ARRAY

- Array is the collection of similar data types or collection of similar entity stored in memory location.
- Array of character is a string.
- Each data item of an array s called element.
- Each element is unique and located in separated memory location
- Each of elements of an array share a variable but each element having different index number known as subscript.
- An array can be a single dimensional or multi-dimensional and number of subscript determines it dimension.
- Number of subscripts is always starts with zero.
- One dimensional array is known as vector
- Two dimensional arrays are known as matrix.

Advantages:

- Array variable can store more than one value at a time when other variable can store one value at a time.

Example

```
int arr[100];
int mark[100];
```

Declaration of an Array

Its Syntax B

```
Data_type array name[size];
int arr[100];
int mark[100];
int a[5]={10,20,30,100,5}
```

The declaration of an array tells the compiler, that the data type, name of the array, size of the array and each element if occupies memory space.

We can represent individual array as,

```
Int arr[5];
```

arr[0], arr[1], arr[3], arr[4] symbolic constant can also be used to specify the size of the array as,

#define size 10;

Initialization of an array

- After declaration element of local array has garbage value
- If it is global or static array, then it will be automatically initialize with zero.

An explicity it can be initialize that

Data_type : array name[size] = {value 1,value 2, value 3…..}

Example

Int arr[5] = {20,60,90,100,120}

- Array subscript always start from zero which is known as lower bound and upper value is known as upper bound.
- Subscript can be expression i.e., integer value.

Accessing of Array element

/* Write a program to input values into an array and display them*/

```
#include<stdio.h>
int main()
{
int arr[5],i;
for(i=0;i<5;i++)
{
printf("Enter a value for arr[%d]\n",i);
scanf("%d",&arr[i]);
}
printf("The array elements are:\n");
for(i=0;i<5;i++)
{
print("%d",arr[i]);
}
return 0;
}
```

Output

Enter a value for arr[0] =12

Enter a value for arr[1] =45

Enter a value for arr[2] =59

Enter a value for arr[3] =98

Enter a value for arr[4] =21

The array elements are 12 45 59 98 21

Types of array

- ❖ Single Dimensional Array
- ❖ Two Dimensional Array

1. Single Dimensional Array

One dimensional array is used to store data in a linear form

Eg: int a[5]={1,2,3,4,5}

Array having on subscript value is called one dimensional array.

Eg: int a[5]={1,2,3,4,5}

It is called a linear array

Initialization of One Dimensional Array

We can initialize the one dimensional array into two types

1. Compiler time initialization
2. Run time initialization

Example 1:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a[5]={1,2,3,4,5};                    //Compile time initialization
int i;
clrscr();
printf("The values are:%d");
for(i=0;i<5;i++)
```

```
{
printf("%d",a[i]);
}
}
```

Output

1

2

3

4

5

Example 2:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a[5]={1,2,3,4,5};                    //Run time initialization
clrscr();
printf("Enter the values");
for(i=0;i<5;i++)
{
scanf("%d",a[i]);
}
Printf("The value are");
For(j=0;j<5;j++)
{
Printf("%d",a[j]);
}
}
```

Output

Enter the values : 1 2 3 4 5

The values are : 1 2 3 4 5

2. Two Dimensional Array

- Two Dimensional array is known as matrix.
- The array declaration in both the array i.e., In single dimensional array single subscript is used and in two dimensional array two subscripts are used.

Syntax:

Data_type array name[row][column

Example

int a[2][3];

Total no of elements = row*column is 2*3=6

It means the matrix consist of 2 rows and 3 columns.

Multi Dimensional Array

An array of arrays is called multi-dimensional array.

Syntax:

Data_type arrayName[1][2][3]….[n];

Example:

Int arr[3][3][3]

Row Column

3x3x3=27

= {1, 2, 3, 4, 5, 6, 7
8, 9, 10, 11, 12, 13
14, 15, 16, 17, 18, 19
20, 21, 22, 23, 24,
25,26,27}

Matrix 1 = $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$

Matrix 2 =

10	11	12
13	14	15
16	17	18

Matrix 3 =

19	20	21
22	23	24
25	26	27

Example:

	0	1	2	3
0	1	2	3	4
1	5	6	7	8
2	9	10	11	12

	0	1	2	3
0	13	14	15	16
1	17	18	19	20
2	21	22	23	24

```
int a= {{
        {1, 2, 3, 4},
        {5, 6, 7, 8}
        {9, 10, 11, 12},
       }
       {
       {13, 14, 15, 16}
       {17, 18, 19, 20}
       {21, 22, 23, 24}
       }
       };
```

If we want access 7

```
a[0]  [1]   [2];
```