# *e-Lecture* on C Programming

for

B.Sc. Second Semester
Paper code: 201

Dr. Rajesh Kumar Goutam
Assistant Professor
Department of Computer Science
University of Lucknow

# Arrays

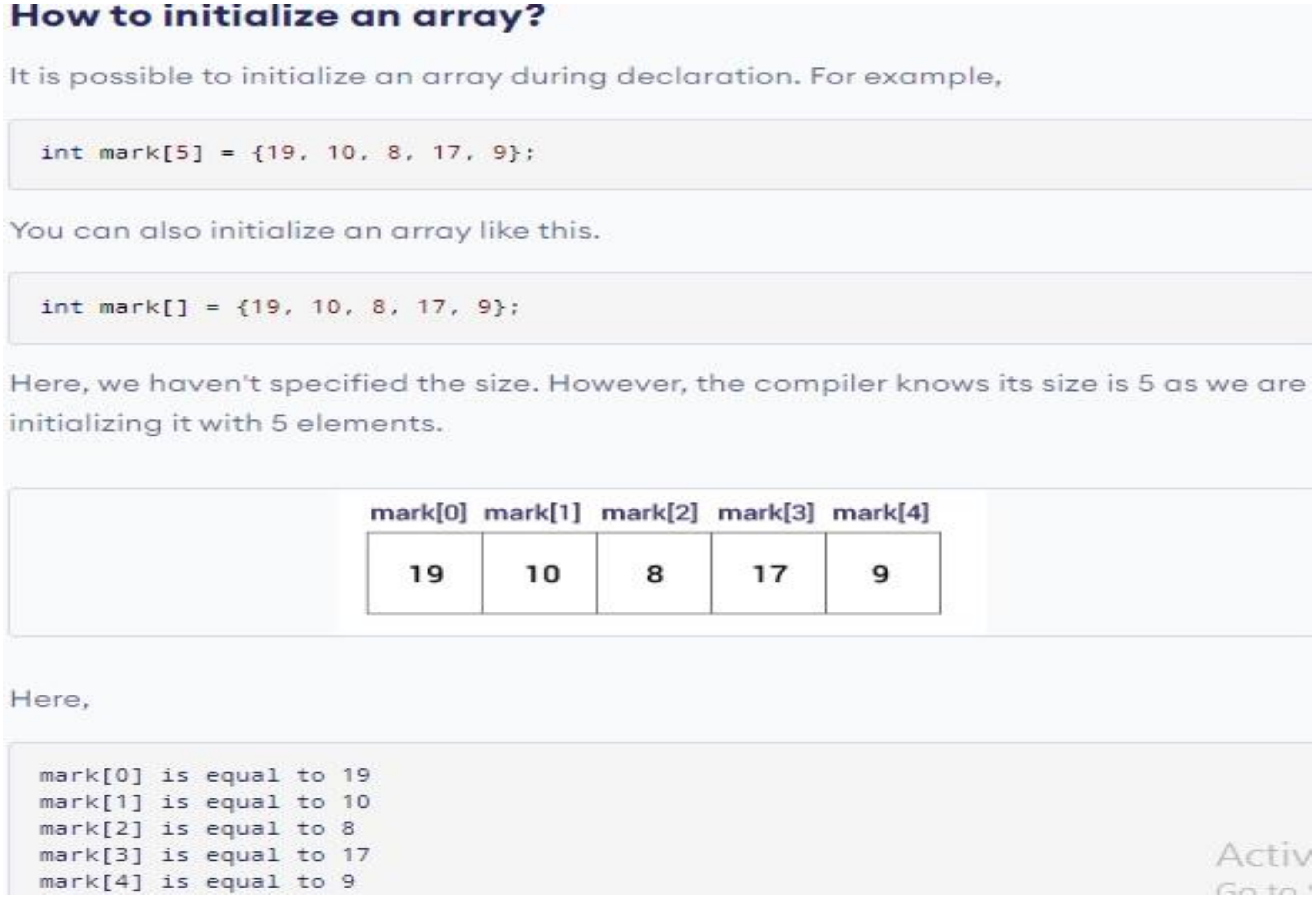
- The array is a data structure in C programming, which can store a fixed-size sequential collection of elements of the same data type.
- An array is a collection of data items, all of the same type, accessed using a common name.
- A one-dimensional array is like a list; A two dimensional array is like a table;

- Syntax:

```
type arrayName [ size ];
```

- This is called a one-dimensional array. An array type can be any valid C data types, and array size must be an integer constant greater than zero.

