JAVA Lab
Lab Experiment number 04

Name: Aamir Ansari **Batch:** A **Roll no.** 01

Aim: Write a JAVA program to implement the concept of inheritance

Theory:

1. Inheritance

Inheritance is an important pillar of OOP (Object Oriented Programming). It is the mechanism in java by which one class is allow to inherit the features(fields and methods) of another class.
The keyword used for inheritance is extends.

example:

```
class derived-class extends base-class  {
        //methods and fields
}
```

2. Types of Inheritance

2.1.Single Inheritance

In single inheritance, subclasses inherit the features of one superclass.

2.2.Multilevel Inheritance

In Multilevel Inheritance, a derived class will be inheriting a base class and as well as the derived class also act as the base class to other class

2.3.Hierarchical Inheritance

In Hierarchical Inheritance, one class serves as a superclass (base class) for more than one sub class

3. Reason for multiple inheritance being unused in JAVA

Multiple inheritance is not used in JAVA, the reason is to prevent ambiguity
Consider a case where class B extends class A and Class C and both class A and C have the same method display().
Now JAVA compiler cannot decide, which display method it should inherit. To prevent such situation, multiple inheritances is not allowed in JAVA.
The two parent classes may define different ways of doing the same thing, and the subclass can't choose which one to pick.

4. Usage of 'super'

The super keyword in Java is a reference variable which is used to refer immediate parent class object.

Whenever you create the instance of subclass, an instance of parent class is created implicitly which is referred by super reference variable.
super keyword can be used to call constructor of parent class, by using
super(//parameters) in the sub class

super keyword can also call variables or methods from parent class in the following way
super.parentMethod();
super.parentVariable;

5. Method Overriding

In any object-oriented programming language, Overriding is a feature that allows a subclass or child class to provide a specific implementation of a method that is already provided by one of its super-classes or parent classes. When a method in a subclass has the same name, same parameters or signature, and same return type(or sub-type) as a method in its super-class, then the method in the subclass is said to override the method in the super-class.
Methods are overridden by using the same name, but with different parameter or by using it in different classes.
When a method is called in sub class, it automatically overrides the method of parent class.

