8 QUEENS PROBLEM

AIM:

To solve the 8 queens problem with all constraints and to print the solution of placing the 8 queens in their respective positions.

CODE:

```
def print_board(board):
    for row in board:
        print(" ".join(['Q' if x else '.' for x in row]))
    print()

def is_safe(board, row, col):
    for i in range(row):
        if board[i][col] == 1:
            return False

    for i, j in zip(range(row-1, -1, -1), range(col-1, -1, -1)):
        if board[i][j] == 1:
            return False

    for i, j in zip(range(row-1, -1, -1), range(col+1, len(board), 1)):
        if board[i][j] == 1:
            return False
    return True

def solve_queens(board, row):
    if row == len(board):
        print_board(board)
        return True
    found_solution = False
    for col in range(len(board)):
        if is_safe(board, row, col):
            board[row][col] = 1
            found_solution = solve_queens(board, row + 1) or
```

```
found_solution
            board[row][col] = 0

    return found_solution

board = [[0 for _ in range(8)] for _ in range(8)]
solve_queens(board, 0)
```

OUTPUT:

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

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

RESULT:

The program have been compiled and executed successfully and the output have been verified.