

**FACULTY OF ENGINEERING AND TECHNOLOGY
UNIVERSITY OF LUCKNOW
LUCKNOW**



**Computer System and Programming in 'C'
CS-101/201**

**Er. Zeeshan Ali Siddiqui
Assistant Professor
Deptt. of C.S.E.**

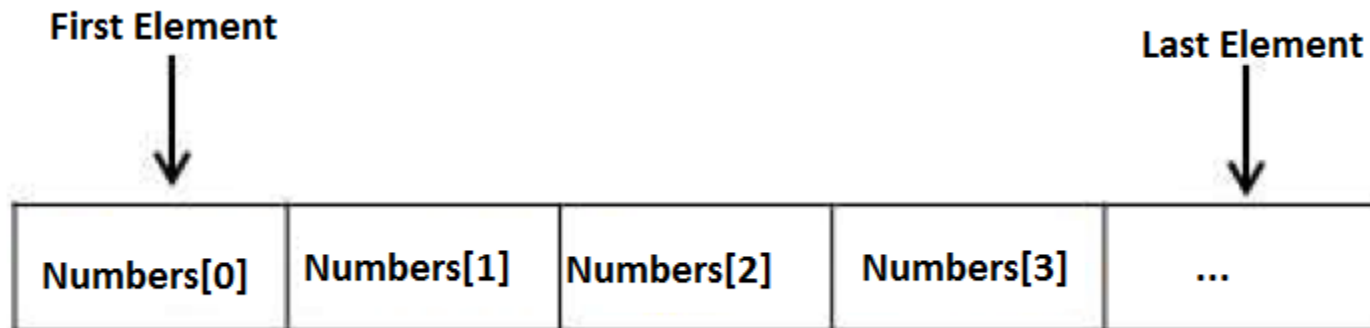
ARRAYS

Arrays-Overview^{1/2}

- An array is a *finite* collection of *similar* data type values stored in *adjacent* memory locations.

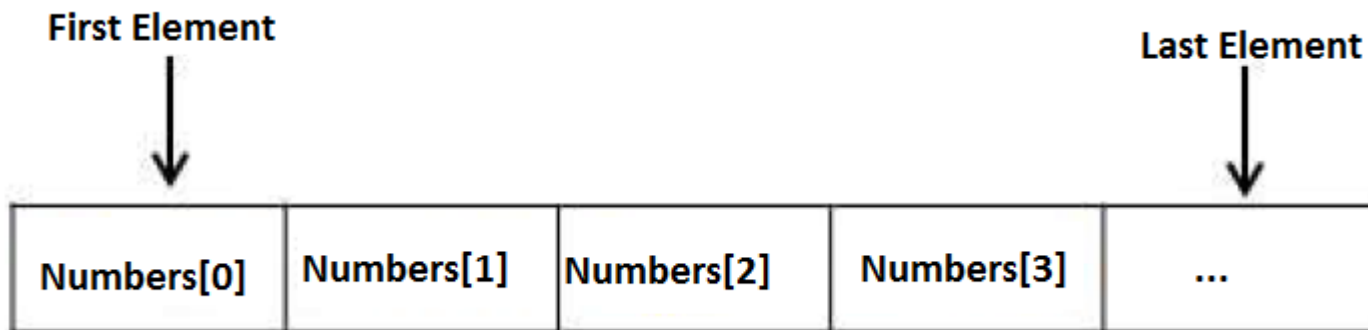
OR

- An array may be think of a list of values that are arranged consecutively in the memory.



Arrays-Overview_{2/2}

- 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.

Types of Arrays

- We have two types of arrays:
 1. One-dimensional arrays
 2. Multidimensional arrays

