Project OS2

Due Date: October 15

Purpose

This project will let you try out the semaphore library available with threads in the Linux POSIX implementation.

Problem

LaTeX is a text formatting system that I use to create your assignment descriptions, among other things. The output of the system is much better than that of most word processors, such as Word, especially for mathematical notation, but this performance comes at a price: it is not WYSIWYG[1]. It is, instead, a text formatter, and as such, the user must type in special control sequences to accomplish the desired output. For example, the control sequence `{\bf hi}` means print "hi" in bold: **hi**. You will write a program that will take similar control sequences and format a document.

Input

The input will be a text file of any length. The program should prompt for the name of the file. The file will contain normal words and punctuation, and may contain the following control sequences which act on **the next word**:

Control Sequence	Meaning
`\c`	capitalize the first letter of the next word
`\C`	capitalize all of the letters of the next word
`\u`	underline the next word (simulate this by `_word_`)

Note that the control sequence will **immediately precede** the word to format; there are no extra blanks between the sequence and the word. You may assume that there are no incorrect control sequences in the input file.

A blank line indicates a new paragraph; the first line of the file also indicates a new paragraph.

For example, the following lines would be valid input:

```
Hi there, \cbob.  What
a big \unose you have!
It's truly \Csmellerific.
```

Output

The program should display the file, carrying out the control sequence instructions and deleting the control sequences themselves. New paragraphs need not be indented, but should be preceded by a blank line (except the first). The resulting text should be both left and right justified. Do this by inserting extra blanks in between words to align the right column. This does not have to be super-pretty, but spreading out the spacing in a nice way will improve your grade. You do **not** have to move words from one line to another for purposes of creating better alignment in the output.

[1] WYSIWYG = What You See Is What You Get.

The output for the above input (based on a line width of 25) might be:

```
Hi   there,   Bob.   What
a  big  _nose_  you have!
It's truly SMELLERIFIC.
```

Note that the last line should not be aligned.

Specifics

Ahhh, what's the catch? You need three threads! One thread should read the text and store it into a two dimensional array. For the purposes of this assignment, assume that no input line will be longer than 60 characters and that there will never be more than 200 lines. The second thread should take care of the control sequences, working on one line at a time. The third thread should align and display the text to a 50 character width, again working one line at a time.[2] In order for this to display properly, the threads must be synchronized so that:

- a thread should process a line as soon as it is completed in a previous thread; that is, the entire text should NOT be read in before other processing begins
- threads work only on lines that are completely stored in the array
- the control sequences are eliminated before the formatting begins
- the text is printed in the proper order

Other items:

- This project must be completed using ANSI C++ and the `pthreads` library.
- Although not strictly required, I urge you to use Linux. I will test your programs in Linux. A major difference between platforms is the use of `eof` to mark the end of an input file (yes, Macs too!); be sure to test your program in Linux to be certain that your files are read properly.
- You may work on this project with **one** partner. If you work with a partner, submit just one project that includes both names. Both partners will get the same grade, so choose wisely.

Notes

Work on this in stages, and be sure your functions do the proper formatting before you work on the threads. When each portion is complete, you can then concentrate solely on thread synchronization.

October break is in the middle of this project; **do not delay!**

Don't forget to comment your code and to follow good programming practices.

Submit your source code by emailing it to me at mgousie@wheatoncollege.edu. Name your file *Last-Name*OS2.cpp as in `GousieOS2.cpp`; in the case of partners, either name is fine. Submit hard copy of your source code in class on October 16.

You're in trouble here.
– LaTeX error message

[2] The extra 10 characters allowed in the input are for control characters that are not printed.