| **Testcase** | **Pass/Fail** |
| --- | --- |
| **Testcase 2001**  Server startup check with default arguments  Instructions:   1. Start the server program   Expected result:   1. The server reports that it is listening for clients by displaying the following message:   *Server listening for clients on port 5555*   1. The server console waits for user input.   Cleanup:  Terminate the server program.  ------------------------------------------------------------------------------------------------------------------------  TEST  SERVER SIDE  *#start*  *Server listening for connections on port 5555* | **PASS** |
| **Testcase 2002**  Client startup check without a login  Instructions:   1. Start the Client program without specifying the loginID as an argument.   Expected result:   1. The client reports it cannot connect without a login by displaying:   *ERROR - No login ID specified. Connection aborted.*   1. The client terminates.   Cleanup: (if client is still active)  Terminate the client program.  ------------------------------------------------------------------------------------------------------------------------ | **CANNOT TEST** |
| **Testcase 2003**  Client startup check with a login and without a server  Instructions:   1. Start the Client program while specifying loginID as an argument.   Expected result:   1. The client reports it cannot connect to a server by displaying:   *ERROR - Can't setup connection! Terminating client.*   1. The client terminates.   Cleanup: (if client is still active)  Terminate the client program. | **Pass~~** |
| **Testcase 2004**  Client connection with default arguments  Instructions:   1. Start a server (Testcase 2001, instruction 1) 2. Start a client (Testcase 2003, instruction 1)   Expected results:   1. The server displays the following messages in sequence:   *A new client has connected to the server.*  *Message received: #login <loginID> from null.*  *<loginID> has logged on.*  **Note:** the server specifies that it received a message from null as this is the first message received from this client. It will record the loginID of this client for later messages. Hence, for later messages, it should display:  *Message received: <user input> from <loginID>*  Where <user input> is the content of the message received and <loginID> is the loginID of the sending client.   1. The client displays message:   *<loginID> has logged on.*   1. The client and the server wait for user input.   Cleanup: (unless proceeding to Testcase 2005)  Terminate the client program.  Terminate the server program.  ------------------------------------------------------------------------------------------------------------------------  TEST  SERVER SIDE  *#start*  *Server listening for connections on port 5555*  *Message received: #login 17 from null. 17 has logged on.*  *A new client has connected to the server.*    CLIENT SIDE  *> 17 has logged on* | **PASS** |
| **Testcase 2005**  Client Data transfer and data echo  Instructions:   1. Start a server and a client using default arguments (Testcase 2004 instructions). 2. Once connected, type in data on the client console and press ENTER.   Expected results:   1. The message is echoed on the client side, but is preceded by the sender's loginID and the greater than symbol (">"). 2. The server displays a message similar to the following:   *Message received: <user input> from <loginID>*  Cleanup:  Terminate the client program.  Terminate the server program.  ------------------------------------------------------------------------------------------------------------------------  TEST  SERVER SIDE  *#start*  *Server listening for connections on port 5555*  *Message received: #login 17 from null. 17 has logged on.*  *Message received: testing testing 123 from testguy*  CLIENT SIDE  ***testing testing 123***  ***> testguy > testing testing 123*** | **PASS** |
| **Testcase 2006**  Multiple local connections  Instructions:   1. Start a server and multiple clients with DIFFERENT loginIDs and connect them to the server using default arguments. (Testcase 2005 instructions). 2. Start typing on all the client consoles AND the server console, pressing ENTER to send each message.   Expected results:   1. All client messages are echoed as in Testcase 2005. 2. All messages from the server console are echoed on the server console and to all clients, but are preceded by "SERVER MESSAGE> ".   Cleanup:  Terminate the clients.  Terminate the server program.  ------------------------------------------------------------------------------------------------------------------------  TEST  SERVER SIDE  *#start*  *Server listening for connections on port 5555*  *Message received: #login 17 from null. 17 has logged on.*  *Message received: #login 18 from null. 18 has logged on.*  *hi*  *Message received: hey from anotherguy*  *Message received: heyo! from testguy*  CLIENT SIDE 0 (test guy)  *> <SERVER MSG> : hi*  *> anotherguy > hey*  *heyo!*  *> testguy > heyo!*  CLIENT SIDE 1 (another guy)  *> <SERVER MSG> : hi*  *hey*  *> anotherguy > hey*  *> testguy > heyo!* | **PASS** |
