution)
DEG	:	Angle unit = degrees
RAD	:	Angle unit = radians
GRA	:	Angle unit = grads
[MEMO]	:	DATA BANK data search, display, input, editing
[IN]	:	DATA BANK data input, editing
[EDIT]	:	DATA BANK data editing, BASIC program editing (using EDIT command)
DEFM	:	DEFM mode (for execution of CASIO PB-100 series programs)
[LIB]	:	Scientific library function mode
PRT	:	Print mode (output of display contents to printer)
TR	:	Trace mode (traces execution of BASIC programs)
STOP	:	Display suspended (by BASIC STOP command or PRINT statement)

1-4 KEYBOARD



A look at the keyboard of the unit reveals characters and symbols located above the keys. These are accessed using the [CAPS] and [MEMO] keys.

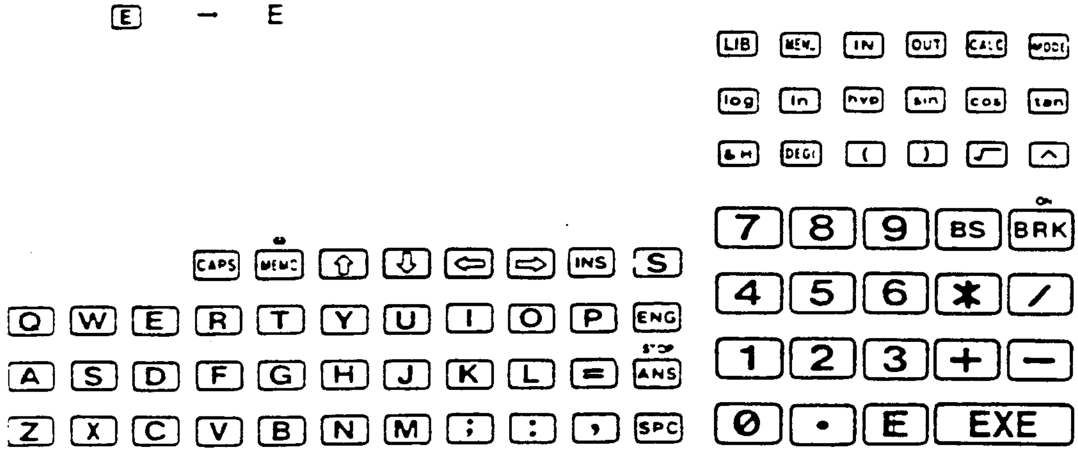
1-4-1 Keytop Functions

Normal Mode

In this mode, each key inputs the characters, symbols, or commands noted on the keys themselves. (This status is automatically set when power is switched ON and immediately following the ALL RESET procedure.)

EXAMPLE:

Operation		Display
[A]	→	A
[E]	→	E

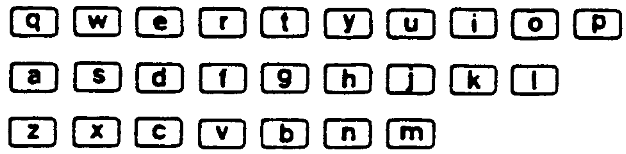


Lower Case Mode

Pressing the [CAPS] key shifts the alphabetic keys (only) to lower case characters, indicated by the CAPS symbol disappearing from the display. Pressing the [CAPS] key once locks the keyboard into the lower case mode, while pressing again returns to upper case.

EXAMPLE:

Operation		Display
[CAPS] [A]	→	a
[B]	→	b
[D]	→	d

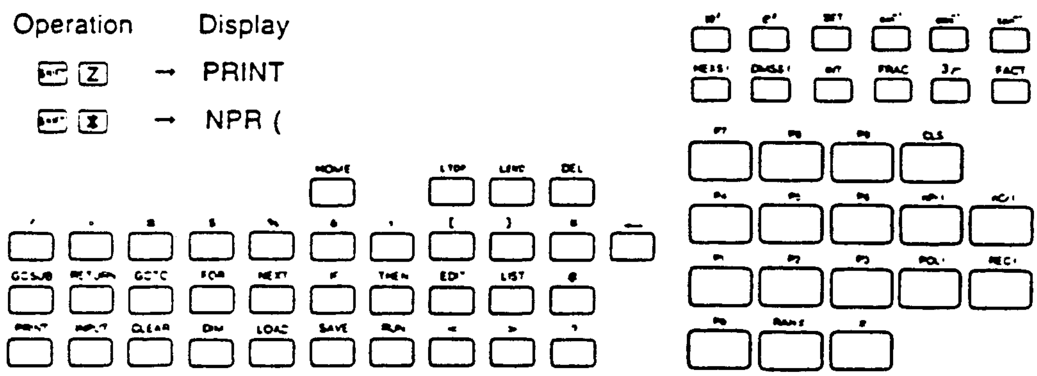


1-4-2 Functions Noted Above the Keys

The BASIC one-key commands, and the symbols and commands noted above the keys are entered when the corresponding keys are pressed following the [SHIFT] key. Note, however, that pressing the numeric keys ([0] ~ [9]) after [SHIFT] in the CAL mode executes the program in the corresponding program area, while, in the BASIC mode, switches to the corresponding program area.

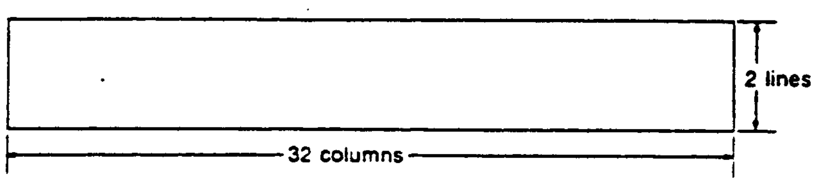
EXAMPLE:

Operation		Display
[SHIFT] [Z]	→	PRINT
[SHIFT] [X]	→	NPR (



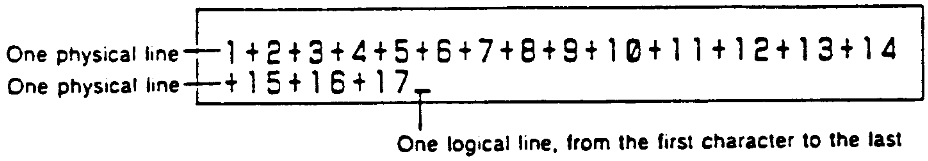
1-5 SCREEN

The screen is a 32-column × 2-line liquid crystal display. Characters are formed by a 5 × 7 dot matrix.



1-5-1 Physical Lines and Logical Lines

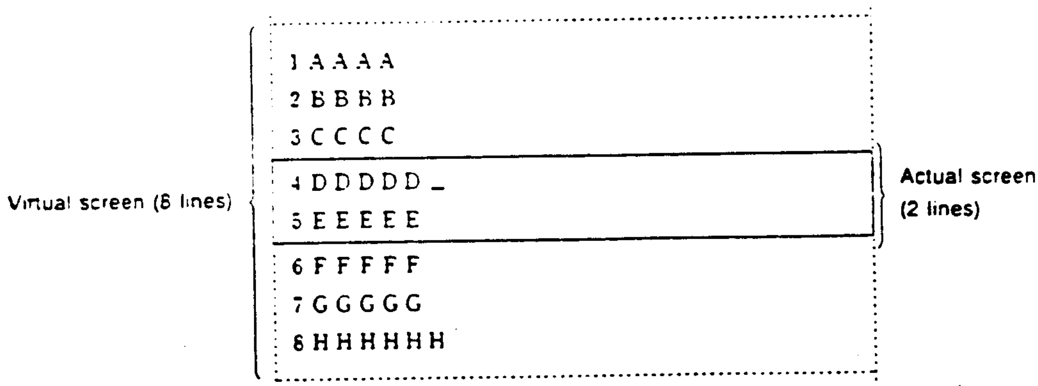
The maximum display capacity of one line is 32 columns, but internally the unit is capable of handling lines up to 255 characters long. The display capacity line (32 characters) is referred to as the physical line, while the internal capacity line is called a logical line. A logical line is a continuous line of characters in which any column on the extreme right of the screen is not a null.



Pressing moves the cursor to the beginning of the logical line, while moves the cursor to the end of the logical line. These operations are useful in determining the extent of logical lines.

1-5-2 Virtual Screen

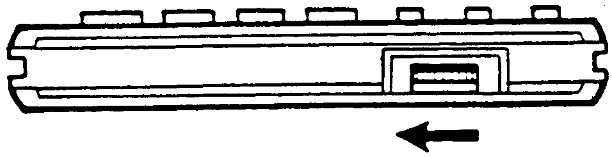
The screen can display two lines at one time, and as the 3rd line is input, the first line scrolls off the top of the screen. Lines that scroll off of the screen can, however, be brought back into view using the cursor (/) keys, because the unit is able to store up to eight lines internally. These eight lines make up the virtual screen, while the two lines actually displayed are called the actual screen.



1-5-3 Screen Editor

Any program lines or data included on the virtual screen can be edited. First the portion of the program or data is brought onto the actual screen, and then the cursor is located at the position to be edited.

1-5-4 Display Contrast



The display may appear dark or dim depending upon the strength of the batteries or the viewing angle. The contrast of the display can be adjusted to the desired level by rotating the control dial. Rotating the dial down (arrow direction) darkens the display, while rotating it up lightens the display.
A weak display when contrast is set to a high level indicates weakened batteries, and batteries should be replaced as soon as possible (see page 9).

1-6 DISPLAY CHARACTERS

The relationship between characters and character codes is illustrated in the following table.

Character Code Table

High-order digit ⟶ ; Low-order digit ↓

		0	16	32	48	64	80	95	112	128	144	160	176	192	208	224	240
	HEX	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0			␣	0	@	P	`	p	Å	°	␣	ー	タ	ミ	≥	×
1	1		(DEL)	!	1	A	Q	a	q	∫	¹	。	ア	チ	ム	≤	円
2	2	([illegible])	(INS)	"	2	B	R	b	r	√	²	「	イ	ツ	メ	≠	年
3	3			#	3	C	S	c	s	·	³	」	ウ	テ	モ	¡	月
4	4	([illegible])		$	4	D	T	d	t	Σ	⁴	、	エ	ト	ヤ	ー	日
5	5	(LINE CANCEL)		%	5	E	U	e	u	Ω	⁵	・	オ	ナ	ユ	!	千
6	6	(LINE END)		&	6	F	V	f	v	█	⁶	ヲ	カ	ニ	ヨ	→	万
7	7	(BEL)		'	7	G	W	g	w	█	⁷	ァ	キ	ヌ	ラ	x	£
8	8	(BS)		(	8	H	X	h	x	α	⁸	ィ	ク	ネ	リ	♠	¢
9	9	([illegible])		)	9	I	Y	i	y	β	⁹	ゥ	ケ	ノ	ル	♥	±
10	A	(LF)		*	:	J	Z	j	z	γ	⁺	ェ	コ	ハ	レ	♦	∓
11	B	(HOME)		+	;	K	[	k	|	ε	⁻	ォ	サ	ヒ	ロ	♣	⁰
12	C	(CLS)	([illegible])	,	<	L	¥	l	¦	θ	ⁿ	ャ	シ	フ	ワ	コ	
13	D	(CR)	([illegible])	-	=	M	]	m	|	μ	ˣ	ュ	ス	ヘ	ン	○	
14	E	([illegible])	([illegible])	.	>	N	^	n	-	σ	⁻¹	ョ	セ	ホ	゛	△	
15	F	([illegible])	([illegible])	/	?	O	_	o		ϕ	÷	ッ	ソ	マ	゜	＼	

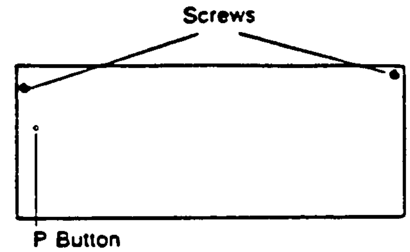
- Blank segments are not output.
- Notations in parentheses are control codes and are not displayed.
- Characters which cannot be displayed using keyboard input can be displayed using the CHR$ function.

1-7 POWER SUPPLY

This unit is equipped with a main power supply (two CR2032 lithium batteries) and a back up power supply (one CR1220 lithium battery). Batteries should be replaced whenever the display remains dim, even after contrast adjustment. Batteries should also be replaced once every two years regardless of how much the unit has been used.

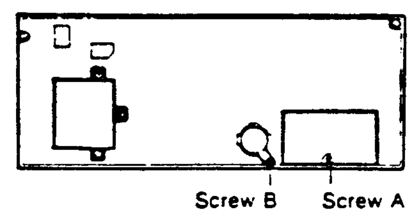
■ Battery Replacement

1) Switch the power of the unit OFF and remove the rear panel of the unit after removing the two screws holding it in place.



2) Remove the main battery holder by removing screw A, and/or the back up battery holder by removing screw B.

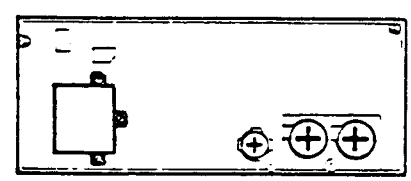
3) Remove the old batteries by turning the unit so that the battery compartment faces downward and tapping gently.



4) Wipe the surfaces of new batteries with a dry cloth and load them into the battery compartment ensuring that the positive pole (+) is facing upwards.

5) Replace the battery holder screw while pressing down on the batteries.

6) Replace the rear panel of the unit and two screws to hold it in place.



- Simultaneously removing both the main battery and back up battery causes programs and data stored in memory to be deleted.
- Be sure to remove dead batteries from the unit to avoid damage due to battery leakage.
- Be sure to replace both main batteries.
- Always ensure that battery polarity is correct.
- Never dispose of batteries in such a way that they will be incinerated. Exposing batteries to high heat can cause them to explode.
- P Button
  Hardware reset button to halt misoperation caused by static electricity. Be sure to switch power OFF and ON when the P button is pressed.

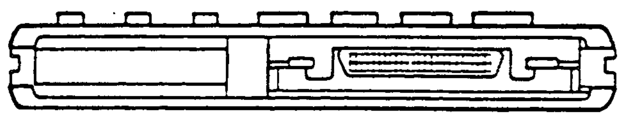
- ***Keep batteries out of the reach of small children. Consult a physician immediately if inadvertently swallowed.***

1-8 AUTO POWER OFF

The power of the unit is automatically switched OFF approximately 6 minutes after the last key operation (except during program execution), or the last input for an INPUT statement or PRINT statement. Power can be resumed by either switchng the power switch OFF and then ON again, or by pressing the BRK key.

* Program and data cotents are retained even when power is switched OFF, but settings such as the number of digits or the mode (i. e. BASIC mode, MEMO IN mode) are canceled.

1-9 CONNECTOR



The connector noted in the illustration is equipped for connection of peripheral devices (i.e. FA-6). Be sure to leave the connector cover in place when peripheral devices are not connected.

PART 2

FUNDAMENTAL OPERATION

This section covers the various modes available with the FX-850P using a series of simple examples. These procedures should be mastered before attempting more complex operations.

2-1 CAL MODE

The CAL mode is in effect each time the power of the unit is switched ON. Arithmetic calculations, function calculations, scientific library execution, formula storage calculations, program execution, and data recall can be performed in this mode.

EXAMPLE:

2.5 + 3.5 − 2 =

OPERATION:

2 [.] 5 [+] 3 [.] 5 [−] 2 [EXE]

```
2.5+3.5-2
 4
```

Though the [EXE] key is used instead of the [=] key, operation is identical to that used in a standard calculator.
The CAL mode can be entered from another mode by pressing [MODE] [0]. See PART 3 CALCULATION FUNCTION (page 15) for details.

2-2 BASIC MODE

The BASIC mode is used for the creation, execution and editing of BASIC programs. The BASIC mode can be entered from another mode by pressing [MODE] [1].

EXAMPLE:

Create and execute a program which calculates the sum of two values A and B.

PROGRAM INPUT

[MODE] 1

[SHIFT] [CLS]

10 [A] [=] 5 [EXE]

20 [B] [=] 6 [EXE]

30 [P] [R] [I] [N] [T] [A] [+] [B] [EXE]

40 [E] [N] [D] [EXE]

PROGRAM EXECUTION

[R] [U] [N] [EXE]

```
RUN
 11
```

See PART 6 BASIC PROGRAMMING (page 45) for details on using the BASIC language.

2-3 FORMULA STORAGE FUNCTION

The formula storage function makes it possible to store often used formulas in memory for calculation when values are assigned to variables. This function is applied in the CAL mode using the [IN], [OUT], and [CALC] keys.

EXAMPLE:

Determine the selling price of a product by applying a profit rate based on the purchase price and selling price.

SELLING PRICE = PURCHASE PRICE ÷ (1 − PROFIT%)

KEY INPUT

[MODE] [0]

[S] [E] [L] [L] [=] [P] [U] [R] [C] [H] [A] [S] [E] [/] [(] 1 [−] [P] [R] [O] [F] [I] [T] [)] [IN]

Required to store formula in memory

Ensure that input of the formula is correct by pressing the [OUT] key.

OPERATION:

[OUT]

```
SELL=PURCHASE/(1-PROFIT)
SELL=PURCHASE/(1-PROFIT)_
```

Now calculate the selling prices of the following:

PURCHASE PRICE	PROFIT
$1,000	30%
$960	25%

Key operation	Display
[CALC]	PURCHASE?_
1000 [EXE]	PURCHASE?1000 PROFIT?_
0 [.] 3 [EXE]	PROFIT?0.3 SELL= 1428.571429
[CALC]	PURCHASE?_
960 [EXE]	PURCHASE?960 PROFIT?_
[.] 25 [EXE]	PROFIT?.25 SELL= 1280

As can be seen in this example, once a formula is input, it can be used repeatedly by simply assigning values for the variables. See PART 4 FORMULA STORAGE FUNCTION (page 33) for details.

* The [BRK] key can be used to terminate this function.

2-4 DATA BANK FUNCTION (MEMO IN MODE)

The DATA BANK function allows the storage of large volume of data for recall when required. Pressing [MODE] [9] enters the MEMO IN mode, and causes the cursor to appear at the upper left of the display waiting for data input.

EXAMPLE:

J. SMITH 347-237-4811

[MODE] [9]
[S] [M] [I] [T] [H] [SPC] [J] [.] [SPC] 347 [-] 237 [-] 4811 [EXE]

Display
CAPS DEG MEMO ± :
SMITH J. 347-237-4811

Once data are stored, specific data items can be recalled using the [MEMO] key.

Key operation	Display	
[MODE] 0	–	(CAL mode)
[S] [MEMO]	SMITH J. 347-237-4811	(Search for data item beginning with "S".)

The [BRK] key can be used to terminate the DATA BANK function.
For details on the storage and retrieval of data, see PART 5 DATA BANK FUNCTION (page 37).

2-5 BUILT-IN SCIENTIFIC LIBRARY

This function provides a wide variety of useful scientific library that can be recalled and used in calculations in the CAL mode.
Mathematical/Statistical operations — 116 types
Operations are selected using the [illegible], [↑], [↓], [EXE] and [LIB] keys.
* For details, see PART 11 SCIENTIFIC LIBRARY (page 176).

2-6 SUMMARY

Function Table

Function name	Key operation
CAL mode	[MODE] [0]
BASIC mode	[MODE] [1]
Data bank	[illegible] [9]
Formula storage	[illegible] [0] + [IN], [OUT], [CALC]
Built-in scientific library	[illegible] [0] + Library No. [LIB]

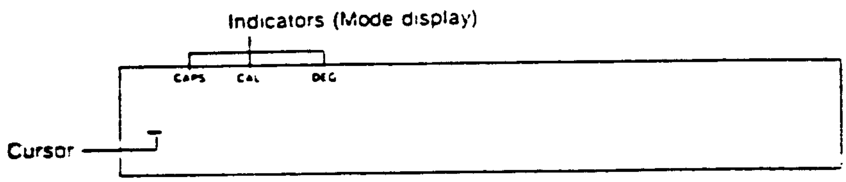
PART 3

CALCULATION FUNCTION

This section covers fundamental arithmetic calculations and function calculations which are performed manually.

3-1 MANUAL CALCULATION PREPARATIONS

Switch the Power of the Unit ON



The display illustrated above appears whenever the power is switched ON. It indicates the CAL mode in which manual calculations can be performed. Currently specified angle unit, however, is retained even when the power is switched OFF.

3-2 MANUAL CALCULATION INPUT AND CORRECTION

Perform the following fundamental calculations to become familiar with this mode.

EXAMPLE:

123 + 456 = 579

Key operation	Display	
123 [+] 456	123+456_	(Formula input)
[EXE]	123+456 579	(Obtains result)

As can be seen here, the [EXE] key is pressed in place of [=]. The [×] key is used for multiplication and [/] is used for division.

The following procedure can be used to correct entered data.

EXAMPLE:

33 × 5 + 16 = 181

For the sake of example, the value 33 here will be mistakenly entered as 34.

34 [×] 5 [+] 16	34*5+16_	

Press [⇦] six times to move cursor back to position of 4. This can also be accomplished by [SHIFT] [L.TOP] [⇦].

[⇦] [⇦] [⇦] [⇦] [⇦] [⇦]	34*5+16	
3	33*5+16	(Replaces 4 with 3.)
[EXE]	33*5+16 181	

EXAMPLE:

33 × 5 + 16 = 181

For the sake of example, the above calculation will be performed with the value 33 mistakenly entered as 34.

34 [×] 5 [+] 16 [EXE]	34*5+16 166	
[⇧] [⇦] [⇦] [⇦]	34*5+16 166	(Move cursor to position for correction.)
3	33*5+16 166	
[EXE]	33*5+16 181	(Reexecute calculation.)

EXAMPLE:

33 × 5 + 16 = 181

For the sake of example, the multiplication sign (*) here will be mistakenly omitted and calculated.

335 [+] 16 [EXE]	335+16 351	
[⇧] [⇦] [⇦]	335+16 351	(Move cursor to position for insert of *.)
[INS]	33_5+16 351	(Opens up one character space.)
[×]	33*5+16 351	(Enter multiplication symbol.)
[EXE]	33*5+16 181	(Obtain result again.)

As can be seen in the above example, the [INS] key is used to insert spaces at the current cursor location for input of characters or symbols.

EXAMPLE:

33 × 5 + 16 = 181

For the sake of example, the above calculation will be performed with the value 16 mistakenly entered as 216.

33 [×] 5 [+] 216 [EXE]

```
33*5+216
 381
```

[⇧] [⇦]

```
33*5+216
 381
```

(Moves cursor back to position of 2.)

[SHIFT] [DEL]

```
33*5+16
 381
```

(Delete 2.)

[EXE]

```
33*5+16
 181
```

(Reexecute calculation.)

As can be seen in the above example, the [DEL] key is used to delete characters at the current cursor location.

- The [BS] key can also be used to delete characters, but its operation is slightly different from [SHIFT] [DEL].

[SHIFT] [DEL] .... Deletes the character at the current cursor location.

```
ABCDEFGH  --[SHIFT][DEL]-->  ABCEFGH
```

[BS] .... Deletes the character at the position to the left of the current cursor location.

```
ABCDEFGH  --[BS]-->  ABDEFGH
```

Practice the following examples to become familiar with the fundamental calculation procedure.

EXAMPLE 1:

9 + 7.8 ÷ 6 − 3.5 × 2 = 3.3

OPERATION:

9 [+] 7.8 [/] 6 [−] 3.5 [×] 2 [EXE]

```
9+7.8/6-3.5*2
 3.3
```

EXAMPLE 2:

56 × (− 12) ÷ (− 2.5) = 268.8

OPERATION:

56 [×] [−] 12 [/] [−] 2.5 [EXE]

```
56×-12/-2.5
 268.8
```

Negative values are entered by pressing the [−] key before entering the value.

EXAMPLE 3:

$(4.5 \times 10^{75}) \times (-2.3 \times 10^{-78}) = -0.01035$

OPERATION:

4.5 [E] 75 [×] [−] 2.3 [E] [−] 78 [EXE]

```
4.5E75×-2.3E-78
-0.01035
```

Exponents are entered by pressing the [E] key (or the alphabetic [E] key) before entering the value.

The following example shows how the result of one calculation can be immediately incorporated into a subsequent calculation.

EXAMPLE 4:

$(23 + 456) \times 567 = 271593$

OPERATION:

23 [+] 456 [EXE]

```
23+456
 479
```

continuing with

[×] 567 [EXE]

```
 479×567
 271593
```

The last result obtained can be entered at any point in a subsequent calculation by pressing the [ANS] key.

EXAMPLE 5:

$81.2 \div (5.6 + 8.9) = 5.6$

(5.6 + 8.9) — This process performed first

OPERATION:

5.6 [+] 8.9 [EXE]

```
5.6+8.9
 14.5
```

continuing with

81.2 [/] [ANS]

```
 14.5
81.2/ 14.5_
```

[EXE]

```
81.2/ 14.5
 5.6
```

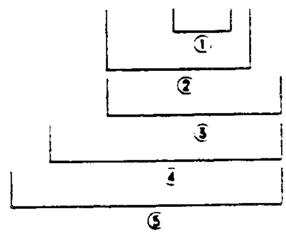
3-3 PRIORITY SEQUENCE

Arithmetic, relational and logical operations are performed in the following priority sequence:

1. (,)
2. Functions
3. Power
4. Signs (+, −)
5. ∗, /, ¥, MOD
6. +, −
7. Relational operators
8. NOT
9. AND
10. OR, XOR

EXAMPLE:

2 + 4 ∗ COS (14 + 16)^2 = 5



NOTE:

a. Calculations are performed from left to right when the priority sequence is identical.
b. Complex functions (sin cos 60) are performed from right to left.
c. Consecutive powers (5^4^3) is performed from left to right.

Calculation results are displayed in the following manner.

1. Integer less than 1×10^{10}:	Integer
2. 10 digits or less in fractional part:	Decimal
3. Others:	Exponential

Number of Digits

- Internal calculations are performed with a 12-digit mantissa + 2-digit exponent. PI, however, is expressed in 11 digits (3.1415926536): rounded and displayed in 10 digits (3.141592654).
- Calculation results are displayed rounded off to a 10-digit mantissa + 2-digit exponent.
- Up to 255 characters can be entered for a single line.

Specifying the Number of Significant Digits and the Number of Decimal Places

"SET" is used for these specifications.

Specification of number of decimal places........ [SHIFT] [SET] [F] (0 − 9) [EXE]
Specification of number of significant digits....... [SHIFT] [SET] [E] (0 − 9) [EXE]

- "SET E0" used to specify the number of significant digits specifies 10 digits.

Release of specification...................... [SHIFT] [SET] [N] [EXE]

- When a specification is made, the result is displayed by the number of specified digits. (The digit next to the last specified digit is rounded off.) The original value remains in the computer.

EXAMPLE:

Specified number of decimal places: 2

OPERATION:

[SHIFT] [SET] [F] 2 [EXE]	SET F2 _
10 [/] 3 [EXE]	10/3 3.33

EXAMPLE:

Specified number of significant digits: 3

OPERATION:

[SHIFT] [SET] [E] 3 [EXE]	SET E3 _
12 [*] 34 [EXE]	12*34 4.08E+02

3-4 SCIENTIFIC CALCULATIONS

The scientific functions (see the scientific function table on page 28) can be used either within programs or for manual calculations. For the sake of explanation, all of the examples here will cover only manual calculations.

Trigonometric and Inverse Trigonometric Functions

sin: sine	$\sin^{-1}$: arc sine
cos: cosine	$\cos^{-1}$: arc cosine
tan: tangent	$\tan^{-1}$: arc tangent

These functions return a trigonometric function value for a given angle, or an angle value of a given trigonometric function value. The ANGLE command should be used to specify the unit for the angle value when these functions are used. Angle unit specification is only required once for all subsequent trigonometric/inverse trigonometric functions. Angle units can be specified using either the [MODE] key or the ANGLE command.

ANGLE 0.....DEG (degrees) → ([MODE] [4])
ANGLE 1.....RAD (radians) → ([MODE] [5])
ANGLE 2.....GRAD (grads) → ([MODE] [6])

The relationship among these three specifications is:

$$90 \text{ degrees} = \frac{\pi}{2} \text{ radians} = 100 \text{ grads}$$

The current angle unit is retained when the power of the unit is switched OFF, and the angle unit becomes ANGLE 0 (DEG) when the ALL RESET button is pressed.

The value for π can be directly entered into a formula using "PI" (3.141592654).

EXAMPLE 1:

$\sin 30° = 0.5$

OPERATION:

[SHIFT] [SET] [N] [EXE]

[MODE] 4

```
CAPS   CAL      DEG

_
```

[sin] (or [S] [I] [N]) 30 [EXE]

```
CAPS   CAL      DEG

SIN30
 0.5
```

EXAMPLE 2:

$\cos \frac{\pi}{3} = 0.5$

OPERATION:

[MODE] 5

```
CAPS   CAL      RAD

_
```

[cos] (or [C] [O] [S])
[(] [SHIFT] [π] [/] 3 [)] [EXE]

```
CAPS   CAL      RAD

COS(PI/3)
 0.5
```

[SHIFT] [π] can also be entered as [P] [I] .

EXAMPLE 3:

$2\sin \frac{\pi}{3} + \cos \frac{\pi}{3} = 2.232050808$

OPERATION:

[MODE] 5

```
CAPS   CAL      RAD

_
```

2 [*] [sin] [(] [SHIFT] [π] [/] 3 [)] [+] [cos] [(] [SHIFT] [π] [/] 3 [)] [EXE]

```
CAPS   CAL      RAD

2*SIN(PI/3)+COS(PI/3)
 2.232050808
```

EXAMPLE 4:

tan 60° = 1.732050808

OPERATION:

[MODE] 4

```
CAPS  CAL  DEG
_
```

[tan] (or [T] [A] [N]) 60 [EXE]

```
CAPS  CAL  DEG
TAN60
 1.732050808
```

EXAMPLE 5:

$\sin^{-1} 0.5 = 30°$

OPERATION:

[SHIFT] [sin⁻¹] 0.5 [EXE]

```
CAPS  CAL  DEG
ASN0.5
 30
```

EXAMPLE 6:

$$\cos^{-1}\frac{2^{0.5}}{2} = 45°$$

OPERATION:

[SHIFT] [cos⁻¹] [(] 2 [^] 0.5 [/] 2 [)] [EXE]

```
CAPS  CAL  DEG
ACS(2^0.5/2)
 45
```

EXAMPLE 7:

$\tan^{-1}\sqrt{3} = 60° = 1.047197551$

OPERATION:

[SHIFT] [tan⁻¹] [√] 3 [EXE]

```
CAPS  CAL  DEG
ATNSQR3
 60
```

[MODE] 5

```
CAPS  CAL  RAD
ATNSQR3
 60
```

[SHIFT] [tan⁻¹] [√] 3 [EXE]

```
CAPS  CAL  RAD
ATNSQR3
 1.047197551
```

Hyperbolic and Inverse Hyperbolic Functions

sinh:	hyperbolic sine	$\sinh^{-1}$:	hyperbolic arc sine
cosh:	hyperbolic cosine	$\cosh^{-1}$:	hyperbolic arc cosine
tanh:	hyperbolic tangent	$\tanh^{-1}$:	hyperbolic arc tangent

EXAMPLE 1:

sinh 5 = 74.20321058

OPERATION:

[HYP] [sin] 5 [EXE]

```
HYPSIN5
 74.20321058
```

(The HYPSIN function is used for sinh.)

EXAMPLE 2:

$\cosh^{-1}$1.5 = 0.9624236501

OPERATION:

[HYP] [SHIFT] [$\cos^{-1}$] 1.5 [EXE]

```
HYPACS1.5
 0.9624236501
```

(The HYPACS function is used for $\cosh^{-1}$.)

Logarithmic Functions, Exponential Functions

$\log_{10}$:	common logarithm	e^x:	exponent
$\log_e$:	natural logarithm		

EXAMPLE 1:

$\log_e$ 123 = 4.812184355

OPERATION:

[ln] 123 [EXE]

```
LN123
 4.812184355
```

(The LN function is used for $\log_e$)

EXAMPLE 2:

$\log_{10}$ 100 = 2

OPERATION:

[log] 100 [EXE]

```
LOG100
 2
```

(The LOG function is used for $\log_{10}$)

EXAMPLE 3:

e^5 = 148.4131591

OPERATION:

[SHIFT] [e^x] 5 [EXE]

```
EXP5
 148.4131591
```

(The EXP function is used for e^x)

Other Functions

SGN:	Sign	RAN#:	Random number	ABS:	Absolute value
INT:	Integer value	FIX:	Integer part	FRAC:	Fraction
ROUND:	Rounding				

• SGN

For SGN (x), returns a 1 when x>0, a −1 when x<0, and a 0 when x=0.

OPERATION:

SGN6 [EXE]

```
SGN6
 1
```

SGN−2 [EXE]

```
SGN-2
-1
```

• RAN#

Generates a random number between 0 and 1 with up to 10 decimal places. For details, see PART 10 COMMAND REFERENCE.

OPERATION:

[SHIFT] [RAN#] [(] [−] 1 [)] [EXE]

```
RAN#(-1)
 0.2466393388                 *
```

* The above is only a sample value.

• ABS

Returns the absolute value of x for ABS (x).

|78.9÷−5.6| = 14.08928571

OPERATION:

ABS [(] 78.9 [/] [−] 5.6 [)] [EXE]

```
ABS(78.9/-5.6)
 14.08928571
```

• INT

For INT (x), returns the largest integer which does not exceed the value of x.

OPERATION:

[SHIFT] [INT] [−] 64.5 [EXE]

```
INT-64.5
-65
```

• FIX

Returns the integer part of x for FIX (x).

Integer part of 8000÷96.

OPERATION:

FIX [(] 8000 [/] 96 [)] [EXE]

```
FIX(8000/96)
 83
```

- **FRAC**

Returns the fractional part of x for FRAC (X).

Fractional part of 8000÷96.

OPERATION:

FRAC [(] 8000 [/] 96 [)] [EXE]

```
FRAC(8000/96)
 0.3333333333
```

- **ROUND**

The function ROUND (X, −Y) rounds the result of X at the Yth decimal place (resulting in Y−1 decimal places).
Round result of 8000/96 to three decimal places.

OPERATION:

ROUND [(] 8000 [/] 96 [,] [−] 4 [)] [EXE]

```
ROUND(8000/96,-4)
 83.333
```

Decimal ↔ Sexagesimal conversions

DEG: Sexagesimal → Decimal
DMS$: Decimal → Sexagesimal

EXAMPLE 1:

12°34′56″ = 12.58222222°

OPERATION:

[DEG] 12 [,] 34 [,] 56 [)] [EXE]

```
DEG(12,34,56)
 12.58222222
```

EXAMPLE 2:

12.3456° = 12°20′44.16″

OPERATION:

[S-ƒ] [DMS$] 12.3456 [)] [EXE]

```
DMS$(12.3456)
 12°20'44.16
```

Decimal ↔ Hexadecimal conversions

&H: Hexadecimal → Decimal
HEX$: Decimal → Hexadecimal

EXAMPLE 1:

$10_{(16)} = 16_{(10)}$

OPERATION:

[&H] 10 [EXE]

```
&H10
 16
```

EXAMPLE 2:

$1000_{(10)} = 3E8_{(16)}$

OPERATION:

[SHIFT] [HEX$] 1000 [)] [EXE]

```
HEX$(1000)
03E8
```

* Hexadecimal A, B, C, D, E, F corresponds to decimal 10, 11, 12, 13, 14, 15.

FACT, NPR, NCR

These function return the factorial, permutation, and combination of entered values.

EXAMPLE 1:

10! = 3628800

OPERATION:

[SHIFT] [FACT] 10 [EXE]

```
FACT10
 3628800
```

EXAMPLE 2:

${}_{10}P_4 = 5040$

OPERATION:

[SHIFT] [NPR] 10 [,] 4 [)] [EXE]

```
NPR(10,4)
 5040
```

EXAMPLE 3:

${}_{10}C_4 = 210$

OPERATION:

[SHIFT] [NCR] 10 [,] 4 [)] [EXE]

```
NCR(10,4)
 210
```

REC ↔ POL

Converts rectangular coordinates to polar coordinates, and vice versa.

EXAMPLE 1:

Convert polar coordinates (5, $\frac{\pi}{6}$) to rectangular coordinates (X, Y).

OPERATION:

[MODE] 5 (Angle unit = RAD)

[SHIFT] [REC(] 5 [,] [SHIFT] [π] [/] 6 [)] [EXE]

```
REC(5,PI/6)
 4.330127019
```
(X coordinate)

[Y] [EXE]

```
Y
 2.5
```
(Y coordinate)

EXAMPLE 2:

Convert rectangular coordinates (1, 1) to polar coordinates (r, θ).

OPERATION:

[SHIFT] [POL(] 1 [,] 1 [)] [EXE]

```
POL(1,1)
 1.414213562
```
(r coordinate)

[Y] [EXE]

```
Y
 0.7853981634
```
(θ coordinate)

Scientific Function Table

Function Name	Formula	Format	Details
Trigonometric	sin	SIN (numeric expression)	$-1440° <$ numeric expression $< 1440°$ (8πrad, 1600grad)
	cos	COS (numeric expression)	$-1440° <$ numeric expression $< 1440°$ (8πrad, 1600grad)
	tan	TAN (numeric expression)	$-1440° <$ numeric expression $< 1440°$ (8πrad, 1600grad) * Except when numeric expression is (2n-1) × 90° ($\pi/2$rad, 100grad) * n : integer
Inverse Trigonometric	$\sin^{-1}$	ASN (numeric expression)	|numeric expression| $\leqq 1$, $-90° \leqq$ ASN (numeric expression) $\leqq 90°$
	$\cos^{-1}$	ACS (numeric expression)	|numeric expression| $\leqq 1$, $0° \leqq$ ACS (numeric expression) $\leqq 180°$
	$\tan^{-1}$	ATN (numeric expression)	|numeric expression| $< 10^{100}$, $-90° \leqq$ ATN (numeric expression) $\leqq 90°$
Hyperbolic	sinh	HYPSIN (numeric expression)	|numeric expression| $\leqq 230.2585092$
	cosh	HYPCOS (numeric expression)	|numeric expression| $\leqq 230.2585092$
	tanh	HYPTAN (numeric expression)	|numeric expression| $\leqq 10^{100}$, $-1 \leqq$ HYPTAN (numeric expression) $\leqq 1$
Inverse Hyperbolic	$\sinh^{-1}$	HYPASN (numeric expression)	|numeric expression| $< 5 \times 10^{99}$
	$\cosh^{-1}$	HYPACS (numeric expression)	$1 \leqq$ numeric expression $< 5 \times 10^{99}$
	$\tanh^{-1}$	HYPATN (numeric expression)	|numeric expression| < 1
Exponential	e^x	EXP (numeric expression)	$-227 \leqq$ numeric expression $\leqq 230.2585092$
Natural logarithm	$\log_e x$	LN (numeric expression)	numeric expression > 0
Common logarithm	$\log_{10} x$	LOG (numeric expression)	numeric expression > 0
Square root	$\sqrt{x}$	SQR (numeric expression)	numeric expression $\geqq 0$
Cube root	$\sqrt[3]{x}$	CUR (numeric expression)	numeric expression $\geqq 0$
Absolute value	$\|x\|$	ABS (numeric expression)	Returns absolute value of numeric expression
Sign		SGN (numeric expression)	numeric expression < 0 : -1 numeric expression $= 0$: 0 numeric expression > 0 : 1
Integer		INT (numeric expression)	Gauss function: Returns maximum integer value that does not exceed numeric expression value.
Fraction		FRAC (numeric expression)	Returns fractional part of numeric expression.

Function Name	Formula	Format	Details
Rounding		ROUND (x, y) x, y : numeric expression	Rounds x at position specified by y.
Fix		FIX (numeric expression)	Returns integer part of x.
Degree	Sexagesimal →Decimal	DEG (d [,m[,s]]) d, m, s : numeric expression	Converts sexagesimal to decimal.
PI	π	PI	3.141592654
Random number		RAN# (numeric expression)	Returns a random number with 10 decimal places. 0 < RAN# < 1
Factorial	x !	FACT (numeric expression)	integer : $0 \leq$ numeric expression ≤ 69
Permutation	nPr	NPR (n, r)	integer : $0 \leq r \leq n \leq 10^{10}$
Combination	nCr	NCR (n, r)	integer : $0 \leq r \leq n \leq 10^{10}$
Coordinate conversions		POL (x, y)	Converts rectangular coordinates specified by $\|x\| + \|y\| > 0$ to polar coordinates.
		REC (r, θ)	Converts polar coordinates specified by $0 \leq r < 10^{100}$ to rectangular coordinates.

- Except for ROUND, DEG, NPR, NCR, POL and REC, any values used with these functions need not be included in parentheses.

3-5 CALCULATIONS USING VARIABLES

Algebraic calculations can also be performed using variables. The following list of calculations, for example, becomes much easier to perform if a variable is assigned for the common term.

2 × 3.1415 + 5 =
3 × 3.1415 + 6 =
4 × 3.1415 + 7 =
5 × 3.1415 + 8 =

1. First, assign the value 3.1415 to the variable X.

X [=] 3.1415 [EXE]

```
X=3.1415
_
```

2. Then use the variable in place of the value for each of the calculations.

2 [×] X [+] 5 [EXE]

```
2*X+5
 11.283
```

3 [×] X [+] 6 [EXE]

```
3*X+6
 15.4245
```

Variables

The following rules apply to variable names for all types of variables used with the unit.

Variable names:

1. Are character strings with an upper case alphabetic character (A – Z, internal decimal code 65 – 90) or lower case alphabetic character (a – z, internal decimal code 97 – 122) in the leading (first) position. (See the character code table on page 395 for internal codes.)
2. Are composed of upper or lower case alphabetic characters or numbers (0 – 9, internal decimal code 48 – 57) following the leading alphabetic character.
3. Cannot use reserved words (see page 400) as the leading characters.
4. Can be up to 15 characters long.

3-6 OTHER CALCULATIONS

Besides the fundamental arithmetic operations of addition, subtraction, multiplication, and division, and exponential calculations the FX-850P is also capable of employing a variety of other arithmetic and relational operators.

Arithmetic Operators

The following arithmetic operators are used in formulas:

Signs	(+, −)
Addition	(+)
Subtraction	(−)
Multiplication	(*)
Division	(/)
Power	(∧)
Integer division	(¥)
Remainder of integer division	(MOD)

The values used with the ¥ and MOD operators are limited to the range of − 32768 through 32767, and the fractional part of non-integer values is truncated.

EXAMPLE:

5¥2.9 = 2 (5÷2.9 = 2 .5)
7.3 MOD2.7 = 1* (7÷2 = 3····· 1)

(The fractional parts crossed out with "x" are truncated before the calculation is performed.)

* When a variable is used instead of 7.3, a space is required between the variable and the MOD operator.

With both ¥ and MOD, the values are converted to their absolute values before division is performed. The sign assigned to the result of the ¥ operation follows the rules of normal division, while the sign assigned to the result of the MOD operation is the sign of the dividend.

EXAMPLE:

−15¥7 = −2
−15MOD7 = −1

−15÷7 = −2 ..... −1
(−2 : −15¥7, −1 : −15MOD7)

Logical Operators

The application of logical operators is similar to that of arithmetic operators. The fractional parts of the data are truncated and the specified logical operation is performed bit-by-bit (each bit of the result is obtained by examining the bit in the same position for each argument). There are four different logical operators available with the unit.

NOT Makes an expression not true.
AND Expression is true if both parts are true, otherwise expression is false.
OR Expression is true if either part is true, otherwise expression is false.
XOR Expression is false if either part is true or either part is false, expression is true if one part is true and one part is false.

Negation

X	NOT X
0	1
1	0

Logical product

X	Y	X AND Y
0	0	0
0	1	0
1	0	0
1	1	1

Logical sum

X	Y	X OR Y
0	0	0
0	1	1
1	0	1
1	1	1

Excluxive OR

X	Y	X XOR Y
0	0	0
0	1	1
1	0	1
1	1	0

EXAMPLE:

Determine the logical sum for 10 and 3.

10 [O] [R] 3 [EXE]

```
10OR3
 11
```

10 = 1010(2)
3 = 0011(2)

11 = 1011(2)

* 1010(2) represents the binary value of 10.

Character Operator

The only string operator available is the plus (+) operator. The length of the result is limited to 255 characters.

EXAMPLE:

"A" + "B" → "AB"

Relational Operators

The following operators can be used within programs (only) to compare two values or strings. A true result returns a value of −1, while a false result returns 0.

Equal to	=
Not equal to	< >, > <
Less than	<
Greater than	>
Less than or equal to	= <, < =
Greater than or equal to	= >, > =

With character string comparisons, each character in the string to the left of the operator is compared with each character at the corresponding position in the string to the right of the operator. Comparisons are made using the character code for each character. If two strings are of different length and the shorter string is identical to the leading characters of the longer string, the shorter string is judged to be the lesser of the two.

EXAMPLE:

```
10  PRINT 125>12
20  PRINT "DEF"<"ABCD"
30  PRINT "ABCD"="ABC"
```

[MODE] 1 [R] [U] [N] [EXE]

```
RUN
-1
```

Since 125 is, in fact, greater than 12, a value of −1 (TRUE) is returned.

[EXE]

```
-1
0
```

The character code of "DEF" is greater than that of "ABCD", so 0 (FALSE) is returned.

[EXE]

```
0
0
```

The string "ABCD" is not equal to string "ABC", so 0 (FALSE) is returned.

* Character strings are compared until a difference is found, and judgment is made upon the first difference encountered. In the above example, the "A" in the first position of one string differs from the "D" in the first position of the other string, so the comparison is based upon "A" and "D". Though the string "ABCD" is longer, string "DEF" is considered to be the greater of the two because the character code of "D" is greater than the character code of "A".

PART 4

FORMULA STORAGE FUNCTION

The formula storage function is very useful when perfoming repeat calculations. Three different keys are used when working with the formula storage function.

[IN] key..........Stores presently displayed formula.
[OUT] key..........Displays formula stored in memory.
[CALC] key..........Assigns values to variables in formula, and displays formula calculation result.

Sample Application

EXAMPLE:

Obtain the value of y for each of the values assigned to x when y = 3.43 cosx. (Calculate in three decimal places.)

x	8°	15°	22°	27°	31°
y					

OPERATION:

First specify the angle unit and number of decimal places.

[MODE] 0 [EXE] 4 (Angle unit: "DEG")
[SHIFT] [SCI] [F] 3 [EXE] (Obtain in three decimal places by rounding off the 4th decimal place.)

Next, input a formula, and press the [IN] key to store it.

[Y] [=] 3 [.] 43 [×] [cos] [X] [IN]

Press the [OUT] key to confirm that the formula has been stored.

[SHIFT] [CLS] → `-`
[OUT] → `Y=3.43×COSX_`

Then, start calculating by pressing the [CALC] key.

Key operation	Display
[CAL]	X?_
8 [EXE]	X?8 Y= 3.397
[EXE]	Y= 3.397 X?_
15 [EXE]	X?15 Y= 3.313
[EXE]	Y= 3.313 X?_
22 [EXE]	X?22 Y= 3.180
[EXE]	Y= 3.180 X?_
27 [EXE]	X?27 Y= 3.056
[EXE]	Y= 3.056 X?_
31 [EXE]	X?31 Y= 2.940
[BRK]	Y= 2.940 _

The [CAL] key can be used in place of the [EXE] key to perform repeat calculations.

* The [BRK] key can be used to terminate this function to automatically return to the CAL mode.

4-1 UTILIZATION FOR PREPARING TABLES

Multiple formulas can be written by separating with colons (:). Tables such as that shown below can be easily prepared by using this method.

EXAMPLE:

Complete the following table. (Calculate in three decimal places by rounding off.)

X	Y	P=X*Y	Q=X/Y
4.27	1.17		
8.17	6.48		
6.07	9.47		
2.71	4.36		
1.98	3.62		

OPERATION:

[SHIFT] [SET] [F] 3 [EXE] .... Specification of number of decir al places

[P] [=] [X] [*] [Y] [:] [Q] [=] [X] [/] [Y] [IN] ...... Storing the formula

Key operation	Display	
[CALC]	X?_	(Calculation starts)
4 [.] 27 [EXE]	X?4.27 Y?_	(X value)
1 [.] 17 [EXE]	Y?1.17 P= 4.996	(Y value)
[EXE]	P= 4.996 Q= 3.650	
[EXE]	Q= 3.650 X?_	

⋮

Continue to input the values of X and Y in this mar..ier, and the values of P and Q will be calculated in successive order and the table will be completed as shown below.

X	Y	P=X∗Y	Q=X/Y
4.27	1.17	4.996	3.650
8.17	6.48	52.942	1.261
6.07	9.47	57.483	0.641
2.71	4.36	11.816	0.622
1.98	3.62	7.168	0.547

Variable names can consist of up to 15 upper case or lower case alphabetic characters. This means that variable names can be created which actually describe their contents. Remarks can also be affixed following variable names by enclosing the remarks within square brackets [ ]. Any character except for commas can be used within the remarks brackets.

EXAMPLE:

Complete the following table. (Calculate in two decimal places by rounding off.)

Radius r (m)	Height h (m)	Volume of a cylinder ($V_0 = \pi r^2 h$) (m^3)	Volume of a cone ($V_1 = {}^1/_3 V_0$) (m^3)
1.205	2.227		
2.174	3.451		
3.357	7.463		

OPERATION:

`SHIFT` `SET` `F` 2 `EXE`
`C` `Y` `L` `I` `N` `D` `E` `R` `SHIFT` `[` `M` 3 `SHIFT` `]` `=` `SHIFT` `π` `×` `R` `A` `D` `I` `U` `S` `SHIFT` `[` `M`
`SHIFT` `]` `^` 2 `×` `H` `E` `I` `G` `H` `T` `SHIFT` `[` `M` `SHIFT` `]` `:` `C` `O` `N` `E` `SHIFT` `[` `M` 3 `SHIFT` `]`
`=` `C` `Y` `L` `I` `N` `D` `E` `R` `/` 3 `IN`

Key operation	Display	
`CALC`	RADIUS[M]?_	(Calucualtion starts.)
1 `.` 205 `EXE`	RADIUS[M]?1.205 HEIGHT[M]?	(Radius)
2 `.` 227 `EXE`	HEIGHT[M]?2.227 CYLINDER[M3]= 10.16	(Height)
`EXE`	CYLINDER[M3]= 10.16 CONE[M3]= 3.39	
`EXE`	CONE[M3]= 3.39 RADIUS[M3]?_	
2 `.` 174 `EXE`	RADIUS[M2]?2.174 HEIGHT[M]?_	(Radius)

⋮

If the values of radius (r) and height (h) are input in this manner, volume (V_0) of the cylinder and volume (V_1) of the cone will be calculated successively and the table will be completed as shown below.

Radius r (m)	Height h (m)	Volume of a cylinder ($V_0 = \pi r^2 h$) (m^3)	Volume of a cone ($V_1 = 1/3\, V_0$) (m^3)
1.205	2.227	10.16	3.39
2.174	3.451	51.24	17.08
3.357	7.463	264.22	88.07

IMPORTANT

1. Up to 255 characters can be stored using the `IN` key. Storing new formula clears the currently stored formula.
2. Memory contents are retained even when power is switched OFF, either manually or by the auto power OFF function.
3. The `CALC` key can only be used to execute numeric expressions stored using the `IN` key.
4. An error is generated when an entry stored by the `IN` key is not a numeric expression.
5. Strings and arrays are simply displayed as stored when recalled.
6. The same limitations that apply to BASIC variables apply to formula storage function variables (see page 30).
7. Calculations are terminated under the following conditions:
   - Pressing the `BRK` key.
   - When an error is generated.

PART 5
DATA BANK FUNCTION

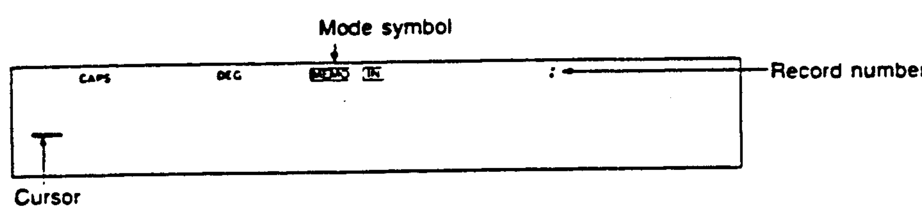
The DATA BANK function built into this unit gives it the capability to totally replace a standard notebook. For the sake of example here, the following scientific constant table will be input into the unit's DATA BANK.

SCIENTIFIC CONSTANT TABLE

Name	Symbol	Numeric value	Unit	Remarks
Acceleration of free fall	g	9.80665	ms^{-2}	FREE FALL
Speed of light (in space)	c	2.99792458×10^{8}	ms^{-1}	SPEED LIGHT
Planck's constant	h	6.626176×10^{-34}	Js	PLANCK'S
Gravitational constant	G	6.672×10^{-11}	Nm^2kg^{-2}	GRAVITATION
Elementary charge	e	$1.6021892 \times 10^{-19}$	C	ELEMENTARY
Electron mass	me	9.109534×10^{-31}	kg	ELECTRON
Atomic mass	u	$1.6605655 \times 10^{-27}$	kg	ATOMIC
Avogadro constant	N_A	6.022045×10^{23}	mol^{-1}	AVOGADRO
Boltzmann's constant	k	1.380662×10^{-23}	JK^{-1}	BOLTZMANN'S
Molar volume of ideal gas at s.t.p.	Vm	2.241383×10^{-2}	m^3mol^{-1}	IDEAL GAS

5-1 DATA INPUT

The MEMO mode must be entered using the operation [key] [9] to allow input of data into the DATA BANK. At this time, the display should appear as illustrated below:



The symbols appearing in the center of the top line of the display indicate that the current mode is the MEMO IN mode. The value on to the upper right indicates the record number, which is actually DATA BANK data line number. The record number 1 indicates that there is still no data stored. The following is the procedure to enter the constant for the acceleration of free fall:

OPERATION:

[CAPS] [G] [,] → Lower case for input of g

9 [.] 80665 [,] → Numeric value input

[M] [S] [-] 2 → Lower case for input of ms^{-2}

[SPC] [CAPS] [F] [R] [E] [E] [SPC] [F] [A] [L] [L] [EXE] → Upper case for remark input

Multiple items (i.e. symbols and values) can be included within a line by separating them with commas. The final step of the operation is the [EXE] key which writes the data into memory. This operation also causes the cursor to disappear from the display. Either press [EXE] again to display the cursor at the upper left or simply enter the first character for the next record. Either procedure switches to the next record number for entry of the next item.

OPERATION:

[CAPS] [C] [,] → Lower case for input of C

2.99792458 [E] 8 [,] → Numeric value input (exponent entered using [E])

[M] [S] [-] 1 → Lower case for input of ms^{-1}

[SPC] [CAPS] [S] [P] [E] [E] [D] [SPC] [L] [I] [G] [H] [T] [EXE] → Upper case for remark input

In this example, the value used as an exponent is entered using the [E] key. Note that both upper case and lower case letters were used in the first two lines. Always check the display for the current mode. The indicator **CAPS** indicates the upper case mode, while a clear display at the **CAPS** position indicates lower case.
Repeat the procedures outlined above until all ten constants are stored, and then switch to the CAL mode by pressing [MODE] [0] .

5-2 DATA DISPLAY

All of the data stored can now be displayed to check for proper input. While in the CAL mode, press the [▬] key to display records 1 and 2. Note here that only record 1 is displayed if its length exceeds 32 characters.

```
g.9.80665.ms-2 FREE FALL
c.2.99792458E8.ms-1 SPEED LIGHT
```

Pressing [↓] or [EXE] at this time displays records 2 and 3. Pressing [↑] or [SHIFT] [EXE] at any time displays the preceding record.

[↓]

```
c.2.99792458E8.ms-1 SPEED LIGHT
h.6.626176E-34.Js PLANCK'S
```

* Note that the scrolling key operations noted above differ as follows:

OPERATION	RESULT
[EXE]	Scrolls one record down.
[SHIFT] [EXE]	Scrolls one record up.
[↑]	Scrolls one line up.
[↓]	Scrolls one line down.

5-3 DATA EDITING

Editing of stored data is performed in the MEMO IN mode ([MODE] [9]). Of course, data may also be changed during the input procedure (before [EXE] is pressed) by moving the cursor to the desired location using the cursor keys ([⇦] and [⇨] keys) and then entering the correct data.

The following procedure is used to edit data which has already been stored.

1. Press [MODE] [9] (cursor not displayed)
2. Press [MODE] .
3. Locate record to be changed in the first line of the dipslay.
4. Press [⇦] or [⇨] to display cursor ([EDIT] appears on display to indicate EDIT mode).
5. Move cursor to desired location and enter correct data.
6. Press [EXE] ([EDIT] disappears from display).

The following example assumes that an error is discovered in RECORD 5 (elementary charge) during display in the CAL mode.

```
CAPS  CAL  DEG                    S
e.1.6.21892E-19.C ELEMENTARY
me.9.109534E-31.kg ELECTRON
```

Press [MODE] [9] to enter the MEMO IN mode.

[MODE] 9

```
CAPS       DEG     MEMO IN        S
e.1.6.21892E-19.C ELEMENTARY
me.9.109534E-31.kg ELECTRON
```

Here, the cursor can be displayed by pressing [⇦] or [⇨] . At this time, the [EDIT] symbol also appears to indicate the EDIT mode.

[⇨]

```
CAPS       DEG     MEMO IN EDIT   S
e.1.6.21892E-19.C ELEMENTARY
```

Cursor

Move the cursor to the desired location and enter the correct data. Finally, press [EXE] to complete the procedure ([EDIT] disappears from display).

[⇨] [⇨] [⇨] [⇨] [⇨] 0

```
CAPS       DEG     MEMO IN EDIT   S
e.1.6021892E-19.C ELEMENTARY
```

[EXE]

```
CAPS       DEG     MEMO IN        S
e.1.6021892E-19.C ELEMENTARY
me.9.109534E-31.kg ELECTRON
```

5-4 ADDING RECORDS

New records can be added to previously input records. Records can either be appended to the end of existing records, or inserted between two existing records.

5-4-1 Data Append

1. Press [MODE] [9]. Unit standing by for input of next successive record following previously stored records.
2. Enter data to append new record.
3. Press [EXE] to complete procedure.

5-4-2 Data Insert

1. Press [MODE] [9].
2. Press [INS] key.
3. Use [EXE] or [SHIFT] [EXE] to display existing record to follow newly inserted record.
4. Enter data.
5. Press [EXE] to complete procedure.

The following example describes how to enter a record containing the constant for the absolute temperature of water at 0°C between record 8 (Avogadro constant) and record 9 (Boltzmann's constant).

Name	Symbol	Numeric values	Unit	Remarks
Absolute temperature of water at 0°C	T0	273.15	K	ABS TEMP

Enter the above data after displaying RECORD 8 on the first line of the display.

OPERATION:

[T] 0 [,] 273 [.] 15 [,] [K] [SPC] [A] [B] [S] [SPC] [T] [E] [M] [P] [EXE]

The result of the operation is as follows:

RECORD 8 Avogadro constant
RECORD 9 Absolute temperature of water at 0°C
RECORD 10 Boltzmann's constant
RECORD 11 Molar volume of ideal gas at s.t.p

• To insert new data into RECORD 1, press [SHIFT] [EXE] after displaying RECORD 1, and then enter data for record 1. At this time, all following records are shifted downwards.

5-6-1 Conditional Search

Conditional searc. nakes it possible to designate a specific letter, value, or word (up to eigh characters long) in ler to quickly locate a desired record within a large file. Entering <objec data> ▤ display e first data item in which the <object data> appears immediatel following a comma. ach subsequent press of ▤ displays the following data item whic contains the <objec ata> following a comma.
In the following example, enter [N] ▤ to locate the Avogadro constant.

```
CAPS CAL      DEG  MEMO              N
G.6.672E-11.Nm2kg-2 GRAVITATION
e.1.6021892E-19.C ELEMENTARY
```

The first record to appear is record 4 (gravitational constant) because it contains the lette N following a comma. Press ▤ again to display the next data item which satisfies the statec condition.

```
CAPS CAL      DEG  MEMO              S
Na.6.022045E23.mol-1 AVOGADRO
k.1.380662E-23.JK-1 BOLTZMANN'S
```

Here, the desired data item is located. Of course, the ▤ key can be pressed as many times as desired until the <object data> are located. If none of the records contained the specified <object data> the cursor is displayed and the unit stands by for further input.

5-7 USING DATA BANK DATA IN PROGRAMS

Data stored within the DATA BANK can also be accessed from a BASIC program using the following program commands.

READ#

The standard READ command is generally used to read DATA statements contained within a program. The READ# command, on the other hand, reads data from the DATA BANK. Data are read in units from the beginning of a group of data up to the next comma.

FORMAT: READ# variable name [, variable name]

As shown above, multiple variable names can be specified, with variable names being separated by commas. As with the standard READ command, numeric data can only be assigned to numeric variables, and string data to string variables. Mismatching variable types results in a TM error, and executing the READ# command when data do not exist produces a DA error. Any leading spaces in a group of data are skipped, unless the group is included within quotation marks.

5-5 DATA DELETE AND ALL CLEAR

5-5-1 Data Delete

The following procedure is used to delete specific records from previously stored data.

1. Press [MODE] [9].
2. Press [MEMO] key.
3. Press [EXE] and recall record number to be deleted.
4. Press cursor key ([⇦] or [⇨]) to display EDIT symbol.
5. Press [SHIFT] [CLS] [EXE] to delete currently displayed record. All following records are shifted upwards.

5-5-2 Data All Clear

Data bank contents are retained when the power of the unit is switched off and when the NEW, NEW ALL and CLEAR commands are executed. The following procedure is used to clear all current contents of the data bank.

1. Press [MODE] [1] to enter the BASIC mode.
2. Enter [N] [E] [W] [SHIFT] [#] [EXE] to execute NEW# command and clear all data stored in DATA BANK.

IMPORTANT
Data cleared using the procedures outlined above cannot be recovered. Only delete or clear data when it is no longer required.

5-6 DATA SEARCH

Pressing the [MEMO] key while in the CAL mode or MEMO IN mode displays record 1. Now, each press of the [MEMO] key shifts the cursor to the data to the right of the next comma following the current cursor position.
The [EXE] key can also be used to shift the cursor to the next data item, and [SHIFT] [EXE] can be used to shift to the previous data item.

```
CAPS CAL     DEG
g.9.80665.ms-2 FREE FALL
c.2.99792458E8.ms-1 SPEED LIGHT
```

↓ [MEMO]

```
CAPS CAL     DEG
g.9.80665.ms-2 FREE FALL
c.2.99792458E8.ms-1 SPEED LIGHT
```

↓ [MEMO]

```
CAPS CAL     DEG
g.9.80665.ms-2 FREE FALL
c.2.99792458E8.ms-1 SPEED LIGHT
```

RESTORE#

As with the standard RESTORE command, RESTORE# can be used to designate a specific position from which the READ# operation is to be performed.

FORMAT: RESTORE#

Simply executing RESTORE# specifies that the next READ# or WRITE# operation is to be performed at the beginning of data currently stored in the DATA BANK.

FORMAT: RESTORE# "object string"

Including an object string with the RESTORE# command specifies that the next READ# or WRITE# operation is to be performed from the data item which begins with the specified object string contained in the DATA BANK. A DA error is generated when the specified object string does not exist. The maximum capacity for a WRITE# operation is 255 characters, and exceeding this value results in an error.

FORMAT: RESTORE# "object string", 0

The above format is identical to RESTORE# "object string".

FORMAT: RESTORE# "object string", 1

The above format specifies that the next READ# or WRITE# operation is to be performed from the record which begins with the specified object string.

FORMAT: RESTORE# "object string", {0 or 1}, {line number or # program area number}

The above format designates a jump to the specified line number or program area number for the next READ# or WRITE# operation when the specified object string does not exist.

WRITE#

The WRITE# command is used within a program to rewrite or delete DATA BANK data.

FORMAT: WRITE# DATA BANK data

The above format replaces existing data items with the specified DATA BANK data, starting from the current READ#/WRITE# position. In the case that data A, B, C exist in the DATA BANK, with data B specified for the next READ#/WRITE# operation, executing WRITE# Y, Z results in the DATA BANK data file being changed to A, Y, Z. Executing WRITE# "Y, Z" results in A, Y, Z, C. The data line specified for the next READ#/WRITE# operation is deleted when the WRITE# command is executed without specifying DATA BANK data.

5-8 DATA BANK FUNCTION APPLICATIONS

The data bank function can be used to perform a variety of tasks in addition to the applications outlined in this section of the manual. Virtually any data imaginable can be stored.

EXAMPLE:

The formula storage function can be used in combination with DATA BANK to store, recall and execute formulas whenever they are needed.

1	V=4*PI*R^3/3
2	S=PI*R^2
3	Y=3*X^2+4
4	Z=SINX+COSY
5	A=Z*1.13-X*1.24

The five formulas listed above are stored in the DATA BANK. Recall the third formula, transfer it to the formula storage function, and then execute it (in the CAL mode).

Key	Display	
[illegible]	V=4*PI*R^3/3 S=PI*R^2	(Displays 1st formula.)
EXE	S=PI*R^2 Y=3*X^2+4	(Displays 2nd formula.)
EXE	Y=3*X^2+4 Z=SINX+COSY	(Displays 3rd formula.)
IN	Z=SINX+COSY A=Z*1.13+X*1.24	(Stores 3rd formula in memory.)
CAL	X?_	(Executes formula stored in memory.)

IMPORTANT

Note that DATA BANK record lines are limited to 255 characters. Care should be exercised when making changes using the WRITE# command not to exceed this limit. Doing so results in an error.

PART 6

BASIC PROGRAMMING

Standard BASIC is employed as the programming language for this unit, and this section covers application of the BASIC language.

6-1 FEATURES OF BASIC

1. BASIC is much easier to use than other programming languages such as FORTRAN, making it suitable even for novices.
2. Writing programs is also easier because program creation, editing and execution are all performed by interacting with the computer itself.

The following functions are also available:

1. High-precision calculations are made possible by display of numeric values with 10-digit mantissas and 2-digit exponents (13-digit mantissa and 2-digit exponent for internal operations).

2. A wide selection of built-in functions makes operation easier.

① Standard mathematical functions
SIN COS TAN ASN ACS ATN LOG LN EXP SQR ABS SGN
INT FIX FRAC PI ROUND RAN# DEG

② Powerful string handling functions
CHR$ STR$ MID$ LEFT$ RIGHT$ HEX$ DMS$ ASC VAL LEN

③ High level mathematical functions
POL REC NCR NPR HYPSIN HYPCOS HYPTAN HYPASN HYPACS
HYPATN CUR

3. 10 independent program areas
Up to ten programs can be stored independently in memory at the same time (P0 ~ 9).

4. Extended variable names
Variable names up to 15 characters long can be used, making it possible to use names that makes contents easy to understand.

5. Powerful debugging function
A TRON command displays the number of the program line currently being executed, making it possible to easily trace execution and locate mistakes in programming.

6. Powerful screen editor
Programs can be easily modified and corrected on the screen.

7. Virtual screen function
Though the actual physical display of the unit has a 32-column × 2-line capacity, the virtual screen is 32 columns × 8 lines. The virtual screen can be easily scrolled using the cursor keys.

```
                    1 AAAA
                    2 BBBB
                    3 CCCC
V  ual Screen      ┌──────────────┐
(  ines)           │4 DDDDD_      │  Actual Screen
                   │5 EEEEE       │  (2 lines)
                   └──────────────┘
                    6 FFFFF
                    7 GGGGG
                    8 HHHHHH
```

8. Expanded file management
Such standard BASIC commands as OPEN, CLOSE, INPUT # and PRINT # are all available for data file reading and writing.

6-2 BASIC PROGRAM CONFIGURATION

6-2-1 BASIC Program Format

The following is a typical BASIC program which calculates the volume of a cylinder.

EXAMPLE:

```
10  REM CYLINDER
20  R=15
30  INPUT "H=" ; H
40  V=PI*R^2*H          (PI indicates π)
50  PRINT "V=" ; V
60  END
```

As can be seen, the BASIC program is actually a collection of lines (six lines in the above program). A line can be broken down into a line number and a statement.

```
20   R=15
└┘   └──┘
Line    Statement
number
```

Computers execute programs in the order of their line numbers. In the sample program listed above, the execution sequence is 10, 20, 30, 40, 50, 60. Program lines can be input into the computer in any sequence, and the computer automatically arranges the program within its memory in order from the smallest line number to the highest. Lines can be numbered using any value from 1 through 65535.

```
20  R=15                       10  REM CYLINDER
40  V=PI*R^2*H                 20  R=15
60  END               ➡        30  INPUT "H=" ; H
10  REM CYLINDER               40  V=PI*R^2*H
30  INPUT "H=" ; H             50  PRINT "V=" ; V
50  PRINT "V=" ; V             60  END

    Input sequence                 Memory contents
```

Following the line number is a statement or statements which actually tell the computer which operation to perform. The following returns to the sample program to explain each statement in detail

```
10  REM CYLINDER..............REM stands for "remarks". Nothing in this line is executed.
20  R=15......................Assigns the constant 15 (radius) to variable R.
30  INPUT "H=" ; H............Displays H ? to prompt a value input for height.
40  V=PI*R^2*H................Calculates volume (V) of cylinder.
50  PRINT "V=" ; V............Prints result of line 40.
60  END.......................Ends program.
```

As can be seen, a mere six lines of programming handles quite a bit of data. Multiple BASIC program lines can also be linked into a single line using colons.

EXAMPLE:

```
100  R=15:A=7:B=8
```

Such a program line is known as a "multistatement".

Details concerning the other operations contained in the above program can be found in other sections of PART 6.

6-3 BASIC PROGRAM INPUT

6-3-1 Preparation

First switch the power of the computer ON. At this time, the display should appear as illustrated below.

```
 CAPS  CAL      DEC
_
```

This is the CAL mode, so the operation [illegible] 1 should first be performed to allow input of BASIC programs. The display should now appear as illustrated below.

```
 CAPS        BASIC DEC
P 0 1 2 3 4 5 6 7 8 9        3536B
Ready P0
```

Note that the indicator CAL has been replaced by BASIC to indicate the BASIC mode. This is the mode used for BASIC program input. The other indicators on the display in the BASIC mode have the following meanings.

P : Program area

0 – 9 : Program area numbers. The numbers of program areas which already contain programs are replaced by asterisks.

EXAMPLE:
Program stored in area 3

```
CAPS    BASIC DEG
P  0 1 2 * 4 5 6 7 8 9      3521B
Ready P0
```

3536B : Capacity (number of bytes) remaining in area for writing programs and data (free area). The value will be 3536B when there are no programs or data stored in memory, with this value decreasing as storage space is used.

Ready P0: Current program area = area 0. The current program area can be switched by pressing [SHIFT] followed by the desired program area.

EXAMPLE:
Switching to program area 5

[SHIFT] [P5]

```
CAPS    BASIC DEG
P  0 1 2 3 4 5 6 7 8 9      3536B
Ready P5
```

Previously stored programs can be deleted using one of two different procedures.

NEW : Deletes program stored in current program area only.
NEW ALL : Clears all programs stored in memory.

EXAMPLE:

NEW ALL [EXE]

```
CAPS    BASIC DEG
P  0 1 2 3 4 5 6 7 8 9      3536B
Ready P0
```

This operation clears all programs stored in memory and returns to current program area to 0.

6-3-2 Program Input

The following input procedure inputs the sample program for calculation of the volume of a cylinder (page 46).

[1] [0] [R] [E] [M] [SPC] [C] [Y] [L] [I] [N] [D] [E] [R] [EXE]
[2] [0] [R] [=] [1] [5] [EXE]
[3] [0] [I] [N] [P] [U] [T] [SHIFT] ["] [H] [=] [SHIFT] ["] [;] [H] [EXE]
[4] [0] [V] [=] [P] [I] [*] [R] [^] [2] [*] [H] [EXE]
[5] [0] [P] [R] [I] [N] [T] [SHIFT] ["] [V] [=] [SHIFT] ["] [;] [V] [EXE]
[6] [0] [E] [N] [D] [EXE]

Note that the [EXE] key is pressed at the end of each line. A program line is not entered into memory unless the [EXE] key is pressed.

ONE-KEY INPUT
The one-key BASIC commands help to make program input even easier.

EXAMPLE:
Line 30 input.

[3] [0] [SHIFT] [INPUT] [SHIFT] [SPC] [H] [=] [SHIFT] [SPC] [;] [H] [EXE]

6-3-3 Program Editing

The procedure used for making corrections or changes to a program depends upon what step of program input the changes are to be made.

① Changes in a line before [EXE] is pressed
② Changes in a line after [EXE] is pressed
③ Changes within a program already input
④ Changes within a program following the EDIT command

1. Changes in a line before [EXE] is pressed

EXAMPLE:
20 E = 15 mistakenly input for 20 R = 15

Key	Display	
	10 REM CYLINDER 20 E=15_	
[⇦] [⇦] [⇦] [⇦]	10 REM CYLINDER 20 E=15	(Move cursor to E)
[R]	10 REM CYLINDER 20 R=15	(Input correct character)
[EXE]	20 R=15 _	(Editing complete)

Note that once the desired changes are made, the [EXE] key must be pressed to store the entered line into memory.

2. Changes in a line after [EXE] is pressed

EXAMPLE:
40 V = P1 * R ^ 2 * H mistakenly input for 40 V = PI * R ^ 2 * H

Key	Display	
	40 V=P1*R^2*H _	
[↑] [⇨] [⇨] [⇨] [⇨] [⇨] [⇨]	40 V=P1*R^2*H	(Move cursor to 1)
[I]	40 V=PI*R^2*H	(Input correct character)
[EXE]	40 V=PI*R^2*H _	(Editing complete)

Again, the [EXE] key must be pressed to store the corrected line into memory after changes are made.
Procedures 1 and 2 show the procedures for simple exchanges of one character for another.

Characters can also be inserted and deleted using the following procedures.

i) Insert

40 V=PI*R2*H mistakenly input for 40 V=PI*R^2*H

```
40 V=PI*R2*H_
```

[⇦] [⇦] [⇦]

```
40 V=PI*R2*H
```

(Move cursor to location of insertion)

[INS]

```
40 V=PI*R_2*H
```

(Open one space)

[^] [EXE]

```
40 V=PI*R^2*H
_
```

(Input correct character and press [EXE])

ii) Delete

40 V=PI*RR^2*H mistakenly input for 40 V=PI*R^2*H

```
40 V=PI*RR^2*H_
```

[⇦] [⇦] [⇦] [⇦] [⇦]

```
40 V=PI*RR^2*H
```

(Move cursor to character to be deleted)

[SHIFT] [DEL]

```
40 V=PI*R^2*H
```

(Delete character)

[EXE]

```
40 V=PI*R^2*H
_
```

(Editing complete)

The [BS] key works rather similarly to the [SHIFT] [DEL] operation. The difference between the two operations is as follows.

Difference Between [SHIFT] [DEL] and [BS]

• [SHIFT] [DEL]

Deletes the character at the current cursor location and shifts everything to the right of the cursor one space to the left.

A B C D E F G → [SHIFT] [DEL] → A B C E F G
(Cursor under D)

• [BS]

Deletes the character to the left of the current cursor location and shifts everything from the cursor position right one space to the left.

A B C D E F G → [BS] → A B D E F G

...hanges within a program already input

he LIST command displays the program stored in the current program area from beginning ... end.

IST [EXE]

```
10 REM CYLINDER
20 R=15
  :
60 END
Ready P0
```

he last line of the program is displayed when the LIST operation is complete.

[↑] [↑] [↑] [↑] [↑] [↑]

```
10 REM CYLINDER
20 R=15
```

he 8-line virtual screen of the computer now makes it possible to use the cursor keys to roll to preceding lines not shown on the display (see page 7).

```
Ready P0
10  REM CYLINDER
20  R=15
30  INPUT "H=" ; H
40  V=PI*R^2*H
50  PRINT "V=" ; V
60  END
Ready P0
```

...ne virtual ...een)

2-line display
[↑] [↓] keys move display

...en a program greater than eight lines is stored in memory, the LIST operation should performed by specifying the line numbers to be displayed.

AMPLE:
...playing from line 110 to line 160 on the virtual screen.
...T 110 — 160 [EXE]

```
LIST 110—160
110  A=1
120  FOR I=1 TO 100
130  B=LOG ( I )
140  PRINT B
150  NEXT I
160  E=A*B
Ready P0
```

...e virtual ...en)

2-line display

...nges can be made in a program displayed by the LIST operation by using the same proce...s outlined for 1 and 2 above.

...e [BRK] key can be used to terminate the LIST operation. The [STOP] key suspends the opera...n, and listing can be resumed by pressing [EXE].

4. Changes within a program following the EDIT command

The EDIT command makes it easier to edit or review programs already stored in memory.

EDIT [EXE]	CAPS BASIC DEG EDIT 10 REM CYLINDER 20 R=15	

From this display, [↓] (or [EXE]) advances to the following line, while [↑] (or [SHIFT] [EXE]) returns to the previous line.

[↓] [↓]	30 INPUT "H=";H 40 V=PI+R^2*H	(Displays lines 30 and 40)

Here, a correction will be made in line 40.

[↓]	40 V=PI+R^2*H 50 PRINT "V=";V	(Displays line 40 at upper line of display)
[⇨]	40 V=PI+R^2*H	(Enables program editing)
[⇨] [⇨] [⇨] [⇨] [⇨] [⇨] [⇨] [*] [EXE]	40 V=PI*R^2*H 50 PRINT "V=";V	(Correction)
[BRK]	Ready P0	([BRK] key exits EDIT mode)

6-4 BASIC PROGRAM EXECUTION

6-4-1 Program Execution

Once a BASIC program is stored in memory, it can be executed using one of the two following procedures.

1. Using [S-P] (program area) in CAL mode

   EXAMPLE: [S-P] [9]
   Executes the program in program area 9.

2. Entering RUN command in BASIC mode

   EXAMPLE: RUN [EXE]
   Executes the program in the current program area.

Execute the program input in the previous section to determine the volume of a cylinder with a height of 10 (radius fixed as 15).

RUN [EXE]	RUN H=?_	(Executes program)
10 [EXE]	H=?10 V= 7068.583471	(Corresponding volume displayed when height is entered.)
[EXE]	V= 7068.583471 Ready P0	

isplay of the volume when this program is executed causes the STOP symbol to appear
 the upper right of the display. This symbol indicates that program execution has been uspended because of execution of the PRINT command. Program execution can be resumed this time by pressing the [EXE] key again. Ending a PRINT statement with a semicolon causes ecution to continue when the PRINT statement is executed, causing the display of the xt PRINT statement to appear immediately following the previous display.

XAMPLE 1:

```
10  PRINT "BASIC  "
20  PRINT "PROGRAM"
30  END
```

xecution of this program results in the following display.

```
UN [EXE]        RUN
                BASIC

                BASIC
                PROGRAM

                PROGRAM
                Ready P0
```

XAMPLE 2:

```
10  PRINT "BASIC  ";
20  PRINT "PROGRAM"
30  END
```

cluding a semicolon at the end of the first PRINT statement produces the following display.

```
JN [EXE]        RUN
                BASIC  PROGRAM

                BASIC  PROGRAM
                Ready  P0
```

4-2 Errors

times, the results produced by a program are not what is expected. Such irregular executions can be broadly divided under two major classifications.

Executions that produce errors
Simple programming errors
Program logic errors
Irregular execution that do not produce errors
Mostly program logic errors

Executions that produce errors

Simple programming errors
s is the most common type of program error and is generally caused by mistakes in program ntax. Such errors result in the following message being displayed:

error P0—10

s message indicates that a syntax error has been detected in line 10 of the program stored program area 0. The indicated program line should be checked and corrected to allow per execution.

ii) Program logic errors

This type of error is generally caused by such illegal operations as division by zero or LOG(0). Such errors result in the following message being displayed:

MA error P0—10

As before, this message indicates that a mathematical error has been detected in line 10 of the program stored in program area 0. In this case, however, program lines related to the indicated program line as well as indicated program line itself should be examined and corrected. When an error message is displayed, the following two operations can be used to display the line number in which the error was detected.

{ LIST 10 EXE
{ EDIT 10 EXE

{ LIST . EXE
{ EDIT . EXE

- The periods contained in **LIST .** and **EDIT .** instruct the computer to automatically display the last program line executed.
- Change to the BASIC mode if a BASIC program was executed in the CAL mode.

2. Irregular execution that do not produce errors

Such errors are also caused by a flaw in the program, and must be corrected by executing the LIST or EDIT command to examine the program to detect the problem. The TRON command can also be used to help trace the execution of the program.
Entering **TRON** EXE causes the TR symbol to appear on the display to indicate that the trace mode is ON. Now executing a BASIC program displays the program area and line number as execution is performed, and halts execution until EXE is pressed. This allows confirmation of each program line, making it possible to quickly identify problem lines. Executing **TROFF** EXE cancels the trace mode.

6-5 COMMANDS

Though there are a variety of commands available in BASIC for use in programs, the nine fundamental commands listed below are the most widely used.

The following program reads data items contained within the program itself, with the number of data items read being determined by input from an operator. The operator may input any value, but note that values greater than 5 are handled as 5 (because there are only 5 data items in line 180). The program then displays the sum of the data read from line 180, followed by the square root and cube root of the sum. Program execution is terminated when a zero is entered by the operator.

```
10 S=0.............................Clears current total assigned to S
20 RESTORE.........................Specifies read operation should begin with first data item
30 INPUT N.........................Input of number of data items to be read
40 IF N>5 THEN N=5.................Input of value greater than 5 should be treated as 5
50 IF N=0 THEN GOTO 130............Jump to line 130 when input is zero
60 FOR I=1 TO N ----------------┐
70 READ X  Data read            |__This section repeated the number of times specified by
80 S=S+X   Sum of data calculation|  operator input in line 30
90 NEXT I ----------------------┘
```

```
100  GOSUB 140 ................................Branch to subroutine starting from line 140
110  PRINT S ; Y ; Z ..........................Displays contents of variables S, Y, Z
120  GOTO 10 ..................................Jump to line 10
130  END ......................................Program end
140  REM SQUARE ROOT/CUBE ROOT ................Remarks
150  Y = SQR S ................................Square root calculation
160  Z = CUR S ................................Cube root calculation
170  RETURN ...................................Return to the statement following the statement which
                                               called the subroutine
180  DATA 9, 7, 20, 28, 36 ....................Data read by READ statement in line 70
```

① REM

The REM command (line 140) is actually short for the word "remarks". The computer dis-
egards anything following a REM command, and so it is used for such purposes as labels
n order to make the program list itself easier to follow. Note that a single quotation mark
( ) can be used in place of the letters "REM".

② INPUT

he INPUT command (line 30) is used to allow input from the computer's keyboard during
rogram execution. The data input are assigned to a variable immediately following the INPUT
ommand. In the above example, input numeric data are assigned to the variable N. Note
hat a string variable must be used for string input.

XAMPLE:

0 INPUT A$ (string input)

③ PRINT

he PRINT command (line 110) is used to display data on the computer's display. In this
xample, this command is used to display the results of the sum, square root, and cube root
alculations. When the data are displayed, the STOP symbol appears and program execu-
on is suspended. Execution can be resumed by pressing the EXE key.

END

he END command (line 130) brings execution of the program to an end, and can be included
nywhere within a program.

IF – THEN –

he IF/THEN command (lines 40 and 50) is used for comparisons of certain conditions, bas-
g the next operation upon whether the comparison turns out to be true or false. Line 40
hecks whether or not value assigned to N is greater than 5, and assigns a value of 5 to
when the original value is greater. When a value of 5 or less is originally assigned to N,
xecution proceeds to the next line, with N retaining its original value. Line 50, checks whether
not the value assigned to N is zero. In the case of zero, program execution jumps to line
30, while execution proceeds to the next line (line 60) when N is any other value besides zero.
Line 50 can also be abbreviated as follows:

50 IF N=0 THEN 130

Program areas can also be specified as jump destinations:

IF A = 1 THEN GOTO #2 (Program stored in program area 2 executed when A equals 1)

⑥ GOTO

The GOTO command (lines 50 and 120) performs a jump to a specified line number or program area. The GOTO statement in line 120 is an unconditional jump, in that execution always returns to line 10 of the program whenever line 120 is executed. The GOTO statement in line 50, on the other hand, is a conditional jump, because the condition of the IF ~ THEN statement must be met before the jump to line 130 is made.

* Program area jumps are specified as GOTO #2 (to jump to program area 2).

⑦ FOR/NEXT

The FOR/NEXT combination (lines 60 and 90) forms a loop. All of the statements within the loop are repeated the number of times specified by a value following the word "TO" in the FOR statement. In the example being discussed here, the loop is repeated N number of times, with the value of N being entered by the operator in line 30.

⑧ READ/DATA/RESTORE

These statements (lines 70, 180, 20) are used when the amount of data to be handled is too large to require keyboard input with every execution. In this case, data are included within the program itself. The READ command assigns data to variables, the DATA statement holds the data to be read, and the RESTORE command is used to specify from which point the read operation is to be performed.

In the sample program here, the READ command reads the number of data items specified by the input for variable N. Though the DATA statement holds only five data items, the RESTORE command in line 20 always returns the next read position to the first data item, the READ statement never runs out of data to read.

⑨ GOSUB/RETURN

The GOSUB/RETURN commands (lines 100 and 170) are used for branching to and from subroutines. Subroutines (lines 140 through 170) are actually mini programs within the main program and usually represent routines which are performed repeatedly at different locations within the main program. This means that GOSUB/RETURN makes it possible to write the repeated operation once, as a subroutine, instead of writing each time it is needed within the main program.

EXAMPLE:

```
120  GOSUB 1000
      :
370  GOSUB 1000
      :
```

Execution of the RETURN statement at the end of a subroutine returns execution of the program back to the statement following the GOSUB command. In this sample program, execution returns to line 110 after the RETURN command in line 170 is executed.

* GOSUB routines can also be used to branch to other program areas, as in GOSUB #3 (branches to program area 3). Note, however, that a return must be made back to the original program area using the RETURN command before an END command is executed.
* See PART 10 COMMAND REFERENCE for further details on BASIC commands.

6-6 OPERATORS

The following are the operators used for calculations which involve variables.

Operators	Arithmetic operators	Signs	+, −
		Addtion	+
		Subtraction	−
		Multiplication	*
		Division	/
		Power	^
		Integer division	¥
		Integer remainder of integer division	MOD
	Relational operators	Equal to	=
		Does not equal	< >, > <
		Less than	<
		Greater than	>
		Less than or equal to	= <, < =
		Greater than or equal to	= >, > =
	Logical operators	Negation	NOT
		Logical product	AND
		Logical sum	OR
		Exculsive OR	XOR
	String operator		+

. Arithmetic Operators (+, −, *, /, ^, ¥, MOD)

Fractions are truncated in ¥ and MOD calculations, when the operands on both sides of the operator are not integers.

In ¥ and MOD calculations, the division is performed with the absolute values of both operands. In integer division (¥), the quotient is truncated to an integer. With the MOD operator, the remainder is given the sign of the dividend.

EXAMPLES:

- 15¥7 = − 2
- 15 MOD7 = 1

− 15÷7 = −2……−1 (−2: −15¥7, −1: −15MOD7)

The length of an expression is limited to 255 characters.

. Relational Operators (=, < >, > <, <, >, = <, < =, = >, > =)

Relational operations can be performed only when the operators are both strings or numeric values.

With strings, character codes are compared one-by-one from the beginning of the strings. This is to say that the first position of string A is compared with the first position of string B, the second position of string A with the second position of string B, etc. The result of the comparison is based upon the character codes of the first difference between the strings detected, regardless of the length of the strings being compared.

EXAMPLES:

STRING A	STRING B	RESULT	
ABC	ABC	A = B	
ABC	ABCDE	A < B	
ABC	XYZ	A < B	(character code for A less than that for X)
XYZ	ABCDE	A > B	(character code for X greater than that for A)

A result of −1 is returned when the result of a relational operation is true (conditions met), while 0 is returned when the result is false (conditions not met).

EXAMPLE:

```
10  PRINT 10>3................................−1 returned because 10>3 is true
20  PRINT 7<1..................................0 returned because 7<1 is false
30  PRINT "ABC" = "XYZ"............0 returned because ABC = XYZ is false
40  END
```

3. Logical Operators

The operands of logical operations are truncated to integers and the operation is performed bit-by-bit to obtain the result.

Negation

X	NOT X
0	1
1	0

Logical product

X	Y	X AND Y
0	0	0
0	1	0
1	0	0
1	1	1

Logical sum

X	Y	X OR Y
0	0	0
0	1	1
1	0	1
1	1	1

Excluxive OR

X	Y	X XOR Y
0	0	0
0	1	1
1	0	1
1	1	0

4. String Operators (+)

Strings may be concatenated using a + sign.
The result of the operation (including intermediate results) may not exceed 255 characters.

EXAMPLE:
A$ = "AD" + "1990"

The above example results in the string "AD1990" being assigned to variable A$.

Order of Operations

thmetic, relational and logical operations are performed in the following order of cedence:

(,)
Scientific function
Power
Sign (+ , −)
*, /, ¥, MOD
Addition and subtraction
Relational operators
NOT
AND
OR, XOR

erations are performed from left to right when the order of precedence is identical.

7 CONSTANTS AND VARIABLES

7-1 Constants

e following shows the constants included in the sample program on page 46:

OGRAM	CONSTANTS
R = 15	15
INPUT "H = " ; H	"H = "
V = PI * R ^ 2 * H	2
PRINT "V = " ; V	"V = "

these, 15 and 2 are numeric constants, while "H = " and "V = " are string constants.

ımeric Constants

Numeric Notation
Decimal notation
Hexadecimal notation

Numeric Value Precision
Internal numeric operations
12-digit mantissa, 2-digit exponent (PI = 11 digits: 3.1415926536; displayed in 10 digits: 3.141592654)
Results
10-digit mantissa, 2-digit exponent (exponent rounded to 10 digits)
Number of characters per line
255 characters per line
Result Display

Integers less than 1×10^{10}	:	Integer display
Decimal portion less than 11 digits	:	Decimal display
Other	:	Exponential display
Display rounding	:	Results are rounded off at the 10th digit and displayed.

String Constants

Strings within quotation marks (i.e. "ABC", "H=")
Closing quotation marks at the end of a line may be omitted (10 PRINT "TEST" can be written 10 PRINT "TEST)
Multiple strings can be connected with a "+" sign.

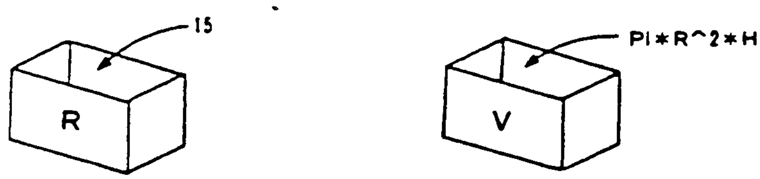
6-7-2 Variables

Numeric Variables

The following shows the numeric variables included in the sample program on page 46:

PROGRAM	NUMERIC VARIABLES
20 R=15	R
30 INPUT "H=" ; H	H
40 V=PI*R^2*H	V

Numeric variables are so named because their contents are handled as numbers. Numeric variable names can be up to 15 characters long, and are used within programs to store calculation results or constants in memory. In the sample program, the value 15 is stored in H, while V, which is the result of the calculation, holds the value which represents the volume of the cylinder. As can be seen, assignment to a variable is performed using the "=" symbol. This differs from an equal sign in that it declares that what is to the right should be assigned to what is to the left. Actually, a variable can be thought of as a kind of box as illustrated below:



String Variables

Another type of variable is known as a string variable, which is used to store character string data. String variable names are indicated by "$" following the name.

EXAMPLE:

```
10  A$="AD"...............................Assigns "AD" to string variable A$.
20  INPUT "YEAR=" ; B$............Assigns keyboard input to variable B$.
30  C$=A$+B$..............................Assigns combination of A$ and B$ to C$.
40  PRINT C$................................Displays contents of C$.
50  END
```

In the above example program, entering a year such as 1990 in line 20 results in a display of AD1990 in line 40.

- With string variables, "+" can be used to connect two strings.
- Note here that strings cannot be assigned to numeric variables such as A, and numeric values cannot be assigned to string variables such as A$.

ay Variables

h numeric variables and string variables can store only one data item per variable. Be-se of this, large amounts of data are better handled using array variables (usually referred s simply "arrays"). Before an array variable can be used within a program, a DIM state-it must appear at the beginning of the program to "declare" to the computer that an y variable is to be employed.

AMPLE:

clare array variable A for storage of 21 data items.

```
DIM A (20)
```

he above format is used to declare "ARRAY VARIABLE NAME (NUMBER OF LEMENTS)".

declared value of 20 makes it possible to store 21 data items (see page 63 for details).

AMPLE:

d the sum (X) and the sum of the squares (Y) for the following 10 data items:

12, 9, 11, 13, 14, 11, 12, 9, 10

following program would be required to perform the calculation if only simple numeric ables are used:

```
A1 = 10 : A2 = 12 : A3 = 9 : A4 = 11 : A5 = 13      } Assigns values to each variable
A6 = 14 : A7 = 11 : A8 = 12 : A9 = 9 : A10 = 10     }
X = A1 + A2 + A3 + A4 + A5 + A6 + A7 + A8 + A9 + A10......Calculates sum
Y = A1^2 + A2^2 + A3^2 + A4^2 + A5^2 + A6^2 +
    A7^2 + A8^2 + A9^2 + A10^2..........................Calculates sum of squares
```

program becomes much simpler when an array is used.

```
DIM A (10)....Declares array
A (1) = 10 : A (2) = 12 : A (3) = 9 : A (4) = 11 : A (5) = 13    } Assigns values to array
A (6) = 14 : A (7) = 11 : A (8) = 12 : A (9) = 9 : A (10) = 10   }
X = 0 : Y = 0
FOR I = 1 TO 10                   }
X = X + A ( I ) : Y = Y + A ( I )^2   } Calculates sum and sum of squares
NEXT I                            }
```

irst glance, the array may appear to be rather troublesome to use, but it actually makes gramming simpler when large volumes of data are being assigned.

