Arrays

Arrays:

- It is a non-primitive data type.
- It is a continuous block of memory which is used to store multiple data's.

Characteristics of Array:

- Array size is fixed.
- It follows homogeneous type of data (Similar type of data).
- It always follow index values which starts from 0 to size-1(n-1).

Declaration of an Array:

Syntax:

```
data type[] reference variable;
```

Initialization of array:

We can initialize an array in 2 ways,

1. Using new keyword
2. Without using new keyword

1.Using new keyword:

Syntax:

```
new data type [size];
```

Initialization and declaration:

Syntax:

```
data type[]  reference variable =  new data type[size];
```

Reference variable:

- A variable which is used to store address of an array is known as reference variable.

Store data:

Syntax:

reference variable[index value] = value/expression/variable;

Fetch data:

Syntax:

reference variable[index value];

Array length: A length is a variable.

Syntax:

reference variable .length;

Note: *Length is predefined variable.*

Array Index out of Bound Exception (AIOBE):

It is a runtime exception in java that occurs when we access an array element with an index value is either negative or greater than or equals to size of an array (or) when we try to access an array element that doesn't exist.

2.Without Using new keyword:

Syntax:

data type [ ] reference variable = {value1, value2,……..etc.};

Sample program:

```
public class Using_new_keyword {

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

```
		int[] a = new int[4];
		System.out.println(a);
		a[0] = 10;
		a[3] = 40;
		int b = 20;
		a[2] = b;
		a[3] = b + 30;
		System.out.println(a[0]);
		System.out.println(a[3]);
		System.out.println(a[1]);

	}

}
/*
O/P:

[I@5ca881b5
10
50
0
*/
```

Dynamic initialization of an Array:

Sample program:

```
package Arrays;

import java.util.Scanner;
import java.util.Arrays;

public class Dynamic_initialization_of_an_Array {

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

		Scanner scn = new Scanner(System.in);
		System.out.println("Enter size of an array");
		int size=scn.nextInt();
		int[]arr=new int[size];

		for(int i=0;i<arr.length-1;i++) {
			System.out.println("Enter Array element at pirticular index
position:"+i);
			arr[i]=scn.nextInt();
		}

		System.out.println("----------------------");
		for(int i=0;i<arr.length-1;i++){
			System.out.println(arr[i]+"");

		}
```

```
			System.out.println("----------------------");
			System.out.println(Arrays.toString(arr));

		}

}

/*
O/P:

Enter size of an array
5
Enter Array element at pirticular index position:0
2
Enter Array element at pirticular index position:1
*/
```

Method in Arrays:

Sample program:

```
package Arrays;

public class Methods_in_Arrays {

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

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

		}

		private static void demo(int[] a) {
			System.out.println(a[0]);

		}

}
/*
O/P:

