CASIO®

SH1026218A 英 Printed in Japan

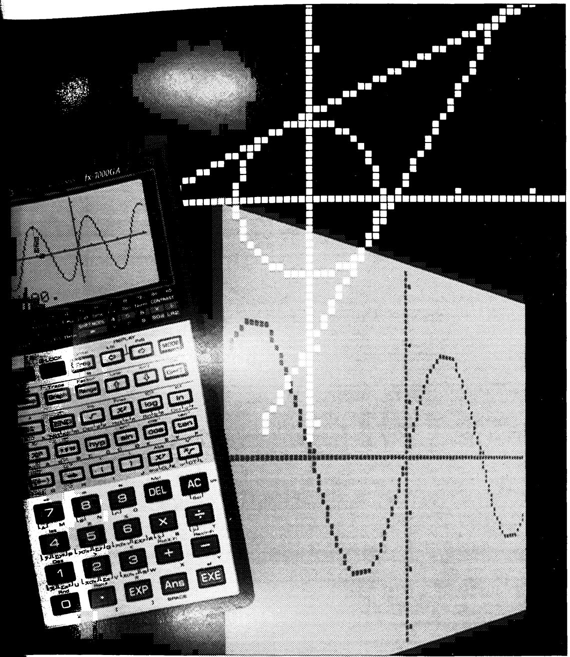
fx-7000GA

OWNER'S MANUAL

CASIO®

fx-7000GA

OWNER'S MANUAL



CASIO®

FOREWORD

Thank you for your purchase of the CASIO fx-7000GA.
This unit is a totally new type of advanced programmable computer. Besides 82 scientific functions, graph functions also make it possible to produce a wide variety of useful graphs.
Manual computations can be easily performed following written formulas (true algebraic logic). A replay function is provided that allows confirmation or correction when key operation errors occur. Programs can also be input by following true algebraic logic, so repeat and/or complex computations are simplified.

This manual is composed of four sections:

1. Configuration and Operation
2. Manual Computations
3. Graphs
4. Program Computations

Section 1 should be read first to become familiar with the nomenclature, handling and cautions concerning this unit. Sections 2, 3 and 4 can then be read in order to master each type of functions through samples and explanations.

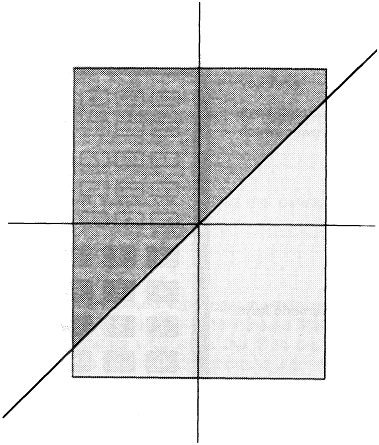
CONTENTS

HANDLING PRECAUTIONS

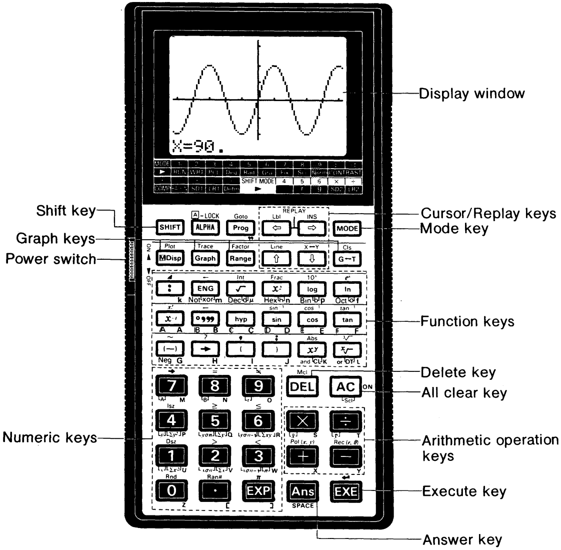
- This unit is composed of precision electronic components and should never be disassembled. Do not drop it or otherwise subject it to sudden impacts or sudden temperature changes. Be especially careful to avoid storing the unit or leaving it in areas exposed to high temperature, humidity or large amounts of dust. When exposed to low temperatures, the unit will require more time to display answers and may even fail to operate. The display will return to normal once normal temperature is attained.
- Batteries should be replaced every 2 years even if the unit is not used for extended periods. Never leave dead batteries in the battery compartment. They can leak and cause damage to the unit.
- Avoid using volatile liquids such as thinner or benzine to clean the unit. Wipe the unit with a soft, dry cloth or a cloth that has been dipped in a neutral detergent solution and wrung out.
- If malfunction of the unit should occur, either bring or send the unit to your retailer or the nearest CASIO dealer.
  Be sure to clearly explain the problem in detail.
- Before assuming malfunction of the unit, be sure to carefully reread this manual and ensure that the problem is not due to insufficient battery power, programming or operational errors.

1.

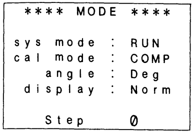
CONFIGURATION AND OPERATION



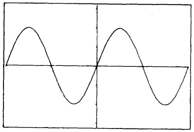
1-1 NOMENCLATURE AND FUNCTIONS



Display window



```
**** MODE ****

sys mode : RUN
cal mode : COMP
   angle : Deg
 display : Norm

   Step    0
```



The display window is capable of displaying 16-character by 8-line text and symbols. Graphs are produced on a 95 by 63-dot matrix. A system display as shown on the left indicates the following: the system mode (sys mode), calculation mode (cal mode), angle unit (angle), number of decimal places or number of significant digits (display), and key input buffer status (Step).

The display on the right shows a sine graph as a representative example of the graphs.

The letter "O" is distinguished from zero by adding a slash for the zero (0).

Power switch

Power is turned ON by sliding the power switch up. Sliding the power switch down turns power OFF.

Special operation keys

SHIFT Shift key

Press when using the function commands and functions marked in brown on the key panel. An S will blink on the display to indicate that SHIFT has been pressed. Pressing SHIFT again will cause the S to disappear from the display and the unit to return to the status it was in before SHIFT was originally pressed.

MODE Mode Key

Use the MODE key in combination with ·, 1 through 9, +, −, and ÷ to specify the calculation mode and the unit of angular measurement.

[MODE] [1] ... For manual computations and program execution.
[MODE] [2] ... For writing or checking programs.
[MODE] [3] ... For clearing programs.
[MODE] [4] ... Deg displayed. If [EXE] is pressed, unit of angular measurement is specified as degress.
[MODE] [5] ... Rad displayed. If [EXE] is pressed, unit of angular measurement is specified as radians.
[MODE] [6] ... Gra displayed. If [EXE] is pressed, unit of angular measurement is specified as grads.
[MODE] [7] ... Fix displayed. Entering a value from 0 to 9 followed by [EXE] will specify the number of decimal places according to the value entered.
Ex. [MODE] [7] [3] [EXE] → Three decimal places
[MODE] [8] ... Sci displayed. Entering a value from 0 to 9 followed by [EXE] will specify the number of significant digits from 1 to 10.
Ex. [MODE] [8] [5] [EXE] → 5 significant digits
[MODE] [9] ... Norm displayed. Pressing [EXE] will cancel the specified number of decimal places or the specified number of significant digits.
[MODE] [·] ... Defm displayed. Entering a value followed by [EXE] will specify the number of memories available.
Ex. [MODE] [·] [1] [0] [EXE] → Number of memories available increased by 10.
If [EXE] is pressed without entering a value, the current number of memories available and remaining steps will be displayed. (See page 23.)
Ex. [MODE] [·] [EXE]

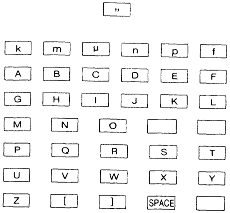
```
**Defm**

 Program : 56

  Memory : 36

 286 Bytes Free
```

[MODE] [+] ... Specifies COMP mode for arithmetic computation or function computation (program execution possible).
[MODE] [−] ... For binary, octal or hexadecimal computations/conversions.

[MODE] [×] ... For standard deviation computations (SD1 mode).
[MODE] [÷] ... For regression computations (LR1 mode).
[SHIFT] [MODE] [×] ... For production of a bar graph, line graph or normal distribution curve according to single variable statistical data (SD2 mode).
[SHIFT] [MODE] [÷] ... For production of a regression line according to paired variable statistical data (LR2 mode).
The x^y and $\sqrt[x]{\ }$ functions are not available in the LR1 mode. To use these functions, first perform the statistical operations and then press [MODE] [+] to enter the COMP mode.
[SHIFT] [MODE] [4] ... Pressed after a numeric value representing degrees (°) is input.
[SHIFT] [MODE] [5] ... Pressed after a numeric value representing radians $(^r)$ is input.
[SHIFT] [MODE] [6] ... Pressed after a numeric value representing grads is input.

[ALPHA] Alphabet key

Press to input alphabetic characters or special characters. Pressing [ALPHA] displays **A** and allows the input of only one character. After that, the unit returns to the status it was in before the [ALPHA] key was originally pressed. Pressing [SHIFT] followed by [ALPHA] will lock the unit in this mode and allow consecutive input of alphabetic characters until [ALPHA] is pressed again.



Goto [Prog] Program/Goto key

Press [Prog], enter a value from 0 to 9 and then press [EXE] to execute a program.
Ex. [Prog] [1] [EXE] → Execution of Program 1 begins.
Pressing [SHIFT] followed by [Goto] (Goto [Prog] key) will cause Goto to appear on the display. This is a jump command used in programs.

Lbl REPLAY INS ⇦ ⇨ Line X↔Y ⇧ ⇩ Cursor/Replay keys

Press to move the cursor (blinking "_") left, right, up, and down on the display. The ⇦ key moves the cursor to the left, ⇨ moves the cursor to the right, ⇧ moves the cursor up, and ⇩ moves the cursor down. Holding any of the keys down will cause the cursor to continuously move in the respective direction.

Once a formula or numeric value is input and EXE is pressed, the ⇦ key and ⇨ key become "replay" keys. In this case, pressing ⇨ displays the formula or numeric value from the beginning, while pressing ⇦ displays it from the end. This allows the formula to be executed again by changing the values.

Pressing the cursor key following SHIFT changes their functions to those marked above the keys.

Lbl (⇦) is used to input labels within programs.

INS (⇨) inserts a space at the current position of the cursor. Line (⇧) makes it possible to produce line graphs or regression lines. The X↔Y (⇩) key makes it possible to switch the X and Y coordinate display during graph trace operations.

⇦ and ⇨ following the MODE key are used for contrast adjustments. (See page 11.)

Mcl DEL Delete key

Press to delete the character at the current position of the cursor. When the character is deleted, everything to the right of the cursor position will shift one space to the left.

Pressing SHIFT Mcl (DEL) EXE will clear the memory contents.

AC All clear key

Press to completely clear the displayed formulas, numeric values or texts, and to clear all of the input buffer contents. Also used to release errors indicated by error message displays, and to restore power after reactivation of the auto power off function. (See page 25.)

EXE Execute key

Press to obtain the result of a computation or to draw a graph. Pressed after data input for a programmed computation or to advance to the next execution after a computation result is obtained.

Ans Answer key

Pressing Ans followed by EXE will recall the last computation result. It can be recalled by Ans EXE even after it has been cleared using the AC key or by switching the power of the unit OFF. When used during program execution, the last result computed is recalled.

■ 1 ~ 9, ·, EXP Numeric/Decimal point/Exponent input keys

When entering numeric values, enter the number in order. Press the · key to enter the decimal point in the desired position.

To input 1.23×10^{-6}, press 1 · 23 EXP (−) 6.

SHIFT key combinations for the various modes are as follows:

COMP mode (MODE +)

➡	=	≠
Isz	≥	≤
Dsz	>	<
Rnd	Ran#	π

BASE-N mode (MODE −)

➡	=	≠
Isz	≥	≤
Dsz	>	<

Pol(, Rec(, Rnd, Ran# and π cannot be used in this mode.

SD mode (MODE ×)

➡	=	≠
Isz	≥	≤
$\bar{x}$	$x\sigma_n$	$x\sigma_{n-1}$
Rnd	Ran#	π

Standard deviation functions can be used.

LR mode (MODE ÷)

A	B	r
$\bar{y}$	$y\sigma_n$	$y\sigma_{n-1}$
$\bar{x}$	$x\sigma_n$	$x\sigma_{n-1}$
Rnd	Ran#	π

Paired variable statistic functions can be used.

■ Computation keys

+ − × ÷ Arithmetic operation keys

For addition, subtraction, multiplication and division, enter the computation as it reads. SHIFT key combinations for the various modes are as follows:

COMP mode or SD mode

Pol(Rec((Pol(x,y) + and Rec(r,θ) − keys) ... Coordinate transformation

LR mode

$\hat{x}$ $\hat{y}$ ($\hat{x}$ × $\hat{y}$ ÷ keys) ... Estimated value computation of x and y

Pol(Rec(... Coordinate transformation

■ Graph keys

Used to produce a variety of graphs (see page 51 for details). These keys cannot be used in the Base-n mode.

Plot
M Disp Mode display/Plot key

- Used to confirm the status of the system mode, calculation mode, angle unit and rounding. Setting status is displayed only while this key is pressed.
- Pressed following SHIFT to plot a point on the graph screen.

Trace
Graph Graph/Trace key

- Pressed before entering a formula to be used for a graph ("Graph Y=" appears on the display).
- Pressed following SHIFT to trace over an existing graph and display the x or y coordinate value.

Factor
Range Range/Factor key

- Used to confirm or set the range and size of graphs.
- Pressed following SHIFT to magnify or reduce the upper and lower ranges of graphs.

Cls
G↔T Graph-text/Clear screen key

- Switches between the graph display and text display (see page 19).
- SHIFT Cls EXE clears the graph display. The text display cannot be cleared using this operation.

■ Function keys

Press for functional computation. Various uses are available in combination with the SHIFT key, and/or depending on the mode being used.

◢
: Multistatement/Display key
K

- Press to separate formulas or commands in programmed computations or consecutive computations.
  The result of such combinations is known as a multistatement.
  (See page 32.)
- When pressed following the SHIFT key, the results of each section of the programmed computations or consecutive computations are sequentially displayed with each press of EXE.

←
ENG Engineering/Negation key
Not xor m

- Press to convert a computaion result to an exponential display whose exponent is a multiple of three.
  $(10^3=\overset{kilo}{k},\ 10^6=\overset{mega}{M},\ 10^9=\overset{giga}{G},\ 10^{-3}=\overset{milli}{m},\ 10^{-6}=\overset{micro}{\mu},\ 10^{-9}=\overset{nano}{n},\ 10^{-12}=\overset{pico}{p})$

- When obtaining logical negation for a value in the Base-n mode, press prior to entering the value.
- Press following the SHIFT key in the Base-n mode to obtain the exclusive logical sum.

Int
√ Root/Integer key
Dec d μ

- Press prior to entering a numeric value to obtain the square root of that value.
- When pressed following the SHIFT key, the integer portion of a value can be obtained.
- Press followed by EXE in the Base-n mode to specify the decimal computation mode.
- When pressed following the SHIFT key in the Base-n mode, the subsequently entered value is specified as a decimal value.

Frac
x^2 Square/Fraction key
Hex h n

- Press after a numeric value is entered to obtain the square of that value.
- When pressed following the SHIFT key, the decimal portion of a value can be obtained.
- Press followed by EXE in the Base-n mode to specify the hexadecimal computation mode.
- When pressed following the SHIFT key in the Base-n mode, the subsequently entered value is specified as a hexadecimal value.

10^x
log Common logarithm/Antilogarithm key
Bin b p

- Press prior to entering a value to obtain the common logarithm of that value.
- When pressed following the SHIFT key, the subsequently entered value becomes an exponent of 10.
- Press followed by EXE in the Base-n mode to specify the binary computation mode.
- When pressed following the SHIFT key in the Base-n mode, the subsequently entered value is specified as a binary value.

e^x
ln Natural logarithm/Anti-natural logarithm key
Oct o f

- Press prior to entering a value to obtain the natural logarithm of that value.
- When pressed following the SHIFT key, the subsequently entered value becomes an exponent of e.
- Press followed by EXE in the Base-n mode to specify the octal computation mode.
- When pressed following the SHIFT key in the Base-n mode, the subsequently entered value is specified as an octal value.

x^{-1} Reciprocal/Factorial key

- Press after entering a value to obtain the reciprocal of that value.
- When pressed following the SHIFT key, the factorial of a previously entered value can be obtained.
- Press in the Base-n mode to enter A (10_{10}) of a hexadecimal value.

° ’ ” Degree/minute/second key (decimal↔sexagesimal key)

- Press to enter sexagesimal value. (degree/minute/second or hour/minute/second)
  Ex. 78°45’12”→78 ° ’ ” 45 ° ’ ” 12 ° ’ ”
- When pressed following the SHIFT key, a decimal based value can be displayed in degrees/minutes/seconds (hours/minutes/seconds).
- Press in the Base-n mode to enter B (11_{10}) of a hexadecimal value.

hyp Hyperbolic key

- Pressing hyp, and then sin, cos, or tan prior to entering a value produces the respective hyperbolic function (sinh, cosh, tanh) for the value.
- Pressing SHIFT, then hyp and then sin, cos, or tan prior to entering a value produces the respective inverse hyperbolic function ($\sinh^{-1}$, $\cosh^{-1}$, $\tanh^{-1}$) for the value.
- Press in the Base-n mode to enter C (12_{10}) of a hexadecimal value.

sin cos tan Trigonometric function/Inverse trigonometric function keys

- Press one of these keys prior to entering a value to obtain the respective trigonometric function for the value.
- Press SHIFT and then one of these keys prior to entering a value to obtain the respective inverse trigonometric function for the value.
- Press in the Base-n mode to enter D, E, F (13_{10}, 14_{10}, 15_{10}) of a hexadecimal value.

(−) Minus key

- Press prior to entering a numeric value to make that value negative.
  Ex. −123→ (−) 1 2 3
- When pressed following the SHIFT key, the same numeric value can be assigned to multiple memories.
  Ex. To assign the value 456 to memories A through F: 4 5 6 → ALPHA A SHIFT ~ ALPHA F EXE
- Press in the Base-n mode prior to entering a value to obtain the negative of that value. The negative number is the two's complement of the value entered.

→ Assignment key

- Press prior to entering a memory to assign the result of a computation to that memory.
  Ex. To assign the result of 12+45 to memory A: 1 2 + 4 5 → ALPHA A EXE
- During execution of program computations or consecutive computations, press following the SHIFT key to enter a numeric value.

( ) Parenthesis keys

- Press the open parenthesis key and the closed parenthesis key at the position required in a formula.
- When pressed following the SHIFT key, a comma or semicolon can be inserted to separate the arguments in coordinate transformation or consecutive computations.

x^y Power/Absolute value key

- Enter x (any number), press this key and then enter y (any number) to compute x to the power of y.
  In the SD or LR mode, this function is only available after pressing the SHIFT key.
- Press following the SHIFT key to obtain the absolute value of a subsequently entered numeric value.
- Press in the Base-n mode to obtain a logical product ("and").
- Press in the SD or LR mode to delete input data.

$\sqrt[x]{\ }$ Root/Cube root key

- Enter x, press this key and then enter y to compute the xth root of y. In the SD or LR mode, this function is only available after pressing the SHIFT key.
- Press following the SHIFT key to obtain the cube root of a subsequently entered numeric value.
- Press in the Base-n mode to obtain a logical sum ("or").
- Used as a data input key in the SD or LR mode.

■ Contrast adjustment

Pressing the ⇦ or ⇨ key following the MODE key adjusts the contrast of the display. Pressing ⇦ makes the screen lighter, while ⇨ makes it darker. Holding either key down will cause the display to successively become respectively lighter or darker.

Pressing any other key besides MODE, ⇦, or ⇨ (as well as ⇧, ⇩) cancels contrast adjustment.

* *Light display contrast even at the darkest setting indicates that battery power is too low. In this case, replace batteries as soon as possible.*

* *Contrast adjustment is impossible during range display using the* Range *key. (See page 51.)*

1-2 POWER AND BATTERY REPLACEMENT

PRECAUTIONS:
Incorrectly using batteries can cause them to burst or leak, possibly damaging the interior of the unit. Note the following precautions:

- Be sure that the positive (+) and negative (−) poles of each battery are facing in the proper direction.
- Never mix batteries of different types.
- Never mix old batteries and new ones.
- Never leave dead batteries in the battery compartment.
- Remove batteries if you do not plan to use the unit for long periods.
- Replace batteries at least once every 2 years, no matter how much the unit is used during that period.
- Never try to recharge the batteries supplied with the unit.
- Do not expose batteries to direct heat, let them become shorted, or try to take them apart.

Keep batteries out of the reach of small children. If swallowed, consult with a physician immediately.

Power is supplied to this unit by three lithium batteries (CR2032C). If the power of the batteries should diminish, the display will weaken and become difficult to read. A weak display even after contrast adjustment (see page 11) may indicate power is too low, so the batteries should be replaced. When making replacements, be sure to replace all three batteries.

* *If batteries are used for longer than two years, there is the danger of leakage. Be sure to replace batteries at least once every two years even if the unit is not used during that period.*

* *The life of the original batteries supplied with the calculator is calculated from the date of installation at the factory, not from the date of purchase.*

* *Stored programs or data are erased when batteries are replaced. Therefore, it is recommended that programs and data required for later use be recorded on a coding sheet before replacing batteries.*

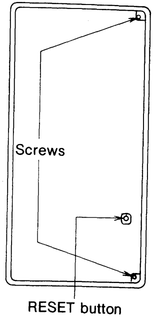
* *Be sure to use batteries specified by Casio.*

■ **Procedure**

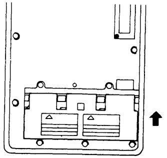
① Slide the power switch to the OFF position, remove the two screws on the back of the unit with a screwdriver, and remove the back cover.

② Slide the battery pressure plate in the direction indicated by the arrows and remove it.

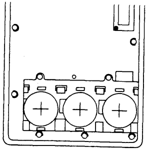
③ Remove the three old batteries from the unit.
(This can be done easily by turning the unit so the battery compartment is facing downwards, and then lightly tapping the unit.)



④ Wipe the surfaces of three new batteries with a soft, dry cloth and load them into the unit ensuring that the positive ⊕ sides are facing upwards.



⑤ Fasten the battery pressure plate in place, and replace the back cover.



* *IMPORTANT: Never dispose of old batteries in such a way that they will be incinerated. Batteries may explode if exposed to fire.*

1-3 BEFORE BEGINNING COMPUTATIONS...

■ Computation priority sequence

This unit employs true algebraic logic to compute the parts of a formula in the following order:

1. Coordinate transformation Pol (x, y), Rec (r, θ)
2. Type A functions* x^2, x^{-1}, $x!$, $^{\circ}$, ^{r}, ^{g}, °’’’
3. Power/root x^y, $\sqrt[x]{\ }$
4. Abbreviated multiplication format in front of π or memory
   2π, 4R, etc.
5. Type B functions* $\sqrt{\ }$, $\sqrt[3]{\ }$, log, 10^x, ln, e^x, sin, cos, tan, $\sin^{-1}$, $\cos^{-1}$, $\tan^{-1}$, sinh, cosh, tanh, $\sinh^{-1}$, $\cosh^{-1}$, $\tanh^{-1}$, (−), Abs, Int, Frac, h, d, b, o, Neg, Not
6. Abbreviated multiplication format in front of Type B functions or parenthesis 3sin5, $6\sqrt{7}$, 2sin30cos60, etc.
7. ×, ÷
8. +, −
9. and
10. or, xor
11. Relational operators <, >, =, ≠, ≤, ≥

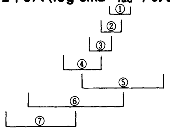
* Functions are divided into two types.
Type A functions are entered after the argument, while Type B functions are entered before the argument.

* *When functions with the same priority are used in series, execution is performed from right to left: e.g.,* $e^x\ln\sqrt{120} \rightarrow e^x\{\ln\sqrt{120}\}$.
Otherwise, execution is from left to right.

* *Compound functions are executed from right to left:*
e.g., sin $\cos^{-1}$0.6 → sin ($\cos^{-1}$0.6).

* *Everything contained within parentheses receives highest priority.*

Ex. 2+3×(log sin2π^2_{rad}+6.8)=22.07101691

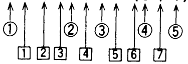


■ Number of stacks

This unit features a memory known as a stack for the temporary storage of low priority numeric values and commands (functions, etc). The numeric value stack has eight levels, while the command stack has twenty. If a complex formula is employed that exceeds the stack space available, a stack error (Stk ERROR) message will appear on the display.

Ex. Stack counting method

2× ((3+4× (5+4) ÷3) ÷5) +8=



Numeric value stack

①	2
②	3
③	4
④	5
⑤	4
⋮	

Command stack

$\boxed{1}$	×
$\boxed{2}$	(
$\boxed{3}$	(
$\boxed{4}$	+
$\boxed{5}$	×
$\boxed{6}$	(
$\boxed{7}$	+
⋮	

* *Computations are performed in the order of the highest computation priority first. Once a computation is executed, it is cleared from the stack.*

■ Computation modes

This unit features modes for manual computations, storing programs, and modes for general as well as statistical computations. The proper mode to suit computational requirements should be employed.

● Operation modes

There are a total of three operation modes.

1. RUN mode
   Graph production as well as manual computations and program executions.
2. WRT mode
   Program storage and editing. (See Section 4.)
3. PCL mode
   Deletion of stored programs. (See Section 4.)

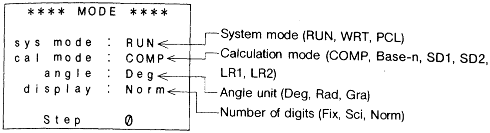
● Computation modes

There are a total of six computation modes which are employed according to the type of computation.

1. COMP mode
   General computations, including functional computations.
2. Base-n mode
   Binary, octal, decimal, hexadecimal conversion and computations, as well as logical operations. (See page 40.) Function computations and graph drawing cannot be performed.
3. SD1 mode
   Standard deviation computation (single variable statistics). (See page 44.)
4. SD2 mode
   For production of bar graph, line graph or normal distribution curve according to single variable statistical data. (See page 73.)
5. LR1 mode
   Regression computation (paired variable statistics). (See page 46.)
6. LR2 mode
   For production of regression line graph according to paired variable statistical data. (See page 76.)

With so many modes available, computations should always be performed after confirming which mode is active.

* *IMPORTANT: When the power of the unit is switched OFF (including auto power off), the current system mode is cancelled, and the unit will be set to the RUN mode when switched ON again. However, the calculation mode, number of decimal place setting ($\boxed{\text{MODE}}$ $\boxed{7}$ n), number of significant digits ($\boxed{\text{MODE}}$ $\boxed{8}$ n), and angle unit (Deg, Rad, Gra) will be retained in memory. The mode setting is displayed when the power of the unit is switched ON. Confirm whether the desired mode is set before performing calculations.*



```
**** MODE ****

sys mode  :  RUN
cal mode  :  COMP
   angle  :  Deg
 display  :  Norm

  Step       0
```

● RESET operation

The RESET operation can be used to clear the unit if it becomes locked up because of static electrical charge. Note, however, that the RESET operation clears all data and programs from memory. With power ON, use a thin pointed object to press the RESET key on the back of the unit. Once the unit is RESET, it will be in the following initial modes.

Mode	Initial Setting
System	RUN
Calculation	COMP
Angle	DEG
Display	NORM

* *The RESET operation clears all programs and data from memory. Be sure to keep written copies of all important programs and data.*

■ Number of input/output digits and computation digits

- The allowable input/output range (number of digits) of this unit is 10 digits for a mantissa and 2 digits for an exponent. Computations, however, are internally performed with a range of 13 digits for a mantissa and 2 digits for an exponent.

Ex. $3 \times 10^5 \div 7 =$

3 $\boxed{\text{EXP}}$ 5 $\boxed{\div}$ 7 $\boxed{\text{EXE}}$	42857.14286
3 $\boxed{\text{EXP}}$ 5 $\boxed{\div}$ 7 $\boxed{-}$ 42857 $\boxed{\text{EXE}}$	0.14285714

* *Computation results greater than 10^{10} (10 billion) or less than 10^{-2} (0.01) are automatically displayed in exponential form.*

Ex. $123456789 \times 9638 =$

123456789 $\boxed{\times}$ 9638 $\boxed{\text{EXE}}$	1.189876532 E+12

Mantissa　Exponent

Once a computation is completed, the mantissa is rounded off to 10 digits and displayed. And the displayed mantissa can be used for the next computation.

Ex. $3 \times 10^5 \div 7 =$

3 [EXP] 5 [÷] 7 [EXE]	42857.14286
[−] 42857 [EXE]	0.14286

* *Values are stored in memory with 13 digits for the mantissa and 2 digits for the exponent.*

■ Overflow and errors

If the computational range of the unit is exceeded, or incorrect inputs are made, an error message will appear on the display window and subsequent operation will be impossible. This is the error check function. The following operations will result in errors:

(1) The answer, whether intermediate or final, or any value in memory exceeds the value of $\pm 9.999999999 \times 10^{99}$.

(2) An attempt is made to perform functional computations that exceed the input range. (See page 156.)

(3) Improper operation during statistical computations.
(Ex. Attempting to obtain $\bar{x}$ or $x\sigma_n$ without data input.)

