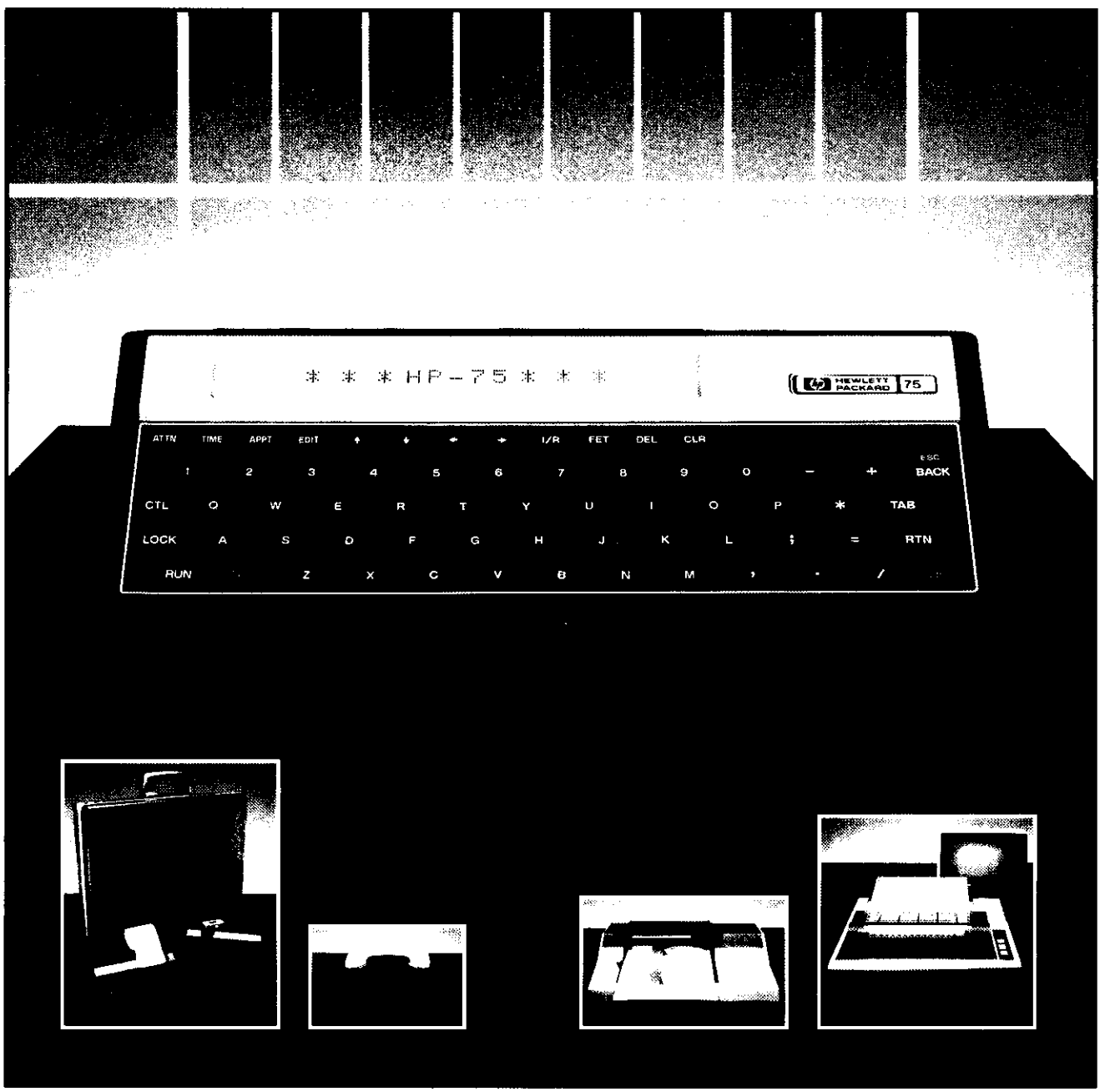
HEWLETT-PACKARD

HP-75

REFERENCE MANUAL



HEWLETT PACKARD

HP-75

Reference Manual

August 1982

00075-90004

Printed in U.S.A.

Contents

Introduction

This reference manual was designed to be your principal reference tool after you have learned to use and program your HP-75. It is compact enough to go virtually anywhere your computer can go, yet comprehensive enough to provide most of the information you might need for routine use of the HP-75.

The reference manual is divided into four sections, each with a different purpose:

- Section 1, Operating the HP-75, contains information about the operation of the computer and its systems. The topics in this section are presented in the same order as in part I and II of the owner's manual, and each topic contains the principal facts relating to the operation covered by the topic. Refer to this section when you need information about some aspect of the computer's operation.
- Section 2, Programming Concepts, contains information about writing and running programs. The topics in this section are presented in the same order as in part III of the owner's manual. Each topic lists, and briefly defines, the instructions used to implement the programming concepts covered by the topic. Refer to this section when you need general information about a programming concept.
- Section 3, HP-75 Instruction Set, is the heart of the reference manual and contains a complete dictionary of every operator, function, statement, and command in the HP-75 instruction set. Refer to this section when you need detailed information about a particular HP-75 instruction.
- Section 4, Reference Tables, provides detailed information about the character set, error conditions, system memory requirements, display escape codes, machine defaults, and keyword abbreviations.

Throughout this manual you will find bold-type page numbers inside parentheses. These page numbers refer to the page in the owner's manual where the principal coverage of the operation, concept, or instruction is located, and they provide a way to access the more complete coverage of certain topics that the owner's manual provides.

Operating the HP-75

Keyboard and Display Control

Key Functions

Typewriter Keys

[A] through [Z]	Letters.
[0] through [9]	Digits.
[ ]	Space.
[.]	Period. Used as a decimal point in numbers and as the final character in abbreviations.
[+] [-] [*] [/] [^] [\] ([CTL] [/])	Arithmetic symbols: addition (plus), subtraction (minus), multiplication (asterisk), division (slash), exponentiation (circumflex), and integer division (back slash).
[,]	Comma. Used to separate items in commands, statements, and functions.
[(] [)]	Parentheses. Used to key in numeric expressions.
[!]	Exclamation mark. Used for end-of-line comments in program statements and for appointment notes.
["] [']	Double and single quotation marks. Used to enclose filenames and other literal strings.
[#]	Number sign. Used to specify file numbers of BASIC files in ASSIGN #, PRINT #, RESTORE #, and READ # statements and to assign timer numbers in ON TIMER # and OFF TIMER # statements. Also used for inequality in relational tests.
[$]	Dollar sign. Used to specify string variables and string functions.
[&]	Ampersand. Used to concatenate, or join, string expressions.
[[] []]	Opening and closing brackets. Used to dimension string variables and to specify substrings.
[@]	Commercial at. Used to form multistatement program lines.
[;]	Semicolon. Used to separate items in PRINT, DISP, INPUT, PRINT #, and READ # statements.
[=]	Equals. Used to assign variable values and to test for equality.
[<]	Less-than. Used in relational tests.
[>]	Greater-than. The BASIC prompt for programs and command appointments. Also used in relational tests.
[:]	Colon. The text-editing prompt. Also used to delimit device codes.

[%] Percent sign.

[?] Question mark. The default prompt for the INPUT statement.

Editing Keys

[SHIFT] Shift. Causes keys to perform their shifted functions. In uppercase mode, causes the letter keys to display lowercase letters.

[CTL] Control. Used to generate special display characters and to form a variety of keystroke combinations.

[LOCK] Lock. Used when locking the keyboard in uppercase letters or when enabling the numeric keypad.

[←][→] Left-arrow and right-arrow. Move the cursor across the display. If necessary, cause the display to scroll.

[↑][↓] Up-arrow and down-arrow. Move the display up and down through BASIC, text, and appointment files and through the system and mass storage catalogs.

[I/R] Insert/Replace. Exchanges the replace cursor (▮) with the insert cursor (▮).

[CLR] Clear. Clears the display.

[BACK] Backspace. Backspaces the cursor and erases the character there.

[DEL] Delete. Deletes a character and left-shifts the trailing characters in the display line.

[TAB] Tab. Moves the cursor across TIME and APPT mode display fields.

System Keys

The system keys and the [↑] and [↓] keys send a carriage-return/line-feed to the display and DISPLAY IS devices before performing their designated functions.

[ATTN] Attention.Turns on the HP-75; interrupts programs, listings, auto line-numbering, card reader operations, and HP-IL operations. Acknowledges due appointments.

[TIME] Time. Switches the HP-75 to TIME mode.

[APPT] Appointment. Switches the HP-75 to APPT mode. Causes unacknowledged appointments to be displayed.

[EDIT] Edit. Switches the HP-75 to EDIT mode.

[FET] Fetch. In EDIT mode, functions as a typing aid for FETCH command.

[RTN] Return. Causes an expression, statement, or command in the display to be evaluated, stored, or executed.

[RUN] Run. Runs the current BASIC program in EDIT mode. Processes due appointments in APPT mode.

Keystroke Combinations

System Keystrokes

[SHIFT] [CTL] [CLR] Causes the HP-75 to perform a system reset.*

[SHIFT] [ATTN] Causes the HP-75 to turn off the display and to disable all keys except [ATTN].

[SHIFT] [LOCK] Locks the HP-75 in uppercase. Pressing [LOCK] again restores the unshifted keys.

[CTL] [LOCK] Enables the numeric keypad. Pressing [LOCK] again restores the normal keyboard.

[SHIFT] [FET] Displays the most recent ERROR, WARNING, card reader, trace, or HP-IL message for as long as the [FET] key is pressed. The message is cleared after [CLR], [ATTN], [TIME], [APPT], [EDIT], [FET], [↑], [↓], [RTN], or [RUN] is pressed.

[SHIFT] [RUN] Single-steps through a program, beginning from the current line.

Editing Keystrokes

[SHIFT] [↑] Moves the file pointer to the first line of the current file, to the first entry in a CAT ALL or mass storage catalog listing, or to the first appointment in APPT mode.

[SHIFT] [↓] Moves the file pointer to the last line of the current file, to the last entry in a CAT ALL or mass storage catalog listing, or to the last appointment in APPT mode.

[CTL] [FET] In EDIT mode, brings the last entry to the display and makes it available for editing. The entry is displayed intact, so long as no other keys have been pressed since the last [RTN]. The following commands use the input buffer, causing your last entry to be lost: FETCH, FETCH KEY, LIST, LOCK, PLIST, and TRANSFORM.

[SHIFT] [DEL] In EDIT and TIME modes, deletes from the cursor to the end-of-line. In APPT mode, deletes the currently displayed appointment or replaces it with an edited appointment.

[SHIFT] [TAB] In APPT and TIME modes, shifts the cursor leftwards across the display fields, the opposite direction of [TAB] alone.

[SHIFT] [←] Shifts the cursor to the beginning of the display line.

[SHIFT] [→] Shifts the cursor to the end of the display line.

[CTL] [←] Left-shifts the cursor 32 positions or to the beginning of the display line, whichever distance is shorter.

[CTL] [→] Right-shifts the cursor 32 positions or to the end of the display line, whichever distance is shorter.

Display Character Keystroke

[SHIFT] [I/R] Causes the next key or keystroke combination to display the character associated with it regardless of key redefinitions.

---

* [SHIFT] [CTL] [M], [I], [J], [8], and [I/R] also cause a system reset if held for one second or longer.

Escape Keystroke

CTL BACK	Generates decimal code 27 (ESC) as the initial character of escape code sequences.
	The HP-75 responds to 12 escape codes. (Refer to the table of display escape codes in the reference table section.)

Display Control

CTL FET	Recalls input buffer to display for editing.
DELAY	Controls display rate of messages and output.
MARGIN	Determines character position where end-of-line signal appears.
PWIDTH	Sets line length of output to printer devices.
SHIFT I/R	Causes next key or keystroke pressed to display character associated with it.
WIDTH	Sets line length of output to display devices.

Character Manipulation Functions

CHR$	Returns the character associated with the specified decimal character code.
NUM	Returns the decimal character code of the specified character.

Syntax Guidelines

The following syntax conventions are used throughout this manual:

DOT MATRIX TYPE	Words in dot matrix (like LIST) may be entered in lowercase or uppercase letters.
italic type	Items in italic are the parameters you supply.
' ',""	Filenames and other character strings can be enclosed by single or double quotes and can be entered in lowercase or uppercase letters.
[ ]	This type of brackets enclose optional parameters.
...	An ellipsis indicates that the optional item may be repeated.
stacked items	When two or more items are placed one above the other, one and only one of them may be used.
or	When two or more items are separated by "or", one or more instances of either or both items may be included.

