Phase 1

Testcase 1 - Server startup non-default port
Instructions:
1. At the console, enter java Echoserver 5565
Expected result:
1. The server reports that it is listening for clients by displaying the following message: Server listening for clients on port 5565
Why: We want to make sure that a server can run with a non-default port without crashing

Testcase 2 - Remote Client Disconnect
Instructions:
1. Start a server and remote connection
2. Terminate the client
3. Start another client connection
Expected result:
1. Server does not display any information regarding disconnection of first client, then accepts the second connection.
Why: We want to let one client disconnect from a remote server without bringing the whole server down

Testcase 3 - Empty string echo
Instructions:
1. Start a server and client connection
2. Press enter in client
Expected result:
1. New line is printed with '>' character
Why: We want to make sure that the client handles empty strings smoothly

Testcase 4 - Multiple servers same port
Instructions:
1. Start a server with default arguments
2. Start another server on the same machine with default arguments
Expected result:
1. First server starts as normal
2. Second server fails to start with message 'ERROR - Could not listen for clients!"
Why: It would not make sense for two servers to share a port on one computer, so we want to make sure the second server shuts down cleanly without bringing down the first server

Testcase 5 - Multiple servers different port
Instructions:
1. Start a server with default arguments
2. Start another server on the same machine with argument 5565
Expected result:

1. First server starts as normal with message ‘Server listening for connections on port 5555’
2. Second server starts as normal with message ‘Server listening for connections on port 5565’
Why: We want to be able to run multiple chat servers based off of one machine

Testcase 6 - Starting server with incorrect arguments
Instructions:
1. Start a server with argument ‘abcd’
Expected result:
1. Server starts as normal with message ‘Server listening for connections on port 5555’
Why: We want the server to gracefully handle arbitrary command line arguments

Testcase 7 - Multiple servers client connection
Instructions:
1. Start a server with default arguments and a server with argument 5565 on the same machine
2. Start a client on the same machine
Expected result:
1. Both servers start
2. Client does not display anything
Why: We want to be able to run multiple chat servers based off of one machine

Testcase 8 - Multiple servers client chat
Instructions:
1. Start a server with default arguments and a server with argument 5565 on the same machine
2. Start a client on the same machine
3. Type a message into the client and press enter
Expected result:
1. Both servers start
2. Client echos back message
3. Server one displays ‘Message received: MESSAGE from localhost (127.0.0.1)’ with the message typed into the client
4. Server two doesn’t display anything
Why: We want to be able to run multiple chat servers based off of one machine

Testcase 9 - Multiple servers client remote connection
Instructions:
1. Start a server with default arguments and a server with argument 5565 on the same machine
2. Start a client on a different machine
Expected result:
1. Both servers start
2. Client does not display anything and is waiting for input
Why: We want to be able to run multiple chat servers based off of one machine

Testcase 10 - Multiple servers client chat
Instructions:
1. Start a server with default arguments and a server with argument 5565 on the same machine

2. Start a client on a different machine
3. Type a message into the client and press enter
Expected result:
1. Client echos back message
2. Server one displays ‘Message received: MESSAGE from SOURCE’ with the message typed into the client
3. Server two doesn’t display anything
Why: We want to be able to run multiple chat servers based off of one machine

Phase 2.1

Testcase 1 - Logoff
Instructions:
1. Start a server and client connection
2. Type #logoff into the client
3. Type any message into the client
Expected result:
1. Message indicates client has logged off
2. Attempts to type the second message will be met with an error

Testcase 2 - Get info about connection
Instructions:
1. Start a server and client connection
2. Type #gethost into the client
3. Type #getport into the client
Expected result:
1. Host server is printed
2. Server port is printed

Testcase 3 - Login
Instructions:
1. Start a server and client connection
2. Type #logoff into the client
3. Type #login into the client
4. Type any message into the client
Expected result:
1. Client tells you you have disconnected
2. Client tells you you have reconnected
3. The message is successfully received by the server

Testcase 4 - Switch servers
Instructions:
1. Start two servers, one with default arguments and one with port 5565, and client connection
2. Type #logoff into the client

3. Type #sethost <second_server> with the name of the second server
4. Type #setport 5565
5. Type #login into the client
6. Type any message into the client

Expected result:

1. Client tells you you have set the port and host
2. Message is received by the second server

Testcase 5 - Trying to switch servers while connected

Instructions:

1. Start a server and client connection
2. Type #sethost HOST
3. Type #setport 5565

Expected result:

1. Client tells you you cannot set host while connected
2. Client tells you you cannot set port while connected

Testcase 6 - Server trying to switch ports while active

Instructions:

1. Start a server
2. Type #setport 5565

Expected result:

1. Server tells you you cannot set port while active

Testcase 7 - Start server while active

Instructions:

1. Start a server
2. Type #start

Expected result:

1. Server tells you you cannot start while active

Testcase 8 - Stop server while inactive #stop

Instructions:

1. Start a server
2. Type #stop
3. Type #stop

Expected result:

1. Server stops
2. Server tells you you cannot stop while already inactive

Testcase 9 - Stop server while inactive #close

Instructions:

1. Repeat testcase 8 with #close

Expected result:

1. Server stops

2. Server tells you you cannot stop while already inactive

Testcase 10 - Status messages
Instructions:
1. Start a server and client connection
2. Type #logoff in the client
3. Type #login in the client
Expected result:
1. Server displays messages in the server console indicating client disconnected and reconnected

Phase 2.2

Testcase 1 - Client block
Instructions:
1. Start a server and two client connections
2. Type #block <user_id> with the other user's id into one client
3. In the other client, type a message
Expected result:
1. Message is displayed for the server and the other client
2. No message is displayed for the first client

Testcase 2 - Client unblock
Instructions:
1. Start a server and two client connections
2. Type #block <user_id> with the other user's id into one client
3. Type #unblock <user_id> with the other user's id into that client
4. In the other client, type a message
Expected result:
1. Message is displayed in all three consoles

Testcase 3 - Server block
Instructions:
1. Start a server and client connection
2. Type #block <user_id> with the client's id into the server
3. In the client, type a message
Expected result:
1. Message is displayed for the client
2. No message is displayed for the server

Testcase 4 - Server unblock
Instructions:
1. Start a server and client connection
2. Type #block <user_id> with the client's id into the server

3. Type #unblock <user_id> with the client's id into the server
4. In the client, type a message
Expected result:
1. Message is displayed in both consoles

Testcase 5 - Client check who is blocked empty
Instructions:
1. Start a server and client connection
2. Type #whoiblock into the client console
Expected result:
1. Client indicates that no users are blocked by the client

Testcase 6 - Server check who is blocked empty
Instructions:
1. Start a server
2. Type #whoiblock into the server console
Expected result:
1. Server indicates that no users are blocked

Testcase 7 - Client check who is blocking empty
Instructions:
1. Start a server and client connection
2. Type #whoblocksme into the client console
Expected result:
1. Client indicates that no users are blocking the client

Testcase 8 - Server check who is blocking empty
Instructions:
1. Start a server
2. Type #whoblocksme into the server console
Expected result:
1. Server indicates that no users are blocking the server

Testcase 9 - Server check who is blocked
Instructions:
1. Start a server and client connection
2. Type #block <client> with the client id into the server console
2. Type #whoiblock into the server console
Expected result:
1. Server indicates that the client is blocked

Testcase 10 - Server check who is blocking
Instructions:
1. Start a server and client connection
2. Type #block <client> with the client id into the server console

2. Type #whoblocksme into the client console
Expected result:
1. Client indicates that the server is blocking the client