Rajat Sethi – ECE 4380 – HW 3

1a.)

The code segment uses approach one (Always send an ACK).

To change the code segment to approach two, an if-statement needs to be added at line 106 to determine if the (seq_no == LFR + 1) or the frame is not in window, in which an ACK will be sent on the next line.

1b.)

Time	Frame Recv.	Action	NFE	Send ACK (Approach 1)	Send ACK (Approach 2)
1	0	Deliver 0 to app	1	0	0
2	1	Deliver 1 to app	2	1	1
3	4	Buffer 4, do not send to app	2	1	
4	5	Buffer 5, do not send to app	2	1	
5	2	Deliver 2 to app	3	2	2
6	3	Deliver 3 to app	6	5	5
7	4	Discard Duplicate 4	6	5	

2a.)

5000 bits / 10,000,000 bps = 0.5 ms

(A $\rightarrow$ B) = 0.5 ms + 0.5 ms (Prop Delay) = 1 ms

(B $\rightarrow$ C) = 0.5 ms + 1 ms (Prop Delay) = 1.5 ms

500 bits / 10,000,000 bps = 0.05 ms (ACK)

(C $\rightarrow$ B (ACK)) = 0.05 ms + 1 ms (Prop Delay) = 1.05 ms

(B $\rightarrow$ A (ACK)) = 0.05 ms + 0.5 ms = 0.55 ms

Total Throughput = 4500 bits / 4.1 ms = 1.098 Mbps

2b.)

Transmission time per packet = 5000 bits / 10 Mbps = 0.5 ms

8.2 packets can go through before first ACK comes back (4.1 ms)

36,500 payload bits (36000 from first 8 packets, 1000 from 9th packet (500 bits are header))

36,500 payload-bits / 4.1 ms = 8.9 Mbps

2c.)

Since 8.2 frames maximizes output, the window needs to round-up to **9 frames.**

3a.)

Prop Delay (A → B) = 2000 miles * 10 mic-seconds/mile = 20 ms

3 frame window = 3000 bits at once.

Tx = 3000 bits / 100,000 bps = 30 ms

3000 bits / 50 ms = **60.0 kbps**

3b.)

60.0 kbps (Same answer as 3a since the input into node B is the lower bound)

3c.)

Prop Delay (B → C) = 500 miles * 10 mic-seconds / mile = 5 ms

Tx for 1 frame = 1000 bits / 100,000 bps = 10 ms

(A → B) = 10 ms + 20 ms = 30 ms

(B → C) = 10 ms + 5 ms = 15 ms

(C → B (ACK)) = 5 ms

(B → A (ACK)) = 20 ms

Total time for 1 ACK = 70 ms

70 ms / 10 ms per frame = **7 Frames**

4a.)

First frame fails, time-out is 4 seconds.

Eight more frames go in one-at-a-time, 4 seconds each (32 seconds total).

Transfer time is **36 seconds.**

4b.)

t=0 to t=4 First four frames are sent (Takes 4 seconds) t=4 By this time, frame 1 fails to receive ACK, so frame 1 is resent.

t=5 ACK Frame 2 comes back

t=6 ACK Frame 3 comes back

t=7 ACK Frame 4 comes back

t=8 ACK Frame 1 comes back, window shifts, Frame 5 is sent

t=9 Frame 6 is sent

t=10 Frame 7 is sent

t=11 Frame 8 is sent

t=12 ACK Frame 5 comes back.

t=13 ACK Frame 6 comes back.

t=14 ACK Frame 7 comes back.

t=15 ACK Frame 8 comes back.

Total Time = **15 seconds.**

4c.)

t=0 Frame 1 gets sent (Ends up Failing)

t=1, Frame 2 gets sent (Thrown out because it isn't Frame 1)

t=2, Frame 3 gets sent (Thrown out because it isn't Frame 1)

t=3, Frame 4 gets sent (Thrown out because it isn't Frame 1)

t=4, Frame 1 does not receive ACK, resends Frame 1

t=5, Frame 2 is resent

t=6, Frame 3 is resent

t=7, Frame 4 is resent

t=8, ACK Frame 1 received, sends Frame 5

t=9, ACK Frame 2 received, sends Frame 6

t=10, ACK Frame 3 received, sends Frame 7

t=11, ACK Frame 4 received, sends Frame 8

t=12, ACK Frame 5 received

t=13, ACK Frame 6 received

t=14, ACK Frame 7 received

t=15, ACK Frame 8 received

Total Time = **15 seconds**

4d.)

