**Lab Session 4**

Submission deadline: March 1, 11:59pm

Please submit your lab results through CatCourses, including Makefile and a short report (up to one page).

For how to compile a pthreaded program and other information related to pthread, you may find this website useful for this lab session (https://computing.llnl.gov/tutorials/pthreads/).

**1. Monte Carlo Estimation of PI**

Implement a Monte Carlo estimation of PI based on Pthreads (see Slides 19-22) and output the value of PI. Change the number of threads (4, 8, 16, and 32) and measure the execution time of your program.

Note: sample\_points\_per\_thread must be sufficiently big. You can use 10,000.

PI =4 \* (total\_hits) / (number\_of\_threads \* sample\_points\_per\_thread)

**2. Bank account simulation**

Download the bank account simulation code (bankAccount.h and bankAccount.c) from CatCourses. Build and run several times, for each of 2, 4, 8, 16 … threads, and check the output.

Read the code and identify the problem of the code. Fix the problem and explain your method.

**3. A read-write lock based on Pthread mutex and condition variables**

In the lecture, I give some hints on how to implement a read-write lock. Implement a read-write lock based on Pthreads mutex and condition variables.

This website (see test03.c)

(<https://www.ibm.com/support/knowledgecenter/en/ssw_aix_72/com.ibm.aix.genprogc/using_readwrite_locks.htm>)

gives an example of using pthread\_rwlock. Replace pthread\_rwlock with your read-write lock. Compare the output of your program and the example program to verify the execution correctness of your program. Change the number of threads and ensure there is no deadlock in your program.