CS2302 - Data Structures

Summer 2020

Lab 4

Due Wednesday, July 22, 2020

An anagram is a permutation of the letters of a word that produces another word. For example, the word *corona* is an anagram of the word *racoon* (and viceversa). Your task for this lab consists of writing a program to find all the anagrams of all the words in a large set of English words. You will implement two data structures for this: a hash table and a binary search tree.

To find the anagrams, your program should do the following:

1. Read the words from the file *words_ alpha.txt* and store them as a list of strings. Make sure you remove the end of line character ('\n') from every word.
2. Remove from the list all the words that have less than 3 characters.
3. If using the binary search tree, shuffle the list of words. This will prevent building an unbalanced tree, as the words in the original list are sorted alphabetically.
4. For every word w in the word list:
   (a) Find the key k representing w, which will be used to determine the location of w in the table or tree. k is the string that contains the same characters as w but sorted in alphabetical order. For example, if w ='corona', then k ='acnoor'. Notice that if two words are anagrams, they have the same key k.
   (b) If k is already in the table or tree, append w to the data in k's record.
   (c) If k is not in the table or tree, insert a record containing k as the key and a list containing (only) w as the data.
5. For every word w in the word list, retrieve the list of its anagrams. This should be done by repeatedly calling a function that receives a hash table or binary search tree and a word, and returns the list containing the word's anagrams. Your TA will call this function to evaluate your program.
6. Compute and display statistics describing your hash table or binary search tree. For example, load factor, percentage of empty buckets, longest bucket, number of nodes in the tree, and height of the tree. Feel free to suggest others.
7. Display the time it took to build the hash table or BST and the time it took to retrieve the sets of anagrams for each word. Make sure you don't include time to perform input and output in your measurements.

As usual, write a report describing your work and results. Indicate running times and discuss the relative performance of the algorithms implemented.

Notes about implementation: The hash tables implemented in the code provided in class should work with little or no modification. You will need to modify the *bst.py* program to allow to store keys and data. To be consistent with the hash table implementation, I suggest you replace the attribute name *data* by *key* in the program, and add an attribute *data* to the *BSTNode* class. *data* will contain the list of anagrams. You will also need to modify the *insert* function accordingly.