EEL 5934/4930 Advanced Systems Programming
Assignment 2
Due: Friday, February 5th by midnight

In this assignment you are going to use Pthreads library and implement a multithreaded program that reads a file of words and generates some statistics such as the count of each word and the count of words that start with the same letter. You are encouraged to use your implementation of Mapper and Reducer programs from the previous assignment. Unlike the previous assignment, your solution will consist of a single process, which divides the work among multiple threads.

You should have the following type of threads in your solution:

- **Mapper Pool Updater:** Reads all the words from a file and puts those that start with the same letter to a separate entry in the *Mapper pool*. For increased performance, by the time it reads a word that starts with a new letter it should have stored all the words that started with the previous letter in an entry the Mapper pool.
- **Mapper:** Gets an entry from the Mapper pool and creates tuples of the form (word, 1) for each word in its workspace and writes it to an entry in the *Reducer pool*.
- **Reducer:** Gets an entry from the Reducer pool. The entry includes tuples of the form (word,1), where all the words start with the same letter and creates tuples of the form (word, count) and writes it to an entry in the *Summarizer pool*.
- **(optional) Summarizer:** Gets an entry from the Summarizer pool. The entry includes tuples of the form (word, count), where all the words start with the same letter. Creates a tuple of the form (letter, totalCount) and writes it to an entry in the *Letter Count Table*.
- **Word Count Writer:** Reads from the Summarizer Pool and writes it to file wordCount.txt.
- **(optional) Letter Count Writer:** Reads from the Letter Count Table and writes it to the file letterCount.txt.

In addition to the input file name, your program should get three additional parameters from the command line in the following order: number of mapper threads, number of reducer threads, and number of summarizer threads. Note that there will be one Mapper Pool Updater thread, one Word Count Writer thread, and one Letter Count Writer thread (optional). If you do not implement the optional parts, then you should get two parameters only.

Assuming that contents of input.txt is as follows:

course cap class culture class cap course course cap culture concurrency course cap class door date date date door date door door error error eraser error eraser eraser eraser eraser field floor field field field field group group group group group group gap group group group gap gap

gap hole hole hole hole hole heat heat heat house heat hierarchy inconsistency ice ice ice ice inconsistency

And let's assume your executable is named wordStatistics, an example usage will be

$ wordStatistics input.txt 2 3 1

where 2 Mapper threads , 3 reducer threads, and 1 Summarizer thread will work in cooperation to produce wordCount.txt and letterCount.txt as follows:

Contents of wordCount.txt:

(course,4)
(cap,4)
(class,3)
(culture,2)
(concurrency,1)
(door,4)
(date,4)
(error,3)
(eraser,5)
(field,5)
(floor,1)
(group,9)
(gap,4)
(hole,5)
(heat,4)
(house,1)
(hierarchy,1)
(inconsistency,2)
(ice,4)

Contents of letterCount.txt:

(c,14)
(d,8)
(e,8)
(f,6)
(g,13)
(h,11)
(i,6)

Important Note: You should recognize the producer-consumer relationship between your threads and provide necessary synchronization for mutual exclusion as well as waiting for data to arrive as well as an entry become available in the output pool. So you should make your threads sleep when there is no work to do and have other cooperating threads wake them up when there is some work to. To get full credit for your solution, you should use synchronization primitives of the Pthreads library effectively.

Submission: You should submit your source code for the three programs along with a Readme and preferably a Makefile on CANVAS.