MY NAME IS: _____

Question 1: Consider the following schedule of periodic tasks and the aperiodic tasks A_1, A_2, and A_3 ($A(r, e)$ indicates an arrival time of r and an execution time of e). Insert the aperiodic tasks into the schedule (top graph) and then repeat the complete schedule (bottom graph), but utilize slack stealing. In both graphs, the minor cycle size is 5 and periodic tasks are nonpreemptive.
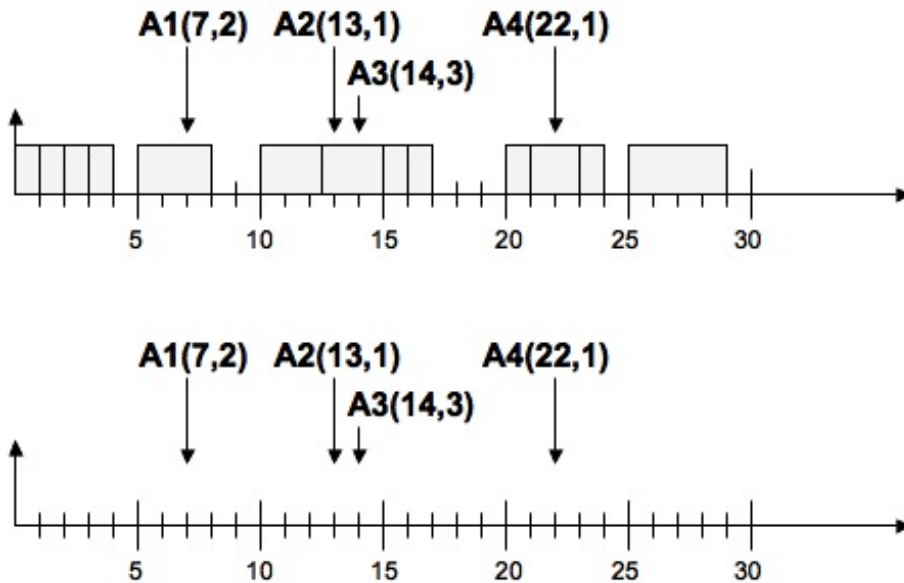
Question 2: Show by means of an example that if $d_i < P_i$, the $U \leq 1$ condition is not sufficient to ensure EDF-schedulability.

Question 3: Find the maximum value of e_1 for which the task set $\{(5,e_1),(7,4)\}$ is schedulable under the
(a) RMS algorithm (use the sufficient test),
(b) EDF algorithm (use the exact test).

Question 4: Consider the periodic tasks $T_1(2.5, 1)$, $T_2(4, 0.5)$, $T_3(5, 0.75)$, and a server $(2, 0.5)$ for aperiodic jobs. The scheduler uses RMS and the following aperiodic jobs arrive at times 3 and 7.5, respectively: $A_1(0.75)$ and $A_2(0.6)$. Give the task schedule and the budget consumption graph for a deferrable server (until time 10). What are the response times of the aperiodic jobs?

Question 5: Consider the following task set, where the numbers in parenthesis are the release time, execution time, and priority (the higher the number the higher the priority). Draw a schedule diagram for the five tasks from time 0 through 20 (use timelines below) using a) the NPCS protocol, b) priority inheritance, and c) the priority ceiling protocol. For each task, its corresponding critical section starts 1 time unit after the task started executing. Each nested critical section starts 1.5 time units after the begin of the previous critical section.

