

Linux Programming

2장. 리눅스 사용

sisong@ut.ac.kr

한국교통대학교 컴퓨터공학전공

송석일



고정 IP 설정

- 가상머신 기본 Network 설정은 DHCP
 - 재 booting 시 ip 변경 가능
- 고정 IP 설정 방법
 - /etc/netplan/00-installer-config.yaml 파일을 다음과 같이 수정

```
network:
  ethernets:
    enp0s3:
      addresses:
        - 192.168.0.10/24
      nameservers:
        addresses: [8.8.8.8, 8.8.4.4]
      routes:
        - to: default
          via: 192.168.0.1
  version: 2
```

❖ 빨간 색은 자신의
가상머신에 맞도록 수정

고정 IP 설정

- yaml 파일이 존재하지 않을 경우
 - `sudo netplan generate`
- 파일을 수정한 후
 - `sudo netplan apply`

2.1 기본 명령어

리눅스 기본 명령어

```
$ date
```

```
2023. 01. 01. (토) 12:26:10 KST
```

```
$ hostname
```

```
ubuntu-ds02
```

```
$ uname
```

```
Linux
```

```
$ uname -a
```

```
Linux Ubuntu 5.11.0-31-generic #33-Ubuntu SMP Wed Aug 11 13:19:04  
UTC 2021 x86_64 x86_64 x86_64 GNU/Linux
```

```
$ whoami
```

```
$ who
```

```
chang          :0          2022-01-01 12:29 (:0)
```

리눅스 기본 명령어

\$ ls

공개 다운로드 문서 바탕화면 비디오 사진 음악 템플릿

\$ passwd

xxxx에 대한 암호 변경 중

현재 비밀번호:

새 암호:

새 암호 재입력:

passwd: 암호를 성공적으로 업데이트했습니다

\$ clear

리눅스 기본 명령어

\$ man date ### 리눅스 명령어에 대한 매뉴얼 제공

 ### date 명령어에 대한 매뉴얼

User Commands DATE(1)

NAME

 date - print or set the system date and time

SYNOPSIS

 date [OPTION]... [+FORMAT]

 date [-u|--utc|--universal] [MMDDhhmm[[CC]YY][.ss]]

DESCRIPTION

 Display the current time in the given FORMAT, or set the system
short date. Mandatory arguments to long options are mandatory for
options too.

...

