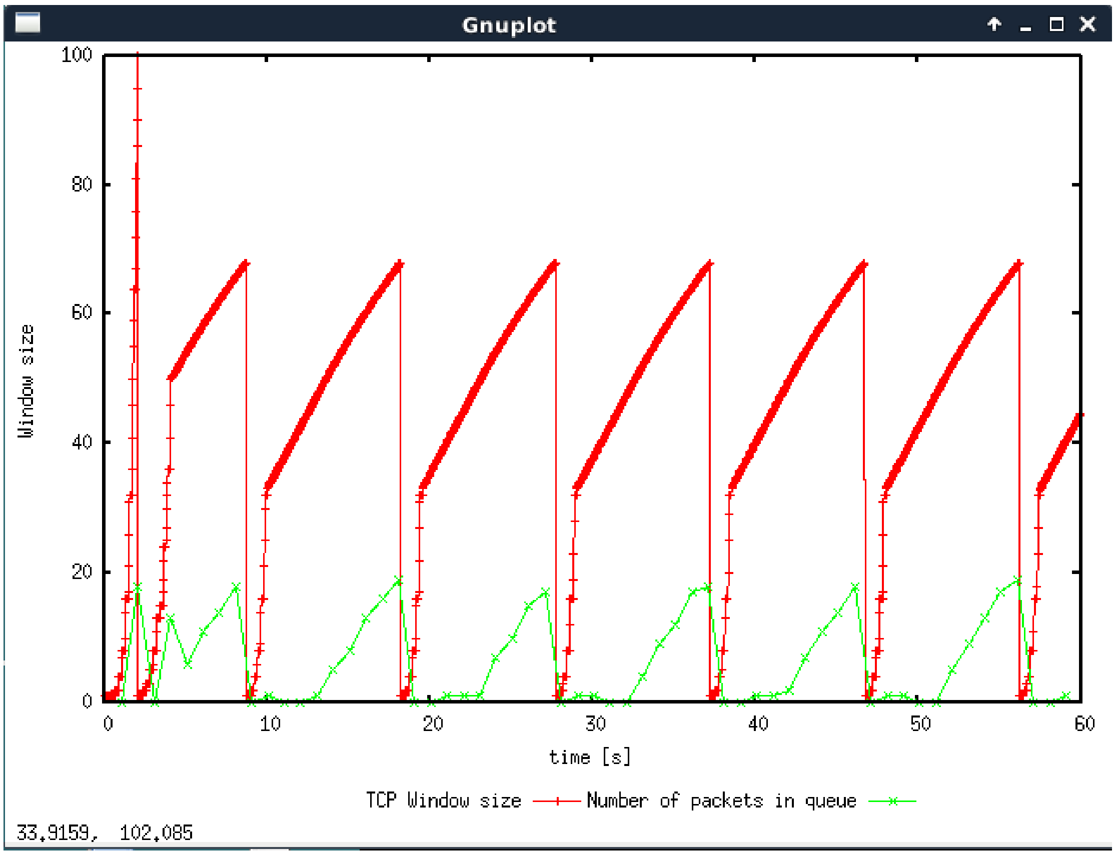
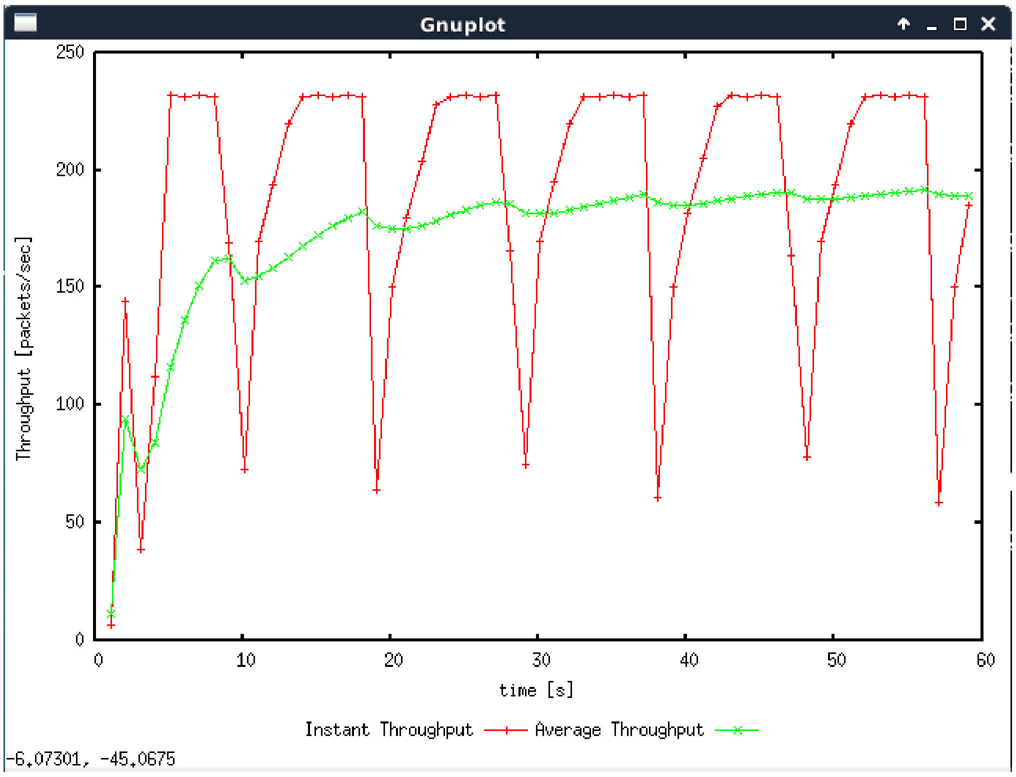
Exercise 1

