/**

 * Project Report Template

 * Project 3 (Map Routing), ECE368

 **/

Name: Matthew Bolda

Login: MBolda

/**

 * Explain your overall approach to the problem and a short

 * general summary of your solution and code.

 **/

For my approach to the problem I wanted to start very small and adjust my code to work with larger and larger problems. Because of this, I worked almost exclusively with the toy map and created the map and query as seen in pdf of the project. First, I created a helper function to get a number from the file. The first two numbers were used as the number of vertices and the number of edges. I then created a linked list of the vertices and used the number of the edges to fill in an array of connected vertices to any given vertices.

From there I switched over to the query text file. Using the query file, I used Dijkstra's algorithm to find the distance from the source vertex to every other vertex. Once the distances were found I went to the node in the linked list that corresponded to the destination node and found the shortest path by joining backwards from there. Finally, I printed that order backwards and cleared the distances to do it again for the next query.

Now that I got the function to work for small maps I needed to change it to work for larger maps. For that I attempted to use a hash table but then realized that each vertex has a unique name so I created a array of vertex where the index was the vertex's name. This helped me quickly get to the vertex I needed to instead of parsing through the entire list every time. Along with this array I changed the way I was scanning for numbers from the file. Originally, I was doing fgetc until I hit a space or new line and then converted those characters to an integer, so I switched to fscanf which improved speed.

NOTE: I used the math library for the pow function. I used the pow function to find the distance between two vertices.

/**

 * Known bugs / limitations of your program / assumptions made.

 **/

I do not know of any bugs in my program however I did not test maps with more than 100,000 vertices.

/**

 * List whatever help (if any) that you received.

 **/

Even though we learned about queues in class I used geeks for geeks to learn more/brush up and this helped guide my functions that corresponded to using Queues. I only used this source for creating my queue and the simple operations on the queue.

LINK: https://www.geeksforgeeks.org/queue-set-1introduction-and-array-implementation/

A youtube video I used for learning more about Dijkstra's Algorithm helped me create my strategy for this project. The video had no code or anything other than visualizations to help guide me.

LINK: https://www.youtube.com/watch?v=pVfj6mxhdMw

/**

 * Describe any serious problems you encountered.

 **/

My first problem occurred rather quickly, before I had a queue I tried to use recursion and that caused problems of missing vertices. Luckily the very next lecture we had he told us about the common pitfalls we could encounter and that a queue would help for my specific problem so I was not stuck for very long.

For a while I received segmentation faults when running the bigger files but that happened because of buggy code using the original way I received the next number in the file. Once I fixed that most of my problems were small and easy to fix.

```
/*********************************************************************
 *  List any other comments/feedback here (e.g., whether you
 *  enjoyed doing the exercise, it was too easy/tough, etc.).
 *********************************************************************/
```

Something to note about my program is that before it outputs any paths it takes a long time to initialize all the information needed. Once the information is initialized finding the shortest path for any query is very quick.

The compile line I used for this project was:

```
gcc -Werror -Wall adjacent.c -o adjacent -lm
```

so that

```
./adjacent (insert map.txt) (insert query.txt)
```

will start the program.

This is because I used the math.h library and could not get it to work without the -lm flag. All in all I enjoyed this project more than the rest, I think the base case for the project was easy but it was difficult to make the project more efficient.