EXAMPLE:
100 data items

Numeric variables

```
10   A1=61 : A2=38 : A3=90 : A4=37 : A5=99
20   A6=12 : A7=17 : A8=94 : A9=39 : A10=75
30   A11=24 : A12=84 : A13=46 : A14=18 : A15=55
40   A1.=24 : A1. ... A18=51 : A19=91 : A20=3.
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
150  A71=31 : A77=69 : A7. ... =30
160  A76=40 : A77=69 : A78=51 : A19=91 : A20=30      Assigns values to variables
170  A81=91 : A82=46 : A83=23 : A84=37 : A85=84
180  A86=65 : A87=23 : A88=98 : A89=51 : A90=30
190  A91=57 : A92=78 : A93=16 : A94=39 : A95=46
200  A96=59 : A97=24 : A98=32 : A99=74 : A100=47
210  X=A1+A2+A3+............+A49+A50                 Calculates sum
220  X=X+A51+A52+..........+A99+A100
230  Y=A1^2+A2^2+.........+A39^2+A40^2
240  Y=Y+A41^2+A42^2+..........+A79^2+A80^2          Calculates sum of squares
250  Y=Y+A81^2+A82^2+..........+A99^2+A100^2
```

Array

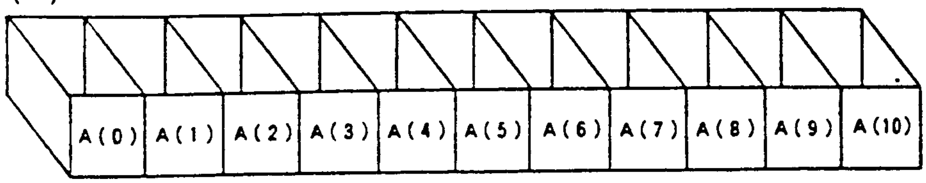
```
10   DIM A (100)                                     Declares array
20   FOR I=1 TO 100 : READ A (I) : NEXT I            Assigns values to array
30   X=0 : Y=0
40   FOR I=1 TO 100                                  Calculates sum and sum of squares
50   X=X+A (I) : Y=Y+A (I)^2
60   NEXT I
70   DATA 61, 38, 90, 37, 99
80   DATA 12, 17, 94, 39, 75
90   DATA 24, 84, 46, 18, 55
100  DATA 46, 65, 51, 91, 30
     ... 26, 11, 88, 78                              Data
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
220  DATA 47, ...
230  DATA 91, 46, 28, 31, 64
240  DATA 65, 23, 98, 51, 30
250  DATA 57, 78, 16, 39, 46
260  DATA 59, 24, 32, 74, 47
```

A look at these programs reveals that an increase in data entails virtually no change in the portion which calculates the sum and sum of squares. The only changes would be in lines 10, 20, and 40, where the constant would be changed from 10 to 100.

Again, the concept of the array can be better grasped by thinking of them as boxes. Previously, a simple variable was described as a single box. Arrays, on the other hand, would be a series of numbered boxes which form a set.

AMPLE:

ay A (10)



llustrated above, the array A(10) actually contains a total of eleven boxes, numbered from) through A(10), with each box being capable of holding a different value. The actual term d to refer to a box is "element". Recalling a stored value is performed by simply specify- the corresponding element number.

\MPLE:

:all value stored in element 4 of array A

(4)

r

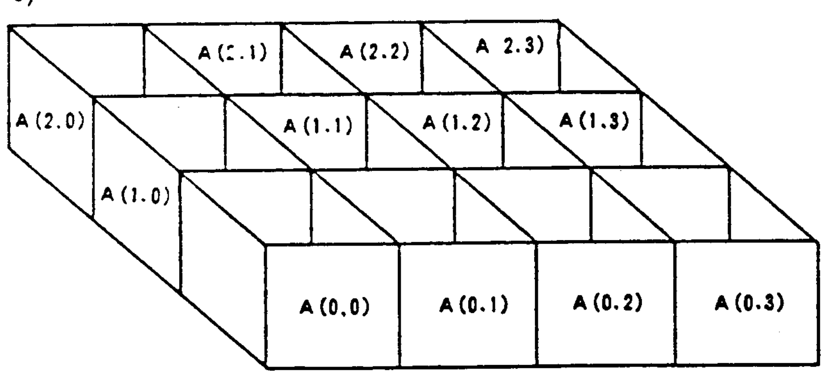
4 : Y = A (X)

value which specifies an element in an array (4 above) is called a subscript.

l now, the only arrays covered have been those formed by a single line of elements or xes". These are known as "one-dimensional" arrays. Arrays may also contain more than dimension with elements connected vertically and horizontally into two-dimensional and e-dimensional arrays.

MPLE:

A (2, 3)



declaration in this example sets up an array of three lines and four columns, making pable of storing 12 different values.

ieric arrays and string arrays

vith simple variables, arrays can also be declared to hold strings by using the "$" sym- ollowing the array variable name. Again remember, numeric values cannot be assigned tring arrays and strings cannot be assigned to numeric arrays.

EXAMPLE:
The following procedure is used to declare an array and store the data for five individuals and their points scored during a certain game.

String array N$(5) declared for names
Numeric array P(5) declared for points

```
10  DIM N$ (5), P (5)......................Declaration of arrays to store names and points
20  FOR I=1 TO 5
30  READ A$, X
40  N$ ( I )=A$............................Stores names to string array
50  P ( I )=X..............................Stores points to numeric array
60  NEXT I
70  END
80  DATA SMITH, 70, BROWN, 68, JONES, 87, CARTER, 80, MILLS, 74
```

6-7-3 Summary

Variable Types

The three following types of variables are available for use with this unit.

1. Numeric variables (up to 12-digit mantissa)		A, a, NUMBER, POINTS
2. String variables (up to 255 characters)		A$, STRING$
3. Array variables	Numeric array	A (10), XX (3, 3, 3)
	String array	A$ (10), ARRAY$ (2, 2)

Variable Names

- Variabel names can consist of upper, lower case or numeric characters, but a numeric character cannot be used in the first position of the variable name (i.e. 1AE, 3BC$ are illegal).
- Reserved words (see page 400) cannot be used as the leading characters of a variable name (i.e. RUNON, LIST1$ are illegal).
- The maximum length of a variable name is 15 characters.

Arrays

1. Arrays are declared by DIM statements.
2. Elements described by subscripts which are integers greater than 0. Fractions are disregarded.
3. The number of dimensions is limited by stack capacity.
4. The maximum value of subscripts is limited by memory capacity.

Variable/Array Application

1. Variables and arrays can be used jointly by all program areas.
2. Arrays cannot be used unless first declared using the DIM statement.

unting Bytes Used by Variables

ɜ following outlines the number of bytes reserved when a variable appears the first time ɿin a program.

umeric Variables

ɿiable name length + 12) bytes in variable area

tring Variables

ɿiable name length + 4) bytes in variable area and (string length + 1) bytes in string area as are reserved for array variables when the array is declared by the DIM statement.

umeric Array Variables

ɿiable name length + 4) + (array size × 8) + (dimension × 2 + 1) bytes in variable area

AMPLE:

1 XYZ (3, 3, 5, 2)

ne : 3
ɜ : 4 × 4 × 6 × 3 = 288
ɿension : 4
culation : (3 + 4) + (288 × 8) + 4 × 2 + 1 = 2320 bytes

ring Array Variables

ɿiable name length + 4) + (array size) + (dimension × 2) bytes in variable area.
lengths of individual strings are required in the variable area when strings are assigned ɿe array.

AMPLE:

DIM AB$(3, 3)
AB$ (0, 0) = "*****"

ne : 2
ɜ : 4 × 4 = 16
ension : 2
culations : (2 + 4) + 16 + (2 × 2) + 5 bytes

culating Program Length

following shows points which must be considered when calculating memory requireɿts for programs.

ɜ numbers : 2 bytes per line number, regardless of number length (1 ~ 65535)
ɿmands : 2 bytes per command
ctions : 2 bytes per function
ɿeric/alphabetic
ɿacters : 1 byte per character (spaces also counted as characters)
key : 1 byte per EXE key operation at end of program line (for storage of line)
yte added to sum of the above

AMPLE:

A = SIN X

ne number) + 1 (space following line number) + 1 (A) + 1 (=) + 2 (SIN) + 1 (space) 1 (X) + 1 (EXE) + 1 = 11

s calculation indicates that a total of 11 bytes are required for storage of the above program. ɿe space following the line number is added automatically.

6-8 PROGRAM SAVE AND LOAD

The following save and load procedures can only be performed when the FA-6 interface unit is used.

6-8-1 Program Save

Programs stored in the memory of the unit are protected by the memory back up battery even when the power of the unit is switched OFF. The entire contents of the memory, however, are deleted whenever both the main power supply batteries and memory back up batteries are removed from the unit at the same time, or when the NEW ALL command is executed. Program area contents can be stored onto standard cassette tapes to protect against loss of important data, or to make room for further programming when all program areas are full. The following two commands are available for such save operations.

SAVE : Saves contents of current program area.
SAVE ALL : Saves entire contents of all program areas.

EXAMPLE:
Executing SAVE in this case saves the contents of program area P0, while SAVE ALL would save the contents of program areas P0 through P9.

SAVE EXE

Display	
SAVE	(Saves program in program area 0)
SAVE Ready P0	(Save complete)

Filenames up to eight characters long can also be assigned to programs stored on cassette tapes using the SAVE and SAVE ALL commands.

SAVE "BASIC" EXE

Display	
SAVE"BASIC"	(Saves program under filename "BASIC")
SAVE"BASIC" Ready P0	(Save complete)

6-8-2 Program Verify

The VERIFY command makes it possible to verify whether or not the program saved using SAVE or SAVE ALL was copied correctly to the cassette tape.

EXAMPLE:
Verify correct save of the program BASIC

VERIFY "BASIC" EXE

Display	
VERIFY"BASIC"	(Verification of saved program)
BASIC B	(Finds specified program and verifies)
BASIC B Ready P0	(Verification complete)

If the Ready prompt does not appear after some time, check whether or not the filename entered with the VERIFY command is correct. If it is correct, adjust the volume level of the cassette recorder being used and repeat the verification procedure.

Display
P0 error 0 Ready P0

ıe error message illustrated above indicates that the program was not saved correctly. In s case, check the following items:

/erify the program again, this time appending "CAS1:" before the filename (VERIFY "CAS1 BASIC" in the above example).
Ensure that connections between the computer and cassette tape recorder are correct and secure.
Ensure that the volume level of the recorder is set to in the vicinity of its maximum.
Check whether the cassette tape is damaged.
Check whether the recorder heads are soiled.

ote also that an error will be generated if a program exists on the tape with the same name that currently present in computer memory, but the contents of the two programs are ferent.
The VERIFY command automatically determines whether the program being checked was saved using the SAVE or SAVE ALL command.

8-3 Program Load

ograms stored on cassette tapes using the SAVE and SAVE ALL commands can be load- into the computer using the LOAD and LOAD ALL commands.

AMPLE:
ad the program "BASIC" from cassette tape into memory

AD EXE	LOAD	(Program load command)
	BASIC B	(Program filename)
	Ready P0	(Load complete)

te that executing the LOAD and LOAD ALL commands while programs are already stored memory deletes the current memory contents.

e LOAD ALL command can be used to load programs to all of the program areas (P0 – P9). ecifying a filename in the LOAD and LOAD ALL commands causes the unit to search for specified filename for loading into memory. The following table shows the relationship ween the LOAD, LOAD ALL, SAVE and SAVE ALL commands.

	LOAD	LOAD "filename"	LOAD ALL	LOAD ALL "filename"
SAVE	o	×	×	×
SAVE "filename"	o	o	×	×
SAVE ALL	×	×	o	×
SAVE ALL "filename"	×	×	o	o

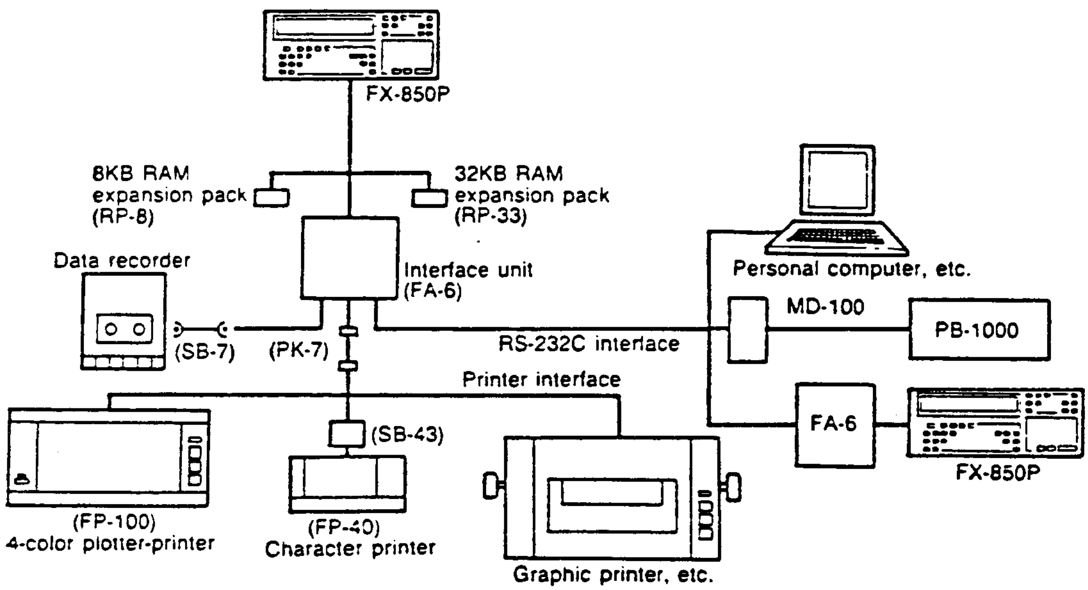
TE:
e PART 7 PERIPHERAL DEVICES for details on using the SAVE and LOAD commands.

PART 7

PERIPHERAL DEVICES

A variety of peripheral devices are available for connection to this unit to provide even more computing power.

System Configuration



7-1 CASSETTE INTERFACE UNIT FA-6

7-1-1 Features

The FA-6 is an interface unit which makes it possible to use a cassette tape recorder as an external data storage device. Besides a cassette interface, the FA-6 is also equipped with an RS-232C interface and a Centronics standard printer interface.
The RS-232C interface connector, printer interface, cassette interface and an AC adaptor jack are located on the back of the FA-6. The battery compartment is located on the bottom of the unit. Batteries are loaded by removing the battery compartment cover and inserting batteries while ensuring that their polarities (⊕ ⊖) are as illustrated in the compartment.

ıe error message illustrated above indicates that the program was not saved correctly. In s case, check the following items:

/erify the program again, this time appending "CAS1:" before the filename (VERIFY "CAS1 BASIC" in the above example).
Ensure that connections between the computer and cassette tape recorder are correct and secure.
Ensure that the volume level of the recorder is set to in the vicinity of its maximum.
Check whether the cassette tape is damaged.
Check whether the recorder heads are soiled.

ɔte also that an error will be generated if a program exists on the tape with the same name that currently present in computer memory, but the contents of the two programs are ferent.
The VERIFY command automatically determines whether the program being checked was saved using the SAVE or SAVE ALL command.

8-3 Program Load

ɔgrams stored on cassette tapes using the SAVE and SAVE ALL commands can be load- into the computer using the LOAD and LOAD ALL commands.

AMPLE:
ad the program "BASIC" from cassette tape into memory

ıAD EXE

```
LOAD                      (Program load command)
BASIC        B            (Program filename)
Ready P0                  (Load complete)
```

te that executing the LOAD and LOAD ALL commands while programs are already stored memory deletes the current memory contents.

e LOAD ALL command can be used to load programs to all of the program areas (P0 – P9). ecifying a filename in the LOAD and LOAD ALL commands causes the unit to search for specified filename for loading into memory. The following table shows the relationship ween the LOAD, LOAD ALL, SAVE and SAVE ALL commands.

	LOAD	LOAD "filename"	LOAD ALL	LOAD ALL "filename"
SAVE	o	×	×	×
SAVE "filename"	o	o	×	×
SAVE ALL	×	×	o	×
SAVE ALL "filename"	×	×	o	o

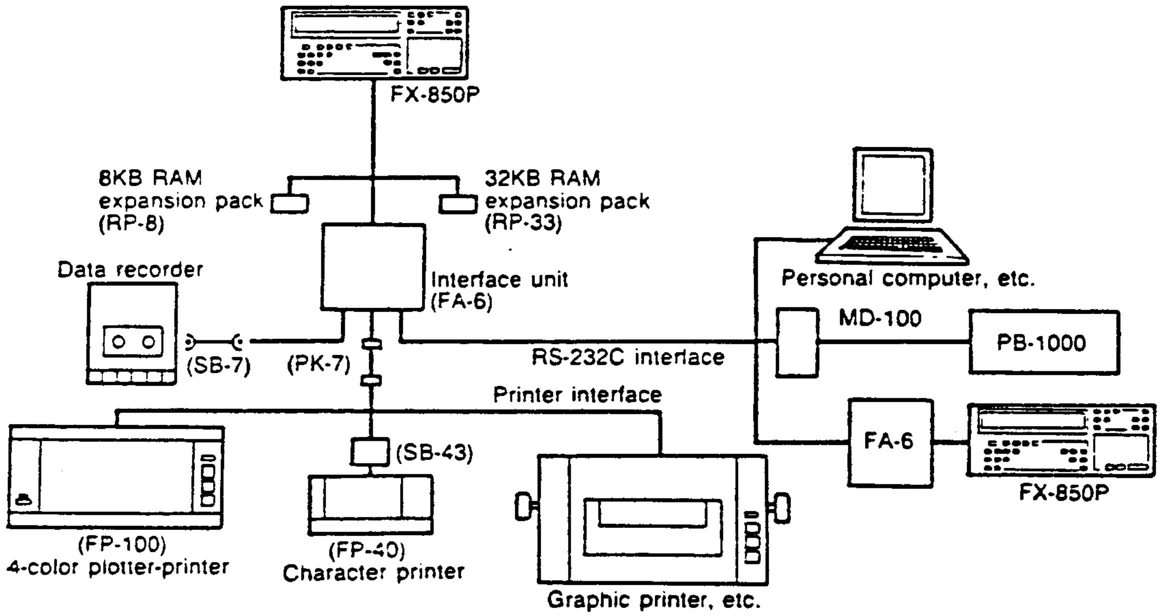
ıTE:
e PART 7 PERIPHERAL DEVICES for details on using the SAVE and LOAD commands.

PART 7

PERIPHERAL DEVICES

A variety of peripheral devices are available for connection to this unit to provide even more computing power.

System Configuration

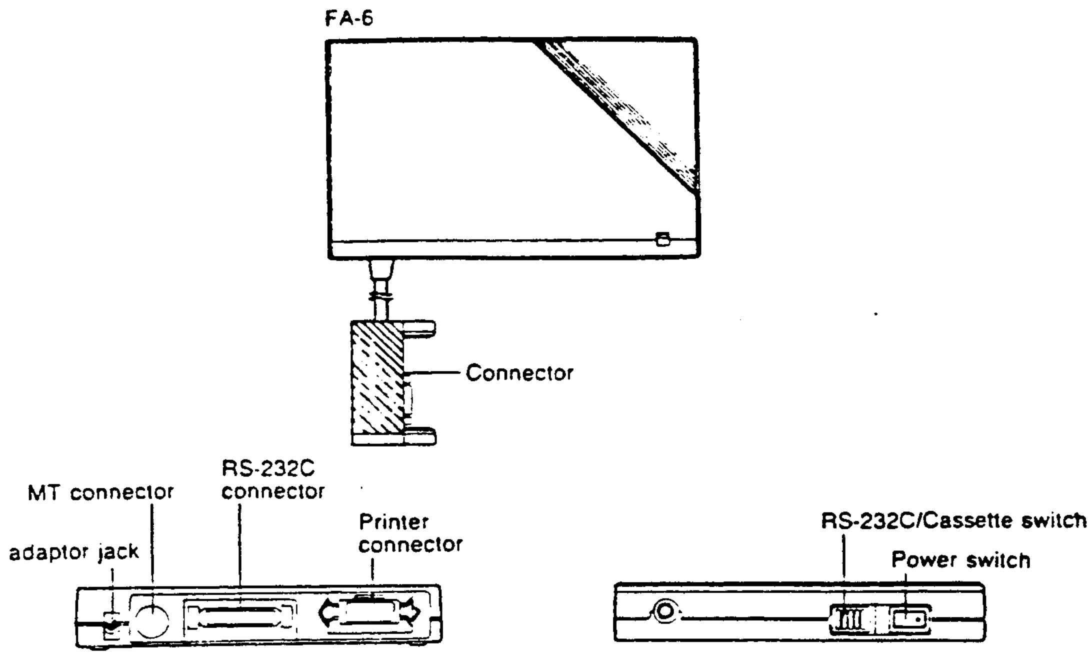


7-1 CASSETTE INTERFACE UNIT FA-6

7-1-1 Features

The FA-6 is an interface unit which makes it possible to use a cassette tape recorder as an external data storage device. Besides a cassette interface, the FA-6 is also equipped with an RS-232C interface and a Centronics standard printer interface.

The RS-232C interface connector, printer interface, cassette interface and an AC adaptor jack are located on the back of the FA-6. The battery compartment is located on the bottom of the unit. Batteries are loaded by removing the battery compartment cover and inserting batteries while ensuring that their polarities (⊕ ⊖) are as illustrated in the compartment.



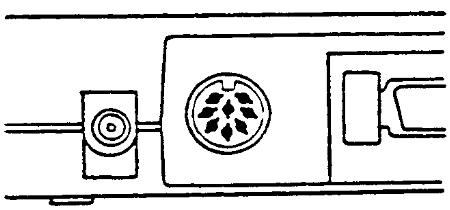
-2 Connections

;ure that the power of both the computer and the interface unit is switched OFF before ·mpting connections. Once connected, power should be switched ON for the computer . and then the interface unit.

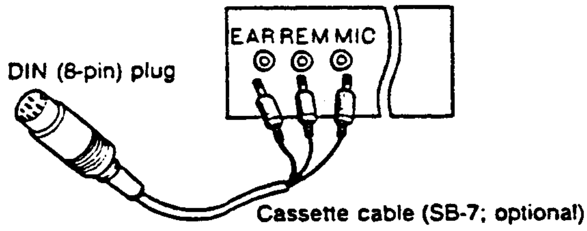
·3 Cassette Interface

: cassette interface is used for connection of a cassette recorder to make it possible to ·e programs on cassette tapes and to later reload the programs into computer memory. ınection to the cassette tape recorder is accomplished using the optional SB-7 connect- cable. The red plug is inserted into the MIC or LINE IN jack (labeling differs according ype of recorder used) of the recorder, while the white plug is inserted into the EAR or E OUT jack of the recorder. The black plug is inserted into the REM jack of recorders ;ipped with a remote function.

FA-6 cassette interface unit terminals



Data recorder terminals



EAR : Output (earphone) - White plug
REM: Remote control - Black plug
MIC : Input (microphone) - Red plug

NOTE:
The remote plug is not used when the recorder being used is not equipped with a remote function.
The recorder should be set to its RECORD mode when performing recording of programs or data. For program loading, set the recorder to its PLAYBACK mode after executing the LOAD command.

Single Program Save

SAVE "file descriptor" [EXE] (file descriptor may be omitted)
The file descriptor can contain any symbols, characters, or numbers (except quotation marks).

EXAMPLE:
SAVE "CAS0: AD1990" [EXE]
* "CAS0:" may be omitted.

Single Program Load

LOAD "file descriptor" [EXE] (file descriptor may be omitted)
If a file descriptor is not specified, the unit loads the first program found on the tape.

EXAMPLE:
LOAD "CAS0: AD1990" [EXE]
* "CAS0:" may be omitted.

Loading/Saving All Programs

SAVE ALL "file descriptor" [EXE] }
LOAD ALL "file descriptor" [EXE] } (file descriptor may be omitted)

The SAVE ALL command saves all of the programs stored in program areas P0 through P9 to cassette tape. LOAD ALL, on the other hand, loads programs saved using the SAVE ALL command. The LOAD ALL command also clears any contents present in the program areas and replaces them with the programs from the cassette tape.

Saving and Loading Data Bank Data

The SAVE# and LOAD# commands are used for the saving and loading of memo data stored in the DATA BANK. The procedure for using these commands is identical to that described for SAVE and LOAD above.

Saved File Verification

The VERIFY command checks whether the program or data saved to the cassette tape matches exactly the current memory contents.
VERIFY "file descriptor" (file descriptor may be omitted)

Tape Recorder Operation

The recorder should be set to its RECORD mode before the SAVE command is executed. For program loading, set the recorder to its PLAYBACK mode and then execute the LOAD command.
The SAVE, SAVE ALL, LOAD, LOAD ALL, SAVE#, and LOAD# commands cannot be used in the CAL mode or within programs. They can, however, be used during manual operations in the BASIC mode.
* See PART 9 FILE HANDLING FUNDAMENTALS for details on file descriptors.

1-4 RS-232C Interface

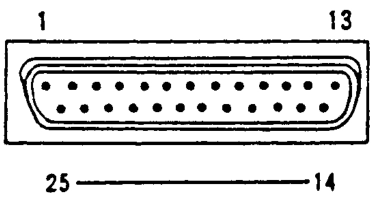
;-232C Switch

e RS-232C interface can be used for data communications after the RS-232C/Cassette itch is set to RS-232C.

›ecifications

ımmunication method : Start-stop (asynchronous) full-duplex mode only
ansmission speed : 150, 300, 600, 1200, 2400, 4800 baud
rity bit : Odd, Even, None
aracter bit length : 7 or 8 bits
ɔp bits : 1 or 2 bits
'S signal control : Control/no control
;R signal control : Control/no control
) signal control : Control/no control
.sy control : XON/XOFF control/no control
ɔut/output code system : SI/SO control/no control

ı Configuration

Terminal number	Signal name	Pin connection	I/O
1	FG		
2	TXD		OUTPUT
3	RXD		INPUT
4	RTS		OUTPUT
5	CTS		INPUT
6	DSR		INPUT
7	GND		
8	DCD		INPUT
9	NC		
10	NC		
11	NC		
12	NC		
13	NC		
14	NC		
15	NC		
16	NC		
17	NC		
18	NC		
19	NC		
20	DTR		OUTPUT
21	NC		
22	NC		
23	NC		
24	NC		
25	NC		

See PART 9 FILE HANDLING FUNDAMENTALS for details on using the RS-232C interface.

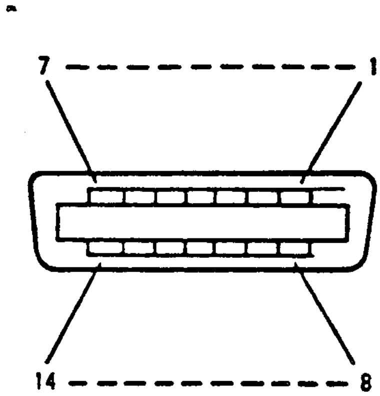
7-1-5 Centronics Interface (Printer Interface)

General

The Centronics interface is used to output data processing results or program lists to a printer. Any Centronics printer can be connected to the computer via the FA-6 interface unit.

Pin Configuration

Terminal number	Signal name	Pin Connection
1	$\overline{PSTB}$	
2	PDB0	
3	PDB1	
4	PDB2	
5	PDB3	
6	PDB4	
7	PDB5	
8	PDB6	
9	PDB7	
10	NC	
11	BUSY	
12	NC	
13	NC	
14	GND	



BASIC Printer Commands

Command	Function
LLIST	Outputs program contents to printer
LPRINT	Outputs specified characters to printer
TAB	Outputs spaces up to a specified position to printer

PRT ON Mode

Setting the unit to the PRT ON mode ([MODE] [7]) and then executing the PRINT, LIST or VARLIST commands prints out the results of such command execution and object data specified by the TRON command on the printer. Including [M] [O] [D] [E] [7] within a program prints out all contents of subsequent print commands. The PRT ON mode can be canceled by [MODE] [8] or [M] [O] [D] [E] [8] (PRT OFF).

* See the FA-6 manual for details on its proper operation.

2 PLOTTER-PRINTER (FP-100)

: FP-100 is a four-color plotter-printer capable of printing on A-4 size paper.
: FP-100 has both a character mode and a graphics mode which makes it possible to
:t on virtual any type of computer output.

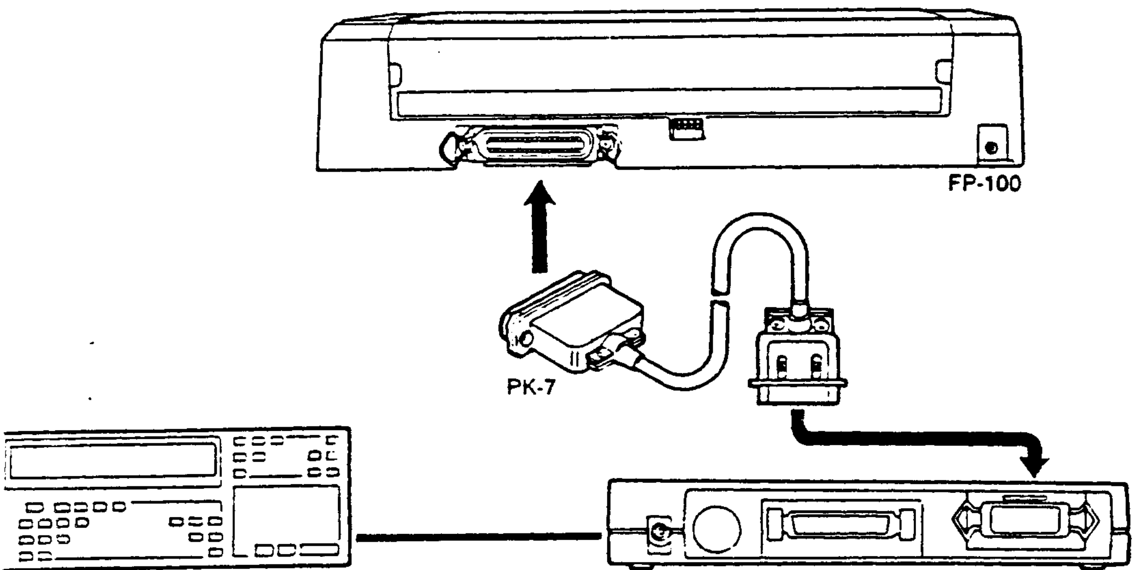
iracter mode : Program lists, calculation results
phics mode : Graphics produced by graphics commands

-1 Specifications

ɔlor printing in black, red, blue, green
ʳacter effects : Italics
.t resolution : 0.1mm/step
ʼer width : Postcard size to letter size
ɪracter size : 1.0mm × 1.2mm (S0, 0) – 16.0mm × 19.2mm (S15, 15); 256 types

:-2 Connections

ɜ unit is connected to FP-100 via the FA-6 interface unit and PK-7 printer cable.
se only CASIO PK-7 printer cable



2-3 Data Printing

ɔgram lists are output to the printer using the BASIC LLIST command. Execution of the
ST command prints out the currently accessed program.
e LPRINT command is used to print out data within a program, while execution of the
ST# command in the BASIC mode ([MODE] [1]) prints data bank contents.
ɜee the FP-100 manual for details on its proper operation.

7-3 CHARACTER PRINTER (FP-40)

The FP-40 can be used to print out data or program lists of programs written on the FX-850P.

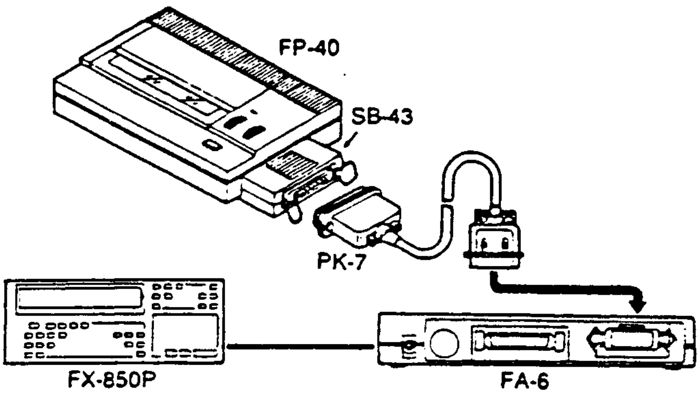
* Addition of an optional interface pack also makes it possible to use the FP-40 with PB-100 series and PB-700 series computers.

7-3-1 Specifications

Print method	:	Thermal print system (non-impact)
Columns	:	40 standard (normal mode)
		80 maximum (80CHR mode)
Print speed	:	Approximately 0.65 lines/sec (normal mode)
Paper feed	:	1/6 inch or 1/9 inch
Roll paper	:	Width 112mm, o.d. 30mm maximum; thermal paper (TRP-112)

7-3-2 Connection

This unit is connected to the FP-40 via the FA-6 interface unit, PK-7 printer cable, and SB-43 interface pack.

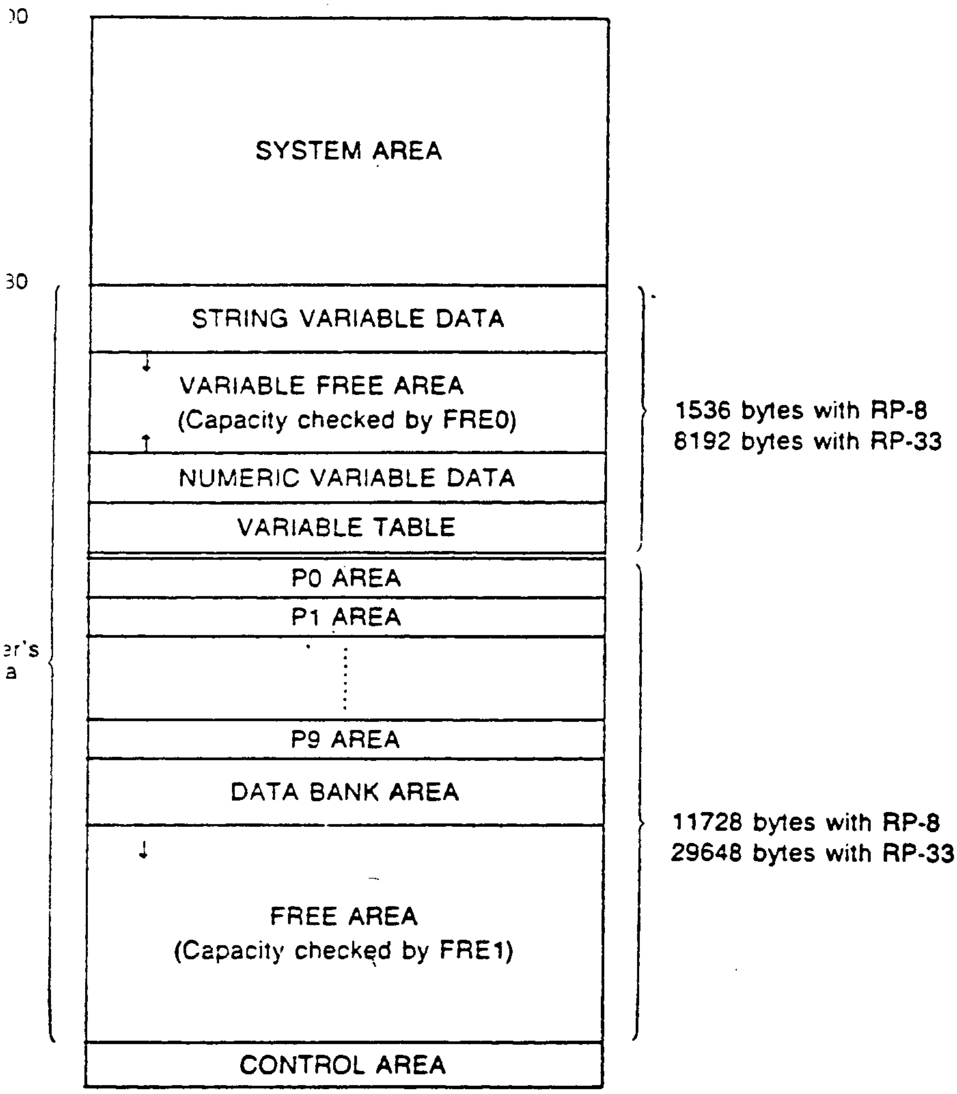


* See the FP-40 manual for details on its proper operation.

ł RAM EXPANSION PACK (RP-8 (8KB)), (RP-33 (32KB))

: unit comes equipped with a standard RAM of 8K bytes. RAM expansion packs are also ɔnally available for larger programs and for handling larger quantities of data. Addition ıe RP-8 RAM pack expands memory capacity to 16K bytes, while the RP-33 RAM pack ands memory to 40K bytes.

-1 Expanded Memory Map



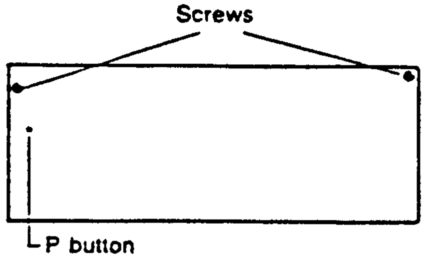
7-4-2 Handling RAM Packs

Preparation

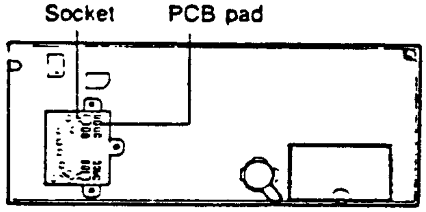
Static electrical charges can damage internal circuitry of RAM packs. Be sure to touch a door knob or some other metal fixture to discharge static electricity before handling RAM packs.

Procedure

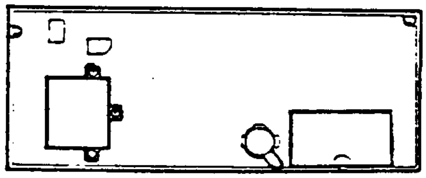
① Switch the power of the unit OFF.
② Remove the back cover of the unit after removing the two screws holding it in place.



③ Insert the RAM pack into the socket provided in the unit and fasten it in place using the three screws provided.
* Never touch the RAM pack connector or PCB pad.



④ Replace the back cover of the unit and fasten it in place using the two screws.
⑤ Switch the power of the unit ON and press the P button. Next, press the ALL RESET button and switch the power of the unit OFF and then ON again.



* Failure to press the P button and ALL RESET button after inserting or removing the RAM pack can result in altered unit memory contents.
* Dirt, dust, or finger prints on the RAM pack connector or PCB pad can result in poor connection and malfunction. Never touch connectors.
* Be sure to store RAM packs in their original cases when removed from the computer and store in an area free of dirt or dust.

PART 8

PB-100 SERIES COMPATIBILITY

; unit is capable of loading data and programs written for the PB-100 series* computers of executing PB-100 programs. Certain special commands are required, however, to w program compatibility.

1e PB-series includes the following models:
B-100, PB-100F, PB-110, PB-220, PB-240, PB-410, FX-720P, FX-730P, FX-770P, FX-785P, X-790P

1 PB-100 SERIES PROGRAM INPUT/EDITING

: following conversions are required to allow execution of PB-100 series programs on this :.

PB-100 SERIES	FX-850P
AT, STATLIST, STATCLEAR)X, EOY	Use library function or create a program.
JT, GET	Change using OPEN "CAS0:" Example 1: PUT$→OPEN "CAS0:" FOR OUTPUT AS#1: PRINT#1, A$:CLOSE Example 2: GET$→OPEN "CAS0:" FOR INPUT AS#1: INPUT #1, A$:CLOSE
	< =, = <
	> <, < >
	> =, = >
	^
	PI

; recommended that the following command conversions also be performed to ensure npatibility between PB-100 series programs and FX-850P programs.

PB-100 SERIES COMMANDS	FX-850P COMMANDS
/AC	CLEAR
F~;~	IF~THEN~
CSR	LOCATE or TAB
KEY, KEY$	INKEY$
RND	ROUND
MID (location, number of characters)	MID$ ($, location, number of characters)
GOTO (numeric expression), GOSUB (numeric expression)	ON~GOTO, ON~GOSUB
MODE 4/5/6	ANGLE 0/1/2
MODE 7/8	LPRINT

Though direct input of PB-100 series characters is not possible with this unit, they can be displayed using the CHR$ function.

EXAMPLE:

Display ≦
PRINT CHR$ (&HE1)

See CHARACTER CODE TABLE on page 395 for details on character codes.

NOTES

- A space must be included before the THEN of an IF ~ THEN statement when the character preceding the THEN is alphabetic.

  Example: IF3 = ATHEN → IF3 = A␣THEN

- The jump destination of a GOTO or GOSUB statement must be enclosed in parentheses when it is an expression beginning with a numeric value.

  Example: GOTO 10*A → GOTO (10*A)

- A space must be included before the TO in the TO portion of a FOR ~ NEXT loop when the initial value is represented by an alphabetic character.

  Example: FORA = BTOC → FORA = B␣TOC

- An INPUT statement followed by a comma will not produce a question mark as a prompt. The question mark is displayed only when the INPUT statement is followed by a semicolon.

8-2 PB-100 SERIES PROGRAM EXECUTION

8-2-1 DEFM Mode

The PB-100 series uses variables A ~ Z and A$ ~ Z$ as arrays. The DEFM mode can be used when executing PB-100 series programs on this unit to use A ~ Z and A$ ~ Z$ in the same manner as PB-100 series computer. [D] [E] [F] [M] **number of variables to be expanded** [EXE] performs variable expansion as PB-100 series computers.

* PB-100 series programs which use DEFM arrays should always be executed in the DEFM mode.

e variables for this unit become as follows when DEFM mode arrays are used.

		DIM mode variables
A (0)		→ A
A (1)	B (0)	→ B
⋮	⋮	⋮
A (25)	B (24)	→ Z
A (26)	B (25)	→ A (0)
A (27)	B (26)	→ A (1)
⋮	⋮	⋮
A$ (0)		→ A$
A$ (1)	B$ (0)	→ B$
⋮	⋮	
A$ (25)	B$ (24)	→ Z$
A$ (26)	B$ (25)	→ A$ (0)
A$ (27)	B$ (26)	→ A$ (1)
⋮	⋮	⋮

, and A$ are independent of each other

e DEFM mode is canceled by declaring an array using the DIM statement or by DIM [EXE].

AMPLE:

M A (m, n, o) [EXE]	Declaring 3-dimensional array A (m, n, o)
M [EXE]	Only cancels DEFM mode.

:FM mode → DIM mode
M mode → DEFM mode
:FM → DEFM n [EXE]

At this time, variables A( ) and A$( ) are erased. This cannot be used within a FOR – NEXT loop.

: number of variables to be expanded

2-2 Using DEFM Statement Arrays

hen defining arrays using the DEFM statement, a variable area should be reserved for e variables used. Failure to do so will result in an OM error (memory over error) when DEFM specified or when the program is executed. Should an OM error occur, the CLEAR state-ent must be used to reserve a variable area before the next DEFM specification. The fol-ving shows calculation of the required variable area size.

variable	: 62 + 3 (bytes)
- Z variables	: (8 + 4) × number of variables (bytes)
; – Z$ variables	: (8 + 5) × number of variables (bytes)
ıriable expansion	: 7 + 8 × number of variables to be expanded + 7 + 9 × number of variables to be expanded

8-2-3 DEFM Mode Displays

Using the DEFM specification in manual (direct key input) execution displays the number of variables in the DEFM array. This DEFM display does not appear when DEFM is specified within a program.

EXAMPLE:

[D] [E] [F] [M] [EXE]	A--Z:26 DEFM:0
[D] [E] [F] [M] 10 [EXE]	A--Z:26 DEFM:10

8-2-4 CLEAR Command, DIM Command, DEFM Command In DIM Mode and DEFM Mode

Executing the CLEAR statement with this unit clears the contents of variables and reserves a variable area. Executing this command in each mode produces the following results.

Operation	DIM Mode	DEFM Mode
[SHIFT] [CLEAR] [EXE] ([V] [A] [C])	Deletes variable contents	Deletes variable contents
[SHIFT] [CLEAR] Variable area [EXE]	Deletes variable contents and reserves variable area	Deletes variable contents, cancels DEFM mode and reserves variable area
[D] [E] [F] [M] [EXE]	Enters DEFM mode and internally executes ERASE A, A$:DIM A( ), A$( )	Displays number of arrays when executed manually
[D] [I] [M] [EXE]	Executes DIM	Cancels DEFM and executes ERASE A, A$

• DEFM specification is not required for FX-790P and FX-730P programs which employ DIM statements.

8-3 LOADING PB-100 SERIES PROGRAMS

The following commands are executed in the BASIC mode to load PB-100 series programs from cassette tape.

PBLOAD : Program file
PBLOAD ALL : All files

Executing these commands loads the programs and automatically modifies them to allow execution on the FX-850P.

EXAMPLE:
PBLOAD from a cassette tape containing the following programs:

TEST 1 (memo file)
TEST 2 (data file)
TEST 3 (all files)
TEST 4 (program file)

P B L O A D SHIFT ␣ T E S T 4 SHIFT ␣ EXE

```
TESTl       MF
```

```
TEST2       VF
```

```
TEST3       AF
```

Search

```
TEST4       PF
```

Load

```
TEST4       PF
Converting...
```

Code conversion

```
Converting...
Ready P0
```

End

8-4 READING PB-100 SERIES DATA

The following commands are available for reading of PB-100 series data files and DATA BANK files:
PBLOAD# (DATA BANK file)
PBGET (data file)

EXAMPLE:
Execute PBGET for filename TEST 5 containing the data $, A, B, C, D.

P B G E T SHIFT ␣ T E S T 5 SHIFT ␣ EXE

```
TEST5       VF
```

```
GET $.4data
Converting...
```

```
Converting...
Ready P0
```

- Program, data, DATA BANK data, and all files saved to cassette tape using this unit cannot be read by PB-100 series computers.

8-5 COMMAND FORMAT

PBLOAD $\left[\begin{Bmatrix}0\\1\end{Bmatrix}\right]$ **["filename"]**

1. 0 : positive phase 1 : reverse phase (default = positive phase)
   Attempt using the phase which is opposite the current setting if problems are experienced during PBLOAD operations.
2. This command loads file under the specified filename into the current program area.
3. This command must be executed in the BASIC mode.
4. Execution of this command reads the specified program from tape and converts it from PB-100 series format to FX-850P/FX-880P format.
5. Execution of this command can be terminated at any time by pressing the BRK key.

PBLOAD ALL $\left[\begin{Bmatrix}0\\1\end{Bmatrix}\right]$ **["filename"]**

1. This command loads all files under the specified filename into program areas P0 throu P9.

Operation is identical to PBLOAD.

PBLOAD# $\left[\begin{Bmatrix}0\\1\end{Bmatrix}\right]$ **["filename"] [, M]**

1. This command reads the data under the specified filename into the DATA BANK.
2. The [, M] specification appends the read data to the end of the data currently stored the DATA BANK.

Operation is identical to PBLOAD.

PBGET $\left[\begin{Bmatrix}0\\1\end{Bmatrix}\right]$ **["filename"]**

1. This command reads the data file contents under the specified filename and assign then to $ variable, A( ) array, or A$( ) array.
2. String data and numeric data are automatically assigned to the proper corresponding array
3. This command can be executed in both the BASIC mode and CAL mode, but must be preceded by array declaration using the DIM statement or DEFM statement.
4. Data are automatically converted (Converting···displayed) before being assigned to variables.

NOTES

- Suspending execution of PBLOAD, PBLOAD ALL, PBLOAD#, and PBGET using the STOP key and then resuming operation should be avoided.
- A PO error (program error) or DA error (data error) is generated when program or data load is unsuccessful using a PB command.
- The following variables are used as work areas when any of the PB commands are executed:
  a, b, c, d, e, f, g, h, i, j, k, l, m, n,
  o, p, q, r, s, t, u, v, w, x, y
  a$, b$, c$, d$, e$, f$, g$, h$, i$, j$, k$, l$, m$, n$,
  o$, p$
- STAT, EOX, EOY, PUT and GET commands are converted to the code ??? when loaded to a PB series computer. Executing programs with this code generates an SN error (syntax error). See section 8-1 for details on manual conversion of code ???.
- Execution of a PB command while in the PRT mode (PRT symbol on display) automatically cancels the PRT mode.

PART 9

FILE HANDLING FUNDAMENTALS

9-1 FILING DEVICES

Besides execution of programs currently stored in memory, this unit can also employ cassette tapes for data and program storage. Data and programs can also be exchanged with other devices via a communications circuit. The OPEN, CLOSE, PRINT#, INPUT#, SAVE, and LOAD commands are used for these purposes.

• Device Names

When using the commands noted above for file interchanges, it is first necessary to specify whether cassette tape or the communications circuit is to be employed. This is known as specifying a "device name". The following table shows the available device names:

DEVICE NAME	MEANING
CAS0:	Cassette tape recorder (positive phase)
CAS1:	Cassette tape recorder (reverse phase)
COM0:	Communications circuit (RS-232C)

• Filenames

Once the device to be used is determined, the next thing to do is assign a "filename" to the file. A filename can be any combination of numeric and alphabetic characters up to eight characters in length.

The unit automatically disregards any filename input exceeding the first eight characters.

* Only one file can be open at any time. A program or subroutine, which successively opens and closes files as they are required, should be prepared when multiple files need to be accessed.

• File Descriptors

A "file descriptor" is actually a combination of a device name and filename. When using the communications circuit, the file descriptor also specifies various communications parameters in addition to the device name and filename.

9-2 FILE DESCRIPTORS

9-2-1 Cassette Tapes

The following shows the file descriptor when a cassette tape is used for file storage.

$$\left\{\begin{matrix}\text{CAS0 :}\\ \text{CAS1 :}\end{matrix}\right\}\left[\begin{matrix}\text{(S)}\\ \text{(F)}\end{matrix}\right]\ \text{filename}$$

Phase specification	{	CAS0 : Positive phase CAS1 : Reverse phase
Speed specification	{	S : 300 baud (300 bits/second) F : 1200 baud (1200 bits/second)

EXAMPLE 1:
Reading data of a file named TEST, at positive phase, 300 baud

OPEN "CAS0 : (S) TEST" FOR INPUT AS #1

The file descriptor in this case is CAS0 : (S) TEST.

EXAMPLE 2:
Writing data to a file named SAMPLE, at positive phase, 1200 baud

SAVE "CAS0 : (F) SAMPLE"

The file descriptor in this case is CAS0 : (F) SAMPLE.

EXAMPLE 3:
Reading of a program under the filename AD1990

LOAD "AD1990"

This is the same as LOAD "CAS0 : (F) AD1990"

The file descriptor commands which can be used as outlined above are:

OPEN, LOAD, SAVE, LOAD#, SAVE#, and VERIFY

- CAS0 : and CAS1 : specify the read phase (positive/reverse) of data recorded on tape. CAS0 : is generally used, but CAS1 : can be used for certain tape recorders.
- An attribute is automatically assigned when a program or data is written to cassette tape. Attributes can be checked when the tape is read.

ATTRIBUTE	MEANING
B	Binary file (program file)
A	All file (program file saved using SAVE ALL command)
S	Sequential file (data file, memo file, or program file saved in ASCII format)

9-2-2 Communications Circuit

The file descriptor for communications circuit data interchange is as follows:

COM0 : [[Speed], [Parity], [Data], [Stop], [CS], [DS], [CD], [Busy], [Code]]

① Speed (baud rate : data transfer speed)

6 : 4800 baud (4800 bits/sec)
5 : 2400 baud (2400 bits/sec)
4 : 1200 baud (1200 bits/sec)
3 : 600 baud (600 bits/sec)
2 : 300 baud (300 bits/sec)
1 : 150 baud (150 bits/sec)

NOTE: 2400 (5) or less should be specified when the communications circuit is specified using the OPEN statement.

②Parity (parity bit : check bit for data send)

N : No parity bit
E : Even parity
O : Odd parity

③Data (data bit : number of bits representing one character)

7 : JIS 7 bit (7 bits/character)
8 : JIS 8 bit (8 bits/character)

④Stop (stop bit : bit (s) at end of a character signifying character end)

1 : 1 bit
2 : 2 bits

⑤CS (Clear to send (CTS) : control function to inform partner device whether or not data can be sent)

C : Used — The sending device waits until CS is ON.
N : Not used

⑥DS (Data set ready (DSR) : function to inform normal operation of partner device)

D : Used — An NR error is generated when data are received while DS is OFF. When sending, the device waits until DS is ON.
N : Not used

⑦CD (Carrier detect (CD) : function to inform partner device ready to receive data)

C : Used — An NR error is generated when data are received while CD is OFF.
N : Not used

⑧Busy (Busy, (XON/OFF), function to temporarily suspend data send)

B : Used — A send suspend request is sent to the partner device when the remaining number of characters to be read at one time is less than 64 during data receive. During data send, send is temporarily suspended when a send suspend request is received from the partner device. Sending is resumed upon a send start request.
N : Not used

⑨Code (Input/Output code system (SI/SO) : Used to send character expressed as eight bits when data bits specified as seven)

S : Used — This function is only applicable when the data bit parameter (above ③) is specified as seven bits. The SO code ($0E_H$) is sent before codes 80_H or higher in this case to enter the SO mode. Codes $7F_H$ or lower are sent preceded by the SI code ($0F_H$) to enter the SI mode.
The S specification should be used whenever it is necessary to send data with codes 80_H or higher as 7 bit data.
N : Not used

Parameter Default Values

COM0	:	2, E, 8, 1, N, N, N, B, N
Baud rate	:	300 baud (300 bits/sec)
Parity	:	Even
Data bits	:	8 bits
Stop bit	:	1 bit
CS	:	Not used
DS	:	Not used
CD	:	Not used
Busy	:	XON/XOFF
Code	:	SI/SO - Not used

EXAMPLE 1:
Send the character string noted below to RS-232C using the parameters listed.

Baud rate	: 300 baud → 2
Parity	: Even → E
Data bits	: 8 bits → 8
Stop bit	: 1 bit → 1
CS	: Not used → N
DS	: Not used → N
CD	: Not used → N
Busy	: Used → B
Shift in/out	: Not used → N

```
10  OPEN "COM0 : 2, E, 8, 1, N, N, B, N" AS #1
20  PRINT #1, "HELLO."
30  CLOSE
```

EXAMPLE 2:
Receive character string from RS-232C as above.

```
10  OPEN "COM0 : 2, E, 8, 1, N, N, N, B, N" AS #1
20  INPUT #1, A$
30  CLOSE
```

- Communications are performed via the RS-232C terminal.
- Data interchange is performed using a full-duplex (both sides can communicate simultaneously, as with a standard telephone), start-stop system (typical computer communications system, also known as asynchronous).

PART 10

COMMAND REFERENCE

FORMAT ELEMENTS

The method for entering statements is explained below.

- Words in bold type are commands or functions, and they must be entered as shown.
- Braces indicate that one of the parameters enclosed must be specified.
- Commas contained in braces must be written in the position shown.
- Brackets indicate that the parameters enclosed may be omitted. Brackets themselves are not entered.
- An asterisk indicates that the term preceding it may appear more than once.
- Numeric expressions—Constants, expressions, and numeric variables (e.g. 10, 10 + 25, A, unit cost * quantity)
- String expressions—String constants, string variables, and string expressions (e.g. "ABC", A$, and A$ + B$)
- Expressions—General term for numeric and string expressions
- Arguments—Elements used by commands and functions
- Ⓟ.........Can only be executed in a program.
- Ⓜ.........Can only be executed manually.
- Ⓐ.........Can be executed both manually and in a program.
- Ⓕ.........Function instruction that can be executed both manually and in a program.

Example: MID$ function

MID$ (string array , position [, number of characters])

- string array — String expression
- position — Numeric expression
- number of characters — Numeric expression

The term "string expression" under "string array" describes that array. Likewise, "numeric expression" under "position" and "numeric expression" under "number of characters" are descriptors. Also, since the comma and number of characters are enclosed in brackets, they may be omitted.

Example: GOSUB Statement

GOSUB { branch line number (Line number) | # program area number (Single character; 0~9) }

This example illustrates two descriptors for GOSUB: the line number of the subroutine to which the program branches and filename to which the program branches.

PASS

Ⓐ

PURPOSE: Specifies or cancels a password.

FORMAT: PASS "password"
String expression

EXAMPLE: PASS "TEXT"

PARAMETERS:
1. Registering a single password makes it the password for all program areas (P0 ~ P9) and for DATA BANK function.
2. The password must be a string of 1 ~ 8 characters.
3. All characters after the first 8 are ignored when 9 or more characters are entered.

EXPLANATION:
1. The password is used to protect programs and DATA BANK data.
2. The password can be registered in both the CAL mode and BASIC mode.
3. Executing this command registers a password when no password previously exists.
4. Executing the PASS statement using a previously registered password cancels the password. Specifying a password that is different from that registered, results in a PR error.
5. The following operations and commands cannot be executed when a password is registered:
   Program write
   MEMO IN mode specification
   MEMO search
   LIST, LLIST, LIST ALL, LLIST ALL, LIST#, LLIST#, NEW, NEW ALL, NEW#
   EDIT
   SAVE, SAVE# to RS-232C
   SAVE, SAVE# to cassette tape in ASCII format
6. Executing SAVE and SAVE ALL to cassette tape applies the password to the saved program.
7. Loading a program (using LOAD or LOAD ALL) which is protected by a password into the computer causes the password of the loaded program to be registered as the computer password. A PR error is generated when the current password differs from the password of the loaded program.

NEW [ALL]

Ⓜ

PURPOSE: Deletes a program.

FORMAT: NEW [ALL]

EXAMPLE: NEW

EXPLANATION:

1. Deletes the program in the currently specified program area when ALL is omitted. Variables are not cleared.
2. "Ready P*n*" is displayed on the screen after the program is deleted, and the computer stands by for command input.
3. All files that are currently opened are closed.
4. This command cannot be executed for program files that are protected by a password.
5. Attempting to use this command in the CAL mode results in an FC error.
6. Specifying NEW ALL clears the programs in all program areas and all variables.
7. This command cannot be included within a program.

CLEAR

Ⓐ

PURPOSE: Clears all variables and determines the variable area size in accordance with the parameter entered. Also closes any files that are open.

FORMAT: CLEAR [variable area size]

EXAMPLE: CLEAR 400

PARAMETERS: variable area size: Numeric expression
Determines the areas used for variables. The initial setting when ALL RESET is executed depends upon total memory capacity.

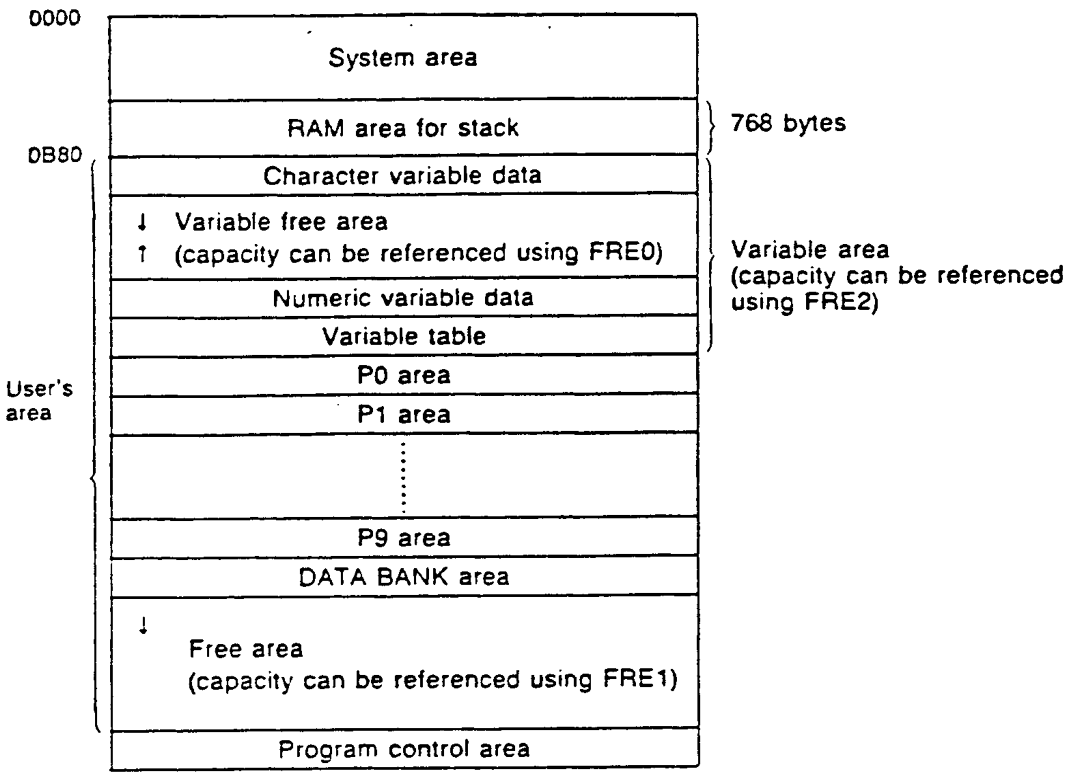
MEMORY CAPACITY	VARIABLE AREA SIZE
Less than 32KB	1536 bytes
32KB and over	8192 bytes

EXPLANATION:

1. Clears all variables.
2. Closes all open files and clears the FOR~NEXT and GOSUB stack.
3. Variable area cannot be set during program execution.

SEE: FRE

MEMORY MAP

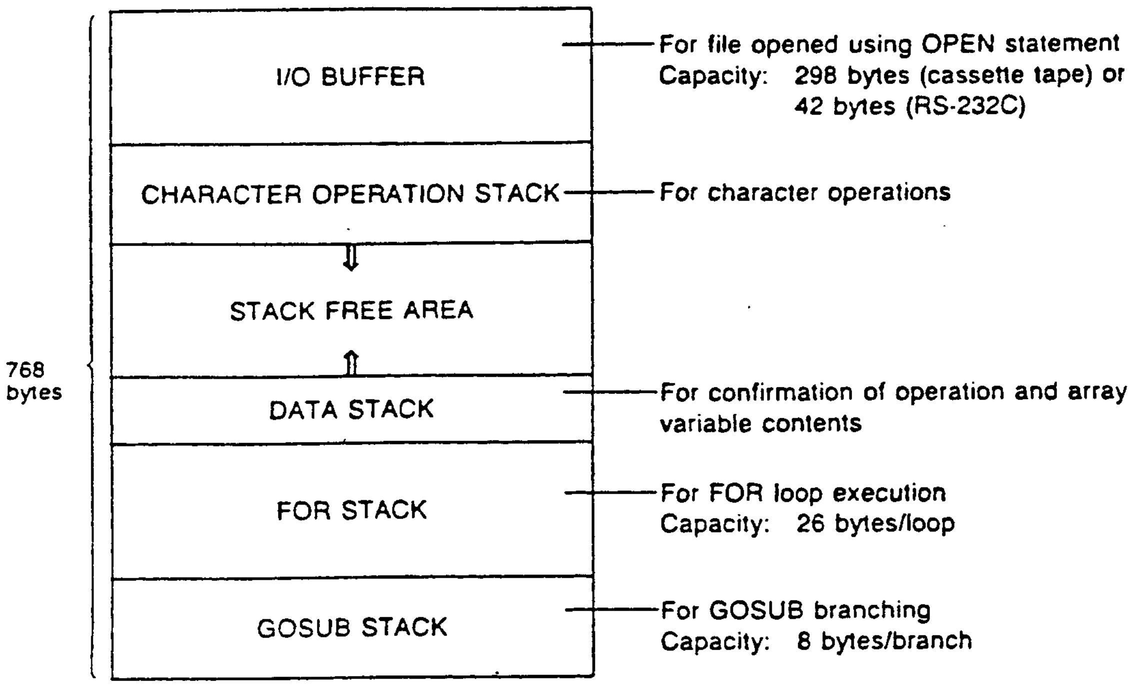


EXPANDED MEMORY CONFIGURATION (UNIT = BYTES)

		FX-850P
Standard	FRE 1	3536
	FRE 2	1536
	User's area	5072
RP-8 RAM expansion pack	FRE 1	11728
	FRE 2	1536
	User's area	13624
RP-33 RAM expansion pack	FRE 1	29648
	FRE 2	8192
	User's area	37840

FRE 1 capacity value is when no programs or DATA BANK data are stored.
FRE 2 capacity can be changed using the CLEAR command.

STACK AREA



FRE

Ⓕ

PURPOSE: Returns memory area size in accordance with argument.

FORMAT: FRE (argument / Numeric expression)

EXAMPLE: PRINT FRE 0

PARAMETERS: argument: Integer in the range of $0 \leq$ argument < 3

EXPLANATION:

1. parameter = 0 : Returns unused memory in variable area in byte units
2. parameter = 1 : Returns unused memory in program or in DATA BANK area in byte units
3. parameter = 2 : Returns overall variable area in byte units

SEE: CLEAR

LIST [ALL]

PURPOSE: Displays all or a part of the currently specified program.

FORMAT:

```
      { [ start line number ]  [ - [ end line number ] ] }
LIST  {     Line number               Line number        }
      { [ . ]                                            }
      { [ALL]                                            }
```

EXAMPLE:
LIST 100
LIST 100 - 300
LIST - 400
LIST

PARAMETERS:
1. start line number: Integer in the range of 1 ≦ line number ≦ 65535 (first line number when omitted)
2. end line number: Integer in the range of 1 ≦ line number ≦ 65535 (end line number when omitted)

EXPLANATION:

1. Displays the currently specified program in the range specified by the line numbers.
2. A minus sign must be used as the delimiter between line numbers.
3. The following five examples illustrate specification of the display range.
   - a) LIST EXE (All lines from beginning of program)
   - b) LIST 30 EXE (Line 30)
   - c) LIST 50 - 100 EXE (Lines 50 through 100)
   - d) LIST 200 - EXE (From line 200 through end of program)
   - e) LIST - 80 EXE (From beginning of program through line 80)
4. Using a period in place of the line number displays the most recently handled (i.e. written, edited, executed). If a program is halted during execution by an error, executing "LIST ." displays the line in which the error was generated.
5. When the specified start line number does not exist, the first line number above that specified is taken as the start line number.
6. When the specified end line number does not exist, the greatest line number not exceeding that specified is taken as the end line.
7. The start line number must be smaller than the end line number.
8. LIST command execution can be halted by pressing the BRK key.
9. Press the STOP key to momentarily halt LIST command execution. To restart execution, press the EXE key or one of the alphanumeric keys.
10. The computer stands by for command input after the program list is displayed.
11. This command cannot be used when a password is registered.
12. This command cannot be used in the CAL mode.
13. Specifying ALL displays all programs in sequence from area P0 through P9.

SEE: EDIT, VARLIST, LLIST

EDIT

Ⓜ

PURPOSE: Enters the BASIC edit mode.

FORMAT: EDIT { [start line number] / Line number or period / [.] }

EXAMPLE: EDIT 100

PARAMETERS: start line number: Integer in the range of 1 ≦ line number ≦ 65535 (first line number when omitted)

EXPLANATION:

1. Enters the BASIC edit mode and displays the program from the specified line number. The cursor is displayed and editing becomes possible when either the ⇦ or ⇨ key is pressed.
2. Using a period in place of the line number displays the most recently handled (i.e. written, edited, executed). If a program is halted during execution by an error, executing "EDIT ." displays the line in which the error was generated.
3. When the specified start line number does not exist, the first line number above that specified is taken as the start line number.
4. This command cannot be used when a password is registered.
5. This command cannot be used in the CAL mode.
6. This mode is canceled by pressing the BRK key.

SEE: LIST, LLIST

VARLIST

Ⓐ

PURPOSE: Displays variable names and array names.

EXAMPLE: VARLIST

EXPLANATION:

1. Displays all currently existing variable names and array names.
2. Press the STOP key to momentarily halt VARLIST command execution. To restart execution, press the EXE key or one of the alphanumeric keys.

SAMPLE EXECUTION: VARLIST EXE

```
A        AB        R        AC$( )
_
```

This command displays all variable names and array names currently stored in memory.

RUN

Ⓜ

PURPOSE: Executes a program.

FORMAT: RUN [execution start line]
Line number

EXAMPLE: RUN
RUN 100

PARAMETERS: start line number: Integer in the range of 1 ≦ line number ≦ 65535

EXPLANATION:

1. Execution starts from the beginning of the program when the line number is omitted.
2. When the specified start line number does not exist, the first line number above that specified is taken as the start line number.
3. This command closes all files that are open.
4. Variable and array values are not cleared.
5. This command cannot be used within a program.
6. This command cannot be used in the CAL mode.

SAMPLE PROGRAM:

```
RUN 100
```

Executes program from line number 100.

TRON

Ⓐ

PURPOSE: Specifies the trace mode.

EXAMPLE: TRON

EXPLANATION:

1. Switches the trace mode ON and TR appears on the display.
2. All subsequent program execution is accompanied by a display of the area name and line number. The first two lines are displayed, and execution is suspended.
   Program execution can be resumed at this time by pressing EXE.
3. The program stays in the TRON mode until the TROFF statement is executed or the power is switched OFF.

SEE: TROFF

SAMPLE EXECUTION: TRON EXE
RUN EXE

```
CAPS     BASIC DEG                    TR STOP
RUN
P0-10
```

P0 represents currently specified program area and 10 currently executed line number.

TROFF

Ⓐ

PURPOSE: Cancels the trace mode.

EXAMPLE: TROFF

EXPLANATION:
Cancels the trace mode (entered using the TRON statement).

SEE: TRON

END

Ⓟ

PURPOSE: Terminates program execution.

EXAMPLE: END

EXPLANATION:

1. Terminates program execution, and the computer stands by for command input.
2. Closes all files that are open.
3. Variables and arrays are not cleared.
4. Any number of END statements can be used in a single program. Program execution is terminated and open files are closed automatically at the end of the program even if an END statement is not included.

SAMPLE PROGRAM:

```
10  FOR I=1 TO 20
20  IF I>10 THEN END
30  PRINT I ;
40  NEXT I
```

Displays values of I in FOR~NEXT loop.
Program ends when I exceeds 10.

STOP

Ⓟ

PURPOSE: Temporarily halts program execution.

EXAMPLE: STOP

EXPLANATION:

1. Temporarily halts program execution and STOP appears on the display. Execution can be resumed by pressing EXE.
2. Pressing SHIFT STOP while execution is halted by the STOP command displays the current program area and line number.
3. Such commands as PRINT can be executed while execution is halted by the STOP command. Manual calculations can also be performed in the CAL mode.
4. Open files, variable values and array values are retained as they are at the point when execution is halted.
5. The STOP status is canceled when an error is generated, the mode is changed, or the program is edited while program execution is halted.

SAMPLE PROGRAM:

```
10  FOR I=1 TO 10
20  IF I=6 THEN STOP : PRINT
30  PRINT I ;
40  NEXT I
```

Displays values of I in FOR ~ NEXT loop.
Execution is halted when I equals 6. Next, pressing EXE resumes execution.

GOTO

Ⓟ

PURPOSE: Branches unconditionally to a specified branch destination.

FORMAT:

GOTO { branch destination line number (Line number) | # program area number (Single character; 0~9) }

SAMPLE:

GOTO 1000
GOTO #7

PARAMETERS:

1. branch destination line number: Integer in the range of 1 ≦ line number ≦ 65535
2. program area number: Single character, 0~9

EXPLANATION:

1. Specifying a line number causes program execution to jump to that line number in the current program area.
2. Specifying a program area number causes program execution to jump to the first line number of the specified program area.
3. A UL error is generated when the specified line number does not exist.

SAMPLE PROGRAM:

```
10 PRINT "PRESS [BRK]" ;
20 PRINT "TO HALT EXECUTION" ;
30 GOTO 10
```

Line 30 returns execution to line 10.
This loop continues until [BRK] is pressed.

GOSUB

Ⓟ

URPOSE: Jumps to a specified subroutine.

ORMAT:

GOSUB { branch destination line number (Line number) / # program area number (Single character; 0−9) }

XAMPLE: GOSUB 100
GOSUB #6

ARAMETERS: 1. branch destination line number: Integer in the range of 1 ≦ line number ≦ 65535
2. program area number: Single character, 0−9

XPLANATION:

. Program execution branches to the subroutine that starts at the specified line number. Execution is returned from the subroutine by the RETURN statement.
. Subroutines can be nested up to 96 levels. Exceeding this value results in an OM error.
. A UL error is generated when the specified line number does not exist.
. CLEAR command cannot be used within a subroutine.

;EE: RETURN

;AMPLE PROGRAM:

```
10  REM***MAIN***
20  GOSUB 40
30  END
40  REM***SUBROUTINE 1***
50  PRINT "SUBROUTINE 1" ;
60  GOSUB 80
70  RETURN
80  REM***SUBROUTINE 2***
90  PRINT "SUBROUTINE 2"
100  RETURN
```

Line 20 branches to subroutine beginning at line 40, and line 60 branches to subroutine beginning at line 80.

RETURN

Ⓟ

PURPOSE: Returns execution from a subroutine to the main program.

FORMAT: RETURN

EXAMPLE: RETURN

EXPLANATION:

1. Returns program execution to the statement immediately following the statement which originally called a subroutine.
2. A GS error is generated when the RETURN statement is executed without first executing a GOSUB statement.

SEE: GOSUB, ON～GOSUB

SAMPLE PROGRAM:

```
10  REM SUBROUTINE
20  GOSUB 100
30  END
100  PRINT "SUBROUTINE 1"
110  GOSUB 200
120  RETURN
200  PRINT "SUBROUTINE 2"
210  RETURN
```

RETURN in line 120 returns to line 20, while line 210 returns to line 110.

ON GOTO ℗

ɔRPOSE: Jumps to a specified branch destination in accordance with a specified branching condition.

ɔRMAT: ON condition GOTO [branch destination] [, [branch destination]]*

Numeric expression (condition)

Branch destination: { destination branch line number (Line number) / # program area number (Single character; 0~9) }

‹AMPLE: ON A GOTO 100, 200, 300

ʌRAMETERS:
1. branch condition: Numeric expression truncated to an integer
2. line number: Integer in the range of 1 ≦ line number ≦ 65535
3. program area number: Single character, 0~9

‹PLANATION:
The GOTO statement is executed in accordance with the value of the expression used for the branch condition. For example, execution jumps to the first branch destination specified when the value is 1, to the second destination when the value is 2, etc.
Program execution does not branch and execution proceeds to the next statement when the value of the branch condition is less than 1, or if a branch destination corresponding to that value does not exist.
Up to 99 branch destinations may be specified.

ʌMPLE PROGRAM

```
10 INPUT "1 OR 2" ; A
20 ON A GOTO 40, 50
30 END
40 PRINT "ONE" : END
50 PRINT "TWO"
```

Execution jumps to line 40 if 1 EXE is entered or to line 50 if 2 EXE is entered. Otherwise, execution terminates at line 30.

ON GOSUB

Ⓟ

PURPOSE: Jumps to a specified subroutine in accordance with a specified branching condition.

FORMAT: ON condition (Numeric expression) GOSUB [branch destination] [, [branch destination]]*

Branch destination: { destination branch line number (Line number) / # program area number (Single character; 0~9) }

EXAMPLE: ON A GOSUB 1000, 1100, 1200

PARAMETERS:
1. branch condition: Numeric expression truncated to an integer
2. line number: Integer in the range of $1 \leq$ line number ≤ 65535
3. program area number: Single character, 0~9

EXPLANATION:
1. The GOSUB statement is executed in accordance with the value of the expression used for the branch condition. For example, execution jumps to the first branch destination specified when the value is 1, to the second destination when the value is 2, etc.
2. Program execution does not branch and execution proceeds to the next statement when the value of the branch condition is less than 1, or if a branch destination corresponding to that value does not exist.
3. Up to 99 branch destinations may be specified.

SEE: RETURN

SAMPLE PROGRAM:

```
10  S1=0 : S2=0
20  FOR I=1 TO 100
30  ON (I MOD 2)+1 GOSUB 1000, 1100
40  NEXT I
50  PRINT "S1=" ; S1
60  PRINT "S2=" ; S2
70  END
1000  S1=S1+I : RETURN
1100  S2=S2+I : RETURN
```

S1 calculates sum of even numbers from 1 to 100, S2 calculates sum of odd numbers from 1 to 100.

F ~ THEN ~ ELSE/IF ~ GOTO ~ ELSE ℗

RPOSE: Executes the THEN statement or GOTO statement when the specified condition is met. The ELSE statement is executed when the specified condition is not met.

RMAT:

```
     condition      { THEN   statement             }  [      { statement          } ]
IF   ----------     {        [ : statement]        }  [ ELSE { [ : statement]     } ]
     Numeric        { GOTO   branch destination    }  [      { branch destination } ]
     expression
```

```
                        { destination branch line number }
                        {          Line number           }
Branch destination:     { # program area number          }
                        {     Single character; 0 ~ 9    }
```

AMPLE:

```
IF A=0 THEN 300 ELSE 400
IF K$="Y" THEN PRINT X ELSE PRINT Y
```

RAMETERS:
1. branch condition: Numeric expression truncated to an integer
2. line number: Integer in the range of 1 ≦ line number ≦ 65535
3. program area number: Single character, 0 ~ 9

PLANATION:

he statement following the THEN clause is executed, or execution jumps to the destination specified by the GOTO statement when the branch condition is met.

 the branch condition is not met, the statement following the ELSE statement is executed, or the program jumps to the specified branch destination. Execution proceeds to the next program line when the ELSE statement is omitted.

The format "IF A THEN ~" results in the condition being met when value of the expression (A) is not 0 (absolute value of A < 1×10^{-99}). The condition is not met when the value of the expression is 0.

F statements can be nested (an IF statement may contain other IF statements). In this case, the THEN ~ ELSE statements are related by their proximity. The GOTO ~ ELSE combinations have the same relationships.

```
F ~ THEN IF THEN ~ ELSE IF ~ THEN ~ ELSE ~ ELSE ~
```

MPLE PROGRAM:

```
10  INPUT "1 TO 9" ; A
20  IF (0<A) AND (A<10) THEN PRINT "GOOD!" ELSE 10
```

"GOOD" is displayed for input values from 1 to 9. Re-input is requested for other values.

FOR ~ NEXT

Ⓟ

PURPOSE: Executes the program lines between the FOR statement and NEXT statement and increments the control variable, starting with the initial value. Execution is terminated when value of the control variable exceeds the specified final value.

FORMAT: FOR control variable name = initial value (Numeric expression)

TO final value (Numeric expression) [STEP increment] (Numeric expression)

NEXT [Control variable name] [, Control variable name]*

EXAMPLE:

```
FOR I=1 TO 10 STEP 0.1
 ⁞
NEXT I
```

PARAMETERS:

1. control variable name: Array variables cannot be used.
2. initial value: Numeric expression
3. final value: Numeric expression
4. increment: Numeric expression (default value = 1)

EXPLANATION:

1. None of the statements between FOR and NEXT are executed and the program proceeds to the next executable statement after NEXT when the initial value is greater than the final value.
2. Each FOR requires a corresponding NEXT.
3. FOR ~ NEXT loops can be nested (a FOR ~ NEXT loop can be placed inside another FOR ~ NEXT loop). Nested loops must be structured as shown below with NEXT appearing in reverse sequence of the FOR (e.g. FOR A, FOR B, FOR C ~ NEXT C, NEXT B, NEXT A).

```
┌─10 FOR I=1 TO 12 STEP 3
│┌20 FOR J=1 TO 4 STEP 0.5
││30 PRINT I, J
│└40 NEXT J
└─50 NEXT I
  60 END
```

4. FOR ~ NEXT loops can be nested up to 29 levels.
5. The control variable may be omitted from NEXT. However, use of the control variable in the NEXT statement is recommended when using nested loops.

NEXT statements can be chained by including them under one NEXT statement, separated by commas.

```
10  FOR I = 1 TO 12 STEP 3
20  FOR J = 1 TO 4 STEP 0.5
30  PRINT I, J
40  NEXT J
50  NEXT I
60  END
```

```
┌─10  FOR I = 1 TO 12 STEP 3
│┌20  FOR J = 1 TO 4 STEP 0.5
││30  PRINT  I, J
└┴40  NEXT J, I
  50  END
```

The control variable retains the value which exceeds the final value (and terminates the loop) when loop execution is complete. With the loop FOR I = 3 TO 10 STEP 3, for example, the value of control variable I is 12 when execution of the loop is complete.
Jumping out of a FOR – NEXT loop is also possible. In this case, the current control variable value is retained in memory, and the loop can be resumed by returning with a GOTO statement.

REM(')

Ⓟ

PURPOSE: Allows remarks or comments to be included within a program. This command is not executed.

FORMAT: {REM / ' } comments (String expression)

EXAMPLE: REM or '

PARAMETERS: comments: String expression (internal codes 20 to 7E and 80 to FB)

EXPLANATION:

1. Including an apostrophe or REM statement following the line number indicates that the following text is comments and should be ignored in program execution.
2. The apostrophe may be included at the end of any executable statement to indicate that the following text is comments. The REM statement can only be used at the beginning of a line.
3. Any command following the REM statement is treated as a comment and is not executed.

   PRINT A: REM 123 — 123 is treated as a comment. (123 → Comments)

   PRINT A REM 123 — SN error occurs.

   PRINT A ' 123 — 123 is treated as a comment. (123 → Comments)

4. An apostrophe is entered by pressing the ' key following the [F] key.

SAMPLE PROGRAM:

```
10  ' REM( ' ) indicates comment
```

LET

Ⓐ

PURPOSE: Assigns the value of an expression on the right side of an equation to the variable on the left side.

FORMAT: [LET] numeric variable name = Numeric expression
[LET] string variable name = String expression

EXAMPLE: LET A = 15
LET K$ = "123"

EXPLANATION:
1. Assigns the value of an expression on the right side of an equation to the variable on the left side.
2. Numeric expressions can only be assigned to numeric variables, and string expressions can only be assigned to string variables. A TM error is generated when an attempt is made to assign a string expression to a numeric variable, and vice versa.
3. LET may be omitted.

SAMPLE PROGRAM:

```
10  LET A=10
20  B=20
30  PRINT A ; B
```

Assigns 10 to variable A and 20 to variable B, and displays both.

DATA

Ⓟ

PURPOSE: Holds data for reading by the READ statement.

FORMAT: DATA [data] [, [data]]*
Constant Constant

EXAMPLE: DATA 10, 5, 8, 3
DATA CAT, DOG, LION

PARAMETERS:
1. data: String constants or numeric constants
2. string constants: Quotation marks are not required unless the string contains a comma which is part of the data. A null data string (length 0) is assumed when data is omitted from this statement.

EXPLANATION:
1. This statement can be used anywhere in the program to hold data to be read by the READ command.
2. Multiple data items are separated by commas.

SEE: READ, RESTORE

SAMPLE PROGRAM:

```
10  READ A$
20  RESTORE 60
30  READ B$
40  PRINT A$+ "  " + B$
50  DATA AD 1990, ABC
60  DATA DEFG
```

Character data "AD1990" and "DEFG" read from lines 50 and 60, and displayed.

READ Ⓟ

RPOSE: Reads the contents of the DATA statement into memory.

RMAT: READ Variable name [,Variable name]*

AMPLE: READ A, B
READ C$, X, Y

.RAMETERS: Variable name

.PLANATION:

Assigns the data contained in a DATA statement to the variables on a one-by-one basis. Numeric data can only be assigned to numeric variables, and string data can only be assigned to string variables. A TM error is generated when an attempt is made to assign string data to a numeric variable, and vice versa.
The data in DATA statements is read from the lowest line number in ascending order. Data are read in order from the beginning of a DATA statement.
The first execution of the READ statement reads the first data item contained in the first DATA statement. Subsequent executions read data items in sequential order.
The data line to be read can be specified using the RESTORE statement.

EE: DATA, RESTORE

AMPLE PROGRAM:

```
10  READ X
20  IF X< >0 THEN PRINT X ; : GOTO 10
30  END
100  DATA 1, 2, 3, 4, 5, 6, 7, 8, 9
110  DATA 9, 8, 7, 6, 5, 4, 3, 2, 1
120  DATA 0
```

Sequentially reads data beginning at line 100 and stops execution when 0 is encountered as data.

RESTORE

Ⓟ

PURPOSE: Specifies a DATA line for reading by the READ statement.

FORMAT: RESTORE [line number]
Numeric expression

EXAMPLES:
RESTORE
RESTORE 1000
RESTORE (10*10)
└─line 100

PARAMETERS: line number: Integer in the range of 1 ≦ line number ≦ 65535

EXPLANATION:

1. The first DATA line in the program file containing the READ statement is the default option when the line number is omitted.
2. When a line number is specified, the first data item in the specified DATA line is read by the next READ statement execution. A UL error is generated when the specified line number does not exist, while a DA error is generated when no data exist in the specified line.
3. A numeric expression can be used for line number specification. In this case, the numeric expression must be enclosed in parentheses.

SEE: READ, DATA

SAMPLE PROGRAM:

```
10  READ X
20  IF X< >0 THEN PRINT X ; : GOTO 10
30  RESTORE 110
40  READ X
50  IF X< >0 THEN PRINT X ; : GOTO 40
60  END
100  DATA 1, 2, 3, 4, 5, 6, 7, 8, 9
110  DATA 9, 8, 7, 6, 5, 4, 3, 2, 1
120  DATA 0
```

Lines 10 ~ 20 read data from lines 100 ~ 120, while lines 30 ~ 50 read data from lines 110 ~ 120 then display them.

PRINT

Ⓐ

URPOSE: Displays data on the screen.

ORMAT: PRINT [output data] {; ,} [output data]*

Output data: TAB (Tab specification), numeric expression, string array

XAMPLE: PRINT "AD1990"

ARAMETERS: output data: Output control function, numeric expression, or string expression

XPLANATION:

Output of a numeric or string expression displays the value or string on the screen. Control function output results in the operation determined by the function being performed.
Numeric expressions are displayed in decimal notation with values longer than 10 digits.
a) Integers: Values less than 1×10^{10}
b) Fraction: Decimal fractions smaller than 10 digits
c) Exponent: Other values
A space is added after displayed numeric expressions, with negative expressions preceded by a minus sign, and positive expressions preceded by a space. Expressions are displayed as integers, fractions, or exponential expressions, with the display format automatically selected according to the value of the expression.
String expressions are displayed unchanged. There are, however, special operations for internal codes 00_H ~ $1F_H$, $7F_H$ (see CHARACTER CODE TABLE on page 395).
Output is displayed on the screen from the current position of the cursor to the right. A line feed results when the cursor reaches the last column on the last line of the screen (lower right), scrolling the entire screen upwards. Subsequent output is displayed from the beginning of the bottom line of the screen (lower left).
Separating output data with commas causes execution to be halted with each display (STOP appears on display). Pressing EXE executes a carrier return/line feed and proceeds to the next display.
Separating output data with semicolons causes each output to be displayed immediately following the previous output.
Including a semicolon at the end of this statement causes the cursor to remain at position immediately following the displayed output.
Ending this command with output data or a comma, causes execution to be halted following display of the output data (STOP appears on display). Pressing EXE executes a line change and proceeds to the next display.

9. Omitting the output data (PRINT command only) executes a line change without halting execution.
10. Execution is not halted when this statement is executed while in the print mode ([MODE] [7]).
11. Execution is not halted when this statement is executed while in the manual mode.

SEE: TAB

SAMPLE PROGRAM:

```
10  PRINT "PRINT DISPLAYS MESSAGES"
20  PRINT "ON THE SCREEN"
```

PRINT statement displays message on screen.

ˉAB

Ⓔ

:POSE: Outputs a horizontal tab specification to the screen or printer.

:MAT: TAB (tab specification)
Numeric constant or numeric variable

.MPLE: PRINT TAB (5) ; "ABC"

:AMETERS: tab specification: Numeric expression truncated to an integer in the range of 0 ≦ tab specification < 256.

'LANATION:

sed in the PRINT, LPRINT, and PRINT# statements to specify a display position on line. Spaces are inserted from the left end of the line to the specified position.
he display position is determined by counting from the left end of the line (position 0 nd) to the right, up to the specified value.
. tab specification value which is less than the current printhead position causes the tabu- ation to be performed following a carrier return/line feed.

Ξ: PRINT, LPRINT, PRINT#

ΛPLE PROGRAM:

```
10  FOR I=0 TO 25
20  PRINT TAB ( I ) ; "ABCDEFG" ;
30  NEXT I
```

Prints successive lines of "ABCDEFG", with each line proceeding to the right.

LOCATE Ⓐ

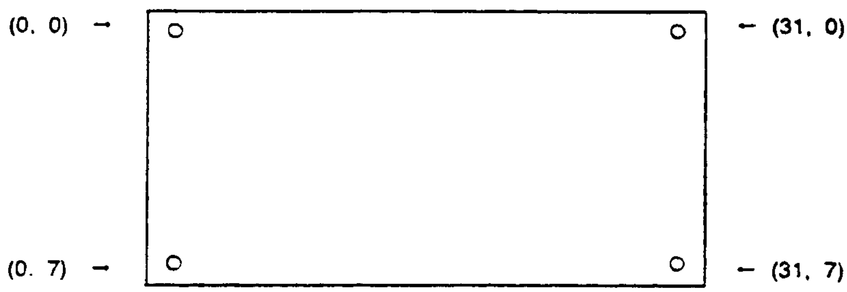
PURPOSE: Moves the cursor to a specified position on the virtual screen.

FORMAT: LOCATE X-coordinate (Numeric expression) , Y-coordinate (Numeric expression)

EXAMPLE: LOCATE 10, 0

PARAMETERS:
1. X-coordinate: Numeric expression truncated to an integer in the range of 0 ≦ X-coordinate < 32
2. Y-coordinate: Numeric expression truncated to an integer in the range of 0 ≦ Y-coordinate < 8

EXPLANATION:
1. Locates the cursor at a specified position on the virtual screen.
2. The origin of the coordinates is the upper left corner of the screen (0, 0). The X coordinate value is incremented for each character position to the right. The Y value coordinate is incremented form each line down.



SAMPLE PROGRAM:

```
10 CLS
20 LOCATE 0, 0
30 PRINT "SCREEN UPPER LEFT" ;
40 GOTO 20
```

Displays message from upper left of display.

:LS

Ⓐ

ʔPOSE: Clears the display screen.

ᐠMPLE: CLS

ʾLANATION:

screen is cleared and the cursor is located at the home position. Pressing the [key] [CLS key] or executing PRINT CHR$(12) ; produces the same result.

ʌPLE PROGRAM:

```
10  REM CLEAR SCREEN
20  CLS
```

Clears screen.

SET

Ⓐ

PURPOSE: Specifies output format of numeric data.

FORMAT:

SET { F number of digits (Single character; 0~9) / E number of digits (Single character; 0~9) / N }

EXAMPLE: SET F3

PARAMETERS:

F number of digits (Single character; 0~9)	Specifies number of decimal places.
E number of digits (Single character; 0~9)	Specifies number of significant digits.
N	Cancels current specification.

EXPLANATION:

1. This command specifies the number of decimal places and the number of significant digits for numeric data output to the display, printer, tape recorder, or RS-232C terminal.
2. The number of decimal places can be specified within the range of 0 through 9.
3. The number of significant digits can be specified within the range of 1 through 10. The statement SET E0 specifies the number of significant digits as 10.
4. SET N cancels both specifications.
5. Output values are rounded to the specified decimal places or to the specified significant digits.
6. This command is only valid for output data. The mantissa part for internal calculations is still 12 digits.

SAMPLE PROGRAM:

```
10  A=10/3
20  SET F2
30  PRINT A
40  SET E2
50  PRINT A
60  END
```

Key	Display
[R][U][N][EXE]	RUN 3.33
[EXE]	3.33 3.3E+00
[ANS]	3.3E+00 3.33333333_
[EXE]	3.3333333 _

Result of 10/3 displayed with 2 decimal places, 2 significant digits, and specificatioin canceled.

BEEP

Ⓐ

PURPOSE: Sounds the buzzer.

FORMAT: BEEP [{0 / 1} / Numeric expression]

EXAMPLE: BEEP 1

EXPLANATION:

1. A low tone is specified by BEEP or BEEP 0.
2. A high tone is specified by BEEP 1.
3. Numeric expressions can be in place of 0 and 1.

SAMPLE PROGRAM:

```
10  BEEP 1 : BEEP 0 : BEEP 1 : BEEP 0
```

INPUT

Ⓟ

PURPOSE: Assigns keyboard data input to a variable.

FORMAT: INPUT ["message" {;/,}] variable name [, ["message" {;/,}], variable name]

EXAMPLE: INPUT "YEAR=", Y, "MONTH=", M, "DAY=", D

PARAMETERS: 1. message: Character string beginning with a string constant
2. variable name: Numeric variable name or string variable name

EXPLANATION:

1. Data can be input to the specified variable from the keyboard.
2. Messages included in the INPUT statement are displayed. A question mark is displayed following the message when a semicolon is included following the message specification.
3. A question mark only is displayed when a message is not specified.
4. The EXE key must be pressed following each data input.
5. Numeric expressions can only be assigned to numeric variables, and string expressions can only be assigned to string variables. A TM error is generated when an attempt is made to assign a string expression to a numeric variable.
6. Quotation marks are not used when entering string data. Enclosing a string in quotation marks causes the quotation marks to be stored as part of the string.
7. Pressing the EXE key without entering data inputs a string of length 0 for a string variable, while a numeric variable retains its current value.
8. Generally, the logical line immediately following the message is input. The cursor can, however, be moved to any position on the virtual screen (using the cursor keys), and all data from the current cursor position to the end of the current logical line are input when EXE is pressed.
9. Numeric expressions may be used for numeric value input.
10. Pressing the IN key or changing modes during execution of the INPUT statement terminates program execution.
11. Pressing the IN key during execution of INPUT leaves program execution.
12. Input data can be edited using the INS key, cursor keys, etc.
Character data can be input within the range of character codes 20 through 7E and 80 through FF.

SAMPLE PROGRAM:

```
10 INPUT "INPUT STRING" ; S$
20 PRINT "S$=" ; S$
30 END
```

Displays string entry.

INKEY$

Ⓕ

JRPOSE: Assigns a single character input from the keyboard to a variable.

(AMPLE: A$ = INKEY$

(PLANATION:

Returns the character or performs the function corresponding to the key pressed during execution of this statement. A null string is returned if a key is not pressed.

The following operations are performed when the keys listed below are pressed during execution of INKEY$.

BRK : Terminates program execution.

STOP : Suspends program execution.

One-key commands and one-key functions : Return a null string.

The cursor is not displayed during data input stand by, and input characters are not displayed. Control codes (00H – 1FH) can be input, but the corresponding operations will not be performed.

EE: INPUT$

AMPLE PROGRAM:

```
10  PRINT "PRESS ANY KEY" ;
20  C$=INKEY$
30  IF C$=" " THEN 20
40  CLS: PRINT "YOU PRESS " ; C$ ; "KEY"
50  END
```

Displays character corresponding to key input.

INPUT$

Ⓕ

PURPOSE: Assigns a specified number of characters from the keyboard to a variable.

FORMAT: INPUT$ (number of characters)
Numeric expression

EXAMPLE: A$ = INPUT$ (3)

PARAMETERS: number of characters: Numeric expression truncated to an integer in the range of 0 ≦ number of characters < 256

EXPLANATION:

1. A string of the length specified by the number of characters is read from the keyboard buffer. Execution waits for the keyboard input when the buffer is empty.
2. The following operations are performed when the keys listed below are pressed during execution of INPUT$.
   BRK : Halts program execution.
   One-key commands and one-key functions : Return a null string.
3. The cursor is not displayed during data input stand by, and input characters are not displayed. Control codes (&H00 ~ &H1F) can be input, but the corresponding operations will not be performed.

SEE: INKEY$

SAMPLE PROGRAM:

```
10 PRINT "ENTER SECRET CODE" ;
20 ID$=INPUT$ (4)
30 IF ID$<> "9876" THEN 10
40 PRINT:PRINT "OK"
```

Checks for validity of input secret code 9876.

DIM

Ⓐ

PURPOSE: Declares an array.

FORMAT:

DIM array name (subscript maximum value [, subscript maximum value]*)
(subscript maximum value: Numeric expression)

[, array name (subscript maximum value [, subscript maximum value]*)]*
(subscript maximum value: Numeric expression)

XAMPLE: DIM A$ (10), B$ (10), X (2, 2, 2)

ARAMETERS: 1. array name: Variable name
2. subscript maximum value: Numeric expression truncated to an integer

XPLANATION:

Declares an array of the dimensions determined by the number of subscript maximum values. The size of the array is determined by each subscript maximum value.
Array elements range from 0 through the specified subscript maximum value.
All elements of a newly declared array are set to their initial value. For numeric arrays, the initial value is 0, while string arrays assigned null strings (length 0).
The size of an array is limited by available memory capacity. Declaration by the DIM statement is subjected to the limitations specified for logical lines (255 characters).
Declaring identical (same array name, same subscript maximum value) in the same program causes second declaration to be disregarded. Declaring two arrays with identical names and different subscript maximum values results in a DD error.
An array variable cannot be used unless they are first declared in a DIM statement.

