

TUTORIAL 1: Examining a Java class: SOLUTION

The following Java class represents a circle shape on a diagram. Look at the code and answer the questions overleaf:

```
public class Circle
{
    private String label; // a label to display on the circle
    private int x, y;     // coordinates of centre of circle
    private int radius;  // radius of circle

    public Circle(String label, int x, int y, int radius)
    {
        this.label = label;
        this.x = x;
        this.y = y;
        this.radius = radius;
    }

    public int getRadius()
    {
        return radius;
    }

    public void setRadius(int radius)
    {
        this.radius = radius;
    }

    public String getLabel()
    {
        return label;
    }

    public void move(int dx, int dy)
    {
        x = x + dx;
        y = y + dy;
    }

    public double area()
    {
        return Math.PI * radius * radius;
    }
}
```

1. List the instance variables and methods of the **Circle** class

Instance variables: label, x, y, radius

Methods: getRadius, setRadius, getLabel, move, area

2. What is the effect of the keyword **private** associated with the instance variables?

The variable values cannot be read or changed by other classes

3. Explain the purpose of the methods **getRadius** and **setRadius**.

To allow radius value to be read and changed by other classes (getter and setter methods)

4. There is a **getLabel** method, but no **setLabel** method – what is the result of this?

The value of label can be read but not changed by other classes

5. How can the values of **x** and **y** be changed?

Other classes can only do so by calling the move method

6. What does the word **this** refer to in the constructor of the class? Why is it used here?

this before variable name indicates that variable is an instance variable of the object for which this code is being run -the object being constructed - and distinguishes between instance variable and parameter or local variable with the same name

7. Which *one* of the following is a valid way of creating a **Circle** object?

a. `Circle myCircle = new Circle();`

b. `Circle myCircle = new Circle("MY CIRCLE", 5, 5, 3);`

c. `Circle myCircle = new Circle("MY CIRCLE", 5, 5);`

d. `Circle myCircle = new Circle(5, 5, 3, "MY CIRCLE");`

8. Describe the signature of the **move** method.

name: move, return type: void, parameter list: int, int

9. Assuming a **Circle** object called **myCircle** has been instantiated, which of the following are valid ways of using the object?

- a. **myCircle.move(2,3);**
- b. `int moved = myCircle.move(2,3);`
- c. `myCircle.move(5);`
- d. **myCircle.move(5,4);**
- e. `int myArea = myCircle.area();`
- f. **double myArea = myCircle.area();**
- g. `double myArea = myCircle.area(5);`
- h. **String myLabel = myCircle.getLabel();**
- i. `myCircle.setLabel("MY MODIFIED CIRCLE");`
- j. `myCircle.radius = 5;`
- k. **myCircle.setRadius(5);**

10. You decide that you would like **Circle** objects to be created by default with centre at (0,0) and radius of 5. Write down an additional constructor which would allow this, and a statement which would instantiate a **Circle** object with default values.

```
public Circle()  
{  
    this.label = "DEFAULT";  
    this.x = 0;  
    this.y = 0;  
    this.radius = 0;  
}
```