Module 3 Assessment: The Transport Layer

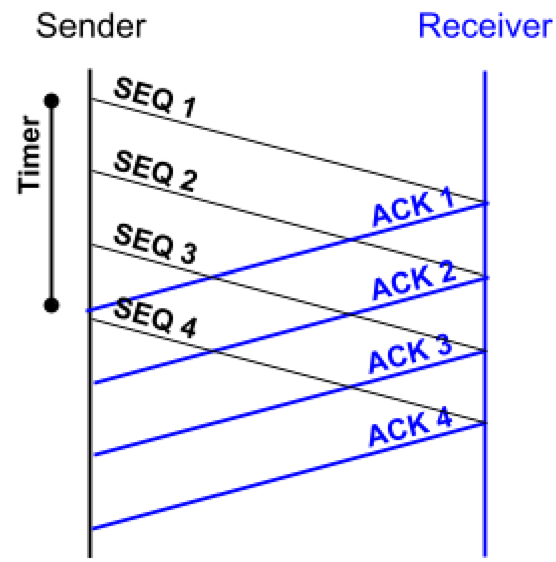
Please keep each question on a single page to make grading simpler. You do not need to use the entire page to answer each question. You may use any resource (except for other people) to help you answer these questions. You may ask LLMs questions to help you understand concepts however the answers you submit should represent your understanding of the material, not merely the output of an AI tool. Questions should be answered within the context of this course's material.

1. Assume a computer is sending the following 3 bytes of data: 89 241 72 over **UDP**. The data is being sent from port 59700 to port 443. What is the checksum that should be included with the packet? Your answer should be 16 bits long. Make sure you include both the packet header and the payload in the checksum calculation. **Show your work**. **(15 points)**

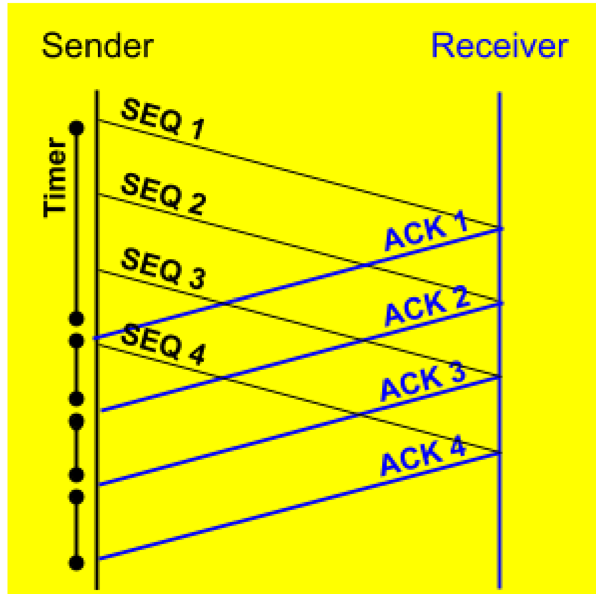
1. Source Port + Destination Port = 11101001 00110100 + 00000001 10111011 = 11101010 11101111
2. Result 1 + Length = 11101010 11101111 + 00000000 00001001 = 11101010 11111000
3. Result 2 + Payload byte 1 = 11101010 11111000 + 01011001 11110001 = 1 01000100 11101001
4. Result 3 + Overflow = 1 01000100 11101001 + 00000000 00000001 = 01000100 11101010
5. Result 4 + Payload byte 2 = 01000100 11101010 + 01001000 00000000 = 10001100 11101010
6. 10001100 11101010 -> **01110011 00010101 (One's complement) = checksum**

2. GBN Protocol (12.5 points)

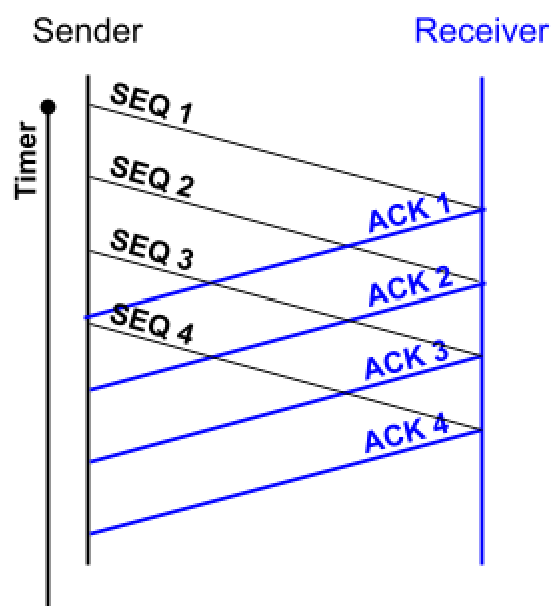
a. Four timing diagrams are shown below. Only **one** represents valid behavior for the **Go-Back-N algorithm**. Highlight the valid timing diagram.