E: ERASE, CLEAR

MPLE PROGRAM:

```
10 DIM AS (5)
20 FOR I=65 TO 70
30 AS (I-65)=CHRS ( I )
40 PRINT AS (I-65) ;
50 NEXT I
```

Respectively assigns A through F to array cells A$(0) through A$(5).

ERASE

Ⓐ

PURPOSE: Erases a specified array.

FORMAT: ERASE [array name [, array name]*]

EXAMPLE: ERASE A$, X

PARAMETERS: array name: Variable name

EXPLANATION:

1. Erases the specified array from memory.
2. An error does not result when the specified array does not exist, and the program proceeds to the next executable statement.
3. The ERASE statement cannot be used in a FOR–NEXT loop.
4. To declare an array using a name already assigned to an existing array, first erase the existing array with the ERASE statement.

SEE: DIM

SAMPLE PROGRAM:

```
10 CLEAR
20 DIM A$ (10), B$ (10)
30 ERASE A$
40 VARLIST
```

Declares arrays A$ and B$, and then erases array A$.

PEEK

Ⓕ

JRPOSE: Returns the value stored at the specified memory address.

)RMAT: PEEK (address)
Numeric expression

AMPLE: PEEK (&H100)

RAMETERS: address: Numeric expression truncated to an integer in the range of −32769 < address < 65536. Negative addresses are added to 65536 and the contents of the resulting address are returned (i.e. PEEK (−1) is identical to PEEK (65535)).

PLANATION:
Returns the value stored in memory at the specified address.
The actual address is specified using the DEFSEG statement.
DEFSEG = 2
A = PEEK (&H100)
The above does not directly read the contents of address &H100 (256). Instead, the contents of &H120 (288) are read (16 × 2 + 256 = 288).
Further information of segments can be found under DEFSEG.

E: POKE, DEFSEG

MPLE PROGRAM:

```
 5  DEFSEG=&H0
10  FOR I=&H0C00 TO &H0D00
20  PRINT HEX$ (PEEK ( I ) ) ; " ␣ " ;
30  NEXT I
```

Prints memory contents from &H0C00 to &H0D00 in hexadecimal.

POKE

Ⓟ

PURPOSE: Writes data to a specified address.

FORMAT: POKE address , data

address: Numeric expression
data: Numeric expression

EXAMPLE: POKE &H7000, 0

PARAMETERS:
1. address: Numeric expression truncated to an integer in the range of $-32769 < \text{address} < 65536$. Negative addresses are added to 65536 and data are written to the resulting address (i.e. POKE −1, is identical to POKE 65535, data).
2. data: Numeric expression truncated to an integer in the range of $0 \leq \text{data} < 256$

EXPLANATION:
1. Writes data to the specified address in memory.
2. Runaway execution may result if the contents of an address outside the user work area is altered using the POKE statement.
3. The actual address is specified using the DEFSEG statement.

```
DEFSEG = 2
POKE &H100, 0
```

The above does not directly write data to address &H100 (256). Instead, 0 is written to address &H120 (288).

$(16 \times 2 + 256 = 288)$

Further information of segments can be found under DEFSEG.

SEE: PEEK, DEFSEG

SAMPLE PROGRAM:

```
10 DEFSEG=&H0
20 FOR I=&H7000 TO &H7010
30 POKE I, 0
40 NEXT I
50 END
```

Clears (assigns zeros) memory from 7000_H to 7010_H.

DEFSEG

Ⓐ

PURPOSE: Specifies segment base address.

FORMAT: DEFSEG <u>segment address</u>
Numeric expression

EXAMPLE: DEFSEG =16

PARAMETERS: segment address: Integer within the range of $-32768 \leq$ segment address < 65536

EXPLANATION:

. Specifies the segment base address for use with the PEEK and POKE commands. The relationship between the address (offset address) and the segment address within the PEEK and POKE format is as follows:
actual address = segment address × 16 + offset address

. The initial specification for DEFSEG is 0 whenever power is switched ON, or the P button or ALL RESET button is pressed.

SEE: PEEK, POKE

AMPLE PROGRAM:

```
10  DEFSEG=&H1000
20  A=PEEK (&H00F0)
```

In this case, the value assigned to A is that contained in address 100F0H.

ON ERROR GOTO ⓟ

PURPOSE: Specifies the line number to which execution branches when an error is generated.

FORMAT: ON ERROR GOTO <u>branch destination line number</u>
Line number

EXAMPLE: ON ERROR GOTO 1000

PARAMETERS: branch destination line number:
Integer in the range of 0≦line number≦65535

EXPLANATION:

1. Specifies the line number to which program execution branches when an error is generated. The program returns to normal operations when a RESUME statement is executed after the error handling routine (starting at the specified line number) is executed.
2. An error is generated and program execution is halted when the branch destination line number is 0.
3. An error generated after execution branches to the specified line number causes an error message to be displayed and program execution to be halted.
4. An ON ERROR GOTO statement must be followed by a corresponding RESUME statement in the same program area. Branching to another program area using ON ERROR GOTO generates an error when the RESUME statement in the other program area is executed.

* The operations outlined are limited to BASIC program execution.

SEE: ERR, ERL, RESUME

SAMPLE PROGRAM:

```
10  ON ERROR GOTO 40
20  **ERROR**
30  END
40  PRINT " OOPS! ERROR!!! " : BEEP 1
50  RESUME 30
```

Execution of line 40, followed by line 30 if error generated. The program shown here is only an error subroutine and does nothing by itself.

RESUME

Ⓟ

JRPOSE: Returns from an error handling routine to the main routine.

)RMAT:

```
         [      NEXT            ]
RESUME   [ {return line number} ]
         [ {  Line number     } ]
```

AMPLE: RESUME NEXT
RESUME 100

.RAMETERS: 1. NEXT
2. return line number: Integer in the range of 1 ≦ line number ≦ 65535

PLANATION:

This statement is entered at the end of an error handling routine.
The statement that generated the original error is the default option when the return destination (NEXT or return line number) is omitted.
Program execution returns to the statement following the statement that generated the original error when NEXT is specified.
Return line number specifies the line to which program execution is to be resumed.
A RESUME statement without a return destination or a RESUME statement that specifies the line in which the original error was generated as the return line number cannot be written at the beginning of the error handling routine. This would result in an endless loop between the statement in which the error was generated and the error handling routine.
A RESUME statement must always be included in the same program area as the ON ERROR GOTO statement.

E: ERR, ERL, ON ERROR GOTO

MPLE PROGRAM:

```
10  ON ERROR GOTO 1000
20  INPUT A
30  D=1/A
40  PRINT "1/" ; A ; "=" ; D
50  GOTO 20
1000  PRINT "0 IS ILLEGAL"
1010  RESUME 20
```

Calculates reciprocals of input values and returns to line 20 if a 0 is entered (resulting in division by 0).

ERL

Ⓕ

PURPOSE: Returns the number of a line in which an error has been generated.

FORMAT: ER = ERL

EXPLANATION:

The value of ERL can only be changed within a program, and the value is cleared when a program is executed or when the power of the unit is switched OFF.

SEE: ERR, ON ERROR GOTO

SAMPLE PROGRAM:

```
10  ON ERROR GOTO 40
20  **ERROR**
30  END
40  PRINT "ERROR LINE=" ; ERL
50  RESUME 30
```

Error is generated in line 20 and corresponding error code is displayed in line 40.

ERR

Ⓕ

PURPOSE: Returns the error code which corresponds to a generated error.

FORMAT: PRINT ERR

EXPLANATION:

The value of ERR can only be changed within a program, and the value is cleared when a program is executed or when the power of the unit is switched ON. See the error message table on page 397 for details concerning error codes and their corresponding error messages.

SEE: ON ERROR GOTO, ERL, Error Message Table

SAMPLE PROGRAM:

```
10  ON ERROR GOTO 40
20  **ERROR**
30  END
40  PRINT "ERROR CODE=" ; ERR
50  RESUME 30
```

An error is generated in line 20 and the corresponding error code is displayed in line 40.

ANGLE

Ⓐ

JRPOSE: Specifies the angle unit.

ⅮRMAT: ANGLE angle specification
Numeric expression

XAMPLE: ANGLE 0

ARAMETERS: angle specification: Numeric expression truncated to an integer in the range of $0 \leq$ angle specification < 3

XPLANATION:

. The angle units for the trigonometric function can be specified using the values 0, 1, and 2.
 0: DEG (degrees)
 1: RAD (radians)
 2: GRAD (grads)

. The relationships between the angle units are as follows:

Angle Unit	DEG	RAD	GRAD
1DEG =	1	$\frac{\pi}{180}$	$\frac{100}{90}$
1RAD =	$\frac{180}{\pi}$	1	$\frac{200}{\pi}$
1GRAD =	$\frac{90}{100}$	$\frac{\pi}{200}$	1

$90° = \frac{\pi}{2}$ rad = 100 grad

3. ANGLE 0 is set whenever NEW ALL is executed.
4. The angle unit can also be specified using the [illegible] key.

SAMPLE PROGRAMS:

```
10 ANGLE 0 'DEGREE
20 PRINT SIN 30 ;
30 ANGLE 1 'RADIAN
40 PRINT SIN (PI/6) ;
50 ANGLE 2 'GRAD
60 PRINT SIN (100/3)
```

Calculates and displays sin 30° in the degree mode, sin $\frac{\pi}{6}$ in the radian mode, and sin $\frac{100}{3}$ in the grad mode.

SIN
COS
TAN

Ⓔ

PURPOSE: Returns the value of the corresponding trigonometric function value for the argument.

FORMAT:

SIN (argument) — Numeric expression

COS (argument) — Numeric expression

TAN (argument) — Numeric expression

* The parentheses enclosing the argument can be omitted when the argument is a numeric value or variable.

EXAMPLE: SIN (30), COS (PI/2)

PARAMETERS: argument: Numeric expression (angle)

|argument| < 1440 (DEG)
|argument| < 8π (RAD)
|argument| < 1600 (GRAD)

EXPLANATION:

Returns the value of the corresponding trigonometric function for the argument.

SIN SINE
COS COSINE
TAN TANGENT

SEE: ANGLE, ASN, ACS, ATN

SAMPLE PROGRAM:

```
10  ANGLE 0
20  INPUT "DEGREE=" , D
30  PRINT "SIN ( " ; D ; " )=" ; SIN D
40  PRINT "COS ( " ; D ; " )=" ; COS D
50  PRINT "TAN ( " ; D ; " )=" ; TAN D
60  GOTO 20
```

Displays trigonometric function values for input angles.

ASN
ACS
ATN

Ⓕ

PURPOSE: Returns the value of the corresponding inverse trigonometric function for the argument.

FORMAT:

ASN (argument) — Numeric expression

ACS (argument) — Numeric expression

ATN (argument) — Numeric expression

- The parentheses enclosing the argument can be omitted when the argument is a numeric value or variable.

SAMPLE: ASN (0.1)

PARAMETERS: argument: Numeric expression in the range of $-1 \leqq$ argument $\leqq 1$ (ASN, ACS)

EXPLANATION:

Returns the value of the corresponding inverse trigonometric function for the argument.

ASN ARCSINE
ACS ARCCOSINE
ATN ARCTANGENT

Function values are returned within the following ranges:

$-90° \leqq ASN\ (x) \leqq 90°$, $0° \leqq ACS\ (x) \leqq 180°$

$-90° \leqq ATN\ (x) \leqq 90°$

SEE: ANGLE, SIN, COS, TAN

SAMPLE PROGRAM:

```
10 ANGLE 1
20 INPUT "INPUT NUMBER (-1 TO 1)" ; N
30 PRINT N ; "=SIN (" ; ASN N ; "RAD)"
40 PRINT N ; "=COS (" ; ACS N ; "RAD)"
50 PRINT N ; "=TAN (" ; ATN N ; "RAD)"
60 ANGLE 0 : END
```

Displays trigonometric angles in radians for each input in range of −1 to 1.

HYP SIN
HYP COS
HYP TAN

Ⓕ

PURPOSE: Returns the value of the corresponding hyperbolic function for the argument.

FORMAT:

HYP SIN (argument) — Numeric expression

HYP COS (argument) — Numeric expression

HYP TAN (argument) — Numeric expression

* The parentheses enclosing the argument can be omitted when the argument is a numeric value or variable.

EXAMPLE: HYP SIN (1.5)

PARAMETERS: argument: Numeric expression

HYP SIN $|argument| \leqq 230.2585092$

HYP COS $|argument| \leqq 230.2585092$

EXPLANATION:

Returns the value of the corresponding hyperbolic function for the argument.

HYP SIN (x) : $\sinh x = (e^x - e^{-x})/2$

HYP COS (x) : $\cosh x = (e^x + e^{-x})/2$

HYP TAN (x) : $\tanh x = (e^x - e^{-x})/(e^x + e^{-x})$

SEE: HYP ASN, HYP ACS, HYP ATN

SAMPLE PROGRAM:

```
10  INPUT "INPUT NUMBER (UP TO 230)" ; N
20  PRINT "HSN (" ; N ; ")=" ; HYPSIN N
30  PRINT "HCS (" ; N ; ")=" ; HYPCOS N
40  PRINT "HTN (" ; N ; ")=" ; HYPTAN N
50  END
```

Displays the hyperbolic functions for numeric input up to 230.

HYP ASN
HYP ACS
HYP ATN

Ⓕ

PURPOSE: Returns the value of the corresponding inverse hyperbolic function for the argument.

FORMAT:

HYP ASN (argument) — Numeric expression

HYP ACS (argument) — Numeric expression

HYP ATN (argument) — Numeric expression

- The parentheses enclosing the argument can be omitted when the argument is a numeric value or variable.

SAMPLE: HYP ASN (10)

PARAMETERS: argument: Numeric expression

HYP ASN $|\text{argument}| < 5 \times 10^{99}$ (5E+99)
HYP ACS $1 \leqq \text{argument} < 5 \times 10^{99}$ (5E+99)
HYP ATN $-1 < \text{argument} < 1$

EXPLANATION:
Returns the value of the corresponding inverse hyperbolic function for the argument.

HYP ASN (x) : $\sinh^{-1}x = \log_e(x + \sqrt{x^2+1})$

HYP ACS (x) : $\cosh^{-1}x = \log_e(x + \sqrt{x^2-1})$

HYP ATN (x) : $\tanh^{-1}x = \frac{1}{2} \log_e \frac{1+x}{1-x}$

SEE: HYP SIN, HYP COS, HYP TAN

SAMPLE PROGRAM:

```
10 INPUT "INPUT NUMBER (1 OR GREATER)" ; N
20 PRINT "HAS (" ; N ; ")=" ; HYPASN N
30 PRINT "HAC (" ; N ; ")=" ; HYPACS N
40 END
```

Displays inverse hyperbolic function value for numeric input of 1 or greater.

EXP

Ⓕ

PURPOSE: Returns the value of the exponential function for the argument.

FORMAT: EXP (argument)
Numeric expression

- The parentheses enclosing the argument can be omitted when the argument is a numeric value or variable.

EXAMPLE: EXP (1)

PARAMETERS: argument: Numeric expression in the range of argument ≦ 230.2585092

EXPLANATION:

Returns the value of the exponential function value for the argument.

EXP $(x) = e^x$

SEE: LOG, LN

SAMPLE PROGRAM:

```
10  INPUT "e^X (UP TO 230)" ; N
20  PRINT "e^" ; N ; "=" ; EXP N
30  END
```

Displays exponential function value for numeric input up to 230.

LOG
LN

Ⓔ

PURPOSE: Returns the value of the corresponding logarithm function for the argument.

FORMAT:

LOG (argument) — Numeric expression

LN (argument) — Numeric expression

- The parentheses enclosing the argument can be omitted when the argument is a numeric value or variable.

SAMPLE: LOG (2), LN (3)

PARAMETERS: argument: Numeric expression
LOG : 0 < argument
LN : 0 < argument

EXPLANATION:

Returns the value of the corresponding logarithm function value for the argument.

LOG : Common logarithm $\log_{10}x$, $\log x$
LN : Natural logarithm $\log_e x$, $\ln x$

SAMPLE PROGRAM:

```
10 INPUT "INPUT NUMBER" ; N
20 PRINT "LOG" ; N ; "=" ; LOG N
30 PRINT "LN" ; N ; "=" ; LN N
40 END
```

Displays logarithm function values for numeric input greater than 0.

SQR

Ⓕ

PURPOSE: Returns the square root of the argument.

FORMAT: SQR (argument)
Numeric expression

* The parentheses enclosing the argument can be omitted when the argument is a numeric value or variable.

EXAMPLE: SQR (4)

PARAMETERS: argument: Numeric expression in the range of $0 \leqq$ argument

EXPLANATION:

Returns the square root of the argument.

SQR (x) : $\sqrt{x}$

SAMPLE PROGRAM:

```
10  FOR I=0 TO 10
20  PRINT "SQR" ; I ; SQR I
30  NEXT I
40  END
```

Displays square roots of values from 0 through 10.

CUR

Ⓕ

PURPOSE: Returns cube root of argument.

FORMAT: CUR (argument)
Numeric expression

* The parentheses enclosing the argument can be omitted when the argument is a numeric value or variable.

EXAMPLE: X = CUR (Y)

PARAMETERS: argument: Numeric expression

EXPLANATION:

Returns the cube root of the argument.

CUR (x) : $\sqrt[3]{x}$

SAMPLE PROGRAM:

```
10  A=27
20  PRINT A ; " , CUBE ROOT" ; CUR A
30  END
```

Returns cube root of value assigned to variable A.

ABS

Ⓕ

PURPOSE: Returns the absolute value of the argument.

FORMAT: ABS (argument)
Numeric expression

- The parentheses enclosing the argument can be omitted when the argument is a numeric value or variable.

SAMPLE: ABS (−1.5)

PARAMETERS: argument: Numeric expression

EXPLANATION:

Returns the absolute value of the argument.

ABS (x) : $|x|$

SAMPLE PROGRAM:

```
10  INPUT "INPUT NUMBERS" ; N
20  A=ABS N
30  PRINT N ; "ABS ( )=" ; A
40  END
```

Displays the absolute value of an input value.

SGN Ⓕ

PURPOSE: Returns a value which corresponds to the sign of the argument.

FORMAT: SGN (argument)
Numeric expression

* The parentheses enclosing the argument can be omitted when the argument is a numeric value or variable.

EXAMPLE: SGN (A)

PARAMETERS: argument: Numeric expression

EXPLANATION:
Returns a value of −1 when the argument is negative, 0 when the argument equals 0, and 1 when the argument is positive.

Argument (X)	SGN (X)
X<0	−1
X=0	0
X>0	1

SAMPLE PROGRAM:

```
10  INPUT "INPUT NUMBER" ; N
20  S=SGN N
30  IF S<>0 THEN PRINT "NOT ZERO" : END
40  PRINT "ZERO" : END
```

Uses SGN function to determine whether or not an input value equals 0.

INT Ⓕ

PURPOSE: Returns the largest integer which does not exceed the value of the argument.

FORMAT: INT (argument)
Numeric expression

- The parentheses enclosing the argument can be omitted when the argument is a numeric value or variable.

SAMPLE: INT (1.3)

PARAMETERS: argument: Numeric expression

EXPLANATION:
Returns the largest integer which does not exceed the value of the argument.
INT (x) is equivalent to FIX (x) when x is positive, and FIX (x) $-$ 1 when x is negative.

SEE: FIX, FRAC

SAMPLE PROGRAM:

```
10  FOR I=1 TO 10
20  N=RAN#*10
30  LPRINT "INT (" ; N ; ")=" ; INT N
40  NEXT I
50  END
```

Converts random values to integers and outputs results to printer.

FIX Ⓕ

PURPOSE: Returns the integer part of the argument.

FORMAT: FIX (argument)
Numeric expression

- The parentheses enclosing the argument can be omitted when the argument is a numeric value or variable.

SAMPLE: FIX (−1.5)

PARAMETERS: argument: Numeric expression

EXPLANATION:
Returns the integer part of the argument.

SEE: INT

SAMPLE PROGRAM:

```
10  INPUT A
20  PRINT "FIX (" ; A ; ")=" ; FIX A
30  GOTO 10
```

Displays the integer part of input values.

FRAC

Ⓕ

PURPOSE: Returns the fractional part of the argument.

FORMAT: FRAC (argument)
Numeric expression

- The parentheses enclosing the argument can be omitted when the argument is a numeric value or variable.

EXAMPLE: FRAC (3.14)

PARAMETERS: argument: Numeric expression

EXPLANATION:

1. Returns the fractional part of the argument.
2. The sign (±) of the value is the same as that for the argument.

SAMPLE PROGRAM:

```
10  FOR I=1 TO 10
20  N=RAN#*10
30  LPRINT "FRAC (" ; N ; ")=" ; FRAC N
40  NEXT I
50  END
```

Isolates fractional parts of random values and outputs results to printer.

ROUND

Ⓕ

PURPOSE: Rounds the argument at the specified digit.

FORMAT: ROUND (argument, digit)

EXAMPLE: ROUND (A, −3)

PARAMETERS:
1. argument: Numeric expression
2. digit: Numeric expression truncated to an integer in the range of −100<digit<100

EXPLANATION:
Rounds the argument (to the nearest whole number) at the specified digit.

SAMPLE PROGRAM:

```
10  N=RAN#*1000
20  PRINT N
30  INPUT "WHERE" ; R
40  PRINT ROUND (N, R)
50  END
```

Displays random value and then rounds value at digit specified by numeric input.
For example, responding to prompt "WHERE" with input of −2 when N=610.5765383 produces result of 610.6.

RAN#

Ⓔ

PURPOSE: Returns a random value in the range of 0 to 1.

FORMAT: RAN# (argument)
Numeric expression

- The parentheses enclosing the aurgument can be omitted when the argument is a numeric value or variable.

EXAMPLE: RAN# * 10

PARAMETERS: argument: Numeric expression

EXPLANATION:

Returns a random value in the range of 0 to 1. (0<RAN#(X)<1)
Random numbers are generated from the same table when X=1.
The last random number generated is repeated when X=0.
Random numbers are generated from the random table when X= −1.
Random number generation begins with the same value each time a program is executed. This means that the same series of numbers is generated unless the argument of RAN# is omitted or is equal to −1.

SAMPLE PROGRAM:

```
10  R=RAN# (1) : PRINT R
20  R=RAN# (0) : PRINT R
30  R=RAN# (-1) : PRINT R
40  GOTO 10
```

Generates random numbers using each type (positive, negative, zero) of argument.

PI Ⓕ

PURPOSE: Returns the value of π.

FORMAT: PI

EXAMPLE: S=2*PI*R

EXPLANATION:

1. Returns the value of π.
2. The value of π used for internal calculations is 3.1415926536.
3. The displayed value is rounded off to 10 digits, so the value of π is displayed as 3.141592654.

SAMPLE PROGRAM:

```
10  INPUT "RADIUS" ; R
20  PRINT "CIRCUMFERENCE=" ; 2*PI*R
30  PRINT "AREA=" ; R^2*PI
40  END
```

Calculates circumference and area of circle after input of radius.

FACT Ⓕ

PURPOSE: Returns factorial of argument.

FORMAT: FACT (argument)
Numeric expression (argument)

EXAMPLE: A=FACT (10)

PARAMETER: argument : Integer in the range of 0≦argument≦69

EXPLANATION:

1. Returns the factorial of the argument.
   FACT x : x!
2. A fractional value as the argument generates an error.

SAMPLE PROGRAM:

```
10  X=5
20  Y=FACT X
30  PRINT X ; "!=" ; Y
```

Assigns factorial of the value of variable X to variable Y and displays the result.

NPR

Ⓕ

PURPOSE: Returns the permutation nPr for the values of n and r.

FORMAT: NPR (n value , r value)

n value: Numeric expression; r value: Numeric expression

SAMPLE: X = NPR (69, 20)

PARAMETERS: n :, r : Integer in the range of $0 \leqq r \leqq n < 10^{10}$

EXPLANATION:

Returns the permutation: $nPr = \dfrac{n!}{(n-r)!}$

A fractional value as either n or r generates an error.

SAMPLE PROGRAM:

```
10 N=10 : P=5
20 X=NPR (N, R)
30 PRINT X
```

Calculates $_{10}P_5$ and displays the result.

NCR

Ⓕ

PURPOSE: Returns the combination nCr for the values of n and r.

FORMAT: NCR (n value , r value)

n value: Numeric expression; r value: Numeric expression

SAMPLE: X = NCR (70, 35)

PARAMETERS: n :, r : Integer in the range of $0 \leqq r \leqq n < 10^{10}$

EXPLANATION:

Returns the combination: $nCr = \dfrac{n!}{r!\,(n-r)!}$

A fractional value as either n or r generates an error.

SAMPLE PROGRAM:

```
10 N=8 : R=4
20 X=NCR (N, R)
30 PRINT X
```

Calculates $_8C_4$ and displays the result.

POL

Ⓕ

PURPOSE: Converts rectangular coordinates (x, y) to polar coordinates (r, θ).

FORMAT: POL (x-coordinate , y-coordinate)

x-coordinate: Numeric expression; y-coordinate: Numeric expression

EXAMPLE: POL (3, 2)

PARAMETERS: x-coordinate :, y-coordinate : Numeric expression. $|x| + |y| > 0$

EXPLANATION:

1. Converts rectangular coordinates (x, y) into polar coordinates (r, θ). The following relational expressions are used at this time:

$$r = \sqrt{x^2+y^2} \qquad \cos\theta = \frac{x}{\sqrt{x^2+y^2}} \qquad \sin\theta = \frac{y}{\sqrt{x^2+y^2}}$$

2. The value of r is automatically assigned to variable X, while θ is automatically assigned to variable Y.
3. The value of θ is given as follows:

$-180° < \theta \leqq 180°$ (DEG)

$-\pi < \theta \leqq \pi$ (RAD)

-200 grads $< \theta \leqq 200$ grads (GRA)

SAMPLE PROGRAM:

```
10  A=5 : B=3
20  Z=POL (A, B)
30  PRINT X ; Y
```

Converts rectangular coordinate point (5, 3) to polar coordinates.

REC

Ⓔ

PURPOSE: Converts polar coordinates (r, θ) to rectangular coordinates (x, y).

FORMAT: REC (distance r , angle θ)

distance r: Numeric expression; angle θ: Numeric expression

SAMPLE: REC (10, 15)

PARAMETERS:
distance r: $0 \leq r < 10^{100}$
angle θ: $-1440° < \theta < 1440°$ (DEG)
$-8\pi < \theta < 8\pi$ (RAD)
-1600 (grads) $< \theta < 1600$ (grads) (GRA)

EXPLANATION:

Converts polar coordinates (r, θ) to rectangular coordinates (x, y). The following relational expressions are used at this time:

$x = r\cos\theta$, $y = r\sin\theta$

The value of x is automatically assigned to variable X, while y is automatically assigned to variable Y.

SAMPLE PROGRAM:

```
10  A=2 : B=30
20  Z=REC (A, B)
30  PRINT X ; Y
```

Converts polar coordinate point (2, 30) to rectangular coordinates.

CHR$ ⓕ

PURPOSE: Returns a single character which corresponds to the specified character code.

FORMAT: CHR$ (code)
Numeric expression

EXAMPLE: CHR$ (65)

PARAMETERS: code: Numeric expression truncated to an integer in the range of $0 \leq code < 256$

EXPLANATION:
Variables can also be used as a parameter, and decimal parts of numeric values are truncated. A null is returned when a character does not exist for the specified character code.

SEE: ASC

SAMPLE PROGRAM:

```
10  FOR I=65 TO 90
20  PRINT CHR$ ( I ) ;
30  NEXT I
```

Displays characters from 65 through 90 in character code.

ASC Ⓕ

PURPOSE: Returns the character code corresponding to the character in the first (leftmost) position of a string.

FORMAT: ASC (string)
String expression

EXAMPLE: ASC ("A")

PARAMETERS: string: String expression

EXPLANATION:
Returns the character code corresponding to a character. The character code for the first (leftmost) character only is returned for a string of two or more characters long.
A value of 0 is returned for a null string.

SEE: CHR$, Character Code Table

SAMPLE PROGRAM:

```
10  INPUT "INPUT CHARACTERS" ; A$
20  B$=LEFT$ (A$,1)
30  C=ASC (A$)
40  PRINT "FIRST CHAR=" ; B$ ; "CODE=" ; C
50  END
```

Displays first character and corresponding character code for string input.

STR$ Ⓔ

PURPOSE: Converts the argument (numeric value or numeric expression value) to a string.

FORMAT: STR$ (argument)
String expression

EXAMPLE: STR$ (123), STR$ (255+3)

PARAMETERS: argument: Numeric expression

EXPLANATION:

1. Converts decimal values specified in the argument to strings.
2. Converted positive values include a leading space and converted negative values are preceded by a minus sign.

SEE: VAL

SAMPLE PROGRAM:

```
10  INPUT "INPUT NUMBERS" ; N
20  S$=STR$ (N)
30  C$=MID$ (S$, 2, 1)
40  PRINT "FIRST CHARACTER=" ; C$
50  END
```

Converts numeric input to a string. Next, the first number of converted string is displayed as character.

VAL

Ⓕ

PURPOSE: Converts a numeric character string to a numeric value.

FORMAT: VAL (string)
String expression

SAMPLE: A = VAL ("345")

PARAMETERS: string: String expression

EXPLANATION:

Converts a numeric character string to a numeric value.
Numeric characters are converted up to the point in the string that a non-numeric character is encountered. All subsequent characters are disregarded from the non-numeric character onwards. (i.e. when A = VAL ("123A456"), A = 123).
The value of this function becomes 0 when the length of the string is 0 or when the leading character is non-numeric.

SEE: STR$

SAMPLE PROGRAM:

```
10 INPUT "VALUE1" , A$
20 INPUT "VALUE2" , B$
30 C$ = A$ + B$
40 C = VAL (A$) + VAL (B$)
50 PRINT C$, C
```

Performs string addition and numeric addition of two input strings.

VALF

ⓕ

PURPOSE: Performs calculation of numeric expression expressed as string, and returns the result.

FORMAT: VALF (string)
String expression

EXAMPLE: VALF (X$)

PARAMETERS: string: String expression

EXPLANATION:

1. Performs calculation of numeric expressions which are expressed as strings, and returns their results.
2. An error is generated when an intermediate or final result of calculation exceeds 10^{100}.
3. VALF cannot be used within a VALF argument.

SAMPLE PROGRAM:

```
10  XS = "123 + 456"
20  PRINT VALF (XS)
30  PRINT VALF ("EXP 2")

RUN
579
7.389056099
```

Executes strings "123 + 456" and "EXP 2" as numeric expressions and displays results.

MID$

Ⓕ

PURPOSE: Returns a substring of a specified length from a specified position within a string.

FORMAT: MID$ (string , position [, number of characters])
String expression / Numeric expression / Numeric expression

SAMPLE: MID$ (A$, 5, 3)

PARAMETERS:
1. string: String expression
2. position: Numeric expression truncated to an integer in the range of $1 \leq$ position < 256
3. number of characters: Numeric expression truncated to an integer in the range of $0 \leq$ number of characters < 256. The default option is from the specified position to the end of the string when this parameter is omitted.

EXPLANATION:

Returns a substring of a specified length from a specified position within a string. A substring from the specified position to the end of the string is returned when the length of the substring is not specified.

A substring of length 0 (null) is returned when the specified position exceeds the length of the string.

A substring from the specified position to the end of the string is returned when the specified number of characters is greater than the number of characters from the specified position to the end of the string.

SEE: RIGHT$, LEFT$

SAMPLE PROGRAM:

```
10  A$="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
20  INPUT "1 TO 26 FROM" ; B
30  PRINT "1 TO" ; 27- B ; "TO" ;
40  INPUT E
50  S$=MID$ (A$, B, E)
60  PRINT S$
70  END
```

Uses numeric input to produce alphabetic series of a specified number of characters starting from a specified location.

RIGHT$

Ⓕ

PURPOSE: Returns a substring of a specified length counting from the right of a string.

FORMAT: RIGHT$ (string , number of characters)
String expression — Numeric expression

EXAMPLE: RIGHT$ ("ABCDEF", 3)

PARAMETERS:
1. string: String expression
2. number of characters: Numeric expression truncated to an integer in the range of $0 \leq$ number of characters < 256.

EXPLANATION:
1. Returns a substring of a specified length counting from the right of string.
2. The entire string is returned as the substring when the specified number of characters is greater than the number of characters in the string.

SEE: MID$, LEFT$

SAMPLE PROGRAM:

```
10 A$="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
20 PRINT A$
30 INPUT "1 TO 26 HOW MANY GET" ; N
40 PRINT RIGHT$ (A$, N)
50 END
```

Uses numeric input to display specified number of characters from end of alphabetic sequence.

LEFT$

Ⓔ

PURPOSE: Returns a substring of a specified length counting from the left of a string.

FORMAT: LEFT$ (string , number of characters)

String expression — Numeric expression

SAMPLE: LEFT$ ("ABCDEF", 3)

PARAMETERS: 1. string: String expression
2. number of characters: Numeric expression truncated to an integer in the range of $0 \leqq$ number of characters < 256.

EXPLANATION:

Returns a substring of a specified length counting from the left of string.
The entire string is returned as the substring when the specified number of characters is greater than the number of characters in the string.

SEE: MID$, RIGHT$

SAMPLE PROGRAM:

```
10  A$="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
20  PRINT A$
30  INPUT "1 TO 26 HOW MANY GET" ; N
40  PRINT LEFT$ (A$, N)
50  END
```

Uses numeric input to display specified number of characters from beginning of alphabetic sequence.

LEN

Ⓕ

PURPOSE: Returns a value which represents the number of characters contained in a string.

FORMAT: LEN (string)
String expression

EXAMPLE: LEN (A$)

PARAMETERS: string: String expression

EXPLANATION:
Returns a value which represents the number of character contained in a string, including characters that don't appear on the display (character codes from &H0 ~ &H1F) and spaces.

SAMPLE PROGRAM:

```
10 INPUT "INPUT CHARACTERS" ; C$
20 PRINT "LENGTH=" ; LEN (C$)
30 END
```

Determines the length of an input string.

HEX$

Ⓕ

PURPOSE: Returns a hexadecimal string for a decimal value specified in the argument.

FORMAT: HEX$ (argument)
Numeric expression

EXAMPLE: HEX$ (15)

PARAMETERS: argument: Numeric expression truncated to an integer in the range of −32769 < argument < 65536. Values more than 32767 are converted by subtracting 65536.

EXPLANATION:
Returns a 4-digit hexadecimal string for a decimal value specified in the argument.

SEE: &H

SAMPLE PROGRAM:

```
10 PRINT "DECIMAL" ; TAB (10) ; "HEX"
20 FOR I=0 TO 16
30 PRINT I ;
40 PRINT TAB (10) ; HEX$ ( I ) ; : PRINT
50 FOR J=0 TO 250 : NEXT J
60 NEXT I
70 END
```

Displays the decimal values from 0 through 16 along with their hexadecimal equivalents.

&H

Ⓐ

PURPOSE: Converts the 1 through 4-digit hexadecimal value following &H to a decimal value.

FORMAT: &H argument
Hexadecimal value

SAMPLE: A = &HAF

PARAMETERS: $0_H \leq$ argument $\leq FFFF_H$

EXPLANATION:

The hexadecimal value is expressed using values 0 through 9, plus characters A through F. In the manual mode, &H is entered followed by the hexadecimal value. Pressing [EXE] produces the decimal equivalent.

Example: &H1B7F [EXE] → 7039

The following shows a typical application within a program. Since a numeric variable cannot be used following &H, the hexadecimal value is appended to &H as a string and then converted to a decimal value using the VAL function.

SEE: HEX$

SAMPLE PROGRAM:

```
10 REM &H SAMPLE
20 INPUT "&H" ; A$
30 H=VAL ("&H"+A$)
40 PRINT "&H" ; A$ ; "=" ; H
50 GOTO 10
```

Converts an entered hexadecimal value (4 digits max) to a decimal value. Program execution is terminated using the [BRK] key.

DEG Ⓕ

PURPOSE: Converts a sexagesimal value to a decimal value.

FORMAT: DEG (degrees [, minutes [, seconds]])

- degrees: Numeric expression
- minutes: Numeric expression
- seconds: Numeric expression

EXAMPLE: DEG (1, 30, 10)

PARAMETERS: Degrees, minutes, seconds: $|DEG (degrees, minutes, seconds)| < 10^{100}$

EXPLANATION:

Converts the degrees, minutes, and seconds of sexagesimal values to decimal values as follow:

DEG (degrees, minutes, seconds) = degrees + minutes/60 + seconds/3600

SAMPLE PROGRAM:

```
10  INPUT "DEGREES=" , A
20  INPUT "MINUTES=" , B
30  INPUT "SECONDS=" , C
40  D=DEG (A, B, C)
50  PRINT D
60  END
```

Converts values entered for degrees, minutes, and seconds into a decimal value.

DMS$ Ⓕ

PURPOSE: Converts a decimal value to a sexagesimal string.

FORMAT: DMS$ (argument)
(argument: Numeric expression)

EXAMPLE: DMS$ (1.52)

PARAMETERS: argument:
Numeric expression in the range of |numeric expression| < 10^{100}

EXPLANATION:

Converts decimal values to sexagesimal strings.
Minutes and seconds are not displayed when the argument is in the range of numeric expression ≧ 1 × 10^{6} (1E6). In this case, the absolute value of the input value is converted to a string as it is.

SAMPLE PROGRAM:

```
10  INPUT "INPUT NUMBER" ; N
20  PRINT "=" ; DMS$ (N)
30  END
```

Converts input decimal values to sexagesimal strings.

LLIST

Ⓜ

PURPOSE: Outputs program contents to the printer.

FORMAT: LLIST { [starting line number] [– [ending line number]] / [.] / [ALL] }
(starting line number: Line number; ending line number: Line number)

EXAMPLE: LLIST 50 – 100

PARAMETERS: Both the starting line number and ending line number are within the range of 1 ≦ line number ≦ 65535. The last line number used by BASIC is specified when "." is used.

1. starting line number: Program line number from which program content printout is to begin. The default option is the first line of the program.
2. ending line number:* Program line number at which program content printout is to end. The default option is the last line of the program.
3. Specifying ALL sequentially outputs all program contents in areas P0 through P9.

EXPLANATION:

1. Outputs program contents to the printer within the specified range.
2. This statement differs from LIST in that output is to the printer without showing program contents on the display.
3. LLIST cannot be used in the CAL mode.

SAMPLE EXECUTION: LLIST EXE

Outputs contents of current program area to printer.

LPRINT Ⓐ

PURPOSE: Outputs text to the printer.

FORMAT: LPRINT [output data] [{, ;} [output data]]*

Output data: [TAB (numeric expresion) / Numeric expression / String expression]

SAMPLE: LPRINT A, B

PARAMETERS: output data: Output control function, numeric expression, or string expression

EXLANATION:

Outputs data to the printer. When the output data is a control function, the corresponding operation is performed. Numeric or string expressions as output data result in printout of the resulting value.

Numeric expression values are printed in decimal, and the print format is the same as that for the PRINT statement (see PRINT).

ring expression values are output as they are to the printer.

Including a comma between output data causes a zone tab to be inserted between output data at output.

›ne tabs are set at 14-character intervals (counting from 0, within a range of 255 charac- ·s) following the last carrier return instruction, and zone tab outputs spaces from the rrent location to the next zone tab. Consequently, the printing of the first character of output data following a comma is performed at the next zone tab.

```
10 LPRINT
20 FOR I=1 TO 20 : LPRINT "*" , : NEXT I
30 LPRINT
40 END
```

:luding a semicolon between output data causes the output data to be output sequentially.

```
10 LPRINT
20 FOR I=1 TO 50
30 LPRINT "(" ; I ; ")"
40 NEXT I
50 LPRINT
60 END
```

:luding a semicolon at the end of the statement causes the location immediately follow- ȷ printout of the last output data to be the next printing position.

:luding a comma at the end of an LPRINT statement performs a zone tab following print- t of the last output data.

:arrier return is performed when a semicolon or comma is not included at the end of ᴈ statement. Print positions are counted from 0 through 255, and the count is reset to vhen it exceeds 255. Zone tabs and the TAB function are performed in accordance with ᴈ print position count. CR–LF (internal code 0DH, 0AH) is performed at this time.

9. Actual printing begins when a carrier return/line feed code is sent, and carrier return/line feed is performed automatically when printing reaches the extreme right of the paper

SEE: PRINT

SAMPLE PROGRAM:

```
10  LPRINT
20  FOR I=1 TO 14 : LPRINT "*" ; : NEXT I
30  LPRINT
40  END
```

Outputs a series of 14 asterisks to printer.

OPEN

Ⓟ

PURPOSE: Declares a file open for use.

FORMAT:

OPEN "file descriptor" [FOR { INPUT / OUTPUT }] AS[#] file number
(file number: Numeric expression)

SAMPLE: OPEN "DATA1" FOR OUTPUT AS #1

PARAMETERS:
1. file descriptor: String expression
2. file number: Numeric expression truncated to an integer in the range of $1 \leq$ file number < 2

EXPLANATION:

ens the file specified by the file descriptor as the specified file number. Subsequent
ut to and output from open files is performed by designating the file numbers.
S0: is the default option when the device name is omitted from the file descriptor.
cifying FOR INPUT makes sequential file input possible.
cifying FOR OUTPUT makes sequential file output possible. A new file is created on cassette tape.
following two conditions are specified when either FOR INPUT or FOR OUTPUT is specified:
Cassette tape (CAS0: , CAS1:)
Error generated
Communications circuit (COM0:)
Sequential file input/output specified
y one file (#1) can be open at any given time. Attempting to open two or more files
ults in an OP error.
empting to open a file which is already open results in an OP error.
le buffer is automatically retained within the stack area. An OM error is generated
enever the stack area becomes full.
s command can only be executed within a program.

CLOSE

SAMPLE PROGRAMS:

1)
```
10 OPEN "CAS0 : TEST" FOR OUTPUT AS #1
20 PRINT #1, "WRITE TEST"
30 CLOSE
```

Creates sequential file on cassette tape under filename "TEST".

2)
```
10 OPEN "CAS0 : TEST" FOR INPUT AS #1
20 INPUT #1, A$
30 CLOSE
```

Reads sequential file created in SAMPLE 1.

CLOSE Ⓐ

PURPOSE: Closes files and declares an end to the use of the I/O (input/output) buffer.

FORMAT: CLOSE

EXAMPLE: CLOSE

EXPLANATION:

1. Closes a file and clears the file buffer.
2. An error is not generated even if a file is not open when this command is executed.

SEE: OPEN

SAMPLE PROGRAM:

```
10 OPEN "CAS0 : TEST" FOR INPUT AS #1
20 INPUT #1, A$ : PRINT A$ ;
30 IF EOF (1)=0 THEN 20
40 CLOSE
```

Reads and displays data from sequential file TEST (stored on cassette tape) until all data have been read.

PRINT # ⓟ

PURPOSE: Outputs data to a sequential file.

FORMAT: PRINT # file number [, output data [{;/,} [output data]]*]
(file number: Numeric expression)

Output data: TAB / String expression / Numeric expression

SAMPLE: PRINT #1, A$

PARAMETERS: file number: Numeric expression truncated to an integer in the range of $1 \leq$ file number < 2

EXPLANATION:

Sequentially outputs data to the sequential file specified by the file number.
:e contents of the output data are the same as those output to the printer by the LPRINT ıtement (see LPRINT, PRINT).
CR-LF (0DH, 0AH) is output following the last output data when a semicolon and comma ǝ not included.
is statement is valid for sequential files opened for output (FOR OUTPUT), and for com-ɹnication circuit (COM0:) input/output files.

INPUT #, PRINT, LPRINT

SAMPLE PROGRAMS:

1)
```
10 OPEN "CAS0 : TEST" FOR OUTPUT AS #1
20 INPUT "DATA = " , A$
30 IF A$ = " " THEN 60
40 PRINT #1, A$
50 GOTO 20
60 CLOSE : END
```

Creates sequential file on cassette tape for input of characters from keyboard.

2)
```
10 OPEN "CAS0 : TEST" FOR INPUT AS #1
20 INPUT #1, A$
30 CLOSE
```

Reads sequential file on cassette tape created in SAMPLE PROGRAM 1.

INPUT

Ⓟ

PURPOSE: Reads data from a sequential file.

FORMAT: INPUT # file number (Numeric expression) , variable name [, variable name]*

EXAMPLE: INPUT #1, A

PARAMETERS: file number: Numeric expression truncated to an integer in the range of 1 ≤ file number < 2

EXPLANATION:

1. Reads data from the file specified by the file number.
2. Data are input in the same format as data input using the INPUT statement (see INPUT). Consequently, data are delimited using commas, quotation marks, CR codes ($0D_H$) or CR, LF codes ($0D_H$, $0A_H$). Internal codes 00_H through $1F_H$ and $7F_H$ cannot be input, and leading spaces (spaces preceding that data) are disregarded.
3. This statement is valid for sequential files opened for input (FOR INPUT), and for communication circuit (COM0:) input/output files.
4. Spaces can also be used as delimiters when data are read to numeric variables.
5. An ST error is generated when data read exceeds 256 characters. Execution continued using an ON ERROR statement begins from the 257th character.

SAMPLE PROGRAM:

```
10  OPEN "CAS0 : TEST" FOR INPUT AS #1
20  INPUT #1, A$
30  PRINT A$ ;
40  IF EOF (1)=0 THEN 20
50  CLOSE : END
```

Reads and displays data in a sequential file on cassette tape until no more data remain.

INPUT$

Ⓕ

PURPOSE: Reads the specified number of characters from a sequential file.

FORMAT: INPUT$ (number of characters , [#] file number)

Numeric expression (number of characters); Numeric expression (file number)

EXAMPLE: INPUT$ (16, #1)

PARAMETERS:
1. number of characters: Numeric expression truncated to an integer in the range of 0≦number of characters<256
2. file number: Numeric expression truncated to an integer in the range of 1≦file number<2

EXPLANATION:
1. Reads the specified number of characters from a sequential file.
2. All codes (00_H ~ FF_H) are read as they are.
3. This statement is valid for sequential files opened for input (FOR INPUT), and for communication circuit (COM0:) input/output files.

SAMPLE PROGRAM:

```
10 OPEN "CAS0 : TEST" FOR INPUT AS #1
20 CH$=INPUT$ (5, #1)
30 CLOSE
40 PRINT CH$
```

Reads and displays first five characters in a sequential file on cassette tape.

EOF

PURPOSE: Indicates the end of file reading.

FORMAT: EOF (file number)
Numeric expression

EXAMPLE: IF EOF (1) THEN END

PARAMETERS: file number: Numeric expression truncated to an integer in the range of 1 ≦ file number < 2

EXPLANATION:

1. Indicates the end of reading for the file specified by the file number. Generally, this function is assigned a value of 0, but the value becomes −1 when the last record of a file is read.
2. A value of −1 is returned when the receive buffer (for RS-232C applications) becomes empty.
3. This statement is valid for sequential files opened for input (FOR INPUT), and for communications circuit (COM0:) input/output files.
4. Generally, a 0 is returned for sequential files opened for output (FOR OUTPUT).

SAMPLE PROGRAM:

```
10  OPEN "CAS0 : TEST" FOR INPUT AS #1
20  INPUT #1, A$
30  PRINT A$
40  IF EOF (0) THEN 20
50  CLOSE
60  END
```

Reads and displays data in sequential file on cassette tape until no more data remain.

SAVE, SAVE ALL

PURPOSE: Saves a program to a specified file.

FORMAT: SAVE [ALL] "file descriptor" [, A]
(file descriptor: String expression)

EXAMPLE: SAVE "DEMO1"

PARAMETERS:
1. ALL: Outputs all programs from P0 through P9. Can only be specified for output to cassette tape.
2. file descriptor: String expression
3. , A: Specifies ASCII format. Binary internal format is the default option when omitted. Cannot be specified while SAVE ALL is specified.

EXPLANATION:
1. Outputs the currently specified program area contents to the file specified by the file descriptor.
2. Specifying ALL outputs programs from areas P0 through P9 to cassette tape as an ALL file.
3. CAS0: is the default option when the device name is omitted from the fie descriptor. When the entire file descriptor is omitted, the file is output to cassette tape and saved without a filename.
4. Specifying ", A" causes the program to be converted to and saved in ASCII format. This format uses alphabetic characters such as those which appear when the LIST command is executed.
5. Data are output as they are in binary format when ", A" is not specified. However, files are saved in ASCII format whenever COM0: is specified in the file descriptor, regardless of the ", A" specification.
6. Files for which a password has been registered cannot be saved in ASCII format.
7. This command causes all open files to be closed and enters command input standby once the SAVE execution is complete.
8. This command cannot be executed while program execution is halted ("STOP" displayed).
9. This command cannot be executed in the CAL mode.

SEE: LOAD, PASS, LOAD ALL

SAMPLE EXECUTION: SAVE "CAS0 : TEST"

Saves a program on cassette tape under filename TEST.

LOAD, LOAD ALL

Ⓜ

PURPOSE: Reads from a file into memory.

FORMAT: LOAD [ALL] "file descriptor" [, A]
String expression

EXAMPLE: LOAD "DEMO1"

PARAMETERS:
1. ALL: Inputs programs to program areas P0 through P9. Can only be specified for input from cassette tape.
2. file descriptor: String expression
3. , A: Specifies ASCII format for cassette tape. Binary format is the default option when , A is omitted. ASCII format is the default option for the communications circuit, whether specified or not.

EXPLANATION:
1. Reads from the file specified by the file descriptor to the currently specified program area. The format of the file can be either internal or ASCII format.
2. CAS0: is the default option when the device name is omitted from the file descriptor.
3. Files already in existence before execution of this command are deleted, and the specified files are loaded in their place.
4. This command closes all open files and the computer waits for command input once load is complete.
5. Passwords and program loading.

Computer	Loaded program	Result
Password	Password	LOAD possible when passwords are identical only
Password	No password	LOAD possible
No password	No password	LOAD possible
No password	Password	LOAD possible (under program password)

6. Specifying ALL reads ALL files (files with attribute A, created using SAVE ALL) from cassette tape into areas P0 through P9.
7. This command cannot be executed in the CAL mode.
8. This command cannot be executed while program execution is halted.
9. The first file on the cassette tape with an attribute which matches the one specified is the default option when the entire file descriptor is omitted.
   LOAD: first file saved in internal format (attribute B)
   LOAD ALL: first file saved in ALL format (attribute A)
   LOAD , A: first file saved in ASCII format (attribute S)

SEE: SAVE

SAMPLE EXECUTION: LOAD "CAS0 : TEST"

Reads program under filename TEST from cassette tape.

VERIFY

PURPOSE: Verifies the contents of a file stored on cassette tape.

FORMAT: VERIFY "file descriptor"
String expression

EXAMPLE: VERIFY "CAS0: DEMO"

PARAMETERS: file descriptor: String expression

EXPLANATION:

- Verifies the contents of a file stored on cassette tape.
- Parity and checksum data included within the file itself are used for checking.
- This command cannot be executed in the CAL mode.
- This command closes all open files.
- The first program found is checked when the filename is omitted.
- This command cannot be executed while program execution is halted ("STOP" displayed).

SEE: SAVE, LOAD

SAMPLE EXECUTION: VERIFY "CAS0 : TEST"

Confirms whether or not program on cassette tape has been correctly stored under filename TEST.

DATA BANK COMMANDS

NEW#

PURPOSE: Clears DATA BANK data.

EXPLANATION:
1. Clears all data stored under the DATA BANK function.
2. This command cannot be executed for data protected by a password.
3. This command cannot be executed in the CAL mode, but in the BASIC mode.

SAMPLE
EXECUTION: NEW# [EXE]

Clears DATA BANK data.

LIST#

PURPOSE: Displays all DATA BANK data.

EXPLANATION:
1. Displays in record sequence all data stored under the DATA BANK function.
2. The display shows the record number and DATA BANK data.
3. The listing can be halted at any time by pressing [STOP], and resumed by pressing any key other than [BRK] and [STOP].
4. The listing can also be halted at any time by pressing [BRK].
5. This command cannot be executed for data protected by a password.
6. This command cannot be executed in the CAL mode, but in the BASIC mode.

SEE: LLIST#

SAMPLE
EXECUTION: LIST# [EXE]

Lists DATA BANK data on display.

LLIST#

Ⓜ

PURPOSE: Outputs all DATA BANK data to printer.

EXPLANATION:

1. Outputs to the printer in record sequence all data stored under the DATA BANK function.
2. The record number and DATA BANK data are both printed.
3. This command cannot be executed for data protected by a password.
4. This command cannot be executed in the CAL mode, but in the BASIC mode.

SEE: LIST#

SAMPLE EXECUTION: LLIST# EXE

Outputs DATA BANK data to printer.

SAVE#

Ⓜ

PURPOSE: Outputs DATA BANK data to file specified by file descriptor.

FORMAT: SAVE# [file descriptor].
String expression

EXAMPLE: SAVE# "CAS0 : TEST"

PARAMETERS: file descriptor: String expression

EXPLANATION:

1. Outputs DATA BANK data to a file specified by the file descriptor.
2. Data are output in ASCII format.
3. CAS0: is the default option when the device name is omitted from the file descriptor.
4. When the entire file descriptor is omitted, the file is output to cassette tape and saved without a filename.
5. This command cannot be executed in the CAL mode, but in the BASIC mode.
6. This command cannot be executed while program execution is halted ("STOP" displayed).
7. This command cannot be executed for data protected by a password.

SEE: LOAD#

SAMPLE EXECUTION: SAVE# "CAS0 : TEL" EXE

Save DATA BANK data to cassette tape under filename TEL.

LOAD

Ⓜ

PURPOSE: Reads data into DATA BANK area.

FORMAT: LOAD # { [file descriptor] / String expression } [, M]

EXAMPLE: LOAD # "CAS0 : TEST"

PARAMETERS: 1. file descriptor: String expression
2. , M: Indicates that current execution is append to existing data.

EXPLANATION:

1. Reads data to the DATA BANK area from the file specified by the file descriptor.
2. The current contents of the DATA BANK area are deleted when ", M" is not specified. Specifying ", M" indicates that the new data are to be appended to the end of the current contents of the DATA BANK area.
3. CAS0: is the default option when the device name is omitted from the file descriptor.
4. The first file on the cassette tape with an attribute (S) which matches the one specified is the default option when the entire file descriptor is omitted.
5. This command cannot be executed in the CAL mode, but in the BASIC mode.

SAMPLE EXECUTION: LOAD # "CAS0 : TEL2" , M EXE

Reads memo data file stored on cassette tape under filename TEL2 and appends to current DATA BANK area contents.

READ # ℗

PURPOSE: Reads data from DATA BANK area.

FORMAT: READ# variable name [, variable name]*

EXAMPLE: READ# A$, X

PARAMETERS: variable name

EXPLANATION:

1. Sequentially reads data stored in the DATA BANK area and assigns them to variables.
2. Numeric data can only be read into numeric variables, and string data only into string variables. Mismatching data and variables generates an error.
3. Data items can be delimited by commas.
4. A DA error is generated when data are not present to be read.
5. The read sequence can be altered using the RESTORE# command.
6. Spaces in front of data items are skipped.
7. Data included within quotation marks are read as a single string.
8. This command cannot be executed in the CAL mode, but in the BASIC mode.

SEE: RESTORE#, WRITE#

SAMPLE PROGRAM:

```
10 RESTORE# "RED" , 0, 50
20 READ# A$
30 PRINT A$
40 GOTO 10
50 PRINT "NO DATA!"
```

Searches and displays data items which start with "RED" within DATA BANK area. Message "NO DATA!" appears when such data items are not found.

RESTORE# ℗

PURPOSE: Searches specific data in the DATA BANK area and changes the read sequence of DATA BANK data.

FORMAT: RESTORE# [{"object string" / String expression} [, {0 / 1}] [, {line number / #program area number}]]

EXAMPLE: RESTORE# "SMITH"

PARAMETERS:
1. object string: String expression
2. line number: Numeric expression. Integer within the range of $0 < \text{line number} < 65536$
3. program area number: Numeric expression. Integer within the range of $0 \leq \text{program area number} < 10$

EXPLANATION:
1. Searches specific data in the DATA BANK area and sets the DATA BANK area pointer position. Subsequent executions of the READ# and WRITE# commands are performed from the new pointer position.
2. The relationship between the parameters and the object string are as follows:
   i) RESTORE#
   Omitting all parameters sets the DATA BANK area pointer to the beginning of the data to be read by the next READ# command.
   ii) RESTORE# "object string"
   Sets the DATA BANK area pointer to the position of the specified object string. Strings are delimited by commas, and not by spaces. A DA error is generated when the object string cannot be found.
   iii) RESTORE# "object string", {0 / 1}
   0: Same as ii above.
   1: The first data of the record (line) that includes object string is read by the following READ# statement.
   iv) RESTORE# "object string" [, {0 / 1}], {line number / #program area number}
   Execution branches to specified line or program area when the object string is not found.

- Search is conducted from the present pointer position forward to the higher record number. The following procedure is used to search from the beginning of entire data:
  RESTORE# : RESTORE# "object string"

SAMPLE PROGRAM:

```
10 RESTORE# "YOU", 0, 50
20 READ# A$
30 PRINT A$
40 GOTO 10
50 PRINT "NO DATA!"
```

Searches for data beginning with "YOU", and displays "NO DATA!" if not found.

WRITE

Ⓟ

PURPOSE: Rewrites and deletes DATA BANK data.

FORMAT: WRITE # [DATA BANK data / Expression , [DATA BANK data / Expression]]

EXAMPLE; WRITE # "ABCDEF"

PARAMETERS: DATA BANK data: String or numeric expression

EXPLANATION:

1. Sequentially writes DATA BANK data from the current DATA BANK area pointer (see RESTORE #).
2. New data are written regardless of whether or not data already exist at the pointer location.
3. The entire record (line) is cleared when this command is executed without any DATA BANK data expressions.
4. Multiple data items can be delimited using commas.
5. The DATA BANK area pointer stays at the next data item written after execution of this command. Further data item writing begins from this point unless the pointer position is changed using RESTORE #.
6. 255 characters per line can be written using this command, and an error is generated when this limit is exceeded.
7. Numeric expressions written using this command are written using the same format as PRINT statement display. Note, however, that the SET statement has no effect here.
8. This command cannot be used to write character codes &H1F or lower.

SAMPLE PROGRAM:

```
10 RESTORE #
20 RESTORE # "YOU", 0, 50
30 WRITE # "SHE"
40 GOTO 20
50 PRINT "COMPLETE!"
60 END
```

Changes DATA BANK data beginning with "YOU" to "SHE".

PART 11

SCIENTIFIC LIBRARY

11-1 LIBRARY EXECUTION

11-1-1 Activating The Library

The library function of the FX-850P provides a total of 116 different utilities divided into a mathematical library, a statistical library, and physics and scientific library. The two methods described below can be used to activate the desired library in the CAL mode.

1. Library number + [LIB] key

Activation of the library using this method is achieved by first entering a library number and then pressing the [LIB] key.

EXAMPLE

Activation of the library utility for solution of a quadratic equation (Library Number 5050).

```
_
```
(Switch power ON)

5050 [LIB]

```
ax²+bx+c=0
a= 1 ?_
```
(Library number entry)

* The cursor moves to the next line with no further operation when an invalid library number is entered.

One of the following two operations is performed when the [LIB] key is pressed without entry of a library number.

i) Pressing [LIB] immediately after power is switched ON

```
_
```
(Switch power ON)

[LIB]

```
Prime factors        (2≦Base<10¹⁰)
Base ?_
```
(Press [LIB])

This operation activates the prime factor analysis library utility (Library Number 5010).

ii) Pressing [LIB] after execution of a library utility

[BRK]

```
Break
_
```
(Press to break)

[LIB]

```
y=ax+b -- (x1.y1).(x2.y2)
x1= 0 ?_
```
(Press [LIB])

WRITE

Ⓟ

PURPOSE: Rewrites and deletes DATA BANK data.

FORMAT: WRITE# [DATA BANK data / Expression , [DATA BANK data / Expression]]

EXAMPLE; WRITE# "ABCDEF"

PARAMETERS: DATA BANK data: String or numeric expression

EXPLANATION:

1. Sequentially writes DATA BANK data from the current DATA BANK area pointer (see RESTORE#).
2. New data are written regardless of whether or not data already exist at the pointer location.
3. The entire record (line) is cleared when this command is executed without any DATA BANK data expressions.
4. Multiple data items can be delimited using commas.
5. The DATA BANK area pointer stays at the next data item written after execution of this command. Further data item writing begins from this point unless the pointer position is changed using RESTORE#.
6. 255 characters per line can be written using this command, and an error is generated when this limit is exceeded.
7. Numeric expressions written using this command are written using the same format as PRINT statement display. Note, however, that the SET statement has no effect here.
8. This command cannot be used to write character codes &H1F or lower.

SAMPLE PROGRAM:

```
10 RESTORE#
20 RESTORE# "YOU", 0, 50
30 WRITE# "SHE"
40 GOTO 20
50 PRINT "COMPLETE!"
60 END
```

Changes DATA BANK data beginning with "YOU" to "SHE".

PART 11

SCIENTIFIC LIBRARY

11-1 LIBRARY EXECUTION

11-1-1 Activating The Library

The library function of the FX-850P provides a total of 116 different utilities divided into a mathematical library, a statistical library, and physics and scientific library. The two methods described below can be used to activate the desired library in the CAL mode.

1. Library number + [LIB] key

Activation of the library using this method is achieved by first entering a library number and then pressing the [LIB] key.

EXAMPLE

Activation of the library utility for solution of a quadratic equation (Library Number 5050).

Key	Display	
	_	(Switch power ON)
5050 [LIB]	ax²+bx+c=0 a= 1 ?_	(Library number entry)

* The cursor moves to the next line with no further operation when an invalid library number is entered.

One of the following two operations is performed when the [LIB] key is pressed without entry of a library number.

i) Pressing [LIB] immediately after power is switched ON

Key	Display	
	_	(Switch power ON)
[LIB]	Prime factors (2≦Base<10¹⁰) Base ?_	(Press [LIB])

This operation activates the prime factor analysis library utility (Library Number 5010).

ii) Pressing [LIB] after execution of a library utility

Key	Display	
[BRK]	Break _	(Press to break)
[LIB]	y=ax+b -- (x1.y1).(x2.y2) x1= 0 ?_	(Press [LIB])

In this case, the previous library utility (here, Library Number 5510; STRAIGHT LINE PASSING THROUGH TWO POINTS) is reactivated.

- In this example, the [LIB] key was pressed immediately following [BRK]. The same result is produced when manual calculations or a BASIC program is executed following [BRK].

2. Selection using the [MENU] key

Pressing the [MENU] key produces a display of the library utilities built into the FX-850P. The following operations can be used to locate a specific utility.

i) [↑] and [↓] keys are used to respectively scroll up and down through the utility list. Holding either key down results in high speed display change.

[←] displays the first library utility (memory calculation, Library Number 1000).
[→] displays the last library utility (ratio difference test, Library Number 6772).
[MENU] displays following library utility contents.

ii) Pressing either [EXE] or [LIB] when the desired library utility is displayed executes the utility.

EXAMPLE 1

Activation of numerical solution (method of bisection) after power is switched ON.

Key	Display	
	_	(Switch power ON)
[MENU]	5010:Prime factors Base = a x b x c x ...	
[↓] [↓] [↓] [↓] [↓] [↓]	5090:Numerical solution f(x)=0 Method of bisection	(Press [↓] 6 times)
[LIB]	Method of bisection f(x)=0 1:f(x),x0,x1 2:ε,loop	(Execution using [LIB])

EXAMPLE 2

Termination of bisection method in EXAMPLE 1 and activation of quadratic equation solution utility.

Key	Display	
[BRK]	Break _	(Termination of library utility)
[MENU]	5090:Numerical solution f(x)=0 Method of bisection	(Library list display)
[↑] [↑] [↑]	5050:ax²+bx+c=0	(Press [↑] three times)
[EXE]	ax²+bx+c=0 a= 1 ?_	(Activation of library utility)

11-1-2 Library Termination

Execution of a library utility can be terminated by pressing the [BRK] key.

Key	Display	
	ax²+bx+c=0 a= 1 ?_	
[BRK]	Break _	(Termination of library utility)

11-1-3 Library Activation Display

The displays that appear immediately following activation of the library are of two types, and are referred to throughout this manual as follows.

1. Initial display

Display immediately following library activation for value input, YES/NO selection, or list display.

EXAMPLE 1

Immediately following activation of prime factor analysis library utility (Library Number 5010).

```
Prime factors      (2≦Base<10^10)
Base ?_
```

EXAMPLE 2

Immediately following activation of interval estimation library utility (Library Number 6610).

```
N(μ,σ2) a<μ<b   σ2:known
input new data (Y/N) ?
```

EXAMPLE 3

Immediately following activation of formula library utility (Library Number 5800).

```
a2-b2                                [1]
=(a+b)(a-b)
```

2. Menu display

Display immediately following library activation for process selection.

EXAMPLE 1

Immediately following activation of Newton's method library utility (Library Number 5080).

```
Newton's method   f(x)=0
1:f(x),x0         2:h,ε,loop
```

EXAMPLE 2

Immediately following activation of matrix operation library function (Library Number 5100).

```
Matrix  A(2,2):B(2,2)
>A,B,D,I,T,K,+,-,*,M,L,C,P ?_
```

11-1-4 Examples Used in This Manual

The examples shown in this manual are generally presented as being performed immediately following library activation. When the library is activated, certain values (0 or 1) are stored for the variables used within the library. Continuously using the library without a break causes the values which have been entered or calculated to be retained. When inputting data, the values assigned to variables are displayed as shown in the display illustrated below (actual display differs according to the library utility being used).

```
B(1.p)a<p<b
n=0  ?_
```

Ratio interval estimation; Library Number 6670

At this time, the [EXE] key can be pressed without changing the displayed value, or the displayed value can be changed before pressing [EXE].

11-1-5 Precautions When Using the Library

① Library executions can be performed in the CAL mode only.
② A number of different types of variables are used in library calculations. Using a large number of variables in various library utilities may cause library execution speed to decrease. Speed can be increased, in this case, by executing the CLEAR statement before activating the library function. *It should be noted, however, that the CLEAR statement clears all variables currently stored in memory.*
③ Activation of the library automatically switches the PRINT mode OFF and executes the DIM command. This means that a PRINT mode ON specification or DEFM command executed before the library is activated is canceled.
④ Numeric values used during library executions should have mantissas of 10 digits or fewer.
⑤ Library variable names consist of single lower case alphabetic characters (a ~ z). Statistical variable names in the library are preceded by the letter "s" (sa ~ sz).
⑥ All library utilities are created using the BASIC language.
⑦ The [LB] symbol is shown on the display during library execution and list display.
⑧ [EXE] can be pressed while the previously entered data are displayed during data input to enter the displayed data again.

EXAMPLE

```
test     Ho:p=po      H:p≠po
po  =0.02  ?_
```

Pressing [EXE] here inputs 0.02 as the value for Po.

Execution of certain library utilities automatically switches to the lower case mode or the RAD angle unit (from DEG) mode. Since pressing the [BRK] key terminates execution while retaining the lower case or angle unit mode, the mode automatically switched to should be manually changed as required before execution of another library utility or calculation.

- Library utilities automatically switching to lower case mode

5080, 5090, 5200, 5220

- Library utilities automatically switching angle unit to RAD

5080, 5090, 5200, 5220, 5625, 6230, 6240, 6430, 6440, 6450, 6620, 6650, 6660, 6720, 6721, 6722, 6740, 6741, 6742, 6750, 6751, 6752

1000 MEMORY CALCULATIONS

This function makes it possible to use the cursor keys to perform the four key memory (MC, MR, M−, M+) operations.

The following list shows the corresponding memory operation that corresponds to each cursor key.

[⇧] : MC (Memory Clear) Clears data stored in memory
[⇩] : MR (Memory Recall) Recalls data stored in memory
[⇦] : M− (Memory minus) Subtracts from memory
[⇨] : M+ (Memory plus) Adds to memory

Both the calculation result and memory contents are simultaneously shown at the bottom of the display.

```
MC[↑]    MR[↓]    M-[←]    M+[→]
0                 0
```

16-digit calculation result | 16-digit memory contents

Values can be corrected using the [BS] key (1-character delete) or [SHIFT] [CLS] (all value clear).

Besides the four basic arithmetic functions, numeric scientific function, logical operation, and comparison calculations can all be performed. One-key commands, however, cannot be used for numeric scientific function calculations, and direct function keys cannot be used.

EXAMPLE

[sin] [3] [0] [EXE] cannot be used to enter sin30°. It mused be entered as [S] [I] [N] [3] [0] [EXE].

The formula memory is used for memory calculations. Therefore, it should be noted that contents of the formula memory are changed when memory calculations are performed.

OPERATION

1000 [LIB]

```
MC[↑]    MR[↓]    M-[←]    M+[→]
0                 0
```

EXAMPLE 1

Perform the calculation: 15÷3+7−6=6

```
MC[↑]    MR[↓]    M-[←]    M+[→]
0                 0
```

15 [/] 3 [+] 7 [−] 6

```
MC[↑]    MR[↓]    M-[←]    M+[→]
15/3+7-6_
```
(Formula input)

[EXE]

```
MC[↑]    MR[↓]    M-[←]    M+[→]
6                 0
```
(Formula execution)

EXAMPLE 2

Perform the following calculations: 120 × 1.4 = 168
1.4 × 170 = 238

Key operation	Display	
[⇧]	MC[↑] MR[↓] M-[-] M+[-] 0 0	(Memory clear)
1.4 [⇨]	MC[↑] MR[↓] M-[-] M+[-] 1.4 1.4	(Storage of 1.4 in memory)
120 [×] [⇩]	MC[↑] MR[↓] M-[-] M+[-] 120×1.4_	(Formula input)
[EXE]	MC[↑] MR[↓] M-[-] M+[-] 168 1.4	(Formula execution)
[⇩] [×] 170	MC[↑] MR[↓] M-[-] M+[-] 1.4×170_	(Recall of 1.4 from memory)
[EXE]	MC[↑] MR[↓] M-[-] M+[-] 238 1.4	(Formula execution)

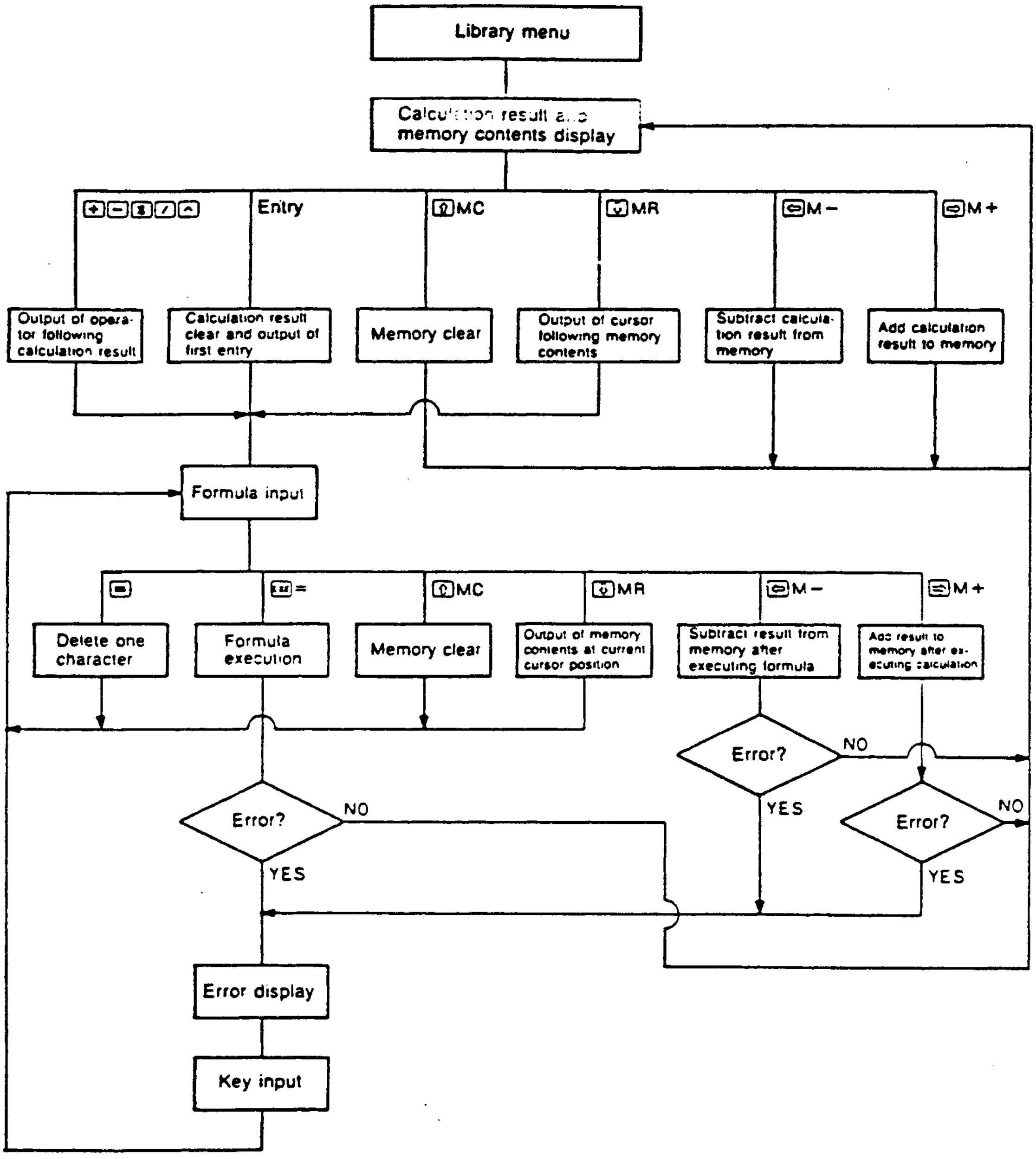
EXAMPLE 3

Perform the following calculation: 3 + 7 + sin30° (angle unit = degree)

Key operation	Display	
[⇧]	MC[↑] MR[↓] M-[-] M+[-] 0 0	(Memory clear)
3 [⇨]	MC[↑] MR[↓] M-[-] M+[-] 3 3	(Storage of value in memory)
7 [⇨]	MC[↑] MR[↓] M-[-] M+[-] 7 10	(Add to memory)
SIN30 [⇨]	MC[↑] MR[↓] M-[-] M+[-] 0.5 10.5	(Add to memory following function calculation)

Set the mode for the desired angle unit (DEG, RAD, GRA) before activating the library.

MEMORY CALCULATION FLOWCHART



5010 PRIME FACTOR ANALYSIS

Performs prime factor analysis on an input value base. The input range of entered value a is an integer within the range of $2 \leq a < 10^{10}$. The analysis is performed by first determining if the value input for a is divisible by 2 or by b, which is assigned sequential odd numbers (3, 5, 7...).
When b is a prime factor, the formula $ai = \frac{ai - 4}{b}$ is applied and division is repeated until $\sqrt{ai} + 1 \leq b$.

OPERATION

5010 [LIB]

```
Prime factors      (2≦Base<10^10)
Base ?_
```

EXAMPLE

Perform prime factor analysis for a base of 100.

```
Prime factors      (2≦Base<10^10)
Base ?_
```

100

```
Prime factors      (2≦Base<10^10)
Base ?100_
```

(Base input)

[EXE]

```
Prime factors      (2≦Base<10^10)
 100 = .....
```

(Calculation)

```
Prime factors      (2≦Base<10^10)
 100 = 2^2 * 5^2
```

(Result display)

[EXE]

```
Prime factors      (2≦Base<10^10)
Base ?_
```

(Return to initial display)

Here, the result of the prime factor analysis is $100 = 2^2 \times 5^2$.

5020 GREATEST COMMON MEASURE/ LEAST COMMON MULTIPLE

Determines the greatest common measure (GCM) and least common multiple (LCM) for two entered integers (a, b), within the range of $1 \leqq a < 10^{10}$, $1 \leqq b < 10^{10}$. The GCM and LCM are determined using the Euclidean method.

OPERATION

5020 [LIB]

```
G.C.M. & L.C.M.    (1≦a.b<10¹⁰)
a= 1 ?_
```

EXAMPLE

Determine the GCM and LCM when a=5 and b=2.

```
G.C.M. & L.C.M.    (1≦a.b<10¹⁰)
a= 1 ?_
```

5 [EXE]

```
G.C.M. & L.C.M.    (1≦a.b<10¹⁰)
b= 1 ?_
```

(Value a input)

2 [EXE]

```
G.C.M.= 1
L.C.M.= 10
```

(Value b input)

[EXE]

```
G.C.M. & L.C.M.    (1≦a.b<10¹⁰)
a= 5 ?_
```

(Return to initial display)

Here, the GCM of 2 and 5 is 1, while the LCM is 10.

5040 SIMULTANEOUS EQUATIONS (GAUSS-JORDAN ELIMINATION)

Solves for $x_1 \sim x_n$ in the following n simultaneous equations ($2 \leqq n \leqq 7$) for input of coefficients $a_1 \sim a_n$, $b_1 \sim b_n$ ··· and $y_1 \sim y_n$.

$a_1 \cdot x_1 + b_1 \cdot x_2 + c_1 \cdot x_3 + \cdots = y_1$
$a_2 \cdot x_2 + b_2 \cdot x_2 + c_2 \cdot x_3 + \cdots = y_2$
$\vdots$
$a_n \cdot x_n + b_n \cdot x_n + c_n \cdot x_n + \cdots = y_n$

Solutions may not be exact for coefficients with a difference in excess of 1×10^{10} due to internal rounding.

OPERATION

5040 [LIB]

```
ax1+bx2+cx3+···=y      (2≦n≦7)
n= 2 ?_
```

Pressing [SHIFT] [EXE] during coefficient input returns to the previous coefficient entry.
Pressing [⇩] or [EXE] during display of a solution scrolls to the following solution, while [⇧] scrolls to the previous solution.
The message "not found" appears on the display when a solution cannot be found.

EXAMPLE

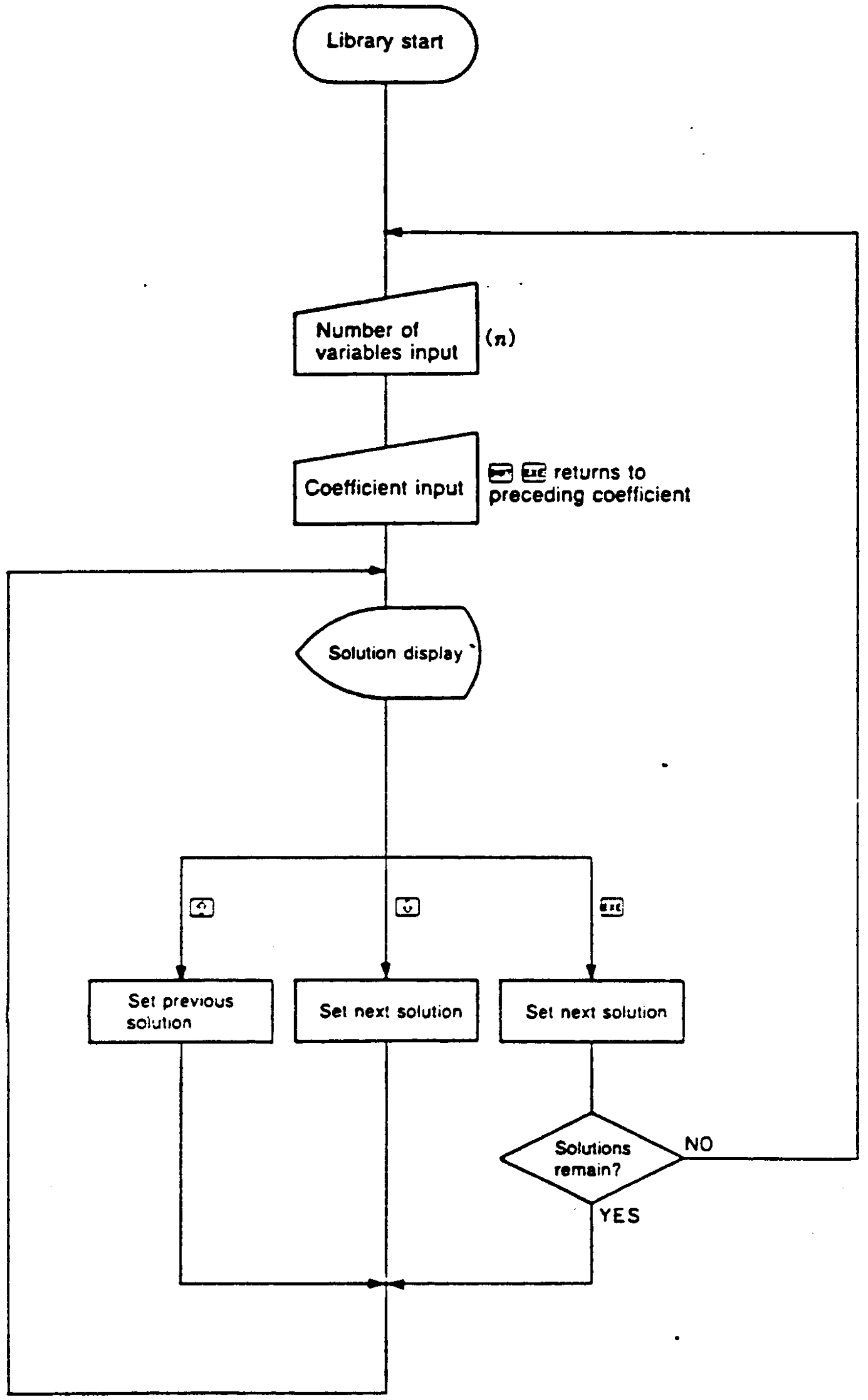
Solve the following simultaneous cubic equations for x_1, x_2, and x_3.

$2x_1 + 3x_2 - x_3 = 15$
$3x_1 - 2x_2 + 2x_3 = 4$
$5x_1 + 3x_2 - 4x_3 = 9$

	ax1+bx2+cx3+···=y (2≦n≦7) n= 2 ?_	
3 EXE	ax1+bx2+cx3=y 1:a= 0 ?_	(Input 3 to specify cubic equations)
2 EXE	ax1+bx2+cx3=y 1:b= 0 ?_	(Input coefficients for first equation)
3 EXE	ax1+bx2+cx3=y 1:c= 0 ?_	(Input coefficients)
−1 EXE	ax1+bx2+cx3=y 1:y= 0 ?_	
15 EXE	ax1+bx2+cx3=y 2:a= 0 ?_	
3 EXE −2 EXE 2 EXE 4 EXE	ax1+bx2+cx3=y 3:a= 0 ?_	(Input coefficients for second equation)
5 EXE 3 EXE −4 EXE 9 EXE	ax1+bx2+cx3=y x1 = ...	(Input coefficients for third equation)
	ax1+bx2+cx3=y x1 = 2	(Display value for x_1)
EXE	ax1+bx2+cx3=y x2 = 5	(Display value for x_2)
EXE	ax1+bx2+cx3=y x3 = 4	(Display value for x_3)
EXE	ax1+bx2+cx3+···=y (2≦n≦7) n= 3 ?_	(Return to initial display)

Here, the solutions of the simultaneous equations are $x_1 = 2$, $x_2 = 5$, $x_3 = 4$.

SIMULTANEOUS EQUATION FLOWCHART



5050 QUADRATIC EQUATION

Determines the solution for α and β when coefficients a, b, and c are input for the quadratic equation $ax^2+bx+c=0$.
Root equations are used to determine the solution.

Root equation: $x=\frac{-b\pm\sqrt{b^2-4ac}}{2a}$

When $d=b^2-4ac$:

i) When $d>0$, two real roots (α, β) are present: $\alpha=\frac{-b+\sqrt{d}}{2a}$, $\beta=\frac{-b-\sqrt{d}}{2a}$

ii) When $d=0$, one real root (α) is present: $\alpha=\frac{-b}{2a}$ (multiple root)

iii) When $d<0$, two imaginary roots (α, β) are present:

$$\alpha=\frac{-b}{2a}+\frac{\sqrt{-d}}{2a}i,\ \beta=\frac{-b}{2a}-\frac{\sqrt{-d}}{2a}i$$

OPERATION

5050 [LIB]

```
ax²+bx+c=0
a= 1 ?_
```

EXAMPLE

Determine the solution of the following quadratic equation:
$2x^2-5x+3=0$

Key operation	Display	
	ax²+bx+c=0 / a= 1 ?_	
2 [EXE]	ax²+bx+c=0 / b= 0 ?_	(Coefficient a input)
−5 [EXE]	ax²+bx+c=0 / c= 0 ?_	(Coefficient b input)
3 [EXE]	ax²+bx+c=0 /	(Coefficient c input)
	a(x−α)(x−β)=0 / α = 1.5	(Solution α display)
[EXE]	a(x−α)(x−β)=0 / β = 1	(Solution β display)
[EXE]	ax²+bx+c=0 / a= 2 ?_	(Return to initial display)

Here, the solutions of $2x^2-5x+3=0$ are $\alpha=1.5$, $\beta=1$.

SOLUTION DISPLAY

Pressing [EXE] or [⇩] scrolls from display of α to β (only α displayed for multiple root). Pressing [⇧] while β is displayed returns to the display of α.

5060 CUBIC EQUATIONS

Determines the solution for α, β and γ when coefficients a, b, c, and d are input for the cubic equation $ax^3+bx^2+cx+d=0$.
Root equations are used to determine the solution.
Transformation to $y^3+3py+q=0$ can be performed
when $x=y-\frac{b}{3a}$, $p=\frac{c}{3a}-\frac{b^2}{9a^2}$, $q=\frac{2b^3}{27a^3}-\frac{bc}{3a^2}+\frac{b}{a}$ are substituted in $ax^3+bx^2+cx=0$.
Here, substituting $A=\sqrt[3]{\frac{q+\sqrt{c}}{2}}$, $B=\sqrt[3]{\frac{q-\sqrt{c}}{2}}$, $c=q^2+4p^3$ results in the following:

i) When $c>0$, one real root (α), and two imaginary roots (β, γ) are present:

$$\alpha=-(A+B),\ \beta=\frac{A+B}{2}+\frac{\sqrt{3}}{2}(A-B)i,\ \gamma=\frac{A+B}{2}-\frac{\sqrt{3}}{2}(A-B)i$$

ii) When $c=0$, $p=0$, one real root (α) is present: $\alpha=-(A+B)$

iii) When $c=0$, $p\neq0$, two real roots (α, β) are present:

$$\alpha=-(A+B),\ \beta=\frac{A+B}{2} \quad \text{(multiple roots)}$$

iv) When $c<0$, three real roots (α, β, γ) are present:

$$\alpha=-2\sqrt{-p}\cos\theta,\ \beta=-2\sqrt{-p}\cos(\theta+120°),$$
$$\gamma=-2\sqrt{-p}\cos(\theta+240°) \qquad \theta=\frac{1}{3}\cos^{-1}\frac{q}{2\sqrt{-p^3}}$$

OPERATION

5060 [LIB]

```
ax³+bx²+cx+d=0
a=  1  ?_
```

EXAMPLE

Determine the solution of the following cubic equation:
$2x^3+x^2-13x+6=0$

Key	Display	
	ax³+bx²+cx+d=0 a= 1 ?_	
2 [EXE]	ax³+bx²+cx+d=0 b= 0 ?_	(Coefficient a input)
1 [EXE]	ax³+bx²+cx+d=0 c= 0 ?_	(Coefficient b input)
−13 [EXE]	ax³+bx²+cx+d=0 d= 0 ?_	(Coefficient c input)
6 [EXE]	ax³+bx²+cx+d=0	(Coefficient d input)
	a(x−α)(x−β)(x−γ)=0 α =−3	(Solution α display)
[EXE]	a(x−α)(x−β)(x−γ)=0 β = 2	(Solution β display)

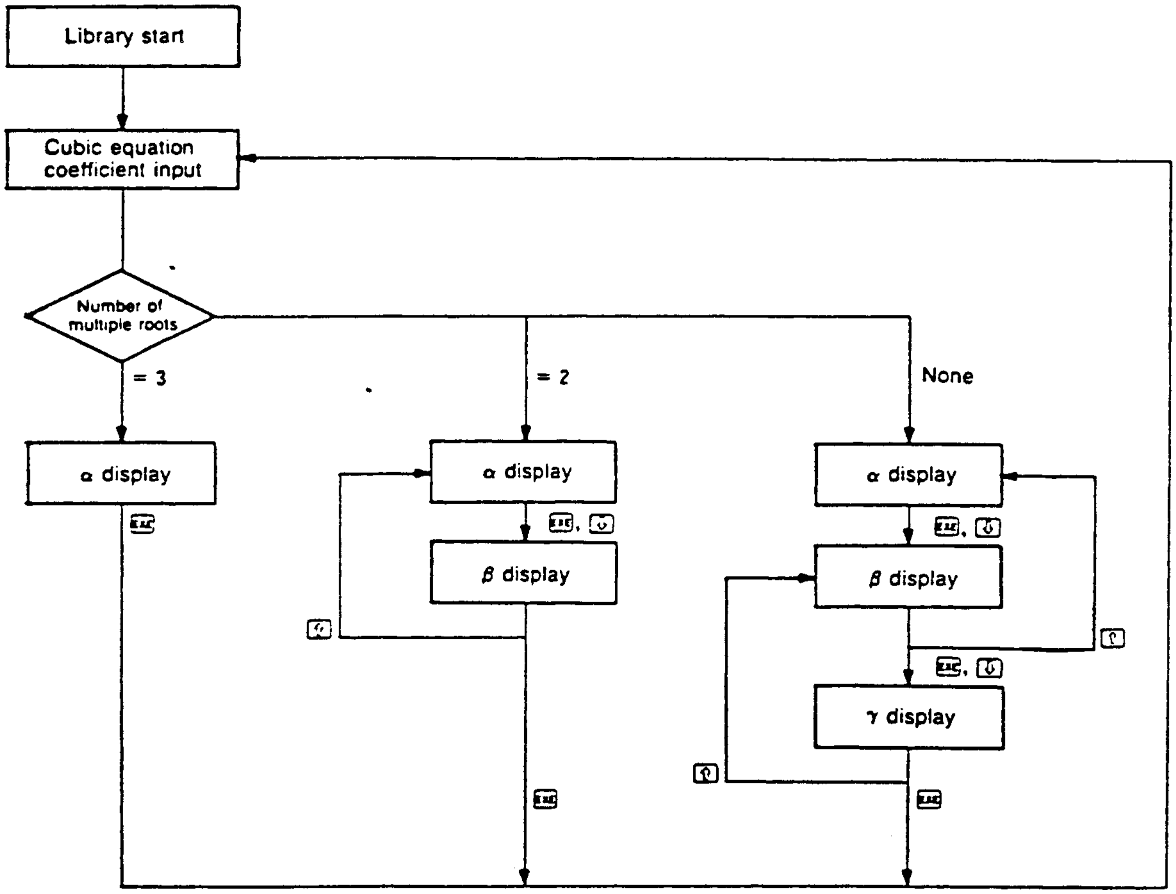
Key	Display	
[EXE]	a (x − α) (x − β) (x − γ) = 0 γ = 0.5	(Solution γ display)
[EXE]	a x ³ + b x ² + c x + d = 0 a = 2 ?_	(Return to initial display)

Here, the solutions of $2x^3 + x^2 - 13x + 6 = 0$ are $\alpha = -3$, $\beta = 2$, $\gamma = 0.5$.

SOLUTION DISPLAY

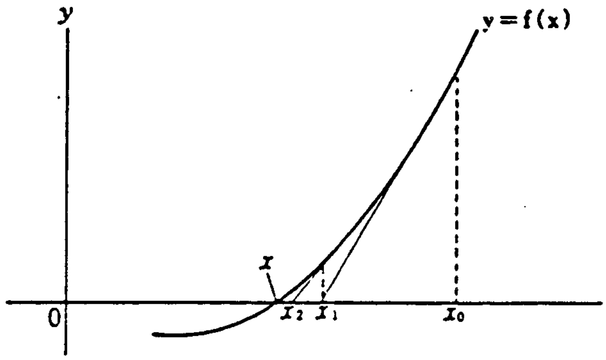
Pressing [EXE] or [⇩] displays α, β and γ in sequence. Pressing [⇧] while β or γ is displayed returns to the display of α or β. Only α or α and β are displayed in the case of multiple roots.

CUBIC EQUATION FLOWCHART



5080 NUMERIC SOLUTION OF AN EQUATION (NEWTON'S METHOD)

Determines the solution of the function y = f(x) graphed below for f(x) = 0, using Newton's Method.



(Angle unit = radians)

The following parameters are specified in order to determine the numeric solution using Newton's Method.

x_0 : Initial value

h : Minute interval for x-axis when performing numerical differentiation at point (x, f(x))

$$f'(x) = \frac{f(x+h) - f(x)}{h}$$

ϵ : Solution convergence ($\epsilon > |x_{n+1} - x_n|$: continuously calculate and return value of ϵ as long as inequality is true)

loop : Maximum number of convergences (positive integer)

- The following arithmetic operators and functions can be applied here:
  +, −, *, /, ^ , SIN, COS, TAN, ASN, ACS, ATN, LOG, LN, EXP, SQR, HYP
- The variable name for the function f(x) must be represented by x.
- The value input for ϵ must be 1E−90 or more. Since internal calculations are performed in 12 digits, smaller values have little meaning.

OPERATION

5080 [LIB]

```
Newton's method  f(x)=0
1:f(x),x0       2:h,ε,loop
```

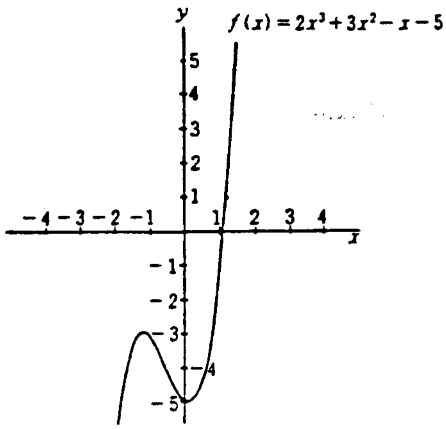
The menu display illustrated above appears when the library is activated. Either 1 or 2 should be selected in accordance with the type of processing to be performed.

1 : Function f(x) specification/initial value input

2 : Input of minute interval, convergence condition, and maximum number of convergences

EXAMPLE

Determine the f(x) = 0 solution of the following equation for $f(x) = 2x^3 + 3x^2 - x - 5$, where the minute interval is 0.00001, the convergence condition is 0.0001, and the maximum number of convergences is 30.



Key operation	Display	
	Newton's method f(x)=0 1:f(x).x0 2:h.ε.loop	
2	f'(x)=(f(x+h)-f(x))/h (h>0) h= 0.00001 ?_	(Parameter input selection)
0.00001 EXE	Err |Xn+1-Xn|<ε (ε>0) ε= 0.0000001 ?_	(Minute interval input)
0.0001 EXE	Max loop (n>0) n= 20 ?_	(Convergence condition input)
30 EXE	Newton's method f(x)=0 1:f(x).x0 2:h.ε.loop	(Maximum number of convergences input)
1	Define function f(x) ?_	(Function/initial value input selection)
2*x^3+3*x^2-x-5 EXE	f(x) = 2*x^3+3*x^2-x-5 x0= 0 ?_	(Function input)
1 EXE	f(x) = 2*x^3+3*x^2-x-5	(Initial value input)
	f(x) = 2*x^3+3*x^2-x-5 x = 1.0849	(Solution display)
EXE	Newton's method f(x)=0 1:f(x).x0 2:h.ε.loop	(Return to initial display)

This display indicates that the solution for the example equation is 1.0849.

