Experiment Number : 05 **Date:**

INHERITANCE

Aim:

To understand and implement inheritance concepts in Java.

PRE LAB EXERCISE

QUESTIONS

1. **What is inheritance?**
   - Inheritance is an OOP feature where a child class gets properties of a parent class.
   - It helps in creating new classes from existing ones.
   - It improves code reuse and program structure.
2. **What is code reusability?**
   - Code reusability means using the same code in multiple places.
   - It avoids rewriting the same logic again.
   - It makes programs easier to maintain and update.
3. **What is the use of extends keyword?**
   - extends is used to inherit one class into another in Java.
   - It allows access to parent class methods and variables.
   - It supports inheritance and code reusability.

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();
    }
}
```

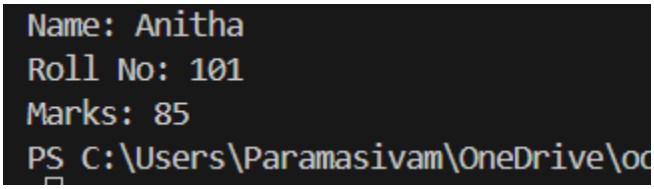
Output:

Name: RAM

Roll No: 101

Marks: 85

OUTPUT:



2. Bank Account System (Hierarchical Inheritance)

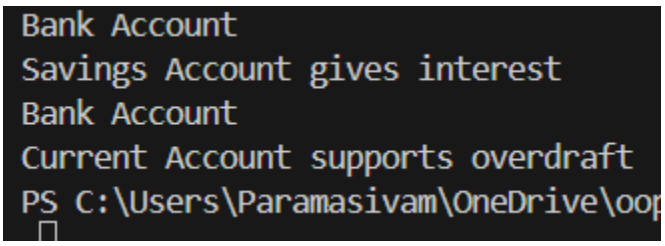
Question:
A bank has Savings and Current accounts. Both inherit from a common Account class.

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:



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

OUTPUT:

```
Vehicle starts
Car uses petrol
Electric car uses battery
PS C:\Users\Paramasivam\OneDrive\c
```

POST LAB EXERCISE

1. **Why Java does not support multiple inheritance using classes and how it is implemented?**

   • Java does not support multiple inheritance with classes to avoid ambiguity

   • If two parent classes have the same method, the compiler gets confused about which one to inherit.

   • Java solves this using interfaces, where multiple inheritance is allowed with clear implementation rules.

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

   • `super` is used to refer to the parent class object.
   • It is used to access parent class variables and methods.
   • It is also used to call the parent class constructor.
   Example:
   `super();` → calls parent constructor
   `super.display();` → calls parent method

3. **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 same class only.

   • This supports data hiding and encapsulation in Java.

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

   Hybrid inheritance involves a combination of multiple inheritance types.

   • It indirectly leads to multiple inheritance ambiguity.

   • Since Java does not support multiple inheritance with 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		