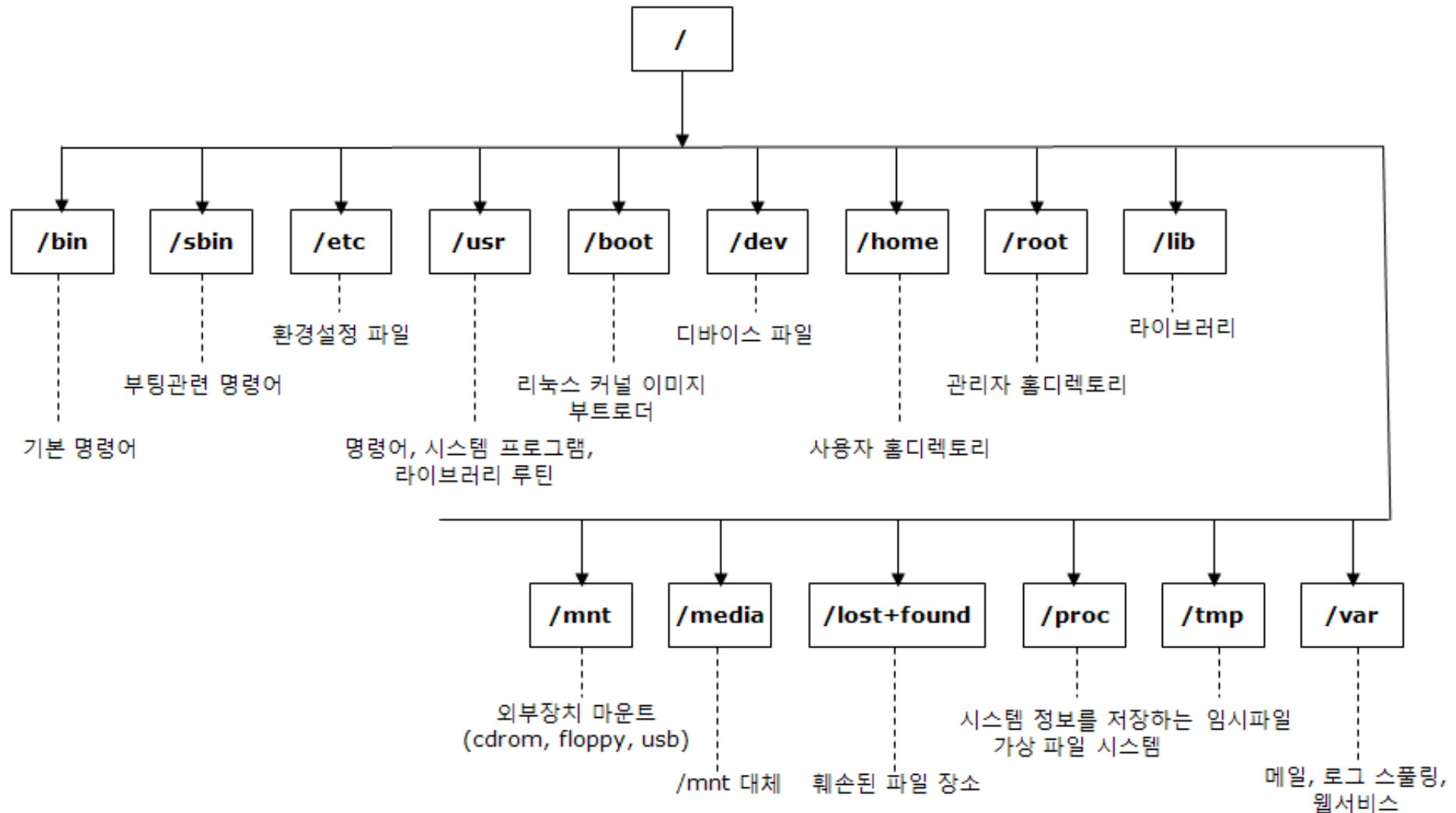
2.2 파일과 디렉토리

리눅스의 파일 종류

- 일반 파일(ordinary file)
 - 일반적인 데이터를 저장하는 파일
- 디렉터리(폴더)
 - 파일들을 계층적으로 조직화하는 데 사용되는 특수 파일
 - 디렉터리 파일은 디렉터리의 하위 파일, 디렉토리 이름 등 정보 저장
- 특수 파일(special file)
 - 물리적인 장치들을 파일로 표현
 - 키보드(stdin), 모니터(stdout), 프린터 등은 리눅스에서 파일
 - 리눅스 운영체제 안에서 사용되는 프로세스간 통신 수단 등을 파일로 표현
- 심볼릭 링크 파일
 - 어떤 파일을 가리키는 또 하나의 경로명을 저장하는 파일

디렉터리 계층구조

■ 리눅스 디렉터리



홈 디렉터리/현재 작업 디렉터리

- 홈 디렉터리(home directory)
 - 각 사용자마다 별도의 홈 디렉터리가 있음
 - 사용자가 로그인하면 홈 디렉터리에서 시작
- 현재 작업 디렉터리(current working directory)
 - 현재 작업 중인 디렉터리
 - 로그인 하면 홈 디렉터리가 현재 작업 디렉터리
- pwd(print working directory)
 - 현재 작업 디렉터리를 프린트

```
$ pwd
```

