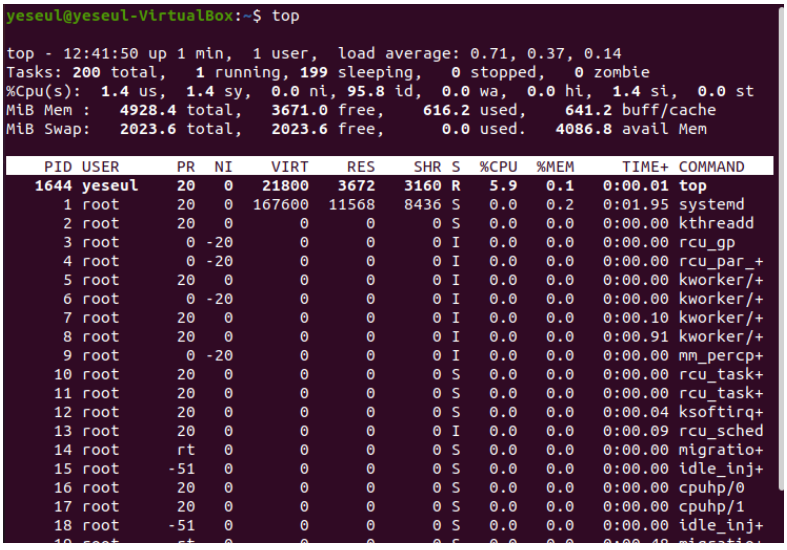
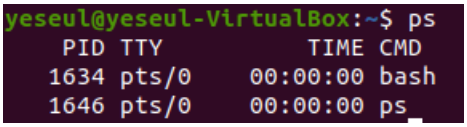
1. 과제 개요

1. mytop - 리눅스 파일 시스템 내에 있는 정보들을 수집하고 계산하여 리눅스의 top 명령어를 최대한 유사하게 구현. 헤더에 포함된 수행시간, load average, tasks, CPU사용률, MEM 사용률, SWAP MEM 정보와 프로세스에 대한 PID, USER, PR, NI, VIRT, RES, SHR, S, %CPU, %MEM, TIME+, COMMAND 등의 정보를 리눅스 파일시스템 내에서 찾아서 출력하도록 한다.



top명령어 실행 화면 (리눅스 명령어)

2. myps – 리눅스 파일 시스템 내에 있는 정보들을 수집하여 리눅스의 ps 명령어를 최대한 유사하게 구현한다. 각 프로세스의 PID, TTY, TIME(수행시간), CMD 등을 출력한다. ps 명령어에는 다양한 옵션이 있으며 그 중 a, u, x에 대한 옵션을 구현한다.



옵션이 없는 기본 ps 명령어 실행 화면 (리눅스 명령어)

3. mylscpu – 현재 사용중인 CPU정보를 수집하여 리눅스의 lscpu 명령어의 출력 결과와 최대한 유사한 결과를 출력할 수 있도록 구현한다. 이번 과제에서는 필수구현으론 CPU Vendor ID, CPU 모델명, CPU 속도, 캐쉬크기(L1i, L1d, L2) 이 있고 나머지 항목에 대한 구현은 option으로 한다.

**********(option 구현 항목)**********

Architecture, Address sizes, CPU(s), CPU family, model, Stepping, BogoMIPS, flags, Vulnerability관련 정보들

****

```
yeseul@yeseul-VirtualBox:~$ lscpu
Architecture:                    x86_64
CPU op-mode(s):                  32-bit, 64-bit
Byte Order:                      Little Endian
Address sizes:                   39 bits physical, 48 bits virtual
CPU(s):                          4
On-line CPU(s) list:             0-3
Thread(s) per core:              1
Core(s) per socket:              4
Socket(s):                       1
NUMA node(s):                    1
Vendor ID:                       GenuineIntel
CPU family:                      6
Model:                           142
Model name:                      Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz
Stepping:                        9
CPU MHz:                         2711.996
BogoMIPS:                        5423.99
Hypervisor vendor:               KVM
Virtualization type:             full
L1d cache:                       128 KiB
L1i cache:                       128 KiB
L2 cache:                        1 MiB
L3 cache:                        12 MiB
NUMA node0 CPU(s):               0-3
Vulnerability Itlb multihit:     KVM: Mitigation: VMX unsupported
Vulnerability L1tf:              Mitigation; PTE Inversion
Vulnerability Mds:               Mitigation; Clear CPU buffers; SMT Host state un
                                 known
Vulnerability Meltdown:          Mitigation; PTI
Vulnerability Spec store bypass: Vulnerable
Vulnerability Spectre v1:        Mitigation; usercopy/swapgs barriers and __user
                                 pointer sanitization
Vulnerability Spectre v2:        Mitigation; Full generic retpoline, STIBP disabl
                                 ed, RSB filling
Vulnerability Srbds:             Unknown: Dependent on hypervisor status
Vulnerability Tsx async abort:   Not affected
Flags:                           fpu vme de pse tsc msr pae mce cx8 apic sep mtrr
                                  pge mca cmov pat pse36 clflush mmx fxsr sse sse
                                 2 ht syscall nx rdtscp lm constant_tsc rep_good
                                 nopl xtopology nonstop_tsc cpuid tsc_known_freq
```

