Ans1.

a) The given window size is 4, and the receiver is expecting the k^{th} packet. The last 4 acknowledgments would be numbered (k-1), (k-2), (k-3), (k-4). Now the sender may have received all of these acknowledgments, none of these acknowledgments, or just some of these acknowledgments.

Senders window if all of them are received:	k, k+1, k+2, k+3
If none of them received:	k-4, k-3, k-2, k-1
If k-4 received:	k-3, k-2, k-1, k
If k-4, k-3 received:	k-2, k-1, k, k+1
If k-4, k-3, k-2 received:	k-1, k, k+1, k+2

b) Now, if the receiver is waiting to receive the k^{th} packet. That means it has received and sent acknowledgments for the last 4 packets, *i.e.* k-4, k-3, k-2, k-1. Now the possible values of ACK fields depend upon how many ACKs has the sender received. Given the receiver is waiting for the k^{th} packet and a window size of 4, the range of ACKs currently propagating back to the sender can be in the range [k-4, k-1].

Ans2.

It can be implied from figure 2 that Selective Repeat is used to communicate between the sender and the receiver. We can imply that by paying attention to pkt2, which was lost in transmission, but even though receiver received pkt3 instead of pkt2, it sent back an ACK for pkt3 implying the protocol used can receive data in non-chronological order and out of Go-Back-N and Selective Repeat, only Selective Repeat is capable of that.

a) Sender: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
Receiver: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

b) After time t, the sender will start to receive acknowledgments, here are the possibilities that might occur:
All acknowledgments are lost, in which case the sender will resend all the data after a timeout.
- One of the acknowledgments(either ack0, ack1 or ack3) is lost, in which case the sender will resend the data after a timeout.
- Two of the acknowledgments are lost, in which case the sender will resend the two packets after a timeout.
- In any case, sender will have to resend pkt2 because the receiver will never the send out an ack for pkt2.

Ans3.

a) The first segment will have 20 bytes of data. In TCP sequence number denotes the first byte of the segment, which is 90 in our given case, the last byte of the segment is 109 hence the segment has 20 bytes of data.

b) TCP when sending out acknowledgments sends out the number of the segment it expects next, And TCP uses cumulative acknowledgments. So, the acknowledgment number sent out would be 90.

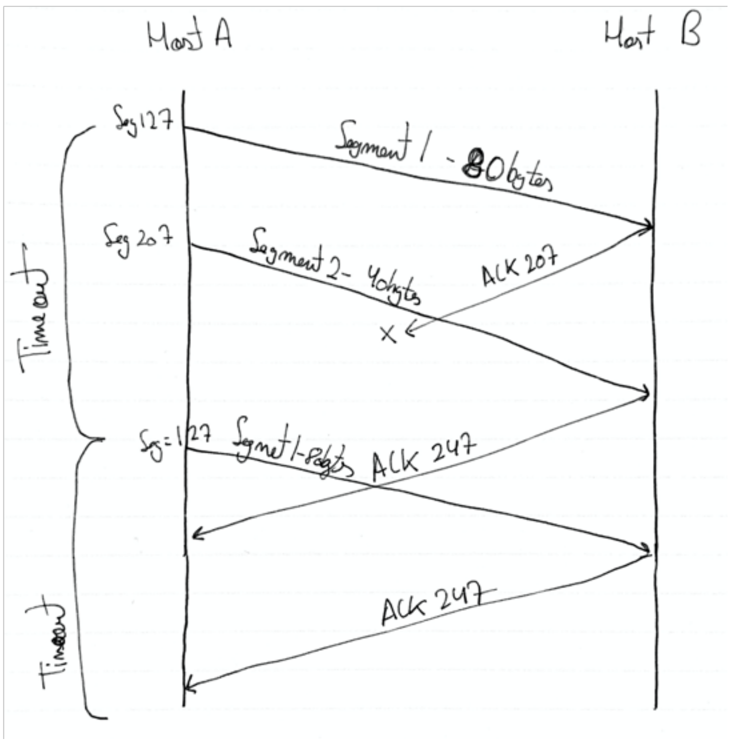
Ans4.

