def nearest\_neighbor\_tsp(dist):

    n = len(dist)

    best\_cost = float('inf')

    best\_path = []

    for start in range(n):

        visited = [start]

        cost = 0

        current = start

        while len(visited) < n:

            next\_city = min(

                (city for city in range(n) if city not in visited),

                key=lambda c: dist[current][c]

            )

            cost += dist[current][next\_city]

            visited.append(next\_city)

            current = next\_city

        cost += dist[current][start]  # Return to start

        visited.append(start)

        print(f"Start at {start}: Path = {visited}, Cost = {cost}")

        if cost < best\_cost:

            best\_cost = cost

            best\_path = visited

    print("\nBest Path:", best\_path)

    print("Minimum Cost:", best\_cost)

# --- Input Section ---

n = int(input("Enter number of cities: "))

print("Enter the distance matrix:")

dist = [list(map(int, input().split())) for \_ in range(n)]

nearest\_neighbor\_tsp(dist)

Enter number of cities: 3

Enter the distance matrix:

3 5 7

1 3 7

3 8 9

Start at 0: Path = [0, 1, 2, 0], Cost = 15

Start at 1: Path = [1, 0, 2, 1], Cost = 16

Start at 2: Path = [2, 0, 1, 2], Cost = 15

Best Path: [0, 1, 2, 0]

Minimum Cost: 15