CS: 114 **OPPDA** **Spring, 2025**

Project6 (100 points) **Due 4/16/2025**

Submit your solutions to canvas. For programming assignments do not send the entire project. All that is needed are the files ending in .java. Each class will be defined in its own separate .java file. All driver code should be put in one .java file. Please make sure your name is included at the top of each .java file. Make one zip file that includes all the .java files and submit that one zip file to canvas.

Project Goals

1. Design and implement your own superclasses and subclasses for an OO 2D graphics program. Utilize inheritance, and polymorphism.
    a. Supply class diagrams
    b. Show the class relationships of the class diagrams
2. Program an interactive graphics program via the **event loop**
    a. This is accomplished by registering event callback methods

Problem1

You are to create an interactive 2D graphics program. Please download `Lect10_driver.java` from canvas and rename it `Project6_driver.java`. It is a framework for you to build your 2D graphics program. It refers to 3 classes. The three classes are:

1. `CanvasFrame`(window) class
    - Constructs the frame(window) object
    - Calls the constructor for the panel(canvas) object
2. `CanvasPanel`(draw area) class
    - Constructs the panel object
    - Constructs two circles
    - Registers the keyboard callback methods `keyPressed` and `keyReleased`
    - Sets up the simulate/render loop via the Swing `Timer` class
3. `Circle` class
    - A hard-coded class for a `Circle` class

This framework will be used for your final project which will be a 2D interactive game. Some possible games you might consider are pong, breakout, a car driving game, pac-man, or tetris. If you have another idea, please see me for approval.

One requirement you will have to meet in this project is to implement an abstract `Shape2D` superclass and create a subclass for each of the various shapes. You are required to support 3 different shapes. For this assignment your system should support rectangle, circle, and oval.

For the next assignment plan to add polygon, an extended polygon, and an animated sprite. The `Shape2D` class will be `abstract` and it will contain a polymorphic method called `Draw` with a method prototype that looks like this:

```
public abstract void Draw(Graphics g);
```

Please examine the code from lect10 for details. Each subclass of `Shape2D` will implement a concrete `Draw` method. Here is a link to all the possible things you can draw:

https://docs.oracle.com/javase/7/docs/api/java/awt/Graphics.html

You are to create 3 subclasses from `Shape2D`. They are: `Circle2D`, `Rectangle2D`, and `Oval2D`. The first subclass to build should be `Circle2D`. You are to look at the `Circle` class from lect10 and separate it into two classes. Those two classes are `Shape2D`, and `Circle2D`. Any data members and methods that will be used by all shapes should go into the `Shape2D` class. Data members and methods that are specific to a specific subclass should go into that specific subclass. For example, diameter should go into `Circle2D`, and `xPos`, `yPos`, `fillColor`, and `fillColorIndex` should all go into the `Shape2D` class. You should be able to render each shape as filled, outlined, or both. There should be a filled color index, and an associated filled color of type `Color`.

Rename `CanvasPanel` to `CanvasPanel_P6` Add a data structure to `CanvasPanel_P6` that will hold the 2D shapes of your scene. You should use an `ArrayList` or a `LinkedList`. You will be replacing this code:

```
private Circle circle1;
private Circle circle2;
```

with this:

```
private List<Shape2D> shapesList;
```

The `CanvasPanel_P6` class contains code that registers a callback for keyboard events. Those callback methods are called: `keyPressed` and `keyReleased.` You can use the key to spawn off behaviors and actions.

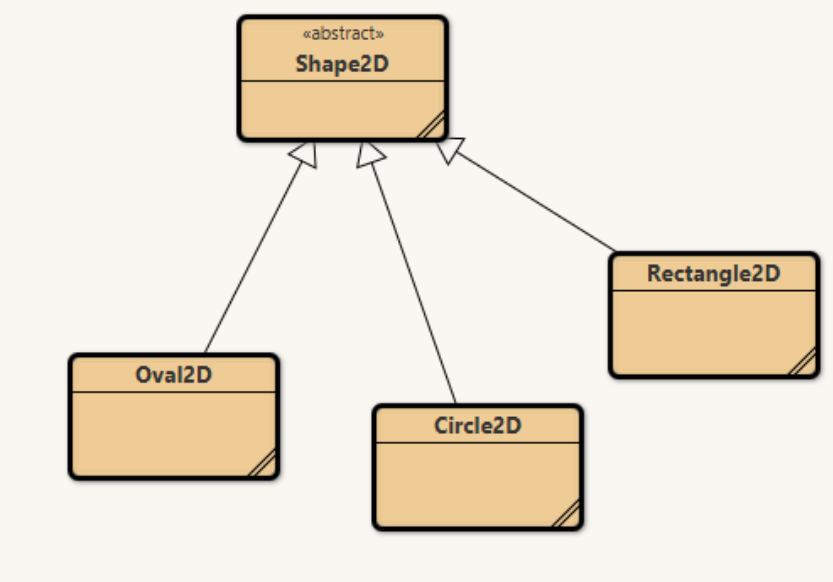
You are to turn in class diagrams, and class relationships, along with a working interactive 2D graphics program that shows the three required shapes moving across the screen. You are welcome to create more than 3 shapes (i.e. 3 circles, 4 rectangles, 1 oval) and store them in in shapes data structure. You also can use the keys to start and stop action. Your code should be documented such that javadocs can easily be generated.

It will make your code more readable to add these constants at the top of the `Shape2D` class

```
public final static int RED     = 0;
public final static int GREEN   = 1;
public final static int BLUE    = 2;
public final static int BLACK   = 3;
public final static int GREY    = 4;
public final static int WHITE   = 5;
public final static int YELLOW  = 6;
public final static int CYAN    = 7;
public final static int MAGENTA = 8;
public final static int BROWN   = 9;
```

Here are some common data members that you might consider for all Shape2d objects that probably should have getters and setters:

```
private int     xPos;               // xPos, location on scree for x
private int     yPos;               // yPos, location on screen for y
private int     xVel;               // xVel, velocity in x direction
private int     yVel;               // yVel, velocity in y direction
private Color   fillColor;          // the fill color for Draw
private int     fillColorIndex;     // the index of the color
private Color   outlineColor;       // the outline color for Draw
private int     outlineColorIndex;  // the outline color index
private boolean fill;               // fill flag on/true, off/false
private boolean outline;            // outline flag on/true, off/false
```



Useful methods for rendering:

```
g.setColor
g.drawOval
```

```
g.fillOval
g.drawRect
g.fillRect
```

Problem2

Please use the following code to test you system and get an image of the Blockhead below.

```
// Create a blockhead from rectangles, circles and ovals
//                       color  x   y  width height
shapes.add(new Rectangle2D(2, 145, 50, 100, 140));
shapes.add(new Rectangle2D(7, 185, 120, 20, 20));
//                  color x   y   d1  d2
shapes.add(new Oval2D(6, 200, 90, 40, 20));
shapes.add(new Oval2D(6, 150, 90, 40, 20));
shapes.add(new Rectangle2D(0, 165, 150, 60, 20));
//                   color x   y diameter
shapes.add(new Circle2D(3,160, 93, 15));
shapes.add(new Circle2D(3,215, 93, 15));
```

The render loop should look something like this and appear at the bottom of the `paintComponent` method:

```
for (Shape2D shape : shapes)
{
   shape.Draw(g);
}
```