CNT4007 – Programming Assignment 3
Manu Kolehmainen
UFID: 93281150
Email: manu.kolehmainen@ufl.edu

Compilation

The code works on Windows/Linux/Unix, and can be compiled and ran the same way on each device. The program is written in Java, so a JDK (preferably 1.8 or higher) or JRE is required to be installed on the device to compile and run the code. The code can be compiled into jar files and ran using the commands:
`make linkstate`
`java -jar linkstate.jar [fileName]`

Alternatively, the code can be run directly without compiling into a jar by using
`cd src`
to change the directory to the source directory, and running the following command:
`javac linkstate.java`
to compile the component, and then using
`java linkstate [fileName]`
to run the component.

Note that the parameter is required for the component to function. Additionally, the fileName given to the Linkstate must exist and be readable by Linkstate. This is best accomplished by placing the file to be read in the same directory as the jar files (ie. the root directory).

Functions

Only the source files in /src/ are required for the program to function, ie. linkstate.java. The project is structured to allow the source files to be compiled into jar files, but this is not required for them to function. None of the components accept user input, all input must be given as command line arguments when running them. The linkstate terminates itself once it finishes running.

Results

When testing, no bugs were found in the system. The system behaves exactly as specified in the specification document. When a valid input filename is passed as an argument to the system and the file has data formatted correctly (ie. 3 numbers per row indicating the edges of the graph), the program functions correctly. It performs Dijkstra's algorithm on each node until it has a full linkstate routing graph, and then prints it.

CNT4007 – Programming Assignment 3
Manu Kolehmainen
UFID: 93281150
Email: manu.kolehmainen@ufl.edu

Code Structure

The code defines three classes: linkstate, graph, and node. The node class stores the key of that node and a map of all the edges for that node, avoiding duplicates. Once the node is created, any number of unique edges can be added to it, and the cost to any of its neighbors can be retrieved. The graph class stores the list of all nodes and allows the program to find the cost between any two nodes, returning -1 if they are not neighbors.

When the program starts, the input file is read into a new instance of the graph class, and adds each node to the graph, avoiding duplicates. It then runs Dijkstra's algorithm, starting from the first node and visiting every node in the graph, calculating the cost to visit that node and storing all the costs in an array. The program prints the array on every iteration of the algorithm. If a node cannot be reached or the distance to it has not been calculated yet, it is displayed as '-'.

Sample Output

```
C:\Users\manu2\Documents\$School\CNT4007\CalculatorServer\LinkStateRouting>make linkstate
javac -d bin/ -cp src/ src/linkstate.java
jar cfm linkstate.jar info/linkstate_manifest.txt -C bin .

C:\Users\manu2\Documents\$School\CNT4007\CalculatorServer\LinkStateRouting>java -jar linkstate.jar network2.txt
# Read graph:
# 1 1 0
# 1 2 2
# 1 3 5
# 1 4 1
# 2 1 2
# 2 2 0
# 2 3 3
# 2 4 2
# 3 1 5
# 3 2 3
# 3 3 0
# 3 4 3
# 3 5 1
# 3 6 5
# 4 1 1
# 4 2 2
# 4 3 3
# 4 4 0
# 4 5 1
# 5 3 1
# 5 4 1
# 5 5 0
# 5 6 2
# 6 3 5
# 6 5 2
# 6 6 0

Step,D1,P1,D2,P2,D3,P3,D4,P4,D5,P5,D6,P6
0,0,1,2,1,5,1,1,1,-,-,-
1,0,1,2,1,4,4,1,1,2,4,-,-
2,0,1,2,1,3,5,1,1,2,4,4,5
3,0,1,2,1,3,5,1,1,2,4,4,5
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\manu2\Documents\$School\CNT4007\CalculatorServer\LinkStateRouting>
```