ONE-DIMENSIONAL ARRAYS

Declaration of 1 D Arrays^{1/2}

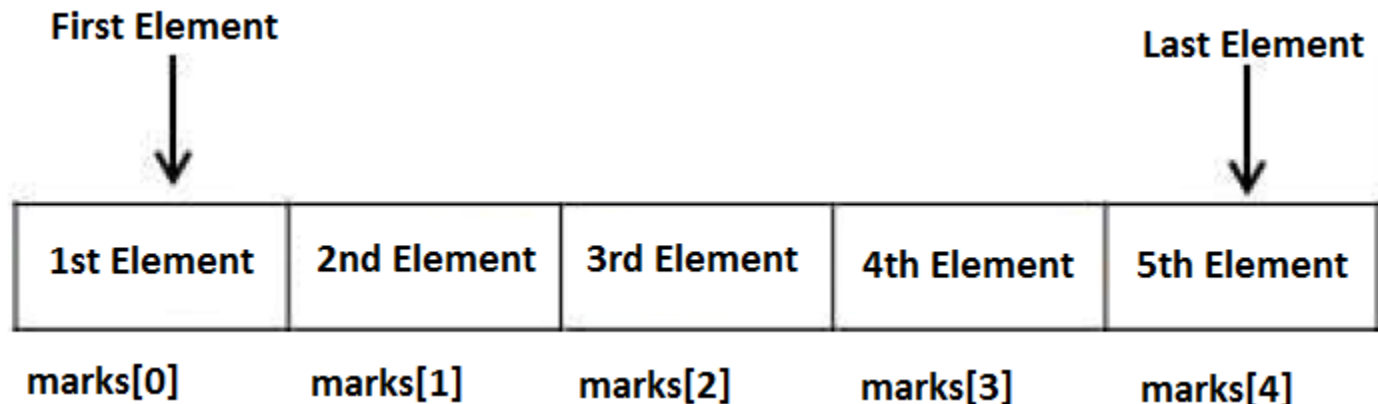
- **Syntax:**
- `data_type array_name[max_size];`

Where:

- **data_type:** what sort of values it can store, for example int, char, float, double
- **array_name:** name given by the user to identify the array
- **max_size:** maximum number of values that the array can hold.

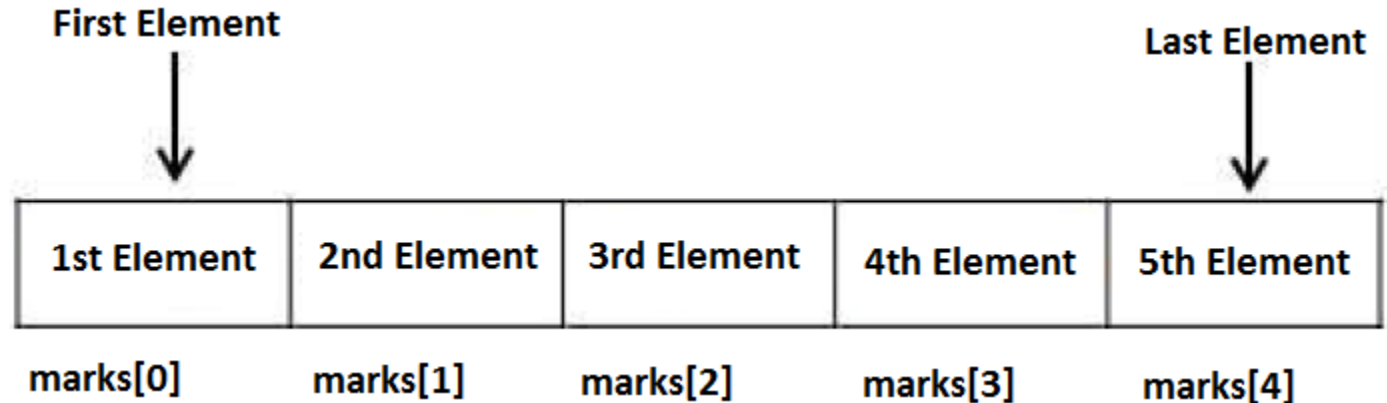
- **Example:**

- `int marks[5];`



Declaration of 1 D Arrays^{2/2}

- `int marks[5];`



- In an array of size N, the index ranges from **0 to N-1**.
- C language does not allow declaring an array whose *number of elements is not known at the compile time*. Therefore:

- `int marks[];`

OR

- `int marks[N];` (where value of N is given at runtime)

Illegal declaration of an array.

Initialization of 1 D Arrays

- Arrays may be initialized when they are declared.

```
int marks[5]={12, 5, 66, 3, 88};
```

- An array may be partially initialized, by providing fewer data items than the size of the array. The remaining array elements will be automatically initialized to zero.

```
int marks[10]={12, 5, 66, 3, 88};
```

- If an array is to be completely initialized, the dimension of the array is not required. If the size of the array is not given, then the largest initialized position determines the size of the array.

```
int marks[]={12, 5, 66, 3, 88};
```

Sample Program-1

```
//Program to print elements of an array  
int main()  
{  
    int marks[5]={12, 5, 66, 3, 88}, i;  
    for(i=0;i<5;i++)  
    {  
        printf("%d\t",marks[i]);  
    }  
    return 0;  
}
```