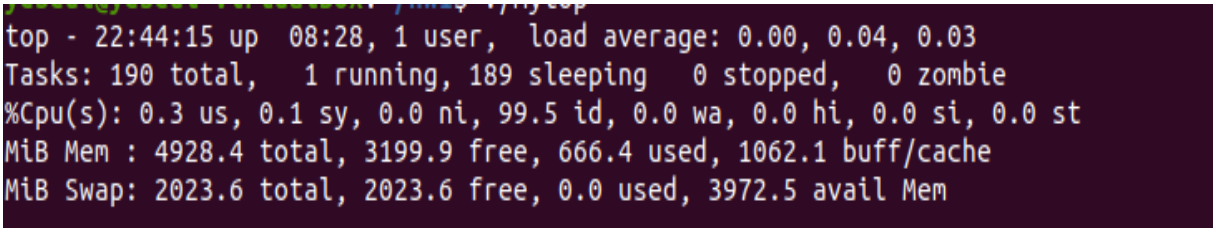
(↑ lscpu 명령어 실행 화면 (리눅스 명령어))

2. 소스코드 설명 및 결과

2-1. mytop.c 구현

2-1-(1). *헤더*

헤더에는 총 5행이 포함된다.



<1행>

- ***top*** : 시스템 현재 시간으로, localtime() 으로 받아온 time_t 구조체를 통해 정보를 얻어왔다.
- ***up*** : OS가 살아있는 시간, 즉 부팅한 시점으로부터의 시간으로 /proc/uptime 에서 얻어온 첫번째 토큰의 second로만 이루어진 정보를 hour와 minutes으로 계산하여 출력하였다.
- ***user*** : 현재 접속중인 유저 세션 수로 /var/run/utmp에 있는 utent를 특정 함수를 통해 읽어들이고, 그 로그인이 USER의 상태를 갖고 있다면 user의 수를 늘려가며 계산하였다.
- ***load average*** : 최근 1분/5분/15분 간 CPU Load의 이동평균을 말하는 것으로 /proc/loadavg 에서 1,2,3번째 정보를 파싱해와서 출력하였다.

<2행>

- ***Tasks*** : 현재 프로세스들의 상태를 나타내주는 영역이다. 전체 프로세스인 Total은 /proc에서 확인 가능한 모든 프로세스 디렉토리의 갯수를 구하여 출력하였고 나머지 running, sleeping, stopped, zombies는 각 프로세스의 /proc/[PID]/stat 파일에서 알 수 있었다. 해당 파일에는 많은 정보들이 있었으나 그 중 Status를 나타내는 정보만 파싱하여 갯수를 구하였다.

** 실행 결과 실제 top 명령어와 sleeping의 갯수가 다름을 인지하였고, 프로세스의 state가 D, I, S일 때 모두 sleeping상태로 계산한 결과 실제 top 명령어의 갯수와 같았습니다.

<3행>

%***Cpu***(s) : CPU가 어떻게 사용되고 있는지 그 사용률을 보여주는 영역이다. 각 요소는 아래와 같다.

- us : 프로세스의 유저 영역에서의 CPU 사용률 (/proc/stat 의 1 번째 token)
- sy : 프로세스의 커널 영역에서의 CPU 사용률 (/proc/stat 의 3 번째 token)
- ni : 프로세스의 우선순위 설정에 사용하는 CPU 사용률 (/proc/stat 의 2 번째 token)
- id : 사용하고 있지 않는 비율 (/proc/stat 의 4 번째 token)
- wa : IO 가 완료될때까지 기다리고 있는 CPU 비율 (/proc/stat 의 5 번째 token)
- hi : 하드웨어 인터럽트에 사용되는 CPU 사용률 (/proc/stat 의 6 번째 token)
- si : 소프트웨어 인터럽트에 사용되는 CPU 사용률 (/proc/stat 의 7 번째 token)
- st : CPU 를 VM 에서 사용하여 대기하는 CPU 비율 (/proc/stat 의 8 번째 token)

최근 refresh한 기록이 있다면 그 이후로부터의 CPU사용률(전 기록이 없다면 OS 시작 이후부터의 CPU 사용률)

<4~5행>

1. ***Mem total***: 물리적 메모리의 전체 크기

- /proc/meminfo 의 MemTotal

2. ***Mem free***: free 상태인 메모리의 크기

- /proc/meminfo 의 MemFree

3. ***Mem used***: 현재 사용 중인 메모리 크기

- /proc/meminfo 의 MemTotal - MemFree - Buffers - Cached – Sreclaimable
- Sreclaimable 은 커널에 의해 할당된 것 중 반환가능 한 크기

4. ***buff/cache***: buffer / Cache 메모리 크기

    - /proc/meminfo 의 Buffers + Cache + SReclaimable

5. ***Swap total***: 전체 Swap 크기

    - /proc/meminfo 의 SwapTotal

6. ***Swap free***: Swap 공간 중 사용가능한 공간 크기

    - /proc/meminfo 의 SwapFree

7. ***Swap used***: Swap 에서 사용중인 메모리 크기

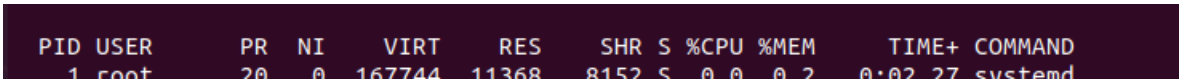
    - SwapTotal - SwapFree

8. ***avail Mem***: 사용 가능한 메모리 크기

    - /proc/meminfo 의 MemAvailable

2-1-(2). *프로세스*

프로세스에는 총 12 열이 있다.



