Green University of Bangladesh
Department of Computer Science and Engineering(CSE)

Faculty of Sciences and Engineering

Semester: (Spring, Year:2021), B.Sc. in CSE (Day)

KSA Assignment NO 01

Course Title: Algorithms

Course Code: CSE 205 Section: DB

Student Details

Name		ID
1.	MD.Mehedi Hassan Nayeem	213902045

Lab Date : _

Submission Date : _

Course Teacher's Name : **Md. Sultanul Islam Ovi**

[For Teachers use only: Don't Write Anything inside this box]

Lab Report Status

Marks: ………………………………… **Signature:**.....................

Comments:... **Date:**..............................

Problem Name : "Lo Shu Magic Square Problem"

Goal : In this assignment, My task will be to check the grid as Lo Shu Magic Square. Lo Shu Magic Square uses the formula of $n(n^2+1)/2 = 15$, (if n is 3, it means it's a 3 x 3 grid). Thus, all rows, all columns, and all diagonals will have the sum of 15.In this assignment, My task will be to use a two-dimensional array to simulate a magic square.

```cpp
#include <bits/stdc++.h>
using namespace std;

int colSum, row, col, sumDiag1=0, sumDiag2=0, matrix[50][50], n,
m;

int isMagicSquare(int matrix[50][50], int n, int m){

    for (row = 0; row < n; row++)
        sumDiag1 += matrix[row][row];

    for (row = 0; row < n; row++)
        sumDiag2 += matrix[row][n - 1 - row];

    if(sumDiag1 != sumDiag2)
        return 0;
    for (row = 0; row < n; row++){
        int rowSum = 0;

        for (col = 0; col < m; col++)
            rowSum += matrix[row][col];

        if (rowSum != sumDiag1)
            return 0;
    }
    for (col = 0; col < n; col++){
        colSum = 0;

        for (row = 0; row < n; row++)
            colSum += matrix[row][col];

        if (colSum != sumDiag1)
            return 0;
    }

    return 1;
}
int main(){
    cout<<"Enter the number of rows of the matrix: ";
    cin >> m;
    cout<<"Enter the number of columns of the matrix: ";
    cin >> n;

    if(n != m){
        cout<<"The matrix must be a square matrix.";
        exit(0);
    }

    cout << "\nEnter the elements of the matrix: \n";
    for(row = 0; row < n; row++)
        for(col = 0; col < m; col++)
            cin >> matrix[row][col];

    if (isMagicSquare(matrix, n, m))
        cout << "\nThe matrix follows the Lo Shu magic square.";
    else
        cout << "\nThe matrix does not follow the Lo Shu magic
square.";
}
```

Problem 02

Name of the problem : The grid addition.

Goal : In this assignment, My first task will be the grid addition. Starting with any number on the top row, make your way to the bottom, adding the numbers as I can go.
Each time I go to the next row, I can move straight down, or one place to the left or right.

Code :

```cpp
#include <bits/stdc++.h>
#include <vector>
#include <algorithm>
using namespace std;

int main() {
    int n;
    cout << " Enter the grid dimensions: ";
    cin >> n;

    vector<vector<int>> grid(n, vector<int>(n));
    vector<vector<int>> max_sum(n, vector<int>(n));
    vector<vector<int>> min_sum(n, vector<int>(n));

    cout << "Please enter the desired input:" << endl;
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            cin >> grid[i][j];
            max_sum[i][j] = grid[i][j];
            min_sum[i][j] = grid[i][j];
        }
    }
    for (int i = 1; i < n; i++) {
        for (int j = 0; j < n; j++) {
            if (j > 0) {
                max_sum[i][j] = max(max_sum[i-1][j-1],
max_sum[i-1][j]);
                min_sum[i][j] = min(min_sum[i-1][j-1],
min_sum[i-1][j]);
            }
            if (j < n-1) {
                max_sum[i][j] = max(max_sum[i][j],
max_sum[i-1][j+1]);
                min_sum[i][j] = min(min_sum[i][j],
min_sum[i-1][j+1]);
            }
            max_sum[i][j] += grid[i][j];
            min_sum[i][j] += grid[i][j];
        }
    }

    int highest_sum = *max_element(max_sum[n-1].begin(),
max_sum[n-1].end());
    int lowest_sum = *min_element(min_sum[n-1].begin(),
min_sum[n-1].end());

    cout << "Highest Sum: " << highest_sum << endl;
    cout << "Lowest Sum: " << lowest_sum << endl;

    return 0;
}
```