Question 1



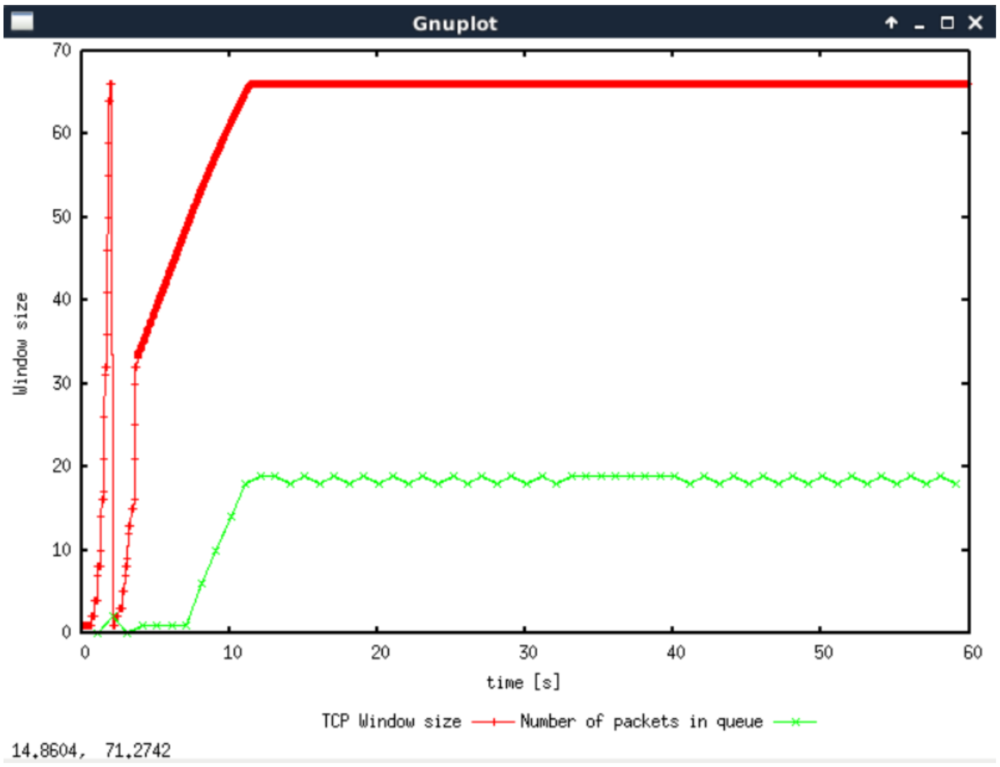
In this graph, the maximum size of the congestion window is 100 MSS, but the threshold is 150 MSS during the whole process. When the cwnd reaches this value, sender will stop increasing the congestion window size and set the ssthresh equal to half of the cwnd and then change the value of cwnd to 1. After that, a new slow start phase will be performed. If cwnd increases to ssthresh again, congestion avoidance will run until the queue becomes full again, which creates packet loss. Finally, the TCP connection will be back to slow start phase and repeat this process.

