# Online on xv6 - Scheduler

Section: C1 Time: 50 minutes

Your task is to implement a simple priority based scheduler. You are given a user program testloop.c, which simply simulates a long running job by iterating a loop a given number of times. This program takes two arguments: iteration count and priority. You will also need to implement the system call to set priority for each process.

The process with the highest priority will keep running till completion, unless another process with higher priority arrives. In that case, the higher priority process will run till completion.

- **All processes have a default priority of 300**
- **If a process consumes 10 clock ticks (timer interrupts) consecutively, decrement its priority by 1 and reset its consecutive clock ticks count to 0. Check the `trap.c` file, especially usertrap() function to better understand how to handle this counter.**
- **If the priority reaches 0, no need to decrement it further.**
- **Avoid decrementing priority for PIDs 1 and 2.**

You will also have to implement the `setpriority` system call, as used by the `testloop` program.

**Sample I/O:** See the provided sampleio.txt file

**Note:**

Set CPUS := 1 in the Makefile.

Notice how the priorities change the execution order of each process. At first, the process with the highest priority keeps running until it finishes. Then comes a process with priority 10, so it keeps running for a while. As its priority gets decremented and becomes equal to the last remaining process, they both chase each other and run in a RR order.

**Submission**:

```
git add --all
git diff HEAD > ../2105010.patch
```