Brian Rieder

ECE368 – P3

Prof. Felix Lin

"Dijkstra's Algorithm – Milestone 1"

Dijkstra's algorithm is a graph search algorithm designed to find the shortest path between two given vertices by computing the distances from a smallest vertex and then maintaining the distances to each neighbor. Each distance "weight" is maintained and only the smaller values are kept; that is, the distance value stored at a vertex in the graph is only updated if the new distance is smaller. When all of the respective smallest node's neighbors have been updated or maintained, that vertex is considered "removed" and marked as visited. In the end, the "destination" vertex will be given a distance value that is equal to the smallest distance.

My approach to the first input file consists of parsing the data and storing the coordinates of each vertex in a Node structure which is a part of a linked list (order in this list is how it is read, nothing more). Then, to store the second half of the first input file that contains the edges of the graph, I will be using an adjacency linked list to store the edges between respective vertices. Therefore, to store the graph vertices themselves, I will be using a Node data type that contains the vertex value (for instance, an input of "0 1000 2400" will have a vertex value of 0) as well as it's X- and Y- coordinates (in the previous example: x = 1000, y = 2400) along with the "distance" value to be used by the algorithm (initially pseudo-infinite/very large). The adjacency linked list following will just be a linked list of Nodes that are told be adjacent to the given Node.

To interpret the second input file, I will read the number of queries in order to execute the input thoroughly. Then, I will read the two numbers denoting the origin and destination to then traverse through the list of created nodes from the initial input file. Each query will be read into an array where the origin and destination will be in alternating positions. Once this has been determined, traversal through the tree and application of Dijkstra's algorithm can begin. In order to implement the parsing and implementation of the second file, there only needs to be an array of the specified size created in order to contain the origin and destination nodes of the query (for instance, two queries of "0 5" and "4 3" will be stored in the array [0,5,4,3]).

After all of the data has been parsed, application of Dijkstra's algorithm can begin. In order to do so, I will find the given origin node and begin traversing all of its edges and updating distance (using the traditional Distance Formula). Using the same process as described above (most likely iteratively), the program will run until it's met the destination node and assigned it a value of minimum distance as returned by the algorithm. The process of storing the path used to reach it will be done *after* the minimum distance has been found by working backwards through the graph; that is, starting from the destination, and finding the previous node with the smallest value, thus generating the shortest path (backwards) and then reversing it (forwards). After this, the result will be output to the terminal and the program will exit successfully.