

lect2 HW

시스템프로그래밍 1 분반 12180626 성시열

5.1 Do following steps.

1) Log in to a Linux server. Find your login directory with "pwd". Find your user ID with "whoami".

```
login as: 12180626
12180626@165.246.38.151's password:
Last login: Sun Jun 19 21:58:58 2022 from 183.91.239.24
-bash-4.2$ pwd
/home/sp5/12180626
-bash-4.2$ whoami
12180626
```

학번을 통해 리눅스 서버에 로그인 한 뒤 로그인 디렉토리를 찾기 위해 pwd 명령어를 사용한다. pwd 는 present working directory 의 약자로 로그인한 위치를 알려주는 명령어인데 로그인 디렉토리가 /home/sp5/12180626 임을 확인하였다. whoami 명령어는 user ID 를 알려주는 명령어이다. whoami 를 통해 user ID 가 12180626 임을 확인하였다.

2) Go up the file tree with "..", display the current location with "pwd", and find other students directory name with "ls".

```
-bash-4.2$ cd ..
-bash-4.2$ pwd
/home/sp5
-bash-4.2$ ls
12161699 12170629 12171769 12181350 12181855 12192119 12211813
12161749 12171470 12180566 12181772 12190654 12200687 12211827
12170550 12171747 12180626 12181821 12191816 12211767 12213532
```

cd .. 명령어는 상위 디렉토리로 현재 위치를 이동시킨다. cd ..을 통해 상위 디렉토리로 이동한 뒤 pwd 명령어를 통해 현재 위치가 /home/sp5 임을 확인하였다. ls 명령어는 현재 디렉토리에 포함된 모든 파일들을 보여주는 명령어이다. ls 를 통해 /home/sp5 에 12161699 12170629 ... 12211767 12213532 등의 파일들이 존재함을 확인하였다.

3) Pick one of the student IDs (suppose it was 12345) and try to enter his/her directory with "cd".

```
-bash-4.2$ cd 12190654
-bash: cd: 12190654: Permission denied
```

ls 를 통해 확인한 파일 중 임의로 하나를 골라 cd 12190654 를 실행하였다. 여기서 cd 는 change dirctory 의 약자로 현재 위치를 바꿔주는 명령어이다. 12190654 로 위치를 변경하려 하였으나 접근 권한이 없어 접근할 수 없다는 Permission denied 메시지가 출력되었다.

4) Use "ls -l" to see the file permission of the directory you tried to enter.

```
-bash-4.2$ ls -l
total 84
drwx----- 2 12161699 12161699 4096 Jun 16 11:33 12161699
drwx----- 4 12161749 12161749 4096 Jun 20 09:32 12161749
drwx----- 3 12170550 12170550 4096 Jun 20 09:25 12170550
drwx----- 2 12170629 12170629 4096 Jun 16 11:33 12170629
drwx----- 2 12171470 12171470 4096 Jun 16 11:33 12171470
drwx----- 2 12171747 12171747 4096 Jun 20 09:29 12171747
drwx----- 3 12171769 12171769 4096 Jun 20 09:09 12171769
drwx----- 2 12180566 12180566 4096 Jun 16 11:33 12180566
drwx----- 2 12180626 12180626 4096 Jun 16 11:33 12180626
drwx----- 2 12181350 12181350 4096 Jun 17 23:38 12181350
drwx----- 3 12181772 12181772 4096 Jun 20 09:30 12181772
drwx----- 2 12181821 12181821 4096 Jun 17 23:38 12181821
drwx----- 2 12181855 12181855 4096 Jun 20 09:17 12181855
drwx----- 2 12190654 12190654 4096 Jun 16 11:33 12190654
drwx----- 6 12191816 12191816 4096 Jun 20 09:35 12191816
drwx----- 4 12192119 12192119 4096 Jun 20 00:38 12192119
drwx----- 2 12200687 12200687 4096 Jun 20 09:24 12200687
drwx----- 2 12211767 12211767 4096 Jun 16 11:33 12211767
drwx----- 3 12211813 12211813 4096 Jun 20 09:22 12211813
drwx----- 2 12211827 12211827 4096 Jun 17 23:38 12211827
drwx----- 2 12213532 12213532 4096 Jun 16 11:34 12213532
```

ls 명령어는 현재 위치에서의 모든 파일을 보여주는 명령어이다. 명령어에 옵션을 사용하기 위해서 '-' 기호를 사용하고 그 뒤에 옵션을 붙여주는데, ls -l 은 long 의 약자로 파일들의 세부사항까지 출력해주는 명령어이다. ls -l 을 통해 각 디렉토리의 owner 들은 rwx 권한이 있음을 확인하였다. 다른 사용자와 다른 구성원들은 그런 권한을 가지고 있지 않다. 12180626 파일의 정보를 확인해보면 user ID, group ID 와 파일이름이 12180626 임을 알 수 있고, 2 개의 연결된 파일을 가지고 있으며 6 월 16 일 11:33 에 만들었다는 것을 확인하였다. 크기는 4096 바이트이다.

5) Find out your current location again with "pwd".

```
-bash-4.2$ pwd
/home/sp5
```

현재 위치가 /home/sp5 임을 알 수 있다.

6) Go to the root directory with "cd /" and make sure you are really at the top directory with "pwd".

```
-bash-4.2$ cd /
-bash-4.2$ pwd
/
```

cd 명령어를 통해 현재 위치를 바꿀 수 있는데 cd /를 하게 되면 root 로 현재 위치를 이동한다. pwd 를 통해 현재 위치가 /(root)임을 알 수 있다.

7) How many files you have in the root directory? Some of them are not directory files. Find them with "ls -l". Use "file" command to see more detailed info.

```
-bash-4.2$ ls -l
total 66
drwxr-xr-x.  3 root root  4096 May  3  2013 app
lrwxrwxrwx.  1 root root    7 Feb 25  2013 bin -> usr/bin
dr-xr-xr-x.  4 root root  1024 Jun  4  2013 boot
drwxr-xr-x. 19 root root  3500 Jun 15 13:30 dev
drwxr-xr-x. 158 root root 12288 Jun 17 23:38 etc
drwxr-xr-x. 12 root root  4096 Jun 16 11:31 home
lrwxrwxrwx.  1 root root    7 Feb 25  2013 lib -> usr/lib
lrwxrwxrwx.  1 root root    9 Feb 25  2013 lib64 -> usr/lib64
drwx-----  2 root root 16384 Feb 25  2013 lost+found
drwxr-xr-x.  2 root root  4096 Jul 19  2012 media
drwxr-xr-x.  2 root root  4096 Jul 19  2012 mnt
drwxr-xr-x.  3 root root  4096 Jul 24  2013 opt
dr-xr-xr-x 252 root root    0 Jun 15 13:29 proc
dr-xr-x--- 12 root root  4096 Aug 30  2016 root
drwxr-xr-x  43 root root  1340 Jun 15 13:30 run
lrwxrwxrwx.  1 root root    8 Feb 25  2013/sbin -> usr/sbin
drwxr-xr-x.  2 root root  4096 Jul 19  2012 srv
dr-xr-xr-x 13 root root    0 Jun 15 13:29 sys
drwxrwxrwt.  9 root root   200 Jun 20 09:54 tmp
drwxr-xr-x. 13 root root  4096 Feb 25  2013 usr
drwxr-xr-x. 21 root root  4096 Feb 25  2013 var
-bash-4.2$ file *
app:      directory
bin:      symbolic link to `usr/bin'
boot:     directory
dev:      directory
etc:      directory
home:     directory
lib:      symbolic link to `usr/lib'
lib64:    symbolic link to `usr/lib64'
lost+found: directory
media:    directory
mnt:      directory
opt:      directory
proc:     directory
root:     directory
run:      directory
sbin:     symbolic link to `usr/sbin'
srv:      directory
sys:      directory
tmp:      sticky directory
usr:      directory
var:      directory
-bash-4.2$ file lib
lib:      symbolic link to `usr/lib'
```

ls -l을 통해 하위 디렉토리의 모든 파일의 세부사항까지 알 수 있다. 현재 위치가 root 이므로 root 의 모든 디렉토리를 파악할 수 있고, 총 파일의 개수는 21 개임을 알 수 있다.

file * 명령어는 file 의 type 을 알려주는 명령어인데, * 기호를 통해 모든 파일을 전부 매칭시키는 역할을 한다. 즉 모든 파일의 type 을 알려주는 명령어이다. file * 명령어를 통해 모든 파일의 type 을 확인하였다.

file lib 명령어는 다양한 파일 중 lib 의 file type 을 알 수 있다. file lib 명령어를 통해 usr/lib 파일에 연결된 symbolic type 임을 확인하였다.

