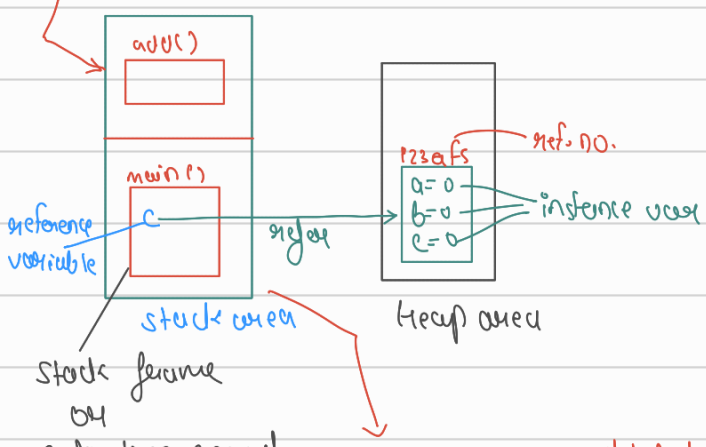
# Method & Accuracy

# Method

21st oct 2022

```
class Calc
{   int a; int b; int c;
    void add() { }
}
public class Launch
{
    Calc c = new Calc();
    c.add();
}
```



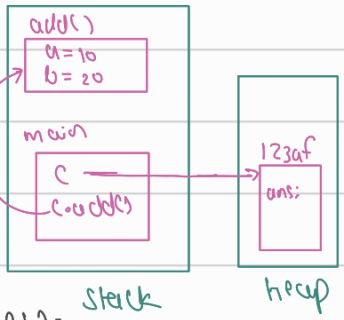

both record will be deleted, after execution, Garbage collector check if there is any object in heap area w/o any reference in stack area, if found it'll be deleted by G.C.

→
```
class Calc
{   int ans;

    void add(int a, int b) // taker : parameter
    {
        ans = a+b;
        sop(ans);
    }
}

public class Launch
{
    main()
    {   Calc c = new Calc();
        c.add(10,20); // given : argument
    }
}
```

local var




# Method Overloading : / Compile-Time Polymorphism

- Refers to the process of writing more than one method with same name & diff parameters within same class.
- Early Binding / Compile time polymorphism
- Compiler resolve the conflict

a) No. of parameters  b) Data type of parameters  c) order of DT of parameters

→ Main method can be overloaded.

```
// return type has no role to play, Its only method name and parameters
class Calculator3
{
    int add(int a, int b)
    {
        return a+b;
    }

    void add(int a, int b)
    {
        int res=a+b;
        System.out.println(res);
    }

    float add(int a, float b)
    {
        return a+b;
    }

}
public class LaunchMOCS {

    Run | Debug
    public static void main(String[] args) {
        // TODO Auto-generated method stub


            Calculator3 cal=new Calculator3();
            cal.add(a: 10,b: 20);
    }
```

Same name error

Can we overload main method ? Yes

```java
// Can we overload main method in Java?

public class LaunchMOMain {

    Run | Debug
    public static void main(String[] args) {

        System.out.println(x: "Its actual main method ");

    }

    public static void main(int[] args) {

        System.out.println(x: "accepting int args ");

    }

    public static void main(double b) {

        System.out.println(x: " double value");

    }
```

O/P, as JVM only calls String[] args main only.

```java
class Calculator5
{

    float add(float a, int b)
    {
        return a+b;
    }

    float add(int c, float a)
    {
        return a+c;
    }
}
public class LaunchMOCS3 {

    Run | Debug
    public static void main(String[] args) {
        // TODO Auto-generated method stub
    Calculator5 cal=new Calculator5();
    cal.add(a: 10, b: 20);
    }

}
```

Ambiguity

// error

```java
// Method overloading with Numeric type promotion
class Calculator4 {

    float add(float a, int b) {
        return a + b;
    }

    float add(float a, float b, int c) {
        return a + b + c;
    }
}

public class LaunchMOSC2 {

    Run | Debug
    public static void main(String[] args) {
        // TODO Auto-generated method stub

        Calculator4 cal = new Calculator4();
        System.out.println(cal.add(a: 10, b: 20));

    }
```

implicit type casting

# Arrays:

Problem: many variables with diff names ---

→ It is Index based DS to store large vol of data using single name (var-name).

