OOP Lab Practical – 5

Name: Divyam Kumar

SAPID: 500083141

Roll No: R214220434

Batch: B2

**Title: Inheritance**

1. Write a Java program to show that private member of a super class cannot be accessed from derived classes.
2. Write a program in Java to create a Player class. Inherit the classes Cricket \_Player, Football \_Player and Hockey\_ Player from Player class.
3. Write a class Worker and derive classes DailyWorker and SalariedWorker from it. Every worker has a name and a salary rate. Write method ComPay (int hours) to compute the week pay of every worker. A Daily Worker is paid on the basis of the number of days he/she works. The Salaried Worker gets paid the wage for 40 hours a week no matter what the actual hours are. Test this program to calculate the pay of workers. You are expected to use the concept of polymorphism to write this program.
4. Design a class employee of an organization. An employee has a name, empid, and salary. Write the default constructor, a constructor with parameters (name, empid, and salary) and methods to return name and salary. Also write a method *increaseSalary* that raises the employee’s salary by a certain user specified percentage. Derive a subclass Manager from employee. Supply a test program that uses theses classes and methods.

Question 1:

ALGORITHM

Step 1: START

Step 2: Create a Super class

Step 3: initialize an integer variable as private

Step 4: Define method mul(int,int) as private.

Step 5: Print the product for the statement of the method mul()

Step 6 : Create the Base Class

Step 7: Creating the objects call mul () and the integer variable

Step 8: END

CODE:

class Inheritance

{

private int a =100;

private static void mul(int a,int b)

{

System.out.println("Product:"+(a\*b));

}

}

class Base extends Inheritance

{

public static void main(String[] args) {

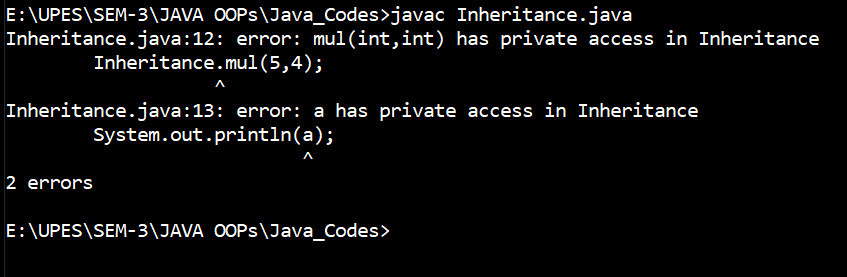
Inheritance.mul(5,4);

System.out.println(a);

}

}

OUTPUT:



Question 2:

ALGORITHM

Step 1: START

Step 2: Create a class Player

Step 3: Declare global variable String name, integer age, double weight

Step 4: Initialize name = n , age =a and weight = wt in constructor

Step 5: Create method void show()

Step 6: Print name, age and weight as the statement of the method

Step 7: Create base class Player\_Cricket

Step 8: create method void show() and invoke void show of Parent class Player using keyword super.

Step 9: Create base class Player\_Football

Step 10: create method void show() and invoke void show of Parent class Player using keyword super.

Step 11: Create base class Player\_Hockey

Step 12: create method void show() and invoke void show of Parent class Player using keyword super.

Step 13:Create class Driver

Step 14: Define main

Step 15: Create instance of Player\_Cricket , Player\_Football

and Player\_Hockey as cp , fp and hp respectively

Step 16: Display cp.show(), fp.show() and hp.show()

Step 17: END

CODE:

class Player

{

String name;

int age;

double weight;

Player(String n,int a, double wt)

{

name = n;

age = a;

weight = wt;

}

void show()

{

System.out.println("Player name: "+name);

System.out.println("Age: "+age);

System.out.println("Weight: "+weight);

}

}

class Player\_Cricket extends Player

{

String game;

Player\_Cricket(String n,String g,int a, double wt)

{

super(n,a,wt );

game=g;

}

public void show()

{

super.show();

System.out.println("Game : "+game);

}

}

class Player\_Football extends Player

{

String game;

Player\_Football(String n,String g,int a,double wt)

{

super(n,a,wt);

game=g;

}

public void show()

{

super.show();

System.out.println("Player game : "+game);

}

}

class Player\_Hockey extends Player

{

String game;

Player\_Hockey(String n,String g,int a,double wt)

{

super(n,a,wt);

game=g;

}

public void show()

{

super.show();

System.out.println("Player game : "+game);

}

}

class Driver

{

public static void main(String args[])

{

Player\_Cricket cp=new Player\_Cricket("Max","Cricket",19,60.2);

Player\_Football fp=new Player\_Football("Rohit","Football",20,55.8);

Player\_Hockey hp=new Player\_Hockey("Bharat","Hockey",15,69.5);

cp.show();