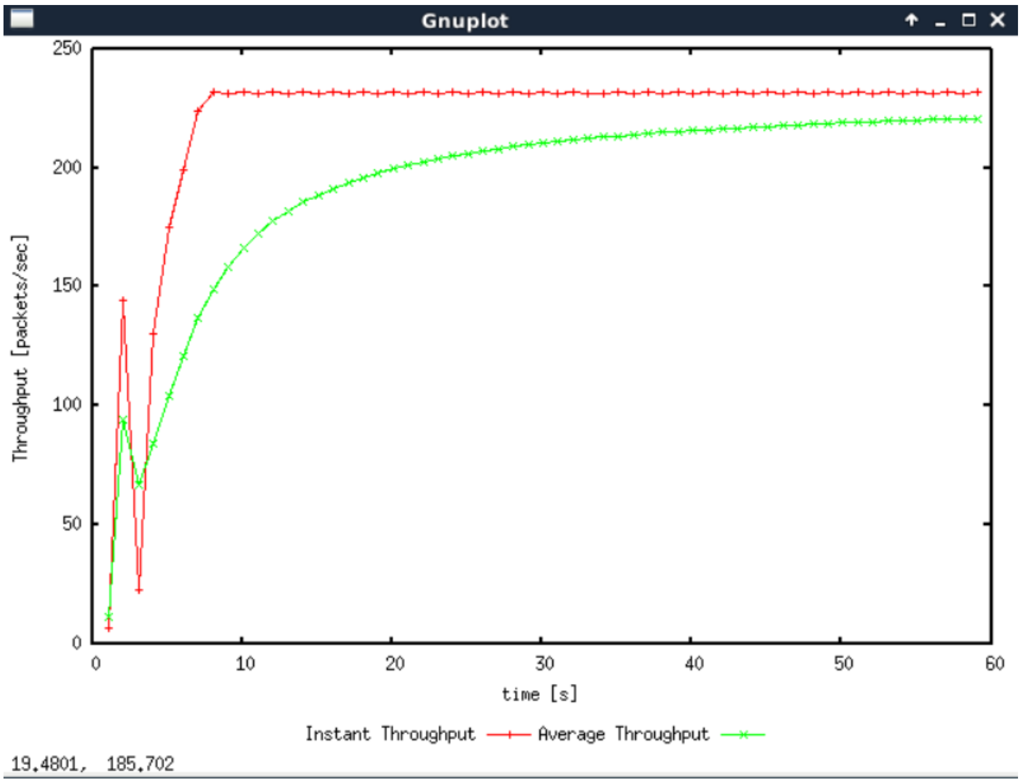
Question 2



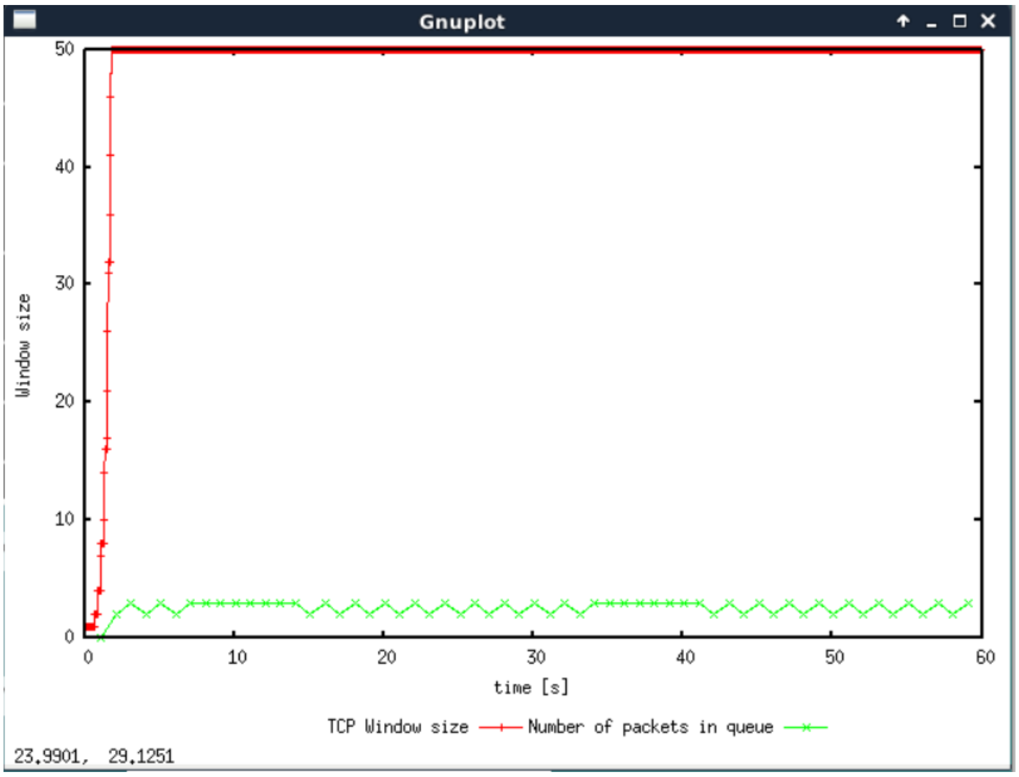
Due to the sable trend of average throughput after 20th second, the average throughput of TCP can be viewed as about 190 packets per second. According to the question, we can get the size of packet is (500+20+20)*8=4320 bits, therefore the throughput is 190 *4320=820.8kbps.