# Array Initialization



## How to initialize an array?

It is possible to initialize an array during declaration. For example,

```
int mark[5] = {19, 10, 8, 17, 9};
```

You can also initialize an array like this.

```
int mark[] = {19, 10, 8, 17, 9};
```

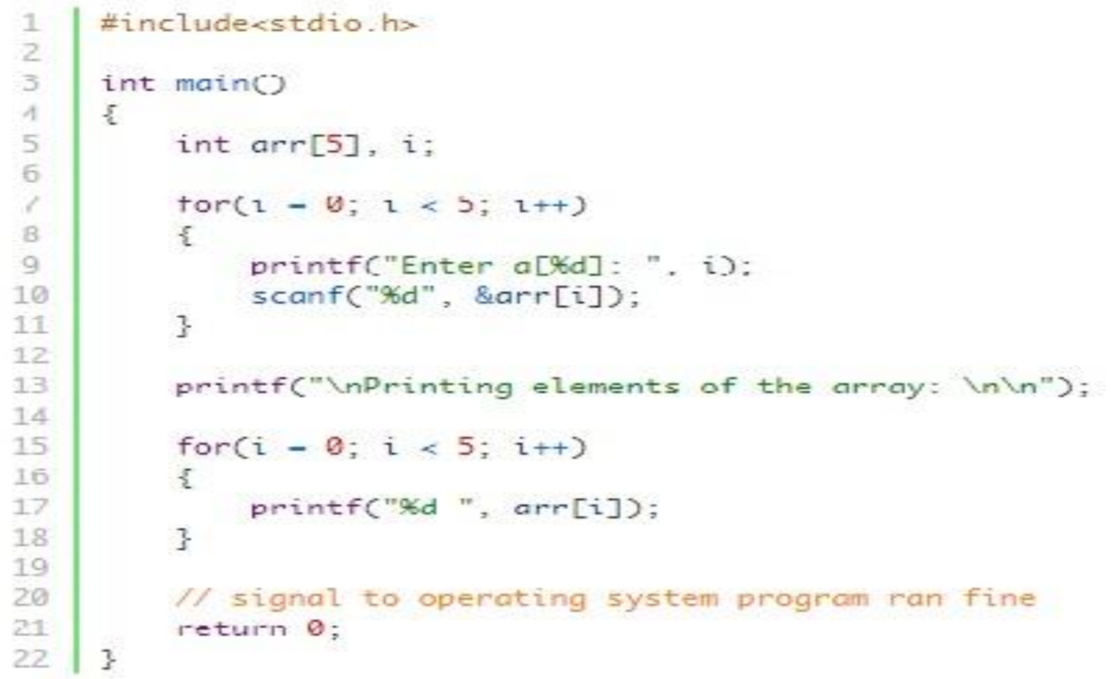
Here, we haven't specified the size. However, the compiler knows its size is 5 as we are initializing it with 5 elements.

| mark[0] | mark[1] | mark[2] | mark[3] | mark[4] |
|---|---|---|---|---|
| 19 | 10 | 8 | 17 | 9 |

