

Experiment No. 06
Experiment Name: Experiment with OOP Features

Course title: Programming Language II(Java) Lab
Course code:
Spring 2025

Date of Submission:



Submitted to-

Md. Rafsan Jani
Assistant Professor
Department of Computer Science and Engineering

Sl	Class Roll	Name
01	2023000010034	Md Arafat Rahman

Hw 1: Create a Student class with private fields:

name,

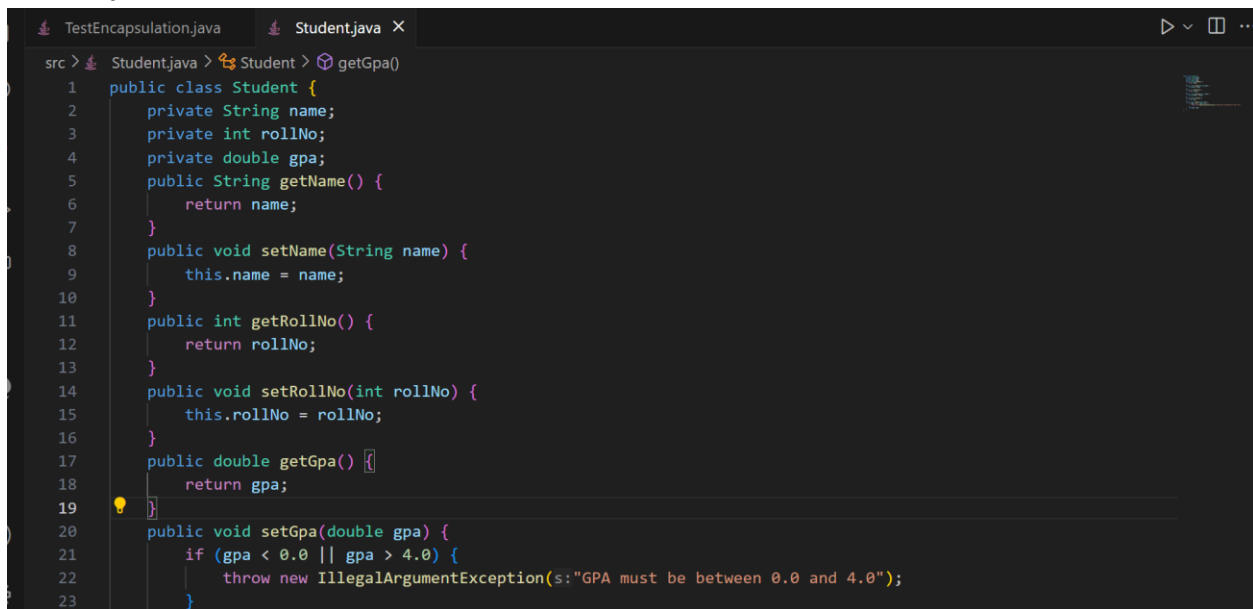
rollNo,

gpa

Provide public getters and setters.

Validate that gpa cannot be negative or over 4.0.

Student.java



```
src > Student.java > Student > getGpa()
1 public class Student {
2     private String name;
3     private int rollNo;
4     private double gpa;
5     public String getName() {
6         return name;
7     }
8     public void setName(String name) {
9         this.name = name;
10    }
11    public int getRollNo() {
12        return rollNo;
13    }
14    public void setRollNo(int rollNo) {
15        this.rollNo = rollNo;
16    }
17    public double getGpa() {
18        return gpa;
19    }
20    public void setGpa(double gpa) {
21        if (gpa < 0.0 || gpa > 4.0) {
22            throw new IllegalArgumentException(s: "GPA must be between 0.0 and 4.0");
23        }
24    }
25 }
```

```
public class Student {
    private String name;
    private int rollNo;
    private double gpa;
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public int getRollNo() {
        return rollNo;
    }
    public void setRollNo(int rollNo) {
        this.rollNo = rollNo;
    }
}
```

```

    public double getGpa() {
        return gpa;
    }
    public void setGpa(double gpa) {
        if (gpa < 0.0 || gpa > 4.0) {
            throw new IllegalArgumentException("GPA must be between 0.0 and
4.0");
        }
        this.gpa = gpa;
    }
}

```

TestEncapsulation.java

```

public class TestEncapsulation {
    public static void main(String[] args) {
        Student obj = new Student();
        obj.setName("Arafat");
        obj.setRollNo(51);
        try {
            obj.setGpa(3.5);
        } catch (IllegalArgumentException e) {
            System.out.println( e.getMessage());
        }
        System.out.println("Student's name: " + obj.getName());
        System.out.println("Student's rollNo: " + obj.getRollNo());
        System.out.println("Student's gpa: " + obj.getGpa());
    }
}

```

Hw 2:

Create a base class Shape with method getArea().

Derive classes

Circle,

Rectangle,

Square, and

Triangle.

Override getArea() appropriately in each subclass.

