**Background:**

The approach used for solving the problem is an approximation technique based on Minimum Spanning Tree (MST) followed by Depth-First Search (DFS) traversal, aiming to provide a near-optimal solution to the Traveling Salesman Problem (TSP). While this heuristic does not guarantee the absolute shortest route, it delivers a reasonable approximation that performs efficiently. Alternative methods, such as Branch and Bound or Dynamic Programming, are computationally expensive, particularly for large problems like TSP. Two heuristic approaches were considered:

* TSP with Genetic Algorithm (TSP\_GA): A population-based approach that uses random initial populations. While this approach provides a solution, the results exhibit high variability in the tour cost due to the randomness in the initial population.
* TSP with DFS MST: This approximation always yields consistent results with relatively stable tour costs, showing less variability. The time taken to compute the result using this method is roughly 1/12th of the time needed by the Genetic Algorithm.

Through experimentation, it became clear that the problem could be effectively addressed by the DFS MST TSP algorithm, with the key challenge being the incorporation of the requirement that every 10th stop must be a petrol station.

**Problem Formulation:**

Initially, the problem was thought to be a multi-objective TSP, where the additional condition of requiring the driver to visit a petrol station every 10th stop added complexity. However, further analysis revealed that the problem essentially involves a single objective: finding the optimal route with the added constraint that every 10th stop must be a petrol station. Thus, the problem is reduced to finding the best route while considering petrol stations as a constraint, not an additional objective.

**Methodology**:

The solution is based on the DFS MST TSP, which is an approximation algorithm. This method consists of two main parts:

Part 1: DFS MST TSP Algorithm (on Stops 0 to 48)

This part of the algorithm constructs a Minimum Spanning Tree (MST) over the delivery destinations (Stops 1 to 48), starting from the depot (Stop 0). After building the MST, a Depth-First Search (DFS) is performed to generate a route. This forms the TSP approximation where the driver visits each delivery destination exactly once. In the end it appends the list with the start point as we want start and ending points to be same.

Part 2: Petrol Station Selection

In this part, the algorithm accounts for the requirement that every 10th stop must be a petrol station. Given the MST tour generated in Part 1, we introduce a brute-force approach to find the best petrol stations to visit. Specifically, for every 9th node, the algorithm identifies the optimal petrol station to visit as the 10th stop, minimizing the additional travel cost. This can be expressed as the following formula:

**Code Summary**

The code begins by loading data from a CSV file containing stops with their respective coordinates. The Euclidean distance between all pairs of stops is calculated and stored in a matrix. A Minimum Spanning Tree (MST) is then constructed using Prim's Algorithm to connect all delivery destinations with the shortest possible edges. A Depth-First Search (DFS) is performed on the MST to generate an approximate TSP path, which is then extended to form a cyclic route starting and ending at the depot. The code use Brute force ensuring that a petrol station is visited every tenth stop, minimizing the additional distance. Finally, the computed route is visualized on a 2D plane, and the total distance of the route, including delivery destinations and petrol stations, is calculated.

**Appendix**

**Appendix A: Shortest Path**



**Appendix B: Map of shortest route**

