제2장 리눅스 사용

2.1 기본 명령어

간단한 명령어 사용

$ date : 날짜 출력

$ hostname : 호스트 이름 확인 (시스템 관리자가 부여한 이름)

$ uname : 운영체제 확인 (설치된 운영체제 이름)

$ who : 로그인 한 사용자 확인

$ ls : 현재 디렉토리 내부의 파일 목록 확인

$ clear : 화면을 정리하고 첫째줄에 프롬프트 표시

$ passwd : 암호 변경

$ man : 온라인 매뉴얼

기본 명령어 사용

- 날짜 및 시간 확인

```
$ date
2016년 12월 26일 월요일 오후 01시 52분 02초
```

- 시스템 정보 확인

```
$ hostname  doname
linux.yeungnam.ac.kr
$ uname  unixname : OS
Linux
$ uname -a  세부정보
Linux linux.yeungnam.ac.kr 3.10.0-123.el7.x86_64 #1 SMP Mon
Jun 30 12:09:22 UTC 2014 x86_64 x86_64 x86_64 GNU/Linux
```

기본 명령어 사용

- 사용자 정보 확인

```
$ whoami
user127

$ who
user127 pts/1 2017-07-12 11:05 (:10.0)
brain pts/5 2017-07-12 13:46 (203.153.155.35)
...
```

- 디렉터리 내용 확인

```
$ ls
Desktop Music Templates Documents Pictures Videos ...
```

기본 명령어 사용

- 패스워드 변경

```
$ passwd
passwd: user127용 암호를 변경하는 중
기존 로그인 암호를 입력하십시오:
새 암호:
새 암호를 다시 입력하십시오:
passwd: 암호(user127용)가 성공적으로 변경되었습니다.
```

- 화면 정리

```
$ clear
```

온라인 매뉴얼: man

```
$ 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 ..
Manual page ls(1) line 1 (press h for help or q to quit)
```

명령어에 대한 간단한 설명: whatis

```
$ whatis ls
ls (1) - 경로의 내용을 나열한다.
ls (1p) - list directory contents
```

명령어 내용 man보다 덜함

2.2 파일 및 디렉토리

파일의 종류

- 일반 파일(ordinary file)
  - 데이터를 가지고 있으면서 디스크에 저장된다.
- 디렉토리 파일 (directory file)/폴더(folder)
  - 디렉토리(폴더) 자체도 하나의 파일로 한 디렉토리는 다른 디렉토리들을 포함함으로써 계층 구조를 이룬다.
  - 부모 디렉토리는 다른 디렉토리들을 서브 디렉토리로 갖는다.
  - 일반 파일로서의 디렉토리 파일
    - 디렉토리 파일의 내용은 무엇일까?

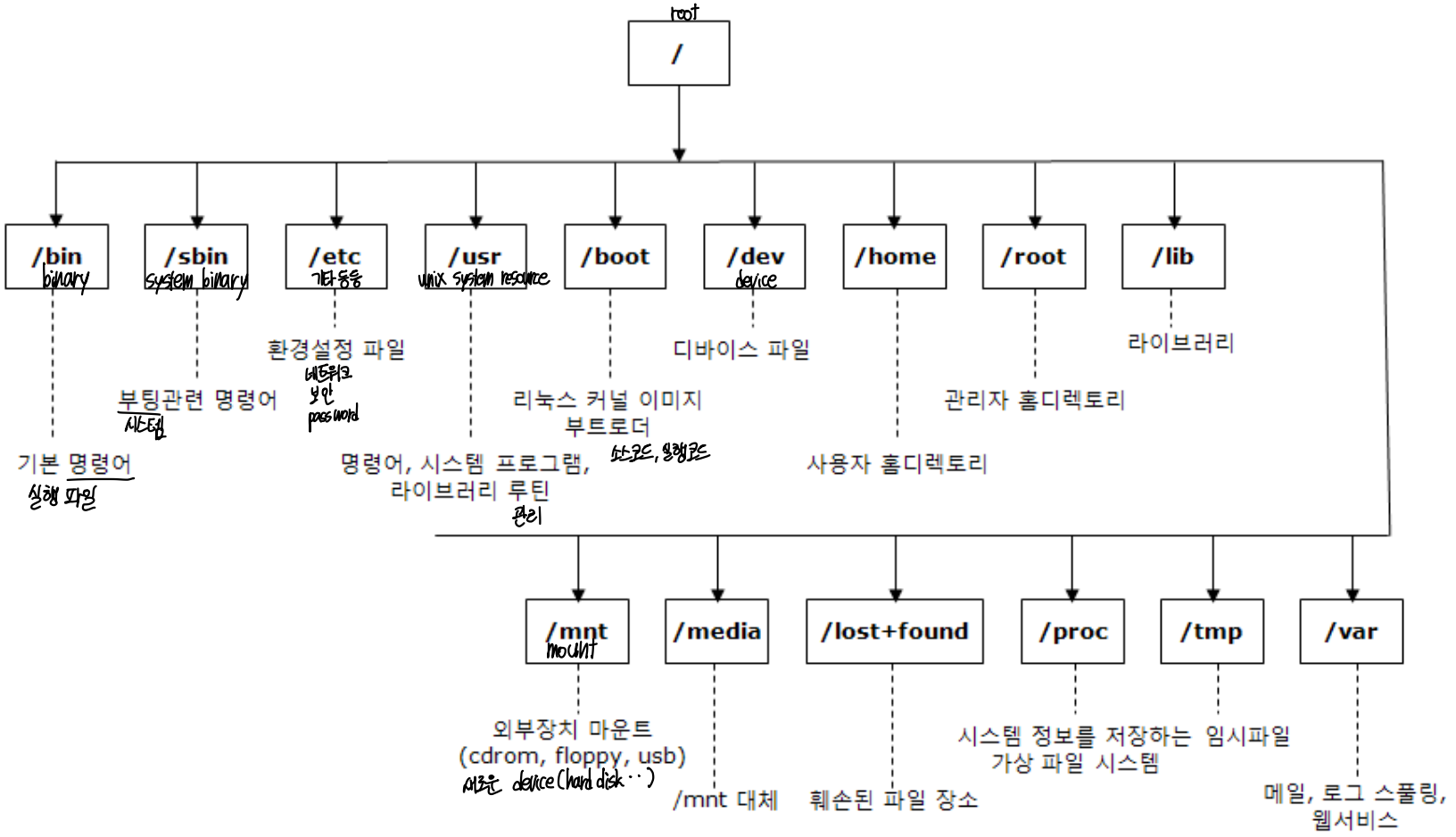
디렉토리 drwxrwxr-x 2
일반파일 -rw-rw-r-- 1

- 특수 파일(special file)
  - 물리적인 장치(device)에 대한 내부적인 표현
  - 키보드(stdin), 모니터(stdout), 프린터 등도 파일처럼 사용
    - 프린터 출력 → 프린터 파일에 쓰기 연산 수행

