Assignment #1: A simple multi-client chat server

Problem description

In this assignment, you will write a server code and client code to implement a multi-client chat server application. The clients will chat with one other and the server will manage the overall communication.

Server behavior. The server will always listen (in an infinite loop) for incoming client connections. Upon accepting a client connection, the server will assign a unique ID (e.g., 101) to the client. This ID will be sent to the client as the first message from the server. The message to be sent is "Your ID is <ID>." For example, if a new client joins and the server assigns the next ID 105 to this client, then the server will send the first message to the client as "Your ID is 105".

- Afterwards, the server will continue to receive messages from the client (should be implemented as a separate thread in the server) in the format "<DEST_ID> <MESSAGE>". For example, a client (with ID 101) may send a message "How are you?" to another client with ID 110 in the format "110 How are you?." After receiving this message, server thread should forward (i.e., send) the message to the client 110 in the format "<SRC_ID> > <MESSAGE>". For example, the above message from client 101 will be forwarded to the client 110 as "101> How are you?"

Client behavior. Each client will connect to the server. After successful connection, the client receiver thread will receive the first message from server which is the client ID. A client sender thread will continuously send messages to the other clients as described above (message will be read from the user). The server thread will receive the message and forward (i.e., send) this to the appropriate client. In this way, clients will chat with one another with the help of the server. A client may leave the chat server by sending a special message "Bye" to the server.

Other features.

- Server acts as a router of messages. Server does not initiate any communication. Only clients send message to one another.
- Client ID numbers must be unique. A simple solution is to start with 101 and then incrementing the ID after each successful client connection to the server.
- Server needs to keep track of ID numbers against the sockets that handles the respective client. It will be needed for message routing. When a client wants to send a message to client 110 and server receives this message, server needs to track the appropriate socket that was created for handling communication for client 110. A global array may be used for this purpose. Note that a thread can use the global variables declared within the same program.
- A client can send a message to any other client (they only need to know the ID of the destination client). However, if the server receives a message where <DEST_ID> is invalid, then the server can just ignore the message.
- For this assignment, you need not require using any synchronization solutions.
- For any other queries, please let us know.