Map Routing

Programming Description:

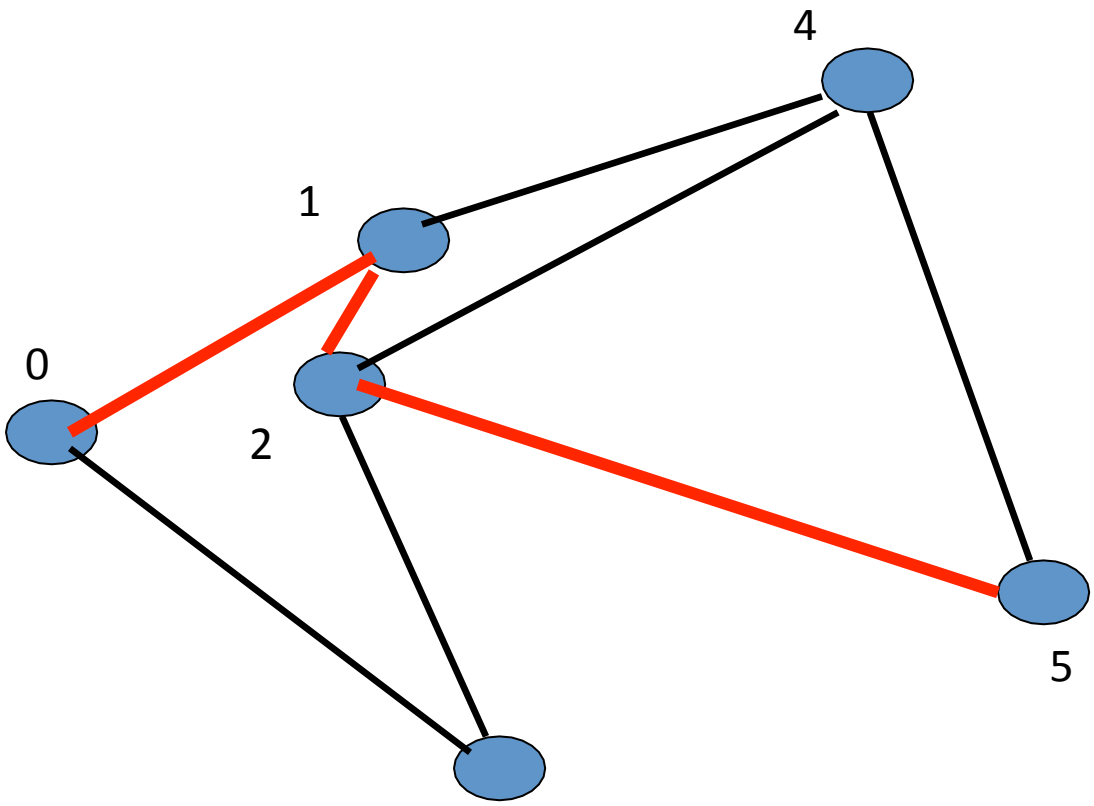
In this, we will implement Dijkstra's shortest path algorithm. This algorithm is widely used in geographic information systems (GIS) including MapQuest and GPS-based car navigation systems.

Maps. Maps are graphs whose vertices as cities and edges as the roads connecting them. The edge weights are the distances between cities. To represent a map in a file, we list the number of vertices and edges, then list the vertices (index followed by its x and y coordinates), then list the edges (pairs of vertices). For example, below is an example of a map with 6 cites.

```
6 9

0  1000 2400
1  2800 3000
2  2400 2500
3  4000    0
4  4500 3800
5  6000 1500

0 1
0 3
1 2
1 4
2 4
2 3
2 5
3 5
4 5
```



A map query asks: given a starting city and a destination city, what is the shortest path to take (i.e., the sequence of intermediate cities to travel) to take? And what is the shortest distance of this path?

For example: above map shows that to go from city 0 to city 5, the shortest path to follow is 0-1-2-5, with shortest distance between city 0 and 5 is 6274.

Your task:

- Implement the basic Dijkstra's shortest path algorithm
- However, the priority queue data structure, which if implemented together with the Dijkastra's algorithm, can dramatically improve the running time of the algorithm.
-

Suggested Steps:

1. Understand well how to initialize the graph data structure (adjacent list) and read into a USA map (provided in the blackboard). You can test if it is done correctly by printing out the graph.
2. Implement the basic Dijkstra's algorithm and test it.
3. Augment it using priority queue and test it.

Format required for testing:

When executed, your program should prompt to ask the source city and destination city; when I input them, it will tell me the sequence of cities to travel (including the source city and destination city) and the distance of this travel.