

# homework #7

인공지능학부 211852 조나현

# homework #7

1.

```
nahyune@nahyune-virtual-machine:~$ ps -ef | grep init
root      1      0  5 14:29 ?        00:00:11 /sbin/init auto noprompt splash
nahyune   2079   1490  2 14:31 ?        00:00:03 /usr/libexec/gnome-initial-setup --existing-user
nahyune   2411   2264  0 14:33 pts/0    00:00:00 grep --color=auto init
```

- grep :

'grep'는 유닉스 기반 운영 체제에서 지정된 패턴 또는 정규 표현식을 하나 이상의 파일 또는 다른 명령어의 출력 내에서 검색하는 명령 줄 유틸리티 프로그램이다.

- pid of init program: 1

# homework #7

2.

```
Tasks: 328 total, 3 running, 325 sleeping, 0 stopped, 0 zombie
%Cpu(s): 27.8 us, 15.6 sy, 0.0 ni, 54.4 id, 1.1 wa, 0.0 hi, 1.1 si, 0.0 st
MiB Mem : 3888.8 total, 1038.2 free, 1823.1 used, 1027.4 buff/cache
MiB Swap: 2140.0 total, 2140.0 free, 0.0 used, 1794.8 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
2841	root	20	0	102912	85360	2332	R	48.5	2.1	0:01.47	apparno+
2843	root	20	0	54480	36768	2208	R	26.7	0.9	0:00.81	apparno+
2581	root	20	0	949832	37508	19328	S	16.2	0.9	0:03.22	snappd
824	root	20	0	466984	14092	11024	S	3.3	0.4	0:00.94	udisksd
446	root	20	0	26824	7208	4748	S	1.7	0.2	0:01.65	systemd+
768	message+	20	0	11104	6720	4048	S	1.7	0.2	0:02.13	dbus-da+
866	root	20	0	316944	12120	10272	S	1.3	0.3	0:00.37	ModemMa+
1691	nahyune	20	0	4048772	252376	122832	S	1.3	6.3	0:15.44	gnome-s+
2723	root	20	0	26824	6756	4216	S	1.3	0.2	0:00.25	systemd+
2739	root	20	0	26824	5756	3216	S	1.3	0.1	0:00.29	systemd+
1	root	20	0	167988	13364	8332	S	1.0	0.3	0:12.61	systemd
1630	nahyune	20	0	399748	10452	8556	S	1.0	0.3	0:00.79	gvfs-ud+
2721	root	20	0	26824	5812	3272	S	1.0	0.1	0:00.15	systemd+
2724	root	20	0	26824	5660	3180	S	1.0	0.1	0:00.12	systemd+
2727	root	20	0	26824	5176	2636	S	1.0	0.1	0:00.24	systemd+
2731	root	20	0	26824	6180	3640	S	1.0	0.2	0:00.25	systemd+
2732	root	20	0	26824	5888	3404	S	1.0	0.1	0:00.22	systemd+

- space : 화면을 갱신하고 최신 프로세스 정보를 표시
- M: 메모리 사용량을 기본으로 프로세스를 정렬하기
- P : CPU 사용량을 기반으로 프로세스를 정렬하기
- q : top 프로그램 종료하기

# homework #7

3.

