ARRAY

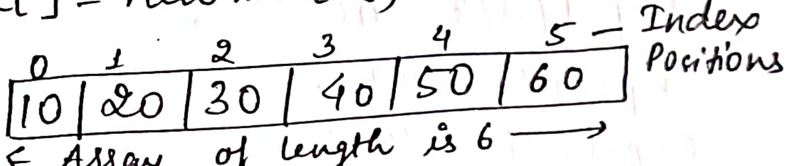
→ An array is an object that holds a fixed number of values of homogeneous or similar data-type.

→ Or say An Array is a Data Structures where we store similar elements.

→ The length of an array is assigned when the array is created and After creation, its length is fixed. → Stored in Heap Memory due to object.

→ For example : int a[]= new int[6];

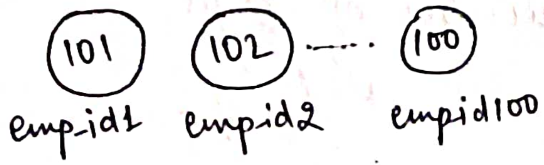
It will create an array of length 6 and index value will always start from 0.



→ Syntax : datatype []var_name ; // int ~~a[]~~ [] a;
or datatype var_name[]; // int a[]

```
int empid1 = 101;
int empid2 = 102;
int empid3 = 103;
    :
int empid100 = 100;
```

This takes 100 times memory occupy. so that your project is going to be slow.

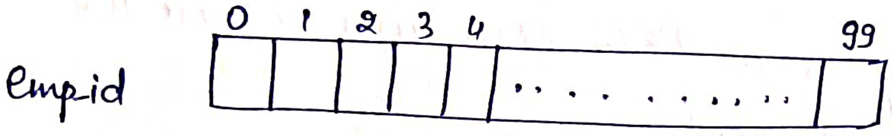


After that Array Concept is introduced :

or int[] empid;

int empid[]; // declares an array of Integer

empid = new int[100]; // allocates memory for 100 integer

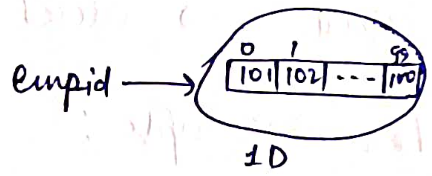


OR

int empid[] = new int[100]; // combining both statements in one

Now, we initialize 1st element .....

empid[0] = 101;
empid[1] = 102;
empid[2] = 103;
empid[3] = 104;
⋮
empid[99] = 1000;



← S.O.P. (~~emp~~[3]); (empid[3]);

S.O.P. ("Element at index 0:" + empid[0]);
S.O.P. (" Element at index 1:" + empid[1]);
⋮
S.O.P. ("Element at index 99:" + empid[99];

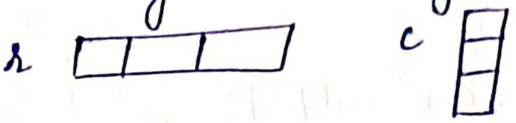
OR

int[] empid = {101, 102, 103, 104, 105, 106 ..... 1000};

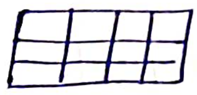
→ length is determined by number of values provided b/w braces { } & separated by commas (,).

Types of Arrays

① Single Dimensional → Single row / single column
↳ 1D



② Multi Dimensional → multiple rows / multiple columns (Array of Arrays)
↳ 2D
↳ 3D



1 Dimensional Arrays

At the time of Declaration, we cannot define size like as

```
int[3] empid;  }  Both are
String[3] a;   }  invalid
```

① Declare — `int[] empid;`
`empid = new int[5];` // creation & allocating

② Declare the size at the time of creation
OR (combination of both declare & creation)

$\boxed{\texttt{int[] emp-id = new int[5];}}$ 1D

③ `int[] empid = new int[5];` → compile ✓
run ✓

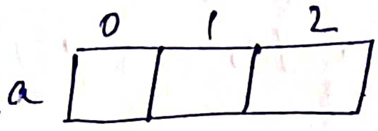
But

`int[] empid = new int[-5];` → compile ✓
run ✗ (exceptions throws)

$\boxed{\text{NegativeArraySizeException}}$ → Throws

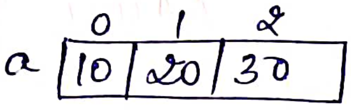
Eg:

```
int[] a;
a = new int[3];
    OR
int[] a = new int[3];
```



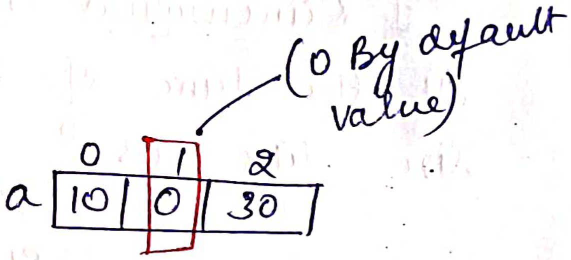
Initialize elements

```
→ a[0] = 10;
  a[1] = 20;
  a[2] = 30;
```



By default

a[1] = ? (Not Given) then



→ $\boxed{a[3] = 40;}$ Exception Throws i.e. ArrayIndexOutofBoundException.