pwd

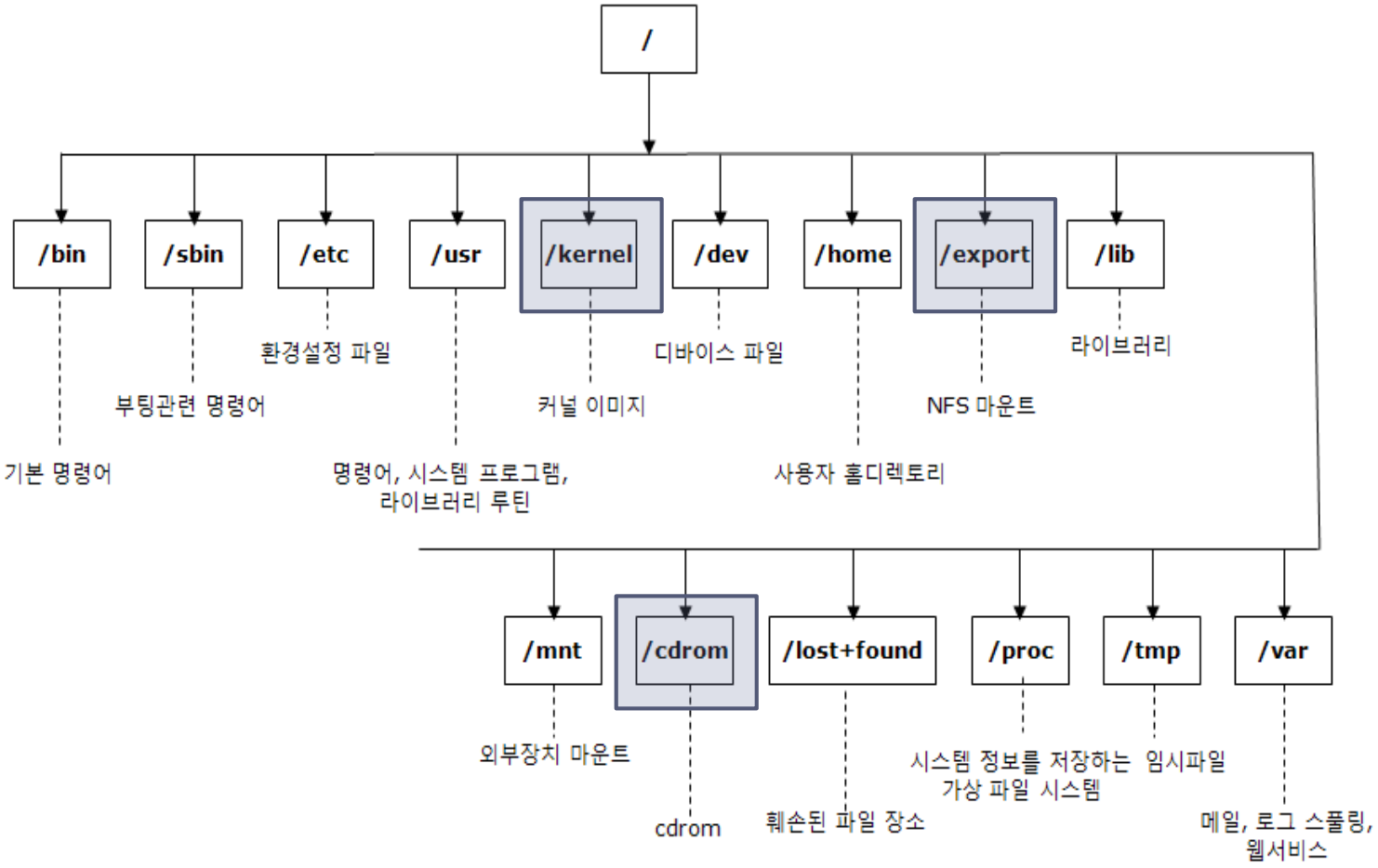
디렉토리 계층구조

- **리눅스** 디렉토리 : 루트로부터 시작, Tree 형태의 계층구조를 이룬다.



디렉토리 계층구조

- **유닉스** 디렉토리 : 리눅스와 비슷하나 대체로 다음과 같다.



홈 디렉토리/현재 작업 디렉토리

- 홈 디렉토리(home directory)
  - 각 사용자마다 별도의 홈 디렉토리가 있음
  - 사용자가 로그인하면 홈 디렉토리에서 작업을 시작

- 현재 작업 디렉토리(current working directory) 현재 내가 있는 directory
  - 현재 작업 중인 디렉토리
  - 로그인 하면 홈 디렉토리에서부터 작업이 시작

디렉토리 관련 명령

- pwd(print working directory)
  - 현재 작업 디렉토리를 프린트
  - $ pwd

- cd(change directory)
  - 현재 작업 디렉토리를 이동
  - $ cd [디렉토리]

생략 0

현재 작업 디렉터리 출력: pwd(print working directory)

- 사용법

```
$ pwd
현재 작업 디렉터리의 절대 경로명을 출력한다.
```

- 현재 작업 디렉터리(current working directory)
  - 현재 작업 중인 디렉터리
  - 로그인 하면 홈 디렉터리에서부터 작업이 시작된다.

- 예

```
$ pwd
/home/chang/Desktop
$ cd ~
$ pwd
/home/chang
```

디렉터리 이동: cd(change directory)

- 사용법

```
$ cd [디렉터리]
```

현재 작업 디렉터리를 지정된 디렉터리로 이동한다.

디렉터리를 지정하지 않으면 홈 디렉터리로 이동한다.

- 예

```
$ cd
$ cd ~
$ cd Desktop
$ pwd
/home/chang/Desktop
$ cd ..
```

명령어의 경로 확인: which

- 사용법

```
$ which 명령어
```

명령어의 절대경로를 보여준다.

- 예

```
$ which ls
/bin/ls
$ which pwd
/usr/pwd
$ which passwd
/usr/passwd
```

디렉토리 관련 명령

- mkdir(make directory)
  - 새 디렉토리를 생성
  - $ mkdir 디렉토리

디렉터리 생성: mkdir(make directory)

- 사용법

```
$ mkdir [-p] 디렉터리⊕ 하나 이상
```

디렉터리(들)을 새로 만든다.

-p : 중간 디렉터리 자동 생성 옵션 (다음 페이지 예제 참조)

- 예

```
$ cd ~     // 홈 디렉터리로 이동
$ mkdir test
$ mkdir test temp
$ ls -l
drwxrwxr-x. 2 chang chang 6 5월 12 10:12 temp
drwxrwxr-x. 2 chang chang 6 5월 12 10:12 test
```

디렉터리 생성: mkdir

- 중간 디렉터리 자동 생성 옵션 -p
  - 필요한 경우에 중간 디렉터리를 자동으로 만들어 준다.