a) In the second segment sent from Host A, the sequence number would be 126 + 80 + 1 = 207. The source port number would be = 302, a destination port number would be = 80.

b) If the first segment arrives before the second segment, the ACK number would be 207. The port number would be 302 and the destination port number would be 80.

c) If the second segment arrives before the first segment the ACK number would be 127, indicating that the receiver has not yet received all the data following the 127 bytes.

d)



When ACK207 is lost, and the sender times out it resends segment 1 then. Meanwhile, the receiver receives the segment 2 and sends out ACK247. The receiver receives segment 1 again and sends out ACK247 once again signaling it has received all the segments before 247.

Ans5.

a) Intervals [1,6] and [23,26] is where TCP slow start starts operating.

b) Intervals [6,16] and [17,22] is where TCP congestion avoidance starts operating.

c) Yes, the packet loss is detected by a triple duplicate ACK after the 16th transmission, that's why the congestion window size dropped to 24. If a timeout had occurred the congestion window size would've dropped to 1.

d) The segment loss is detected by a dropout because the congestion window dropped to 1.

e) The initial ssthresh value **is 32** because after 33 the slow start stops and congestion avoidance kicks in.

f) The sstresh value at the 18^{th} transmission round **is 21**, because the threshold is = half the value of the congestion window when packet loss is detected, and packet loss was detected at window size = 42.

g) Again, the threshold is = half the value of the congestion window when packet loss is detected. Packet loss, in this case, is detected at 26 hence the threshold **is 13.**

h) Packet 1 is sent in 1^{st} round, Packets 2-3 are sent in the second round, Packet 4-7 are sent in the third round, Packet 7-16 are sent in 4^{th} round, packet 16-32 are sent in the 5^{th} round, Packet 32-63 are sent in 6^{th} round, packet 63-95th are sent in 7^{th} round. **So 70^{th} Packet is sent in the 7^{th} round.**

i) The window size will be set to half of the size at what packet loss occurred, which is 8 in this case. Hence, the window size will be set to 8.

j) The sshtresh will be 21. The congestion window will be reset to 1 and will grow to 4 in the 18^{th} round.

k) Packets sent on 17^{th} = 1
18^{th} = 2
19^{th} = 4
20^{th} = 8
21^{st} =16
22^{nd} = 32
Total = 63
We get 63 if we assume TCP will check against the ssthreash after the transmission round.
If TCP checks against the ssthreash before the transmission round, then on the 22^{nd} transmission round we'd have **21 instead of 32 and the total would be 51**

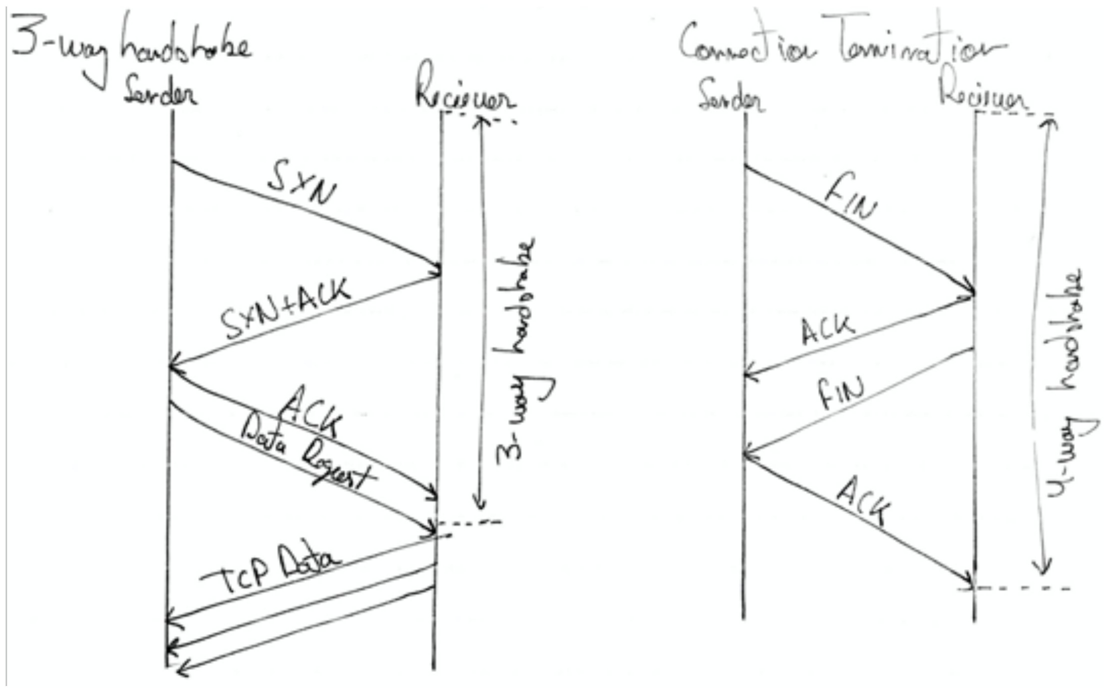
Ans6.

