Overview

Overview: For this practical we were asked to build a pair of client and server programs that can communicate with each other and send messages between each other using TCP and sending ASCII strings. We also had to make these programs communicate over the host servers provided by the school. I achieved all the requirements to a full extent.

Design

Both Classes

I check the argument length to see if the user has input the required 2 arguments of IP address and port and if they haven't then it outputs the correct usage.

Then I declare the necessary variables for the rest of the program like the response and received strings.

Then inside a try statement I create a socket with the IP and port, an input and output stream, and a reader and writer.

Simple Client

I use the socket to get the local address and I send that with the writer to the server class, so the server class knows the client's correct address for when they write it in.

Then I have a while loop that will loop the main "back and forth" communication section of the program until the server's "BYE" has been received and then the loop is finished and the socket is closed.

Finally, there is the catch statement which will print out the relevant I/O error if one is caused by trying to make a wrong socket.

Simple Server

Similarly, the server receives the client's address and then saves it for later.

Then there is the same while loop for the communication except in this one there is an if statement to check first of all if the client sends their real address, which it can check using the one it received, and if it has then it sets it response to the server's address. Then checks for a "BYE" and if one is received then the finished variable is set to true and the response is set to "BYE". Then the writer sends out the response.

When the while loop is closed the socket is closed, and again there is the same catch statement as before.

Testing

General Conversation and Socket Creation Testing

```
dtl1@lyrane:~/Documents/cs2003 $ javac SimpleServer.java
dtl1@lyrane:~/Documents/cs2003 $ java SimpleServer 21929
```

```
dtl1@klovia:~/Documents/cs2003 $ javac SimpleClient.java
dtl1@klovia:~/Documents/cs2003 $ java SimpleClient 138.251.22.77 21929
hi
SERVER: HELLO
HELLO
SERVER: HELLO
hey
SERVER: HELLO
ADDRESS: /138.251.69.69
SERVER: HELLO
ADDRESS: /138.251.22.78
SERVER: ADDRESS: /138.251.22.77
hi
SERVER: ADDRESS: /138.251.22.77
BYE
SERVER: BYE
```

As you can see from these screenshots the Server successfully starts with the port 21929 on the lyrane host server.

Then the client connects to the right IP for the lyrane host server from the klovia host sever.

You can see the server replies with HELLO until the client inputs their correct IP.
Even when the client inputs an IP with the correct format, the server still knows it is not correct.
Then after the correct IP is input the server responds with its IP and continues to until the client inputs BYE.
When BYE is input, the server responds and the program ends.

These IP's are correct and match up with the host server table.

Hostname	*IP address*
klovia.cs.st-andrews.ac.uk	138.251.22.78
lyrane.cs.st-andrews.ac.uk	138.251.22.77
palain.cs.st-andrews.ac.uk	138.251.22.76
trenco.cs.st-andrews.ac.uk	138.251.22.79

Args correct usage testing

```
dtl1@klovia:~/Documents/cs2003 $ java SimpleClient a
Wrong number of arguments
Correct Usage: java SimpleClient <ip address> <port>
```

```
dtl1@lyrane:~/Documents/cs2003 $ java SimpleServer
Wrong number of arguments
Correct Usage: java SimpleServer <port>
```

Incorrect IP testing

```
dtl1@klovia:~/Documents/cs2003 $ java SimpleClient 138.251.22.69 21929
I/O errorjava.net.NoRouteToHostException: No route to host (Host unreachable)
```

Evaluation

To evaluate, my program was successful and achieved everything it was asked to do, it compiles and runs without any known errors and achieves every task described in the specification and was sufficiently tested/

Conclusion

To conclude, I achieved a complete and robust program, I found it difficult to work out what to do with the host servers at first and as well using ssh and transferring files to my linux machine using git as it's the first time I did that. I'm not sure what I would have done had I had more time, maybe made the console output laid out nicer.