(4) The capacity of the numeric value stack or the command stack is exceeded.
(Ex. Entering nineteen successive [(] 's followed by [2] [+] [3] [×] [4])

(5) Even though memory has not been expanded, a memory name such as Z [2] is used. (See page 23 for details on memory.)

(6) Input errors are made.
(Ex. [5] [+] [+] [3] [EXE])

(7) When improper arguments are used in commands or functions that require arguments. (i.e. Input of an argument outside of the range of 0~9 for Sci or Fix.)

The following error messages will be displayed for the operations noted above:

(1)~(3) Ma ERROR
(4) Stk ERROR
(5) Mem ERROR
(6) Syn ERROR
(7) Arg ERROR

Besides these, there are an "Ne ERROR" (nesting error) and a "Go ERROR". These errors mainly occur when using programs. See page 89 or the Error Message Table on page 155.

■ Number of input characters

This unit features a 127-step area for computation execution.
One function comprises one step. Each press of numeric or [+], [−], [×] and [÷] keys comprise one step. Though such operations as [SHIFT] [x!] ([x^{-1}] key) require two key operations, they actually comprise only one function and, therefore, only one step.
These steps can be confirmed using the cursor. With each press of the [⇦] or [⇨] key the cursor is moved one step.
Input characters are limited to 127-steps. Usually the cursor is represented by a blinking "—", but once the 122nd step is reached the cursor changes to a blinking "■". If the "■" appears during a computation, the computation should be divided at some point and performed in two parts.

* *When numeric values or computation commands are input, they appear on the display window from the left. Computational results, however, are displayed from the right.*

■ Graphic and text displays

This unit has a graph display for production of graphs, as well as a text display for production of formulas and commands. These two types of display contents are stored independently of each other.
Switching between graph and text displays is performed using the [G↔T] key. Each press of [G↔T] switches from the current type of display to the other.
Operations to clear the display depend upon the type of display being shown:

Graphs: [SHIFT] [Cls] [EXE]
Text: [AC]

Pressing the [AC] key causes a cleared text display to appear if pressed during a graph display.

■ Display registers

This unit has separate registers for storing text and graph displays. Both of these two registers are unaffected by key operations except for those related to their functions (calculations or [AC] key operation during text display; graph drawing, switching to text display by [G↔T] after clearing graph display by [SHIFT] [Cls] [EXE]).
Since the register stores the previous calculation results, they can be recalled. This is especially useful in the text mode for binary, octal, decimal, and hexadecimal conversions, as well as decimal and significant digit settings.
The following commands will produce previous calculation results:

- Lbl ○
- Dsz ○
- Isz ○
- Mcl
- Hex
- Dec
- Bin
- Oct
- Deg
- Rad
- Gra
- Fix ○
- Sci ○
- Norm
- Rnd
- Scl
- Prog ○

Ex. Perform the calculation 123×456, and then clear the graph display.

* *The [SHIFT] [Cls] [EXE] operation during graph display does not affect the calculation, so the previous calculation result appears on the display.*

[AC] 123 [×] 456 [EXE]	123×456 56088.
[SHIFT] [Cls] [EXE]	123×456 56088. Cls 56088.

A calculation result displayed as shown here is cleared to 0 by pressing [AC], or if the power of the unit is switched OFF (including auto power off).

Corrections

- To make corrections in a formula that is being input, use the [⇦] and [⇨] keys to move to the position of the error and press the correct keys.

Ex. To change an input of 122 to 123:

[1] [2] [2]	122_
[⇦]	122
[3]	123_

Ex. To change an input of cos60 to sin60:

[cos] [6] [0]	cos 60_
[⇦] [⇦] [⇦]	cos 60
[sin]	sin 60

* *If, after making corrections, input of the formula is complete, the answer can be obtained by pressing [EXE]. If, however, more is to be added to the formula, advance the cursor using the [⇨] key to the end of the formula for input.*

- If an unnecessary character has been included in a formula, use the [⇦] and [⇨] keys to move to the position of the error and press the [DEL] key. Each press of [DEL] will delete one command (one step).

Ex. To correct an input of 369××2 to 369×2:

[3] [6] [9] [×] [×] [2]	369××2_
[⇦] [⇦] [DEL]	369×2

- If a character has been omitted from a formula, use the [⇦] and [⇨] keys to move to the position where the character should have been input, and press [SHIFT] followed by the [INS] key. Press [SHIFT] [INS] and insertions can be subsequently performed as desired.

Ex. To correct an input of 2.36^2 to sin2.36^2:

[2] [·] [3] [6] [x^2]	2.36^2_
[⇦] [⇦] [⇦] [⇦] [⇦]	2.36^2
[SHIFT] [INS]	⌜2⌟.36^2
[sin]	sin ⌜2⌟.36^2

* *When [SHIFT] [INS] are pressed, the letter at the insertion position is surrounded by "⌜ ⌟" and blinks. As many letters and/or commands as desired can be inserted at this position until [⇦], [⇨], [⇧], [⇩], or [AC] is pressed. This blinking ⌜ ⌟ is indicated by "⌞ ⌟" in the alphabet mode ([ALPHA] key), while it is indicated by "⌜ ⌝" in the shift mode ([SHIFT]).*

Memory

This unit contains 26 standard memories. Memory names are composed of the 26 letters of the alphabet. Numeric values with 13 digits for a mantissa and 2 digits for an exponent can be stored.

Ex. To store 123.45 in memory A:

123.45 [→] [ALPHA] [A]	123.45→A_
[EXE]	123.45

Values are assigned to a memory using the → key followed by the memory name.

Ex. To store the sum of memory A+78.9 in memory B:

ALPHA A + 78.9 → ALPHA B
EXE

```
A+78.9→B_
     202.35
```

Ex. To add 74.12 to memory B:

ALPHA B + 74.12 → ALPHA B
EXE

```
B+74.12→B_
     276.47
```

- To check the contents of a memory, press the name of the memory to be checked followed by EXE.

ALPHA A EXE

```
     123.45
```

- To clear the contents of a memory (make them 0), proceed as follows:

Ex. To clear the contents of memory A only:

0 → ALPHA A EXE

```
         0.
```

Ex. To clear the contents of all the memories:

SHIFT Mcl
EXE

```
Mcl_
         0.
```

- To store the same numeric value to multiple memories, press SHIFT followed by ~ ((−) key).

Ex. To store a value of 10 in memories A through J:

10 → ALPHA A SHIFT ~ ALPHA J
EXE

```
10→A~J_
        10.
```

Memory expansion

Though there are 26 standard memories, they can be expanded by changing program storage steps to memory. Memory expansion is performed by converting 8 steps to one memory.

* *See page 93 for information on the number of program steps.*

Number of memories	26	27	28	...	36	...	76	...	78
Number of steps	422	414	406	...	342	...	22	...	6

Memory is expanded in units of one. A maximum of 52 memories can be added for a maximum total of 78 (26 + 52). Expansion is performed by pressing MODE, followed by ·, a value representing the size of the expansion, and then EXE.

Ex. To expand the number of memories by 30 to bring the total to 56:

MODE · 30

```
Defm 30_
```

EXE

```
**Defm**
 Program : 0
  Memory : 56
 182 Bytes Free
```

Number of program steps used

Number of memories

Current number of remaining program steps

The number of steps used, number of memories and number of remaining steps are displayed. The number of remaining steps indicates the current unused area, and will differ according to the size of the program stored. To check the current number of memories, press MODE, followed by · and then EXE.

MODE · EXE

```
**Defm**
 Program : 0
  Memory : 56
 182 Bytes Free
```

To initialize the number of memories (to return the number to 26), enter a zero for the value in the memory expansion sequence outlined above.

MODE · 0 EXE

```
**Defm**
 Program : 0
  Memory : 26
  422 Bytes Free
```

* *Though a maximum of 52 memories can be added, if a program has already been stored and the number of remaining steps is less than the desired expansion, an error will be generated. The size of the memory expansion must be equal to or less than the number of steps remaining.*
* *The expansion procedure (MODE · expansion value) can also be stored as a program.*

● Using expanded memories
Expanded memories are used in the same manner as standard memories, and are referred to as Z [1], Z [2], etc. The letter Z followed by a value in brackets indicating the sequential position of the memory is used as the memory name. (Brackets are formed by ALPHA · for " [" and ALPHA EXP for "] ".)
After the number of memories has been expanded by 5, memories Z [1] through Z [5] are available.
The use of these memories is similar to that of a standard computer array, with a subscript being appended to the name. For more information concerning an array, see page 111.

■ Answer (Ans) function

This unit has an answer function that stores the result of the most recent computation. Once a numeric value or numeric formula is entered and EXE is pressed, the result (the answer in the case of the numeric formula) is stored by this function. To recall the stored value, press the Ans key.
When Ans is pressed, "Ans" will appear on the display, and can be used in this form in subsequent calculations.
* *Hereinafter, Ans will be referred to as the Ans memory.*

Ex. $123+456=\underline{579}$
$789-\underline{579}=210$

1 2 3 + 4 5 6 EXE

7 8 9 − Ans EXE

```
123+456
            579.
789-Ans
            210.
```

Numeric values with 13 digits for a mantissa and 2 digits for an exponent can be stored in the Ans memory. The Ans memory is not erased even if the power of the unit is switched OFF. Each time EXE is pressed, the value in the Ans memory is replaced with the new value produced by the computation executed.
When a value is stored to another memory using the EXE key, that value is not stored in the Ans memory.

Ex. Perform computation 78+56=134, then store the value 123 to memory A:

7 8 + 5 6 EXE

Ans EXE ... Checking the content of Ans memory

1 2 3 → ALPHA A EXE

Ans EXE

```
78+56
            134.
Ans
            134.
123→A
            123.
Ans
            134.
```

The Ans memory can be used in the same manner as the other memories, thus making it possible to use it in computation formulas. In multiplication operations, the × immediately before Ans can be omitted.

Ex. $15\times 3=\underline{45}$
$78\times\underline{45}-23=3487$

1 5 × 3 EXE

7 8 Ans − 2 3 EXE

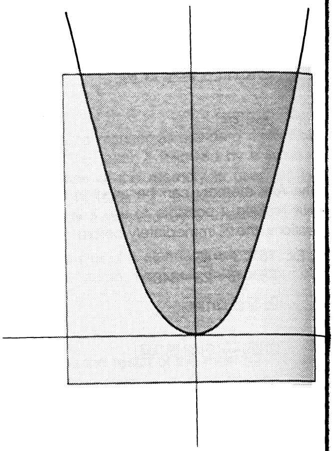
```
15×3
            45.
78Ans-23
          3487.
```

■ Auto power off function

The power of the unit is automatically switched off approximately 6 minutes after the last key operation (except during program computations). Once this occurs, power can be restored either by switching the power of the unit OFF and then ON again, or by pressing the AC key. (Numeric values in the memories, programs or computation modes are unaffected when power is switched off.)

2.

MANUAL COMPUTATIONS



2-1 BASIC COMPUTATIONS

Arithmetic operations

- Arithmetic operations are performed by pressing the keys in the same order as noted in the formula.
- For negative values, press $\boxed{(-)}$ before entering the value.

Example	Operation	Display
$23+4.5-53=-25.5$	23 $\boxed{+}$ 4.5 $\boxed{-}$ 53 $\boxed{\text{EXE}}$	-25.5
$56\times(-12)\div(-2.5)=268.8$	56 $\boxed{\times}$ $\boxed{(-)}$ 12 $\boxed{\div}$ $\boxed{(-)}$ 2.5 $\boxed{\text{EXE}}$	268.8
$12369\times7532\times74103=$ 6.903680613×10^{12} (6903680613000)	12369 $\boxed{\times}$ 7532 $\boxed{\times}$ 74103 $\boxed{\text{EXE}}$	6.903680613 E+12

* *Results greater than 10^{10} (10 billion) or less than 10^{-2} (0.01) are displayed in exponential form.*

Example	Operation	Display
$(4.5\times10^{75})\times(-2.3\times10^{-79})$ $=-1.035\times10^{-3}$ (−0.001035)	4.5 $\boxed{\text{EXP}}$ 75 $\boxed{\times}$ $\boxed{(-)}$ 2.3 $\boxed{\text{EXP}}$ $\boxed{(-)}$ 79 $\boxed{\text{EXE}}$	-1.035 E-03
$(2+3)\times10^{2}=500$	$\boxed{(}$ 2 $\boxed{+}$ 3 $\boxed{)}$ $\boxed{\times}$ 1 $\boxed{\text{EXP}}$ 2 $\boxed{\text{EXE}}$	500
$(1\times10^{5})\div7=14285.71429$	1 $\boxed{\text{EXP}}$ 5 $\boxed{\div}$ 7 $\boxed{\text{EXE}}$	14285.71429
$(1\times10^{5})\div7-14285=$ 0.7142857	1 $\boxed{\text{EXP}}$ 5 $\boxed{\div}$ 7 $\boxed{-}$ 14285 $\boxed{\text{EXE}}$	0.71428571

* *Internal computations are computed in 13 digits for a mantissa, and the result is displayed rounded off to 10 digits.*

- For mixed basic arithmetic operations, multiplication and division are given priority over addition and subtraction.

Example	Operation	Display
$3+\underline{5\times6}=33$	3 $\boxed{+}$ 5 $\boxed{\times}$ 6 $\boxed{\text{EXE}}$	33.
$\underline{7\times8}-\underline{4\times5}=36$	7 $\boxed{\times}$ 8 $\boxed{-}$ 4 $\boxed{\times}$ 5 $\boxed{\text{EXE}}$	36.
$1+2-\underline{3\times4\div5}+6=6.6$	1 $\boxed{+}$ 2 $\boxed{-}$ 3 $\boxed{\times}$ 4 $\boxed{\div}$ 5 $\boxed{+}$ 6 $\boxed{\text{EXE}}$	6.6

■ Parenthesis computations

Example	Operation	Display
100−(2+3)×4=80	100 − (2 + 3) × 4 EXE	80.
2+3×(4+5)=29	2 + 3 × (4 + 5 EXE	29.
* *Closed parentheses occurring immediately before operation of the* EXE *key may be omitted, no matter how many are required.*		
(7−2)×(8+5)=65	(7 − 2) (8+5 EXE	65.
* *A multiplication sign (×) occurring immediately before an open parenthesis can be omitted.*		
10− {2+7×(3+6)} =−55	10 − (2 + 7 (3 + 6 EXE	−55.
* *Henceforth, abbreviated style will not be used in this manual.*		
$\frac{2\times3+4}{5}=(2\times3+4)\div5=2$	(2 × 3 + 4) ÷ 5 EXE	2.
$\frac{5\times6+6\times8}{15\times4+12\times3}=0.8125$	(5 × 6 + 6 × 8) ÷ (15 × 4 + 12 × 3) EXE	0.8125
$(1.2\times10^{19})-\{(2.5\times10^{20})$ $\times\frac{3}{100}\}=4.5\times10^{18}$	1.2 EXP 19 − (2.5 EXP 20 × 3 ÷ 100) EXE	4.5E+18
$\frac{6}{4\times5}=0.3$	6 ÷ (4 × 5) EXE	0.3
* *The above is the same as* 6 ÷ 4 ÷ 5 EXE.		

■ Memory computations

- The contents of memories are not erased when power is switched OFF. They are cleared by pressing SHIFT followed by Mcl (DEL key) and then EXE.

Example	Operation	Display
9.874×7=69.118	9.874 → ALPHA A EXE	9.874
9.874×12=118.488	ALPHA A × 7 EXE	69.118
9.874×26=256.724	ALPHA A × 12 EXE	118.488
9.874×29=286.346	ALPHA A × 26 EXE	256.724
	ALPHA A × 29 EXE	286.346
* *The* → *key is used to input numeric values in memory. (Clearing a memory before input is not required, because the previous value in the memory will be automatically replaced with the new value.)*		

Example	Operation	Display
23+9=32	23 + 9 → ALPHA B EXE	32.
53−6=47	53 − 6 EXE	47.
−)45×2=90	ALPHA B + Ans → ALPHA B EXE	79.
99÷3=33	45 × 2 EXE	90.
Total 22	ALPHA B − Ans → ALPHA B EXE	−11.
	99 ÷ 3 EXE	33.
	ALPHA B + Ans → ALPHA B EXE	22.
12×(2.3+3.4)−5=63.4	2.3 + 3.4 → ALPHA G EXE	5.7
	12 × ALPHA G − 5 EXE	63.4
30×(2.3+3.4+4.5)−15 ×4.5=238.5	4.5 → ALPHA H EXE	4.5
	30 × (ALPHA G + ALPHA H) − 15 ALPHA H EXE	238.5
* *Multiplication signs (×) immediately before memory names can be omitted.*		

■ Specifying the number of decimal places, the number of significant digits and the exponent display

- To specify the number of decimal places, press MODE followed by 7, a value indicating the number of places (0−9) and then EXE.
- To specify the number of significant digits, press MODE followed by 8, a value indicating the number of significant digits (0−9 to set from 1 to 10 digits) and then EXE.
- Pressing the ENG key or SHIFT followed by ← (ENG key) will cause the exponent display for the number being displayed to change in multiples of 3.
- The specified number of decimal places or number of significant digits will not be cancelled until another value or MODE 9 is specified using the sequence: MODE, 9, EXE. (Specified values are not cancelled even if power is switched OFF or an other mode (besides MODE 9) is specified.)
- Even if the number of decimal places and number of significant digits are specified, internal computations are performed in 13 digits for a mantissa, and the displayed value is stored in 10 digits. To convert these values to the specified number of decimal places and significant digits, press SHIFT followed by Rnd (0 key) and then EXE.

* *You cannot specify the display format (Fix, Sci) while the calculator is in the Base-n mode. Such specifications can only be made if you first exit the Base-n mode.*

Example	Operation	Display
100÷6=16.66666666...	100 [÷] 6 [EXE]	16.66666667
	[MODE] [7] [4] [EXE] (Four decimal places specified.)	16.6667
	[MODE] [9] [EXE] (Specification cancelled.)	16.66666667
	[MODE] [8] [5] [EXE] (Five significant digits specified.)	1.6667E+01
	[MODE] [9] [EXE] (Specification cancelled.)	16.66666667
* *Values are displayed rounded off to the place specified.*		
200÷7×14=400	[MODE] [7] [3] [EXE] (Three decimal places specified.)	16.667
	200 [÷] 7 [EXE]	28.571
(Continues computation with 10-digit display.)	[×]	28.57142857×_
	14 [EXE]	400.000
	If the same computation is performed with the specified number of digits:	
	200 [÷] 7 [EXE]	28.571
	(Value stored internally cut off at specified decimal place.) [SHIFT] [Rnd] [EXE]	28.571
	[×]	28.571×_
	14 [EXE]	399.994
	[MODE] [9] [EXE] (Specification cancelled.)	399.994
123m×456=56088m	123 [×] 456 [EXE]	56088.
=56.088km	[ENG]	56.088E+03
78g×0.96=74.88g	78 [×] 0.96 [EXE]	74.88
=0.07488kg	[SHIFT] [ENG]	0.07488E+03

2-2 SPECIAL FUNCTIONS

■ Continuous computation function

Even if computations are concluded with the [EXE] key, the result obtained can be used for further computations. In this case, computations are performed with 10 digits for the mantissa which is displayed.

Ex. 3×4=12 Continuing ÷3.14=

3 [×] 4 [EXE]

```
3×4
                12.
```

(Continuing) [÷] 3.14 [EXE]

```
12.÷3.14
         3.821656051
```

Ex. To compute 1÷3×3

1 [÷] 3 [×] 3 [EXE]

```
1÷3×3
                  1.
```

1 [÷] 3 [EXE]

```
1÷3
        0.3333333333
```

(Continuing) [×] 3 [EXE]

```
0.3333333333×3
        0.9999999999
```

This function can be used with memory and Type A functions (x^2, x^{-1}, $x!$: see page 39), and +, −, x^y, $\sqrt[x]{\ }$, °'" .

Ex. To store the result of 12×45 in memory C:

12 [×] 45 [EXE]

```
12×45
                540.
```

(Continuing) [→] [ALPHA] [C] [EXE]

```
540.→C
                540.
```

Ex. To square the result of 78÷6 (see page 39):

78 [÷] 6 [EXE]

```
78÷6
                 13.
```

(Continuing) [x^2] [EXE]

```
13.²
                169.
```

■ Replay function

- This function stores formulas that have been executed. After execution is complete pressing either the [⇨] or [⇦] key will display the formula executed.
  Pressing [⇨] will display the formula, with the cursor located under the first character.
  Pressing [⇦] will display the formula, with the cursor located at the space following the last character.
  Then using [⇨], [⇦], [⇧] and [⇩] to move the cursor, the formula can be checked and numeric values or commands can be changed for subsequent execution.

Ex.

123 [×] 456 [EXE]

```
123X456
          56088.
```

[⇨]

```
123X456
```

* *The formula appears after clearing the display.*

[EXE]

```
123X456
          56088.
```

[⇦]

```
123X456_
```

- If an error is generated during computation execution, an error check function eliminates the need to clear the error using [AC] and then restarting input from the beginning. Pressing either [⇨] or [⇦] will automatically move the cursor to the point in the formula that generated the error and display it.

Ex. When 14÷0X2.3 is mistakenly entered for 14÷10X2.3:

14 [÷] 0 [×] 2.3 [EXE]

```
14÷0X2.3
   Ma   ERROR
    Step     4
```

[⇨] (or [⇦])

```
14÷0X2.3
```

Error generated here.

[⇦] [SHIFT] [INS] 1 [EXE]

```
14÷10X2.3
            3.22
```

* *As with the number of input characters (see page 19), the replay function can accept input up to 127 steps.*

* *The replay function is cleared when the [AC] key is pressed, when power is switched OFF or when the mode is changed.*

■ Multistatement function

- The multistatement function (using colons to separate formulas or statements) available in program computations can also be used for manual computations.
- The multistatement function allows formulas to be separated by colons to make consecutive, multiple statement computations possible.
- When [EXE] is pressed to execute a formula input using the multistatement format, the formula is executed in order from the beginning.
- Inputting "◢" ([SHIFT] [:]) in place of the colon will display the computational result up to that point during execution.

Ex. 6.9X123=848.7
123÷3.2=38.4375

123 [→] [ALPHA] [A] [:] 6.9 [×]
[ALPHA] [A] [SHIFT] [◢]
[ALPHA] [A] [÷] 3.2 [EXE]

```
123→A:6.9XA◢
A÷3.2
            848.7
          - Disp -
```

The display halted by the ◢ command is represented with —Disp—

[EXE]

```
123→A:6.9XA◢
A÷3.2
            848.7
          38.4375
```

* *Even if "◢" is not input at the end of a formula, the final result will be displayed.*

* *Consecutive computations using multistatements cannot be performed.*

123X456 : +5
└ Invalid

2-3 FUNCTIONAL COMPUTATIONS

■ Angular measurement units

- ● The unit of angular measurement (degrees, radians, grads) is set by pressing [MODE] followed by a value from 4 through 6 and then [EXE].
- ● The numeric value from 4 through 6 specifies degrees, radians and grads respectively.
- ● Once a unit of angular measurement is set, it remains in effect until a new unit is set. Settings are not cleared when power is switched OFF.
- ● The unit of angular measurement can be checked by pressing the [M Disp] key.
- ● You cannot specify the unit of angular measurement (degrees, radians, grads) while the calculator is in the BASE-N mode. Such specifications can only be made if you first exit the BASE-N mode.

Example	Operation	Display
Conversion of 4.25 rad to degrees	[MODE] [4] [EXE] 4.25 [SHIFT] [MODE] [5] [EXE]	243.5070629
Conversion of 1.23 grad to radians	[MODE] [5] [EXE] 1.23 [SHIFT] [MODE] [6] [EXE]	0.01932079482
Conversion of 7.89 degrees to grads	[MODE] [6] [EXE] 7.89 [SHIFT] [MODE] [4] [EXE]	8.766666667
Result displayed in degrees 47.3°+82.5 rad= 4774.20181	[MODE] [4] [EXE] 47.3 [+] 82.5 [SHIFT] [MODE] [5] [EXE]	4774.20181
12.4°+8.3 rad−1.8 gra= 486.33497	12.4 [+] 8.3 [SHIFT] [MODE] [5] [−] 1.8 [SHIFT] [MODE] [6] [EXE]	486.33497
Result displayed in radians 24°6'31"+85.34 rad= 85.76077464	[MODE] [5] [EXE] 24 [° ' "] 6 [° ' "] 31 [° ' "] [SHIFT] [MODE] [4] [+] 85.34 [EXE]	85.76077464
Result displayed in grads 36.9°+41.2 rad= 2663.873462	[MODE] [6] [EXE] 36.9 [SHIFT] [MODE] [4] [+] 41.2 [SHIFT] [MODE] [5] [EXE]	2663.873462

■ Trigonometric functions and inverse trigonometric functions

- ● Be sure to set the unit of angular measurement before performing trigonometric function and inverse trigonometric function computations.

Example	Operation	Display
sin 63°52'41"= 0.897859012	[MODE] [4] [EXE] [sin] 63 [° ' "] 52 [° ' "] 41 [° ' "] [EXE]	0.897859012
$\cos\left(\frac{\pi}{3}\text{ rad}\right)=0.5$	[MODE] [5] [EXE] [cos] [(] [SHIFT] [π] [÷] 3 [)] [EXE]	0.5
tan (−35 gra)= −0.6128007881	[MODE] [6] [EXE] [tan] [(−)] 35 [EXE]	−0.6128007881
2·sin 45° × cos 65°= 0.5976724775	[MODE] [4] [EXE] 2 [×] [sin] 45 [×] [cos] 65 [EXE] Can be omitted.	0.5976724775
$\cot 30^\circ=\frac{1}{\tan 30^\circ}$ =1.732050808	[MODE] [4] [EXE] 1 [÷] [tan] 30 [EXE]	1.732050808
$\sec\left(\frac{\pi}{3}\text{rad}\right)$ $=\frac{1}{\cos(\pi/3\text{ rad})}=2$	[MODE] [5] [EXE] 1 [÷] [cos] [(] [SHIFT] [π] [÷] 3 [)] [EXE]	2
$\text{cosec } 30^\circ=\frac{1}{\sin 30^\circ}$ =2	[MODE] [4] [EXE] 1 [÷] [sin] 30 [EXE]	2
$\sin^{-1} 0.5=30^\circ$ (Determine the value of x when $\sin x=0.5$.)	[SHIFT] [sin⁻¹] 0.5 [EXE] Can be entered as .5	30.
$\cos^{-1}\frac{\sqrt{2}}{2}=$ 0.7853981634 rad $=\frac{\pi}{4}$rad	[MODE] [5] [EXE] [SHIFT] [cos⁻¹] [(] [√] 2 [÷] 2 [)] [EXE] [÷] [SHIFT] [π] [EXE]	0.7853981634 0.25
$\tan^{-1} 0.741=$ 36.53844577° =36°32'18.4"	[MODE] [4] [EXE] [SHIFT] [tan⁻¹] 0.741 [EXE] [SHIFT] [← ° ' "]	36.53844577 36°32'18.4"

* *If the total number of digits for degrees/minutes/seconds exceeds eleven digits, the high-order values (degrees and minutes) are given display priority, and any lower-order values are not displayed. However, the entire value is stored within the unit as a decimal value.*

Example	Operation	Display
$2.5\times(\sin^{-1}0.8-\cos^{-1}0.9)$ $=68°13'13.53''$	2.5 $\boxed{\times}$ $\boxed{(}$ $\boxed{\text{SHIFT}}$ $\boxed{\sin^{-1}}$ 0.8 $\boxed{-}$ $\boxed{\text{SHIFT}}$ $\boxed{\cos^{-1}}$ 0.9 $\boxed{)}$ $\boxed{\text{EXE}}$ $\boxed{\text{SHIFT}}$ $\boxed{\overset{\leftarrow}{\circ\ '\ ''}}$	68°13'13.53"
$\sin 18°\times\cos 0.25\text{rad}=$ 0.2994104044	$\boxed{\sin}$ 18 $\boxed{\times}$ $\boxed{\cos}$ 0.25 $\boxed{\text{SHIFT}}$ $\boxed{\text{MODE}}$ $\boxed{5}$ $\boxed{\text{EXE}}$	0.2994104044

* *The above is computed in radians, and is the same as* $\boxed{\sin}$ 18 $\boxed{\text{SHIFT}}$ $\boxed{\text{MODE}}$ $\boxed{4}$ $\boxed{\times}$ $\boxed{\cos}$ 0.25 $\boxed{\text{EXE}}$.

