Q1

1.
    a. 3. One for each host.
    b. Since UDP does not make distinctions between different ports, the socket on S1 will only have 1 socket for host B. Therefore, the packet will be sent to that socket.
    c. Just one. Each host makes a single socket to connect to a single server, but since the applications are communicating to the same server, each host needs only 1.
    d. The packet will get sent to B2's only socket that interfaces with S1. Since UDP has a single socket for all S1 connections, the packet will reach that socket and the correct application will read the packet.
2.
    a. 6. Each port and IP needs a new socket.
    b. The packet will be sent to the socket with the specific port and B2's IP on S1.
    c. A will need 1, B will need 3, and C will need 2. Each host needs a separate socket for each application because each application uses a different port.
    d. The packet will be sent to a specific socket that corresponds to the connection between B2 and S1 with a specific port.

Q2

1. TCP - very reliable. Banking apps can’t have dropped packets.
2. Hmmm. UDP would be ok if there wasn’t a focus on reliability. Since there are so many devices it would be nice if you didn’t have to make a new socket for each port.
3. UDP - needs a large amount of data delivered fast. If some data is dropped the user will likely be able to infer what the missing information is.
4. UDP - needs a large amount of throughput and users do not expect perfect audio quality.

Q3

1. A needs to wait for an ACK from B every time.
   100Mbps = around 100000000bps
   So three packets =
   3 * 2* (1000bits/100000000 bps + 3ms) = **18.06ms**
   Throughput = 3000bits/18.06ms = **166,113 bps**
2. A sends 3 packets to B, then waits for 3 acks.
   (1000bits/100000000 bps)*2 + 3ms * 2 = **6.02ms**
   Throughput = 3000bits/6.02ms = **498,339 bps**

Q4

1.
    a. 0
    b. There will be a timeout where A doesn’t see Ack1.
    c. It will send packet 1, 2, 3.
2.
    a. 2
    b. There will be a timeout when A doesn’t see Ack1.
    c. It will only send packet 2.

Q5

1. They're all ack146 because A never resends packet 146!
2.
    a. After the timer for packet 146 expires
    b. Assuming window size of 6: 146, 162, 178, 196, 212, whatever packet is next
    c. Ack162
3.
    a. After the 3rd duplicate ack146
    b. Just packet 146
    c. Ack220, since the rest of the packets delivered correctly

Q6

1. 8. After cwnd reaches cwnd=8, the graph switches from slow start (the exponential growth) to additive increase (the linear sections).
2. There was a packet loss (3 duplicate acks)! After that happens, Reno specifies that cwnd should return to ssthresh, which in this case is 8.
3. There was another packet loss due to 3 duplicate acks. Ssthresh is set to 5 because the packet was lost at cwnd=11, and 11/2=5. Fast recovery specifies that cwnd should return to ssthresh=5.
4. A packet was lost due to timeout, causing the drop to 1.
5. Ssthresh=6 because the last time a packet was lost was when cwnd=12, so 12/2=6. Therefore, additive increase is used. 8+1=**9**.