디렉터리 관련 명령

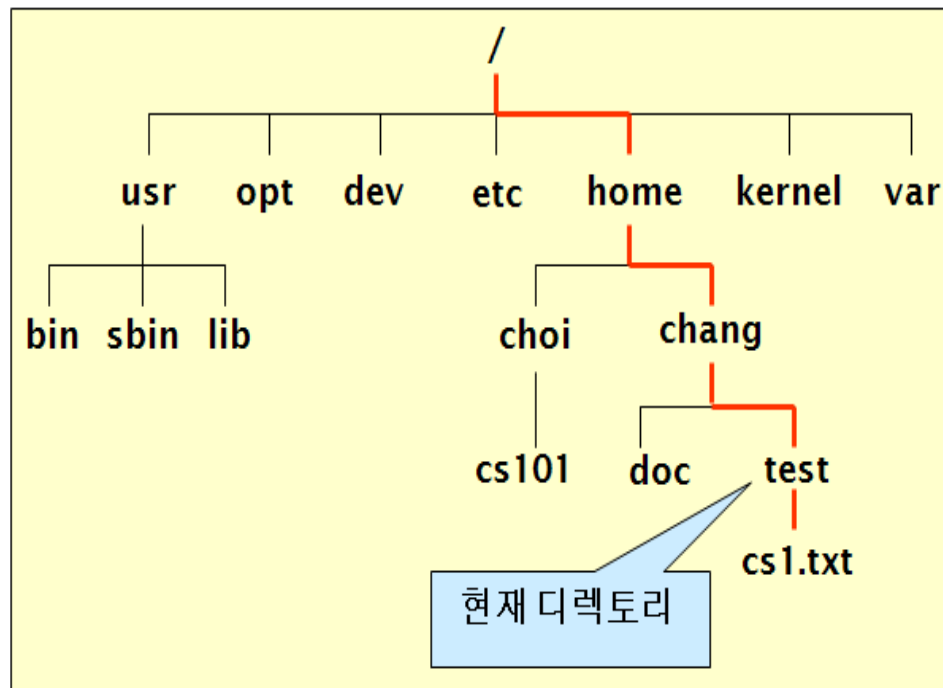
- mkdir(make directory)
 - 새 디렉터리 생성
- cd(change directory)
 - 현재 작업 디렉터를 이동

```
$ mkdir 디렉터리
$ mkdir test

$ cd [디렉터리]
$ cd test
$ pwd
/home/chang/test
$ cd
$ pwd
/home/chang
```

경로명

- 파일이나 디렉터리에 대한 정확한 이름
- 절대 경로명(absolute pathname)
 - 루트 디렉터리로부터 시작하여 경로 이름을 정확하게 적는 것
- 상대 경로명(relative path name)
 - 현재 작업 디렉터리부터 시작해서 경로 이름을 적는 것



~ : 홈 디렉터리
.: 현재 디렉터리
.. : 부모 디렉터리

cs1.txt의 절대 경로명
/home/chang/test/cs1.txt

cs1.txt의 상대 경로명
cs1.txt

디렉터리 리스트

- ls(list)
 - 디렉터리의 내용을 리스트

```
$ ls
cs1.txt
$ ls -s                                ### -s(size)
합계 4
4 cs1.txt
$ ls -a                                ### -a(all)
. .. cs1.txt

$ ls -l                                ### -l(long)
-rw-rw-r- 1 user1 user1 2088 4월 16일 13:37 cs1.txt
$ ls -asl
합계 12
4 drwxr-xr-x 2 user1 user1 4096 4월 16 13:37 .
4 drwx----- 3 user1 user1 4096 4월 16 13:37 ..
4 -rw-rw-r-- 1 user1 user1 2088 4월 16 13:37 cs1.txt

$ ls -F /                             ### *: 실행파일, /: 디렉터리, @:심볼릭 링크
bin@ dev/ home/ lib64@ mnt/ proc/ run/ srv/ tmp/ var/
boot/ etc/ lib@ media/ opt/ root/ sbin@ sys/ usr/

ls -R                                  ### -R(Recursive) 옵션은 모든 하위 디렉터리 내용 출력
```

디렉터리 관련 명령어

명령어	의미
ls	파일 및 디렉터리 리스트
ls -a	모든 파일과 디렉터리 리스트
ls -asl	모든 파일 자세히 리스트
mkdir	디렉터리 만들기
cd 디렉터리	디렉터리로 이동
cd	홈 디렉터리로 이동
cd ~	홈 디렉터리로 이동
cd ..	부모 디렉터리로 이동
pwd	현재 작업 디렉터리 프린트

2.3 파일 관련 명령어

파일 내용 리스트

■ 파일 내용 출력과 관련된 명령어

- cat, more, head, tail, wc, ...

```
$ 명령어 파일          ### 하나의 파일에 대해서 명령어 적용
$ 명령어 파일*         ### 여러 개(0개 이상) 파일에 대해서 명령어 적용
$ cat cs1.txt           ### 파일 내용을 화면에 출력
$ cat                   ### 키보드 입력 내용을 화면에 출력
...
^D

$ cat > cs1.txt         ### 키보드 입력 내용을 파일에 저장
...
^D

$ more 파일+           ### 파일을 화면에 출력
                        ### 여러 개(1개 이상) 파일에 대해서 명령어 적용
                        ### 각 파일의 내용을 페이지 단위로 화면에 출력

$ more cs1.txt
Unix is a multitasking, multi-user computer operating system
...
However, the term Unix is often used informally to denote any to
large-scale adoption of Unix(particularly of the BSD variant,
--계속--(59%)
```

