CSP

By: **Vivek Vittal Biragoni, 211AI041**
19/05/23

Problem Statement
The problem is to find optimal locations for a given number of airports in a set of cities. The objective is to minimize the sum of squared distances between each city and its nearest airport. The cities are represented by an adjacency matrix, where each cell contains the distance between two cities. The task is to determine the locations for the airports and calculate the sum of squared distances.

Design
The solution approach consists of two main parts:

1. TSP Solution: A function `tsp(adj_matrix, city_names)` is implemented to find a solution to the Traveling Salesman Problem (TSP). It uses a greedy algorithm to construct a path that visits each city exactly once and returns to the starting city. The function takes an adjacency matrix and a list of city names as inputs and returns the TSP path.

2. Clustering Solution: A function `minimize_squared_distances(adj_matrix, city_names, num_airports)` is implemented to find the optimal airport locations using k-means clustering. It converts the given adjacency matrix to a distance matrix and applies k-means clustering with the specified number of airports. It then calculates the sum of squared distances between each city and its nearest airport. The function takes the adjacency matrix, city names, and the number of airports as inputs and returns the sum of squared distances and the coordinates of the airport locations.

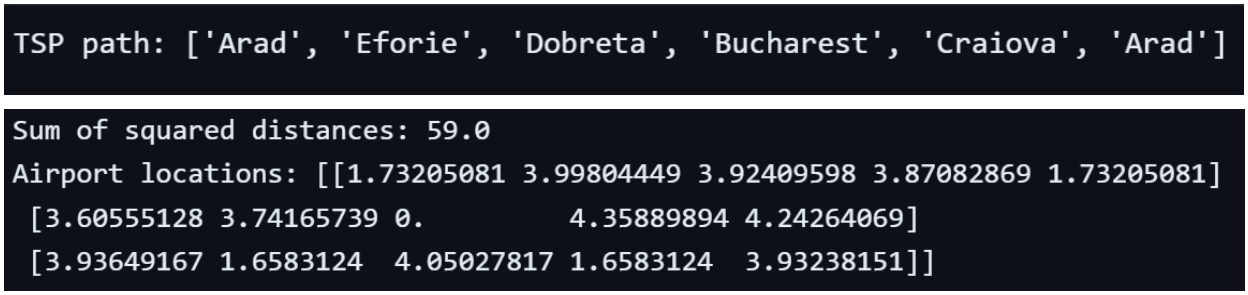
Sample Input and Output
```
# TSP Solution
adj_matrix = [
    [0, 15, 13, 16, 12],
    [15, 0, 14, 11, 17],
    [13, 14, 0, 19, 18],
    [16, 11, 19, 0, 14],
    [12, 17, 18, 14, 0]
]
city_names = ["Arad", "Bucharest", "Craiova", "Dobreta", "Eforie"]

tsp_path = tsp(adj_matrix, city_names)
print("TSP path:", tsp_path)
```

```
# Clustering Solution
num_airports = 3

sum_squared_distances, airport_locations = minimize_squared_distances(adj_matrix,
city_names, num_airports)
print("Sum of squared distances:", sum_squared_distances)
print("Airport locations:", airport_locations)
```

Output:

```
TSP path: ['Arad', 'Eforie', 'Dobreta', 'Bucharest', 'Craiova', 'Arad']
Sum of squared distances: 59.0
Airport locations: [[1.73205081 3.99804449 3.92409598 3.87082869 1.73205081]
 [3.60555128 3.74165739 0.         4.35889894 4.24264069]
 [3.93649167 1.6583124  4.05027817 1.6583124  3.93238151]]

```

Output screenshot



In the above sample input and output, the TSP path represents the optimal path that visits each city exactly once and returns to the starting city. The sum of squared distances indicates the quality of the solution, with lower values indicating a better solution. The airport locations represent the coordinates of the optimal locations for the specified number of airports.