File Manipulations

File Types

B	BASIC file.
PB	Private BASIC file.
T	Text file.
A	Appointment file.
L	LEX file.
I	Interchange or LIF1 file.
?	Unknown file.

File Specifiers

File specifiers name and describe the characteristics of files.

Definitions:

filename A string expression which evaluates to a string of one to eight characters. The first character must be a letter or a period; the remaining characters can be letters or digits.

device code A string expression which evaluates to one or two letters, a letter and a digit, or a digit and a letter.

password A string expression which evaluates to one to four letters or digits.

File Specifier Syntax

File Type	Specifiers
Files in Memory	'*filename*' APPT KEYS
Card File Specifier	CARD ':CARD' ':PCRD' '*filename*:CARD' '*filename*:PCRD' '*filename*:CARD/*password*' '*filename*:PCRD/*password*'
Mass Storage File Specifier	'*filename*:*device code*' '*filename*:*device code*/*password*'

File Parameters

Line numbers must be unsigned integers, 0 through 9999.

Increment values and file numbers must round to integers, 1 through 9999.

File specifiers may be specified by string variables and expressions.

File Management Commands

For the Current File Only	For Any BASIC File in Memory	For Any Text File in Memory
AUTO	CALL	CAT
DELETE	CAT	COPY
FETCH	COPY	EDIT
MERGE	EDIT	LIST
NAME	LIST	PLIST
RENUMBER	PLIST	PURGE
	PURGE	RENAME
	RENAME	TRANSFORM
	RUN	
	TRANSFORM	

Executing COPY, DELETE, MERGE, NAME, PURGE, RENAME, or TRANSFORM on an initialized BASIC file will deallocate the program.

Operators and Functions

Precedence of Operators (89)

The table below lists HP-75 operators in their order of precedence. Expressions are evaluated from left to right for operators at the same level.

Performed first.

( )

functions.

^

NOT

*, /, DIV or \ (CTL /)

+, −

=, >, >=, <, <=, <> or #

AND

OR, EXOR

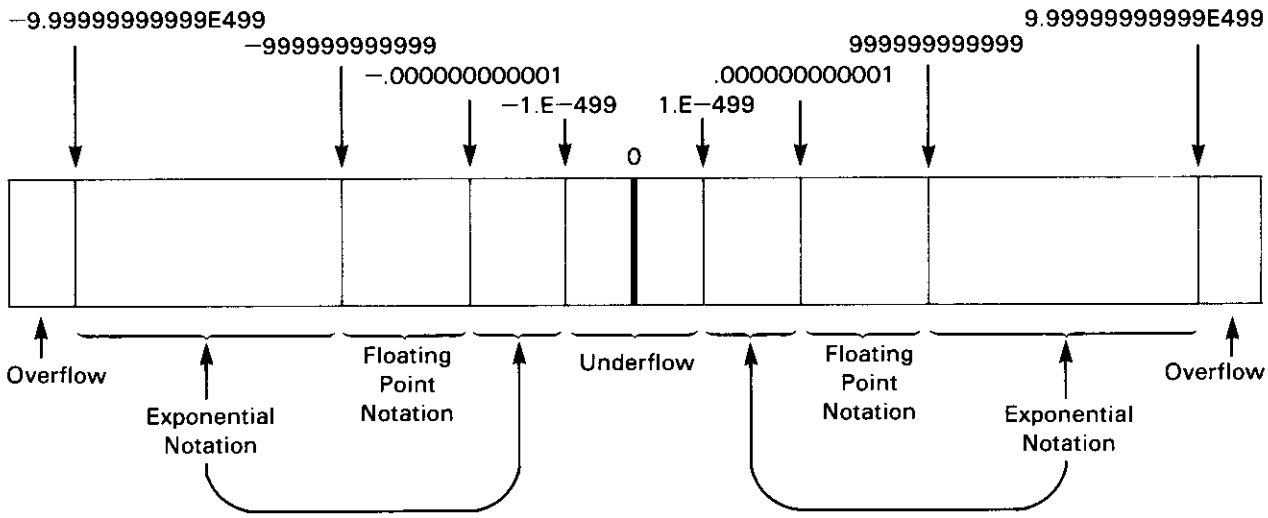
Performed last.

Nested parentheses are evaluated from the inside out.

Numeric Precision (73)

Type	Precision	Maximum Value
REAL	12 digits	$\pm 9.99999999999 \times 10^{\pm 499}$
SHORT	5 digits	$\pm 9.9999 \times 10^{\pm 99}$
INTEGER	5 digits	±99999

Range of Numbers (76)



Variables (78)

Simple Numeric Variables

Identifier: *Letter* [*digit*]

Default type: REAL. SHORT or INTEGER may be declared.

Numeric Array Variables

Identifier: *Letter* [*digit*] (*subscript* [, *subscript*])

Simple numeric and numeric array variables can have identical identifiers. One or two dimensions may be specified.

Default type: REAL. SHORT or INTEGER may be declared.

Default lower bound: 0. OPTION BASE 1 produces lower bound 1.

Default upper bound: 10. Upper bound can be specified by DIM, REAL, SHORT, or INTEGER declarations.

Entire arrays may be referenced in PRINT # and READ # statements by specifying the array name with parentheses and no subscripts, for example C(,).

Numeric Functions

Numeric function parameters x and y can be any numeric expression.

Number Alteration Functions (82)

ABS(x)	FP(x)	FLOOR(x)
IP(x)	INT(x)	CEIL(x)

General Mathematical Functions (83)

SQR(x)	MAX(x,y)	PI
MOD(x,y)	MIN(x,y)	INF
SGN(x)	RMD(x,y)	EPS

Logarithmic Functions (84)

LOG(x)	EXP(x)	LOG10(x)

Trigonometric Commands (85)

OPTION ANGLE DEGREES
OPTION ANGLE RADIANS

Trigonometric Functions (85)

SIN(x)	TAN(x)	SEC(x)
ASIN(x)	ATN(x)	CSC(x)
COS(x)	ANGLE(x)	RAD(x)
ACOS(x)	COT(x)	DEG(x)

Numeric Expressions (87)

A numeric expression can take any of the following forms:

- A numeric constant.
- A numeric variable.
- A numeric function.
- Any of the above combined by one or more operators (arithmetic, relational, or logical) or pairs of parentheses.

TIME Mode Operations (92)

SET	Sets the clock.
ADJST	Adjusts the clock setting.
EXACT	Calibrates the clock (EXACT-ADJST-EXACT).
RESET	Clears speed adjustment factor.
STATS	Specifies DMY-MDY and AM/PM-24 hour formats, YEAR-EXTD appointment calendar, and time accuracy.

APPT Mode Operations (100)

Alarm Types:
0 Beep suppressed.
1 Short chirp.
2 Long, low tone.
3 Two-tone pattern, repeated three times.
4 Series of high tones.
5 Long, low tone followed by long, high tone.
6 Series of eight siren sounds.
7 Repeating type 2 alarm.
8 Repeating type 4 alarm.
9 Repeating type 6 alarm.

Appointment Types: N (normal)
A (acknowledge/reschedule)
R (reschedule)

! or >	Specifies message or command field.
ATTN	Acknowledges due appointments.
SHIFT DEL	Deletes appointments.
SHIFT APPT	Displays extended appointment information.
SHIFT ATTN	Turns HP-75 off, processing due appointments.
COPY appt TO *'file specifier'*	Copies appt file from memory to a magnetic card or mass storage.
COPY *'file specifier'* TO appt	Merges an appt type file from a magnetic card or mass storage with the appt file in memory.

Card Reader Operations (114)

CAT CARD	Displays a card file catalog entry.
COPY ['*filename*'] TO *'card file specifier'*	Copies a file from memory to card.
COPY *'card file specifier'* TO *'filename'*	Copies a file from card to memory.
PROTECT	Write-protects a card file.
UNPROTECT	Removes the write-protection from a card file.

HP-IL Operations (124)

ASSIGN IO	Assigns the devices on the loop.
DISPLAY IS ':' *device code'*	Declares display devices.
PRINTER IS ':' *device code'*	Declares printer devices.
CLEAR LOOP	Resets loop devices.
OFF IO	Turns off loop communications.
RESTORE IO	Restores loop communications.
INITIALIZE ':' *device code'*	Initializes mass storage medium.

CAT ':*device code*'	Catalogs mass storage medium.
COPY ['*filename*'] TO '*file specifier*'	Copies a file from memory to mass storage.
COPY '*file specifier*' TO '*filename*'	Copies a file from mass storage to memory.
RENAME '*file specifier*' TO '*filename*'	Renames a mass storage file.
PURGE '*file specifier*'	Purges a mass storage file.
PACK ':*device code*'	Packs a mass storage medium.

Key Redefinitions (142)

FETCH KEY '*key*'	Fetches the definition of the specified key or keystroke.
DEF KEY '*key*', '*key def*' [;]	Defines the key or keystroke.
SHIFT I/R	Displays the key (or keystroke) display character.
EDIT KEYS	Edits the keys file.
RENAME KEYS TO '*filename*'	Renames the keys file and disables key redefinitions.
RENAME '*filename*' TO KEYS	Renames a keys type file to keys and enables the key redefinitions it contains.
COPY KEYS TO '*file specifier*'	Copies the keys file to a magnetic card or mass storage.
COPY '*file specifier*' TO KEYS	Copies a keys type file from a magnetic card or mass storage to memory and enables the key definitions it contains.
PURGE KEYS	Purges the keys file and eliminates all the key definitions it contains.

Section 2

Programming Concepts

Running Programs (158)

RUN, [RUN]	Begins program execution.
[ATTN]	Interrupts program execution.
CONT	Restarts program execution after [ATTN] or STOP.

Editing Programs (159)

LIST, PLIST	Lists program file to display or printer.
FETCH	Fetches specific lines of program file.

Fundamental Statements (165)

LET	Assigns values to variables.
END	Terminates program and deallocates the program.
STOP	Inerrupts program execution without deallocation.
WAIT	Interrupts program execution for a specified period of time.
REM, !	Delimits program remarks.
DISP, PRINT, TAB	Display or print information.
INPUT	Allows input of data from the keyboard.

Branches, Loops, and Subroutines (176)

GOTO	Unconditionally branches to a line number.
IF...THEN...ELSE	Tests condition and branches.
FOR-NEXT, STEP	Repeatedly executes a series of statements a predetermined number of times.
GOSUB-RETURN	Branches to a series of statements and returns.
ON...GOTO	Branches to the line number determined by the value of the included expression.
ON...GOSUB	Branches to the subroutine determined by the value of the included expression.
POP	Bypasses a pending subroutine return.

Program Timers (186)

ON TIMER #	Sets a program timer.
OFF TIMER #	Disables a program timer.
ON TIMER # GOTO	Branches to a statement when a timer goes off.
ON TIMER # GOSUB	Branches to a subroutine when a timer goes off.

Arrays (192)

OPTION BASE	Defines the lower bound of all arrays in a program.
DIM, REAL, SHORT, INTEGER	Declare and dimension arrays.

Strings

String Variables (196)

String Identifier: *letter* [*digit*] $

Dimensioning a String: DIM *string identifier* [*subscript*]

Default size of a string variable is 32 characters.

Substring Identifier: *letter* [*digit*] $[*subscript* [, *subscript*]]

Two subscripts separated by a comma specify beginning and ending character positions, respectively. A single subscript specifies a beginning character—the substring extends to the end of the string.

String Expressions (196)

A string expression can take any of the following forms:

- A string constant.
- A string variable.
- A substring.
- A string function.
- A string expression consisting of any concatenation of the above using the & operator.

String Functions (198)

The parameters s$ and t$ can be any string expression.

LEN(s$)
POS(s$, t$)
VAL(s$)
STR$(*numeric expression*)
UPRC$(s$)
KEY$
CAT$(*file number*)

User Defined Functions (205)

DEF FN	Defines a single-line user-defined function or the first line of a multiline function.
LET FN	Assigns a value to the function in a multiline function.
END DEF	Labels the end of a multiline function.

Storing and Retrieving Data

Within a Program (210)

DATA	Statement Contains numeric or string constants for use by READ.
READ	Assigns values from DATA statements to variables.
RESTORE	Resets the data pointer to the first or specified DATA statement in the file.

From Data Files (216)

ASSIGN #	Associates a file number with a file name.
PRINT #	Stores data items in a data file.
READ #	Retrieves data items from a data file.
RESTORE #	Resets the data pointer to the first or specified line of a data file.

Program Calls (230)

CALL	Calls a program from within another program. Execution returns to the calling program at the line after the CALL statement when the called program ends.

