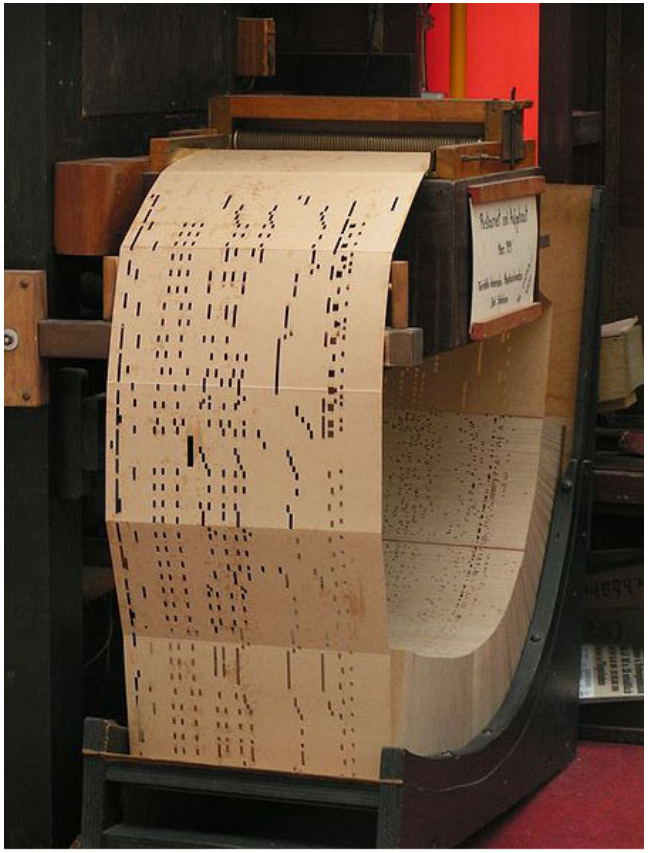
오픈소스소프트웨어

Open-Source Software

ICT융합학부 조용우

File?

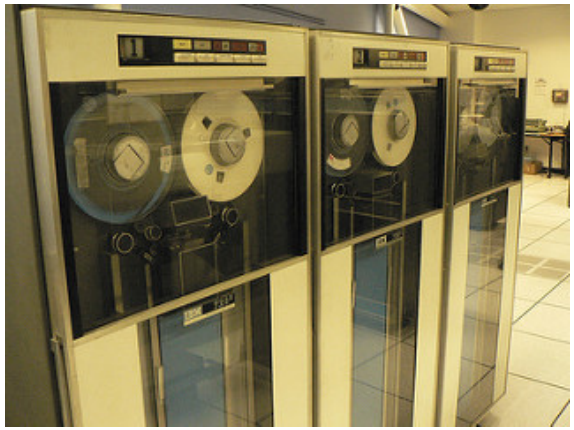
File?

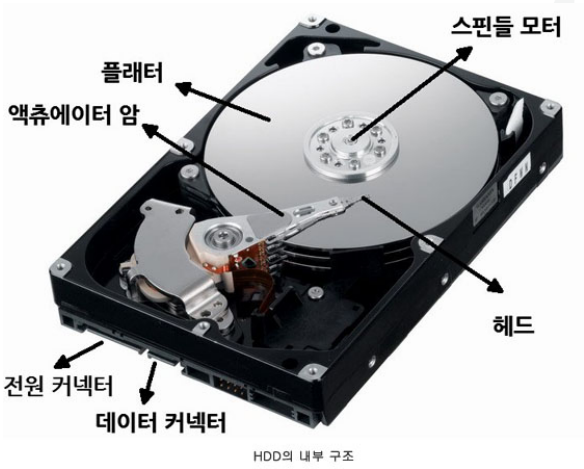


초기 컴퓨터에서 사용하던 천공카드는 기록 용량이 작고 보관도 불편했다
사진 출처: http://commons.wikimedia.org/wiki/File:Lochkarte_Tanzorgel.jpg

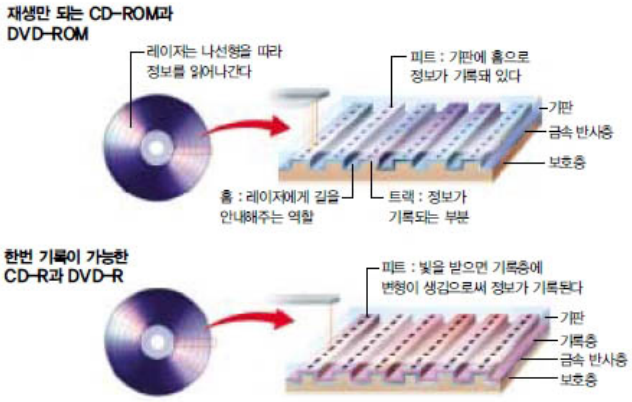
IBM 305 RAMAC에 적용된 세계 최초의 하드디스크 드라이브(1956년) – 사진 출처: IBM 홈페이지 http://www-03.ibm.com/ibm/history/exhibits/storage/storage_350.html

File?