head/tail/wc

- head 명령어
 - 파일의 앞부분(10줄)을 출력
- tail 명령어
 - 파일의 뒷부분(10줄)을 출력
- wc(word count)
 - 파일에 저장된 줄, 단어, 문자의 개수를 세서 출력

```
$ wc cs1.txt
38      318      2088 cs1.txt

$ wc -l cs1.txt
38 cs1.txt
```

cp (copy) 명령어

```
$ cp [-i] 파일1 파일2      ### 파일1을 파일2에 복사

$ cp cs1.txt cs2.txt
$ ls -l cs1.txt cs2.txt
-rw-rw-r-- 1 user1 user1 2088 4월 16 13:37 cs1.txt
-rw-rw-r-- 1 user1 user1 2088 4월 16 13:45 cs2.txt

$ cp 파일 디렉터리          ### 파일1의 복사본을 디렉터리 내에 만듦

$ cp cs1.txt /tmp

$ cp -i                      ### 대화형 옵션

$ cp -i cs1.txt cs2.txt
cp: overwrite 'cs2.txt'? n
```

mv (move) 명령어

```
$ mv [-i] 파일1 파일2          ### 파일1의 이름을 파일2로 변경
```

```
$ mv cs2.txt cs3.txt
```

```
$ ls -l
```

```
-rw-r--r-- 1 chang chang 2088 4월 16일 13:37 cs1.txt
```

```
-rw-r--r-- 1 chang chang 2088 4월 16일 13:56 cs3.txt
```

```
$ mv 파일 디렉터리          ### 파일을 지정된 디렉터리로 이동
```

```
$ mv cs3.txt /tmp
```

파일/디렉터리 삭제

- rm(remove) 명령어
 - 명령줄 인수로 받은 파일(들)을 삭제
- rmdir(remove directory) 명령어
 - 명령줄 인수로 받은 디렉터리(들)을 삭제

```
$ rm [-ri] 파일+
$ rm cs1.txt

$ rm -r 디렉터리          ### -r : 디렉터리 아래의 모든 것 삭제

$ rmdir 디렉터리+        ### 디렉터리가 비어있어야 함
$ rmdir test
```

링크

■ 링크

- 기존 파일에 대한 또 하나의 새로운 이름

■ 사용법

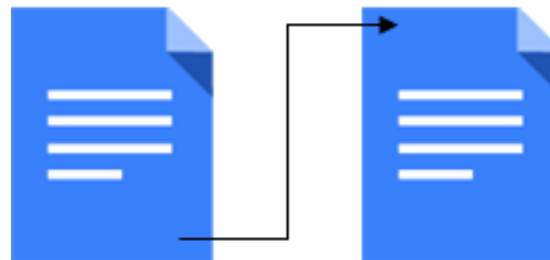
```
$ ln [-s] 파일1 파일2      ### 파일1에 대한 새로운 이름(링크)로 파일2 생성.  
                             ### -s 옵션은 심볼릭 링크  
$ ln [-s] 파일1 디렉터리   ### 파일1에 대한 링크를 지정된 디렉터리에  
                             ### 같은 이름으로 생성
```

파일1 파일2



파일2

파일1



하드 링크(hard link)

- 하드 링크
 - 기존 파일에 대한 새로운 이름
 - 기존 파일을 대표하는 i-노드를 공유

```
$ ln hello.txt hi.txt
$ ls -l
-rw----- 2 user1 user1 15 11월 7 15:31 hello.txt
-rw----- 2 user1 user1 15 11월 7 15:31 hi.txt
```

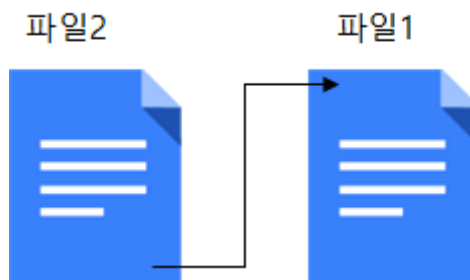
심볼릭 링크 (symbolic link)