Local and Global Declarations (233)

A *global* declaration is an HP-75 setting that remains in effect until:

- The setting is changed by a new declaration, executed either from the keyboard or from a running program.
- The HP-75 is reset.

A *local* declaration affects only keyboard calculations or the program in which the declaration occurs.

Local	Global
DATA	ALARM OFF, ALARM ON
DEF FN, LET FN, END DEF	ASSIGN IO, OFF IO, RESTORE IO
DIM	ASSIGN #
INTEGER, SHORT, REAL	BEEP OFF, BEEP ON
IMAGE	DEFAULT OFF, DEFAULT ON
LET	DEF KEY
OFF ERROR, ON ERROR	DELAY
ON TIMER #	DISPLAY IS, PRINTER IS
OPTION BASE	ENDLINE
	LOCK
	MARGIN
	OFF TIMER #
	OPTION ANGLE DEGREES
	OPTION ANGLE RADIANS
	STANDBY ON, STANDBY OFF
	TRACE FLOW, TRACE VARS, TRACE OFF
	WIDTH, PWIDTH

Formatting Output (238)

IMAGE — Specifies the output format for DISP USING and PRINT USING statements.

DISP USING — Displays information according to the specified IMAGE statement or the included format string.

PRINT USING — Prints information according to the specified IMAGE statement or the included format string.

Debugging Operations (252)

TRACE FLOW — Traces program flow. Shows all branches in program execution.

TRACE VARS — Traces variables. Shows all changes in values of variables.

TRACE OFF — Turns off all trace operations.

ON ERROR — Initiates user-defined error trapping.

OFF ERROR — Disables user-defined error trapping.

ERRN — Displays error number of last error.

ERRL — Displays line number where last error occurred.

HP-75 Instruction Set

Operators

Arithmetic Operators (69)

[+]	Floating point addition.
[-]	Floating point subtraction.
[*]	Floating point multiplication.
[/]	Floating point division.
[CTL][/] or DIV	Integer division (no remainder).
[^]	Exponentiation.

Relational Operators (82)

[=]	Equal to.
[<]	Less than.
[<][=]	Less than or equal to.
[>]	Greater than.
[>][=]	Greater than or equal to.
[<][>] or [#]	Not equal to.

Logical Operators (88)

Logical operators are used with numeric expressions to return Boolean values. If an expression evaluates to 0 it is false; if it evaluates to a nonzero value it is true. For all numeric expressions *A* and *B*:

A AND *B*	Is true if and only if both *A* and *B* are true.
A OR *B*	Is true if *A* or *B* or both are true.
A EXOR *B*	Is true if *A* is true or *B* is true but not both.
NOT *A*	Is true if *A* is false.

A	B	A AND B	A OR B	A EXOR B	NOT A
T	T	T	T	F	F
T	F	F	T	T	F
F	T	F	T	T	T
F	F	F	F	F	T

Functions

In the following tables, X and Y specify any two numeric expressions and S$ and T$ specify any two string expressions.

Where an abbreviation exists for the function name, it is shown in parentheses following the function description.

Numeric Functions

Given the appropriate number of arguments (zero or more) and type of arguments (numeric or string expressions), each numeric function returns a single numeric constant.

ABS(X)	Absolute value of X. **(82)**
ACOS(X)	Arccosine of X, in Quadrant I or II (abbreviation: ac.). **(86)**
ANGLE(X,Y)	Arctangent of Y/X, in proper quadrant. That is, returns the angle θ formed between the x-axis and the point (x,y), such that $-\pi < \theta \leq \pi$ (abbreviation: an.). **(86)**
ASIN(X)	Arcsine of X, in Quadrant I or IV. **(86)**
ATN(X)	Arctangent of X, in Quadrant I or IV. **(86)**
CEIL(X)	Smallest integer $\geq$ X (abbreviation: ce.). **(82)**
COS(X)	Cosine of X. **(86)**
COT(X)	Cotangent of X. **(86)**
CSC(X)	Cosecant of X. **(86)**
DATE	The date in *yyddd* format based on the clock setting. **(98)**
DEG(X)	Radian-to-degree conversion of X. **(86)**
EPS	Smallest machine number (1.E-499). **(83)**
ERRL	The line number of the most recent error or warning. **(260)**
ERRN	The identification number of the most recent error or warning (abbreviation: er.). **(260)**
EXP(X)	e^X. **(84)**
FLOOR(X)	Same as INT(X) (abbreviation: fl.). **(82)**
FP(X)	Fractional part of X. **(82)**
INF	Largest machine number (9.9999999999E499). **(83)**
INT(X)	Largest integer $\leq$ X. **(82)**
IP(X)	Integer part of X. **(82)**
LEN(S$)	The length of string S$. **(198)**
LOG(X)	Natural logarithm of X, X > 0. **(84)**
LOG10(X)	Log to the base 10 of X, X > 0 (abbreviation: lo.). **(84)**
MAX(X,Y)	If X > Y then X, else Y. **(83)**
MEM	The number of bytes of available memory. **(47)**
MIN(X,Y)	If X < Y then X, else Y. **(83)**

`MOD(X,Y)` X modulo Y: `X-Y*INT(X/Y)`. **(83)**

`NUM(S$)` The decimal code of the first character of `S$`. **(41)**

`PI` `3.14159265359`. **(83)**

`POS(S$,T$)` Searches string `S$` for the first occurrence of `T$`. Returns the starting position if found; `0` if not. **(198)**

`RAD(X)` Degree-to-radian conversion of `X`. **(86)**

`RES` The last numeric result to be displayed or printed. **(71)**

`RMD(X,Y)` Remainder of `X/Y`: `X-Y*IP(X/Y)`. **(83)**

`RND` Next number, R, in sequence of pseudo-random numbers, $0 \leq R < 1$. **(83)**

`SEC(X)` Secant of `X`. **(86)**

`SGN(X)` The sign of `X`: -1 if $X < 0$, `0` if $X = 0$, 1 if $X > 0$. **(83)**

`SIN(X)` Sine of `X`. **(86)**

`SQR(X)` Positive square root of `X`. **(83)**

`TAN(X)` Tangent of `X`. **(86)**

`TIME` The number of seconds since midnight of the current day. **(98)**

`VAL(S$)` Returns the numeric value of a string composed of digits, decimal point, and/or exponent. **(198)**

String Functions

String functions return zero or more characters of information.

`CAT$(X)` The catalog entry of the specified file, 32 characters in length. Files are numbered in order of their appearance in the system catalog. `CAT$(0)` returns the catalog of the current EDIT file. For $X < 0$, `CAT$(X)` returns the catalog of the currently initialized BASIC file, if any (abbreviation: `c.`). **(198)**

`CHR$(X)` The character whose decimal code is `MOD(X,256)` (abbreviation: `ch.`). **(41)**

`DATE$` The date in a *yy/mm/dd* format. **(98)**

`KEY$` The display character of the currently depressed key or keystroke combination. Returns the null string if no key is depressed (abbreviation: `k.`). **(198)**

`STR$(X)` The string information contained in the digits, decimal point, sign, and exponent of `X`. **(198)**

`TIME$` The time in a *hh:mm:ss* format, using 24-hour notation (abbreviation: `ti.`). **(98)**

`UPRC$(S$)` Converts `S$` to all uppercase letters (abbreviation: `upr.`). **(198)**

`VER$` A six-character string indicating the operating system version. **(267)**

Print Function

TAB(X) **Causes the following DISP or PRINT item to be output beginning at column X, where column 1 is the left margin. (167)**

Basic Statements and Commands

In the list that follows, the shortest abbreviation for each keyword is shown in parentheses following the keyword. The word "none" in the parentheses indicates that no abbreviation exists for that keyword.

The bold number in parentheses at the far right on the keyword line is the principal page reference for that keyword in the owner's manual. (This is the same page reference given in the Instruction set Index inside the back cover of the owner's manual.)

ALARM (al.) (106)

```
ALARM OFF
```

Causes the HP-75 to ignore due appointments.

- Global Declaration.

```
ALARM ON
```

Restores the normal handling of due appointments. All appointments that came due after the last ALARM OFF will come due immediately.

- Global declaration.

ASSIGN # (as.#) (216)

```
ASSIGN #file number TO 'filename' [, BASIC]
                                  [, TEXT ]
```

```
100 ASSIGN # 14 TO 'DART'
110 ASSIGN # V8 TO 'RATE',TEXT
```

Assigns the specified file number to the named data file and sets the corresponding data pointer to the first line of the file. A new file is created if the named file doesn't exist. If the file type is not specified, the default is BASIC.

- Global declaration.

(219)

```
ASSIGN # file number TO *
                        ''
                        '*'
```

```
120 ASSIGN # 14 TO *
130 ASSIGN # V8 TO ''
```

Ends the association between the specified file pointer and a file. Reclaims the memory required by the assignment.

- Global declaration.

ASSIGN IO (as.) (126)

```
ASSIGN IO [':device code [,:device code...]']
```

```
ASSIGN IO
140 ASSIGN IO ':TV,:CA'
```

Initiates interactive assignments of device codes to peripherals or assigns them as specified in the optional device code list. Devices are assigned according to their order on the loop.

- Global declaration.

AUTO (a.) (51)

```
AUTO [beginning line number [,increment value]]
```

```
auto
auto 500
auto 5,5
```

Causes the computer to begin automatic line numbering of text or BASIC files; numbering begins at *beginning line number* and increases by *increment*. Cancel by pressing [ATTN]. Defaults are current line plus 10 and increment value 10.

BEEP (be.) (30)

```
BEEP [frequency in Hz [,duration in seconds]]
```

```
150 BEEP
160 BEEP 400,1.5
170 BEEP W/(2*PI),T/60
```

Causes a tone to sound at the specified *frequency* for the specified *duration in seconds*. Defaults to 1400 Hz and 0.1 seconds.

```
BEEP OFF
```

Disables the beeper until a BEEP ON command is executed. Alarm types 6 and 9 will still sound.

- Global declaration.

```
BEEP ON
```

Restores beeper operation after a BEEP OFF command.

- Global declaration.

BYE (b.) (29)

```
BYE
```

Turns the HP-75 off.

CALL (none) (230)

```
CALL 'filename'
```

```
180 CALL 'PART2'
```

Causes program execution to branch to the specified program. Execution returns to the calling program when an END is reached during execution of the called program.

CAT ALL (C, .) (49)

```
CAT ALL
```

Accesses the complete system catalog.

CAT (none) (49)

```
CAT APPT
```

Displays the catalog entry of the appt file.

(117)

```
CAT CARD
```

Displays the catalog information recorded on a card track.

(134)

```
CAT ':device code'
```

```
cat ':C2'
```

Accesses the catalog of the medium in the specified mass storage device.

```
CAT ['filename']
```

```
cat
cat 'names'
190 CAT F1$
```

Displays the catalog entry of the current BASIC or text file or the specified file in memory.

(134)

```
CAT 'mass storage file specifier'
```

```
cat 'testdat:c2'
200 CAT T1$ & ':C2'
```

Displays the catalog entry of the specified mass storage file.

(144)

```
CAT KEYS
```

Displays the catalog entry of the `keys` file.

CLEAR (cl.) (131)

```
CLEAR ':device code [,:device code...]'
```

```
210 CLEAR ':TP,:TV'
```

Resets specified HP-IL devices to their initial states. (Refer to peripheral owner's manuals for initial state conditions.)

CLEAR LOOP (cl.l.) (131)

```
CLEAR LOOP
```

Resets all HP-IL devices on the loop to their initial states.

CLEAR VARS (cl..) (81)

```
CLEAR VARS
```

Clears the values of all calculator and program variables and reclaims the memory required for them.

CONT (none) (159)

```
CONT [line number]
```

```
cont
cont 550
```

Continues execution of the current program at the next statement or at the beginning of the specified line.

COPY (co.) (118/135)

```
COPY [ 'filename' | APPT | KEYS ] TO 'filename' | CARD | 'card file specifier' | 'mass storage file specifier'
```

```
copy to card
copy 'oldfile' to 'newfile'
copy appt to 'app10:ca'
220 COPY 'FILE6' TO ':PCRD'
copy 'file5' to 'file5:tp'
230 COPY F1$ TO F2$
```

Copies the specified file in memory to the specified destination. If the source file is not specified, copies the current file.

(119)

```
COPY CARD          TO 'filename'
     'card file specifier'    APPT
                              KEYS
```

```
copy card to 'ditto'
240 COPY 'DATE:PCRD' TO 'NEWS'
```

Copies the specified source card file to the specified destination filename in memory.