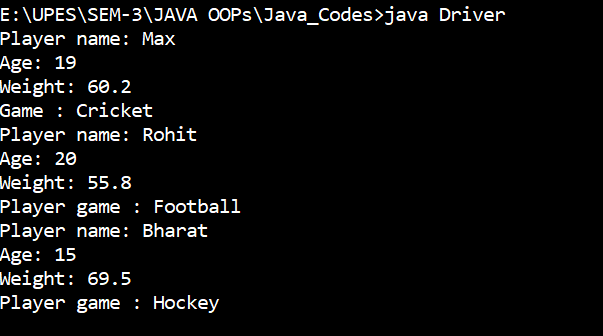
fp.show();

hp.show();

}

}

OUTPUT:



Question 3:

ALGORITHM

Step 1: START

Step 2: Create a class Worker

Step 3: Declare global variable String name, integer empno

Step 4: Initialize name = n , empno= no in constructor

Step 5: Create method void show()

Step 6: Print name, and empno as the statement of the method

Step 7: Create base Daily\_Worker

Step 8: Using Super keyword invoke variables from parent class

Step 9: Create method void show and invoke show from parent class using super

Step 10: Calculate and print the salary with the given formula

Step 11: Create base class SalariedWorker

Step 12:Using Super keyword invoke variables from parent class

Step 13: Create method void Compay and method invoke show from parent class using super

Step 14: Calculate and print the salary with the given formula

Step 15: Create Class Salary

Step 16: Define main function

Step 17: Create instance of Dailyworker and Salariedworker as dw and sw respectively

Step 18: Call dw.show() and sw.Compay(40)

Step 19: END

CODE:

class Worker

{

String name;

int empno;

Worker(String n,int no)

{

name =n;

empno = no;

}

void show()

{

System.out.println("Employee Name: "+name);

System.out.println("Employee ID: "+empno);

}

}

class DailyWorker extends Worker

{

DailyWorker(String n, int no)

{

super(n,no);

}

void show(){

super.show();

System.out.println("No. of Days worked: 50");

double sal=5.0\*50;

System.out.println("Salary with 5.0 rate: "+sal);

}

}

class SalariedWorker extends Worker

{

SalariedWorker(String n, int no)

{

super(n,no);

}

double rate = 6.0;

void Compay(int h)

{

super.show();

System.out.println("Salary for 40 hours of work with 6.0 rate:"+rate\*h);

}

}

class Salary

{

public static void main(String[] args)

{

DailyWorker dw = new DailyWorker("Rohan",102);

SalariedWorker sw = new SalariedWorker("Shayam",208);

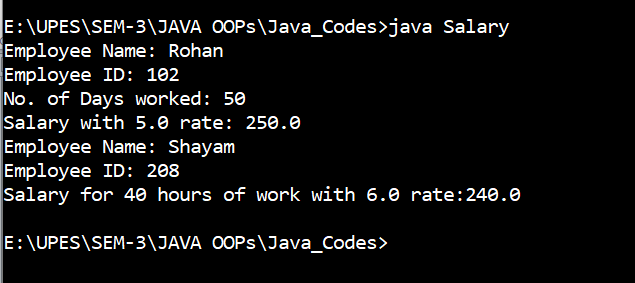
dw.show();

sw.Compay(40);

}

}

OUTPUT:

****

Question 4:

ALGORITHM

Step 1: START

Step 2: Create a class Employee

Step 3: Declare global variable String name, integer empno and double salary

Step 4: Create Constructor and print “Default constructor created”

Step 5: Create parameterized constructor

Step 6: Initialize name = n , empno= no and salary = slr in constructor

Step 7: Create method void display()

Step 8: Print Employee name and Salary in the function statement

Step 9: Create method Double increaseSalary(Double perc)

Step 10: Calculate the increase in the salary and return salary

Step 11: Derive Manager as a base class of Employee

Step 12: Create class Emp

Step 13:Define main function

Step 14: Create instances of class Employee and Manager emp and mn respectively

Step 15: Pass (101,”Mohit”,25000) to Employee constructor

Step 16: Initialize mn.name = ‘Jeff Bezoz’

Step 17: Initialize mn.salary = 40000.0

Step 18: Call mn.display()

Step 19: Call emp.increaseSalary(12.5)

Step 20: Call emp.display()

Step 21:END

CODE:

class Employee

{

int empno;

Double salary;

String name;

Employee()

{

System.out.println("Default constructor created");

}

Employee(int no,String n,Double slr)

{

empno=no;

name=n;

salary=slr;

}

void display()

{

System.out.println("Employee name is :"+name);

System.out.println("Salary= "+salary);

}

Double increaseSalary(Double perc)

{

salary=salary+((perc/100)\*salary);

return salary;

}

}

class Manager extends Employee

{

}

class Emp

{

public static void main(String args[])

{

Employee emp=new Employee(101,"Mohit",25000.0);

Manager mn=new Manager();

mn.name="Jeff Bezoz";

mn.salary=40000.00;

mn.display();

emp.increaseSalary(12.5);

emp.display();

}

}

OUTPUT:

