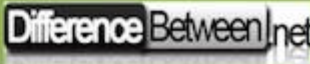


Array :

Array is a collection of homogenous data stored under unique name. The values in an array is called as 'elements of an array.' These elements are accessed by numbers called as 'subscripts or index numbers.' Arrays may be of any variable type.

Array is also called as 'subscripted variable.'

ARRAY VERSUS STRING

| Array | String |
|---|--|
| Arrays are a sequential collection of elements of similar data types. | Strings refer to a sequence of characters represented as a single data type. |
| Elements of arrays are stored contiguously in increasing memory locations. | Strings can be stored in any manner in memory locations. |
| An array is a special variable that can hold more than one value at a time. | Strings can only hold char data which are the most commonly used data types. |
| Arrays are mutable meaning the fields can be modified. | Strings are immutable meaning the value cannot be changed in memory once created. |
| The length of an array is fixed. | The size of a string is not fixed. |
| |  |

Other differences includes

Array is not ended with null character by default. But, string is ended with null ('\\0') character by default.

Arrays can be one or multidimensional, Strings are always two-dimensional

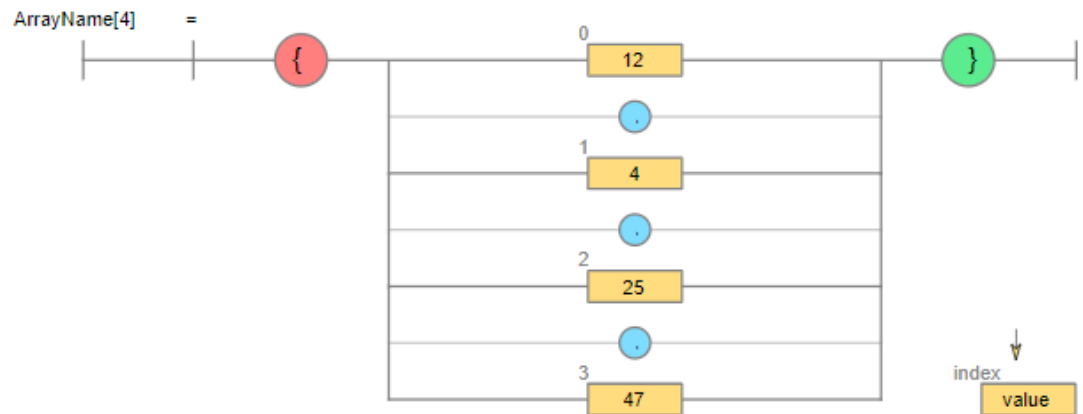
Types of an Array :

1. [One / Single Dimensional Array](#)
2. [Two Dimensional Array](#)

Single / One Dimensional Array :

The array which is used to represent and store data in a linear form is called as 'single or one dimensional array.'

An array which has only one subscript is known as **one dimensional array** i.e) `int arr[10]`.



Syntax:

`<data-type> <array_name> [size];`

Example:

```
int a[] = {2, 3, 5};  
int a[];  
char string[];
```

```
char ch[20] = "TechnoExam" ;  
float stax[3] = {5003.23, 1940.32, 123.20} ;
```

Total Size (in Bytes):

total size = length of array * size of data type

In above example, a is an array of type integer which has storage size of 3 elements. The total size would be $3 * 2 = 6$ bytes.

*** Memory Allocation :**

| | | | | |
|---------|-------|-------|-------|-------|
| a [i] | a [0] | a [1] | a [2] | a [n] |
| element | 5 | 7 | 2 | 12 |
| Address | 1000 | 1002 | 1004 | n |

Fig : Memory allocation for one dimensional array

Program :

/* Program to demonstrate one dimensional array.

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int a[3], i;;
    printf("\n\t Enter three numbers : ");
    for(i=0; i<3; i++)
    {
        scanf("%d", &a[i]); // read array
    }
    printf("\n\n\t Numbers are : ");
    for(i=0; i<3; i++)
    {
        printf("\t %d", a[i]); // print array
    }
    getch();
}
```

Output :

Enter three numbers : 9 4 6

Numbers are : 9 4 6_

Features :

- Array size should be positive number only.

- String array always terminates with null character ('\0').
- Array elements are countered from 0 to n-1.
- Useful for multiple reading of elements (numbers).

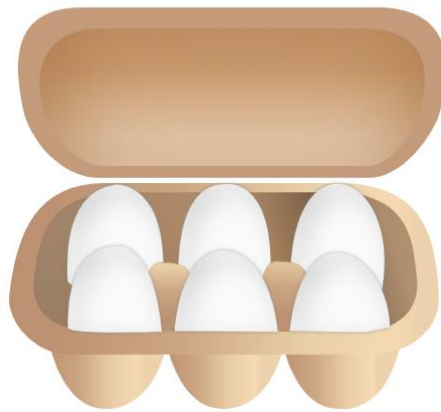
Disadvantages :

- There is no easy method to initialize large number of array elements.
- It is difficult to initialize selected elements.

Two Dimensional Array :

The array which is used to represent and store data in a tabular form is called as 'two dimensional array.' Such type of array specially used to represent data in a matrix form.

An array, which has two subscript is known as two dimensional array. By using two-dimensional arrays, programmers can manipulate data structures. It has two subscripts, first one represents row and second one represents column. Subscripts are always starts with 0.



The following syntax is used to represent two dimensional array.

Syntax:

```
<data-type> <array_nm> [row_subscript][column-subscript];
```

Example:

```
int a[3][3];
```

In above example, a is an array of type integer which has storage size of 3 * 3 matrix. The total size would be 3 * 3 * 4 = 36 bytes.

It is also called as 'multidimensional array.'

*** Memory Allocation :**

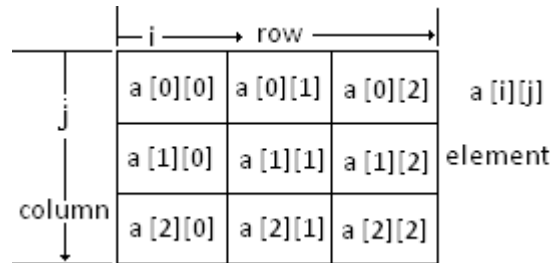


Fig : Memory allocation for two dimensional array

Program :

/* Program to demonstrate two dimensional array.

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int a[3][3], i, j;
    printf("\n\t Enter matrix of 3*3 : ");
    for(i=0; i<3; i++)
    {
        for(j=0; j<3; j++)
        {
            scanf("%d",&a[i][j]); //read 3*3 array
        }
    }
    printf("\n\t Matrix is : \n");
    for(i=0; i<3; i++)
    {
        for(j=0; j<3; j++)
        {
            printf("\t %d",a[i][j]); //print 3*3 array
        }
        printf("\n");
    }
    getch();
}
```

Output :

Enter matrix of 3*3 : 3 4 5 6 7 2 1 2 3

Matrix is :

| | | |
|---|---|----|
| 3 | 4 | 5 |
| 6 | 7 | 2 |
| 1 | 2 | 3_ |

Limitations of two dimensional array :

- We cannot delete any element from an array.
- If we dont know that how many elements have to be stored in a memory in advance, then there will be memory wastage if large array size is specified.