Name : Nikam Khushi Sahebrao.
PRN : 22620004
Branch : Information Technology
Batch : S1
Subject : Java Programming

ASSIGNMENT 02

- Answer the following.

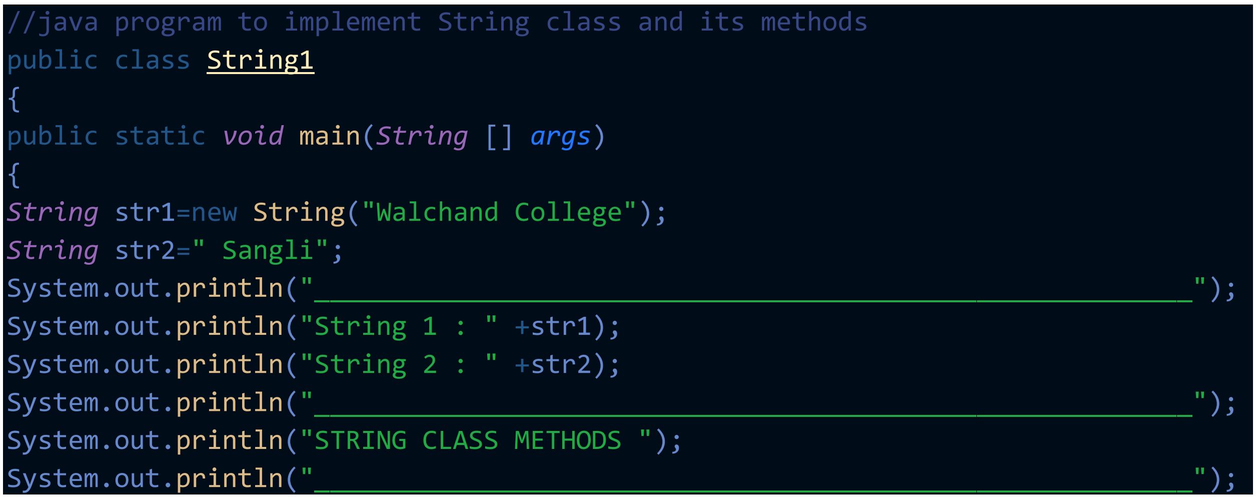
1. Explain difference between method overloading and method overriding.

Ans :

1. Method Overloading:
   1) Method overloading is a feature that allows a class to have two or more methods with the same name but different parameters in the same class.
   2) In Java, method overloading is achieved by defining two or more methods with the same name but different parameters.
   3) The compiler decides which method to call based on the number and types of arguments passed to the method during the method call.
2. Method Overriding:
   1) Method overriding is a feature that allows a subclass to provide its own implementation of a method that is already defined in its superclass.
   2) In Java, method overriding is achieved by defining a method in the subclass with the same name, return type, and parameters as a method in the superclass.

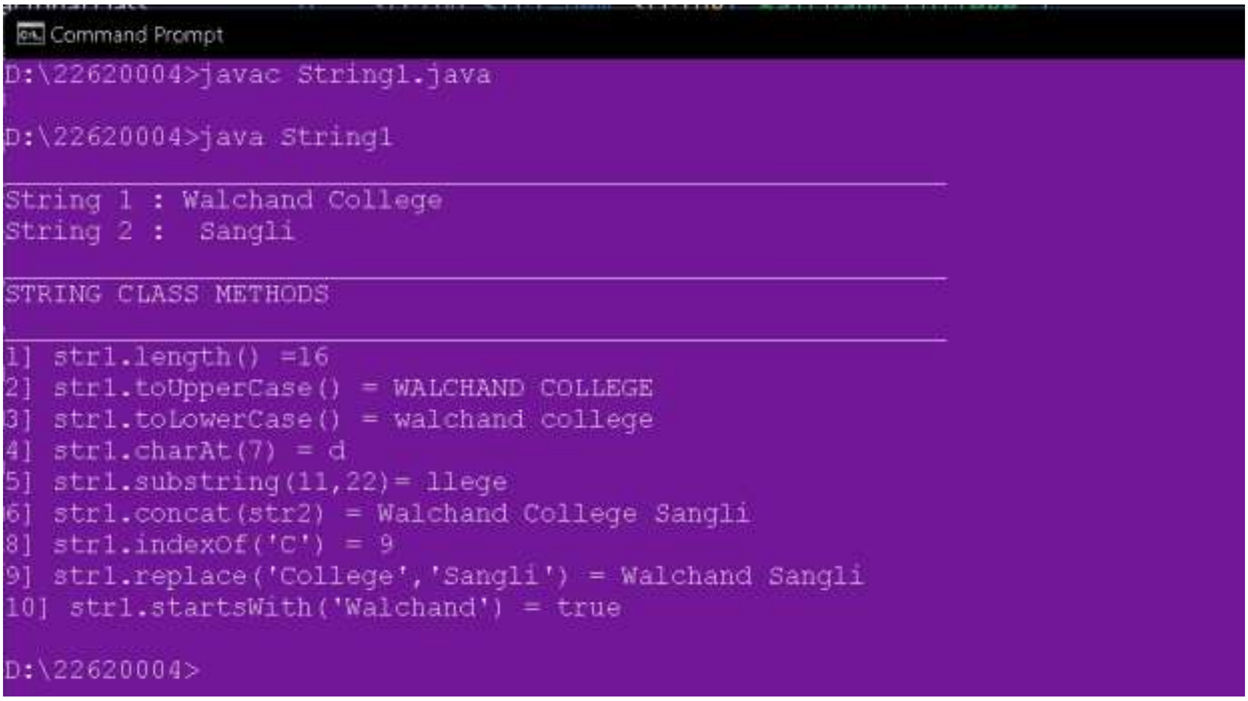
2. Implement all string functions in java.

Program :



```
//java program to implement String class and its methods
public class String1
{
public static void main(String [] args)
{
String str1=new String("Walchand College");
String str2=" Sangli";
System.out.println("_________________________________________________________");
System.out.println("String 1 : " +str1);
System.out.println("String 2 : " +str2);
System.out.println("_________________________________________________________");
System.out.println("STRING CLASS METHODS ");
System.out.println("_________________________________________________________");
System.out.println("1] str1.length() ="+str1.length());
System.out.println("2] str1.toUpperCase() = "+str1.toUpperCase());
System.out.println("3] str1.toLowerCase() = "+str1.toLowerCase());
System.out.println("4] str1.charAt(7) = "+str1.charAt(7));
System.out.println("5] str1.substring(11,22)= "+str1.substring(11,16));
System.out.println("6] str1.concat(str2) = "+str1.concat(str2));
System.out.println("8] str1.indexOf('C') = "+str1.indexOf('C'));
System.out.println("9] str1.replace('College','Sangli') =
"+str1.replace("College","Sangli"));
System.out.println("10] str1.startsWith('Walchand') =
"+str1.startsWith("Walchand"));
}
}
```

Output :



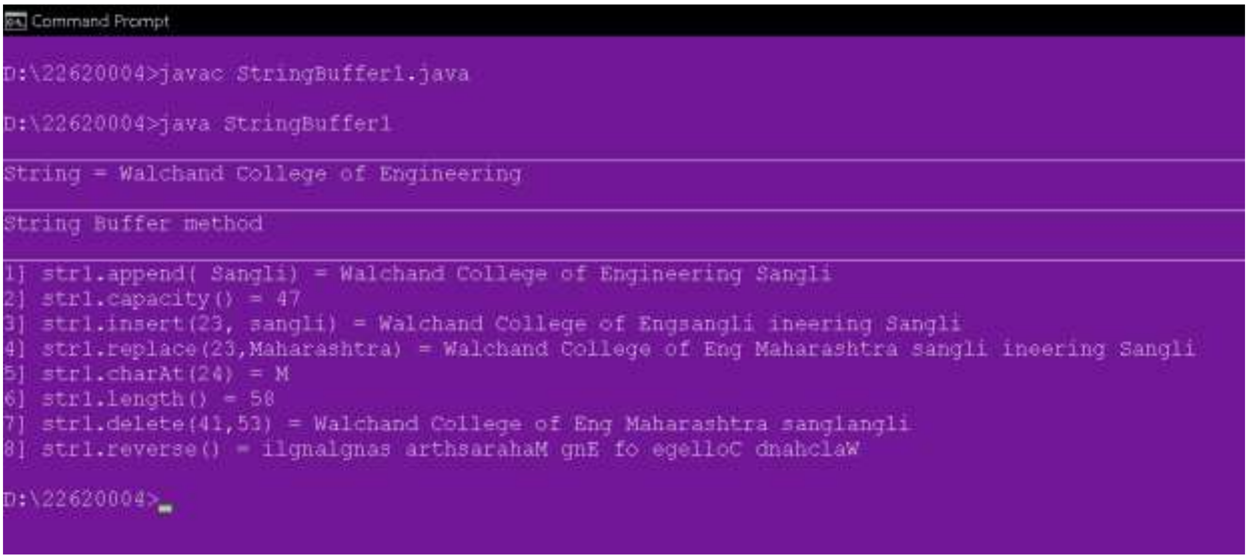
```
D:\22620004>javac String1.java

D:\22620004>java String1
________________________________________________________________
String 1 : Walchand College
String 2 :  Sangli
________________________________________________________________
STRING CLASS METHODS
________________________________________________________________
1] str1.length() =16
2] str1.toUpperCase() = WALCHAND COLLEGE
3] str1.toLowerCase() = walchand college
4] str1.charAt(7) = d
5] str1.substring(11,22)= llege
6] str1.concat(str2) = Walchand College Sangli
8] str1.indexOf('C') = 9
9] str1.replace('College','Sangli') = Walchand Sangli
10] str1.startsWith('Walchand') = true

D:\22620004>
```

3. Implement all stringbuffer functions in java.

Program :

```
//String Buffer class and its method
import java.io.*;
public class StringBuffer1 {
public static void main(String[] args)
{
StringBuffer str1 = new StringBuffer("Walchand College of Engineering");
System.out.println("____________________________________________________________________________________________________________________________________");
System.out.println("String = "+str1);
System.out.println("____________________________________________________________________________________________________________________________________");
System.out.println("String Buffer method ");
System.out.println("____________________________________________________________________________________________________________________________________");
System.out.println("1] str1.append( Sangli) = "+ str1.append(" Sangli"));
System.out.println("2] str1.capacity() = " +str1.capacity());
System.out.println("3] str1.insert(23, sangli) = " + str1.insert(23,"sangli "));
System.out.println("4] str1.replace(23,Maharashtra) = " +str1.insert(23,"
Maharashtra "));
System.out.println("5] str1.charAt(24) = " + str1.charAt(24));
System.out.println("6] str1.length() = " + str1.length());
System.out.println("7] str1.delete(41,53) = " + str1.delete(41,53) );
System.out.println("8] str1.reverse() = " + str1.reverse());
}
}
```

Output :



```
D:\22620004>javac StringBuffer1.java

D:\22620004>java StringBuffer1

String = Walchand College of Engineering

String Buffer method

1] str1.append( Sangli) = Walchand College of Engineering Sangli
2] str1.capacity() = 47
3] str1.insert(23, sangli) = Walchand College of Engsangli ineering Sangli
4] str1.replace(23,Maharashtra) = Walchand College of Eng Maharashtra sangli ineering Sangli
5] str1.charAt(24) = M
6] str1.length() = 58
7] str1.delete(41,53) = Walchand College of Eng Maharashtra sanglangli
8] str1.reverse() = ilgnalgnas arthsarahaM gnE fo egelloC dnahclaW

D:\22620004>
```

4. Explain with example declaration of string using string literal and new keyword.

Ans :

1. Using string literal :

   When we use a string literal, a string object is created in the string pool. The string pool is a special area of heap memory where all the string objects are stored.

