2 Approximation Algorithm for Traveling Salesman Problem

A Traveling Salesman Problem (TSP) premise states:

- "Given a set of cities and distance between every pair of cities, the problem is to find the shortest possible route that visits every city exactly once and returns to the starting point."

Two important things about the problem statement are that:

- The salesman visits every city exactly once.
- The goal is to cover the shortest path possible.

The main algorithm follows the next structure :

1. Let vertex 0 be the starting and ending point for the salesman.
2. Construct a Minimum Spanning Tree with 0 as its root, using Prim's Algorithm.
3. List vertices visited in preorder walk/Depth First Search of the constructed MST and add the source node at the end.

The TSP uses a graph as its data structure, to represent the cities and their paths. In this case, we will represent the graph using an adjacency matrix. Adjacency matrices have a characteristic that will help us find the MST, which is the triangle inequality.

- Triangle-Inequality: The least distant path to reach a vertex j from i is always to reach j directly from i, rather than through some other vertex k (or vertices), i.e., dis(i, j) is always less than or equal to dis(i, k) + dist(k, j).

The solution to the TSP problem presented here is the 2 approximation algorithm. It was developed with *Python* scripts (ver. 3.9.2) in a macOS device. It contains three scripts:

- CSVHandler.py
- GraphsMST.py
- Main.py

The most important file here is *GraphsMST.py*, which houses a class named Graph that lets us build graphs and build MSTs out of them.

Execution instructions:
Within the .zip file, there is a folder called *input*. This folder contains a file in CSV format called *graph.csv*, which is from where the code will get the graph of the TSP. The file already contains a simple graph. If there is a need to test another graph, the CSV file can be updated. This file contains a graph as an adjacency matrix.

1. Download and extract the .zip file containing the code.
2. From a terminal or CMD, enter the *TSP-2-approx* folder.
3. Once in the folder, simply type “python3 Main.py” into the terminal (or “python Main.py” for CMD), and the scripts should run.
4. The terminal/CMD will print out the results of the algorithm.