(135)

```
COPY 'mass storage file specifier' TO APPT
                                      KEYS
                                      'filename'
                                      'mass storage file specifier'
```

```
copy 'app10:ca' to appt
copy 'file5:tp' to 'file5'
250 COPY 'RECORD5:CA' TO 'BUFFER'
260 COPY F1$ & ':C1' TO F1$ & ':C2'
```

Copies the specified source mass storage file to the specified destination file in memory or on mass storage.

DATA (da.)

(210)

```
DATA number or text [,number or text]
```

```
270 DATA 6, 72.3, dollars, 1E44, 'pounds'
```

Provides numeric and quoted or unquoted strings from which READ statements can obtain values for numeric and string variables. DATA statements can be located anywhere in the program. Data items are read from left to right within a DATA statement and from the lowest-numbered to highest-numbered DATA statement in the program.

- Local declaration.
- Not executable from the keyboard.
- Not allowed after THEN, ELSE, ON ERROR, or ON TIMER.
- Must be the only instruction on the line.

DEFAULT (defa.)

(89)

```
DEFAULT OFF
```

Cancels the use of default values for improper mathematical expressions. Program execution is halted and an error message is returned when errors 1 through 8 are encountered. (Refer to Error Conditions, section 4.)

- Global declaration.

```
DEFAULT ON
```

Restores the use of default values for improper mathematical expressions. Errors 1 through 8 produce warning messages and program execution continues.

- Global declaration.

DEF FN (none) (205)

```
DEF FN numeric variable name [(parameter [,parameter...])] = numeric expression
DEF FN string variable name [(parameter [,parameter...])] = string expression
```

```
280 FNA (A1, C4) = A1 * TAN(C4)
290 DEF FNB (X) = SQR(X^2+3*X+1)
300 DEF FNC$ = CHR$(X4)
```

Completely defines a single-line user-defined function. Parameters can be any combination of simple numeric or string variables and are local to the function definition.

- Local declaration.
- Not executable from the keyboard.
- Not allowed after THEN, ELSE, ON ERROR, or ON TIMER.
- Must be the only instruction on the line.

(207)

```
DEF FN numeric variable name [(parameter [,parameter...])]
DEF FN string variable name [(parameter [,parameter...])]
```

```
310 DEF FNA (B1, B2)
320 DEF FNC$ (N1, R$)
```

Defines the beginning of a multiline user-defined function. Parameters can be any combination of simple numeric or string variables and are local to the function defined. Multiline functions can return string variables up to 32 characters long.

- Local declaration.
- Not executable from the keyboard.
- Not allowed after THEN, ELSE, ON ERROR, or ON TIMER.

(207)

```
[LET] FN numeric variable name = numeric expression
[LET] FN string variable name = string expression
```

```
330 LET FNA = M
340 FNC$ = 'length'
```

In multiline user-defined functions, assigns to the function a value which is passed back to the program. There must be at least one LET FN statement in every multiline function.

- Not executable from the keyboard.

(207)

```
END DEF
```

Defines the end of a multiline user-defined function.

- Local declaration.
- Not executable from the keyboard.
- Not allowed after THEN, ELSE, ON ERROR, or ON TIMER.
- Must be the last statement on a line.

DEF KEY (d..) (143)

```
DEF KEY 'key display character','key definition' [;]
```

```
def key '~','cont';
def key 'J','CLEAR LOOP'
```

Redefines the key or keystroke combination specified by *key display character* as either a typing aid (; included) or an immediate-execute key (; omitted).

- Global declaration.

DELAY (d.) (39)

```
DELAY numeric expression
```

```
delay 3
350 DELAY 0
```

Specifies the length of time, in seconds, that the computer will wait between display lines—accurate to tenths of a second. The range of *numeric expression* is 0 to 2^{26} seconds.

- Global declaration.

DELETE (dele.) (59)

```
DELETE [beginning line number [, ending line number]]
```

```
delete 120
360 DELETE 250, 280
```

Deletes the specified line or block of lines from the current BASIC or text file. DELETE by itself deletes the pending line. If no *ending line number* is specified, the line number specified by *beginning line number* is deleted.

DIM (none) (194)

```
DIM item [,item...]
```

```
370 DIM X(20), Y1(5,10), Z$[80]
```

Declares the maximum length for string variables and declares the upper bounds for one- and two-dimensional numeric arrays. A string variable name must be followed by the length in brackets; a numeric array name must be followed by the maximum subscript values in parentheses (separated by a comma if the array is two-dimensioned).

- Local declaration.
- Not allowed after THEN, ELSE, ON ERROR, or ON TIMER.
- Must be the last statement on the line.
- Must occur before any references to dimensioned variables.

DISP (di.) (166)

```
DISP [display list][;]
```

```
380 DISP X,Y(3,3),Z$[5,10]
390 DISP F7(I2+2,I2+2);' units ',X*Y;A$
```

Evaluates the items in the *display list* and outputs their values to the HP-75 display and any DISPLAY IS devices. Use semicolons as item separators for compact spacing; use commas to produce display fields of 21 characters. A terminating comma or semicolon will suppress the CR/LF that is normally output following the last item. DISP with an empty *display list* outputs a blank line.

DISPLAY IS (di..) (128)

```
DISPLAY IS *
           ''
```

Cancels all display device assignments. The HP-75 display becomes the only display device.

- Local declaration.

(128)

```
DISPLAY IS ':device code [,:device code...]'
```

```
display is ':p2,:tv'
```

Designates specified HP-IL peripherals as display devices.

- Global declaration.

DISP USING (di.us.) (238)

```
DISP USING line number          [;disp using list]
           'image format string'
```

```
400 DISP USING 540; 'NAME',A$,I(J)
410 DISP USING 550
420 DISP USING 560; C, B, M, K
430 DISP USING '3A, 2X, 4D.DD'; T$, K
```

Sends listed items to display devices according to the included image format string or the format string in the IMAGE statement specified by the *line number*.

EDIT (e./e.ba./e.t.) (62)

```
EDIT [ 'filename' [ , BASIC ]
                  [ , TEXT  ]
     [ BASIC ]
     [ TEXT  ]
```

```
edit
edit 'datefile',text
edit basic
```

Moves the file pointer to the first line of a new `workfile` or the specified file. When a new file is created the file type is either the same as the current file or as specified. Displays the catalog entry of the file. Used to create new files.

EDIT KEYS (e.keys) (151)

```
EDIT KEYS
```

Moves the file pointer to the first line of the `keys` file.

END (none) (165)

```
END
```

Terminates program execution and deallocates the program.

- Not executable from the keyboard.

END DEF (end d.) (207)

```
END DEF
```

Refer to `DEF FN`.

ENDLINE (en.) (140)

```
ENDLINE [0-3 character string expression]
```

```
440 ENDLINE CHR$(13)&CHR$(10)&CHR$(10)
450 ENDLINE ''
460 ENDLINE
```

Defines the end-of-line characters (up to 3) sent to `PRINTER IS` devices. `ENDLINE` with a null string (`''`) specifies that output will be in a continuous string with no end-of-line delimiters. `ENDLINE` by itself restores normal CR/LF end-of-line.

- Global declaration.

FETCH (f. / FET) (53)

```
FETCH [ line number
        search string [, line number] ]
```

```
fetch 45
fetch 'int'
fetch 'call',250
```

Fetches the pending line, the specified line, the next line that contains the search string, or—beginning at the specified line—the next line containing the search string. The search string may be any string expression.

(146)

```
FETCH KEY 'single character string expression'
```

```
fetch key 'M'
fetch key chr$(255)
```

Recalls the current definition of the specified key or keystroke combination and makes it available for editing.

FOR (none)

(179)

```
FOR loop counter = initial value TO final value [STEP increment value]
```

```
470 FOR I=1 TO 15 @ DISP I; X5 @ NEXT I
480 FOR J=2*K TO 15*K STEP K
```

Defines the beginning of a FOR-NEXT loop. The *loop counter* must be a simple numeric variable; the *initial, final,* and optional *increment values* may be any numeric expression. The optional STEP parameter specifies the amount by which the *loop counter* is incremented when the loop's NEXT statement is executed.

- Not executable from the keyboard (unless the entire FOR-NEXT loop is contained on one line).
- Not allowed after THEN, ELSE, ON ERROR, or ON TIMER.

GOSUB (gos.)

(182)

```
GOSUB line number
```

```
490 GOSUB 1700
```

Causes program execution to branch to the subroutine beginning at the specified line number.

- Not executable from the keyboard.

GOTO (g.)

(176)

```
GOTO
```

```
500 GOTO 2000
```

Causes program execution to branch to the specified line number.

- Not executable from the keyboard.
- Instructions following this one on a line will never be executed.

IF...THEN...ELSE (none...th....el.)

(177)

```
                                    line number                          line number
IF expression THEN [ allowable statement... or command... ] ELSE [ allowable statement... or command... ]
```

```
510 IF X4 THEN 2350 ELSE RESTORE @ GOTO 200
520 IF W1$[5,6]='LF' THEN C$=10 ELSE C$=0
530 IF X<Y THEN M1=X @ M2=Y ELSE M1=Y @ M2=X
```

Provides conditional branching. When the *expression* evaluates to a nonzero value (true), the THEN part of the statement is executed. Program execution branches to the line number specified or continues with the instructions listed after THEN. When the *expression* evaluates to zero (false), program execution continues at the next line of the program or, if the optional ELSE part is present, the program branches to the statement specified or continues with the instructions listed after ELSE.

- Not allowed after THEN, ELSE, ON ERROR, or ON TIMER.
- Must be the last instruction on the line.
- The following statements are not allowed after THEN or ELSE:

```
FOR        END DEF     ON TIMER
NEXT       IF          OPTION BASE
DATA       IMAGE       REAL
DEF FN     INTEGER     SHORT
DIM        ON ERROR
```

IMAGE (im.) (238)

IMAGE *image format string*

```
540 IMAGE 4A, 3X, 10A, 5D.3D
550 IMAGE 2X, 'RESULTS'//
560 IMAGE DDDcDDD.DDD, 2x, zzz.d, 3x, 2(*****.dd)
```

Specifies the output format for DISP USING and PRINT USING statements. The *image format string* consists of one or more field specifiers separated by commas or slashes.

Image Symbol	Type of Output	Comments	May be Replicated
x, X	Blank	Specifies a blank between items.	Yes
'', ""	Literal text	Used to delimit string items in format strings or entire specifiers in DISP USING and PRINT USING statements.	no
a, A	Character	Specifies a character position; text is left-justified.	yes
d, D	Digit	Specifies a digit position to left or right of radix symbol; leading blanks and trailing zeros.	yes
z, Z	Digit	Specifies digit position to left of radix; leading zeros.	yes
*	Digit	Specifies digit position to left of radix; leading asterisks.	yes
s, S	Sign	Specifies sign, + or −.	no
m, M	Sign	Specifies sign, blank or −.	no
e, E	Exponential notation	Outputs numeric with exponent E, sign, and three digits.	no
.	Decimal point radix	Outputs a decimal point as radix in that position.	no
r, R	Comma radix	Outputs a comma as radix in that position.	no
c, C	Comma separator	Outputs a comma as a digit separator in the specified position.	no
p, P	Period separator	Outputs a period as a digit separator in the specified position.	no
()	Field	Allows enclosed image specifiers to be replicated.	yes
k, K	Compacted item	Causes both strings and numerics to be output with no leading or trailing blanks.	no
/	CR/LF	Causes a carriage-return/line-feed; may also delimit items.	yes

- Local declaration.
- Not executable from the keyboard.
- Not allowed after THEN, ELSE, ON ERROR, or ON TIMER.
- Must be the only instruction on the line.

INITIALIZE (ini.) (132)

```
INITIALIZE ' :device code' [, number of file entries]
```

```
570 INITIALIZE ':CA', 300
initialize ':T1'
```

Prepares the medium in the specified mass storage device to store information and allows the medium to store either 128 files or the specified number of files (up to 453).

INPUT (i.) (168)

```
INPUT variable [, variable...]
INPUT 'prompt'; variable [, variable...]
INPUT 'prompt', prompt string expression; variable [, variable...]
```

```
580 INPUT U3, A$
590 INPUT 'Price';P1
600 INPUT 'Your Name',N$;N$
```

Allows assignments to variables from the keyboard during program execution. The optional second prompt can be overwritten and is read as input.

INTEGER (int.) (194)

```
INTEGER numeric variable [(subscripts)][, numeric variable [(subscripts)]...]
```

```
610 INTEGER S, D(3,5), K(50)
```

Dimensions and reserves memory for integer precision numeric variables. Valid for both simple and array variables.

- Local declaration.
- Not allowed after THEN, ELSE, ON ERROR, or ON TIMER.
- Must be the last instruction on the line.
- If used, must occur before any references to declared variables.

