Lab7

W.A.C Fernando (26545)

1.

```
final class Student {
   final int marks = 100;
   final void display();
}
```

Here, we have a final class "Student". A final class cannot be subclassed, meaning you cannot create any classes that extend this "Student" class. In this class, there is also a final instance variable "marks" with a value of 100, which means its value cannot be changed once initialized. Additionally, there is a final method "display()", which cannot be overridden in any subclasses.

```
class Undergraduate extends Student {}
```

This class "Undergraduate" is trying to extend the "Student" class, but the "Student" class is marked as final, so this will result in a compilation error. You cannot subclass a final class, so trying to extend it is not allowed.

Since "Undergraduate" cannot extend "Student" due to the "final" modifier on the "Student" class, the "Undergraduate" class will not be able to inherit any members or methods from "Student".

2.

```
// Shape class with an abstract method and a non-abstract method
abstract class Shape {
   abstract double calculateArea();

   void display() {
      System.out.println("This is a shape.");
   }
}

// Concrete class representing a Rectangle
class Rectangle extends Shape {
   private double length;
   private double width;

   public Rectangle(double length, double width) {
      this.length = length;
      this.width = width;
   }

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

```
    @Override
    void display() {
        System.out.println("This is a rectangle with length: " + length + " and
width: " + width);
    }
}

// Concrete class representing a Circle
class Circle extends Shape {
    private double radius;

    public Circle(double radius) {
        this.radius = radius;
    }

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

    @Override
    void display() {
        System.out.println("This is a circle with radius: " + radius);
    }
}

public class Main {
    public static void main(String[] args) {
        // Instantiate a Rectangle and calculate/display its area
        Rectangle rectangle = new Rectangle(5.0, 3.0);
        rectangle.display();
        System.out.println("Area of Rectangle: " + rectangle.calculateArea());

        // Instantiate a Circle and calculate/display its area
        Circle circle = new Circle(4.0);
        circle.display();
        System.out.println("Area of Circle: " + circle.calculateArea());
    }
}
```

In this code, the "Shape" class is an abstract class with an abstract method "calculateArea()" and a non-abstract method "display()". We then have two concrete classes, "Rectangle" and "Circle", that extend the "Shape" class and provide their own implementations for the "calculateArea()" method. When we instantiate these shapes, we pass the required values (e.g., length, width, radius) in the constructor of each shape class