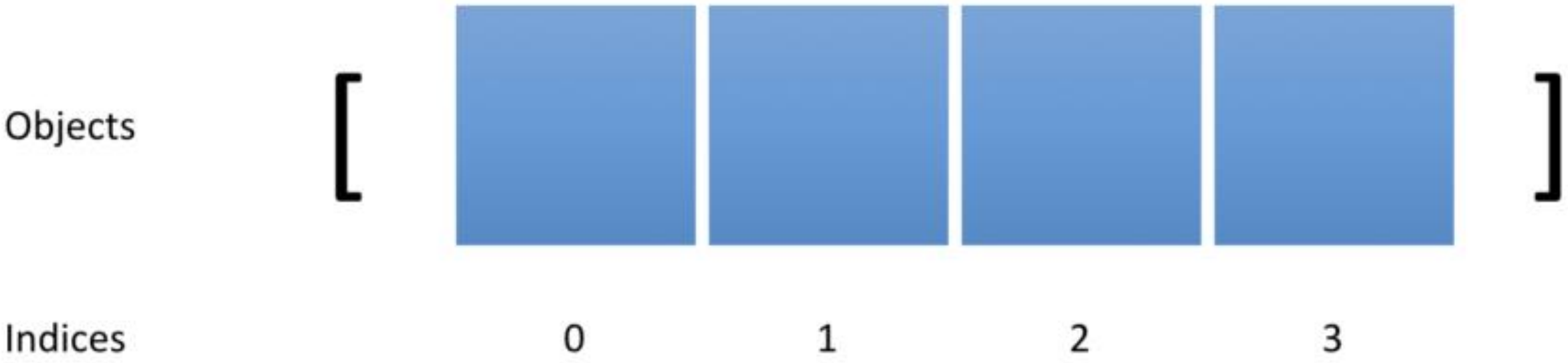
ARRAYS



IN THE NAME OF ALLAH

بسم الله الرحمن الرحيم

THE GRACIOUS, THE MERCIFUL.

GETS - GETTING EDUCATION WITH TECHNOLOGICAL SYSTEM

Instructor: Sir A.Rehman Ali Brohi

LECTURE: 11
Arrays

1. **INTRODUCTION TO ARRAY**
   - What is Array?
   - Features of Array
   - Advantages of Array.
   - Disadvantages of Array.

2. **TYPES OF ARRAY**
   - Single Dimensional Array
     1D Array
   - Multidimensional Array
     2D Array
     3D Array

3. **Array Declaration, Creation, Initialization**

4. Retrieve elements from an array by using for and for-each loop

5. Arrays of Objects

6. Get Class name for an array

7. What are Matrix and Jagged Arrays

8. Anonymous Arrays

9. Array Program(Sorting, Searching, Merging, Deleting etc)

WHAT IS ARRAY?

- An Array is used to store multiple variables.
- Array is the collection of Homogeneous (or Similar) data types.

