Java Programming Lab

Experiment - 4

**Class, Objects, Methods, and Constructors**

Submitted by: Submitted to:

Akansha Saurabh Jain

500117744

1. **Write a JAVA program to implement class mechanism. – Create a class, define data members, methods (setData(args) ,getData()) and invoke them inside main method.**

**Algorithm:**

* Create a class named `Stu` to store student information.
* In the `Stu` class, declare three properties: `Roll` to store the student's roll number (an integer), `Name` to store the student's name (a string), and `Course` to store the name of the course the student is enrolled in (a string).
* Create a method named `setData` in the `Stu` class. This method takes three parameters: `Roll\_no` (integer), `Namee` (string), and `Crse` (string). The purpose of this method is to assign the value of `Roll\_no` to `Roll`, assign the value of `Namee` to `Name`, and assign the value of `Crse` to `Course`.
* Create another method named `getData` in the `Stu` class. This method doesn't take any parameters and is used to display the student's information. When called, it prints the roll number, name, and course name to the console.
* Create an instance of the `Stu` class named `s1`.
* Call the `setData` method on `s1` and pass `1`, `"Akansha"`, and `"Mca"` as arguments to set the student's roll number, name, and course, respectively.
* Call the `getData` method on `s1` to print out the student's information that was just set.

**Program:**

class *Stu*

{

  int Roll;

  String Name;

  String Course;

 public void *setData*(int Roll\_no, String Namee, String Crse)

 {

 Roll= Roll\_no;

 Name= Namee;

 Course= Crse;

 }

    public void *getData*()

    {

*System*.*out*.*println*("Roll number is "+ Roll);

*System*.*out*.*println*("Name is "+ Name);

*System*.*out*.*println*("Course is "+ Course);

    }

    public static void *main*(String args[])

    {

        Stu s1= new *Stu*();

*s1*.*setData*(1,"Akansha","Mca");

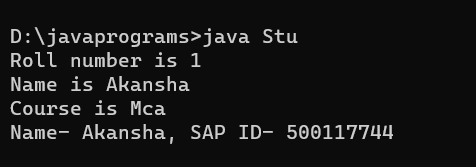
*s1*.*getData*();

*System*.*out*.*println*("Name- Akansha, SAP ID- 500117744");

    }

}

**Output:**

****

1. **Write software to prepare the results of the student for school examination. The Roll no of the students and their marks in subjects of Physics, Chemistry and Maths are supplied through the Scanner class. The result of the subject is prepared and displayed on the output screen by finding the students who fail and pass the exam. If the marks are below 40% in subject, then the student has failed in the subject. If the student passes and the aggregate score is above 75% then he scores Distinction. If the student passes and aggregate marks are 60%,50%, and 40% then he scores First, Second, and Third Division. Mention all the cases in output.**

**Algorithm:**

* Create a `Software` class with properties for roll number and marks in physics, maths, and chemistry.
* Include a `setData` method to input these details from the user.
* Add a `result` method to calculate the total and average marks, then determine the grade based on the average (Distinction, First division, Second division, Third division, or Failed).
* In the `main` method, create an instance of `Software`, call `setData` to collect data, then `result` to display the outcome.
* Print a statement with the name and SAP ID.

**Program:**

 import *java*.util.Scanner;

class *Software* {

    int Roll;

    int Marks\_phy;

    int Marks\_maths;

    int Marks\_chem;

    public void *setData*() {

        Scanner s = new *Scanner*(*System*.*in*);

*System*.*out*.*println*("Enter Roll number");

        Roll = *s*.*nextInt*();

*System*.*out*.*println*("Enter Physics marks");

        Marks\_phy = *s*.*nextInt*();

*System*.*out*.*println*("Enter Maths marks");

        Marks\_maths = *s*.*nextInt*();

*System*.*out*.*println*("Enter Chemistry marks");

        Marks\_chem = *s*.*nextInt*();

    }

    public void *result*() {

        int totalMarks = Marks\_phy + Marks\_maths + Marks\_chem;

        int avg = (totalMarks \* 100) / 300;

**if** (avg >= 75 && avg <= 100) {

*System*.*out*.*println*("Distinction");

        } **else** **if** (avg >= 60 && avg < 75) {

*System*.*out*.*println*("First division");

        } **else** **if** (avg >= 50 && avg < 60) {

*System*.*out*.*println*("Second division");

        } **else** **if** (avg >= 40 && avg < 50) {

*System*.*out*.*println*("Third division");

        } **else** **if** (avg < 40) {

*System*.*out*.*println*("You Failed");

        }

    }

    public static void *main*(String args[]) {

        Software so = new *Software*();

*so*.*setData*();

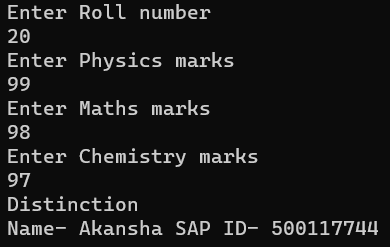
*so*.*result*();

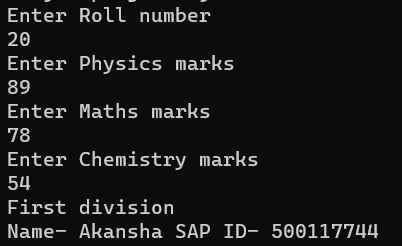
*System*.*out*.*println*("Name- Akansha SAP ID- 500117744");

