ARRAYS AND STRINGS

INTRODUCTION TO ARRAYS:

An array is a collection of homogeneous (similar) data items that are stored under one common name. Individual data item (array elements) in an array is identified by index or subscript enclosed in square brackets with array name.

- ✓ Array elements can be integers, floating point numbers and so on, but they must be the same type and same storage class.

Features of Array

- ✓ An array is a derived data type. It is used to represent a collection of elements of the same data type.
- ✓ Array elements are counted from 0 to size-1.
- ✓ The elements can be accessed with array name and the index. The index specifies the position of the element.
- ✓ The elements in an array are stored in continuous memory location. The starting memory location is represented by the array name and it is known as the base address of the Array.

Advantage of Array

1) Code Optimization: Less code to the access the data.

2) Easy to traverse data: By using the for loop, we can retrieve the elements of an array easily.

3) Easy to sort data: To sort the elements of array, we need a few lines of code only.

4) Random Access: We can access any element randomly using the array.

Types of array

Arrays can be classified into

- ✓ One Dimensional Array
- ✓ Two Dimensional Array
- ✓ Multi Dimensional Array

ONE DIMENSIONAL ARRAY

If the array has only one subscript then it is called one dimensional or single dimensional array. An array is a collection of homogeneous (similar) data items that are stored under one common name.

Characteristics of One Dimensional Array

- ✓ Array size must be positive number.
- ✓ Array elements are counted from 0 to size-1.
- ✓ String arrays are terminated with null character ('\0').

Declaration of an Array

Arrays must be declared before they are used so that the compiler can allocate space for them in memory.

Syntax for array declaration

data type array_name [size];

- The data type specifies the array elements data type.
- Size indicates the maximum number of elements that can be stored in the array.

Example

float height [10];

The above array declaration represents the array name is height, we can store a maximum of 10 elements and the array elements are floating point data type.

Different data type declaration of array

1. int a[5];

a[0]	a[1]	a[2]	a[3]	a[4]

2. float b[5];

b[0]	b[1]	b[2]	b[3]	b[4]

3. char m[5];

m[0]	m[1]	m[2]	m[3]	m[4]

Different data type declaration of array

Array Initialization

The array elements can be initialized when they are declared otherwise they will take garbage values.

Compile time initialization

Arrays can be initialized at compile time.

Syntax:

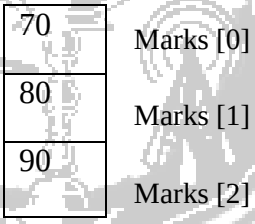
data type array_name [size] = {value 0, value 1, . . . , value n-1)

The initialized values are specified within curly braces separated by commas.

Example:

int Marks [3] = {70, 80, 90};

This statement declares the variable Marks as an array of 3 elements and will be assigned the values specified in list as below.



Array Initialization

Like ordinary variables, the values to the array can be initialized as follows.

```
int Marks[3];
Marks [0] = 70;
Marks [1] = 80;
Marks [2] = 90;
```

Character array can be initialized as follows:

```
char gender[2] = {'M','F'};
```

Runtime initialization

Arrays can be initialized at run time.

Example:

```
int a[2];
scanf("%d%d",&a[0],&a[1]);
```

Program: Calculate the average marks of the student

```
# include<stdio.h>
#include<conio.h>
void main()
{
        int m[5],i,sum=0,n;
        float avg;
        printf("enter number of subject \n");
        scanf("%d",&n);
        printf("enter marks \n");
        for(i=0;i<n;i++)
        {
                scanf("%d",&m[i]);
        }
        for(i=0;i<n;i++)
        sum=sum+m[i];
        avg=(float)sum/n;
        printf("average=%f",avg);
        getch()
}
```

Output:

```
Enter number of subject
5
Enter marks of students
55
60
78
85
90
Average=73.6
```