Worksheet 5
Pthreads Synchronization

In this lab, you will learn the basics of multi-threaded programming, and synchronizing multiple threads using locks and condition variables. You will use the `pthreads` thread API in this assignment.

Before you begin Please familiarize yourself with the `pthreads` API thoroughly. Many helpful tutorials and sample programs are available online. Practice writing simple programs with multiple threads, using locks and condition variables for synchronization across threads. Some example programs for you to write are:

I. Write a program that has a counter as a global variable. Spawn 10 threads in the program, and let each thread increment the counter 1000 times in a loop. Print the final value of the counter after all the threads finish—the expected value of the counter is 10000. Run this program first without using locking across threads, and observe the incorrect updation of the counter due to race conditions (the final value will be slightly less than 10000). Next, use locks when accessing the shared counter and verify that the counter is now updated correctly.

II. Write a program with N threads. Thread i must print number i in a continuous loop. Without any synchronization between the threads, the threads will print their numbers in any order. Now, add synchronization to your code such that the numbers are printed in the order 1, 2, ..., N, 1, 2, ..., N, and so on. You may want to start with N = 2 and then move on to larger values of N.

Submission instructions

- You can use C and C++ for this worksheet.
- Place these files and any other files you wish to submit in your submission directory, with the directory name being your roll number (say, 12345678).
- Tar and gzip the directory using the command `tar -zcvf 12345678.tar.gz 12345678` to produce a single compressed file of your submission directory. Submit this tar gzipped file on blackboard.