■ Logarithmic and exponential functions

Example	Operation	Display
$\log 1.23(\log_{10}1.23)=$ 0.08990511144	$\boxed{\log}$ 1.23 $\boxed{\text{EXE}}$	0.08990511144
$\ln 90(\log_e 90)=$ 4.49980967	$\boxed{\ln}$ 90 $\boxed{\text{EXE}}$	4.49980967
$\log 456\div\ln 456=$ 0.4342944819 (log/ln ratio=constant M)	$\boxed{\log}$ 456 $\boxed{\div}$ $\boxed{\ln}$ 456 $\boxed{\text{EXE}}$	0.4342944819
Solve $4^x=64$ $x\cdot\log 4=\log 64$ $x=\dfrac{\log 64}{\log 4}=3$	$\boxed{\log}$ 64 $\boxed{\div}$ $\boxed{\log}$ 4 $\boxed{\text{EXE}}$	3
$10^{1.23}=16.98243652$ (To obtain the antilogarithm of common logarithm 1.23)	$\boxed{\text{SHIFT}}$ $\boxed{10^x}$ 1.23 $\boxed{\text{EXE}}$	16.98243652
$e^{4.5}=90.0171313$ (To obtain the antilogarithm of natural logarithm 4.5)	$\boxed{\text{SHIFT}}$ $\boxed{e^x}$ 4.5 $\boxed{\text{EXE}}$	90.0171313
$10^4\cdot e^{-4}+1.2\cdot 10^{2.3}=$ 422.5878667	$\boxed{\text{SHIFT}}$ $\boxed{10^x}$ 4 $\boxed{\times}$ $\boxed{\text{SHIFT}}$ $\boxed{e^x}$ $\boxed{(-)}$ 4 $\boxed{+}$ 1.2 $\boxed{\times}$ $\boxed{\text{SHIFT}}$ $\boxed{10^x}$ 2.3 $\boxed{\text{EXE}}$	422.5878667
$5.6^{2.3}=52.58143837$	5.6 $\boxed{x^y}$ 2.3 $\boxed{\text{EXE}}$	52.58143837

Example	Operation	Display
$\sqrt[7]{123}\,(=123^{\frac{1}{7}})=$ 1.988647795	7 $\boxed{\sqrt[x]{\ }}$ 123 $\boxed{\text{EXE}}$	1.988647795
$(78-23)^{-12}=$ $1.305111829\times10^{-21}$	$\boxed{(}$ 78 $\boxed{-}$ 23 $\boxed{)}$ $\boxed{x^y}$ $\boxed{(-)}$ 12 $\boxed{\text{EXE}}$	1.305111829E−21
$2+\underline{3\times\sqrt[3]{64}}-4=10$	2 $\boxed{+}$ 3 $\boxed{\times}$ 3 $\boxed{\sqrt[x]{\ }}$ 64 $\boxed{-}$ 4 $\boxed{\text{EXE}}$	10.
$2\times3.4^{(5+6.7)}=3306232.001$	2 $\boxed{\times}$ 3.4 $\boxed{x^y}$ $\boxed{(}$ 5 $\boxed{+}$ 6.7 $\boxed{)}$ $\boxed{\text{EXE}}$	3306232.001

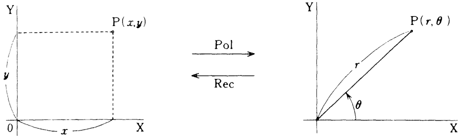
* x^y *and* $\sqrt[x]{\ }$ *given computation priority over* × *and* ÷

■ Hyperbolic functions and inverse hyperbolic functions

Example	Operation	Display
$\sinh 3.6=18.28545536$	$\boxed{\text{hyp}}$ $\boxed{\sin}$ 3.6 $\boxed{\text{EXE}}$	18.28545536
$\cosh 1.23=1.856761057$	$\boxed{\text{hyp}}$ $\boxed{\cos}$ 1.23 $\boxed{\text{EXE}}$	1.856761057
$\tanh 2.5=0.9866142982$	$\boxed{\text{hyp}}$ $\boxed{\tan}$ 2.5 $\boxed{\text{EXE}}$	0.9866142982
$\cosh 1.5-\sinh 1.5=$ 0.2231301601	$\boxed{\text{hyp}}$ $\boxed{\cos}$ 1.5 $\boxed{-}$ $\boxed{\text{hyp}}$ $\boxed{\sin}$ 1.5 $\boxed{\text{EXE}}$	0.2231301601
$=e^{-1.5}$ (Proof of $\cosh x \pm\sinh x=e^{\pm x}$)	(Continuing) $\boxed{\ln}$ $\boxed{\text{Ans}}$ $\boxed{\text{EXE}}$	−1.5
$\sinh^{-1}30=4.094622224$	$\boxed{\text{SHIFT}}$ $\boxed{\text{hyp}}$ $\boxed{\sin^{-1}}$ 30 $\boxed{\text{EXE}}$	4.094622224
$\cosh^{-1}\left(\dfrac{20}{15}\right)=$ 0.7953654612	$\boxed{\text{SHIFT}}$ $\boxed{\text{hyp}}$ $\boxed{\cos^{-1}}$ $\boxed{(}$ 20 $\boxed{\div}$ 15 $\boxed{)}$ $\boxed{\text{EXE}}$	0.7953654612
Determine the value of x when $\tanh 4x=0.88$ $x=\dfrac{\tanh^{-1}0.88}{4}=$ 0.3439419141	$\boxed{\text{SHIFT}}$ $\boxed{\text{hyp}}$ $\boxed{\tan^{-1}}$ 0.88 $\boxed{\div}$ 4 $\boxed{\text{EXE}}$	0.3439419141
$\sinh^{-1}2\times\cosh^{-1}1.5=$ 1.389388923	$\boxed{\text{SHIFT}}$ $\boxed{\text{hyp}}$ $\boxed{\sin^{-1}}$ 2 $\boxed{\times}$ $\boxed{\text{SHIFT}}$ $\boxed{\text{hyp}}$ $\boxed{\cos^{-1}}$ 1.5 $\boxed{\text{EXE}}$	1.389388923
$\sinh^{-1}\left(\dfrac{2}{3}\right)+\tanh^{-1}\left(\dfrac{4}{5}\right)=$ 1.723757406	$\boxed{\text{SHIFT}}$ $\boxed{\text{hyp}}$ $\boxed{\sin^{-1}}$ $\boxed{(}$ 2 $\boxed{\div}$ 3 $\boxed{)}$ $\boxed{+}$ $\boxed{\text{SHIFT}}$ $\boxed{\text{hyp}}$ $\boxed{\tan^{-1}}$ $\boxed{(}$ 4 $\boxed{\div}$ 5 $\boxed{)}$ $\boxed{\text{EXE}}$	1.723757406

Coordinate transformation

- Rectangular coordinates
- Polar coordinates



- Computation results are stored in memories I and J. (Contents of memory I displayed.)

  Pol→I$=r$, J$=\theta$

  Rec→I$=x$, J$=y$
- With polar coordinates, θ can be computed within a range of $-180^\circ < \theta \leqq 180^\circ$. (The computation range is the same with radians or grads.)

Example	Operation	Display
If $x=14$ and $y=20.7$, what are r and θ?	MODE 4 EXE SHIFT Pol(14 SHIFT , 20.7) EXE (Continuing) ALPHA J EXE SHIFT °'"	24.98979792(r) 55°55'42.2"(θ)
If $x=7.5$ and $y=-10$, what are r and θ rad?	MODE 5 EXE SHIFT Pol(7.5 SHIFT , (−) 10) EXE (Continuing) ALPHA J EXE	12.5(r) −0.927295218(θ)
If $r=25$ and $\theta=56$, what are x and y?	MODE 4 EXE SHIFT Rec(25 SHIFT , 56) EXE (Continuing) ALPHA J EXE	13.97982259(x) 20.72593931(y)
If $r=4.5$ and $\theta=\frac{2}{3}\pi$ rad, what are x and y?	MODE 5 EXE SHIFT Rec(4.5 SHIFT , (2 ÷ 3 × SHIFT π)) EXE (Continuing) ALPHA J EXE	−2.25(x) 3.897114317(y)

Other functions ($\sqrt{\ }$, x^2, x^{-1}, $x!$, $\sqrt[3]{\ }$, Ran#, Abs, Int, Frac)

Example	Operation	Display
$\sqrt{2}+\sqrt{5}=3.65028154$	$\sqrt{\ }$ 2 + $\sqrt{\ }$ 5 EXE	3.65028154
$2^2+3^2+4^2+5^2=54$	2 x^2 + 3 x^2 + 4 x^2 + 5 x^2 EXE	54.
$\frac{1}{\frac{1}{3}-\frac{1}{4}}=12$	(3 x^{-1} − 4 x^{-1}) x^{-1} EXE	12.
$8!\ (=1\times2\times3\times\cdots\times8)=40320$	8 SHIFT $x!$ EXE	40320.
$\sqrt[3]{36\times42\times49}=42$	SHIFT $\sqrt[3]{\ }$ (36 × 42 × 49) EXE	42.
Random number generation (pseudorandom number from 0.000 to 0.999)	SHIFT Ran# EXE	(Ex) 0.792
$\sqrt{13^2-5^2}+\sqrt{3^2+4^2}=17$	$\sqrt{\ }$ (13 x^2 − 5 x^2) + $\sqrt{\ }$ (3 x^2 + 4 x^2) EXE	17.
$\sqrt{1-\sin^2 40^\circ}=0.7660444431=\cos 40^\circ$	MODE 4 EXE $\sqrt{\ }$ (1 − (sin 40) x^2) EXE	0.7660444431
(Proof of $\cos\theta=\sqrt{1-\sin^2\theta}$)	(Continuing) SHIFT $\cos^{-1}$ Ans EXE	40.
$\frac{1}{2!}+\frac{1}{4!}+\frac{1}{6!}+\frac{1}{8!}=0.5430803571$	2 SHIFT $x!$ x^{-1} + 4 SHIFT $x!$ x^{-1} + 6 SHIFT $x!$ x^{-1} + 8 SHIFT $x!$ x^{-1} EXE	0.5430803571
What is the absolute value of the common logarithm of $\frac{3}{4}$? $\left\lvert\log\frac{3}{4}\right\rvert=0.1249387366$	SHIFT Abs log (3 ÷ 4) EXE	0.1249387366
What is the integer part of $\frac{7800}{96}$?	SHIFT Int (7800 ÷ 96) EXE	81.
What is the fraction part of $\frac{7800}{96}$?	SHIFT Frac (7800 ÷ 96) EXE	0.25
What is the aliquot part of 2512549139÷2141?	2512549139 ÷ 2141 EXE SHIFT Frac (2512549139 ÷ 2141) EXE	1173540. 0.99953

2-4 BINARY, OCTAL, DECIMAL, HEXADECIMAL COMPUTATIONS

- Binary, octal, decimal and hexadecimal computations, conversions and logical operations are performed in the BASE-N mode (press MODE −).
- The number system (2, 8, 10, 16) is set by respectively pressing Bin, Oct, Dec or Hex, followed by EXE.
- Number systems are specified for specific values by pressing SHIFT, then the number system designator (b, o, d or h), immediately followed by the value.
- General function computations cannot be performed in the BASE-N mode.
- Only integers can be handled in the BASE-N mode. If a computation produces a result that includes a decimal value, the decimal portion is cut off.
- Octal, decimal and hexadecimal computations can be handled up to 32 bits, while binary can be handled up to 16 bits.

Binary	Up to 16 digits
Octal	Up to 11 digits
Decimal	Up to 10 digits
Hexadecimal	Up to 8 digits

- The total range of numbers handled in this mode is 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. If values not valid for the particular number system are used, attach the corresponding designator (b, o, d or h), or an error message will appear.

Valid values

Binary	0, 1
Octal	0, 1, 2, 3, 4, 5, 6, 7
Decimal	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
Hexadecimal	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

- Negative numbers in binary, octal and hexadecimal are expressed as two's complements.
- To distinguish the A, B, C, D, E, F used in the hexadecimal system from standard letters they appear as: 𝔸, 𝔹, ℂ, 𝔻, 𝔼, 𝔽.

- Computation range (in BASE-N mode)

Binary	Positive: $111111111111111 \geqq x \geqq 0$
	Negative: $1111111111111111 \geqq x \geqq 1000000000000000$
Octal	Positive: $17777777777 \geqq x \geqq 0$
	Negative: $37777777777 \geqq x \geqq 20000000000$
Decimal	Positive: $2147483647 \geqq x \geqq 0$
	Negative: $-1 \geqq x \geqq -2147483648$
Hexadecimal	Positive: $7FFFFFFF \geqq x \geqq 0$
	Negative: $FFFFFFFF \geqq x \geqq 80000000$

- You cannot specify the unit of angular measurement (degrees, radians, grads) or the display format (Fix, Sci) while the calculator is in the BASE-N mode. Such specifications can only be made if you first exit the BASE-N mode.

■ Binary,octal, decimal, hexadecimal conversions

Example	Operation	Display
	MODE −	
What are the decimal values for $2A_{16}$ and 274_8?	Dec EXE SHIFT h 2A EXE SHIFT o 274 EXE	 42. 188.
What are the hexadecimal values for 123_{10} and 1010_2?	Hex EXE SHIFT d 123 EXE SHIFT b 1010 EXE	 00000007𝔹 0000000𝔸
What are the octal values for 15_{16} and 1100_2?	Oct EXE SHIFT h 15 EXE SHIFT b 1100 EXE	 00000000025 00000000014
What are the binary values for 36_{10} and $3B7_{16}$?	Bin EXE SHIFT d 36 EXE SHIFT h 3B7 EXE	 0000000000100100 0000001110110111

■ Negative expressions

Example	Operation	Display
	[MODE] [−]	
How is 110010_2 expressed as a negative?	[Bin] [EXE] [Neg] 110010 [EXE]	 1111111111001110
How is 72_8 expressed as a negative?	[Oct] [EXE] [Neg] 72 [EXE]	 37777777706
How is $3A_{16}$ expressed as a negative?	[Hex] [EXE] [Neg] 3A [EXE]	 FFFFFFC6

■ Basic arithmetic operations using binary, octal, decimal and hexadecimal values

Example	Operation	Display
	[MODE] [−]	
$10111_2+11010_2=110001_2$	[Bin] [EXE] 10111 [+] 11010 [EXE]	 0000000000110001
$B47_{16}-DF_{16}=A68_{16}$	[Hex] [EXE] B47 [−] DF [EXE]	 00000A68
$123_8 \times ABC_{16}=37AF4_{16}$ $=228084_{10}$	[SHIFT] [o] 123 [×] ABC [EXE] [Dec] [EXE]	00037AF4 228084
$1F2D_{16}-100_{10}=7881_{10}$ $=1EC9_{16}$	[SHIFT] [h] 1F2D [−] 100 [EXE] [Hex] [EXE]	7881 00001EC9
$7654_8 \div 12_{10}$ $=334.3333333_{10}$ $=516_8$	[Dec] [EXE] [SHIFT] [o] 7654 [÷] 12 [EXE] [Oct] [EXE]	 334 00000000516
* *Computation results are displayed with the decimal portion cut off.*		
$1234+1EF_{16}\div 24_8=2352_8$ $=1258_{10}$	[SHIFT] [d] 1234 [+] [SHIFT] [h] 1EF [÷] 24 [EXE] [Dec] [EXE]	 00000002352 1258
* *For mixed basic arithmetic operations, multiplication and division are given computation priority over addition and subtraction.*		

■ Logical operations

Logical operations are performed through logical product (AND), logical sum (OR), exclusive logical sum (XOR) and negation (NOT).

Example	Operation	Display
	[MODE] [−]	
19_{16} AND $1A_{16}=18_{16}$	[Hex] [EXE] 19 [and] 1A [EXE]	 00000018
1110_2 AND $36_8=1110_2$	[Bin] [EXE] 1110 [and] [SHIFT] [o] 36 [EXE]	 0000000000001110
23_8 OR $61_8=63_8$	[Oct] [EXE] 23 [or] 61 [EXE]	 00000000063
120_{16} OR $1101_2=12D_{16}$	[Hex] [EXE] 120 [or] [SHIFT] [b] 1101 [EXE]	 0000012D
1010_2 AND $(A_{16}$ OR $7_{16})=$ 1010_2	[Bin] [EXE] 1010 [and] [(] [SHIFT] [h] A [or] [SHIFT] [h] 7 [)] [EXE]	 0000000000001010
5_{16} XOR $3_{16}=6_{16}$	[Hex] [EXE] 5 [SHIFT] [xor] 3 [EXE]	 00000006
42_{10} XOR $B_{16}=33_{10}$	[Dec] [EXE] 42 [SHIFT] [xor] [SHIFT] [h] B [EXE]	 33
Negation of 1234_8	[Oct] [EXE] [Not] 1234 [EXE]	 37777776543
Negation of $2FFFED_{16}$	[Hex] [EXE] [Not] 2FFFED [EXE]	 FFD00012

2-5 STATISTICAL COMPUTATIONS

■ Standard deviation

- Standard deviation computations are performed in the SD1 mode. (Press MODE ×.)
- Before beginning computations, the statistical memories are cleared by pressing SHIFT followed by Scl (AC Scl key) and then EXE.
- Individual data is input using DT ($\sqrt[x]{\ }$ key).
- Multiple data of the same value can be input either by repeatedly pressing DT or by entering the data, pressing SHIFT, followed by ;, that represents the number of times the data is repeated, and then DT.
- Standard deviation

$$\sigma_n = \sqrt{\frac{\sum\limits_{i=1}^{n}(x_i-\bar{x})^2}{n}} = \sqrt{\frac{\Sigma x^2-(\Sigma x)^2/n}{n}}$$

(Using the entire data of a finite population to determine the standard deviation for the population.)

$$\sigma_{n-1} = \sqrt{\frac{\sum\limits_{i=1}^{n}(x_i-\bar{x})^2}{n-1}} = \sqrt{\frac{\Sigma x^2-(\Sigma x)^2/n}{n-1}}$$

(Using sample data for a population to determine the standard deviation for the population.)

- Mean

$$\bar{x} = \frac{\sum\limits_{i=1}^{n} x_i}{n} = \frac{\Sigma x}{n}$$

* *The values for n, Σx, and Σx^2 are stored in memories W, V, and U respectively, and can be obtained by pressing* ALPHA *followed by the memory name and then* EXE *(i.e.* ALPHA W EXE*).*

<table>
<tr><th>Example</th><th>Operation</th><th>Display</th></tr>
<tr><td>Data 55, 54, 51, 55, 53, 53, 54, 52</td><td>MODE ×
SHIFT Scl EXE (Memory clear)
55 DT 54 DT 51 DT 55 DT 53 DT DT 54 DT 52 DT</td><td>52.</td></tr>
<tr><td colspan="3">* <i>Results can be obtained in any order desired.</i></td></tr>
<tr><td></td><td>(Standard deviation σ_n) SHIFT $x\sigma n$ EXE</td><td>1.316956719</td></tr>
<tr><td></td><td>(Standard deviation σ_{n-1}) SHIFT $x\sigma n$-1 EXE</td><td>1.407885953</td></tr>
<tr><td></td><td>(Mean $\bar{x}$) SHIFT $\bar{x}$ EXE</td><td>53.375</td></tr>
<tr><td></td><td>(Number of data n) ALPHA W EXE</td><td>8.</td></tr>
<tr><td></td><td>(Sum total Σx) ALPHA V EXE</td><td>427.</td></tr>
<tr><td></td><td>(Sum of squares Σx^2) ALPHA U EXE</td><td>22805.</td></tr>
<tr><td rowspan="5">What is deviation of the unbiased variance, the difference between each datum and the mean of the above data?</td><td>(Continuing) SHIFT $x\sigma n$-1 x^2 EXE</td><td>1.982142857</td></tr>
<tr><td>55 − SHIFT $\bar{x}$ EXE</td><td>1.625</td></tr>
<tr><td>54 − SHIFT $\bar{x}$ EXE</td><td>0.625</td></tr>
<tr><td>51 − SHIFT $\bar{x}$ EXE</td><td>−2.375</td></tr>
<tr><td>⋮</td><td>⋮</td></tr>
<tr><td rowspan="9">What is $\bar{x}$ and $x\sigma_{n-1}$ for the following table?
<table>
<tr><th>Class No.</th><th>Value</th><th>Frequency</th></tr>
<tr><td>1</td><td>110</td><td>10</td></tr>
<tr><td>2</td><td>130</td><td>31</td></tr>
<tr><td>3</td><td>150</td><td>24</td></tr>
<tr><td>4</td><td>170</td><td>2</td></tr>
<tr><td>5</td><td>190</td><td>3</td></tr>
</table></td><td>SHIFT Scl EXE</td><td></td></tr>
<tr><td>110 SHIFT ; 10 DT</td><td>110.</td></tr>
<tr><td>130 SHIFT ; 31 DT</td><td>130.</td></tr>
<tr><td>150 SHIFT ; 24 DT</td><td>150.</td></tr>
<tr><td>170 DT DT</td><td>170.</td></tr>
<tr><td>190 DT DT DT</td><td>190.</td></tr>
<tr><td>ALPHA W EXE</td><td>70.</td></tr>
<tr><td>SHIFT $\bar{x}$ EXE</td><td>137.7142857</td></tr>
<tr><td>SHIFT $x\sigma n$-1 EXE</td><td>18.42898069</td></tr>
</table>

* Erroneous data clearing/correction I (correct data operation: 51 DT)

① If 50 DT is entered, enter correct data after pressing CL (x^y key).

② If 49 DT was input a number of entries previously, enter correct data after pressing 49 CL.

* Erroneous data clearing/correction II (correct data operation: 130 [SHIFT] [;] 31 [DT])

① If 120 [SHIFT] [;] is entered, enter correct data after pressing [AC].
② If 120 [SHIFT] [;] 31 is entered, enter correct data after pressing [AC].
③ If 120 [SHIFT] [;] 30 [DT] is entered, enter correct data after pressing [CL].
④ If 120 [SHIFT] [;] 30 [DT] was entered previously, enter correct data after pressing 120 [SHIFT] [;] 30 [CL].

■ Regression computation

- ● Regression computations are performed in the LR1 mode. (Press [MODE] [÷].)
- ● Before beginning computations, the tabulation memories are cleared by pressing [SHIFT] followed by [Scl] and then [EXE].
- ● Individual data are entered as x data [SHIFT] [;] y data [DT].
- ● Multiple data of the same value can be entered by repeatedly pressing [DT]. This operation can also be performed by entering x data [SHIFT] [;] y data [SHIFT] [;] followed by a value representing the number of times the data is repeated, and then [DT].
- ● If only x data is repeated (x data having the same value), enter [SHIFT] [;] y data [DT] or [SHIFT] [;] y data [SHIFT] [;] followed by a value representing the number of times the data is repeated, and then [DT].
- ● If only y data is repeated (y data having the same value), enter x data [DT] or x data [SHIFT] [;] followed by a value representing the total number of times the data is repeated, and then [DT].
- ● The regression formula is $y=\mathrm{A}+\mathrm{B}x$, and constant term A and regression coefficient B are computed using the following formulas:

Constant term of regression formula

$$\mathrm{A}=\frac{\Sigma y-\mathrm{B}\cdot\Sigma x}{n}$$

Regression coefficient of regression formula

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

- ● Estimated values $\hat{x}$, and $\hat{y}$ based on the regression formula can be computed using the following formulas:

$$\hat{y}=\mathrm{A}+\mathrm{B}x \qquad \hat{x}=\frac{y-\mathrm{A}}{\mathrm{B}}$$

(To obtain the estimated value $\hat{y}$, [SHIFT] [$\hat{y}$] is used, and to obtain estimated value $\hat{x}$, [SHIFT] [$\hat{x}$] is used.)

- ● The correlation coefficient r for input data can be computed using the following formula:

$$r=\frac{n\cdot\Sigma xy-\Sigma x\cdot\Sigma y}{\sqrt{\{n\cdot\Sigma x^2-(\Sigma x)^2\}\{n\cdot\Sigma y^2-(\Sigma y)^2\}}}$$

** The values for $n, \Sigma x, \Sigma x^2, \Sigma xy, \Sigma y$, and Σy^2 are stored in memories W, V, U, R, Q and P respectively, and can be obtained by pressing [ALPHA] followed by the memory name and then [EXE] (i.e. [ALPHA] [W] [EXE]).*

◆ Linear regression

Temp.	Length
10℃	1003mm
15	1005
20	1010
25	1011
30	1014

Example	Operation	Display
• Temperature and the length of a steel bar	[MODE] [÷]	
	[SHIFT] [Scl] [EXE] (Memory clear)	
	10 [SHIFT] [;] 1003 [DT]	10.
	15 [SHIFT] [;] 1005 [DT]	15.
	20 [SHIFT] [;] 1010 [DT]	20.
	25 [SHIFT] [;] 1011 [DT]	25.
	30 [SHIFT] [;] 1014 [DT]	30.
Using this table the regression formula and correlation coefficient can be obtained. Based on the coefficient formula, the length of the steel bar at 18°C and the temperature at 1000mm can be estimated. Furthermore, the critical coefficient (r^2) and covariance $\left(\frac{\Sigma xy-n\cdot\bar{x}\cdot\bar{y}}{n-1}\right)$ can also be computed.	(Constant term A) [SHIFT] [A] [EXE]	997.4
	(Regression coefficient B) [SHIFT] [B] [EXE]	0.56
	(Correlation coefficient r) [SHIFT] [r] [EXE]	0.9826073689
	(Length at 18°C) 18 [SHIFT] [$\hat{y}$] [EXE]	1007.48
	(Temperature at 1000mm) 1000 [SHIFT] [$\hat{x}$] [EXE]	4.642857142
	(Critical coefficient) [SHIFT] [r] [x^2] [EXE]	0.9655172414
	(Covariance) [(] [ALPHA] [R] [−] [ALPHA] [W] [×] [SHIFT] [$\bar{x}$] [×] [SHIFT] [$\bar{y}$] [)] [÷] [(] [ALPHA] [W] [−] 1 [)] [EXE]	35.

* Erroneous data clearing/correction (correct data operation: 10 [SHIFT] [;] 1003 [DT])

① If 11 [SHIFT] [;] 1003 is entered, enter correct data after pressing [AC].

② If 11 SHIFT , 1003 DT is entered, enter correct data after pressing CL.

③ If 11 SHIFT , 1003 DT was entered previously, enter correct data after pressing 11 SHIFT , 1003 CL.

◆ Logarithmic regression

- The regression formula is $y=A+B\cdot\ln x$. Enter the x data as the logarithm (ln) of x, and the y data inputs the same as that for linear regression.
- Estimated values $\hat{x}$, and $\hat{y}$ based on the regression formula can be computed using the following formulas:

$$\hat{y}=A+B\cdot\ln x \qquad \hat{x}=\exp\left(\frac{y-A}{B}\right)$$

- The same operation as with linear regression can be used to obtain the regression coefficient and for making corrections. To obtain the estimated value $\hat{y}$, ln x SHIFT $\hat{y}$ EXE is used, and to obtain estimated value $\hat{x}$, y SHIFT $\hat{x}$ EXE SHIFT e^x Ans EXE is used. Furthermore, Σx, Σx^2, and Σxy are obtained as $\Sigma\ln x$, $\Sigma(\ln x)^2$, and $\Sigma\ln xy$ respectively.

Example x_i	Example y_i	Operation	Display
		MODE ÷	
		SHIFT Scl EXE	
29	1.6	ln 29 SHIFT , 1.6 DT	3.36729583
50	23.5	ln 50 SHIFT , 23.5 DT	3.912023005
74	38.0	ln 74 SHIFT , 38.0 DT	4.304065093
103	46.4	ln 103 SHIFT , 46.4 DT	4.634728988
118	48.9	ln 118 SHIFT , 48.9 DT	4.770684624
		(Constant term A) SHIFT A EXE	−111.1283976
		(Regression coefficient B) SHIFT B EXE	34.0201475
		(Correlation coefficient r) SHIFT r EXE	0.9940139466
		($\hat{y}$ when $xi=80$) ln 80 SHIFT $\hat{y}$ EXE	37.94879482
		($\hat{x}$ when $yi=73$) 73 SHIFT $\hat{x}$ EXE SHIFT e^x Ans EXE	224.1541313

Through logarithmic regression of the above data, the regression formula and correlation coefficient are obtained. Furthermore, respective estimated values $\hat{y}$ and $\hat{x}$ can be obtained for $xi=80$ and $yi=73$ using the regression formula.

◆ Exponential regression

- The regression formula is $y=A\cdot e^{B\cdot x}$($\ln y=\ln A+Bx$). Enter the y data as the logarithm of y(ln), and the x data the same as that for linear regression.
- Estimated values $\hat{x}$, and $\hat{y}$ based on the regression formula can be computed using the following formulas:

$$\hat{y}=A\cdot e^{Bx} \qquad \hat{x}=\frac{\ln y-\ln A}{B}$$

- Correction is performed the same as in linear regression. Constant term A is obtained by SHIFT e^x SHIFT A EXE, estimated value $\hat{y}$ is obtained by x SHIFT $\hat{y}$ EXE SHIFT e^x Ans EXE, and estimated value $\hat{x}$ is obtained by ln y SHIFT $\hat{x}$ EXE. Σy, Σy^2 and Σxy are obtained by $\Sigma\ln y$, $\Sigma(\ln y)^2$ and $\Sigma x\cdot\ln y$ respectively.

Example x_i	Example y_i	Operation	Display
		MODE ÷	
		SHIFT Scl EXE	
6.9	21.4	6.9 SHIFT , ln 21.4 DT	6.9
12.9	15.7	12.9 SHIFT , ln 15.7 DT	12.9
19.8	12.1	19.8 SHIFT , ln 12.1 DT	19.8
26.7	8.5	26.7 SHIFT , ln 8.5 DT	26.7
35.1	5.2	35.1 SHIFT , ln 5.2 DT	35.1
		(Constant term A) SHIFT e^x SHIFT A EXE	30.49758743
		(Regression coefficient B) SHIFT B EXE	−0.04920370831
		(Correlation coefficient r) SHIFT r EXE	−0.997247352
		($\hat{y}$ when $xi=16$) 16 SHIFT $\hat{y}$ EXE SHIFT e^x Ans EXE	13.87915739
		($\hat{x}$ when $yi=20$) ln 20 SHIFT $\hat{x}$ EXE	8.574868046

Through exponential regression of the above data, the regression formula and correlation coefficient are obtained. Furthermore, the regression formula is used to obtain the respective estimated values $\hat{y}$ and $\hat{x}$ when $xi=16$ and $yi=20$.

