```
#include <iostream>
#include <vector> using
namespace std;
// Function to print the solution
void printSolution(const vector<vector<char>>& board) { for
    (const auto& row : board) {
            for (char cell : row)
             cout << " " << cell << " ";
        cout << endl;
    }
}
// Function to check if a queen can be placed on board[row][col] bool
isSafe(const vector<vector<char>>& board, int row, int col) {
    int i, j;
    int n = board.size();

    // Check the row on the left side  for
    (i = 0; i < col; i++)
        if (board[row][i] == 'Q')
            return false;
    // Check upper diagonal on the left side
    for (i = row, j = col; i >= 0 && j >= 0; i--, j--) if
        (board[i][j] == 'Q')
            return false;
    // Check lower diagonal on the left side
    for (i = row, j = col; j >= 0 && i < n; i++, j--) if
        (board[i][j] == 'Q')
            return false; return
    true;
}
// Recursive function to solve N Queens problem
bool solveNQUtil(vector<vector<char>>& board, int col) {  int
    n = board.size();// If all queens are placed, return true
if (col >= n)
return true;
```

```
    // Consider this column and try placing this queen in all rows one by one for
    (int i = 0; i < n; i++) {
      // Check if the queen can be placed on board[i][col] if
      (isSafe(board, i, col)) {
        // Place this queen in board[i][col]  board[i][col]
        = 'Q';
        // Recur to place rest of the queens  if
        (solveNQUtil(board, col + 1))
            return true;
        // If placing queen in board[i][col] doesn't lead to a solution, then remove queen from
board[i][col]
        board[i][col] = '-';
      }
    }
    // If the queen cannot be placed in any row in this column, then return false
    return false;
}
// Function to solve N Queens problem for 4 queens
void solve4Queens() {
    int n = 4;
    vector<vector<char>> board(n, vector<char>(n, '-'));
     if (solveNQUtil(board, 0) == false) {
      cout << "Solution does not exist" << endl; return;
    }
    printSolution(board);
}
// Driver function
 int main() {
    cout << "Solution for 4 Queens problem:" << endl;
    solve4 Queens();
    return 0;
}
```