Example :

String str1 = "Hello";

In this example, a new string object is created with the value "Hello" in the string pool.

2. Using new keyword :

When we use the new keyword to create a string, a new string object is created in the heap memory.

Example :

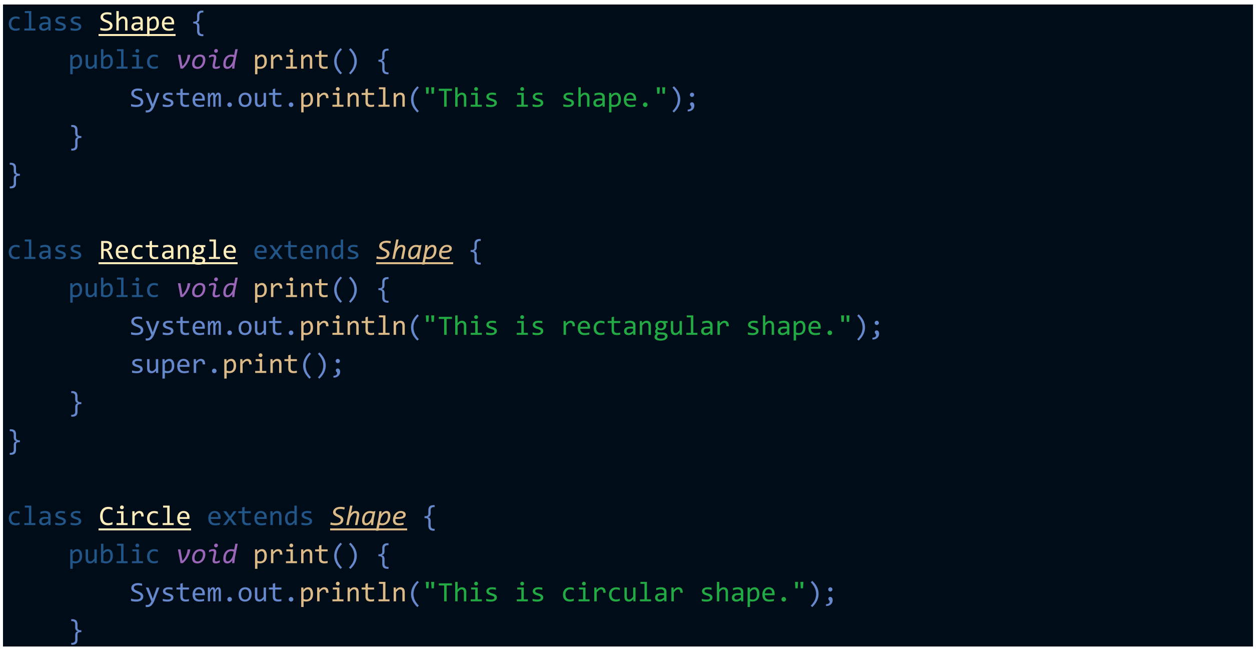
String str2 = new String("Hello");

In this example, a new string object is created in the heap memory with the value "Hello”.

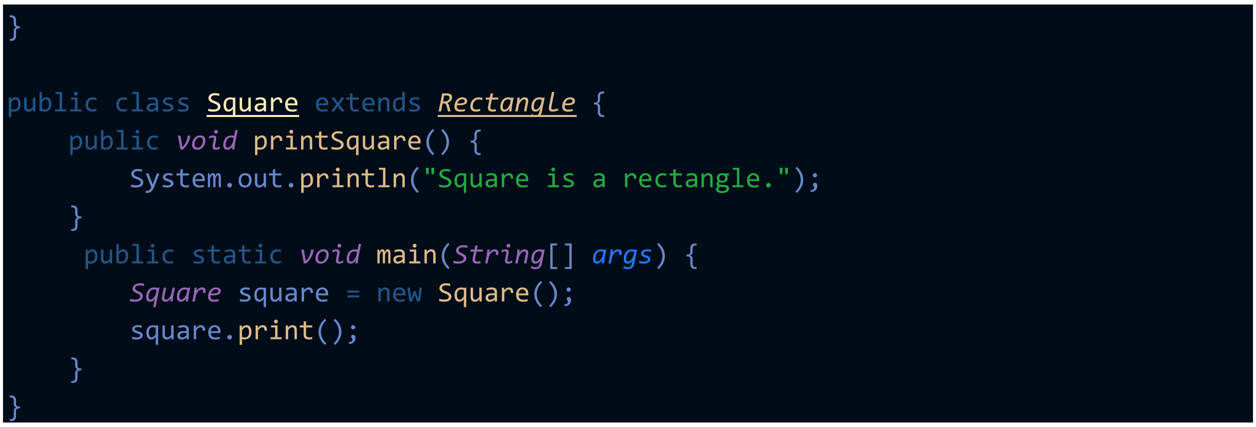
5. Create a class named 'Shape' with a method to print "This is shape". Then create two other classes named 'Rectangle', 'Circle' inheriting the Shape class, both having a method to print "This is rectangular shape" and "This is circular shape" respectively. Create a subclass 'Square' of 'Rectangle' having a method to print "Square is a rectangle". Now call the method of 'Shape' and 'Rectangle' class by the object of 'Square' class.

Ans :

Program :

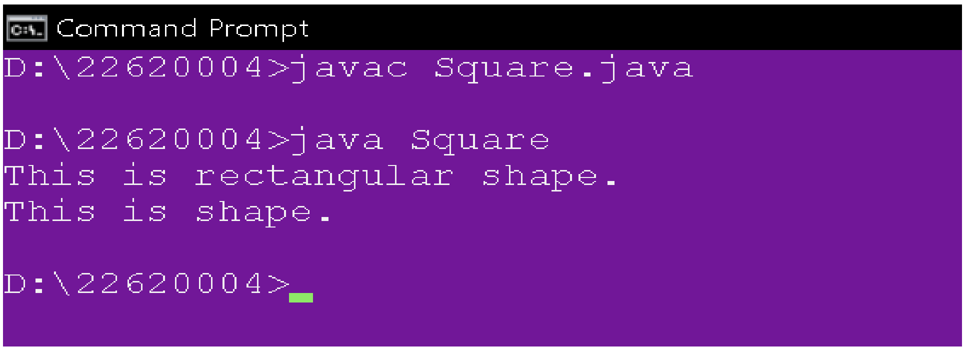


```
class Shape {
    public void print() {
        System.out.println("This is shape.");
    }
}

class Rectangle extends Shape {
    public void print() {
        System.out.println("This is rectangular shape.");
        super.print();
    }
}

class Circle extends Shape {
    public void print() {
        System.out.println("This is circular shape.");
    }
```



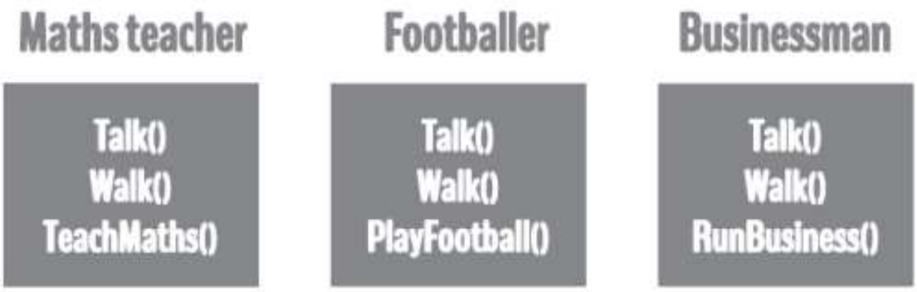
```
}

public class Square extends Rectangle {
    public void printSquare() {
        System.out.println("Square is a rectangle.");
    }
     public static void main(String[] args) {
        Square square = new Square();
        square.print();
    }
}
```

Output :

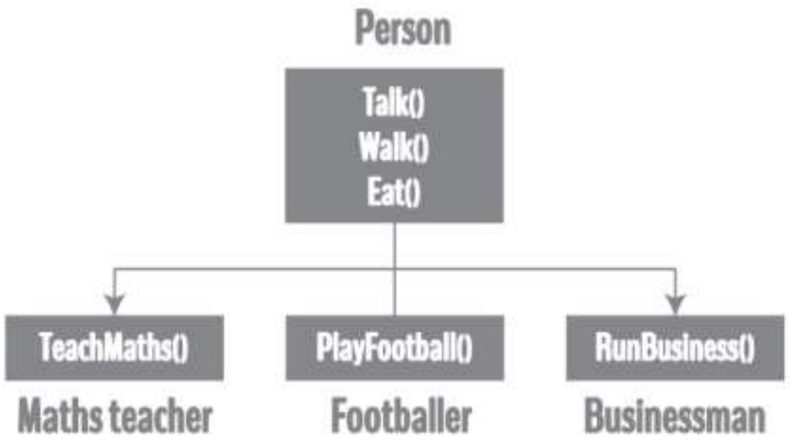


```
D:\22620004>javac Square.java

D:\22620004>java Square
This is rectangular shape.
This is shape.

D:\22620004>
```

6. Create game characters using the concept of inheritance. Suppose, in your game, you want three characters - a maths teacher, a footballer and a businessman. Since, all of the characters are persons, they can walk and talk. However, they also have some special skills. A maths teacher can teach maths, a footballer can play football and a businessman can run a business. You can individually create three classes who can walk, talk and perform their special skill as shown in the figure below.