```
nahyune@nahyune-virtual-machine:~$ ps u
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
nahyune    1551  0.0  0.1 172348 6352 tty2      Ss+  14:30   0:00 /usr/libexec/
nahyune    1560  0.0  0.3 232992 15764 tty2      Sl+  14:30   0:00 /usr/libexec/
nahyune    2264  0.0  0.1 21108 5524 pts/0     Ss   14:31   0:00 bash
nahyune    3313  0.0  0.0 22636 3808 pts/0     R+   14:41   0:00 ps u
nahyune@nahyune-virtual-machine:~$ bash
nahyune@nahyune-virtual-machine:~$ bash
nahyune@nahyune-virtual-machine:~$ bash
nahyune@nahyune-virtual-machine:~$ ps l
F  UID      PID    PPID  PRI   NI     VSZ   RSS WCHAN    STAT TTY          TIME COMMAND
4  1000    1551    1481   20    0 172348 6352 do_pol  Ss+  tty2         0:00 /usr/li
0  1000    1560    1551   20    0 232992 15764 do_pol  Sl+  tty2         0:00 /usr/li
0  1000    2264    2246   20    0 21108 5524 do_wai  Ss   pts/0        0:00 bash
0  1000    3315    2264   20    0 21112 5528 do_wai  S    pts/0        0:00 bash
0  1000    3322    3315   20    0 21112 5528 do_wai  S    pts/0        0:00 bash
0  1000    3328    3322   20    0 21112 5572 do_wai  S    pts/0        0:00 bash
0  1000    3334    3328   20    0 22636 3696 -       R+   pts/0        0:00 ps l
nahyune@nahyune-virtual-machine:~$ ps f
    PID TTY          STAT TIME COMMAND
    2264 pts/0      Ss    0:00 bash
    3315 pts/0      S      0:00 \_ bash
    3322 pts/0      S      0:00 \_ bash
    3328 pts/0      S      0:00 \_ bash
    3335 pts/0      R+     0:00 \_ ps f
    1551 tty2      Ss+    0:00 /usr/libexec/gdm-wayland-session env GNOME_SHELL_SE
    1560 tty2      Sl+    0:00 \_ /usr/libexec/gnome-session-binary --session=ubu
nahyune@nahyune-virtual-machine:~$ exit
exit
nahyune@nahyune-virtual-machine:~$ exit
exit
```

- parent process of the process “ps f”:  
init 프로세스

# homework #7

4.

```
nahyune@nahyune-virtual-machine:~$ vi a
[1]+  Stopped                  vi a
nahyune@nahyune-virtual-machine:~$ jobs
[1]+  Stopped                  vi a
nahyune@nahyune-virtual-machine:~$ sleep 10000 &
[2] 3340
nahyune@nahyune-virtual-machine:~$ jobs
[1]+  Stopped                  vi a
[2]-  Running                  sleep 10000 &
nahyune@nahyune-virtual-machine:~$ vi b
[3]+  Stopped                  vi b
nahyune@nahyune-virtual-machine:~$ jobs
[1]-  Stopped                  vi a
[2]  Running                  sleep 10000 &
[3]+  Stopped                  vi b
nahyune@nahyune-virtual-machine:~$ vi c
[4]+  Stopped                  vi c
nahyune@nahyune-virtual-machine:~$ jobs
[1]  Stopped                  vi a
[2]  Running                  sleep 10000 &
[3]-  Stopped                  vi b
[4]+  Stopped                  vi c
nahyune@nahyune-virtual-machine:~$ fg
vi c
nahyune@nahyune-virtual-machine:~$ jobs
[1]-  Stopped                  vi a
[2]  Running                  sleep 10000 &
[3]+  Stopped                  vi b
```

- + : 가장 최근에 foreground에서 활동한 job
- - : 이전 foreground job

# homework #7

5.

```
nahyune@nahyune-virtual-machine: ~  
nahyune@nahyune-virtual-machine:~$ fg %1  
vi a  
nahyune@nahyune-virtual-machine:~$ jobs  
[2]-  Running                  sleep 10000 &  
[3]+  Stopped                  vi b  
nahyune@nahyune-virtual-machine:~$ fg  
vi b  
nahyune@nahyune-virtual-machine:~$ jobs  
[2]+  Running                  sleep 10000 &  
nahyune@nahyune-virtual-machine:~$ kill -9 $(ps | grep sleep | awk '{print $1}')nahyune@nahyune-virtual-machine:~$ jobs  
[2]+  Killed                   sleep 10000  
nahyune@nahyune-virtual-machine:~$
```

1. "ps | grep sleep | awk '{print \$1}'" 명령어를 사용하여 "sleep"이라는 이름을 가진 프로세스의 PID(Process ID)를 추출합니다. 여기서 "ps" 명령어는 현재 실행 중인 모든 프로세스를 나열하고, "grep sleep" 명령어는 이 중에서 "sleep"이라는 문자열을 포함하는 프로세스를 필터링하며, "awk '{print \$1}'" 명령어는 출력된 결과에서 첫 번째 필드인 PID만 추출합니다.

2. 이렇게 추출된 PID를 "kill -9" 명령어를 사용하여 강제 종료시킵니다. 여기서 "-9"는 SIGKILL 시그널을 보내어 프로세스를 즉시 중단시키는 것을 의미합니다.

# homework #7

6.

-(3)