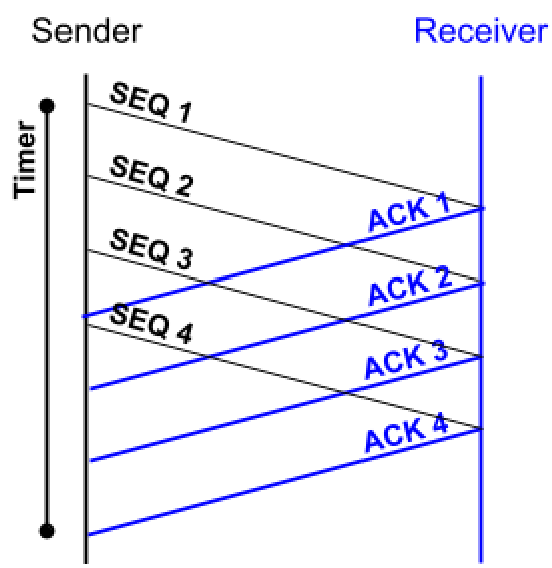
(a)



(b)

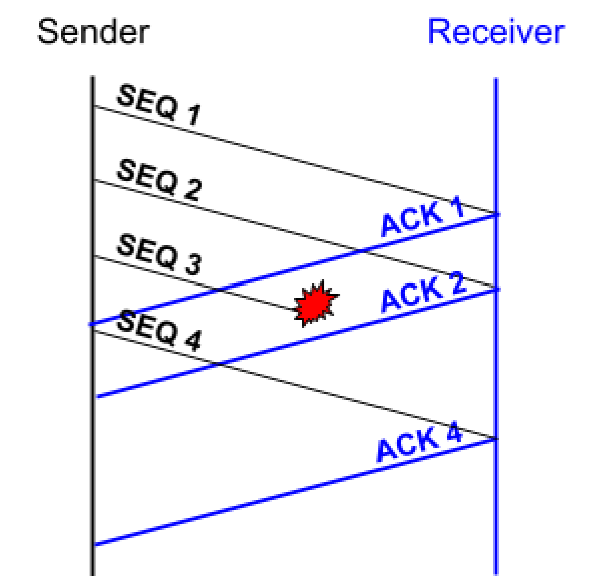


(c)

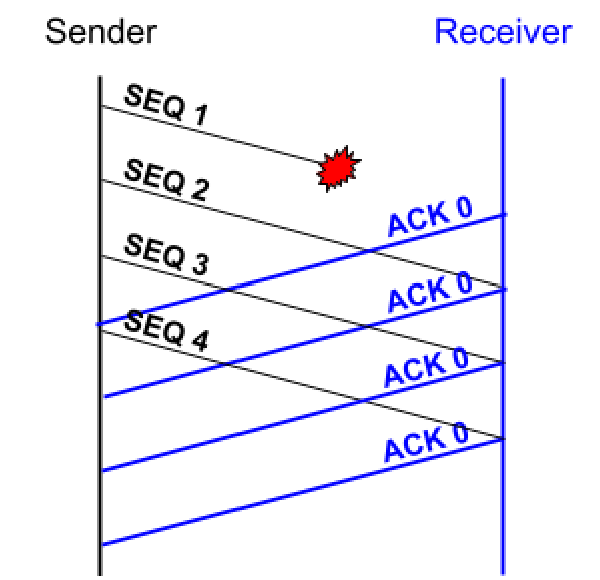


(d)

