This practice midterm is DEFINITELY MUCH TOO LONG. The real midterm will be shorter.

HOMEWORK (but NOT lab) problems are also possible.

PLEASE WRITE YOUR NAME AND ANSWERS ON ALL 5 QUESTION SHEETS. You may use the backs of the question sheets to continue your answers. You may also use the blank sheets after question 5 to further continue your answers. Scrap paper is available. GOOD LUCK!!

1A. Draw the process state diagram from the notes. This diagram contains nodes (i.e. circles) labeled with the possible states for a process. It also contains arcs (i.e. arrows) showing the various state transitions possible. Label the nodes and arcs (for example one node should be labeled running and one arc should be labeled schedule).

1B. Some of the arcs are associated with medium term scheduling. As you know, this scheduling is done on a time scale of seconds or minutes. Give an example of a medium term scheduling event and tell why it is not done on a faster time scale (say every 0.1 seconds).

1C. Shortest Job First and Preemptive Shortest First have a very favorable property. What is it? Why are they impractical for general purpose operating systems? Even if they were practical, they also have an unfavorable property that would make them unsuitable purpose systems. What is it? What would you add to these scheduling algorithms to alleviate this unfavorable property.

1D. Some arcs change the NUMBER of processes in the system. Which arcs are these and what system calls do they correspond to.
2A. Define, but do **NOT** solve the Readers Writers problem.

2B. Some systems simply treat the readers writers problems as critical section problems and hence the implementation simply use P and V. What requirement of the Readers Writers problem does this implementation not satisfy?
3. Consider the following set of processes, each of which performs no I/O (i.e., no process blocks). All times are in milliseconds and context switching takes zero time. The CPU time (column TWO) is the total time required for the process. The creation time (column THREE) is the time when the process is created. So P1 is created when the problem begins and P3 is created 3 milliseconds later. A process created at time T is eligible to be run at T. For example, if a scheduling decision is made at time 1, P1 and P2 (but not P3) would be eligible.

<table>
<thead>
<tr>
<th>Process</th>
<th>CPU Time</th>
<th>Creation Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>P2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>P3</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

A. Assume RR (Round Robin) scheduling with a quantum of 2 milliseconds. At what time does each process finish. Show your work.

B. Do the same for RR with a quantum of 100 milliseconds.

C. Repeat A and B if context switching to a DIFFERENT process takes 0.1 milliseconds.

D. Repeat C for SJF.

E. Repeat D for PSJF.
4A. Draw a resource allocation graph (also called a reusable resource graph) that represents a deadlock state. Your graph must contain at least two resources and at least two tasks. Each resource must contain 3 units.

4B. Of course when execution started there were no arcs in the resource allocation graph. Give a scenario starting from this initial condition of no arcs and ending in the graph you gave for 4A. That is, tell what requests and releases occur and in what order. For this part you should assume a naive (i.e., optimistic) resource manager that grants every request as soon as it can.

4C. Recall that the Banker’s algorithm for resource allocation never enters a deadlocked state like the one in part A. Consider the scenario you gave for part B and tell at what point the Banker’s algorithm will depart from the scenario. That is, indicate the first request that the Banker’s algorithm will refuse to grant that the optimistic manager did grant. Don’t forget that whenever you deal with Banker’s algorithm or (un)safe states, the claims of each process are important.

4D. Explain in detail why the Banker’s algorithm refused to grant the request you noted in part C.
5A. Draw a reusable resource graph that represents an **unsafe** state that is **NOT** deadlocked. (Hopefully you remembered that the claims of the processes are important).

5B. Process A is already written and requests the printer, the plotter, the tape drive, and the robotic arm, in that order. You need to develop processes B and C. Both need the printer and the arm, B needs the tape drive, and C needs the plotter. What order should the requests be made? Why.