Project 7

**PRIME FACTORIZATION USING AN ARRAY-BASED STACK**

Full Name: Prashul Shrestha

Section #: 2

Project #: 6

Due Date: March 27, 2017

**Design** **Document**

**Introduction**

**A stack** is a basic data structure that can be logically thought as linear structure represented by a real physical stack or pile, a structure where insertion and deletion of items takes place at one end called top of the stack.

**Data** **Structures**

The program uses a class called Class **Concordance()**, a char **array**[] named **Word**, and a singly linked list to store each character into the node inputted by the user. First, the file is read from the .txt file and stored in a **ifstream** file and then is send to public **insert()** function. From there, one word is taken and stored in char array Word[] and send to private **help\_insert**() function. There we use recursion and **strcmp()** to check for same words already present in the list and to alphabetically arrange the list.

**Functions**

The program uses **Six** functions to implement the Concordance through Singly-linked List. The functions are called from main() and some are member function to return the result within the function which called it. The list of the functions are given below:

* Concordance() : A constructor to initialize the list to be empty.
* ~Concordance(): A destructor to delete nodes in the list and is called by default.
* **Void insert**(**ifstream& in\_File**) – This will help private insert function to send one word as char array in private insert() function.
* Bool is\_empty(): checks if the linked list is empty or not.
* Output - Overloads the "<<" operator to write the invoking concordance to an output stream.
* **get**\_**node**(**entry**, node p, int count) – Returns or initializes new node to store the char array in to singly linked list.

The program uses **string**.**size**() from **string**.h library to get the length of the string. It also uses **strcmp()** and **strcpy()** to compare the char array and the string, and copy one char array into another.

**Menu**() is a the main function from where we first create an object or an instance of class **Concordance** called **Cord** and which helps invoke the functions such as **Cord.insert**(in\_File); and **Cord.get\_count(entry)** to get the count of the word in the list.

**The Main Program**

**Menu**() is a the main function from where we first create an object or an instance of class **Cord** and which helps invoke the functions such as **Cord.insert**(**in\_File**); to input characters into the node. The program is quite simple first, the strings are inserted into the node while checking through the list using **strcmp()** and inserting if it’s not present or inserting and incrementing the count while removing the inserted same string.

User Document

A **concordance** of a text is an alphabetical table of the words that appear in the text and the number of times each word appears. Concordances summarize the frequencies of words in text and are used in statistical analyses of authors' works and to determine authorship of disputed works. It is used in many online software of checking plagiarism. Here, we implement singly-linked list to get the concordance of inputted file.

The main program named **main**.**cpp** can be compiled and run, using the code:

**g++ main**.**cpp**

**a.out**

**g++** function will compile the function and make it ready to be run using **a.out**. The function will prompt the following output:

**Ouptut:**

**Word Count**

**------------------------**

**a 2**

**brown 2**

**fox 1**

**jumps 1**

**lazy 1**

**over 1**

**quick 1**

**------------------------**

**The File contains 7 distinct words.**

**Enter the word: brown**

**Count of the word brown = 2**

**Summary**

Completing this project, I learnt the use of singly-linked list in the real-world experience. Since in this project we sued char array[] to store each string and passed it to the insert function, I learnt that strcmp() doesn’t work with strings as whole, but as characters present in that string.

Other thing, that I learnt while doing this project is that, while we pass a char array[], a pointer cannot point to the array as whole, so we need to use **strcpy()** to copy the elements of the array into another.

Working with pointers and their semantics is helping ,y broaden my knowledge and analyze me the use of pointer in making complex program into simpler versions.