

# **CMSC 3180:**

Data Communications and Networking

Socket Programming Assignment

By:

Anthony Beitko

Margo Bonal

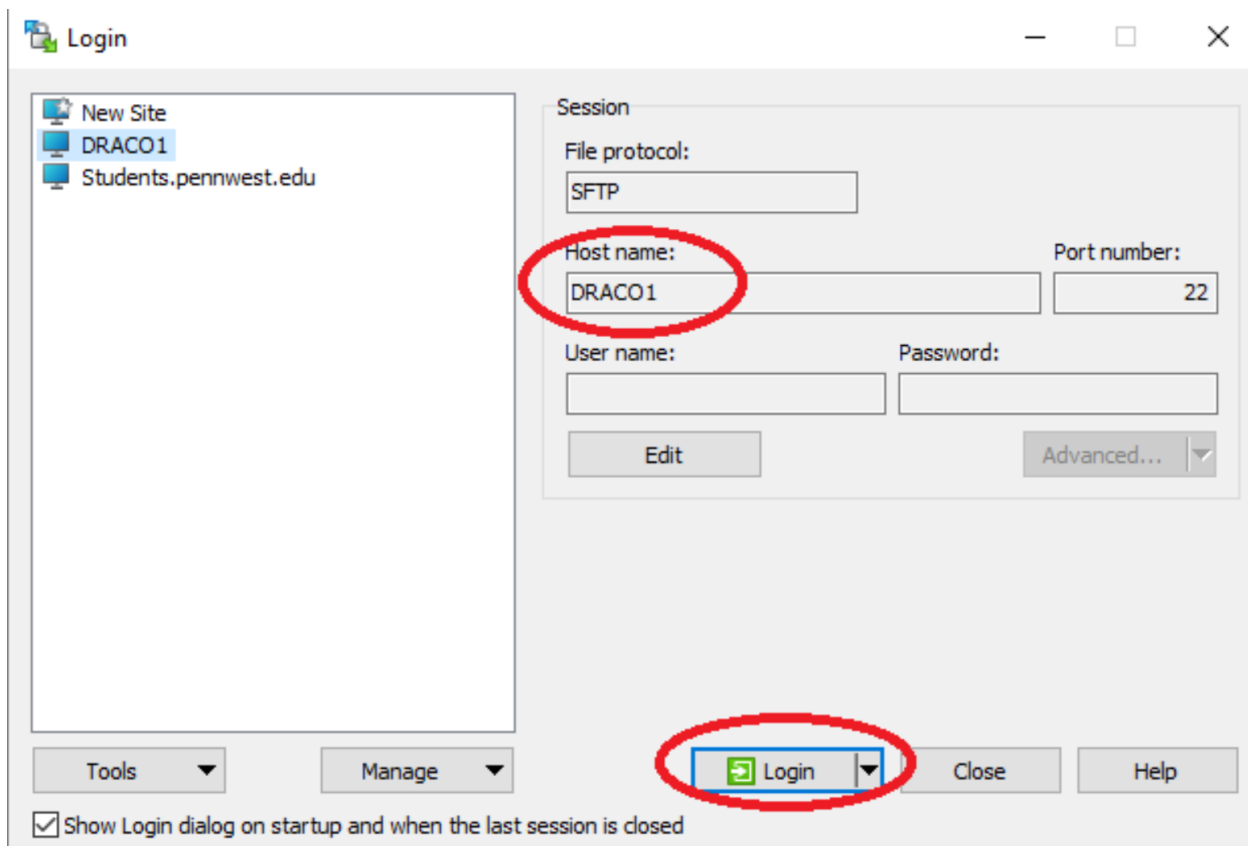
In order to successfully run the socket programming assignment code, you must follow several steps. This document will serve as the instruction manual for running the code in order to witness the workings of the project. It will also define what each student helped with in order to give the professor a good idea on what grade each student should receive.

