PSA 1: Turtle Shapes and Turtle Fun

Read all instructions in this document before starting any coding!

Due: Sunday, October 15th, 11:59pm

Now that you have become more familiar with DrJava, you're ready to create more programs with Turtles and get practice using methods for code reusability.

As described in class, **you may complete at most 4 of the quarter's programming assignment using pair programming.** If you have decided to work in pairs, please review the guidelines for pair programming. In addition, note the following details about working with a partner:

- For this assignment, if you decided to work in pairs, find a partner before you start on the assignment. Or else, you have violated the policy on pair programming.
- You will submit only ONE version between the two of you though your group can submit as many times as you want to.

Starter code: LINK HERE

Helpful Information:

Online Communication: Using Piazza, Opening Regrade Requests

Getting Help from Tutor, TA, and Professor's Hours

Lab and Office Hours (Always refer to this calendar before posting.)

Academic Integrity: What You Can and Can't Do in CSE 8A

Remote Lab Access and File Transfer Guide or **Yingjun's SSH and VNC post**

Turtle Documentation

Table of Contents:

Getting Started

Instructions for working on a B230 lab machine:

1) Open a terminal
2) Copy the starter code from the public folder.

Paste the following lines of code one after the other in the terminal i.e. paste the first line, press enter and then the second line and so on:

```
cd ~/
mkdir psa1
cd psa1
cp ../../public/psa1/Turtle.java .
ls
```

(Do you recall what each of the above commands mean. If you are not able to, review PSA0, where each of the commands is explained.)
3) Verify that 'Turtle.java' is shown as the result of the last command.
Following these procedures, your file will be named correctly and be located in the correct place. For every assignment you need to follow correct naming assignments in the correct locations to get credit for your work.
4) Proceed to start your programming assignment.

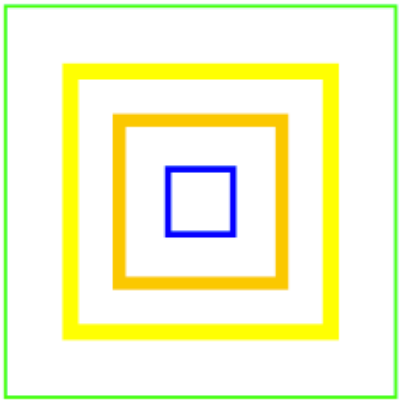
Instructions for working on your Local Machine (Your Laptop/Computer):

1) Create a psa1 folder on your machine.
2) Save Turtle.java inside the psa1 folder: LINK HERE

Problem

Part A: Turtle Shapes [6 points]

This program requires you to create a program called **CreateShapes.java** and modify the existing Turtle.java file in order to draw **four** copies of the same shape (e.g., octagon, star, or some other shape, see below for restrictions) but with varying sizes and line thicknesses so as to look something like this:



As in the example, you are required to have all four of the shapes be the same shape and be nested within one another. However, you **cannot** do *triangles, rectangles, diamonds* or *squares* for this part as that would be too easy. Your shape must be two dimensional (a line does not count as a shape). The point of the assignment is to challenge yourself. So be creative to make simple shapes but also to push the limits of what you can do. The documentation on how to use the Turtle class can be found here.

Requirements for this assignment:

1) Add an additional drawShape method to the Turtle class.

You must add this method AT THE BOTTOM of Turtle.java, but BEFORE the brace } that ends the class definition. Some code for this method is provided in the sample code section near the end of this document. The rest of the method needs to be completed. Refer to the comments in the sample code for specifications and hints.

2) Use the size parameter appropriately in the drawShape method **(1 point)**.

3) Use the thickness parameter appropriately in the drawShape method **(1 point).**

4) Create CreateShapes.java. Some sample code is provided near the end of this document.

Your program will do the following in the order listed:

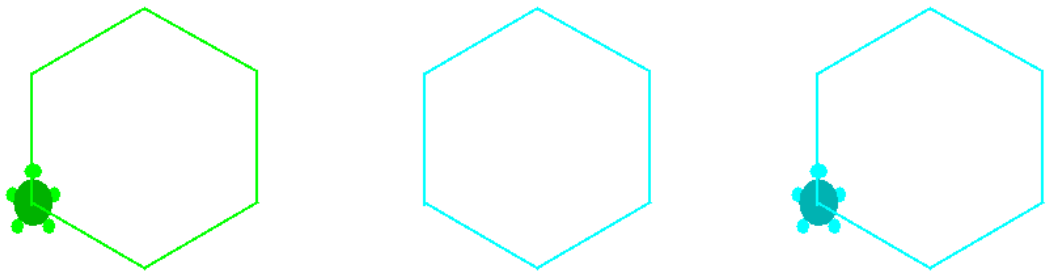
1) Create a World for turtles.

