**Assignment No : 1**

**Title: Implement Travelling Salesman Problem using Brute force and Dynamic Programming approach.**

import itertools

def calculate\_distance(city1, city2):

return ((city1[0] - city2[0])\*\*2 + (city1[1] - city2[1])\*\*2)\*\*0.5

def total\_distance(path, cities):

distance = 0

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

distance += calculate\_distance(cities[path[i]], cities[path[i+1]])

distance += calculate\_distance(cities[path[-1]], cities[path[0]])

return distance

def main():

num\_cities = int(input("Enter the number of cities: "))

cities = []

for i in range(num\_cities):

x, y = map(int, input(f"Enter coordinates for city {i+1} (x y): ").split()) cities.append((x, y))

min\_distance = float('inf')

optimal\_path = []

for path in itertools.permutations(range(num\_cities)):

distance = total\_distance(path, cities)

if distance < min\_distance:

min\_distance = distance

optimal\_path = path

print("Optimal path:", optimal\_path)

print("Optimal distance:", min\_distance)

main()

**OUTPUT:**

Enter the number of cities: 4

Enter coordinates for city 1 (x y): 12 13 Enter coordinates for city 2 (x y): 2 4 Enter coordinates for city 3 (x y): 23 15 Enter coordinates for city 4 (x y): 23 5 Optimal path: (0, 1, 3, 2)

Optimal distance: 55.65775997620129

**Time complexity:**

O(n2\*2n)