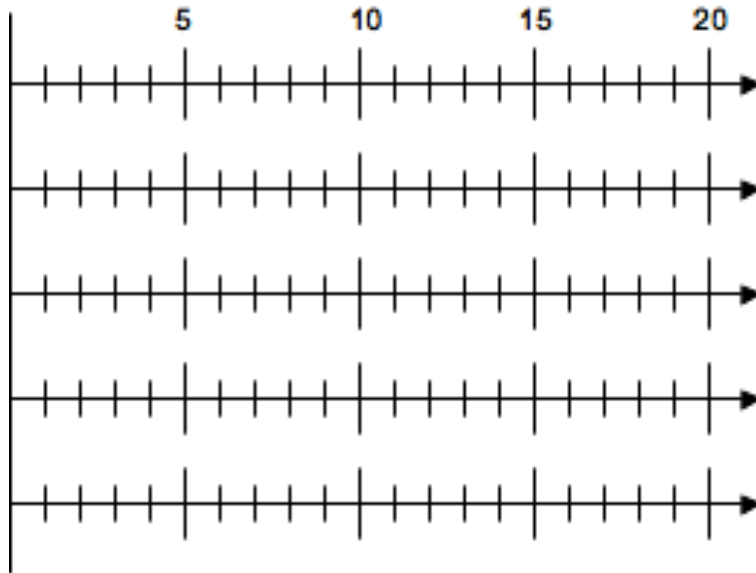| Task i	Parameters	Resources
1	(7,3,5)	requires resource R1 for 1 time unit
2	(5,3,4)	requires resource R2 for 1 time unit
3	(4,2,3)	requires no resources
4	(2,6,2)	requires resource R1 for 4 and resource R2 for 1.5 time units (nested)
5	(0,6,1)	requires resource R2 for 4 time units

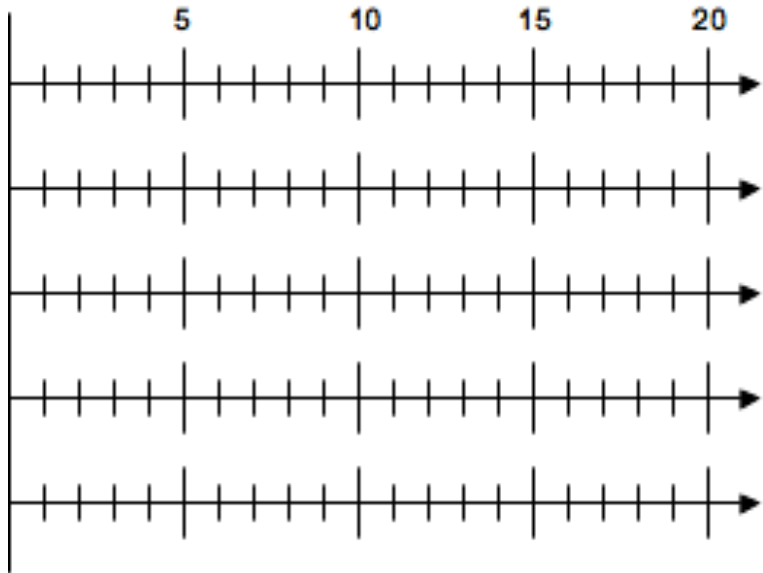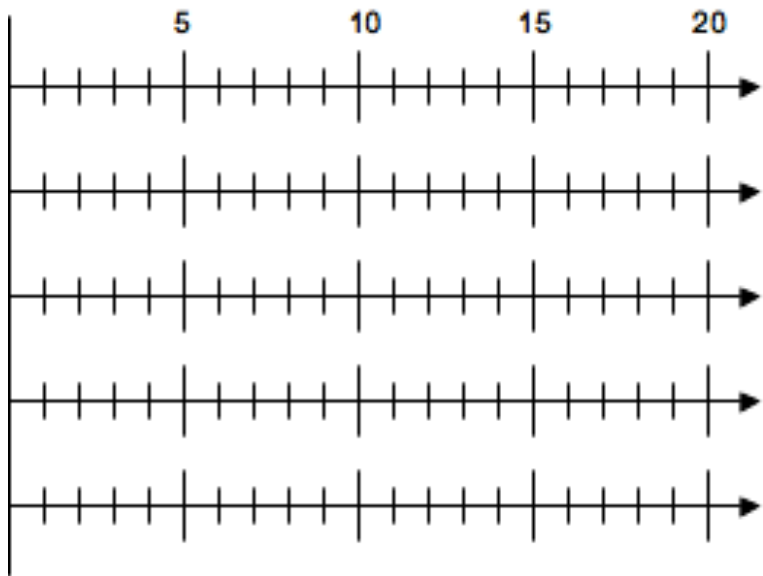Figure 1: NPCS (Non-preemptive critical section protocol).

Figure 2: Priority inheritance.

Figure 3: Priority ceiling protocol.