The problem:

In a Cartesian coordinate system, we have 20 shapes. Each of these shapes can be either a circle or a rectangle. The position and size of each circle can be described by a center point and a radius value, while location and size of a rectangle is described by a center point and values for the length and width, having all rectangles’ edges in parallel either to the x-axis or the y-axis.

We need to know whither a point p belongs to the area defined by one shape or more, and find those shapes.

A solution:

Consider the following class diagram which specifies five classes and their relationships:

Implementations of classes Point and TestShape are given.
Determine abstract methods and classes then implement classes: Shape, Circle and Rectangle based on the class diagram and description below:

Class **Shape** has:
- An attributes of type Point, specifies the center of the shape object.
- A constructor that allows to initialize the center attribute with the value of the passed parameter
- A method that takes an object of type Point as a parameter and returns true if the point resides within the shape’s area, and false otherwise.

Class **Circle** has:
- An attribute of type integer specifies the radius measure of the circle
- A constructor that takes a Point parameter to initialize the center and an integer parameter to initialize the radius
- A getRadius method to return the value of the attribute radius
- An overriding version of toString method to return the attribute values of a Circle object as String

Class **Rectangle** has:
- Two integer attributes represents the length and width of the Rectangle object
- A constructor to initialize the center, length and width attribute for a new Rectangle object
- Optional: A constructor that initializes center, length and width given two opposite corners.
- Methods getLength and getWidth returns the values of attributes length and width respectively
- An overriding version of toString method to return the attribute values of a Rectangle object as a String

```java
class Point {
    private int x;
    private int y;

    public Point(int x, int y) {
        this.x = x;
        this.y = y;
    }

    public int getX() { return x; }
    public int getY() { return y; }
    public double distanceTo(Point p) {
        return Math.sqrt( (x-p.getX())*(x-p.getX()) +
                        (y-p.getY())*(y-p.getY()) );
    }
    public String toString() {
        return "("+x+", "+y+")";
    }
}

class TestShape {
    private static Shape[] shapes;

    private static Point getRandPoint() {
        return new Point((int)(Math.random()*201-100),
                        (int)(Math.random()*201-100));
    }
}
```
private static Shape getRandShape(int type) {
    // type is assumed to be 0 or 1
    Point center = getRandPoint();
    if (type == 0)
        return new Circle(center, (int)(Math.random()*101));
    else
        return new Rectangle(center,
            (int)(Math.random()*101),
            (int)(Math.random()*101) );
}

private static Shape getRandShape() {
    int type = (int)(Math.random()*2);
    return getRandShape(type);
}

public static void main(String[] args) {
    shapes = new Shape[20];
    for (int i=0; i<20; i++)
        switch (i%3) {
            case 0: shapes[i] = getRandShape(0); break;
            case 1: shapes[i] = getRandShape(1); break;
            default: shapes[i] = getRandShape();
        }
    Point p = getRandPoint();
    System.out.println("Test shapes against point:"+p.toString());

    System.out.println("\nCircles info:");
    System.out.println("*************");
    for (int i=0; i<20; i++)
        if (shapes[i] instanceof Circle)
            System.out.println(shapes[i].toString()+"\t"+shapes[i].contains(p));

    System.out.println("\nRectangles info:");
    System.out.println("*************");
    for (int i=0; i<20; i++)
        if (shapes[i] instanceof Rectangle)
            System.out.println(shapes[i].toString()+"\t"+shapes[i].contains(p));
}