**Introduction**

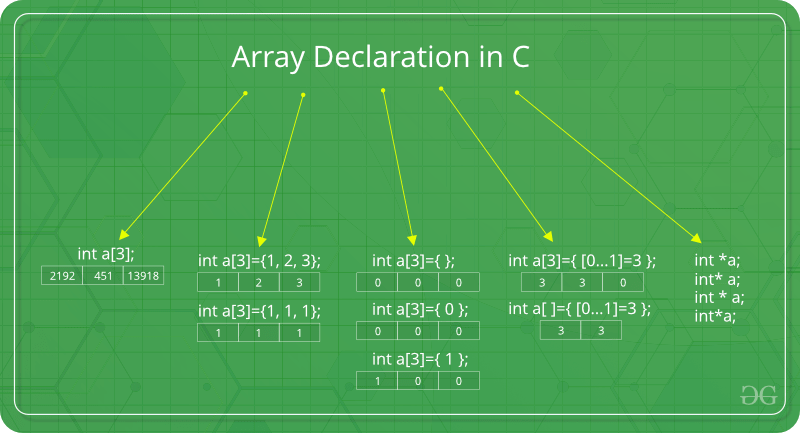
An array in C or C++ is a collection of items stored at contiguous memory locations and elements can be accessed randomly using indices of an array. They are used to store similar type of elements as in the data type must be the same for all elements. They can be used to store collection of primitive data types such as int, float, double, char, etc of any particular type. To add to it, an array in C or C++ can store derived data types such as the structures, pointers etc. Given below is the picturesque representation of an array.



Why do we need arrays?

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

Array declaration in C/C++:



There are various ways in which we can declare an array. It can be done by specifying its type and size, by initializing it or both.

**1 Array declaration by specifying size**

int arr1[10];

**2.Array declaration by initializing elements**

int arr[] = { 10, 20, 30, 40 }

**3.Array declaration by specifying size and initializing elements**

int arr[6] = { 10, 20, 30, 40 }

Advantages of an Array in C/C++:

1. Random access of elements using array index.
2. Use of less line of code as it creates a single array of multiple elements.
3. Easy access to all the elements.
4. Traversal through the array becomes easy using a single loop.
5. Sorting becomes easy as it can be accomplished by writing less line of code.

Disadvantages of an Array in C/C++:

1. Allows a fixed number of elements to be entered which is decided at the time of declaration. Unlike a linked list, an array in C is not dynamic.
2. Insertion and deletion of elements can be costly since the elements are needed to be managed in accordance with the new memory allocation.

**Example 1:**

#include <stdio.h>

int main()

{

int arr[5];

arr[0] = 5;

arr[2] = -10;

arr[3 / 2] = 2; // this is same as arr[1] = 2

arr[3] = arr[0];

printf("%d %d %d %d", arr[0], arr[1], arr[2], arr[3]);

return 0;

}

**Output:**

5 2 -10 5

## **Example 2:**

#include <stdio.h>

int main () {

int n[ 10 ];

int i;

for ( i = 0; i < 10; i++ ) {

n[ i ] = i + 100; /\* set element at location i to i + 100 \*/

}

for (i= 0; i < 10; i++ ) {

printf("Element[%d] = %d\n", i, n[i] );

}

return 0;

}

**Output:**

Element[0] = 100

Element[1] = 101

Element[2] = 102

Element[3] = 103

Element[4] = 104

Element[5] = 105

Element[6] = 106

Element[7] = 107

Element[8] = 108

Element[9] = 109

**Example 3:**

**C program to store and calculate the sum of 5 numbers entered by the user using arrays.**

#include<stdio.h>

int main()

{

int num[5],sum=0,i;

for( i=0; i<5;i++)

{

printf("Enter %d number:",i+1);

scanf("%d",&num[i]);

}

for( i=0;i<5;i++)

{

sum=sum+num[i];

}

printf("Total Sum=%d",sum);

return 0;

}

**Output:**

Enter 1 number:1

Enter 2 number:12

Enter 3 number:1

Enter 4 number:1

Enter 5 number:23

Total Sum=38

**Types of Arrays**

In c programming language, arrays are classified into **two types**. They are as follows...

1. **Single Dimensional Array / One Dimensional Array**
2. **Multi Dimensional Array**

# **Single Dimensional Array**

In c programming language, single dimensional arrays are used to store list of values of same datatype. In other words, single dimensional arrays are used to store a row of values. In single dimensional array, data is stored in linear form. Single dimensional arrays are also called as **one-dimensional arrays**, **Linear Arrays** or simply **1-D Arrays**.

Declaration of Single Dimensional Array

We use the following general syntax for declaring a single dimensional array...

datatype arrayName [ size ] ;

# **Example Code**

int rollNumbers [60] ;

The above declaration of single dimensional array reserves 60 continuous memory locations of 2 bytes each with the name **rollNumbers** and tells the compiler to allow only integer values into those memory locations.