```
peterpan@nahyune-virtual-machine:~$ at now
warning: commands will be executed using /bin/sh
at Thu May 11 14:59:00 2023
at> /bin/ls -l > ~/ls.out
at> <EOT>
job 1 at Thu May 11 14:59:00 2023
peterpan@nahyune-virtual-machine:~$ ls -l ls.out
-rw-rw-r-- 1 peterpan peterpan 63  5월 11 15:00 ls.out
peterpan@nahyune-virtual-machine:~$ cat ls.out
total 0
-rw-rw-r-- 1 peterpan peterpan 0  5월 11 15:00 ls.out
peterpan@nahyune-virtual-machine:~$
```

-(4)

```
peterpan@nahyune-virtual-machine:~$ echo '/bin/ls -l > ~/ls.out.1' > ls.sh
peterpan@nahyune-virtual-machine:~$ at now + 1 minutes < ls.sh
warning: commands will be executed using /bin/sh
job 21 at Thu May 11 15:56:00 2023
peterpan@nahyune-virtual-machine:~$ atq
21      Thu May 11 15:56:00 2023 a peterpan
peterpan@nahyune-virtual-machine:~$ atq
peterpan@nahyune-virtual-machine:~$ ls -l ls.out*
-rw-rw-r-- 1 peterpan peterpan 63  5월 11 15:45 ls.out
-rw-rw-r-- 1 peterpan peterpan 177  5월 11 15:56 ls.out.1
peterpan@nahyune-virtual-machine:~$
```

-(4).

1) 첫 명령어 즉, 1분이 지나기 전에 atq 명령어를 실행하면 작업번호, 예약시간의 출력이 나타난다. 왜냐하면 ls.sh 스크립트가 아직 실행되지 않았기 때문이다

그러나 1분이 지난 후에 atq 명령어를 실행하면, 예약이 실행된 후이기 때문에 아무것도 나타나지 않는다.

2) "at now + 1 minutes" 명령어는 현재 시간으로부터 1분 후를 지정하는 것이며, "<ls.sh" 명령어는 "ls.sh" 스크립트를 "at" 명령어로 지정한 시간에 실행하는 것을 의미한다

# homework #7

-(5).

```
peterpan@nahyune-virtual-machine: ~  
peterpan@nahyune-virtual-machine:~$ at 9:00 tomorrow < ls.sh  
warning: commands will be executed using /bin/sh  
job 22 at Fri May 12 09:00:00 2023  
peterpan@nahyune-virtual-machine:~$ at 13:30 19.06.04 < ls.sh  
at: refusing to create job destined in the past  
peterpan@nahyune-virtual-machine:~$ at 13:30 04.06.50 < ls.sh  
warning: commands will be executed using /bin/sh  
job 23 at Sat Jun 4 13:30:00 2050  
peterpan@nahyune-virtual-machine:~$ at 8:00 + 3days < ls.sh  
warning: commands will be executed using /bin/sh  
job 24 at Sun May 14 08:00:00 2023  
peterpan@nahyune-virtual-machine:~$ at 17:00 + 3 weeks < ls.sh  
warning: commands will be executed using /bin/sh  
job 25 at Thu Jun 1 17:00:00 2023  
peterpan@nahyune-virtual-machine:~$ atq  
25 Thu Jun 1 17:00:00 2023 a peterpan  
23 Sat Jun 4 13:30:00 2050 a peterpan  
24 Sun May 14 08:00:00 2023 a peterpan  
22 Fri May 12 09:00:00 2023 a peterpan  
peterpan@nahyune-virtual-machine:~$ sudo ls -l /var/spool/cron/atjobs  
total 16  
-rwx----- 1 peterpan daemon 2932 5월 11 16:01 a0001601ac39a0  
-rwx----- 1 peterpan daemon 2932 5월 11 16:02 a0001702856dce  
-rwx----- 1 peterpan daemon 2932 5월 11 16:02 a0001801ac44a4  
-rwx----- 1 peterpan daemon 2932 5월 11 16:03 a0001901acac00  
peterpan@nahyune-virtual-machine:~$
```

- 2050년을 지율려면 : atrm 18을 해주면 된다.

-> atq 명령어를 실행하였을 때 2050년으로 예약된 스케줄의 숫자가 18이기 때문이다.



# homework #7

7.  
-(2).

- “0 5 \* \* \* /bin/ps -ef > ~peterpan/ps.out”: 해당 명령어는 cron에 등록하여 매일 오전 5시에 /bin/ps -ef 명령어를 실행하고 결과를 peterpan 계정의 홈 디렉토리에 ps.out 파일로 저장하는 것을 의미한다.
- crontab -r 명령어는 현재 사용자의 crontab 파일을 삭제하는 명령어이다.

