

소프트웨어프로젝트 I

리눅스 (Linux) - 사용자 계정과 프로세스
- 표준 입출력

2021학년도 1학기

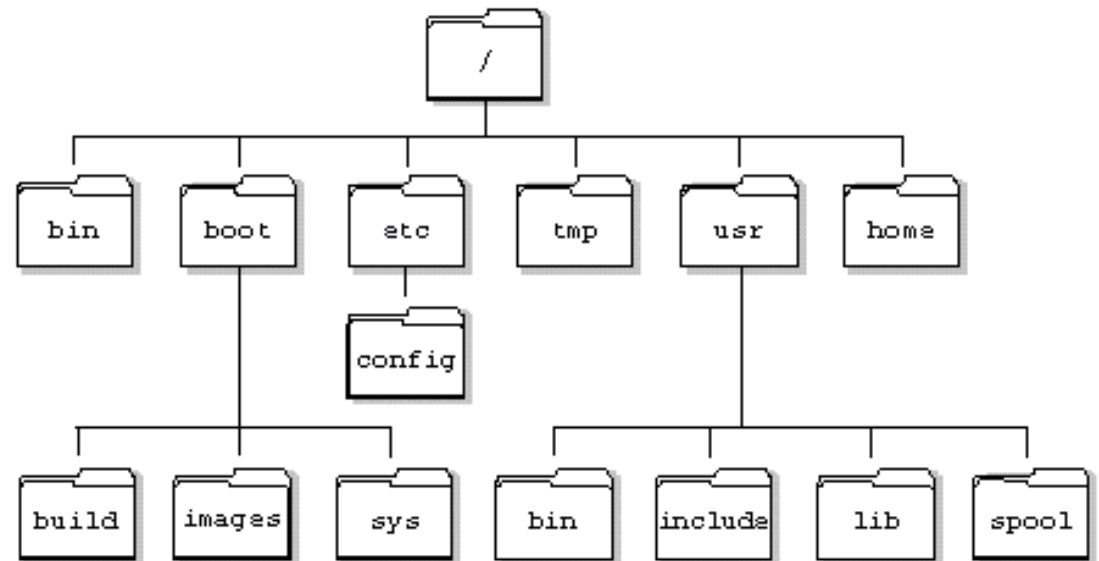
국민대학교 소프트웨어학부

지난 수업에는...

Files & Directories

- 파일
- 디렉토리

- 리눅스 파일 조작 실습



사용자, 사용자 계정

리눅스 시스템은 여러 사용자가 동시에 시스템을 사용하는 것을 지원

→ 다중 사용자 시스템 (multi-user system)

사용자는 시스템에 로그인하여 "프로세스" 를 통하여 작업을 실행

일반적인 사용자 (실제로 사람!) 도 있지만 그렇지 않은 사용자들도 있음

→ 시스템 프로세스 등을 실행하는 데 이용하기 위함

어떤 사용자들이 있는지 보자

\$ more /etc/passwd

시스템에 등록된 사용자들을 기록한 파일

- 리눅스 배포판에 따라서는 (아무나) 읽지 못하게 되어 있을 수도

```
sheayun@sheayun-ubuntu: ~  
root:x:0:0:root:/root:/bin/bash  
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin  
bin:x:2:2:bin:/bin:/usr/sbin/nologin  
sys:x:3:3:sys:/dev:/usr/sbin/nologin  
sync:x:4:65534:sync:/bin:/bin/sync  
games:x:5:60:games:/usr/games:/usr/sbin/nologin  
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin  
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin  
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin  
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin  
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin  
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin  
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin  
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin  
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin  
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin  
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin  
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin  
systemd-timesync:x:100:100:systemd Time Synchronization:/etc/systemd:/bin/false
```

