Introduction to Arrays in Java

An array is a collection of items of same data type stored at contiguous memory locations.

This makes it easier to calculate the position of each element by simply adding an offset to a base value, i.e., the memory location of the first element of the array (generally denoted by the name of the array). The base value is index 0 and the difference between the two indexes is the offset.

For simplicity, we can think of an array as a fleet of stairs where on each step is placed a value (let's say one of your friends). Here, you can identify the location of any of your friends by simply knowing the count of the step they are on.
Remember: Location of next index depends on the data type we use.

Ways to Create Array in Java

Here are four different ways to create an array of size 3 with elements 10, 20, and 30 in Java:

1. Declare and Allocate Memory Separately:

In this method, first, you declare an array, then allocate memory for it, and finally assign values to its elements.

// Declare array

int a[];

```
// Allocate memory for 3 elements

a = new int[3];
a[0] = 10;
a[1] = 20;
a[2] = 30;
```

2. Declare and Allocate Memory in One Statement:

In this method, both the array declaration and memory allocation are done in a single statement.

```
// Declare and allocate memory for 3 elements

int[] a = new int[3];
a[0] = 10;
a[1] = 20;
a[2] = 30;
```

3. Initialize the Array Inline:

Here, you can directly initialize the array with values during the declaration.

```
// Declare and initialize array with values

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

4. Use a Loop to Assign Values:

In this method, the array is declared and memory is allocated first, then a loop is used to assign values to the array elements.

```
// Declare and allocate memory for 3 elements

int[] a = new int[3];
for (int i = 0; i < a.length; i++) {
a[i] = (i + 1) * 10;
}
```

What are the different operations on the array?

Arrays allow random access to elements. This makes accessing elements by position faster. Hence operation like searching, insertion, and access becomes really efficient. Array elements can be accessed using the loops.

1. Insertion in Array:

We try to insert a value to a particular array index position, as the array provides random access it can be done easily using the assignment operator.

Pseudo Code:

```
// to insert a value= 10 at index position 2;


arr[ 2 ] = 10;
```

Time Complexity:

- O(1) to insert a single element
- O(n) to insert all the array elements [where n is the size of the array]

2. Access elements in Array:

Accessing array elements become extremely important, in order to perform operations on arrays.

Pseudo Code:

// to access array element at index position 2, we simply can write

return arr[2] ;

Time Complexity: O(1)

3. Searching in Array:

We try to find a particular value in the array, in order to do that we need to access all the array elements and look for the particular value.

Pseudo Code:

// searching for value 2 in the array;

Loop from i = 0 to 5:
    check if arr[i] = 2:
        return true;

Time Complexity: O(n), where n is the size of the array.

Here is the code for working with an array in Java:

```
class GfG {

    public static void main(String[] args) {


        // Creating an integer array

        // named arr of size 10.

        int[] arr = new int[10];
```

```
        // accessing element at 0 index
        // and setting its value to 5.
        arr[0] = 5;


        // access and print value at 0
        // index we get the output as 5.
        System.out.println(arr[0]);
    }
}
```

Output

```
5
```