



(<https://www.educba.com/software-development/>)

← (<https://www.educba.com/3d-arrays-in-c/>)

→ (<https://www.educba.com/array-functions-in-c/>)

Multidimensional Array in C

	Column 0	Column 1	Column 2
Row 0	[0][0]	[0][1]	[0][2]
Row 1	[1][0]	[1][1]	[1][2]
Row 2	[2][0]	[2][1]	[2][2]

www.educba.com

Introduction to Multidimensional Array in C

This article focuses on the multidimensional array (<https://www.educba.com/multidimensional-array-in-c/>) in c which is predominantly used in computer and research analysis. Generally, an array linearly focuses a piece of information which is said to be one-dimensional. Since a one-dimensional array stores data only single information like regno of the students. In some situations, it



is necessary to store data in a table format that comprises rows and columns or to handle



(<https://www.educba.com/software-development/>)

How to declare a multidimensional array in C?

Start Your Free Software Development Course

Web development, programming languages, Software testing & others

Syntax:

The general declaration of Multidimensional array is given as:

```
type name [ size] [size]..... N;
```

- Here, data type name – It denotes the type of elements (integer, float).
- Array name – Denotes name assigned to the dimensional array.
- Row-size – No. of row elements ex. row-size = 8 , then array has 8 rows.
- Column- size – No. of column elements.

How to Initialize the Multidimensional Array in C?

The size of the [multidimensional arrays \(https://www.educba.com/powershell-multidimensional-array/\)](https://www.educba.com/powershell-multidimensional-array/) is predicted by multiplying the size of various dimensions. And they store values in the form of two ways like row-major and column-major. And the memory allocation validates both length and rank properties.

In C, Multidimensional array has three types:



1. Two-dimensional array



(<https://www.educba.com/software-development/>)

Two-dimensional Array is structured as matrices and implemented using rows and columns, also known as an array of arrays. The memory allocation is done either in row-major and column-major. And the default format is Row-Major. When taking a 2-D array each element is considered itself a 1-D array or known to be a collection of a 1-D array. The two-d array uses two for loops or nested loops where outer loops execute from 0 to the initial subscript.

Syntax:

```
type array name [ no. of rows] [ no. of Columns];
```

Example:

```
int td [4][3];
```

here 4 is the no. of rows and 3 is the no. of columns.

Initialization of Two-Dimensional Array

Initialization in the 2-D array is done in multiple ways, it is shown here.

```
int m [3][2] = {{10,3} {4,2} {6,4} {5,4} {3,4}};  
int di [2][4] = {10,5,8,12,45,13,11,61};
```

🔗 Popular Course in this category





(<https://www.educba.com/software-development/>)

★★★★★ 4.5 (8,612 ratings)

Course Price

\$79 ~~\$399~~

[View Course](#)

(<https://www.educba.com/software-development/courses/c-programming-course/?btnz=edu-blg-inline-banner1>)

Related Courses

C++ Training (4 Courses, 5 Projects, 4 Quizzes) (<https://www.educba.com/software-development/courses/c-course/?btnz=edu-blg-inline-banner1>)

Java Training (40 Courses, 29 Projects, 4 Quizzes) (<https://www.educba.com/software-development/courses/java-course/?btnz=edu-blg-inline-banner1>)

Here, we have mentioned the no. of rows and columns in the box. It is mandatory to assign the second index to make understand compiler about the ending and start of the row. The below table shows the memory allocation of the 2-D array.

	Column 0	Column 1	Column 2
Row 0	[0][0]	[0][1]	[0][2]
Row 1	[1][0]	[1][1]	[1][2]
Row 2	[2][0]	[2][1]	[2][2]

The number of elements is determined by manipulating a number of rows and columns and multiplying no. of rows and columns respectively. For instance, the no. of elements an array holds B [-2...4, -3.6]. It is calculated by Lower bound and upper bound.





[\(https://www.educba.com/software-development/\)](https://www.educba.com/software-development/)

Implementation

It is done using Row major and column-major Implementations

Row-Major:

The formula for address manipulation is given as:

$$= B + W [n(I-1) + [J-1]]$$

Where b- is the base address and n- No. of columns for W bytes.

Column Major:

$$= B + W [r(j-1) + [i-1]]$$

where r – is the no. of rows.

