CS 372 Introduction to Computer Networks
Self-Check Exercises: Lecture 17 **Solutions**

1) For TCP, please select whether the service is implemented, and indicate how it is implemented.
Connection-Oriented? Yes, implemented by a 3-way handshake at start of connection.
Full Duplex Service? Yes, for example Telnet, which echoes back every character.
In-order delivery? Yes, via sequence numbers in TCP header information
Pipelining? Yes, though pipeline is limited by congestion and flow control
Flow Control? Yes, receiver advertises a window size, and sender limits output accordingly.
Congestion Control? Yes, implemented by restricting pipeline size based on receipt of acknowledgements (or lake thereof)
Bandwidth Guarantee? No – no bandwidth guarantees.
Reliable Delivery(mostly)? Yes – Acknowledgement for delivery of packets
Jitter Threshold? No – no jitter threshold.

2) For UDP, please select whether the service is implemented, and indicate how it is implemented.
Connection-Oriented? No - Connectionless
Full Duplex Service? No – One way service
In-order delivery? No – Packets may be delivered out of order
Pipelining? No pipeline, BUT it is fire and forget (so you can transmit all you want)
Flow Control? No – Sender may overwhelm receiver
Congestion Control? No – Ignores possibility of congestion in the internet.
Bandwidth Guarantee? No – no bandwidth guarantees.
Reliable delivery(mostly)? No – “Fire and Forget” service (“Send it, and, FORGET IT!”)
Jitter Threshold? No – no jitter threshold.

3) What are the minimum and maximum sizes (in bytes) of a TCP header?
20 bytes minimum, due to the five 4-byte rows. 60 bytes maximum, since the “header length” field of the TCP header is 4-bits, so the largest value it can hold is 15. This indicates the number of 4-byte “lines” in the header (4 * 15 = 60).

4) A UDP segment has a “length” field the gives the size (in bytes) of the entire UDP segment, so the receiver can easily calculate the number of bytes of data in the “application data” section. A TCP segment has no “length” indicator. How can the receiver determine how many bytes of data are in the “application data” section of a TCP segment?
TCP transmits a numbered byte stream. The receiver has to count the number of bytes in each received segment. But the segment header (overhead) is not part of the data. So the

receiver counts the total bytes in the segment (call it BC), extracts the header length (call it HL), and finds the number of data bytes as BC – (HL * 4).

5) How is the sequence number for a given TCP segment (from sender to receiver) derived? Give an example.
The sequence number of a segment is the byte-order number of the first byte in the segment, added on to the initial random offset value, minus one. So with an offset value of, e.g. 4013 (this is the sequence number of the first segment, and also the byte-order number of the first byte in the first segment) and considering a segment which is the second segment in a particular connection, with the first segment having contained 768 bytes of data….
The byte-number of the first byte in the second segment is 769 (since 768 bytes were in the first segment).
The offset is 4013.
4013 + 769 - 1 = 4781 (the sequence number of the second segment).
Why minus one? Think about the byte numbers from the first segment…
Byte 1: Number 4013
Byte 2: Number 4014
…
Byte 768: Number 4780
So the first byte of the second segment, Byte #769 in the stream, gives a sequence number 4781 (as shown above).

6) How is the acknowledgement number for a given TCP segment (from receiver to sender) derived? Give an example.
The ACK number is always the next expected byte-order number. Even when receiving an out-of-order segment, the receiver will ACK back the number of the byte that it is looking for next in order to have an in-order byte stream. From the example above, the ACK number in response to the first segment would be the sequence number of the first segment (4013) plus the number of data bytes delivered by that segment (768). This gives 4013 + 768 = 4781. This is the ACK number for this packet.

Considering a different scenario, if the first and second segments are sent, but only the second segment is received, the ACK number for this (out of order) segment will be 4013 – the next byte expected by the receiver – the sequence number of the first segment.

7) How does TCP handle dropped segments? What additional complexity does this add?
TCP handles dropped segments with re-transmission of those segments, upon the end of a countdown timer which started when the packet was transmitted. The receiver has to keep track of sequence number to guarantee it doesn’t pass up duplicate data in the case of a re-transmission when the original segment actually did arrive.