I

If

The IF statement sets conditions which determine program flow.

Syntax

```
IF <logical expression> [THEN]
    <statement(s)>
[ELSE
    <statement(s)>]
ENDIF
```

Example

```
IF A>B
    Print "Hello"
ENDIF

IF Myfunc(Z) THEN
    A=1
ELSE
    A=2
ENDIF
```

Inc

The INC command increments the value of an integer variable. This is equivalent to but much faster than var=var+1.

Syntax

```
INC <integer variable name>
```

Example

```
INC A
```

Initmenu

The Initmenu command creates a Menu Bar for the application and adds an 'Apple" menu.
The Initmenu command must be executed before any other menu commands will function.

Syntax

```
INITMENU
```

Example
InitMenu

Inkey

The INKEY statement waits for a character to be pressed (when the console is open) and then resumes program execution.

Syntax
INKEY

Input

The INPUT statement allows a user to enter keyboard data into the Console and assign it to a program variable.

Syntax
INPUT <variable name>

Example
INPUT Mystring
INPUT Myint

Note
The variable to which data is Input must be of type Integer,String or Float. Hexadecimal numbers may be Input in the form &H12FA or &FFFF.

Inputbox(

The Inputbox function displays a Modal Dialog box on the screen with the given prompt displayed within it. The user may enter string data into the InputBox and this is returned to the program once the OK button is pressed.

Syntax
INPUTBOX(<string expression>)

Example
Response=InputBox("Enter a string")

Input#

This statement reads data from the file specified by the file_number parameter. The data read must be of the Integer or String data types.

Syntax
INPUT #<file_number>,<variable list>

Example
INPUT #1,Myint,Mystr

Note
Input# is normally used to read data written with the WRITE# statement.

Instr(

Searches to see if a character string is present within another string and returns its position.

Syntax
INSTR(<string expression>,<string expression>)

Example
A=INSTR("ABCDEF","CD") .. would set A to 3.

Note
If the string is not present then zero is returned.

Integer(

The INTEGER(function performs type conversion into INTEGER datatype format.

Syntax
INTEGER(<expression>)

Example
I=INTEGER(W)

Note
If the <expression> yields an Byte or Word result then it is extended
to 32 bits by the INTEGER(function.
If the <expression> yields a Float result then it is truncated to an Integer.
If the <expression> yields a Char, Str255 or String result then the result will be equivalent to the numeric value of the string.

L

Left(

This function returns a string made up of the leftmost n characters of a string expression.

Syntax
LEFT(<string expression>,<length expression>)

Example
S=LEFT("ABCDEF",3) .. S is set to "ABC"
S=LEFT ("AB",6) .. S is set to "AB"

Len(

The LEN(function returns the length (number of characters) of a String type variable.

Syntax
LEN(<string expression>)

Example
I=LEN("ABCD") .. I is set to 4

Let

The LET statement assigns the value of an expression to a variable.

Syntax
[LET] <variable name> = <expression>

Note
The LET keyword is optional

Example
LET A=2
MyAry(I)=Val(S)

Line Input#

The LINE INPUT# statement reads a string of characters from a file, terminated by a carriage return, and assigns them to a string variable.

Syntax
LINE INPUT #<file_number>,<string variable>

Example
LINE INPUT #1,Mystr

Note
LINE INPUT# may be used to read data from a standard TEXT file or from a file written using the PRINT# statement.

Local

The Local statement declares variables which are local in scope. The Local statement may only be declared within a FUNCTION or PROCEDURE construct.

Syntax
LOCAL <variable name> AS <variable type> [<string length>] [<dimensions>]

Example
LOCAL F AS FLOAT.
See GLOBAL for other examples.

Lpeek(

The Lpeek (Long Peek) function returns the 32 bit value stored in a given memory location.

Syntax
LPEEK(<integer value>)

Example
Myint=LPEEK(10538)

Lpoke

The Lpoke command places a 32 bit value into a given memory location. Use with care.

Syntax
LPOKE ,<integer memory address>,<integer value>

Example

LPOKE 10538,A

M

Mid(

This function returns <length expression> characters from <string expression> starting at the <start expression> character.

Syntax

MID(<string expression>,<start expression>,<length expression>)

Example

```
S1=MID("ABCDEF",2,3)          .. S1 is set to "BCD"
S2=MID("ABC",3,6)               .. S2 is set to "C"
```

Msgbox(

The MsgBox function displays a message to the user inside a Modal Dialog Box.

