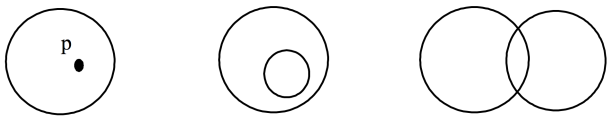
ASDV 2420, MP2
South Louisiana Community College

Problem Description:
Define the Circle2D class that contains:

- Two double data fields named x and y that specify the center of the circle with get methods.
- A data field radius with a get method.
- A no-arg constructor that creates a default circle with (0, 0) for (x, y) and 1 for radius.
- A constructor that creates a circle with the specified x, y, and radius.
- A method getArea() that returns the area of the circle.
- A method getPerimeter() that returns the perimeter of the circle.
- toString()
- equals( Object o ).
- A method contains(double x, double y) that returns true if the specified point (x, y) is inside this circle. See Figure 10.14(a).
- A method contains(Circle2D circle) that returns true if the specified circle is inside this circle. See Figure 10.14(b).
- A method overlaps(Circle2D circle) that returns true if the specified circle overlaps with this circle. See the figure below.




**(a)** **(b)** **(c)**

**Figure**
*(a) A point is inside the circle. (b) A circle is inside another circle. (c) A circle overlaps another circle.*

Write a test program that creates a Circle2D object c1 (new Circle2D(2, 2, 5.5)), displays its area and perimeter, and displays the result of c1.contains(3, 3), c1.contains(new Circle2D(4, 5, 10.5)), and c1.overlaps(new Circle2D(3, 5, 2.3)).

You my use this main for testing contains and ovelaps:

```
public static void main(String[] args) {
  Circle2D c1 = new Circle2D(2, 2, 5.5);
```

```
    System.out.println("Area is " + c1.getArea());
    System.out.println("Perimeter is " + c1.getPerimeter());
    System.out.println(c1.contains(3, 3));
    System.out.println(c1.contains(new Circle2D(4, 5, 10.5)));
    System.out.println(c1.overlaps(new Circle2D(3, 5, 2.3)));
  }
}
```

**A) Implement the class Circle2D**

```
  public boolean contains(double x, double y) {

    double d = distance(x, y, this.x, this.y) ;
    return d <= radius;
  }



  public boolean overlaps(Circle2D circle) {
    // Two circles overlap if the distance between the two centers
    // are less than or equal to this.radius + circle.radius
    // MyPoint is defined in Exercise9_4
    return ?
  }

  private static double distance(double x1, double y1,
      double x2, double y2) {
    return Math.sqrt((x1 - x2) * (x1 - x2) + (y1 - y2) * (y1 - y2));
  }

/**
     * Sort the array in ascending order by perimeter size.
     *
     * @param ar the array to used for sorting, not altered.
     * @return a new new sorted array;
     */
    public static Circle2D[] sortCirclesByPerimeter(Circle2D[] ar)
    {
        return null;
    }

   /**
    * Sort the array in ascending order by the times a circle overlaps all
    * other circles in the array. It returns a new sorted array. You compare
    * every circle with all other circles. If overlaps with none should be
    * placed first. If overlaps with all others it should be placed last.
    *
    * @param ar the array to used for sorting, not altered.
    * @return a new new sorted array;
    */
    public static Circle2D[] sortCirclesByNumberOfTimesOverlapping(Cir-
cle2D[] ar)
    {
        return null;
    }
```