2) Create four turtles in that world (where you will use each 1 to draw one copy of the shape) **(1 point)**.

3) For each turtle, call the drawShape method with different parameters to nest the shapes so that they have the same center point. **(3 points)**.

Part B: Turtle Fun [4 points]

This program requires you to create a program called **TurtleFun.java** and to create three visually separate shapes using only **TWO** Turtle objects but **MULTIPLE** references so as to look something like this:



As in the example, you are required to have all three shapes that are distinguishable from one another. If it LOOKS like there is one shape (i.e. they are drawn one on top of the other) on the screen then we consider that to be one shape. You **must** use a method to draw the shapes and may use drawShape from part A.

Requirements for this Assignment:
1) Create TurtleFun.java. Some sample code is provided near the end of this document.

Your program will do the following in the order listed:

1) Create a World for the turtles.

2) Create THREE different Turtle references (variables) **(1 point)**

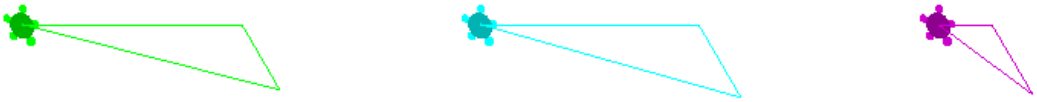
3) Create only TWO turtle objects, and use the assignment operator so that one of these Turtle objects has TWO references pointing to it, while the other has only one **(2 points)**.

4) Draw at least THREE shapes, each one drawn with a DIFFERENT REFERENCE (Hint: penUp(), penDown(), and moveTo() may be helpful here. See pp.71 in the book for more details on these methods) **(1 points)**

NOTE: These three shapes do not have to be the same! You **cannot** do triangles, rectangles, or squares and diamonds for this part. All shapes can share the same color.

Part C: Turtle Random [7 points]

This program requires you to create a program called **CreateTriangles.java** and to create three randomly shaped triangles using Java's Random library, it should look something like this:



As in the example above, make sure that all three triangles are distinguishable and are not touching each other. The shape of the triangle may vary each time you run the program because of the randomness. Each side should have a random length, the length of the third side is determined by how much distance is needed to finish. The turtle can begin moving at any direction. Also, the first angle that the turtle turns can be a set amount. You should use a final variable to define this set amount if needed.

Requirements for this assignment:

1) Add an additional drawTriangle method to the Turtle class.

Some code for this method is provided in the sample code section near the end of this document. The rest of the method needs to be completed. Refer to the comments in the sample code for specifications and hints.

2) Use the x and y parameters to set the starting point of the triangle in drawTriangle method **(1 point).**

3) Use the random parameter limit in the drawTriangle method **(2 point)**.

4) Create CreateTriangles.java. Sbme sample code is provided near the end of this document.

Your program will do the following in the order listed:

1) Create a World for turtles.

2) Create three turtles in that world. **(1 point)**.

3) For each turtle, call the drawTriangle method to draw triangles. Make sure you give enough space on the starting point so that the triangles do not overlap with each other. **(2 points)**.

Once you have completed Part C, at the bottom of CreateTriangles.java write in comment describe 2 scenarios that may cause your code to have an outcome you did not expect. In computer scientists' world, these are called logical error because your code compiles and runs, but does not work as expected. For example, what happens when part of a triangle reaches the end of the canvas? Provide concrete examples of what you passed into drawTriangle to make the program not produce the triangle it was supposed to make. **(1 point)**.

