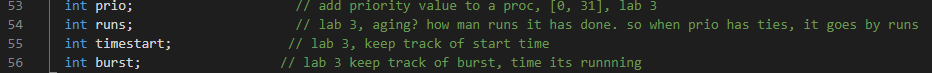
Mark Gameng

CS 450 – Duan Yue

## Lab 3 – Process Management

## Change Priority

**Proc.h**

**Proc.c**

A screenshot of a computer

Description automatically generated with medium confidence

Since were updating the priority of a process, we gotta lock. Also make sure that once it changes the priority value, it then transfers the control to the scheduler, sched(),. Immediately because the priority list has been updated. I also added a print statement, so its easier to see the process id and the priority value when I do the testing afterwards.

With this added system call, have to update other files

**User.h**



**Defs.h**



**Usys.s**



**Syscall.c**





**Syscall.h**



**Sysproc.c**

Text

Description automatically generated

**Initialization of priority and runs (aging) value:**

**Proc.c in allocproc()**

Graphical user interface, text, chat or text message

Description automatically generated

**In fork()**

Graphical user interface

Description automatically generated

These extra variables, runs, burst, timestart allow me to get the stats for each process so I can compare the priority and what these stats are at the end. Tracks the scheduling performance of each process as well as if my priority scheduling works or not.

**Update scheduler()**

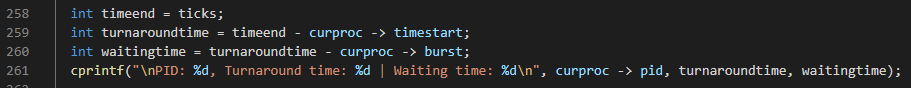
Text

Description automatically generated

Text

Description automatically generated

Basically, the scheduler is, it goes through the list of processes and finds and stores the one with the highest priority and least amount of runs. The least amount of runs is for when there are two processes with the same priority. It will then take the one that has ran less. Once it finds the process to run, it then decreases its own priority and increments the runs var while increasing the priority of the other processes by 1. This is the aging of priority. I also calculated the total burst time, by having the ticks before starting and ticks afterwards. This is for calculating the waiting time.

**In exit()**

In exit, I then calculate the turnaround times and waiting time as well as printing those with the PID. The PID is so I can cross reference which process had the original priority values and see if my priority scheduling works.

## Testing

For testing, I just used the code the TA showed in one of his slides.

Text

Description automatically generated

Though, I added a way for me to change the priority via the parameters when running in cmd. Also increased the limit to have a longer run time.

These were the results.

Text

Description automatically generated

Text

Description automatically generated

You can basically ignore the first two prints of stats, for example, PIDs 17, 19, 40, and 42. The pids with the priority is the one that matters. In both images, those with higher priority finished earlier, which is to be expected because all these processes have the same code/ same time to finish. For example, PID 39, with original priority of 1, had a turnaround time of 94 with waiting time 50, while PID 41 with originally priority of 10, had a turnaround time of 92 with a waiting time of 51. Even though, PID 39 started later than PID 41, since its priority is higher, then it ran first and ended first. The turnaround and waiting times tend to increase the lower priority it is, which is to be expected. Thus, my priority scheduling works.