김연주

▼ 1주차

강의는 linux 기준 코드 제공

git bash, wsl 이용

▼ BASH 코드

```
kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ echo hello
hello

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ echo Hello World
Hello World

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ echo Hello\ World
Hello World

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ echo $PATH
/mingw64/bin:/usr/bin:/c/Users/kjhoe/bin:/c/Windows/system32:/c/Windo

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ which echo
/usr/bin/echo

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ pwd
/c/Users/kjhoe

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ ls
 anaconda3
 AppData
'Application Data'
 Contacts
```

```
 Cookies
 Custom_NeuralProphet_Logs
 Documents
 Downloads
 Favorites
 Links
'Local Settings'
 Music
'My Documents'
 NetHood
 NTUSER.DAT
 ntuser.dat.LOG1
 ntuser.dat.LOG2
 NTUSER.DAT{a2332f18-cdbf-11ec-8680-002248483d79}.TM.blf
 NTUSER.DAT{a2332f18-cdbf-11ec-8680-002248483d79}.TMContainer0
 NTUSER.DAT{a2332f18-cdbf-11ec-8680-002248483d79}.TMContainer0
 ntuser.ini
 OneDrive
 PATH
 PrintHood
 Recent
'Saved Games'
 Searches
 SendTo
 Templates
 Videos
'시작 메뉴'

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ ls -l
total 22365
drwxr-xr-x 1 kjhoe 197609       0 Oct  6 12:54  anaconda3
drwxr-xr-x 1 kjhoe 197609       0 Dec  1  2023  AppData
lrwxrwxrwx 1 kjhoe 197609      30 Dec  1  2023 'Application Data' → /c/Us
drwxr-xr-x 1 kjhoe 197609       0 Dec  1  2023  Contacts
lrwxrwxrwx 1 kjhoe 197609      58 Dec  1  2023  Cookies → /c/Users/kjhoe
drwxr-xr-x 1 kjhoe 197609       0 Jan 13 17:13 'cs 1강'
drwxr-xr-x 1 kjhoe 197609       0 Nov 15 01:44  Custom_NeuralProphet_Lo
drwxr-xr-x 1 kjhoe 197609       0 Dec  2  2023  Documents
drwxr-xr-x 1 kjhoe 197609       0 Jan 13 16:26  Downloads
```

```
drwxr-xr-x 1 kjhoe 197609       0 Dec  1  2023  Favorites
-rw-r--r-- 1 kjhoe 197609       6 Jan 13 17:17  hello.txt
-rw-r--r-- 1 kjhoe 197609      12 Jan 13 17:19  hello2.txt
drwxr-xr-x 1 kjhoe 197609       0 Dec  1  2023  Links
lrwxrwxrwx 1 kjhoe 197609      28 Dec  1  2023 'Local Settings' → /c/Users
-rw-r--r-- 1 kjhoe 197609      51 Jan 13 17:21  ls.txt
drwxr-xr-x 1 kjhoe 197609       0 Aug 25 12:46  Music
lrwxrwxrwx 1 kjhoe 197609      24 Dec  1  2023 'My Documents' → /c/Use
lrwxrwxrwx 1 kjhoe 197609      66 Dec  1  2023  NetHood → '/c/Users/kjhc
-rw-r--r-- 1 kjhoe 197609 14417920 Jan 12 23:27  NTUSER.DAT
-rw-r--r-- 1 kjhoe 197609  3629056 Dec  1  2023  ntuser.dat.LOG1
-rw-r--r-- 1 kjhoe 197609  3629056 Dec  1  2023  ntuser.dat.LOG2
-rw-r--r-- 1 kjhoe 197609    65536 Dec  1  2023  NTUSER.DAT{a2332f18-c
-rw-r--r-- 1 kjhoe 197609   524288 Dec  1  2023  NTUSER.DAT{a2332f18-
-rw-r--r-- 1 kjhoe 197609   524288 Dec  1  2023  NTUSER.DAT{a2332f18-
-rw-r--r-- 1 kjhoe 197609      20 Dec  1  2023  ntuser.ini
drwxr-xr-x 1 kjhoe 197609       0 Jan 13 14:25  OneDrive
-rw-r--r-- 1 kjhoe 197609      27 Jan 13 15:44  PATH
lrwxrwxrwx 1 kjhoe 197609      66 Dec  1  2023  PrintHood → '/c/Users/kjh
lrwxrwxrwx 1 kjhoe 197609      55 Dec  1  2023  Recent → /c/Users/kjhoe/
drwxr-xr-x 1 kjhoe 197609       0 Dec  1  2023 'Saved Games'
drwxr-xr-x 1 kjhoe 197609       0 Dec  2  2023  Searches
lrwxrwxrwx 1 kjhoe 197609      55 Dec  1  2023  SendTo → /c/Users/kjhoe/
lrwxrwxrwx 1 kjhoe 197609      58 Dec  1  2023  Templates → /c/Users/kjh
drwxr-xr-x 1 kjhoe 197609       0 Dec  4  2023  Videos
lrwxrwxrwx 1 kjhoe 197609      59 Dec  1  2023 '시작 메뉴' -> '/c/Users/kjh

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ mkdir "CS 1강"

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ ls
 anaconda3
 AppData
'Application Data'
 Contacts
 Cookies
'CS 1강'
 Custom_NeuralProphet_Logs
 Documents
```

```
 Downloads
 Favorites
 Links
'Local Settings'
 Music
'My Documents'
 NetHood
 NTUSER.DAT
 ntuser.dat.LOG1
 ntuser.dat.LOG2
 NTUSER.DAT{a2332f18-cdbf-11ec-8680-002248483d79}.TM.blf
 NTUSER.DAT{a2332f18-cdbf-11ec-8680-002248483d79}.TMContainer0
 NTUSER.DAT{a2332f18-cdbf-11ec-8680-002248483d79}.TMContainer0
 ntuser.ini
 OneDrive
 PATH
 PrintHood
 Recent
'Saved Games'
 Searches
 SendTo
 Templates
 Videos
'시작 메뉴'

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ echo hello > hello.txt

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ cat < hello.txt
hello

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ cat < hello.txt > hello2.txt

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ cat hello2.txt
hello

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
```

```
$ cat hello.txt >> hello2.txt

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ cat hello2.txt
hello
hello

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ ls -l /
total 5804
drwxr-xr-x  1 kjhoe 197609       0 Jan 13 16:28 bin
drwxr-xr-x  1 kjhoe 197609       0 Jan 13 16:28 cmd
drwxr-xr-x  1 kjhoe 197609       0 Jan 13 16:28 dev
drwxr-xr-x  1 kjhoe 197609       0 Jan 13 16:28 etc
-rwxr-xr-x  1 kjhoe 197609  139144 Nov 25 12:16 git-bash.exe
-rwxr-xr-x  1 kjhoe 197609  138616 Nov 25 12:16 git-cmd.exe
-rw-r--r--  1 kjhoe 197609   18765 Nov 25 12:27 LICENSE.txt
drwxr-xr-x  1 kjhoe 197609       0 Jan 13 16:28 mingw64
dr-xr-xr-x 11 kjhoe 197609       0 Jan 14 12:04 proc
-rw-r--r--  1 kjhoe 197609  272435 Nov 25 12:27 ReleaseNotes.html
drwxr-xr-x  1 kjhoe 197609       0 Jan 14 12:01 tmp
-rw-r--r--  1 kjhoe 197609 1319350 Jan 13 16:28 unins000.dat
-rwxr-xr-x  1 kjhoe 197609 3384040 Jan 13 16:26 unins000.exe
-rw-r--r--  1 kjhoe 197609   24183 Jan 13 16:28 unins000.msg
drwxr-xr-x  1 kjhoe 197609       0 Jan 13 16:28 usr

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ ls -l / | tail -n1
drwxr-xr-x  1 kjhoe 197609       0 Jan 13 16:28 usr

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ ls -l / | tail -n1 > ls.txt

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ cat ls.txt
drwxr-xr-x  1 kjhoe 197609       0 Jan 13 16:28 usr

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ curl --head --silent google.com
HTTP/1.1 301 Moved Permanently
```

```
Location: http://www.google.com/
Content-Type: text/html; charset=UTF-8
Content-Security-Policy-Report-Only: object-src 'none';base-uri 'self';scr
Date: Tue, 14 Jan 2025 03:06:21 GMT
Expires: Thu, 13 Feb 2025 03:06:21 GMT
Cache-Control: public, max-age=2592000
Server: gws
Content-Length: 219
X-XSS-Protection: 0
X-Frame-Options: SAMEORIGIN

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ curl --head --silent google.com | grep -i content-length
Content-Length: 219

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ curl --head --silent google.com | grep -i content-length | cut --del
imiter=' ' -f2
219
```

▼ WSL 코드

```
kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ wsl
su: user kjhoekjy does not exist or the user entry does not contain all the r

root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe#  su - kjhoeakjy
To run a command as administrator (user "root"), use "sudo <command>"
See "man sudo_root" for details.

kjhoeakjy@BOOK-EJ6QAG1NJL:~$ /bin/echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/l
kjhoeakjy@BOOK-EJ6QAG1NJL:~$ pwd
/home/kjhoeakjy
kjhoeakjy@BOOK-EJ6QAG1NJL:~$ cd /home
kjhoeakjy@BOOK-EJ6QAG1NJL:/home$ ls
```

```
kjhoeakjy
kjhoeakjy@BOOK-EJ6QAG1NJL:/home$ cd kjhoeakjy/
kjhoeakjy@BOOK-EJ6QAG1NJL:~$ ls
kjhoeakjy@BOOK-EJ6QAG1NJL:~$ pwd
/home/kjhoeakjy
kjhoeakjy@BOOK-EJ6QAG1NJL:~$ cd -
/home
kjhoeakjy@BOOK-EJ6QAG1NJL:/home$ cd -
/home/kjhoeakjy
kjhoeakjy@BOOK-EJ6QAG1NJL:~$ 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      with -l, scale sizes by SIZE when printing them;
                             e.g., '--block-size=M'; 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
                             change 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]         color the output WHEN; more info below
  -d, --directory            list directories themselves, not their contents
  -D, --dired                generate output designed for Emacs' dired mode
  -f                         list all entries in directory order
  -F, --classify[=WHEN]      append indicator (one of */=>@|) to entries WH
      --file-type            likewise, except do not append '*'
      --format=WORD          across -x, commas -m, horizontal -x, long -l,
                             single-column -1, verbose -l, vertical -C

      --full-time            like -l --time-style=full-iso
  -g                         like -l, but do not list owner
```

```
  --group-directories-first
                group directories before files;
                can be augmented with a --sort option, but any
                use of --sort=none (-U) disables grouping

-G, --no-group          in a long listing, don't print group names
-h, --human-readable    with -l and -s, print sizes like 1K 234M 2G etc.
  --si              likewise, but use powers of 1000 not 1024
-H, --dereference-command-line
                follow symbolic links listed on the command line
  --dereference-command-line-symlink-to-dir
                follow each command line symbolic link
                that points to a directory

  --hide=PATTERN      do not list implied entries matching shell PATTE
                (overridden by -a or -A)

  --hyperlink[=WHEN]    hyperlink file names WHEN
  --indicator-style=WORD
                append indicator with style WORD to entry names:
                none (default), slash (-p),
                file-type (--file-type), classify (-F)

-i, --inode             print the index number of each file
-I, --ignore=PATTERN    do not list implied entries matching shell PATTE
-k, --kibibytes         default to 1024-byte blocks for file system usage;
                used only with -s and per directory totals

-l                      use a long listing format
-L, --dereference       when showing file information for a symbolic
                link, show information for the file the link
                references rather than for the link itself

-m                      fill width with a comma separated list of entries
-n, --numeric-uid-gid   like -l, but list numeric user and group IDs
-N, --literal           print entry names without quoting
-o                      like -l, but do not list group information
-p, --indicator-style=slash
                append / indicator to directories
-q, --hide-control-chars   print ? instead of nongraphic characters
```

```
  --show-control-chars   show nongraphic characters as-is (the default,
                         unless program is 'ls' and output is a terminal)

-Q, --quote-name          enclose entry names in double quotes
  --quoting-style=WORD   use quoting style WORD for entry names:
                         literal, locale, shell, shell-always,
                         shell-escape, shell-escape-always, c, escape
                         (overrides QUOTING_STYLE environment variable)

-r, --reverse             reverse order while sorting
-R, --recursive           list subdirectories recursively
-s, --size                print the allocated size of each file, in blocks
-S                        sort by file size, largest first
  --sort=WORD            sort by WORD instead of name: none (-U), size (-
                         time (-t), version (-v), extension (-X), width

  --time=WORD            select which timestamp used to display or sort;
                           access time (-u): atime, access, use;
                           metadata change time (-c): ctime, status;
                           modified time (default): mtime, modification;
                           birth time: birth, creation;
                         with -l, WORD determines which time to show;
                         with --sort=time, sort by WORD (newest first)

  --time-style=TIME_STYLE
                         time/date format with -l; see TIME_STYLE below
-t                        sort by time, newest first; see --time
-T, --tabsize=COLS        assume tab stops at each COLS instead of 8
-u                         with -lt: sort by, and show, access time;
                         with -l: show access time and sort by name;
                         otherwise: sort by access time, newest first

-U                         do not sort; list entries in directory order
-v                        natural sort of (version) numbers within text
-w, --width=COLS          set output width to COLS.  0 means no limit
-x                        list entries by lines instead of by columns
-X                        sort alphabetically by entry extension
-Z, --context             print any security context of each file
  --zero                 end each output line with NUL, not newline
-1                        list one file per line
```

```
      --help        display this help and exit
      --version     output version information and exit

The SIZE argument is an integer and optional unit (example: 10K is 10*102
Units are K,M,G,T,P,E,Z,Y,R,Q (powers of 1024) or KB,MB,... (powers of 10(
Binary prefixes can be used, too: KiB=K, MiB=M, and so on.

The TIME_STYLE argument can be full-iso, long-iso, iso, locale, or +FORM
FORMAT is interpreted like in date(1).  If FORMAT is FORMAT1<newline>FO
then FORMAT1 applies to non-recent files and FORMAT2 to recent files.
TIME_STYLE prefixed with 'posix-' takes effect only outside the POSIX loc
Also the TIME_STYLE environment variable sets the default style to use.

The WHEN argument defaults to 'always' and can also be 'auto' or 'never'

Using color to distinguish file types is disabled both by default and
with --color=never.  With --color=auto, ls emits color codes only when
standard output is connected to a terminal.  The LS_COLORS environment
variable can change the settings.  Use the dircolors(1) command to set it.

Exit status:
 0  if OK,
 1  if minor problems (e.g., cannot access subdirectory),
 2  if serious trouble (e.g., cannot access command-line argument).

GNU coreutils online help: <https://www.gnu.org/software/coreutils/>
Report any translation bugs to <https://translationproject.org/team/>
Full documentation <https://www.gnu.org/software/coreutils/ls>
or available locally via: info '(coreutils) ls invocation'

kjhoeakjy@BOOK-EJ6QAG1NJL:~$ cd /sys
kjhoeakjy@BOOK-EJ6QAG1NJL:/sys$ ls
block  bus  class  dev  devices  firmware  fs  kernel  module
kjhoeakjy@BOOK-EJ6QAG1NJL:/sys$ cd class/
kjhoeakjy@BOOK-EJ6QAG1NJL:/sys/class$ ls
bdi     input    net          rtc          uio
block   iommu    pci_bus      scsi_device  vc
bsg     ipvtap   phy          scsi_disk    vfio
cuse    macvtap  power_supply  scsi_generic  virtio-ports
devlink  mem     ppp          scsi_host    vtconsole
```

```
drm     misc    pps         thermal
hidraw  nd      ptp         tty
kjhoeakjy@BOOK-EJ6QAG1NJL:/sys/class$ sudo find -iname '*brightness
[sudo] password for kjhoeakjy:
kjhoeakjy@BOOK-EJ6QAG1NJL:/sys/class$ sudo find -iname '*brightness

#wsl 환경에서는 backlight, leds 부
```

▼ Exercise (#1,2,3)

```
#1 - wsl
kjhoeakjy@BOOK-EJ6QAG1NJL:~$ mkdir /tmp/missing
kjhoeakjy@BOOK-EJ6QAG1NJL:~$ ls /tmp
missing
snap-private-tmp
systemd-private-6395cfe18c6c4d389c3e1e2932efc09d-systemd-logind.
systemd-private-6395cfe18c6c4d389c3e1e2932efc09d-systemd-resolve
systemd-private-6395cfe18c6c4d389c3e1e2932efc09d-systemd-timesyn
systemd-private-6395cfe18c6c4d389c3e1e2932efc09d-wsl-pro.service-

#2 - wsl
kjhoeakjy@BOOK-EJ6QAG1NJL:~$ man touch


TOUCH(1)                User Commands                TOUCH(1)

NAME
       touch - change file timestamps

SYNOPSIS
       touch [OPTION]... FILE...

DESCRIPTION
       Update  the access and modification times of each FILE to the
       current time.

       A FILE argument that does not exist is created empty,  unless
       -c or -h is supplied.
```

```
       A  FILE  argument string of - is handled specially and causes
       touch to change the times of the file associated  with  stan-
       dard output.

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

       -a     change only the access time

       -c, --no-create
              do not create any files

       -d, --date=STRING
              parse STRING and use it instead of current time

       -f     (ignored)

       -h, --no-dereference
 Manual page touch(1) line 1 (press h for help or q to quit)


#3 - wsl
kjhoeakjy@BOOK-EJ6QAG1NJL:~$ cd /tmp/missing
kjhoeakjy@BOOK-EJ6QAG1NJL:/tmp/missing$ touch semester
kjhoeakjy@BOOK-EJ6QAG1NJL:/tmp/missing$ ls
semester
```

▼ 2주차

▼ 복습 코드

```
kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ foo=bar

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ echo $foo
bar

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
```

```
$ foo = bar
bash: foo: command not found

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ #띄어쓰기 중요. 위처럼 하면 foo 프로그램의 첫번째 argument =, 두번째 argum

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ echo "Hello"
Hello

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ echo 'World'
World

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ echo "Value is $foo"
Value is bar

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ echo 'Value is $foo'
Value is $foo

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ #double quote 써야 $foo가 bar로 변환

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ vim mcd.sh

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ #파일 열리고 esc, :wq 입력하면 터미널로 돌아옴

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ #mcd.sh라는 스크립트를 여는 명령어. 현재는 저 스크립트가 비어 있음

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ mcd () {
>     mkdir -p "$1"
>     cd "$1"
> }
```

```
kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ mcd test

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~/test
$ cd ..

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ $1

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ rmdir test

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ mkdir test

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ cd $_

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~/test
$ cd ..

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ mkdir /new
mkdir: cannot create directory ‘/new’: Permission denied

kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ sudo mkdir /new
bash: sudo: command not found
#WSL 이용


kjhoe@BOOK-EJ6QAG1NJL MINGW64 ~
$ wsl
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# sudo mkdir /mnt/new
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# ls -l /mnt/new
total 0

root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# echo "Hello"
Hello
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# echo $?
```

```
0
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# grep foobar mcd.sh
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# echo $?
1
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# true
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# echo $?
0
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# false
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# echo $?
1
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# false || echo "Oops fail"
Oops fail
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# true || echo "Oops fail"
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# true && echo "Oops fail"
Oops fail
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# false && echo "Oops fail"
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# false ; echo "Oops fail"
Oops fail
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# foo=$(pwd)
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# echo $foo
/mnt/c/Users/kjhoe
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# echo "We are in $(pwd)"
We are in /mnt/c/Users/kjhoe
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# echo 'We are in $(pwd)'
We are in $(pwd)

root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# ls *.sh
mcd.sh
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# mkdir project42
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# mkdir project1
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# ls project?
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# ls
 1
 AppData
'Application Data'
'CS 1강'
 Contacts
 Cookies
 Custom_NeuralProphet_Logs
 Documents
```

```
 Downloads
 Favorites
 Links
'Local Settings'
 Music
'My Documents'
 NTUSER.DAT
 NTUSER.DAT{a2332f18-cdbf-11ec-8680-002248483d79}.TM.blf
 NTUSER.DAT{a2332f18-cdbf-11ec-8680-002248483d79}.TMContainer0
 NTUSER.DAT{a2332f18-cdbf-11ec-8680-002248483d79}.TMContainer0
 NetHood
 OneDrive
 PATH
 PrintHood
 Recent
'Saved Games'
 Searches
 SendTo
 Templates
 Videos
 anaconda3
 hello.txt
 hello2.txt
 ls.txt
 mcd.sh
 ntuser.dat.LOG1
 ntuser.dat.LOG2
 ntuser.ini
 project
 project1
 project42
 test
'시작 메뉴'
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# #wsl에서는 불가능해보임

root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# shellcheck mcd.sh

In mcd.sh line 1:
mcd () {
^-- SC2148 (error): Tips depend on target shell and yours is unknown. Ad
```

```
In mcd.sh line 3:
        cd "$1"
        ^-----^ SC2164 (warning): Use 'cd ... || exit' or 'cd ... || return' in case

Did you mean:
        cd "$1" || exit

For more information:
  https://www.shellcheck.net/wiki/SC2148 -- Tips depend on target shell a
  https://www.shellcheck.net/wiki/SC2164 -- Use 'cd ... || exit' or 'cd ... |...
```

▼ 스크립트 작성법

```
#mcd.sh 스크립트를 만들고 싶다!
vim mcd.sh
#위처럼 입력하면 스크립트 창 등장
#i 입력하면 insert 가능. 내용 작성.
#esc 누르고 :wq 입력 (w는 저장, q는 종료 의미)
```

▼ 따옴표

- $( ) : 명령어 치환. 괄호 안 명령어 실행 결과를 해당 위치에 삽입.
- 큰따옴표: 문자열 내에서 명령어 치환과 변수 치환이 동작
- 작은 따옴표: 모든 치환 비활성화

▼ echo $?

`$?` : Bash에서 **직전에 실행된 명령어의 종료 상태 코드 반환

- **0**: 명령어가 성공적으로 실행됨.
- **1** 또는 기타 숫자: 명령어가 실패하거나 오류가 발생.

`grep foobar mcd.sh` :

- `mcd.sh` 파일에서 `"foobar"`라는 문자열을 검색.
- 결과가 없으면 `grep`의 종료 상태는 **1**.
- **`echo $?`**: 직전 명령어(`grep`)의 종료 상태를 출력 → 검색 실패로 결과는 1.

`true`: 항상 성공(종료 상태 0)을 반환

▼ 논리연산자 | |, &&, ;

`false || echo "Oops fail"`

- **`||`**: OR →왼쪽 명령(`false`)이 실패하면 오른쪽 명령(`echo "Oops fail"`) 실행.
- 결과: `"Oops fail"` 출력

`true || echo "Oops fail"`

- 왼쪽 명령(`true`)이 성공했으므로 오른쪽 명령(`echo "Oops fail"`)은 실행 X.
- 결과: 출력 X.

`true && echo "Oops fail"`

- **`&&`**: AND . 왼쪽 명령(`true`)이 성공하면 오른쪽 명령(`echo "Oops fail"`)이 실행.
- 결과: `"Oops fail"` 출력.

`false ; echo "Oops fail"`

- **`;`**: 명령어를 순차적으로 실행. 첫 번째 명령의 성공 여부와 관계없이 다음 명령 실행.
- 결과: `false`: 실패, 종료 상태 `1`.

  `echo "Oops fail"`: 실행되고 `"Oops fail"` 출력.

▼ 파일찾기

find: 현재 dir부터 하위 dir 포함한 파일 시스템 탐색

- `.` : 현재 디렉토리 → `find .:` 현재 디렉토리와 그 하위 디렉토리에서 검색
- `name` 옵션: 파일이나 디렉토리의 이름을 검색 조건으로
- `-name src` :이름이 정확히 `src`인 파일 또는 디렉토리 검색
- `type` 옵션
  - `d`: directory

    - `f`: file
    - `l`: symbolic link
- `path:` 검색할 파일 또는 디렉토리의 경로를 패턴으로 지정
- `'**/test/**/*.py'`: Global Pattern을 사용하여 특정 디렉토리 구조와 파일 확장자에 매칭
    - `*`: 0개 이상의 디렉토리 계층
    - `/test/`: 경로에 `test` 디렉토리가 포함되어야 함
    - `/**/*.py`: `test` 디렉토리 내부 및 하위 디렉토리에 `.py`로 끝나는 파일을 검색
- 대소문자 구분: `-ipath` 옵션 사용
- `mtime` 옵션: 파일 수정 시간 기분 검색 (24시간 단위)
    - `1` : 지난 1일 이내 수정된 파일 검색
    - 부호 `-,+`: 1일보다 더 최근, 오래전에 수정된 파일.
    - `mmin` 옵션 사용하면 특정 시간 단위 검색 가능
- `size` 옵션: 파일 크기 기준 검색
    - `+500k` : 500KB보다 큰 파일
    - `+`: 지정한 크기보다 큰 파일.
    - `k`(킬로바이트), `M`(메가바이트), `G`(기가바이트)
    - 기본적으로 블록(1블록 = 512바이트) 단위로 작동

▼ 코드 찾기

`grep` :텍스트 파일에서 특정 패턴(문자열)을 검색

→ 대안으로 `rg` 사용

▼ history

history | grep find: 과거 내 command 중에서 find 포함된 것 출력

▼ exercise

```
#1
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# ls -alh --color=auto -t
total 20M
```

```
drwxrwxrwx 1 root root 4.0K Jan 21 11:41  OneDrive
-rwxrwxrwx 1 root root  14M Jan 21 01:54  NTUSER.DAT
drwxrwxrwx 1 root root 4.0K Jan 21 01:21  .
-rwxrwxrwx 1 root root   36 Jan 21 01:21  mcd.sh
drwxrwxrwx 1 root root 4.0K Jan 21 01:06  fao
drwxrwxrwx 1 root root 4.0K Jan 21 01:06  bar
-rwxrwxrwx 1 root root    0 Jan 21 01:02  foo10
-rwxrwxrwx 1 root root    0 Jan 21 01:02  foo2
-rwxrwxrwx 1 root root    0 Jan 21 01:02  foo1
-rwxrwxrwx 1 root root    0 Jan 21 01:02  foo
drwxrwxrwx 1 root root 4.0K Jan 21 00:58  project1
drwxrwxrwx 1 root root 4.0K Jan 21 00:57  1
drwxrwxrwx 1 root root 4.0K Jan 21 00:57  project
drwxrwxrwx 1 root root 4.0K Jan 21 00:56  project42
drwxrwxrwx 1 root root 4.0K Jan 21 00:28  test
-rwxrwxrwx 1 root root  695 Jan 21 00:22  .viminfo
drwxrwxrwx 1 root root 4.0K Jan 20 17:30  Downloads
-rwxrwxrwx 1 root root   52 Jan 14 12:05  ls.txt
-rwxrwxrwx 1 root root   12 Jan 14 12:04  hello2.txt
-rwxrwxrwx 1 root root    6 Jan 14 12:03  hello.txt
drwxrwxrwx 1 root root 4.0K Jan 14 12:00 'CS 1강'
-rwxrwxrwx 1 root root  182 Jan 13 16:32  .bash_history
-rwxrwxrwx 1 root root  156 Jan 13 16:30  .gitconfig
-rwxrwxrwx 1 root root   27 Jan 13 15:44  PATH
drwxrwxrwx 1 root root 4.0K Nov 15 01:44  Custom_NeuralProphet_Logs
drwxrwxrwx 1 root root 4.0K Oct 24 13:55  .cache
drwxrwxrwx 1 root root 4.0K Oct 10 10:41  .vscode-R
drwxrwxrwx 1 root root 4.0K Oct  6 19:31  .keras
drwxrwxrwx 1 root root 4.0K Oct  6 12:54  anaconda3
drwxrwxrwx 1 root root 4.0K Aug 25 12:46  Music

#-a: 모든 파일, 숨겨진 파일 표시
#-h: 파일 크기 표시
#-t: 수정 시간 기준 정렬
#--color=auto: 색상 적용 (권한, 유형)
#-l: 자세한 정보 (권한, 소유자, 크기, 수정일 등)



#2
```

```
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# vim marco.sh

#!/bin/bash

case "$1" in
        marco)
                export SAVED_DIR="$(pwd)"
                echo "Current dir saved: $SAVED_DIR"
                ;;
        polo)
                if [ -n "SAVED_DIR" ]; then
                        cd "SAVED_DIR" || echo "Failed to navigate to $SAVED_DIR'
                        echo "Returned to saved directory: $SAVED_DIR"
                else
                        echo "No dir saved yet. Run 'marco' first."
                fi
                ;;
esac

root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# source ./marco.sh marco
Current dir saved: /mnt/c/Users/kjhoe
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# source ./marco.sh polo
-bash: cd: SAVED_DIR: No such file or directory
Failed to navigate to /mnt/c/Users/kjhoe
Returned to saved directory: /mnt/c/Users/kjhoe




#3
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# vim command.sh

#!/bin/bash
n=$(( RANDOM % 100 ))

if [[ n -eq 42 ]]; then
   echo "Something went wrong"
   >&2 echo "The error was using magic numbers"
   exit 1
```

```
fi

echo "Everything went according to plan"


root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# vim scrpt1.sh

#!/bin/bash

STDOUT_LOG="stdout.log"
STDERR_LOG="stderr.log"

count=0

while ./command.sh >>"$STDOUT_LOG" 2>>"$STDERR_LOG"; do
   ((count++))
done

echo "Command failed after $count executions."
echo "Check logs: $STDOUT_LOG, $STDERR_LOG"

root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# chmod +x command.sh
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# chmod +x scrpt1.sh
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# ./scrpt1.sh
Command failed after 55 executions.
Check logs: stdout.log, stderr.log


#4
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# vim zip.sh

#!/bin/bash

find . -type f -name "*.html" -print0 | xargs -0 zip html_files.zip

root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# chmod +x zip.sh
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# ./zip.sh


#5
```

```
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe/csstudy# rg --files
zip.sh
scrpt1.sh
mcd.sh
marco.sh
command.sh
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe/csstudy# rg --files | xargs
1737439674 zip.sh
1737439551 mcd.sh
1737439577 marco.sh
1737439628 command.sh
1737439655 scrpt1.sh
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe/csstudy# rg --files | xargs
1737439674 zip.sh
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe/csstudy# rg --files | xargs
1737439674 zip.sh
1737439655 scrpt1.sh
1737439628 command.sh
1737439577 marco.sh
1737439551 mcd.sh
```

▼ 3주차

▼ less

- 파일이나 명령어 출력 내용을 스크롤로 탐색 가능하게 하는 텍스트 뷰어
- less [파일명]
- command | less
- space bar, 화살표 등 단축키로 스크롤 가능
- 검색 기능 보유
  - '/텍스트': 아래 방향 검색
  - '?텍스트': 위 방향 검색

▼ sed

- sed ‘명령여/정규표현식/flag’ [파일명]
- s: 치환
  - 정규표현식 뒤에 치환할 문자열 입력
  - echo "Hello World" | sed 's/World/SED/'

    : World를 SED로 치환. Hello SED 출력
  - sed 's/foo//' [파일명]

    파일에서 foo에 해당하는 부분 삭제 (빈 문자열로 치환)
  - sed 's/foo/bar/g' [파일명]

    파일에서 foo를 모두 bar로 치환
  - sed '2s/foo/bar/' [파일명]

    두번째 줄에서만 foo를 bar로 치환

▼ greedy & non-greedy

- 정규표현식의 *, +: 최대한 많은 텍스트 일치시키려함 (패턴이 일치하는 여러 부분 중 가장 긴 문자열 채택)
- 문제 상황 예시: bold text and italic text
  - 패턴 <.*>: 모든 HTML 태그 제거 목적
  - greedy matching → and 출력
  - non-greedy matching → bold test and italic text 출력
- perl 명령어의 *?, +? 사용하면 non-greedy matching 가능

▼ exercise

2. grep -E '(.*a*.){3,}' /usr/share/dict/words | grep -v 's$' | sed 'y/ABCDEFGHIJKLMNOPQRSTUVWXYZ/abcdefghijklmnopqrstuvwxyz/’ | awk '{print substr($0, length($0)-1, 2)}' | sort | uniq | wc -l

- '(.*a*.){3,}': a가 세번 이상 포함된 단어 필터링하는 정규표현식
- grep -v 's$’: grep 부정문. s로 끝나는 단어 제외
- sed 'y/ABCDEFGHIJKLMNOPQRSTUVWXYZ/abcdefghijklmnopqrstuvwxyz/’: 대문자를 소문자로 치환

- awk '{print substr($0, length($0)-1, 2)}’: 마지막 두 글자 추출
- sort | uniq -c | sort -nr: 첫번째 줄을 가장 빈번한 두 글자 조합으로 만듦, 중복 제거
- | sort | uniq | wc -l: 개수 세기
- 도전문제: echo {a..z}{a..z} | tr ' ' '\n' > all_combinations.txt 사용
  - 26개의 알파벳으로 만들 수 있는 676개의 조합
  - 이후 존재하는 조합과의 교집합 구하기

3. ‘>’는 원본 파일을 비우고 덮어씀. sed가 원본 파일에서 데이터를 읽기 전에 파일이 지워질 수 있음. 파일 내용을 동시에 읽고 덮으면 데이터 손실의 위험이 있으므로 임시 파일 생성하는 것이 안전
   - -i 옵션 사용: 파일을 직접 수정하면서 임시 파일을 사용
4. 단계별로 설명

   (1) journalctl --list-boots | head -n 10 > bootlog_index
   journalctl -b -10..-1 | grep -E "Logs begin at|Startup finished in" > bootlog

   : 최근 10번 부팅 로그 추출

   (2) cat bootlog | awk '{print $2}' | sed -E "s/^.:(.)/\1/" | xargs -n2 | awk '{print $2"-"$1}' | bc > boot_times

   : 부팅 시간 계산 (초단위)

   (3) cat boot_times | R --slave -e 'x ← scan(file="stdin", quiet=TRUE); summary(x)'

   : R 사용하여 통계값 계산

5.

6. htmlq 사용

```
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# sudo snap install html
htmlq v0.4.0 from Barry Price (bp) installed
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# curl -s https://stats.wi
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# cat wikipedia_stats.htr
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# awk 'NR % 2 == 1 {pri
Min:  Max: 254444
```

```
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# awk 'NR % 2 == 1 {firs
Sum of differences: 225795680
```

▼ 4주차

▼ Job Control

- `SIGINT`: Signal Interrupt (2번 시그널). 사용자가 키보드에서 `Ctrl + C`를 입력하면 발생. 이 시그널을 받은 프로세스는 기본적으로 종료됨.
  - sleep 3 (3초 sleep) 중에 Ctrl+C 입력하면 프로세스 종료됨
- `SIGQUIT`: 프로세스를 종료할 때 생성되는 시그널 (3번 시그널). `Ctrl + \` 키를 입력하면 발생. `SIGINT` (`Ctrl + C`)와 비슷하지만, `SIGQUIT`는 프로세스의 **코어 덤프(Core Dump)**를 생성하면서 종료하는 점이 다름 → 코어 덤프 파일(`core` 또는 `core.<pid>`)이 생성됨.
  - Core Dump: 프로세스가 비정상적으로 종료될 때 해당 시점의 메모리 상태를 저장한 파일 → 디버깅 시 프로그램이 어떤 상태로 죽었는지 분석 가능
  - SIGQUIT, SIGSEGV, abort() 등으로 강제 종료 시 생성
  - core.<pid>에서 pid는 process id.
  - GDB (GNU debugger)로 코어 덤프 분석 가능

```
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# python3 untitled-1.py
31^C
I got a SIGINT, but I am not stopping
50^C
I got a SIGINT, but I am not stopping
68^C
I got a SIGINT, but I am not stopping
82^\Quit (core dumped)



root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# python3 untitled-1.py
27^\Quit
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# ls -lh core*
-rwxrwxrwx 1 root root 0 Feb  2 21:46 core
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# gdb python3 core
GNU gdb (Ubuntu 15.0.50.20240403-0ubuntu1) 15.0.50.20240403-git
Copyright (C) 2024 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.
```

```
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from python3...
(No debugging symbols found in python3)
"/mnt/c/Users/kjhoe/core" is not a core dump: file format not recognized
(gdb) bt
#라고 입력하면 backtrace 확인이 가능해야 하는데 아무리 해도 계속 no stack 나옴
```

- `SIGTERM`(15번 시그널): 프로세스를 정상적으로 종료하도록 요청. `SIGKILL(9)`처럼 강제 종료하는 것이 아니라, 프로세스가 종료할 기회 제공.
- 아래는 `SIGKILL`

```
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# nohup sleep 2000 &
[1] 5733
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# nohup: ignoring input and
^C
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# jobs
[1]+  Running                 nohup sleep 2000 &
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# kill %1
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe# jobs
[1]+  Terminated              nohup sleep 2000
```

Exercise

1. *From what we have seen, we can use some `ps aux | grep` commands to get our jobs' pids and then kill them, but there are better ways to do it. Start a `sleep 10000` job in a terminal, background it with `Ctrl-Z` and*

continue its execution with `bg`*. Now use* `pgrep` *to find its pid and* `pkill` *to kill it without ever typing the pid itself. (Hint: use the* `-af` *flags).*

```
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ sleep 10000 &
[1] 1239
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ pgrep -af "sleep 1000
1239 sleep 10000
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ pkill -af "sleep 10000"
pkill: invalid option -- 'a'

Usage:
 pkill [options] <pattern>

Options:
 -<sig>                  signal to send (either number or name)
 -H, --require-handler     match only if signal handler is present
 -q, --queue <value>       integer value to be sent with the signal
 -e, --echo              display what is killed
 -c, --count             count of matching processes
 -f, --full              use full process name to match
 -g, --pgroup <PGID,...>   match listed process group IDs
 -G, --group <GID,...>     match real group IDs
 -i, --ignore-case         match case insensitively
 -n, --newest            select most recently started
 -o, --oldest            select least recently started
 -O, --older <seconds>     select where older than seconds
 -P, --parent <PPID,...>   match only child processes of the given pare
 -s, --session <SID,...>   match session IDs
    --signal <sig>       signal to send (either number or name)
 -t, --terminal <tty,...>  match by controlling terminal
 -u, --euid <ID,...>       match by effective IDs
 -U, --uid <ID,...>        match by real IDs
 -x, --exact             match exactly with the command name
 -F, --pidfile <file>      read PIDs from file
 -L, --logpidfile          fail if PID file is not locked
 -r, --runstates <state>   match runstates [D,S,Z,...]
 -A, --ignore-ancestors    exclude our ancestors from results
 --cgroup <grp,...>        match by cgroup v2 names
 --ns <PID>              match the processes that belong to the same
                      namespace as <pid>
```

```
 --nslist <ns,...>        list which namespaces will be considered for
                          the --ns option.
                          Available namespaces: ipc, mnt, net, pid, user, uts

 -h, --help     display this help and exit
 -V, --version  output version information and exit

 For more details see pgrep(1).
```

→ `pgrep sleep`을 실행하면 `sleep`이라는 이름의 실행 중인 프로세스의 PID를 출력

→ `a` : 프로세스 전체 경로를 출력

→ `f` : 프로세스의 전체 커맨드 라인을 기준으로 검색

→ `pgrep`과 `pkill`을 활용하면 PID를 직접 입력하지 않고 프로세스를 종료 가능

2. *Say you don't want to start a process until another completes, how you would go about it? In this exercise our limiting process will always be `sleep 60 &`. One way to achieve this is to use the `wait` command. Try launching the sleep command and having an `ls` wait until the background process finishes.*

```
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ sleep 60 &
[2] 1243
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ pgrep sleep | wait && ls
 1
 AppData
'Application Data'등등 ls 목록
#60초 기다리지 않고 바로 실행됨


#함수 사용
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ pidwait() {
  while kill -0 $1 2>/dev/null
  do
    sleep 1
  done
  ls
```

```
}
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ sleep 60 &
pidwait $(pgrep -n sleep)
[2] 1570
[2]+  Done                    sleep 60
 1
 AppData
'Application Data' 등등 ls 목록이 60초 뒤에 출력됨
```

- `$1`: 함수에 전달된 첫 번째 인자(PID)
- `kill -0 $1`:
  - `kill -0`은 실제로 프로세스를 종료하지 않고, 프로세스가 존재하는지 확인하는 기능
  - `0` 신호는 아무 작업도 하지 않고, 프로세스가 존재하는지만 체크
  - 존재하면 0 반환, 존재하지 않으면 에러 코드 반환
  - kill [시그널] [PID]

    → 예제: `kill -9 12345` → PID `12345`의 프로세스를 강제 종료(`9` 시그널)
- `2>/dev/null`: 표준 에러 출력을 숨김 (불필요한 에러 메세지 빼고 종료 상태 코드만 확인하려는듯?)
- `while` 루프는 해당 PID가 종료될 때까지 1초 간격으로 체크.
- 종료되면 `ls` 실행

▼ Terminal Multiplexers

: 하나의 터미널에서 여러 개의 session, window, pane 동시에 사용할 수 있도록 함.

TMUX 이용

- tmux key binding: C-b-x.
  - leader key: Ctrl + b (<C-b>)
  - Ctrl+b 눌렀다가 떼고 x 누르면 현재 활성화된 Pane 종료

`<C-b> c`	새로운 윈도우 생성
`<C-b> %`	현재 창을 세로 분할
`<C-b> "`	현재 창을 가로 분할
`<C-b> x`	현재 패널 닫기
`<C-b> n`	다음 윈도우로 이동
`<C-b> p`	이전 윈도우로 이동
`<C-b> d`	tmux에서 빠져나오기 (세션 유지)

- 여러 개의 터미널 창을 하나의 터미널에서 관리 가능
- 터미널 세션을 유지하면서 백그라운드에서 실행 가능 (`nohup` 없이도 가능)
- SSH 연결이 끊겨도 세션을 다시 연결 가능

1. Sessions

```
tmux ls # 현재 실행 중인 tmux 세션 목록
tmux new -s <name> # 새로운 세션 생성
tmux attach -t <name> #특정 세션에 다시 연결
tmux kill-session -t <name> #특정 세션 종료
tmux kill-server #실행중인 모든 세션 종료
```

- Ctrl+b → d: 세션에서 나가기 (종료 아님)

2. Windows

- Ctrl+b → c: 새 윈도우 생성
- <C-b> n : 다음 윈도우로 이동
- <C-b> p : 이전 윈도우로 이동
- <C-b> N : N번째 윈도우로 이동 (ex: <C-b> 1 → 1번 윈도우로 이동)
- <C-b> w : 현재 열린 윈도우 목록 표시
- <C-b> , : 현재 윈도우 이름 바꾸기
- <C-d> : 현재 윈도우 닫기

3. Panes

- <C-b> " : 현재 창을 가로로 분할

- <C-b> % : 현재 창을 세로로 분할
- <C-b> <방향키> : 해당 방향의 패널로 이동 (예: <C-b> → → 오른쪽 패널로 이동)
- <C-b> <키> : 특정 키로 크기 조절 가능
- <C-b> :resize-pane -D 10 # 아래로 10줄 크기 조절
- <C-b> :resize-pane -U 10 # 위로 10줄 크기 조절
- <C-b> :resize-pane -L 10 # 왼쪽으로 10줄 크기 조절
- <C-b> :resize-pane -R 10 # 오른쪽으로 10줄 크기 조절
- <C-b> x : 현재 패널 닫기
- <C-b> <space> : 모든 패널을 한번에 정렬
- <C-b> z : 패널을 전체 화면으로 확대, 다시 입력하면 원래 크기로 복귀

4. tmux 스크롤, 복사 모드

- <C-b> [: 스크롤 모드 활성화 → 방향키/ `PgUp` / `PgDn` 키를 사용해 스크롤 가능
- <C-b> [: 복사 모드 활성화
  - <C-b>] : 붙여넣기

Exercise

1. *Follow this* `tmux` *tutorial and then learn how to do some basic customizations following these steps.*

▼ Alias

```
alias ll="ls -lh --color=auto" #ll 압력하면 뒤 내용 실행됨
alias #현재 정의된 모든 alias 확인
alias ll #alias 내용 확인
```

- `\명령어`: alias가 충돌하거나 원래 명령어 그대로 사용하고 싶을 때 이용
- `unalias ll` : 해당 alias 삭제

- `unalias -a` : 모든 alias 삭제
- 별칭은 간단한 명령어를 단축하는 데에만 사용하고 더 복잡한 작업은 함수 사용하기
- 위험한 명령어 보호에 이용하면 유용할듯

```
alias rm="rm -i"  # 삭제할 때 확인 메시지 표시
alias mv="mv -i"  # 이동할 때 덮어쓰기 확인
alias cp="cp -i"  # 복사할 때 덮어쓰기 확인

rm test.txt
rm: remove regular file 'test.txt'? y  # 'y' 입력 시 삭제
mv oldfile.txt newfile.txt
mv: overwrite 'newfile.txt'? n  # 'n' 입력하면 덮어쓰기 안 됨
$ cp source.txt destination.txt
cp: overwrite 'destination.txt'? y  # 'y' 입력하면 덮어쓰기 진행

\rm filename  # -i 옵션 없이 강제 삭제
```

Exercise

1. *Create an alias `dc` that resolves to `cd` for when you type it wrongly.*

```
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ alias dc="cd"
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ dc
root@BOOK-EJ6QAG1NJL:~$ dc -
/mnt/c/Users/kjhoe
```

2. *Run `history | awk '{$1="";print substr($0,2)}' | sort | uniq -c | sort -n | tail -n 10` to get your top 10 most used commands and consider writing shorter aliases for them.*

```
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ history | awk '{$1="";p
      3 vim zip.sh
      4 rg --files
      4 sudo apt update
```

```
    4 sudo sysctl -w kernel.core_pattern=core
    4 tmux ls
    4 vim marco.sh
    5 jobs
    7 gdb python3 core
    7 ls -lh core*
    7 python3 untitled-1.py
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ alias vz="vim zip.sh"
alias vm="vim marco.sh"
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ vm
```

▼ Dotfiles

: Linux, MacOS에서 configuration 파일을 저장하는 txt file.

→ 파일명이 .으로 시작해서 ls 명령어 사용 시 나타나지 않음

Exercise

.....n

▼ Remote Machines (SSH, Port Forwarding)

SSH (Secure Shell): 원격 서버에 안전하게 접속, 파일 전송, port forwarding 기능 제공.

- SSH 기능: 원격 로그인 (다른 서버에 접속), 명령 실행 (원격 시스템에서 명령 실행), 파일 전송 (SCP, SFTP-보안 강화된 방식의 파일 전송), 포트 포워딩 (원격 시스템의 특정 포트를 로컬에서 사용/ 로컬 시스템의 특정 포트를 원격 시스템에서 사용), SSH 키 인증 (비밀번호가 아닌 키 기반의 인증으로 보안 강화)
- 기본 사용법

```
ssh 주소.168.1.100 #접속 예제 (원격 서버의 IP주소/ 도메인 필요)
#비밀번호 입력

ssh user@remote_host "ls -lah" #접속 없이 원격으로 명령 실
```

```
scp file.txt user@remote_host:/home/user/ #txt 파일을 원격 서버에 저장 (전송
scp user@remote_host:/home/user/file.txt . #원격 서버에서 txt 파일 가져오가
```

- 키 기반 인증

:암호 없이 SSH 서버에 안전하게 로그인

- Public key, Private key 쌍으로 이용
- public key: 서버에 저장됨 (`~/.ssh/authorized_keys` 파일에 추가). 누구나 볼 수 있지만, 단독으로는 인증 불가
- private key: 클라이언트(로컬 컴퓨터)에 저장됨 (`~/.ssh/id_rsa`). 비밀 키이며, 사용자만 알고 있음.
- 작동 원리: 클라이언트(로컬)가 SSH 로그인 시도 → 개인 키를 사용해 인증 정보 생성 → 서버(원격)는 `~/.ssh/authorized_keys`에 등록된 공개 키를 이용해 인증을 검증 → 일치하면 로그인 성공 (비밀번호 입력 필요 없음)

```
ssh-keygen -t rsa -b 4096 -C "emailaddress@gmail.com"'
#-t rsa: RSA 암호화 방식 (기본 방식임)
#-b 4096: 4096 비트 길이의 키 생성
#-C 이메일: 키 식별을 위한 주석

root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa
Your public key has been saved in /root/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:VSmj7P7MHsD9rGt8dk2dHl+6ryd4as2lUnHqVbpKdTl kjhoeakjy@
The key's randomart image is:
+---[RSA 4096]----+
|          ..  |
|         o..   |
|      . ..o    |
|      .+o. .  . |
|      .S+Eo. ..=|
```

```
|       +oo*  .o+|
|      +.+.*o. o |
|     ..+oBoO o .|
|      ..*O*oB.  |
+----[SHA256]-----+

ssh-copy-id user@remote_host #생성한 SSH 키를 원격 서버에 등록
#공개 키(id_rsa.pub)를 자동으로 서버의 ~/.ssh/authorized_keys 파일에 추가해

ssh user@remote_host #비밀번호 없이 원격 서버에 접속 가능
```

- Port Forwarding

: SSH 연결을 통해 특정 포트의 트래픽을 전달하여 보안 유지.

→ 방화벽 우회, 원격 서버의 특정 서비스를 로컬에서 사용 가능

*port: 컴퓨터 네트워크에서 데이터를 주고받기 위한 채널. 한 대의 컴퓨터에서 여러 개의 프로그램이 동시에 네트워크할 수 있도ㄹ고 하는 주소

→ IP주소: 서버 식별 주소 (같은 네트워크 내의 컴퓨터들이 이용)

→ 포트 번호: 같은 서버 내에서 여러 네트워크 프로그램 구분. 0부터 65535까지 사용 가능.

→ IP주소가 아파트 주소라면 포트 번호는 호수를 의미

포트 범위	설명	예제
0~1023	**잘 알려진 포트(Well-Known Ports)**	SSH(22), HTTP(80), HTTPS(443)
1024~49151	**등록된 포트(Registered Ports)**	MySQL(3306), PostgreSQL(5432)
49152~65535	**동적/임시 포트(Dynamic/Private Ports)**	클라이언트 프로그램에서 임시로 사용

서비스	포트 번호	설명
SSH	`22`	원격 접속 (Secure Shell)
FTP	`21`	파일 전송 (File Transfer Protocol)
HTTP	`80`	웹사이트 접속 (HyperText Transfer Protocol)
HTTPS	`443`	보안 웹사이트 접속 (HyperText Transfer Protocol Secure)
DNS	`53`	도메인 이름 변환 (Domain Name System)
SMTP	`25`	이메일 발송 (Simple Mail Transfer Protocol)
IMAP/POP3	`143/110`	이메일 수신 (Internet Message Access Protocol)
MySQL	`3306`	데이터베이스 서비스
PostgreSQL	`5432`	데이터베이스 서비스
Redis	`6379`	캐시 및 데이터 저장소

→ 웹 서버 실행하면 기본적으로 80번/ 443번 포트 사용

```
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ netstat -tulnp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 10.255.255.254:53       0.0.0.0:*               LISTEN      -

tcp        0      0 127.0.0.54:53           0.0.0.0:*               LISTEN      258/syster
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      258/systen
udp        0      0 127.0.0.54:53           0.0.0.0:*                           258/systemd-
udp        0      0 127.0.0.53:53           0.0.0.0:*                           258/systemd-
udp        0      0 10.255.255.254:53       0.0.0.0:*                           -

udp        0      0 127.0.0.1:323           0.0.0.0:*                           -

udp6       0      0 ::1:323                 :::*
```

→ 53번, 323번 포트 사용중

*Port Forwarding

- 목적: 방화벽이 차단된 서비스 접근 (내부망에서만 접속 가능한 db 등), 보안 강화를 위한 SSH 암호화 터널링, 원격 서버의 포트를 로컬에서 사용/ 그 반대

- 종류: Local, Remote, Dynamic

1. Local Port Forwarding: 내 컴퓨터에서 특정 포트를 열고 원격 서버의 서비스에 접근
    - 내 컴퓨터에서 특정 포트를 열어줌 → 해당 포트로 들어오는 모든 트래픽을 SSH 터널을 통해 원격 서버로 전달 → 원격 서버 트래픽을 해당 서비스로 전달

```
ssh -L [로컬포트]:[대상호스트]:[대상포트] user@remote_host
#-L: 로컬 포트 포워딩
#로컬 포트: 내 컴퓨터에서 사용할 포트 번호
ssh -L 9999:localhost:8080 user@remote_host
#내 로컬 브라우저에서 http://localhost:9999에 접속하면 원격 서버의 808
```

2. Remote Port Forwarding: 원격 서버의 특정 포트를 열어 내 컴퓨터로 연결
    - 원격 서버에서 특정 포트 열어둠 → 해당 포트로 들어오는 모든 트래픽을 SSH 터널을 통해 내 컴퓨터로 전달 → 내 콤퓨터는 트래픽을 특정 서비스로 전달

```
ssh -R [원격포트]:[로컬호스트]:[로컬포트] user@remote_host
ssh -R 8080:localhost:3000 user@remote_host
# 원격 서버에서 http://localhost:8080에 접속하면 내 로컬 컴퓨터의 3000
```

Exercise

1. *Go to* `~/.ssh/` *and check if you have a pair of SSH keys there. If not, generate them with* `ssh-keygen -o -a 100 -t ed25519`*. It is recommended that you use a password and use* `ssh-agent` *, more info here.*

```
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ ls ~/.ssh/
id_rsa  id_rsa.pub
```

2. *Edit* `.ssh/config` *to have an entry as follows*

```
Host vm
  User username_goes_here
  HostName ip_goes_here
  IdentityFile ~/.ssh/id_ed25519
  LocalForward 9999 localhost:8888
```

By default the SSH configuration file may not exist so you may need to create `ssh config` *by* `touch ~/.ssh/config`*. This file must be readable and writable only by the user, and not accessible by others:* `chmod 600 ~/.ssh/config`*. When copy-pasting make sure to check for IdentityFile whether it should be* `~/.ssh/id_e25519` *or* `~/.ssh/id_rsa` *depending on the filename in which the private key is stored. (Source)*

```
Host vm
   User kjhoeakjy
   HostName 172.29.42.5
   IdentityFile ~/.ssh/id_rsa
   LocalForward 9999 localhost:8888
```

```
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ vim ~/.ssh/config
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ chmod 600 ~/.ssh/cor
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ ssh vm
The authenticity of host '172.29.42.5 (172.29.42.5)' can't be establishe
ED25519 key fingerprint is SHA256:kiqdYlQk8vesjoyxntQeIVYZ0GWw
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Warning: Permanently added '172.29.42.5' (ED25519) to the list of knc
kjhoeakjy@172.29.42.5's password:
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 5.15.167.4-microsoft-stanc

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Mon Feb  3 12:52:58 KST 2025

  System load:  0.03              Processes:             52
  Usage of /:   0.2% of 1006.85GB   Users logged in:       1
```

```
  Memory usage: 3%                IPv4 address for eth0: 172.29.42.5
  Swap usage:   0%

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how
   just raised the bar for easy, resilient and secure K8s cluster deploym

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent perm
applicable law.

Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 5.15.167.4-microsoft-stanc

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Mon Feb  3 09:35:49 KST 2025

  System load:  2.14              Processes:             76
  Usage of /:   0.2% of 1006.85GB   Users logged in:       0
  Memory usage: 3%                IPv4 address for eth0: 172.29.42.5
  Swap usage:   0%

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how
   just raised the bar for easy, resilient and secure K8s cluster deploym

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 5.15.167.4-microsoft-stanc

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Mon Feb  3 12:52:58 KST 2025
```

```
  System load:  0.03              Processes:             52
  Usage of /:   0.2% of 1006.85GB   Users logged in:       1
  Memory usage: 3%                IPv4 address for eth0: 172.29.42.5
  Swap usage:   0%

 * Strictly confined Kubernetes makes edge and IoT secure. Learn hov
   just raised the bar for easy, resilient and secure K8s cluster deploym

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

This message is shown once a day. To disable it please create the
/home/kjhoeakjy/.hushlogin file.
kjhoeakjy@BOOK-EJ6QAG1NJL:~$
kjhoeakjy@BOOK-EJ6QAG1NJL:~$ exit
logout
Connection to 172.29.42.5 closed.
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$
```

✅ 1 `chmod 600`을 사용하면 어떤 권한이 설정되나?

📌 `600`의 의미 (8진수 퍼미션)

`600` → 소유자만 읽고(4) & 쓸 수 있음(2), 다른 사용자는 접근 불가(0).

권한	소유자(User)	그룹(Group)	기타(Others)
`6` (읽기+쓰기)	✅ 읽기 (4) + ✅ 쓰기 (2)	❌ 없음 (0)	❌ 없음 (0)

즉, 소유자는 파일을 읽고(`r`), 쓸 수 있으며(`w`), 다른 사용자는 아무 권한이 없음.

1. *Use* `ssh-copy-id vm` *to copy your ssh key to the server.*

```
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ ssh-copy-id vm
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id_
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filt
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prc
kjhoeakjy@172.29.42.5's password:

Number of key(s) added: 1
```

```
Now try logging into the machine, with:   "ssh 'vm'"
and check to make sure that only the key(s) you wanted were added.
```

2. *Start a webserver in your VM by executing* `python -m http.server 8888`*. Access the VM webserver by navigating to* `http://localhost:9999` *in your machine.*

```
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ python3 -m http.server 88
Serving HTTP on 0.0.0.0 port 8888 (http://0.0.0.0:8888/) ...
```

http://172.29.42.5:8888/ 위 코드 실행하고 이 링크 들어가면 작동 확인 가능

* `ip a | grep inet` : localhost ip주소 확인

3. *Edit your SSH server config by doing* `sudo vim /etc/ssh/sshd_config` *and disable password authentication by editing the value of* `PasswordAuthentication`*. Disable root login by editing the value of* `PermitRootLogin`*. Restart the* `ssh` *service with* `sudo service sshd restart`*. Try sshing in again.*

```
sudo vim /etc/ssh/sshd_config
```

```
#LoginGraceTime 2m
PermitRootLogin no
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

#PubkeyAuthentication yes

# Expect .ssh/authorized_keys2 to be disregarded by default in future.
#AuthorizedKeysFile     .ssh/authorized_keys .ssh/authorized_keys2

#AuthorizedPrincipalsFile none

#AuthorizedKeysCommand none
#AuthorizedKeysCommandUser nobody

# For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
#HostbasedAuthentication no
# Change to yes if you don't trust ~/.ssh/known_hosts for
# HostbasedAuthentication
#IgnoreUserKnownHosts no
# Don't read the user's ~/.rhosts and ~/.shosts files
#IgnoreRhosts yes

# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication no
#PermitEmptyPasswords no
```

```
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ sudo vim /etc/ssh/sshd_c
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ sudo vim /etc/ssh/sshd_c
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ sudo service sshd restart
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ sudo systemctl restart ssh
root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ ssh vm
Enter passphrase for key '/root/.ssh/id_rsa':
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 5.15.167.4-microsoft-standard

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Mon Feb  3 13:12:23 KST 2025

  System load:  0.08              Processes:             51
  Usage of /:   0.2% of 1006.85GB   Users logged in:       1
  Memory usage: 3%                IPv4 address for eth0: 172.29.42.5
  Swap usage:   0%

* Strictly confined Kubernetes makes edge and IoT secure. Learn how Mi
```

```
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 5.15.167.4-microsoft-standard

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Mon Feb  3 09:35:49 KST 2025

  System load:  2.14               Processes:             76
  Usage of /:   0.2% of 1006.85GB   Users logged in:       0
  Memory usage: 3%                 IPv4 address for eth0: 172.29.42.5
  Swap usage:   0%

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how Mi
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge
Last login: Mon Feb  3 12:52:58 2025 from 172.29.42.5
kjhoeakjy@BOOK-EJ6QAG1NJL:~$
```

4. *(Challenge) Install* `mosh` *in the VM and establish a connection. Then disconnect the network adapter of the server/VM. Can mosh properly recover from it?*

   ```
   root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ ssh vm
   Enter passphrase for key '/root/.ssh/id_rsa':
   kjhoeakjy@BOOK-EJ6QAG1NJL:~$ exit
   logout
   Connection to 172.29.42.5 closed.
   root@BOOK-EJ6QAG1NJL:/mnt/c/Users/kjhoe$ mosh vm
   Enter passphrase for key '/root/.ssh/id_rsa':
   [mosh is exiting.]
   ```

   → ssh vm 연결 후 기기 네트워크를 끄면 자동으로 ssh는 끊김. 반면 mosh vm은 네트워크를 껐다가 다시 연결해도 끊기지 않고 안정적으로 작동

→ 그러나! wsl에서는 ssh vm도 와이파이 없이 잘 작동해서 실험의 의미가 없었다...!

5. *(Challenge) Look into what the `-N` and `-f` flags do in `ssh` and figure out what a command to achieve background port forwarding.*

`-N` : 원격 명령 실행하지 않고 포드 포워딩 기능만 활성화.

`-f` : SSH 프로그램을 백그라운드에서 실행 (터미널 차지 없이 포드 포워딩만 수행)

```
ssh -fN -L 9999:localhost:8888 vm
```

: 로컬 머신의 `9999` 포트를 원격 서버(`vm`)의 `8888` 포트로 포워딩. 백그라운드에서 실행(`-f`)하여 터미널을 차지하지 않음, SSH는 포트 포워딩만 수행하고 명령 실행 없이 유지됨(`-N`)

▼ 5주차

▼ add, commit

git을 왜 쓰나? 저장을 이미 했는데 이전 상태로 돌아가고 싶음. 이전 수정 내용을 하나하나 다른 파일로 저장해 놓을 수 없으니 git 이용.

터미널에 `git init` → 해당 폴더를 git이 감시

`git add <파일명> <파일명2>` : 기록할 파일 선택 (staging area)

`git add .` : 모든 파일 staging.

`git commit -m "<메세지>"` : 기록 저장소에 메세지와 함께 옮김 (repository)

`git commit -am` : add, commit 한번에 하기 (단, 새로 만든 파일이 아닐 때에만 가능. 새로 만든 파일은 따로 add 해줘야 함)

`git status` : staging, 수정된 파일들 확인 가능. 이미 commit된 파일들은 안 보임.

`git log` : commit 내역 확인 가능

VS code의 왼쪽 bar에 git 아이콘 누르면 사용 가능. VS 사용자들은 이걸 쓰는게 백만 배 나을듯

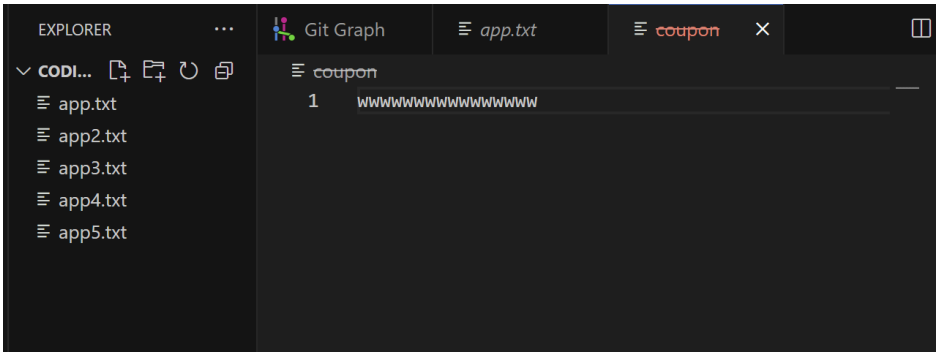
`git difftool` : 수정사항 기재된 vim editor

```
PS C:\Users\kjhoe\OneDrive\바탕화~1-LAPTOP-VGOF9FJ7-25904429\cod
15f9e7d (HEAD → main) ee
b823461 숙제
d917bc2 메세지입력
PS C:\Users\kjhoe\OneDrive\바탕화~1-LAPTOP-VGOF9FJ7-25904429\cod

This message is displayed because 'diff.tool' is not configured.
See 'git difftool --tool-help' or 'git help config' for more details.
'git difftool' will now attempt to use one of the following tools:
kompare emerge vimdiff nvimdiff

Viewing (1/2): 'app4.txt'
Launch 'vimdiff' [Y/n]? Y
#vim editor



PS C:\Users\kjhoe\OneDrive\바탕화~1-LAPTOP-VGOF9FJ7-25904429\cod
PS C:\Users\kjhoe\OneDrive\바탕화~1-LAPTOP-VGOF9FJ7-25904429\cod

Viewing (1/2): 'app4.txt'
Launch 'vscode' [Y/n]? Y
#VS editor
```

→ 이것도 git graph 설치해서 사용하는 게 훨씬 편함.

▼ branch

: 여러 방법을 시도해보고 싶을 때 가지처럼 만들어서 시도하고 괜찮은 것만 고를 때 이용



coupon 파일은 coupon branch에서 만들어서 main으로 돌아오면 목록에 안 뜸.

`git branch <가지명>` : branch 생성

`git switch <가지명>`: 해당 branch로 이동

`git merge <가지명>`: 해당 branch를 현재 작업중인 branch로 합침. 이때 같은 파일을 서로 다른 branch에서 작업했다면 충돌 발생.

`git branch -d (merge 안 한 건 -D) <브랜치명>`: 삭제

▼ Github

local repository: git add, commit 했던 폴더에서 숨긴 파일 표시 → .git 폴더

online repository: Github

```
PS C:\Users\kjhoe\OneDrive\바탕화~1-LAPTOP-VGOF9FJ7-25904429\cod
PS C:\Users\kjhoe\OneDrive\바탕화~1-LAPTOP-VGOF9FJ7-25904429\cod
[main (root-commit) 46d8e9f] a 만듦
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 a
PS C:\Users\kjhoe\OneDrive\바탕화~1-LAPTOP-VGOF9FJ7-25904429\cod
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 205 bytes | 41.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/kjhoeakjy/practice.git
 * [new branch]      main → main
branch 'main' set up to track 'https://github.com/kjhoeakjy/practice.git/ma
```

- `git push -u <repository 주소> <branch명>`
- `-u` 한번 쓰고 나면 다음 commit은 주소, branch 안 써도 됨

다른 사람이 해당 repository에 새로운 파일 올리면 `git push` 불가 → `git pull` 하고 다시 `git push`하기

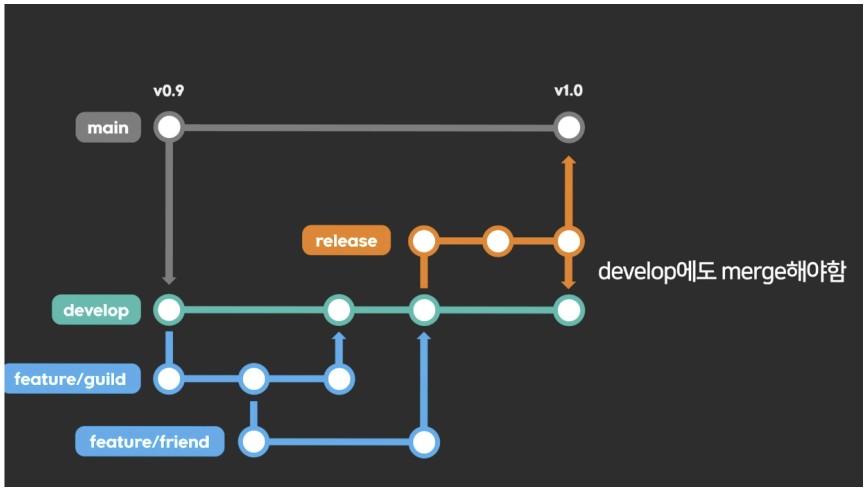
- 원래는 `git pull <repository 주소> <branch명>`. 위에서 `-u` 썼으면 그냥 `git pull`만 써도 됨.

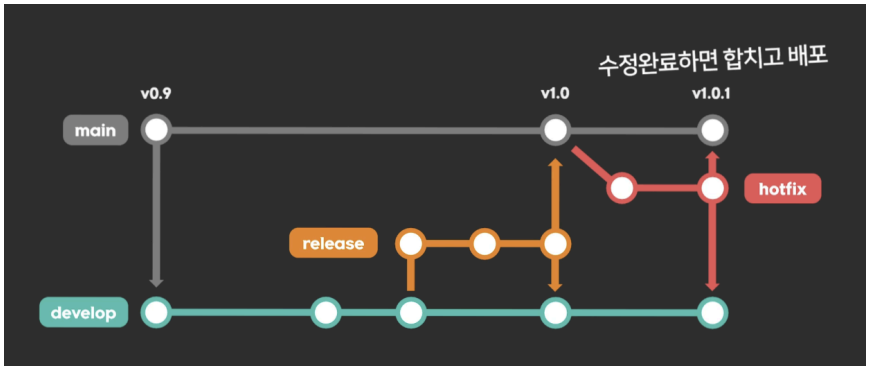
```
PS C:\Users\kjhoe\OneDrive\바탕화~1-LAPTOP-VGOF9FJ7-25904429\cod
PS C:\Users\kjhoe\OneDrive\바탕화~1-LAPTOP-VGOF9FJ7-25904429\cod
Switched to branch 'aaa'
PS C:\Users\kjhoe\OneDrive\바탕화~1-LAPTOP-VGOF9FJ7-25904429\cod
[aaa e1e5805] aaa만듦
 1 file changed, 2 insertions(+), 1 deletion(-)
PS C:\Users\kjhoe\OneDrive\바탕화~1-LAPTOP-VGOF9FJ7-25904429\cod
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Writing objects: 100% (3/3), 263 bytes | 65.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'aaa' on GitHub by visiting:
remote:      https://github.com/kjhoeakjy/practice/pull/new/aaa
remote:
To https://github.com/kjhoeakjy/practice.git
 * [new branch]      aaa → aaa
```

- github에 새로운 branch 추가됨

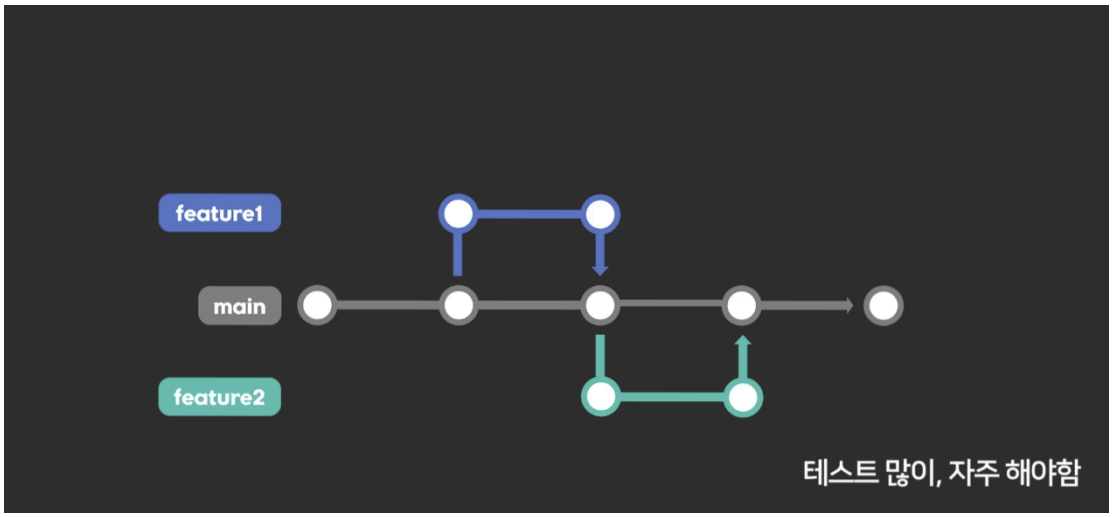
pull request: repository 안에 있는 여러 branch에 대한 merge를 요청하는 것. Github에서 할 수 있음.

gitflow:





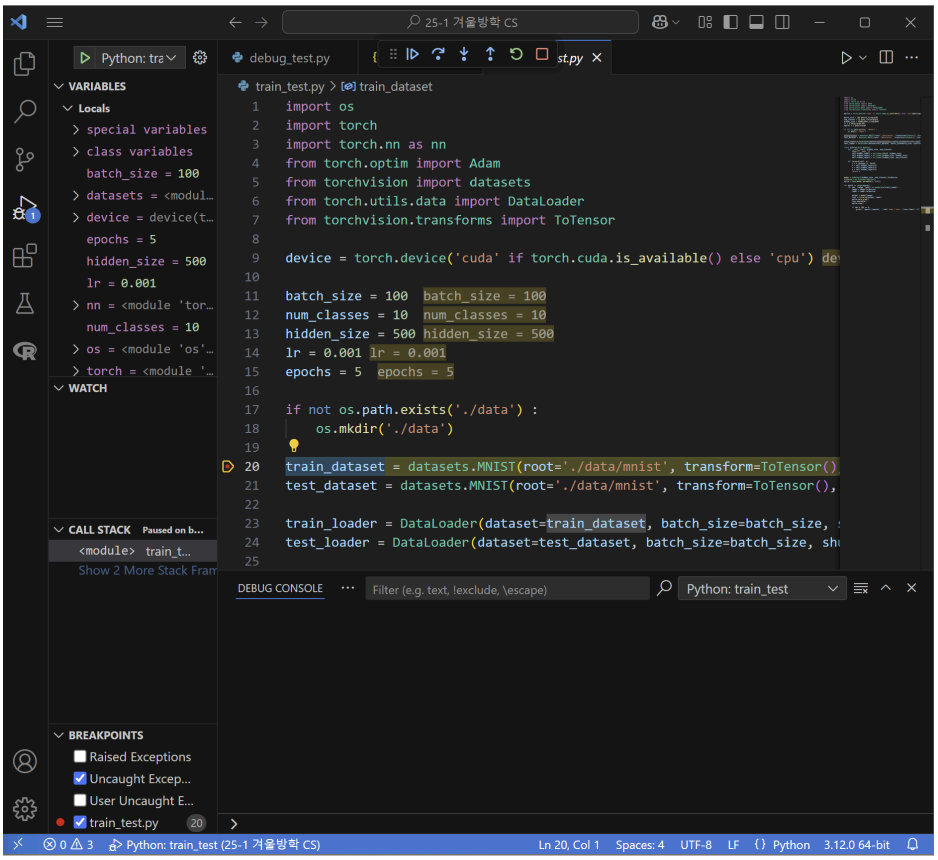
trunk-based:

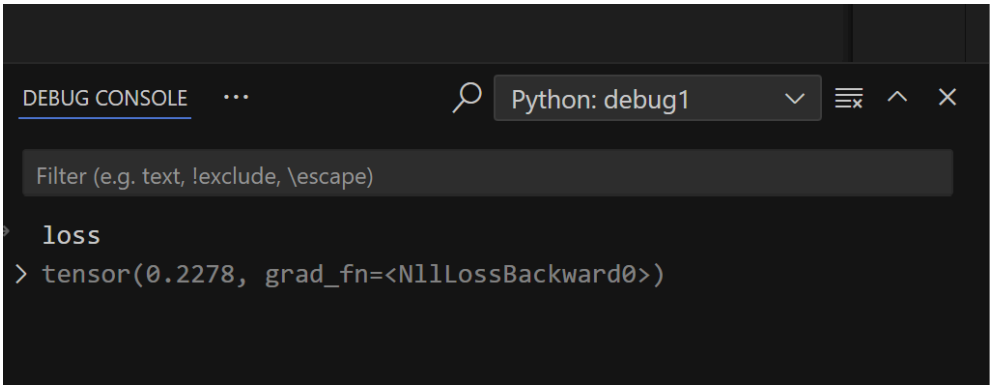


▼ 6주차

DebugMode.pdf

pathSeparator: 환경에 맞는 경로 seperator로 인식






```
 8
 9   image_size = 28  image_size = 28
10   batch_size == 100
```

Exception has occurred: NameError (note: full exception trace is shown but execution is paused at: _run_module_as_main)

```
name 'batch_size' is not defined
  File "C:\Users\kjhoe\OneDrive\바탕화~1-LAPTOP-VGOF9FJ7-25904429\KUBIG\25-1 겨울방학 CS\debug2.py", line 10, in <module>
    batch_size == 100
    ^^^^^^^^^^
  File "C:\Users\kjhoe\AppData\Local\Programs\Python\Python312\L
line 88, in _run_code
    exec(code, run_globals)
  File
```

```
48
49  for ep in epoch:  epoch = 5
```

Exception has occurred: TypeError (note: full exception trace is shown but execution is paused at: _run_module_as_main)

```
'int' object is not iterable
  File "C:\Users\kjhoe\OneDrive\바탕화~1-LAPTOP-VGOF9FJ7-25904429\KUBIG\25-1 겨울방학 CS\debug2.py", line 49, in <module>
    for idx, (image, label) in enumerate(train_loader):

        image = image.to(device)
```