**Polymorphism is the ability of an object to take on many forms.**

* There are two types of polymorphism in java: compile time polymorphism and runtime polymorphism.
* We can perform polymorphism in java by method overloading and method overriding.

**Note:-**

* In compile time polymorphism method call is resolved in compile time.
* In runtime polymorphism method call is resolved in run time.

**Method OverLoading:-**

If a class have multiple methods by same name but difference in number of parameters,or type of parameters or order of parameters. it is known as Method Overloading.

class Calculation{

void sum(int a,int b){System.out.println(a+b);}

void sum(int a,int b,int c){System.out.println(a+b+c);}

public static void main(String args[]){

Calculation obj=new Calculation();

obj.sum(10,10,10);

obj.sum(20,20);

}

}

**Note:-**

In method overloading the methods should have same name, but return type of those methods need not to be same.

**Method Overriding:-**

* If subclass (child class) has the same method as declared in the parent class, it is known as method overriding in java.
* Method overriding is used to provide specific implementation of a method that is already provided by its super class.

**Rules of method overloading:-**

The signature of the sub class method should be same as the parent class, here the signature for a method is nothing but name,parameters, return type, and checked exception.

* The access level cannot be more restrictive than the overridden method's access level. **For** **example**: if the superclass method is declared public then the overridding method in the sub class cannot be either private or protected.
* Constructors cannot be overridden.
* An overriding method can throw any uncheck exceptions, regardless of whether the overridden method throws exceptions or not. However the overriding method should not throw checked exceptions that are new or broader than the ones declared by the overridden method. The overriding method can throw narrower or fewer exceptions than the overridden method.
* **A method declared final cannot be overridden.**