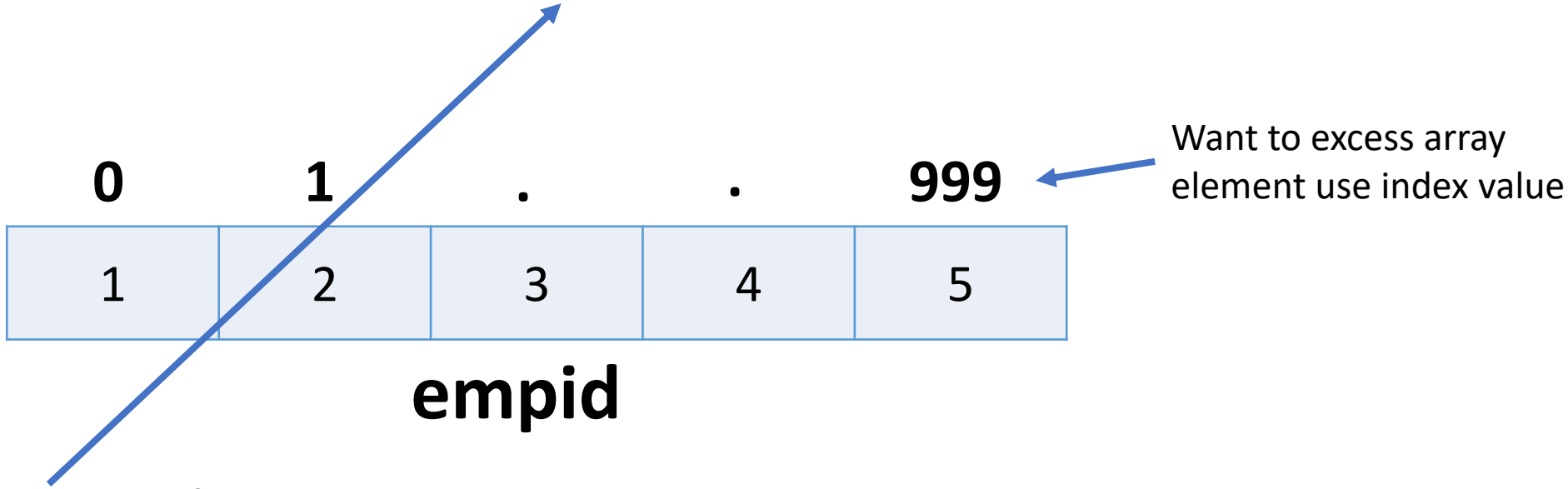
Lets Suppose we have 1000 int Variable for employee like

```
int empid = 1;
int empid = 2;
int empid = 3;
int empid = 4;
.
.
.
int empid = 1000;
```

Then our program will become heavy and performance will be slow

What if we will create an array and assign it 1000 data within it

int empid[] = new int[1000];



1. **In java Array is Object | then the superclass of predefine class of array is object**
2. **Array occupied memory from HEAP**
3. **In Java all object stored at HEAP memory.**
4. **So array is also object then array is also stored at HEAP memory**

What is Array ?

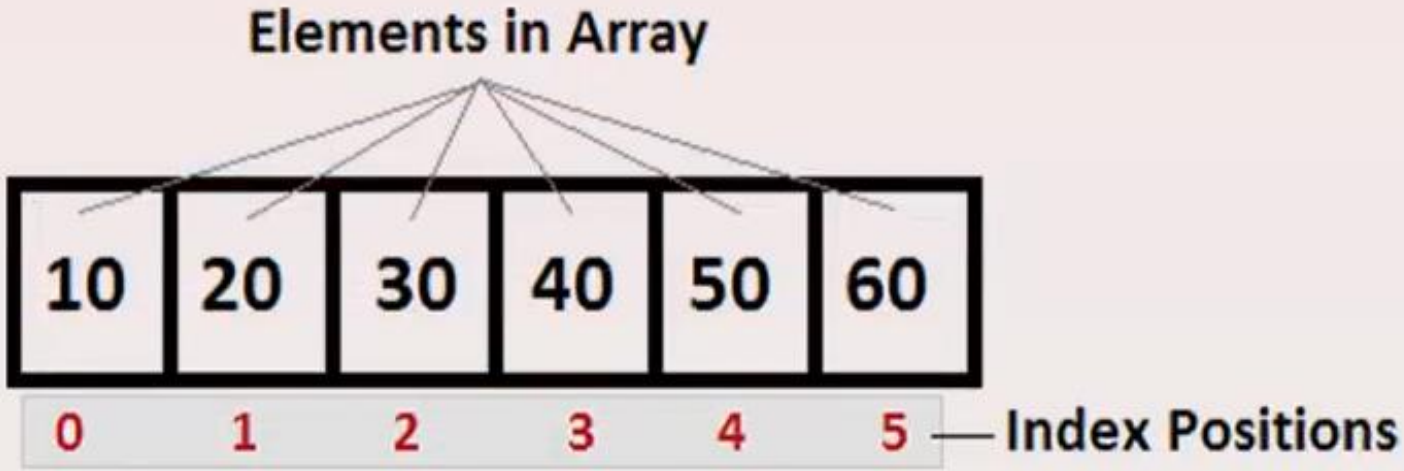
- An array is an object that holds a fixed number of values of homogeneous or similar data-type.