8) * is a wild card character meaning it will be replaced by all file names in the current directory.

```
-bash-4.2$ file *
app:      directory
bin:      symbolic link to `usr/bin'
boot:     directory
dev:      directory
etc:      directory
home:     directory
lib:      symbolic link to `usr/lib'
lib64:    symbolic link to `usr/lib64'
lost+found: directory
media:    directory
mnt:      directory
opt:      directory
proc:     directory
root:     directory
run:      directory
sbin:     symbolic link to `usr/sbin'
srv:      directory
sys:      directory
tmp:      sticky directory
usr:      directory
var:      directory
```

```
-bash-4.2$ file app bin boot dev etc home lib lib64 lost+found media mnt opt pro
c root run sbin srv sys tmp usr var
app:      directory
bin:      symbolic link to `usr/bin'
boot:     directory
dev:      directory
etc:      directory
home:     directory
lib:      symbolic link to `usr/lib'
lib64:    symbolic link to `usr/lib64'
lost+found: directory
media:    directory
mnt:      directory
opt:      directory
proc:     directory
root:     directory
run:      directory
sbin:     symbolic link to `usr/sbin'
srv:      directory
sys:      directory
tmp:      sticky directory
usr:      directory
var:      directory
```

```
-bash-4.2$ file b*
bin:      symbolic link to `usr/bin'
boot:     directory
```

```
-bash-4.2$ file bin boot
bin:      symbolic link to `usr/bin'
boot:     directory
```

file *를 하였을 때 현재 위치가 root 이므로 모든 위치의 파일에 대한 type 을 확인할 수 있다. file app bin boot dev ... 와 같이 모든 파일의 이름을 넣는다면 file *과 마찬가지로 모든 파일에 대한 type 을 확인할 수 있다.

또한 file b* 명령어 중에서 b*는 b로 시작하는 파일 찾아내는 역할을 한다. 즉 b로 시작하는 파일들의 type을 알려주는 명령어이고, 실행 결과 bin, boot의 파일 type을 확인하였다. 이는 file bin boot와 같은 결과임을 알 수 있다.

9) If you want to go back to your login directory (suppose it was /home/sp1/12345), you can cd with absolute path, cd with relative path, or just "cd".

```
-bash-4.2$ pwd
/
-bash-4.2$ cd /home/sp5/12180626
-bash-4.2$ pwd
/home/sp5/12180626
-bash-4.2$ cd /
-bash-4.2$ pwd
/
-bash-4.2$ cd home/sp5/12180626
-bash-4.2$ pwd
/home/sp5/12180626
-bash-4.2$ cd /
-bash-4.2$ pwd
/
-bash-4.2$ cd
-bash-4.2$ pwd
/home/sp5/12180626
```

pwd를 통해 현재 디렉토리가 root임을 확인한다. 이후 cd를 통해 현재 위치를 변경하는데, 이 때 /로 파일 위치를 시작하여 root부터 차근차근 해당 파일 위치를 찾아간다. 이를 절대 경로라 한다. cd /home/sp5/12180626을 통해 현재 위치를 /home/sp5/12180626로 변경하고, pwd를 통해 현재 위치를 확인한다.

이후 다시 현재 디렉토리 위치를 root로 변경하고 현재 위치가 root임을 확인한다. 이번엔 cd home/sp5/12180626 명령어를 사용한다. 이 때 파일 위치가 /로 시작하지 않으므로 이를 상대경로라고 하고, 현재 디렉토리로부터 위치를 찾아나간다. 현재 위치가 root이므로 root부터 home, sp5, 12180626까지 차근차근 위치를 찾아나간다. pwd를 통해 현재 위치가 12180626임을 확인하였다.

다시 현재 디렉토리 위치를 root로 변경하고 현재 위치가 root임을 확인한다. 이후 cd 명령어를 입력한다. cd 명령어 뒤에 아무런 파일 위치를 입력하지 않으면 로그인 디렉토리로 이동한다는 것을 알 수 있다. pwd를 통해 현재 위치가 로그인 디렉토리임을 알 수 있습니다.

10) Confirm your current location with "pwd".

```
-bash-4.2$ pwd
/home/sp5/12180626
```

pwd를 통해 현재 위치를 파악할 수 있다. 현재 위치는 /home/sp5/12180626임을 확인하였다.

11) List all files in your directory with "ls" command.

```
-bash-4.2$ ls
```

ls 를 통해 현재 디렉토리에 포함된 모든 파일들을 볼 수 있다. 12180626 디렉토리 하위에는 따로 추가한 파일이 없으므로 아무것도 출력되지 않는다.

12) Try "echo" command.

```
-bash-4.2$ echo korea
korea
-bash-4.2$ echo hello
hello
```

echo 명령어는 echo 뒤의 문자열을 출력해주는 명령어이다. echo korea, echo hello 를 실행하여 korea 와 hello 문자열을 출력한다.

11) Try "echo" with ">" symbol. ">" is called "standard output redirection".

```
-bash-4.2$ echo hello > f1
```

echo 명령어는 echo 뒤에 문자열을 출력해주는 명령어였지만 '>' 기호와 함께 사용하면 뒤 문자열을 출력하지 않고 '>' 뒤의 문자열의 이름을 가진 파일을 생성하고, echo 뒤의 문자열을 해당 파일의 내용으로 입력하는 명령어이다. 위 실행문을 통해 f1 파일을 생성하고 해당 파일의 내용으로 hello 가 입력된다.

12) Do "ls" to see you can find f1 in the current directory. Show its content with "cat".

```
-bash-4.2$ ls
f1
-bash-4.2$ cat f1
hello
```

ls 명령어를 통해 현재 디렉토리의 모든 파일들을 보여준다. 이전 문제에서 생성해준 f1 파일이 있음을 확인하였다. cat 명령어는 cat 뒤에 나오는 파일의 내용을 출력하는 명령어이다. cat f1 을 통해 f1 파일 내용이 hello 임을 알 수 있고, 이전 문제에서 넣어줬던 내용이 들어가있음을 확인할 수 있다.

13) Make a directory, d1, with "mkdir".

```
-bash-4.2$ mkdir d1
-bash-4.2$ ls
d1 f1
```

mkdir 명령어는 make directory 의 약자로 디렉토리를 생성하는 명령어이다.

d1 이라는 디렉토리를 생성하고 ls 를 통해 d1 디렉토리가 생성되었음을 확인하였다. f1 과 달리 d1 은 파일이 아닌 디렉토리이므로 파란색으로 나와있는 것을 확인할 수 있다.

14) Copy f1 into directory d1.

```
-bash-4.2$ cp f1 d1
-bash-4.2$ cd d1
-bash-4.2$ ls
f1
-bash-4.2$ cat f1
hello
-bash-4.2$ cd ..
-bash-4.2$ cp f1 d1/f2
-bash-4.2$ cd d1
-bash-4.2$ ls
f1 f2
-bash-4.2$ cat f2
hello
```

cp 명령어는 파일을 디렉토리 내부로 복사하는 명령어이다.

cp f1 d1 명령어를 통해 f1 파일을 d1 디렉토리 내부로 복사하였고, cd 명령어를 통해 d1 으로 이동한 뒤 ls 명령어를 통해 d1 디렉토리에 f1 파일이 있음을 확인하였다.

cat f1 명령어를 통해 f1 파일 내용이 hello 임을 알 수 있다. cd.. 명령어를 통해 상위 디렉토리로 현재 위치를 이동시킨다.

cp f1 d1/f2 는 f1 파일을 d1 디렉토리에 복사하되 f1 파일의 내용을 f2 라는 파일을 생성하고 거기에 붙여넣는 방식이다. cd d1 명령어를 통해 d1 로 이동한 뒤, ls 를 통해 파일을 파악하면 f1, f2 두 개의 파일이 생성되어있음을 알 수 있다. 또한 cat f2 명령어를 통해 f2 의 파일 내용을 확인해보면 f1 의 내용이 복사되었으므로 똑같이 hello 가 내용에 들어가 있음을 알 수 있다.

