CSCI 5150 **Fall 2014**
Homework 3 **Due: November 3**

In this assignment, you will simulate peers sharing their lists of known neighbors with other peers. Add some screen output to the server component that shows the list of known neighbors, created in previous assignments; number these starting at 1. Then prompt the user to choose one of the neighbors by that number, where 0 means "none". When the choice is >= 1, this peer should contact that neighbor using requestType = 2 (query), where the requestString simply says "ping". The neighbor should respond with requestType = 3 (response), where the requestString says "alive". Then this peer should send another message to the neighbor using a ***new*** requestType = 4 (share), where the request payload is the top three neighbors (or less if there aren't that many) in this semicolon delimited format:

#; ip address; port number; ip address; port number; ip address; port number

The # is the number of pairs of ip address-port numbers in the rest of the message (3 or less). When the neighbor receives these peers, it should add them to its list of neighbors if they are new; display a message displaying any that are not new (they already exist in its list) and identifying them as such, and then display its entire list. This server should also echo what it is sending to the neighbor, and who that neighbor is, for debugging purposes. This server should then re-display the list and the prompt after the list has been sent. Of course, the neighbor, and any other peers, will be displaying the same kind of prompt.

Next, implement a simple query-for-file system. Add an option to the client menu for sending a query to another peer – this can be the same option from Homework 2, if you did that correctly. This time, prompt the client's user for the name of a file, no more than 12 characters (for now). Send a type 2 query to one of the others on the list of peers with the file name in the payload and "lookup" in the requestType. That peer should search for the file in a subdirectory called "content" and return "found" in the requestString if it has a file of that name along with the contents of the file in the payload field of a type 3 message– but remember that (for now) the payload is limited to 256 characters, so use short files. If the other peer does not have the file, return a type 3 message with "not found" in the requestString. Create some simple test files in the "content" subdirectory.

Make sure you test this program on both einstein and cs5150. If any of the above needs clarification, ask away – I'd be surprised if it didn't need some more explanation.

You can work in pairs for this program. Email me your zipped code at barrettm@etsu.edu.