BCA – 401: Java Programming

Rahul Kumar Singh

In today's Class we have discussed on methods overloading and Constructors overloading in Java.

Methods (Function) overloading:-

When a class has two or more methods by the same name but different parameters, it is known as method overloading. It is different from overriding. In overriding, a method has the same method name, type, number of parameters, etc.

If a class has multiple methods having same name but different in parameters, it is known as Method Overloading.

Suppose you have to perform addition of the given numbers but there can be any number of arguments, if you write the method such as a(int,int) for two parameters, and b(int,int,int) for three parameters then it may be difficult for you as well as other programmers to understand the behavior of the method because its name differs.

So, we perform method overloading to figure out the program quickly.

Let’s consider the example of finding minimum numbers of integer type and double type. Then the concept of overloading will be introduced to create two or more methods with the same name but different parameters.

Example:-

```
public class ExampleOverloading
{
    // for integer
  public static int minFunction(int n1, int n2)
  {
    int min;
    if (n1 > n2)
      min = n2;
    else
      min = n1;
    return min;
  }
 // for double
  public static double minFunction(double n1, double n2)
  {
    double min;
    if (n1 > n2)
      min = n2;
    else
```

```
        min = n1;
    return min;
  }
  public static void main(String[] args)
  {
    int a = 11;
    int b = 6;
    double c = 7.3;
    double d = 9.4;
    int result1 = minFunction(a, b);
    // same function name with different parameters
    double result2 = minFunction(c, d);
    System.out.println("Minimum Value = " + result1);
    System.out.println("Minimum Value = " + result2);
  }
}
```

Output:-

```
Minimum Value = 6
Minimum Value = 7.3
```

Overloading methods makes program readable. Here, two methods are given by the same name but with different

parameters. The minimum number from integer and double types is the result.

Different ways to overload the method:-

There are two ways to overload the method in java.

- ➤ By changing number of arguments
- ➤ By changing the data type

Method Overloading by changing number of arguments:-

In this example, we have created two methods, first add() method performs addition of two numbers and second add method performs addition of three numbers.

In this example, we are creating static methods so that we don't need to create instance for calling methods.

Example:-

```
class Adder
{
static int add(int a,int b)
{
    return a+b;
}
static int add(int a,int b,int c)
```

```
{
    return a+b+c;
}
}
class TestOverloading1
{
public static void main(String[] args)
{
System.out.println(Adder.add(11,11));
System.out.println(Adder.add(11,11,11));
}
}
```

Output:

22

33

Method Overloading by changing data type of arguments-

In this example, we have created two methods that differs in data type. The first add method receives two integer arguments and second add method receives two double arguments.

Example:-

```
class Adder
{
static int add(int a, int b)
{
        return a+b;
}
static double add(double a, double b)
{
        return a+b;
}
}
class TestOverloading2
{
public static void main(String[] args)
{
System.out.println(Adder.add(11,11));
System.out.println(Adder.add(12.3,12.6));
}
}
```

Output:-

22

24.9

Constructor Overloading in Java:-

In Java, a constructor is just like a method but without return type. It can also be overloaded like Java methods.

Constructor overloading in Java is a technique of having more than one constructor with different parameter lists. They are arranged in a way that each constructor performs a different task. They are differentiated by the compiler by the number of parameters in the list and their types.

Example of Constructor Overloading:-

```
class Student
{
    int id;
    String name;
    int age;
    //creating two arg constructor
    Student(int i,String n)
   {
    id = i;
```

```
 name = n;
 }
 //creating three arg constructor
 Student(int i,String n,int a)
{
 id = i;
 name = n;
 age=a;
 }
 void display()
{
    System.out.println(id+" "+name+" "+age);
 }
 public static void main(String args[])
 {
 Student s1 = new Student(111,"Karan");
 Student s2 = new Student(222,"Aryan",25);
 s1.display();
 s2.display();
}
```

```
}
```

Output:

```
111 Karan 0
222 Aryan 25
```