Objectives

After performing this lab, students shall be able to:

- ✓ Dynamically allocate memory of 2D arrays
- ✓ Pass pointers to 2D arrays to functions
- ✓ Use new and delete operators
- ✓ Solve some basic problems related to matrices

Important Note:

- **All allocations of 1D/2D pointers should be dynamic. Do not consume extra bytes.**
- **Delete the array when it is no longer needed.**
- **All the data will be given by user.**
- **Pass the pointers to functions instead of [].**
- **Debug your code to find errors/bugs.**
- **Use offset notation to traverse the arrays (implement your functions using subscript notation, test their functionalities and then change subscript notation to offset notation)**

Exercise 1: Write a function **int** AllocateMemory(int& rows, int& cols)** that takes size of matrix (rows and columns) from user, allocates memory for the matrix and return its pointer.

What is the advantage of sending the two parameters by reference?

Exercise 2: Write a function **void InputMatrix(int** matrix, const int& rows, const int& cols)** that inputs the matrix elements from user.

Why are we passing the parameters as const?

Exercise 3: Write a function **void DisplayMatrix(int** matrix, const int& rows, const int& cols)** that displays the matrix in proper format.

Exercise 4: Write an inline function **void DeallocateMemory(int** matrix, const int& rows)** that deallocates all the memory. Make all the above mentioned functions inline.

Test your program. An example run is given below.

```
Enter total rows:4
Enter total columns:3
The matrix is:
0 0 0
0 0 0
0 0 0
0 0 0
```

(Suppose user entered 0 for all elements)

Exercise 5: Write a function called `MaxOfColumn` that takes as parameters a pointer to a 2D array and its dimensions. It should return the largest element in each column of the array. Since there is more than one column in 2D array, you have to return an array that contains largest of each column.

For example, if the **Sample Matrix** is

1	4	**8**	5
9	1	6	**10**
5	**7**	2	8

Your function will return array containing maximum elements of all the columns i.e.
9, 7, 8, 10. This is just a sample matrix; your program should run for any matrix/dimensions given by user. **Display the returned array in main function.**

Exercise 6: Write a function called **GetDiagonal** that takes as parameters a pointer to a 2D array (nxn matrix) and its dimensions and return the diagonal elements of array

For example, if the **Sample Matrix** is

1	4	8
9	1	6
5	7	2

For example for the sample matrix given above, your function will return
1, 1, 2. **Display the result in main.**

Exercise 7: Write a program that allows the user to enter the last names of five candidates in a local election and the number of votes received by each candidate. The program should then output each candidate's name, the number of votes received, and the percentage of the total votes received by the candidate. Your program should also output the winner of the election. A sample output is:

```
Candidate          Votes Received       % of Total Votes

Johnson                 5000                 25.91
Miller                  4000                 20.73
Duffy                   6000                 31.09
Robinson                2500                 12.95
Ashtony                 1800                  9.33
Total                  19300
```

The Winner of the Election is Duffy.

You have to fill the 2D and 1D dynamic array in the main and pass to **void Election(const** char **canidates,**const** int *Votes,const int &size) .The function will display the winner name.

Exercise 8: A square matrix in which all the entries below the main diagonal are zero is called upper triangular. Write a function **IsMatrixUpperTriangular** which takes a matrix and returns true if the matrix is upper triangular and false otherwise.
For example matrix A shown below is upper triangular while matrix B is not upper triangular.

-1	2	7	0
0	5	0	-1
0	0	7	0
0	0	0	0

Matrix A

-1	2	7	0
0	5	0	-1
-9	0	7	0
0	0	0	0

Matrix B

END