10

*/
```

Types of Arrays:

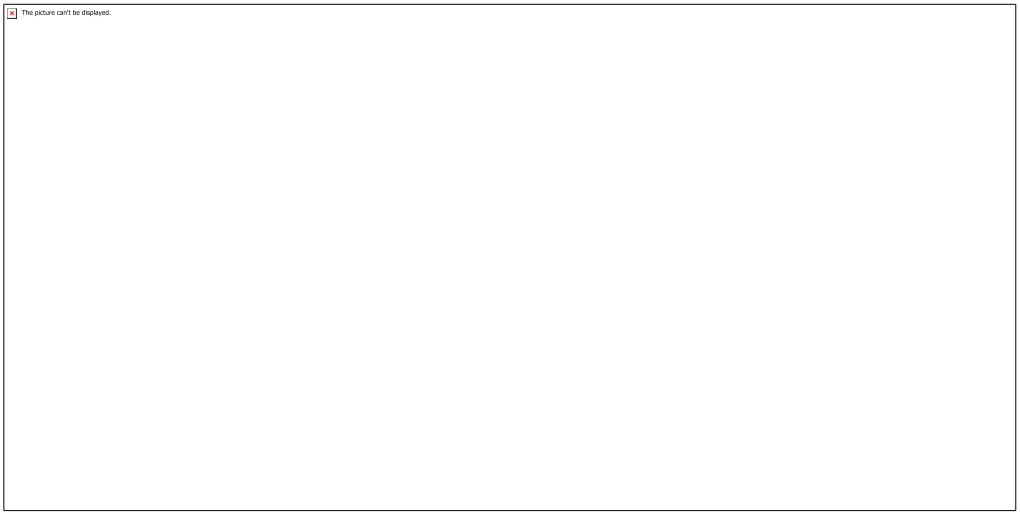
There are 2 types of Arrays,

1. On the basis of Size.
    1. Fixed Size Arrays
    2. Dynamic Size Arrays

2. On the basis of Dimensions.

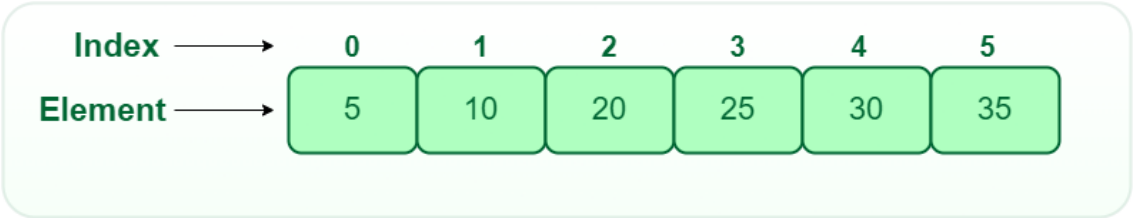
    1. One Dimensional Arrays
    2. Multi Dimensional Arrays.

        a. Two-Dimensional Arrays
        b. Three-Dimensional Arrays



One Dimensional Arrays:

- An Array which stores an elements as a row one after another is known as One-Dimensional Array.



Synatax:

```
datatype array name [array size];
```

Example:

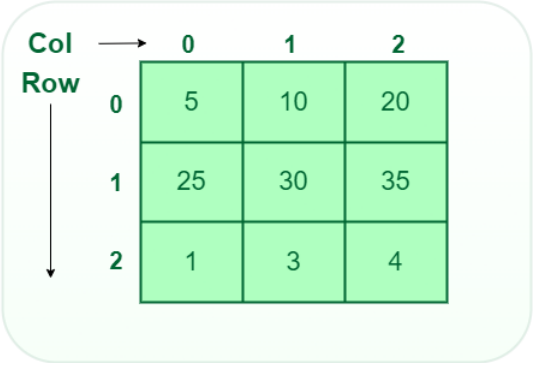
Int num [5];

Sample Program :

```
package Arrays;

public class OneDArrayExample {
	public static void main(String[] args) {
		// Declare & initialize an array of 5 integers
		int[] numbers = { 10, 20, 30, 40, 50 };

		int sum = 0; // Variable to store sum of array elements

		// Loop through the array to calculate sum
		for (int i = 0; i < numbers.length; i++) {
			sum += numbers[i]; // Add each element to sum
		}

		double average = (double) sum / numbers.length; // Calculate average

		// Print the array, sum, and average
		System.out.print("Array Elements: ");
		for (int num : numbers) {
			System.out.print(num + " "); // Print each element
		}

		System.out.println("\nSum: " + sum); // Print total sum
		System.out.println("Average: " + average); // Print average
	}
}
/*
O/P:

Array Elements: 10 20 30 40 50
Sum: 150
Average: 30.0
*/
```

Two-Dimensional Arrays:

- An Array which stores an elements as a rows and columns is known as Two-Dimensional Arrays.



Synatax:

datatype array name [array size 1] [array size 2];

Example:

Int num [5] [5];

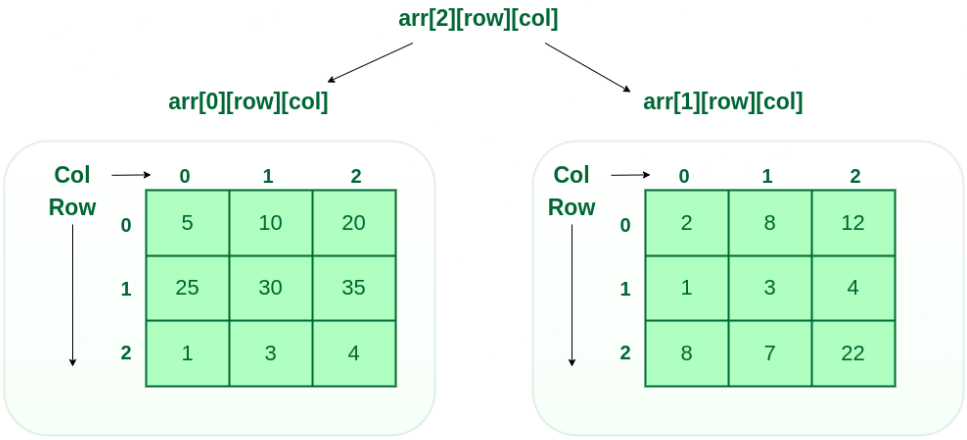
Sample Program:

```
package Arrays;

//Program to demonstrate 2D array operations
public class TwoDArrayExample {
	public static void main(String[] args) {

		// 1. Declare & initialize a 2D array (3 rows x 2 columns)
		int[][] matrix = { { 10, 20 }, // Row 0
				{ 30, 40 }, // Row 1
				{ 50, 60 } // Row 2
		};

		// 2. Access & print element at row 1, column 0 (output: 30)
		System.out.println("Element at [1][0]: " + matrix[1][0]);

		// 3. Print entire matrix using nested loops
		System.out.println("\nFull Matrix:");
		for (int i = 0; i < matrix.length; i++) { // Loop through rows
			for (int j = 0; j < matrix[i].length; j++) { // Loop through
columns
				System.out.print(matrix[i][j] + " "); // Print each
element
			}
			System.out.println(); // New line after each row
		}

		// 4. Modify value at row 2, column 1 (change 60 to 99)
		matrix[2][1] = 99;

		// 5. Print updated value
		System.out.println("\nUpdated value at [2][1]: " + matrix[2][1]);
	}
}
/*
```

```
 * O/P:
Element at [1][0]: 30

Full Matrix:
10 20
30 40
50 60

Updated value at [2][1]: 99
 */
```

Three-Dimensional Arrays:

- An Array which stores an elements in 3 dimensions is known as Three-Dimensional Arrays.



Synatax:

datatype array name [array size 1] [array size 2] [array size 3];

Example:

Int num [5] [5][5];

Sample program:

```
package Arrays;

public class ThreeDArrayDemo {
	public static void main(String[] args) {
		// 1. Declare & initialize a 3D array (2x3x4)
		int[][][] arr = { { { 1, 2, 3, 4 }, { 5, 6, 7, 8 }, { 9, 10, 11, 12 } },
// Layer 0
				{ { 13, 14, 15, 16 }, { 17, 18, 19, 20 }, { 21, 22, 23, 24
} } // Layer 1
```

```
            };

            // 2. Access and print elements
            for (int i = 0; i < arr.length; i++) { // Loop through layers (1st
dimension)
                  System.out.println("Layer " + i + ":");
                  for (int j = 0; j < arr[i].length; j++) { // Loop through rows
(2nd dimension)
                        for (int k = 0; k < arr[i][j].length; k++) { // Loop
through columns (3rd dimension)
                              System.out.print(arr[i][j][k] + " ");
                        }
                        System.out.println(); // New line after each row
                  }
                  System.out.println(); // New line after each layer
            }
      }
}
/*
O/P:

Layer 0:
1 2 3 4
5 6 7 8
9 10 11 12

Layer 1:
13 14 15 16
17 18 19 20
21 22 23 24
*/
```