LECTURE NOTE 19

ARRAYS

Introduction

A *data structure* is the way data is stored in the machine and the functions used to access that data. An easy way to think of a data structure is a collection of related data items. An *array* is a data structure that is a collection of variables of one type that are accessed through a common name. Each element of an array is given a number by which we can access that element which is called an index. It solves the problem of storing a large number of values and manipulating them.

Arrays

Previously we use variables to store the values. To use the variables we have to declare the variable and initialize the variable i.e, assign the value to the variable. Suppose there are 1000 variables are present, so it is a tedious process to declare and initialize each and every variable and also to handle 1000 variables. To overcome this situation we use the concept of array .In an Array values of same type are stored. An array is a group of memory locations related by the fact that they all have the same name and same type. To refer to a particular location or element in the array we specify the name to the array and position number of particular element in the array.

One Dimensional Array

Declaration:

Before using the array in the program it must be declared

Syntax:

data_type array_name[size];

data_type represents the type of elements present in the array.

array_name represents the name of the array.

Size represents the number of elements that can be stored in the array.

Example:

int age[100];

float sal[15];

char grade[20];

Here age is an integer type array, which can store 100 elements of integer type. The array sal is floating type array of size 15, can hold float values. Grade is a character type array which holds 20 characters.

Initialization:

We can explicitly initialize arrays at the time of declaration.

Syntax:

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

Value1, value2, valueN are the constant values known as initializers, which are assigned to the array elements one after another.

Example:

int marks[5]={10,2,0,23,4};

The values of the array elements after this initialization are:

marks[0]=10, marks[1]=2, marks[2]=0, marks[3]=23, marks[4]=4

NOTE:

1. In 1-D arrays it is optional to specify the size of the array. If size is omitted during initialization then the compiler assumes the size of array equal to the number of initializers.

Example:

int marks[]={10,2,0,23,4};

Here the size of array marks is initialized to 5.

2. We can't copy the elements of one array to another array by simply assigning it.

Example:

```
int a[5]={9,8,7,6,5};
int b[5];
b=a;      //not valid
```

we have to copy all the elements by using for loop.

```
for(a=i; i<5; i++)
    b[i]=a[i];
```

Processing:

For processing arrays we mostly use for loop. The total no. of passes is equal to the no. of elements present in the array and in each pass one element is processed.

Example:

```
#include<stdio.h>

main()

{

 int a[3],i;

 for(i=0;i<=2;i++)                    //Reading the array values

{

 printf("enter the elements");

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

}

 for(i=0;i<=2;i++)                    //display the array values

{

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

printf("\n");

}

}
```

This program reads and displays 3 elements of integer type.

Example:1

C Program to Increment every Element of the Array by one & Print Incremented Array.

```
#include <stdio.h>

void main()

{

    int i;

   int array[4] = {10, 20, 30, 40};

for (i = 0; i < 4; i++)

    arr[i]++;

 for (i = 0; i < 4; i++)

    printf("%d\t", array[i]);

}
```

Example: 2

C Program to Print the Alternate Elements in an Array

```
 #include <stdio.h>
  void main()
 {
    int array[10];
 int i, j, temp;
    printf("enter the element of an array \n");
 for (i = 0; i < 10; i++)
    scanf("%d", &array[i]);
    printf("Alternate elements of a given array \n");
 for (i = 0; i < 10; i += 2)
     printf( "%d\n", array[i]) ;
 }
```

Example-3

C program to accept N numbers and arrange them in an ascending order

```
#include <stdio.h>
void main()
{
 int i, j, a, n, number[30];
printf("Enter the value of N \n");
scanf("%d", &n);
printf("Enter the numbers \n");
for (i = 0; i < n; ++i)
          scanf("%d", &number[i]);
for (i = 0; i < n; ++i)
{
    for (j = i + 1; j < n; ++j)
    {
        if (number[i] > number[j])
        {
            a =number[i];
            number[i] = number[j];
            number[j] = a;
        }
    }
}
  printf("The numbers arranged in ascending order are given below \n");
for (i = 0; i < n; ++i)
       printf("%d\n", number[i]);
}
```