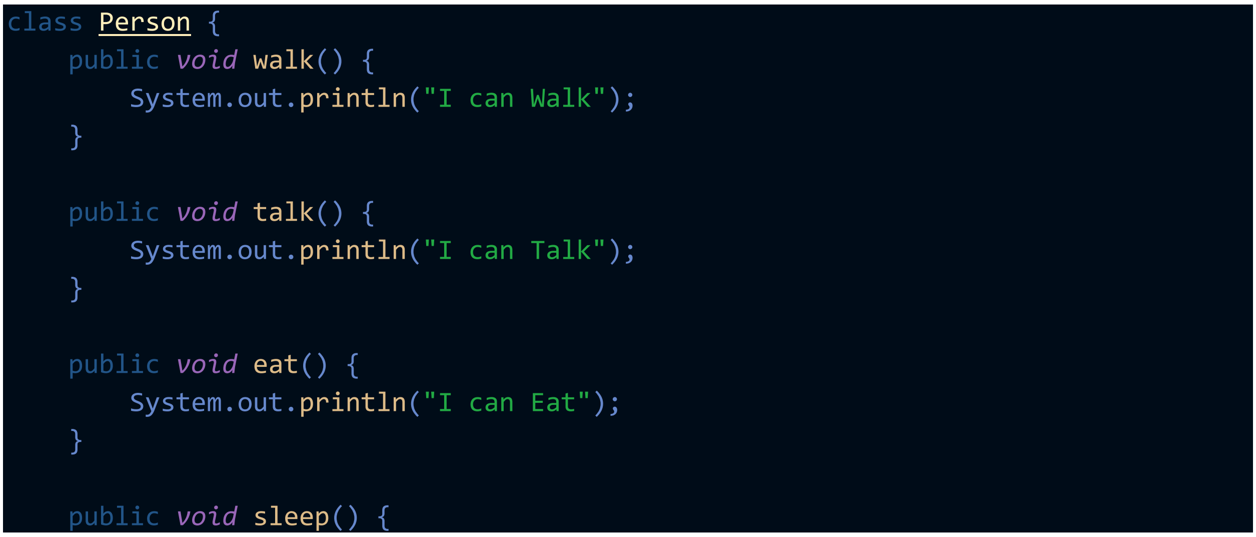
In each of the classes, you would be copying the same code for walk and talk for each character. If you want to add a new feature - eat, you need to implement the same code for each character. This can easily become error prone (when copying) and duplicate codes. It'd be a lot easier if we had a Person class with basic features like talk, walk, eat, sleep, and add special skills to those features as per our characters. This is done using inheritance.



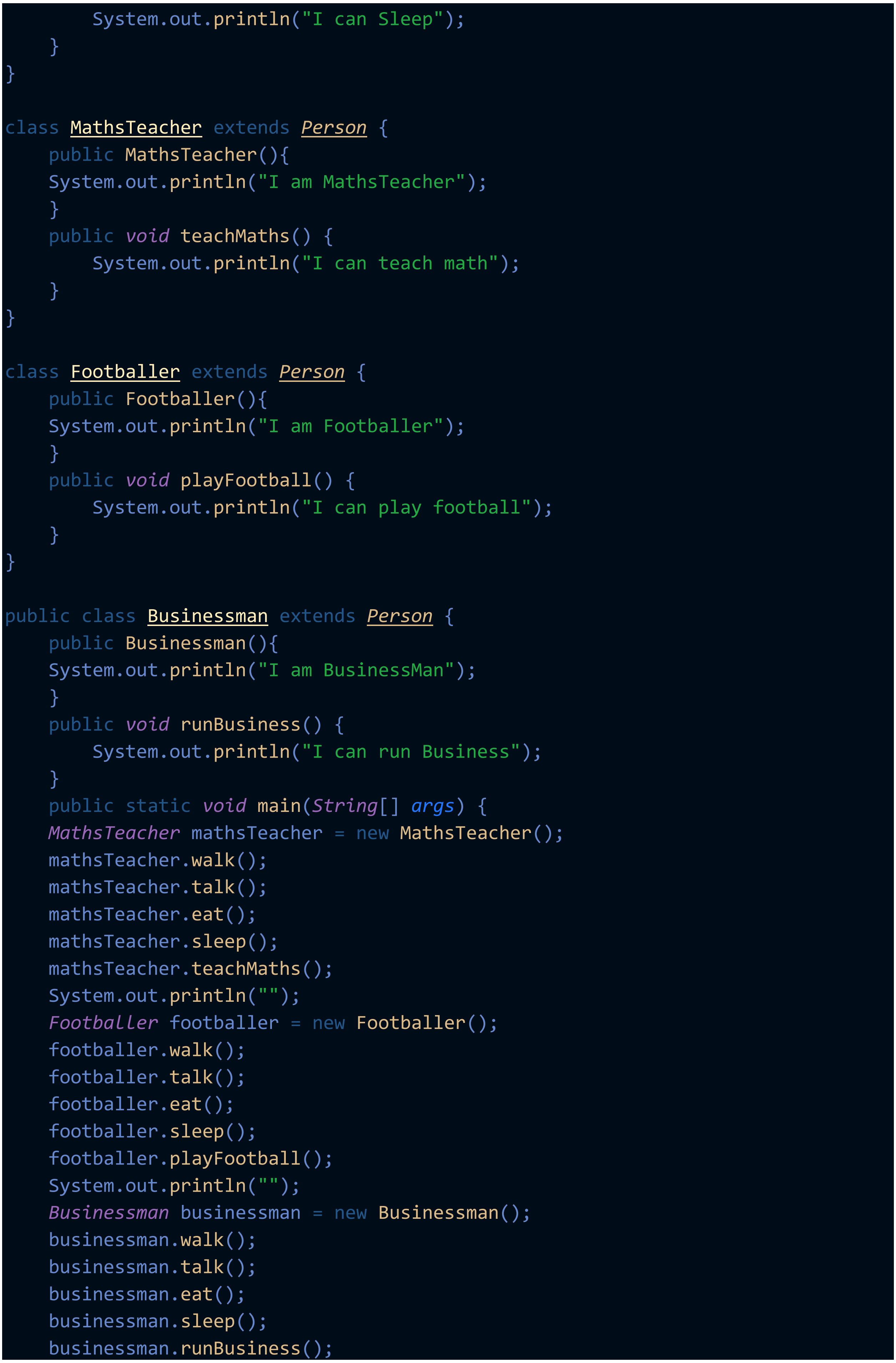
Using inheritance, now you don't implement the same code for walk and talk for each class. You just need to inherit them. So, for Maths teacher (derived class), you inherit all features of a Person (base class) and add a new feature TeachMaths. Likewise, for a footballer, you inherit all the features of a Person and add a new feature PlayFootball and so on.

Ans :

Program :



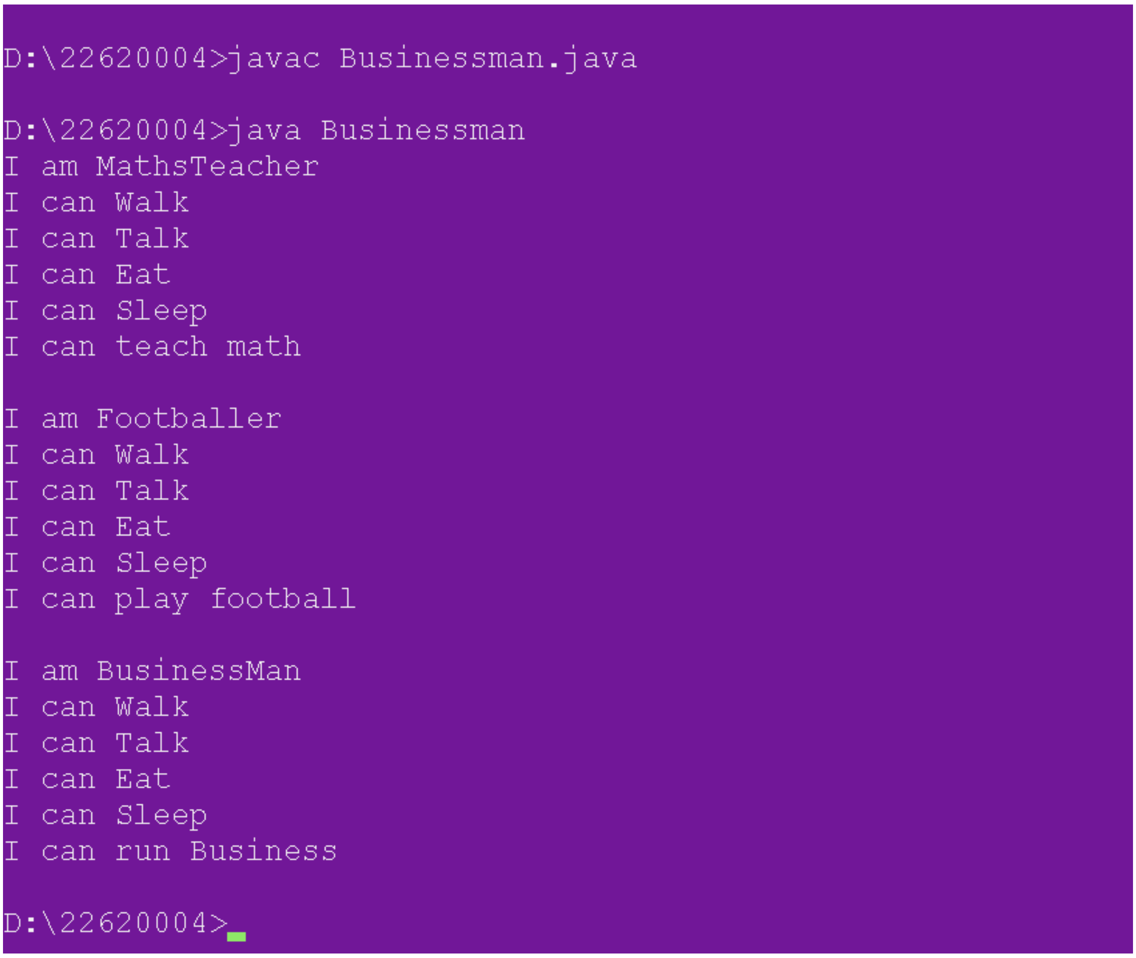
```
class Person {
    public void walk() {
        System.out.println("I can Walk");
    }

    public void talk() {
        System.out.println("I can Talk");
    }

    public void eat() {
        System.out.println("I can Eat");
    }

    public void sleep() {
```



```
        System.out.println("I can Sleep");
    }
}

class MathsTeacher extends Person {
    public MathsTeacher(){
    System.out.println("I am MathsTeacher");
    }
    public void teachMaths() {
        System.out.println("I can teach math");
    }
}

class Footballer extends Person {
    public Footballer(){
    System.out.println("I am Footballer");
    }
    public void playFootball() {
        System.out.println("I can play football");
    }
}

public class Businessman extends Person {
    public Businessman(){
    System.out.println("I am BusinessMan");
    }
    public void runBusiness() {
        System.out.println("I can run Business");
    }
    public static void main(String[] args) {
    MathsTeacher mathsTeacher = new MathsTeacher();
    mathsTeacher.walk();
    mathsTeacher.talk();
    mathsTeacher.eat();
    mathsTeacher.sleep();
    mathsTeacher.teachMaths();
    System.out.println("");
    Footballer footballer = new Footballer();
    footballer.walk();
    footballer.talk();
    footballer.eat();
    footballer.sleep();
    footballer.playFootball();
    System.out.println("");
    Businessman businessman = new Businessman();
    businessman.walk();
    businessman.talk();
    businessman.eat();
    businessman.sleep();
    businessman.runBusiness();
```

```
}
}
```

