ISTA 130 – Fall 2017
Programming Assignment 1

Due **Thursday**, **September 7**, at **11:59 pm**

Instructions and suggestions:

- **File names and function signatures must be exactly as specified.**
- Use `template.py` and `docstring.py` as guides to correct format and documentation.
- You can type the following within a python shell to read the documentation of the turtle module. Use enter to scroll and press 'q' when you're done reading documentation. The same information can be found at: https://docs.python.org/3.5/library/turtle.html.

```
>>> import turtle
>>> help(turtle.Turtle)
```

- End your functions with a `return None` statement. This is not a Python requirement, but lays the groundwork for future work we will be doing.
- For convenience, here is a short list of some useful functions (Note: you do not need to type the inline comments. Those are there as little notes to you):
    - `turtle.penup() # don't forget turtle.pendown() afterwards!`
    - `turtle.pendown()`
    - `turtle.goto(x, y)`
    - `turtle.setheading(direction) # direction is a number between 0 and 360.  East is 0, north is 90, etc.`
    - `turtle.left(angle) # changes turtle's direction by x degrees to the left`
    - `turtle.right(angle)`
    - `turtle.forward(distance)`
    - `turtle.backward(distance)`
    - `turtle.pencolor("colorstring")`
    - `turtle.pensize(size)`
    - `turtle.circle(radius, arc_size_in_degrees)`
- Just try things!
- If you have a bug that you just can't find (you likely will), ask for help. Don't get so frustrated that your head feels like it will explode.

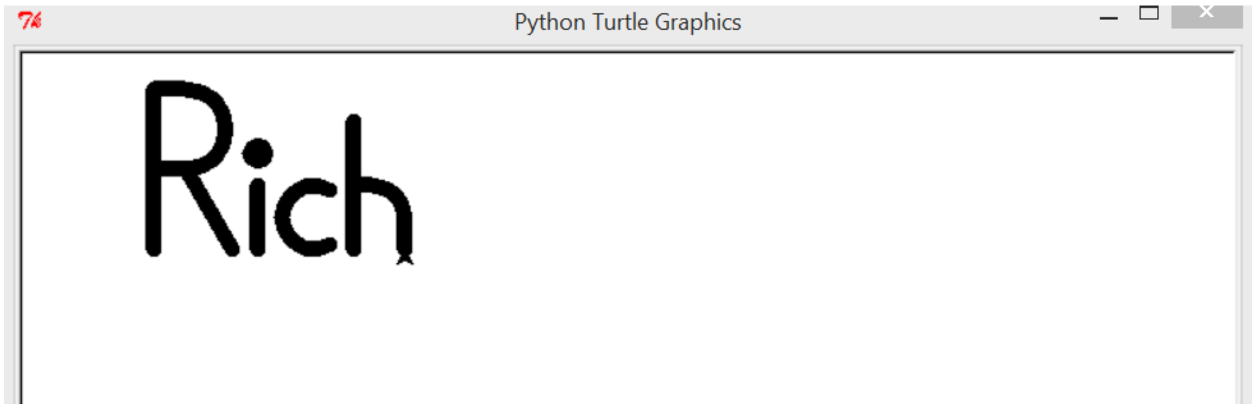
- When I was doing these problems, it really helped me to use a pen and paper to map a plan, keep track of positions on the turtle screen, etc. Graph paper would have been even better.

Problem 1: Name (34 points)

For this problem, you will create a program that will write your first name with a turtle. You can use a nickname, but it must be at least three letters long, not including any letters that we did the code for in class ('C', 'O', 'L', 'A'). It can include these letters and you can use the code from class for them, but it must include at least three letters you create yourself. If your name is long, you can just write the first three letters of it. Creating these functions is an example of decomposition and makes more code reuse possible.

1) Download "template.py" from D2L and open it in Sublime, Notepad++, or some other plain text editor, and save it as "name.py". Update the comment at the top, filling in the data as appropriate.
2) Create (at least) three functions that each draw one letter. Each will have a signature that looks like:
   - `def draw_k(draw_turtle):`

   Feel free to use the code from class, posted on D2L as a guide, and if you need letters that we did in class, you can use the class code, but you must create at least three of your own. At the end of each of your functions, the turtle should be pointing east.
3) Your `main()` function will call each of these functions to spell out your name. Repositioning the turtle between drawing letters should happen in `main`. Set the turtle's speed to 0. All letters must be drawn by functions. Example output:



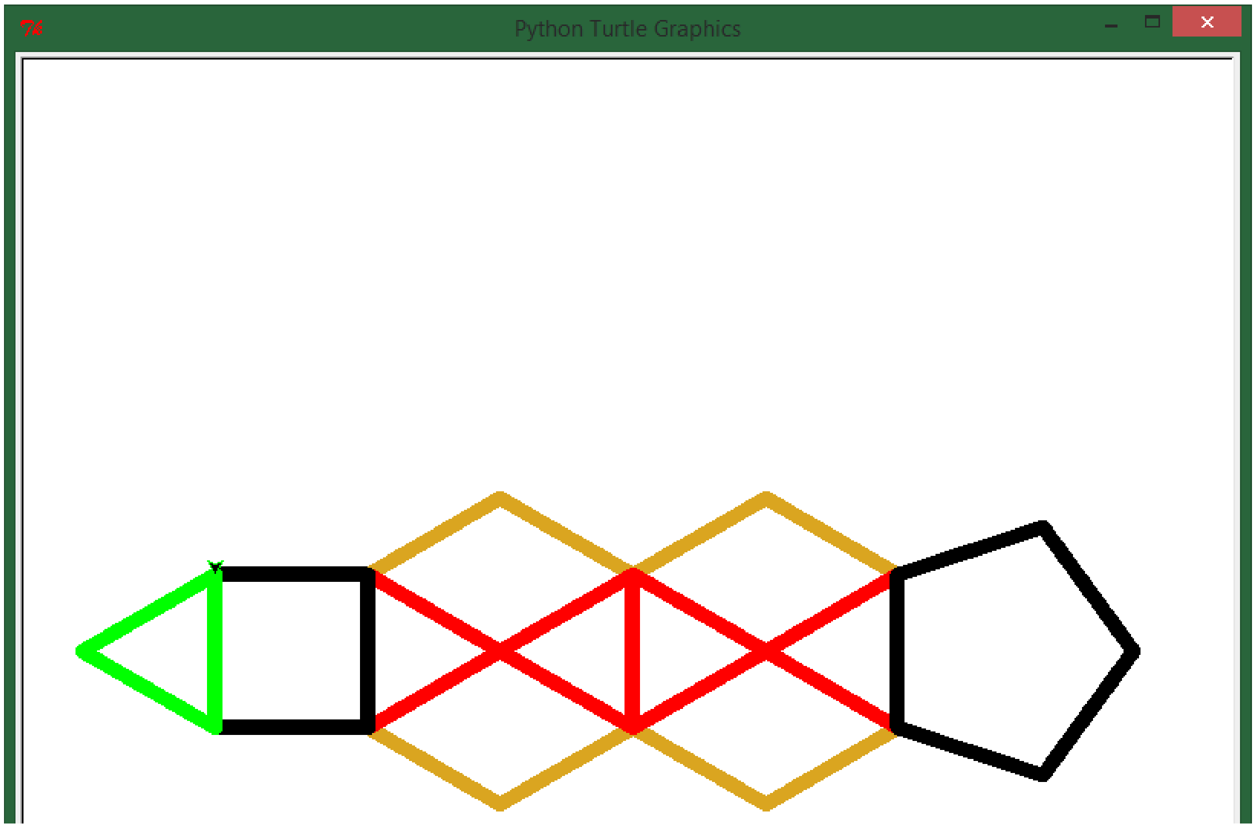
4) Your code does not have to use `turtle.circle()`. In other words, you can use blocky letters, but your SL must be able to tell what you mean. If your section leader wonders is that an "R" or an "A", you will lose points.

5) Make sure that you have added documentation to each of your functions (in addition to the top). Test adequately. Upload to your dropbox.

Problem 2: Snake (33 points)

For this problem, you will create a program that will draw a snake, using functions that draw a triangle, a square, a pentagon, and a hexagon.

1) Download “template.py” from D2L and open it in Sublime, Notepad++, or some other editor, and save it as “snake.py”. Update the comment at the top, filling in the data as appropriate.
2) Write a function called `triangle` that draws an equilateral triangle. The function signature has two parameters. The first is for a turtle and the second is for a numeric value giving the length of the sides of the triangle.
    - For example:
        i. Calling your function like this: `triangle(leonardo, 100)` will make the turtle named `leonardo` draw a large triangle.
        ii. Calling your function like this: `triangle(raphael, 10)` will make the turtle named `raphael` draw a small triangle.
3) Write a function called `square` that draws a square. The function signature has parameters. The first is for a turtle and the second is for a numeric value giving the length of the sides of the square.
4) Write a function called `pentagon` that draws a pentagon. The function signature has two parameters. The first is for a turtle and the second is for a numeric value giving the length of the sides of the pentagon.
5) Write a function called `hexagon` that draws a hexagon. The function signature has two parameters. The first is for a turtle and the second is for a numeric value giving the length of the sides of the hexagon.
6) In the `main` function, write code that will:
    a. Create a new turtle.
    b. Set the turtle’s speed to 0 (fastest) and the pensize to 10.
    c. Use the `triangle`, `square`, `pentagon`, and `hexagon` functions (and only these functions – no `triangle2`, etc.) you wrote to draw the picture shown in the following figure:



7) You can use different colors if you would like, but there must be at least three. I used “goldenrod”, “green”, “red”, and “black”. http://www.w3schools.com/html/html_colornames.asp.
8) Edge lengths are 100, however you small scale them down if that’s too large on your screen. However, your section leader must be able to clearly distinguish each part.
9) Your pattern of overlapping doesn't have to match the drawing. In other words, the order in which you draw your polygons doesn't matter as long as you end up with the snake in the end.
10) Make sure that you have added documentation to each of your functions (in addition to the top). Test adequately. Upload to your dropbox.

