1. Write a program that allows the users to create their own dictionaries. They should be able: a) to insert a new source word plus its translation into the dictionary, b) look up a word, c) look up all the source words, d) delete entries, and e) check whether a certain translation is in the dictionary. They should be presented with these options repeatedly until they select: f) an option to quit.

In other words, you are providing an interface for users to choose an option, give the necessary input, and then the action is performed internally.

Note: assume only one translation per word for this question.

2. (a) Write a program that reads in the POS tagged text from file \texttt{vm.pos} and creates a POS lexicon that stores every POS tag and its frequency. Use a dictionary to store these data. Then print a list of the POS tags and their frequencies.

(b) Extend the program from part #a so that it prints the POS tags based on their frequency in descending order.

Note: do not use NLTK tools for this question.

3. At the top of a program, create a small dictionary of translations, e.g., 20 words from two of your favorite languages, e.g., an English-to-Spanish dictionary. Allow for multiple translations by having each key point to a list of possible translations. Write code which takes a L1-to-L2 dictionary and gives you an L2-to-L1 dictionary, e.g., Spanish-to-English.

Be sure to test it with one-to-many (one L1 with many L2 translations) and many-to-one (many L1 words with the same L2 translation) mappings.

4. Extend the program from question #4 on assignment 6, so that the dictionary values are more complicated. Specifically, the dictionary should have each word as a key, with its value being a list of the all the words which appear immediately after the word in the original text.