**UNX511/DSP912 – Lab 7: Server with Multiple Clients**

**Due: Monday, March 16th, 11:59PM**

In this lab you will create a server that communicates with three clients using **select()**. The client code has been given to you along with the Makefile, a start-up script, and a stop script in case you have to manually stop all the clients. You can retrieve these from the following links:

[client.cpp](https://scs.senecac.on.ca/~miguel.watler/courses/unx511/Lab7/client.cpp),

[Makefile](https://scs.senecac.on.ca/~miguel.watler/courses/unx511/Lab7/Makefile),

[startClient.sh](https://scs.senecac.on.ca/~miguel.watler/courses/unx511/Lab7/startClient.sh),

[stopClient.sh](https://scs.senecac.on.ca/~miguel.watler/courses/unx511/Lab7/stopClient.sh).

Your server will use stream sockets in the internet domain. Your socket will bind to the **localhost** and the port number will be specified from the command line. For instance, if you want to use port 1153, you would start your server and clients with:

$ **sudo ./server 1153**

$ **sudo ./startClient.sh 1153**

Be sure to start your server first.

**startClient.sh** will start three clients with a 1 second delay between each.

Once your socket is bound, you will listen for connections.

Be sure to distinguish between a master file descriptor and the connection file descriptors.

You will have one master file descriptor for listening and accepting, but three connection file descriptors for each of the three clients.

Be sure to add your master file descriptor to your list of active file descriptors at the beginning. Then as you accept your connections, be sure to add each connection file descriptor to your list of active file descriptors.

With your **select()** function, you will have to distinguish between all possible communication.

* Is someone requesting a new connection? This will be done on the master file descriptor
* Is client 1 requesting communication? This will be done on the first connection file descriptor.
* Is client 2 requesting communication? This will be done on the second connection file descriptor.
* Is client 3 requesting communication? This will be done on the third connection file descriptor.

When a client requests a new connection, the server will accept it and add the connection file descriptor to the list of active file descriptors. Once that is done, the server will send a message to the client to send text. This is done by writing **“Send Text”** to the client. The client will interpret this correctly and then send a lot of text to the server.

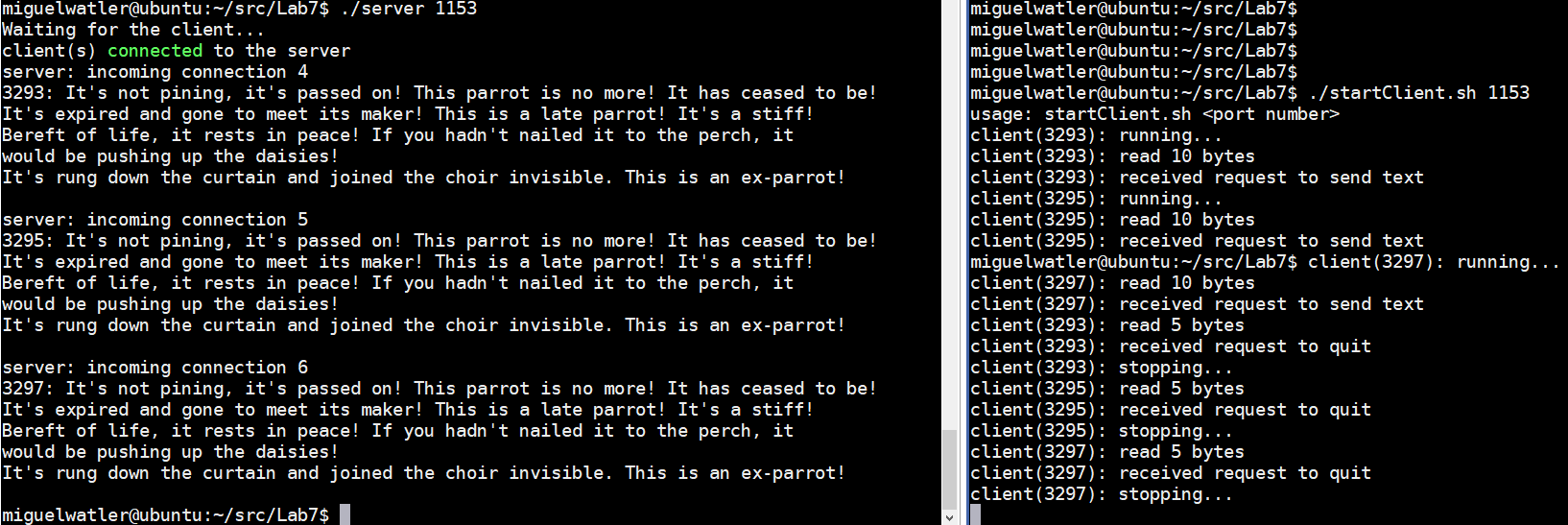
When a client sends text to the server, the server simply prints it out on the screen. For instance, if the server read the text into a variable **buf**, the server will then print it out with:

**cout<<buf<<endl;**

There will come a time when all clients are connected to the server and all have sent their text to the server. At this point the server code will be left hanging on the **select()** function. Be sure to add a timeout of five seconds to the **select()** function so **select()** does not hang the server code.

When **select()** times out on the server, it is time to finish. The server will send **“Quit”** to each client, close all connections, and exit. The clients will interpret **“Quit”** to mean it is time to finish and they will shutdown as well.

A successful run on the server side (left) and the client side (right) will look something like:



For the following questions, please recall the file descriptors for the **stdin**, **stdout**, and **stderr** are 0, 1, and 2 respectively. Please answer the following:

1. In the above there are the debug printout’s:

**server: incoming connection 4**

**server: incoming connection 5**

**server: incoming connection 6**

What do you think the file descriptors are for the first, second and third connection descriptors based on the above debug statements?

1. What do you think the file descriptor is for the master file descriptor?

**Lab Submission:**

Please embed your answers to the above questions in your **server.cpp** and mail your **server.cpp** to:

miguel.watler@senecacollege.ca

**NB: My last name is Watler, not Walter.**