The Davinci Cheat Sheet

(For more information, function definitions, and installation help, see http://davinci.asu.edu/wiki)

Data Types

Numerics: Byte, short, int, float, double

```
a = 1                    # Signed integer
b = 2.0                  # Float
c = byte(255)            # Byte(), short(), double()

byte(256) == 255         # Truncation, not wrapping
byte(-1) == 0
short(-50000) == -32768
```

Strings and Texts:

```
str = "mary had" + "a little lamb"             # Creation
str[1:4] == "mary"                             # Addressing

text = cat("its fleece", "was white", axis=y)  # Creation
text[5:9, 2] == "white"                        # Addressing

files = syscall("ls -l *.png")                 # Read STDIN
```

Structures:

```
obj = { first = 1, second = 2.0 } + { third = "three" }  # Creation
obj[3] == "three" && obj.third == "three"                # Addressing

anon = { 4, 5.0, "six", obj }             # Anonymous creation
anon[4][3] == "three"                     # Nested addressing

insert_struct(obj, value="8.9", name="foo", before="second")
val = remove_struct(obj, name="first")

keys = get_struct_keys(obj)               # Text containing key names
get_struct(obj, keys[,3]) == "three"
```

Operators

```
+  add                                  <   less than
-  subtract                             >   greater than
*  multiply                             <=  less than or equal
/  divide                               >=  greater than or equal
%  modulo                               ==  equal
=  equivalence                          !=  not equal
^  exponent                             ||  logical OR
                                        &&  logical AND
[]     range
//     concatenation
where  partial replacement
```

```
quad = (-b + sqrt(b^2 - 4*a*c)) / (2*a)
lexical = "one" < "two" && "two" > "three"
array = 1 // 2 // 3                        # 3x1x1 array of ints
image[where stddev(image) > 3] = 0
```

Flow Control

If/else, for, while:

```
for ( init ; condition ; increment ) {
     if (condition) {
          break;
     } else {
          while (condition) {
               continue;
          }
     }
}
```

User-Defined Functions

```
define name([namedArg,namedArg2,...] [numargs, maxargs]) {
     $1 == ARGV[1]
     $ARGC == length($ARGV)
     if (HasValue(namedArg1)) {
          ...
     }
     return(0)
}

edit(name)              # Edit an existing function
edit("name.dv")         # Edit (and execute) a file
```

Math Functions

Transcendental

`acos() / cos()`
(arc)cosine, radians

`asin() / sin()`
(arc)sine, radians

`atan() / tan()`
(arc)tangent, radians

`cosd(), sind(), tand()`
`acosd(), asind(), atand()`
As above, in degrees

Statistics

`min(), max(), avg(),`
`median(),sum(), stddev()`
Basic statistics

`moment()`
Compute several statistics values

`moments()`
Structured version of moment()

`histogram()`
Compute a histogram

`entropy()`
Compute entropy

Rounding

`ceil() / floor()`
Truncate up/down to nearest integer

`round()`
Round to nearest integer

Logarithms

`log() / log10()`
Base 2/10 logarithms

`exp() / pow()`
Exponential / power functions

Matrix

`mxm()`
Matrix multiply

`minvert()`
Matrix invert

Command-Line Options

```
-w              # Don't use X windows
-f filename     # Read commands from file
-l filename     # Redirect log file
-e 'cmd'        # Execute given string
-vN             # Set verbose level
-q              # Quick start, don't read history or .dvrc
-H              # Force loading of history
-V              # Show version information
--              # Last option (everything else is passed to script)

Example:
    davinci -qwe 'write(hstretch(read($1)), $1, png)' file.png
```

History and Editing

```
Command-line editing (emacs mode)
    ^a/^e : Move to beginning/end of line
    ^b/^f : Move back/forward one character
    ^p/^n : Move back/forward one line in history
    ^u/^k : Kill before/after cursor
    ^y    : Paste killed text
    tab   : Complete / show completions
    ^_    : Undo
    ^x(   : Start macro
    ^x)   : End macro
    ^xe   : Execute macro
    ^c    : Interrupt the current process
```

Plotting w/ Gnuplot

```
xplot(data1, "title 'foo' with points linetype 3", Xaxis=my_axis)
plot("set nokey")                       # Turn off plot key
plot("set xrange [0:1]")
plot("set term postscript color")       # These three
plot("set output 'myfile.ps'")          # lines create
plot("replot")                          # a postscript file
plot("set term x11")
```

Functions, cont'd.

Filtering

`convolve()`
Sliding-window kernel convolution

`boxfilter()`
(Fast) uniform mask convolution and stats

`window( type=[min,max,median] )`
Compute windowed statistics

Strings

`basename() / dirname()`
Filename manipulations

`strlen()`
Returns length of string(s)

`strstr()`
Find first occurrence of a substring

`strsub()`
String substitution using regex

`grep()`
String finding using regex

I / O Commands

`read()`
Read standard image format

`read_text()`
Read ASCII file into text

`ascii( filename, x, y, z, format, column, row, delim)`
Read ASCII file

`load_raw(filename, x, y, z, org, format, header)`
Read binary file

`load_specpr(filename, record)`
Read a SpecPR file

`load_PDS(filename, data=0|1)`
Read a PDS file, or just a header

`load_fits, load_vanilla(), isis()`
Misc.

`write(), write_fits(), write_pds()`
Misc.