Texas Tech University Whitacre College of Engineering

CS 2413/5401 – Data Structures
Spring 2022
Lab Assignment 3

Acknowledge your collaborators or source of solutions, if any. **Online submission is required.**

While designing your programs or answering items, you are free to come up with your own assumptions based upon concepts and material learned in the course, if every potential specification is not given to you. Just be reasonable and document your assumptions. Such assumptions should not conflict with concepts and material learned in the course.

Your compliance with the "PROGRAMMING STYLE GUIDELINE" for CS 2413/5401 will affect your actual grade. All assignments will be checked for academic misconduct (cheating, plagiarism, collusion, falsifying academic records, misrepresenting facts, violations of published professional ethics/standards, and any act or attempted act designed to give unfair academic advantage to oneself or another student) defined by "OP 34.12: Grading Procedures, Including Academic Integrity" of TTU. Special software will be used to uncover such attempts.

A subset of answers submitted in this lab may be graded.

Objective: Practice using C structs and linked lists

Tasks:

1. You may work on this assignment by yourself, or you may work with one other student in this course as a team to complete this lab assignment.
    a. Teams larger than 2 people will incur a 25% penalty off the total lab points per extra person.
    b. It is expected that each team member contributes equitably and participates in coding and design ideas.
    c. If a team member is dissatisfied with the performance of the other team member, you are allowed to dissolve the team and continue individually.
        i. Whatever code each team member has contributed may be taken with that team member.
        ii. Try not to let such a decision wait for the day the assignment is due as no extensions will be given if a team is dissolved.
2. Write a C program, problem1.c, to read in a text file, called text.txt, and build a concordance file, called concordance.txt, for the file showing the number of unique words, the unique words, and the number of times each unique word occurs in the text file. A word only has alphabetic characters. Test your program with text.txt files of differing contents.
    a. Example
        i. Text File – text.txt (test empty, one word, and larger text files)
            1. A concordance of a text file is an alphabetical list of the unique words in the text file.
        ii. Concordance File – concordance.txt
            1. There are 13 distinct words in the text file:
            2. a 2
            3. alphabetical 1
            4. an 1
            5. concordance 1
            6. file 2
            7. in 1
            8. is 1
            9. list 1
            10. of 2

11. text 2
12. the 2
13. unique 1
14. words 1

b. Use an ordered linked list to store the words and keep a count of the number of times each word occurs in the text file.
   - **i.** The linked list operations, such as insert, search, and others, should be written as separate functions.
   - **ii.** The head of the list should be a pointer that points to the first node in the linked list.
   - **iii.** In the above example, head should point to the list node containing "a" and 2.

c. The main function should be a driver to call other functions to perform the required tasks to build a concordance and store the concordance to the file "concordance.txt".
   - **i.** No global variables should be used but define macro constants and typedef's may be used.
   - **ii.** The C library files string.h and ctype.h may be used, but please use the safe string operations, such as *strncmp* in string.h.

d. After the linked list of words is built and concordance.txt is made, delete the stop words in the file stopwords.txt from the list one at a time. Then output a new concordance file called "concordance_wo_stop_words.txt". Test your program with stopwords.txt files of differing contents and order of stop words.
   - **i.** Stop words file – stopwords.txt (test empty, one word, and larger stop word files)
     1. a
     2. of
     3. in
     4. an
     5. the
   - **ii.** Concordance File – concordance_wo_stop_words.txt
     1. There are 8 distinct words in the text file:
     2. alphabetical 1
     3. concordance 1
     4. file 2
     5. is 1
     6. list 1
     7. text 2
     8. unique 1
     9. words 1

e. Report: In the comments at the end of the program, give the following information:
   - **i.** Team Member Names
     1. For each team member, detail the work on the program concerning specific work, test cases, and code and functions designed, implemented, and modified, such as
        - **a.** Name
          - **i.** void insert (node_t **head, int num); - designed/implemented/modified
          - **ii.** created text.txt and stopwords.txt for test case 1, 2, 3, …
          - **iii.** …
     2. A grade penalty of up to 25% of the total assignment points, which will be in addition to any other penalties, may be considered if inequitable contributions are made, not enough detail is present to be convincing, and/or all team members have the same list;

i.e., all team members allegedly did the same thing (if both team members do the same thing – one team member is not needed).

ii. Test Cases and Status
    1. Example text.txt and stopwords.txt – passed/failed
    2. text.txt with one word and empty stopwords.txt – passed/failed
    3. text.txt with one stop word and stopwords.txt with that stop word – passed/failed
    4. empty text.txt and nonempty stopwords.txt – passed/failed
    5. text.txt with > 500 total words and stopwords.txt with > 10 total words – passed/failed
    6. …

iii. Ordered Linked List Analysis
    1. Example of a worst case for inserting n words into an ordered linked list
    2. Big O of the worst case for inserting n words into an ordered linked list
    3. Big O of the storage requirements for a linked list of size n

Learning Outcomes:

- Understand C program concepts, such as structs and pointers
- Understand how to implement linked lists

Grading: 50 points

- Standard Deductions - 14 points
- Problem 1 – 36 points (problem1.c, text.txt, stopwords.txt, concordance.txt, concordance_wo_stop_words.txt) with the rest of the test cases as text1.txt, stopwords1.txt, …)
    - Linked List – 12 points
    - Report – 12 points
    - Test Cases – 12 points

Due Date:

2/4/2022, 11:59pm (submitted on Blackboard by ONE team member)