1. ***PID*** : 각 프로세스에 해당하는 고유의 번호

   - /proc 내에 모든 pid 가 존재한다.

2. ***USER*** : 프로세스를 소유하고 있는 사용자의 이름

   - /proc/[pid]/stat 에서 알아낸 uid 로 getpwuid() 함수를 통해 user 명 알아냄

3. ***PR*** : 실행 우선 순위(priority)

   - /proc/[pid]/stat 에서 18 번째 token

4. ***NI*** : 실행 우선 순위 관련 NICE 값(일의 nice value 값. 마이너스를 가지는 NICE 값은 우선순위가 높다)

   - /proc/[pid]/stat 에서 19 번째 token

5. ***VIRT*** : 가상 메모리 사용량 (SWAP + RES)

   - /proc/[pid]/status 에서 VmSize 값

6. ***RES*** : 현재 페이지가 상주하고 있는 크기(물리 메모리 사용량)

   - /proc/[pid]/status 에서 VmRss 값

7. ***SHR*** : 분할된 프로세스에 의해 사용된 메모리를 나눈 메모리의 총합

   - /proc/[pid]/status 에서 RssFile 값

8. ***S*** : 프로세스의 상태 [S(sleeping), R(running), W(swapped out process), Z(zombies)]

- /proc/[pid]/stat 에서 3 번째 token

9. ***%CPU*** : 프로세스가 사용하는 CPU 의 사용률

- ((utime + stime) / hertz) / (uptime – (startTime/hertz)) * 100
- Sysconf(_SC_CLK_TCK)를 통하여 hertz 값을 얻었다.
- utime : user mode 에 scheduled 되는 process 의 시간의 양(/proc/[pid]/stat 의 14 번째 token)
- stime : kernel mode 에 scheduled 되는 process 의 시간의 양(/proc/[pid]/stat 의 15 번째 token)
- startTime : OS 부팅 후 프로세스 시작까지의 시간의 양(/proc/[pid]/stat 의 22 번째 token)
- uptime : OS 부팅 후 지난 시간(초 단위), /proc/uptime 의 첫번째 token

10. ***%MEM*** : 프로세스가 사용하는 메모리의 사용률

- RES/memTotal 값으로 계산한다.

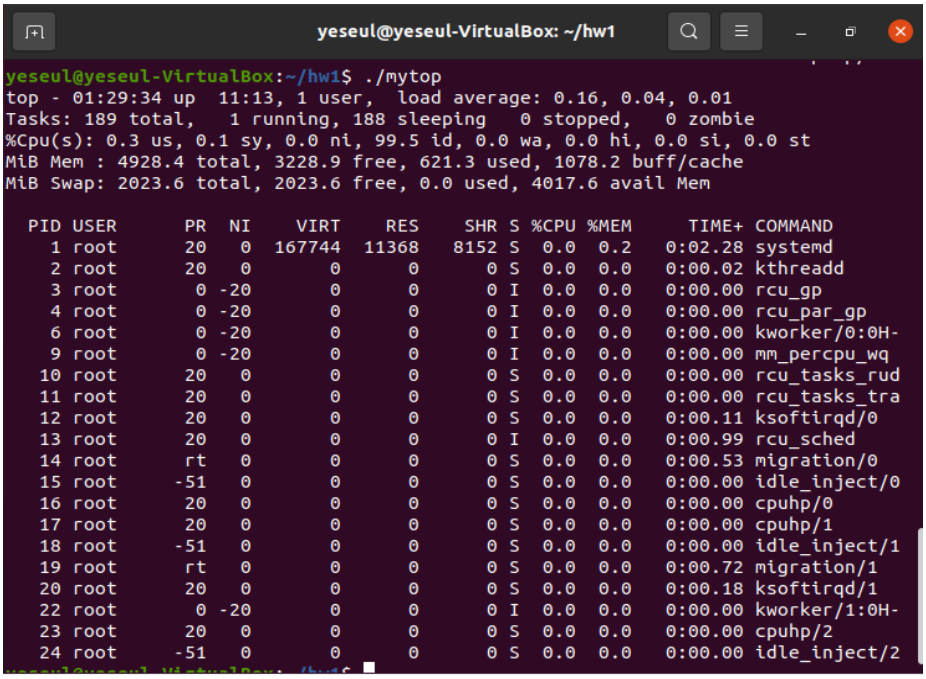
11. ***TIME+*** : CPU 사용시간 (0.01 초 단위)

