**Assignment 3 DAA**

**1. Objective**

The goal of this assignment was to apply **Prim’s** and **Kruskal’s algorithms** to find the **Minimum Spanning Tree (MST)** of a weighted undirected graph representing a city’s transportation network.  
The task aimed to minimize total road construction cost while ensuring all districts remain connected.

**2. Input Data**

The input data was provided in the file **ass\_3\_input.json**, containing two graphs:

| **Graph ID** | **Vertices (districts)** | **Edges (roads)** |
| --- | --- | --- |
| 1 | 5 (A, B, C, D, E) | 7 |
| 2 | 4 (A, B, C, D) | 5 |

Each edge represents a possible road with a specific construction cost (weight).

**3. Results**

The algorithms were executed for both graphs, and the results were saved in **ass\_3\_output.json**.

**Graph 1 Results**

| **Algorithm** | **MST Edges** | **Total Cost** | **Operations** | **Execution Time (ms)** |
| --- | --- | --- | --- | --- |
| **Prim** | B–C (2), A–C (3), B–D (5), D–E (6) | **16** | 42 | 1.52 |
| **Kruskal** | B–C (2), A–C (3), B–D (5), D–E (6) | **16** | 37 | 1.28 |

**Graph 2 Results**

| **Algorithm** | **MST Edges** | **Total Cost** | **Operations** | **Execution Time (ms)** |
| --- | --- | --- | --- | --- |
| **Prim** | A–B (1), B–C (2), C–D (3) | **6** | 29 | 0.87 |
| **Kruskal** | A–B (1), B–C (2), C–D (3) | **6** | 31 | 0.92 |

4. Analysis and Comparison

**Correctness**

* Both algorithms produced **identical total MST costs** for each graph (16 and 6), confirming correctness.
* The structure of MST was also the same for both algorithms.

**Performance Comparison**

| **Criterion** | **Prim’s Algorithm** | **Kruskal’s Algorithm** |
| --- | --- | --- |
| **Approach** | Grows MST one vertex at a time using a priority queue | Adds edges in increasing order of weight using Union-Find |
| **Best for** | Dense graphs (many edges) | Sparse graphs (few edges) |
| **Implementation complexity** | Slightly more complex due to adjacency checks | Easier to implement with sorting |
| **Performance (Graph 1)** | Slightly slower (1.52 ms, 42 ops) | Faster (1.28 ms, 37 ops) |
| **Performance (Graph 2)** | Slightly faster (0.87 ms) | Slightly slower (0.92 ms) |

5. Conclusions

* Both algorithms effectively minimized road construction cost, producing the same MST result.
* **Kruskal’s algorithm** performed slightly better on **sparser graphs** (Graph 1), while  
  **Prim’s algorithm** was more efficient on **smaller or denser graphs** (Graph 2).
* The difference in execution time was minor, but operation counts show Kruskal may be more efficient when sorting is less expensive relative to the number of edges.
* **Recommendation:**
  + Use **Prim’s algorithm** for **dense city networks** with many potential roads.
  + Use **Kruskal’s algorithm** for **large or sparse networks** with fewer possible connections.