```java
import java.io.*;
import java.util.Arrays;

class NQueen{

static int N = 8;

static void printSolution(int board[][])
{
  int N = board.length;
  for(int i = 0; i < N; i++)
  {
    for(int j = 0; j < N; j++)
      System.out.printf("%2d ", board[i][j]);

    System.out.printf("\n");
  }
}

static boolean isSafe(int row, int col,
,         int slashCode[][],
,         int backslashCode[][],
,         boolean rowLookup[],
,         boolean slashCodeLookup[],
,         boolean backslashCodeLookup[])
{
  if (slashCodeLookup[slashCode[row][col]] ||
    backslashCodeLookup[backslashCode[row][col]] ||
    rowLookup[row])
    return false;

  return true;
}

static boolean solveNQueensUtil(
, int board[][], int col, int slashCode[][],
, int backslashCode[][], boolean rowLookup[],
, boolean slashCodeLookup[],
, boolean backslashCodeLookup[])
{
  int N = board.length;

  if (col >= N)
    return true;

  for(int i = 0; i < N; i++)
  {
    if (isSafe(i, col, slashCode, backslashCode,
        rowLookup, slashCodeLookup,
        backslashCodeLookup))
    {
      board[i][col] = 1;
      rowLookup[i] = true;
      slashCodeLookup[slashCode[i][col]] = true;
      backslashCodeLookup[backslashCode[i][col]] = true;

      if (solveNQueensUtil(
          board, col + 1, slashCode,
          backslashCode, rowLookup,
          slashCodeLookup,
          backslashCodeLookup))
        return true;

      board[i][col] = 0;
      rowLookup[i] = false;
      slashCodeLookup[slashCode[i][col]] = false;
      backslashCodeLookup[backslashCode[i][col]] = false;
    }
  }

  return false;
}

static boolean solveNQueens()
{
  int board[][] = new int[N][N];

  // Helper matrices
  int slashCode[][] = new int[N][N];
  int backslashCode[][] = new int[N][N];

  boolean[] rowLookup = new boolean[N];

  boolean slashCodeLookup[] = new boolean[2 * N - 1];
  boolean backslashCodeLookup[] = new boolean[2 * N - 1];

  for(int r = 0; r < N; r++)
    for(int c = 0; c < N; c++)
    {
      slashCode[r] = r + c;
      backslashCode[r] = r - c + 7;
    }

  if (solveNQueensUtil(board, 0, slashCode,
            backslashCode, rowLookup,
            slashCodeLookup,
            backslashCodeLookup) == false)
  {
    System.out.printf("Solution does not exist");
    return false;
  }

  printSolution(board);
  return true;
}

public static void main(String[] args)
{
  solveNQueens();
}
}
```