MATH.H is used to calculate the distance between two nodes.

```
/**********************************************************************
 * Project Report Template
 * Project 3 (Map Routing), ECE368
 **********************************************************************/
```

Name: Jing, Jiangshan
Login: jjing

```
/**********************************************************************
 *  Explain your overall approach to the problem and a short
 *  general summary of your solution and code.
 **********************************************************************/
```

First, the map information is read and stored in an array of pointers. There are two pointers on each node, one pointed to a data structure which contains the X, Y coordinates, the previous point, and the distance from the origin. The other pointed to the next available location from that node. The next available location is stored in a linked list.
Then, the main part of the algorithm starts by initializing the array so that the distances towards the origin are all infinite. Also, another array containing unvisited nodes will be built (For my milestone 1, I wanted to use a priority queue. However, I really do not have time to implement the queue.) If the value at index M is 0, means node M is visited. Otherwise, M is unvisited. Starting at the origin, the algorithm looks for all its unvisited neighbor nodes, and calculate the distance in between. If the distance is smaller than the current distance stored at that neighbor, update the distance, as well as the previous node. After calculated all the neighbor’s distance, move on by removing it from the unvisited set. Until either the destination has been marked visited, or the every reachable node has updated distance (YES, some nodes are unreachable for the USA map), the algorithm is finished.
Now, trace back from the ending point, and get the path.

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

The biggest bug is the fact that the priority queue is missing, which significantly increased the running time… Unfortunately, I have totally 3 projects (368, 362, 301) to work on and 2 finals (369, 362) before dead week. Having 3 projects in two weeks is not fun. ☹
Also, the distance in between two points are fixed. If I can not only store the neighbors, but also store the distance towards that neighbor, would save a lot of running time.

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

I talked 5 minutes to the Professor on this. Otherwise, I really want to say THANK YOU to XCODE. It made my work much easier and enjoyable. ☺

```
/**********************************************************************
 *  Describe any serious problems you encountered.
 **********************************************************************/
```

I do not have any.

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

YES, this project is enjoyable, because it is shorter than project 2, and it has significant practical usage. I believe it is alright to most people. However, with so many things going on these two weeks, I really do not have time to improve my solution. I wish I can receive this project one month before the due day, so that I can have more time to work on.