Tyler Goodwin

Program 4

October 25, 2016

We are to use the randomized dictionary document given to us and create an array of 26 MyLinkedList objects for each letter of the alphabet. We are to notice that all words are lowercase. After this we are to look at the book “oliver” and look at the first character then traverse one of the 26 linked lists. We are to search each word of the book with the corresponding Linked List, and if not found this will add to a counter for words not found. When a word is found, it will then be add to the counter for words found. Along with these algorithms we must count the number of string comparisons done during the search. We are required to use the string parser method used in our previous assignments. At the end of the program we are to compute then display the average number of string comparisons for words found and not found.

Algorithm one was to create a method that reads the randomized dictionary document and create 26 MyLinkedList objects for each letter of the alphabet. The file is to be imported and read through the algorithm, then once a word is found beginning with a, it is added to the A MyLinkedList object. We continue doing this until all words beginning with a-z are found and placed into individual MyLinkedList objects depending on their first character. We are to also notice that all words are to be lowercase.

Algorithm two was to create a method that reads the “oliver” book and examine the first character of each word, traversing one of the 26 linked lists. We must search each word from the book with the corresponding linked list, and if it is not found, we are to add to the counter of the list of words not found. If the word is found, we are to add to the counter of words found.

Algorithm three was to implement the contains method of a previous lab to count the number of string comparisons that are done during the search. We are to create two counters, one for the number of comparisons for words that were found, and the other for the number of comparisons for the words not found. To find the average words found we are to implement the equation words found = number of words found / total number of comparisons for words found, likewise for the average amount of words not found.

The results of our program should show that there is a larger amount of words found than not found, along with a larger average string comparison for words not found than words found. My observations agreed with this solution, my amount of words found did indeed show a much larger amount that the amount of words not found (914045 to 64546). Along with the second part results, my program showed that the average comparisons for words found was much smaller than the amount for words not found (3554 to 7426). Unlike suggested my code did take quite a while longer than the suggested time of fifteen seconds, taking almost up to two minutes.

Outputs:

run:

Words found:914054

Words not found:64537

BUILD SUCCESSFUL (total time: 45 seconds)