The message "not found" is displayed when a solution cannot be found.

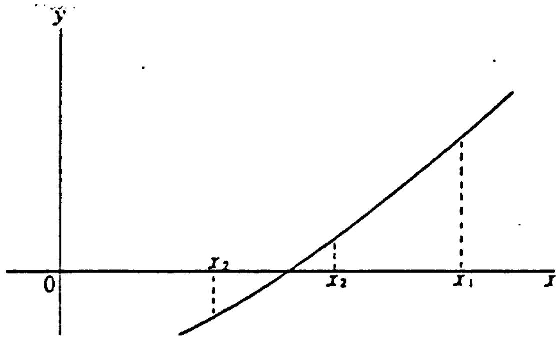
```
f(x) = x^2+1
not found
```

Pressing EXE at this time returns the display to point at which calculation was discontinued. Pressing EXE again returns to menu for numeric solution of an equation (certain calcualtions may not initially display discontinued point display).

```
f(x) = x^2+1
loop = 20 :Xn = 1.070479459
```

5090 NUMERIC SULOTION OF AN EQUATION
(BISECTION METHOD)

Determines the solution of the function y = f(x) graphed below for f(x) = 0, using the bisection method.



(Angle unit = radians)

The following parameters are specified in order to determine the numeric solution using the bisection method.

x_0, x_1 : Initial value
ϵ : Solution convergence ($\epsilon > |x_{n+1} - x_n|$: continuously calculate and return value of ϵ as long as inequality is true)
loop : Maximum number of convergences (positive integer)

- The following arithmetic operators and functions can be applied here:
  +, −, *, /, ^ , SIN, COS, TAN, ASN, ACS, ATN, LOG, LN, EXP, SQR, HYP
- The variable name for the function f(x) must be represented by x.
- The value input for ϵ must be 1E−90 or more. Since internal calculations are performed in 12 digits, smaller values have little meaning.

OPERATION

5090 [LIB]

```
Method of bisection  f(x)=0
1:f(x).x0.x1     2:ε.loop
```

The menu display illustrated above appears when the library is activated. Either 1 or 2 should be selected in accordance with the type of processing to be performed.

1 : Function f(x) specification/initial value input
2 : Input of convergence conditin and maximum number of convergences

EXAMPLE

Determine the $f(x)=0$ solution of the following equation for $f(x)=2x^3-x^2-8x-11$, where the convergence condition is 0.0001, the maximum number of convergences is 40, and initial values are $x_0=-5$, $x_1=5$.

Key operation	Display	Comment
	Method of bisection f(x)=0 1:f(x).x0.x1 2:ε.loop	
2	Err |Xn+1−Xn|<ε (ε>0) ε= 0.0000001	(Parameter input selection)
0.0001 EXE	Max loop (n>0) n= 30 ?_	(Convergence condition input)
40 EXE	Method of bisection f(x)=0 1:f(x).x0.x1 2:ε.loop	(Maximum number of convergence input)
1	Define function f(x) ?_	(Function/initial value input selection)
2*x ^ 3 − x ^ 2 − 8*x − 11 EXE	f(x) = 2*x^3−x^2−8*x−11 x0= 0 ?_	(Function input)
−5 EXE	f(x) = 2*x^3−x^2−8*x−11 x1= 0 ?_	(Initial value x_0 input)
5 EXE	f(x) = 2*x^3−x^2−8*x−11 x = ...	(Initial value x_1 input)
	f(x) = 2*x^3−x^2−8*x−11 x = 2.7171	(Solution display)
EXE	Method of bisection f(x)=0 1:f(x).x0.x1 2:ε.loop	(Return to initial display)

This display indicates that the solution for the example equation is 2.7171.

The message "not found" is displayed when a solution cannot be found.

f(x) = x^2+1 not found

5100 MATRIX OPERATIONS

Matrix operations make it possible to perform addition, subtraction, multiplication, scalar product, determinant, inverse matrix, and transposed matrix calculations.

OPERATION

5100 [LIB]

```
Matrix  A(2.2):B(2.2)
>A.B.D.I.T.K.+.-.*.M.L.C.P ?_
```

The following process can be selected from the menu display illustrated above.

A	:	Definition of MATRIX A and data input
B	:	Definition of MATRIX B and data input
D	:	Determinant of MATRIX A (det A)
I	:	Inverse matrix of MATRIX A and assignment of result to MATRIX A ($A^{-1} \rightarrow A$)
T	:	Transposed matrix of MATRIX A and assignment of result to MATRIX A ($A^t \rightarrow A$)
K	:	Scalar product of MATRIX A and assignment of result to MATRIX A ($kA \rightarrow A$)
+	:	Addition of MATRIX A and MATRIX B and assignment of result to MATRIX A ($A+B \rightarrow A$)
−	:	Subtraction of MATRIX A and MATRIX B and assignment of result to MATRIX A ($A-B \rightarrow A$)
*	:	Multiplication of MATRIX A and MATRIX B and assignment of result to MATRIX A ($A \cdot B \rightarrow A$)
M	:	Assignment of MATRIX A contents to MEMORY MATRIX M ($A \rightarrow M$)
L	:	Assignment of MEMORY MATRIX M contents to MATRIX A ($M \rightarrow A$)
C	:	Exchange of MATRIX A and MATRIX B contents ($A \leftrightarrow B$)
P	:	Display of MATRIX A contents
·	:	Help display

MATRIX SET UP

Select either [A] (MATRIX A) or [B] (MATRIX B) from the menu display for matrix set up.

EXAMPLE 1

Set up the 3-row by 4-column matrix shown to the right.

Column (n) →

Row (m) ↓

$$\begin{bmatrix} 1 & 0 & 3 & 4 \\ 2 & 1 & 0 & -1 \\ 3 & 1 & -2 & 3 \end{bmatrix}$$

Key	Display	
	`Matrix  A(2.2):B(2.2)` / `>A.B.D.I.T.K.+.-.*.M.L.C.P ?_`	
A	`A(m.n) = A(2.2)` / `m ?_`	(MATRIX A specification)
3 [EXE]	`A(m.n) = A(3.2)` / `n ?_`	(Row input)
4 [EXE]	`a(1.1) = 0` / `?_`	(Column input)

* A 2-row by 2-column $\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$ matrix is automatically set up when this library is activated.

Now enter the elements in the sequence shown in the illustration to the right (①–⑫).

$$\begin{bmatrix} \underset{①}{1} & \underset{②}{0} & \underset{③}{3} & \underset{④}{4} \\ \underset{⑤}{2} & \underset{⑥}{1} & \underset{⑦}{0} & \underset{⑧}{-1} \\ \underset{⑨}{3} & \underset{⑩}{1} & \underset{⑪}{-2} & \underset{⑫}{-3} \end{bmatrix}$$

Key	Display	
	a(1.1)=0 ?_	
1 EXE	a(1.2) = 0 ?_	(Matrix element entries)
0 EXE	a(1.3) = 0 ?_	

The unit returns to the menu display once input of all of the elements is complete. At this point, it is advisable to review the values to confirm that input was performed correctly.

Key	Display	
A	A(m.n) = A(3.4) m ?_	(MATRIX A specification)
EXE	A(m.n) = A(3.4) n ?_	(Press EXE after confirmation)
EXE	a(1.1) = 1 ?_	
EXE	a(1.2) = 0 ?_	
⋮	⋮	
EXE	a(3.3) =-2 ?_	
EXE	a(3.4) =-3 ?_	
EXE	Matrix A(3.4):B(2.2) >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	(Return to initial display)

CORRECTION

Errors discovered before the EXE key is pressed can be corrected by simply entering the correct value and then pressing EXE. After EXE is pressed, press ⇦ EXE to return to the previous value display and then make necessary corrections.

* The P command can also be used to view matrix contents.

Matrix addition/subtraction/multiplication

EXAMPLE 2

$$A = \begin{bmatrix} 1 & 1 \\ 2 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix}$$

Perform A+B, A−B, A·B, and B·A for the two following matrices.

Perform the following operation from the menu display.

```
Matrix  A(2.2):B(2.2)
>A.B.D.I.T.K.+.-.x.M.L.C.P ?_
```

A	A(m.n) = A(2.2) m ?_	(MATRIX A specification)
2 EXE 2 EXE	a(1.1) = 0 ?_	(2-row/2-column specification)
1 EXE 1 EXE 2 EXE 1 EXE		
	Matrix A(2.2):B(2.2) >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	(Element input)
B	B(m.n) = B(2.2) m?_	(MATRIX B specification)
2 EXE 2 EXE	b(1.1) = 0 ?_	(2-row/2-column specification)
2 EXE 3 EXE 2 EXE 1 EXE		
	Matrix A(2.2):B(2.2) >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	(Element input)
M	Memory A → M(2.2) >A.B.D.I.T.K.+.-.x.M.L.C.P ?	(Transfer of MATRIX A to MATRIX M)
	Matrix A(2.2):B(2.2) >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	

The results of most matrix operations are stored in MATRIX A, deleting any contents currently stored in MATRIX A. Therefore, it is advisable to first transfer the contents from MATRIX A to MATRIX M so they can be recalled if later required before performing a matrix operation.

Once matrix set up is complete, proceed with the following calculations.

• A+B

+	A+B → A(2.2) >A.B.D.I.T.K.+.-.x.M.L.C.P ?	(Addition specification)
	a(1.1) = 3 >A.B.D.I.T.K.+.-.x.M.L.C.P ?	(Result display)
EXE	a(1.2) = 4 >A.B.D.I.T.K.+.-.x.M.L.C.P ?	(Press EXE after confirmation)
EXE	a(2.1) = 4 >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	
EXE	a(2.2) = 2 >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	
EXE	Matrix A(2.2):B(2.2) >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	(Return to initial display)

Here, the result of A+B is $\begin{bmatrix} 3 & 4 \\ 4 & 2 \end{bmatrix}$.

• A−B

L	Load A ← M(2.2) >A.B.D.I.T.K.+.-.x.M.L.C.P ?	(Transfer of MATRIX M elements to MATRIX A)
	Matrix A(2.2):B(2.2) >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	
−	A−B → A(2.2) >A.B.D.I.T.K.+.-.x.M.L.C.P ?	(Subtraction specification)
	a(1.1) =-1 >A.B.D.I.T.K.+.-.x.M.L.C.P ?	(Result display)
EXE	a(1.2) =-2 >A.B.D.I.T.K.+.-.x.M.L.C.P ?	(Press EXE after confirmation)
EXE	a(2.1) = 0 >A.B.D.I.T.K.+.-.x.M.L.C.P ?	
EXE	a(2.2) = 0 >A.B.D.I.T.K.+.-.x.M.L.C.P ?	

Key	Display	Notes
EXE	Matrix A(2.2):B(2.2) >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	(Return to initial display)

Here, the result of A − B is $\begin{bmatrix} -1 & -2 \\ 0 & 0 \end{bmatrix}$.

• A•B

Key	Display	Notes
L	Load A − M(2.2) >A.B.D.I.T.K.+.-.x.M.L.C.P ?	
	Matrix A(2.2):B(2.2) >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	
*	AxB − A(2.2) >A.B.D.I.T.K.+.-.x.M.L.C.P ?	(Multiplication specification)
	a(1.1) = 4 >A.B.D.I.T.K.+.-.x.M.L.C.P ?	(Result display)
EXE	a(1.2) = 4 >A.B.D.I.T.K.+.-.x.M.L.C.P ?	(Press EXE after confirmation)
EXE	a(2.1) = 6 >A.B.D.I.T.K.+.-.x.M.L.C.P ?	
EXE	a(2.2) = 7 >A.B.D.I.T.K.+.-.x.M.L.C.P ?	
EXE	Matrix A(2.2):B(2.2) >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	(Return to initial display)

Here, the result of A•B is $\begin{bmatrix} 4 & 4 \\ 6 & 7 \end{bmatrix}$.

• B•A

Key	Display	Notes
L	Load A − M(2.2) >A.B.D.I.T.K.+.-.x.M.L.C.P ?	
	Matrix A(2.2):B(2.2) >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	(Transfer of MATRIX M elements to MATRIX A)
C	Change A(2.2) -- B(2.2) >A.B.D.I.T.K.+.-.x.M.L.C.P ?	(Exchange of MATRIX A and MATRIX B elements)
	Matrix A(2.2):B(2.2) >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	
*	AxB − A(2.2) >A.B.D.I.T.K.+.-.x.M.L.C.P ?	(Multiplication specification)
	a(1.1) = 8 >A.B.D.I.T.K.+.-.x.M.L.C.P ?	(Result display)
EXE	a(1.2) = 5 >A.B.D.I.T.K.+.-.x.M.L.C.P ?	(Press EXE after confirmation)
EXE	a(2.1) = 4 >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	
EXE	a(2.2) = 3 >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	
EXE	Matrix A(2.2):B(2.2) >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	(Return to initial display)

Here, the result of B•A is $\begin{bmatrix} 8 & 5 \\ 4 & 3 \end{bmatrix}$.

EXAMPLE 3

Calculate the determinant for the following matrix.

$$\begin{bmatrix} 2 & 1 \\ 0 & -1 \\ 1 & 3 \end{bmatrix} \begin{bmatrix} 3 & -1 & 1 \\ 0 & 2 & 1 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 1 \\ 2 & -3 & 0 \\ 0 & 0 & 2 \end{bmatrix}$$

First perform the multiplication in the first term by setting up the following matrices and then executing A• B.

$$A = \begin{bmatrix} 2 & 1 \\ 0 & -1 \\ 1 & 3 \end{bmatrix}, \quad B = \begin{bmatrix} 3 & -1 & 1 \\ 0 & 2 & 1 \end{bmatrix}$$

Key operation	Display	Remarks
	Matrix A(2.2).B(2.2) >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	
A3 EXE 2 EXE	a(1.1) = 0 ?_	(MATRIX A set up)
2 EXE 1 EXE 0 EXE −1 EXE 1 EXE 3 EXE	Matrix A(3.2):B(2.2) >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	(Element input)
B2 EXE 3 EXE	b(1.1) = 0 ?_	(MATRIX B set up)
3 EXE −1 EXE 1 EXE 0 EXE 2 EXE 1 EXE	Matrix A(3.2):B(2.3) >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	(Element input)
*	AxB - A(3.3) >A.B.D.I.T.K.+.-.x.M.L.C.P ?	(A• B calculation)
	a(1.1) = 6 >A.B.D.I.T.K.+.-.x.M.L.C.P ?	(Result display)

Next perform calculation for second term.

Key operation	Display	Remarks
B3 EXE 3 EXE	b(1.1) = 0 ?_	(MATRIX B set up)
1 EXE 0 EXE 1 EXE 2 EXE −3 EXE 0 EXE 0 EXE 0 EXE 2 EXE	Matrix A(3.3):B(3.3) >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	(Element input)
+	A+B - A(3.3) >A.B.D.I.T.K.+.-.x.M.L.C.P ?	
	a(1.1) = 7 >A.B.D.I.T.K.+.-.x.M.L.C.P ?	
EXE	a(1.2) = 0 >A.B.D.I.T.K.+.-.x.M.L.C.P ?	(Press EXE after confirmation)
EXE	a(1.3) = 4 >A.B.D.I.T.K.+.-.x.M.L.C.P ?	
EXE	a(2.1) = 2 >A.B.D.I.T.K.+.-.x.M.L.C.P ?	
EXE	a(2.2) =-5 >A.B.D.I.T.K.+.-.x.M.L.C.P ?	
EXE	a(2.3) =-1 >A.B.D.I.T.K.+.-.x.M.L.C.P ?	

EXE	`a(3.1) = 3` `>A.B.D.I.T.K.+.-.×.M.L.C.P ?`	
EXE	`a(3.2) = 5` `>A.B.D.I.T.K.+.-.×.M.L.C.P ?`	
EXE	`a(3.3) = 6` `>A.B.D.I.T.K.+.-.×.M.L.C.P ?`	
EXE	`Matrix A(3.3):B(3.3)` `>A.B.D.I.T.K.+.-.×.M.L.C.P ?_`	(Return to initial display)

Here, the result of the calculation is

$$\begin{bmatrix} 2 & 1 \\ 0 & -1 \\ 1 & 3 \end{bmatrix} \begin{bmatrix} 3 & -1 & 1 \\ 0 & 2 & 1 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 1 \\ 2 & -3 & 0 \\ 0 & 0 & 2 \end{bmatrix} = \begin{bmatrix} 7 & 0 & 4 \\ 2 & -5 & -1 \\ 3 & 5 & 6 \end{bmatrix}$$

Determinant, inverse matrix, and transposed matrix

EXAMPLE 4

Determine the determinant, inverse matrix and transposed matrix for the following 3-column/3-row matrix.

$$\begin{bmatrix} 2 & 0 & 0 \\ 3 & 1 & 2 \\ 4 & 2 & 3 \end{bmatrix}$$

	`Matrix A(2.2):B(2.2)` `>A.B.D.I.T.K.+.-.×.M.L.C.P ?_`	
A3 EXE 3 EXE	`a(1.1) = 0` `?_`	(MATRIX A set up)
2 EXE 0 EXE 0 EXE 3 EXE 1 EXE 2 EXE 4 EXE 2 EXE 3 EXE	`Matrix A(3.3):B(2.2)` `>A.B.D.I.T.K.+.-.×.M.L.C.P ?_`	(Element input)
M	`Memory A - M(3.3)` `>A.B.D.I.T.K.+.-.×.M.L.C.P ?`	(Transfer of MATRIX A to MEMORY MATRIX M)
	`Matrix A(3.3):B(2.2)` `>A.B.D.I.T.K.+.-.×.M.L.C.P ?_`	

• Determinant (det A)

D	`Determinant A = .....` `>A.B.D.I.T.K.+.-.×.M.L.C.P ?`	(Determinant)
	`Determinant A =-2` `>A.B.D.I.T.K.+.-.×.M.L.C.P ?`	(Result display)
EXE	`Matrix A(3.3):B(2.2)` `>A.B.D.I.T.K.+.-.×.M.L.C.P ?_`	(Return to menu display)

Here, the determinant of MATRIX A is −2.

• Inverse matrix (A^{-1})

I	`Inverse A - A` `>A.B.D.I.T.K.+.-.×.M.L.C.P ?`	(Inverse matrix)
	`a(1.1) = 0.5` `>A.B.D.I.T.K.+.-.×.M.L.C.P ?`	(Inverse matrix display)

Key	Display	
EXE	a(1.2) = 0 >A.B.D.I.T.K.+.-.×.M.L.C.P ?	(Press EXE after confirmation)
EXE	a(1.3) = 0 >A.B.D.I.T.K.+.-.×.M.L.C.P ?	
EXE	a(2.1) = 0.5 >A.B.D.I.T.K.+.-.×.M.L.C.P ?	
EXE	a(2.2) =-3 >A.B.D.I.T.K.+.-.×.M.L.C.P ?	
EXE	a(2.3) = 2 >A.B.D.I.T.K.+.-.×.M.L.C.P ?	
EXE	a(3.1) =-1 >A.B.D.I.T.K.+.-.×.M.L.C.P ?	
EXE	a(3.2) = 2 >A.B.D.I.T.K.+.-.×.M.L.C.P ?	
EXE	a(3.3) =-1 >A.B.D.I.T.K.+.-.×.M.L.C.P ?	
EXE	Matrix A(3.3):B(2.2) >A.B.D.I.T.K.+.-.×.M.L.C.P ?_	(Return to initial display)

Here, the inverse matrix of MATRIX A (A^{-1}) is $\begin{bmatrix} 0.5 & 0 & 0 \\ 0.5 & -3 & 2 \\ -1 & 2 & -1 \end{bmatrix}$

• Transposed matrix (A')

Key	Display	
L	Load A ← M(3.3) >A.B.D.I.T.K.+.-.×.M.L.C.P ?	(Transfer of MEMORY MATRIX M to MATRIX A)
	Matrix A(3.3):B(2.2) >A.B.D.I.T.K.+.-.×.M.L.C.P ?_	
T	Transpose A ← A(3.3) >A.B.D.I.T.K.+.-.×.M.L.C.P ?	(Transposed matrix)
	a(1.1) = 2 >A.B.D.I.T.K.+.-.×.M.L.C.P ?	(Transposed matrix display)
EXE	a(1.2) = 3 >A.B.D.I.T.K.+.-.×.M.L.C.P ?	(Press EXE after confirmation)
EXE	a(1.3) = 4 >A.B.D.I.T.K.+.-.×.M.L.C.P ?	
EXE	a(2.1) = 0 >A.B.D.I.T.K.+.-.×.M.L.C.P ?	
EXE	a(2.2) = 1 >A.B.D.I.T.K.+.-.×.M.L.C.P ?	
EXE	a(2.3) = 2 >A.B.D.I.T.K.+.-.×.M.L.C.P ?	
EXE	a(3.1) = 0 >A.B.D.I.T.K.+.-.×.M.L.C.P ?	
EXE	a(3.2) = 2 >A.B.D.I.T.K.+.-.×.M.L.C.P ?	
EXE	a(3.3) = 3 >A.B.D.I.T.K.+.-.×.M.L.C.P ?	
EXE	Matrix A(3.3):B(2.2) >A.B.D.I.T.K.+.-.×.M.L.C.P ?_	(Return to initial display)

Here, the transposed matrix for MATRIX A (A') is $\begin{bmatrix} 2 & 3 & 4 \\ 0 & 1 & 2 \\ 0 & 2 & 3 \end{bmatrix}$.

Scalar product

EXAMPLE 5

Calculate the scalar products for the following matrices.

$$3\begin{bmatrix}1 & 2\\ 2 & 1\end{bmatrix}\begin{bmatrix}3 & 1\\ 0 & 2\end{bmatrix}$$

$$A=\begin{bmatrix}1 & 2\\ 2 & 1\end{bmatrix},\quad B=\begin{bmatrix}3 & 1\\ 0 & 2\end{bmatrix}$$

Multiply MATRIX B by the result of MATRIX A times three.

Key operation	Display	
	Matrix A(2.2):B(2.2) >A.B.D.I.T.K.+.-.*.M.L.C.P ?_	
A2 [EXE] 2 [EXE]	a(1.1) = 0 ?_	(MATRIX A set up)
1 [EXE] 2 [EXE] 2 [EXE] 1 [EXE]	Matrix A(2.2):B(2.2) >A.B.D.I.T.K.+.-.*.M.L.C.P ?_	(Element input)
K	k*A(2.2) → A k = 1 ?_	(Scalar product)
3 [EXE]	a(1.1) = 3 >A.B.D.I.T.K.+.-.*.M.L.C.P ?	(Multiplication by 3 result assigned to MATRIX A)
B2 [EXE] 2 [EXE]	b(1.1) = 0 ?_	(MATRIX B set up)
3 [EXE] 1 [EXE] 0 [EXE] 2 [EXE]	Matrix A(2.2):B(2.2) >A.B.D.I.T.K.+.-.*.M.L.C.P ?_	
*	A*B → A(2.2) >A.B.D.I.T.K.+.-.*.M.L.C.P ?	(A and B multiplication)
[EXE]	a(1.1) = 9 >A.B.D.I.T.K.+.-.*.M.L.C.P ?	(Display result)
[EXE]	a(1.2) = 15 >A.B.D.I.T.K.+.-.*.M.L.C.P ?	(Press [EXE] after confirmation)
[EXE]	a(2.1) = 18 >A.B.D.I.T.K.+.-.*.M.L.C.P ?	
[EXE]	a(2.2) = 12 >A.B.D.I.T.K.+.-.*.M.L.C.P ?	
[EXE]	Matrix A(2.2):B(2.2) >A.B.D.I.T.K.+.-.*.M.L.C.P ?_	(Return to initial display)

Here, the result of the example calculation is $3\begin{bmatrix}1 & 2\\ 2 & 1\end{bmatrix}\begin{bmatrix}3 & 1\\ 0 & 2\end{bmatrix}=\begin{bmatrix}9 & 15\\ 18 & 12\end{bmatrix}$.

• HELP-menu

Pressing [.] in the menu display produces a HELP display which explains the meaning of each command.

```
 A : Input A(m.n)
 B : Input B(m.n)
```

Pressing [⇧], [⇩] or [EXE] at this time scrolls through the commands. Pressing either [⇦] or [⇨] returns to the initial display.

• Matrix display

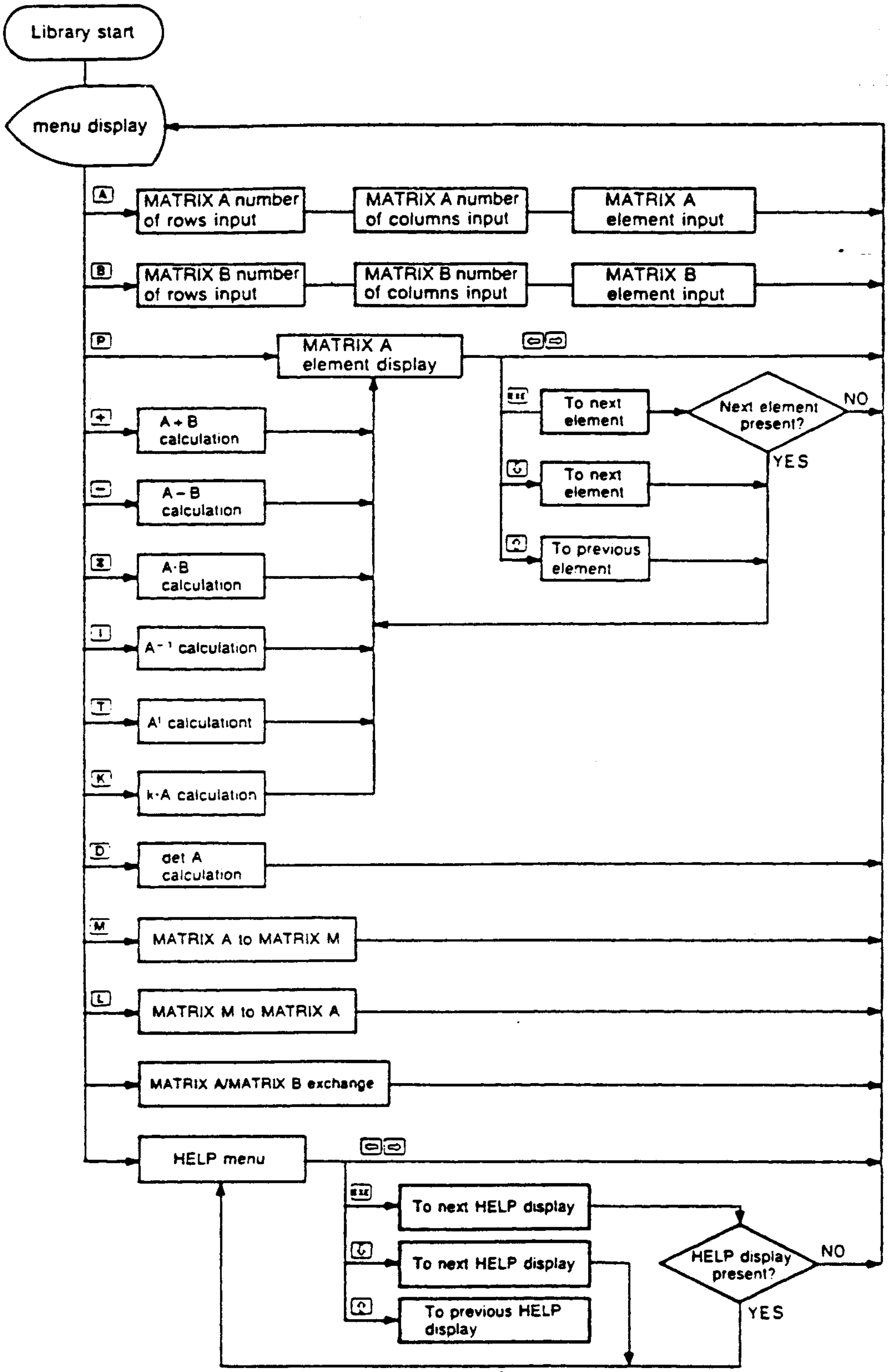
After performing matrix addition, subtraction, multiplication, scalar product, determinant, inverse matrix, and transposed matrix calculations, the result of the calculation (contents of MATRIX A) is shown on the display. As with the HELP menu, [⇩] and [⇧] ([EXE]) can be used to scroll through MATRIX A.

* The operation of [⇩] and [EXE] is identical, with display being performed in the same sequence as the matrix element input. The [⇧] key displays the elements in reverse sequence.
* Pressing [⇦] or [⇨] returns to the menu display regardless of the current display.
* The [P] key can be used from the menu display to display the contents of MATRIX A. [⇧], [⇩], [EXE], [⇨] and [⇦] can also be used as desired.

EXAMPLE

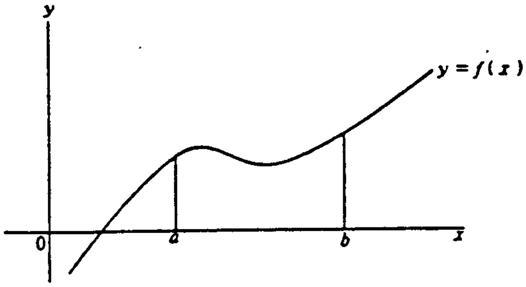
Key	Display	
	Matrix A(2.2):B(2.2) >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	
P	a(1.1) = 1 >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	(MATRIX A element display selection)
[⇩]	a(1.2) = 2 >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	(Confirmation of each element)
[EXE]	a(2.1) = 3 >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	
[⇧]	a(1.2) = 2 >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	
[⇨]	Matrix A(2.2):B(2.2) >A.B.D.I.T.K.+.-.x.M.L.C.P ?_	(Return to initial display)

MATRIX OPERATION FLOWCHART



5200 NUMERIC INTEGRATION (ROMBERG'S METHOD)

Determines the integral value of interval [a, b] of the function $y = f(x)$ graphed below using Romberg's Method.



(Angle unit = radians)

The following parameters are specified in order to determine the numeric integration using Romberg's Method.

a, b : Interval
ϵ : Error conditions to determine number of divisions ($\epsilon > |\text{area}_{n+1} - \text{area}_n|$)
loop : Maximum number of divisions (positive integer)

The initial value of the area is determined using the trapezoidal formula.

- The variable name for the function f(x) must be represented by x.
- The value input for ϵ must be 1E−90 or more. Since internal calculations are performed in 12 digits, smaller values have little meaning.

OPERATION

5200 LIB

```
Romberg's method  ∫f(x)dx [a.b]
1:f(x).[a.b]      2:ε.loop
```

The menu display illustrated above appears when the library is activated. Either 1 or 2 should be selected in accordance with the type of processing to be performed.

1 : Function f(x) specification/interval input
2 : Error condition/maximum number of divisions input

EXAMPLE

Determine the integral values in intervals [3, 5] when f(x) = lnx. The error condition (ϵ) = 0.0001, and the maximum number of divisions (loop) is 2^{10}.

Key	Display	
	Romberg's method ∫f(x)dx [a.b] / 1:f(x).[a.b] 2:ε.loop	
2	Err |An+1−An|<ε (ε>0) / ε= 0.0000001 ?_	(Parameter input selection)
0.0001 EXE	Max loop 2ⁿ (n>0) / n= 8 ?_	(Error condition input)
10 EXE	Romberg's method ∫f(x)dx [a.b] / 1:f(x).[a.b] 2:ε.loop	(Maximum number of divisions input)
1	Define function / f(x) ?_	(Function specification, interval input selection)

Key operation	Display	
[ln] [X] [EXE]	∫ LNx dx [a.b] a= 0 ?_	(Function specification)
3 [EXE]	∫ LNx dx [a.b] b= 0 ?_	(Interval input)
5 [EXE]	∫ LNx dx [a.b] ∫f(x)dx = ...	
	∫ LNx dx [a.b] ∫f(x)dx = 2.7514	(Integral value display)
[EXE]	Romberg's method ∫f(x)dx_[a.b] 1:f(x).[a.b] 2:ε.loop	(Return to menu display)

This display indicates that the integral value for the example is 2.7514.

The message "not found" is displayed when an integral value cannot be found.

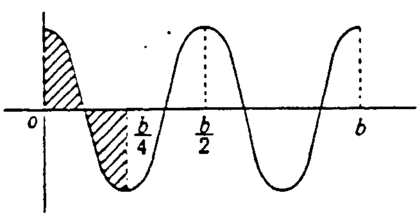
Display	
∫ LNx dx [a.b] not found	(Interval of [0.1] in the same example)

IMPORTANT

Depending on the type of integration function or the integration range, large errors may be generated in values obtained through integration. The following points should be carefully noted to ensure accurate integral values.

1. Periodic Functions and Symmetric Functions

Perform calculations for each period or symmetrical cycle.

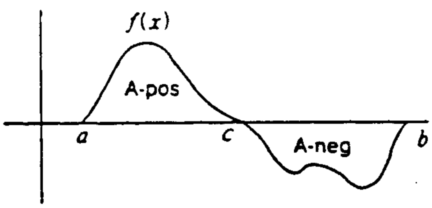


$$\int_0^b f(x)dx = \int_0^{\frac{b}{n}} n \cdot f(x)dx$$

In the graph to the left, $n = 4$.

2. Positive/Negative Integral Values According to Integral

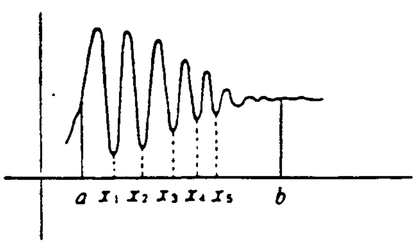
Divide into positive portion and negative portion, and calculate individually.



$$\int_a^b f(x)dx = \underbrace{\int_a^c f(x)dx}_{\text{Positive portion A-pos}} + \underbrace{\int_c^b f(x)dx}_{\text{Negative portion A-neg}}$$

3. Large Fluctuation in Integral Values Due to Minute Fluctuation in Integration Range

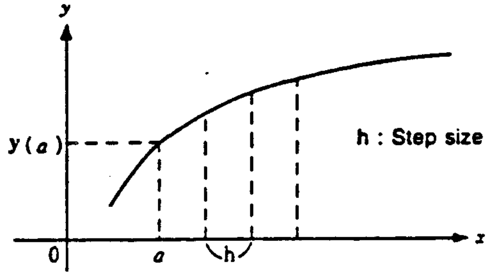
Divide the integral interval (make the interval smaller where the fluctuation is large), and calculate individually.



$$\int_a^b f(x)dx = \int_a^{x_1} f(x)dx + \int_{x_1}^{x_2} f(x)dx + \cdots\cdots + \int_{x_5}^b f(x)dx$$

5220 ORDINARY DIFFERENTIAL EQUATION
(RUNGE-KUTTA METHOD)

The differential equation expressed as $\frac{dy}{dx} = f(x, y)$ returns $x = a$, $y = y(a)$ as the initial condition to obtain the numeric solution.



In the figure to the left, initial condition $x = a$, $y = y(a)$ is returned when the differential equation $\frac{dy}{dx}$ for the unknown function $y = f(x)$ is known, and the numeric solution for x in the unknown function is calculated.

OPERATION

5220 [LIB]	Define function dy/dx ?	

EXAMPLE

Express the differential equation $f(x, y) = \frac{3y}{1+x}$, (initial condition : $y(0) = 1$) with a numeric solution where the step size is 0.1.

3∗Y/(1+X) [EXE]	dy/dx = 3∗y/(1+x) x0= 0 ?_	(Enter ordinary differential equation)
0 [EXE]	dy/dx = 3∗y/(1+x) y0= 0 ?_	(Enter initial value of x)
1 [EXE]	Step-size ⊿h (⊿h>0) ⊿h= 1 ?	(Enter initial value of y)
0 . 1 [EXE]	dy/dx = 3∗y/(1+x) y(0) = 1	(Enter step size and display numeric solution when x = 0)
[EXE]	dy/dx = 3∗y/(1+x)	(Display numeric solution when x = 0.1)

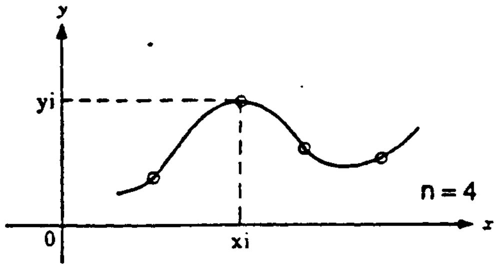
• "...." displayed as shown during calculation of numeric solution.

⋮	dy/dx = 3∗y/(1+x) y(0.1) = 1.330983302	
	⋮	
[EXE]	dy/dx = 3∗y/(1+x) y(5) = 215.9911132	(Display numeric solution when x = 5)
[EXE]	dy/dx = 3∗y/(1+x) ?3∗y/(1+x)	(Return to initial display)

While the numeric solution is displayed, [⇩] (or [EXE]) displays the next numeric solution while [⇧] displays the last numeric solution. The [⇦] and [⇨] keys return to the initial display. Also, the numeric solution can be displayed up to step size × n ($1 \leqq n \leqq 50$).

5230 LAGRANGE'S INTERPOLATION

An nth degree polynomial is created to connect n + 1 points on a plane, and the data are interpolated according to the polynomial. This unit is capable of handling points within the range of $2 \leq n \leq 200$ (n = integer).



Determine the n polynomial for the curve which passes through the four points noted on the left when n = 4.

OPERATION

5230 [LIB]

Lagrange's interpolation 1:x 2:set data

The following two operations can be selected from the initial display:
1 : Data interpolation
2 : Input of n number of points

EXAMPLE

Create a 3rd degree polynomial which connects the following three points and determine the value when x = 4.

P_1 (1, 3), P_2 (3, 1), P_3 (5, 2)

Key	Display	
2	Number of data n= 2 ?_	(Point input)
3 [EXE]	Number of data = 3 x1= 0 ?_	(Number of points)
1 [EXE]	Number of data = 3 y1= 0 ?_	(x-coordinate of point P_1)
3 [EXE]	Number of data = 3 x2= 0 ?_	(y-coordinate of point P_1)
3 [EXE] 1 [EXE] 5 [EXE] 2 [EXE]	Lagrange's interpolation 1:x 2:set data	(Coordinates of remaining points)
1	Lagrange's interpolation x?_	(Data interpolation)
4 [EXE]	Lagrange's interpolation x?4 :y= 1.125	(x-data input and 1.125 is obtained for y)
[EXE]	Lagrange's interpolation x?_	(x-data input requested)
[EXE]	Lagrange's interpolation 1:x 2:set data	(Return to menu display)

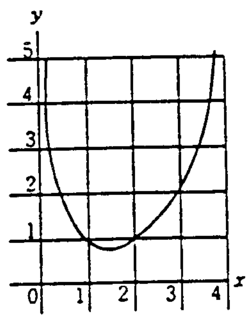
Here it can be seen that a value of 1.125 is obtained when x = 4.

- The "not found" message as illustrated below appears when interpolation is not performed using the nth degree polynomial.

```
not found
```

5250 GAMMA FUNCTION Γ(x)

Determines the value of the gamma function within the range of $0 < x \leqq 70$ with six significant digits.



The gamma function is expressed as the graph shown on the left.

OPERATION

5250 LIB

```
Gamma function    (0<x≤70)
x = 1 ?_
```

EXAMPLE

Determine the value of the gamma function when x = 3.

3 EXE

```
Gamma function    (0<x≤70)
Γ(3) = 2
```

(x value input)
(Result display)

EXE

```
Gamma function    (0<x≤70)
x = 3 ?_
```

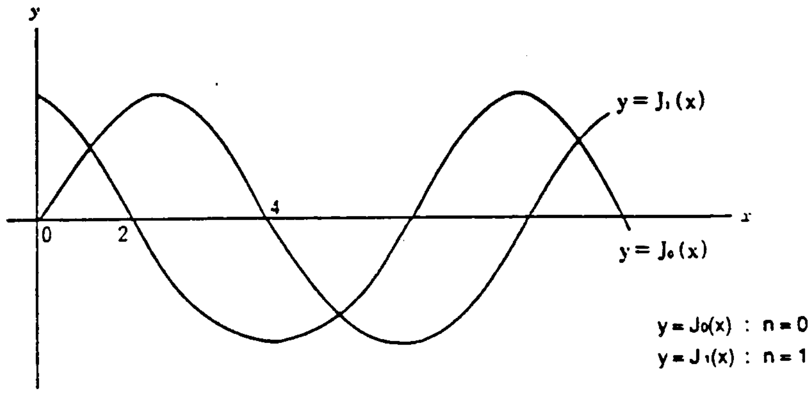
(Return to initial display)

Here, the value of the gamma function is 2.

- A total of six entries (including decimal points) can be made for input of x.

5260 BESSEL FUNCTION Jn(x)

Determines the elementary solution Jn(x) of the Bessel differential equation $\frac{d^2y}{dx^2} + \frac{1}{x} \cdot \frac{dy}{dx} + (1 - \frac{n^2}{x^2})\, y = 0$ within the range of $0 \leqq n \leqq 9$ (integer), $0 \leqq x \leqq 30$ (condition of x) with six significant digits.



OPERATION

5260 [LB]

```
Jn(x)          (0≦n≦9.0≦x≦30)
n?_    :x?     :J=
```

EXAMPLE

Determine the Bessel function Jn(x) when n = 2 and x = 3.

2 [EXE] 3 [EXE]

```
Jn(x)          (0≦n≦9.0≦x≦30)
n?2    :x?3    :J= .....
```

(n and x value input)

```
Jn(x)          (0≦n≦9.0≦x≦30)
n?2    :x?3    :J= 0.486091
```

(Result display)

[EXE]

```
Jn(x)          (0≦n≦9.0≦x≦30)
n?_    :x?     :J=
```

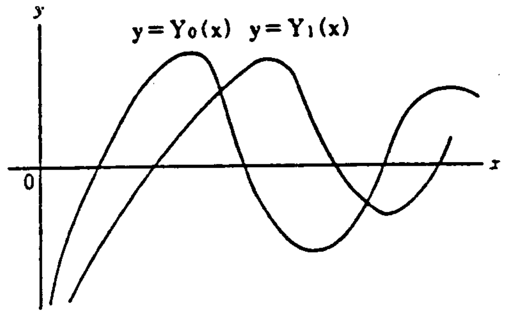
(Return to initial display)

Here, the Bessel funciton value is 0.486091.

• A total of six entries (including decimal points) can be made for input of x.

5270 BESSEL FUNCTION Yn(x)

Determines the elementary solution Yn(x) of the Bessel differential equation $\frac{d^2y}{dx^2} + \frac{1}{x} \cdot \frac{dy}{dx} + (1 - \frac{n^2}{x^2})\, y = 0$ within the range of $0 \leqq n \leqq 9$ (integer), $0 < x \leqq 30$ (condition of x) with six significant digits.



OPERATION

5270 [LIB]

Yn(x)	(0≦n≦9.0<x≦30)
n?_ :x?	:Y=

[EXAMPLE]

Determine the Bessel function Yn(x) when n=3 and x=4.

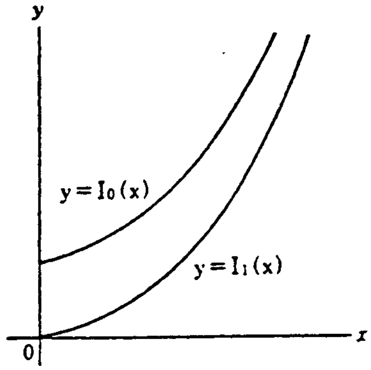
Key	Display		Note
3 [EXE] 4 [EXE]	Yn(x) n?3 :x?4	(0≦n≦9.0<x≦30) :Y=	(n and x value input)
	Yn(x) n?3 :x?4	(0≦n≦9.0<x≦30) :Y=-0.182022	(Result display)
[EXE]	Yn(x) n?_ :x?	(0≦n≦9.0<x≦30) :Y=	(Return to initial display)

Here, the Bessel function value is −0.182022.

* A total of six entries (including decimal points) can be made for input of x.

5280 MODIFIED BESSEL FUNCTION In(x)

Determines the elementary solution In(x) of the modified Bessel differential equation $\frac{d^2y}{dx^2} + \frac{1}{x} \cdot \frac{dy}{dx} - (1 + \frac{n^2}{x^2})\, y = 0$ within the range of $0 \leqq n \leqq 9$ (integer), $0 \leqq x \leqq 10$ (condition of x) with six significant digits.



$y = I_0(x)$: $n = 0$
$y = I_1(x)$: $n = 1$

OPERATION

5280 [LIB]

```
In(x)             (0≦n≦9.0≦x≦10)
n?_     :x?       :I=
```

EXAMPLE

Determine the modified Bessel function when n = 3 and x = 5.

3 [EXE] 5 [EXE]

```
In(x)             (0≦n≦9.0≦x≦10)
n?3     :x?4      :I= . . . .
```
(n and x value input)

```
In(x)             (0≦n≦9.0≦x≦10)
n?3     :x?5      :I= 10.3312
```
(Result display)

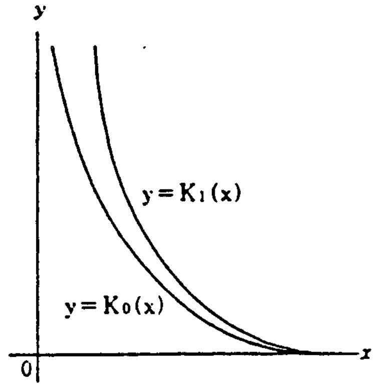
[EXE]

```
In(x)             (0≦n≦9.0≦x≦10)
n?_     :x?       :I=
```
(Return to initial display)

Here, the modified Bessel function value is 10.3312.
A total of six entries (including decimal points) can be made for input of x.

5290 MODIFIED BESSEL FUNCTION Kn(x)

Determines the elementary solution Kn(x) of the modified Bessel differential equation $\frac{d^2y}{dx^2} + \frac{1}{x} \cdot \frac{dy}{dx} - (1 + \frac{n^2}{x^2})\,y = 0$ within the range of $0 \leqq n \leqq 9$ (integer), $0 < x \leqq 10$ (condition of x) with six significant digits.



$y = K_0(x)$: $n = 0$
$y = K_1(x)$: $n = 1$

OPERATION

5290 [LB]

```
Kn(x)              (0≦n≦9.0<x≦10)
n?_     :x?         :K=
```

[EXAMPLE]

Determine the modified Bessel function Kn(x) when n = 4 and x = 6.

Key operation	Display	
4 [EXE] 6 [EXE]	Kn(x) (0≦n≦9.0<x≦10) n?4 :x?6 :K=	(n and x value input)
	Kn(x) (0≦n≦9.0<x≦10) n?4 :x?6 :K= 0.00416387	(Result display)
[EXE]	Kn(x) (0≦n≦9.0<x≦10) n?_ :x? :K=	(Return to initial display)

Here, the modified Bessel function value is 0.00416387.
* A total of six entries (including decimal points) can be made for input of x.

5300 COMPLEX NUMBER

Complex number calculations encompass arithmetic operations, and to determine absolute values, arguments, squares, square roots, and reciprocal numbers.
This unit is capable of a wide variety of complex number calculations, with the allowable range of input value<1E50.

OPERATION

5300 [LB]

```
 0
>A.G.I.S.^.+.-.*./.M.L.C ?_
```

The complex number menu display allows selection of the following processes:

A :	Input of complex number A (a+bi)	
G :	Absolute value (r) and arguments (θ) for complex number A (resulting angle unit determined by current mode setting)	
I :	Reciprocal number for complex number A	$1/(a+bi) \rightarrow (a+bi)$
S :	Square root of complex number A	$\sqrt{(a+bi)} \rightarrow (a+bi)$
^ :	Square of complex number A	$(a+bi)^2 \rightarrow (a+bi)$
+ :	Addition of complex number A and complex number B (c+di)	$(a+bi)+(c+di) \rightarrow (a+bi)$
− :	Subtraction of complex number A and complex number B	$(a+bi)-(c+di) \rightarrow (a+bi)$
* :	Multiplication of complex number A and complex number B	$(a+bi)\times(c+di) \rightarrow (a+bi)$
/ :	Division of complex number A and complex number B	$(a+bi)\div(c+di) \rightarrow (a+bi)$
M :	Assigns contents of complex number A to complex number memory M (e+fi)	$(a+bi) \rightarrow (e+fi)$
L :	Assigns contents of complex number memory M (e+fi) to complex number A	$(a+bi) \leftarrow (e+fi)$
C :	Exchanges contents of complex number A and complex number B	$(a+bi) \leftrightarrow (c+di)$
. :	Help (explanation of each operation)	

• Complex Number Specification

Complex number specification is performed by pressing [A] while in the menu display.

EXAMPLE

Assign 5+7i to complex number A.

[A]

```
Complex number  A(a+bi)
a= 0 ?_
```

(Specification of complex number input)

5 [EXE]

```
Complex number  A(a+bi)
b= 0 ?_
```

7 [EXE]

```
 5 + 7i
>A.G.I.S.^.+.-.*./.M.L.C ?_
```

• Arithmetic Operations

EXAMPLE

Perform the following operations:

(2 + 3i) + (3 − 2i)

[A]	Complex number A(a+bi) a= 0 ?_	(Specification of complex number input)
2 [EXE] 3 [EXE]	2 + 3i <A.G.I.S.^.+.−.*./.M.L.C ?_	(Input of complex number A)
[+]	Complex number B(c+di) c= 0 ?_	(Addition)
3 [EXE] [−] 2 [EXE]	5 + i >A.G.I.S.^.+.−.*./.M.L.C ?_	(Input of complex number B)

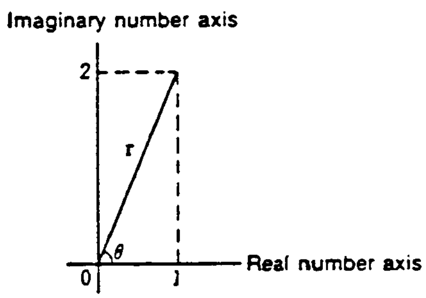
This display indicates (2 + 3i) + (3 − 2i) = 5 + i.
The same procedure can be performed for subtraction, multiplication and division.

• Absolute Values/Arguments

EXAMPLE

Determine the absolute value (r) and argument (θ) for (1 + 2i).

Angle unit : DEG ([MODE] [4])



[A]	Complex number A(a+bi) a= 0 ?_	(Specification of complex number input)
1 [EXE] 2 [EXE]	1 + 2i >A.G.I.S.^.+.−.*./.M.L.C?_	(Input of complex number A)
[G]	r = 2.236067977 θ = 63.43494882	(Calculation of absolute value and argument)

Here, the absolute value (r) for (1 + 2i) is 2.236067977, and the argument is 63.43494882 (DEG). The resulting angle unit is determined by the current ANGLE mode setting.

* The angle unit is specified as follows:

[MODE] [4] : Degrees
[MODE] [5] : Radians
[MODE] [6] : Grads

• Square/Square Root/Reciprocal number

EXAMPLE

Calculate the following:

① $(2+i)^2$ ② $\sqrt{(-7+24i)}$ ③ $\dfrac{1}{3+2i}$

① Square

Key operation	Display	
[A]	Complex number A(a+bi) a= 0 ?_	(Specification of complex number input)
2 [EXE] 1 [EXE]	2 + i >A.G.I.S.^.+.-.x./.M.L.C ?_	(Input of complex number A)
[^]	3 + 4i >A.G.I.S.^.+.-.x./.M.L.C ?_	(Square)

This display indicates $(2+i)^2=3+4i$.

② Square Root

Key operation	Display	
[A]	Complex number A(a+bi) a= 0 ?_	(Specification of complex number input)
[−] 7 [EXE] 24 [EXE]	-7 + 24i >A.G.I.S.^.+.-.x./.M.L.C ?_	(Input of complex number A)
[S]	3 + 4i >A.G.I.S.^.+.-.x./.M.L.C ?_	(Square root)

This display indicates $\sqrt{(-7+24i)}=3+4i$.

③ Reciprocal Number

Key operation	Display	
[A]	Complex number A(a+bi) a= 0 ?_	(Specification of complex number input)
3 [EXE] 2 [EXE]	3 + 2i >A.G.I.S.^.+.-.x./.M.L.C ?_	(Input of complex number A)
[I]	0.2307692 - 0.1538462i >A.G.I.S.^.+.-.x./.M.L.C ?_	(Reciprocal number)

• Memory Calculations

EXAMPLE

Perform the following calculations using the memory function:

$(3+2i) + (4+6i)$

$(3+2i) - (-3+9i)$

Key operation	Display	
[A]	Complex number A(a+bi) a= 0 ?_	(Specification of complex number input)
3 [EXE] 2 [EXE]	3 + 2i >A.G.I.S.^.+.-.x./.M.L.C ?_	(Input of complex number A)
[M]	3 + 2i >A.G.I.S.^.+.-.x./.M.L.C ?_	(Assigns complex number A to complex number memory)
[+]	Complex number B(c+di) c= 0 ?_	(Addition)
4 [EXE] 6 [EXE]	7 + 8i >A.G.I.S.^.+.-.x./.M.L.C ?_	(Assigns 4+6i to B)

This display indicates $(3+2i) + (4+6i) = 7+8i$.

[L]

```
 3 + 2i
>A.G.I.S.^.+.-.*./.M.L.C ?_
```

(Assigns complex number in memory to complex number A in place of reinput)

[-]

```
Complex number B(c+di)
c= 4 ?_
```

(Subtraction)

[-] 3 [EXE] 9 [EXE]

```
 6 - 7i
>A.G.I.S.^.+.-.*./.M.L.C ?_
```

(Assigns −3+9i to B)

This display indicates (3+2i) − (−3+9i)=6−7i.

• Exchange

EXAMPLE

Set the following two complex numbers for complex numbers A and B:
(5+2i), (3+4i)

[A]

```
Complex number A(a+bi)
a= 0 ?
```

(Complex number A input)

3 [EXE] 4 [EXE]

```
 3 + 4i
>A.G.I.S.^.+.-.*./.M.L.C ?_
```

(First set 3+4i for complex number A)

[C]

```
0
>A.G.I.S.^.+.-.*./.M.L.C ?_
```

(Assign the contents of complex number A to complex number B)

[A]

```
Complex number A(a+bi)
a= 0 ?
```

(Input complex number A)

5 [EXE] 2 [EXE]

```
 5 + 2i
>A.G.I.S.^.+.-.*./.M.L.C.?_
```

(Set 5+2i for complex number A)

The above operation sets 5+2i for complex number A, and 3+4i for complex number B.

• Help Display

Pressing [•] while in the menu display produces an explanation of each command.

```
 A : Input A
 G : Gauss  r=|A|:θ=argument(A)
```

At this time [⇧], [EXE], and [⇩] can be used to scroll the display. Each press of [EXE] ([⇩]) advances to the next command, while pressing [⇧] returns to the previous command. Pressing [⇦] or [⇨] returns to the menu display. The menu display is also returned to after the final command is displayed.

5350 BINARY-DECIMAL-HEXADECIMAL

Binary, decimal and hexadecimal calculations encompass basic arithmetic operations, logical operations, twos complement, logical shift, and conversions.
This unit is capable of combining binary, decimal and hexadecimal values, with the allowable range values being −2147483648 ~ 2147483647 (32-bit).

OPERATION

5350 [LB]

```
[DEC]  0
>I.B.D.H.+.-.*./.A.O.X.N.C.L.R?_
```

The binary/decimal/hexadecimal calculation menu display allows selection of the following processes:

I : Value input
B : Converts displayed value to binary number
D : Converts displayed value to decimal number
H : Converts displayed value to hexadecimal number
+ : Addition
− : Subtraction
* : Multiplication
/ : Division
A : AND (logical product)
O : OR (logical sum)
X : XOR (exclusive logical sum)
N : NOT (negation)
C : Twos complement
L : Logical shift left
R : Logical shift right
. : Help (explanation of each operation)

• Operations and Display

1. The following indicators in the upper left of the display in the menu indicate the current base mode setting:
   [DEC] : Decimal mode
   [HEX] : Hexadecimal mode
   Blank : Binary mode
2. Entering values besides 0 and 1 for binary calculations, values besides 0 ~ 9 for decimal calculations, values besides 0 ~ 9/A ~ F (upper case or lower case) for hexadecimal calculations, or values greater than 32 bits causes the entered value to be disregarded. Binary, decimal and hexadecimal values may be used in combination in a single calculation.

EXAMPLE

The following operations may be used to enter values regardless of the current base mode setting:

15, D : Decimal 15 (hexadecimal F, binary 1111)
15, H : Hexadecimal 15 (decimal 21, binary 10101)
1010, B : Binary 1010 (decimal 10, hexadecimal A)

Results are always displayed using the current base mode setting.

• Arithmetic Operations

EXAMPLE

Perform the following calculations:

① $1011001_B + 1100_B$
② $2AC_H \times 1B_H$
③ $FF00_H + 1010_B$

①

Operation	Display	
	[DEC] 0 >I.B.D.H.+.-.x./.A.O.X.N.C.L.R?_	
[B]	00000000000000000000000000000000 >I.B.D.H.+.-.x./.A.O.X.N.C.L.R?_	(Binary mode)
[I]	Input data x (.B.D.H) [BIN] x ?_	(Value input specification)
1011001 [EXE]	00000000000000000000000001011001 >I.B.D.H.+.-.x./.A.O.X.N.C.L.R?_	(Value input)
[+]	00000000000000000000000001011001 + _	(Addition)
1100 [EXE]	00000000000000000000000001100101 >I.B.D.H.+.-.x./.A.O.X.N.C.L.R?_	(Value input)

This display indicates $1011001_B + 1100_B = 1100101_B$

②

Operation	Display	
[H]	[HEX] 00000000 >I.B.D.H.+.-.x./.A.O.X.N.C.L.R?_	(Hexadecimal mode)
[I]	Input data x (.B.D.H) [HEX] x ?_	(Value input specification)
2 [A] [C] [EXE]	[HEX] 000002AC >I.B.D.H.+.-.x./.A.O.X.N.C.L.R?_	(Value input)
[x]	[HEX] 000002AC x _	(Multiplication)
1 [B] [EXE]	[HEX] 00004824 >I.B.D.H.+.-.x./.A.O.X.N.C.L.R?_	(Value input)

This display indicates $2AC_H \times 1B_H = 4824_H$.
The same procedure can be performed for subtraction and division.

③

Operation	Display	
[H]	[HEX] 00000000 >I.B.D.H.+.-.x./.A.O.X.N.C.L.R?_	(Hexadecimal mode)
[I]	Input data x (.B.D.H) [HEX] x ?_	(Value input specification)
[F] [F] 00 [EXE]	[HEX] 0000FF00 >I.B.D.H.+.-.x./.A.O.X.N.C.L.R?_	(Value input)

[+]	[HEX] 0000FF00 + _	(Addition)
1010 [,] [B] [EXE]	[HEX] 0000FF0A >I.B.D.H.+.-.x./.A.O.X.N.C.L.R?_	(Binary value input)

This display indicates $FF00_H + 1010_B = FF0A_H$

• Logical Operations

EXAMPLE

Perform the following operations for $A = 110101_B$ and $B = 101111_B$.

①A OR B (logical sum) ②A AND B (logical product)
③A XOR B (exclusive logical sum) ④A NOT (negation)

①

	[DEC] 0 >I.B.D.H.+.-.x./.A.O.X.N.C.L.R?_	
[B]	00000000000000000000000000000000 >I.B.D.H.+.-.x./.A.O.X.N.C.L.R?_	(Binary mode)
[I]	Input data x (.B.D.H) [BIN] x ?_	(Value input specification)
110101 [EXE]	00000000000000000000000000110101 >I.B.D.H.+.-.x./.A.O.X.N.C.L.R?_	(Value input)
[O]	00000000000000000000000000110101 OR _	(OR)
101111 [EXE]	00000000000000000000000000111111 >I.B.D.H.+.-.x./.A.O.X.N.C.L.R?_	(Value input)

This display indicates A OR B = 111111_B.

②

[I]	Input data x (.B.D.H) [BIN] x ?_	(Value input specification)
110101 [EXE]	00000000000000000000000000110101 >I.B.D.H.+.-.x./.A.O.X.N.C.L.R?_	(Value input)
[A]	00000000000000000000000000110101 AND	(AND)
101111 [EXE]	00000000000000000000000000100101 >I.B.D.H.+.-.x./.A.O.X.N.C.L.R?_	(Value input)

This display indicates A AND B = 100101_B.

③

[I]	Input data x (.B.D.H) [BIN] x ?_	(Value input specification
110101 [EXE]	00000000000000000000000000110101 >I.B.D.H.+.-.x./.A.O.X.N.C.L.R?_	(Value input)
[X]	00000000000000000000000000110101 XOR _	(XOR)
101111 [EXE]	00000000000000000000000000011010 >I.B.D.H.+.-.x./.A.O.X.N.C.L.R?_	(Value input)

This display indicates A XOR B = 11010_B.

④

Key	Display	
[I]	Input data x (.B.D.H) [BIN] x ?_	(Value input specification)
110101 [EXE]	00000000000000000000000000110101 >I.B.D.H.+.-.×./.A.O.X.N.C.L.R?_	(Value input)
[N]	11111111111111111111111111001010 >I.B.D.H.+.-.×./.A.O.X.N.C.L.R?_	(NOT)

This display indicates NOT A=11111111111111111111111111001010B.

• Complement/Shift Operations

EXAMPLE

Perform the following operations:

① Twos complement of 11001010B

② 1-bit logical shift left of 110000B

③ 2-bit logical shift right of 1FCH

①

Key	Display	
	[DEC] 0 >I.B.D.H.+.-.×./.A.O.X.N.C.L.R?_	
[B]	00000000000000000000000000000000 >I.B.D.H.+.-.×./.A.O.X.N.C.L.R?_	(Binary mode)
[I]	Input data x (.B.D.H) [BIN] x ?_	(Value input specification)
11001010 [EXE]	00000000000000000000000011001010 >I.B.D.H.+.-.×./.A.O.X.N.C.L.R?_	(Value input)
[C]	11111111111111111111111100110110 >I.B.D.H.+.-.×./.A.O.X.N.C.L.R?_	(Twos complement)

This display indicates that the twos complement of 11001010B is 11111111111111111111111100110110B.

②

Key	Display	
[I]	Input data x (.B.D.H) [BIN] x ?_	(Value input specification)
110000 [EXE]	00000000000000000000000000110000 >I.B.D.H.+.-.×./.A.O.X.N.C.L.R?_	(Value input)
[L]	00000000000000000000000001100000	(Shift left)

This display indicates that shifting 110000B one bit to the left results in 1100000B.

③

Key	Display	
[H]	[HEX] 00000060 >I.B.D.H.+.-.×./.A.O.X.N.C.L.R?_	(Hexadecimal mode)
[I]	Input data x (.B.D.H) [HEX] x ?_	(Value input specification)
1 [F] [C] [EXE]	[HEX] 000001FC >I.B.D.H.+.-.×./.A.O.X.N.C.L.R?_	(Value input)
[R] [R]	[HEX] 0000007F >I.B.D.H.+.-.×./.A.O.X.N.C.L.R?_	(2-bit shift right)

This display indicates that shifting 1FCH two bits to the right results in 7FH.

• Base conversion

EXAMPLE

Convert the hexadecimal value AF3C to its decimal and binary equivalents.

Key	Display	
	[DEC] 0 >I.B.D.H.+.-.*./.A.O.X.N.C.L.R?_	
[H]	[HEX] 00000000 >I.B.D.H.+.-.*./.A.O.X.N.C.L.R?_	(Hexadecimal mode)
[I]	Input data x (.B.D.H) [HEX] x ?_	(Value input specification)
[A] [F] 3 [C] [EXE]	[HEX] 0000AF3C >I.B.D.H.+.-.*./.A.O.X.N.C.L.R?_	(Value input)
[D]	[DEC] 44860 >I.B.D.H.+.-.*./.A.O.X.N.C.L.R?_	(Decimal mode)
[B]	00000000000000001010111100111100 >I.B.D.H.+.-.*./.A.O.X.N.C.L.R?_	(Binary mode)

This display indicates that the decimal equivalent of hexadecimal AF3C is 44860, and the binary equivalent is 1010111100111100B.

• Help Display

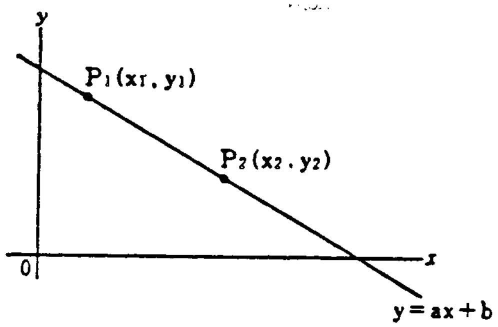
Pressing [·] while in the menu display produces an explanation of each command.

```
I : input data
B : binary mode
```

At this time [⇧], [EXE], and [⇩] can be used to scroll the display. Each press of [EXE] ([⇩]) advances to the next command, while pressing [⇧] returns to the previous command. Pressing [⊟] or [⇐] returns to the menu display. The menu display is also returned to after the final command is displayed.

5510 STRAIGHT LINE PASSING THROUGH TWO POINTS

Determines the straight line y ($y = ax + b$) which passes through points P_1 (x_1, y_1) and P_2 (x_2, y_2) on a plane.



OPERATION

5510 [LIB]

```
y=ax+b -- (x1,y1),(x2,y2)
x1= 0 ?_
```

EXAMPLE

Determine the line which passes through points P_1 (2,5) and P_2 (6,4)

Key operation	Display	
2 [EXE] 5 [EXE]	y=ax+b -- (x1,y1),(x2,y2) x1= 0 ?_	(P_1 input)
6 [EXE] 4 [EXE]	y=ax+b -- (x1,y1),(x2,y2) a =-0.25	(a value displayed following P_2 input)
[EXE]	y=ax+b -- (x1,y1),(x2,y2) b = 5.5	(b value display)
[▲]	y=ax+b -- (x1,y1),(x2,y2) a=-0.25	(Redisplay of a value)
[▼]	y=ax+b -- (x1,y1),(x2,y2) b = 5.5	(Redisplay of b value)
[EXE]	y=ax+b -- (x1,y1),(x2,y2) x1= 2 ?_	(Return to initial display)

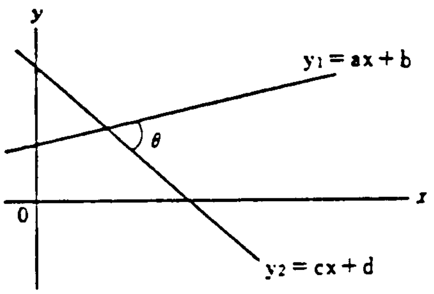
Here, the straight line is $y = -0.25x + 5.5$.

5520 ANGLE OF INTERSECTION FOR TWO STRAIGHT LINES

Determines the angle of intersection created by the two lines $y_1 = ax + b$ and $y_2 = cx + d$. The calculated angle for y_1 and y_2 is within the range of $-90° < \theta < 90°$. The resulting angle unit is determined by the current angle mode setting.

• The angle unit is specified as follows:

[MODE] [4] : Degrees
[MODE] [5] : Radians
[MODE] [6] : Grads



OPERATION

5520 [LB]

Angle(θ) -- y=ax+b・y=cx+d a= 0 ?_

[EXAMPLE]

Determine the angle of intersection (in DEG mode) for the straight lines $y_1 = \frac{1}{2}x + 2$ and $y_2 = 3x + 8$.

Key	Display	Note
1 [/] 2 [EXE] 3	Angle(θ) -- y=ax+b・y=cx+d c= 0 ?3_	(Entry of each line's slope)
[EXE]	Angle(θ) -- y=ax+b・y=cx+d θ = 45	(Angle of intersection = 45°)
[EXE]	Angle(θ) -- y=ax+b.y=cx+d a= 0.5 ?_	(Return to initial display)

[EXAMPLE]

Determine the angle of intersection (in DEG mode) for the straight lines $y_1 = 4x + 5$ and $y_2 = 4x + 7$.

Key	Display	Note
4 [EXE] 4	Angle(θ) -- y=ax+b・y=cx+d c= 0 ?4_	(Entry of each line's slope)
[EXE]	Angle(θ) -- y=ax+b・y=cx+d Parallel	(Indicates lines are parallel)
[EXE]	Angle(θ) -- y=ax+b.y=cx+d a= 4 ?_	(Return to initial display)

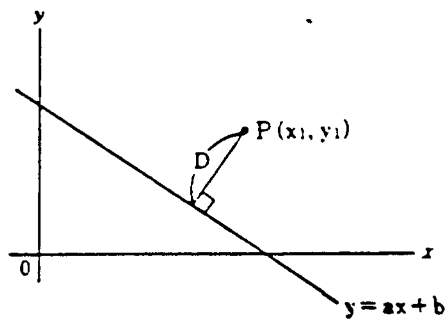
EXAMPLE

Determine the angle of intersection (in DEG mode) for the straight lines $y_1 = \frac{1}{2}x + 3$ and $y_2 = -2x + 4$.

1 [/] 2 [EXE] [−] 2	Angle(θ) -- y=ax+b・y=cx+d c= 0 ?-2_	(Entry of each line's slope)
[EXE]	Angle(θ) -- y=ax+b・y=cx+d Right angle	(Indicates angle of intersection is right angle)
[EXE]	Angle(θ) -- y=ax+b.y=cx+d a= 0.5 ?_	(Return to initial display)

5530 DISTANCE BETWEEN POINT AND STRAIGHT LINE

Determines length D of a perpendicular line from point P (x_1, y_1) and straight line $y = ax + b$.



OPERATION

5530 [LIB]	Distance -- y=ax+b・(x1・y1) a= 0 ?_

EXAMPLE

Determine length D of a perpendicular line from point P (6, 4) to straight line $y = 5x + 2$.

5 [EXE] 2 [EXE]	Distance -- y=ax+b・(x1・y1) x1= 0 ?_	(Straight line's slope and intercept)
6 [EXE] 4 [EXE]	Distance -- y=ax+b.(x1・y1) d = 5.491251784	(Distance displayed when coordinates of P are input)
[EXE]	Distance -- y=ax+b.(x1..y1) a= 5 ?_	(Return to initial display)

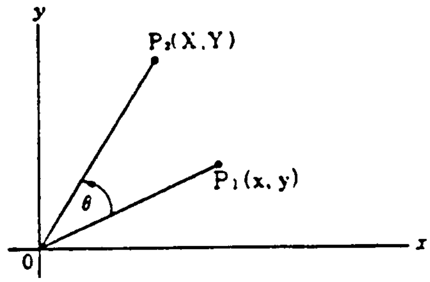
Here, the length of the perpendicular line is 5.491251784.

5540 ROTATIONAL MOVEMENT

Determines coordinates of point P_2 (X, Y) when a rotation of angle θ occurs from point P_1 (x_1, y_1). The angle unit is determined by the current angle mode setting.

* The angle unit is specified as follows:

[MODE] [4] : Degrees
[MODE] [5] : Radians
[MODE] [6] : Grads



OPERATION

Key operation	Display
5540 [LIB]	(X,Y) -- (x,y),angle(θ) x= 0 ?_

EXAMPLE

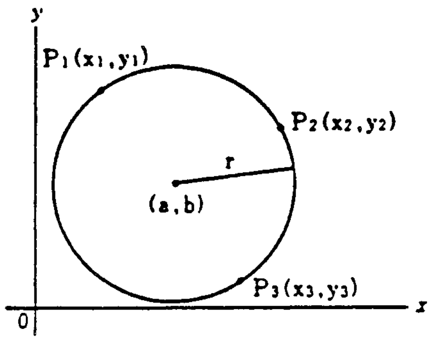
Determine the coordinates of point P_2 (X,Y) for rotation θ of 45 (in DEG mode) from point P_1 (4, 8).

Key operation	Display	Remarks
4 [EXE] 8 [EXE]	(X,Y) -- (x,y),angle(θ) θ= 0 ?_	(P_1 coordinates)
45 [EXE]	(X,Y) -- (x,y),angle(θ) X =-2.828427125	(X-coordinate displayed when angle input)
[EXE]	(X,Y) -- (x,y),angle(θ) Y = 8.485281374	(Y-coordinate displayed)
[↑]	(X,Y) -- (x,y),angle(θ) X =-2.828427125	(Redisplay of X-coordinate)
[↓]	(X,Y) -- (x,y),angle(θ) Y = 8.485281374	(Redisplay of Y-coordinate)
[EXE]	(X,Y) -- (x,y),angle(θ) x= 4 ?_	(Return to initial display)

Here, the coordinates of P_2 are (−2.828427125, 8.485281374).
The result is displayed in the sequence of X, Y, and the display can be scrolled to view following values using [↓] (or [EXE]), and previous values can be viewed using [↑].

5550 CIRCLE PASSING THROUGH THREE POINTS

Determines the equation $(x-a)^2+(y-b)^2=r^2$ for a circle passing through the points P_1 (x_1, y_1), P_2 (x_2, y_2), P_3 (x_3, y_3).



OPERATION

5550 [LB]	Circle (x1, y1), (x2, y2), (x3, y3) x1= 0 ?_	

EXAMPLE

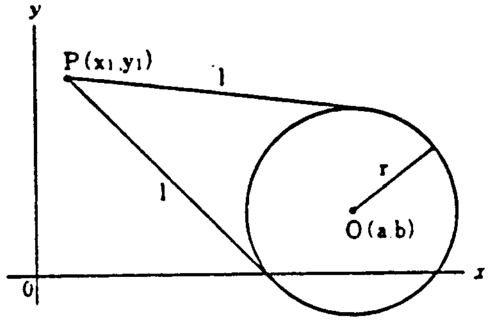
Determine the equation $(x-a)^2+(y-b)^2=r^2$ for the circle which passes through points P_1 (3, 6), P_2 (5, 4), P_3 (6, 2).

3 [EXE] 6 [EXE]	Circle (x1, y1), (x2, y2), (x3, y3) x2= 0 ?_	(P_1 coordinate input)
5 [EXE] 4 [EXE]	Circle (x1, y1), (x2, y2), (x3, y3) x3= 0 ?_	(P_2 coordinate input)
6 [EXE] 2 [EXE]	Circle (x−a)²+(y−b)²=r² a =−1.5	(a displayed following P_3 coordinate input)
[EXE]	Circle (x−a)²+(y−b)²=r² b =−0.5	(b display)
[EXE]	Circle (x−a)²+(y−b)²=r² r = 7.90569415	(r display)
[⇧]	Circle (x−a)²+(y−b)²=r² b =−0.5	(Redisplay of b)
[⇩]	Circle (x−a)²+(y−b)²=r² r = 7.90569415	(Redisplay of r)
[EXE]	Circle (x1, y1), (x2, y2), (x3, y3) x1= 3 ?_	(Return to initial display)

Here, the equation for the circle become $(x+1.5)^2+(y+0.5)^2=7.90569415^2$.
The result is displayed in the sequence of a, b, r, and the display can be scrolled to view following values using [⇩] (or [EXE]), and previous values using [⇧].

5560 LENGTH OF TANGENT LINES FROM A POINT TO A CIRCLE

Determines length l from point P (x_1, y_1) to a circle expressed by the equation $(x-a)^2 + (y-b)^2 = r^2$.



OPERATION

Key	Display
5560 LIB	(x−a)2+(y−b)2=r2.(x1.y1) a= 0 ?_

EXAMPLE .

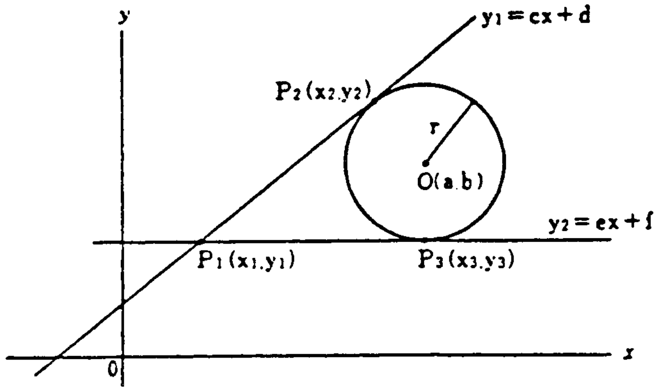
Determine the length l of a tangent line from point P (2, 5) to a circle with center point O (6, 2) and a radius of 4.

Key	Display	Note
6 EXE 2 EXE	(x−a)2+(y−b)2=r2.(x1.y1) r= 0 ?_	(Coordinates of circle's center point)
4 EXE	(x−a)2+(y−b)2=r2.(x1.y1) x1= 0 ?_	(Circle's radius)
2 EXE 5 EXE	(x−a)2+(y−b)2=r2.(x1.y1) l:length = 3	(Tangent line length display following input of point coordinates)
EXE	(x−a)2+(y−b)2=r2.(x1.y1) a= 6 ?_	(Return to initial display)

Here, the length of tangent line l is 3.
The message "not found" appears on the display when the coordinates of point P are within the circle.

5570 TANGENT LINE EQUATION

Determines the equations for two lines $y_1 = cx + d$, $y_2 = ex + f$, and their points of tangency P_2 (x_2, y_2), P_3 (x_3, y_3) from point P_1 (x_1, y_1) to circle O represented by the equation $(x-a)^2 + (y-b)^2 = r^2$.



OPERATION

5570 LIB	(x−a)2+(y−b)2=r2.(x1.y1) a= 0 ?_	

EXAMPLE

Determine the equations for tangent lines and points of tangency from point P_1 (1, 2) to a circle centered on point O (4, 3) with a radius of 2.

4 EXE 3 EXE	(x−a)2+(y−b)2=r2.(x1.y1) r= 0 ?_	(x, y coordinates of circle's center point)
2 EXE	(x−a)2+(y−b)2=r2.(x1.y1) x1= 0 ?_	(Circle's radius)
1 EXE 2 EXE	(x2.y2).y=cx+d : (x3.y3).y=ex+f x2 = 2.310102051	(Tangent point P_2 x-coordinate displayed following input of P_1 x, y coordinates)
EXE	(x2.y2).y=cx+d : (x3.y3).y=ex+f y2 = 4.069693846	(Display of tangent point P_2 y-coordinate)
EXE	(x2.y2).y=cx+d : (x3.y3).y=ex+f c = 1.579795897	(Display of line y_1 slope)
EXE	(x2.y2).y=cx+d : (x3.y3).y=ex+f d = 0.4202041029	(Display of line y_1 intercept)
EXE	(x2.y2).y=cx+d : (x3.y3).y=ex+f x3 = 3.289897949	(Display of tangent point P_3 x-coordinate)
EXE	(x2.y2).y=cx+d : (x3.y3).y=ex+f y3 = 1.130306154	(Display of tangent point P_3 y-coordinate)
EXE	(x2.y2).y=cx+d : (x3.y3).y=ex+f e =-0.3797958971	(Display of line y_2 slope)
EXE	(x2.y2).y=cx+d : (x3.y3).y=ex+f f = 2.379795897	(Display of line y_2 intercept)

Key	Display	
[↑]	(x2.y2).y=cx+d : (x3.y3).y=ex+f e =-0.3797958971	(Redisplay of line y_2 slope)
[↓]	(x2.y2).y=cx+d : (x3.y3).y=ex+f f = 2.379795897	(Redisplay of line y_2 intercept)
[EXE]	(x-a)²+(y-b)²=r².(x1.y1) a= 4 ?_	(Return to initial display)

Here, the two points of tangency are P_2 (2.310102051, 4.069693846), P_3 (3.289897949, 1.130306154). The equations for the lines which pass through these points are:

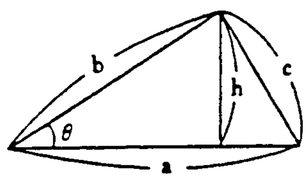
$y_2 = 1.579795897x + 0.4202041029$

$y_3 = -0.3797958971x + 2.379795897$

The result is displayed in the sequence of x_2, y_2, c, d, x_3, y_3, e, f, and the display can be scrolled to view following values using [↓] (or [EXE]) and previous values using [↑]. Corresponding values of c and e are omitted when the equations for the tangent lines are parallel.

5600 AREA OF A TRIANGLE

Determines the area (S) of a triangle using one of the three following formulas:



① $S = \frac{1}{2}ah$

② $S = ab \cdot \sin\frac{\theta}{2}$ (The result depends on the currently specified angle unit.)

③ $S = \sqrt{s(s-a)(s-b)(s-c)}$ $\left(s = \frac{1}{2}(a+b+c)\right)$

OPERATION

Key	Display
5600 [LIB]	Area (triangle) 1:ah/2 2:ab·sinθ/2 3:f(a.b.c)

EXAMPLE

Determine the area of triangle (a = 10, h = 5).

Key	Display	
1	ah/2 a:base h:height a= 0 ?_	
10 [EXE] 5 [EXE]	ah/2 a:base h:height Area = 25	(Triangle's base and height)
[EXE]	Area (triangle) 1:ah/2 2:ab·sinθ/2 3:f(a.b.c)	(Return to menu display)

Here, the area of the triangle is 25.

EXAMPLE

Determine the area of triangle (a = 10, b = 5, θ = 30 (DEG)).

2	ab·sinθ/2 a:b:side a= 0 ?_	
10 EXE 5 EXE 30 EXE	ab·sinθ/2 a:b:side Area = 12.5	(Two sides and included angle)
EXE	Area (triangle) 1:ah/2 2:ab·sinθ/2 3:f(a.b.c)	(Return to menu display)

Here, the area of the triangle is 12.5.

EXAMPLE

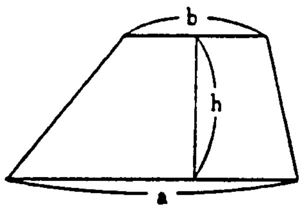
Determine the area of triangle (a = 5, b = 4, c = 3).

3	√(s(s−a)(s−b)(s−c)).s=(a+b+c)/2 a= 0 ?_	
5 EXE 4 EXE 3 EXE	√(s(s−a)(s−b)(s−c)).s=(a+b+c)/2 Area = 6	(Three sides)
EXE	Area (triangle) 1:ah/2 2:ab·sinθ/2 3:f(a.b.c)	(Return to menu display)

Here, the area of the triangle is 6.

5605 AREA OF A TRAPEZOID

Determines the area (S) of a trapezoid using the following formula:



$$S = (a+b)\cdot\frac{h}{2}$$

OPERATION

5605 LIB	(a+b)h/2 a:b:base h:height a= 0 ?_	

EXAMPLE

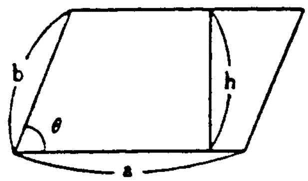
Determine the area of trapezoid (a = 10, b = 5, h = 4).

10 EXE 5 EXE 4 EXE	(a+b)h/2 a:b:base h:height Area = 30	(Bases, height)
EXE	(a+b)h/2 a:b:base h:height a= 10 ?_	(Return to initial display)

Here, the area of the trapezoid is 30.

5610 AREA OF A PARALLELOGRAM

Determines the area (S) of a parallelogram using one of the two following formulas:



S = ah

S = ab·sinθ (The result depends on the currently specified angle mode.)

OPERATION

5610 [LB]

```
Area (parallelogram)
1:ah        2:ab·sinθ
```

EXAMPLE

Determine the area of parallelogram (a = 10, h = 5).

Key operation	Display	
	ah a:base h:height a= 0 ?_	
10 [EXE] 5 [EXE]	ah a:base h:height Area = 50	(Base and height)
[EXE]	Area (parallelogram) 1:ah 2:ab·sinθ	(Return to initial display)

Here, the area of the parallelogram is 50.

EXAMPLE

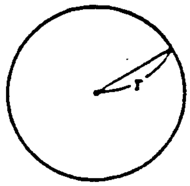
Determine the area of parallelogram (a = 10, b = 6, θ = 30 (DEG)).

Key operation	Display	
	ab·sinθ a:b:side a= 0 ?_	
10 [EXE] 6 [EXE] 30 [EXE]	ab·sinθ a:b:side Area = 30	(Two sides and included angle)
[AC]	Area (parallelogram) 1:ah 2:ab·sinθ	(Return to menu display)

Here, the area of the parallelogram is 30.

5615 AREA OF A CIRCLE

Determines the area (S) of a circle using the following formula:



$S = \pi r^2$

OPERATION

5615 [UB]

```
πr2          r:radius
r= 0  ?_
```

[EXAMPLE]

Determine the area of a circle with radius r=5.

5 [EXE]

```
πr2          r:radius
Area  =  78.53981634
```

(Circle's radius)

[EXE]

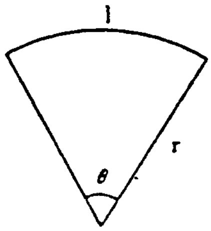
```
πr2          r:radius
r=  5  ?_
```

(Return to initial display)

Here, the area of the circle is 78.53981634.

5620 AREA OF A SECTOR

Determines the area (S) of a sector using one of the following formulas:



① $S = \frac{lr}{2}$

② $S = \pi r^2 \cdot \frac{\theta}{360}$ (Angle unit = degrees)

OPERATION

5620 [LIB]

```
Area (sector)
1: lr/2        2: πr²θ/360
```

[EXAMPLE]

Determine the area of sector (l = 6, r = 8).

Key operation	Display	Remarks
	lr/2 l: circular arc r: radius l= 0 ?_	
6 [EXE] 8 [EXE]	lr/2 l: circular arc r: radius Area = 24	(Arc's length and radius)
[EXE]	Area (sector) 1: lr/2 2: πr²θ/360	(Return to initial display)

Here, the area of the sector is 24.

[EXAMPLE]

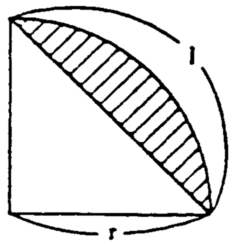
Determine the area of sector (r = 8, θ = 30 (DEG)).

Key operation	Display	Remarks
	πr²θ/360 r: radius θ: DEG r= 0 ?_	
8 [EXE] 30 [EXE]	πr²θ/360 r: radius θ: DEG Area = 16.75516082	(Radius and angle)
[EXE]	Area (sector) 1: lr/2 2: πr²θ/360	(Return to menu display)

Here, the area of the sector is 16.75516082.

5625 AREA OF A SEGMENT

Determines the area (S) of a segment using the following formula:



$S = (lr - r^2\sin(\frac{l}{r}))\frac{1}{2}$ (Angle unit = radians)

OPERATION

Key	Display	
5625 [LIB]	(lr-r²sin(l/r))/2 l:arc l= 0 ?_	

EXAMPLE

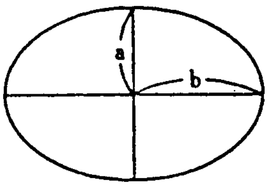
Determine the area of segment (l = 30, r = 10).

Key	Display	
30 [EXE]	(lr-r²sin(l/r))/2 r:radius r= 0 ?_	(Arc's length)
10 [EXE]	(lr-r²sin(l/r))/2 r:radius Area = 142.9439996	(Radius)
[EXE]	(lr-r²sin(l/r))/2 l:arc l= 30 ?_	(Return to initial display)

Here, the area of the segment is 142.9439996.

5630 AREA OF AN ELLIPSE

Determines the area (S) of an ellipse using the following formula:



$S = \pi ab$

OPERATION

5630 [LIB]

```
πab                 a:b:radius
a= 0 ?_
```

EXAMPLE

Determine the area of ellipse (a = 4, b = 6).

4 [EXE] 6 [EXE]

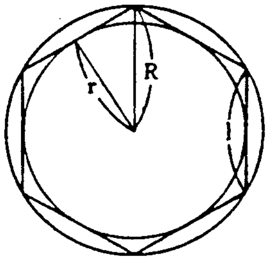
```
πab                 a:b:radius
Area = 75.39822369
```
(Radii)

```
πab                 a:b:radius
a= 4 ?_
```
(Return to initial display)

Here, the area of the ellipse is 75.39822369.

5635 AREA OF A POLYGON

Determines the area (S) of a polygon using one of the following formulas: Angle unit = DEG



① $S = f(n, r) = nr^2 \tan\frac{\pi}{n}$
② $S = f(n, R) = \frac{1}{2}nR^2 \sin\frac{2\pi}{n}$
③ $S = f(n, l) = \frac{1}{4}nl^2 \cot\frac{\pi}{n}$ • $\cot x = 1/\tan x$

* n indicates the number of sides in the polygon. This means that n = 6 for a regular hexagon.

OPERATION

5635 LIB	Area (polygon) 1: n・r−A 2: n・R−A 3: n・l−A	

EXAMPLE

Determine the area of regular hexagon (r = 5 (n = 6)).

1	Polygon n: number n= 0 ?_	
6 EXE 5 EXE	Polygon (n=6) r: inside Area = 86.60254038	(Specifies hexagon and radius of inscribed circle)
EXE	Area (polygon) 1: n・r−A 2: n・R−A 3: n・l−A	(Return to menu display)

Here, the area of the regular hexagon is 86.60254038.

EXAMPLE

Determine the area of regular hexagon (R = 6).

2	Polygon n: number n= 0 ?_	
6 EXE 6 EXE	Polygon (n=6) R: outside Area = 93.53074361	(Specifies hexagon and radius of inscribed circle.)
EXE	Area (polygon) 1: n・r−A 2: n・R−A 3: n・l−A	(Return to menu display)

Here, the area of the regular hexagon is 93.53074361.

EXAMPLE

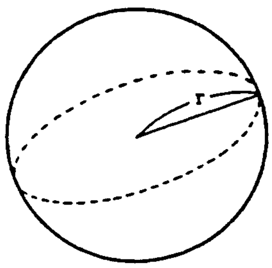
Determine the area of regular hexagon (l=4).

3	Polygon n:number n= 0 ?_	
6 EXE 4 EXE	Polygon (n=6) l:side Area = 41.56921938	(Specifies hexagon and one side)
EXE	Area (polygon) 1:n·r-A 2:n·R-A 3:n·l-A	(Return to menu display)

Here, the area of the regular hexagon is 41.56921938.

5650 SURFACE AREA OF A SPHERE

Determines the surface area (S) of a sphere using the following formula:



r : Radius of sphere

$S = f(r) = 4\pi r^2$

OPERATION

5650 LIB	4πr2 r:radius r= 0 ?_

EXAMPLE

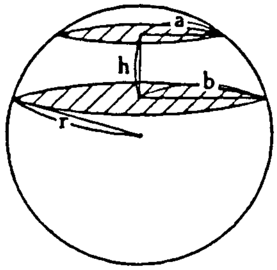
Determine the surface area of sphere r=8.

8 EXE	4πr2 r:radius Surface = 804.2477193	(Radius)
EXE	4πr2 r:radius r= 8 ?_	(Return to initial display)

Here, the surface area of the sphere is 804.2477193.

5655 SURFACE AREA OF A ZONE OF A SPHERE

Determines the surface area (S) of a zone of a sphere using the following formula:



$S = f(r, h, a, b) = 2\pi rh + \pi(a^2 + b^2)$

OPERATION

5655 [LIB]

```
2πrh+π(a²+b²)      h:height
h=  0  ?_
```

EXAMPLE

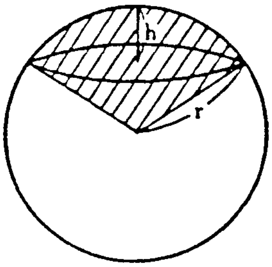
Determine the surface area of zone h = 2, a = 4, b = 5, r = 6 of a sphere.

Key	Display	
2 [EXE]	2πrh+π(a²+b²) a:b:r:radius a= 0 ?_ 2πrh+	(Height)
4 [EXE]	2πrh+π(a²+b²) a:b:r:radius b= 0 ?_	(Upper radius)
5 [EXE]	2πrh+π(a²+b²) a:b:r:radius r= 0 ?_	(Lower radius)
6 [EXE]	2πrh+π(a²+b²) a:b:r:radius Surface = 204.2035225	(Sphere radius)
[EXE]	2πrh+π(a²+b²) h:height h= 2 ?_	(Return to initial display)

Here, the surface area of the zone is 204.2035225.

5660 SURFACE AREA OF A SPHERICAL SECTOR

Determines the surface area (S) of a spherical sector using the following formula:



r : Radius
h : Height

$S = f(r,h) = 2\pi rh + \pi ar$ $(a = \sqrt{h(2r-h)})$

OPERATION

5660 [LIB]

```
2πrh+πar・a=√(h(2r−h))    r:radius
r=  0  ?_
```

EXAMPLE

Determine the surface area of spherical sector (r = 5, h = 3).

5 [EXE]

```
2πrh+πar.a=√(h(2r−h))    h:height
h=  0  ?_
```

(Radius)

3 [EXE]

```
2πrh+πar.a=√(h(2r−h))    h:height
Surface  =  166.2307103
```

(Height)

[EXE]

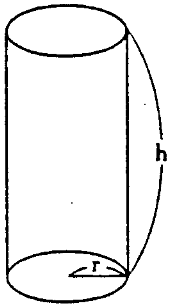
```
2πrh+πar・a=√(h(2r−h))    r:radius
r=  5  ?_
```

(Return to initial display)

Here, the surface area of the spherical sector is 166.2307103.

5665 SURFACE AREA OF A CIRCULAR CYLINDER

Determines the surface area (S) of a circular cylinder using the following formula:



r : Radius
h : Height

$S = f(r, h) = 2\pi rh + 2\pi r^2$

OPERATION

5665 LIB

```
2πrh+2πr²      r:radius   h:height
r=  0  ?_
```

EXAMPLE

Determine the surface area of circular cylinder (r = 6, h = 10).

6 EXE (Radius)

```
2πrh+2πr²      r:radius   h:height
h=  0  ?_
```

10 EXE (Height)

```
2πrh+2πr²      r:radius   h:height
Surface  =  603.1857895
```

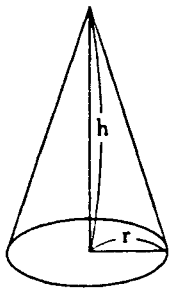
EXE (Return to initial display)

```
2πrh+2πr²      r:radius   h:height
r=  6  ?_
```

Here, the surface area of the circular cylinder is 603.1857895.

5670 SURFACE AREA OF A CIRCULAR CONE

Determines the surface area (S) of a circular cone using the following formula:



r : Radius
h : Height

$S = f(r, h) = \pi r \sqrt{(r^2 + h^2)} + \pi r^2$

OPERATION

5670 [LIB]

```
πr√(r²+h²)+πr²     r:radius
r= 0 ?_
```

EXAMPLE

Determine the surface area of circular cone (r = 6, h = 10).

6 [EXE] (Radius)

```
πr√(r²+h²)+πr²     h:height
h= 0 ?_
```

10 [EXE] (Height)

```
πr√(r²+h²)+πr²     h:height
Surface = 332.9190432
```

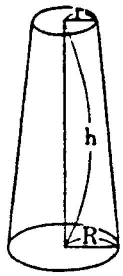
[EXE] (Return to initial display)

```
πr√(r²+h²)+πr²     r:radius
r= 6 ?_
```

Here, the surface area of the circular cone is 332.9190432.

5675 SURFACE AREA OF A FRUSTUM OF A CIRCULAR CONE

Determines the surface area (S) of a frustum of a circular cone using the following formula:



r : Upper radius
R: Lower radius
h : Height

$$S = f(r, R, h) = \pi (R + r)\sqrt{h^2 + (R - r)^2} + \pi (R^2 + r^2)$$

OPERATION

5675 [LIB]	π(R+r)√(h²+(R−r)²)+π(R²+r²) r= 0 ?_	

EXAMPLE

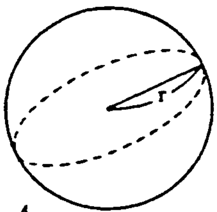
Determine the surface area of the frustum of circular cone (r = 4, R = 6, h = 10).