- Output

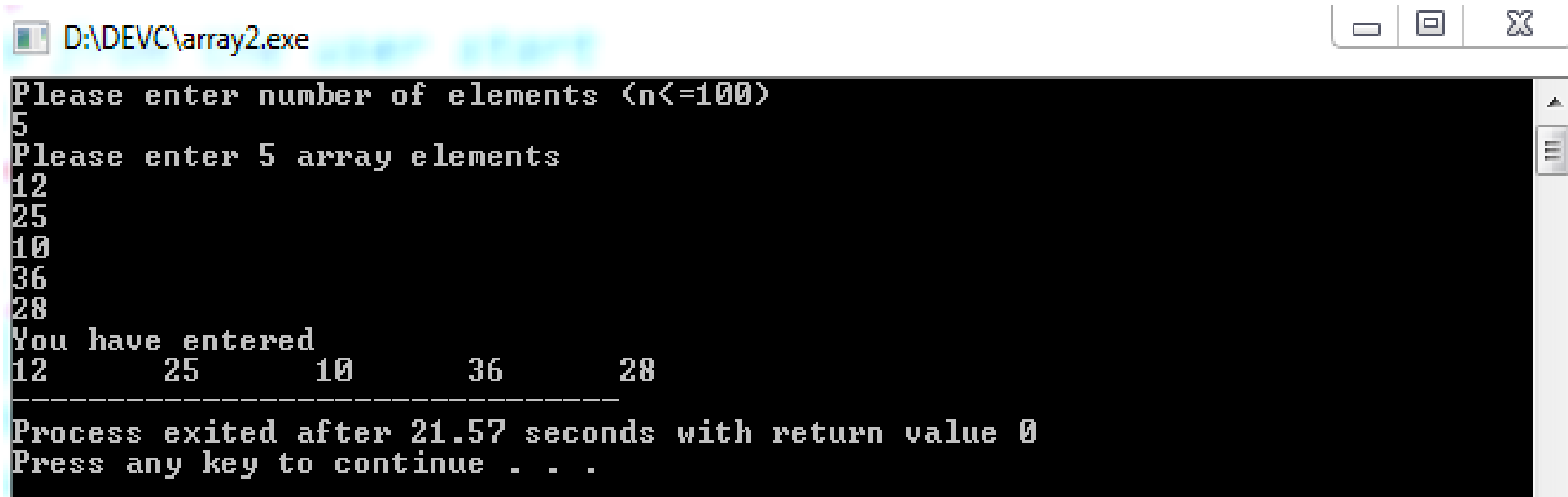


```
D:\DEV\array1.exe  
12      5      66      3      88  
-----  
Process exited after 0.005199 seconds with return value 0  
Press any key to continue . . .
```

Sample Program-2

```
/*Program to take array elements from the user  
and print those elements*/  
int main()  
{  
    int marks[100],n,i;  
    printf("Please enter number of elements (n<=100)\n");  
    scanf("%d",&n);  
    //Taking values from the user start  
    printf("Please enter %d array elements\n",n);  
    for(i=0;i<n;i++)  
    {  
        scanf("%d",&marks[i]);  
    }  
    //Taking values from the user end  
    //Printing values from the user start  
    printf("You have entered\n");  
    for(i=0;i<n;i++)  
    {  
        printf("%d\t",marks[i]);  
    }  
    //Printing values from the user end  
    return 0;  
}
```

Output of Sample Program-2



```
D:\DEV\array2.exe
Please enter number of elements (n<=100)
5
Please enter 5 array elements
12
25
10
36
28
You have entered
12      25      10      36      28
-----
Process exited after 21.57 seconds with return value 0
Press any key to continue . . .
```

Sample Program-3

```
/*Program to take array elements from the user and print
maximum, minimum and average of array elements*/
int main()
{
    int marks[100],n,i,max,min,sum=0; float avg;
    printf("Please enter number of elements (n<=100)\n");
    scanf("%d",&n);
    //Taking values from the user start
    printf("Please enter %d array elements\n",n);
    for(i=0;i<n;i++)
    {
        scanf("%d",&marks[i]);
    }
    //Taking values from the user end
    max=min=sum=marks[0];
    for(i=1;i<n;i++)
    {
        if(marks[i]>max)
            max=marks[i];

        if(marks[i]<min)
            min=marks[i];

        sum=sum+marks[i];
    }
    avg=(float) sum/n;
    printf("Max=%d, Min=%d, and Average=%.2f",max,min,avg);
    return 0;
}
```

