#include <iostream>

#include <vector>

using namespace std;

bool isSafe(vector<vector<int>>& board, int row, int col, int N) {

// Check if there is a queen in the same row

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

if (board[row][i] == 1) {

return false;

}

}

// Check upper diagonal on the left side

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

if (board[i][j] == 1) {

return false;

}

}

// Check lower diagonal on the left side

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

if (board[i][j] == 1) {

return false;

}

}

return true;

}

bool solveNQueensUtil(vector<vector<int>>& board, int col, int N) {

if (col >= N) {

return true; // All queens have been placed

}

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

if (isSafe(board, row, col, N)) {

board[row][col] = 1; // Place the queen

// Recursively place the rest of the queens

if (solveNQueensUtil(board, col + 1, N)) {

return true;

}

// If placing the queen in board[row][col] doesn't lead to a solution,

// then backtrack and try placing the queen in the next row of the current column.

board[row][col] = 0; // Remove the queen

}

}

return false; // No solution found

}

void solveNQueens(int N) {

vector<vector<int>> board(N, vector<int>(N, 0));

if (solveNQueensUtil(board, 0, N)) {

// Print the board with queens placed

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

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

cout << board[i][j] << " ";

}

cout << endl;

}

} else {

cout << "No solution exists!" << endl;

}

}

int main() {

int N;

cout << "Enter the value of N: ";

cin >> N;

solveNQueens(N);

return 0;

}

//abhishek

#include <iostream>

using namespace std;

/\*Function to check the given position is safe to place the Queen\*/

bool isSafe(int \*\*arr, int x, int y, int n)

{

/\*Check for upper side of column\*/

for (int row = 0; row < x; row++)

{

if (arr[row][y] == 1)

return false;

}

/\*Check for left diagonal\*/

int row = x;

int col = y;

while (row >= 0 && col >= 0)

{

if (arr[row][col] == 1)

return false;

row--;

col--;

}

/\*Check for right diagonal\*/

row = x;

col = y;

while (row >= 0 && col < n)

{

if (arr[row][col] == 1)

return false;

row--;

col++;

}

return true;

}

bool nQueen(int \*\*arr, int x, int n)

{

if (x >= n)

return true;

for (int col = 0; col < n; col++)

{

if (isSafe(arr, x, col, n))

{

arr[x][col] = 1;

if (nQueen(arr, x + 1, n))

return true;

arr[x][col] = 0;

}

}

return false;

}

int main()

{

int n;

cout<<" Enter the box size:"<<endl;

cin >> n;

int \*\*arr = new int \*[n];

for (int i = 0; i < n; i++)

{

arr[i] = new int[n];

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

{

arr[i][j] = 0;

}

}

if (nQueen(arr, 0, n))

{

for (int i = 0; i < n; i++)

{

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

{

cout << arr[i][j] << " ";

}

cout << endl;

}

}

}