■ 심볼릭 링크

- 다른 파일을 가리키고 있는 **별도의 파일**
- 실제 파일의 경로명을 저장하고 있는 일종의 특수 파일
- 이 경로명이 다른 파일에 대한 간접적인 포인터 역할

```
$ ln -s hello.txt hi.txt
$ ls -l
-rw----- 1 user1 user1 15 11월 7 15:31 hello.txt
lrwxrwxrwx 1 user1 user1 9 1월 24 12:56 hi.txt -> hello.txt

$ ln -s /usr/bin/gcc cc
$ ls -l cc
lrwxrwxrwx. 1 user1 user1 12 7월 21 20:09 cc -> /usr/bin/gcc
```



파일 관련 명령어

명령어	의미
cat 파일*	파일 내용 출력
more 파일+	페이지 단위로 파일 내용 출력
head 파일*	파일의 앞부분 출력
tail 파일*	파일의 뒷부분 출력
wc 파일*	줄/단어/문자 수 세기
cp 파일1 파일2	파일1을 파일2로 복사
mv 파일1 파일2	파일1을 파일2로 이름 변경
rm 파일+	파일 삭제
rmdir 디렉터리+	디렉터리 삭제
ln 파일1 파일2	링크 만들기

2.4 파일 속성

파일 속성(file attribute)

- 파일의 이름, 타입, 크기, 소유자, 접근권한, 수정 시간

```
$ ls -sl cs1.txt
```

```
4 -rw-rw-r-- 1 user1 user1 2088 4월 16 13:37 cs1.txt
```

파일 속성	의미
블록 수	파일을 구성하는 블록의 개수(KB 단위)
파일 종류	일반 파일(-), 디렉터리(d), 링크(l), 파이프(p), 소켓(s), 디바이스(b 혹은 c) 등의 파일 종류
접근권한	파일에 대한 소유자, 그룹, 기타 사용자의 읽기(r)/쓰기(w)/실행(x) 권한
하드 링크 수	파일에 대한 하드 링크 개수
소유자 및 그룹	파일의 소유자 ID 및 소유자가 속한 그룹
파일 크기	파일의 크기(바이트 단위)
최종 수정 시간	파일을 생성 혹은 최후로 수정한 시간

접근권한(permission mode)

- 파일 및 디렉터리에 대한 읽기(r), 쓰기(w), 실행(x) 권한

권한	파일	디렉터리
r	파일에 대한 읽기 권한	디렉터리 내에 있는 파일명을 읽을 수 있는 권한
w	파일에 대한 쓰기 권한	디렉터리 내에 파일을 생성하거나 삭제할 수 있는 권한
x	파일에 대한 실행 권한	디렉터리 내로 탐색을 위해 이동할 수 있는 권한

- 파일의 접근권한은 소유자(owner)/그룹(group)/기타(others)로 구분하여 관리

- 예

소유자	그룹	기타
rw-	rw-	r--

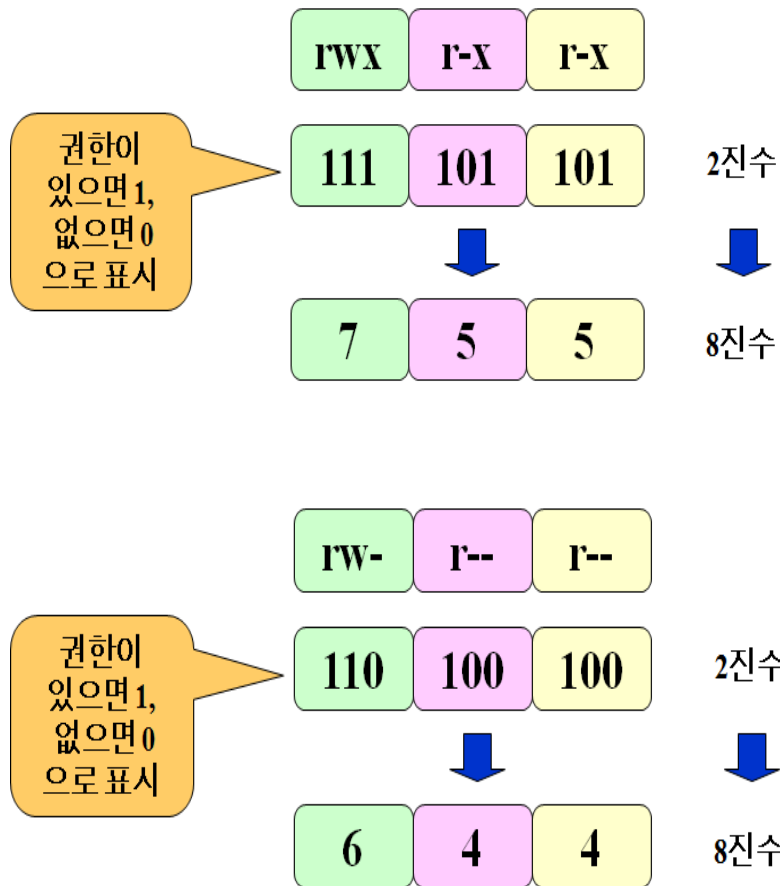
접근권한 변경: chmod(change mode)