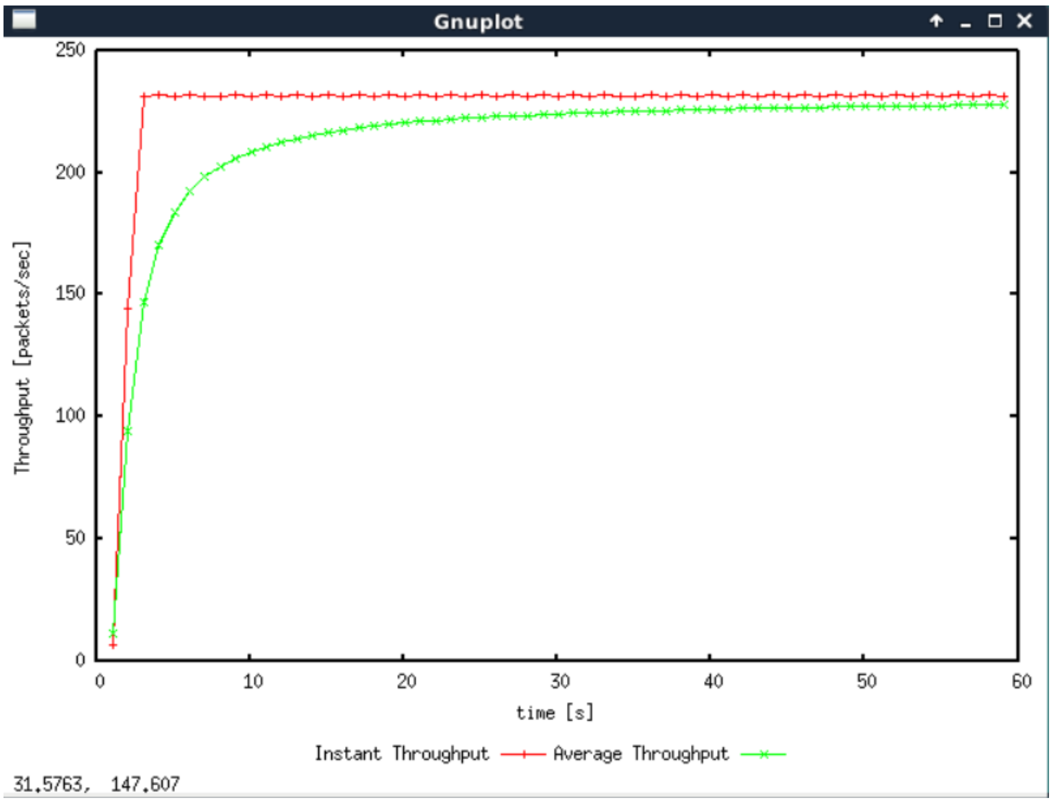
Question 3





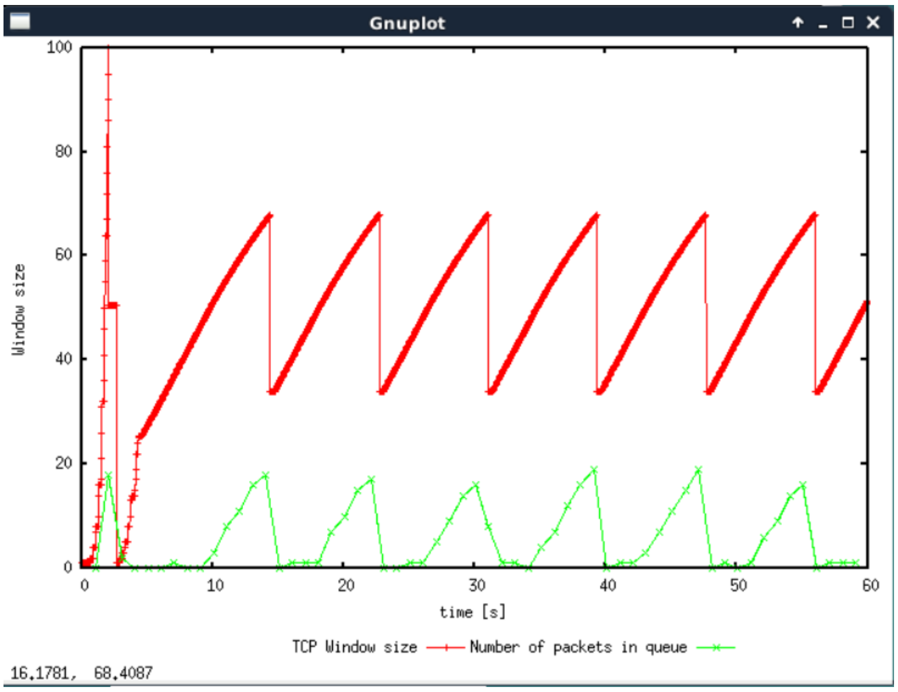
The value of the maximum congestion window at which TCP stops oscillating is 66. After the first slow start phase, it gets a balance situation when cwnd change to half of itself, which means that the queue never gets full and packets never are lost. However, there is also another situation. If cwnd=50, we can see:

