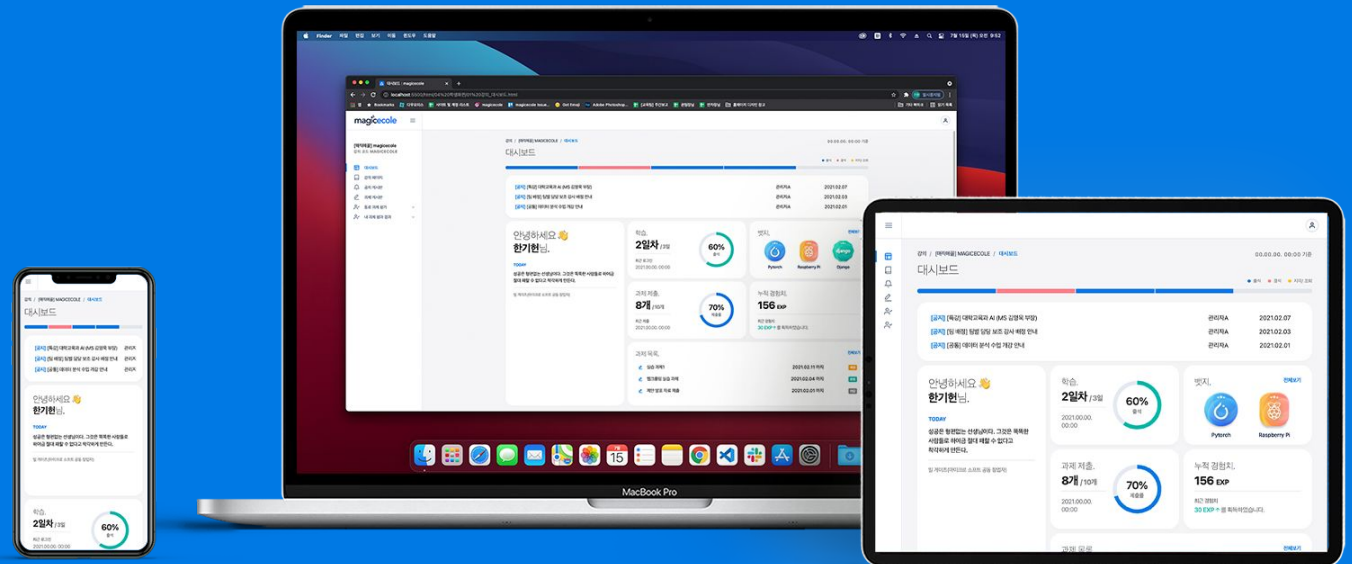


DevOps – Linux

파일과 디렉토리



다룰 내용

- 리눅스의 파일과 디렉토리
- 디렉토리 관련 명령
- 파일 관련 명령

파일(File)?

내 컴퓨터에서의 파일

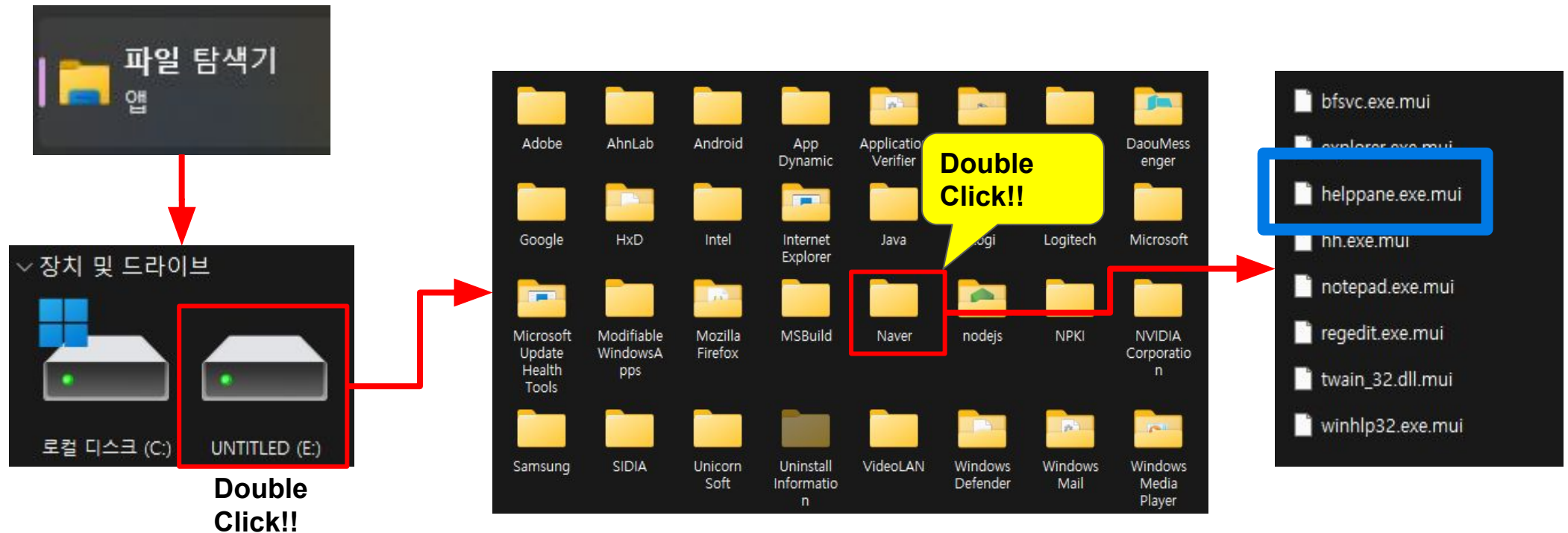
내 PC

C: 또는 D: 또는 ...



내 컴퓨터에서의 파일

파일 조회 / 검색 / 실행



내 컴퓨터에서의 파일

파일 이동 / 복사 / 삭제



Right Click!!



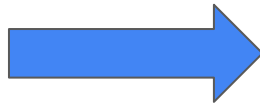
파일 저장



파일?



종이 형태의 **문서**
(서류)를 목적에 따라
정리



참조를 목적

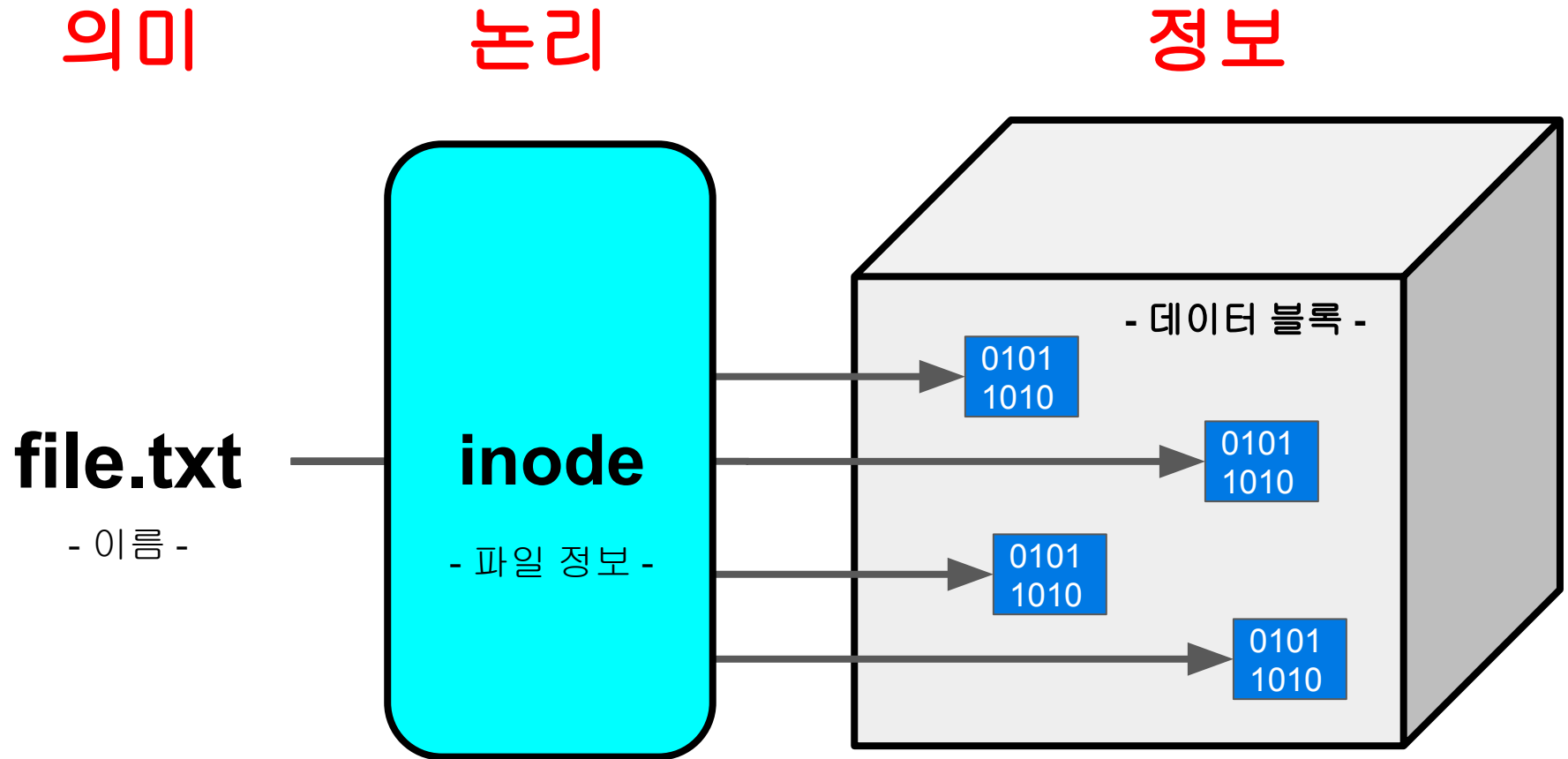
컴퓨터 파일의 정의

컴퓨터 파일

문서 토론

위키백과, 우리 모두의 백과사전.

컴퓨터 파일(computer file, 순화어: 기록철)은 컴퓨터 등의 기기에서 의미가 있는 정보를 담은 논리적인 단위이다. 하드디스크, CD, DVD 등 저장매체에 대하여 추상화된 정보 단위이다. 운영체제는 파일 조작에 관련된 기능을 API로 제공한다. 일반적으로 파일의 이름과 확장자로 식별하며, 운영 체제에 따라 대소문자를 구별하거나 구별하지는 않는다.



컴퓨터의 데이터를 이름으로 참조 가능

리눅스 파일의 종류

일반 파일

디렉토리

파일 링크

장치 파일

파이프

소켓

일반 파일

```
merged_personal_info.txt - 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
name : zcbyg
age : 25
e-mail : graxagsf@needle.worm
division : ptn
telephone : 010-9025-3590
sex : male

name : mftul
age : 16
e-mail : qoqueqbm@needle.worm
division : aub
telephone : 010-6716-4967
sex : others

name : gynhk
age : 35
e-mail : puvututy@needle.worm
division : bex
telephone : 010-4335-4243
sex : others

name : uridx
age : 82
e-mail : ksakszqi@needle.worm
division : haw
telephone : 010-7482-7774
sex : male

name : hjmxk
age : 41
e-mail : fjrhctfy@needle.worm
division : fev
telephone : 010-7841-0378
sex : male
```