4 [EXE]	π(R+r)√(h²+(R−r)²)+π(R²+r²) R= 0 ?_	(Upper radius)
6 [EXE]	π(R+r)√(h²+(R−r)²)+π(R²+r²) h= 0 ?_	(Lower radius)
10 [EXE]	π(R+r)√(h²+(R−r)²)+π(R²+r²) Surface = 483.7436629	(Height)
[EXE]	π(R+r)√(h²+(R−r)²)+π(R²+r²) r= 4 ?_	(Return to initial display)

Here, the surface area of the frustum of the circular cone is 483.7436629.

5700 VOLUME OF A SPHERE

Determines the volume (V) of a sphere using the following formula:



r : Radius

$$V = f_{(r)} = \frac{4}{3}\pi r^3$$

OPERATION

5700 [LIB]

```
4πr³/3                r:radius
r=  0  ?_
```

EXAMPLE

Determine the volume of sphere (r = 6).

6 [EXE]

```
4πr³/3                r:radius
Volume  =  904.7786842
```

(Radius)

[EXE]

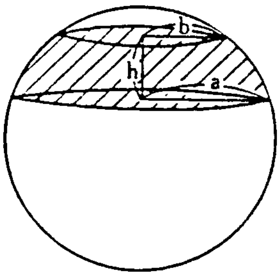
```
4πr³/3                r:radius
r=  6  ?_
```

(Return to initial display)

Here, the volume of the sphere is 904.7786842.

5705 VOLUME OF THE ZONE OF A SPHERE

Determines the volume (V) of the zone of a sphere using the following formula:



a : Lower radius
b : Upper radius
h : Height

$$V=f(a,b,h)=\frac{1}{6}\pi h\,(3a^2+3b^2+h^2)$$

OPERATION

5705 [LIB]	πh(3a²+3b²+h²)/6　　a:b:radius a=　0　?_	

EXAMPLE

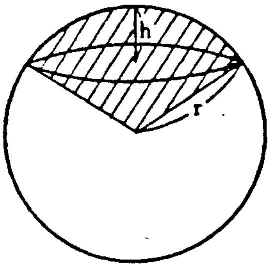
Determine the volume of the zone of sphere (a = 6, b = 4, h = 2).

6 [EXE]	πh(3a²+3b²+h²)/6　　a:b:radius b=　0　?_	(Lower radius)
4 [EXE]	πh(3a²+3b²+h²)/6　　h:height h=　0　?_	(Upper radius)
2 [EXE]	πh(3a²+3b²+h²)/6　　h:height Volume = 167.5516082	(Height)
[EXE]	πh(3a²+3b²+h²)/6　　a:b:radius a=　6　?_	*(Return to initial display)*

Here, the volume of the zone of the sphere is 167.5516082.

5710 VOLUME OF A SPHERICAL SECTOR

Determines the volume (V) of a spherical sector using the following formula:



r : Radius
h : Height

$V = f(r, h) = \frac{2}{3}\pi r^2 h$

OPERATION

5710 [LB]

```
2xr²h/3        r:radius    h:height
r=  0  ?_
```

EXAMPLE

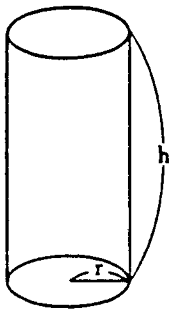
Determine the volume of spherical sector (r=6, h=2).

Key	Display	
6 [EXE]	2xr²h/3 r:radius h:height h= 0 ?_	(Radius)
2 [EXE]	2xr²h/3 r:radius h:height Volume = 150.7964474	(Height)
[EXE]	2xr²h/3 r:radius h:height r= 6 ?_	(Return to initial display)

Here, the volume of the spherical sector is 150.7964474.

5715 VOLUME OF A CIRCULAR CYLINDER

Determines the volume (V) of a circular cylinder using the following formula:



r : Radius
h : Height

$V = f(r, h) = \pi r^2 h$

OPERATION

5715 [LIB]

```
πr²h            r:radius    h:height
r=  0  ?_
```

EXAMPLE

Determine the volume of circular cylinder (r = 5, h = 10).

5 [EXE]

```
πr²h            r:radius    h:height
h=  0  ?_
```

(Radius)

10 [EXE]

```
πr²h            r:radius    h:height
Volume  =  785.3981634
```

(Height)

[EXE]

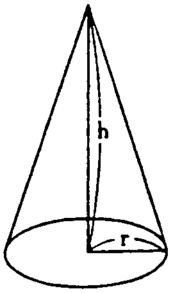
```
πr²h            r:radius    h:height
r=  5  ?_
```

(Return to initial display)

Here, the volume of the circular cylinder is 785.3981634.

5720 VOLUME OF A CIRCULAR CONE

Determines the volume (V) of a circular cone using the following formula:



r : Radius
h : Height

$$V = f(r, h) = \frac{1}{3}\pi r^2 h$$

OPERATION

5720 [LB]

```
πr²h/3         r:radius   h:height
r=  0  ?_
```

EXAMPLE

Determine the volume of circular cone (r = 5, h = 10).

5 [EXE]

```
πr²h/3         r:radius   h:height
h=  0  ?_
```

(Radius)

10 [EXE]

```
πr²h/3         r:radius   h:height
Volume  =  261.7993878
```

(Height)

[EXE]

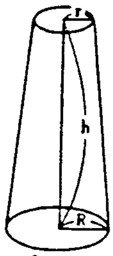
```
πr²h/3         r:radius   h:height
r=  5  ?_
```

(Return to initial display)

Here, the volume of the circular cone is 261.7993878.

5725 VOLUME OF THE FRUSTUM OF A CIRCULAR CONE

Determines the volume (V) of the frustum of a circular cone using the following formula:



r : Upper radius
R: Lower radius
h : Height

$$V=f(r,R,h)=\frac{1}{3}\pi h(r^2+rR+R^2)$$

OPERATION

5725 [LIB]	πh(r²+rR+R²)/3 r:R:radius r= 0 ?_

EXAMPLE

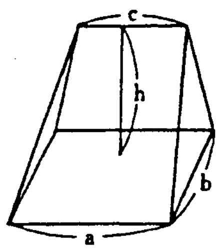
Determine the volume of the frustum of circular cone (r=4, R=6, h=10).

4 [EXE]	πh(r²+rR+R²)/3 r:R:radius R= 0 ?_	(Upper radius)
6 [EXE]	πh(r²+rR+R²)/3 h:height h= 0 ?_	(Lower radius)
10 [EXE]	πh(r²+rR+R²)/3 h:height Volume = 795.8701389	(Height)
[EXE]	πh(r²+rR+R²)/3 r:R:radius r= 4 ?_	(Return to initial display)

Here, the volume of the frustum of the circular cone is 795.8701389.

5730 VOLUME OF A WEDGE

Determines the volume (V) of a wedge using the following formula:



a : b : c : Sides
h : Height

$V = f(a, b, c, h) = \frac{1}{6} bh(2a + c)$

OPERATION

5730 LIB

```
bh(2a+c)/6   a:b:c:edge  h:height
a=  0  ?_
```

EXAMPLE

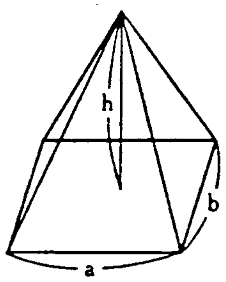
Determine the volume of wedge (a = 6, b = 8, c = 4, h = 5).

Key	Display	
6 EXE	`bh(2a+c)/6   a:b:c:edge  h:height` `b=  0  ?_`	(One side a)
8 EXE	`bh(2a+c)/6   a:b:c:edge  h:height` `c=  0  ?_`	(One side b)
4 EXE	`bh(2a+c)/6   a:b:c:edge  h:height` `h=  0  ?_`	(One side c)
5 EXE	`bh(2a+c)/6   a:b:c:edge  h:height` `Volume  =  106.6666667`	(Height)
EXE	`bh(2a+c)/6   a:b:c:edge  h:height` `a=  6  ?_`	(Return to initial display)

Here, the volume of the wedge is 106.6666667.

5735 VOLUME OF A PYRAMID

Determines the volume (V) of a pyramid using the following formula:



a : b : Sides
h : Height

$V = f(a, b, h) = \frac{1}{3}abh$

OPERATION

5735 [LIB]

```
abh/3          a:b:edge   h:height
a=  0  ?_
```

EXAMPLE

Determine the volume of pyramid (a = 4, b = 5, h = 6).

4 [EXE] 5 [EXE]

```
abh/3          a:b:edge   h:height
h=  0  ?_
```

(Base dimensions)

6 [EXE]

```
abh/3          a:b:edge   h:height
Volume  =  40
```

(Height)

[EXE]

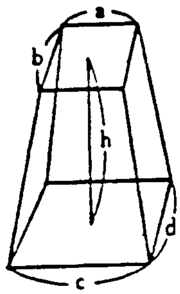
```
abh/3          a:b:edge   h:height
a=  4  ?_
```

(Return to initial display)

Here, the volume of the pyramid is 40.

5740 VOLUME OF THE FRUSTUM OF A PYRAMID

Dermines the volume (V) of the frustum of a pyramid using the following formula:



a : b : Upper sides
c : d : Lower sides
h : Height

$V = f(a, b, c, d, h) = \frac{h}{3}(ab + cd + \sqrt{abcd})$

OPERATION

5740 LIB	h(ab+cd+√(abcd))/3 a:b:c:d:edge a= 0 ?_	

EXAMPLE

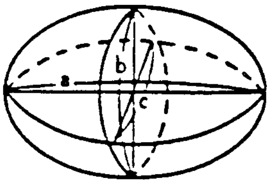
Determine the volume of the frustum of pyramid (a = 3, b = 4, c = 6, d = 8, h = 12).

3 EXE 4 EXE	h(ab+cd+√(abcd))/3 a:b:c:d:edge c= 0 ?_	(Two sides of a & b)
6 EXE 8 EXE	h(ab+cd+√(abcd))/3 h:height h= 0 ?_	(Two sides of c & d)
12 EXE	h(ab+cd+√(abcd))/3 h:height Volume = 336	(Height)
EXE	h(ab+cd+√(abcd))/3 a:b:c:d:edge a= 3 ?_	(Return to initial display)

Here, the volume of the frustum of the pyramid is 336.

5745 VOLUME OF AN ELLIPSOID

Determines the volume (V) of an ellipsoid using the following formula:



a:b:c : Raius

$V = f(a \cdot b \cdot c) = \frac{4}{3}\pi abc$

OPERATION

5745 [LB]	4 x a b c / 3 a : b : c : r a d i u s a = 0 ?_	

EXAMPLE

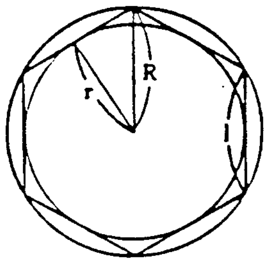
Determine the volume of ellipsoid (a = 10, b = 6, c = 5).

10 [EXE]	4 x a b c / 3 a : b : c : r a d i u s b = 0 ?_	(Radius a)
6 [EXE]	4 x a b c / 3 a : b : c : r a d i u s c = 0 ?_	(Radius b)
5 [EXE]	4 x a b c / 3 a : b : c : r a d i u s Volume = 1256.637061	(Radius c)
[EXE]	4 x a b c / 3 a : b : c : r a d i u s a = 10 ?_	(Return to initial display)

Here, the volume of the ellipsoid is 1256.637061.

5750 INSCRIBED CIRCLE AND CIRCUMSCRIBED CIRCLES OF A POLYGON

Determines the radius of the inscribed circle and the circumscribed circle and the length of one side of a polygon from a regular polygon's area.



Angle unit used is the DEG mode.

OPERATION

5750 [LB]

```
Polygon (r.R.l)     A:area
A= 0 ?_
```

EXAMPLE

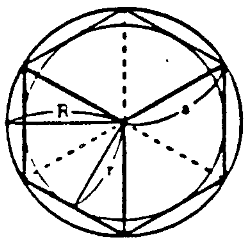
Determine the radius of the inscribed circle and circumscribed circle and one side of a regular pentagon with an area of 450.

Key	Display	Note
450 [EXE]	Polygon (r.R.l) n:number n= 3 ?_	(Enter area A)
[EXE]	Polygon (r.R.l) r:inside = 11.12988647	(Radius of inscribed circle displayed when the number of sides of polygon is input)
[EXE]	Polygon (r.R.l) R:outside = 13.75729626	(Display of circumscribed circle radius)
[EXE]	Polygon (r.R.l) l:side = 16.17267171	(Display of one side of pentagon)
[EXE]	Polygon (r.R.l) A:area A= 450 ?_	(Return to initial display)

ere, the radius of the inscribed circle is 11.12988647, the radius of the circumscribed circle 13.75729626, and one side of the regular pentagon is 16.17267171.

5760 REGULAR POLYHEDRON

Determines four the following parameters for a regular polyhedron when one parameter is input:



a : Length of one side
r : Radius of inscribed sphere
R : Radius of circumscribed sphere
S : Surface area
V : Volume

OPERATION

5760 [LIB]

Select number of face 1:4f 2:6f 3:8f 4:12f 5:20f	

One of the following regular polyhederons can be selected from the menu illustrated above.

EXAMPLE

Find the length of one side (a), radius of inscribed sphere (r), radius of circumscribed sphere (R), and volume (V) of a regular octahedron with a surface area of 100cm^2.

Key	Display	Note
3	Select input data 1:a 2:r 3:R 4:S 5:V	(3 : 8f selection from menu)
4	8(octahedron) S:surface S= 0 ?_	(4 : S selection from menu)
100 [EXE]	8(octahedron) 3.12e.6v a:edge = 5.372849659	(Entry of surface area displays length of one side)
[EXE]	8(octahedron) 3.12e.6v r:inside = 2.193456688	(Radius of inscribed circle display)
[EXE]	8(octahedron) 3.12e.6v R:outside = 3.799178428	(Radius of circumscribed circle display)
[EXE]	8(octahedron) 3.12e.6v V:volume = 73.11522294	(Volume display)
[↑]	8(octahedron) 3.12e.6v R:outside = 3.799178428	(Redisplay of radius of circumscribed circle)
[↓]	8(octahedron) 3.12e.6v V:volume = 73.11522294	(Redisplay of volume)
[EXE]	Select number of face 1:4f 2:6f 3:8f 4:12f 5:20f	(Return to menu display)

Here, the following data is calculated for the regular octahedron:

Length of one side	:	Approximately 5.37cm
Radius of inscribed circle	:	Approximately 2.19cm
Radius of circumscribed circle	:	Approximately 3.80cm
Surface area	:	$100cm^2$
Volume	:	Approximately $73.12cm^3$

EXAMPLE

Find the radius of inscribed sphere (r), radius of circumscribed sphere (R), surface area (S) and volume (V) of a regular icosahedron which has a length of one side of 5cm.

Key operation	Display	Comment
5 EXE	Select input data 1:a 2:r 3:R 4:S 5:V	(5 : 20f selection from menu)
1 EXE	20(icosahedron) a:edge a= 0 ?_	(1 : a selection from menu)
5 EXE	20(icosahedron) 3.30e.12v r:inside = 3.77880657	(Entry of length of one side displays radius of inscribed circle)
EXE	20(icosahedron) 3.30e.12v R:outside = 4.755282581	(Radius of circumscribed circle display)
EXE	20(icosahedron) 3.30e.12v S:surface = 216.5063509	(Surface area display)
EXE	20(icosahedron) 3.30e.12v V:volume = 272.7118738	(Volume display)
EXE	Select number of face 1:4f 2:6f 3:8f 4:12f 5:20f	(Return to menu display)

Here, the following data is calculated for the regular icosahedron:

Length of one side	:	5cm
Radius of inscribed circle	:	Approximately 3.78cm
Radius of circumscribed circle	:	Approximately 4.76cm
Surface area	:	Approximately $216.51cm^2$
Volume	:	Approximately $272.71cm^3$

5800 FACTORIZATION

Displays the following 23 factorized 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. $a^2\pm 2ab+b^2=(a\pm b)^2$
6. $a^3\pm 3a^2b+3ab^2\pm b^3=(a\pm b)^3$
7. $(a\pm b)^2\mp 4ab=(a\mp b)^2$
8. $a^2+b^2+c^2+2bc+2ca+2ab=(a+b+c)^2$
9. $a^4+a^2b^2+b^4=(a^2+ab+b^2)(a^2-ab+b^2)$
10. $a^3+b^3+c^3-3abc=(a+b+c)(a^2+b^2+c^2-bc-ca-ab)$
11. $(ac-bd)^2+(ad+bc)^2=(a^2+b^2)(c^2+d^2)$
12. $(ac+bd)^2+(ad-bc)^2=(a^2+b^2)(c^2+d^2)$
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. $a^2(b-c)+b^2(c-a)+c^2(a-b)=-(b-c)(c-a)(a-b)$
16. $(b-c)^3+(c-a)^3+(a-b)^3=3(b-c)(c-a)(a-b)$
17. $a^4+b^4+c^4-2b^2c^2-2c^2a^2-2a^2b^2=(a+b+c)(b-c-a)(c-a-b)(a-b-c)$
18. $x^2+(a+b)x+ab=(x+a)(x+b)$
19. $x^3+(a+b+c)x^2+(bc+ca+ab)x+abc=(x+a)(x+b)(x+c)$
20. $a^2-b^2-c^2-2bc=(a+b+c)(a-b-c)$
21. $(a+b-c)(bc+ca+ab)-abc=(b+c)(c+a)(a+b)$
22. $(a+b+c)^3-(a^3+b^3+c^3)=3(b+c)(c+a)(a+b)$
23. $a^3(b-c)+b^3(c-a)+c^3(a-b)=-(b-c)(c-a)(a-b)(a+b+c)$

OPERATION

5800 [UB]	`a2-b2 [1]` `=(a+b)(a-b)`	

[↓] (or [EXE]) scrolls to the following formula, [↑] to the previous formula, [⊟] to the first formula, and [⊟] to the last (23rd) formula.

EXAMPLE

Display a desired factorized formula.

[↓]	`a3±b3 [2]` `=(a±b)(a2∓ab+b2)`	(Formula 2)
[↓] [↓] [↓]	`a2±2ab+b2 [5]` `=(a±b)2`	(Formula 5)
[↓] [↓] [↓] [↓] [↓]	`a3+b3+c3-3abc [10]` `=(a+b+c)(a2+b2+c2-bc-ca-ab)`	(Formula 10)
[↑] [↑] [↑]	`(a±b)2∓4ab [7]` `=(a∓b)2`	(Formula 7)
[⊟]	`a2-b2 [1]` `=(a+b)(a-b)`	(Formula 1)
[⊟]	`a3(b-c)+b3(c-a)+c3(a-b) [23]` `=-(b-c)(c-a)(a-b)(a+b+c)`	(Formula 23)
[↑] [↑] [↑] [↑]	`x3+(a+b+c)x2+(bc+ca+ab)x+abc[19]` `=(x+a)(x+b)(x+c)`	(Formula 19)

5810 TRIGONOMETRIC FUNCTIONS

Displays the following 38 trigonometric equations:

1. $\sin^2\theta+\cos^2\theta=1$
2. $1+\tan^2\theta=\sec^2\theta$
3. $1+\cot^2\theta=\mathrm{cosec}^2\theta$
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. $\tan(\alpha\pm\beta)=\dfrac{\tan\alpha\pm\tan\beta}{1\mp\tan\alpha\cdot\tan\beta}$
7. $\cot(\alpha\pm\beta)=\dfrac{\cot\alpha\cdot\cot\beta\mp1}{\cot\beta\pm\cot\alpha}$
8. $\sin2\theta=2\sin\theta\cdot\cos\theta$
9. $\cos2\theta=\cos^2\theta-\sin^2\theta$
10. $\cos2\theta=1-2\sin^2\theta$
11. $\cos2\theta=2\cos^2\theta-1$
12. $\tan2\theta=\dfrac{2\tan\theta}{1-\tan^2\theta}$
13. $\sin\dfrac{\theta}{2}=\pm\sqrt{\dfrac{1-\cos\theta}{2}}$
14. $\cos\dfrac{\theta}{2}=\pm\sqrt{\dfrac{1+\cos\theta}{2}}$
15. $\tan\dfrac{\theta}{2}=\pm\sqrt{\dfrac{1-\cos\theta}{1+\cos\theta}}$
16. $\tan\dfrac{\theta}{2}=\dfrac{1-\cos\theta}{\sin\theta}$
17. $\tan\dfrac{\theta}{2}=\dfrac{\sin\theta}{1+\cos\theta}$
18. $\tan\dfrac{\theta}{2}=\mathrm{cosec}\,\theta-\cot\theta$
19. $\cot\dfrac{\theta}{2}=\pm\sqrt{\dfrac{1-\cos\theta}{1-\cos\theta}}$
20. $\cot\dfrac{\theta}{2}=\dfrac{\sin\theta}{1-\cos\theta}$
21. $\cot\dfrac{\theta}{2}=\dfrac{1+\cos\theta}{\sin\theta}$
22. $\cot\dfrac{\theta}{2}=\mathrm{cosec}\,\theta+\cot\theta$
23. $\sin3\theta=3\sin\theta-4\sin^3\theta$
24. $\cos3\theta=4\cos^3\theta-3\cos\theta$
25. $\tan3\theta=\dfrac{3\tan\theta-\tan^3\theta}{1-3\tan^2\theta}$
26. $2\sin\alpha\cdot\cos\beta=\sin(\alpha+\beta)+\sin(\alpha-\beta)$
27. $2\cos\alpha\cdot\sin\beta=\sin(\alpha+\beta)-\sin(\alpha-\beta)$
28. $2\cos\alpha\cdot\cos\beta=\cos(\alpha+\beta)+\cos(\alpha-\beta)$
29. $2\sin\alpha\cdot\sin\beta=-(\cos(\alpha+\beta)-\cos(\alpha-\beta))$
30. $\sin\alpha+\sin\beta=2\sin\left(\dfrac{\alpha+\beta}{2}\right)\cdot\cos\left(\dfrac{\alpha-\beta}{2}\right)$
31. $\sin\alpha-\sin\beta=2\cos\left(\dfrac{\alpha+\beta}{2}\right)\cdot\sin\left(\dfrac{\alpha-\beta}{2}\right)$
32. $\cos\alpha+\cos\beta=2\cos\left(\dfrac{\alpha+\beta}{2}\right)\cdot\cos\left(\dfrac{\alpha-\beta}{2}\right)$
33. $\cos\alpha-\cos\beta=-2\sin\left(\dfrac{\alpha+\beta}{2}\right)\cdot\sin\left(\dfrac{\alpha-\beta}{2}\right)$
34. $\tan(45^\circ\pm\dfrac{\theta}{2})=\sec\theta\pm\tan\theta$
35. $\tan(45^\circ\pm\dfrac{\theta}{2})=\dfrac{1\pm\sin\theta}{\cos\theta}$
36. $\tan(45^\circ\pm\dfrac{\theta}{2})=\cot(45^\circ\mp\dfrac{\theta}{2})$
37. $\tan(45^\circ+\theta)=\dfrac{1+\tan\theta}{1-\tan\theta}$
38. $\cot(45^\circ-\theta)=\dfrac{1+\cot\theta}{1-\cot\theta}$

OPERATION

5810 [LIB]

```
sin²θ+cos²θ              [1]
=1
```

[⇩] (or [EXE]) scrolls to the following equation [⇧] to the previous equation, [⇦] to the first equation, and [⇨] to the last (38th) equation.

EXAMPLE

Display a desired trigonometric equation.

[⇩]	1+tan²θ [2] =sec²θ	(Equation 2)
[⇩] [⇩] [⇩] [⇩]	tan(α±β) [6] =(tanα±tanβ)/(1∓tanα·tanβ)	(Equation 6)
[⇩] [⇩] [⇩]	cos2θ [9] =cos²θ−sin²θ	(Equation 9)
[⇧] [⇧] [⇧] [⇧]	cos(α±β) [5] =cosα·cosβ∓sinα·sinβ	(Equation 5)
[⇧] [⇧]	1+cot²θ [3] =cosec²θ	(Equation 3)
[⇨]	cot(45°−θ) [38] =(1+cotθ)/(1−cotθ)	(Equation 38)
[⇦]	sin²θ+cos²θ [1] =1	(Equation 1)

5820 DIFFERENTIALS

Displays the following 38 differential equation:

1.	$y=c$	$y'=0$
2.	$y=x^n$	$y'=nx^{n-1}$
3.	$y=x$	$y'=1$
4.	$y=\frac{1}{x}$	$y'=-\frac{1}{x^2}$
5.	$y=\sqrt{x}$	$y'=\frac{1}{2\sqrt{x}}$
6.	$y=a^x$	$y'=a^x\log a$
7.	$y=e^x$	$y'=e^x$
8.	$y=e^{nx}$	$y'=ne^{nx}$
9.	$y=\log x$	$y'=\frac{1}{x}$
10.	$y=x^x$	$y'=x^x(\log x+1)$
11.	$y=\sin x$	$y'=\cos x$
12.	$y=\cos x$	$y'=-\sin x$
13.	$y=\tan x$	$y'=\sec^2 x$
14.	$y=\cot x$	$y'=-\mathrm{cosec}^2 x$
15.	$y=\sec x$	$y'=\sec x\cdot\tan x$
16.	$y=\mathrm{cosec}\, x$	$y'=-\mathrm{cosec}\, x\cdot\cot x$
17.	$y=\sin ax$	$y'=a\cdot\cos ax$
18.	$y=\cos ax$	$y'=-a\cdot\sin ax$
19.	$y=\tan ax$	$y'=a\cdot\sec^2 ax$
20.	$y=\cot ax$	$y'=-a\cdot\mathrm{cosec}^2 ax$
21.	$y=\sin^{-1}x$	$y'=\frac{1}{\sqrt{1-x^2}}$ $(\lvert y\rvert<\frac{\pi}{2})$
22.	$y=\cos^{-1}x$	$y'=-\frac{1}{\sqrt{1-x^2}}$ $(0<y<\pi)$
23.	$y=\tan^{-1}x$	$y'=\frac{1}{1+x^2}$ $(\lvert y\rvert<\frac{\pi}{2})$
24.	$y=\cot^{-1}x$	$y'=-\frac{1}{1+x^2}$ $(\lvert y\rvert<\frac{\pi}{2})$
25.	$y=\sec^{-1}x$	$y'=\frac{1}{x\sqrt{x^2-1}}$ $(0<y<\pi, x^2>1)$
26.	$y=\mathrm{cosec}^{-1}x$	$y'=\frac{1}{x\sqrt{x^2-1}}$ $(\lvert y\rvert<\frac{\pi}{2}, x^2>1)$
27.	$y=\sinh x$	$y'=\cosh x$
28.	$y=\cosh x$	$y'=\sinh x$
29.	$y=\tanh x$	$y'=\mathrm{sech}^2 x$
30.	$y=\coth x$	$y'=-\mathrm{cosech}^2 x$
31.	$y=\mathrm{sech}\, x$	$y'=-\mathrm{sech}\, x\cdot\tanh x$
32.	$y=\mathrm{cosech}\, x$	$y'=-\mathrm{cosech}\, x\cdot\coth x$
33.	$y=\sinh^{-1}x$	$y'=\frac{1}{\sqrt{1+x^2}}$
34.	$y=\cosh^{-1}x$	$y'=\frac{1}{\sqrt{x^2-1}}$ $(y>0, x^2>1)$
35.	$y=\tanh^{-1}x$	$y'=\frac{1}{1-x^2}$ $(x^2<1)$
36.	$y=\coth^{-1}x$	$y'=\frac{1}{1-x^2}$ $(x^2>1)$
37.	$y=\mathrm{sech}^{-1}x$	$y'=-\frac{1}{x\sqrt{1-x^2}}$ $(0<x<1)$
38.	$y=\mathrm{cosech}^{-1}x$	$y'=-\frac{1}{x\sqrt{x^2+1}}$

OPERATION

5820 [LB]

```
y  =C                                   [1]
y' =0
```

[↓] (or [EXE]) scrolls to the following equation, [↑] to the previous equation, [⇐] to the first equation, and [⇒] to the last (38th) equation.

EXAMPLE

Display a desired differential equation.

[↓]	`y =xn` / `y' =nxn-1` [2]	(Equation 2)
[↓] [↓] [↓] [↓]	`y =ax` / `y' =axloga` [6]	(Equation 6)
[↓] [↓] [↓] [↓] [↓] [↓]	`y =cosx` / `y' =-sinx` [12]	(Equation 12)
[↑] [↑] [↑]	`y =logx` / `y' =1/x` [9]	(Equation 9)
[⇐]	`y =C` / `y' =0` [1]	(Equation 1)
[⇒]	`y =cosech-1x` / `y' =-1/(x√(x2+1))` [38]	(Equation 38)

5830 INTEGRATION

Displays the following 34 integration equation:

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

OPERATION

5830 [UB]

```
∫ dx                                  [1]
=x+C
```

⇓ (or [EXE]) scrolls to the following equation, [⇧] to the previous equation, [⊖] to the first equation, and [⊜] to the last (34th) equation.

EXAMPLE

Display a desired integration equation.

⇓

```
∫ xⁿ dx                               [2]
=xⁿ⁺¹/(n+1)+C               (n+1≠0)
```
(Equation 2)

⇓ [⇩] [⇩] [⇩]

```
∫ eⁿˣ dx                              [6]
=eⁿˣ/n+C
```
(Equation 6)

⇓ [⇩] [⇩]

```
∫ logx dx                             [9]
=x(logx-1)+C
```
(Equation 9)

⇑ [⇧]

```
∫ aˣ dx                               [7]
=aˣ/loga+C                  (a>0,a≠1)
```
(Equation 7)

⇒

```
∫ 1/(x²-a²) dx                       [34]
=1/2a·log((x-a)/(x+a))+C      (x>a)
```
(Equation 34)

⇑ [⇧] [⇧]

```
∫ 1/(a²+x²) dx                       [31]
=1/a·tan⁻¹(x/a)+C
```
(Equation 31)

⇐ [illegible]

```
∫ dx                                  [1]
=x+C
```
(Equation 1)

5840 LAPLACE TRANSFORMATION

Displays the following 36 Laplace transformation equations:

	F(p)	f(t)
(1)	$\frac{1}{p}$	1
(2)	$\frac{1}{p^2}$	t
(3)	$\frac{1}{p^n}$	$\frac{t^{n-1}}{(n-1)!}$ $(n=1,2,3,\cdots)$
(4)	$\frac{1}{p \pm m}$	$e^{\mp mt}$
(5)	$\frac{1}{p(p+m)}$	$\frac{1}{m}(1-e^{-mt})$
(6)	$\frac{1}{p^2(p+m)}$	$\frac{1}{m^2}(e^{-mt}+mt-1)$
(7)	$\frac{a}{p^2+a^2}$	$\sin at$
(8)	$\frac{p}{p^2+a^2}$	$\cos at$
(9)	$\frac{1}{p^2+a^2}$	$\frac{1}{a}\sin at$
(10)	$\frac{a}{p^2-a^2}$	$\sinh at$
(11)	$\frac{p}{p^2-a^2}$	$\cosh at$
(12)	$\frac{1}{p^2-a^2}$	$\frac{1}{a}\sinh at$
(13)	$\frac{1}{p(p^2+a^2)}$	$\frac{1}{a^2}(1-\cos at)$
(14)	$\frac{1}{p^2(p^2+a^2)}$	$\frac{1}{a^3}(at-\sin at)$
(15)	$\frac{1}{(p+m)(p+n)}$	$\frac{1}{n-m}(e^{-mt}-e^{-nt})$
(16)	$\frac{p}{(p+m)(p+n)}$	$\frac{1}{m-n}(me^{-mt}-ne^{-nt})$
(17)	$\frac{1}{(p+m)^2}$	te^{-mt}
(18)	$\frac{1}{(p+m)^n}$	$\frac{1}{(n-1)!}t^{n-1}e^{-mt}$ $(n=1,2,3\cdots)$
(19)	$\frac{p}{(p+m)^2}$	$e^{-mt}(1-mt)$
(20)	$\frac{1}{p(p+m)^2}$	$\frac{1}{m^2}(1-(1+mt)e^{-mt})$
(21)	$\frac{1}{p^2(p+m)^2}$	$\frac{t}{m^2}-\frac{2}{m^3}+\frac{2e^{-mt}}{m^3}-\frac{te^{-mt}}{m^2}$
(22)	$\frac{p+n}{(p+m)^2}$	$((n-m)t+1)e^{-mt}$
(23)	$\frac{1}{(p^2+a^2)^2}$	$\frac{1}{2a^3}(\sin at - at\cdot\cos at)$

No.	F(p)	f(t)
24	$\dfrac{p}{(p^2+a^2)^2}$	$\dfrac{t}{2a}\sin at$
25	$\dfrac{p^2}{(p^2+a^2)^2}$	$\dfrac{1}{2a}(\sin at+at\cdot\cos at)$
26	$\dfrac{p^2-a^2}{(p^2+a^2)^2}$	$t\cdot\cos at$
27	$\dfrac{1}{(p+m)^2+n^2}$	$\dfrac{1}{n}e^{-mt}\sin nt$
28	$\dfrac{p+m}{(p+m)^2+n^2}$	$e^{-mt}\cos nt$
29	$\dfrac{1}{p^4-a^4}$	$\dfrac{1}{2a^3}(\sinh at-\sin at)$
30	$\dfrac{p}{p^4-a^4}$	$\dfrac{1}{2a^2}(\cosh at-\cos at)$
31	$\dfrac{p^2}{p^4-a^4}$	$\dfrac{1}{2a}(\sinh at+\sin at)$
32	$\dfrac{p^3}{p^4-a^4}$	$\dfrac{1}{2}(\cosh at+\cos at)$
33	$\dfrac{p}{p^4+4a^4}$	$\dfrac{1}{2a^2}\cdot\sin at\cdot\sinh at$
34	$\dfrac{4a^3}{p^4+4a^4}$	$\sin at\cdot\cosh at-\cos at\cdot\sinh at$
35	$\dfrac{1}{p}\left(\dfrac{p-m}{p+m}\right)$	$-1+2e^{-mt}$
36	$\dfrac{1}{p^2}\left(\dfrac{p-m}{p+m}\right)$	$\dfrac{2}{m}-t-\dfrac{2}{m}e^{-mt}$

OPERATION

5840 [LIB]

```
F(p)=1/p                                  [1]
1
```

[⇓] or [EXE]) scrolls to the following equation, [⇑] to the previous equation, [⇐] to the first equation, and [⇒] to the last (36th) equation.

EXAMPLE

Display a desired Laplace transformation equation.

Key	Display	
⇓	F(p)=1/p² [2] t	(Equation 2)
⇓ ⇓ ⇓ ⇓ ⇓	F(p)=a/(p²+a²) [7] sinat	(Equation 7)
⇑ ⇑ ⇑	F(p)=1/(p±m) [4] e∓mt	(Equation 4)
⇓ ⇓	F(p)=1/(p²(p+m)) [6] 1/m²·(e-mt+mt-1)	(Equation 6)
⇒	F(p)=1/p²·((p-m)/(p+m)) [36] 2/m-t-2/m·e-mt	(Equation 36)
⇑ ⇑ ⇑ ⇑ ⇑	F(p)=p²/(p⁴-a⁴) [31] 1/2a·(sinhat+sinat)	(Equation 31)
⇐	F(p)=1/p [1] 1	(Equation 1)

5900 PERIODIC TABLE

Displays the periodic table of elements and atomic weight of selected elements.

• Periodic table of elements

								1 H 1.00794									Gas
																	0
1 A	2 A	Metallic elements										Non-metallic elements 3 B	4 B	5 B	6 B	7 B	2 He 4.00260
3 Li 6.941	4 Be 9.01218											5 B 10.81	6 C 12.011	7 N 14.0067	8 O 15.9994	9 F 18.998403	10 Ne 20.179
11 Na 22.98977	12 Mg 24.305	3 A	4 A	5 A	6 A	7 A	8			1 B	2 B	13 Al 26.98154	14 Si 28.0855	15 P 30.97376	16 S 32.06	17 Cl 35.453	18 Ar 39.948
19 K 39.0983	20 Ca 40.08	21 Sc 44.9559	22 Ti 47.88	23 V 50.9415	24 Cr 51.996	25 Mn 54.9380	26 Fe 55.847	27 Co 58.9332	28 Ni 58.69	29 Cu 63.546	30 Zn 65.38	31 Ga 69.72	32 Ge 72.59	33 As 74.9216	34 Se 78.96	35 Br 79.904	36 Kr 83.80
37 Rb 85.4678	38 Sr 87.62	39 Y 88.9059	40 Zr 91.22	41 Nb 92.9064	42 Mo 95.94	43 Tc (98)	44 Ru 101.07	45 Rh 102.9055	46 Pd 106.42	47 Ag 107.8682	48 Cd 112.41	49 In 114.82	50 Sn 118.69	51 Sb 121.75	52 Te 127.60	53 I 126.9045	54 Xe 131.29
55 Cs 132.9054	56 Ba 137.33	57 – 71 ■	72 Hf 178.49	73 Ta 180.9479	74 W 183.85	75 Re 186.207	76 Os 190.2	77 Ir 192.22	78 Pt 195.08	79 Au 196.9665	80 Hg 200.59	81 Tl 204.383	82 Pb 207.2	83 Bi 208.9804	84 Po (209)	85 At (210)	86 Rn (222)
87 Fr (223)	88 Ra 226.0254	89 – 103 ■■															

■ Lanthanides elements
■■ Actinides elements

■	57 La 138.9055	58 Ce 140.12	59 Pr 140.9077	60 Nd 144.24	61 Pm (145)	62 Sm 150.36	63 Eu 151.96	64 Gd 157.25	65 Tb 158.9254	66 Dy 162.50	67 Ho 164.9304	68 Er 167.26	69 Tm 168.9342	70 Yb 173.04	71 Lu 174.967
■■	89 Ac 227.0278	90 Th 232.0381	91 Pa 231.0359	92 U 238.0289	93 Np 237.0482	94 Pu (244)	95 Am (243)	96 Cm (247)	97 Bk (247)	98 Cf (251)	99 Es (252)	100 Fm (257)	101 Md (258)	102 No (259)	103 Lr (260)

Atomic weight (1)

Atomic number	Element	Symbol	Atomic weight
1	Hydrogen	H	1.00794 ± 7
2	Helium	He	4.00260
3	Lithium	Li	6.94_1
4	Beryllium	Be	9.01218
5	Boron	B	10.81
6	Carbon	C	12.011
7	Nitrogen	N	14.0067
8	Oxygen	O	15.999_4
9	Fluorine	F	18.998403
10	Neon	Ne	20.17_9
11	Sodium	Na	22.98977
12	Magnesium	Mg	24.305
13	Aluminium	Al	26.98154
14	Silicon	Si	28.085_5
15	Phosphorus	P	30.97376
16	Sulfur	S	32.06
17	Chlorine	Cl	35.453
18	Argon	Ar	39.948
19	Potassium	K	39.0983
20	Calcium	Ca	40.08
21	Scandium	Sc	44.9559
22	Titanium	Ti	47.8_8
23	Vanadium	V	50.9415
24	Chromium	Cr	51.996
25	Manganese	Mn	54.9380

• Atomic weight (2)

Atomic number	Element	Symbol	Atomic weight
26	Iron	Fe	55.84_7
27	Cobalt	Co	58.9332
28	Nickel	Ni	58.69
29	Copper	Cu	63.54_6
30	Zinc	Zn	65.38
31	Gallium	Ga	69.72
32	Germanium	Ge	72.5_9
33	Arsenic	As	74.9216
34	Selenium	Se	78.9_6
35	Bromine	Br	79.904
36	Krypton	Kr	83.80
37	Rubidium	Rb	85.467_8
38	Strontium	Sr	87.62
39	Yttrium	Y	88.9059
40	Zirconium	Zr	91.22
41	Niobium	Nb	92.9064
42	Molybdenum	Mo	95.94
43	Technetium	Tc	(98)
44	Ruthenium	Ru	101.0_7
45	Rhodium	Rh	102.9055
46	Palladium	Pd	106.42
47	Silver	Ag	107.868_2
48	Cadmium	Cd	112.41
49	Indium	In	114.82
50	Tin	Sn	118.6_9
51	Antimony	Sb	121.7_5
52	Tellurium	Te	127.60
53	Iodine	I	126.9045
54	Xenon	Xe	131.2_9
55	Caesium[3]	Cs	132.9054
56	Barium	Ba	137.33
57	Lanthanum	La	138.905_5
58	Cerium	Ce	140.12
59	Praseodymium	Pr	140.9077
60	Neodymium	Nd	144.2_4
61	Promethium	Pm	(145)
62	Samarium	Sm	150.3_6
63	Europium	Eu	151.96
64	Gadolinium	Gd	157.2_5
65	Terbium	Tb	158.9254

Atomic weight (3)

Atomic number	Element	Symbol	Atomic weight
66	Dysprosium	Dy	162.5_0
67	Holmium	Ho	164.9304
68	Erbium	Er	167.2_6
69	Thulium	Tm	168.9342
70	Ytterbium	Yb	173.0_4
71	Lutetium	Lu	174.967
72	Hafnium	Hf	178.4_9
73	Tantalum	Ta	180.9479
74	Tungsten	W	183.8_5
75	Rhenium	Re	186.207
76	Osmium	Os	190.2
77	Iridium	Ir	192.2_2
78	Platinum	Pt	195.0_8
79	Gold	Au	196.9665
80	Mercury	Hg	200.5_9
81	Thallium	Tl	204.383
82	Lead	Pb	207.2
83	Bismuth	Bi	208.9804
84	Polonium	Po	(209)
85	Astatine	At	(210)
86	Radon	Rn	(222)
87	Francium	Fr	(223)
88	Radium	Ra	226.0254
89	Actinium	Ac	227.0278
90	Thorium	Th	232.0381
91	Protactinium	Pa	231.0359
92	Uranium	U	238.0289
93	Neptunium	Np	237.0482
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

5900 [LIB]

```
   1a 2a 3a 4a 5a 6a 7a 8  8  8
1:H
```

Pressing [↓] displays the following periodic element, while pressing [↑] displays the previous periodic element. Pressing [⇦] displays groups 1a – 8, while [⇨] displays groups 1b – 7b and 0. Pressing [EXE] enters input stand by, during which inputting a symbol of an element displays its atomic weight.

EXAMPLE

Display the periodic table at a specific location and display the atomic weight of silicon.

Key	Display	Remark
[↓]	`   1a 2a 3a 4a 5a 6a 7a 8  8  8` `2:Li Be`	(#2 periodic element)
[↓] [↓] [↓]	`   1a 2a 3a 4a 5a 6a 7a 8  8  8` `5:Rb Sr Y  Zr Nb Mo Tc Ru Rh Pd`	(#5 periodic element)
[⇨]	`   1b 2b 3b 4b 5b 6b 7b 0` `5:Ag Cd In Sn Sb Te I  Xe`	(Group 1b – 7b and 0 with same period)
[↓] [↓] [↓]	`6:Lanthanoid: La Ce Pr Nd Pm Sm` `         Eu Gd Tb Dy Ho Er Tm Yb Lu`	(Lanthanoid)
[↓]	`7:Actinoid:Ac Th Pa U  Np Pu Am` `            Cm Bk Cf Es Fm Md No Lr`	(Actinoid)
[EXE]	`Input the symbol : _` `Push [EXE] then return`	
[S] [I] [EXE]	`Input the symbol : _` `Si Silicon 14(3-4b) 28.0855`	(Enter Si to display atomic weight of silicon)
[EXE]	`7:Actinoid:Ac Th Pa U  Np Pu Am` `            Cm Bk Cf Es Fm Md No Lr`	(Return to initial display)

5910 SCIENTIFIC CONSTANTS

Displays the following 22 scientific constants. Alphabet keys A~Z can be used to assign displayed values to numeric variables A through Z.

NAME & SYMBOL		VALUE	UNIT SI	UNIT CGS
Faraday constant	F	9.648456	10^{4} C·mol^{-1}	10^{3} emu·mol^{-1}
Gravitational constant	G	6.6720	10^{-11} m^{3}·s^{-2}·kg^{-1}	10^{-8} cm^{3}·s^{-2}·g^{-1}
Avogadro constant	N_A	6.022045	10^{23} mol^{-1}	10^{23} mol^{-1}
olar gas constant	R	8.31441	J·mol^{-1}·K^{-1}	10^{7} erg·mol^{-1}·K^{-1}
ydberg constant	R_∞	1.097373177	10^{7} m^{-1}	10^{5} cm^{-1}
olar volume of ideal gas at s.t.p.	Vm	22.41383	10^{-3} m^{3}·mol^{-1}	10^{3} cm^{3}·mol^{-1}
ohr radius	a_0	5.2917706	10^{-11} m	10^{-9} cm
eed of light in vacuum	c	299792458	m·s^{-1}	10^{2} cm·s^{-1}
ementary charge	e	1.6021892	10^{-19} C	10^{-20} emu
avitational acceleration	g	9.80665	m·s^{-2}	10^{2} cm·s^{-2}
anck constant	h	6.626176	10^{-34} J·s	10^{-27} erg·s
ltzmann constant	k	1.380662	10^{-23} J·K^{-1}	10^{-16} erg·K^{-1}
ectron rest mass	me	9.109534	10^{-31} kg	10^{-28} g
utron rest mass	mn	1.6749543	10^{-27} kg	10^{-24} g
oton rest mass	mp	1.6726485	10^{-27} kg	10^{-24} g
omic mass unit	u	1.6605655	10^{-27} kg	10^{-24} g
rmittivity of vacuum	$\epsilon 0$	8.854187818	10^{-12} F·m^{-1}	
rmeability of vacuum	$\mu 0$	12.5663706144	10^{-7} H·m^{-1}	
hr magneton	μb	9.274078	10^{-24} J·T^{-1}	10^{-21} erg·G^{-1}
ctron magnetic moment	μe	9.284832	10^{-24} J·T^{-1}	10^{-21} erg·G^{-1}
oton magnetic moment	μp	1.4106171	10^{-26} J·T^{-1}	10^{-23} erg·G^{-1}
fan-Boltzmann's constant	σ	5.67032	10^{-8} W·m^{-2}·K^{-4}	10^{-5} erg·s^{-1}·cm^{-2}·K^{-4}

ne values of these scientific constants are based on JIS Z-8202-1978 (JIS = Japan Indusal Standard).

OPERATION

5910 [LIB]

```
F   =9.648456×10⁴  [C·mol⁻¹]
>constant  :  key A-Z ?        [1]
```

Pressing [↓][EXE] displays the following constant, while pressing [↑] displays the previous constant. Pressing [⇦] displays in SI units, while pressing [⇨] displays in CGS unit. Alphabet keys A - Z can be used to assign displayed values to numeric variables A through Z.

EXAMPLE

Display the molar volume of ideal gas at s.t.p. and assign the value to numeric variable V in CGS units. Then display the Avogadro constant and assign the value to numeric variable N.

Key operation	Display	Comment
[↓] [↓] [↓] [↓] [↓]	Vm =22.41383x10^{-3} [$m^3\cdot mol^{-1}$] >constant : key A-Z ? [6]	(Molar volume of ideal gas at s.t.p.) (Displayed in CGS units)
[V]	[V] =22.41383x10^{-3} [$m^3\cdot mol^{-1}$] >constant : key A-Z ? [6]	(Value assigned to numeric variable V)
[↑] [↑] [↑]	Na =6.022045x10^{23} [mol^{-1}] >constant : key A-Z ? [3]	(Avogadro constant)
[N]	[N] =6.022045x10^{23} [mol^{-1}] >constant : key A-Z ? [3]	(Value assigned to numeric variable N)
[BRK]	Break _	(Exit currently specified constant)
[V] [EXE]	V 0.02241383	(Contents of variable V)
[N] [EXE]	N 6.022045E+23	(Contents of variable N)

* Constants assigned to numeric variables are retained even when power is switched OFF. Numeric variables such as N and V can be used in BASIC programs.

5920 ELECTROLYTIC DISSOCIATION CONSTANTS

Displays the following eight ionization equilibrium formulas:

IONIZATION EQUILIBRIUM FORMULAS OF ACID	IONIZATION EQUILIBRIUM CONSTANT ka (mol/l)
$HCOOH \rightleftharpoons HCOO^- + H^+$	1.77×10^{-4}
$CH_3COOH \rightleftharpoons CH_3COO^- + H^+$	1.75×10^{-5}
$C_6H_5COOH \rightleftharpoons C_6H_5COO^- + H^+$	6.31×10^{-5}
$H_2CO_3 \rightleftharpoons H^+ + HCO_3^-$	4.45×10^{-7}
$C_6H_5OH \rightleftharpoons C_6H_5O^- + H^+$	1.00×10^{-7}

IONIZATION EQUILIBRIUM FORMULAS OF BASE	ELECTROLYTIC DISSOCIATION CONSTANT kb (mol/l)
$NH_3 + H_2O \rightleftharpoons NH_4^+ + OH^-$	1.78×10^{-5}
$C_5H_5N + H_2O \rightleftharpoons C_5H_5NH^+ + OH^-$	1.5×10^{-9}
$C_6H_5NH_2 + H_2O \rightleftharpoons C_6H_5NH_3^+ + OH^-$	3.8×10^{-10}

PERATION

5920 [UB]

```
HCOOH -- HCOO- + H+
Ka = 1.77x10-4 [mol/l]        [1]
```

[↓] (or [EXE]) scrolls to the following formula, [↑] to the previous formula, [⇐] to the first formula, and [⇒] to the last (8th) formula.

EXAMPLE

Display a desired ionization equilibrium formula.

Key	Display	
[↓]	CH3COOH -- CH3COO- + H+ / Ka = 1.75x10-5 [mol/l] [2]	(Formula 2)
[↓] [↓] [↓]	C6H5OH -- C6H5O- + H+ / Ka = 1.00x10-7 [mol/l] [5]	(Formula 5)
[⇒]	C6H5NH2 + H2O -- C6H5NH3+ + OH- / Kb = 3.8x10-10 [mol/l] [8]	(Formula 8)
[↑] [↑]	NH3 + H2O -- NH4+ + OH- / Kb = 1.78x10-5 [mol/l] [6]	(Formula 6)
[⇐]	HCOOH -- HCOO- + H+ / Ka = 1.77x10-4 [mol/l] [1]	(Formula 1)

5930 MOTION AND ENERGY

Displays the following 20 scientific formulas:

NAME	FORMULA
Uniformly accelerated motion	$v=v_0+at,\ a=\frac{\Delta v}{\Delta t},\ s=v_0t+\frac{1}{2}at^2$
Newton's equation of motion	$F=ma$
Circular motion (1)	$T=\frac{2\pi r}{v}=\frac{2\pi}{\omega}=\frac{1}{f}$
Circular motion (2)	$\omega=\frac{2\pi}{T}=2\pi f=\frac{v}{r},\ F=mr\omega^2=\frac{mv^2}{r}$
Simple harmonic oscillation	$x=r\cdot\sin\omega t,\ v=r\omega\cdot\cos\omega t,\ a=-\omega^2x$
Hooke's law	$F=-kx$
Spring oscillation	$a=F/m=-\frac{k}{m}x,\ T=2\pi\sqrt{\frac{m}{k}}$
Simple pendulum	$a=F/m=-\frac{g}{l}x,\ T=2\pi\sqrt{\frac{l}{g}}$
Potential energy (spring)	$Ep=mgh$
Elastic energy	$Ee=\frac{1}{2}kx^2$
Kinetic energy	$Ek=\frac{1}{2}mv^2$
Coefficient of friction	$F=\mu N$
Work	$W=Fs$
Kepler's law	$T^2/r^3=\text{Constant}$
Universal gravitation	$F=G\cdot\frac{Mm}{r^2},\ G=6.7\times10^{-11}(N\cdot m^2/kg^2)$
Potential energy (interplanetary)	$Up=-G\frac{Mm}{r}$
Kinetic energy (interplanetary)	$Ek=\frac{1}{2}\cdot mr^2\omega^2$
Moment of inertia	$I=mr^2,\ E=\frac{1}{2}I\omega^2$
Angular momentum	$J=I\omega$
Conservation of momentum	$mv_1+MV_1=mv_2+MV_2$

NOTE: Universal gravitational constant displayed as rounded value (see 5910 for details)

OPERATION

5930 [LIB]

```
Uniformly accelerated motion [1]
v=vo+at . a=Δv/Δt . s=vot+at²/2
```

[↓] (or [EXE]) scrolls to the following formula, [↑] to the previous formula, [⇐] to the first formula, and [⇒] to the last (20th) formula.

EXAMPLE

Display a desired scientific formula.

Key	Display	
[↓]	Newton's equation of motion [2] F=ma	(Formula 2)
[↓] [↓] [↓] [↓]	Hooke's law [6] F=−kx	(Formula 6)
[⇒]	Conservation of momentum [20] mv1 +MV1=mv2+MV2	(Formula 20)
[↑] [↑] [↑]	Kinetic energy (planet) [17] Ek=1/2·mr²ω²	(Formula 17)
[↑] [↑] [↑] [↑] [↑]	Coefficient of friction [12] F=μN	(Formula 12)
[⇐]	Uniformly accelerated motion [1] v=v0+at . a=Δv/Δt . s=v0t+at²/2	(Formula 1)

5932 WAVE MOTION

Displays the following 16 scientific formulas:

NAME	FORMULA
Wave	$v=\frac{\lambda}{T}=f\lambda,\ y=a\cdot\sin 2\pi\left(\frac{t}{T}-\frac{x}{\lambda}\right)$
Velocity of transverse wave on a string	$v=\sqrt{\frac{F}{\rho}}$
Interference	$l_2-l_1=(2n+1)\frac{\lambda}{2},\ l_2-l_1=n\lambda$
Stationary wave	$l=n\frac{\lambda}{2},\ l=(2n-1)\frac{\lambda}{4}\quad(n\neq 0)$
Refraction of wave	$n=\sin\theta/\sin\phi=v_1/v_2=\lambda_1/\lambda_2$
Natural frequency	$f=\frac{1}{2l}\sqrt{\frac{T}{\rho}}$
Velocity of sound	$v=331.5+0.61T$
Doppler effect	$f=f0\frac{v-v_1}{v-v_2}$
Beat	$f=f_1-f_2\quad(f_1>f_2)$
Reflectivity of light	$R0=\left(\frac{n_1-n_2}{n_1+n_2}\right)^2$
Critical angle	$\sin\theta=\frac{n_1}{n_2}$
De Broglie wave	$\lambda=\frac{h}{mv}$
Quantum condition	$2\pi r=\frac{nh}{mv}=n\lambda$
Photoelectric effect	$\frac{1}{2}mv^2=h\nu-W$
Frequency condition	$h\nu=Em-En\quad(m>n)$
Light wave	$\lambda=c/\nu,\ c=2.998\times10^8$ (m/s)

OPERATION

5932 [LB]

```
Wave                              [1]
v=λ/T=fλ  .  y=a·sin2π(t/T-x/λ)
```

[↓] (or [EXE]) scrolls to the following formula, [↑] to the previous formula, [⊜] to the first formula, and [⊜] to the last (16th) formula.

EXAMPLE

Display a desired wave formula.

[↓]

```
Wave of string                    [2]
v=√(F/ρ)
```
(Formula 2)

[↓] [↓] [↓]

```
Refraction                        [5]
n=sinθ/sinφ=v1/v2=λ1/λ2
```
(Formula 5)

[⊜]

```
Light wave                       [16]
λ=c/ν  .  c=2.998x10⁸ (m/s)
```
(Formula 16)

[↑] [↑] [↑] [↑]

```
de Broglie wave                  [12]
λ=h/mv
```
(Formula 12)

[⊜]

```
Wave                              [1]
v=λ/T=fλ  .  y=a·sin2π(t/T-x/λ)
```
(Formula 1)

5934 AC & DC CIRCUITS

Displays the following 16 scientific formulas:

NAME	FORMULA
Ohm's law	$V=IR \quad (I=\frac{Q}{t},\ R=\rho\cdot\frac{l}{S})$
Electric resistance (parallel, series)	$R=R_1+R_2,\ \frac{1}{R}=\frac{1}{R_1}+\frac{1}{R_2}$
DC circuit	$V=E-IR$
DC power and Joule heat	$P=IV=I^2R,\ W=IVt=Pt$
Conductance	$G=\frac{1}{R}=\frac{I}{V}$
Kirchhoff's law	$\Sigma\pm I=0,\ \Sigma\pm V=0$
Wheatstone bridge	$R_0R_1=R_2R_3$
Instantaneous value (AC voltage and current)	$V=V_0\sin\omega t,\ I=I_0\sin\omega t$
Effective value	$I=\frac{I_0}{\sqrt{2}},\ V=\frac{V_0}{\sqrt{2}}$
AC power	$P=VI=\frac{1}{2}V_0I_0$
Power factor	$P=VI\cdot\cos\phi$
Transformer	$I_1V_1=I_2V_2,\ \frac{N_2}{N_1}=\frac{V_2}{V_1}$
Reactance	$X=\omega L=2\pi fL,\ X=\frac{1}{\omega C}=\frac{1}{2\pi fC}$
Impedance	$Z=\sqrt{R^2+(\omega L-\frac{1}{\omega C})^2},\ V_0=ZI_0$
Natural frequency (Natural oscillation)	$f_0=\frac{1}{2\pi\sqrt{LC}}$
Electric oscillation	$\frac{1}{2}\cdot\frac{Q^2}{C}+\frac{1}{2}LI^2=\text{Constant}$

OPERATION

5934 [LIB]

```
Ohm's law                      [1]
V=IR    (I=Q/t. R=ρ·l/S)
```

[↓] (or [EXE]) scrolls to the following formula, [↑] to the previous formula, [⇐] to the first formula, and [⇒] to the last (16th) formula.

EXAMPLE

Display a desired electrical formula.

[↓]
```
Resistance                     [2]
R=R1+R2 .  1/R=1/R1+1/R2
```
(Formula 2)

[↓][↓][↓][↓]
```
Kirchhoff's law                [6]
Σ±I=0 . Σ±V=0
```
(Formula 6)

[⇒]
```
Electric oscillation          [16]
1/2·Q²/C+1/2·LI²=Constant
```
(Formula 16)

[↑][↑][↑]
```
Reactance                     [13]
X=ωL=2πfL . X=1/ωC=1/2πfC
```
(Formula 13)

[↑][↑]
```
Power factor                  [11]
P=VI·cosφ
```
(Formula 11)

[⇐]
```
Ohm's law                      [1]
V=IR    (I=Q/t. R=ρ·l/S)
```
(Formula 1)

5936 ELECTRIC AND MAGNETIC FIELDS

Displays the following 17 scientific formulas:

NAME	FORMULA
Coulomb's law (Electric field)	$F = k_0 \frac{Q_1 Q_2}{r^2}$, $k_0 = 9 \times 10^9 (N \cdot m^2/C^2)$
Electric field	$E = \frac{V}{d}$, $F = QE$, $W = QV$
Electrical capacity	$Q = CV$, $C = \varepsilon_0 \cdot \frac{S}{d}$
Electrical capacity (parallel, series)	$C = C_1 + C_2$, $\frac{1}{C} = \frac{1}{C_1} + \frac{1}{C_2}$
Dielectric constant ε_0 (Relative dielectric constant ε)	$D = \varepsilon_0 E$, $C = \varepsilon C_0$
Electrostatic energy	$U = \frac{1}{2} QV = \frac{1}{2} CV^2$
Electron in electrical field	$a = \frac{QE}{m}$, $\frac{1}{2} mv^2 = eV$
Coulomb's law (magnetic field)	$F = k_0 \frac{m_1 m_2}{r^2}$, $k_0 = \frac{10^7}{(4\pi)^2}$
Magnetic field H	$H = \frac{I}{2\pi r}$, $H = \frac{I}{2r}$, $H = nI$
Magnetic field	$F = \mu_0 IH l = IB l$
Magnetic flux density	$B = \frac{m}{4\pi r^2} = \mu_0 H$
Lorentz force	$F = QvB$, $r = \frac{mv}{QB}$
Electron in magnetic field	$\frac{1}{2} mv^2 = \frac{Q^2 B^2 r^2}{2m}$, $\omega = \frac{v}{r} = \frac{QB}{m}$
Faraday's law of induction	$V = -n \frac{\Delta \phi}{\Delta t}$
Electromagnetic induction	$V = E\ell = vB\ell$, $I = \frac{vB\ell}{R}$
Mutual induction	$V_2 = -M \frac{\Delta I_1}{\Delta t}$
Self-induction	$V' = -L \frac{\Delta I}{\Delta t}$

OPERATION

5936 [UB]

```
Coulomb's law (electric f.)   [1]
F=ko·Q1Q2/r2 ·  ko=9x10⁹[N·m2/C2]
```

[⇩] (or [EXE]) scrolls to the following formula, [⇧] to the previous formula, [⇐] to the first formula, and [⇒] to the last (17th) formula.

EXAMPLE

Display a desired scientific (electric and magnetic field) formula.

Key	Display	
⇩	Electric field [2] E=V/d . F=QE . W=QV	(Formula 2)
⇩ ↓ ↓ ↓ ↓	Electrons in electric field [7] a=QE/m . 1/2·mv2=eV	(Formula 7)
⇨	Self-induction [17] V'=-L·ΔI/Δt	(Formula 17)
⇧ ↑ ↑ ↑	Electrons in magnetic field [13] 1/2·mv2=Q2B2r2/2m . w=v/r=QB/m	(Formula 13)
⇧	Lorentz force [12] F=QvB . r=mv/QB	(Formula 12)
⇦	Coulomb's law (electric f.) [1] F=ko·Q1Q2/r2 · ko=9x109[N·m2/C2]	(Formula 1)

5938 THERMODYNAMICS AND OTHERS

Displays the following 13 scientific formulas:

NAME	FORMULA
Absolute temperature	$T(^\circ K)=t(^\circ C)+273.15$
Heat capacity	$Q=CT=mcT$
Mechanical equivalent of heat	$W=JQ,\ J=4.19[J/cal]$
Boyle's law	$PV=\text{Constant}\quad (T=\text{constant})$
Thermal expansion (volume and temperature)	$V=V_0(1+\frac{T}{273})$
Charles' law	$\frac{V}{V_0}=\frac{T}{T_0}$
Equation of state	$PV=nRT,\ R=8.31[J/K]$
Law of partial pressures	$P=P_1+P_2+P_3+\cdots\cdots$
Pressure	$P=\frac{1}{3}nm\bar{v}^2$
Internal energy	$U=\frac{1}{2}m\bar{v}^2\ N=\frac{3}{2}nRT$
Specific heat	$C_v=\frac{\Delta U}{\Delta T}=\frac{3R}{2},\ C_p=\frac{\Delta U}{\Delta T}+R=\frac{5R}{2}$
Half life	$N=N_0\left(\frac{1}{2}\right)^x\quad (x=\frac{1}{T})$
Mass-energy relation	$E=mc^2$

OPERATION

5938 [LIB]

```
Absolute temperature          [1]
T[°K]=t[°C]+273.15
```

[↓] (or [EXE]) scrolls to the following formula, [↑] to the previous formula, [⇐] to the first formula, and [⇒] to the last (13th) formula.

EXAMPLE

Display a desired scientific (thermodynamic and others) formula.

[↓]

```
Heat capacity                 [2]
Q=CT=mcT
```
(Formula 2)

[↓] [↓] [↓] [↓]

```
Charle's law                  [6]
V/Vo=T/To
```
(Formula 6)

[⇒]

```
Mass-energy relation         [13]
E=mc²
```
(Formula 13)

[↑] [↑] [↑] [↑] [↑]

```
Law of partial pressures      [8]
P=P1+P2+P3+···
```
(Formula 8)

[↑]

```
Equation of state             [7]
PV=nRT , R=8.31 [J/K]
```
(Formula 7)

[⇐]

```
Absolute temperature          [1]
T[°K]=t[°C]+273.15
```
(Formula 1)

5950 METRIC CONVERSIONS FOR LENGTH

Displays the following 30 conversion formulas. Pressing [EXE] stores the currently displayed formula which then can be applied for calculation.

CONVERSION UNIT	CONVERSION FORMULA			CONVERSION UNIT	CONVERSION FORMULA		
[cm]	×	0.01	[m]	x [ft]	×	30.48	[cm]
		0.393701	[in]			0.3048	[m]
		0.0328084	[ft]			12	[in]
		0.0109361	[yd]			0.333333	[yd]
		0.00000621371	[mile]			0.000189394	[mile]
[m]	×	100	[cm]	[yd]	×	91.44	[cm]
		39.3701	[in]			0.9144	[m]
		3.28084	[ft]			36	[in]
		1.09361	[yd]			3	[ft]
		0.000621371	[mile]			0.000568182	[mile]
[in]	×	2.54	[cm]	[mile]	×	160934.4	[cm]
		0.0254	[m]			1609.344	[m]
		0.0833333	[ft]			63360	[in]
		0.0277778	[yd]			5280	[ft]
		0.0000157828	[mile]			1760	[yd]

OPERATION

5950 [LIB]

```
Metric conversion (length)        [1]
x[cm] -- 0.01x[m]
```

[⇩] or [EXE] scrolls to the following formula, [⇧] to the previous formula, [⇐] to the first formula, and [⇒] to the last (30th) formula.
pressing of [EXE] executes a conversion of the currently displayed units.

EXAMPLE

Display a desired conversion formula.

Key	Display	
[⇩]	Metric conversion (length) [2] x[cm] -- 0.393701x[in]	(Formula 2)
[⇩] [⇩] [⇩]	Metric conversion (length) [5] x[cm] -- 0.00000621371x[mile]	(Formula 5)
[⇒]	Metric conversion (length) [30] x[mile] -- 1760x[yd]	(Formula 30)
[⇧] [⇧] [⇧] [⇧]	Metric conversion (length) [26] x[mile] -- 160934.4x[cm]	(Formula 26)
[⇐]	Metric conversion (length) [1] x[cm] -- 0.01x[m]	(Formula 1)

EXAMPLE

Convert 110m and 300m to yards.

Key operation	Display	Remark
[↓] [↓] [↓] [↓] [↓] [↓] [↓] [↓]	Metric conversion (length) [9] x[m] -- 1.09361x[yd]	(Formula 9)
[EXE]	x[m]?_	(Stores Formula 9 in memory)
110 [EXE]	x[m]?110 X[yd]= 120.2971	(110m = 120.2971 yards)
[EXE]	X[yd]= 120.2971 x[m]?_	
300 [EXE]	x[m]?300 X[yd]= 328.083	(300m = 328.183 yards)

* Once calculation is complete, a different conversion can be selected by first pressing [EXE] followed by the [LIB] key.

IMPORTANT

This library function is executed by first storing the conversion formula into the formula storage memory. Note that the current formula memory contents are cleared by this procedure.

5960 METRIC CONVERSIONS FOR AREA

Displays the following 12 conversion formulas. Pressing [EXE] stores the currently displayed formula which then can be applied for calculation.

CONVERSION UNIT	CONVERSION FORMULA			CONVERSION UNIT	CONVERSION FORMULA		
x (m^2)	×	0.01	(a)	x (acre)	×	4046.86	(m^2)
		0.000247105	(acre)			40.4686	(a)
		0.000000386102	($mile^2$)			0.0015625	($mile^2$)
(a)		100	(m^2)	($mile^2$)		2589990	(m^2)
		0.0247105	(acre)			25899.9	(a)
		0.0000386102	($mile^2$)			640	(acre)

OPERATION

5960 [LIB]

```
Metric conversion (area)      [1]
x[m2] -- 0.01x[a]
```

[↓] scrolls to the following formula, [↑] to the previous formula, [⇤] to the first formula, and [⇥] to the last (12th) formula.
Pressing of [EXE] executes a conversion of the currently displayed units.

EXAMPLE

Display a desired conversion formula.

[↓]

```
Metric conversion (area)      [2]
x[m2] -- 0.000247105x[acre]
```

(Formula 2)

[↓] [↓] [↓]

```
Metric conversion (area)      [5]
x[a] -- 0.0247105x[acre]
```

(Formula 5)

[⇥]

```
Metric conversion (area)     [12]
x[mile2] -- 640x[acre]
```

(Formula 12)

[↑] [↑]

```
Metric conversion (area)     [10]
x[mile2] -- 2589990x[m2]
```

(Formula 10)

[⇤]

```
Metric conversion (area)      [1]
x[m2] -- 0.01x[a]
```

(Formula 1)

SAMPLE

Convert $300m^2$ to acres.

[↓]

```
Metric conversion (area)      [2]
x[m2] -- 0.000247105x[acre]
```

(Formula 2)

[EXE]

```
x[m2]?_
```

(Stores Formula 2 in memory and executes)

300 [EXE]

```
x[m2]?300
X[acre]= 0.0741315
```

($300m^2$ = 0.074 acres)

Once calculation is complete, a different conversion can be selected by first pressing [EXE] followed by the [LIB] key.

IMPORTANT
This library function is executed by first storing the conversion formula into the formula storage memory. Note that the current formula memory contents are cleared by this procedure.

5970 METRIC CONVERSIONS FOR VOLUME

Displays the following 30 conversion formulas. Pressing [EXE] stores the currently displayed formula which then can be applied for calculation.

CONVERSION UNIT	CONVERSION FORMULA		CONVERSION UNIT	CONVERSION FORMULA	
x cm³	× 0.000001	(m³)	ft³	28316.8	(cm³)
	0.0610237	(in³)		0.0283168	(m³)
	0.0000353147	(ft³)		1728	(in³)
	0.001	(l)		28.3168	(l)
	0.000264172	(gal(US))		7.48052	(gal(US))
	0.000219968	(gal(UK))		6.22882	(gal(UK))
m³	1000000	(cm³)	x l	× 1000	(cm³)
	61023.7	(in³)		0.001	(m³)
	35.3147	(ft³)		61.0237	(in³)
	1000	(l)		0.0353147	(ft³)
	264.172	(gal(US))		0.264172	(gal(US))
	219.968	(gal(UK))		0.219968	(gal(UK))
in³	16.3871	(cm³)			
	0.0000163871	(m³)			
	0.000578704	(ft³)			
	0.0163871	(l)			
	0.00432900	(gal(US))			
	0.00360464	(gal(UK))			

OPERATION

5970 [LIB]

```
Metric conversion (volume)    [1]
x[cm³] -- 0.000001x[m³]
```

[↓] scrolls to the following formula, [↑] to the previous formula, [⊖] to the first formula, and [⊖] to the last (30th) formula.
Pressing of [EXE] executes a conversion of the currently displayed units.

EXAMPLE

Display a desired conversion formula.

Key	Display	
[↓]	Metric conversion (volume) [2] x[cm³] -- 0.0610237x[in³]	(Formula 2)
[↓] [↓] [↓] [↓] [↓]	Metric conversion (volume) [7] x[m³] -- 1000000x[cm³]	(Formula 7)
[⇒]	Metric conversion (volume) [42] x[gal(UK)] -- 0.832672x[gal(US)]	(Formula 30)
[↑] [↑] [↑] [↑]	Metric conversion (volume) [38] x[gal(UK)] -- 0.00454609x[m³]	(Formula 26)
[⇐]	Metric conversion (volume) [1] x[cm³] -- 0.000001x[m³]	(Formula 1)

EXAMPLE

Convert 1800cm³ to gallons (US).

Key	Display	
[↓] [↓] [↓] [↓]	Metric conversion (volume) [5] x[cm³] -- 0.000264172x[gal(US)]	(Formula 5)
[EXE]	x[cm³]?_	(Stores Formula 5 in memory and executes)
1800 [EXE]	x[cm³]?1800 X[gal(US)]= 0.4755096	(1800cm³ = approximately 0.48 gallons)

Once calculation is complete, a different conversion can be selected by first pressing [BRK] followed by the [LIB] key.

IMPORTANT

This library function is executed by first storing the conversion formula into the formula storage memory. Note that the current formula memory contents are cleared by this procedure.

5980 METRIC CONVERSIONS FOR WEIGHT

Displays the following 12 conversion formulas. Pressing [EXE] stores the currently displayed formula which then can be applied for calculation.

CONVERSION UNIT	CONVERSION FORMULA	CONVERSION UNIT	CONVERSION FORMULA
x [g]	× 0.001 [kg] 0.0352740 [oz] 0.00220462 [lb]	x [oz]	× 28.3495 [g] 0.0283495 [kg] 0.0625 [lb]
x [kg]	1000 [g] 35.2740 [oz] 2.20462 [lb]	x [lb]	453.59237 [g] 0.45359237 [kg] 16 [oz]

OPERATION

5980 [LIB]

```
Metric conversion (weight)    [1]
x[g] -- 0.001x[Kg]
```

[↓] scrolls to the following formula, [↑] to the previous formula, [⊟] to the first formula, and [⊟] to the last (12th) formula.
Pressing of [EXE] executes a conversion of the currently displayed units.

EXAMPLE

Display a desired conversion formula.

Key	Display	
[↓]	Metric conversion (weight) [2] x[g] -- 0.0352740x[oz]	(Formula 2)
[↓] [↓] [↓] [↓]	Metric conversion (weight) [6] x[Kg] -- 2.20462x[lb]	(Formula 6)
[⊟]	Metric conversion (weight) [12] x[lb] -- 16x[oz]	(Formula 12)
[↑] [↑] [↑]	Metric conversion (weight) [9] x[oz] -- 0.0625x[lb]	(Formula 9)
[⊟]	Metric conversion (weight) [1] x[g] -- 0.001x[Kg]	(Formula 1)

EXAMPLE

Convert 2.5kg to ounces.

Key	Display	
[↓] [↓] [↓] [↓]	Metric conversion (weight) [5] x[Kg] -- 35.2740x[oz]	(Formula 5)
[EXE]	x[Kg]?_	(Stores Formula 5 in memory and executes)
2 [.] 5 [EXE]	x[Kg]?2.5 X[oz]= 88.185	(2.5kg = 88.185 ounces)

- Once calculation is complete, a different conversion can be selected by first pressing [EXE] followed by the [LIB] key.

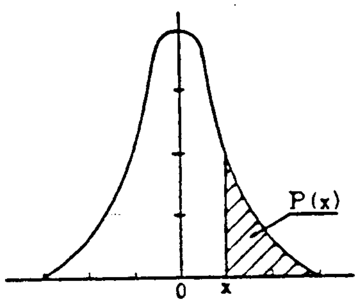
IMPORTANT

This library function is executed by first storing the conversion formula into the formula storage memory. Note that the current formula memory contents are cleared by this procedure.

6210 UPPER PROBABILITY INTEGRALS (NORMAL DISTRIBUTION)

Determines upper probability for normal distribution with five significant digits using the following formula:

$$f(x) = \int_x^{\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx$$



OPERATION

6210 [UB]

```
Upper probability     N(0.1²)
x= 0 ?_
```

EXAMPLE

Determine the upper probability for normal distribution when x = 1.53.

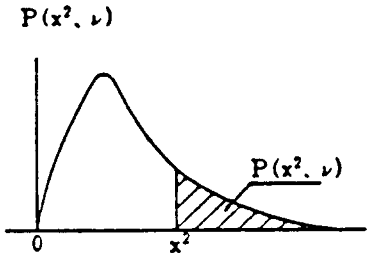
Key	Display	
[EXE]	Upper probability N(0.1²) x= 0 ?_	
1 [.] 53 [EXE]	Upper probability N(0.1²) p= 0.063008	
[EXE]	Upper probability N(0.1²) x= 1.53 ?_	(Return to initial display)

Here, the upper probability integral is 0.063008.

6220 UPPER PROBABILITY INTEGRALS
(x^2 DISTRIBUTION)

Determines upper probability for x^2 distribution with five significant digits using the following formula:

$$P(x^2, \nu) = \int_{x^2}^{\infty} \frac{1}{2\Gamma(\frac{\nu}{2})} \left(\frac{x^2}{2}\right)^{\frac{\nu}{2}-1} e^{-\frac{x^2}{2}} dx^2$$ (ν : degree of freedom)



OPERATION

6220 [UB]	Upper probability X²(x², ν) ν= 1 ?_	

EXAMPLE

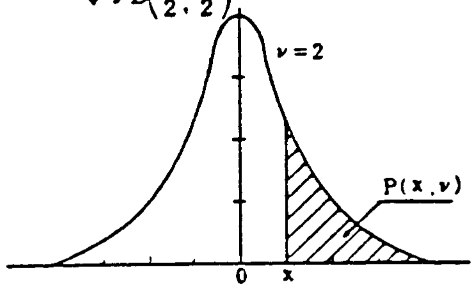
Determine the upper probability for x^2 distribution when degree of freedom (ν) = 4, and x^2 = 2.

4 [EXE]	Upper probability X²(x², ν) x²= 0 ?_	(Degree of freedom)
2 [EXE]	Upper probability X²(x², ν)	(Value of x²)
	Upper probability X²(x², ν) p= 0.73576	(The upper probability integral is 0.73576.)
[EXE]	Upper probability X²(x², ν) ν= 4 ?_	(Return to initial display)

6230 UPPER PROBABILITY INTEGRALS (t DISTRIBUTION)

Determines upper probability for t distribution with five significant digits using the following formula:

$$P(x, \nu) = \int_x^\infty \frac{\left(1+\frac{x^2}{\nu}\right)^{-\frac{(\nu+1)}{2}}}{\sqrt{\nu}\, B\left(\frac{1}{2}, \frac{\nu}{2}\right)} dx \qquad (\nu : \text{degree of freedom})$$



OPERATION

6230 [LIB]

Upper probability t(x.ν) ν= 1 ?_

EXAMPLE

Determine the upper probability for t distribution when degree of freedom (ν) = 2, and x = 2.92.

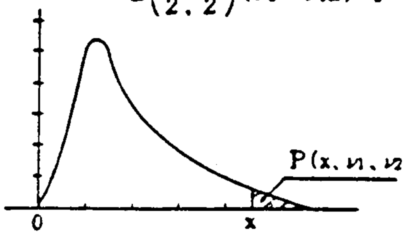
Key	Display	Note
2 [EXE]	Upper probability t(x.ν) x= 0 ?_	(Degree of freedom)
2 [.] 92 [EXE]	Upper probability t(x.ν)	(Enter value of x)
	Upper probability t(x.ν) p= 0.05	(The upper probability integral is 0.05.)
[EXE]	Upper probability t(x.ν) ν= 2 ?_	(Return to initial display)

6240 UPPER PROBABILITY INTEGRALS
(F DISTRIBUTION)

Determines upper probability for F distribution with five significant digits using the following formula:

$$P(x, \nu_1, \nu_2) = \int_x^{\infty} \frac{\nu_1^{\frac{\nu_1}{2}} \nu_2^{\frac{\nu_2}{2}} \cdot x^{\frac{\nu_1}{2}-1}}{B\left(\frac{\nu_1}{2}, \frac{\nu_2}{2}\right)(\nu_2 + \nu_1 x)^{\frac{\nu_1+\nu_2}{2}}} dx$$

(ν_1: degree of freedom 1; ν_2: degree of freedom 2)



OPERATION

6240 [LIB]	Upper probability F(x, ν1, ν2) ν1= 1 ?_	

EXAMPLE

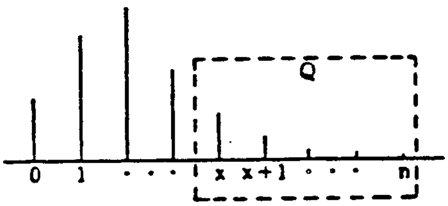
Determine the upper probability for F distribution when degree of freedom 1 (ν_1) = 5, degree of freedom 2 (ν_2) = 3 and x = 9.01.

5 [EXE]	Upper probability F(x, ν1, ν2) ν2= 1 ?_	(Degree of freedom 1)
3 [EXE]	Upper probability F(x, ν1, ν2) x= 0 ?_	(Degree of freedom 2)
9 [·] 01 [EXE]	Upper probability F(x, ν1, ν2)	(Value of x)
	Upper probability F(x, ν1, ν2) p= 0.050026	(The upper probability integral is 0.050026.)
[EXE]	Upper probability F(x, ν1, ν2) ν1= 5 ?_	(Return to initial display)

6310 UPPER CUMULATIVE FREQUENCY (BINOMIAL DISTRIBUTION)

Determines upper cumulative frequency for binomial distribution with five significant digits using the following formula:

$$B(x, n, p) = \sum_{y=x}^{n} \binom{n}{y} P^{y}(1-P)^{n-y}$$



n : maximum value of x
p : probability
Q: Sum of frequencies produces past x (cumulative frequency)

OPERATION

6310 [LB]	Cumulative frequency B(x.n.p) n= 2 ?_	

[EXAMPLE]

Determine the upper cumulative frequency for binomial distribution when the maximum value of x (n) = 5, probability (p) = 0.5 and x = 4.

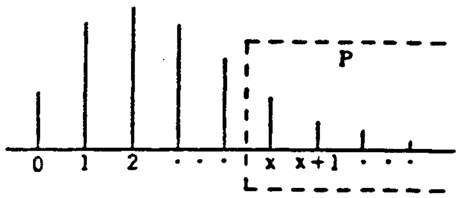
5 [EXE]	Cumulative frequency B(x.n.P) P= 0 ?_	(Maximum value of x)
0 [.] 5 [EXE]	Cumulative frequency B(x.n.P) x= 0 ?_	(Probability)
4 [EXE]	Cumulative frequency B(x.n.p)	(Value of x)
	Cumulative frequency B(x.n.P) p= 0.1875	(The upper cumulative frequency is 0.1875.)
[EXE]	Cumulative frequency B(x.n.P) n= 5 ?_	(Return to initial display)

6320 UPPER CUMULATIVE FREQUENCY (POISSON DISTRIBUTION)

Determines upper cumulative frequency for Poisson distribution with five significant digits using the following formula:

$$P(x, \lambda) = \sum_{y=x}^{\infty} e^{-\lambda} \cdot \frac{\lambda^y}{y!}$$

λ : mean value
P : Sum of frequencies produces past x (cumulative frequency)



OPERATION

6320 [LB]

```
Cumulative frequency   P(x,λ)
λ= 0 ?_
```

EXAMPLE

Determine the upper cumulative frequency for Poisson distribution when mean value (λ) = 2, and x = 4.

2 [EXE]

```
Cumulative frequency   P(x,λ)
x= 0 ?_
```

(Mean value)

4 [EXE]

```
Cumulative frequency   P(x,λ)
. . . . . .
```

(Value of x)

```
Cumulative frequency   P(x,λ)
p= 0.14288
```

(The upper cumulative frequency is 0.14288.)

[EXE]

```
Cumulative frequency   P(x,λ)
λ= 2 ?_
```

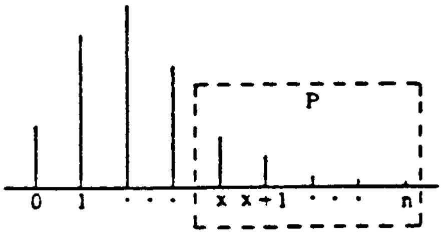
(Return to initial display)

6330 UPPER CUMULATIVE FREQUENCY (HYPERGEOMETRIC DISTRIBUTION)

Determines upper cumulative frequency for hypergeometric distribution with five significant digits using the following formula:

$$H(x, n, M, N) = \sum_{r=x}^{n} \frac{\binom{M}{r}\binom{N-M}{n-r}}{\binom{N}{n}}$$

P : Sum of frequencies produces past x (cumulative frequency)



OPERATION

6330 [LIB]

```
Cumulative frequency H(x.n.M.N)
N= 2 ?_
```

EXAMPLE

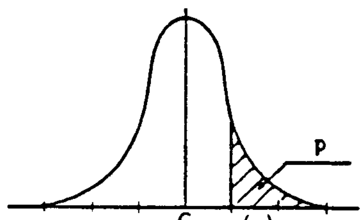
Determine the upper cumulative frequency for hypergeometric distribution when N = 3, M = 2, n = 1, and x = 1.

Key	Display	
3 [EXE]	Cumulative frequency H(x.n.M.N) M= 1 ?_	(Value of N)
2 [EXE]	Cumulative frequency H(x.n.M.N) n= 1 ?_	(Value of M)
1 [EXE]	Cumulative frequency H(x.n.M.N) x= 0 ?_	(Value of n)
1 [EXE]	Cumulative frequency H(x.n.M.N)	(Value of x)
	Cumulative frequency H(x.n.M.N) p= 0.66667	(The upper cumulative frequency is 0.66667.)
[EXE]	Cumulative frequency H(x.n.M.N) N= 3 ?_	(Return to initial display)

6410 PERCENTAGE POINT NORMAL DISTRIBUTION

Determines percentage point for normal distribution with five significant digits using the following formula:

$$x(p) : \int_{x}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx = p \quad \text{(p : probability)}$$



OPERATION

6410 [UB]	Percentage points N(0.1²) p= 0 ?_	

EXAMPLE

Determine the percentage point for normal distribution when p = 0.05.

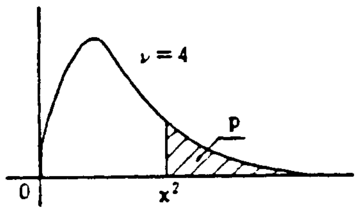
0 [·] 05 [EXE]	Percentage points N(0.1²)	(Probability)
	Percentage points N(0.1²) x= 1.6449	(The percentage point is 1.6449.)
[EXE]	Percentage points N(0.1²) p= 0.05 ?_	(Return to initial display)

6420 PERCENTAGE POINT (x^2 DISTRIBUTION)

Determines percentage point for x^2 distribution with five significant digits using the following formula:

$$\chi^2 p(\nu) : \int_{x^2 p}^{\infty} \frac{1}{2\Gamma(\frac{\nu}{2})}\left(\frac{x^2}{2}\right)^{\frac{\nu}{2}-1} \cdot e^{-\frac{x^2}{2}}\, dx^2 = p$$

(ν : degree of freedom
Γ : gamma function
p : probability)



OPERATION

Key	Display
6420 [LIB]	Percentage points X² (x², ν) ν= 1 ?_

EXAMPLE

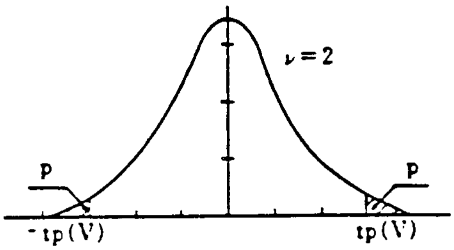
Determine the percentage point for x^2 distribution when degree of freedom $(\nu) = 2$, and probability $p = 0.5$.

Key	Display	Comment
2 [EXE]	Percentage points X² (x², ν) p= 0 ?_	(Degree of freedom)
0 [.] 5 [EXE]	Percentage points X² (x², ν)	(Probability)
	Percentage points X² (x², ν) x²= 1.3863	(The percentage point is 1.3863)
[EXE]	Percentage points X² (x², ν) ν= 2 ?_	(Return to initial display)

6430 PERCENTAGE POINT (t DISTRIBUTION)

Determines percentage point for t distribution with five significant digits using the following formula:

$$tp(\nu):\ \int_{tp}^{\infty}\frac{\left(1+\frac{t^2}{\nu}\right)^{-(\nu+1)/2}}{\sqrt{\nu B\left(\frac{1}{2},\ \frac{\nu}{2}\right)}}\,dt=p \qquad \left(\begin{array}{l}\nu:\ \text{degree of freedom}\\ p:\ \text{probability}\end{array}\right)$$



OPERATION

Key operation	Display	
6430 [LB]	Percentage points t(x.ν) ν= 1 ?_	

EXAMPLE

Determine the percentage point for t distribution when degree of freedom (ν) = 1, and probability (p) = 0.05.

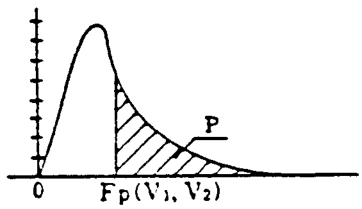
Key operation	Display	Remarks
1 [EXE]	Percentage points t(x.ν) p= 0 ?_	(Degree of freedom)
0 [.] 05 [EXE]	Percentage points t(x.ν)	(Probability)
	Percentage points t(x.ν) x= 6.3137	(The percentage point is 6.3137)
[EXE]	Percentage points t(x.ν) ν= 1 ?_	(Return to initial display)

440 PERCENTAGE POINT
(F DISTRIBUTION)

Determines percentage point for F distribution with five significant digits using the following formula:

$$F_p(\nu_1, \nu_2) : \int_{F_p}^{\infty} \frac{\nu_1^{\frac{\nu_1}{2}} \nu_2^{\frac{\nu_2}{2}} F^{\frac{\nu_1}{2}-1}}{B(\frac{\nu_1}{2}, \frac{\nu_2}{2})(\nu_2 + \nu_1 F)^{\frac{\nu_1+\nu_2}{2}}} dF = p$$

(ν_1 : degree of freedom 1
ν_2 : degree of freedom 2
p : probability)



OPERATION

6440 [LB]	Percentage points F(x.ν1.ν2) ν1= 1 ?_

EXAMPLE

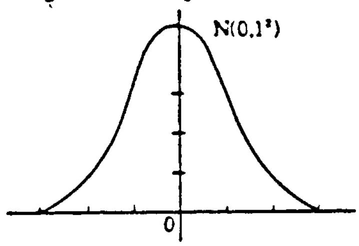
Determine the percentage point for F distribution when degree of freedom 1 (ν_1) = 2, degree of freedom 2 (ν_2) = 3 and probability (p) = 0.05.

2 [EXE]	Percentage points F(x.ν1.ν2) ν2= 1 ?_	(Degree of freedom 1)
3 [EXE]	Percentage points F(x.ν1.ν2) p= 0 ?_	(Degree of freedom 2)
0 [.] 05 [EXE]	Percentage points F(x.ν1.ν2)	(Probability)
	Percentage points F(x.ν1.ν2) x= 9.5521	(The percentage point is 9.5521.)
[EXE]	Percentage points F(x.ν1.ν2) ν1= 2 ?_	(Return to initial display)

6450 NORMAL RANDOM NUMBERS

Generates random numbers contained in the standard normal distribution N (0, 1^2). This unit creates two independent normal random numbers (u, ν) based upon two uniform random numbers (x, y).

$$u = \sqrt{-2 \cdot \log_e x} \cdot \cos(2\pi y)$$
$$\nu = \sqrt{-2 \cdot \log_e x} \cdot \sin(2\pi y)$$



OPERATION

6450 [LIB]

```
0.6103300096
```

[EXAMPLE]

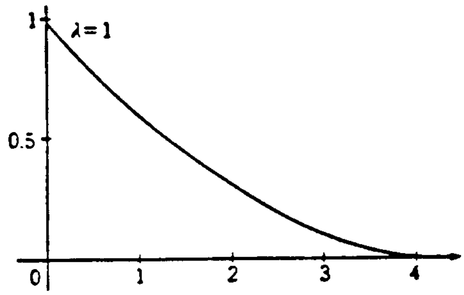
Generate a series of normal random numbers.

[EXE]	0.6103300096 0.5713331954
[EXE]	0.5713331954 -1.664304086
[EXE]	-1.664304086 0.5420575429

5460 EXPONENTIAL RANDOM NUMBERS

ıerates random numbers contained in the exponential distribution E (λ, t). This unit cre- random numbers in accordance with exponential distribution using uniform random ıbers.

$t=\frac{1}{\lambda}\ln x$



(λ : mean value)

:RATION

5O [LIB]

```
E(λ.t)
λ= 1 ?_
```

ıMPLE

ərate a series of exponential random numbers when the mean value (λ) = 3.

```
E(λ.t)
λ= 1 ?_
```
(Mean value)

```
E(λ.t)
 0.1164873901
```

```
 0.1164873901
 0.7074505617
```

```
 0.7074509817
 0.6282426639
```

ı return to the mean value input display, first press [BRK] to terminate the library. Next, əss [LIB] to return to the initial display.

6500 SINGLE VARIABLE STATISTICS

Determines the following statistics and determines the deviation value for input of n data items.

Number of data items	CNT : n	
Sum of data	SUMX : $\sum x$	
Sum of squares of data	SUMX2 : $\sum x^2$	
Mean of data	MEANX : $\sum x/n$	
Population standard deviation of data	SDXN : $x\sigma n$	$\sqrt{\frac{n\sum x^2-(\sum x)^2}{n^2}}$
Sample standard deviation of data	SDX : $x\sigma n\text{-}1$	$\sqrt{\frac{n\sum x^2-(\sum x)^2}{n(n-1)}}$

OPERATION

6500 [LB]

```
Statistics  [x]
>In.Del.Clear.List.T-score.P ?_
```

The menu illustrated above is displayed for single variable statistical calculations. The following six items can be selected from this menu:

1. I : Data input (does not clear data already present in memory)
2. D: Data deletion (deletes erroneous or unnecessary data)
3. C: Data clear
4. L: Statistic display
   Displays number of data items, sum of data, sum of squares of data, mean of data, population standard deviation of data, and sample standard deviation of data in sequence. [↓] (or [EXE]) scrolls to the following data item, [↑] to the previous data item, and [⊜] or [⊜] terminate statistic display.
5. T: Calculates deviation value of obtained value.
6. P: Outputs all statistics to printer

EXAMPLE

Enter the following five test scores and display statistics. Also determine the deviation value for the score of 88.

