



**Laiba tanveer**

**69268**

**Oop**

**Lab 12**

**Bscs 2**

## Lab Exercises

## task 1

Create an abstract class Vehicle with an abstract method start(). Implement two subclasses, Car and Motorcycle, that extend Vehicle. Include a parameterized constructor in each subclass. Implement the start() method to print a message indicating the vehicle is starting.

### Program

```
abstract class Vehicle {  
    abstract void start();  
}  
  
class Car extends Vehicle {  
    String model;  
    Car(String model) {  
        this.model = model;  
    }  
    void start() {  
        System.out.println(model + " Car is starting...");  
    }  
}  
  
class Motorcycle extends Vehicle {  
    String brand;  
    Motorcycle(String brand) {  
        this.brand = brand;  
    }  
    void start() {  
        System.out.println(brand + " Motorcycle is starting...");  
    }  
}  
  
class Main {
```

```

public static void main(String[] args) {

    Vehicle car = new Car("Toyota");

    Vehicle bike = new Motorcycle("Honda");

    car.start();

    bike.start();

}

}

```

The screenshot shows the Programiz Online Java Compiler interface. On the left, the code editor contains the provided Java code. On the right, the output window displays the results of running the code, showing the output of the println statements and a success message.

```

Main.java
1 // Online Java Compiler
2 // Use this editor to write, compile and run your Java code online
3
4
5- abstract class Vehicle {
6     abstract void start();
7 }
8- class Car extends Vehicle {
9     String model;
10-    Car(String model) {
11         this.model = model;
12     }
13-    void start() {
14         System.out.println(model + " Car is starting...");
15     }
16 }
17- class Motorcycle extends Vehicle {
18     String brand;
19-    Motorcycle(String brand) {

```

**Output**

```

Toyota Car is starting...
Honda Motorcycle is starting...

== Code Execution Successful ==

```

## Lab Task 2

1. Abstract Class “Seat” o Create an abstract class Seat with an abstract method calculateSeatPrice(int numberOfSeats). o Use throws IllegalArgumentException for seat numbers. Wherever you use it.
2. BusinessClass, FirstClass, and EconomyClass o Extend Seat in three concrete classes: BusinessClass, FirstClass, and EconomyClass. o Each class implements calculateSeatPrice(int numberOfSeats) with different pricing logic.

3. Main Class “AirlineTicketSystem”  
o Create instances of all seat types and call the calculateSeatPrice() method for each.  
o Display the calculated seat prices on the console

**program**

```
abstract class Seat {  
  
    abstract double calculateSeatPrice(int numberOfSeats);  
}  
  
class BusinessClass extends Seat {  
  
    double pricePerSeat = 50000;  
  
    double calculateSeatPrice(int numberOfSeats) {  
        if (numberOfSeats <= 0) {  
            System.out.println("Invalid seat number!");  
            return 0;  
        }  
        return numberOfSeats * pricePerSeat;  
    }  
}  
  
class FirstClass extends Seat {  
  
    double pricePerSeat = 80000;  
  
    double calculateSeatPrice(int numberOfSeats) {  
        if (numberOfSeats <= 0) {  
            System.out.println("Invalid seat number!");  
            return 0;  
        }  
        double total = numberOfSeats * pricePerSeat;  
        double tax = total * 0.10; // 10% luxury tax  
    }  
}
```

```
        return total + tax;  
    }  
  
}  
  
class EconomyClass extends Seat {  
  
    double pricePerSeat = 20000;  
  
    double calculateSeatPrice(int numberOfSeats) {  
        if (numberOfSeats <= 0) {  
            System.out.println("Invalid seat number!");  
            return 0;  
        }  
        if (numberOfSeats > 5) {  
            double total = numberOfSeats * pricePerSeat;  
            double discount = total * 0.05; // 5% discount  
            return total - discount;  
        } else {  
            return numberOfSeats * pricePerSeat;  
        }  
    }  
}  
  
class Main {  
  
    public static void main(String[] args) {  
        Seat business = new BusinessClass();  
        Seat first = new FirstClass();  
        Seat economy = new EconomyClass();
```

```
        System.out.println("Business Class : Rs. " + business.calculateSeatPrice(3));  
        System.out.println("First Class Rs. " + first.calculateSeatPrice(2));  
        System.out.println("Economy Class Rs. " + economy.calculateSeatPrice(6));  
    }  
}
```

The screenshot shows the Programiz Online Java Compiler interface. On the left, there's a sidebar with icons for various languages: Python, C, C++, C#, JavaScript, and Java. The main area has tabs for 'Main.java' and 'Output'. The 'Main.java' tab contains the provided Java code. The 'Output' tab displays the results of the code execution:

```
Business Class : Rs. 150000.0  
First Class Rs. 176000.0  
Economy Class Rs. 114000.0  
== Code Execution Successful ==
```