- 예 : ~/dest 디렉터리가 없는 경우

```
$ mkdir ~/dest/dir1
mkdir: '/home/chang/dest/dir1' 디렉터리를 만들 수 없습니다: 그런 파일이나 디렉터리가 없습니다
$ mkdir -p ~/dest/dir1
```

경로지정까지

mkdir -p 4/b //현재 디렉토리에서

디렉터리 삭제 : rmdir(remove directory)

- 사용법

  $ rmdir 디렉터리+

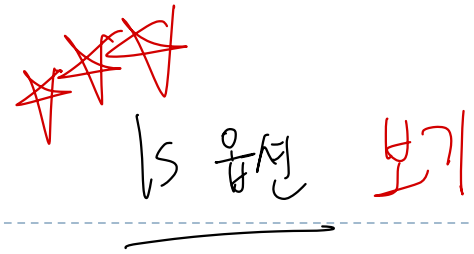
  디렉터리(들)을 삭제한다.

  - 주의: 빈 디렉토리만 삭제할 수 있다.

- 예

```
$ rmdir test
rmdir: failed to remove ‘test’: 디렉터리가 비어있지 않음
```

디렉토리 리스트



- ls(list)
  - 디렉토리의 내용을 리스트

- $ ls

  cs1.txt
- $ ls –s　　-s(size) : 크기 표기

  총 6

  6 cs1.txt
- $ ls –a　　-a(all) : 숨김 파일 표기

  . .. cs1.txt

디렉토리 리스트

- $ ls –l -l(long) : 자세한 정보 표기

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

- $ ls –asl : 옵션의 결합 가능

  총 10 파일 크기, 숨긴 파일, 자세한 정보 표기

  2 drwxr-xr-x 2 chang faculty 512 4월 16일 13:37 .

  2 drwxr-xr-x 3 chang faculty 512 4월 16일 13:37 ..

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

디렉토리 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

디렉터리 리스트: ls(list)

- 사용법

+: one or more

$ ls(혹은 dir) [-aslFR] (디렉터리* 파일*) *: zero or more

지정된 디렉터리의 내용을 리스트 한다. 디렉터리를 지정하지 않으면 현재 디렉터리 내용을 리스트 한다. 또한 파일을 지정하면 해당 파일만을 리스트 한다.

- 예

```
$ ls /
bin dev home lib64 mnt proc run srv tmp var
boot etc lib media opt root sbin sys usr
$ ls ~   자신의 홈
Desktop Downloads Pictures Templates pl 다운로드
Documents Music Public Videos linux tmp 사진
$ cd Desktop
$ ls
cs1.txt
```

ls 명령어 옵션

- 주요 옵션

옵션	기능
-a	숨겨진 파일을 포함하여 모든 파일을 리스트한다.
-s	파일의 크기를 K 바이트 단위로 출력한다.
-l	파일의 상세 정보를 출력한다.
-F	파일의 종류를 표시하여 출력한다. 파일 파일 디렉토리 파일명/
-R	모든 하위 디렉터리들을 리스트한다. A/B/C/D/E 하나의 명령어로 다양한 디렉토리를 볼수있음

recursive

ls 명령어 옵션

- **ls -s**
  - -s(size) 옵션
  - 디렉터리 내에 있는 모든 파일의 크기를 K 바이트 단위로 출력

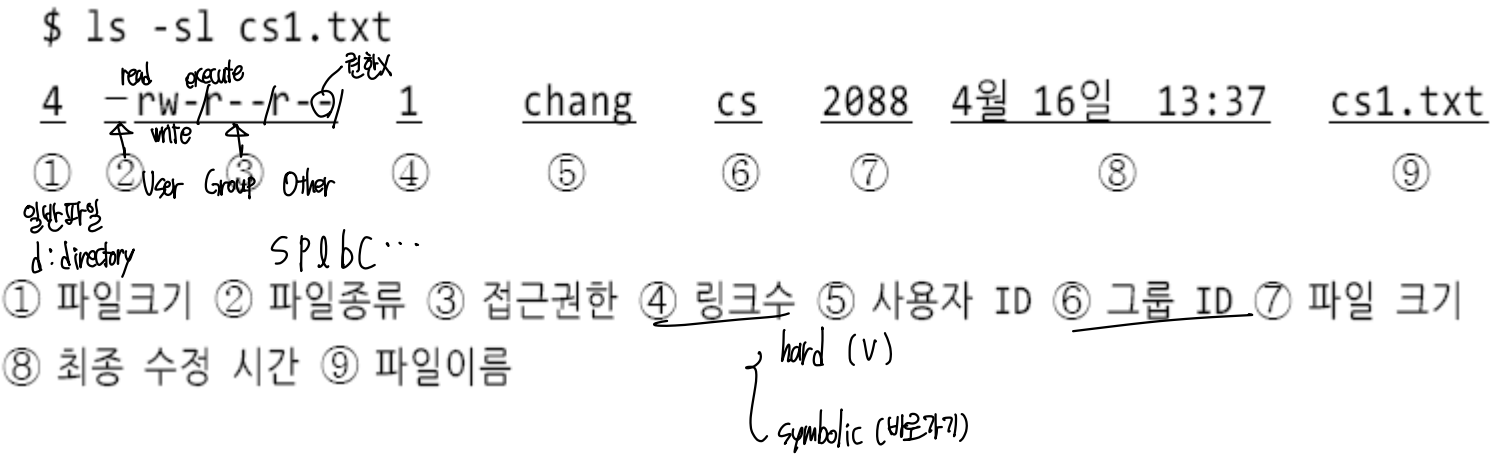
```
$ ls -s
총 4
4 cs1.txt
```

- **ls -a**
  - -a(all) 옵션
  - 숨겨진 파일들을 포함하여 모든 파일과 디렉터리를 리스트
  - "."은 현재 디렉터리, ".."은 부모 디렉터리

```
$ ls -a
. .. cs1.txt
```

ls 명령어 옵션

- **ls -l**
  - -l(long) 옵션
  - 파일 속성(file attribute) 출력
    - 파일 이름, 파일 종류, 접근권한, 소유자, 크기, 수정 시간 등



ls 명령어 옵션

- **ls –asl**

```
$ ls -asl
총 8
0 drwxr-xr-x 2 chang cs 20 4월 16일 13:37 .
4 drwx------ 3 chang cs 4096 4월 16일 13:37 ..
4 -rw-r--r-- 1 chang cs 2088 4월 16일 13:37 cs1.txt
```

ls 명령어 옵션

- **ls –F**
  - 기호로 파일의 종류를 표시

    *: 실행파일, /: 디렉터리, @:심볼릭 링크

- 예

```
$ ls -F /
bin@ dev/ home/ lib64@ mnt/ proc/ run/ srv/ tmp/ var/
boot/ etc/ lib@ media/ opt/ root/ sbin@ sys/ usr/
```

ls 명령어 옵션

- **ls –R**
  - -R(Recursive) 옵션
  - 모든 하위 디렉터리 내용을 리스트 한다.

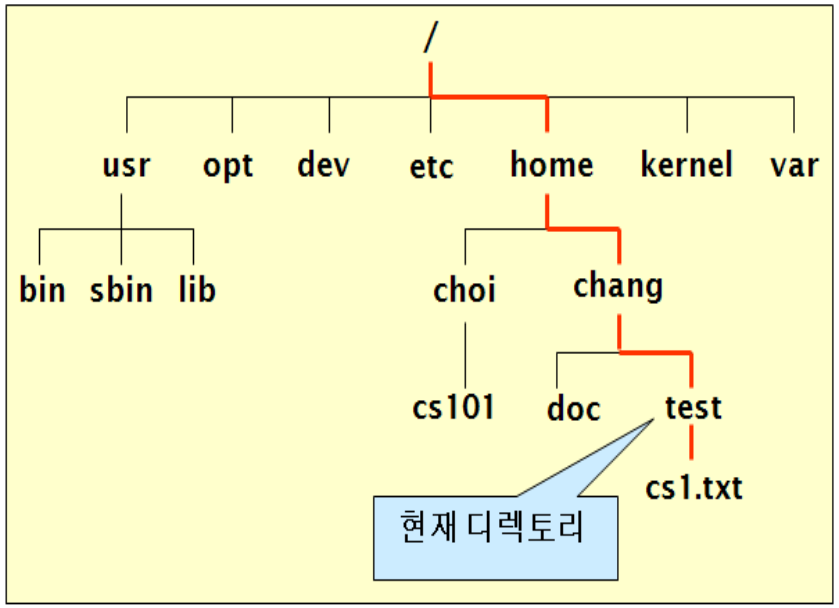
- 예

```
$ ls –R
$ ls –R /
```

경로명

- 파일이나 디렉토리에 대한 정확한 이름
- 절대 경로명(absolute pathname)
  - 루트 디렉토리로부터 시작하여 경로 이름을 정확하게 적는 것 cd../../ ··· /usr
- 상대 경로명(relative path name)
  - 현재 작업 디렉토리부터 시작해서 경로 이름을 적는 것 cd /usr

~ : 홈 디렉토리
. : 현재 디렉토리
.. : 부모 드렉터리



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

cs1.txt의 상대 경로명
cs1.txt

파일 내용 리스트

- 파일 내용 출력과 관련된 다음 명령어들
  - cat, more, head, tail, wc 등
  - 사용 사례

$ 명령어 파일명

$ 명령어 파일명* (*는 0개 혹은 그 이상 개를 의미함, 0개인 경우 키보드로부터 입력 받음)

$ more 파일명+ (+는 1개 혹은 그 이상 개를 의미함)

e.g., more test1 test2

cat 명령어

concatenate 결합

- 파일 내용 출력

$ cat cs1.txt

$ cat (키보드로부터 입력받아 그대로 출력)

