ARRAY

What is Array ?

An array is a fixed-size collection of similar data items stored in contiguous memory locations. It can be used to store the collection of primitive data types such as int, char, float, etc., and also derived and user-defined data types such as pointers, structures, etc.

Array Declaration:

We have to declare the array like any other variable before using it. We can declare an array by specifying its name, the type of its elements, and the size of its dimensions. When we declare an array, the compiler allocates the memory block of the specified size to the array name.

Syntax of Array Declaration:

data_type array_name [size];

or

data_type array_name [*size1*] [*size2*]...[*sizeN*];

Array Initialization

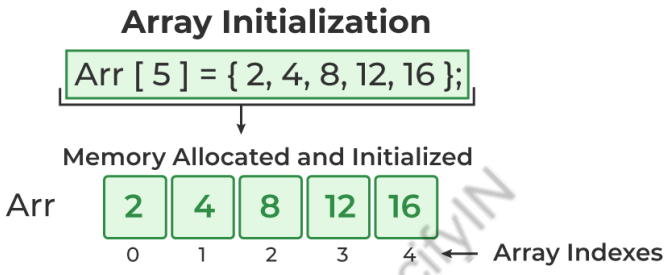
Initialization is the process to assign some initial value to the variable. When the array is declared or allocated memory, the elements of the array contain some garbage value. So, we need to initialize the array to some meaningful value. There are multiple ways in which we can initialize an array.

1. Array Initialization with Declaration:

In this method, we initialize the array along with its declaration. We use an initializer list to initialize multiple elements of the array. An initializer list is the list of values enclosed within braces { } separated b a comma.

Example:

data_type array_name [*size*] = {value1, value2, ... valueN};



2. Array Initialization with Declaration without Size:

If we initialize an array using an initializer list, we can skip declaring the size of the array as the compiler can automatically deduce the size of the array in these cases. The size of the array in these cases is equal to the number of elements present in the initializer list as the compiler can automatically deduce the size of the array.

Example:

data_type array_name[] = {1,2,3,4,5};

3. Array Initialization after Declaration (Using Loops):

We initialize the array after the declaration by assigning the initial value to each element individually. We can use for loop, while loop, or do-while loop to assign the value to each element of the array.

```
for (int i = 0; i < N; i++) {
   array_name[i] = valuei;
}
```

Access Array Elements:

We can access any element of an array using the array subscript operator [ ] and the index value *i* of the element.

array_name [*index*];

One thing to note is that the indexing in the array always starts with 0, i.e., the first element is at index 0 and the last element is at N – 1 where N is the number of elements in the array.

Update Array Element:

We can update the value of an element at the given index i in a similar way to accessing an element by using the array subscript operator [ ] and assignment operator =.

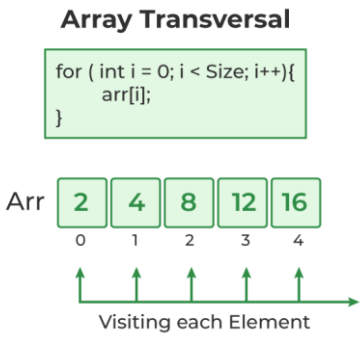
array_name[*i*] = *new_value*;

Array Traversal:

Traversal is the process in which we visit every element of the data structure. For array traversal, we use loops to iterate through each element of the array.

Syntax Array Traversal using for Loop:

```
for (int i = 0; i < N; i++) {
   array_name[i];
}
```



Types of Array :

There are two types of arrays based on the number of dimensions it has. They are as follows:

1. One Dimensional Arrays (1D Array)
2. Multidimensional Arrays

1. One Dimensional Array :

The One-dimensional arrays, also known as 1-D arrays are those arrays that have only one dimension.

Syntax of 1D Array :

array_name [size];

2. Multidimensional Array :

Multi-dimensional Arrays are those arrays that have more than one dimension. Some of the popular multidimensional arrays are 2D arrays and 3D arrays. We can declare arrays with more dimensions than 3d arrays but they are avoided as they get very complex and occupy a large amount of space.

A. Two-Dimensional Array :

A Two-Dimensional array or 2D array is an array that has exactly two dimensions. They can be visualized in the form of rows and columns organized in a two-dimensional plane.

Syntax of 2D Array :

array_name[size1] [size2];

Here,

- size1: Size of the first dimension.
- size2: Size of the second dimension.

Properties of Arrays :

It is very important to understand the properties of the array so that we can avoid bugs while using it. The following are the main properties of arrray:

1. Fixed Size

The array in C is a fixed-size collection of elements. The size of the array must be known at the compile time and it cannot be changed once it is declared.

2. Homogeneous Collection

We can only store one type of element in an array. There is no restriction on the number of elements but the type of all of these elements must be the same.

3. Indexing in Array

The array index always starts with 0. It means that the index of the first element of the array will be 0 and the last element will be N – 1.

4. Dimensions of an Array

A dimension of an array is the number of indexes required to refer to an element in the array. It is the number of directions in which you can grow the array size.

5. Contiguous Storage

All the elements in the array are stored continuously one after another in the memory. It is one of the defining properties of the array which is also the reason why random access is possible in the array.

6. Random Access

The array provides random access to its element i.e. we can get to a random element at any index of the array in constant time complexity just by using its index number.

7. No Index Out of Bounds Checking

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