Mostafa Vahidi

CS 4310.01 Program 2

Prof. Diaz 10/19/18

CS 4310.01 Program 2 Answers

**1 Getting Ready:**

(1) Does the code implement mesa style or hoare style monitors? Justify your answer.

This code implements mesa style monitors. Because when a wake() or wakeAll() is called to wake another thread, the woken thread is simply put on the ready list, and it is the responsibility of the woken thread to reacquire the lock which is taken care of in sleep().

(2) In class, we learn that the value of semaphore S is the number of units of the resource that are currently available. What are the possible values of the semaphore waiter in Condition.java?

Semaphore waiter is the number of waiting threads that are available. Number of waiters.

(3) In sleep(), why is the lock released before waiter.P() and acquired after? When will conditionLock.acquire() be called?

It is released right before waiter.p() because the waiter semaphore will acquire it when it sleeps. Thus, the lock needs to be available for that operation.

(4) Why is waitQueue needed? Can we just use a single semaphore?

There is a waitQueue because some other thread can acquire the lock and change data structures, before the woken thread gets a chance to run. And also makes it a lot easier to implement.

(5) In sleep() and wake(), a shared object waitQueue will be accessed. What prevents it being accessed by two threads at the same time?

The assertion that the condition lock is held by the current thread at the very top of sleep() and wake() prevents the waitQueue to be accessed by two threads at the same time.