SHARP

Pocket computer

PYTHAGORUS

SHARP ELSIMATE

model PC-1300

English translation of the original Sharp PC-1300 manual and PC-1300S differences manual.
Two extra programs added with line-by-line descriptions.

Geoffrey S. Quickfall MSc, Palynologist, ATPL
Vancouver, British Columbia, Canada
geoffqf@telus.net
April, 2024

PREAMBLE:

Thank you for purchasing the Sharp Pocket Computer PC-1300. The supplemental publication for the 1300S is on page 64.

Read this instruction manual carefully prior to using the computer. In addition, save this instruction manual together with the "warranty card" and "service and customer consultation desk location list"

Notes:

- **program code inputs required and given as an example will be highlighted in yellow. All characters in the highlight are required for the code to work correctly including commas where needed.**
    - **Ex: insert the following code DISP "ABC", HLT◣**
- **Some figures are copied and the text is then translated from the original Sharp PC-1300 Japanese manual.**
- **'Key' illustrations used in the text may differ from original manual and are noted here:**
    - **,** comma key required for program coding.
    - , comma key for normal text punctuation.
    - ▁ cursor.
    - **◣** equivalent to the 
    - **=>** equivalent to key 
    - **F** equivalent to key 
    - **E** equivalent to key 
    - **CA** equivalent to key 

- **Unique mini Fortran for this computer. Programming structure terms are explained:**
    - **LINE NUMBER, LINE, DIGIT, STEP, STATEMENT, SENTENCE**
        - **DIGITS: a single space designed to hold a character.**
        - **STEP: is a single character, or function consisting of more than one character.**
        - **LINE #: line number in PRO mode.**
        - **STATEMENT: single entity on display separated by a ,**
        - **SENTENCE: the complete line composing of steps and statements.**
        - **LINE: LINE number and sentence.**

DIGITS: **there are 31 digits.**
00; A + B => C , SIN C + SIN B => D , DSP " D = ", HLT

STEPS: **there are 21 step this sentence**
00; A + B => C , SIN C + SIN B => D , DSP " D = " , HLT

LINE #: **the line number is 00;**
00; A + B => C , SIN C + SIN B => D , DSP " D = ", HLT

STATEMENTS: **there are 4 statements.**
00; A + B => C , SIN C + SIN B => D , DSP " D = " , HLT

SENTENCE: **this is a sentence.**
00; A + B => C , SIN C + SIN B => D , DSP " D = ", HLT

LINE:
00; A + B => C , SIN C + SIN B => D , DSP " D = ", HLT

INDEX:

INDEX:

INDEX:

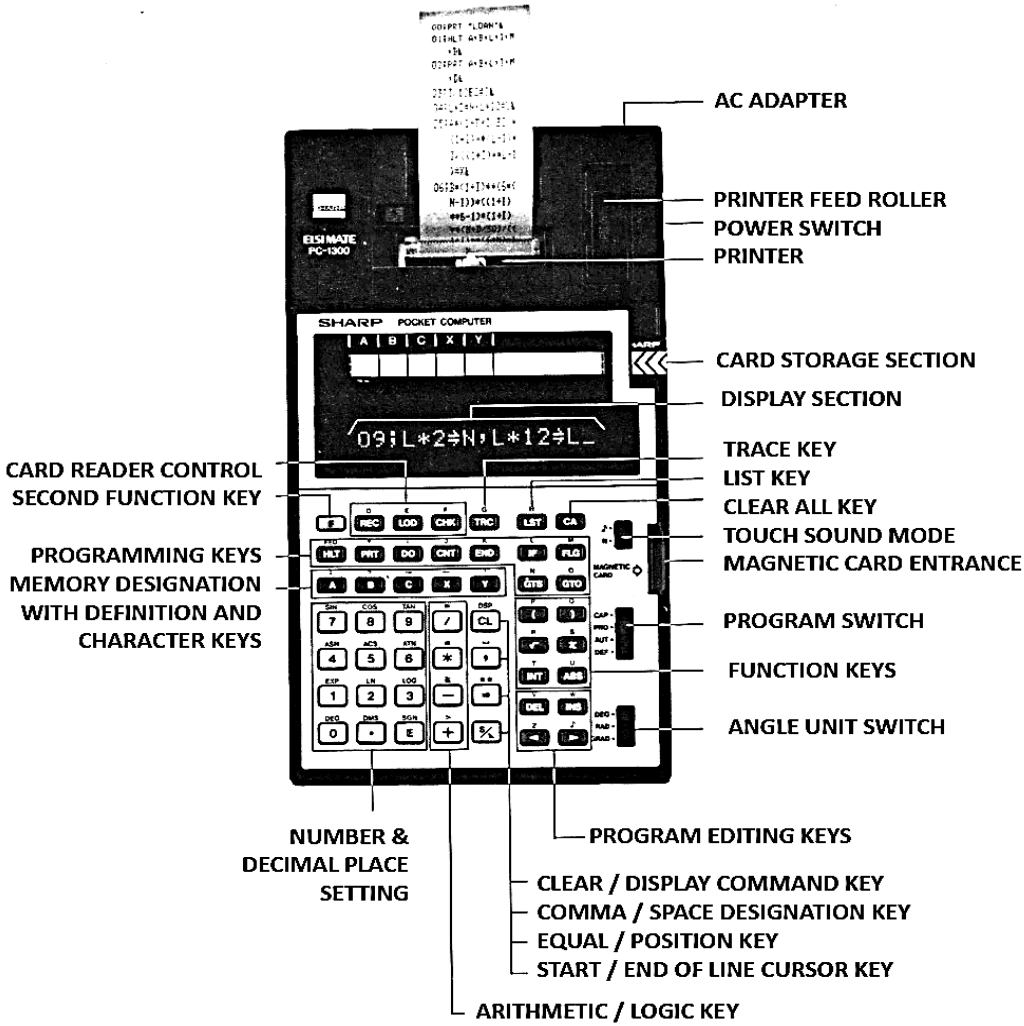
CAUTION:

This calculator consists of precision Toshiba parts, do not place it where it can be hot, in direct sunlight or near heating equipment.

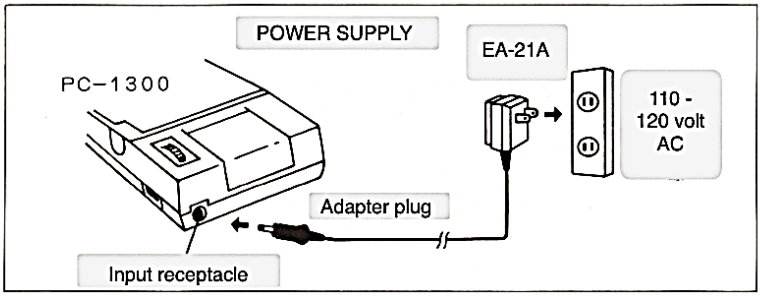
Avoid sudden temperature changes, low temperatures, humidity, dust, strong shocks, etc.

- **This calculator uses a spark printer, do not use it in places where there is a risk of ignition.**
- **Use a dry soft cloth, do not use progenitive liquids such as benzines and thinners, or wet cloths. Volatile liquids, wet wipes, etc. may damage the calculator**
- **Calculator generates RF and may interfere with Radio reception.**
- **If you want to paste and store the paper after printing, please use the following glue:**
    - **Stick glue.**
    - **If cellophane tapes or other adhesives are used, they may obscure the printing due to aging.**
- **Do not disassemble this calculator. In the unlikely event of a breakdown, bring the warranty card to our product dealer or Sharp System Service Co., Ltd. as described in the warranty card.**

KEYBOARD DESCRIPTION:



POWER SUPPLY:

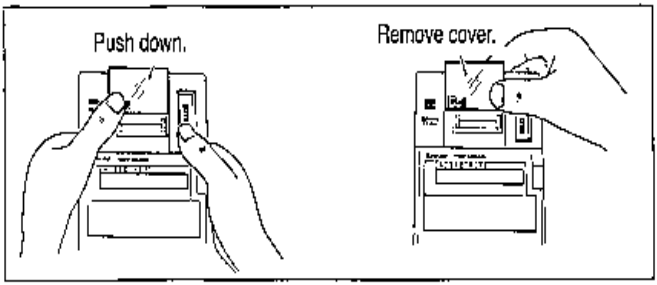


- **The portable internal power supply is a 4 cell NiCd battery pack. The original packs are now 43 years old and should be replaced. Good NiMH batteries are an excellent replacement but will take longer to charge using the supplied adapter.**

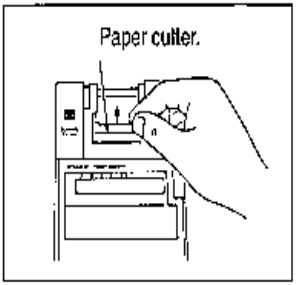
- **Pictured is the original EA-21A adapter. It may be substituted by another 8.2V DC 120 / 240 switching power supply *BUT*, the outer surface of the plug is POSITIVE and the internal connection is NEGATIVE. Therefore, be very careful when selecting a new adapter as the wiring for the plug may have to be reversed.**

PRINTER:

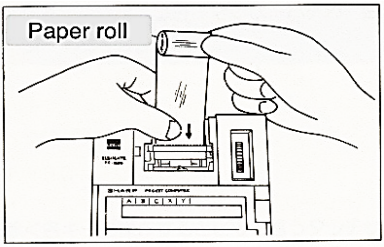
To remove paper cover, gently push down on bottom left corner while sliding upward.

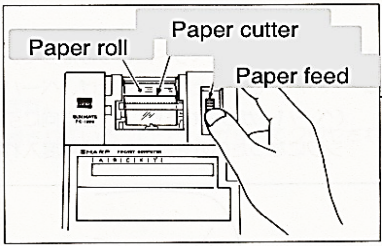


Printer paper access and installation.

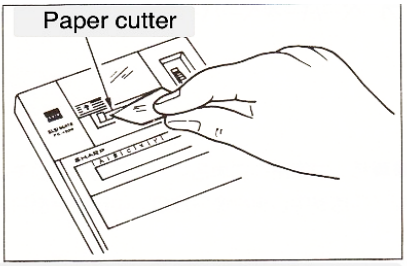


Insert paper roll and feed under paper cutter using paper feed wheel. To ease insertion of the paper, ensure the edge you are inserting is a straight cut.



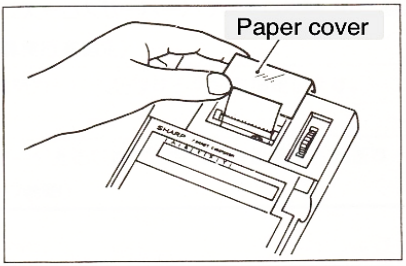


For paper jam, remove paper cover. Remove the roll prior to releasing the jam by tearing it at the base, before it feeds into the printer. Do not pull backwards to clear the jam. Grasp paper and gently untangle and pull forward until the remaining paper is clear of paper cutter.

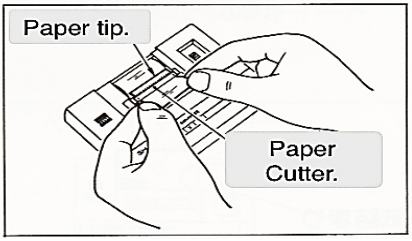


To clear paper jam, remove paper cover.

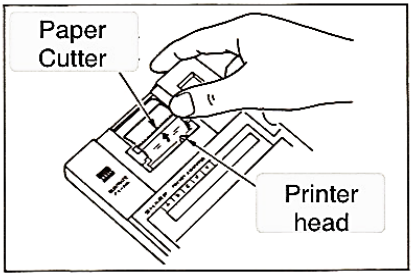
Remove excess paper.



