ISTA 130 – Fall 2018
Programming Assignment 1

Due Thursday, February 8, at 11:59 pm

Instructions and suggestions

- See the syllabus for policies on homework, including collaboration and grading.
- If you start late and don't get everything finished on time, still turn in what you have done. If you turn in some of the required scripts on time and some within the 24 hour late period, you will only receive the 10% deduction on the late scripts.
- If you receive less than a 70, you may contact me about fixing up your code and resubmitting it. There will be an additional penalty.
- **File names and function signatures must be exactly as specified.**
- **Do not post more than one line of code to a D2L discussion about homework! That code must be broken, and you must be asking for help. Do not post code solutions to other students' problems, describe the solution to the issue in English. Failure to follow these rules will result in a 0 for the assignment! If you are not certain about something you are considering posting, email it to the listserv instead.**
- Use the `template.py` and `docstring.py` files as guides to correct format and documentation. Examples of good documentation are also in all of the posted code examples. Documentation is comments that you put in your file so that you and others know what it does and how it works. There are two kinds of comments in Python – multiline comments and inline comments. Multiline comments start and end with three quotes. A special kind of multiline comment is called a docstring, and it is the first text in the body of a function (immediately beneath the function signature). Docstrings are what we will be looking for and basing your documentation grades on.
- Documentation for the turtle module is at: https://docs.python.org/3.6/library/turtle.html.
- For convenience, here is a short list of some useful turtle methods. Use the dot operator to call these on turtle objects that you have constructed.
  - ✓ `penup()`
  - ✓ `pendown()`
  - ✓ `goto(x, y)`
  - ✓ `setheading(direction) # East is 0, north is 90, etc.`
  - ✓ `left(angle)`

- ✓ `right(angle)`
- ✓ `forward(distance)`
- ✓ `backward(distance)`
- ✓ `pencolor(colorstring)`
- ✓ `pensize(size)`
- ✓ `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.

Introduction

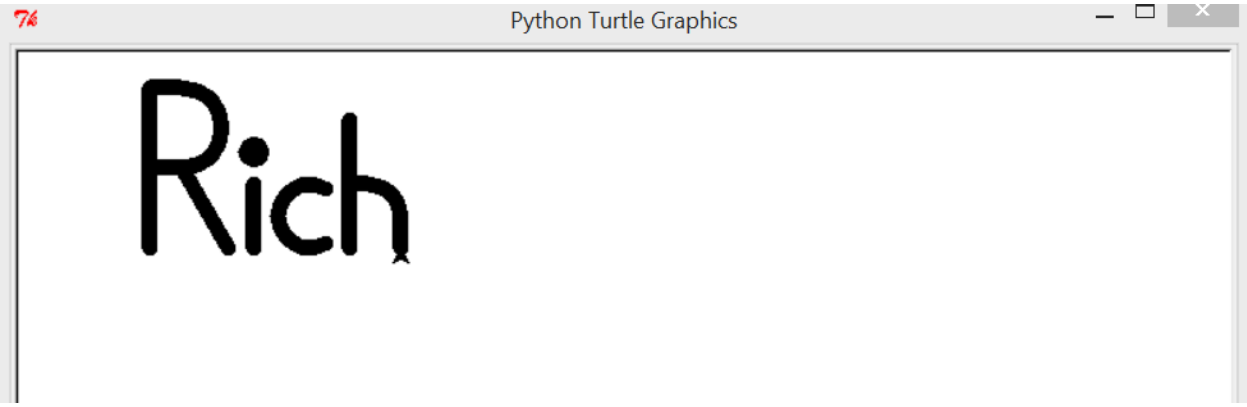
This homework's learning objective is to master the creation and use of functions. Functions allow us to bundle multiple lines of code together, give that bundle of code a name, and to later execute that bundle in one line by calling it (using its name in our code). Functions are incredibly important because we could not write good code without them. Learning how to create and use them is also the hardest thing we will do this semester. A more in-depth look at functions is in `functions_all_about.pdf`.

Problem 1: Name (34 points)

The `coca_cola_fns.py` module, posted on the class website, is your example code for this problem. Your solution should look very much like it. 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. You can use the code from class, but your script 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 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(some_turtle):`
3) At the end of each of your functions, the turtle should be pointing east.

4) 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:



5) 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.

6) Make sure that you have added documentation to each of your functions (in addition to the top). Test adequately. Upload your file to the Hw1 Assignments folder on D2L.

