100707816
Andrew Murdoch

CSCI 3310 Assignment Two Report
Chat Client and Server

I used pthreads in both my client and server programs. A thread is created on the server side for each new client whose connection is accepted. The client's sock id is captured as well as their username. I use a struct to hold each client's username, client index and socket id. The history of the chat is held in a 2D array of strings. Using strcpy and I am able to send the history to each new client. Since the history must contain the last 12 lines in the chat I will need to push new values to the array after it has been filled with 12 strings. I do this by shifting the values of the array to the left for every new message after the twelfth. To send the history to each client I needed to send each string in a for loop. I looped through each different socket descriptor for each new message the server received, therefore sending the message to each client. After a client disconnects I close the appropriate socket and adjust the client socket array.

The client program receives input from the user through standard input file descriptor and sends the input to the server. The server receives this and echoes the data back whilst sending it to other users as well. The client receives data from the server through a thread created in the client program. This is needed because the main thread in the client program is used to receive input from the user. The client infinitely receives input from the server printing it out in console. If the server crashes or disconnects the client is notified, the socket is closed and the program exits.

100707816
Andrew Murdoch

I had many issues while programming this assignment. Most of them were very hard to debug since they were runtime errors. Figuring out the best way to hold and send an array of strings in C was one of them. Knowing when and where to close a socket was also difficult. My biggest issue was allowing a client to receive multiple messages from the server. I figured that creating a separate thread to handle that was my best course of action.