| **Testcase 2007**  Server termination command check  Instructions:   1. Start a server (Testcase 2001 instruction 1) using default arguments. 2. Type "#quit" into the server's console.   Expected result:   1. The server quits.   Cleanup (If the server is still active):  Terminate the server program.  TEST  SERVER SIDE  *#start*  *Server listening for connections on port 5555*  *#quit*  *Shutting down server.*  *Server has stopped listening for connections.* | **PASS** |
| **Testcase 2008**  Server close command check  Instructions:   1. Start a server and connect a client to it. (Testcase 2004) 2. Stop the server using the #stop command. 3. Type "#close" into the server's console.   Expected result:   1. Server displays in sequence:   *Server has stopped listening for connections.*  *<loginID> has disconnected.*   1. The client displays:   *The server has shut down.*   1. The client terminates   Cleanup:  Terminate the client program.  Terminate the server program.  ------------------------------------------------------------------------------------------------------------------------  TEST  SERVER SIDE  *#start*  *Server listening for connections on port 5555*  *Message received: #login 17 from null. 17 has logged on.*  *#stop*  *Server has stopped listening for connections.*  *#close*  *Closing server.*  *Client null a deconnecte au serveur.*  CLIENT SIDE  *> The server has shut down* | **Pass~** |
| **Testcase 2009**  Server restart  Instructions:   1. Start a server. 2. Close the server using the #close command. 3. Type "#start" into the server's console. 4. Attempt to connect a client.   Expected result:   1. The server closes, restarts and then displays:   *Server listening for connections on port 5555.*   1. The client connects normally as described in Testcase 2004.   Cleanup:  Terminate the client program.  Type #quit to kill the server.  ------------------------------------------------------------------------------------------------------------------------  TEST :  SERVER SIDE  *#start*  *Server listening for connections on port 5555*  *#close*  *Closing server.*  *Server has stopped listening for connections.*  *#start*  *Server listening for connections on port 5555*  *Message received: #login 18 from null. 18 has logged on.* | **PASS** |
| **Testcase 2010**  Client termination command check  Instructions:   1. Start a server 2. Connect a client. 3. Type "#quit" into the client's console.   Expected result:   1. Client terminates.   Cleanup: (If client is still active)  Terminate the client program.  ------------------------------------------------------------------------------------------------------------------------  TEST  SERVER SIDE  *#start*  *Server listening for connections on port 5555*  *Message received: #login 17 from null. 17 has logged on.*  *anotherguy disconnected*  *#quit*  *Shutting down server.*  *Server has stopped listening for connections.*  CLIENT SIDE  *#quit*  *Quitting*  *Connection closed*  *Process finished with exit code 0* | **PASS** |
| **Testcase 2011**  Client logoff check  Instructions:   1. Start a server (Testcase 1001, instruction 1), and then connect a single client to this server. 2. Type "#logoff" into this client's console.   Expected results:   1. Client disconnects and displays Connection closed.   Cleanup:  Type "#quit" to kill the client.  ------------------------------------------------------------------------------------------------------------------------  TEST  SERVER SIDE  *#start*  *Server listening for connections on port 5555*  *Message received: #login 19 from null. 19 has logged on.*  *Client null a deconnecte au serveur.*  *#logoff*  *Closing connection to server host localhost on port 5555.*  CLIENT SIDE  *#logoff*  *Closing connection to server host localhost on port 5555.*  *#quit*  *Quitting*  *Unexpected error while reading from console!*  *Process finished with exit code 0* | **PASS** |
| **Testcase 2012**  Starting a server on a non-default port  Instructions:   1. Start a server while specifying port **1234** as an argument.   Expected result:   1. The server displays   *Server listening for connections on port 1234.*  Cleanup:  Type #quit to kill the server.  Server does not take the cmd line arguments: -…   * I am using IntelliJ as I had problems with Eclipse * Was not able to run anything from cmd line ( had to use the run button ) * Was able to insert cmd line arguments while editing the run configuration | **FAIL** |
| **Testcase 2013**  Connecting a client to a non-default port  Instructions:   1. Start a server on port 1234 2. Start a client with the arguments: **<loginID> <host> 1234**   (replace the parameters by appropriate values).  Expected Result:   1. The connection occurs normally.   ------------------------------------------------------------------------------------------------------------------------  TEST ::: The instructions do not specify that the server must have its port set through arguments before starting and thus using #setport does not break the rules of the testcase  SERVER SIDE  *#setport 1234*  *Port set to 1234.*  *#start*  *Server listening for connections on port 1234*  *Message received: #login 17 from null. 17 has logged on.* | **PASS** |