EXP 2 :-

PROGRAM :-

def print\_solution(board):

for row in board:

print(" ".join("Q" if col else "." for col in row))

print("\n")

def is\_safe(board, row, col):

for i in range(col):

if board[row][i]:

return False

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

if board[i][j]:

return False

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

if board[i][j]:

return False

return True

def solve\_n\_queens\_util(board, col):

if col >= len(board):

print\_solution(board)

return True

res = False

for i in range(len(board)):

if is\_safe(board, i, col):

board[i][col] = True

res = solve\_n\_queens\_util(board, col + 1) or res

board[i][col] = False

return res

def solve\_n\_queens(n):

board = [[False for \_ in range(n)] for \_ in range(n)]

if not solve\_n\_queens\_util(board, 0):

print("No solution exists")

# Solve the 8 Queen Problem

solve\_n\_queens(8)

OUTPUT :-