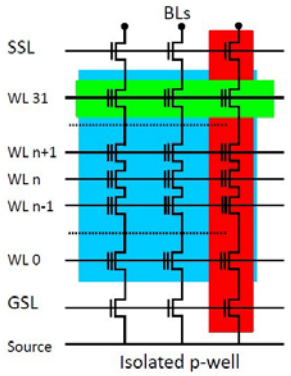




HDD의 내부 구조

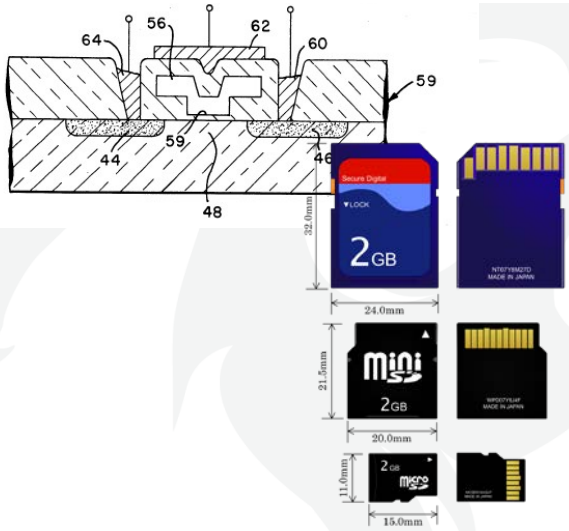


NAND array organization

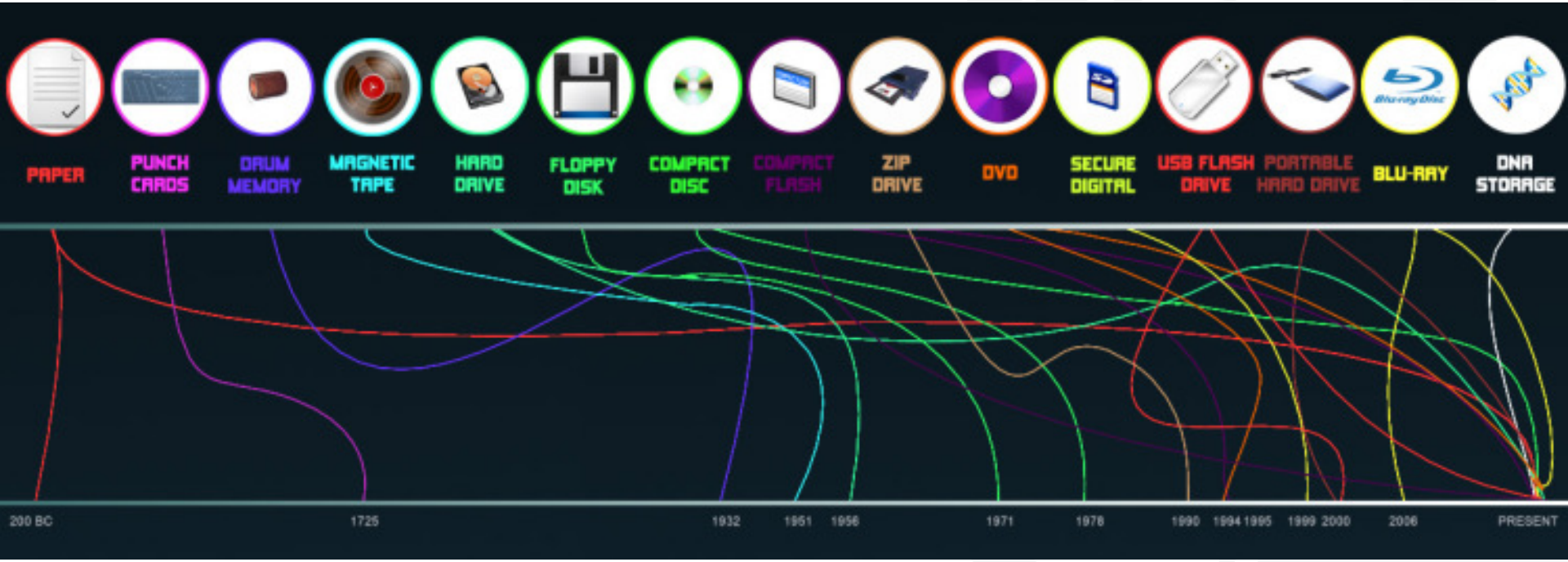


- Page (row)
  - Minimum unit for program
  - 16/32 k cells on the same WL
- String (column)
  - Minimum unit for read
  - Serial connection of 32/64 cells & 2 select transistors
- Block
  - Minimum unit for erase
  - 2-dimensional array of WLs within string
  - 64/128 pages per block

