CS 2200 Homework 9

Fall 2018

Instructions:

- Please print a **double-sided** copy of the assignment and hand write your answers. No electronic submissions are allowed. **There will be a 90 point penalty if you do not.**
- This is an individual assignment. You may discuss concepts but not the answers.
- Due Date: **11/07/18 – 6:00 PM** in recitation. Bring your BuzzCard. Show up on time.

Name:_____ GT Username:_____ Section:_____

1. Refer to the code below. Please fill in the blanks using methods: **thread_mutex_lock(), thread_mutex_unlock(), thread_cond_signal(), thread_cond_wait()** to fix the code. ***Note:*** refer to documentation for proper function signatures! Assume buffer, lock, not_fill, and not_empty are all initialized.

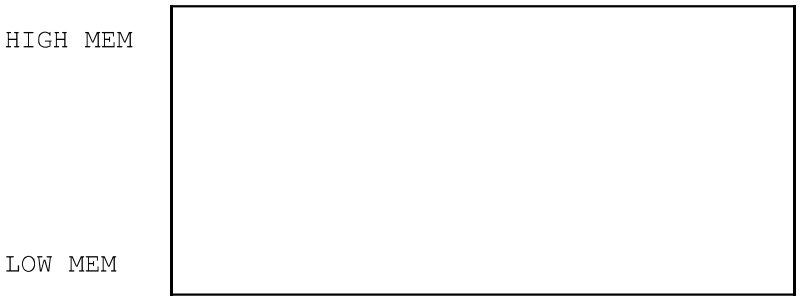
```
frame buffer[MAX_SIZE];
int buffer_size = 0;
cond_var not_full;
cond_var not_empty;
mutex_lock lock;

int consumer(){
        _____
        _____
              _____
        /*Assume code here to consume frame from buffer without thread
        locks/signals*/
        Buffer_size--;
        _____
        _____
}

int producer(){
        _____
        _____
              _____
        /*Assume code here to add frame to buffer without thread locks/signals*/
        Buffer_size++;
        _____
        _____
}
```

   a. Ignoring the blanks and focusing on remaining content, why would there be a problem if we were to issue a thread for both the producer and consumer methods when both are trying access a shared buffer data structure (frame_buffer and buffer_size)

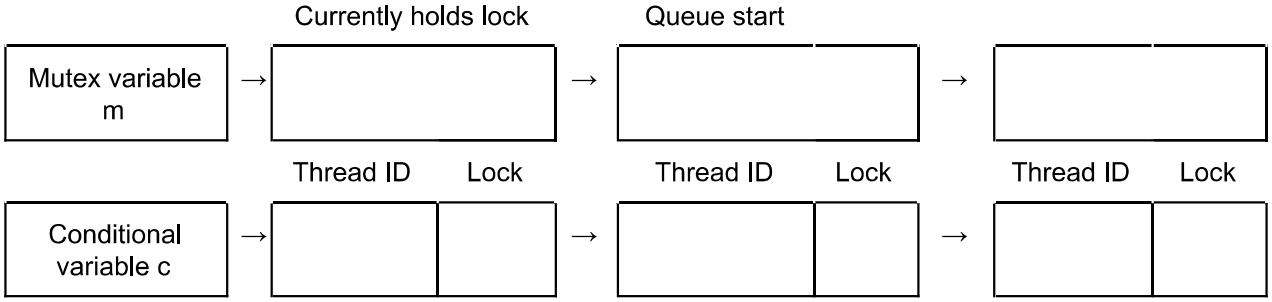
2. Please draw out a diagram of the memory footprint for a multi-threaded process with 4 threads in the rectangle below:



3. Given a mutex lock *m*, and a condition variable *c*, the following events happen in the order of occurrence shown below:
    - T1 executes mutex-lock(m); assume no one has the lock so T1 will win
    - T2 executes mutex-lock(m)
    - T1 executes cond-wait(c, m)
    - T3 executes mutex-lock(m)
    - T2 executes cond-signal(c, m)

    Show the waiting queues for *m* and *c* in the following scenarios:
    - Note: Clearly, show which thread is currently holding the mutex lock, and which threads are in the waiting queue for the lock.
    - Note: if a thread is waiting on a condition variable, you should also show the mutex lock it needs for resuming execution.

a. State of waiting queues before T2 executes cond-signal



b. State of waiting queues after T2 executes cond-signal