**Java Arrays**

Normally, an array is a collection of similar type of elements which has contiguous memory location.

**Java array** is an object which contains elements of a similar data type. Additionally, The elements of an array are stored in a contiguous memory location. It is a data structure where we store similar elements. We can store only a fixed set of elements in a Java array.

Array in Java is index-based, the first element of the array is stored at the 0th index, 2nd element is stored on 1st index and so on.

Unlike C/C++, we can get the length of the array using the length member. In C/C++, we need to use the sizeof operator.

In Java, array is an object of a dynamically generated class. Java array inherits the Object class, and implements the Serializable as well as Cloneable interfaces. We can store primitive values or objects in an array in Java. Like C/C++, we can also create single dimentional or multidimentional arrays in Java.

Moreover, Java provides the feature of anonymous arrays which is not available in C/C++.

* In Java, all arrays are dynamically allocated. (discussed below)
* Arrays may be stored in contiguous memory [consecutive memory locations].
* Since arrays are objects in Java, we can find their length using the object property *length*. This is different from C/C++, where we find length using sizeof.
* A Java array variable can also be declared like other variables with [] after the data type.
* The variables in the array are ordered, and each has an index beginning with 0.
* Java array can also be used as a static field, a local variable, or a method parameter.
* The **size** of an array must be specified by int or short value and not long.
* The direct superclass of an array type is [Object](https://www.geeksforgeeks.org/object-class-in-java/).
* Every array type implements the interfaces Cloneable and [java.io.Serializable](https://www.geeksforgeeks.org/serialization-in-java/).
* This storage of arrays helps us randomly access the elements of an array [Support Random Access].
* The size of the array cannot be altered(once initialized).  However, an array reference can be made to point to another array.

## How to declare an array in Java?

dataType[] arrayName;

The Time Complexity of different operations in an array is:

| **ARRAY OPERATION** | **REAL TIME COMPLEXITY** | **ASSUMED TIME COMPLEXITY** |
| --- | --- | --- |
| Access i-th element | O(√N) | O(1) |
| Traverse all elements | O(N + √N) | O(N) |
| Override element at i-th index | O(√N) | O(1) |
| Insert element E | O(N + √N) | O(N) |
| Delete element E | O(N + √N) | O(N) |

# Space Complexity of Array

The Space Complexity of the above array operations is O(1).

This is because we do not need extra space beyond a fixed number of variables.

For some operations, you may need extra space of the order of O(N). For example, sorting an array using a sorting algorithm that is not in-place.

### **Advantages**

* **Code Optimization:** It makes the code optimized, we can retrieve or sort the data efficiently.
* **Random access:** We can get any data located at an index position.

### **Disadvantages**

* **Size Limit:** We can store only the fixed size of elements in the array. It doesn't grow its size at runtime. To solve this problem, collection framework is used in Java which grows automatically.