SECTION I
Introduction to BASIC

HP BASIC/3000 is a programming language designed for use at a keyboard terminal. It may also be used for batch jobs on paper tape and cards. To use BASIC at a terminal, the terminal must gain access to the BASIC/3000 Interpreter through the HP Multiprogramming Executive Operating System (MPE/3000). The BASIC/3000 Interpreter is the control program for BASIC/3000.

BASIC/3000 consists of statements for writing programs and commands for controlling program operation. This section describes how to log on and log off, how to enter commands and statements and make corrections. A few simple programs are used for illustration, but the actual programming language is not described until Section II.

This manual assumes that the user knows how to connect his terminal, and is familiar with his terminal keyboard. Special keys with particular functions in BASIC/3000 are described in this section.

In this section only, characters typed by the computer are underlined to distinguish them from user input. Subsequent sections assume that this distinction is clear to the user.

Special Keys

return	Must be pressed after every command and statement. It terminates the line and causes the teleprinter to return to the first print position. BASIC returns a linefeed.
linefeed	Advances the teleprinter one line.
CTRL	When pressed simultaneously with another key, converts the key to a control character that is usually non-printing.
CTRL H (H^C)	Deletes the previous character in a line. It prints the character \ for each character deleted.
CTRL X (X^C)	Cancels the line currently being typed. It types three exclamation points on the line and then gives a return and linefeed to the beginning of the next line.
CTRL Y (Y^C)	Suspends a particular BASIC/3000 program or command and returns to the BASIC/3000 Interpreter. To return control to a program, type GO.
BREAK	Stops all BASIC/3000 activity and returns the user to the operating system (MPE/3000). BASIC/3000 can be re-entered by typing RESUME.

Prompt Characters

BASIC/3000 uses a set of prompting characters to signal to the user that certain input is expected or that certain actions are completed.

>	The prompt character for the BASIC/3000 Interpreter; a BASIC command or statement is expected.
:	The prompt character for the MPE Operating System; MPE commands such as HELLO or BASIC are expected.
?	User input is expected during execution of an INPUT statement.
??	Further input is expected during execution of an INPUT statement.
>>	BASIC expects a continuation line when the previous line was terminated by a &.
!!!	A full line has been deleted with CTRL X.
\	A single character was deleted with CTRL H.
???	A BASIC command was mistyped; re-enter it correctly.

Logging On and Off

LOGGING ON

Once the terminal is connected and ready, the user presses the carriage return. MPE responds with a colon (:) at the beginning of the line. The user may now log on. He should know his user and account identification codes, and also the user and account passwords.

To log on, type:

JOANG and STUDENT are sample user and account identification codes. A period must be typed between them. The computer types a mask over which the passwords are typed. This preserves password privacy.

```
:HELLO JOANG.STUDENT

USER PASSWORD?
XXXXXXXXX
ACCOUNT PASSWORD?
XXXXXXXXX
SESSION NUMBER = #S5
WED, MAY 16, 1973,  2:26 PM
HP32000B.Q1.25
```

The last line identifies the Multi-programming Executive Operating System (MPE/3000).

ENTERING BASIC

Following log on, the MPE/3000 Operating System signals it is ready for the next command by printing a colon. The user may now request the BASIC/3000 Interpreter by typing BASIC.

To enter BASIC, type:

BASIC signals that it has control with a greater-than sign at the start of the line.

```
:BASIC

BASIC 01.0
>
```

BASIC commands and statements can now be entered. Each command or statement is prompted by the greater-than sign at the start of a new line.

LEAVING BASIC

When the user is through, he returns control to MPE/3000 with the EXIT command.

To leave BASIC, type:

```
>EXIT
```

The computer prints:

```
END OF PROGRAM
```

and MPE/3000 signals that it has resumed control with a colon.

```
:
```

LOGGING OFF

When a session at the terminal is finished, the user logs off with the MPE/3000 command BYE. He must have already exited from the BASIC Interpreter by typing EXIT. When MPE/3000 prints a colon, the user can type BYE.

To log off, type:

```
:BYE
```

MPE/3000 records the date and the time. It also records the number of minutes the terminal was connected and the seconds of central processor time used.

```
CPU (SEC) = 3
CONNECT (MIN) = 2
WED, MAY 16, 1973,  2:28 PM
END OF SESSION
```

SUSPENDING BASIC

The user may want to return to the MPE/3000 Operating System temporarily. He can leave BASIC, return to MPE/3000 control, enter MPE/3000 commands and then return to the same point in his BASIC program. To do this, he uses the SYSTEM command or the BREAK key. For operation of the BREAK key, see Special Keys, this section.

To suspend BASIC operation:

```
>SYSTEM
```

The computer types a colon:

```
:
```

The user may then enter MPE/3000 commands. When he wishes to return to BASIC, he types the MPE/3000 command RESUME. The system responds with a > .

Correcting Errors

Several types of errors may be made while logging on. We will consider spelling mistakes, format errors and incorrect passwords or codes. The methods for correcting these errors are general and can be used in BASIC as well as under control of MPE/3000.

