#include<iostream>

#define N 4

using namespace std;

// print board

void printSolution(int board[N][N])

{

for (int i = 0; i < N; i++) {

for (int j = 0; j < N; j++)

if(board[i][j])

cout << "Q ";

else cout<<". ";

printf("\n");

}

}

bool isSafe(int board[N][N], int row, int col)

{

int i, j;

// Check column

for (i = 0; i < col; i++)

if (board[row][i])

return false;

// Check upper diagonal

for (i = row, j = col; i >= 0 && j >= 0; i--, j--)

if (board[i][j])

return false;

// Check lower diagonal

for (i = row, j = col; j >= 0 && i < N; i++, j--)

if (board[i][j])

return false;

return true;

}

// A recursive function to solve N Queen problem

bool solveNQUtil(int board[N][N], int col)

{

if (col >= N)

return true;

for (int i = 0; i < N; i++) {

if (isSafe(board, i, col)) {

board[i][col] = 1;

if (solveNQUtil(board, col + 1))

return true;

board[i][col] = 0; // BACKTRACK

}

}

return false;

}

int main()

{

int board[N][N] = { { 0, 0, 0, 0 },

{ 0, 0, 0, 0 },

{ 0, 0, 0, 0 },

{ 0, 0, 0, 0 } };

if (solveNQUtil(board, 0) == false) {

cout << "Solution does not exist";

return false;

}

printSolution(board);

return 0;

}