Lab Number:	06
Student Name:	SHIVAM SHARMA
Roll No :	E-19

Title:

1. To perform Multilevel Inheritance in JAVA. Create a
   Person class representing name, age and address. Inherit person class to employee
   class with emp ID and salary factor. Inherit the Employee class to programmer
   class with technical skills and hike attributes. Implement valid methods to input the
   details from the user in the main method and display for 3 programmers.

2. To perform Hierarchical Inheritance in JAVA. Create an Employee class with
   attributes EmpID and EmpSalary. Also create necessary methods/constructors to
   accept these values from the user. Create classes permenantEmployee and
   TemporaryEmployee which will be derived classes of Employee. Mention hike
   attribute in these derived classes and calculate the total salary using
   generate_salary() method for respective types of employees. Objects of the derived
   classes should be created and salaries for the permanent and temporary employees
   should be calculated and displayed on the screen.

Learning Objective:

● Students will be able to write C++ and java program for various types of
  inheritance .

Learning Outcome:

● Ability to execute a simple C++ and Java program with and without any
  inputs tothe program.
● Understanding inheritance in C++ and Java.

Theory:

What is Inheritance?

Faculty: Ms. Deepali Kayande

Inheritance in C++ is a vital concept and can not be overlooked. Understanding inheritance is critical for understanding the whole point behind object-oriented programming. For instance, you are a human. You inherit from the class 'Humans' characteristic features, such as walking, sitting, running, eating, and so on. The class 'Humans' inherits these characteristic features from the class 'Mammal' which makes the 'Human' class a derived class of 'Mammal'. This 'Mammal' class inherits its characteristic features from another class 'Animal' which makes the 'Mammal' class a derived class of the class 'Animal' and makes the 'Animal' a base class.

One of the most astonishing features of inheritance is code reusability. This reusability also provides you with clean code, and the replication of code gets reduced to almost zero.

Reusing existing codes serves various advantages. It saves time, money, effort, and increases a program's reliability.

Types Of Inheritance:

C++ supports five types of inheritance:

1. Single inheritance

2. Multiple inheritance

3. Hierarchical inheritance

4. Multilevel inheritance

5. Hybrid inheritance

Uses of Inheritance:

Inheritance is a useful concept of object-oriented programming. Inheritance in C++ serves many advantages. There are several reasons why inheritance was introduced in OOPs. You will see some of the major reasons behind the introduction of inheritance in C++, below:

- Inheritance increases the relatability of the code to real-world scenarios drastically.

- Another reason is the idea of reusability. Code reusability ensures that a clean code is provided to the programmer. This also helps in the reduction of rewriting and serves as a bug-free code, as the replication of the code gets reduced to almost zero with the help of reusability. Other advantages of reusability are time management, maintenance, and ease

Faculty: Ms. Deepali Kayande

of extension. You can do manipulations and add some desired features to a class that already exists through inheritance.

- One more reason is the transitive nature of inheritance. Transitive nature implies that if two objects that are in succession show a pattern, then all the objects of that order must show the exact pattern. For example, if a new class TataSafari is declared as a subclass of Car, which itself is a subclass of Vehicle, then TataSafari must also be a Vehicle i.e., inheritance is transitive in nature.

Multilevel Inheritance:
In C++ programming, not only you can derive a class from the base class but you can also derive a class from the derived class. This form of inheritance is known as multilevel inheritance.

Hierarchical Inheritance:
If more than one class in inherited from the base class, it's known as hierarchial inheritance. In hierarchical inheritance, all features that are common in child classes are included in the base class.

Algorithm 1:
STEP 1: start
STEP 2: define two n1 and n2STEP 3: input number1,n1 input number2, n2
STEP 4: addition of two numbers n1 and n2 STEP 5: swapping of two numbers n1 and n2
STEP 6: odd or even
STEP7: stop

