# COMP3331 report

Yun Li (z5231701)

## *Program design:*

I designed my program as client-server form application. Except receiving and sending the file, most of action is handled by server and clients only need to receive the message the server sends to and print it out.

I built array to store following information:

Usernames of people who has logged in and are in the forum now

String of all possible commands

All_fname to store names of all files which should be deleted after the server shutsown

And I built dictionary to store following information:

Name_pass which shows the inputs the username and outputs the password

Thread and its current number of messages (for generating the message id of a message)

Thread and its current uploaded file name

And I used a Boolean variable to show if the forum still alive.

I have a function call threaded, my main design for multi thread is creating threaded function for each thread and process the command from each client in their corresponding threaded function.

I think an interesting part of my design is that I create a class for client in client.py, and I created a thread for receiving message from server so that the sending and the receiving won't disturb each other, and it's also easier for me to receive shut down message from server.

## *Application layer message format:*

About the message format, most of what I did is following the message in assignment specification.

My idea to send feedback or the result back to client regardless the action succeeds or not. If failed, the reason will be given for client's further instruction. During the file transfer, the size will be sent so I also let the receiving side to print the size of file it is going to receive.

To process different command, I wrote a very long if else block. I know it's wrong but I think if I implement a function for each command, the possibility I reuse the function is very low, so I give up. Except for file transfer, all the data I transfer is in type Byte (convert from string type most of time), and I also encode before sending them to protect user's privacy. For file transfer（UDP and DWN）, I directly send them in binary form which is efficient.

## *A brief description:*

TCP is always used while UTP is never used in assignment as Lecture asked.

the loop which processing different command will only

During the sending and receiving of a file, I send the size of file first to know how large is the file then I know where to stop the loop of receiving the file.

I used string.split() to generate the whole command from the message

I used os.path.exists() to find out if a file with given directory.

## *Any design trade-off considered:*

I thought about using loop of small size to receive the function but it’s too complex for me.

I thought about building server by using python class, but I later think it’s not really necessary。

To process different command, I wrote a very long if else block.

## Program possible improvement:

Maybe building a class for server next time.

Separate different command into different sub function under the server class and client class.

Wrote more helper function in client.

## Program environment:

Python 3.7.7, and I do most my work on windows

### *Any code segment borrowed form Web or other sources:*

My idea of multi thread model is from:

https://www.geeksforgeeks.org/socket-programming-multi-threading-python/