Initialization of Single Dimensional Array

We use the following general syntax for declaring and initializing a single dimensional array with size and initial values.

datatype arrayName [ size ] = {value1, value2, ...} ;

# **Example Code**

int marks [6] = { 89, 90, 76, 78, 98, 86 } ;

# 

# **Multi Dimensional Array**

An array of arrays is called as multi dimensional array. In simple words, an array created with more than one dimension (size) is called as multi dimensional array. Multi dimensional array can be of **two dimensional array** or **three dimensional array** or **four dimensional array** or more...

Most popular and commonly used multi dimensional array is **two dimensional array**. The 2-D arrays are used to store data in the form of table. We also use 2-D arrays to create mathematical **matrices**.

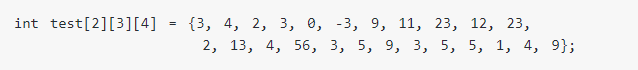
### **Initialisation of two dimensional array**

int test[2][3] = {2, 4, -5, 9, 0, 9};

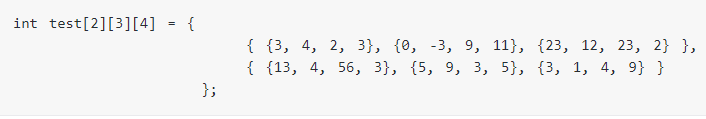
Better way to initialise this array with same array elements as above.

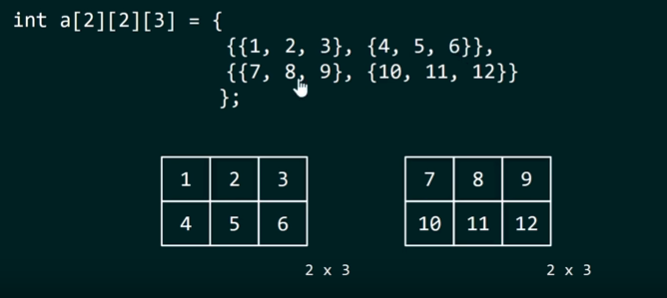
int test[2][3] = { {2, 4, 5}, {9, 0 0}};

### **Initialisation of three dimensional array**



Better way to initialise this array with same elements as above.





Example:

**Example:**

**C Program to display all elements of an initialised two dimensional array.**

#include<stdio.h>

int main()

{

int test[3][2] ={{2, -5},{4, 0},{9, 1}} , i,j;

for( i = 0; i < 3; ++i)

{

for( j = 0; j < 2; ++j)

{

printf("test[%d][%d]=%d \n",i,j,test[i][j]);

}

}

return 0;

}

**Output:**

test[0][0]=2

test[0][1]=-5

test[1][0]=4

test[1][1]=0

test[2][0]=9

test[2][1]=1

**Example:**

**C++ Program to store temperature of two different cities for a week and display it.**

#include<stdio.h>

int CITY = 2;

int WEEK = 7;

int main()

{

int temperature[CITY][WEEK],i,j;

printf( "Enter all temperature for a week of first city and then second city. \n");

// Inserting the values into the temperature array

for ( i = 0; i < CITY; ++i)

{

for( j = 0; j < WEEK; ++j)

{

printf("City %d , Day %d=",i+1,j+1);

scanf("%d",&temperature[i][j]);

}

}

printf("\nDisplaying Values");

// Accessing the values from the temperature array

for ( i = 0; i < CITY; ++i)

{

for( j = 0; j < WEEK; ++j)

{

printf("City %d , Day %d = %d \n",i+1,j+1,temperature[i][j]);

}

}

return 0;

}

**Output:**

Enter all temperature for a week of first city and then second city.

City 1, Day 1 : 32

City 1, Day 2 : 33

City 1, Day 3 : 32

City 1, Day 4 : 34

City 1, Day 5 : 35

City 1, Day 6 : 36

City 1, Day 7 : 38

City 2, Day 1 : 23

City 2, Day 2 : 24

City 2, Day 3 : 26

City 2, Day 4 : 22

City 2, Day 5 : 29

City 2, Day 6 : 27

City 2, Day 7 : 23

Displaying Values:

City 1, Day 1 = 32

City 1, Day 2 = 33

City 1, Day 3 = 32

City 1, Day 4 = 34

City 1, Day 5 = 35

City 1, Day 6 = 36

City 1, Day 7 = 38

City 2, Day 1 = 23

City 2, Day 2 = 24

City 2, Day 3 = 26

City 2, Day 4 = 22

City 2, Day 5 = 29

City 2, Day 6 = 27

City 2, Day 7 = 23