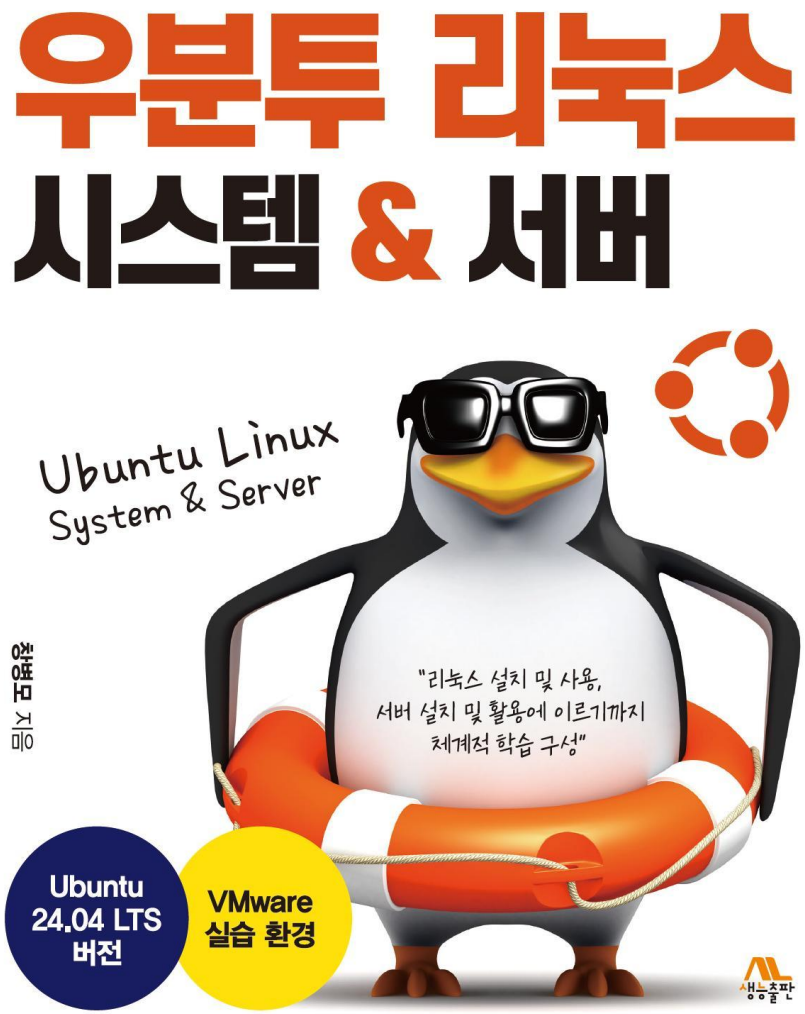
LINUX

Autumn 2025

Daejeon Univ.
Seongbok Baik
sbbaik@dju.ac.kr

Text Book

교재명	우분투 리눅스 시스템 & 서버
저자	창병모
출판사	생능출판사
발행년	2024.07.12
ISBN	979-11-92932-72-9



7.1 파일 시스템

리눅스 파일 시스템 핵심 개념

(1) 하나의 파일은 하나의 i-노드로 관리된다.