9

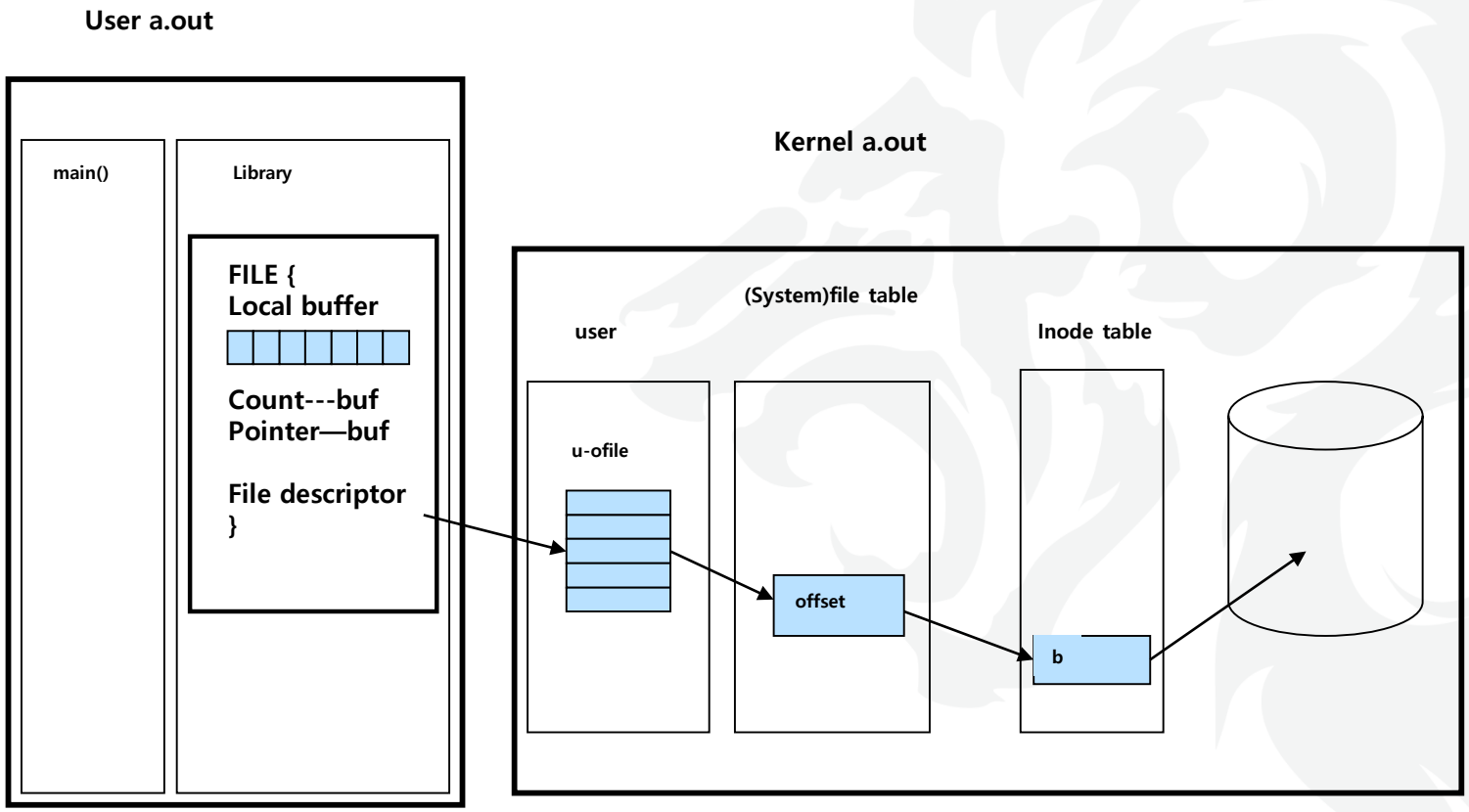


저장장치의 역사

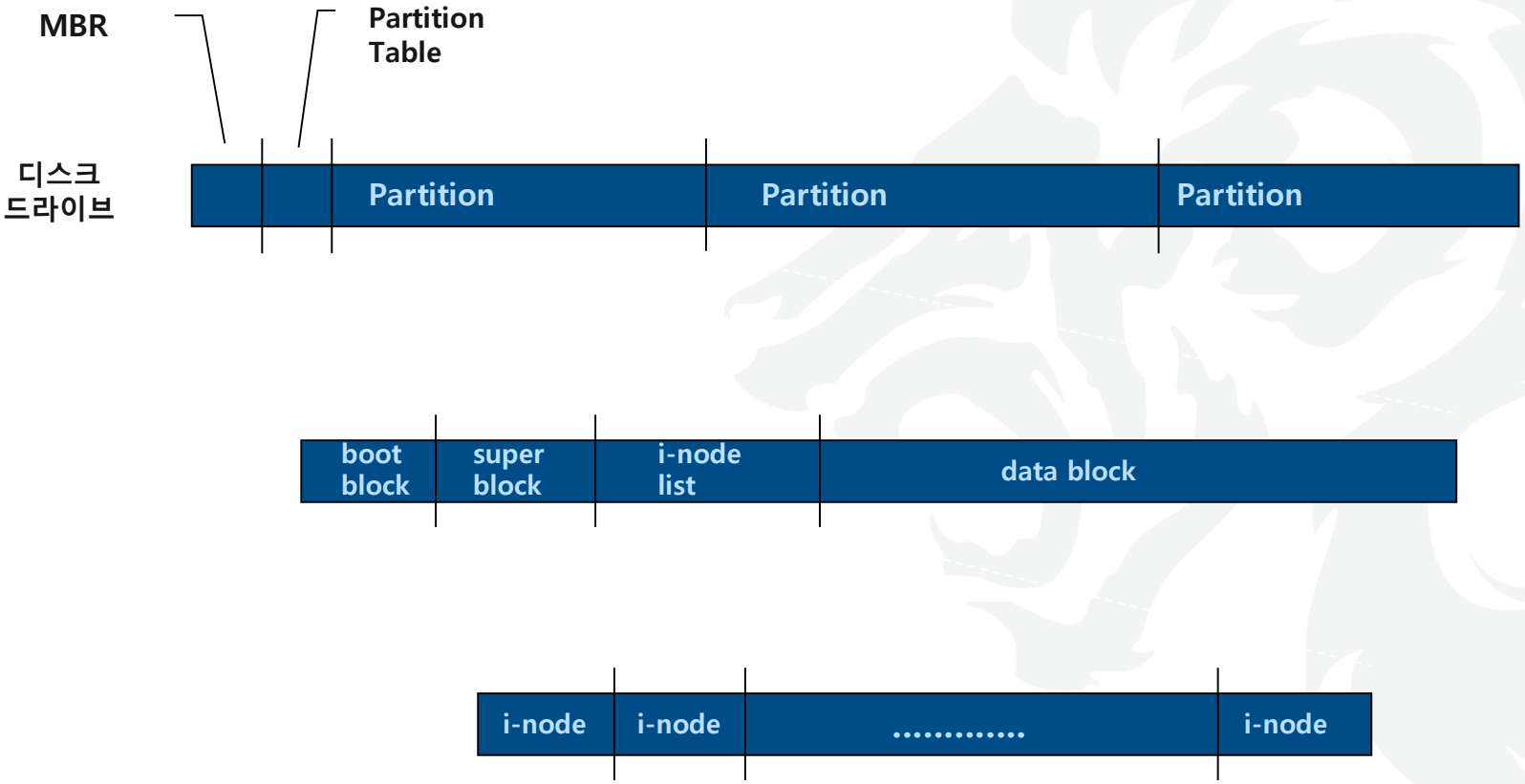

PAPER
PUNCH CARDS
DRUM MEMORY
MAGNETIC TAPE
HARD DRIVE
FLOPPY DISK
COMPACT DISC
COMPACT FLASH
ZIP DRIVE
DVD
SECURE DIGITAL
USB FLASH DRIVE
PORTABLE HARD DRIVE
BLU-RAY
DNA STORAGE
200 BC
1725
1932
1951
1956
1971
1978
1990
1994
1995
1999
2000
2006
PRESENT

