CS 347 QUIZ 4 (August 11, 2016)

Name: _____ ***Roll No.*** _____

1. Consider an application with two threads, and let's call them thrA and thrB. The application logic requires that thrA should execute the function fa1() first, after which thrB should execute fb(), after which thrA should execute fa2(). This cycle fa1(), fb(), fa2() should repeat indefinitely. You need not worry about what these functions themselves do. Your job is to ensure synchronization between the threads, so that this order of execution of the functions by the threads is achieved. Your solution must use only **semaphores** and no other synchronization mechanism. It is possible to solve this problem using only **three** semaphores, and solutions using more than three semaphores will not get credit.

Write down the semaphore variables you need and their initializations here:

Fill in the pseudocode for the threads here:

thrA

```
while(1) {
                                        //write something here
fa1();
                                        //write something here
fa2();
                                        //write something here
}
```

thrB

```
while(1) {
                                        //write something here
fb();
                                        //write something here

}
```

2. Consider the readers and writers problem as discussed in class. Several reader and writer processes wish to access a critical section. Becasue readers do not modify the critical section, multiple readers can access the critical section concurrently. However, a writer can access the critical section only when no other reader or writer is concurrently accessing it. We wish to implement locking/synchronization between readers and writers, while giving **preference to writers**, where no waiting writer should be kept waiting for longer than necessary. For example, suppose reader process R1 is actively reading. And a writer process W1 and reader process R2 arrive while R1 is reading. While it might be fine to allow R2 in, this could prolong the waiting time of W1beyond the absolute minimum of waiting until R1 finishes. Therefore, if we want writer preference, R2 should not be allowed before W1. Your goal is to write down pseudocode for read lock, read unlock, write lock, and write unlock functions that the processes should call, in order to realize read/write locks with writer preference. You must use only simple locks/mutexes and conditional variables in your solution. Please pick sensible names for your variables so that your solution is readable.

Read lock

Read unlock

Write lock

Write unlock