St. Francis Institute of Technology, Mumbai-400 103

Department Of Information Technology

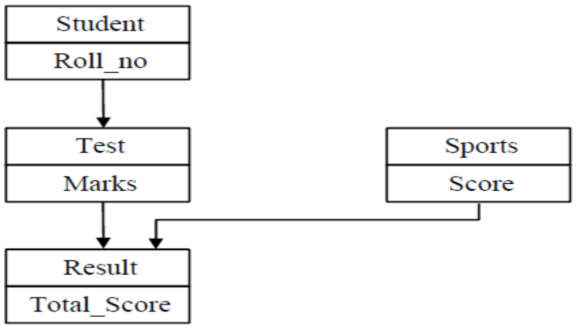
A.Y. 2020-2021

Class: SE-ITA/B, Semester: III

Subject: **Java Labs**

Experiment-4: Java Program to extend and implement Interfaces.

1. **Aim:**
   i. Write a program to implement the following multiple inheritances:



   ii. Create an interface vehicle and classes like bicycle, car, bike etc, having common functionalities and put all the common functionalities in the interface. Classes like Bicycle, Bike, car etc implement all these functionalities in their own class in their own way.

2. **Prerequisite:** Knowledge of Inheritance and Interfaces in Java.

3. **Requirements:** Personal Computer (PC), Windows Operating System, Net beans 8.0.

4. **Pre-Experiment Exercise:**
   Theory:

   a. **Interfaces:**
   An interface is just like Java Class, but it only has static constants and abstract method. Java uses Interface to implement multiple inheritance. A Java class can implement multiple Java Interfaces. All methods in an interface are implicitly public and abstract.
   Syntax for declaring Interface:

```
interface interface_name
{
//Methods
}
```

   Syntax for implementing interfaces:

```
class class_name implements interface_name{}
```

5. Laboratory Exercise

A. Procedure

i. Open Net beans for Java.
ii. Open File and Create New Java Project.
iii. Inside the Java Project rename give name to your Java Class.
iv. Click on Finish.
v. Type the Java Code in the opened class.
vi. Save the code by pressing Ctrl+S.
vii. Run the code by pressing Shift+F6.

B. Program code with comments:

Write and execute your program code to achieve the given aim and attach it **with your own comments with neat indentation**.

6. Post-Experiments Exercise

A. Extended Theory:

1. Explain how to implement Multiple Inheritance in Java with syntax and example.
2. Differentiate between Abstract Class and Interfaces.

B. Results/Observations/Program output:

Present the program input/output results and comment on the same.

C. Questions/Programs:

1. Write a class PoliceCar that implements the IsEmergency and IsLandVehicle interfaces. In addition to the methods you have written for the PoliceCar class, think of a new method or property that police cars have and add it to the class.

D. Conclusion:

1. Write what was performed in the experiment/program.
2. What is the significance of experiment/program?
3. Mention few applications of what was studied.

