1. Problem 5.4: This is an exercise in constructing Huffman codes.

   In the midterm exam, you should be able to answer such a question in 10 minutes. Practice on additional problems in your study group to attain such proficiency.

2. Problem 5.5: This problem is also meant to exercise Huffman code construction.

   To the extent you can, try to specify the “neighborhood” of the 5-dimensional probability mass function \( \left( \frac{1}{3}, \frac{1}{5}, \frac{1}{15}, \frac{2}{15}, \frac{2}{15} \right) \), i.e. the set of all probability mass functions for which the code you constructed continues to be optimal. You need not submit your answer to this part.

3. Problem 5.40 (a): This problem is meant to make you think about the induced probability on the “compressed” sequence.

   Convince yourself that if a source code achieves \( H(P) \), then the sequence of output symbols must have an entropy rate of \( \log_2 D \) bits; i.e. they must be independent and identically (uniformly) distributed.

4. Example 5.26: This problem is meant to make you see the Huffman coding problem as a more generic optimization problem.

   In the same vein, consider a set of \( m \) arrays, with array-lengths \( W_1, W_2, \ldots, W_m \), that you wish to \( \text{merge+sort} \), two arrays at a time. What sequence of mergers minimizes the number of \( \text{compare/copy} \) operations? What is that minimum number?

5. Problem 5.27: Bonus for UGrads, mandatory for Grads.

   If you feel up to it, write a computer program for constructing the set \( S \) of all possible dangling suffixes, and check its validity against your answers to half the codes in Part (c). Then answer the other half using your computer program.

Start preparing for the midterm exam. Review Chapters 1 through 5. Solve lots of problems.