Laboratory Exercise 3

cpe 453 Winter 2026

Due by 11:59:59pm, Monday, February 9th.
The Written Exercises (problems) are to be done individually.
Due to the midterm being on the following Wednesday, late days are limited to one on this assignment.

Problems

1. In Minix (or any other Unix), if user 2 links to a file owned by user 1, then user 1 removes the file, what happens when user 2 tries to read the file? (Tanenbaum and Woodhull, Ch. 1, Ex. 15)

2. Under what circumstances is multiprogramming likely to increase CPU utilization? Why?

3. Suppose a computer can execute 1 billion instructions/sec and that a system call takes 1000 instructions, including the trap and all the context switching. How many system calls can the computer execute per second and still have half the CPU capacity for running application code? (T&W 1-21)

4. What is a *race condition?* What are the symptoms of a race condition?(T&W 2-9)

5. Does the busy waiting solution using the *turn* variable (Fig. 2-10 in T&W) work when the two processes are running on a shared-memory multiprocessor, that is, two CPUs, sharing a common memory? (T&W, 2-13)

6. Describe how an operating system that can disable interrupts could implement semaphores. That is, what steps would have to happen in which order to implement the semaphore operations safely. (T&W, 2-10)

7. Round robin schedulers normally maintain a list of all runnable processes, with each process occurring exactly once in the list. What would happen if a process occurred twice in the list? Can you think of any reason for allowing this? (T&W, 2-25) (And what is the reason. "Yes" or "no" would not be considered a sufficient answer.)

8. Five batch jobs, *A* through *E*, arrive at a computer center, in alphabetical order, at almost the same time. They have estimated running times of 10, 3, 4, 7, and 6 seconds respectively. Their (externally determined) priorities are 3, 5, 2, 1, and 4, respectively, with 5 being the highest priority. For each of the following scheduling algorithms, determine the time at which each job completes and the mean process turnaround time. Assume a 1 second quantum and ignore process switching overhead. (Modified from T&W, 2-28)

   (a) Round robin.

   (b) Priority scheduling.

   (c) First-come, first served (given that they arrive in alphabetical order).

   (d) Shortest job first.

for (a), assume that the system is multiprogrammed, and that each job gets its fair share of the CPU. For (b)–(d) assume that only one job at a time runs, and each job runs until it finished. All jobs are completely CPU bound.

9. Re-do problem 8a with the modification that job *D* is IO bound. After each 500ms it is allowed to run, it blocks for an IO operation that takes 1s to complete. The IO processing itself doesn't take any noticeable time. Assume that jobs moving from the blocked state to the ready state are placed at the end of the run queue. If a blocked job becomes runnable at the same time a running process's quantum is up, the formerly blocked job is placed back on the queue ahead of the other one.

10. A CPU-bound process running on CTSS needs 30 quanta to complete. How many times must it be swapped in, including the first time (before it has run at all)? Assume that there are always other runnable jobs and that the number of priority classes is unlimited. (T&W, 2-29)

What to turn in

For the Written Problems: individually written solutions to the problems according to the guidelines set forth in the syllabus.

Since paper won't work this quarter, submit via `handin` to the `lab03` subdirectory of the `pn-cs453` account, as a pdf or text file.[1]

[1] I only have LibreOffice, and you don't want to see what it'll do to your nicely crafted Word document. . .