CSC 369: Tutorial Exercise Scheduling
Week 5

UtorID:		Name:	
UtorID:		Name:	
UtorID:		Name:	

Question 1:

We have been benchmarking the programs being run on a server and have learned that we can easily classify all of the programs into three categories. Furthermore, we will know which category each program belongs to when it begins to execute:

- Interactive (I/O-bound) jobs with a short average CPU burst time of SBurst.
- Interactive (I/O-bound) jobs with a long average CPU burst time of LBurst where LBurst is approximately 3 times SBurst.
- CPU-bound jobs where the time to complete the job varies widely.

One of the general goals of the scheduling algorithm is to minimize overhead by minimizing the number of context switches. A second goal is to prevent starvation.

a) Given the categories above and your understanding of the example CPU scheduling examples we investigated, list one additional scheduling goal our algorithm should satisfy. Explain how this goal will affect both I/O and CPU bound jobs.

Minimize response time. It should give priority to jobs returning from I/O and may mean CPU bound jobs are delayed.

b) Describe how a scheduling algorithm might use priorities to satisfy the above goals.

Each job category can be treated separately, and the two interactive classes given higher priority than the CPU bound jobs.

c) Describe the factors to consider when choosing an appropriate time quantum for a pre-emptive scheduler. Is it appropriate to have different sized quanta for different priorities?

To minimize the number of context switches, the quantum should be no smaller than SBurst. It would also make sense to have one queue with a quantum of SBurst and one with a quantum of LBurst

d) Since we have so much information about a job when it arrives, can we assign it a priority and never change it, or do we still need dynamically changing priorities? Explain.

If the interactive jobs always have higher priority than the CPU bound jobs we will see starvation.

Question 2:

An OS uses a multi-level feedback scheduler with 2 levels. New and returning processes start at level 0, which uses round robin scheduling with a quantum of 2. A process that uses its entire quantum at level 0 gets moved to level 1, which uses first-come-first-served scheduling. The scheduler always chooses a process from the lowest-numbered non-empty level. Admission of a new process or a process returning from IO does not force a scheduling decision; those processes are placed on the level 0 queue to wait until a scheduling decision is to be made.

Initially, there is one process, P0. New processes P1 and P6 are created at times 1 and 6 respectively. Those are the only three processes you need to schedule. Each process has a CPU burst of 5 time units, an IO burst of 3 time units, and a CPU burst of 1 time unit. During an IO burst, a process is blocked (on a wait queue); it does not require the processor.

Assume context switches take no time. Fill in the timeline below for each process. Note when a process is **New**, when it has the **CPU**, when it loses the CPU due to a **Preempt**, what queue it is on (**L0**, **L1**, **IO**), and when it Exits. The first 5 time units have been filled in for you.

Time	0	1	2	3	4	5	6
P0	**CPU**	**CPU**	**Preempt, L1**	**L1**	**CPU**	CPU	CPU
P1		**New, L0**	**CPU**	**CPU**	**Preempt, L1**	L1	L1
P6							New, L0
Time	7	8	9	10	11	12	13
P0	IO	IO	IO	L0	L0	CPU	Exit
P1	L1	L1	CPU	CPU	CPU	IO	IO
P6	CPU	CPU	Preempt, L1	L1	L1	L1	CPU
Time	14	15	16	17	18	19	20
P0							
P1	IO	L0	CPU	Exit			
P6	CPU	CPU	IO	IO	IO	CPU	Exit

For your solution above, what is the total wait time (time spent ready but not executing) for all processes?

16

For your solution above, what is the average response time (time spent waiting for the processes's first time slice)?

1 + 1 + 0 / 3