Lab 0: Environment, Python, and Testing!

Lab 0 is intended to make sure you have a working development environment, can download starter code from PolyLearn, test your code, and submit files to PolyLearn.

Part 0

Determine and set up your code development and test environment. If you choose to develop code on your own machine using an IDE, remember that the tests we run will be in the server environment.

Part 1

Download the planets.py and planets_tests.py files from PolyLearn to your development environment.

In the plantets.py file, add code to implement the functionality shown by the following sample run:

Sample Run:

```
What do you weigh on earth? 136

On Mars you would weigh 51.68 pounds.
On Jupiter you would weigh 318.24 pounds.
```

Important Information and Requirements:

- To calculate a person's weight on Mars, multiply their weight on earth by 0.38.
- To calculate a person's weight on Jupiter, multiply their weight on earth by 2.34.
- Initially, execute the planets.py and confirm that your output looks like the sample run after you have added the necessary code to the weight_on_planets function.
- After confirming the sample run, then run the unit tests in planets_tests.py to check your solution. Note there is also a test to check that the exception is raised when the input string does not match a planet in the function.
- Add another planet, Venus (multiply weight by 0.91), to the weight_on_planets function, and include a test in planet_tests.py to check the new code.
- For this lab, no need to check for valid input of weight, can assume it will be a positive number
- Submit your modified planets.py and planets_tests.py files to PolyLearn.

For folks new to Python or who need review

The basic Python Tutorial
https://docs.python.org/3/tutorial/

This site gives the essentials for those familiar with Java.
Python for Java Programmers: http://python4java.necaiseweb.org/Fundamentals/Fundamentals

If you want to use an IDE, you are welcome to. If you do not already have an IDE that you are familiar with, PyCharm is a good choice for Python development.
PyCharm IDE: https://www.jetbrains.com/pycharm/

Videos on specific topics.
Installing Python on Mac/Windows: https://www.youtube.com/watch?v=YYXdXT2l-Gg&list=PL- osiE80TeTt2d9bfVyTiXJA-UTHn6WwU

Strings: https://www.youtube.com/watch?v=k9TUPpGqYTo&list=PL-osiE80TeTt2d9bfVyTiXJA-UTHn6WwU&index=2

Conditionals: https://www.youtube.com/watch?v=DZwmZ8Usvnk&list=PL-osiE80TeTt2d9bfVyTiXJA- UTHn6WwU&index=6

Loops: https://www.youtube.com/watch?v=6iF8Xb7Z3wQ&list=PL-osiE80TeTt2d9bfVyTiXJA- UTHn6WwU&index=7

Functions: https://www.youtube.com/watch?v=9Os0o3wzS_I&list=PL-osiE80TeTt2d9bfVyTiXJA-UTHn6WwU&index=8

Modules: https://www.youtube.com/watch?v=CqvZ3vGoGs0&list=PL-osiE80TeTt2d9bfVyTiXJA- UTHn6WwU&index=9

For folks new to Cal Poly: Unix Environment

Unix

The lab machines run a distribution of the Linux operating system. For simplicity, and to gain experience in a, potentially, new environment, we will do our coursework in this environment.

Open a terminal window. To do so, from the system menu on the desktop toolbar, select **Applications → System Tools → Terminal**. The Terminal program will present a window with a command-line prompt. At this prompt you can type Linux commands to list files, move files, create directories, etc. For this lab you will use only a few commands. Additional commands can be found at:

- Unix Tutorial
    - Tutorials 1 & 2
    - Parts 1-5
- Editors
    - emacs tutorial
    - vi tutorial

In the terminal, type **ls** at the prompt and hit <Enter>. This command will list the files in the current directory. (also know as a folder.) If you type **pwd**, the current directory will be printed (it is often helpful to type **pwd** while you are navigating directories). If you type **tree**, then you will see a tree-like listing of the directory structure rooted at the current directory.

Create a new directory for your coursework by typing **mkdir cpe202**. Use **ls** again to see that the new directory has been created.

Change into this new directory with **cd** by typing **cd cpe202**. To move back "up" one directory, type **cd ..**. To summarize

- **ls** list files in the current directory
- **cd** change to another directory
- **mkdir** create a new directory
- **pwd** print (the path of) the current directory

Though these basic commands are enough for now, consider working through a Unix tutorial.

Executing a Program

Download the **.py** files for lab0 from PolyLearn into the **cpe202** directory created above; this can be done via the browser (by selecting the location to save to), via the graphical file manager, or through the use of the **mv** command in the terminal window.

A Python program is written in a plain text file. The program can be run by using a Python interpreter (we are using Python 3 in this class). You can see the contents of *planets.py* by typing **more planets.py** at the command-line prompt.

To execute the program, type **python3 lab1.py** at the command-line prompt.

To summarize

- **mv** move files
- **more** display contents of a file
- **python3** Python 3 interpreter, executes a program by specifying name of program file

Editing

There are many options for editing a Python program. On the department machines, you will find vi, emacs/xemacs, nano, gedit, sublime, and others. The editor that one uses is often a matter of taste. You are not required to use a specific editor, but we will offer some advice (and we will try to help with whichever one you choose). There is lots more information here: http://users.csc.calpoly.edu/~akeen/courses/csc101/handouts/labs/lab1.html

Interactive Interpreter

The Python interpreter can be used in an interactive mode. In this mode, you will be able to type a statement and immediately see the result of its execution. Interactive mode is very useful for experimenting with the language and for testing small pieces of code, but your general development process with be editing and executing a file as discussed previously.

Start the interpreter in interactive mode by typing **python** at the command prompt. You should now see something like the following.

```
Python 3.6.0 (v3.6.0:41df79263a11, Dec 23 2016, 07:18:10) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

The >>> is the interpreter's prompt. You can type an expression at the prompt to see what it evaluates to. Type each of the following (hit enter after each one) to see the result. When you are finished, you can exit the interpreter by typing ctrl-D (i.e., hold the control key and hit d).

- 0 + 1
- 2 * 2
- 19 // 3
- 19 / 3
- 19 / 3.0
- 19.0 // 3.0
- 4 * 2 + 27 // 3 + 4
- 4 * (2 + 27) // 3 + 4