```java

//creating interface
interface Shape{

    double calcArea();
    double calcPerimeter();

}


//class Circle implementing Shape interface
class Circle implements Shape{

private int radius;

public Circle( int rad)
{
    this.radius=rad;
}

@Override
    public double calcArea()
{
    return Math.PI * radius * radius;
}

@Override
    public double calcPerimeter()
{
    return 2*Math.PI*radius;
}

}

//Class Rectangle implementing Shape interface
class Rectangle implements Shape{
    private int length;
    private int breadth;

    public Rectangle(int len,int brd)
    {
```

```
42          this.length=len;
43          this.breadth=brd;
44      }
45
46      @Override
47      public double calcArea()
48      {
49          return length*breadth;
50      }
51
52      @Override
53      public double calcPerimeter() {
54          return 2*(length+breadth);
55      }
56
57
58  }
59
60
61  //Implementing Calculate to demonstrate Dependency
    Injection
62  class Calculate{
63
64      private final Shape shape;
65
66      public Calculate( Shape shape)
67      {
68      this.shape=shape;
69      }
70
71      public void displayResults()
72      {
73          System.out.println("Perimeter: " + shape.
    calcPerimeter());
74          System.out.println("Area: " + shape.calcArea
    ());
75      }
76
77  }
78
79  public class Assignment2 {
```

```
 80
 81 public static  void main(String[] args)
 82 {
 83
 84     // Creating a Circle object and passing it to
    Calculate class
 85     Circle circle =new Circle(5);
 86     Calculate calculate = new Calculate(circle);
 87     System.out.println("For Circle: ");
 88     calculate.displayResults();
 89
 90     // Creating a Rectangle object and passing it to
     Calculate class
 91     Rectangle rect = new Rectangle(10,20);
 92     Calculate recCalc = new Calculate( rect);
 93     System.out.println("For Rectangle");
 94     recCalc.displayResults();
 95
 96 }
 97
 98
 99 }
100
```