- chmod
 - 파일 혹은 디렉터리의 접근권한을 변경

```
$ chmod [-R] 접근권한 파일 혹은 디렉터리.   ### -R 옵션을 사용하면 지정된  
                                           ### 디렉터리 아래의 모든 파일과  
                                           ### 하위 디렉터리에 대해서도  
                                           ### 접근권한을 변경
```

접근권한 표현: 8진수

■ 접근권한 8진수 변환



■ 사용 예

```
$ chmod 644 cs1.txt
$ ls -l cs1.txt
-rw-r--r-- 1 user1 ... cs1.txt
```

접근권한	8진수
rw-rw-rw-	777
rw-r-xr-x	755
rw-rw-r--	664
rw-r--r--	644
rw-r-----	640
rwX-----	700

접근권한 표현: 기호

■ 기호를 이용한 접근권한 변경

사용자범위 연산자 권한
[u|g|a]+ [+|-|=] [r|w|x]+

구분	기호와 의미
사용자 범위	u(user:소유자), g(group:그룹), o(others:기타 사용자), a(all:모든 사용자)
연산자	+(권한 추가), -(권한 제거), =(권한 설정)
권한	r(읽기 권한), w(쓰기 권한), x(실행 권한)

```
$ chmod g+w cs1.txt  
$ ls -l cs1.txt  
-rw-rw-r-- 1 user1 ... cs1.txt
```

소유자 및 그룹 변경: chown, chgrp

- chown 명령어
 - 파일이나 디렉터리의 소유자를 변경할 때 사용
- chgrp 명령어
 - 파일이나 디렉터리의 그룹을 변경할 때 사용
- 파일의 소유자 또한 슈퍼 유저만이 사용 가능

```
$ chown 사용자 파일  
$ chown [-R] 사용자 디렉터리  
  
$ chgrp 그룹 파일  
$ chgrp [-R] 그룹 디렉터리
```


2.5 입출력 재지정 및 파이프

출력 재지정(output redirection)

- 명령어의 표준출력 내용을 모니터 대신에 파일에 저장

\$ 명령어 > 파일

\$ who > names.txt

\$ ls -sl > list.txt

\$ cat > list1.txt

Hi !

This is the first list.

^D

\$ cat > list2.txt

Hello !

This is the second list.

^D

\$ cat list1.txt list2.txt > list3.txt

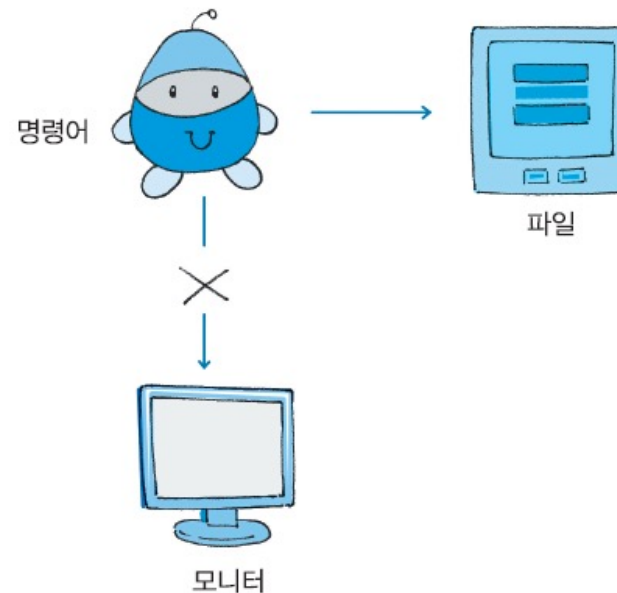
\$ cat list3.txt

Hi !