- Or say An Array is a Data Structure where we store similar elements.

- The length of an array is assigned when the array is created and After creation, its length is fixed.

- For example : int a[ ]=new int[6];

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



Features Of An Array :

- A Java array variable can be declared like other variables with [ ] after the data type.
- The variables in the array are ordered and each have an index beginning from 0.
- In Java, Arrays are objects, and thus they occupy memory in 'Heap Area'.
- The direct superclass of an array type is Object.
- They are always created at runtime.
- The length of an array can be find by using member 'length'.
  This is different from C/C++ where we find length using sizeof.
- The elements of array are stored in consecutive memory locations.

Advantages Of An Array :

- Arrays are used to store multiple data items of same type by using only single name.
- We can access any element randomly by using indexes provided by arrays.
- Arrays can be used to implement other data structures like linked lists, stacks, queues, trees, graphs etc.
- Primitive type to wrapper classes object conversion will not happen so it is fast.

Disadvantages Of An Array :

- **Fixed Size :** We need to mention the size of the array, thus they have fixed size. When array is created, size cannot be changed.

- **Memory Wastage :** There is a lot of chance of memory wastage. Suppose we creat an array of length 100 but only 10 elements are inserted, then 90 blocks are empty and thus memory wasted.

- **Strongly Typed :** Array stores only similar data type, thus strongly typed.

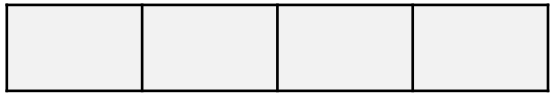
- **Reduce Performance :** The elements of array are stored in consecutive memory locations, thus to delete an element in an array we need to traverse through out the array so this will reduce performance.

- **No Methods :** Arrays does not have add or remove methods.

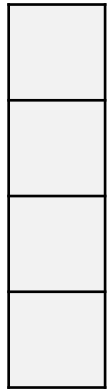
2. TYPES OF ARRAYS

Single Dimensional Array

1D Array



Single Row

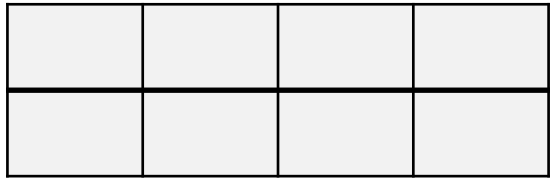


Single Column

Multi Dimensional Array or (Array or Arrays)

2D Array

3D Array



Multiple Rows or Multiple Column

1D Array

DECLARATION >

CREATION

INITIALIZATION

We normally declare variable like **int a;** or **String a;**
but in array we use [] after the data type.

Data Type

Variable Name

```
int[]  a;
int  []a;
int  a[];
int[]a;
```

```
String[] a;
String []a;
String a[];
String[]a;
```

Most **Prefer** used of array

DECLARATION >

CREATION

INITIALIZATION

We will **never declare size during declaration**

int[3] a;

X

String[3] a;

X

Differences

int[] a,b; //a and b both are array.

int []a,b; //a and b both are array.

int a[],b; //a is array b is only variable.

DECLARATION >

CREATION

INITIALIZATION

1. int[] a; //rule to declare

2. int[] a; //prefer
   int []a;…

3. int[3] a; //do not declare size during declaration.
   String[3] a;

4. int[] a,b; //both are array.
   int a[],b; //a is array b is variable.

DECLARATION

CREATION >

INITIALIZATION

- int[] a; //rule to declare
- **a = new int[3];** //rule of creation
- **Declare the size at the time of creation is compulsory.**
- **int[] a = new int[3];**

  Declaration & Creation in single line

DECLARATION

CREATION >

INITIALIZATION

- int[] a = new int[0]; //compile & run successfully

- int[] a = new int[-3]; //compile but not execute
  it will display error
  NegativeArraySizeException

