CSE 330 - Operating Systems
Summer 2019
Project #2
Due: August 1st 2019, 23:59 PM

Strictly Individual Projects

Implementing Semaphores

Using the threads, you have implemented, implement semaphores. Since the threads are non-preemptive, you do not need to ensure atomicity of the semaphores (they are already atomic).

Implement the following: (in a file called sem.h)

1. **Semaphore data structure:** A value field and a queue of TCBs.

2. **InitSem(semaphore, value):** Initializes the value field with the specified value.

3. **P(semaphore):** The P routine decrements the semaphore, and if the value is less than zero then blocks the process in the queue associated with the semaphore.

4. **V(semaphore):** The V routine increments the semaphore, and if the value is 0 or negative, then takes a PCB out of the semaphore queue and puts it into the run queue. Note: **The V routine also "*yields*" to the next runnable process.** //this is important.

5. Implement a set of thread to test the semaphores. You can choose one of two methods.
   Method 1:
   Each thread is an infinite loop, and has a critical section using a mutex semaphore. The thread gets into the CS, prints values of global and local variables (like proj1), sleeps and exists the CS. Then it prints a message and sleeps and then repeats.
   Method 2:
   Write a producer consumer solution, using the code given in class (see notes). Run 2 producers and 2 consumers. You will have to work out some details. If you choose to do this you really should do method 1 first. *Also, doing this will make it easier for you to do project 3.*

If your threads work, but fails when you add semaphores …specially if prod/cons causes trouble: YOU HAVE BUGS IN YOUR QUEUES.

Submission and Grading:

Your project must consist of 4 files

1. TCB.h (uses ucontext.h)
2. q.h (includes TCB.h)
3. threads.h (includes q.h)
4. sem.h (includes threads.h)
5. proj-2.c (includes threads.h) – must contain your name(s) in comments @ beginning (make sure the compile command, "gcc proj-2.c" does the correct compilation).

All 5 files are to be ZIPPED into one zip or gzip file and uploaded in Blackboard. The name of this file should reflect the name of the student (abbreviated, do not make it too long).

Note: Grading is on Ubuntu. It is in your interest to make sure the program compiles and runs in the target test platform.