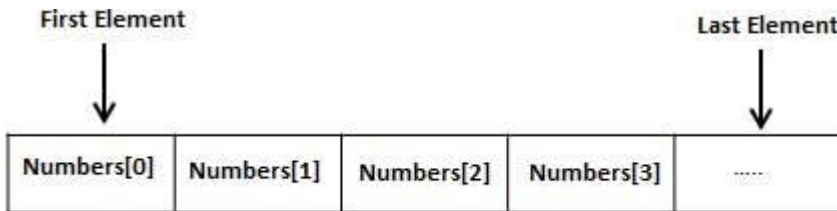


## ARRAYS IN C LANGUAGE

**Need of arrays:** Suppose we want to find largest among 100 numbers and print the largest one, for that we should have 100 variables and also there must be some means through which we can remember the position at which the largest number exists in the list of 100. This becomes tedious by use of 100 variables. To overcome that we use array.

**Definition:** It is a data structure that can store a fixed-size sequential collection of elements of the same type. Instead of declaring individual variables, such as number0, number1, ..., and number99, you declare one array variable such as numbers and use numbers[0], numbers[1], and ..., numbers[99] to represent individual variables. A specific element in an array is accessed by an index.

All arrays consist of contiguous memory locations. The lowest address corresponds to the first element and the highest address to the last element.

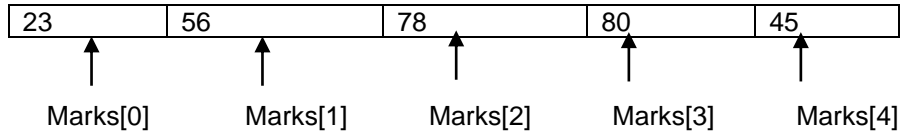


It is a contiguous memory allocation. The data type remains the same. It's static in nature. Once size of array is declared during declaration (compilation) it remains fixed, we cannot increase or decrease it during the execution of program.

**Declaration:** `data type nameofarray[size];`

↑                    ↑                    ↑  
char/int/float    identifier    number of elements stored in an array (either 0 or any integer)

Example: `int Marks[5];` Here marks is the name of array that can store 5 integer values.



Memory representation of an array of 5 elements

**Initializing arrays:** Assigning values to the elements of an array.

```
int marks[5]={23,56,78,80,45}; //for above example
```

```
float B[2]={2.8,6.0}; // here B[0]=2.8; B[1]=6.0;
```

```
char name[4]='A','M','I','T'; // Here name[0]=A, name[1]=M, name[2]=I and name[3]=T;
```

### Storing values in an array



Initializing the elements during declaration

Input values for the elements from the keyboard

Assign values to individual elements

**1. Initializing the elements during declaration**

```
int marks[5]={23,56,78,80,45};
```

**2. Input values for the elements from the keyboard**

```
int i,marks[5];  
for(i=0;i<5;i++)  
    scanf("%d",&marks[i]);
```

### 3. Assign values to individual elements

```
int arr[10],i;
for(i=0;i<10;i++) //fill array with even numbers
arr[i]=i*2;
```

### Accessing elements of an array:

Example 1:

```
#include <stdio.h>
int main ()
{
    //main begins

    int n[ 10 ]; /* n is an array of 10 integers */
    int i,j;

    /* read elements of array */
    for ( i = 0; i < 10; i++ )
    {
        scanf("%d",&n[i]);
    }
    /* output each array element's value */
    for (j = 0; j < 10; j++ )
    {
        printf("Element[%d] = %d\n", j, n[j] );
    }
    return 0;
}
//end of main
```

Example 2

```
#include<stdio.h>
void main()
{ //main begins

    int i;
    char name[4];

    /* read elements of array */
    for ( i = 0; i < 10; i++ )
    {
        scanf("%c",&name[i]);
    }
    /* output each array element's value */
    for (j = 0; j < 10; j++ )
    {
        printf("Element[%d] = %c\n", j, name[j] );
    }

}
//end of main
```

### Operations on Arrays:

1. Traversing an array: means accessing each element of an array.

Example 1 and 2 above

Example 3: Write a program to find largest among n numbers.

```
#include<stdio.h>
void main()
{
    int i, n, arr[20],Large;
    printf("Enter the number of elements in an array:");
    scanf("%d",&n);
    printf("\nRead elements of the array:");
    for(i=0;i<n;i++)
        scanf("%d",&arr[i]);

    Large=arr[0];
    for(i=1;i<n;i++)
    {
        if(Large<arr[i])
            Large=arr[i];
    }
    printf("\nLargest number is %d",large);
}
}
```

2. Inserting an element in an array

The element can be inserted either at the beginning, middle or at the end of the existing array.

