A Simple Guide to Competitive Programming

By Aakash Shah

# Arrays

* An array is a collection of items stored at contiguous memory locations.
* The idea is to store multiple items of the same type together.
* 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).

Types of indexing in array:

* 0 (zero-based indexing): The first element of the array is indexed by subscript of 0
* 1 (one-based indexing): The first element of the array is indexed by subscript of 1
* n (n-based indexing): The base index of an array can be freely chosen. Usually programming languages allowing n-based indexing also allow negative index values and other scalar data types like enumerations, or characters may be used as an array index.
* Why do we need arrays?

We can use normal variables (v1, v2, v3, ..) when we have small number of objects, but if we want to store large number of instances, it becomes difficult to manage them with normal variables. The idea of array is to represent many instances in one variable.

# Array Declaration in C/C++

# Facts about Array in C/C++

* Accessing array elements:

Array elements are accessed by using an integer index. Array index starts with 0 and goes till size of array minus 1.

* No index out of bound Checking:

There is no index out of bound checking in C, for example the following program compiles fine but may produce unexpected output when run.

* Also, In C, it is not compiler error to initialize an array with more elements than specified size. It compiles just fine and shows a warning. However if the array was declared in a c++ source code file, it would generate a compiler error : too many initializers for <array\_name>.
* Array vs Pointers

Arrays and pointer are two different things (we can check by applying sizeof). The confusion happens because array name indicates address of first element and arrays are always passed as pointers (even if we use square bracket).

# More Facts of Arrays in C

* In C, it is possible to have array of all types except void and functions.
* In C, array and pointer are different. They seem similar because array name gives address of first element and array elements are accessed using pointer arithmetic.
* Arrays are always passed as pointer to functions.
* A character array initialized with double quoted string has last element as ‘\0’.
* Like other variables, arrays can be allocated memory in any of the three segments, data, heap, and stack. Dynamically allocated arrays are allocated memory on heap, static or global arrays are allocated memory on data segment and local arrays are allocated memory on stack segment.

# Initialization of a multidimensional array in C/C++

* A multidimensional array can have the **leftmost dimension** as **optional.**
* Except the left most dimension, all other dimensions must be specified.

# Arrays in Java

* All arrays are dynamically allocated
* Since arrays are objects in Java, we can find their length using member 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 have an index beginning from 0.
* Java array can be also be used as a static field, a local variable or a method parameter.
* The **size** of an array must be specified by an int value and not long or short.
* The direct superclass of an array type is Object.
* Every array type implements the interfaces *Cloneable* and *java.io.Serializable*.
* Array can contains primitives data types as well as objects of a class depending on the definition of array. In case of primitives data types, the actual values are stored in contiguous memory locations. In case of objects of a class, the actual objects are stored in heap segment.
* The elements in the array allocated by new will automatically be initialized to **zero** (for numeric types), **false** (for boolean), or **null** (for reference types).
* Obtaining an array is a two-step process. First, you must declare a variable of the desired array type. Second, you must allocate the memory that will hold the array, using new, and assign it to the array variable. Thus, **in Java all arrays are dynamically allocated.**

Multidimensional Arrays in Java

Multidimensional arrays are **arrays of arrays** with each element of the array holding the reference of other array. These are also known as [Jagged Arrays](https://www.geeksforgeeks.org/jagged-array-in-java/). A multidimensional array is created by appending one set of square brackets ([]) per dimension.