COMPUTER SCIENCES AND SOFTWARE ENGINEERING
AUBURN UNVERSITY

COMP 2710
Software Construction
Section 002

Fall 2017

Lab 2
Distributed TigerBook Social Network

Due: October 27, 2017

Points Possible: 100
Due: Design Portion: October 20th, 2017 by 11:55 pm. Submission via Canvas (25 points)
Program Portion: October 27th, 2017 by 11:55 pm via Canvas (75 points)

No late assignments will be accepted.
No collaboration between students. Students should NOT share any project code with each other. Collaborations in any form will be treated as a serious violation of Auburn University's academic integrity code.

Goals:

- To perform Object-Oriented Analysis, Design, and Testing
- To develop a non-trivial application using classes and constructors to implement abstractions (ADTs)
- To learn distributed communication methods and develop a simple but useful distributed social network system
- To learn to use file I/O and take advantage of the Network File System (NFS)

Process – 25 points:
Create a text, doc, or .pdf file named "<username>_2p" (for example, mine might read "lim_2p.txt") and provide each of the following. Please submit a text file, a .doc file or .pdf file (if you want to use diagrams or pictures, use .doc or .pdf). You are free to use tools like Visio to help you, but the final output needs to be .txt, .doc, or .pdf.

1. **Analysis:** Write all the use cases. Remember, these use cases describe how the user interacts with a social network system (what they do, what the systems does in response, etc.). Your use cases should have enough basic details such that someone unfamiliar with the system can have an understanding of what is happening in the social network system interface. They should not include internal technical details that the user is not (and should not) be aware of.

2. **Design**:
   a) Create a Class Diagram. Be sure to include:
      1) The name and purpose of the classes
      2) The member variables and the functions of the classes
      3) Show the interactions and relationships between classes (for example, ownership, use or dependency)
      4) Any relevant notes that don't fit into the previous categories can be added
   b) Create the data flow diagrams to describe the system. Show all the entities and processes that comprise the overall system and show how information flows from and to each process.

3. **Testing**: Develop lists of *specific* test cases:
   1) For the system at large. In other words, describe inputs for "nominal" usage. You may need several scenarios. In addition, suggest scenarios for abnormal usage and show what your program should do (for example, entering a negative number for a menu might ask the user to try again).
   2) For each object. (Later, these tests can be automated easily using a simple driver function in the object).

Remember, test cases are specific, repeatable, and should have a clearly defined result.

4. Implement the Distributed TigerBook Social Network System in C++.

5. **Test Results**: ***After developing the Distributed TigerBook Social Network System***, actually try all of your test cases (both system and unit testing). Actually show the results of your testing (a copy and paste from your program output is fine – don't stress too much about formatting as long as the results are clear). You should have test results for every test case you described. If your system doesn't behave as intended, you should note this.

Program Portion:

The proliferation of Internet applications in smartphones, laptops, ipads, and desktops shows the importance of communications among computing devices. One of the key functionalities is the ability to *post and receive messages from one computer to another over the network*. It is interesting to learn how a simple social network system can provide this functionality. In this Lab 2, you will analyze, design, implement and test a simple distributed social network system. It will have a simple text-based user interface that allows multiple users to share messages through one of the three methods: (1) **broadcast** a message to all users, (2) **multicast** a message to a group, or (3) **unicast** a message to a particular user. In this lab project, all the users must be implemented in *distributed address space*, i.e. *each user must execute on a different computer and message of one user can be read by another through shared common files*. The *common files are shared through the Network File Server (NFS).* Your program must allow a user *U* to display in their **home page** all multicast messages to each group *G* where *U* is a member of, all broadcast messages, and all unicast messages to *U*. Your program must allow a user *U* to display in their **wall page** all messages posted by *U*.

What is a Distributed Social Network System?

