Milestone 1 ECE368 Guanshi He

To parse the input map files

Since we know the format of the input graph file, it is consisted of number of vertices and edges (on the first line), x&y coordinates for all the vertices and all the edges(pairs of vertices).

I am going to use struct for each node. In each node structure, it contains the x and y coordinates integer variables, and then followed by the array of integer which represent the distance to other vertices that are conjoint with the node itself.

Therefore, I can read the header line which is the number of vertices and edges firstly, read the corresponding lines of coordinates for all the vertices and finally get all the index of conjoint vertices into the array.

```
struct node{
        int x;   //x coordinate
        int y;    //y coordinate
        int distance[];  //distance between the current node and other nodes
        int number;  //number of connected nodes
}
```

To parse the input search queries file

It is consisted of the number of the queries(on the first line) and following by the queries in the form of a starting point and a destination. Thus, we can get the corresponding number of starting points and destination points by read the file.

To find the shortest path by using Dijkstra's algorithm

Suppose we have loaded all the data correctly and successfully, then we should find the shortest path between the starting points and the destinations.

1. Set the distances to all the other vertices to infinity (once we calculate a shorter path we will update it, otherwise it means the distance vertices have not been visited or cannot be reached directly).
2. Next, calculate the distance between the current node and all the unvisited neighbors.
3. Check whether the newly calculated path length is shorter than before, if yes, then update the value store in the node struct.
4. Move on to the node that with shortest path length from the former one, and repeat Step 2.
5. If we visit the destination and get the smallest distance, or the distances between the starting point and the unvisited nodes are infinity which means there is no connection between the starting point and unvisited nodes, we stop and output the shortest distance.