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 plain text editor, and save it as `snake.py`. Update the comment at the top, filling in the data as appropriate.

2) Each of the following functions should leave the turtle in the same place pointing in the same direction as it was before the function call.

3) 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:

   - Calling your function like this: `triangle(leonardo, 100)` will make the turtle named `leonardo` draw a triangle with sides of length 100.
   - Calling your function like this: `triangle(raphael, 10)` will make the turtle named `raphael` draw a triangle with sides of length 10.

4) Write a function called `square` that draws a square. 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 square.

5) 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.

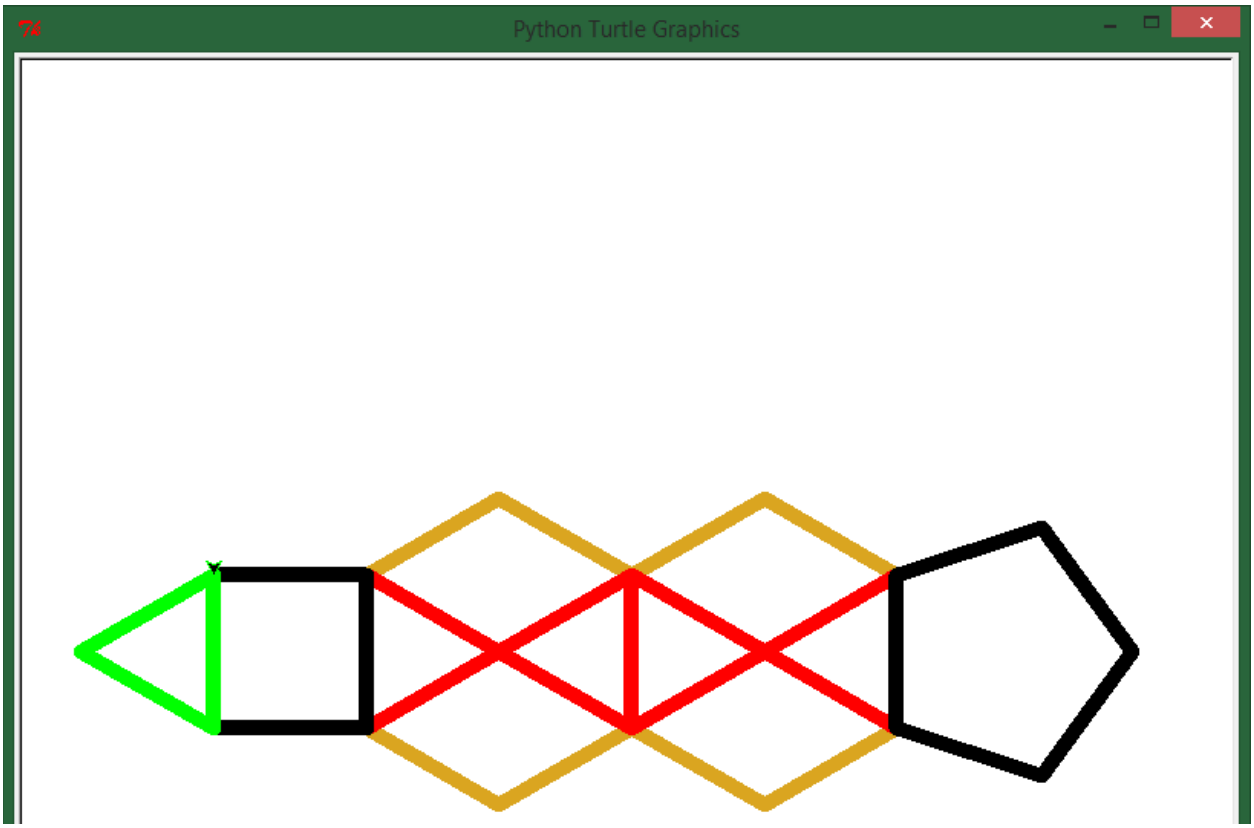
6) Write a function called `hexagon` that draws a hexagon. The function signature has two parameters. The first is for a turtle and the second is for a numeric value giving the length of the sides of the hexagon.

7) 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:



8) You can use different colors if you would like, but there must be at least three. I used "goldenrod", "green", "red", and "black". https://www.w3schools.com/colors/colors_names.asp.

9) 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.

10) 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.