```
peterpan@nahyune-virtual-machine: ~  
peterpan@nahyune-virtual-machine:~$ sudo ls -l /var/spool/cron/crontabs  
total 0  
peterpan@nahyune-virtual-machine:~$ EDITOR=vi; export EDITOR  
peterpan@nahyune-virtual-machine:~$ crontab -e  
no crontab for peterpan - using an empty one  
crontab: installing new crontab  
peterpan@nahyune-virtual-machine:~$ sudo ls -l /var/spool/cron/crontabs  
total 4  
-rw----- 1 peterpan crontab 1131 5월 11 16:11 peterpan  
peterpan@nahyune-virtual-machine:~$ sudo tail -3 /var/spool/cron/crontabs/peterpan  
#  
# m h dom mon dow   command  
0 5 * * * /bin/ps -ef > ~peterpan/ps.out  
peterpan@nahyune-virtual-machine:~$ crontab -r  
peterpan@nahyune-virtual-machine:~$ sudo ls -l /var/spool/cron/crontabs  
total 0  
peterpan@nahyune-virtual-machine:~$
```

# homework #7

-(3).

```
peterpan@nahyune-virtual-machine: ~  
peterpan@nahyune-virtual-machine: $ sudo touch /etc/cron.deny  
peterpan@nahyune-virtual-machine: $ sudo vi /etc/cron.deny  
peterpan@nahyune-virtual-machine: $ crontab -e  
You (peterpan) are not allowed to use this program (crontab)  
See crontab(1) for more information  
peterpan@nahyune-virtual-machine: $ sudo rm /etc/cron.deny  
peterpan@nahyune-virtual-machine: $ crontab -e  
no crontab for peterpan - using an empty one  
No modification made  
peterpan@nahyune-virtual-machine: $
```

- “sudo touch /etc/cron.deny”:  
cron 서비스를 사용하는 사용자들  
중에서 cron 사용을 금지하고자 하는  
사용자의 목록을 작성하기 위해  
/etc/cron.deny 파일을 생성하는 것이  
다.  
즉 이 파일안에 사용을 금지하고자  
하는 peterpan을 적었으니 peterpan  
은 cron 서비스를 사용할 수 없다.  
따라서 첫 crontab -e는 권한이 거부  
되었고, 이 파일이 삭제된 후에야  
cron파일을 이용할 수 있게 된다.

# homework #7 - problems

1.

"sleep 100" 명령은 bash 셸에서 직접 실행됩니다. 이 명령을 실행하면 bash 셸은 새로운 프로세스를 생성하고, 이 프로세스는 "sleep" 명령을 실행하여 100초 동안 대기합니다. 이때, 새로운 프로세스는 부모 프로세스인 bash 셸과 별개로 실행되며, bash 셸이 이 명령을 실행하는 동안 다른 명령을 실행할 수 있습니다.

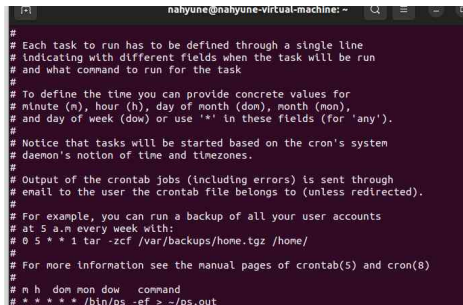
"exec sleep 100" 명령은 현재 실행 중인 bash 셸의 프로세스를 "sleep" 명령을 실행하는 새로운 프로세스로 교체합니다. 이때, bash 셸의 프로세스가 "sleep" 명령을 실행하는 새로운 프로세스로 교체되므로, 이후에 실행되는 명령어는 모두 "sleep" 명령을 실행하는 프로세스에서 실행됩니다.