Examples of Two-Dimensional Array

Examples of Two-Dimensional Array are :

Example #1

Each element of an array A [-10..10, 20...35] needs 1 byte of memory. And the array fits Column major at the address 400, Find the location of A [0,30].



Solution



(<https://www.educba.com/software-development/>)

Address A [0,30] = 400 + 1[(0 - (-10) + 21(30 - 20))] = 400 + (10 + 21 * 10)
= 400 + (10 + 210) = 620

A familiar operation performed in the 2-d array is Algebra of matrices with m * n Matrix of B. The mathematical concept of the matrix is implemented the same as in programming.

The below example stores an element in the matrix format and prints the same.

Code:

```
#include<stdio.h>

int main ()
{
    int a[3][4], i, j;
    for(i = 0; i < 3; i++)
    {
        for(j = 0; j < 4; j++)
        {
            printf("Enter arr[%d][%d]: ", i, j);
            scanf("%d", &a[i][j]);
        }
    }

    printf("\nEntered 2-D array is: \n\n");
    for(i = 0; i < 3; i++)
    {
        for(j = 0; j < 4; j++)
```





[\(https://www.educba.com/software-development/\)](https://www.educba.com/software-development/)

```
}  
return 0;  
}
```

Output:

```
Enter arr[0][0]: 4  
Enter arr[0][1]: 5  
Enter arr[0][2]: 2  
Enter arr[0][3]: 1  
Enter arr[1][0]: 5  
Enter arr[1][1]: 8  
Enter arr[1][2]: 5  
Enter arr[1][3]: 6  
Enter arr[2][0]: 3  
Enter arr[2][1]: 4  
Enter arr[2][2]: 5  
Enter arr[2][3]: 9  
  
Entered 2-D array is:  
  
4   5   2   1  
5   8   5   6  
3   4   5   9
```

Example #2

C program performing the sum of two matrices.

Code:





[\(https://www.educba.com/software-development/\)](https://www.educba.com/software-development/)

```
int s;
printf("Enter number of Rows :");
scanf("%d",&r);
printf("Enter number of Cols :");
scanf("%d",&c);
printf("\nEnter matrix elements :\n");
for(i=0;i< r;i++)
{ for(j=0;j< c;j++)
{
printf("Enter the number of inputs [%d,%d] : ",i+1,j+1);
scanf("%d",&mat[i][j]);
}
}
printf("\n");
for(i=0;i< r;i++)
{
s=0;
for(j=0;j< c;j++)
{
printf("%d\t",mat[i][j]);
s+=mat[i][j];
}
printf("\tSUM : %d",s);

printf("\n");
```





(<https://www.educba.com/software-development/>)

two identical matrices. Through for loop, it takes two input matrix and loops to accept matrix.

Output:

```
Enter number of Rows : 2
Enter number of Cols : 3

Enter matrix elements :
Enter the number of inputs [1,1] : 4
Enter the number of inputs [1,2] : 5
Enter the number of inputs [1,3] : 7
Enter the number of inputs [2,1] : 8
Enter the number of inputs [2,2] : 9
Enter the number of inputs [2,3] : 1

4      5      7      SUM : 16
8      9      1      SUM : 18
```

Example #3

Transpose of a Matrix

Interchanging rows and columns to form a new matrix which is known as the transpose of a matrix.

Example:

```
2  5  3
7  11 4
12 16 10
```

Then Transpose give,

```
2  7  12
5  11 16
```





[\(https://www.educba.com/software-development/\)](https://www.educba.com/software-development/)

```
#include<stdio.h>

int main()
{
    int T[5][5],i,j,a,b;
    printf(" No.of rows?");
    scanf("%d",&a);
    printf("No.of columns?");
    scanf("%d",&b);
    printf("\nEnter the elements in matrix:\n");
    for(i=0;i<b;++i)
    for(j=0;j<a;++j)
    scanf("%d",&T[i][j]);
    printf("\nTranspose matrix is given as:\n");
    for(i=0;i<b;++i)
    {
        for(j=0;j<a;++j)
        printf("%d ",T[j][i]);
        printf("\n");
    }
    return 0;
}
```



In the above program To read a matrix we had used two for loops and to print its transpose the



(<https://www.educba.com/software-development/>)

```
No.of rows? 3
No.of columns? 3

Enter the elements in matrix:
1 3 5
6
7 8
3
9
4

Transpose matrix is given as:
1 6 3
3 7 9
5 8 4
```

2. Three-Dimensional Array

It is Called an Array of Array Elements or An Array of Matrices. It's Quite a Buzzy One but Once You Get Practice Towards the Logic It Makes Easier to Implement. and This 3-D Array Requires More than Three Dimensions and Requires the Bulk of Memory to Store.

