

# 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

# 우분투 리눅스 시스템 & 서버



# 제3장 명령어

# 3.1 기본 명령어

# 기본 명령어 사용

- 날짜 및 시간 확인

```
$ date
```

```
2024. 01. 01. (월) 12:26:10 KST
```

- 시스템 정보 확인

```
$ hostname
```

```
Ubuntu
```

```
$ uname
```

```
Linux
```

```
$ uname -a
```

```
Linux ubuntu 6.8.0-31-generic #31-Ubuntu SMP PREEMPT_DYNAMIC Sat Apr 20  
00:40:06 UTC 2024 x86_64 x86_64 x86_64 GNU/Linux
```

# 기본 명령어 사용

- 사용자 정보 확인

```
$ whoami  
chang
```

```
$ who
```

```
chang seat0 2024-04-04 13:51 (login screen)
```

```
chang tty2 2024-04-04 13:51 (tty2)
```

- 디렉터리 내용 확인

```
$ ls
```

공개    다운로드    문서    바탕화면    비디오    사진    음악    템플릿

# 기본 명령어 사용

- 패스워드 변경

`$ passwd`

chang를 위한 비밀번호 변경하기.

현재 비밀번호:

신규 비밀번호:

신규 비밀번호 재 입력:

`passwd`: 암호를 성공적으로 업데이트했습니다.

- 화면 정리

`$ clear`

# 온라인 매뉴얼: man

```
$ man ls
```

```
LS(1) User Commands LS(1)
```

```
NAME
```

```
ls - list directory contents
```

```
SYNOPSIS
```

```
ls [OPTION]... [FILE]...
```

```
DESCRIPTION
```

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 ..

Manual page ls(1) line 1 (press h for help or q to quit)



