Experiment Number : 05 **Date: 04-02-2**

P.Theeran

24BCS298

CSE-A1

INHERITANCE

Aim:

To understand and implement inheritance concepts in Java.

PRE LAB EXERCISE

QUESTIONS

1. **What is inheritance?**

   Inheritance is a feature of OOP where one class (child/subclass) **acquires the properties and methods** of another class (parent/superclass).

2. **What is code reusability?**

   Code reusability means **using existing code again** without rewriting it, which saves time and reduces errors.

3. **What is the use of extends keyword?**

   The extends keyword is used to **create a child class from a parent class**, so the child class can use the parent class's methods and variables.

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:

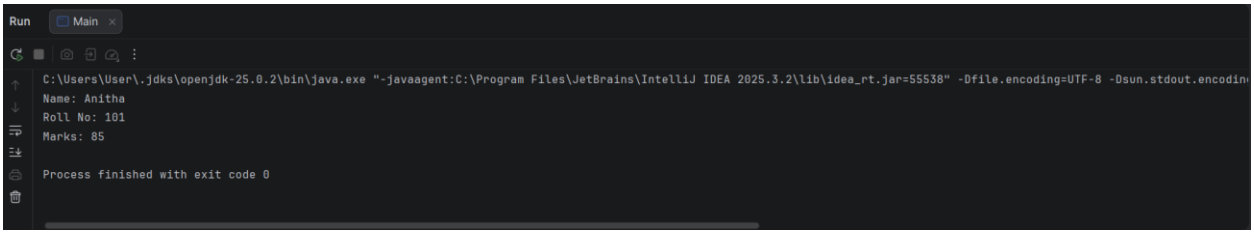
```
class Student {
    String name;
    int rollNo;

    void getDetails() {
        name = "Anitha";
        rollNo = 101;
    }
}

class Result extends Student {
    int marks = 85;

    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 r = new Result();
        r.getDetails();
        r.display();
    }
}
```

Output:

Name: Anitha

Roll No: 101

Marks: 85



2. Bank Account System (Hierarchical Inheritance)

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

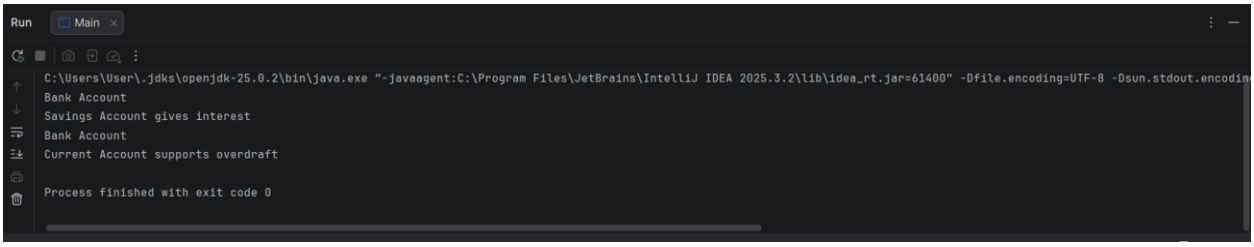
Code:

```
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)

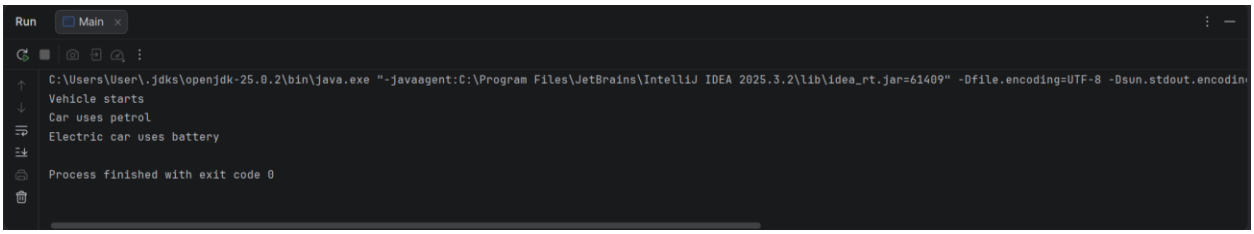
Question:
A company classifies vehicles as Vehicle → Car → ElectricCar.

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:

1. **Vehicle starts**
2. **Car uses petrol**
3. **Electric car uses battery**



POST LAB EXERCISE

4. **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 (Diamond Problem)**.

   It is implemented using **interfaces**, where a class can implement multiple interfaces without confusion.

5. **What is the role of the super keyword? Give examples.**

   The super keyword is used to **refer to the parent class object**.

   It is used to access **parent class variables, methods, and constructors**.

```java
class A {
    int x = 10;
}
class B extends A {
    void show() {
        System.out.println(super.x);
    }
}
```

3.Can a child class access private members of the parent class? Why?

No.

Private members are **accessible only within the same class**, not in subclasses, to maintain **data security and encapsulation**.

4.Explain why hybrid inheritance is not supported in Java.

Hybrid inheritance involves **multiple inheritance**, which leads to **ambiguity**.

Since Java does not support multiple inheritance using classes, **hybrid inheritance is also not supported**.

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	