(ASCII) 텍스트 파일

메모장으로 열어 볼 수 있는 파일

```
CL5500-BPRHS_V2.9...R)_r1677_TEST.bin  CL5500-BPHRS V2.94.3(m)_r1812.bin  CL5500_EUR
0000fca0 CD 01 D8 01 D3 01 D6 01 D9 01 DC 01 DF 01 E2 01 .....
0000fcb0 E5 01 E8 01 EB 01 EE 01 F1 01 F4 01 F8 01 FA 01 .....
0000fcc0 FC 01 FE 01 FF 01 00 02 01 02 02 02 03 02 04 02 .....
0000fcd0 22 00 22 00 22 00 22 00 22 00 22 00 22 00 .....
0000fce0 22 00 22 00 25 32 64 20 25 31 30 73 20 25 73 00 .....
0000fcf0 25 33 64 20 20 25 36 6C 64 20 20 25 32 64 00 00 .....
0000fd00 25 34 64 20 25 2D 31 33 73 25 35 64 25 36 6C 64 .....
0000fd10 00 00 00 00 58 57 50 54 49 4E 44 39 00 00 00 00 .....
0000fd20 25 30 32 64 20 3A 20 00 80 25 00 00 00 4B 00 00 .....
0000fd30 00 96 00 00 00 E1 00 00 00 C2 01 00 80 25 00 00 .....
0000fd40 00 4B 00 00 00 96 00 00 00 E1 00 00 00 C2 01 00 .....
0000fd50 52 6F 75 6E 64 65 64 20 54 61 72 65 00 00 00 00 .....
0000fd60 25 33 64 20 25 36 73 25 33 73 20 25 2E 31 34 73 .....
0000fd70 00 00 00 00 25 33 64 20 25 36 2E 32 66 25 25 20 .....
0000fd80 25 73 00 00 25 32 64 20 20 20 20 25 63 20 20 20 .....
0000fd90 20 20 20 25 36 2E 32 66 20 20 20 20 25 33 64 20 .....
0000fda0 3A 25 30 33 33 64 2C 33 64 2C 33 64 2C .....
0000fdb0 25 30 33 64 FF FF FF FF FF FF FF FF FF FF .....
0000fdc0 00 00 00 00 FF FF FF FF FF FF FF FF FF FF .....
0000fdd0 12 01 00 02 02 01 02 .....@..@W....
0000fde0 03 01 00 00 09 02 43 00 02 01 00 C0 00 09 04 00 .....C.....
0000fdf0 00 01 02 02 01 00 05 24 00 10 01 05 24 01 00 01 .....$.$.$.$.
0000fe00 04 24 02 02 05 24 06 00 01 07 05 82 03 08 00 FF .....$.$.$.$.
0000fe10 09 04 01 00 02 0A 00 00 00 07 05 03 02 40 00 00 .....@.....
0000fe20 07 05 81 02 40 00 00 00 04 03 09 04 26 03 53 00 .....@.....&.S.
0000fe30 54 00 40 00 69 00 63 00 72 00 6F 00 65 00 6C 00 .....T.M.i.c.r.o.e.l.
0000fe40 65 00 63 00 74 00 72 00 6F 00 6E 00 69 00 63 00 .....e.c.t.r.o.n.i.c.
0000fe50 73 00 00 00 34 03 53 00 54 00 52 00 39 00 31 00 .....s...4.S.T.R.9.1.
0000fe60 78 00 20 00 56 00 69 00 72 00 74 00 75 00 61 00 .....x..U.i.r.t.u.a.
0000fe70 6C 00 20 00 43 00 4F 00 4D 00 20 00 50 00 6F 00 .....l..C.O.M..P.o.
0000fe80 72 00 74 00 20 00 20 00 16 03 44 00 65 00 6D 00 .....r.t....D.e.m.
0000fe90 6F 00 20 00 31 00 2E 00 30 00 30 00 30 00 00 00 .....o..1...0.0.0...
0000fea0 43 4C 31 31 30 38 45 55 30 44 30 32 32 39 34 00 .....CL1108EU0002294.
0000feb0 28 52 29 00 20 20 00 00 25 33 64 2E 25 33 64 2C .....(R)...%3d.%3d,
0000fec0 25 2E 31 36 73 00 00 00 00 00 21 15 11 0E 00 0C .....%.16s.....!.....
0000fed0 0B 0B 0A 0A 09 09 09 09 09 08 08 08 08 08 08 .....
0000fee0 07 07 07 07 07 07 07 07 07 07 07 07 07 00 00 .....
0000fef0 30 31 32 33 34 35 36 37 38 39 61 62 63 64 65 66 .....0123456789abcdef
```

파일의 실체

바이너리 파일을
ASCII로 변환한 내용

바이너리(이진) 파일

실행 파일, 미디어 / 이미지 파일 등



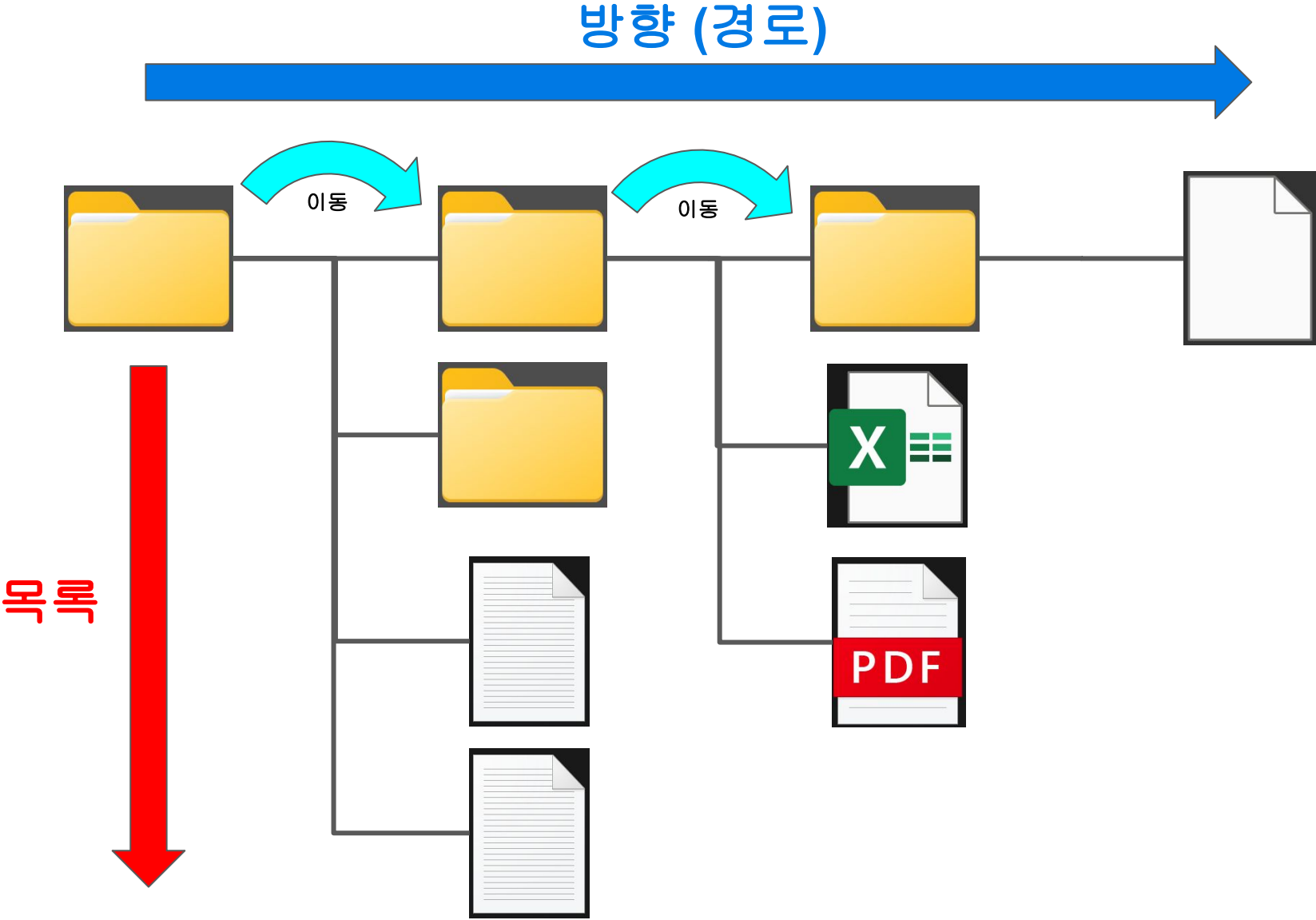
흔히 [폴더]라 부르는 것

디렉토리의 의미?

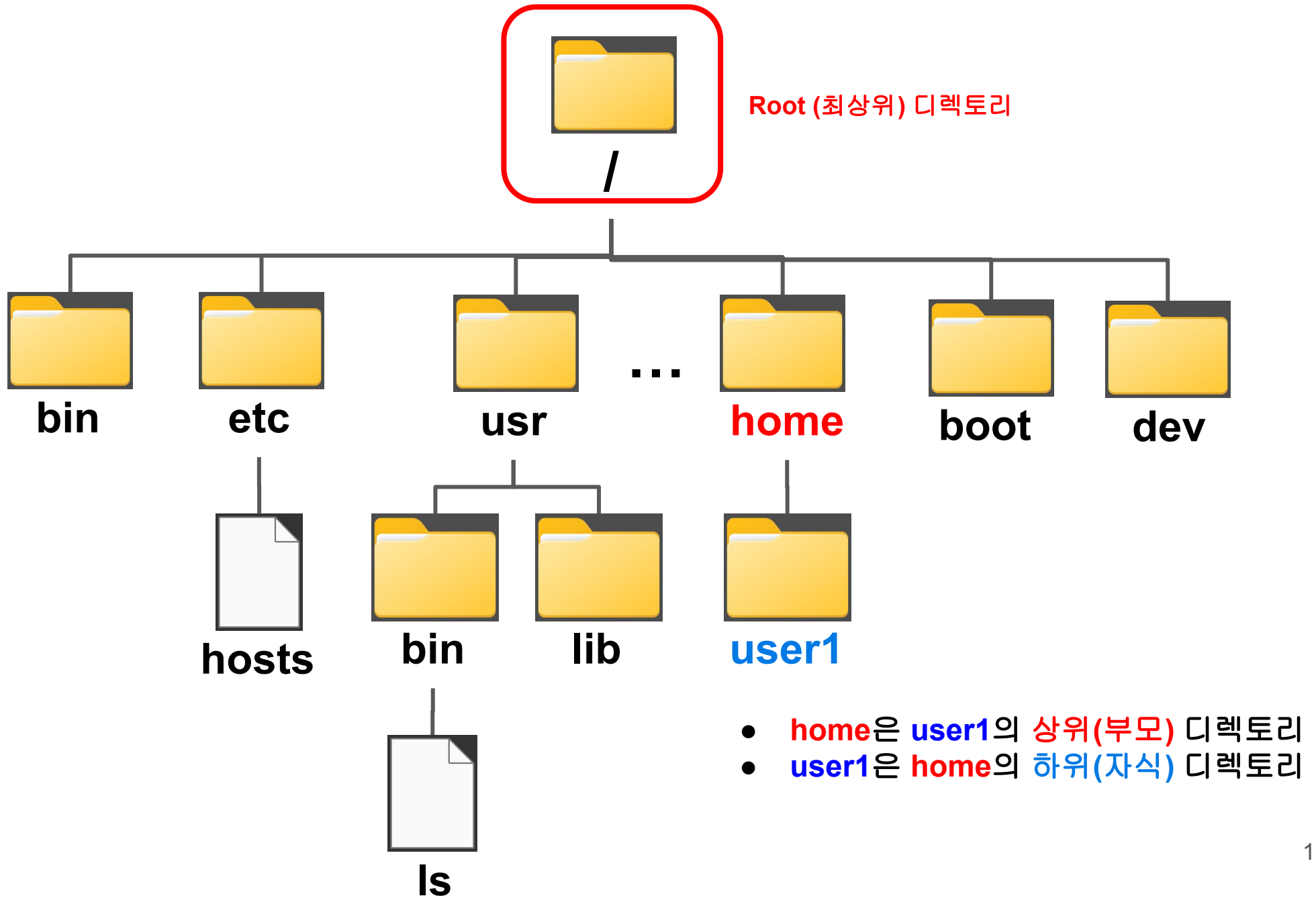
Directory

방향
(Direction)

목록, 저장소
(List)