Here,

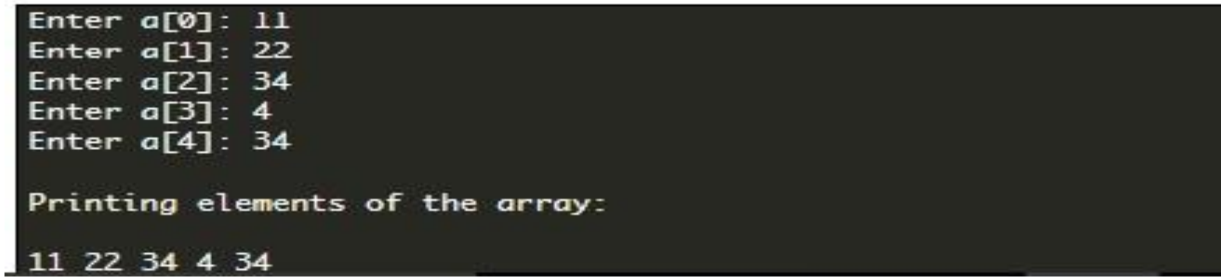
```
mark[0] is equal to 19
mark[1] is equal to 10
mark[2] is equal to 8
mark[3] is equal to 17
mark[4] is equal to 9
```

# Array Example



```c
#include<stdio.h>

int main()
{
    int arr[5], i;

    for(i = 0; i < 5; i++)
    {
        printf("Enter a[%d]: ", i);
        scanf("%d", &arr[i]);
    }

    printf("\nPrinting elements of the array: \n\n");

    for(i = 0; i < 5; i++)
    {
        printf("%d ", arr[i]);
    }

    // signal to operating system program ran fine
    return 0;
}
```

**Expected Output:**



```
Enter a[0]: 11
Enter a[1]: 22
Enter a[2]: 34
Enter a[3]: 4
Enter a[4]: 34

Printing elements of the array:

11 22 34 4 34
```

# Two Dimensional Array

- Two Dimensional Array in C is the simplest form of Multi-Dimensional Array. In C Two Dimensional Array, data is stored in row and column wise
- *Declaration of Two Dimensional Array in*

Data_Type  Array_Name[Row_Size][Column_Size]

*Data_type:*

This will decide the type of elements will accept by two dimensional array in C. For example, If we want to store integer values then we declare the Data Type as int, If we want to store Float values then we declare the Data Type as float etc

*Array_Name:*

This is the name you want to give it to this C two dimensional array. For example students, age, marks, employees etc