5.2 Do followings and explain what happens and why.

\$ cd

```
-bash-4.2$ cd
```

cd 명령어를 통해 로그인 디렉토리로 이동한다. 현재 위치는 /home/sp5/12180626 일 것이다.

\$ ls

```
-bash-4.2$ ls
d1 f1
```

ls 명령어를 통해 12180626 디렉토리 하위의 있는 모든 파일들을 확인한다. 12180626 디렉토리 하위의 d1 디렉토리와 f1 파일이 있음을 알 수 있다.

\$ ls -l

```
-bash-4.2$ ls -l
total 8
drwxrwxr-x 2 12180626 12180626 4096 Jun 20 10:50 d1
-rw-rw-r-- 1 12180626 12180626    6 Jun 20 10:38 f1
```

ls -l 명령어를 통해 d1 디렉토리와 f1 파일의 세부사항까지 모두 확인한다.

\$ ls -al

```
-bash-4.2$ ls -al
total 16
drwx----- 3 12180626 12180626 4096 Jun 20 10:46 .
drwxr-xr-x 23 root      root    4096 Jun 17 23:38 ..
drwxrwxr-x 2 12180626 12180626 4096 Jun 20 10:50 d1
-rw-rw-r-- 1 12180626 12180626    6 Jun 20 10:38 f1
```

ls -al 명령어에서 a 옵션이 hidden 파일까지 모두 보여주는 옵션이다. ls-al 을 통해 이전에 보이지 않던 hidden 파일의 세부사항까지 모두 출력해준다.

\$ cd /

```
-bash-4.2$ cd /
```

cd 명령어를 통해 현재 디렉토리의 위치를 변경한다. cd /를 통해 root 로 이동한다.

\$ cd bin

```
-bash-4.2$ cd bin
```

cd bin 을 통해 bin 디렉토리로 이동한다. 이 때 bin 은 상대 참조를 사용하여 현재 있는 위치(root)를 기준으로 bin 디렉토리를 찾아나간다.


```
-bash-4.2$ ls
[lslocks
a2p      lspgpot
ab       lss3
abrt-action-analyze-backtrace  lsusb
abrt-action-analyze-c         lsusb.py
abrt-action-analyze-ccpp-local ltrace
abrt-action-analyze-core      lua
abrt-action-analyze-oops      luac
abrt-action-analyze-python     lupdate-qt4
abrt-action-analyze-vmcore     lwp-download
abrt-action-analyze-xorg       lwp-dump
abrt-action-generate-backtrace lwp-mirror
abrt-action-generate-core-backtrace lwp-request
abrt-action-install-debuginfo  lz
```

ls 를 통해 현재 위치인 bin 의 디렉토리의 모든 파일을 출력해준다.

```
-bash-4.2$ ls bz*
bzcat  bzcmp  bzdiff  bzgrep  bzip2  bzip2recover  bzless  bzmores  bzip
```

```
-bash-4.2$ cd
```

```
-bash-4.2$ pwd
/home/sp5/12180626
```

pwd 를 통해 현재 위치가 /home/sp5/12180626(로그인 디렉토리)임을 알 수 있다.

\$ man ls

```
-bash-4.2$ man ls
LS(1)                                User Commands                                LS(1)

NAME
    ls - list directory contents

SYNOPSIS
    ls [OPTION]... [FILE]...

DESCRIPTION
    List information about the FILES (the current directory by default).
    Sort entries alphabetically if none of -cftuvSUX nor --sort is speci-
    fied.

    Mandatory arguments to long options are mandatory for short options
    too.

    -a, --all
        do not ignore entries starting with .

    -A, --almost-all
        do not list implied . and ..

    --author
        with -l, print the author of each file

    -b, --escape
```

(출력 결과가 너무 길어 일부 생략하여 첨부하였습니다.)

man 은 커맨드에 대한 설명을 보여주는 커맨드이다. man ls 를 실행하여 ls 와 관련된 커맨드의 설명을 확인할 수 있다. q 를 누르면 출력결과가 닫을 수 있다.

\$ echo hello

```
-bash-4.2$ echo hello
hello
```

echo hello 를 통해 hello 라는 문자열을 스크린에 출력하였다.

\$ echo hello > f4

```
-bash-4.2$ echo hello > f4
```

echo hello > f4 를 통해 hello 를 스크린에 출력하지 않고, f4 라는 파일을 생성한 뒤 해당 파일의 내용으로 hello 가 입력된다.

\$ ls

```
-bash-4.2$ ls
dl  f1  f4
```

현재 위치가 12180626(로그인 디렉토리)이므로 로그인 디렉토리의 모든 파일들을 출력한다.

\$ cp f4 f2

```
-bash-4.2$ cp f4 f2
```

cp f4 f2 를 통해 f4 의 내용을 복사하여 f2 에 붙여 넣는다.

\$ cat f4

```
-bash-4.2$ cat f4  
hello
```

cat f4 를 통해 f4 의 내용을 확인하면 f2 에서 복사된 hello 라는 파일 내용이 그대로 f4 에 들어가 있음을 알 수 있다.

\$ cat f2

```
-bash-4.2$ cat f2  
hello
```

cat f2 를 통해 f2 의 내용을 확인하면 hello 가 그대로 있음을 알 수 있다.

\$ cat f2 > f3

```
-bash-4.2$ cat f2 > f3
```

cat f2 > f3 는 '>'를 통해 f2 의 내용을 그대로 복사하여 f3 를 생성하고 내용을 붙여넣는 명령어이다. 위 실행문을 통해 f3 라는 파일이 생성되고, f2 의 내용을 복사한 내용이 f3 에 들어가 있을 것이다.

\$ ls -l f*

```
-bash-4.2$ ls -l f*  
-rw-rw-r-- 1 12180626 12180626 6 Jun 20 10:38 f1  
-rw-rw-r-- 1 12180626 12180626 6 Jun 20 11:17 f2  
-rw-rw-r-- 1 12180626 12180626 6 Jun 20 11:18 f3  
-rw-rw-r-- 1 12180626 12180626 6 Jun 20 11:17 f4
```

ls -l 을 통해 현재 디렉토리에 존재하는 파일의 세부사항들을 파악할 수 있는데 f*를 추가하여 f로 시작하는 파일들만 출력한다. 출력결과를 보면 f3 가 생성되었음을 알 수 있다.

\$ rm f2

```
-bash-4.2$ rm f2
```

rm 명령어는 remove 의 약자로 파일을 삭제하는 명령어이다. rm f2 를 통해 f2 파일을 삭제한다.

\$ ls

```
-bash-4.2$ ls  
d1 f1 f3 f4
```

ls 를 통해 12180626 디렉토리에 존재하는 파일들이 d1 디렉토리와 f1, f3, f4 임을 확인하였다. f2 파일이 삭제됨을 알 수 있다.

\$ cat f4

```
-bash-4.2$ cat f4  
hello
```

cat f4 를 통해 f4 의 파일 내용이 hello 임을 확인하였다.

\$ xxd f4

```
-bash-4.2$ xxd f4
00000000: 6865 6c6c 6f0a                hello.
```

xxd 명령어는 파일내용을 16 진수로 표현하는 명령어이다. xxd f4 를 통해 hello 라는 문자열이 16 진수로 표현됨을 확인할 수 있다.

\$ mkdir d2

```
-bash-4.2$ mkdir d2
```

mkdir d2 를 통해 d2 라는 디렉토리를 생성한다.

