Eight Queens Problem

Aim: To **solve the 8-Queens problem** using **backtracking**.

#solved 8 Queens problem using backtracking method.

```
N = 8

ld = [0] * (2 * N)

rd = [0] * (2 * N)

cl = [0] * N


def printSolution(board):
    for i in range(N):
        for j in range(N):
            if board[i][j] == 1:
                    print("Q", end=" ")
            else:
                    print(".", end=" ")
        print()
def solveNQUtil(board, col):
    if col >= N:
        return True
    for row in range(N):
        if (ld[row - col + N - 1] != 1 and
                    rd[row + col] != 1 and cl[row] != 1):
            board[row][col] = 1
            ld[row - col + N - 1] = rd[row + col] = cl[row] = 1


            if solveNQUtil(board, col + 1):
```

```
                return True


            board[row][col] = 0
            ld[row - col + N - 1] = rd[row + col] = cl[row] = 0
    return False
def solveNQ():
    board = [[0 for _ in range(N)] for _ in range(N)]
    if solveNQUtil(board, 0) == False:
        print("Solution does not exist")
        return False
    printSolution(board)
    return True
if __name__ == '__main__'
solveNQ()
```

Result:

```
. Q . . . . . .
. . . . Q . . .
. . . . . . Q .
. . . . . . . Q
Q . . . . . . .
. . Q . . . . .
. . . . . Q . .
. . . Q . . . .
```