아주 특별한 사용자
(잠시 후에 만나보자)

starblood:x:1000:1000:starblood,,,:/home/starblood:/bin/bash

사용자이름

패스워드 UID

GID

사용자 ID 정보

홈디렉토리

명령어 shell

내 계정도 들어있는지?

```
postfix:x:112:119::/var/spool/postfix:/bin/false
dovecot:x:113:121:Dovecot mail server,,,:/usr/lib/dovecot:/bin/false
dovenull:x:114:122:Dovecot login user,,,:/nonexistent:/bin/false
sshd:x:115:65534::/var/run/sshd:/usr/sbin/nologin
sheayun:x:1000:1000:sheayun,,,:/home/sheayun:/bin/bash
lightdm:x:116:125:Light Display Manager:/var/lib/lightdm:/bin/false
whoopsie:x:117:127::/nonexistent:/bin/false
avahi-autoipd:x:118:128:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/bin/false
avahi:x:119:129:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/bin/false
```

시스템은 모든 사용자를 uid (user id) 라고 불리는

숫자로 인식 (그러나 보기 편하게 하기 위해 이름도 붙여 있음)

login-name:passwd:uid:gid:comment:homedir:login-shell

아주 특별한 uid: 0 (root)

사람이 아닌 사용자들

\$ ls -l /var/log

```
-rw-r----- 1 syslog adm 16663 4 6 16:48 syslog
-rw-r----- 1 syslog adm 8155014 4 6 16:29 syslog.1
-rw-r----- 1 syslog adm 29179 4 4 14:55 syslog.2.gz
-rw-r----- 1 syslog adm 376041 7 10 2017 syslog.3.gz
-rw-r----- 1 syslog adm 36790 5 16 2017 syslog.4.gz
-rw-r----- 1 syslog adm 75176 2 24 2017 syslog.5.gz
-rw-r----- 1 syslog adm 87942 2 22 2017 syslog.6.gz
```

\$ ps aux

```
www-data 2552 0.0 0.0 258300 912 ? S 16:29 0:00 apache2
www-data 2553 0.0 0.0 258300 912 ? S 16:29 0:00 apache2
www-data 2554 0.0 0.0 258300 912 ? S 16:29 0:00 apache2
www-data 2555 0.0 0.0 258300 912 ? S 16:29 0:00 apache2
www-data 2556 0.0 0.0 258300 912 ? S 16:29 0:00 apache2
```

사용자 그룹

```
sheayun@sheayun-ubuntu:~$ id
uid=1000(sheayun) gid=1000(sheayun) 그룹들=1000(sheayun),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),110(lxd),114(sambashare),124(lpadmin)
sheayun@sheayun-ubuntu:~$
```

↑
\$ id

\$ more /etc/group

→

```
root:x:0:
daemon:x:1:
bin:x:2:
sys:x:3:
adm:x:4:syslog,sheayun
tty:x:5:
disk:x:6:
lp:x:7:
mail:x:8:
news:x:9:
uucp:x:10:
man:x:12:
proxy:x:13:
kmem:x:15:
dialout:x:20:
fax:x:21:
voice:x:22:
cdrom:x:24:sheayun
floppy:x:25:
tape:x:26:
sudo:x:27:sheayun
audio:x:29:pulse
dip:x:30:sheayun
--More-- (25%)
```

모든 사용자는 하나 이상의
사용자 그룹에 속해 있음

group-name:x:gid:members

파일 소유자 정보

모든 파일 (디렉토리 포함) 은 user owner 와 group owner 가 정해져 있음

```
-rw-r----- 1 syslog      adm      16663  4  6 16:48 syslog
-rw-r----- 1 syslog      adm    8155014  4  6 16:29 syslog.1
-rw-r----- 1 syslog      adm     29179  4  4 14:55 syslog.2.gz
-rw-r----- 1 syslog      adm    376041  7 10 2017 syslog.3.gz
-rw-r----- 1 syslog      adm     36790  5 16 2017 syslog.4.gz
-rw-r----- 1 syslog      adm     75176  2 24 2017 syslog.5.gz
-rw-r----- 1 syslog      adm     87942  2 22 2017 syslog.6.gz
```

위에 나열한 파일들에 대하여:  user permission 과 group permission 적용됨:

user owner = syslog

user - rw (읽기/쓰기)

group owner = adm

group - r (읽기만)

other - None (없음)

