**ChatBot Report**

**ChatBot Instructions to run program**

The chatbot program should be run in terminal, with two windows one for the CBLinkSender and one for the CBLinkReceiver. The CBLinkReceiver program will ask for user input for the error rate in type integer (0,50,60,75,90). CBLinkReceiver will also be asked if they would like to have the trace feature on or off, and the user should enter either “on” or “off”.

The CBLinkSender program will ask for the user’s name ID (3 character name), and trace to be on or off, and finally it will ask the user to enter their message (40 character message).

\*Both the programs are copied versions of the provided Link files – just renamed to keep organized in IDE\*

The program files are located in the src folder and test runs are in the test runs folder.

**Description of chat rules**

The chat rules are maintained by the rules engine, which will determine how to respond to a client’s message. It was written in a series of if, else statements to provide the response.

First it will check the message for “bye”, in order to send its “bye” response and end the program. If the message contains either “hello” or “hi”, the response will greet “hello” back. If the sender asks “what’s up?” or “how are you?” the chat bot will respond with “I'm great thanks! hbu? :)”. If the sender sends a message containing a heart “<3” the response will be “I <3 you”. If the senders message contains a smiley face “: )” the reply will be another smiley face “ :D”. When the sender asks a question using “?” or uses “!” the chat bot will respond with “I don’t know” or “that’s cool” respectively. And lastly, as a default/catch-all for any other types of messages that don’t fall under the previous, the chat bot will respond with “I don’t understand”.

**Analysis on statistics**

When the program terminates, the statistics will be calculated and printed.

During my test runs, I can see that with 0% error rate no messages are damaged and the theoretical transmission is 10 transmissions for 10 messages. When running the 50% error rate on 10 messages 41 messages were damaged, and the theoretical transmission was 20 transmissions. For the 60% error rate, on 10 messages 35 messages were damaged and the theoretical transmission was 25 transmissions. The 75% error rate on 10 messages had 35 damaged messages and a theoretical transmission of 40 transmissions. For the 90% error rate on 10 messages, 28 were damaged and the theoretical transmission was 100 transmissions.

Since I only ran the tests with 10 messages, the theoretical transmission statistics are likely lower than I would have expected from their expected values.

**What I learned from the project**

In this project I learned how to use and manipulate byte arrays, with both strings, integers. Although it’s similar to an array of Strings or Integers, it was still a little unfamiliar for example copying to add two byte arrays together. Additionally I learned how to create errors in the byte array by flipping the values.

**Obstacles encountered**

* + I first was confused about creating the errors in the messages, but after it was discussed in class everything mostly worked out.
  + I had issues setting the error rate from the client program. I couldn’t figure out a way to fix this issue so instead I am getting the user input from the server side.
  + Lastly, I had issues with the requirement where the maximum number of retransmissions of each chat message is 10. So after 10 retransmissions, the message should transmit correctly. I was able to get my default reply to happen for any message retransmitted 10 times, but had difficulty getting the rules engine to be used – since was making the errors on the receivingBuffer and therefore unable to pull a corrected message from it. To fix this I created another buffer retransmitBuffer to hold the message retransmitting-

**Comments/ suggestions**

* Found this program interesting to work with. My nested do while loops might have made things more complicated than needed, so I’d like to see a solutions/ways to do this program.
* I interviewed for an internship a few weeks ago, and they showed me a similar program and asked for me to find the bug. So it was definitely helpful that I’d seen something similar!
* It was also very helpful that we were given the crc, LinkSender, LinkReceiver classes.

**Known bugs**

* I have not used any exception handling, so if some things are entered incorrectly the program will crash. For example entering a non-integer for the error rate will crash the program.
* If the message length is too long, the program will crash. Additionally if I just hit enter without typing in a message, the program will crash, due to no bytes being put into the array to send, so index out of bounds.
* If the client ID is too long, however only the first three chars of the input will be used, the rest is cut off.
* Lastly sort of a bug – since I used: contains(“hi”) , any words that have the letters hi, will respond with hello. “This” = hello..