Data Structures and Algorithms
Electric Vehicle Charging

- By **Yash Bagia** (20395995)

Executive Summary:

The EVCharging program is a sophisticated tool designed to optimize electric vehicle charging by providing users with various functionalities such as locating charging stations, determining optimal charging paths, and analyzing charging costs. This formal document aims to provide an in-depth understanding of the program's architecture, highlighting the key data structures, algorithms, and their implementations. Through thorough analysis and explanation, this document showcases the program's efficiency and effectiveness in addressing the challenges of electric vehicle infrastructure management.

Table of Contents:

Introduction:

Electric vehicles (EVs) have emerged as a sustainable transportation solution, but efficient charging infrastructure is essential for their widespread adoption. The EVCharging program addresses this need by offering functionalities to locate charging stations, find optimal charging paths, and minimize charging costs. This document elucidates the program's inner workings, focusing on data structures, algorithms, and complexity analysis.

Data Structures:

The EVCharging program employs several key data structures:

Map: Utilized to store information about charging stations, indexed by their location index.

Weighted Graph: Implemented using adjacency lists to represent connections between locations and associated edge weights.

Priority Queue: Utilized in Dijkstra's algorithm and Prim's algorithm for minimum spanning trees.

Vector: Used for storing and manipulating lists of elements efficiently.

Algorithms:

Dijkstra's Algorithm:

```
Dijkstra(graph, source):
    Initialize distances and priority queue
    Set distance to source as 0 and enqueue source
    while priority queue is not empty:
        Extract vertex u with minimum distance
        For each neighbor v of u:
            Update distance if shorter path found
```

Prim's Algorithm (Minimum Spanning Tree):

Prim(graph, source):

Initialize keys and priority queue

Set key for source as 0 and enqueue source

while priority queue is not empty:

Extract vertex u with minimum key

For each adjacent vertex v:

Update key if edge weight is less than current key

Minimum Span Tree:

```
Shortest power line from Parramatta to all other stations.
                From                  To          Length
           Kingswood             Penrith             3.7
          Rooty hill           Kingswood            20.8
           Kingswood             Windsor              22
           Blacktown          Rooty hill            10.3
          Parramatta           Blacktown            15.3
          Parramatta        Olympic Park            11.6
        Olympic Park             Burwood             7.1
             Burwood           Lilyfield             6.8
           Lilyfield             Central             6.6
             Central        North Sydney             7.5
        North Sydney           Lane Cove             8.6
           Lane Cove        Macquarie Pk             5.8
           Wahroonga       Pennant Hills             5.4
         Glendenning          Kings Park              12
          Rooty hill         Glendenning             3.7
             Central              Mascot             6.4
             Burwood          Kingsgrove             8.5
        Olympic Park           Bankstown            10.1
           Bankstown           Liverpool            14.5
           Liverpool            Prestons             7.3
            Prestons        Campbelltown            17.6
              Pymble           Wahroonga               4
        Macquarie Pk              Pymble             5.9

Total length of the power line: 221.5
```

Sorting Algorithm (Used in listAvailableStationsByPrice):

SortByPrice(stations):

Sort stations by charging price using custom comparator

Complexity Analysis:

Dijkstra's Algorithm: O((V + E) log V), where V is the number of vertices and E is the number of edges.

Prim's Algorithm: O(V^2), where V is the number of vertices.

Sorting Algorithm: O(n log n), where n is the number of available charging stations.

Conclusion:

The EVCharging program exemplifies the effective utilization of data structures and algorithms to optimize electric vehicle charging infrastructure. By employing techniques such as Dijkstra's algorithm and Prim's algorithm, it offers efficient solutions for locating charging stations, determining optimal charging paths, and minimizing charging costs. This formal document provides a comprehensive overview of the program's architecture and functionality, demonstrating its capability to address the challenges of electric vehicle infrastructure management professionally and efficiently.