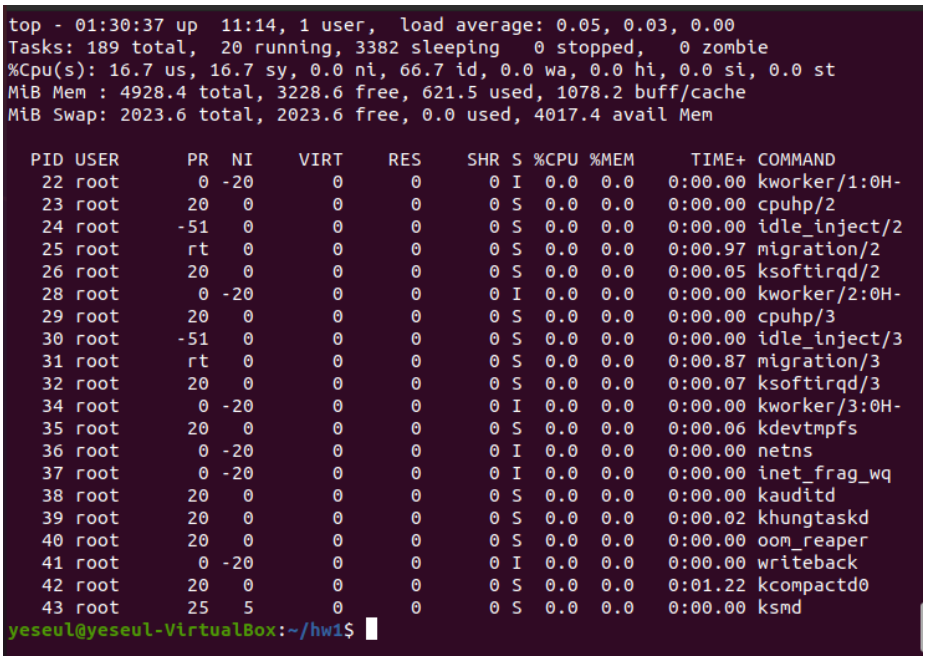
- Utime 과 stime 을 이용하여 계산한다.

12. ***COMMAND*** : 프로세스 실행 시 입력 된 명령어

- /proc/[pid]/stat 의 두번째 token 에서 얻을 수 있다.

2-1-(3). *결과화면*





아래/위 방향키를 통해 계속 이어서 프로세스의 정보를 볼 수 있게 하였습니다.

2-2. myps.c 구현

2-2-(1). *aux 옵션*

```
yeseul@yeseul-VirtualBox:~$ ps aux
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
```

최종적으로 옵션 a, u, x를 구현하기 위해서 필요한 정보들은 위의 사진과 같다.

- a 옵션 : 터미널과 연관된 프로세스를 출력하는 옵션. (PID, TTY, STAT, TIME, COMMAND)
- u 옵션 : 프로세스의 소유자를 기준으로 출력하는 옵션. (USER, PID, %CPU, %MEM, VSZ, RSS, TTY, STAT, START, TIME, COMMAND)
- x 옵션 : 데몬 프로세스처럼 터미널에 종속되지 않는 프로세스까지 출력한다. (PID, TTY, STAT, TIME, COMMAND)

2-2-(2). *구현에 필요한 정보*

1. ***USER*** : 프로세스를 소유하고 있는 사용자의 이름

- /proc/[pid]/stat 에서 알아낸 uid 로 getpwuid() 함수를 통해 user 명 알아냄

2. ***PID*** : 각 프로세스에 해당하는 고유의 번호

- /proc 내에 존재하는 숫자로만 이루어진 디렉토리 파일 이름을 얻어옴.

3. ***%CPU*** : 프로세스가 사용하는 CPU 의 사용률

- ((utime + stime) / hertz) / (uptime – (startTime/hertz)) * 100
- Sysconf(_SC_CLK_TCK)를 통하여 hertz 값을 얻었다.
- utime : user mode 에 scheduled 되는 process 의 시간의 양(/proc/[pid]/stat 의 14 번째 token)
- stime : kernel mode 에 scheduled 되는 process 의 시간의 양(/proc/[pid]/stat 의 15 번째 token)
- startTime : OS 부팅 후 프로세스 시작까지의 시간의 양(/proc/[pid]/stat 의 22 번째 token)
- uptime : OS 부팅 후 지난 시간(초 단위), /proc/uptime 의 첫번째 token

4. ***%MEM*** : 프로세스가 사용하는 메모리의 사용률

- RES/memTotal 값으로 계산한다.

5. ***VSZ*** : 가상 메모리 사용량 (SWAP + RES)

- /proc/[pid]/status 에서 VmSize 값

6. ***RSS*** : 현재 페이지가 상주하고 있는 크기(물리 메모리 사용량)

- /proc/[pid]/status 에서 VmRss 값

7. ***TTY*** : 프로세스와 연결된 터미널

- /proc/[pid]/fd/0 이 symbolic link 로 가리키는 파일명을 확인한다.
- 일반 사용자 권한으로 해당 파일에 접근 불가능한 문제 발생
  - ✓ /dev 내 문자 디바이스 파일 중 statbuf.st_rdev 와 /proc/[pid]/stat 의 ttyNr 값이 같은 것을 찾아 해당 파일 명을 획득한다.
- 위 두 방법으로도 찾지 못한 TTY 는 "?"로 설정하였다.