This is the first list.

Hello !

This is the second list.



출력 추가

- 명령어의 표준출력을 모니터 대신에 파일에 추가

```
$ 명령어 >> 파일

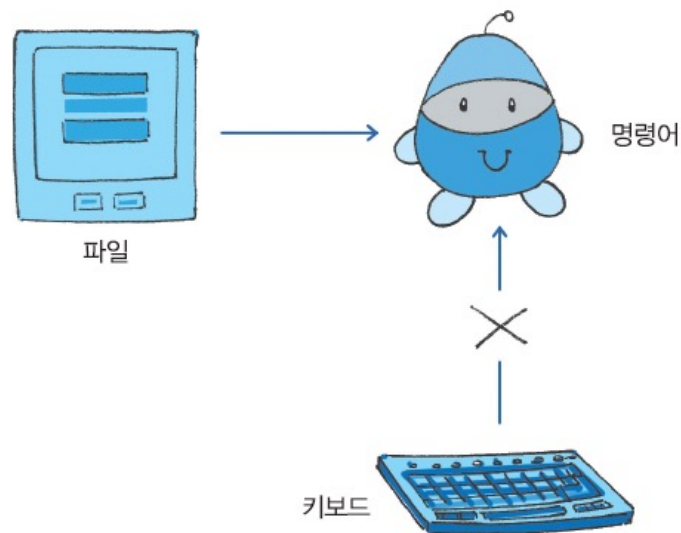
$ cat >> list1.txt
Bye !
This is the end of the first list.
^D

$ cat list1.txt
Hi !
This is the first list.
Bye !
This is the end of the first list.
```

입력 재지정(input redirection)

- 명령어의 표준입력을 키보드 대신에 파일에서 받음

```
$ 명령어 < 파일  
$ wc < list1.txt  
4  17  71 list1.txt
```



문서 내 입력(here document)

- 명령어의 표준 입력을 키보드 대신에 단어와 단어 사이의 입력 내용으로 받음
- 보통 스크립트 내에서 입력을 줄 때 사용

```
$ 명령어 << 단어
...
단어

$ wc << end
hello !
word count
end
2    4   20
```

파이프

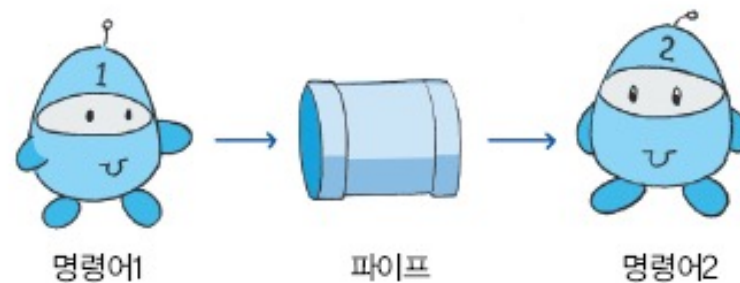
- | : 두개 이상의 명령어의 입출력을 연결

현재 디렉터리 내의 파일 이름들을 `sort -r` 명령어를 사용해서 내림차순으로
정렬해서 출력하기 위해서는 아래의 두 명령어를 연속해서 실행

```
$ ls > ls.txt  
$ sort -r < ls.txt
```

\$ 명령어1 | 명령어2 ### 명령어1의 표준출력을 명령어2의 표준입력으로

```
$ ls | sort -r  
$ ls | wc -l
```



입출력 재지정 및 파이프 요약

명령어	의미
명령어 > 파일	표준 출력을 파일로 재지정
명령어 >> 파일	표준 출력을 파일에 추가
명령어 < 파일	표준 입력을 파일로 재지정
명령어1 명령어2	명령어1의 표준출력이 파이프를 통해 명령어2의 표준입력이 됨
cat 파일1 파일2 > 파일3	파일1과 파일2를 연결하여 파일3 만듦

2.6 텍스트 편집기

gedit

- GNOME이 제공하는 GUI 기반 문서편집기
- 사용방법
 - [프로그램 표시] -> [텍스트 편집기]
 - \$ gedit [파일이름]
 - 파일 관리자에서 텍스트 파일을 클릭하면 자동으로 실행