파일 소유자를 변경해 보자 - chown

1. 홈 디렉토리 아래에 "testdir" 라는 이름으로 새 디렉토리를 만들고
2. /etc/vim 아래에 있는 모든 파일을 복사

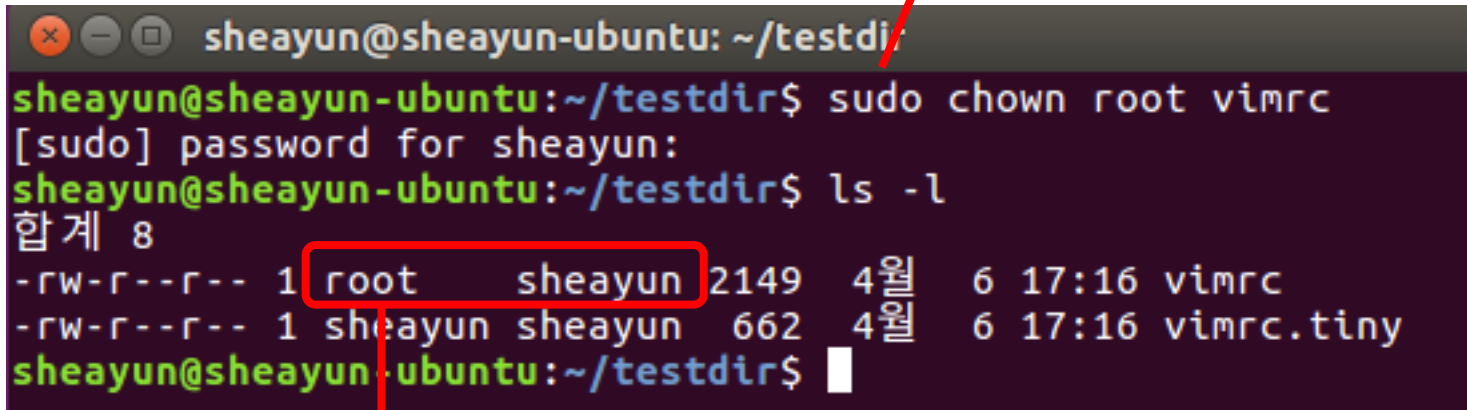
```
sheayun@sheayun-ubuntu: ~/testdir
sheayun@sheayun-ubuntu:~/testdir$ ls -al
합계 16
drwxrwxr-x  2 sheayun sheayun 4096  4월  6 17:16 .
drwxr-xr-x 21 sheayun sheayun 4096  4월  6 17:15 ..
-rw-r--r--  1 sheayun sheayun 2149  4월  6 17:16 vimrc
-rw-r--r--  1 sheayun sheayun  662  4월  6 17:16 vimrc.tiny
sheayun@sheayun-ubuntu:~/testdir$
```

3. "vimrc" 파일의 소유자를 root 로 변경해 보자 → chown root vimrc

```
sheayun@sheayun-ubuntu: ~/testdir
sheayun@sheayun-ubuntu:~/testdir$ chown root vimrc
chown: 'vimrc'의 소유자 변경: 명령을 허용하지 않음
sheayun@sheayun-ubuntu:~/testdir$
```

수퍼유저 (root) 권한이 필요 - sudo

이 명령은 내 권한이 아니라 root 의 권한으로 실행해 주시오.



```
sheayun@sheayun-ubuntu: ~/testdir
sheayun@sheayun-ubuntu:~/testdir$ sudo chown root vimrc
[sudo] password for sheayun:
sheayun@sheayun-ubuntu:~/testdir$ ls -l
합계 8
-rw-r--r-- 1 root sheayun 2149 4월 6 17:16 vimrc
-rw-r--r-- 1 sheayun sheayun 662 4월 6 17:16 vimrc.tiny
sheayun@sheayun-ubuntu:~/testdir$
```

A terminal window screenshot showing the execution of the command `sudo chown root vimrc`. The prompt is `sheayun@sheayun-ubuntu: ~/testdir`. The command is entered, and the system prompts for the password: `[sudo] password for sheayun:`. After the password is entered, the command is executed. The user then runs `ls -l` to check the file permissions. The output shows two files: `vimrc` and `vimrc.tiny`. The `vimrc` file is now owned by `root` (indicated by `root sheayun` in the output), while `vimrc.tiny` remains owned by `sheayun`. A red box highlights the `root` user in the output, and a red arrow points from the text above to the `sudo` command in the terminal.

