Method Overloading in Java

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

Advantage of method overloading

Method overloading *increases the readability of the program*.

// Java program to demonstrate working of method overloading in Java

```
public class Sum
{
        public int sum(int x, int y)
            {
                return (x + y);
            }
        public int sum(int x, int y, int z)
        {
            return (x + y + z);
        }
        public double sum(double x, double y)
        {
            return (x + y);
        }
        public static void main(String args[])
        {
```

```
        Sum s = new Sum();

        System.out.println(s.sum(10, 20));

        System.out.println(s.sum(10, 20, 30));

        System.out.println(s.sum(10.5, 20.5));

    }

}
```

Different ways to overload the method

There are two ways to overload the method in java

1. By changing number of arguments
2. By changing the data type

1) Method Overloading: changing no. 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.

```
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));
}
}
```

2) Method Overloading: changing data type of arguments

```
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));
}
}
```

// Java Program to Illustrate Method Overloading By Changing Data Types of the Parameters

```
import java.io.*;
class Product
 {
        public int Prod(int a, int b, int c)
        {
                int prod1 = a * b * c;
                return prod1;
        }
        public double Prod(double a, double b, double c)
        {
                double prod2 = a * b * c;
                return prod2;
        }
}
class GFG
 {
        public static void main(String[] args)
        {
```

```
                Product obj = new Product();
                int prod1 = obj.Prod(1, 2, 3);
                System.out.println( "Product of the three integer value :" +prod1);
                double prod2 = obj.Prod(1.0, 2.0, 3.0);
                System.out.println( "Product of the three double value :" + prod2);
        }
}
```

Q) Why Method Overloading is not possible by changing the return type of method only?

In java, method overloading is not possible by changing the return type of the method only because of ambiguity. Let's see how ambiguity may occur:

```
class Adder
{
static int add(int a,int b)
{
return a+b;
}
static double add(int a,int b)
{
return a+b;
}
}
class TestOverloading3
{
```

```
public static void main(String[] args)
{
System.out.println(Adder.add(11,11));//ambiguity
}
}
```