Example: Write a program to insert value at a given location in an array.

```
#include<stdio.h>
void main()
{
    int i, n, arr[10],p,num;
    printf("Enter the number of elements in an array:");
    scanf("%d",&n);
    printf("\nRead elements of the array:");
    for(i=0;i<n;i++)
        scanf("%d",&arr[i]);

    printf("\nEnter the number you want to insert in array:");
    scanf("%d",&num);
    printf("\nEnter the position where you want to insert element num");
    scanf("%d",&p);

    for(i=n-1;i>=p;i--)
        arr[i+1]=arr[i];
    arr[p]=num;
    n++;
    printf("\nThe array after insertion of %d is",num);
    for(i=0;i<n;i++)
        printf("\n%d",arr[i]);

} // end of main
```

### 3. Deleting an element from an array

Similarly, the element can be deleted either at the beginning, middle or at the end of the existing array.

Example: Write a program to delete an element from a given location in an array.

```
#include<stdio.h>
void main()
{
    int i, n, arr[10],p,num;
    printf("Enter the number of elements in an array:");
    scanf("%d",&n);
    printf("\nRead elements of the array:");
    for(i=0;i<n;i++)
        scanf("%d",&arr[i]);

    printf("\nEnter the position from where you want to delete an element from array:");
    scanf("%d",&p);
    for(i=p;i<n-1;i++)
        arr[i]=arr[i+1];
    n--;
    printf("\nThe array after deletion :");
    for(i=0;i<n;i++)
        printf("\narr[%d]=%d",i,arr[i]);

} // end of main
```

#### 4. Searching an element in an array

It means to find whether a particular value is present in the array or not. If it is present then search is successful and the search process gives the location of that value in the array, otherwise displays that value is not present in the array. In our example we are using Linear Search.

Example: Write a program to find whether value is present in the array or not.

```
#include<stdio.h>
void main()
{
    int i, n, a[10],p=-1,num,F=0;
    printf("Enter the number of elements in an array:");
    scanf("%d",&n);
    printf("\nEnter the elements of the array:");
    for(i=0;i<n;i++)
        scanf("%d",&a[i]);

    printf("\nEnter the number you want to search in array:");
    scanf("%d",&num);

    for(i=0;i<n;i++)
    {
        if( a[i]==num)
        {
            F=1;
            p=i;
            printf("\n%d is found in the array at position %d",num,i);
            break; //comes out of the loop
        }
    }
    //end of for loop

    if(F==0)
        printf("\n%d not found in the array",num);
}
// end of main
```

#### 5. Sorting an array in ascending or descending order

It is arranging elements of an array either in ascending or descending order. In our example we are using Bubble sort.

```
#include<stdio.h>
void main()
{
    int i, j,n, a[20],temp;
    printf("Enter the number of elements in an array:");
    scanf("%d",&n);
    printf("\nRead elements of the array:");
    for(i=0;i<n;i++)
        scanf("%d",&a[i]);

    for(i=0;i<n-1;i++)
```

```
{
  for(j=0;j<n-i-1;j++)
  {
    if( a[j]>a[j+1];
    {
      temp = a[j];
      a[j]=a[j+1];
      a[j+1]=temp;
    }
  }
}
printf("\nThe array sorted in ascending order:");
for(i=0;i<n;i++)
  printf("\n%d",a[i]);

} // end of main
```

### References:

1. [https://www.tutorialspoint.com/cprogramming/c\\_arrays.htm](https://www.tutorialspoint.com/cprogramming/c_arrays.htm)
2. Book Computer Fundamentals and Programming in C by Reema Thareja
3. Book B.S. Gottfried, Schaum's Outline of Theory and Problems of Programming with C, McGraw-Hill