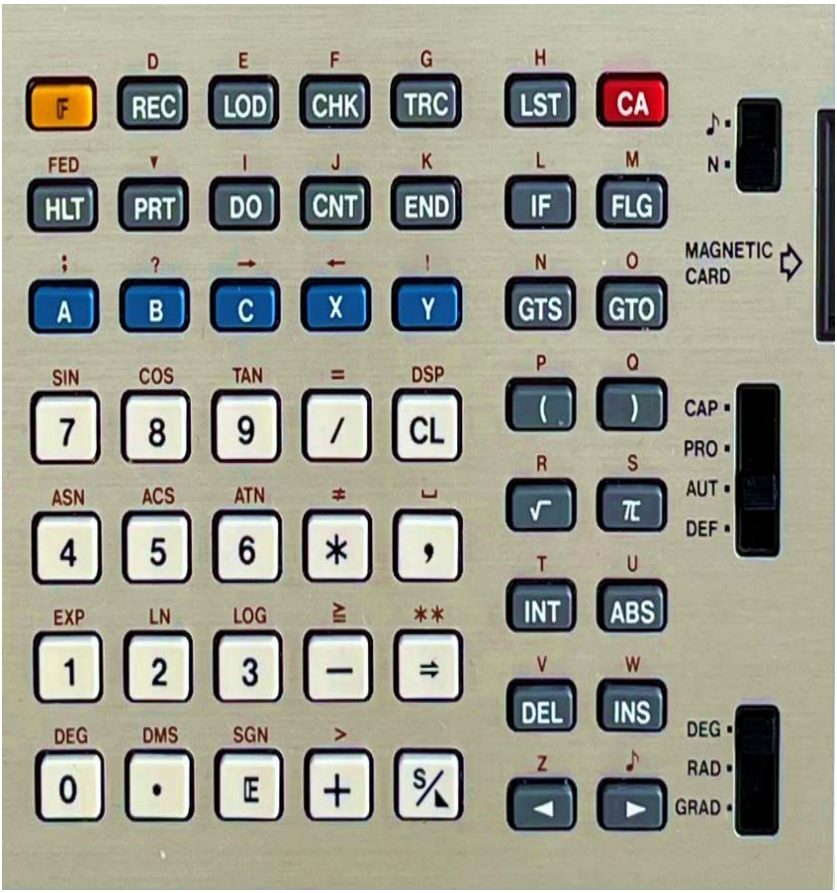
Extend paper tip to clear the paper cutter as illustrated



Correct position for paper and printer head in parked position.



KEYBOARD:


D E F G H
REC LOD CHK TRC LST CA
FED I J K L M
HLT PRT DO CNT END IF FLG
; ? → ← ! N O
A B C X Y GTS GTO
SIN COS TAN = DSP P Q
7 8 9 / CL ( )
ASN ACS ATN ≠ ⌴ R S
4 5 6 * , √ π
EXP LN LOG ≧ ** T U
1 2 3 − ⇒ INT ABS
DEG DMS SGN > V W
0 • E + % DEL INS
Z
◄ ►
N
MAGNETIC CARD
CAP
PRO
AUT
DEF
DEG
RAD
GRAD

KEYBOARD UNSHIFTED:

KEY	PAGE	KEY	PAGE
REC LOD CHK	8	0 → 9 & .	11
TRC	8	E	12
LST	8	/ * - +	12
CA	9	CL	12
HLT	9	,	13
PRT	9	=>	13
DO	9	◣	13
CNT	9	( )	15
END	10	√	15
IF	10	π	16
FLG	10	INT	16
A B C X Y	10	ABS	14
GTS	10	DEL	16
GTO	11	INS	16
FED	11	◄ ►	16

KEYBOARD SHIFTED

KEY	PAGE	KEY	PAGE
F	8	LN	14
A → Z	16	LOG	14
" ; ? ← → !	11	DEG DMS SGN	14
♪	Switch 16, 17 key	= ≠ ≥ >	15
SIN COS TAN	13	DSP	15
ASN ACS ATN	14	**	14
EXP	14		

OPERATION KEY	EXPLANATION OF OPERATION
OFF / ON	**POWER SWITCH**
 **Page 52**	**PROGRAM RAM SWITCH:** • **CAP: In conjunction with CA will clear all RAM (data memory is protected).** • **PRO: places calculator in program entry or edit mode.** • **AUT: calculation mode and program run mode.** • **DEF: specifies the execution mode of the defined program. This is akin to USER mode on HP's. When selected the A, B, C, X, Y keys will be assigned to the specific routine resident in the RAM program.**
	**DEGREE UNITS SWITCH:** • **DEG: degrees** • **RAD: radians** • **GRAD: grads (90° = π/2(rad) = 100°** **Controls the degree units. If a program is running, the switch will not select to the desired degree unit. The degree unit must be selected to match the program requirement prior to running the program.**
 **Page 17**	**AUT / PRO MODE:** • **∫ when selected, plays a chime for every key press.**

F (Shift key)	ALL MODES: • select secondary functions and alpha characters Ex: F SIN 30° => (the sin of 30° displays in the arithmetic line.
REC (record key) Page 57	• transfers program / data from computer to magnetic card. • Used with ,
LOD (load key) Page 58	• Transfers program / data from magnetic card to computer. • Used with ,
CHK (check key) Page 58	• Checks the data or program contents recorded on the magnetic card with the contents recorded on the calculator, and points to the line at which the error was detected. • Used with ,
TRC (print trace mode key)	AUT MODE: • Used to set up or cancel trace printing mode. • All keys, strokes and results are printed in sequential order.
LST (list program key) Page 66	• Prints program starting at current position. If complete program print desired, in AUT mode, press the CA key. This will set the program at line 00; and the LST key will then print the complete program. • You can list from a line to another line inclusive.

CA (clear all key) (clear program key) (line position 00; key)	**PRO MODE:** • Reset the program to line 00; **AUT MODE:** • display and clear the calculation process so far. • Reset the program to line 00; • Cancel the error message. **CAP MODE:** • Clear the contents of the program memory and resets program to line 00;
HLT (halt key) Page 39	**PRO MODE:** • Used in program mode to halt (stop) program at that point for data entry or to stop and display during a routine. to display a result found in memory "A": Ex: A=> HLT • Used for skip function (page 39). Ex: HLT , ,
PRT (print key) Page 53	**PRO MODE:** • Used to request a print display. Ex: B + C => A , PRT A
DO (do key) Page 50	**PRO MODE:** • Used in conjunction with IF and conditionals to control program and direct to subroutines.
CNT (count key) (loop control key) Page 50	**PRO MODE:** • In combination with the IF and DO commands will control the number of iterations in a loop.

END (end key) Page 42	PRO MODE: • Delineates the end of the program. No further instructions to follow. • Separates two different programs. • Ends a subroutine and directs the program back to the next line after the GTS command. • Two END commands are required to signify the end of the program and the end of the card REC LOD CHK operation.
IF (if key) Page 47	PRO MODE: • Followed by the conditional key. Ex: IF X >= 0 , GTO "SUB"
FLG (flag key) Page 49	PRO MODE: • Points to the order to set or reset a flag. Flag used as a conditional.
A **B** **C** **X** **Y** Page 52	PRO, AUT MODE: • Memory A, B, C, X, Y specification. • Indirect memory location when used with memory 'X'. If you operate the memory specified key following, 'X' it allows access to the second set of memories found on the PC-1300S. DEF MODE: • Function in a program as a defined by keys A, B, C, X, Y (akin to the LBL A through E on an HP 67). Pressing the associated key executes the labeled function key.
GTS Page 44	PRO MODE: • Go to subroutine and when encountering an END command, return to NEXT line after the GTS command.

GTO **Page 42**	**PRO MODE:** • **Unconditional jump instruction.** • **Can be a line number or subroutine. Does not return on END.** **AUT MODE:** • **Goes to the directed line number of program. Good for editing.**
FED **(feed key)** **Page 54**	**A** **LL MODES:** • **Designates the one-line paper sending order.**
; **?** **→** **←** **!** **(character keys)** **(page 54)**	**ALL MODES:** • **" " are used to display characters.** **ALL MODES:** • **Character keys for alpha.**
0 ~ 9 **(numeric key)**	**ALL MODES:** • **Numerical selection keys.** • **Can be part of alpha prompt** • **Can be used as a label** **Ex: DSP "POINT 1"** **Ex: "2", routine 2.**
• **(decimal or period)**	**Used as a character between " " and a decimal place in numeric calculations.**

E	**ALL MODES:** • **Specify the scientific notation display.** **Ex: 2.3 E 3 = 2.3X10^3**
/ **(divide key)** **Page 26, 52**	**ALL MODES:** • **Division order.**
***** **(multiplication key)** **Page 26, 52**	**ALL MODES:** • **Multiplication order.**
+ **-** **(add and subtract key)** **Page 25, 52**	**ALL MODES:** • **Add or subtract order.**
CL **(clear key)**	**PRO MODE:** • **Places cursor at first position on a sentence.** **AUT MODE:** • **Clears in the middle of series of key operations (clears the key if pressed in error)** • **If you operate the CL key in the middle of a series of key operations (with the key operation not completed), it will clear the display and the operation process so far.**

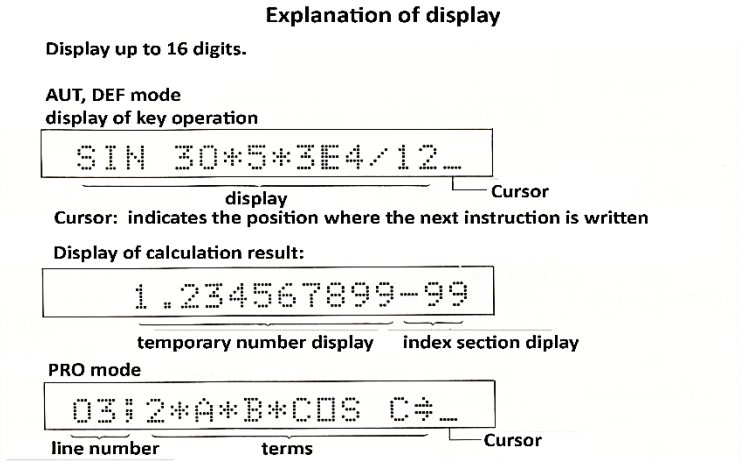
, (comma key)	**PRO MODE:** • Indicates a delineation between statements. • Points to the execution of the order. **AUT MODE:** • Causes the execution of the calculation. If you perform the calculation of <equation> with this key, the calculation is completed, and the result is next. • Similar to the ◣ key.
=> (equals key)	**PRO MODE:** • specify the calculation execution instruction of… Ex: A => the value in register A. **AUT MODE:** • Orders the execution of the manual calculation as an equals key • Ex: 2 + 3 => 5
◣ (start / line end command key)	**PRO MODE:** • Specify the line end. Causes next line in program to illuminate. **AUT MODE:** • Start the execution of the program. • Continue the execution of the program calculation that is suspended due to the HLT.
SIN **COS** **TAN** (transcendental keys) Page 28	• Transcendental functions. • Ex: SIN 30 => .5.

ASN **ACS** **ATN** (transcendental keys) Page 29	• Transcendental functions. • Ex: ASN .5 => 30
EXP (Eulers constant) **LN** (log base e) **LOG** (log base 10) Page 29	
****** (number to the power key) Page 30	• Can be used as 10^X o 10 ** 4 = 10000 • Can be used as Y^X or X^Y o 3.14 **5 = 305.2447762
DEG (decimal degrees) Page 30	Converts degrees.minutes-seconds to decimal degrees.
DMS (degree minutes seconds) Page 30	Converts decimal degrees to degrees.minutes-seconds.
SGN (code function) Page 31	Specify the arithmetic order for the number of sections. Code function: SGN X X > 0 +1 X = 0 0 X < 0 -1 A function that takes a value of positive = 1
ABS (absolute) Page 31	Takes the absolute value of a positive or negative number and returns the positive value of the number.