File System

파일 접근

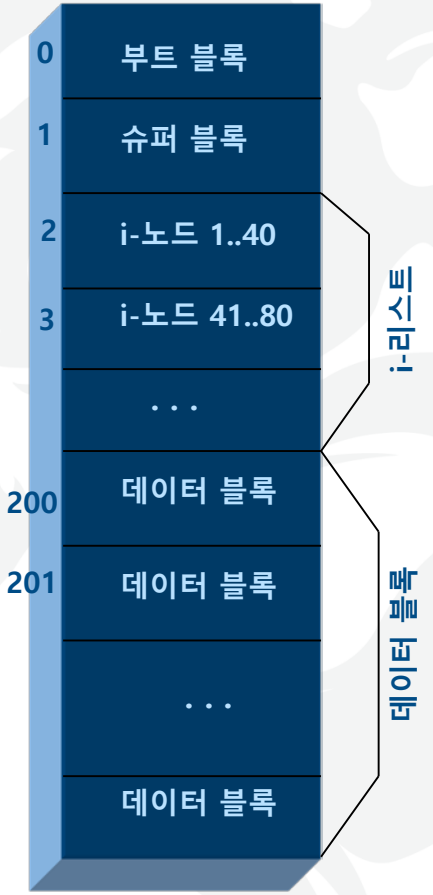

User a.out
main()
Library
FILE {
Local buffer
Count---buf
Pointer—buf
File descriptor
}
Kernel a.out
(System)file table
user
Inode table
u-ofile
offset
b

디스크 파일 시스템


MBR
Partition Table
디스크 드라이브
Partition
Partition
Partition
boot block
super block
i-node list
data block
i-node
i-node
.....
i-node