#### Step 1:

The first step of running the code is downloading the correct programs. The first thing that should be downloaded is the WinSCP. This will allow you to transfer a file from you laptop or PC to the DRACO1 server. If you are off campus, you will also need to install a VPN to connect to DRACO1 in the WinSCP application.

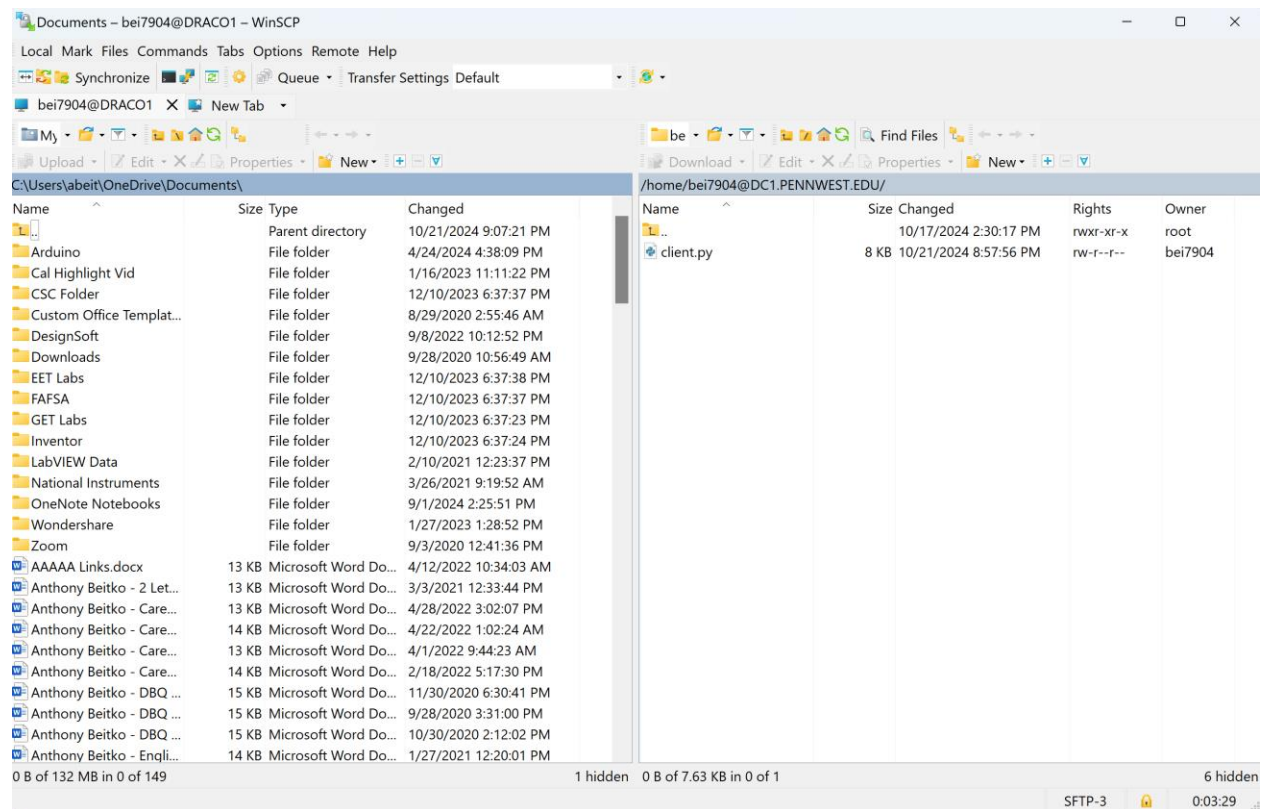
#### Step 2:

The next step is to run the WinSCP program and in the Host name section, type DRACO1. Below that, it asks for you username and password. Enter your Cal U username and password and press the green “Login” button. This will allow you to drag and drop files from your laptop onto the DRACO1 server.



