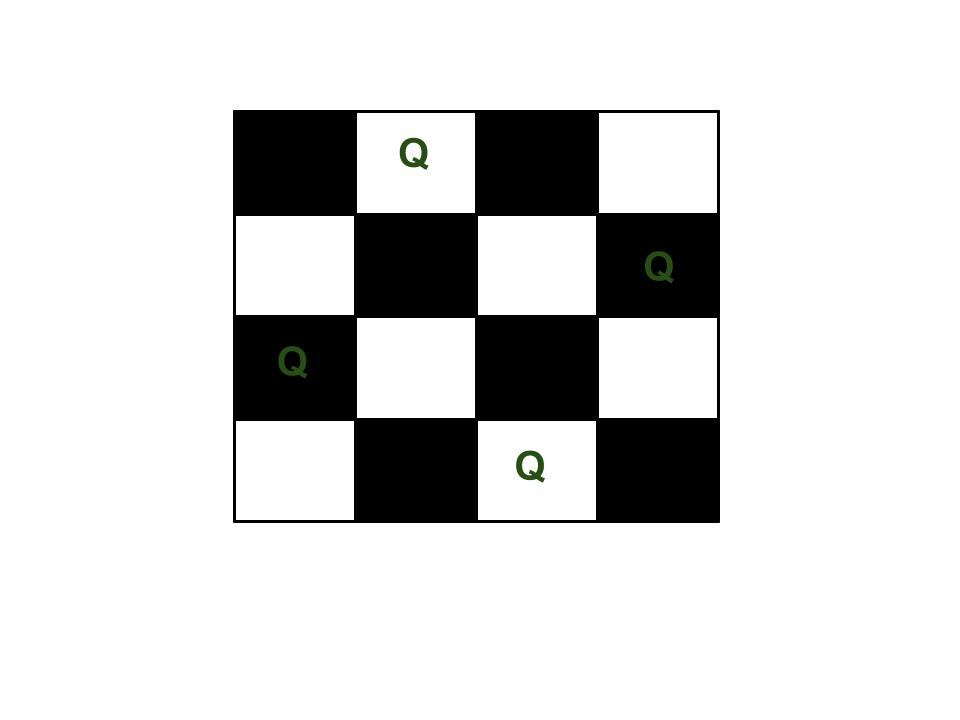
|  |  |
| --- | --- |
|  | DEPARTMENT OF ARTIFICIAL INTELLIGNECE & DATA SCIENCE |

|  |  |
| --- | --- |
| Subject: Analysis of Algorithm | Course Code: CSC402 |
| Semester: 4 | Course: AI & DS |
| Laboratory No: 205 | Name of Subject Teacher: Pramod Bhavarthe |
| Name of Student: Sahil Shaikh | Roll Id: VU2S2223012 |

|  |  |
| --- | --- |
| Title of Practical | N-queen problem. |

**Theory –**

**The N Queen** is the problem of placing N chess queens on an N×N chessboard so that no two queens attack each other. For example, the following is a solution for 4 Queen problem.

**Program –**

global N

N = 4

def printSolution(board):

    for i in range(N):

        for j in range(N):

            print (board[i][j],end=' ')

        print()

def isSafe(board, row, col):

    for i in range(col):

        if board[row][i] == 1:

            return False

    for i, j in zip(range(row, -1, -1), range(col, -1, -1)):

        if board[i][j] == 1:

            return False

    for i, j in zip(range(row, N, 1), range(col, -1, -1)):

        if board[i][j] == 1:

            return False

    return True

def solveNQUtil(board, col):

    if col >= N:

        return True

    for i in range(N):

        if isSafe(board, i, col):

            board[i][col] = 1

            if solveNQUtil(board, col + 1) == True:

                return True

            board[i][col] = 0

    return False

    board = [ [0, 0, 0, 0],

            [0, 0, 0, 0],

            [0, 0, 0, 0],

            [0, 0, 0, 0]

            ]

    if solveNQUtil(board, 0) == False:

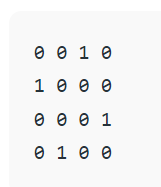
        print ("Solution does not exist")

        return False

    printSolution(board)

    return True

solveNQ()

****

**Output –**

**Conclusion –**

**Therefore, we have successfully understood and Implemented N-queen problem.**

|  |  |  |  |
| --- | --- | --- | --- |
| **Grade and Dated Signature of Teacher** | **Total (10)** | **Remark** | **Dated signature of teacher** |
|  |  |  |