You probably have used other distributed social network systems over the Internet. Although there are many different types of social network systems, you will implement the Distributed TigerBook Social Network System that *allows users to post and read messages from another user executing on a different computer*. For this assignment's purposes, the Distributed TigerBook Social Network System is a program (or collection of programs) that does the following:

1) There are many *distributed and concurrent* users that use the Distributed TigerBook Social Network System for posting their own messages and sharing them with their friends, i.e. all users may use the system simultaneously on different computers to post and read messages. In this assignment you must implement **at least four users**.
2) Each user posts their message using either broadcast, multicast or unicast communication methods. All messages, including broadcast, multicast and unicast messages sent by any user will be stored in files depending on the type of the message. Broadcast messages are stored in the file "*_All";* multicast messages are stored in the file with the group name (e.g. "*_Tigers")*; and unicast messages are stored the file with the recipient user name. The format of the broadcast, multicast and unicast messages will contain the user's name, e.g. if Bill multicast a message "`Greetings, everybody!`" to group "*#Tigers"*, then the message is stored in the file "*_Tigers"* as "`{!173!}<!Bill!>Greetings, everybody!`", where 173 is the timestamp. All users who are members of the #Tigers group can then read the file "*_Tigers*" and display the user name and the multicast messages in reverse chronological order together with other (broadcast, multicast and unicast) messages. *In all cases, the program must allow multiple lines of a message to be posted by each user.*
3) When a user ***broadcast*** a message, it will be seen by all users. You can use the group name "*#All"* to be associated with each broadcast message. All broadcast messages will be stored in a file called "*_All"*. Since the file is maintained by NFS, another user will also see the same file "*_All"*, although it is executing on a different computer. The Distributed TigerBook Social Network System can then read the broadcast messages posted by different users. *The program must allow multiple lines of a message to be broadcast by each user.*
4) ***Multicast*** messages allow users to share messages with only the users who are members of a particular group. When a user multicast a message, the system will prompt for the group name. The system must check if the group name is valid. All multicast messages will be stored in a file associated with the group name, e.g. a multicast message to group "*#Tigers"* is stored in the file called "*_Tigers"*. The format of the multicast messages is as described in (2).
5) The group manager will maintain the different groups used in the system, including creating and deleting a group, adding users into a group and removing users from a group. When adding a user in the group, the system must check that the user to be added is a valid current user.
6) When a user ***unicast*** a message, it will be seen by only the recipient user. The system will first prompt for the recipient user name. All unicast messages will be stored in a file with the recipient name. For instance, when `Joe` sends a unicast message to `Mary`, the message is stored in a file called "*Mary"* where the message is formatted as follows: "`{!214)!}<!Joe!>Good Morning, Sunshine!`, where 214 is the timestamp.

7) Since the users are each running on different computer, you must keep track of the time ordering of the messages. There are at least two methods for keeping track of the time ordering of the messages. The first method is to use the time() function to obtain the time and storing the time with the message. The second method is to maintain a file called "time", shared by all the users. The file "time" contains an integer value which is incremented by one every time a user read the number to time-stamp their message. For example the number in the time file is initially 0. When Jane posts a message, the system will read the time file and time-stamp Jane's message with 0. It then increments the time value and writes back 1 into the time file, over-writing the previous 0 value.
8) Each user can display their home page. The **home page** of a user displays all the messages that are received from broadcast, multicast and unicast messages and all of their own messages, in reverse chronological order in which they are created, i.e. the most recent messages are displayed before the less recent messages. When displaying the messages, display the group name for multicast, but not the recipient name for unicast or "#All" for broadcast; only display the name of the sender and the messages. For multicast messages, the user must be a member of the group. For unicast message, the user must be the recipient.
9) Each user can display their wall page. The **wall page** of a user displays all the messages that are sent by the user, including broadcast, multicast and unicast messages, in reverse chronological order in which they are created. When displaying the messages, **you must** display the group name for multicast, recipient name for unicast or "#All" for broadcast; followed by the messages.
10) In the home or wall pages, the program will first display only the two most recent messages (multicast, unicast, or broadcast) from other users or sent by themselves, and prompt if the user wants to display more messages. If the response is "yes", then all messages from themselves and all other users and will be displayed, otherwise, the program will stop the display.
11) Since this assignments implements Distributed TigerBook Social Network System, it allows many users to be logged in concurrently.

