COSC2430: Programming and Data Structures

Print a file content in alphabetical order using AVL trees

Introduction

In this homework, you will create a C++ program that uses an **AVL tree** to read an input file and print the words in the text in alphabetical order.

Input and Output

The input is a single text file containing a paragraph of unknown length.

Example 1 of input

Once upon a time, there was a little girl who lived in a village near the forest.

The output of your program should be a text file containing all the words in the original file in alphabetical order. **Only words must enter the tree (follow rules of HW3.** Punctuation and numbers (including words including numbers or special characters) will not be included in the output. Furthermore, duplicates are not allowed in the tree. All words should be listed on a new line with the first character uppercase.

At the end of the output, you will write the number of rotations required to keep the tree balanced.

Example 1 of output

A
Forest
Girl
In
Little
Lived
Near
Once
The
There
Time
Upon
Village
Was
Who
4

The main C++ program will become the executable to be tested by the TAs. The result should be written on another text file (output file), provided on the command line. The input and output files are specified in the command line, not inside the C++ code.

The general call to the executable (sum_rowcol, in this example) is as follows:

```
tree_sort "A=<file>;C=<file>"
```

Call example with one input file and another output file.

```
tree_sort "A=a.txt;C=c.out"
```

Requirements

- **It is NOT allowed to use vector classes or other classes provided in the STL.**
- Your C++ code must be clear, indented and commented.
- Your program will be tested with GNU C++. Therefore, you are encouraged to work on Unix, or at least make sure that your code compiles and runs successfully on the server.
- You can use other C++ compilers, but the TAs cannot provide support or test your programs with other compilers.
- **Bonus:** +5 points for implementing tree traversal using function as parameters in this homework.

Testing for grading:

- Your main cpp program will be automatically compiled. If there is an error in compilation it will not be executed. **Programs that do not compile on the server will receive a score of 0.** You are responsible of the content of your submission folder.
- **The name of your submission folder must be hw4**. The folder must not be nested in another folder.
- Submitted files must be limited to headers and source files (.h and .cpp).
- Your program should not crash, halt unexpectedly, take an unusual amount of time to run (more than 10 seconds) or produce unhandled exceptions.
- Your program will be tested with 10 test cases, going from easy to difficult.

DEADLINE: November 15, 10:00PM – NO LATE SUBMISSIONS

The due date cannot be changed. No exceptions, unless there are medical or exceptional reasons.

Grading

- A program not submitted by the deadline is zero points.
- A program that compiles but does not work program is worth 10 points. .
- A code with no comments will receive a 5 points penalty.

Plagiarism and cheating: C++ code is individual: it cannot be shared (other than small fragments used in class or in lab examples). Programs will be compared for plagiarism. If the TAs detect that a portion of your C++ code is highly similar to C++ on the Internet or other student the grade will be decreased at least 50%.

About punctuation

Please read the following comments that clarify what you should expect in term of punctuation and how to handle it:

Scenario	Example	Admit in stack	Notes
Words containing only letters	word	word	
Words including numbers	word11	no	
Words including apostrophes	word's I've	word's I've	
Words preceded by regular punctuation	"word	word	Admitted before a word: ("
Words followed by regular punctuation	word.	word	Admitted after a word:) , . ; : ? " !
Words with special characters	w@rd	no	
Words with punctuation in the wrong place	wo.rd	no	

You can assume that the test cases will not include more than one punctuation character consecutively. Therefore, you will NOT be tested on examples such as:

- (finally!)
- "How are you?"
- Fine…