OMIS 30: Intro to Programming (with Python)

Week 1, Class 2

Introduction to Programming

Instructor: Denis Vrdoljak

Goals for the week

- Cover Intros and Intro Material
- Get your tools and environment set up
- Get familiar with the command line

By the end of this, week you should:

- Have Python installed and your IDE set up
- Be able to write a Hello World program in Python, and run it from the command line

Command Line

- Why learn command line?
- Used as the basic interface on servers & virtual machines (cloud)
  - Uses less resources (memory/storage) than a graphical interface
  - Less network traffic transmitted back and forth from the server to your computer
- Important to know to navigate on the servers
- Can do a: **man <command>** or **help <command>** to see the help page on that command
- Dos/Windows vs Linux/Unix: https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/4/html/Step_by_Step_Guide/ap-doslinux.html

Command Line Commands

- Navigation
  - **ls** - shows the contents of the present folder (**dir** for windows)
    - **ls -alh** - anything after a hyphen is called an option, flag, or switch & modifies the original command
      - a = all files including hidden ones (ones that start with a .)
      - l = long list
      - h = human readable (so a size of 4096 bytes = 4.0K instead)
  - **cd <dir>** = change directory
    - **cd ..** = back one directory
    - **cd .** = this directory
  - **pwd** = print working directory - the full directory name of your current directory (**cd** for windows)

Command Line Commands

- Making files & directories
  - **touch <file_name>** = make a blank file of that name (N/A on windows)
  - **mkdir <directory_name>** = make a new directory of that name
- Deleting
  - **rm <file or directory name>** = removes / deletes that file or directory
    - CAUTION: if you remove it it is gone (no ‘recycling bin’ etc to recover it from!)
    - **rm -r <directory_name>**= removes all the directories under that directory name

Command Line Commands

- Writing to the command line
  - **echo <message>** = print that message back to the screen
  - **clear** = clears the screen (**cls** in windows)
- Writing to a file
  - **> <file_name>** = push the output of that command to that file_name (and overwrites the file)
    - echo Hi > hi.txt
  - **>> <file_name>** = appends the output of that command to that file_name
    - echo How are you? >> hi.txt
- See contents of a file:
  - **cat <file_name>** = view the contents of that file_name

Command Line Commands

- Moving, renaming & copying
  - **mv <source_file> <destination_file>** = moves that file from one spot to the other
    - The file will no longer be in the source directory
    - Rename a file: use the ‘mv’ command to ‘move’ a file to the same spot with a different name
  - **cp <source_file> <destination_file>** = copies the file to another spot
    - The file will be in both locations

Advanced Command Line Commands

- **grep** = search a file for the matching string
  - grep Hi hi.txt
- | (pipe) = string two commands together in a sequence
  - cat hi.txt | grep Hi
- **python --version**
  - **--** = ‘long’ option sent to the python command
  - Some commands use the - and some use the --
  - Usually the -- is more spelled out (hence long)
  - **python -V** = the short version and does the same thing

Advanced Command Line Commands

- **vi (vim)** - file text editors
  - Best to hit the INS key to start typing (hit INS twice to go to replace mode)
  - To exit hit ESC - :wq <enter>
    - w = write
    - q = quit
    - If you don't want to write the changes type :q! (quit and disregard changes)
- **chmod 755 <file_name>** - modify permission to a file
  - First Number 7 - Read, write, and execute for user
    Second Number 5 - Read and execute for group
    Third Number 5 - Read and execute for other
  - Can do 777 for Read, write, and execute for everyone (but less secure)

Command Line Review

- Navigation
  - **ls** - shows the contents of the present folder (**dir** for windows)
    - **ls -alh** - anything after a hyphen is called an option, flag, or switch & modifies the original command
      - a = all files including hidden ones (ones that start with a .)
      - l = long list
      - h = human readable (so a size of 4096 bytes = 4.0K instead)
    - dir (Windows/DOS)
  - **cd**= change directory
    - chdir (Windows/DOS)
    - **cd ..** = back one directory
  - **pwd** = print working directory - shows full path (location) of current directory
  - **cd** for Windows/DOS