Power regression

- The regression formula is $y=\mathrm{A}\cdot x^{\mathrm{B}}(\ln y=\ln\mathrm{A}+\mathrm{B}\ln x)$. Enter both data x and y as logarithms (ln).
- Estimated values $\hat{x}$, and $\hat{y}$ based on the regression formula can be computed using the following formulas:

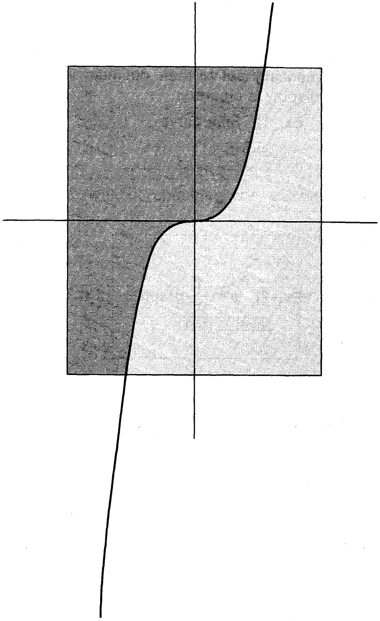
$$\hat{y}=\mathrm{A}\cdot x^{\mathrm{B}} \qquad \hat{x}=\exp\left(\frac{\ln y-\ln \mathrm{A}}{\mathrm{B}}\right)$$

- Correction is performed the same as in linear regression. Constant term A is obtained by $\boxed{\text{SHIFT}}\ \boxed{e^x}\ \boxed{\text{SHIFT}}\ \boxed{\text{A}}\ \boxed{\text{EXE}}$, estimated value $\hat{y}$ is obtained by $\boxed{\ln}\ x\ \boxed{\text{SHIFT}}\ \boxed{\hat{y}}\ \boxed{\text{EXE}}\ \boxed{\text{SHIFT}}\ \boxed{e^x}\ \boxed{\text{Ans}}\ \boxed{\text{EXE}}$, and estimated value $\hat{x}$ is obtained by $\boxed{\ln}\ y\ \boxed{\text{SHIFT}}\ \boxed{\hat{x}}\ \boxed{\text{EXE}}\ \boxed{\text{SHIFT}}\ \boxed{e^x}\ \boxed{\text{Ans}}\ \boxed{\text{EXE}}$. Σx, Σx^2, Σy, Σy^2 and Σxy are obtained by $\Sigma\ln x$, $\Sigma(\ln x)^2$, $\Sigma\ln y$, $\Sigma(\ln y)^2$ and $\Sigma\ln x\cdot\ln y$ respectively.

<table>
<tr><th>Example</th><th>Operation</th><th>Display</th></tr>
<tr><td rowspan="12"><table><tr><th>x_i</th><th>y_i</th></tr><tr><td>28</td><td>2410</td></tr><tr><td>30</td><td>3033</td></tr><tr><td>33</td><td>3895</td></tr><tr><td>35</td><td>4491</td></tr><tr><td>38</td><td>5717</td></tr></table></td><td>$\boxed{\text{MODE}}\ \boxed{\div}$</td><td></td></tr>
<tr><td>$\boxed{\text{SHIFT}}\ \boxed{\text{Scl}}\ \boxed{\text{EXE}}$</td><td></td></tr>
<tr><td>$\boxed{\ln}$ 28 $\boxed{\text{SHIFT}}\ \boxed{,}\ \boxed{\ln}$ 2410</td><td></td></tr>
<tr><td>$\boxed{\text{DT}}$</td><td>3.33220451</td></tr>
<tr><td>$\boxed{\ln}$ 30 $\boxed{\text{SHIFT}}\ \boxed{,}\ \boxed{\ln}$ 3033</td><td></td></tr>
<tr><td>$\boxed{\text{DT}}$</td><td>3.401197382</td></tr>
<tr><td>$\boxed{\ln}$ 33 $\boxed{\text{SHIFT}}\ \boxed{,}\ \boxed{\ln}$ 3895</td><td></td></tr>
<tr><td>$\boxed{\text{DT}}$</td><td>3.496507561</td></tr>
<tr><td>$\boxed{\ln}$ 35 $\boxed{\text{SHIFT}}\ \boxed{,}\ \boxed{\ln}$ 4491</td><td></td></tr>
<tr><td>$\boxed{\text{DT}}$</td><td>3.555348061</td></tr>
<tr><td>$\boxed{\ln}$ 38 $\boxed{\text{SHIFT}}\ \boxed{,}\ \boxed{\ln}$ 5717</td><td></td></tr>
<tr><td>$\boxed{\text{DT}}$</td><td>3.63758616</td></tr>
<tr><td rowspan="5">Through power regression of the above data, the regression formula and correlation coefficient are obtained.
Furthermore, the regression formula is used to obtain the respective estimated values $\hat{x}$ and $\hat{y}$ when $xi=40$ and $yi=1000$.</td><td>(Constant term A)
$\boxed{\text{SHIFT}}\ \boxed{e^x}\ \boxed{\text{SHIFT}}\ \boxed{\text{A}}\ \boxed{\text{EXE}}$</td><td>0.2388010724</td></tr>
<tr><td>(Regression coefficient B)
$\boxed{\text{SHIFT}}\ \boxed{\text{B}}\ \boxed{\text{EXE}}$</td><td>2.771866153</td></tr>
<tr><td>(Correlation coefficient r)
$\boxed{\text{SHIFT}}\ \boxed{\text{r}}\ \boxed{\text{EXE}}$</td><td>0.9989062542</td></tr>
<tr><td>($\hat{y}$ when $xi=40$)
$\boxed{\ln}$ 40 $\boxed{\text{SHIFT}}\ \boxed{\hat{y}}\ \boxed{\text{EXE}}\ \boxed{\text{SHIFT}}$
$\boxed{e^x}\ \boxed{\text{Ans}}\ \boxed{\text{EXE}}$</td><td>6587.67458</td></tr>
<tr><td>($\hat{x}$ when $yi=1000$)
$\boxed{\ln}$ 1000 $\boxed{\text{SHIFT}}\ \boxed{\hat{x}}\ \boxed{\text{EXE}}\ \boxed{\text{SHIFT}}$
$\boxed{e^x}\ \boxed{\text{Ans}}\ \boxed{\text{EXE}}$</td><td>20.2622568</td></tr>
</table>

3.

GRAPHS



3-1 BUILT-IN FUNCTION GRAPHS

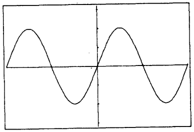
The COMP mode of the RUN mode should be used when graphing functions. Some graphs can be produced in the SD and LR modes, but certain graphs cannot be produced in these modes. The BASE-N mode cannot be used for graphs. This unit contains a total of 20 built-in graphs making it possible to produce the graphs of basic functions.

- sin
- cos
- tan
- $\sin^{-1}$
- $\cos^{-1}$
- $\tan^{-1}$
- sinh
- cosh
- tanh
- $\sinh^{-1}$
- $\cosh^{-1}$
- $\tanh^{-1}$
- $\sqrt{\ }$
- x^2
- log
- ln
- 10^x
- e^x
- x^{-1}
- $\sqrt[3]{\ }$

Any time a built-in graph is executed, the ranges (see page 54) are automatically set to their optimum values, and any graph previously on the display is cleared.

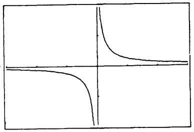
Ex. 1) Sine curve

MODE +
Graph sin EXE



Ex. 2) $y=\frac{1}{x}$ graph

Graph x^{-1} EXE



Overwriting built-in function graphs

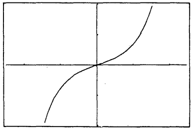
Two or more different built-in function graphs can be written together on the same display. Since the range for the first graph is automatically set, all subsequent graphs on the same display are produced according to the range of the first graph.

The first graph is produced by using the previously mentioned operation (Graph [function key] EXE).

Subsequent graphs are produced using the variable X in the operation Graph [function key] ALPHA X EXE (X : + key). By inputting ALPHA X after the function key, the range is unchanged and the next graph is produced without clearing the existing display. (See page 59 for details.)

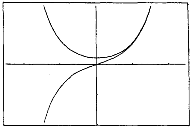
Ex. Overwrite the graph for y=cosh x on the graph for y sinh x.
First, draw the graph for y=sinh x.

Graph hyp sin EXE



Next, draw the graph for y=cosh$x$ without changing the existing range.

Graph hyp cos ALPHA X EXE



⟨Note⟩
Built-in function graphs cannot be used in multistatements (see page 33) and cannot be written into programs.

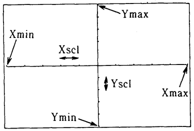
3-2 USER GENERATED GRAPHS

Built-in function graphs can also be used in combination with each other. Graphing a formula such as $y=2x^2+3x-5$ makes it possible to visually represent the solution.
Unlike built-in functions, the ranges of user generated graphs are not set automatically, so graphs produced outside of the display range do not appear on the display.

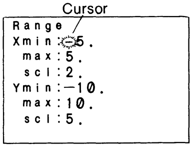
■ Ranges

The ranges of the x and y-axes, as well as the scale (distance between points) for both axes can be set or checked using the $\boxed{\text{Range}}$ key.

- Ranges contents
  Ranges consist of Xmin (x-axis minimum value), Xmax (x-axis maximum value), Xscl (x-axis scale), Ymin (y-axis minimum value), Ymax (y-axis maximum value), and Yscl (y-axis scale).



- Range display
  Ranges are displayed as shown on the right when the $\boxed{\text{Range}}$ key is pressed. The range value at the cursor position can be changed.



```
Range
Xmin:-5.
  max:5.
  scl:2.
Ymin:-10.
  max:10.
  scl:5.
```

* Values shown here are only an example.
Actual values may differ.

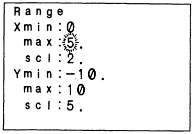
- Range setting
  Range settings are made from the current cursor position and proceed in the order of Xmin→Xmax→Xscl→Ymin→Ymax→Yscl. Input a numeric value at the cursor position and then press $\boxed{\text{EXE}}$. Any value input while the cursor is at the first (extreme left) digit of the displayed value will replace the displayed value when $\boxed{\text{EXE}}$ is pressed.
  If the $\boxed{⇨}$ key is used to move the cursor to the second or subsequent digit of the displayed value, only the portion of the displayed value starting from the cursor position will be affected by the new input when $\boxed{\text{EXE}}$ is pressed.

Here, let's try changing the currently set range values to those listed below:

Xmin	→	0	Ymin	→	−5
Xmax	→	5	Ymax	→	15
Xscl	→	1	Yscl	→	5

① Input 0 for Xmin.

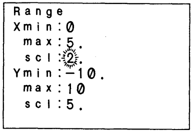
0 $\boxed{\text{EXE}}$



```
Range
Xmin:0
  max:5.
  scl:2.
Ymin:-10.
  max:10
  scl:5.
```

② The Xmax value is the same, so simply press $\boxed{\text{EXE}}$.

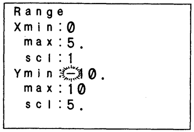
$\boxed{\text{EXE}}$
($\boxed{⇩}$ key can also be used.)



```
Range
Xmin:0
  max:5.
  scl:2.
Ymin:-10.
  max:10
  scl:5.
```

③ Input 1 for Xscl.

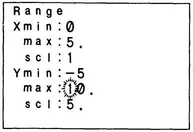
1 $\boxed{\text{EXE}}$



```
Range
Xmin:0
  max:5.
  scl:1
Ymin:-10.
  max:10
  scl:5.
```

④ To change Ymin to −5, use the $\boxed{⇨}$ key to move the cursor one digit to the right and input 5.

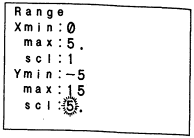
$\boxed{⇨}$ 5 $\boxed{\text{EXE}}$



```
Range
Xmin:0
  max:5.
  scl:1
Ymin:-5
  max:10.
  scl:5.
```

⑤ To change Ymax to 15, use the $\boxed{\Rightarrow}$ key to move the cursor one digit to the right and input 5.

$\boxed{\Rightarrow}$ 5 $\boxed{\text{EXE}}$



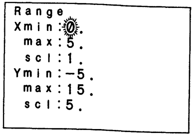
```
Range
Xmin:0
 max:5.
 scl:1
Ymin:-5
 max:15
 scl:5.
```

⑥ The Yscl value is the same, so simply press $\boxed{\text{EXE}}$.

$\boxed{\text{EXE}}$

Once all settings are complete, the display that was shown before pressing the $\boxed{\text{Range}}$ key is retrieved.
Press the $\boxed{\text{Range}}$ key again to confirm whether settings are correct.

$\boxed{\text{Range}}$



```
Range
Xmin:0.
 max:5.
 scl:1.
Ymin:-5.
 max:15.
 scl:5.
```

The $\boxed{\Uparrow}$ and $\boxed{\Downarrow}$ keys can be used to move the cursor from line to line in the range display without affecting the range values. The cursor can only be moved upwards as far as Xmin, and downwards as far as Yscl. Press $\boxed{\text{Range}}$ to return to the display that was shown before entering the range display.

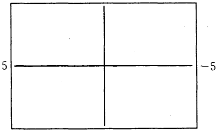
- * *The input range for graph ranges is −9.9999E+98 through 9.99999E+98.*
- * *Only numeric value keys from $\boxed{0}$ through $\boxed{9}$, $\boxed{.}$, $\boxed{\text{EXP}}$, $\boxed{(-)}$, $\boxed{\Leftarrow}$, $\boxed{\Rightarrow}$, $\boxed{\Uparrow}$, $\boxed{\Downarrow}$, and $\boxed{\text{Range}}$ can be used during range display. Other key operation is ignored.*
(Use the $\boxed{(-)}$ key for negative value input.)
- * *To completely change an existing range setting, ensure that the cursor is located at the first digit (all the way to the left) of the displayed value. If the cursor has been moved to another digit of the value, only the portion of the value from the cursor position (to the right) will be changed. The portion of the value to the left of the cursor will remain unchanged.*

Ex.

	−25
$\boxed{\Rightarrow}$	−25
3	−35
$\boxed{\text{EXE}}$	−3

- * *Values up to nine significant digits can be input.*
Values less than 10^{-2} and equal to or greater than 10^{8} are displayed with a 6-digit mantissa (including negative sign) and a 2-digit exponent.
- * *If input is improper (outside the allowable calculation range or inputting only a negative sign), the existing value will remain unchanged. (The improper input, however, will be temporarily displayed.)*
- * *Inputting 0 for Xscl or Yscl does not set any scale.*
- * *Inputting a maximum value that is less than the minimum value will reverse the respective axis.*

Ex. Xmin : 5
Xmax : −5



* *If the maximum and minimum values of an axis are equal, an error (Ma ERROR) will be generated when an attempt is made to produce a graph.*
* *When a range setting is used that does not allow display of the axes, the scale for the y-axis is indicated on either the left or right edge of the display, while that for the x-axis is indicated on either the top or bottom edge. (In both cases, the location of the scale is the edge which is closest to the origin (0, 0)).*
* *When range values are changed (reset), the graph display is cleared and the newly set axes only are displayed.*
* *Range settings may cause irregular scale spacing.*
* *If the range is set too wide, the graph produced may not fit on the display.*
* *Points of deflection sometimes exceed the capabilities of the display with graphs that change drastically as they approach the point of deflection.*
* *An Ma ERROR may be generated when a range value is specified that exceeds the allowable range.*

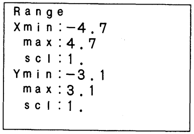
Ex. Xmin 9.E99
Xmax 9.9E99
Xscl 1.E99 ⇒ Falls outside of range.
⋮

* *An Ma ERROR is generated when ranges are extremely narrow.*

● **Range reset**

Range values are reset to their initial values by pressing SHIFT DEL during range display.

Range (Not required when range display is already being shown.)
SHIFT DEL



```
Range
Xmin:-4.7
 max:4.7
 scl:1.
Ymin:-3.1
 max:3.1
 scl:1.
```

⟨Reference⟩

Range settings are performed within programs using the following format:

Range Xmin value, Xmax value, Xscl value, Ymin value, Ymax value, Yscl value

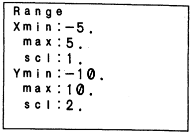
Up to six data items are programmed after the Range command. When less than six items are programmed, range setting is performed in the order from the beginning of the above format.

User generated function graphs

After performing range settings, user generated graphs can be drawn simply by entering the function (formula) after pressing Graph.

Here, let's try drawing a graph for $y=2x^2+3x-4$.

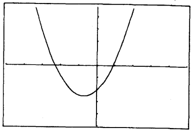
Set the ranges to the values shown below.



```
Range
Xmin:-5.
 max:5.
 scl:1.
Ymin:-10.
 max:10.
 scl:2.
```

Input the functional formula after pressing the Graph key.

Graph 2 ALPHA X x^2 +
3 ALPHA X − 4 EXE



The result produces a visual representation of the formula.

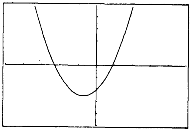
Function graph overwrite

Two or more function graphs can be overwritten which makes it easy to determine intersection points and solutions that satisfy all the equations.

Ex. **Here, let's find the intersection points of the previously used $y=2x^2+3x-4$ and $y=2x+3$.**

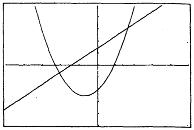
First, clear the graph screen in preparation for the first graph.

SHIFT Cls EXE
Graph 2 ALPHA X x^2 +
3 ALPHA X − 4 EXE



Next, overwrite the graph for $y=2x+3$.

Graph 2 ALPHA X ⊞ 3 EXE



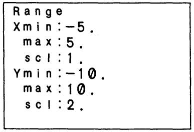
In this way it can be easily seen that there are two intersections for the two function graphs. The approximate coordinates for these two intersections can be found using the trace function described in the following section.

* *Be sure to input variable X (ALPHA X) into the function when using built-in graphs for overwrite.*
If variable X is not included in the second formula, the second graph is produced after clearing the first graph.

■ Trace function

The pointer (blinking dot) can be moved using the cursor keys (⇦ ⇨) to determine the x and y coordinates of any point on a graph.
After a graph is produced on the display, press SHIFT Trace and the point will appear at the extreme left plot of the graph. The x-coordinate value (X=. . . .) will appear on the bottom line of the display. The pointer can be moved using the ⇦ and ⇨ cursor keys, and the x-coordinate value changes as the pointer moves. To change from the x-coordinate to the y-coordinate value, press SHIFT X↔Y. The displayed coordinate switches between x and y with each press of SHIFT X↔Y.

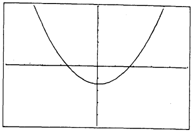
Ex. Determine the points of intersection of the graphs for $y=x^2-3$ and $y=-x+2$.
The range values should be set as follows:



```
Range
Xmin:-5.
 max:5.
 scl:1.
Ymin:-10.
 max:10.
 scl:2.
```

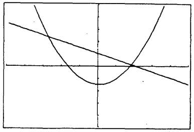
First, draw the graph for $y=x^2-3$.

Graph ALPHA X x^2 ⊟ 3 EXE



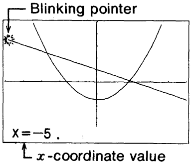
Next, draw the graph for $y=-x+2$.

Graph (−) ALPHA X ⊞ 2 EXE



Finally, let's use the trace function.

SHIFT Trace

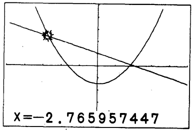


Blinking pointer

x-coordinate value

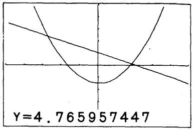
The pointer appears at the extreme left plot of the graph. The ⇨ key moves the pointer to the right along the graph. Each press of ⇨ moves the pointer one point, while holding it down causes continuous movement.

⇨ ~
(Hold down)



Hold [⇨] down until the pointer reaches the intersection of the two graphs. Note the x-coordinate value, and then press [SHIFT] [x↔Y] for the y-coordinate value.

[SHIFT] [x↔Y]



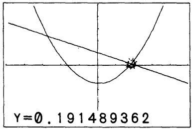
Y=4.765957447

In this way, it can be determined that the coordinates of the first intersection are $x=-2.765957447$ and $y=4.765957447$.

- *The pointer does not move at the fixed distance because the distance is located along the dots of the display. Therefore, the x-y coordinates for the point of intersection are approximate values.*

Similarly, press [⇨] to move the pointer to the next point of intersection.

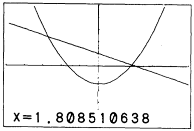
[⇨] ~



Y=0.191489362

This time, press [SHIFT] [x↔Y] to display the x-coordinate value.

[SHIFT] [x↔Y]



X=1.808510638

Using the operations outlined above, the approximate x-y coordinates for points along graphs can be obtained.

- *The trace function can only be used immediately after a graph is drawn. This function cannot be used if other calculations or operations (except [M Disp], [Range], or [G↔T]) have been employed after a graph has been drawn.*
- *The x-y coordinate values at the bottom of the display consist of a 10-digit mantissa or a 5-digit mantissa plus a 2-digit exponent.*

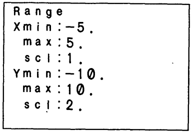
- *The trace function cannot be written into a program.*
- *The trace function can be used during a "—DISP—" display.*
- *When the format [Graph] formula ◢ [Graph] formula [EXE] is executed and a graph is drawn by pressing [EXE] directly after executing the trace function during halt status, the previous coordinate value remains on the display. After the trace function is executed and the text display is brought up using the [G↔T] key, pressing [EXE] causes the next graph to appear and the coordinate value to clear.*
  Examine the above using [Graph] [ALPHA] [X] [x^2] [SHIFT] [◢] [Graph] 2 [ALPHA] [X] [+] 5

■ Plot function

The plot function is used to mark a point on the screen of a graph display. The point can be moved left, right, up and down using the cursor keys, and the coordinates for the graph displayed can be read. Two points can also be connected by a straight line (see Line function, page 65).

Press [SHIFT] [Plot] and specify the x and y-coordinates after the "Plot" message.

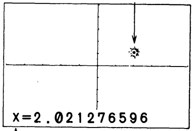
Ex. Plot a point at $x=2$ and $y=2$ on the axes created by the following range values:



```
Range
Xmin:-5.
 max:5.
 scl:1.
Ymin:-10.
 max:10.
 scl:2.
```

[SHIFT] [Plot] 2 [SHIFT] [,] 2 [EXE]

Blinking dot



X=2.021276596

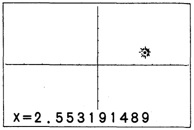
x-coordinate value display

The blinking pointer is positioned at the specified coordinates.

- *Due to limitations caused by the resolution of the display, the actual position of the pointer can only be approximate.*

The pointer can be moved left, right, up, and down using the cursor keys. The current position of the pointer is always shown at the bottom of the display.

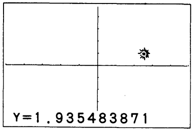
⇨ ⇨ ⇨ ⇨ ⇨



```
X=2.553191489
```

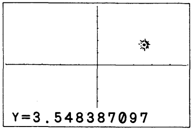
To find the y-coordinate value:

SHIFT X↔Y



```
Y=1.935483871
```

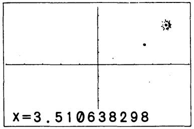
⇧ ⇧ ⇧ ⇧ ⇧



```
Y=3.548387097
```

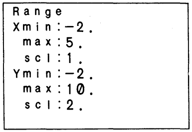
Now, inputting a new coordinate value causes the new pointer to blink without clearing the present pointer.

SHIFT Plot 3.5 SHIFT , 6.5 EXE



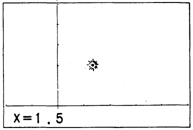
```
X=3.510638298
```

If x-y coordinates are not specified for the plot function, the pointer appears at the center of the screen.
Set the following range values:



```
Range
Xmin:-2.
 max:5.
 scl:1.
Ymin:-2.
 max:10.
 scl:2.
```

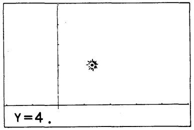
SHIFT Plot EXE



```
X=1.5
```

To find the Y-coordinate value:

SHIFT X↔Y



```
Y=4.
```

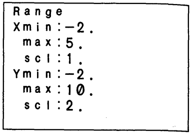
- * *Attempting to plot a point outside of the preset range is disregarded.*
- * *The x and y-coordinates of the pointer used in the plot function are respectively stored in the X memory and Y memory.*
- * *A blinking pointer becomes a fixed point (not blinking) when a new pointer is created.*

■ Line function

The line function makes it possible to connect two points (including the blinking pointer) created with the plot function with a straight line. With this funciton, user generated lines can be added to graphs to make them easier to read.

Ex. Draw perpendiculars from point (2,0) on the x-axis to its intersection with the graph for $y=3x$. Then draw a line from the point of intersection to the y-axis.

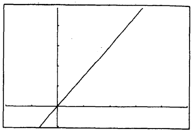
The range values for the graph are as follows:



```
Range
Xmin:-2.
 max:5.
 scl:1.
Ymin:-2.
 max:10.
 scl:2.
```

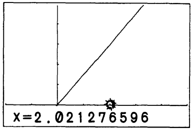
Clear the graph display and draw the graph for $y=3x$.

[SHIFT] [Cls] [EXE]
[Graph] 3 [ALPHA] [X] [EXE]



Next, use the plot function to locate a point at (2,0).

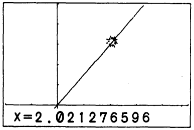
[SHIFT] [Plot] 2 [SHIFT] [,] 0 [EXE]



```
X=2.021276596
```

Now plot a point at (2,0) again and use the cursor key ([⇧]) to move the pointer up to the point on the graph ($y=3x$).

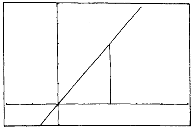
[SHIFT] [Plot] 2 [SHIFT] [,] 0 [EXE]
[⇧] ~ [⇧]
(Move the pointer up to the point on the graph for $y=3x$.)



```
X=2.021276596
```

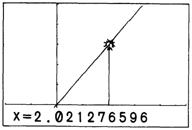
Draw a line using the line function.

[SHIFT] [Line] [EXE]



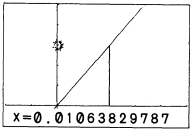
Next, a perpendicular will be drawn from the same point on the graph to the y-axis. First, plot the point on the graph and use the cursor key ([⇦]) to move the pointer to the y-axis. This can be accomplished using Plot X, Y since the x-y coordinates of the point on the graph are stored in the X and Y memories.

[SHIFT] [Plot] [ALPHA] [X] [SHIFT]
[,] [ALPHA] [Y] [EXE]



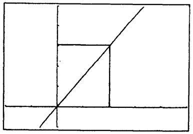
```
X=2.021276596
```

[⇦] ~ [⇦]
(Move the pointer to the y-axis.)



```
X=0.01063829787
```

[SHIFT] [Line] [EXE]



* *The line function can only be used to draw lines between the blinking pointer and a fixed point created using the plot function.*

Factor function

The factor function is used to magnify or reduce the range of a graph centered around the blinking pointer provided with the plot function or trace function.

For magnification, the minimum value and maximum value of the range are multiplied by $1/n$. For reduction, they are multiplied by n.

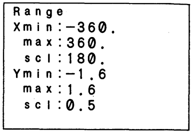
● Operation

`SHIFT` `Factor` m `SHIFT` `,` n `EXE` ············· x is magnified m times and y is magnified n times centered around the pointer.

`SHIFT` `Factor` n `EXE` ························· x and y are both magnified n times centered around the pointer.

The graph display is cleared when the factor function is executed because of changes in the range values.

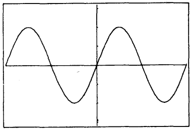
Ex. After setting the range values specified below, magnify the graph for y=sin x centered on the origin.



```
Range
Xmin:-360.
 max:360.
 scl:180.
Ymin:-1.6
 max:1.6
 scl:0.5
```

Draw the graph for y=sin x after setting the range values.

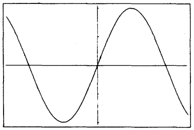
`Graph` `sin` `ALPHA` `X` `EXE`



Now use the plot function to blink the pointer at the origin of the graph and then use the factor function to magnify the graph 1.5 times.

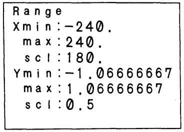
`SHIFT` `Plot` `:` `SHIFT` `Factor` 1.5 `:`
`Graph` `sin` `ALPHA` `X` `EXE`

* *The multistatement function is used to produce the graph in a single step.*



The following shows the resulting range values:

`Range`

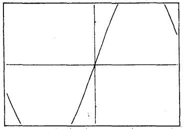


```
Range
Xmin:-240.
 max:240.
 scl:180.
Ymin:-1.06666667
 max:1.06666667
 scl:0.5
```

This indicates that the range values for the x and y-axes are equal to 1/1.5 of their original values.

Now let's try magnifying the graph another 1.5 times.
This time, it is not necessary to input any further commands. The existing graph is magnified by simply pressing `EXE`. Since the original magnification was accomplished using the multistatement function, the replay function becomes operational.

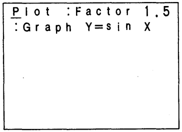
`EXE`



Now the graph is so large that little of it remains on the display. Let's try to reduce the graph to half its present size to make it more manageable.

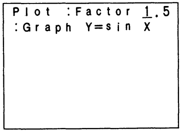
The replay function is used to change the magnification value from 1.5 to 0.5.

`⇨`



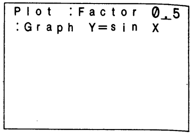
```
Plot :Factor 1.5
:Graph Y=sin X
```

`⇨` `⇨` `⇨`



```
Plot :Factor 1.5
:Graph Y=sin X
```

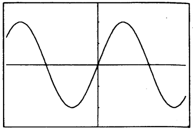
0



```
Plot :Factor 0.5
:Graph Y=sin X
```

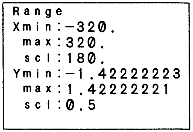
Now execute the function.

[EXE]



The following display shows the new range values:

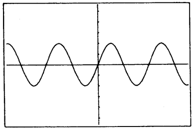
[Range]



```
Range
Xmin:-320.
 max:320.
 scl:180.
Ymin:-1.42222223
 max:1.42222221
 scl:0.5
```

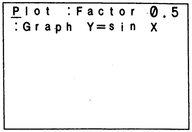
To reduce the graph by half again:

[EXE]



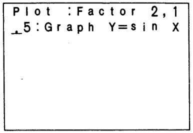
Now let's double the x-axis and increase the y-axis by 1.5 times.

[⇨]



```
Plot :Factor 0.5
:Graph Y=sin X
```

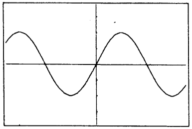
[⇨] [⇨] [⇨] [SHIFT] [INS]
2 [SHIFT] [,]
[SHIFT] [INS] 1



```
Plot :Factor 2,1
.5:Graph Y=sin X
```

Now execute the function.

[EXE]



Using the operations outlined in this section, graphs can be magnified or reduced. In the examples given here, the graphs were magnified and reduced centered around the origin, but any pointer on the display can be used as a central point for magnification and reduction.

■ Instant factor function

The instant factor function can be used to quickly magnify the size of a graph by 2^n or reduce it by $1/2^n$. The change in size is centered at the pointer when it is displayed, and at the center of the graph when the pointer is not displayed.

● Operation

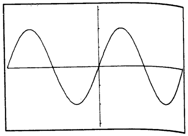
[SHIFT] [×] ... 2X magnification in both x and y directions. Pressing [SHIFT] [×] again magnifies by 2^2 or 4X, and a third press magnifies by 2^3 or 8X.

[SHIFT] [÷] ... 1/2 reduction in both x and y directions. Pressing [SHIFT] [÷] again reduces by $1/2^2$ or 1/4, and a third press reduces by $1/2^3$ or 1/8.

Since range contents are switched to their inverse proportions, the graphic display is cleared each time the instant factor function is executed.

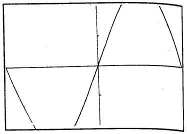
Ex. Graph $y=\cos x$ using the built-in function, and change the size by 2X and 1/2.

$\boxed{\text{Graph}}$ $\boxed{\sin}$ $\boxed{\text{EXE}}$



Now magnify the graph 2 X at the center of the display.

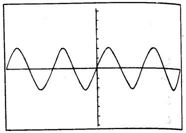
$\boxed{\text{SHIFT}}$ $\boxed{\times}$



Next, reduce the original $y=\cos x$ graph by 1/2.

$\boxed{\text{SHIFT}}$ $\boxed{\div}$ (Returns to original graph.)

$\boxed{\text{SHIFT}}$ $\boxed{\div}$



In the above examples, the changes in the graph size were performed at the center of the display. If the pointer is shown on the display, the magnification/reduction is applied centered at the pointer.

3-3 SINGLE VARIABLE STATISTICAL GRAPHS

- Single variable statistical graphs are drawn in the SD2 mode ($\boxed{\text{SHIFT}}$ $\boxed{\text{MODE}}$ $\boxed{\times}$).
- Bar graphs, line graphs, and normal distribution curves can be produced as single variable statistical graphs.
- Function graphs are also possible in the SD2 mode, so graphs of theoretical values and graphs of actual values can be overwritten.
  * *Abs and $\sqrt[3]{\ }$ cannot be used in the SD2 mode.*
- Number of data is determined by expanding memories.
- Graphs are drawn with the x-coordinate as the data range and the y-coordinate as the number of items (frequency) of each data.
- The $\boxed{\text{DT}}$ key ($\boxed{\sqrt[x]{\ }}$) is used for data input.
- The $\boxed{\text{CL}}$ key ($\boxed{x^y}$) is used for data correction.

■ Drawing single variable statistical graphs

- Procedure

① Specify the SD2 mode ($\boxed{\text{SHIFT}}$ $\boxed{\text{MODE}}$ $\boxed{\times}$).
② Set the range values ($\boxed{\text{Range}}$).
③ Expand the memory in accordance with the number of bars ($\boxed{\text{MODE}}$ $\boxed{\cdot}$ n $\boxed{\text{EXE}}$).
④ Clear the statistical memories ($\boxed{\text{SHIFT}}$ $\boxed{\text{Scl}}$ $\boxed{\text{EXE}}$).
⑤ Input data (Data $\boxed{\text{DT}}$ ($\boxed{\sqrt[x]{\ }}$)).
⑥ Draw the graph.
  • Bar graph.. $\boxed{\text{Graph}}$ $\boxed{\text{EXE}}$
  • Line graph...................................... $\boxed{\text{Graph}}$ $\boxed{\text{SHIFT}}$ $\boxed{\text{Line}}$ $\boxed{\text{EXE}}$
  • Normal distribution curve............... $\boxed{\text{Graph}}$ $\boxed{\text{SHIFT}}$ $\boxed{\text{Line}}$ 1 $\boxed{\text{EXE}}$
  * *Data input method in step 5 is the same as that for standard deviation computations (see page 44).*

Ex. Use the following data to draw a ranked graph.

Rank No.	1	2	3	4	5	6	7	8	9	10	11
Rank	0	10	20	30	40	50	60	70	80	90	100
Frequency	1	3	2	2	3	5	6	8	15	9	2

Perform graph preparation in accordance with the following procedure:

① Specify the SD2 mode (SHIFT MODE X).

② Set the range values.
The highest value to be plotted on the x-axis is 100, but for graphing purposes the maximum value (Xmax) is set at 110. (The general rule is that the minimum value should be equal to or greater than the minimum range value and the maximum value should be less than the maximum range value, so here we set the x-axis ranges to 0 through 110.)
Ymax value is set to 20 for the y-axis because the maximum frequency is 15.

```
Range
Xmin:0.
 max:110.
 scl:10.
Ymin:0.
 max:20.
 scl:2.
```

③ Since the number of bars is 11(0～9, 10～19, 20～29. . . . 100～109) expand memories by 11.

MODE . 11 EXE

```
** Defm **

 Program :  0

  Memory :37

  334 Bytes Free
```

④ Clear the statistical memory.

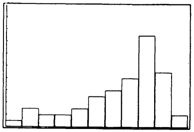
SHIFT Scl EXE

⑤ Input the data.

0 DT 10 DT DT DT 20 DT DT 30 DT DT 40 DT DT DT
50 SHIFT ; 5 DT 60 SHIFT ; 6 DT 70 SHIFT ; 8 DT
80 SHIFT ; 15 DT 90 SHIFT ; 9 DT 100 DT DT

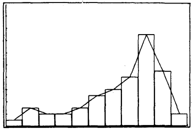
⑥ First, draw a bar graph.

Graph EXE



Next, overwrite a line graph.

Graph SHIFT Line EXE

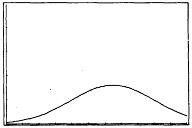


- Finally, draw a normal distribution curve. Since the y-axis value is relatively small when compared with the bar and line graphs, the same range values cannot be used. Change the range values to those shown below.

```
Range
Xmin:0.
 max:110.
 scl:10.
Ymin:0.
 max:0.05
 scl:0.01
```

Graph SHIFT Line 1 EXE

Inputting the number 1 causes a normal distribution curve to be drawn.



- Be sure to expand the memory in accordance with the number of bars. A Mem-error is generated if memory expansion is not performed.
- If the number of expanded memories is changed during data input, the number of data divisions also changes, thus making it impossible to produce a proper graph.
- When a value that exceeds the preset ranges is input, it is input to the statistical memory, but not into the graph memory.
- When more data than the preset y-axis range is input, the bar graph is drawn to the upper limit of the display, and the points outside the range cannot be connected.
- The formula used for normal distribution curves is:

$$y = \frac{1}{\sqrt{2\pi}\,\sigma} e^{-\frac{(x-m)^2}{2\sigma^2}}$$

* *Keyboard designation of σ is $x\sigma n$. m is $\bar{x}$.*

- The following must be true in the case of range settings: Xmin<Xmax.
- After a bar or line graph is executed, "done" is displayed in the text display.

3-4 PAIRED VARIABLE STATISTICAL GRAPHS

- Paired variable graphs are drawn in the LR2 mode ($\boxed{\text{SHIFT}}$ $\boxed{\text{MODE}}$ $\boxed{\div}$).
- Paired variable graphs can be drawn as regression lines.
- Standard function graphs can also be drawn in the LR2 mode, so theoretical graphs, data distribution and regression line graphs can be overwritten.
- After data input in the LR2 mode, points are displayed immediately, and data is input to the statistical memory.
- When a value that exceeds the preset range is input, it is input to the statistical memory, the point is not displayed.
- Data is input using the $\boxed{\text{DT}}$ ($\boxed{\sqrt[x]{\ }}$) key in the following format: x data $\boxed{\text{SHIFT}}$ $\boxed{,}$ y data $\boxed{\text{SHIFT}}$ $\boxed{;}$ frequency $\boxed{\text{DT}}$.
- The $\boxed{\text{CL}}$ ($\boxed{x^y}$) key is used to edit data after input is complete, but points that are produced on the display are not cleared. (Point appears even when data is corrected by the $\boxed{\text{CL}}$ key).
- Points on the display cannot be retrieved if the display is cleared ($\boxed{\text{SHIFT}}$ $\boxed{\text{Cls}}$ $\boxed{\text{EXE}}$).

■ Drawing paired variable statistical graphs

- Procedure

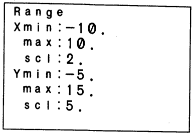
① Specify the LR2 mode ($\boxed{\text{SHIFT}}$ $\boxed{\text{MODE}}$ $\boxed{\div}$).
② Set the range values ($\boxed{\text{Range}}$).
③ Clear the statistical memory ($\boxed{\text{SHIFT}}$ $\boxed{\text{Scl}}$ $\boxed{\text{EXE}}$).
④ Input data (x data $\boxed{\text{SHIFT}}$ $\boxed{,}$ y data $\boxed{\text{SHIFT}}$ $\boxed{;}$ frequency $\boxed{\text{DT}}$).
⑤ Draw the graph ($\boxed{\text{Graph}}$ $\boxed{\text{SHIFT}}$ $\boxed{\text{Line}}$ 1 $\boxed{\text{EXE}}$).

* *Data input method in step 4 is the same as that for Regression computation (Page 46).*

Ex. Perform linear regression on the following data and draw a regression line graph.

x_i	y_i
−9	−2
−5	−1
−3	2
1	3
4	5
7	8

① Specify the LR2 mode ($\boxed{\text{SHIFT}}$ $\boxed{\text{MODE}}$ $\boxed{\div}$).
② Set the range values to those shown in the table.



```
Range
Xmin:-10.
  max:10.
  scl:2.
Ymin:-5.
  max:15.
  scl:5.
```

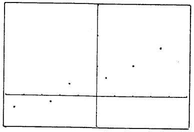
* *According to the general rule of the x-axis range values, the values for x are: $-10 \leqq x < 10$.*

③ Clear the statistical memories.

$\boxed{\text{SHIFT}}$ $\boxed{\text{Scl}}$ $\boxed{\text{EXE}}$

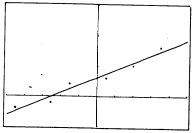
④ Input the data.

$\boxed{(-)}$ 9 $\boxed{\text{SHIFT}}$ $\boxed{,}$ $\boxed{(-)}$ 2 $\boxed{\text{DT}}$
$\boxed{(-)}$ 5 $\boxed{\text{SHIFT}}$ $\boxed{,}$ $\boxed{(-)}$ 1 $\boxed{\text{DT}}$
$\boxed{(-)}$ 3 $\boxed{\text{SHIFT}}$ $\boxed{,}$ 2 $\boxed{\text{DT}}$
1 $\boxed{\text{SHIFT}}$ $\boxed{,}$ 3 $\boxed{\text{DT}}$
4 $\boxed{\text{SHIFT}}$ $\boxed{,}$ 5 $\boxed{\text{DT}}$
7 $\boxed{\text{SHIFT}}$ $\boxed{,}$ 8 $\boxed{\text{DT}}$



⑤ Draw the graph.

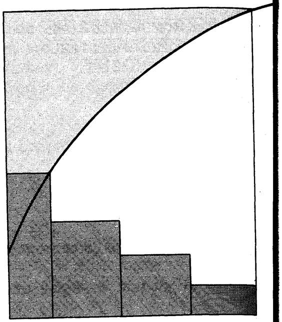
$\boxed{\text{Graph}}$ $\boxed{\text{SHIFT}}$ $\boxed{\text{Line}}$ 1 $\boxed{\text{EXE}}$



* *When data is input that is outside of the preset range values, a point does not appear.*
* *An Ma ERROR is generated when there is no data input and the following key operation is performed:* $\boxed{\text{Graph}}$ $\boxed{\text{SHIFT}}$ $\boxed{\text{Line}}$ 1 $\boxed{\text{EXE}}$.
* *The following must be true in the case of range settings: Xmin< Xmax.*

4.

PROGRAM COMPUTATIONS

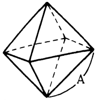


4-1 WHAT IS A PROGRAM?

This unit has a built-in program feature that facilitates repeat computations. The program feature is used for the consecutive execution of formulas in the same way as the "multistatement" feature is used in manual computations. Programs will be discussed here with the aid of illustrative examples.

EXAMPLE:

Find the surface area and volume of a regular octahedron when the length of one side is given.



Length of one side (A)	Surface area (S)	Volume (V)
10cm	( )cm^2	( )cm^3
7	( )	( )
15	( )	( )

* Fill in the parentheses.

① Formulas

For a surface area S, volume V and one side A, S and V for a regular octahedron are defined as:

$$S=2\sqrt{3}A^2 \qquad V=\frac{\sqrt{2}}{3}A^3$$

② Programming

Creating a program based on computation formulas is known as "programming". Here a program will be created based upon the formulas given above. The basis of a program is manual computation, so first of all, consider the operational method used for manual computation.

Surface area (S): 2 $\boxed{\times}$ $\boxed{\sqrt{\ }}$ 3 $\boxed{\times}$ Numeric value A $\boxed{x^2}$ $\boxed{\text{EXE}}$

Volume (V): $\boxed{\sqrt{\ }}$ 2 $\boxed{\div}$ 3 $\boxed{\times}$ Numeric value A $\boxed{x^y}$ 3 $\boxed{\text{EXE}}$

In the above example, numeric value A is used twice, so it should make sense to store it in memory A before the computations.

Numeric value A → $\boxed{\text{ALPHA}}$ $\boxed{\text{A}}$ $\boxed{\text{EXE}}$
2 $\boxed{\times}$ $\boxed{\sqrt{\ }}$ 3 $\boxed{\times}$ $\boxed{\text{ALPHA}}$ $\boxed{\text{A}}$ $\boxed{x^2}$ $\boxed{\text{EXE}}$ .... S
$\boxed{\sqrt{\ }}$ 2 $\boxed{\div}$ 3 $\boxed{\times}$ $\boxed{\text{ALPHA}}$ $\boxed{\text{A}}$ $\boxed{x^y}$ 3 $\boxed{\text{EXE}}$ V

With this unit, the operations performed for manual computations can be used as they are in a program. Once program execution starts, it will continue in order without stopping. Therefore, commands are required to request the input of data and to display results. The command to request data input is "?", while that to display results is "◢".

A "?" within a program will cause execution to stop temporarily and a "?" to appear on the display as the unit waits for data input. This command cannot be used independently, and is used together with → as " SHIFT ? → memory name". To store a numeric value in memory A, for example:

?→A

When "?" is displayed, calculation commands and numeric values can be input within 111 steps.

The "◢" command causes program execution to stop temporarily and the latest formula result or alphanumeric characters and symbols (see page 116) to be displayed. This command is used to mark positions in formulas where results are to be displayed. Since programs are ended and their final results displayed automatically, this command can be omitted at the end of program. However, if the BASE-N mode is specified for base conversion during a program, do not omit final "◢".

Here these two commands will be used in the previously presented procedure:

SHIFT ? → ALPHA A : 2 × √ 3 × ALPHA A x^2 SHIFT ◢

Input to memory A — Display S

√ 2 ÷ 3 × ALPHA A x^y 3

◢ omitted

Now the program is complete.

③ Program storage

The storage of programs is performed in the WRT mode which is specified by pressing MODE 2 .

Operation **Display**

MODE 2

```
sys mode  : WRT
cal mode  : COMP
   angle  : Deg
 display  : Norm

  422 Bytes Free

 Prog 0123456789
```

When MODE 2 are pressed, the system mode changes to the WRT mode. Then, the number of remaining steps (see page 93) is indicated. The number of remaining steps is decreased when programs are input or when memories are expanded. If no programs have been input and the number of memories equals 26 (the number of memories at initialization), the number of usable steps should equal 422.

The larger figures located below indicate the program areas (see page 95). If the letter "P" is followed by the numbers 0 through 9, it indicates that there are no programs stored in areas P0 through P9. The blinking zero here indicates the current program area is P0.

Areas into which programs have already been stored are indicated by "_" instead of numbers.

```
sys mode  : WRT
cal mode  : COMP
   angle  : Deg
 display  : Norm

  248 Bytes Free

 Prog 01_34_6789
```

Here the previously mentioned program will be stored to program area P0 (indicated by the blinking zero):

Operation **Display**

EXE (Start storage)

```
_
```

SHIFT ? → ALPHA A : 2 × √ 3
× ALPHA A x^2 SHIFT ◢

```
?→A:2×√3×A²◢
_
```

√ 2 ÷ 3 × ALPHA A x^y 3

```
?→A:2×√3×A²◢
√2÷3×A x^y 3_
```

After these operations are complete, the program is stored.

* *The system display appears only while the* M Disp *key is pressed.*

M Disp (Displayed while pressed)

```
****  MODE  ****

sys mode  : WRT
cal mode  : COMP
   angle  : Deg
 display  : Norm

    Step P0-20
```

* *After the program is stored, press* MODE 1 *to return to the RUN mode.*

④ Program execution

Programs are executed in the RUN mode (MODE 1). The program area to be executed is specified using the Prog key.

To execute P0: Prog 0 EXE
To execute P3: Prog 3 EXE
To execute P8: Prog 8 EXE

Here the sample program that has been stored will be executed. The surface (S) and volume (V) for the regular octahedron in the sample problem are computed as:

Length of one side (A)	Suface area (S)	Volume (V)
10cm	(346.4101615)cm²	(471.4045208)cm³
7	(169.7409791)	(161.6917506)
15	(779.4228634)	(1590.990258)

Operation **Display**

MODE 1

```
**** MODE ****

sys mode  : RUN
cal mode  : COMP
   angle  : Deg
 display  : Norm

   Step     0
```

Prog 0 EXE

```
?→A : 2×√3×A ²◢
√2÷3×A xʸ3
Prog 0
?
```

10 EXE
(Value of A)

```
?→A : 2×√3×A ²◢
√2÷3×A xʸ3
Prog 0
?
10
          346.4101615
           — Disp —
```

(S when A = 10)

Indicates answer displayed by ◢.

EXE

```
?→A : 2×√3×A ²◢
√2÷3×A xʸ3
Prog 0
?
10
          346.4101615
          471.4045208
```

(V when A = 10)

Prog 0 EXE

```
√2÷3×A xʸ3
Prog 0
?
10
          346.4101615
          471.4045208
Prog 0
?
```

7 EXE (Value of A)

```
10
          346.4101615
          471.4045208
Prog 0
?
7
          169.7409791
           — Disp —
```

(S when A = 7)

EXE

```
10
          346.4101615
          471.4045208
Prog 0
?
7
          169.7409791
          161.6917506
```

(V when A = 7)

[Prog] 0 [EXE]

```
      471.4045208
Prog 0
?
7
      169.7409791
      161.6917506
Prog 0
?
```

15 [EXE]
(Value of A)

```
7
      169.7409791
      161.6917506
Prog 0
?
15
      779.4228634
        - Disp -
```

(S when A = 15)

[EXE]

```
7
      169.7409791
      161.6917506
Prog 0
?
15
      779.4228634
      1590.990258
```

(V when A = 15)

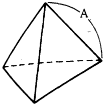
* *Program computations are performed automatically with each press of [EXE] when it is pressed after data is input or after the result is read.*
* *Directly after a program in P0 is executed by pressing [Prog] 0 [EXE] as in this example, the Prog 0 command is stored by the replay function. Therefore, subsequent executions of the same program can be performed by simply pressing [EXE].*

Operation	[Prog] 0 [EXE]	(P0 program execution)
	10 [EXE]	(Input 10 for A)
	[EXE]	(Display V when A = 10)
	[EXE]	(Reexecute)
	7 [EXE]	(Input 7 for A)
	[EXE]	(Display V when A = 7)
	⋮	

4-2 PROGRAM CHECKING AND EDITING (CORRECTION, ADDITION, DELETION)

Recalling a stored program can be performed in order to verify its contents. After specifying the desired program area using [⇦] or [⇨] in the WRT mode ([MODE] [2]), the program contents will be displayed by pressing the [EXE] key. Once the program is displayed, the [⇨] (or [⇦], [⇧], [⇩]) key is used to advance the program one step at a time for verification. When the program has been improperly stored, editing can also be performed by adding to it or erasing portions. Here a new program will be created by checking and editing the previous sample program (the surface area and volume of a regular octahedron).

EXAMPLE:
Find the surface area and volume of a regular tetrahedron when the length of one side is given.



Length of one side (A)	Surface area (S)	Volume (V)
10 cm	( )cm²	( )cm³
7.5	( )	( )
20	( )	( )

① Formulas

For a surface area S, volume V and one side A, S and V for a regular tetrahedron are defined as:

$$S=\sqrt{3}\,A^2 \qquad V=\frac{\sqrt{2}}{12}A^3$$

② Programming

As with the previous example, the length of one side is stored in memory A and the program then constructed.

Numeric value A → [ALPHA] [A] [EXE]
[√] 3 [×] [ALPHA] [A] [x²] [EXE] ·· S
[√] 2 [÷] 12 [×] [ALPHA] [A] [xʸ] 3 [EXE] ··································· V

When the above is formed into a program, it appears as follows:

[SHIFT] [?] [→] [ALPHA] [A] [:] [√] 3 [×] [ALPHA] [A] [x²] [SHIFT] [◢]
[√] 2 [÷] 12 [×] [ALPHA] [A] [xʸ] 3

③ Program editing

First, a comparison of the two programs would be helpful.

Octahedron: [SHIFT] [?] [→] [ALPHA] [A] [:] 2 [×] [√] 3 [×] [ALPHA] [A] [x^2] [SHIFT] [◢]
[√] 2 [÷] 3 [×] [ALPHA] [A] [x^y] 3

Tetrahedron: [SHIFT] [?] [→] [ALPHA] [A] [:] [√] 3 [×] [ALPHA] [A] [x^2] [SHIFT] [◢]
[√] 2 [÷] 12 [×] [ALPHA] [A] [x^y] 3

The octahedron program can be changed to a tetrahedron program by deleting the parts marked with wavy lines, and changing those that are marked with straight lines.

In actual practice, this would be performed as follows:

Operation **Display**

[MODE] [2]

```
sys mode  : WRT
cal mode  : COMP
   angle  : Deg
 display  : Norm

  402 Bytes Free

 Prog _123456789
```

[EXE]

```
?→A:2×√3×A²◢
√2÷3×Ax^y3
```

Cursor located at beginning. Press [SHIFT] [EXE] to bring cursor to end.

[⇨] [⇨] [⇨] [⇨]
[DEL] [DEL]

```
?→A:√3×A²◢
√2÷3×Ax^y3
```

Locate cursor at position to be deleted, and delete two characters.

[⇩] [⇦] [SHIFT]
[INS] 12

```
?→A:√3×A²◢
√2÷123×Ax^y3
```

Insert two characters.

[DEL]

```
?→A:√3×A²◢
√2÷12×Ax^y3
```

Delete unnecessary 3.

[MODE] [1]

```
**** MODE ****

sys mode  : RUN
cal mode  : COMP
   angle  : Deg
 display  : Norm

  Step     0
```

Editing cmoplete. Return to the RUN mode.

④ Program execution

Now this program will be executed.

Length of one side (A)	Surface area (S)	Volume (V)
10 cm	(173.2050808)cm²	(117.8511302)cm³
7.5	(97.42785793)	(49.71844555)
20	(692.820323)	(942.8090416)

Operation **Display**

[MODE] [1]

```
**** MODE ****

sys mode  : RUN
cal mode  : COMP
   angle  : Deg
 display  : Norm

  Step     0
```

[Prog] 0 [EXE]

```
?→A:√3×A²◢
√2÷12×Ax^y3
Prog 0
?
```

10 [EXE]

```
?→A:√3×A²◢
√2÷12×Ax^y3
Prog 0
?
10
         173.2050808
              - Disp -
```

[EXE]

```
?→A:√3×A²◢
√2÷12×Ax^y3
Prog 0
?
10
         173.2050808
         117.8511302
```

Prog 0 EXE

```
√2÷12×Axʸ3
Prog 0
?
10
          173.2050808
          117.8511302
Prog 0
?
```

7.5 EXE

```
10
          173.2050808
          117.8511302
Prog 0
?
7.5
          97.42785793
             — Disp —
```

EXE

```
10
          173.2050808
          117.8511302
Prog 0
?
7.5
          97.42785793
          49.71844555
```

Prog 0 EXE

```
          117.8511302
Prog 0
?
7.5
          97.42785793
          49.71844555
Prog 0
?
```

20 EXE

```
7.5
          97.42785793
          49.71844555
Prog 0
?
20
           692.820323
             — Disp —
```

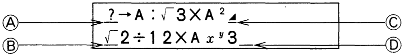
EXE

```
7.5
          97.42785793
          49.71844555
Prog 0
?
20
           692.820323
          942.8090416
```

〈Reference〉

Cursor movement

Pressing the cursor keys (⇦, ⇨, ⇧, ⇩) causes the cursor to move as follows:



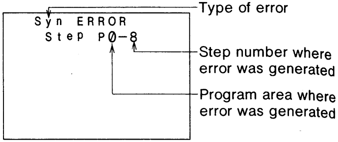
Cursor position	⇦	⇨	⇧	⇩
Ⓐ	Invalid	1 position right	Invalid	1 line down (Ⓑ)
Ⓑ	1 position left (Ⓒ)	1 position right	1 line up (Ⓐ)	End of line (Ⓓ)
Ⓒ	1 position left	1 position right (Ⓑ)	Beginning of line (Ⓐ)	1 line down (Ⓓ)
Ⓓ	1 position left	Invalid	1 line up (Ⓒ)	Invalid

4-3 PROGRAM DEBUGGING (CORRECTING ERRORS)

After a program has been created and input, it will sometimes generate error messages when it is exeuted, or it will produce unexpected results. This indicates that there is an error somewhere within the program that needs to be corrected. Such programming errors are referred to as "bugs", while correcting them is called "debugging".

■ Debugging when an error message is generated

An error message is displayed as follows:



```
Syn ERROR
 Step P0-8
```

The error message informs the operator of the program area (P0 to P9) in which the error was generated. It also states the type of error, which gives an idea of the proper countermeasure to be taken. The step number indicates in which step of the program area the error was generated.

■ Error messages

There are a total of seven error messages.

① **Syn ERROR** (Syntax error)
Indicates a mistake in the formula or a misuse of program commands.

② **Ma ERROR** (Mathematical error)
Indicates the computation result of a numeric expression exceeds 10^{100}, an illogical operation (i.e. division by zero), or the input of an argument that exceeds the input range of the function.

③ **Go ERROR** (Jump error)
Indicates a missing Lbl for the Goto command (see page 100), or that the program area (see page 95) for the Prog command (see page 107) does not contain a program.

④ **Ne ERROR** (Nesting error)
Indicates a subroutine nesting overflow by the Prog command.

⑤ **Stk ERROR** (Stack error)
Indicates the computation performed exceeds the capacity of the stack for numeric values or for commands (see page 15).

⑥ **Mem ERROR** (Memory error)
Indicates the attempt to use a memory name such as Z [5] without having expanded memories.