Output of Sample Program-3

D:\DEV\array3...exe



```
Please enter number of elements (n<=100)
```

```
5
```

```
Please enter 5 array elements
```

```
12
```

```
25
```

```
10
```

```
35
```

```
24
```

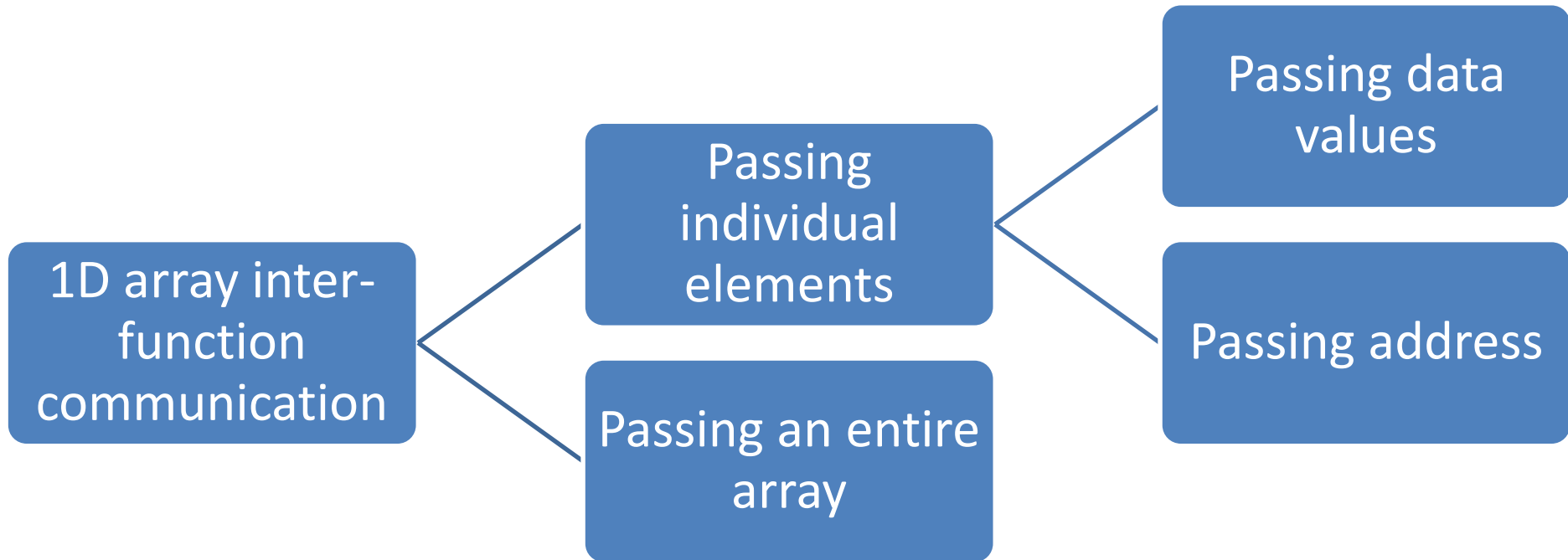
```
Max=35, Min=10, and Average=21.20
```

```
-----  
Process exited after 15.22 seconds with return value 0
```

```
Press any key to continue . . .
```

PASSING ARRAYS TO FUNCTIONS

1D array inter-function communication



Passing individual elements

Passing data values: Example

```
void display(int num)
{
    printf("%d", num);
}
int main()
{
    int data[]={12,25,36};
    display(data[1]); //Passing array element data[1] only.
    return 0;
}
```