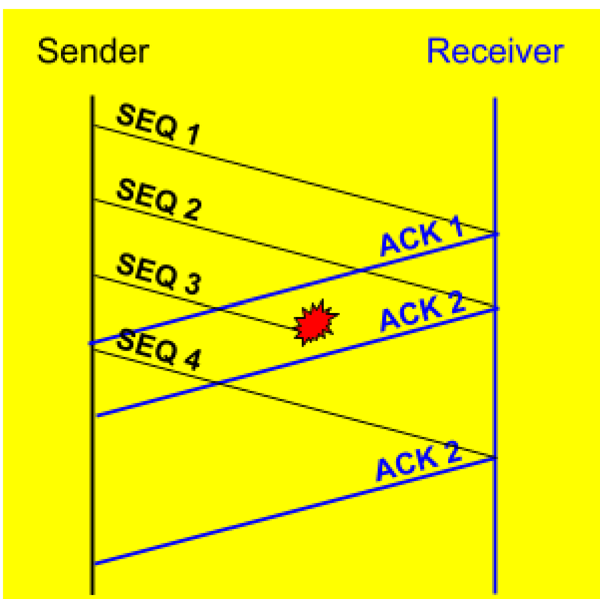
b. Four timing diagrams are shown below. Only **one** represents valid behavior for the **Go-Back-N algorithm**. Highlight the valid timing diagram. **Timers are not shown for these problems; no problems are related to the absence of a timer.**



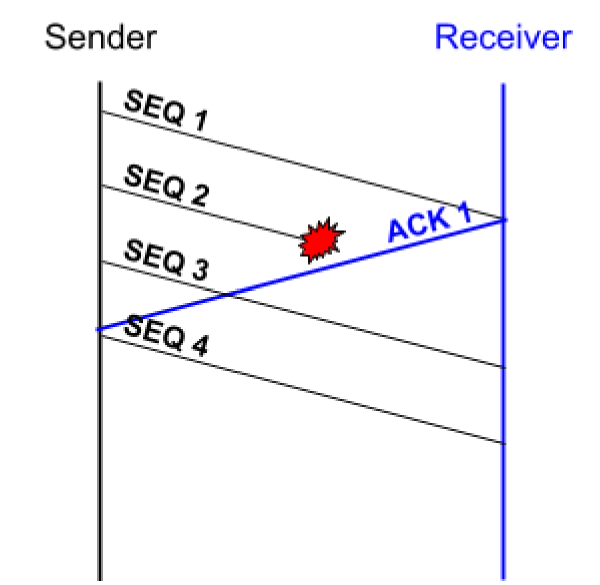
(a)



(b)

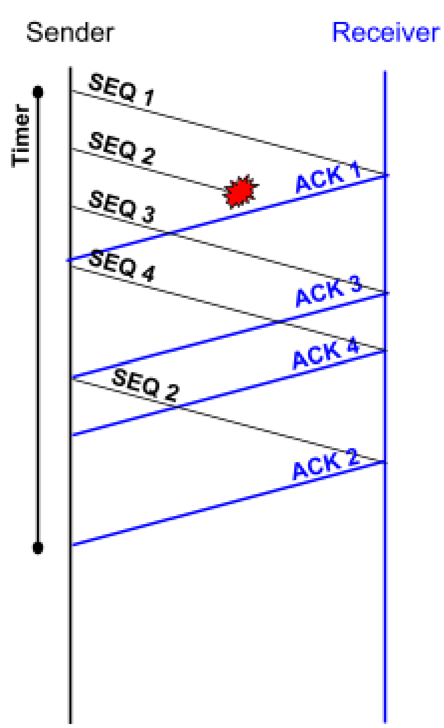


(c)

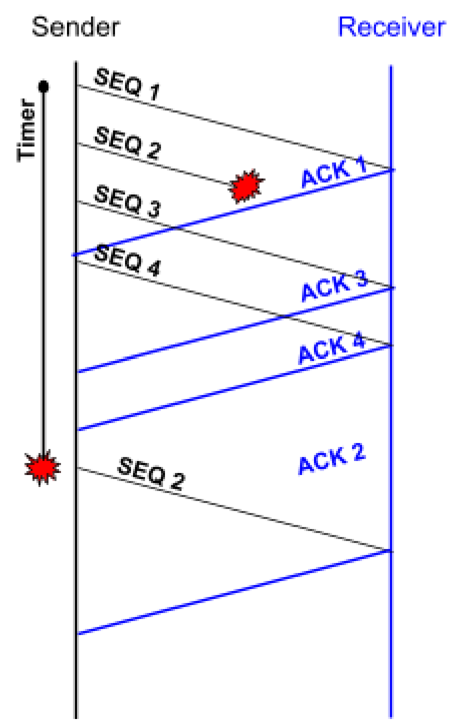


(d)