Syntax

MSGBOX(<string expression>,<string expression>,<string expression>)

Example

WhichButton=MsgBox("Are you shure","Yes","No")

Note
The first parameter specifies the string which will appear in the Message Box.
The second parameter specifies the caption on the default button within the Message Box.
The third parameter specifies the caption on the second button within the Message Box. If a null string is supplied to this parameter then the second button will be invisible.
The value returned from MsgBox is an Integer which specifies which button was pressed, 1 for the default button, and 2 for the second button.

N

Next

The Next statement terminates a FOR / NEXT loop.

Syntax
NEXT <variable name>

Example
See FOR.

Not

Performs a logical NOT upon a conditional expression.

Syntax
NOT <conditional expression>

Example
If NOT A>B THEN

O

Open

Opens a disk file.

Syntax
OPEN <filespec> [FOR <mode>] AS [#]<file number> [LEN=<record length>]

Example
OPEN "Myfile" FOR OUTPUT AS #1
OPEN "Randfile" FOR RANDOM As #2 LEN=10

Note
The filespec may be given as a single filename (which must exist in the current folder)
or as a full path name.
The following modes are permissible :
INPUT for reading only, this is the default.
OUTPUT for writing only, an existing file of the same name will be
overwritten if it exists.
APPEND appends to an existing file. The file must exist.
RANDOM for random input/output. See PUT and GET for examples of its use.
The file number may be any integer and is used to uniquely identify the open

file.
The LEN parameter applies only to files opened in RANDOM mode. It specifies the length of each record.
When unsucessful, the OPEN command sets the ERR variable.

Open Console

Opens the console window.

Syntax
OPEN CONSOLE

Note
The console is automatically opened whenever something is written to the console using the PRINT statement.

Or

Performs a logical OR on 2 conditional expressions or a bitwise OR on two integer expressions.

Syntax
<conditional expression> OR <conditional expression>
<integer expression> OR <integer expression>

Example
IF A>B OR C>D
B=C AND &H00FF

P

Peek(

The Peek function returns the 8 bit value stored in a given memory location.

Syntax
PEEK(<integer value>)

Example
Myint=PEEK(10538)

Poke

The Poke command places a 8 bit value into a given memory location. Use

with care.

Syntax

POKE ,<integer memory address>,<integer value>

Example

POKE 10538,A

Parameter

The Parameter statement defines parameters passed into a Function or Procedure.

Syntax

PARAMETER <parameter name> AS <data type> [BYREF]

Example

```
PROCEDURE Myproc(A,B,C)
PARAMETER A AS INTEGER
PARAMETER B AS STRING
PARAMETER C AS INTEGER BYREF
```

Note

When a parameter is marked as BYREF, it must also be declared as BYREF(in the calling statement. A BYREF parameter may have its value changed during execution of the function / procedure. All other parameters may not.

```
Do Myproc(X,BYREF(Y))
........
........
FUNCTION Myproc(P,Q) RETURNING INTEGER
PARAMETER P AS INTEGER
PARAMETER Q AS STRING BYREF
     Q="Changed String"
RETURN 99
```

Precision

The Precision command sets the number of digits, after the decimal point, which will be present when a Float type variable is converted to a String, for example to be used in a Print statement, or explicitly converted using the Str(function.

Syntax

PRECISION <integer expression>

Example

```
Precision 4
MyFloat=1.44444444
Print MyFloat

.. will print 1.4444
```

Note
The default value for Precision is 2.

Print

The print statement outputs data to the console.

Syntax

```
PRINT <expression>[;][,][<expression>] ...
```

Example

```
PRINT A
PRINT B;C,D
```

Note
Placing a semi-colon after an expression will suppress the automatic newline which is the default for the Print command. A comma will insert a Tab character.

Print#

The PRINT# statement outputs character data to a disk file.

Syntax

```
PRINT #<file_number>,<expression>[;][,][<expression>] ...
```

Example

```
Print #1,Mystr
Print #2,Myint;Mystr,Myint2
```

Procedure

The Procedure statement declares the start of a user defined procedure.

Syntax

```
PROCEDURE <name>(<parameter list>)
```

Example

PROCEDURE Myproc(A,B,C)

Declares the start of a procedure, Myproc, which receives 3 parameters (defined
by the PARAMETER statement).

