***N-Queen Problem***

public class Main {

static int count;

private static boolean kyaQueenRakhun(int[][] board, int cr, int cc, int n) {

for(int row = 0; row <= cr - 1; row++) {

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

return false;

}

}

int row = cr;

int col = cc;

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

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

return false;

}

row--;

col--;

}

row = cr;

col = cc;

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

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

return false;

}

row--;

col++;

}

return true;

}

private static void queenComb(int[][] board, int cr, int n) {

// TODO Auto-generated method stub

if(cr == n) {

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

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

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

System.out.print("Q ");

} else {

System.out.print("\_ ");

}

}

System.out.println();

}

System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

count++;

return;

}

for(int cc = 0; cc < n; cc++) {

if(kyaQueenRakhun(board, cr, cc, n)) {

board[cr][cc] = 1;

queenComb(board, cr + 1, n);

board[cr][cc] = 0;

}

}

}

public static void main(String[] args) {

// TODO Auto-generated method stub

int n = 8;

int[][] board = new int[n][n];

queenComb(board, 0, n);

System.out.println(count);

}

}