Assignment #6
Loops and Functions Using Turtle in Python
Due: Sunday, 11/08/15, 11:59pm

You need to use the TAs office hours and class exercise groups to get extra help in understanding the material and what is required from an assignment or lab!!!

In this assignment, you will design a computer program using sequential, conditional, looping, and functions. **You are not allowed to use the built-in write() function in this assignment.** You are going to learn to read Python documentation to create a graphical program using the turtle library. [https://docs.python.org/3.2/library/turtle.html](https://docs.python.org/3.2/library/turtle.html)

Another resource to learn how to use the turtle class is: [http://openbookproject.net/thinkcs/python/english3e/hello_little_turtles.html](http://openbookproject.net/thinkcs/python/english3e/hello_little_turtles.html)

**NOTE:** If you are a Mac user and did not have an Xserver already installed to run turtle, then you need to install xquartz [http://www.xquartz.org/](http://www.xquartz.org/). After the install, reboot your machine (turn it off and back on!!!), and then launch an xterm from within xquartz (not your normal terminal!!!!). Now, you can use ssh –Y.

(40 pts) First, write a program to draw a start like the one below, when the user clicks the turtle, and make sure you clear the screen and re-draw the star each time the user clicks the turtle:

![Star](image)

Hints:
- Try this on a piece of paper, moving and turning your cellphone as if it was a turtle. Watch how many complete rotations your cellphone makes before you complete the star. Since each full rotation is 360 degrees, you can figure out the total number of degrees that your phone was rotated through. If you divide that by 5, because there are five points to the star, you'll know how many degrees to turn the turtle at each point.

- You can hide a turtle behind its invisibility cloak if you don’t want it shown. It will still draw its lines if its pen is down. The method is invoked as `my_turtle.hideturtle()`. To make the turtle visible again, use `my_turtle.showturtle()`.

Begin by designing you program using these steps, and write steps 1, 2, and 4 on paper or in a text editor. Then, implement the program using Python.
**Step 1: Problem Analysis.** (10 pts)

a. Comments about the problem to aid in understanding it.

b. Description of the knowledge base (this list would include what you would be expected to know to follow the solution).

**Step 2: Program Design.** (10 pts) List the specific steps needed to get your turtle to make a star on the screen, upon a user mouse click. Remember, you have to be very explicit here to make sure the computer can accomplish the task using your directions.

1.

2.

**Step 3: Program Implementation.** (10 pts) This is the Python code that moves your turtle around the screen to draw a star, i.e. your .py file that gets saved before running your program.

**Step 4: Program Testing.** (10 pts)

Create a Test Plan with several test cases including the average and extreme cases.

(60 pts) Now, write a program to draw your name on the screen, when the user clicks the turtle, and make sure you clear the screen and re-draw your name each time the user clicks the turtle. I would highly suggest you use graphing paper or some kind of paper to design what will be needed to get the turtle to draw your name. Have fun, it's your own creation!

Begin by designing your program using these steps, and write steps 1, 2, and 4 on paper or in a text editor. Then, implement the program using Python.

**Step 1: Problem Analysis.** (10 pts)

a. Comments about the problem to aid in understanding it.

b. Description of the knowledge base (this list would include what you would be expected to know to follow the solution).

**Step 2: Program Design.** (20 pts) List the specific steps needed to get your turtle to draw your name on the screen, upon a user mouse click. Remember, you have to be very explicit here to make sure the computer can accomplish the task using your directions.

1.

2.

**Step 3: Program Implementation.** (20 pts) This is the Python code that moves your turtle around the screen to draw your name, i.e. your .py file that gets saved before running your program.

**Step 4: Program Testing.** (10 pts)

Create a Test Plan with several test cases including the average and extreme cases.

Electronically submit your two Python programs as a .py and design document as a pdf by the assignment due date, using TEACH:

https://secure.engr.oregonstate.edu:8000/teach.php?type=want_auth