Output :

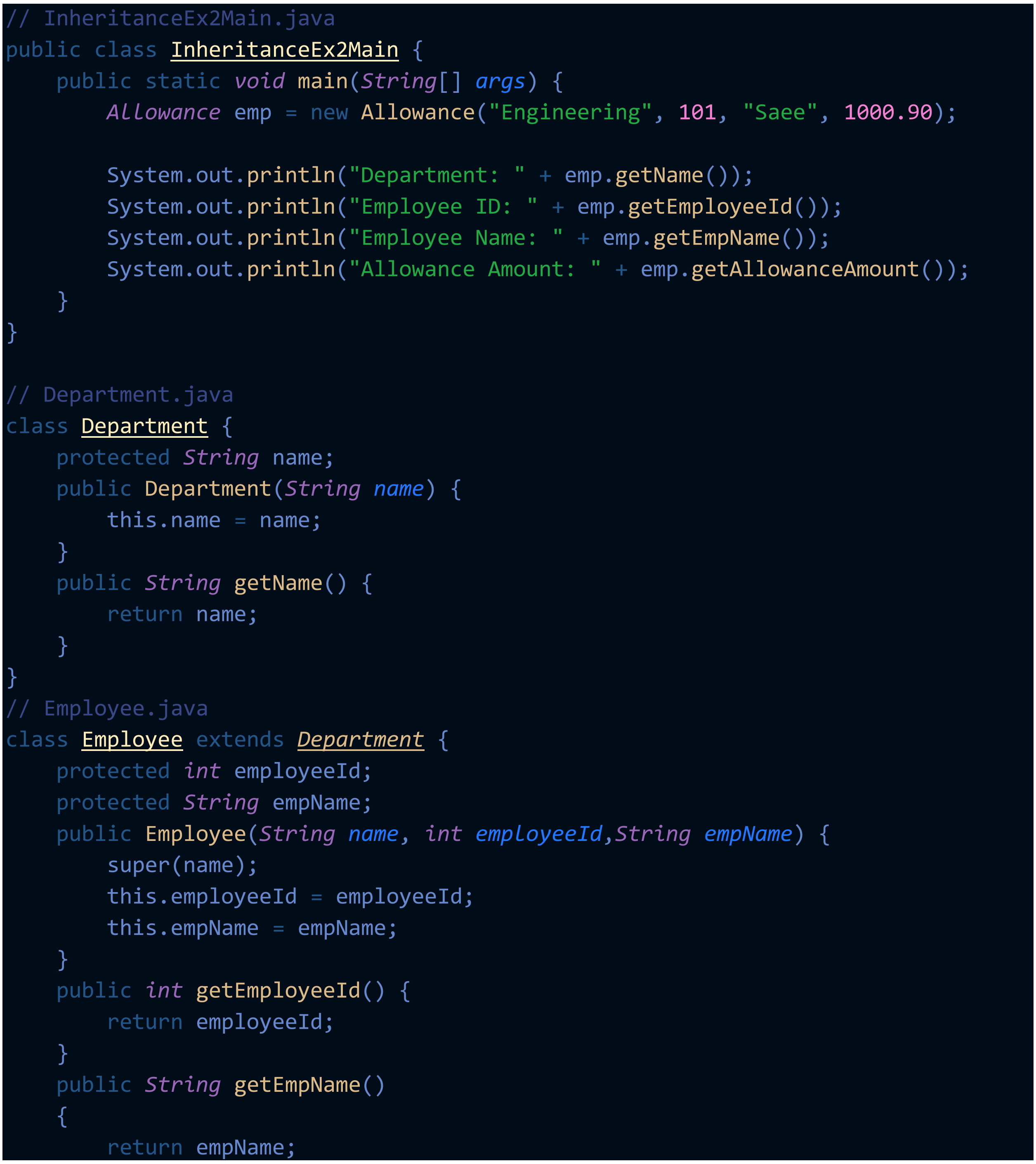


```
D:\22620004>javac Businessman.java

D:\22620004>java Businessman
I am MathsTeacher
I can Walk
I can Talk
I can Eat
I can Sleep
I can teach math

I am Footballer
I can Walk
I can Talk
I can Eat
I can Sleep
I can play football

I am BusinessMan
I can Walk
I can Talk
I can Eat
I can Sleep
I can run Business

D:\22620004>_
```

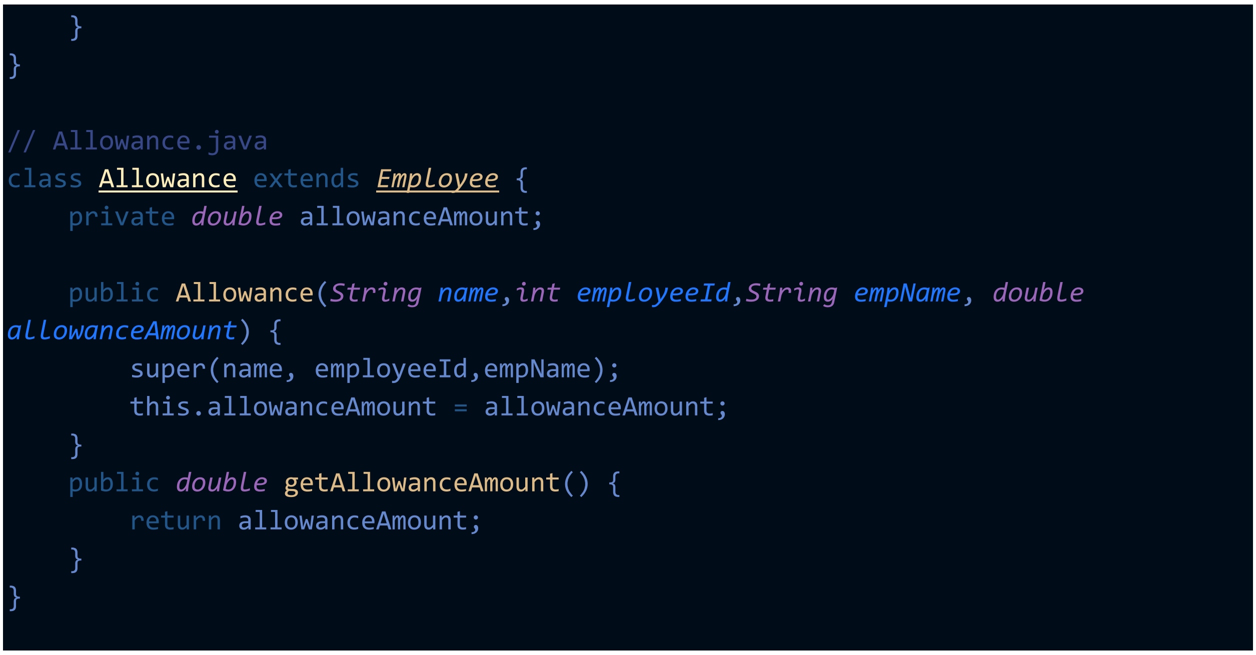
7 . WAP to manage the employee allowance from a specific department by
creating class structure as follow,
InheritanceEx2
|
|
InheritanceEx2Main.java
|
|- dept | Deparment.java
|
|
|
|- emp | Employee.java extends Department
|

|
|
|- allowance | Allowance.java extends Employee
|
| [Multilevel Inheritance]

Ans :
Program :

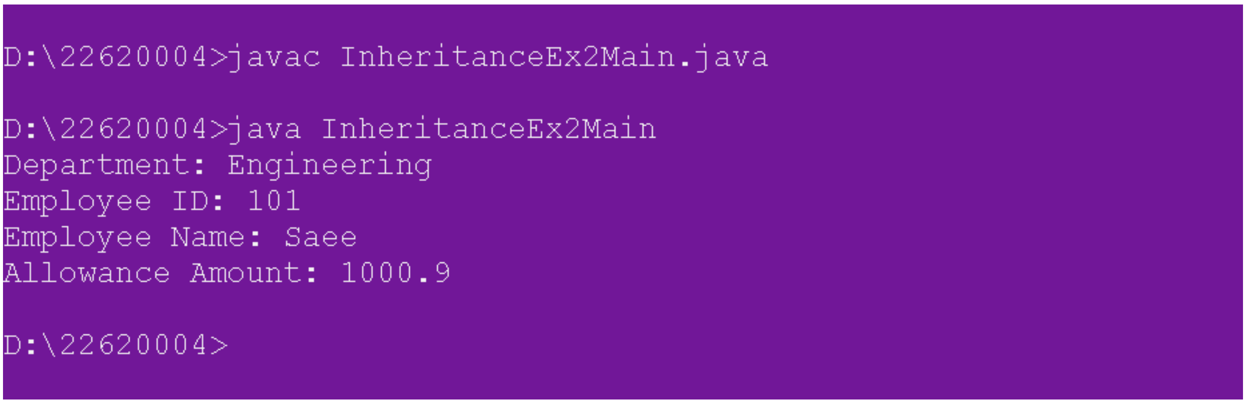


```java
// InheritanceEx2Main.java
public class InheritanceEx2Main {
    public static void main(String[] args) {
        Allowance emp = new Allowance("Engineering", 101, "Saee", 1000.90);

        System.out.println("Department: " + emp.getName());
        System.out.println("Employee ID: " + emp.getEmployeeId());
        System.out.println("Employee Name: " + emp.getEmpName());
        System.out.println("Allowance Amount: " + emp.getAllowanceAmount());
    }
}

// Department.java
class Department {
    protected String name;
    public Department(String name) {
        this.name = name;
    }
    public String getName() {
        return name;
    }
}
// Employee.java
class Employee extends Department {
    protected int employeeId;
    protected String empName;
    public Employee(String name, int employeeId,String empName) {
        super(name);
        this.employeeId = employeeId;
        this.empName = empName;
    }
    public int getEmployeeId() {
        return employeeId;
    }
    public String getEmpName()
    {
        return empName;
```



```
    }
}

// Allowance.java
class Allowance extends Employee {
    private double allowanceAmount;

    public Allowance(String name,int employeeId,String empName, double
allowanceAmount) {
        super(name, employeeId,empName);
        this.allowanceAmount = allowanceAmount;
    }
    public double getAllowanceAmount() {
        return allowanceAmount;
    }
}
```

Output :