Command Line Review

- Writing to a file
  - **> <file_name>** = push the output of that command to that file_name (and overwrites the file!!)
    - echo Hi > hi.txt
  - **>> <file_name>** = appends the output of that command to that file_name
    - echo How are you? >> hi.txt
- See contents of a file:
  - **cat <file_name>** = view the contents of that file_name

Command Line Review

- See contents of a file:
  - **less <file_name>** = view the full contents of that file_name, with scrolling for large files, press ‘q’ to exit (Linux/Mac tool, some versions of Windows CMD)
  - **more <file_name>** = view the contents of that file_name with scrolling for large files (Windows CMD/Dos tool, also ported to all Linux/Mac)
    - up/down arrows to scroll
    - **f** to page down, **b** to page up
    - **/<string>** to search for next occurence of “string”
    - **q** to exit
    - Example to show running processes: **ps -ef | more**
      - *(“|” is the vertical pipe above \)

Command Line: PC vs Mac/Linux

- echo
  - **Linux/Mac: echo “text goes here (parentheses need to be in quotes!)”**
  - **PC: echo text goes here (and parentheses are ok)**
- **| (pipe): pipe output of one command (eg, cat logfile.txt) to another command or script (eg, grep “error”)**
  - **E.g.: cat logfile.txt | grep “error”**

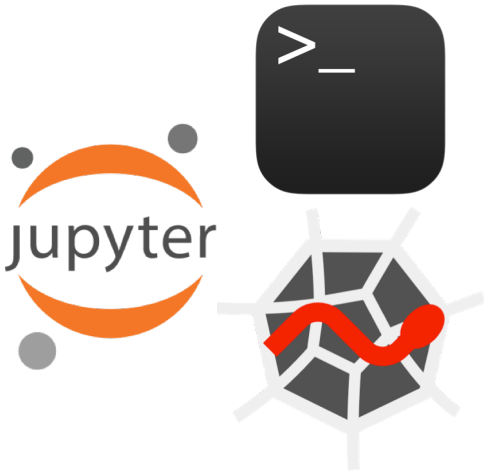
Command Line: Making an Executable

- At top of file:
  - #!/bin/sh
  - (This is called a “shebang”)
  - **Make executable (chmod +x script.sh)**
  - **Can now run as ./script.sh**
- **For more details/answers:**
  - **https://stackoverflow.com/questions/8779951/how-do-i-run-a-shell-script-without-using-sh-or-bash-commands**

Running Python

3 ways to start Python and input code:

- Command-line interface
- Jupyter Notebook
- Spyder



Python from the command line

- Find and open your terminal, this will be powershell on PC and shell/terminal on mac
- Launch python from the shell, by typing ‘python’
- Should look like this:

```
(py3) D:\omis30>python
Python 3.6.6 |Anaconda custom (64-bit)| (default, Jun 28 2018, 11:27:44) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

- Make sure the version is Python 3.6

Python from the command line - simple commands

- Type in 100 + 100 and hit enter - should get a response of 200

```
(py3) D:\omis30>python
Python 3.6.6 |Anaconda custom (64-bit)| (default, Jun 28 2018, 11:27:44) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> 100 + 100
200
>>>
```

- Next type in a simple print command in Python:
- print("Hello World")

```
Python 3.6.6 |Anaconda custom (64-bit)| (default, Jun 28 2018, 11:27:44) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> 100 + 100
200
>>> print("Hello World")
Hello World
>>>
```

Python from the command line

- This interpreter is useful for short commands or testing of some code but probably isn't practical for a large program
- To exit: type exit() or CTRL-D

```
(py3) D:\omis30>python
Python 3.6.6 |Anaconda custom (64-bit)| (default, Jun 28 2018, 11:27:44) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> 100 + 100
200
>>> print("Hello World")
Hello World
>>> exit()

(py3) D:\omis30>
```

Homework 1.1

- Install Anaconda on your machine
  - Instructions here:
  - https://github.com/denisvrdoljak/OMIS30_Fall2018/blob/master/AnacondaInstallation.pdf
- Submit a short introduction to yourself and what you hope to learn from this course (1 paragraph max)
  - email to dvrdoljak@scu.edu
- Survey:
  - https://goo.gl/forms/T6EBefwNIvG4EiAN2

Reading Assignment 1.2

Learning Python: Chapter 1 (cont'd)

Learning Python: Chapter 2