CSCI 1133, Fall 2019
Programming Assignment 1
Due: 11:55pm, Wednesday September 11, 2019

Due Date: Submit your solutions to GitHub by 11:55 p.m., Wednesday, September 11th. We will do a pull from this time point. Do not upload anything to Canvas and PLEASE be sure to use proper naming conventions for the file, classes, and functions. We will NOT change anything to run it using our scripts.

Unlike the computer lab exercises, this is not a collaborative assignment. See the syllabus and read section "Academic Dishonesty" for information concerning cheating. Always feel free to ask the instructor or the TAs if you are unsure of something. They will be more than glad to answer any questions that you have. We want you to be successful and learn so give us the chance to help you.

Instructions: This assignment consists of 1 problem, worth 20 points. You must edit the provided Python file, `hw1.py`, and submit it to your repository through git. If you have questions, ask!

Because your homework file is submitted and tested electronically, the following are very important:

- Submit the correct file, hw1.py, through Github by the due date deadline.
- Do not change any code in hw1.py: you are only editing the comments for this assignment. If running the file gives you errors, you're probably not using comments correctly.

Push your work into Github under your own repo. The specific hosting directory should be: repo-<username>/hw1, where you replace <username> with your U of M user name. For instance, if your email address is bondx007@umn.edu, you should push your hw1 to this directory: repo-bondx007/hw1. More details on this are given below.

The following will result in a score reduction equal to a percentage of the total possible points:

- Incorrectly named/submitted source file (20%)
- Failure to execute due to syntax errors (30%)

Problem A. (20 *points*) **Documentation and Uploading to GitHub**

Setup

In this problem, we'll be going over the process of documenting and submitting homework. But first, you'll need to get Python 3 running on your computer. See "Using Python From Your Laptop: A Handy Guide" on Canvas for instructions: I recommend using IDLE for this unless you really liked one of the other options showcased in Lab 1.

You'll also need to install Git, unless you're using a lab computer (either directly, or through VOLE/SSH) to do the assignment. You should be able to go to https://git-scm.com/downloads and simply choose all of the default options for installation (if this doesn't work, email the TA alias or come into office hours). Then open a terminal (on Windows you can right-click within the folder where you want to clone your repository and select "Git Bash here"; make sure you have nothing selected when you do this). Once you have a terminal open in the desired location, follow the same steps you took in Lab 1 to clone your repository:

First, run the following command, replacing the text user0000 with your own Internet ID (note: if you already have a personal git account not associated with the U of M in use on your computer, don't use --global as this will interfere with your previous setup):

```
git config --global user.email "user0000@umn.edu"
```

Next, run this command, substituting your real name for the text, Your Name:

```
git config --global user.name "Your Name"
```

Finally, create a clone of the repository by running:

```
git clone https://github.umn.edu/umn-csci-1133-F19/repo-user0000.git
```

again, replacing user0000 with your own Internet ID.

This will either require you to enter your password in the console, or may create a textbox for you to type it into. Either way, when this is complete, the folder repo-user0000 (with user0000 replaced by your own Internet ID) should exist on your machine in whatever directory you ran the command from: this is your cloned repository, so be sure to do all of your homework within it so that you can push it to GitHub.

Create a folder called hw1 within your cloned repository. Then, download the `hw1.py` template from Canvas, and move it into your hw1 folder. Open the file, using IDLE or your favorite text editor. You should see two functions defined: `cents` and `draw_M`. You don't need to understand the code for these functions for this assignment, but you will have to figure out what the functions do by running them. Below the functions are the two lines:

```
vers = platform.python_version()
assert vers[0] == '3', "You must use Python 3, "+vers+" is not acceptable"
```

These lines do nothing so long as you run them in Python version 3.0 or higher, but if you use some version of Python 2, they'll cause an error to be thrown similar to this one:

```
AssertionError: You must use Python 3, 2.7.6 is not acceptable
```

Run the file now, either in IDLE (use the Run menu or shortcut key F5) or from the command line. Nothing should happen unless you're using the wrong version of Python, in which case you'll get an AssertionError similar to the one above. If you do get the AssertionError above, you should determine why you're running Python 2 instead of Python 3, and fix it. Come to office hours or email the TA alias if you need help.

