C212/A592 Spring 17 Lab 9

Intro to Software Systems

Instructions:

- Review the requirements given below and Complete your work. Please submit all files through Canvas.
- The grading scheme is provided on Canvas

Lab9: Bouncing Balls Game

- We will be removing the functionality of drawing Squares, Ellipses, Rectangles and Triangle (just comment those key events out)
- The application draws Circles when the C key is pressed
- For this application, as soon as the ball is drawn it should start moving
    - The Circles should bounce off the perimeter of the frame
    - The Circles of different color should bounce off each other, and when they do they should swap colors
    - The Circle of same Color will merge together by absorbing the smaller circle into the bigger circle. The size and the direction of the larger circle may not change.
- ShapeDriver will now need a *Timer*, and will also need to implement the *ActionListener* interface
    - Add the following to you ShapeDriver class:

```
import javax.swing.Timer;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class ShapeDriver implements ActionListener {

        private Timer timer

        public ShapeDriver() {
                // the second argument to the Timer Constructor takes an ActionListener
                // the this key word informs the JVM to look inside this class for
                // the actionPerformed method that must be overridden when
                // ActionListener is implemented
                // Every tick of the clock will now run the actionPerformed method
                timer = new Timer(1000/60, this);
                timer.start();
        }

        // Method that must be implemented since the class implements ActionListener
        public void actionPerformed(ActionEvent e) {
                // move each circle
                // check if circle is in bounds, and bounce off the borders if need be
                // check if circle hits another circle of different color,
                // bounce circles off each other and swap colors
```

```
                // if circle of one color hits a circle of same color, remove smaller circle

                // call repaint
                this.repaint();
        }
}
```

- **Note:** The x and y location is actually in the top left hand corner of the circle
  - This is how the AWT Graphics draws a circle
  - In the Circle class, I created another Point called center
  - When moving the location of the circle I also updated the center location
    - this makes collision detection better than using the x and y location for drawing the circle
  - First just work towards using the *location Point* in the Shape class
    - Once this is working, add the center Point to your Circle class and update its x and y value the same as you update *location*
    - Then use *center* to calculate the distance
  - For example, in the Circle Class:
    - I added *Point center* instance field
    - I overrided the *move()* method from the Shape class
    - I called super.move() inside of *move()* to do what the method was doing in the super class, then updated the center field local to the Circle class
  - **Note:** Remember subclasses do not have access to private super class fields and methods
  - Adhere to good Object Oriented Principles:
    - keep data in classes private, and access with get and set methods
    - only have methods public if other classes use them. Make them private if only the class needs it