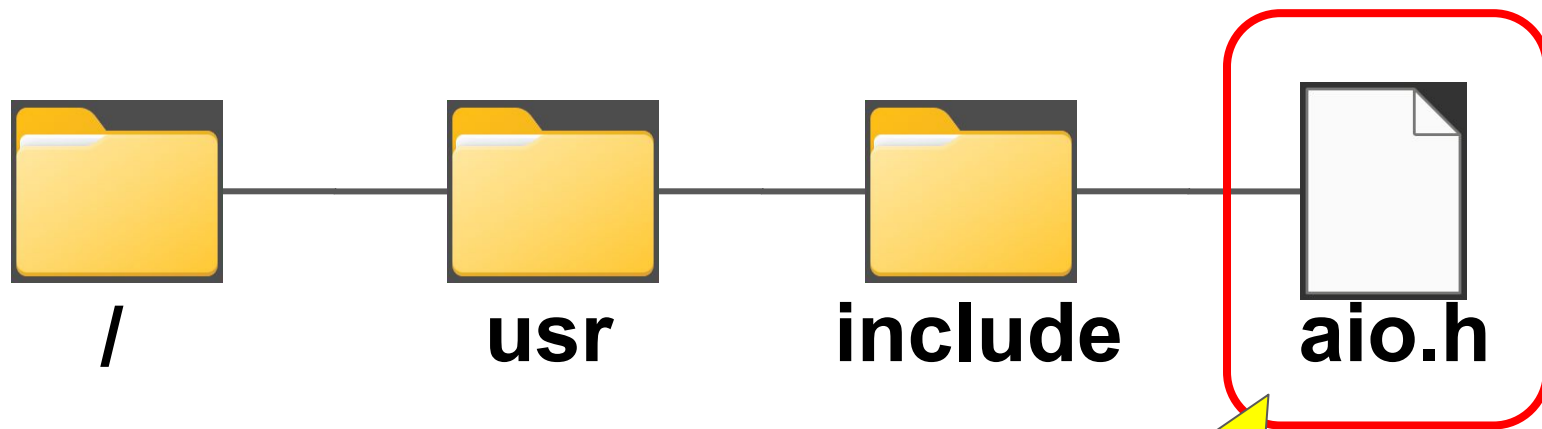
디렉토리 - 리눅스 디렉토리 계층



디렉토리 - 리눅스의 주요 디렉토리

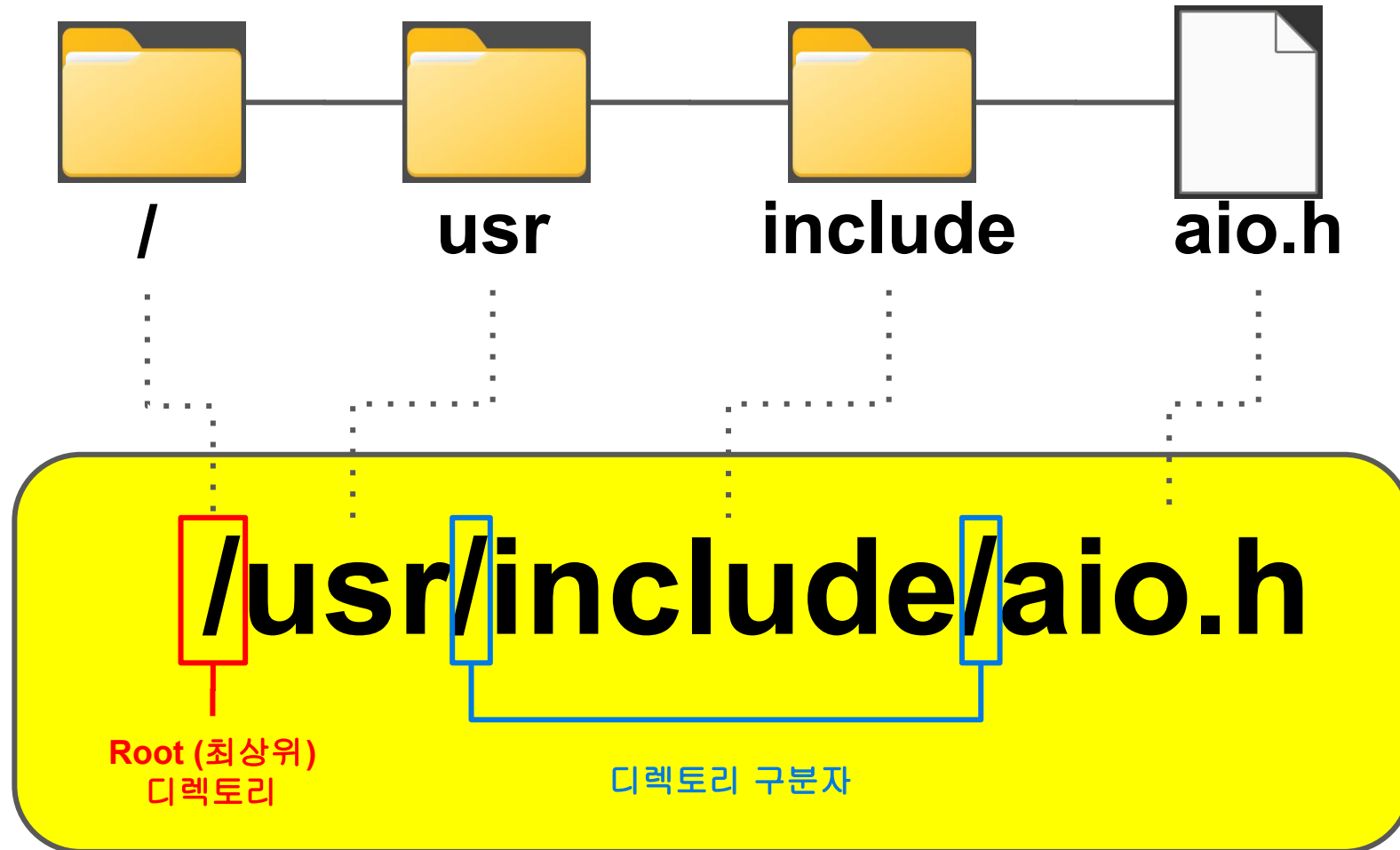
디렉토리	기능
dev	장치 파일이 담긴 디렉토리다.
home	사용자 홈 디렉토리가 생성되는 디렉토리다.
root	관리자(root) 계정의 홈 디렉토리다. Root(/) 디렉토리와 다름에 주의한다.
usr	기본 실행 파일과 라이브러리 파일, 헤더 파일 등이 있다. 참고로 usr는 “Unix System Resource”의 약자이다.
boot	부팅에 필요한 커널 파일을 가지고 있다.
etc	리눅스 설정을 위한 각종 파일을 가지고 있다.
mnt	파일 시스템을 마운트하는 디렉토리다.
proc	프로세스 정보 등 커널 관련 정보가 저장되는 디렉토리다.
tmp	시스템 사용 중에 발생하는 임시 데이터가 저장된다. 이 디렉토리에 있는 파일은 재시작하면 모두 삭제된다.
var	시스템 운영 중에 발생하는 데이터나 로그 등 내용이 자주 바뀌는 파일이 주로 저장된다.

디렉토리 - 파일 경로(Path)



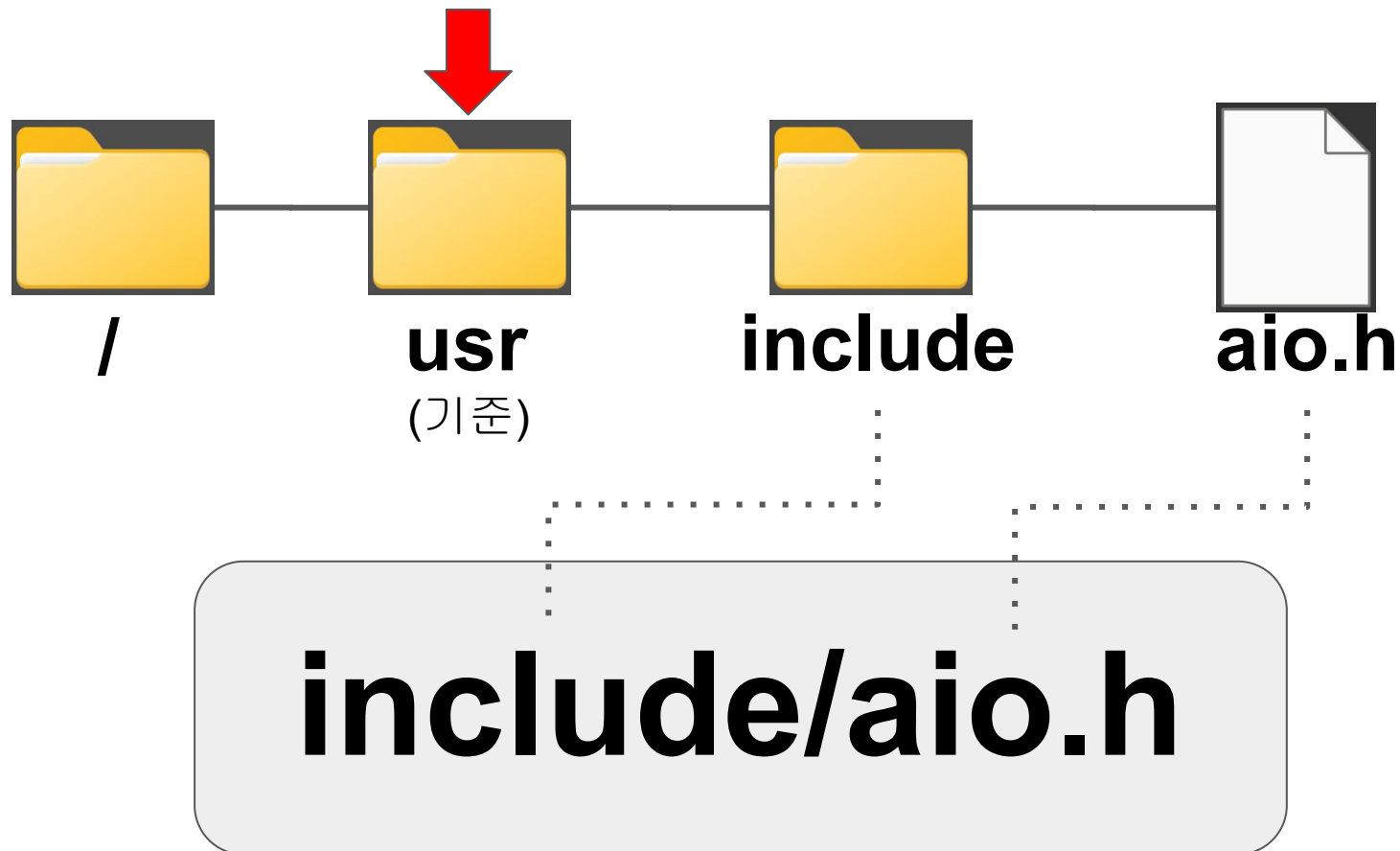
이 파일의 경로명은?

