What is a Method in Java ?
---
* A method is a self defined block which is used for writing the business logic, calculating the result, printing the data and so on
* Methods are divided into two types :

Note : We have also abstract method which we will learn later stage.

Example of static method :

```
public static void accept();
{

}
```

Example of non static method :

```
public void accept()
{

}
```

All possible combinations for writing a method in Java :

Case 1 : Method with no parameter and no return type :
Example :
```
public void accept()
{

}
```

Case 2 : Method with parameter and no return type :
Example :
```
public void accept(int num)
{

}
```

Case 3 : Method without parameter and with return type :
Example :
```
public double accept()
{

  return 0.0;
}
```

Case 4 : Method with Parameter and with return type
Example :
```
public int doSum(int x, int y)
{

  return x + y;
}
```

Definition of Method Overloading :

Allows us to write two or more methods in the same class or in super and sub classes such that method names are same, but they differ in their parameter list which are as follows :

a) Number of Parameter
b) Type of Parameter
c) Order of Parameter (Sequence of data type)

Example :
a) Number of Parameter
```
public void display(int a, int b) { }
public void display(int a, int b,int c ) {
```

b) Type of Parameter
```
public void show(int a) { }
public void show(double a) { }
```

c) Order of Parameter (Sequence of data type)
```
public void test(int a, double b) { }
public void test(double a, int b) { }
```

Basic Program on Method Overloading :

MOL.java [Based on Number of Parameters]

```
void main()
{
  IO.println("Hello main");
  accept(10,20);
  accept(10,20,30);
}

public void accept(int x, int y)
{
  IO.println("Sum of two integer is "+(x+y));
}

public void accept(int x , int y , int z)
{
  IO.println("Sum of three integer is "+(x+y+z));
}
```

MOL2.java [Based on Type of Parameters]

```
void main()
{
  accept(12D);
}

public void accept(long l)
{
  IO.println("Long Type");
}

public void accept(double d)
{
  IO.println("Double Type");
}
```

MOL3.java [Based on Order of Parameters]

```
void main()
{
  accept(11, 15L);
}

public void accept(long l, int x)
{
  IO.println("Long & int Corrida");
}

public void accept(int x, long y)
{
  IO.println("Int & long Caetro");
}
```

Rules for Overloading a method :
---
1. Methods must have the same name.
2. Can change the number of args, order of parameters.
3. Can change the return type.
4. Access modifiers can change.
5. Can throw different exceptions.
6. Type promotion applies automatically.
7. Cannot overload by only changing return type

WAP that describes while overloading a method we can change the return type of Method :

Note : **If any method return type is void then we cannot call that method directly from IO.println() statement because println() is a method present in the library which accept void as an parameter.**
```
void main()
{
  IO.println(m1()); //Error
}

public void m1() //return type is void
{
}
```

MOL4.java :

```
void main()
{
IO.println("Sum of two Integer is :"+accept(10,20));
IO.println("Sum of two Double is :"+accept(10.5,20.5));
}

public int accept(int x, int y) //return type is int
{
  return x+y;
}

public double accept(double x , double y) //return type is double
{
return x+y;
}
```