GAURAV KISHOR PATIL

ROLL NO: 54

DIV:2 BATCH:C

Experiment No. 4
Method Overloading.
Date of Performance:25/08/2023
Date of Submission:29/08/2023

Aim:- To use concept of method overloading in class and object in java.

Objective:- To use concept of method overloading in a java program to create a class to calculate area of 3 geometrical figures using method overloading.

Theory:- Method Overloading is a feature that allows a class to have more than one method having the same name, if their argument lists are different. It is similar to constructor overloading in Java, that allows a class to have more than one constructor having different argument lists.

Example: This example to show how method overloading is done by having different number of parameters for the same method name.

```
Class DisplayOverloading
{
   public void disp(char c)
   {
       System.out.println(c);
   }
   public void disp(char c, int num)
   {
       System.out.println(c + " "+num);
   }
}
Class Sample
{
  Public static void main(String args[])
  {
     DisplayOverloading obj = new DisplayOverloading();
     Obj.disp('a');
     Obj.disp('a',10);
  }
}
```

Output:

```
A
A 10
```

Code:-

1)Method Overloading by Changing datatype of arguments

```
class Addition {
    int add(int x, int y) {
        return x + y;
    }

    double add(double x, double y) {
        return x + y;
    }
}

class OverloadingDemo {

    public static void main(String[] args) {
        Addition a1 = new Addition();
        int num1 = a1.add(20, 49);
        double num2 = a1.add(20.65, 49.69);
        System.out.println("Add1 when int=" + num1);
        System.out.println("Add2 when double=" + num2);

    }

}
```

```
Install the latest PowerShell for new features and improvements

PS G:\Programs\JAVA>  & 'C:\Program Files\Java\jdk-20\bin\java.
odeDetailsInExceptionMessages' '-cp' 'C:\Users\GAURAV\AppData\R
31ef07c5754485dab625cca33c18cc3\redhat.java\jdt_ws\JAVA_e16f3d6
Add1 when int=69
Add2 when double=70.34
PS G:\Programs\JAVA>
```

2) Method Overloading by Changing number of arguments

```
class Addition1 {
   int add(int x, int y) {
      return x + y;
   }

   int add(int x, int y, int z) {
      return x + y + z;
   }
}

class OverloadingDemo1 {

   public static void main(String[] args) {
      Addition1 a1 = new Addition1();
      int num1 = a1.add(20, 40);
      int num2 = a1.add(20, 40, 50);
      System.out.println("Add1 when 2 arguments=" + num1);
      System.out.println("Add2 when 3 arguments =" + num2);

   }

}
```

```
625cca33c18cc3\redhat.java\jdt_ws
Add1 when 2 arguments=60
Add2 when 3 arguments =110
PS G:\Programs\JAVA>
```

3) Finding area of semicircle rectangle and cylinder using MO

```
class area1 {
```

```
    Double area(double r) {
        return (Math.PI * r * r) / 2;
    }

    int area(int l, int b) {
        return l * b;

    }

    double area(double r, double h) {
        return 2 * r * Math.PI * (r + h);
    }
}

class mo3 {
    public static void main(String[] args) {
        area1 a1 = new area1();
        double d1, d3;
        int d2;
        d1 = a1.area(5.34);
        d2 = a1.area(8, 7);
        d3 = a1.area(2.3, 7.8);
        System.out.println("Area of semicircle: " + d1);
        System.out.println("Area of rectangle: " + d2);
        System.out.println("Area of cylinder: " + d3);
    }
}
```

```
Roaming\Code\User\workspaceStorage\a31ef0
jdt_ws\JAVA_e16f3d66\bin' 'mo3'
Area of semicircle: 44.79219973635255
Area of rectangle: 56
Area of cylinder: 145.95839468578177
PS G:\Programs\JAVA>
```

Conclusion:-

In Java, method overloading involves defining multiple methods in the same class with the same name but different parameter lists. This allows you to perform similar operations with varying input types or numbers of arguments. The conclusion is that method overloading enhances code clarity and reusability by providing a convenient way to handle different input scenarios using the same method name. It simplifies code maintenance and usage, making it easier to understand and work with classes that offer multiple ways of performing related tasks.