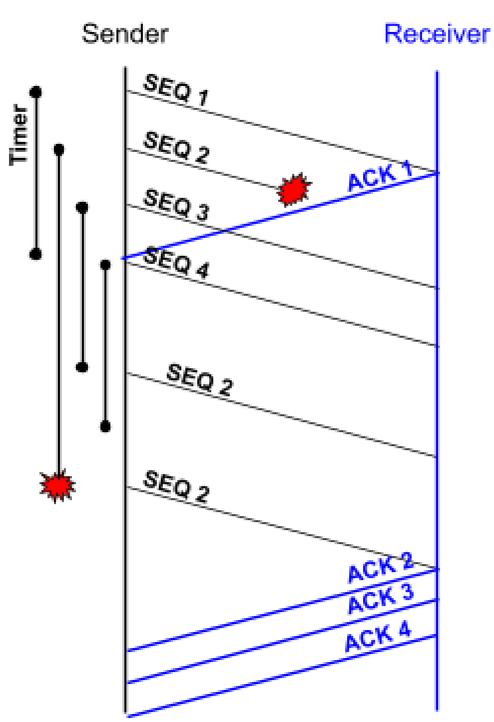
c. Four timing diagrams are shown below. Only **one** represents valid behavior for the **Go-Back-N algorithm**. Highlight the valid timing diagram. **An explosion on the timer means it expired.**



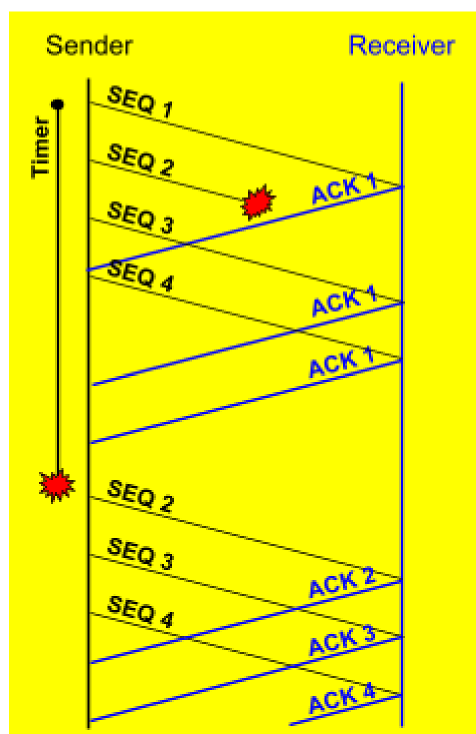
(a)



(b)



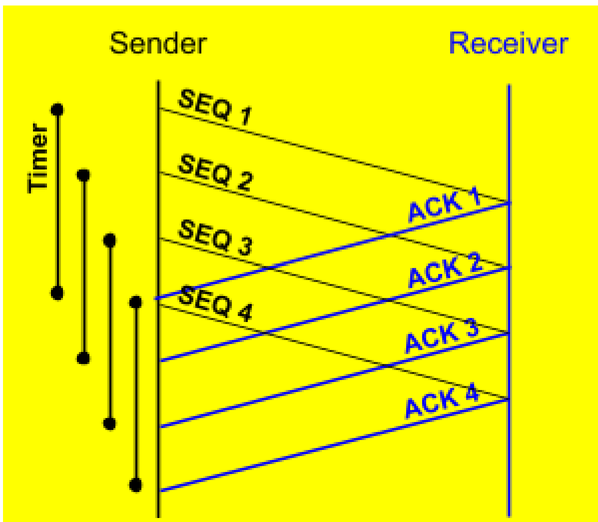
(c)