LET (none) (165)

```
[LET] simple variable [, simple variable...] = numeric expression
[LET] string variable [, string variable...] = string expression
```

```
620 LET X,Y(8) = Z^2-3*Z+8
630 A$[5,9] = CHR$(K)&' ips'
```

Assigns a value to one or more variables.

- Local declaration.

LET FN (le..) (207)

```
[LET] FN numeric variable name = numeric expression
[LET] FN string variable name = string expression
```

Refer to DEF FN.

LIST (l.) (56)

```
LIST ['filename' | KEYS] [, beginning line number [, ending line number]]
```

```
640 LIST 200,400
650 LIST 'DROP',150
```

Lists one or more lines of the current or specified file on the display or DISPLAY IS devices.

LIST IO (l..) (127)

```
LIST IO
```

Lists the device codes of the currently assigned HP-IL devices on the display or DISPLAY IS devices.

LOCK (none) (28)

```
LOCK 'password'
```

```
lock 'gandalf'
```

Locks the HP-75 against use without the specified password. LOCK '' cancels the use of a password.

- Global declaration.

MARGIN (ma.) (40)

```
MARGIN number of characters
```

```
margin 25
660 MARGIN LEN(B$)-5
```

Sets the character position at which a beep sounds to signal the end of a line of input.

- Global declaration.

MERGE (m.) (60)

```
MERGE 'filename'[, beginning line number [, ending line number]]
```

```
merge 'program2'
670 MERGE 'TEST',150,200
```

Merges the specified number of lines from the specified file into the current BASIC or text file. (File types must agree.) If no *ending line number* is specified, the file is merged from the *beginning line number* to the end-of-file. If neither line number is specified, the entire file is merged.

NAME (n.) (64)

```
NAME 'filename'
```

```
name 'newname'
```

Renames the current file and creates another workfile of the same type (BASIC or text).

NEXT (ne.) (179)

```
NEXT loop counter
```

```
680 NEXT J
```

Defines the end of a FOR-NEXT loop. Execution is returned to the corresponding FOR statement and the *loop counter* is incremented (by 1 or by the STEP value).

- Not executable from the keyboard.
- Not allowed after THEN, ELSE, ON ERROR, or ON TIMER.

OFF ERROR (of.e.) (259)

```
OFF ERROR
```

Cancels the current ON ERROR declaration.

- Local declaration.
- Not executable from the keyboard.

OFF IO (none) (130)

```
OFF IO
```

Suspends HP-IL communications, but device code assignments are retained in memory.

- Global declaration.

OFF TIMER # (of.t.#) (187)

```
OFF TIMER # timer number
```

```
690 OFF TIMER # 22
700 OFF TIMER # J7
```

Disables the specified timer.

- Global declaration.
- Not executable from the keyboard.

ON ERROR (o..) (258)

```
ON ERROR [allowable statement...] or [command...]
```

```
710 ON ERROR GOSUB 500
720 ON ERROR X2=X2-1 @ RESTORE @ OFF ERROR
```

Specifies the sequence of execution when an error condition occurs, without halting execution.

- Local declaration.
- Not executable from the keyboard.
- Not allowed after THEN, ELSE, ON ERROR, or ON TIMER.
- The following statements are not allowed after ON TIMER.

FOR	END DEF	ON TIMER
NEXT	IF	OPTION BASE
DATA	IMAGE	REAL
DEF FN	INTEGER	SHORT
DIM	ON ERROR	

ON...GOSUB (none) (183)
ON...GOTO (none) (182)

ON *numeric expression* GOSUB *line number* [, *line number*...]
ON *numeric expression* GOTO *line number* [, *line number*...]

```
730 ON F9 GOSUB 3100,3200,3300,3400
740 ON MOD(N5,5) GOTO 3650,3660,3670,3680,3690
```

Causes branching to a statement or subroutine specified in the statement list, based on the value of the expression. The expression must evaluate to values which round to: 1, 2, 3, ..., n.

- Not executable from the keyboard.
- Instructions following ON...GOTO on a line will never be executed.

ON TIMER # (o.t.) (186)

ON TIMER # *timer number* , *seconds* [*allowable statement*...] or [*command*...]

```
750 ON TIMER #21, 3600 RESTORE @ GOSUB 450
760 ON TIMER #K2, 60*M9 DISP 'time' @ BEEP
```

Sets the specified timer (*timer number*) and its interrupt interval (*seconds*) and specifies the sequence of execution when the interrupt occurs.

- Local declaration.
- Not executable from the keyboard.
- Not allowed after THEN, ELSE, ON ERROR, or ON TIMER.
- The following statements are not allowed after ON TIMER.

FOR	END DEF	ON TIMER
NEXT	IF	OPTION BASE
DATA	IMAGE	REAL
DEF FN	INTEGER	SHORT
DIM	ON ERROR	

OPTION ANGLE (op.a.d./op.a.r.) (85)

```
OPTION ANGLE DEGREES
             RADIANS
```

Sets the trigonometric mode to *degrees* or *radians*.

- Global declaration.

OPTION BASE (op..) (193)

```
OPTION BASE 0
            1
```

Specifies the lower bound for numeric arrays.

- Local declaration.
- Not executable from the keyboard.
- Not allowed after THEN, ELSE, ON ERROR, or ON TIMER.
- If used, must occur before any references to array variables.
- Only one OPTION BASE statement is allowed in a program.

PACK (pa.) (137)

```
PACK ':device code'
```

```
pack ':CA'
```

Packs the medium on the specified mass storage device.

PLIST (pl.) (56)

```
PLIST [ 'filename' ]  [, beginning line number [, ending line number]]
      [ KEYS       ]
```

```
770 PLIST F1$
plist 'names'
```

Lists one or more lines of the current or specified file, either BASIC or text. Listing is sent to all PRINTER IS devices or to the display if no PRINTER IS devices are assigned.

POP (p.) (183)

```
POP
```

Cancels the pending return from the most recent GOSUB statement.

- Not executable from the keyboard.

PRINT (pri.) (167)

```
PRINT [print list][;]
```

```
780 PRINT B4$;
790 PRINT 16,T2(4*I),H5$[2,5]
```

Sends listed items to PRINTER IS devices or to the display if no PRINTER IS devices are assigned. PRINT with no print list outputs a blank line.

PRINT # (pri,#) (217)

```
PRINT #file number[,line number];expression[,expression...]
```

```
800 PRINT #6; H$,'days',05
810 PRINT #J7,I+1;D$&E$,120
```

Prints the listed items to the specified file at the next or specified line.

(221)

```
PRINT #file number,line number
```

Moves the file pointer, in the file specified by *file number*, to the specified DATA statement, *deletes* the line, but leaves the file pointer positioned at that line.

- Not executable from the keyboard.

PRINTER IS (pri,,) (128)

```
PRINTER IS ':device code[,:device code...]'
```

```
820 PRINTER IS ':PR'
```

Designates the specified device as a printer device.

- Global declaration.

(129)

```
PRINTER IS *
           ''
```

Cancels all printer device assignments.

- Global declaration.

PRINT USING (pri,us,) (238)

```
PRINT USING line number          ;[print list]
            image format string
```

```
830 PRINT USING 540;'data',D1$,X5
840 PRINT USING "'new sample',10('-')";
```

Sends listed items to PRINTER IS devices according to the included *image format string* or the format string in the IMAGE statement specified by *line number*.

PROTECT (pr.) (121)

```
PROTECT
```

Protects a magnetic card from being overwritten.

PURGE (pu.) (50/137)

```
PURGE [ ' filename '
        KEYS
        APPT
        ' file specifier ' ]
```

```
850 purge 'TEMP:CA'
purge appt
purge 'file12'
```

Erases the current or specified file from memory or the specified file from a mass storage medium and reclaims the space occupied by that file.

PUT (none) (204)

```
PUT one-character string expression
```

```
860 PUT CHR$(172)
```

Simulates the pressing of the corresponding key or keystroke combination. Stores the character code of the *one-character string expression* in the key waiting buffer.

PWIDTH (pw.) (39)

```
PWIDTH number of characters
```

```
870 PWIDTH 10*I
pwidth 24
```

Sets the line length for PRINT and PLIST instructions.

- Global declaration.

RANDOMIZE (ra.) (83)

```
RANDOMIZE [numeric expression]
```

```
880 RANDOMIZE
890 RANDOMIZE 7^(9*Z2)
```

Computes a new random number seed using the system clock reading or the specified numeric expression.

- Global declaration.

READ (none) (211)

```
READ variable name [, variable name...]
```

```
900 READ R$,W3(7)
910 READ V(,)
```

Assigns numeric or string constants in DATA statements to the listed variables. Array variables without subscripts inside the parentheses will have values assigned to all elements of the array.

- Not executable from the keyboard.

READ # (r,#) (219)

```
READ #file number [, line number]; variable [, variable ...]
```

```
920 READ #2,10*X; S4$,M
930 READ #I+1; A,B()
```

Assigns the constants in the next or specified DATA statement, in the data file specified by *file number*, to the listed variables.

- Not executable from the keyboard.

(221)

```
READ #file number , line number
```

Moves the file pointer, in the data file specified by *file number*, to the beginning of the specified DATA statement.

- Not executable from the keyboard.

REAL (re.) (194)

```
REAL numeric variable [( subscripts )][, numeric variable [( subscripts )]...]
```

```
940 REAL X,H7(30),Q5(15,15)
```

Dimensions and reserves memory for real (full-precision) variables. Valid for both simple and array variables.

- Local declaration.
- Not allowed after THEN, ELSE, ON ERROR, or ON TIMER.
- Must be the last instruction on the line.
- If used, must occur before any references to the declared variables.

REM (none) (166)

```
REM [character [character...]]
 !
```

```
950 REM  -DOW subroutine
960 ! Wait loop.
970 LET L = L + 1     !Increment line count.
```

Provides for documenting program listings. All characters that follow the REM statement are treated as comments. The ! may also be used as a comment delimiter. REM must be at the beginning of the line; ! can go anywhere in the line (all characters following ! are treated as comments).

RENAME (ren.) (59)

```
RENAME['old filename']TO 'new filename'
```

```
980 RENAME TO 'OLDATA'
990 RENAME 'THISFILE' TO 'THATFILE'
rename keys to 'keyfil3'
```

Changes the name of the current or specified file in memory.

(137)

```
RENAME 'file specifier'  TO 'file specifier'
                            'filename'
RENAME 'filename' TO 'file specifier'
```

```
1000 RENAME 'FILE20:CA' TO 'FILE30'
rename 'namefile' to 'newnames:tp'
```

Changes the name of the specified mass storage file.

RENUMBER (renu.) (57)

```
RENUMBER[beginning line number[,increment value[,from old line number
  [,through old line number]]]]
```

```
1010 RENUMBER
renumber 100,5
renumber 100,5,10,250
```

Renumbers all or the specified portion of the current BASIC or text file.

RESTORE (res.) (213)

```
RESTORE[line number]
```

```
1020 RESTORE
1030 RESTORE 250
```

Resets the data pointer to the beginning of the specified or lowest-numbered DATA statement in the BASIC file in which the RESTORE statement is located.

- Not executable from the keyboard.

RESTORE # (res.#) (221)

```
RESTORE #file number[,line number]
```

```
1040 RESTORE #3
1050 RESTORE #I+1,L-100
```

Moves the data pointer in the data file specified by *file number* to the beginning of the specified or lowest-numbered DATA statement in the file.

- Not executable from the keyboard.

RESTORE IO (res.) (130)

```
RESTORE IO
```

Restores HP-IL communications subsequent to OFF IO and certain loop errors. Assumes the loop configuration is the same as in the last ASSIGN IO declaration.

- Global declaration.

RETURN (ret.) (182)

```
RETURN
```

Causes the program to branch from the current subroutine to the statement following the branching statement that referenced the subroutine.

- Not executable from the keyboard.
- Instructions following this one on a line will never be executed.

RUN (none) (158)

```
RUN [ line number
      'filename' [,line number] ]
```

```
run 150
run 'money',1000
```

Causes execution, of the current or specified program, to begin at the first or specified *line number*.

SHORT (sh.) (194)

```
SHORT numeric variable [(subscripts)][,numeric variable [(subscripts)]...]
```

```
1060 SHORT B(4,4),D8,C(3)
```

Dimensions and reserves memory for short precision numeric variables. Valid for both simple and array variables.

- Local declaration.
- Not allowed after THEN, ELSE, ON ERROR, or ON TIMER.
- Must be the last instruction on the line.
- If used, must occur before any references to declared variables.

