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

#define N 8

void printBoard(vector<vector<int>>& board)
{
    for (int i = 0; i < N; i++)
    {
        for (int j = 0; j < N; j++)
        {
            cout << (board[i][j] ? "Q " : ". ");
        }
        cout << endl;
    }
}
bool isSafe(vector<vector<int>>& board, int row, int col)
{
    //left row
    for (int i = 0; i < col; i++)
    {
        if (board[row][i])
        {
            return false;
        }
    }
    //upper left diagonal
    for (int i = row, j = col; i >= 0 && j >= 0; i--, j--)
    {
        if (board[i][j])
        {
            return false;
        }
    }
    //lower left diagonal
    for (int i = row, j = col; i < N && j >= 0; i++, j--) {
        if (board[i][j]) return false;
    }

    return true;
}

bool solveNQueenUtil(vector<vector<int>>& board, int col)
{
    if (col >= N)
    {
        return true;
    }
    for (int i = 0; i < N; i++)
```

```
    {
        if (isSafe(board, i, col))
        {
            board[i][col] = 1;
            if (solveNQueenUtil(board, col + 1))
            {
                return true;
            }
            board[i][col] = 0; // Backtrack
        }
    }
    return false; // No solution found
}

void solveNQueen()
{
    vector<vector<int>> board(N, vector<int>(N, 0)); // Initialize the board
    if (solveNQueenUtil(board, 0))
    {
        printBoard(board);
    } else {
        cout << "No solution exists." << endl;
    }
}

int main()
{
    solveNQueen();
    return 0;
}
```