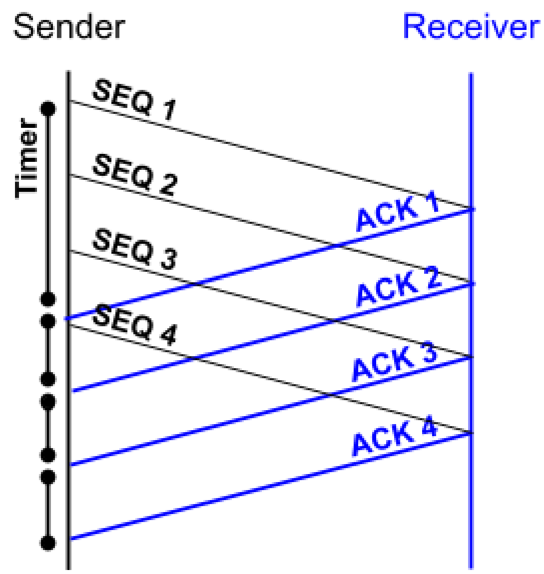
(d)

3. Selective Repeat Protocol (12.5 points)

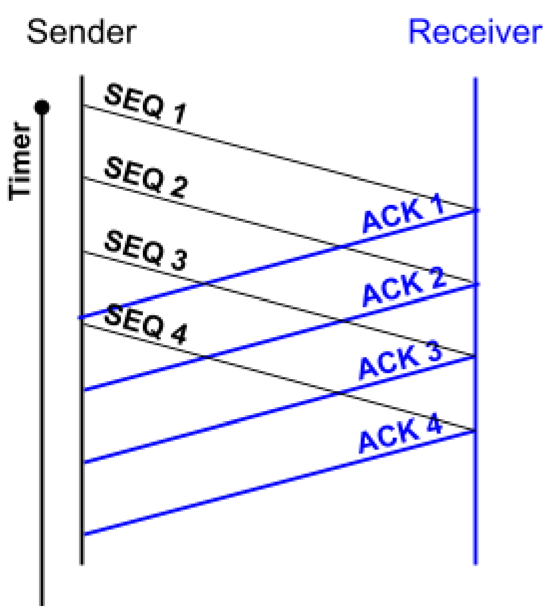
a. Four timing diagrams are shown below. Only **one** represents valid behavior for the **Selective Repeat algorithm**. Highlight the valid timing diagram.



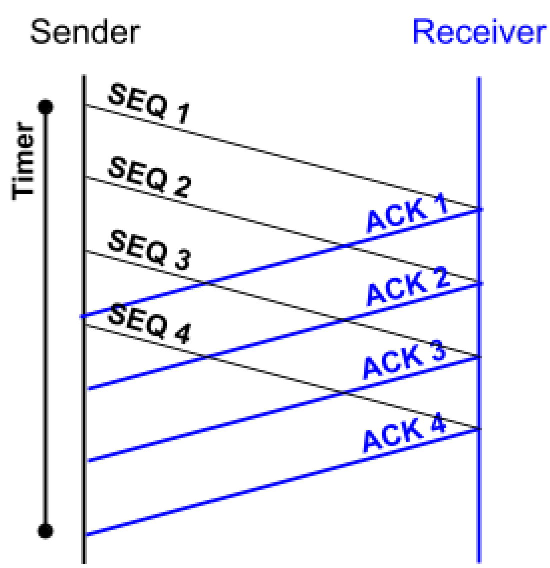
(a)



(b)

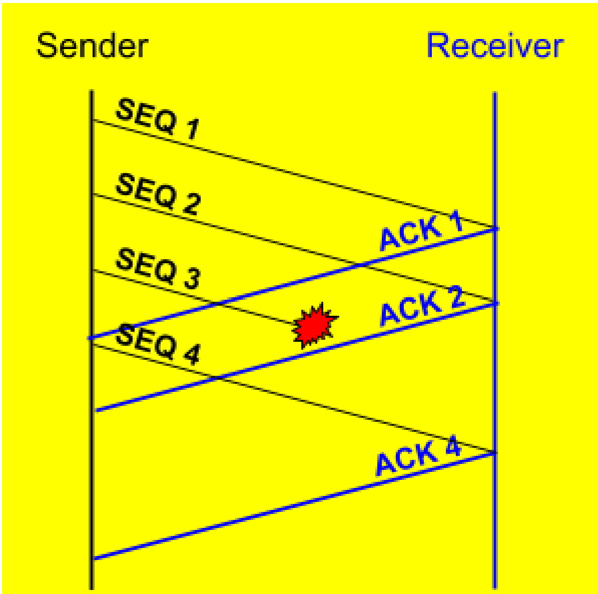


(c)