The screenshot shows an IDE with the following components:

- EXPLORER:** Lists files in the 'src' directory: Circle.java, Rectangle.java, Shape.java, Square.java, TestShapes.java, and Triangle.java.
- TESTSHAPES:** A sub-project containing .vscode, bin, lib, and src folders.
- TestShapes.java:** The main file being edited, containing the following code:

```
1 public class TestShapes {
2     public static void main(String[] args) {
3         Shape circle = new Circle(radius:5);
4         Shape rectangle = new Rectangle(length:4, width:6);
5         Shape square = new Square(side:4);
6         Shape triangle = new Triangle(base:3, height:5);
7
8         System.out.println("Circle Area: " + circle.getArea());
9         System.out.println("Rectangle Area: " + rectangle.getArea());
10        System.out.println("Square Area: " + square.getArea());
11        System.out.println("Triangle Area: " + triangle.getArea());
12    }
13 }
14
```
- TERMINAL:** Shows the output of the program:

```
Run: TestShapes
Square Area: 16.0
Triangle Area: 7.5
```

Shape.java

```
public class Shape {
    public double getArea() {
        return 0.0;
    }
}
```

Circle.java

```
public class Circle extends Shape {

    private double radius;

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

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

Rectangle.java

```
public class Rectangle extends Shape {
```

```

private double length;
private double width;

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

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

```

Square.java

```

public class Square extends Shape {
    private double side;

    public Square(double side) {
        this.side = side;
    }

    @Override
    public double getArea() {
        return side * side;
    }
}

```

Triangle.java

```

public class Triangle extends Shape {
    private double base;
    private double height;

    public Triangle(double base, double height) {
        this.base = base;
        this.height = height;
    }

    @Override
    public double getArea() {
        return 0.5 * base * height;
    }
}

```

```
}
```

TestShapes.java

```
public class TestShapes {  
    public static void main(String[] args) {  
        Shape circle = new Circle(5);  
        Shape rectangle = new Rectangle(4, 6);  
        Shape square = new Square(4);  
        Shape triangle = new Triangle(3, 5);  
  
        System.out.println("Circle Area: " + circle.getArea());  
        System.out.println("Rectangle Area: " + rectangle.getArea());  
        System.out.println("Square Area: " + square.getArea());  
        System.out.println("Triangle Area: " + triangle.getArea());  
    }  
}
```

HW 3: Create a class Animal with a method makeSound().

Create subclasses

Dog,

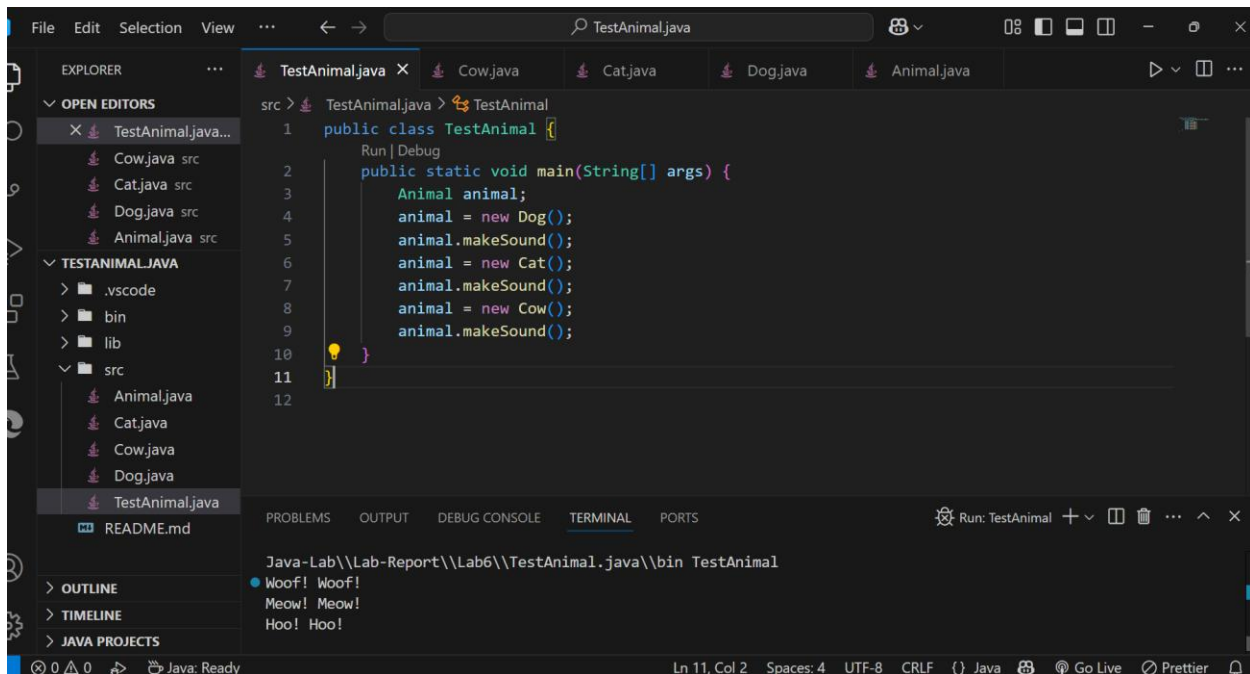
Cat, and

Cow,

override makeSound() to print unique sounds.

Use dynamic method dispatch to demonstrate

polymorphism.



Animal.java

```
public class Animal {
    public void makeSound() {
        System.out.println("Some generic animal sound");
    }
}
```

Dog.java

```
public class Dog extends Animal {
    @Override
    public void makeSound() {
        System.out.println("Woof! Woof!");
    }
}
```

Cat.java

```
public class Cat extends Animal {
    @Override
    public void makeSound() {
        System.out.println("Meow! Meow!");
    }
}
```

Cow.java

```
public class Cow extends Animal {  
    @Override  
    public void makeSound() {  
        System.out.println("Hoo! Hoo!");  
    }  
}
```

TestAnimal.java

```
public class TestAnimal {  
    public static void main(String[] args) {  
        Animal animal;  
        animal = new Dog();  
        animal.makeSound();  
        animal = new Cat();  
        animal.makeSound();  
        animal = new Cow();  
        animal.makeSound();  
    }  
}
```