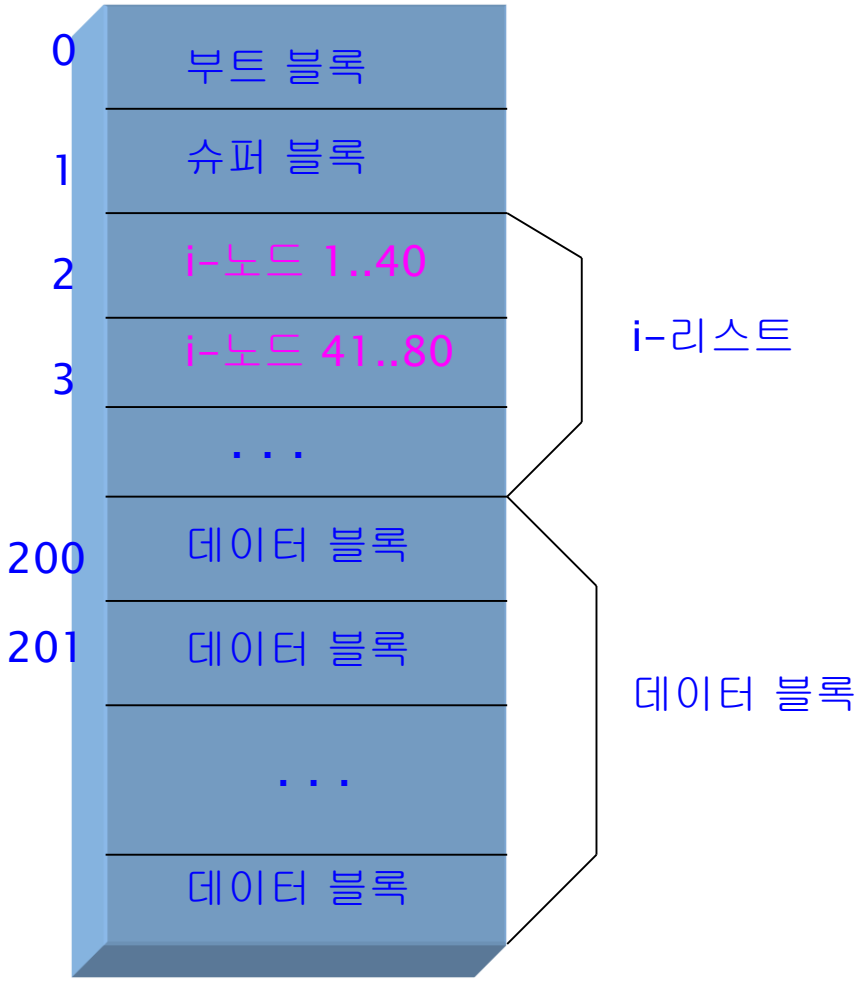
(2) 디렉터리는 단순히 파일명의 목록을 가지고 있는 일종의 파일이다.

(3) 특수 파일을 통해 장치에 접근할 수 있다.

파일 시스템 구조

- 부트 블록(Boot block)
  - 파일 시스템 시작부에 위치하고 보통 첫 번째 섹터를 차지
  - 부트스트랩 코드가 저장되는 블록
- 슈퍼 블록(Super block)
  - 전체 파일 시스템에 대한 정보를 저장
    - 총 블록 수, 사용 가능한 i-노드 개수, 사용 가능한 블록 비트 맵, 블록의 크기, 사용 중인 블록 수, 사용 가능한 블록 수 등
- i-리스트(i-list)
  - 각 파일을 나타내는 모든 i-노드들의 리스트
  - 한 블록은 약 40개 정도의 i-노드를 포함
- 데이터 블록(Data block)
  - 파일의 내용(데이터)을 저장하기 위한 블록들

파일 시스템 구조


0
부트 블록
1
슈퍼 블록
2
i-노드 1..40
3
i-노드 41..80
. . .
200
데이터 블록
201
데이터 블록
. . .
데이터 블록
i-리스트
데이터 블록