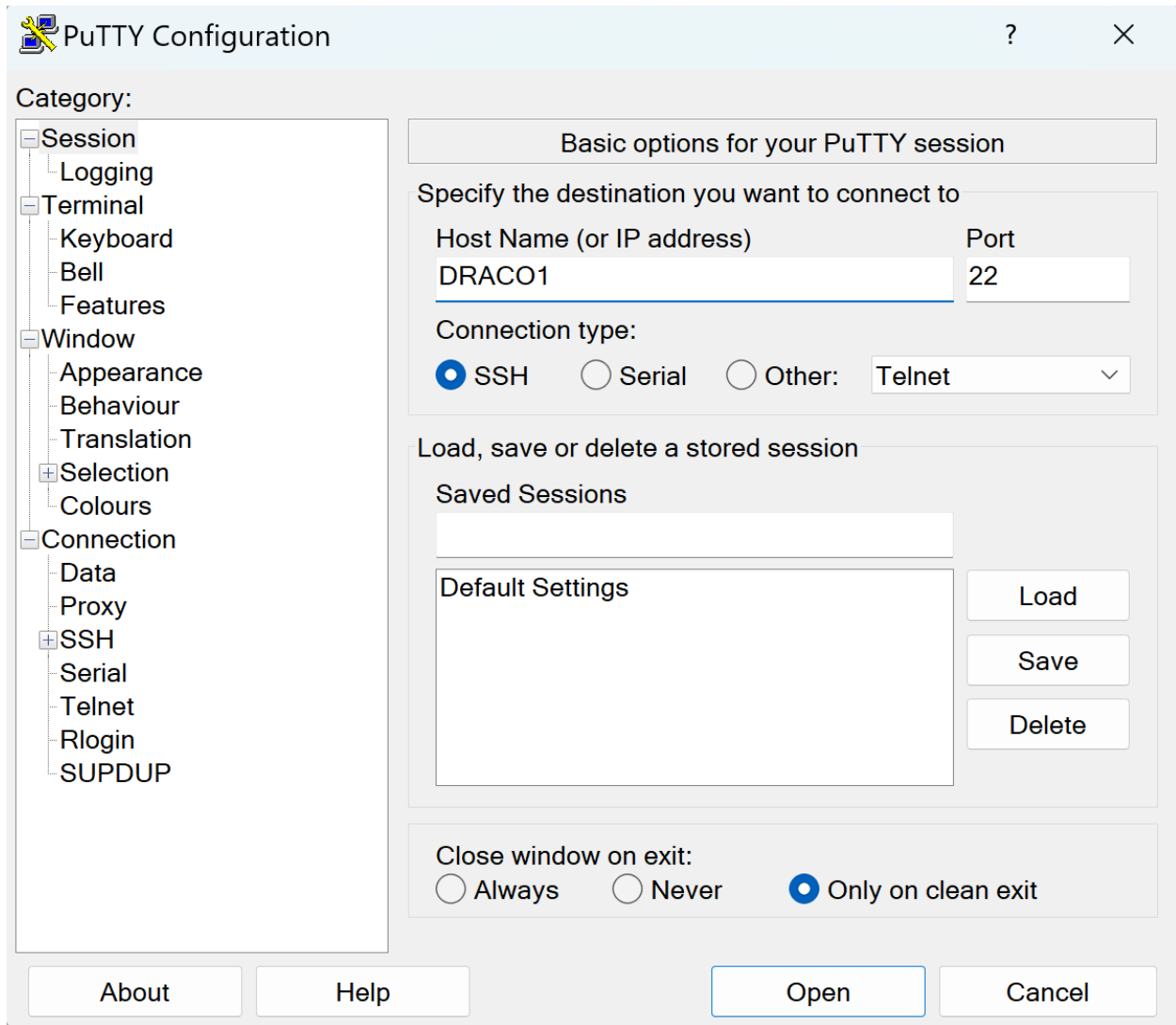
### Step 3:

After this, you should download the code provided. In our case, it is named `client.py` since it is a python file. Once downloaded, move it out of your downloads and into your documents folder. You can now reopen the WinSCP and scroll until you find “`client.py`.” Once you find it, drag it over towards the right side and drop it in. It should now appear on the right side of your screen.