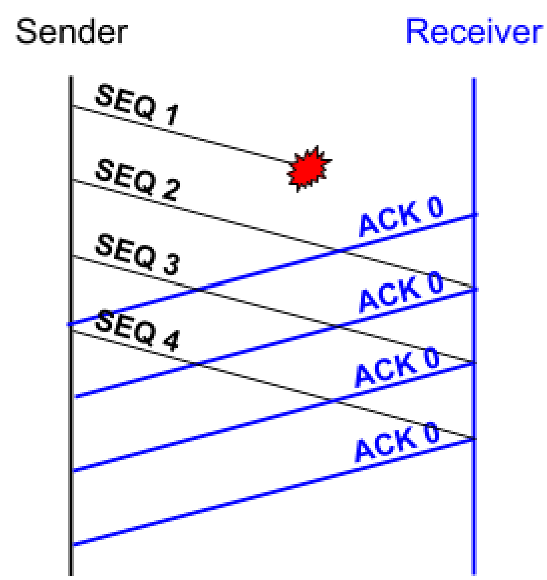


(d)

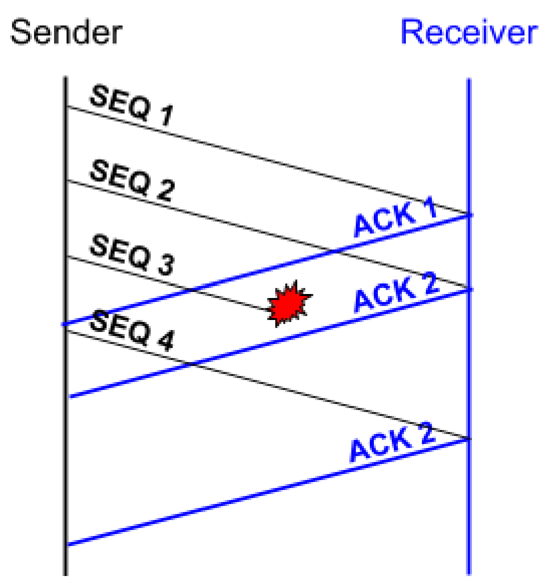
b. Four timing diagrams are shown below. Only **one** represents valid behavior for the **Selective Repeat** algorithm. Highlight the valid timing diagram. **Timers are not shown for these problems; no problems are related to the absence of a timer.**



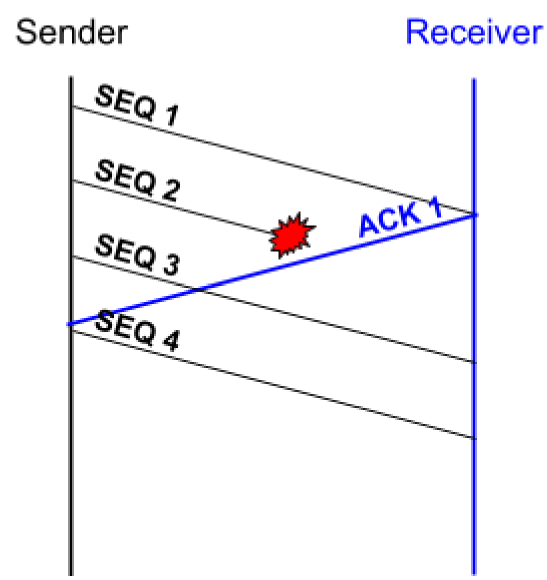
(a)



(b)

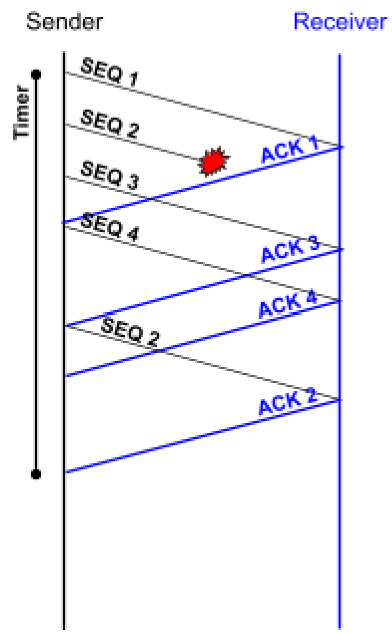


(c)

