Data Structs HW4 Notes

* For 5.1 the table size is 10, even though that's a bad choice (because it's not prime). For 5.2 the new table size is 19 and the hash functions change accordingly. Take a look at the instructions on the assignment for written 1 and 2.
* change hash function to h(x) = x mod 19
* You're not actually inserting the line numbers into the hash table. There are two steps.

Populate the hash table using the dictionary file.

Scan through the input text file line-by-line and word-by-word. For each word check if it is in the hash table. If it is not, print a set of suggested corrections that are in the hash table.

* For the first step you won't be able to do it any faster than O(n) where n is the number of word tokens in the input. You definitely need to look at each one of them.

Can you provide an example of a contraction that we would need to check for?

* You don't need to check for contractions specifically, I just indicated that you can strip off trailing and leading punctuation.

Can't would go in as can't

Jones' would go in as jones

Do we make only one swap adjacent? For example it should suggest "hello" if we check "helol" but what about "ehlol" where we swap adjacent letters twice?

* You use only one swap adjacent, so "ehlol" would not be corrected to "hello".
* You can use the replaceAll method of String to remove punctuation. A simple Google search will tell you how to use this method to remove certain characters. There shouldn't be any need to do extensive reading on regex.

+ the replaceAll method is one of a few ways to remove punctuation from a string. Also make sure the replaceAll method doesn't remove words that are contractions. You can manipulate a String to make sure you don't remove the single quote out of a contraction.

How much should be included in our main method for Programming 1? Can we read the files with a scanner, check if the word exists in the dictionary, or if it can be fixed with the rules, etc. all in the main method?

* All of this should happen when you run the Main method. How you structure that method it up to you. It often makes sense to split up functionality into different methods that solve subproblems independently. This allows you to reuse these methods if you want to write another program that shares some but not all functionality. It's also nice to be able to change the behavior of your main method for testing purposes. If parts of the functionality are I separate methods, you can comment out the method calls.