xv6 (OS Assignment)
Report

Name: Sajal Asati

Original xv6 Scheduler

- Original xv6 scheduler is a round robin scheduler
- On testing the original scheduler with 3 “myprog” (test programs)   programs, it was observed that the process which started earlier finished earlier.
- Also,   there is no way to set priority or make a runnable program running   unless a program exits (because of overshooting out of time quantum given to it) .

Priority Based xv6 Scheduler

- We’ve implemented a priority based scheduler
- Priority number can be between 1 to 100. Lower the number higher will be the priority. All the processes are initially given priority 60.
- On testing the priority based scheduler with 3 “myprog” (test programs) programs, it was observed that the process which higher priority finished earlier.
- We can adjust priority of the processes with set_priority command.

For Round Robin Based xv6 Scheduler

- Two processes with same priority have been created with command myprog &

```
----first process created
$ myprog &
$ ps
NAME PID STATE PRIORITY
init 1    SLEEPING 60
sh 2     SLEEPING 60
ps 5     RUNNING 60
myprog 4        RUNNABLE 60

-----second process created
$ myprog &
$ ps
NAME PID STATE PRIORITY
init 1    SLEEPING 60
```

```
sh 2     SLEEPING 60
ps 8     RUNNING 60
myprog 4        RUNNING 60
myprog 7        RUNNABLE 60
```

----now note that each of the following ps were done in gap of 1 second.

```
$ ps
NAME PID STATE PRIORITY
init 1    SLEEPING 60
sh 2     SLEEPING 60
ps 9     RUNNING 60
myprog 4        RUNNING 60
myprog 7        RUNNABLE 60

$ ps
NAME PID STATE PRIORITY
init 1    SLEEPING 60
sh 2     SLEEPING 60
ps 10   RUNNING 60
myprog 4        RUNNING 60
myprog 7        RUNNABLE 60

$ ps
NAME PID STATE PRIORITY
init 1    SLEEPING 60
sh 2     SLEEPING 60
ps 11   RUNNING 60
myprog 4        RUNNABLE 60
myprog 7        RUNNING 60
```

Here we can clearly see (in **BOLD** font) that once the time quantum for pid=5 elapses, its state changes from Running to Runnable. The processes swap irrespective of their priorities : Round-robin Scheduling.

For Priority Based xv6 Scheduler

- Here   we simply ran the code with modified scheduling algorithm. Now, set priority of the last process to 10, so that it is executed before all others.
- Create two processes similar to previous experiment

---create first program

```
$ myprog &
$ ps
NAME PID STATE PRIORITY
init 1    SLEEPING 60
```

```
sh 2     SLEEPING 60
ps 5     RUNNING 60
myprog 4        RUNNABLE 60

---create second program
$ myprog &
$ ps
NAME PID STATE PRIORITY
init 1   SLEEPING 60
sh 2     SLEEPING 60
ps 8     RUNNING 60
myprog 4        RUNNING 60
myprog 7        RUNNABLE 60

---change the priority of one process higher than the others
$ set_priority 7 10
pid=7, new-priority=10
----following ps were run at time gap of 2 seconds.
$ ps
NAME PID STATE PRIORITY
init 1   SLEEPING 60
sh 2     SLEEPING 60
ps 10    RUNNING 60
myprog 4        RUNNABLE 60
myprog 7        RUNNING 10
$ ps
NAME PID STATE PRIORITY
init 1   SLEEPING 60
sh 2     SLEEPING 60
ps 11    RUNNING 60
myprog 4        RUNNABLE 60
myprog 7        RUNNING 10
$ ps
NAME PID STATE PRIORITY
init 1   SLEEPING 60
sh 2     SLEEPING 60
ps 12    RUNNING 60
myprog 4        RUNNABLE 60
myprog 7        RUNNING 10
$ ps
NAME PID STATE PRIORITY
init 1   SLEEPING 60
sh 2     SLEEPING 60
ps 13    RUNNING 60
myprog 4        RUNNABLE 60
myprog 7        RUNNING 10
```

--- so here we can clearly see once the process with highest priority comes, it’s executed first (state=RUNNING for pid=7) and then others are executed (see pid=4 after set priority).