

Name Manasi Kale

Roll no B13

* def is\_safe(row, col, columns, diagonals1, diagonals2):

return col not in columns and (row - col) not in diagonals1 and (row + col) not in diagonals2

def solve\_n\_queens(n):

solutions = []

board = [-1] \* n # board[i] = column of queen in row i

columns = set()

diagonals1 = set() # row - col

diagonals2 = set() # row + col

def backtrack(row):

if row == n:

# Found a valid solution

solutions.append(board[:])

return

for col in range(n):

if is\_safe(row, col, columns, diagonals1, diagonals2):

# Place queen

board[row] = col

columns.add(col)

diagonals1.add(row - col)

diagonals2.add(row + col)

backtrack(row + 1)

# Remove queen (backtrack)

columns.remove(col)

diagonals1.remove(row - col)

diagonals2.remove(row + col)

board[row] = -1

backtrack(0)

return solutions

def print\_solutions(solutions, n):

for idx, sol in enumerate(solutions):

print(f"Solution {idx + 1}:")

for row in sol:

line = ['.'] \* n

line[row] = 'Q'

print(" ".join(line))

print()

# Example: Solve 8-Queens Problem

n = 8

solutions = solve\_n\_queens(n)

print(f"Total solutions for {n}-Queens: {len(solutions)}")

print\_solutions(solutions, n)