⑦ **Arg ERROR** (Argument error)
Indicates the argument of a command or specification in a program exceeds the input range (i.e. Sci 10, Goto 11).

Further operation will become impossible when an error message is displayed. Press [AC], [⇦], or [⇨] to cancel the error.
Pressing [AC] cancels the error and new key input becomes possible. With this operation, the RUN mode is maintained.
Pressing [⇦] or [⇨] cancels the error and changes the system mode to the WRT mode. The cursor is positioned at the location where the error was generated to allow modification of the program to eliminate the error.

■ Checkpoints for each type of error

The following are checkpoints for each type of error:

① **Syn ERROR**
Verify again that there are no errors in the program.

② **Ma ERROR**
For computations that require use of the memories, check to see that the numeric values in the memories do not exceed the range of the arguments. This type of error often occurs with division by 0 or the computation of negative square roots.

③ **Go ERROR**
Check to see that there is a corresponding Lbl *n* when Goto *n* is used. Also check to see that the program in P *n* has been correctly input when Prog *n* is used.

④ **Ne ERROR**
Check to ensure that the Prog command is not used in the branched program area to return execution to the original program area.

⑤ **Stk ERROR**
Check to see that the formula is not too long thus causing a stack overflow. If this is the case, the formula should be divided into two or more parts.

⑥ **Mem ERROR**
Check to see that memories were properly expanded using " MODE · n EXE " (Defm). When using array-type memories (see page 110), check to see that the subscripts are correct.

⑦ **Arg ERROR**
Check whether values specified by MODE 7 (Sci) or MODE 8 (Fix) are within the range of 0 ~ 9. Also check whether values specified by Goto, Lbl, or Prog commands are within 0—9. Also ensure that memory expansion using MODE · (Defm) is performed within the remaining number of steps and that the value used for expansion is not negative.

4-4 COUNTING THE NUMBER OF STEPS

The program capacity of this unit consists of a total of 422 steps.
The number of steps indicates the amount of storage space available for programs, and it will decrease as programs are input. The number of remaining steps will also be decreased when steps are converted to memories. (See page 23).
There are two methods to determine the current number of remaining steps:

① When MODE · EXE are pressed in the RUN mode, the number of remaining steps will be displayed together with the number of memories.

Example:

MODE · EXE

```
**Defm**

 Program : 19 ←— Number of steps used
                  for programming
  Memory : 26 ←— Number of memories

  403 Bytes Free← Number of remaining
                  steps
```

② Specify the WRT mode (MODE 2), and the number of remaining steps will appear. At this time the status of the program areas can also be determined.

MODE 2

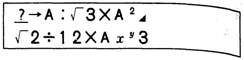
```
sys mode : WRT
cal mode : COMP
   angle : Deg
 display : Norm

  403 Bytes Free← Number of remaining
                  steps
Prog _123456789
```

Basically, one function requires a single step, but there are some commands where one function requires two steps.
- One function/one step: sin, cos, tan, log, (,), :, A, B, 1, 2, 3, etc.
- One function/two steps: Lbl 1, Goto 2, Prog 8, etc.

Each step can be verified by the movement of the cursor:

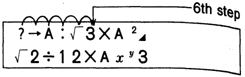
Example:

Present cursor position→



```
?→A:√3×A²◢
√2÷12×A xʸ3
```

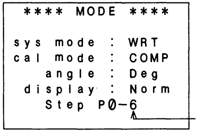
At this time, each press of a cursor key (⇦ or ⇨) will cause the cursor to move to the next sequential step. For example:



```
?→A:√3×A²◢
√2÷12×A xʸ3
```

The display will show at what step of the program the cursor is currently located as long as M Disp is pressed.

M Disp ~
(Hold down)



```
  ****  MODE  ****

sys mode  :  WRT
cal mode  :  COMP
   angle  :  Deg
 display  :  Norm
   Step  P0-6
```

Indicates cursor is located at 6th step.

4-5 PROGRAM AREAS AND COMPUTATION MODES

This unit contains a total of 10 program areas (P0 through P9) for the storage of programs. These program areas are all utilized in the same manner, and 10 independent programs can be input. One main program (main routine) and a number of secondary programs (subroutines) can also be stored. The total number of steps available for storage in program areas P0 through P9 is 422 maximum.

Specification of a program area is performed as follows:

RUN mode: Press any key from 0 through 9 after pressing the Prog key. Then press EXE .

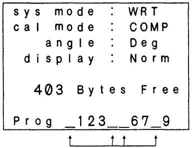
Example: P0 Prog 0 EXE
P8 Prog 8 EXE

* *In this mode, program execution begins when EXE is pressed.*

WRT mode: Use ⇦ or ⇨ to move the cursor under the program area to be specified and press EXE.

Only the numbers of the program areas that do not yet contain programs will be displayed. "_" symbols indicate program areas which already contain programs.

Example:



```
sys mode  :  WRT
cal mode  :  COMP
   angle  :  Deg
 display  :  Norm

  403  Bytes  Free

Prog  _123__67_9
```

Programs already stored in these program areas.

■ Program area and computation mode specification in the WRT mode

Besides normal function computations, to perform binary, octal, decimal and hexadecimal computations and conversions, standard deviation computations, and regression computations in a program, a computation mode must be specified. Program mode specification and program area specification are performed at the same time.

First the WRT mode is specified (MODE 2), and then a computation mode is specified. Next, the program area is specified, and, when EXE is pressed, the computation mode is memorized in the program area.
Henceforth, stored programs will be accompanied with the computation mode.

Example: Memorizing the BASE-N mode in P2

MODE 2

```
sys mode  : WRT
cal mode  : COMP
   angle  : Deg
 display  : Norm

  422 Bytes Free

Prog 0123456789
```

Assuming that nothing is stored.

⇨ ⇨

```
sys mode  : WRT
cal mode  : COMP
   angle  : Deg
 display  : Norm

  422 Bytes Free

Prog 0123456789
```

Specify P2.

MODE −

```
sys mode  : WRT
cal mode  :BASE-N
             Dec


  422 Bytes Free

Prog 0123456789
```

Specify the BASE-N mode.

EXE

```
—
```

As shown above, the computation mode will be memorized into a program area.

Cautions concerning the computation modes

All key operations available in each computation mode can be stored as programs, but, depending on the computation mode, certain commands or functions cannot be used.

BASE-N mode

- Function computations cannot be performed.
- Units of angular measurement cannot be specified.
- All program commands can be used.
- Be sure to include a "◢" at the final result output to return to the previous computation mode when a program execution is terminated. Failure to do so may result in a decimal display or an error.

SD1, SD2 mode

- Among the functions, Abs and $\sqrt[3]{\ }$ cannot be used.
- Among the program commands, Dsz, > and < cannot be used.

LR1, LR2 mode

- Among the functions, Abs and $\sqrt[3]{\ }$ cannot be used.
- Among the program commands, ⇒, =, ≠, Isz, ≥, ≤, Dsz, > and < cannot be used.

4-6 ERASING PROGRAMS

Erasing of programs is performed in the PCL mode. Press MODE 3 to specify the PCL mode. There are two methods used to erase programs: erasing a program located in a single program area, and erasing all programs.

■ Erasing a single program

To erase a program in a single program area, specify the PCL mode and press the AC key after specifying the program area.

Example: Erase the program in P3 only.

Operation **Display**

MODE 3

```
sys mode  : PCL
cal mode  : COMP
   angle  : Deg
 display  : Norm

  324 Bytes Free

 Prog _12_45678_
```

P0, P3 and P9 already contain programs.

⇨ ⇨ ⇨

```
sys mode  : PCL
cal mode  : COMP
   angle  : Deg
 display  : Norm

  324 Bytes Free

 Prog _12_45678_
```

Align cursor with P3.

AC

```
sys mode  : PCL
cal mode  : COMP
   angle  : Deg
 display  : Norm

  367 Bytes Free

 Prog _12345678_
```

Number 3 appears after deletion.

MODE 1

```
**** MODE ****

sys mode  : RUN
cal mode  : COMP
   angle  : Deg
 display  : Norm

   Step    0
```

Return to RUN mode.

■ Erasing all programs

To erase all programs stored in program areas 0 through 9, specify the PCL mode and press SHIFT and then DEL.

Example: Erase the programs stored in P0, P4, P8 and P9.

Operation **Display**

MODE 3

```
sys mode  : PCL
cal mode  : COMP
   angle  : Deg
 display  : Norm

  295 Bytes Free

 Prog _123_567__
```

SHIFT DEL

```
sys mode  : PCL
cal mode  : COMP
   angle  : Deg
 display  : Norm

  422 Bytes Free

 Prog 0123456789
```

MODE 1

```
**** MODE ****

sys mode  : RUN
cal mode  : COMP
   angle  : Deg
 display  : Norm

   Step    0
```

4-7 CONVENIENT PROGRAM COMMANDS

The programs for this unit are made based upon manual computations. Special program commands, however, are available to allow the selection of the formula, and repetitive execution of the same formula.
Here, some of these commands willl be used to produce more convenient programs.

■ Jump commands

Jump commands are used to change the flow of program execution. Programs are executed in the order that they are input (from the lowest step number first) until the end of the program is reached. This system is not very convenient when there are repeat computations to be performed or when it is desirable to transfer execution to another formula. It is in these cases, however, that the jumps commands are very effective. There are three types of jump commands: a simple unconditional jump to a branch destination, conditional jumps that decide the branch destination by whether a certain condition is true or not, and count jumps that increase or decrease a specific memory by one and then decide the branch destination after checking whether the value stored equals zero or not.

◆ Unconditional jump

The unconditional jump is composed of "Goto" and "Lbl". When program execution reaches the statement "Goto n" (where n is a number from 0 through 9), execution then jumps to "Lbl n" (n is the same value as Goto n). The unconditional jump is often used in simple programs to return execution to the beginning for repetitive computations, or to repeat computations from a point within a program.
Unconditional jumps are also used in combination with conditional and count jumps.

Example: The previously presented program to find the surface area and volume of a regular tetrahedron will be rewritten using "Goto 1" and "Lbl 1" to allow repeat computations.

The previous program contained:

?, →, A, :, $\sqrt{\ }$, 3, ×, A, x^2, ◢,
$\sqrt{\ }$, 2, ÷, 1, 2, ×, A, x^y, 3 19 steps

* *Hereinafter, commas (,) will be used to separate steps for the sake of clarity.*

Add "Goto 1" to the end of the program, and add "Lbl 1" to the beginning of the program as the branch destination.
If this is simply left the way it is, however, the volume will not be displayed and execution will move immediately to the input of one side at the beginning. To prevent this situation, insert a display command (◢) in front of the "Goto 1".

The complete program with the unconditional jump added should look like this:

Lbl, 1, :, ?, →, A, :, $\sqrt{\ }$, 3, ×, A, x^2, ◢,
$\sqrt{\ }$, 2, ÷, 1, 2, ×, A, x^y, 3, ◢, Goto, 1 25 steps

Now let's try executing this program.

* *For details on inputting programs and editing programs, see sections 4-1 and 4-2.*

**Henceforth, the displays will only show computation result output.*

Operation	Display	
[Prog] [0] [EXE]	?	Stored in P0.
10 [EXE]	173.2050808	The length of the side=10
[EXE]	117.8511302	
[EXE]	?	
7.5 [EXE]	97.42785793	The length of the side=7.5
[EXE]	49.71844555	
[EXE]	?	

Since the program is in an endless loop, it will continue execution. To terminate execution, press [MODE] [1].

[MODE] [1]

```
**** MODE ****

sys mode  : RUN
cal mode  : COMP
   angle  : Deg
 display  : Norm

   Step     0
```

Besides the beginning of the program, branch destinations can be designated at any point within the program.

Example: Compute y=ax+b when the value for x changes each time, while a and b can also change depending upon the computation.

Program

?, →, A, :, ?, →, B, :, Lbl, 1, :, ?, →, X, :,
A, ×, X, +, B, ◢, Goto, 1 23 steps

When this program is executed, the values for a and b are stored in memories A and B respectively. After that, only the value for x can be changed.
In this way an unconditional jump is made in accordance with "Goto" and "Lbl", and the flow of program execution is changed. When there is no "Lbl n" to correspond to a "Goto n", an error (Go ERROR) is generated.

◆ Conditional jumps

The conditional jumps compare a numeric value in memory with a constant or a numeric value in another memory. If the condition is true, the statement following the "⇒" is executed, and if the condition is not true, execution skips the statement and continues following the next "↵", ":" or "◢".
Conditional jumps take on the following form:

Left side Relational operator Right side ⇒ Statement {↵ : ◢}* Statement

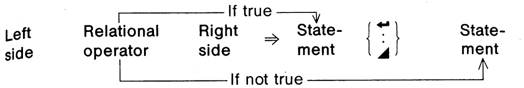
---

* ↵ *represents carriage return function (see page 109).*
* *Anyone can be used.*

One memory name (alphabetic character from A through Z), constant numeric values or computation formulas (A×2, D−E, etc.) are used for "left side" and "right side".
The relational operator is a comparison symbol. There are 6 types of relational operators: =, ≠, ≥, ≤, >, <.

Left side = right side (left side equals right side)
Left side ≠ right side (left side does not equal right side)
Left side ≥ right side (left side is greater than or equal to right side)
Left side ≤ right side (left side is less than or equal to right side)
Left side > right side (left side is greater than right side)
Left side < right side (left side is less than right side)

The "⇒" is displayed when SHIFT 7 are pressed. If the condition is true, execution advances to the statement following ⇒. If the condition is not true, the statement following ⇒ is skipped and execution jumps to the statement following the next "↵", ":" or "◢".



A statement is a computation formula (sin A×5, etc.) or a program command (Goto, Prog, etc.), and everything up to the next "↵", ":" or "◢" is regarded as one statement.

Example: If an input numeric value is greater than or equal to zero, compute the square root of that value. If the input value is less than zero, reinput another value.

Program

Lbl, 1, :, ?, →, A, :, A, ≥, 0, ⇒, √‾, A, ◢, Goto, 1
16 steps

In this program, the input numeric value is stored in memory A, and then it is tested to determine whether it is greater than, equal to or less than zero. If the contents of memory A are greater than or equal to 0 (not less than zero), the statement (computation formula) located between "⇒" and "◢" will be executed, and then Goto 1 returns execution to Lbl 1. If the contents of memory A are less than zero, execution will skip the following statement to the next "◢" and returned to Lbl 1 by Goto 1.

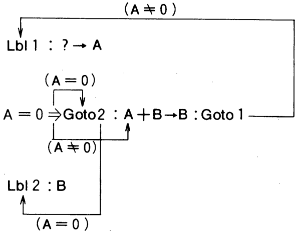
Example: Compute the sum of input numeric values. If a 0 is input, the total should be displayed.

Program

0, →, B, :,
Lbl, 1, :, ?, →, A, :, A, =, 0, ⇒, Goto, 2, :,
A, +, B, →, B, :, Goto, 1, :,
Lbl, 2, :, B 31 steps

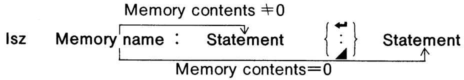
In this program, a 0 is first stored in memory B to clear it for computation of the sum. Next, the value input by "?→A" is stored in memory A by "A=0⇒" and it is determined whether or not the value stored in memory A equals zero. If A=0, Goto 2 causes execution to jump to Lbl 2. If memory A does not equal 0, Goto 2 will be skipped and the command A+B→B which follows ":" is executed, and then Goto 1 returns execution to Lbl 1.
Execution from Lbl 2 will display the sum that has been stored in memory B. Actually, the display command "◢" is inserted following B, but here it can be omitted.

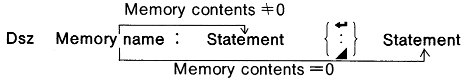
The following illustration shows the flow of the program:



◆ Count jumps

The count jumps cause the value in a specified memory to be increased or decreased by 1. If the value does equal 0, the following statement is skipped, and the statement following the next "↵", " : " or "◢" is executed. The "Isz" command is used to increase the value in memory by 1 and decide the subsequent execution, while the "Dsz" command is used to decrease the value by 1 and decide.





Example: Increase memory A by one ······ Isz A
Decrease memory B by one ······ Dsz B

Example: Determine the average of 10 input numeric values.

Program

```
1 , 0, →, A, : , 0, →, C, : ,
Lbl, 1 , : , ? , →, B, : , B, +, C, →, C, : ,
Dsz, A, : , Goto, 1 , : , C, ÷, 1 , 0                32 steps
```

In this program, first 10 is stored in memory A, and 0 is stored in memory C. Memory A is used as the "counter" and countdown is performed the specified number of times by the Dsz command. Memory C is used to store the sum of the inputs, and so first must be cleared by inputting a 0. The numeric value input in response to "?" is stored in memory B, and then the sum of the input values is stored in memory C by "B+C→C ". The statement Dsz A then decreases the value stored in memory A by 1. If the result does not equal 0, the following statement, Goto 1 is executed. If the result equals 0, the following Goto 1 is skipped and "C÷10" is executed.

Example: **Determine the altitude at one-second intervals of a ball thrown into the air at an initial velocity of Vm/sec and an angle of S°. The formula is expressed as: $h=V\sin\theta\, t-\frac{1}{2}gt^2$, with $g=9.8$, with the effects of air resistance being disregarded.**

Program

```
Deg, : , 0, →, T, : , ? , →, V, : , ? , →, S, : ,
Lbl, 1 , : , Isz, T, : , V, ×, sin, S, ×, T, −,
9 , • , 8 , ×, T, x², ÷, 2 , ◢, Goto, 1                38 steps
```

In this program the unit of angular measurement is set and memory T is first initialized (cleared). Then the initial velocity and angle are input into memories V and S respectively.

Lbl 1 is used at the beginning of the repeat computations. The numeric value stored in memory T is counted up (increased by 1) by Isz T. In this case, the Isz command is used only for the purpose of increasing the value stored in memory T, and the subsequent jump does not depend upon any comparison or decision. The Isz command can also be used in the same manner as seen with the Dsz command for jumps that require decisions, but, as can be seen here, it can also be used to simply increase values. If, in place of the Isz command, another method such as "T+1→T" is used, five steps are required instead of the two for the (Isz T) method shown here. Such commands are convenient ways of conserving memory space.

Each time memory T is increased, computation is performed according to the formula, and the altitude is displayed. It should be noted that this program is endless, so when the required value is obtained, MODE 1 are pressed to terminate the program.

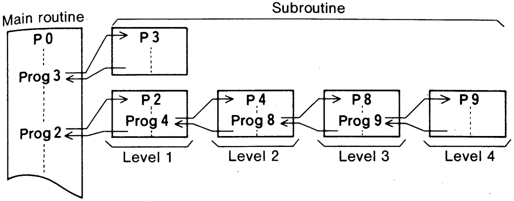
〈Summary〉

Command	Formula	Operation
Unconditional jump	Lbl *n* Goto *n* (*n*=natural number from 0 through 9)	Performs unconditional jump to Lbl *n* corresponding to Goto *n*.
Conditional jumps	Left side Relational operator Right side ⇒ Statement { ↵ : ◢ } Statement (Relational operators: =, ≠, >, <, ≥, ≤)	Left and right sides are compared. If the conditional expression is true, the statement after ⇒ is executed. If not true, execution jumps to the statement following the next ↵, : or ◢. Statements include numeric expressions, Goto commands, etc.
Count jumps	Isz Memory name: Statement { ↵ : ◢ } Statement Dsz Memory name: Statement { ↵ : ◢ } Statement (Memory name consists of single character from A through Z, A[ ], etc.)	Numeric value stored in memory is increased (Isz) or decreased (Dsz) by one. If result equals 0, a jump is performed to the statement following the next ↵, : or ◢. Statements include numeric expressions, Goto commands, etc.

■ Subroutines

A program contained in a single program area is called a "main routine". Often used program segments stored in other program areas are called "subroutines".

Subroutines can be used in a variety of ways to help make computations easier. They can be used to store formulas for repeat computations as one block to be jumped to each time, or to store often used formulas or operations for call up as required.



The subroutine command is "Prog" followed by a number from 0 through 9 which indicates the program area.

Example: Prog 0 ······Jump to program area 0
Prog 2 ······Jump to program area 2

After the jump is performed using the Prog command, execution continues from the beginning of the program stored in the specified program area. After execution reaches the end of the subroutine, the program returns to the statement following the Prog *n* command in the original program area. Jumps can be performed from one subroutine to another, and this procedure is known as "nesting". Nesting can be performed to a maximum of 10 levels, and attempts to exceed this limit will cause an error (Ne ERROR) to be generated. Attempting to use Prog to jump to a program area in which there is no program stored will also result in an error (Go ERROR).

* *A Goto n contained in a subroutine will jump to the corresponding Lbl n contained in that program area.*

Example: Simultaneously execute the two previously presented programs to compute the surface areas and volumes of a regular octahedron and tetrahedron.
Express the result in three decimal places.

This example employs two previously explained programs, and the first step is to input the specified number of decimal places (MODE 7 3).

Now let's review the two original programs.

Regular octahedron

P0 Fix, 3, :, ?, →, A, :, 2, ×, √‾‾, 3, ×, A, x^2, ◢,
√‾‾, 2, ÷, 3, ×, A, x^y, 3 23 steps

Regular tetrahedron

P1 Fix, 3, :, ?, →, A, :, √‾‾, 3, ×, A, x^2, ◢,
√‾‾, 2, ÷, 1, 2, ×, A, x^y, 3 22 steps
Total: 45 steps

If the two programs are compared, it is evident that the underlined portions are identical. If these portions are incorporated into a common subroutine, the programs are simplified and the number of steps required is decreased.

Furthermore, the portions indicated by the wavy line are not identical as they stand, but if P1 is modified to: √‾‾, 2, ÷, 3, ×, A, x^y, 3, ÷, 4, the two portions become identical.

Now the portions underlined by the straight line will be stored as an independent routine in P9 and those underlined with the wavy line will be stored in P8.

P9 Fix, 3, :, ?, →, A, :, √‾‾, 3, ×, A, x^2 12 steps
P8 √‾‾, 2, ÷, 3, ×, A, x^y, 3 8 steps

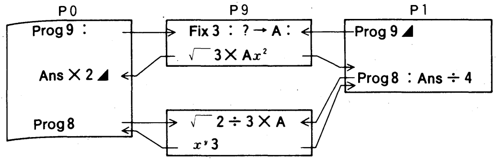
After the common segments have been removed, the remainder of the regular octahedron formula is stored in P0, and that of the regular tetrahedron is stored in P1. Of course, the "Prog 9" and "Prog 8" must be added to jump to subroutines P9 and P8.

P0 Prog, 9, :, Ans, ×, 2, ◢, Prog, 8 9 steps
P1 Prog, 9, ◢, Prog, 8, :, Ans, ÷, 4 9 steps
Total: 38 steps

With this configuration, execution jumps to program P9 at the beginning of programs P0 and P1, three decimal places are specified, the value for one side is entered, and the surface area of the tetrahedron is computed. The expression "2×" of the original octahedron formula was omitted in P9, so when execution returns to P0, "Ans×2" is used to obtain the surface of the octahedron. In the case of P1, the result of P9 needs no further modification and so is immediately displayed upon return to P1.

Computation of the volumes is also performed in a similar manner. After a jump is made to P8 for computation, execution returns to the main routines. In P0, the program ends after the volume of the octahedron is displayed. In P1, however, the result computed in P8 is divided by four to obtain the volume of the tetrahedron. By using subroutines in this manner, steps can be shortened and programs become neat and easy to read.

The following illustration shows the flow of the program just presented.



By isolating the common portions of the two original programs and storing them in separate program areas, steps are shortened and programs take on a clear configuration.

■ Carriage return function

With the carriage return function, EXE is used in place of : to separate commands to produce easy-to-read displays.

```
Deg:0→T:?→V:?→S:
Lbl 1:Isz T:V×si
n S×T−9.8×T²÷2◢
Goto 1
```

Using the carriage return function in the program shown above produces the following display:

```
Deg
0→T:?→V:?→S
Lbl 1:Isz T:V×si
n S×T−9.8×T²÷2◢
Goto 1
```

EXE pressed at these two locations. Nothing is displayed at the point where EXE is pressed, and the display advances to the next line.

This makes angle unit setting and looped operations, etc. easier to follow.

- **Operation procedure**

[MODE] [4] [EXE] (Press in place of [:])

[0] [→] [ALPHA] [T] [:] [SHIFT] [?] [→] [ALPHA] [V] [:] [SHIFT] [?] [→] [ALPHA] [S] [EXE]

[SHIFT] [Lbl] [1] [:] ······

* To include the carriage return function in a program that has already been input, first press [SHIFT] [INS] to specify the insert mode and then press [EXE]. Then, delete the " : ".

```
Deg:0→T:?→V:?→S:
Lbl 1:Isz T:V×si
n S×T−9.8×T²÷2◢
Goto 1
```

Align the cursor with the " : " following "Deg" and press [SHIFT] [INS] [EXE].

[⇨] [SHIFT] [INS] [EXE]

```
Deg
:0→T:?→V:?→S:Lbl
 1:Isz T:V×sin S
×T−9.8×T²÷2◢
Goto 1
```

Delete the " : ".

[DEL]

```
Deg
0→T:?→V:?→S:Lbl
1:Isz T:V×sin S×
T−9.8×T²÷2◢
Goto 1
```

Align the cursor with the " : " following "?→S". As above, first insert [EXE] and then delete the " : ".

[⇨] ~ [⇨] [SHIFT] [INS]
[EXE] [DEL]

```
Deg
0→T:?→V:?→S
Lbl 1:Isz T:V×si
n S×T−9.8×T²÷2◢
Goto 1
```

* Carriage return can be used in manual operations by pressing [SHIFT] [EXE].

4-8 ARRAY-TYPE MEMORIES

Using array-type memories

Up to this point all of the memories used have been referred to by single alphabetic characters such as A, B, X, or Y.

With the array-type memory introduced here, a memory name (one alphabetic character from A through Z) is appended with a subscript such as [1] or [2].

** Brackets are input by* [ALPHA] [·] *and* [ALPHA] [EXP].

Standard memory	Array-type memory	
A	A[0]	C[−2]
B	A[1]	C[−1]
C	A[2]	C[0]
D	A[3]	C[1]
E	A[4]	C[2]

Proper utilization of subscripts shortens programs and makes them easier to use. Negative values used as subscripts are counted in relation to memory zero as shown above.

Example: Input the numbers 1 through 10 into memories A through J.

Using standard memories

1, →, A, :, 2, →, B, :, 3, →, C, :, 4, →, D, :,
5, →, E, :, 6, →, F, :, 7, →, G, :, 8, →, H, :,
9, →, I, :, 1, 0, →, J 40 steps

Using array-type memories

0, →, Z, :, Lbl, 1, :, Z, +, 1, →, A, [, Z,], :,
Isz, Z, :, Z, <, 1, 0, ⇒, Goto, 1 26 steps

In the case of using standard memories, inputting values into memories one by one is both inefficient and time consuming. What happens, if we want to see a value stored in a specific memory?

Using standard memories

```
Lbl, 1 , :, ?, →, Z, :,
Z, =, 1 , ⇒, A, ◢, Z, =, 2 , ⇒, B, ◢,
Z, =, 3 , ⇒, C, ◢, Z, =, 4 , ⇒, D, ◢,
Z, =, 5 , ⇒, E, ◢, Z, =, 6 , ⇒, F, ◢,
Z, =, 7 , ⇒, G, ◢, Z, =, 8 , ⇒, H, ◢,
Z, =, 9 , ⇒, I, ◢, Z, =, 1 , 0, ⇒, J, ◢,
Goto, 1                                          70 steps
```

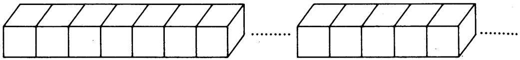
Using array-type memories

```
Lbl, 1 , :, ?, →, Z, :, A, [, Z, —, 1 , ], ◢,
Goto, 1                                          16 steps
```

The difference is readily apparent. When using the standard memories, the input value is compared one by one with the value assigned to each memory (i.e. A=1, B=2, . .).
With the array-type memories, the input value is immediately stored in the proper memory determined by "[Z—1]". Formulas (Z—1, A+10, etc.) can even be used for the subscript.

■ Cautions when using array-type memories

When using array-type memories, a subscript is appended to an alphabetic character that represents a standard memory from A through Z.
Therefore, care must be taken to prevent overlap of memories.
The relation is as follows:



A[0] A[1] A[2] A[3] A[4] A[5] A[6]
B[-1] B[0] B[1] B[2] B[3] B[4] B[5]
C[-2] C[-1] C[0] C[1] C[2] C[3] C[4]
G[-6] G[-5] G[-4] G[-3] G[-2] G[-1] G[0]

A[23] A[24] A[25] A[26] A[27]
B[22] B[23] B[24] B[25] B[26]
C[21] C[22] C[23] C[24] C[25]
G[17] G[18] G[19] G[20] G[21]
X[0] X[1] X[2] X[3] X[4]
Y[-1] Y[0] Y[1] Y[2] Y[3]
Z[-2] Z[-1] Z[0] Z[1] Z[2]

The following shows a case in which array-type memories overlap with standard format memories. This situation should always be avoided.

