Assignment 2 Part 2B

David Gray s2947315

Contents

1. Problem Statement

The goal of Part 2b of the assignment was to create a spell checking program that reads in a dictionary file (dict.txt) and another file to spell check (newEisenhowerSpell.txt). Using the dictionary file, the program parses the words within the file to spell check and if there is a word that it is not in the dictionary, the program returns suggested words as replacements.

2. User Requirements

There are no user requirements for this program apart from running the executable.

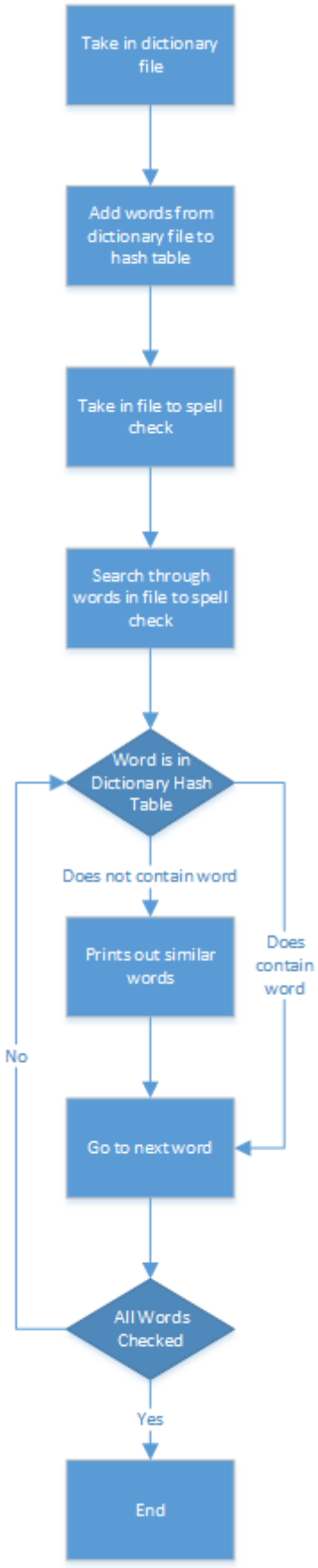
3. Software Requirements

The following outlines the software requirements for the program:

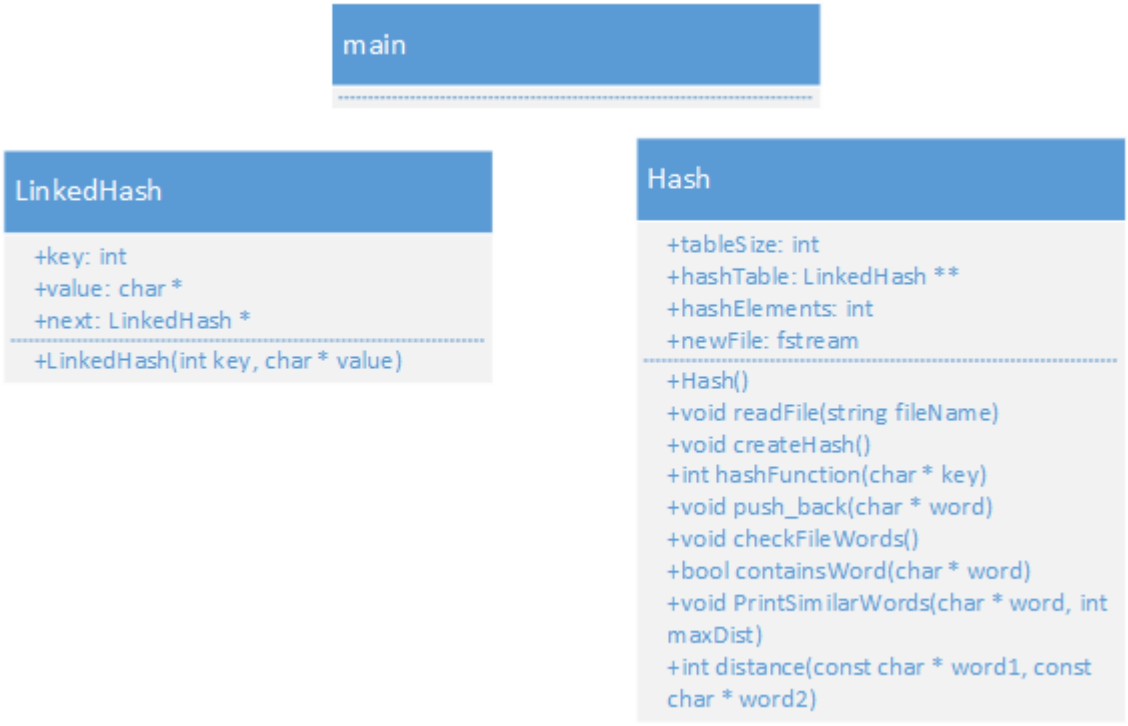
- The program shall read in a dictionary file used for spell checking (dict.txt)
- The program shall store the words from the dictionary file into a hash table
- The program shall read in a file to spell check (newEisenhowerSpell.txt)
- The program shall parse through the words in the file to spell check and check to see if they are in the dictionary hash table.
- If a word from the file to spell check is not in the dictionary hash table, similar words in the dictionary will be found and printed to the user
- The program shall use C++ file functions for all file IO and C string functions for string handling

