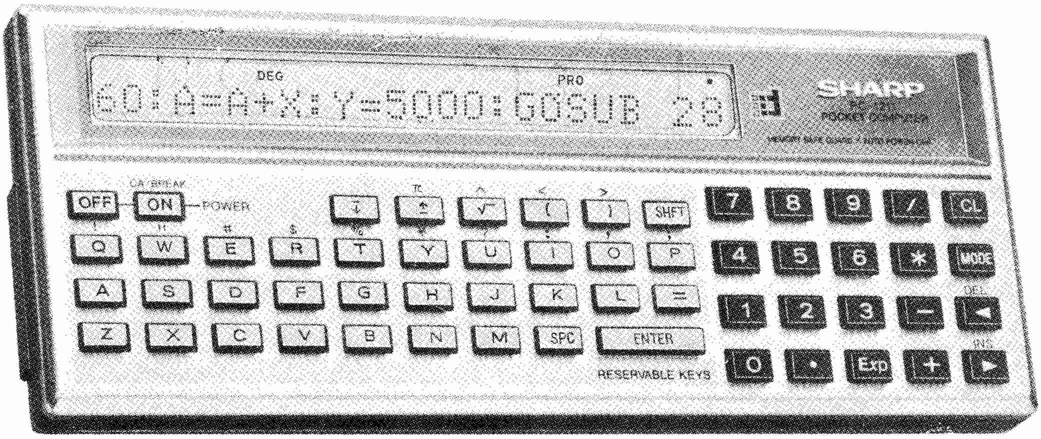
SHARP

Pocket Computer Model PC-1211
Beginner's Textbook for "BASIC"



SHARP CORPORATION

PREFACE

This text has been prepared for fast and easy self-learning of the SHARP Pocket Computer PC-1211.

The text begins with the chapter dealing with manual operation and develops further into higher programming techniques and deals with programming of higher frequency as pages go by. Problem, its Solution and Notes are provided in each section and Exercises are also available at the end of each chapter. You can advance to a next page when you are correct in solving the Problem, or you must go through the Notes if you were to fail to make out proper understanding of the Problem and Solution, and you have to try to make further understanding through Exercises.

In each section and Problem are affixed a grade marking according to a level of problem, and those who bound with studying time can proceed omitting some of high grade marks. (Mark with single asterisk (☆) denotes a middle grade, the one with double asterisks (☆ ☆), a high grade and triple asterisks (☆ ☆ ☆), a special grade. As the keyboard chart of the computer is attached in the Supplement-3 at the end of this text, anyone without the computer at hand may refer to it. Also, "BASIC" instruction sets and their mnemonics are provided in the Supplement—1, and the reader can use the list for convenience.

With respect to detailed description of instructions, please refer to the Instruction Manual.

TABLE OF CONTENTS

FOREWORD   Before studying "BASIC"

Recently, the word "BASIC" is often heard not only among computer engineers but among those ranging from elementary school to college students and white collars. It is one of computer languages which is the most prevailing among others. Why is it so? The word "BASIC" is the abbreviation derived from "Beginner's All purpose Symbolic Instruction Code", which in itself implies easy understanding by the beginners. The program consists of conversational language without much restriction in its syntax and written in plain English and known for high degree of population because of easy comparison with the flowchart and advantages owing to fairly small numbers of instruction set used.

Like human languages such as English, Japanese and German, varieties of languages are used for the computer languages and that BASIC is also one of computer ianguages like FORTRAN, COBOL, APL, etc. As there are dialogues in English like western and southern dialogues, there are also dialogues in BASIC as differently developed by each computer manufacturers. But, the fundamentals of the language is all the same. Although such kind of peculiarity may be seen in the BASIC used for PC-1211, it will be more powerful and effective as this class of compact computer and it will be welcomed by the users for its ease of operation and substantial functional performances. We are sure that you will appreciate these advantages. We hope use of this computer will help to increase routine efficiencies and, in addition, knowledge you have acquired with this BASIC will assist you a great deal in using a higher class of machine in future time.

As in car driving and golfing, simply reading a book would not help you much. You have to manoeuvre the steering wheel by yourself or swing a gold club by yourself in order to attain progress. It is natural that you can not attain much in the beginning as you can not make a par first time you are on a golf course. Don't being called awkward or being laughed at, else you will be further delaying your progression. Key to progress is to practice programming and use the machine whenever possible. Well, let's start from the Chapter 1 with determination.

Chapter 1 Manual operation

Intension of this chapter is to master various fundamentals of operation in the manual mode.

Preparation 1: Switch on

Depress the ON key in the first place. Then, the prompt symbol ">" will appear on the lefthand side of the display.

Preparation 2: Mode selection

When power is on, either one of small indications, DEG, RAD, or GRAD, is brought up on the upper left of the display, and one of DEF, RUN, PRO, or RESERVE is also indicated on the upper right of the display. The former represents an angular mode which is used in the operation of trigonometric function, and the latter represents an operation mode, which will start changing from DEF → RUN → PRO → RESERVE → DEF in rotational sequence each time the MODE key is depressed. To perform manual operation, it has to be set to [RUN] mode.

NOTE: As the auto power off function is adopted in the computer, the power will be automatically turned off if the machine is left unused for approx. 7 minutes without performing any computation or key operation. (The power will be instantly shut off when the OFF key is depressed.)

A tiny dot symbol " • " appearing on the upper right of the display is a battery indicator, by which need for battery replacement is called for when this mark disappears from the display.

Section 1 Fundamental operation

[Problem] Perform the following calculations.

(1) $2+3\times 4=$

(2) $5\times(-6)=$

(3) $5\times(-6)+7=$

(4) $36\div(1+2)=$

(5) $\dfrac{4+5}{2+3}=$

(6) $1+2(-3+4(5+6(7-8)))=$

(7) $5\times 10^{3}\div(4\times 10^{-3})=$

(8) $-1.2\times(-4)+(-5)\times 6=$

(9) $2\pi\times 5=$

[Solution]

(1)	2 + 3 * 4 ENTER	14.
(2)	5 * − 6 ENTER	−30.
(3)	5 * − 6 + 7 ENTER	−23.
(4)	36 / (1 + 2) ENTER	12.
(5)	(4 + 5) / (2 + 3) ENTER	1.8
(6)	1 + 2 * (− 3 + 4 * (5 + 6 * (7 − 8))) ENTER	−13.
(7)	5 EXP 3 / 4 EXP − 3 ENTER	1250000.
(8)	− 1.2 * − 4 + − 5 * 6 ENTER	−25.2
(9)	2 π * 5 ENTER	31.41592654

[Notes]

○ Basic keys

Add	+
Subtract	−
Multiply	*
Divide	/
Symbol	+ or −
Execution	ENTER

○ Operational priorities

With this pocket computer, everything can be entered through the keyboard according to what the expression indicates including bracket and functions. Decision for sequential priority of data and processing of intermediate results can be automatically handled by the computer itself.

Priority of computation takes place in the following sequence:

1. Function (such as sin, cos, etc.)
2. Power calculation
3. Symbol
4. Multiplication & division
5. Addition & subtraction
6. Logical operation (such as >, >=, etc.)

NOTE:
- Bracket "( )" takes precedence over every operations.
- Calculation of a compound function takes place from right to left (i.e. sin cos 30).
- Serial calculation of power is performed from right to left (i.e. $2 \wedge 3 \wedge 2$).

○ Do not make mistake in entering the bracket in the Problem (5).
Compare the followings:

$$4+\frac{5}{2}+3 \quad , \quad \frac{4+5}{2}+3 \quad , \quad 4+\frac{5}{2+3}$$

○ Operational buffer

As seen in the Problem (4), number (36), in this case and function (such as /, (etc.) that can not be calculated at one time is put into memory temporarily and execution takes place from the higher priority. Such a location as to store data temporarily is called a buffer, and the following numbers of stages are provided in this pocket computer.

Numerical buffer	8 stages
Functional buffer	16 stages

○ Length of formula

80 key operations are possible for an input of one formula (until the ENTER key is depressed).