Example: Store the numeric values from 1 through 5 in memories A[1] through A[5] respectively.

```
5 , →, C, :, Lbl, 1 , :, C, →, A, [, C, ], :,
Dsz, C, :, Goto, 1 , :,
A, [, 1 , ], ◢, A, [, 2 , ], ◢, A, [, 3 , ], ◢,
A, [, 4 , ], ◢, A, [, 5 , ]                        44 steps
```

In this program, the values 1 through 5 are stored in the array-type memories A[1] through A[5], and memory C is used as a counter memory. When this program is executed, the following results are obtained:

Operation	Display
Prog 0 EXE	1.
EXE	0.
EXE	3.
EXE	4.
EXE	5.

As can be seen, the second displayed value (which should be 2) in A[2] is incorrect. This problem has occurred because memory A[2] is the same as memory C.

A	B	C	D	E	F
	A[1]	A[2]	A[3]	A[4]	A[5]

The content of memory C (A[2]) is decreased from 5 to 0 in steps of 1. Therefore, the content of memory A[2] is displayed as 0.

■ Application of the array-type memories

It is sometimes required to treat two different types of data as a single group. In this case, memories for data processing and those for data storage should be kept separate.

Example: **Store data x and y in memories. When an x value is input, the corresponding y value is displayed. There will be a total of 15 pieces of data.**

Example program 1

Memory A is used as the data control memory, and memory B is used for temporary storage of the x data. The x data are stored in memories C[1] (memory D) through C[15] (memory R), and the y data are stored in memories C[16] (memory S) through C[30] (memory Z[7]).

```
1 , →, A, :, Defm, 7 , :,
Lbl, 1 , :, ? , →, C, [, A, ], :,
? , →, C, [, A, +, 1 , 5 , ], :,
Isz, A, :, A, =, 1 , 6 , ⇒, Goto, 2 , :, Goto, 1 , :,
Lbl, 2 , :, 1 , 5 , →, A, :, ? , →, B, :,
B, =, 0, ⇒, Goto, 5 , :,
Lbl, 3 , :, B, =, C, [, A, ], ⇒, Goto, 4 , :,
Dsz, A, :, Goto, 3 , :, Goto, 2 , :,
Lbl, 4 , :, C, [, A, +, 1 , 5 , ], ◢, Goto, 2 , :,
Lbl, 5                                              98 steps
```

In this program, memories are used as follows:

x data

C [1]	C [2]	C [3]	C [4]	C [5]	C [6]	C [7]	C [8]
D	E	F	G	H	I	J	K
C [9]	C [10]	C [11]	C [12]	C [13]	C [14]	C [15]	
L	M	N	O	P	Q	R	

y data

C [16]	C [17]	C [18]	C [19]	C [20]	C [21]	C [22]	C [23]
S	T	U	V	W	X	Y	Z
C [24]	C [25]	C [26]	C [27]	C [28]	C [29]	C [30]	
Z (1)	Z (2)	Z (3)	Z (4)	Z (5)	Z (6)	Z (7)	

Example program 2

The same memories are used as in Example 1, but two types of memory names are used and the x and y data kept separate.

```
1 , →, A, :, Defm, 7 , :,
Lbl, 1 , :, ? , →, C, [, A, ], :,
? , →, R, [, A, ], :,
Isz, A, :, A, =, 1 , 6 , ⇒, Goto, 2 , :, Goto, 1 , :,
Lbl, 2 , :, 1 , 5 , →, A, :, ? , →, B, :,
B, =, 0, ⇒, Goto, 5 , :,
Lbl, 3 , :, B, =, C, [, A, ], ⇒, Goto, 4 , :,
Dsz, A, :, Goto, 3 , :, Goto, 2 , :,
Lbl, 4 , :, R, [, A, ], ◢, Goto, 2 , :,
Lbl, 5                                              92 steps
```

Memories are used as follows:

x data

C [1]	C [2]	C [3]	C [4]	C [5]	C [6]	C [7]	C [8]
D	E	F	G	H	I	J	K
C [9]	C [10]	C [11]	C [12]	C [13]	C [14]	C [15]	
L	M	N	O	P	Q	R	

y data

R [1]	R [2]	R [3]	R [4]	R [5]	R [6]	R [7]	R [8]
S	T	U	V	W	X	Y	Z
R [9]	R [10]	R [11]	R [12]	R [13]	R [14]	R [15]	
Z (1)	Z (2)	Z (3)	Z (4)	Z (5)	Z (6)	Z (7)	

In this way, the memory names can be changed. However, since memory names are restricted to the letters from A through Z, the expanded memories (MODE ⊡) can only be used as array-type memories.

* *The memory expansion command (Defm) can be used in a program.*

Example: **Expand the number of memories by 14 to make a total of 40 available.**

```
Defm, 1, 4, :, ......
```

4-9 DISPLAYING ALPHA-NUMERIC CHARACTERS AND SYMBOLS

Alphabetic characters, numbers, computation command symbols, etc. can be displayed as messages. They are enclosed in quotation marks ([ALPHA] [Prog]).

■ Alpha-numeric characters and symbols

- **Characters and symbols displayed when pressed following [ALPHA] :**
  [,], k, m, μ, n, p, f, space,
  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, Z
- **Other numbers, symbols, calculation commands, program commands**
  0, 1, 2, 3, 4, 5, 6, 7, 8, 9,
  (,), √, E, +, −, ×, ÷, ...
  sin, cos, tan, log, ln, ...
  =, ≠, ≥, ≤, >, <, ...
  A, B, C, D, E, F, d, h, b, o
  Neg, Not, and, or, xor
  $\bar{x}$, $\bar{y}$, $x\sigma_n$, $x\sigma_{n-1}$, ...
  ° ([SHIFT] [MODE] [4]), ' ([SHIFT] [MODE] [5]), ^{g} ([SHIFT] [MODE] [6])

* *All of the above noted characters can be used in the same manner as the alphabetic characters.*

In the preceding example requiring an input of two types of data (x, y), the prompt "?" does not give any information concerning the type of input expected. A message can be inserted before the "?" to verify the type of data required for input.

Lbl, 1, :, ?, →, X, :, ?, →, Y, :, ...

The messages "X=" and "Y=" will be inserted into this program.

Lbl, 1, :, ", X, =, ", ?, →, X, :,
", Y, =, ", ?, →, Y, :, ...

If messages are included as shown here, the display is as follows:
(Assuming that the program is stored in P1)

[Prog] 1 [EXE]	X= ?
10 [EXE]	Y= ?
⋮	⋮

Messages are also convenient when displaying result in program computations.

Example:

Lbl, 0, :, ", N, =, ", ?, →, B, ~, C, :,
0, →, A, :,
Lbl, 1, :, C, ÷, 2, →, C, :, Frac, C, ≠, 0, ⇒, Goto, 3,
:, Isz, A, :, C, =, 1, ⇒, Goto, 2, :, Goto, 1, :,
Lbl, 2, :, ", X, =, ", ◢, A, ◢, Goto, 0, :,
Lbl, 3, :, ", N, O, ", ◢, Goto, 0 70 steps

This program computes the x power of 2. A prompt of "N=?" appears for data input. The result is displayed by pressing [EXE] while "X=" is displayed. When an input data is not the x power of 2, the display "NO" appears and execution returns to the beginning for reinput.

* ***Always follow a message with a ◢ whenever a formula follows the message.***

Assuming that the program is stored in P2:

[Prog] 2 [EXE]	N= ?
4096 [EXE]	X=
[EXE]	12.
[EXE]	N= ?
3124 [EXE]	NO
[EXE]	N= ?
512 [EXE]	X=
[EXE]	9.

Strings longer than 16 characters are displayed in two lines. When alphabetic characters are displayed at the end of the bottom line, the entire display shifts upwards and the uppermost line disappears from the display.

Prog 0

```
123+45
            168.
852-87
            765.
968+125-65
           1028.
Prog 0
```

EXE

```
123+45
            168.
852-87
            765.
968+125-65
           1028.
Prog 0
ABCDEFGHIJKLMNOP
```

↓ After a while

```
            168.
852-87
            765.
968+125-65
           1028.
Prog 0
ABCDEFGHIJKLMNOP
QRSTUVWXYZ
```

4-10 USING THE GRAPH FUNCTION IN PROGRAMS

Using the graph function within programs makes it possible to graphically represent long, complex equations and to overwrite graphs repeatedly. All graph commands (except the trace function) can be included in programs. Range values can also be written into the program.

Generally, manual graph operations can be used in programs without modification.

Ex. 1) **Graphically determine the number of solutions (real roots) that satisfy both of the following two equations.**

$$y=x^4-x^3-24x^2+4x+80$$
$$y=10x-30$$

The range values are as follows:

```
Range
Xmin:-10.
 max:10.
 scl:2.
Ymin:-120.
 max:150.
 scl:50.
```

First, program the range settings. Note that values are separated from each other by commas “, ”.

Range, (−), 1, 0, , , 1, 0, , , 2, , , (−), 1, 2, 0, , , 1, 5, 0, , , 5, 0

Next, program the equation for the first graph.

Graph, X, x^y, 4, −, X, x^y, 3, −, 2, 4, X, x^2, +, 4, X, +, 8, 0

Finally, program the equation for the second graph.

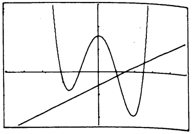
Graph, 1, 0, X, −, 3, 0 Total 49 steps

When inputting this program, press EXE after input of the ranges and the first equation.

```
Range -10,10,2,-
120,150,50
Graph Y=Xxʸ 4-Xxʸ
3-24X²+4X+80
Graph Y=10X-30_
```

The following should appear on the display when the program is executed:

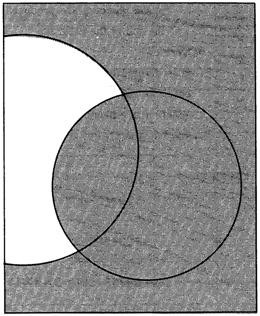
Prog 0 EXE



A "◢" can be input in place of the EXE key operation after the first equation to suspend execution after the first graph is produced. To continue execution to the next graph, press EXE.

The procedure outlined above can be used to produce a wide variety of graphs.
The library at the end of this manual includes a number of examples of graph programming.

PROGRAM LIBRARY



〈Prior to use〉

- Always check the number of remaining steps before attempting to store programs.
- The library is divided into two parts: a calculation section and a graph section. The calculation section shows only answers, while the graph section shows whole displays.
- To make programs in the graph section easier to follow, ↵ is used to indicate carriage returns. The EXE key should be pressed wherever ↵ appears (↵ does not appear on the display).
- Press the Graph key whenever "Graph" appears within a program (Graph Y = indicated).
- If it is necessary to specify a calculation mode (e.g. Base-n, SD1) in a program, be sure to specify it after pressing MODE 2 (WRT mode).
  Then start programming by pressing EXE.

CASIO PROGRAM SHEET

Program for	No.
Prime factor analysis	1

Description

Prime factors of arbitrary positive integers are produced.

For $1 < m < 10^{10}$

prime numbers are produced from the lowest value first. "END" is displayed at the end of the program.

〈Overview〉

m is divided by 2 and by all successive odd numbers ($d = 3, 5, 7, 9, 11, 13, \ldots\ldots$) to check for divisibility.

Where d is a prime factor, $m_i = m_{i-1}/d$ is assumed, and division is repeated until $\sqrt{m_i}+1 \leqq d$.

Example

〈1〉

$119 = 7 \times 17$

〈2〉

$1234567890 = 2 \times 3 \times 3 \times 5 \times 3607 \times 3803$

〈3〉

$987654321 = 3 \times 3 \times 17 \times 17 \times 379721$

Preparation and operation

- ●Store the program written on the next page.
- ●Execute the program as shown below in the RUN mode (MODE 1).

Step	Key operation	Display	Step	Key operation	Display
1	Prog 0 EXE	M ?	11	EXE	3803.
2	119 EXE	7.	12	EXE	END
3	EXE	17.	13	EXE	M ?
4	EXE	END	14	987654321 EXE	3.
5	EXE	M ?	15	EXE	3.
6	1234567890 EXE	2.	16	EXE	17.
7	EXE	3.	17	EXE	17.
8	EXE	3.	18	EXE	(After 12 seconds) 379721.
9	EXE	5.	19	EXE	END
10	EXE	(After 74 seconds) 3607.	20		

No. 1

Line	MODE 2 Program															Notes	Number of steps
1	Mcl	:															2
2	Lbl	0	:	"	M	"	?	→	A	:	Goto	2	:				15
3	Lbl	1	:	2	◢	A	÷	2	→	A	:	A	=	1	⇒		30
4	Goto	9	:														33
5	Lbl	2	:	Frac	(	A	÷	2	)	=	0	⇒	Goto	1	:		48
6	3	→	B	:													52
7	Lbl	3	:	√	A	+	1	→	C	:							62
8	Lbl	4	:	B	≥	C	⇒	Goto	8	:	Frac	(	A	÷	B		77
9	)	=	0	⇒	Goto	6	:										84
10	Lbl	5	:	B	+	2	→	B	:	Goto	4	:					96
11	Lbl	6	:	A	÷	B	×	B	−	A	=	0	⇒	Goto	7		111
12	:	Goto	5	:													115
13	Lbl	7	:	B	◢	A	÷	B	→	A	:	Goto	3	:			129
14	Lbl	8	:	A	◢												134
15	Lbl	9	:	"	E	N	D	"	◢	Goto	0						145
16																	
17																	
18																	
19																	
20																	
21																	
22																	
23																	
24																	
25																	
26																	
27																	
28																	

Memory contents	A	m_i	H		O		V	
	B	d	I		P		W	
	C	$\sqrt{m_i}+1$	J		Q		X	
	D		K		R		Y	
	E		L		S		Z	
	F		M		T			
	G		N		U			

CASIO PROGRAM SHEET

Program for **Definite integrals using Simpson's rule** No. 2

Description

$$I=\int_a^b f(x)dx=\frac{h}{3}\{y_0+4(y_1+y_3+\dots+y_{2m-1})+2(y_2+y_4+\dots+y_{2m-2})+y_{2m}\}$$

$$h=\frac{b-a}{2m}$$

The right-hand portion of the above equation can be transformed as follows.

$$I=\frac{h}{3}\{y_0+\sum_{i=1}^{m}(4y_{2i-1}+2y_{2i})-y_{2m}\}$$

Let $f(x)=\frac{1}{x^2+1}$

Example

⟨1⟩ $a=0,\ b=1,\ 2m=10$

$$I=\int_0^1\frac{1}{x^2+1}d_x=0.7853981537$$

⟨2⟩ $a=2,\ b=5,\ 2m=20$

$$I=\int_2^5\frac{1}{x^2+1}d_x=0.2662526769$$

Preparation and operation

- Store the program written on the next page.
- Execute the program as shown below in the RUN mode (MODE 1).

Step	Key operation	Display	Step	Key operation	Display
1	Prog 0 EXE	A ?	11		
2	0 EXE	B ?	12		
3	1 EXE	2 M ?	13		
4	10 EXE	0.7853981535	14		
5	EXE	A ?	15		
6	2 EXE	B ?	16		
7	5 EXE	2 M ?	17		
8	20 EXE	0.2662526769	18		
9			19		
10			20		

Line	MODE 2 Program															Notes	Number of steps
1	P 0																
2	Lbl	1	:	Mcl	:												5
3	"	A	"	?	→	A	:	"	B	"	?	→	B	:	"		20
4	2	M	"	?	→	M	:										27
5	A	→	G	:	Prog	1	:	P	→	I	:	(	B	−	A		42
6	)	÷	M	→	D	:	M	÷	2	→	O	:					54
7	Lbl	2	:	G	+	D	→	G	:	Prog	1	:	I	+	P		69
8	×	4	→	I	:												74
9	G	+	D	→	G	:	Prog	1	:	I	+	P	×	2	→		89
10	I	:	O	−	1	→	O	:									97
11	O	≠	0	⇒	Goto	2	:										104
12	B	→	G	:	Prog	1	:	I	−	P	→	I	:				117
13	D	×	I	÷	3	◢											123
14	Goto	1															125
15																	
16	P 1																
17	1	÷	(	G	×	G	+	1	)	→	P						11
18																	
19																Total 136 steps	
20																	
21																	
22																	
23																	
24																	
25																	
26																	
27																	
28																	

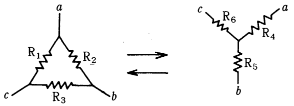
Memory contents	A	a	H		O	m (Number of repetitions)	V	
	B	b	I	I	P		W	
	C		J		Q		X	
	D	$h=\frac{b-a}{2m}$	K		R		Y	
	E		L		S		Z	
	F		M	$2m$	T			
	G	x	N		U			

CASIO PROGRAM SHEET

Program for △ ↔ Y transformation

No. 3

Description



1) △→Y

$$R_4=\frac{R_1\cdot R_2}{R_1+R_2+R_3}$$

$$R_5=\frac{R_2\cdot R_3}{R_1+R_2+R_3}$$

$$R_6=\frac{R_3\cdot R_1}{R_1+R_2+R_3}$$

2) Y→△

$$R_1=\frac{R_4R_5+R_5R_6+R_6R_4}{R_5}$$

$$R_2=\frac{R_4R_5+R_5R_6+R_6R_4}{R_6}$$

$$R_3=\frac{R_4R_5+R_5R_6+R_6R_4}{R_4}$$

Example

〈1〉

$R_1=12(\Omega)$

$R_2=47(\Omega)$

$R_3=82(\Omega)$

〈2〉

$R_4=100(\Omega)$

$R_5=150(\Omega)$

$R_6=220(\Omega)$

Preparation and operation

- ●Store the program written on the next page.
- ●Execute the program as shown below in the RUN mode (MODE 1).

Step	Key operation	Display	Step	Key operation	Display
1	Prog 0 EXE	D→Y:1,Y→D:2?	11	EXE	D→Y:1,Y→D:2?
2	1 EXE	R 1= ?	12	2 EXE	R 4= ?
3	12 EXE	R 2= ?	13	100 EXE	R 5= ?
4	47 EXE	R 3= ?	14	150 EXE	R 6= ?
5	82 EXE	R 4=	15	220 EXE	R 1=
6	EXE	4.	16	EXE	466.6666667
7	EXE	R 5=	17	EXE	R 2=
8	EXE	27.33333333	18	EXE	318.1818182
9	EXE	R 6=	19	EXE	R 3=
10	EXE	6.978723404	20	EXE	700.

No. 3

Line	MODE 2							Program								Notes	Number of steps
1	Lbl	1	:	”	D	→	Y	:	1	,	Y	→	D	:	2		15
2	”	?	→	N	:												20
3	N	=	2	⇒	Goto	2	:	N	≠	1	⇒	Goto	1	:			34
4	”	R	1	=	”	?	→	A	:								43
5	”	R	2	=	”	?	→	B	:								52
6	”	R	3	=	”	?	→	C	:								61
7	A	+	B	+	C	→	D	:									69
8	”	R	4	=	”	◢	A	×	B	÷	D	◢					81
9	”	R	5	=	”	◢	B	×	C	÷	D	◢					93
10	”	R	6	=	”	◢	A	×	C	÷	D	◢					105
11	Goto	1	:														108
12	Lbl	2	:														111
13	”	R	4	=	”	?	→	E	:								120
14	”	R	5	=	”	?	→	F	:								129
15	”	R	6	=	”	?	→	G	:								138
16	E	×	F	+	F	×	G	+	G	×	E	→	H	:			152
17	”	R	1	=	”	◢	H	÷	F	◢							162
18	”	R	2	=	”	◢	H	÷	G	◢							172
19	”	R	3	=	”	◢	H	÷	E	◢							182
20	Goto	1															184
21																	
22																	
23																	
24																	
25																	
26																	
27																	
28																	

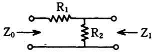
Memory contents	A	R_1	H	$R_4R_5+R_5R_6+R_6R_4$	O		V	
	B	R_2	I		P		W	
	C	R_3	J		Q		X	
	D	$R_1+R_2+R_3$	K		R		Y	
	E	R_4	L		S		Z	
	F	R_5	M		T			
	G	R_6	N	For judgement	U			

CASIO PROGRAM SHEET

Program for	No.
Minimum loss matching	4

Description

Calculate R_1 and R_2 which match Z_0 and Z_1 with loss minimized. $(Z_0 > Z_1)$



$$R_1 = Z_0\sqrt{1-\frac{Z_1}{Z_0}} \qquad R_2 = \frac{Z_1}{\sqrt{1-\frac{Z_1}{Z_0}}}$$

$$\text{Minimum loss } L_{min} = 20 \log\left(\sqrt{\frac{Z_0}{Z_1}} + \sqrt{\frac{Z_0}{Z_1} - 1}\right) [\text{dB}]$$

Example

Calculate the values of R_1, R_2 and L_{min} for $Z_0 = 500\Omega$ and $Z_1 = 200\Omega$.

Preparation and operation

- ● Store the program written on the next page.
- ● Execute the program as shown below in the RUN mode (MODE 1).

Step	Key operation	Display	Step	Key operation	Display
1	Prog 0 EXE	Z 0= ?	11		
2	500 EXE	Z 1= ?	12		
3	200 EXE	R 1=	13		
4	EXE	387.2983346	14		
5	EXE	R 2=	15		
6	EXE	258.1988897	16		
7	EXE	LMIN =	17		
8	EXE	8.961393328	18		
9			19		
10			20		

No. 4

Line	MODE 2 Program															Notes	Number of steps
1	"	Z	0	=	"	?	→	Y	:								9
2	"	Z	1	=	"	?	→	Z	:								18
3	√	(	1	−	Z	÷	Y	)	→	A	:						29
4	Y	×	A	→	R	:	Z	÷	A	→	S	:	Y	÷	Z		44
5	→	B	:	2	0	×	log	(	√	B	+	√	(	B	−		59
6	1	)	)	→	T	:											65
7	"	R	1	=	"	◢	R	◢									73
8	"	R	2	=	"	◢	S	◢									81
9	"	L	M	I	N	=	"	:	T								90
10																	
11																	
12																	
13																	
14																	
15																	
16																	
17																	
18																	
19																	
20																	
21																	
22																	
23																	
24																	
25																	
26																	
27																	
28																	

Memory contents	A	$\sqrt{1-\frac{Z_1}{Z_0}}$	H		O		V	
	B	$\frac{Z_0}{Z_1}$	I		P		W	
	C		J		Q		X	
	D		K		R	R 1	Y	Z_0
	E		L		S	R 2	Z	Z_1
	F		M		T	Lmin		
	G		N		U			

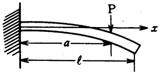
CASIO PROGRAM SHEET

Program for

Cantilever under concentrated load

No. 5

Description



E : Young's modulus 〔kg/mm²〕
I : Geometrical moment of inertia 〔mm⁴〕
a : Distance of concentrated load from support 〔mm〕
P : Load 〔kg〕
x : Distance of point of interest from the support 〔mm〕

Deflection y 〔mm〕, Angle of deflection s 〔°〕, Bending moment M 〔kg・mm〕

① $\ell > x > a$

$$y = \frac{Pa^3}{6EI} - \frac{Pa^2}{2EI}x$$

$$s = \tan^{-1}\left(-\frac{Pa^2}{2EI}\right)$$

M = 0 (shearing load Ws = 0)

② $x \leqq a$

$$y = \frac{P}{6EI}x^3 - \frac{Pa}{2EI}x^2$$

$$s = \tan^{-1}\left\{\frac{Px}{2EI}(x - 2a)\right\}$$

M = P (x − a) (shearing load Ws = P)

Example

E = 4000 kg/mm²
I = 5 mm⁴
a = 30 mm
P = 2 kg

What are deflection, angle of deflection, bending moment and shearing load at x = 25 mm and x = 32 mm ?

Preparation and operation

- Store the program written on the next page.
- Execute the program as shown below in the RUN mode (MODE 1).

Step	Key operation	Display	Step	Key operation	Display
1	Prog 0 EXE	E = ?	11	EXE	−10.
2	4000 EXE	I = ?	12	EXE	X = ?
3	5 EXE	A = ?	13	32 EXE	Y =
4	30 EXE	P = ?	14	EXE	−0.99
5	2 EXE	X = ?	15	EXE	S =
6	25 EXE	Y =	16	EXE	−2.57657183
7	EXE	−0.6770833333	17	EXE	M =
8	EXE	S =	18	EXE	0.
9	EXE	−2.505092867	19	Repeat from step 5	
10	EXE	M =	20		

No. 5

Line	MODE 2	Program														Notes	Number of steps
1	Deg	:	”	E	=	”	?	→	E	:	”	I	=	”	?		15
2	→	I	:	”	A	=	”	?	→	A	:	”	P	=	”		30
3	?	→	P	:													34
4	Lbl	1	:	”	X	=	”	?	→	X	:						45
5	X	≤	A	⇒	Goto	2	:										52
6	”	Y	=	”	◢	P	×	A	x^2	÷	(	2	×	E	×		67
7	I	)	×	(	A	÷	3	−	X	)	◢						78
8	”	S	=	”	◢	$\tan^{-1}$	(	(−)	P	×	A	x^2	÷	(	2		93
9	×	E	×	I	)	)	◢	”	M	=	”	◢	0	◢			107
10	Goto	1	:														110
11	Lbl	2	:														113
12	”	Y	=	”	◢	P	×	X	x^2	÷	(	2	×	E	×		129
13	I	)	×	(	X	÷	3	−	A	)	◢						139
14	”	S	=	”	◢	$\tan^{-1}$	(	P	×	X	÷	(	2	×	E		154
15	×	I	)	×	(	X	−	2	×	A	)	)	◢				167
16	”	M	=	”	◢	P	×	(	X	−	A	)	◢				180
17	Goto	1															182
18																	
19																	
20																	
21																	
22																	
23																	
24																	
25																	
26																	
27																	
28																	

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

CASIO PROGRAM SHEET

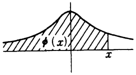
Program for **Normal distribution** No. 6

Description

Obtain normal distribution function $\phi\ (x)$ (by Hastings' best approximation).

$$\phi\ (x) = \int_{-\infty}^{t} \phi\, t d x$$

$$\phi\, t = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$$



Put $t = \frac{1}{1 + \mathrm{P}x}$

$\phi\ (x) \fallingdotseq 1 - \phi\, t\ (c_1 t + c_2 t^2 + c_3 t^3 + c_4 t^4 + c_5 t^5)$

$\mathrm{P} = 0.2316419$

$\mathrm{C}_1 = 0.31938153$

$\mathrm{C}_2 = -0.356563782$

$\mathrm{C}_3 = 1.78147937$

$\mathrm{C}_4 = -1.821255978$

$\mathrm{C}_5 = 1.330274429$

Example

Calculate the values of $\phi\ (x)$ at $x = 1.18$ and $x = 0.7$.

Preparation and operation

- Store the program written on the next page.
- Execute the program as shown below in the RUN mode ($\boxed{\text{MODE}}\ \boxed{1}$).

Step	Key operation	Display	Step	Key operation	Display
1	$\boxed{\text{Prog}}$ 0 $\boxed{\text{EXE}}$	X = ?	11		
2	1.18 $\boxed{\text{EXE}}$	PX =	12		
3	$\boxed{\text{EXE}}$	0.880999696	13		
4	$\boxed{\text{Prog}}$ 0 $\boxed{\text{EXE}}$	X = ?	14		
5	0.7 $\boxed{\text{EXE}}$	PX =	15		
6	$\boxed{\text{EXE}}$	0.7580361367	16		
7			17		
8			18		
9			19		
10			20		

No. 6

Line	$\boxed{\text{MODE}}\ \boxed{2}$ Program															Notes	Number of steps
1	"	X	=	"	?	→	X	:									8
2	1	÷	(	1	+	0	.	2	3	1	6	4	1	9	×		23
3	X	)	→	T	:	1	÷	√‾	(	2	×	π	)	×	e^x		38
4	(	(−)	X	x^2	÷	2	)	→	Q	:							48
5	"	P	X	=	"	◢	1	−	Q	×	(	0	.	3	1		63
6	9	3	8	1	5	3	×	T	+	(−)	0	.	3	5	6		78
7	5	6	3	7	8	2	×	T	x^2	+	1	.	7	8	1		93
8	4	7	9	3	7	×	T	x^y	3	+	(−)	1	.	8	2		108
9	1	2	5	5	9	7	8	×	T	x^y	4	+	1	.	3		123
10	3	0	2	7	4	4	2	9	×	T	x^y	5	)				136
11																	
12																	
13																	
14																	
15																	
16																	
17																	
18																	
19																	
20																	
21																	
22																	
23																	
24																	
25																	
26																	
27																	
28																	

Memory contents								
	A		H		O		V	
	B		I		P		W	
	C		J		Q	$\phi\, t$	X	x
	D		K		R		Y	
	E		L		S		Z	
	F		M		T	t		
	G		N		U			

CASIO PROGRAM SHEET

Program for

Graph variation by parameters

No. 7

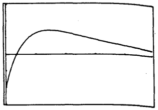
Description

Damped vibration

(i) $\varepsilon > n$ (Overdamping)

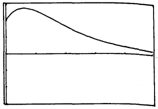
$$P_1 = -\varepsilon + \sqrt{\varepsilon^2 - n^2} \quad , \quad P_2 = -\varepsilon - \sqrt{\varepsilon^2 - n^2}$$

$$x = \frac{v_0 - x_0 P_2}{P_1 - P_2} e^{P_1 t} - \frac{v_0 - x_0 P_1}{P_1 - P_2} e^{P_2 t}$$



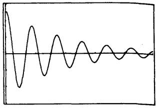
(ii) $\varepsilon = n$ (Critical damping)

$$x = \{x_0 + (v_0 + \varepsilon x_0) t\} e^{-\varepsilon t}$$



(iii) $\varepsilon < n$ (Damping vibration)

$$x = e^{-\varepsilon t} \{x_0 \cos\sqrt{n^2 - \varepsilon^2}\, t + \frac{v_0 + \varepsilon x_0}{\sqrt{n^2 - \varepsilon^2}} \cdot \sin\sqrt{n^2 - \varepsilon^2}\, t\}$$



Example

Draw a graph of the damping vibration that possesses the following parameters:

(1)	(2)	(3)
$\varepsilon = 0.1$	$\varepsilon = 0.2$	$\varepsilon = 0.2$
$n = 1.5$	$n = 0.2$	$n = 0.18$
$x_0 = 2.5$	$x_0 = 2$	$x_0 = -2$
$v_0 = 1$	$v_0 = 0.6$	$v_0 = 1.5$

Preparation and operation

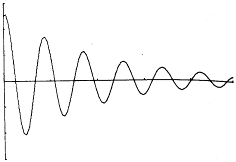
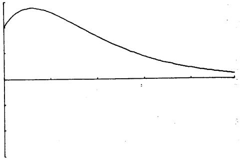
- ● Store the program written on the next page.

