ELEC 3120: Computer Communication Networks
Prof. Tsang, Danny H. K.
(Fall 2020)

Department of Electronic and Computer Engineering
The Hong Kong University of Science and Technology

Programming Assignment 1

Notes:

1) In your source code, please write down proper comments. You may discuss ideas with others in the class, but you should implement your own program. Do not copy from others.

2) You are welcome to use any programming language. Python 3 is preferred, because you can refer to the lecture slides and textbook for rich sample programs.

3) **Source code submission:** Please compress the source and executable files into one ZIP file, with your student ID as the file name. Upload the ZIP file to the "programming assignment 1" field, under the "assignments" section of the ELEC3120 Canvas website at http://canvas.ust.hk. **The deadline for source code submission is hard at 5 p.m. on November 6, 2020, with NO late submission acceptable.**

Echo Server

In the first part of this assignment you are going to implement an echo server which receives text message from client and sends it back to the client. (That's why we call it echo server) When the server receives the text message from client, it capitalizes the odd characters of the text message. For example, when the client sends a text message of "hello elec 3120 student", the server should return a text message of "HeLlO ElEc 3120 StUdEnT".

Detailed requirements:

1. Implement the server in both TCP and UDP and allow the user to choose which protocol to use at runtime.
2. For TCP programming, your program should be a persistent server so that after transmitting each packet, the TCP connection should be kept alive.
3. The server only needs to serve one client at a time (no multithreading).
4. When the server receives a message from the client, print it out on the screen with the client's IP address on the server side.
5. After printing the message on the screen, the server capitalizes the odd characters of the message and sends the modified version back to the client.
6. When a client disconnects, the server closes the socket and waits for the next client to connect.
7. Handle common errors in socket programming (e.g., assigned port being occupied by other programs, client resetting the connections via control-c).

Echo Client

In the second part of this assignment you are going to implement an echo client which connects and sends plain text messages to an echo server. Moreover, when it receives message from the echo server, it directly prints the message out on the screen.

Detailed requirements:

1. Implement the client in both TCP and UDP and allow the user to choose which protocol to use at runtime.

2. For TCP programming, your program should be "persistent", so that after each packet's transmission, the TCP connection should be kept alive.
3. The client only connects to one echo server at a time (no multithreading).
4. Users input messages using the keyboard, and the client transmits messages to the connected server.
5. When a message is received from the server, print it out on the screen on the client side.
6. When a user types "bye", the client program closes all connections and quits.
7. Handle common errors in socket programming (e.g., assigned port being occupied by other programs, client resetting the connections).