i-노드

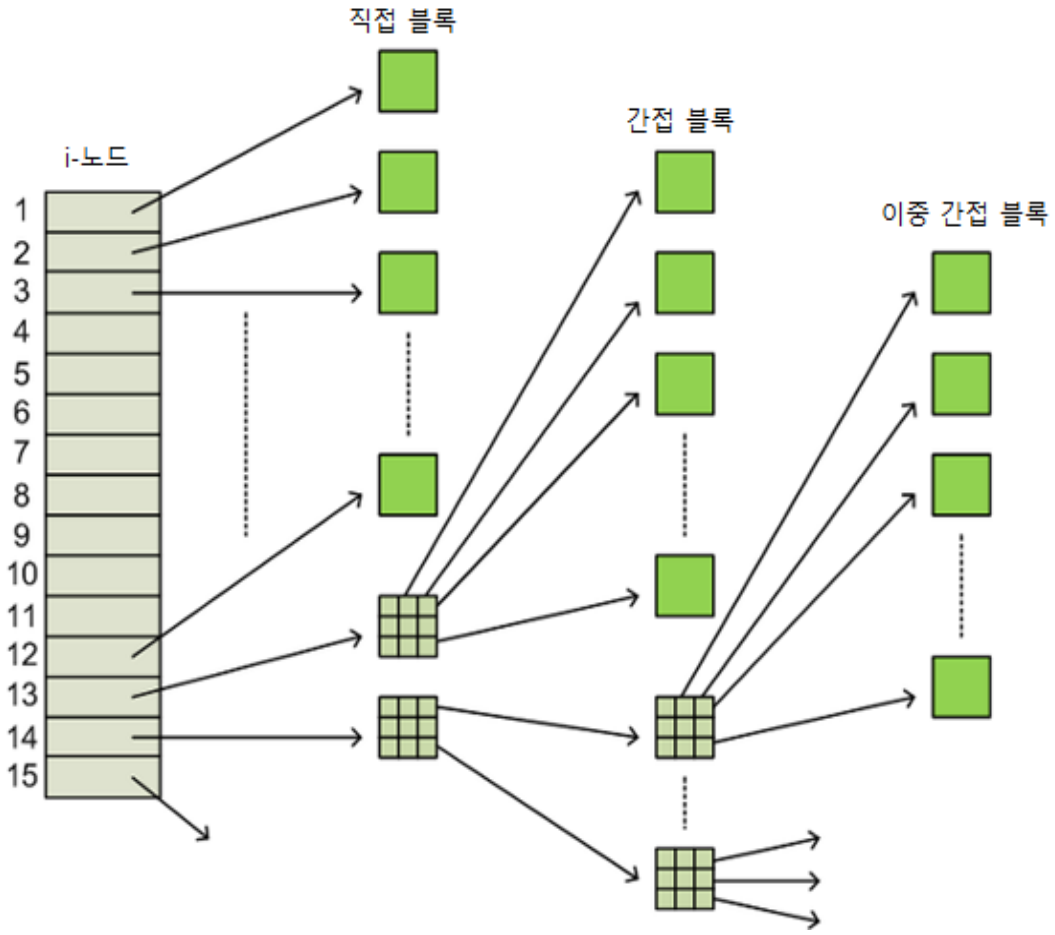
- 한 파일은 하나의 i-노드를 갖는다.

```
$ ls -i cs1.txt
1196554 cs1.txt
```

- 파일에 대한 모든 정보를 가지고 있음

파일 상태 정보	의미
블록 수	파일을 구성하는 블록의 개수(K 바이트 단위)
파일 종류	파일 종류를 나타낸다.
접근권한	파일에 대한 소유자, 그룹, 기타 사용자의 읽기/쓰기/실행 권한
하드 링크 수	파일에 대한 하드 링크 개수
소유자 및 그룹	파일의 소유자 ID 및 소유자가 속한 그룹
파일 크기	파일의 크기(바이트 단위)
최종 접근 시간	파일을 최후로 접근한 시간
최종 수정 시간	파일을 생성 혹은 최후로 수정한 시간
데이터 블록 주소	실제 데이터가 저장된 데이터 블록의 주소

i-노드와 블록 포인터



블록 포인터

- 데이터 블록에 대한 포인터
  - 파일의 내용을 저장하기 위해 할당된 데이터 블록의 주소

- 하나의 i-노드 내의 블록 포인터
  - 직접 블록 포인터 12개
  - 간접 블록 포인터 1개
  - 이중 간접 블록 포인터 1개

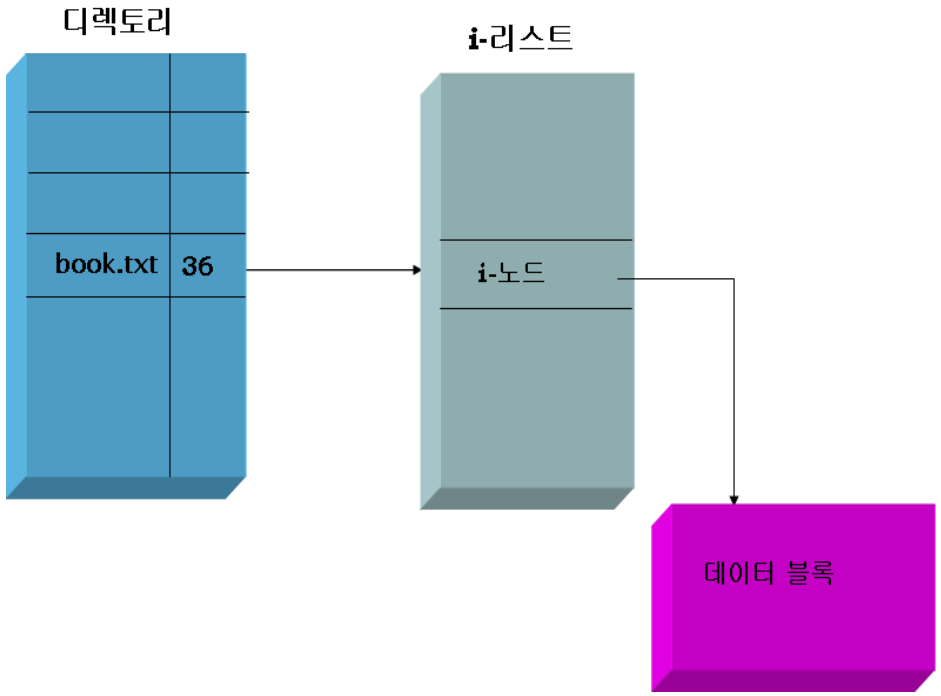
- 최대 몇 개의 데이터 블록을 가리킬 수 있을까?
  - 한 블록 크기: 4096바이트
  - 블록 포인터: 8바이트