Memory contents	A	x_0	H		O		V	
	B	v_0	I		P	$P_1 = -\varepsilon + \sqrt{\varepsilon^2 - n^2}$	W	
	C	$\sqrt{n^2 - \varepsilon^2}$	J		Q	$P_2 = -\varepsilon - \sqrt{\varepsilon^2 - n^2}$	X	t
	D		K		R		Y	x
	E	ε	L		S		Z	
	F		M		T			
	G		N	n	U			

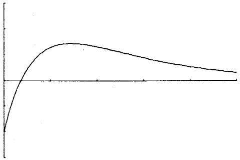
No. 7

Line	MODE 2 Program															Notes	Number of steps
1	Rad	↵															2
2	Range	0	,	2	5	,	5	,	(−)	3	,	3	,	1	↵		17
3	"	E	P	S	I	L	O	N	=	"	?	→	E	↵			31
4	"	N	=	"	?	→	N	↵									39
5	"	X	0	=	"	?	→	A	↵								48
6	"	V	0	=	"	?	→	B	↵								57
7	E	>	N	⇒	Goto	1	↵										64
8	E	=	N	⇒	Goto	2	↵										71
9	√	(	N	x^2	−	E	x^2	)	→	C	↵						82
10	Graph	e^x	(	(−)	E	X	)	(	A	cos	(	C	X	)	+		97
11	(	B	+	E	A	)	C	x^{-1}	sin	(	C	X	)	)	↵		112
12	Goto	0	↵														115
13	Lbl	1	↵														118
14	(−)	E	+	√	(	E	x^2	−	N	x^2	)	→	P	↵			132
15	(−)	E	−	√	(	E	x^2	−	N	x^2	)	→	Q	↵			146
16	Graph	(	B	−	A	Q	)	(	P	−	Q	)	x^{-1}	e^x	(		161
17	P	X	)	−	(	B	−	A	P	)	(	P	−	Q	)		176
18	x^{-1}	e^x	(	Q	X	)	↵										183
19	Goto	0	↵														186
20	Lbl	2	↵														189
21	Graph	(	A	+	(	B	+	E	A	)	X	)	e^x	(	(−)		204
22	E	X	)	↵													208
23	Lbl	0															210
24																	
25																Total 210 steps	
26																	
27																	
28																	
29																	
30																	
31																	
32																	
33																	
34																	
35																	
36																	

Program for
Graph variation by parameters
No. 7

Step	Key operation	Display
1	$\boxed{\text{Prog}}$ 0 $\boxed{\text{EXE}}$ 0.1 $\boxed{\text{EXE}}$ 1.5 $\boxed{\text{EXE}}$ 2.5 $\boxed{\text{EXE}}$	Prog 0 EPSILON=? 0.1 N=? 1.5 X0=? 2.5 V0=?
2	1 $\boxed{\text{EXE}}$	
3	$\boxed{\text{Prog}}$ 0 $\boxed{\text{EXE}}$ 0.2 $\boxed{\text{EXE}}$ 0.2 $\boxed{\text{EXE}}$ 2 $\boxed{\text{EXE}}$	Prog 0 EPSILON=? 0.2 N=? 0.2 X0=? 2 V0=?
4	0.6 $\boxed{\text{EXE}}$	

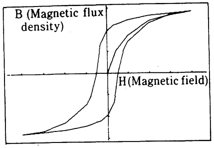
Program for
Graph variation by parameters
No. 7

Step	Key operation	Display
5	$\boxed{\text{Prog}}$ 0 $\boxed{\text{EXE}}$ 0.2 $\boxed{\text{EXE}}$ 0.18 $\boxed{\text{EXE}}$ $\boxed{(-)}$ 2 $\boxed{\text{EXE}}$	Prog 0 EPSILON=? 0.2 N=? 0.18 X0=? −2 V0=?
6	1.5 $\boxed{\text{EXE}}$	
7		
8		

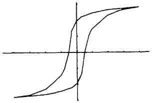
CASIO PROGRAM SHEET

Program for	No.
Hysteresis loop	8

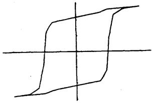
Description



When a ferromagnetic specimen is sustained in a magnetic field, the specimen becomes magnetized. The B-H relationship can be represented by a hysteresis curve.



Soft magnetic substance



Ferromagnetic substance

Example

Hysteresis curve of soft magnetic material

	1	2	3	4	5	6	7	8	9
H	0.4	1.0	2.0	3.0	4.0	2.0	1.0	0.5	0.3
B	0.5	0.86	1.2	1.32	1.4	1.31	1.22	1.13	1.1

	10	11	12	13	14	15	16	17
H	0	−0.3	−0.5	−0.8	−1.0	−2.0	−3.0	−4.0
B	0.96	0.66	0	−0.53	−0.72	−1.15	−1.33	−1.4

- Number of data items: 17
- Number of data items in the main loop: 12
- Within 20 data items.

Preparation and operation

● Store the program written on the next page.

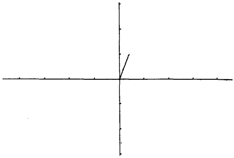
Memory contents								
	A	Number of data items	H		O		V	
	B	Number of data items in the main loop	I		P		W	
	C		J		Q		X	
	D		K		R		Y	
	E		L		S		Z	
	F		M		T			Z〔1〕~Z〔20〕 B
	G	F〔1〕~F〔20〕 H	N		U			

No.	8

Line	MODE 2 Program															Notes	Number of steps
1	Range	(−)	4	.	7	,	4	.	7	,	1	,	(−)	1			15
2	5	5	,	1	.	5	5	,	0	.	5	↵					27
3	Defm	2	0	↵													31
4	”	N	O	.	SPACE	O	F	SPACE	D	A	T	A	”	?	→		46
5	A	↵	Lbl	9	↵												51
6	”	M	A	I	N	SPACE	L	O	O	P	↵						62
7	N	O	.	SPACE	O	F	SPACE	D	A	T	A	”	?	→	B		77
8	↵																78
9	B	>	2	0	⇒	Goto	9	↵									86
10	1	→	C	:	Plot	0	,	0	↵								95
11	Lbl	0	:	”	H	=	”	?	→	F	〔	C	〕	↵			109
12	”	B	=	”	?	→	Z	〔	C	〕	↵						120
13	Plot	F	〔	C	〕	,	Z	〔	C	〕	:	Line	◢				133
14	C	+	1	→	C	↵											139
15	C	≠	A	+	1	⇒	Goto	0	↵								148
16	A	−	B	+	1	→	D	↵									156
17	Lbl	1	:	Plot	(−)	F	〔	D	〕	,	(−)	Z	〔	D	〕		171
18	:	Line	↵														174
19	D	+	1	→	D	↵											180
20	D	≠	A	+	1	⇒	Goto	1	↵								189
21	”	E	N	D	”												194
22																	
23																	
24																Memory 20×8=160	
25																	
26																Total 354 steps	
27																	
28																	
29																	
30																	
31																	
32																	
33																	
34																	
35																	
36																	

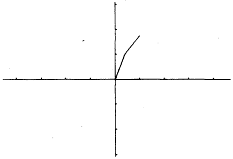
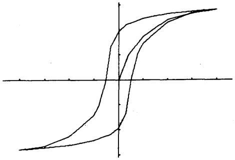
Program for **Hysteresis loop**

No. 8

Step	Key operation	Display
1	Prog 0 EXE	Prog 0 NO. OF DATA?
2	17 EXE	Prog 0 NO. OF DATA? 17 MAIN LOOP NO. OF DATA?
3	12 EXE	Prog 0 NO. OF DATA? 17 MAIN LOOP NO. OF DATA? 12 H=?
4	0.4 EXE 0.5 EXE	

Program for **Hysteresis loop**

No. 8

Step	Key operation	Display
5	EXE 1.0 EXE 0.86 EXE	
6	Input data in order. : : :	
7	EXE	−1.33 done H=? −4 B=? −1.4 done END
8	G↔T	

CASIO PROGRAM SHEET

Program for **Regression curve** No. 9

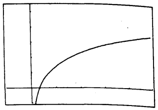
Description

i Logarithmic regression curve

Regression formula: $y=\mathrm{A}+\mathrm{B}\ln x$

$$\mathrm{B}=\frac{n\cdot\Sigma(y\cdot\ln x)-\Sigma\ln x\cdot\Sigma y}{n\,\Sigma(\ln x)^2-(\Sigma\ln x)^2}$$

$$\mathrm{A}=\frac{\Sigma y-\mathrm{B}\cdot\Sigma\ln x}{n}$$

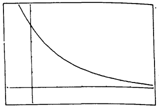


ii Exponential regression curve

Regression formula: $y=\mathrm{A}\cdot e^{\mathrm{B}x}$

$$\mathrm{B}=\frac{n\,\Sigma(x\ln y)-\Sigma x\cdot\Sigma\ln y}{n\cdot\Sigma x^2-(\Sigma x)^2}$$

$$\mathrm{A}=e^{\left(\frac{\Sigma\ln y-\mathrm{B}\cdot\Sigma x}{n}\right)}$$

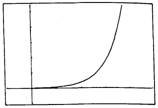


iii Power regression curve

Regression formula: $y=\mathrm{A}\cdot x^{\mathrm{B}}$

$$\mathrm{B}=\frac{n\,\Sigma(\ln x\cdot\ln y)-\Sigma\ln x\cdot\Sigma\ln y}{n\cdot\Sigma(\ln x)^2-(\Sigma\ln x)^2}$$

$$\mathrm{A}=\frac{\Sigma\ln y-\mathrm{B}\cdot\Sigma\ln x}{n}$$



- See page 144 for an example.

Preparation and operation

● Store the program written on the next page.

Memory contents	A	A or ln A	H	$\Sigma(\ln x)^2$	O		V	Σx
	B	B	I		P	Σy^2	W	n
	C	$\Sigma\ln x$	J		Q	Σy	X	x data
	D	$\Sigma\ln y$	K		R	Σxy	Y	y data
	E	$\mathrm{X}\Sigma\ln y$	L		S	For selection of 1~3	Z	
	F	$\mathrm{Y}\Sigma\ln x$	M		T			
	G	$\Sigma(\ln x\cdot\ln y)$	N		U	Σx^2		

No. 9

Line	MODE 2	Program														Notes	Number of steps
1	P0	SHIFT	MODE	÷	→	LR	2										
2	Scl	:	Cls	:	0	→	C	~	H	↵							10
3	"	Range	O	K	?	"	◢										17
4	"	D	A	T	A	SPACE	I	N	~	E	N	D	→	↵			31
5	A	C	→	Prog	1	SPACE	E	X	E	"	↵						42
6	Lbl	1	↵														45
7	"	X	:	"	?	→	X	↵									53
8	"	Y	:	"	?	→	Y	↵									61
9	ln	X	+	C	→	C	:	ln	Y	+	D	→	D	:	X		76
10	ln	Y	+	E	→	E	:	Y	ln	X	+	F	→	F	:		91
11	ln	X	×	ln	Y	+	G	→	G	:	(	ln	X	)	x^2		106
12	+	H	→	H	↵												111
13	X	,	Y	DT	◢												116
14	Goto	1															118
15																	
16	P1	MODE	+	→	COMP												
17	"	Y	=	A	+	B	ln	X	SPACE	→	1	↵					12
18	Y	=	A	×	e^x	(	B	X	)	SPACE	→	2	↵				25
19	Y	=	A	×	X	x^y	B	SPACE	SPACE	→	3	↵					37
20	1	~	3	:	"	?	→	S	↵								46
21	S	=	1	⇒	Prog	7	↵										53
22	S	=	2	⇒	Prog	8	↵										60
23	S	=	3	⇒	Prog	9	↵										67
24	"	E	N	D	"												72
25																	
26	P7	SHIFT	MODE	÷	→	LR	2										
27	(	W	F	−	C	Q	)	(	W	H	−	C	x^2	)	x^{-1}		15
28	→	B	:	(	Q	−	B	C	)	W	x^{-1}	→	A	↵			29
29	Graph	A	+	B	ln	X	◢										36
30	"	A	:	"	◢	A	◢										43
31	"	B	:	"	◢	B	◢										50
32																	
33																	
34																	
35																	
36																	

CASIO PROGRAM SHEET

Program for	No.
Regression curve	9

Example

Perform exponential regression of the following data:

x i	2.2	5.6	9.5	13.8	18.0	23.2	29.9	37.8
y i	35.6	28.1	23.0	17.9	12.9	10.2	6.2	4.0

Draw an exponential regression curve, and use the trace function to estimate the value for y when X = 20. Also, obtain the values of A and B of the regression formula.

Range values:

X min : −10 Y min : −10
X max : 50 Y max : 55
X scl : 10 Y scl : 10

Preparation and operation

- Store the program written on the next page.

Memory contents

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

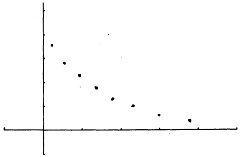
No.	9

Line	MODE 2 Program															Notes	Number of steps
1	P8	SHIFT	MODE	÷	→	LR	2										
2	(	W	E	—	V	D	)	(	W	U	—	V	x^2	)	x^{-1}		15
3	→	B	:	(	D	—	B	V	)	W	x^{-1}	→	A	↵			29
4	Graph	e^x	A	×	e^x	B	X	◢									37
5	"	A	:	"	◢	e^x	A	◢									45
6	"	B	:	"	◢	B	◢										52
7																	
8	P9	SHIFT	MODE	÷	→	LR	2										
9	(	W	G	—	C	D	)	(	W	H	—	C	x^2	)	x^{-1}		15
10	→	B	:	(	D	—	B	C	)	W	x^{-1}	→	A	↵			29
11	Graph	e^x	A	×	X	x^y	B	◢									37
12	"	A	:	"	◢	e^x	A	◢									45
13	"	B	:	"	◢	B	◢										52
14																	
15																Total 344 steps	
16																	
17																	
18																	
19																	
20																	
21																	
22																	
23																	
24																	
25																	
26																	
27																	
28																	
29																	
30																	
31																	
32																	
33																	
34																	
35																	
36																	

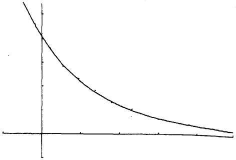
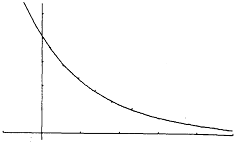
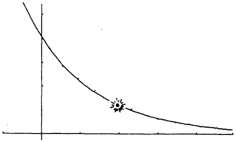
Program for **Regression curve** No. 9

Step	Key operation	Display
1	Prog 0 EXE (Range setting check)	Prog 0 Range OK? — Disp —
2	Set range values. Range (−) 10 EXE 50 EXE 10 EXE (−) 10 EXE 55 EXE 10	Range Xmin:−10 max:50 scl:10 Ymin:−10 max:55 scl:10_
3	EXE EXE After data input is complete, press the AC key and execute the program in Prog 1.	Prog 0 Range OK? DATA IN ~END→ AC→Prog 1 EXE X: ?
4	2.2 EXE 35.6 EXE	DATA IN ~END→ AC→Prog 1 EXE X: ? 2.2 Y: ? 35.6 2.2 — Disp —

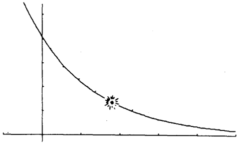
Program for **Regression curve** No. 9

Step	Key operation	Display
5	EXE	DATA IN ~END→ AC→Prog 1 EXE X: ? 2.2 Y: ? 35.6 2.2 X: ?
6	Input data in order. ⋮	
7	4.0 EXE	6.2 29.9 X: ? 37.8 Y: ? 4.0 37.8 — Disp —
8	G↔T	

Program for **Regression curve** No. 9

Step	Key operation	Display
9	AC Prog 1 EXE	Prog 1 Y=A+B ln X →1 Y=A×e(BX) →2 Y=A×X x^y B →3 1~3 : ?
10	2 EXE (Select exponential regression).	
11	SHIFT Trace	 X=-4.893617021
12	⇨ ~ Move pointer to X=20	 X=20.

Program for **Regression curve** No. 9

Step	Key operation	Display
13	SHIFT X↔Y	 Y=11.86149086
14	EXE EXE	Y=A×e(BX) →2 Y=A×X x^y B →3 1~3 : ? 2 done A : 40.68214077 — Disp —
15	EXE EXE	1~3 : ? 2 done A : 40.68214077 B : -0.06162460519 — Disp —
16	EXE	1~3 : ? 2 done A : 40.68214077 B : -0.06162460519 END

CASIO PROGRAM SHEET

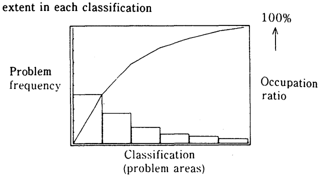
Program for	No.
Parade diagram	10

Description

One example of a parade diagram application is problem solving in QC activities. The problem is quantitatively analyzed based on actual data concerning its extent, and the main points demanding attention are determined.

Horizontal axis : Problem classification
(Item 6 in this example)

Vertical axis : (Right) Occupation ratio
(Left) Problem extent in each classification



Example

Create a parade diagram using the data on the right.

Problem areas	Frequency
A	105
B	65
C	35
D	20
E	15
Others	10

Preparation and operation

● Store the program written on the next page.

Memory contents								
	A	Input data	H		O		V	
	B		I		P		W	*n*
	C		J		Q		X	Count of data
	D		K		R		Y	
	E		L		S	Display count	Z	Sum of data
	F		M		T			Z〔1〕~Z〔6〕
	G		N		U			

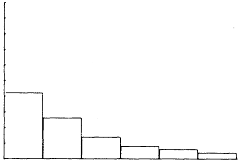
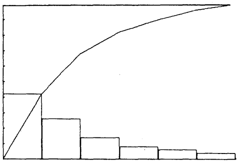
No. 10

Line	MODE 2 Program															Notes	Number of steps
1	P0	SHIFT	MODE	X	→	SD2											
2	Scl	:	Mcl	:	Defm	6	↵										7
3	Range	0	,	6	,	1	,	0	,	5	0	0		5	0		22
4	↵																23
5	Lbl	1	↵														26
6	"	D	A	T	A	"	?	→	A	↵							36
7	X	;	A	DT	↵												41
8	X	+	1	→	X	:	X	≤	5	⇒	Goto	1	↵				54
9	Range	,	,	,	,	W	,	W	÷	1	0	↵					66
10	Graph	◢															68
11	Plot	0	,	0	↵												73
12	1	→	S	↵													77
13	Lbl	2	↵														80
14	Z	〔	S	〕	+	Z	→	Z	↵								89
15	Plot	S	,	Z	:	Line	↵										96
16	S	+	1	→	S	:	S	≤	6	⇒	Goto	2	↵				109
17	Graph	W															111
18																	
19																Memory 6X8=48	
20																	
21																Total 159 steps	
22																	
23																	
24																	
25																	
26																	
27																	
28																	
29																	
30																	
31																	
32																	
33																	
34																	
35																	
36																	

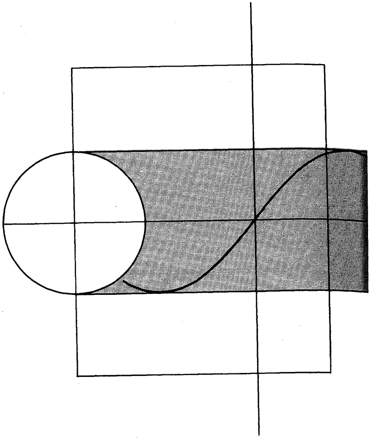
Program for		No.
Parade diagram		10

Step	Key operation	Display
1	Prog 0 EXE	Prog 0 DATA?
2	105 EXE	Prog 0 DATA? 105 DATA?
3	65 EXE	Prog 0 DATA? 105 DATA? 65 DATA?
4	Input data in order. ⋮	

Program for		No.
Parade diagram		10

Step	Key operation	Display
5	10 EXE (Bar graph display)	
6	EXE (Parade diagram display)	
7		
8		

REFERENCE MATERIAL



■ Error messages

Message	Meaning	Countermeasure
Syn ERROR	①Computation formula contains an error. ②Formula in a program contains an error.	①Use ⇦ or ⇨ to display the point where the error was generated and correct it. ②Use ⇦ or ⇨ to display the point where the error was generated, press AC and then correct the program in the WRT mode.
Ma ERROR	①Computation result exceeds computation range. ②Computation is performed outside the input range of a function. ③Illogical operation (division by zero, etc.)	①②③ Check the input numeric value and correct it. When using memories, check that the numeric values stored in memories are correct.
Go ERROR	①No corresponding Lbl *n* to Goto *n*. ②No program stored in program area P *n* which corresponds to Prog *n*.	①Correctly input a Lbl *n* to correspond to the Goto *n*, or delete the Goto *n* if not required. ②Store a program in program area P *n* to correspond to Prog *n*, or delete the Prog *n* if not required.
Ne ERROR	• Nesting of subroutines by Prog *n* exceeds 10 levels.	• Ensure that Prog *n* is not used to return from subroutines to main routine. If used, delete any unnecessary Prog *n*. • Trace the subroutine jump destinations and ensure that no jumps are made back to the original program area. Ensure that returns are made correctly.

Stk ERROR	• Execution of computations that exceed the capacity of the stack for numeric values or stack for computations.	• Simplify the formulas to keep stacks within 8 levels for the numeric values and 20 levels for the computations. • Divide the formula into two or more parts.
Mem ERROR	• Attempt to use a memory such as Z[5] when no memory has been expanded.	• Expand memories using MODE ⊡ (Defm). • Use memories within the current number of memories.
Arg ERROR	Incorrect argument specification for a command that requires an argument.	Correct the argument. • Sci n, Fix n: $n=$ natural number from 0 through 9. • Goto n, Lbl n, Prog n: n = natural number from 0 through 9. • Defm n: n = natural number between 0 to the number of remaining steps.

■ Input range of functions (general principles)

Function name	Input range
sin$x$, cos$x$, tan$x$	$\lvert x\rvert \leqq 9\times10^9$ degree $\lvert x\rvert \leqq 5\times10^7 \pi$ rad $\lvert x\rvert < 10^{10}$gra
sin^{-1}x, cos^{-1}x	$\lvert x\rvert \leqq 1$
tan^{-1}x	$\lvert x\rvert < 10^{100}$
e^x	$-10^{100} < x \leqq 230.2585092$
sinh$x$, cosh$x$	$\lvert x\rvert \leqq 230.2585092$
tanh$x$	$\lvert x\rvert < 10^{100}$
sinh^{-1}x	$\lvert x\rvert < 5\times10^{99}$
cosh^{-1}x	$1 \leqq x < 5\times10^{99}$
tanh^{-1}x	$\lvert x\rvert < 1$

log$x$, ln$x$	$0 < x < 10^{100}$
10^x	$-10^{100} < x < 100$
$\sqrt{x}$	$0 \leqq x < 10^{100}$
x^2	$\lvert x\rvert < 10^{50}$
$x^{-1}(^1/_x)$	$\lvert x\rvert < 10^{100}$, $x \neq 0$
$\sqrt[3]{x}$	$\lvert x\rvert < 10^{100}$
$x!$	$0 \leqq x \leqq 69$ (x is an integer.)
x^y	When $x<0$, y is a natural number. $x=0 \rightarrow y>0$
$\sqrt[y]{x}$ $(x^{1/y})$	$x \geqq 0$, $y \neq 0$
Pol (x, y)	$\lvert x\rvert < 10^{100}$, $\lvert y\rvert < 10^{100}$ However, $\sqrt{x^2+y^2} < 10^{100}$
Rec (r, θ)	$\lvert r\rvert < 10^{100}$, $\lvert\theta\rvert \leqq 9\times10^9$ degree $\lvert\theta\rvert \leqq 5\times10^7 \pi$ rad $\lvert\theta\rvert < 10^{10}$gra

Binary number	(Positive) $111111111111111 \geqq x \geqq 0$ (Negative) $1111111111111111 \geqq x \geqq 1000000000000000$
Octal number	(Positive) $17777777777 \geqq x \geqq 0$ (Negative) $37777777777 \geqq x \geqq 20000000000$
Hexadecimal number	(Positive) 7 FFFFFFF $\geqq x \geqq 0$ (Negative) FFFFFFFF $\geqq x \geqq 80000000$
Decimal→ sexagesimal	$\lvert x\rvert \leqq 9999999.999$. If degrees,minutes and seconds exceed a total of 11 digits, the higher (degrees, minutes) values will be given priority, and displayed in 11 digits.
Statistical computation	$\lvert x\rvert < 10^{50}$, $\lvert y\rvert < 10^{50}$, $\lvert n\rvert < 10^{100}$

* As a rule, the accuracy of a result is ±1 at the 10th digit.

* Errors may be cumulative with such internal continuous computations with the functions, x^y, $x^{1/y}$, $x!$, $\sqrt[3]{x}$, and accuracy is sometimes affected.

* In tan$x$, $\lvert x\rvert \neq 90^\circ\times(2n+1)$, $\lvert x\rvert \neq {}^{\pi}/_{2}\text{rad}\times(2n+1)$, $\lvert x\rvert \neq 100$ gra $(2n+1)$, (n is an integer.)

* With sinh x and tanh x, when $x=0$, errors are cumulative and accuracy is affected.

SPECIFICATIONS

Model: fx-7000GA

Computations

Basic computation functions:	Negative numbers, exponents, parenthetical addition/subtraction/multiplication/division(with priority sequence judgement function—true algebraic logic).
Built-in functions:	Trigonometric/inverse trigonometric functions (units of angular measurement: degrees, radians, grads), hyperbolic/inverse hyperbolic functions, logarithmic/exponential functions, reciprocals, factorials, square roots, cube roots, powers, roots, squares, decimal-sexagesimal conversions, binary-octal-hexadecimal conversions/computations, coordinate transformations, π, random numbers, absolute values, integers, fractions.
Statistical computation functions:	Standard deviation—number of data, sum, sum of squares, mean, standard deviation (two types). Linear regression—number of data, sum of x, sum of y, sum of squares of y, sum of squares of x, mean of x, mean of y, standard deviation of x (two types), standard deviation of y (two types), constant term, regression coefficient, correlation coefficient, estimated value of x, estimated value of y.
Memories:	26 standard (78 maximum)
Computation range:	$\pm 1 \times 10^{-99} \sim \pm 9.999999999 \times 10^{99}$ and 0. Internal operation uses 13-digit mantissa.
Rounding:	Performed according to the specified number of significant digits or the number of specified decimal places.

Programs

Number of steps:	422 maximum
Jump function:	Unconditional jump (Goto), 10 maximum Conditional jump ($=$, $\neq$, $>$, $<$, $\geqq$, $\leqq$) Count jumps (Isz, Dsz)
Subroutines:	9 levels
Number of stored programs:	10 maximum (P0 to P9)
Check function:	Program checking, debugging, deletion, addition, etc.

Graph function

Built-in function graphs:	(20 types) sin, cos, tan, $\sin^{-1}$, $\cos^{-1}$, $\tan^{-1}$, sinh, cosh, tanh, $\sinh^{-1}$, $\cosh^{-1}$, $\tanh^{-1}$, log, ln, 10^x, e^x, x^2, $\sqrt{\ }$, $\sqrt[3]{\ }$, x^{-1}
Graph commands:	Graph, Range, Plot, Trace, Factor, Line, X$\longleftrightarrow$Y, Instant factor
Graphs:	User generated functions, statistical graphs (bar graphs, line graphs, normal distribution curves, regression lines)

Common section

Power supply:	Three lithium batteries (CR2032)
Power consumption:	0.04 W
Battery life:	Approximately 100 hours on type CR2032.
Auto power off:	Power is automatically switched off approximately 6 minutes after last operation.
Ambient temperature range:	0°C—40°C(32°F—104°F)
Dimensions:	14mmH × 83.5mmW × 167mmD ($^{9}/_{16}$"H × $3^{1}/_{4}$"W × $6^{5}/_{8}$"D)
Weight:	152g (5.4oz) including batteries