2D_array

1

```
#include <iostream>
using namespace std;
//here in the print function there must be the number of column
//passed by reference
void print(int arr[][100] , int n ,int m){
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < m; j++)
        {
            cout << arr[i][j] << " ";
        }
        cout << endl;
    }
}
int main(){
    int arr[100][100];
    int n,m;
    cin>>n>>m;
    //take input
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < m; j++)
        {
            cin >> arr[i][j];
        }
    }
    // print the input
    print(arr,n,m);
    return 0;
}
```

2

```
//2D char array
#include <bits/stdc++.h>
using namespace std;
int main(){
    char numbers[7][10] = {
        "one",
        "two",
        "three",
        "four",
        "five",
        "six",
        "seven"
    };

    cout << numbers[2] << endl;
    cout << numbers[2][5] << endl;
    return 0;
}
```

3

```
//sprial order printing
#include <iostream>
using namespace std;
void print(int arr[][4] , int n ,int m){
    //here we have to take four variable
    int startRow = 0;
    int endRow = n-1;
    int startCol = 0;
    int endCol = m-1;
    //outer loop
    while(startCol <= endCol and startRow <= endRow){
        //start row
        for (int col = startCol; col <=endCol; col++)
        {
            cout << arr[startRow][col]<< " ";
        }
        //end col
        for (int row = startRow+1; row<=endRow;row++)
        {
            cout << arr[row][endCol]<<" ";
        }
        //end row
        for (int col = endCol - 1 ; col >=startCol; col--)
        {
            //avoid duplicate printing of elements
            if(startRow==endRow){
                break;
            }
            cout << arr[endRow][col] << "" ;
        }
        //start colmun
        for (int row = endRow - 1; row>=startRow+1 ; row--)
        {
            //avoid duplicate printing of elements
            if(startCol==endCol){
                break;
            }
            cout << arr[row][startCol] <<" ";
        }
```

```cpp
        //update the variable so that they point to the inner spiral
        startRow++;
        endRow--;
        startCol++;
        endCol--;
    }
}
int main(){
    int n,m;
    cin >> n >> m;
    int arr[4][4];
    for (int i = 0; i < n; ++i)
    {
        for (int j = 0; j < m; ++j)
        {
            cin >> arr[i][j];
        }
    }
    print(arr,n,m);

}
```

4

```cpp
//staircase search
#include <iostream>
using namespace std;
pair<int,int> staircaseSearch(int arr[][4],int n,int m,int key){
    //Later if the element is not found return -1,-1
    if(key < arr[0][0] or key > arr[n-1][m-1]){
        return {-1,-1};
    }
    //staircase search
    int i=0;
    int j=m-1;
    while(i<n and j>=0){
        if(arr[i][j]==key){
            return {i,j};
        }
        else if(arr[i][j]>key){
            j--;
        }
        else{
            i++;
        }
    }
    return {-1,-1};
}
```

```
int main(){
    int arr[4][4];
    for (int i = 0; i < 4; ++i)
    {
        for (int j = 0; j < 4; ++j)
        {
            cin >> arr[i][j];
        }
    }
    int element;
    cin >> element;
    pair<int,int> answer = staircaseSearch(arr,4,4,element);
    cout << answer.first << " " <<answer.second <<endl;
}
```

5

```
//mango tree problem
//goal : get the maximum mango tree
/*intput
0 1 1 0 0 0
1 0 0 1 1 0
0 1 0 0 0 0
0 1 1 0 0 1
1 0 0 1 1 0
0 1 0 0 0 0
*/
#include <iostream>
using namespace std;
void maximum_submatrix_tree(int arr[6][6],int sum[6][6]){
    for (int i = 1; i < 6; ++i)
    {
        sum[0][i] = arr[0][i] + sum[0][i-1];
    }
    for (int j = 0; j < 6; j++)
    {
        sum[j][0] = arr[j][0] + sum[j-1][0];
    }
    for(int i = 1; i<6; i++){
        for (int j = 1; j < 6; j++)
        {
            if(arr[i][j]==0){
                sum[i][j] = sum[i-1][j] + sum[i][j-1] -sum[i-1][j-1];
            }
            else{
                sum[i][j] = 1 + sum[i-1][j] + sum[i][j-1] -sum[i-1][j-1];
            }
        }
```

```cpp
    }
}
void maximum_submatrix_sum(int sum[6][6]){
    for (int i = 0; i < 6; ++i)
    {
        for (int j = 0; j < 6; j++)
        {

            int s1 = sum[i][j];
            int s2 = sum[i][5] - s1;
            int s3 = sum[5][j] - s1;
            int s4 = sum[5][5] - s1 - s2 -s3;
            if(s1==s2 && s2==s3){
                cout <<"{ " <<i <<" , "<<j<<" }"<<endl;
            }
            else if(s1==s2 && s2==s4){
                cout <<"{ " <<i <<" , "<<j<<" }"<<endl;
            }
            else if(s1==s3 && s3==s4){
                cout <<"{ " <<i <<" , "<<j<<" }"<<endl;
            }
            // cout << s1 << " " << s2 << " " <<s3 <<" " <<s4<<" "<<endl;
        }
    }
}
void print(int arr[6][6] , int n ,int m){
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < m; j++)
        {
            cout << arr[i][j] << " ";
        }
        cout << endl;
    }
}
int main(){
    int arr[6][6] = {0};
    for (int i = 0; i < 6; ++i)
    {
        for (int j = 0; j < 6; ++j)
        {
            cin >> arr[i][j];
        }
    }
    int sum[6][6] = {0};
    maximum_submatrix_tree(arr,sum);
    print(sum,6,6);
    maximum_submatrix_sum(sum) ;
```

```
}
```