Data : 98, 88, 62, 90, 78

C

```
Statistics  [x]
clear data  (Y/N) ?
```

(Data clear)

Y

```
Statistics  [x]
>In.Del.Clear.List.T-score.P ?_
```

(Data clear confirmation)

I

```
Input data (x)          [EXE]:menu
x?_
```

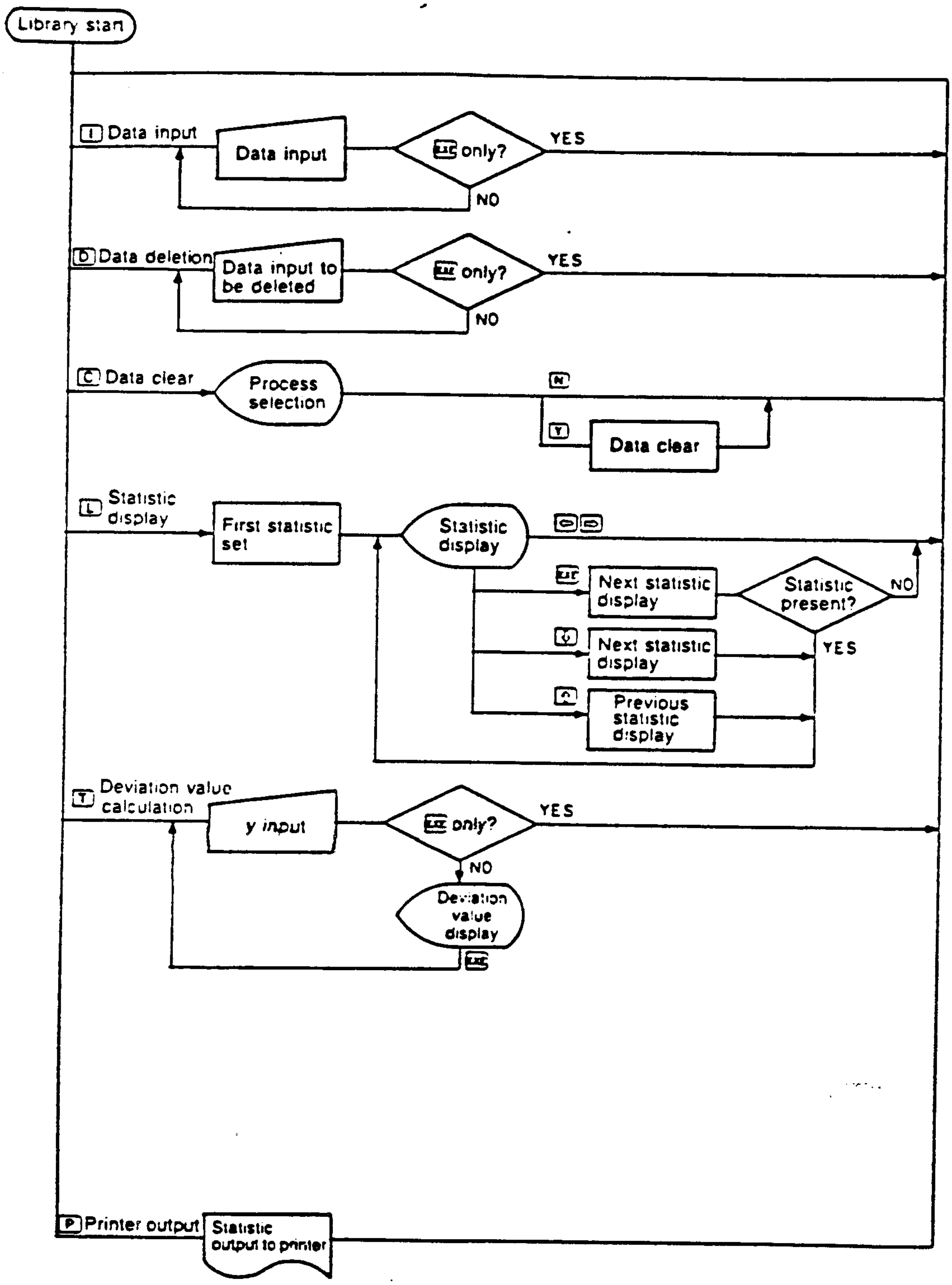
(Data input)

8 EXE 88 EXE 62 EXE 90 EXE 78 EXE

Display	
Input data (x) [EXE]:menu x?_	(Input of each score)
Statistics [x] >In.Del.Clear.List.T-score.P ?_	(Return to menu display)
CNT :n = 5 SUMX :Σx = 416	(Statistic display showing number of data and sum)
SUMX :Σx = 416 SUMX2 :Σx² = 35376	(Sum of squares)
SUMX2 :Σx² = 35376 MEANX :Σx/n = 83.2	(Mean)
MEANX :Σx/n = 83.2 SDXN :xσn = 12.36769987	(Population standard deviation)
SDXN :xσn = 12.36769987 SDX :xσn-1 = 13.82750881	(Sample standard deviation)
Statistics [x] >In.Del.Clear.List.T-score.P ?_	(Return to menu display)
Statistics [x] x?_	(Deviation value)
Statistics [x] x?88 :T= 53.9	(Data for calculation of deviation value to be displayed)
Statistics [x] x?_	
Statistics [x] >In.Del.Clear.List.T-score.P ?_	(Return to menu display)

re, the deviation value of the 88 score is 53.9.

SINGLE VARIABLE STATISTICS FLOWCHART



6510

LINEAR REGRESSION ANALYSIS
(y = a + bx)

erforms linear regression analysis on n data groups (x, y) and calculates the statistics listed elow. Also determines the following on the regression line:

Estimated value of x in relation to y (EOX)
Estimated value of y in relation to x (EOY)

TATISTIC TABLE

Number of data items	CNT : n	
Sum of x data	SUMX : Σx	
Sum of y data	SUMY : Σy	
Sum of squares of x data	SUMX2 : Σx^2	
Sum of squares of y data	SUMY2 : Σy^2	
um of products of x and y data	SUMXY : Σxy	
lean of x data	MEANX : $\Sigma x/n$	
lean of y data	MEANY : $\Sigma y/n$	
opulation standard deviation of x data	SDXN : $x\sigma n$	$\sqrt{\frac{n\Sigma x^2-(\Sigma x)^2}{n^2}}$
opulation standard deviation of y data	SDYN : $y\sigma n$	$\sqrt{\frac{n\Sigma y^2-(\Sigma y)^2}{n^2}}$
ample standard deviation of x data	SDX : $x\sigma n\text{-}1$	$\sqrt{\frac{n\Sigma x^2-(\Sigma x)^2}{n(n-1)}}$
ample standard deviation of y data	SDY : $y\sigma n\text{-}1$	$\sqrt{\frac{n\Sigma y^2-(\Sigma y)^2}{n(n-1)}}$
near regression constant term	LRA : a	$\frac{\Sigma y-b\cdot\Sigma x}{n}$
near regression coefficient	LRB : b	$\frac{n\Sigma xy-\Sigma x\cdot\Sigma y}{n\Sigma x^2-(\Sigma x)^2}$
orrelation coefficient	COR : r	$\frac{n\Sigma xy-\Sigma x\cdot\Sigma y}{\sqrt{(n\Sigma x^2-(\Sigma x)^2(n\Sigma y^2-(\Sigma y)^2)}}$

ERATION

10 [LIB]

```
Regression analysis        (y=a+bx)
>In.Del.Clear.List.eoX.eoY.P ?_
```

e menu illustrated above is displayed for linear regression calculations. The following seven ns can be selected from this menu:

I : Data input
D: Data deletion (deletes erroneous or unnecessary data)
C: Data clear

4. L : Statistic display
   Displays number of data items, sum of x data, sum of y data, sum of squares of x data, sum of squares of y data, sum of products of x and y data, mean of x data, mean of y data, population standard deviation of x data, population standard deviation of y data, sample standard deviation of x data, sample standard deviation of y data, linear regression constant term, linear regression coefficient, and correlation coefficient. [↓] (or [EXE]) scrolls to the following data item, [↑] to the previous data item, and [=] or [=] terminates statistic display.
5. X: Calculates x value for y on regression line.
6. Y: Calculates y value for x on regression line .
7. P: Outputs all statistics to printer.

[EXAMPLE]

Enter the following five sets of height/weight, and display statistics. Also estimate the weight for a person whose height is 170cm.

	1	2	3	4	5
Height (x)	160	158	175	163	172
Weight (y)	43	45	60	46	58

Key operation	Display	
C	Regression analysis [y=a+bx] clear data (Y/N) ?	(Data clear)
Y	Regression analysis [y=a+bx] >In.Del.Clear.List.eoX.eoY.P ?_	(Data clear confirmation)
I	Input data (x,y) [EXE]:menu x?_ :y?	(Data input)
160 [EXE]	Input data (x,y) [EXE]:menu x?160 :y?_	(x input)
43 [EXE]	Input data (x,y) [EXE]:menu x?_ :y?	(y input)
158 [EXE] 45 [EXE] 175 [EXE] 60 [EXE] 163 [EXE] 46 [EXE] 172 [EXE] 58 [EXE]	Input data (x,y) [EXE]:menu x?_ :y?	(Remaining x, y data input)
[EXE]	Regression analysis [y=a+bx] >In.Del.Clear.List.eoX.eoY.P ?_	(Return to menu display)
L	CNT :n = 5 SUMX :Σx = 828	(Statistic display showing number of data and sum of x data)
[EXE]	SUMX :Σx = 828 SUMY :Σy = 252	(Sum of y data)
[EXE]	SUMY :Σy = 252 SUMX2 :Σx² = 137342	(Sum of squares of x data)
[EXE]	SUMX2 :Σx² = 137342 SUMY2 :Σy² = 12954	(Sum of squares of y data)
[EXE]	SUMY2 :Σy² = 12954 SUMXY :Σxy = 41864	(Sum of products of x and y data)
[EXE]	SUMXY :Σxy = 41864 MEANX :Σx/n = 165.6	(Mean of x data)
[EXE]	MEANX :Σx/n = 165.6 MEANX :Σy/n = 50.4	(Mean of y data)
[EXE]	MEANY :Σy/n = 50.4 SDXN :xσn = 6.711184694	(Population standard deviation of x data)

Key	Display	Note
	SDXN :xσn = 6.711184694 SDYN :yσn = 7.11617875	(Population standard deviation of y data)
	SDYN :yσn = 7.11617875 SDX :xσn-1 = 7.503332593	(Sample standard deviation of x data)
	SDX :xσn-1 = 7.503332593 SDY :yσn-1 = 7.956129712	(Sample standard deviation of y data)
	SDY :yσn-1 = 7.956129712 LRA :a =-120.7886323	(Linear regression constant term)
	LRA :a =-120.7886323 LRB :b = 1.03374778	(Linear regression coefficient)
	LRB :b = 1.03374778 COR :r = 0.9749154035	(Correlation coefficient)
	Regression analysis [y=a+bx] >In.Del.Clear.List.eoX.eoY.P ?_	(Return to menu display)
	Estimation of y [y=a+bx] x?_	(Estimation of weight)
0 EXE	Estimation of y [y=a+bx] x?170 :ŷ= 54.94849023	(Estimated value for weight following input of height)
	Estimation of y [y=a+bx] x?_	
	Regression analysis [y=a+bx] >In.Del.Clear.List.eoX.eoY.P ?_	(Return to menu display)

re, these data produce the line y= −120.7886323 + 1.03374778x. Also, input of a height 170cm results in an estimated weight of 54.9kg.

6520 LOGARITHMIC REGRESSION ANALYSIS

(y = a + blnx)

Performs logarithmic regression analysis on n data groups (x, y) and calculates the statistics listed below. Also determines the following on the logarithmic curve:

- Estimated value of x in relation to y (EOX)
- Estimated value of y in relation to x (EOY)

STATISTIC TABLE

Number of data items	CNT : n	
Sum of x data logarithmic values	SUMLNX : Σlnx	$\Sigma(lnx)$
Sum of y data	SUMY : Σy	
Sum of squares of x data logarithmic values	SUMLNX2 : Σlnx^2	$\Sigma(lnx)^2$
Sum of squares of y data	SUMY2 : Σy^2	
Sum of products of x data logarithmic values and of y data	SUMLNXY : $\Sigma lnxy$	$\Sigma((lnx)\cdot y)$
Mean of x data logarithmic values	MEANLNX : $\Sigma lnx/n$	$\Sigma(lnx)/n$
Mean of y data	MEANY : $\Sigma y/n$	
Population standard deviation of x data logarithmic values	SDLNXN : $lnx\sigma n$	$\sqrt{\frac{n\Sigma(lnx)^2-(\Sigma lnx)^2}{n^2}}$
Population standard deviation of y data	SDYN : $y\sigma n$	$\sqrt{\frac{n\Sigma y^2-(\Sigma y)^2}{n^2}}$
Sample standard deviation of x data logarithmic values	SDLNX : $lnx\sigma n_{-1}$	$\sqrt{\frac{n\Sigma(lnx)^2-(\Sigma lnx)^2}{n(n-1)}}$
Sample standard deviation of y data	SDY : $y\sigma n_{-1}$	$\sqrt{\frac{n\Sigma y^2-(\Sigma y)^2}{n(n-1)}}$
Regression constant term	RA : a	$\frac{\Sigma y-b\cdot\Sigma lnx}{n}$
Regression coefficient	RB : b	$\frac{n\Sigma(lnx)y-\Sigma lnx\cdot\Sigma y}{n\Sigma(lnx)^2-(\Sigma lnx)^2}$
Correlation coefficient	COR : r	$\frac{n\Sigma(lnx)y-\Sigma lnx\cdot\Sigma y}{\sqrt{(n\Sigma(lnx)^2-(\Sigma lnx)^2)(n\Sigma y^2-(\Sigma y)^2)}}$

OPERATION

6520 [LB]

```
Regression analysis  [y=a+blnx]
>In.Del.Clear.List.eoX.eoY.P ?_
```

The menu illustrated above is displayed for logarithmic regression calculations. The following seven items can be selected from this menu:

1. I : Data input
2. D: Data deletion (deletes erroneous or unnecessary data)
3. C: Data clear

L : Statistic display

Displays number of data items, sum of x data logarithmic values, sum of y data, sum of squares of x data logarithmic values, sum of squares of y data, sum of products of x data logarithmic values and y data, mean of x data logarithmic values, mean of y data, population standard deviation of x data logarithmic values, population standard deviation of y data, sample standard deviation of x data logarithmic values, sample standard deviation of y data, regression constant term, regression coefficient, and correlation coefficient. [⇩] (or [EXE]) scrolls to the following data item, [⇧] to the previous data item, and [⊜] or [⊜] terminates statistic display.

X : Calculates x value for y on logarithm curve.

Y : Calculates y value for x on logarithm curve.

P : Outputs all statistics to printer.

XAMPLE

ter the following measured data for microbes, perform logarithmic regression, and dis- y the statistics. Also estimate the number of microbes with a temperature of 18 degrees ng the logarithm curve obtained.

	1	2	3	4	5
Temperature (x)	5°	12°	20°	27°	36°
Microbes (y)	680	1100	1300	1440	1600

```
Regression analysis  [y=a+blnx]
clear data  (Y/N) ?
```
(Data clear)

```
Regression analysis  [y=a+blnx]
>In.Del.Clear.List.eoX.eoY.P ?_
```
(Data clear confirmation)

```
Input data (x,y)     [EXE]:menu
x?_              :y?
```
(Data input)

XE

```
Input data (x,y)     [EXE]:menu
x?5              :y?_
```
(x input)

0 [EXE]

```
Input data (x,y)     [EXE]:menu
x?_              :y?
```
(y input)

[EXE] 1100 [EXE] 20 [EXE] 1300 [EXE] 27 [EXE] 1440 [EXE] 36 [EXE] 1600 [EXE]

```
Input data (x,y)     [EXE]:menu
x?_              :y?
```
(Remaining x, y data input)

```
Regression analysis  [y=a+blnx]
>In.Del.Clear.List.eoX.eoY.P ?_
```
(Return to menu display)

```
CNT     :n     =  5
SUMlnx  :Σlnx  =  13.96943264
```
(Statistic display showing number of data and sum of x data logarithmic values)

```
SUMlnx  :Σlnx  =  13.96943264
SUMY    :Σy    =  6120
```
(Sum of y data)

```
SUMY    :Σy    =  6120
SUMlnX2 :Σlnx2 =  41.44361194
```
(Sum of squares of x data logarithmic values)

```
SUMlnX2 :Σlnx2 =  41.44361194
SUMY2   :Σy2   =  7996000
```
(Sum of squares of y data)

```
SUMY2   :Σy2   =  7996000
SUMlnXY :Σlnxy =  18201.90244
```
(Sum of products of x data logarithmic values and y data)

```
SUMlnXY :Σlnxy   =  18201.90244
MEANlnX :Σlnx/n  =  2.793886528
```
(Mean of x data logarithmic values)

```
MEANlnX :Σlnx/n  =  2.793886528
MEANY   :Σy/n    =  1224
```
(Mean of y data)

EXE	MEANY :Σy/n = 1224 SDlnXN :lnxσn = 0.6949247842	(Population standard deviation of x data logarithmic values)
EXE	SDlnXN :lnxσn = 0.6949247842 SDYN :yσn = 317.8427284	(Population standard deviation of y data)
EXE	SDYN :yσn = 317.8427284 SDlnX :lnxσn-1= 0.7769495284	(Sample standard deviation of x data logarithmic values)
EXE	SDlnX :lnxσn-1= 0.7769495284 SDY :yσn-1 = 355.3589734	(Sample standard deviation of y data)
EXE	SDY :yσn-1 = 355.3589734 RA :a =-52.62523046	(Regression constant term)
EXE	RA :a =-52.62523046 RB :b = 456.935247	(Regression coefficient)
EXE	RB :b = 456.935247 COR :r = 0.9990337973	(Correlation coefficient)
EXE	Regression analysis [y=a+blnx] >In.Del.Clear.List.eoX.eoY.P ?_	(Return to menu display)
Y	Estimation of y [y=a+blnx] x?_	(Estimation of y)
18 EXE	Estimation of y [y=a+blnx] x?18 ŷ= 1268.087503	(Estimated value for y following input of 18 degrees)
EXE	Estimation of y [y=a+blnx] x?_	
EXE	Regression analysis [y=a+blnx] >In.Del.Clear.List.eoX.eoY.P ?_	(Return to menu display)

Here, these data produce the curve y = −52.62523046 + 456.935247·lnx. Also, input of a temperature of 18 degrees results in an estimated total of 1,268 microbes.

6530 EXPONENTIAL REGRESSION ANALYSIS
($y = ab \hat{} x$)

erforms exponential regression analysis on n data groups (x, y) and calculates the statis-
:s listed below. Also determines the following on the exponential curve:

Estimated value of x in relation to y (EOX)
Estimated value of y in relation to x (EOY)

TATISTIC TABLE

umber of data items	CNT : n	
ım of x data	SUMX : Σx	
ım of y data logarithmic values	SUMLNY : Σlny	$\Sigma(lny)$
ım of squares of x data	SUMX2 : Σx^2	
ım of squares of y data garithmic values	SUMLNY2 : Σlny^2	$\Sigma(lny)^2$
ım of products of x data and data logarithmic values	SUMXLNY : $\Sigma xlny$	
ean of x data	MEANX : $\Sigma x/n$	
ean of y data logarithmic values	MEANLNY : $\Sigma lny/n$	
opulation standard deviation of data	SDXN : $x\sigma n$	$\sqrt{\frac{n\Sigma x^2-(\Sigma x)^2}{n^2}}$
opulation standard deviation of data logarithmic values	SDLNYN : $lny\sigma n$	$\sqrt{\frac{n\Sigma(lny)^2-(\Sigma lny)^2}{n^2}}$
ample standard deviation of data	SDX : $x\sigma n_{-1}$	$\sqrt{\frac{n\Sigma x^2-(\Sigma x)^2}{n(n-1)}}$
ample standard deviation of data logarithmic values	SDLNY : $lny\sigma n_{-1}$	$\sqrt{\frac{n\Sigma(lny)^2-(\Sigma lny)^2}{n(n-1)}}$
egression constant term	RA : a	$EXP\left(\frac{\Sigma(lny)-b\cdot\Sigma x}{n}\right)$
egression coefficient	RB : b	$EXP\left(\frac{n\Sigma xlny-\Sigma x\cdot\Sigma lny}{n\Sigma x^2-(\Sigma x)^2}\right)$
orrelation coefficient	COR : r	$\frac{n\Sigma xlny-\Sigma x\cdot\Sigma lny}{\sqrt{(n\Sigma x^2-(\Sigma x)^2)(n\Sigma(lny)^2-(\Sigma lny)^2}}$

'ERATION

;30 [UB] ·

```
Regression analysis    (y=ab^x)
>In.Del.Clear.List.eoX.eoY.P ?_
```

e menu illustrated above is displayed for exponential regression calculations. The follow- ; seven items can be selected from this menu:

I : Data input
D: Data deletion (deletes erroneous or unnecessary data)
C: Data clear

4. L: Statistic display
   Displays number of data items, sum of x data, sum of y data logarithmic values, sum of squares of x data, sum of squares of y data logarithmic values, sum of products of x data and y data logarithmic values, mean of x data, mean of y data logarithmic values, population standard deviation of x data, population standard deviation of y data logarithmic values, sample standard deviation of x data, sample standard deviation of y data logarithmic values, regression constant term, regression coefficient, and correlation coefficient. [↓] (or [EXE]) scrolls to the following data item, [↑] to the previous data item, and [⇦] or [⇨] terminate statistic display.
5. X: Calculates x value for y on regression line. .
6. Y: Calculates y value for x on regression line.
7. P: Outputs all statistics to printer.

EXAMPLE

Enter the following data for the amount of sales per customer and number of customers for a store, perform exponential regression, and display the statistics. Also estimate the amount of sales per customer for 150 customers using the exponential curve obtained.

	1	2	3	4	5
Customers (x)	115	124	130	138	142
Sales/customer (y) ($)	40	41.6	43.0	46.0	46.5

Key	Display	
C	Regression analysis [y=ab^x] clear data (Y/N) ?	(Data clear)
Y	Regression analysis [y=ab^x] >In.Del.Clear.List.eoX.eoY.P ?_	(Data clear confirmation)
I	Input data (x.y) [EXE]:menu x?_ :y?	(Data input)
115 [EXE]	Input data (x.y) [EXE]:menu x?115 :y?_	(x input)
40 [EXE]	Input data (x.y) [EXE]:menu x?_ :y?	(y input)

124 [EXE] 41.6 [EXE] 130 [EXE] 43 [EXE] 138 [EXE] 46 [EXE] 142 [EXE] 46.5 [EXE]

Key	Display	
	Input data (x.y) [EXE]:menu x?_ :y?	(Remaining x, y data input)
[EXE]	Regression analysis [y=ab^x] >In.Del.Clear.List.eoX.eoY.P ?_	(Return to menu display)
L	CNT n = 5 SUMX Σx = 649	(Statistic display showing number of data and sum of x data)
[EXE]	SUMX :Σx = 649 SUMlnY :Σlny = 18.84527345	(Sum of y data logarithmic values)
[EXE]	SUMlnY :Σlny = 18.84527345 SUMX2 :Σx² = 84709	(Sum of squares of x data)
[EXE]	SUMX2 :Σx² = 84709 SUMlnY2 :Σlny² = 71.0530778	(Sum of squares of y data logarithmic values)
[EXE]	SUMlnY2 :Σlny² = 71.0530778 SUMXlnY :Σxlny = 2449.016314	(Sum of products of x data and y data logarithmic values)
[EXE]	SUMXlnY :Σxlny = 2449.016314 MEANX :Σx/n = 129.8	(Mean of x data)

Key	Display	Explanation
	MEANX :Σx/n = 129.8 MEANlnY :Σlny/n= 3.769254689	(Mean of y data logarithmic values)
	MEANlnY :Σlny/n= 3.769254689 SDXN :xσn = 9.68297475	(Population standard deviation of x data)
	SDXN :xσn = 9.68297475 SDlnYN :lnyσn = 5.774640647E-02	(Population standard deviation of y data logarithmic values)
	SDlnYN :lnyσn = 5.774640647E-02 SDX :xσn-1 = 10.82589488	(Sample standard deviation of x data)
	SDX :xσn-1 = 10.82589488 SDlnY :lnyσn-1= 6.456244516E-02	(Sample standard deviation of y data logarithmic values)
	SDlnY :lnyσn-1= 6.456244516E-02 RA :a = 20.1317721	(Regression constant term)
	RA :a = 20.1317721 RB :b = 1.005926239	(Regression coefficient)
	RB :b = 1.005926239 COR :r = 0.9907846423	(Correlation coefficient)
	Regression analysis [y=ab^x] >In.Del.Clear.List.eoX.eoY.P ?_	(Return to menu display)
	Estimation of y [y=ab^x] x?_	(Estimation of y)
EXE	Estimation of y [y=ab^x] x?150 :ŷ= 48.84301552	(Estimated value for y following input of 150 customers)
	Estimation of y [y=ab^x] x?_	
	Regression analysis [y=ab^x] In.Del.Clear.List.eoX.eoY.P ?_	(Return to menu display)

e, these data produce the curve $y = 20.1317721 \times 1.005926239^x$. Also, input of a total 50 customers results in an estimated amount per customer of $48.843.

6540 POWER REGRESSION ANALYSIS
(y=ax^b)

Performs power regression analysis on n data groups (x, y) and calculates the statistics listed below. Also determines the following on the power curve:

- Estimated value of x in relation to y (EOX)
- Estimated value of y in relation to x (EOY)

STATISTIC TABLE

Number of data items	CNT	: n	
Sum of x data logarithmic values	SUMLNX	: Σlnx	
Sum of y data logarithmic values	SUMLNY	: Σlny	
Sum of squares of x data logarithmic values	SUMLNX2	: Σlnx^2	$\Sigma(lnx)^2$
Sum of squares of y data logarithmic values	SUMLNY2	: Σlny^2	$\Sigma(lny)^2$
Sum of products of x data logarithmic values and y data logarithmic values	SUMLNXLNY	: $\Sigma lnxlny$	$\Sigma(lnx \cdot lny)$
Mean of x data logarithmic values	MEANLNX	: $\Sigma lnx/n$	
Mean of y data logarithmic values	MEANLNY	: $\Sigma lny/n$	
Population standard deviation of x data logarithmic values	SDLNXN	: $lnx\sigma n$	$\sqrt{\frac{n\Sigma(lnx)^2-(\Sigma lnx)^2}{n^2}}$
Population standard deviation of y data logarithmic values	SDLNYN	: $lny\sigma n$	$\sqrt{\frac{n\Sigma(lny)^2-(\Sigma lny)^2}{n^2}}$
Sample standard deviation of x data logarithmic values	SDLNX	: $lnx\sigma n\text{-}1$	$\sqrt{\frac{n\Sigma(lnx)^2-(\Sigma lnx)^2}{n(n-1)}}$
Sample standard deviation of y data logarithmic values	SDLNY	: $lnx\sigma n\text{-}1$	$\sqrt{\frac{n\Sigma(lny)^2-(\Sigma lnx)^2}{n(n-1)}}$
Regression constant term	RA	: a	$\frac{\Sigma lny-b\cdot\Sigma lnx}{n}$
Regression coefficient	RB	: b	$\frac{n\Sigma lnx\cdot lny-\Sigma lnx\cdot\Sigma lny}{n\Sigma(lnx)^2-(\Sigma lnx)^2}$
Correlation coefficient	COR	: c	$\frac{n\Sigma lnx\cdot lny-\Sigma lnx\Sigma lny}{\sqrt{(n\Sigma(lnx)^2-(\Sigma lnx)^2)(n\Sigma(lny)^2-(\Sigma lny)^2)}}$

OPERATION

6540 [UB]

```
Regression analysis      [y=ax^b]
>In.Del.Clear.List.eoX.eoY.P ?_
```

The menu illustrated above is displayed for power regression calculations. The following seven items can be selected from this menu:

1. I : Data input
2. D: Data deletion (deletes erroneous or unnecessary data)
3. C: Data clear

L: Statistic display
Displays number of data items, sum of x data logarithmic values, sum of y data logarithmic values, sum of squares of x data logarithmic values, sum of squares of y data logarithmic values, sum of products of x data logarithmic values and y data logarithmic values, mean of x data logarithmic values, mean of y data logarithmic values, population standard deviation of x data logarithmic values, population standard deviation of y data logarithmic values, sample standard of x data logarithmic values, sample standard of y data logarithmic values, regression constant term, regression coefficient, and correlation coefficient.
[↓] (or [EXE]) scrolls to the following data item, [↑] to the previous data item, and [⊜] or [⊜] terminate statistic display.

X: Calculates x value for y on power curve.
Y: Calculates y value for x on power curve.
P: Outputs all statistics to printer.

[AMPLE]

...r the following data for the characteristics of voltage and current for a semiconductor, ...orm power regression, and display the statistics. Also produce an estimated value for ...ent at 40V.

	1	2	3	4	5
Voltage (x)	10	15	20	25	30
Current (y)	13	22	31	38	43

	Display	
	Regression analysis [y=ax^b] clear data (Y/N) ?	(Data clear)
	Regression analysis [y=ax^b] >In.Del.Clear.List.eoX.eoY.P ?_	(Data clear confirmation)
	Input data (x.y) [EXE]:menu x?_ :y?	(Data input)
[EXE]	Input data (x.y) [EXE]:menu x?10 :y?_	(x input)
[EXE]	Input data (x.y) [EXE]:menu x?_ :y?	(y input)

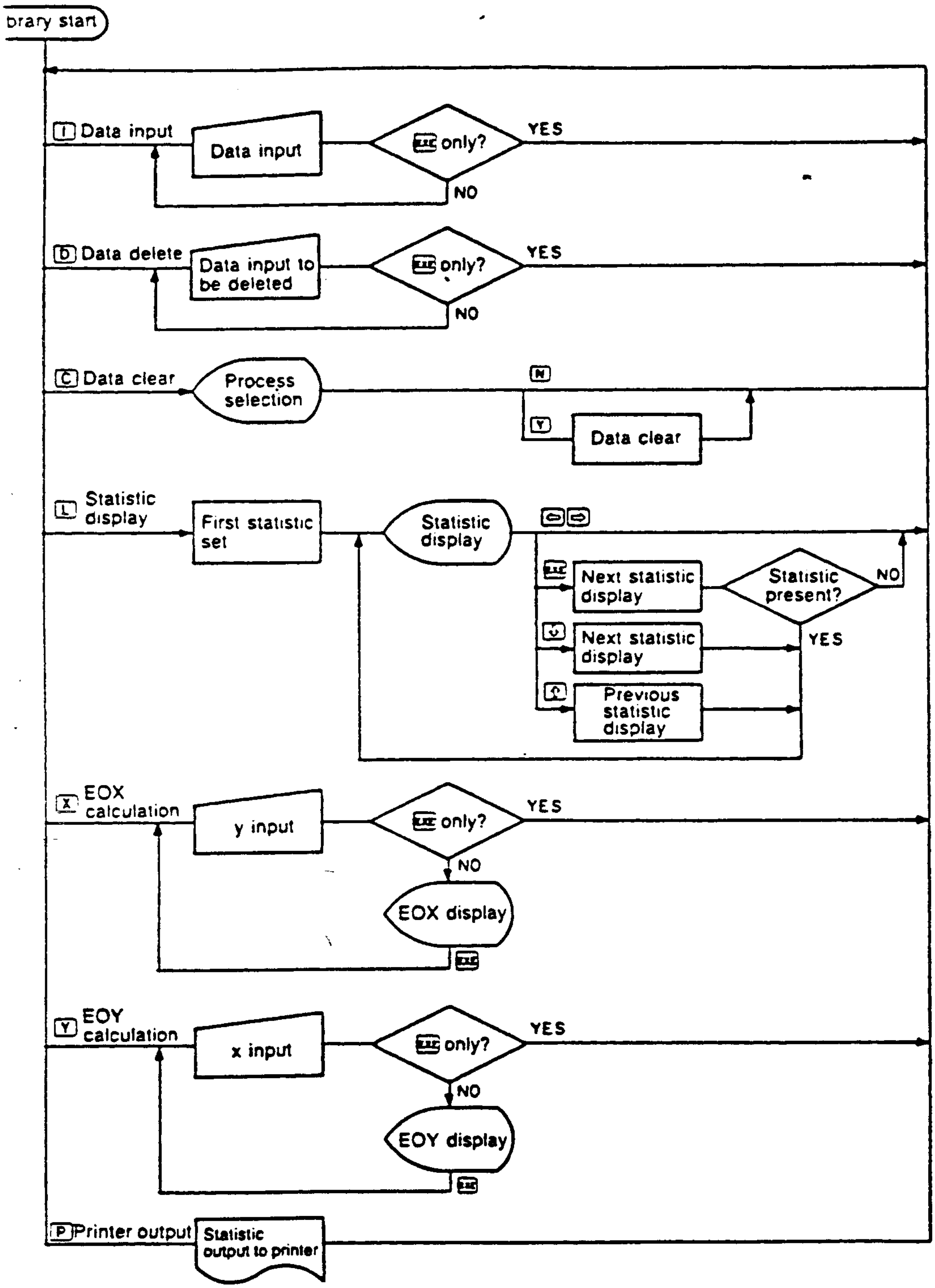
[EXE] 22 [EXE] 20 [EXE] 31 [EXE] 25 [EXE] 38 [EXE] 30 [EXE] 43 [EXE]

Display	
Input data (x.y) [EXE]:menu x?_ :y?	(Remaining x, y data input)
Regression analysis [y=ax^b] >In.Del.Clear.List.eoX.eoY.P ?_	(Return to menu display)
CNT :n = 5 SUMlnX :Σlnx = 14.62644077	(Statistic display showing number of data and sum of x data logarithmic values)
SUMlnX :Σlnx = 14.62644077 SUMlnY :Σlny = 16.48876529	(Sum of y data logarithmic values)
SUMlnY :Σlny = 16.48876529 SUMlnX2 :Σlnx² = 43.53915106	(Sum of squares of x data logarithmic values)
SUMlnX2 :Σlnx² = 43.53915106 SUMlnY2 :Σlny² = 55.30443616	(Sum of squares of y data logarithmic values)

Key	Display	Explanation
EXE	SUMlnY2 :Σlny2 = 55.30443616 SUMlnXlnY:Σlnxlny= 49.06554072	(Sum of products of x data logarithmic values and y data logarithmic values)
EXE	SUMlnXlnY:Σlnxlny= 49.06554072 MEANlnX :Σlnx/n= 2.925288155	(Mean of x data logarithmic values)
EXE	MEANlnX :Σlnx/n= 2.925288155 MEANlnY :Σlny/n= 3.297753058	(Mean of y data logarithmic values)
EXE	MEANlnY :Σlny/n= 3.297753058 SDlnXN :lnxσn = 0.3879683282	(Population standard deviation of x data logarithmic values)
EXE	SDlnXN :lnxσn = 0.3879683282 SDlnYN :lnyσn = 0.4309431503	(Population standard deviation of y data logarithmic values)
EXE	SDlnYN :lnyσn = 0.4309431503 SDlnX :lnxσn-1= 0.4337617775	(Sample standard deviation of x data logarithmic values)
EXE	SDlnX :lnxσn-1= 0.4337617775 SDlnY :lnyσn-1= 0.4818090893	(Sample standard deviation of y data logarithmic values)
EXE	SDlnY :lnyσn-1= 0.4818090893 RA :a = 1.069436811	(Regression constant term)
EXE	RA :a = 1.069436811 RB :b = 1.104376978	(Regression coefficient)
EXE	RB :b = 1.104376978 COR :r = 0.9942455045	(Correlation coefficient)
EXE	Regression analysis [y=ax^b] >In.Del.Clear.List.eoX.eoY.P ?_	(Return to menu display)
Y	Estimation of y [y=ax^b] x?_	(Estimation of y)
40 EXE	Estimation of ŷ [y=ax^b] x?40 :ŷ= 62.6685293	(Estimated value for y following input of 40 volts)
EXE	Estimation of y [y=ax^b] x?_	
EXE	Regression analysis [y=ax^b] >In.Del.Clear.List.eoX.eoY.P ?_	(Return to menu display)

Here, these data produce the power curve $y = 1.069436811 \times x^{1.104376978}$. Also, input of 40 volts results in an estimated current of 62.9mA.

EGRESSION ANALYSIS FLOWCHART (6510, 6520, 6530, 6540)



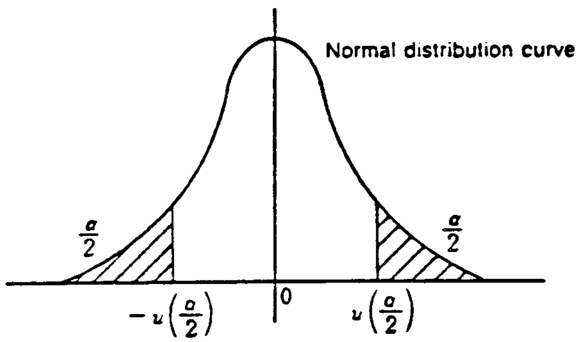
6610 MEAN INTERVAL ESTIMATION (FOR KNOWN VARIANCE)

Performs estimation of the confidence interval of μ in normal distribution N $(\mu, \sigma^2$; where μ : unknown, σ^2 : known).

CALCULATIONS

When an n-size sample $(x_1, x_2 \cdots x_n)$ is taken from normal distribution N (μ, σ^2), the following confidence interval $(1-\alpha)$ of confidence level for μ is obtained:

$$\bar{x} - u\left(\frac{\alpha}{2}\right)\frac{\sigma}{\sqrt{n}} < \mu < \bar{x} + u\left(\frac{\alpha}{2}\right)\frac{\sigma}{\sqrt{n}}$$



μ : population mean
σ^2: population variance
$\bar{x}$: sample mean
α : significance level
$1-\alpha$: confidence level

OPERATION

6610 LIB

```
N(μ,σ2) a<μ<b   σ2:known
input new data (Y/N) ?
```

The display appears as indicated above once the library is activated. At this time, either Y or N should be pressed to perform the following procedures:

Y : New data input followed by interval estimation, additional data input, data edit, statistic check.
N : Interval estimation using previously stored data, interval estimation when data are known.

(1) Y

Y

```
Input data (x)
>Input.Delete.Clear.List.End ?_
```

The menu display illustrated above appears when Y is pressed. One of the following character keys is then pressed to perform the corresponding function.

I (Input) : Data input (for input or addition of data).
D (Delete) : Data delete (for deletion of erroneous or unnecessary data).
C (Clear) : Data clear (for deletion of previously stored data. This operation also clears statistics).

(List) : Statistic display (for display of number of data items, sum, sum of squares, mean, population standard deviation, sample standard deviation).
[↓] (or [EXE]) scrolls to the following data item, [↑] to the previous item, and [⊟] or [⊟] exits the statistic display and returns to the menu.

(End) : Advances to the interval estimation display (same as when N is pressed in the first step above).

N

```
N(μ.σ²) a<μ<b   σ²:known
n= 0?_
```
(Interval estimation display)

› display appears as illustrated above when the [N] key is pressed. The value indicated n shows the number of data currently stored in memory.

= 0 : Interval estimation cannot be performed, so this should be corrected to the required ata.

ımber of data input (following [Y] above) and value of n differ : Confirm that some data ave not been omitted during the input or that two or more data items have been input ›gether for a single entry. In either case, terminate the library operation. Enter the library gain and add, delete, or reinput data as required.

umber of data input (following [Y] above) matches value of n : Press [EXE]

AMPLE

› table below shows the number of customers at a store over a 5-day period. Using this a, perform interval estimation for the number of customers with a confidence level of 99%. › population standard deviation of the customers is previously known to be 120.3.

	1	2	3	4	5
NUMBER OF CUSTOMERS	580	430	612	498	591

```
N(μ.σ²) a<μ<b   σ²:known
input new data (Y/N) ?
```

```
Input data (x)
>Input.Delete.Clear.List.End ?_
```
(Select new data input.)

```
Input data (x)
clear data  (Y/N)?
```
(Select data clear.)

```
Input data (x)
>Input.Delete.Clear.List.End ?_
```
(Data cleared.)

```
Input data (x)          [EXE]:menu
x?_
```
(Select data input.)

[EXE]

```
Input data (x)          [EXE]:menu
x?_
```
(Enter first data item.)

[EXE] 612 [EXE] 498 [EXE] 591 [EXE]

```
Input data (x)          [EXE]:menu
x?_
```
(Enter remaining data items.)

```
Input data (x)
>Input.Delete.Clear.List.End ?_
```
(Return to menu.)

```
N(μ.σ²) a<μ<b   σ²:known
n= 5 ?_
```
(Select End to proceed to interval estimation.)

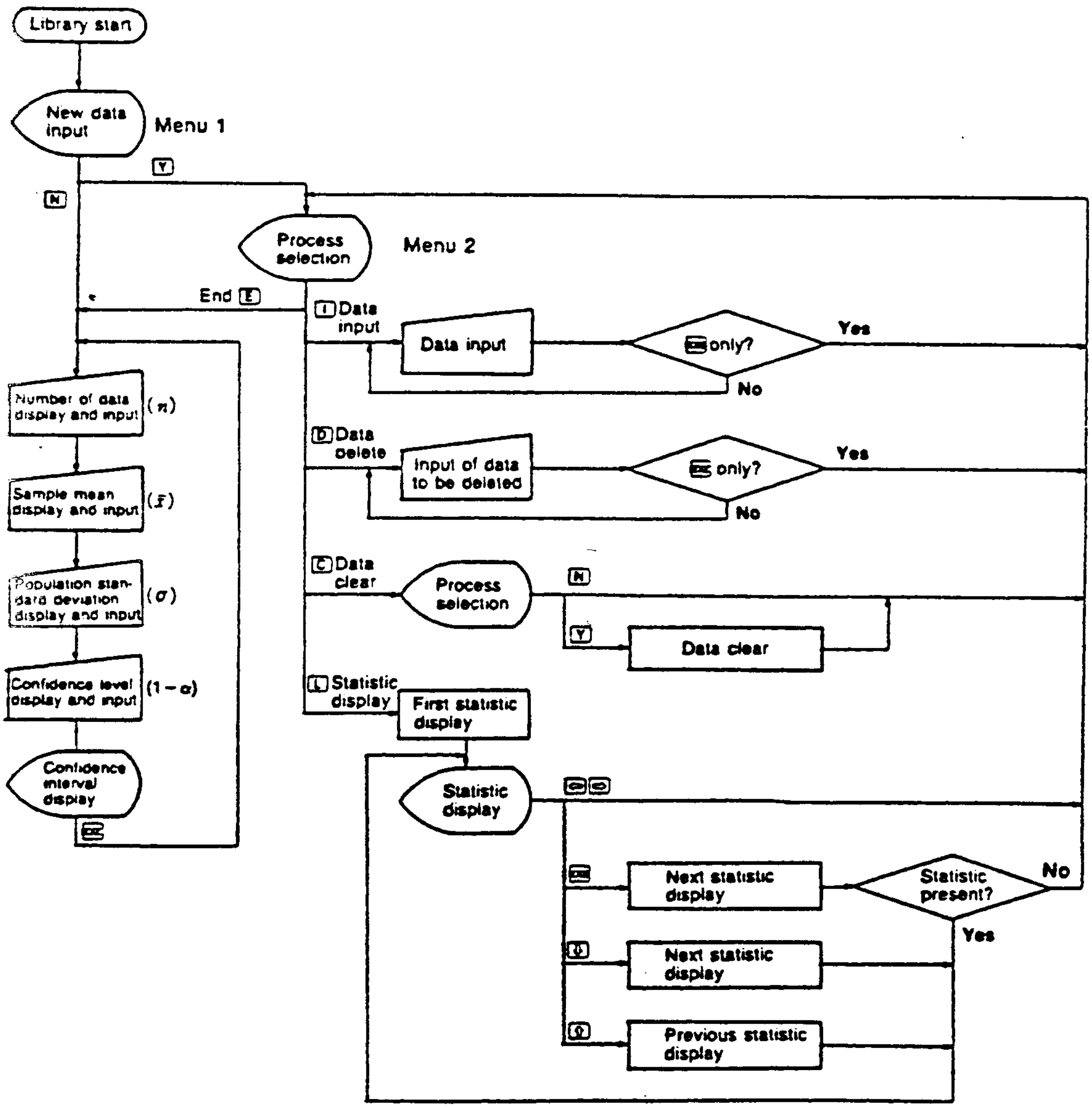
```
N(μ.σ²) a<μ<b   σ²:known
x̄= 542.2 ?_
```
(Press [EXE] after checking number of data items.)

```
N(μ.σ²) a<μ<b   σ²:known
c= 0 ?_
```
(Press [EXE] after checking mean.)

120.3 EXE	Confidence level (1−α)(%) 1−α= 95 ?_	(Press EXE after inputting population standard deviation.)
99 EXE	N(μ.σ²) 99 % 403.6 < μ < 680.8	(Enter confidence level to display mean confidence interval.)
EXE	N(μ.σ²) a<μ<b σ²:known n= 5 ?_	

Here, it is determined that the mean for number of customers μ with a confidence level of 99% is $403.6<\mu<680.8$.

MEAN INTERVAL ESTIMATION FLOWCHART (FOR KNOWN VARIANCE)



6620 MEAN INTERVAL ESTIMATION (FOR UNKNOWN VARIANCE)

rforms estimation of the confidence interval of μ in normal distribution N (μ, σ^2 ; where unknown, σ^2 : unknown).

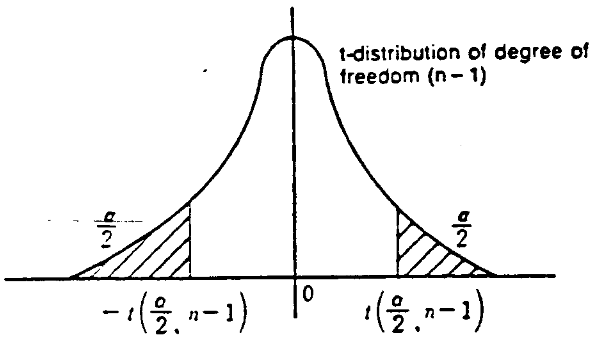
LCULATIONS

hen an n-size sample (x_1, $x_2 \cdots x_n$) is taken from normal distribution N (μ, σ^2),

$$\bar{x} - t\left(\frac{\alpha}{2},\ n-1\right)\sqrt{\frac{V}{n}} < \mu < \bar{x} + t\left(\frac{\alpha}{2},\ n-1\right)\sqrt{\frac{V}{n}}$$

obtained in accordance with degree of freedom (n − 1) of the t-distribution.

(μ : population mean
σ^2 : population variance
α : significance level
$\bar{x}$: sample mean
V : unbiased variance
$1-\alpha$: confidence level)



$$V = \frac{\sum(x-\bar{x})^2}{n-1}$$

ERATION

620 UB

```
N(μ.σ²) a<μ<b
input new data (Y/N) ?
```

e display appears as indicated above once the library is activated. At this time, either Y N should be pressed to perform the following procedures:

New data input followed by interval estimation, additional data input, data edit, statistic check.
Interval estimation using previously stored data, interval estimation by inputting each value.

Y

```
Input data (x)
>Input.Delete.Clear.List.End ?_
```

e menu display illustrated above appears when Y is pressed. One of the following character s is then pressed to perform the corresponding function.

(Input) : Data input (for input or addition of data).
(Delete) : Data delete (for deletion of erroneous or unnecessary data).
(Clear) : Data clear (for deletion of previously stored data. This operation also clears statistics).

L (List) : Statistic display (for display of number of data items, sum, sum of squares, mean, population standard deviation, sample standard deviation).
[↓] (or [EXE]) scrolls to the following data item, [↑] to the previous item, and [⊖] or [⊖] exits the statistic display and returns to the menu.

E (End) : Advances to the interval estimation display (same as when N is pressed in the first step above).

(2) N

Key	Display	
[N]	N (μ . σ²) a<μ<b n= 5 ?_	(Interval estimation display)

The display appears as illustrated above when the [N] key is pressed. The value indicated for n shows the number of data currently stored in memory.

- n = 0 : Interval estimation cannot be performed, so this should be corrected to the required data.
- Number of data input (following [Y] above) and value of n differ : Confirm that some data have not been omitted during the input or that two or more data items have been input together for a single entry. In either case, terminate the library operation. Enter the library again and add, delete, or reinput data as required.
- Number of data input (following [Y] above) matches value of n : Press [EXE].

EXAMPLE

The table below shows the number of customers over a 5-day period for five drugstores selected at random in a certain area. Using this data, perform interval estimation for the number of customers at one drugstore with a confidence level of 95%.

	1	2	3	4	5
NUMBER OF CUSTOMERS	245	366	271	493	306

Key	Display	
	N (μ . σ²) a<μ<b input new data (Y/N) ?	
[Y]	Input data (x) >Input.Delete.Clear.List.End ?_	(Select new data input.)
[C]	Input data (x) clear data (Y/N) ?	(Select data clear.)
[Y]	Input data (x) >Input.Delete.Clear.List.End ?_	(Data cleared.)
[I]	Input data (x) [EXE]:menu x?_	(Select data input.)
245 [EXE]	Input data (x) [EXE]:menu x?_	(Enter first data item.)
366 [EXE] 271 [EXE] 493 [EXE] 306 [EXE]	Input data (x) [EXE]:menu x?_	(Enter remaining data items.)
[EXE]	Input data (x) >Input.Delete.Clear.List.End ?_	(Return to menu)
[E]	N (μ . σ²) a<μ<b n= 5 ?_	(Select End to proceed to interval estimation.)
[EXE]	N (μ . σ²) a<μ<b x̄= 336.2 ?_	(Press [EXE] after checking number of data items.)
[EXE]	N (μ . σ²) a<μ<b V= 9738.7 ?_	(Press [EXE] after checking mean.)

```
Confidence level (1-α)[%]
1-α= 95 ?_
```

(Press [EXE] after checking unbiased variance.)

```
N(μ.σ2)       95 %
. . . . .
```

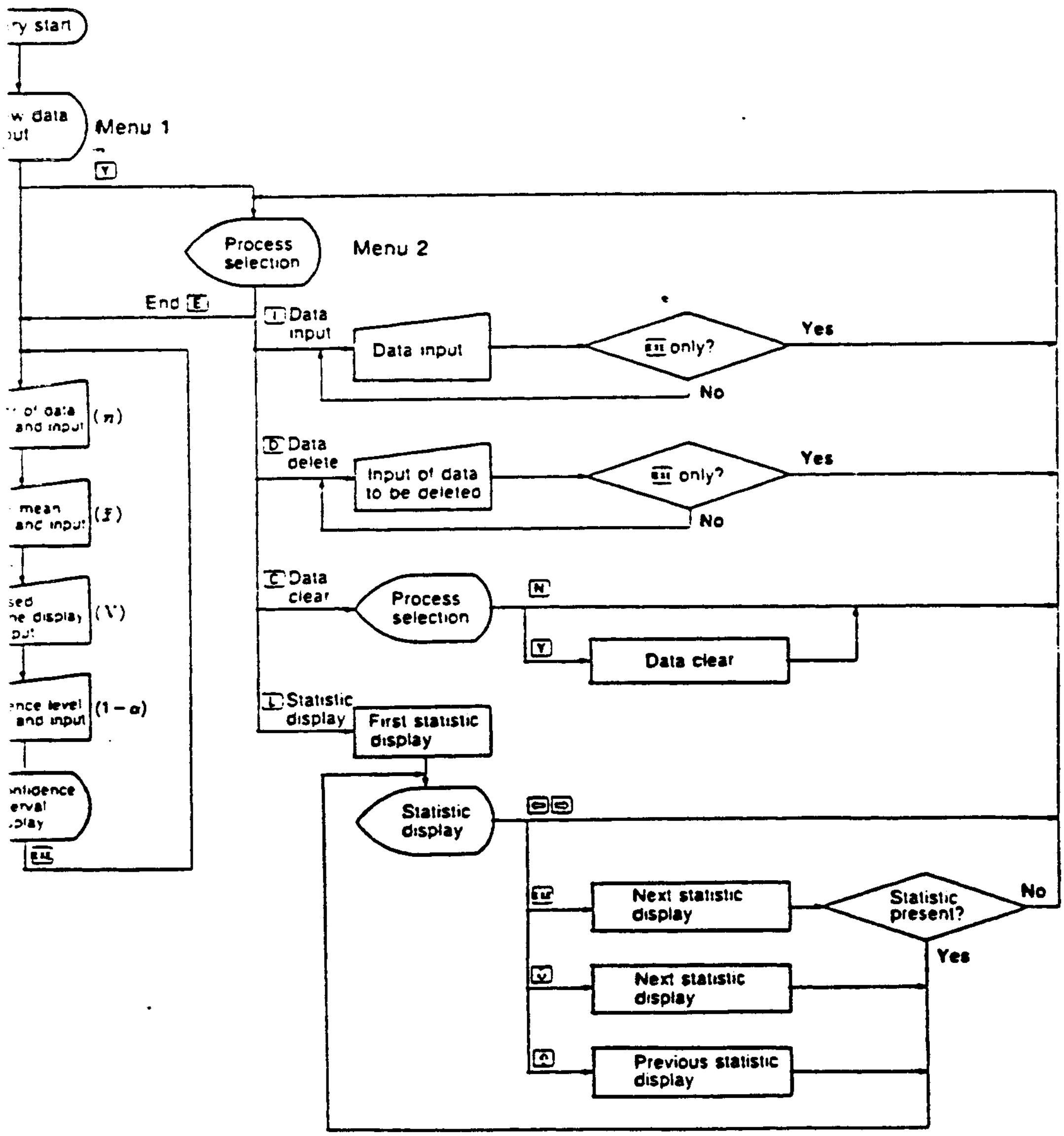
```
N(μ.σ2)       95 %
 213.7 < μ < 458.7
```

(Confidence level of 95% is already set, so confidence interval is displayed after [EXE] is pressed.)

```
N(μ.σ2)  a<μ<b
n= 5 ?_
```

…e, it is determined that the mean for number of customers μ with a confidence level of …% is $213.7<\mu<458.7$.

…AN INTERVAL ESTIMATION FLOWCHART (FOR UNKNOWN VARIANCE)



6630 VARIANCE INTERVAL ESTIMATION

Performs estimation of the confidence interval of σ^2 in normal distribution N (μ, σ^2 ; where μ : unknown, σ^2 : unknown).

CALCULATIONS

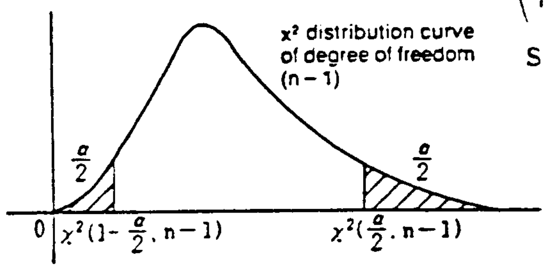
When an n-size sample (x_1, $x_2 \cdots x_n$) is taken from normal distribution N (μ, σ^2), the confidence interval of the confidence level $(1-\alpha)$ of σ^2 is obtained by

$$\frac{S}{\chi^2\left(\frac{\alpha}{2} n-1\right)} < \sigma^2 < \frac{S}{\chi^2\left(1-\frac{\alpha}{2},\ n-1\right)}$$

in accordance with x^2 distribution of the degree of freedom $(n-1)$.

(μ : population mean
σ^2 : population variance
α : significance level
S : sum of squares
$1-\alpha$: confidence level)

$S = \sum (x - \bar{x})^2$



OPERATION

6630 [LIB]

```
N(μ.σ²)  a<σ²<b
input new data (Y/N) ?
```

The display appears as indicated above once the library is activated. At this time, either Y or N should be pressed to perform the following procedures:

Y : New data input followed by interval estimation, additional data input, data edit, statistic check.

N : Interval estimation using previously stored data, interval estimation by inputting each value.

(1) Y

[Y]

```
Input data (x)
>Input.Delete.Clear.List.End ?_
```

The menu display illustrated above appears when [Y] is pressed. One of the following character keys is then pressed to perform the corresponding function.

I (Input) : Data input (for input or addition of data).
D (Delete) : Data delete (for deletion of erroneous or unnecessary data).
C (Clear) : Data clear (for deletion of previously stored data. This operation also clears statistics).
L (List) : Statistic display (for display of number of data items, sum, sum of squares, mean, population standard deviation, sample standard deviation).
[⇩] (or [EXE]) scrolls to the following data item, [⇧] to the previous item, and [⇦] or [⇨] exits the statistic display and returns to the menu.
E (End) : Advances to the interval estimation display (same as when N is pressed in the first step above).

(2) N

Key	Display	Note
[N]	N(μ . σ2) a<σ2<b n= 5 ?_	(Interval estimation display)

The display appears as illustrated above when the [N] key is pressed. The value indicated for n shows the number of data currently stored in memory.

- n = 0 : Interval estimation cannot be performed, so this should be corrected to the required data.
- Number of data input (following [Y] above) and value of n differ : Confirm that some data have not been omitted during the input or that two or more data items have been input together for a single entry. In either case, terminate the library operation. Enter the library again and add, delete, or reinput data as required.
- Number of data input (following [Y] above) matches value of n : Press [EXE].

EXAMPLE

The table below shows the number of pins contained in five different boxes of the same size produced by the same manufacturer. Using this data, perform interval estimation with 99% confidence level for the pin variance.

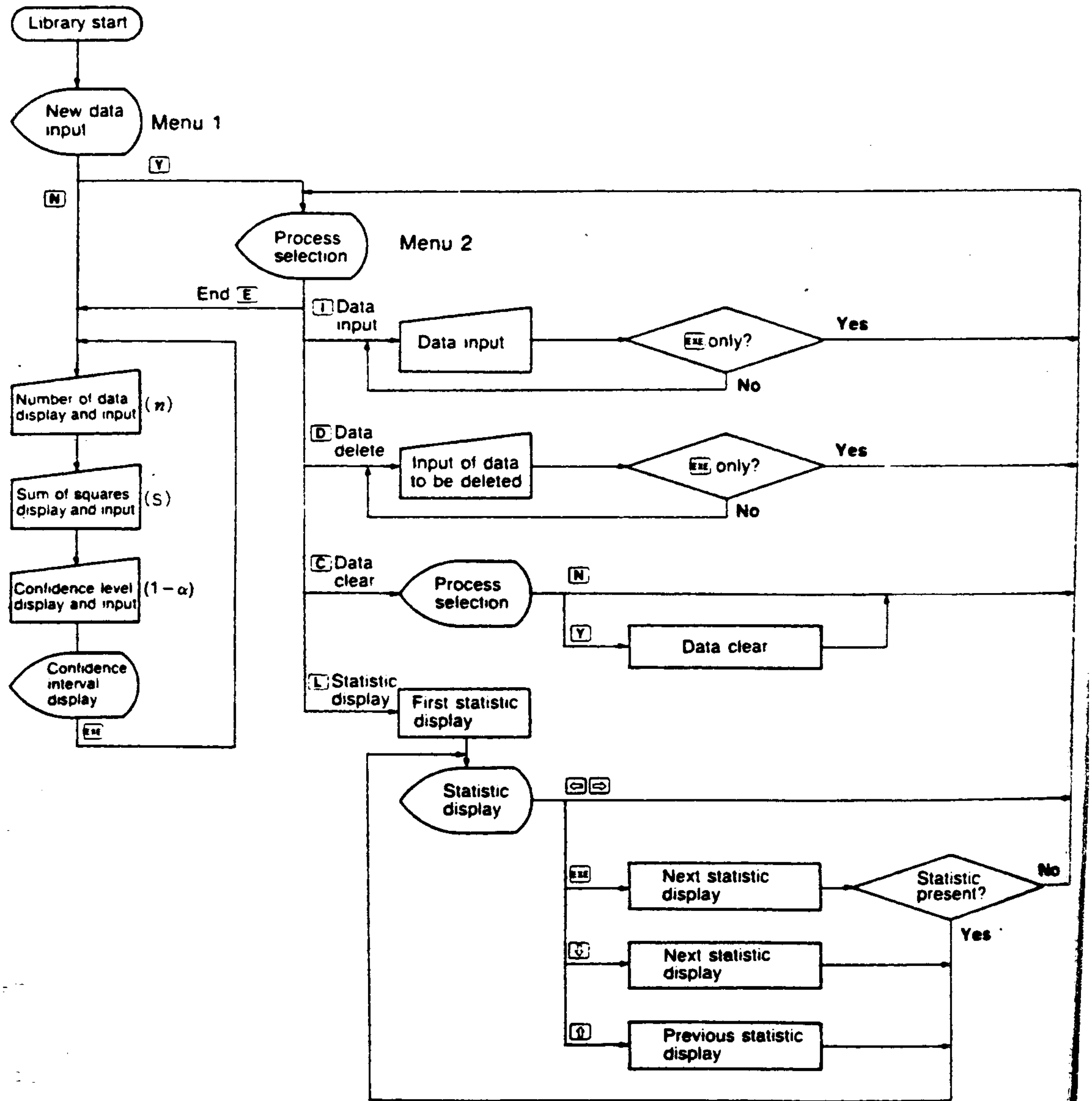
	1	2	3	4	5
NUMBER OF PINS	102	99	100	103	101

Key	Display	Note
	N(μ . σ2) a<σ2<b input new data (Y/N) ?	
[Y]	Input data (x) >Input.Delete.Clear.List.End ?_	(Select new data input.)
[C]	Input data (x) clear data (Y/N) ?	(Select data clear.)
[Y]	Input data (x) >Input.Delete.Clear.List.End ?_	(Data cleared.)
[I]	Input data (x) [EXE]:menu x?_	(Select data input.)
102 [EXE]	Input data (x) [EXE]:menu x?_	(Enter first data item.)
99 [EXE] 100 [EXE] 103 [EXE] 101 [EXE]	Input data (x) [EXE]:menu x?_	(Enter remaining data items.)
[EXE]	Input data (x) >Input.Delete.Clear.List.End ?_	(Return to menu.)

E	N(μ, σ²) a<σ²<b n= 5 ?_	(Select End to proceed to interval estimation.)
EXE	N(μ, σ²) a<σ²<b S= 10 ?_	(Press EXE after checking number of data items.)
EXE	Confidence level (1−α)[%] 1−α= 95 ?_	(Press EXE after checking sum of squares.)
99 EXE	N(μ, σ²) 99 %	(Enter confidence level value to display confidence interval.)
	N(μ, σ²) 99 % 0.6729 < σ² < 48.31	
EXE	N(μ, σ²) a<σ²<b n= 5 ?_	

Here, it is determined that the variance of the number of pins σ^2 with a confidence level of 99% is $0.6729<\sigma^2<48.31$.

VARIANCE INTERVAL ESTIMATION FLOWCHART



6640 STANDARD DEVIATION INTERVAL ESTIMATION

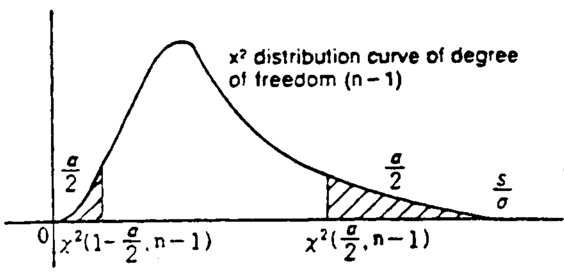
erforms estimation of the confidence interval of σ in normal distribution N (μ, σ^2 ; where : unknown, σ^2 : unknown).

ALCULATIONS

hen an n-size sample (x_1, $x_2 \cdots x_n$) is taken from normal distribution N (μ, σ^2), the confince interval of the confidence level ($1-\alpha$) of σ^2 is obtained by

$$\sqrt{\frac{S}{\chi^2(\frac{\alpha}{2}, n-1)}} < \sigma < \sqrt{\frac{S}{\chi^2(1-\frac{\alpha}{2}, n-1)}}$$

accordance with the x^2 distribution of the degree of freedom ($n-1$).



μ : population mean
σ^2 : population variance
α : significance level
S : sum of squares
$1-\alpha$: confidence level

$S = \Sigma(x-\bar{x})^2$

ERATION

40 [LIB]

```
N(μ.σ²) a<σ<b
input new data (Y/N) ?
```

display appears as indicated above once the library is activated. At this time, either Y N should be pressed to perform the following procedures:

New data input followed by interval estimation, additional data input, data edit, statistic check.
Interval estimation of previously stored data, interval estimation by inputting each value.

Y

```
Input data (x)
>Input.Delete.Clear.List.End ?_
```

menu display illustrated above appears when [Y] is pressed. One of the following character is then pressed to perform the corresponding function.

nput) : Data input (for input or addition of data).
elete) : Data delete (for deletion of erroneous or unnecessary data).
lear) : Data clear (for deletion of previously stored data. This operation also clears statistics).

L (List) : Statistic display (for display of number of data items, sum, sum of squares, mean, population standard deviation, sample standard deviation). [⇩] (or [EXE]) scrolls to the following data item, [⇧] to the previous item, and [⇦] or [⇨] exits the statistic display and returns to the menu.

E (End) : Advances to the interval estimation display (same as when N is pressed in the first step above).

(2) N

Key	Display	Note
[N]	N (μ . σ²) a < σ < b n= 5 ?	(Interval estimation display)

The display appears as illustrated above when the [N] key is pressed. The value indicated for n shows the number of data currently stored in memory.

- n = 0 : Interval estimation cannot be performed, so this should be corrected to the required data.
- Number of data input (following [Y] above) and value of n differ : Confirm that some data have not been omitted during the input or that two or more data items have been input together for a single entry. In either case, terminate the library operation. Enter the library again and add, delete, or reinput data as required.
- Number of data input (following [Y] above) matches value of n : Press [EXE].

EXAMPLE

The table below shows the measured volume of the contents of five different randomly selected cans of a soft drink produced by the same manufacturer. Using this data, perform interval estimation with 99% confidence level for the sample standard deviation of the content volume.

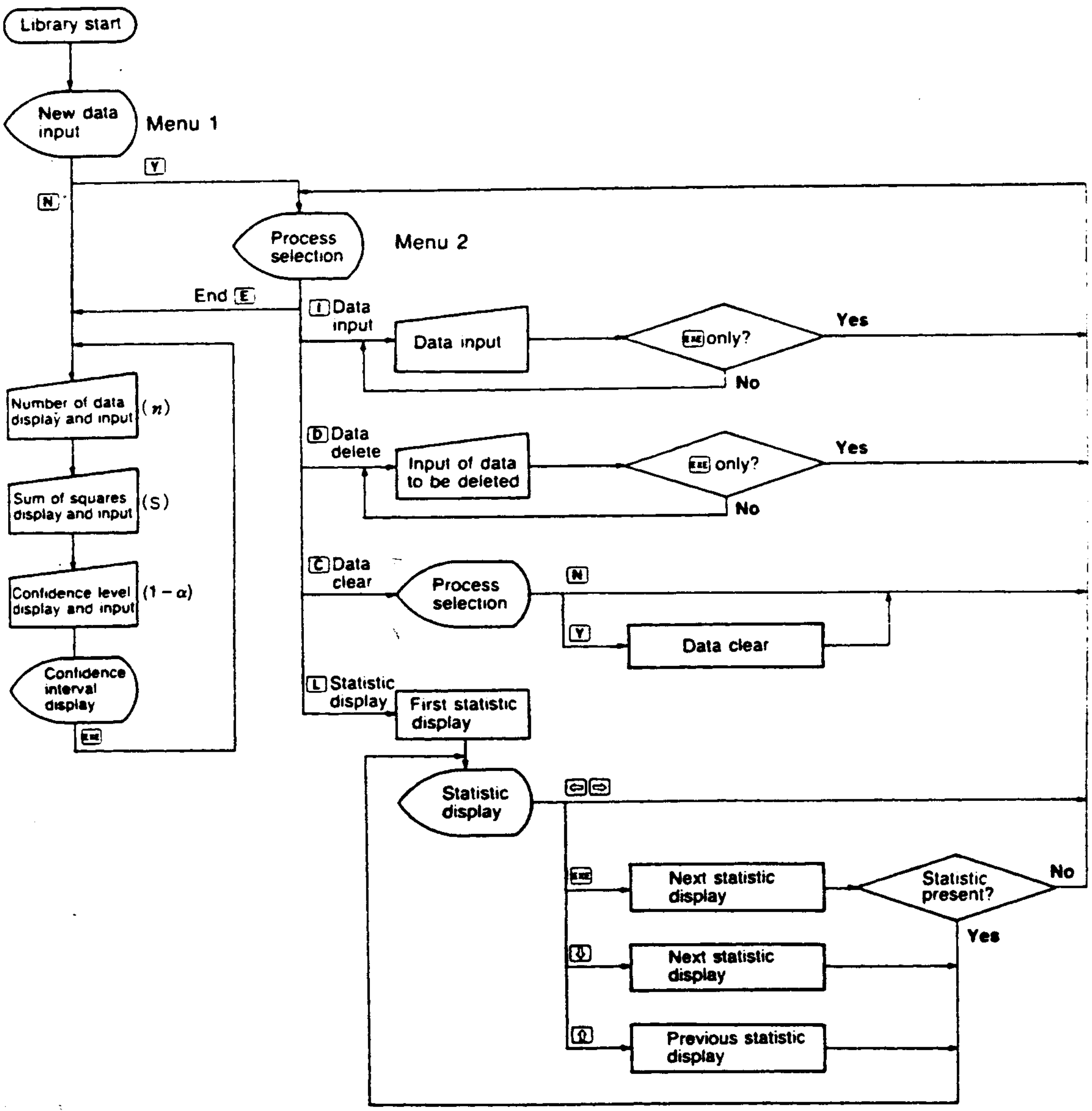
	1	2	3	4	5
VOLUME	1.20	1.08	1.15	1.22	1.17

Key	Display	Note
	N (μ . σ²) a < σ < b input new data (Y/N) ?	
[Y]	Input data (x) >Input.Delete.Clear.List.End ?_	(Select new data input.)
[C]	Input data (x) clear data (Y/N) ?	(Select data clear.)
[Y]	Input data (x) >Input.Delete.Clear.List.End ?_	(Data cleared.)
[I]	Input data (x) [EXE]:menu x?_	(Select data input.)
1.20 [EXE]	Input data (x) [EXE]:menu x?_	(Enter first data item.)
1.08 [EXE] 1.15 [EXE] 1.22 [EXE] 1.17 [EXE]	Input data (x) [EXE]:menu x?_	(Enter remaining data items.)
[EXE]	Input data (x) >Input.Delete.Clear.List.End ?_	(Return to menu.)
[E]	N (μ . σ²) a < σ < b n= 5 ?_	(Select End to proceed to interval estimation)
[EXE]	N (μ . σ²) a < σ < b S= 0.01172 ? _	(Press [EXE] after checking number of data items.)
[EXE]	Confidence level (1−α)(%) 1−α= 95 ?_	(Press [EXE] after checking sum of squares.)

Key operation	Display	Note
99 EXE	N (μ . σ2) 99 %	(Enter confidence level value to display confidence interval.)
	N (μ . σ2) 99 % 0.02808 < σ < 0.238	
EXE	N (μ . σ2) a < σ < b n= 5 ?_	

Here, it is determined that the sample standard deviation of the volume of the cans' contents σ with a confidence level of 99% is $0.02808<\sigma<0.238$.

STANDARD DEVIATION INTERVAL ESTIMATION FLOWCHART



6650 VARIANCE RATIO INTERVAL ESTIMATION

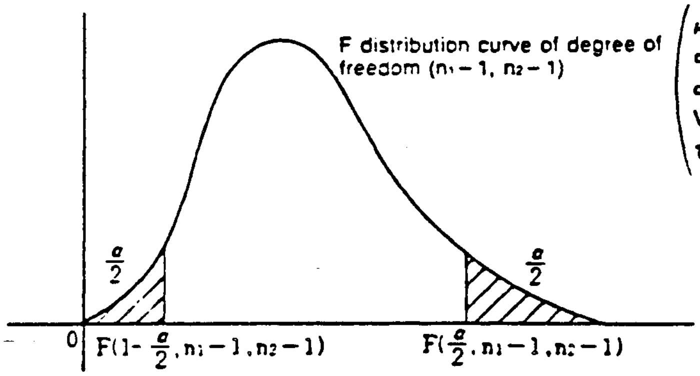
Performs estimation of the confidence interval of $\frac{\sigma_2^2}{\sigma_1^2}$ for the two normal distributions N (μ_1, σ_1^2) and N (μ_2, σ_2^2), where μ_1, σ_1^2, μ_2 and σ_2^2 are all unknown.

CALCULATIONS

When n_1-size sample x_1 $(x_{11}, x_{12} \cdots x_{1n_1})$ is taken from normal distribution N (μ_1, σ_1^2), and n_2-size sample x_2 $(x_{21}, x_{22} \cdots x_{2n_2})$ is taken from normal distribution N (μ_2, σ_2^2), the confidence interval of the confidence level $(1-\alpha)$ of $\frac{\sigma_2^2}{\sigma_1^2}$ is obtained by

$$\frac{V_2}{V_1} \cdot \frac{1}{F(\frac{\alpha}{2}, n_2-1, n_1-1)} < \frac{\sigma_2^2}{\sigma_1^2} < \frac{V_2}{V_1} \cdot F(\frac{\alpha}{2}, n_1-1, n_2-1)$$

in accordance with the F distribution of the degrees of freedom (n_1-1, n_2-1).



- μ_1, μ_2 : population means
- σ_1^2, σ_2^2 : population variances
- α : significance level
- V_1, V_2 : unbiased variances
- $1-\alpha$: confidence level

$$V = \frac{\Sigma(x-\bar{x})^2}{n-1}$$

OPERATION

6650 [LB]

```
N(μ1.σ1²).N(μ2.σ2²)  a<σ2²/σ1²<b
input new data x1  (Y/N) ?
```

The display appears as indicated above once the library is activated. At this time, either Y or N should be pressed to perform the following procedures:

Y : New data input followed by interval estimation, additional data input, data edit, statistic check.

N : Interval estimation using previously stored data, interval estimation by inputting each value.

(1) Y

[Y]

```
Input data (x1)
>Input.Delete.Clear.List.End ?_
```

The menu display illustrated above appears when [Y] is pressed. One of the following character keys is then pressed to perform the corresponding function.

npui) : Data input (for input or addition of data).
Delete) : Data delete (for deletion of erroneous or unnecessary data).
Clear) : Data clear (for deletion of previously stored data. This operation also clears statistics).
.ist) : Statistic display (for display of number of data items, sum, sum of squares, mean, population standard deviation, sample standard deviation).
[⇩] (or [EXE]) scrolls to the following data item, [⇧] to the previous item, and [⊖] or [⊖] exits the statistic display and returns to the menu.
End) : Advances to the interval estimation display (same as when N is pressed in the first step above).

ote that the data input referred to here is for data items x_{1_1} through x_{1n_1}.

N

```
N(μ1.σ1²).N(μ2.σ2²)  a<σ2²/σ1²<b
input new data x2 (Y/N) ?
```

display appears as illustrated above when the [N] key is pressed. Note that this display most identical to the initial display which appears immediately after entering library operns. The difference, however, is that the question concerning new data input here is for items x_{2_1} through x_{2n_2}, while the data input being queried on the original display is for items x_{1_1} through x_{1n_2}.

) **Y**

e result as that produced by pressing [Y] in step (1) above. Note, however, that the data g entered or corrected here is x_{2_1} through x_{2n_2}.

) **N**

```
N(μ1.σ1²).N(μ2.σ2²)  a<σ2²/σ1²<b
n1= 5 ?_
```

(Number of data display)

display appears as illustrated above when the [N] key is pressed. The value indicated i shows the number of x_1 (x_{1_1} – x_{1n_1}) data currently stored in memory.

=0 : Interval estimation cannot be performed, so this should be corrected to the quired data.
mber of data input (following [Y] above) and value of n differ : Confirm that some data ve not been omitted during the input or that two or more data items have been input gether for a single entry. In either case, terminate the library operation. Enter the library ain and add, delete, or reinput data as required.
mber of data input (following [Y] above) matches value of n : Press [EXE].
1en [EXE] is pressed, a display similar to that above is produced for x_2 (x_{2_1} – x_{2n_2}) data ms. After confirmation and/or corrections as described in (2-2), press [EXE] to continue.

MPLE

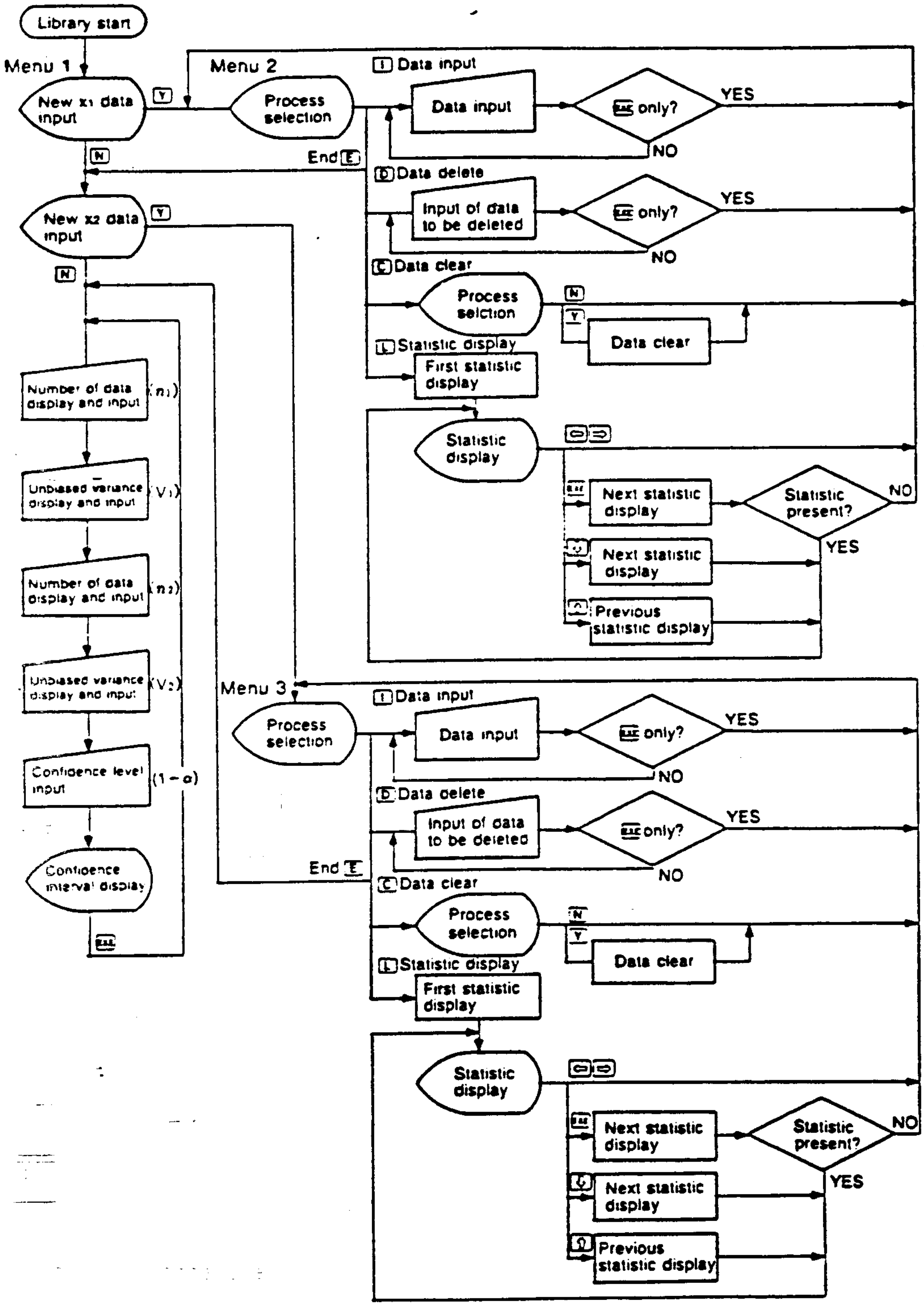
table below shows the measured diameters of ten randomly selected ball bearings. The ory producing the ball bearings uses two separate production lines (A and B), so five om samples were taken from each line. Using this data, perform interval estimation with confidence level for the variance ratio of the diameters.

		1	2	3	4	5
DIAMETER	A	1.01	1.00	1.01	1.02	1.00
	B	1.00	0.99	1.01	1.00	0.98

	$N(\mu 1.\sigma 1^2).N(\mu 2.\sigma 2^2)\ a<\sigma 2^2/\sigma 1^2<b$ Input new data x_1 (Y/N) ?	
[Y]	Input data (x_1) >Input.Delete.Clear.List.End ?_	(Select new x_1 data input.)
[C]	Input data (x_1) clear data (Y/N) ?	(Select data clear.)
[Y]	Input data (x_1) >Input.Delete.Clear.List.End ?_	(Data cleared.)
[I]	Input data (x_1) [EXE]:menu x_1?_	(Select data input.)
1.01 [EXE]	Input data (x_1) [EXE]:menu x_1?_	(Enter first data item for line A.)
1.00 [EXE] 1.01 [EXE] 1.02 [EXE] 1.00 [EXE]	Input data (x_1) [EXE]:menu x_1?_	(Enter remaining data items.)
[EXE]	Input data (x_1) >Input.Delete.Clear.List.End ?_	(Return to menu.)
[E]	$N(\mu 1.\sigma 1^2).N(\mu 2.\sigma 2^2)\ a<\sigma 2^2/\sigma 1^2<b$ input new data x_2 (Y/N) ?	(Select End to clear x_1 data menu.)
[Y]	Input data (x_2) >Input.Delete.Clear.List.End ?_	(Select new x_2 data input.)
[C]	Input data (x_2) clear data (Y/N) ?	(Select data clear.)
[Y]	Input data (x_2) >Input.Delete.Clear.List.End ?_	(Data cleared.)
[I]	Input data (x_2) [EXE]:menu x_2?_	(Select data input.)
1.00 [EXE]	Input data (x_2) [EXE]:menu x_2?_	(Enter first data item for line B.)
0.99 [EXE] 1.01 [EXE] 1.00 [EXE] 0.98 [EXE]	Input data (x_2) [EXE]:menu x_2?_	(Enter remaining data items.)
[EXE]	Input data (x_2) >Input.Delete.Clear.List.End ?_	(Return to menu.)
[E]	$N(\mu 1.\sigma 1^2).N(\mu 2.\sigma 2^2)\ a<\sigma 2^2/\sigma 1^2<b$ n_1= 5 ?_	(Select End to clear x_2 data menu.)
[EXE]	$N(\mu 1.\sigma 1^2).N(\mu 2.\sigma 2^2)\ a<\sigma 2^2/\sigma 1^2<b$ V_1= 0.00007 ?_	(Press [EXE] after checking number of data items n_1.)
[EXE]	$N(\mu 1.\sigma 1^2).N(\mu 2.\sigma 2^2)\ a<\sigma 2^2/\sigma 1^2<b$ n_2= 5 ?_	(Press [EXE] after checking unbiased variance V_1.)
[EXE]	$N(\mu 1.\sigma 1^2).N(\mu 2.\sigma 2^2)\ a<\sigma 2^2/\sigma 1^2<b$ V_2= 0.00013 ?_	(Press [EXE] after checking number of data items n_2.)
[EXE]	Confidence level (1−α)[%] 1−α= 95 ?_	(Press [EXE] after checking unbiased variance V_2.)
[EXE]	$N(\mu 1.\sigma 1^2).N(\mu 2.\sigma 2^2)$ 95 %	(Enter confidence level. Since 95% is already set, press [EXE].)
	$N(\mu 1.\sigma 1^2).N(\mu 2.\sigma 2^2)$ 95 % 0.1934 < $\sigma 2^2/\sigma 1^2$ < 17.84	
[EXE]	$N(\mu 1.\sigma 1^2).N(\mu 2.\sigma 2^2)\ a<\sigma 2^2/\sigma 1^2<b$ n_1= 5 ?_	

Here, it is determined that the variance ratio between the two lines with a confidence level of 95% is $0.1934<\frac{\sigma_2^2}{\sigma_1^2}<17.84$.

IANCE RATIO INTERVAL ESTIMATION FLOWCHART



6660 MEAN DIFFERENCE INTERVAL ESTIMATION

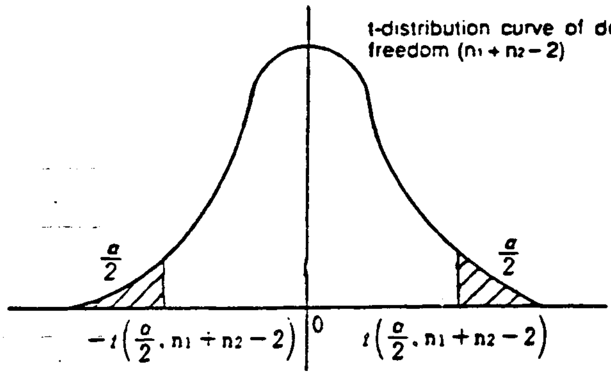
Performs estimation of the confidence interval $\mu_1-\mu_2$ for two equal distributions N (μ_1, σ^2) and N (μ_2, σ^2), where μ_1, μ_2 and σ^2 are all unknown.

CALCULATIONS

When n_1-size sample x_1 $(x_{11}, x_{12} \cdots x_{1n_1})$ is taken from normal distribution N (μ_1, σ^2), and n_2-size sample x_2 $(x_{21}, x_{22} \cdots x_{2n_2})$ is taken from normal distribution N (μ_2, σ^2), the confidence interval of the confidence level $(1-\alpha)$ of $\mu_1-\mu_2$ is obtained by

$$\bar{x}_1-\bar{x}_2-t\left(\frac{\alpha}{2}, n_1+n_2-2\right)\sqrt{\left(\frac{1}{n_1}+\frac{1}{n_2}\right)\left(\frac{S_1+S_2}{n_1+n_2-2}\right)} < \mu_1-\mu_2 < \bar{x}_1-\bar{x}_2+t\left(\frac{\alpha}{2}, n_1+n_2-2\right)\cdot\sqrt{\left(\frac{1}{n_1}+\frac{1}{n_2}\right)\left(\frac{S_1+S_2}{n_1+n_2-2}\right)}$$

in accordance with t-distribution of the degree of freedom (n_1+n_2-2).



- μ_1, μ_2 : population means
- σ^2 : population variance (same for both populations)
- α : significance level
- $1-\alpha$: confidence level
- $\bar{x}_1, \bar{x}_2$: sample means
- S_1, S_2 : sums of squares

$S=\Sigma(x-\bar{x})^2$

OPERATION

6660 [LIB]

```
N(μ1.σ2).N(μ2.σ2)  a<μ1-μ2<b
input new data x1  (Y/N) ?
```

The display appears as indicated above once the library is activated. At this time, either Y or N should be pressed to perform the following procedures:

Y : New data input followed by interval estimation, additional data input, data edit, statistic check.

N : Interval estimation of previously stored data, interval estimation by inputting each value.

(1) Y

[Y]

```
Input data (x1)
>Input.Delete.Clear.List.End ?_
```

The menu display illustrated above appears when [Y] is pressed. One of the following character keys is then pressed to perform the corresponding function.

I (Input) : Data input (for input or addition of data).
D (Delete) : Data delete (for deletion of erroneous or unnecessary data).
C (Clear) : Data clear (for deletion of previously stored data. This operation also clears statistics).
L (List) : Statistic display (for display of number of data items, sum, sum of squares, mean, population standard deviation, sample standard deviation).
[↓] (or [EXE]) scrolls to the following data item, [↑] to the previous item, and [⇦] or [⇨] exits the statistic display and returns to the menu.
E (End) : Advances to the interval estimation display (same as when N is pressed in the first step above).

* Note that the data input referred to here is for data items x_{1_1} through x_{1n_1}.

(2) N

[N]

```
N(μ1.σ2).N(μ2.σ2)  a<μ1−μ2<b
Input new data x2 (Y/N) ?
```

The display appears as illustrated above when the [N] key is pressed. Note that this display is almost identical to the initial display which appears immediately after entering library operations. The difference, however, is that the question concerning new data input here is for data items x_{2_1} through x_{2n_2}, while the data input being queried on the original display is for data items x_{1_1} through x_{1n_1}.

(2-1) Y

Same result as that produced by pressing [Y] in step (1) above. Note, however, that the data being entered or corrected here is x_{2_1} through x_{2n_2}.

(2-2) N

[N]

```
N(μ1.σ2).N(μ2.σ2)  a<μ1−μ2<b
n1= 5 ?_
```

(Number of data display)

The display appears as illustrated above when the [N] key is pressed. The value indicated for n shows the number of x_1 ($x_{1_1} - x_{1n_1}$) data currently stored in memory.

- $n_1 = 0$: Interval estimation cannot be performed, so this should be corrected to the required data.
- Number of data input (following [Y] above) and value of n differ : Confirm that some data have not been omitted during the input or that two or more data items have been input together for a single entry. In either case, terminate the library operation. Enter the library again and add, delete, or reinput data as required.
- Number of data input (following [Y] above) matches value of n : Press [EXE].
When [EXE] is pressed, a display similar to that above is produced for x_2 ($x_{2_1} - x_{2n_2}$) data items. After confirmation and/or corrections as described in (2–2), press [EXE] to continue.

EXAMPLE

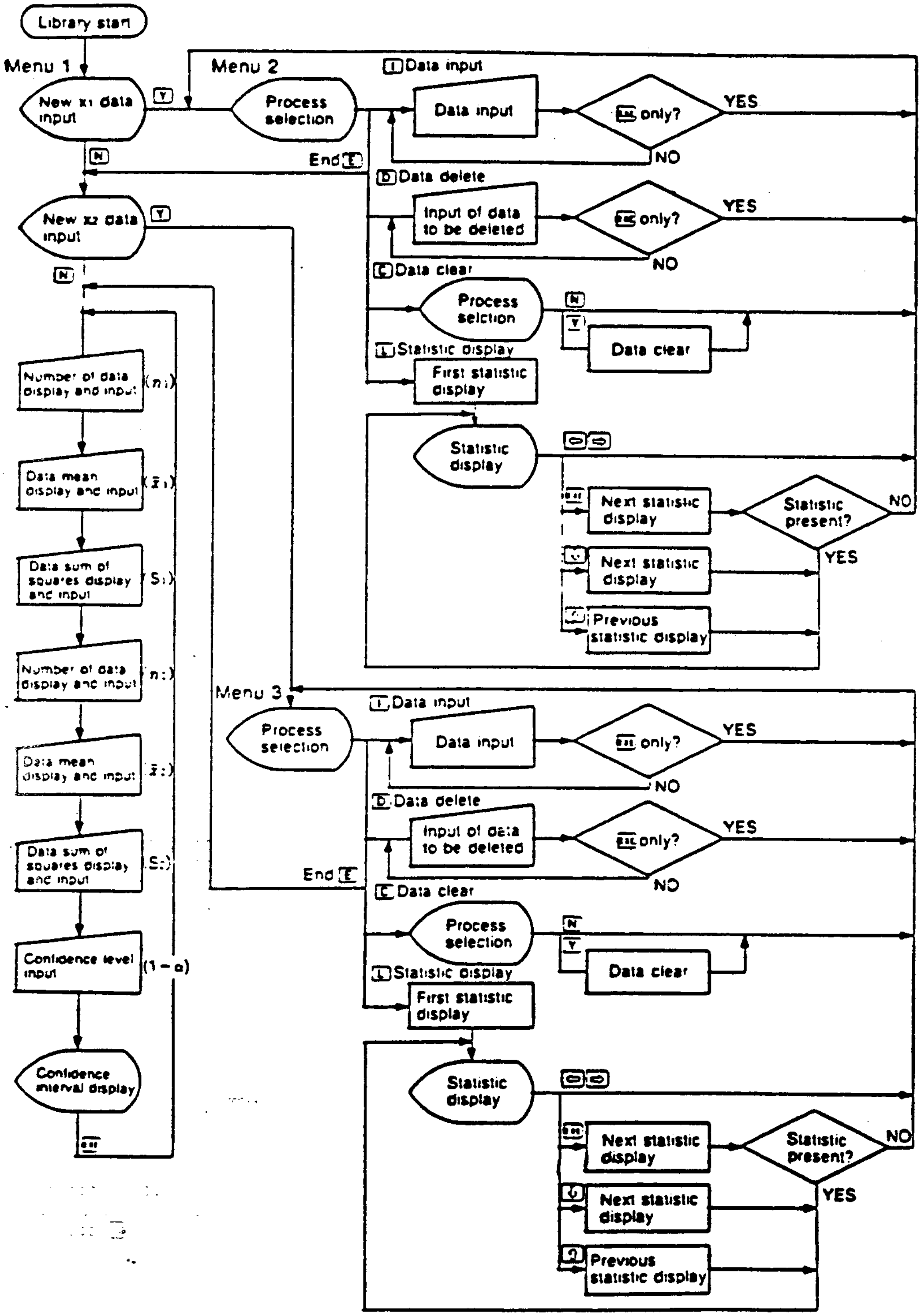
The table below shows a comparison of the production volume for a factory for two consecutive weeks. Using this data, perform interval estimation with 95% confidence level for the difference in the mean for the two weeks.

		MON	TUE	WED	THU	FRI
PRODUCTION VOLUME (t)	WEEK 1	53	59	56	60	54
	WEEK 2	55	62	60	61	58

	N(μ1.σ2).N(μ2.σ2) a<μ1−μ2<b input new data x1 (Y/N) ?	
[Y]	Input data (x1) >Input.Delete.Clear.List.End ?_	(Select new x_1 data input.)
[C]	Input data (x1) clear data (Y/N)?	(Select data clear.)
[Y]	Input data (x1) >Input.Delete.Clear.List.End ?_	(Data cleared.)
[I]	Input data (x1) [EXE]:menu x1?_	(Select data input.)
53 [EXE]	Input data (x1) [EXE]:menu x1?_	(Enter first data item for week 1.)
59 [EXE] 56 [EXE] 60 [EXE] 54 [EXE]		
	Input data (x1) [EXE]:menu x1?_	(Enter remaining data items.)
[EXE]	Input data (x1) >Input.Delete.Clear.List.End ?_	(Return to menu.)
[E]	N(μ1.σ2).N(μ2.σ2) a<μ1−μ2<b input new data x2 (Y/N) ?	(Select End to clear x_1 data menu.)
[Y]	Input data (x2) >Input.Delete.Clear.List.End ?_	(Select new x_2 data input.)
[C]	Input data (x2) clear data (Y/N)?	(Select data clear.)
[Y]	Input data (x2) >Input.Delete.Clear.List.End ?_	(Data cleared.)
[I]	Input data (x2) [EXE] menu x2?_	(Select data input.)
55 [EXE]	Input data (x2) x2?_	(Enter first data item for week 2.)
62 [EXE] 60 [EXE] 61 [EXE] 58 [EXE]		
	Input data (x2) x2?_	(Enter remaining data items.)
[EXE]	Input data (x2) >Input.Delete.Clear.List.End ?_	(Return to menu.)
[E]	N(μ1.σ2).N(μ2.σ2) a<μ1−μ2<b n1= 5 ?_	(Select End to clear x_2 data menu.)
[EXE]	N(μ1.σ2).N(μ2.σ2) a<μ1−μ2<b x̄1= 55.4 ?_	(Press [EXE] after checking number of data items n_1.)
[EXE]	N(μ1.σ2).N(μ2.σ2) a<μ1−μ2<b S1= 37.2 ?_	(Press [EXE] after checking data mean $\bar{x}_1$.)
[EXE]	N(μ1.σ2).N(μ2.σ2) a<μ1−μ2<b n2= 5 ?_	(Press [EXE] after checking sum of squares S_1.)
[EXE]	N(μ1.σ2).N(μ2.σ2) a<μ1−μ2<b x̄2= 59.2 ?_	(Press [EXE] after checking number of data items n_2.)
[EXE]	N(μ1.σ2).N(μ2.σ2) a<μ1−μ2<b S2= 30.8 ?_	(Press [EXE] after checking data mean $\bar{x}_2$.)
[EXE]	Confidence level (1−α)[%] 1−α= 95 ?_	(Press [EXE] after checking sum of squares S_2.)
[EXE]	N(μ1.σ2).N(μ2.σ2) 95	(Enter confidence level. Since 95% is already set, press [EXE].)
	N(μ1.σ2).N(μ2.σ2) 95 % -7.052 < μ1−μ2 < 1.452	
[EXE]	N(μ1.σ2).N(μ2.σ2) a<μ1−μ2<b n1= 5 ?_	

Here, it is determined that the difference in means $\mu_1-\mu_2$ between the two weeks with a confidence level of 95% is $-7.052<\mu_1-\mu_2<1.452$

AN DIFFERENCE INTERVAL ESTIMATION FLOWCHART



6670 RATIO INTERVAL ESTIMATION

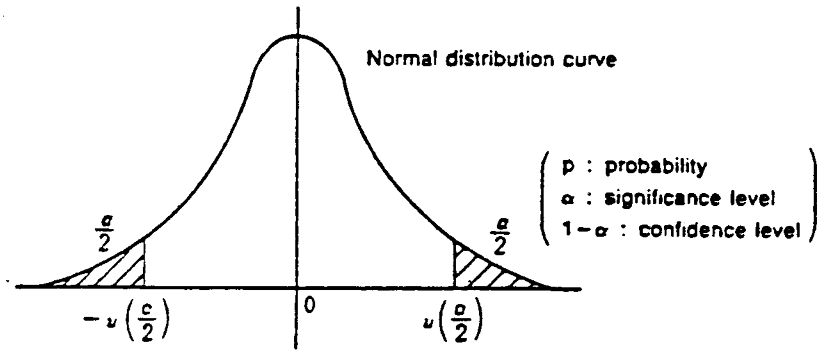
Performs estimation of the confidence interval p for binomial distribution B(1, p).

