C Programming: Two Dimensional Array

Lecture-13

Dr. Asif Uddin Khan

Multidimensional array

Declaration

type name[size1][size2]...[sizeN];

Example:

int twodim[5][10];

Two-dimensional Arrays

- A two-dimensional array is a list of one-dimensional arrays.

Declaration of two-dimensional array

```
type arrayName [ x ][ y ];
```

Example:

```
int mat[3][3];
```

Two-dimensional Arrays

- A two-dimensional array can be considered as a table which will have x number of rows and y number of columns.
- A two-dimensional array **a**, which contains three rows and four columns can be shown as follows –

	Column 0	Column 1	Column 2	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]

Two-dimensional Arrays

- Thus, every element in the array **a** is identified by an element name of the form **a[i][j]**, where 'a' is the name of the array, and 'i' and 'j' are the subscripts that uniquely identify each element in 'a'.

Initializing Two-Dimensional Arrays

```
int a[3][4] = {
   {0, 1, 2, 3} ,   /*  initializers for row indexed by 0 */
   {4, 5, 6, 7} ,   /*  initializers for row indexed by 1 */
   {8, 9, 10, 11}   /*  initializers for row indexed by 2 */
};
```

```
int a[3][4] = {0,1,2,3,4,5,6,7,8,9,10,11};
```

Accessing elements

```
int val = a[2][3];
```

```
#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;
}
```

Matrix addition

```
#include <stdio.h>
int main() {
  int m, n, i, j;
  printf("Enter the number of rows and columns of the matrices: ");
  scanf("%d%d", &m, &n);
  int a[m][n], b[m][n], c[m][n];
  printf("Enter the elements of matrix A: \n");
  for (i = 0; i < m; i++) {
    for (j = 0; j < n; j++) {
      scanf("%d", &a[i][j]);
    }
  }
  printf("Enter the elements of matrix B: \n");
  for (i = 0; i < m; i++) {
    for (j = 0; j < n; j++) {
      scanf("%d", &b[i][j]);
    }
  }
  // add the matrices
  for (i = 0; i < m; i++) {
    for (j = 0; j < n; j++) {
      c[i][j] = a[i][j] + b[i][j];
    }
  }
  // print the result
  printf("The sum of the two matrices is: \n");
  for (i = 0; i < m; i++) {
    for (j = 0; j < n; j++) {
      printf("%d ", c[i][j]);
    }
    printf("\n");
  }
  return 0;
}
```

Properties of Array

- Each element of an array is of same data type and carries the same size, i.e., int = 4 bytes.
- Elements of the array are stored at contiguous memory locations where the first element is stored at the smallest memory location.
- Elements of the array can be randomly accessed since we can calculate the address of each element of the array with the given base address and the size of the data element.

Advantage of C Array

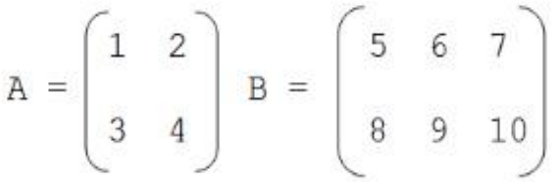
- **1) Code Optimization**: Less code to the access the data.
- **2) Ease of traversing**: By using the for loop, we can retrieve the elements of an array easily.
- **3) Ease of sorting**: To sort the elements of the array, we need a few lines of code only.
- **4) Random Access**: We can access any element randomly using the array.

Disadvantage of C Array

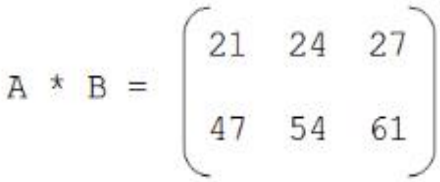
- **1) Fixed Size**: Whatever size, we define at the time of declaration of the array, we can't exceed the limit. So, it doesn't grow the size dynamically like LinkedList which we will learn later.

Logic of Matrix multiplication in C

- In matrix multiplication first matrix one row element is multiplied by second matrix all column elements.