Eg: for String Array

String strArray[] = { "Python", "JAVA", "C", "C++", "PHP" }

Multi Dimensional Array

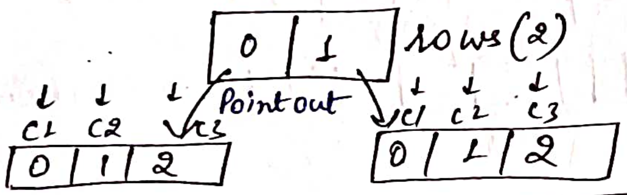
2D Array

```
int[][] a;            // declare
a = new int[][];      // creation
        Or
int[][] a = new int[2][3]
    2 rows , 3 columns
```

	c1	c2	c3
r1	0,0	0,1	0,2
r2	1,0	1,1	1,2

m x n

⇓ represents

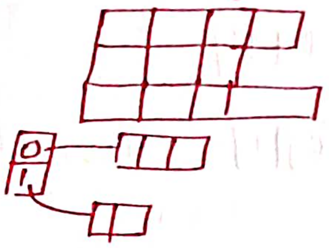


col 3

```
int a[2][3]
  or
int a[2][3] = {{1,2,3},{4,5,6}};
      ↑  ↑
     row col
```

① MATRIX ARRAY
(3,3)

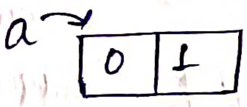
② JAGGED ARRAY
(not same r,c)



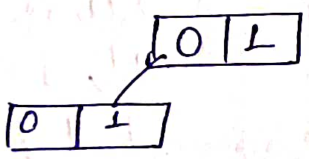
3D Array

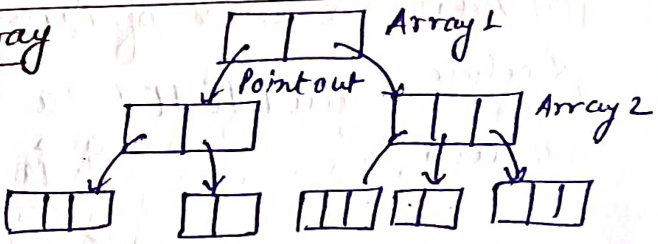
3D, Array is an array of 2D Arrays

```
int[][][] a;
a = new int[2][][];
```



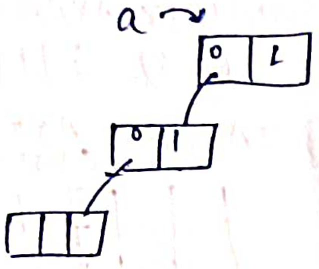
```
a[0]= new int[2][];
```



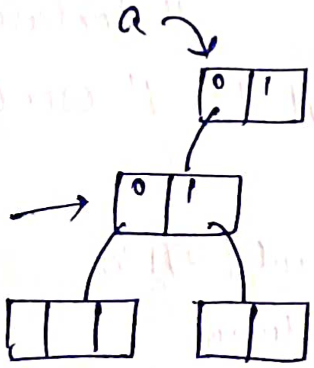


③ Anonymous Array

a[0][0] = new int[3];

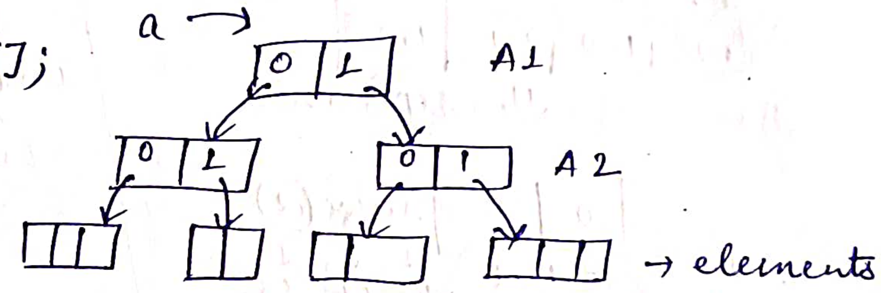


a[0][1] = new int[2];



Same as at array 1th Positions

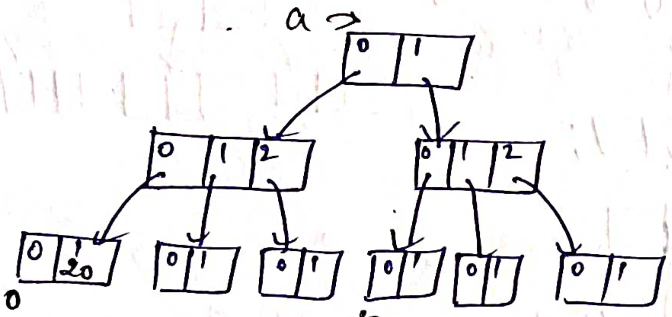
a[1] = new int[2][];
a[1][0] = new int[2];
a[1][1] = new int[3];

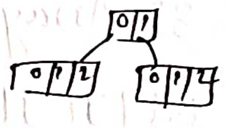


① Adv & DisAdv of Arrays.

Declare & Creation

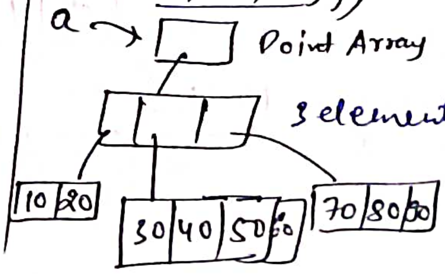
int[][][] a = new int[2][3][2];





a[0][0] = 10; ✗
CompileTime Error

int[][][] a = {{{10,20}},
{{30,40,50,60}},
{{70,80,90}}}; //d,c,i



Initialization

a[0][0][0] = 10;
a[0][0][1] = 20;
a[1][0][0] = 30;