\$ cp f4 d2

```
-bash-4.2$ cp f4 d2
```

cp f4 d2 를 통해 f4 라는 파일을 복사하여 d2 라는 디렉토리에 d2 에 붙여넣기한다.

\$ cd d2

```
-bash-4.2$ cd d2
```

cd d2 를 통해 현재 위치를 d2 로 이동시킨다.

\$ pwd

```
-bash-4.2$ pwd
/home/sp5/12180626/d2
```

pwd 를 통해 현재 위치가 /home/sp5/12180626/d2 임을 확인하였다.

\$ ls

```
-bash-4.2$ ls
f4
```

ls 를 통해 d2 라는 디렉토리에 있는 모든 파일을 출력한다. d2 디렉토리 안에 f4 파일이 있다는 것을 확인하였다.

\$ cd ..

```
-bash-4.2$ cd ..
```

cd ..을 통해 상위 디렉토리로 이동한다. 현재 d2 디렉토리의 상위 디렉토리인 12180626 디렉토리로 이동하였을 것이다.

```
$ grep -nr "he" *
```

```
-bash-4.2$ grep -nr "he" *
d1/f2:1:hello
d1/f1:1:hello
d2/f4:1:hello
f1:1:hello
f3:1:hello
f4:1:hello
```

grep 명령어는 문자열을 찾아주는 명령어이다. "he"를 통해 he가 포함된 파일을 찾을 수 있다. '-nr'은 옵션인데, '-n'옵션은 찾고싶은 문자열이 발견된 line의 번호를 알 수 찾아내고, '-r'옵션은 하위 디렉토리를 내려가면서 모두 찾아내는 옵션이다. '*'을 통해 현재 디렉토리 내의 모든 파일에 대해 탐색한다.

```
$ ps
```

```
-bash-4.2$ ps
  PID TTY          TIME CMD
26414 pts/5        00:00:00 bash
26905 pts/5        00:00:00 ps
```

ps 명령어는 모든 process에 대한 설명을 출력하는 명령어이다.

PID : processes ID

TTY : process의 터미널 ID

TIME : process에 소요된 시간

CMD : process의 실행 이름

```
$ ps -ef
```

```
-bash-4.2$ ps -ef
UID          PID    PPID  C STIME TTY          TIME CMD
root           1         0  0 Jun15 ?        00:00:05 /usr/lib/systemd/systemd --switch
root           2         0  0 Jun15 ?        00:00:00 [kthreadd]
root           3         2  0 Jun15 ?        00:00:00 [ksoftirqd/0]
root           5         2  0 Jun15 ?        00:00:00 [kworker/0:0H]
root           7         2  0 Jun15 ?        00:00:00 [kworker/u:0H]
root           8         2  0 Jun15 ?        00:00:00 [migration/0]
root           9         2  0 Jun15 ?        00:00:00 [rcu_bh]
root          10         2  0 Jun15 ?        00:00:54 [rcu_sched]
root          11         2  0 Jun15 ?        00:00:00 [watchdog/0]
root          12         2  0 Jun15 ?        00:00:00 [watchdog/1]
```

(출력 결과가 너무 길어 일부 생략하여 첨부하였습니다.)

ps -ef는 ps에 '-ef'옵션이 추가된 것이다. ps -ef를 통해 모든 사용자의 모든 process를 표시해준다.

\$ ps -ef | more

```
-bash-4.2$ ps -ef | more
UID          PID  PPID  C STIME TTY          TIME CMD
root           1      0  0 Jun15 ?        00:00:05 /usr/lib/systemd/systemd --switch
ed-root --system --deserialize 23
root           2      0  0 Jun15 ?        00:00:00 [kthreadd]
root           3      2  0 Jun15 ?        00:00:00 [ksoftirqd/0]
root           5      2  0 Jun15 ?        00:00:00 [kworker/0:0H]
root           7      2  0 Jun15 ?        00:00:00 [kworker/u:0H]
root           8      2  0 Jun15 ?        00:00:00 [migration/0]
root           9      2  0 Jun15 ?        00:00:00 [rcu_bh]
root          10      2  0 Jun15 ?        00:00:54 [rcu_sched]
root          11      2  0 Jun15 ?        00:00:00 [watchdog/0]
root          12      2  0 Jun15 ?        00:00:00 [watchdog/1]
root          13      2  0 Jun15 ?        00:00:00 [migration/1]
root          14      2  0 Jun15 ?        00:00:00 [ksoftirqd/1]
root          16      2  0 Jun15 ?        00:00:00 [kworker/1:0H]
root          17      2  0 Jun15 ?        00:00:01 [watchdog/2]
root          18      2  0 Jun15 ?        00:00:00 [migration/2]
root          19      2  0 Jun15 ?        00:00:00 [ksoftirqd/2]
root          21      2  0 Jun15 ?        00:00:00 [kworker/2:0H]
root          22      2  0 Jun15 ?        00:00:01 [watchdog/3]
root          23      2  0 Jun15 ?        00:00:00 [migration/3]
root          24      2  0 Jun15 ?        00:00:00 [ksoftirqd/3]
root          26      2  0 Jun15 ?        00:00:00 [kworker/3:0H]
--More--
```

ps -ef 의 목록이 매우 많으므로 | more 을 추가해주어 결과값을 화면별로 보여줄 수 있다.(enter 키 위에 있는 '|'임에 주의)

5.3 Run following commands and explain what happens.

1) chmod

```
-bash-4.2$ chmod
chmod: missing operand
Try 'chmod --help' for more information.
```

chmod 명령어를 실행시켜본 결과, 피연산자가 누락되었다는 메시지가 출력되었다.
더 많은 정보를 위해 chmod --help를 실행 시켜보았다.

```
-bash-4.2$ chmod --help
Usage: chmod [OPTION]... MODE[,MODE]... FILE...
       or:  chmod [OPTION]... OCTAL-MODE FILE...
       or:  chmod [OPTION]... --reference=RFILE FILE...
Change the mode of each FILE to MODE.
With --reference, change the mode of each FILE to that of RFILE.

  -c, --changes           like verbose but report only when a change is made
  -f, --silent, --quiet  suppress most error messages
  -v, --verbose           output a diagnostic for every file processed
                        --no-preserve-root  do not treat '/' specially (the default)
                        --preserve-root   fail to operate recursively on '/'
                        --reference=RFILE  use RFILE's mode instead of MODE values
  -R, --recursive        change files and directories recursively
      --help             display this help and exit
      --version          output version information and exit

Each MODE is of the form '[ugoa]*([-+=]([rwxXst]*|[ugo]))+|[-+=][0-7]+'.
```

Report chmod bugs to bug-coreutils@gnu.org
GNU coreutils home page: <http://www.gnu.org/software/coreutils/>
General help using GNU software: <http://www.gnu.org/gethelp/>
For complete documentation, run: `info coreutils 'chmod invocation'`

chmod는 파일 권한을 변경할 수 있는 명령어로 chmod 000 [파일이름]을 통해 파일의 권한을 삭제할 수 있다.

```
-bash-4.2$ ls -l
total 20
drwxrwxr-x 2 12180626 12180626 4096 Jun 20 10:50 d1
drwxrwxr-x 2 12180626 12180626 4096 Jun 20 13:18 d2
-rw-rw-r-- 1 12180626 12180626    6 Jun 20 10:38 f1
-rw-rw-r-- 1 12180626 12180626    6 Jun 20 11:18 f3
-rw-rw-r-- 1 12180626 12180626    6 Jun 20 11:17 f4
-bash-4.2$ chmod 000 d1
-bash-4.2$ ls -l
total 20
d----- 2 12180626 12180626 4096 Jun 20 10:50 d1
drwxrwxr-x 2 12180626 12180626 4096 Jun 20 13:18 d2
-rw-rw-r-- 1 12180626 12180626    6 Jun 20 10:38 f1
-rw-rw-r-- 1 12180626 12180626    6 Jun 20 11:18 f3
-rw-rw-r-- 1 12180626 12180626    6 Jun 20 11:17 f4
```

ls -l을 통해 d1, d2, f1, f3, f4 파일들에 어떤 권한들이 있는지 확인하였다. 이후 chmod 000 d1을 통해 d1의 권한을 제거한 뒤 다시 ls -l을 통해 파일들의 권한을 확인한 결과, d1의 권한이 삭제된 것을 확인하였다.