Put

The PUT statement will write a record to a file which has been opened in RANDOM mode.

Syntax
PUT #<file_number>,<record_number>,<variable_name>

Example
PUT #1,123,Mystr

Note
The variable from which the data is written may be a STRING or a STRUCTURE type.
Take care to specify the record_number correctly as the file will be extended to occupy the highest record_number written.

R

Redim

The Redim statement changes the size of an array to have dimensions which are greater or smaller than the original.

Syntax
REDIM <array name> AS <integer expression>

Example
REDIM Myarray As 50

Note
Redim changes the size of the last array dimension declared. ie.
GLOBAL Myarray AS Integer(3,10) ... followed by
REDIM Myarray As 20 ..
would result in an array of size (3,20) will all data preserved.

Repeat

The section of the program between Repeat and Until is executed until the conditional expression following the Until command is fulfilled.

Syntax

```
REPEAT
```

Example

```
I=0
REPEAT
    INC I
    PRINT I
UNTIL I=10          ... will print the numbers 1 through 10.
```

Return

Specifies a Return from either a Procedure or a Function.

Syntax (Procedure)

```
RETURN
```

Syntax(Function)

```
RETURN <expression>
```

Note
The expression following the RETURN statement for a function must result in the same data type as specified in the RETURNING parameter in the Function statement.

Right(

This function returns a string made up of the rightmost n characters of a string expression.

Syntax

```
RIGHT(<string expression>,<length expression>)
```

Example

```
S=RIGHT("ABCDEF",3)          .. S is set to "DEF"
S=RIGHT ("AB",6)               .. S is set to "AB"
```

S

Sgn(

The SGN function determines whether the result of an expression is negative, zero or positive.

Syntax

SGN(<numeric expression>)

Example

```
PRINT SGN(-6)          .. prints -1
PRINT SGN(0)             .. prints 0
PRINT SGN(99.4)        .. prints 1
```

Str(

Converts an Integer or Float data type into a string.

Syntax

STR(<numeric expression>)

Example

```
PRINT STR(123)       ... would print the string "123"
```

String(

The String function converts data from other data types into the String data type.

Syntax

STRING(<expression>)

Example

```
I=STRING(A+B)
```

Note
If the expression yields a Byte result then the resulting string will contain the ascii representation of the value of the Byte.
If the expression yields A Word ,Integer or Float result then the resulting string will contain the string representation of the expression. See STR(.
If the expression yields a Char or Str255 result then this will be converted to a String data type.

Str255(

The Str255 function converts data from other data types into the Pascal String data type.

Syntax

STR255(<expression>)

Example

I=STR255(A+B)

Note
If the expression yields a Byte result then the resulting Pascal String will contain the ascii representation of the value of the Byte.
If the expression yields A Word, Integer or Float result then the resulting Pascal String will contain the string representation of the expression.
If the expression yields a Char or String result then this will be converted to a Pascal String data type.

Sub

The SUB command performs the subtraction function on two operators.

Syntax

SUB <integer variable name>,<numeric expression>

Example

```
SUB Myvar,10      -- Subtracts 10 from variable Myvar
SUB A,B             -- Subtracts variable B from A
```

T

True

True, when used in an expression yields a result of -1 and is used as a documentation aid.

Syntax

TRUE

Example

If A=True

U

Until

The Until statement terminates a REPEAT/UNTIL loop.

Syntax
UNTIL <expression>

Example
See Repeat.

Upper(

The Upper function translates a String type expression into Upper Case.

Syntax
UPPER(<string expression>)

Example
S1=UPPER("abc123Xyz) ... sets S1 to "ABC123XYZ"

V

Varptr(

The Varptr function returns the address in memory of a given variable.

Syntax
VARPTR(<variable name>)

Example
Myint=VARPTR(Mystr)

Val(

The Val function returns the integer value of the leading portion of a string expression.

Syntax
VAL(<string expression>)

Example
Myint=VAL(Mystr)

Note
Use FLOAT to convert a string into a Floating Point value.

Val?(

The Val? functions returns the number of characters which would make up the Integer portion of a string expression.

Syntax

VAL?(<string expression>)

Example

I= Val?("-123XYZ") .. would set I to 4 (-123)

W

Write#

This statement writes the value of program variables to a disk file.

Syntax

WRITE #<file_number>,<expression list>

Example

Write #1,Myint,Mystring

Note

Write# is normally used to write data to be read with the INPUT# statement.