Java Developer Homework

Graph Problem Response

## Problem Description:

Write a small graph library to create (add edges and vertices), manipulate (add/remove edges and vertices) and query (retrieve edges for vertex, determine connectivity) a graph data structure.

## Solution:

I’ve implemented this graph problem as a network routing problem where I’ve edges are called links and vertices are called nodes. The problem graph is a Network Map and the solution through it is model by the Route object. The best route is created using Dijkstra’s Algorithm and the problem graph is read from a file using JAXB XML Binding. The solution supports both symmetric and asymmetric network links (directional edges).

Download Eclipse Project ZIP file: <https://github.com/prattjam/CodeSamples.git>

## Dependencies:

* Java JDK 1.8
* Log4j 1.2.17
* Junit 4.8.2
* Xerces 2.11.0

## Classes:

* Link.java – class modeling the Edge of the Problem Graph
* NetworkMap.java – class models the Problem Graph (contains: nodeMap(edges/links) and nodeSet(map vertices))
* Route.java – class that contains the solution/list of links that provide a path through the Graph
* BestRoute.java – contain logic that determines the best route through the network map
* RouteTest.java – Junit Test class for running Unit Tests

## Unit Testing

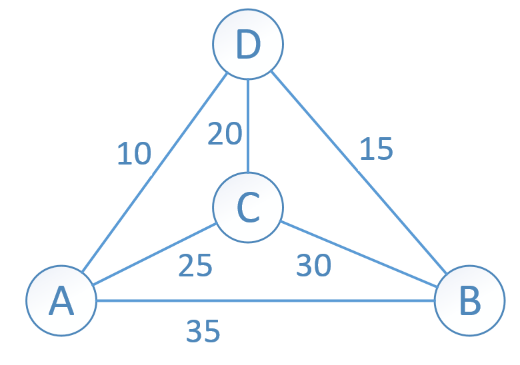
To run Junit Test Cases execute “mvn test” from Command Prompt at the project root directory.

Or run RouteTest.java as a Junit Test

### Test 1:

A simple network map (symmetric non-directional edges): network-map1.xml

Find best route from Node: A to Node: B



Results Test 1:

Best route found was: A to D to B

Output:

The World Map1:

Link [node1=A to node2=D, cost=10, id=link1]

Link [node1=A to node2=C, cost=25, id=link2]

Link [node1=A to node2=B, cost=35, id=link3]

Link [node1=C to node2=D, cost=20, id=link4]

Link [node1=D to node2=B, cost=15, id=link5]

Link [node1=C to node2=B, cost=30, id=link6]

Map Node distances to source: A:

Distance[A] = 0

Distance[B] = 25

Distance[C] = 25

Distance[D] = 10

The Final Route:

Route [Source=A to Destination=B, Total Cost=25, Number of Links=2]

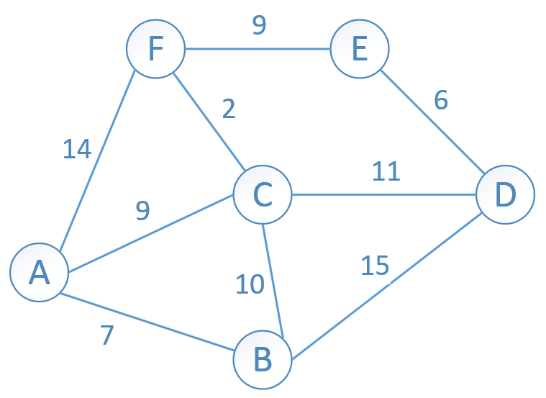
Link [node1=A to node2=D, cost=10, id=link1]

Link [node1=D to node2=B, cost=15, id=link5]

### Test 2:

A larger network map (symmetric non-directional edges): network-map2.xml

Find best route from Node: A to Node: E



Results Test 2:

Best route found was: A to C to F to E

Output:

The World Map2:

Link [node1=A to node2=F, cost=14, id=link1]

Link [node1=A to node2=C, cost=9, id=link2]

Link [node1=A to node2=B, cost=7, id=link3]

Link [node1=C to node2=F, cost=2, id=link4]

Link [node1=D to node2=B, cost=15, id=link5]

Link [node1=C to node2=B, cost=10, id=link6]

Link [node1=C to node2=D, cost=11, id=link7]

Link [node1=D to node2=E, cost=6, id=link8]

Link [node1=F to node2=E, cost=9, id=link9]

Map Node distances to source: A:

Distance[A] = 0

Distance[B] = 7

Distance[C] = 9

Distance[D] = 20

Distance[E] = 20

Distance[F] = 11

The Final Route:

Route [Source=A to Destination=E, Total Cost=20, Number of Links=3]

Link [node1=A to node2=C, cost=9, id=link2]

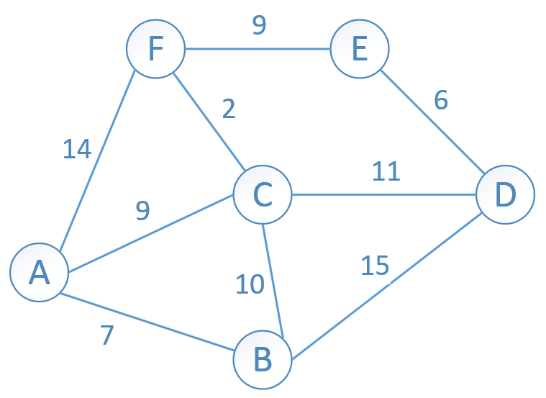
Link [node1=C to node2=F, cost=2, id=link4]