DECLARATION

CREATION

INITIALIZATION >

- int[] a; //Declaration
- a = new int[3]; //Creation

- Int[] a = new int[3]; //Declaration + Creation

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

0	1	2
10	**20**	**30**

a

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

Declaration, Creation and Initialization in single line

int[] a=new int[]{10,20,30}; //also can

Retrieve an Array by using for loop.

```java
package com.company;
public class Main {
    public static void main(String[] args) {
        int[] a = {10,20,30};

        for(int i=0; i<a.length; i++){
            System.out.println(a[i]);
        }
    }
}
```

```
"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" "-javaagent:C:\Program Fi
10
20
30
```

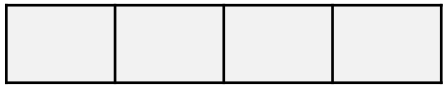
Retrieve an Array by using for-each loop.

```java
package com.company;
public class Main {
    public static void main(String[] args) {
        int[] a = {10,20,30};

        for (int i: a) {
            System.out.println(i);
        }
    }
}
```

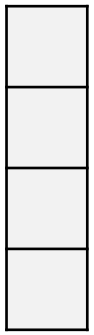
```
"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" "-javaagent:C:\Program File
10
20
30

Process finished with exit code 0
```

Single Dimensional Array

1D Array



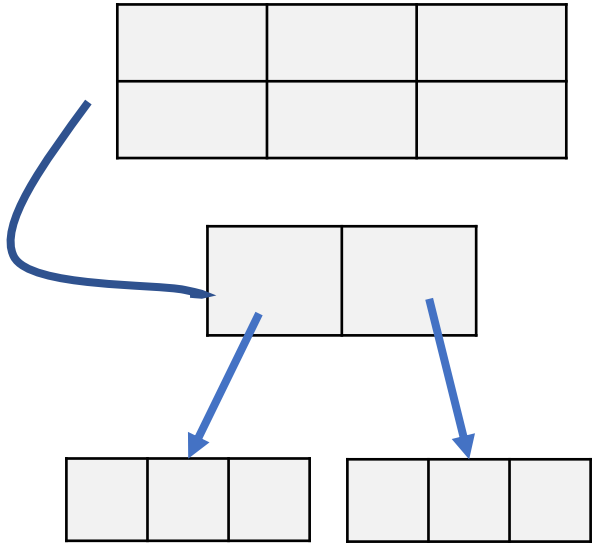
Single Row



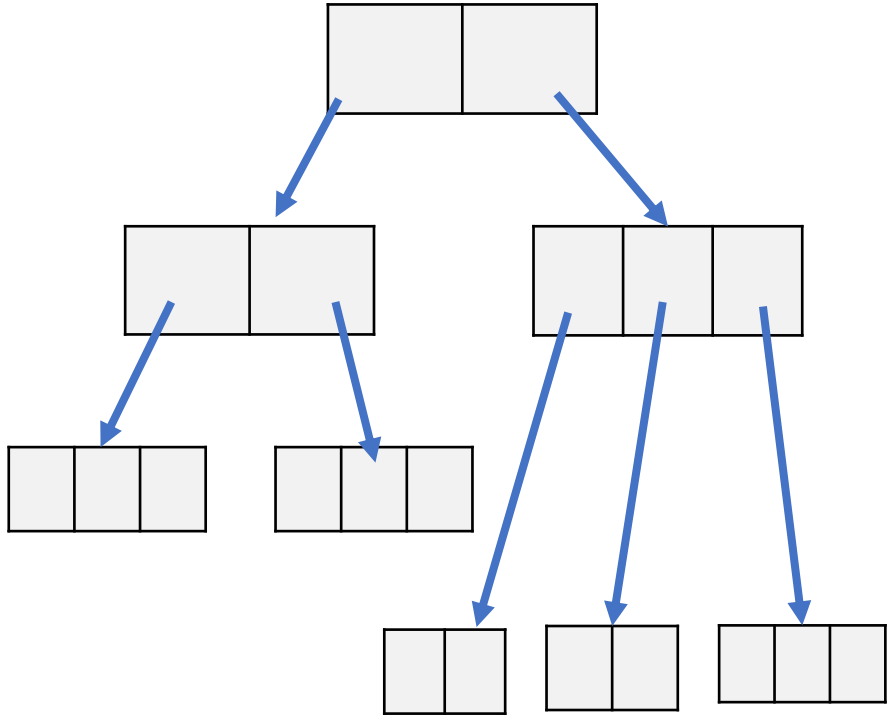
Single Column

Multi Dimensional Array

2D Array



3D Array



2D Array

DECLARATION >

CREATION

INITIALIZATION

```
int[] a;
```

1D array Declaration

```
int[][] a;
```

2D array declaration

```
int[][] a;
int [][]a;
int[] []a;
int a[][];
int[] a[];
```