```
nahyune@nahyune-virtual-machine:~$ bash
nahyune@nahyune-virtual-machine:~$ sleep 100
^Z
[1]+  Stopped                  sleep 100
nahyune@nahyune-virtual-machine:~$ exec sleep 100
^Z
[1]+  Stopped                  bash
nahyune@nahyune-virtual-machine:~$ ps -ef | grep bash
nahyune      2338      2320  0 12:51 pts/0    00:00:00 bash
nahyune      2793      2338  0 12:59 pts/0    00:00:00 grep --color=auto bash
nahyune@nahyune-virtual-machine:~$ ps -ef | grep sleep
nahyune      2347      2338  0 12:52 pts/0    00:00:00 sleep 100
nahyune      2353      2347  0 12:52 pts/0    00:00:00 [sleep] <defunct>
nahyune      2798      2338  0 12:59 pts/0    00:00:00 grep --color=auto sleep
```

->2338 이라는 같은 숫자가 "exec sleep 100" 명령이 같은 bash 셸에서 sleep 명령을 실행하는 새로운 프로세스로 교체하는 것을 보여준다

# homework #7 - problems

2.



```
nahyune@nahyune-virtual-machine: ~  
#  
# Each task to run has to be defined through a single line  
# indicating with different fields when the task will be run  
# and what command to run for the task  
#  
# To define the time you can provide concrete values for  
# minute (m), hour (h), day of month (dom), month (mon),  
# and day of week (dow) or use '*' in these fields (for 'any').  
#  
# Notice that tasks will be started based on the cron's system  
# daemon's notion of time and timezones.  
#  
# Output of the crontab jobs (including errors) is sent through  
# email to the user the crontab file belongs to (unless redirected).  
#  
# For example, you can run a backup of all your user accounts  
# at 5 a.m every week with:  
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/  
#  
# For more information see the manual pages of crontab(5) and cron(8)  
#  
# m h dom mon dow   command  
# * * * * * /bin/ps -ef > ~/ps.out
```

<방법>

1. 터미널을 열고 "crontab -e" 명령을 입력하여 crontab 파일을 연다

2. 에디터가 열리면 맨 마지막줄에 "\* \* \* \* \* /bin/ps -ef > ~/ps.out"를 작성하여 매 분마다 "/bin/ps -ef > ~/ps.out" 명령을 실행하도록 설정한다.

3. 파일을 수정한 후, "Ctrl + X" 키를 누르고, "Y" 키를 누르고, "Enter" 키를 눌러 crontab 파일을 저장합니다.

# homework #7 - problems

3.

```
nahyune@nahyune-virtual-machine:~$ crontab -l
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow  command
# * * * * * /bin/ps -ef > ~/ps.out
# 10,40 * * * * /bin/ps -ef > ~/ps.out
# 0 7 * * 1 /bin/ps -ef > ~/ps.out
# 0-50/2 9 1 * * /bin/ps -ef > ~/ps.out
# 13 17 * * 1-5 /bin/ps -ef > ~/ps.out
```

1) 10,40 \* \* \* \* /bin/ps -ef > ~/ps.out

2) 0 7 \* \* 1 /bin/ps -ef > ~/ps.out

3) 0-50/2 9 1 \* \* /bin/ps -ef > ~/ps.out

4) 13 17 \* \* 1-5 /bin/ps -ef > ~/ps.out