2) clear

```
165.246.38.151 - PuTTY
or: chmod [OPTION]... OCTAL-MODE FILE...
or: chmod [OPTION]... --reference=RFILE FILE...
Change the mode of each FILE to MODE.
With --reference, change the mode of each FILE to that of RFILE.

-c, --changes          like verbose but report only when a change is made
-f, --silent, --quiet  suppress most error messages
-v, --verbose          output a diagnostic for every file processed
--no-preserve-root     do not treat '/' specially (the default)
--preserve-root        fail to operate recursively on '/'
--reference=RFILE      use RFILE's mode instead of MODE values
-R, --recursive        change files and directories recursively
--help                display this help and exit
--version              output version information and exit

Each MODE is of the form '[ugoa]*([-+]=([rwxXst]*|[ugo]))+|[-+]=[0-7]+'

Report chmod bugs to bug-coreutils@gnu.org
GNU coreutils home page: <http://www.gnu.org/software/coreutils/>
General help using GNU software: <http://www.gnu.org/gethelp/>
For complete documentation, run: info coreutils 'chmod invocation'
-bash-4.2$ ls -l
total 20
drwxrwxr-x 2 12180626 12180626 4096 Jun 20 10:50 d1
drwxrwxr-x 2 12180626 12180626 4096 Jun 20 13:18 d2
-rw-rw-r-- 1 12180626 12180626    6 Jun 20 10:38 f1
-rw-rw-r-- 1 12180626 12180626    6 Jun 20 11:18 f3
-rw-rw-r-- 1 12180626 12180626    6 Jun 20 11:17 f4
-bash-4.2$ chmod 000 d1
-bash-4.2$ ls -l
total 20
d----- 2 12180626 12180626 4096 Jun 20 10:50 d1
drwxrwxr-x 2 12180626 12180626 4096 Jun 20 13:18 d2
-rw-rw-r-- 1 12180626 12180626    6 Jun 20 10:38 f1
-rw-rw-r-- 1 12180626 12180626    6 Jun 20 11:18 f3
-rw-rw-r-- 1 12180626 12180626    6 Jun 20 11:17 f4
-bash-4.2$
```

clear 를 하기 전 전체 화면의 모습이다.

```
165.246.38.151 - PuTTY
-bash-4.2$
```

clear 를 입력함과 동시에 모든 내용이 사라진 것을 확인하였다.

3) gzip

```
-bash-4.2$ gzip
gzip: compressed data not written to a terminal. Use -f to force compression.
For help, type: gzip -h
-bash-4.2$ gzip f3
-bash-4.2$ ls -l
total 20
d----- 2 12180626 12180626 4096 Jun 20 10:50 d1
drwxrwxr-x 2 12180626 12180626 4096 Jun 20 13:18 d2
-rw-rw-r-- 1 12180626 12180626    6 Jun 20 10:38 f1
-rw-rw-r-- 1 12180626 12180626   29 Jun 20 11:18 f3.gz
-rw-rw-r-- 1 12180626 12180626    6 Jun 20 11:17 f4
-bash-4.2$
```

gzip 은 파일이 압축되어있음을 확인하는 명령어이다. gzip f3 를 통해 파일 f3 를 압축하였고, ls -l 을 통해 f3 가 압축되어있음을 확인하였다.

```
-bash-4.2$ gzip d1
gzip: d1: Permission denied
-bash-4.2$ gzip d2
gzip: d2 is a directory -- ignored
```

gzip d1 을 실행할 때는 권한이 없으므로 압축이 되지 않았고,
gzip d2 을 실행할 때는 압축의 대상이 directory 일 경우에 압축이 되지 않음을 확인하였다.

4) date

```
-bash-4.2$ date
Mon Jun 20 14:28:16 KST 2022
```

date 는 아래와 같이 현재 시각 및 날짜를 출력해주는 명령어이다.

5) dd

```
-bash-4.2$ cd d2
-bash-4.2$ ls
f4
-bash-4.2$ dd if=f4 of=d2 bs=512
0+1 records in
0+1 records out
6 bytes (6 B) copied, 5.1715e-05 s, 116 kB/s
```

dd 는 블록 단위로 파일을 복사하거나 파일 변환을 할 수 있는 명령어이다.

dd if=[원본 FILE] of=[저장할 FILE 경로] bs=[block SIZE]의 형태로 사용한다.

상대참조를 사용하기 위해 cd d2 를 통해 현재 디렉토리를 변경하고, ls 를 통해 d2 디렉토리에 f4 파일이 존재하는 것을 확인하였다.

이후 dd if=f4 of=d2 bs=512 를 통해 f4 의 파일은 d2 디렉토리에 복사하여 저장하고 block SIZE 는 512 로 설정한다.

```
-bash-4.2$ ls
d2 f4
```

잘 복사되었는지 확인하기 위해 ls 명령어를 사용하면 d2 가 추가되어있음을 알 수 있다.

```
-bash-4.2$ cat d2
hello
-bash-4.2$ cat f4
hello
```

cat d2 와 cat f4 를 통해 파일내용이 복사가 되었음을 확인하였다.

6) df

```
-bash-4.2$ df
Filesystem            1K-blocks      Used Available Use% Mounted on
devtmpfs               3926452         0   3926452  0% /dev
tmpfs                  3938652         84   3938568  1% /dev/shm
tmpfs                  3938652    137672   3800980  4% /run
tmpfs                  3938652         0   3938652  0% /sys/fs/cgroup
p
/dev/mapper/fedora_linuxer1-root 51475068 28382376 20471252  59% /
tmpfs                  3938652         8   3938644  1% /tmp
/dev/sda3              487652      99620   362432  22% /boot
/dev/mapper/fedora_linuxer1-home 168737864 3696072 156463744  3% /home
```

df 명령어는 하드디스크의 용량을 확인할 수 있는 명령어이다.

```
-bash-4.2$ df -h
Filesystem            Size  Used Avail Use% Mounted on
devtmpfs              3.8G     0  3.8G   0% /dev
tmpfs                 3.8G   84K  3.8G   1% /dev/shm
tmpfs                 3.8G  135M  3.7G   4% /run
tmpfs                 3.8G     0  3.8G   0% /sys/fs/cgroup
/dev/mapper/fedora_linuxer1-root 50G   28G   20G  59% /
tmpfs                 3.8G   8.0K  3.8G   1% /tmp
/dev/sda3             477M   98M  354M  22% /boot
/dev/mapper/fedora_linuxer1-home 161G   3.6G  150G   3% /home
```

'-h' 옵션을 붙여 사용한다면 각 수치들에 단위를 붙여 하드디스크의 용량을 파악할 수 있다.

```
-bash-4.2$ df -a
Filesystem            1K-blocks      Used Available Use% Mounted on
rootfs               51475068 28385960 20467668  59% /
proc                  0         0         0    - /proc
sysfs                 0         0         0    - /sys
devtmpfs             3926452         0   3926452  0% /dev
securityfs            0         0         0    - /sys/kernel/s
ecurity
tmpfs                 3938652         84   3938568  1% /dev/shm
devpts                0         0         0    - /dev/pts
tmpfs                 3938652    141748   3796904  4% /run
tmpfs                 3938652         0   3938652  0% /sys/fs/cgroup
p
cgroup                0         0         0    - /sys/fs/cgroup
p/systemd
pstore               0         0         0    - /sys/fs/pstor
e
cgroup                0         0         0    - /sys/fs/cgroup
p/cpuset
cgroup                0         0         0    - /sys/fs/cgroup
p/cpu,cpuacct
cgroup                0         0         0    - /sys/fs/cgroup
p/memory
cgroup                0         0         0    - /sys/fs/cgroup
p/devices
```

(출력 결과가 너무 길어 일부 생략하여 첨부하였습니다.)

