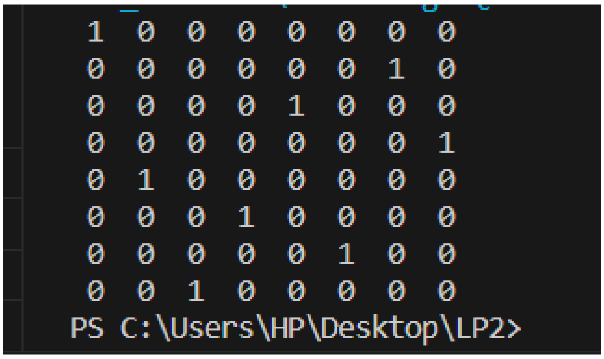
8 QUEENS PROBLEM

```
public class EightQueensProblem {
    static int N = 8;
    static boolean isSafe(int[][] board, int row, int col) {
        int i, j;
        for (i = 0; i < col; i++)
            if (board[row][i] == 1)
                return false;
        for (i = row, j = col; i >= 0 && j >= 0; i--, j--)
            if (board[i][j] == 1)
                return false;
        for (i = row, j = col; j >= 0 && i < N; i++, j--)
            if (board[i][j] == 1)
                return false;
        return true;
    }
    static boolean solveNQUtil(int[][] board, 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;
    }
    static void printSolution(int[][] board) {
        for (int i = 0; i < N; i++) {
            for (int j = 0; j < N; j++)
                System.out.print(" " + board[i][j] + " ");
            System.out.println();
        }
    }
    static void solveNQ() {
        int[][] board = new int[N][N];
        if (!solveNQUtil(board, 0)) {
            System.out.println("Solution does not exist");
            return;
        }
        printSolution(board);
    }
    public static void main(String[] args) {
        solveNQ();
    }
}
```

OUTPUT:



```
 1  0  0  0  0  0  0  0
 0  0  0  0  0  0  1  0
 0  0  0  0  1  0  0  0
 0  0  0  0  0  0  0  1
 0  1  0  0  0  0  0  0
 0  0  0  1  0  0  0  0
 0  0  0  0  0  1  0  0
 0  0  1  0  0  0  0  0
PS C:\Users\HP\Desktop\LP2>
```