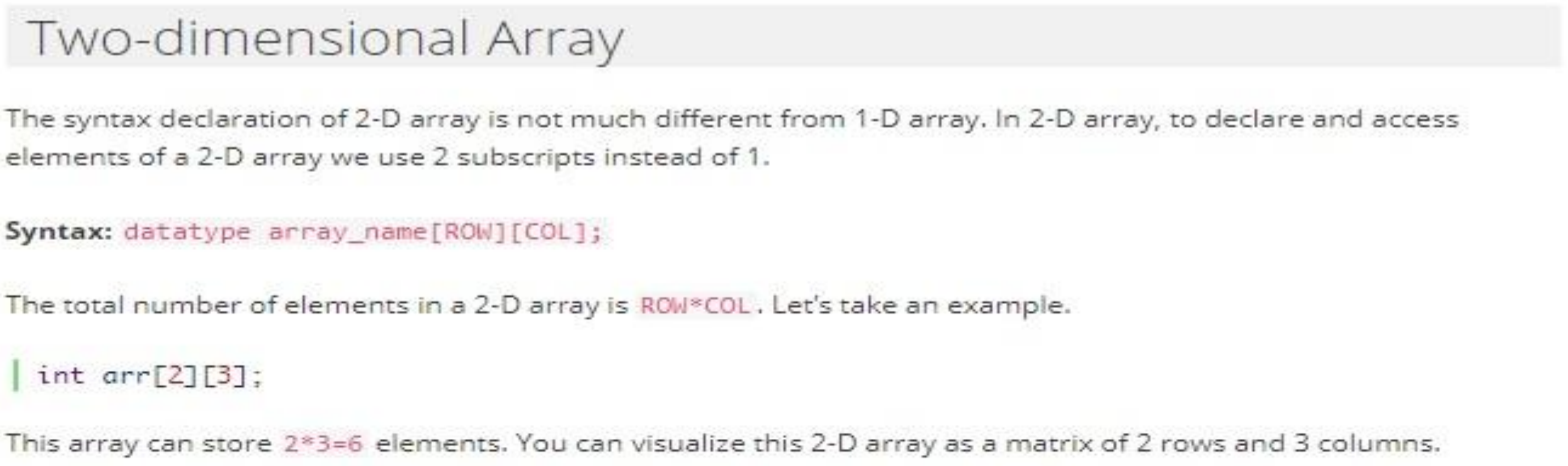
*Row_Size:*

Number of Row elements an array can store. For example, Row_Size =10, then the array will have 10 rows.

*Column_Size:*

Number of Column elements an array can store. For example, Column_Size = 8, the array will have 8 Columns.

# Two Dimensional Array



## Two-dimensional Array

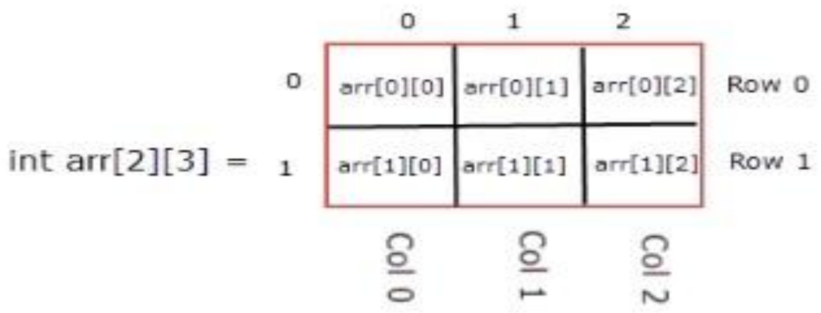
The syntax declaration of 2-D array is not much different from 1-D array. In 2-D array, to declare and access elements of a 2-D array we use 2 subscripts instead of 1.

**Syntax:** `datatype array_name[ROW][COL];`

The total number of elements in a 2-D array is `ROW*COL`. Let's take an example.