'-a' 옵션을 붙여 사용한다면 모든 파일시스템의 하드디스크 용량을 파악할 수 있다.

7) diff

```
-bash-4.2$ cat f1
hello
-bash-4.2$ cat f4
hello
-bash-4.2$ diff f1 f4
```

diff 는 두 개의 파일을 비교해주는 명령어이다. cat f1 와 cat f4 를 통해 파일내용을 알 수 있었다. diff f1 f4 를 통해 두 파일 내용을 비교하고 같다면 아무런 값도 출력되지 않는다.

8) dmesg

```
-bash-4.2$ dmesg
[    0.000000] Initializing cgroup subsys cpuset
[    0.000000] Initializing cgroup subsys cpu
[    0.000000] Linux version 3.9.4-200.fc18.x86_64 (mockbuild@bkernel02) (gcc version 4.7.2 20121109 (Red Hat 4.7.2-8) (GCC) ) #1 SMP Fri May 24 20:10:49 UTC 2013
[    0.000000] Command line: BOOT_IMAGE=/vmlinuz-3.9.4-200.fc18.x86_64 root=/dev/mapper/fedora_linuxer1-root ro rd.md=0 rd.dm=0 rd.lvm.lv=fedora_linuxer1/swap rd.lvm.lv=fedora_linuxer1/root rd.luks=0 vconsole.keymap=us rhgb quiet LANG=en_US.UTF-8
[    0.000000] e820: BIOS-provided physical RAM map:
[    0.000000] BIOS-e820: [mem 0x0000000000000000-0x00000000000009d7ff] usable
[    0.000000] BIOS-e820: [mem 0x00000000000009d800-0x00000000000009ffff] reserved
[    0.000000] BIOS-e820: [mem 0x0000000000000e0000-0x0000000000000fffff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000000100000-0x000000000001fffff] usable
[    0.000000] BIOS-e820: [mem 0x00000000020000000-0x000000000201fffff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000020200000-0x000000000203fffff] usable
[    0.000000] BIOS-e820: [mem 0x00000000040000000-0x000000000401fffff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000040200000-0x00000000040cde01fff] usable
[    0.000000] BIOS-e820: [mem 0x00000000040cde02000-0x00000000040ce4c2fff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000040ce4c3000-0x00000000040ce742fff] ACPI NVS
[    0.000000] BIOS-e820: [mem 0x00000000040ce743000-0x00000000040ce747fff] ACPI data
[    0.000000] BIOS-e820: [mem 0x00000000040ce748000-0x00000000040ce78afff] ACPI NVS
[    0.000000] BIOS-e820: [mem 0x00000000040ce78b000-0x00000000040cedf5fff] usable
[    0.000000] BIOS-e820: [mem 0x00000000040cedf6000-0x00000000040ceff2fff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000040ceff3000-0x00000000040cefffff] usable
[    0.000000] BIOS-e820: [mem 0x00000000040cf800000-0x00000000040df9fffff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000040f8000000-0x00000000040fbfffff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000040fec00000-0x00000000040fec00fff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000040fed00000-0x00000000040fed03fff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000040fed1c000-0x00000000040fed1ffff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000040fee00000-0x00000000040fee00fff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000040ff000000-0x00000000040fffff] reserved
[    0.000000] BIOS-e820: [mem 0x0000000004010000000-0x0000000004021f5ffff] usable
[    0.000000] NX (Execute Disable) protection: active
[    0.000000] SMBIOS 2.7 present.
[    0.000000] DMI: To be filled by O.E.M. To be filled by O.E.M./H77MXV/H77MXV-D, BIOS BB4F1P02 03/13/2012
[    0.000000] e820: update [mem 0x000000000-0x000000fff] usable ==> reserved
[    0.000000] e820: remove [mem 0x000a00000-0x000fffff] usable
[    0.000000] No AGP bridge found
[    0.000000] e820: last_pfn = 0x21f600 max_arch_pfn = 0x400000000
[    0.000000] MTRR default type: uncachable
```

(출력 결과가 너무 길어 일부 생략하여 첨부하였습니다.)

dmesg 는 시스템 부팅메시지를 출력한다.

9) du

```
-bash-4.2$ du
12 .
```

du 는 파일과 디렉토리의 용량을 알려주는 명령어이다. 현 위치가 d2 이므로 d2 디렉토리의 용량을 알 수 있다.

```
-bash-4.2$ du -k
12 .
```

du -k 는 기본단위를 1KB 로 해주는 옵션이다. 이를 통해 d2 디렉토리의 용량이 12KB 임을 확인할 수 있었다.

10) env

```
-bash-4.2$ env
XDG_SESSION_ID=218
HOSTNAME=linuxer1.localdomain
TERM=xterm
SHELL=/bin/bash
HISTSIZE=1000
SSH_CLIENT=165.246.104.245 8474 22
SSH_TTY=/dev/pts/5
QT_GRAPHICSSYSTEM_CHECKED=1
USER=12180626
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or=40;31;01:mi=01;05;37;41:su=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz=01;31:*.arj=01;31:*.taz=01;31:*.lzh=01;31:*.lзма=01;31:*.tlz=01;31:*.txz=01;31:*.zip=01;31:*.z=01;31:*.Z=01;31:*.dz=01;31:*.gz=01;31:*.lz=01;31:*.xz=01;31:*.bz2=01;31:*.tbz=01;31:*.tbz2=01;31:*.bz=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01;31:*.jar=01;31:*.war=01;31:*.ear=01;31:*.sar=01;31:*.rar=01;31:*.ace=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;31:*.rz=01;31:*.jpg=01;35:*.jpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35:*.ppm=01;35:*.tga=01;35:*.xbm=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svgz=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35:*.m2v=01;35:*.mkv=01;35:*.ogm=01;35:*.mp4=01;35:*.m4v=01;35:*.mp4v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:*.asf=01;35:*.rm=01;35:*.rmvb=01;35:*.flc=01;35:*.avi=01;35:*.fli=01;35:*.flv=01;35:*.gl=01;35:*.dl=01;35:*.xcf=01;35:*.xwd=01;35:*.yuv=01;35:*.cgm=01;35:*.emf=01;35:*.axv=01;35:*.anx=01;35:*.ogv=01;35:*.ogx=01;35:*.aac=01;36:*.au=01;36:*.flac=01;36:*.mid=01;36:*.midi=01;36:*.mka=01;36:*.mp3=01;36:*.mpc=01;36:*.ogg=01;36:*.ra=01;36:*.wav=01;36:*.axa=01;36:*.oga=01;36:*.spx=01;36:*.xspf=01;36:
MAIL=/var/spool/mail/12180626
PATH=/usr/lib64/ccache:/usr/local/bin:/usr/bin:/usr/local/sbin:/usr/sbin
PWD=/home/sp5/12180626/d2
LANG=en_US.UTF-8
KDE_IS_PRELINKED=1
KDEDIRS=/usr
HISTCONTROL=ignoredups
SHLVL=1
HOME=/home/sp5/12180626
LOGNAME=12180626
CVS_RSH=ssh
SSH_CONNECTION=165.246.104.245 8474 165.246.38.151 22
LESSOPEN=||/usr/bin/lesspipe.sh %s
XDG_RUNTIME_DIR=/run/user/1232
QT_PLUGIN_PATH=/usr/lib64/kde4/plugins:/usr/lib/kde4/plugins
CCACHE_HASHDIR=
_=/usr/bin/env
OLDPWD=/home/sp5/12180626
```

env 는 환경변수를 출력해주는 명령어이다.

11) exit

```
-bash-4.2$ exit
```

exit 명령어는 콘솔창을 종료시키는 명령어이다. exit 입력 직후 콘솔 창이 종료되었다.