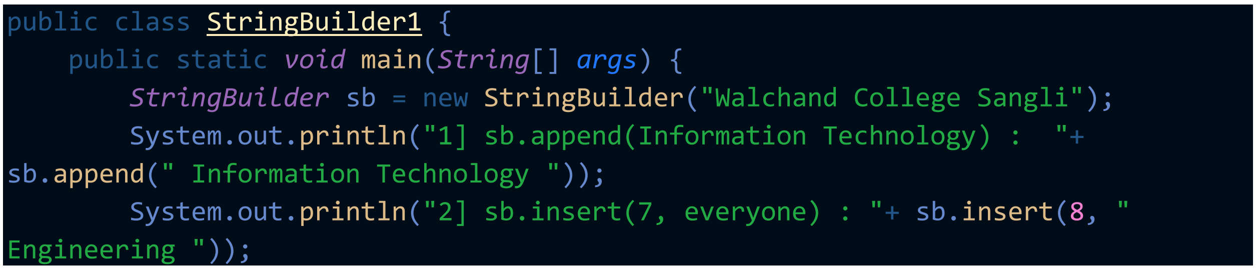


```
D:\22620004>javac InheritanceEx2Main.java

D:\22620004>java InheritanceEx2Main
Department: Engineering
Employee ID: 101
Employee Name: Saee
Allowance Amount: 1000.9

D:\22620004>
```

8. Write a Java Program to demonstrate StringBuilder class methods

Ans :

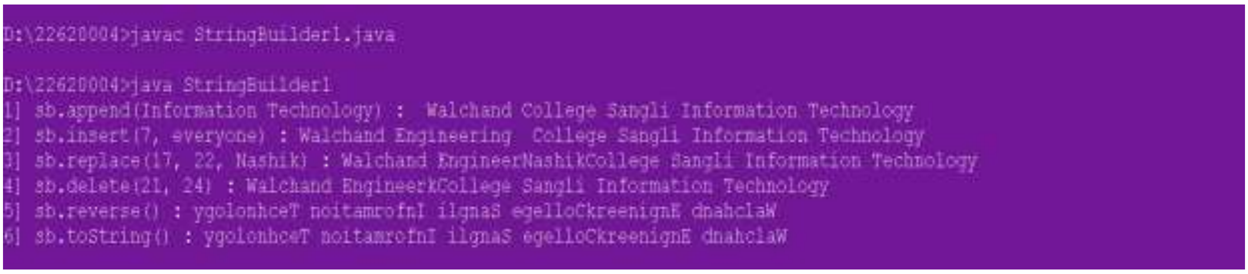
Program :



```
public class StringBuilder1 {
    public static void main(String[] args) {
        StringBuilder sb = new StringBuilder("Walchand College Sangli");
        System.out.println("1] sb.append(Information Technology) :  "+
sb.append(" Information Technology "));
        System.out.println("2] sb.insert(7, everyone) : "+ sb.insert(8, "
Engineering "));
```

```
        System.out.println("3] sb.replace(17, 22, Nashik) : "+ sb.replace(17,
22, "Nashik"));
        System.out.println("4] sb.delete(21, 24) : "+  sb.delete(17, 22));
        System.out.println("5] sb.reverse() : "+ sb.reverse());
        System.out.println("6] sb.toString() : "+sb.toString());
    }
}
```

Output :



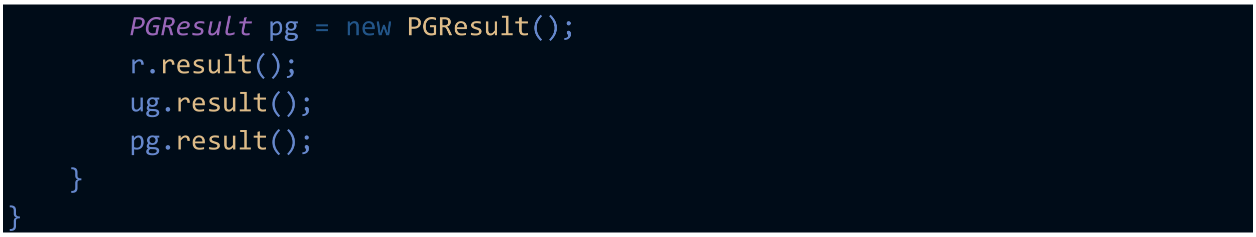
```
D:\22620004>javac StringBuilder1.java

D:\22620004>java StringBuilder1
1] sb.append(Information Technology) :  Walchand College Sangli Information Technology
2] sb.insert(7, everyone) : Walchand Engineering  College Sangli Information Technology
3] sb.replace(17, 22, Nashik) : Walchand EngineerNashikCollege Sangli Information Technology
4] sb.delete(21, 24) : Walchand EngineerkCollege Sangli Information Technology
5] sb.reverse() : ygolonhceT noitamrofnI ilgnaS egelloCkreenignE dnahclaW
6] sb.toString() : ygolonhceT noitamrofnI ilgnaS egelloCkreenignE dnahclaW
```

9. Write a Java Program to demonstrate Method overriding.(create class Result with method result(). Override method result() in UGResult and PGResult class)

Ans :

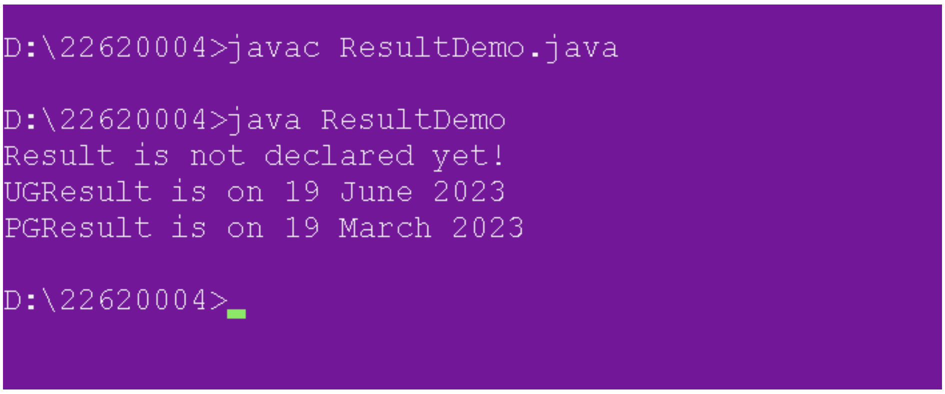
Program :

```
class Result {
    public void result() {
        System.out.println("Result is not declared yet!");
    }
}
class UGResult extends Result {
    public void result() {
        System.out.println("UGResult is on 19 June 2023");
    }
}

class PGResult extends Result {
    public void result() {
        System.out.println("PGResult is on 19 March 2023");
    }
}
public class ResultDemo {
    public static void main(String[] args) {
        Result r = new Result();
        UGResult ug = new UGResult();
```



```
        PGResult pg = new PGResult();
        r.result();
        ug.result();
        pg.result();
    }
}
```

Output :

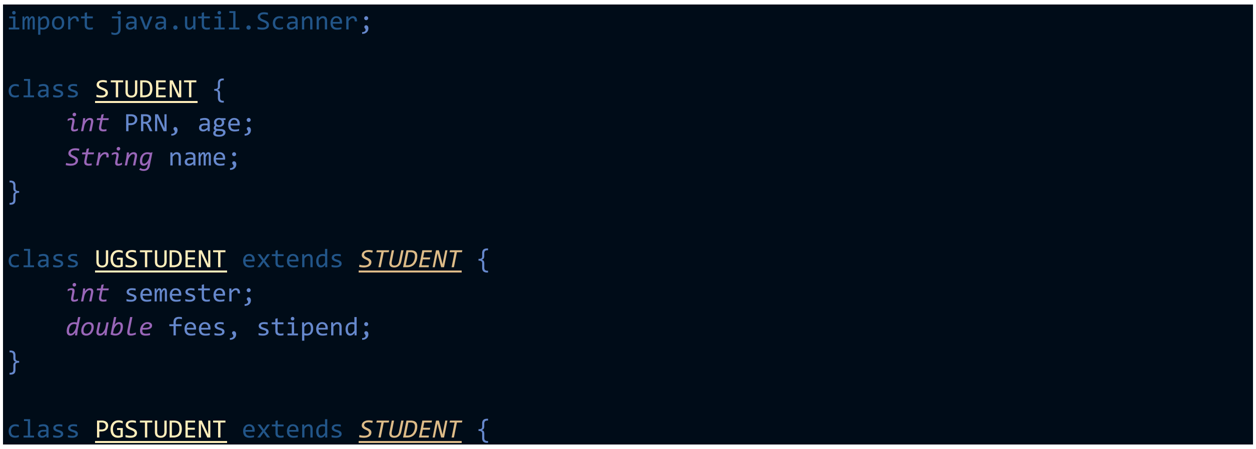


```
D:\22620004>javac ResultDemo.java

D:\22620004>java ResultDemo
Result is not declared yet!
UGResult is on 19 June 2023
PGResult is on 19 March 2023

D:\22620004>
```

