You are encouraged to type in your answers. Homework must be done individually.

1. (10 pts) **[PD] Page 61: Problem 4**

   Calculate the total time required to transfer a 1.5-MB file in the following cases, assuming an RTT of 80 ms, a packet size of 1 KB data, and an initial 2 $\times$ RTT of "handshaking" before data is sent:

   (a) The bandwidth is 10 Mbps, and data packets can be sent continuously.

   (b) The bandwidth is 10 Mbps, but after we finish sending each data packet we must wait on RTT before sending the next.

   (c) The link allows infinitely fast transmit, but limits bandwidth such that only 20 packets can be sent per RTT.

   (d) Zero transmit time as in (c), but during the first RTT we can send on packet, during the second RTT, we can send two packets, during the third we can send four (2^{3-1}), etc. (A justification for such an exponential increase will be given in Chapter 6.)

2. (10 pts) **[PD] Page 152: Problem 7**

   Suppose the following sequence of bits arrive over a link:

   ```
   01101011111010100111111011001111110
   ```

   Show the resulting frame after any stuffed bits have been removed. Indicate any errors that might have been introduced into the frame.

3. (10 pts) **[PD] Page 158: Problem 25**

   Suppose you are designing a sliding window protocol for a 1-Mbps point-to-point link to the stationary satellite evolving around the Earth at an altitude of 3×10^4 km. Assuming that each frame carries 1 KB of data, what is the minimum number of bits you need for the sequence number in the following cases?
   Assume the speed of light is $3 \times 10^8 m/s$

   (a) RWS=1

   (b) RWS=SWS

4. (10 pts) **[PD] Page 159: Problem 32**

   Draw a timeline diagram for the sliding window algorithm with SWS=RWS=4 frames in the following two situations. Assume the receiver sends a duplicate acknowledgment if it does not receive the expected frame. For example, it sends DUPACK[2] when it expects to see Frame[2] but receives Frame[3] instead. Also, the receiver sends a cumulative acknowledgment after it receives all the outstanding frames. For example, it sends ACK[5] when it receives the lost frame Frame[2] after it already received Frame[3], Frame[4], and Frame[5]. Use a timeout interval of about 2$\times$ RTT.

   (a) Frame 2 is lost. Retransmission takes place upon timeout (as usual).

   (b) Frame 2 is lost. Retransmission takes place either upon receipt of the first DUPACK or upon timeout. Does this scheme reduce the transaction time? (Note that some end-to-end protocols, such as variants of TCP, use similar schemes for fast retransmission.)