12) file

```
-bash-4.2$ file
Usage: file [-bchikLlNnprsvz0] [--apple] [--mime-encoding] [--mime-type]
          [-e testname] [-F separator] [-f namefile] [-m magicfiles] file ...
       file -C [-m magicfiles]
       file [--help]
```

```
-bash-4.2$ file d1
d1: directory
```

file 명령어는 파일의 종류와 속성 값을 확인할 수 있다.

file d1 을 통해 d1 이 directory 임을 확인하였다.

13) find

```
-bash-4.2$ find
.
./d2
./d2/d2
./d2/f4
./f3.gz
./f4
./.bash_history
./f1
./d1
find: `./d1': Permission denied
```

find 명령어는 디렉토리에 저장되어 있는 파일들과 디렉토리를 검색할 수 있는 명령어이다.

14) head

```
-bash-4.2$ head f4
hello
```

head 명령어는 파일의 텍스트를 출력한다. head f4 를 통해 f4 에 저장된 파일 내용인 hello 문자열이 출력된다.

15) ifconfig

```
-bash-4.2$ ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 0 (Local Loopback)
    RX packets 7086911 bytes 1279735659 (1.1 GiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 7086911 bytes 1279735659 (1.1 GiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

p1p1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 165.246.38.151 netmask 255.255.255.0 broadcast 165.246.38.255
    inet6 fe80::d227:88ff:fec5:f3b7 prefixlen 64 scopeid 0x20<link>
    ether d0:27:88:c5:f3:b7 txqueuelen 1000 (Ethernet)
    RX packets 3026920 bytes 315138359 (300.5 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 840763 bytes 543323838 (518.1 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

virbr0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    inet 192.168.122.1 netmask 255.255.255.0 broadcast 192.168.122.255
    ether 52:54:00:45:d3:09 txqueuelen 0 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 6 bytes 1181 (1.1 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

ifconfig 명령어는 네트워크 인터페이스를 설정하거나 관련 정보를 조회할 수 있는 명령어이다.

16) ln

```
-bash-4.2$ ls
d1 d2 f1 f3.gz f4 f5
-bash-4.2$ ln f5 f6
-bash-4.2$ ls
d1 d2 f1 f3.gz f4 f5 f6
```

ln 명령어는 파일의 시스템에서 link file 을 만드는 명령어이다.

ls 를 통해 로그인 디렉토리에 있는 파일들을 파악한 뒤 ln f5 f6 을 통해 변화된 점을 다시 한 번 ls 을 통해 확인하면 f6 이라는 link file 이 생성됨을 확인하였다.

17) mount

```
-bash-4.2$ mount
proc on /proc type proc (rw,nosuid,nodev,noexec,relatime)
sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,relatime)
devtmpfs on /dev type devtmpfs (rw,nosuid,size=3926452k,nr_inodes=981613,mode=755)
securityfs on /sys/kernel/security type securityfs (rw,nosuid,nodev,noexec,relatime)
tmpfs on /dev/shm type tmpfs (rw,nosuid,nodev)
devpts on /dev/pts type devpts (rw,nosuid,noexec,relatime,gid=5,mode=620,ptmxmode=000)
tmpfs on /run type tmpfs (rw,nosuid,nodev,mode=755)
tmpfs on /sys/fs/cgroup type tmpfs (rw,nosuid,nodev,noexec,mode=755)
cgroup on /sys/fs/cgroup/systemd type cgroup (rw,nosuid,nodev,noexec,relatime,release_agent=/usr/lib/systemd/systemd-cgroups-agent,name=systemd)
pstore on /sys/fs/pstore type pstore (rw,nosuid,nodev,noexec,relatime)
cgroup on /sys/fs/cgroup/cpuset type cgroup (rw,nosuid,nodev,noexec,relatime,cpuset)
cgroup on /sys/fs/cgroup/cpu,cpuacct type cgroup (rw,nosuid,nodev,noexec,relatime,cpuacct,cpu)
cgroup on /sys/fs/cgroup/memory type cgroup (rw,nosuid,nodev,noexec,relatime,memory)
cgroup on /sys/fs/cgroup/devices type cgroup (rw,nosuid,nodev,noexec,relatime,devices)
cgroup on /sys/fs/cgroup/freezer type cgroup (rw,nosuid,nodev,noexec,relatime,freezer)
cgroup on /sys/fs/cgroup/net_cls type cgroup (rw,nosuid,nodev,noexec,relatime,net_cls)
cgroup on /sys/fs/cgroup/blkio type cgroup (rw,nosuid,nodev,noexec,relatime,blkio)
cgroup on /sys/fs/cgroup/perf_event type cgroup (rw,nosuid,nodev,noexec,relatime,perf_event)
/dev/mapper/fedora_linuxer1-root on / type ext4 (rw,relatime,data=ordered)
systemd-1 on /proc/sys/fs/binfmt_misc type autofs (rw,relatime,fd=31,pgrp=1,timeout=300,minproto=5,maxproto=5,direct)
hugetlbfs on /dev/hugepages type hugetlbfs (rw,relatime)
debugfs on /sys/kernel/debug type debugfs (rw,relatime)
tmpfs on /tmp type tmpfs (rw)
mqueue on /dev/mqueue type mqueue (rw,relatime)
configfs on /sys/kernel/config type configfs (rw,relatime)
/dev/sda3 on /boot type ext4 (rw,relatime,stripe=4,data=ordered)
/dev/mapper/fedora_linuxer1-home on /home type ext4 (rw,relatime,data=ordered)
binfmt_misc on /proc/sys/fs/binfmt_misc type binfmt_misc (rw,relatime)
```

mount 명령어는 하드웨어 장치와 연결하기 위해 특정위치에 연결해주는 명령어이다. mount 명령어 실행을 통해 현재 mount 된 모든 파일 시스템을 확인하였다.

18) netstat

```
-bash-4.2$ netstat
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 165.246.38.151:ssh      165.246.127.125:57299  ESTABLISHED
tcp        0      0 165.246.38.151:ssh      115.21.205.125:65406   ESTABLISHED
tcp        0      0 165.246.38.151:ssh      43.155.114.53:58158    ESTABLISHED
tcp        0      0 165.246.38.151:ssh      39.120.105.85:57889    ESTABLISHED
tcp        0      0 165.246.38.151:ssh      222.100.33.45:50511    ESTABLISHED
tcp        0      0 165.246.38.151:ssh      210.113.112.93:56963   ESTABLISHED
tcp        0      0 165.246.38.151:ssh      165.246.128.28:52431   ESTABLISHED
tcp        0      64 165.246.38.151:ssh      165.246.181.27:10043   ESTABLISHED
tcp6       0      0 localhost:7337          localhost:42097         ESTABLISHED
tcp6       0      0 localhost:7337          localhost:42100         ESTABLISHED
tcp6       0      0 localhost:42098         localhost:7337          ESTABLISHED
tcp6       0      0 localhost:7337          localhost:42099         ESTABLISHED
tcp6       0      0 localhost:42100         localhost:7337          ESTABLISHED
tcp6       0      0 localhost:42006         localhost:7337          ESTABLISHED
tcp6       0      0 localhost:42097         localhost:7337          ESTABLISHED
tcp6       0      0 localhost:7337          localhost:42006         ESTABLISHED
tcp6       0      0 localhost:42102         localhost:7337          ESTABLISHED
tcp6       0      0 localhost:7337          localhost:42102         ESTABLISHED
tcp6       0      0 localhost:42099         localhost:7337          ESTABLISHED
tcp6       0      0 localhost:7337          localhost:42098         ESTABLISHED
udp6       0      0 localhost:44205         localhost:44205         ESTABLISHED
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags       Type       State      I-Node  Path
unix   5        [ ]         DGRAM                    9488    /run/systemd/journal/
socket
unix  31        [ ]         DGRAM                    9490    /dev/log
unix   2        [ ]         DGRAM                   10350    @/org/freedesktop/sys
temd1/notify
unix   2        [ ]         DGRAM                   10448    /run/systemd/journal/
syslog
unix   2        [ ]         DGRAM                   10450    /run/systemd/shutdown
d
unix   3        [ ]         STREAM      CONNECTED    18242
unix   3        [ ]         STREAM      CONNECTED    17552    /var/run/dbus/system_
bus_socket
unix   3        [ ]         STREAM      CONNECTED    16824    /run/systemd/journal/
stdout
unix   3        [ ]         STREAM      CONNECTED    20491    @/tmp/.X11-unix/X0
```

(출력 결과가 너무 길어 일부 생략하여 첨부하였습니다.)

netstat 는 시스템 네트워크 연결의 목록을 보여주는 명령어이다.