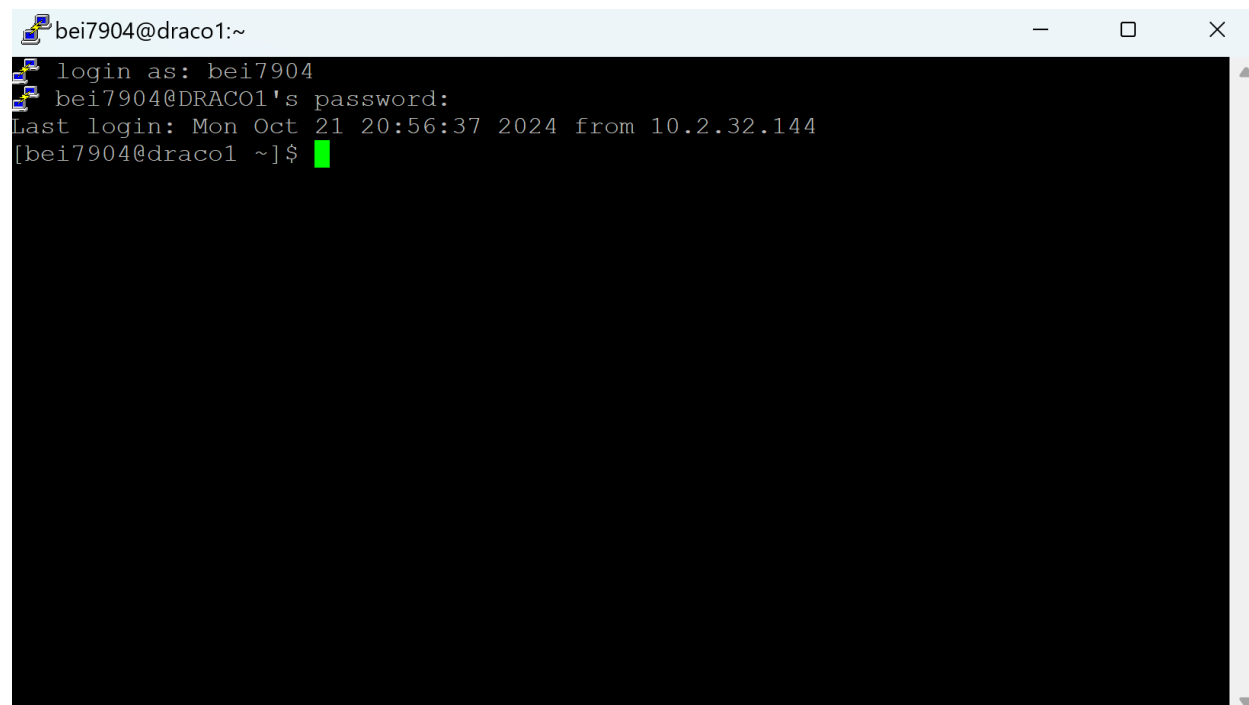
Step 4:

The next step is to open PuTTY on your laptop or PC. You will once again be prompted to enter a Host name. You should type DRACO1.



Step 5:

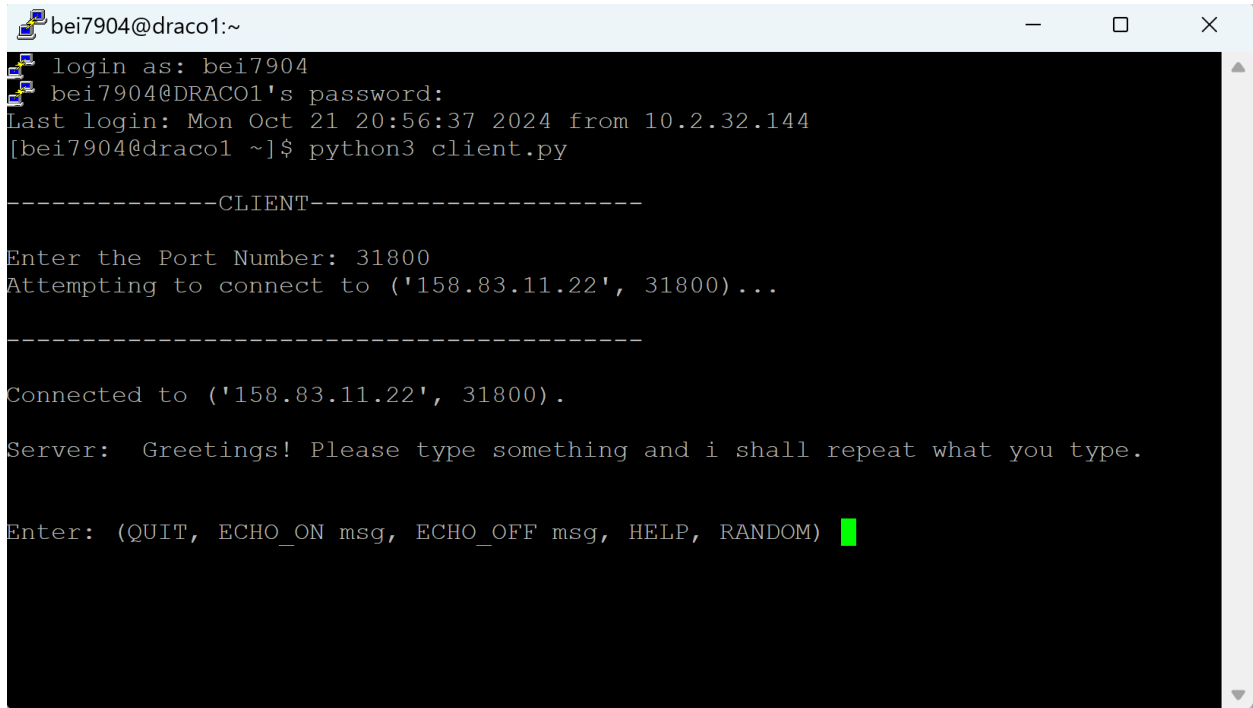
You will now log in one more time using your Cal U credentials.

A terminal window titled 'bei7904@draco1:~' with standard window controls. The terminal output shows a login sequence: 'login as: bei7904', 'bei7904@DRACO1's password:', 'Last login: Mon Oct 21 20:56:37 2024 from 10.2.32.144', and a prompt '[bei7904@draco1 ~]\$' followed by a redacted password. The terminal background is black with white text.

```
login as: bei7904
bei7904@DRACO1's password:
Last login: Mon Oct 21 20:56:37 2024 from 10.2.32.144
[bei7904@draco1 ~]$
```

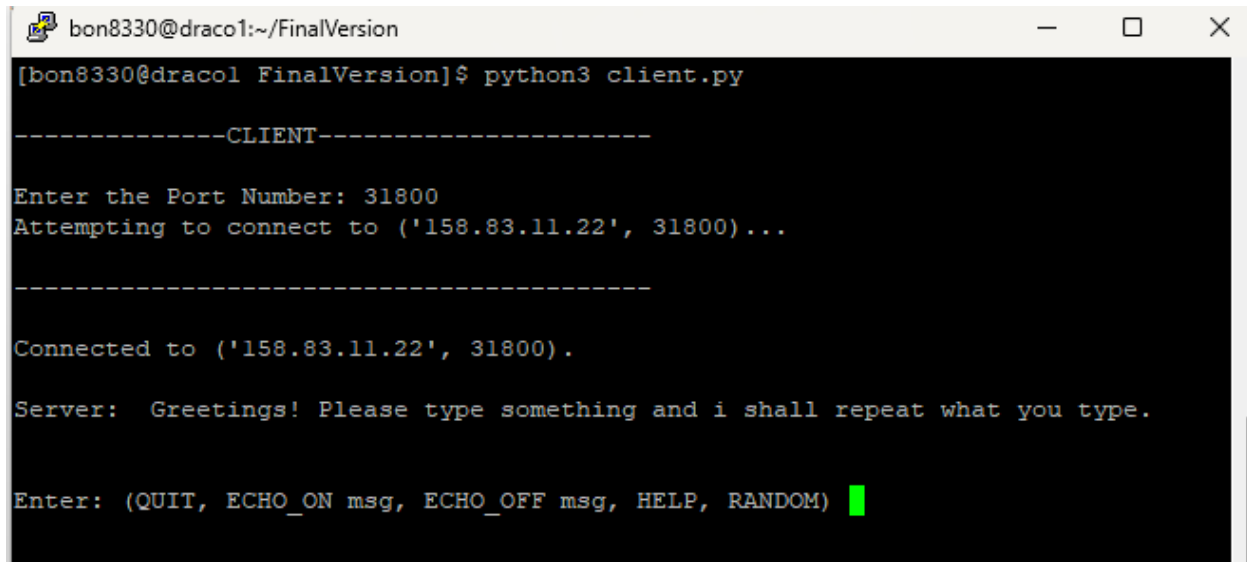
## Step 6:

After you have logged in, you will be free to do what you want. You should type “python3 client.py” which will run the code provided. It will ask for a port number which should be 31800. Once this is complete, you will be free to interact with the program. You are greeted with a message from the server and have many options. These options will be explained below.

A terminal window titled 'bei7904@draco1:~' with standard window controls. The terminal shows a login sequence for user 'bei7904' on host 'DRACO1'. After logging in, the user runs 'python3 client.py'. The program outputs a separator line '-----CLIENT-----', prompts for a port number (31800), and attempts a connection to '158.83.11.22'. Upon successful connection, it displays a greeting from the server and a list of available commands: (QUIT, ECHO\_ON msg, ECHO\_OFF msg, HELP, RANDOM). A green cursor is visible at the end of the command list.

```
bei7904@draco1:~  
login as: bei7904  
bei7904@DRACO1's password:  
Last login: Mon Oct 21 20:56:37 2024 from 10.2.32.144  
[bei7904@draco1 ~]$ python3 client.py  
  
-----CLIENT-----  
  
Enter the Port Number: 31800  
Attempting to connect to ('158.83.11.22', 31800)...  
  
-----  
  
Connected to ('158.83.11.22', 31800).  
  
Server:  Greetings! Please type something and i shall repeat what you type.  
  
Enter: (QUIT, ECHO_ON msg, ECHO_OFF msg, HELP, RANDOM) █
```

# Code Working



```
bon8330@draco1:~/FinalVersion
[bon8330@draco1 FinalVersion]$ python3 client.py

-----CLIENT-----

Enter the Port Number: 31800
Attempting to connect to ('158.83.11.22', 31800)...

-----

Connected to ('158.83.11.22', 31800).

Server:  Greetings! Please type something and i shall repeat what you type.

Enter: (QUIT, ECHO_ON msg, ECHO_OFF msg, HELP, RANDOM) █
```

bon8330@dracol1:~/FinalVersion

```
[bon8330@dracol1 FinalVersion]$ python3 client.py
```

```
-----CLIENT-----
```

Enter the Port Number: 31800

Attempting to connect to ('158.83.11.22', 31800)...

```
-----
```

Connected to ('158.83.11.22', 31800).

Server: Greetings! Please type something and i shall repeat what you type.

