Greg Becker
Programming Assignment 4
28/03/2017

Abstract

The problem presented to us in programming assignment four consisted of three problems in one. First, we had to reconstruct a dictionary into linked lists, each list containing all of the words beginning with a certain letter. After completing this we had to read a text file and search if each word in the file was contained in one of the linked lists. The difficulty during these last two steps was extracting the individual words to be compared from the text file. Our program was designed with two methods. The first assigned each word in the dictionary file to a linked list. The second split up the text file into individual words and compared each word with the contents of the linked list it would appear in if it were in the dictionary file. The last algorithm is more interesting than the first. In it each line would be assigned to a string during the first initial reading of the file. The new string would then be split into an array of words, numbers or non-word strings by spaces. Each word or non-word would be compared by character. During this comparison all letters would be added to form a new word and then this would be searched through the proper linked list. This process results in all non-words phrases (ie “**10**”) to be ignored in the searching process. Problems arising from this method come from hyphenated words and words containing apostrophes. Hyphenated words were split into individual words to avoid the problem of them being searched as compound words. Therefore, hyphenated words are not treated as a single word in this solution. Apostrophes were removed while filling in the linked lists to avoid problems. From this solution we observed a somewhat suspiciously even 3500 average comparisons for words found and 7401 average comparisons for words not found.

```
run:
The average number of comparisons made for words found:
3500
The average number of comparisons made for words not found:
7401
BUILD SUCCESSFUL (total time: 1 minute 9 seconds)
```