파일의 user owner 가 root 로 변경되었음

파일의 그룹 소유자 변경 - chgrp

```
sheayun@sheayun-ubuntu: ~/testdir
sheayun@sheayun-ubuntu:~/testdir$ sudo chgrp adm vimrc
sheayun@sheayun-ubuntu:~/testdir$ ls -l
합계 8
-rw-r--r-- 1 root adm 2149 4월 6 17:16 vimrc
-rw-r--r-- 1 sheayun sheayun 662 4월 6 17:16 vimrc.tiny
sheayun@sheayun-ubuntu:~/testdir$
```

chgrp 또한 root 사용자만 행할 수 있음

하나의 명령으로
할 수도 있음

```
sheayun@sheayun-ubuntu: ~/testdir
sheayun@sheayun-ubuntu:~/testdir$ sudo chown root:adm vimrc.tiny
sheayun@sheayun-ubuntu:~/testdir$ ls -l
합계 8
-rw-r--r-- 1 root adm 2149 4월 6 17:16 vimrc
-rw-r--r-- 1 root adm 662 4월 6 17:16 vimrc.tiny
sheayun@sheayun-ubuntu:~/testdir$
```

chown 과 chmod 에 대한 정리

- 명령어 설명

- chown: 파일과 디렉토리의 소유자 설정
 - 소유자 관련 설정은 수퍼유저만 변경할 수 있음
- chmod: 파일과 디렉토리의 사용권한 설정
 - 파일 소유자와 수퍼유저에게 권한이 주어짐

- 사용법

- chown [option] [owner] [:group] 파일
- chmod [option] mode 파일
- [option]
 - -R 재귀적으로 하위 디렉토리까지 변경
- mode는 다음과 같이 표현됨: Mrwxrwxrwx
 - M은 파일 형태: 파일, 디렉토리, 장치, 심볼릭 링크 중하나
 - 첫 번째 rwx: 파일 소유자가 읽고, 쓰고, 실행할 수 있는 권한
 - 두 번째 rwx: 그룹 소유자에 속한 다른 사용자가 읽고, 쓰고, 실행할 수 있는 권한
 - 세 번째 rwx: 사용자와 관계 없는 다른 사용자가 읽고, 쓰고, 실행할 수 있는 권한

디렉토리에 대한 실행 (x) 권한

→ 해당 디렉토리에 대한 진입

어떤 파일에 접근하려면

/로부터 그 파일에까지 이르는

모든 디렉토리의 진입 권한 필요

수퍼유저 권한을 얻자 - sudo

- 명령어 설명

- 명령어를 다른 사용자의 권한으로 실행시킴
- 다른 사용자로 수퍼유저 (root) 나 다른 일반 사용자 모두 가능함
- 수퍼유저 권한을 얻기 위해서는 /etc/sudoers 파일에서 권한을 설정해 주어야 함
 - 선택된 사용자만 수퍼유저 권한을 얻을 수 있음

- 사용법

- `sudo -u other-user /bin/bash`
 - 사용자가 other-user의 권한으로 셸을 호출
- `sudo /bin/bash`
 - 수퍼 유저의 권한으로 셸을 호출
- `sudo` 명령: 수퍼 유저 권한으로 명령 실행

sudo 실습

```
root@sheayun-ubuntu: ~/testdir
sheayun@sheayun-ubuntu:~/testdir$ sudo /bin/bash
root@sheayun-ubuntu:~/testdir# head -2 /var/log/syslog
Apr  6 16:29:52 sheayun-ubuntu anacron[867]: Job `cron.daily' terminated
Apr  6 16:29:52 sheayun-ubuntu anacron[867]: Normal exit (1 job run)
root@sheayun-ubuntu:~/testdir# ls -l /var/log/syslog
-rw-r----- 1 syslog adm 17947 4월  6 17:34 /var/log/syslog
root@sheayun-ubuntu:~/testdir#
```

수퍼유저 (root) 에게는

이런 것은 통하지 않아!

