Group Project 2: Application for Threads Sorting

CECS 326 – Operating Systems

1. Summary

This project follows the topic of threads and asks you to design a real scenario/application with multiple threads. You will use the threads programming interface, POSIX Threads (Pthreads). You should implement this in Linux, which supports Pthreads as part of the GNU C library.

You should submit the required deliverable materials on BeachBoard by **11:55pm, March 14th (Sunday), 2021**.

2. Description

This project asks to write a multithreaded sorting program that works as follows: A list of integers is divided into two smaller lists of equal size. Two separate threads (which we will term sorting threads) sort each sublist using a sorting algorithm of your choice. The two sublists are then merged by a third thread—a merging thread—which merges the two sublists into a single sorted list. In the above list, x is a placeholder for the student identifier.

Because global data are shared across all threads, perhaps the easiest way to set up the data is to create a global array. Each sorting thread will work on one half of this array. A second global array of the same size as the unsorted integer array will also be established. The merging thread will then merge the two sublists into this second array. Graphically, this program is structured as in the below figure:
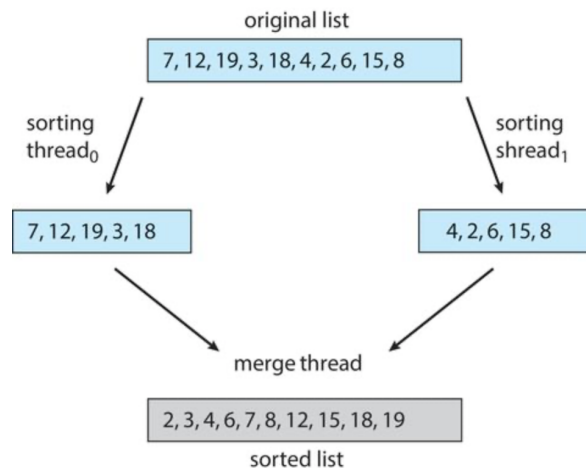
This programming project will require passing parameters to each of the sorting threads. In particular, it will be necessary to identify the starting index from which each thread is to begin sorting.

The parent thread will output the sorted array once all sorting threads have exited.

3: The Required Deliverable Materials

(1) A README file, which describes how we can compile and run your code.
(2) Your source code, you may use a Makefile and be submitted in the required format.
(3) Your report, which discusses the design of your program.
(4) A recorded video shows the output and runtime

3. Submission Requirements

You need to strictly follow the instructions listed below:

1) This is a **group project**, please submit a .zip/.rar file that contains all files, only one submission from one group.

2) Make a **video** to record your code execution and outputs. The video should present your name or time as identification.

3) The submission should include your **source code** and **project report**. Do not submit your binary code. Project report should contain your groupmates name and ID.

4) Your code must **be able to compile**; otherwise, you will receive a grade of zero.

5) Your code should not produce anything else other than the required information in the output file.

7) If you code is **partially completed**, please explain the details in the report what has been completed and the status of the missing parts, we will grade it based on the entire performance.

8) Provide **sufficient comments** in your code to help the TA understand your code. This is important for you to get at least partial credit in case your submitted code does not work properly.

Grading criteria:

Details	Points
Have a README file shows how to compile and test your submission	5 pts
Submitted code has proper comments to show the design	5 pts
Screen a *video* to record code execution and outputs	10 pts
Have a **report** (pdf or word) file explains the details of your entire design	30 pts
Report contains clearly individual contributions of your group mates	5 pts
Code can be compiled and shows correct outputs	45 pts

4. Policies

1) Late submissions will be graded based on our policy discussed in the course syllabus. 2) Code-level discussion is **prohibited**. We will use anti-plagiarism tools to detect violations of this policy.

5. Resources

The Pthreads tutorials at https://computing.llnl.gov/tutorials/pthreads and

http://pages.cs.wisc.edu/~travitch/pthreads_primer.html are good references to learn Pthreads programming.

6. References

[1] POSIX Threads Programming: https://computing.llnl.gov/tutorials/pthreads/

[2] Pthreads Primer: http://pages.cs.wisc.edu/~travitch/pthreads_primer.html

[3] POSIX thread (pthread) libraries:
http://www.yolinux.com/TUTORIALS/LinuxTutorialPosixThreads.html