

JAVA-PBL WEEK 3

Write a program to create a Circle class with the following members

Point p; //Point class contains two members double x,y;

double r;

String color;

Date dateCreated; //check Date API

//methods

Constructors (private and public)

Functional Methods

 //calcArea()

 //calcPeri()

 //int compareTwoCircles(Circle c)

 Returns 0 - overlap

 Returns 1 - touch externally

 Returns 2 - disjoint externally

 Returns -1 - touch internally

 Returns -2 - disjoint internally

Circle implements an interface called Comparable

Go through Comparable interface API and implement body of the method compareTo()

 //write the body for compareTo()

 //logic is Circles are compared based on their area

//Class CircleCollection

//members - Circle[]

sort circles in the Array based on increasing order of areas (Arrays.sort(), Collections.sort())

CODE

POINT CLASS

```
// 190031249 P.MOHITH
```

```
package circle_entity;
```

```
public class Point {
    private double x;
    private double y;
    private Point() {
        this.x=0;
        this.y=0;
    }
    public Point(double a,double b) {
        this();
        this.setCoordinate1(a);
        this.setCoordinate2(b);
    }
    public boolean setCoordinate1(double a) {
        if(a>=0 || a<=0) {
            this.x = a;
            return true;
        }
        return false;
    }
    public boolean setCoordinate2(double b) {
        if(b>=0 || b<=0) {
            this.y = b;
            return true;
        }
        return false;
    }
    public double getCoordinate1() {
        return this.x;
    }
    public double getCoordinate2() {
        return this.y;
    }
    public static double distance(Point a,Point b) {
        return Math.sqrt((b.x - a.x) * (b.x - a.x) + (b.y - a.y)*(b.y -
a.y));
    }
}
```

CIRCLE CLASS

```
package circle_entity;
```

```
import java.util.Date;
```

```
public class Circle implements Comparable<Circle> {
    private double r;
    private String color;
    private Date dateCreated;
    private Point p;
    private Circle() {
        this.r = 0;
    }
}
```

```

        this.color = "Red";
        this.dateCreated = new Date();
        this.p = new Point(0,0);
    }
    public Circle(Point p,double r,String c) {
        this();
        this.setCentre(p);
        this.setRadius(r);
        this.setColor(c);
    }
    public boolean setCentre(Point p) {
        if(p != null) {
            this.p = p;
            return true;
        }
        return false;
    }
    public boolean setRadius(double r) {
        if(r>=0) {
            this.r = r;
            return true;
        }
        return false;
    }
    public void setColor(String c) {
        this.color = c;
    }
    public double getRadius() {
        return this.r;
    }
    public String getColor() {
        return this.color;
    }
    public Point getCentre() {
        return this.p;
    }
    public Date getDate() {
        return this.dateCreated;
    }
    public double calcArea() {
        return Math.PI * this.r*this.r;
    }
    public double calcPerimeter() {
        return 2*Math.PI*this.r;
    }
    public int compareTwoCircles(Circle c1,Circle c2) {
        double d = Point.distance(c1.getCentre(), c2.getCentre());
        if(Math.abs(c1.getRadius() - c2.getRadius()) < d && d <
c1.getRadius() + c2.getRadius())
            return 0;
        else if(d == c1.getRadius() + c2.getRadius())
            return 1;
        else if(d > c1.getRadius() + c2.getRadius())
            return 2;
        else if(d == Math.abs(c1.getRadius() - c2.getRadius()))
            return -1;
        else if(d < Math.abs(c1.getRadius() - c2.getRadius()))
            return -2;
        else

```

```

        return -3;
    }
    public int compareTo(Circle c) {
        if(this.calcArea() == c.calcArea())
            return 0;
        else if(this.calcArea() > c.calcArea())
            return 1;
        else
            return -1;
    }
    public String toString() {
        String out = "";
        out+=String.format("Centre of circle = (%f , %f)
%n",this.getCentre().getCoordinate1(),this.getCentre().getCoordinate2());
        out+=String.format("Perimeter = %f %n", this.calcPerimeter());
        out+=String.format("Area = %f %n", this.calcArea());
        out+=String.format("Color: %s %n", this.getColor());
        out+=String.format("Date Created: %s %n", this.getDate());
        return out;
    }
}

```

CIRCLECOLLECTION CLASS

```

package circle_collectors;

import java.util.*;
import circle_entity.*;

public class CircleCollection {
    private static Scanner sc = new Scanner(System.in);
    static ArrayList<Circle> al = new ArrayList<Circle>();
    int count;
    public static void main(String[] args) {
        boolean repeat = true;
        while(repeat) {
            switch(mainMenu()) {
                case 1: insertCircle();
                        break;
                case 2: displayCircleDetails();
                        break;
                case 3: sortCircles();
                        break;
                default: repeat = false;
            }
        }
    }
    private static void insertCircle() {
        System.out.println("Enter centre of circle:");
        System.out.println("Enter x-coordinate: ");
        double x = sc.nextDouble();
        System.out.println("Enter y-coordinate: ");
        double y = sc.nextDouble();
        Point p = new Point(x,y);
        System.out.println("Enter radius of circle: ");
        double r = sc.nextDouble();
        System.out.println("Enter color of circle: ");
        String color = sc.next();
        Circle c = new Circle(p,r,color);
    }
}

```

```

        al.add(c);
    }
    private static int mainMenu() {
        System.out.println("1.Create Circle");
        System.out.println("2.Display circles");
        System.out.println("3.Sort Circles");
        System.out.println("Enter any other number to exit");
        return sc.nextInt();
    }
    private static void sortCircles() {
        Collections.sort(al);
        System.out.println(al);
    }
    private static void displayCircleDetails() {
        System.out.println(al);
    }
    public String toString() {
        String out="";
        for(int i=0;i<count;i++)
            out+=this.al.get(i).toString();
        return out;
    }
}

```

OUTPUT

```

<terminated> CircleCollection [Java Application] C:\Program Files\Java\jdk-13.0.2\bin\javaw.exe (13-Sep-2020, 5:45:33 PM)
1.Create Circle
2.Display circles
3.Sort Circles
Enter any other number to exit
1
Enter centre of circle:
Enter x-coordinate:
4.5
Enter y-coordinate:
5.5
Enter radius of circle:
6
Enter color of circle:
blue
1.Create Circle
2.Display circles
3.Sort Circles
Enter any other number to exit
2
[Centre of circle = (4.500000 , 5.500000)
Perimeter = 37.699112
Area = 113.097336
Color: blue
Date Created: Sun Sep 13 17:45:59 IST 2020
]

```

```
1.Create Circle
2.Display circles
3.Sort Circles
Enter any other number to exit
1
Enter centre of circle:
Enter x-coordinate:
2.5
Enter y-coordinate:
3.5
Enter radius of circle:
5
Enter color of circle:
white
1.Create Circle
2.Display circles
3.Sort Circles
Enter any other number to exit
2
[Centre of circle = (4.500000 , 5.500000)
Perimeter = 37.699112
Area = 113.097336
Color: blue
Date Created: Sun Sep 13 17:45:59 IST 2020
, Centre of circle = (2.500000 , 3.500000)
Perimeter = 31.415927
Area = 78.539816
Color: white
Date Created: Sun Sep 13 17:46:25 IST 2020
, Centre of circle = (4.500000 , 5.500000)
Perimeter = 37.699112
Area = 113.097336
Color: blue
Date Created: Sun Sep 13 17:45:59 IST 2020
]
1.Create Circle
2.Display circles
3.Sort Circles
Enter any other number to exit
5
```