

A brief summary of IDL commands to start with. **Notes**

- IDL is not case sensitive, but for clarity built-in commands are uppercase, and variable names in lowercase
- To split commands over several lines, use the \$ symbol
- To put several commands on the same line, separate using the & symbol

Routines

IDL routines come in two types: Procedures and functions. Procedures are declared using

```
PRO pname, <variables>
  <commands>
END
```

and called using:

```
IDL> pname, <variables>
```

Functions are declared and called using:

```
FUNCTION fname, <variables>
  <commands>
END
```

```
IDL> result = fname(<variables>)
```

In both cases, <variables> is a comma-separated list of arguments for the routine. **NOTE:** If a variable is passed to a routine, it can be modified inside that routine. IDL is pass-by-reference, like FORTRAN and unlike C.

Printing

The PRINT command is followed with a comma-separated list of values or expressions:

```
IDL> PRINT, "Hello World", 53.2, 10
IDL> PRINT, 2.*!PI
```

For information on defined variables and functions, try

```
IDL> HELP
IDL> HELP, variable
```

To print to a file rather than the terminal, there is the PRINTF command, which has a file number as the first argument:

```
IDL> OPENW, lun, "output.txt", /GET_LUN
IDL> PRINTF, lun, "Hello, World!"
IDL> CLOSE, lun
```

Arrays

Arrays can be created by concatenating values (and other arrays) of the same type:

```
IDL> a = [1,2,3,4]
IDL> a = [7, a, 9, 2, a]
IDL> c = ["Hello", "World!"]
```

Another way is to use the functions INTARR and FLTARR which can create multi-dimensional arrays. The functions INDGEN and FINDGEN create arrays filled with values so that `array[i] = i`

See the manual page on WHERE.

Expressions

Variables and numbers can be combined using the following operations to calculate either numbers or Booleans (True / False) for conditional statements.

- **Infix operations** ($a <op> b$): add (+), subtract (-), multiply (*), divide (/), raise to power (^) and modulus (MOD, the remainder of a / b)
- **Comparisons** ($a <op> b$) which give a Boolean result: True if equal ($a EQ b$), greater than (GT), less than (LT), greater than or equal (GE) and less than or equal (LE).
- **Boolean operations** to combine comparisons: NOT, AND, OR
- **Brackets** work in the way you'd expect, so expressions like $((a GT 4) AND (a LT 3*(b+2)))$ are allowed.
- **Built-in functions** include trigonometric functions (in radians) SIN, COS, TAN and their inverses ASIN, ACOS, and ATAN. Logarithms are ALOG (base e) and ALOG10 (base 10). Random numbers can be generated using RANDOMU and RANDOMN. Statistics functions include MEAN, STDDEV (for standard deviation), and C_CORRELATE (cross-correlation).

The built-in help is started by entering:

```
IDL> ?
```

Conditionals

The IF command has the syntax

```
IF <condition> THEN <command>
```

or to run a whole set of commands:

```
IF <condition> THEN BEGIN
    <commands>
ENDIF
```

There is also the ELSE command which allows multiple conditions. See the manual for details. See also the SWITCH statement.

Loops

FOR loops run a variable (in this case i) over a range which is known beforehand:

```
FOR i=min,max DO <operation>
```

```
FOR i=min,max DO BEGIN
    <commands>
ENDFOR
```

If you don't know how many times a loop needs to repeat, use:

```
REPEAT BEGIN
    .
ENDREP UNTIL <condition>
```

where <condition> is a boolean expression. Loops until the given condition is true.

Plotting

The basic plot command is:

```
IDL> PLOT, y
```

where y is an array of values to plot. To use a given x axis, use:

```
IDL> PLOT, x, y
```

To give the graph titles, there are the optional arguments title, xtitle and ytitle e.g.:

```
IDL> PLOT, x, y, title="My plot", $
    xtitle="Time [s]", $
    ytitle="Data [units]"
```

To create plots of 2D data, there are the SURFACE and CONTOUR commands.