Introduction to Arrays & Arraylist in Java

Why do we need arrays?

⇒ It was simple when we had to store just five integer numbers and now lets assume we have to store 5000 integer numbers. Is is possible to use 5000 variable? NO

To handle these situations, in almost all programming language we have a concept called Array.

⇒ Array is a data structure use to store a collection of data

⇒ Syntax of an Array:

datatype [ ] variable name = new datatype [size];

Eg:- We want to store roll numbers

→ Represent the type of data stored in array → Store 5 roll numbers

int [ ] rollnos = new int [5]

(or)

int [ ] rollnos = { 51, 82, 35, 27, 16 }

All the type of data in array should be same

⇒ Internal working of array :-

int [ ] rollnos; // declaration of array

→ rollnos are getting defined in stack

rollnos = new int [5]; // initialisation

→ actual memory allocation happens here

Here, object is being created in heap memory

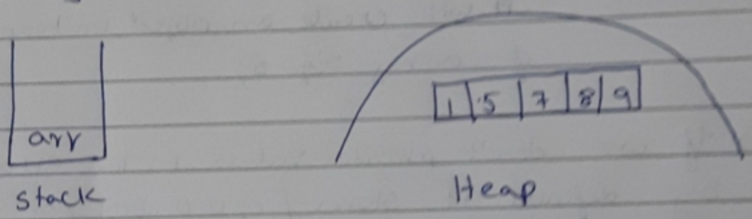
declaration of array
Compile time

initialisation
Runtime

int [] arr = new int [5] ;

↑ datatype ↑ Reff Var ↑ Creating Object in heap memory

=> This above Concept is Known as Dynamic memory allocation which means at runtime (or) execution time memory is allocated.



=> Internal representation of Array :-

- Internally in java, memory allocation totally depends on JVM wheather it be continous or not!

Reason 1 :- Objects are stored in heap memory

Reason 2 :- In JLS (Java language Specifications) it is mentioned that heap objects are not continous

Reason 3 :- Dynamic memory allocation, Hence, array objects in Java may not be continous (depends on Java)

=> Index of a array :

arr →	3	8	9	10	53	33
index	0	1	2	3	4	5

arr [0] = 3
arr [1] = 8
arr [5] = 33

Suppose to change the Value of Certain index
arr [4] = 99

New array will be

3	8	9	10	99	33
0	1	2	3	4	5

=> new keyword :-

Used to creat an object

int [] arr = new int [5];

it will create an object in heap memory of array size 5;

=> If we don't provide values in the array, internally by default it stores [0, 0, 0, 0, 0] for above size of array

* Primitives (int, char etc) are stored in stack

* All other objects are stored in heap memory

=> Arrays.toString (array) -> internally uses for loop and gives the output in proper format

* In an array, since we can change the objects, hence they are mutable

* strings are immutable

=> 2D Array :-

$$\begin{array}{c|ccc} & & 3 & \\ \hline 3 & 1 & 2 & 3 \\ & 4 & 5 & 6 \\ & 7 & 8 & 8 \end{array}$$

row column

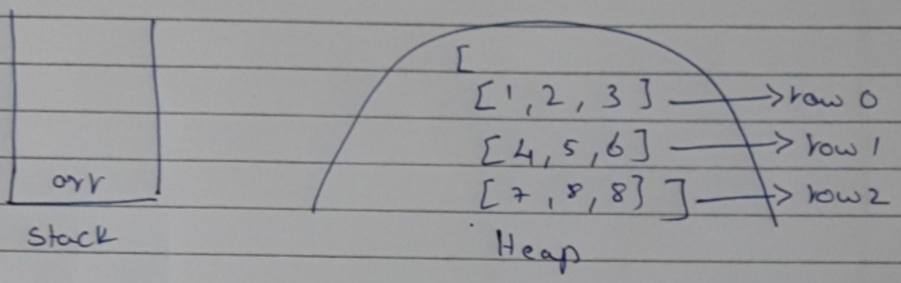
=> int [] [] arr = new int [size] [ ]

mandatory to give size of array

not mandatory to specify the size

(or)

```
int [ ] [ ] arr = {
            {1, 2, 3},
            {4, 5, 6},
            {7, 8, 8}
        }
```



array of arrays

arr[0] = [1, 2, 3]
arr[0][2] = 3

=> Array Lists :

Array list is a part of Collection framework and is present in Java.utils.package. It provides us with dynamic arrays in Java. It is slower than standard arrays

Syntax :

```
ArrayList <Integer> List = new Arraylist <>();
              └> add wrapers
```

=> Internal working of ArrayList :-

- Size is fixed internally
- Suppose arraylist is gets filled by some amount
  (a) it will make an arraylist of say double the size of arraylist initially
  (b) old elements are copied in the new arraylist
  (c) old ones are deleted