Project 2

Task:

Implement the distance vector routing protocol.

DVRouter.java can run at all hosts in the specified network.

At each host, the input to the program is the set of directly attached links and their costs.

The routing program at each host reports the cost

and the next hop for the shortest paths to all other hosts in the network.

Note: the program at each host doesn't know the complete network topology.

==========

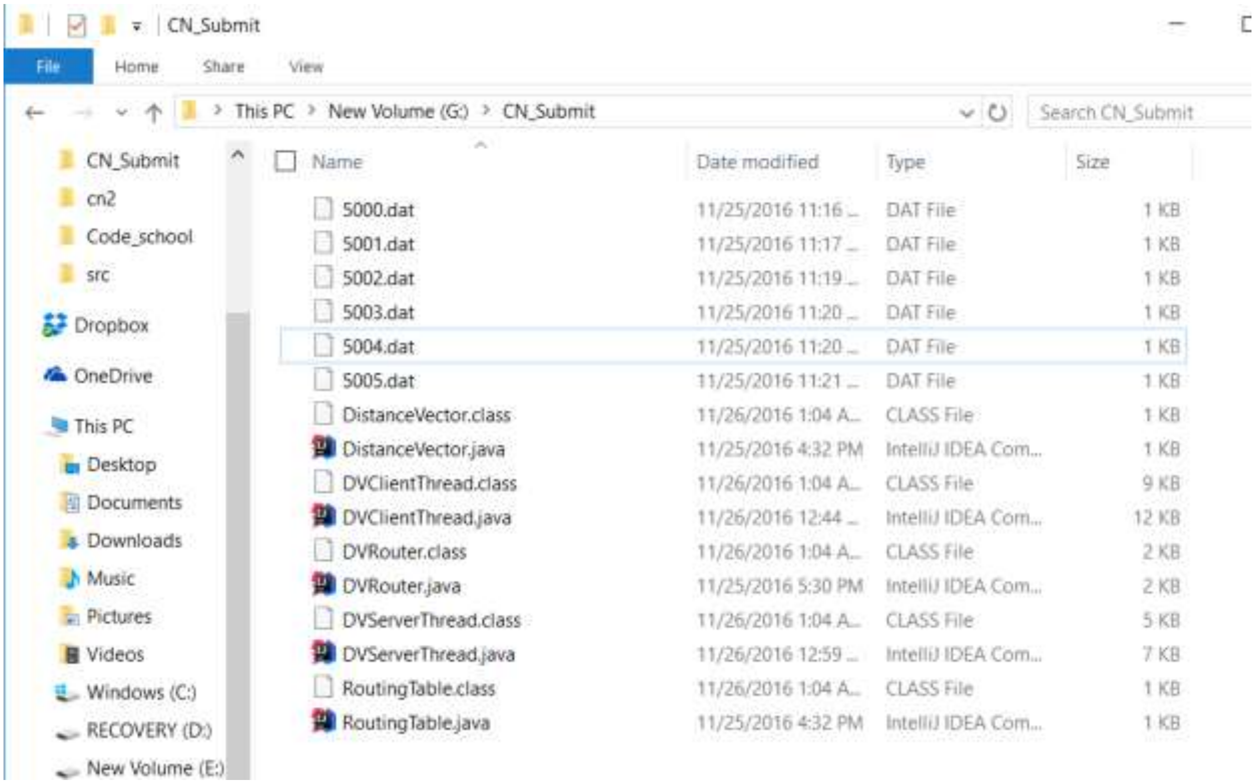
To compile:

Place all the files in a directory or folder (eg. CN)

open cmd root to the folder here it's CN

compile all the java files ---> javac *.java

make sure you have the input files in the same folder or directory



Name	Date modified	Type	Size
5000.dat	11/25/2016 11:16 ...	DAT File	1 KB
5001.dat	11/25/2016 11:17 ...	DAT File	1 KB
5002.dat	11/25/2016 11:19 ...	DAT File	1 KB
5003.dat	11/25/2016 11:20 ...	DAT File	1 KB
5004.dat	11/25/2016 11:20 ...	DAT File	1 KB
5005.dat	11/25/2016 11:21 ...	DAT File	1 KB
DistanceVector.class	11/26/2016 1:04 A...	CLASS File	1 KB
DistanceVector.java	11/25/2016 4:32 PM	IntelliJ IDEA Com...	1 KB
DVClientThread.class	11/26/2016 1:04 A...	CLASS File	9 KB
DVClientThread.java	11/26/2016 12:44 ...	IntelliJ IDEA Com...	12 KB
DVRouter.class	11/26/2016 1:04 A...	CLASS File	2 KB
DVRouter.java	11/25/2016 5:30 PM	IntelliJ IDEA Com...	2 KB
DVServerThread.class	11/26/2016 1:04 A...	CLASS File	5 KB
DVServerThread.java	11/26/2016 12:59 ...	IntelliJ IDEA Com...	7 KB
RoutingTable.class	11/26/2016 1:04 A...	CLASS File	1 KB
RoutingTable.java	11/25/2016 4:32 PM	IntelliJ IDEA Com...	1 KB

For example the above pic shows you that all the input files as well as the *.java files in the same folder

To run:

```
java -cp . DVRouter <port_num> <file_name>

java -cp . DVRouter 5000 5000.dat
```

Run the compiled code multiple times with different port numbers

Make sure the port number of the instance and the file name, as well as the content inside the *.dat files matches accordingly. I have used ports from 5000 to 5005 for convenience.