DSP (display key) Page 54	PRO MODE: • For alpha display prompt or label in program mode. Used in conjunction with the quotes " "
␣ (space key)	• Space key for alpha use as in " ABC␣DEF".
( ) (designation keys) Page 27	• Used to delineate or designate special functions outside of the math hierarchy. Ex: COS ((5+3) * (4 + 5)) => .8480480… • INT and some other 3 letter functions: Ex: INT (I) (see INT)
√ (root key) Page 30	AUT / DEF Ex: √ 25 => 5. Ex: √ (25 – 10) => 3.872983346 PRO / CAP Ex: √ A Ex: √ (A + B)
= ≠ ≥ > FLG Page 47, 49, 50	CONDTIONALS PRO / CAP • conditional tests for use with IF, DO and CNT.
π (constant Pi key)	• Executing inputs the symbol π. AUT mode: Ex: π => 3.141592653 Ex: π * 4 => 12.56637061 PRO mode: remains as π symbol. Ex: 00; SIN A * (π / 4) * R** =>

INT **(integer key)** **Page 31**	• **Pressing illuminates INT followed by the variable.** **Ex: INT 5.25678 => 5**
INS **(insert key)**	• **Used in conjunction with the cursor keys to insert a step between two steps.** • **May be used to insert multiple steps into a sentence**
DEL **(delete key)**	• **Used in conjunction with the cursor keys to remove an object between to other objects.**
◀ **(left cursor key)**	• **Moves cursor to the left and used to DEL or INS.**
▶ **(right cursor key)**	• **Moves cursor to the right and used to DEL or INS.**
A → Z **(memory keys)** **See page 64 for 1300S**	• **Memories A though Z and alpha letters.**
♪ **(sound key)**	**Generates an electronic sound when placed in a sentence.** • **To generate a BEEP: ♪♪♪♪ will generate 4 beeps in sequence. Used in a program as an audio alert** •
♪ **(sound key)**	**Specify the order of occurrence** o **The electronic sound generation instruction can be used freely in the <execution> or any place in the language instruction sentence, and electronic sounds can be generated. No matter what place this order is incorporated into, it won't have any impact on the other programs.** o

DISPLAY:



- **When you operate the key in AUT, DEF and PRO modes, the position (post number) where the next instruction to be operated is written, is displayed with the cursor ␣. If an order is written to the position step indicated by the cursor, the cursor ␣ will not be displayed, but the contents will be flashing. The cursor ␣ in conjunction with the ◄ ► cursor keys, can move freely within the step, statement or sentence.**
- **Use of SIN, COS and other three-digit steps will take up four digits including the blank. However, these four digits constitute only one step of the 640 available on the PC-1300S.**
    - **Ex: SIN_A represent two steps; one for SIN␣ and one for A. There are 5 digits displayed.**

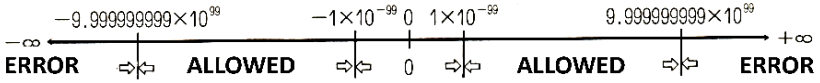
- **If you perform a key operation that requires more than 16 digits, the display will be as follows:**

Single operation	Display part	
12 + F SIN 30 * 42 + 123 - 57 * F LN F COS 30	12+SIN 30*42+ ⌟ 2+SIN 30*42+123 ⌟ +SIN 30 * 42+123 ⌟ 30*42+123-57 ⌟ 30*42+123-57* ⌟ 0*42+123-57*LN ⌟ +123-57*LN COS ⌟ 23-57*LN COS 30 ⌟ …	1 shift left. 2 shift left. + SIN shift left. 3 shift left. + 1 shift left.

The calculator can remember up to 80 steps of instructions (including numbers) by key operation in AUT mode in one sentence.

If the command becomes 80 steps, the far right of the display will flash and the display. At 80 steps, any new step will cause the first step to drop. This will affect the program so remain under 80 steps per sentence.

OPERATION RANGE OF FUNCTIONS:



FUNCTION	OPERATION RANGE	ERROR
SIN X COS X	ABS 1 < 1O^{10}	10 ± 1
TAN X	ABS 1 < 1O^{10} DEG: X ≠ 90 ± 180n RAD: X ≠ π/2 ± n π GRAD: X ≠ 100 ± 200n (N = 0, 1, 2…)	10 ± 1
ASN X ACS X	1 X 1 ≤ 1	10 ± 1
ATN X	TOTAL RANGE OF DEGREES	10 ± 1
LN X LOG X	X > 0	10 ± 1
√X	X ≥ 0	10 ± 1
EXP X	X ≤ 230.2585092	10 ± 1
X**Y (X^{Y})	X ≥ 0 Y . LN X < 230.2585092	10 ± 1

- **Input method:**
  - **Key input.**
  - **Magnetic card load and record.**
    - **One side = 128 steps or 16 data memories.**
- **Output method and display:**
  - **5 X 7 dot matrix VFD light display tube (16 digits).**
  - **Print: discharge (spark) printer on metalized paper.**

- **Calculation method:**
  - **According to standard hierarchy.**
  - **16 function buffers, 9 data buffers.**

FIXED AND FLOATING DISPLAY:

When displaying numerical values, there are cases where they are displayed in fixed-point or in floating-point method (exponential method).

In principle, the numerical values in the next range are displayed in fixed point method, and the numerical values that are out of that range are displayed in the floating-point direction.

Fixed point method:

Ex: $1x10^{-9} \leq x \leq 9999999999$ $-99999999999 \leq x \leq 1x10^{-9}$

Floating point method:

Ex: **0.00012345678** → **1.2345678 X $1x10^{-4}$**

- **For numerical values within the above range that cannot be displayed in the fixed-point method, it will be automatically switched to the floating-point method.**

DATA INPUT METHOD:

Insert the first part of the number, then use E to insert the exponential part:

Ex:	**6.7×10^{8}**	**→**	**6.7 E 8**
	-9.12×10^{-34}	**→**	**-9.12 E -34**

The exponential portion of the number is expecting two digits between 0 and 99. Inputting more digits in the exponential results in truncation of the excess digits.

The decimal portion is expecting ten digits. If greater than 10 digits are input, prior to the exponential, it will result in the truncation of the decimal portion to 10 digits.

Automatic correction for exponent occurs when decimal digits are included prior to the use of the exponent key E .

Ex:	**1234567898765**	**→**	**$1.234567898 \times 10^{12}$**
	9.87654321234	**→**	**9.876543212**
	0.0000000000234567	**→**	**2.34567×10^{-10}**
	0.00001234567 E 24	**→**	**1.234567×10^{19}**

The last two digits of the exponent part will always be correct.

Ex:	**3 E 123**	**→**	**3×10^{23}**
	4 E -3210	**→**	**4×10^{-10}**

COMPOSITION STYLE:

The calculation formula consists of the following instructions.

- **Symbol** **+, -**
- **Four rules** **+, -, *, /**
- **Comma** **,**
- **Function** **SIN, COS, TAN, ASN, ACS, ATN, LN, LOG, EXP, DEG, DMS, INT, ABS, √, **, SGN**
- **Fixed number** **0 ~ 9, (.), π, E**
- **Memory** **A ~ Z**
- **Other** **=, ≠, >, ≥**

To complete these instructions according to the calculation formula, use the => to solve the instructions.

ASSEMBLY STYLE:

1. **Firstly, CL or , to clear or complete the previous formula.**
2. **Operate keys such as constants, four rules, functions, etc. according to the calculation and assembled content. After assembling, => will start the calculation and display the result. Generally, the calculation => in AUT does not necessarily take this same form as a program instruction.**
   - **The content can be arbitrarily corrected using the cursor keys as long as the formula has not been executed.**
   - **While entering the key, the alpha-numeric symbol of the entered key is displayed and the cursor ␣ shows the step where the next key operation is entered. If information (instructions, etc.) have already been entered in the step, the cursor will flash at the content and not display as a cursor.**

SYNTAX:

ENTERING A CHARACTER:

Characters are used for prompting, memory, labelling, etc. The character must fall within the quotes; " " .

Ex: "ABC"
"SIN A => "
"PROGRAM A"

The following cannot be defined as a character:

Ex: F , TRC, " , ◣ , CL, LST, CA, INS, DEL, 

When entering each instruction as a character, firstly F then " to specify the character. Next enter the character by each key, and finally F " are used again, to stop the characters to be displayed in a prompt or label.

Ex:	**F " A B C F "**	**displays as**	**"A B C"**
	F " F SIN A = F "	**displays as**	**"SIN A ="**
	F " A B C F " ,	**displays as**	**ABC**
	F " F SIN A = F " ,	**displays as**	**"SIN A ="**

- **Using CA or CL and the input will be deleted.**

Only 16 characters can be entered between the " ". Any character after the 16th will be ignored.

F LN , F SIN and other 2 and 3 character keys are used in a display (between " ") and they exceed the 16 character, limit they

will not be displayed, only the order from the first instruction will be displayed. When using DSP command, the same following blank applies to the display.

- **You cannot treat a character as data or enter it into memory. Again, characters cannot be mixed, or characters cannot be mixed or calculated.**

THESE WILL GENERATE AN ERROR:

- **“ SIN 30 ” => A**
- **“ A ” + “ B “ ,**
- **“ ABC “ “ XYZ “ ,**

However, “ “ ◣ is allowed as it uses the ◣to separate the statement from other statements. Both the ◣ and , are used to separate statements.

- **GTO F “ A F “ ◣**

MANUAL OPERATIONS:

AUT MODE:

- **In the case of manual operation both => and , will perform as an operation key. Once the => or , are executed the answer appears and the next calculation may be started.**
- **Both the => and , are equivalent operations.**

1) THE FOUR CALCULATIONS:

+ - * /

① **Addition**

5+3 ⇒

−7+4+12 ⇒

AUT mode

Operation	Display	
CL 5 + 3	5+3_	expression
⇒ •	8.	answer
− 7 + 4 + 12	−7+4+12_	expression
⇒ •	9.	answer

② **Subtraction**

72−46 ⇒

−7.4E4−8.7E3⇒ $(-7.4\times10^{4}-8.7\times10^{3}=)$

AUT mode

CL 72 − 46	72−46_	expression
⇒ •	26.	answer
− 7.4 E 4 − 8.7 E 3	−7. 4E4−8. 7E3_	expression
⇒ •	− 82700.	answer

③ **Multiplication**

12✱24 ⇒ (12×24=)

56 E 7✱78 E 8 ⇒ ($56\times10^{7}\times78\times10^{8}$=)

AUT mode

Operation	Display	
CL 12 ✱ 24	12✱24_	instruction
⇒	288.	answer
56 E 7 ✱ 78 E 8	56E7✱78E8_	intruction
⇒	4. 368 18	answer

注) 5×(6+7) を5(6+7)

if you enter 5x(6+7) that omits the 'x' as in 5(6+7), you will get error code 01.

④ **Division**

24／8 ⇒ (24÷8=)

−72／6 E −2／32 ⇒ ($-72\div6\times10^{-2}\div32$=)

AUT mode

Operation	Display	
CL 24 / 8	24／8_	instruction
⇒	3.	answer
−72 / 6 E − 2 / 32	−72／6E−2／32_	instruction
⇒	− 37. 5	answer

⑤ **Mixed calculations**

54+24.3✱7.8／3.4−37.4 ⇒

AUT mode

Operation	Display	
CL 54 + 24.3 ✱ 7.8	54+24. 3✱7. 8_	instruction
/ 3.4 − 37.4	. 3✱7. 8／3. 4−37. 4_	
⇒	72. 3470588	answer

calculation follows accepted hierarchy standards.

2) PARENTHESES: *(the PC-1300S has 52 memories, page 64)*

you can manipulate the hierarchy of mathematical functions by using parentheses.

8+6∗ (5+7)⇒

6∗ {4+6∗(9−4)}⇒

AUT mode

Operation	Display	
CL 8 + 6 ∗	8+6∗_	input
(5 + 7)	8+6∗(5+7) _	
⇒ =	80.	answer
6 ∗ (4 + 6 ∗	6∗(4+6∗_	input
(9 − 4))	6∗(4+6∗(9−4)) _	
⇒	204.	answer

A+B/C →$A+\frac{B}{C}$

(A+B)/C →$\frac{A+B}{C}$

A/C∗D →$\frac{AD}{C}$

A/(C∗D) →$\frac{A}{CD}$

A/B/C →$\frac{\frac{A}{B}}{C}=\frac{A}{BC}$

A/(B/C) →$\frac{\frac{A}{B}}{C}=\frac{AC}{B}$

√A+B →$\sqrt{A}+B$

√(A+B) →$\sqrt{A+B}$

√A∗B →$B\sqrt{A}$

√(A∗B) →$\sqrt{AB}$

A∗B+C →AB+C

A∗(B+C)→A(B+C)

3 Memory

There are 26 memories: 0 to 25, each is named A, B, C,... Z.
To store numerical values:

3 ⇒ A =

5 ⇒ F D =

To recall contents from memory:

X =

F Z =

A + 5 ⇒ B =

A ∗ B + C ⇒ F D =

A ⇒ B ⇒ C =

multiply A by B then add C and store in D.

