CS 420
Lab 2
Computing a Histogram Using Multi-threads in C++11

Description: You are to write one VS 2022 x64 Console C++11 programs to implement two different algorithm designs of algorithms to compute a histogram of the contents of any (text or binary) file. Each implementation must use C++11 threads, uses as many threads as possible, and provide thread-safe access to internal data structures. All threads should independently read a portion of the input file, which must be divided into segments as determined by your code and so that all threads are assigned work.

Ultimately, your program must print the histogram of the input file first by using a global array representing the histogram. To get to the final histogram there are two designs:

(1) As each thread reads the next byte value, k, from the file, the thread increments the corresponding the k^{th} bin of the histogram; or
(2) Each thread has a histogram for the section of the file the thread reads. When all threads are complete, the main histogram is updated by merging the results of the local histograms.

After implementing each approach correctly, you must determine which design-implementation is more time efficient.

Name Your Project and Zip files: CS420Lab02*LastName*.

Due Date and Time: See Blackboard.

Definition of a Histogram: Given a universal set of data, U, and a multisubset $S \subseteq U$, for each value x in S, find the number of occurrences of x in S, h(x). If x is not in S, let h(x) = 0. (Note: multisets allow repetition of elements.)

Background and Implementation Constraints:

- *Fundamentals:* For this problem let U = {0, 1, 2, … 255} since only files of 8-bit bytes are to be used. Each 8-bit binary number is to be interpreted as an unsigned number. (Remember, $|U| = 2^8 = 256$.) Therefore, for this assignment, each histogram must be represented with 256 frequency counts for each file containing only values of {0, 1, …, 255}.
- *Multi-threading:* You must maximize the thread usage in your solution; that is, if you run this on a quad-core with hyper-threading of 2, you should have 8 total threads. For files of size n, where n is divisible by the number of threads without remainder, (i.e., n % num_threads == 0), all threads should sequentially process an equal number of bytes. The entire file should be read. If n % num_threads != 0, one thread should process the remaining part of the input file.
- *Two Solutions and Respective Required Designs:* You must implement the two designs given here.
    1. There is one global histogram. All threads share the same global histogram. The results in this global histogram is used for the output. This output will appear first.
    2. There is one local histogram per thread. After all threads finish, the main thread (or one of any of the threads) accumulates the results from all local histograms into one global histogram that is used for the output. This output will appear second.
    3. Assume all file data can fit into your virtual address space.

- *Running Program and I/O requirements:* Run from the command line.

  `CS420Lab02LastName.exe` *`Filename`*
  - *`Filename`* can be any type of file, text or binary. Binary files include files like .pgm, .jpg, … .
  - Output should be (separate lines, no blank lines where h(x) is long long int value):

```
Run with one global histogram
0: h(0)
1: h(1)
…
255: h(255)
Run with local histograms
0: h(0)
1: h(1)
…
255: h(255)
```

Reference. Please see presentation slides by Dr. Joe Hummel found under the Resources->Concurrency Supplements Folder for this Blackboard course. That will help you understand C++11 threads.

C++ Implementation Requirements.

- Name your VS 2022 Debug version x64 Solution/Project: CS420Lab02LastName.
- The name of your executable corresponds to the name of your Project. I need the name to be exact because of the test scripts I have written.
- Example code of how to read a file into memory is shown below.

Submission Requirements.

- When submitting your solution/project zip the entire VS 2022 top-level solution folder (which includes a folder for your project). I need EVERYTHING in your solution folder. Again, the executable file should be named CS420Lab02LastName.exe.
- The NAME of the zip file that you submit should be CS420Lab02LastName.
- Make sure your test cases (input test files and clearly marked output file that clearly correspond to the input test files) are in one of the folders for your VS 2022 solution. DO NOT put any zip file inside a zip file.

//C++ function to transfer contents of a file into a buffer in RAM

```
void fileToMemoryTransfer(char *fileName, char **data, size_t & numOfBytes) {
	streampos begin, end;
	ifstream inFile(fileName, ios::in | ios::binary | ios::ate);
	if (!inFile)
	{
		cerr << "Cannot open " << fileName << endl;
		inFile.close();
		exit(1);
	}
	size_t size = inFile.tellg();
	char * buffer = new  char[size];
	inFile.seekg(0, ios::beg);
	inFile.read(buffer, size);
	inFile.close();
	*data = buffer;
	numOfBytes = size;
}
```