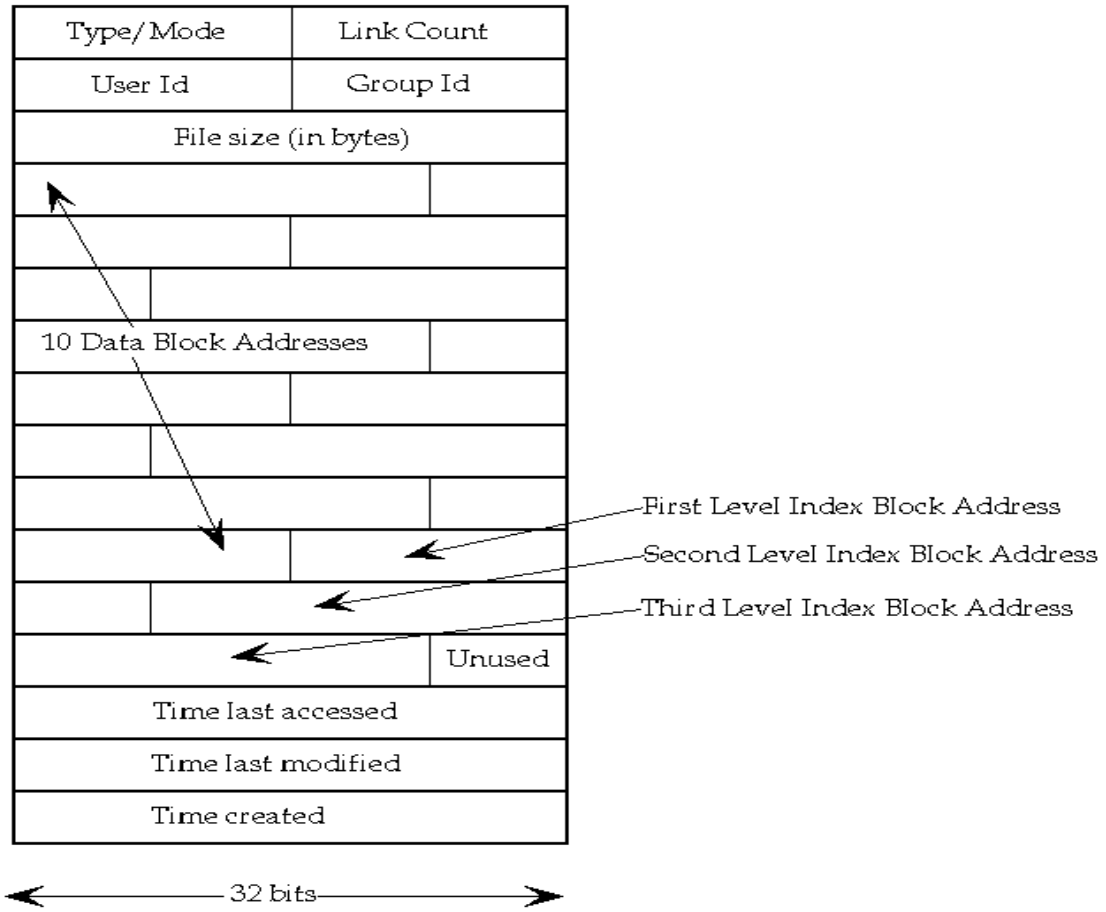
파일 시스템 구조

- 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

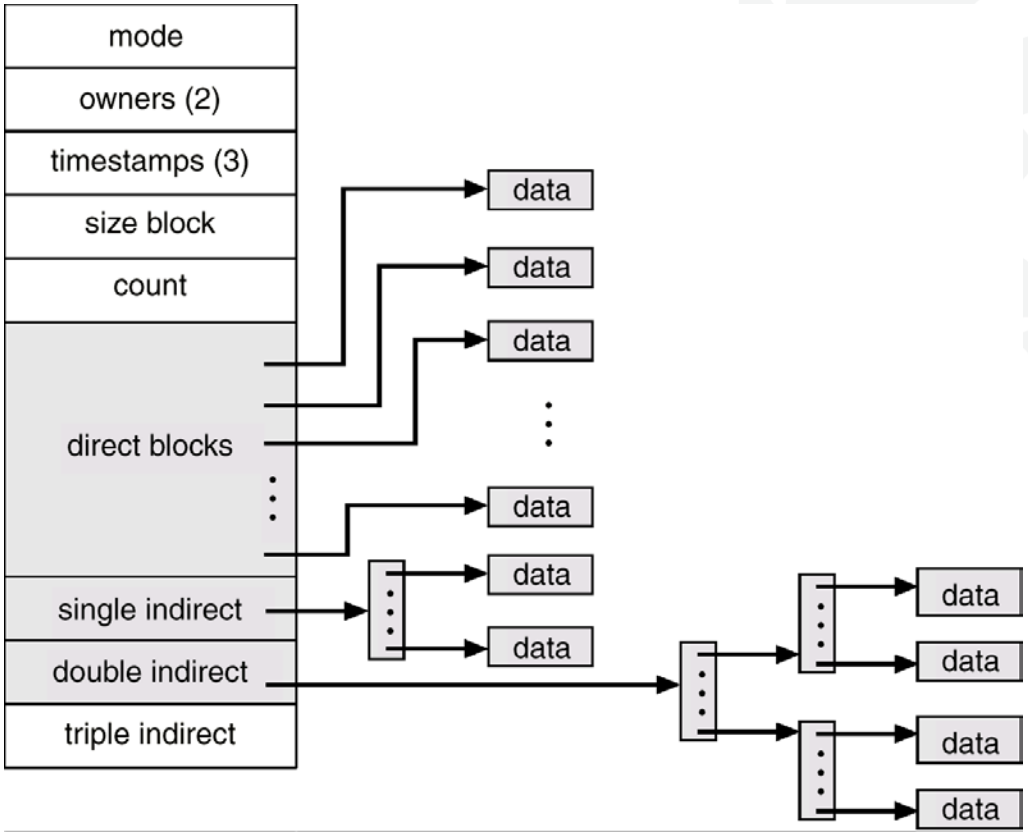


디스크 상의 i-node

- System V
  - 16 4 Byte words
  - 3 Byte block address
- BSD and recent UNIX
  - 32 4 Byte words
  - 4 Byte block address



i-node의 구조


mode
owners (2)
timestamps (3)
size block
count
direct blocks
single indirect
double indirect
triple indirect
data
data
data
data
data
data
data
data
data
data

디렉토리 파일

- i-node는 파일의 이름을 가지고 있지 않다.
- 각 파일의 이름은 디렉토리 파일에 기록된다.
- 디렉토리는 파일의 일종으로 파일의 이름과 해당하는 i-node의 번호를 기록한 파일이다.