t=0, Frame 1 gets sent on Channel 1 (Ends up getting lost)

t=1, Frame 2 gets sent on Channel 2

t=2, Frame 3 gets sent on Channel 3

t=3, Frame 4 gets sent on Channel 4

t=4, Frame 1 does not receive ACK, gets resent on Channel 1

t=5, Frame 2 gets ACK, Frame 5 gets sent on Channel 2

t=6, Frame 3 gets ACK, Frame 6 gets sent on Channel 3

t=7, Frame 4 gets ACK, Frame 7 gets sent on Channel 4

t=8, Frame 1 gets ACK, Frame 8 gets sent on Channel 1

t=9, Frame 5 gets ACK

t=10, Frame 6 gets ACK

t=11, Frame 7 gets ACK

t=12, Frame 8 gets ACK

Total Time = **12 seconds**

5a.)

t=0, 4 packets are sent from A → R. R sends packet 1 to B

t=1, R sends packet 2 to B, B sends ACK for packet 1 to R.

t=2, R/A receives Packet 1's ACK and sends Packet 5 to R. R sends Packet 3 to B. Packet 2 reaches B and sends ACK to R.

t=3, R/A receives Packet 2's ACK and sends Packet 6 to R. R sends Packet 4 to B. Packet 3 reaches B and sends ACK to R.

t=4 R/A receives Packet 3's ACK and sends Packet 7 to R. R sends Packet 5 to B. Packet 4 reaches B and sends ACK to R.

5b.)

Peak queue size requirement is **3 packets** since the max is reached at the beginning.

5c.)

Whenever a packet is sent to B after t=2, the queue goes down to 1 packet. Then the prior packet's ACK gets sent to A, and A sends a new packet, bringing the total back to 2 packets. As such, the largest number of packets is **2 packets.**

6a.)

t=0, Packets 1, 2, and 3 are sent to R, where packet 1 gets immediately transmitted to B and packet 3 is thrown out.

t=1, Packet 1 reaches B and returns an ACK, Packet 2 is sent to B

t=2, Packet 1's ACK reaches R/A. Packet 4 is sent to R and sent to B. Packet 2 reaches B and returns an ACK.

t=3, Packet 2's ACK reaches R/A. Packet 5 is sent to R and sent to B. Packet 4 reaches B and returns an ACK.

t=4, Packet 4's ACK reaches R/A. No packet can be sent because there has been no ACK from Packet 3 yet. Packet 5 reaches B and returns an ACK.

t=5, Packet 5's ACK reaches R/A. No packet can be sent because there has been no ACK from Packet 3 yet.

t=6, Packet 3 timeouts and is retransmitted to B.

t=7, Packet 3 reaches B and sends an ACK.

t=8, Packet 3's ACK reaches R/A, the sliding window shifts and sends Packet 6 and 7 to R. Packet 6 is sent to B.

t=9, Packet 6 reaches B and sends an ACK. Packet 7 is sent to B.

t=10, Packet 6's ACK reaches R/A, Packet 7 reaches B and sends an ACK.

t=11, Packet 7's ACK reaches R/A.

6b.)

t=0, Channel 1 sends Packet 1, Channel 2 sends Packet 2, Channel 3 sends Packet 3. Packet 1 is sent to B, Packet 2 waits in the queue, Packet 3 is thrown out.

t=1, Packet 1 reaches B and sends back an ACK, Packet 2 is sent to B.

t=2, Packet 1's ACK reaches R/A and Channel 1 sends Packet 4, which is sent to B. Packet 2 reaches B and sends back an ACK.

t=3, Packet 2's ACK reaches R/A and Channel 2 sends Packet 5, which is sent to B. Packet 4 reaches B and sends back an ACK.

t=4, Packet 4's ACK reaches R/A and Channel 1 sends Packet 6, which is sent to B. Packet 5 reaches B and sends back an ACK.

t=5, Packet 5's ACK reaches R/A and Channel 2 sends Packet 7, which is sent to B. Packet 6 reaches B and sends back an ACK.

t=6, Packet 6's ACK reaches R/A. Packet 3 timeouts and retransmits on Channel 3, which is sent to B. Packet 7 reaches B and sends back an ACK.

t=7, Packet 7's ACK reaches R/A. Packet 3 reaches B and sends back an ACK.

t=8, Packet 3's ACK reaches R/A.

7a.)

If frame 10 fails to transmit, then only frames 4 places above/below can possibly be sent prior. Since frame 17 is outside of any possible window with RWS=SWS=5 with frame 10, it is impossible for frame 17 to transmit before frame 10.

7b.)

Only **two numbers** are required (0 and 1), which can be represented through 1 bit.

7c.)

Eight numbers are required since the receiver needs to keep track of the entire window in order.