Most Prefer

Differences

```
int [][] a,b; //a and b are 2d array
int[] a[], b; //a is 2d and b is 1d array
int[] a[], b[]; //a and b are 2d array
int[][] a, b[]; //a is 2d and b is 3d array
int[][] a, []b; //a is 2d and b compile time error
```

2D Array

DECLARATION

CREATION >

INITIALIZATION

```
int[][]  a;
```

2D array Declaration

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

2D array Creation

Rows

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

Column

Declaration + Creation

0,0	**0,1**	**0,2**
1,0	1,1	1,2

0	**1**

0	**1**	**2**

0	**1**	**2**

2D Array

DECLARATION

CREATION >

INITIALIZATION

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

Declaration + Creation

0,0	**0,1**	**0,2**
1,0	1,1	1,2

Matrix Array

Which have same columns

0,0	**0,1**	**0,2**	
1,0	1,1		
2,0	2,1	2,2	2,3

Jagged Array

Which have not same columns

2D Array

DECLARATION

CREATION >

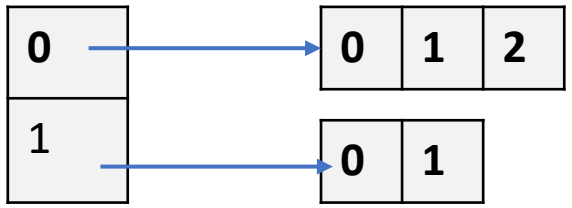
INITIALIZATION

Creating Jagged Array

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

2D array Creation but not defining column

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



2D Array

DECLARATION

CREATION

INITIALIZATION >

Inserting Values in **Matrix Array**

```
int[][] a = new int[2][3];
a[0][0]=10;
a[0][1]=20;
a[1][1]=40;
```

0,0 **10**	0,1 **20**	0,2
1,0	1,1 **40**	1,2

```
int[][] a = { {10,20,30}, {40,50,60} };
```

Declaration + Creation + Initialization

0,0 10	0,1 20	0,2 30
1,0 40	1,1 50	1,2 60

2D Array

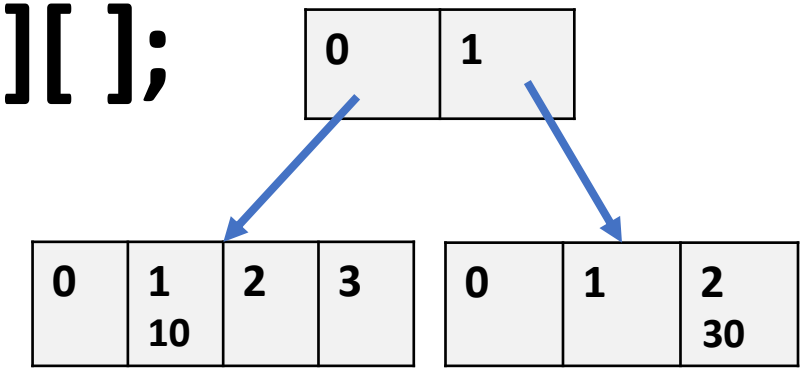
DECLARATION

CREATION

INITIALIZATION >

Inserting Values in **Jagged Array**

```
int[][] a = new int[2][ ];
a[0]= new int[4];
a[1]= new int[3];
```




```
a[0][1]=10;  //initialization
a[1][2]=30;
```

```
Int[][] a = {{10,20,30,40},{50,60,70}}
```

Declaration + Creation + Initialization

2D Array

DECLARATION

CREATION

INITIALIZATION >

Int[][] a = {{10,20,30,40},{50,60},{70,80,90}}

2d array

Int type

Any hash value

S.o.p(a); [[I@....

