In part A, I made a class structure to hold all the relevant information needed to find the ports, IPs, sequence numbers, ack numbers, and window sizes. Since this was a class, I was easily able to extend it to fit my needs for part b. First I made a global array that holds all the unique IP and port combinations. This is what I consider each flow. I check if a new IP packet is a unique flow by creating a new FlowInterpretor object. If they aren't in the flows array, then I would print and then add it to the array. If the flow is in the flows array, then I would check if it's a SYN/ACK, the first ACK after SYN/ACK or anything else. I do not use the first SYN/ACK or first ACK for calculating the throughput. I also don't use it to find the first 2 transactions. If the FlowInterpretor holds a flag that's not SYN/ACK and it isn't the first ACK after SYN/ACK, then I'd add it to the list that holds either sender or receiver transactions. For the sender, I check if the ports/IPs are in the same positions as the one in the flows array. If it's not, then it's the receiver. I also made a variable in the FlowInterpretor class to sum up the throughput. This adds from the first ACK after SYN/ACK to the FIN/ACK or FIN exclusive. Then we hold the timestamp from the PSH/ACK to the timestamp of the flag right before FIN/ACK or FIN. We divide the throughput variable by the difference of those timestamps. This is what's printed. For the congestion window, I save the previous timestamp of the one being checked right now and subtract it from the current timestamp. This will be saved into the packetTime variable. Then we increment the packetCounter variable. This will be the congestion size. We keep doing this until the packetTime variable is > the RTT, which is calculated by finding the timestamp from the SYN/ACK to the first ACK. We do this up to 3 times per flow. This is what we print out.