2D Array Problems Set 4

Assignment Solutions

PW SKILLS

Q1 - Write a user defined function upper() which takes an integer square matrix as an input and its size N and prints the upper half of the matrix. (Easy)

Sample Input:arr[][]=[[1,2,3,4],[5,6,7,8],[9,10,11,12],[13,14,15,16]] N=4
Sample Output: 1 2 3 4
6 7 8
11 12
16

Explanation:
For each row we will just print all those cells where the column number>=row number and ignore the rest.

```
#include <bits/stdc++.h>
using namespace std;
void upper(vector<vector<int>> arr,int n)
{
      cout<<"The upper half is : "<<endl;
    for(int i=0;i<n;i++)
    {
        for(int j=0;j<n;j++)
        {
          if(j>=i)
          {
            cout<<arr[i][j]<<" ";
          }else{
            cout<<" ";
          }
        }
        cout<<endl;
    }
}
int main()
{
    int r;
    cout<<"Enter the row and column size : ";
    cin>>r;
    vector<vector<int>> arr(r,vector<int>(r));
    cout<<"Enter the matrix elements : "<<endl;
    for(int i=0;i<r;i++)
    {
        for(int j=0;j<r;j++)
        {
            cin>>arr[i][j];
        }
    }
    upper(arr,r);
}
```

Output:

```
Enter the row and column size : 4
Enter the matrix elements :
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
The upper half is :
1 2 3 4
  6 7 8
    11 12
      16


...Program finished with exit code 0
Press ENTER to exit console.
```

Q2 - A square matrix is said to be an perfect Matrix if both of the following conditions hold: (Medium)

a) All the elements in the diagonals of the matrix are non-zero integers.
b) All other elements except the diagonal elements are 0.
Given a 2D integer array grid of size n x n representing a square matrix, return true if grid is a perfect matrix. Otherwise, return false using functions.

Sample Input: [[1,0,0,1],[0,2,1,0],[0,1,2,0],[3,0,0,1]]
Sample Output: true
Sample Input: [[5,7,0],[0,3,1],[0,5,0]]
Sample Output: false

Explanation:
We will check if any of the diagonal elements are 0, if yes then we will simply return false, if not then we will replace it with -1 which indicates that the cell has been visited. We will then check if any element is non zero and not -1 then we will return false because it will mean that the element is not diagonal and also non zero.

```
#include <bits/stdc++.h>
using namespace std;
bool helper(vector<vector<int>> grid)
{
    int n =grid.size();
      int m=grid[0].size();
      for(int i=0;i<n;i++)
      {
      if(grid[i][i]==0)
      {
        return false;
      }
      if(grid[i][n-i-1]==0)
      {
        return false;
      }
      grid[i][i]=-1;
      grid[i][n-i-1]=-1;
    }
    for(int i=0;i<n;i++)
    {
      for(int j=0;j<n;j++)
      {
        if(grid[i][j]!=-1 && grid[i][j]!=0)
        {
            return false;
        }
      }
    }
    return true;
}
int main()
```

```
{
    int r;
    int c;
    cout<<"Enter the row and column size : ";
    cin>>r>>c;
    vector<vector<int>> arr(r,vector<int>(c));
    cout<<"Enter the matrix elements : "<<endl;
    for(int i=0;i<r;i++)
    {
        for(int j=0;j<c;j++)
        {
            cin>>arr[i][j];
        }
    }
    int val=helper(arr);
    if(val)
    {
        cout<<"true";
    }else{
        cout<<"false";
    }
}
```

```
Enter the row and column size : 4 4
Enter the matrix elements :
1 0 0 1
0 2 1 0
0 1 2 0
3 0 0 1
true

...Program finished with exit code 0
Press ENTER to exit console.
```