Assignment 2

Mehtab Kayani
ID: 9497

TCP-Chat

1. Describe your algorithm design, showing what interactions occur between the clients and the server.
Answer: The description for the server and client algorithms can be found under the respective headings.

Server:
For running server I am using the **start()** method . The port number I am using is 6789, and I am setting backlog limit equal to 100 users.

Algorithm Design

Try

// connect and have conversation

waitConnection() // waiting for the users to connect to the server

setStreams() // setting up the input and output streams, this method helps to send messages and receive messages, streams are like pathways

duringChat() // it allows to communication with one another during the chat session

Catch

End of the stream

Once the streams are set and users are connected **enableTyping()** method allows the users to type in textArea which was previously inactive before the connection. **sendMessage()** helps the users to send messages.

If the connection from the client side is established the success message is displayed exploiting the function **displayMessage().**

The **disconnect()** method closes the connection by closing the input stream, output stream and sockets. During the chat session, if the user types **"/quit"** in the chat window he/she gets disconnected.

Client:

The Client constructor accepts the serverIP as the parameter.I set the JTEXTFIELD to not editable before the connection.

Algorithm Design

Try

// connect and have conversation

serverConnection() // Connects the user to the server

setStreams() // setting up the input and output streams, this method helps to send messages and receive messages, streams are like pathways

duringChat() // it allows to communication with one another during the chat session

Catch

End of the stream

Once the server is up and running, the client.java is run. The window is displayed showing the message trying to establish connection and the connection is established through the function **serverConnection().**

ServerIP and portNumber are required for the connecting to the server. The user can send and receive messages through the method **sendMessage(),** once the connection is established.
I am using the **actionListener()** and **actionPerformed()** methods, the **actionListener()** listens to the message typed in JTEXTFIELD and **actionPerformed()** method sends the message with the help of **sendmessage()** function.
You can close both the server and client ChatSystem window by simplying closing the window or by typing **"/quit"** in the text field in the window. It works the same for both sides.

3.Test your implementation and report on possible bugs and/or unexpected behaviours you should find.
Answer:
SERVER:
I successfully tested the server in the main method by creating a new object of the server class and calling the **start()** method of the server class.

CLIENT:
I successfully tested the client in the main method by creating a new object of the client class and passing the IP address "127.0.0.1" to it, which connects it to the server. After that I called the **start()** method of the client class on the new created object of the client class.

4. Shortly discuss a possible alternative implementation exploiting the UDP protocol, making a critical comparison between the two.

Answer:

There can be an alternative implementation using UDP but TCP is far better in our case than UDP, because it is reliable and guarantees that the data transmission remains intact and arrives in the same order as it was sent.

While on the other hand UDP has no guarantee that the sent messages or packets would reach at all. Hence, in our case if we use UDP we can lose the chat messages.

In terms of size of programs, TCP is good for the programs with bigger size. The maximum limit for TCP servers is 65536 connections at one time. In case of exceeding this number, we can create a dispatcher server that would send incoming connections to the server depending on the current server load. UPD can faster for small programs. TCP handles congestion control while UDP does not have any option for flow control.