8. ***STAT*** : 프로세스의 상태 코드

- s : sid(/proc/[pid]/stat 의 6 번째 token)과 pid 가 같은 경우
- N : nice 값이 양수인 경우
- < : nice 값이 음수인 경우
- L : vmLck(/proc/[pid]/status 의 19 번째 행)이 0 이 아닌 경우(가상메모리가 Lock 인 경우)
- l : 쓰레드의 갯수(/proc/[pid]/stat 의 20 번째 token)가 1 개 초과인 경우(멀티 쓰레드)
- + : 해당프로세스의 pgid 와 tgpid(/proc/[pid]/stat 의 8 번째 token)가 같은 경우

9. ***START*** : 프로세스가 시작된 시간

- time(NULL) – (uptime-(startTime/hertz))의 값을 이용하여 localtime() 함수를 실행시켜서 시/분값을 얻음

10. ***TIME***: 총 CPU 사용 시간

- (utime + stime)/hertz 를 이용하여 분/초로 값을 나타냄

11. ***COMMAND*** : 프로세스의 실행 명령행

- /proc/[pid]/cmdline 에서 값을 가져옴
- 위의 경로에서 값을 얻지 못했을 경우 /proc/[pid]/comm 파일에서 값을 가져옴

2-2-(3). *myps 결과 화면*

```
yeseul@yeseul-VirtualBox:~/hw1$ ./myps
  PID TTY          TIME COMMAND
 1634 pts/0    00:00:00 bash
 2910 pts/0    00:00:00 ./myps
yeseul@yeseul-VirtualBox:~/hw1$ ./myps a
  PID TTY      STAT   TIME COMMAND
  736 tty2     Ssl+   0:00 /usr/lib/gdm3/gdm-x-session --run-script env GNOME_S
  739 tty2     Sl+    0:08 /usr/lib/xorg/Xorg vt2 -displayfd 3 -auth /run/user/
 1634 pts/0    Ss     0:00 bash
 2911 pts/0    R+     0:00 ./myps a
yeseul@yeseul-VirtualBox:~/hw1$ ./myps u
USER         PID %CPU %MEM     VSZ    RSS TTY      STAT  START   TIME COMMAND
yeseul       736  0.0  0.1  174124   6652 tty2     Ssl+   12:40   0:00 /usr/lib/g
yeseul       739  0.1  1.3 1120748  65328 tty2     Sl+    12:40   0:08 /usr/lib/x
yeseul      1634  0.0  0.1   20812   4972 pts/0    Ss     12:41   0:00 bash
yeseul      2912  0.0  0.1    4784   4104 pts/0    R+     15:41   0:00 ./myps u
yeseul@yeseul-VirtualBox:~/hw1$ ./myps au
USER         PID %CPU %MEM     VSZ    RSS TTY      STAT  START   TIME COMMAND
yeseul       736  0.0  0.1  174124   6652 tty2     Ssl+   12:40   0:00 /usr/lib/g
yeseul       739  0.1  1.3 1120748  65328 tty2     Sl+    12:40   0:08 /usr/lib/x
yeseul      1634  0.0  0.1   20812   4972 pts/0    Ss     12:41   0:00 bash
yeseul      2913  0.0  0.1    4784   3996 pts/0    R+     15:41   0:00 ./myps au
yeseul@yeseul-VirtualBox:~/hw1$
```

(↑ NoOption, Option a, Option u, Option au 구현 결과)

```
yeseul@yeseul-VirtualBox:~/hw1$ ./myps x
  PID TTY      STAT   TIME COMMAND
  713 ?        Ss     0:00 /lib/systemd/systemd --user
  714 ?        S      0:00 (sd-pam)
  724 ?        Ss<l   0:01 /usr/bin/pulseaudio --daemonize=no --log-target=jour
  726 ?        SsNl   0:00 /usr/libexec/tracker-miner-fs
  729 ?        Sl     0:00 /usr/bin/gnome-keyring-daemon --daemonize --login
  735 ?        Ss     0:00 /usr/bin/dbus-daemon --session --address=systemd: --
  736 tty2     Ssl+   0:00 /usr/lib/gdm3/gdm-x-session --run-script env GNOME_S
  739 tty2     Sl+    0:08 /usr/lib/xorg/Xorg vt2 -displayfd 3 -auth /run/user/
  746 ?        Ssl    0:00 /usr/libexec/gvfsd
  767 ?        Sl     0:00 /usr/libexec/gvfsd-fuse /run/user/1000/gvfs -f -o bi
  774 ?        Ssl    0:00 /usr/libexec/gvfs-udisks2-volume-monitor
  785 ?        Ssl    0:00 /usr/libexec/gvfs-goa-volume-monitor
  789 ?        Sl     0:00 /usr/libexec/goa-daemon
  850 ?        Sl     0:00 /usr/libexec/goa-identity-service
  860 ?        Ssl    0:00 /usr/libexec/gvfs-mtp-volume-monitor
  865 ?        Ssl    0:00 /usr/libexec/gvfs-afc-volume-monitor
  871 ?        Ssl    0:00 /usr/libexec/gvfs-gphoto2-volume-monitor
  916 ?        Sl+    0:00 /usr/libexec/gnome-session-binary --systemd --system
  983 ?        Ss     0:00 /usr/bin/ssh-agent /usr/bin/im-launch env GNOME_SHEL
 1014 ?        Ssl    0:00 /usr/libexec/at-spi-bus-launcher
 1020 ?        S      0:00 /usr/bin/dbus-daemon --config-file=/usr/share/defaul
 1051 ?        Ssl    0:00 /usr/libexec/gnome-session-ctl --monitor
 1058 ?        Ssl    0:00 /usr/libexec/gnome-session-binary --systemd-service
 1072 ?        Ssl    0:27 /usr/bin/gnome-shell
 1108 ?        Sl     0:00 ibus-daemon --panel disable --xim
 1112 ?        Sl     0:00 /usr/libexec/ibus-dconf
 1113 ?        Sl     0:04 /usr/libexec/ibus-extension-gtk3
 1115 ?        Sl     0:00 /usr/libexec/ibus-x11 --kill-daemon
 1119 ?        Sl     0:00 /usr/libexec/ibus-portal
 1130 ?        Sl     0:00 /usr/libexec/at-spi2-registryd --use-gnome-session
 1135 ?        Ssl    0:00 /usr/libexec/xdg-permission-store
 1140 ?        Sl     0:00 /usr/libexec/gnome-shell-calendar-server
 1146 ?        Ssl    0:00 /usr/libexec/evolution-source-registry
 1155 ?        Ssl    0:00 /usr/libexec/evolution-calendar-factory
 1169 ?        Sl     0:00 /usr/libexec/dconf-service
 1176 ?        Ssl    0:00 /usr/libexec/evolution-addressbook-factory
 1194 ?        Sl     0:00 /usr/bin/gjs /usr/share/gnome-shell/org.gnome.Shell.
 1202 ?        Sl     0:00 /usr/libexec/gvfsd-trash --spawner :1.4 /org/gtk/gvf
 1212 ?        Ssl    0:00 /usr/libexec/gsd-a11y-settings
```

