5. Utilities

Hyunchan, Park

http://oslab.jbnu.ac.kr

Division of Computer Science and Engineering

Jeonbuk National University

학습 내용

- 검색: find and grep
- 압축: tar and compressions
- 기타

개인 과제 5: 실습 및 C언어 복습

- 실습 과제
  - 실습 내용에서 다루는 명령어를 모두 입력하고, 그 결과를 확인할 것
    - 동영상에서 수행한 내용
  - 제출 방법
    - Old LMS, 개인 과제 5
    - Xshell 로그 파일 1개 제출
    - 파일 명: 학번.txt
- C언어 복습 과제
  - JOTA: hw5-1, hw5-2, hw5-3 문제 수행
- 제출 기한
  - 10/12 (월) 23:59 (지각 감점: 5%p / 12H, 1주 이후 제출 불가)

검색: find and grep

find 명령어

- find 명령어
  - 파일 이름이나 속성을 이용하여 해당하는 파일을 찾는다.

- 사용법

```
$ find 디렉터리 [-옵션]
```

옵션의 검색 조건에 따라 지정된 디렉터리 아래에서 해당되는 파일들을 모두 찾아 출력한다.

- 실습 내용: root 권한으로 /var 로 이동하여 진행
  - $ sudo -s
  - # cd /var

find 명령어

- 예

```
$ find ~ -name hw1.c -print
(~ 디렉토리에서 이름이 hw1.c 인 파일을 찾아 출력하시오)
/home/ubuntu/hw1.c

$ find ~ -name hw1 -ls
(~ 디렉토리에서 이름이 hw1 인 파일을 대상으로 ls 을 수행하시오)
283172      4 drwxr-xr-x   2 ubuntu   ubuntu       4096 Sep 22 15:34 /home/ubuntu/hw1

$ find /usr -name "*.c" -print
(/usr 디렉토리에서 확장자가 c 인 모든 파일을 출력하시오)
 * 따옴표에 유의! * 등 regular expression 을 입력할 때
   꼭 따옴표 혹은 작은 따옴표 사용
```

find 명령어: 검색 조건

검색 조건 및 처리 방법	설명
-name 파일명	파일명으로 찾는다.
-atime +n	접근 시간이 n일 이전인 파일을 찾는다.
-atime -n	접근 시간이 n일 이내인 파일을 찾는다.
-mtime +n	n일 이전에 수정된 파일을 찾는다.
-mtime -n	n일 이내에 수정된 파일을 찾는다.
-perm nnn	접근권한이 nnn인 파일을 찾는다.
-type x	파일 종류가 x인 파일들을 찾는다.
-size n	크기가 n 블록(512바이트)인 파일들을 찾는다.
-links n	링크 개수가 n인 파일들을 찾는다.
-user 사용자명	파일의 소유자가 사용자명인 파일을 찾는다.
-group 그룹명	그룹명을 갖는 그룹에 속한 파일을 찾는다.
-print	찾은 파일의 절대 경로명을 화면에 출력한다.
-ls	찾은 파일에 대해 ls -dils 명령어 실행 결과를 출력한다.
-exec cmd {};	찾은 파일들에 대해 cmd 명령어를 실행한다.

find 명령어: 검색 조건

- 파일의 접근권한(-perm)으로 검색

```
$ find . -perm 700 -ls
```

- 파일의 접근 시간(-atime) 혹은 수정 시간(-mtime)으로 검색

  +n: 현재 시각을 기준으로 n일 이상 전

  n: 현재 시각을 기준으로 n일 전

  -n: 현재 시각을 기준으로 n일 이내

```
$ find . -atime +30 -ls
$ find . -mtime -7 -ls
```

find 명령어: 검색 조건

- 파일의 소유자(-user)로 검색

```
$ find . -user ubuntu –print
```

- 파일 크기(-size)로 검색

```
$ find . -size +16k -ls
```

- 파일 종류(-type)로 검색

```
d : 디렉터리          f: 일반 파일          l: 심볼릭 링크
b: 블록 장치 파일     c: 문자 장치 파일     s: 소켓 파일
```

```
$ find ~ -type d –print
```

find 명령어: 검색 조건 조합

- find 명령어는 여러 검색 옵션을 조합해서 사용할 수 있다.

- 예

```
$ find . -type d -perm 700 -print
$ find . -name “*.log” -size +16k -ls
```

find 명령어: 검색된 파일 처리

- find 명령어의 -exec 옵션
  - 검색한 모든 파일을 대상으로 동일한 작업(명령어)을 수행

- 예

```
$ find . -name core -exec rm -i {} \;
```

(core 라는 이름을 가진 파일을 찾아 rm –i 명령 실행.
주의! 수행하지 말 것!)

```
$ find . -name "*.c" -atime +3 -exec ls -l {} \;
```

(확장자가 c 이고, access time이 3일 이상 지난 파일을 찾아 ls –l 명령 실행)

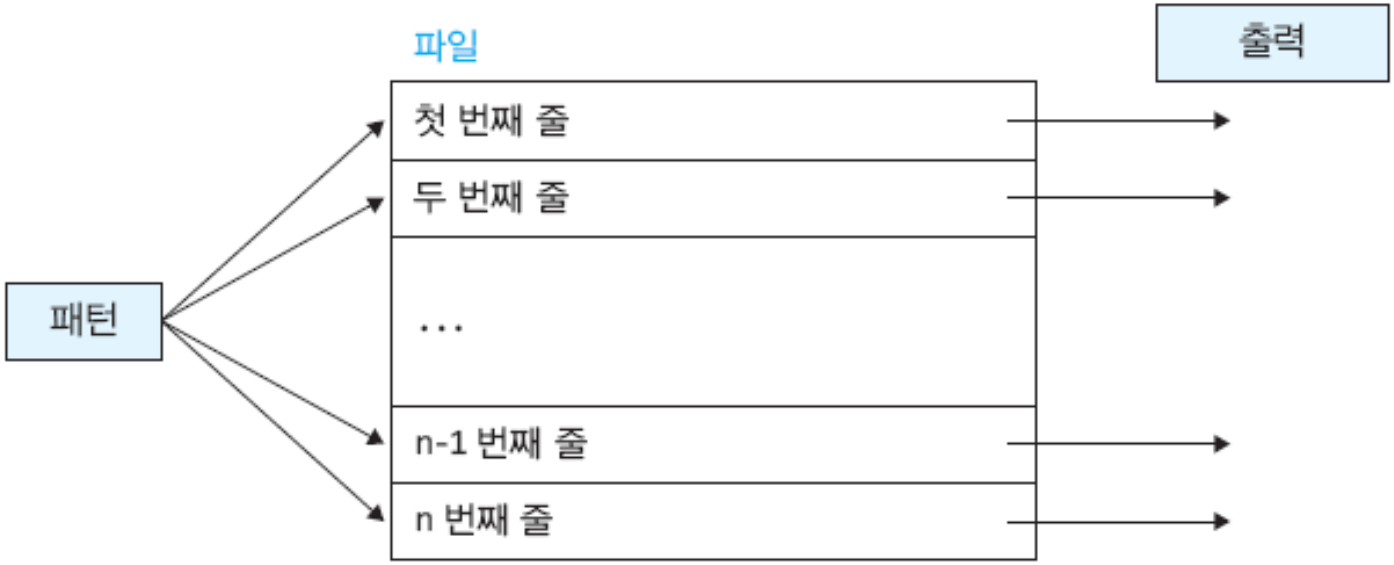
grep 명령어

- 사용법

```
$ grep 패턴 파일*
```

파일(들)을 대상으로 지정된 패턴의 문자열을 검색하고,

해당 문자열을 포함하는 줄들을 출력한다.



grep 명령어의 옵션

옵션	기능
-i	대소문자를 무시하고 검색한다.
-l	해당 패턴이 들어있는 파일 이름을 출력한다.
-n	각 줄의 줄번호도 함께 출력한다.
-v	명시된 패턴을 포함하지 않는 줄을 출력한다.
-c	패턴과 일치하는 줄 수를 출력한다.
-w	패턴이 하나의 단어로 된 것만 검색한다.
-R	하위 디렉토리의 파일들을 recursive 하게 검색한다.

실습 내용: grep

- http://textfiles.com/stories/
  - 영어 동화 스토리 중에서 세 개 골라, 예제 파일로 사용해보자
  - 선택한 동화의 제목에 오른 클릭-> 링크 복사 후, wget 이용
- Wget - The non-interactive network downloader.
  - 웹 주소를 이용해 파일 다운로드
  - 사용법: wget [web url]
- Grep 을 이용해 이후에 나오는 옵션들을 실습해볼 것

```
ubuntu@test2:~$ wget http://textfiles.com/stories/adv_alad.txt
--2020-09-19 13:46:51--  http://textfiles.com/stories/adv_alad.txt
Resolving textfiles.com (textfiles.com)... 208.86.224.90
Connecting to textfiles.com (textfiles.com)|208.86.224.90|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 9908 (9.7K) [text/plain]
Saving to: ‘adv_alad.txt’

adv_alad.txt                                                    100%[======

2020-09-19 13:46:51 (84.9 MB/s) - ‘adv_alad.txt’ saved [9908/9908]

ubuntu@test2:~$ mv adv_alad.txt aladin.txt
ubuntu@test2:~$ ls -al aladin.txt
-rw-rw-r-- 1 ubuntu ubuntu 9908 Aug  1  1999 aladin.txt
ubuntu@test2:~$
```

grep 명령어

- $ grep with you.txt (you.txt 대상으로 with 문자열 검색)

```
Until you come and sit awhile with me
There is no life - no life without its hunger;
But when you come and I am filled with wonder,
```

- $grep -w with you.txt (-w : 단어 단위로 일치하는 경우만)

```
Until you come and sit awhile with me
But when you come and I am filled with wonder,
```

- $grep -n with you.txt (-n : 앞에 줄 번호 표시)

```
4:Until you come and sit awhile with me
15:There is no life - no life without its hunger;
17:But when you come and I am filled with wonder,
```

grep 명령어

- $grep -i when you.txt

```
When I am down and, oh my soul, so weary
When troubles come and my heart burdened be
I am strong, when I am on your shoulders
But when you come and I am filled with wonder,
```

- $grep -v raise you.txt

```
When I am down and, oh my soul, so weary
When troubles come and my heart burdened be
Then, I am still and wait here in the silence
Until you come and sit awhile with me
I am strong, when I am on your shoulders
There is no life - no life without its hunger;
Each restless heart beats so imperfectly;
But when you come and I am filled with wonder,
Sometimes, I think I glimpse eternity
```

중복 옵션 사용 예

```
ubuntu@test2:~$ cp /etc/services sample.txt
ubuntu@test2:~$ grep -in tcp sample.txt
4:# port number for both TCP and UDP; hence, officially ports have two entries
13:tcpmux               1/tcp                           # TCP port service multiplexer
14:echo         7/tcp
16:discard              9/tcp           sink null
18:systat               11/tcp          users
19:daytime              13/tcp
21:netstat              15/tcp
22:qotd         17/tcp          quote
23:chargen              19/tcp          ttytst source
25:ftp-data     20/tcp
26:ftp          21/tcp
28:ssh          22/tcp                          # SSH Remote Login Protocol
29:telnet               23/tcp
30:smtp         25/tcp          mail
31:time         37/tcp          timserver
33:whois                43/tcp          nicname
34:tacacs               49/tcp                          # Login Host Protocol (TACACS)
36:domain               53/tcp                          # Domain Name Server
41:gopher               70/tcp                          # Internet Gopher
42:finger               79/tcp
43:http         80/tcp          www             # WorldWideWeb HTTP
44:kerberos     88/tcp          kerberos5 krb5 kerberos-sec     # Kerberos v5
46:iso-tsap     102/tcp         tsap            # part of ISODE
47:acr-nema     104/tcp         dicom           # Digital Imag. & Comm. 300
48:pop3         110/tcp         pop-3           # POP version 3
49:sunrpc               111/tcp         portmapper      # RPC 4.0 portmapper
51:auth         113/tcp         authentication tap ident
52:nntp         119/tcp         readnews untp   # USENET News Transfer Protocol
54:epmap                135/tcp         loc-srv         # DCE endpoint resolution
55:netbios-ns   137/tcp                         # NETBIOS Name Service
57:netbios-dgm  138/tcp                         # NETBIOS Datagram Service
59:netbios-ssn  139/tcp                         # NETBIOS session service
```

```
ubuntu@test2:~$ grep -l tcp sample.txt
sample.txt
ubuntu@test2:~$ grep -l tcp *
grep: hw1: Is a directory
grep: hw10: Is a directory
grep: hw11: Is a directory
grep: hw12: Is a directory
grep: hw2: Is a directory
grep: hw3: Is a directory
grep: hw4: Is a directory
grep: hw5: Is a directory
grep: hw6: Is a directory
grep: hw7: Is a directory
grep: hw8: Is a directory
grep: hw9: Is a directory
sample.txt
services
grep: vscode-data: Is a directory
ubuntu@test2:~$ grep -lR tcp *
sample.txt
services
ubuntu@test2:~$
```

정규식 (regular expression)

문자	의미	예
?	한 글자 혹은 없음을 의미한다.	'ab?'는 ab 혹은 ab 다음에 한 글자가 오는 문자열
.	임의의 한 문자를 의미한다.	'a...b'는 a로 시작해서 b로 끝나는 5글자 문자열
*	바로 앞의 것을 0번 이상의 반복	'a*b'는 b, ab, aab, aaab, ... 등의 문자열
[ ]	[과] 사이의 문자 중 하나를 의미 – 기호: 문자의 범위를 지정	'[abc]d'는 ad, bd, cd를 뜻한다. [a-z]는 a부터 z까지 중 하나
[^...]	[^ 과] 사이의 문자를 제외한 나머지 문자 중 하나를 의미한다.	'[^abc]d'는 ad, bd, cd는 포함하지 않고 ed, fd 등은 포함. [^a-z]는 소문자가 아닌 모든 문자
^, $	각각 줄의 시작과 끝을 의미한다.	'^문자열'은 문자열로 시작하는 줄을 나타낸다. '문자열$'은 문자열로 끝나는 줄을 나타낸다.

정규식 사용 예

- $ grep “st..” you.txt

  Then, I am still and wait here in the silence

  You raise me up, so I can stand on mountains

  You raise me up, to walk on stormy seas

  I am strong, when I am on your shoulders

  Each restless heart beats so imperfectly;

- $ grep “st.*e” you.txt

  Then, I am still and wait here in the silence

  You raise me up, to walk on stormy seas

  I am strong, when I am on your shoulders

  Each restless heart beats so imperfectly;

- $ grep –w “st.*e” you.txt

  Then, I am still and wait here in the silence

파이프와 함께 grep 명령어 사용

- 파이프와 함께 grep 명령어 사용
  - 어떤 명령어를 실행하고 그 실행 결과 중에서 원하는 단어 혹은 문자열 패턴을 찾고자 할 때 사용함.

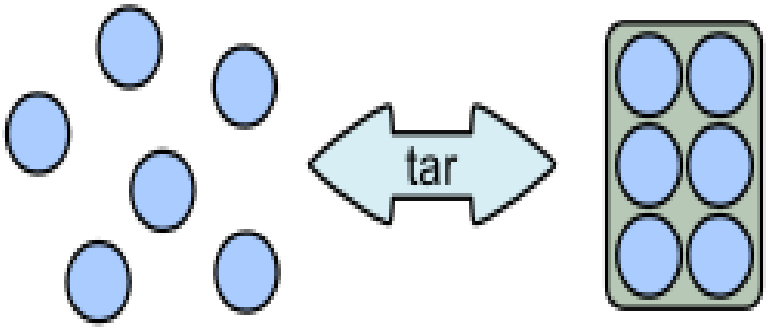
- 예

```
$ ls -l | grep hw
$ ps -ef | grep ubuntu
```

압축: tar and compressions

Tar Archive

- Archive
  - 백업 또는 다른 장소로의 이동을 위해 여러 파일들을 하나로 묶어놓은 묶음
  - 아카이브를 만들거나 푸는데 tar(tape archive) 사용
  - Tar: 이름에서 알 수 있듯, 전통적으로 Archive를 만드는데 사용.
    - Tar 로 생성된 archive 는 .tar 확장자를 가지며, tar file 혹은 tarball 이라고 표현하기도 함
- tar의 동작
  - 단순히 여러 파일의 데이터를 하나의 파일로 묶고,
  - 다시 여러 파일로 복원할 수 있도록 함
  - 파일의 용량은 원본보다 더 큼
    (복원을 위해 기존 파일들의 이름, 크기, 권한 등 필요한 정보를 포함하기 때문)
  - 기본적으로 압축 (Compression)을 수행하지 않음. Archive 와 compression 은 별개!



Tar Archive

- tar 명령어
  - 옵션: c(create), v(verbose), x(extract), t(table of contents), f(file)

  - $ tar -cvf 타르파일 파일+

    여러 파일들을 하나의 타르파일로 묶는다. 보통 확장자로 .tar 사용

  - $ tar -xvf 타르파일

    하나의 타르파일을 풀어서 원래 파일들을 복원한다.

  - $ tar -tvf 타르파일

    타르파일의 내용을 확인한다.

Tar Archive: 사용 예

- 현재 디렉터리에 있는 모든 파일을 다른 곳으로 옮기기

  $ tar -cvf src.tar *

  … src.tar를 다른 곳으로 이동한 후,

  $ tar -tvf src.tar

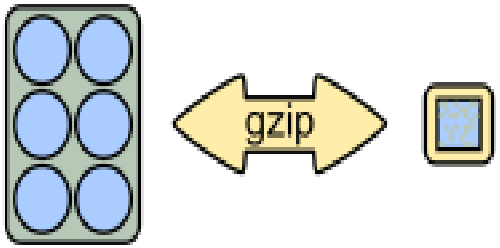
  (내용 확인하고: 생략 가능)

  $ tar -xvf src.tar

  (아카이브를 해제함)

파일 압축: gzip

- gzip 명령어
  - 리눅스 환경에서 많이 사용하는 압축 프로그램
  - 속도가 빠르고 적당한 수준의 압축률을 제공함



$ gzip [옵션] 파일*

파일(들)을 압축하여 .gz 파일을 만든다.

-d : 압축을 해제한다. (decompress)

-l : 압축파일 안에 있는 파일 정보(압축된 크기, 압축률) 출력한다.

-r : 하위 디렉터리까지 모두 압축한다.

-v : 압축하거나 풀 때 압축률, 파일명을 출력한다.

압축 풀기

- 사용법

$ gzip 파일*

gzip으로 파일(들)을 압축한다.

압축된 파일의 이름은 기존 이름 뒤에 .gz 확장자가 붙은 이름이다.

$ gzip –d 파일.gz*

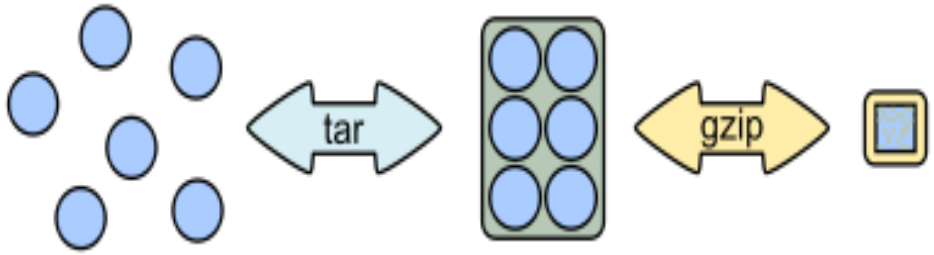
gzip으로 압축된 파일들을 복원한다.

$ gunzip 파일.gz*

gzip으로 압축된 파일들을 복원한다.

사용 예

- 일반적 사용 방법
  - 여러 파일들을 하나의 타르파일로 묶은 후 compress/gzip을 사용해 압축
  - 파일 복원: 압축을 해제하여 Tar 파일로 복원한 후,
    다시 Tar 파일을 풀어서 원래 파일들을 복원



$ tar -cvf src.tar *

$ gzip src.tar

… 이 파일을 원하는 곳으로 이동

$ gzip -d src.tar.gz

$ tar -xvf src.tar

파일 압축: zip, bzip2, xz

- Gzip 외에 리눅스 환경에서 자주 사용하는 압축 프로그램

명령어	확장자	압축률 순위	압축 속도 순위
gzip	.gz	3	1
bzip2	.bz2	2	2
xz	.xz	1 (아주 높음)	3 (아주 느림)

- 사용 방법은 모두 유사함
  - 압축 시는 옵션 없이 파일 지정
  - 해제 시는 -d 옵션

Tar file 생성 시, 압축을 동시에 수행하는 법

- Tar의 Compression options
  - Tar 파일을 생성하거나 해제할 시, 동시에 압축 또는 복원을 수행함
  - -z : gzip 이용
  - -j : bzip2 이용 (소문자 j)
  - -J : xz 이용 (대문자 J)
- 예) 압축

  $ tar -czvf textfiles.tar.gz *.txt

  $ tar -cJf textfiles.tar.xz *.txt
- 예) 복원

  $ tar -xzvf textfiles.tar.gz

  $ tar -xJf textfiles.tar.xz

기타

1. 파일 비교
2. 시스템 관리
3. 명령어 사용

파일 비교: cmp 명령어

- 사용법

```
$ cmp 파일1 파일2
```

파일1과 파일2가 같은지 비교한다.

- 출력
  - 두 파일이 같으면 아무 것도 출력하지 않음.
  - 두 파일이 서로 다르면 서로 달라지는 위치 출력

- 예
  - $ cmp you.txt me.txt
  - you.txt me.txt 다름: 340 자, 10 행

파일 비교: diff

- 사용법

```
$ diff [-i] 파일1 파일2
```

파일1과 파일2를 줄 단위로 비교하여 그 차이점을 출력한다.

-i 옵션은 대소문자를 무시하여 비교한다.

- 출력
  - 첫 번째 파일을 두 번째 파일 내용과 같도록 바꿀 수 있는 편집 명령어 형태

diff 출력 내용 설명

- 추가(a)

  첫 번째 파일의 줄 n1 이후에 두 번째 파일의 n3부터 n4까지의 줄들을 추가하면 두 파일은 서로 같다.

```
n1 a n3,n4
> 추가할 두 번째 파일의 줄들
```

- 예

```
$ diff you.txt me.txt
9a10,13
>
> You raise me up, so I can stand on mountains
> You raise me up, to walk on stormy seas
> I am strong, when I am on your shoulders
```

diff 출력 내용 설명

- 삭제(d)

  첫 번째 파일의 n1부터 n2까지의 줄들을 삭제하면 두 번째 파일의 줄 n3 이후와 서로 같다.

```
n1,n2 d n3
< 삭제할 첫 번째 파일의 줄들
```

- 예

```
$ diff me.txt you.txt
10,13d9
<
< You raise me up, so I can stand on mountains
< You raise me up, to walk on stormy seas
< I am strong, when I am on your shoulders
```

diff 출력 내용 설명

- 변경(c)

  첫 번째 파일의 n1부터 n2까지의 줄들을 두 번째 파일의 n3부터 n4까지의 줄들로 대치하면 두 파일은 서로 같다.

```
n1,n2 c n3,n4
< 첫 번째 파일의 대치될 줄들
--
> 두 번째 파일의 대치할 줄들
```

- 예

```
$ diff 파일1 파일2
1 c 1
< This is the first file
--
> This is the second file.
```

기타

1. 파일 비교
2. 시스템 관리
3. 명령어 사용

패키지 설치

- 리눅스에서 추가 프로그램을 설치하는 세 가지 방법
  - 소스 코드를 다운로드 받아 직접 컴파일하고 설치함
    - 장점: 필요한 경우, 직접 수정하여 사용 가능
    - 단점: 빌드 환경을 구성해야 하는 등, 과정이 복잡함
  - 패키지를 다운로드 받아 설치함
    - 장점: 컴파일 완료된 패키지 파일을 다운받아 설치함
    - 단점: 다른 패키지와의 의존성 문제를 직접 해결해야 함
      (예. A 설치를 위해, B가 필요한데, B는 설치되어 있지 않음)
  - 패키지 관리 프로그램 사용
    - 패키지 명을 지정하면, 자동으로 패키지를 검색해 다운받고, 설치함
    - 장점: 패키지 다운로드도 의존성 문제 등을 자동으로 해결해줌
    - 단점: Customizing , 버전 관리 등이 다소 불편함

패키지 관리 프로그램

- 여러 배포본에 따라 서로 다른 패키지 관리 프로그램을 제공
  - Debian 계열 (Ubuntu, Debian, : apt (or apt-get)
    - 패키지 형태: dpkg
  - Redhat 계열 (CentOS, Fedora,RHEL): yum
    - 패키지 형태: rpm
- apt
  - 기존 apt-get 보다 사용자 편의적인 high-level package management system
  - 사용법: # apt [command] [package name]
  - Commands
    - install: 패키지 설치
    - autoremove: 패키지 자동 제거
    - update: 패키지 정보를 최신으로 업데이트함.
    - upgrade: 최신 버전으로 업데이트 가능한 모든 패키지를 설치함
    - search: 패키지 정보 검색
    - list: 사용 가능한 패키지 리스트 출력. 이미 설치된 경우, [installed] 로 표시됨.

apt 사용 예

```
root@41983:/home/ubuntu# tree

Command 'tree' not found, but can be installed with:

snap install tree  # version 1.8.0+pkg-3fd6, or
apt  install tree  # version 1.8.0-1

See 'snap info tree' for additional versions.

root@41983:/home/ubuntu# apt install tree
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  tree
0 upgraded, 1 newly installed, 0 to remove and 49 not upgraded.
Need to get 43.0 kB of archives.
After this operation, 115 kB of additional disk space will be used.
Get:1 http://nova.clouds.archive.ubuntu.com/ubuntu focal/universe amd64 tree amd64 1.8.0-1 [43.0 kB]
Fetched 43.0 kB in 2s (27.6 kB/s)
Selecting previously unselected package tree.
(Reading database ... 99891 files and directories currently installed.)
Preparing to unpack .../tree_1.8.0-1_amd64.deb ...
Unpacking tree (1.8.0-1) ...
Setting up tree (1.8.0-1) ...
Processing triggers for man-db (2.9.1-1) ...
```

설치가 정상 완료된 경우, 특별한 메시지 없이 종료됨

```
root@41983:/home/ubuntu# tree
.
├── hw1
│   └── hw1.c
├── hw10
│   └── hw10.c
├── hw11
│   └── hw11.c
├── hw12
│   └── hw12.c
├── hw2
│   └── hw2.c
├── hw3
│   └── hw3.c
├── hw4
│   └── hw4.c
├── hw5
│   └── hw5.c
├── hw6
│   └── hw6.c
├── hw7
│   └── hw7.c
├── hw8
│   └── hw8.c
├── hw9
│   └── hw9.c
├── test
└── vscode-data
    ├── extensions
    ├── heartbeat
    ├── logs
    │   ├── 20200914T080741
    │   │   └── remoteagent.log
    │   └── 20200914T080743
    │       └── remoteagent.log
    └── machineid

17 directories, 17 files
root@41983:/home/ubuntu#
```

디스크 사용: df

- df 명령어
  - 파일시스템에 대한 정보를 보여준다.

$ df 파일시스템*

파일 시스템의 사용중이거나 사용 가능한 디스크 공간에 대한 정보를 보여준다.

- 사용 예

```
$ df
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/mapper/root      51606140   7570736  41413964  16% /
tmpfs                  1030972       676   1030296   1% /dev/shm
/dev/sda1               495844     29048    441196   7% /boot
/dev/mapper/home     424544656   3577668 399401344   1% /home
```

디스크 사용: du

- du 명령어

$ du [-s] 파일*

파일이나 디렉토리가 사용하는 디스크 사용량(블록 수)을 알려준다.

  - 파일을 명시하지 않으면 현재 디렉터리의 사용 공간을 보여준다.
  - 자주 사용하는 형태 -ahd 1
    - 모든 파일에 대해 (-a), 단위를 포함하여 (-h), 한 레벨 아래의 하위 디렉토리까지 (-d 1) 출력

디스크 사용: du

```
ubuntu@test2:~$ du
8       ./hw12
8       ./hw1
4       ./.config/procps
8       ./.config
8       ./hw10
8       ./.ssh
8       ./hw3
4       ./vscode-data/extensions
16      ./vscode-data/logs/20200912T022900
8       ./vscode-data/logs/20200912T022857
28      ./vscode-data/logs/20200915T045539
56      ./vscode-data/logs
68      ./vscode-data
8       ./hw4
4       ./.cache
8       ./hw8
8       ./hw9
8       ./hw7
8       ./hw11
8       ./hw6
8       ./hw2
8       ./hw5
260     .
ubuntu@test2:~$ du -s
260     .
ubuntu@test2:~$
```

```
ubuntu@test2:~$ du -ahd 1
4.0K    ./.profile
8.0K    ./sample.txt.gz
8.0K    ./hw12
8.0K    ./hw1
4.0K    ./.Xauthority
4.0K    ./.bash_history
8.0K    ./.config
8.0K    ./hw10
8.0K    ./.ssh
8.0K    ./hw3
68K     ./vscode-data
8.0K    ./hw4
16K     ./services
4.0K    ./.bash_logout
4.0K    ./.cache
12K     ./.viminfo
4.0K    ./temp
8.0K    ./aladin.txt.gz
8.0K    ./hw8
4.0K    ./.bashrc
4.0K    ./.lesshst
8.0K    ./hw9
8.0K    ./hw7
8.0K    ./hw11
8.0K    ./hw6
8.0K    ./hw2
8.0K    ./hw5
0       ./.sudo_as_admin_successful
260K    .
ubuntu@test2:~$
```

```
root@41983:/home/ubuntu# tree -dsh
.
├── [4.0K]  hw1
├── [4.0K]  hw10
├── [4.0K]  hw11
├── [4.0K]  hw12
├── [4.0K]  hw2
├── [4.0K]  hw3
├── [4.0K]  hw4
├── [4.0K]  hw5
├── [4.0K]  hw6
├── [4.0K]  hw7
├── [4.0K]  hw8
├── [4.0K]  hw9
└── [4.0K]  vscode-data
    ├── [4.0K]  extensions
    └── [4.0K]  logs
        ├── [4.0K]  20200914T080741
        └── [4.0K]  20200914T080743

17 directories
root@41983:/home/ubuntu#
```

* tree 로도 유사한 동작 가능

IP 주소

- 사용법

```
$ ip addr
```

사용중인 시스템의 IP 주소를 출력한다.

- 예

```
ubuntu@test2:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1450 qdisc fq_codel state UP group default qlen 1000
    link/ether fa:16:3e:02:e2:31 brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.55/24 brd 10.0.0.255 scope global dynamic ens3
       valid_lft 55141sec preferred_lft 55141sec
    inet6 fe80::f816:3eff:fe02:e231/64 scope link
       valid_lft forever preferred_lft forever
ubuntu@test2:~$
```

IP 주소

- $ ifconfig
  - Network Interface 에 대한 configuration 을 수행함
  - 단순히 ip addr 과 같이 네트워크 정보를 확인하기 위해서도 많이 사용함

```
root@41983:/home/ubuntu# ifconfig
ens3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1450
        inet 10.0.0.249  netmask 255.255.255.0  broadcast 10.0.0.255
        inet6 fe80::f816:3eff:feb2:6ece  prefixlen 64  scopeid 0x20<link>
        ether fa:16:3e:b2:6e:ce  txqueuelen 1000  (Ethernet)
        RX packets 30287  bytes 46600720 (46.6 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 25219  bytes 3094333 (3.0 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 1190  bytes 99962 (99.9 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1190  bytes 99962 (99.9 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@41983:/home/ubuntu#
```

기타

1. 파일 비교
2. 시스템 관리
3. 명령어 사용

별명

- alias 명령어
  - 문자열이 나타내는 기존 명령에 대해 새로운 이름을 별명으로 정의

  $ alias 이름=문자열

  $ alias dir='ls –aF'

  $ dir

  $ alias h=history

  $ alias ll='ls –l'

```
ubuntu@41983:~$ alias
alias alert='notify-send --urgency=low -i "$([ $? = 0 ] && echo terminal || echo error)" "
alias egrep='egrep --color=auto'
alias fgrep='fgrep --color=auto'
alias grep='grep --color=auto'
alias l='ls -CF'
alias la='ls -A'
alias ll='ls -alF'
alias ls='ls --color=auto'
ubuntu@41983:~$
```

- 현재까지 정의된 별명들을 확인

  $ alias # 별명 리스트

  alias dir='ls –aF'

  alias h=history

  alias ll='ls –l'

- 이미 정의된 별명 해제

  $ unalias 단어

히스토리

- 입력된 명령들을 기억하는 기능

  $ history [-rh] [번호]

- 기억할 히스토리의 크기

  $ HISTSIZE=100

- 로그아웃 후에도 히스토리가 저장되도록 설정

  $ HISTFIESIZE=100

```
$ history
1 ls
2 who
3 env
4 vi test.sh
5 chmod +x test.sh
6 test.sh
7 ls
8 date
9 history
...
```

재실행

형태	의미
!!	바로 전 명령 재실행
!n	이벤트 번호가 n인 명령 재실행
! 시작스트링	시작스트링으로 시작하는 최후 명령 재실행
!? 서브스트링	서브스트링을 포함하는 최후 명령 재실행

- 예

```
$ !!          # 바로 전 명령 재실행
$ !20         # 20번 이벤트 재실행
$ !gcc        # gcc로 시작하는 최근 명령 재실행
$ !?test.c          # test.c를 포함하는 최근 명령 재실행
```