Programming Assignment 1: Producer-Consumer Problem
CSC 4103: Operating Systems, Spring 2016

Due Date: March 6th, Sunday (by 11:59 PM)
Total Points: 10 points

Instructions: Compile and test-run your code on the classes server. Submit your work as instructed and verify your submission. The verify command will display your submission date/time. Include your name and classes account in all source code files. 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 Threads and synchronization control.

Background

Modern operating systems provide features for a process to contain multiple threads of control and allow them to cooperate together. Manipulating shared data requires careful synchronization. Two widely used synchronization mechanisms are mutex lock (spinlock) and semaphore.

Programming Task

In the classic producer-consumer problem, a producer thread inserts items into a buffer, and a consumer thread consumes items from the buffer simultaneously. In this project, you are required to program in C or Java and use POSIX Pthreads or Java Threads to solve an N-producer-M-consumer problem by generating N producer threads and M consumer threads to simultaneously access the buffer.

In specific, your program should satisfy the following requirements:

(1) Your program should accept 4 input parameters, namely "the number of producer threads", "the number of consumer threads", "the size of the buffer", and "the number of items to be produced". For example, the following command starts your program with 4 producer threads, 5 consumer threads, a buffer of 10 items, and generate 1000 items in total.

```
$ ./producer-consumer  4  5  10 1000
```

(2) Similar to the producer-consumer problem, a buffer is maintained in your program and shared among all producers and consumers. Each item in buffer is an integer.

(3) Each producer thread generates a random integer each time, inserts it into the buffer, increments a counter, **num_produced**, which is shared by all producers to track how many items being created, and records the time and the item being produced in an event log. The event log should record the events in the order of timestamps.

(4) Each consumer thread consumes the items of the buffer in the First-In-First-Out (FIFO) order, meaning that if item A is inserted into the buffer earlier than B, item A should be consumed earlier than B, increments a counter, **num_consumed**, which is shared by all consumers to track how many blocks are consumed, and record the time and the item being consumed in an event log. The event log should record the events in the order of timestamps.

(5) Use either semaphore or mutex lock (spinlock) to synchronize threads working on the shared resources in your program.

(6) When the specified number of items are produced *and* consumed, your program should exit and generate the following output: (i) print the total number of produced items, (ii) print the total number of consumed items, and (iii) output the even log files, named "producer-event.log" and "consumder-event.log". Each record of the log file should be formatted as the following:

```
<Timestamp (in nanoseconds)> <Thread type ("Producer" or "Consumer")>
<Thread ID> <Buffer Entry Index> <Item>.
```

Basically, each record logs when and which thread produced an item into or consumed an item from an entry in the shared buffer.

*****How to Submit Your Work*****

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

Name:	Your Name (Last, First)
Project:	PA-1 (Producer-Consumer)
File:	filename
Instructor:	Feng Chen
Class:	cs4103-sp16
LogonID:	cs4103xx

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. Submit your assignment to the grader by typing the following command:

~cs4103_chf/bin/p_copy 1

This command copies everything in your prog1 directory to the grader's account. Check whether if all required files have been submitted successfully:

~cs4103_chf/bin/verify 1