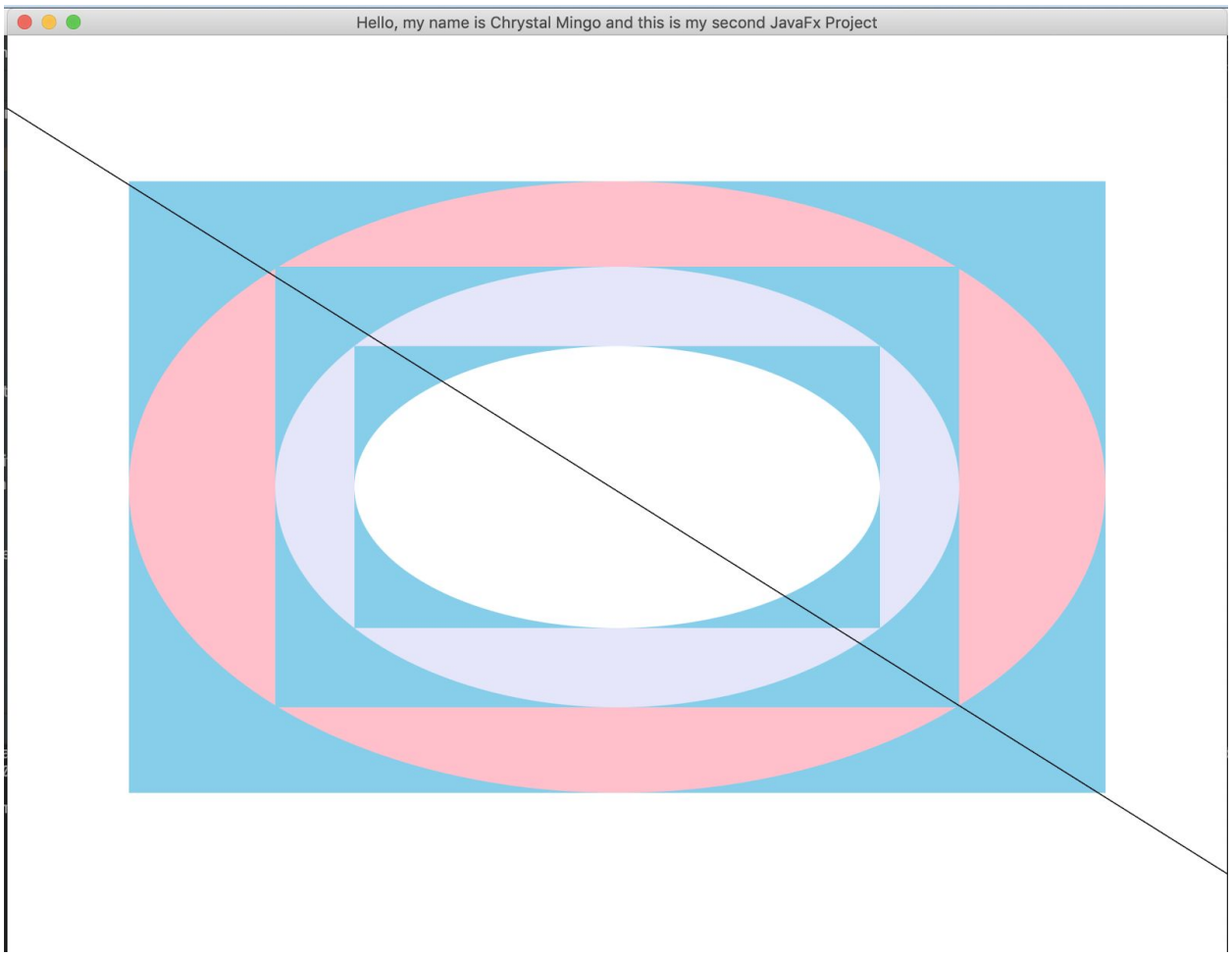
Chrystal Mingo

CSC 22100

Professor Auda

Output for Project 2:



Below I will include all of my classes and there is also comments within each class:

xxxShape.java

```
package sample;
import javafx.scene.canvas.GraphicsContext;
import javafx.scene.paint.Color;
public abstract class xxxShape implements xxxPositionInterface {
    private double x;
    private double y;
    private Color color;
    public xxxShape(double x, double y, Color color) {
        // TODO Auto-generated constructor stuff
        this.x = x;
        this.y = y;
        this.color = color;
    }
    //setX, setY, setColor – set the point (x, y) and color for the xxxShape object;
    public void setX(double X) {
        this.x = X;
    }
    public void setY(double Y) {
        this.y = Y;
    }
    public void setColor(Color color) {
        this.color = color;
    }
    //getX, getY, getColor – return the point (x, y) and color of the xxxShape object;
    public double getX(){
        return this.x;
    }
```

```
    public double getY(){
        return this.y;
    }
    public Color getColor(){
        return this.color;
    }
//Gets points similar to toString function from previous project
    public String getPoint(){
        return " The x is :" + getX() + "The y is :" +getY();
    }
//Moves functions
    public void moveTo(double AddX, double AddY){
        this.x = this.x + AddX;
        this.y = this.y+ AddY;
    }
    //toString() returns the object's description as a String
    public abstract String toString();
    //draws on canvas
    public abstract void draw(GraphicsContext gc);
}
```

xxxOval.java

//Oval was quite similar to the xxxCircle, just needed to set and get a second radius and make changes depending on this new addition of this second radius. I added functions to get and set RadiusOne and RadiusTwo. I also made changes to the ToString, when it came to how to calculate the area and perimeter of an oval. I also implemented the getBounded() and doOverlap() function here as well.

```
package sample;
import javafx.scene.canvas.GraphicsContext;
import javafx.scene.paint.Color;
public class xxxOval extends xxxShape {
    private double radiusOne;
    private double radiusTwo;
    private double area; //need for toString();
    private double perimeter;
    private double x;
    private double y;
    private Color color;
    public xxxOval(double x, double y, Color color, double r1, double r2){
        // TODO Auto-generated constructor stub
        super( x, y, color);
        this.radiusOne = r1;
        this.radiusTwo = r2;
        this.x = x;
        this.y = y;
        this.color = color;
    }
    //getRadius — returns the radius of the xxxCircle object;
    //setRadius — sets the radius of the xxxCircle object;
    public void setRadiusOne(double r1) {
```

```
        this.radiusOne = r1;
    }

    public void setRadiusTwo(double r2) {

        this.radiusTwo = r2;
    }
    public double getRadiusOne() {
        return this.radiusOne;
    }
    public double getRadiusTwo() {
        return this.radiusTwo;
    }
    //I made a setter and getter function to collect the area and perimeter needed for the toString()
function
    public void setArea(double a) {
        this.area = a;
    }
    public void setPerimeter(double p) {
        this.perimeter = p;
    }
    public double getArea() {
        area = Math.PI*radiusOne*radiusTwo; //formula of area of a oval
        return area;
    }
    public double getPerimeter() {//formula for the perimeter of a oval
        perimeter = 2*Math.PI*Math.sqrt((Math.pow(radiusOne,2)+Math.pow(radiusTwo,2)/2));
        return perimeter;
    }
```

```
    //returns a string representation of the xxxCircle object: radius,
    //perimeter, and area;
    public String toString(){
        return "Radius One :" + getRadiusOne() + "Radius Two" + getRadiusTwo() + " Area :" +
getArea() + " Perimeter :" +getPerimeter();
    }
    //draws a xxxCircle object of radius radius. The center point of the circle is
    //defined in class xxxShape.
    public void draw(GraphicsContext gc){
        gc.setFill(color);
        gc.setStroke(color);
        gc.strokeOval(x, y, radiusOne, radiusTwo);
        gc.fillOval(x, y, radiusOne, radiusTwo);
    }
    public xxxRectangle getBounded(){
        double uX = x;
        double lX = y;
        double w = radiusOne;
        double h = radiusTwo;
        Color c = color;
        xxxRectangle rect = new xxxRectangle(uX,lX,w,h,c.SKYBLUE);
        return rect;
    }
    public boolean doOverLap(xxxRectangle R1){
        xxxRectangle R2 = this.getBounded();
        if(R2.getUpperX() + R2.getWidth() <= R1.getUpperX()){
            return false;
        }
        if(R1.getUpperX() + R1.getWidth() <= R2.getUpperX()){
```

```
            return false;
        }
        if(R2.getUpperY()+ R2.getHeight() <= R1.getUpperY()){
            return false;
        }
        if(R1.getUpperY() + R1.getHeight() <= R2.getUpperY()){
            return false;
        }
        return true;
    }
}
```

xxxRectangle.java

// for the rectangle class, I followed the same structure as I did with the polygon, circle, and line. I initialized all the variables I would need to implement the rectangle, created setters and getter for each variable. Also created a draw() function using fill and stroke rect() function. Calculated the area and perimeter of a rectangle and called it in my toString() function. In addition I added getBounded() as well.

```
package sample;
import javafx.scene.canvas.GraphicsContext;
import javafx.scene.paint.Color;
public class xxxRectangle {
    //upperX, lowerX, width, height and color
    private double upperX;
    private double upperY;
    private double width;
    private double height;
    private Color color;
    private double area;
    private double perimeter;
    public xxxRectangle(double uX, double uY, double w,double h, Color color ){
        this.upperX = uX;
        this.upperY = uY;
        this.width = w;
        this.height = h;
        this.color = color;
    }
    //area, perimeter, toString, draw, getBounding
    public void setUpperX(double x1) {
        this.upperX = x1;
    }
    public void setUpperY(double y1) {
```

```
        this.upperY = y1;
    }
    public void setWidth(double w) {
        this.width = w;
    }
    public void setHeight(double h) {
        this.height = h;
    }
    public void setColor(Color color) {
        this.color = color;
    }
    public double getUpperX() {
        return this.upperX;
    }
    public double getUpperY() {
        return this.upperY;
    }
    public double getWidth() {
        return this.width;
    }
    public double getHeight() {
       return this.height;
    }
    public Color getColor(Color color) {
        return this.color = color;
    }
    //I made a setter and getter function to collect the area and perimeter needed for the toString()
function
    public void setArea(double a) {
```

```
        this.area = a;
    }
    public void setPerimeter(double p) {
        this.perimeter = p;
    }
    public double getArea() {
        area = width*height; //formula of area of a rectangle
        return area;
    }
    public double getPerimeter() {//formula for the perimeter of a rectangle
        perimeter = 2*height + 2*width;
        return perimeter;
    }
    public String toString(){
        return " Area :" + getArea() + " Perimeter :" +getPerimeter();
    }
    //draws a xxxCircle object of radius radius. The center point of the circle is
    //defined in class xxxShape.
    public void draw(GraphicsContext gc){
        gc.setFill(color);
        gc.setStroke(color);
        gc.fillRect(upperX,upperY,width,height);
        gc.strokeRect(upperX,upperY,width,height);
    }
    public xxxRectangle getBounded() {
        xxxRectangle rect = new xxxRectangle(upperX, upperY, width, height, color);
        return rect;
    }}
```

xxxLine.java

//i used xxxLine that I implemented from the previous project, I added I added getBounded() as well.

```
package sample;
import javafx.scene.canvas.GraphicsContext;
import javafx.scene.paint.Color;
public class xxxline extends xxxShape {
    private double x1;
    private double y1;
    private Color color;
    private double x2;
    private double y2;
    //Set values in constructor;
    public xxxline(double x1, double y1, Color color, double x2, double y2) {
        super(x1, y1, color);
        this.x1 = x1;
        this.y1 = y1;
        this.x2 = x2;
        this.y2 = y2;
        this.color = color;
    }

    //Need a setter for each x1; y1; x2; y2; and color
    public void setX1(double X1) {
        this.x1 = X1;
    }
    public void setY1(double Y1) {
        this.y1 = Y1;
    }
    public void setX2(double X2) {
```

```
        this.x2 = X2;
    }
    public void setY2(double Y2) {
        this.y2 = Y2;
    }
    public void setColor(Color color) {
        this.color = color;
    }
    //After setting it is necessary to implement a getter for each x1; y1; x2; y2; and color
    public double getX1() {
        return this.x1;
    }
    public double getY1() {
        return this.y1;
    }
    public double getX2() {
        return this.x2;
    }
    public double getY2() {
        return this.y2;
    }
    public Color getColor() {
        return this.color;
    }
    //I implemented this function called getDistance() to calculate the length, that would be called
in the toString() function.
    public double getDistance() { //calculating distance = length
        double length;
        length = Math.sqrt(Math.pow((x2 - x1), 2) + Math.pow((y2 - y1), 2)); //distance formula
```

```
        return length;
    }
    //I implemented this function called getAngle() to calculate the angle of the line, that would be called in the toString() function.
    public double getAngle() {
        double angle;
        angle = (y2 - y1) * (x2 - x1); //calculations for angle --> formula is the same as slope
        return angle;
    }
    //toString — returns a string representation of the xxxLine object: length and angle with the x-axis;
    public String toString(){ //put length and angle into toString
        return "Length :" + getDistance() + " Angle :" + getAngle();
    }
    //draw — draws a xxxLine object [(x1, y1), (x2, y2)].
    public void draw(GraphicsContext gc){ //fill body with the necessary code to draw a line
        gc.setStroke(color);
        gc.strokeLine(x1, y1, x2, y2);
    }
    public xxxRectangle getBounded(){
        double uX = x1;
        double uY = y1;
        double w = distanceTo(x1,0.0,x2,0.0);
        double h = distanceTo(0.0,y1,0.0,y2);;
        Color c = color;
        xxxRectangle rect = new xxxRectangle(uX,uY,w,h,c.SKYBLUE);
        return rect;    }
}
```

xxxPositionInterface.java

//implemented the interface and added the getPointMethod(), moveTo(), and implemented distanceTo() using default

```
package sample;
interface xxxPositionInterface {
   public String getPoint();
   public void moveTo(double AddX, double AddY);
   default double distanceTo(double x1, double y1, double x2, double y2) { //calculating distance
      double length;
      length = Math.sqrt(Math.pow((x2 - x1), 2) + Math.pow((y2 - y1), 2)); //distance formula
      return length;
   }}
```

xxxShapePositionInterface.java

//implemented the doOverLap() function here

```
package sample;
 interface xxxShapePositionInterface extends xxxPositionInterface {
     public xxxRectangle getBoundingBox();
     default boolean doOverLap(xxxRectangle R1){
        xxxRectangle R2 = this.getBoundingBox();
        if(distanceTo(R2.getUpperX(),0.0,R1.getUpperX(),0.0) >= R2.getWidth()){
           if(distanceTo(0,R2.getUpperY(),0,R1.getUpperY()) >= R2.getHeight()){
              return true;
           }
        }
        if(distanceTo(R2.getUpperX(),0,R1.getUpperX(),0) >= R1.getWidth()){
           if(distanceTo(0,R2.getUpperY(),0,R1.getUpperY()) >= R1.getHeight()){
              return true;
           } }
        return false;}}
```

xxxCircle.java

//i used xxxCircle that I implemented from the previous project, I added I added getBounded() as well.

```
package sample;
import javafx.scene.canvas.GraphicsContext;
import javafx.scene.paint.Color;
public class xxxCircle extends xxxShape {
    private double radius;
    private double area; //need for toString();
    private double perimeter;
    private double x;
    private double y;
    private Color color;
    public xxxCircle(double r, double x, double y, Color color){
        // TODO Auto-generated constructor stub
        super(x,y,color);
        this.radius = r;
        this.x = x;
        this.y = y;
        this.color = color;
    }
    //getRadius — returns the radius of the xxxCircle object;
    //setRadius — sets the radius of the xxxCircle object;
    public void setRadius(double r) {
        this.radius = r;
    }
    public double getRadius() {
        return this.radius;
    }
```

```
    //I made a setter and getter function to collect the area and perimeter needed for the toString()
function
    public void setArea(double a) {
        this.area = a;
    }
    public void setPerimeter(double p) {
        this.perimeter = p;
    }
    public double getArea() {
        area = Math.PI*radius*radius; //formula of area of a circle
        return area;
    }
    public double getPerimeter() {//formula for the perimeter of a circle
        perimeter = 2*Math.PI*radius;
        return perimeter;
    }
    //returns a string representation of the xxxCircle object: radius,
    //perimeter, and area;
    public String toString(){
        return "Radius :" + getRadius() + " Area :" + getArea() + " Perimeter :" +getPerimeter();
    }
    //draws a xxxCircle object of radius radius. The center point of the circle is
    //defined in class xxxShape.
    public void draw(GraphicsContext gc){
        gc.setFill(color);
        gc.setStroke(color);
        gc.strokeOval(x, y, radius, radius);
        gc.fillOval(x, y, radius, radius);
    }
```

```
  public xxxRectangle getBounded(){
     double uX = x;
     double uY = y;
     double w = radius;
     double h = radius;
     Color c = color;
     xxxRectangle rect = new xxxRectangle(uX,uY,w,h,c.SKYBLUE);
     return rect;
  }
}
```

xxxPolygon.java

//i used xxxPolygon that I implemented from the previous project, I added I added getBounded() as well.

```
package sample;
import javafx.scene.paint.Color;
import javafx.scene.canvas.GraphicsContext;
public class xxxPolygon extends xxxShape {
    int n; //number of sides
    double sideLength;
    double x;
    double y;
    Color color;
    double radius;
    public xxxPolygon(double x, double y, Color color, double radius, int n) {
        super(x, y, color);
        this.radius = radius;
        this.n = n;
        this.x = x;
        this.y = y;
        this.color = color;
    }
    public double perimeter() {
        return this.n * this.sideLength;
    }
    public double radius() {
        double radii = sideLength / (2 * Math.sin(Math.PI / n));
        return radii;

    }
```

```
    public double apothem() {
        double apothem = (this.sideLength) / (2 * Math.tan(180.0 / this.n));
        return apothem;
    }
    public double area() {
        double area = (perimeter() * apothem()) / 2;
        return area;
    }
    //One of the biggest challenges was the Polygon, setting up an array that collects x and y points.
    //but the hardest part was getting the correct formulas
    public double[] xarray() {
        double[] xpoints = new double[n];
        for (int i = 0; i < n; i++) {
            xpoints[i] = x + radius * Math.cos(2 * Math.PI * i / n);
        }
        return xpoints;
    }
    public double[] yarray() {
        double[] ypoints = new double[n];
        for (int i = 0; i < n; i++) {
            ypoints[i] = y + radius * Math.sin(2 * Math.PI * i / n);
        }
        return ypoints;
    }
    public double interiorAngles() {
        return (this.n - 2) / 180;
    }
```

```
    //returns a string representation of the xxxPolygon object: side length, interior angle,
perimeter, and area;
    public String toString() {
        return "Side Length: " + this.n + " Area: " + area() + " Perimeter: " + perimeter() + " Interior
Angles: " + interiorAngles();
    }
    //draws a xxxPolygon object and inscribed in a circle of radius radius. The center point of the
circle is defined in class xxxShape.
    public void draw(GraphicsContext gc) {
        gc.setFill(color);
        gc.setStroke(color);
        gc.fillPolygon(xarray(), yarray(), n);
        gc.strokePolygon(xarray(), yarray(), n);
        //getChildren().addAll(gc);
    }
    public xxxRectangle getBounded(){
        double uX = x;
        double uY = y;
        double w = radius;
        double h = radius;
        Color c = color;
        xxxRectangle rect = new xxxRectangle(uX,uY,w,h,c.SKYBLUE);
        return rect;
    }
}
```

Main.java

```
//Chrystal Mingo
//Project Two
//CSC 22100
package sample;
import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Group;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.paint.Color;
import javafx.stage.Stage;
import javafx.scene.canvas.Canvas;
import javafx.scene.canvas.GraphicsContext;

public class Main extends Application {

    @Override
    public void start(Stage primaryStage) throws Exception{
        Parent root = FXMLLoader.load(getClass().getResource("sample.fxml"));
        primaryStage.setTitle("Hello, my name is Chrystal Mingo and this is my second JavaFx
Project");
        primaryStage.setScene(new Scene(root, 300, 275));

        //creating my circle objects
        //xxxCircle circle = new xxxCircle(800, 0 + 100, 0 + 25 , Color.PURPLE);
        //xxxCircle circle2 = new xxxCircle(650, 75 + 100, 75 + 25, Color.SKYBLUE);
        //xxxCircle circle3 = new xxxCircle(530, 135 + 100, 135 + 25, Color.LAVENDER);
        //xxxCircle circle4 = new xxxCircle(430, 185 + 100, 185 + 25, Color.WHITE);
```

```
//creating my polygon objects
//xxxPolygon Poly = new xxxPolygon(400 + 100, 400 + 25, Color.PINK, 400,5);
//xxxPolygon Poly2 = new xxxPolygon(400 + 100, 400 + 25, Color.YELLOW, 325,5);
//xxxPolygon Poly3 = new xxxPolygon(400 + 100, 400 + 25, Color.LIGHTBLUE, 265,5);

//creating my line objects
xxxline line = new xxxline(0,60, Color.BLACK, 1500,1000);
//xxxline line2 = new xxxline(0,860, Color.BLACK, 1000,0);

//CREATING MY OVAL OBJECT
xxxOval Oval1 = new xxxOval(100, 120, Color.PINK, 800, 500);
xxxOval Oval2 = new xxxOval(220,  190, Color.LAVENDER, 560, 360);
xxxOval Oval3 = new xxxOval(285, 255, Color.WHITE, 430, 230);

//Creating my rectangle objects
xxxRectangle rect1;
xxxRectangle rect2;
xxxRectangle rect3;

//Bounded Rect
rect1 = Oval1.getBounded();
rect2 = Oval2.getBounded();
rect3 = Oval3.getBounded();

Group group = new Group();
Canvas canvas = new Canvas(1000, 1000);
GraphicsContext gc = canvas.getGraphicsContext2D();
//circle.draw(gc);
```

```
        //Poly.draw(gc);
        //circle2.draw(gc);
        //Poly2.draw(gc);
        //circle3.draw(gc);
        //Poly3.draw(gc);
        //circle4.draw(gc);
        //line2.draw(gc);
        rect1.draw(gc);
        Oval1.draw(gc);
        rect2.draw(gc);
        Oval2.draw(gc);
        rect3.draw(gc);
        Oval3.draw(gc);
        line.draw(gc);
        System.out.println(Oval1.getPoint());
        System.out.print("The distance of point (0,0) to (100,0) is ");
        System.out.println(Oval1.distanceTo(0,0,100,0));
        System.out.print("Does the outer most oval and inner most oval overlap? Answer: ");
        System.out.println(Oval1.doOverLap(rect3));
        group.getChildren().add(canvas);
        Scene circScene = new Scene(group,1000, 1000);
        primaryStage.setScene(circScene);
        primaryStage.show();
    }
    public static void main(String[] args) {
        launch(args);
    }
}
```