```
-bash-4.2$ netstat -i
Kernel Interface table
Iface      MTU     RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
lo         65536   7098364 0      0 0      7098364 0      0      0 0 LRU
plp1       1500    3033573 0      0 0      842940 0      0      0 0 BMRU
virbr0     1500    0      0      0 0      6      0      0      0 0 BMU
```

-i 옵션은 인터페이스별 통계 값을 출력하는 명령어이다. netstat-i 를 통해 시스템 네트워크 연결의 목록을 인터페이스별 통계 값으로 출력할 수 있다.

19) stat

```
-bash-4.2$ pwd
/home/sp5
-bash-4.2$ stat 12180626
  File: '12180626'
  Size: 4096          Blocks: 8          IO Block: 4096   directory
Device: fd02h/64770d Inode: 2883590      Links: 4
Access: (0700/drwx-----)  Uid: ( 1232/12180626)   Gid: ( 1232/12180626)
Access: 2022-06-20 18:10:41.386781537 +0900
Modify: 2022-06-20 18:10:39.100794931 +0900
Change: 2022-06-20 18:10:39.100794931 +0900
 Birth: -
```

stat 는 파일과 파일 시스템의 상태를 조회하는 명령어이다. cd 를 통해 sp5 파일로 이동한 뒤, stat 12180626 을 통해 파일 시스템의 상태를 조회한다.

20) tail

```
-bash-4.2$ cd
-bash-4.2$ tail f4
hello
```

tail 명령어는 파일 텍스트를 출력하는 명령어이다. 파일 텍스트를 출력한다는 것은 head 와 동일하지만 tail 명령어는 뒤에서부터 출력한다. cd 를 통해 로그인 디렉토리로 이동한 뒤 해당 디렉토리에 포함된 f4 라는 파일을 tail 을 통해 호출하면 hello 라는 텍스트가 출력되는 것을 확인할 수 있다.

21) time

```
-bash-4.2$ time

real    0m0.000s
user    0m0.000s
sys     0m0.000s
```

time 명령어는 작업의 수행시간을 알려주는 명령어이다.

real : 명령어 실행 후 완료되기 까지 실제 소요시간

user : 실행 시간동안 user space 에서 실제 사용된 CPU 개수

sys : 실행 시간동안 kernel space 에서 실제 사용된 CPU 개수

```
-bash-4.2$ time cd ..

real    0m0.000s
user    0m0.000s
sys     0m0.000s
-bash-4.2$ time ls
12161699 12170629 12171769 12181350 12181855 12192119 12211813
12161749 12171470 12180566 12181772 12190654 12200687 12211827
12170550 12171747 12180626 12181821 12191816 12211767 12213532

real    0m0.002s
user    0m0.000s
sys     0m0.001s
```

상위 디렉토리로 이동하는 cd ..는 0 초가 걸림을 알 수 있고

ls 명령어 작업을 수행하는데 소요된 시간은 위의 결과값을 보면 알 수 있다.

22) touch

```
-bash-4.2$ cd 12180626
-bash-4.2$ touch f7
-bash-4.2$ ls
d1 d2 f1 f3.gz f4 f5 f6 f7
-bash-4.2$ cat f7
```

touch 명령어는 비어있는 파일을 만들거나 타임스탬프를 변경할 수 있다.

12180626 디렉토리로 이동한 뒤 touch f7 을 통해 비어있는 파일 하나를 생성하고 ls 를 통해 f7 파일까지 생성되어있음을 확인하였다. cat f7 을 통해 출력값이 없는 것을 보아 비어있는 값을 확인하였다.

23)tty

```
-bash-4.2$ tty
/dev/pts/3
```

tty 명령어는 터미널의 이름을 출력하는 명령어이다. 실행결과 터미널 이름이 /dev/pts/3 임을 확인하였다.

24) gunzip

```
-bash-4.2$ ls
d1 d2 f1 f3.gz f4 f5 f6 f7
-bash-4.2$ gunzip f3
-bash-4.2$ ls -l
total 28
d----- 2 12180626 12180626 4096 Jun 20 10:50 d1
drwxrwxr-x 2 12180626 12180626 4096 Jun 20 15:03 d2
-rw-rw-r-- 1 12180626 12180626 6 Jun 20 10:38 f1
-rw-rw-r-- 1 12180626 12180626 6 Jun 20 11:18 f3
-rw-rw-r-- 1 12180626 12180626 6 Jun 20 11:17 f4
-rw-rw-r-- 2 12180626 12180626 6 Jun 20 18:10 f5
-rw-rw-r-- 2 12180626 12180626 6 Jun 20 18:10 f6
-rw-rw-r-- 1 12180626 12180626 0 Jun 20 18:32 f7
-bash-4.2$ ls
d1 d2 f1 f3 f4 f5 f6 f7
```

gunzip 명령어는 gzip 명령어로 압축시킨 파일을 압축 해제하는 명령어이다.

ls 를 통해 압축된 파일이 f3 임을 확인하고,

gunzip f3 을 통해 f3 파일을 압축 해제시킨다.

이후 ls 를 통해 f3 파일이 압축 해제됨을 확인하였다.

25) whereis

```
-bash-4.2$ whereis
Usage:
  whereis [options] file

Options:
  -f <file>  define search scope
  -b         search only binaries
  -B <dirs>  define binaries lookup path
  -m         search only manual paths
  -M <dirs>  define man lookup path
  -s         search only sources path
  -S <dirs>  define sources lookup path
  -u         search from unusual entities
  -V         output version information and exit
  -h         display this help and exit

See how to use file and dirs arguments from whereis(1) manual.
```



```
-bash-4.2$ whereis ls
ls: /bin/ls /usr/bin/ls /usr/share/man/man1p/ls.1p.gz /usr/share/man/man1/ls.1.gz
```

whereis 명령어는 명령어의 실행 파일위치와 소스위치, manual 페이지 파일의 위치를 찾아주는 명령어이다. whereis ls 를 통해 ls 명령어의 실행 파일 위치와 소스위치, manual 페이지 파일의 위치를 확인하였다.

26) which

```
-bash-4.2$ which
Usage: /usr/bin/which [options] [--] COMMAND [...]
Write the full path of COMMAND(s) to standard output.

--version, -[vV] Print version and exit successfully.
--help,          Print this help and exit successfully.
--skip-dot       Skip directories in PATH that start with a dot.
--skip-tilde     Skip directories in PATH that start with a tilde.
--show-dot       Don't expand a dot to current directory in output.
--show-tilde     Output a tilde for HOME directory for non-root.
--tty-only       Stop processing options on the right if not on tty.
--all, -a        Print all matches in PATH, not just the first
--read-alias, -i Read list of aliases from stdin.
--skip-alias     Ignore option --read-alias; don't read stdin.
--read-functions Read shell functions from stdin.
--skip-functions Ignore option --read-functions; don't read stdin.

Recommended use is to write the output of (alias; declare -f) to standard
input, so that which can show aliases and shell functions. See which(1) for
examples.

If the options --read-alias and/or --read-functions are specified then the
output can be a full alias or function definition, optionally followed by
the full path of each command used inside of those.

Report bugs to <which-bugs@gnu.org>.
```

```
-bash-4.2$ which ls
alias ls='ls --color=auto'
/usr/bin/ls
```

which 명령어는 특정한 명령어의 위치를 찾는 명령어이다. which ls 를 통해 ls 명령어의 위치를 확인하였다.

27) whoami

```
-bash-4.2$ whoami
12180626
```

whoami 명령어는 사용자 명을 출력하는 명령어이다. 사용자 명이 12180626 임을 확인하였다.