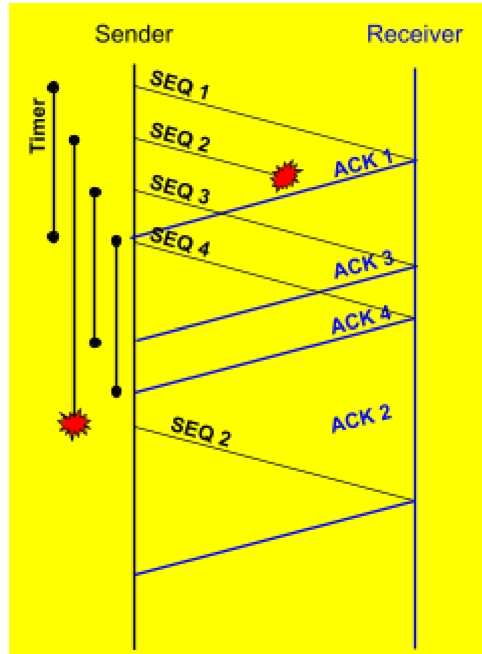


(d)

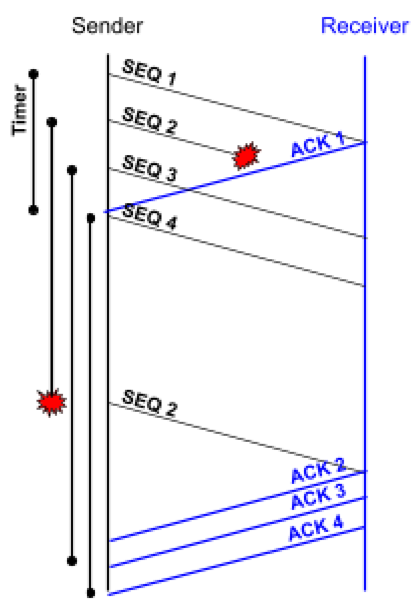
c. Four timing diagrams are shown below. Only **one** represents valid behavior for **Selective Repeat**. Highlight the valid timing diagram. **An explosion on the timer means it expired.**



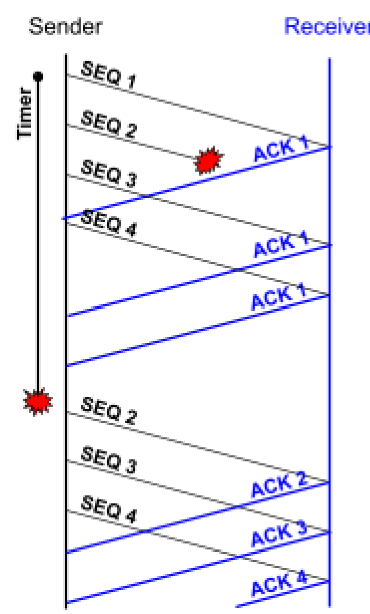
(a)



(b)



(c)



(d)