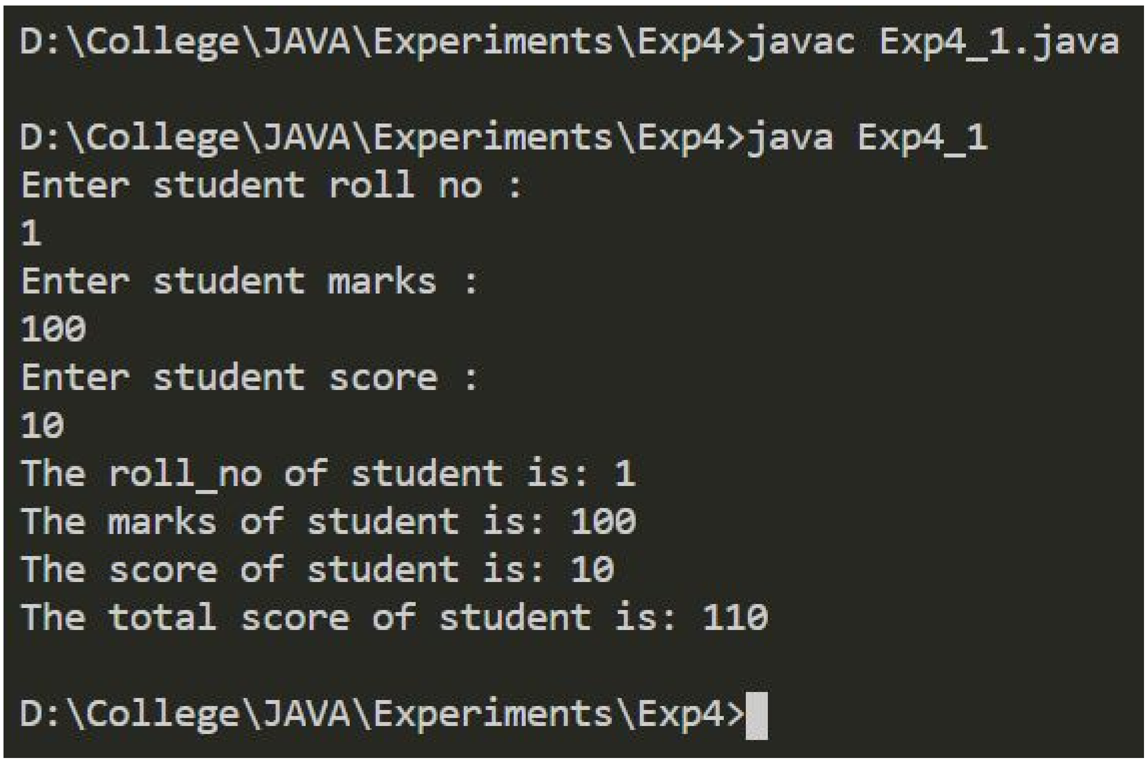
E. References

1. Balguruswamy, “Programming with java A primer”, Fifth edition, Tata McGraw Hill Publication.
2. Let Us Java-Yashwant Kanetkar.
3. Learn to Master JAVA, from Star EDU solutions , by ScriptDemics.
4. Java 8 Programming-Black Book,by-Dreamtech Publications.
5. www.programmingsimplified.com.

Program 1:

```java
import java.util.Scanner;

//defining interface student
interface Student{
    void getRoll_no();
    void setRoll_no();
}
//Interface test inherits interface student
interface Test extends Student{
    void getMarks();
    void setMarks();
}
//interface sports inherits interface Student
interface Sports extends Student{
    void getScore();
    void setScore();
}

//class result inherits interfaces test and sports
class Result implements Test, Sports{

    private int roll_no, marks, score;

    Scanner scanner = new Scanner(System.in);

    //Implementing inherited abstract methods
    public void getRoll_no() {
        System.out.println("The roll_no of student is: " + roll_no);
    }

    public void setRoll_no() {
        System.out.println("Enter student roll no : ");
        roll_no = scanner.nextInt();
    }

    public void getMarks() {
        System.out.println("The marks of student is: " + marks);
    }
    public void setMarks() {
        System.out.println("Enter student marks : ");
        marks = scanner.nextInt();
    }

    public void getScore() {
        System.out.println("The score of student is: " + score);
    }

    public void setScore() {
        System.out.println("Enter student score : ");
        score = scanner.nextInt();
    }
    //Method to display total
    public void getTotal() {
        System.out.println("The total score of student is: " + (marks+score));
    }
}

//Driver class
public class Exp4_1{
    public static void main(String[] args) {

        //Object of class result
        Result res = new Result();
        res.setRoll_no();
        res.setMarks();
        res.setScore();

        //Displaying result
        res.getRoll_no();
        res.getMarks();
        res.getScore();
        res.getTotal();
    }
}
```

Output:

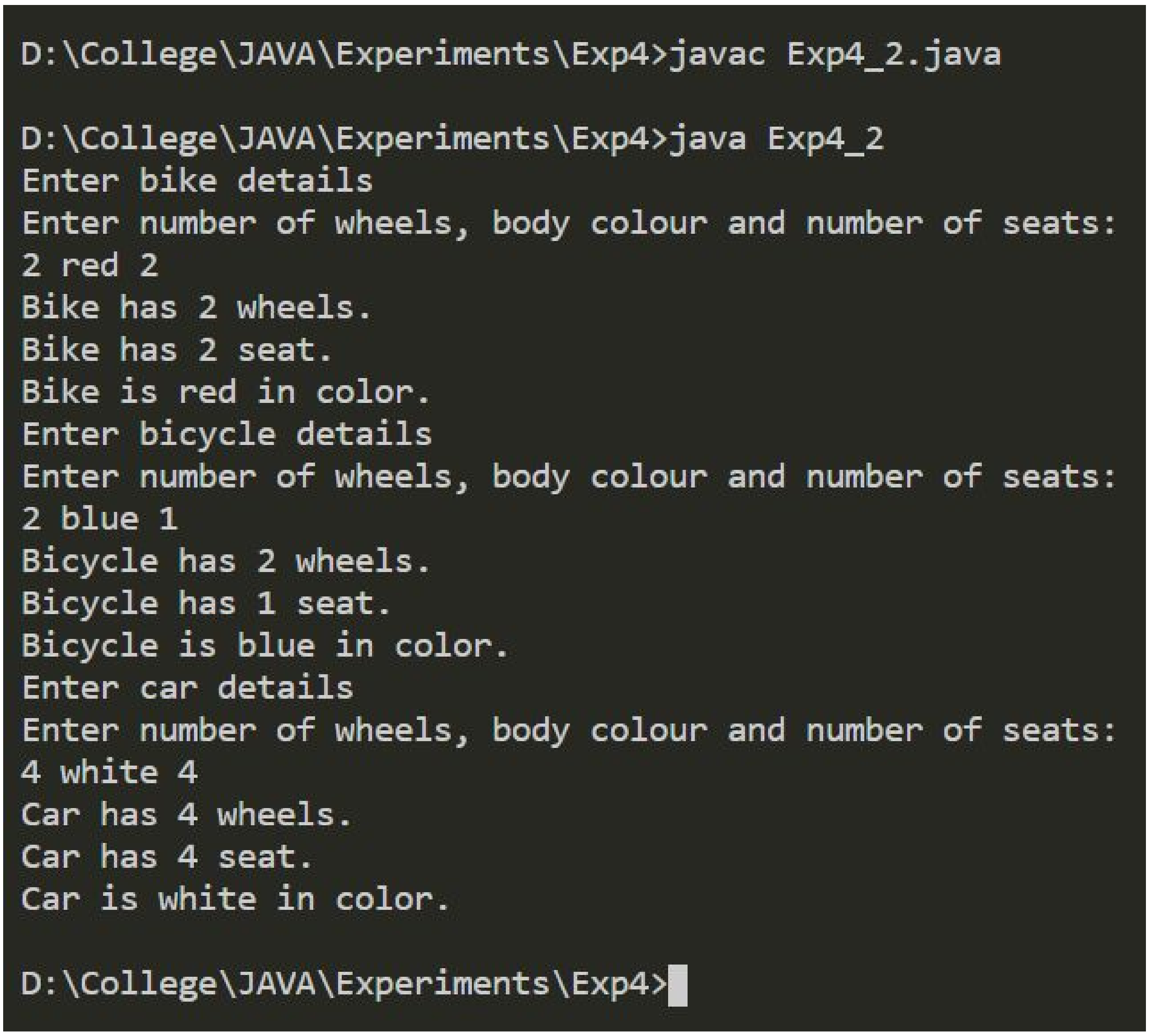


```
D:\College\JAVA\Experiments\Exp4>javac Exp4_1.java

D:\College\JAVA\Experiments\Exp4>java Exp4_1
Enter student roll no :
1
Enter student marks :
100
Enter student score :
10
The roll_no of student is: 1
The marks of student is: 100
The score of student is: 10
The total score of student is: 110

D:\College\JAVA\Experiments\Exp4>
```

Program 2:

```java
/**
 * Create an interface vehicle and classes like bicycle, car, bike etc, having
 *common functionalities and put all the common functionalities in the
 *interface. Classes like Bicycle, Bike, car etc implement all these
 *functionalities in their own class in their own way.
 */
import java.util.Scanner;
//Defining interface Vehicle
interface Vehicle{
    void wheelsCount();
    void bodyColor();
    void seats();
    void setData();
}
//Class bicycle inherits interface vehicle
class Bicycle implements Vehicle{
    private String color;
    private int seats,count;
    //Implementing inherited abstract methods
    Scanner scanner = new Scanner(System.in);
    public void wheelsCount(){
        System.out.println("Bicycle has " + count + " wheels.");
    }
    public void bodyColor(){
        System.out.println("Bicycle is " + color + " in color.");
    }
    public void  seats(){
        System.out.println("Bicycle has " + seats +" seat.");
    }
    public void setData(){
        System.out.println("Enter number of wheels, body colour and number of
seats:");
        count=scanner.nextInt();
        color = scanner.next();
        seats = scanner.nextInt();
    }
}
class Bike implements Vehicle{
    private String color;
    private int seats,count;
    //Implementing inherited abstract methods
    Scanner scanner = new Scanner(System.in);
    public void wheelsCount(){
        System.out.println("Bike has " + count + " wheels.");
    }
    public void bodyColor(){
        System.out.println("Bike is " + color + " in color.");
    }
    public void  seats(){
        System.out.println("Bike has " + seats +" seat.");
    }
    public void setData(){
        System.out.println("Enter number of wheels, body colour and number of 
seats:");
        count=scanner.nextInt();
        color = scanner.next();
        seats = scanner.nextInt();
    }
}
class Car implements Vehicle{
    private String color;
    private int seats,count;
    Scanner scanner = new Scanner(System.in);
    //Implementing inherited abstract methods
    public void wheelsCount(){
        System.out.println("Car has " + count + " wheels.");
    }
    public void bodyColor(){
        System.out.println("Car is " + color + " in color.");
    }
    public void  seats(){
        System.out.println("Car has " + seats +" seat.");
    }
    public void setData(){
        System.out.println("Enter number of wheels, body colour and number of 
seats:");
        count=scanner.nextInt();
        color = scanner.next();
        seats = scanner.nextInt();
    }
}
//Driver class
public class Exp4_2 {
    public static void main(String[] args) {
        //Object of class bike
        Bike bike = new Bike();
        System.out.println("Enter bike details");
        bike.setData();
        bike.wheelsCount();
        bike.seats();
        bike.bodyColor();
        //object of class bicycle
        Bicycle bicycle = new Bicycle();
        System.out.println("Enter bicycle details");
        bicycle.setData();
        bicycle.wheelsCount();
        bicycle.seats();
        bicycle.bodyColor();
        //object of class car
        Car car = new Car();
        System.out.println("Enter car details");
        car.setData();
        car.wheelsCount();
        car.seats();
        car.bodyColor();
    }
}
```

Output:



```
D:\College\JAVA\Experiments\Exp4>javac Exp4_2.java

D:\College\JAVA\Experiments\Exp4>java Exp4_2
Enter bike details
Enter number of wheels, body colour and number of seats:
2 red 2
Bike has 2 wheels.
Bike has 2 seat.
Bike is red in color.
Enter bicycle details
Enter number of wheels, body colour and number of seats:
2 blue 1
Bicycle has 2 wheels.
Bicycle has 1 seat.
Bicycle is blue in color.
Enter car details
Enter number of wheels, body colour and number of seats:
4 white 4
Car has 4 wheels.
Car has 4 seat.
Car is white in color.

D:\College\JAVA\Experiments\Exp4>
```

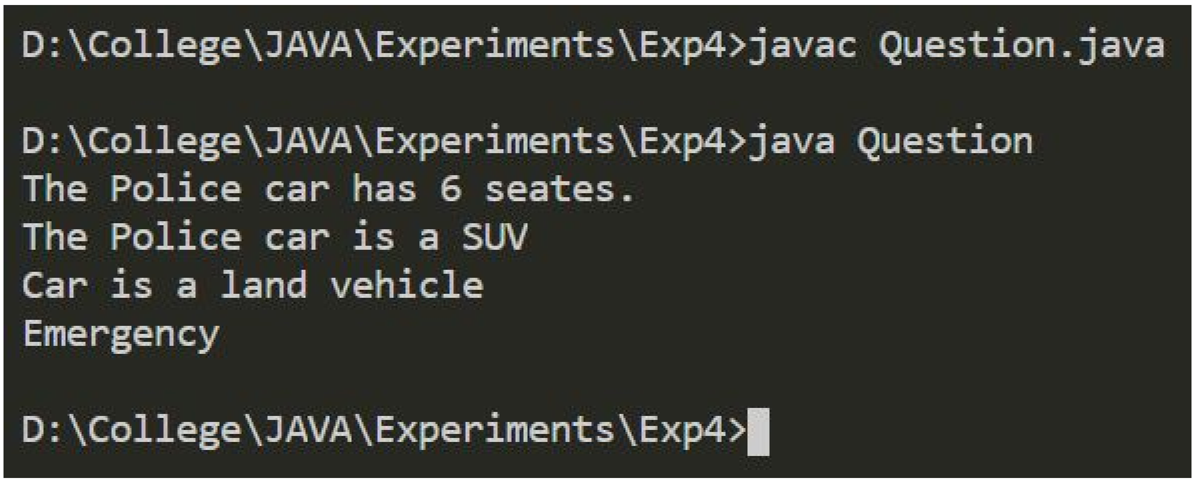
Questions:

Question 1:

```
/**
 * Write a class PoliceCar that implements the IsEmergency and IsLandVehicle
interfaces. In addition to the methods you have written for the PoliceCar class,
think of a new method or property that police cars have and add it to the class.
 */

 //defining Interface isEmergency
interface IsEmergency{
    void Emergency();
}
//defining Interface isLandVehicle
interface IsLandVehicle{
    void LandVehicle();
}

//class policeCar inherits interfaces isEmergency and isLandVehicle
class PoliceCar implements IsEmergency, IsLandVehicle{

    private int no_seates;
    private String type;

    public int getNo_seates() {
        return no_seates;
    }

    public void setNo_seates(int no_seates) {
        this.no_seates = no_seates;
    }

    public String getType() {
        return type;
    }

    public void setType(String type) {
        this.type = type;
    }
```

```java
    //implementing inherited abstract methods
    public void Emergency(){
        System.out.println("Emergency");
    }
    public void LandVehicle(){
        System.out.println("Car is a land vehicle");
    }

}

//Driver class
public class Question {
    public static void main(String[] args) {

        //Object of class PoliceCar
        PoliceCar pCar = new PoliceCar();
        pCar.setNo_seates(6);
        pCar.setType("SUV");
        System.out.println("The Police car has " + pCar.getNo_seates() + "
seates.");
        System.out.println("The Police car is a " + pCar.getType());
        pCar.LandVehicle();
        pCar.Emergency();
    }
}
```

Output:



```
D:\College\JAVA\Experiments\Exp4>javac Question.java

D:\College\JAVA\Experiments\Exp4>java Question
The Police car has 6 seates.
The Police car is a SUV
Car is a land vehicle
Emergency

D:\College\JAVA\Experiments\Exp4>
```

Page No :
Date :

Jash Mahajan SE IT B 04

A) Extended Theory

1) Explain how to implement multiple inheritance in Java with syntax and example.
Multiple Inheritance is a feature of object oriented concept where a class can inherit properties of more than one parent class.
Java does not support multiple inheritance, however a class can implement one or more interface which has helped Java get rid of ~~interface~~ impossiblety of multiple inheritance.

Eg)
```
class A { }
class B { }
class C extends A, B { }
```
This inheritance using multiple class is wrong as two class cannot be inherited.

To achieve multiple inheritance, the correct way is extending only one class or implementing two or more interfaces

Eg:-
```
interface A { }
interface B { }
class C implements ~~interf~~ A, B { }
```

Attitude

Yash Mahajan SE IT B 04.

2) Differentiate between abstract class and interface.

Abstract Class	Interface.
1) Abstract class can have abstract & non-abstract methods	1) Interface can have only abstract methods.
2) Doesn't support multiple inheritance	2) supports multiple inheritance
3) Abstract class can have final, non-final, static, non static variable	3) Interface can only have static and final variable.
4) The abstract keyword is used to declare an abstract class.	4) The interface keyword is used to declare an interface.
5) A Java abstract class can have class members of all visibility access.	5) Members of Java interface are public by default.

Page No :

Date :

Yash Mahajan SE IT B 04

D) Conclusion :-

In this experiment we have studied the concept about interfaces. To understand how interfaces are implemented we have written programs in which classes implement interfaces.

An interface in Java programming language is an abstract type that is used to specify a behaviour that classes must implement. One important advantage of using interfaces is that we can implement multiple inheritance.

Interfaces help us achieve total abstraction. ~~I~~ It also helps us to achieve loose coupling. They also enable us to implement multiple inheritance.

Attitude