3) TRIGONOMETRIC FUNCTIONS: *(transcendental functions)*

SIN COS TAN

① **Trig functions**

SIN 30⇒ (sin 30=)

COS(π/4)⇒ ($\cos\frac{\pi}{4}$ =)

TAN 150⇒ (tan 150=)

AUT mode

Operation	Display	
DEG CL F SIN 30	SIN 30_	
⇒	0. 5	SIN 30°
RAD F COS (π / 4)	COS (π/4)_	
⇒	7. 071067812-01	COS π/4(rad)
GRND F TAN 150	TAN 150_	
⇒	− 1.	TAN 150°

• COS(π/4) **fractional calculation must use parenthese**

3) TRIGONOMETRIC FUNCTIONS: (transcendental functions): ASN ACS ATN LOG EXP

② Trigometric functions

例 ASN (−0.5) ⇒ ($\sin^{-1}$ (−0.5) =)
ACS (−0.5+0.1) ⇒ ($\cos^{-1}$ (−0.5+0.1) =)

AUT mode

Operation	Display	
DEG [CL] [2nd] [ASN] [(] [−] .5 [)] [=] [▼]	ASN (−. 5)_	
	− 30.	answer
RAD [2nd] [ACS] [(] [−] .5	ACS (−. 5_	
[+] .1 [)] [=] [▼]	ACS (−. 5+. 1)_	
	1. 982313173	answer

•

③ Logarithmic function

例 LN 7.4 ⇒ (ln 7.4=)
LOG 100 ⇒ (log 100=)

Operation	Display	
[CL] [2nd] [LN] 7.4 [=] [▼]	LN 7. 4_	
	2. 00148	answer
[2nd] [LOG] 100 [=] [▼]	LOG 100_	
	2.	answer

④ Exponential function

例 EXP (−13.6) ⇒ ($e^{-13.6}$=)

Operation	Display	
[CL] [2nd] [EXP] [(] [−] 13.6 [)] [=] [▼]	EXP (−13. 6) _	
	1. 24049508−06	answer

3) TRIGONOMETRIC FUNCTIONS:

√ ** DEG DMS

⑤ **Square root**

√73 ⇒ (√73 =)

√(3∗3+4∗4) ⇒ ($\sqrt{3^3+4^3}$ =)

AUT mode

Operation	Display	
CL √ 73	√73_	
= ▼	8. 544003745	answer
√ (3 ∗ 3 +	√(3∗3+_	
4 ∗ 4)	√(3∗3+4∗4) _	
= ▼	5.	answer

⑥ **Power function**

5∗∗3⇒ (5^3 =)

(5+7)∗∗7.8 ⇒ ($(5+7)^{7.8}$ =)

AUT mode

Operation	Display		
CL 5 ∗∗ 3	5∗∗3_		
= ▼	125.	5^3	same as
(5 + 7) ∗∗ 7.8	(5+7)∗∗7.8_		
= ▼	261585531.3	$(5+7)^{7.8}$	same as

⑦

DEG : **dd.mmss: converted to decimal degrees**

DMS : **dd.xxxx: converted to dd.mmss...**

OO.OOOOOO

dd mm ss end

15°24'45" **convert to decimal degrees**

15.4125° **convert to dd.mm.ss...**

Operation	Display	
CL DEG 15.2445	DEG 15.2445_	
= ▼	15.4125	15.4125°
DMS 15.4125	DMS 15.4125_	
= ▼	15.2445	15°24'45"

4) OTHER:
INT ABS SGN CONDITIONALS

⑧ Integer only

INT (25／0.3) ⇒

Operation	Display	
CL INT (25 ／ .3) ⇒ ▼	INT (25／. 3)_ 83.	

⑨ Absolute value

ABS (5−9) ⇒ (|5−9|)

Operation	Display	
CL ABS (5 − 9) ⇒ ▼	ABS (5−9)_ 4.	

⑩ Code function

functions take the following values: positive 1
negative -1
zero 0

SGN (5−9) ⇒

Operation	Display	
CL F SGN (5 − 9) ⇒ ▼	SGN (5−9)_ − 1.	

⑪ Logical functions (conditional tests)

>、≥、=、≠ compare two values; if true do next command

$(7.5 \geq \frac{a}{b}) * \frac{1}{(a+b)*c}$ $\left(7.5 \geq \frac{a}{b} \quad \frac{1}{(a+b)*c}\right)$

a : 15、b : 2.1 、c : 4×10^{-2}

Operation	Display	
CL 15 ⇒ A ▼	15.	a→A
2.1 ⇒ B ▼	2. 1	b→B
4 F − 2 = C ▼	0.04	c→C
(7.5 F ≥ A ／	(7. 5≥A／_	
B) * (1 ／	(7. 5≥A／B)*(1／_	
((A + B) *	A／B)*(1／((A+B)*_	
C)) ⇒ ▼	1. 461988304	

5) ARITHMETIC HIERARCHY (PEMDAS):

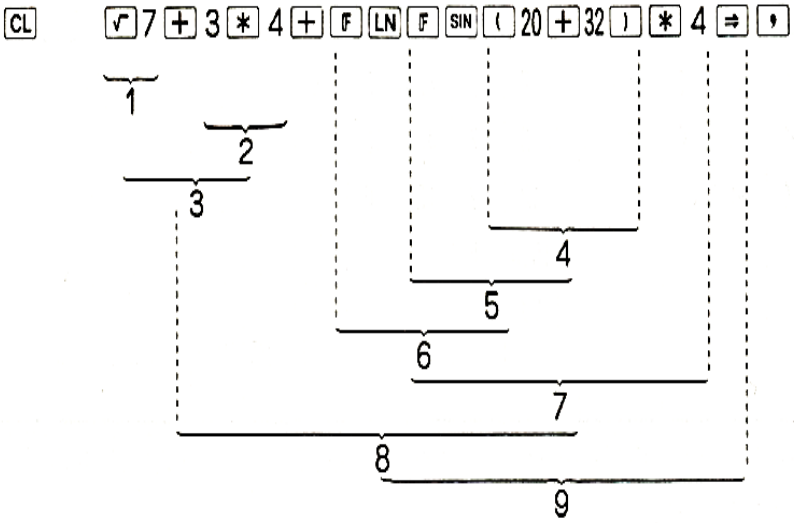
1. **PARENTHESES**
2. **EXPONENTS (working from right to left for stacked exponents)**
3. **MULTIPLICATION / DIVISION**
4. **ADDITION / SUBTRACTION**

- the calculation is read from right to left: power 4, power 3 on integer 2

例 2**3**4⇒ 2^{3^4}

(2**3)**4⇒ $(2^3)^4$

Example of arithmetic order



6) BUFFER:

6. Buffer

This computer uses PEMDAS hierarchy. Numeric values that are not processed immediately are temporarily stored in a buffer. There are 16 function and 9 numeric buffers.

3*2* (12+A/ (7+5))⇒
A : 42

function buffer								numerical buffer						
	1	2	3	4	5	6	...		1	2	3	4	5	...
3								3						
*	*							3	3					
2	*							2	3					
*	*							6	6					
(	(	*						6	6					
12	(	*						12	6					
+	+	(	*					12	12	6				
A	MA	+	(	*				12	12	6				
/	/	+	(	*				42	42	12	6			
(	(	/	+	(	*			42	42	12	6			
7	(	/	+	(	*			7	42	12	6			
+	+	(	/	+	(	*		7	7	42	12	6		
5	+	(	/	+	(	*		5	7	42	12	6		
)	/	+	(	*				12	42	12	6			
)	*							15.5	6					
⇒								93						
,														

X (number)

use memory buffer after X is displayed

As in the above, parentheses are also remembered in the function buffer. The parentheses can have depth within the range that can be remembered in this buffer.

7. Correct INPUT of formula.

SQRT (4^2 + 3^)=>

CL ↶ (4 * 4 + 3 * 3) ⇒ ,

①

7) CORRECTING INPUTS: CL CURSOR DEL INS

AUT mode

Operation	Display	
CL √ (4 × 4	√ (4×4_	correct input
−	√ (4×4−_	
→	√ (4×4−_	
+	√ (4×4+_	
3 × 3)	√ (4×4+3×3)_	
⇒ =	5.	

② **Inserting missing component**

Operation	Display	
CL √ (4 × 4	√ (4×4_	back spacing using cursor then INS to insert a space for the '+" key.
3 × 3)	√ (4×43×3)_	
← ← ← ←	√ (4×43×3)	
INS	√ (4×4_3×3)	
+	√ (4×4+3×3)	
← ← ← ←	√ (4×4+3×3)_	
⇒ =	5.	

- **examples 2 and 3 illustrate editing capabilities using the cursor keys and functions INS and DEL**

- **in some cases, for multiple insertions repeat the INS command as many times as required.**

③ **Removing input error and replacing with correct input.**

Operation	Display	
CL √ (4 × 4	√ (4×4_	
+ 3 × 3	√ (4×4+3×3_	
× 3)	√ (4×4+3×3×3)_	
← ← ←	√ (4×4+3×3×3)	
DEL	√ (4×4+3×33)	
DEL	√ (4×4+3×3)	
←	√ (4×4+3×3)_	
⇒ =	5.	

- **Here we see two errors removed by the DEL key with no insertion required.**

PROGRAMMING:

BEFORE PROGRAMMING:

PROGRAM METHOD *(see preamble)*

A LINE #:

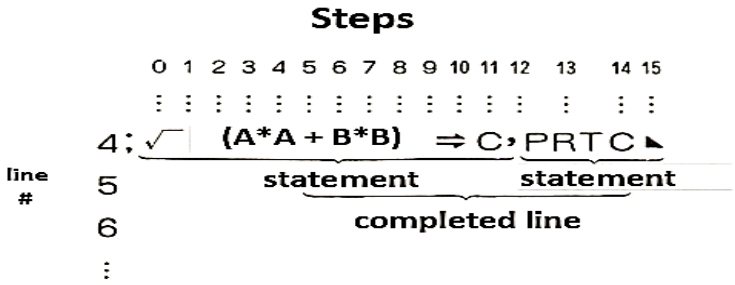
The code is entered in a side by side sentence format. The horizontal row of the program is called a line. The line number 00; → 99; is attached from 00 to the line on the left. Also the end of a line is the ◣

A STATEMENT:

One line consists of operation instruction(s) called a statement(s). A comma , is used to deliniate statements on the same line.

A SENTENCE:

Refers to a completed line up to and including the ◣



PROCEDURE:

1. **In AUT or PRO the CA key results in the RAM being positioned at line 00; regardless of any program still resident in the program RAM.**
2. **To erase all programs in the Program RAM place the calculator in CAP mode, press CA to erase all program RAM and position the cursor at line 00; Memory contents are not erased. Place the switch back to PRO mode for entering a program. *(ensure switch is in PRO and not CAP mode while programming to eliminate any accidental code erasure.)***
3. **in PRO mode, at line 00; enter the code in correct sequence.**
4. **At the end of each sentence, press ◣ signifying the end of the sentence.**
5. **The next line appears after the ◣ key is pressed.**
6. **Complete inputting the program.**
7. **When the program is loaded, finish with END END ◣**

Side note:

Sharp named this hand held computer the 'Pythagoras'. Fittingly, the first program example in the manual (follows) is the Pythagoras theorem. Therefore you are loading Pythagoras with Pythagoras!

1. PYTHAGORAS theorem: Simple program calling for variables A and B and solving for C.

formula $c = \sqrt{a^2 + b^2}$

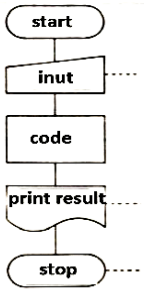
PRO mode

a = 3 b = 4

Operation	
3 [⇒] [A] [•] 4 [⇒] [B] [•]	3→A、4→B
[√] [(] [A] [*] [A] [+] [B] [*] [B]	$\sqrt{(A^2+B^2)}$
[)] [⇒] [•]	

line #	Code display
0	HLT A, B◣
1	√(A*A+B*B)⇒C◣
2	PRT C◣
3	END END◣

simple program without conditionals, loops or sub-routines.



halt program for data input

input A, input B

run routine

print result

two ends: one ends program. second ends card read.

29

- **There will be two END commands to terminate the program. This is useful if a second program independent of the first is loaded or when listing RAM (PRO mode) contents.**
- **Ensure you are not it CAP mode. Check your input in PRO mode by pressing CA key. This will position the cursor on line 00; Use the cursor to advance through each statement confirming its correctness. Correct with DEL and INS keys to delete or insert a blank for use for another statement.**
- **In AUT mode you can use the GTO key followed by a line number. Switching to PRO mode will now be positioned at the requested line number.**

TO EXECUTE A PROGRAM

1. **In AUT press CA**
2. **Press R/S to start execution of program.**
    a. **The execution stops at a HLT (halt) command.**
    b. **This is the opportunity to enter a variable (data).**
    c. **Press R/S to continue program.**
3. **The program is terminated by the END command.**

Example of GTO command in AUT mode to direct the desired line number to appear in PRO mode.

