# COSC402 Lab 3: Multi-user Chatting System using Select

## Description

In this lesson, you will practice to:

- use select() to read data from multiple inputs;
- implement a multi-user chatting system using select().

## Programming I

Copy the *Lab3* folder in the pickup directory to the "402lab" directory in your local machine. In this task you will implement a concurrent chatting server using select(). The following functions need to be implemented based on the skeleton code given within the functions.

### chatLoop()

- processing the communication between the chatters and the server using select();

### processChat()

- send the data received from each chatter to all the other chatters.

### addChater()

- add a new chatter to the chat room

### remove_chatter()

- remove the leaving chatter from the chat room.

### Testing

Change into the *Lab3* directory, use **make** command to compile the source codes. Do the following when the executable files have been generated with no errors.

1. Open one terminal window, change into the *Lab3* directory, and type the following command to run the server program: **./chatserver 127.0.0.1 65530**
2. Open multiple terminal windows, change into the *Lab3* directory, and type the following command in each window to run the client program: **./tcp_client 127.0.0.1 65530**
3. Input multiple lines in each window, and press the return key multiple times and see what happens. Is the output of each window exactly what you want? Try to figure out why this happens by checking the source code of the chatter client in tcp_client.c.

## Programming II

The chatting client also needs to read data from two inputs: stdin and the socket. The implementation in tcp_client.c using fgets() and readline() that are blocking reading, that is, it will return until the amount of data has been read or the end-of-file is reached. In this task you will implement the chatter client to make the chatting system work as you want using select(). Implement the function given in chatclient.c based on the skeleton code given within the function.

**chatUser()**

- use select() to control reading data from stdin and socket.

NOTE: What is the practical difference, if any, between stdin and STDIN_FILENO in C?
stdin is a FILE * as defined by the standard c library. You can use some of the higher level interfaces like fread, fwrite, and fprintf. On the other hand, STDIN_FILENO is just a file descriptor (almost certainly 0). This uses a slight lower level interface through the likes of read and write.

**Test 1: single machine**

1. Open one terminal window, change into the *Lab3* directory, and type the following command to run the server program: **./chatserver 127.0.0.1 65530**
2. Open multiple terminal windows, change into the *Lab3* directory, and type the following command in each window to run the client program: **./chatclient 127.0.0.1 65530**
3. Input multiple lines in each window, and check the output in each window.

**Test 2: network**

1. Open one terminal window, type **ifconfig** to get the IPv4 address of your local machine.
2. Repeat step 1 in Test 1 by replacing the IP address with the IPv4 address of your local machine.
3. Copy the source package (402lab) to other machines (your laptop with Internet connection or a machine in the lab). Recompile the source files in the Lab3 folder. Repeat step 2 in Test 1 by replacing the IP address with the IPv4 address of the machine you run the server program (i.e. the IPv4 address you get in step 1).
4. Open multiple terminal windows and run the chatclient program at the same time. Check the output in each window.

# Assignment 1

- Due by **4pm on August 19**. Late submission will receive penalty (5% per day).
- This assignment contains all the code you developed for the multi-user chat program in Lab3. Send me chatserver.c and chatclient.c via email.
- This assignment will be marked based on the correctness and robustness of your program.