COLUMBIA UNIVERSITY
CSEEW4119 COMPUTER NETWORKS
Programming Assignment 1 - Simple Chat Application

Abstract:

The objective of this programming assignment is to implement a simple chat application with at least 3 clients and a server using UDP. The program has two modes of operation, one is the client, and the other is the server. The client instances communicate directly with each other. The server instance is used to set up clients and for book-keeping purposes. The server is also used to store off-line messages from clients and broadcast channel messages to all clients within a predefined communication channel (group chat).

Author:
Liping Hu, lh3009

Files:

Purpose	File Name
Main program	ChatApp.py
README file	README.pdf
4 example runs of the program	test.txt

To Run The Program:

1. Run this in terminal to start the server instance:

   `ChatApp -s <port>`

2. Run this in terminal to start any clients instance:

   `ChatApp -c <name> <server-ip> <server-port> <client-port>`

Internal Design:

- General structure:

  For the client part, I used two threads, the main thread and the listening thread. In the main thread, the program constantly monitors if the client is inputting anything, and if there is, it validates the input and sends the request to the server or specific client receiver based on the input message. Inside the listening thread, the client constantly listens to replies from the server. And if any reply from the server/client comes through, the program sends ACK or displays the received message to the appropriate recipient.

  For the server part, I also used two threads. What is different is that for one request, there might be multiple messages from different clients. Therefore, I used a queue for storing the messages received at the server side for each request in a listening thread by using queue.put(). Eg: multiple acknowledgements from multiple clients of one "send_all" message. Then in the main thread, I am dealing with those messages one by one by calling queue.get(), after each message is dealt with, the message is removed from the queue.

  I also used dictionary to store the clients information and it is used to store both serverTable and clientTable