…

^D (ctrl-D: 입력의 끝을 나타냄)

$ cat > cs1.txt (키보드로 입력받은 내용을 파일로 저장)

내용 입력 : 문서 출력

…

^D

cat의 다양한 기능 : 문서 보기, 문서결합, 문서편집

1. 파일 읽기 : cat a [nothing] a
2. 파일 결합 : cat a b ← a b
3. > 와 함께 파일 내용 이동 cat a b > c (결합)
4. 파일 라인 단위 편집 : cat > file_name (편집후 ctrl-c)

more/head/tail/wc

- more 명령어
  하나 혹은 그 이상의 파일 이름을 받을 수 있으며,
  각 파일의 내용을 페이지 단위로 출력
  다음 페이지 보기 [space-bar], 보기 종료 : q

(MD 결과를 한페이지씩 보기
%>(MD | more
pipe

- head 명령어
  파일의 앞부분(10줄)을 출력한다.
  옵션으로 줄 수 조정 가능 : head -5 cs1.txt (앞부분 5줄)

언제사용??

- tail 명령어
  파일의 뒷부분(10줄)을 출력한다.

who 로그인 한줄당 1명
wc : 로그인 몇명했는지 확인 가능

- wc(word count)
  파일에 저장된 줄(-l), 단어(-w), 문자(-c)의 개수를 세서 출력

man wc

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

줄 문자

파일 내용 보기: cat

- 사용법

$ cat [-n] 파일* 0개 이상

파일(들)의 내용을 그대로 화면에 출력한다. 파일을 지정하지 않으면 표준입력 내용을 그대로 화면에 출력한다.

- 예

```
$ cat cs1.txt
Unix is a multitasking, multi-user computer operating system originally
developed in 1969 by a group of AT&T employees at Bell Labs, including
Ken Thompson, Dennis Ritchie, Brian Kernighan, Douglas McIlroy,
and Joe Ossanna.
...
```

파일 내용 보기: cat

- 예

```
$ cat -n cs1.txt
1 Unix is a multitasking, multi-user computer operating system originally
2 developed in 1969 by a group of AT&T employees at Bell Labs, including
3 Ken Thompson, Dennis Ritchie, Brian Kernighan, Douglas McIlroy,
4 and Joe Ossanna.
...
```

```
$ cat          // 지정 파일 없음 → 입력을 그대로 출력
Hello World !
Hello World !
Bye!
Bye!
^D
```

페이지 단위로 파일 내용 보기: more

- 사용법

1개 이상

```
$ more 파일(+)
```

파일(들)의 내용을 페이지 단위로 화면에 출력한다.

- 예

```
$ more cs1.txt
Unix is a multitasking, multi-user computer operating system originally
developed in 1969 by a group of AT&T employees at Bell Labs, including
Ken Thompson, Dennis Ritchie, Brian Kernighan, Douglas McIlroy,
and Joe Ossanna.
...
During the late 1970s and early 1980s, the influence of Unix in academic
circles led to large-scale adoption of Unix(particularly of the BSD variant,
--계속--(59%)
```

파일 앞부분보기: head

- 사용법

0개 이상

```
$ head [-n] 파일*
```

기본 10줄

파일(들)의 앞부분을 화면에 출력한다. 파일을 지정하지 않으면 표준입력 내용을 대상으로 한다.

- 예

```
$ head -5 cs1.txt
Unix is a multitasking, multi-user computer operating system originally
developed in 1969 by a group of AT&T employees at Bell Labs, including
Ken Thompson, Dennis Ritchie, Brian Kernighan, Douglas McIlroy,
and Joe Ossanna.
```

head 파일* : 가령 head 만 입력하여도 키보드로부터 입력 받아 이를 그대로 모니터로 출력

파일 뒷부분보기: tail

- 사용법

종료시키면 모두 다 출력

```
$ tail [-n] 파일*
```

0개 이상

파일(들)의 뒷부분을 화면에 출력한다. 파일을 지정하지 않으면 표준입력 내용을 대상으로 한다.

- 예

```
$ tail cs1.txt          // 기본적으로 마지막 10줄 출력
Linux, which is used to power data centers, desktops, mobile phones,
and embedded devices such as routers, set-top boxes or e-book readers.
Today, in addition to certified Unix systems such as those already
mentioned, Unix-like operating systems such as MINIX, Linux, Android,
and BSD descendants (FreeBSD, NetBSD, OpenBSD, and DragonFly BSD) are
commonly encountered.
The term traditional Unix may be used to describe a Unix or
an operating system that has the characteristics of either Version 7
Unix or UNIX System V.
```

단어 세기: wc(word count)

- 사용법

```
$ wc [-lwc] 파일* 0개 이상
```

파일에 저장된 줄(l), 단어(w), 문자(c)의 개수를 세서 출력한다.