시스템의 설정/관리 등에

수퍼유저 권한이 요구되는 것들이 많음

→ 이후에 실습하게 됨

(주의) 수퍼유저 권한으로의 실행을

절대로 남발하지 말자!

(습관이 되면 곤란)

실습했던 과거는 깔끔히 정리

```
sheayun@sheayun-ubuntu: ~/testdir
sheayun@sheayun-ubuntu:~/testdir$ sudo /bin/bash
root@sheayun-ubuntu:~/testdir# head -2 /var/log/syslog
Apr  6 16:29:52 sheayun-ubuntu anacron[867]: Job `cron.daily' terminated
Apr  6 16:29:52 sheayun-ubuntu anacron[867]: Normal exit (1 job run)
root@sheayun-ubuntu:~/testdir# ls -l /var/log/syslog
-rw-r----- 1 syslog adm 17947  4월  6 17:34 /var/log/syslog
root@sheayun-ubuntu:~/testdir# exit
exit
sheayun@sheayun-ubuntu:~/testdir$
```

```
sheayun@sheayun-ubuntu: ~
sheayun@sheayun-ubuntu:~/testdir$ cd
sheayun@sheayun-ubuntu:~$ rm -rf testdir
sheayun@sheayun-ubuntu:~$ ls
Desktop    Downloads  Pictures   Templates  test.txt
Documents  Music      Public     Videos
sheayun@sheayun-ubuntu:~$
```

파일을 지우는 데에는
그 파일에 대한 쓰기 권한이
필요하지 않음! (그럼?)

프로세스

- 사용자가 요청하는 일을 수행하기 위해서 리눅스 커널은 프로세스를 생성함
 - ls, cat, find 와 같은 명령어들을 수행하기 위해 프로세스가 생성됨
- 생성된 프로세스는 리눅스가 처리하는 일의 단위
- 사용자 프로세스 / 시스템 프로세스 / 데몬 프로세스등의 종류가 있음

어떤 프로세스들이 있는지 보자 - ps

```
sheayun@sheayun-ubuntu: ~  
sheayun@sheayun-ubuntu:~$ ps  
  PID TTY          TIME CMD  
 20729 pts/6        00:00:00 bash  
 21537 pts/6        00:00:00 ps  
sheayun@sheayun-ubuntu:~$
```

이 명령을 실행한 로그인 셸

아, ps 도 프로세스로 만들어졌네!

그런데, 에게? 시스템에 프로세스가 고작 이것밖에 없다구?

→ ps aux 해보자!

프로그램 실행하면 프로세스가 생성됨

```
sheayun@sheayun-ubuntu: ~/ptest
sheayun@sheayun-ubuntu:~/ptest$ cat infloop.py
while 1:
    pass
sheayun@sheayun-ubuntu:~/ptest$ python infloop.py

sheayun@sheayun-ubuntu: ~
sheayun@sheayun-ubuntu:~$ ps aux | grep infloop
sheayun  21608 75.2  0.5 25840 5972 pts/6    R+   17:54   0:05 python infloop.py
sheayun  21610  0.0  0.0 15508  932 pts/2    S+   17:54   0:00 grep --color=auto infloop
sheayun@sheayun-ubuntu:~$
```

무한 반복 순환문이 실행
→ 끝나지 않는 프로그램

다른 터미널에서 ps 를 통해 프로세스 조회
아, grep 도 별도 프로세스가 실행되었군!