○ Correction of erroneous key operation

Anyone can make mistake. Therefore, the following method should be remembered to cope with erroneous key input.

i) Depression of the CL key makes all display contents deleted with a > symbol appearing on the lefthand side of the display. Then, you can resume the operation from the beginning.

ii) When other wrong key is depressed.
Keep the ◄ key depressed until the error made sector appears flickering on the display, then make correct key entry. (Continuous depression of the ► or ◄ key makes the cursor moving.)

```
Ex:   When 2 + 3 * 4  is entered as  2 + 3 / 4  in the Problem (1),
      2 + 3 / 4
            ↑
            └── Let this portion be flickering, then depress * and ENTER .
```

iii) When an extra key is depressed unnecessarily.
Keep the ◄ key depressed until the unnecessary entry appears flickering on the display, then cancell that entry by depressing the DEL key.
Depressing the DEL key instead of the SPC key will cause the same effect. ("DEL" stands for "delete".)

```
Ex:   When 2 + 3 */ 4  is entered in the Problem (1),
      2 + 3 * / 4
              ↑
              └── This section flickers on the display.

      Depress the DEL key,
      2 + 3 * 4
              ↑
              └── This point flickers on the display.

      ENTER                       14. (correct)
```

iv) When necessary key entry is skipped.
Keep the ◄ key depressed until the skipped portion comes flickering on the display, then depress the INS key. ("INS" stands for "insert".)
As there will appear one column of blank on the display, enter the skipped data or function to that blank area.

Ex: When 2 + 3 4 is entered in the Problem (1),

```
Depress the [◄] key        2 + 3 4
                                 ↑
                                 └─── Flickers

Depress the [INS] key      2 + 3 □ 4
                                 ↑
                                 └─── Flickers

Depress the ∗ key          2 + 3 ∗ 4
                                   ↑
                                   └─── Flickers

Depress the [ENTER] key         14. (correct)
```

○ Playback function

When either the [◄] or [►] key is depressed after the execution of operation with the [ENTER] key, it will restore the previous execution on the display. This will be convenient to make confirmation of the executed operation or when the execution is to be utilized successively after making some modifications. Especially, when an error is resulted after the execution with the [ENTER] key termination, it will be much convenient to locate the error portion as it is being flickered on the display.

Ex: When an error is occurred with 3 6 / 1 + 2) [ENTER] in the Problem (4).

```
Depress the [◄] key

        3 6 / 1 + 2 )
                    ↑
                    └─── Flickering
```

○ Execution of continuous computation

Depression of the [ENTER] key makes the result shown on the display, but the result can be further utilized in a successive operation.

Ex:

```
1 + 2 [ENTER]                3.
∗ 4 [ENTER]                 12.  (execution of 3 x 4)
```

○ Cursor

An underline "_" appears on the display when the key entry is made or it will appear flickering when error correction is made. This underline shows that the next key entry is placed in this portion. This is called a cursor.

○ Error indication (see the List of Error Code in the Supplement–2)

Operation error is indicated with "1 · · · · · · · · · · · · · · ·" on the display.

Section 2 Functional operation

In this section discussion is given about the performance of functions.

[Problem] Perform the following calculations.

(1) $\sin 30° + \cos 60° \times \tan 45° =$

(2) $\tan \frac{\pi}{4} =$

(3) $\sin (15° + 30° \div 2) =$

(4) $\sqrt{3^2 + 4^2} =$

(5) $\sin^2 30° =$

(6) $\log 5 + \ln 5 =$

(7) Convert 30°30′ in sexagesimal notation into decimal notation.

(8) Convert 30.5° in decimal notation into sexagesimal notation.

[Solution] Set the angle mode to DEG in (1), (3) and (5), and to RAD in (2).

```
(1)  SIN30+COS 60* TAN45 ENTER                 1.
(2)  TAN (π/4) ENTER                           1.
(3)  SIN(15+30/2) ENTER                       0.5
(4)  √(3^2+4^2) ENTER                          5.
(5)  SIN30^2 ENTER                           0.25
(6)  LOG 5+LN 5 ENTER                 2.308407917
(7)  DEG 30.30 ENTER                         30.5
(8)  DMS 30.5 ENTER                          30.3
```

[Notes]

○ Designation of angle mode

It is essential to set up the angle mode to one of the following designation when any of trigonometric or inverse trigonometric function is to be processed.

Angle mode	Set up sequence	Note
Degree	DEGREE ENTER	A unit with which the right angle is expressed by 90°.
Radian	RADIAN ENTER	A unit with which the right angle is expressed by $\frac{\pi}{2}$.
Grad	GRAD ENTER	A unit with which the right angle is expressed by 100.

○ Outline of functional performance of the PC-1211.

Almost all of the scientific calculators are provided with individual function keys such as SIN and COS, but, on the other hand, the most of the BASIC performance pocket computers are designed to enter the function code through the alphabetical keyboard, entering in the sequence of S, I, and N to express SIN, just as though through the typewriter keyboard.

Function	Symbol in general	Operation by this machine
Trigonometric function	sin cos tan	SIN COS TAN
Inverse trigonometric function	$\sin^{-1}$ $\cos^{-1}$ $\tan^{-1}$	ASN ACS ATN
Common logarithm	log	LOG
Natural logarithm	ln	LN
Exponential function	exp	EXP
Decimal-to-sexagesimal conversion		DMS
Sexagesimal-to-decimal conversion		DEG
Integerization		INT
Absolute value	$\|x\|$	ABS
Square root	$\sqrt{\ }$	$\sqrt{\ }$
Sign function		SGN

○ Typewriter keyboard layout

Does anybody not familiar with the typewriter keyboard think it is annoying to hit keys like S, I and N, and think it is far much better to use individual keys like $\boxed{\text{SIN}}$ and $\boxed{\text{COS}}$? No, he is wrong. Operating multiples of keys counted up almost to 100, which are essential to perform varieties of functions, is not an easy practice. There are many instructions such as SIN, GOTO · · · ·, etc., for the operation of computer. It will be a hard job to remember all of key layout for any of individual key operated machine. Apart from this, different key layout is adopted depending on the manufacturer. On the other hand, the typewriter keyboard is standardized worldwide under such as ASCII and JIS, and you will be accustomed to pick up any key once you have been familiar with the typewriter keyboard which is composed of 30 to 40 keys.

Being familiar with the typewriter keyboard will help you a great deal when you are to operate an upper class of computer.

However, with this pocket computer, a reservable key method (Section 4) and mnemonics (Supplement—1) are provided for avoiding key annoyance.

Section 3 Memory operation

In this section is discussed memory configuration and its usage.

[Problem] Perform the following calculations.

(1) (i) Store 3 in the memory A and 4 in the memory B.

(ii) Compute $\sqrt{A^2 + B^2}$ and store the result in the memory C.

(iii) Check the contents of the memories A, B, and C in turn.

(2) (i) Add 1 to each of A and B, above.

(ii) Compute $\sqrt{A^2 + B^2}$ and store the result in the memory C.

(iii) Check the contents of the memories A, B, and C in turn.

(3) (i) Store the word "SHARP" in the memory D and "PC-1211" in the memory E.

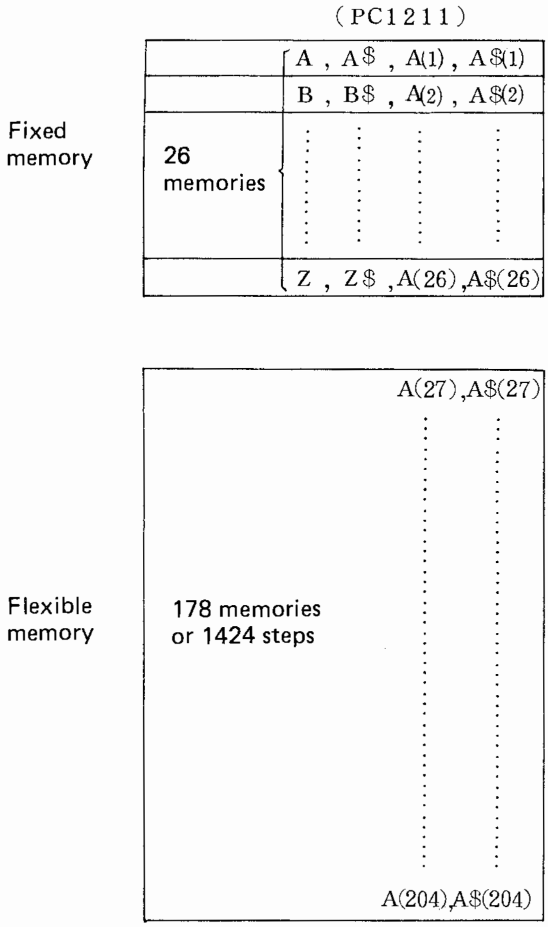
(ii) Recall memories D and E and check their contents through the display.