디렉토리 - 파일 경로(Path) - 절대 경로



디렉토리 - 파일 경로(Path) - 상대 경로

현재 **/usr** 에서 작업 중일 때

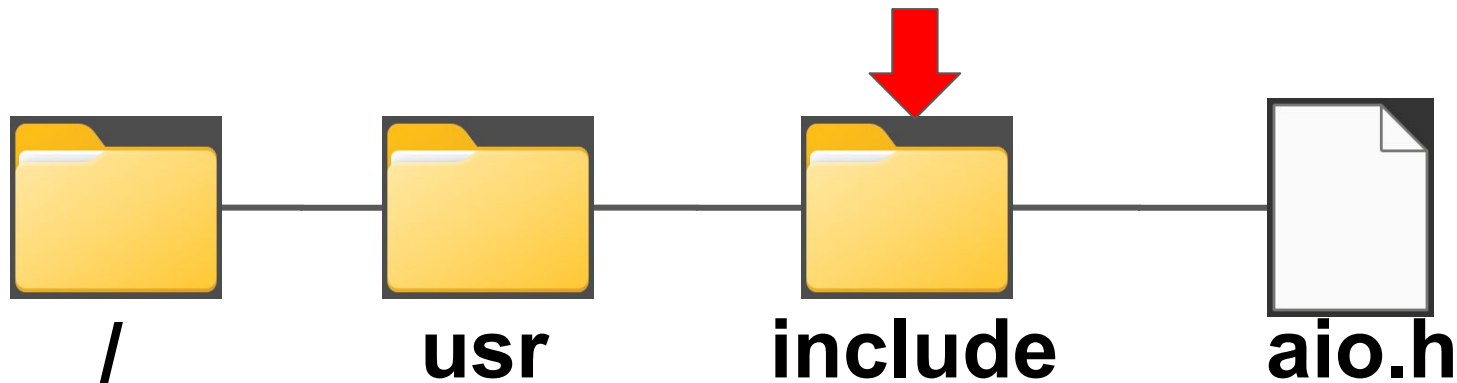


디렉토리 - 파일 경로(Path) - 상대 경로

./ : 현재 디렉토리

../ : 이전(상위) 디렉토리

현재 **/usr/include**에서 작업 중일 때



/usr/include = ./

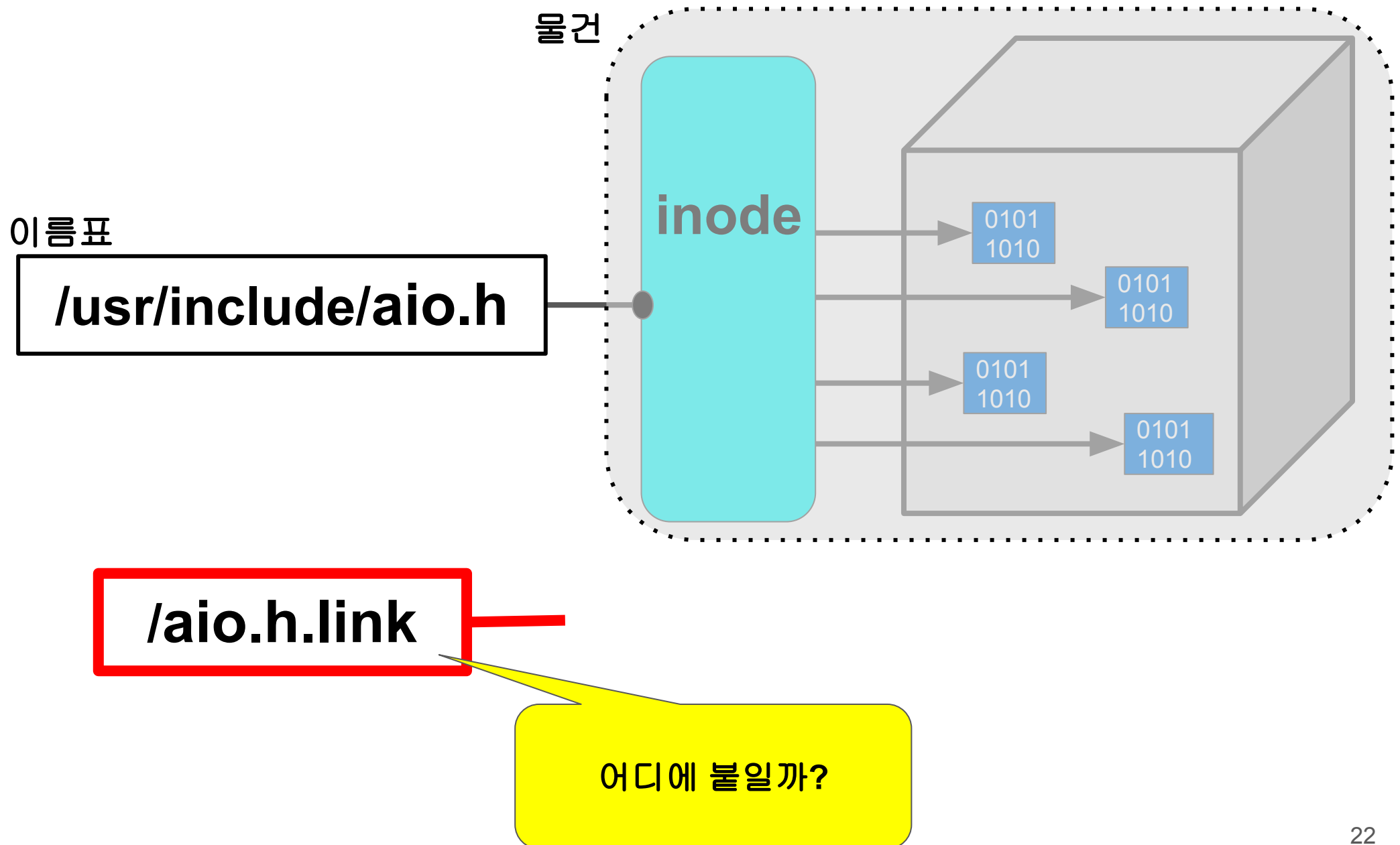
/usr = ../

/ = ../../

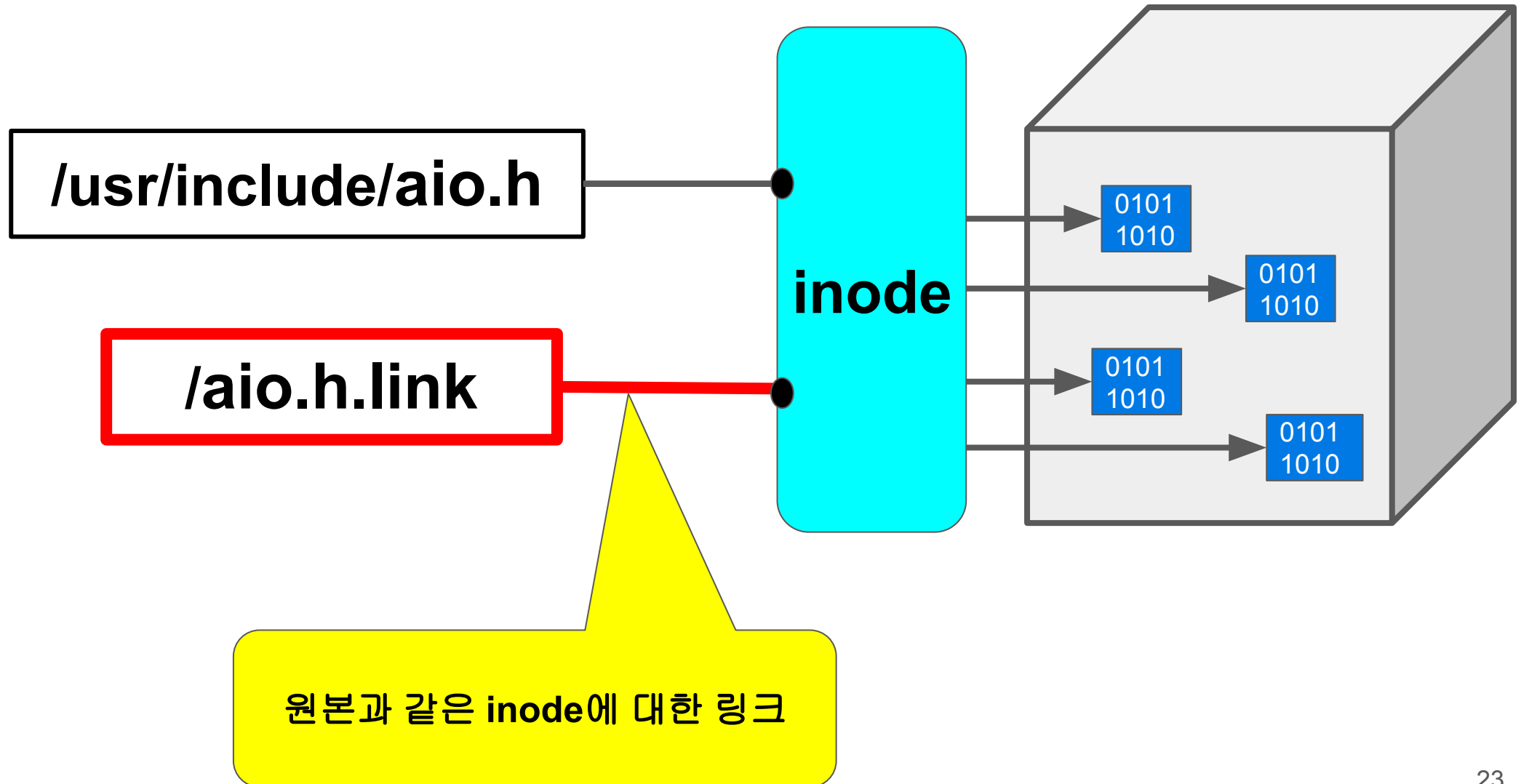
으로 표현 가능

파일에 붙인
새로운 이름

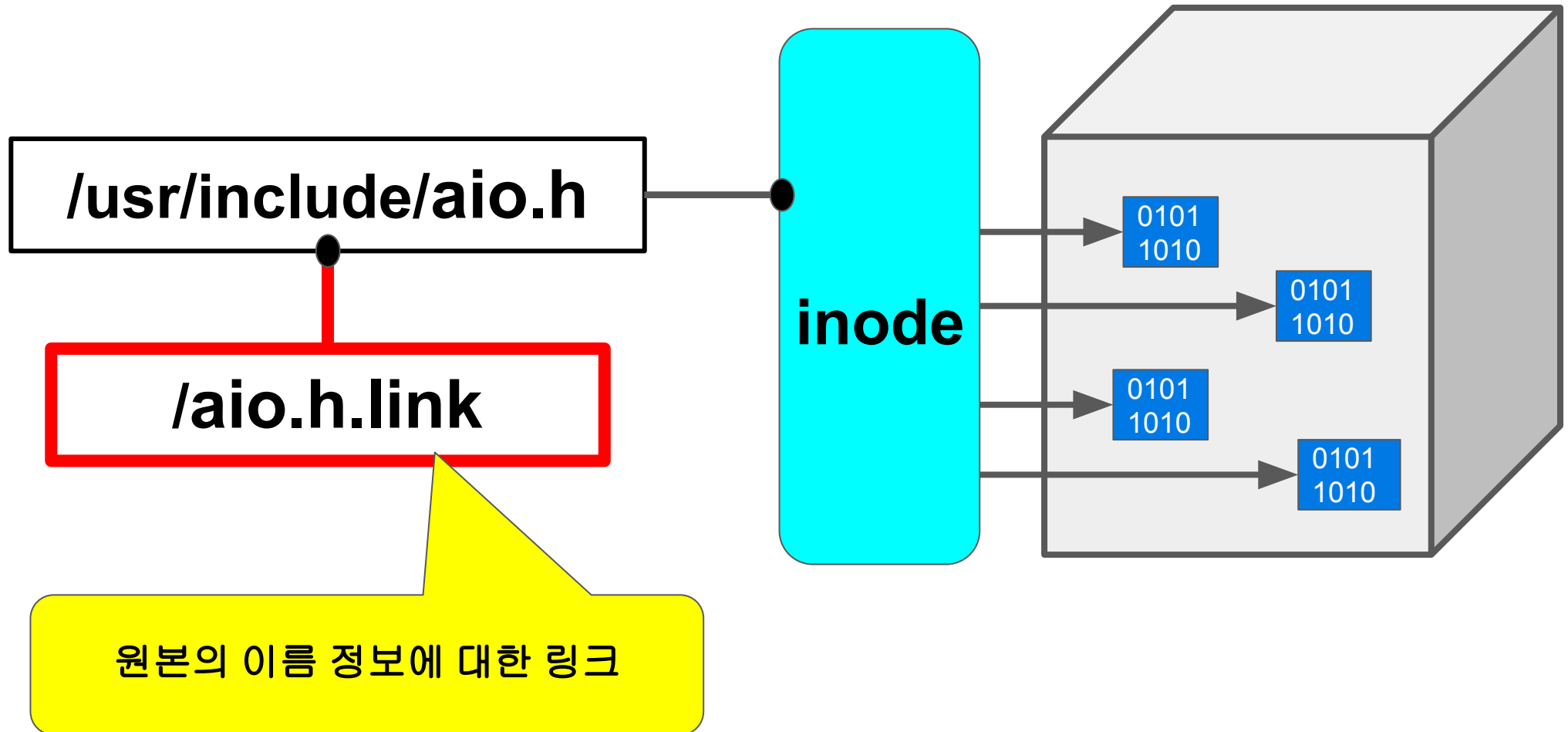
파일 링크



파일 링크 - 하드 링크



파일 링크 - 심볼릭 링크



심볼릭 링크

- 파일과 디렉토리 모두 지원
- 파일 또는 디렉토리 이름에 대한 링크
- 원본 파일이 삭제될 경우 액세스 불가능
- 윈도우의 바로가기

하드 링크

- 파일만 지원
- 원본 파일의 **inode**에 대한 포인터
- 원본 파일이 삭제되어도 액세스 가능

근데 이것 왜 쓸까? 쓰면 뭐가 좋을까?

생각해 보세요^^

리눅스에서는
장치도 파일이다?

장치 파일



/dev

```
autofs      hvc3        mem          stdout      tty27      tty47      ttyprintk   ttyS28      vcsa
block       hvc4        memory_bandwidth tty          tty28      tty48      ttyps0      ttyS29      vcsa1
fs-control  hvc5        mq           tty0        tty29      tty49      ttyps1      ttyS30      vcsa2
            hvc6        net          tty1        tty3       tty5       ttyps10     ttyS31      vcsa3
            hvc7        network_latency tty10       tty30      tty50      ttyps11     ttyS32      vcsa4
            hwrng      network_throughput tty11       tty31      tty51      ttyps12     ttyS33      vcsa5
core        initctl     null         tty12       tty32      tty52      ttyps13     ttyS34      vcsa6
cpu_dma_latency input       parport0     tty13       tty33      tty53      ttyps14     ttyS35      vfio
cuse        kmsg        port         tty14       tty34      tty54      ttyps15     ttyS36      vga_arbiter
disk        lightnvmm   ppp          tty15       tty35      tty55      ttyps16     ttyS37      vhci
dri         log         psaux        tty16       tty36      tty56      ttyps17     ttyS38      vhost-net
ecryptfs    loop0       ptmx         tty17       tty37      tty57      ttyps18     ttyS39      vhost-vsock
fb0         loop1       pts          tty18       tty38      tty58      ttyps19     ttyS40      xen
fd          loop2       random       tty19       tty39      tty59      ttyps20     ttyS41      xvda
full        loop3       rfkill       tty2         tty4       tty6       ttyps21     ttyS42      xvda1
fuse        loop4       rtc          tty20       tty40      tty60      ttyps22     ttyS43      zero
hidraw0     loop5       rtc0         tty21       tty41      tty61      ttyps23     ttyS44
hpet        loop6       shm          tty22       tty42      tty62      ttyps24     ttyS45
hugepages   loop7       snapshot     tty23       tty43      tty63      ttyps25     ttyS46
hvc0        loop-control snd          tty24       tty44      tty7       ttyps26     ttyS47
hvc1        mapper      stderr       tty25       tty45      tty8       ttyps27     ttyS48
hvc2        mcelog     stdin        tty26       tty46      tty9       ttyps28     ttyS49
```

- Device Driver와의 Interface
- 장치 파일을 통해 하드웨어 제어/통신 가능

문자 장치 (Character Device)

순차적으로 데이터 처리



사운드 카드

블록 장치 (Block Device)

블록 단위로 데이터 처리



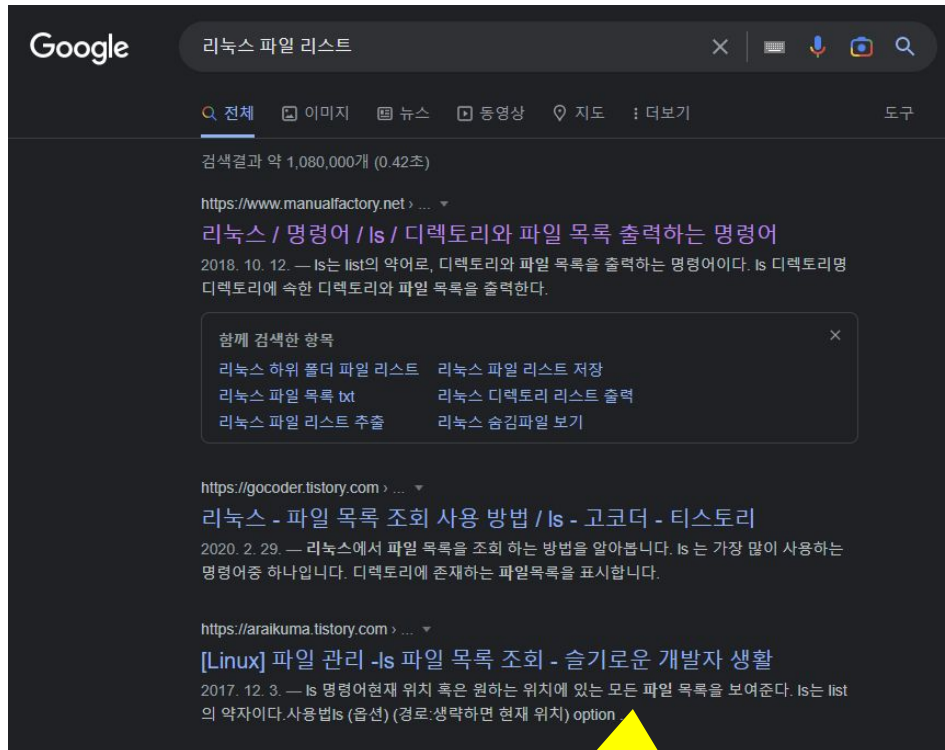
[파일 시스템] 단원에서
좀 더 자세히

관련 주요 명령어

명령어 학습 방법

1. 구글 검색

- 상황에 따라 어떤 명령을 써야 하는지 알고 싶을때



2. help 옵션, man page

- 특정 명령어의 사용법을 알고자 할 때 (세부 옵션 등)

```
root@linux-test:~# ls --help
Usage: ls [OPTION]... [FILE]...
List information about the FILES (the current directory by default).
Sort entries alphabetically if none of -cftuvSUX nor --sort is specified.

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            print C-style escapes for nongraphic characters
--block-size=SIZE       scale sizes by SIZE before printing them; e.g.,
                        '--block-size=M' prints sizes in units of
                        1,048,576 bytes; see SIZE format below
-B, --ignore-backups    do not list implied entries ending with ~
-c                      with -lt: sort by, and show, ctime (time of last
                        modification of file status information);
                        with -l: show ctime and sort by name;
                        otherwise: sort by ctime, newest first
-C                      list entries by columns
--color[=WHEN]          colorize the output; WHEN can be 'always' (default
                        if omitted), 'auto', or 'never'; more info below
```

```
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 specified.

  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
    print C-style escapes for nongraphic characters

  --block-size=SIZE
    scale sizes by SIZE before printing them; e.g., '--block-size=M' prints sizes in units of 1,048,576 bytes;
    see SIZE format below

  -B, --ignore-backups
    do not list implied entries ending with ~
```

man ls 명령

ChatGPT도 써보세요!!

디렉토리 명령

- **pwd** (**P**rint **W**orking **D**irectory)
 - **기능** : 현재 디렉토리 위치 출력 (절대경로)
 - **형식** : `pwd`

1. 현재 위치한 디렉토리 경로 표시 : `pwd`

```
root@linux-test:~# pwd  
/root
```

관리자(root) 계정의
홈 디렉토리는 /root

디렉토리 명령

● **cd** (C**h**ange **D**irectory)

- **기능** : 지정한 디렉토리로 이동
- **형식** : **cd** [디렉토리]

1. 절대 경로를 사용하여 /usr/include 디렉토리로 이동 : **cd /usr/include**

```
root@linux-test:~# cd /usr/include
root@linux-test:/usr/include#
```

이동한 디렉토리

2. 상대 경로를 사용하여 /usr/bin 디렉토리로 이동 : **cd ../bin**

```
root@linux-test:/usr/include# cd ../bin
root@linux-test:/usr/bin#
```

3. 직전에 위치했던 디렉토리로 이동 : **cd -**

```
root@linux-test:/usr/bin# cd -
/usr/include
root@linux-test:/usr/include#
```

4. 접속한 계정의 홈 디렉토리로 이동 : **cd ~**

```
root@linux-test:/usr/include# cd ~
root@linux-test:~#
```


디렉토리 명령

● ls (List)

- 기능 : 디렉토리의 내용을 출력
- 형식 : ls [옵션] [디렉토리(파일)]

1. 현재 디렉토리 내용 확인 : ls

```
root@linux-test:~# ls
root@linux-test:~#
```

2. Root 디렉토리 내용 확인 : ls /

```
root@linux-test:~# ls /
bin    home    lib      media   root    srv     var
boot   home1   lib32    mnt     run     sys     vmlinuz
dev    initrd.img  lib64    opt     sbin    tmp     vmlinuz.old
etc    initrd.img.old  lost+found  proc    snap    usr
```

3. 현재 디렉토리의 모든 파일 표시 : ls -a

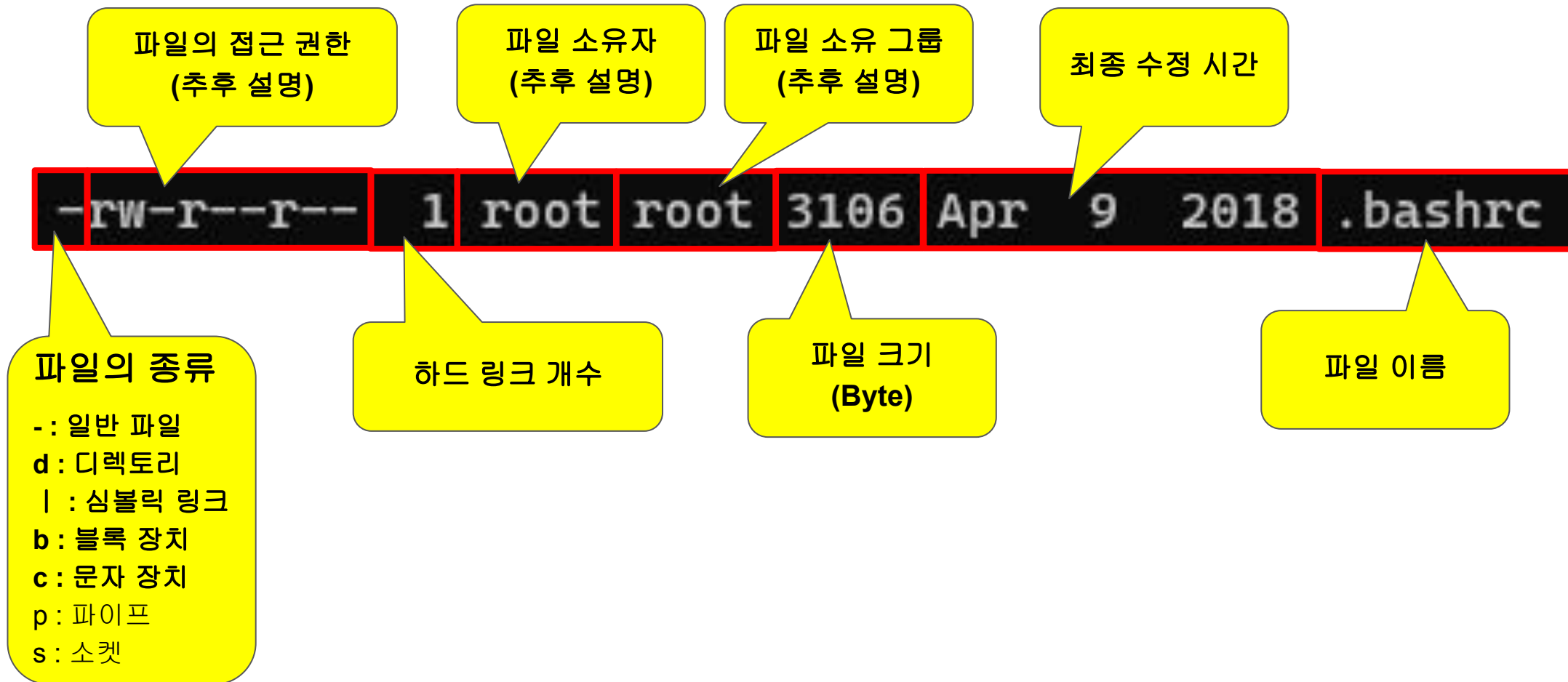
```
root@linux-test:~# ls -a
.  .bash_history  .cache  .local  .profile  .ssh
.. .bashrc       .gnupg  .nsight .selected_editor  .vim
```

‘.’으로 시작하는 파일명
=> 숨김 파일 표시

4. /tmp 디렉토리의 모든 파일의 상세정보 표시 : ls -al /tmp ls -a -l /tmp

```
root@linux-test:~# ls -al /tmp
total 36
drwxrwxrwt  9 root root 4096 Feb 15 10:05 .
drwxr-xr-x 25 root root 4096 Oct 13  2020 ..
drwxrwxrwt  2 root root 4096 Feb 15 09:34 .font-unix
drwxrwxrwt  2 root root 4096 Feb 15 09:34 .ICE-unix
-rw-r--r--  1 root root    0 Feb 15 09:34 .nu_err.log
-rw-r--r--  1 root root    0 Feb 15 09:34 .nu.log
drwx-----  3 root root 4096 Feb 15 09:34 systemd-private-1
drwx-----  3 root root 4096 Feb 15 09:34 systemd-private-1
drwxrwxrwt  2 root root 4096 Feb 15 09:34 .Test-unix
drwxrwxrwt  2 root root 4096 Feb 15 09:34 .X11-unix
drwxrwxrwt  2 root root 4096 Feb 15 09:34 .XIM-unix
```

파일의 상세정보



디렉토리 명령

5. 현재 디렉토리의 모든 파일에 대한 inode 번호 검색: **ls -ia**

ls -i -a

```
root@linux-test:~# ls -ia
524289 .      524323 .bash_history  524292 .cache  524305 .local  524291 .profile  524298 .ssh
  2 ..    524290 .bashrc      524296 .gnupg   524303 .nsight  524301 .selected_editor  524300 .vim
```

inode 번호

명령어 입력 팁

자동 완성으로 명령어 입력(Tab 키 사용)

```
root@linux-test:~# l 치고 Tab!  
Display all 105 possibilities? (y or n)
```

l	lexgrog	logger	lsmem	lvmsadc
la	libnetcfg	login	lsmod	lvmsar
landscape-sysinfo	link	loginctl	lsns	lvreduce
laptop-detect	linux32	logname	lsof	lvremove
last	linux64	logout	lspci	lvrename
lastb	linux-boot-prober	logrotate	lspgpot	lvresize
lastlog	linux-check-removal	logsave	lsusb	lvs
lcf	linux-update-symlinks	look	ltrace	lvscan
ld	linux-version	lorder	luksformat	lxc
ldattach	ll	losetup	lvchange	lxcfs
ld.bfd	ln	lowntfs-3g	lvconvert	lxd
ldconfig	lnstat	ls	lvcreate	lzcat
ldconfig.real	loadkeys	lsattr	lvdisplay	lzcmp
ldd	loadunimap	lsblk	lvextend	lzdiff
ld.gold	local	lsb_release	lvm	lzegrep
less	locale	lscpu	lvmconf	lzfgrep
lessecho	locale-check	lshw	lvmconfig	lzgrep
lessfile	localectl	lsinitramfs	lvmdiskscan	lzless
lesskey	localedef	lsipc	lvmdump	lzma
lesspipe	locale-gen	lslocks	lvmetad	lzmainfo
let	locate	lslogins	lvmpolld	lzmore

'l'로 시작하는 명령어 참조

```
root@linux-test:~# ls SPACE  
치고 Tab!
```

현재 경로의 파일 / 디렉토리 참조
(인자)

./	.bashrc	.local/	.selected_editor	.viminfo
../	.cache/	.nsight/	.ssh/	.vscode-server/
.bash_history	.gnupg/	.profile	.vim/	.wget-hsts

디렉토리 명령

- **mkdir** (MaKeDIRectory)
 - **기능** : 디렉토리 생성
 - **형식** : **mkdir** [옵션] [디렉토리]

1. temp라는 이름의 디렉토리 1개 생성 : **mkdir temp**

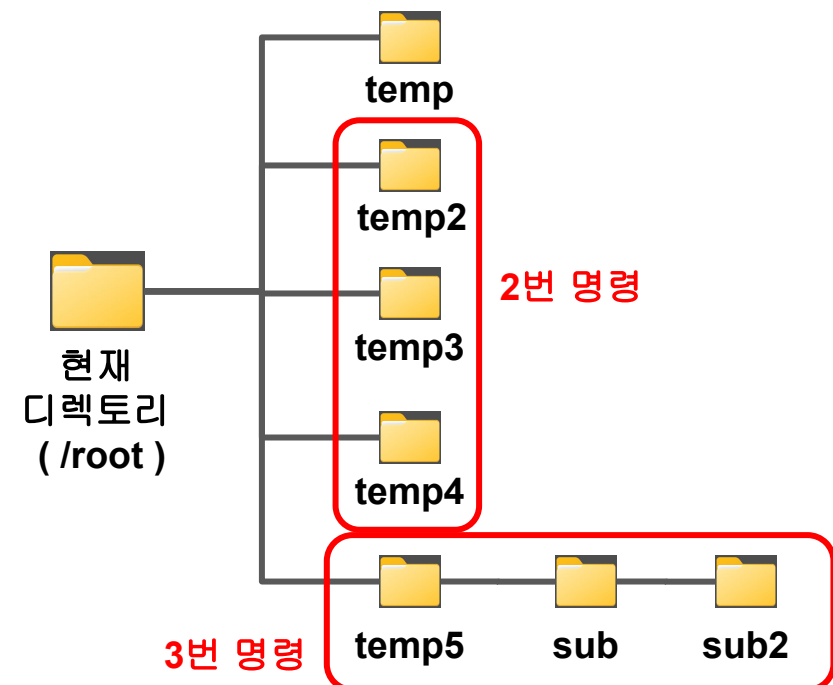
```
root@linux-test:~# mkdir temp
root@linux-test:~# ls
temp
```

2. 여러 디렉토리를 동시에 생성 : **mkdir temp2 temp3 temp4**

```
root@linux-test:~# mkdir temp2 temp3 temp4
root@linux-test:~# ls
temp temp2 temp3 temp4
```

3. 하위 디렉토리 함께 생성 : **mkdir -p temp5/sub/sub2**

```
root@linux-test:~# mkdir -p temp5/sub/sub2
root@linux-test:~# ls
temp temp2 temp3 temp4 temp5
root@linux-test:~# ls -R temp5
temp5:
sub
temp5/sub:
sub2
```



디렉토리 명령

- **rmdir** (ReMoveDIRectory)
 - **기능** : 디렉토리 삭제
 - **형식** : **rmdir** [옵션] [디렉토리]

1. temp 디렉토리 삭제 : **rmdir temp**

```
root@linux-test:~# ls
temp  temp2  temp3  temp4  temp5
root@linux-test:~# rmdir temp
root@linux-test:~# ls
temp2  temp3  temp4  temp5
```

2. 빈 디렉토리가 아닐 경우 에러 발생

```
root@linux-test:~# rmdir temp5
rmdir: failed to remove 'temp5': Directory not empty
```

temp5에는 하위 디렉토리가 존재 (비어있지 않음)

파일 명령

- **cat** (con**CAT**nate : 사슬같이 있다)
 - **기능** : 파일 내용을 화면에 출력한다
 - **형식** : **cat** [옵션] [파일]

1. .profile 파일의 내용 출력 : **cat .profile**

```
root@linux-test:~# cat .profile
# ~/.profile: executed by Bourne-compatible login shells.

if [ "$BASH" ]; then
  if [ -f ~/.bashrc ]; then
    . ~/.bashrc
  fi
fi

mesg n || true
```

2. 행 번호를 붙여서 출력 : **cat -n .profile**

```
root@linux-test:~# cat -n .profile
1 # ~/.profile: executed by Bourne-compatible login shells.
2
3 if [ "$BASH" ]; then
4   if [ -f ~/.bashrc ]; then
5     . ~/.bashrc
6   fi
7 fi
8
9 mesg n || true
```

행 번호 표시

3. .profile, .selected_editor 파일의 내용을 붙여서 출력 : **cat .profile .selected_editor**

```
root@linux-test:~# cat .profile .selected_editor
# ~/.profile: executed by Bourne-compatible login shells.

if [ "$BASH" ]; then
  if [ -f ~/.bashrc ]; then
    . ~/.bashrc
  fi
fi

mesg n || true
# Generated by /usr/bin/select-editor
SELECTED_EDITOR="/usr/bin/vim.basic"
```

.profile

.selected_editor

파일 명령

● more

- **기능** : 파일 내용을 화면 단위로 출력한다
- **형식** : **more** [옵션] [파일]

1. .bashrc 파일의 내용 출력 : **more .bashrc**

```
root@linux-test:~# more .bashrc
# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
# for examples

# If not running interactively, don't do anything
[ -z "$PS1" ] && return

# don't put duplicate lines in the history. See bash(1) for more options
# ... or force ignoredups and ignorespace
HISTCONTROL=ignoredups:ignorespace

# append to the history file, don't overwrite
shopt -s histappend

# for setting history length see HISTSIZE and HISTFILESIZE in bash(1)
HISTSIZE=1000
HISTFILESIZE=2000

# check the window size after each command and, if necessary,
# update the values of LINES and COLUMNS.
shopt -s checkwinsize

# make less more friendly for non-text input files, see lesspipe(1)
[ -x /usr/bin/lesspipe ] && eval "$(SHELL=/bin/sh lesspipe)"

# set variable identifying the chroot you work in (used in the prompt below)
if [ -z "$debian_chroot" ] && [ -r /etc/debian_chroot ]; then
    debian_chroot=$(cat /etc/debian_chroot)
fi

--More--(31%)
```

파일 내용

Space bar : 다음 화면 출력
Enter : 한 줄씩 스크롤
q : 나가기 (quit)

파일 명령

- **less** (기능이 추가된 more)
 - **기능** : 파일 내용을 화면 단위로 출력한다
 - **형식** : **less [파일]**

1. .bashrc 파일의 내용 출력 : **less .bashrc**

```
# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
# for examples

# If not running interactively, don't do anything
[ -z "$PS1" ] && return

# don't put duplicate lines in the history. See bash(1) for more options
# ... or force ignoredups and ignorespace
HISTCONTROL=ignoredups:ignorespace

# append to the history file, don't overwrite it
shopt -s histappend

# for setting history length see HISTSIZE and HISTFILESIZE in bash(1)
HISTSIZE=1000
HISTFILESIZE=2000

# check the window size after each command and, if necessary,
# update the values of LINES and COLUMNS.
shopt -s checkwinsize

# make less more friendly for non-text input files, see lesspipe(1)
[ -x /usr/bin/lesspipe ] && eval "$(SHELL=/bin/sh lesspipe)"

# set variable identifying the chroot you work in (used in the prompt below)
if [ -z "$debian_chroot" ] && [ -r /etc/debian_chroot ]; then
    debian_chroot=$(cat /etc/debian_chroot)
fi

# set a fancy prompt (non-color, unless we know we "want" color)
case "$TERM" in
    .bashrc
```

상하 방향키 : 위,아래 스크롤
 Space bar : 다음 화면 출력
 Ctrl + b : 이전 화면 출력
 q : 나가기 (quit)

파일 명령

- head

- 기능 : 파일 앞 부분의 몇 줄을 출력한다 (기본 10줄)
- 형식 : head [옵션][파일]

1. .bashrc 파일의 앞부분 10줄 출력 : head .bashrc

```
root@linux-test:~# head .bashrc
# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
# for examples

# If not running interactively, don't do anything
[ -z "$PS1" ] && return

# don't put duplicate lines in the history. See bash(1) for more options
# ... or force ignoredups and ignorespace
HISTCONTROL=ignoredups:ignorespace
```

2. .bashrc 파일의 앞부분 3줄 출력 : head -3 .bashrc

```
root@linux-test:~# head -3 .bashrc
# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
# for examples
```

파일 명령

- tail

- **기능** : 파일 마지막 부분의 몇 줄을 출력한다 (기본 10줄)
- **형식** : **tail** [옵션][파일]

1. .bashrc 파일의 마지막 10줄 출력 : **tail .bashrc**

```
root@linux-test:~# tail .bashrc
if [ -f ~/.bash_aliases ]; then
    . ~/.bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
#if [ -f /etc/bash_completion ] && ! shopt -oq posix; then
#     . /etc/bash_completion
#fi
```

2. .bashrc 파일의 마지막 5줄 출력 : **tail -5 .bashrc**

```
root@linux-test:~# tail -5 .bashrc
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
#if [ -f /etc/bash_completion ] && ! shopt -oq posix; then
#     . /etc/bash_completion
#fi
```

파일 명령

● cp (CoPy)

- 기능 : 파일이나 디렉토리를 복사
- 형식 : **cp** [옵션] [SOURCE 파일(디렉토리)]... [DEST 파일(디렉토리)]
무엇을 어디로

1. .bashrc 파일을 현재 디렉토리에 bashrc_copy라는 이름의 파일로 복사 : **cp .bashrc bashrc_copy**

```
root@linux-test:~# ls -a
.  .bash_history  .cache  .local  .profile  .ssh  temp3  temp5
.. .bashrc      .gnupg  .nsight  .selected_editor  temp2  temp4  .vim
root@linux-test:~# cp .bashrc bashrc_copy
root@linux-test:~# ls -a
.  .bash_history  bashrc_copy  .gnupg  .nsight  .selected_editor  temp2  temp4  .vim
.. .bashrc      .cache      .local  .profile  .ssh            temp3  temp5
root@linux-test:~# ls
bashrc_copy  temp2  temp3  temp4  temp5
```

숨김 파일이 아니므로 ls 명령으로도 확인 가능

```
root@linux-test:~# head bashrc_copy
# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
# for examples

# If not running interactively, don't do anything
[ -z "$PS1" ] && return

# don't put duplicate lines in the history. See bash(1) for more options
# ... or force ignoredups and ignorespace
HISTCONTROL=ignoredups:ignorespace
```

head bashrc_copy : 복사된 파일 내용 확인

2. .bashrc 파일을 temp2 디렉토리에 복사 :

cp .bashrc temp2

상대 경로 지정 (./temp2)

cp .bashrc /root/temp2

절대 경로 지정

cp .bashrc temp2/bashrc_copy

cp .bashrc /root/temp2/bashrc_copy

원본과 다른 파일명 지정

```
root@linux-test:~# cp .bashrc temp2
```

```
root@linux-test:~# ls -a temp2
```

```
.. .bashrc
```

```
root@linux-test:~# cp .bashrc temp2/bashrc_copy
```

```
root@linux-test:~# ls -a temp2
```

```
.. .bashrc bashrc_copy
```

3. .bashrc, .profile 파일을 temp3 디렉토리에 복사 : `cp .bashrc .profile temp3`

마지막에 지정한 디렉토리에 복사

```
root@linux-test:~# ls -a temp3
.  ..
root@linux-test:~# cp .bashrc .profile temp3
root@linux-test:~# ls -a temp3
.  ..  .bashrc  .profile
```

4. 이미 존재하는 파일인 경우 덮어쓸 것인지 질문 : `cp -i .bashrc temp2/bashrc_copy`

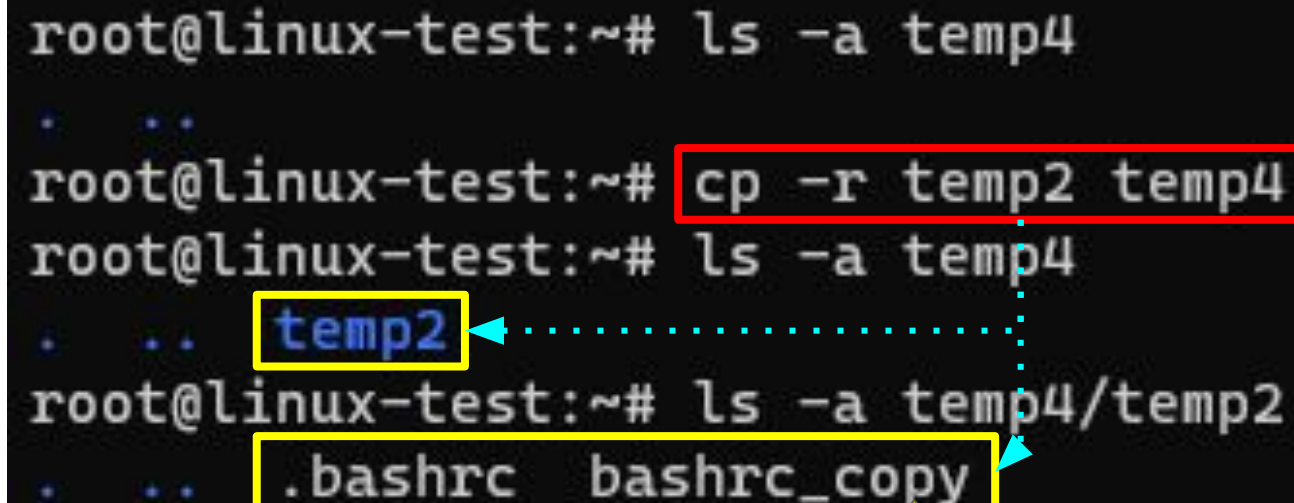
해당 경로에 같은 이름의 파일이 이미 존재하는 경우

```
root@linux-test:~# cp -i .bashrc temp2/bashrc_copy
cp: overwrite 'temp2/bashrc_copy'?
```

y : 덮어쓰기, n : 취소

5. temp2 디렉토리를 temp4 디렉토리에 복사 : `cp -r temp2 temp4`

```
root@linux-test:~# ls -a temp4
.  ..
root@linux-test:~# cp -r temp2 temp4
root@linux-test:~# ls -a temp4
.  ..  temp2
root@linux-test:~# ls -a temp4/temp2
.  ..  .bashrc  bashrc_copy
```



temp2 디렉토리가 temp4 안에 복사

파일 명령

- **mv** (MoVe)

- **기능** : 파일이나 디렉토리를 이동하거나 이름을 바꾼다
- **형식** : **mv** [옵션] [SOURCE 파일(디렉토리)]... [DEST 파일(디렉토리)]
무엇을 어디로

1. bashrc_copy 파일을 bashrc_mv 파일로 이동(이름을 바꿈) : **mv bashrc_copy bashrc_mv**

```
root@linux-test:~# ls
bashrc_copy temp2 temp3 temp4 temp5
root@linux-test:~# mv bashrc_copy bashrc_mv
root@linux-test:~# ls
bashrc_mv temp2 temp3 temp4 temp5
```


2. bashrc_mv 파일을 temp2 디렉토리로 이동 :

```
mv bashrc_mv temp2  
mv bashrc_mv /root/temp2  
mv bashrc_mv temp2/bashrc_moved  
mv bashrc_mv /root/temp2/bashrc_moved
```

이동하면서 파일명 변경

```
bashrc_mv temp2 temp3 temp4 temp5  
root@linux-test:~# mv bashrc_mv temp2/bashrc_moved  
root@linux-test:~# ls  
temp2 temp3 temp4 temp5  
root@linux-test:~# ls temp2  
bashrc_copy bashrc_moved
```

3. temp2/bashrc_copy, temp2/bashrc_moved 파일을 temp3 디렉토리로 이동 :

```
mv temp2/bashrc_copy temp2/bashrc_moved temp3
```

마지막으로 지정한 디렉토리로 이동

```
root@linux-test:~# ls temp2
bashrc_copy  bashrc_moved
root@linux-test:~# ls temp3
root@linux-test:~# mv temp2/bashrc_copy temp2/bashrc_moved temp3
root@linux-test:~# ls temp3
bashrc_copy  bashrc_moved
```

4. 이미 존재하는 파일인 경우 덮어쓸 것인지 질문 : `mv -i temp3/bashrc_copy temp3/bashrc_moved`

해당 경로에 같은 이름의 파일이
이미 존재하는 경우

```
root@linux-test:~# ls temp3
bashrc_copy  bashrc_moved
root@linux-test:~# mv -i temp3/bashrc_copy temp3/bashrc_moved
mv: overwrite 'temp3/bashrc_moved'?
```

y : 덮어쓰기, n : 취소

파일 명령

- **rm** (ReMove)
 - **기능** : 파일 또는 디렉토리를 삭제한다
 - **형식** : **rm** [옵션] [파일 또는 디렉토리]

파일 / 디렉토리를 삭제할 때는 항상 신중할 것!
root 계정에서는 더더욱!!!!
(리눅스에는 휴지통이 없다!!!!)

파일 명령

1. temp3/bashrc_moved 파일을 삭제 : **rm temp3/bashrc_moved**

```
root@linux-test:~# ls temp3
bashrc_copy bashrc_moved
root@linux-test:~# rm temp3/bashrc_moved
root@linux-test:~# ls temp3
bashrc_copy
```

bashrc_moved 파일 삭제됨

2. temp2 디렉토리를 삭제 : **rm -r temp2**
rm -rf temp2 (-f : 강제 삭제 옵션)

```
root@linux-test:~# ls
temp2 temp3 temp4 temp5
root@linux-test:~# rm -r temp2
root@linux-test:~# ls
temp3 temp4 temp5
```

temp2 디렉토리 삭제됨

파일 명령

● ln (LiNk)

- 기능 : 파일의 링크를 생성
- 형식 : ln [옵션] [원본 파일] [링크 파일]

1. /etc/hosts 파일에 대한 하드 링크 생성 : ln /etc/hosts host_link

원본 파일의 inode 번호 확인

```
root@linux-test:~# ls -il /etc/hosts
2883719 -rw-r--r-- 1 root root 189 Sep 28 2020 /etc/hosts
```

```
root@linux-test:~# ln /etc/hosts hosts_link
```

```
root@linux-test:~# ls -ail
total 60
```

하드 링크 생성

하드 링크의 inode 번호 확인
=> 원본과 동일

```
524289 drwx----- 11 root root 4096 Feb 16 11:02 .
  2 drwxr-xr-x 25 root root 4096 Oct 13 2020 ..
524323 -rw----- 1 root root 0 Feb 16 09:39 .bash_history
524290 -rw-r--r-- 1 root root 3106 Apr 9 2018 .bashrc
524292 drwx----- 2 root root 4096 Feb 9 11:14 .cache
524296 drwx----- 3 root root 4096 Oct 6 2020 .gnupg
2883719 -rw-r--r-- 2 root root 189 Sep 28 2020 hosts_link
524305 drwxr-xr-x 3 root root 4096 Oct 6 2020 .local
524303 drwxr-xr-x 2 root root 4096 Feb 16 09:31 .nsight
524291 -rw-r--r-- 1 root root 1024 Aug 18 2015 .profile
524301 -rw-r--r-- 1 root root 1024 Oct 6 2020 .selected_editor
524298 drwxr-xr-x 3 root root 4096 Oct 6 2020 .ssh
524295 drwxr-xr-x 2 root root 4096 Feb 16 10:50 temp3
524304 drwxr-xr-x 2 root root 4096 Feb 16 10:50 temp4
524308 drwxr-xr-x 2 root root 4096 Feb 16 10:50 temp5
524300 drwxr-xr-x 2 root root 4096 Oct 12 2020 .vim
```

하드 링크의 수 증가
=> 하드 링크를 지우면 감소

하드 링크와 복사의 차이

```
root@linux-test:~# cp /etc/hosts hosts_cp
root@linux-test:~# ls -ail
total 64
524289 drwx----- 11 root root 4096 Feb 16 11:11 .
2 drwxr-xr-x 25 root root 4096 Oct 13 2020 ..
-rw----- 1 root root 0 Feb 16 09:39 .bash_history
-rw-r--r-- 1 root root 3106 Apr 9 2018 .bashrc
2 drwx----- 2 root root 4096 Feb 9 11:14 .cache
524296 drwx----- 3 root root 4096 Oct 6 2020 .gnupg
524309 -rw-r--r-- 1 root root 189 Feb 16 11:11 hosts_cp
2883719 -rw-r--r-- 2 root root 189 Sep 28 2020 hosts_link
524305 drwxr-xr-x 3 root root 4096 Oct 6 2020 .local
524303 drwxr-xr-x 2 root root 4096 Feb 16 09:31 .nsight
-rw-r--r-- 1 root root 148 Aug 18 2015 .profile
-rw-r--r-- 1 root root 75 Oct 6 2020 .selected_editor
-rwx----- 2 root root 4096 Oct 6 2020 .ssh
524295 drwxr-xr-x 2 root root 4096 Feb 16 10:50 temp3
524304 drwxr-xr-x 2 root root 4096 Feb 16 10:50 temp4
524308 drwxr-xr-x 2 root root 4096 Feb 16 10:50 temp5
524300 drwxr-xr-x 2 root root 4096 Oct 12 2020 .vim
```

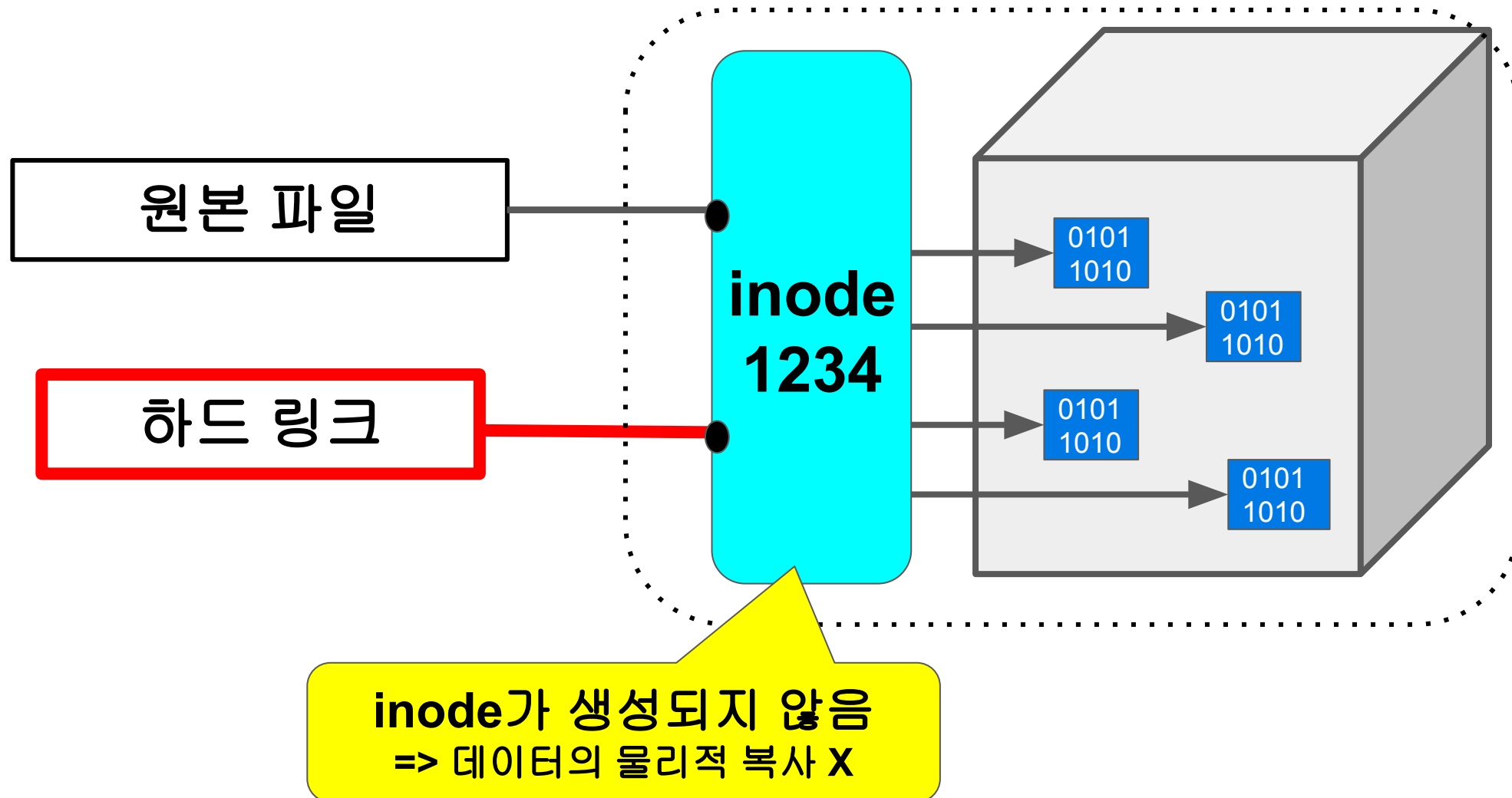
/etc/hosts 파일을 복사

복사한 파일의 inode
=> 원본과 다름

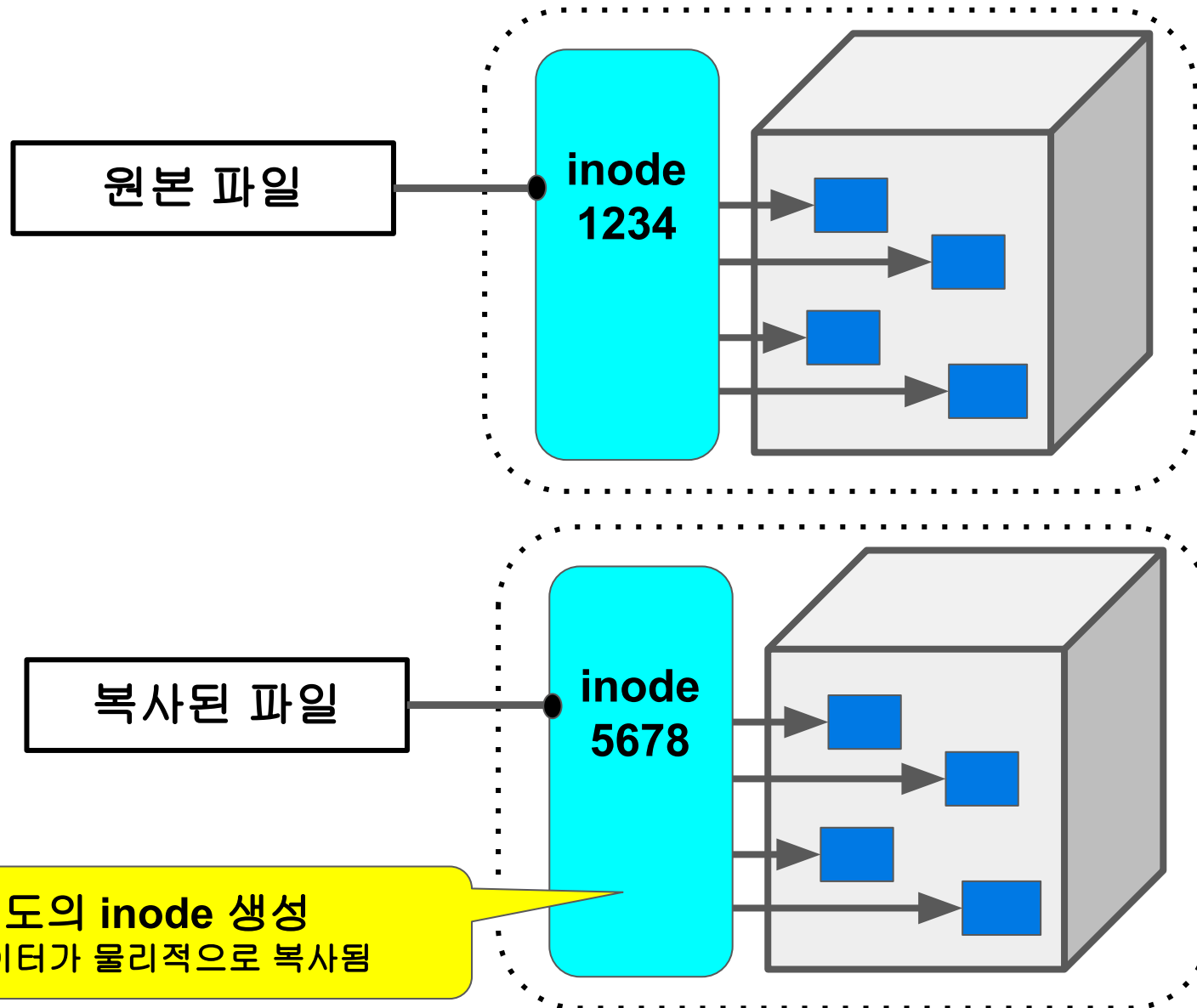
하드 링크의 inode
=> 원본과 같음

파일 명령

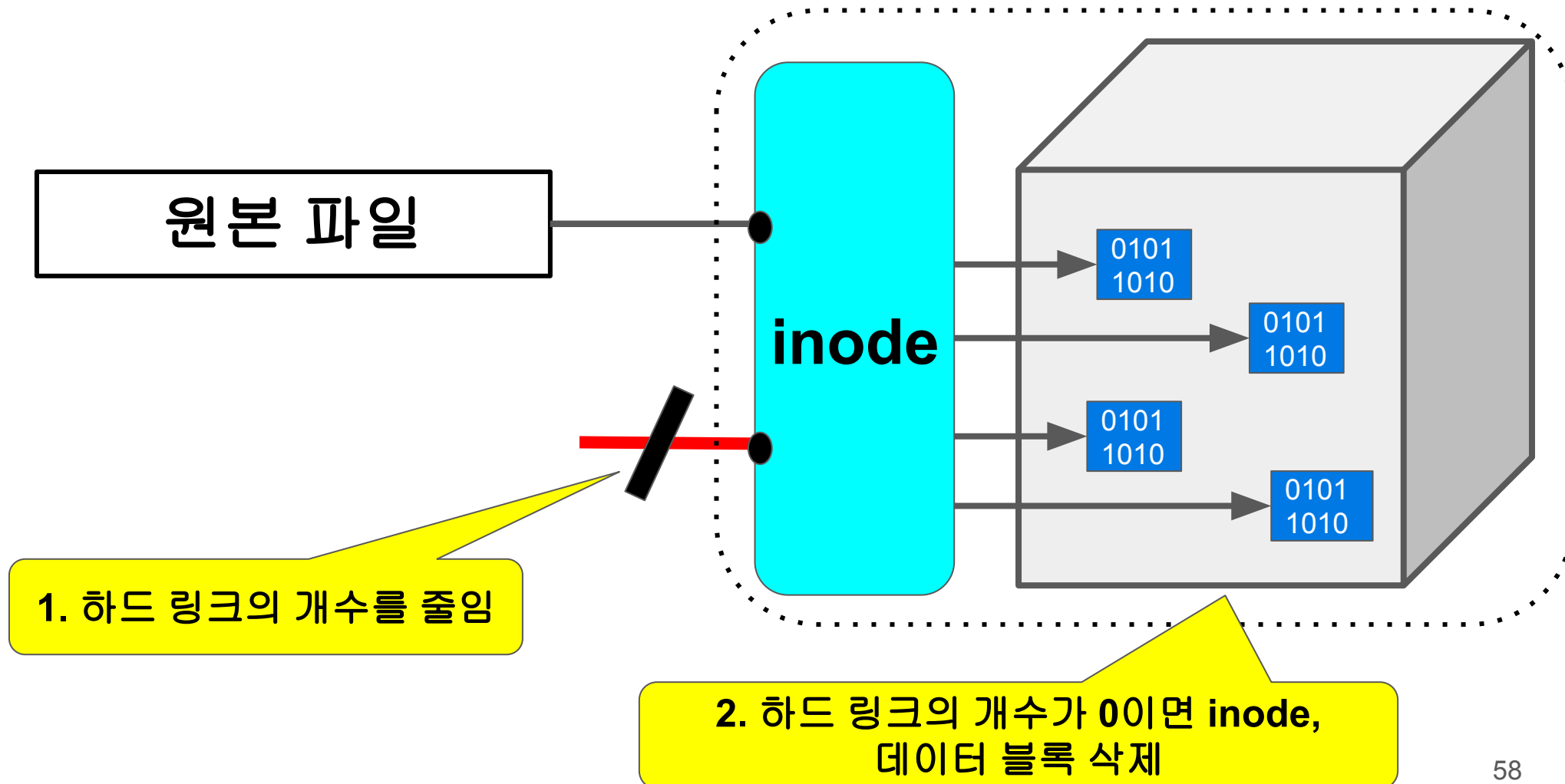
하드링크 : inode 값이 같다



파일 복사 : inode 값이 다르다



rm 명령의 의미



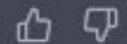
rm 명령의 의미 - ChatGPT한테 물어보면?



리눅스에서 어떤 파일의 하드링크가 여러 개일 경우 rm 명령을 실행하면 어떻게 되지?



리눅스에서 파일에 대한 하드링크는 동일한 파일 시스템 내에서 파일 이름에 대한 링크이며, 해당 파일의 inode를 참조합니다. 따라서 동일한 파일에 대한 하드링크가 여러 개 있으면 해당 파일의 데이터 블록은 여전히 유지되고, 파일에 대한 링크 수만 감소합니다. 실제로 파일이 삭제되고 디스크 공간이 회수되는 시점은 파일에 대한 모든 링크가 삭제된 후입니다.



그러므로, `rm` 명령은 파일의 링크 수를 감소시키고, 해당 파일에 대한 링크가 더 이상 없을 때 파일이 삭제됩니다. 즉, 동일한 파일에 대한 여러 개의 하드링크가 있어도, `rm` 명령이 해당 파일의 링크 수를 감소시키면 파일이 삭제되기 전까지는 여전히 다른 하드링크를 통해 파일에 접근할 수 있습니다.

따라서, 하드링크된 파일 중 하나를 `rm` 명령으로 삭제하면, 해당 파일 이름으로는 더 이상 접근할 수 없게 되지만, 다른 하드링크로는 여전히 파일에 접근할 수 있습니다. 그러나 모든 하드링크가 삭제되면 해당 파일이 완전히 삭제됩니다.

파일 명령

2. : /etc/sudoers 파일에 대한 심볼릭 링크(바로가기) 생성 : `ln -s /etc/sudoers sudoers_link`

원본 파일의 inode 번호 확인

```
root@linux-test:~# ls -il /etc/sudoers
2884027 -r--r----- 1 root root 755 Jan 18 2018 /etc/sudoers
root@linux-test:~# ln -s /etc/sudoers sudoers_link
root@linux-test:~# ls -ail
total 64
524289 drwx----- 11 root root 4096 Feb 17 09:57 .
  2 drwxr-xr-x 25 root root 4096 Oct 13 2020 ..
524311 -rw----- 1 root root 0 Feb 17 09:36 .bash_history
524290 -rw-r--r-- 1 root root 3106 Apr 9 2018 .bashrc
524292 drwx----- 2 root root 4096 Feb 9 11:14 .cache
524296 drwx----- 3 root root 4096 Oct 6 2020 .gnupg
524309 -rw-r--r-- 1 root root 189 Feb 16 11:11 hosts_cp
2883719 -rw-r--r-- 2 root root 189 Sep 28 2020 hosts_link
524305 drwxr-xr-x 3 root root 4096 Oct 6 2020 .local
524303 drwxr-xr-x 2 root root 4096 Feb 16 13:01 .nsight
524291 -rw-r--r-- 1 root root 148 Aug 18 2015 .profile
524301 -rw-r--r-- 1 root root 75 Oct 6 2020 .selected_editor
524298 drwx----- 2 root root 4096 Oct 6 2020 .ssh
524312 lrwxrwxrwx 1 root root 12 Feb 17 09:57 sudoers_link -> /etc/sudoers
524295 drwxr-xr-x 2 root root 4096 Feb 16 10:50 temp3
524304 drwxr-xr-x 2 root root 4096 Feb 16 10:50 temp4
524308 drwxr-xr-x 2 root root 4096 Feb 16 10:50 temp5
524300 drwxr-xr-x 2 root root 4096 Oct 12 2020 .vim
```

심볼릭 링크 생성

심볼릭 링크의 inode
=> 원본과 다름

원본 파일 정보

파일 명령

• touch

- **기능** : 빈 파일 만들기, 접근/수정 시간 변경
- **형식** : **touch** [옵션] [파일]

1. test라는 이름의 빈 파일 생성 : **touch test**

```
root@linux-test:~# ls
hosts_cp hosts_link sudoers_link temp3 temp4 temp5
root@linux-test:~# touch test
root@linux-test:~# ls -l
total 20
-rw-r--r-- 1 root root 189 Feb 16 11:11 hosts_cp
-rw-r--r-- 2 root root 189 Sep 28 2020 hosts_link
lrwxrwxrwx 1 root root 12 Feb 17 09:57 sudoers_link -> /etc/sudoers
drwxr-xr-x 2 root root 4096 Feb 16 10:50 temp3
drwxr-xr-x 2 root root 4096 Feb 16 10:50 temp4
drwxr-xr-x 2 root root 4096 Feb 16 10:50 temp5
-rw-r--r-- 1 root root 0 Feb 17 15:46 test
```

파일 사이즈 0 확인

2. test 파일의 수정 시간 변경 : **touch test**

```
root@linux-test:~# ls -l
total 20
-rw-r--r-- 1 root root 189 Feb 17 15:50 hosts_cp
-rw-r--r-- 2 root root 189 Sep 28 2020 hosts_link
lrwxrwxrwx 1 root root 12 Feb 17 09:57 sudoers_link -> /etc/sudoers
drwxr-xr-x 2 root root 4096 Feb 16 10:50 temp3
drwxr-xr-x 2 root root 4096 Feb 16 10:50 temp4
drwxr-xr-x 2 root root 4096 Feb 16 10:50 temp5
-rw-r--r-- 1 root root 0 Feb 17 15:46 test
root@linux-test:~# touch test
root@linux-test:~# ls -l
total 20
-rw-r--r-- 1 root root 189 Feb 17 15:50 hosts_cp
-rw-r--r-- 2 root root 189 Sep 28 2020 hosts_link
lrwxrwxrwx 1 root root 12 Feb 17 09:57 sudoers_link -> /etc/sudoers
drwxr-xr-x 2 root root 4096 Feb 16 10:50 temp3
drwxr-xr-x 2 root root 4096 Feb 16 10:50 temp4
drwxr-xr-x 2 root root 4096 Feb 16 10:50 temp5
-rw-r--r-- 1 root root 0 Feb 17 15:57 test
```

test 파일이 존재하는
상태에서 실행

수정 시간 변경 확인

• grep

- **기능** : 텍스트 파일에서 지정한 패턴이 포함된 행을 검색
- **형식** : **grep** [옵션] [패턴] [파일]

1. **.bashrc** 파일에서 'alias'라는 문자열이 포함된 행을 행 번호 포함하여 출력 : **grep -n "alias" .bashrc**

```
root@linux-test:~# grep -n "alias" .bashrc
68:# enable color support of ls and also add handy aliases
71:  alias ls='ls --color=auto'
72:  #alias dir='dir --color=auto'
73:  #alias vdir='vdir --color=auto'
75:  alias grep='grep --color=auto'
76:  alias fgrep='fgrep --color=auto'
77:  alias egrep='egrep --color=auto'
80:# some more ls aliases
81:alias ll='ls -aF'
82:alias la='ls -A'
83:alias l='ls -CF'
87:# ~/.bash_aliases, instead of adding them here directly.
90:if [ -f ~/.bash_aliases ]; then
91:  . ~/.bash_aliases
```

주요 옵션

- i : 대소문자 구분하지 않음
- w : 단어 단위로 검색
- e : 여러 문자열 검색
- n : 문자열이 검색된 행 번호 출력
- r : 하위 디렉토리까지 검색

파일 명령

• find

- **기능** : 지정한 위치에서 조건에 맞는 파일 및 디렉토리를 검색
- **형식** : **find** **[경로]** **[조건]** **[동작]**

1. **/etc** 디렉토리에서 **cron**이라는 이름의 파일을 검색하고 **긴 목록으로 출력** : **find /etc -name "cron" -ls**

```
root@linux-test:~# find /etc -name "cron" -ls
2883687      4 -rw-r--r--    1 root    root          150 Nov 16  2017 /etc/default/cron
2883723      4 -rwxr-xr-x    1 root    root        3049 Nov 16  2017 /etc/init.d/cron
2883832      4 -rw-r--r--    1 root    root          606 Nov 16  2017 /etc/pam.d/cron
```

2. **/etc** 디렉토리에서 **cron**이라는 이름의 파일을 검색하고 **inode 값을 출력** :
find /etc -name "cron" -exec ls -li {} \;

```
root@linux-test:~# find /etc -name "cron" -exec ls -li {} \;
2883687 /etc/default/cron
2883723 /etc/init.d/cron
2883832 /etc/pam.d/cron
```


find 명령 형식

경로

절대 경로, 상대 경로

```
find /etc -name "cron" -ls
```

검색 조건

-name 파일명 : 파일명 검색

-type 파일 종류 : 파일 종류로 검색

f(파일), d(디렉토리), l(심볼릭 링크) 등

-user 사용자 : 사용자가 소유한 파일 검색

-perm 권한 값 : 사용 권한이 일치하는 파일 검색

동작

-exec 명령 {} \; : 검색된 파일에 명령 실행

-ok 명령 {} \; : 사용자의 확인을 받아서 명령 실행

-ls : 검색 결과를 긴 목록으로 출력

파일 명령

• whereis

- **기능** : 지정된 이름의 바이너리 파일이나 매뉴얼 파일의 위치 검색
- **형식** : **whereis** [옵션] [파일]

1. ls 파일의 위치 검색 : **whereis ls**

```
root@linux-test:~# whereis ls
ls: /bin/ls /usr/share/man/man1/ls.1.gz
```

주요 옵션

- b : 바이너리 파일만 검색
- m : 매뉴얼 파일만 검색
- s : 소스 파일만 검색

• which

- **기능** : 명령(실행) 파일의 위치를 찾아서 경로 출력
- **형식** : **which** [파일]

```
root@linux-test:~# which ls
/bin/ls
```

- file

- **기능** : 파일의 종류 표시
- **형식** : **file** **[파일]**

1. .bashrc파일의 종류 표시 : **file .bashrc**

```
root@linux-test:~# file .bashrc  
.bashrc: ASCII text
```

.bashrc는 ASCII 텍스트 파일

2. /dev/stdin 파일의 종류 표시 : **file /dev/stdin**

```
root@linux-test:~# file /dev/stdin  
/dev/stdin: symbolic link to /proc/self/fd/0
```

/dev/stdin은 심볼릭 링크

Thank You