CNT4007 – Programming Assignment 3

Manu Kolehmainen
UFID: 93281150
Email: manu.kolehmainen@ufl.edu

```
C:\Users\manu2\Documents\$School\CNT4007\CalculatorServer\LinkStateRouting>java -jar linkstate.jar network.txt
# Read graph:
# 1 1 0
# 1 2 3
# 1 3 2
# 2 1 3
# 2 2 0
# 2 5 5
# 2 8 5
# 2 9 3
# 3 1 2
# 3 3 0
# 3 4 1
# 3 5 4
# 3 6 5
# 4 3 1
# 4 4 0
# 4 7 2
# 5 2 5
# 5 3 4
# 5 5 0
# 5 9 4
# 5 14 7
# 6 3 5
# 6 6 0
# 6 7 2
# 6 14 6
# 7 4 2
# 7 6 2
# 7 7 0
# 7 15 5
# 8 2 5
# 8 8 0
# 8 9 7
# 8 10 1
# 9 2 3
# 9 5 4
# 9 8 7
# 9 9 0
# 9 11 2
# 9 13 6
# 10 8 1
# 10 10 0
# 10 11 9
# 10 12 3
# 11 19 6
# 11 20 4
# 11 9 2
# 11 10 9
# 11 11 0
# 12 20 5
# 12 10 3
# 12 12 0
# 13 17 3
# 13 19 5
# 13 9 6
# 13 13 0
# 13 14 4
# 14 17 7
# 14 5 7
# 14 6 6
# 14 13 4
# 14 14 0
# 14 15 8
# 15 16 2
# 15 18 6
# 15 7 5
# 15 14 8
# 15 15 0
# 16 16 0
# 16 18 3
# 16 15 2
# 17 17 0
# 17 18 8
# 17 13 3
# 17 14 7
# 18 16 3
# 18 17 8
# 18 18 0
# 18 23 5
# 18 26 3
# 18 15 6
```

CNT4007 – Programming Assignment 3
Manu Kolehmainen
UFID: 93281150
Email: manu.kolehmainen@ufl.edu

```
# 18 15 6
# 19 19 0
# 19 22 8
# 19 23 2
# 19 11 6
# 19 13 5
# 20 20 0
# 20 21 4
# 20 22 7
# 20 11 4
# 20 12 5
# 21 20 4
# 21 21 0
# 21 24 9
# 22 19 8
# 22 20 7
# 22 22 0
# 22 24 9
# 23 18 5
# 23 19 2
# 23 23 0
# 23 24 4
# 23 25 5
# 24 21 9
# 24 22 9
# 24 23 4
# 24 24 0
# 24 25 2
# 25 23 5
# 25 24 2
# 25 25 0
# 25 26 9
# 26 18 3
# 26 25 9
# 26 26 0
```

```
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,D24,P24,D
25,P25,D26,P26
0,0,1,3,1,2,1,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-
1,0,1,3,1,2,1,3,3,6,3,7,3,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-
2,0,1,3,1,2,1,3,3,6,3,7,3,5,4,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-
3,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-
4,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,-,-,-,-,-,-,-,-,-,-,10,7,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-
5,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,-,-,8,9,-,-,12,9,-,-,10,7,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-
6,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,-,-,8,9,-,-,12,9,13,5,10,7,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-
7,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,-,-,8,9,-,-,12,9,13,5,10,7,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,-
8,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,17,11,8,9,-,-,12,9,13,5,10,7,-,-,-,-,-,-,14,11,12,11,-,-,-,-,-,-,-,-,-,-,-,-
9,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,-,-,12,9,13,5,10,7,-,-,-,-,-,-,14,11,12,11,-,-,-,-,-,-,-,-,-,-,-,-
10,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,12,10,12,9,13,5,10,7,-,-,-,-,-,-,14,11,12,11,-,-,-,-,-,-,-,-,-,-,-,-
11,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,12,10,12,9,13,5,10,7,12,15,-,-,16,15,14,11,12,11,-,-,-,-,-,-,-,-,-,-,-,-
12,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,12,10,12,9,13,5,10,7,12,15,-,-,16,15,14,11,12,11,16,20,19,20,-,-,-,-,-,-,-,-
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,-,-,15,16,14,11,12,11,16,20,19,20,-,-,-,-,-,-,-,-
14,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,12,10,12,9,13,5,10,7,12,15,15,13,15,16,14,11,12,11,16,20,19,20,-,-,-,-,-,-,-,-
15,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,12,10,12,9,13,5,10,7,12,15,15,13,15,16,14,11,12,11,16,20,19,20,-,-,-,-,-,-,-,-
16,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,12,10,12,9,13,5,10,7,12,15,15,13,15,16,14,11,12,11,16,20,19,20,-,-,-,-,-,-,-,-
17,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,12,10,12,9,13,5,10,7,12,15,15,13,15,16,14,11,12,11,16,20,19,20,16,19,-,-,-,-,-,-
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,-,-,-,-,18,18
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,-,-,-,-,18,18
20,0,1,3,1,2,1,3,3,6,3,7,3,5,4,8,2,6,2,9,8,8,9,12,10,12,9,13,5,10,7,12,15,15,13,15,16,14,11,12,11,16,20,19,20,16,19,20,23,21,23,18,18
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
```