[Solution]

```
(1)  (i)    A=3 ENTER  }
            B=4 ENTER  }      or,    A=3,B=4 ENTER
     (ii)   C=√(A*A+B*B) ENTER
     (iii)  A ENTER                                        3.
            B ENTER                                        4.
            C ENTER                                        5.
     (i)    A=A+1 ENTER  }
            B=B+1 ENTER  }    or,    A=A+1,B=B+1 ENTER
     (ii)   C=√(A*A+B*B) ENTER
     (iii)  A ENTER                                        4.
            B ENTER                                        5.
            C ENTER                               6.403124237
     (i)    D$="SHARP"   ENTER
            E$="PC1211"  ENTER
     (ii)   D$   ENTER                SHARP
            E$   ENTER                PC1211
```

[Notes]

○ Memory configuration



(Fig. 1)

What is generally called a memory can be divided into the following categories. (In an upper class of machine the data memory is usually called a "variable".)

- Memory
  - Data memory (Variable)
    - Numerical memory (Numerical variable)
    - Character memory (Character variable)
  - Program memory

Memory configuration of this pocket computer consists of the fixed memory and the flexible memory.

Fixed memory .... Data storage consisting of 26 memories, A through Z.

Flexible memory .... This memory can be used for both program and data memory. For programming, 8 steps of programs equals one data memory.
For instance, if a 16 steps of program is stored, it will reserve 202 data memories, since 16 steps consists of 2 data memories. (204 − 2 = 202)

○ Storing numerical number to the numerical variable

A = 3 ENTER

A = A + 1 ENTER

As shown above, entry must be made in sequence of "numerical variable = formula ENTER ". (Numerical variable = formula is called a substitution.)

○ Storing of character to the numerical variable
The numerical variable is represented by affixing a $ (dollar mark) after the variable and up to a maximum of 7 characters can be stored in the numerical variable.

Character variable = "character" ENTER , or

Character variable = character variable ENTER (Such an operation as "character variable" = "character" is also called a substitution.)

In the BASIC programming, the character variable is called a string and "A$" is also called as A string or simply A dollar.

○ Other way of variable designation
As shown in Fig. 1, such as B, B$, C, C$ can also be designated as A (2), A$ (2), A (3) and A$ (3). If it were to exceed 26 memory capacity, it has to be designated like A (27), A$ (27), A (28), A$ (28), using this method.

○ Cancelling all data memory area
Designation of "CLEAR ENTER " will clear all the data memory contents.

Section 4 Reservable key operation

[Problem]

(1) Reserve SIN, COS and TAN to A, S, and D keys and compute sin 30°, cos 30° and tan 30°.

(2) Reserve "×0.06 ÷ 365" to the Z key and perform the following computation:

365 × 0.06 ÷ 365 =

100 × 0.06 ÷ 365 =

[Solution]

(1) (i) Set to the RESERVE mode (depression of the MODE key)

```
SHFT  A  SIN  ENTER
SHFT  S  COS  ENTER
SHFT  D  TAN  ENTER
```

(ii) Set to the RUN mode (depression of the MODE key)

```
SHFT  A  30  ENTER                        0.5
SHFT  S  30  ENTER            8.660254038E-01
SHFT  D  30  ENTER            5.773502692E-01
```

(2) (i) Set to the RESERVE mode (depression of the MODE key)

```
SHFT  Z  *.06/365  ENTER
```

(ii) Set to the RUN mode (depression of the MODE key)

```
365  SHFT  Z  ENTER                      0.06
100  SHFT  Z  ENTER           1.643835616E-02
```

[Notes]

Any constant data or instruction can be reserved through 18 reservable keys on the keyboard (A, S, D · · · · · , =, Z, · · · · · , SPC).

How to make reservation

(i) Set to the RESERVE mode.

(ii) $\text{SHFT} \quad \underbrace{\text{A}}_{\text{Reservation key}} \quad \underbrace{\text{SIN} \qquad\qquad\qquad\qquad}_{\text{Key operation to be reserved}} \text{ENTER}$

Reserving frequently used instruction and key operation in any desired reservable key will allow you simple operation (SHFT A, SHFT S, etc.). Your pocket computer may become custom tailored when you have reserved often used instructions in reservable keys, and it will give you convenience when you have noted the reservable key assignment on your template.

Ex.

$\boxed{A}$	$\boxed{S}$	$\boxed{D}$	………
sin	cos	tan	
$\boxed{Z}$	$\boxed{X}$	$\boxed{C}$	

Your template

NOTE: Reservation up to a total of 48 steps is possible with 18 reservable keys.

[Exercise]

1. $123-456+789=$

2. $5\times(-3)\div4=$

3. $\frac{4}{25}+13\times0.2=$

4. $2+3\div6=$

5. $\frac{2+3}{6}=$

6. $\frac{2+3}{6+4}=$

7. $\frac{2+3}{6+4}\times\frac{-3+2}{6-4}=$

8. $\frac{5-\sqrt{5\times5-4\times2\times3}}{2\times2}=$

9. $\sin30^\circ+\cos30^\circ=$

10. $\sin(30^\circ+45^\circ)=$

11. $\cos\frac{20^\circ+40^\circ}{2}=$

12. $\sin^2 30^\circ+\cos^2 30^\circ=$

13. $\sin(-30°) =$

☆ 14. $2^{-2} + 64^{2/3} =$

15. $\log 3 + \ln 3 =$

16. If the consumer price index rises by 7% each year, how many times the consumer price index will be after ten years?

☆ 17. Store −2, 1 and 3 to memories A, B and C, and compute the following expression.

$$\frac{-B + \sqrt{B^2 - 4AC}}{2A}$$

$$\frac{-B - \sqrt{B^2 - 4AC}}{2A}$$

☆ 18. Assume A = 9, B = −10, C = 1 in the equation $Y = Ax^2 + Bx + C$, and obtain the value of Y,

when $x = 0$;

when $x = -2$;

when $x = 3$;

when $x = 0.5$;

Chapter 2 Programmed operation (1)

Section 1 What is the programmed operation?

There may be repetition of a same routine in different calculation from time to time. Let's take the Pythagorean theorem for instance.
The following routine procedure is observed to obtain the length of a pypotenuse of a triangle of which edges composing the right angle are 3m and 4m.

√‾ (3 * 3 + 4 * 4) $\boxed{\text{ENTER}}$

What if the edges composing the right angle are 4m and 5m?

√‾ (4 * 4 + 5 * 5) $\boxed{\text{ENTER}}$

To obtain the hypotenuse, the same process routine is executed, except the parameters which may vary. Fixed routine like this may be stored permanently in the computer for the convenience of eliminating the repetition of the same routine. The machine can follow the routine process as if performed by human being.
To store a certain procedure in the computer is called a programming or called "to make a program". In this chapter, we will discuss the programming.

Section 2 Linear sequence programming (INPUT, LET, PRINT, NEW, RUN, LIST, MEM)

[Problem]

(1) Prepare a program which can produce area by the input of a radius.

(2) Prepare a program which can produce length of a hypotenuse by the input of the length of two edges composing the right angle of a right triangle.

[Solution]

(1)
```
10    INPUT   R
20    LET    S=R*R*π
30    PRINT   S
40    END
```

(2)
```
10    INPUT   A , B
20    LET    C=√‾ (A*A+B*B)
30    PRINT   C
40    END
```

[Notes]

○ Set the machine to the PRO mode before preparing a program. (Depression of the ENTER key)

○ With Input of "NEW ENTER ", the program and data previously stored are all cleared, except the reserved memory. (The memory memorizing the reservable key (Section 4, Chapter 1) is called a reserve memory.)

○ Line

- With the BASIC, program is composed of several lines, and statement with various significances is contained in a single line.
- Line number is accompanied with integer of 1 through 999.
- To make up program in a line, the following procedure is required.

$\underbrace{\qquad\quad}_{\text{Line number}}$ $\underbrace{\qquad\qquad\qquad}_{\text{Statement}}$ ENTER (With ENTER , termination of a line is defined.)

- As the program takes place from younger line number, line numbers must be written in order starting from low order. However, it is the most common practice to furnish line numbers in an interval of ten considering occasional insertion of an additional line numbers. (i.e. 10, 20, 30 · · · · · etc.)

○ Statement

- Instruction or process formula having a specific significance is called a statement.
- A line may consist of a single statement or several statements. A colon " : " has to be inserted between statement when more than two statements are used.

○ INPUT instruction

INPUT instruction is used to designate input of numerical value or character. The numerical variable or character variable has to be written after INPUT.

INPUT A (a numerical value is assigned to variable A.)

INPUT C$ (a character is assigned to variable C$.)

If multiple numbers of numerical values or characters are to be input in succession, the following type of statement may be preferable.

INPUT A, B

INPUT C$, D$

○ LET instruction

LET instruction is used to prepare a substitution. The substitution is to be written after LET like "LET S = R ∗ R ∗ π," but LET may be omitted usually.

○ PRINT instruction

When a variable is designated after PRINT, the contents of the variable is shown on the display.

PRINT A (displays the contents of the numerical variable on right hand 16 columns of the display. However, it will be displayed in exponential expression when exceeds 16 columns capacity.)