Corrections can be made while the line is being entered if the error is noticed before *return* is pressed. The control character *CTRL H (H^C)* can be used to correct a few characters just typed, or the control character *CTRL X (X^C)* can be used to cancel the line and start fresh.

Suppose the user misspells the command HELLO. H^C will delete the last character and print a back slash. The user retypes the character correctly and finishes the line. When he presses *return*, the line is entered correctly.

```
:HELO\LO JOANG.STUDENT
USER PASSWORD?
```

If several characters have been typed after the error, H^C must be typed for each character to be deleted.

In this case, four characters including the blank are deleted.

```
:HELO JO\\\\LO JOANG.STUDENT
USER PASSWORD?
```

Another method is to use X^C to cancel the line. X^C must be typed before *return* is pressed.

To cancel the line, type X^C
Three exclamation points are typed and the computer responds with a carriage return and linefeed. The user retypes the line:

```
:HELO!!!
HELLO JOANG.STUDENT
```

BASIC Commands and Statements

Commands

BASIC/3000 commands instruct the BASIC/3000 Interpreter to perform certain control functions. Commands differ from the statements used to write a program in the BASIC/3000 language. A command instructs the interpreter to perform some action immediately, while a statement is an instruction to perform an action only when the program is run. A statement is always preceded by a statement number; a command never is.

Any BASIC/3000 command can be entered following the BASIC prompt character >. Each command is a single word that must be typed in its entirety with no embedded blanks. If misspelled, the computer will return an error message. Some commands have parameters to further define command operation.

For instance, EXIT is a command that signals completion of a BASIC program and return to the operating system. It has no parameters. Another command, LIST, prints the program currently being entered. It may have parameters to specify that only part of the program is to be listed, or to indicate a particular list destination.

Statements

Statements are used to write a BASIC/3000 program that will subsequently be executed. Each statement performs a particular function. Every statement entered becomes part of the current program and is kept until explicitly deleted or the user exits from BASIC with EXIT.

A statement is always preceded by a statement number. This number is an integer between 1 and 15999. The statement number indicates the order in which the statements will be executed. Statements are ordered by BASIC from the lowest to the highest statement number. Since this order is maintained by the interpreter, it is not necessary for the user to enter statements in execution order so long as the numbers are in that order.

Following each statement, *return* must be pressed to inform the interpreter that the statement is complete. The interpreter generates a linefeed and prints the prompt character > on the next line to signal that the statement is accepted. If an error is made entering the statement, the computer prints an error message.

BASIC/3000 statements have a free format. This means that blanks are ignored.

For instance, all these statements are equivalent.

```
>30 PRINT S
>30     PRINT   S
>30PRINTS
>  3 0 P R I N T S
```

Any statement except REM (to introduce remarks) can continue on more than one line. Each line to be continued must end with the character &; only the first line has a statement number. When the computer expects a continuation line, it prompts with two greater-than characters.

The statement 100 PRINT 35+5 is entered on two lines:

```
>100 PRINT&
>>35+5
```

Error Messages

If an error is made in a line and the line is entered with *return*, the interpreter types a message. The message consists of the word ERROR followed by @ and a number indicating about how many non-blank characters were read before an error was detected.

If this line is entered; the computer prints a message.

```
>30 PRING S
ERROR@2
```

The user then presses *return* and enters the correct line after the BASIC prompt character > .

If the mistake is not obvious, type any character after the message instead of pressing *return.* The computer will print a diagnostic message.

For instance:

```
>30 PRING S
ERROR@2;
UNRECOGNIZABLE STATEMENT TYPE
```

Typing a semi-colon causes the diagnostic message to be printed. Any other character, except a colon, could have been typed with the same result. A colon will cause an abort.

Changing or Deleting a Statement

If an error is made before *return* is pressed, the error can be corrected with *CTRL H (H^C)* or the line may be cancelled with *CTRL X (X^C)*. (See Correcting Errors, above). After *return* is pressed, the error can be corrected by deleting or changing the statement.

To change a statement, simply type the statement number followed by the correct statement.

To change this statement: `>30 PRINT X`
retype it as: `>30 PRINT S`

A change such as this can be made any time before the program is run.

To delete a statement, type the statement number followed by *return*.

Statement 30 is deleted: `>30`

The DELETE command, described in section II, is useful to delete a group of statements.

BASIC Programs

Any statement or group of statements that can be executed constitutes a program.

A program can have as few as one statement.

This is an example of a program with only one statement.

```
>100 PRINT 35+5
```

100 is the statement number. PRINT is the key word or instruction that tells the interpreter the kind of action to perform. In this case, it prints the result of the expression that follows. 35+5 is an arithmetic expression. It is evaluated by the interpreter, and when the program is run, the result is printed.

The statement 100 PRINT 35+5 is a complete program since it can run with no other statements and produce a result. Usually a program contains more than one statement.

These three statements are a program:

```
>10 INPUT A,B,C,D,E
>20 LET S = (A+B+C+D+E)/5
>30 PRINT S
```

This program, which calculates the average of five numbers, is shown in the order of its execution. It could be entered in any order if the statement numbers assigned to each statement were not changed.

