Programming Assignment #1: Multithreading
CSC 4103: Operating Systems, Fall 2021

Due Date: October 20, Wednesday (by 11:59 PM)
Total Points: 10

Instructions:

(1) Compile and test-run your code on the classes server.
(2) Include a README file with your code.
(3) Submit your work and verify your submission on the classes server.
(4) Submit a tar ball of your code on Moodle for backup.

Late submissions will be penalized at the rate of 10% per day late and no more than 3 days late.

Objective

To learn the use of POSIX Pthreads or Java Threads

Background

Modern operating systems provide features for a process to use multiple threads for parallel operations. In the class, we have learned the concepts associated with multithreaded computer systems, such as multithreading models. There are various thread libraries. POSIX Pthreads and Java Threads are widely used for creating and managing threads. The textbook (Threads, Chapter 4) gives examples of multithreaded programs using these libraries. Parallel threads need to deal with the race condition problem. We have also introduced several synchronization methods, such as spinlock and semaphore, for coordinating parallel threads. The textbook (Process Synchronization, Chapter 6) contains more details.

Programming Language

C/C++ or Java

Programming Task

You need to implement the **Project – Sudoku Solution Validator** given in the textbook (see the Project 1, Page 197 in the 9th edition). Your program will need to validate whether a given Sudoku solution is correct or not. In your implementation, your program should create **at least 3 concurrent threads, each validates a part of the Sudoku solution**. For solving the race condition problem when generating the output, your program will need to use **at least one synchronization tool, spinlock or semaphore**. Please read the description in the textbook, and use either POSIX Pthreads or Java threads to implement this program.

Input: Your program should accept an input file that describes a Sudoku solution. An example input file containing a valid Sudoku solution is available on the course webpage. Your program should be tested using the supplied input file.

For your program, an example command line is as follows. The argument specifies the name of the file containing the solution (./solution.txt).

```
$ ./sudoku-validator ./solution.txt
```

Output: Your program should generate the validation result of each row, column, and subgrid. Each line of the output should contain the following information:

(1) The thread ID of the thread that performed the validation.
(2) The row, column, or subgrid that was examined.
(3) The validation result of the row, column, or subgrid (valid or not).

Below is an example output that shows the first three lines of the validation results for Row #2, Column #5, and the top-left Subgrid (Row 1-3 and Column 1-3).

```
Thread 2, Column 5, Invalid
Thread 1, Row 2, Valid
Thread 3, Subgrid R123-C123, Valid
```

NOTE:

(1) Compile and test-run your code on the classes.csc.lsu.edu server. **Your code should be compilable and runnable in the Linux environment of the classes server.** Windows code will NOT be accepted.
(2) Include an README file to clearly explain how to compile and run your code, such as the command, argument, expected input and output, etc. The README file should also include your full name, LSU ID, and email address.
(3) Submit your work by the deadline on both classes server and Moodle. Verify your submission, which will display your submission date/time. Note that if you make multiple submissions, the prior submission will be overwritten. The final submission date/time will be the latest one.
(4) Late submissions will be penalized by 10% per day late and no more than 3 days late.

Submitting Your Work

You need to submit a complete package on **both classes server and Moodle.** The grader will grade your code on the classes server. Your submission on Moodle will be used as a backup. If you made multiple submissions, the final submission date/time will be the latest one.

All files you submit must have a **header** with the following (enclosed in comment lines):

Name:	Your Name (Last, First)
Email:	Your LSU email
Project:	PA-1 (Multithreading)
Instructor:	Feng Chen
Class:	cs4103-sp21
Login ID:	cs4103xx

(1) Submission to the classes server

You need to use the server "classes.csc.lsu.edu" to work on the assignment. You can login to your account in the server using SSH. Create a directory **prog1** (by typing **mkdir prog1)** in your home directory in which you create your program or source code.

Make sure that you are in the **prog1** directory while submitting your program. It is suggested to pack all the files in a tar tall (or a Zip file) before submission. Submit your assignment to the grader by typing the following command:

```
$ p_copy 1
```

This command copies everything in your prog1 directory to the grader's account.

Verify that all the files have been submitted successfully:

```
$ verify 1
```

(2) Submission on Moodle

You should create a tar ball (or a Zip file) of your complete code and submit it on Moodle.

Note: The submission on Moodle will be only used as a backup. You still need to submit on the classes server as instructed above.