**AI FOR LLM- CSA1704**

**9. Travelling Salesman Problem (TSP)**

**CODE:**

import itertools

# Distance matrix between cities (symmetric)

# Example: 4 cities

distance\_matrix = [

[0, 10, 15, 20],

[10, 0, 35, 25],

[15, 35, 0, 30],

[20, 25, 30, 0]

]

def calculate\_path\_cost(path, matrix):

cost = 0

for i in range(len(path) - 1):

cost += matrix[path[i]][path[i + 1]]

cost += matrix[path[-1]][path[0]] # Return to start

return cost

def tsp\_brute\_force(matrix):

n = len(matrix)

cities = list(range(n))

min\_cost = float('inf')

best\_path = []

for perm in itertools.permutations(cities):

cost = calculate\_path\_cost(perm, matrix)

if cost < min\_cost:

min\_cost = cost

best\_path = perm

return best\_path, min\_cost

# Run the TSP solver

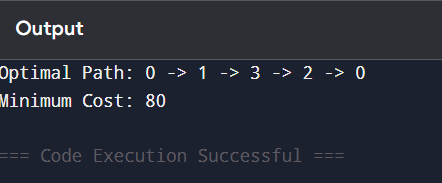
path, cost = tsp\_brute\_force(distance\_matrix)

# Display results

print("Optimal Path:", " -> ".join(map(str, path)) + f" -> {path[0]}")

print("Minimum Cost:", cost)

**OUTPUT:**

****