Message Format

For this assignment, the messages must be stored in the files using the following format. There must be exactly *one message file* for each group, one for all broadcasts, and one for each recipient of unicast messages. In each of the user message file, the messages must be stored in reverse chronological order from which they were created. Each message must contain the timestamp and the user message using the following format "{!Timestamp!}<!sender!>message text". Each message must be of variable length. Different lines of a message must be separated by a '`\n`' character, i.e. each line of the message must be terminated by a newline character. For examples, if `John` first multicast the message "`Welcome to the group!\nGreat to hear from you all!`" to the group "#Tigers". And later `John` multicast "`I'll update you all soon`" to the group "#Tigers", then message file called "_Tigers" must contain "`{!253!}<!John!>I'll update you all soon{!216!} <!John!>Welcome to the group!\nGreat to hear from you all!`". The string "`{!253!}`" is the timestamp enclosed in the "`{!`" and "`!}`" strings. Assume that messages will not contain these markers. The timestamp allows multiple messages in multiple files sent by different users to be ordered correctly in reverse chronological order from which they were posted. The purpose for this standard message format is that group's message files should end up being compatible, i.e. every user should be able to work with every message file.

Your program must meet this requirement for message format, otherwise significant points will be deducted.

The user interface

Write a menu-based and text-based user interface for interacting with the Distributed TigerBook Social Network System. When the Distributed TigerBook System is first started, it will first display the banner and *prompts for the user's name*. After the user's name is entered, the system will print a banner welcoming the user.

```
        ============================================================
        |            Distributed TigerBook System!                 |
        ============================================================

Please enter user name: Bill

        ============================================================
        |      Welcome to Distributed TigerBook System, Bill!      |
        ============================================================
```

The program will then make Bill the current user. The main menu has (at least) 8 options to choose from as shown below. *Abbreviate the menu so that all the 8 options can be shown in only one or two lines.* The user interface will accept the option and all related inputs from the user, perform the operation and then repeatedly shows this menu of options again and prompts for the next option.

1) **Broadcast a message (b)**: When selected, the program will prompt for the message and stores the message in the message buffer. *Your program must allow multiple lines of messages. To end a message, the user will enter a new line with a string "^!"followed by the enter key. The final string "^!" must not be stored in the message buffer, neither should it be displayed in the wall or home pages.*
2) **Multicast a message (m)**: When selected, the program will first prompt for *group* name and then it prompts for the message. As in broadcast messages, your program must allow multiple lines of messages. To end a message, the user will enter a new line with a string "^!" followed by the enter key.
3) **Unicast a message (u)**: When selected, the program will first prompt for *user* name and then it prompts for the message. As in broadcast messages, your program must allow multiple lines of messages. To end a message, the user will enter a new line with a string "^!" followed by the enter key.
4) **Display wall page (w)**: When selected, the program will first display a title indicating that it is displaying the current user's wall page, e.g. "`Joe's Wall Page`". It then displays the *two* latest messages in the current user's wall page, in reverse chronological order. The wall page must display the group (for multicast) or recipient (for unicast) or "#All" (for broadcast), and the user's messages and in reverse chronological order. Messages from different posting must be separated by a blank line. After displaying the two latest messages, it will then prompt the user if they want more messages. If the response is "no", then it will stop displaying messages, but if the response is "yes", it will display all the remaining messages from that user. If there are two or fewer messages then the program must not prompt for more messages.
5) **Display home page (h)**: When selected, it first displays the current user home page title, e.g. "`Joe's Home Page`", and then it displays only the two latest messages (either from broadcast, multicast or unicast messages), whichever are the two latest messages. When displaying the message from each user, it must display the sender name, followed by the group name (or #All or recipient name) in parentheses and followed by a string " > " and a '\n' and then followed by the message. The group name or "#All" or the recipient name (for unicast) must be displayed. Messages from different users must be separate by a blank line. After displaying the latest two messages, it will then prompt the user if they want more. If the response is "no", then it will stop displaying messages, but if the response is "yes", it will display all the remaining (multicast, broadcast or unicast) messages for that user. If there are two or fewer messages then the program must not prompt for more messages. In both the cases above, the messages are all displayed in reverse chronological order, i.e. the most recently posted messages will be displayed before the earlier messages.
6) **Create a group (g)**: When selected, the program will then prompt for the name of the group. It then checks if the group's name is already an existing group. If so, it will display an error message and prompts for another group name.
7) **Join a group (j)**: When selected, the program will first prompt for the name of the group. It then checks if the group's name is already an existing group. If not, it will display an error message and prompts for another group name. Next, the program will add the current user name to the group.
8) **Quit the Distributed TigerBook System:** When selected, the program will exit gracefully.

