**NAME: - Rushikesh Sunil Mangalkar ROLL NO- 3202007**

**ASSIGNMENT 4**

**CODE:**

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

for i in range(col):

if board[row][i] == 1:

return False

i, j = row, col

while i >= 0 and j >= 0:

if board[i][j] == 1:

return False

i -= 1

j -= 1

i, j = row, col

while i < n and j >= 0:

if board[i][j] == 1:

return False

i += 1

j -= 1

return True

def solve\_nq\_limited(board, col, n, solutions, required\_solutions):

if col >= n:

solution = []

for row in board:

solution.append(row.copy())

solutions.append(solution)

return len(solutions) >= required\_solutions

for i in range(n):

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

board[i][col] = 1

if solve\_nq\_limited(board, col + 1, n, solutions, required\_solutions):

return True

board[i][col] = 0

return False

def find\_solutions(n, required\_solutions):

board = []

for i in range(n):

row = []

for j in range(n):

row.append(0)

board.append(row)

solutions = []

solve\_nq\_limited(board, 0, n, solutions, required\_solutions)

return solutions

def print\_solutions(solutions, required\_solutions):

total\_solutions = len(solutions)

if total\_solutions == 0:

print("No solutions exist")

return

if total\_solutions < required\_solutions:

print(f"\nNote: Only {total\_solutions} solutions exist for {n}-Queens problem.")

print(f"\nFound {total\_solutions} solution(s):")

for i, solution in enumerate(solutions, 1):

print(f"\nSolution {i}:")

for row in solution:

print(" ".join(str(x) for x in row))

n = int(input("Enter the size of the board (n): "))

required\_solutions = int(input(f"How many solutions would you like to see? : "))

if required\_solutions < 1:

print("Please enter a positive number of solutions.")

else:

solutions = find\_solutions(n, required\_solutions)

print\_solutions(solutions, required\_solutions)

**OUTPUT:**

Enter the size of the board (n): 8

How many solutions would you like to see? : 6

Found 6 solution(s):

Solution 1:

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

Solution 2:

1 0 0 0 0 0 0 0

0 0 0 0 0 0 1 0

0 0 0 1 0 0 0 0

0 0 0 0 0 1 0 0

0 0 0 0 0 0 0 1

0 1 0 0 0 0 0 0

0 0 0 0 1 0 0 0

0 0 1 0 0 0 0 0

Solution 3:

1 0 0 0 0 0 0 0

0 0 0 0 0 1 0 0

0 0 0 0 0 0 0 1

0 0 1 0 0 0 0 0

0 0 0 0 0 0 1 0

0 0 0 1 0 0 0 0

0 1 0 0 0 0 0 0

0 0 0 0 1 0 0 0

Solution 4:

1 0 0 0 0 0 0 0

0 0 0 0 1 0 0 0

0 0 0 0 0 0 0 1

0 0 0 0 0 1 0 0

0 0 1 0 0 0 0 0

0 0 0 0 0 0 1 0

0 1 0 0 0 0 0 0

0 0 0 1 0 0 0 0

Solution 5:

0 0 0 0 0 1 0 0

1 0 0 0 0 0 0 0

0 0 0 0 1 0 0 0

0 1 0 0 0 0 0 0

0 0 0 0 0 0 0 1

0 0 1 0 0 0 0 0

0 0 0 0 0 0 1 0

0 0 0 1 0 0 0 0

Solution 6:

0 0 0 1 0 0 0 0

1 0 0 0 0 0 0 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 0 0 0 1 0

0 0 1 0 0 0 0 0

0 0 0 0 0 1 0 0