**AI Assignment 4**

**Code:**

class NQueens:

def \_\_init\_\_(self, n):

self.n = n

self.board = [[0] \* n for \_ in range(n)]

self.solutions = []

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

for i in range(row):

if self.board[i][col] == 1:

return False

if 0 <= col - row + i < self.n and self.board[i][col - row + i] == 1:

return False

if 0 <= col + row - i < self.n and self.board[i][col + row - i] == 1:

return False

return True

def solve\_backtracking(self, row=0):

if row == self.n:

self.solutions.append([row[:] for row in self.board])

return True

for col in range(self.n):

if self.is\_safe(row, col):

self.board[row][col] = 1

self.solve\_backtracking(row + 1)

self.board[row][col] = 0

def print\_solutions(self):

for i, solution in enumerate(self.solutions):

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

for row in solution:

print(" ".join("Q" if cell == 1 else "-" for cell in row))

print()

n\_queens = NQueens(4)

n\_queens.solve\_backtracking()

print("Solutions:")

n\_queens.print\_solutions()

**Output:**

Solutions:

Solution 1:

- Q - -

- - - Q

Q - - -

- - Q -

Solution 2:

- - Q -

Q - - -

- - - Q

- Q - -