1. Do question #2a in chapter 2 of the textbook (p. 65), regarding your own SOUNDEX algorithm.

2. Pretend we have a bigram array, as in the given table, where the first letter of the bigram is given in the vertical letters, and the second letter is given across the top.

<table>
<thead>
<tr>
<th>first</th>
<th>second</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>e</td>
</tr>
<tr>
<td>e</td>
<td>?</td>
</tr>
<tr>
<td>f</td>
<td>?</td>
</tr>
<tr>
<td>g</td>
<td>?</td>
</tr>
</tbody>
</table>

(a) Make this into a positional bigram array, namely one which captures the position “middle of word”. Provide a word which justifies each 1 you put in the chart.

(b) Each of the 1s you put in the chart may not be equal, in that they may not be equally likely. Discuss your intuitions as to how the chart would change were you to put in frequencies or probabilities. (Feel free to provide such a hypothetical chart.)

3. A user types in *hwo* when they meant to type *howl*. Draw the directed graph and describe how minimum edit distance is calculated. (Adapted from question #3 in chapter 2 (p. 66).)

4. Consider the misspelling *ried*, and assume our edit distance calculations have insertions, deletions, substitutions, and transpositions. Describe how probabilities are used to rank *ride, reed, lied*, and any other words you think of. Do you have an intuition as to which should be highest, and why?

5. I have quasi-randomly selected a text from Project Gutenberg, *More About the Squirrels* ([http://www.gutenberg.org/files/51031/51031-0.txt](http://www.gutenberg.org/files/51031/51031-0.txt)).

   - Note: for this question, you will want to make sure your answer is well-organized. Documenting everything properly in a sensible fashion is maybe 80% of what it takes to do this question well.

   (a) Pick a sentence from this text and write it down. Preferably, pick one which contains a word or two that have fallen out of fashion.

   (b) Corrupt a word or two (or three) in the sentence several times. i.e., for the same word, create several different kinds of misspellings (ones which are more phonetically-based, ones which are close or far off from the original, ones which have a confusable word, etc.)

   (c) Run the sentences through various spelling checkers and report the suggestions for each one.

      i. A standard spelling corrector for text editing: HunSpell, aspell/ispell, Microsoft Word, etc.;
      ii. A smartphone or instant messaging corrector;
      iii. A search engine (for this, use only key terms—and feel free to try different sets of key terms, especially if they vary in interesting ways).

   (d) Given what we discussed in class about correction for different contexts, discuss why you get different results, if you do, or why you get the same results, if you don’t. Where are the “boundaries” for accurate detection or correction of errors? What problems seem fixable with some tweaks to the correction system? For which problems are you just out of luck? (And so forth.)

6. **Bonus:** We don’t have an “honors” version of this course, but for those who want more of a challenge, give a line-by-line explanation of Peter Norvig’s 21-line spelling correction system: [http://norvig.com/spell-correct.html](http://norvig.com/spell-correct.html)

   Then, choose (and possibly) tweak the implementation in the programming language of your choice and test its accuracy. Try different texts to train the spelling corrector, too: can you, e.g., develop a spelling corrector which works really well in a given domain?