7.2 디렉터리

디렉터리 구현

- 디렉터리 내에는 무엇이 저장되어 있을까?
- 디렉터리 엔트리

```
#include <dirent.h>
struct dirent {
    ino_t d_ino; // i-노드 번호
    char d_name[NAME_MAX + 1];
    // 파일 이름
}
```

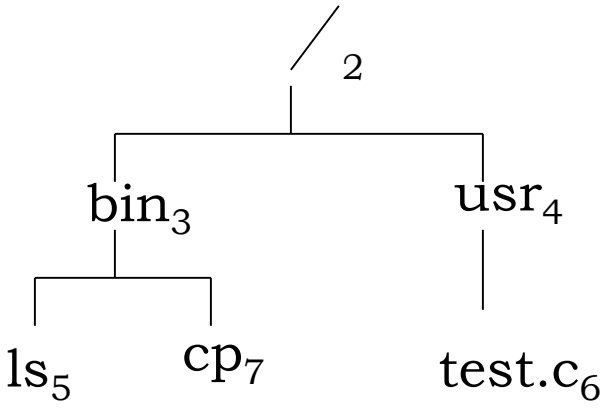


디렉터리 구현

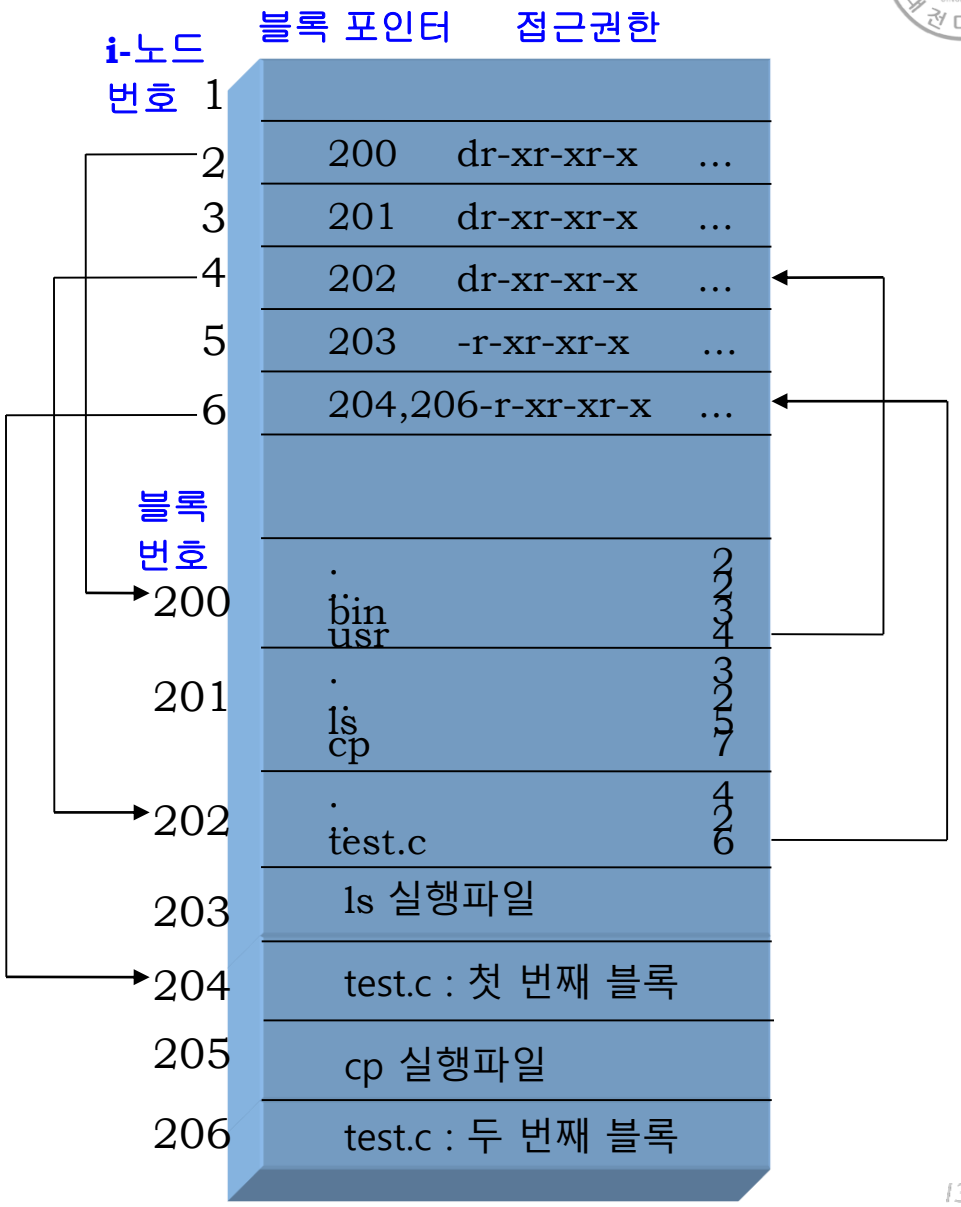
- 그림의 파일 시스템 구조를 보자.
  - 디렉터리를 위한 별도의 구조는 없다.

- 파일 시스템 내에서 디렉터리를 어떻게 구현할 수 있을까?
  - 디렉터리도 일종의 파일로 다른 파일처럼 구현된다.
  - 디렉터리도 다른 파일처럼 하나의 i-노드로 표현된다.
  - 디렉터리의 내용은 디렉터리 엔트리(파일이름, i-노드 번호)

디렉토리 계층구조 구현



/usr/test.c



7.3 파일 시스템 관련 유틸리티

파일 시스템 보기

- 사용법

```
$ df 파일시스템*

파일 시스템에 대한 디스크 사용 정보를 보여준다.
```

- 사용 예

```
$ df
파일 시스템     1K-블록          사용          가용     사용%     마운트위치
tmpfs          401048      1804     399244       1%    /run
/dev/sda2  51287520 10831320   37818532      23%    /
tmpfs        2005232         0    2005232       0%    /dev/shm

$ df /
파일 시스템     1K-블록          사용          가용     사용%     마운트위치
/dev/sda2  51287520 10831320   37818532      23%    /
```

  - / 루트 파일 시스템 현재 23% 사용
  - /run 임시 파일을 저장하기 위한 파일 시스템
  - /dev/shm 프로세사 사이의 통신을 위한 공유 메모리

파일 시스템 마운트

- 마운트 설정
  - mount 명령어를 사용하여 할 수 있으며
  - 마운트 정보는/etc/fstab 파일에 저장된다.

- fstab 구조 예

```
<file system> <mount point> <type> <options> <dump> <pass>
/dev/disk/by-uuid/d2f11711-... / ext4 defaults 0 1
```

  - <file system>: 마운트할 장치나 파일 시스템의 경로
  - <mount point>: 파일 시스템을 마운트할 디렉토리 경로
  - <type>: 파일 시스템의 유형으로, 예를 들어 ext4, ntfs 등
  - <options>: 마운트 옵션(읽기/쓰기, 자동 마운트 등을 설정)
  - <dump>: 파일 시스템의 백업 여부를 지정. 보통 0으로 설정
  - <pass>: 부팅할 때 fsck(파일 시스템 체크) 실행 순서 지정

mount 명령어

- 사용법

```
# mount [옵션] 장치명 디렉터리
장치를 지정한 디렉터리에 마운트한다.

# umount 장치명 또는 디렉터리
마운트된 장치 또는 디렉터리를 마운트 해제한다.
```

- -o 옵션
  - -o ro: 읽기 전용으로 마운트
  - -o rw: 읽기/쓰기로 마운트
  - -o remount: 마운트를 다시 하며, 읽기 전용에서 읽기/쓰기 모드로 또는 그 반대로 변경할 때 사용됨
- 예 /dev/sr0(cdrom)
  - # mount /dev/sr0 /mnt
  - # ls /mnt
  - # umount /mnt

디스크 사용량 보기

- 사용법

```
$ du [-s] 파일명*
```

파일 혹은 디렉터리의 사용량을 보여준다. 파일을 명시하지 않으면 현재 디렉터리 내의 모든 파일들의 사용 공간을 보여준다.

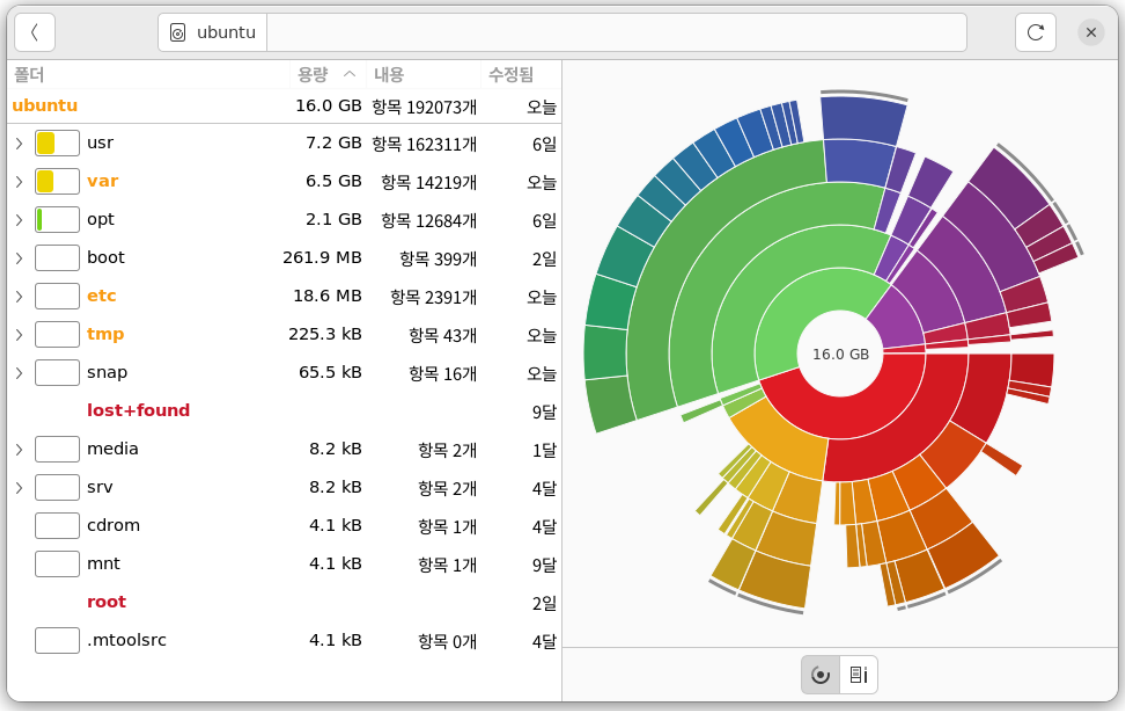
- 예

```
$ du
208 ./사진
4     ./.local/share/nautilus/scripts
8     ./.local/share/nautilus
144 ./.local/share/gvfs-metadata
4     ./.local/share/icc
...
```

```
$ du -s
22164 .
```

디스크 사용량 분석기(Disk Usage Analyzer)

- 시각적 분석
  - 디스크의 사용 현황을 트리맵(tree map) 또는 막대그래프로 표시
  - 사용 중인 공간을 시각적으로 보여준다.
- 디렉터리 및 파일 크기 확인
- 디스크 공간 최적화



7.4 파일 속성으로 파일 찾기

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.

find 명령어

- 파일명을 명시하는 -name 옵션

> $ find 디렉터리 –name 파일명 -print 혹은 -ls
>
> 지정된 디렉터리 아래에서 파일명에 해당되는 파일들을 모두 찾아 그 경로를 출력한다.

- 예

```
$ find ~ -name src -print
/home/chang/linux/src

$ find ~ -name src -ls
89090 4 drwxrwxr-x 13 chang cs 4096 9월22 /home/chang/linux/src

$ find /usr -name *.c -print
```

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 소유자(-user)로 검색

```
$ find . -user chang -print
```

- 파일 크기(-size)로 검색

```
$ find . -size +1024 -print
```

- 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

- 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 -print
$ find . -mtime -7 -print
```

find 명령어: 검색 조건 조합

- find 명령어는 여러 검색 옵션을 조합해서 사용할 수 있다.

- 예

```
$ find . -type d -perm 700 -print
$ find . -name core -size +2048 -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 {} \;
$ find . -name *.c -atime +30 -exec ls -l {} \;
```

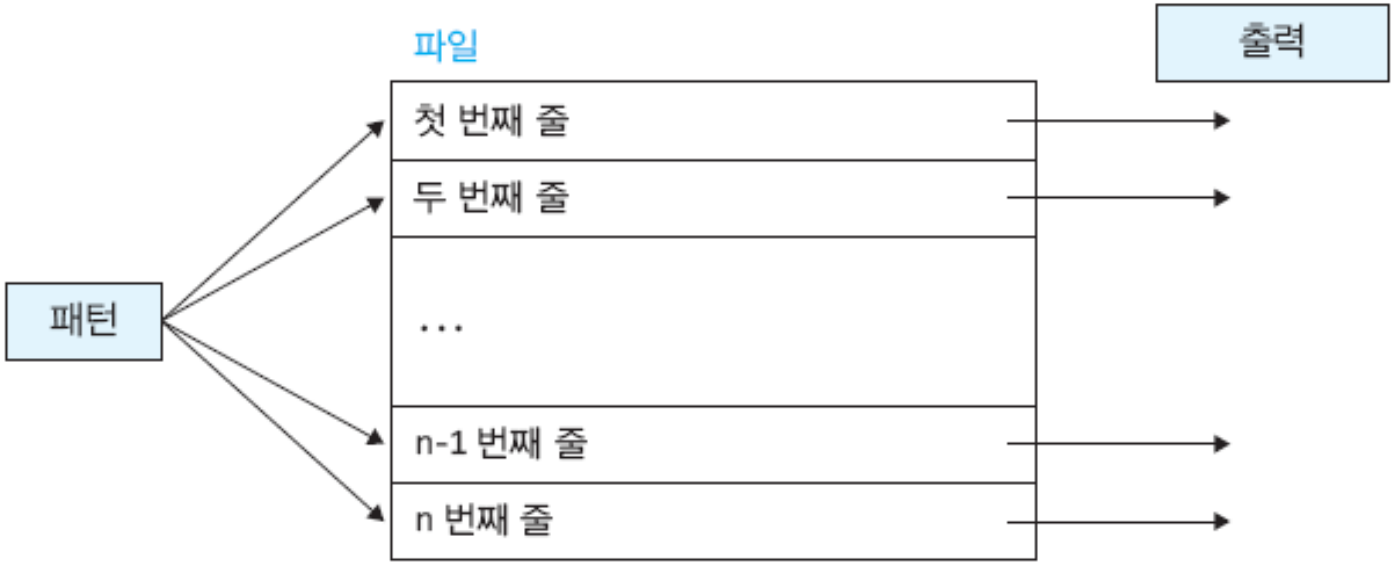
7.5 파일 필터링

grep 명령어

- 사용법

```
$ grep 패턴 파일*
```

파일(들)을 대상으로 지정된 패턴의 문자열을 검색하고, 해당 문자열을 포함하는 줄들을 출력한다.



grep 명령어

- $ grep with you.txt
  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
  Until you come and sit awhile with me
  But when you come and I am filled with wonder,

- $ grep -n with you.txt
  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 명령어의 옵션

옵션	기능
-i	대소문자를 무시하고 검색한다.
-l	해당 패턴이 들어있는 파일명을 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.

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

정규식

문자	의미	예
.	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 chang
$ ps -ef | grep chang
```

7.6 파일 정렬

정렬: sort 명령어

- 사용법

```
$ sort [-옵션] 파일*
```

텍스트 파일(들)의 내용을 줄 단위로 정렬한다. 옵션에 따라 다양한 형태로 정렬한다.

- 정렬 방법
  - 정렬 필드를 기준으로 줄 단위로 오름차순으로 정렬한다.
  - 기본적으로는 각 줄의 첫 번째 필드가 정렬 필드로 사용된다.
  - -r 옵션을 사용하여 내림차순으로 정렬할 수 있다.

sort 명령어 예

```
$ sort you.txt
But when you come and I am filled with wonder,
Each restless heart beats so imperfectly;
I am strong, when I am on your shoulders
Sometimes, I think I glimpse eternity
Then, I am still and wait here in the silence
There is no life - no life without its hunger;
Until you come and sit awhile with me
When I am down and, oh my soul, so weary
When troubles come and my heart burdened be
You raise me up, so I can stand on mountains
You raise me up, to more than I can be
You raise me up, to walk on stormy seas
```

sort 명령어 예

```
$ sort -r you.txt
You raise me up, to walk on stormy seas
You raise me up, to more than I can be
You raise me up, so I can stand on mountains
When troubles come and my heart burdened be
When I am down and, oh my soul, so weary
Until you come and sit awhile with me
There is no life - no life without its hunger;
Then, I am still and wait here in the silence
Sometimes, I think I glimpse eternity
I am strong, when I am on your shoulders
Each restless heart beats so imperfectly;
But when you come and I am filled with wonder,
```

정렬 필드 지정

필드 지정	기능
-k 필드번호	필드번호에 해당하는 필드를 기준으로 정렬한다. 이 옵션에서 필드번호는 1부터 시작된다.
+시작필드 -종료필드	시작필드부터 종료필드-1까지의 필드들을 기준으로 정렬한다. 이 때 필드 번호는 0부터 시작된다.

정렬 필드 지정 예

```
$ sort -k 3 you.txt 혹은 sort +2 -3 you.txt
Then, I am still and wait here in the silence
When I am down and, oh my soul, so weary
Until you come and sit awhile with me
When troubles come and my heart burdened be
Each restless heart beats so imperfectly;
You raise me up, so I can stand on mountains
You raise me up, to more than I can be
You raise me up, to walk on stormy seas
There is no life - no life without its hunger;
I am strong, when I am on your shoulders
Sometimes, I think I glimpse eternity
But when you come and I am filled with wonder,
```

sort 명령어의 옵션

옵션	기능
-b	앞에 붙는 공백은 무시한다.
-c	정렬이 되지 않은 상태로 출력한다.
-d	숫자, 문자, 공백만 비교하여 사전식 순서로 정렬한다.
-f	대소문자를 구분하지 않고 정렬한다.
-n	숫자 문자열의 숫자 값에 따라 비교하여 정렬한다.
-r	역순(내림차순)으로 정렬한다.
-t 문자	지정한 문자를 필드 구분자로 사용한다.

sort 명령어의 옵션 예

- -o 출력파일 옵션
  - 정렬된 내용을 지정된 파일에 저장할 수 있다.

```
$ sort -o sort.txt you.txt
```

- -n 옵션
  - 숫자 문자열의 경우에 숫자가 나타내는 값의 크기에 따라 비교하여 정렬할 수 있다.
  - 예: “49”와 “100”

필드 구분 문자 지정

```
$ sort -t: -k 3 -n /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
...
```

7.7 파일 비교

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: 편집 명령어

- 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: 편집 명령어

- 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: 편집 명령어

- 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.
```

핵심 개념

- 리눅스 파일 시스템은 부트 블록, 슈퍼 블록, i-노드 리스트, 데이터 블록 등으로 구성된다.
- 파일 하나당 하나의 i-노드가 있으며 i-노드 내에 파일에 대한 모든 상태 정보가 저장되어 있다.
- 디렉터리는 일련의 디렉터리 엔트리들을 포함하고 각 디렉터리 엔트리는 파일 이름과 그 파일의 i-노드 번호로 구성된다.
- find 명령어는 파일명이나 속성을 이용하여 해당하는 파일을 찾는데 사용된다.
- grep 명령어는 파일들을 대상으로 지정된 패턴의 문자열을 검색하고, 해당 문자열을 포함하는 줄들을 출력한다
- sort 명령어는 텍스트 파일을 줄 단위로 정렬한다.
- cmp 명령어는 두 파일이 같은지 비교한다.
- diff 명령어는 두 파일이 서로 다른지 비교한다.