Enter: (QUIT, ECHO\_ON msg, ECHO\_OFF msg, HELP, RANDOM) ECHO\_ON hello

ECHO Mode ON

```
-----
```

Client: hello

Server: hello

```
-----
```

Enter: (QUIT, ECHO\_ON msg, ECHO\_OFF msg, HELP, RANDOM) echo\_on trial

ECHO Mode ON

```
-----
```

Client: trial

Server: trial

```
-----
```

Enter: (QUIT, ECHO\_ON msg, ECHO\_OFF msg, HELP, RANDOM) ECHO\_OFF class

ECHO Mode OFF

```
-----
```

Client: class

```
-----
```

Enter: (QUIT, ECHO\_ON msg, ECHO\_OFF msg, HELP, RANDOM) █



```
Enter: (QUIT, ECHO_ON msg, ECHO_OFF msg, HELP, RANDOM) HELP
```

```
-----
```

OPTIONS:

1. QUIT: Close Connection.
  2. ECHO\_ON <msg>: Enable echo mode and send message.
  3. ECHO\_OFF <msg>: Disable echo mode and send message.
  4. HELP: Display each command and information.
  5. RANDOM: Send a random message to server.
- ```
-----
```

Client: HELP

Message sent to server, no echo.

```
Enter: (QUIT, ECHO_ON msg, ECHO_OFF msg, HELP, RANDOM) █
```

```
Message sent to server, no echo.
```

```
Enter: (QUIT, ECHO_ON msg, ECHO_OFF msg, HELP, RANDOM) RANDOM
```

```
-----
```

Client RANDOM: Anthony

```
-----
```

```
Enter: (QUIT, ECHO_ON msg, ECHO_OFF msg, HELP, RANDOM) RANDOM
```

```
-----
```

Client RANDOM: Margo

```
-----
```

```
Enter: (QUIT, ECHO_ON msg, ECHO_OFF msg, HELP, RANDOM) RANDOM
```

```
-----
```

Client RANDOM: Comp Networking

```
-----
```

```
Enter: (QUIT, ECHO_ON msg, ECHO_OFF msg, HELP, RANDOM) random
```

```
-----
```

Client RANDOM: Margo

```
-----
```

```
Enter: (QUIT, ECHO_ON msg, ECHO_OFF msg, HELP, RANDOM) █
```

```

Enter: (QUIT, ECHO_ON msg, ECHO_OFF msg, HELP, RANDOM) echo_on program

ECHO Mode ON
-----
Client: program
Server: program
-----

Enter: (QUIT, ECHO_ON msg, ECHO_OFF msg, HELP, RANDOM) QUIT
Closing connection...
[bon8330@dracol FinalVersion]$

```

```

-----CLIENT-----

Enter the Port Number: abcd
***INVALID Port Number! Please re-enter a Valid Integer***

Enter the Port Number:

```

```

Connected to ('158.83.11.22', 31800).

Server: Greetings! Please type something and i shall repeat what you type.

Enter: (QUIT, ECHO_ON msg, ECHO_OFF msg, HELP, RANDOM) bad command
**INVALID COMMAND! Please enter a valid option.**

Enter: (QUIT, ECHO ON msg, ECHO OFF msg, HELP, RANDOM)

```

```

-----CLIENT-----

Enter the Port Number: 0000
Attempting to connect to ('158.83.11.22', 0)...

-----

Failed to connect to server at ('158.83.11.22', 0). Please check the server and
try again.

```

# OPTIONS

QUIT – This will close the connection and end the program.

ECHO\_ON <msg> - The message typed will be repeated by the server and sent back to you.

ECHO\_OFF <msg> - The message typed will be acknowledged, but the server will not reply.

HELP – This will explain all of the options available along with what each option does.

RANDOM – This will choose a random message for the client to send to the server which is acknowledged but not repeated by the server.

# Contributions

## Anthony Beitko

- Assisted with downloading of programs and writing the first version of code
- Visited Office Hours to ask questions about the program
- Wrote Instruction Manual
- Tested code in several forms in an attempt to find bugs

## Margo Bonal

- Researched socket strategies
- Programmed/ built client.py
- Designed/ implemented Echo On and Off modes
- Designed/ implemented protocol and quit methods
- Added 2 Bonus features: Random Message and Help Screen
- Documentation of code/created readme.txt
- Added validity checks on all data entry points
- Added readability/ formatting
- Visited Office Hours to ask questions about the program
- Tested/ debugged using Dracool, PuTTY, WinSCP, and VPN