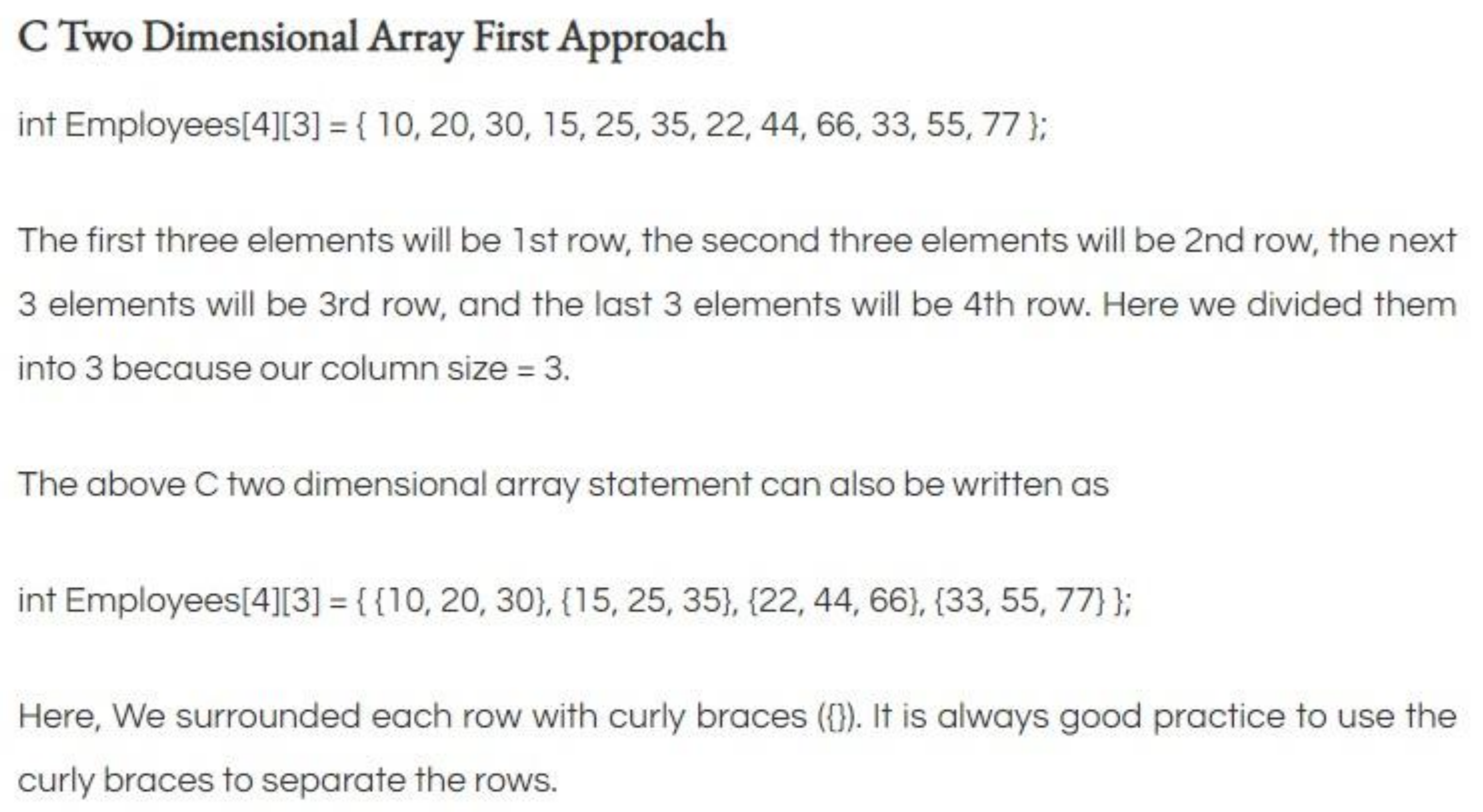
```
int arr[2][3];
```

This array can store `2*3=6` elements. You can visualize this 2-D array as a matrix of 2 rows and 3 columns.



*A conceptual representation of 2-D array*

# Two Dimensional Array Declaration



## C Two Dimensional Array First Approach

```
int Employees[4][3] = { 10, 20, 30, 15, 25, 35, 22, 44, 66, 33, 55, 77 };
```

