Readme

Usage (for the purpose of this readme):
$java lsnode 1111 2222 3
$java lsnode 2222 1111 3 3333 1
$java lsnode 3333 2222 1 last

tracert simulation:
commands:

- table - Displays the routing table
- tracert <port_number> - trace a route to the designated port number
- exit - terminates the program

Link verification:
Each node will send to its neighbors a verification message with its associated cost.

Sample verification message:
From 1111 to 2222 : ".verify 3"

Means that according to 1111, 2222 is 3 units away.

Each node will also ACK to the sender a verification message with its associated cost.

Sample:
From 2222 to 1111: ".ack .verify 3"

Should the cost differ, then node 2222 will shutdown first because it is the recipient of the original verification message, and node 1111 will shutdown second because it did not receive an ACK from node 2222 within 500 msecs.

The verification process will spread throughout each node in the network. After each node waits 1sec after the verification process, the network will start message flooding.

Message flooding:
Each node will send to its neighbors a "flood" message with information of all its neighbors and costs. (The input arguments), but the last node will start first.

Sample flood message:
From 3333 to 2222 ".flood 3333 2222 1"

Node 2222 already knows 3333 from before, however the message is broadcasted to all of 2222's neighbors.
Node 1111 will now know that 3333 exists in the network and can connect to 2222 in 3 units.

The process repeats until each node no longer has any ACKs to receive from its neighbor nodes. Then it will wait 5 seconds.
By the end of this process, each node will know the entire network topology as well as the costs of each link. Then all possible paths are determined recursively. And the routing table is both printed to string and to screen.

Routing Table Construction:

Each node will then construct its own routing table. After routing table is complete, then the node will broadcast to other nodes regarding its own routing table status. This process will propagate through the entire network. Once every node in the network constructed the routing table, then tracert simulation will begin.

Data Structures:

HashMaps were used to keep track of all the nodes, paths, and their associated costs due to their flexibility and ease of implementation.

Each node, after flooding, will create its internal database of all possible paths and destinations:

```
// Paths grouped into hops (Node 6666)
Hops: 0
          [6666, 5555]: 2
          [6666, 4444]: 5
Hops: 1
          [6666, 5555, 3333]: 3
          [6666, 4444, 5555]: 6
          [6666, 4444, 2222]: 8
          [6666, 5555, 4444]: 3
          [6666, 4444, 1111]: 10
          [6666, 4444, 3333]: 8
Hops: 2
          [6666, 4444, 3333, 2222]: 10
          [6666, 5555, 3333, 2222]: 5
          [6666, 4444, 3333, 5555]: 9
          [6666, 5555, 4444, 3333]: 6
          [6666, 4444, 2222, 3333]: 10
          [6666, 4444, 1111, 2222]: 12
          [6666, 4444, 1111, 3333]: 11
          [6666, 4444, 5555, 3333]: 7
          [6666, 5555, 4444, 2222]: 6
          [6666, 5555, 4444, 1111]: 8
          [6666, 4444, 3333, 1111]: 9
          [6666, 5555, 3333, 1111]: 4
          [6666, 5555, 3333, 4444]: 6
          [6666, 4444, 2222, 1111]: 10
Hops: 3
          [6666, 4444, 5555, 3333, 1111]: 8
          [6666, 5555, 3333, 4444, 1111]: 11
          [6666, 4444, 3333, 2222, 1111]: 12
          [6666, 4444, 1111, 3333, 5555]: 12
          [6666, 5555, 4444, 1111, 3333]: 9
          [6666, 4444, 1111, 3333, 2222]: 13
          [6666, 5555, 4444, 2222, 3333]: 8
          [6666, 5555, 4444, 3333, 1111]: 7
          [6666, 5555, 3333, 4444, 2222]: 9
          [6666, 5555, 3333, 2222, 4444]: 8
          [6666, 5555, 4444, 2222, 1111]: 8
          [6666, 5555, 4444, 3333, 2222]: 8
          [6666, 5555, 3333, 1111, 2222]: 6
          [6666, 5555, 3333, 2222, 1111]: 7
          [6666, 4444, 2222, 3333, 1111]: 11
          [6666, 4444, 5555, 3333, 2222]: 9
          [6666, 4444, 2222, 3333, 5555]: 11
          [6666, 4444, 3333, 1111, 2222]: 11
          [6666, 4444, 1111, 2222, 3333]: 14
          [6666, 5555, 4444, 1111, 2222]: 10
          [6666, 5555, 3333, 1111, 4444]: 9
          [6666, 4444, 2222, 1111, 3333]: 11
Hops: 4
          [6666, 4444, 5555, 3333, 2222, 1111]: 11
          [6666, 5555, 3333, 4444, 2222, 1111]: 11
          [6666, 5555, 4444, 1111, 3333, 2222]: 11
          [6666, 4444, 2222, 1111, 3333, 5555]: 12
          [6666, 5555, 4444, 3333, 2222, 1111]: 10
          [6666, 5555, 3333, 1111, 4444, 2222]: 12
          [6666, 4444, 1111, 2222, 3333, 5555]: 15
          [6666, 5555, 3333, 1111, 2222, 4444]: 9
          [6666, 5555, 4444, 1111, 2222, 3333]: 12
          [6666, 5555, 4444, 3333, 1111, 2222]: 9
          [6666, 4444, 5555, 3333, 1111, 2222]: 10
          [6666, 5555, 4444, 2222, 3333, 1111]: 9
          [6666, 5555, 3333, 4444, 1111, 2222]: 13
          [6666, 5555, 4444, 2222, 1111, 3333]: 9
          [6666, 5555, 3333, 2222, 1111, 4444]: 12
          [6666, 5555, 3333, 2222, 4444, 1111]: 13
```

The routing table has the following structure:

```
1111, [5555, 2]
2222, [5555, 5]
3333, [5555, 3]
4444, [5555, 3]
5555, [5555, 2]
```

First value is the destination port. The second value is an Array-List that contains the next path, and the cost of the entire path.
If node 6666 wants to send to any other nodes, according to this table, will have to send it through node 5555.
Each node’s routing table will be different.

Tracert simulation:

The node will start the tracert simulation. Tracert was chose to demonstrate the functionalities of the program due to similar functions.

Using the routing table, tracert will then forward a trace message to the next node. This process repeats until the destination is reached.

During each hop, the path and the cost is displayed.

Error Handling:

The program terminates if:

- Link costs do not match
- Nodes cannot be reached
- Incorrect user inputs