CALCULATIONS

When n-size sample ($x_1, x_2 \cdots x_n$) is taken from binomial distribution B (1, p), the confidence interval of the confidence level ($1-\alpha$) of p is obtained by

$$\frac{\Sigma x}{n} - u\left(\frac{\alpha}{2}\right)\sqrt{\frac{1}{n}\left(\frac{\Sigma x}{n}\left(1-\frac{\Sigma x}{n}\right)\right)} < p < \frac{\Sigma x}{n} + u\left(\frac{\alpha}{2}\right)\sqrt{\frac{1}{n}\left(\frac{\Sigma x}{n}\left(1-\frac{\Sigma x}{n}\right)\right)}$$

in accordance with an approximation of the standard normal distribution N $(0, 1^2)$.



OPERATION

6670 [LB]

```
B(1.p)  a<p<b
n=  0  ?_
```

EXAMPLE

12 defects are found for 1000 bolts produced by a certain factory. Using this data, perform interval estimation with 99% confidence level for the defect rate of the bolts.

```
B(1.p)  a<p<b
n=  0  ?_
```

1000 [EXE] (Enter total number of samples.)

```
B(1.p)  a<p<b
Σx=  0  ?_
```

12 [EXE] (Enter number of defects.)

```
Confidence level (1-α)(%)
1-α=  95  ?_
```

99 [EXE] (Enter confidence level.)

```
B(1.p)       99 %
 0.003131  <  p  <  0.02087
```

[EXE]

```
B(1.p)  a<p<b
n=  1000  ?_
```

Here, it is determined that the defect rate p for the bolts with a confidence level of 99% is $0.003131 < p < 0.02087$.

6680 RATIO DIFFERENCE INTERVAL ESTIMATION

-forms estimation of the confidence interval p_1-p_2 for two binomial distributions $B(1, p_1)$ d $B(1, p_2)$.

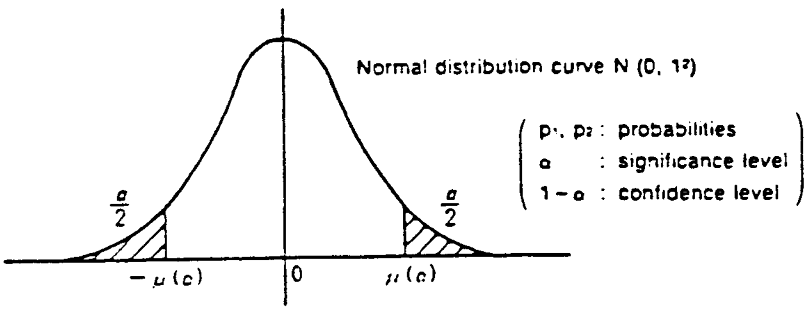
.LCULATIONS

ıen n_1-size sample x_1 $(x_{11}, x_{12} \cdots x_{1n_1})$ is taken from binomial distribution $B(1, p_1)$, and n_2-e sample x_2 $(x_{21}, x_{22} \cdots x_{2n_2})$ is taken from binomial distribution $B(1, p_2)$, the confidence erval of the confidence level $(1-\alpha)$ of p_1-p_2 is obtained by

$$\left(\frac{\Sigma x_1}{n_1}-\frac{\Sigma x_2}{n_2}\right) u\left(\frac{\alpha}{2}\right)\sqrt{\frac{1}{n_1}\left(\frac{\Sigma x_1}{n_1}\left(1-\frac{\Sigma x_1}{n_1}\right)\right)+\frac{1}{n_2}\left(\frac{\Sigma x_2}{n_2}\left(1-\frac{\Sigma x_2}{n_2}\right)\right)} < p_1-p_2 < \left(\frac{\Sigma x_1}{n_1}-\frac{\Sigma x_2}{n_2}\right)$$

$$u\left(\frac{\alpha}{2}\right)\sqrt{\frac{1}{n}\left(\frac{\Sigma x_1}{n_1}\left(1-\frac{\Sigma x_1}{n_1}\right)\right)+\frac{1}{n_2}\left(\frac{\Sigma x_2}{n_2}\left(1-\frac{\Sigma x_2}{n_2}\right)\right)}$$

accordance with an approximation of the standard normal distribution $N(0, 1^2)$.



ERATION

```
580 SUB    B(1.p1).B(1.p2)  a<p1-p2<b
           n1=  0  ?_
```

:AMPLE

e table below shows a comparison of the number of defects for a factory for two consecu- months. Using this data, perform interval estimation with 95% confidence level for the erence in the rates of defect.

	Finished products	Number of defects
MONTH 1	1500	23
MONTH 2	1200	15

Key operation	Display	Note
	B(1.p_1).B(1.p_2) $a<p_1-p_2<b$ n_1= 0 ?_	
1500 EXE	B(1.p_1).B(1.p_2) $a<p_1-p_2<b$ Σx_1= 0 ?_	(Input finished products for MONTH 1.)
23 EXE	B(1.p_1).B(1.p_2) $a<p_1-p_2<b$ n_2= 0 ?_	(Input number of defects for MONTH 1.)
1200 EXE	B(1.p_1).B(1.p_2) $a<p_1-p_2<b$ Σx_2= 0 ?_	(Input finished products for MONTH 2.)
15 EXE	Confidence level (1−α)[%] 1−α= 95 ?_	(Input number of defects for MONTH 2.)
EXE	B(1.p_1).B(1.p_2) 95 % $-0.006009 < p_1-p_2 < 0.01168$	(Enter confidence level. Since 95% is already set, press EXE.)
EXE	B(1.p_1).B(1.p_2) $a<p_1-p_2<b$ n_1= 1500 ?_	

Here, it is determined that the difference in probabilities p_1-p_2 between the two months with a confidence level of 95% is $-0.006009<p_1-p_2<0.01168$.

6710 POPULATION MEAN TEST (TWO-SIDED) : FOR KNOWN VARIANCE

forms hypothesis testing of μ in normal distribution N (μ, σ^2 ; where μ : unknown, σ^2 : wn).

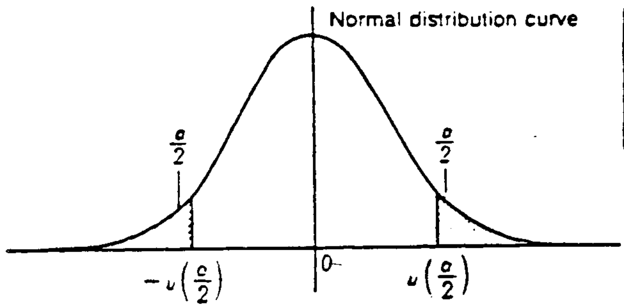
LCULATIONS

n-size sample ($x_1, x_2 \cdots x_n$) is taken from normal distribution N (μ, σ^2). At this time, criti- regions are established on both sides of the normal distribution as shown in the illustra- when:

othesis to be tested (Null hypothesis) $H_0 : \mu = \mu_0$
rnative hypothesis $H_1 : \mu \neq \mu_0$

test is performed using

$$\left| \frac{\bar{x} - \mu_0}{\frac{\sigma}{\sqrt{n}}} \right| > u\left(\frac{\alpha}{2}\right)$$



μ_0 : population mean
σ^2 : population variance
σ : population standard deviation
$\bar{x}$: sample mean
α : significance level

ERATION

10 LIB

```
Test        H0:μ=μ0   H1:μ≠μ0
input new data (Y/N) ?_
```

display appears as illustrated above once the library is activated. At this time, either r N should be pressed to perform the following procedures:

New data input followed by test, additional data input, data edit, statistic check.
Test of previously stored data, test by inputting each value.

Y

```
Input data (x)
>Input.Delete.Clear.List.End ?_
```

menu display illustrated above appears when [Y] is pressed. One of the following character 's is then pressed to perform the corresponding function.

I (Input) : Data input (for input or addition of data).
D (Delete) : Data delete (for deletion of erroneous or unnecessary data).
C (Clear) : Data clear (for deletion of previously stored data. This operation also clears statistics operations).
L (List) : Statistic display (for display of number of data items, sum, sum of squares, mean, population standard deviation, sample standard deviation).
[⇩] (or [EXE]) scrolls to the following data item, [⇧] to the previous item, and [⊖] or [⊖] exits the statistic display and returns to the menu.
E (End) : Advances to the test display (same as when N is pressed in the first step above).

(2) N

[N]	Test H0:μ=μ0 H1:μ≠μ0 μ0= 0 ?_	(Test display)

The display appears as illustrated above when the [N] key is pressed. From this point, various parameters are entered for the test.

EXAMPLE

The table below shows the measured speed of five new football players over 100 meters. These times will be used to determines whether or not these players meet the team standards. Perform a test on the data with a significance level of 5%. The mean time for the entire team is 11.4 seconds, with a standard deviation of 1.30.

	1	2	3	4	5
TIME	12.3	11.6	10.9	12.8	11.4

	Test H0:μ=μ0 H1:μ≠μ0 input new data (Y/N) ?_	
[Y]	Input data (x) >Input.Delete.Clear.List.End ?_	(Select new data input.)
[C]	Input data (x) clear data (Y/N) ?	(Select data clear.)
[Y]	Input data (x) >Input.Delete.Clear.List.End ?_	(Data cleared.)
[I]	Input data (x) [EXE]:menu x?_	(Select data input.)
12.3 [EXE]	Input data (x) [EXE]:menu x?_	(Enter first data item.)
11.6 [EXE] 10.9 [EXE] 12.8 [EXE] 11.4 [EXE]	Input data (x) [EXE]:menu x?_	(Enter remaining data items.)
[EXE]	Input data (x) >Input.Delete.Clear.List.End ?_	(Return to menu.)
[E]	Test H0:μ=μ0 H1:μ≠μ0 μ0= 0 ?_	(Select End to proceed to test.)
11.4 [EXE]	Test H0:μ=μ0 H1:μ≠μ0 σ= 0 ?_	(Enter mean.)
1.30 [EXE]	Test H0:μ=μ0 H1:μ≠μ0 n= 5 ?_	(Enter population standard deviation.)
[EXE]	Test H0:μ=μ0 H1:μ≠μ0 x̄= 11.8 ?_	(Press [EXE] after checking number of data.)
[EXE]	Significance level α(%) α= 5 ?_	(Press [EXE] after checking data mean.)

```
Test        H0:μ=μ0   H1:μ≠μ0
 . . . . .
```

(Enter significance level. 5% is already set, so simply press EXE.)

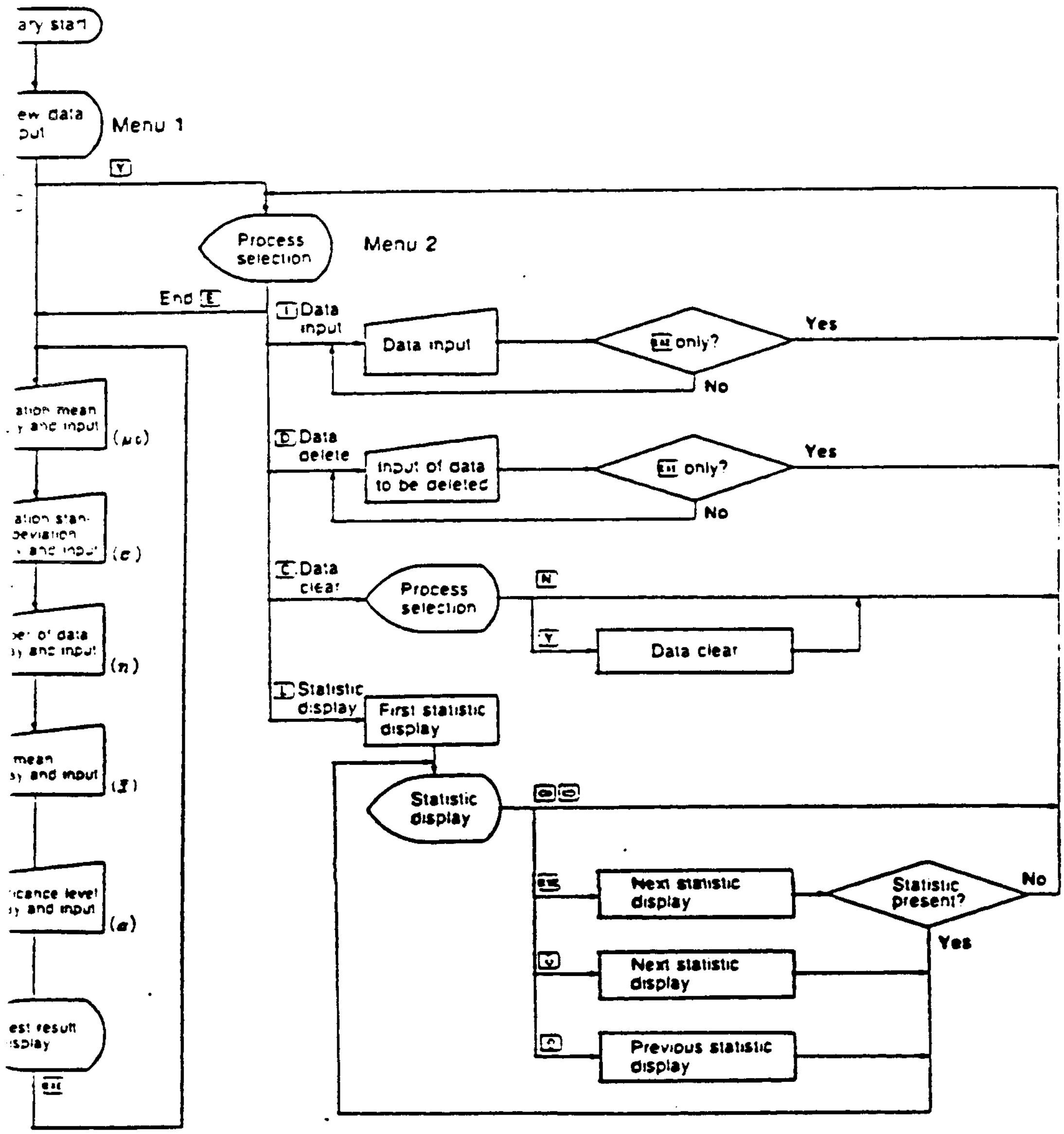
```
Test        H0:μ=μ0   H1:μ≠μ0
 0.688 ≦ 1.96 : Accept
```

(Display test result.)

```
Test        H0:μ=μ0   H1:μ≠μ0
μ0= 11.4 ?_
```

e, it is determined that the speeds of the new players meet the team standards. In this mple, the number of data items was limited to five for ease of understanding. In actual s, a small number of data may cause erroneous results (standard: $n \geqq 50$).

PULATION MEAN TEST FLOWCHART (TWO-SIDED FOR KNOWN VARIANCE)



6711 POPULATION MEAN TEST (RIGHT SIDED) : FOR KNOWN VARIANCE

Performs hypothesis testing of μ in normal distribution N (μ, σ^2 ; where μ : unknown, σ^2 : known).

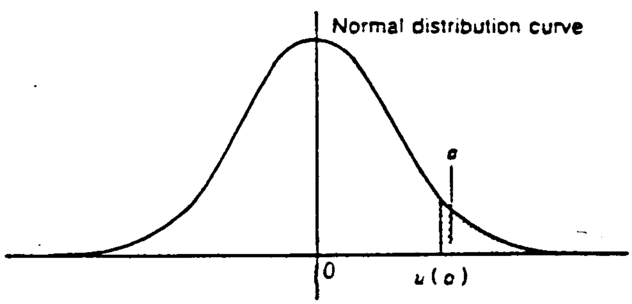
CALCULATIONS

An n-size sample (x_1, $x_2 \cdots x_n$) is taken from normal distribution N (μ, σ^2). At this time, the critical region is established on the right side of the normal distribution as shown in the illustration when:

Hypothesis to be tested (Null hypothesis) $H_0 : \mu = \mu_0$
Alternative hypothesis $H_1 : \mu > \mu_0$

The test is performed using

$$\frac{\bar{x} - \mu_0}{\frac{\sigma}{\sqrt{n}}} > u(\alpha)$$



(μ_0 : population mean
σ^2 : population variance
σ : population standard deviation
$\bar{x}$: sample mean
α : significance level)

OPERATION

6711 [LIB]

```
Test        Ho:μ=μo  H1:μ>μo
input new data (Y/N) ?_
```

The display appears as illustrated above once the library is activated. At this time, either Y or N should be pressed to perform the following procedures:

Y : New data input followed by test, additional data input, data edit, statistic check.
N : Test of previously stored data, test by inputting each value.

(1) Y

[Y]

```
Input data (x)
>Input.Delete.Clear.List.End ?_
```

The menu display illustrated above appears when [Y] is pressed. One of the following character keys is then pressed to perform the corresponding function.

(Input) : Data input (for input or addition of data).
(Delete) : Data delete (for deletion of erroneous or unnecessary data).
(Clear) : Data clear (for deletion of previously stored data. This operation also clears statistics).
(List) : Statistic display (for display of number of data items, sum, sum of squares, mean, population standard deviation, sample standard deviation).
[↓] (or [EXE]) scrolls to the following data item, [↑] to the previous item, and [⊟] or [⊟] exits the statistic display and returns to the menu.
(End) : Advances to the test display (same as when N is pressed in the first step above).

N

```
Test         H0:μ=μ0   H1:μ>μ0
μ0= 0 ?_
```
(Test display)

display appears as illustrated above when the [N] key is pressed. From this point, various istical values are entered for the test.

AMPLE

actory is considering replacing 50 obsolete machines with newer models. Management ms, however, that the capacity of the new machines are the same as those currently in . The data included in the table below are the results of tests performed on five units e new machines. Using these results, determine whether or not the capacity of the new chines is equal to the existing machines by performing a test on the data with a significance el of 5%. The capacity of the exiting machines is 432 units/hour, with a standard deviaof 15.

	1	2	3	4	5
UNITS/HOUR	475	501	483	492	487

```
Test         H0:μ=μ0   H1:μ>μ0
input new data (Y/N) ?_
```

```
Input data (x)
>Input.Delete.Clear.List.End ?_
```
(Select new data input.)

```
Input data (x)
clear data  (Y/N) ?
```
(Select data clear.)

```
Input data (x)
>Input.Delete.Clear.List.End ?_
```
(Data cleared.)

```
Input data (x)               [EXE]:menu
x?_
```
(Select data input.)

[EXE]

```
Input data (x)               [EXE]:menu
x?_
```
(Enter first data item.)

[EXE] 483 [EXE] 492 [EXE] 487 [EXE]

```
Input data (x)               [EXE]:menu
x?
```
(Enter remaining data items.)

```
Input data (x)
>Input.Delete.Clear.List.End ?_
```
(Return to menu.)

```
Test         H0:μ=μ0   H1:μ>μ0
μ0= 0 ?_
```
(Select End.)

[EXE]

```
Test         H0:μ=μ0   H1:μ>μ0
σ= 0 ?_
```
(Enter mean.)

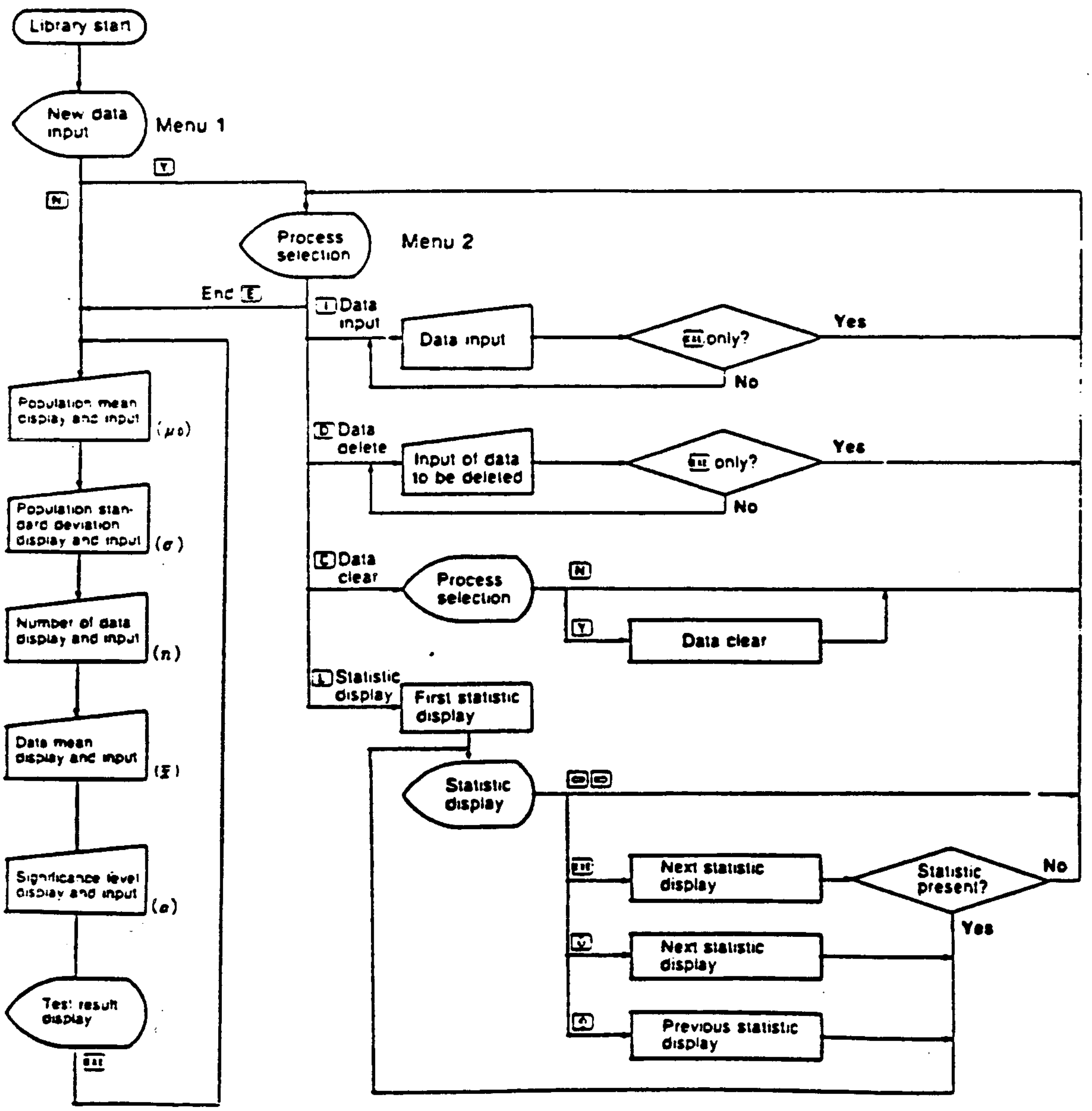
[EXE]

```
Test         H0:μ=μ0   H1:μ>μ0
n= 5 ?_
```
(Enter population standard deviation.)

EXE	Test Ho:μ=μo H1:μ>μo $\bar{x}$= 487.6 ?_	(Press EXE after checking number of data.)
EXE	Significance level α(%) α= 5 ?_	(Press EXE after checking data mean.)
EXE	Test Ho:μ=μo H1:μ>μo 8.288 > 1.645 : ReJect	(Enter significance level. 5% is already set, so simply press EXE.)
EXE	Test Ho:μ=μo H1:μ>μo μo= 432 ?_	

Here, it is determined that it cannot be said that the capacity of the new machines are identical to that of the existing machines. The new machines have higher capacities. In this example, the number of data items was limited to five for ease of understanding. In actual tests, smaller number of data may cause erroneous results (standard: $n \geqq 50$).

POPULATION MEAN TEST FLOWCHART (RIGHT SIDED FOR KNOWN VARIANCE)



5712 POPULATION MEAN TEST (LEFT SIDED) : FOR KNOWN VARIANCE

rforms hypothesis testing of μ in normal distribution N (μ, σ^2 ; where μ : unknown, σ^2 : own).

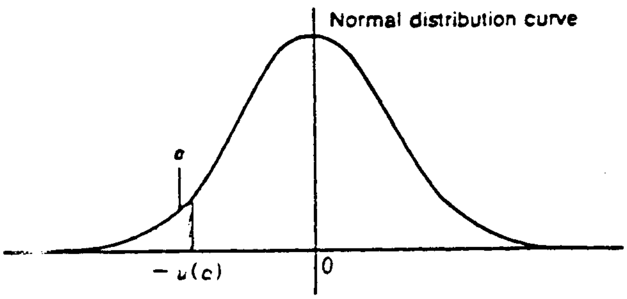
.LCULATIONS

n-size sample (x_1, $x_2 \cdots x_n$) is taken from normal distribution N (μ, σ^2). At this time, the tical region is established on the left side of the normal distribution as shown in the illus- tion when:

pothesis to be tested	(Null hypothesis)	$H_0 : \mu = \mu_0$
ernative hypothesis		$H_1 : \mu < \mu_0$

e test is performed using

$$\frac{\bar{x}-\mu_0}{\frac{\sigma}{\sqrt{n}}} < -u(\alpha)$$



- μ_0 : population mean
- σ^2 : population variance
- σ : population standard deviation
- $\bar{x}$: sample mean
- α : significance level

ERATION

712 [UB]

```
Test       H0:μ=μ0   H1:μ<μ0
input  new  data  (Y/N)  ?_
```

e display appears as illustrated above once the library is activated. At this time, either or N should be pressed to perform the following procedures:

: New data input followed by test, additional data input, data edit, statistic check.
: Test of previously stored data, test by inputting each value.

) Y

```
Input  data  (x)
>Input.Delete.Clear.List.End ?_
```

ie menu display illustrated above appears when [Y] is pressed. One of the following character ys is then pressed to perform the corresponding function.

I (Input) : Data input (for input or addition of data).
D (Delete) : Data delete (for deletion of erroneous or unnecessary data).
C (Clear) : Data clear (for deletion of previously stored data. This operation also clears statistics).
L (List) : Statistic display (for display of number of data items, sum, sum of squares, mean, population standard deviation, sample standard deviation).
[⇩] (or [EXE]) scrolls to the following data item, [⇧] to the previous item, and [⊖] or [⊖] exits the statistic display and returns to the menu.
E (End) : Advances to the test display (same as when N is pressed in the first step above).

(2) N

Key	Display	Note
[N]	Test H0: μ=μ0 H1: μ<μ0 μ0= 0 ?_	(Test display)

The display appears as illustrated above when the [N] key is pressed. From this point, various statistical values are entered for the test.

EXAMPLE

A company is considering replacing 500 lights has been approached by a salesman who claims to have lights which are less expensive, but with a comparable service life. The data included in the table below are the results of tests performed on five units of the new lights. Using these results, determine whether or not the capacity of these lights is equal to the existing lights by performing a test on the data with a significance level of 5%. The mean service life of the existing lights is 1,234 hours/light, with a standard deviation of 7.6.

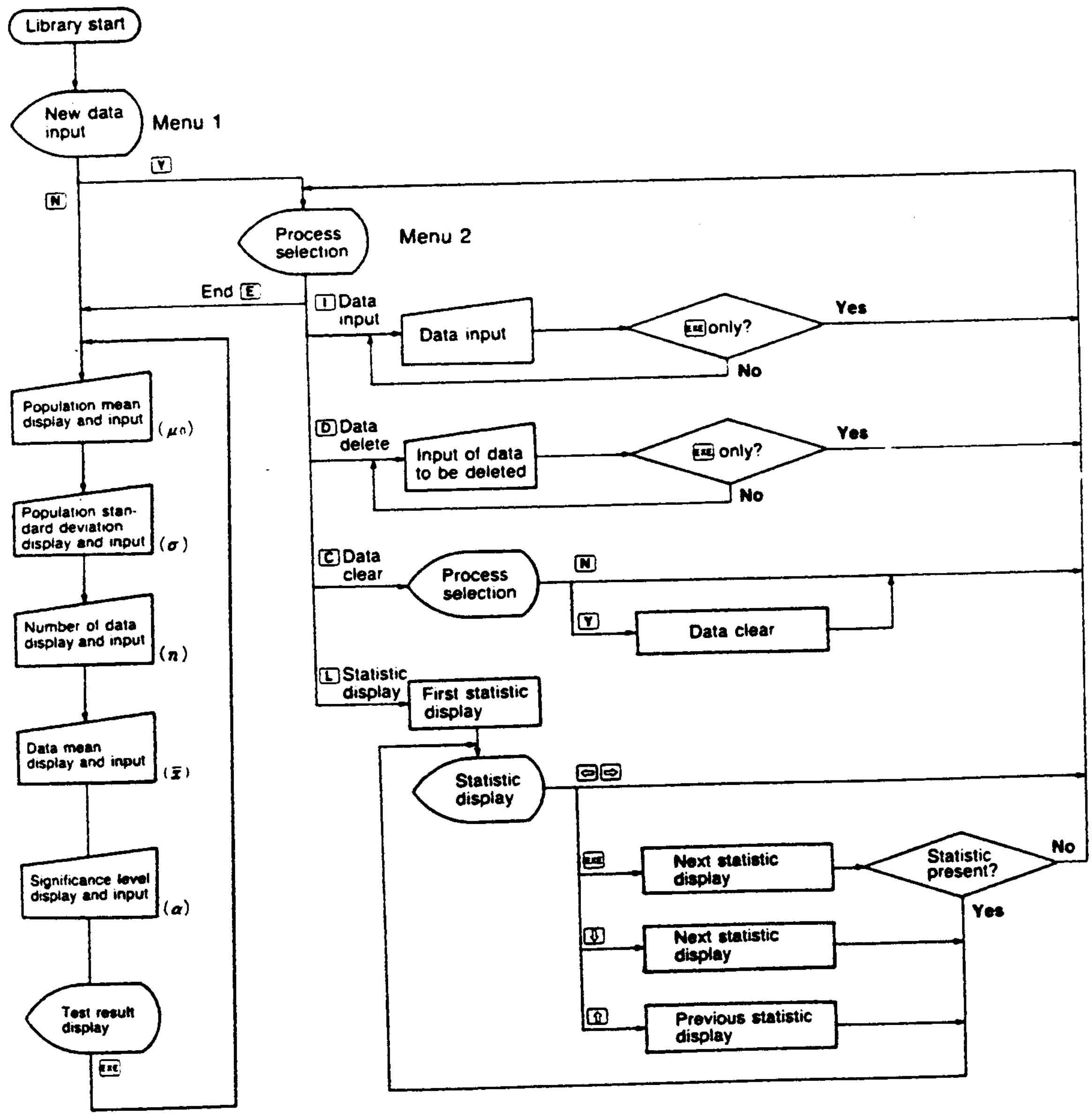
	1	2	3	4	5
SERVICE LIFE (HOURS)	1229	1201	1234	1225	1247

Key	Display	Note
	Test H0: μ=μ0 H1: μ<μ0 input new data (Y/N) ?_	
[Y]	Input data (x) >Input.Delete.Clear.List.End ?_	(Select new data input.)
[C]	Input data (x) clear data (Y/N) ?	(Select data clear.)
[Y]	Input data (x) >Input.Delete.Clear.List.End ?_	(Data cleared.)
[I]	Input data (x) [EXE]:menu x?_	(Select data input.)
1229 [EXE]	Input data (x) [EXE]:menu x?_	(Enter first data item.)
1201 [EXE] 1234 [EXE] 1225 [EXE] 1247 [EXE]	Input data (x) [EXE]:menu x?_	(Enter remaining data items.)
[EXE]	Input data (x) >Input.Delete.Clear.List.End ?_	(Return to menu.)
[E]	Test H0: μ=μ0 H1: μ<μ0 μ0= 0 ?_	(Select End.)
1234 [EXE]	Test H0: μ=μ0 H1: μ<μ0 σ= 0 ?_	(Enter population mean.)
7.6 [EXE]	Test H0: μ=μ0 H1: μ<μ0 n= 5 ?_	(Enter population standard deviation.)
[EXE]	Test H0: μ=μ0 H1: μ<μ0 x̄= 1227.2 ?_	(Press [EXE] after checking number of data.)

Key	Display	Note
EXE	Significance level α[%] α= 5 ?_	(Press EXE after checking data mean.)
EXE	Test Ho: μ=μo H1: μ<μo	(Enter significance level. 5% is already set, so simply press EXE.)
	Test Ho: μ=μo H1: μ<μo -2.001 < 1.645 : ReJect	(Display test result.)
EXE	Test Ho: μ=μo H1: μ<μo μo= 1234 ?_	

Here, it is determined that it cannot be said that the service life of the cheaper lights is identical to that of the existing lights. The cheaper lights have shorter lives. In this example, the number of data items was limited to five for ease of understanding. In actual tests, lower number of data may cause erroneous results (standard: n≧50).

POPULATION MEAN TEST FLOWCHART (LEFT SIDED FOR KNOWN VARIANCE)



6720 POPULATION MEAN TEST (TWO-SIDED) : FOR UNKNOWN VARIANCE

Performs hypothesis testing of μ in normal distribution N (μ, σ^2 ; where μ : unknown, σ^2 : unknown).

CALCULATIONS

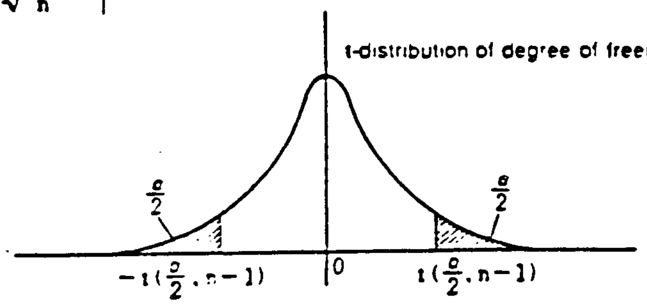
An n-size sample (x_1, $x_2 \cdots x_n$) is taken from normal distribution N (μ, σ^2). At this time, critical regions are established on both sides of t-distribution in accordance with the t-distribution of the degree of freedom (n − 1) as shown in the illustration when:

Hypothesis to be tested (Null hypothesis) $H_0 : \mu = \mu_0$

Alternative hypothesis $H_1 : \mu \neq \mu_0$

The test is performed using

$$\left| \frac{\bar{x} - \mu_0}{\sqrt{\frac{V}{n}}} \right| > t\left(\frac{\alpha}{2}, n-1\right)$$



(μ : population mean
V : unbiased variance
$\bar{x}$: sample mean
α : significance level)

$$V = \frac{\sum (x - \bar{x})^2}{n-1}$$

OPERATION

6720 [UB]

```
Test       H0:μ=μ0  H1:μ≠μ0
Input new data (Y/N) ?_
```

The display appears as illustrated above once the library is activated. At this time, either Y or N should be pressed to perform the following procedures:

Y : New data input followed by test, additional data input, data edit, statistic check.

N : Test of previously stored data, test by inputting each value.

(1) Y

[Y]

```
Input data (x)
>Input.Delete.Clear.List.End ?_
```

The menu display illustrated above appears when [Y] is pressed. One of the following character keys is then pressed to perform the corresponding function.

I (Input) : Data input (for input or addition of data).
D (Delete) : Data delete (for deletion of erroneous or unnecessary data).
C (Clear) : Data clear (for deletion of previously stored data. This operation also clears statistics).
L (List) : Statistic display (for display of number of data items, sum, sum of squares, mean, population standard deviation, sample standard deviation). [⇩] (or [EXE]) scrolls to the following data item, [⇧] to the previous item, and [⇦] or [⇨] exits the statistic display and returns to the menu.
E (End) : Advances to the test display (same as when N is pressed in the first step above).

(2) N

[N]
```
Test      Ho:μ=μo   H1:μ≠μo
μo= 0 ?_
```
(Test display)

The display appears as illustrated above when the [N] key is pressed. From this point, various parameters are entered for the test.

EXAMPLE

The following data represent test scores for a group of students. The same test has been conducted more than one hundred times in the past, with a mean score of 5.4. Use the data to determine whether or not the scores for this group of students are equivalent to past scores with a significance level of 5%.

	1	2	3	4	5
DATA	5.3	6.2	6.0	5.8	5.5

```
Test      Ho:μ=μo   H1:μ≠μo
input new data (Y/N) ?_
```
(Select new data input.)

[Y]
```
Input data (x)
>Input.Delete.Clear.List.End ?_
```
(Select data clear.)

[C]
```
Input data (x)
clear data  (Y/N) ?
```
(Data cleared.)

[Y]
```
Input data (x)
>Input.Delete.Clear.List.End ?_
```
(Select data input.)

[I]
```
Input data (x)         [EXE]:menu
x?_
```
(Enter first data item.)

5.3 [EXE]
```
Input data (x)         [EXE]:menu
x?_
```

6.2 [EXE] 6.0 [EXE] 5.8 [EXE] 5.5 [EXE]
```
Input data (x)         [EXE]:menu
x?_
```
(Enter remaining data items.)

(Return to menu.)

[EXE]
```
Input data (x)
>Input.Delete.Clear.List.End ?_
```
(Select End.)

[E]
```
Test      Ho:μ=μo   H1:μ≠μo
μo= 0 ?_
```
(Enter mean.)

5.4 [EXE]
```
Test      Ho:μ=μo   H1:μ≠μo
n= 5 ?_
```
(Press [EXE] after checking number of data.)

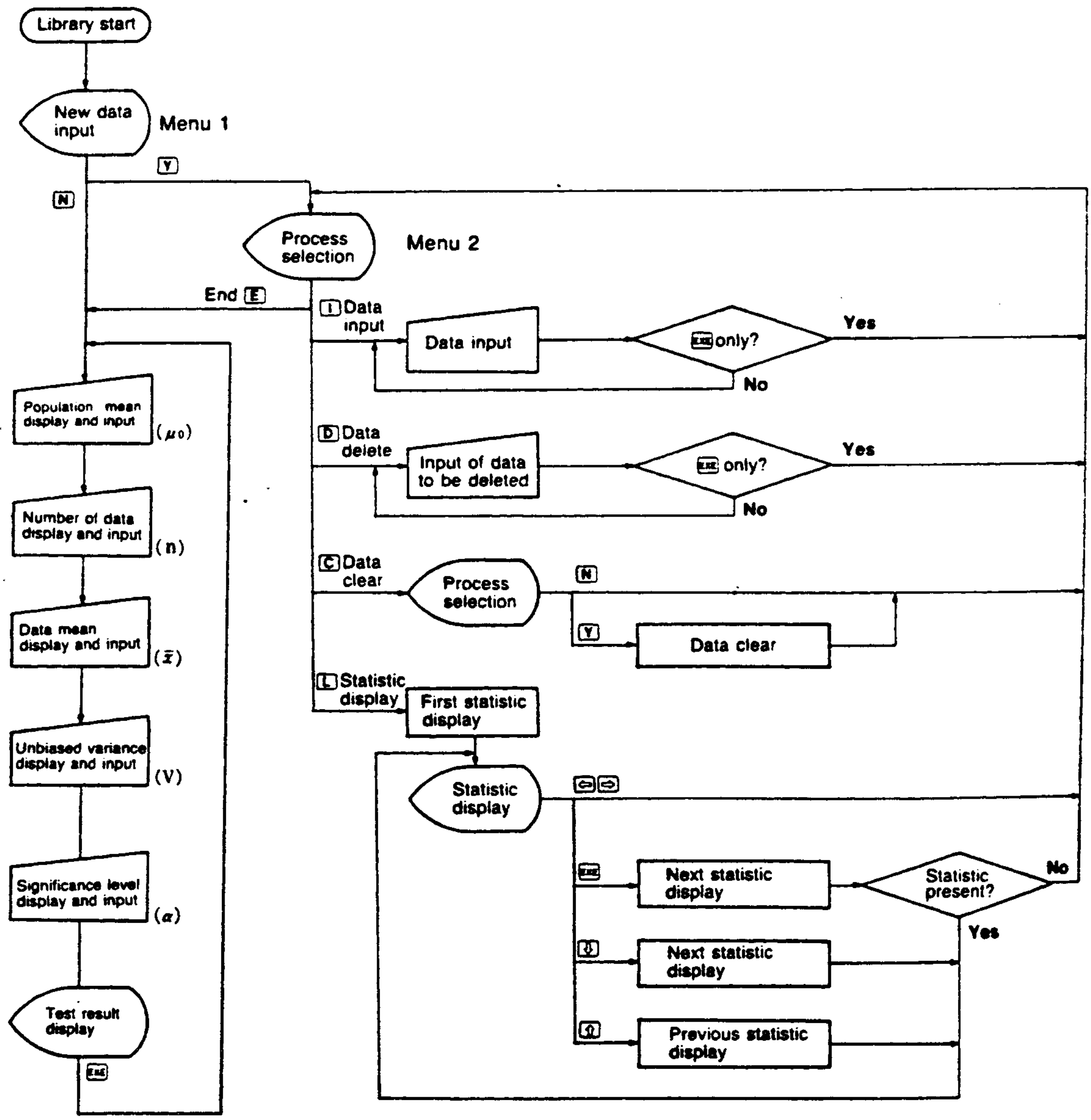
[EXE]
```
Test      Ho:μ=μo   H1:μ≠μo
x̄= 5.76 ?_
```
(Press [EXE] after checking data mean.)

[EXE]
```
Test      Ho:μ=μo   H1:μ≠μo
V= 0.133 ?_
```

EXE	Significance level α[%] α= 5 ?_	(Press EXE after checking unbiased variance.)
EXE	Test Ho : $\mu = \mu$o H1 : $\mu \neq \mu$o	(Enter significance level. 5% is already set, so simply press EXE.)
	Test Ho : $\mu = \mu$o H1 : $\mu \neq \mu$o 2.207 ≤ 2.776 : Accept	(Display test result.)
EXE	Test Ho : $\mu = \mu$o H1 : $\mu \neq \mu$o μo= 5.4 ?_	

Here, the test results can be said to be equivalent.

POPULATION MEAN TEST FLOWCHART (TWO-SIDED FOR UNKNWON VARIANCE)



6721 POPULATION MEAN TEST (RIGHT SIDED) : FOR UNKNOWN VARIANCE

Performs hypothesis testing of μ in normal distribution N (μ, σ^2 ; where μ : unknown, σ^2 : unknown).

CALCULATIONS

An n-size sample (x_1, $x_2 \cdots x_n$) is taken from normal distribution N (μ, σ^2). At this time, the critical region is established on the right side of the t-distribution in accordance with the t-distribution of the degree of freedom (n − 1) as shown in the illustration when:

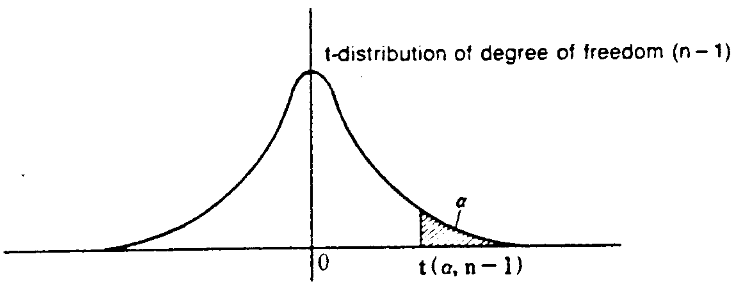
Hypothesis to be tested (Null hypothesis) $H_0 : \mu = \mu_0$
Alternative hypothesis $H_1 : \mu > \mu_0$

The test is performed using

$$\frac{\bar{x}-\mu_0}{\sqrt{\frac{V}{n}}} > t(\alpha, n-1)$$

μ_0 : population mean
V : unbiased variance
$\bar{x}$: sample mean
α : significance level

$$V = \frac{\Sigma(x-\bar{x})^2}{n-1}$$



OPERATION

6721 [LIB]

```
Test      Ho:μ=μo   H1:μ>μo
input  new  data  (Y/N) ?_
```

The display appears as illustrated above once the library is activated. At this time, either Y or N should be pressed to perform the following procedures:

Y : New data input followed by test, additional data input, data edit, statistic check.
N : Test of previously stored data, test by inputting each value.

(1) Y

[Y]

```
Input data (x)
>Input.Delete.Clear.List.End ?_
```

The menu display illustrated above appears when [Y] is pressed. One of the following character keys is then pressed to perform the corresponding function.

I (Input) : Data input (for input or addition of data).
D (Delete) : Data delete (for deletion of erroneous or unnecessary data).
C (Clear) : Data clear (for deletion of previously stored data. This operation also clears statistics).

L (List) : Statistic display (for display of number of data items, sum, sum of squares, mean, population standard deviation, sample standard deviation). [↓] (or [EXE]) scrolls to the following data item, [↑] to the previous item, and [⇦] or [⇨] exits the statistic display and returns to the menu.

E (End) : Advances to the test display (same as when N is pressed in the first step above).

(2) N

Key	Display	
[N]	Test Ho : μ=μo H1 : μ>μo μo= 0 ?_	(Test display)

The display appears as illustrated above when the [N] key is pressed. From this point, various parameters are entered for the test.

EXAMPLE

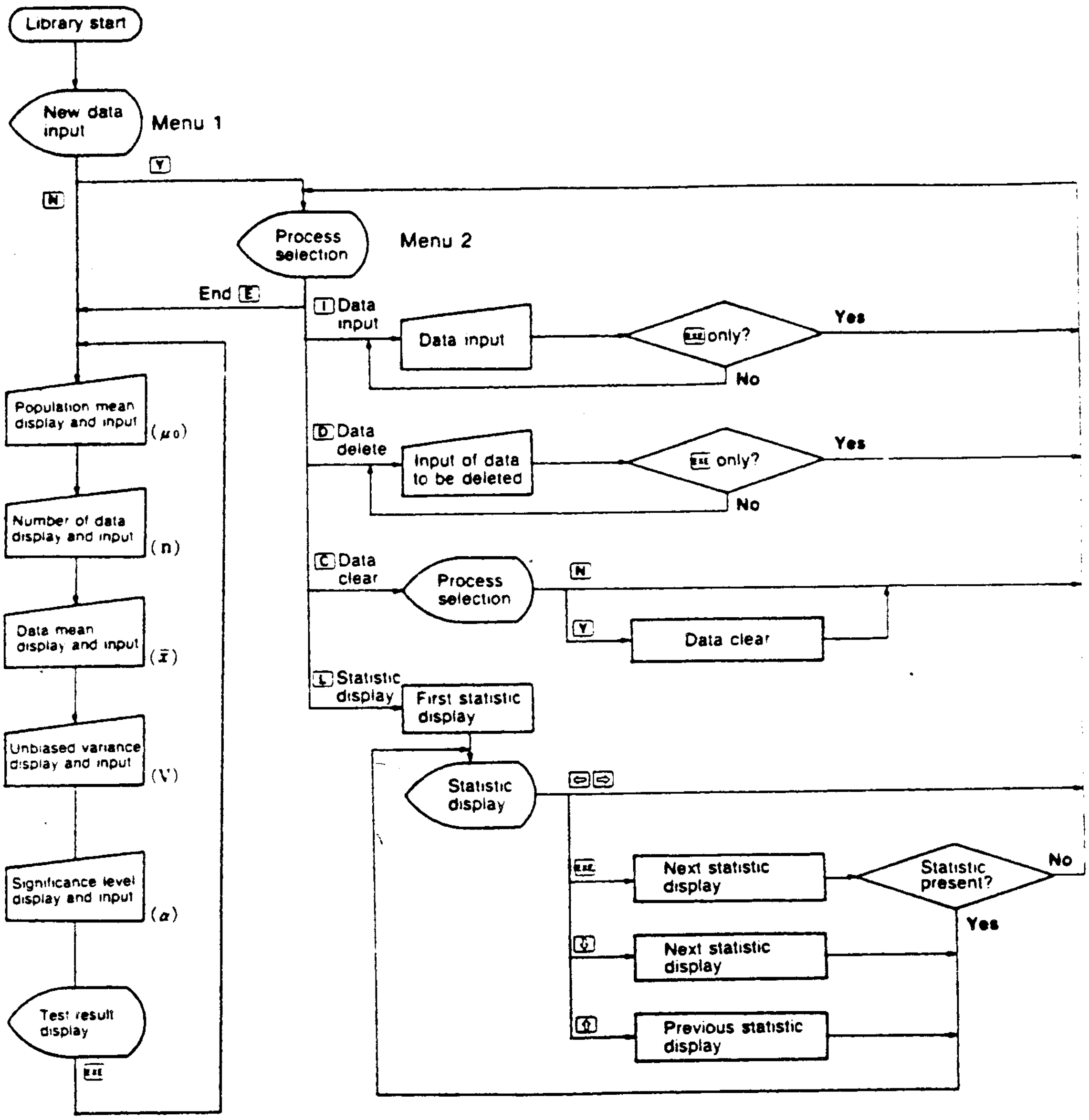
A company has conducted a survey of automobile expenses over a 5-month period. A previous 1-month survey revealed expenditures of $54.3. Using these results, determine whether or not expenditures have risen, with a significance level of 5%.

	JAN	FEB	MAR	APR	MAY
EXPENDITURES	72.4	62.3	58.4	55.4	64.8

Key	Display	
	Test Ho : μ=μo H1 : μ>μo Input new data (Y/N) ?_	
[Y]	Input data (x) >Input.Delete.Clear.List.End ?_	(Select new data input.)
[C]	Input data (x) clear data (Y/N) ?	(Select data clear.)
[Y]	Input data (x) >Input.Delete.Clear.List.End ?_	(Data cleared.)
[I]	Input data (x) [EXE]:menu x?_	(Select data input.)
72.4 [EXE]	Input data (x) [EXE]:menu x?_	(Enter first data item.)
62.3 [EXE] 58.4 [EXE] 55.4 [EXE] 64.8 [EXE]	Input data (x) [EXE]:menu x?_	(Enter remaining data items.)
[EXE]	Input data (x) >Input.Delete.Clear.List.End ?_	(Return to menu.)
[E]	Test Ho : μ=μo H1 : μ>μo μo= 0 ?_	(Select End.)
54.3 [EXE]	Test Ho : μ=μo H1 : μ>μo n= 5 ?_	(Enter population mean.)
[EXE]	Test Ho : μ=μo H1 : μ>μo x̄= 62.66 ?_	(Press [EXE] after checking number of data.)
[EXE]	Test Ho : μ=μo H1 : μ>μo V= 42.608 ?_	(Press [EXE] after checking data mean.)
[EXE]	Significance level α[%] α= 5 ?_	(Press [EXE] after checking unbiased variance.)
[EXE]	Test Ho : μ=μo H1 : μ>μo	(Enter significance level. 5% is already set, so simply press [EXE].)
	Test Ho : μ=μo H1 : μ>μo 2.864 > 2.132 : ReJect	(Display test result.)
[EXE]	Test Ho : μ=μo H1 : μ>μo μo= 54.3 ?_	

Here, it can be said that automobile expenses have increased.

POPULATION MEAN TEST FLOWCHART (RIGHT SIDED FOR UNKNOWN VARIANCE)



6722 POPULATION MEAN TEST (LEFT SIDED) : FOR UNKNOWN VARIANCE

Performs hypothesis testing of μ in normal distribution N (μ, σ^2 ; μ : unknown, σ^2 : unknown).

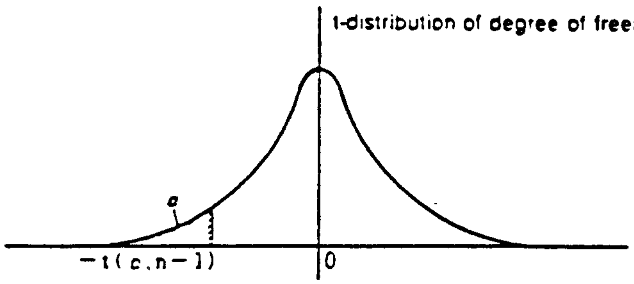
CALCULATIONS

An n-size sample (x_1, $x_2 \cdots x_n$) is taken from normal distribution N (μ, σ^2). At this time, the critical region is established on the left side of the t-distribution in accordance with the t-distribution of the degree of freedom (n − 1) as shown in the illustration when:

Hypothesis to be tested (Null hypothesis) $H_0 : \mu = \mu_0$
Alternative hypothesis $H_1 : \mu < \mu_0$

The test is performed using

$$\frac{\bar{x} - \mu_0}{\sqrt{\frac{V}{n}}} < -t(\alpha, n-1)$$



(μ_0 : population mean
V : unbiased variance
$\bar{x}$: sample mean
α : significance level)

$$V = \frac{\Sigma(x - \bar{x})^2}{n-1}$$

OPERATION

6722 [LIB]

```
Test        H0 : μ=μ0   H1 : μ<μ0
input new data (Y/N) ?_
```

The display appears as illustrated above once the library is activated. At this time, either Y or N should be pressed to perform the following procedures:

Y : New data input followed by test, additional data input, data edit, statistic check.
N : Test of previously stored data, test by inputting each value.

(1) Y

[Y]

```
Input data (x)
>Input.Delete.Clear.List.End ?_
```

The menu display illustrated above appears when [Y] is pressed. One of the following character keys is then pressed to perform the corresponding function.

I (Input) : Data input (for input or addition of data).
D (Delete) : Data delete (for deletion of erroneous or unnecessary data).
C (Clear) : Data clear (for deletion of previously stored data. This operation also clears statistics).

L (List) : Statistic display (for display of number of data items, sum, sum of squares, mean, population standard deviation, sample standard deviation).
[⇩] (or [EXE]) scrolls to the following data item, [⇧] to the previous item, and [⇦] or [⇨] exits the statistic display and returns to the menu.

E (End) : Advances to the test display (same as when N is pressed in the first step above).

(2) N

[N]
```
Test      Ho:μ=μo   H1:μ<μo
μo= 0 ?_
```
(Test display)

The display appears as illustrated above when the [N] key is pressed. From this point, various parameters are entered for the test.

EXAMPLE

The table below shows the number of requests for after service of a product at a company which recently has changed its after service procedures. Under the old system, an average of 23 requests were received per month. Use the data to determine whether the new after service system has resulted in an improvement with a significance level of 1%.

	JAN	FEB	MAR	APR	MAY
REQUESTS	16	14	11	12	7

```
Test      Ho:μ=μo   H1:μ<μo
input new data (Y/N) ?_
```

[Y]
```
Input data (x)
>Input.Delete.Clear.List.End ?_
```
(Select new data input.)

[C]
```
Input data (x)
clear data  (Y/N) ?
```
(Select data clear.)

[Y]
```
Input data (x)
>Input.Delete.Clear.List.End ?_
```
(Data cleared.)

[I]
```
Input data (x)         [EXE]:menu
x?_
```
(Select data input.)

16 [EXE]
```
Input data (x)         [EXE]:menu
x?_
```
(Enter first data item.)

14 [EXE] 11 [EXE] 12 [EXE] 7 [EXE]
```
Input data (x)         [EXE]:menu
x?_
```
(Enter remaining data items.)

[EXE]
```
Input data (x)
>Input.Delete.Clear.List.End ?_
```
(Return to menu.)

[E]
```
Test      Ho:μ=μo   H1:μ<μo
μo= 0 ?_
```
(Select End.)

23 [EXE]
```
Test      Ho:μ=μo   H1:μ<μo
n= 5 ?_
```
(Enter population mean.)

[EXE]
```
Test      Ho:μ=μo   H1:μ<μo
x̄= 12 ?_
```
(Press [EXE] after checking number of data.)

[EXE]
```
Test      Ho:μ=μo   H1:μ<μo
V= 11.5 ?_
```
(Press [EXE] after checking data mean.)

[EXE]
```
Significance level α[%]
α= 5 ?_
```
(Press [EXE] after checking unbiased variance.)

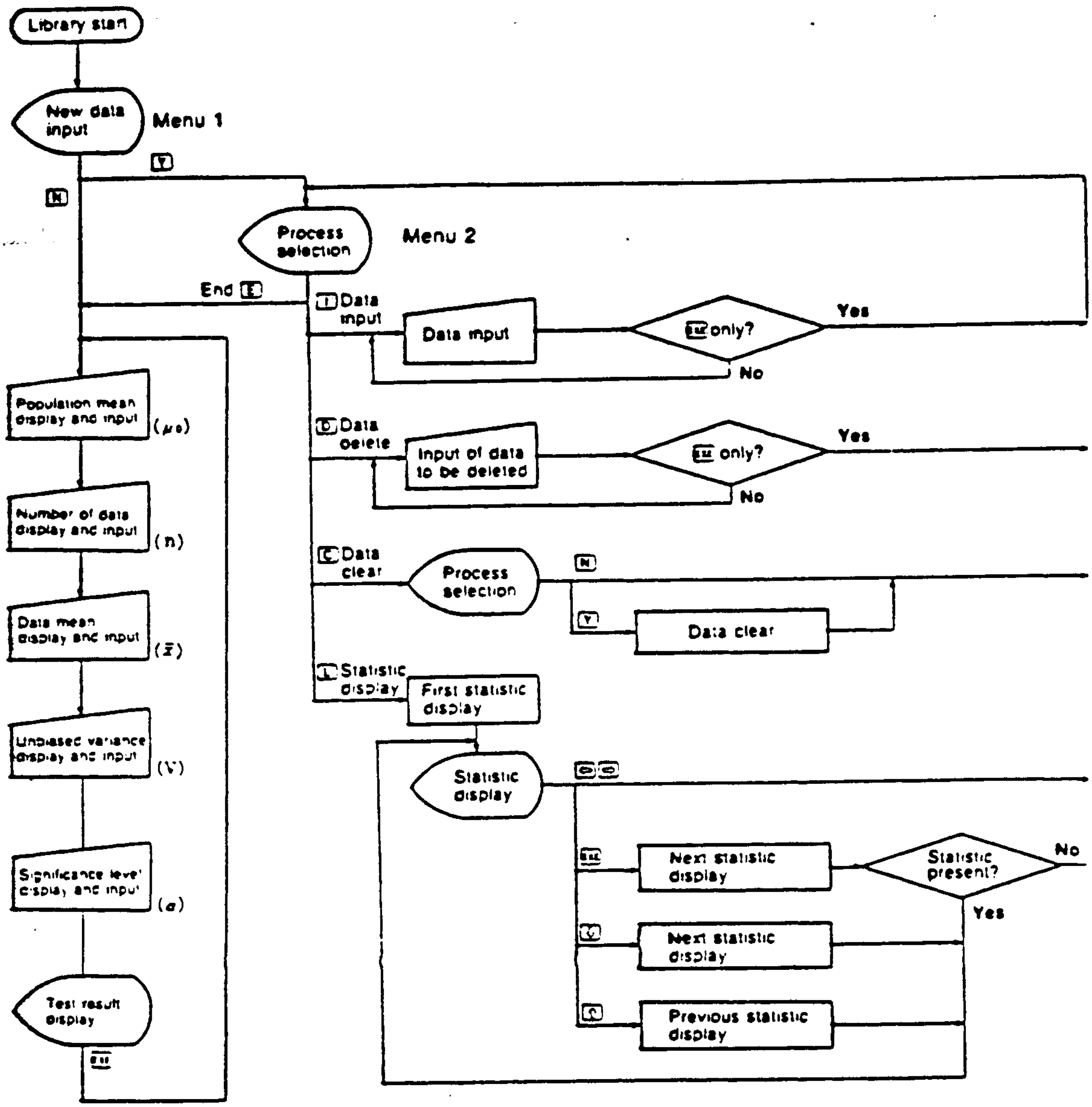
1 [EXE]
```
Test      Ho:μ=μo   H1:μ<μo
.....
```
(Enter significance level.)

```
Test      Ho:μ=μo   H1:μ<μo
-7.253 < 3.747 : Reject
```
(Display test result.)

[EXE]
```
Test      Ho:μ=μo   H1:μ<μo
μo= 23 ?_
```

Here, it can be said that the number of requests for after service has decreased under the new system.

POPULATION MEAN TEST FLOWCHART (LEFT SIDED FOR UNKNOWN VARIANCE)



6730 POPULATION VARIANCE TEST (TWO-SIDED)

Performs hypothesis testing of σ^2 in normal distribution N (μ, σ^2 ; where μ : unknown, σ^2 : unknown).

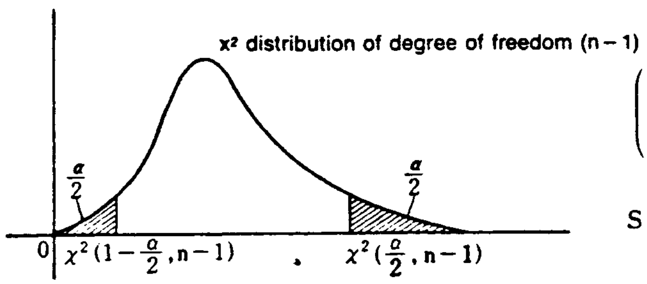
CALCULATIONS

An n-size sample (x_1, $x_2 \cdots x_n$) is taken from normal distribution N (μ, σ^2). At this time, critical regions are established on both sides of the x^2 distribution in accordance with the x^2-distribution of the degree of freedom (n − 1) as shown in the illustration when:

Hypothesis to be tested (Null hypothesis) $H_0 : \sigma^2 = \sigma_0^2$
Alternative hypothesis $H_1 : \sigma^2 \neq \sigma_0^2$

The test is performed using

$$\frac{S}{\sigma_0^2} < \chi^2\left(1-\frac{\alpha}{2}, n-1\right) \text{ or } \frac{S}{\sigma_0^2} > \chi^2\left(\frac{\alpha}{2}, n-1\right)$$



σ_0^2 : population variance
S : sum of squares
α : significance level

$$S = \Sigma (x - \bar{x})^2$$

OPERATION

6730 LB

```
Test      H0:σ2=σ02   H1:σ2≠σ02
Input new data (Y/N) ?_
```

The display appears as illustrated above once the library is activated. At this time, either Y or N should be pressed to perform the following procedures:

Y : New data input followed by test, additional data input, data edit, statistic check.
N : Test of previously stored data, test by inputting each value.

(1) Y

[Y]

```
Input data (x)
>Input.Delete.Clear.List.End ?_
```

The menu display illustrated above appears when [Y] is pressed. One of the following character keys is then pressed to perform the corresponding function.

I (Input) : Data input (for input or addition of data).
D (Delete) : Data delete (for deletion of erroneous or unnecessary data).
C (Clear) : Data clear (for deletion of previously stored data. This operation also clears statistics).

L (List) : Statistic display (for display of number of data items, sum, sum of squares, mean, population standard deviation, sample standard deviation).
[↓] (or [EXE]) scrolls to the following data item, [↑] to the previous item, and [⊟] or [⊟] exits the statistic display and returns to the menu.

E (End) : Advances to the test display (same as when N is pressed in the first step above).

(2) N

Key	Display	
[N]	Test Ho:σ²=σo² H1:σ²≠σo² σo²= 0 ?_	(Test display)

The display appears as illustrated above when the [N] key is pressed. From this point, various parameters are entered for the test.

EXAMPLE

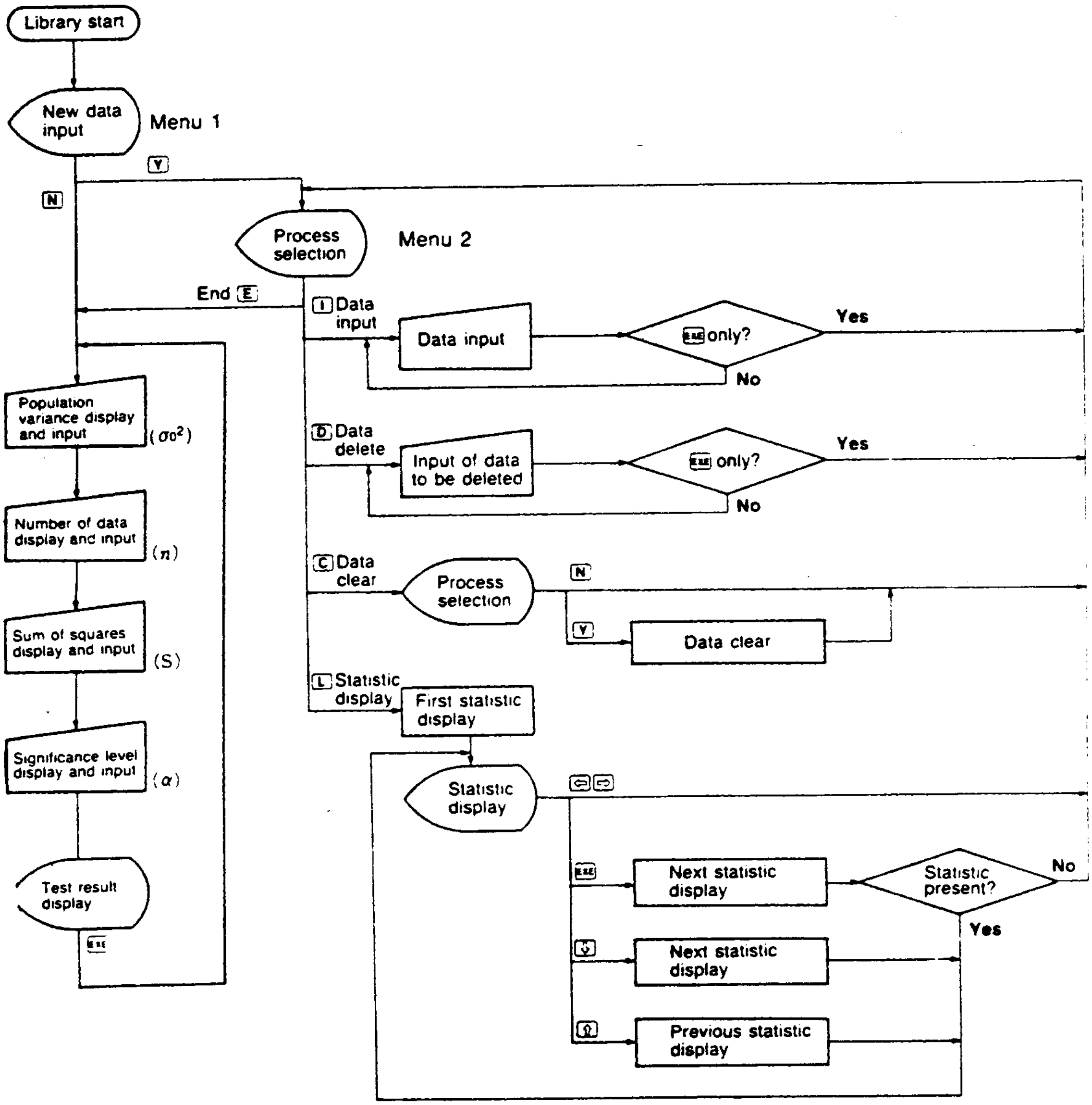
The following data represent the entrance examination results of five students. To date, the variance of scores for this test has been 70. Use the data to determine whether or not the variance of this year's scores is equvalent to past scores with a significance level of 1%.

	1	2	3	4	5
POINTS	183	174	191	168	171

Key	Display	
	Test Ho:σ²=σo² H1:σ²≠σo² input new data (Y/N) ?_	
[Y]	Input data (x) >Input.Delete.Clear.List.End ?_	(Select new data input.)
[C]	Input data (x) clear data (Y/N) ?	(Select data clear.)
[Y]	Input data (x) >Input.Delete.Clear.List.End ?_	(Data cleared.)
[I]	Input data (x) [EXE]:menu x?_	(Select data input.)
183 [EXE]	Input data (x) [EXE]:menu x?_	(Enter first data item.)
174 [EXE] 191 [EXE] 168 [EXE] 171 [EXE]		
	Input data (x) [EXE]:menu x?_	(Enter remaining data items.)
[EXE]	Input data (x) >Input.Delete.Clear.List.End ?_	(Return to menu.)
[E]	Test Ho:σ²=σo² H1:σ²≠σo² σo²= 0 ?_	(Select End.)
70 [EXE]	Test Ho:σ²=σo² H1:σ²≠σo² n= 5 ?_	(Enter population variance.)
[EXE]	Test Ho:σ²=σo² H1:σ²≠σo² S= 357.2 ?_	(Press [EXE] after checking number of data.)
[EXE]	Significance level α(%) α= 5 ?_	(Press [EXE] after checking sum of squares.)
1 [EXE]	Test Ho:σ²=σo² H1:σ²≠σo²	(Enter significance level.)
	5.103 ≤ 14.86 5.103 ≥ 0.207 : Accept	(Display test result.)
[EXE]	Test Ho:σ²=σo² H1:σ²≠σo² σo²= 70 ?_	

Here, the variance of this year's scores is equivalent to last year's scores.

POPULATION VARIANCE TEST FLOWCHART (TWO-SIDED)



6731 POPULATION VARIANCE TEST (RIGHT SIDED)

Performs hypothesis testing of σ^2 in normal distribution N (μ, σ^2 ; where μ : unknown, σ^2 : unknown).

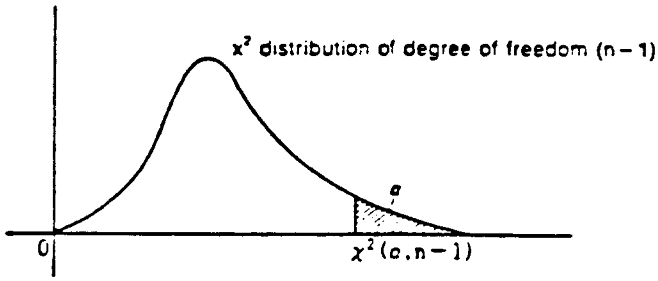
CALCULATIONS

An n-size sample (x_1, $x_2 \cdots x_n$) is taken from normal distribution N (μ, σ^2). At this time, a critical region is established on the right side of the χ^2 distribution in accordance with the χ^2-distribution of the degree of freedom (n − 1) as shown in the illustration when:

Hypothesis to be tested (Null hypothesis) $H_0 : \sigma^2 = \sigma_0^2$
Alternative hypothesis $H_1 : \sigma^2 > \sigma_0^2$

The test is performed using

$$\frac{S}{\sigma_0^2} > \chi^2(\alpha, n-1)$$



(σ_0^2 : population variance
S : sum of squares
α : significance level)

$$S = \Sigma(x - \bar{x})^2$$

OPERATION

6731 [LIB]

```
Test      H0:σ²=σ0²  H1:σ²>σ0²
input new data (Y/N) ?_
```

The display appears as illustrated above once the library is activated. At this time, either Y or N should be pressed to perform the following procedures:

Y : New data input followed by test, additional data input, data edit, statistic check.
N : Test of previously stored data, test by inputting each value.

(1) Y

[Y]

```
Input data (x)
>Input.Delete.Clear.List.End ?_
```

The menu display illustrated above appears when [Y] is pressed. One of the following character keys is then pressed to perform the corresponding function.

I (Input) : Data input (for input or addition of data).
D (Delete) : Data delete (for deletion of erroneous or unnecessary data).
C (Clear) : Data clear (for deletion of previously stored data. This operation also clears statistics).

L (List) : Statistic display (for display of number of data items, sum, sum of squares, mean, population standard deviation, sample standard deviation). [↓] (or [EXE]) scrolls to the following data item, [↑] to the previous item, and [⇐] or [⇒] exits the statistic display and returns to the menu.

E (End) : Advances to the test display (same as when N is pressed in the first step above).

(2) N

Key	Display	
[N]	Test Ho: σ²=σo² H1: σ²>σo² σo²= 0 ?_	(Test display)

The display appears as illustrated above when the [N] key is pressed, From this point, various parameters are entered for the test.

EXAMPLE

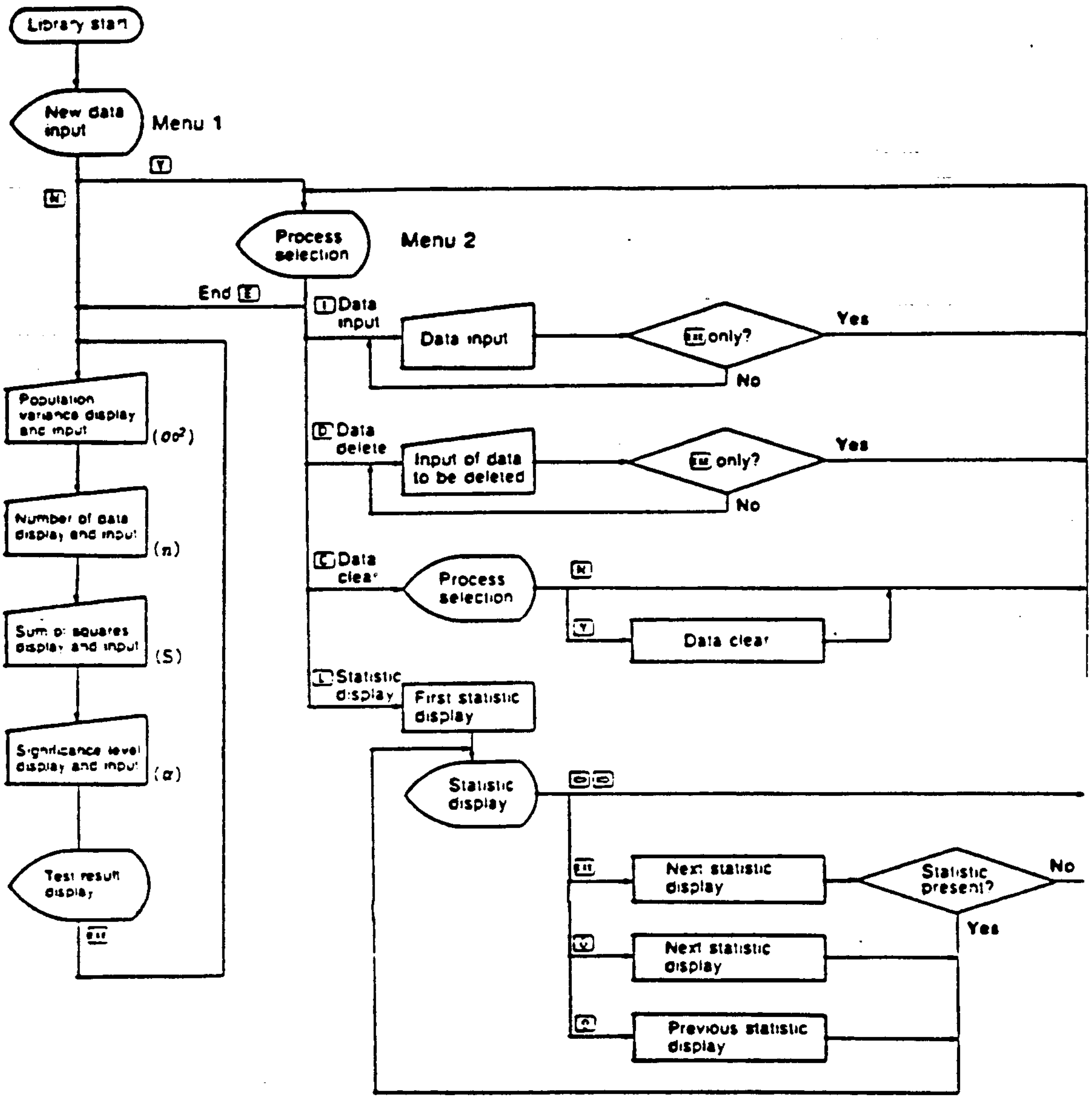
A company purchased blades from Company B because the cost of the blades was less than those purchased from their usual supplier, Company A. The data in the table below represent the measured lengths of items cut with the Company B blades. To date, the variance of lengths of items cut with Company A blades has been 0.015. Use the data to compare the performance of the two companies' blades with a significance level of 1%.

	1	2	3	4	5
LENGTH	10.1	9.8	10.2	9.7	9.8

Key	Display	
	Test Ho: σ²=σo² H1: σ²>σo² input new data (Y/N) ?_	
[Y]	Input data (x) >Input.Delete.Clear.List.End ?_	(Select new data input.)
[C]	Input data (x) clear data (Y/N) ?	(Select data clear.)
[Y]	Input data (x) >Input.Delete.Clear.List.End ?_	(Data cleared.)
[I]	Input data (x) [EXE]:menu x?_	(Select data input.)
10.1 [EXE]	Input data (x) [EXE]:menu x?_	(Enter first data item.)
9.8 [EXE] 10.2 [EXE] 9.7 [EXE] 9.8 [EXE]	Input data (x) [EXE]:menu x?_	(Enter remaining data items.)
[EXE]	Input data (x) >Input.Delete.Clear.List.End ?_	(Return to menu.)
[E]	Test Ho: σ²=σo² H1: σ²>σo² σo²= 0 ?_	(Select End.)
0.015 [EXE]	Test Ho: σ²=σo² H1: σ²>σo² n= 5 ?_	(Enter population variance.)
[EXE]	Test Ho: σ²=σo² H1: σ²>σo² S= 0.188 ?_	(Press [EXE] after checking number of data.)
[EXE]	Significance level α[%] α= 5 ?_	(Press [EXE] after checking sum of squares.)
1 [EXE]	Test Ho: σ²=σo² H1: σ²>σo²	(Enter significance level.)
	Test Ho: σ²=σo² H1: σ²>σo² 12.53 ≦ 13.28 : Accept	(Display test result.)
[EXE]	Test Ho: σ²=σo² H1: σ²>σo² σo²= 0.015 ?_	

Here, it is determined that the performance of both companies' blades are equivalent.

POPULATION VARIANCE TEST FLOWCHART (RIGHT SIDED)



6732 POPULATION VARIANCE TEST (LEFT SIDED)

Performs hypothesis testing of σ^2 in normal distribution N (μ, σ^2 ; where μ : unknown, σ^2 : unknown).

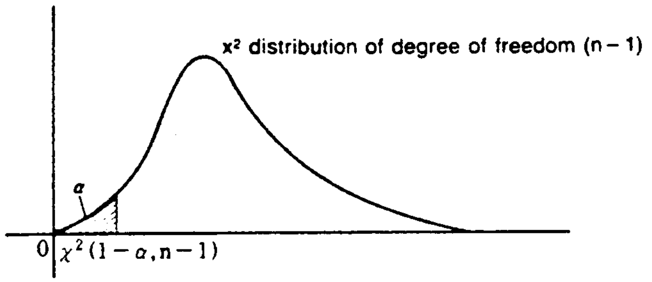
CALCULATIONS

An n-size sample ($x_1, x_2 \cdots x_n$) is taken from normal distribution N (μ, σ^2). At this time, a critical region is established on the left side of the x^2 distribution in accordance with the x^2 distribution of the degree of freedom (n − 1) as shown in the illustration when:

Hypothesis to be tested (Null hypothesis) $H_0 : \sigma^2 = \sigma_0^2$
Alternative hypothesis $H_1 : \sigma^2 < \sigma_0^2$

The test is performed using

$$\frac{S}{\sigma_0^2} < \chi^2(1-\alpha, n-1)$$



σ_0^2 : population variance
S : sum of squares
α : significance level

$S = \Sigma(x - \bar{x})^2$

OPERATION

6732 [LB]

```
Test      H0:σ2=σ02   H1:σ2<σ02
input new data (Y/N) ?_
```

The display appears as illustrated above once the library is activated. At this time, either Y or N should be pressed to perform the following procedures:

Y : New data input followed by test, additional data input, data edit, statistic check.
N : Test of previously stored data, test by inputting each value.

(1) Y

[Y]

```
Input data (x)
>Input.Delete.Clear.List.End ?_
```

The menu display illustrated above appears when [Y] is pressed. One of the following character keys is then pressed to perform the corresponding function.

I (Input) : Data input (for input or addition of data).
D (Delete) : Data delete (for deletion of erroneous or unnecessary data).
C (Clear) : Data clear (for deletion of previously stored data. This operation also clears statistics).

L (List) : Statistic display (for display of number of data items, sum, sum of squares, mean, population standard deviation, sample standard deviation).
[↓] (or [EXE]) scrolls to the following data item, [↑] to the previous item, and [⊟] or [⊟] exits the statistic display and returns to the menu.

E (End) : Advances to the test display (same as when N is pressed in the first step above).

(2) N

Key	Display	
[N]	Test H0:σ²=σ0² H1:σ²<σ0² σ0²= 0 ?_	(Test display)

The display appears as illustrated above when the [N] key is pressed. From this point, various parameters are entered for the test.

EXAMPLE

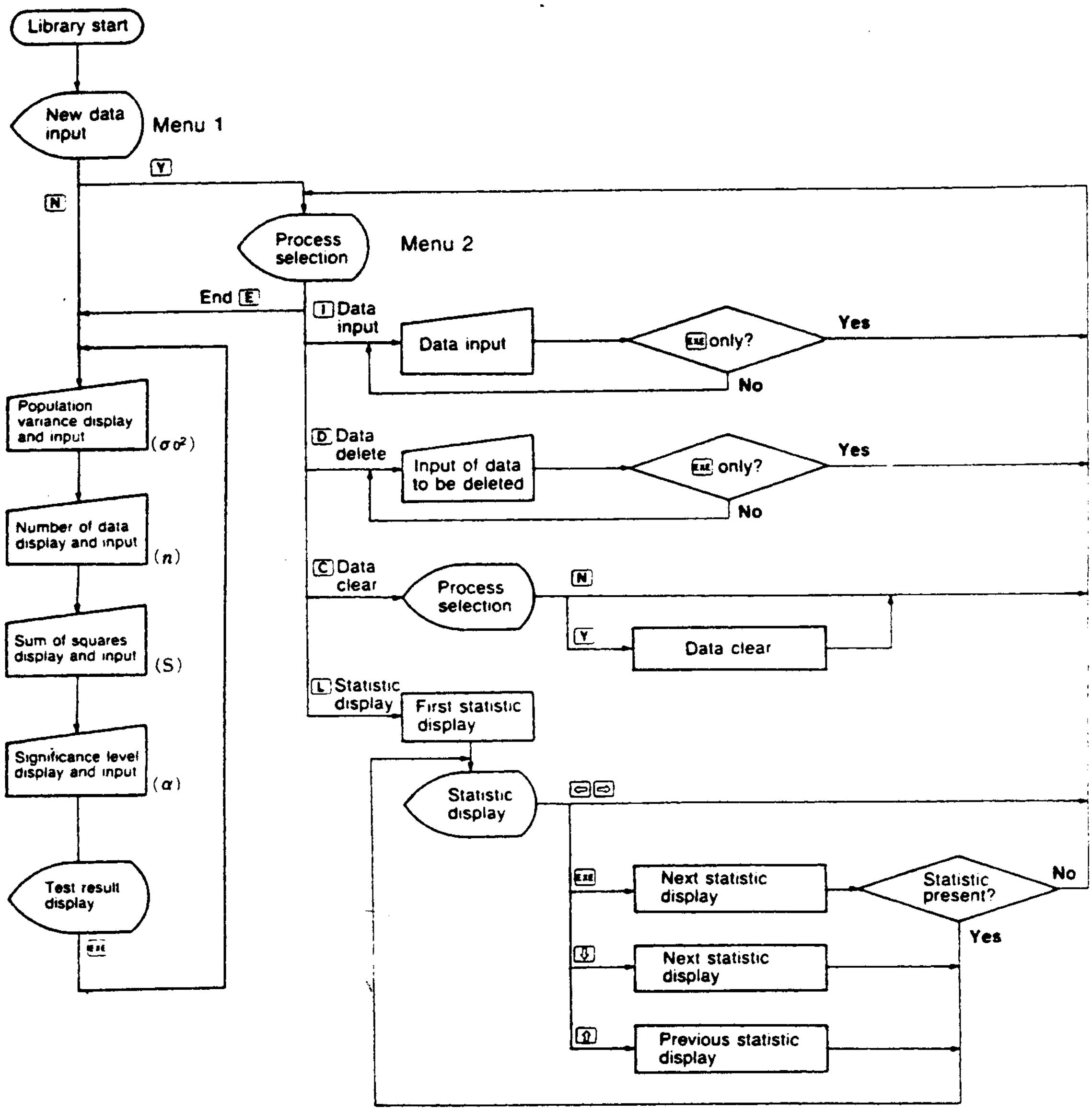
A company has purchased a new production machinery. The data in the table below represent the production capacity of the new machinery. To date, the variance of production capacity for the old machinery has been 0.1. Use the data to compare the performance of the machinery with a significance level of 1%.

	1	2	3	4	5
WEIGHT (g)	70.0	69.9	70.1	70.1	69.8

Key	Display	
	Test H0:σ²=σ0² H1:σ²<σ0² input new data (Y/N) ?_	
[Y]	Input data (x) >Input.Delete.Clear.List.End ?_	(Select new data input.)
[C]	Input data (x) clear data (Y/N) ?	(Select data clear.)
[Y]	Input data (x) >Input.Delete.Clear.List.End ?_	(Data cleared.)
[I]	Input data (x) [EXE]:MENU x?_	(Select data input.)
70.0 [EXE]	Input data (x) [EXE]:MENU x?_	(Enter first data item.)
69.9 [EXE] 70.1 [EXE] 70.1 [EXE] 69.8 [EXE]		
	Input data (x) [EXE]:MENU x?_	(Enter remaining data items.)
[EXE]	Input data (x) >Input.Delete.Clear.List.End ?_	(Return to menu.)
[E]	Test H0:σ²=σ0² H1:σ²<σ0² σ0²= 0 ?_	(Select End.)
0.1 [EXE]	Test H0:σ²=σ0² H1:σ²<σ0² n= 5 ?_	(Enter population variance.)
[EXE]	Test H0:σ²=σ0² H1:σ²<σ0² S= 0.068 ?_	(Press [EXE] after checking number of data.)
[EXE]	Significance level α(%) α= 5 ?_	(Press [EXE] after checking sum of squares.)
1 [EXE]	Test H0:σ²=σ0² H1:σ²<σ0²	(Enter significance level.)
	Test H0:σ²=σ0² H1:σ²<σ0² 0.68 ≧ 0.2971 : Accept	(Display test result.)
[EXE]	Test H0:σ²=σ0² H1:σ²<σ0² σ0²= 0.1 ?_	

Here, it is determined that the performance of the new machinery is equivalent to that of the old machinery.

POPULATION VARIANCE TEST FLOWCHART (LEFT SIDED)



6740 VARIANCE RATIO TEST (TWO-SIDED)

Performs test of hypotheses σ_1^2 and σ_2^2 in two normal distributions N (μ_1, σ_1^2 ; where μ_1 : unknown, σ_1^2 : unknown) and N (μ_2, σ_2^2 ; where μ_2 : unknown, σ_2^2 : unknown).

CALCULATIONS

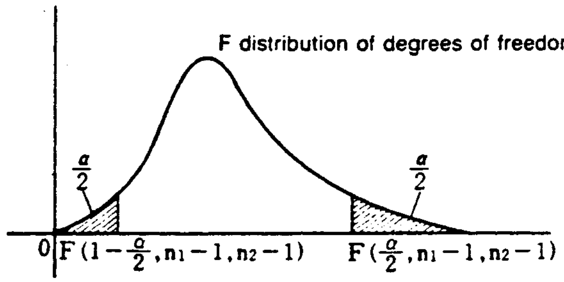
An n_1-size sample (x_{11}, $x_{12} \cdots x_{1n_1}$) is taken from normal distribution N (μ_1, σ_1^2) and an n_2 sample (x_{21}, $x_{22} \cdots x_{2n_2}$) from normal distribution N (μ_2, σ_2^2). At this time, critical regions are established on both sides of the F distribution in accordance with the F distribution of the degrees of freedom ($n_1 - 1$, $n_2 - 1$) as shown in the illustration when:

Hypothesis to be tested (Null hypothesis) H_0 : $\sigma_1^2 = \sigma_2^2$
Alternative hypothesis H_1 : $\sigma_2^2 \neq \sigma_2^2$

The test is performed using

$V_1 < V_2$: $\frac{V_2}{V_1} > F\left(\frac{\alpha}{2}, n_2-1, n_1-1\right)$

$V_1 > V_2$: $\frac{V_1}{V_2} > F\left(\frac{\alpha}{2}, n_1-1, n_2-1\right)$



μ_1 : population mean 1
μ_2 : population mean 2
σ_1^2 : population variance 1
σ_2^2 : population variance 2
V_1 : unbiased variance 1
V_2 : unbiased variance 2
α : significance level

$V = \frac{\Sigma(x-\bar{x})^2}{n-1}$

OPERATION

6740 [LB]

```
Test        Ho:σ1²=σ2²    H1:σ1²≠σ2²
Input new data x1  (Y/N) ?_
```

The display appears as illustrated above once the library is activated. At this time, either Y or N should be pressed to perform the following procedures:

Y : New data input followed by test, additional data input, data edit, statistic check.
N : Test of previously stored data, test by inputting each value.

(1) Y

[Y]

```
Input data (x1)
>Input.Delete.Clear.List.End ?_
```

The menu display illustrated above appears when [Y] is pressed. One of the following character keys is then pressed to perform the corresponding function.

I (Input) : Data input (for input or addition of data).
D (Delete) : Data delete (for deletion of erroneous or unnecessary data).
C (Clear) : Data clear (for deletion of previously stored data. This operation also clears statistics).

L (List) : Statistic display (for display of number of data items, sum, sum of squares, mean, population standard deviation, sample standard deviation).
[⇩] (or [EXE]) scrolls to the following data item, [⇧] to the previous item, and [⇦] or [⇨] exits the statistic display and returns to the menu.

E (End) : Advances to the test display (same as when N is pressed in the first step above).

(2) N

[N]

Test $H_0: \sigma_1^2 = \sigma_2^2$ $H_1: \sigma_1^2 \neq \sigma_2^2$
input new data x_2 (Y/N) ?_

The display appears as illustrated above when the [N] key is pressed. Note that this display is almost identical to the initial display which appears immediately after entering library operations. The difference, however, is that the question concerning new data input here is for data items x_{21} through x_{2n_2}, while the data input being queried on the original display is for data items x_{11} through x_{1n_1}.

(2-1) Y

Same result as that produced by pressing [Y] in step (1) above. Note, however, that the data being entered or corrected here is x_{21} through x_{2n_2}.

(2-2) N

[N]

Test $H_0: \sigma_1^2 = \sigma_2^2$ $H_1: \sigma_1^2 \neq \sigma_2^2$
n_1= 5 ?_

(Number of data display)

The display appears as illustrated above when the [N] key is pressed. The value indicated for n shows the number of x_1 ($x_{11} \sim x_{1n_1}$) data currently stored in memory.

- $n_1 = 0$: Test cannot be performed, so this should be corrected to the reuired data.
- Number of data input (following [Y] above) and value of n differ : Confirm that some data have not been omitted during the input or that two or more data items have been input together for a single entry. In either case, terminate the library operation. Enter the library again and add, delete, or reinput data as required.
- Number of data input (following [Y] above) matches value of n : Press [EXE].
When [EXE] is pressed, a display similar to that above is produced for x_2 ($x_{21} \sim x_{2n_2}$) data items. After confirmation and/or corrections as described in (2-2), press [EXE] to continue.

EXAMPLE

The following data represent measurement results on samples taken from two lines in a factory. Use the data to determine whether or not production on the two lines differ with a significance level of 5%.

	1	2	3	4	5
LINE A	37.2	38.1	39.9	37.5	36.1
LINE B	36.1	35.2	37.7	35.6	—

Test $H_0: \sigma_1^2 = \sigma_2^2$ $H_1: \sigma_1^2 \neq \sigma_2^2$
input new data x_1 (Y/N) ?_

[Y]

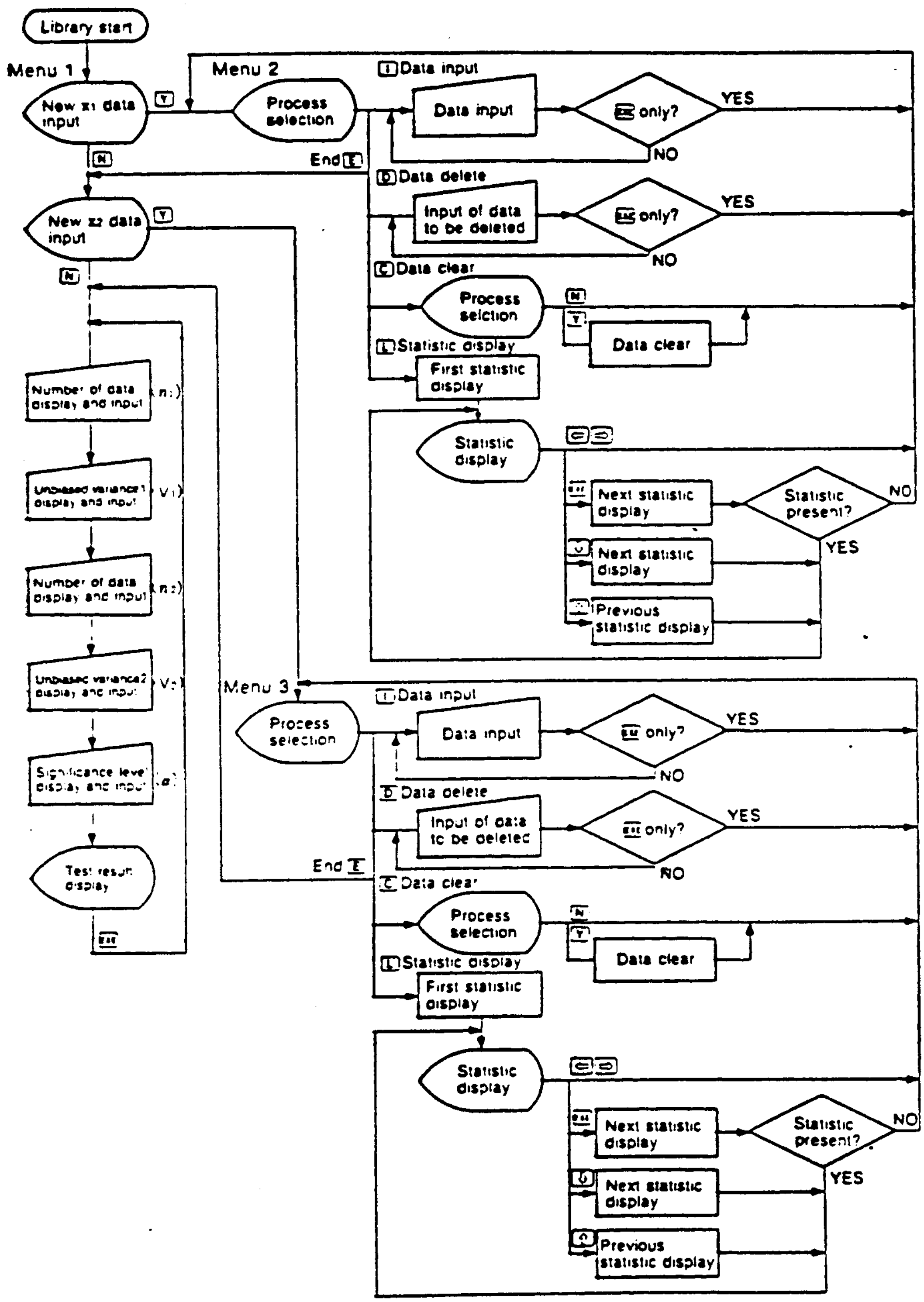
Input data (x_1)
>Input.Delete.Clear.List.End ?_

(Select new data input.)