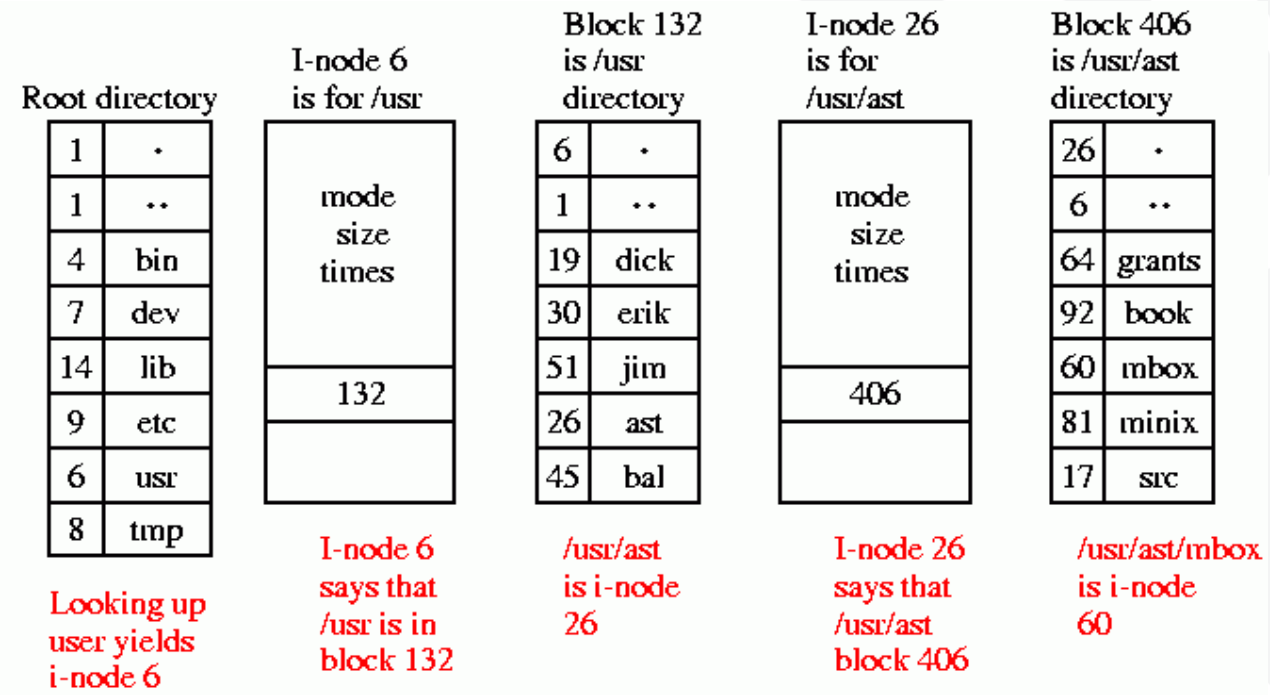
Component Name	Inode Number

Directory Entry

.	1
..	1
unix	117
etc	4
home	18
pro	36
dev	93

파일 접근

- /usr/ast/mbox에 접근하는 방법

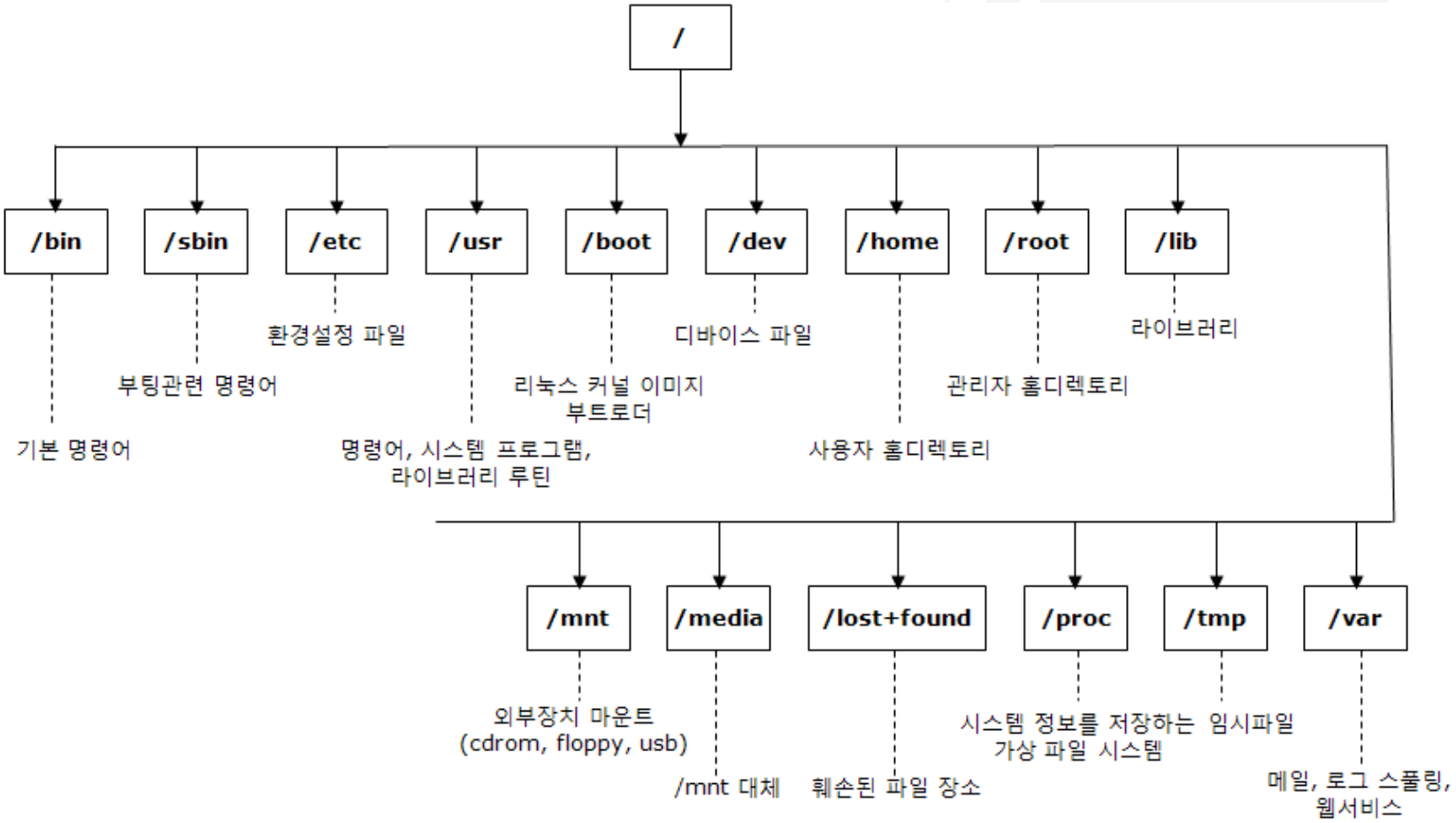


File & Directory

파일의 종류

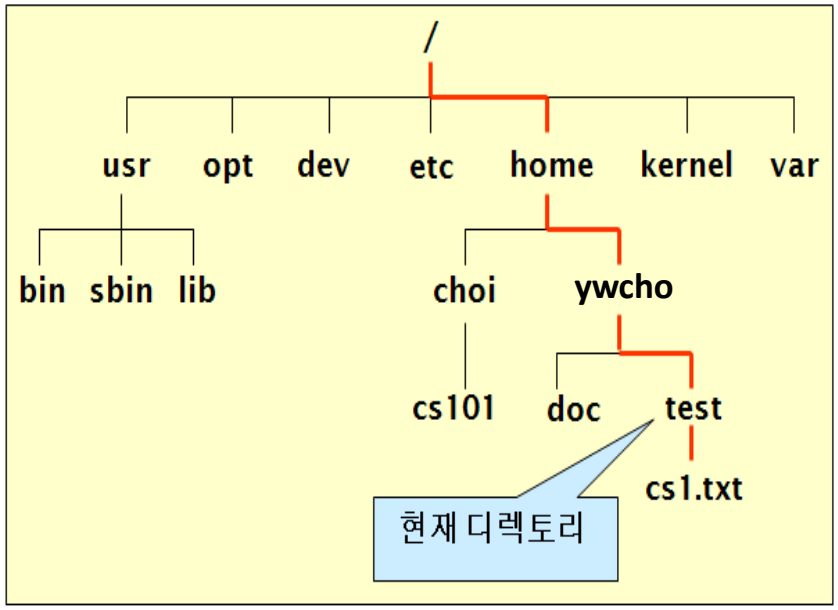
- 일반 파일(ordinary file)
  - 데이터를 가지고 있으면서 디스크에 저장된다.
  - 텍스트 파일, 이진 파일
- 디렉터리(directory) 또는 폴더(folder)
  - 파일들을 계층적으로 조직화하는 데 사용되는 일종의 특수 파일
  - 디렉터리 내에 파일이나 서브디렉토리들이 존재한다.
- 장치 파일(device special file)
  - 물리적인 장치에 대한 내부적인 표현
  - 키보드(stdin), 모니터(stdout), 프린터 등도 파일처럼 사용
- 심볼릭 링크 파일
  - 어떤 파일을 가리키는 또 하나의 경로명을 저장하는 파일