PRINT A$ (displays the contents of the character variable starting from the left hand side of the 24 display columns.)

Character or formula may be written instead of variable.

○ Correction, insertion, and deletion of program

To make correction, insertion or deletion in a program line shown on the display, either the ◄ or ► key is used along with DEL or INS key as we have learned in the chapter of manual operation.

To delete a line, "line number ENTER " should be entered.

Line insertion can be done in the following manner.

Ex: When "S = R $*$ R $*$ π" is omitted in the Problem (1).

```
10    INPUT  R
                  < "S = R * R * π"  has to be inserted here.
20    PRINT  S
30    END
```

In case a program line is to be inserted between the line number 10 and 20, pick up any number from 11 to 19 for program implementation.

For example, "15 LET S = R $*$ R $*$ π ENTER " may be entered. With this the program is corrected in the following manner. (This may explain why the line numbers are assigned by the leap of ten.)

```
10    INPUT  R
15    LET  S = R * R * π
20    PRINT  S
30    END
```

NOTE: The program has not necessarily be written in order of low numbers, since the computer will rearrange them automatically by itself.

NOTE: To change only a line number, use the ◄ key until the respective line number flickers, then change the line number.

○ Program listing

Procedure to check program contents

(i) ↓ key

Each time this key is depressed, contents of a line currently on the display is replaced with a contents of a next line. Successive depression of this key causes continuous replacement.

(ii) ↑ key

Each time this key is depressed, contents of a line currently on the display is replaced with a contents of a preceding line. Successive depression of this key causes continuous replacement.

(iii) LIST instruction

Entry of "LIST ENTER " will bring the youngest line on the display.

Entry of "LIST $\underbrace{\qquad\qquad\qquad}_{\text{Line number}}$ ENTER " will bring the indicated line on the display.

○ How to count up program steps

- ● 80 key operation including the ENTER key is possible in a single line.
- ● One instruction consists of one step. Although "INPUT" requires five key strokes, it is counted up as one step in the computer.
- ● Line number is counted as two steps.
- ● The ENTER key is counted as one step.

So, there are 25 steps in the Problem (1) and 32 steps in the Problem (2).

○ Checking unused program capacity

MEM ENTER

Depression of this key displays unused steps in the flexible memory area and its equivalent to data memory is also indicated.
When this key is operated after NEW,

```
1424STEPS    178MEMORIES
```

After the loading of the program in the Problem (1),

```
1399STEPS    174MEMORIES
```

○ Program execution

(i) Before the execution of a program, set the machine to the RUN mode.

(ii) Then, start execution in the following sequence.

RUN ENTER (starts execution from the youngest line number.)

RUN $\underbrace{\qquad\qquad}_{\text{Line number}}$ ENTER (starts execution from the designated line number.)

(iii) As the machine comes to halt with " ? " displayed when the machine encounters an INPUT statement, the execution should be resumed after entering ENTER .

☆ Section 3 Various display methods (PRINT, USING, PAUSE)

[Problem]

(1) With reference to the Problem (1) in the last section,

Make display of "RADIUS␣R =" when the radius is to be entered.

Bring to indicate the radius on the left half of the display and the area on the right half when the area is output.

(2) With reference to the Problem (2) in the last section,

Make display of "A =   B =   " when input is to be made for two edges.

Display the results in a manner of "A = ○○○   B = ○○○   C = ○○○.○○" when the hypotenuse is output.

[One example of Solution]

(1)
```
10  INPUT "RADIUS ␣R=", A
20  S=A*A*π
30  PRINT  A, S
40  END
```

(2)
```
10  INPUT  "A="; A, "B="; B
20  C=√ (A*A+B*B)
30  PRINT  "A="; USING "###"; A; "␣B="; USING "###"; B;
    "␣C="; USING "###.##"; C
40  END
```

Note: ␣ stands for a SPC key in this text.

[Notes]

(1) Method to display a message at the time of input

(i) INPUT "character", variable

Ex: INPUT "A=", A

(ii) INPUT "character"; variable

Ex: INPUT "B=", B

With (i), entry of numeral causes the message cancelled, but, with (ii), the input data is displayed following to the message, without cancelling the message.

(2) Various use of PRINT instruction

(i) PRINT formula

NOTE: The expression "formula" applies to the expression composed of numerical variable, numericals, numerical expression and numerical variable.
i.e. (A) (5) (5*6) (2*A + 3)

Ex:
PRINT A (The contents of A is displayed on right hand 16 columns of the display. However, it will be displayed in exponential expression when exceeds 10 digits.)

(ii) PRINT character variable

Ex: PRINT A$ (Contents of A$ is displayed from the left hand side of the display.)

(iii) PRINT "character"

Ex: PRINT "SHARP␣PC-1211" (Character is displayed from the left hand side of the display.)

(iv) $\text{PRINT} \left\{\begin{matrix}\text{formula}\\ \text{"character"}\\ \text{character variable}\end{matrix}\right\} , \left\{\begin{matrix}\text{formula}\\ \text{"character"}\\ \text{character variable}\end{matrix}\right\}$

24 columns of the display is divided into 12 columns each, and display of data is right justified and characters are left justified within these divisions.

(v) $\text{PRINT} \left\{\begin{matrix}\text{formula}\\ \text{"character"}\\ \text{character variable}\end{matrix}\right\} ; \left\{\begin{matrix}\text{numerical variable}\\ \text{"character"}\\ \text{character variable}\end{matrix}\right\} ; \cdots ; \left\{\begin{matrix}\text{numerical variable}\\ \text{"character"}\\ \text{character variable}\end{matrix}\right\}$

The contents after the PRINT instruction is displayed in left justified manner within 24 columns.

(3) USING instruction

With the USING instruction, display format of the numericals is designated.

General form of the USING instruction

USING "###" (displays 3 digits of integer including sign.)

USING "###." (displays 3 digits of integer including sign and decimal point.)

USING "###.##" (displays 3 digits of integer and 2 digits below the decimal point, including sign and decimal point.)

USING "##.##∧" (numbers down to 2 digits below the decimal point is displayed in exponential expression.)
(2 digits of integer are displayed for mantissa including sign, and 4 digits of exponent (E-OO), including sign, is automatically taken up in this case.)

(i) Simple use of USING statement

```
10    USING "###.##"
20    PRINT A; B; C
```

Display example: When A = 12.345, B = –34.567, C = 5,

```
 12.34 - 34.56   5.00
```

Space 2 spaces

(ii) To use as a PRINT USING statement, containing a USING instruction within the PRINT statement

```
10    PRINT "A="; USING "###"; A; "_B="; USING "###"; B; "_C=";
      USING "###.###"; C
```

Display example: When A = 4, B = 10, C = 10.7703,

```
 A=   4  B=  10  C=  10.770
```

Spaces

(iii) Release of USING designation

After designating the display format with the USING instruction, format designation in the successive PRINT instruction will be controlled by this designation. Thereby, it has to be released using the above statement when it is not required.

(4) PAUSE instruction

On the contrary to the PRINT instruction that stops the program execution, the PAUSE instruction acts to hold the display for a certain period (about 0.85 second) and resumes the program execution again. Other details are the same as the PRINT instruction.

[Exercise]

1. Prepare the program which will produce the length of circumference after the input of a radius of a circle.

   $\ell = 2\pi r$

2. Prepare the program which will calculate interest after the input of principal and period. Whereas, the rate of annual interest is 6% and truncation must be made below the dollar point.

   $$(\text{Interest} = \text{principal} \times \text{period} \times \frac{0.06}{365})$$

☆ 3. In the above problem, prepare the program which will calculate interest after rounding the figure below the dollar point.

☆ 4. Prepare the program which will produce the area after the input of the length of three edges of a triangle.

   Heron's law: $\text{Area} = \sqrt{s(s-a)(s-b)(s-c)}$

   Where, $s = \frac{a+b+c}{2}$

5. Prepare the program which will generate the following display layout, using the PAUSE instruction.

```
SHARP␣PC-1211␣HAS
```

```
1424␣STEPS␣26␣MEMORIES
```

Chapter 3 Programmed operation (2)

In this chapter more sophisticated programming technique will be discussed. Successful comprehension of the Section 1 will give you almost complete understanding of this pocket computer programming.