(Option x 구현 결과)

```
yeseul@yeseul-VirtualBox:~/hw1$ ./myps ax
  PID TTY      STAT   TIME COMMAND
    1 ?        Ss     0:02 /sbin/init splash
    2 ?        S      0:00 [kthreadd]
    3 ?        I<     0:00 [rcu_gp]
    4 ?        I<     0:00 [rcu_par_gp]
    6 ?        I<     0:00 [kworker/0:0H-events_highpri]
    7 ?        I      0:00 [kworker/0:1-mm_percpu_wq]
    9 ?        I<     0:00 [mm_percpu_wq]
   10 ?        S      0:00 [rcu_tasks_rude_]
   11 ?        S      0:00 [rcu_tasks_trace]
   12 ?        S      0:00 [ksoftirqd/0]
   13 ?        I      0:00 [rcu_sched]
   14 ?        S      0:00 [migration/0]
   15 ?        S      0:00 [idle_inject/0]
   16 ?        S      0:00 [cpuhp/0]
   17 ?        S      0:00 [cpuhp/1]
   18 ?        S      0:00 [idle_inject/1]
   19 ?        S      0:00 [migration/1]
   20 ?        S      0:00 [ksoftirqd/1]
   22 ?        I<     0:00 [kworker/1:0H-events_highpri]
   23 ?        S      0:00 [cpuhp/2]
   24 ?        S      0:00 [idle_inject/2]
```

(Option ax 구현 결과 중간 생략)

```
yeseul@yeseul-VirtualBox:~/hw1$ ./myps ux
USER         PID %CPU %MEM     VSZ    RSS TTY      STAT  START   TIME COMMAND
yeseul       713  0.0  0.2   19280  10372 ?        Ss    12:40   0:00 /lib/syste
yeseul       714  0.0  0.1  103348   3356 ?        S     12:40   0:00 (sd-pam)
yeseul       724  0.0  0.4 1418428  18692 ?        Ss<l  12:40   0:01 /usr/bin/p
yeseul       726  0.0  0.5  521608  24528 ?        SsNl  12:40   0:00 /usr/libex
yeseul       729  0.0  0.1  250408   7412 ?        Sl    12:40   0:00 /usr/bin/g
yeseul       735  0.0  0.1    8108   5232 ?        Ss    12:40   0:00 /usr/bin/d
yeseul       736  0.0  0.1  174124   6652 tty2     Ssl+  12:40   0:00 /usr/lib/g
yeseul       739  0.1  1.3 1120748  65328 tty2     Sl+   12:40   0:09 /usr/lib/x
yeseul       746  0.0  0.2  249844   7872 ?        Ssl   12:40   0:00 /usr/libex
yeseul       767  0.0  0.1  378344   6396 ?        Sl    12:40   0:00 /usr/libex
yeseul       774  0.0  0.2  324180   9140 ?        Ssl   12:40   0:00 /usr/libex
yeseul       785  0.0  0.1  246008   6124 ?        Ssl   12:40   0:00 /usr/libex
yeseul       789  0.0  0.7  555492  36364 ?        Sl    12:40   0:00 /usr/libex
yeseul       850  0.0  0.2  325020   9120 ?        Sl    12:40   0:00 /usr/libex
yeseul       860  0.0  0.1  245804   6256 ?        Ssl   12:40   0:00 /usr/libex
yeseul       865  0.0  0.2  326856   8812 ?        Ssl   12:40   0:00 /usr/libex
yeseul       871  0.0  0.1  248080   6820 ?        Ssl   12:40   0:00 /usr/libex
yeseul       916  0.0  0.3  198524  13776 ?        Sl+   12:40   0:00 /usr/libex
yeseul       983  0.0  0.0    6040    456 ?        Ss    12:40   0:00 /usr/bin/s
yeseul      1014  0.0  0.1  305368   6848 ?        Ssl   12:40   0:00 /usr/libex
yeseul      1020  0.0  0.1    7248   4284 ?        S     12:40   0:00 /usr/bin/d
yeseul      1051  0.0  0.1  100180   4280 ?        Ssl   12:40   0:00 /usr/libex
yeseul      1058  0.0  0.3  634344  16500 ?        Ssl   12:40   0:00 /usr/libex
yeseul      1072  0.2  6.6 5004596 335228 ?        Ssl   12:40   0:27 /usr/bin/g
yeseul      1108  0.0  0.2  321172   8428 ?        Sl    12:40   0:00 ibus-daemo
```