STANDBY (s.) (29)

```
STANDBY OFF
```

Sets the HP-75 to turn off after 5 minutes of inactivity and to limit the number of unsuccessful attempts to communicate with HP-IL devices.

- Global declaration.

```
STANDBY ON
```

Sets the HP-75 to stay on indefinitely and to wait indefinitely for HP-IL devices to respond.

- Global declaration.

STOP (none) (165)

```
STOP
```

Interrupts program execution without deallocating the program.

- Not executable from the keyboard.
- Instructions following this one on a line will never be executed.

TRACE FLOW (t . .) (252)

```
TRACE FLOW
```

Sets the HP-75 to display the source and destination line numbers of each branch in program execution.

- Global declaration.

TRACE OFF (t r . o .) (253)

```
TRACE OFF
```

Cancels all trace operations.

- Global declaration.

TRACE VARS (t r . v .) (253)

```
TRACE VARS
```

Sets the HP-75 to display the line number and variable name (and the values of numeric variables) each time the value of a variable changes.

- Global declaration.

TRANSFORM (t r .) (275)

```
TRANSFORM['filename']INTO  BASIC
                           TEXT
                           LIF1
```

```
transform into basic
1070 TRANSFORM 'DATA10' INTO LIF1
```

Transforms one type of file in memory (BASIC, text, or interchange) into another.

UNPROTECT (u.) (121)

```
UNPROTECT
```

Removes the write-protection from a magnetic card.

WAIT (wa.) (166)

```
WAIT number of seconds
```

```
1080 WAIT 2
1090 WAIT M/60
```

Delays program execution for the specified time.

WIDTH (w.) (39)

```
WIDTH number of characters
```

```
width 32
1100 WIDTH 2*B2
```

Sets the line length for DISP and LIST instructions.

- Global declaration.

Section 4

Reference Tables

Character Set

The CHR$ function returns the display character of a given decimal code, 0 through 255. Arguments are rounded to integer values and converted to the proper range (modulo 256). An underlined display character has a decimal code 128 larger than its non-underlined equivalent.

The NUM function returns the decimal code of the first character of a given character string.

Characters whose decimal codes are 32 through 126 are standard printable characters as defined by the American Standard Code for Information Interchange (ASCII). Refer to the HP-IL peripheral owner's manuals to determine the response of individual peripherals to decimal codes 0 through 31 and 127 through 255. To avoid unexpected device behavior during program listings, use the CHR$( ) function to represent special characters (e.g., the form-feed, or FF, character) rather than the display character itself (e.g., use CHR$(12) instead of μ).

An asterisk (*) indicates that [SHIFT] [I/R] must first be pressed in order to display the character associated with the key or keystroke combination. Where no keystroke sequence is shown, the display character can only be displayed by using the CHR$ command.

Decimal Code	Display Character	Keystrokes
0	△	[CTL] [space bar]
1	▫	[CTL] [A]
2	▒	[CTL] [B]
3	←	[CTL] [C]
4	α	[CTL] [D]
5	β	[CTL] [E]
6	Γ	[CTL] [F]
7	♣	[CTL] [G]
8	BS	[CTL] [H]
9	σ	[CTL] [I]
10	LF	[CTL] [J]
11	∩	[CTL] [K]
12	μ	[CTL] [L]
13	CR	[CTL] [M]
14	↑	[CTL] [N]
15	‡	[CTL] [O]
16	Θ	[CTL] [P]
17	Ω	[CTL] [Q]
18	ä	[CTL] [R]

Decimal Code	Display Character	Keystrokes
128	△	* [ATTN]
129	▫	* [TIME]
130	▒	* [APPT]
131	←	* [EDIT]
132	α	* [↑]
133	β	* [↓]
134	Γ	* [←]
135	♣	* [→]
136	∩	* [I/R]
137	σ	* [FET]
138	⊢	* [DEL]
139	∩	* [CLR]
140	μ	* [LOCK]
141	▮	* [RUN]
142	↑	* [TAB]
143	‡	
144	Θ	
145	Ω	
146	ä	

Decimal Code		Display Character	Keystrokes
19	13	€	CTL S
20	14	π	CTL T
21	15	Ä	CTL U
22	16	ä	CTL V
23	17	Ö	CTL W
24	18	ö	CTL X
25	19	Ü	CTL Y
26	1A	ü	CTL Z
27	1B	ESC	CTL BACK
28	1C	Σ	CTL +
29	1D	≠	CTL =
30	1E	£	CTL ;
31	1F	▒	CTL 8
32	20		space bar
33	21	!	SHIFT 1
34	22	"	SHIFT 2
35	23	#	SHIFT 3
36	24	$	SHIFT 4
37	25	%	SHIFT 5
38	26	&	SHIFT 6
39	27	'	SHIFT 7
40	28	(	SHIFT 8
41	29	)	SHIFT 9
42	2A	*	*
43	2B	+	+
44	2C	,	,
45	2D	-	-
46	2E	.	.
47	2F	/	/
48	30	0	0
49	31	1	1
50	32	2	2
51	33	3	3
52	34	4	4
53	35	5	5
54	36	6	6
55	37	7	7
56	38	8	8
57	39	9	9
58	3A	:	SHIFT ;
59	3B	;	;
60	3C	<	SHIFT ,

Decimal Code		Display Character	Keystrokes
147	93	£	
148	94	π	
149	95	Ä	
150	96	ä	
151	97	Ö	
152	98	ö	
153	99	Ü	
154	9A	ü	
155	9B	←	
156	9C	Σ	
157	9D	≠	
158	9E	£	
159	9F	▒	
160	A0	_	* SHIFT ATTN
161	A1	¡	* SHIFT TIME
162	A2	¢	* SHIFT APPT
163	A3	#	* SHIFT EDIT
164	A4	$	* SHIFT ↑
165	A5	¥	* SHIFT ↓
166	A6	¦	* SHIFT ←
167	A7	§	* SHIFT →
168	A8	¿	* SHIFT I/R
169	A9	⌐	* SHIFT FET
170	AA	ª	* SHIFT DEL
171	AB	«	* SHIFT CLR
172	AC	¬	* SHIFT LOCK
173	AD	¡	* SHIFT RUN
174	AE	«	* SHIFT TAB
175	AF	¿	
176	B0	°	* CTL 0
177	B1	±	* CTL 1
178	B2	²	* CTL 2
179	B3	³	* CTL 3
180	B4	´	* CTL 4
181	B5	µ	* CTL 5
182	B6	¶	* CTL 6
183	B7	·	* CTL 7
184	B8	¸	* CTL 8
185	B9	¹	* CTL 9
186	BA	º	
187	BB	»	
188	BC	¼	

Decimal Code	Display Character	Keystrokes
61	=	[=]
62	>	[SHIFT] [.]
63	?	[SHIFT] [/]
64	@	[SHIFT] [+]
65	A	[SHIFT] [A]
66	B	[SHIFT] [B]
67	C	[SHIFT] [C]
68	D	[SHIFT] [D]
69	E	[SHIFT] [E]
70	F	[SHIFT] [F]
71	G	[SHIFT] [G]
72	H	[SHIFT] [H]
73	I	[SHIFT] [I]
74	J	[SHIFT] [J]
75	K	[SHIFT] [K]
76	L	[SHIFT] [L]
77	M	[SHIFT] [M]
78	N	[SHIFT] [N]
79	O	[SHIFT] [O]
80	P	[SHIFT] [P]
81	Q	[SHIFT] [Q]
82	R	[SHIFT] [R]
83	S	[SHIFT] [S]
84	T	[SHIFT] [T]
85	U	[SHIFT] [U]
86	V	[SHIFT] [V]
87	W	[SHIFT] [W]
88	X	[SHIFT] [X]
89	Y	[SHIFT] [Y]
90	Z	[SHIFT] [Z]
91	[	[SHIFT] [0]
92	\	[CTL] [/]
93	]	[SHIFT] [-]
94	^	[SHIFT] [=]
95	_	[CTL] [-]
96	`	[CTL] [7]
97	a	[A]
98	b	[B]
99	c	[C]
100	d	[D]
101	e	[E]
102	f	[F]
103	g	[G]

Decimal Code	Display Character	Keystrokes
189	=	
190	>	
191	?	
192	@	* [CTL] [ATTN]
193	A	* [CTL] [TIME]
194	B	* [CTL] [APPT]
195	C	* [CTL] [EDIT]
196	D	* [CTL] [↑] ⊤
197	E	* [CTL] [↓]
198	F	* [CTL] [←]
199	G	* [CTL] [→]
200	H	* [CTL] [I/R]
201	I	* [CTL] [FET]
202	J	* [CTL] [DEL]
203	K	* [CTL] [CLR]
204	L	* [CTL] [LOCK]
205	M	* [CTL] [RUN]
206	N	* [CTL] [TAB]
207	O	
208	P	
209	Q	
210	R	
211	S	
212	T	
213	U	
214	V	
215	W	
216	X	
217	Y	
218	Z	
219	[	
220	\	
221	]	
222	^	
223	_	
224	`	* [SHIFT] [CTL] [ATTN]
225	a	* [SHIFT] [CTL] [TIME]
226	b	* [SHIFT] [CTL] [APPT]
227	c	* [SHIFT] [CTL] [EDIT]
228	d	* [SHIFT] [CTL] [↑]
229	e	* [SHIFT] [CTL] [↓]
230	f	* [SHIFT] [CTL] [←]
231	g	* [SHIFT] [CTL] [→]

Decimal Code	Display Character	Keystrokes
104	h	H
105	i	I
106	j	J
107	k	K
108	l	L
109	m	M
110	n	N
111	o	O
112	p	P
113	q	Q
114	r	R
115	s	S
116	t	T
117	u	U
118	v	V
119	w	W
120	x	X
121	y	Y
122	z	Z
123	{	CTL ,
124	|	SHIFT =
125	}	CTL ·
126	~	CTL *
127	←	CTL 9

Decimal Code	Display Character	Keystrokes
232	▯	* SHIFT CTL I/R
233	▯	* SHIFT CTL FET
234	▯	* SHIFT CTL DEL
235	▯	* SHIFT CTL CLR
236	▯	* SHIFT CTL LOCK
237	▯	* SHIFT CTL RUN
238	▯	* SHIFT CTL TAB
239	▯	
240	▯	
241	▯	
242	▯	
243	▯	
244	▯	
245	▯	
246	▯	
247	▯	
248	▯	
249	▯	
250	▯	
251	▯	
252	▯	
253	▯	
254	▯	BYE†
255	▯	

Error Conditions

Occurrences

When an error occurs, the HP-75 sounds the beep, lights the **ERROR** annunciator, and displays an ERROR or WARNING message according to the current DELAY rate. If the error occurs during program execution, the line number of the statement that caused the error is also displayed. Pressing SHIFT FET displays the message again for as long as the FET key is held down. Typing errn RTN returns the identification number of the error or warning. Pressing CLR, ATTN, TIME, APPT, EDIT, FET, ↑, ↓, RTN, or RUN turns off the **ERROR** annunciator and clears the message.

A WARNING condition causes the HP-75 to supply default values and allows execution to continue (unless ON ERROR has been declared). An ERROR condition interrupts program execution at the statement that caused the error. If the interrupted program is also the current file, then the file pointer will be set to the line in which the error occurred; pressing FET and then RTN fetches the line to the display. The program remains initialized until you edit any line or run another program.

---

† When decimal code 254 is assigned to a key, pressing the redefined key will execute the BYE command. Using CHR$ to display character 254 will cause the ▯ character to be displayed.

Mathematical Warnings and Errors

Number	Message and Condition	Default Values (with DEFAULT ON)
1	num too small Number lies between ±EPS.	0
2	num too large • Larger than ±INF. • Larger than SHORT precision allows. • Larger than INTEGER precision allows.	±9.9999999999E499 ±9.9999E99 ±99999
3	COT or CSC inf COT or CSC of $n \times 180°$; n = integer.	9.9999999999E499
4	TAN or SEC inf TAN or SEC of $n \times 90°$; n = odd integer.	9.9999999999E499
5	0^neg Zero raised to a negative power.	9.9999999999E499
6	0^0 Zero raised to zero power.	1
7	no value • The value of a string variable is referenced before the variable has been assigned a value. • The value of a simple numeric variable or a numeric array element is referenced before the variable has been assigned a value. Although a default value will be supplied to the program with DEFAULT ON, the variable value will remain undefined.	'' 0
8	/zero Division by zero.	±9.9999999999E499
(The remaining mathematical errors do not default.)		
9	neg^non-integer Negative value raised to a non-integer power.	
10	SQR(neg number) Square root of a negative number.	
11	arg out of range Argument too large or too small.	
12	LOG(0) Logarithm of zero.	
13	LOG(neg number) Logarithm of a negative number.	