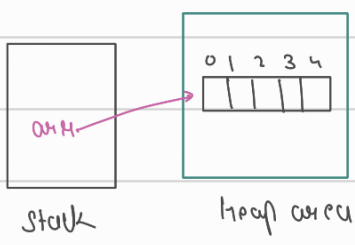
→ Homogeneous

→ objects → Heap

→ JVM treat array as object that is why we have to create with new int[5];

1-D Array:

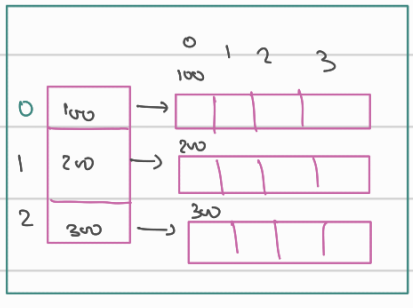
```
int [] arr = new int[5];
```




2-D array:

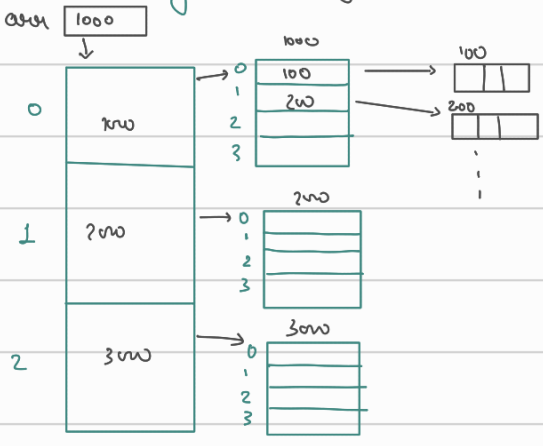
```
int [][] arr = new int[3] [4];
```

Regular | jagged




Heap

3D-Array: 3 colleges, 4 class, & each class 3 students



## 3-D Jagged Array:

| college | class | students |
|---|---|---|
| 0 | 0 | 4 |
| | 1 | 2 |
| | 2 | |
| 1 | 0 | 3 |
| | 1 | 1 |
| | 2 | 5 |
| | 3 | |
| 2 | 0 | 3 |
| | 1 | 4 |
| | 2 | 7 |
| | 3 | 5 |
| | 4 | |

```
int[][][] arr = new int[3][][];   // variable size changing

arr[0] = new int[2][];
arr[1] = new int[4][];
arr[2] = new int[3][];

a[0][0] = new int[4];
arr[0][1] = new int[2];

a[1][0] = new int[3];
a[1][1] = new int[1];
a[1][2] = new int[5];
 :    :    :    :
```

→ `int[] arr = new int[5];`

`arr.length;` → 5

→ `Scanner sc = new Scanner(System.in);`

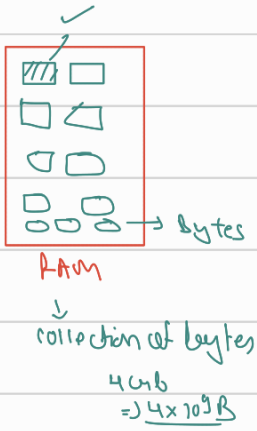
→ `int[][] arr = new int[3][4];`

```
for(int i=0 ; i<arr.length; i++)
{ for(int j=0 ; j<arr[i].length ; j++)
  {   sop("");
      arr[i][j] = sc.nextInt();
  }
}
```

## Buffer Overrun:

`int[3];` → 3 × 4B = 12 Bytes

```
a[0] = 10;
a[1] = 20;
a[2] = 30;
a[3] = 40; // Buffer Overrun
```




In java: there is no Buffer Run

arrays in java are guarded with boundary

i.e. (objects) → m/o allocation.