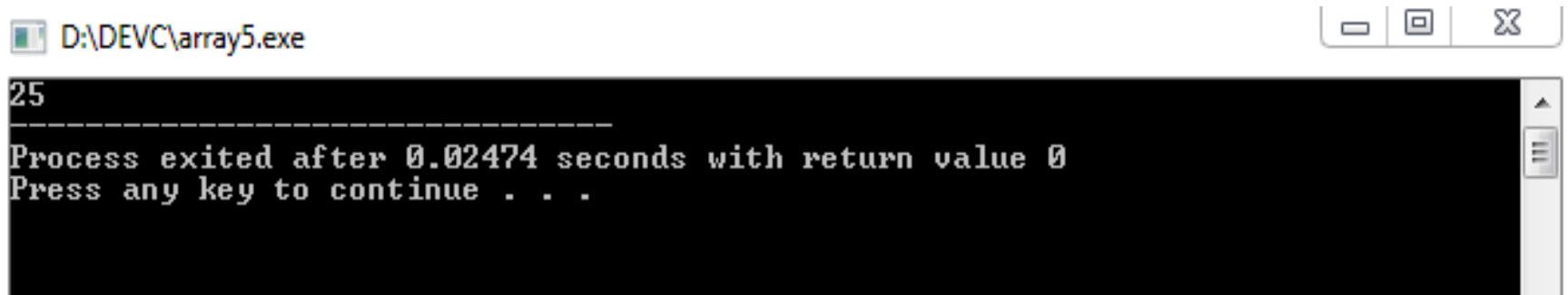
D:\DEV\array4.exe

25

Process exited after 0.005179 seconds with return value 0
Press any key to continue . . .

Passing address: Example

```
void display(int *num)
{
    printf("%d",*num);
}
int main()
{
    int data[]={12,25,36};
    display(&data[1]); //Passing address of array element data[1].
    return 0;
}
```



```
D:\DEV\array5.exe
25
-----
Process exited after 0.02474 seconds with return value 0
Press any key to continue . . .
```

Passing an entire array

Passing an entire array: Example

*/*C program to pass an array containing marks of a student to a function. This function will return average marks and display the average marks in main function.*/*

```
float avgmarks(float a[]);
int main()
{
    float avg, marks[]={32.4, 45, 19.6, 2.50, 10.2, 33};
    avg=avgmarks(marks); /* Only name of array is passed as argument. */
    printf("Average age=%.2f",avg);
    return 0;
}
float avgmarks(float a[])
{
    int i; float avg, sum=0.0;
    for(i=0;i<6;++i)
    {
        sum+=a[i];
    }
    avg =(sum/6);
    return avg;
}
```

D:\DEV\array6.exe

Average age=23.78

Process exited after 0.005514 seconds with return value 0

Press any key to continue . . .

MULTI-DIMENSIONAL ARRAYS

Multi-Dimensional Array

- Multi-dimensional array are known as array of arrays.
- To access a particular element from the array we have to use two subscripts:
 - one for **row** number and
 - other for **column** number.
- The notation is of the form **array[i][j]** where i stands for row subscripts and j stands for column subscripts. The array holds **$i*j$** elements.
- Suppose there is a multidimensional array **array[p][q][r]**. Then this array can hold $p*q*r$ numbers of data.
- In the same way, the array of any dimension can be initialized in C programming.

Declaration of 2 D Arrays

- **Syntax:**

- `data_type array_name[row_size][column_size];`

Where:

- **data_type:** what sort of values it can store, for example int, char, float, double
- **array_name:** name given by the user to identify the array
- **row_size:** maximum number of rows.
- **column_size:** maximum number of columns.

- **Example:**

- `int data[3][3];`

Initialization of 2 D Arrays

- Examples:

1. `int marks[2][3] = {48, 47, 35, 29, 44, 37};`
2. `int marks[2][3] = {{48, 47, 35}, {29, 44, 37}};`
3. `int marks[][3] = {{48, 47, 35}, {29, 44, 37}};`

- To initialize entire two-dimensional array to zero

```
int marks[2][3] = {0};
```

- If some values are missing in the initializer then it is automatically set to zero.

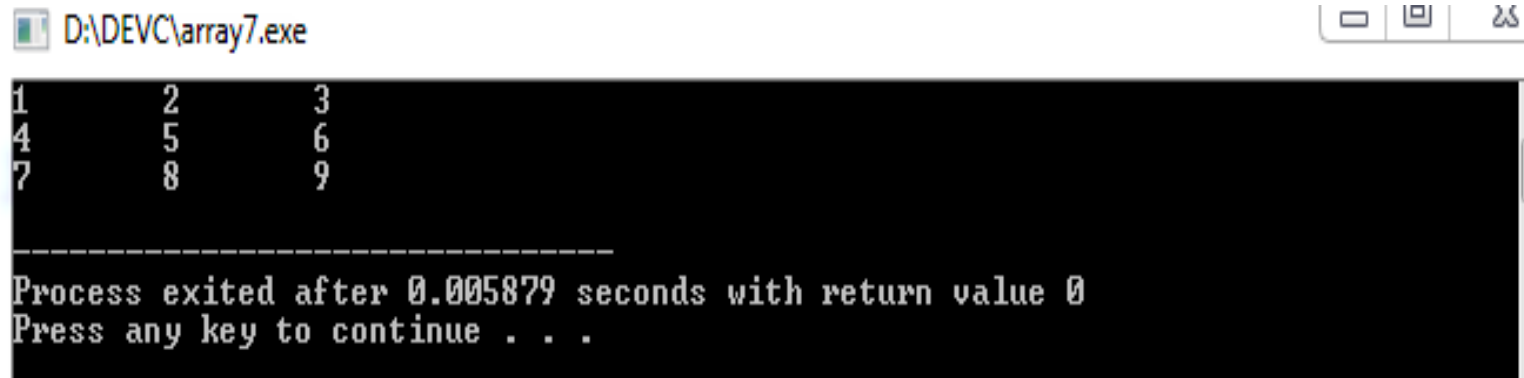
```
int marks[2][3] = {{48, 47, 35}};
```

