MATLAB Quick Reference Guide – Intro to Programming (Part 1)

Script File Names

- Script name cannot start with a number:
  - `33abc.m` Not Allowed
  - `abc33.m` Allowed
- Script names cannot contain spaces or periods, but hyphens or underscores are OK:
  - `abc 33.m` Not Allowed
  - `abc.33.m` Not Allowed
  - `abc_33.m` Allowed
  - `abc-33.m` Allowed
- Do not use names of existing Matlab functions:
  - `sin.m` Not Allowed
  - `sin_func.m` Allowed

Housekeeping

Place these commands at the beginning of your scripts:

```
clear          clears all variables from memory
close all      close all figure windows
clc            clear command window
```

Suppressing Output

Placing a semicolon at the end of a line of code will prevent the results from being displayed to the screen.

```
a = 4;         does not print value to command line
b = 6          this does print the value of b
```

Comments

The "%" sign may be used to document your code

```
x0 = 3;          % initial value of x
```

Wrap Long Lines

Use three periods to continue a long ling onto the next line.

```
x = a0 + a1 * h + (1/2) * h^2 + ...
    (1/6) * h^3;
```

Input

Prints a message to the command line asking the user to enter a number.

```
N = input('enter N:  ');
```

Formatted Output

```
fprintf(FORMAT, variables)
```

Prints the variables to the command line based on formatting specified in the `FORMAT` string:

`%i`	integer	`%e`	scientific (exponential)
`%f`	floating point	`%s`	string
`%g`	tries to use most concise, easy-to-read format		

Special characters:

`\t`	horizontal tab	`\n`	new line

Formatted Output Examples

```
a = 3
fprintf('a = %i\n', a)      prints:  a = 3
fprintf('a = %4i\n', a)     prints:  a =    3
fprintf('a = %04i\n', a)    prints:  a = 0003

b = sqrt(2)
fprintf('b = %g\n', b)      prints:  b = 1.41421
fprintf('b = %.3g\n', b)    prints:  b = 1.41
fprintf('b = %.3g\n', b)    prints:  b = 1.4142136
fprintf('b = %.3f\n', b)    prints:  b = 1.414
fprintf('b = %.3e\n', b)    prints:  b = 1.414e+00
```

Reading a Data File

If the data file contains only columns of number and each column has the same number of rows, then use the `load()` command to read the contents into a matrix we call `data`. Individual columns may then be extracted from the data matrix:

```
data = load('file.txt');
a    = data(:,1);          % column 1
b    = data(:,2);          % column 2
```

If the data file contains a mixture of numbers and text, then use the `textread()` command. Here's an example with three columns: the first is text, the second two are numbers. In this case, the data are read directly into the individual arrays `txt`, `a` and `b`.

```
[txt,a,b] = textread('file.txt','%s %f %f');
```

This example shows how to skip 3 header lines:

```
[txt,a,b] = textread('file.txt',. . .
   '%s %f %f','headerlines', 3);
```

Saving Workspace Variables

To save all variables in your workspace either click "Save Workspace" under the "Home" tab. Or use the `save()` command. The saved file will have extension `.mat`.

```
save('my_variables');
```

To save just two variables `x` and `y` use (note the variable names must be in single quotes):

```
save('my_variables', 'x', 'y');
```

Loading Saved Workspace Variables

To load saved variables back into memory simply double-click on the `.mat` file within Matlab. You may also use the load() command:

```
load('my_variables.mat');
```