11) Make sure that you have added documentation to each of your functions (in addition to the top). Test adequately. Upload your file to the Hw1 Assignments folder on D2L.

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

For this problem, you will get creative. Some of the most radical drawings will get posted to the class website.

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

2) If you draw an outrageous picture, you don't have to follow the rules below.

3) 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.

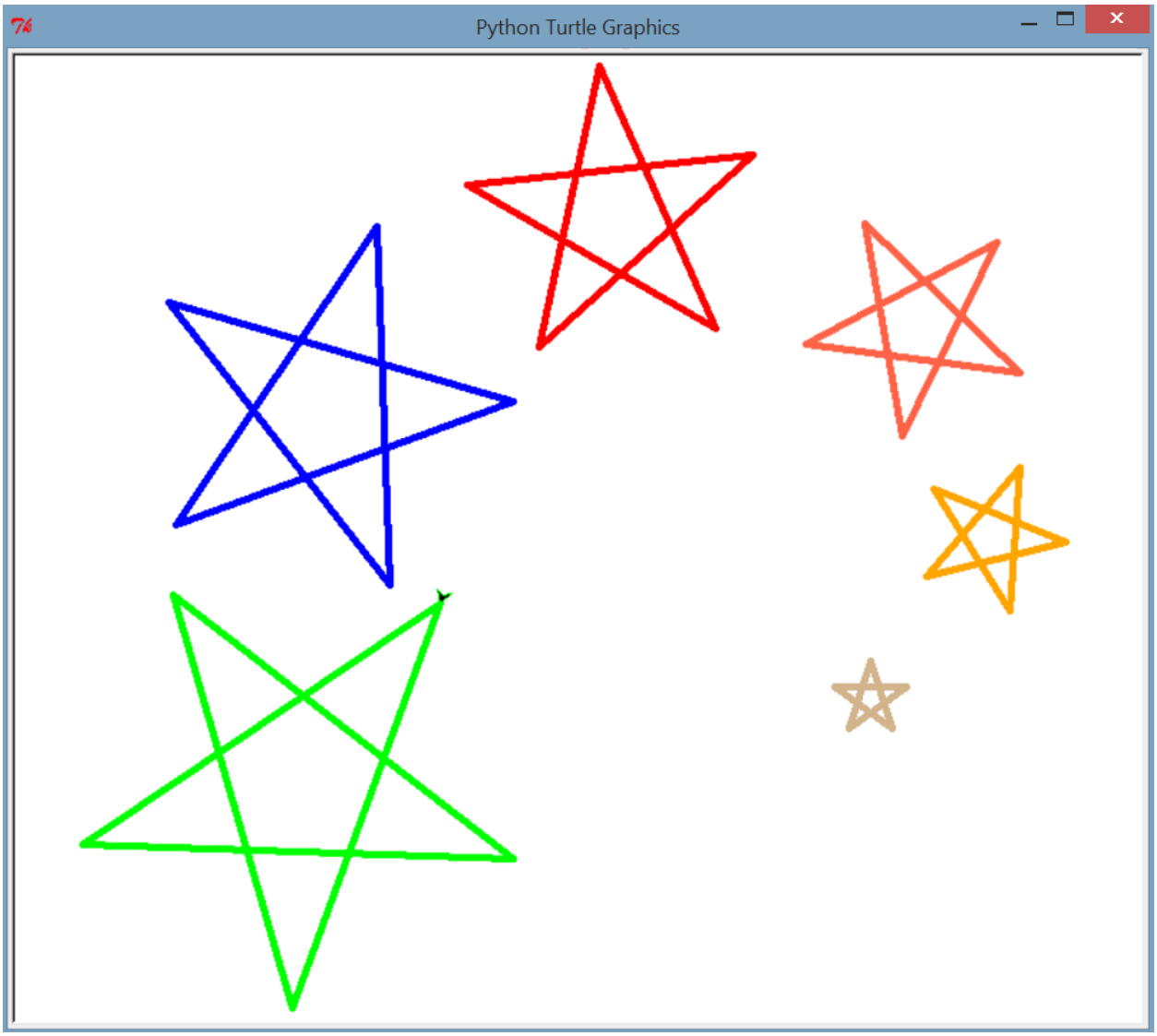
4) Don't use any shapes we have already done, or the star shown below.

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

    a. Create a new turtle.

    b. Set the turtle's speed to 0.

