1. 142. You are given a list of cities represented by their coordinates. Develop a program that utilizes exhaustive search to solve the TSP. The program should:
    1. Define a function distance(city1, city2) to calculate the distance between two cities (e.g., Euclidean distance).

Code:

```
import math

def distance(city1, city2):
    return math.sqrt((city1[0] - city2[0]) ** 2 + (city1[1] - city2[1]) ** 2)
from itertools import permutations

def tsp_exhaustive_search(cities):

    all_permutations = permutations(cities)

    min_distance = float('inf')
    best_tour = None
    for perm in all_permutations:
        current_distance = 0
        for i in range(len(perm) - 1):
            current_distance += distance(perm[i], perm[i+1])
        current_distance += distance(perm[-1], perm[0])


        if current_distance < min_distance:
            min_distance = current_distance
            best_tour = perm

    return best_tour, min_distance

cities = [(0, 0), (1, 1), (2, 2), (3, 3)]

best_tour, min_distance = tsp_exhaustive_search(cities)
print("Best Tour:", best_tour)
print("Minimum Distance:", min_distance)
```

output:

```
PS C:\Users\karth> & C:/Users/karth/AppData/Local/Programs/Python/Python312/python.exe c:/Users/karth/OneDrive/Documents/OriginLab/problems.py
Best Tour: ((0, 0), (3, 3), (2, 2), (1, 1))
Minimum Distance: 8.48528137423857
PS C:\Users\karth>
```

Time complexity:f(n)=0(m*n)