SWE2007-42: Software Experiment2 (Fall 2018)

Programming Assignment #2

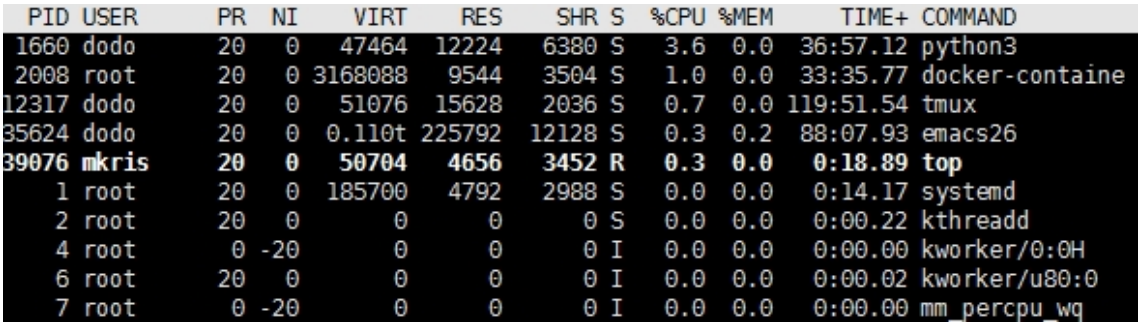
Due: Nov. 3rd, PM 23:59

1. Purpose

The purpose of the 2nd assignment is making a program such as UNIX top using "man" and "File I/O". But, the scope of your program is restricted compared with UNIX top, which is described as following "2. Scope".

2. Scope

Your program outputs the result as PID, USER, PR, NI, VIRT, RES, SHR, S, %CPU, %MEM, TIME+, COMMAND using only c programing language not the others (you could use any Header, **but should not use "exec" system call family - if use those, you get no point**). The processes is sorted in descending order by CPU Utilization (first) and in ascending order by PID (second), then top 10 of the processes prints out at terminal by the cycle of three seconds.



```
  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 1660 dodo      20   0   47464  12224   6380 S   3.6  0.0  36:57.12 python3
 2008 root      20   0 3168088   9544   3504 S   1.0  0.0  33:35.77 docker-containe
12317 dodo      20   0   51076  15628   2036 S   0.7  0.0 119:51.54 tmux
35624 dodo      20   0 0.110t 225792  12128 S   0.3  0.2  88:07.93 emacs26
39076 mkris     20   0   50704   4656   3452 R   0.3  0.0   0:18.89 top
    1 root      20   0  185700   4792   2988 S   0.0  0.0   0:14.17 systemd
    2 root      20   0       0      0      0 S   0.0  0.0   0:00.22 kthreadd
    4 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 kworker/0:0H
    6 root      20   0       0      0      0 I   0.0  0.0   0:00.02 kworker/u80:0
    7 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 mm_percpu_wq
```

[Hints]

- ☞ Caution: new processes could be generated or the old processes could be died in the progress of your top program
- ☞ USER name: use "getpwuid" system call
- ☞ man top, man proc(see section /proc/stat, /proc/meminfo, /proc/[PID]/stat, proc/[PID]/statm)
- ☞ Process CPU Utilization Calculation
  - CPU time since boot = user + nice + system + idle + iowait + irq + softirq + steal
  - Current Process CPU time (= cur) = utime + stime + cutime + cstime
  - Previous Process CPU time (= prev) = utime + stime + cutime + cstime
  - Period (for three senconds) = Current CPU time since boot - previous CPU time since boot
  - Process CPU percentage = (cur-prev) / (period / # of cpu cores) X 100 (cpu cores depends on hardware).
- ☞ standard-out refresh:

https://unix.stackexchange.com/questions/43075/how-to-change-the-contents-of-a-line-on-the-terminal-as-opposed-to-writing-a-new

3. Rating

- Total point is 100 points
  - ✓ The cycle of standard-out refresh at Terminal should be three seconds equal to that of UNIX top: **20 points**
  - ✓ Sorting the processes in descending order by CPU Utilization and in ascending order by PID (second), and top 10 of the processes should print out at Terminal (Only top 10 printed) : **20 points**
  - ✓ The results should be correct compared to the results of UNIX top: **60 points**
    - ☑ PID, USER: 5 points
    - ☑ PR, NI: 5 points
    - ☑ VIRT, RES: 5 points
    - ☑ SHR, S: 5 points

☑ %CPU: 15 points
☑ %MEM: 15 points
☑ TIME+: 5 points
☑ COMMAND: 5 points

4. Penalty

✓ you should handle "Error": there will occur error, because the contents in "proc" would be changed during the calculation. Otherwise, you will get –50 points.
✓ If using "exec" system call family, you will get 0 point.
✓ 15 points deducted per day. After 3 days, you will get 0 point.
✓ You should submit a source code file (**named "pa2.c"**) on i-campus. **Otherwise, you will get –30 points**
✓ The source code should be compiled successfully (compile option: "-Wall –W"). Otherwise, you will get 0 point.

※ In this class, there will be assignments the instead of tests (mid-term and final-term). Thereby, the assignments are very important and the question of program implementation issues would be not accepted.