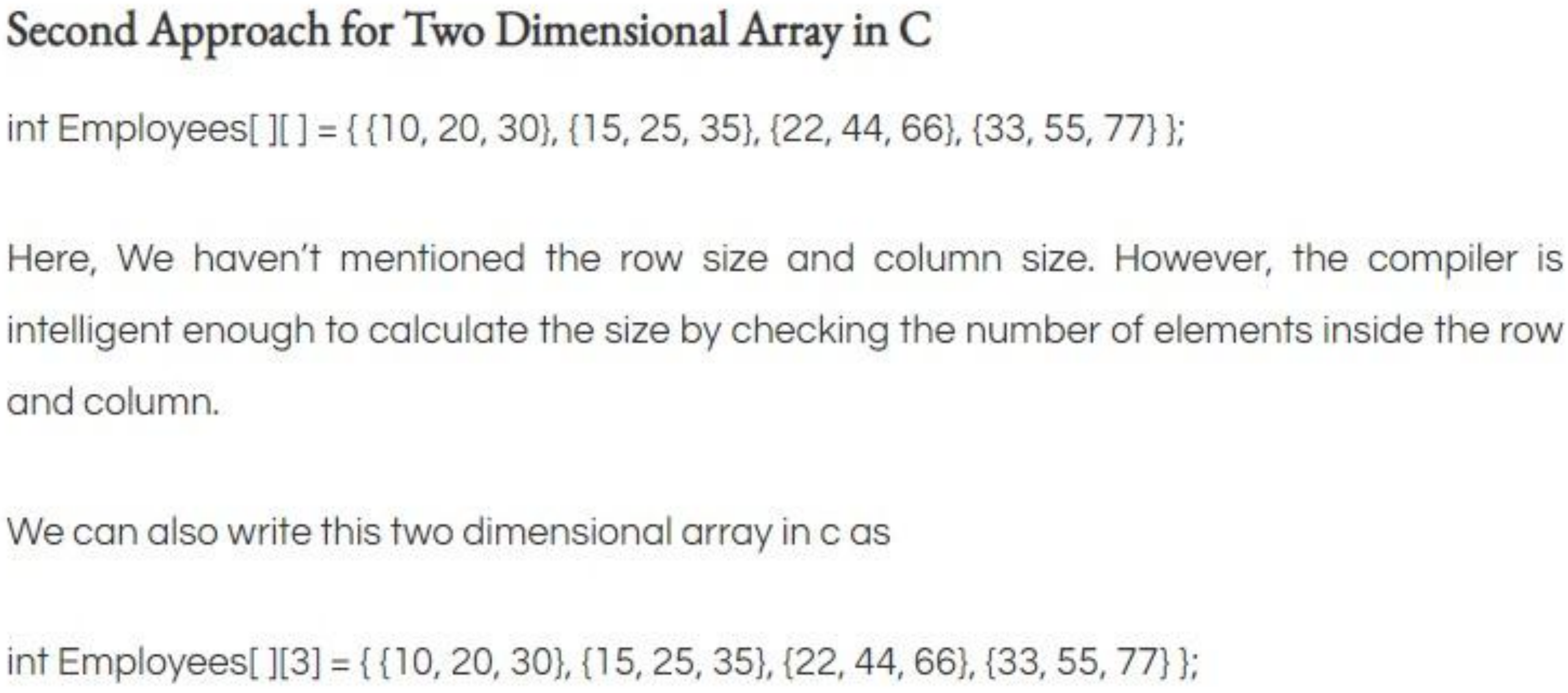
The first three elements will be 1st row, the second three elements will be 2nd row, the next 3 elements will be 3rd row, and the last 3 elements will be 4th row. Here we divided them into 3 because our column size = 3.

The above C two dimensional array statement can also be written as

```
int Employees[4][3] = { {10, 20, 30}, {15, 25, 35}, {22, 44, 66}, {33, 55, 77} };
```

Here, We surrounded each row with curly braces ({}). It is always good practice to use the curly braces to separate the rows.

# Two Dimensional Array Declaration



## Second Approach for Two Dimensional Array in C

```
int Employees[ ][ ] = { {10, 20, 30}, {15, 25, 35}, {22, 44, 66}, {33, 55, 77} };
```

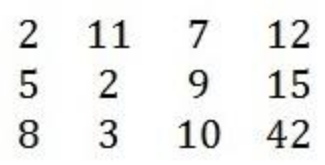
Here, We haven't mentioned the row size and column size. However, the compiler is intelligent enough to calculate the size by checking the number of elements inside the row and column.

We can also write this two dimensional array in c as

```
int Employees[ ][3] = { {10, 20, 30}, {15, 25, 35}, {22, 44, 66}, {33, 55, 77} };
```