Since it's difficult to run the program on six different machines I have used different ports with same hostname= "localhost".

the program at each port will repeatedly send the routing table information to its neighbors

and a slow-starting neighbor will eventually get the information.

Node A = 5000

Node B = 5001

Node C = 5002

Node D = 5003

Node E = 5004

Node F = 5005

==========

Design and Implementation Strategies

At each port, the program is being both a server and a client by running with two different threads.

In one thread, server is always listening on new updates, and once get something new it will update distance vector (the hashtable)

by updating cost or add more destination hosts to the table.

In the other thread, client sends out routing table. For the first time, client reads the local host information which is "# of neighbors and cost to each neighbors from current port"; for the rest of time, client thread analyzes the distance vector on this host and convert to routing table to send out.

I had used UDP protocol to communicate between the nodes through different port numbers.

Class DistanceVector: information host-node as a client has

Class RoutingTable: format of information host-node as a server receives

A data structure of hashtable is maintained to keep record of each host's distance vectors. There is a vector for each destination on this host node, figure as below shows

how data structure looks like.

```
                       Hashtable
                         ||
                         ||
                         ||
        \/
                |                       |               |
                |                       |                       |
Source  | Neighbor 1 | Neighbor 2 | Neighbor 3
```

```
--------| -----------------------------------

Dest1    | <---------- Vector

--------| -----------------------------------

Dest2    | <---------- Vector

--------|-------------------------------------

Dest2    | <---------- Vector

--------|-------------------------------------

                    |

                    |
```

<Key : Value> in the hashtap is <Source-Destination : Vector<DistanceVector>>, in which "Source-Destination" is type of String.

The hashtable has to change and grow dynamically because either some cost between two hosts can be replaced with a smaller value or more destination host nodes are being added.

1. Find a new destination, add a new entry in the hashtable

2. Find a smaller cost, replace old cost with the new cost

==========

Test

1. copy "a.dat", "b.dat", ... in the same machine

each files represents an port number

a.dat --> 5000.dat

b.dat --> 5001.dat

c.dat --> 5002.dat

d.dat --> 5003.dat

e.dat --> 5004.dat

f.dat --> 5005.dat

4. In the same machine run multiple instances of java -cp . DVRouter <port_num> <file_name> with different port numbers make sure port number and the file name and the data present inside the files matches for convenience.

