Method Over loading:

Why method overloading come into picture?

Let’s Suppose we have addition operation

One time we might have to add two int numbers.

One time we might have to add three int numbers

One time we might have to add two int numbers and one float number.

....................................................so on other types

In that case if we write different method name for each method it would be confusing rather, we can specify a same method name to all add operations with change in the parameters and return type.

int add(int a,int b)

{

return a+b;

}

int add(int a,int b,int c)

{

return a+b+c;

}

float add(int a,float b)

{

return a+b;

}

In the above format we can see same method name but varying the parameters which will allow the developer to net get Confused with different method names.

<https://onlinegdb.com/ARVg6Rnqa>

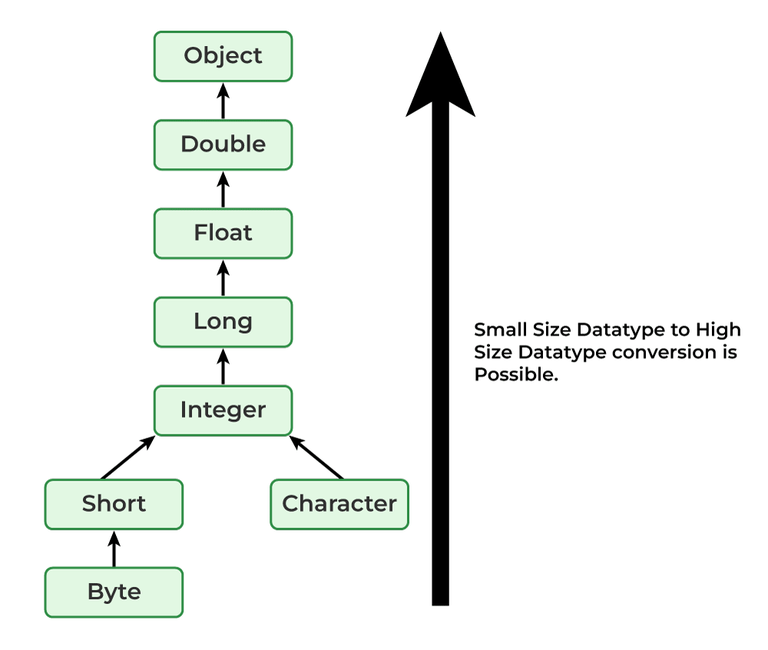
//Method Overloading:

Multiple Methods with same name but differ in the number/type/order of parameters

## **Different Ways of Method Overloading in Java**

* Changing the Number of Parameters.
* Changing Data Types of Arguments.
* Changing the Order of the Parameters of Methods

Method overloading in Java is also known as Compile-time Polymorphism, Static Polymorphism, or Early binding.



//In case of Method Overloading Compiler will resolve the issue which method that has to called based on the Number of parameters

Type of parameters

Order of Parameters

We called the above Compile time Polymorphism.

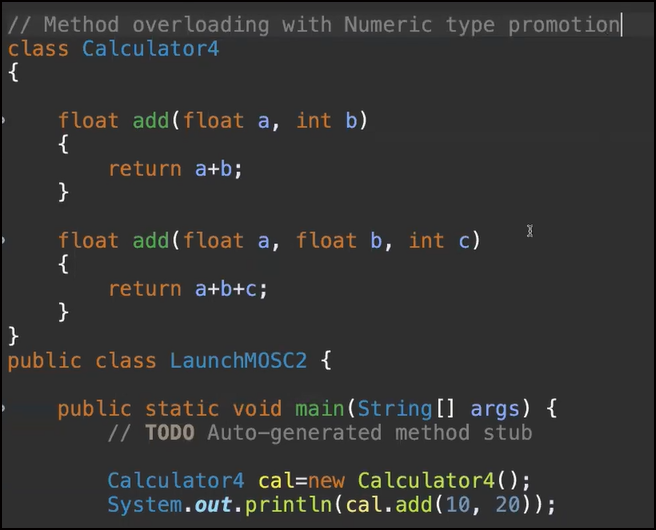
Polymorphism: 1 to Many -- One method multiple Operations. --But in reality, for performing each operation we will have different method.

println()---Is also example for in-built Method Overloading

Compile-time polymorphism is also known as static polymorphism or early binding. Compile-time polymorphism is resolved during the compilation process. Overloading of methods is called through the reference variable of a class.

The compiler will resolve the issue of which method must be called.

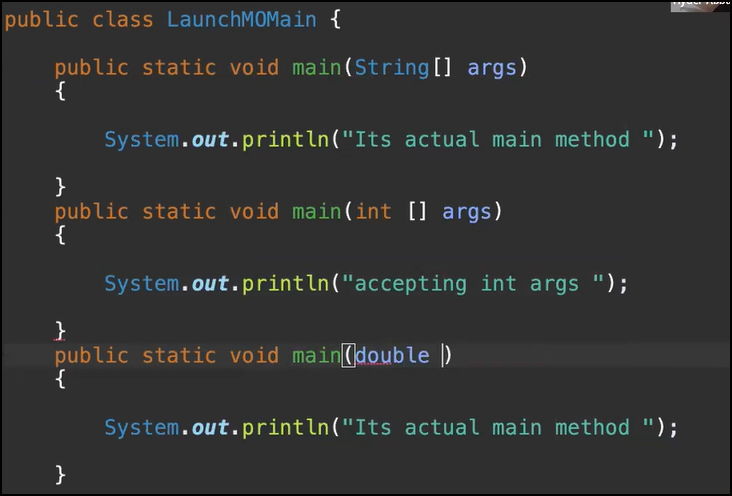
False Polymorphism –Here there will multiple methods for doing each of the operation



Can we Overload main method ?:

Yes, we can overload the main () method.