This program runs exactly like the program above.

```
>20 LET S=(A+B+C+D+E)/5
>10 INPUT A,B,C,D,E
>30 PRINT S
```

It is generally a good idea to number statements in increments of 10. This allows room to intersperse additional statements as needed.

User's Work Area

When statements are typed at the terminal, these statements become part of the user's work area. All statements in the user's work area constitute the current program.

Any statement in the user's work area can be edited or corrected; the resulting statement will then replace the previous version in the user's work area.

When the user exits from BASIC with the EXIT command, the work area is cleared. If, however, he only suspends BASIC operation with the SYSTEM command, the *BREAK* key, or the *CTRL Y* keys, the user's work area is not changed.

Listing a Program

At any time while a program is being entered, the LIST command can be used to produce a listing of the statements that have been accepted by the computer. LIST causes the computer to print a listing of the current program at the terminal.

After deleting or changing a line, LIST can be used to check that the deletion or correction has been made.

A correction is made while entering a program:

```
>10 U\INPUT A,B,C,D,E
>20 PR\\LET S = (A+B+C+D+E)/5
>30 PRINT S
```

To check the correction, list the program:

```
>LIST
  10 INPUT A,B,C,D,E
  20 LET S=(A+B+C+D+E)/5
  30 PRINT S
>
```

Note that the greater-than prompt character is not printed in the listing, but is printed when the list is complete to signal that BASIC is ready for the next command or statement.

Should the statements have been entered out of order, the LIST command will cause them to be printed in ascending order by statement number.

For instance, the program is entered in this order:

```
>20 LET S = (A+B+C+D+E)/5
>30 PRINT S
>10 INPUT A,B,C,D,E
```

The list is in correct order for execution:

```
>LIST
  10 INPUT A,B,C,D,E
  20 LET S=(A+B+C+D+E)/5
  30 PRINT S
>
```

Running a Program

After the program is entered and, if desired, checked with LIST, it can be executed with the RUN command. RUN will be illustrated with two sample programs.

The first program has one line:

```
>100 PRINT 35+5
```

When run, the result of the expression 35+5 is printed:

```
>RUN
 40
```

Because the program contains a PRINT statement, the result is printed when the program is run.

The second sample program averages a group of five numbers. The numbers must be input by the user:

```
>10 INPUT A,B,C,D,E
>20 LET S=(A+B+C+D+E)/5
>30 PRINT S
```

Each of the letters following the word INPUT and separated by commas names a variable that will contain a value input by the user from the terminal. When the program is run, the interpreter signals that input is expected by printing a question mark. The user enters the values following the question mark. They are entered with a comma between each successive value.

The statement LET S = (A+B+C+D+E)/5 assigns the value of the expression to the right of the equal sign to the variable S on the left of the equal sign. The expression first adds the variable values within parentheses and then divides them by 5. The result is the value of S.

When the program is run, the user enters input values and the computer prints the result:

```
>RUN
?7,5,6,8,9
 7
```

Deleting a Program

If a program that has been entered and run is no longer needed, it can be deleted with the SCRATCH command. Typing SCR or SCRATCH deletes whatever program has been entered by the user during the current session.

The first program entered was 100 PRINT 35+5. After it has run, it should be scratched before entering the next program. Otherwise both programs will run when RUN is typed. They will run in the order of their statement numbers. For instance, if both programs are currently in the user's work area, the program with numbers 10 through 30 executes before line 100.

Both programs will run when RUN is typed:

```
>100 PRINT 35+5
>10 INPUT A,B,C,D,E
>20 LET S=(A+B+C+D+E)/5
>30 PRINT S
>RUN
?7,5,6,8,9
 7
 40
```

To avoid confusing results, a program that has been entered and run can be deleted with SCRATCH:

After entering and running:

```
>100 PRINT 35+5
>RUN
 40
```

the program is scratched:

```
>SCRATCH
```

The users work area is now cleared and another program can be entered.

The second program is entered:

```
>10 INPUT A,B,C,D,E
>20 LET S=(A+B+C+D+E)/5
>30 PRINT S
>RUN
?15,25,32,11,29
 22.4
```

Unless this program is to be run again, it can now be scratched and a third program entered.

Documenting a Program

Remarks that explain or comment can be inserted in a program with the REM statement. Any remarks typed after REM will be printed in the program listing but will not affect program execution. The remarks cannot be continued on the next line, but as many REM statements can be entered as are needed.

The sample program to average 5 numbers can be documented with several remarks:

```
>5 REM  THIS PROGRAM AVERAGES
>7 REM  5 NUMBERS
>15 REM 5 VALUES MUST BE INPUT
>25 REM S CONTAINS THE AVERAGE
```

The statement numbers determine the position of the remarks within the existing program. A list will show them in order:

List of sample program including remarks:

```
>LIST
   5 REM  THIS PROGRAM AVERAGES
   7 REM  5 NUMBERS
  10 INPUT A,B,C,D,E
  15 REM 5 VALUES MUST BE INPUT
  20 LET S=(A+B+C+D+E)/5
  25 REM S CONTAINS THE AVERAGE
  30 PRINT S
>
```

When run, the program will execute exactly as it did before the remarks were entered.