

Multidimensional Arrays in C

By Atiya Jokhio

Multiple-Subscripted Arrays

- An array of arrays is called as multi dimensional array. In simple words, an array created with more than one dimension (size) is called as multi dimensional array. Multi dimensional array can be of **two dimensional array** or **three dimensional array** or **four dimensional array**...

Most popular and commonly used multi dimensional array is **two dimensional array**. The 2-D arrays are used to store data in the form of table. We also use 2-D arrays to create mathematical **matrices**.

Multiple-Subscripted Arrays

- Multiple subscripted arrays
 - Tables with rows and columns (m by n array)
 - Like matrices: specify row, then column

	Column	Column 1	Column	Column 3
Row 0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 2	a[2][0]	a[2][1]	a[2][2]	a[2][3]

Array name Row subscript

Column subscript

Multiple-Subscripted Arrays (cont'd)

- Initialization

- `int b[2][2] = { { 1, 2 }, { 3, 4 } };`
- Initializers grouped by row in braces
- If not enough, unspecified elements set to zero
`int b[2][2] = { { 1 }, { 3, 4 } };`

1	2
3	4

- Referencing elements

- Specify row, then column
`printf("%d", b[0][1]);`

1	0
3	4

Multiple-Subscripted Arrays (cont'd)

The 2-D array arrangement is shown below.

Remember the counting of rows and columns begin with zero.

	col. no. 0	col. no. 1
row no. 0	1234	56
row no. 1	1212	33
row no. 2	1434	80
row no. 3	1312	78

Multiple-Subscripted Arrays (cont'd)

Memory Map of a 2-Dimensional Array:

Let us reiterate the arrangement of array elements in a two-dimensional array of students, which contains roll nos. in one column and the marks in the other.

The arrangement of array elements of a two-dimensional array in memory is shown below:

$s[0][0]$	$s[0][1]$	$s[1][0]$	$s[1][1]$	$s[2][0]$	$s[2][1]$	$s[3][0]$	$s[3][1]$
1234	56	1212	33	1434	80	1312	78
65508	65510	65512	65514	65516	65518	65520	65522

Multiple-Subscripted Arrays (cont'd)

```
#include <stdio.h>

int main () {

    /* an array with 5 rows and 2 columns*/
    int a[5][2] = { {0,0}, {1,2}, {2,4}, {3,6},{4,8}};
    int i, j;

    /* output each array element's value */
    for ( i = 0; i < 5; i++ ) {

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

    return 0;
}
```

```
a[0][0]: 0
a[0][1]: 0
a[1][0]: 1
a[1][1]: 2
a[2][0]: 2
a[2][1]: 4
a[3][0]: 3
a[3][1]: 6
a[4][0]: 4
a[4][1]: 8
```

Multiple-Subscripted Arrays (cont'd)

Example: Addition of two matrices

```
#include <stdio.h>
int main() {
    int r, c, a[100][100], b[100][100], sum[100][100], i, j;
    printf("Enter the number of rows (between 1 and 100): ");
    scanf("%d", &r);
    printf("Enter the number of columns (between 1 and 100): ");
    scanf("%d", &c);

    printf("\nEnter elements of 1st matrix:\n");
    for (i = 0; i < r; ++i)
        for (j = 0; j < c; ++j) {
            printf("Enter element a%d%d: ", i + 1, j + 1);
            scanf("%d", &a[i][j]);
        }

    printf("Enter elements of 2nd matrix:\n");
    for (i = 0; i < r; ++i)
        for (j = 0; j < c; ++j) {
            printf("Enter element a%d%d: ", i + 1, j + 1);
            scanf("%d", &b[i][j]);
        }
}
```


Multiple-Subscripted Arrays (cont'd)

Example: Addition of two matrices

```
// adding two matrices
for (i = 0; i < r; ++i)
    for (j = 0; j < c; ++j) {
        .....
        sum[i][j] = a[i][j] + b[i][j];
    }

// printing the result
printf("\nSum of two matrices: \n");
for (i = 0; i < r; ++i){
    .....
    for (j = 0; j < c; ++j) {
        .....
        printf("%d  ", sum[i][j]);
    }
    printf("\n");
}
return 0;
}
```

Multiple-Subscripted Arrays (cont'd)

Online links videos for operations of 2-D Array

- 1D Array

https://www.youtube.com/watch?v=AT14lCXuMKI&list=PLdo5W4Nhv31bbKJzrsKfMpo_grxuLI8LU

- 2-D Introduction

<https://www.youtube.com/watch?v=vyOZcg4cPRY>

<https://www.youtube.com/watch?v=KDQXUysHLL8>

- 2-D array Matrix Multiplication

<https://www.youtube.com/watch?v=2K0nuYFGETM>

- Sum of Diagonal Elements

<https://www.youtube.com/watch?v=QCrBuYMqlwg>

- Transpose of Matrix

<https://www.youtube.com/watch?v=-arlesoWEol>

Array of Strings

A string is a 1-D array of characters, so an array of strings is a 2-D array of characters.

Just like we can create a 2-D array of `int`, `float` etc; we can also create a 2-D array of character or array of strings. Here is how we can declare a 2-D array of characters.

```
char ch_arr[3][10] = {  
    {'s', 'p', 'i', 'k', 'e', '\0'},  
    {'t', 'o', 'm', '\0'},  
    {'j', 'e', 'r', 'r', 'y', '\0'}  
};
```

```
char ch_arr[3][10] = {  
    "spike",  
    "tom",  
    "jerry"  
};
```

Array of Strings

In memory allocation:

1000 1001 1002 1003 1004 1005 1006 1007 1008 1009

s	p	i	k	e	\0	\0	\0	\0	\0
---	---	---	---	---	----	----	----	----	----

1010 1011 1012 1013 1014 1015 1016 1017 1018 1019

t	o	m	\0	\0	\0	\0	\0	\0	\0
---	---	---	----	----	----	----	----	----	----

1020 1021 1022 1023 1024 1025 1026 1027 1028 1029

j	e	r	r	y	\0	\0	\0	\0	\0
---	---	---	---	---	----	----	----	----	----

Array of Strings

In memory allocation:

```
#include<stdio.h>
int main()
{
    char name[10][20];
    int i,n;

    printf("Enter the number of names (<10): ");
    scanf("%d",&n);

    //reading string from user
    printf("Enter %d names:\n",n);
    for(i=0;i<n;i++)
        scanf("%s",name[i]);

    //Displaying names
    printf("\nEntered names are:\n");
    for(i=0;i<n;i++)
        puts(name[i]);

    return 0;
}
```

Exercise

- Write a C program to read through an array of any type. Write a C program to scan through this array to find a particular value.
- Write a program to copy the contents of one array into another in the reverse order.
- Write a program to pick up the largest number from any 5 row by 5 column matrix.
- Write a program to add two 6 * 6 matrices.