Print the 2 Dimensional array class

0 10	1 20	2	3 30
0 40	1 50		
0 60	1 70	2 80	

S.o.p(a[0]); [I@...

Print the Single Dimensional array class

S.o.p(a[0][0]); //10

Print the value

S.o.p(a.length); //3

2D Array

DECLARATION

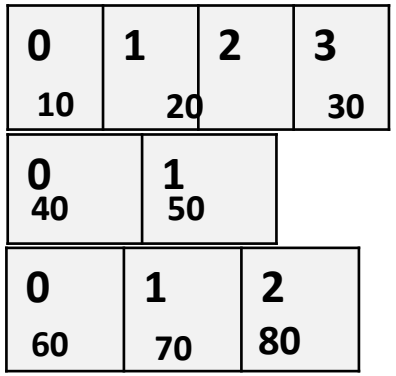
CREATION

INITIALIZATION >

S.o.p(a.length); //3 total rows

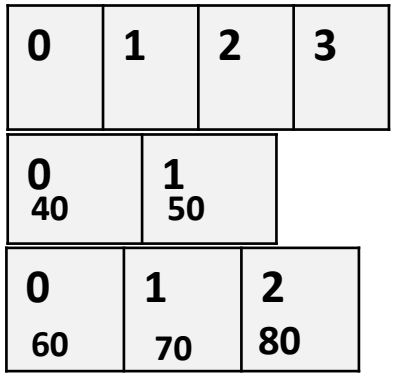
S.o.p(a[0].length); //4 total 1st row

S.o.p(a[0][0].length); //Error



2D Array Retrieve

```
Int[][] a = {{10,20,30,40},{50,60},{70,80,90}}

for(int i=0; i<a.length; i++){
    for(int j=0; j<a[i].length; j++){
        System.out.print(a[i][j]+" ");

    }
System.out.println();
}
```



Main.java

```java
package com.company;
public class Main {
    public static void main(String[] args) {
        int[][] a = {{10,20,30},{40,50},{60,70,80}};

        for(int i=0; i<a.length; i++){
            for(int j=0; j<a[i].length; j++){
                System.out.print(a[i][j]+" ");
            }
            System.out.println();
        }
    }
}
```

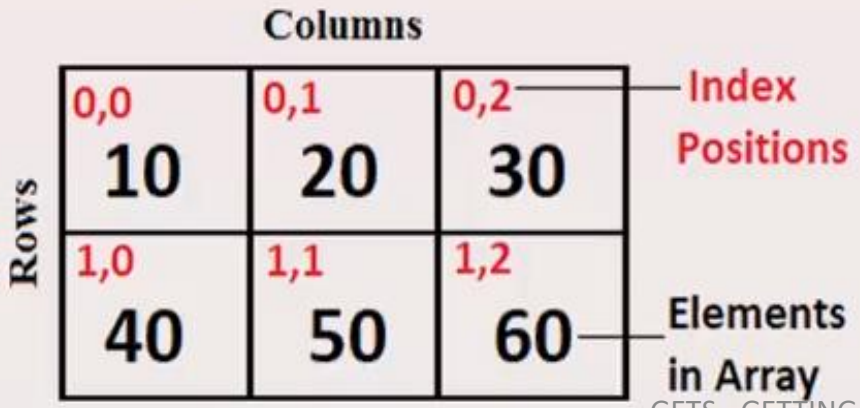
Main

```
"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" "-javaagent:C:\Program File
10 20 30
40 50
60 70 80
```