But JVM will invoke such a main method that has (String args[]) as main method.



If we write the same method twice it will result in compile time Error.

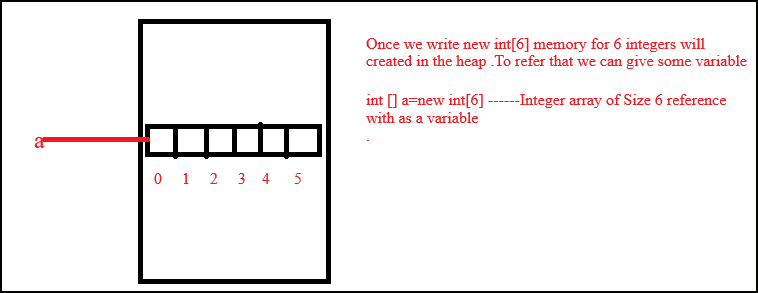
Array :

Array :

* Indexed based data structure to store large volume of data.
* Homogenous data
* Arrays in Java are treated as Objects—Memory will be allocated in Heap

new int [5]--

int [] a=new int [5]



a is variable refereeing to array of integers.

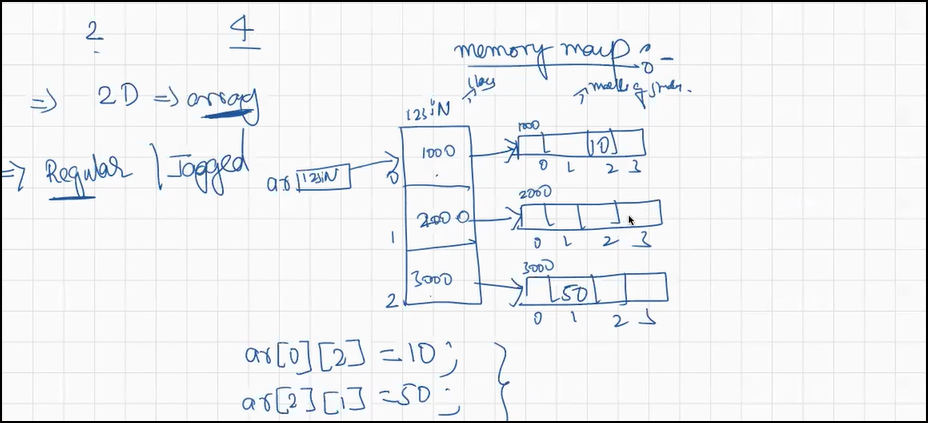
For adding data:

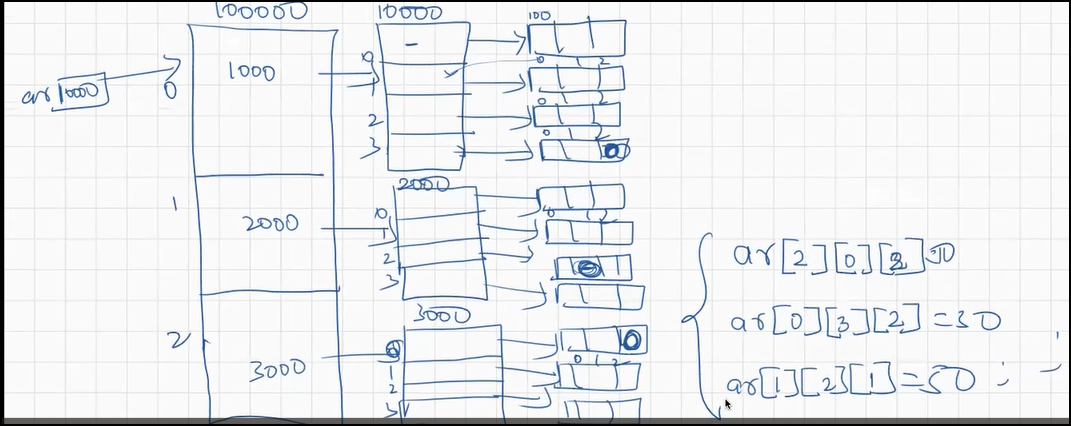
a[0]=10;

a[4]=20;

1D Array – int [] marks = new int[5]

2D Array – int [][] marks = new int[3][4]





3D Array :

**Regular Array:**

When all the internal dimensions are same then we can call that as a Regular Array.

int [][][]a=new int [2][3][5]

Jagged Array:

When all the internal dimensions are not the same then we can call that a Jagged Array.

int [] [] [] b=new int [2][3] [] -- We have not mentioned third as the number of elements varies.

Consider a Scenario when a School has 3 class and different number of students(5,3,4) in each class then we cannot use regular array.

int a[][]=new int[3][] ///number of students column is left empty as number of student in each class varied.

a[0]=new int [5] ----Refers Class1 which can hold 5 students data

a[1]=new int[3]

a[2]=new int[4]

3-D Jagged Array: 3 Colleges but different number of Classes and students in each.

int ar[][][]=new int[3][][]

