Grading Rubric: COS 318 Project 3

Fall 2012

This project is worth a total of **10 points**, plus 2 points extra credit.

First of all, please make sure your code can be compiled successfully using the given Makefile with the –Werror flag on.

A. Preemptively scheduled processes. 2 pts.

- A01: Processes can be preempted. 1 pt.
- A02: Preemption preserves all registers. 1 pt.

B. Blocking sleep. 2 pts.

- B01: Processes block on sleep. 0.5 pts.
- B02: Threads block on do_sleep. 0.5 pts.
- B03: Scheduler survives when all processes are sleeping. 0.5 pts.
- B04: Sleep counts time correctly. Specially, if your code passes the test_blocksleep case with some acceptable error, it should be fine. 0.5 pts.

C. Condition variables. 2 pts.

- C01: Condition signal wakes one. 0.5 pts.
- C02: Condition broadcast wakes all. 0.5 pts.
- C03: Producer/consumer works when building a queue using your CV.signal. 0.5 pts.
- C04: Producer/consumer works when building a queue using your CV.broadcast. 0.5 pts.

D. Semaphores. 2 pts.

- D01: Producer/consumer works when building a queue using your semaphores. 1 pt.
- D02: Semaphores are fair. Specially, if your code works as we discussed in the design review, it should be fine. 1pt.

E. Barriers. 2 pts.

- E01: Barriers successfully wait for all processes. 1pt.
- E02: Barriers are automatically renewed. Specially, if your code works as we discussed in the design review, it should be fine. 1 pt.

X (extra credit). Prioritized task scheduling. 1 pt.

- X01: Priority with preemption. Tasks should be scheduled according to their priorities, and without starvation. Specially, if you have task A with priority 100, task B with priority 200, task B should be scheduled roughly twice as many as task A. We accept a fairly wide error here, say, ±15%. Besides, if you have task A with priority 100000, task B with priority 1, you should make sure task B somehow will get a chance to run. 0.5 pts.
- X02: Priority with yield. Similar to X01, except that tasks would yield at some points. 0.5 pts.

Y (extra credit). Deadlock detection. 1 pt.

- Y01: Deadlock sensibility. Your code should detect the deadlock when it happens without false positive. 0.5 pts.
- Y02: Deadlock specificity. Specially, it's also treated as a deadlock if a task acquires a lock that has already been held by it. 0.5 pts.

Additionally, we reserve the right to remove as much as 1 point for submissions which are extremely confusing, obfuscated or overcomplicated. Please write simple, readable code with comments. Don't forget to attach a brief readme in text format.

Recall that this class has a firm late submission policy, which is detailed on the COS 318 website.