Operation	Display	
AUT GTO 1 . PRO	1 . 01: (A*A+B*B)⇒ C cursor position (original display)	1 line designation 1 display of the contents of the line

If an error is found in the written program it can be corrected in the following manner:

1. **In AUT mode specify the line with the error and use GTO xx (where xx is the line number).**
2. **Switch to PRO mode and operate the cursor to position at the erroneous statement.**
    a. **Use INS to add blank spaces required to enter spaces to then add the missing command.**
    b. **Use DEL to remove erroneous incorrect statement.**
    c. **Use both DEL to remove and INS to insert spaces to replace an incorrect statement.**

Operation	Display	use of prompt
CA	0.	
R/S	␣ 0.	prompt
3	␣ 3_	
R/S	␣ 3.	prompt
4	␣ 4_	
R/S	5.	
R/S	␣ 5.	prompt
36	␣ 36_	
R/S	␣ 36.	prompt
27	␣ 27_	
R/S	45.	

During program execution , the BUSY symbol flashes.

1) HLT: (HALT)

- **The HLT command is used in a program to stop execution and allow variables (data) to be entered and stored in memory.**

- **It can also be used to stop a program to check interim results.**
- **When execution of the program halts the ␣ is displayed on the left side.**
- **Once the execution of the program is stopped, manual calculations can be performed. In addition, the result can be stored as dta in a specified memory.**

The following manual calculation can be performed when the program execution of the Pythagoras theorem is stopped.

A = 72 * COS 50

B = 97 * SIN 32

AUT mode

Operation	Display	
DEG [CA]	0.	
[S/A]	␣ 0.	
72 [×] [F] [COS] 60 [⇒]	␣ 36.	A data
[S/A]	␣ 36.	
97 [×] [F] [SIN] 32 [⇒]	␣ 51. 40216862	B data
[S/A]	62. 75494353	

The program will continue with the => regardless of data input. There is no skip function in the program with the HLT and =>.

The program will continue with a number or data calcualtion, even if you complete the operation with CL or => or if you enter a character with " ", it will continue or skip the operation with the ◣ operation.

An example of a program with a skip action:

A program to find the average value.

Step #	Program
0	0⇒A⇒B◣
1	B, HLT X, , X+A⇒A, 1+B⇒B, GTO 1◣
2	PRT A, B, A/B◣
3	END END◣

Operation	Display	Print out and result
CA % 527 % 279 % 430 % 91 % %	0. 1. 2. 3. 4. 331.75	1327. 4. 331.75

As shown in the above example, if you want to write an expression in the statement immediately after the HLT command, use two commas after the HLT command.

The double comma explanation is needed if a formula follows the HLT command as in this example. If the HLT command is followed by a statement such as PRT or DSP then only one comma is required.

2) END (END).

- **The END command and double END command have a few uses. If used in conjunction with the GTS (go to subroutine and return):**
    - **Upon reading the END found at the end of the subroutine it will return to the main program and the next line after the GTS command.**
    - **It acts as a list termination instruction. You can then list individual programs in RAM without printing all the programs in the RAM and separate the programs for easy listing.**
    - **A termination for a REC command**
- **A termination for a LOD command**
- **A termination for a CHK command.**

3) GTO (GO TO).

ABSOLUTE JUMP **Ex: GTO 12**

A program is usually executed in order according to the line number, but if you want to skip to any line, GTO followed by the line number:

Ex: 12; 1 + B => C …
23; GTO 12

You can specify and expression after the GTO instead of a line number. The expression cannot exceed 99.

Ex: PRO mode GTO 12 * 36 ◣ (jump to line 36)

Ex: AUT mode GTO " 18 " ◣ (jump to line 18)

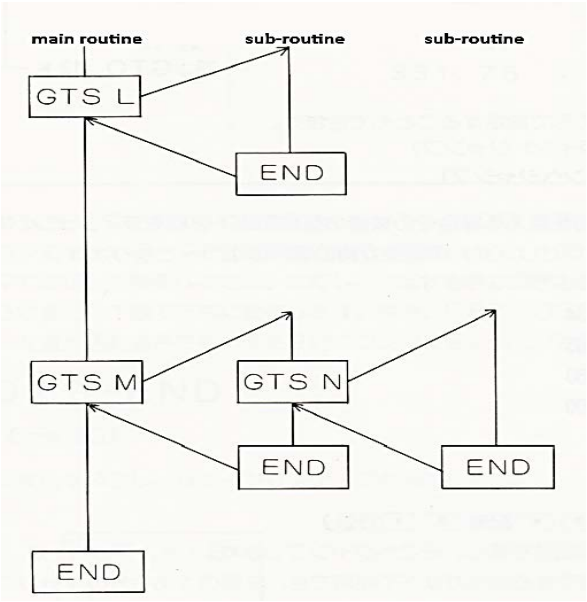
GTO " 18 " directs the calculator to jump to a line starting with the " 18 " label. Symbols may also be used such as " ABC " or " LON ".

The computer always jumps to 00; to find the label used in the GTO command. Thus if two labels have the same name the one with the lesser line number will be jumped too.

4) GTS (GO TO SUBROUTINE).

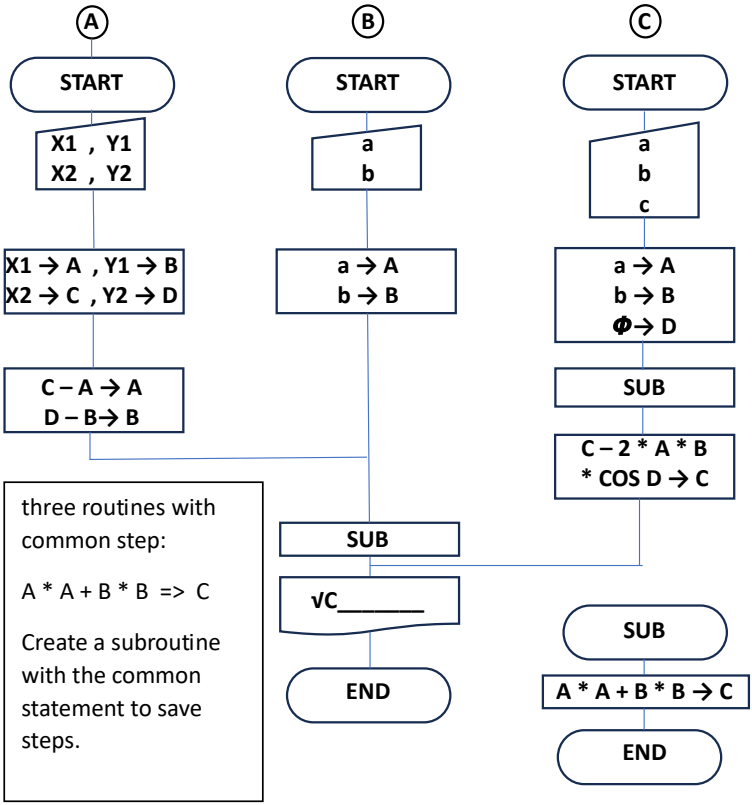
In a program when you use a certain calculation procedure several times, you can shorten the program by making he calcualton procedure a subroutine.

If the is a GTS instruction during program execution, it wil jump to the speciffied line and run the program from that line as a subroutine. The execution of the subroutine ends with the END instruction, and then returns to the next line of the GTS instruction and executes the instruction on that line.



THREE SEPARATE ROUTINES IN ONE LISTING:

0	HLT A, B ◣		Input A, B
1	√ (A*A + B*B) => C ◣	A	Solve for C
2	PRT C ◣		Print C
3	END ◣		
4	HLT A, B, C ◣		Input A, B, D
5	A * A + B * B => C ◣	B	Solve for C
6	2 * A * B * COS D => D ◣		Print C
7	PRT √ C (C – D) ◣		
8	END ◣		Input A, B, C, D
9	HLT A, B, C, D ◣	C	Solve for A and B
10	C – A => A, D – B => B ◣		Solve for C
11	√ (A * A + B * B) => C ◣		Print C
12	PRT C ◣		
13	END ◣		



three routines with common step:

A * A + B * B => C

Create a subroutine with the common statement to save steps.

Program 4

Line #	Program	
0	HLT A, B ◣	
1	▼1▼, GTS ▼SUB▼◣	Ⓐ label 1, to the subroutine a jump
2	▼2▼, PRT √C ◣	Ⓒ share
3	END ◣	Ⓑ share
4	HLT A, B, D◣	
5	GTS ▼SUB▼◣	
6	C−2∗A∗B∗COS D⇒C◣	Ⓑ jump to the subroutine
7	GTO ▼2▼◣	jump to label 2
8	END ◣	
9	HLT A, B, C, D◣	
10	C−A⇒A, D−B⇒B◣	Ⓒ
11	GTO ▼1▼◣	jump to label 1
12	END ◣	
13	▼SUB▼, A∗A+B∗B⇒C◣	
14	END END ◣	Ⓐ, Ⓑ, Ⓒ: share

Create the program in sections, in this case, 4 sections as indicated by the END commands. You can see the Label "SUB" has the two END END commands to terminate and return from the subroutine and then to terminate the program,

② execution of the program

Ⓐa = 4, b = 3

Ⓑa = 7, b = 5, θ = 30°

Ⓒa : (2, 5), b : (4, 7)

Program	Display	
AUT DEG		
CA [A]	0.	program A start
4 [A]	4.	
3 [A]	5.	result & print
GTO 4 [A]	4.	program B start
7 [A]	7.	
5 [A]	5.	
30 [A]	3.65762515	result & print
GTO 9 [A]	9.	program C start
2 [A]	2.	
5 [A]	5.	
4 [A]	4.	
7 [A]	2.828427125	result & print

5) IF (IF followed by conditionals).

Used with conditional. If there is an IF command during the execuction of the program, execute the conditional composed of the following instructions:

= ≠ ≥ > FLG

And if true, continue to exeute from the next line.

1 IF condiional:

When the value of the expression is not 0 it is considered true and executed from the next line.

Ex: 10; IF A > B, GTO 15 ◣

11; A + 1 => A ◣...

15; A * B => C ◣

- **A > B GTO line 15**
- **A ≤ B ignore the next statement, which is the GTO 15 command, and instead execute the following line, in this case, line 11;**

2 IF "variable" (memory):

If you define the IF memory "A", consider the test true when the value of A memory is not 0. If true execute from the next statement , and if false and execute from the next line.

3 IF FLG (IF FLG ,)

When the flag is set,

- **consider it as true and execute the next statement, else execute the next line.**

  Ex: 10; IF FLG , GTO 15 ◣

  11; A + 1 => A ◣ …

  15; A * B => C ◣

  - **Consider it true when FLG is set and execute from the statement "GTO 15".**
  - **Consider it false when the FLG is reset and execute from line 11;**

Example:

When X ≥ 5 execute statement: Y => LN (X^2 + √(1 + X^2))

When X < 5 then skip next statement and instead, execute the next line: Y => LN X

00; HLT X ◣

01; IF X ≥ 5 , LN (X * X + √(1 + X * X)) => Y , GTO 3 ◣

02; LN X => Y ◣

03; PRT Y ◣

04; END END ◣

6) FLG (FLAG)

(1) FLG : (FLG :)

a set of flags example.

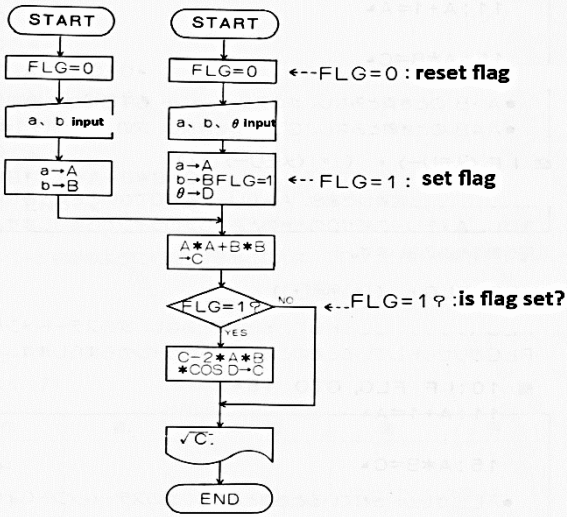
(2) FLG FLG : (FLG FLG :)

reset the flag.

(3) I F FLG : (IF FLG :)

test the flag status and then reset the flag

A program to find the Pythagoras theorem and the cosine theorem.