디렉터리 계층구조


/
/bin
/sbin
/etc
/usr
/boot
/dev
/home
/root
/lib
기본 명령어
부팅관련 명령어
환경설정 파일
명령어, 시스템 프로그램,
라이브러리 루틴
리눅스 커널 이미지
부트로더
디바이스 파일
사용자 홈디렉토리
관리자 홈디렉토리
라이브러리
/mnt
/media
/lost+found
/proc
/tmp
/var
외부장치 마운트
(cdrom, floppy, usb)
/mnt 대체
훼손된 파일 장소
시스템 정보를 저장하는 임시파일
가상 파일 시스템
메일, 로그 스풀링,
웹서비스

Home directory

- 홈 디렉터리(home directory)
  - 각 사용자마다 별도의 홈 디렉터리가 있음
  - 사용자가 로그인하면 홈 디렉터리에서 작업을 시작함



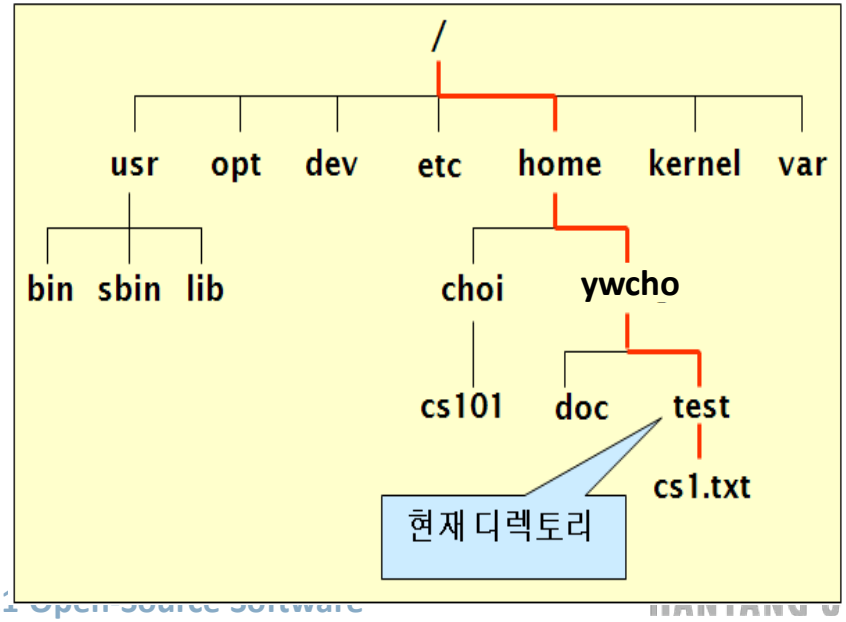
cs1.txt의 절대 경로명
/home/ywcho/test/cs1.txt

cs1.txt의 상대 경로명
cs1.txt

~ : 홈 디렉터리
. : 현재 디렉터리
.. : 부모 디렉터리

경로명

- 파일이나 디렉터리에 대한 정확한 이름
- 절대 경로명(absolute pathname)
  - 루트 디렉터리로부터 시작하여 경로 이름을 정확하게 적는 것
- 상대 경로명(relative path name)
  - 현재 작업 디렉터리부터 시작해서 경로 이름을 적는 것



cs1.txt의 절대 경로명
/home/ywcho/test/cs1.txt

cs1.txt의 상대 경로명
cs1.txt

디렉터리 명령어

디렉터리 리스트 : ls(list)

- 사용법

$ ls(혹은 dir) [-aslFR] 디렉터리* 파일*

지정된 디렉터리의 내용을 리스트 한다. 디렉터리를 지정하지 않으면 현재 디렉터리 내용을 리스트 한다.

또한 파일을 지정하면 해당 파일만을 리스트 한다.

```
$ ls
Desktop Video Music Documents Downloads Pictures Class
```

ls 명령어 옵션

옵션	기능
-a	숨겨진 파일을 포함하여 모든 파일을 리스트한다.
-s	파일의 크기를 K 바이트 단위로 출력한다.
-l	파일의 상세 정보를 출력한다.
-F	파일의 종류를 표시하여 출력한다.
-R	모든 하위 디렉터리들을 리스트한다.

ls 명령어 옵션

- **ls -s**
  - -s(size) 옵션
  - 디렉터리 내에 있는 모든 파일의 크기를 K 바이트 단위로 출력

```
$ ls -s
총 4
4 cs1.txt
```

- **ls -a**
  - -a(all) 옵션
  - 숨겨진 파일들을 포함하여 모든 파일과 디렉터리를 리스트
  - "."은 현재 디렉터리, ".."은 부모 디렉터리

```
$ ls -a
. .. cs1.txt
```

ls 명령어 옵션

- **ls -l**
  - -l(long) 옵션
  - 파일 속성(file attribute) 출력
    - 파일 이름, 파일 종류, 접근권한, 소유자, 크기, 수정 시간 등
- **ls -sl**
  - -s(size) 옵션과 -l(long) 옵션을 함께 출력

```
$ ls -sl cs1.txt
4  -rw-r--r--  1  ywcho  cs  2088  4월 16일  13:37  cs1.txt
①  ②    ③     ④   ⑤     ⑥   ⑦      ⑧          ⑨
```

① 파일크기 ② 파일종류 ③ 접근권한 ④ 링크수 ⑤ 사용자 ID ⑥ 그룹 ID ⑦ 파일 크기
⑧ 최종 수정 시간 ⑨ 파일이름

접근권한

- 파일에 대한 읽기(r), 쓰기(w), 실행(x) 권한

권한	파일	디렉터리
r	파일에 대한 읽기 권한	디렉터리 내에 있는 파일명을 읽을 수 있는 권한
w	파일에 대한 쓰기 권한	디렉터리 내에 파일을 생성하거나 삭제할 수 있는 권한
x	파일에 대한 실행 권한	디렉터리 내로 탐색을 위해 이동할 수 있는 권한