System Errors (14 through 18)	
Number	**Message and Condition**
14	`Low batteries` Replace battery pack or plug in the ac adapter/recharger.
15	`system error` The HP-75 may require a reset.
16	`not enough memory` • Not enough available memory exists to copy a card or tape file to memory. • A file or program requires too much memory. Possible solutions: • Delete one or more lines from the current file. • Execute `CLEAR VARS` to clear calculator variables. • Lessen the precisions or reduce the dimensions of program variables. • Purge one or more files from memory. • Execute `ASSIGN # TO *`, specifying any unused data file pointers. • Execute `CATALL` then press ↑ or ↓ until you see a file that can be purged, then press EDIT and execute `PURGE`. • Execute `ASSIGN IO *` to clear HP-IL loop.
17	`RAM is invalid` • Indicates a defective circuit. Unit may require service. • Indicates memory lost.
18	`ROM missing` Missing a necessary plug-in ROM or LEX file.

Card Reader Warnings (19 through 26)	
Number	**Message and Condition**
19	`write protected` Attempting to copy to a write-protected card. Execute `UNPROTECT` first.
20	`not this file` A track does not belong to the same file that is being copied to memory.
21	`verify failed` The same track requires two more passes through the card reader, once to write to the track and once more to verify. Clean the card first.
22	`unknown card` The information on the track is not recognized by the HP-75.
23	`bad read/write` A failure to read a track or write to the track properly. Clean the card and try again.
24	`pulled too fast` Pull the card through the card reader more slowly.
25	`pulled too slow` Pull the card through the card reader faster.
26	`wrong name` The file specifier in a `COPY` command doesn't match the name of the card file.

Program Errors (27 through 54)	
Number	**Message and Condition**
27	`invalid subscript` • An array subscript is out of bounds. • A string variable substring is out-of-bounds.
28	`record overflow` A random `PRINT #` statement tried to force the data pointer past the end of line.
29	`ON ERROR overflow` An `ON ERROR` routine was requested to handle error 49—`GOSUB overflow`. Rather than create another subroutine and add to the problem, the HP-75 halts execution and displays this message.
30	`OPTION BASE` • Out of range `OPTION BASE` specified. • `OPTION BASE` statement appears after an array variable reference. • Multiple `OPTION BASE` statements in one program.
31	`CONT before RUN` `CONT` is executed or [SHIFT] [RUN] is pressed before a program has been initialized or after a program has been deallocated.
32	`missing line` • Missing a line that is referenced in a `GOTO` or `GOSUB` statement. • Missing a line that is referenced is a `DISP USING` or `PRINT USING` statement.
33	`data type` A `READ` or `READ #` statement tried to read string information into a numeric variable.
34	`no data` • A `READ` or serial `READ #` statement tried to read past the end of file. • A `RESTORE` or `RESTORE #` statement was executed when there was no data in the file. • A `RESTORE`, `RESTORE #`, random `READ #` statement, or random `PRINT #` statement specified an existing but non-`DATA` line in a BASIC file. • A `RESTORE #` or `READ #` statement specified a nonexistent line in a text file.
35	`DIM exist var` Declaring the precision or dimension of a simple numeric variable, string variable, or numeric array variable that has appeared previously in the program.
36	`Invalid DIM` Dimensioning an array improperly after an `OPTION BASE` declaration.
37	`duplicate FN` The same function name appears in two or more `DEF FN` statements.
38	`no END DEF` A multiple-line user-defined function is missing the `END DEF` statement.
39	`FN missing` Attempting to branch into the middle of a multiple-line user-defined function.

Program Errors (27 through 54) (*continued*)

Number	Message and Condition
40	`FN parameter` • Mismatch occurred between the formal parameter list in the function definition and the actual parameter list in the main program. • Attempting to use a user-defined function that isn't defined.
41	`FN calls itself` A user-defined function is defined in terms of itself.
42	`string too long` • Attempting to assign too many characters to a string variable. • Printing a string longer than 251 characters to a BASIC file or longer than 255 characters to a text file. Causes a warning and truncation of the string.
43*	`numeric input` • Attempting to supply characters in response to an `INPUT` statement. • Pressing [RTN] before enough values have been supplied for all the numeric variables in the input List.
44*	`too many inputs` Too many items entered in response to an `INPUT` statement.
45	`missing ASSIGN#` Executing a `PRINT #`, `READ #`, or `RESTORE #` statement without first assigning a file number.
46	`missing NEXT` Missing the `NEXT` statement of a `FOR-NEXT` loop.
47	`no matching FOR` A `NEXT` statement is encountered with no corresponding `FOR` statement. May be caused by an incorrectly nested loop.
48	`FOR overflow` `FOR` nesting exceeds 255 levels.
49	`GOSUB overflow` `GOSUB` nesting exceeds 255 levels.
50	`RETURN w/o GOSUB` A `RETURN` statement was encountered with no pending subroutine condition.
51	`PRINT# to runfile` A program referenced itself in a `PRINT #` statement.
52	`invalid IMAGE` At least one of the specifiers in an `IMAGE`, `DISP USING`, or `PRINT USING` statement is improper. May be caused by an invalid character in the image string.
53	`invalid USING` One of the specifiers in an `IMAGE`, `DISP USING`, or `PRINT USING` statement cannot represent an item to be displayed or printed.
54	`invalid TAB` `TAB` argument rounds to a value less than 1. A warning condition occurs and a value of 1 is supplied.

* An `INPUT` error will cause the HP-75 to prompt again for input. If an `ON ERROR` declaration is in effect at the time of an `INPUT` error, then as many values as possible will be assigned to the `INPUT` variables and the `ON ERROR` instruction will be performed.

HP-IL Errors (55 through 61)

Number	Message and Condition
55	`ASSIGN IO needed` Executing `DISPLAY IS`, `PRINTER IS`, `LIST IO`, `OFF IO`, or `RESTORE IO` without first assigning the devices on the loop.
56	`no loop response` The HP-IL loop is connected, but no devices responded to the `ASSIGN IO` command.
57	`bad transmission` • Hardware error. • Pressing [ATTN] during a loop transmission.
58	`loop timeout` • An intentional or accidental disconnection occurred or power was lost to a peripheral during a loop transmission. • During `STANDBY OFF`, a device took longer than 10 seconds to process a single HP-IL instruction. Make sure the devices are properly connected and powered and then execute `RESTORE IO`. If the disconnection was intentional, connect an HP-IL cable between in and out connectors on the HP-75 and execute `OFF IO`. Execute `STANDBY ON` if a device needs longer than 10 seconds to respond to a command.
59	`too many names` • More names than devices in an `ASSIGN IO` command. Causes a warning; as many devices as exist will be assigned. • More device names in memory than connected devices in the loop were found after a `RESTORE IO` command. Causes an error; no devices are assigned. Connect the original devices and reexecute `RESTORE IO`, or execute a new `ASSIGN IO`.
60	`RESTORE IO needed` The HP-75 tried to issue an HP-IL instruction after an `OFF IO` command.
61	`>31 devices` More than 31 devices, including the HP-75, have been connected in the HP-IL loop.

File and Device Errors (62 through 69)

Number	Message and Condition
62	`file not found` The specified file doesn't exist in memory or on mass storage medium.
63	`invalid filespec` • Invalid name for a file in memory. Restricted to one to eight characters—first character a letter or period, remaining characters letters or digits.. • Invalid name for a mass storage or card file. Restricted to one to eight letters or digits; the first character must be a letter (periods aren't allowed). • Invalid HP-IL device code. Restricted to one or two letters or one letter and one digit. • A `TRANSFORM INTO LIF1` command was executed from a `workfile`. • A display device code is used in a mass storage command.
64	`duplicate name` • Duplicate filename. A file already exists in memory or on a mass storage medium by that name. • Duplicate device code in an `ASSIGN IO` declaration. If `ASSIGN IO` devices are being assigned one by one from the keyboard, causes a warning; the HP-75 prompts for another device code.

File and device Errors (62 through 69) (*continued*)

Number	Message and Condition
65	`access restricted` • Attempting to run an appointment, text, LEX, or LIF1 file. • Attempting to `EDIT`, `LIST`, `COPY`, or `RENAME` a private BASIC file. • Attempting to `PRINT #` to or `READ #` from a private BASIC file, a LEX file, or a LIF1 file. • Attempting to `PRINT #` a numeric value to a text file.
66	`invalid password` • The password for a mass storage or card file must be correctly specified in order to copy the file. • A password is specified when copying a LIF1 file or a file not created by the HP-75 to or from a mass storage medium. Causes a warning; the copy is allowed, but no password is attached to the destination file.
67	`line too long` • A line listed or fetched from a file contains more than three display windows of information. Causes a warning, and the first 94 characters of the line are displayed. If [RTN] is pressed, the fetched line will be truncated at the 94th character, if possible; otherwise, a syntax error will be reported. • Fetching a key definition that exceeds 80 characters. Only the first 80 characters of the definition will be displayed. • Attempting to transform a LIF1 file with excessively long lines into text or BASIC—produces this as a warning and inserts `! ?` immediately after the line number.
68	`wrong file type` • Attempting to rename a non-text file to `keys`. • Attempting to rename a non-appointment file to `appt`. • Attempting to merge one file into a file of a different type. • Attempting to copy an unknown (`?`) mass storage file to memory. • Attempting to copy a text, appointment, LEX, or LIF1 file to a private file; causes a warning and allows a non-private copy to be made.
69	`workfile name?` Attempting to edit another file while editing a non-empty, unnamed workfile. `NAME`, `RENAME`, or `PURGE` the workfile before executing the `EDIT` command or pressing the [EDIT] key during a `CAT ALL`.

Time Mode Warning (70)

Number	Message and Condition
70	`time adjust bad` Attempting to adjust the clock speed by more than ±10%. Causes a warning; the adjustment factor is left at its current setting, and a new adjustment period is begun.

APPT Mode Errors (71 through 77)

Number	Message and Condition
71	`duplicate APPT` Attempting to enter a duplicate appointment. Appointments must differ by at least one character.
72	`day/date mismatch` The day-of-week and the date don't agree.
73	`bad day field` Misspelling the day-of-week.
74	`bad date field` Out-of-range parameter specified for the month, day, or year. May be caused by an illegal character in the `Mo`, `Dy`, or `Yr` fields.
75	`bad time field` Out-of-range parameter specified for the hour or minute. May be caused by an illegal character in the `Hr`, `Mn`, or `AM` fields.
76	`bad rep field` • Invalid Rept (repeating) template caused by an out-of-range parameter or an illegal character. • Pressing a system key while the Rept template is displayed.
77	`bad alarm spec` • Invalid alarm type. Must be a digit, 0 through 9. • Invalid appointment type. Must be `N`, `A`, or `R`.

Syntax Errors (78 through 91)

Number	Message and Condition
78	`syntax` • Incorrect spacing or characters in line. Cursor is positioned to character where error was detected. • A line not understood during a `TRANSFORM INTO BASIC` operation. The error is reported after the entire file has been transformed; unrecognized lines are transformed into program remarks beginning with `! ?`.
79	`; expected` Missing a semicolon between parameters.
80	`) expected` Missing a right parenthesis in an expression.
81	`comma expected` Missing a comma between parameters.
82	`string expected` Unsuccessful attempt to evaluate the expression as a character string.
83	`missing TO` The keyword `TO` with no embedded blanks must be typed in the command.
84	`extra characters` Extra characters appear at the end of the line. May be caused by mistyping an instruction if it is interpreted as a different instruction.

Syntax Errors (78 through 91) (*continued*)	
Number	**Message and Condition**
85	`expr too big` Expression is too big for the HP-75 to evaluate. May be caused by too many nested pairs of parentheses or too many operations in one expression.
86	`illegal context` Statement is not allowed after `THEN`, `ELSE`, `ON TIMER #`, or `ON ERROR`.
87	`bad expression` • A syntax error in an expression; e.g., too many operators between operands. • Attempting to transform a LIF1 file with missing line numbers into text or BASIC. The file remains a LIF1 file.
88	`bad statement` • An incorrect abbreviation. • An incomplete statement. • Attempting to execute a program only statement (e.g., `GOTO`) from the keyboard.
89	`bad parameter` • Using the wrong type of parameter; e.g., using a string argument for a function that takes a numeric argument. • Entering out-of-range parameter, illegal characters, or too many parameters.
90	`bad line number` • Attempting an improper renumbering operation. Causes a warning; default line numbering occurs for the specified portion of the file. • A `PRINT #` statement attempts to create a line number greater than 9999. Causes an error; no printing is done and the data pointer is left at the end of file. • Attempting to specify a starting line number greater than 9999 in an `AUTO` command.
91	`missing parameter` Omitting a necessary parameter from a function, statement, or command.