파일을 지정하지 않으면 표준입력 내용을 대상으로 한다.

- 예

```
$ wc cs1.txt
38 318 2088 cs1.txt
$ wc -l cs1.txt
38 cs1.txt
$ wc -w cs1.txt
318 cs1.txt
$ wc -c cs1.txt
2088 cs1.txt
```

cp 명령어

디렉터리 복사 x
복사하려면 cp -r (복사할 원본) (복사 될 이름)
빈 디렉토리라도 -r 사용
r(recursive)

- $ cp 파일1 파일2
  파일1의 복사본 파일2를 현재 디렉토리 내에 생성

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

- $ cp 파일(들) 디렉토리
  파일(들)의 복사본을 디렉토리 내에 생성
  $ cp cs1.txt /tmp

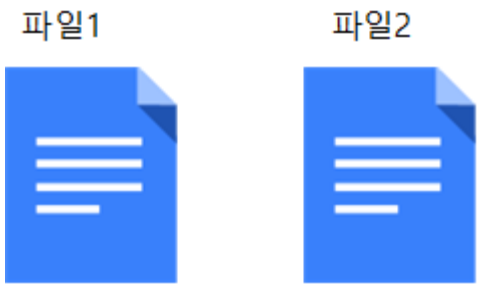
- $ cp –r 디렉토리1 디렉토리2
  디렉토리1을 디렉토리2로 복사
  디렉토리1의 하위 파일도 모두 복사 됨

파일 복사: cp(copy)

- 사용법

```
$ cp [-i] 파일1 파일2
```

파일1을 파일2에 복사한다. -i는 대화형 옵션이다.



- 예

```
$ cp cs1.txt cs2.txt
$ ls -l cs1.txt cs2.txt
-rw-r--r-- 1 chang cs 2088 4월 16일 13:37 cs1.txt
-rw-r--r-- 1 chang cs 2088 4월 16일 13:45 cs2.txt
$ cp /etc/hosts hostnames
```

파일 복사: cp(copy)

- 대화형 옵션: cp -i
  - 복사 대상 파일과 이름이 같은 파일이 이미 존재하면 덮어쓰기(overwrite) !
  - 보다 안전한 사용법: 대화형 –i(interactive) 옵션을 사용

- 예

```
$ cp –i cs1.txt cs2.txt
cp: overwrite ‘cs2.txt’? n
```

파일 복사: cp(copy)

- 파일을 디렉터리로 복사

```
$ cp 파일 디렉터리
```

파일을 지정된 디렉터리에 복사한다.

```
$ cp 파일1 ... 파일n 디렉터리
```

여러 개의 파일들을 지정된 디렉터리에 모두 복사한다.

- 예

```
$ cp cs1.txt /tmp
$ ls -l /tmp/cs1.txt
-rw-r--r-- 1 chang cs 2088 4월 16일 14:31 /tmp/cs1.txt
$ cp cs1.txt cs2.txt /tmp
```

파일 복사: cp(copy)

- 디렉터리 전체 복사 : cp –r

```
$ cp [-r] 디렉터리1 디렉터리2
```

r은 리커전 옵션으로 디렉터리1 전체를 디렉터리2에 복사한다.

  - 하위 디렉터리를 포함한 디렉터리 전체를 복사

- 예

```
$ cp -r test temp
```

일반파일1 일반파일2
어떤일이 발생할까?
됨

mv 명령어

- mv(move)
  파일1의 이름을 파일2로 변경한다. (rename의 개념)

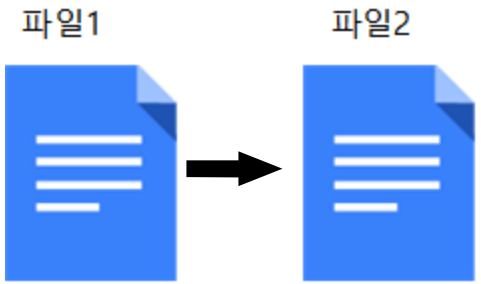
  $ mv 파일1 파일2
  $ mv cs2.txt cs3.txt
  $ ls -l
  -rw-r--r-- 1 chang faculty 2088 4월 16일 13:37 cs1.txt
  -rw-r--r-- 1 chang faculty 2088 4월 16일 13:56 cs3.txt

- 파일을 디렉토리 내로 이동
  $ mv 파일 디렉토리
  $ mv cs3.txt /tmp

파일 이동: mv(move)

- 사용법

> $ mv [-i] 파일1 파일2
>
> 파일1의 이름을 파일2로 변경한다. -i는 대화형 옵션이다.



- 예

```
$ mv cs2.txt cs3.txt
$ ls -l
-rw-r--r-- 1 chang faculty 2088 4월 16일 13:37 cs1.txt
-rw-r--r-- 1 chang faculty 2088 4월 16일 13:56 cs3.txt
```

파일 이동: mv(move)

- 대화형 옵션: mv -i
  - 이동 대상 파일과 이름이 같은 파일이 이미 존재하면 덮어쓰기(overwrite)
  - 보다 안전한 사용법: 대화형 –i(interactive) 옵션을 사용

- 예

```
$ mv –i cs1.txt cs3.txt
mv: overwrite ‘cs3.txt’? n
```

파일 이동: mv(move)

- 파일을 디렉터리로 이동

```
$ mv 파일 디렉터리
```

mv 파일1 파일2
이름 변경

파일을 지정된 디렉터리로 이동한다.

```
$ mv 파일1 ... 파일n 디렉터리
```

디렉터리 디렉터리 파일 (안됨)

여러 개의 파일들을 지정된 디렉터리로 모두 이동한다.

- 예

```
$ mv cs3.txt /tmp
$ ls -l /tmp/cs3.txt
-rw-r--r-- 1 chang cs 2088 4월 16일 13:56 /tmp/cs3.txt
$ mv cs1.txt cs3.txt /tmp
```

파일 이동: mv(move)

- 디렉터리 이름 변경

```
$ mv 디렉터리1 디렉터리2
```

디렉터리1을 지정된 디렉터리2로 이름을 변경한다.

- 예

```
$ mkdir temp
$ mv temp tmp
```

기본적으로 디렉토리도 파일이기 때문에
파일에 작업 가능한 명령어는 디렉토리에서도 가능.

파일/디렉토리 삭제

- rm(remove) 명령어
  명령줄 인수로 받은 파일(들)을 지운다.

  $ rm 파일+
  $ rm cs1.txt

- $ rm -r 디렉토리
  디렉토리 내의 모든 파일 및 하위 디렉토리들을 **단번에** 지운다.

- rmdir(remove directory) 명령어
  명령줄 인수로 받은 디렉토리(들)을 지운다.
  $ rmdir 디렉토리+
  주의: 디렉토리 내에 아무 것도 없어야 한다.
  $ rmdir test

파일 삭제: rm(remove)

- 사용법

```
$ rm [-i] 파일+
```

파일(들)을 삭제한다. -i는 대화형 옵션이다.

- 예

```
$ rm cs1.txt
$ rm cs1.txt cs3.txt
```

- 대화형 옵션 : rm -i

```
$ rm –i cs1.txt
rm: remove 'cs1.txt'? n
```

디렉터리 전체 삭제

- 디렉터리 전체 삭제: rm –r

rm -f 디렉토리
삭제 x

```
$ rm [-ri] 디렉터리
```

-r은 리커전 옵션으로 디렉터리 아래의 모든 것을 삭제한다. -i는 대화형 옵션

- 예

```
$ rm test
rm: cannot remove 'test': 디렉터리입니다
$ rmdir test
rmdir: failed to remove 'test': 디렉터리가 비어있지 않음
$ rm –ri test
rm: descend into directory 'test'? y
rm: remove regular file 'test/cs3.txt'? y
Rm: remove regular file ‘test/cs1.txt’? y
rm: remove directory 'test'? y
```

- –i : 확인 후 삭제
- –f : 무조건 삭제
- –r : 디렉토리 삭제

파일 관련 명령어

명령어	의미
cat 파일*	파일 디스플레이
more 파일+	한 번에 한 페이지씩 디스플레이
head 파일*	파일의 앞부분 디스플레이
tail 파일*	파일의 뒷부분 디스플레이
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
grep 키워드 파일	파일에서 키워드 찾기

(참고) 파일 관련 응용 명령어

- 파일내의 라인, 단어, 문자 개수 (wc)
- 파일 정렬 : 알파벳 순서로 각 행을 정렬 (sort)
- 파일 분할 : 파일을 지정 행수(#행)별로 구분 (split)
- 파일 내 인접행 비교 : 반복행 삭제 (uniq)
- 파일 가공 : 파일내의 특정 필드값 추출 (cut)
- 파일 가공 : 특정 필드의 행별 결합(paste)

- 패턴 검색 (grep)
- 파일 검색 (find)

- 파일 내의 문자 치환 : 파일내 str1을 str2로 치환 (tr)
- 파일 비교 : 두 파일의 동일 여부 검사 (cmp)
- 파일 비교 : 두 파일의 한쪽 혹은 양쪽에 있는 라인 표시 (comm)
- 파일 비교 : 두 파일의 차이점 분석 (diff)

- 라인 단위 편집 (sed)
- 필드 단위의 패턴 처리 (awk)