==========

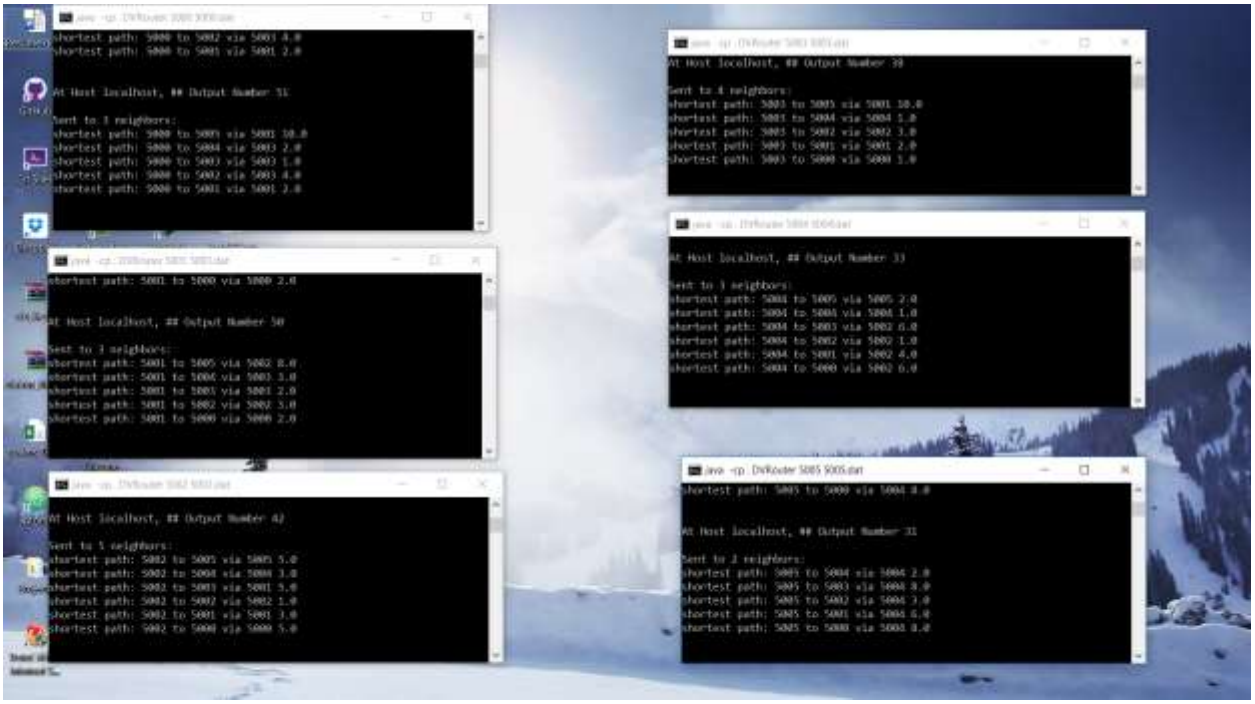
Results

The while loop sequence times to get a stable and final result depends on the time gap to run the program six times with different port numbers.

The bigger time gap to run, the bigger sequence number will be to get the final result.

Results are presented in the form of "shortest path: <startportNo.> to <destPortNo.> via <nextHop PortNo.> <cost>"

Eg.


java -cp . DVRouter 5000 5000.dat
java -cp . DVRouter 5001 5001.dat
java -cp . DVRouter 5002 5002.dat
java -cp . DVRouter 5003 5003.dat
java -cp . DVRouter 5004 5004.dat
java -cp . DVRouter 5005 5005.dat

```
shortest path: 5000 to 5002 via 5003 4.0
shortest path: 5000 to 5001 via 5001 2.0


At Host localhost, ## Output Number 51

Sent to 3 neighbors:
shortest path: 5000 to 5005 via 5001 10.0
shortest path: 5000 to 5004 via 5003 2.0
shortest path: 5000 to 5003 via 5003 1.0
shortest path: 5000 to 5002 via 5003 4.0
shortest path: 5000 to 5001 via 5001 2.0
```

```
shortest path: 5001 to 5000 via 5000 2.0


At Host localhost, ## Output Number 50

Sent to 3 neighbors:
shortest path: 5001 to 5005 via 5002 8.0
shortest path: 5001 to 5004 via 5003 3.0
shortest path: 5001 to 5003 via 5003 2.0
shortest path: 5001 to 5002 via 5002 3.0
shortest path: 5001 to 5000 via 5000 2.0
```

```
At Host localhost, ## Output Number 42

Sent to 5 neighbors:
shortest path: 5002 to 5005 via 5005 5.0
shortest path: 5002 to 5004 via 5004 3.0
shortest path: 5002 to 5003 via 5001 5.0
shortest path: 5002 to 5002 via 5002 1.0
shortest path: 5002 to 5001 via 5001 3.0
shortest path: 5002 to 5000 via 5000 5.0
```

```
At Host localhost, ## Output Number 39

Sent to 4 neighbors:
shortest path: 5003 to 5005 via 5001 10.0
shortest path: 5003 to 5004 via 5004 1.0
shortest path: 5003 to 5002 via 5002 3.0
shortest path: 5003 to 5001 via 5001 2.0
shortest path: 5003 to 5000 via 5000 1.0
```

```
At Host localhost, ## Output Number 33

Sent to 3 neighbors:
shortest path: 5004 to 5005 via 5005 2.0
shortest path: 5004 to 5004 via 5004 1.0
shortest path: 5004 to 5003 via 5002 6.0
shortest path: 5004 to 5002 via 5002 1.0
shortest path: 5004 to 5001 via 5002 4.0
shortest path: 5004 to 5000 via 5002 6.0
```

```
shortest path: 5005 to 5000 via 5004 8.0


At Host localhost, ## Output Number 31

Sent to 2 neighbors:
shortest path: 5005 to 5004 via 5004 2.0
shortest path: 5005 to 5003 via 5004 8.0
shortest path: 5005 to 5002 via 5004 3.0
shortest path: 5005 to 5001 via 5004 6.0
shortest path: 5005 to 5000 via 5004 8.0
```

The each node will re-compute the cost if we change the cost value present in the input file manually, since the program fetches data from the input file for every sequence.

=========

Reference

http://docs.oracle.com/javase/tutorial/networking/datagrams/clientServer.html

http://stackoverflow.com/questions/5433378/run-application-both-as-server-and-client

http://stackoverflow.com/questions/12393231/break-statement-inside-two-while-loops

http://www.java2s.com/Code/Java/JDK-6/ProducerandconsumerbasedonBlockingQueue.htm

=====

Lesson:

Want to array to make use of index,

Think of Vector first

ArrayList

LinkedList

HashMap

Hashtable

By

SIVALINGAM SUBBIAH

800969237