Program 5

Line #	Program	
0	FLG FLG ◣	flag reset/
1	HLT A. B. D.. FLG◣	if all 3 data points entered, flag is set.
2	A*A+B*B⇒C ◣	
3	I F FLG. C-2*A*B* COS D⇒C◣	if the flag is set you have a start of the program.
4	PRT √C◣	print sqrt of C
5	END END◣	

Program execution

(A) a = 7, b = 8

(B) a = 6, b = 9, θ = 42°

Program	Display	
AUTOモード, DEGモード		
[CA] [S]	0.	program start
7 [S]	7.	a input
8 [S]	8.	b input
[B]	10.63014581	continue program execute A
[S]	10.63014581	A execute
6 [S]	6.	a input
9 [S]	9.	b input
42 [S]	6.061382586	angle input B execute

7 DO・CNT ···DO, CONTINUE

Used to execute a certain part of the program repeatedly for the specified number of times using memory A.

If there is a DO instruction while the program is running, the part surrounded by DO and CNT (between the next line of the DO statement and the CNT command) will be executed repeatedly for the specified number of times.

Set DO and CNT as follows

M : DO initial value , final value

⋮

N : CNT ◣

Enter the initial value and the final value following DO.
When the DO commad is first executed in program the initial value is stored in the A memory, and then incremented by 1 each time until the value in A is equal to the final value.

This example)

M : DO 1 , 10 ◣

M+1 :

⋮

N : CNT ◣

Memory A is incremented by 1 for every pass of the loop and commands between the DO and CNT are executed in the loop until the value in A is equal to 10

- **If the DO order is not completed and the next DO is executed, the initial value and final value will be newly set in a later DO instruction.**
- **While being paired with DO - CNT, you cannot use another DO - CNT as a pair.**
- **Program the DO sentence to end the line with that command.**
- **The initial value and final value can also be specified. as an expression. But cannot be a negative value**

例 M : DO B+C , C*D ◣

⋮

N : CNT ◣

M until A memory is from the value B + C to the value of C * D.
The interval between line M and line N is executed repeatedly,
C * D is determined by the values of C and D at the time of the M line execution.

- **In the execution of the DO instruction, the initial value is stored in the A memory, but the final value is stored in other integer parts in the computer internal memory. The initial value and final value are possible in the range of 999, and if greater, error code 04 is displayed. The DO CNT instruction can be used effectively in the following cases**

- **DO CNT** **Ex: DO everything between the DO and CNT commands five times.**

```
M:DO1,5
  :
N:IF  B>C,GTO  P
  :
O:CNT
  :
P:END  END
```

- **DO,CNT** **Ex: two DO CNT loops controlled by a DO CNT loop!**

```
M:DO1,5
  :                repeat execution
N:CNT
  :
O:DO1,20
  :                repeat execution
P:CNT
```

- **CNT** **when using multiple instructions**

```
M:DO1,20
  :
N:IF  C=0,GTO  P      ①
  :                   ②
O:CNT
  :
P:B+D⇒D
  :
Q:CNT
```

① **M + 1 -> N -> O repeat the line.**
② **M + 1 -> N -> P -> Q repeat the line.**
① **to ② is the N line C=0 decided by a conditional.**

- **DO,CNT** **These DO CNT commands will generate error messages due to incorrect syntax.**

```
M:DO1,5
  :
N:CNT
  :
O:CNT
```

- **A running program DO and CNT of 1 to 5 loops**

```
M:DO1,5↘
   ⋮
N:          ←┐
   ⋮         ·
O:CNT↘       ·
   ⋮         ·
P:GTO  N↘  ──┘
```

8 **DEF switch (defining or assigning routines to keys).**

Selecting switch to DEF activates the A B C X Y keys as defined user keys. If any label in a routine uses any of these five letters, that key becomes a single push to allow the execution of the routine. The label is in the form of "A", followed by the comma then the rest of the routine.

- **A, B, C, X, Y If you input a number of data before operating the X label (routine) the data will be automatically stored in the Z memory so you can program without using the HLT function. In this case, corresponding to the key, C, X, Y is not specified, but the key operation causes an error, the Z memory will protect the previous contents.**

PROGRAM 6

Line #	Program	
0	▼A▼, HLT A, B↘	label "A"
1	▼1▼, GTS ▼SUB▼↘	
2	▼2▼, PRT √C↘	
3	END↘	
4	▼B▼, HLT A, B, D↘	label "B"
5	GTS ▼SUB▼↘	
6	C−2∗A∗B∗COS D⇒C↘	
7	GTO ▼2▼↘	
8	END↘	
9	▼C▼, HLT A, B, C, D↘	label "C"
10	C−A⇒A, D−B⇒B↘	
11	GTO ▼1▼↘	
12	END↘	
13	▼SUB▼, A∗A+B∗B⇒C↘	
14	END END↘	

Executing the program

Ⓐ a = 7, b = 12
Ⓑ a = 13, b = 7, θ = 23°25′
Ⓒ a : (2, 1), b : (7, 6)

Inputs	Display	
DEF mode DEG mode		
[A]	␣A	program A start
7 [=]	␣ 7.	
12 [=]	13. 89244399	answer
		program B start
[B]	␣B	
13 [=]	␣ 13.	
7 [=]	␣ 7.	
[F] [DEG] 23.25 [⇒] [=]	7. 140706611	answer
		program C start
[C]	␣C	
2 [=]	␣ 2.	
1 [=]	␣ 1.	
7 [=]	␣ 7.	
6 [=]	7. 071067812	

If you count the data and start the program (other than when it is stopped by the HLT command), the data will automatically be stored in Z memory.

9 PRT order PRINT

Used for printing results

(1) PRT (PRT , , ...)
print the calculation results of the formula sequentially.

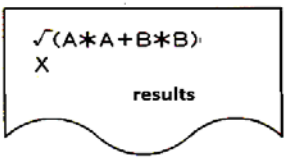
(2) PRT "...", "...", ... ([PRT] [F] [▼] ··· [F] [▼] [▼] [F] [▼] ··· [F] [▼] [▼] ···)
" ", print characters specified sequentially

(3) PRT, "X" ([PRT] [▼] "X" [▼] ···) where "X" is a character or characters.

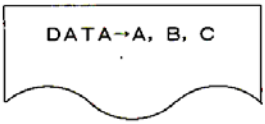
PRT, "...", ... ([PRT] [▼] [F] [▼] ··· [F] [▼] [▼] ···)
If you place a character (s) after the PRT command the printer will print one line then the result of the formula. " " print the characters in the specified field (between quotes)

- to cancel a print order use the ▶ instruction.
- 1 and 2 above can be combined arbitrarily.

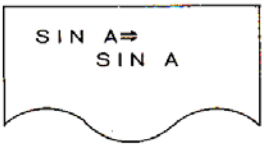
PRT √(A*A+B*B), X▶



PRT ▼DATA→A, B, C▼◣



PRT ▼SIN A⇒▼, SIN A◣



10 FED order FEED

An instruction to feed one blank line.

11 DSPi order DISPLAY

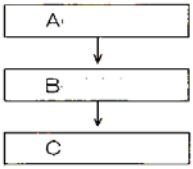
Used to check the results in the middle while the program is running. If there is a DSP instruction while the program is running, it will pause the execution of the program and display the contents of the specified memory, the value of the expression , characters, etc. for about 1 second.

(1) DSP X, X, ··· ([F] [DSP] 〈式〉 [▼] 〈式〉 [▼] ···) **where X is any character.**
The value of the expression (X) will be displayed sequentially.

(2) DSP▼···▼, ··· ([F] [DSP] [F] [▼] ··· [F] [▼] [▼]···)
▼ ▼ The characters will be displayed sequentially.

①DSP A, B, C◣

displays contents of A, B, C



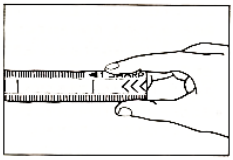
②DSP ▼DATA A, B▼, A*B◣

displays the phrase "DATA A, B" and then the value of A*B.

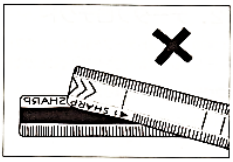
●DSPi **As shown in 1 and 2 above, the basic forms can be arbitrarily specified in combination. The release of the DSP instruction will be terminated with ▶ instuction or separated by a period (.)**

MAGNETIC CARDS:

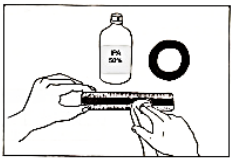
- **Figure 1: Correct handling. Magnetic strip facing down. Labelling should be readable.**



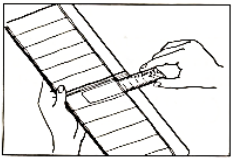
- **Figure 2: Incorrect insertion.**



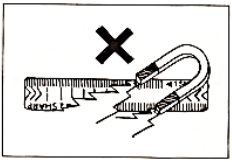
- **Figure 3: Clean with dilute IPA. Do not use abrasive cleaners.**



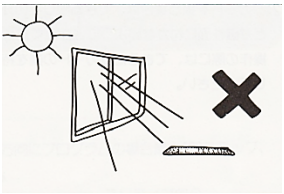
- **Figure 4: Store cards in protective holder.**



- **Figure 5: Do not expose to magnetic fields.**

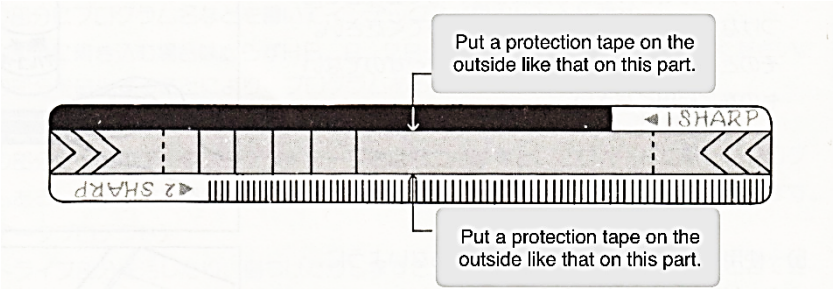


- **Do not expose to direct sunlight.**



Data protection

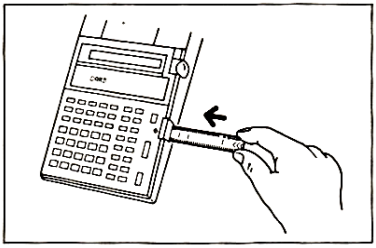
- **By attaching protective tape to the magnetic card, you can prohibit writing on the magnetic card and protect the recording content.**

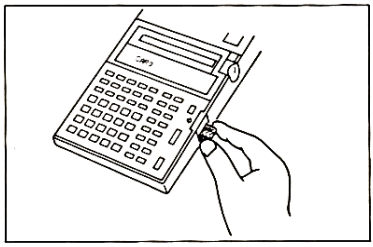


How to protect the tape

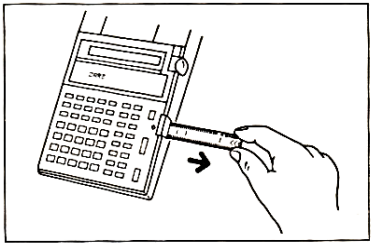
- **If you want to protect the contents recorded on the magnetic card, paste the protective tape on the striped part. If you want to protect the contents recorded on side 2. Place the protection tape from the writing side of the card (the side written as side 1 and side 2) to cover the striped part.**

- **Be careful not to let the tape protrude from the edge or surface of the card. If the tape is jutting out, the card operation will not be smooth, or the tape will peel off.**
- **In addition, please wipe the any dirt and etc. off the card before applying the protection tape.**





When you insert the card, the CARD display flashes (about 3 seconds), so while it is flashing, pull out the card at a constant speed and as much as possible.



REC **AUT / DEF modes. Used for recording cards. Load program in PRO mode. Once program is loaded, switch to AUT / DEF mode, press REC and observe flashing BUSY then CARD prompt. Insert card and observe flashing CARD prompt. Pull card and insert side two if prompted with a solid CARD prompt. Continue with new card if prompted again with new solid CARD prompt. There is a time limit for insertion and pulling the card.**

If error message ……05 then card was not recorded. A completed successful recording on one or more cards is indicated by the 0. display

LOD **AUT / DEF modes. Pressing LOD results in a flashing BUSY followed by a solid CARD prompt. Insert card and observe solid CARD prompt changes to flashing CARD display. Pull card out and either the LOD is complete (one card side) or a solid CARD prompt is displayed asking for the next side of card to be read. There is a time limit for insertion and pulling the card.**

If error message ……05 appears then card was not loaded correctly. A completed successful recording on one or more cards is indicated by the 0. display.

