Baptiste Saliba

CS\_428

Programming Assignment 3

**Compiling:**

g++ -std=c++14 main.cpp -o main

**Execution**

./main <source\_node>

**Part A, B, C**

3 sample graphs saved as `GRAPHS` in the `TESTS` object within the main.cpp file.

Ran Djikstra's algorithm on all 3 of these GRAPHS with `u` as the source and had the output saved in the `tables.txt` file. In the `tables.txt` file, Part A is stored under the `TEST1`, Part B under `TEST2`, and Part C under `TEST3`.

Screenshot:  
A picture containing road, black, sitting, monitor

Description automatically generated

**Design Decisions**

- *Graph representation:*

Map from node\_name to node's adjancency matrix.

Considered 2D adjacency matrix; however a map which seemed to make more sense and felt easier to read.

- *Node representation:*

Nodes are simply represented by a char that can be used to index into the Graph which will return that node's adjacency list.

Making another data structure seemed wasteful and unnecessary;

- *Edge representation:*

Edges were represented as a struct which has a `NODE` destination and length, very standard.

- *N' representation:*

A C++ set of `NODE`S. This would prevent duplicates which could create edge cases.

- *Distance Tracking:*

Tracked by a map of `NODE` to current path length to that `NODE`. This distance is updated every time a `NODE` is added to N'.

- *Back Tracking*:

Utilized a map from `NODE` to `NODE` which kept track of the node used to access the destination node. This map is updated every time the distance data structure is updated.