a) The maximum amount of data that can be transferred without wrapping over = 2^32 bytes
The size of MSS is not required in this case because the sequence number doesn't increase in reference to the MSS but rather increases depending upon how much data is being sent. So, in this case, the amount of data that can be sent **is 2^32 bytes, that is 4.29Gigabytes**

b) Given L = 2^32 bytes
66 bytes of the header is added to each segment, which gives us (2^32) * 66 / 536 = 528857914 bytes.

Hence, total data transmitted = 2^32 + 528857914 = 4.824 * 10^9 bytes. Time to transmit over a 155Mbps link will equal to
= (4.824 * 10^9) / ((155*1000*1000)/8)
= 249 Seconds

Ans7.



Ans8.

The next acknowledgment number will be 81920 + 10240 = 92160 and the next sequence number it will receive is 65530.

Ans9.

Given CWND is smaller than RWND, therefore congestion window size = 3000. 2000 bytes have been sent and haven't been acknowledged yet **so 3000-2000 = 1000 more bytes** can be sent before the congestion window runs out.

Ans10.

Transmission	Congestion Window	Threshold	Receive Window	Operation
1	4kb	12kb	32kb	Slow Start
2	8kb	12kb	32kb	Slow Start
3	16kb	12kb	32kb	Slow Start
4	20kb	12kb	32kb	Additive Increase
5	24kb	12kb	32kb	Additive Increase
6	28kb	12kb	32kb	Additive Increase
7	**32kb**	**12kb**	**32kb**	**Timeout**
8	4kb	16kb	32kb	Slow Start
9	8kb	16kb	32kb	Slow Start
10	16kb	16kb	32kb	Slow Start
11	20kb	16kb	32kb	Additive Increase
12	24kb	16kb	32kb	Additive Increase
13	28kb	16kb	32kb	Additive Increase
14	32kb	16kb	32kb	cwnd = rwnd
15	**32kb**	**16kb**	**32kb**	**Triple Duplicate**
16	16kb	16kb	32kb	Additive Increase
17	20kb	16kb	32kb	Additive Increase
18	24kb	16kb	32kb	Additive Increase
19	28kb	16kb	32kb	Additive Increase
20	32kb	16kb	32kb	cwnd = rwnd
21	32kb	16kb	32kb	cwnd = rwnd
22	32kb	16kb	32kb	cwnd = rwnd
23	32kb	16kb	32kb	cwnd = rwnd
24	32kb	16kb	32kb	cwnd = rwnd

So, on the 24^{th} transmission, the sender will attempt to send 32kb of data.

Formulas used in this question are:

cwnd cannot be bigger than rwnd.

When timeout occurs
cnwd = 1MSS
ssthread = cnwd/2

When duplicate ACKs are received
cnwd = cnwd/2
ssthreadd = cnwd/2