1. Draw (a) non-deterministic, and (b) deterministic, finite-state automata corresponding to the regular expression $a^*b^*$.

2. Without relying on predefined functions or methods from the nltk package, write Python code to calculate (i) the number of unique word types in Moby Dick and (ii) the frequency of the most common word in Moby Dick.

3. A proper noun is basically a name—a noun that refers to a specific entity in the world, rather than to a class of entities. Examples of proper nouns are Google, John, John Smith, Warner Brothers, The Soft Bulletin, and so forth. (Notice that a multi-word sequence may constitute a single proper noun.) Suppose you wanted to estimate the number of proper nouns in the text Moby Dick through a comprehensive, automated scan of the text. It’s not at all easy to do an exact count, but constructing a reasonable estimate isn’t too hard. Give a verbal description of how you might go about constructing your estimate. Then write Python code that calculates your estimate (or some approximation thereof), and run the code on Moby Dick and report the estimate you obtain.

4. Write an accepts() method for the Python DFSA class we looked at on January 16, and use it to demonstrate acceptance of the strings ab, aab, and abb, and rejection of the strings a and ba, for the following DFSA:

That is, your accepts() method should be written such that once you have initialized your DFSA accordingly—let’s suppose that your object instance is called x—each of the following code lines should return True:
x.accepts("ab")
x.accepts("aab")
x.accepts("abb")

and each of the following code lines should return False:

x.accepts("a")
x.accepts("ba")