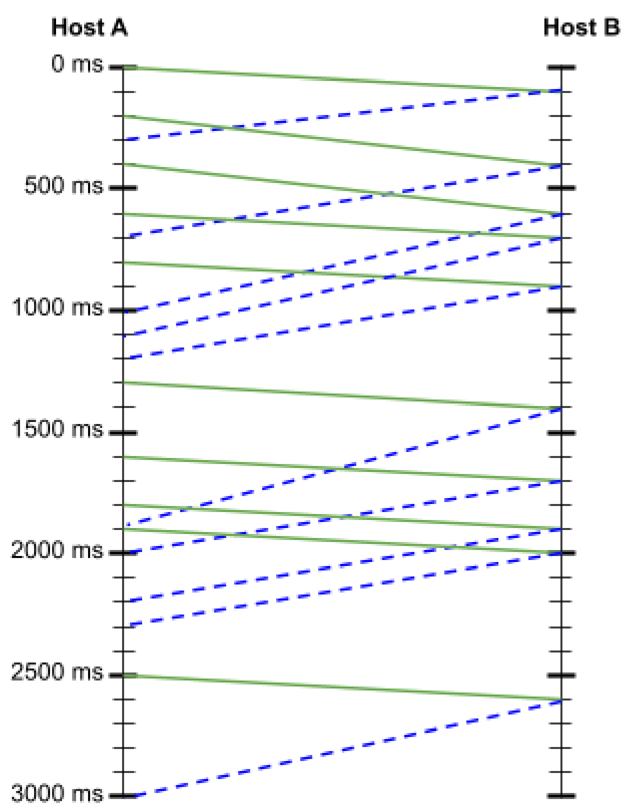
4. TCP estimates its timeout value based on measurements of packets' round trip time (RTT). As RTT increases, the timeout value increases as well. The variability in RTT is also factored into the calculation. If RTT is a consistent value, the timeout will be closer to the measured RTT. If there is a lot of variability in RTT, then then timeout will increase. TCP only tracks the RTT of a single packet at a time. When one ongoing RTT measurement finishes, a new measurement will be started when the next packet is sent. This means that RTT is not measured for every packet sent. These measurements are then combined together using a moving average, so as not to give too much weight to any one reading (which may not be representative of the current state of the network). The below equations are used by TCP to compute its timeout value.

- $\text{EstimatedRTT}_n = (1 - \alpha) * \text{EstimatedRTT}_{n-1} + \alpha * \text{SampleRTT}_n$
- $\text{DevRTT}_n = (1 - \beta) * \text{DevRTT}_{n-1} + \beta * |\,\text{SampleRTT}_n - \text{EstimatedRTT}_n\,|$
- $\text{Timeout}_n = \text{EstimatedRTTn} + 4 * \text{DevRTTn}$

SampleRTT is the most recently measured RTT value. The timeout is updated every time a new SampleRTT is measured. α and β are constants that are used to tune the moving average. **Use values α = 0.1 and β = 0.25 for this problem.** The below ladder diagram shows several packets traveling back and forth between two TCP endpoints. **Your task is to collect each SampleRTT value TCP would measure and then update the timeout value accordingly. Report each value you calculate in the table on the next page.**

Note the following:

- Show your work (use additional pages as needed). To simplify the math, round each calculated value to the nearest integer.
- The highlighted row in the table on the next page contains the state of the timeout, estimatedRTT, and devRTT prior to beginning the problem.
- Assume TCP is ready to begin a new RTT measurement at the start of the problem.
- Enough rows **are** provided to calculate **all** the updates to the timeout value that will occur in the above ladder diagram (there may be a few extra rows as well).



#	Sample RTT (ms)	Estimate RTT (ms)	Dev RTT (ms)	Timeout (ms)
0	___ ms	400 ms	200 ms	1200 ms
1	300	390	218	1292
3	600	411	305	1631
6	900	460	559	2696
9	400	454	460	2294
10	500	459	376	1963

Work:

Packet 1:
EstimateRTT = (1 - 0.1) * 400 + 0.1 * 300 = 390 ms
DevRTT = 0.75 * 200 + 0.75 * |300 - 390| = 217.5 = 218 ms
Timeout = 390 + 4 * 218 = 1292 ms

Packet 3:
EstimateRTT = (1 - 0.1) * 390 + 0.1 * 600 = 411 ms
DevRTT = 0.75 * 218 + 0.75 * |600 - 411| = 305.25 = 305 ms
Timeout = 411 + 4 * 305 = 1631 ms

Packet 6:
EstimateRTT = (1 - 0.1) * 411 + 0.1 * 900 = 459.9 = 460ms
DevRTT = 0.75 * 305 + 0.75 * |900 - 460| = 558.75 = 559 ms
Timeout = 460 + 4 * 559 = 2696 ms

Packet 9:
EstimateRTT = (1 - 0.1) * 460 + 0.1 * 400 = 454 ms
DevRTT = 0.75 * 559 + 0.75 * |400 - 454| = 459.75 = 460 ms
Timeout = 454 + 4 * 460 = 2294 ms

Packet 10:
EstimateRTT = (1 - 0.1) * 454 + 0.1 * 500 = 458.6 = 459 ms
DevRTT = 0.75 * 460 + 0.75 * |500 - 459| = 375.75 = 376 ms
Timeout = 459 + 4 * 376 = 1963 ms