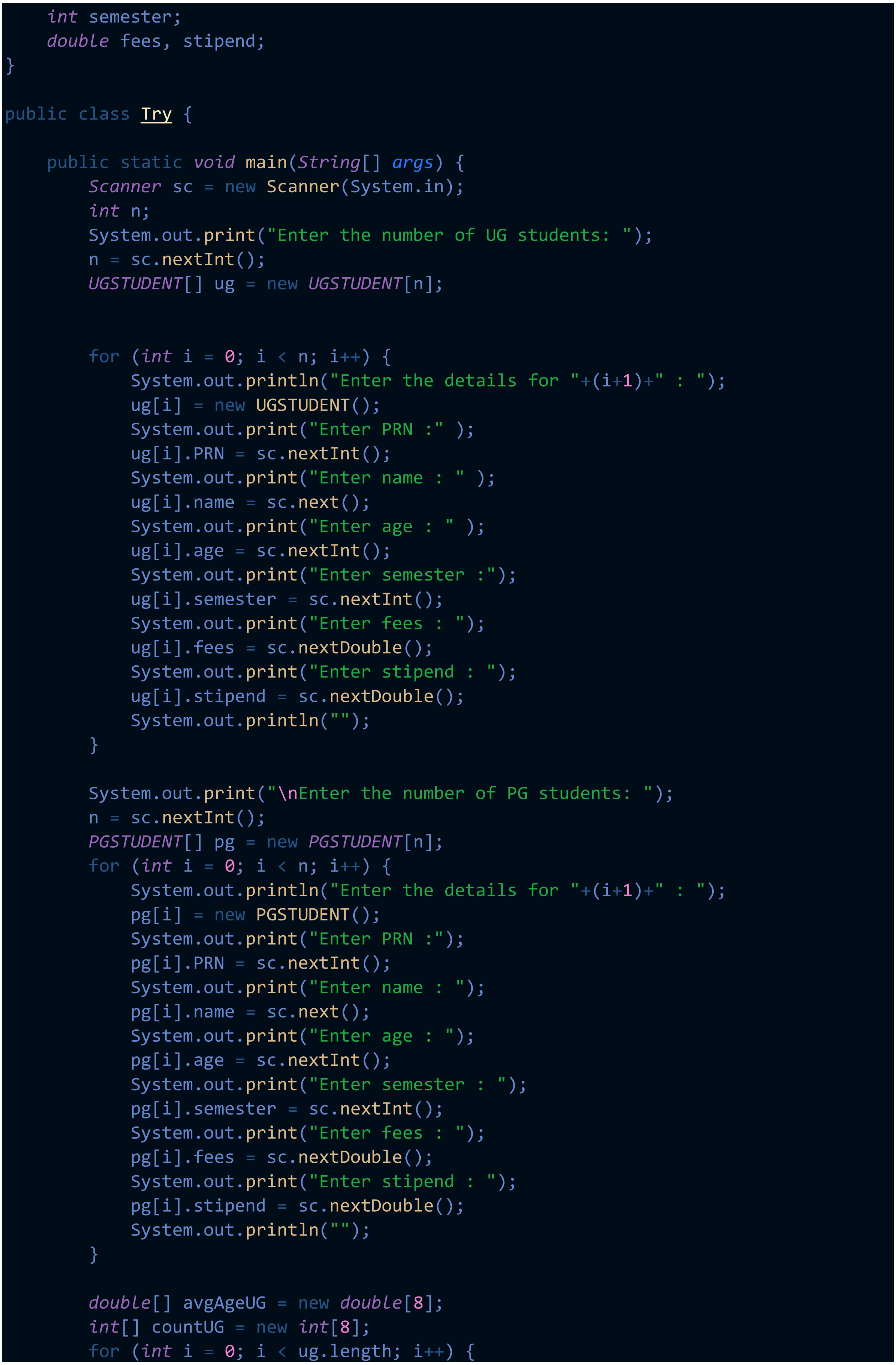
10. Write a java program to create a class called STUDENT with data members PRN, Name and age. Using inheritance, create a classes called UGSTUDENT and PGSTUDENT having fields as semester, fees and stipend. Enter the data for at least 5 students. Find the semester wise average age for all UG and PG students separately.

Ans :

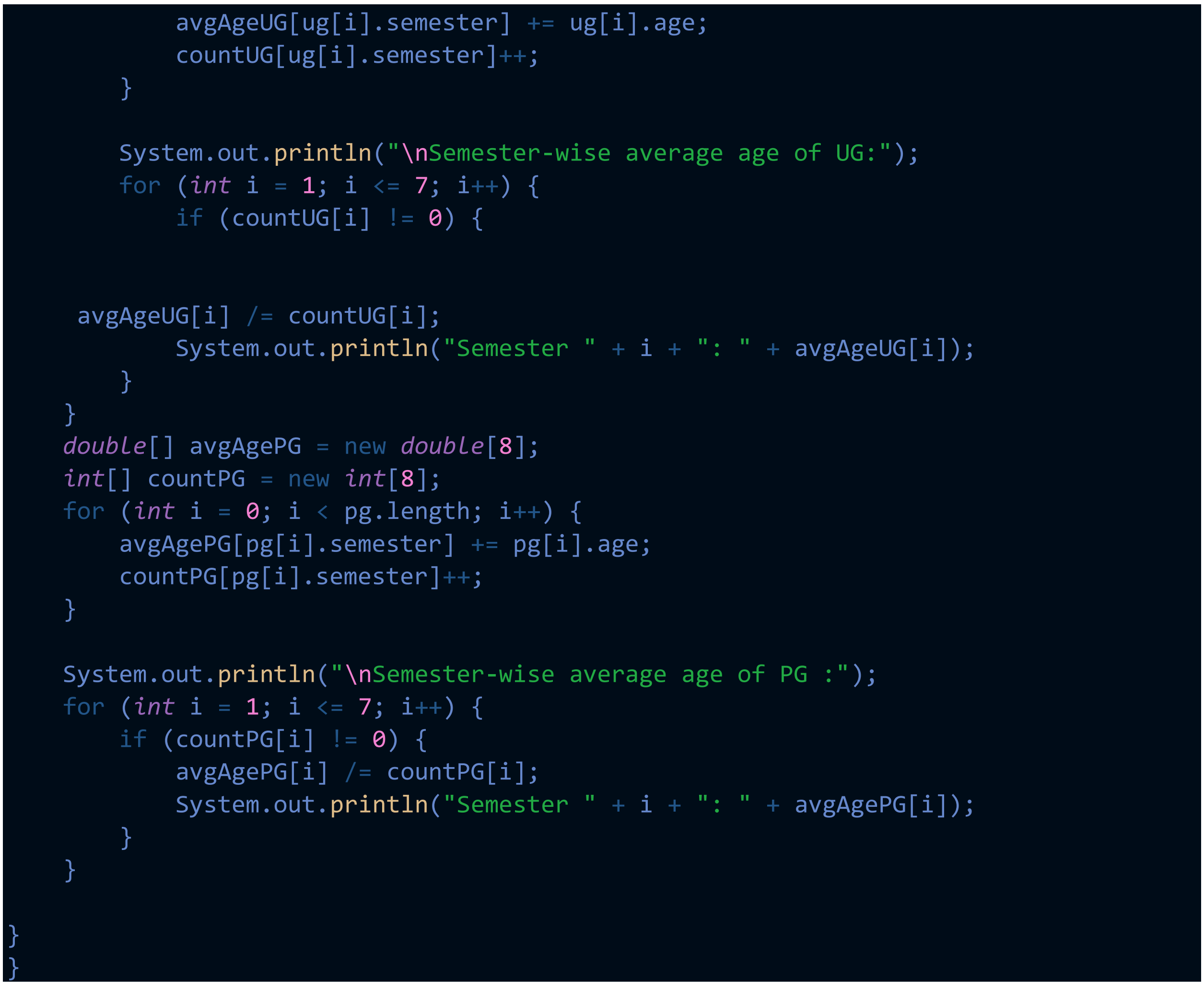
Program :



```
import java.util.Scanner;

class STUDENT {
    int PRN, age;
    String name;
}

class UGSTUDENT extends STUDENT {
    int semester;
    double fees, stipend;
}

class PGSTUDENT extends STUDENT {
```

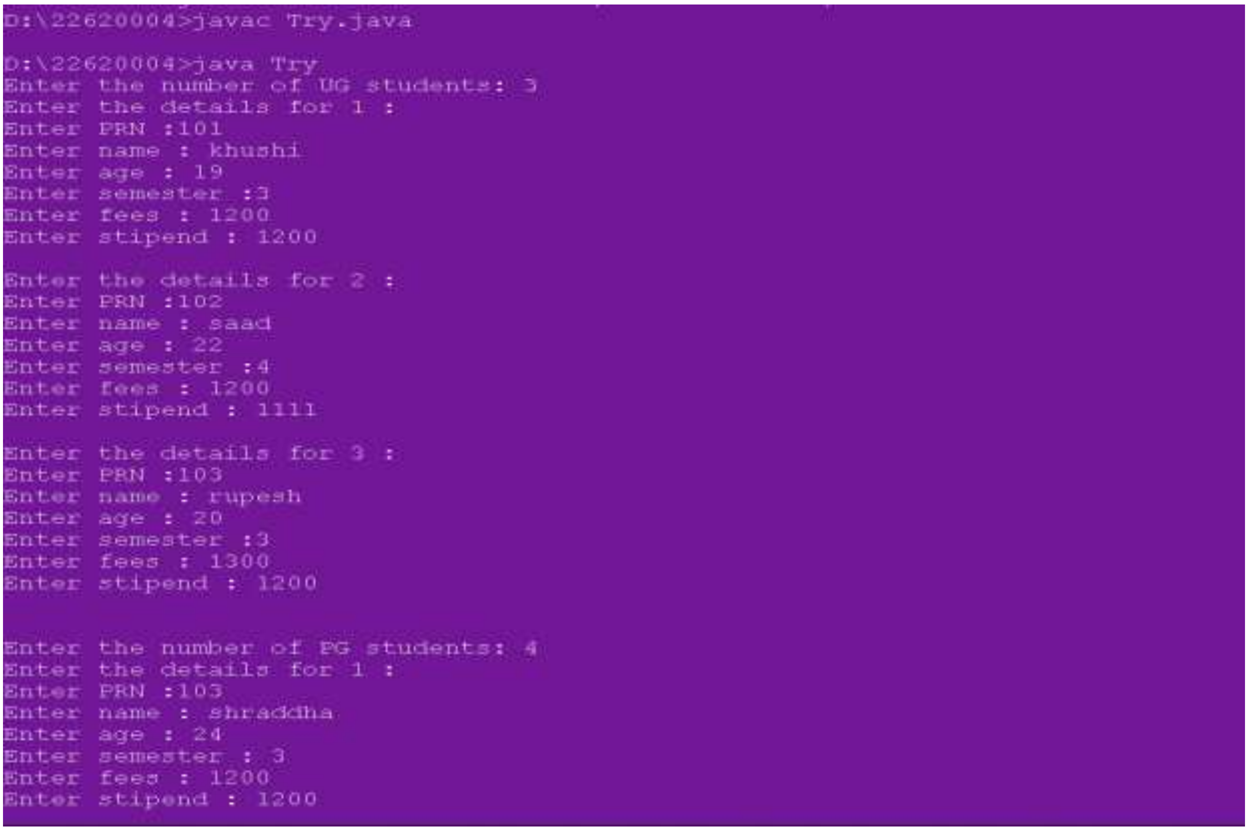


```
    int semester;
    double fees, stipend;
}

public class Try {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n;
        System.out.print("Enter the number of UG students: ");
        n = sc.nextInt();
        UGSTUDENT[] ug = new UGSTUDENT[n];


        for (int i = 0; i < n; i++) {
            System.out.println("Enter the details for "+(i+1)+" : ");
            ug[i] = new UGSTUDENT();
            System.out.print("Enter PRN :" );
            ug[i].PRN = sc.nextInt();
            System.out.print("Enter name : " );
            ug[i].name = sc.next();
            System.out.print("Enter age : " );
            ug[i].age = sc.nextInt();
            System.out.print("Enter semester :");
            ug[i].semester = sc.nextInt();
            System.out.print("Enter fees : ");
            ug[i].fees = sc.nextDouble();
            System.out.print("Enter stipend : ");
            ug[i].stipend = sc.nextDouble();
            System.out.println("");
        }

        System.out.print("\nEnter the number of PG students: ");
        n = sc.nextInt();
        PGSTUDENT[] pg = new PGSTUDENT[n];
        for (int i = 0; i < n; i++) {
            System.out.println("Enter the details for "+(i+1)+" : ");
            pg[i] = new PGSTUDENT();
            System.out.print("Enter PRN :");
            pg[i].PRN = sc.nextInt();
            System.out.print("Enter name : ");
            pg[i].name = sc.next();
            System.out.print("Enter age : ");
            pg[i].age = sc.nextInt();
            System.out.print("Enter semester : ");
            pg[i].semester = sc.nextInt();
            System.out.print("Enter fees : ");
            pg[i].fees = sc.nextDouble();
            System.out.print("Enter stipend : ");
            pg[i].stipend = sc.nextDouble();
            System.out.println("");
        }

        double[] avgAgeUG = new double[8];
        int[] countUG = new int[8];
        for (int i = 0; i < ug.length; i++) {
```

