Experiment Number : 05 **Date: 05/02/2026**

INHERITANCE

Aim:

To understand and implement inheritance concepts in Java.

PRE LAB EXERCISE

QUESTIONS

- ✓ What is inheritance?

  Inheritance is an object-oriented programming concept where one class acquires the properties and behaviors of another class. The class that inherits is called the subclass, and the class being inherited from is called the superclass. Inheritance helps in organizing code efficiently. It also supports the concept of "is-a" relationship. This improves maintainability of programs.

- ✓ What is code reusability?

  Code reusability means using existing code in multiple parts of a program or in different programs. It avoids rewriting the same logic again and again. Reusability reduces development time and errors. It also makes programs easier to maintain and update. Concepts like inheritance and functions support code reusability.

- ✓ What is the use of extends keyword?

  The extends keyword is used to inherit one class from another in Java. It allows a subclass to access the variables and methods of the superclass. This promotes code reusability and hierarchy. Using extends, method overriding can also be achieved. It supports single inheritance in Java.

IN LAB EXERCISE

Objective:

To implement all types of inheritance.

PROGRAMS:

Student Result System (Single Inheritance)

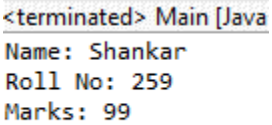
Question:
A school wants to store student details and calculate marks. Create a base class Student and a derived class Result.

Code:

```
package programming.java.com;

class Student {
   String name;
   int rollNo;

   void getDetails() {
      name = "Shankar";
      rollNo = 259;
   }
}

Class Result extends Student {
   int marks = 99;

   void display() {
      System.out.println("Name: " + name);
      System.out.println("Roll No: " + rollNo);
      System.out.println("Marks: " + marks);
   }
}

public class Main {
   public static void main(String[] args) {
      Result = new Result();
      r.getDetails();
      r.display();
   }
}
```

Output:

```
Name: Shankar
Roll No: 259
Marks: 99
```



2. Bank Account System (Hierarchical Inheritance)

Question:
A bank has Savings and Current accounts. Both inherit from a common Account class.

Code:

```
class Account {
    void showAccountType() {
        System.out.println("Bank Account");
    }
}

class SavingsAccount extends Account {
    void interest() {
        System.out.println("Savings Account gives interest");
    }
}

class CurrentAccount extends Account {
    void overdraft() {
        System.out.println("Current Account supports overdraft");
    }
}

public class Main {
    public static void main(String[] args) {
        SavingsAccount s = new SavingsAccount();
        CurrentAccount c = new CurrentAccount();

        s.showAccountType();
        s.interest();
```

```
        c.showAccountType();
        c.overdraft();
    }
}
```

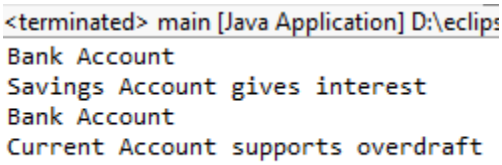
Output:

Bank Account

Savings Account gives interest

Bank Account

Current Account supports overdraft



3. Vehicle System (Multilevel Inheritance)

Question:
A company classifies vehicles as Vehicle → Car → ElectricCar.

Code:

```
class Vehicle {
    void start() {
        System.out.println("Vehicle starts");
    }
}

class Car extends Vehicle {
    void fuelType() {
        System.out.println("Car uses petrol");
    }
```

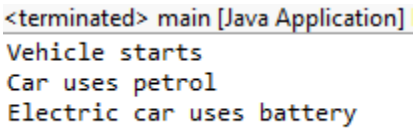
```
}
class ElectricCar extends Car {
    void battery() {
        System.out.println("Electric car uses battery");
    }
}
public class Main {
    public static void main(String[] args) {
        ElectricCar e = new ElectricCar();
        e.start();
        e.fuelType();
        e.battery();
    }
}
```

Output:

Vehicle starts

Car uses petrol

Electric car uses battery



POST LAB EXERCISE

- ✓ Why Java does not support multiple inheritance using classes and how it is implemented?

  Java does not support multiple inheritance using classes to avoid ambiguity problems like the *diamond problem*. If two parent classes have the same method, the compiler gets confused about which one to use. This can lead to complexity and errors. Instead, Java implements multiple inheritance using interfaces. Interfaces allow multiple inheritance without ambiguity since methods are either abstract or explicitly overridden.

- ✓ What is the role of the super keyword? Give examples.

  The super keyword is used to refer to the immediate parent class object. It is commonly used to access parent class variables, methods, and constructors. This helps avoid confusion when child and parent have the same member names.

  Example: super.display(); calls the parent class method, and super(a); calls the parent constructor.

- ✓ Can a child class access private members of the parent class? Why?

  No, a child class cannot directly access private members of the parent class. Private members are restricted to the class in which they are declared. This ensures data hiding and encapsulation. However, the child class can access them indirectly using public or protected methods. This maintains security and controlled access.

- ✓ Explain why hybrid inheritance is not supported in Java.

  Hybrid inheritance combines multiple types of inheritance, including multiple inheritance. Java does not support it with classes due to ambiguity and complexity issues. It can cause method call conflicts and confusion in class hierarchy. This affects code clarity and reliability. However, hybrid inheritance can be achieved in Java using interfaces.

Result:

Thus the different types of inheritance were implemented and executed successfully.

ASSESSMENT

Description	Max Marks	Marks Awarded
Pre Lab Exercise	**5**	
In Lab Exercise	**10**	
Post Lab Exercise	**5**	
Viva	**10**	
Total	**30**	
	Faculty Signature	