Documentation

Above each function is a series of comments: lines that start with the comment symbol #. Anything past a # in a line of a Python file does not execute: these are typically used for writing notes in your code, in English. For homework assignments in this class, we require you to use comments to document what each function you write does, what its input parameters are, and what it returns, using the following template:

```
#========================================
# Purpose:
# Input Parameter(s):
# Return Value:
#========================================
```

If you're not familiar with functions, input parameters, or return values, you should read the sections titled "User-defined function basics", "Function parameters", and "Returning values from functions" in your textbook: these are all part of the Week 2 required reading.

For example, the following function approximates the pressure of a gas, using the ideal gas law:

```
def pressure(volume, molecules, temp):
    boltzmann = 1.380649*(10**-23)
    return molecules*boltzmann*temp/volume
```

So we could provide the following documentation in comments above the function:

```
#========================================
# Purpose:
#   Computes the pressure of a gas using the ideal gas law
# Input Parameter(s):
#   volume - The volume of the gas, in meters cubed
#   molecules - The number of molecules of gas present
#   temp - The temperature of the gas, in Kelvin
# Return Value:
#   The pressure of the gas, in Pascals
#========================================
```

You'll notice that the Purpose and the Return Value are very similar for this function. This happens frequently with functions like the above that don't produce any side effects: their only purpose is to compute the return value and return it.

On the other hand, the following function has side effects (printing things to the console, in this case), but has no return value, so its return value is very different from its purpose.

```
#=========================================
# Purpose:
#   Prints out some silly song lyrics
# Input Parameter(s):
#   adj1 - A string representing an adjective
#   noun1 - A string representing a noun
#   adj2 - A string representing an adjective
#   noun2 - A string representing a noun that rhymes with noun1
# Return Value:
#   None
#=========================================

def mad_lib(adj1,noun1,adj2,noun2):
    print("We must be",adj1,"as a coursing river")
    print("With all the force of a great",noun1)
    print("With all the strength of a",adj2,"fire")
    print("Mysterious as the dark side of the",noun2)
```

It is possible for a function to have both side effects and a return value. It is also possible for a function to have neither, but such functions are pointless, since they don't affect anything or return anything.

For most homework assignments, you will write the functions yourself, so documenting your code should be easy: you have to know what the purpose, input parameters, and return value of the function is before you write it. However, in this assignment, your task is to write out the documentation for two functions that have been written for you: `cents` and `draw_M`. This means that you should first test out the functions to determine what they do. In IDLE this is pretty straightforward: after running the file, you can enter commands into the interpreter that use the functions from the file. You can get an equivalent state when running from the command line by using the -i option, which runs the file in interactive mode:

```
python3 -i hw1.py
```

Once you've done this, try running `cents` a few times to see what it does. `cents` takes in 4 arguments, all of which should be integers, so try things like:

```
>>> cents(1, 1, 0, 3)
>>> cents(2, 0, 3, 5)
```

Hopefully between testing the function and the names of the variables, it should be pretty apparent what the purpose of this function is, so fill out the Purpose, Input Parameters, and Return Value for `cents` in the comments above the function within the `hw1.py` file. If you don't get it, feel free to email us for hints.

Then do the same for `draw_M`. Unlike `cents`, `draw_M` has no parameters, so you should only need to run it once to see what it does. Note that this calls on the turtle graphics module, so you need to be using something that has a graphical display (no SSH without X-forwarding).

```
>>> draw_M()
```

Uploading to GitHub

Once you have filled out the documentation for both functions, run the file one last time to ensure that you didn't accidentally introduce any syntax errors. **You should always run the file one last time before submitting through git, on every homework assignment**. Losing 30% of your grade on an assignment because you accidentally forget a single # somewhere and caused a syntax error feels awful, so ensure it doesn't happen to you.

Open a terminal and navigate to the hw1 folder in command line. Then you need to do the following sequence of commands:

```
git pull
git add hw1.py
git commit -m "Completed hw1"
git push
```

Then, go to https://github.umn.edu/umn-csci-1133-F19/repo-user0000 (where user0000 is replaced by your Internet ID), and check that your repository contains a hw1 folder, with your documented hw1.py inside of it (click on hw1.py and read it, to ensure that it is the correct version).

If so, then Congratulations! You have completed your first CSCI 1133 Programming Assignment.