Lab 2: Using fstream of C++

10 Points Possible

2/19/2024

Add Comment

Details

In this lab, we will learn how to use the `fstream` in C++ by writing a program which implements a simplified version of the `cat` command.

`cat` is a commonly used Linux command to concatenate the content of one or more files and print the combined content into the screen. First, in a Linux terminal, please prepare any two text files, `file1.txt` and `file2.txt`. Then run the command `cat file1.txt file2.txt` and observe the output. As you may see, the cat command simply combine the content of two files together and then output the combination into the terminal.

`cat` also allows users to execute the command with some extra option. Using the `-s` option will squeeze consecutive empty lines into one single empty line. For example, if the `file1.txt` is like the followings:
hello

world

birthday
There are three consecutive empty lines between **hello** and **world** in `file1.txt`. And the `file2.txt` is like the followings:
mountain

hill

There are two consecutive empty lines between **mountain** and **hill**.

Run the command `cat file1.txt file2.txt -s` will output:
hello

world

birthday
mountain

hill

The consecutive empty lines are squeezed into one empty line when using the `-s` option. An empty line does not contain any character. If a line only contains some blank or tab, it is not an empty line. Using `-s` with the `cat` command will not treat any line only containing blank as an empty line. Note that in the output of the above example, there is still one empty line after the **hill** because there is one empty line after **hill** in `file2.txt`.

Please write a C++ program implementing the two above functionalities: concatenate two files with and without squeezing consecutive empty lines. There is a starting point `file_cat.cpp` in Lab 2 (https://osu.instructure.com/courses/160885/files/folder/labs/Lab%202) folder. Your implementation should be a single file named file_cat.cpp. This source file should be able to compile using the command `g++ file_cat.cpp -o file_cat`. After the compilation, the program should be able to run the following command for any two input files `FILE1` and `FILE2`:
`./file_cat FILE1 FILE2`
The output of the above command should be the same as the output of command `cat FILE1 FILE2`. The program complied from your implementation should also execute the following command:
`./file_cat FILE1 FILE2 -s`
The output of the above command should be the same as the output of command `cat FILE1 FILE2 -s`. In summary, the `file_cat` program implements a subset of functionalities of the `cat` command.

The manual of `cat` can be found here (https://www.man7.org/linux/man-pages/man1/cat.1.html). Note that, here we simplify the case of the original `cat` command. We assume the `file_cat.cpp` only needs to consider the case where there are two input files. And if the `-s` is one of the input arguments, `-s` is always the last argument.

Please submit the finished `file_cat.cpp` to Lab 2 (https://www.gradescope.com/courses/697394/assignments/4049685) on Gradescope similar as previous Lab 1 (https://osu.instructure.com/courses/160885/assignments/3952611). The autograder will check if the implementation works as expected. If there are some issues, please try to run the `cat` command and compare the output of the `file_cat.cpp` implementation and the output of `cat` command. CSE department provides Linux machine as shown in here (https://cse.osu.edu/computing-services/resources/remote-access).

Should there be any questions about this lab or the autograder, please email TA at yang.5229@buckeyemail.osu.edu (mailto:yang.5229@buckeyemail.osu.edu).