Key operation	Display	Comment
[C]	Input data (x1) clear data (Y/N) ?	(Select data clear.)
[Y]	Input data (x1) >Input.Delete.Clear.List.End ?_	(Data cleared.)
[I]	Input data (x1) [EXE]:menu x1?_	(Select data input.)
37.2 [EXE]	Input data (x1) [EXE]:menu x1?_	(Enter first data item for LINE A.)
38.1 [EXE] 39.9 [EXE] 37.5 [EXE] 36.1 [EXE]	Input data (x1) [EXE]:menu x1?_	(Enter remaining data items.)
[EXE]	Input data (x1) >Input.Delete.Clear.List.End ?_	(Return to menu.)
[E]	Test $H_0:\sigma_1^2=\sigma_2^2$ $H_1:\sigma_1^2\neq\sigma_2^2$ Input new data x2 (Y/N) ?_	(Select End.)
[Y]	Input data (x2) >Input.Delete.Clear.List.End ?_	(Select new data input.)
[C]	Input data (x2) clear data (Y/N) ?	(Select data clear.)
[Y]	Input data (x2) >Input.Delete.Clear.List.End ?_	(Data cleared.)
[I]	Input data (x2) [EXE]:menu x2?_	(Select data input.)
36.1 [EXE]	Input data (x2) [EXE]:menu x2?_	(Enter first data item for LINE B.)
35.2 [EXE] 37.7 [EXE] 35.6 [EXE]	Input data (x2) [EXE]:menu x2?_	(Enter remaining data items.)
[EXE]	Input data (x2) >Input.Delete.Clear.List.End ?_	(Return to menu.)
[E]	Test $H_0:\sigma_1^2=\sigma_2^2$ $H_1:\sigma_1^2\neq\sigma_2^2$ n_1= 5 ?_	(Select End.)
[EXE]	Test $H_0:\sigma_1^2=\sigma_2^2$ $H_1:\sigma_1^2\neq\sigma_2^2$ V_1= 1.958 ?_	(Press [EXE] after checking number of LINE A data.)
[EXE]	Test $H_0:\sigma_1^2=\sigma_2^2$ $H_1:\sigma_1^2\neq\sigma_2^2$ n_2= 4 ?_	(Press [EXE] after checking unbiased variance of LINE A data.)
[EXE]	Test $H_0:\sigma_1^2=\sigma_2^2$ $H_1:\sigma_1^2\neq\sigma_2^2$ V_2= 1.203333333 ?_	(Press [EXE] after checking number of LINE B data.)
[EXE]	Significance level α(%) α= 5 ?_	(Press [EXE] after checking unbiased variance of LINE B data.)
[EXE]	Test $H_0:\sigma_1^2=\sigma_2^2$ $H_1:\sigma_1^2\neq\sigma_2^2$	(Enter significance level. 5% is already set, so simply press [EXE].)
	Test $H_0:\sigma_1^2=\sigma_2^2$ $H_1:\sigma_1^2\neq\sigma_2^2$ 1.627 ≤ 15.1 : Accept	(Display test result.)
[EXE]	Test $H_0:\sigma_1^2=\sigma_2^2$ $H_1:\sigma_1^2\neq\sigma_2^2$ n_1= 5 ?_	

Here, it is determined that the variance for the output of both lines are equivalent.

RIANCE RATIO TEST FLOWCHART (TWO-SIDED)



6741 VARIANCE RATIO TEST (RIGHT SIDED)

Performs hypotheses testing of σ_1^2 and σ_2^2 in two normal distributions N (μ_1, σ_1^2 ; where μ_1 : unknown, σ_1^2 : unknown) and N (μ_2, σ_2^2 ; where μ_2 : unknown, σ_2^2 : unknown).

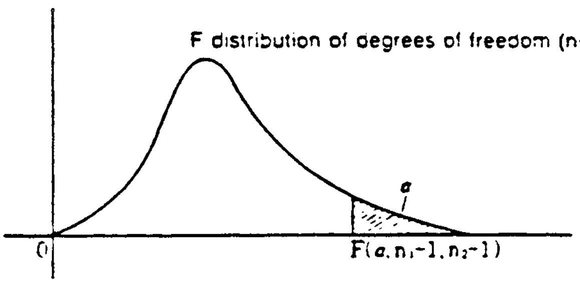
CALCULATIONS

An n_1-size sample (x_{11}, $x_{12} \cdots x_{1n_1}$) is taken from normal distribution N (μ_1, σ_1^2) and an n_2 sample (x_{21}, $x_{22} \cdots x_{2n_2}$) from normal distribution N (μ_2, σ_2^2). At this time, the critical region is established on the right side of the F distribution in accordance with the F distribution of the degrees of freedom ($n_1 - 1$, $n_2 - 1$) as shown in the illustration when:

Hypothesis to be tested (Null hypothesis) $H_0 : \sigma_1^2 = \sigma_2^2$
Alternative hypothesis $H_1 : \sigma_1^2 > \sigma_2^2$

The test is performed using

$$\frac{V_1}{V_2} > F(\alpha, n_1 - 1, n_2 - 1)$$



- μ_1 : population mean 1
- μ_2 : population mean 2
- σ_1^2 : population variance 1
- σ_2^2 : population variance 2
- V_1 : unbiased variance 1
- V_2 : unbiased variance 2
- α : significance level

$$V = \frac{\Sigma (x - \bar{x})^2}{n - 1}$$

OPERATION

6741 [LB]

```
Test          H0 : σ1²=σ2²   H1 : σ1²>σ2²
input new data x1 (Y/N) ?_
```

The display appears as illustrated above once the library is activated. At this time, either Y or N should be pressed to perform the following procedures:

Y : New data input followed by test, additional data input, data edit, statistic check.
N : Test of previously stored data, test by inputting each value.

(1) Y

[Y]

```
Input data (x1)
>Input.Delete.Clear.List.End ?_
```

The menu display illustrated above appears when [Y] is pressed. One of the following character keys is then pressed to perform the corresponding function.

I (Input) : Data input (for input or addition of data).
D (Delete) : Data delete (for deletion of erroneous or unnecessary data).
C (Clear) : Data clear (for deletion of previously stored data. This operation also clears statistics).

(List) : Statistic display (for display of number of data items, sum, sum of squares, mean, population standard deviation, sample standard deviation).
[↓] (or [EXE]) scrolls to the following data item, [↑] to the previous item, and [⊖] or [⊖] exits the statistic display and returns to the menu.

(End) : Advances to the test display (same as when N is pressed in the first step above).

!) N

D

```
Test        H0:σ1²=σ2²  H1:σ1²>σ2²
Input new data x2 (Y/N) ?_
```

he display appears as illustrated above when the [N] key is pressed. Note that this display ; almost identical to the initial display which appears immediately after entering library perations. The difference, however, is that the question concerning new data input here ; for data items x_{2_1} through x_{2n_2}, while the data input being queried on the original display ; for data items x_{1_1} through x_{1n_1}.

2-1) Y

;ame result as that produced by pressing [Y] in step (1) above. Note, however, that the data eing entered or corrected here is x_{2_1} through x_{2n_2}.

2-2) N

D

```
Test        H0:σ1²=σ2²  H1:σ1²>σ2²
n1= 5 ?_
```

(Number of data display)

he display appears as illustrated above when the [N] key is pressed. The value indicated ɔr n shows the number of x_1 ($x_{1_1}-x_{1n_1}$) data currently stored in memory.

$n_1=0$: Test cannot be performed, so the required data should be corrected to the required data.
Number of data input (following [Y] above) and value of n differ : Confirm that some data have not been omitted during the input or that two or more data items have been input together for a single entry. In either case, terminate the library operation. Enter the library again and add, delete, or reinput data as required.
Number of data input (following [Y] above) matches value of n : Press [EXE].
When [EXE] is pressed, a display similar to that above is produced for x_2 ($x_{2_1}-x_{2n_2}$) data items. After confirmation and/or corrections as described in (2-2), press [EXE] to continue.

EXAMPLE

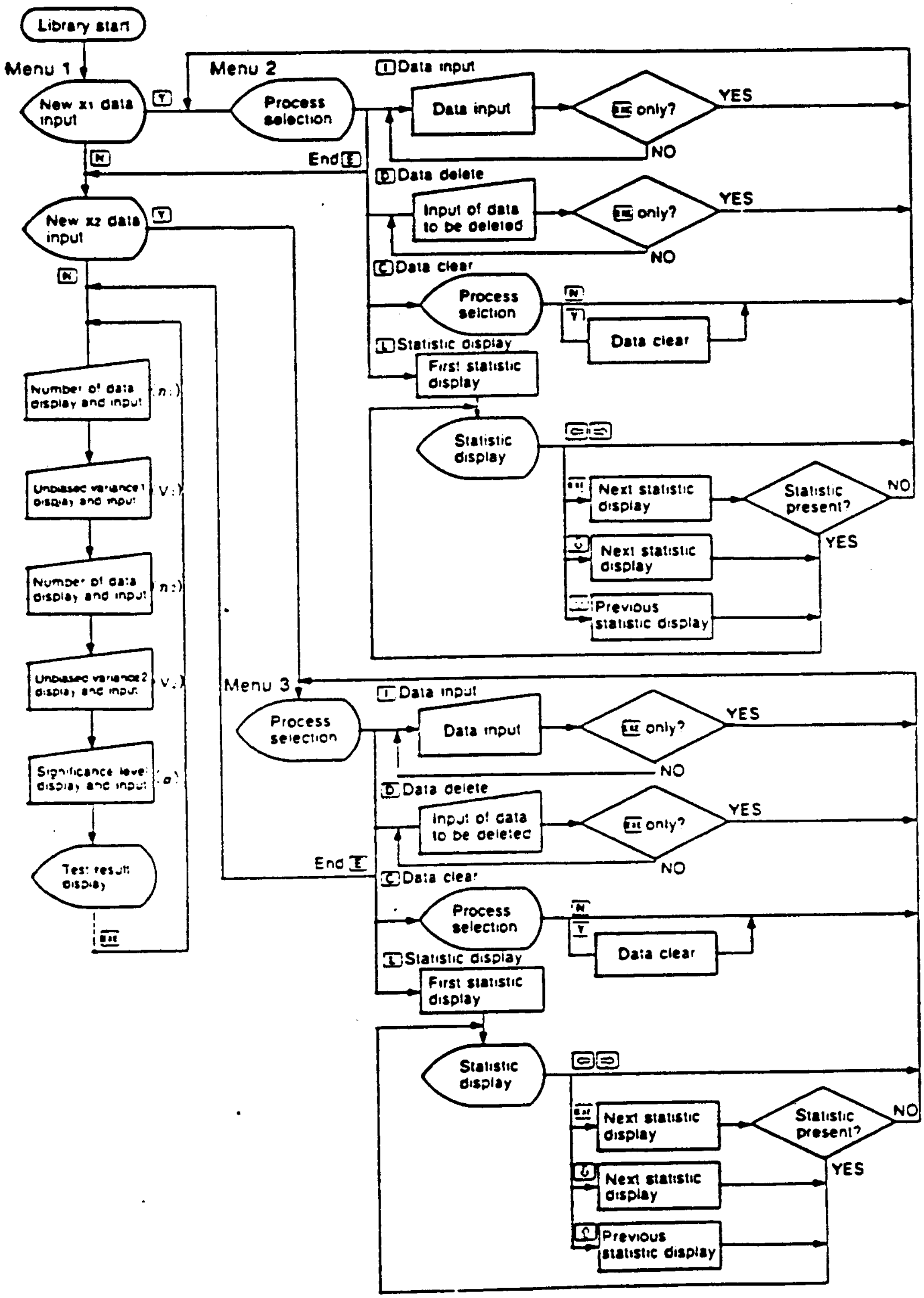
'he following data represent the number of customers at a restaurant before and after recent emodeling. Use the data to determine whether or not the number of customers has been tabilized by the renovation with a significance level of 5%.

		1	2	3	4	5
NUMBER OF CUSTOMERS	BEFORE REMODELING	114	120	78	151	63
	AFTER REMODELING	127	120	138	141	—

	Test $H_0: \sigma_1^2 = \sigma_2^2$ $H_1: \sigma_1^2 > \sigma_2^2$ Input new data (Y/N) ?_	
[Y]	Input data (x1) >Input.Delete.Clear.List.End ?_	(Select new data input.)
[C]	Input data (x1) clear data (Y/N) ?	(Select data clear.)
[Y]	Input data (x1) >Input.Delete.Clear.List.End ?_	(Data cleared.)
[I]	Input data (x1) [EXE]:menu x1?_	(Select data input.)
114 [EXE]	Input data (x1) [EXE]:menu x1?_	(Enter first data item for BEFORE REMODELING.)
120 [EXE] 78 [EXE] 151 [EXE] 63 [EXE]		
	Input data (x1) [EXE]:menu x1?_	(Enter remaining data items.)
[EXE]	Input data (x1) >Input.Delete.Clear.List.End ?_	(Return to menu.)
[E]	Test $H_0: \sigma_1^2 = \sigma_2^2$ $H_1: \sigma_1^2 > \sigma_2^2$ Input new data x2 (Y/N) ?_	(Select End.)
[Y]	Input data (x2) >Input.Delete.Clear.List.End ?_	(Select new data input.)
[C]	Input data (x2) clear data (Y/N) ?	(Select data clear.)
[Y]	Input data (x2) >Input.Delete.Clear.List.End ?_	(Data cleared.)
[I]	Input data (x2) [EXE]:menu x2?_	(Select data input.)
127 [EXE]	Input data (x2) [EXE]:menu x2?_	(Enter first data item for AFTER REMODELING.)
120 [EXE] 138 [EXE] 141 [EXE]		
	Input data (x2) [EXE]:menu x2?_	(Enter remaining data items.)
[EXE]	Input data (x2) >Input.Delete.Clear.List.End ?_	(Return to menu.)
[E]	Test $H_0: \sigma_1^2 = \sigma_2^2$ $H_1: \sigma_1^2 > \sigma_2^2$ n1= 5 ?	(Select End)
[EXE]	Test $H_0: \sigma_1^2 = \sigma_2^2$ $H_1: \sigma_1^2 > \sigma_2^2$ V1= 1228.7 ?_	(Press [EXE] after checking number of BEFORE REMODELING data.)
[EXE]	Test $H_0: \sigma_1^2 = \sigma_2^2$ $H_1: \sigma_1^2 > \sigma_2^2$ n2= 4 ?_	(Press [EXE] after checking unbiased variance of BEFORE REMODELING data.)
[EXE]	Test $H_0: \sigma_1^2 = \sigma_2^2$ $H_1: \sigma_1^2 > \sigma_2^2$ V2= 95 ?_	(Press [EXE] after checking number of AFTER REMODELING data.)
[EXE]	Significance level α(%) α= 5 ?_	(Press [EXE] after checking unbiased variance of AFTER REMODELING data.)
[EXE]	Test $H_0: \sigma_1^2 = \sigma_2^2$ $H_1: \sigma_1^2 > \sigma_2^2$	(Enter significance level. 5% is already set, so simply press [EXE].)
	Test $H_0: \sigma_1^2 = \sigma_2^2$ $H_1: \sigma_1^2 > \sigma_2^2$ 12.93 > 9.117 : ReJect	(Display test result.)
[EXE]	Test $H_0: \sigma_1^2 = \sigma_2^2$ $H_1: \sigma_1^2 > \sigma_2^2$ n1= 5 ?_	

Here, it is determined that the number of customers has not been affected by the remodeling.

RIANCE RATIO TEST FLOWCHART (RIGHT SIDED)



6742 VARIANCE RATIO TEST (LEFT SIDED)

Performs hypotheses testing of σ_1^2 and σ_2^2 in two normal distributions N (μ_1, σ_1^2; where μ_1: unknown, σ_1^2 : unknown) and N (μ_2, σ_2^2; where μ_2 : unknown, σ_2^2 : unknown).

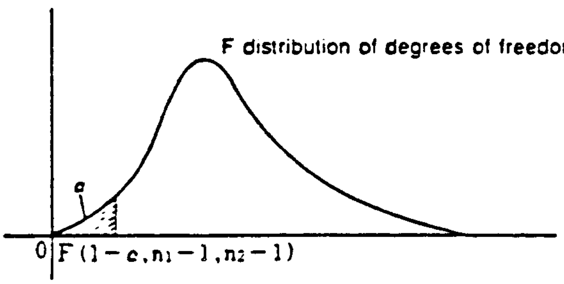
CALCULATIONS

An n_1-size sample ($x_{11}, x_{12} \cdots x_{1n_1}$) is taken from normal distribution N (μ_1, σ_1^2) and an n_2 sample ($x_{21}, x_{22} \cdots x_{2n_2}$) from normal distribution N (μ_2, σ_2^2). At this time, the critical region is established on the right side of the F distribution in accordance with the F distribution of the degrees of freedom (n_1-1, n_2-1) as shown in the illustration when:

Hypothesis to be tested (Null hypothesis) $H_0 : \sigma_1^2 = \sigma_2^2$
Alternative hypothesis $H_1 : \sigma_1^2 < \sigma_2^2$

The test is performed using

$$\frac{V_1}{V_2} < F(1-\alpha, n_1-1, n_2-1) \text{ or } \frac{V_2}{V_1} > F(\alpha, n_2-1, n_1-1)$$



μ_1 : population mean 1
μ_2 : population mean 2
σ_1^2 : population variance 1
σ_2^2 : population variance 2
V_1 : unbiased variance 1
V_2 : unbiased variance 2
α : significance level

$$V = \frac{\sum (x-\bar{x})^2}{n-1}$$

OPERATION

6742 [LB]

```
Test         H0:σ1²=σ2²   H1:σ1²<σ2²
Input new data x1 (Y/N) ?_
```

The display appears as illustrated above once the library is activated. At this time, either Y or N should be pressed to perform the following procedures:

Y : New data input followed by test, additional data input, data edit, statistic check.
N : Test of previously stored data, test by inputting each value.

(1) Y

[Y]

```
Input data (x1)
>Input.Delete.Clear.List.End ?_
```

The menu display illustrated above appears when [Y] is pressed. One of the following character keys is then pressed to perform the corresponding function.

I (Input) : Data input (for input or addition of data).
D (Delete) : Data delete (for deletion of erroneous or unnecessary data).
C (Clear) : Data clear (for deletion of previously stored data. This operation also clears statistics).

L (List) : Statistic display (for display of number of data items, sum, sum of squares, mean, population standard deviation, sample standard deviation).
[⇩] (or [EXE]) scrolls to the following data item, [⇧] to the previous item, and [⇦] or [⇨] exits the statistic display and returns to the menu.

E (End) : Advances to the test display (same as when N is pressed in the first step above).

(2) N

[N]

```
Test       Ho:σ1²=σ2²  H1:σ1²<σ2²
input new data x2 (Y/N) ?
```

The display appears as illustrated above when the [N] key is pressed. Note that this display is almost identical to the initial display which appears immediately after entering library operations. The difference, however, is that the question concerning new data input here is for data items x_{2_1} through x_{2n_2}, while the data input being queried on the original display is for data items x_{1_1} through x_{1n_1}.

(2-1) Y

Same result as that produced by pressing [Y] in step (1) above. Note, however, that the data being entered or corrected here is x_{2_1} through x_{2n_2}.

(2-2) N

[N]

```
Test       Ho:σ1²=σ2²  H1:σ1²<σ2²
n1= 5 ?_
```

(Number of data display)

The display appears as illustrated above when the [N] key is pressed. The value indicated for n shows the number of x_1 (x_{1_1} ~ x_{1n_1}) data currently stored in memory.

- $n_1 = 0$: Test cannot be performed, so this should be corrected to the required data.
- Number of data input (following [Y] above) and value of n differ : Confirm that some data have not been omitted during the input or that two or more data items have been input together for a single entry. In either case, terminate the library operation. Enter the library again and add, delete, or reinput data as required.
- Number of data input (following [Y] above) matches value of n : Press [EXE].
  When [EXE] is pressed, a display similar to that above is produced for x_2 (x_{2_1} ~ x_{2n_2}) data items. After confirmation and/or corrections as described in (2-2), press [EXE] to continue.

EXAMPLE

The following data represent the number of customers at a store before and after a recent change in the main line of products. Use the data to determine whether or not the number of customers has decreased since the change with a significance level of 5%.

		1	2	3	4	5
NUMBER OF CUSTOMERS	PRODUCT A	251	238	261	220	243
	PRODUCT B	241	268	224	230	—

```
Test       Ho:σ1²=σ2²  H1:σ1²<σ2²
input new data x2 (Y/N) ?_
```

Y

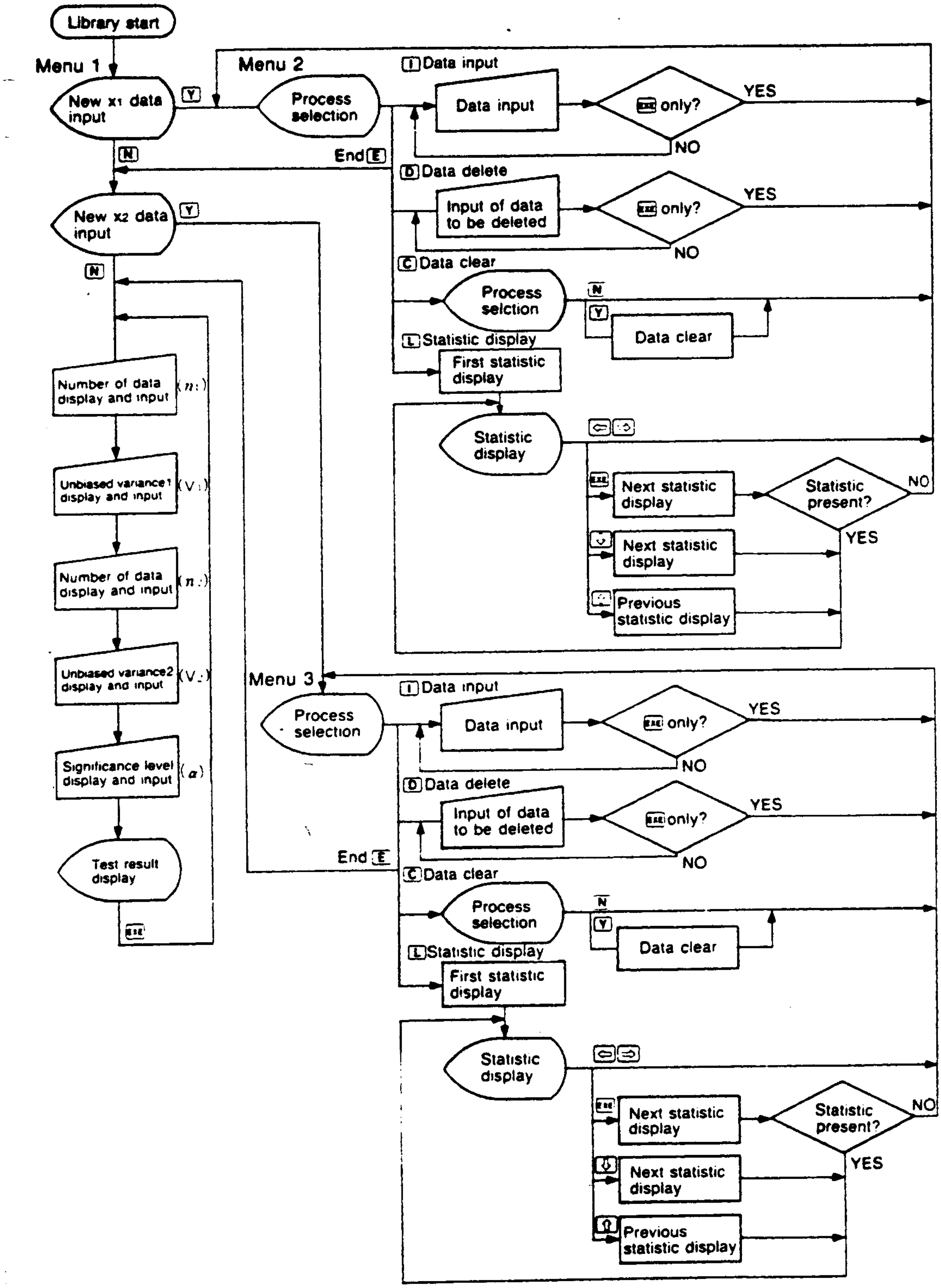
```
Input data (x1)
>Input.Delete.Clear.List.End ?_
```

(Select new data input)

Key operation	Display	Comment
C	Input data (x_1) clear data (Y/N) ?	(Select data clear)
Y	Input data (x_1) >Input.Delete.Clear.List.End ?_	(Data cleared)
I	Input data (x_1) [EXE]:menu x_1?_	(Select data input)
251 [EXE]	Input data (x_1) [EXE]:menu x_1?_	(Enter first data item for PRODUCT A)
238 [EXE] 261 [EXE] 220 [EXE] 243 [EXE]	Input data (x_1) [EXE]:menu x_1?_	(Enter remaining data items)
[EXE]	Input data (x_1) >Input.Delete.Clear.List.End ?_	(Return to menu)
E	Test $H_0:\sigma_1^2=\sigma_2^2$ $H_1:\sigma_1^2<\sigma_2^2$ Input new data x_2 (Y/N) ?_	(Select End)
Y	Input data (x_2) >Input.Delete.Clear.List.End ?_	(Select new data input)
C	Input data (x_2) clear data (Y/N) ?	(Select data clear)
Y	Input data (x_2) >Input.Delete.Clear.List.End ?_	(Data cleared)
I	Input data (x_2) [EXE]:menu x_2?_	(Select data input)
241 [EXE]	Input data (x_2) [EXE]:menu x_2?_	(Enter first data item for PRODUCT B)
268 [EXE] 224 [EXE] 230 [EXE]	Input data (x_2) [EXE]:menu x_2?_	(Enter remaining data items)
[EXE]	Input data (x_2) >Input.Delete.Clear.List.End ?_	(Return to menu)
E	Test $H_0:\sigma_1^2=\sigma_2^2$ $H_1:\sigma_1^2<\sigma_2^2$ n_1= 5	(Select End)
[EXE]	Test $H_0:\sigma_1^2=\sigma_2^2$ $H_1:\sigma_1^2<\sigma_2^2$ V_1= 235.3 ?_	(Press [EXE] after checking number of PRODUCT A data)
[EXE]	Test $H_0:\sigma_1^2=\sigma_2^2$ $H_1:\sigma_1^2<\sigma_2^2$ n_2= 4 ?_	(Press [EXE] after checking unbiased variance of PRODUCT A data)
[EXE]	Test $H_0:\sigma_1^2=\sigma_2^2$ $H_1:\sigma_1^2<\sigma_2^2$ V_2= 379.5833333 ?_	(Press [EXE] after checking number of PRODUCT B data)
[EXE]	Significance level α[%] α= 5 ?_	(Press [EXE] after checking unbiased variance of PRODUCT B data)
[EXE]	Test $H_0:\sigma_1^2=\sigma_2^2$ $H_1:\sigma_1^2<\sigma_2^2$	(Enter significance level. 5% is already set, so simply press [EXE])
	Test $H_0:\sigma_1^2=\sigma_2^2$ $H_1:\sigma_1^2<\sigma_2^2$ 0.6199 ≧ 0.1517 : Accept	(Display test result)
[EXE]	Test $H_0:\sigma_1^2=\sigma_2^2$ $H_1:\sigma_1^2<\sigma_2^2$ n_1= 5 ?_	

Here, it is determined that the number of customers has remained the same since the product change.

VARIANCE RATIO TEST FLOWCHART (LEFT SIDED)



6750 MEAN DIFFERENCE TEST (TWO-SIDED)

Performs hypotheses testing of μ_1 and μ_2 in two normal distributions N (μ_1, σ^2; where μ_1 : unknown, σ^2 : unknown) and N (μ_2, σ^2 ; where μ_2 : unknown, σ^2 : unknown).

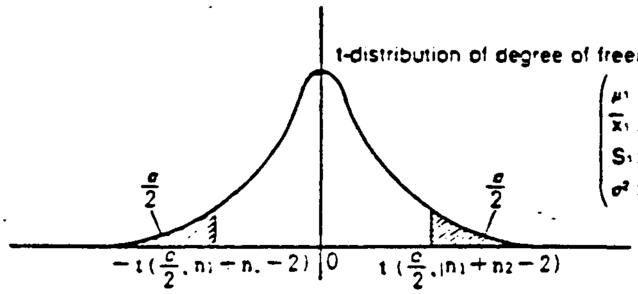
CALCULATIONS

An n_1-size sample (x_{11}, $x_{12} \cdots x_{1n_1}$) is taken from normal distribution N (μ_1, σ^2) and an n_2 sample (x_{21}, $x_{22} \cdots x_{2n_2}$) from normal distribution N (μ_2, σ^2). At this time, critical regions are established on both sides of the t-distribution in accordance with the t-distribution of the degree of freedom ($n_1 + n_2 - 2$) as shown in the illustration when:

Hypothesis to be tested (Null hypothesis) $H_0 : \mu_1 = \mu_2$
Alternative hypothesis $H_1 : \mu_1 \neq \mu_2$

The test is performed using

$$\frac{|\bar{x}_1 - \bar{x}_2|}{\sqrt{\left(\frac{1}{n_1} + \frac{1}{n_2}\right)\left(\frac{S_1 + S_2}{n_1 + n_2 - 2}\right)}} > t\left(\frac{\alpha}{2}, n_1 + n_2 - 2\right)$$



μ_1 : population mean 1 μ_2 : population mean 2
$\bar{x}_1$: sample mean 1 $\bar{x}_2$: sample mean 2
S_1 : sum of squares 1 S_2 : sum of squares 2
σ^2 : population variance α : significance level

$$S = \Sigma(x - \bar{x})^2$$

OPERATION

6750 [LB]

```
Test      H0 : μ1 = μ2   H1 : μ1 ≠ μ2
input new data x1  (Y/N) ?
```

The display appears as illustrated above once the library is activated. At this time, either Y or N should be pressed to perform the following procedures:

Y : New data input followed by test, additional data input, data edit, statistic check.
N : Test of previously stored data, test by inputting each value.

(1) Y

[Y]

```
Input data (x1)
>Input.Delete.Clear.List.End ?_
```

The menu display illustrated above appears when [Y] is pressed. One of the following character keys is then pressed to perform the corresponding function.

I (Input) : Data input (for input or addition of data).
D (Delete) : Data delete (for deletion of erroneous or unnecessary data).
C (Clear) : Data clear (for deletion of previously stored data. This operation also clears statistics).

L (List) : Statistic display (for display of number of data items, sum, sum of squares, mean, population standard deviation, sample standard deviation).
[↓] (or [EXE]) scrolls to the following data item, [↑] to the previous item, and [⇐] or [⇒] exits the statistic display and returns to the menu.

E (End) : Advances to the test display (same as when N is pressed in the first step above).

(2) N

[N]

```
Test        Ho:μ1=μ2   H1:μ1≠μ2
input new data x2 (Y/N) ?
```

The display appears as illustrated above when the [N] key is pressed. Note that this display is almost identical to the initial display which appears immediately after entering library operations. The difference, however, is that the question concerning new data input here is for data items x_{21} through x_{2n_2}, while the data input being queried on the original display is for data items x_{11} through x_{1n_1}.

(2-1) Y

Same result as that produced by pressing [Y] in step (1) above. Note, however, that the data being entered or corrected here is x_{21} through x_{2n_2}.

(2-2) N

[N]

```
Test        Ho:μ1=μ2   H1:μ1≠μ2
n1= 5 ?_
```

(Number of data display)

The display appears as illustrated above when the [N] key is pressed. The value indicated for n shows the number of x_1 (x_{11} ~ x_{1n_1}) data currently stored in memory.

- $n_1 = 0$: Test cannot be performed, so this should be corrected to the required data.
- Number of data input (following [Y] above) and value of n differ : Confirm that some data have not been omitted during the input or that two or more data items have been input together for a single entry. In either case, terminate the library operation. Enter the library again and add, delete, or reinput data as required.
- Number of data input (following [Y] above) matches value of n : Press [EXE].
  When [EXE] is pressed, a display similar to that above is produced for x_2 (x_{21} ~ x_{2n_2}) data items. After confirmation and/or corrections as described in (2-2), press [EXE] to continue.

EXAMPLE

The following data represent the results of durability tests on ten products, five each from two different factories. Use the data to determine whether or not the quality of the products manufactured at the factories differ with a significance level of 5%.

		1	2	3	4	5
DURABILITY (HOURS)	FACTORY A	850	847	855	843	852
	FACTORY B	853	844	850	854	844

```
Test        Ho:μ1=μ2   H1:μ1≠μ2
input new data x1 (Y/N) ?_
```

Y

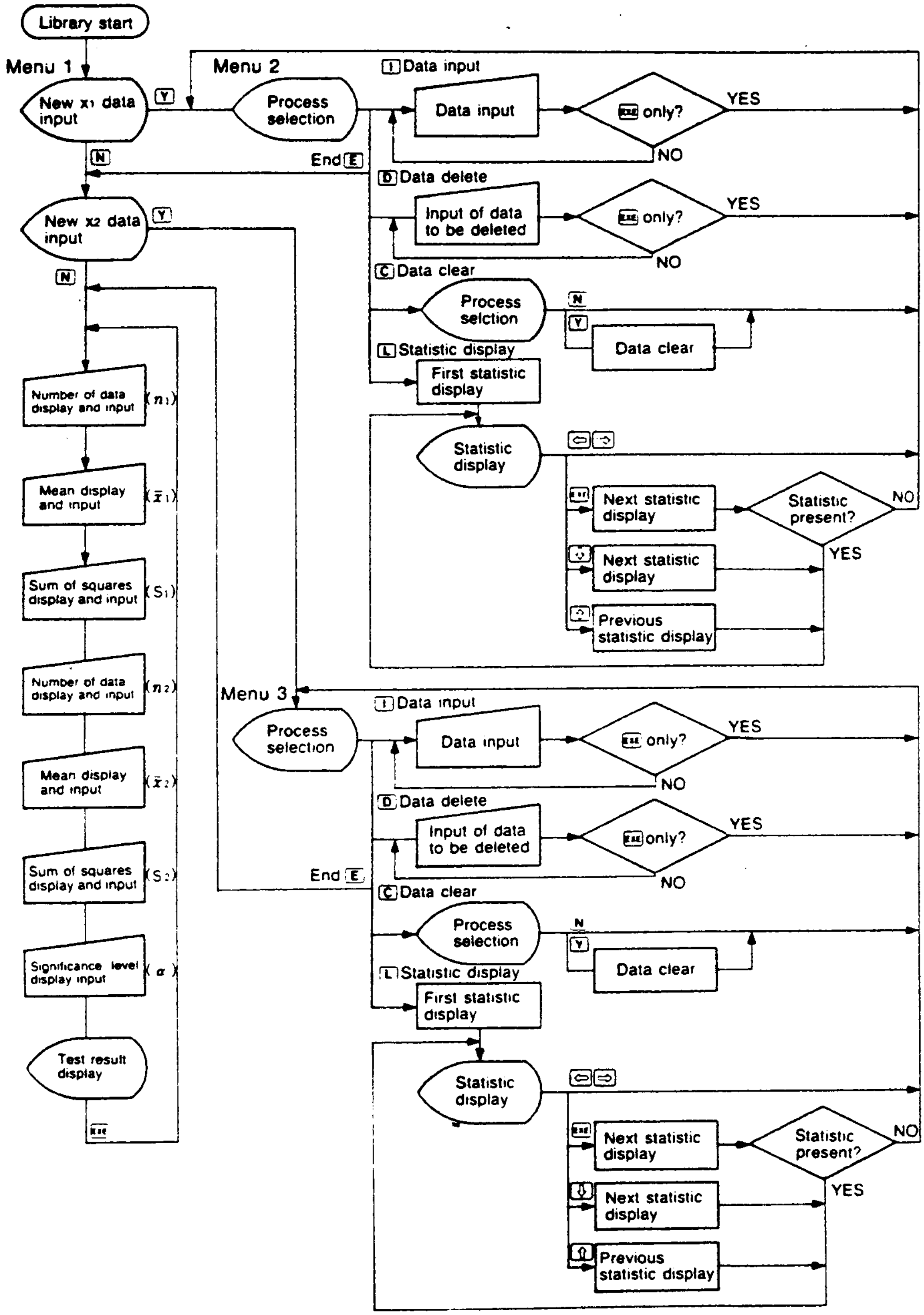
```
Input data (x1)
>Input.Delete.Clear.List.End ?_
```

(Select new data input)

C	Input data (x1) clear data (Y/N) ?	(Select data clear)
Y	Input data (x1) >Input.Delete.Clear.List.End ?_	(Data cleared)
I	Input data (x1) [EXE]:menu x1?_	(Select data input)
850 [EXE]	Input data (x1) [EXE]:menu x1?_	(Enter first data item for FACTORY A)
847 [EXE] 855 [EXE] 843 [EXE] 852 [EXE]	Input data (x1) [EXE]:menu x1?_	(Enter remaining data items)
[EXE]	Input data (x1) >Input.Delete.Clear.List.End ?_	(Return to menu)
E	Test Ho:μ1=μ2 H1:μ1≠μ2 Input new data x2 (Y/N) ?_	(Select End)
Y	Input data (x2) >Input.Delete.Clear.List.End ?_	(Select new data input)
C	Input data (x2) clear data (Y/N) ?	(Select data clear)
Y	Input data (x2) >Input.Delete.Clear.List.End ?_	(Data cleared)
I	Input data (x2) [EXE]:menu x2?_	(Select data input)
853 [EXE]	Input data (x2) [EXE]:menu x2?_	(Enter first data item for FACTORY B)
844 [EXE] 850 [EXE] 854 [EXE] 844 [EXE]	Input data (x2) [EXE]:menu x2?_	(Enter remaining data items)
[EXE]	Input data (x2) >Input.Delete.Clear.List.End ?_	(Return to menu)
E	Test Ho:μ1=μ2 H1:μ1≠μ2 n1= 5 ?_	(Select End)
[EXE]	Test Ho:μ1=μ2 H1:μ1≠μ2 $\bar{x}_1$= 849.4 ?_	(Press [EXE] after checking number of FACTORY A data)
[EXE]	Test Ho:μ1=μ2 H1:μ1≠μ2 S1= 85.2 ?_	(Press [EXE] after checking data mean of FACTORY A data)
[EXE]	Test Ho:μ1=μ2 H1:μ1≠μ2 n2= 5 ?_	(Press [EXE] after checking sum of squares of FACTORY A data)
[EXE]	Test Ho:μ1=μ2 H1:μ1≠μ2 $\bar{x}_2$= 849 ?_	(Press [EXE] after checking number of FACTORY B data)
[EXE]	Test Ho:μ1=μ2 H1:μ1≠μ2 S2= 92 ?_	(Press [EXE] after checking data mean of FACTORY B data)
[EXE]	Significance level α[%] α= 5 ?_	(Press [EXE] after checking sum of squares of FACTORY B data)
[EXE]	Test Ho:μ1=μ2 H1:μ1≠μ2	(Enter significance level. 5% is already set, so simply press [EXE])
	Test Ho:μ1=μ2 H1:μ1≠μ2 0.1344 ≤ 2.306 : Accept	(Display test result)
[EXE]	Test Ho:μ1=μ2 H1:μ1≠μ2 n1= 5 ?_	

Here, it is determined that the quality of goods manufactures at the two factories is equivalent.

MEAN DIFFERENCE TEST FLOWCHART (TWO-SIDED)



6751 MEAN DIFFERENCE TEST (RIGHT SIDED)

Performs hypotheses testing of μ_1 and μ_2 in two normal distributions N (μ_1, σ^2; where μ_1 : unknown, σ^2: unknown) and N (μ_2, σ^2; where μ_2: unknown σ^2: unknown)

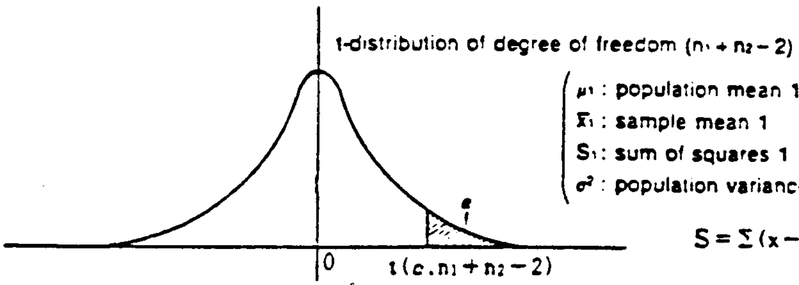
CALCULATIONS

An n_1-size sample (x_{11}, $x_{12} \cdots x_{1n_1}$) is taken from normal distribution N (μ_1, σ^2) and an n_2 sample (x_{21}, $x_{22} \cdots x_{2n_2}$) from normal distribution N (μ_2, σ^2). At this time, the critical region is established on the right of the t-distribution in accordance with the t-distribution of the degree of freedom ($n_1 + n_2 - 2$) as shown in the illustration when:

Hypothesis to be tested (Null hypothesis) $H_0 : \mu_1 = \mu_2$
Alternative hypothesis $H_1 : \mu_1 > \mu_2$

The test is performed using

$$\frac{\overline{x_1} - \overline{x_2}}{\sqrt{\left(\frac{1}{n_1} + \frac{1}{n_2}\right)\left(\frac{S_1 + S_2}{n_1 + n_2 - 2}\right)}} > t\,(\alpha,\ n_1 + n_2 - 2)$$



μ_1 : population mean 1
$\bar{x}_1$: sample mean 1
S_1 : sum of squares 1
σ^2 : population variance
μ_2 : population mean
$\bar{x}_2$: sample mean 2
S_2 : sum of squares 2
α : significance level

$S = \Sigma (x - \bar{x})^2$

OPERATION

6751 [LB]

```
Test        H0:μ1=μ2   H1:μ1>μ2
input new data x1  (Y/N) ?_
```

The display appears as illustrated above once the library is activated. At this time, either Y or N should be pressed to perform the following procedures:

Y: New data input followed by test, additional data input, data edit, statistic check.
N: Test of previously stored data, test by inputting each value.

(1) Y

[Y]

```
Input data (x1)
>Input.Delete.Clear.List.End ?_
```

The menu display illustrated above appears when [Y] is pressed. One of the following character keys is then pressed to perform the corresponding function.

I (Input) : Data input (for input or addition of data).
D (Delete) : Data delete (for deletion of erroneous or unnecessary data).
C (Clear) : Data clear (for deletion of previously stored data. This operation also clears statistics).
L (List) : Statistic display (for display of number of data items, sum, sum of squares, mean, population standard deviation, sample standard deviation).
[⇩] (or [EXE]) scrolls to the following data item, [⇧] to the previous item, and [⇦] or [⇨] exits the statistic display and returns to the menu.
E (End) : Advances to the test display (same as when N is pressed in the first step above).

(2) N

[N]

```
Test        Ho:μ1=μ2   H1:μ1>μ2
input new data x2 (Y/N) ?_
```

The display appears as illustrated above when the [N] key is pressed. Note that this display is almost identical to the initial display which appears immediately after entering library operations. The difference, however, is that the question concerning new data input here is for data items x_{21} through x_{2n_2}, while the data input being queried on the original display is for data items x_{11} through x_{1n_1}.

(2-1) Y

Same result as that produced by pressing [Y] in step (1) above. Note, however, that the data being entered or corrected here is x_{21} through x_{2n_2}.

(2-2) N

[N]

```
Test        Ho:μ1=μ2   H1:μ1>μ2
n1= 5 ?_
```

(Number of data display)

The display appears as illustrated above when the [N] key is pressed. The value indicated for n shows the number of x_1 (x_{11} ~ x_{1n_1}) data currently stored in memory.

- $n_1 = 0$: Test cannot be performed, so this should be corrected to the required data.
- Number of data input (following [Y] above) and value of n differ : Confirm that some data have not been omitted during the input or that two or more data items have been input together for a single entry. In either case, terminate the library operation. Enter the library again and add, delete, or reinput data as required.
- Number of data input (following [Y] above) matches value of n : Press [EXE].
  When [EXE] is pressed, a display similar to that above is produced for x_2 (x_{21} ~ x_{2n_2}) data items. After confirmation and/or corrections as described in (2-2), press [EXE] to continue.

EXAMPLE

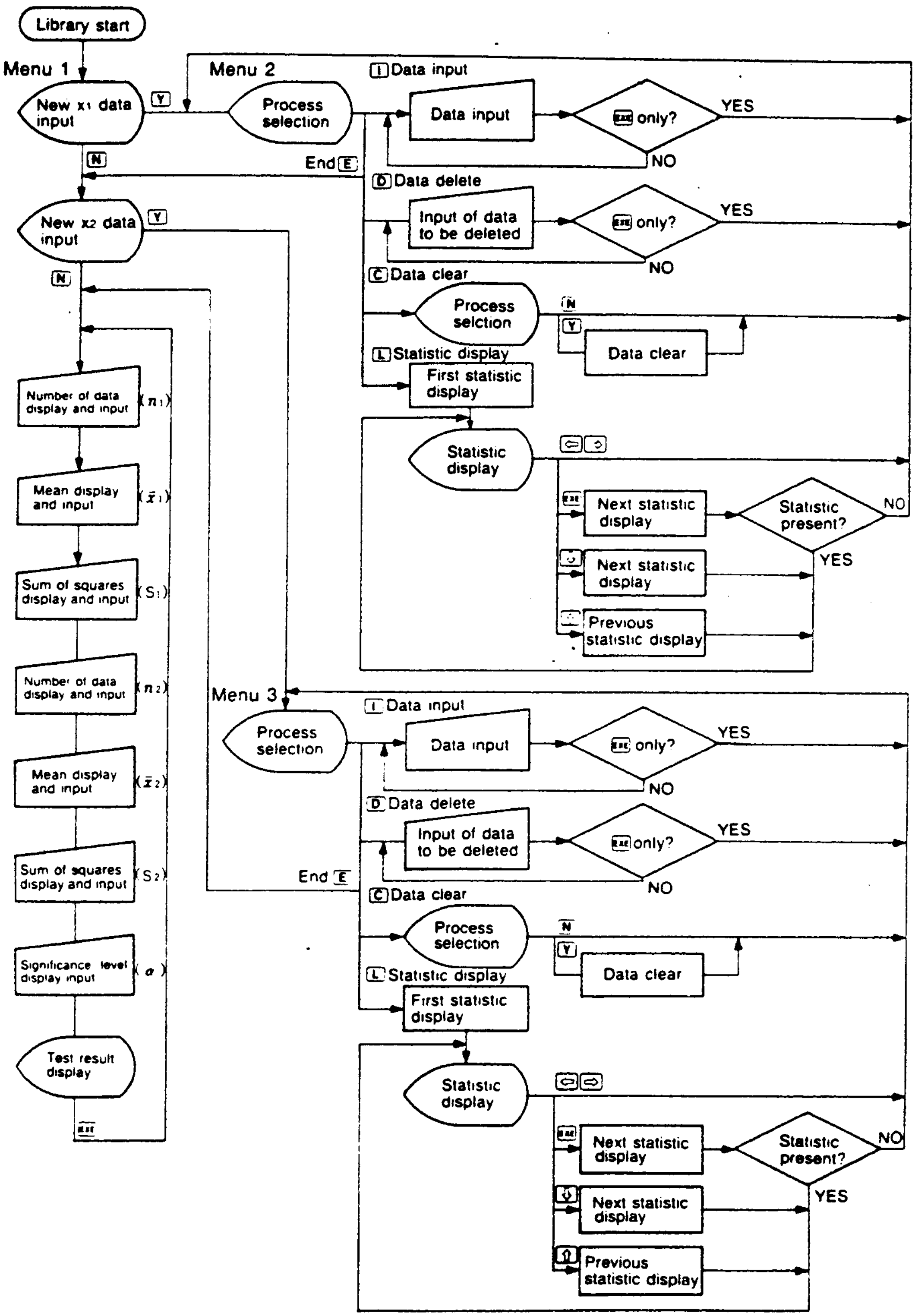
The following data represent the results of tests on light bulb A which is more expensive than light bulb B, but claims a longer service life. Use the data to determine whether or not there is a difference in the service lives of light bulbs A and B with a significance level of 5%.

		1	2	3	4	5
TIME	A	890	880	920	870	900
	B	850	840	870	855	860

Key	Display	Comment
	Test Ho:μ1=μ2 H1:μ1>μ2 Input new data x1 (Y/N) ?_	
Y	Input data (x1) >Input.Delete.Clear.List.End ?_	(Select new data input)
C	Input data (x1) clear data (Y/N) ?	(Select data clear)
Y	Input data (x1) >Input.Delete.Clear.List.End ?_	(Data cleared)
I	Input data (x1) [EXE]:menu x1?_	(Select data input)
890 [EXE]	Input data (x1) [EXE]:menu x1?_	(Enter first data item for LIGHT BULB A)
880 [EXE] 920 [EXE] 870 [EXE] 900 [EXE]	Input data (x1) [EXE]:menu x1?_	(Enter remaining data items)
[EXE]	Input data (x1) >Input.Delete.Clear.List.End ?_	(Return to menu)
E	Test Ho:μ1=μ2 H1:μ1>μ2 Input.new data x2 (Y/N) ?	(Select End)
Y	Input data (x2) >Input.Delete.Clear.List.End ?_	(Select new data input)
C	Input data (x2) clear data (Y/N) ?	(Select data clear)
Y	Input data (x2) >Input.Delete.Clear.List.End ?_	(Data cleared)
I	Input data (x2) [EXE]:menu x2?_	(Select data input)
850 [EXE]	Input data (x2) [EXE]:menu x2?_	(Enter first data item fr LIGHT BULB B)
840 [EXE] 870 [EXE] 855 [EXE] 860 [EXE]	Input data (x2). [EXE]:menu x2?_	(Enter remaining da items)
[EXE]	Input data (x2) >Input.Delete.Clear.List.End ?_	(Return to menu)
E	Test Ho:μ1=μ2 H1:μ1>μ2 n1= 5 ?_	(Select End)
[EXE]	Test Ho:μ1=μ2 H1:μ1>μ2 $\bar{x}$1= 892 ?_	(Press [EXE] after checkir number of LIGHT BUL A data)
[EXE]	Test Ho:μ1=μ2 H1:μ1>μ2 S1= 1480 ?_	(Press [EXE] after checkin data mean of LIGHT BULB A data)
[EXE]	Test Ho:μ1=μ2 H1:μ1>μ2 n2= 5 ?_	(Press [EXE] after checking sum of squares of LIGHT BULB A data)
[EXE]	Test Ho:μ1=μ2 H1:μ1>μ2 $\bar{x}$2= 855 ?_	(Press [EXE] after checking number of LIGHT BULB B data)
[EXE]	Test Ho:μ1=μ2 H1:μ1>μ2 S2= 500 ?_	(Press [EXE] after checking data mean of LIGHT BULB B data)
[EXE]	Significance level α(%) α= 5 ?_	(Press [EXE] after checking sum of squares of LIGHT BULB B data)
[EXE]	Test Ho:μ1=μ2 H1:μ1>μ2	(Enter significance level. 5% is already set, so simply press [EXE])
	Test Ho:μ1=μ2 H1:μ1>μ2 3.718 > 1.86 : ReJect	(Display test result)
[EXE]	Test Ho:μ1=μ2 H1:μ1>μ2 n1= 5 ?_	

Here, it is determined that the service life of LIGHT BULB A is longer than that of LIGHT BULB B.

MEAN DIFFERENCE TEST FLOWCHART (RIGHT SIDED)



6752 MEAN DIFFERENCE TEST (LEFT SIDED)

Performs hypotheses testing of μ_1 and μ_2 in two normal distributions N (μ_1, σ^2; where μ_1: unknown, σ^2: unknown) and N (μ_2, σ^2; where μ_2: unknown, σ^2: unknown)

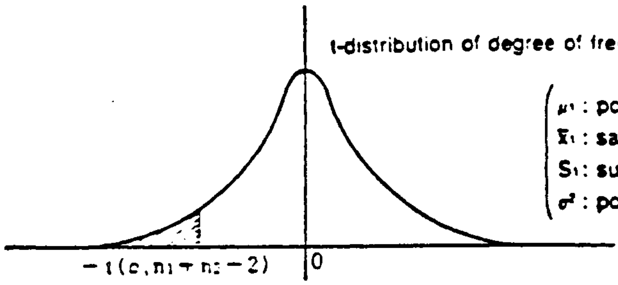
CALCULATIONS

An n_1-size sample ($x_{11}, x_{12} \cdots x_{1n_1}$) is taken from normal distribution N (μ_1, σ^2) and an n_2 sample ($x_{21}, x_{22} \cdots x_{2n_2}$) from normal distribution N (μ_2, σ^2). At this time, the critical region is established on the left of the t-distribution in accordance with the t-distribution of the degree of freedom ($n_1 + n_2 - 2$) as shown in the illustration when:

Hypothesis to be tested (Null hypothesis) $H_0 : \mu_1 = \mu_2$
Alternative hypothesis $H_1 : \mu_1 < \mu_2$

The test is performed using

$$\frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\left(\frac{1}{n_1} + \frac{1}{n_2}\right)\left(\frac{S_1 + S_2}{n_1 + n_2 - 2}\right)}} < -t\,(\alpha \cdot n_1 + n_2 - 2)$$



μ_1 : population mean 1 — μ_2 : population mean
$\bar{x}_1$: sample mean 1 — $\bar{x}_2$: sample mean 2
S_1: sum of squares 1 — S_2: sum of squares 2
σ^2 : population variance — α : significance level

$$S = \Sigma (x - \bar{x})^2$$

OPERATION

6752 [LIB]

```
Test       H0 : μ1 = μ2   H1 : μ1 < μ2
Input new data x1 (Y/N) ?_
```

The display appears as illustrated above once the library is activated. At this time, either Y or N should be pressed to perform the following procedures:

Y : New data input followed by test, additional data input, data edit, statistic check.
N : Test of previously stored data, test by inputting each value.

(1) Y

[Y]

```
Input data (x1)
>Input.Delete.Clear.List.End ?_
```

The menu display illustrated above appears when [Y] is pressed. One of the following character keys is then pressed to perform the corresponding function.

I (Input) : Data input (for input or addition of data).
D (Delete) : Data delete (for deletion of erroneous or unnecessary data).
C (Clear) : Data clear (for deletion of previously stored data. This operation also clears statistics).
L (List) : Statistic display (for display of number of data items, sum, sum of squares, mean, population standard deviation, sample standard deviation).
[⇩] (or [EXE]) scrolls to the following data item, [⇧] to the previous item, and [⇦] or [⇨] exits the statistic display and returns to the menu.
E (End) : Advances to the test display (same as when N is pressed in the first step above).

(2) N

[N]

```
Test        Ho:μ1=μ2   H1:μ1<μ2
Input new data x2  (Y/N) ?_
```

The display appears as illustrated above when the [N] key is pressed. Note that this display is almost identical to the initial display which appears immediately after entering library operations. The difference, however, is that the question concerning new data input here is for data items x_{21} through x_{2n_2}, while the data input being queried on the original display is for data items x_{11} through x_{1n_1}.

(2-1) Y

Same result as that produced by pressing [Y] in step (1) above. Note, however, that the data being entered or corrected here is x_{21} through x_{2n_2}.

(2-2) N

[N]

```
Test        Ho:μ1=μ2   H1:μ1<μ2
n1= 5 ?_
```

(Number of data display)

The display appears as illustrated above when the [N] key is pressed. The value indicated for n shows the number of x_1 ($x_{11} \sim x_{1n_1}$) data currently stored in memory.

- $n_1 = 0$: Test cannot be performed, so this should be corrected to the required data.
- Number of data input (following [Y] above) and value of n differ : Confirm that some data have not been omitted during the input or that two or more data items have been input together for a single entry. In either case, terminate the library operation. Enter the library again and add, delete, or reinput data as required.
- Number of data input (following [Y] above) matches value of n : Press [EXE].
  When [EXE] is pressed, a display similar to that above is produced for x_2 ($x_{21} - x_{2n_2}$) data items. After confirmation and/or corrections as described in (2-2), press [EXE] to continue.

EXAMPLE

The following data represent the results of tests on concrete samples. SAMPLE A is not reinforced, while SAMPLE B is reinforced. Use the data to determine whether or not the reinforcement actually makes the concrete stronger with a significance level of 1%.

		1	2	3	4	5
STRENGTH (kg)	SAMPLE A	18	20	17	19	18
	SAMPLE B	25	24	22	26	24

Key operation	Display	Comment
	Test Ho:$\mu_1=\mu_2$ H1:$\mu_1<\mu_2$ input new data x1 (Y/N) ?_	
Y	Input data (x1) >Input.Delete.Clear.List.End ?_	(Select new data input)
C	Input data (x1) clear data (Y/N) ?	(Select data clear)
Y	Input data (x1) >Input.Delete.Clear.List.End ?_	(Data cleared)
I	Input data (x1) [EXE]:menu x1?_	(Select data input)
18 EXE	Input data (x1) [EXE]:menu x1?_	(Enter first data item for SAMPLE A)
20 EXE 17 EXE 19 EXE 18 EXE	Input data (x1) [EXE]:menu x1?_	(Enter remaining data items)
EXE	Input data (x1) >Input.Delete.Clear.List.End ?_	(Return to menu)
E	Test Ho:$\mu_1=\mu_2$ H1:$\mu_1<\mu_2$ input new data x2 (Y/N) ?_	(Select End)
Y	Input data (x2) >Input.Delete.Clear.List.End ?_	(Select new data input)
C	Input data (x2) clear data (Y/N) ?	(Select data clear)
Y	Input data (x2) >Input.Delete.Clear.List.End ?_	(Data cleared)
I	Input data (x2) [EXE]:menu x2?_	(Select data input)
25 EXE	Input data (x2) [EXE]:menu x2?_	(Enter first data item for SAMPLE B)
24 EXE 22 EXE 26 EXE 24 EXE	Input data (x2) [EXE]:menu x2?_	(Enter remaining data items)
EXE	Input data (x2) >Input.Delete.Clear.List.End ?_	(Return to menu)
E	Test Ho:$\mu_1=\mu_2$ H1:$\mu_1<\mu_2$ n1= 5 ?_	(Select End)
EXE	Test Ho:$\mu_1=\mu_2$ H1:$\mu_1<\mu_2$ $\bar{x}_1$= 18.4?_	(Press EXE after checking number of SAMPLE A data)
EXE	Test Ho:$\mu_1=\mu_2$ H1:$\mu_1<\mu_2$ S1= 5.2?_	(Press EXE after checking data mean of SAMPLE A data)
EXE	Test Ho:$\mu_1=\mu_2$ H1:$\mu_1<\mu_2$ n2= 5 ?_	(Press EXE after checking sum of squares of SAMPLE A data)
EXE	Test Ho:$\mu_1=\mu_2$ H1:$\mu_1<\mu_2$ $\bar{x}_2$= 24.2 ?_	(Press EXE after checking number of SAMPLE B data)
EXE	Test Ho:$\mu_1=\mu_2$ H1:$\mu_1<\mu_2$ S2= 8.8?_	(Press EXE after checking data mean of SAMPLE B data)
EXE	Significance level α[%] α= 5 ?	(Press EXE after checking sum of squares of SAMPLE B data)
1 EXE	Test Ho:$\mu_1=\mu_2$ H1:$\mu_1<\mu_2$	(Enter significance level)
	Test Ho:$\mu_1=\mu_2$ H1:$\mu_1<\mu_2$ -6.932 <-2.896 : ReJect	(Display test result)
EXE	Test Ho:$\mu_1=\mu_2$ H1:$\mu_1<\mu_2$ n1= 5 ?_	

Here, it is determined that the strength SAMPLE B is greater than that of SAMPLE A.

6761 RATIO TEST (RIGHT SIDED)

Performs test of hypothesis of population p ratio in binomial distribution B (1, p).

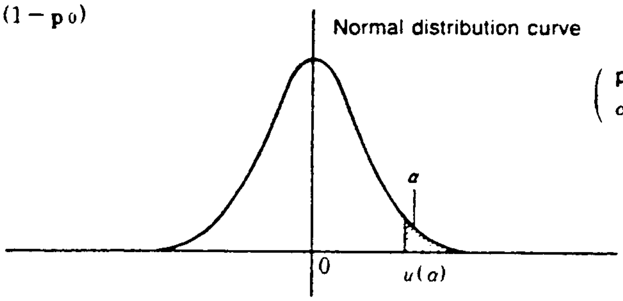
CALCULATIONS

An n-size sample (x_1, $x_2 \cdots x_n$) is taken from binomial distribution B (1, p). At this time, the critical region is established on the right side of the normal distribution in accordance with the approximate standard normal distribution N (0, 1^2) as shown in the illustration when:

Hypothesis to be tested (Null hypothesis) $H_0 : p = p_0$
Alternative hypothesis $H_1 : p > p_0$

The test is performed using

$$\frac{\Sigma x - np_0}{\sqrt{np_0(1-p_0)}} > u(\alpha)$$



(p_0 : population ratio
α : significance level)

OPERATION

6761 LIB	Test Ho : p=po H1 : p>po po= 0 ?_	

EXAMPLE

A company has sent out 2,000 pieces of direct mail advertising printed in four colors and, as a result, received 80 orders. In the past, direct mail has produced a response of 2.5%. Determine whether or not the 4-color direct mailing was as effective as those in the past with a significance level of 5%.

0.025 EXE	Test Ho : p=po H1 : p>po n= 0 ?	(Enter probability)
2000 EXE	Test Ho : p=po H1 : p>po Σx= 0 ?	(Enter number of data items)
80 EXE	Significance level α[%] α= 5 ?_	(Enter number of responses)
EXE	Test Ho : p=po H1 : p>po	(Enter significance level. 5% is already set, so simply press EXE)
	Test Ho : p=po H1 : p>po 4.297 > 1.645 : Reject	(Display test result)
EXE	Test Ho : p=po H1 : p>po po= 0.025 ?_	(Return to original display)

Here, it is determined that the 4-color mailing was more effective than those in the past.

6762 RATIO TEST (LEFT SIDED)

Performs test of hypothesis of population p ratio in binomial distribution B (1, p).

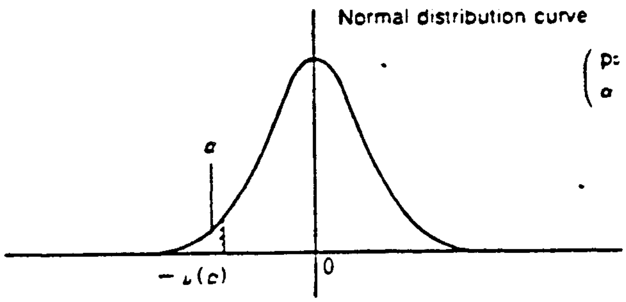
CALCULATIONS

An n-size sample ($x_1, x_2 \cdots x_n$) is taken from binomial distribution B (1, p). At this time, the critical region is established on the left side of the normal distribution in accordance with the approximate standard normal distribution N (0, 1^2) as shown in the illustration when:

Hypothesis to be tested (Null hypothesis) $H_0 : p = p_0$
Alternative hypothesis $H_1 : p < p_0$

The test is performed using

$$\frac{\sum x - np_0}{\sqrt{np_0(1-p_0)}} < -\mu(\alpha)$$



(p_0 : population ratio
α : significance level)

OPERATION

6762 LIB

```
Test      H0:p=p0   H1:p<p0
p0= 0 ?_
```

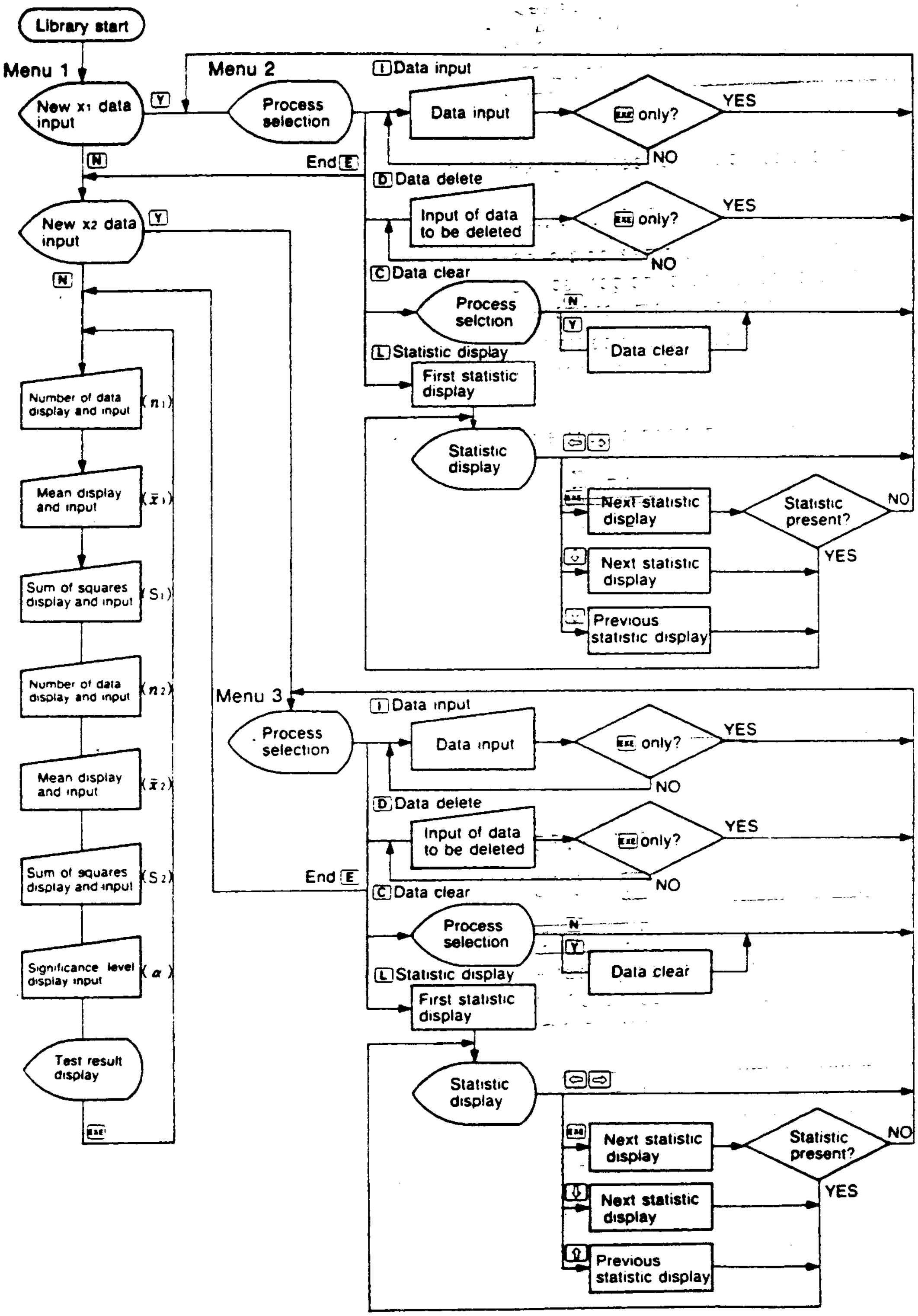
EXAMPLE

A factory had a defect rate of 2.5% of products. After improvements in the process, a total of 18 defective products were detected in 1,000 items. Determine whether or not the improvements have caused the defect rate to decrease with a significance level of 1%.

Key	Display	
0.025 EXE	Test H0:p=p0 H1:p<p0 / n= 0 ?_	(Enter probability)
1000 EXE	Test H0:p=p0 H1:p<p0 / Σx= 0 ?_	(Enter number of data items)
18 EXE	Significance level α(%) / α= 5 ?	(Enter number of defective items)
1 EXE	Test H0:p=p0 H1:p<p0 /	(Enter significance level)
	Test H0:p=p0 H1:p<p0 / -1.418 ≥ 2.326 : Accept	(Display test result)
EXE	Test H0:p=p0 H1:p<p0 / p0= 0.025 ?_	(Return to original display)

Here, it is determined that the improvements have not produced a decrease in the defect rate.

MEAN DIFFERENCE TEST FLOWCHART (LEFT SIDED)



6760 RATIO TEST (TWO-SIDED)

Performs test of hypothesis of population ratio p in binomial distribution B (1, p).

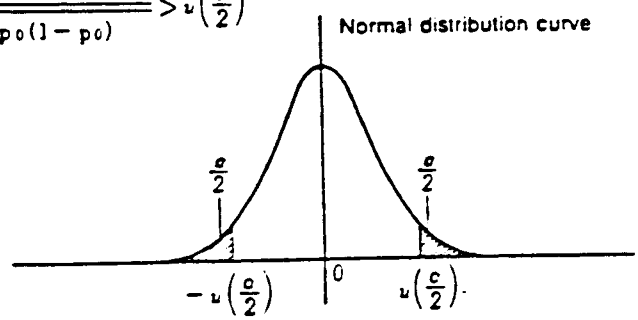
CALCULATIONS

An n-size sample ($x_1, x_2 \cdots x_n$) is taken from binomial distribution B (1, p). At this time, critical regions are established on both sides of the normal distribution in accordance with the approximate standard normal distribution N $(0, 1^2)$ as shown in the illustration when:

Hypothesis to be tested (Null hypothesis) $H_0 : p = p_0$
Alternative hypothesis $H_1 : p \neq p_0$

The test is performed using

$$\frac{|\Sigma x - np_0|}{\sqrt{np_0(1-p_0)}} > u\left(\frac{\alpha}{2}\right)$$



(p_0 : population ratio
α : significance level)

OPERATION

6760 [LB]

```
Test        Ho : p=po   H1 : p≠po
po=  0 ?_
```

EXAMPLE

The crossbreeding of a certain type of bean should result in a ratio of 3:1 of yellow beans to green. An actual sample reveals 310 yellow beans within a total of 400. Determine whether or not this is equivalent to the 3:1 ratio noted above with a significance level of 5%.

(HINT: Test the hypothesis $H: p = \frac{3}{4}$)

Key	Display	Comment
0.75 [EXE]	Test Ho : p=po H1 : p≠po / n= 0 ?_	(Enter probability)
400 [EXE]	Test Ho : p=po H1 : p≠po / Σx= 0 ?	(Enter number of data items)
310 [EXE]	Significance level α(%) / α= 5 ?_	(Enter number of yellow beans)
[EXE]	Test Ho : p=po H1 : p≠po /	(Enter significance level. 5% is already set, so simply press [EXE])
	Test Ho : p=po H1 : p≠po / 1.155 ≤ 1.96 : Accept	(Display test result)
[EXE]	Test Ho : p=po H1 : p≠po / po= 0.75 ?_	(Return to original display)

Here, it is determined that the sample mixture is equivalent to the 3:1 ratio.

6770 RATIO DIFFERENCE TEST (TWO-SIDED)

Performs hypotheses testing of p_1 and p_2 in two binomial distributions B (1, p_1) and B (1, p_2).

CALCULATIONS

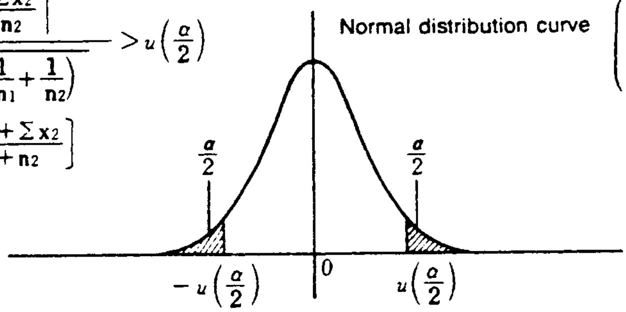
An n_1-size sample (x_{11}, $x_{12} \cdots x_{1n_1}$) is taken from binomial distribution B (1, p_1) and an n_2 sample (x_{21}, $x_{22} \cdots x_{2n_2}$) from binomial distribution B (1, p_2). At this time, critical regions are established on both sides of the normal distribution in accordance with the approximate standard normal distribution N (0, 1^2) as shown in the illustration when:

Hypothesis to be tested (Null hypothesis) $H_0 : p_1 = p_2$
Alternative hypothesis $H_1 : p_1 \neq p_2$

The test is performed using

$$\frac{\left| \frac{\Sigma x_1}{n_1} - \frac{\Sigma x_2}{n_2} \right|}{\sqrt{p(1-p)\left(\frac{1}{n_1}+\frac{1}{n_2}\right)}} > u\left(\frac{\alpha}{2}\right)$$

$$\left[p = \frac{\Sigma x_1 + \Sigma x_2}{n_1 + n_2} \right]$$



(p_1 : sample ratio 1
p_2 : sample ratio 2
α : significance level)

OPERATION

6770 LIB

```
Test      Ho:p1=p2   H1:p1≠p2
n1=  0  ?_
```

EXAMPLE

The following data represent the results of a survey taken for a certain product. Use the data to determine whether or not opinions differ according to gender with a significance level of 5%.

	LIKE	DISLIKE	TOTAL
MALE	260	140	400
FEMALE	180	120	300

```
Test      Ho:p1=p2   H1:p1≠p2
n1=  0  ?_
```

400 EXE

```
Test      Ho:p1=p2   H1:p1≠p2
Σx1=  0  ?_
```

(Enter number of males)

260 EXE

```
Test      Ho:p1=p2   H1:p1≠p2
n2=  0  ?_
```

(Enter number of males answering LIKE)

300 EXE

```
Test      Ho:p1=p2   H1:p1≠p2
Σx2=  0  ?_
```

(Enter number of females)

180 [EXE]	Significance level α(%) α= 5 ?_	(Enter number of females answering LIKE)
[EXE]	Test Ho:p1=p2 H1:p1≠p2	(Enter significance level. 5% is already set, so simply press [EXE])
	Test Ho:p1=p2 H1:p1≠p2 1.355 ≦ 1.96 : Accept	(Display test result)
[EXE]	Test Ho:p1=p2 H1:p1≠p2 n1= 400 ?_	(Return to initial display)

Here, it is determined that there is no difference in the opinions of males and females.

6771 RATIO DIFFERENCE TEST (RIGHT SIDED)

Performs hypotheses testing of p_1 and p_2 in two binomial distributions B (1, p_1) and B (1, p_2).

CALCULATIONS

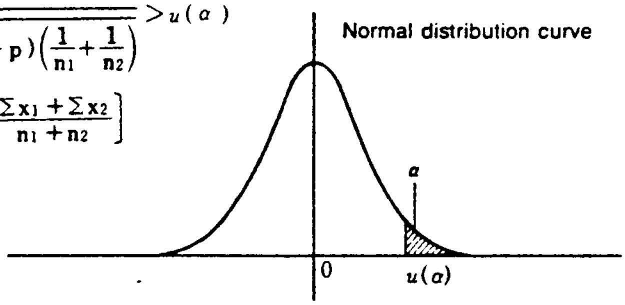
An n_1-size sample ($x_{11}, x_{12} \cdots x_{1n_1}$) is taken from binomial distribution B (1, p_1) and an n_2 sample ($x_{21}, x_{22} \cdots x_{2n_2}$) from binomial distribution B (1, p_2). At this time, a critical region is established on the right side of the normal distribution in accordance with the approximate standard normal distribution N (0, 1^2) as shown in the illustration when:

Hypothesis to be tested (Null hypothesis) $H_0 : p_1 = p_2$
Alternative hypothesis $H_1 : p_1 > p_2$

The test is performed using

$$\frac{\frac{\Sigma x_1}{n_1} - \frac{\Sigma x_2}{n_2}}{\sqrt{p(1-p)\left(\frac{1}{n_1}+\frac{1}{n_2}\right)}} > u(\alpha)$$

$$\left[p = \frac{\Sigma x_1 + \Sigma x_2}{n_1 + n_2} \right]$$



(p_1 : sample ratio 1
p_2 : sample ratio 2
α : significance level)

OPERATION

6771 LIB

```
Test      Ho:p1=p2   H1:p1>p2
n1 = 0 ?_
```

EXAMPLE

The following data represent samples taken of the same product manufactured at two different factories. Use the data to determine whether the defect rate is greater for FACTORY A with a significance level of 5%.

	SAMPLES	DEFECTS
FACTORY A	600	15
FACTORY B	400	5

```
Test      Ho:p1=p2   H1:p1>p2
n1= 0 ?_
```

600 EXE

```
Test      Ho:p1=p2   H1:p1>p2
Σx1= 0 ?_
```

(Enter number of samples from FACTORY A)

15 [EXE]	Test H0:p1=p2 H1:p1>p2 n2= 0 ?_	(Enter number of defects)
400 [EXE]	Test H0:p1=p2 H1:p1>p2 Σx2= 0 ?_	(Enter number of samples from FACTORY B)
5 [EXE]	Significance level α(%) α= 5 ?_	(Enter number of defects)
[EXE]	Test H0:p1=p2 H1:p1>p2	(Enter significance level. 5% is already set, so simply press [EXE])
	Test H0:p1=p2 H1:p1>p2 1.383 ≦ 1.645 : Accept	(Display test result)
[EXE]	Test H0:p1=p2 H1:p1>p2 n1= 600 ?_	(Return to initial display)

Here, it is determined that there is no difference in the defect rate for the two factories.

6772 RATIO DIFFERENCE TEST (LEFT SIDED)

Performs hypotheses testing of p_1 and p_2 in two binomial distributions B (1, p_1) and B (1, p_2).

CALCULATIONS

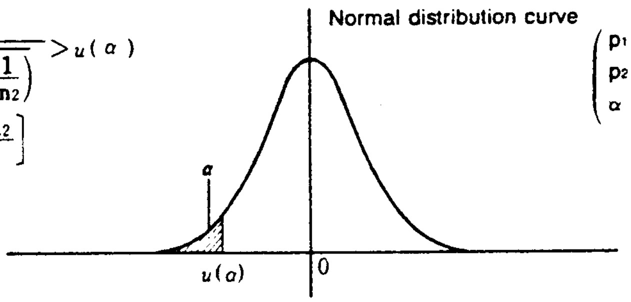
An n_1-size sample (x_{11}, $x_{12} \cdots x_{1n_1}$) is taken from binomial distribution B (1, p_1) and an n_2 sample (x_{21}, $x_{22} \cdots x_{2n_2}$) from binomial distribution B (1, p_2). At this time, a critical region is established on the left side of the normal distribution in accordance with the approximate standard normal distribution N (0, 1^2) as shown in the illustration when:

Hypothesis to be tested (Null hypothesis) $H_0 : p_1 = p_2$
Alternative hypothesis $H_1 : p_1 < p_2$

The test is performed using

$$\frac{\frac{\Sigma x_1}{n_1} - \frac{\Sigma x_2}{n_2}}{\sqrt{p(1-p)\left(\frac{1}{n_1}+\frac{1}{n_2}\right)}} > u(\alpha)$$

$$\left[p = \frac{\Sigma x_1 + \Sigma x_2}{n_1 + n_2}\right]$$



(p_1 : sample ratio 1
p_2 : sample ratio 2
α : significance level)

OPERATION

6772 [LIB]

```
Test       Ho:p1=p2  H1:p1<p2
n1=  0  ?_
```

EXAMPLE

The following data represent the results of a survey taken in two areas concerning recognition of a product. Use the data to determine whether the recognition rate is greater for AREA B with a significance level of 5%.

	KNOW	DON'T KNOW	TOTAL
AREA A	130	90	220
AREA B	160	80	240

```
Test       Ho:p1=p2  H1:p1<p2
n1  =  0  ?_
```

220 [EXE]

```
Test       Ho:p1=p2  H1:p1<p2
Σx1=  0  ?_
```

(Enter number of data from AREA A)

130 EXE	Test Ho:p1=p2 H1:p1<p2 n2= 0 ?_	(Enter number of KNOWs)
240 EXE	Test Ho:p1=p2 H1:p1<p2 Σx2= 0 ?_	(Enter numbe- of dat from AREA B)
160 EXE	Significance level α(%) α= 5 ?_	(Enter number of KNOWs)
EXE	Test Ho:p1=p2 H1:p1<p2	(Enter significance lev- el. 5% is already set, so simply press EXE)
	Test Ho:p1=p2 H1:p1<p2 -1.682 <-1.645 : ReJect	(Display test result)
EXE	Test Ho:p1=p2 H1:p1<p2 n1= 220 ?_	(Return to initial display)

Here, it is determined that the recognition rate in AREA B is greater than that in AREA A.

PART 12

APPENDICES

1 CHARACTER CODE TABLE

HEX	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	16	SPACE 32	0 48	@ 64	P 80	' 96	p 112	Ȧ 128	0 144	SPACE 160	ー 176	タ 192	ミ 208	≥ 224	× 240
1	1	(DEL) 17	! 33	1 49	A 65	Q 81	a 97	q 113	∫ 129	1 145	。 161	ア 177	チ 193	ム 209	≤ 225	円 241
2	(LINE TOP) 2	(INS) 18	" 34	2 50	B 66	R 82	b 98	r 114	√ 130	2 146	「 162	イ 178	ツ 194	メ 210	≠ 226	年 242
3	3	19	# 35	3 51	C 67	S 83	c 99	s 115	/ 131	3 147	」 163	ウ 179	テ 195	モ 211	↑ 227	月 243
4	(SHIFT RELEASE) 4	20	$ 36	4 52	D 68	T 84	d 100	t 116	Σ 132	4 148	、 164	エ 180	ト 196	ヤ 212	← 228	日 244
5	(LINE CANCEL) 5	21	% 37	5 53	E 69	U 85	e 101	u 117	Ω 133	5 149	・ 165	オ 181	ナ 197	ユ 213	↓ 229	千 245
6	(LINE END) 6	22	& 38	6 54	F 70	V 86	f 102	v 118	░ 134	6 150	ヲ 166	カ 182	ニ 198	ヨ 214	→ 230	万 246
7	(BEL) 7	23	' 39	7 55	G 71	W 87	g 103	w 119	■ 135	7 151	ァ 167	キ 183	ヌ 199	ラ 215	π 231	£ 247
8	(BS) 8	24	(40	8 56	H 72	X 88	h 104	x 120	α 136	8 152	ィ 168	ク 184	ネ 200	リ 216	♠ 232	¢ 248
9	(CAPS L-U) 9	25	) 41	9 57	I 73	Y 89	i 105	y 121	β 137	9 153	ゥ 169	ケ 185	ノ 201	ル 217	♥ 233	± 249
A	(LF) 10	26	* 42	: 58	J 74	Z 90	j 106	z 122	γ 138	+ 154	ェ 170	コ 186	ハ 202	レ 218	♦ 234	± 250
B	(HOME) 11	27	+ 43	; 59	K 75	[91	k 107	{ 123	ε 139	− 155	ォ 171	サ 187	ヒ 203	ロ 219	♣ 235	0 251
C	(CLS) 12	(→) 28	, 44	< 60	L 76	¥ 92	l 108	¦ 124	θ 140	n 156	ャ 172	シ 188	フ 204	ワ 220	□ 236	
D	(CR) 13	(←) 29	- 45	= 61	M 77	] 93	m 109	} 125	μ 141	x 157	ュ 173	ス 189	ヘ 205	ン 221	○ 237	
E	(SHIFT SET) 14	(↑) 30	. 46	> 62	N 78	^ 94	n 110	~ 126	σ 142	-1 158	ョ 174	セ 190	ホ 206	゛ 222	△ 238	
F	(CAPS U-L) 15	(↓) 31	/ 47	? 63	O 79	_ 95	o 111	127	φ 143	÷ 159	ッ 575	ソ 191	マ 207	゜ 223	／ 239	

thing is output for character codes for a character or function is not specified (indicated by a blank cell in the table).
ntrol codes are indicated by parentheses and are not displayed.
aracters which cannot be input directly can be displayed using the CHR$ function.
lues in the lower right corner of each cell indicate the decimal value of the corresponding character code.

NOTE:
The special characters in the character code table below only appear on the display an are not printed out by the printer. When a LLIST or LPRINT command is executed, they ar substituted by the differently shaped printer characters corresponding to the respective charac ter codes. Refer to the pocket computer and printer character code tables and compare ther for further details.

Character Code Table

Special characters (144–208)

High-order digit →

↓ Low-order digit

		0	16	32	48	64	80	96	112	128	144	160	176	192	208	224	240
	HEX	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0			SPC	0	@	P	`	p	Å	0	SPC	ー	タ	ミ	(≧)	(×)
1	1		(DEL)	!	1	A	Q	a	q	∫	1	。	ア	チ	ム	(≦)	円
2	2	(LINE TOP)	(INS)	"	2	B	R	b	r	√	2	「	イ	ツ	メ	(≠)	年
3	3			#	3	C	S	c	s	'	3	」	ウ	テ	モ	(↑)	月
4	4	(SHIFT RELEASE)		$	4	D	T	d	t	(Σ)	4	、	エ	ト	ヤ	(←)	日
5	5	(LINE CANCEL)		%	5	E	U	e	u	(Ω)	5	・	オ	ナ	ユ	(↓)	千
6	6	(LINE END)		&	6	F	V	f	v	(F)	6	ヲ	カ	ニ	ヨ	(→)	万
7	7	(BEL)		'	7	G	W	g	w	(I)	7	ァ	キ	ヌ	ラ	(π)	£
8	8	(BS)		(	8	H	X	h	x	α	8	ィ	ク	ネ	リ	(♠)	¢
9	9	(CAPS L→U)		)	9	I	Y	i	y	β	9	ゥ	ケ	ノ	ル	(♥)	±
10	A	(LF)		*	:	J	Z	j	z	(γ)	+	ェ	コ	ハ	レ	(♦)	∓
11	B	(HOME)		+	;	K	[	k	{	ε	−	ォ	サ	ヒ	ロ	(♣)	°
12	C	(CLS)	(⇨)	,	<	L	¥	l	¦	θ	n	ャ	シ	フ	ワ	(□)	(hatched)
13	D	(CR)	(⇦)	−	=	M	]	m	}	(μ)	x	ュ	ス	ヘ	ン	(○)	(hatched)
14	E	(SHIFT SET)	(⇧)	.	>	N	^	n	~	(σ)	⁻¹	ョ	セ	ホ	゛	(△)	(hatched)
15	F	(CAPS U→L)	(⇩)	/	?	O	_	o		ϕ	(÷)	ッ	ソ	マ	゜	(＼)	(hatched)

- Characters marked with a circle (○) are available on the PB-100 series.
- Except for the special characters, all can be printed with the FP-40 and FP-100.
- To display characters that can not be input directly, use the respective CHR$ function.
  Example: Display "Σ"
  CAL mode: CHR$ (132) EXE or CHR$ (&H84) EXE
  BASIC mode: PRINT CHR$ (132) EXE or PRINT CHR$ (&H84) EXE
- The hatched parts in the table (hexadecimal FC – FF (decimal 252–255)) are used internally and therefore not defined.

12-2 ERROR MESSAGE TABLE

Error code	Error message	Meaning	Correction
1	OM error	a) Insufficient memory or system overflow. b) Erroneous CLEAR statement specification.	a) Shorten program and check array dimensioning. b) Check CLEAR statement value. c) Use RAM expansion pack.
2	SN error	Erroneous command or statement format.	a) Check spelling of commands. b) Check program input.
3	ST error	String length exceeds 255 characters.	Shorten string to 255 characters or less.
4	TC error	Formula too complex.	Divide formula into smaller subformulas
5	BV error	a) I/O buffer overflow. b) Line length exceed 255 bytes or 255 characters.	a) Set RS-232C baud rate to lower value or set XON/OFF. b) Keep lines 255 characters or less in length.
6	NR error	I/O device not ready for input/output.	Check connection and power switch of I/O device.
7	RW error	Error generated in I/O device operation.	Check I/O device.
8	BF error	Improper filename specification.	Check filename.
9	BN error	Improper file number specification.	Check file number.
13	OV error	Value exceeds allowable calculation result or input range.	Check values.
14	MA error	a) Mathematical error such as division by zero. b) Argument exceeds allowable calculation range.	Check expressions and values.
15	DD error	Double declaration of identical array.	Either erase previous array or use a different array name.
16	BS error	Subscript or parameter outside of allowable range.	a) Check subscripts. b) Increase size of arrays.
17	FC error	a) Erroneous use of function or statement. b) Illegal command used in direct mode or program mode. c) Illegal command used in CAL mode. d) Attempt to use undeclared array.	a) Check argument values and statements. b) Check for statements that can not be used in respective mode. c) Check statements. d) Declare array using DIM statement.

Error code	Error message	Meaning	Correction
18	UL error	a) Branch destination line number does not exist. b) Input of statement without line number in BASIC editing mode.	a) Check line numbers. b) Always use line numbers in BASIC editing mode.
19	TM error	a) Mismatch of variable type and contents. b) Mismatch of READ statement variable and data. c) Mismatch of INPUT# statement variable and data.	Check for illegal numeric assignment to string variables or string assignment to numeric variable.
20	RE error	RESUME statement outside of error handling routine.	Check RESUME statement location.
21	PR error	Execution of command that cannot be used with password protected files.	Cancel password.
22	DA error	READ statement execution when no data present.	Check READ and DATA statements.
23	FO error	No FOR for NEXT statement.	Check for matching of FOR and NEXT statements.
24	NX error	No NEXT for FOR statement.	Check for matching of FOR and NEXT statements.
25	GS error	Mismatch of GOSUB and RETURN statements.	Check for matching of GOSUB and RETURN statements.
26	OP error	a) Attempt to access unopened file. b) Attempt to open already opened file.	a) Execute OPEN statement. b) CLOSE file and then reopen.
29	AM error	Attempt to use output-related command for device opened for input or vice versa.	Ensure proper use of input-related and output-related commands.
30	FR error	Framing error detected by RS-232C port.	Check RS-232C connection and data transmission method.
31	PO error	Parity error or over run error detected by RS-232C port. Erroneous read from the cassette tape recorder.	a) Check RS-232C connection and data transmission method. b) Adjust the playback volume of the tape recorder. c) Attempt using the phase which is opposite the current setting. d) Clean the head of the tape recorder.

12-3 COMMAND/FUNCTION TABLE

COMMANDS

PASS	ON GOTO	BEEP
NEW (ALL)	ON GOSUB	INPUT
CLEAR	IF ~ THEN ~ ELSE	INKEY$
FRE	IF ~ GOTO ~ ELSE	INPUT$
LIST (ALL)	FOR ~ NEXT	DIM
EDIT	REM	ERASE
VARLIST	LET	PEEK
RUN	DATA	POKE
TRON	READ	DEFSEG
TROFF	RESTORE	ON ERROR GOTO
END	PRINT	RESUME
STOP	TAB	ERL
GOTO	LOCATE	ERR
GOSUB	CLS	
RETURN	SET	

INPUT/OUTPUT COMMANDS

LLIST	INPUT$
LPRINT	EOF
OPEN	SAVE (ALL)
CLOSE	LOAD (ALL)
PRINT #	VERIFY
INPUT #	

DATA BANK COMMAND

NEW #	RESTORE #
LIST #	WRITE #
LLIST #	
SAVE #	
LOAD #	
READ #	

FUNCTIONS

ANGLE	HYPACS	FRAC	STR$
SIN	HYPATN	ROUND	VAL
COS	EXP	RAN #	VALF
TAN	LOG	PI	MID$
ASN	LN	FACT	RIGHT$
ACS	SQR	NPR	LEFT$
ATN	CUR	NCR	LEN
HYPSIN	ABS	POL	&H
HYPCOS	SGN	REC	HEX$
HYPTAN	INT	CHR$	DEG
HYPASN	FIX	ASC	DMS$

12-4 RESERVED WORD LIST

[A] ABS, ACS, ALL, AND, ANGLE, AS, ASC, ASN, ATN

[B] BEEP

[C] CALC, CHR$, CLEAR, CLOSE, CLS, COS, CSR, CUR

[D] DATA, DEF, DEFM, DEFSEG, DEG, DIM, DMS$

[E] EDIT, ELSE, END, EOF, ERASE, ERL, ERR, ERROR, EXP

[F] FACT, FIX, FOR, FRAC, FRE

[G] GOSUB, GOTO

[H] HEX$, HYP

[I] IF, INKEY$, INPUT, INT

[K] KEY

[L] LEFT$, LEN, LET, LIST, LLIST, LN, LOAD, LOCATE, LOG, LPRINT

[M] MID, MID$, MOD, MODE

[N] NCR, NEW, NEXT, NOT, NPR

[O] ON, OPEN, OR, OUT

[P] PASS, PBGET, PBLOAD, PEEK, PI, POKE, POL, PRINT, PUT

[R] RAN#, READ, REC, REM, RESTORE, RESUME, RETURN, RIGHT$, RND, ROUND, RUN

[S] SAVE, SET, SGN, SIN, SQR, STEP, STOP, STR$

[T] TAB, TAN, THEN, TO, TROFF, TRON

[V] VAC, VAL, VALF, VAR, VERIFY

[W] WRITE#

[X] XOR

SPECIFICATIONS

Model:
FX-850P

Basic calculation functions:
Negative numbers, exponents, parenthetical arithmetic operations (with priority sequence judgment function—true algebraic logic), integer division, integer division remainders, logical operators.

Built-in functions:
Trigonometric/inverse trigonometric functions (angle units: degrees, radians, grads), logarithmic/exponential functions, square roots, cube roots, powers, hyperbolic/inverse hyperbolic functions, conversion to integer, deletion of integer portion, absolute values, signs, coordinate conversions, factorials, permutations, combinations, rounding, random numbers, pi, decimal-sexagesimal conversions, decimal-hexadecimal conversions.

Number of built-in scientific library:
116

Commands:
EDIT, LIST, LLIST, LOAD, NEW, NEW ALL, RUN, SAVE, VERIFY, ANGLE, BEEP, CLEAR, CLOSE, CLS, DEFSEG, DIM, ERASE, LET, LOCATE, LPRINT, PASS, POKE, PRINT, SET, TROFF, TRON, VARLIST, DATA, END, FOR ~ NEXT ~ STEP, GOSUB ~ RETURN, GOTO, IF ~ THEN ~ ELSE, INPUT, INPUT #, ON ~ ERROR ~ GOTO, ON ~ GOSUB, ON ~ GOTO, OPEN, PRINT #, READ, REM, RESTORE, RESUME, RETURN, STOP, LIST #, LLIST #, LOAD #, SAVE #, NEW #, READ #, RESTORE #, WRITE #

Program functions:
ASC( ), CHR$( ), INKEY$, INPUT$, LEFT$, LEN( ), MID$( ), RIGHT$( ), STR$( ), TAB( ), VAL( ), VALF( )

Other functions:
EOF( ), ERL, ERR, PEEK( )

Calculation range:
$\pm 1 \times 10^{-99} \sim \pm 9.999999999 \times 10^{99}$ and 0. Internal operation uses 12-digit mantissa.

Program system:
Stored system

Program language:
BASIC

RAM capacity:
Standard 8KB, expandable up to 40KB (Including 3KB in system area).

Number of program areas:
Maximum 10 (P0 through P9)

Number of stacks:
Subroutine: 96 levels
FOR ~ NEXT loop : 29 levels

Display contents:
10-digit mantissa + 2-digit exponent

Display elements:
32-column × 2-line dot matrix liquid crystal display

Main components:
C-MOS VLSI and others

Power supply:
2 lithium batteries (CR2032) for the mainframe
1 lithium battery (CR1220) for memory backup

Power consumption:
0.04W

Battery life:
1. Continuous program execution: Approx. 90 hours
2. Continuous display of 5555555555 at 20° C (68°F): Approx. 150 hours
   4.5 months when unit is used 1 hour per day.

* Note: 1 hour includes 10 minutes of condition 1 and 50 minutes of condition 2.

Memory protection battery:
Approx. 2 years (with main batteries installed)

Auto power-off:
Approximately 6 minutes

Ambient temperature range:
0° C to 40° C (32° F to 104° F)

Dimensions:
11.6 (H) × 193 (W) × 78 (D) mm
(1/2" (H) × 7 5/6" (W) × 3" (D))

Weight:
197g (6.9oz) including batteries.

Accessory:
Hard case

BASIC COMMAND INDEX

R

S

T

V

W

GUIDELINES LAID DOWN BY FCC RULES FOR USE OF THE UNIT IN THE U.S.A. (not applicable to other areas).

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

....reorient the receiving antenna
....relocate the computer with respect to the receiver
....move the computer away from the receiver
....plug the computer into a different outlet so that computer and receiver are on different branch circuits.

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