Main.java

```
/*
- Abstraction
-
- Abstraction is the concept of showing only essential features and hiding non-essential details.
-
- Benefits:
- 1. Simplifies Complex Systems

- 1. Reduces Complexity

- 1. Improves Code Reusability

- 1. Enhances Security

-
- Abstract Class:
- 1. Partial Implementation (provides some implementation)

- 1. Incomplete Class (cannot be instantiated)

- 1. Blueprint for Subclasses

-
- Abstract Method:
- 1. Declaration without Implementation

- 1. Must be Implemented by Subclasses

- 1. Defines Interface for Subclasses

-
- Abstract Keyword:
- 1. Denotes Abstract Classes and Methods

- 1. Indicates Incomplete Implementation

- 1. Forces Subclasses to Implement Abstract Methods

*/

abstract class Shape {
  abstract public double area();

  public void displayMessage() {
  System.out.println("This is an abstract class");
  }
  }

  class Circle extends Shape {
  double radius;

  public Circle(double radius) {
```

```
53    this.radius = radius;
54    }
55
56    @Override
57    public double area() {
58    return 3.14 * (radius * radius);
59    }
60    }
61
62    class Rectangle extends Shape {
63    int length;
64    int width;
65
66    public Rectangle(int length, int width) {
67    this.length = length;
68    this.width = width;
69    }
70
71    @Override
72    public double area() {
73    return length * width;
74    }
75    }
76
77    public class Main {
78    public static void main(String[] args) {
79    Circle myCircle = new Circle(5.8);
80    System.out.println(myCircle.area());
81
82    Rectangle Rec = new Rectangle(8, 5);
83    System.out.println(Rec.area());
84    Rec.displayMessage();
85    }
86    }
```