☆☆Section 1 Decision and Jumping (IF THEN, GOTO)

[Problem]

(1) Prepare the program which will summarize the total of admission charges after the input of the numbers of audience entering the theater.
Where, the ordinary admission fee costs $10 each up to 10 persons, but group discount of $8 is applicable for a group consisting of more than 10 persons.
Display the numbers of audience on the left side of the display and the summary of admission charges on the right hand side.

```
Audience     8      $ 80
```

[Solution]

```
10    INPUT  "AUDIENCE="; A
20    IF A>=10    THEN 60
30    B=10*A
40    PRINT "AUDIENCE"; USING "###"; A; "␣␣ $"; USING "#####"
      ;B
50    END
60    B=8*A : GOTO 40
```

[Another Solution]

```
10    INPUT "AUDIENCE ="; A
20    IF  A>=10    LET B=8*A : GOTO 40
30    B=10*A
40    PRINT "AUDIENCE"; USING "###"; A; " ␣␣ $"; USING "#####"
      ;B
50    END
```

[Notes]

○ IF instruction

Various decision is made with the IF instruction.

```
IF  A>=10
```

The above statement defines to execute the program described after IF (when the IF statement is not satisfied), the prograrn following this line will be executed.
Usually, the IF statement is used in the following forms:

IF	=	XXX	Interrogates if equal
IF	>	XXX	Interrogates if larger than
IF	>=	XXX	Interrogates if equal or larger than
IF	<	XXX	Interrogates if smaller than
IF	<=	XXX	Interrogates if equal or smaller than
IF	<>	XXX	Interrogates if not equal

Such instruction as "THEN 60" is usually used after the IF statement, so as to jump to the line number 60.
Sometimes, it will be much easier to understand if a conditional program is placed after the IF statement as shown in the Another Solution. (The substitution immediately after the IF statement must begin with LET without fail.)

○ GOTO instruction

Usually, the program is executed in order of low line numbers, but execution of this GOTO instruction will cause to jump the program to the line number indicated after the GOTO instruction.

Ex.

GOTO 40	(Jumps to the line number 40.)
GOTO 5*8	(Jumps to the line number 40.)
GOTO A	(Jumps to the line indicated by A. If A = 40, for instance, the program then jumps to the line number 40.)
GOTO "A"	(Jumps to the line number which is denoted with the Label A.)

Ex:

```
20  GOTO  "A"
30  B = 1000*A
40  "A" PRINT "AUDIENCE";···
```

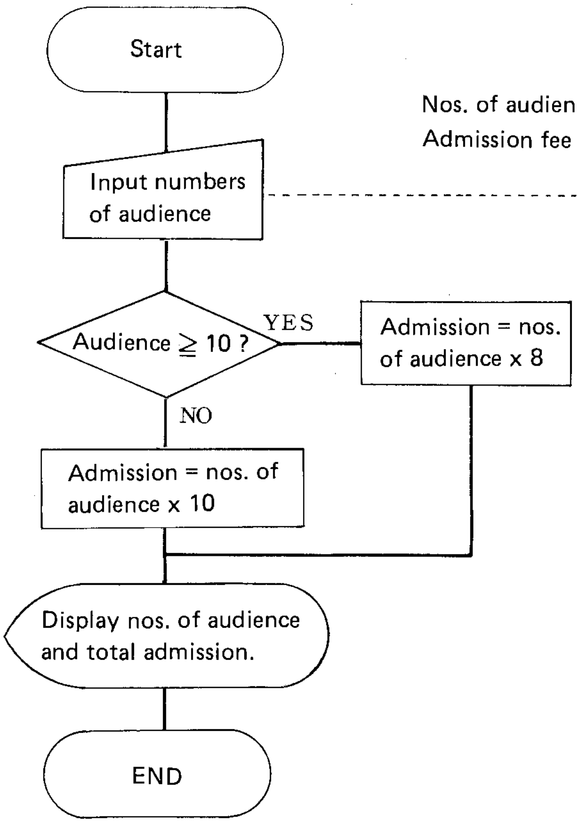
This causes to jump from the line number 20 to 40. Jump like this is called a label jump, which can be designated with character up to 7 characters.

○ How to make up a program

It is not an easy practice to complete program simply looking over the problem. Ordinally, outline of computation is embodied into a drawing to check programming flow. This drawing of flow is called a flowchart. Perfect completion. Therefore, it is the fastest way for better programming to get familiar with flowcharting in the first place.

With the flowchart in completion, it is the time to decide to which variable the data is to be assigned. Then, program coding is carried out using BASIC instructions.

Flowchart of the Problem in this section:


Start
Input numbers of audience
Audience ≧ 10 ?
YES
Admission = nos. of audience x 8
NO
Admission = nos. of audience x 10
Display nos. of audience and total admission.
END

Nos. of audience. . . . A
Admission fee · · · · B

```
10   INPUT A

20   IF A>=10 LET
     B=A*8
     : GOTO 40

30   B=A*,10

40   PRINT A, B

50   END
```

Although various notations are used in the example, it can be thought as a kind of accessories. It is of a primary importance to prepare the flowchart like the above before writing the program.

☆Section 2 Repeated operation (FOR, TO, STEP, NEXT)

[Problem]

Display the results from sin 0° through sin 90° in an interval of 2°, with the angle displayed on the left hand side and the value of sine on the right hand side of the display. (Use the PAUSE instruction.)

[Solution]

```
10   FOR A=0 TO 90 STEP 2
20   B=SIN A
30   PAUSE USING "###" ; A ; USING "####.#####" ; B
40   NEXT A
50   END
```

[Notes]

○ General form of FOR NEXT

For numerical variable = initial value TO final value STEP interval

∫

NEXT numerical variable

The numerical variable will increment by the given interval starting from the initial value and the programs described between FOR and NEXT are repeated until the numerical variable reaches the given final value.

○ The FOR instruction must be used in pair with the NEXT instruction, and that the numerical variable after FOR must be identical with the numerical variable after NEXT.

○ "STEP 1" may be omitted when the interval is 1.

○ Up to 4 stages of programs can be contained in a FOR NEXT statement.

○ Negative number may be used in defining the interval. If the following statement is filled up in the line number 10 of the Problem, it will display the results from sin 90° to sin 0° in an interval of 2°.

```
10  FOR A=90 TO 0 STEP-2
```

☆Section 3 Definable key operation

[Problem]

Prepare programs which produce the area of a circle and the circumference of a circle, and make them executable with the use of the A and Z keys, respectively.

[Solution]

```
10  "A" INPUT "R=" ; R
20      S=R*R*π
30      PRINT "S=" , S
40      END
50  "Z" INPUT "R=" ; R
60      L=2*R*π
70      PRINT "L=" , L
80      END
```

[Notes]

Up to now, the machine is set to RUN mode and "RUN ENTER " is entered to begin execution of program. However, there is another method to start instant execution of program by entering " SHFT A" and such after setting the machine to the DEF mode. If label is assigned to one of 18 reservable keys (A, S, · · · =, Z, · · · , SPC, etc.), it is possible to start the program by the depression of the key corresponding to the label following to the entry of " SHFT " in the DEF mode.

When numbers of programs are in the memory, the desired program can be chosen automatically with this method. It will be much convenient in operating the computer when kinds of programs and names are inscribed on the template.

In this Problem, the following notations shall be given on the template.

A S

Area of circle

Z X

Circumference

Template

- ○ In case the Problem is to be operated under the RUN mode, it is necessary to make selection of program lines. These kinds of decision making is very much troublesome when numbers of programs are to be used in the operation of a single task.

```
RUN    ENTER
RUN  50  ENTER
```

☆☆Section 4 Indirect memory designation (one dimensional arrrangement)

[Problem]

There are 10 kinds of merchandizes. Prepare the program which can summarize total numbers of each merchandize with code output. Whereas, input is merchandize code and quantity.

[Solution]

```
 10  "A" INPUT "NO=" ; Y , "QTY=" ; Z
 20      A(Y)=A(Y)+Z
 30      GOTO "A"
 40 "“Z" FOR Y=1 TO 10
 50      PRINT "NO=" ; USING "###" ; Y ; USING "######." ;A(Y)
 60      NEXT Y
 70      END
 80  "=" CLEAR
 90      PRINT "ALL␣CLEAR"
100      END
```

[Operational example]

Template

A   =
Input   Clear
Z
Total

Set to the DEF mode

SHFT A .... Input process

Repetition
- Merchandize code ENTER
- Quantity ENTER

SHFT Z .... Output of each summary.

Repetition
- ENTER ...... Displays code and quantity

SHFT = ... Clears all

[Exercise]

☆ 1. Prepare the program which can produce the root of the equation "$Ax^2 + Bx + C = 0$". Whereas, the parameters A, B, and C are the input ($A \neq 0$).