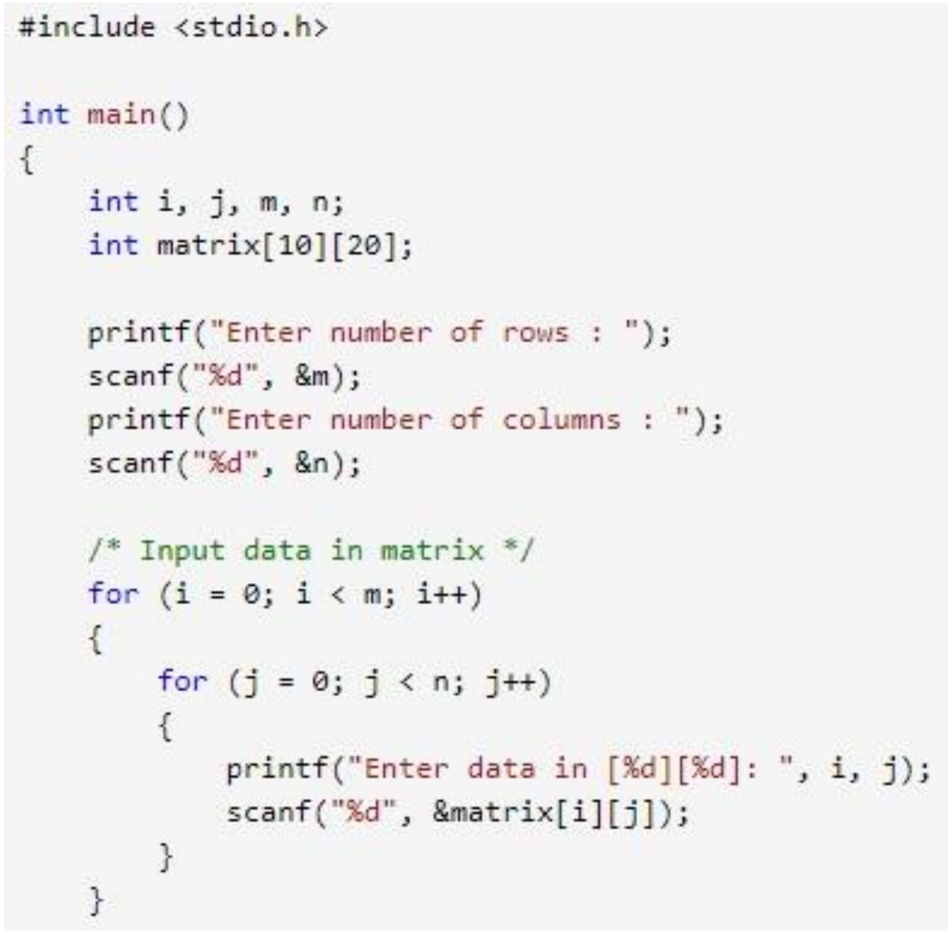
# C Program

For example matrix of size 3 x 4 should display like this:



```
2  11   7  12
5   2   9  15
8   3  10  42
```

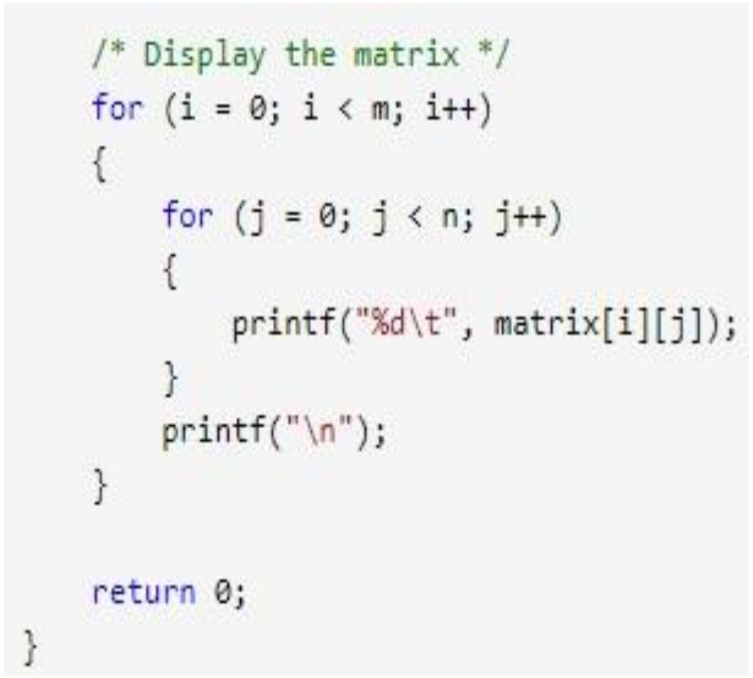
**For taking element**



```c
#include <stdio.h>

int main()
{
    int i, j, m, n;
    int matrix[10][20];

    printf("Enter number of rows : ");
    scanf("%d", &m);
    printf("Enter number of columns : ");
    scanf("%d", &n);

    /* Input data in matrix */
    for (i = 0; i < m; i++)
    {
        for (j = 0; j < n; j++)
        {
            printf("Enter data in [%d][%d]: ", i, j);
            scanf("%d", &matrix[i][j]);
        }
    }
```