Here, values in the first row will be initialize but the elements in the second row will be initialized to zero.

Sample Program-1

```
//Program to print the elements of a 2D array
int main()
{
    int data[3][3]={{1,2,3},{4,5,6},{7,8,9}};
    int i,j;
    for(i=0;i<3;i++)
    {
        for(j=0;j<3;j++)
        {
            printf("%d\t",data[i][j]);
        }
        printf("\n");
    }
    return 0;
}
```

- Output



```
D:\DEV\array7.exe
1      2      3
4      5      6
7      8      9
-----
Process exited after 0.005879 seconds with return value 0
Press any key to continue . . .
```

Sample Program-2

```
//Program to read and display the elements of a 3*3*3 array
int main()
{
    int data[3][3][3],i,j,k;
    printf("Please enter elements of the matrix\n");
    for(i=0;i<3;i++)
    {
        for(j=0;j<3;j++)
        {
            for(k=0;k<3;k++)
            {
                scanf("%d",&data[i][j][k]);
            }
        }
    }
    printf("You have entered\n");
    for(i=0;i<3;i++)
    {
        for(j=0;j<3;j++)
        {
            for(k=0;k<3;k++)
            {
                printf("%d\t",data[i][j][k]);
            }
            printf("\n");
        }
        printf("\n");
    }
    return 0;
}
```

Output of Sample Program-2

```
D:\DEV\array8.exe
Please enter elements of the matrix
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
You have entered
1      2      3
4      5      6
7      8      9

10     11     12
13     14     15
16     17     18

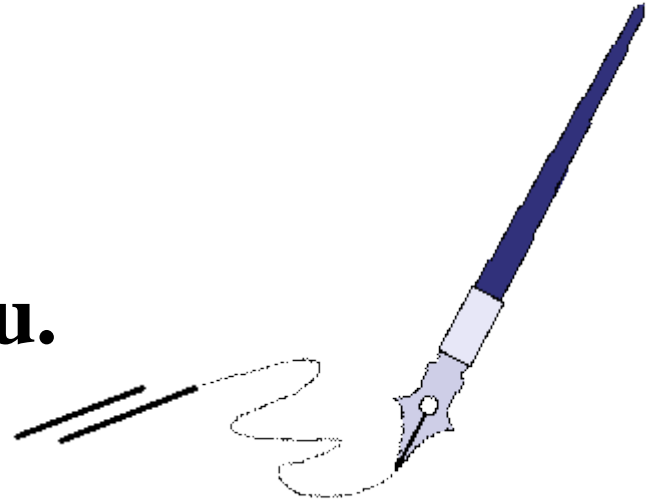
19     20     21
22     23     24
25     26     27

-----
Process exited after 22.36 seconds with return value 0
Press any key to continue . . .
```

Exercise

- What are merits and demerits of arrays? Also, discuss n-dimensional array.
- WAP that takes elements of the 1 D array from the user and finds the sum of these elements.
- WAP to search an element in a 1 D array using Linear Search.
- WAP to add and multiply two matrices of order NxN.
- WAP that finds the sum of diagonal elements of an mxn matrix.
- WAP that inputs two 2-D arrays and saves sum of corresponding elements of these arrays in a third array and prints them.
- WAP to find the minimum and maximum element of the array.
- What happens when an array with a specified size is assigned
 - With values more than the specified size.
 - With values less than the specified size.

Thank You.



BTQ

BTQ: Brain Teaser Question

What is a word made up of 4 letters, yet is also made up of 3. Sometimes is written with 9 letters, and then with 4. Rarely consists of 6, and never is written with 5.

