Important: This is an open book test. You can use any books, notes, or paper but no electronic device. *Do not log into the computer during the test, or use any electronic or communications device. Your cell phones must be on silent mode*. Any other device with an ON-OFF switch should have its switch in the OFF position. Any calculations and rough work can be done on the back side of the test pages. You will lose five points for not writing your name.

1. [6 pt] What is the difference between `clone(2)` and `fork(2)` system calls in Linux?

2. [6 pt] What is a spin lock? Why is a spin lock not useful in a uniprocessor environment?

3. [6 pt] After assigning a bonus value to compute dynamic priority of a process in the Linux kernel, we restrict the value of dynamic priority to a range $[100, 139]$. Why do we have to do that?

4. [6 pt] Why is the resource allocation graph required to be a bipartite graph?

5. [8 pt] What is the difference between deadlock avoidance and deadlock prevention? Can you prevent the circular wait condition by imposing a partial order on the resource requests?

6. [10 pt] Assume a system with four resource types, $C = \langle 7, 6, 6, 5 \rangle$ (this is the total number of resources in the system, and not what is currently available), and the maximum claim table shown below.

Process	R_0	R_1	R_2	R_3
p_0	3	0	3	1
p_1	2	3	0	4
p_2	3	0	0	4
p_3	4	0	2	3
p_4	0	1	2	0

The resource allocator is considering allocating resources according to the following table:

Process	R_0	R_1	R_2	R_3
p_0	2	0	3	1
p_1	2	0	0	0
p_2	3	0	0	0
p_3	0	0	1	1
p_4	0	0	1	0

Run the safety algorithm on this system to determine if this state is safe. If it is safe, give the sequence in which processes can be run. If it is unsafe, enumerate the processes that may get involved in a deadlock. Show your steps from the algorithm.