Christian Toro
UFID: 4169-4846
Email: ctoro97@ufl.edu

CNT4007C – Computer Network Fundamentals
Project Assignment 3

How to compile and run code?

- To compile code simply use the following commands:
  - javac linkstate.java
- To run code, follow these commands (replace [] with correct arguments)
  - java linkstate [filename.txt]

Description of code structure

This code follows a simple structure. Like my previous assignment, I first begin with taking in the command line arguments and reading and formatting the information from the file. Errors are printing on failure of this. Following that, I create an ArrayList of Node objects which I created a class for inside the linkedstate class as I did previously in project assignment 2. I do this by calling a separate method that loops through the file content which is in a string array form. Each index represents a string value (a number). The function initializes each node to have its own ID, a list of neighbors and their distances and a list of every node, including itself, with a parent and distance initialized to '?' and 'i' as requested (with the exception of the node itself). Following this, I then use a print function for printing out the header and follow with another function for running Dijkstra's Algorithm.

For Dijkstra's Algorithm, I create two ArrayLists of type Node. One is for the visited Nodes and the other is for the nodes to visit. I then run a while loop that ends when the nodes to visit list is emptied. Nodes to visit is initialized with just the node we start at. I then begin the loop, using a temporary Node variable called current which gets the Node at the top of the queue. I then check if that node has been visited. If yes, I simply pop it off and move to the next. If not, then I go ahead and add its neighbors and update any distances I need to the neighbors from the origin Node. Each time I add neighbors, I also print out the current step after checking and updating distances and parents.

Execution Results

I tested my code both on Linux and my Windows machine. Upon testing, I was able to successfully go through all the data and print out the correct result. Here are some example results:

```
C:\Users\toroc>java linkstate network.txt
Step,D1,P1,D2,P2,D3,P3,D4,P4,D5,P5,D6,P6
0,0,1,2,1,5,1,1,1,i,?,i,?
1,0,1,2,1,5,1,1,1,i,?,i,?
2,0,1,2,1,5,1,1,1,6,3,10,3
3,0,1,2,1,4,4,1,1,2,4,10,3
4,0,1,2,1,3,5,1,1,2,4,4,5
5,0,1,2,1,3,5,1,1,2,4,4,5

C:\Users\toroc>
```

In this case, I used the small sample provided in the project document. The program successfully visits every node once and updates the distances reached at every node. The 'i' and '?' represent infinite distance and unknown parent respectively. At the end, we get the desired result which compares to the final line in the sample output provided.

```
C:\Users\toroc>java linkstate network.txt
Step,D1,P1,D2,P2,D3,P3,D4,P4,D5,P5,D6,P6,D7,P7,D8,P8,D9,P9,D10,P10,D11,P11,D12,P12,D13,P13,D14,P14,D15,P15,D16,P16,D17,P17,D18,P18,D19,P19,D20,P20,D21,P21,D22,P22,D23,P23,D
24,P24,D25,P25,D26,P26
0,0,1,3,1,2,1,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?
1,0,1,3,1,2,1,i,?,8,2,i,?,i,?,8,2,6,2,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?
2,0,1,3,1,2,1,3,3,6,3,7,3,i,?,8,2,6,2,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?
3,0,1,3,1,2,1,3,3,6,3,7,3,i,?,8,2,6,2,i,?,i,?,i,?,i,?,13,5,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?
4,0,1,3,1,2,1,3,3,6,3,7,3,i,?,8,2,6,2,9,8,i,?,i,?,i,?,13,5,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?
5,0,1,3,1,2,1,3,3,6,3,7,3,i,?,8,2,6,2,9,8,8,9,i,?,12,9,13,5,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?
6,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,i,?,12,9,13,5,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?
7,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,i,?,12,9,13,5,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?
8,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,i,?,12,9,13,5,21,14,i,?,20,14,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?
9,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,12,10,12,9,13,5,21,14,i,?,20,14,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?,i,?
10,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,12,10,12,9,13,5,21,14,i,?,20,14,i,?,14,11,12,11,i,?,i,?,i,?,i,?,i,?,i,?
11,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,12,10,12,9,13,5,21,14,i,?,15,13,i,?,14,11,12,11,i,?,i,?,i,?,i,?,i,?,i,?
12,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,12,10,12,9,13,5,10,7,i,?,15,13,i,?,14,11,12,11,i,?,i,?,i,?,i,?,i,?,i,?
13,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,12,10,12,9,13,5,10,7,12,15,15,13,16,15,14,11,12,11,i,?,i,?,i,?,i,?,i,?,i,?
14,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,12,10,12,9,13,5,10,7,12,15,15,13,16,15,14,11,12,11,i,?,i,?,i,?,i,?,i,?,i,?
15,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,12,10,12,9,13,5,10,7,12,15,15,13,16,15,14,11,12,11,i,?,i,?,i,?,i,?,i,?,i,?
16,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,12,10,12,9,13,5,10,7,12,15,15,13,16,15,14,11,12,11,i,?,22,19,16,19,i,?,i,?,i,?
17,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,12,10,12,9,13,5,10,7,12,15,15,13,16,15,14,11,12,11,16,20,19,20,16,19,i,?,i,?,i,?
18,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,12,10,12,9,13,5,10,7,12,15,15,13,15,16,14,11,12,11,16,20,19,20,16,19,i,?,i,?,i,?
19,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,12,10,12,9,13,5,10,7,12,15,15,13,15,16,14,11,12,11,16,20,19,20,16,19,i,?,i,?,18,18
20,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,12,10,12,9,13,5,10,7,12,15,15,13,15,16,14,11,12,11,16,20,19,20,16,19,28,22,i,?,18,18
21,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,12,10,12,9,13,5,10,7,12,15,15,13,15,16,14,11,12,11,16,20,19,20,16,19,20,23,21,23,18,18
22,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,12,10,12,9,13,5,10,7,12,15,15,13,15,16,14,11,12,11,16,20,19,20,16,19,20,23,21,23,18,18
23,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,12,10,12,9,13,5,10,7,12,15,15,13,15,16,14,11,12,11,16,20,19,20,16,19,20,23,21,23,18,18
24,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,12,10,12,9,13,5,10,7,12,15,15,13,15,16,14,11,12,11,16,20,19,20,16,19,20,23,21,23,18,18
25,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,12,10,12,9,13,5,10,7,12,15,15,13,15,16,14,11,12,11,16,20,19,20,16,19,20,23,21,23,18,18
```

This is the sample from the big output which in turn resulted in the same final line as the sample output given to us. This showing that my code works for both big and small output.

```
C:\Users\toroc>java linkstate network.txt
java.lang.Exception: Network format incorrect
        at linkstate.main(linkstate.java:235)
```

Another example output for when there is no data written in the file. I considered this an extreme and was simple to handle with a quick check in an if statement. The same output is presented when there are sets of data that do not come in three, or non-numerical data.

Bugs, limitations, and missing items

With the semester drawing to a close, I have been really busy. I did originally misinterpret the project to repeat Dijkstra's Algorithm for each node, thus printing a table for each node. My code was structured that way and due to time constraints, I am unable to refactor my code and make it easier to follow and more relevant. I also was not able to fully test everything, so there may be limitations and bugs that I am unaware of.

Additional Reference:
6th edition of the class textbook for Dijkstra's Algorithm.
My previous code from earlier projects in the semester this class.