CHK **AUT / DEF modes. Once card or series of cards is / are read, pressing CHK will confirm the card was recorded correctly. Press CHK and observe flashing BUSY then solid CARD prompt. Insert the card and pull. If read and correctly checked then you will observe 0. or a CARD prompt for the next prompt. There is a time limit for insertion and pulling the card.**

If error message ……05 appears then card was not loaded correctly. A completed successful recording on one or more cards is indicated by the 0. display.

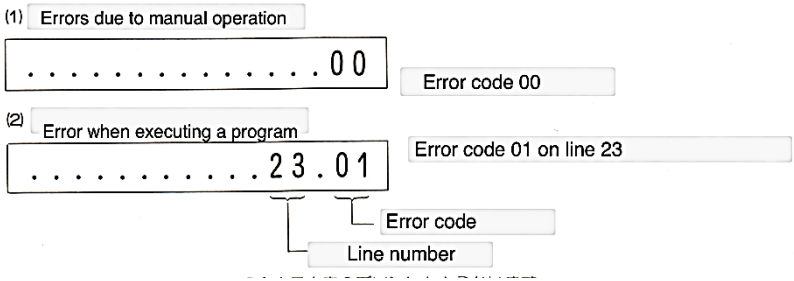
MAGNETIC CARD READER:

- **The magnetic card reader writes and reads data and programs to magnetic cards, and has a card reader head inside.**

Please pay attention to the following points when using the magnetic card reader.

- **Be careful not to mix dirt, oils, dust, and other foreign matter in the insertion of the magnetic card reader.**
- **Do not insert the default magnetic card, cards other than the cleaning card or anything else in the slot.**
- **Please clean the magnetic head from time to time (about once every 100 times of normal card operation) with the included cleaning card.**

ERROR MESSAGES:



CODE	REASON
00	• **If the absolute value is > 1 X 10^{100}** • **Divide by zero.** • **When incorrect input function is attempted.**
01	• **Meaningless operation. Using program functions while in AUT or DEF.** • **Combination of operations that are not defined.** • **Exceed 15 levels of function buffers or nine levels of numerical buffers.** • **When executing program and seeking undefined instructions.** • **Attempting to execute a nonexistent function.** • **Instructions valid only for characters ; -> <- ! ? …**
02	• **A place where the line number or label specified in the GTO or GTS instruction does not exist.** • **When no IF conditional exists, if the next line to be executed is not defined.** • **If the GTS instruction defines the next line (the line to be duplicated by the routine return) written in this instruction or if the line number exceeds 99.** • **When you operate a defined key that is not labelled while in DEF mode.**
03	• **Subroutine exceeds two levels.** • **The DO and CNT order when the CNT order is executed before executing the DO order**

04	• **DO instruction contains a negative number or ≥ 1000 as the initial and final values.** • **When specifying a negative line number in the GTO or GTS instruction.** • **Running a program that exceeds capacity.** • **Enter a program that exceeds capacity.** • **Specifying memory that does not exist in the indirect designation of memory.** • **Specify a step that exceeds the capacity with the cursor up operation.** • **If the following line is not defined in the skip operation of the HLT instruction.**
05	• **If the REC, LOD & CHK instructions at the time of program instruction execution do not define the next line of those written lines.** • **When executing the CHK instruction, if an error occurs due to the check operation.** • **If you execute the LOD instruction on a magnetic care where nothing is recorded (first card only).** • **LOD instruction, CHK for magnetic cards whose contents are broken due to external magnetic fields.** • **If you execute a LOD, CHK or REC instruction while the cover is attached to the magnetic card or on the head of the card reader** • **If the card operation is inappropriate.**
06	• **If you try to enter a program > 100 lines.** • **If you try to execute > 100 lines at the time of program execution.** • **If you try to specify a program that exceeds 100 lines by cursor up operation.**

The error message format is typical of Fortran based computers. If an error is detected the program halts at the line number that generated the error and displays a code explaining the error.

DISPLAY AND MEMORY:

- **The maximum number of digits displayed is 16.**
- **The maximum number displayed or entered for calculations is 10 plus two for exponent.**
- **26 data memories plus 25 indirect memories (PC-1300S)**
- **Max number of program steps is 640 (PC-1300s)**

OPERATING RANGE AND OVERFLOW:

- **This is a 10 digit internal memory machine. The operation is up to 9.999999999 -99 ≤ X ≤ +99 otherwise there is an overflow error or 0.**

FUNCTIONS:

• BASIC referred to by SHARP as the "FOUR FUNCTIONS"	+ - * /
• TRIGOMETRIC	SIN COS TAN
• INVERSE TRIGOMETRIC	ASN ACS ATN
• POWER EXPONENTIAL	(**) EXP
• LOGARITHMIC	LN LOG
• SQUARE ROOT	√
• INTEGER	INT
• ABSOLUTE VALUE	ABS
• RATIO OF CIRCUMFERENCE OF A CIRCLE TO ITS DIAMETER	π
• CODE FUNCTION	SGN
• DECIMAL ←→ DEGREE CONVERSION	DEG DMS
• LOGICAL FUNCTIONS (CONDITIONALS)	FLG = ≠ ≥ >

- **PROGRAM FUNCTIONS.**
    - **SHARP MINI FORTRAN LANGUAGE.**
    - **DIRECT MEMORY AND DESIGNATION.**
    - **JUDGEMENT** — **IF (= ≠ ≥ > FLG)**
    - **UNCONDITIONAL JUMP (ABSOLUTE, LABEL, INDIRECT)** — **GTO**
    - **SUBROUTINE JUMP (ABSOLUTE, LABEL, DIRECT)** — **GTS**
    - **ROUTINE LEVELS** — **2**
    - **DO LOOP (1 STAGE)** — **DO CNT**
    - **FLAG (ONE TYPE)** — **1**
    - **PRINT COMMAND** — **PRT**
    - **INUT AND PAUSE INSTRUCTION** — **HLT**
    - **DISPLAY INSTRUCTION** — **DSP**
    - **MAGNETIC CARD CONTROL COMMANDS** — **REC LOD CHK**
    - **END OF PROGRAM, SUBROUTINE, CARD RECORD** — **END**
    - **EDITING FUNCTIONS (INSERT, DELETE)** — **INS DEL**
    - **CHECKING FUNCTIONS (LIST AND TRACE)** — **LST TRC**

PHYSICAL ATTRIBUTES:

- **LSI**
- **4.8 VDC**
- **100 VAC 50/60 HZ**
- **USAGE TIME ABOUT 4.5 HOURS (NOT PRINTING)**
- **OR ABOUT 5000 LINES PRINTED STARTING AT LINE 00;**
- **WILL EXECUTE 55555555 GTO 0 INSTRUCTIONS.**
- **POWER CONSUMPTION IS 4.8VDC AT 5 WATTS**
- **OPERATING TEMPERATURE 0'C → 40'C**

PAPER:

- **METALIZED ALUMINUM**
- **PAPER ROLL DIMENSIONS** — **WIDTH ≤ 35.8 MM**

シャープ エルシーメイト

形名 PC-1300S 補足説明書

SHARP ELSIMATE PC-1300S

SUPPLEMENTARY INFORMATION

INTRODUCTION:

Thank you for purchasing the Sharp Pocket Computer (PC-1300S)

This is a high performance program machine that enables the execution of a more extensive measure by expanding the program capabilities of the PC-1300 with memory expansion.

Please read this supplementary note for the PC-1300S and PC-1300 instruction manual carefully before use.

Explanation:

the instruction of this machine is almost the same as the PC-1300, so the instruction manual is attached to the PC-1300, but this machine has expanded the program capacity and memory, how to specify the memory, magnetic card useage. There are some differences in the explanation of the order, conditions of error, etc.

① MEMORY:

The PC-1300S has 42 memories, each assigned a numerical value of 0 -> 41. In addition, each of the 26 memories from 0 -> 25 is named A -> Z and is identical to the PC-1300. Memories 26 -> 41 can only be accessed indirectly.

X memory A -> Z where X is 26 -> 41

By operating with, specify the desired memory. For example, when you use memory to store the result of 7 * 4 in memory 40 and recall when needed, operate as follows:

40 [⇨] [A] [▾]	Memory 40 in Memory A
7 [✱] 4 [⇨] [X] [A] [▾]	Memory 40 to remember the result of 7 * 4
([X] [A] ,	(Memory 40 is specified by pressing X A)
[X] [A] [▾]	Realling contents of memory 40

② REC order (REC ,) See REC explanation in PC 1300 manual

This instruction for recording data in RAM on to a magnetic card, and records the data of 16 memories sequentially on to the magnetic card. When it is less than 16 memories from the specified memory to the later memory (memory 41), record the later of that number of memories, and record the rest.

例 REC C , Record the number of memories from C to R on the card.

REC X , X, Y, Z, 26 - 38 Record the contents of the memory on the card.

REC XA , For example, if 30 is stored in A, 30 -> 40 memory are recorded on the card. The rest are not.

③ LOD

2) LOD mode (LOD (memory) -> ,)

Command for loading DATA on a card to the computer. Transferred from the specified memory to the 16th memory. For example, if 3 pieces of data are recorded on the card, and the rest is recorded 0, 0 will be sent to the memory from the 4th to the 6th.

LOD A, data will be sent to the memories A -> P (1 -> 16)

LOD XA, for example, if 25 is stored in memory 25 (Z) The data recorded on the card will be transferred from memory to 40.

④ CHK

2) CHK mode (CHK (memory) -> ,)

In order to check if there is a difference between the data in the memory and the data of the card, from the specified memory. Check includes 16 memories.

- any data that is not included in the specified range will be 0.

⑤ LST

Printing a list of programs includes the number of steps and number of steps at the end. If the program is 340 steps then the end of the listing will indicate steps used and steps unused: ex) 340 300

⑥ About error 03

③ PRO mode PRO mode has two types of errors only

1) program exceeds 640 steps (over capacity)
2) line number exceeds 99.

1) in the case of 1, if you set it in AUT or DEF mode and release the error by CA, when you set it to PRO mode, the error occurred on step 641 will be indicated by cursor display.

⑦ Error code 04

- If the next line is not fixed in the skip work of the HLT instruction, the program will specify the final step (640 steps) and get an error.

⑧ Memory capacity

Program memory	640 steps
Data memory	42 registers

SHARP CORPORATION

9M05T(2374)

TWO PROGRAM EXAMPLES:

To introduce programming an example of the Great Circle track and Great Circle Plotting (also known as Intermediate Latitude from Intermediate Longitude) program is given. The code in single command format is on the left, and explanation on the right.

For the A B C X Y to function correctly as user assigned keys, the switch must be selected too DEF. In the following program "A" and "B" are the subroutine labels. In this example, with the switch selected to DEF, only the A or B keys need to be pressed after executing the load routine. If in AUT mode then GTO "A" => or GTO "B" => would need to be entered.

