Program:

```
#include<iostream>
using namespace std;

bool isSafePlaceForQueen(int **chess, int row, int col, int
n) {
    // Left Diagonal Check
    for (int i = row - 1, j = col - 1; i >= 0 && j >= 0; i--, j--)
    {
        if (chess[i][j] == 1)
            return false;
    }
    // Vertical Check
    for (int i = row - 1, j = col; i >= 0; i--)
    {
        if (chess[i][j] == 1)
            return false;
    }
    // Right Diagonal Check
    for (int i = row - 1, j = col + 1; i >= 0 && j < n; i--, j++)
    {
        if (chess[i][j] == 1)
            return false;
    }
    return true;
}

void printNQueen(int **chess, int n, int row, string asf, int **displayChess)
{
    if (row == n)
    {
        cout << "\n-----------------------------\n";
        cout << asf + "." << endl;
        cout << "-----------------------------\n\n"
        for (int i = 0; i < n; i++)
        {
            for(int j = 0; j < n; j++){
                if(displayChess[i][j] == 1){
                    cout << " * ";
                }
                else{
                    cout << " - ";
                }
            }
```

```
            cout << endl;
        }
        cout << "\n------------------------------\n\n";
        return;
    }

    for (int col = 0; col < n; col++)
    {
        if (isSafePlaceForQueen(chess, row, col, n) == true)
        {
            chess[row][col] = 1;
            displayChess[row][col] = 1;
            printNQueen(chess, n, row + 1, asf + std::to_string(row) + " - " + std::to_string(col) +
", ", displayChess);
            displayChess[row][col] = 0;
            chess[row][col] = 0;
        }
    }
}

int main(){
    cout<<"Enter the size of N x N chessboard Matrix: ";
    int n;
    cin>>n;
    int** chess = new int*[n];
    int** displayChess = new int*[n];
    for(int i = 0; i < n; i++){
        chess[i] = new int[n];
        displayChess[i] = new int[n];
        for(int j = 0; j < n; j++){
            chess[i][j] = 0;
            displayChess[i][j] = 0;
        }
    }
    cout << endl << "Safe Queen Positions: " << endl;
    printNQueen(chess, n, 0, "", displayChess);

    return 0;
}
```

Output:

```
PS H:\STUDY\College\DAA Lab> cd "h:\STUDY\College\DAA Lab\" ; if ($?) {
g++ nQueens.cpp -o nQueens
} ; if ($?) { .\nQueens }
Enter the size of N x N chessboard Matrix: 4

Safe Queen Positions:

------------------------------
0 - 1, 1 - 3, 2 - 0, 3 - 2, .
------------------------------

 _ * _ _
 _ _ _ *
 * _ _ _
 _ _ * _

------------------------------


------------------------------
0 - 2, 1 - 0, 2 - 3, 3 - 1, .
------------------------------

 _ _ * _
 * _ _ _
 _ _ _ *
 _ * _ _

------------------------------

PS H:\STUDY\College\DAA Lab>
```