# 명령어에 대한 간단한 설명: whatis

```
$ whatis ls
```

```
ls (1) - directory contents
```

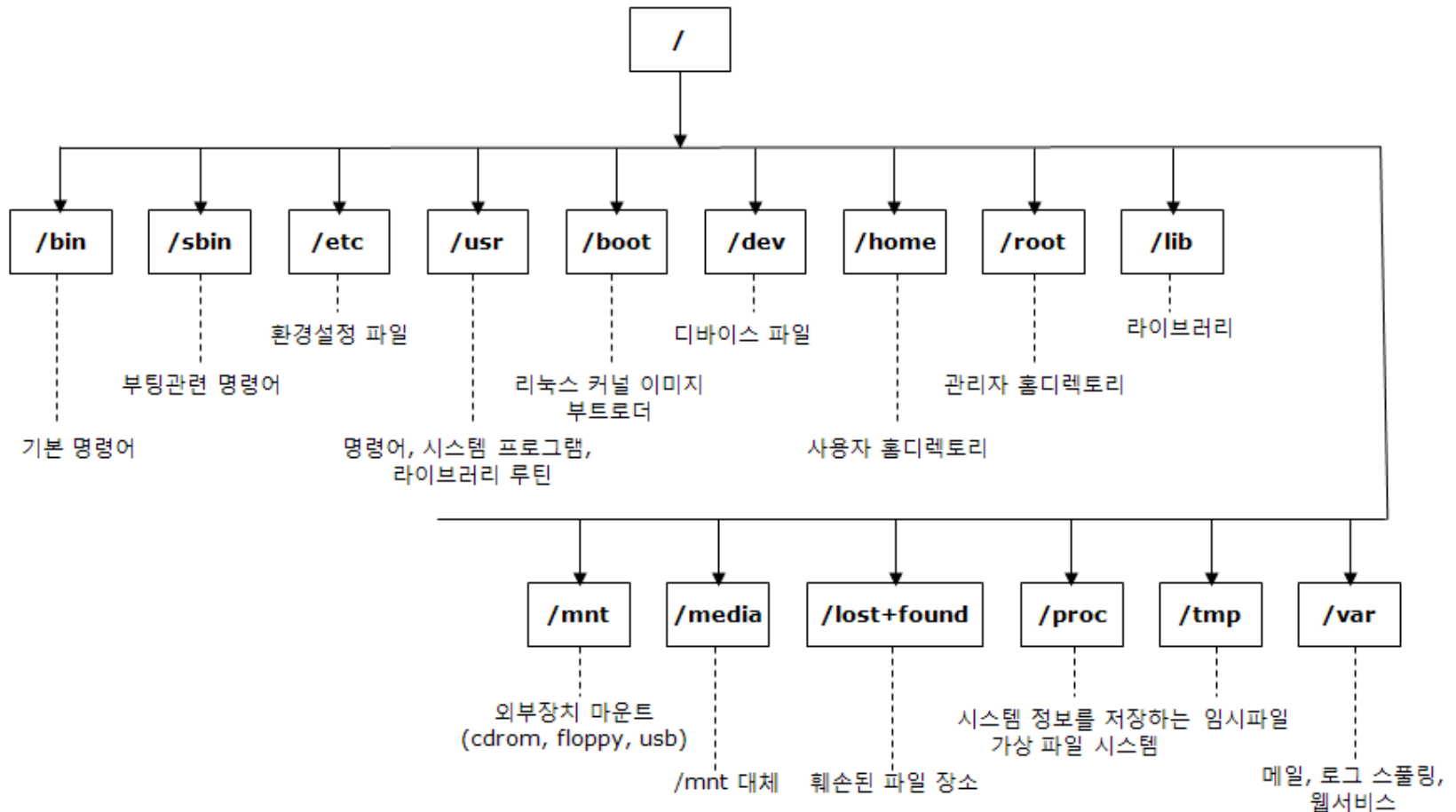
## 3.2 파일과 디렉터리

# 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

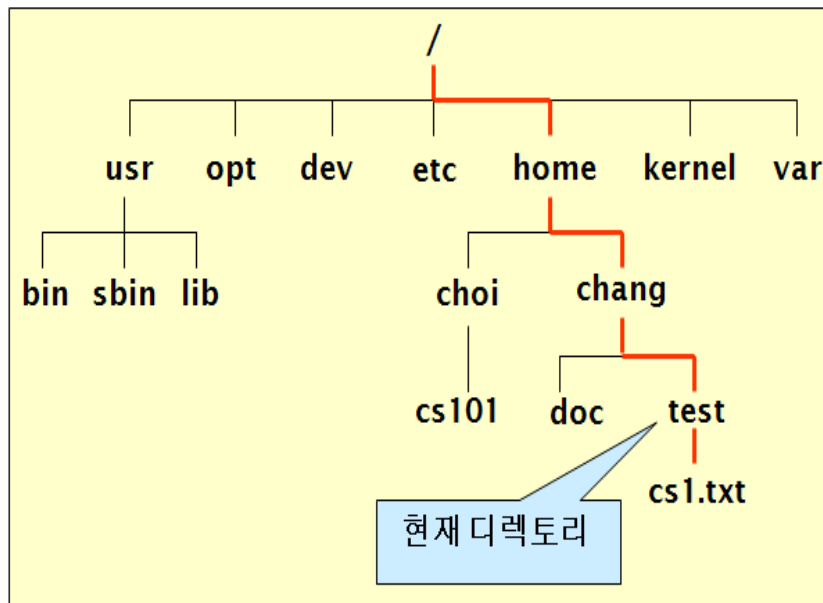
# 디렉터리 계층구조

- 리눅스의 디렉터리는 루트로부터 시작하여 트리 형태의 계층구조를 이룬다.



# 홈 디렉터리

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



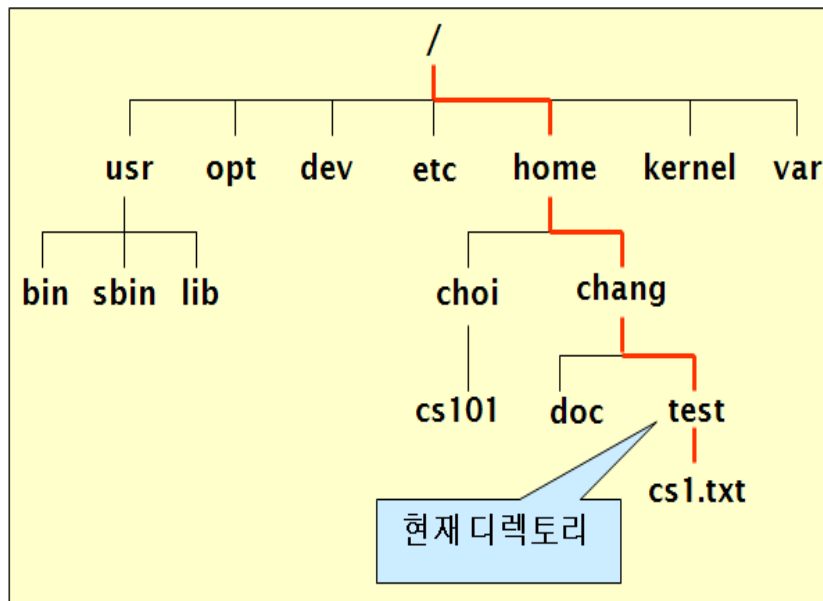
~ : 홈 디렉터리  
 . : 현재 디렉터리  
 .. : 부모 디렉터리

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

cs1.txt의 상대 경로명  
 cs1.txt

# 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/chang/test/cs1.txt**

cs1.txt의 상대 경로명  
**cs1.txt**

# 명령어의 경로 확인: which

- 사용법

`$ which 명령어`

명령어의 절대경로를 보여준다.

