CIS\*2520 Assignment Three

Spell Checker

## Quick Overview

For this assignment, you will develop a program that performs spell checking on a file. Your application must use a Hash Table with an appropriate hash function and collisions strategy. Your Hash Table ADT must be implemented using the API provided. You may extend the API if you wish but all additional functions must follow the design pattern of the API. All ADTs submitted must implement the full API, be your own work, and you may not use third party libraries for this assignment.

# Overview

For this assignment, you will write a program that uses a Hash Table ADT to perform spell checking on a user provided file. Your application will be provided a dictionary of correctly spelt words via a file passed in by a command line argument. All dictionary words will be placed into the Hash Table.

Once the data from your dictionary file is read and loaded into the Hash Table the user will be presented with a menu system. The menu system must support adding/removing a new word to the dictionary, processing a user provided file for incorrectly spelt words, showing the words inside the Hash Table, and quitting the program.

Your program must not hardcode any part of the input file path. You will submit a tar.gz file to the assignment Courselink Dropbox. The tar.gz file will contain your source code, a make file, your documentation, and your testing files. Your submission must follow the submission guidelines, project structure, and your code must follow common coding standards. Additional details about the organization of these files is provided is the submission guidelines.

## Learning Outcomes

When you have finished doing this assignment you should be able to:

* implement and test a Hash Table ADTs
* design and discuss a suitable hash function
* design and implement an collision algorithm

## Required Prior Knowledge

Before you attempt this assignment please ensure that you have a good understanding of the concepts listed below. You can find review material in the course material as well as in the review materials section of the course.

* Associative Containers
* Function Pointers

# Assignment Description

Your program must receive the dictionary file from the command line (via argv/argc) and insert that data into a Hash Table. The dictionary file will consist of a single word per line and will not contain spaces or invalid characters. Once the dictionary file is loaded into the Hash Table the program should present the user with a menu system. The menu system should be text based and may use printf statements to display information.

Users will be presented with the following options:

1. Add a word to Dictionary
2. Remove a word from Dictionary
3. Spell Check a file
4. Show Dictionary Words
5. Quit

When a user selects to add or remove a word to the dictionary, they should be prompted to enter a single word. Once the user has entered that word the program should attempt to add or remove the corresponding word to/from the dictionary respectively. A success or failure message should be indicated to the user based on the result of this operation. Once complete, the user should be returned to the main menu.

When a user selects option 3, they will be prompted to enter a filename to be spell checked. Your program must handle if that file does not exist with an appropriate message and then return the user to the main menu. If the file exists, your program must open the file and process each word in that file. The file format for the user entered file will be identical to the dictionary file. It will contain a single word per line. For each word in user provided file, check if that word exists in the Hash Table. If the word does not exist in the Hash Table your program should print a message and the invalid word to the screen (See Output Section for format). Correct words are not displayed.

Once your program has processed all words in the file it will display a summary of the text file (See Output Section for format). The summary will be displayed until the user presses any key to return to the main menu.

When a user selects option 4, all the contents of the Hash Table will be displayed to the screen. The output format should display

<Index>:<Key>:<word>

Where <Index> is the row in the table, <Key> is the hash value, and <word> is the original word.

The user my quit the program by selecting option 5.

For clarification on any requirements of this assignment, please post in the CourseLink discussion boards.

## Input Format

All text files provided during this assignment will have the same format a single word per line.

## Output Format

The summary of the spell checking process should provide output in the following format

File processed by Spell Check <filename>:

<Word> was not found in the dictionary

<Word> was not found in the dictionary

<Word> was not found in the dictionary

….

Summary:

Correctly spelt words: <N>

Incorrectly spelt words: <M>

Where <N> and <M> are variables for the number of occurrences for each field.

# Hash Table API

The Hash Table API is available at:

<https://github.com/ian-james/CIS2520/blob/master/labs/APIs/HashTableAPI.h>

You may alter the API to include LinkedList instead of struct Node \* next if you wish. The last functions in the API which are uncommented do not have to be implemented.

# Submission Guidelines

Your source code must be organized and styled according to the course coding guidelines. Submit a single tar.gz file to the Dropbox. **Do not** use zip, rar, or any other compression program. The tar file should unpack to a folder that is your login name. Inside that folder should be your src/ bin/ include/ assets/ docs/ and lib/ folders. Do not submit any binary files.

**Submit** all source files, your testing files, a makefile, and a README file.

Follow all other submission guidelines provided for this course.

**In your readme file, create a section called hash function and collision resolution strategy**. In this section, describe in 5-10 sentences the hash function and collision resolution design and selection process.

# Grading

This assignment is worth 10% of your final mark and will be graded out of 100 marks. You can lose marks for not following the submission guidelines, but will not be given marks for that step. Marks are given for successfully demonstrating mastery of the learning outcomes for this assignment.

# Marking

The following is not a complete list of all the aspects of an assignment that will be evaluated but provides an overview of the major evaluation components.

# /50 Hash Table ADT

## /45 Hash Table

* Hash Table is an ADT
* Contains and implements all API functions.
* Manages and frees Memory
* Properly integrates Reusable Hash Function
* Collision Resolution Strategy
* Hash Table Testing

## /5 Collision Resolution and Hash Discussion

## /50 Program

## /5 File Information

* + Dictionary is properly loaded from command line argument

## /40 Spell Checker

* + Correctly adds/removes words from dictionary
  + Search Dictionary to find words
  + Spell Check process
    - Identifies words that don’t occur in dictionary
    - Output Information
    - Calculates Summary
  + Menu System Implementation

## /5 Output Format

* + Information is displayed in requested order and format.