It can be declared as:

```
data_type array_name [table name] [ no. of row] [ no. of column]
int L[m][n] [p];
```

int L [3][4][2]; Here the array L can hold 24 elements. And all these can be initialized during the compilation process, but when uninitialized there are put into a garbage value.

Initialization can be done in the same way as a two-dimensional array. Here is a sample,



```
int L [3][4][2] = { { { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24 } } }
```



[\(https://www.educba.com/software-development/\)](https://www.educba.com/software-development/)

Here are some examples of the three-dimensional array which are given below:

Example #1

Below comes a simple example in C programming illustrating (<https://www.educba.com/career-in-c-programming/>) three-dimensional Array. It is done using for a loop by considering 3 for loops for 3d elements.

Code:

```
#include <stdio.h>

void main()
{
    printf("three dimensional array!\n\n");
    int i, j, k, s[2][1][2], siz;
    siz=2*1*2;
    printf("Enter %d elements: \n",siz);
    for(i = 0; i < 2; ++i)
    {
        for (j = 0; j < 1; ++j)
        {
            for(k = 0; k < 2; ++k )
            {
                scanf("%d", &s[i][j][k]);
            }
        }
    }
}
```





(<https://www.educba.com/software-development/>)

```
{
for(k = 0; k < 2; k++)
{
printf("sample[%d][%d][%d] = %d\n", i, j, k, s[i][j][k]);
}
}
}
}
```

Output:

```
three dimensional array!
Enter 4 elements:
5
6
7
2
The stored values are:
sample[0][0][0] = 5
sample[0][0][1] = 6
sample[1][0][0] = 7
sample[1][0][1] = 2
```

Example #2

Another example of a 3-D array to print elements automatically.

Code:





[\(https://www.educba.com/software-development/\)](https://www.educba.com/software-development/)

```
{ {1,2}, {3,5}, {6,5} },
{ {8,3}, {8,7}, {9,11} }
};
for (int i = 0; i <2; ++i)
{
    for (int j = 0; j <3; ++j)
    {
        for (int k = 0; k <2; ++k)
            printf("Value at m[%d][%d][%d] = %d\n", i, j, k, m[i][j][k]);
    }
}
return 0;
}
```

Output:

```
Value at m[0][0][0] = 1
Value at m[0][0][1] = 2
Value at m[0][1][0] = 3
Value at m[0][1][1] = 5
Value at m[0][2][0] = 6
Value at m[0][2][1] = 5
Value at m[1][0][0] = 8
Value at m[1][0][1] = 3
Value at m[1][1][0] = 8
Value at m[1][1][1] = 7
Value at m[1][2][0] = 9
Value at m[1][2][1] = 11
```



3. Four-Dimensional Array



[\(https://www.educba.com/software-development/\)](https://www.educba.com/software-development/)

Type array name [1][2][3][4] [n] where 1,2 denotes the dimensions and n implies nth dimensions.

Example:

```
int state [5][6][7][8];
```

Example of Four-dimensional array

C program to implement 4- D array.

Code:

```
#include <stdio.h>
int main()
{
    int i, j, k, l, s;
    int d[2][2][2][2];
    s = 2;
    d[0][0][0][0] = 4;
    d[0][0][0][1] = 3;
    d[0][0][1][0] = 2;
    d[0][0][1][1] = 6;
    d[0][1][0][0] = 6;
    d[0][1][0][1] = 8;

    d[0][1][1][0] = 1;
```





[\(https://www.educba.com/software-development/\)](https://www.educba.com/software-development/)

```
d[1][0][1][1] = 1;
d[1][1][0][0] = 9;
d[1][1][0][1] = 7;
d[1][1][1][0] = 5;
d[1][1][1][1] = 7;
for (i = 0; i < s; i++) {
    for (j = 0; j < s; j++) {
        for (k = 0; k < s; k++) {
            for (l = 0; l < s; l++) {
                printf("Value of stdio[%d][%d][%d][%d]: %d ", i, j, k, l, d[i][j][k][l]);
                printf("\n");
            }
        }
    }
}
return 0;
}
```