- 예

`$ which ls`

`/bin/ls`

`$ which pwd`

`/usr/pwd`

`$ which passwd`

`/usr/passwd`

## 3.3 디렉터리 명령어



## 현재 작업 디렉터리 출력: `pwd`(print working directory)

- 사용법

```
$ pwd
```

현재 작업 디렉터리의 절대 경로명을 출력한다.

- 현재 작업 디렉터리(current working directory)

- 현재 작업 중인 디렉터리
- 로그인 하면 홈 디렉터리에서부터 작업이 시작된다.

- 예

```
$ pwd
```

```
/home/chang/바탕화면
```

```
$ cd ~
```

```
$ pwd
```

```
/home/chang
```

# 디렉터리 이동: cd(change directory)

- 사용법

```
$ cd [디렉터리]
```

현재 작업 디렉터리를 지정된 디렉터리로 이동한다.

디렉터리를 지정하지 않으면 홈 디렉터리로 이동한다.

- 예

```
$ cd
```

```
$ cd ~
```

```
$ cd 바탕화면
```

```
$ pwd
```

```
/home/chang/바탕화면
```

```
$ cd .. // 부모 디렉터리로 이동
```

# 디렉터리 생성: mkdir(make directory)

- 사용법

```
$ mkdir [-p] 디렉터리+
```

디렉터리(들)을 새로 만든다.

- 예

```
$ cd ~ // 홈 디렉터리로 이동
```

```
$ mkdir test
```

```
$ mkdir test temp
```

```
$ ls -l
```

```
drwxrwxr-x. 2 chang chang 6 5월 12 10:12 temp
```

```
drwxrwxr-x. 2 chang chang 6 5월 12 10:12 test
```

## 디렉터리 생성: mkdir

- 중간 디렉터리 자동 생성 옵션 `-p`
  - 필요한 경우에 중간 디렉터리를 자동으로 만들어 준다.
- 예 : `~/dest` 디렉터리가 없는 경우

```
$ mkdir ~/dest/dir1
```

`mkdir: '/home/chang/dest/dir1' 디렉터리를 만들 수 없습니다: 그런 파일이나 디렉터리가 없습니다`

```
$ mkdir -p ~/dest/dir1
```

## 디렉터리 삭제 : rmdir(remove directory)

- 사용법

```
$ rmdir 디렉터리+
```

디렉터리(들)을 삭제한다.

- 주의: 빈 디렉토리만 삭제할 수 있다.

- 예

```
$ rmdir test
```

rmdir: 'test' 제거 실패: 디렉터리가 비어있지 않음

## 3.4 디렉터리 리스트

# 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.

- 예

```
$ ls /
```

```
bin dev home lib64 mnt proc run srv tmp var  
boot etc lib media opt root sbin sys usr
```

```
$ ls ~
```

```
test  공개      다운로드  문서     바탕화면  비디오   사진     음악     템플릿
```

```
$ cd test
```

```
$ ls
```

```
cs1.txt
```

# ls 명령어 옵션

- 주요 옵션

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



## ls 명령어 옵션

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

```
$ ls -s
```

```
합계 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(long)
  - 파일 속성(file attribute) 출력
  - 블록 수, 파일 종류, 접근권한, 링크 수, 소유자명, 크기, 수정 시간, 파일 이름 등

```
$ ls -sl cs1.txt
```

합계 4

```
4 -rw-rw-r-- 1 chang chang 2088 4월 16 13:37 cs1.txt
```

①②      ③      ④      ⑤      ⑥      ⑦      ⑧      ⑨

① 블록 수 ② 파일 종류 ③ 접근권한 ④ 링크 수 ⑤ 소유자명 ⑥ 그룹명

⑦ 파일 크기 ⑧ 최종 수정 시간 ⑨ 파일이름

# ls 명령어 옵션

- ls -asl

```
$ ls -asl
```

```
합계 12
```

```
4 drwxr-xr-x 2 chang chang 4906 4월 16일 13:37 .
4 drwx----- 3 chang chang 4096 4월 16일 13:37 ..
4 -rw-r--r-- 1 chang chang 2088 4월 16일 13:37 cs1.txt
```

# ls 명령어 옵션

- ls -F

- 기호로 파일의 종류를 표시

\*: 실행파일, /: 디렉터리, @:심볼릭 링크

- 예

