

목차

- A Shell과의 첫 만남
- B 디렉터리 구조 알아보기 (탐색하기, 생성/삭제하기)
- C 지금쯤 알아둬야 하는 것들 (command의 구조, man pages)
- D 파일과 입출력 다루기
- E 피곤하지만 효율적인 것들 (job, process, programming)
- F 자잘하지만 필요한 것들 (user, ssh, 환경 변수)
- G 몰라도 되지만 알면 편한 것들

Shell과의 첫 만남



Shell을 소개합니다



Shell은 조개 껌데기 - 무슨 의미일까



Shell은 조개 껍데기 - 뭔갈 감싸고 있다



그럼 “전산에서의 shell”은 뭘 감싸나?



컴퓨터의 속내 - 하드웨어



(쉬운 설명을 위해 과장되었음)

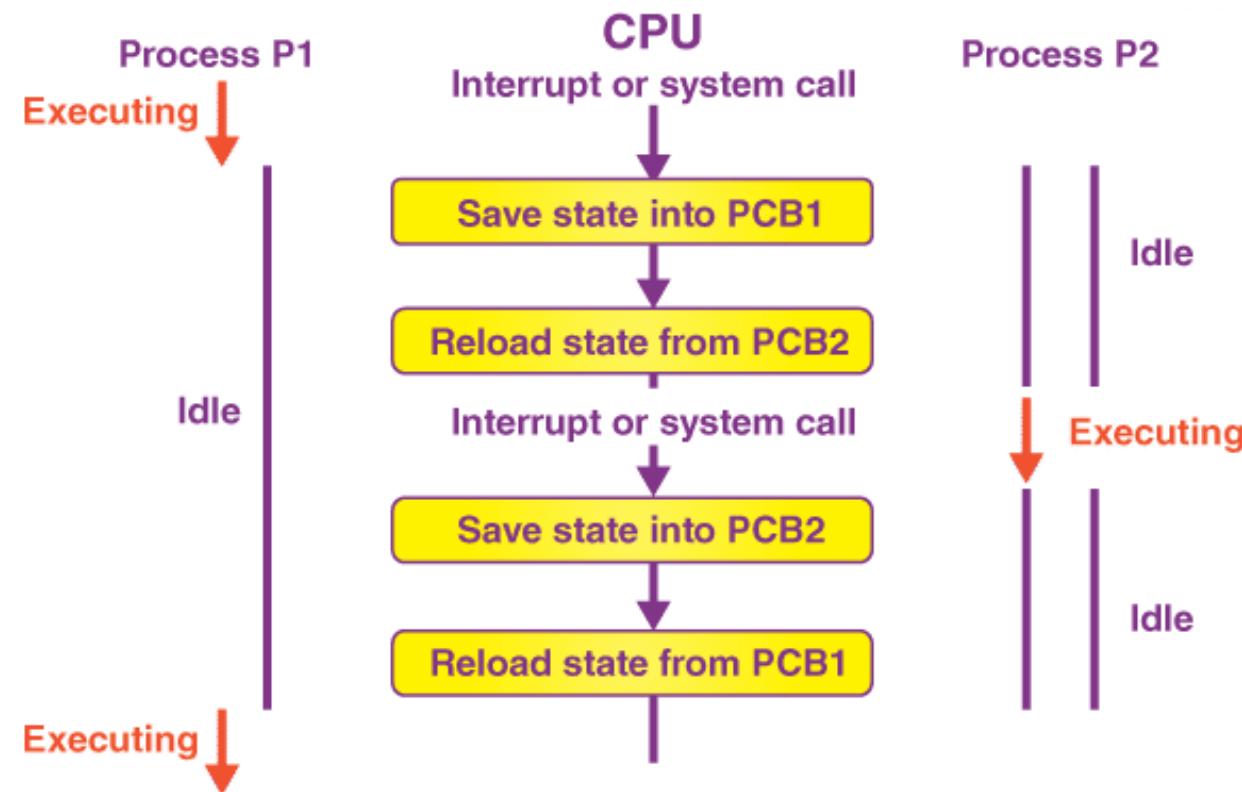
일을 시키면 합니다. 예시로…

데이터를 달라면 줍니다
입력을 받으라면 받습니다
기억하라면 기억합니다

그런데…
그 능력이 제한되어 있고
굉장히 단순한 능력들임

컴퓨터의 속내 - 예시: 멀티태스킹

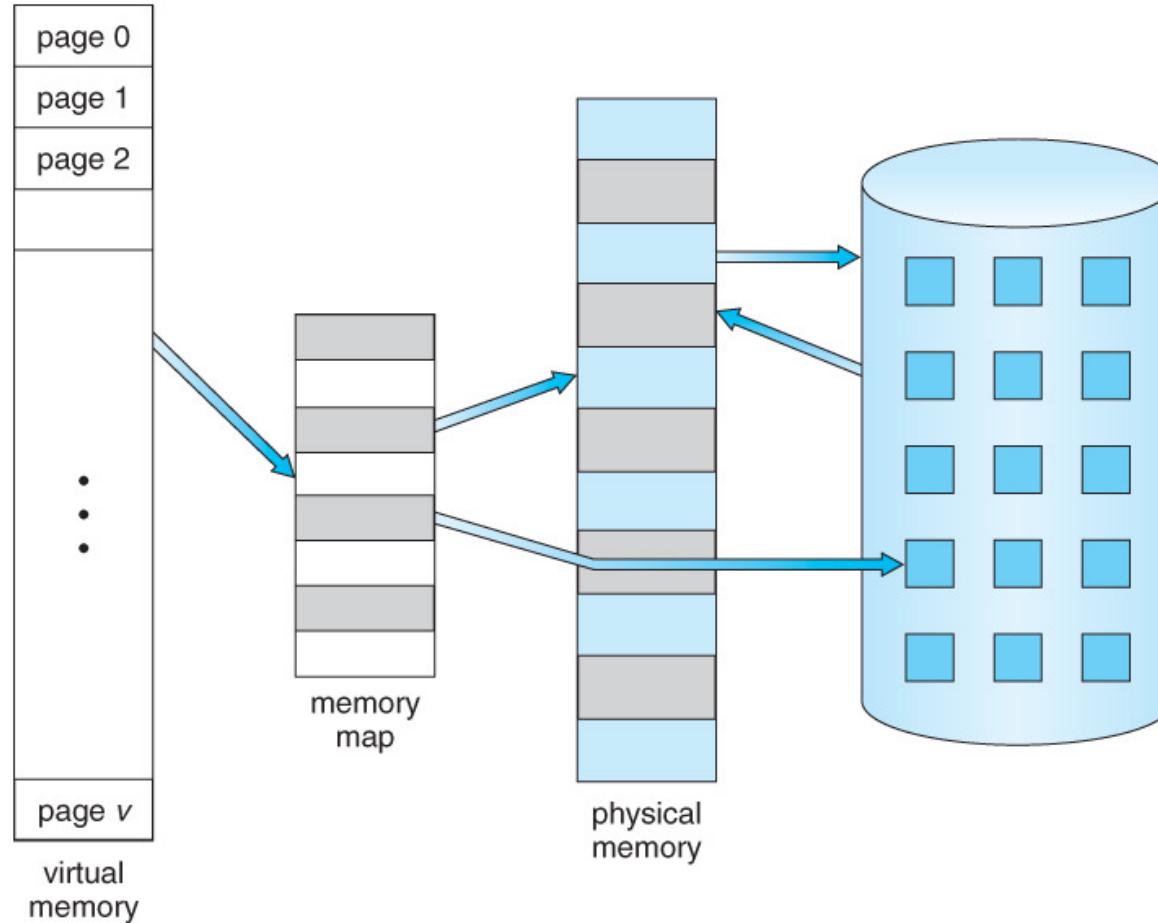
두 가지 일을 동시에 하고 싶다면?



- CPU야? 1번 일 해 줘.
(아주 잠깐 있다가)
- CPU야? 멈추고 2번 일 해 줘.
(아주 잠깐 있다가)
- CPU야? 멈추고 1번 일 해 줘.
(아주 잠깐 있다가)
- ...
- (아주 잠깐 있다가)
- CPU야? 멈추고 1번 일 해 줘.
(아주 잠깐 있다가)
- CPU야? 멈추고 2번 일 해 줘.
(아주 잠깐 있다가)
- ...

* 무조건 이렇지는 않습니다. 설명을 위한 예시임.

컴퓨터의 속내 - 예시: 메모리 사용



두 프로그램이 메모리를 쓴다면?

1번 프로그램: 메모리 4096번째 칸부터 4096칸 쓸게~
2번 프로그램: 메모리 4096번째 칸부터 4096칸 쓸게~
(파국)

그럼 어떻게 하느냐?

1번 프로그램: 메모리 4096번째 칸부터 4096칸 쓸게~
(1번 프로그램에게)
메모리의 8192번째 칸부터 12287번째 칸을
4096번째 칸부터 8191번째 칸이라고 속인다.)
1번 프로그램: 메모리 4096번째 칸부터 4096칸 쓸게~
(2번 프로그램에게)
메모리의 12288번째 칸부터 16383번째 칸을
4096번째 칸부터 8191번째 칸이라고 속인다.)

컴퓨터의 속내 - Kernel의 등장

두 가지 일을 동시에 하고 싶다면?

CPU야? 1번 일 해 줘.
(아주 잠깐 있다가)
CPU야? 멈추고 2번 일 해 줘.
(아주 잠깐 있다가)
CPU야? 멈추고 1번 일 해 줘.
(아주 잠깐 있다가)
...
(아주 잠깐 있다가)
CPU야? 멈추고 1번 일 해 줘.
(아주 잠깐 있다가)
CPU야? 멈추고 2번 일 해 줘.
(아주 잠깐 있다가)
...

* 무조건 이렇지는 않습니다. 설명을 위한 예시임.

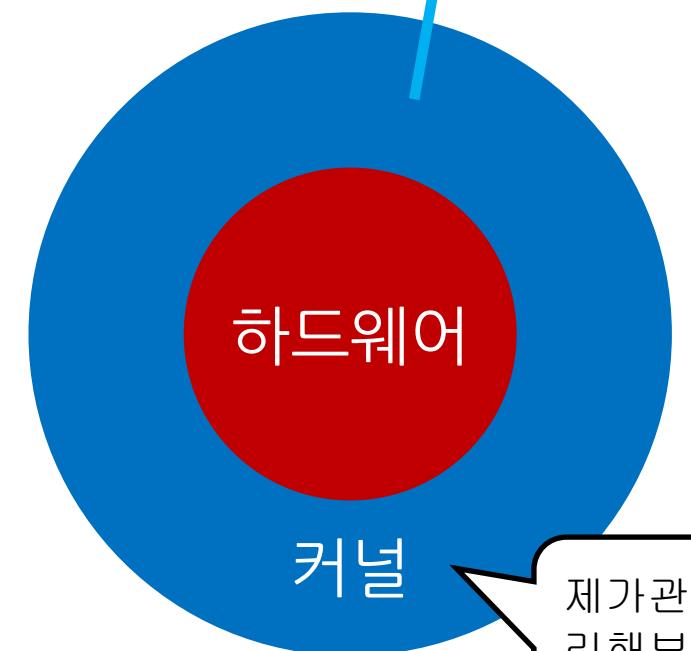
두 프로그램이 메모리를 쓴다면?

1번 프로그램: 메모리 4096번째 칸부터 4096칸 쓸게~
2번 프로그램: 메모리 4096번째 칸부터 4096칸 쓸게~
(파국)

그럼 어떻게 하느냐?

1번 프로그램: 메모리 4096번째 칸부터 4096칸 쓸게~
(1번 프로그램에게)
메모리의 8192번째 칸부터 12287번째 칸을
4096번째 칸부터 8191번째 칸이라고 속인다.)
1번 프로그램: 메모리 4096번째 칸부터 4096칸 쓸게~
(2번 프로그램에게)
메모리의 12288번째 칸부터 16383번째 칸을
4096번째 칸부터 8191번째 칸이라고 속인다.)

OS에서 과제로 일부 만들(었던) 것



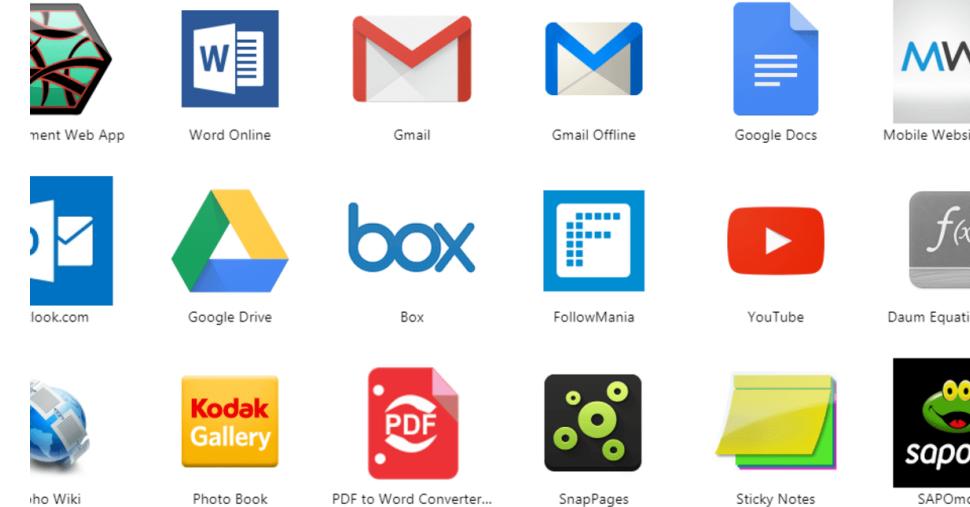
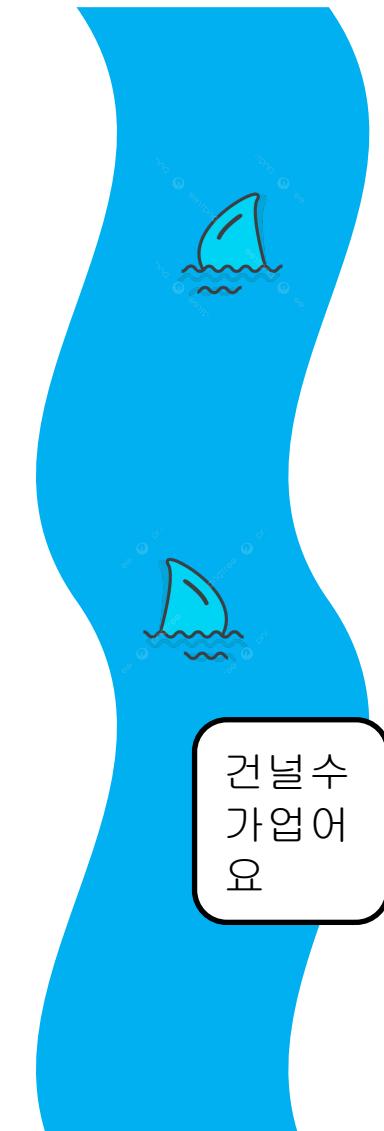
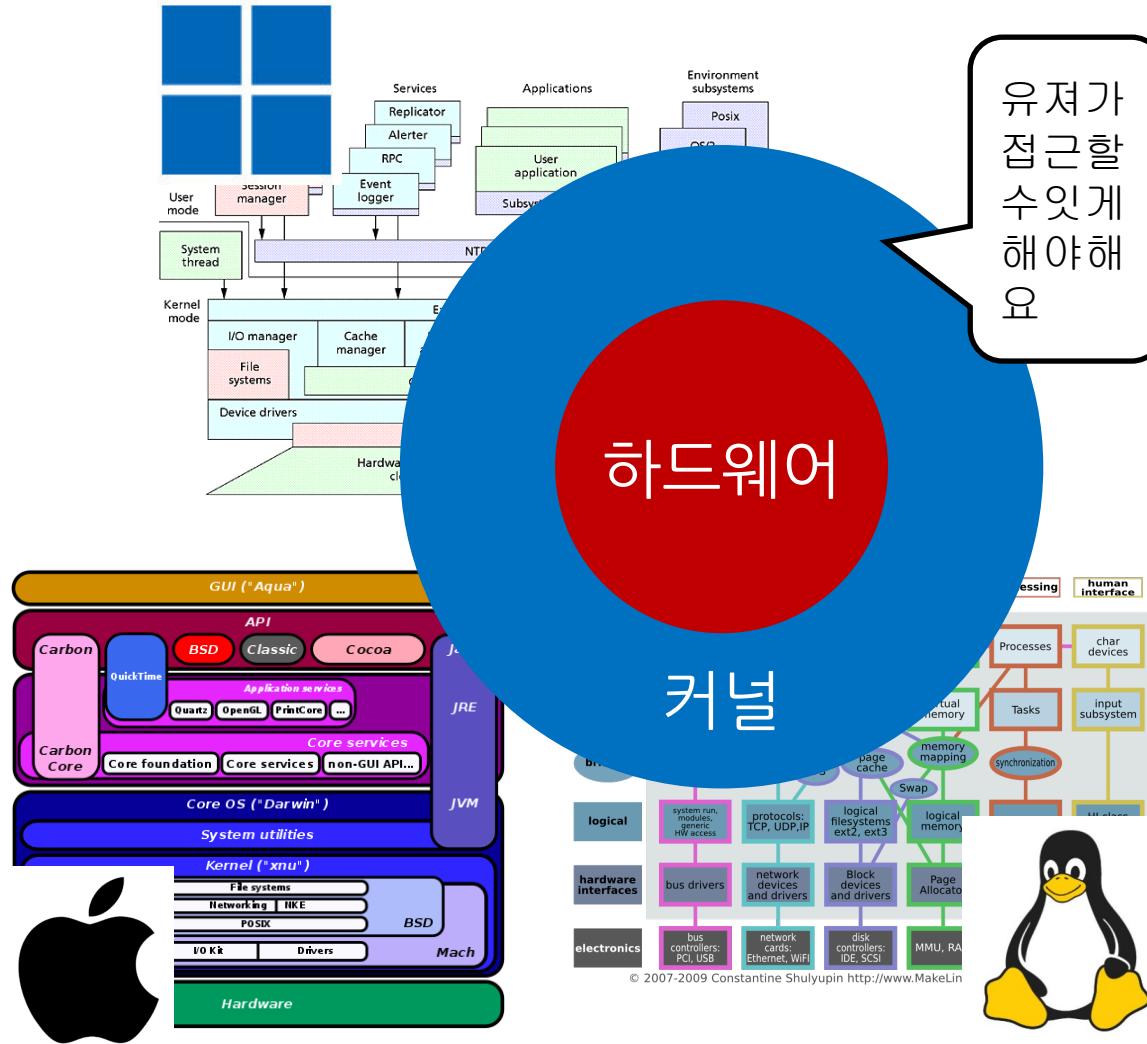
제가 관리해보겠습니다

누가 CPU한테 이렇게 명령해줄 건데?
(누가 프로세싱 자원을 관리할 건데?)

누가 이렇게 프로그램 속여줄 건데?
(누가 메모리 자원을 관리할 건데?)

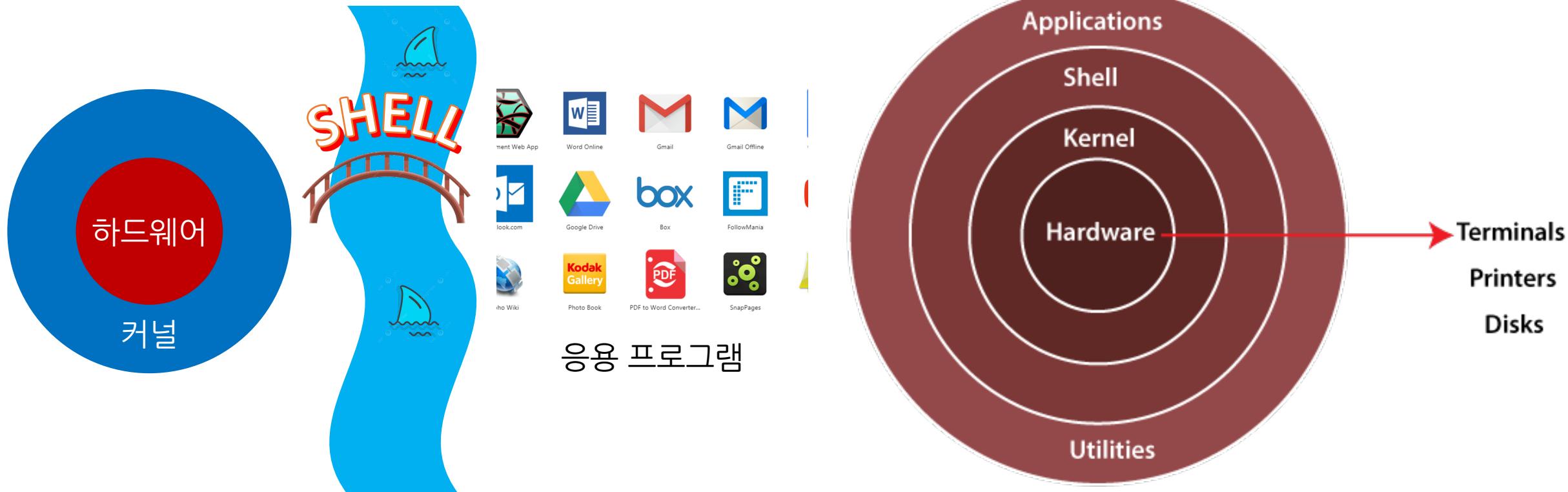


컴퓨터의 속내 - Kernel을 어떻게 써먹나



응용 프로그램

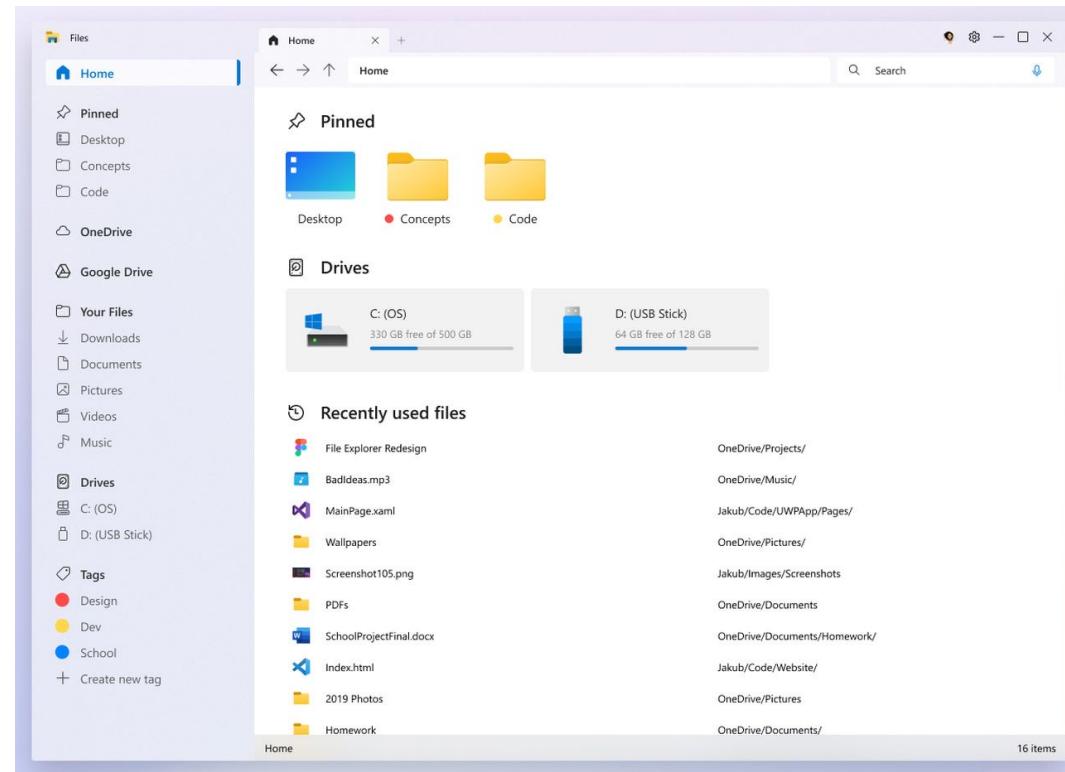
컴퓨터의 속내 - Shell의 등장



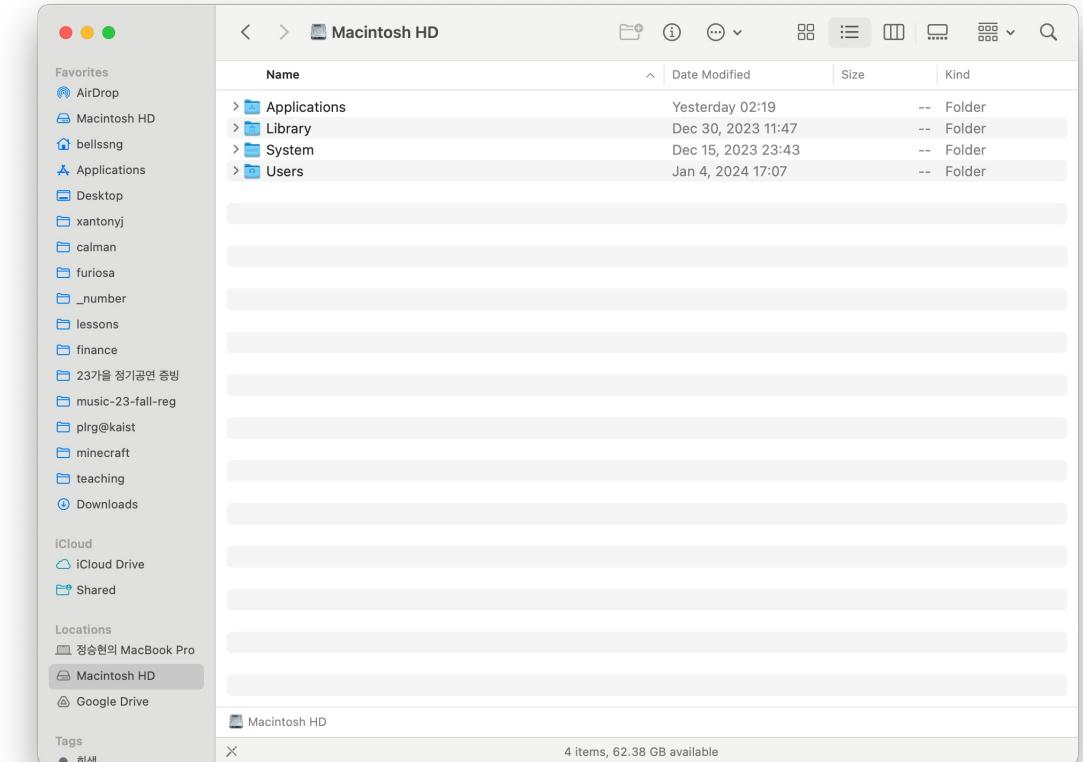
Shell은 kernel을 감싸고,
사용자는 shell을 통해 kernel의 기능을 이용함

Shell의 모습은? - GUI

“생각보다 우리는 shell과 친하다.”



Windows의 Explorer



MacOS의 Finder

Shell의 모습은? - CLI

“어… 친한가?”

Administrator: Command Prompt

```

ATTRIB Displays or changes file attributes.
BREAK Sets or clears extended CTRL+C checking.
BCDEDIT Sets properties in boot database to control boot loading.
CALCS Displays or modifies access control lists (ACLs) of files.
CALL Calls one batch program from another.
CD Displays the name of or changes the current directory.
CHCP Displays or sets the active code page number.
CHDIR Displays the name of or changes the current directory.
CHKDSK Checks a disk and displays a status report.
CHKNTFS Displays or modifies the checking of disk at boot time.
CLS Clears the screen.
CMD Starts a new instance of the Windows command interpreter.
COLOR Sets the default console foreground and background colors.
COMP Compares the contents of two files or sets of files.
COMPACT Displays or alters the compression of files on NTFS partitions.
CONVERT Converts file system between NTFS, FAT16, and FAT32.
COPY Copies files from one location to another.
DATE Displays or changes the date.
DEL Deletes files.
DIR Displays a list of files in a directory.
DISKPART Manages disk partitions.
DOSKEY Edits command history.
DRIVERQUERY Displays information about drivers.
ECHO Prints text to the screen.
ENDLOCAL Cancels local scope of a setlocal command.
ERASE Deletes files.
EXIT Exits a batch program.
  
```

Windows PowerShell

```

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\aseem> ■
  
```

Windows의 명령 프롬프트와 PowerShell

john — zsh — 82x24

```

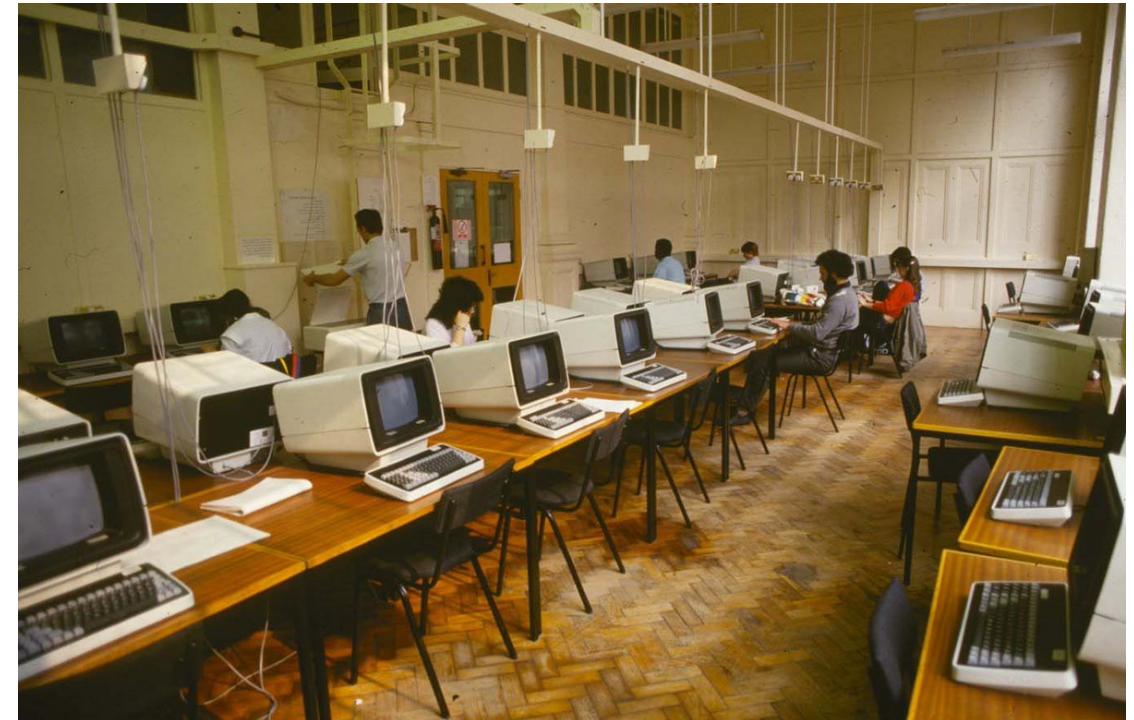
john@Johns-MacBook-Pro ~ % sudo /Applications/Install\ macOS\ Sonoma.app/Contents/Resources/createinstallmedia --volume /Volumes/MyVolume
Password:
Ready to start.
To continue we need to erase the volume at /Volumes/MyVolume.
If you wish to continue type (Y) then press return: Y
Erasing disk: 0%... 10%... 20%... 30%... 100%
Making disk bootable...
Copying to disk: 0%... 10%... 20%... 30%... 40%... 50%... 100%
Install media now available at "/Volumes/Install macOS Ventura"
john@Johns-MacBook-Pro ~ %
  
```

MacOS의 Terminal

Shell의 모습은? - 재밌는 옛날 이야기 1

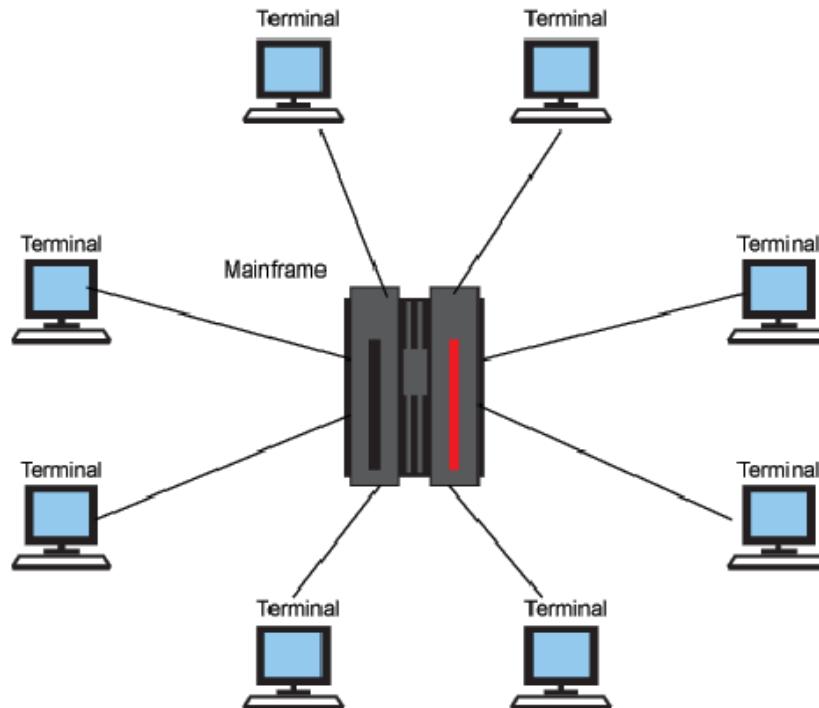


옛날 대학교 컴퓨터실의 모습



옛날 대학교 터미널실의 모습

Shell의 모습은? - 재밌는 옛날 이야기 2

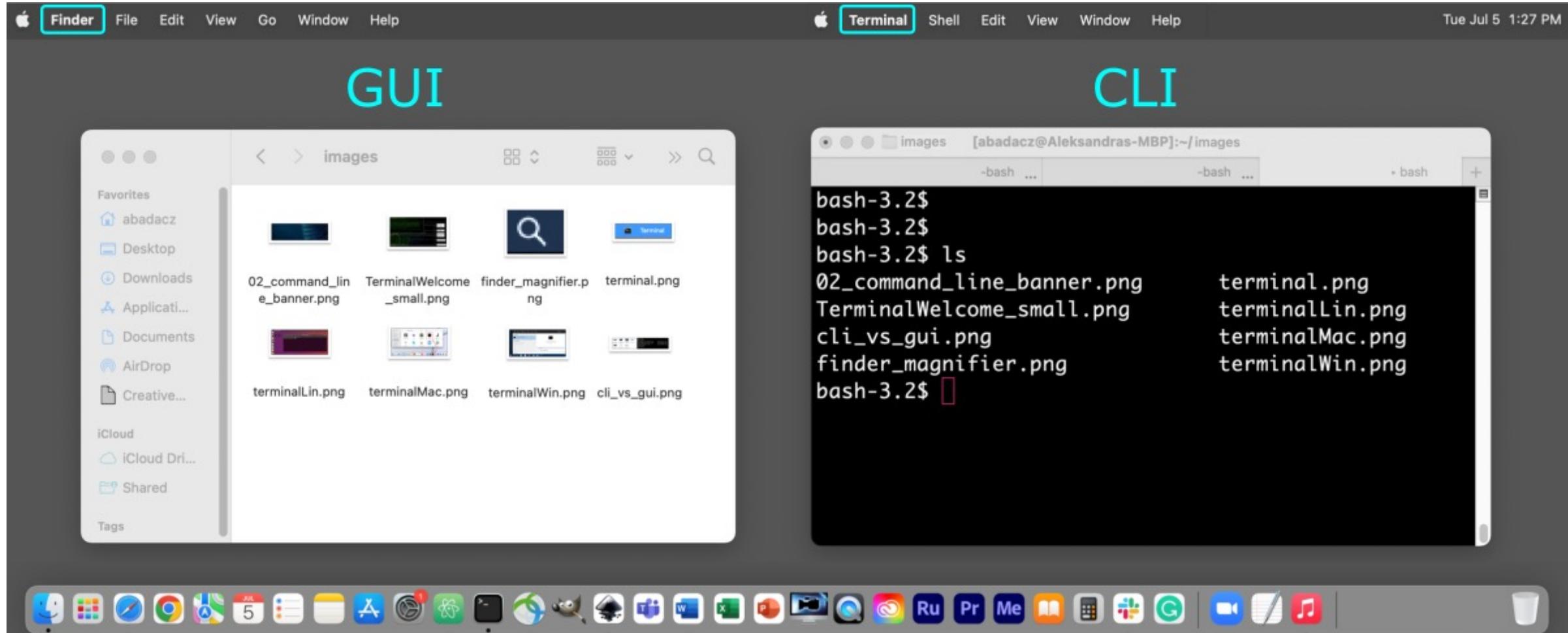


이런 구조였음



좀 나중에는 개인 PC가 나옴
(그래도 아직 CLI shell을 씀)

CLI shell을 익혀보자 . . . 도당체 왜?



CLI shell을 익혀보자 . . . 글쎄 좋다니까

필요하니까

언제나 GUI shell이 반겨주지 않습니다
코딩을 할 거라면 / 전산학부를 갈 거라면 해야 합니다

간지나니까

생각보다 중요하고 훌륭한 동기부여 방법

신속하니까

믿을 수 없겠지만...

간단하니까

믿을 수 없겠지만...

어떤 shell을 배워 볼까? - POSIX shell

- <POSIX> 시놉시스 -

UNIX라는 인기 좋은 OS가 있었다.

1990년대 초반,
여러 기업은 독자적인 UNIX를 만들었고,
사용자들은 다양하게 커스터마이즈된 UNIX들에서
원하는 프로그램을 실행하거나
다른 UNIX로 이전을 하기가 너무 힘들었다 흐규ㅠ

이 혼란을 잠재우기 위해
IEEE는 호환성과 이식성을 위한 표준을 만드는데…



Portable Operating System Interface for UNIX

POSIX

지금부터 C API는
이렇습니다
CeilCey

지금부터 shell은
요로코롬 만듭니다
sheelshee



어떤 shell을 배워 볼까? - bash



그리고 약간의
Linux 사용법

대부분의 UNIX-like OS에 기본으로 탑재되는 shell로,
POSIX에서 규정하는 shell 표준을 대부분 지키고 있다.

필요하니까

언제나 GUI shell이 반겨주지 않습니다

bash는 거의 항상 반겨 줌

코드를 할 거라면 / 전산학부를 갈 거라면 해야 합니다

실습 안내



실습 안내 - 접속 방법

Windows 사용자:

PowerShell을 켜 주세요.

MacOS 사용자:

Terminal을 켜 주세요

다음과 같이 입력합니다.

```
ssh -p <port번호> <user이름>@<host이름>
```

실습 안내 - 중요 공지사항

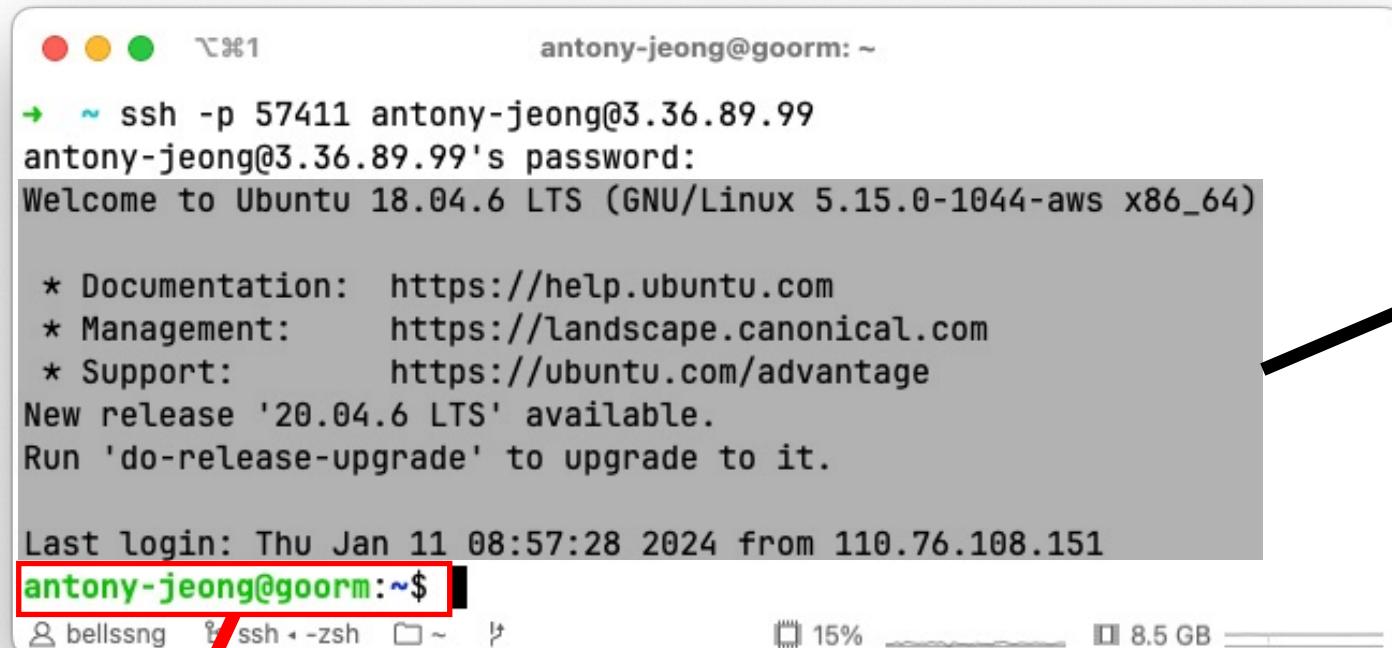
모두 함께 사용하는 공간입니다!

중간에 높은 권한을 한 번 줄 건데… 트롤링 ↴

bash 훈어보기



접속했을 때



```
antony-jeong@goorm: ~
→ ~ ssh -p 57411 antony-jeong@3.36.89.99
antony-jeong@3.36.89.99's password:
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.15.0-1044-aws x86_64)

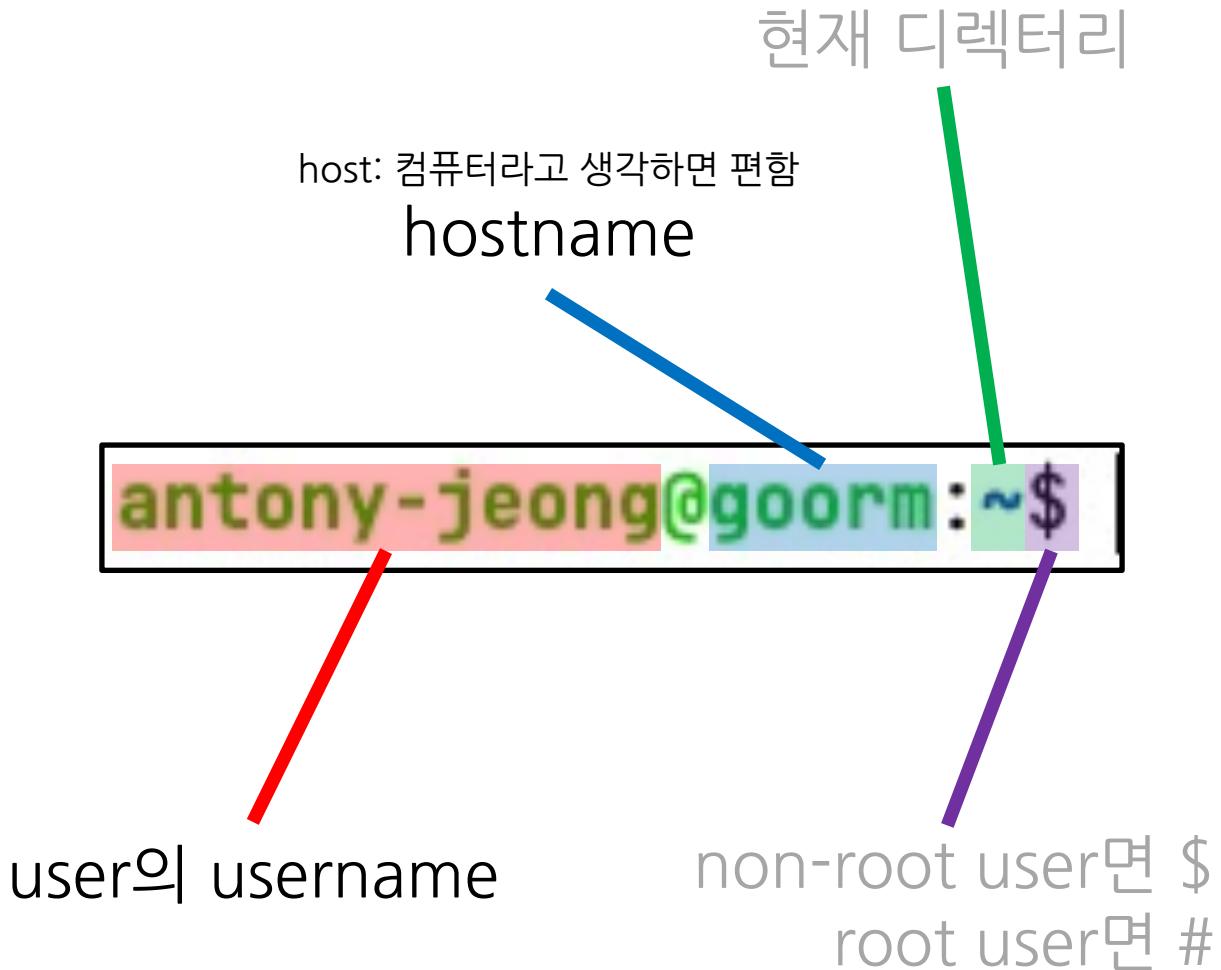
 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
New release '20.04.6 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Thu Jan 11 08:57:28 2024 from 110.76.108.151
antony-jeong@goorm:~$
```

접속했을 때 쓰는 메시지

prompt: 명령을 입력하기 전에 보이는 부분

prompt 해부하기



명령 입력해보기 - 나는 누구: whoami

```

antony-jeong@goorm:~$ whoami
antony-jeong
antony-jeong@goorm:~$ cat /etc/os-release
NAME="Ubuntu"
VERSION="18.04.6 LTS (Bionic Beaver)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 18.04.6 LTS"
VERSION_ID="18.04"
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
VERSION_CODENAME=bionic
UBUNTU_CODENAME=bionic
antony-jeong@goorm:~$ uname -m
x86_64
antony-jeong@goorm:~$ 

```

bellssng ssh -zsh ~ ↻

13% 8.4 GB

whoami

내가 누구? antony-jeong.

cat /etc/os-release

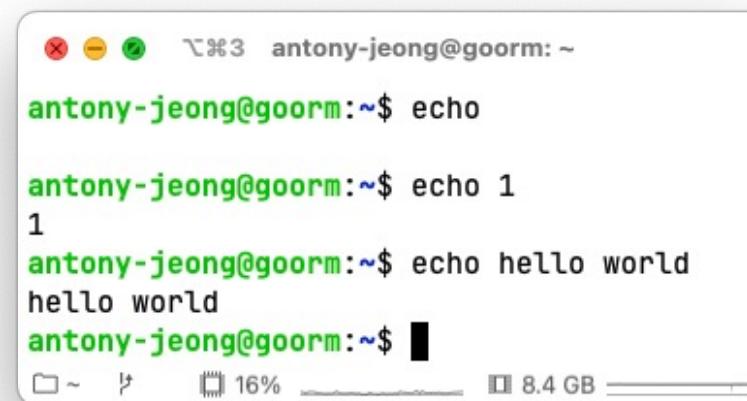
이 OS는 무엇인가

uname -m

이 기계의 architecture는 무엇인가

* 아주 나중에 어떤 이유로 입력해보게 될 것…

명령 입력해보기 - 날 따라해봐요: echo



The screenshot shows a terminal window with the following session:

```
antony-jeong@goorm:~$ echo  
antony-jeong@goorm:~$ echo 1  
1  
antony-jeong@goorm:~$ echo hello world  
hello world  
antony-jeong@goorm:~$
```

The terminal window has a title bar with three colored circles (red, yellow, green) and a status bar at the bottom showing icons for file, terminal, battery (16%), and storage (8.4 GB).

echo <뭐든>
shell이 따라함.

명령 입력해보기 - 재우기: sleep



The screenshot shows a macOS terminal window with the following content:

```
antony-jeong@goorm: ~
antony-jeong@goorm:~$ sleep 5
antony-jeong@goorm:~$ sleep 0.1
antony-jeong@goorm:~$ █
```

At the bottom of the terminal window, there is a status bar with icons for battery (18%), signal strength, and disk usage (8.3 GB).

sleep <#seconds>

지정한 시간만큼 아무 것도 안 함.

도중에 **ctrl+C**로 중단 가능함. SIGINT

^ 이거 진짜 진짜 진짜 꿀팁임.

명령 입력해보기 - 비우기: clear



A screenshot of a terminal window. At the top, there are three colored dots (red, yellow, green) followed by the text '��%3 antony-jeong@goorm: ~'. Below this, the prompt 'antony-jeong@goorm:~\$' is followed by a black terminal cursor. At the bottom of the window, there are standard Mac OS X window controls (close, minimize, maximize), a battery icon showing '12%', and a disk usage icon showing '8.3 GB'. The terminal window has a light gray background and a white foreground for the text.

```
��%3 antony-jeong@goorm: ~
antony-jeong@goorm:~$
```

clear

터미널이 비워짐. 깔끔!

← 입력한 후 화면임

이 PPT 만들 때 제일 많이 씀.

명령 입력해보기 - 비밀번호 변경: passwd

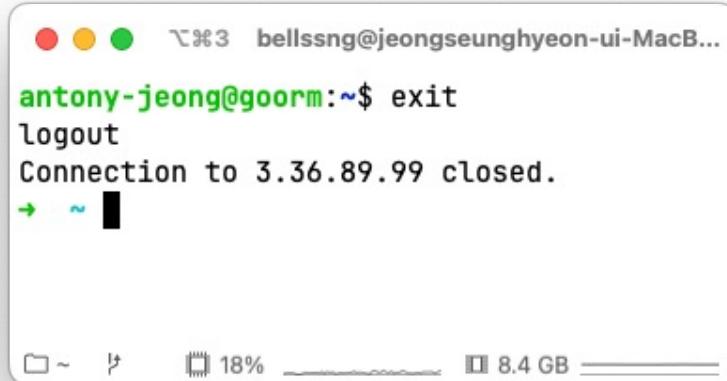
passwd

라고 입력 후,

shell이 지시하는 대로 수행해
비밀번호를 바꿔 봅시다.

(지금까지 나온 것들 중에 가장 UI스러운 CLI tool을 이용해 보았습니다)

명령 입력해보기 - 나가기: exit



A screenshot of a macOS terminal window. The title bar shows the user's name and the host as 'bellssng@jeongseunghyeon-ui-MacB...'. The terminal window contains the following text:
antony-jeong@goorm:~\$ exit
logout
Connection to 3.36.89.99 closed.
→ ~ █

The status bar at the bottom shows battery level at 18% and disk space usage at 8.4 GB.

exit

그 shell을 나감.

(ssh 명령으로 들어왔으면 ssh를 썼던 로컬 shell로 돌아옴.)

bash는 아니지만 아마 로컬 shell도 exit 하면 나가질 걸?

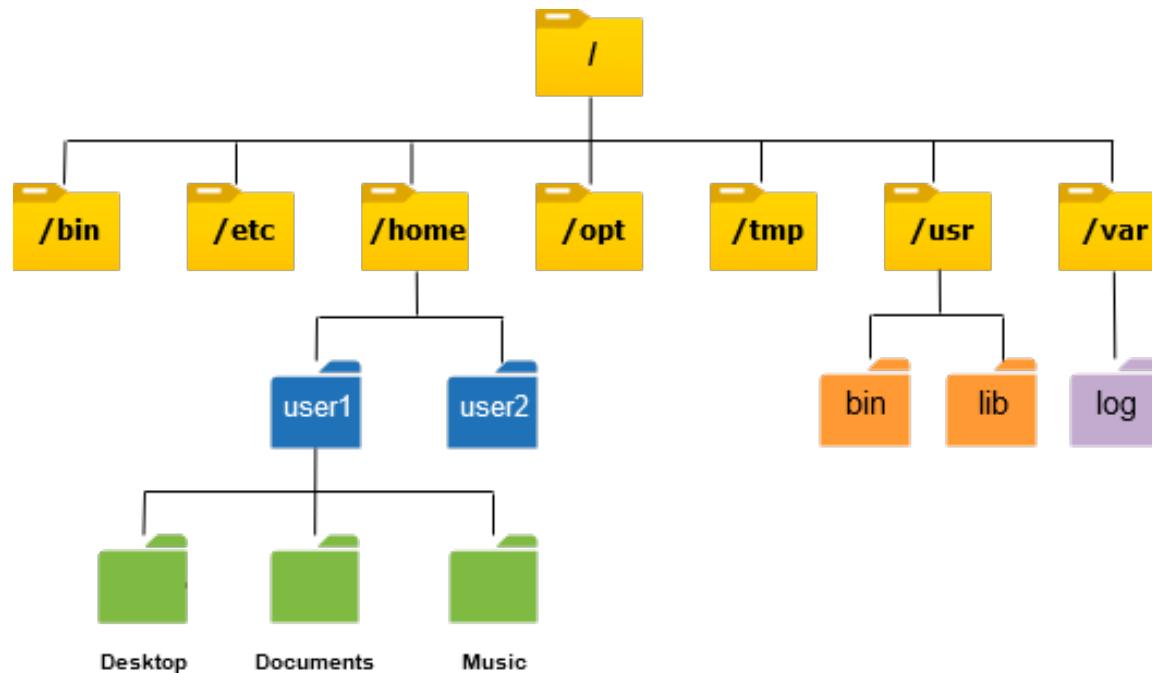
명령 입력해보기 - 다시 들어가보자

다시 접속해봅시다~

디렉터리 구조 알아보기

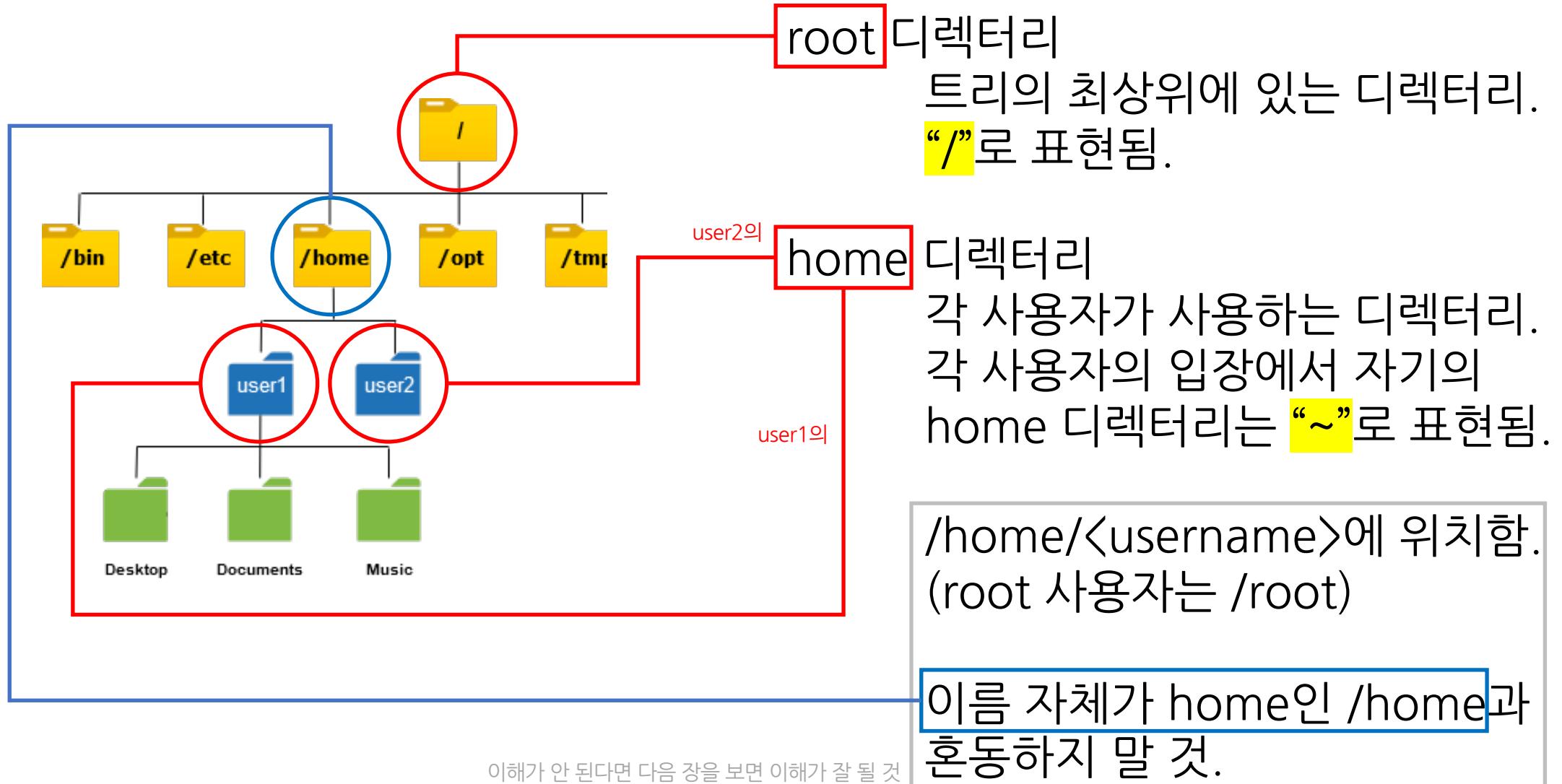


Linux의 디렉터리 구조 - 트리 구조

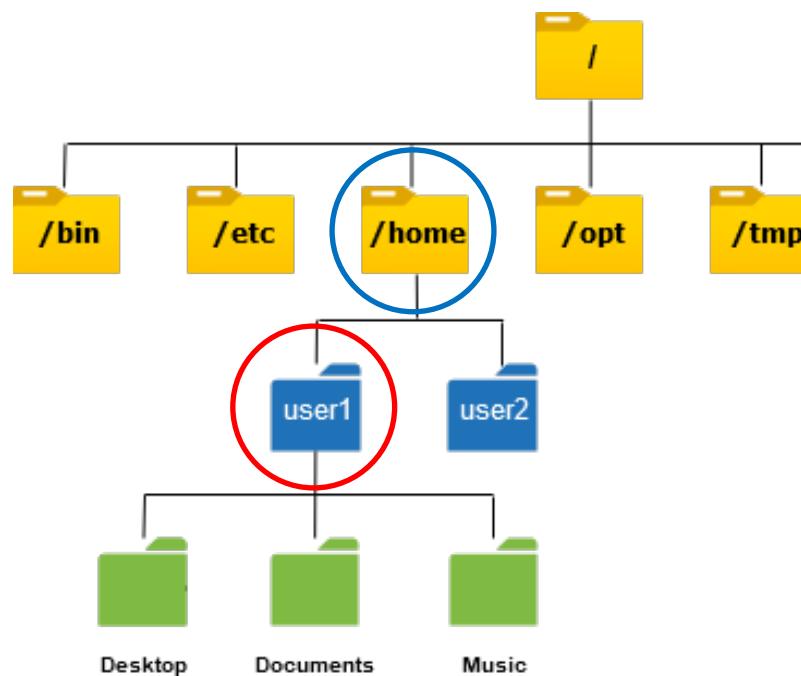


Windows, MacOS 모두 동일 → 우리에게 익숙한 형태

Linux의 디렉터리 구조 - root와 home



Linux의 디렉터리 구조 - “.”과 “..”



현재 디렉터리(current directory)
“.”로 표현됨.

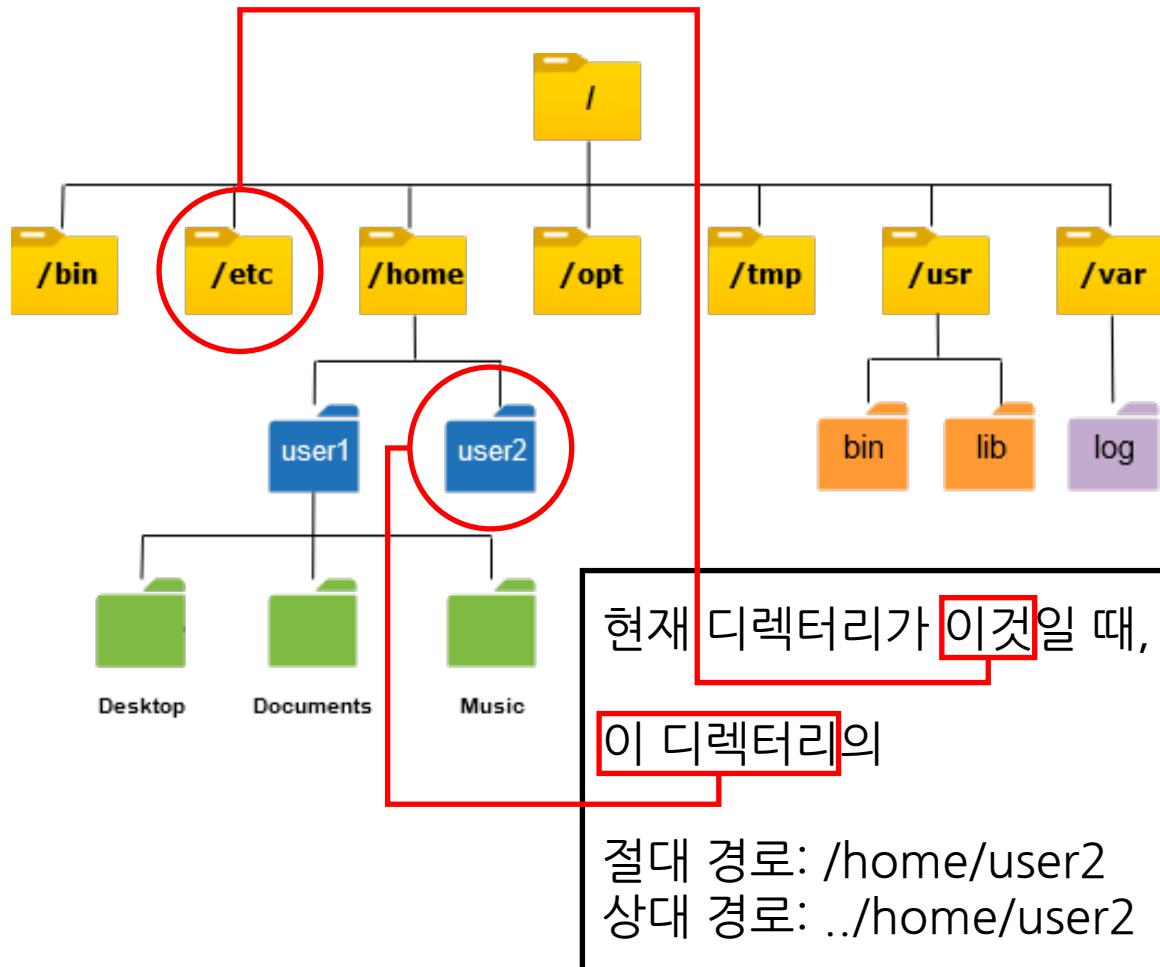
user1의 home을 기준으로 user1의 home은?
“.”이나 “././.” 등

부모 디렉터리(parent directory)
“..”로 표현됨.

user1의 home을 기준으로 root directory는?
“../../”이나 “../user1/../../” 등

* 상대적인 개념임

Linux의 디렉터리 구조 - path



경로(path)

목적지까지 가면서 경유한 모든 디렉터리와 목적지를 순서대로 “/”로 연결한 문자열.

절대 경로(absolute path)

root(/)로부터 출발한 경로.
“/”로 시작함.

상대 경로(relative path)

현재 디렉터리로부터 출발한 경로.
“/”로 시작하지 않음.

여기는 어디: pwd

print working directory



The screenshot shows a macOS terminal window with a light gray background. At the top, there are three colored dots (red, yellow, green) followed by the text 'Terminal' and the user information 'antony-jeong@goorm: ~'. Below this, the command 'pwd' is entered and its output '/home/antony-jeong' is displayed. The prompt 'antony-jeong@goorm:~\$' is shown again at the end. At the bottom of the window, there is a toolbar with icons for home, recent documents, and system status, including battery level (12%) and disk space (9.4 GB).

```
● ● ● Terminal antony-jeong@goorm: ~  
antony-jeong@goorm:~$ pwd  
/home/antony-jeong  
antony-jeong@goorm:~$ █
```

현재 디렉터리의 절대 경로가 출력됨

살펴보기: ls - 사용법 1

list segments

현재 디렉터리의
모든 항목(파일, 디렉터리, 링크)의
이름을 보여줌.

all
“-a”를 붙이면 숨김 파일도 보여줌.
(Linux에서 숨김파일은 “.”로 시작함.)

long
“-l”을 붙이면 자세한 정보를 알려줌.

“-a”와 “-l”은 “-al”(이나 “-la”)로 둘 다 적용할 수 있음.
(사실 “-a”와 “-l”을 따로따로 적어도 됨.)

살펴보기: ls - 사용법2

list segments

```

● ● ● ۮ۰۱      antony-jeong@goorm: ~

antony-jeong@goorm:~$ ls -la /home
total 48
drwxr-xr-x 12 root      root      4096 Jan  9  01:12 .
drwxr-xr-x  83 root      root      4096 Jan  9  01:17 ..
drwxr-xr-x   5 antony-jeong antony-jeong 4096 Jan 10 23:22 antony-jeong
drwxr-xr-x   3 cheavvn    cheavvn    4096 Jan  9 13:19 cheavvn
drwxr-xr-x   2 chiakpeach chiakpeach 4096 Jan  9  01:08 chiakpeach
drwxr-xr-x   3 derick321  derick321  4096 Jan  9  01:14 derick321
drwxr-xr-x   2 gyuch-an02 gyuch-an02  4096 Jan  9  01:10 gyuch-an02
drwxr-xr-x   2 jhw20050321 jhw20050321 4096 Jan  9  01:11 jhw20050321
drwxr-xr-x   2 listenjae  listenjae   4096 Jan  9  01:11 listenjae
drwxr-xr-x   2 p-has-done p-has-done   4096 Jan  9  01:12 p-has-done
drwxr-xr-x   2 tory453    tory453    4096 Jan  9  01:11 tory453
drwxr-xr-x   4 yeonsuh    yeonsuh    4096 Jan  9 13:19 yeonsuh
antony-jeong@goorm:~$ ls ..
antony-jeong  chiakpeach  gyuch-an02  listenjae  tory453
cheavvn       derick321   jhw20050321  p-has-done  yeonsuh
antony-jeong@goorm:~$ ls ~
number_1.mp3  receipts    synopsis.txt
antony-jeong@goorm:~$ ls ~/..
antony-jeong  chiakpeach  gyuch-an02  listenjae  tory453
cheavvn       derick321   jhw20050321  p-has-done  yeonsuh
antony-jeong@goorm:~$ 
```

bellssng ۵ ssh -zsh ۷ ~ ۸ ۰ 14% ۰ 9.5 GB

명령어 뒤에 경로를 써 보자!

절대 경로와 상대 경로 모두 가능
“~” 또한 사용할 수 있다!

경로의 목적지가 디렉터리면?

그 디렉터리에 있는 모든 항목에 대해 알려줌.
절대 경로와 상대 경로 모두 가능
“~”도 사용할 수 있다!

경로의 목적지가 디렉터리가 아니면?

그 항목에 대해서만 알려줌.

살펴보기: ls - 읽는 법

list segments

```

● ○ ● ⌂ 1
antony-jeong@goorm: ~

antony-jeong@goorm:~$ ls -la
total 36
drwxr-xr-x  5 antony-jeong antony-jeong 4096 Jan 10 23:22 .
drwxr-xr-x 12 root        root        4096 Jan  9  01:12 ..
-rw-----  1 antony-jeong antony-jeong   757 Jan 10 23:20 .bash_history
-rw-r--r--  1 antony-jeong antony-jeong   220 Jan  7 17:05 .bash_logout
-rw-r--r--  1 antony-jeong antony-jeong 3771 Jan  7 17:05 .bashrc
drwx----- 2 antony-jeong antony-jeong 4096 Jan  7 17:06 .cache
drwx----- 3 antony-jeong antony-jeong 4096 Jan  9 12:13 .config
-rw-r--r--  1 antony-jeong antony-jeong  807 Jan  7 17:05 .profile
-rw-rw-r--  1 antony-jeong antony-jeong     0 Jan 10 23:08 number_1.mp3
drwxrwxr-x  2 antony-jeong antony-jeong 4096 Jan 10 23:17 receipts
-rw-rw-r--  1 antony-jeong antony-jeong     0 Jan 10 23:08 synopsis.txt

antony-jeong@goorm: $ █
  ↗ bellssng  ↗ ssh -zsh  ↗ ~  ↗

```

d: 디렉터리

-: 파일

ℓ: 링크

권한

이 항목의 주인

이 항목의 그룹

용량

이름

최종 수정 시각

귀찮을 땐 tab: autocomplete

antony-jeong@goorm:~\$ ls -la n█

TAB →

antony-jeong@goorm:~\$ ls -la number_1.mp3 █

Return ←

antony-jeong@goorm:~\$ ls -la number_1.mp3
-rw-rw-r-- 1 antony-jeong antony-jeong 0 Jan 10 23:08 number_1.mp3

antony-jeong@goorm:~\$ ls -la .█

TAB →

antony-jeong@goorm:~\$ ls -la .█

TAB →

antony-jeong@goorm:~\$ ls -la .
./ .bash_history .bashrc .config/
../.bash_logout .cache/ .profile

TAB을 통해 자동완성을 할 수 있음

“이 다음에 입력할 건 이것 밖에 없는데?”
가 통하는 선까지만 딱 자동완성해줌

후보가 여러 개면 반응을 안 함
한 번 더 누르면 후보를 보여줌

움직이기: cd

change directory

```
antony-jeong@goorm:~$ pwd
/home/antony-jeong
antony-jeong@goorm:~$ ls
number_1.mp3 receipts synopsis.txt
antony-jeong@goorm:~$ cd receipts/
antony-jeong@goorm:~/receipts$ pwd
/home/antony-jeong/receipts
antony-jeong@goorm:~/receipts$ cd ~
antony-jeong@goorm:~$ pwd
/home/antony-jeong
antony-jeong@goorm:~$ cd ..
antony-jeong@goorm:/home$ pwd
/home
antony-jeong@goorm:/home$ cd /
antony-jeong@goorm:$ pwd
/
antony-jeong@goorm:$ cd ~/receipts/
antony-jeong@goorm:~/receipts$ pwd
/home/antony-jeong/receipts
antony-jeong@goorm:~/receipts$ cd ../../..
antony-jeong@goorm:/home$ pwd
/home
antony-jeong@goorm:/home$
```

bellssng ssh -zsh ~ ↻

16% 9.4 GB

cd 뒤에 디렉터리의 경로를 쓰면
그 디렉터리로 이동함

```
antony-jeong@goorm:~$ cd ../../..
antony-jeong@goorm:~$ pwd
/home/antony-jeong
antony-jeong@goorm:~$
```

bellssng ssh -zsh ~ ↻

15% 9.6 GB

make directory

생성과 삭제 - mkdir

```
antony-jeong@goorm:~/music_team
antony-jeong@goorm:~$ mkdir music_team
antony-jeong@goorm:~$ ls -l
total 8
drwxrwxr-x 2 antony-jeong antony-jeong 4096 Jan 11 00:06 music_team
-rw-rw-r-- 1 antony-jeong antony-jeong     0 Jan 10 23:08 number_1.mp3
drwxrwxr-x 2 antony-jeong antony-jeong 4096 Jan 10 23:17 receipts
-rw-rw-r-- 1 antony-jeong antony-jeong     0 Jan 10 23:08 synopsis.txt
antony-jeong@goorm:~$ cd music_team
antony-jeong@goorm:~/music_team$ ls -a
.
..
antony-jeong@goorm:~/music_team$ ■
● ○ ●  ☰ 1% antony-jeong@goorm:~/music_team
antony-jeong@goorm:~/music_team$ mkdir number_1 number_2 number_3
antony-jeong@goorm:~/music_team$ ls -l
total 12
drwxrwxr-x 2 antony-jeong antony-jeong 4096 Jan 11 00:12 number_1
drwxrwxr-x 2 antony-jeong antony-jeong 4096 Jan 11 00:12 number_2
drwxrwxr-x 2 antony-jeong antony-jeong 4096 Jan 11 00:12 number_3
antony-jeong@goorm:~/music_team$ ■
贝尔斯宁 1g ssh -zsh 15% 8.5 GB
```

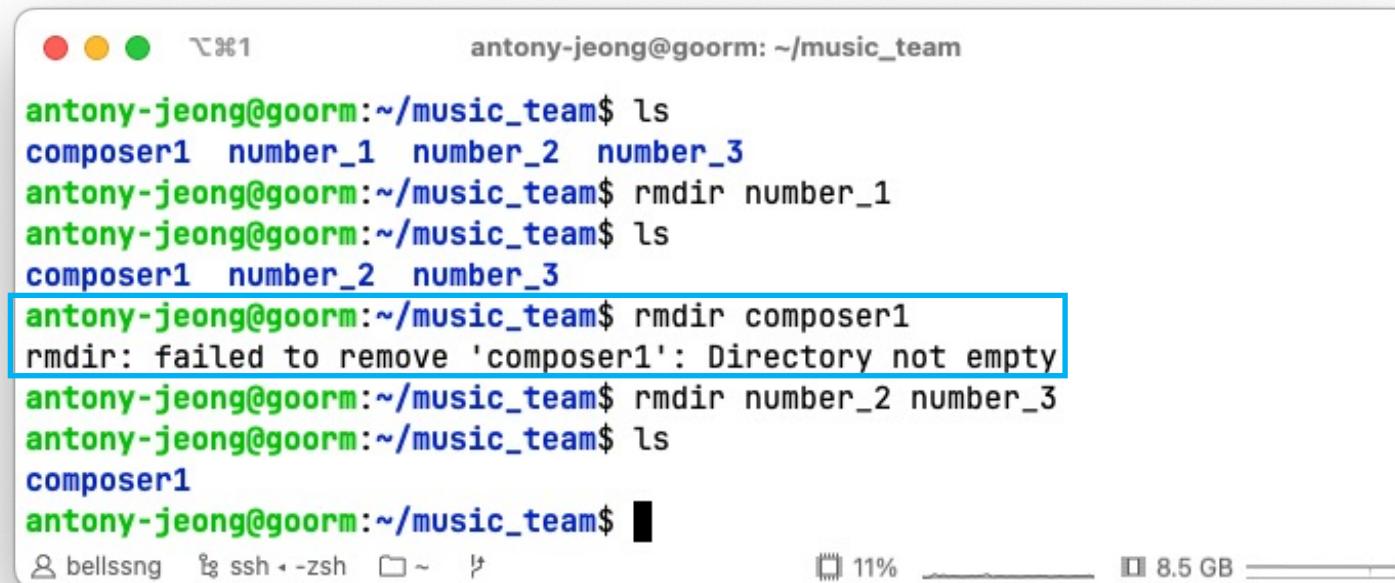
```
antony-jeong@goorm:~/music_team/composer1/composer2/composer3
antony-jeong@goorm:~/music_team$ mkdir -p composer1/composer2/composer3
antony-jeong@goorm:~/music_team$ ls
composer1  number_1  number_2  number_3
antony-jeong@goorm:~/music_team$ cd composer1
antony-jeong@goorm:~/music_team/composer1$ ls
composer2
antony-jeong@goorm:~/music_team/composer1$ cd composer2
antony-jeong@goorm:~/music_team/composer1/composer2$ ls
composer3
antony-jeong@goorm:~/music_team/composer1/composer2$ cd composer3
antony-jeong@goorm:~/music_team/composer1/composer2/composer3$ ls
antony-jeong@goorm:~/music_team/composer1/composer2/composer3$ ■
贝尔斯宁 1g ssh -zsh 18% 8.6 GB
```

빈 디렉터리를 원하는 이름으로 하나 또는 여럿 만들
("-p"를 붙이면 경로 중간의 없는 디렉터리도 만들어 줌)

parent

remove directory

생성과 삭제 - rmdir



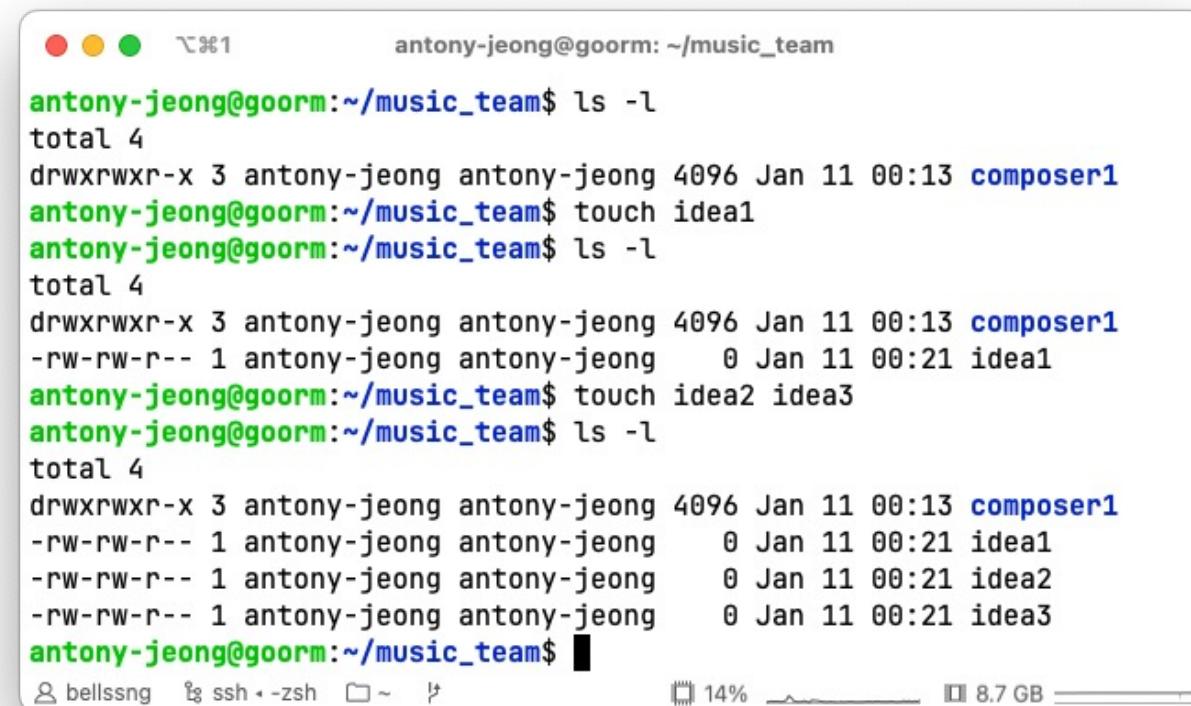
```
antony-jeong@goorm: ~/music_team
antony-jeong@goorm:~/music_team$ ls
composer1  number_1  number_2  number_3
antony-jeong@goorm:~/music_team$ rmdir number_1
antony-jeong@goorm:~/music_team$ ls
composer1  number_2  number_3
antony-jeong@goorm:~/music_team$ rmdir composer1
rmdir: failed to remove 'composer1': Directory not empty
antony-jeong@goorm:~/music_team$ rmdir number_2 number_3
antony-jeong@goorm:~/music_team$ ls
composer1
antony-jeong@goorm:~/music_team$
```

The terminal window shows a user named 'antony-jeong' at a host 'goorm' in the directory '~/music_team'. The user runs 'ls' to list files: 'composer1', 'number_1', 'number_2', and 'number_3'. Then they run 'rmdir number_1', which fails because 'number_1' is not empty. Next, they run 'rmdir number_2 number_3', which also fails for the same reason. Finally, they run 'ls' again and see only 'composer1' left. The bottom of the terminal shows a status bar with icons for user, session, and battery.

지정된 빈 디렉터리를 하나 또는 여럿 제거함 (비어 있지 않으면 안 해줌)

bash에서 “제거”를 하면 복구할 수 있는 휴지통 같은 것이 없으니 주의!

생성과 삭제 - touch

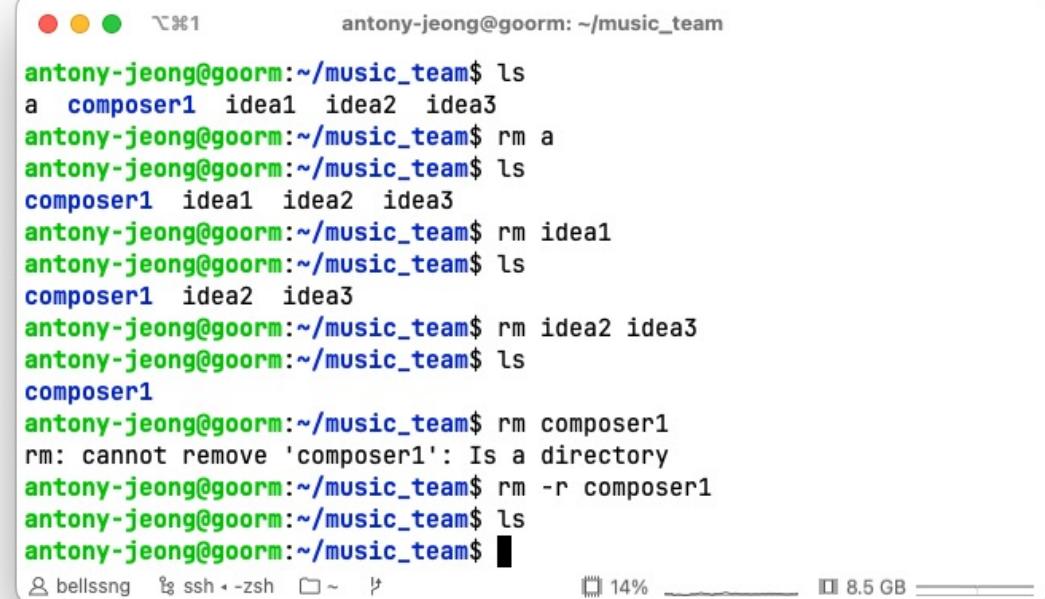


```
antony-jeong@goorm: ~/music_team
antony-jeong@goorm:~/music_team$ ls -l
total 4
drwxrwxr-x 3 antony-jeong antony-jeong 4096 Jan 11 00:13 composer1
antony-jeong@goorm:~/music_team$ touch idea1
antony-jeong@goorm:~/music_team$ ls -l
total 4
drwxrwxr-x 3 antony-jeong antony-jeong 4096 Jan 11 00:13 composer1
-rw-rw-r-- 1 antony-jeong antony-jeong    0 Jan 11 00:21 idea1
antony-jeong@goorm:~/music_team$ touch idea2 idea3
antony-jeong@goorm:~/music_team$ ls -l
total 4
drwxrwxr-x 3 antony-jeong antony-jeong 4096 Jan 11 00:13 composer1
-rw-rw-r-- 1 antony-jeong antony-jeong    0 Jan 11 00:21 idea1
-rw-rw-r-- 1 antony-jeong antony-jeong    0 Jan 11 00:21 idea2
-rw-rw-r-- 1 antony-jeong antony-jeong    0 Jan 11 00:21 idea3
antony-jeong@goorm:~/music_team$
```

빈 파일을 원하는 이름으로 하나 또는 여럿 만듦
(이미 있는 이름이면 그 파일의 최종 수정 시각을 업데이트함 ← 이름의 어원)

생성과 삭제 - `rm`

`remove`



```

antony-jeong@goorm:~/music_team$ ls
a composer1 idea1 idea2 idea3
antony-jeong@goorm:~/music_team$ rm a
antony-jeong@goorm:~/music_team$ ls
composer1 idea1 idea2 idea3
antony-jeong@goorm:~/music_team$ rm idea1
antony-jeong@goorm:~/music_team$ ls
composer1 idea2 idea3
antony-jeong@goorm:~/music_team$ rm idea2 idea3
antony-jeong@goorm:~/music_team$ ls
composer1
antony-jeong@goorm:~/music_team$ rm composer1
rm: cannot remove 'composer1': Is a directory
antony-jeong@goorm:~/music_team$ rm -r composer1
antony-jeong@goorm:~/music_team$ ls
antony-jeong@goorm:~/music_team$ █

```

The terminal window shows a user named 'antony-jeong' in a directory called 'music_team'. The user runs 'ls' to list files: 'a', 'composer1', 'idea1', 'idea2', and 'idea3'. Then, they run 'rm a' to delete the file 'a'. After that, they run 'ls' again to show the remaining files: 'composer1', 'idea1', 'idea2', and 'idea3'. Next, they run 'rm idea1' to delete 'idea1'. They then run 'ls' again to show the files: 'composer1', 'idea2', and 'idea3'. Finally, they run 'rm idea2 idea3' to delete 'idea2' and 'idea3'. They run 'ls' again to show the remaining file: 'composer1'. They then run 'rm composer1' but receive an error message because 'composer1' is a directory. Finally, they run 'rm -r composer1' to delete the directory 'composer1'. They run 'ls' again to show the empty directory.

recursive: 재귀적인

지정한 이름의 파일을 삭제함

“-r”을 붙이면 지정한 이름의 파일이나 디렉터리를 삭제함
마찬가지로 여러 개 지정하면 여러 개 삭제함

이동과 복사 - 이동: mv

```

antony-jeong@goorm: ~/music_team
antony-jeong@goorm:~/music_team$ tree
.
└── number_1
    ├── idea1
    └── idea2
  └── number_2
  └── number_3

3 directories, 2 files
antony-jeong@goorm:~/music_team$ mv number_1 number_2
antony-jeong@goorm:~/music_team$ tree
.
└── number_2
    └── number_1
        ├── idea1
        └── idea2
    └── number_3

3 directories, 2 files
antony-jeong@goorm:~/music_team$ mv number_2/number_1 number4
antony-jeong@goorm:~/music_team$ tree
.
└── number4
    ├── idea1
    └── idea2
  └── number_2
  └── number_3

3 directories, 2 files
antony-jeong@goorm:~/music_team$ mv number4 number13
antony-jeong@goorm:~/music_team$ tree
.
└── number13
    ├── idea1
    └── idea2
  └── number_2
  └── number_3

```

move

mv <source> <dest>

만약 <dest>가 실존하는 디렉터리면
<dest> 내부로 <source>를 이동
아니면

<dest>의 위치로 옮기고
<dest>에 명시된 이름으로 이름을 바꿈

그래서 파일이나 디렉터리의 이름을 바꿀 때도 mv을 씀

파일이 이동할 때 이미 그런 이름의 파일이 있으면 덮어씀
이미 그런 이름의 디렉터리가 있으면 덮어쓰지 않음
디렉터리가 이동할 때 이미 그런 이름의 빈 디렉터리가 있으면 덮어씀
그렇지 않으면 덮어쓰지 않음
이런 걸 다 기억하기는 무리이고,
아무튼 덮어쓴다고 생각하고 확실할 때 쓰는 게 편함
그럼 덮어쓰기를 안 하고 싶으면? 매뉴얼을 읽어보자.

이동과 복사 - 복사: ^{copy}cp

```

antony-jeong@goorm: ~/music_team
antony-jeong@goorm:~/music_team$ tree
.
└── number1
    └── number2
        └── template

2 directories, 1 file
antony-jeong@goorm:~/music_team$ cp template number1
antony-jeong@goorm:~/music_team$ cp template number2/template_for_number2
antony-jeong@goorm:~/music_team$ tree
.
├── number1
│   └── template
├── number2
│   └── template_for_number2
└── template

2 directories, 3 files
antony-jeong@goorm:~/music_team$ █

```

bellssng

```

antony-jeong@goorm: ~/music_team
antony-jeong@goorm:~/music_team$ cp -r number1 template number2
antony-jeong@goorm:~/music_team$ tree
.
└── number1
    └── template
└── number2
    └── number1
        └── template
    └── template
    └── template_for_number2
    └── template

3 directories, 5 files
antony-jeong@goorm:~/music_team$ █

```

bellssng 16% 9.0 GB

`cp <source> <dest>`

만약 `<dest>`가 실존하는 디렉터리이면
`<dest>` 내부에 `<source>`를 복사
복사본 이름은 `<source>` 경로에 명시된 이름
그게 아니면
`<dest>` 경로가 복사본을 가리키도록 복사

기본적으로 경고 없이 덮어쓰기를 함

그럼 덮어쓰기를 안 하고 싶으면? 매뉴얼을 읽어보자.

`cp <source_1> <source_2> ... <source_n> <dest>`
`<dest>`는 실존하는 디렉터리여야 함
source 각각의 복사본을 `<dest>` 안에

디렉터리를 복사하려면 “-r” 붙이기

예시: `cp -r scripts ../scripts`

링크: ln - 심볼릭 링크

link

```
antony-jeong@goorm:~/music_team$ ln -s template link_to_template
antony-jeong@goorm:~/music_team$ tree
.
├── link_to_template -> template
├── number1
│   └── template
└── number2
    ├── number1
    │   └── template
    ├── template
    └── template_for_number2
└── template

3 directories, 6 files
```

```
antony-jeong@goorm:~/music_team$ ln -s number2/template_for_number2 link_to_template_for_number2
antony-jeong@goorm:~/music_team$ tree
.
├── link_to_template -> template
├── link_to_template_for_number2 -> number2/template_for_number2
├── number1
│   └── template
└── number2
    ├── number1
    │   └── template
    ├── template
    └── template_for_number2
└── template

3 directories, 7 files
```

```
antony-jeong@goorm:~/music_team$ cp link_to_template templatitto
antony-jeong@goorm:~/music_team$ tree
.
├── link_to_template -> template
├── link_to_template_for_number2 -> number2/template_for_number2
├── number1
│   └── template
└── number2
    ├── number1
    │   └── template
    ├── template
    └── template_for_number2
└── template
└── templatitto

3 directories, 8 files
```

바로가기 같은 것임

path를 기반으로 본체를 찾아가는 원리!

본체처럼 사용될 수 있으나,
본체가 자리에서 없어지면 가능 x
(근데 같은 이름을 가진 다른 게 생기면
그걸 가리키게 됨)

링크: ln - 하드 링크

link

```
antony-jeong@goorm: ~/music_team
antony-jeong@goorm: ~/music_team$ ln template template_itself
antony-jeong@goorm: ~/music_team$ tree
```

```
.
├── number1
│   └── template
├── number2
│   └── template_for_number2
└── template
    └── template_itself
```

```
2 directories, 4 files
antony-jeong@goorm: ~/music_team$ rm template
antony-jeong@goorm: ~/music_team$ tree
```

```
.
├── number1
│   └── template
├── number2
│   └── template_for_number2
└── template_itself
```

```
2 directories, 3 files
antony-jeong@goorm: ~/music_team$
```

```
antony-jeong@goorm: ~/music_team
antony-jeong@goorm: ~/music_team$ tree
.
├── number1
│   └── template
├── number2
│   └── template_for_number2
└── template_itself

2 directories, 3 files
antony-jeong@goorm: ~/music_team$ ln number1 number1_itself
ln: number1: hard link not allowed for directory
antony-jeong@goorm: ~/music_team$
```

“링크”라기보다는…

실제로는 같은 것을 의미하는 파일을 만드는 것임

→ 본체와 링크 중 어떤 것이 본체인지는 중요하지 않게 됨

* 사실상 링크가 아님 ㅋㅋ

나타나는 현상:

→ 하나를 수정하면 다른 것들도 수정됨

→ 하나가 남아 있으면 나머지를 삭제해도 그 하나가 같은 정보를 가지고 있음

→ 2GB인 파일의 하드 링크를 만들어도 저장 공간을 2GB 더 잡아먹지 않음

디렉터리의 하드 링크는 만들 수 없음



지금쯤 알아둬야 하는 것들

커マン드의 구조

```
antony-jeong@goorm:~/music_team$ cp -r number1 template number2
```

<command>	<flag(s)>	<option(s)>	<argument(s)>
-----------	-----------	-------------	---------------

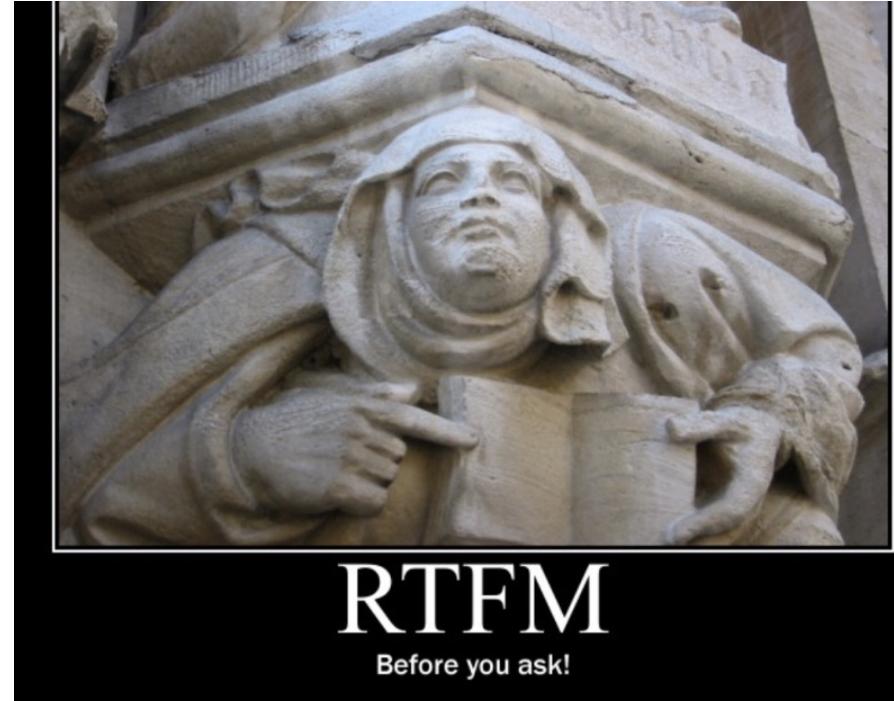
command 어떤 명령을 내릴 것이냐

flag/option 명령 커스텀하기
예시:
rm -r: 디렉터리면 그 안의 모든 항목을 대상으로 rm -r해줘 (recursive)

argument 명령을 수행할 때 대상 - 프로그래밍에서 함수의 argument같은 것임

read the f**king manual

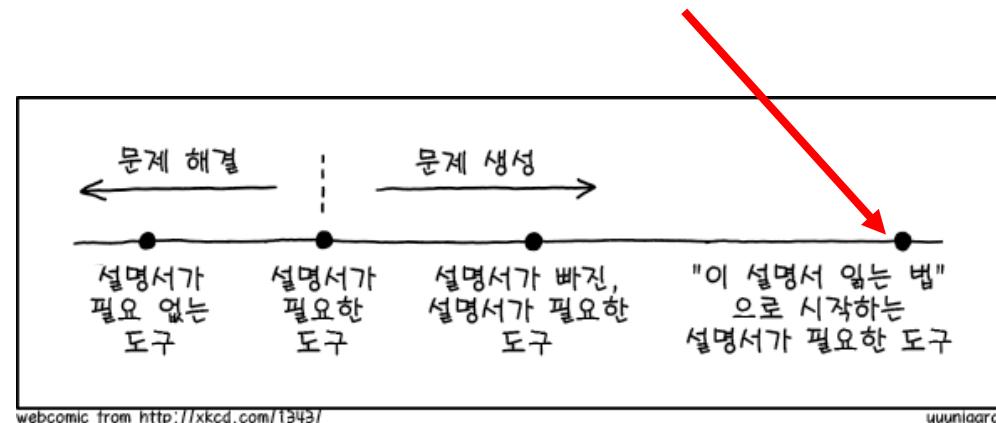
스스로 알아내는 법 - RTFM



이제 “<어떤 명령>에 이런 기능은 없나?” “<어떤 명령을> 어떤 조건 하에 수행하는 법은 없나?” 같은 질문에 대한 답을 스스로 찾아볼 만한 때가 되었습니다.

근데 *manual*이 어디에 있는데요

스스로 알아내는 법 - man pages



man

shell 명령어에 대한 manual을 볼 수 있는 명령어
그 내용물(manual) 혹은 이 도구 자체를 man pages라고 부른다.

사실 shell 명령어 말고 다른 것들에 대한 manual도 많이 포함되어 있음
예시로, 시스템프로그래밍을 수강하면 C언어에서 뭐가 뭔지 모를 때 애용하게 될 것…

조교에게 질문을 하면 “man pages를 찾아보세요”라고 답하는 경우가 많을 것…

그래서 시프를 듣고 나면 man pages에 안 나올 만한 것만 질문하고 있는 나를 발견하게 될 것…

스스로 알아내는 법 - man

```

man cp
General Commands Manual
CP(1)

NAME
    cp - copy files

SYNOPSIS
    cp [-R [-H | -L | -P]] [-fi | -n] [-alpSsvXx] source_file target_file
    cp [-R [-H | -L | -P]] [-fi | -n] [-alpSsvXx]
        source_file ... target_directory
    cp [-f | -i | -n] [-alPpSsvXx] source_file target_file
    cp [-f | -i | -n] [-alPpSsvXx] source_file ... target_directory

DESCRIPTION
    In the first synopsis form, the cp utility copies the contents of the
    source_file to the target_file. In the second synopsis form, the
    contents of each named source_file is copied to the destination
    target_directory. The names of the files themselves are not changed. If
    cp detects an attempt to copy a file to itself, the copy will fail.

    The following options are available:

    -H      If the -R option is specified, symbolic links on the command line
            are followed. (Symbolic links encountered in the tree traversal
            are not followed.)

```

bellssng % less + sh + sh + zsh ⌂ ~ ⌂ 15% ⌂ 8.5 GB ⌂

man <알아볼 명령어>

[man pages 조작법]

스페이스
리턴이나 k

b
j

q

한 페이지씩 아래로
한 줄씩 아래로
한 페이지씩 위로
한 줄씩 위로

나가기

나중에 less 배울 때 자세한 조작법을 알 수 있을 것

스스로 알아내는 법 - “--help”

man pages 말고도 도움말을 얻을 수 있는 방법

대부분의 명령어(혹은 CLI 도구들)는 “**--help**”를 flag로 주면 도움말을 보여 줌

예시: cp --help

커マン드를 잘못 입력하면 “**--help**”를 써 보라라는 메시지를 볼 수도 있음.

예시: arguments 없이 “cp”라고만 명령한다든지…

*man pages*보다 자세할 수도 있고 간략할 수도 있음 (대부분은 *man pages*보다는 간략함)

```
antony-jeong@goorm:~$ cp --help
Usage: cp [OPTION]... [-T] SOURCE DEST
       or: cp [OPTION]... SOURCE... DIRECTORY
       or: cp [OPTION]... -t DIRECTORY SOURCE...
Copy SOURCE to DEST, or multiple SOURCE(s) to DIRECTORY.

Mandatory arguments to long options are mandatory for short options too.
-a, --archive           same as -dR --preserve=all
--attributes-only      don't copy the file data, just the attributes
--backup[=CONTROL]     make a backup of each existing destination file
-b
```

```
antony-jeong@goorm:~$ cp
cp: missing file operand
Try 'cp --help' for more information.
antony-jeong@goorm:~$
```



파일과 입출력 다루기

텍스트 파일 읽기 - cat, head, tail

concatenate

```

● ● ● ၂၇၁ antony-jeong@goorm: ~/music_team

antony-jeong@goorm:~/music_team$ ls
memo

antony-jeong@goorm:~/music_team$ cat memo
The day you couldn't come to my birthday party
The day Heyjin got in so much trouble
The day Jiwon broke up with his girlfriend
The day without you he was always
The day he showed up dressed to the nines
Heard him say

We can go wherever you like
Baby, say the words and I'm down
All I need is you on my side
We can go whenever you like
Now, where are you?

antony-jeong@goorm:~/music_team$ head -n3 memo
The day you couldn't come to my birthday party
The day Heyjin got in so much trouble
The day Jiwon broke up with his girlfriend

antony-jeong@goorm:~/music_team$ tail -n3 memo
All I need is you on my side
We can go whenever you like
Now, where are you?

antony-jeong@goorm:~/music_team$ █

```

bell... ssh ~ ↵ 16% 8.4 GB

cat

내용을 보여줌.

head

처음만 보여줌.

tail

마지막만 보여줌.

head와 tail 모두 $-n <\#lines>$ 로 몇 줄 볼지 정할 수 있음.
(지정하지 않으면 자동 10줄)

익숙치 않을 수 있는 사실: 텍스트 파일이 꼭 *.txt일 필요는 없음.

재미있는 옛날 이야기

more은 less의 전신으로,

남아있기는 하나 모든 기능이 less에도 있기 때문에 아무도 안 씀

텍스트 파일 읽기 - less, more

```
antony-jeong@goorm:~/music_team$ less memo
```

The day you couldn't come to my birthday party
The day Heyjin got in so much trouble
The day Jiwon broke up with his girlfriend
The day without you he was always
The day he showed up dressed to the nines
Heard him say

:|

터미널 화면 전체가 바뀌면서
TUI(text user interface) 텍스트 파일 뷰어 화면으로 바뀜.

"less --help"를 하면 이 뷰어의 사용 방법을 less로 읽을 수 있음.
q로 나갈 수 있음. (man pages가 less를 씀.)

SUMMARY OF LESS COMMANDS

Commands marked with * may be preceded by a number, N.
Notes in parentheses indicate the behavior if N is given.
A key preceded by a caret indicates the Ctrl key; thus ^K is ctrl-K.

h H	Display this help.
q :q Q :Q ZZ	Exit.

MOVING

e ^E j ^N CR	* Forward one line (or <u>N</u> lines).
y ^Y k ^K ^P	* Backward one line (or <u>N</u> lines).
f ^F ^V SPACE	* Forward one window (or <u>N</u> lines).
b ^B ESC-v	* Backward one window (or <u>N</u> lines).
z	* Forward one window (and set window to <u>N</u>).
w	* Backward one window (and set window to <u>N</u>).
ESC-SPACE	* Forward one window, but don't stop at end-of-file.
d ^D	* Forward one half-window (and set half-window to <u>N</u>).
u ^U	* Backward one half-window (and set half-window to <u>N</u>).
ESC-) RightArrow	* Right one half screen width (or <u>N</u> positions).
ESC-(LeftArrow	* Left one half screen width (or <u>N</u> positions).
ESC-} ^RightArrow	Right to last column displayed.
ESC-{ ^LeftArrow	Left to first column.
F	Forward forever; like "tail -f".
ESC-F	Like F but stop when search pattern is found.
r ^R ^L	Repaint screen.
R	Repaint screen, discarding buffered input.

Default "window" is the screen height.
Default "half-window" is half of the screen height.

SEARCHING

/pattern	* Search forward for (<u>N</u> -th) matching line.
?pattern	* Search backward for (<u>N</u> -th) matching line.
n	* Repeat previous search (for <u>N</u> -th occurrence).
N	* Repeat previous search in reverse direction.

텍스트 세기: wc

word count

줄 수, 단어 수, 글자 수를 세 줌.

```
● ○ ● ⌂ 1 antony-jeong@goorm: ~/music_team
```

```
antony-jeong@goorm:~/music_team$ ls
memo
antony-jeong@goorm:~/music_team$ cat memo
The day you couldn't come to my birthday party
The day Heyjin got in so much trouble
The day Jiwon broke up with his girlfriend
The day without you he was always
The day he showed up dressed to the nines
Heard him say
```

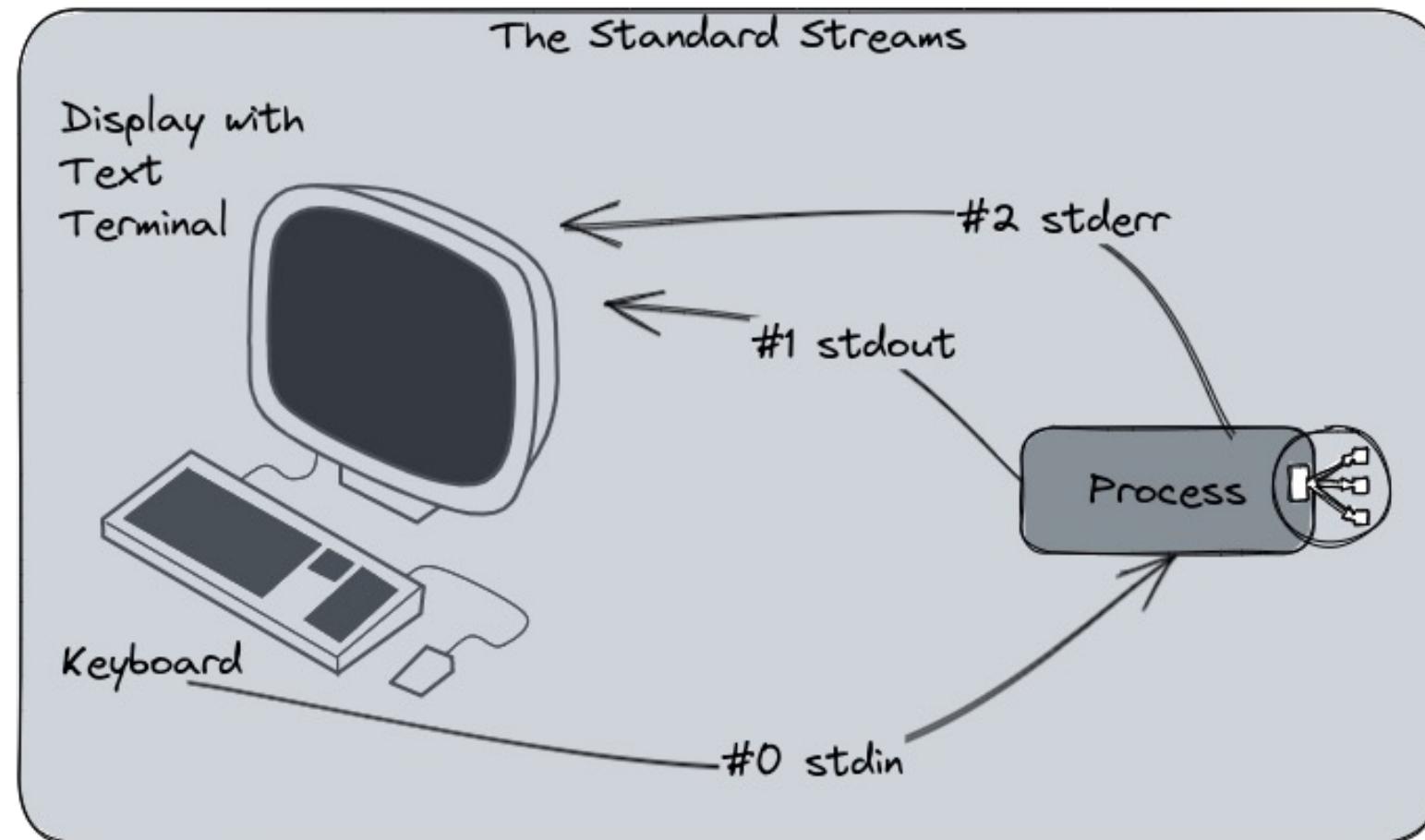
```
We can go wherever you like
Baby, say the words and I'm down
All I need is you on my side
We can go whenever you like
Now, where are you?
antony-jeong@goorm:~/music_team$ head -n3 memo
```

```
● ○ ● ⌂ 1
```

```
antony-jeong@goorm: ~/music_team
```

```
antony-jeong@goorm:~/music_team$ wc memo
12 75 357 memo
antony-jeong@goorm:~/music_team$ █
```

입출력 - The standard streams

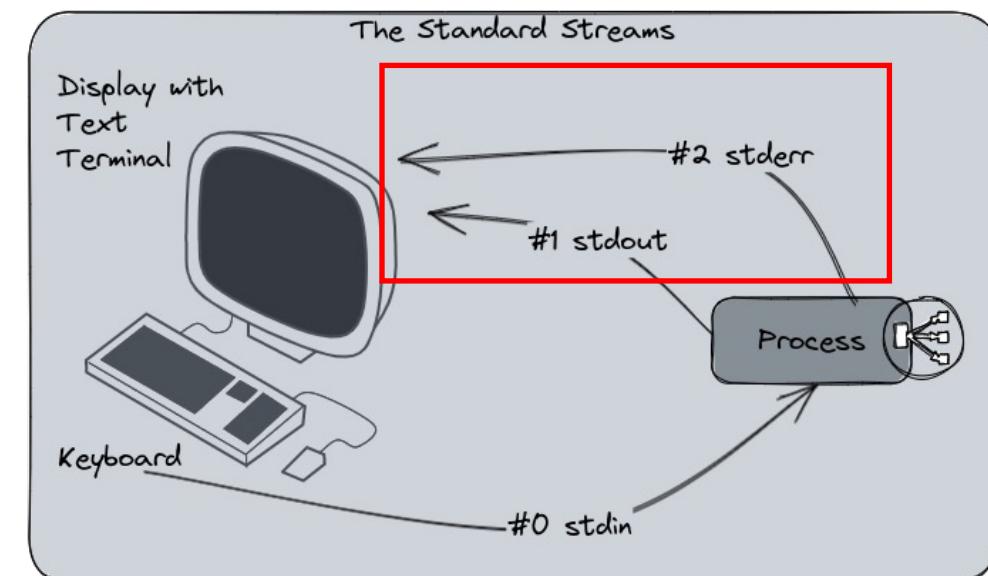


입출력 - redirection 1

```

● ● ● 反响1 antony-jeong@goorm: ~/music_team
antony-jeong@goorm:~/music_team$ wc memo
 12 75 357 memo
antony-jeong@goorm:~/music_team$ wc memo > result
antony-jeong@goorm:~/music_team$ cat result
 12 75 357 memo
antony-jeong@goorm:~/music_team$ wc mmemo
wc: mmemo: No such file or directory
antony-jeong@goorm:~/music_team$ wc mmemo > result
wc: mmemo: No such file or directory
antony-jeong@goorm:~/music_team$ cat result
antony-jeong@goorm:~/music_team$ wc mmemo 2> error
antony-jeong@goorm:~/music_team$ cat error
wc: mmemo: No such file or directory
antony-jeong@goorm:~/music_team$ █

```



- > 파일을 하나 만들어서 거기에 출력을 기록함.
stdout을 redirect
- 2> 파일을 하나 만들어서 거기에 오류를 기록함.
stderr를 redirect

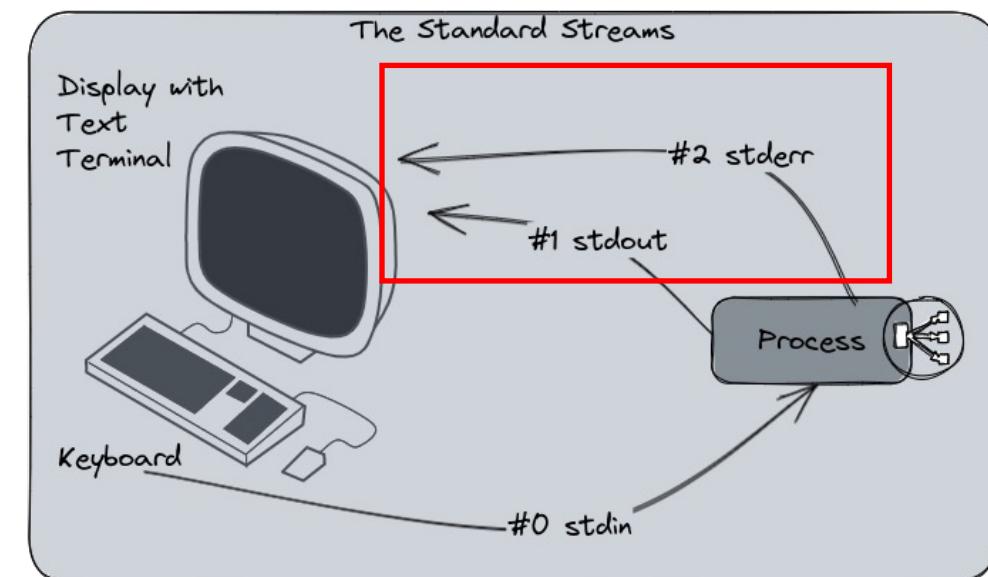
입출력 - redirection 2

```

● ● ● ✘ 1 antony-jeong@goorm: ~/music_team

antony-jeong@goorm:~/music_team$ cat error
wc: mmemo: No such file or directory
antony-jeong@goorm:~/music_team$ wc memo 2>>error
antony-jeong@goorm:~/music_team$ cat error
wc: mmemo: No such file or directory
wc: memo: No such file or directory
antony-jeong@goorm:~/music_team$ wc memo >> error
antony-jeong@goorm:~/music_team$ cat error
wc: mmemo: No such file or directory
wc: memo: No such file or directory
  12  75 357 memo
antony-jeong@goorm:~/music_team$ █
  ↗ bel...  ↗ ssh  ↗ ~  ↗ 17%  ↗ 8.6 GB

```



>>

파일에 출력을 append함. 파일이 없으면 만듦.
stdout을 redirect

2>>

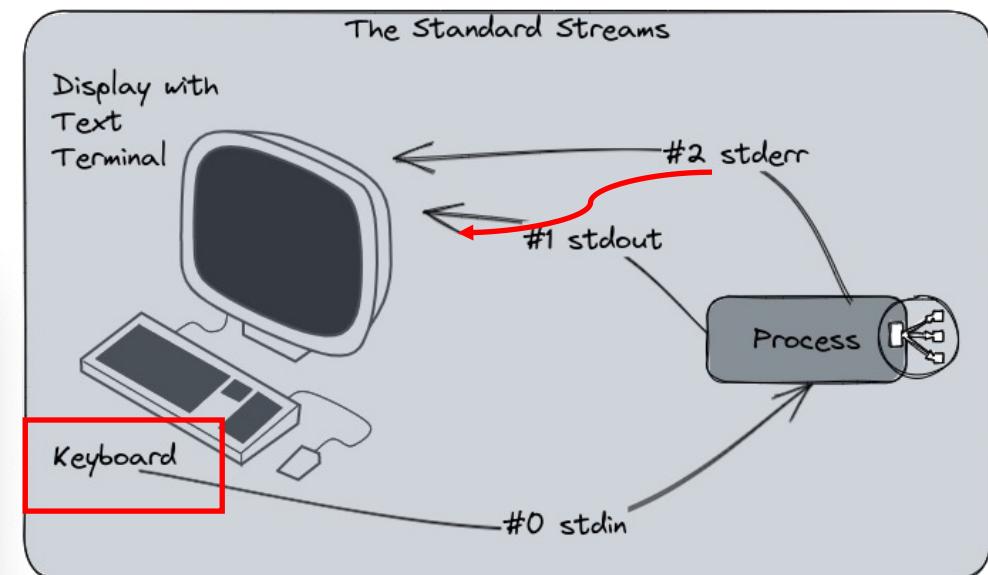
파일에 오류를 append함. 파일이 없으면 만듦.
stderr를 redirect

입출력 - redirection 3

```

antony-jeong@goorm:~/music_team$ cat error
wc: mmemo: No such file or directory
wc: memoo: No such file or directory
12 75 357 memo
antony-jeong@goorm:~/music_team$ wc < error
3 18 91
antony-jeong@goorm:~/music_team$ wc < error > result_and_error 2>&1
antony-jeong@goorm:~/music_team$ cat result_and_error
3 18 91
antony-jeong@goorm:~/music_team$ wc mmemo >> result_and_error 2>&1
antony-jeong@goorm:~/music_team$ cat result_and_error
3 18 91
wc: mmemo: No such file or directory
antony-jeong@goorm:~/music_team$ 
  
```

bellssng 1g ssh -zsh 18% 8.7 GB



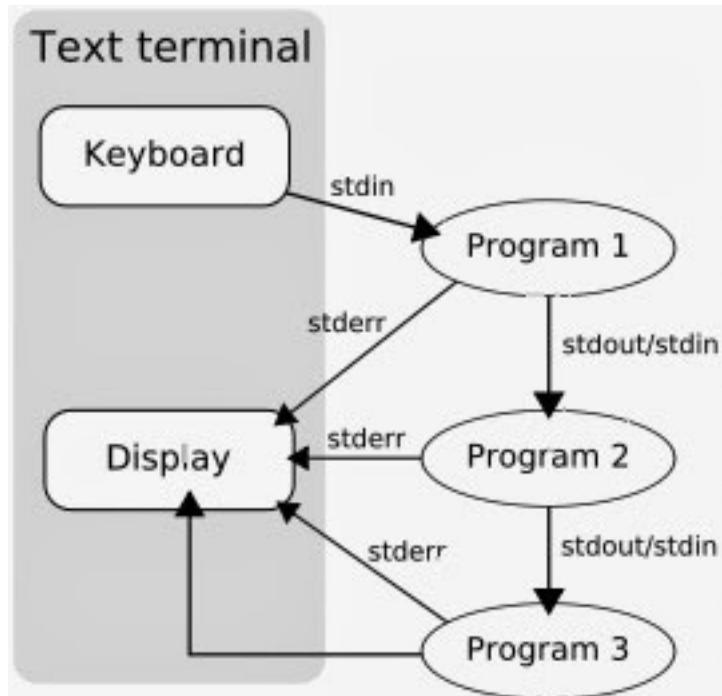
<

키보드 입력 대신 파일의 내용을 씀.
stdin을 redirect

2>&1

stdout이 가는 곳으로 stderr도 보냄.
stderr을 redirect

입출력 - pipeline



```

antony-jeong@goorm:~/music_team$ ls -la
total 20
drwxrwxr-x 2 antony-jeong antony-jeong 4096 Jan 11 02:51 .
drwxr-xr-x 6 antony-jeong antony-jeong 4096 Jan 11 02:23 ..
-rw-rw-r-- 1 antony-jeong antony-jeong 91 Jan 11 02:47 error
-rw-rw-r-- 1 antony-jeong antony-jeong 357 Jan 11 02:23 memo
-rw-rw-r-- 1 antony-jeong antony-jeong 0 Jan 11 02:40 result
-rw-rw-r-- 1 antony-jeong antony-jeong 46 Jan 11 02:51 result_and_error
antony-jeong@goorm:~/music_team$ ls -la | head -n3
total 20
drwxrwxr-x 2 antony-jeong antony-jeong 4096 Jan 11 02:51 .
drwxr-xr-x 6 antony-jeong antony-jeong 4096 Jan 11 02:23 ..
antony-jeong@goorm:~/music_team$ ls -la | head -n3 | tail -n2
drwxrwxr-x 2 antony-jeong antony-jeong 4096 Jan 11 02:51 .
drwxr-xr-x 6 antony-jeong antony-jeong 4096 Jan 11 02:23 ..
antony-jeong@goorm:~/music_team$ █
  
```

bellssng ssh -zsh ~ ↻ 17% 8.6 GB

A | B

A의 stdout 출력이 B의 stdin 입력으로 들어가게 함.
A | B | C | D처럼 계속 이어붙일 수 있음.

입출력 - tee



```
● ● ●  ~*1    antony-jeong@goorm: ~/music_team  
antony-jeong@goorm:~/music_team$ wc memo | tee result  
12 75 357 memo  
antony-jeong@goorm:~/music_team$ cat result  
12 75 357 memo  
antony-jeong@goorm:~/music_team$ █  
👤 belss...  🛡 ssh  ⌂ ~  ↻  17%  ━━━━━━  8.3 GB ━━━━━━
```

tee <filename>

입력을 (1) <filename>이라는 파일에도 저장하고, (2) stdout으로도 내보냄.

텍스트 가공 - grep

```

● ● ● ၂၃၁ antony-jeong@goorm: ~/music_team

antony-jeong@goorm:~/music_team$ grep you memo
The day you couldn't come to my birthday party
The day without you he was always
We can go wherever you like
All I need is you on my side
We can go whenever you like
Now, where are you?
antony-jeong@goorm:~/music_team$ cat memo | grep you
The day you couldn't come to my birthday party
The day without you he was always
We can go wherever you like
All I need is you on my side
We can go whenever you like
Now, where are you?
antony-jeong@goorm:~/music_team$ █

@ bell... ေ ssh ၂ ~ ၂ ၂၀၅% ၂၁၈.၅ GB

● ● ● ၂၃၁ antony-jeong@goorm: ~/music_team

antony-jeong@goorm:~/music_team$ cat memo | grep -v you
The day Heyjin got in so much trouble
The day Jiwon broke up with his girlfriend
The day he showed up dressed to the nines
Heard him say

Baby, say the words and I'm down
antony-jeong@goorm:~/music_team$ █

@ bell... ေ ssh ၂ ~ ၂ ၁၄% ၂၁၈.၅ GB ၂၁၈.၅ GB

```

사실 더 많은 걸 할 수 있는 친구임
근데 나도 깊게 잘 모름
궁금해졌다면… Read man pages

입력되는 line 중에서 <pattern>을 포함하는 것만 출력함.

“-v” flag를 주면 포함하지 않는 것만 출력함.

텍스트 가공 - uniq, sort, join

```
antony-jeong@goorm:~/music_team$ cat weird
The day you couldn't come to my birthday party
The day Heyjin got in so much trouble
The day Jiwon broke up with his girlfriend
The day Jiwon broke up with his girlfriend
The day Jiwon broke up with his girlfriend
The day without you he was always
The day he showed up dressed to the nines
Heard him say
Heard him say

We can go wherever you like
Baby, say the words and I'm down
All I need is you on my side
We can go whenever you like
Now, where are you?
antony-jeong@goorm:~/music_team$ uniq weird
The day you couldn't come to my birthday party
The day Heyjin got in so much trouble
The day Jiwon broke up with his girlfriend
The day without you he was always
The day he showed up dressed to the nines
Heard him say

We can go wherever you like
Baby, say the words and I'm down
All I need is you on my side
We can go whenever you like
Now, where are you?
antony-jeong@goorm:~/music_team$
```

중복되는 줄 없이 출력하기

```
antony-jeong@goorm:~/music_team$ cat list
Book
Joy
Exit
Abrupt
King
Crown
Dose
Europe
antony-jeong@goorm:~/music_team$ sort list
Abrupt
Book
Crown
Dose
Europe
Exit
Joy
King
antony-jeong@goorm:~/music_team$
```

줄을 사전 순서대로 정렬하기

```
antony-jeong@goorm:~/music_team$ cat list1
A
B
C
E
F
G
G
H
antony-jeong@goorm:~/music_team$ cat list2
A
C
D
E
G
I
I
antony-jeong@goorm:~/music_team$ join list1 list2
A
C
E
G
G
antony-jeong@goorm:~/music_team$
```

교집합만 남기기

텍스트 가공 - diff, patch

```
antony-jeong@goorm:~/music_team$ cat memo
The day you couldn't come to my birthday party
The day Heyjin got in so much trouble
The day Jiwon broke up with his girlfriend
The day without you he was always
The day he showed up dressed to the nines
Heard him say

We can go wherever you like
Baby, say the words and I'm down
All I need is you on my side
We can go whenever you like
Surprise! Mother, father.
Now, where are you?
antony-jeong@goorm:~/music_team$
```

차이



```
antony-jeong@goorm:~/music_team$ cat memo_modified
The day you couldn't come to my birthday party
The day without you he was always
The day he showed up dressed to the nines
Heard him say

We can get whatever you want
Baby, say the words and I'm down
All I need is you on my side
We can go whenever you like
Surprise! Mother, father.
Now, where are you?
antony-jeong@goorm:~/music_team$
```

```
antony-jeong@goorm:~/music_team$ diff memo memo_modified
2,3d1
< The day Heyjin got in so much trouble
< The day Jiwon broke up with his girlfriend
8c6
< We can go wherever you like
---
> We can get whatever you want
11a10
> Surprise! Mother, father.
antony-jeong@goorm:~/music_team$
```

diff: 두 파일의 차이를 정해진 형식으로 알려 줌.

사실 더 많은 걸 할 수 있는 친구들임
근데 나도 깊게 잘 모름
궁금해졌다면… Read man pages

```
antony-jeong@goorm:~/music_team$ diff memo memo_modified > memo.diff
antony-jeong@goorm:~/music_team$ patch memo memo.diff
patching file memo
antony-jeong@goorm:~/music_team$ cat memo
The day you couldn't come to my birthday party
The day without you he was always
The day he showed up dressed to the nines
Heard him say

We can get whatever you want
Baby, say the words and I'm down
All I need is you on my side
We can go whenever you like
Surprise! Mother, father.
Now, where are you?
antony-jeong@goorm:~/music_team$
```

patch:

기존 파일 A와 원하는 결과물 B가 있다고 할 때,
diff A B의 결과로 나올 것을 주면 B를 만들어 줌.

텍스ㅌ 편집: vi

visual editor



TUI 텍스트 에디터임.
메모장처럼 기본 프로그램인데
깊게 파고들면 기능이 매우 많음.

vi <파일명>

[vi 초간단 조작법]

명령 모드에서는…

:q리턴

나가기

:wq리턴

저장하고 나가기

:q!리턴

변경 사항 저장하지 않고 나가기

i

입력 모드로 전환

입력 모드에서는…

문자 입력

입력하기

방향키

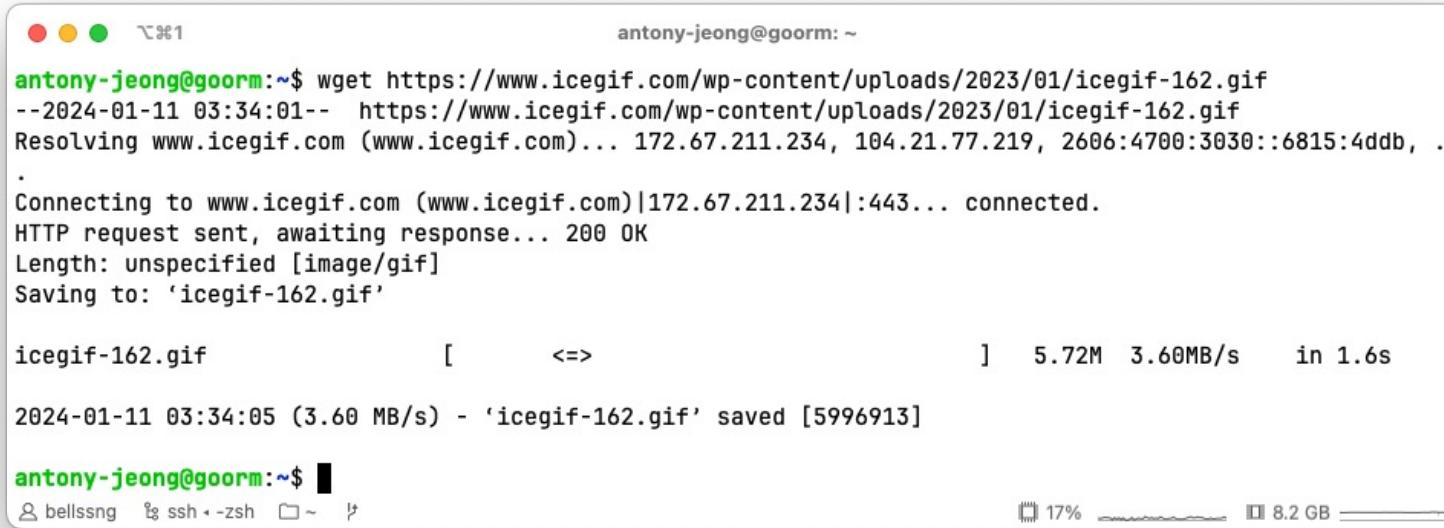
커서 이동하기

Esc

명령 모드로 전환

파일 다운로드: wget, curl

web get



```
antony-jeong@goorm:~$ wget https://www.icegif.com/wp-content/uploads/2023/01/icegif-162.gif
--2024-01-11 03:34:01-- https://www.icegif.com/wp-content/uploads/2023/01/icegif-162.gif
Resolving www.icegif.com (www.icegif.com)... 172.67.211.234, 104.21.77.219, 2606:4700:3030::6815:4ddb, ...
.
Connecting to www.icegif.com (www.icegif.com)|172.67.211.234|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [image/gif]
Saving to: 'icegif-162.gif'

icegif-162.gif          [           =>                      ]  5.72M  3.60MB/s   in 1.6s

2024-01-11 03:34:05 (3.60 MB/s) - 'icegif-162.gif' saved [5996913]

antony-jeong@goorm:~$
```

The terminal window shows the execution of a wget command to download a GIF file from the URL <https://www.icegif.com/wp-content/uploads/2023/01/icegif-162.gif>. The output shows the connection details, the HTTP request, the file length, and the download progress. The file is saved as 'icegif-162.gif'.

wget <url>

url을 방문해 파일을 다운받음.

사실 wget은 인터넷과 관련해 훨씬 더 다양한 일을 할 수 있음.
curl도 그런 도구임.

인터넷을 이용한 복잡한 일을 할 때는 curl이 wget보다 주류이나,
단순하게 파일을 다운로드 받을 때는 wget이면 충분함.
나도 curl 어떻게 쓰는지 잘 모름 ;) Read man pages

tape archives

파일 압축: tar, zip

어떤 전산과/전자과 과목 교수님들은 과제 제출을 .tar.gz 압축 파일로 하라고 함.

나도 안 하다 보면 까먹어서 검색해서 하는 방법 알아보니 여러분도 그렇게 하도록.
(자주 쓰면 기억하게 됨.)

피곤하지만 효율적인 것들



커맨드 여러 개 다루기

```
● ○ ● ၮၯ၁      antony-jeong@goorm: ~/music_team
antony-jeong@goorm:~/music_team$ cat memo; cat memo.diff; wc memo
The day you couldn't come to my birthday party
The day without you he was always
The day he showed up dressed to the nines
Heard him say

We can get whatever you want
Baby, say the words and I'm down
All I need is you on my side
We can go whenever you like
Surprise! Mother, father.
Now, where are you?
2,3d1
< The day Heyjin got in so much trouble
< The day Jiwon broke up with his girlfriend
8c6
< We can go wherever you like
---
> We can get whatever you want
11a10
> Surprise! Mother, father.
 11 62 303 memo
antony-jeong@goorm:~/music_team$ █
👤 bellssng 腠 ssh -zsh က ~ ပဲ ၁၉% ၈.၄ GB
```

```
● ○ ● ၮၯ၁      antony-jeong@goorm: ~/music_team
antony-jeong@goorm:~/music_team$ wc memo && wc mmemo && wc memo
 11 62 303 memo
wc: mmemo: No such file or directory
antony-jeong@goorm:~/music_team$ wc mmemo || wc memo || wc mmemo
wc: mmemo: No such file or directory
 11 62 303 memo
antony-jeong@goorm:~/music_team$ █
👤 bellssng 腠 ssh -zsh က ~ ပဲ ၁၆% ၈.၃ GB
```

;, &&, ||로 커맨드를 연결하면 커맨드를 순서대로 실행하는데…

;

무조건 다 실행함.

&&

실패하면 그만둠.

||

성공하면 그만둠.

job이란

말 그대로 “작업”

기본적으로 bash에서는 user가 한 번에 내린 커맨드 덩어리이긴 하나,
예시:

sleep 5 | sleep 10 | wc

sleep 1 && echo tick && sleep 1 && echo tock

wc qwerty || wc memo || wc qwerty && wc memo || wc qwerty

예외가 있음.

예외:

“;”로 연결된 것은 따로따로 내린 커맨드로 취급함.

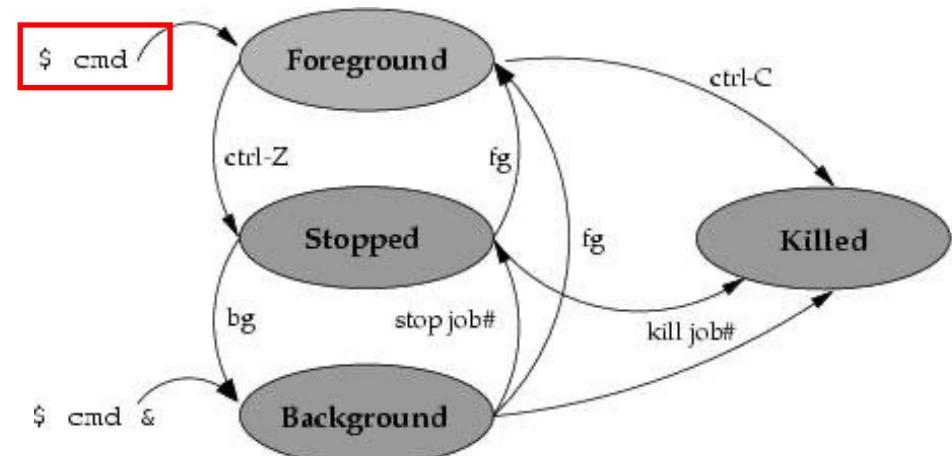
“&”로 연결된 것은 따로따로 내린 커맨드로 취급함.

별 의미는 없고 *shell implementor*가 임의로 만든 기준임.

(job은 kernel의 기능이 아닌 shell의 기능임. Shell의 종류마다 작동 방식이 다를 수 있음.)

job의 상태(state)

- Foreground
 - 여러분이 지금까지 배운 방법으로 명령을 내리면 foreground에서 job을 수행함.
 - stdin, stdout, stderr가 모두 user와 terminal로 연결된 상태
 - 또한 job이 끝날 때까지 shell 기능으로 돌아오지 않음!
- Stopped
 - 작업이 정지되었음.
- Background
 - stdin, stdout, stderr가 terminal에 연결되지 않은 채 작업을 수행함.
- Killed/Terminated
 - 작업이 중지되었으나 아직 보고되지 않았음.
- Done
 - 작업이 끝났으나 아직 보고되지 않았음.

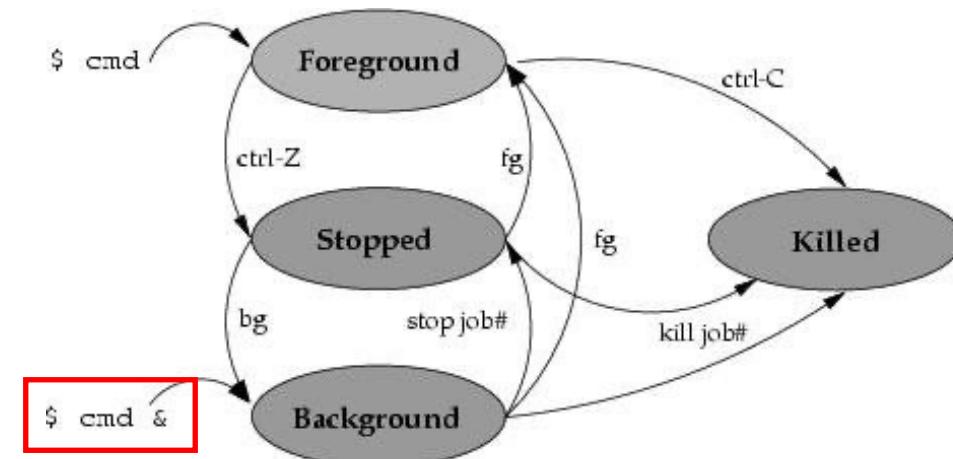


background에서 job 시작하기: &

```

● ○ ● ℗1 antony-jeong@goorm: ~
antony-jeong@goorm:~$ sleep 5 &
[1] 963
antony-jeong@goorm:~$ █
† 12% █ 8.9 GB █

```



뒤에 &을 붙이면 작업을 background에서 시작함.

입력할 때 & 뒤에 다른 job을 이어서 입력할 수 있음.
(별도 job으로 처리됨.)

foreground에서 job 정지하기: ctrl+Z

```

● ● ●  ~%1 antony-jeong@goorm:~  

antony-jeong@goorm:~$ sleep 10  

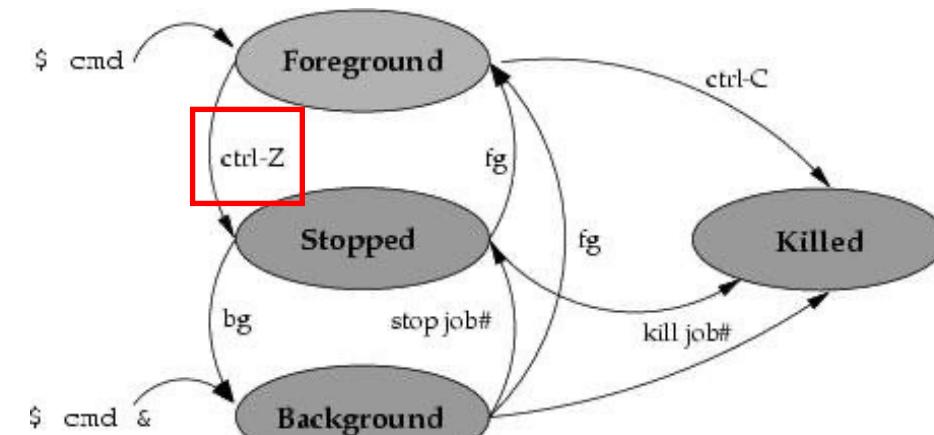
^Z  

[1]+  Stopped                 sleep 10  

antony-jeong@goorm:~$  

% ssh  ~  ↵  17%  ━━━━━━  8.9 GB

```



작업 도중에 `ctrl+Z`를 누르면

- (1) 컴퓨터는 하던 작업을 멈추고
- (2) user는 관련 메시지와 함께 shell prompt를 다시 마주함.

현재 job 모두 보기: jobs



```
antony-jeong@goorm:~$ sleep 1 & sleep 2 & sleep 10 & sleep 10 &
[1] 1380
[2] 1381
[3] 1382
[4] 1383
antony-jeong@goorm:~$ jobs
[1]  Done                  sleep 1
[2]  Done                  sleep 2
[3]- Running               sleep 10 &
[4]+ Running               sleep 10 &
antony-jeong@goorm:~$ jobs
[3]- Running               sleep 10 &
[4]+ Running               sleep 10 &
antony-jeong@goorm:~$ jobs
[3]- Done                  sleep 10
[4]+ Done                  sleep 10
antony-jeong@goorm:~$ jobs
antony-jeong@goorm:~$
```

job의 번호, job의 상태, user가 시킨 job을 보여줌.

Done이 jobs를 통해 보고되면 job이 없어짐.
Foreground에서 job이 끝나도 없어짐.

job 상태 바꾸기: bg, fg

background

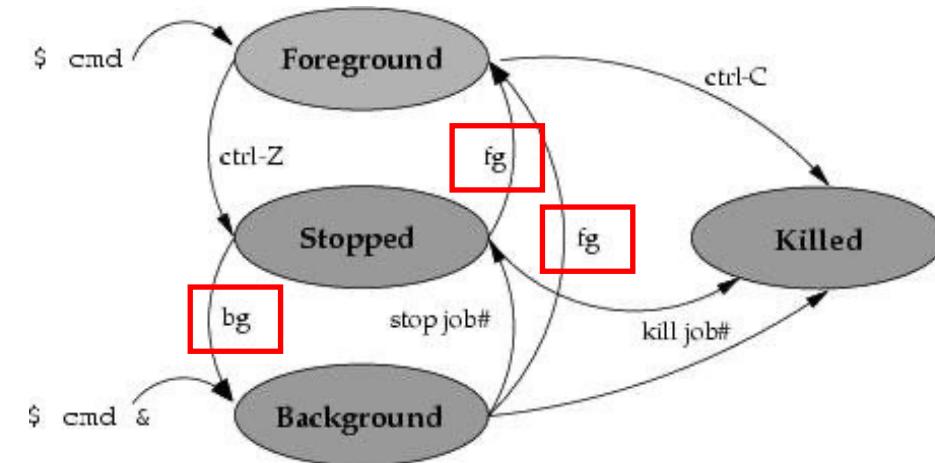
foreground

```

● ● ●  ~%1      antony-jeong@goorm: ~
antony-jeong@goorm:~$ sleep 60
^Z
[1]+  Stopped          sleep 60
antony-jeong@goorm:~$ jobs
[1]+  Stopped          sleep 60
antony-jeong@goorm:~$ bg 1
[1]+ sleep 60 &
antony-jeong@goorm:~$ jobs
[1]+  Running          sleep 60 &
antony-jeong@goorm:~$ fg 1
sleep 60
^Z
[1]+  Stopped          sleep 60
antony-jeong@goorm:~$ jobs
[1]+  Stopped          sleep 60
antony-jeong@goorm:~$ fg 1
sleep 60
^Z
[1]+  Stopped          sleep 60
antony-jeong@goorm:~$ jobs
[1]+  Stopped          sleep 60
antony-jeong@goorm:~$ fg 1
sleep 60

```

bel... ssh ~ 18% 8.9 GB



bg/fg <stopped인 job의 번호>
background/foreground에서 재개함

fg <background에서 실행 중인 job의 번호>
fg로 바꿈 (입출력이 user와 연결됨)

job이 끝나기를 기다리기: wait



```
● ● ●  ~%1  antony-jeong@goorm: ~  
antony-jeong@goorm:~$ sleep 10 &  
[1] 2161  
antony-jeong@goorm:~$ jobs  
[1]+  Running                  sleep 10 &  
antony-jeong@goorm:~$ wait %1  
[1]+  Done                    sleep 10  
antony-jeong@goorm:~$ █  
↳ ssh  □ ~  ↵  █ 17%  █ 8.9 GB  █
```

A screenshot of a terminal window titled 'antony-jeong@goorm: ~'. The user runs 'sleep 10 &' which creates a background job with ID [1] 2161. Then, they run 'jobs' to list the job. Finally, they run 'wait %1' to wait for the job to finish. The terminal shows the job running in the background and then completed when the user presses enter again.

wait %<job의 번호>

job이 끝날 때까지 기다림.

job을 foreground로 보내는 것이 아님.
그냥 끝날 때까지 기다림.

process란

이것도… 일의 단위인데 커맨드 하나당 process가 하나 이상 만들어짐.
(built-in 커맨드는 제외)

커맨드 하나 자체를 담당하는 process는 하나일 것
자세한 것은 시프/전프구 시간에

process 보기: ps - 사용법

process



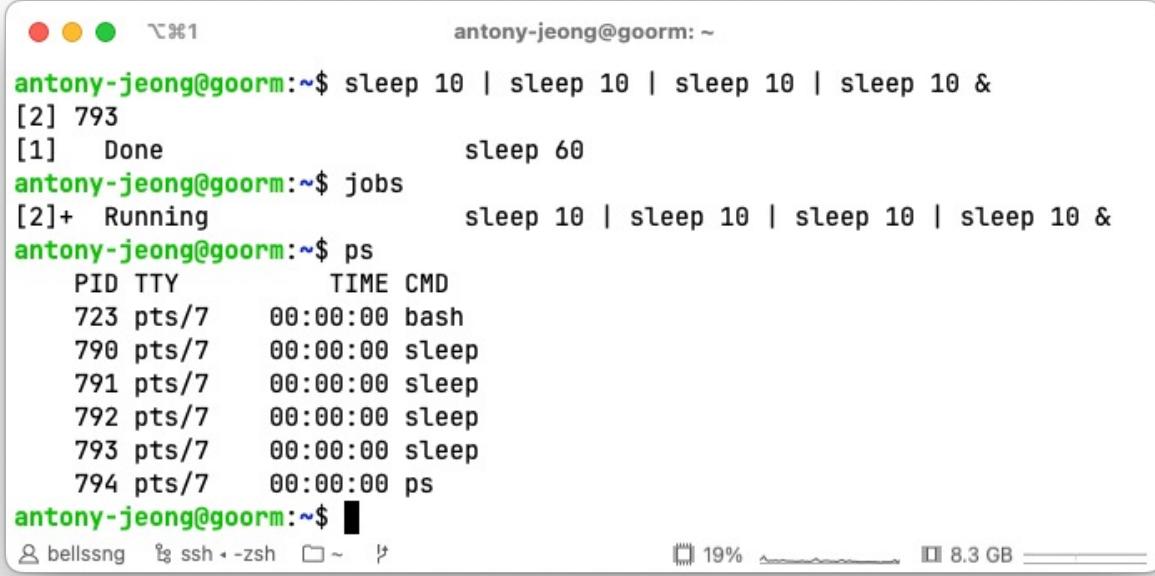
A screenshot of a terminal window titled 'antony-jeong@goorm: ~'. The user runs 'sleep 60 &' which creates a new process [1] with ID 750. Then, they run 'ps' to list all processes. The output shows three processes: bash (PID 723), sleep (PID 750), and ps (PID 751). The terminal window has a dark background with light-colored text. At the bottom, there are icons for a home folder, a search bar, battery level (24%), and RAM usage (8.2 GB).

```
antony-jeong@goorm:~$ sleep 60 &
[1] 750
antony-jeong@goorm:~$ ps
 PID TTY      TIME CMD
 723 pts/7    00:00:00 bash
 750 pts/7    00:00:00 sleep
 751 pts/7    00:00:00 ps
antony-jeong@goorm:~$
```

지금 shell에 로그인해서 실행한 process들을 보여 줌.

- e 모든 사용자의 모든 process들 보기
- u <username> 특정 사용자의 process들 보기
- f 자세한 정보 보기
- p <pid> 특정 process 보기

process 보기: ps - 예시: pipelined job process



The screenshot shows a macOS terminal window with the following session:

```
antony-jeong@goorm:~$ sleep 10 | sleep 10 | sleep 10 | sleep 10 &
[2] 793
[1] Done                  sleep 60
antony-jeong@goorm:~$ jobs
[2]+  Running              sleep 10 | sleep 10 | sleep 10 | sleep 10 &
antony-jeong@goorm:~$ ps
  PID TTY      TIME CMD
  723 pts/7    00:00:00 bash
  790 pts/7    00:00:00 sleep
  791 pts/7    00:00:00 sleep
  792 pts/7    00:00:00 sleep
  793 pts/7    00:00:00 sleep
  794 pts/7    00:00:00 ps
antony-jeong@goorm:~$
```

The terminal window has a title bar "antony-jeong@goorm: ~". At the bottom, there are status icons for bell, ssh, zsh, battery (19%), and disk usage (8.3 GB).

pipeline된 커マン드들은 job으로는 하나지만 각각의 process임

process 끝내기 - 방법

foreground에서 가장 편하고 반사적으로 누르기 좋아서 가장 많이 쓰임

foreground에서 실행 중인 process:

ctrl+C

ctrl+\

SIGINT로 종료함

SIGQUIT으로 종료함

shell에서:

kill <pid>

kill -9 <pid>

SIGTERM으로 종료함.

SIGKILL로 종료함.

프로그램 제작자는
프로그램이 이것들에 반응하지 않도록
혹은 원하는 방식으로 반응하도록
조정할 수 있음.

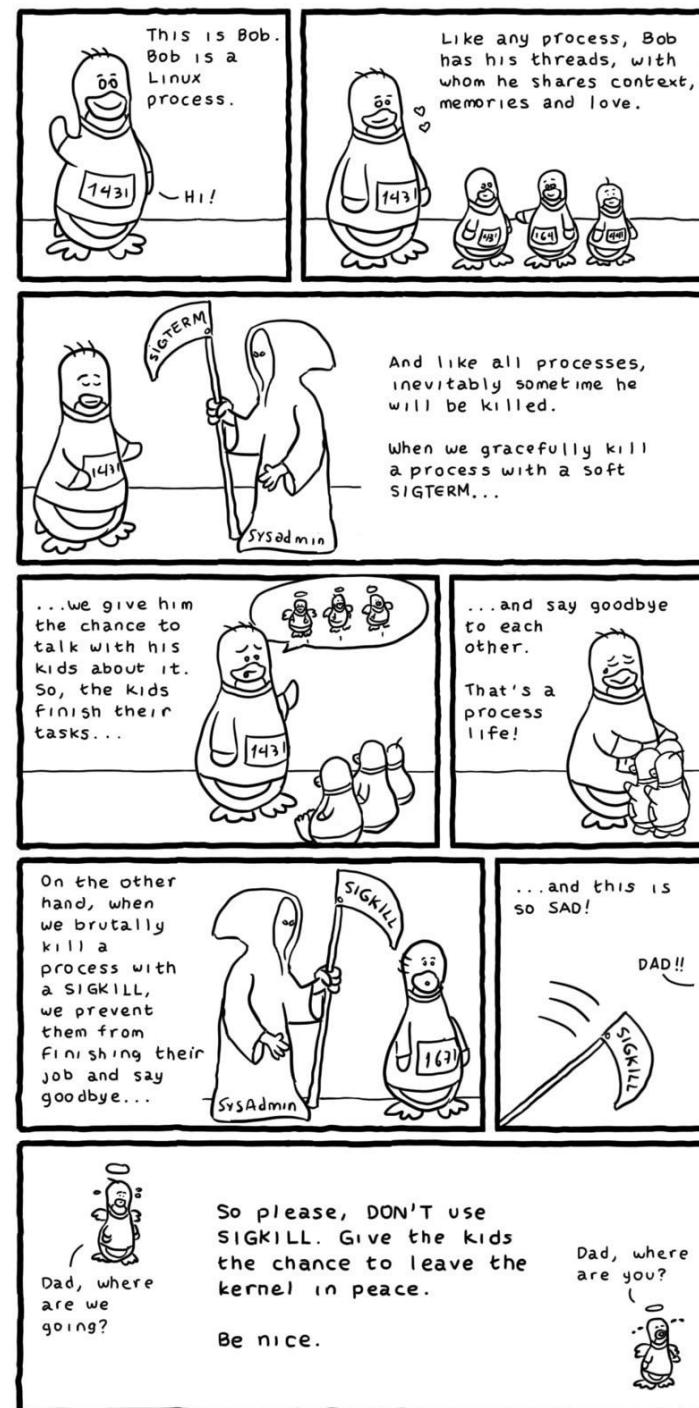
그러나 SIGKILL은 거부할 수 없음.

SIG뭐시기가 뭔지는 시프에서 자세히 배우는 걸로…

process 끝내기 - 좋은 방법

사실 잘 만든 프로그램이라면
SIGTERM을 주면 잠시 후 정상적으로 잘 종료되어야 함.
왜냐하면 SIGTERM은
“어차피 끝낼 거긴 한데,
부작용 없게 어떤어떤 일은 마무리하자”
의 취지로 주는 신호이기 때문.

SIGKILL은 부작용이 있을 “수도” 있음.



process가 끝나기를 기다리기: wait



A screenshot of a terminal window titled 'antony-jeong@goorm: ~'. The terminal shows the following command sequence:

```
antony-jeong@goorm:~$ sleep 60 &
[1] 1130
antony-jeong@goorm:~$ wait 1130
```

The terminal window has a light gray background and a dark gray border. At the bottom, there is a status bar with icons for file, search, and system information, showing '24%' and '7.6 GB'.

wait <pid>

지정한 프로세스가 끝날 때까지 기다림.

bash로 프로그래밍하기

bash는 프로그래밍 언어이기도 함. (정의 상으로는 이 아니라 실제로 많이 함)

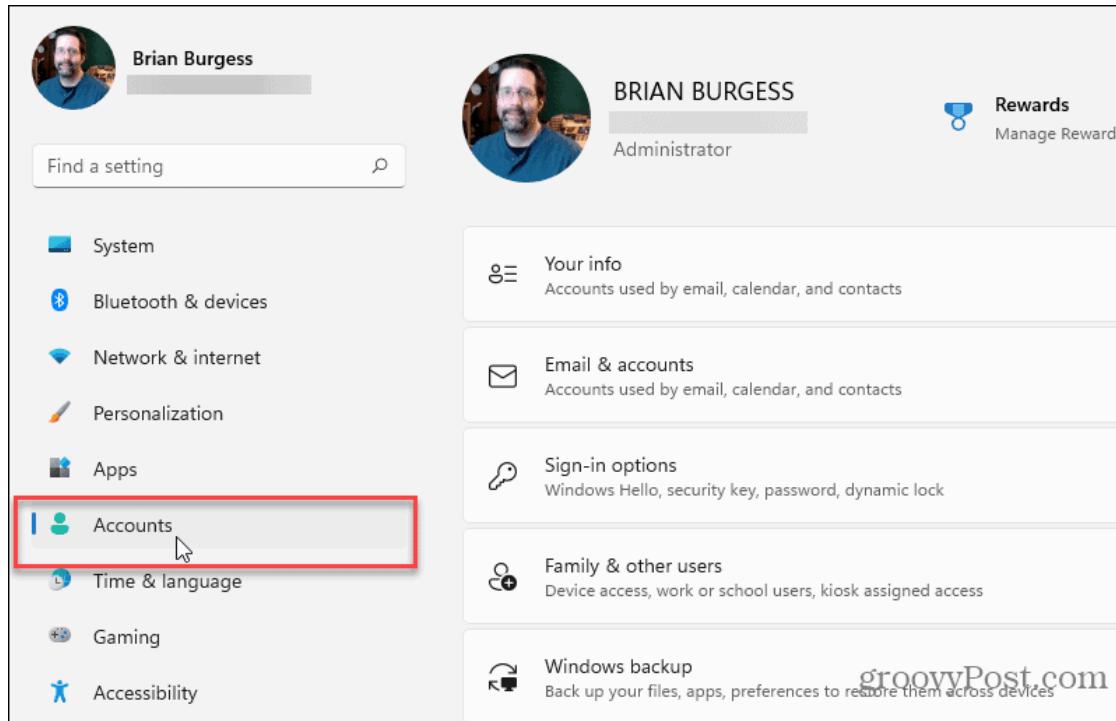
변수를 선언할 수 있고,
조건문과 반복문도 쓸 수 있음.

복잡한 걸 해 보고 싶다면 열심히 검색해보자~



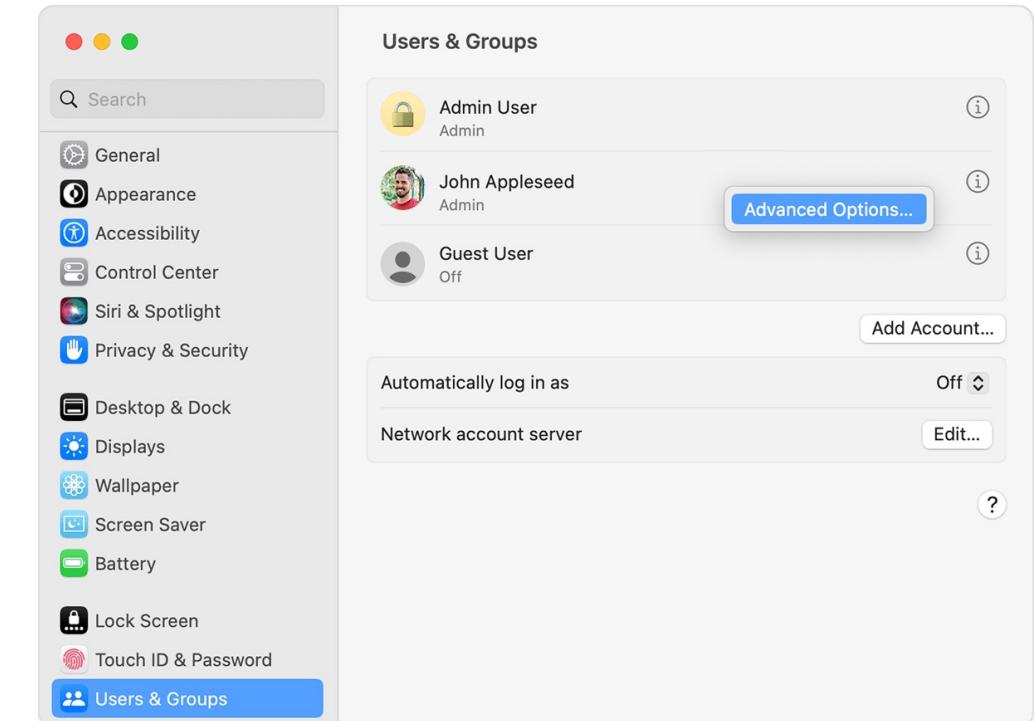
자잘하지만 필요한 것들

user란



Windows의 account

컴퓨터에 접속하는 주체 각각을 컴퓨터가 구별할 수 있는 한 단위임.



MacOS의 user

누구의 것인가: ownership - 무엇인가

d: 디렉터리

-: 파일

l: 링크

권한

이 항목의 주인

이 항목의 그룹

용량

이름

최종 수정 시각

```
● ○ ● ☰ 1 antony-jeong@goorm: ~
antony-jeong@goorm:~$ ls -la
total 36
drwxr-xr-x  5 antony-jeong antony-jeong 4096 Jan 10 23:22 .
drwxr-xr-x 12 root         root          4096 Jan  9  01:12 ..
-rw-----  1 antony-jeong antony-jeong   757 Jan 10 23:20 .bash_history
-rw-r--r--  1 antony-jeong antony-jeong  220 Jan  7 17:05 .bash_logout
-rw-r--r--  1 antony-jeong antony-jeong 3771 Jan  7 17:05 .bashrc
drwx----- 2 antony-jeong antony-jeong 4096 Jan  7 17:06 .cache
drwx----- 3 antony-jeong antony-jeong 4096 Jan  9 12:13 .config
-rw-r--r--  1 antony-jeong antony-jeong  807 Jan  7 17:05 .profile
-rw-rw-r--  1 antony-jeong antony-jeong    0 Jan 10 23:08 number_1.mp3
drwxrwxr-x  2 antony-jeong antony-jeong 4096 Jan 10 23:17 receipts
-rw-rw-r--  1 antony-jeong antony-jeong    0 Jan 10 23:08 synopsis.txt
antony-jeong@goorm:~$
```

16% 9.4 GB

누구의 것인가: ownership - 설정 방법



The screenshot shows a terminal window with the following session:

```
antony-jeong@goorm: ~
antony-jeong@goorm:~$ ls -la synopsis.txt
-rw-rw-r-- 1 antony-jeong antony-jeong 0 Jan 10 23:08 synopsis.txt
antony-jeong@goorm:~$ sudo chown derick321:derick321 synopsis.txt
antony-jeong@goorm:~$ ls -la synopsis.txt
-rw-rw-r-- 1 derick321 derick321 0 Jan 10 23:08 synopsis.txt
antony-jeong@goorm:~$
```

The terminal window has a title bar with three colored dots (red, yellow, green) and the identifier "TextEdit 1". The user is "antony-jeong" at host "goorm". The session starts with "ls -la synopsis.txt" showing the file belongs to "antony-jeong". Then "sudo chown derick321:derick321 synopsis.txt" is run, changing the owner to "derick321". A final "ls -la synopsis.txt" shows the file now belongs to "derick321". The bottom of the window shows system status: "bellssng" is running via ssh, battery level is 12%, and disk usage is 8.6 GB.

chown

주인을 바꿈.
(그룹도 함께 바꿀 수 있음.)

chgrp

그룹을 바꿈.

뭘 할 수 있는가: permission - 무엇인가

d: 디렉터리

-: 파일

l: 링크

```
antony-jeong@goorm:~$ ls -la
total 36
drwxr-xr-x  5 antony-jeong antony-jeong
drwxr-xr-x 12 root        root
-rw-----  1 antony-jeong antony-jeong
-rw-r--r--  1 antony-jeong antony-jeong
-rw-r--r--  1 antony-jeong antony-jeong
drwx-----  2 antony-jeong antony-jeong
drwx-----  3 antony-jeong antony-jeong
-rw-r--r--  1 antony-jeong antony-jeong
-rw-r--r--  1 antony-jeong antony-jeong
drwxrwxr-x  2 antony-jeong antony-jeong
-rw-rw-r--  1 antony-jeong antony-jeong
anton-jeong@goorm:~$
```

이 항목의 주인이 가진 권한

이 항목의 그룹 인원이 가진 권한

다른 모든 이들이 가진 권한

read

읽을 권한

write

쓸 권한 (편집)

execute

실행할 권한

뭘 할 수 있는가: permission - 설정 방법

```
derick321@goorm: /home/antony-jeong/music_team
derick321@goorm: /home/antony-jeong/music_team$ ls -la memo
-rw-rw-r-- 1 antony-jeong antony-jeong 303 Jan 11 03:25 memo
derick321@goorm: /home/antony-jeong/music_team$ cat memo
The day you couldn't come to my birthday party
The day without you he was always
The day he showed up dressed to the nines
Heard him say

We can get whatever you want
Baby, say the words and I'm down
All I need is you on my side
We can go whenever you like
Surprise! Mother, father.
Now, where are you?
derick321@goorm: /home/antony-jeong/music_team$
```

13% 8.7 GB



```
antony-jeong@goorm: ~/music_team$ chmod o-r memo
antony-jeong@goorm: ~/music_team$
```

ssh -zsh ~ 20% 8.7 GB



```
derick321@goorm: /home/antony-jeong/music_team
derick321@goorm: /home/antony-jeong/music_team$ ls -la memo
-rw-rw---- 1 antony-jeong antony-jeong 303 Jan 11 03:25 memo
derick321@goorm: /home/antony-jeong/music_team$ cat memo
cat: memo: Permission denied
derick321@goorm: /home/antony-jeong/music_team$
```

bellssng ssh -zsh ~ 19% 8.7 GB

chmod

권한을 바꿈.

[ugoa][+-][rwx]

u: owner user
g: group
o: others
a: all

+: 권한 주기
-: 권한 뺏기

다른 사용법도 있음. Read man pages

뭐든지 할 수 있는 root



A screenshot of a terminal window on a Mac OS X desktop. The window title bar shows three colored dots (red, yellow, green) and the text "root@goorm: ~". The terminal prompt is "root@goorm:~#". The user types "pwd" and receives the output "/root". The terminal prompt changes to "root@goorm:~#". At the bottom right of the terminal window, there is a status bar with the text "8.7 GB" and a progress bar.

```
root@goorm:~# pwd
/root
root@goorm:~#
```

특수한 user인 root user는 뭐든지 할 수 있음.
(비밀번호도 기존 비밀번호 안 써도 바꿀 수 있음. 다른 사람 것도.)

prompt가 #으로 끝남.
home directory가 /root임. (/home에 있지 않음.)

잠깐 빌리자: sudo - 사용법과 조건

super user do ~~just shut up and do~~

sudo는 root의 권한을 빌려 커マン드를 실행함.

사용법

- (1) 커マン드 앞에 sudo를 붙여 커マン드를 실행한다.
- (2) 비밀번호를 입력하라고 하면 입력한다. (sudo를 사용한지 얼마 안 됐으면 안 물어볼 수도 있음.)

각 user는 sudo를 쓸 수 있는지 아닌지 /etc/sudoers에 써 있는데,
그건 sudo를 쓸 수 있는 user가 바꿔줄 수 있음.

잠깐 빌리자: sudo - 예시: 파일 읽기

super user do ~~just shut up and do~~

```

antony-jeong@goorm:~$ ls -la /etc/sudoers
-r--r----- 1 root root 786 Jan  7 17:10 /etc/sudoers
antony-jeong@goorm:~$ cat /etc/sudoers
cat: /etc/sudoers: Permission denied
antony-jeong@goorm:~$ sudo cat /etc/sudoers
[sudo] password for antony-jeong:
#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults      env_reset
Defaults      mail_badpass
Defaults      secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin"

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root      ALL=(ALL:ALL) ALL
antony-jeong    ALL=(ALL:ALL) ALL

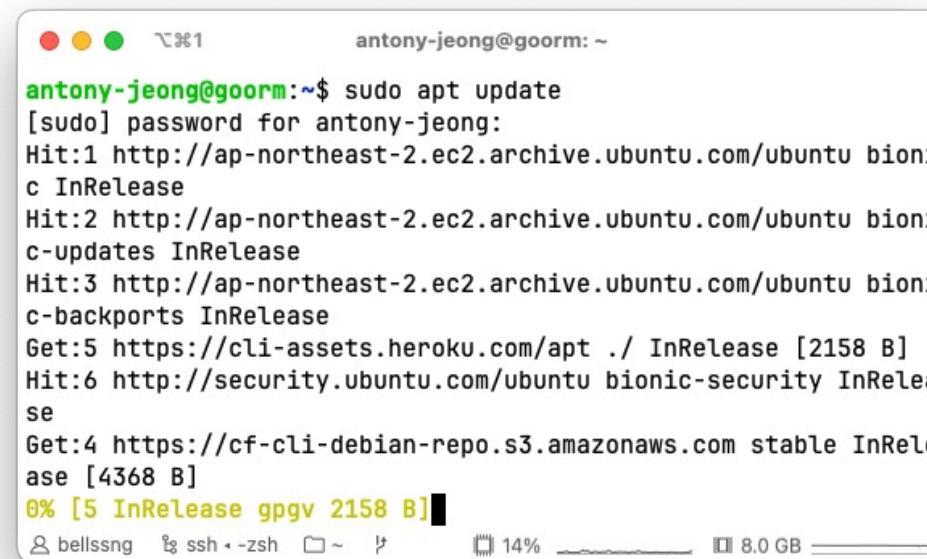
# Members of the admin group may gain root privileges

```

읽기 권한이 없었으나
sudo를 통해 읽었음.

잠깐 빌리자: sudo - 예시: 패키지 관리자

super user do ~~just shut up and do~~



```
antony-jeong@goorm:~$ sudo apt update
[sudo] password for antony-jeong:
Hit:1 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu bionic InRelease
Hit:2 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu bionic-updates InRelease
Hit:3 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu bionic-backports InRelease
Get:4 https://cli-assets.heroku.com/apt ./ InRelease [2158 B]
Hit:5 http://security.ubuntu.com/ubuntu bionic-security InRelease
Get:6 https://cf-cli-debian-repo.s3.amazonaws.com stable InRelease [4368 B]
0% [5 InRelease gpgv 2158 B]
```

The screenshot shows a terminal window with a light gray background and a dark gray border. The title bar at the top has three colored circles (red, yellow, green) and the text "antony-jeong@goorm: ~". The main area of the terminal contains the command "sudo apt update" followed by its output. The output shows several hits from the Ubuntu repositories and one get command from Heroku's apt repository. The progress bar at the bottom is at 0% completion, showing "[5 InRelease gpgv 2158 B]". Below the terminal window, there is a status bar with icons for user, terminal type, current directory (~), battery level (14%), and disk usage (8.0 GB).

apt

Ubuntu에서는 프로그램이나 라이브러리를 (웬만하면) 이것으로 설치함.
apt로 원가를 설치/삭제하고 싶으면 root 권한이 필요함.

비밀번호 바꾸기: passwd

password



antony-jeong@goorm:~/music_team\$ passwd
Changing password for antony-jeong.
(current) UNIX password:
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
antony-jeong@goorm:~/music_team\$

passwd

계정의 비밀번호를 바꿈.

- (1) 기존 비밀번호를 입력
- (2) 새로운 비밀번호를 입력
- (3) 2에서 입력한 비밀번호를 다시 입력

user 생성과 삭제: adduser, deluser

add user delete user

adduser 여러분의 계정을 만들었던 방법
deluser 계정을 삭제함.

root 권한을 가지고 뭔가 할 때가 온다면 한 번씩 해 보세요~
어떻게 쓰냐고요? Read man pages

user 전환: su

switch user

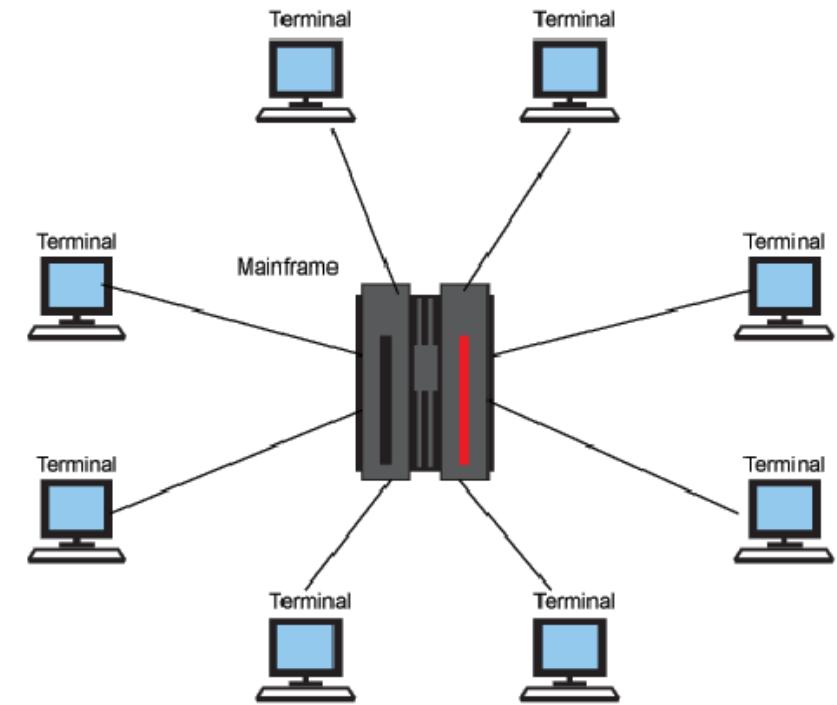


```
● ● ●  ~%1 antony-jeong@goorm: ~
antony-jeong@goorm:~$ su derick321
Password:
derick321@goorm:/home/antony-jeong$ cd ~
derick321@goorm:~$ exit
exit
antony-jeong@goorm:~$ █
```

The screenshot shows a terminal window with a light gray background and a dark gray border. At the top left, there are three colored dots (red, yellow, green). The prompt is ~%1 antony-jeong@goorm: ~. The user types 'su derick321' followed by their password. After logging in, the prompt changes to derick321@goorm:/home/antony-jeong\$. They then type 'cd ~' to change to their home directory. Finally, they type 'exit' to log back out, and the prompt returns to antony-jeong@goorm:~\$. At the bottom of the terminal window, there is a status bar with icons for file, search, battery (19%), and memory (8.2 GB).

다른 user로 로그인된 shell을 실행함.

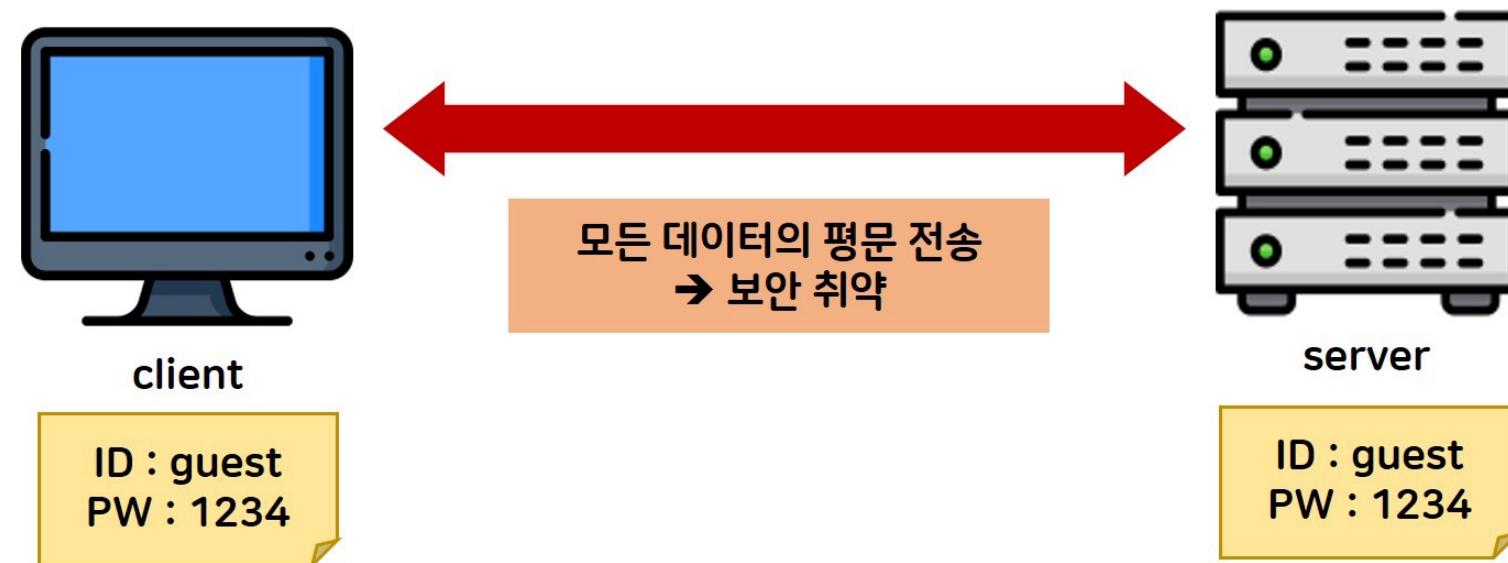
원격 접속 - 우리는 이걸 왜 하는가



전산 과목들이나 연구실, 관련 직종에서 제공해주는 개발 환경을 이용하는 방식은
옛날 컴퓨터실의 이것과 매우 흡사함.

실제로 연산을 하는 컴퓨터는 원격에 놓고,
내 기계로는 접속해서 입력과 출력만 하는 것

원격 접속 - 재미있는 옛날 이야기: telnet



원격 접속 방법: ssh - 비밀번호 입력하기

secure shell

ssh <username>@<hostname>

ssh -p <port> <username>@<hostname>

원격 접속 방법: ssh - 키 등록하기

secure shell

비밀번호를 쓰는 대신에

ssh-keygen 등으로 나만의 열쇠를 만들어 비밀번호 입력 없이 접속할 수 있음!
방법은 알아서 찾아보자~ (키워드: ssh 키 등록하기)

시연은 함 해 줄게요

원격 파일 다루기 - 복사하기: scp

secure copy

scp <username>@<hostname>:<path> <path>
remote에 있는 파일을 local로 가져오기

scp <path> <username>@<hostname>:<path>
local에 있는 파일을 remote에 보내기

원격 파일 다루기 - CLI tool: sftp

secure file transfer protocol

ssh 기반으로 파일을 다루는 CLI tool임.

잘 안 쓰기는 하는데 궁금하면 알아서 찾아보자~

환경 변수 - 이게 뭔데

shell에서 전역 변수처럼 설정해 놓고 쓰는 값.

새로운 프로그램을 설치했을 때/만들었을 때:

아무데서나 이 프로그램 이름을 쓰면 실행하게 하고 싶은데…

shell은 PATH라는 환경 변수를 이용해서 프로그램을 찾는구나!

→ PATH에 프로그램의 위치를 추가

환경 변수 - 어떻게 쓰는데 - 설정하기

`export <var>`

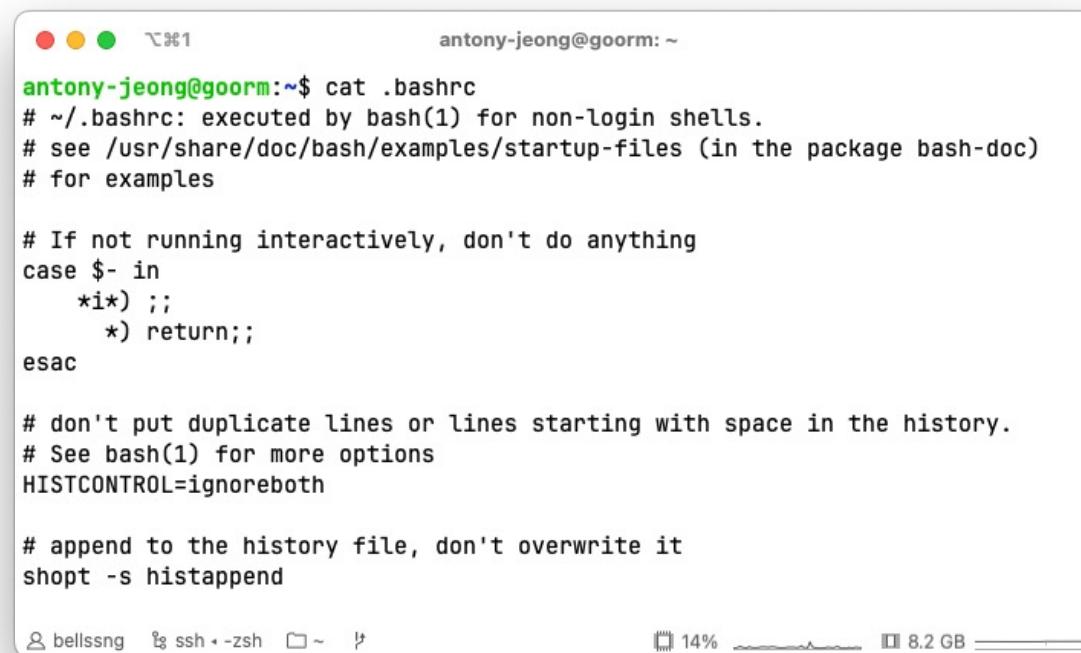
shell 변수를 환경 변수로 만듦.

대입까지 한 번에 쓸 수도 있음. (`export <var>=<value>`)

그 안에서 새로운 shell를 열어도 보존됨.

시연해볼게요~

환경 변수 - 어떻게 쓰는데 - 영구 설정하기



```

antony-jeong@goorm:~$ cat .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
case $- in
    *i*) ;;
    *) return;;
esac

# don't put duplicate lines or lines starting with space in the history.
# See bash(1) for more options
HISTCONTROL=ignoreboth

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

antony-jeong@goorm:~$ 

```

The screenshot shows a terminal window on a Mac OS X desktop. The title bar says "antony-jeong@goorm: ~". The command "cat .bashrc" is run, displaying the contents of the .bashrc file. The file contains standard bash startup code, including history control settings and a case statement for interactive shells. The bottom of the terminal shows the user's name, the command run, and the prompt again.

run commands

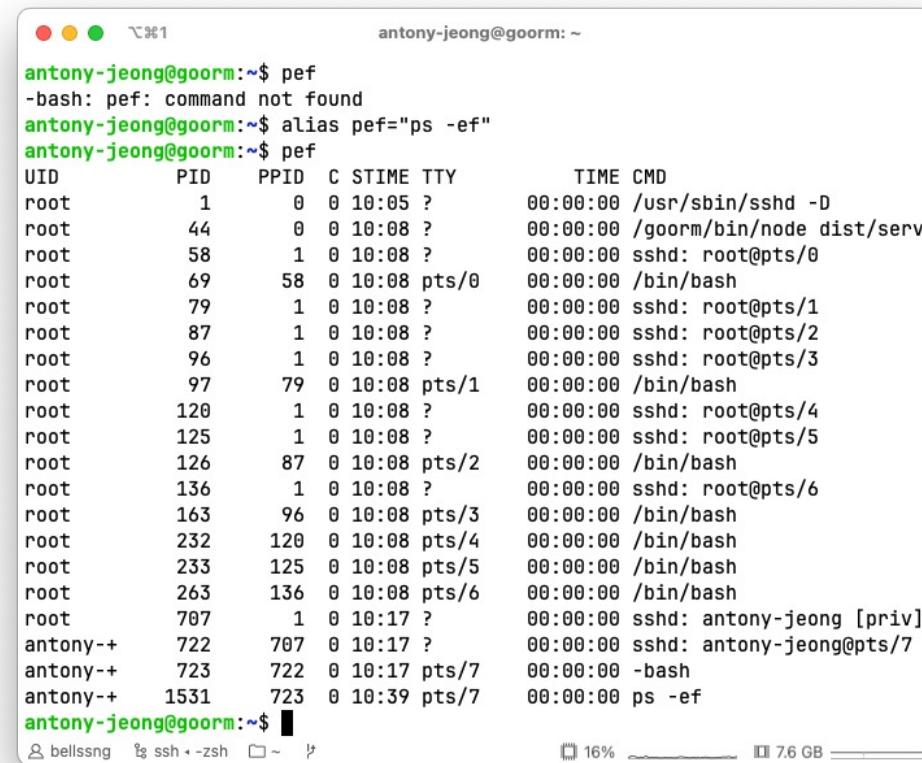
내가 bash에 로그인할 때마다 나의 home에 있는 ~/.bashrc가 실행됨.

→ ~/.bashrc에 추가해놓으면 앞으로 로그인할 때 환경 변수 사용 가능

몰라도 되지만
알면 편한 것들



커맨드 별다줄 - alias



The screenshot shows a terminal window with the following session:

```
antony-jeong@goorm:~$ pef
-bash: pef: command not found
antony-jeong@goorm:~$ alias pef="ps -ef"
antony-jeong@goorm:~$ pef
UID      PID    PPID   C STIME TTY          TIME CMD
root        1      0  0 10:05 ?
root        44     0  0 10:08 ?
root        58     1  0 10:08 ?
root        69     58  0 10:08 pts/0
root        79     1  0 10:08 ?
root        87     1  0 10:08 ?
root        96     1  0 10:08 ?
root        97     79  0 10:08 pts/1
root       120     1  0 10:08 ?
root       125     1  0 10:08 ?
root       126     87  0 10:08 pts/2
root       136     1  0 10:08 ?
root       163     96  0 10:08 pts/3
root       232    120  0 10:08 pts/4
root       233    125  0 10:08 pts/5
root       263    136  0 10:08 pts/6
root       707     1  0 10:17 ?
antony-+  722    707  0 10:17 ?
antony-+  723    722  0 10:17 pts/7
antony-+ 1531    723  0 10:39 pts/7
antony-jeong@goorm:~$
```

The terminal window also displays system status at the bottom, including battery level (16%) and memory usage (7.6 GB).

커맨드에 flag(s), option(s), argument(s)를 이미 적용한 걸 줄여서 쓸 수 있음.

커맨드 어디서 왔니 - type



```
● ● ●  ↵%1 antony-jeong@goorm: ~  
antony-jeong@goorm:~$ type cd  
cd is a shell builtin  
antony-jeong@goorm:~$ type pwd  
pwd is a shell builtin  
antony-jeong@goorm:~$ type vi  
vi is /usr/bin/vi  
antony-jeong@goorm:~$ █  
█ 13% ━━━━━━ █ 7.6 GB ━━━━━━
```

커맨드가 shell의 built-in인지, binary인지 등을 알려줌.

접누있? - who, w

The image shows two terminal windows side-by-side. The left window, titled 'antony-jeong@goorm:~\$' (ssh), displays the output of several 'type' commands for shell builtins like 'cd', 'ls', and 'pwd', followed by the output of the 'who' command, which lists multiple users (root, antony-jeong, derick321) with their respective terminal types (pts/0-7, pts/8) and login times (Jan 11 10:08-10:17). The right window, also titled 'antony-jeong@goorm:~\$' (ssh), shows the standard Ubuntu welcome message and system information, including documentation links and a note about minimizing the system. It also displays the 'derick321@goorm:~\$' prompt.

```

antony-jeong@goorm:~$ type cd
cd is a shell builtin
antony-jeong@goorm:~$ type ls
ls is aliased to `ls --color=auto'
antony-jeong@goorm:~$ type cd
cd is a shell builtin
antony-jeong@goorm:~$ type pwd
pwd is a shell builtin
antony-jeong@goorm:~$ type vi
vi is /usr/bin/vi
antony-jeong@goorm:~$ who
root    pts/0      Jan 11 10:08 (172.17.0.1)
root    pts/1      Jan 11 10:08 (172.17.0.1)
root    pts/2      Jan 11 10:08 (172.17.0.1)
root    pts/3      Jan 11 10:08 (172.17.0.1)
root    pts/5      Jan 11 10:08 (172.17.0.1)
root    pts/4      Jan 11 10:08 (172.17.0.1)
root    pts/6      Jan 11 10:08 (172.17.0.1)
antony-jeong pts/7      Jan 11 10:17 (110.76.108.151)
derick321 pts/8      Jan 11 10:41 (110.76.108.151)
antony-jeong@goorm:~$ [REDACTED]

```

```

antony-jeong@goorm:~$ 
derick321@goorm:~$ [REDACTED]

```

현재 어떤 user가 접속 중인지 보여줌.

어디로 가야 하오 - whereis, which

```

antony-jeong@goorm: ~/music_team
antony-jeong@goorm:~/music_team$ whereis ls
ls: /bin/ls /usr/share/man/man1/ls.1.gz
antony-jeong@goorm:~/music_team$ which ls
/bin/ls
antony-jeong@goorm:~/music_team$ whereis --help

Usage:
  whereis [options] [-BMS <dir>... -f] <name>
Locate the binary, source, and manual-page files for a command.

Options:
  -b      search only for binaries
  -B <dirs> define binaries lookup path
  -m      search only for manuals and infos
  -M <dirs> define man and info lookup path
  -s      search only for sources
  -S <dirs> define sources lookup path
  -f      terminate <dirs> argument list
  -u      search for unusual entries
  -l      output effective lookup paths

  -h, --help    display this help
  -V, --version display version

For more details see whereis(1).
antony-jeong@goorm:~/music_team$ █

```

bellssng 1g ssh -zsh ~ ↵ 19% 8.0 GB

command의 실체(바이너리 등)가 어디에 있는지 알아봄.

whereis

후보를 나열해 줌.

which

지금 명령하면 선택되는 기본 후보를 알려줌.

disk usage

파일&디렉터리 용량 - du

파일이나 디렉터리의 사용 용량을 보여 줌.



The screenshot shows a terminal window on a Mac OS X desktop. The title bar says "disk usage". The command entered is "du -h /*". The output lists various files and directories with their sizes. The bottom of the window shows system status: user "bellssng", session "ssh -zsh", and battery level "22% 8.2 GB".

```
antony-jeong@goorm: ~/music_team
antony-jeong@goorm:~/music_team$ du -h ./*
4.0K ./error
20K ./index.html
4.0K ./list
4.0K ./list1
4.0K ./list2
4.0K ./memo
4.0K ./memo.diff
4.0K ./memo_modified
4.0K ./memo_version_1
4.0K ./result
4.0K ./result_and_error
4.0K ./weird
antony-jeong@goorm:~/music_team$
```

bellssng ssh -zsh ~ ⚡ 22% 8.2 GB

텍스트 가공 - cut, sed, awk, printf

강력한 텍스트 가공 도구들이니 bash로 뭔가를 만들고 싶다면 찾아보세요~

나도 잘은 모름 Read man pages

no hang-up

내가 떠나도 일을 계속하게 - nohup

보통은 shell을 나가면 background에서 하던 일도 모두 없어지는데,
앞에 nohup을 붙임으로써 떠나도 일을 계속하라고 할 수 있음.

table of processes

시스템 실시간 모니터링 - top, htop

Hisham Muhammad가 개조한 top

```

● ● ● ၮၯ၃
antonny-jeong@goom: ~

top - 09:48:55 up 58 days, 3:32, 8 users, load average: 1.22, 1.50, 1.52
Tasks: 20 total, 1 running, 19 sleeping, 0 stopped, 0 zombie
%Cpu(s): 32.2 us, 4.9 sy, 0.0 ni, 61.8 id, 0.8 wa, 0.0 hi, 0.3 si, 0.0
KiB Mem : 15986228 total, 7811688 free, 3156992 used, 5017548 buff/cache
KiB Swap: 8388604 total, 8384508 free, 4096 used. 12450228 avail Mem

PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+
 1 root 20 0 72308 6140 5384 S 0.0 0.0 0:00.05
 42 root 20 0 566148 35224 27832 S 0.0 0.2 0:00.20
 58 root 20 0 108000 7204 6196 S 0.0 0.0 0:00.01
 60 root 20 0 108000 7240 6228 S 0.0 0.0 0:00.01
 83 root 20 0 108000 7244 6232 S 0.0 0.0 0:00.01
 91 root 20 0 108000 7220 6204 S 0.0 0.0 0:00.01
101 root 20 0 108000 7116 6104 S 0.0 0.0 0:00.00
117 root 20 0 108000 7212 6200 S 0.0 0.0 0:00.01
128 root 20 0 108000 7188 6180 S 0.0 0.0 0:00.01
188 root 20 0 20072 3900 3396 S 0.0 0.0 0:00.01
189 root 20 0 20072 3968 3460 S 0.0 0.0 0:00.01
190 root 20 0 20072 3928 3416 S 0.0 0.0 0:00.00
191 root 20 0 19944 3776 3420 S 0.0 0.0 0:00.00
192 root 20 0 20072 4040 3532 S 0.0 0.0 0:00.01
211 root 20 0 19944 3680 3324 S 0.0 0.0 0:00.00

bellssng ssh -zsh ~ ↵ 16% 8.2 GB

```

top

```

● ● ● ၮၯ၃
antonny-jeong@goom: ~

1 [|||||] 38.1% Tasks: 20, 6 thr; 1 running
2 [|||||] 43.3% Load average: 1.50 1.55 1.54
3 [|||||] 36.7% Uptime: 58 days, 03:32:09
4 [|||] 19.2%
Mem[|||||] 3.11G/15.2G
Swp[] 4.00M/8.00G

PID USER PRI NI VIRT RES SHR S CPU% MEM% TIME+ Command
 1 root 20 0 72308 6140 5384 S 0.0 0.0 0:00.05 /usr/sbin/s
 52 root 20 0 552M 35224 27832 S 0.0 0.2 0:00.00 /goorm/bin/
 53 root 20 0 552M 35224 27832 S 0.0 0.2 0:00.00 /goorm/bin/
 54 root 20 0 552M 35224 27832 S 0.0 0.2 0:00.01 /goorm/bin/
 55 root 20 0 552M 35224 27832 S 0.0 0.2 0:00.01 /goorm/bin/
 56 root 20 0 552M 35224 27832 S 0.0 0.2 0:00.00 /goorm/bin/
 57 root 20 0 552M 35224 27832 S 0.0 0.2 0:00.00 /goorm/bin/
 42 root 20 0 552M 35224 27832 S 0.0 0.2 0:00.20 /goorm/bin/
 58 root 20 0 105M 7204 6196 S 0.0 0.0 0:00.01 sshd: root@...
 60 root 20 0 105M 7240 6228 S 0.0 0.0 0:00.01 sshd: root@...
 83 root 20 0 105M 7244 6232 S 0.0 0.0 0:00.01 sshd: root@...
 91 root 20 0 105M 7220 6204 S 0.0 0.0 0:00.01 sshd: root@...

F1 Help F2 Setup F3 Search F4 Filter F5 Tree F6 SortBy F7 Nice -F8 Nice +F9 Kill F10 Q
bellssng ssh -zsh ~ ↵ 14% 8.3 GB

```

htop

그 외

df
find
service
netstat
ifconfig
perl
cmp
diff3
comm
rev

등등...

맺는 말

사용할수록 편해지는 shell [CLI] 많이많이 연습합시다.

man pages는 사실 너무 읽기 힘든 경우도 많으니 구글링을 먼저 해 보세요.
그러나 시프 할 때 알아봐야 하는 C stdlib 함수들은 man pages가 진짜 편함

