READ

PURPOSE

This is a very important I/O command that reads data into variables

1. from a mass storage file;
2. from within a CALLed DATAPLOT sub-program;
3. from the terminal.

DESCRIPTION

The rules regarding READ are as follows:

1. If there are k variables listed on the READ statement, then each line image must have at least k data values. Only the first k data values are read (this is how the READ command differs from the SERIAL READ command), and these are read into successive elements of the individual variables. Thus

```
READ X Y
1 1 1
2 4 8
3 9 27
END OF DATA
```

   results in X containing the 3 elements 1, 2, and 3; Y containing the 3 elements 1, 4, and 9; and the values 1, 8, and 27 not being read at all.
2. In scanning for the k variables, the full line image is scanned (for reading from a mass storage file, the full line image is 132 columns; for reading from within a sub-program and for reading from the terminal, the full line image is 80 columns). For variations on this, see the COLUMN LIMITS command.
3. Data values on a line image must be separated by at least one blank or comma.
4. Data values may be free-format--they need not be aligned in specific columns.
5. The format of individual data values is general. It can be integer, floating point, or exponential. However, it is stored internally as a single precision real number. Complex numbers are not supported (but you can read the real and imaginary parts as separate real numbers).
6. By default, all reads start from the beginning of the file (to override this, see the SKIP and ROW LIMITS commands).
7. The analyst need not be concerned about the number of observations for each variable. DATAPLOT automatically determines and reports this value at the end of the read.
8. The read terminates when a line image is encountered which consists of

```
END OF DATA (or END DATA)
```

   or when the end of the file is reached.

SYNTAX 1

READ <x1> <x2> ... <xk>
where <x1>, <x2>, ... <xk> are the desired names for the variables into which the data are read.

This syntax is used to read from the terminal or from within a macro file. All lines are read until an END OF DATA is encountered. An example of this syntax is: READ Y X

SYNTAX 2

READ <file> <x1> <x2> <x3> etc.
where <file> is the name of the mass storage file where the data resides;
and <x1>, <x2>, ... <xk> are the desired names for the variables into which the data in <file> are read.

This syntax is used to read from a file. All lines are read until an END OF DATA or the physical end of file is encountered. An example of this syntax is: READ PONTIUS.DAT Y X

EXAMPLES

```
READ CALIB. PRES TEMP TIME
READ ASTM. Y1 Y2 Y3 X DAY LAB
READ Y1 Y2 Y3 X DAY LAB
```

NOTE 1
By default, DATAPLOT does free format reads. However, it has the capability for supporting Fortran style formats. Formatted reads can be about 10 times faster on many systems which can be helpful for large data files. See the documentation for READ FORMAT for more details.

NOTE 2
Blank lines in data files are ignored. The analyst can purposely insert blank lines to delineate natural chunks of the data.

NOTE 3
DATAPLOT supports the ability to embed comment lines within the data file. See the documentation for COMMENT CHECK (in the Support chapter) for details.

NOTE 4
In order to determine whether the first argument is a file name or a variable name, it looks for a period in the name. If it finds one, it assumes the argument is a file name. If it does not, it assumes the argument is a variable name. If your file name does not contain a period, attach a trailing period (no spaces) to the file name on the READ command.

NOTE 5
DATAPLOT has no restrictions on the file name other than it be a valid file name on the local operating system and that it contain a period "." in the file name itself or as a trailing character. DATAPLOT strips off trailing periods on those systems where it is appropriate to do so. On systems where trailing periods can be a valid file name (e.g., Unix), DATAPLOT tries to open the file with the trailing period. If this fails, it then tries to open the file with the trailing period stripped off.

Some users prefer to give all data files a ".DAT" or ".dat" extension. Although this is a useful method for keeping track of data files, it is strictly a user convention and is not enforced by DATAPLOT in any way.

NOTE 6
File names are case sensitive on Unix file systems. For Unix, DATAPLOT attempts to open the file as given. If this fails, it attempts to open the file as all upper case characters. If this fails, it attempts to open the file as all lower case characters. All other currently supported systems are not case sensitive regarding file names.

As a further caution for Unix hosts, certain expansion characters (specifically ~ to refer to your home directory) are interpreted by the shell and are not recognized by the Fortran compiler. These expansion characters are interpreted as literal characters and do not yield the intended file name.

NOTE 7
DATAPLOT searches for the specified file in the current working directory. If it does not find the file, it tries to find the file in the default DATAPLOT directory. For this reason, all of the DATAPLOT sample data files will be found by simply entering the name with no additional directory information needed. For example,

```
SKIP 25
READ MAVRO.DAT Y
PRINT Y
```

The list of sample data files is given in the DATAPLOT Reference Files chapter. If your implementation of DATAPLOT does not automatically find the sample data files, contact your local site installer.

DEFAULT
1. If no file name is specified in a READ command and a CALL to a macro file is being executed, then the data values should be listed directly in the DATAPLOT sub-program immediately after the READ command (do not forget the END OF DATA statement).
2. If no file name is specified in a READ command and the commands are being manually entered/executed one at a time from the terminal, then the data should be entered directly from the terminal immediately after the READ command (also terminated by an END OF DATA statement).

SYNONYMS
None

RELATED COMMANDS
READ FUNCTION	=	Read a function.
READ MATRIX	=	Read a matrix.

READ PARAMETER	=	Read a parameter.
READ STRING	=	Read a string.
SERIAL READ	=	Perform a serial read.
SET READ FORMAT	=	Define a FORTRAN style format for reads.
DATA (LET)	=	Enter data values into a variable.

APPLICATIONS

Data input

IMPLEMENTATION DATE

Pre-1987

PROGRAM

```
SKIP 25
READ LEW.DAT Y
PLOT Y
```