Operating Systems Lab (CS 470):

Lab 3: Given a sequence of numbers, $x_0, x_1, \ldots, x_n^2 \in R$ in a text file- separated by a '\t' character, write a simulation process in C/C++ under Linux/MacOS as follows: a) start M threads, b) each thread is generating an (i,j) pair randomly such as 0<=i<=j<n, and each thread is sorting the values from x_i to x_j in the text file and write the values in the file once sorted. The threads are finished when the list is completely sorted otherwise a new (i,j) pair is to be generated and repeat the previously mentioned process. Each thread each time after generating a new (i,j) pair should randomly select which sorting algorithm to consider. The possible algorithms should be a) quick sort, b) insert sort and c) bubble sort.

Overview

Reading and modifying (increment, decrement, assign) a shared variable/file between different threads needs extra attention due to inconsistency which might occur if the processes run in parallel. In order to avoid this, different mechanism are implemented to solve the critical section problem.

Instructions

The numbers in the text file should be generated randomly. After each update (see sorting) the values of the elements should be written in the text file. No storage in the memory is allowed for the list elements. The simulations end when all elements in the list of elements are sorted.
To lock the files consider the `fcntl()` function. To protect the critical section consider `pthread_mutex_lock()` and `pthread_mutex_unlock()` functions, respectively. The stopping criteria should run in a separate thread.

Notes

- The number of elements (n) should be provided as command line argument, while the number of threads (M) should be read from the console.
- Each step in the simulation process should be traceable. Printing is to be provided.

Rubric

Task	**Points**
Input/Error handling	3
Result printing	2
Problem	10