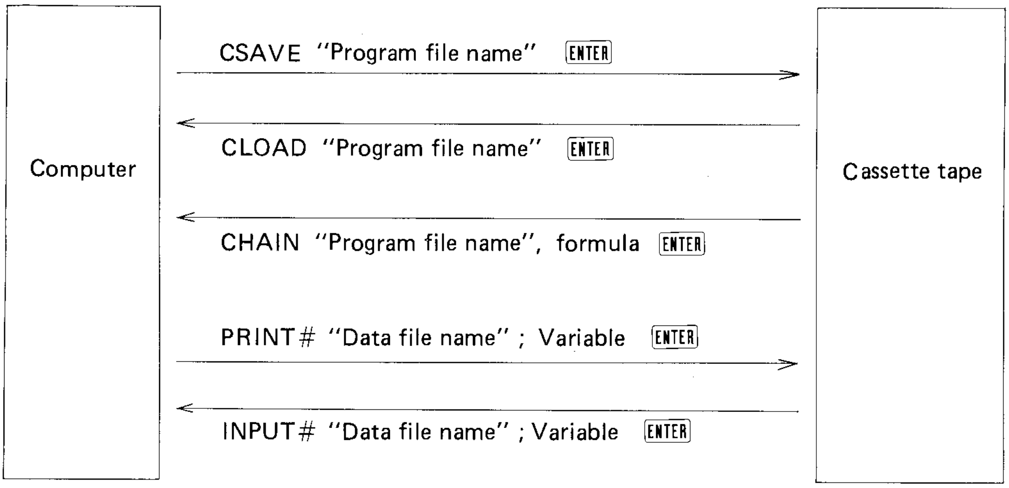
However, { display two roots on the first and second halves of the display, when $B^2 - 4AC = 0$, and display "NO ANSWER", when $B^2 - 4AC < 0$.

2. Prepare the program which can solve "m + (m + 1) + (m + 2) + · · · + n" after substituting m and n. m and n are integer and m < n.

☆ 3. Prepare the program which can solve "n!" after the input of n. (n! is a factorial of n. i.e. 1 x 2 x 3 · · · x n).

4. Prepare the program which can produce the value of sine from sin 0° through sin 90° in an interval of 10°.

Chapter 4 External memory (cassette tape) (CSAVE, CLOAD, CLOAD?, CHAIN, PRINT#, INPUT#)


Computer
CSAVE "Program file name" ENTER
CLOAD "Program file name" ENTER
CHAIN "Program file name", formula ENTER
PRINT# "Data file name" ; Variable ENTER
INPUT# "Data file name" ; Variable ENTER
Cassette tape

(1) CSAVE "program file name" ENTER
Entire programs in the computer are stored in the cassette tape with their program file names. Up to 7 characters of file name is acceptable.

(2) CLOAD "program file name" ENTER
With this statement, the program with the given program name is automatically searched in the cassette tape to be transferred to the computer.

(3) CLOAD? "program file name" ENTER
With this statement, the program in the computer is collated with the program in the cassette tape on the basis of the given program file name.
Train yourself to observe this operation whenever CSAVE or CLOAD is executed. (This is mandatory to safely implement your precious program to the computer.)

(4) CHAIN "program file name", formula ENTER
When this statement is encountered in the middle of the program, the program in the cassette tape with the given program file name is transferred to the computer, then program execution takes place automatically from the line specified by the formula. Omittance of the line number will cause to start from the youngest line number.
This technique is very useful to execute a long program that can be contained in the computer at one time.

(5) PRINT# "data file name"; Variable ENTER
With this statement, all the data after the data memory specified by the variable in the statement are transferred to the cassette tape with its data file name.

(6) INPUT# "data file name"; variable ENTER

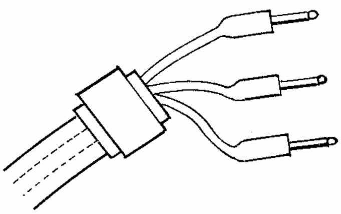
Data in the cassette tape with the data file name specified in the statement is transferred to the computer after the variable indicated in the statement.

NOTE: For the transfer and collation of the reserved program, steps (1) through (3) are executed in the RESERVE mode.

(Handling of the cassette tape)

Use the CE-121 to connect this pocket computer with the cassette tape unit.

○ How to make connection



Gray plug (large) —— Connect to the EAR PHONE jack (or the MONITOR jack) of the cassette tape recorder.

Red plug (large) —— Connect to the MIC jack of the cassette tape recorder.

Black plug (small) —— Connect to the REMOTE jack of the cassette tape recorder.

From CE-121

○ ON/OFF of the tape recorder is automatically controlled by the computer when the plug is connected with the REMOTE jack. Manual operation is necessary when the REMOTE jack is not used.

○ Turn the MIX knob fully to the SOURCE side when a tape recorder with mixing function is to be used.

○ CSAVE
PRINT#
} For these operations, the tape recorder must be kept in recording condition by manual control before the operation.

○ CLOAD
CLOAD?
CHAIN
INPUT#
} For these operations, the tape recorder has to be kept in playing condition by manual control before the operation. The volume control switch, however, should be set to the maximum.

○ At the start up of the tape recorder operation, continuous sound is generated for a period of 4 to 5 seconds at the first and then comes intermitted sound. Leading edge is being provided on the tape while the continuous sound is on, and program or data transmission is taking place while the intermitted sound is on.

○ Be sure to thread and bring the leading edge over the magnetic head in satisfactory manner before the operation of the tape.

○ Before putting the tape into the storage, be sure to take up entire tape in either side of the cassette so as to protect the tape from deformation.

○ It is recommended to have extra tape recorded in order to save precious data and programs.

○ Get accustomed to arrange programs and data in tapes by taking notes of file names, etc. (Loosing the file name causes a good deal of trouble, because you can not afford to take out proper program or data.)

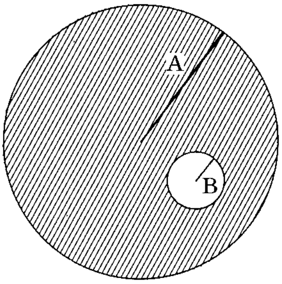
○ Use one of intermediate conversion plugs sold at the shop, if the size of the jack does not conform to the jack on the tape recorder.

○ For those which have two way tape speeds like in a micro cassette recorder unit, set the speed to the highest.

Chapter 5 Others

☆☆Section 1 Subroutine

[Problem]



With respect to two circles in the figure, prepare the program which can produce the area of the hatched portion after the input of radii of two circles.

[Solution]

```
10 INPUT "A=";A,"B=";B
20 R=A:GOSUB 70
30 C=S:R=B:GOSUB 70
40 D=S
50 PRINT C-D
60 END
70 S=R*R*π
80 RETURN
```

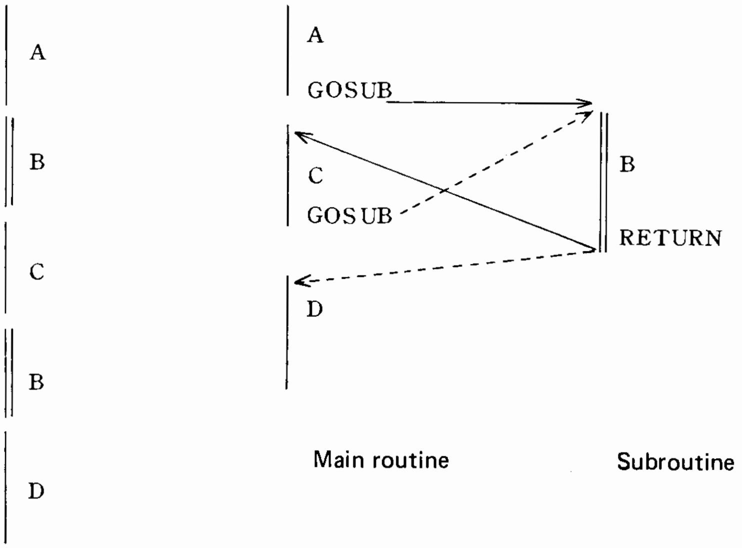
A:	radius of larger circle
B:	radius of smaller circle
C:	area of larger circle
D:	area of smaller circle

Subroutines

R:	radius of circle
S:	area of circle

[Notes]