Program 1:
Input:

```java
import java.util.*;

class Person {

String name;

int age;

String
address;
```

```java
public Person() {
name = "";
age = 0;
address = "";
}
void getdata() {
Scanner s = new
Scanner(System.in);
System.out.print("enter name:
");
name = s.nextLine();
System.out.println()
;
System.out.print("enter age:
");
age = s.nextInt();
System.out.println()
;
System.out.print("enter address:
");
s.nextLine();
address =
s.nextLine();
System.out.println()
;
}
void putdata() {
System.out.println("name is: " +
name);
System.out.println()
;
System.out.println("age is::" +
age);
System.out.println()
;
System.out.println("address is: " +
address);
```

```java
System.out.println()
;
}
}
class employee extends Person
{
int emp_id;
int salary_factor;

public employee() {
emp_id = 0;
salary_factor = 0;

}
void getdetails() {
Scanner s = new
Scanner(System.in);
System.out.print("enter employee id:
");
emp_id = s.nextInt();
System.out.println()
;
System.out.print("enter Salary Factor:
");
salary_factor =
s.nextInt();
System.out.println()
;

}
void putdetails() {
System.out.println("employee id is ::" +
emp_id);
System.out.println()
;
System.out.println("Salary Factor is::" +
salary_factor);
System.out.println()
;
```

```java
}
}
class programmer extends employee
{
int hike;
String technical_skills = "";

public programmer()
{
hike = 0;
technical_skills = "";

}
void getd() {
Scanner s = new
Scanner(System.in);
System.out.print("enter hike:
");
hike = s.nextInt();
System.out.println()
;
System.out.print("enter technical skills:
");
s.nextLine();
technical_skills = s.next();
System.out.println()
;
}
void putd() {
System.out.println("hike is ::" +
hike);
System.out.println()
;
System.out.println("techincal skills is::" +
technical_skills);
System.out.println()
;
}
}
```

```java
public class Main {

public static void main(String[] args)
{
programmer r[] = new
programmer[4];
r[0] = new
programmer();
r[1] = new
programmer();
r[2] = new
programmer();

for (int i = 0; i<3; i++) {
System.out.println("Enter details of employee " +
(i+1) );

r[i].getdata()
;
r[i].getdetails()
;
r[i].getd();
r[i].putdata()
;
r[i].putdetails()
;
r[i].putd()
;
}
}
}
```

Output:

```
Enter details of employee 1
enter name: shivam

enter age: 20

enter address: kalyan

enter employee id: 19

enter Salary Factor: 25000

enter hike: 500

enter technical skills: java

name is: shivam

age is::20

address is: kalyan

employee id is ::19

Salary Factor is::25000

hike is ::500

techincal skills is::java
```

PROGRAM 2 :

Algorithm 2

Step 1 : start
Step 2 :Creating the parent class employee and initialize its data members.(EmpId
,EmpSalary) and a basic function get details() to print the details.
Step 3 : Create 2 child class permanent employee and temporary employee that inherit
employee class publically.
Step 4 : In this classes , create generate salary() that return the employee salary + hike in
their salary
 Step 5 :In main function, Create the object of derived class and print their respective details.
Step 6 : End

Input 2:

```java
import java.util.*;
    class Employee
    {
    Scanner s=new
    Scanner(System.in);
    int emp_id;
    int emp_salary;

    Employee()
    {
    System.out.println("enter
    empid::");
    emp_id=s.nextInt()
    ;
    System.out.println("enter
    empsalary::");
    emp_salary=s.nextInt()
    ;
    }

    void getDetails()
    {
    System.out.println("EmployeeID
    is ::");
    System.out.println("EmployeeSalary
    is ::");
    }

    }

    class permanant_Employee extends
    Employee
    {
    int hike;

    permanant_Employee( int
    increment)
    {
```

```java
hike = increment;
}

void getDetails()
{
System.out.println("EmployeeID is ::" +
emp_id);
System.out.println("Employee total salary is ::" +
generate_salary());
}

int
generate_salary()
{
return (emp_salary +
hike);
}
}

class temporary_Employee extends
Employee
{
int hike;

temporary_Employee( int
increment)
{
hike = increment;
}

void getDetails()
{
System.out.println("EmployeeID is ::" +
emp_id);
System.out.println("Employee total salary is ::" +
generate_salary());
}

int
generate_salary()
{
```

```
return (emp_salary +
hike);
}
}

public class Main {

public static void main(String[] args)
{
permanant_Employee p = new
permanant_Employee(3200);
p.getDetails();

temporary_Employee t = new
temporary_Employee(1600);
t.getDetails();

}
}
```

Output:

```
enter empid::
19
enter empsalary::
12000
EmployeeID is ::19
Employee total salary is ::15200
enter empid::
10
enter empsalary::
14000
EmployeeID is ::10
Employee total salary is ::15600


...Program finished with exit code 0
Press ENTER to exit console.
```