Matthew Bolda

ECE 368

11/10/19

Project 3: Milestone 1

For this project we will be using Dijkstra's algorithm to calculate the shortest distance from a given graph of points. The first question asked of us is how we will parse the input files. I think the quest way to do that is to split it up into two sections: map and queries.

The map input will again need to be divided into two parts; locations and connections. The locations on the map are the vertices of nodes on the map, essentially their coordinates. The connections are which nodes are linked to each other and this information will be much harder to parse. My goal for parsing the map will be to create a struct that has the following information: x coordinate, y coordinate, distance, visited, list of links, previous link, and vertex index. What I plan on doing is use the first line find how many vertices and how many links. From there I will fill in the struct described above with the info for that vertex. With this information I hope I will be able to recreate a map like the one we were given in the pdf for the project.

The second input file are the paths we need to check. Given I can successfully implement the struct the way I want using the map file this should be much simpler. I plan to find the distance from the source node to the destination node using only linked nodes to cross them. Once I know the path that gets to the destination in the shortest distance I can backtrack through the previous nodes that were used and print them out.

Dijkstra's algorithm is a way to find the shortest path. The way I plan to implement this is to start at the source node and find the distance to any connected nodes. From their I will pick the closest node and repeat the process, this will allow me to know the distance between the first

node and the third node. As long as I keep track of the nodes, I'm using to get from the source node to the node I'm examining and keeping track of distance I know how far every node is from the source. Basically, you can build a table that tracks previous node and distance and be able to find the shortest doing that.

Most of the work is done when you input the first file. Hopefully I can find a way to implement the struct the way I described and that will allow me to recreate the map to traverse. The second input file is the source and final destinations that I will have to use the map to find the best path.