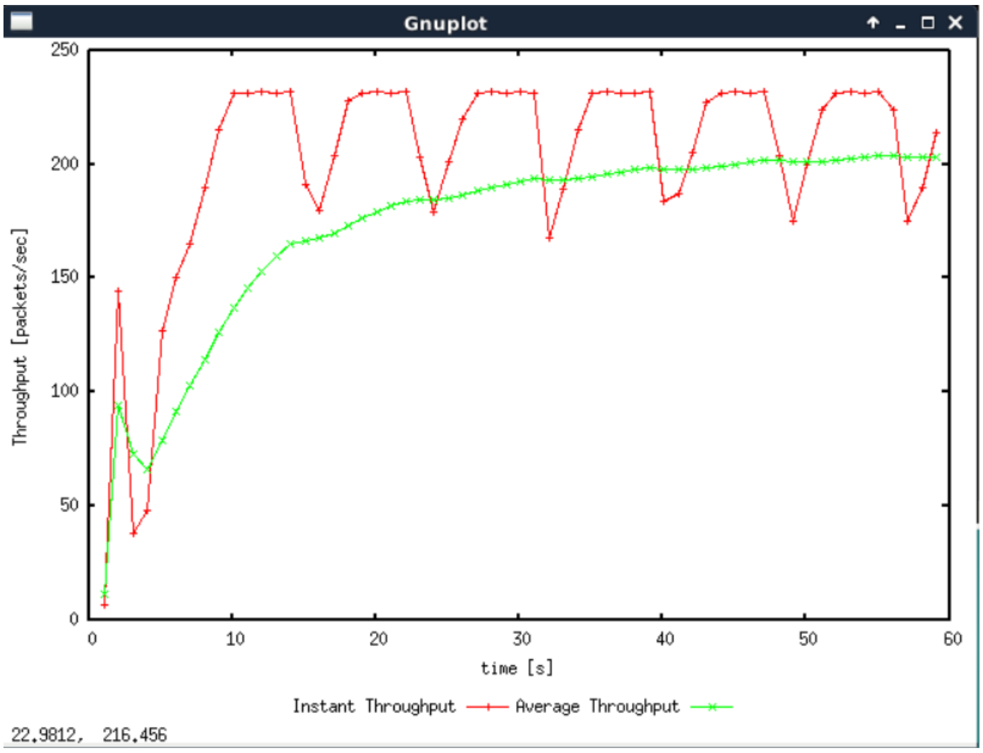




TCP gets a balance situation after the first slow start phase, which stops oscillating. Therefore, if the average throughput of TCP can be viewed as about 220 packets per second. the throughput is:220*4320=950.4kbps.
The actual average throughput compare to the link capacity (1Mbps), it is very close.

Question 4

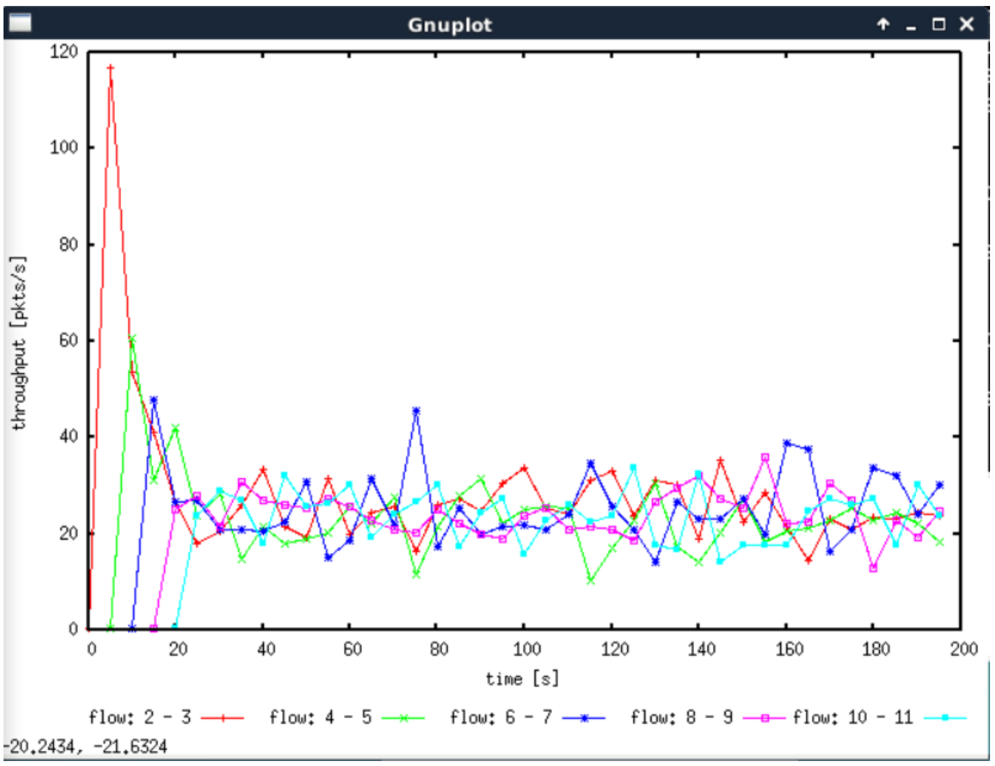




TCP Tahoe set cwnd to 1 MASS no matter triple duplicate ACK or timeout and then begin slow start phase, however, TCP Reno set cwnd to 1 when timeout happened and make it equal to half of itself(or 1/2cwnd+3) on triple duplicate ACK, and then begin fast recovery. The average throughput of TCP Reno can be viewed as about 200 packets per second and the roughput is 200 *4320=864kbps, which is higher than TCP Tahoe. This indicates differences between slow start phase and fast recovery.

Exercise 2

Question 1



Yes, each flow gets an equal share of the capacity of the common link. As the above graph, when a new flow connect to the common link, the throughput of last flow will decrease to an average level with others.