`a[3] = 40;` // exception (Index out of bound)

→ Drawbacks of Array

→ It can store only homogenous data
→ m/o of arr is of fixed size
→ Contiguous m/o allocations
→ It's not a collection class so, no inbuilt methods.

```
class Stud { int marks;
             String name;
             int rollNo;
}

main
{ Stud[] s = new Stud[5];
  s[0].marks = 10;
  s[0].name = "yam";
  s[1].mark = 20;
}
```

→ We can create array of Any User defined Class.

## For-Each loop: (Advance for loop)

```
int a[] = {10,20,30};
```

31 oct 2022

int → ele, a → arr name

```
for (DataType var name : a)
{ sop(ele); }
```

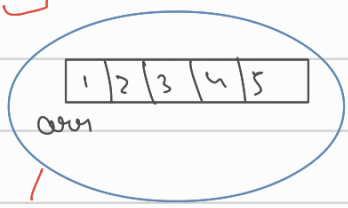
→ Array is Treated as Object

```
int[] ar = new int[5];
```

int[] → primitive

int → Byte, short, int, char

× long, float, double, boolean




object

→

```
int[][] ar = {{1,2,3},{1,2,3},
              {1,2,3,4}};
```

```
for(int[] row : ar)
{ for (int elem : row)
  { sop(elem); }
}
```

elem: 1 2 3 4

adv: we don't need to specify start pt & end pt.

Limitation:

→ can't access index

→ Traverse/Iterate → only in forward direction

→

```
int[] a = new int['a'];
```

'a' → 97

int ← char implicit

→ If I'm able to create object, then some class is there...

```
int[] a = new int[5];
```

int[] → some class is there, but it is only for JVM internal purpose.

Q. WAP to Access alternate value in an Array.

collection API → for each / Iterator

```
int[][] a;
int a[][];
int[]a[];
int[] a,b;
a = new int[5];
b = new int[6];
```

```
int[] x[],y;
int[][] xy[],y2;
```

(2D, 1D; 3D, 2D)

```
int[]a, []b; ✗ CE
int[]a, b[][];
```

int[] → [I → 3-proxy class which would be useful for JVM at runtime.

int[][] → [[I

char[] → [C no .class file is avail like Scanner.class, System.class, String.class...

double[] → [D

```
sop(a.getClass().getName());
```

→ we can't use [[I.sout → properties.

→ for using properties/methods we have:

Arrays class

call methods inside this class is static, so

- sort
- fill
- BinarySearch

no need to create object, just call using class name (Arrays)

# Array Exception:

1 nov 2022

1. Array Index Out Of Bounds / Runtime

```
int [] a = new int [2];
a[3] = 10;
```
(2 ✗)

2. Negative Array Size Exception:

```
int [] a = new int [-2]; // no CE
```
(-2: Byte, int, char, short) — Error as it is int

3. Array Store Exception:

```
int [] a = new int [2];
a[0] = 10;
a[1] = "yash"; // CE
```

4. Out Of Memory:

```
int [] a = new int [long];
```
long: out of range

Q. WAP to sum of elements in an Array.

Q. WAP to get min & max elem in an Array.
↳ traversing → loops
conditions → if-else

Q. WAP Linear search.

Q. WAP Binary Search.

```
                0 1 2 3 4 5 6
→ int [] a = {1,2,0,0,0,4,5}
Arrays.fill(ar, 2, 5, 10);   (values, idx)
o/p {1,2,10,10,10,4,5}
```

utility class

```
int res = Arrays.binarySearch(arr, 10);
```

return idx: if found

return -(idx+1); if not found

where it should be if found

## ⇒ Sorting:

- → Bubble sort
- → Selection sort
- → Insertion sort
- → Merge sort
- → Quick sort.

Q. Project Question Game.

Entrance

→ Guess the no. b/w 1 to 10