The user interface must check for correct input value from the users. If there is any error, e.g. invalid user name, then the program must display the error and continue to prompt for the correct input. Your program must not crash, exit or terminate when there is an incorrect input value.

The name of your program must be called <username>_2.cpp (for example, mine would read "lim_2.cpp"

Use comments to provide a heading at the top of your code containing your name, Auburn Userid, and filename. You will lose points if you do not use the specific program file name, or do not have a comment block on **EVERY** program you hand in.

Your program's output need not exactly match the style of the sample output (see the end of this file for one example of sample output).

Important Notes:
You must use an object-oriented programming strategy in order to design and implement this Distributed TigerBook Social Network system (in other words, you will need to write class definitions and use those classes, you can't just throw everything in main()). A well-done implementation will produce a number of robust classes, many of which may be useful for future programs in this course and beyond. Some of the classes in your previous lab project may also be re-used here, possibly with some modifications. Remember

good design practices discussed in class:
 a) A class should do one thing, and do it well
 b) Classes should NOT be highly coupled
 c) Classes should be highly cohesive

- You should follow standard commenting guidelines. Follow the C++ programming style guideline that was handed out.

- You DO NOT need any graphical user interface for this simple, text-based application. If you want to implement a visualization of some sort, then that is extra credit.

Error-Checking:

You should provide enough error-checking that a moderately informed user will not crash your program. This should be discovered through your unit-testing. Your prompts should still inform the user of what is expected of them, even if you have error-checking in place.

Please submit your program through the Canvas system online. If for some disastrous reason Canvas goes down, instead e-mail your submission to TA – Yue Cui – at (yzc0058@tigermail.auburn.edu). Canvas going down is not an excuse for turning in your work late.

You should submit the two files in digital format. No hardcopy is required for the final submission:

<username>_2.cpp
<username>_2p.txt (script of sample normal execution and a script of the results of testing)

Sample Usage:

Bold letters indicate user's input.

The following is executed in tux201:

> lim_2

```
        ============================================================
        |             Distributed TigerBook System!                |
        ============================================================

        Please enter user name: George

        ============================================================
        |     Welcome to Distributed TigerBook System, George!     |
        ============================================================

Broadcast (b), Multicast (m), Unicast(u), Wall (w), Home (h),
Create Group (g), Join Group (j), Quit (q)
Enter option: g
Please enter the group name: #Tigers

        ============================================================
        |                New group #Tigers created                 |
        ============================================================

Broadcast (b), Multicast (m), Unicast(u), Wall (w), Home (h),
Create Group (g), Join Group (j), Quit (q)
```

```
Enter option: j
Please enter the group name: #Tigers

        ============================================================
        |                George has joined group #Tigers           |
        ============================================================

Broadcast (b), Multicast (m), Unicast(u), Wall (w), Home (h),
Create Group (g), Join Group (j), Quit (q)
Enter option: b
Enter message: Join our new #Tigers message group!
It was fun at the beach.
How was your summer?
^!

        ============================================================
        |            George has broadcasted a new message          |
        ============================================================

Broadcast (b), Multicast (m), Unicast(u), Wall (w), Home (h),
Create Group (g), Join Group (j), Quit (q)
Enter option: q

        ============================================================
        |  Thank you for using the Distributed TigerBook System    |
        ============================================================
```

The following is executed in tux202:

> lim_2

```
        ============================================================
        |             Distributed TigerBook System!                |
        ============================================================

        Please enter user name: Mary

        ============================================================
        |     Welcome to Distributed TigerBook System, Mary!       |
        ============================================================

Broadcast (b), Multicast (m), Unicast(u), Wall (w), Home (h),
Create Group (g), Join Group (j), Quit (q)
Enter option: j
Please enter the group name: #Tigers

        ============================================================
        |                Mary has joined group #Tigers             |
        ============================================================

Broadcast (b), Multicast (m), Unicast(u), Wall (w), Home (h),
Create Group (g), Join Group (j), Quit (q)
Enter option: m
Enter the group name: #Tigers
Enter message: Welcome back, guys!
We had so much fun at Ireland!
^!
        ============================================================
        |       Mary has multicasted a message to group #Tigers    |
        ============================================================
```

```
Broadcast (b), Multicast (m), Unicast(u), Wall (w), Home (h),
Create Group (g), Join Group (j), Quit (q)
Enter option: q

        ============================================================
        |  Thank you for using the Distributed TigerBook System    |
        ============================================================
```

The following is executed in tux203:

> lim_2

```
        ============================================================
        |             Distributed TigerBook System!                |
        ============================================================

        Please enter user name: Abraham

        ============================================================
        |    Welcome to Distributed TigerBook System, Abraham!     |
        ============================================================

Broadcast (b), Multicast (m), Unicast(u), Wall (w), Home (h),
Create Group (g), Join Group (j), Quit (q)
Enter option: j
Enter the group name: #Tigers

Broadcast (b), Multicast (m), Unicast(u), Wall (w), Home (h),
Create Group (g), Join Group (j), Quit (q)
Enter option: u
Enter the recipient user name: Mary
Enter message: Good Morning, Sunshine!
When did you get back from Ireland?
^!

        ============================================================
        |           Abraham has unicasted a message to Mary        |
        ============================================================

Broadcast (b), Multicast (m), Unicast(u), Wall (w), Home (h),
Create Group (g), Join Group (j), Quit (q)
Enter option: g
Please enter the group name: #Eagles

        ============================================================
        |                New group #Eagles created                 |
        ============================================================

Broadcast (b), Multicast (m), Unicast(u), Wall (w), Home (h),
Create Group (g), Join Group (j), Quit (q)
Enter option: j
Please enter the group name: #Eagles

        ============================================================
        |               Abraham has joined group #Eagles           |
        ============================================================

Broadcast (b), Multicast (m), Unicast(u), Wall (w), Home (h),
Create Group (g), Join Group (j), Quit (q)
```

```
Enter option: b
Enter message: Just created a new #Eagles group!
Come one, come all!
Have a blast!
^!

        ============================================================
        |            Abraham has broadcasted a new message         |
        ============================================================

Broadcast (b), Multicast (m), Unicast(u), Wall (w), Home (h),
Create Group (g), Join Group (j), Quit (q)
Enter option: m
Enter the group name: #Eagles
Enter message: Eagles have landed!
War eagle, hey!
^!

        ============================================================
        |  Abraham has multicasted a new message to group #Eagles  |
        ============================================================

Broadcast (b), Multicast (m), Unicast(u), Wall (w), Home (h),
Create Group (g), Join Group (j), Quit (q)
Enter option: m
Enter the group name: #Tigers
Enter message: I'm enjoying your group too.
Spent my summer in South Africa; it was amazing!
^!

        ============================================================
        |  Abraham has multicasted a new message to group #Tigers  |
        ============================================================

Broadcast (b), Multicast (m), Unicast(u), Wall (w), Home (h),
Create Group (g), Join Group (j), Quit (q)
Enter option: q

        ============================================================
        |  Thank you for using the Distributed TigerBook System    |
        ============================================================
```

The following is executed in tux204:

> lim_2

```
        ============================================================
        |             Distributed TigerBook System!                |
        ============================================================

        Please enter user name: Kate

        ============================================================
        |     Welcome to Distributed TigerBook System, Kate!       |
        ============================================================

Broadcast (b), Multicast (m), Unicast(u), Wall (w), Home (h),
Create Group (g), Join Group (j), Quit (q)
Enter option: j
Enter the group name: #Eagles
```

```
        ============================================================
        |               Kate has joined group #Eagles              |
        ============================================================

Broadcast (b), Multicast (m), Unicast(u), Wall (w), Home (h),
Create Group (g), Join Group (j), Quit (q)
Enter option: m
Enter the group name: #Eagles
Enter message: It's time for Auburn Football!
War Eagle!
^!
        ============================================================
        |  Kate has multicasted a new message to group #Eagles     |
        ============================================================

Broadcast (b), Multicast (m), Unicast(u), Wall (w), Home (h),
Create Group (g), Join Group (j), Quit (q)
Enter option: h


        ================================================
        |             Kate's Home Page                 |
        ================================================

Kate (#Eagles) >
It's time for Auburn Football!
War Eagle!

Abraham (#Eagles) >
Eagles have landed!
War eagle, hey!
                      More message? (yes/no): yes

Abraham (#All) >
Just created a new #Eagles group!
Come one, come all!
Have a blast!

George (#All) >
Join our new #Tigers message group!
It was fun at the beach.
How was your summer?

        ============================================================
        |                     End of Kate's Home Page               |
        ============================================================

Broadcast (b), Multicast (m), Unicast(u), Wall (w), Home (h),
Create Group (g), Join Group (j), Quit (q)
Enter option: q

        ============================================================
        |  Thank you for using the Distributed TigerBook System    |
        ============================================================
```

The following is executed in tux205:

> lim_2

```
        ============================================================
        |             Distributed TigerBook System!                |
        ============================================================

        Please enter user name: Mary

        ============================================================
        |     Welcome to Distributed TigerBook System, Mary!       |
        ============================================================

Broadcast (b), Multicast (m), Unicast(u), Wall (w), Home (h),
Create Group (g), Join Group (j), Quit (q)
Enter option: h

        ============================================
        |             Mary's Home Page             |
        ============================================

Abraham (#Tigers) >
I'm enjoying your group too.
Spent my summer in South Africa; it was amazing!

Abraham (#All) >
Just created a new #Eagles group!
Come one, come all!
Have a blast!

                      More message? (yes/no): yes
Abraham >
Good Morning, Sunshine!
When did you get back from Ireland?

Mary (#Tigers) >
Welcome back, guys!
We had so much fun at Ireland!

George (#All) >
Join our new #Tigers message group!
It was fun at the beach.
How was your summer?

        ============================================================
        |                    End of Mary's Home Page                |
        ============================================================

Broadcast (b), Multicast (m), Unicast(u), Wall (w), Home (h),
Create Group (g), Join Group (j), Quit (q)
Enter option: w

        ============================================
        |             Mary's Wall Page             |
        ============================================
Mary (#Tigers) >
Welcome back, guys!
We had so much fun at Ireland!

        ============================================================
        |                   End of Mary's Wall Page                |
        ============================================================

Broadcast (b), Multicast (m), Unicast(u), Wall (w), Home (h),
```

```
Create Group (g), Join Group (j), Quit (q)
Enter option: q
        ============================================================
        |  Thank you for using the Distributed TigerBook System    |
        ============================================================
```

The following is executed in tux206:

> lim_2

```
        ============================================================
        |             Distributed TigerBook System!                |
        ============================================================

        Please enter user name: Abraham

        ============================================================
        |    Welcome to Distributed TigerBook System, Abraham!     |
        ============================================================

Broadcast (b), Multicast (m), Unicast(u), Wall (w), Home (h),
Create Group (g), Join Group (j), Quit (q)
Enter option: w

        ==========================================
        |            Abraham's Wall Page          |
       ==========================================

Abraham (#Tigers) >
I'm enjoying your group too.
Spent my summer in South Africa; it was amazing!

Abraham (#Eagles) >
Eagles have landed!
War eagle, hey!
                      More message? (yes/no): yes

Abraham (#All) >
Just created a new #Eagles group!
Come one, come all!
Have a blast!

Abraham (Mary) >
Good Morning, Sunshine!
When did you get back from Ireland?
        ============================================================
        |                    End of Abraham's Wall Page            |
        ============================================================

Broadcast (b), Multicast (m), Unicast(u), Wall (w), Home (h),
Create Group (g), Join Group (j), Quit (q)
Enter option: q

        ============================================================
        |  Thank you for using the Distributed TigerBook System    |
        ============================================================
```