Multiplication of two matrixes:

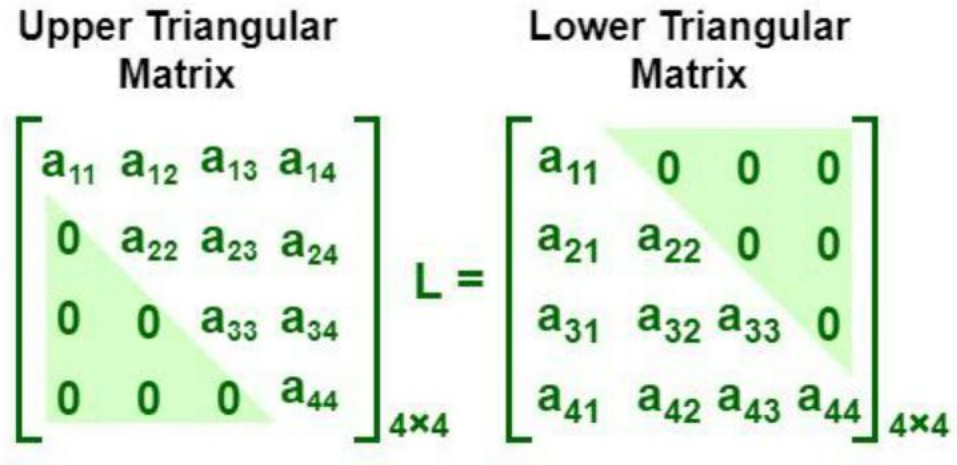


JavaTpoint

```
for(i=0;i<r;i++)
{
for(j=0;j<c;j++)
{
mul[i][j]=0;
for(k=0;k<c;k++)
{
mul[i][j]+=a[i][k]*b[k][j];
}
}
}
```

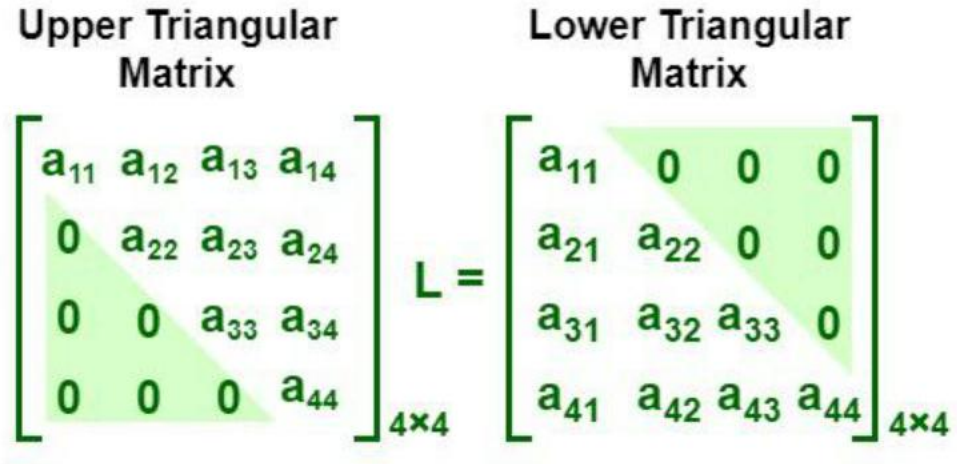
Logic for sum of upper triangle

```
// To calculate sum of upper triangle
    for (i = 0; i < r; i++)
        for (j = 0; j < c; j++) {
            if (i <= j) {
                upper_sum += mat[i][j];
            }
        }
```



Logic for sum of lower triangle

```
// To calculate sum of lower
    for (i = 0; i < r; i++)
        for (j = 0; j < c; j++) {
            if (i>=j) {
                lower_sum += mat[i][j];
            }
        }
```



References

1. C programming by E Balaguruswami
2. Programming C by Y. kanitkar
3. Programming C by Denis Ritchie
4. NPTEL Lecture note of Dr. Partha Pratim Das, Department of Computer Science and Engineering, Indian Institute of Technology, Kharagpur.
5. https://docs.oracle.com/cd/E18752_01/html/817-6223/chp-typeopexpr-2.html
6. https://data-flair.training/blogs/escape-sequence-in-c/
7. Internet source