- 소유자(owner)/그룹(group)/기타(others)로 구분하여 관리
- 예: rwx r-x r-x

접근권한의 예

접근권한	의미
rwxrwxrwx	소유자, 그룹, 기타 사용자 모두 읽기,쓰기,실행 가능
rwxr-xr-x	소유자만 읽기,쓰기,실행 가능, 그룹, 기타 사용자는 읽기,실행 가능
rw-rw-r--	소유자와 그룹만 읽기,쓰기 가능, 기타 사용자는 읽기만 가능
rw-r--r--	소유자만 읽기,쓰기 가능, 그룹과 기타 사용자는 읽기만 가능
rw-r-----	소유자만 읽기,쓰기 가능 그룹은 읽기만 가능
rwx------	소유자만 읽기,쓰기,실행 가능

ls 명령어 옵션

- ls –asl
  - -a(all) 옵션,-s(size) 옵션, -l(long) 옵션을 함께 출력

```
$ ls -asl
총 8
0 drwxr-xr-x 2 ywcho cs 20 3월 16일 13:37 .
4 drwx------ 3 ywcho cs 4096 3월 16일 13:37 ..
4 -rw-r--r-- 1 ywcho cs 2088 3월 16일 13:37 cs1.txt
```

ls 명령어 옵션

- ls –F
  - 기호로 파일의 종류를 표시

    *: 실행파일, /: 디렉터리, @:심볼릭 링크

```
$ ls -F /
Desktop/ Video/ Music/ Documents/ Downloads/ Pictures/
Class/
```

ls 명령어 옵션

- ls –R
  - -R(recursive) 옵션
  - 모든 하위 디렉터리 내용을 리스트 한다.

```
$ ls -R
.:
Desktop Video Music Documents Downloads Pictures Class
./Desktop
./Video
./Music
./Documents
./Downloads
./Pictures
./Class
cs1.txt
```

현재작업 디렉터리 출력 : pwd(print working directory)

$ pwd

현재 작업 디렉터리의 절대 경로명을 출력한다.

- 현재 작업 디렉터리(current working directory)
  - 현재 작업 중인 디렉터리
  - 로그인 하면 홈 디렉터리에서부터 작업이 시작된다.

```
$ pwd
/home/ywcho
```

디렉터리 이동 : cd(change directory)

$ cd [디렉터리]

현재 작업 디렉터리를 지정된 디렉터리로 이동한다.

디렉터리를 지정하지 않으면 홈 디렉터리로 이동한다.

```
$ pwd
/home/ywcho
$ ls
Desktop Video Music Documents Downloads Pictures Class
$ cd Desktop
$ pwd
/home/ywcho/Desktop
$ cd
$ pwd
/home/ywcho
```

명령어의 경로 확인 : which

$ which 명령어

명령어의 절대경로를 보여준다.

```
$ which ls
/bin/ls
$ which pwd
/usr/pwd
$ which passwd
/usr/passwd
```

디렉터리 생성 : mkdir(make directory)

$ mkdir [-p] 디렉터리

디렉터리(들)을 새로 만든다.

```
$ cd
$ mkdir test
$ mkdir test2 temp
$ ls -l
drwxrwxr-x. 2 ywcho cs 6 3월 16 10:12 temp
drwxrwxr-x. 2 ywcho cs 6 3월 16 10:12 test2
drwxrwxr-x. 2 ywcho cs 6 3월 16 10:12 test
```

디렉터리 생성 : mkdir(make directory)

$ mkdir [-p] 디렉터리

디렉터리(들)을 새로 만든다.

- **중간 디렉터리 자동 생성 옵션 : -p**
  - 필요한 경우에 중간 디렉터리를 자동으로 만들어 준다.

~/dest 디렉터리가 없는 경우

```
$ mkdir dest/dir1
mkdir: '/home/ywcho/dest/dir1' 디렉터리를 만들 수 없습니다: 그런 파일이나 디렉터리가 없습니다
$ mkdir -p dest/dir1
```

디렉터리 삭제 : rmdir(remove directory)

$ rmdir 디렉터리

디렉터리(들)을 삭제한다.

- 주의: 빈 디렉터리만 삭제할 수 있다.

```
$ rmdir class
rmdir: failed to remove ‘class’: 디렉터리가 비어있지 않음
```

파일 생성 및 편집

간단한 파일 만들기

$ vi [파일이름]

리눅스에서 제공하는 파일 편집기이다.

- File은 사용자가 편집하고자 하는 파일의 이름
- File명을 지정하지 않고 시작할 수도 있음

간단한 파일 만들기

- vi로 작업을 하면 모든 데이터는 편집버퍼(editing buffer)에 유지됨
- 입력모드(input mode)
  - 입력하는 모든 것이 편집버퍼에 입력됨
- 명령모드(command mode)
  - 입력하는 모든 것이 명령어로 해석됨
  - 입력모드에서 명령모드로 전환하는 방법 : ESC키

vi 명령어를 배우는 전략

- 편집버퍼에 데이터를 넣을 때
  1. 데이터를 쓰고 싶은 곳으로 커서를 옮긴다.
  2. 입력모드로 바꾸기 위한 명령을 사용한다.
  3. 데이터를 입력한다.
  4. 명령모드로 바꾸기 위해 ESC를 누른다.

- vi 마치기
  - 종료 명령을 입력할 수 있는 명령 모드에 있어야 함
  - :wq : 작업 내용을 저장하고 종료
  - :q! : 작업 내용을 저장하지 않고 종료
  - 종료시 데이터를 저장했는지 점검

생성 파일 출력 : cat

$ cat [-n] [파일이름]

파일(들)의 내용을 그대로 화면에 출력한다. 파일을 지정하지 않으면 표준입력 내용을 그대로 화면에 출력한다.

```
$ cat cs1.txt
Hello world!
```