How to access two dimensional array using pointers?

Suppose I have a pointer `array_ptr` pointing at base address of one dimensional array.

To access nth element of array using pointer

we use `*(array_ptr + n)`

(where `array_ptr` points to 0th element of array, `n` is the nth element to access and nth element starts from 0).

Now we know two dimensional array is array of one dimensional array. Hence let us see how to access a two dimensional array through pointer.

Let us suppose a two-dimensional array

```
int matrix[3][3];
```

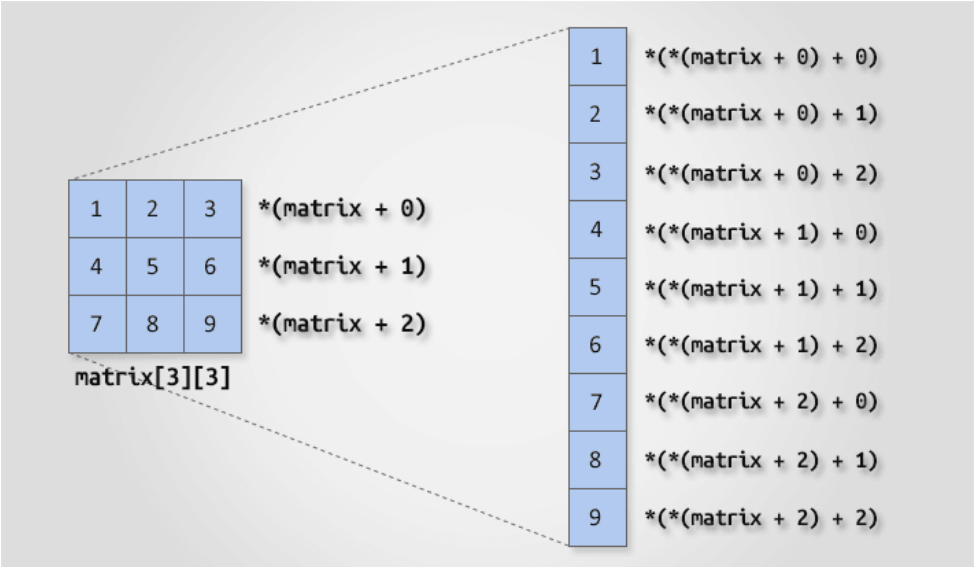
For the above array,

```
matrix                =>    Points to base address of two-dimensional array.
                            Since array decays to pointer.

*(matrix)             =>    Points to first row of two-dimensional array.
*(matrix + 0)         =>    Points to first row of two-dimensional array.
*(matrix + 1)         =>    Points to second row of two-dimensional array.

**matrix              =>    Points to matrix[0][0]
*(*(matrix + 0))      =>    Points to matrix[0][0]
*(*(matrix + 0) + 0) =>    Points to matrix[0][0]
*(*matrix + 1)        =>    Points to matrix[0][1]
*(*(matrix + 0) + 1) =>    Points to matrix[0][1]
*(*(matrix + 2) + 2) =>    Points to matrix[2][2]
```

Two dimensional array access using pointer is summarized in the below image.



Two dimensional array access using pointer

Program to access a two dimensional array using pointer

```c
#include<stdio.h>

int main()
{
    int arr[3][4] = {
                        {11,22,33,44},
                        {55,66,77,88},
                        {11,66,77,44}
                    };

    int i, j;

    for(i = 0; i < 3; i++)
    {
        printf("Address of %d th array %u \n",i , *(arr + i));
        for(j = 0; j < 4; j++)
        {
             printf("arr[%d][%d]=%d\n", i, j, *( *(arr + i) + j) );
        }
        printf("\n\n");
    }

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

Note: The fferences between %p and %x in C or C++. The %p is used to print the pointer value, and %x is used to print hexadecimal values. Though pointers can also be displayed using %u, or %x. If we want to print some value using %p and %x then we will not feel any major differences. The only difference that can be noticed is that the %p will print some leading zeros, but %x doesn't.

Example

```
#include<stdio.h>

main() {

   int x = 59;

   printf("Value using %%p: %p\n", x);

   printf("Value using %%x: %x\n", x);

}
```

Output

```
Value using %p: 000000000000003B
Value using %x: 3b
```

Write a C program to read elements in a matrix and check whether the matrix is upper triangular matrix or not. C program to check upper triangular matrix. Logic to find upper triangular matrix in C programming.

Example

Input

```
Input elements of matrix:
1 2 3
0 5 6
0 0 9
```

Output

```
Matrix is upper triangular
```

Write a C program to read elements in a matrix and find sum of upper triangular matrix. How to find sum of upper triangular matrix in C. Logic to find sum of upper triangular matrix.

Example

Input

```
Input matrix elements:
1 2 3
0 5 6
```

```
0 0 9
```

Output

```
Sum of upper triangular matrix = 11
```

Write a C program to read elements in a matrix and check whether the given matrix is symmetric matrix or not. How to check symmetric matrix in C. Logic to check symmetric matrix in C programming.

Example

Input

```
Input matrix elements:
1 2 3
2 4 5
3 5 8
```

Output

```
Given matrix is symmetric matrix.
```

What is Symmetric Matrix?

Symmetric matrix is a square matrix which is equal to its transpose. A symmetric matrix is always a square matrix. Symmetric matrix **A** is defined as - $A = A^T$

$$\begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 3 & 5 & 8 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 3 & 5 & 8 \end{bmatrix}^T$$

Symmetric matrix

Write a C program to read elements in a matrix and check whether matrix is an Identity matrix or not. C program for finding Identity matrix. Logic to check identity matrix in C programming.

Example

Input

```
Input elements in matrix:
1 0 0
0 1 0
0 0 1
```

Output

```
It is an Identity matrix
```

Write a C program to read elements in a matrix and interchange elements of primary(major) diagonal with secondary(minor) diagonal. C program for interchanging diagonals of a matrix. Logic to interchange diagonals of a matrix in C programming.

Example

Input

```
Input matrix elements:
1 2 3
4 5 6
7 8 9
```

Output

```
Matrix after interchanging its diagonal:
3 2 1
4 5 6
9 8 7
```