(Option ux 구현 결과 중간 생략)

```
root         360  0.0  0.0       0      0 ?        I<    12:40   0:00 [cryptd]
systemd+     508  0.0  0.3   24100  13360 ?        Ss    12:40   0:00 /lib/systemd/
systemd+     509  0.0  0.1   90456   6216 ?        Ssl   12:40   0:00 /lib/systemd/
root         542  0.0  0.2  248500   7652 ?        Ssl   12:40   0:00 /usr/lib/acco
root         543  0.0  0.0    2548    784 ?        Ss    12:40   0:00 /usr/sbin/acp
avahi        546  0.0  0.1    8532   3408 ?        Ss    12:40   0:00 avahi-daemon:
root         547  0.0  0.1   19352   2908 ?        Ss    12:40   0:00 /usr/sbin/cro
root         548  0.0  0.2   37988   8392 ?        Ss    12:40   0:00 /usr/sbin/cup
message+     549  0.0  0.1    8848   5884 ?        Ss    12:40   0:01 /usr/bin/dbus
root         552  0.0  0.4  494108  19860 ?        Ssl   12:40   0:01 /usr/sbin/Net
root         563  0.0  0.1   81836   3596 ?        Ssl   12:40   0:00 /usr/sbin/irq
root         565  0.0  0.4   49264  20048 ?        Ss    12:40   0:00 /usr/bin/pyth
root         567  0.0  0.2  251416  10792 ?        Ssl   12:40   0:00 /usr/lib/poli
syslog       569  0.0  0.1  224356   4756 ?        Ssl   12:40   0:00 /usr/sbin/rsy
root         571  0.0  0.6 1146920  28240 ?        Ssl   12:40   0:03 /usr/lib/snap
root         572  0.0  0.1  245668   6200 ?        Ssl   12:40   0:00 /usr/libexec/
root         574  0.0  0.2   16908   8324 ?        Ss    12:40   0:00 /lib/systemd/
root         579  0.0  0.2  392824  11960 ?        Ssl   12:40   0:00 /usr/lib/udis
root         584  0.0  0.1   13688   4784 ?        Ss    12:40   0:00 /sbin/wpa_sup
avahi        590  0.0  0.0    8352    324 ?        S     12:40   0:00 avahi-daemon:
root         603  0.0  0.0       0      0 ?        I     12:40   0:03 [kworker/3:3-
root         635  0.0  0.2  178400  12528 ?        Ssl   12:40   0:00 /usr/sbin/cup
root         638  0.0  0.4  128052  22620 ?        Ssl   12:40   0:00 /usr/bin/pyth
root         647  0.0  0.2  313760  10596 ?        Ssl   12:40   0:00 /usr/sbin/Mod
```

(Option aux 구현결과 앞뒤 생략)

2-3. mylscpu.c 구현

2-3-(1). *필수 구현 요소*

1. ***Vendor_ID*** : 제조사 ID

- /proc/cpuinfo 내의 vendor_id 정보 파싱

2. ***Model_name*** : CPU 모델명

- /proc/cpuinfo 내의 model name 정보 파싱

3. ***CPU_MHz***: CPU 속도

- /proc/cpuinfo 내의 cpu MHz 정보 파싱

4. ***Cache_size(L1d, L1i, L2)*** : cache 크기

- /sys/devices/system/cpu/cpu#/cache/index#/size 경로를 통해 정보를 가져옴.
- cpu 의 개수가 여러개라면 하나의 파일에서만 가져오면 그만큼 오차가 있기 때문에 cpuinfo 에서 가져온 cpu core 의 개수를 통해 cpu#의 캐쉬사이즈를 가져올 수 있도록 함.
- Index0 -> L1d cache // index1 -> L1i cache // index2 -> L2 cache

2-3-(2). *optional 구현 요소*

1. ***Architecture***

- /lib 아래의 모든 파일 이름을 읽고 "linux-gnu" 문자열이 포함된 파일명을 가져온 후 앞의 architecture 부분만 파싱해서 얻어옴.