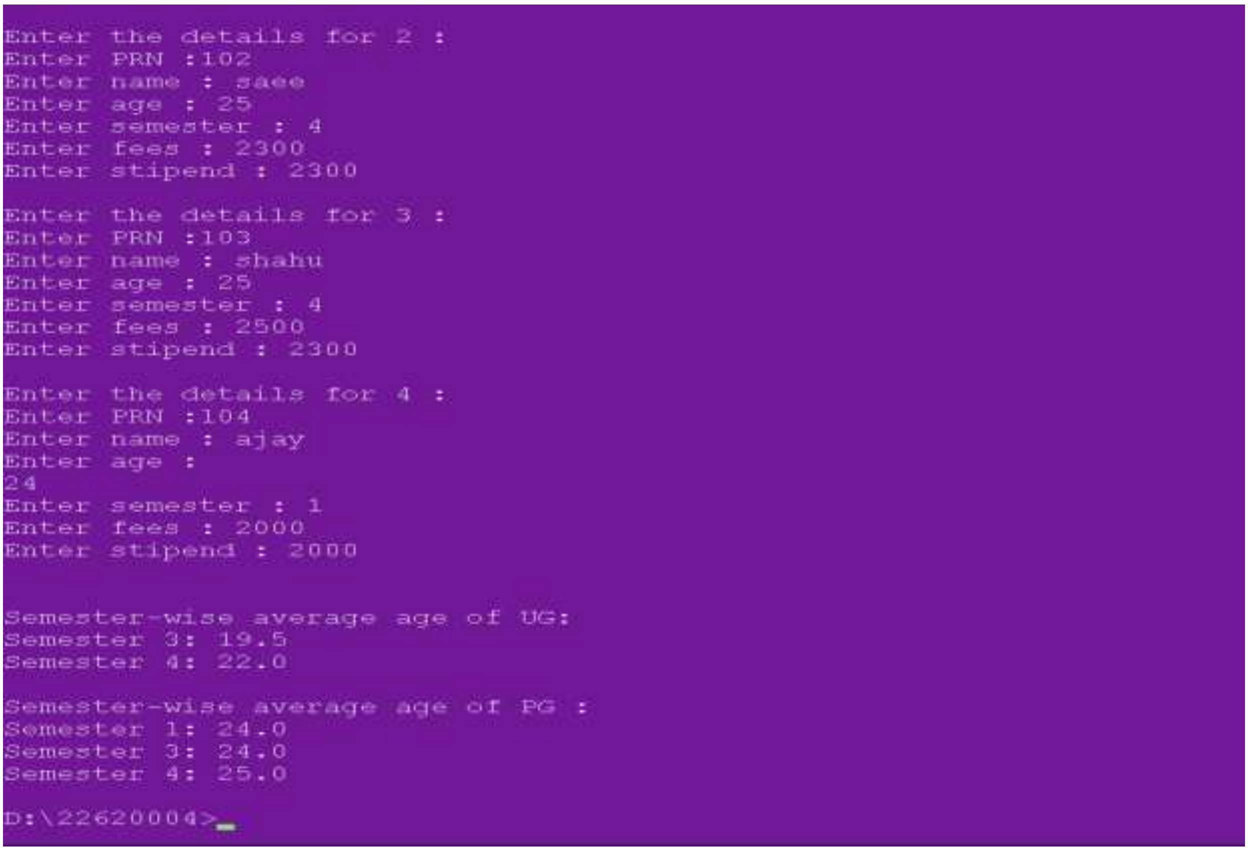


```
            avgAgeUG[ug[i].semester] += ug[i].age;
            countUG[ug[i].semester]++;
        }

        System.out.println("\nSemester-wise average age of UG:");
        for (int i = 1; i <= 7; i++) {
            if (countUG[i] != 0) {


     avgAgeUG[i] /= countUG[i];
            System.out.println("Semester " + i + ": " + avgAgeUG[i]);
        }
    }
    double[] avgAgePG = new double[8];
    int[] countPG = new int[8];
    for (int i = 0; i < pg.length; i++) {
        avgAgePG[pg[i].semester] += pg[i].age;
        countPG[pg[i].semester]++;
    }

    System.out.println("\nSemester-wise average age of PG :");
    for (int i = 1; i <= 7; i++) {
        if (countPG[i] != 0) {
            avgAgePG[i] /= countPG[i];
            System.out.println("Semester " + i + ": " + avgAgePG[i]);
        }
    }

}
}
```

Output:



```
D:\22620004>javac Try.java

D:\22620004>java Try
Enter the number of UG students: 3
Enter the details for 1 :
Enter PRN :101
Enter name : khushi
Enter age : 19
Enter semester :3
Enter fees : 1200
Enter stipend : 1200

Enter the details for 2 :
Enter PRN :102
Enter name : saad
Enter age : 22
Enter semester :4
Enter fees : 1200
Enter stipend : 1111

Enter the details for 3 :
Enter PRN :103
Enter name : rupesh
Enter age : 20
Enter semester :3
Enter fees : 1300
Enter stipend : 1200


Enter the number of PG students: 4
Enter the details for 1 :
Enter PRN :103
Enter name : shraddha
Enter age : 24
Enter semester : 3
Enter fees : 1200
Enter stipend : 1200
```

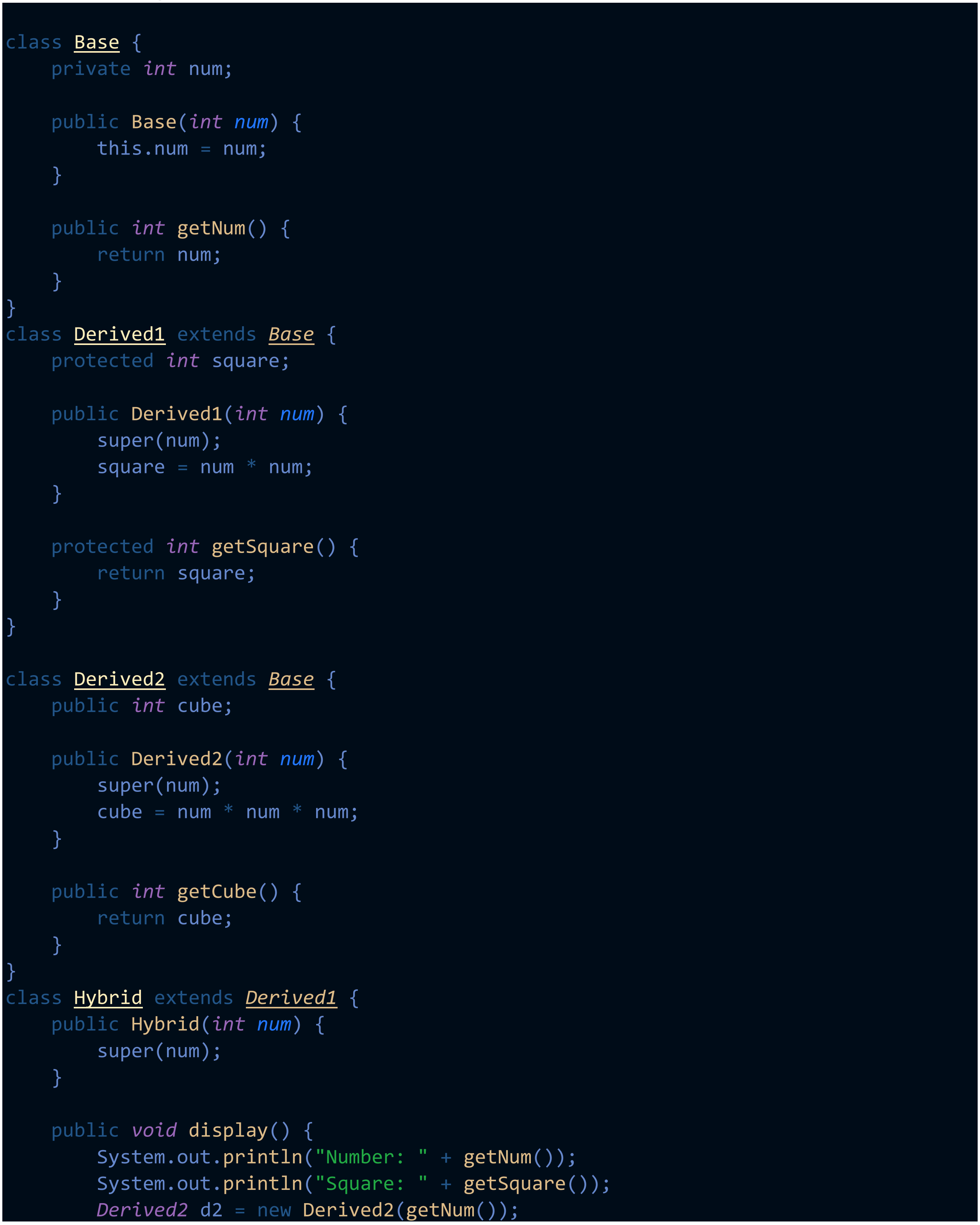


```
Enter the details for 2 :
Enter PRN :102
Enter name : saee
Enter age : 25
Enter semester : 4
Enter fees : 2300
Enter stipend : 2300

Enter the details for 3 :
Enter PRN :103
Enter name : shahu
Enter age : 25
Enter semester : 4
Enter fees : 2500
Enter stipend : 2300

Enter the details for 4 :
Enter PRN :104
Enter name : ajay
Enter age :
24
Enter semester : 1
Enter fees : 2000
Enter stipend : 2000


Semester-wise average age of UG:
Semester 3: 19.5
Semester 4: 22.0

Semester-wise average age of PG :
Semester 1: 24.0
Semester 3: 24.0
Semester 4: 25.0

D:\22620004>_
```

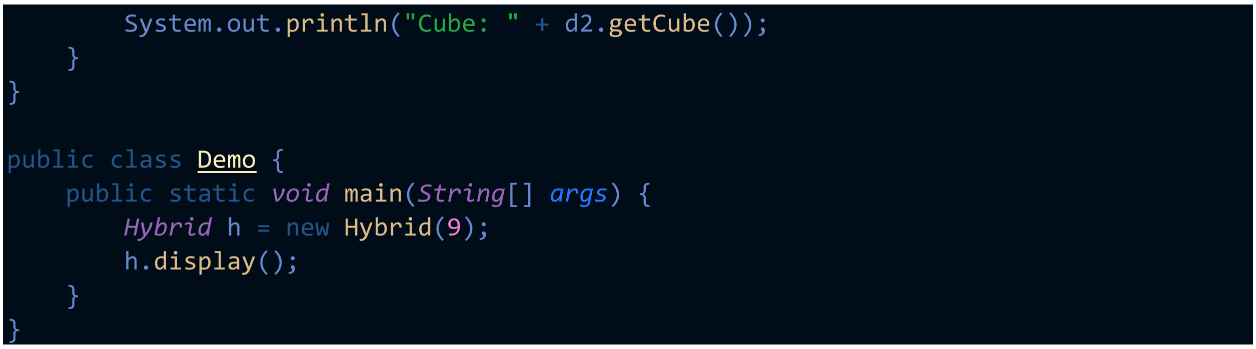
11. Implement hybrid inheritance using all access specifiers (public, private, protected).