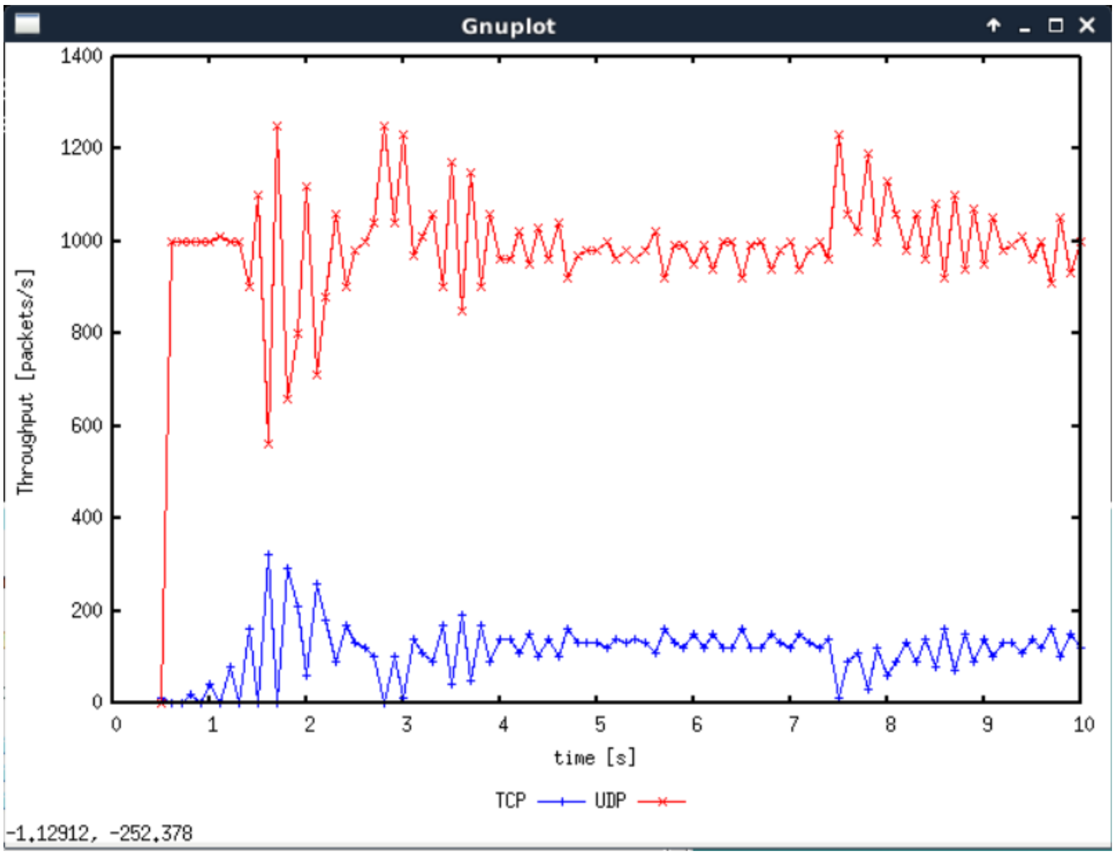
Question 2

The throughput of the pre-existing TCP flows decreases when a new flow is created because it shares the link with the new flow. The mechanisms of TCP which contribute to this behavior is congestion control, which indicates that every flow should have the same condition. Therefore, it is a fair behavior.

Exercise 3

Question 1

I think UDP should have higher throughput compared with TCP. This is because that there is no any congestion control mechanism of UDP.



Question 2

The reason why UDP flow achieves higher throughput than TCP is that the congestion control mechanism may lead to a lower transmission rate, which is also the reason why the DNS query tends to use UDP rather than TCP because UDP doesn't have congestion control.

Question 3

First, one of the attractive features of UDP is that since it does not need to retransmit lost packets nor does it do any connection setup, sending data incurs less delay. This lower delay makes UDP an appealing choice for delay-sensitive applications like audio and video. Second, in some cases, it may increase the transmission rate, because UDP doesn’t have network congestion control, which indicates senders can send packets by a maximum rate. However, if everyone chooses UDP instead of TCP, the network connection will be paralysis such as packet loss and large delay because of the unlimited transmission rate.