Homework: Synchronization-Application

1. Explain why spinlocks are not appropriate for single-processor systems yet are often used in multiprocessor systems.

**Spinlocks can be taxing and inefficient for single-processor systems, but can be effective in multiprocessor systems where there is lower contention**

1. Assume that a system has multiple processing cores. For each of the following scenarios, describe which is a better locking mechanism—a spinlock or a mutex lock where waiting processes sleep while waiting for the lock to become available:
   1. **The lock is to be held for a short duration.**
   2. The lock is to be held for a long duration.
   3. A thread may be put to sleep while holding the lock.
2. Programming Project (Java):

A university computer science department has a teaching assistant (TA) who helps undergraduate students with their programming assignments during regular office hours. The TA's office is rather small and has room for only one desk with a chair and computer. There are three chairs in the hallway outside the office where students can sit and wait if the TA is currently helping another student. When there are no students who need help during office hours, the TA sits at the desk and takes a nap. If a student arrives during office hours and finds the TA sleeping, the student must awaken the TA to ask for help. If a student arrives and finds the TA currently helping another student, the student sits on one of the chairs in the hallway and waits. If no chairs are available, the student will come back at a later time. When the TA finishes helping a student, the TA must check to see if there are students waiting for help in the hallway. If so, the TA must help each of these students in turn. If no students are present, the TA may return to napping.

The Java version of this project may be completed using Java synchronization tools. Synchronization may depend on either (a) monitors using synchronized/wait()/notify() or (b) semaphores and reentrant locks.

To begin, please first create *n* students where each student will run as a separate thread. The TA will run as a separate thread as well. Student threads will alternate between programming for a period of time and seeking help from the TA.

Please **use** the given source codes on GitHub as your start codes. <https://github.com/yilianz/CSCI411/tree/master/hw2sourcecode>