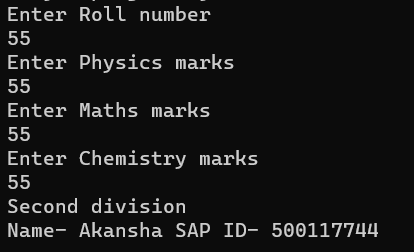
    }

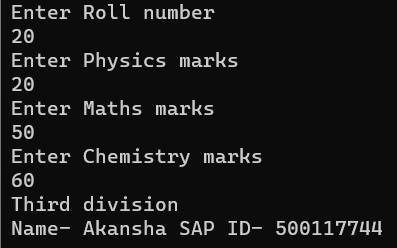
}

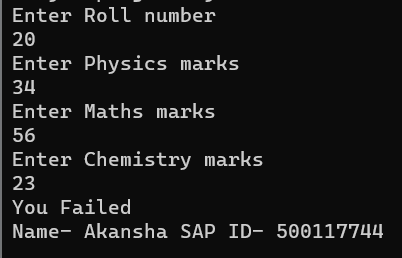
**Output:**











1. **Write a JAVA program to implement compile time polymorphism.**

**Algorithm:**

* Define a class named `OverLoad` that demonstrates method overloading by defining three versions of a method called `mthd`.
* The first version of `mthd` takes no arguments and simply prints the string "No argument" to the console.
* The second version of `mthd` takes a single integer argument and prints "one argument" to indicate it was called with one parameter.
* The third version of `mthd` accepts two integer arguments and prints "two argument" to show it was invoked with two parameters.
* Instantiate an object of the `OverLoad` class.
* Call the first version of `mthd` without any arguments to demonstrate the no-argument method call.
* Call the second version of `mthd`, passing in a single integer to demonstrate the method that requires one argument.
* Finally, call the third version of `mthd`, providing two integers to demonstrate the method that takes two arguments.

**Program:**

public class OverLoad {

public void mthd()

{

System.out.println("No argument");

}

public void mthd(int num)

{

System.out.println("one argument");

}

public void mthd(int num1 , int num2)

{

System.out.println("two argument");

}

public static void main(String[] args) {

OverLoad m = new OverLoad();

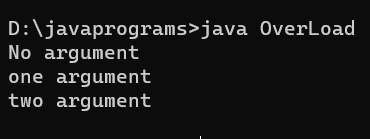
m.mthd();

m.mthd(0);

m.mthd(1, 2);

}}

**Output:**



1. **Write a JAVA program to implement type promotion in method overloading.**

**Algorithm:**

* Define a class named `TypePromotion`.
* Inside the class, define a method named `typePromo1` that accepts a single float parameter. When called, this method prints "Float promotion" to the console, indicating that a float type parameter is being used.
* Define another method within the same class named `typePromo2` that accepts a single double parameter. This method prints "Double promotion" when invoked, signaling the use of a double type parameter.
* Inside main function create an instance of the `TypePromotion` class.
* Call the `typePromo1` and ‘typePromo2` method.

**Program:**

public class TypePromotion {

public void typePromo1(float one)

{

System.out.println("Float promotion");

}

public void typePromo2(double one)

{

System.out.println("Double promotion");

}

public static void main(String[] args) {

TypePromotion type = new TypePromotion();

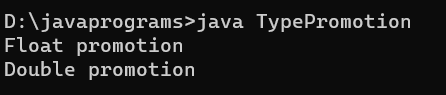
type.typePromo1(1);

type.typePromo2(15);

}

}

**Output:**

****

1. **Write a JAVA program to implement constructor overloading.**

**Algorithm:**

• Define a class named `Rectangle`.

• Add two instance variables, `length` and `breadth`, to hold the rectangle's dimensions.

• Construct a constructor for `Rectangle` that takes two parameters, `length` and `breadth`, for rectangular shapes.

• Include another constructor in the `Rectangle` class that accepts a single parameter, `side`, for square shapes.

• Implement a method named `getArea` to calculate and return the area of the rectangle or square. The area is found by multiplying `length` by `breadth`.

• In the main method, instantiate a `Rectangle` object named `rectangle1` with specific `length` and `breadth` to represent a rectangle. Print the area of `rectangle1` using the `getArea` method.

• Create another `Rectangle` object named `rectangle2` using the constructor designed for squares, passing a single value for `side`. Print the area of `rectangle2.

**Program:**

public class Rectangle {

double length;

double breadth;

public Rectangle(double length, double breadth) {

this.length = length;

this.breadth = breadth;

}

public Rectangle(double side) {

this.length = side;

this.breadth = side;

}

public double getArea() {

return length \* breadth;

}

public static void main(String[] args) {

Rectangle rectangle1 = new Rectangle(4.5, 8.0);

System.out.println("Area of rectangle1: " + rectangle1.getArea());

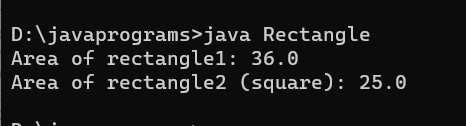
Rectangle rectangle2 = new Rectangle(5.0);

System.out.println("Area of rectangle2 (square): " + rectangle2.getArea());

}

}

**Output:**

****