Ans :

Program :

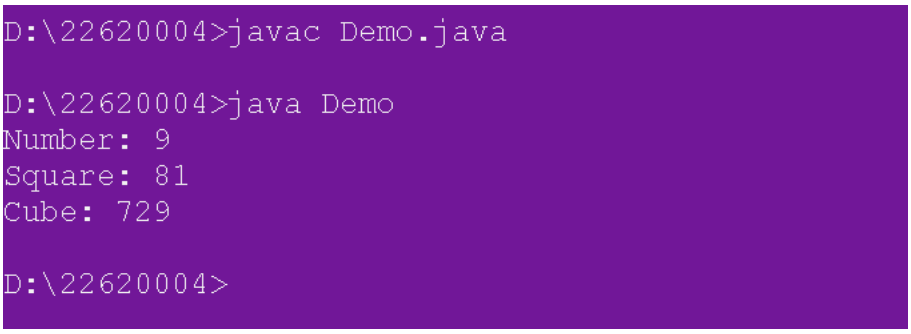


```java
class Base {
    private int num;

    public Base(int num) {
        this.num = num;
    }

    public int getNum() {
        return num;
    }
}
class Derived1 extends Base {
    protected int square;

    public Derived1(int num) {
        super(num);
        square = num * num;
    }

    protected int getSquare() {
        return square;
    }
}

class Derived2 extends Base {
    public int cube;

    public Derived2(int num) {
        super(num);
        cube = num * num * num;
    }

    public int getCube() {
        return cube;
    }
}
class Hybrid extends Derived1 {
    public Hybrid(int num) {
        super(num);
    }

    public void display() {
        System.out.println("Number: " + getNum());
        System.out.println("Square: " + getSquare());
        Derived2 d2 = new Derived2(getNum());
```



```
        System.out.println("Cube: " + d2.getCube());
    }
}

public class Demo {
    public static void main(String[] args) {
        Hybrid h = new Hybrid(9);
        h.display();
    }
}
```

Output :

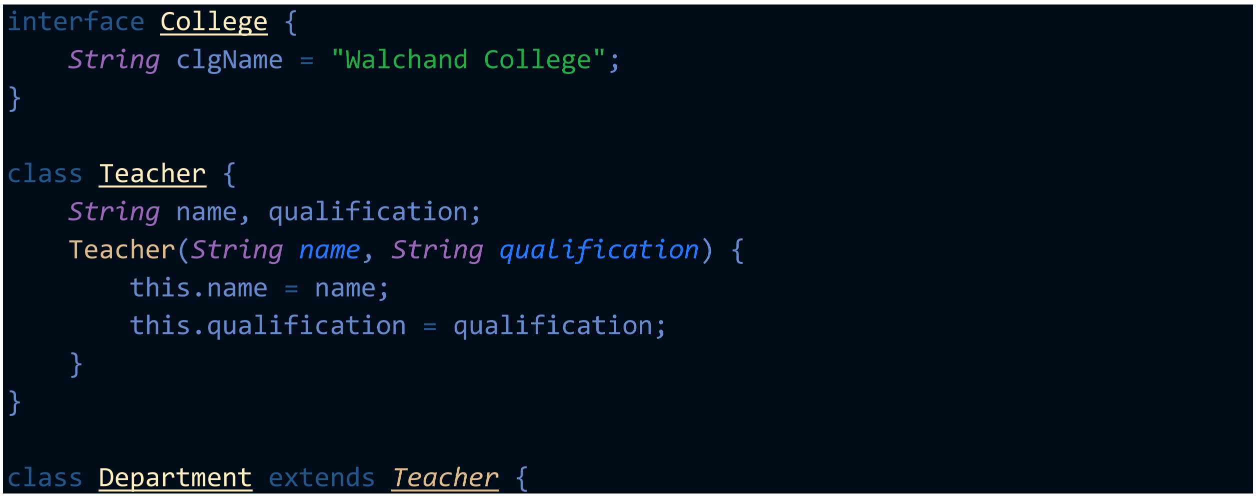


```
D:\22620004>javac Demo.java

D:\22620004>java Demo
Number: 9
Square: 81
Cube: 729

D:\22620004>
```

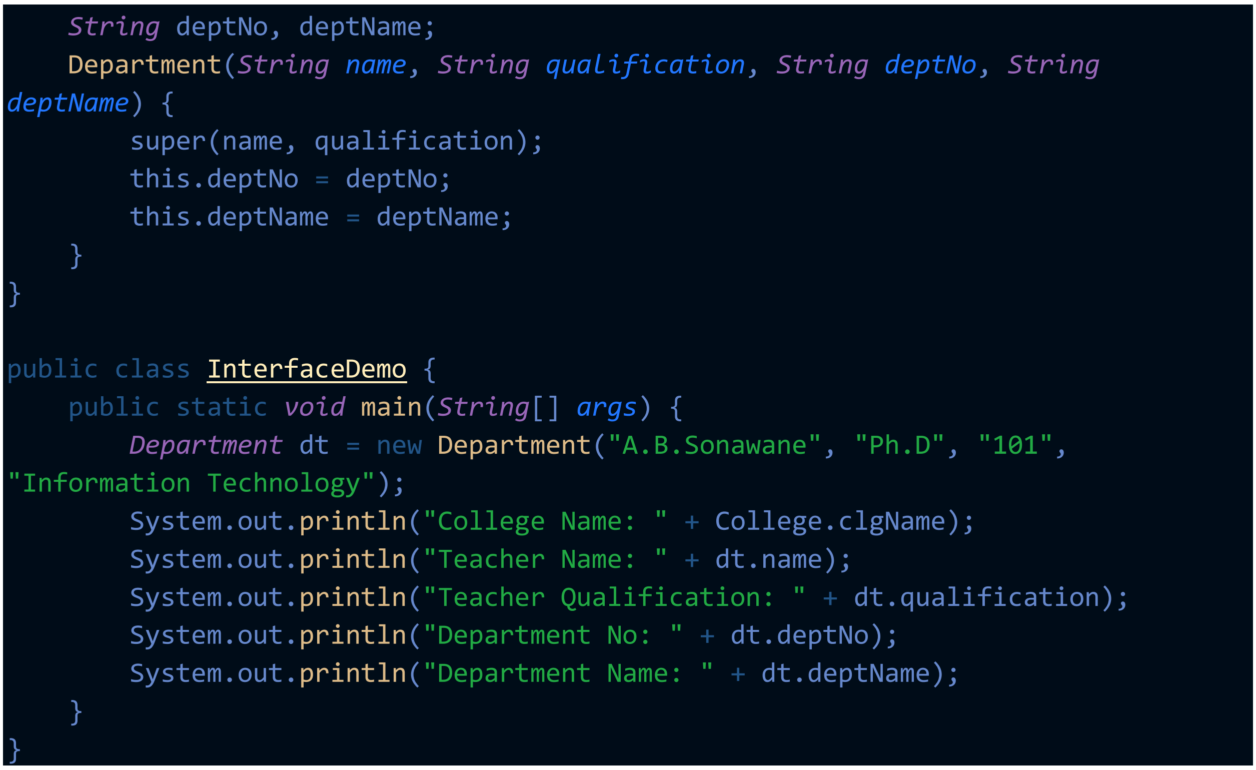
12. Write a program to implement a class Teacher contains two fields Name and Qualification. Extend the class to Department, it contains Dept. No and Dept. Name. An Interface named as College it contains one field Name of the College. Using the above classes and Interface get the appropriate information and display it.

Ans :

Program :

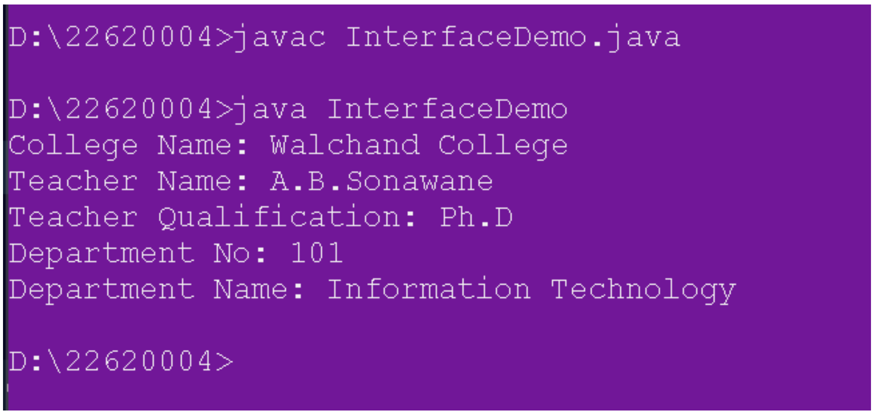


```
interface College {
    String clgName = "Walchand College";
}

class Teacher {
    String name, qualification;
    Teacher(String name, String qualification) {
        this.name = name;
        this.qualification = qualification;
    }
}

class Department extends Teacher {
```



```
    String deptNo, deptName;
    Department(String name, String qualification, String deptNo, String
deptName) {
        super(name, qualification);
        this.deptNo = deptNo;
        this.deptName = deptName;
    }
}

public class InterfaceDemo {
    public static void main(String[] args) {
        Department dt = new Department("A.B.Sonawane", "Ph.D", "101",
"Information Technology");
        System.out.println("College Name: " + College.clgName);
        System.out.println("Teacher Name: " + dt.name);
        System.out.println("Teacher Qualification: " + dt.qualification);
        System.out.println("Department No: " + dt.deptNo);
        System.out.println("Department Name: " + dt.deptName);
    }
}
```

Output :



```
D:\22620004>javac InterfaceDemo.java

D:\22620004>java InterfaceDemo
College Name: Walchand College
Teacher Name: A.B.Sonawane
Teacher Qualification: Ph.D
Department No: 101
Department Name: Information Technology

D:\22620004>
```