Tasks:

1. Add SCHED\_SHORT, OVERDUE\_SCHED\_SHORT statuses (maxim)
2. Add (define) sched\_setschedular() system call with all the wrappers (dean)
   1. **Steps for changes in the sched\_setscheduler system call(sched.c):**
      1. check the parameters are valid
         1. line 1169 in sched.c should add a condition for policy!=SCHED\_SHORT
         2. line 1179- we should define in the sched\_param sched\_priority for the short proccess and add there a condition accordingly
      2. 2. Extract the given process from it's current relevant list (e.g from the OTHER list)
      3. Update relevant fields for the SHORT policy
      4. Add the process you extracted to the relevant SHORT list
      5. Turn on the context switch flag
3. Add request time & number of processes properties somewhere to all the SCHED\_SHORT processes. (arye)
4. Go over the schedule() function before the context switch (ALL)

IDEA:

Since the short-processes behave as if they are between RT processes & OTHER processes, but use the same priority range as OTHER processes, perhaps a good implementation would be as such:

We will add another prio\_array to the runqueue. It will hold the SHORT processes.

We will also add another queue that will store all the SHORT\_OVERDUE processes.

We will go through all the scenarios in which a process can be switched, and apply the changes according to the h.w. description - “short” after “RT”, “short-overdue” after “other”.