4. Software Design

High Level Design – Logical Block Diagram



Structure Chart



List of main functions in the software.

createHash(): goes through the input file and adds each word to a char array and stores the char array into the dictionary hash table.

checkFileWords(): goes through the spell check file and adds each word to a char array. Each word is checked to see if it is in the dictionary hash table. If it is not, the PrintSimilarWords function is called to give the user suggested words.

push_back(char * word): adds a word to the dictionary hash table by first getting the hash value of the word by calling the hashFunction method, and storing the word into the hash value location of the hash table.

containsWord(char * key): gets the hash value of the word being checked by calling the hashFunction method. The function then checks the hash table if there is an element at the hash value location and if it is the same as the word being checked. If there is, the function returns true. If there is not, the function returns false.

PrintSimilarWords(char * key, int maxDist): goes through the hash table to find similar words to the word passed to the function. It goes through each element of the hash table and calculates the distance between the word being checked and the current hash table element. If the returned distance value is less than the maxDist value passed through, the word is added to a vector of char array strings that are printed out as suggested words to the user.

distance(const char * word1, const char * word2): checks to see the distance between two words. It calculates the variances in letters between the words and the word length differences between them. These totals are added together and returned.

hashFunction(char * key): uses the djb2 algorithm to calculate a hash value for the char array string passed to the function. The djb2 algorithm is used as it creates minimal collisions for the hash table.

List of all data structures in the software. (eg linked lists, trees, arrays etc)

The main data structures this program uses are a hash table, linked lists, and arrays.

The linked lists are used to store the nodes of the hash table so that the program can traverse the dictionary elements.

Arrays are used to store the C string words from the dictionary and the file to spell check.

The hash table is used to store the C string words from the dictionary. It is used during the spell checking process to see if the word currently being looked at from the file to spell check is in the hash table. If it is not, the hash table is traversed and similar words to the word currently being checked are found and printed to the user.

5. Requirement Acceptance Tests

Software Requirement No	Test	Implemented (Full /Partial/ None)	Test Results (Pass/ Fail)	Comments (for partial implementation or failed test results)
1	Correctly takes in input file	Full	Pass	
2	Correctly parses file contents into a hash table	Full	Pass	
3	Correctly checks for words contained in hash table	Full	Pass	
4	Correctly identifies words not contained in the hash table	Full	Pass	
5	Correctly creates hash value for each word to be stored in hash table	Full	Pass	
6	Correctly prints out similar words to words being checked from the spell check file that are not in the dictionary hash table	Full	Pass	
7	Correctly identifies distance between two words by counting the amount of each letter the words have and the difference between them, along with the word sizes	Full	Pass	

6. Detailed Software Testing

Test	Expected Results	Actual Results
Hash Table Creation		
Tested input registration by entering dict.txt	Program should detect that the input value was a valid file	As expected
Tested string creation by parsing through contents of dict.txt and adding each word to the hash table	Program should print out the content stored in the hash table from the dictionary file	As expected
Tested hash value collisions by checking returned hash values of multiple words	Program should return little to no collisions as the djb2 algorithm effectively calculates hash values	As expected
Tested hash table containsWord method by searching for zoot in the hash table	Program should print out that the word zoot was found	As expected
Tested PrintSimilarWords function by passing through the word zooot	Program should print out the words zoot or zoot's for suggested words	As expected
Tested the distance method of calculating the distance between two words by passing through the words "zoot" and "your"	Prints out a distance between the words of 6	As expected

7. User Instructions

If running the program using the Visual Studio 2010, open the project and then run the program in debug mode.

If using the executable, double click on the file and let it run.

When prompted, enter the name of the dictionary file (dict.txt).

When prompted, enter the name of the file to be spell checked (newEisenhowerSpell.txt).