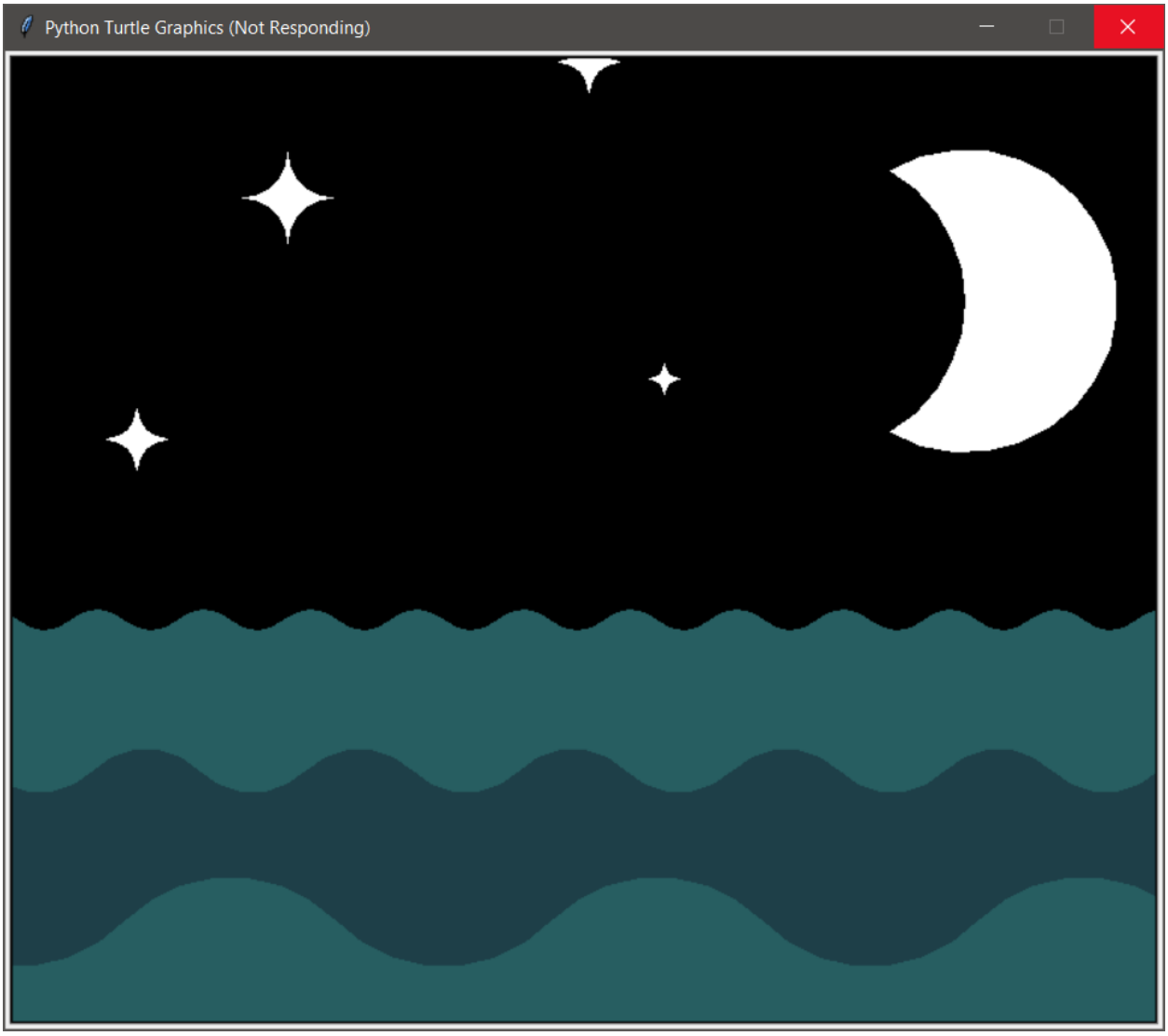
    c. Draw your shape in at least three different locations in at least three different sizes using at least three different colors.



6) Make sure that you have added documentation to each of your functions (in addition to the top). Test adequately. Upload your file to the Hw1 Assignments folder on D2L.

7) While your shapes can be relatively simple like the above example, I encourage you to use your imagination! I will post a video art gallery of some of my favorites. Students in the past have even made them function like movies. Here are a few past favorites of mine (I am in an ocean kind of mood):


Python Turtle Graphics (Not Responding)


Python Turtle Graphics (Not Responding)

And, of course, my all-time favorite: