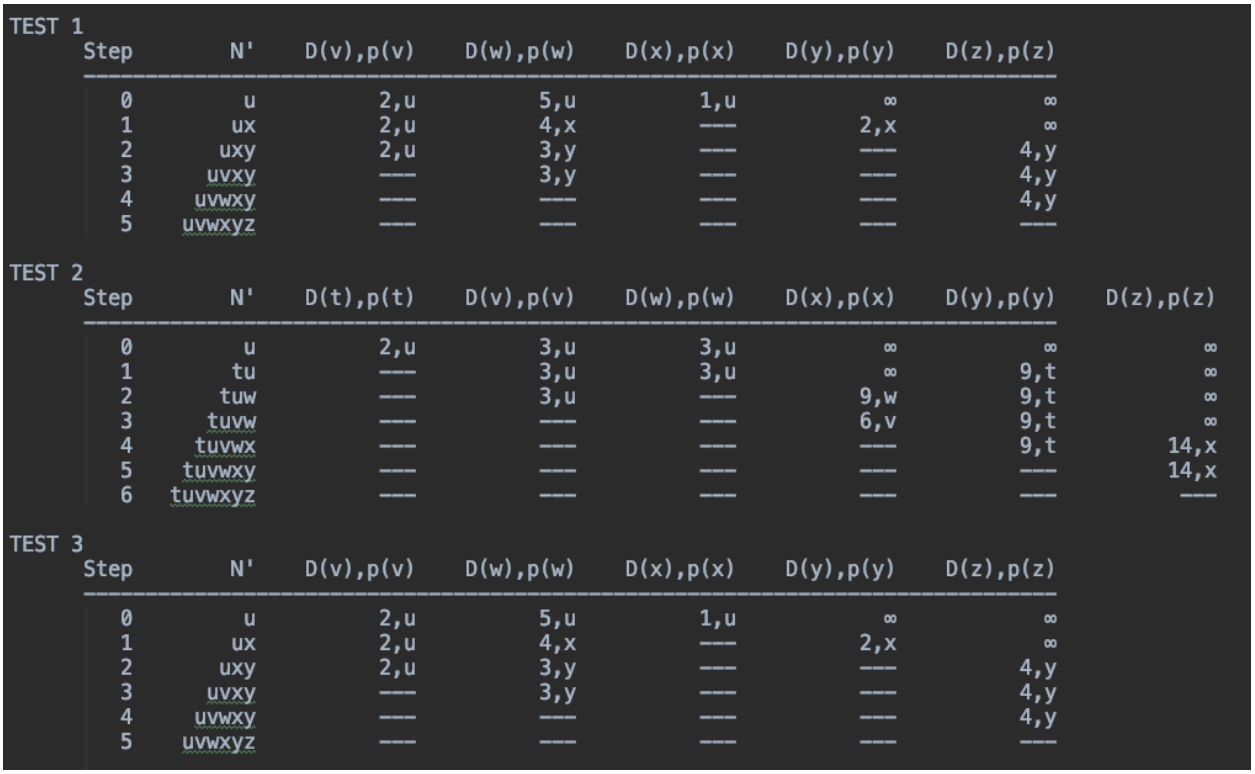
Baptiste Saliba
CS_428
Programming Assignment 3

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:



```
TEST 1
      Step        N'    D(v),p(v)    D(w),p(w)    D(x),p(x)    D(y),p(y)    D(z),p(z)
      ---------------------------------------------------------------------------------
        0          u          2,u          5,u          1,u            ∞            ∞
        1         ux          2,u          4,x          ---          2,x            ∞
        2        uxy          2,u          3,y          ---          ---          4,y
        3       uvxy          ---          3,y          ---          ---          4,y
        4      uvwxy          ---          ---          ---          ---          4,y
        5     uvwxyz          ---          ---          ---          ---          ---

TEST 2
      Step        N'    D(t),p(t)    D(v),p(v)    D(w),p(w)    D(x),p(x)    D(y),p(y)    D(z),p(z)
      ---------------------------------------------------------------------------------
        0          u          2,u          3,u          3,u            ∞            ∞            ∞
        1         tu          ---          3,u          3,u            ∞          9,t            ∞
        2        tuw          ---          3,u          ---          9,w          9,t            ∞
        3       tuvw          ---          ---          ---          6,v          9,t            ∞
        4      tuvwx          ---          ---          ---          ---          9,t         14,x
        5     tuvwxy          ---          ---          ---          ---          ---         14,x
        6    tuvwxyz          ---          ---          ---          ---          ---          ---

TEST 3
      Step        N'    D(v),p(v)    D(w),p(w)    D(x),p(x)    D(y),p(y)    D(z),p(z)
      ---------------------------------------------------------------------------------
        0          u          2,u          5,u          1,u            ∞            ∞
        1         ux          2,u          4,x          ---          2,x            ∞
        2        uxy          2,u          3,y          ---          ---          4,y
        3       uvxy          ---          3,y          ---          ---          4,y
        4      uvwxy          ---          ---          ---          ---          4,y
        5     uvwxyz          ---          ---          ---          ---          ---
```

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.