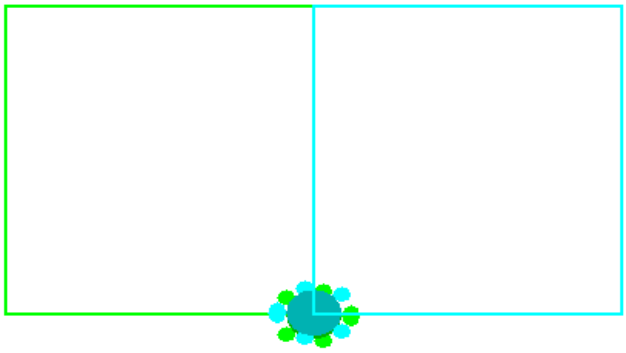
Commenting your code [3 points]

Remember to properly comment your code and place your partner's name if any, emails, logins, and the date in the header of each required file. Keep in mind that you must provide all the history of your partner information from PSA 0 to this PSA. **(3 points).**

Star point [Optional]

If you do any of the following, you will potentially earn a star point. Be sure to **note in your header comment** which of these extensions you have done, if any.

1. Draw an elaborate shape (house, tree, face, 3D shape etc.) by creating a method **elaborateShape**. Whether or not the shape is considered elaborate enough for securing a star point, is up to the judgment and discretion of the graders.
2. Create a single method called **drawStarPointShape** which uses more than one Turtle to draw a single shape. You can't create any Turtle objects inside this method. HINT: you will need to pass at least one Turtle object into your draw method (and thus you will have access to two Turtles--the calling Turtle and the one you passed in). Also both of the turtle should draw half of that shape, so the finished shape should be a symmetric shape. An example is shown below:



Note:

1. You only need to do one of the star point problems to receive star points.
2. You should put your star point method into Turtle.java.
3. Please put all the code for the star points to a new file. It can be named as PSA1StarPoint.java. It contains the main method. And you also need to include it in the psa1 folder you submit.

How to Turn in your Homework

1. Place your psa1 folder in the home directory of your student account (cs8afxx).

2. Ensure that the psa1 folder contains at least all the following files: **CreateShapes.java, TurtleFun.java, CreateTriangles.java,** and **Turtle.java**.

3. Run the command: cse8aturnin psa1

4. Follow the prompts.

5. Verify that your homework was turned in with the command: cse8averify psa1

Refer back to psa0 turnin instructions for additional details.

Sample Code for Part A

```
// This method should be put in Turtle.java, the file you obtained from the public folder.
/* Method : drawShape
 * Created by: Shuaiqi Xia, cs8a3 and Tony Xia, cs8a4
 * Date:
 */
public void drawShape(int size, int thickness)
{
   //Put all your commands to use the size and thickness parameters
   //to make one copy of your shape. BE CREATIVE!
   //Then REMOVE THESE COMMENTS.
}


/* Filename: CreateShapes.java
* Created by: Shuaiqi Xia, cs8a3 and Tony Xia, cs8a4
* Date: 4/6/2016
*/

public class CreateShapes
{
   //The line below is magic, you don't have to understand it (yet)
   public static void main (String[] args)
   {
      //Put all your commands to call drawShape 4 times on 4 different Turtle objects
      //Then REMOVE THIS COMMENT.
   }
}
```

Sample Code for Part B

```
/* Filename: TurtleFun.java
 * Created by: Shuaiqi Xia, cs8a3 and Tony Xia, cs8a4
 * Date:
 */

public class TurtleFun
{
   //The line below is magic, you don't have to understand it (yet)
   public static void main (String[] args)
   {
      //Put all your commands to call drawShape at least 3 times with 3 different Turtle
      //references using 2 Turtle objects
      //Then REMOVE THESE COMMENTS.
   }
}
```

Sample Code for Part C

```
/* Method : drawTriangle
 * Created by: PSA Group
 * Date: Fall 2017
 */
public void drawTriangle(Random random, int x, int y, int limit)
{
   //Use x and y as the starting point of the triangle
   //Generate random numbers up to l imit and use them to move the turtle.
   //It is up to you to determine how to draw your triangles. Your triangle does not have to look
   //like the ones in this instruction. The triangle this method draws can start at a different
   //direction every time this method is called.
   //Then REMOVE THESE COMMENTS
}

/* Filename: CreateTriangles.java
 * Created by: PSA Group
 * Date: Fall 2017
 */
import java.util.Random;

public class CreateTriangles
{
   //The line below is magic, you don't have to understand it (yet)
   public static void main (String[] args)
   {
      //Put all your commands to call drawTriangle 3 times on 3 different turtle objects
      //Make sure that these three triangles do not overlap anywhere.
      //Then REMOVE THESE COMMENTS.
   }
}
```