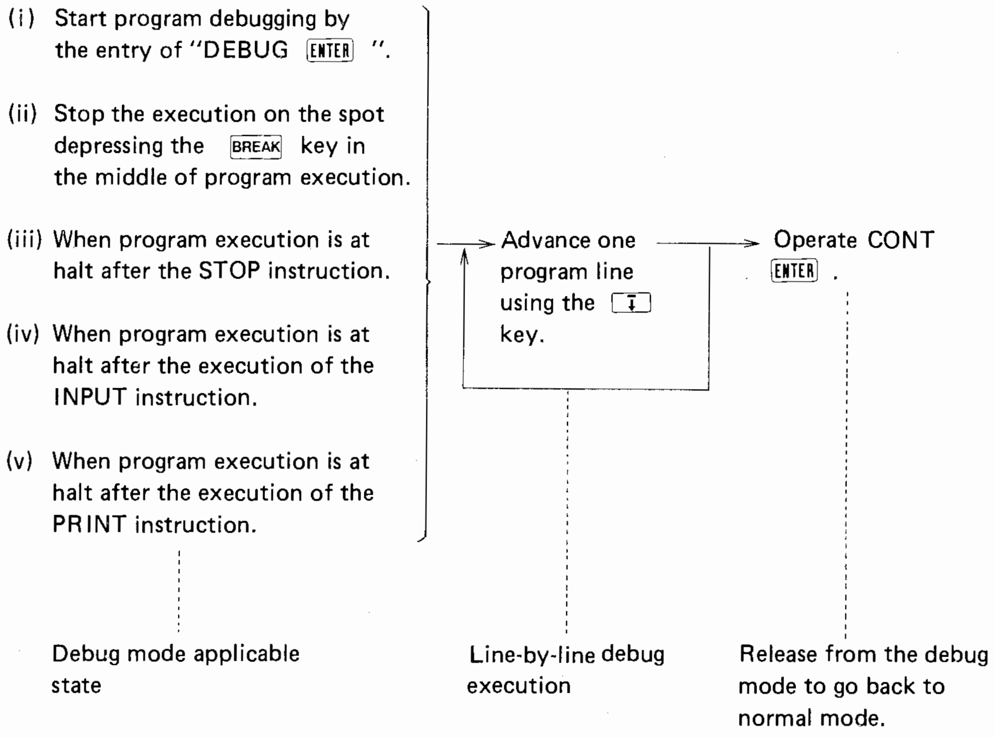
○ When there are needs of repeating the same procedure, it will become convenient if that procedure is handled as a subroutine. Only a fraction of such case is represented with this Problem. Use of a subroutine helps to make the program shorter when there are frequent use of common procedures or when program in common procedure is too long.



When there is a common program portion (B) as in the above illustration in the left, implementation of the subroutine changes the program flow as in the right of the illustration. To jump to the subroutine, the line number where the subroutine is stored should be written after GOSUB instruction. With the RETURN instruction, the program returns to the main routine and restores the execution from line number next to the one previously executed. (The original program where GOSUB is written is called a main routine against the subroutine.)

☆☆☆Section 2 Debugging and others

Sometimes, you will wish to trace the program line by line when result is not properly obtained due to indefinite running of program and so on. So, we will discuss the way to trace the program line by line. (With this debugging technique, you will be able to cope with the program trouble cold hearted, even if the program is not running in proper manner.)



(i) Start program debugging by the entry of "DEBUG ENTER ".

(ii) Stop the execution on the spot depressing the BREAK key in the middle of program execution.

(iii) When program execution is at halt after the STOP instruction.

(iv) When program execution is at halt after the execution of the INPUT instruction.

(v) When program execution is at halt after the execution of the PRINT instruction.

Advance one program line using the ↓ key.

Operate CONT ENTER .

Debug mode applicable state

Line-by-line debug execution

Release from the debug mode to go back to normal mode.

NOTE: CA key

When the CA key is depressed during the manual operation mode or the BREAK state (which is caused after the STOP instruction, depression of the BREAK key or during the debugging), it clears all the display contents (but displaying " > " symbol only) and reset the internal state of entire computer, the same condition as at power up time. (This key functions as a power ON key, BREAK key and CA key, depending on the situation in the computer.

Other instructions

○ BEEP instruction

General form	BEEP formula	Makes sound as many times as specified by the next formula
Example:	BEEP 5	Beeps 5 times.

○ STOP instruction

When this instruction is executed, it makes the program halted temporarily. With the entry of "CONT ENTER ", it resumes the normal operation. This instruction is written in the line where the program has to be stopped for debugging purpose.

○ REM instruction

It stands for REMARK and has nothing to do with program execution (statements after REM are ignored by the computer). It is used to indicate notation for easier recognition of programming, when a long program is in use.

Ex:

```
   10     REM    DATE   CALCULATION
    :
    :
    :
  200     REM    SUBROUTINE
    :
    :
```

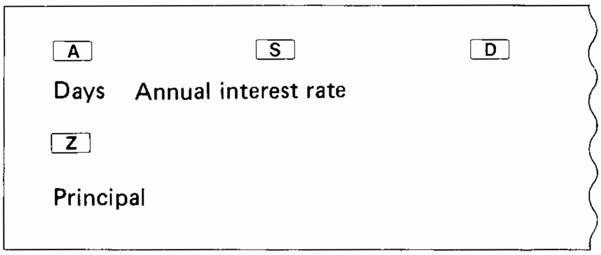
○ AREAD instruction

```
100  "A"     AREAD  A
110          END
```

When this instruction is written to the next of the definition label and used in the DEF mode, the contents of the display before the program execution is INPUT to the variable as the program is initiated with " [SHFT] A" command. Use of the INPUT instruction requires either one of the following procedure, requiring many key operations.

In the DEF mode	In the RUN mode
[SHFT] A	RUN 100 [ENTER]
Numerical [ENTER]	Numerical [ENTER]

It will improve great deal of operationability when the AREAD instruction is put into use after assigning the A key to the period, S key to the annual interest and the Z key to the function calculating the interest after the input of the principal.



```
10    "A"  AREAD  A
20         END
```

A:	Days
B:	Annual interest rate
C:	Principal
D:	Interest

```
30  "S" AREAD B
40      END
50  "Z" AREAD C
60      D=A*B*C/36500
70      PRINT D
80      END
```

(Operational guide) Set to the DEF mode.

Days SHFT A

Annual interest rate SHFT S

Principal SHFT Z

Chapter 6 Solutions and Answers to Excercises (Chapter 1–3) are given below:

Chapter 1

1. 456
2. −3.75
3. 2.76
4. 2.5
5. 8.333 333 333 E−01
6. 0.5
7. −0.25
8. 1
9. 1.366025404
10. 9.659258263 E−01
11. 8.660254038 E−01
12. 1
13. −0.5
14. 16.25
15. 1.575733543
16. 1.967151357
17. Set to RUN mode

    A = −2 ENTER B = 1 ENTER C = 3 ENTER

    Set to RESERVE mode

    SHFT A √ (B*B − 4*A*C) ENTER

    Set to RUN mode

    (−B + SHFT A) / (2*A) ENTER ······· −1

    (−B − SHFT A) / (2*A) ENTER ······· 1.5

18. A = 9 ENTER

    B = −10 ENTER

    C = 1 ENTER

    Set to RESERVE mode

    SHFT B A*X*X + B*X + C ENTER

    Set to RUN mode

    X = 0 ENTER

    SHFT B ENTER ······· 1

    X = −2 ENTER

    SHFT B ENTER ······· 57

```
X = 3  ENTER
SHFT  B  ENTER    ········52
X = 0.5  ENTER
SHFT  B  ENTER    ········−1.75
```

Chapter 2

```
1.    10  INPUT  R
      20  L = 2*π*R
      30  PRINT  L
      40  END
2.    10  INPUT  A, B
      20  C = INT  (A*B*0.06 / 365)
      30  PRINT  C
      40  END
3.    10  INPUT  A, B
      20  C = INT  (A*B*0.06 / 365 + 0.5)
      30  PRINT  C
      40  END
4.    10  INPUT  A, B, C
      20  S = (A + B + C) / 2
      30  D = √ (S*(S − A)*(S − B)*(S − C))
      40  PRINT  D
      50  END
5.    10  PAUSE "SHARP   PC   1211   HAS"
      20  PAUSE "1424   STEPS   26   MEMORIES"
      30  GOTO  10
      40  END
```

Chapter 3

1.

```
10  INPUT  A, B, C
20  D = B*B - 4*A*C
30  IF  D<0          PRINT  "NO   ANSWER" : END
30  PRINT  (-B + √ D) / (2*A) ,  (-B - √ D) / (2*A)
40  END
```

2.

```
10  INPUT  "M=" ; M ,  "N=" ; N
20  S = 0
30  FOR  A = M   TO   N
40  S = S + A
50  NEXT  A
60  PRINT  A
70  END
```

3.

```
10  INPUT  N
20  S = 1
30  FOR  A = 1  TO  N
40  S = S*A
50  NEXT  A
60  PRINT  S
70  END
```

4.

```
10  FOR  A = 0  TO  90  STEP  10
20  B = SIN  A
30  PAUSE   A, B
40  NEXT  A
50  END
```

Supplement—1

Instruction	Abbreviation	Usability in RUN and DEF modes	Programmability	Example	Note
=		○	○	A = 10	
+		○	○	A = B + C	
−		○	○	A = B − C	
*		○	○	A = B * C	
/		○	○	A = B / C	
∧		○	○	A = B ∧ C	
( )		○	○	A = (B + C) / D	
=		○	○	IF A = B	
>		○	○	IF A > B	
>=		○	○	IF A >= B	
<		○	○	IF A < B	
<=		○	○	IF A <= B	
<>		○	○	IF A <> B	≠ (Not equal)
SIN	SI.	○	○	A = SIN B	
COS		○	○	A = COS B	
TAN	TA.	○	○	A = TAN B	
ASN	AS.	○	○	A = ASN B	
ACS	AC.	○	○	A = ACS B	
ATN	AT.	○	○	A = ATN B	
LOG	LO.	○	○	A = LOG B	Common logarithm
LN		○	○	A = LN B	Natural logarithm
EXP	EX.	○	○	A = EXP B	A = e^B
√		○	○	A = √	
DMS	DM.	○	○	A = DMS B	Conversion to sexagesimal notation

Instruction	Abbreviation	Usability in RUN and DEF modes	Programmability	Example	Note
DEG		○	○	A = DEG B	Conversion to decimal notation
INT		○	○	A = INT B	Obtains integer within a range that does not exceed integer portion of B.
ABS	AB.	○	○	A = ABS B	Obtains the absolute value A = | B |.
SGN	SG.	○	○	A = SGN B	If B > 0, A = 1 B = 0, A = 0 B < 0, A = 1
AREAD	A.		○	AREAD A	Only when it is in the first line of the executing program in the DEF mode, the contents of the display shown before the execution is read into A.
BEEP	B.		○	BEEP A	Sound buzzer as many times as A.
CLEAR	CL.	○	○	CLEAR	Clears all data variables.
DEGREE	DEG.	○	○	DEGREE	Sets the angle mode to DEG (decimal notation).
END	E.		○	END	Terminates the program execution.
FOR	F.		○	10FORA=0 TO 10 STEP 2 100 NEXT A	Increments by 2 from A = 0 to A = 10, during which time program lines up to NEXT A are repeated. Repeats the lines 10 through 100 for 6 times as A = 0 advances to 2, 4, · · · , 10.
GOTO	G.		○	10 GOTO 100	Jumps to the line number 100.
GOSUB	GOS.		○	10 GOSUB 100	Jumps to the subroutine in the line number 100.
GRAD		○	○	GRAD	Sets the angle mode to GRAD.
IF			○	10 IF A = B	Decision instruction with which the successive statement is executed when the IF statement is satisfied, or executes the next line when the IF statement is not satisfied.

Instruction	Abbreviation	Usability in RUN and DEF modes	Programmability	Example	Note
INPUT	I.		○	INPUT A INPUT A$	Data input through the keyboard
LET	LE.		○	LET A = 10 LET A$ = "SHARP"	Substitute instruction (can be omitted except immediately after the IF statement).
NEXT	N.		○	NEXT A	Used in pair with FOR (see FOR).
PAUSE	PA.		○	PAUSE A	Holds the display for a period of 0.85 second.
PRINT	P.		○	PRINT A PRINT A, B PRINT A;B;C	Displays A. Displays A and B on the right and left hand sides. Displays A, B and C in succession from the left.
RADIAN	RA.	○	○	RADIAN	Set the angle mode to RAD (radian).
REM			○	REM "INTEREST"	A comment statement without any effect to the program execution.
RETURN	RE.		○	RETURN	End of subroutine. The program execution returns to execute the statement next to the GOSUB instruction after the execution of this instruction.
STEP	STE.		○		See FOR.
STOP	S.		○	STOP	Suspends program execution.
THEN	T.		○	IF ··· THEN 60	Written after the IF instruction to indicate jump line number.
USING	U.		○	PRINT USING "###.##";A	Designates the format in relation with PRINT instruction. In this example, A is designated to 3 digits of integer and 2 digits of figure below the decimal point.
CONT	C.	○		CONT	With this instruction the normal operation is resumed from the suspended state by the execution of the STOP instruction, BREAK key, and debug mode.

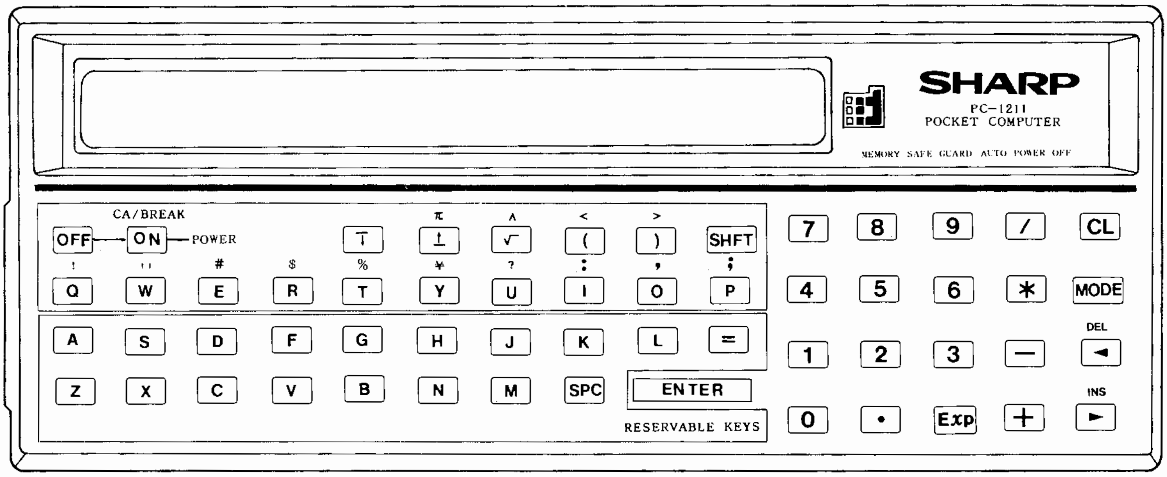
Instruction	Abbreviation	Usability in RUN and DEF modes	Programmability	Example	Note
DEBUG	D.	○		DEBUG	Direct the program execution under the debug mode.
LIST	L.	Only applicable for the PRO mode.		LIST LIST 100	Displays the contents of the program (line number 100).
MEM	M.	○ (Executable even in the PRO mode.)			Unused portion of the flexible memory is indicated on the display in term of numbers of steps and data memories.
NEW		○ (Executable even in the PRO mode.)		NEW	Clears the program and data memories. Starts
RUN	R.	○		RUN	Starts program execution (from the line with the youngest number).
				RUN 100	Starts program execution from the line number 100.
CSAVE	CS.	○ (Executable even in the PRO mode.)		CSAVE "File name"	Stores the program in the computer to the tape with its file name.
CLOAD	CLO.	○ (Executable even in the PRO mode.)		CLOAD "File name"	Program recorded in the tape with the file name is transferred to the computer.
CLOAD?	CLO.?	○ (Executable even in the PRO mode.)		CLOAD? "File name"	Collates the program in the tape with the program in the computer on the basis of file name.
CHAIN	CH.		○	CHAIN "File name" CHAIN "File name", 10	The program in the tape specified by the file name is transferred to the computer and executed. With the latter statement in the examples, program is transferred after the line number 10 and executed.
PRINT#	P. #	○	○	PRINT# "File name" PRINT# "File name", C	The contents of the data memory (after C) is recorded in the tape with its file name.
INPUT#	I. #	○	○	INPUT# "File name" INPUT# "File name", C	The data stored in the tape corresponding to the file name is transferred to the data memory area in the computer (after C).

Error code	Kinds of error	Description
1	● Syntax error ● Arithmetic operation error ● Memory designation error	● When the absolute value of the result exceeds 1 x 100^{100} or when the divisor is 0. ● When the numerical value has been defined in the memory or when designated as a character variable, and vice versa.
2	Line error	● When the line or label specified by GOTO, GOSUB, RUN, DEBUG, or LIST statement is not existing.
3	Depth error	● When the depth exceeds the 4 stages in a GOSUB or FOR/NEXT statement. ● When the RETURN instruction is to be executed even if the preceding GOSUB statement is absent. ● When the NEXT statement is to be executed even if the corresponding FOR statement is absent.
4	Step over error	● When a program whose length is longer than the remaining steps in the computer is tried to load. ● When a non-existing dimension memory is designated.
5	Magnetic tape control error	When an error is generated during the execution of the magnetic tape control instruction (such as verify error, check sum error).
6	Format error	When data can not be displayed with the given format in displaying a numerical data by the PRINT or PAUSE statement during the execution of the program.

MEMO

MEMO

Supplement–3



Printed in Japan
0K2T(2826)