I created A TCP and Port struct that works the same as socket\_t struct. CMDTestServer() finds a free port and changes the port state to listening, the node can now receive data from a client on this port. CMDTestClient() will then find a free port on the client and change the state to AWAITING, which means the node is trying to establish a connection, the client then sends out its connection request with a random initial sequence number and port to the server. The server receives the initial sequence number and sends back its own SYN with an ACK and the server’s own initial sequence number. The client receives this packet and checks the sequence number, if the expected is the same as the next sequence number, the connection can then be established. Once the server knows the connection is established, it checks how much buffer space it has available and sends this ADWindow to the client with Flag = sendServerData, the client receives this flag and knows it can start sending data up to ADWindow. The client sends this data with flag clientSentData and the server receives this packet. The server then checks if the rest of the data can be written into the buffer or not, it then sends back a packet asking client to send next bytes of data according to buffer size available. This continues until all data is sent. CMDCloseClient then activates connection termination on client and sends out FIN packet to server to initiate termination. Each node terminates its connection and changes port state from ESTABLISHED TO AVAILABLE.