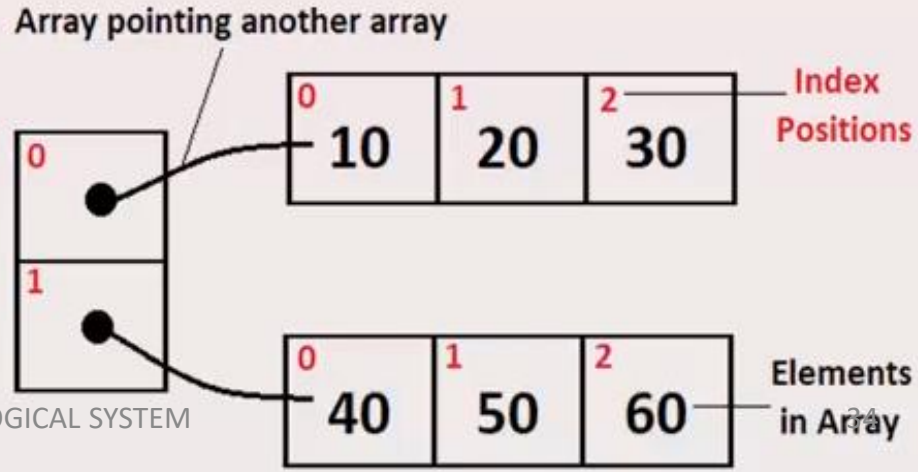
Multi-Dimensional (2-D)

What is Multi-Dimensional Array :

- An array having multiple rows or columns is known as multi-dimensional array.
- These are also known as array of arrays because array is present in another array.
- There are two types of multi-dimensional array :
  1. 2-D Array
  2. 3-D Array
- We can represent 2-D multi-dimensional array as follows :



OR



Multi-Dimensional (2-D)

Declaration Of 2-D Multi-Dimensional Array :

- Diferent Ways of Declaration of Arrays are :

  1. int[ ][ ] a;
  2. int [ ][ ]a;
  3. int[ ][ ]a;
  4. int a[ ][ ];
  5. int[ ] a[ ];

- Most Preferred Declaration is ‘int[ ][ ] a;’, because here ‘a’ is two dimensional int array, thus name is clearly separated with type.

- We cannot provide size at the time of array declaration i.e. ‘int[2][3] a;’ or ‘int a[2][3];’, this type of any statement is incorrect.

- Note that, there is difference between below statements :

  int[ ][ ] a,b; // here ‘a’ and ‘b’ both are 2-D Arrays.
  int[ ] a[ ], b; // here ‘a’ is 2-D and ‘b’ is 1-D Array.
  int[ ] a[ ], b[ ]; // ‘a’ and ‘b’ both are 2-D Array.
  int[ ] [ ]a, [ ] b; // compile time error.
  int[ ] [ ]a, b[ ]; // ‘a’ is 2-D and ‘b’ is 3-D Array.

Multi-Dimensional (2-D)

Creation Of 2-D Multi-Dimensional Array :

- We can create an array after declaration as follows :

```
int[ ][ ] a;  //array declaration
a=new int[2][3];  // array creation
```

Matrix Array Creation

```
int[ ][ ] a; //array declaration
a=new int[2][ ]; //array creation
a[0]=new int[4];
a[1]=new int[3];
```

Jagged Array Creation

- It is compulsory to declare the size of an array at the time of creation.
- We can declare and create 2-D array within a single line as follows :

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

Matrix Array Creation

```
int[ ][ ] a=new int[2][ ];
a[0]=new int[4];
a[1]=new int[3];
```

Jagged Array Creation

3D Array

DECLARATION >

CREATION

INITIALIZATION

```
int[][][]  a;
```

3D array Declaration

```
int[][][] a;
int[] [][]a;
int  [][]a[];
int  []a[][];
```

Most Prefer

3D Array

DECLARATION

CREATION >

INITIALIZATION

```
int[][][]  a;
```

3D array Declaration

3D array Creation

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

3D Array

DECLARATION

CREATION

INITIALIZATION>

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

3D array Creation

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

3D array Initialization

```
int[][][]  a = {{{10,20},{30,40,50,60},{70,80,90}}}
```

3D array Declaration +Creation+Initialization

Main.java

```java
package com.company;
public class Main {
    public static void main(String[] args) {
        int[][][] a = {{{10,20,30,40},{50,60,70},{80,90,100}}};

        for(int i=0; i<a.length; i++){
            for(int j=0; j<a[i].length; j++){
                for(int k=0; k<a[i][j].length; k++){
                    System.out.print(a[i][j][k]+" ");
                }
                System.out.println();
            }
        }
    }
}
```

Main

```
"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBr
10 20 30 40
50 60 70
80 90 100
```