Mass Memory Errors (92 through 97)	
Number	**Message and Condition**
92	`dev not mass mem` A non-mass storage device was requested to perform a mass storage operation.
93	`mass mem error` The mass storage device is experiencing difficulty, perhaps due to low batteries.
94	`no medium` Load a medium into the mass storage device.
95	`medium full` The medium in the mass storage device has no room for other files. Purge one or more files, pack the medium, or load another medium.
96	`invalid medium` The mass storage device cannot read from or write to the medium. May be caused by an unformatted medium; execute an `INITIALIZE` command for the device. May be caused by a physically damaged or defective medium. May also be caused by dirty recording heads; clean the heads as directed in the peripheral manual.
97	`invalid pack` Interrupting a `PACK` operation. The medium may need to be reformatted with an `INITIALIZE` command.

Alphabetical Listing

Message	Number
access restricted	65
arg out of range	11
ASSIGN IO needed	55
bad alarm spec	77
bad date field	74
bad day field	73
bad expression	87
bad line number	90
bad parameter	89
bad read/write	23
bad rep field	76
bad statement	88
bad time field	75
bad transmission	57
comma expected	81
CONT before RUN	31
COT or CSC inf	3
data type	33
day/date mismatch	72
dev not mass mem	92
DIM exist var	35
duplicate APPT	71
duplicate FN	37
duplicate name	64
expr too big	85
extra characters	84
file not found	62
FN calls itself	41
FN missing	39
FN parameter	40
FOR overflow	48
GOSUB overflow	49
illegal context	86
invalid DIM	36
invalid filespec	63
invalid IMAGE	52
invalid medium	96
invalid pack	97
invalid password	66
invalid subscript	27
invalid TAB	54
invalid USING	53
line too long	67
LOG(neg number)	13
LOG(0)	12
loop timeout	58
low batteries	14
mass mem error	93
medium full	95
missing ASSIGN#	45
missing line	32
missing NEXT	46
missing parameter	91
missing TO	83
neg^non-integer	9
no data	34
no END DEF	38
no loop response	56
no matching FOR	47
no medium	94
no value	7
not enough memory	16
not this file	20
num too large	2
num too small	1
numeric input	43
ON ERROR overflow	29
OPTION BASE	30
PRINT# to runfile	51
pulled too fast	24
pulled too slow	25
RAM is invalid	17
record overflow	28
RESTORE IO needed	60
RETURN w/o GOSUB	50
ROM missing	18
SQR(neg number)	10
string expected	82
string too long	42
syntax	78
system error	15
TAN or SEC inf	4
time adjust bad	70
too many inputs	44
too many names	59
unknown card	22
verify failed	21
workfile name?	69
write protected	19
wrong file type	68
wrong name	26
/zero	8
0^neg	5
0^0	6
>31 devices	61
) expected	80
; expected	79

System Memory Requirements

To determine the size of an initialized program, first interrupt execution (press [ATTN]) and determine available memory (type mem [RTN]). Then deallocate the program by performing a line edit. The amount of additional memory that becomes available plus the size of the deallocated file (as indicated in the catalog entry) is the total size.

Item	Memory Required
Appointments	15 bytes for the appointment file *plus* 7 bytes/appointment *plus* 1 byte/character in the Note field *plus* 5 bytes/repeating appointment.
Data items	5 bytes/DATA statement *plus* 4 bytes/integer *plus* 9 bytes/real *plus* 2 bytes/item *plus* 1 byte/character in strings.
Data pointers	15 bytes/data pointer created by an ASSIGN # statement.
Files (BASIC and text)	15 bytes/file *plus* 3 bytes/line *plus* 1-3 bytes/keyword (BASIC) or 1 byte/character (text).
Files (LEX and interchange)	Refer to the catalog entry for the file.
HP-IL assignments	7 bytes/device declared in an ASSIGN IO command.
Key redefinitions	3 bytes/redefinition *plus* 1 byte/character in the the key definition string *plus* 1 byte for final semicolon.
Mass storage commands (CAT, COPY, PURGE, RENAME)	43 bytes/command.
Mass storage devices	105 bytes/device.
PACK command	256 bytes *plus* 6 bytes/file.
Plug-in ROMs	Refer to the ROM owner's manual for the number of bytes of system RAM used by the ROM.
Program calls	30 bytes/CALL statement *plus* 2 bytes/calling program variable *plus* the number of bytes required by the variables' sizes or precisions.
Timers	63 bytes/timer set by an ON TIMER statement *plus* one or more bytes per timer instruction.
TRANSFORM command	A maximum of 255 bytes while the transformation is occurring.
Variables (both calculator variables and initialized program variables)	
Simple numeric variables	
REAL	12 bytes/variable
SHORT	8 bytes/variable
INTEGER	7 bytes/variable
Numeric array variables	10 bytes/array variable *plus*
REAL	8 bytes/element
SHORT	4 bytes/element
INTEGER	3 bytes/element
String variables	8 bytes/variable *plus* dim length.

Display Escape Codes

ESC represents the escape character, decimal code 27. Displaying [ESC]—that is, [CTL][BACK]—or displaying `CHR$(27)` sends an escape character to the HP-75 display and other `DISPLAY IS` devices. Printing `CHR$` sends an escape character to `PRINTER IS` devices.

The HP-75 display and most HP-IL devices have special responses to predefined *escape codes*, that is, to display or print instructions which contain sequences formed by the escape character and the characters immediately following. A given escape code may be ignored by one device while interpreted by another device as a machine instruction. For example, an ESC `C` is ignored by the HP 82162A Thermal Printer but causes the HP 82163 Video Interface to move the cursor one position to the right. An ESC `C` may be sent to the HP-75 display and other display devices using `DISP CHR$(27)&'C'`, for example.

The HP-75 display responds to 12 escape codes. Other escape sequences used in `DISP` or `PRINT` statements are ignored by the display.

The 32 display window positions are identified as 0 through 31.

Escape Code	Instruction	Description
ESC `C`	Cursor Right*	Moves the cursor one position to the right.
ESC `D`	Cursor Left*	Moves the cursor one position to the left.
ESC `E`	Clear All*	Moves the cursor to column 0 and clears the display.
ESC `G`	Cursor Return	Moves the cursor to column 0.
ESC `H`	Cursor Home*	Moves the cursor to column 0.
ESC `J`	Clear From Cursor*	Clears the display from cursor location.
ESC `K`	Clear to End of Line	Clears the display from the cursor location.
ESC `O`	Delete Character With Wraparound	Deletes the character at the cursor location and left-shifts all trailing characters with wraparound.
ESC `P`	Delete Character	Deletes the character at the cursor location and left-shifts all trailing characters.
ESC `<`	Cursor Off*	Turns off the cursor.
ESC `>`	Cursor On*	Turns on the cursor.
ESC `%cr`	Cursor to Address*	Moves the cursor to the display position (0 through 31) specified by the decimal code of the first display character modulo 32. The decimal code of the second character is used by the video interface as a row parameter, but is ignored by the HP-75. For example, `DISP CHR$(27);'%';CHR$(16);CHR$(8);'here'` positions the cursor to column 16 (and row 8 of the video interface) and displays `here`, beginning from that location.

An asterisk (*) indicates that the HP 82163 Video Interface responds similarly when declared a `DISPLAY IS` device. Refer also to the owner's manual for the video interface.

Type a semicolon after an escape code to suppress the carriage-return/line-feed that normally terminates a DISP or PRINT statement.

In addition, pressing [CTL] [↑] sends an ESC T (roll up one line) to DISPLAY IS devices and pressing [CTL] [↓] sends an ESC S (roll down one line) to DISPLAY IS devices. The HP-75 display window ignores these characters.

Machine Defaults

Condition	After a machine reset or after the battery pack is first installed.	After turning on following a timeout period.
ALARM	ON	Previous setting.
ASSIGN IO	No HP-IL assignments.	Previous HP-IL assignments.
ASSIGN #	No file number assignments.	Previous file number assignments.
BEEP	ON	Previous setting.
[CTL] [FET]	Set-time template.	Blank line.
DEFAULT	ON	Previous setting.
DEF KEY	Identity assignments	Previous key definitions.
DELAY	1 second.	Previous setting.
DISPLAY IS	* (That is, the HP-75 display.)	Previous display devices.
ENDLINE	CR/LF	Previous setting.
ERRL	0	Previous error line.
ERRN	0	Previous error number.
EXACT	EXACT marks cleared.	EXACT marks unchanged.
FETCH	Line 0 of workfile.	Previously pending line.
Files	BASIC workfile (0 bytes, time and date of clock setting.)	Previous BASIC files, text files, appointment files, keys files, LEX files, LIF1 files.
Keyboard	Unshifted.	Previous keyboard mode.
LOCK	No password.	Previous password.
MARGIN	91	Previous setting.
Mode	TIME	EDIT
PRINTER IS	* (The HP-75 display.)	Previous printer devices.
PWIDTH	32 columns.	Previous setting.
RES	0	Last numeric result.
RND	.529199358633	Next number in sequence.
[SHIFT] [FET]	Blank display.	Blank display.
STANDBY	OFF	Previous setting.
STATS template	MDY, ~, AM, YEAR	Previous settings.
TIME display	Set-time values.	Current time.
Trigonometric mode.	OPTION ANGLE RADIANS	Previous setting.
Variables:		
calculator	None.	Previous assignments.
program	None.	Previous assignments if allocated program.
WIDTH	32 columns.	Previous setting.

Abbreviations

The following are the shortest distinct abbreviations recognized by the HP-75. Longer abbreviations are allowed, providing that the period doesn't match the final character of the keyword. Abbreviations may not contain embedded blanks.

The HP-75 supplies the complete spelling of keywords when listing and fetching lines.

Keyword	Abbreviation
ACOS()	ac.()
ALARM OFF	al.off
ALARM ON	al.on
ANGLE()	ang.()
ASSIGN #	as.#
ASSIGN IO	as..
AUTO	a.
BEEP	be.
BEEP OFF	be.off
BEEP ON	be.on
BYE	b.
CAT$()	c.()
CAT ALL	c..
CEIL()	ce.()
CHR$()	ch.()
CLEAR *device code*	cl.':*dc*'
CLEAR LOOP	cl.l.
CLEAR VARS	cl..
COPY	co.
DATA	da.
DEFAULT OFF	defa.off
DEFAULT ON	defa.on
DEF KEY	d..
DELAY	d.
DELETE	dele.
DISP	di.
DISPLAY IS	di..
DISP USING	di.us.
EDIT	e.
EDIT BASIC	e.ba.
EDIT TEXT	e.t.
ELSE	el.
END DEF	end d.
ENDLINE	en.
ERRN	er.
FETCH	f.
FETCH KEY	f..
FLOOR()	fl.()
GOSUB	gos.
GOTO	g.

Keyword	Abbreviation
IMAGE	im.
INITIALIZE	ini.
INPUT	i.
INTEGER	int.
KEY$	k.
LIST	l.
LIST IO	l..
LET FN	le..
LOG10()	lo.()
MARGIN	ma.
MERGE	m.
NAME	n.
NEXT	ne.
OFF ERROR	of.e.
OFF TIMER #	of.t.#
ON ERROR	o..
ON TIMER #	o.t.#
OPTION ANGLE DEGREES	op.a.d.
OPTION ANGLE RADIANS	op.a.r.
OPTION BASE	op..
PACK	pa.
PLIST	pl.
POP	p.
PRINT	pri.
PRINT #	pri.#
PRINTER IS	pri..
PRINT USING	pri.us.
PROTECT	pr.
PURGE	pu.
PWIDTH	pw.
RANDOMIZE	ra.
READ #	r.#
REAL	re.
RENAME	ren.
RENUMBER	renu.
RESTORE	res.
RESTORE #	res.#
RESTORE IO	res..
RETURN	ret.
RUN	r.

Keyword	Abbreviation
SHORT	sh.
STANDBY OFF	s.off
STANDBY ON	s.on
THEN	th.
TIME$	ti.
TRACE FLOW	t.
TRACE OFF	tr.o.
TRACE VARS	tr.v.
TRANSFORM	tr.

Keyword	Abbreviation
UNPROTECT	u.
UPRC$()	up.()
VER$	v.
WAIT	wa.
WIDTH	w.

HP-75 Reference Manual

HEWLETT PACKARD

Corvallis Division
1000 N.E. Circle Blvd., Corvallis, OR 97330, U.S.A.

00075-90004

Printed in U.S.A. 8/82