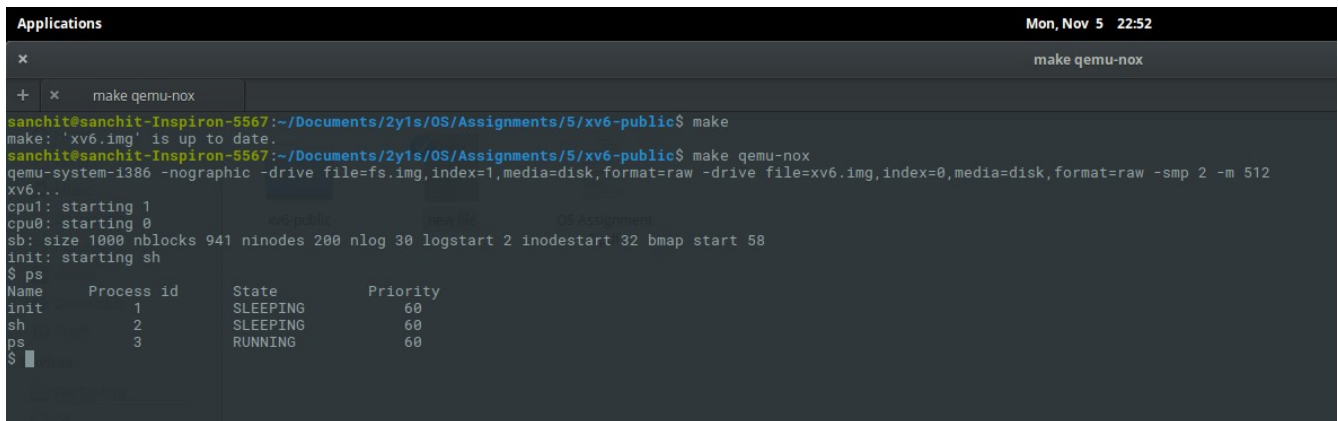


OS ASSIGNMENT 5 REPORT

- Implemented ps system call which displays list of processes with their names, id, status and priority
- All Processes are initially given 60 priority.
- Initially if we run ps command, output is-



The screenshot shows a terminal window titled "Applications" with a timestamp of "Mon, Nov 5 22:52". The terminal displays the following commands and output:

```
sanchit@sanchit-Inspiron-5567:~/Documents/2y1s/OS/Assignments/5/xv6-public$ make
make: 'xv6.img' is up to date.
sanchit@sanchit-Inspiron-5567:~/Documents/2y1s/OS/Assignments/5/xv6-public$ make qemu-nox
qemu-system-i386 -nographic -drive file=fs.img,index=1,media=disk,format=raw -drive file=xv6.img,index=0,media=disk,format=raw -smp 2 -m 512
xv6...
cpu1: starting 1
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap start 58
init: starting sh
$ ps
```

Name	Process id	State	Priority
init	1	SLEEPING	60
sh	2	SLEEPING	60
ps	3	RUNNING	60

Function for Testing – test

If we create 2 processes with same priority by test & command, then output is -

```
$ test & test &
$ ps
Name      Process id  State      Priority
init       1      SLEEPING    60
sh         2      SLEEPING    60
ps         8      RUNNING    60
test       7      RUNNABLE    60
test       6      RUNNING    60
$ ps
Name      Process id  State      Priority
init       1      SLEEPING    60
sh         2      SLEEPING    60
ps         9      RUNNING    60
test       7      RUNNING    60
test       6      RUNNABLE    60
$ ps
Name      Process id  State      Priority
init       1      SLEEPING    60
sh         2      SLEEPING    60
ps        10      RUNNING    60
test       7      RUNNABLE    60
test       6      RUNNING    60
$ ps
Name      Process id  State      Priority
init       1      SLEEPING    60
sh         2      SLEEPING    60
ps        11      RUNNING    60
test       7      RUNNING    60
test       6      RUNNABLE    60
$ ps
Name      Process id  State      Priority
init       1      SLEEPING    60
sh         2      SLEEPING    60
ps        12      RUNNING    60
test       7      RUNNABLE    60
test       6      RUNNING    60
$ ps
Name      Process id  State      Priority
init       1      SLEEPING    60
sh         2      SLEEPING    60
ps        13      RUNNING    60
test       7      RUNNABLE    60
test       6      RUNNING    60
$ ps
Name      Process id  State      Priority
init       1      SLEEPING    60
sh         2      SLEEPING    60
ps        14      RUNNING    60
test       7      RUNNING    60
test       6      RUNNABLE    60
$
```

A priority based scheduler selects the process with the highest priority. If two or more processes have same priority, the scheduler selects the process with the lowest priority of a process can be in the range of 0 to 100. priority. Set the default priority of a process to 0.

Implement the syscalls - **ps** and **set_priority** - where **ps** prints out the currently running processes and their priorities. **set_priority** is used to change the priority of a process.

int set_priority(int) - is the function declaration.

Submit a report with a small example where you compare the performance of the original round robin approach with the priority based approach.

Hint: Modify the proc structure for storing priority.

Write a sample benchmark program (main.c) which can be used to compare the performance of the original round robin approach with the priority based approach.

If we change the priority of the second process to 50 (i.e. higher priority), then the output is -

```
$ ps
Name      Process id  State      Priority
init      1             SLEEPING   60
sh        2             SLEEPING   60
ps        21            RUNNING    60
test      20            RUNNING    60
test      19            RUNNABLE   60
```

```
$ set_priority 19 50
pid = 19, priority = 50
```

```
$ ps
Name      Process id  State      Priority
init      1             SLEEPING   60
sh        2             SLEEPING   60
ps        23            RUNNING    60
test      20            RUNNABLE   60
test      19            RUNNING    50
```

```
$ ps
Name      Process id  State      Priority
init      1             SLEEPING   60
sh        2             SLEEPING   60
ps        24            RUNNING    60
test      20            RUNNABLE   60
test      19            RUNNING    50
```

```
$ ps
Name      Process id  State      Priority
init      1             SLEEPING   60
sh        2             SLEEPING   60
ps        25            RUNNING    60
test      20            RUNNABLE   60
test      19            RUNNING    50
```

```
$ ps
Name      Process id  State      Priority
init      1             SLEEPING   60
sh        2             SLEEPING   60
ps        26            RUNNING    60
test      20            RUNNABLE   60
test      19            RUNNING    50
```

```
$ ps
Name      Process id  State      Priority
init      1             SLEEPING   60
sh        2             SLEEPING   60
ps        27            RUNNING    60
test      20            RUNNABLE   60
test      19            RUNNING    50
```

```
$ ps
Name      Process id  State      Priority
init      1             SLEEPING   60
sh        2             SLEEPING   60
ps        28            RUNNING    60
test      20            RUNNABLE   60
test      19            RUNNING    50
```

```
$ ps
Name      Process id  State      Priority
init      1             SLEEPING   60
sh        2             SLEEPING   60
ps        29            RUNNING    60
test      20            RUNNABLE   60
test      19            RUNNING    50
```

Comparison of Priority Based Scheduling and Round-Robin Scheduling

- In round-robin all processes are scheduled in a circular manner.
- Whereas in priority scheduling a process is preempted if a higher process with higher priority comes and rescheduling is done.
- In priority scheduling starvation takes place I.e., lower priorities are not allocated CPU until all the processes with higher priority are added.