Problem 3: Player’s choice! (33 points)

For this problem, you will get creative. Some of the most radical drawings will get shown in class.

1) Download “template.py” from D2L and open it in Sublime, Notepad++, or some other editor, and save it as “drawing.py”. Update the comment at the top, filling in the data as appropriate.

2) Write a function called `shape` that draws the shape of your choice. The function signature has two parameters. The first is for a turtle and the second is for a numeric value giving the length of each side of the shape. The body of the function must contain at least 5 instructions.

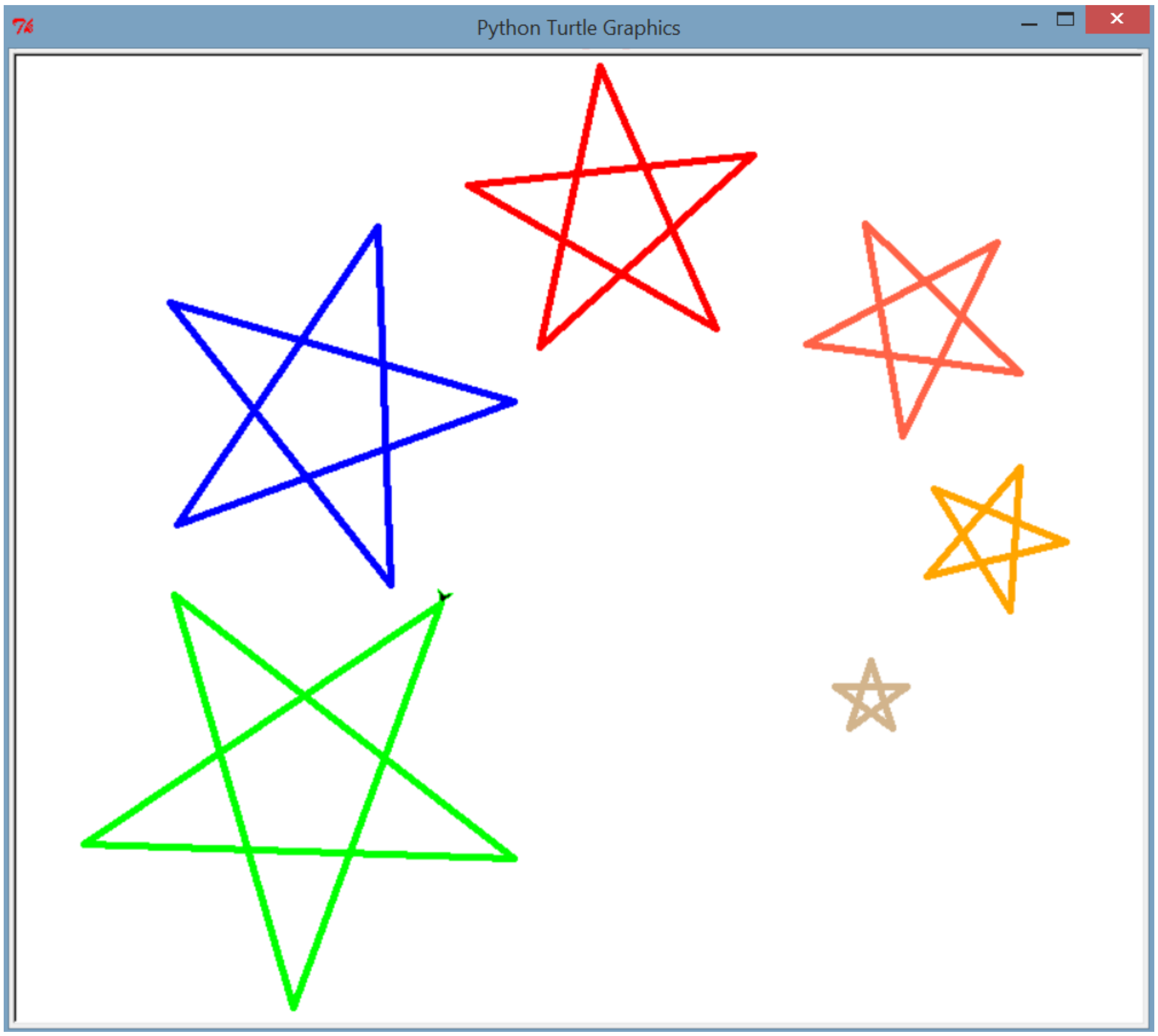
3) Don't use any shapes we have already done, or the star shown below.

4) In the `main` function, write code that will:

    a. Create a new turtle.

    b. Set the turtle's speed to 0.

    c. Draw your shape in at least three different locations in at least three different sizes using at least three different colors.



5) I will show my favorite `drawing.py` modules in class. If you make one that is very elaborate, you can break the rules above, but it has to be extravagant.

6) Make sure that you have added documentation to each of your functions (in addition to the top). Test adequately. Upload to your D2L Assignments folder.