FORMULA: GREAT CIRCLE

When the terrestrial latitudes and longitudes of two points are given, this formula allows the calculation of the shortest distance between the two points and the azimuth (true track) at the start point.

$$DISTANCE\ (NM) =$$

$$60 * COS - 1\{SIN\ LA1 * SIN\ LA2 + COS\ LA1 * COS\ LA2 * COS\ (LO2 - LO1)\}$$

$$TRUE\ TRACK = ACOS\{\frac{SIN\ LO2 - SINLO1 * COS\left(\frac{D}{60}\right)}{SIN\ \left(\frac{D}{60}\right)}\}$$

IF SIN (LO2-LO1) ≥ 0; THEN Φ = Φ

IF SIN (LO2-LO1) < 0; THEN $\Phi = 360° - \Phi$

- **ENTER NORTH AND WEST POINTS AS POSITIVE NUMBER.**
- **ENTER SOUTH AND EAST POINTS AS NEGATIVE NUMBER.**
- **TRUE TRACK = Φ**

FORMULA: INTERMEDIATE LATITUDE

When the terrestrial latitudes and longitudes of two points are given, this formula allows the calculation of an intermediate latitude given an intermediate longitude between the two initial points.

$$LATi = ATAN\left(\frac{A - B}{SIN(LONd - LONs)}\right)$$

$$A = (TAN\ LATd * COS\ LONs - TAN\ LATs * COS\ LONd) * SIN\ LONi$$

$$B = (TAN\ LATd * SIN\ LONs - TAN\ LATs * SIN\ LONd) * COS\ LONi$$

WHERE:

- **LATs = LATITUDE STARTING**
- **LONs = LONGITUDE STARTING**
- **LATd = LATITUDE DESTINATION**
- **LONd = LONGITUDE DESTINATION**
- **LONi = LONGITUDE INTERMEDIATE**

Each line of the program is explained starting with 00;

LINE	CODE	*Explanation.*
00;	DSP “CO-ORDS + ILO” ◣	*Display the title of the program and pause. DSP has an automatic temporary pause before executing the next line.*
01;	HLT A,B,C,D,E	*HLT halts the program and sequentially allows the variables to be entered. After each variable is entered, the* ◣ *key must be pressed to continue to the next variable and then to execute the remaining lines of the program*
02; 03; 04; 05; 06;	DEG A => A *departure latitude* DEG B => B *departure longitude* DEG C => C *arrival latitude* DEG D => D *arrival longitude* DEG E => E *intermediate longitude*	*These sequences of code change the degree, minute and seconds into decimal degrees for mathematical computation. The resulting answers, if required, can be converted back to degrees, minutes and seconds. Each variable is stored in a memory. There are 26 main memories available as well as indirect memories. The main memories are A → Z. Here the memories A →E are used.*

07;	DSP “GC -> A OR IL -> B” ◣	*This display prompts the user to select which routine they are interested. In this case pressing the “A” key on the keyboard will execute the Great Circle program.*
08;	HLT	*This is used to stop the program to allow the execution of the “A” key or the “B” key. Otherwise, the display would only pause and then continue with the next line of code instead of the desired selection.*
09;	“A” , DSP “GREAT CIRCLE” ◣	*The “A”, is the label for the Great Circle program. When the DEF switch is selected on the keyboard, the **A,B,C,X,Y** become active and will execute any routine or subroutine with the same label.* *The comma after the “A” separates two commands on the same line.* *In this case, because the keyboard switch is in DEF (defined) and the **A** key is pressed, the result is a display of*

09;	"A" , DSP "GREAT CIRCLE" *continued*	*GREAT CIRCLE followed by a pause and then an Automatic execution of the rest of the code until a HLT, END or another DSP command is read.*
10;	(ACS (SIN A * SIN C + COS A * COS C * COS (D – B))) * 60 => I ◣	*The distance calculation given the departure and destination latitudes and longitudes.* *The distance is stored in memory I.*
11;	DSP "DISTANCE ="◣	*The command DSP is used to displace the output message that the next number will be the distance between the two points.*
12;	INT (I) => I , I => HLT, PRT I ◣	*INT is the integer command. That is, select only the integer portion of the result and store it in memory I, which replaces the integer/ decimal value with the integer value only. This is a choice in this program but may differ for another program.* *I=>HLT stops the program at this point to display the answer.*

12;	INT (I) => I , I => HLT, PRT I *continued*	*PRT is used to print the answer.* *Note the use of the comma to delineate commands on the same line.* *When the ▶ is pressed after the HLT command in this line to continue execution, the result is printed and the program executes the track computation of the Great Circle routine.*
13;	ACS ((SIN C – SIN A * COS (I / 60)) / (SIN (I / 60) * COS A)) => J ◣	*The track computation of the Great Circle routine which is stored in memory J.*
14;	IF SIN (D-A) ≥ 0 , GTO "360" ◣	*An example of the conditional ≥ to determine if the initial true track is to the west or the east.* *The direction is determined by the positive or negative value of the SIN (see formula). If positive GTO (do not return) a routine labelled 360. If negative continue to the next line.*

14;	IF SIN (D-A) ≥ 0 , GTO “360” ◣ *continued*	*The command IF is used in this line as with all conditionals.*
15;	DSP “TRACK=“	*Display the name of the result and pause (not stop).*
16;	INT (J) => J , J => HLT ◣	*As with the distance calculation, only display the integer and ignore the fractional part. This was done to simplify the displayed answer as the trailing decimal points are not required.*
17;	“360”, 360 – J => J ◣	*The label 360 which is used to correct the direction of the initial track and resulted from the conditional test ≥ from line*
18;	DSP “TRACK=“ ◣	*Display the name of the result and pause (not stop).*
19;	INT (J) => J , J => HLT ◣	*As with the distance calculation, only display the integer and ignore the fractional part. This was done to simplify the displayed answer as the trailing decimal points are not required.*
20;	“B” , DSP “ INT. LATITUDE” ◣	*Automatic execution of the rest of the code until a HLT, END or another DSP command is read.*

21;	(TAN C * COS B – TAN A * COS D) * SIN E => F ◣	*Part 1 of computation of the Intermediate Latitude routine which is stored in memory F.*
22;	(TAN C * SIN B – TAN A * SIN D) * COS E => G ◣	*Part 2 of the computation of the Intermediate Latitude routine which is stored in memory G.*
23;	ATN ((F – G) / SIN (D – B) => H ◣	*Part 3 of computation of the Intermediate Latitude routine which is stored in memory F.*
24;	DMS H => H ◣	*Convert the decimal degrees into degrees, minutes and seconds.*
25;	DSP “INT. LATITUDE” ◣	*Label the output and pause.*
26;	H => HLT , PRT H ◣	*Display the numerical result, then when ▶ is pressed print the result.*
27;	END END ◣	*Two END commands are required. One to end the program and another to end the record function on a card.*

PROGRAM OPERATION				
step	KEY	INPUT	OUTPUT	DESCRIPTION
Program initialization				
1[1]	Select DEF , CA			Program start.
2	R/S		CO-ORDS + ILO	Displays routine name.
3	R/S		LA1->LO2 + ILO	Display and data prompt.
4	R/S	Departure latitude	Departure latitude	
5	R/S	Departure longitude	Departure longitude	
6	R/S	Arrival latitude	Arrival latitude	
7	R/S	Arrival longitude	Arrival longitude	
8	R/S	Intermediate longitude	Intermediate longitude	
Prompt for desired program				
9[2,3]	R/S		GC -> "A" OR IL -> "B"	Display and prompt for Great Circle or Intermediate Latitude routine.
Great Circle Distance and Track				
10	"A"		GREAT CIRCLE	Displays routine name.
11	R/S		Distance in nm.	
12	R/S		True track in degrees	Program halts.
Intermediate Latitude from given Longitude [4]				
13	"B"		INTERMEDIATE LATITUDE	Displays routine name.
14			INTERMEDIATE LATITUDE	
15			latitude	In ddd.mmss

1 - select DEF key to initialize user assigned keys. Load program.
2 - at step 9 press "A" instruction to execute the Great Circle routine.
3 - at step 9 press "B" instruction would bypass the Great Circle routine and go directly to the Intermediate Latitude routine.
4 - uses data from program start including ILO to output in degrees, minutes & seconds (ddd.mmss).
* - block letters indicate alpha display.

PROGRAM EXAMPLE

FIND: GREAT CIRCLE DISTANCE INTIAL TRUE TRACK INTERMEDIATE LATITUDE
GIVEN: VANCOUVER = N49 WEST 123 LONDON = N53 WEST 0 INTERMEDIATE LONGITUDE= WEST 060

step	KEY	INPUT	OUTPUT	DESCRIPTION
Program initialization				
1[1]	Select DEF, CA			Program start.
2	℅		CO-ORDS + ILO	Displays routine name.
3	℅		LA1->LO2 + ILO	Display and data prompt.
4	℅	49	49	DepartureLatitude
5	℅	123	123	Departure longitude
6	℅	53	53	Arrival latitude
7	℅	0	0	Arrival longitude
8	℅	60	60	Intermediate longitude
Prompt for desired program				
9[2, 3]	℅		GC -> "A" OR IL -> "B"	Display and prompt for Great Circle or Intermediate Latitude routine.
Great Circle Distance and Track				
10	"A"		GREAT CIRCLE	Displays routine name.
11			DISTANCE=	
12	℅		4031	
13			TRUE TRACK=	
14	℅		33s	Program halts.
Intermediate Latitude between two points and given Longitude				
15	"B"		INTERMEDIATE LATITUDE	Displays routine name.
16			INTERMEDIATE LATITUDE=	
17			60.5645	In ddd.mm.ss

FORMULA: ALTITUDE CORRECTION FROM FAA DOCUMENT

When temperatures are below 0' Celsius the baro altimeters will give erroneous ASL readings and must be corrected for the effect of cold temperatures. The actual altitude will be lower than the indicated altitude on the Baro-altimeter. The following formula will correct the altitudes to allow for a safe approach to the runway.

https://www.faa.gov/air_traffic/flight_info/aeronav/digital_products/dtpp/search/

TBL 7–3–1
ICAO Cold Temperature Error Table

HEIGHT ABOVE AIRPORT IN FEET

REPORTED TEMP °C	200	300	400	500	600	700	800	900	1000	1500	2000	3000	4000	5000
+10	10	10	10	10	20	20	20	20	20	30	40	60	80	90
0	20	20	30	30	40	40	50	50	60	90	120	170	230	280
-10	20	30	40	50	60	70	80	90	100	150	200	290	390	490
-20	30	50	60	70	90	100	120	130	140	210	280	420	570	710
-30	40	60	80	100	120	140	150	170	190	280	380	570	760	950
-40	50	80	100	120	150	170	190	220	240	360	480	720	970	1210
-50	60	90	120	150	180	210	240	270	300	450	590	890	1190	1500

$$\text{Correction} = H * \frac{15 - t_0}{273 + t_0 - 0.5 * L_0 * (H + H_{SS})}$$

H = minimum height above the altimeter setting source
(the setting source is normally the aerodrome, unless otherwise specified)

$T_0 = t_{aerodrome} + L_0 * h_{aerodrome}$ aerodrome (or specified temperature reporting point) temperature adjusted to sea level

L_0 = 0.0065°C per metre or 0.00198°C per foot

H_{SS} = altimeter setting source elevation

$t_{aerodrome}$ = aerodrome (or specified temperature reporting point) temperature

$h_{aerodrome}$ = aerodrome (or specified temperature reporting point) elevation

Vancouver airport 14' ASL
Temperature -25'C

LINE	CODE	explanation
00;	DSP "ALTITUDE CORRECTION" ◣	*Routine name.*
01;	DSP "ATMP AASL FASL" ◣	*Variable prompt order.*
02;	HLT A,B,C ◣	*Input:* - *airport temperature.* - *Airport altitude above seal level.* - *Fix altitude (this is the variable that changes each time the subroutine is called).*
03;	.00198 => R ◣	*constant*
04;	A + R * B => T ◣	
05;	"A" ◣	Subroutine which is used for the next corrected altitude computation.
06;	C *((15 – T) / ((273 + T) -.5 * R * (B + C)))=> H ◣	
07;	H + C => H ◣	
08;	INT (H) => H ◣	
09;	DSP " CORR. ALTITUDE=" ◣	
10;	H => HLT, PRT H ◣	
11;	DSP "NEXT FIX. ALT?" ◣	*prompt request for the next fix altitude*
12;	HLT C ◣	*stop and input next altitude. Press the "A" key and the new corrected altitude is displayed.*
13;	END END ◣	

Minimum Safe Altitude (MSA)	**10,00'**
FINAL APPROACH FIX (FAF)	**1400'**
DH / DA / MDA	**214'**

Once the first two inputs are made, they remain constant and the routine only requires the MSA → MDA.

PROGRAM OPERATION

step	KEY	INPUT	OUTPUT	DESCRIPTION
Program initialization				
1[1]	Select DEF, CA		ALTITUDE CORRECTION*	Program start.
2	R/S		ATMP AASL FASL	Displays routine name.
3	R/S	Airport temperature		Display and data prompt.
4	R/S	Airport altitude		
5	R/S	Fix altitude		
6	R/S		Corrected altitude	Uses subroutine new to enter successive Fix altitudes: MSA, SECTOR, IF, FAF,MDA/DH, MAP
7	"A"	New Fix altitude	Corrected altitude	
8	"A"	New Fix altitude	Corrected altitude	

[1] - places program at line 000 and ready for execution
* - block letters indicate alpha display.

PROGRAM EXAMPLE

FIND: CORRECTED ALTITUDES.
GIVEN: CYVR; altitude 14' asl airport temperature -25°C MSA (10,000') FAF (1,400') MDA/DH (214')

step	KEY	INPUT	OUTPUT	DESCRIPTION
Program initialization.				
1[1]	Select DEF, CA			Program start.
2	R/S		ALTITUDE CORRECTION	Displays routine name.
3	R/S		ATMP AASL FASL	Display and data prompt.
4	R/S	-25	-25	After entering the initial three variables only the altitude will be prompted for as the airport altitude and temperature remain the same. Subroutine "A" saves input labour and time.
5	R/S	14	14	
6	R/S	10,000	11678	
7	"A"	1400	1626	
8	"A"	214	248	

NOTES:

NOTES:

NOTES: