Graded Lab 3

Rohan Verma, 1510110508

Point.java

```
package SNU.gr3;

import java.util.*;

public class Point{
        private double x,y;

        public Point(double a, double b){
                this.x = a;
                this.y = b;
        }

        public Point(){
                this(0.0, 0.0);
        }

        public double getX(){
                return this.x;
        }

        public double getY(){
                return this.y;
        }

        public void setX(double z){
                x = z;
        }

        public void setY(double z){
                y = z;
        }

        public void setFromUser(){
                Scanner s = new Scanner(System.in);
                System.out.println("Enter the data for the Point");
                System.out.print("x: ");
                x = s.nextDouble();
                System.out.print("y: ");
                y = s.nextDouble();
        }

        @Override
    public boolean equals(Object o) {

        if (o == this) {
            return true;
        }

        if (!(o instanceof Point)) {
            return false;
        }
```

```
        Point c = (Point) o;

        if(this.getX() == c.getX() && this.getY() == c.getY()){
                return true;
        }
        else
                return false;
    }

}
```

Rectangle.java

```
ackage SNU.gr3;

import java.util.*;

public class Rectangle extends GeometricObject implements
Comparable<GeometricObject>{

        private Point a, b, c, d;
        private double length, breadth;
        private Point centre;

        public Rectangle(Point p, Point q, Point r, Point s){

                if(IsRectangleAnyOrder(p,q,r,s)){
                        this.a = p;
                        this.b = q;
                        this.c = r;
                        this.d = s;

                        length = Math.abs(distance(a,b));
                        breadth = Math.abs(distance(a,d));

                        this.centre.setX((this.a.getX() + this.c.getX())/2);
                        this.centre.setY((this.a.getY() + this.c.getY())/2);

                }
                else{
                        throw new ArithmeticException("Invalid Rectangle");
                }
        }

        public double distance(Point a, Point b){
                return Math.sqrt((a.getX()-b.getX())*(a.getX()-b.getX()) +
(a.getY()-b.getY())*(a.getY()-b.getY()));
        }


        boolean IsOrthogonal(Point a, Point b, Point c)
        {
            return (b.getX() - a.getX()) * (b.getX() - c.getX()) + (b.getY() -
a.getY()) * (b.getY() - c.getY()) == 0;
        }

        boolean IsRectangle(Point a, Point b, Point c, Point d)
        {
            return
```

```
        IsOrthogonal(a, b, c) &&
        IsOrthogonal(b, c, d) &&
        IsOrthogonal(c, d, a);
}

boolean IsRectangleAnyOrder(Point a, Point b, Point c, Point d)
{
    return IsRectangle(a, b, c, d) ||
           IsRectangle(b, c, a, d) ||
           IsRectangle(c, a, b, d);
}

public void setLength(double l){
        length = l;
}

public double getLength(){
        return length;
}

public void setBreadth(double b){
        breadth = b;
}

public double getBreadth(){
        return breadth;
}

public Point getCentre(){
        return centre;
}

public void setFromUser(){
        Scanner s = new Scanner(System.in);
        System.out.println("Enter data for Point a");
        a.setFromUser();
        System.out.println("Enter data for Point b");
        b.setFromUser();
        System.out.println("Enter data for Point c");
        c.setFromUser();
        System.out.println("Enter data for Point d");
        d.setFromUser();

        length = Math.abs(distance(a,b));
        breadth = Math.abs(distance(a,d));

        this.centre.setX((this.a.getX() + this.c.getX())/2);
        this.centre.setY((this.a.getY() + this.c.getY())/2);
}

@Override
public double getPerimeter(){
        return 2*(length+breadth);
};

@Override
public double getArea(){
        return length*breadth;
};

@Override
public int compareTo(GeometricObject o) {

if(this.getArea() > o.getArea()){
```

```
                        return 1;
                }
                else if(this.getArea() < o.getArea()){
                        return -1;
                }
                else
                        return 0;

    }
}
```

Octagon.java

```
package SNU.gr3;

import java.util.*;

public class Octagon extends GeometricObject implements
Comparable<GeometricObject>{

        private Point a, b, c, d, e, f, g, h;
        private double side;

        public Octagon(ArrayList<Point> p){

                double s = distance(p.get(0), p.get(1));

                for(int i = 1; i < 8; i++){
                        if(distance(p.get(i), p.get(i-1)) != s){
                                throw new ArithmeticException("Invalid Octagon!
Not Regular!");
                        }
                }

                        this.a = p.get(0);
                        this.b = p.get(1);
                        this.c = p.get(2);
                        this.d = p.get(3);
                        this.e = p.get(4);
                        this.f = p.get(5);
                        this.g = p.get(6);
                        this.h = p.get(7);

                        this.side = Math.abs(distance(a,b));
        }

        public double distance(Point a, Point b){
                return Math.sqrt((a.getX()-b.getX())*(a.getX()-b.getX()) +
(a.getY()-b.getY())*(a.getY()-b.getY()));
        }

        public void setFromUser(){
                Scanner s = new Scanner(System.in);
                System.out.println("Enter data for Point a");
                a.setFromUser();
                System.out.println("Enter data for Point b");
                b.setFromUser();
                System.out.println("Enter data for Point c");
                c.setFromUser();
                System.out.println("Enter data for Point d");
                d.setFromUser();
                System.out.println("Enter data for Point a");
                e.setFromUser();
```

```
                System.out.println("Enter data for Point b");
                f.setFromUser();
                System.out.println("Enter data for Point c");
                g.setFromUser();
                System.out.println("Enter data for Point d");
                h.setFromUser();

                this.side = Math.abs(distance(a,b));

        }

        @Override
        public double getPerimeter(){
                return 8*(side);
        };

        @Override
        public double getArea(){
                return 2*(1+Math.sqrt(2))*side*side;
        };

        @Override
    public int compareTo(GeometricObject o) {

        if(this.getArea() > o.getArea()){
                        return 1;
                }
                else if(this.getArea() < o.getArea()){
                        return -1;
                }
                else
                        return 0;

    }

}
```

GeometricObject.java

```
package SNU.gr3;

public abstract class GeometricObject implements Comparable<GeometricObject> {
  private String color = "white";
  private boolean filled;
  private java.util.Date dateCreated;

  protected GeometricObject() {
    dateCreated = new java.util.Date();
  }

  protected GeometricObject(String color, boolean filled) {
    dateCreated = new java.util.Date();
    this.color = color;
    this.filled = filled;
  }

  public String getColor() {
    return color;
  }

  public void setColor(String color) {
    this.color = color;
  }
```

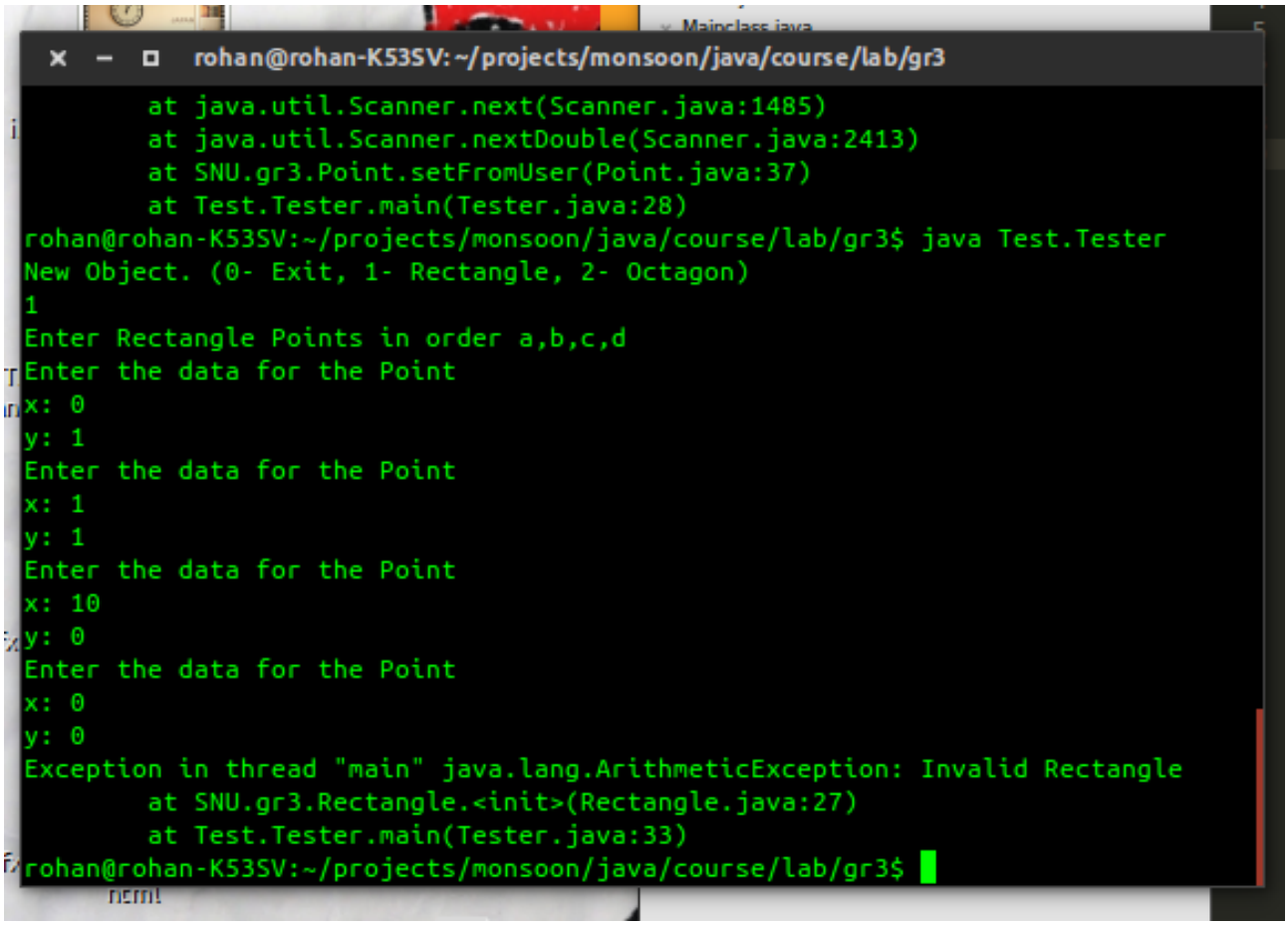
```
  public boolean isFilled() {
    return filled;
  }

  public void setFilled(boolean filled) {
    this.filled = filled;
  }

  public java.util.Date getDateCreated() {
    return dateCreated;
  }

  @Override
  public String toString() {
    return "created on " + dateCreated + "\ncolor: " + color +
      " and filled: " + filled;
  }

  public abstract double getArea();

  public abstract double getPerimeter();

  public abstract int compareTo(GeometricObject o);
}
```

Tester.java

```
package Test;

import java.util.*;
import SNU.gr3.*;

public class Tester {
        public static void main(String[] args) {
                ArrayList<GeometricObject> l = new ArrayList<GeometricObject>();

                Scanner s = new Scanner(System.in);

                boolean exit = false;

                int choice = -1;

                while(!exit){

                        if(l.isEmpty()){
                                System.out.println("New Object. (0- Exit, 1- Rectangle, 2-
                                choice = s.nextInt();

                                if(choice == 0) exit = true;
                                else if(choice == 1){
                                        System.out.println("Enter Rectangle Points in orde

                                        Point a = new Point(),b = new Point(),c = new Poin

                                        a.setFromUser();
                                        b.setFromUser();
                                        c.setFromUser();
                                        d.setFromUser();
```

```
                        Rectangle r = new Rectangle(a,b,c,d);

                        l.add(r);
                }
                else if(choice == 2){
                        System.out.println("Enter Octagon Points in order

                        ArrayList<Point> p = new ArrayList<Point>();
                        for(int i = 0; i < 7; i++){
                                Point t = new Point();
                                t.setFromUser();
                        }

                        Octagon o = new Octagon(p);

                        l.add(o);
                }
                else{
                        System.out.println("Invalid Choice");
                }

        }
        else{
                System.out.println("What to do? (0- Exit, 1- Add New, 2- C
                choice = s.nextInt();

                if(choice == 0) exit = true;

                else if(choice == 1){
                        System.out.println("New Object. (0- Exit, 1- Recta
                        choice = s.nextInt();

                        if(choice == 0) exit = true;
                        else if(choice == 1){
                                System.out.println("Enter Rectangle Points

                                Point a = new Point(),b = new Point(),c =

                                a.setFromUser();
                                b.setFromUser();
                                c.setFromUser();
                                d.setFromUser();

                                Rectangle r = new Rectangle(a,b,c,d);

                                l.add(r);
                        }
                        else if(choice == 2){
                                System.out.println("Enter Octagon Points i

                                ArrayList<Point> p = new ArrayList<Point>(
                                for(int i = 0; i < 7; i++){
                                        Point t = new Point();
                                        t.setFromUser();
                                }

                                Octagon o = new Octagon(p);

                                l.add(o);
                        }
                }
```

```
                        else if(choice == 2){
                                int p1, p2;

                                p1 = s.nextInt();
                                p2 = s.nextInt();

                                if(p1 > l.size() || p2 > l.size())
                                        {//throw new Exception("Index out of Bound
                                        }
                                else{
                                        int rval = l.get(p1).compareTo(l.get(p2));
                                        if(rval == 0){
                                                System.out.println("Same Area");
                                        }
                                        else{
                                                System.out.println("Different Area
                                        }
                                }
                        }

                        else if(choice == 3){
                                GeometricObject m = max(l);

                                System.out.println("The maximum element has area:
                        }

                }

        }

        public static GeometricObject max(ArrayList<GeometricObject> l){

                GeometricObject m = l.get(0);

                for(int i = 1; i < l.size(); i++){
                        if(m.compareTo(l.get(i)) < 0){
                                m = l.get(i);
                        }
                }

                return m;
        }

}
```



Input 1

2 100 107.0711 107.0711 100 90 82.92893 82.92893 90 100 Y 100 107.0711 117.0711 124.1421 124.1421 117.0711 107.0711 100 100

Output

Invalid Octagon