lecture_05

January 26, 2017

1 Good coding habits

1.1 naming folders and files

Stanford file naming best practices

1. Include information to distinguish file name e.g. project name, objective of function, name/initials, type of data, conditions, version of file,
2. if using dates, use YYYYMMDD, so the computer organizes by year, then month, then day
3. avoid special characters e.g. !, #, $, ...
4. avoid using spaces if not necessary, some programs consider a space as a break in code use dashes - or underscores _ or CamelCase

1.2 Commenting your code

Its important to comment your code to mention what a variable's units are, what the function is supposed to do, etc.

```
In [1]: function i=code(j)
            % Example of bad variable names and bad function name
            for w=1:j
                i(w)=w;
            end
        end
```

```
In [2]: help code
```

```
'code' is a command-line function

 Example of bad variable names and bad function name


Additional help for built-in functions and operators is
available in the online version of the manual.  Use the command
'doc <topic>' to search the manual index.

Help and information about Octave is also available on the WWW
at http://www.octave.org and via the help@octave.org
mailing list.
```

1.3 Choose variable names that describe the variable

```
In [7]: function count_vector=counting_function(max_value)
            % Good variable names and better help documentation
            %
            % counting function creates a vector from 1 to max_value where each index, i, is
            % stored in each vector spot
            for i=1:max_value
                count_vector(i)=i; % set each element in count_vector to i
            end
        end
```

```
In [6]: help counting_function

'counting_function' is a command-line function

 Good variable names and better help documentation

 counting function creates a vector from 1 to max_value where each index, i, is
 stored in each vector spot


Additional help for built-in functions and operators is
available in the online version of the manual.  Use the command
'doc <topic>' to search the manual index.

Help and information about Octave is also available on the WWW
at http://www.octave.org and via the help@octave.org
mailing list.
```

1.4 Putting it all together

1. Clone your homework_1 to your computer
2. open Matlab (cli, jupyter or gui)
3. Change working directory to homework_1 *e.g.* Windows:`cd('C:\Users\rcc02007\Documents\Github\homework_1')`, Mac: `cd('/Users/rcc02007/Documents/Github/homework_1')`
4. You have already created your first script `myscript.m` (if not see lecture_4)
5. Run `>> my_script.m`
6. Create a new m-file called nitrogen_pressure.m
7. Create a function based upon the ideal gas law for nitrogen, Pv=RT
    1. R=0.2968 kJ/(kg-K)
    2. inputs to function are v (specific volume m^3/kg), and T, temperature (K)
    3. output is P, pressure (kPa)
8. Once the function works, commit the change to the repository (add a message, like 'added file nitrogen_pressure.m'
9. After file is 'committed', 'push' the changes to your github account

for the command-line git user, this is steps 8 and 9: 1. `$ git add *` 2. `$ git commit -m 'added file nitrogen_pressure.m'` 3. `$ git push -u origin master` Username for `'https://github.uconn.edu':rcc02007 <enter>`

In [ ]: