CPSC 1181 – Assignment 5 Part 1

Submission:

- **This is the first part of a multi-part assignment.**
  - The Part 1 submission is not graded, **but is required.**
  - Part 2 will build upon this and be the final graded submission for Assignment 5
  - If you don't make this submission, the final submission for Assignment 5 will be penalized 25%
  - We expect you to honestly attempt to complete at least two thirds of Part 1 in the initial submission. You can correct and complete your work in the final submission.
  - Penalties for submitting late, not submitting in a ZIP file/etc. will be applied to your final submission
  - **NO EXTENSIONS ALLOWED.** Submit what you have done by the due date.
- Submit a **zip file** containing **only** the .java files to Brightspace prior to the **due date set in Brightspace**. Do not include class or other files
- Submissions that are less than 24 hours late receive a 1% per hour late penalty **applied to the final submission**. Submission that are more than 24 hours late will not be accepted.
- Submissions that are unzipped or that contain .class or other unneeded files will be penalized.
- **IMPORTANT: If you are unsure of your submission for any reason, submit it AND email it to me.**

Working With a Partner

- You must tell me **in-person** with your partner that you are working as partners to have permission. This must be done before the assignment is posted.
- If one partner no longer has an extension, neither partner can use an extension.
- Only one group submission is required: submitted by either partner
  - If both submit, we will choose which to mark
- **Add a second @author to your JavaDocYou must inform me if you are not working with your partner from the previous assignment.**

JavaFX Notes

It is important, even more so than usual, to follow the methods taught in the slides/lecture for JavaFX. If you stray from them, you will likely receive significant penalties. If there is something you'd like to try and are unsure if it is allowed, contact your instructor. **Use the techniques from the slides.**

JavaFX Shape Classes You Can Use: Rectangles, Ellipses, Lines, Polygons, Polylines (like polygon, but doesn't connect the ends), Arcs, Text.

You are going to create a basic JavaFX Application. In this application you are going to create a background and then draw some sort of item on top of that background. You can choose what is in your background and what your item is to draw on top. You must choose your own background and item. For example, you might make some simple clouds as your background, and draw birds on top. Or, you could make an underwater background and draw submarines on top. This is similar to the lecture example where trees are being drawn onto a simple background. **You are not adding a button the user can click, just the graphics.** It is ok if you use trees if you make them significantly different than the example and don't use any of that code.

Draw Background

Set up the basic JavaFX program. Make sure to give the stage an appropriate title. Make your Scene at least

400x400 pixels so you have space to work with. Then draw your background. Your background must include at least 4 shapes of at least 2 different classes. For example, you could use two rectangles and two ellipses.

Create Foreground Item

(You can skip straight to the next item, but I it will probably be easier for most to do this first)
Now, after drawing the background, draw a foreground item on top. Choose one item to draw. Your item must include at least 6 shapes of at least 3 different classes. One of those must be using the Arc class to draw a partial ellipse. If you use Arc to draw a complete ellipse, it does not count.

Create Inner Class for your Foreground Item

Now, you will create an inner class that can be used to create multiple instances of your foreground item. This will be similar to the inner Tree class from the lecture examples. **You must follow the example where Tree extends Group.**

- Keep all of the item's shape elements as data members. In the future we may add methods to this class that need to access them.
- Create a single constructor that takes the x and y coordinates, width, and height, of the item to be drawn.
    - The x and y coordinates can be negative
    - As the width and height change, all of the elements should change the same amount. So, if the width is doubled, everything should be twice as wide.
        - Just do this directly by creating the shapes with the correct values. Don't use JavaFX methods to stretch/shrink an already created shape.
    - You may (but don't need to) add more parameters to the constructor to change the color or some other aspect of the item.

Randomly Place Two or More Items on the Background

Whenever the program is run, randomly place at least two of your foreground items onto the background.

- They should have a random location and random size.

Style, Documentation

You only need to document the outer class at the top of the file. No documentation needed at the method level.