**Programming Fundamentals**

|  |  |
| --- | --- |
| Lab 12 | |
| **Topic** | 2D ARRAYS |
| **Objective** | * 2D ARRAYS   + Concept of 2D array as array of arrays.   + Declaring and initializing 2D arrays.   + Memory model of 2D array.   + Discuss why column size of each row must be same.   + Accessing individual elements in 2D array.   + Traversing 2D array (row-wise, column-wise, diagonally)   + Handling 2D arrays using 1D array functions.   + Problem solving involving 2D arrays. |

**Lab Description:**

This lab is basically designed for the basic understanding of 2D array and its use.

**Two-dimensional Arrays:**

The simplest form of multidimensional array is the two-dimensional array. A two-dimensional array is, in essence, a list of one-dimensional arrays. To declare a two-dimensional integer array of size [rowSize][ColumnSize], you would write something as follows –

**SYNTAX**:

dataType arrayName[rowSize][columnSize];

To **declare** a 2D array of type Integer with 2 rows and 4 columns:

int arr[3][4];

**Initializing Two-Dimensional Arrays:**

Multidimensional arrays may be initialized by specifying bracketed values for each row. Following is an array with 3 rows and each row has 4 columns.

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

The nested braces, which indicate the intended row, are optional. The following initialization is equivalent to the previous example

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

**Memory model of Two-Dimensional Array:**

Two-Dimensional array basically define as matrix form, but in actual it contains same memory map like one dimensional. Each index of a 2D array mapped on a 1D array.

**Example:**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **2D array:**   |  |  |  | | --- | --- | --- | | **arr** | **[0]** | **[1]** | | **[0]** | 11 | 21 | | **[1]** | 35 | 14 | | **[2]** | 51 | 63 | | **1D array:**   |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | |  | **[0]** | **[1]** | **[2]** | **[3]** | **[4]** | **[5]** | | **arr** | 11 | 21 | 35 | 14 | 51 | 63 | |

**Accessing Two-Dimensional Array Elements:**

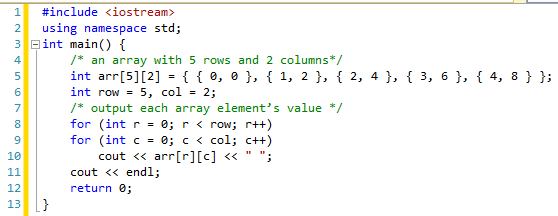
An element in a two-dimensional array is accessed by using the subscripts, i.e., row index and column index of the array.

**Example:**

int val = arr[2][3];

The above statement will take the 4th element from the 3rd row of the array. You can verify it in the above figure. Let us check the following program where we have used a nested loop to handle a two-dimensional array.

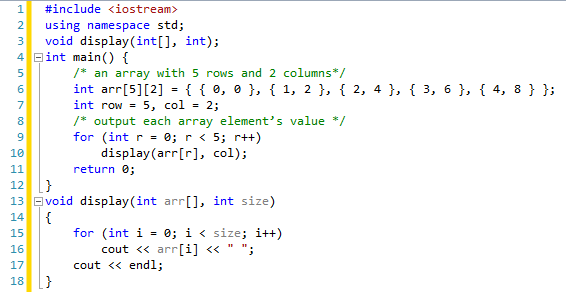
**Example:**



**Handling 2D arrays using 1D array functions:**

We can handle 2D array using 1D functions. Because each row of a 2D array can be used as 1D array. So we can use all function for each row of a 2D array which we define for 1D array.

