Eric Schulze
C202 Fall 2016
Programming Assignment 4
10/25/16

# Abstract

Program Assignment 4 requires us to compare a text file, that holds the contents of a book, against a dictionary to see how many words in the book are not in the dictionary. We are also required to construct the dictionary based on a provided dictionary text file. I have split my solution into two distinct parts, one for construction of the dictionary and a second for the reading and comparison of the book.

For the construction of the dictionary, I used an array of 26 MyLinkedList objects, with each MyLinkedList referring to a letter of the alphabet. I use a method called populateDictionary() to read in the dictionary.txt file and put each word into the appropriate linked list based on the first letter of the word.

The majority of my solution is dedicated to reading in the words from the book and comparing them against the dictionary. This is accomplished by reading in each string into a holder variable, then examining each character in the string. If the character is a lower case letter, I move the letter to a second comparison variable. If the character is an upper case letter, I make it lower case and then add it to the comparison variable. If the character is a hyphen, I assume that the hyphen indicates two separate words, so I call the comparing method on the current comparison variable, ignore the hyphen, set the comparison variable to empty, and let the loop continue with the second word. If there string has any other kind of characters, they are ignored. Whenever I call my isInDictionary method, I increase one of my counters, either wordsFound or wordsNotFound, depending on the return of the isInDictionary method.

My isInDictionary method uses an overloaded contains method in the MyLinkedList. This method takes in a value and an integer array with only one element. I use an array because I can pass by reference with an array. In the overloaded method, I have it count the number of comparisons that are done and place the value in the array. My isInDictionary method then has access to that count and can update the global variables accordingly.

Output:

```
File Successfully Compared to Dictionary

Total words compared to dictionary: 973648

Total words found in dictionary: 889409
Total words not found in dictionary: 84239

Total comparisons: 3685756655
Total comparisons if word was found: 3138151636
Total comparisons if word was not found: 547605019

Average number of comparisons if word was found: 3528

Average number of comparisons if word was not found: 6500

Time required for comparison: 32.53689657 seconds
```

My output showed a total of 973,648 words from the text file were compared to the dictionary. Less than 10% were not in the dictionary. There were over 3.6 billion comparisons made. The average number of comparisons that were made if the word was found in the dictionary were 3528. The average

Eric Schulze
C202 Fall 2016
Programming Assignment 4
10/25/16

number of comparisons that were made if the word was not found in the dictionary were 6500. It was expected that if the word was in the dictionary, we would have to search through an average of half of the words in the list. If the word was not in the dictionary, we would have to look at the entire list. With the average number of comparisons for not found words at 6500, we can assume that there are about 6500 items in each list. Then 3528 is just a little higher than half the number of items in the list. These are skewed from our theoretical numbers due to the frequencies of the different letters in the english language.