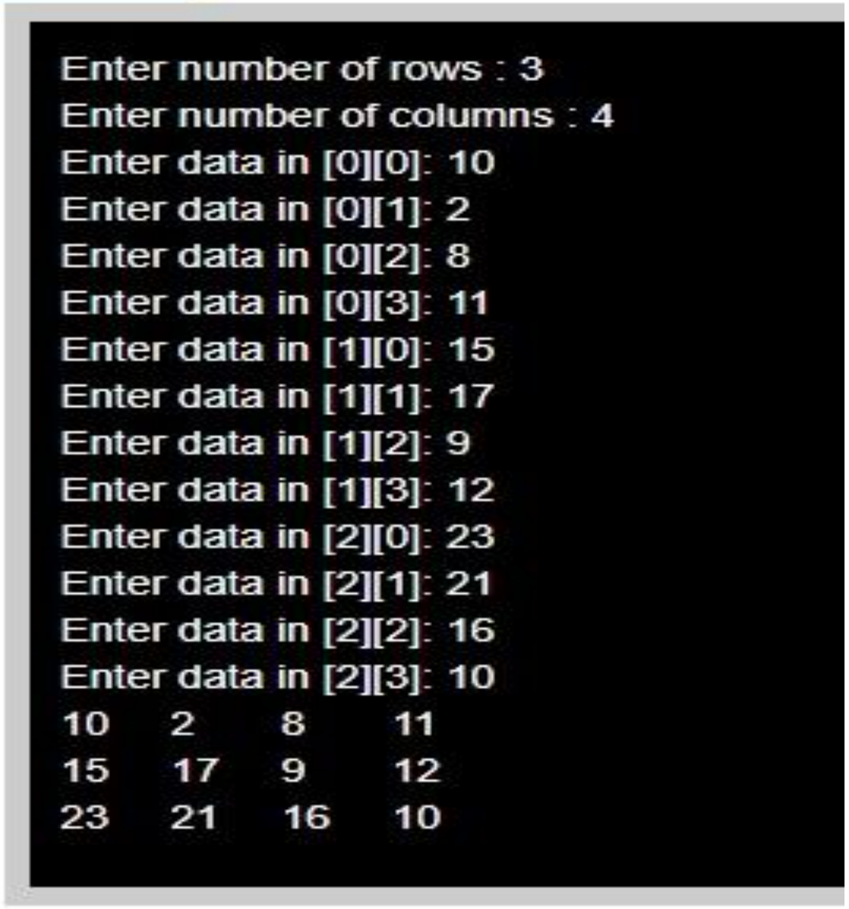
# C Program

**Printing Matrix**



```c
    /* Display the matrix */
    for (i = 0; i < m; i++)
    {
        for (j = 0; j < n; j++)
        {
            printf("%d\t", matrix[i][j]);
        }
        printf("\n");
    }

    return 0;
}
```

Output



```
Enter number of rows : 3
Enter number of columns : 4
Enter data in [0][0]: 10
Enter data in [0][1]: 2
Enter data in [0][2]: 8
Enter data in [0][3]: 11
Enter data in [1][0]: 15
Enter data in [1][1]: 17
Enter data in [1][2]: 9
Enter data in [1][3]: 12
Enter data in [2][0]: 23
Enter data in [2][1]: 21
Enter data in [2][2]: 16
Enter data in [2][3]: 10
10      2       8       11
15      17      9       12
23      21      16      10
```

# Thanks

Dear Students

If you have queries, Please feel free to contact me at

e-Mail: rajeshgoutam82@gmail.com
Mobile No: 9453838526