Abstraction:

```
abstract class Shape {
    String color;
    abstract double area();
    public abstract String toString();

    public Shape(String color)
    {
        System.out.println("Shape constructor called");
        this.color = color;
    }

     public String getColor() {
      return color;
 }
}
class Circle extends Shape {
    double radius;

    public Circle(String color, double radius)
    {

         super(color);
        System.out.println("Circle constructor called");
        this.radius = radius;
    }

    @Override double area()
    {
        return Math.PI * Math.pow(radius, 2);
    }

    @Override public String toString()
    {
        return "Circle color is " + super.getColor()
            + "and area is : " + area();
    }
}
class Rectangle extends Shape {

    double length;
    double width;

    public Rectangle(String color, double length,
                     double width)
    {
```

```
        super(color);
        System.out.println("Rectangle constructor called");
        this.length = length;
        this.width = width;
    }

    @Override double area() { return length * width; }

    @Override public String toString()
    {
        return "Rectangle color is " + super.getColor()
            + "and area is : " + area();
    }
}
public class Test {
    public static void main(String[] args)
    {
        Shape s1 = new Circle("Red", 2.2);
        Shape s2 = new Rectangle("Yellow", 2, 4);

        System.out.println(s1.toString());
        System.out.println(s2.toString());
    }
}
```