Programming in C

(1–D Array)

Prepared By

Alok Haldar

Assistant professor

Department of Computer Science & BCA

Kharagpur College

Arrays in C

In C language, `arrays` are reffered to as structured data types. An array is defined as **finite ordered collection of homogenous** data, stored in contiguous memory locations.

Here the words,

- **finite** *means* data range must be defined.
- **ordered** *means* data must be stored in continuous memory addresses.
- **homogenous** *means* data must be of similar data type.

An array is defined as the collection of similar type of data items stored at contiguous memory locations. Arrays are the derived data type in C programming language which can store the primitive type of data such as int, char, double, float, etc. It also has the capability to store the collection of derived data types, such as pointers, structure, etc. The array is the simplest data structure where each data element can be randomly accessed by using its index number.

Example where arrays are used,

- to store list of Employee or Student names,
- to store marks of students,
- or to store list of numbers or characters etc.

Since arrays provide an easy way to represent data, it is classified amongst the data structures in C. Other data structures in c are **structure**, **lists**, **queues**, **trees** etc. Array can be used to represent not only simple list of data but also table of data in two or three dimensions.

Advantage of C Array

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

2) Ease of traversing: By using the for loop, we can retrieve the elements of an array easily.

3) Ease of sorting: To sort the elements of the array, we need a few lines of code only.

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

Disadvantage of C Array

1) Fixed Size: Whatever size, we define at the time of declaration of the array, we can't exceed the limit. So, it doesn't grow the size dynamically like LinkedList which we will learn later.

Declaration of C Array

We can declare an array in the c language in the following way.

1. data_type array_name[array_size];

Now, let us see the example to declare the array.

1. Int marks[5];

Initialization of Array

The simplest way to initialize an array is by using the index of each element. We can initialize each element of the array by using the index. Consider the following example.

1. marks[0]=80;//initialization ofarray
2. marks[1]=60;
3. marks[2]=70;
4. marks[3]=85;
5. marks[4]=75;

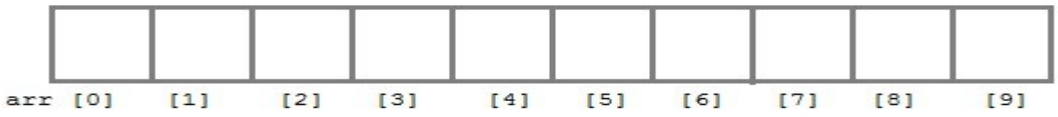
Declaring an Array

Like any other variable, arrays must be declared before they are used. General form of array declaration is,

```
data-type variable-name[size];
```

```
/* Example of array declaration */
```

```
int arr[10];
```



Here `int` is the data type, `arr` is the name of the array and 10 is the size of array. It means array `arr` can only contain 10 elements of `int` type.

Index of an array starts from `0` to **size-1** i.e first element of `arr` array will be stored at `arr[0]` address and the last element will occupy `arr[9].`

Initialization of an Array

After an array is declared it must be initialized. Otherwise, it will contain **garbage** value(any random value).

The general form of initialization of array is,

```
data-type array-name[size] = { list of values };
```

```
/* Here are a few examples */
int marks[4]={ 67, 87, 56, 77 };    // integer array initialization
```

```
float area[5]={ 23.4, 6.8, 5.5 };   // float array initialization
```

```
int marks[4]={ 67, 87, 56, 77, 59 };    // Compile time error
```

One important thing to remember is that when you will give more initializer(array elements) than the declared array size than the **compiler** will give an error.

```
#include<stdio.h>

void main()
{
    int i;
    int arr[] = {2, 3, 4};      // Compile time array initialization
    for(i = 0 ; i < 3 ; i++)
    {
        printf("%d\t",arr[i]);
    }
}
```

2 3 4

To initialize arrays with user specified values.

Example,

```
#include<stdio.h>

void main()
{
    int arr[4];
    int i, j;
    printf("Enter array element");
    for(i = 0; i < 4; i++)
    {
        scanf("%d", &arr[i]);    //Run time array initialization
    }
    for(j = 0; j < 4; j++)
    {
        printf("%d\n", arr[j]);
    }
}
```

Here, int is the *data_type*, marks are the *array_name*, and 5 is the *array_size*.

Example of Array

```
#include<stdio.h>
int main(){
int i=0;
int marks[5];//declaration of array
marks[0]=80;//initialization of array
marks[1]=60;
marks[2]=70;
marks[3]=85;
marks[4]=75;
//traversal of array
for(i=0;i<5;i++){
printf("%d \n",marks[i]);
}//end of for loop
return 0;
}
```

Output

```
80
60
70
85
75
```

Array Example: Sorting an array

In the following program, we are using bubble sort method to sort the array in ascending order.

```
#include<stdio.h>
void main()
{
int i,j,temp;
int a[10]={10,9,7,101,23,44,12,78,34,23};
for(i=0;i<10;i++)
{
   for(j=i+1;j<10;j++)
  {
      if(a[j]>a[i])
     {
        temp=a[i];
        a[i]=a[j];
        a[j]=temp;
      }
  }
}
printf("Printing Sorted Element List...\n");
for(i=0;i<10;i++)
{
printf("%d\n",a[i]);
}
}
```

Program to print the largest and second largest element of the array.

```
#include<stdio.h>
void main()
{
int arr[100],i,n,largest,sec_largest;
printf("Enter the size of the array?");
scanf("%d",&n);
printf("Enter the elements of the array?");
for(i=0;i<n;i++)
{
scanf("%d",&arr[i]);
}
largest=arr[0];
sec_largest=arr[1];
for(i=0;i<n;i++)
{
    if(arr[i]>largest)
   {
      sec_largest=largest;
      largest=arr[i];
    }
    else if(arr[i]>sec_largest&&arr[i]!=largest)
          {
               sec_largest=arr[i];
          }
  }
   printf("largest=%d, second largest=%d",largest,sec_largest);
}
```