Output:

```
Value of stdio[0][0][0][0]: 4
```





(<https://www.educba.com/software-development/>)

```
Value of stdio[1][0][0][0]: 6
Value of stdio[1][0][0][1]: 9
Value of stdio[1][0][1][0]: 5
Value of stdio[1][0][1][1]: 1
Value of stdio[1][1][0][0]: 9
Value of stdio[1][1][0][1]: 7
Value of stdio[1][1][1][0]: 5
Value of stdio[1][1][1][1]: 7
```

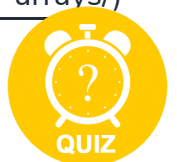
Conclusion

To the end, in this article, we discussed [multidimensional arrays](https://www.educba.com/multidimensional-array-in-python/) and their subtypes in C programming. And also, their declaration and accessing the elements in a matrix format. These techniques are applied in the concept like binary searching and sorting implementation. Here an index plays a key role in as they specify an element in the array structure.

Recommended Articles

This is a guide to Multidimensional Array in C. Here we discuss how to initialize the multidimensional array in C along with examples. You may also look at the following articles to learn more-


1. [Best C Compilers](https://www.educba.com/best-c-compilers/)
2. [2D Arrays in C#](https://www.educba.com/2d-arrays-in-c-sharp/)
3. [2-D Arrays in C](https://www.educba.com/2-d-arrays-in-c/)
4. [C# Multidimensional Arrays](https://www.educba.com/c-sharp-multidimensional-arrays/)



C PROGRAMMING TRAINING (3 COURSES) 5



[\(https://www.educba.com/software-development/\)](https://www.educba.com/software-development/)

 34+ Hours

☒ Verifiable Certificate of Completion

☒ Lifetime Access

Learn More

[_https://www.educba.com/software-development/courses/c-programming-course/?btnz=educba-inline-banner3\)](https://www.educba.com/software-development/courses/c-programming-course/?btnz=educba-inline-banner3)

About Us

Blog (<https://www.educba.com/blog/?source=footer>)

Who is EDUCBA? (<https://www.educba.com/about-us/?source=footer>)

Sign Up (<https://www.educba.com/software-development/signup/?source=footer>)

Corporate Training (<https://www.educba.com/corporate/?source=footer>)

Certificate from Top Institutions (<https://www.educba.com/educbalive/?source=footer>)

Contact Us (<https://www.educba.com/contact-us/?source=footer>)





[\(https://www.educba.com/software-development/\)](https://www.educba.com/software-development/)

Privacy Policy (<https://www.educba.com/privacy-policy/?source=footer>)

Apps

iPhone & iPad (<https://itunes.apple.com/in/app/educba-learning-app/id1341654580?mt=8>)

Android (<https://play.google.com/store/apps/details?id=com.educba.www>)

Resources

Free Courses (<https://www.educba.com/software-development/free-courses/?source=footer>)

Java Tutorials (<https://www.educba.com/software-development/software-development-tutorials/java-tutorial/?source=footer>)

Python Tutorials (<https://www.educba.com/software-development/software-development-tutorials/python-tutorial/?source=footer>)

All Tutorials (<https://www.educba.com/software-development/software-development-tutorials/?source=footer>)

Certification Courses

All Courses (<https://www.educba.com/software-development/courses/?source=footer>)

Software Development Course - All in One Bundle
(<https://www.educba.com/software-development/courses/software-development-course/?source=footer>)

Become a Python Developer (<https://www.educba.com/software-development/courses/python-certification-course/?source=footer>)

Java Course (<https://www.educba.com/software-development/courses/java-course/?source=footer>)





[\(https://www.educba.com/software-development/\)](https://www.educba.com/software-development/)

VB.NET Course (<https://www.educba.com/software-development/courses/vb-net-course/?source=footer>)

PHP Course (<https://www.educba.com/software-development/courses/php-course/?source=footer>)

© 2022 - EDUCBA. ALL RIGHTS RESERVED. THE CERTIFICATION NAMES ARE THE TRADEMARKS OF THEIR RESPECTIVE OWNERS.

