CSE 451
Problem Set #1

Due: 9:00pm, Tuesday, April 29, 2014
To be done INDIVIDUALLY

For this assignment, use Mesa-style locks and condition variables and follow best practices in terms of ensuring correctness.

1. With data parallel programming, the computation executes in parallel across a data set, with each thread operating on a different partition of the data. Once all threads have completed their work, they can safely use each other's results in the next (data parallel) step in the algorithm. MapReduce is an example of data parallel programming, but there are many other systems with the same structure.

For this to work, we need a way to efficiently check whether all \n\ threads have finished a given step. This is called a synchronization barrier. A synchronization barrier has one operation, \checkin\. A thread calls \checkin\ when it has completed its work; no thread may return from \checkin\ until all \n\ threads have checked in. Once all threads have checked in, it is safe for the threads to use the results produced by other threads in the previous step.

a. Implement a single-use synchronization barrier. The barrier is initialized with the number of threads that will check in. You may assume that each of the \n\ threads calls \checkin\ exactly once.

b. Most data parallel programs execute a series of steps, with a barrier after each step. Explain what can happen if the program tries to reuse the same barrier after each step, given your implementation.

Extra credit: implement a reusable synchronization barrier, one that is safe to use after each step.