**Polymorphism in java** is a concept by which we can perform a *single action by different ways*. Eg.: going to Mumbai either by old highway or by new highway. Polymorphism is derived from 2 greek words: poly and morphs. The word "poly" means many and "morphs" means forms. So polymorphism means many forms.

* Static Polymorphism / Method **Overloading** / Compile time Polymorphism
* Dynamic Polymorphism / Method **Overriding** / Run time Polymorphism

Variations in **Overloading** a Method:

* Number of parameters passed
* Data type of parameters
* Sequence of data type of parameters
* **Ex:**
* void sum (int a , int b);
* void sum (int a , int b, int c);
* void sum (float a, double b);

**Method Overriding**

Method overriding is: when one of the methods in the super class is **redefined** in the sub-class. In this case, the signature of the method remains the same. **Return type** of overridden methods **can be different Ex:**

class X

{

public int sum()

{

// some code

}

}

class Y extends X

{

public int sum()

{

//overridden method

//signature is same

}

}

**IMP** : An overloaded method **may or may not have different return types**. But return type alone is not sufficient for the compiler to determine which method is to be executed at run time. Method Overloading means to have two or more methods with same name in the same class with different arguments.

# **What is method hiding in Java and how to use it?**

When super class and sub class contains same method including parameters and if they are static.

The method in the super class will be hidden by the one that is in the sub class. This mechanism is known as method hiding.

## Example

class Demo{

public static void demoMethod() {

System.out.println("method of super class");

}

}

public class Sample extends Demo {

public static void demoMethod() {

System.out.println("method of sub class");

}

public static void main(String args[] ) {

Sample.demoMethod();

}

}

## Output

method of sub class

**Effect of exception on overloading and overriding** :

 **If the superclass method does not declare an exception**

* If the superclass method does not declare an exception, subclass overridden method cannot declare the checked exception but it can declare unchecked exception.

 **If the superclass method declares an exception**

* If the superclass method declares an exception, subclass overridden method can declare same, subclass exception or no exception but cannot declare parent exception.