**Example:**



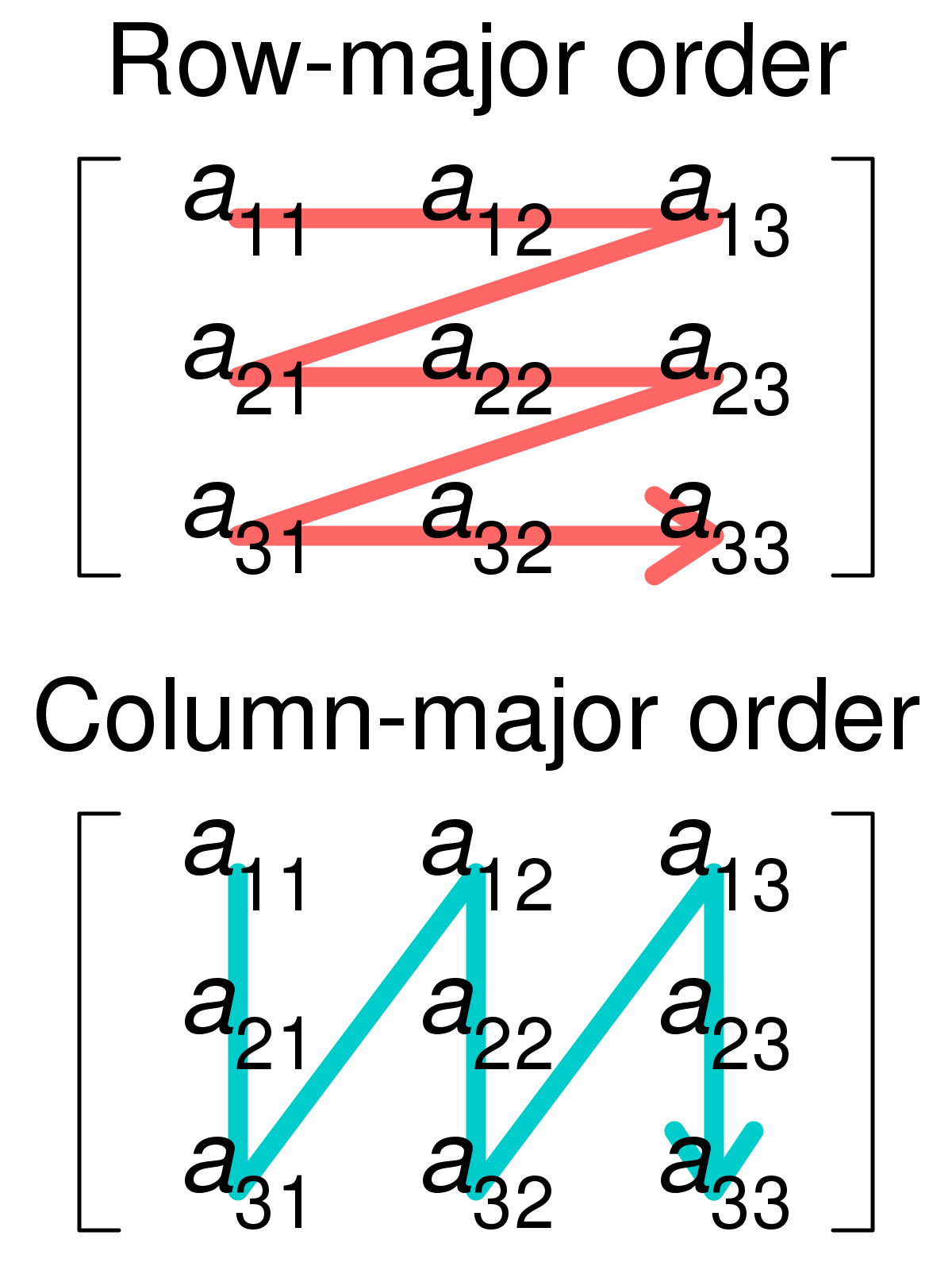
**LAB TASKS**

# Task 1:

Write a C++ program in which your task is to create a 3x3 matrix (2d Array). Take input from the user and display all the values of array.

# Task 2:

Write a C++ program in which your task is to create a 5x5 matrix (2d Array). Take input from the user and display all the values of array row major order and column major order.



# Task 3:

Write a C++ program in which your task is to create a 6x6 matrix (2d Array). Take input from the user. After the input your program should calculate the Sum of both even and odd values present in the matrix and store the sum in variables EvenSum and OddSum respectively. At the end of the program Display all the values of matrix and Sum values.

# Task 4:

Write a C++ program in which user takes integer values in a 4x6 matrix (2d Array) your task is to calculate the sum of all three rows and display the array and its sum row wise at the end of every row.

# Task 5:

Create a 3x3 matrix (2D Array) and take input in the created matrix (only 1s and 0s). Check that the matrix is an Identity matrix or not.

**Identity Matrix is a matrix which has 1s on it main diagonal and 0s in all remaining places**