Program:

```
//code
import java.util.*;

class Student {
    // instance variable => general information of all students
    String name;
    int age;
    String programme;

    // constructor
    Student(String name, int age, String programme) {
        this.name = name;
        this.age = age;
        this.programme = programme;
    }

    // method to display
    void display_student_info() {
        System.out.println("Name of Student : "+name);
        System.out.println("Age of Student : "+age);
        System.out.println("Programme of Student : "+programme);
    }
}

class ReasearchStudent extends Student {
    // instance variable => information of Reasearch Student
    String specialization;
    double workingExperience;

    // constructor
    ReasearchStudent(String name, int age, String programme, String specialization, double
workingExperience) {
        super(name, age, programme);    // calls super constructor
        this.specialization = specialization;
```

```
        this.workingExperience = workingExperience;
    }

    // getters and setters of ReasearchStudent
    String getSpecialization() {
        return this.specialization;
    } void setSpecialization(String specialization) {
        this.specialization = specialization;
    }

    double getWorkingExperience() {
        return this.workingExperience;
    } void setWorkingExperience(double workingExperience) {
        this.workingExperience = workingExperience;
    }

    // method to display
    void display_student_info() {
        System.out.println("*   Name  of   Student   :       "+name);
        System.out.println("*   Age   of   Student   :       "+age);
        System.out.println("*   Programme of Student :       "+programme);
        System.out.println("*   specialization  of student :  "+specialization);
        System.out.println("*   Work experience of student :  "+workingExperience);
    }
}

class GradStudent extends Student {
    // instance variable => information of Grad Student
    double percentage;
    String stream;

    // constructor
    GradStudent(String name, int age, String programme, double percentage, String stream) {
        super(name, age, programme);    // calls super constructor
        this.percentage = percentage;
```

```
        this.stream = stream;
    }

    // getters and setters of Grad Student
    double getPercentage() {
        return this.percentage;
    } void setPercentage(double percentage) {
        this.percentage = percentage;
    }

    String getStream() {
        return this.stream;
    } void setStream(String stream) {
        this.stream = stream;
    }

    // method to display
    void display_student_info() {
        System.out.println("*  Name of Student  : "+name);
        System.out.println("*  Age of Student  : "+age);
        System.out.println("*  Programme of Student :  "+programme);
        System.out.println("*  Percentage  of student :  "+percentage);
        System.out.println("*  Stream of student  : "+stream);
    }
}

class Inheritance {

    public static void main(String args[]) {

        Scanner sc = new Scanner(System.in);
        int choice;
        int age;
        String name, programme, specialization, stream;
        double workingExperience, percentage;
```

```
        while (true) {
            System.out.println("*1 Reasearch Student");
            System.out.println("*2 Graduate  Student");
            System.out.println("*3 EXIT");
            System.out.print("Enter the Type of student  :  ");
            choice = sc.nextInt();
            sc.nextLine();

            switch (choice) {
                case 1:    // Reasearch Student
                    System.out.print("Enter Name  of   Student   :  ");
                    name = sc.nextLine();

                    System.out.print("Enter Age   of   Student   :  ");
                    age = sc.nextInt();sc.nextLine();

                    System.out.print("Enter Programme of Student :  ");
                    programme = sc.nextLine();

                    System.out.print("Enter specialization  of student :  ");
                    specialization = sc.nextLine();

                    System.out.print("Enter Work experience of student :  ");
                    workingExperience = sc.nextDouble();

                    ReasearchStudent rStudent = new ReasearchStudent(name, age, programme,
specialization, workingExperience);
                    System.out.println();
                    System.out.println("**********************************************");
                    rStudent.display_student_info();
                    System.out.println("**********************************************");
                    break;

                case 2:     // Grag Student
```

```
            System.out.print("Enter Name  of   Student   :  ");
            name = sc.nextLine();

            System.out.print("Enter Age   of   Student   :  ");
            age = sc.nextInt();sc.nextLine();

            System.out.print("Enter Programme of Student  :  ");
            programme = sc.nextLine();

            System.out.print("Enter Percentage of student :  ");
            percentage = sc.nextDouble();sc.nextLine();

            System.out.print("Enter Stream  of  student   :  ");
            stream = sc.nextLine();

            GradStudent gStudent = new GradStudent(name, age, programme, percentage, stream);
            System.out.println();
            System.out.println("************************************************");
            gStudent.display_student_info();
            System.out.println("************************************************");
            System.out.println();
            break;

        case 3:
            System.out.println();
            System.out.println("*** E X I T I N G ***");
            System.exit(1);

        default:
            System.out.println("INVALID INPUT");

      }
    }
  }
}
```

Output

```
E:\Aamir\Sem-3\LabWork - Assignments\OOPM\Lab Assignment 4>javac Inheritance.java

E:\Aamir\Sem-3\LabWork - Assignments\OOPM\Lab Assignment 4>java Inheritance
*1 Reasearch Student
*2 Graduate  Student
*3 EXIT
Enter the Type of student  :  1
Enter Name  of   Student   :  Aamir Ansari
Enter Age   of   Student   :  18
Enter Programme of Student :  Full time
Enter specialization  of student :  Psychology
Enter Work experience of student :  12.5

*********************************************
*   Name  of   Student   :        Aamir Ansari
*   Age   of   Student   :        18
*   Programme of Student :        Full time
*   specialization  of student :  Psychology
*   Work experience of student :  12.5
*********************************************


*1 Reasearch Student
*2 Graduate  Student
*3 EXIT
Enter the Type of student  :  2
Enter Name  of   Student    :  Krishna Subramanian
Enter Age   of   Student    :  18
Enter Programme of Student  :  Part Time
Enter Percentage of student :  95.21
Enter Stream  of  student   :  Science

***********************************************
*   Name  of   Student   :  Krishna Subramanian
*   Age   of   Student   :  18
*   Programme of Student :  Part Time
*   Percentage  of student :  95.21
*   Stream  of  student    :  Science
***********************************************

*1 Reasearch Student
*2 Graduate  Student
*3 EXIT
Enter the Type of student  :  3

*** E X I T I N G ***
```