Experiment Number: 6

NAME: Harshwardhan Patil  ROLLNO: 50

CLASS: TY-IT-B BATCH: B1

**Problem Statement:**

**Time complexity analysis of Huffman Encoding using Greedy strategy**

public class NQueens {

public static void main(String[] args) {

int n = 4;

boolean[][] board = new boolean[n][n];

System.out.println(queens(board, 0));

}

static int queens(boolean[][] board, int row) {

if (row == board.length) {

display(board);

System.out.println();

return 1;

}

int count = 0;

// placing the queen and checking for every row and col

for (int col = 0; col < board.length; col++) {

// place the queen if it is safe

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

board[row][col] = true;

count += queens(board, row + 1);

board[row][col] = false;

}

}

return count;

}

private static boolean isSafe(boolean[][] board, int row, int col) {

// check vertical row

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

if (board[i][col]) {

return false;

}

}

// diagonal left

int maxLeft = Math.min(row, col);

for (int i = 1; i <= maxLeft; i++) {

if(board[row-i][col-i]) {

return false;

}

}

// diagonal right

int maxRight = Math.min(row, board.length - col - 1);

for (int i = 1; i <= maxRight; i++) {

if(board[row-i][col+i]) {

return false;

}

}

return true;

}

private static void display(boolean[][] board) {

for(boolean[] row : board) {

for(boolean element : row) {

if (element) {

System.out.print("Q ");

} else {

System.out.print("X ");

}

}

System.out.println();

}

}

}