Link [node1=F to node2=E, cost=9, id=link9]

### Test 3:

Modify the Network Map2 from the previous example: network-map2.xml

Find best route from Node: A to Node: E after a map link have added and another has been removed



Results Test 3:

New best route found was: A to C to E with a total cost of 18

Output:

Modify the Network Map2:

Add Link(C, E, 10)

Remove Link(E, D, 6)

Map Node distances to source: A:

Distance[A] = 0

Distance[B] = 7

Distance[C] = 9

Distance[D] = 20

Distance[E] = 19

Distance[F] = 11

The New Solution Route:

Route [Source=A to Destination=E, Total Cost=19, Number of Links=2]

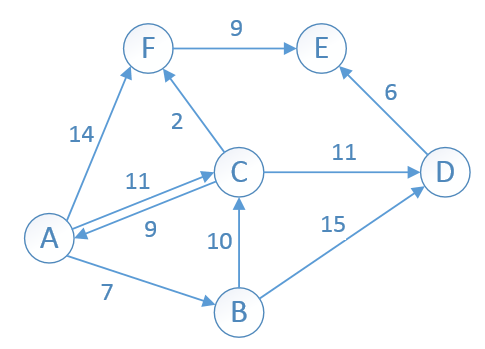
Link [node1=A to node2=C, cost=9, id=link2]

Link [node1=C to node2=E, cost=10, id=link10]

### Test 4:

A directional network map (asymmetric/directional edges): network-map3.xml

Find best route from Node: A to Node: E



Results Test 4:

Best route found was: A to C to F to E

Output:

The World Map3:

Link [node1=A to node2=F, cost=14, id=link1]

Link [node1=C to node2=A, cost=9, id=link2]

Link [node1=A to node2=B, cost=7, id=link3]

Link [node1=C to node2=F, cost=2, id=link4]

Link [node1=B to node2=D, cost=15, id=link5]

Link [node1=B to node2=C, cost=10, id=link6]

Link [node1=C to node2=D, cost=11, id=link7]

Link [node1=D to node2=E, cost=6, id=link8]

Link [node1=F to node2=E, cost=9, id=link9]

Link [node1=A to node2=C, cost=11, id=link10]

Map Node distances to source: A:

Distance[A] = 0

Distance[B] = 7

Distance[C] = 11

Distance[D] = 22

Distance[E] = 22

Distance[F] = 13

The Final Route:

Route [Source=A to Destination=E, Total Cost=22, Number of Links=3]

Link [node1=A to node2=C, cost=11, id=link10]

Link [node1=C to node2=F, cost=2, id=link4]

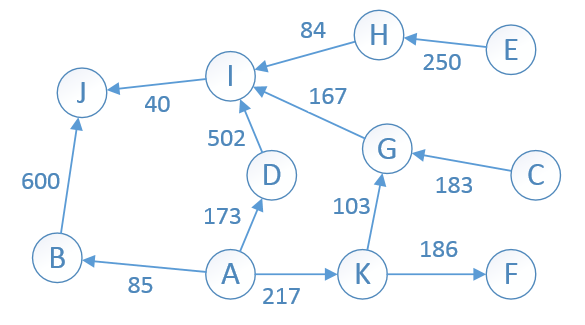
Link [node1=F to node2=E, cost=9, id=link9]

### Test 5:

A larger directional network map with inaccessible and dead end nodes

(asymmetric/directional edges): network-map4.xml

Find best route from Node: A to Node: I



Results Test 5:

Best route found was: A to K to G to I

Output:

The World Map4:

Link [node1=A to node2=K, cost=217, id=link1]

Link [node1=A to node2=D, cost=173, id=link2]

Link [node1=A to node2=B, cost=85, id=link3]

Link [node1=B to node2=J, cost=600, id=link4]

Link [node1=D to node2=I, cost=502, id=link5]

Link [node1=K to node2=G, cost=103, id=link6]

Link [node1=K to node2=F, cost=186, id=link7]

Link [node1=C to node2=G, cost=183, id=link8]

Link [node1=G to node2=I, cost=167, id=link9]

Link [node1=I to node2=J, cost=40, id=link10]

Link [node1=H to node2=I, cost=84, id=link11]

Link [node1=E to node2=H, cost=250, id=link12]

Map Node distances to source: A:

Distance[A] = 0

Distance[B] = 85

Distance[C] = 2147483647

Distance[D] = 173

Distance[E] = 2147483647

Distance[F] = 403

Distance[G] = 320

Distance[H] = 2147483647

Distance[I] = 487

Distance[J] = 527

Distance[K] = 217

The Final Route:

Route [Source=A to Destination=I, Total Cost=487, Number of Links=3]

Link [node1=A to node2=K, cost=217, id=link1]

Link [node1=K to node2=G, cost=103, id=link6]

Link [node1=G to node2=I, cost=167, id=link9]

The Final Route:

Route [Source=A to Destination=I, Total Cost=487, Number of Links=3]

Link [node1=A to node2=K, cost=217, id=link1]

Link [node1=K to node2=G, cost=103, id=link6]

Link [node1=G to node2=I, cost=167, id=link9]