실행 중인 프로세스에 대한 정보 - top

```
sheayun@sheayun-ubuntu: ~  
top - 17:57:02 up 1:32, 1 user, load average: 1.08, 0.65, 0.28  
Tasks: 200 total, 2 running, 198 sleeping, 0 stopped, 0 zombie  
%Cpu(s): 98.9 us, 0.7 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.4 si, 0.0 st  
KiB Mem : 1016012 total, 129640 free, 556040 used, 330332 buff/cache  
KiB Swap: 1048572 total, 652452 free, 396120 used. 259316 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
21608	sheayun	20	0	25840	5972	3580	R	96.0	0.6	2:51.68	python
20371	sheayun	20	0	1292136	158232	41928	S	2.3	15.6	2:52.20	compiz
1971	root	20	0	374492	45204	12096	S	1.3	4.4	0:41.81	Xorg
1	root	20	0	185584	4664	3064	S	0.0	0.5	0:02.29	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:01.27	ksoftirqd/0
5	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	kworker/0:0H

CPU 점유율 96.0% ???

언제나 실습 과거는 깔끔히 정리

```
sheayun@sheayun-ubuntu: ~/ptest
sheayun@sheayun-ubuntu:~/ptest$ cat infloop.py
while 1:
    pass

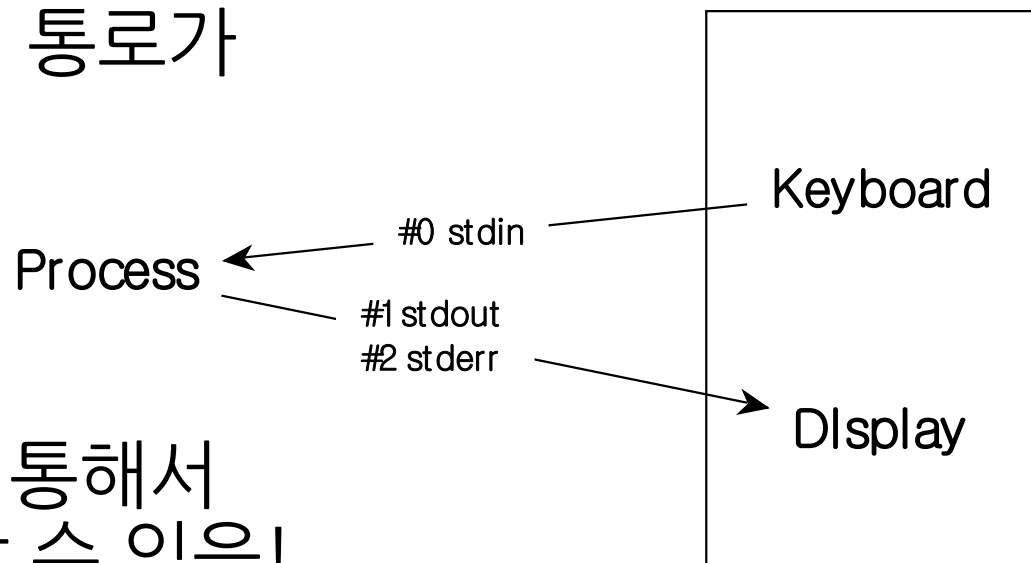
sheayun@sheayun-ubuntu:~/ptest$ python infloop.py
^C
Traceback (most recent call last):
  File "infloop.py", line 2, in <module>
    pass
KeyboardInterrupt
sheayun@sheayun-ubuntu:~/ptest$
```

ctrl-C 키를 눌러서

무한 반복을 실행하고 있던 프로그램을 강제 종료

프로세스의 표준 입출력

- 표준 입출력은 컴퓨터 프로그램과 그 환경 사이에 미리 연결된 입출력 통로를 가리킴
 - 대부분의 운영체제에서 사용
- 일반적으로 유닉스에서 동작하는 프로그램은 실행시 3개의 통로가 자동으로 열림
 - 표준 입력 (stdin)
 - 표준 출력 (stdout)
 - 표준 오류 (stderr)
- Redirection, Pipeline을 통해서 입/출력 흐름을 재정의 할 수 있음!



표준 출력의 Redirection

- Redirection
 - 표준 출력으로 나가는 내용들의 방향을 바꾸는 것
- > 사용
 - 어떤 명령의 결과를 파일로 보내고 싶다면 그 명령어 다음에 ">" 문자를 사용하고 파일 이름을 사용
 - 예) `$ ls -l > dir_list`
- >> 사용
 - 표준 출력 내용을 기존 파일에 append 하려면 ">>" 를 사용
 - 예) `$ date >> dir_list`

표준 출력의 Redirection

```
sheayun@localhost: ~/testdir
sheayun@localhost:~/testdir$ ls -l
total 0
sheayun@localhost:~/testdir$ ls -l ~ > dir_list
sheayun@localhost:~/testdir$ ls -l
total 4
-rw-rw-r-- 1 sheayun sheayun 591 4월 10 11:50 dir_list
sheayun@localhost:~/testdir$ cat dir_list
total 48
drwxr-xr-x 2 sheayun sheayun 4096 3월 29 17:32 Desktop
drwxr-xr-x 2 sheayun sheayun 4096 3월 29 17:32 Documents
drwxr-xr-x 2 sheayun sheayun 4096 3월 29 17:32 Downloads
-rw-r--r-- 1 sheayun sheayun 8980 4월 20 2016 examples.desktop
drwxr-xr-x 2 sheayun sheayun 4096 3월 29 17:32 Music
drwxr-xr-x 2 sheayun sheayun 4096 3월 29 17:32 Pictures
drwxr-xr-x 2 sheayun sheayun 4096 3월 29 17:32 Public
drwxr-xr-x 2 sheayun sheayun 4096 3월 29 17:32 Templates
drwxrwxr-x 2 sheayun sheayun 4096 4월 10 11:50 testdir
drwxr-xr-x 2 sheayun sheayun 4096 3월 29 17:32 Videos
sheayun@localhost:~/testdir$
```

표준 출력의 Redirection

```
sheayun@localhost: ~/testdir
sheayun@localhost:~/testdir$ date
2018. 04. 10. (화) 11:51:45 KST
sheayun@localhost:~/testdir$ date >> dir_list
sheayun@localhost:~/testdir$ cat dir_list
total 48
drwxr-xr-x 2 sheayun sheayun 4096 3월 29 17:32 Desktop
drwxr-xr-x 2 sheayun sheayun 4096 3월 29 17:32 Documents
drwxr-xr-x 2 sheayun sheayun 4096 3월 29 17:32 Downloads
-rw-r--r-- 1 sheayun sheayun 8980 4월 20 2016 examples.desktop
drwxr-xr-x 2 sheayun sheayun 4096 3월 29 17:32 Music
drwxr-xr-x 2 sheayun sheayun 4096 3월 29 17:32 Pictures
drwxr-xr-x 2 sheayun sheayun 4096 3월 29 17:32 Public
drwxr-xr-x 2 sheayun sheayun 4096 3월 29 17:32 Templates
drwxrwxr-x 2 sheayun sheayun 4096 4월 10 11:50 testdir
drwxr-xr-x 2 sheayun sheayun 4096 3월 29 17:32 Videos
2018. 04. 10. (화) 11:51:49 KST
sheayun@localhost:~/testdir$
```


표준 입력의 Redirection

- 표준 입력(즉, 키보드 입력)을 파일로 Redirection시키고 싶은 경우
 - 셸에게 그 프로그램이 데이터 입력을 키보드로부터가 아니라 파일로부터 읽어 들이도록 지시
- < 사용
 - 실행하고자 하는 프로그램(명령어) 다음에 "<" 문자를 사용한 후 입력 파일의 이름을 적용
 - \$ sort < input
- 표준 입력과 출력을 함께 Redirection 하는 것도 가능
 - \$ sort < unsorted > sorted

표준 입력의 Redirection

```
sheayun@localhost: ~/testdir
sheayun@localhost:~/testdir$ cat input
1
3
12
27
9
6
sheayun@localhost:~/testdir$ sort < input
1
12
27
3
6
9
sheayun@localhost:~/testdir$
```

```
sheayun@localhost: ~/testdir
sheayun@localhost:~/testdir$ cat input
1
3
12
27
9
6
sheayun@localhost:~/testdir$ sort -n < input
1
3
6
9
12
27
sheayun@localhost:~/testdir$
```

표준 입출력의 Redirection

표준 입력은:

파일 "input" 으로부터

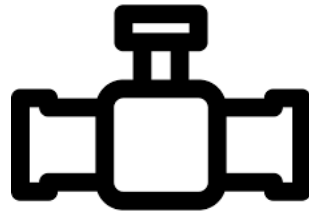
표준 출력은:

파일 "output" 으로

```
sheayun@localhost: ~/testdir
sheayun@localhost:~/testdir$ cat input
1
3
12
27
9
6
sheayun@localhost:~/testdir$ sort < input > output
sheayun@localhost:~/testdir$ cat output
1
12
27
3
6
9
sheayun@localhost:~/testdir$
```

표준 입출력의 파이프라인

- 한 프로세스의 표준 출력을 다른 프로세스의 표준 입력에 연결



- "|" 사용
 - 프로그램(명령어) 이름 다음에 "|" (수직 막대)를 사용
 - 예) sort프로그램의 결과를 lpr (데이터를 인쇄하는 명령어) 로 보내려면:
 - `$ sort input | lpr`

표준 입출력의 파이프라인

```
sheayun@localhost: ~/testdir
sheayun@localhost:~/testdir$ ls -al
total 20
drwxrwxr-x  2 sheayun sheayun 4096  4월 10 12:13 .
drwxr-xr-x 17 sheayun sheayun 4096  4월 10 11:49 ..
-rw-rw-r--  1 sheayun sheayun  624  4월 10 11:51 dir_list
-rw-rw-r--  1 sheayun sheayun   14  4월 10 12:11 input
-rw-rw-r--  1 sheayun sheayun   14  4월 10 12:13 output
sheayun@localhost:~/testdir$ ls -al | wc
    6    47   286
sheayun@localhost:~/testdir$
```

ls -al 결과 표준 출력의 라인 수, 단어 수, 글자 수

표준 출력의 파이프라인 분리 - tee

- T자 관(tee)
 - 자신에게 들어오는 입력 데이터를 표준 출력 (stdout) 으로 출력할 뿐 아니라 지정된 파일로도 보내는 장치
- T관을 만들기 위해서는 tee 명령을 사용
 - `$ tee [-a] files` // "-a" 는 append 를 의미함
- 일반적으로 파이프라인과 함께 사용하여, 출력 결과를 화면 (표준 출력) 으로 표시함과 동시에 파일로 저장하기 위하여 사용함
 - `$ sort < unsorted | tee sorted`



표준 출력의 파이프라인 분리 - tee

```
sheayun@localhost: ~/testdir
sheayun@localhost:~/testdir$ ls -l
total 12
-rw-rw-r-- 1 sheayun sheayun 624  4월 10 11:51 dir_list
-rw-rw-r-- 1 sheayun sheayun  14  4월 10 12:11 input
-rw-rw-r-- 1 sheayun sheayun  14  4월 10 12:13 output
sheayun@localhost:~/testdir$ cat input
1
3
12
27
9
6
sheayun@localhost:~/testdir$ sort < input | tee sorted
1
12
27
3
6
9
sheayun@localhost:~/testdir$ ls -l
total 16
-rw-rw-r-- 1 sheayun sheayun 624  4월 10 11:51 dir_list
-rw-rw-r-- 1 sheayun sheayun  14  4월 10 12:11 input
-rw-rw-r-- 1 sheayun sheayun  14  4월 10 12:13 output
-rw-rw-r-- 1 sheayun sheayun  14  4월 10 12:26 sorted
sheayun@localhost:~/testdir$ cat sorted
1
12
27
3
6
9
sheayun@localhost:~/testdir$
```

표준 입력은:

파일 "input" 으로부터

표준 출력은:

원래의 표준 출력 (화면) 으로도

표시함과 동시에,

같은 내용을 파일 "sorted" 으로도 저장