CSc 415 Project 2

(This document available on iLearn.)
due Monday 10/7/2019 11:59pm
(standard 48-hour grace period, for 75% credit)
(8% of your grade)

This is an individual assignment. Work on your own!

All the files you need are here: http://unixlab.sfsu.edu/~whsu/csc415/P2/

Recall for Project 1, we looked at fileStats.c, similar to the Unix utility wc. Rewrite fileStats.c so that each input text file is handled by a different POSIX thread. Suppose there are N input text files to be counted. Follow these steps:

```
    Main thread creates N concurrent threads
    Each thread:
        computes counts for one file
        writes its counts to a FileInfo struct
        prints its thread ID and its counts to stdout
```

Main thread waits for each thread to exit, reads the FileInfo struct from the exiting thread, and accumulates counts. When all threads are done, the main thread reports the totals.

Make sure there are N concurrent threads! Otherwise points will be deducted.

Make sure that your code works with any reasonable N. That is, if you need an array with N elements, allocate the array dynamically, not statically.

Suppose the multi-threaded version of fileStats is called mtFileStats. The output produced (same as mpFileStats):

```
unixlab: ./mtFileStats Oliver.txt Pessoa.txt Davis.txt
Process 53587 Oliver.txt: 7 lines, 85 words, 453 characters
Process 53588 Pessoa.txt: 1 lines, 8 words, 55 characters
Process 53589 Davis.txt: 10 lines, 123 words, 723
characters
Total: 18 lines, 216 words, 1231 characters
unixlab:
```

The counts for each file should be accurate, though the threads may report counts in a different order.

Submission:

Submit a .c file (or, if there’s more than one file, a tar/zip file containing all source code and headers) using the iLearn submission link. Each source file should have a header with accurate instructions on compiling and running your code on the Unix command line. If your instructions don’t work perfectly, you may get a zero on the project.