2. ***address_sizes***

- /proc/cpuinfo 내의 address_sizes 정보 파싱

3. ***CPUs***

- /proc/cpuinfo 내의 cpu cores 정보 파싱

4. ***CPU_family***

- /proc/cpuinfo 내의 cpu family 정보 파싱

5. ***model***

- /proc/cpuinfo 내의 model 정보 파싱

6. ***stepping***

- /proc/cpuinfo 내의 Stepping 정보 파싱

7.***bogoMIPS***

- /proc/cpuinfo 내의 bogomips 정보 파싱

8. ***Vulnerability 관련 정보 (각 경로를 통해 정보 가져옴)***

- Vulnerabilities Itlb multihit : "/sys/devices/system/cpu/vulnerabilities/itlb_multihit"
- Vulnerabilities L1tf : "/sys/devices/system/cpu/vulnerabilities/l1tf"
- Vulnerabilities Mds : "/sys/devices/system/cpu/vulnerabilities/mds"
- Vulnerabilities Meltdown : "/sys/devices/system/cpu/vulnerabilities/meltdown"
- Vulnerabilities Spec store bypass : "/sys/devices/system/cpu/vulnerabilities/spec_store_bypass"
- Vulnerabilities Spectre v1 : "/sys/devices/system/cpu/vulnerabilities/spectre_v1"
- Vulnerabilities Spectre v2 : "/sys/devices/system/cpu/vulnerabilities/spectre_v2"
- Vulnerabilities Srbds : "/sys/devices/system/cpu/vulnerabilities/srbds"
- Vulnerabilities Tsx async abort : "/sys/devices/system/cpu/vulnerabilities/tsx_async_abort"

9. ***flags***

- /proc/cpuinfo 내의 flags 정보 파싱

2-3-(3). *mylscpu 결과 화면*

```
yeseul@yeseul-VirtualBox:~/hw1$ ./mylscpu
Architecture:                    x86_64
Address sizes:                   39 bits physical, 48 bits virtual
CPU(s):                          4
Vendor ID:                       GenuineIntel
CPU family:                      6
Model:                           142
Model Name:                      Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz
Stepping:                        9
CPU MHz:                         2711.996
BogoMIPS:                        5423.99
L1d cache:                       128kiB
L1i cache:                       128KiB
L2 cache:                        1MiB
Vulnerability Itlb multihit:     KVM: Mitigation: VMX unsupported
Vulnerability L1tf:              Mitigation: PTE Inversion
Vulnerability Mds:               Mitigation: Clear CPU buffers; SMT Host state unknown
Vulnerability Meltdown:          Mitigation: PTI
Vulnerability Spec store bypass: Vulnerable
Vulnerability Spectre v1:        Mitigation: usercopy/swapgs barriers and __user pointer sanitization
Vulnerability Spectre v2:        Mitigation: Full generic retpoline, STIBP: disabled, RSB filling
Vulnerability Srbds:             Unknown: Dependent on hypervisor status
Vulnerability Tsx async abort:   Not affected
flags:                           fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflu
sh mmx fxsr sse sse2 ht syscall nx rdtscp lm constant_tsc rep_good nopl xtopology nonstop_tsc cpuid tsc_known_
freq pni pclmulqdq ssse3 cx16 pcid sse4_1 sse4_2 x2apic movbe popcnt aes xsave avx rdrand hypervisor lahf_lm a
bm 3dnowprefetch invpcid_single pti fsgsbase avx2 invpcid rdseed clflushopt md_clear flush_l1d
yeseul@yeseul-VirtualBox:~/hw1$
```

3. 과제를 하면서 느낀 점

그냥 명령어를 출력만 했다면 뭐가 뭔지 몰랐을 정보들이 직접 찾아가며 알아냈더니 어떤 정보가 뭘 나타내고 이는 어떤 값들로 이루어졌는지 잘 알 수 있었다.

무엇보다 프로세스의 상태를 볼 수 있는 눈을 키웠다고 생각한다. 윈도우에만 익숙해져있어서 리눅스의 프로세스 상태나 출력값에 대해 의문을 가진점이 많았는데 좀 더 각 값들에 대한 이해도가 높아지면서 나중에 리눅스를 사용중에 프로세스에 문제가 생기더라도 프로세스의 상태를 확인하고 적절히 조치할 수 있을거라고 생각된다.

과제를 하면서 아쉬운 점으로는, mytop 명령어에서 window size에 따라 command쪽에 변화를 주고자 하였으나, window size를 구하는 데에만 성공하고 그를 활용하는 방법을 찾지 못해 구현하지 못하였다. 원래의 top 명령어처럼 오른쪽/왼쪽 방향키를 사용할 수 없다는 점이 아쉽다. 또한 같은 문제로 mylscpu의 출력에서도 flags 값 처럼 긴 문자열이 나올 때 window size에 맞춰 칸을 맞추고 싶었으나 구현하지 못하여 깔끔한 화면이 출력되지 못한 점이 아쉽다.

저번 리눅스 시스템 프로그래밍 과목을 들으며 배우기만 했던 함수들을 직접 더 많이 써보고 더 찾아볼 수 있는 기회가 생겨 좋았다. 비록 모든 코드를 처음부터 작성하진 못하고 참고한 코드들이 많지만 그래도 그 코드들을 통해 더 많은 함수와 더 많은 리눅스의 기능을 알 수 있었다.