```
$ ls -F /
```

```
bin@ dev/ home/ lib64@ mnt/ proc/ run/ srv/ tmp/ var/  
boot/ etc/ lib@ media/ opt/ root/ sbin@ sys/ usr/
```

# ls 명령어 옵션

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

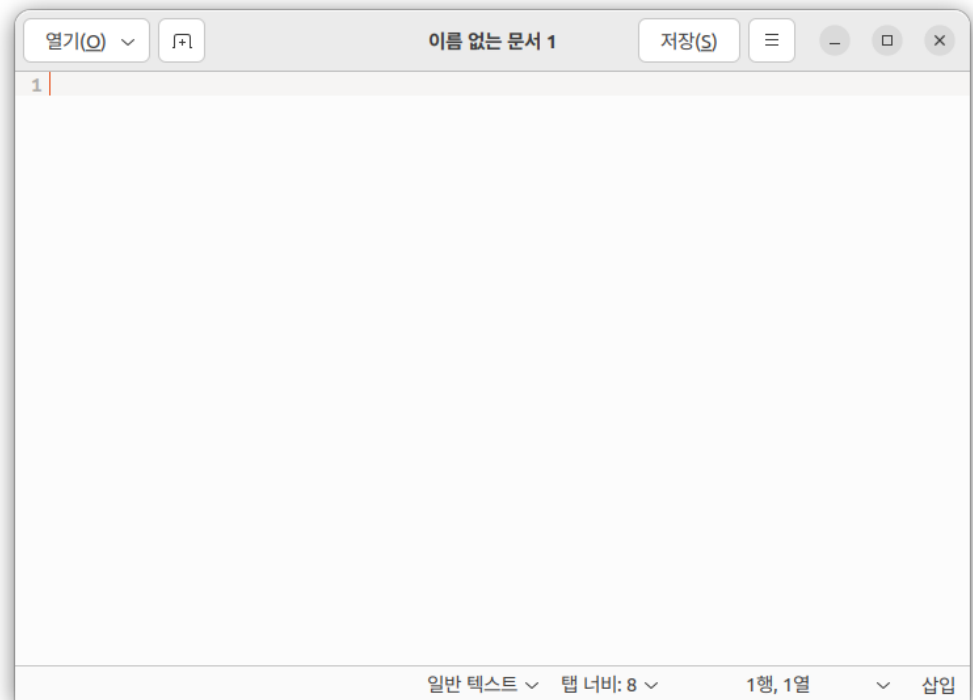
- 예

```
$ ls -R
$ ls -R /
```

## 3.5 파일 내용 출력

# 간단한 파일 만들기: gedit

- GNOME이 제공하는 GUI 기반 문서편집기
- 사용방법
  - [프로그램] -> [보조 프로그램] -> [텍스트 편집기]
  - \$ gedit [파일이름] &
- 기능
  - 열기: 파일 열기
  - 저장: 파일 저장
  - 찾기, 바꾸기
  - 보기: 강조 모드
  - 도구: 맞춤법 검사
  - 도움말



# 간단한 파일 만들기: cat

- cat 명령어 사용

```
$ cat > 파일
```

표준입력 내용을 모두 파일에 저장한다. 파일이 없으면 새로 만든다.

- 예

```
$ cat > cs1.txt
```

```
...
```

```
^D
```



## 간단한 파일 만들기: touch

- touch 명령어 사용

```
$ touch 파일
```

파일 크기가 0인 이름만 있는 빈 파일을 만들어 준다.

- 예

```
$ touch cs1.txt
```

```
$ ls -asl cs1.txt
```

```
0 -rw-rw-r--. 1 chang chang 0 5월 9 15:10 cs1.txt
```

# 파일 내용 출력

- 파일 내용 출력과 관련된 다음 명령어들
  - cat, more, head, tail, wc, 등

\$ 명령어 파일

\$ 명령어 파일\*

# 파일 내용 보기: cat

- 사용법

```
$ cat [-n] 파일*
```

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

- 예

```
$ cat cs1.txt
```

Unix is a multitasking, multi-user computer operating system originally developed in 1969 by a group of AT&T employees at Bell Labs, including Ken Thompson, Dennis Ritchie, Brian Kernighan, Douglas McIlroy, and Joe Ossanna.

...

# 파일 내용 보기: cat

- 예

```
$ cat -n cs1.txt
```

```
1 Unix is a multitasking, multi-user computer operating system originally
2 developed in 1969 by a group of AT&T employees at Bell Labs, including
3 Ken Thompson, Dennis Ritchie, Brian Kernighan, Douglas McIlroy,
4 and Joe Ossanna.
```

```
...
```

```
$ cat // 지정 파일 없음
```

```
Hello World !
```

```
Hello World !
```

```
Bye!
```

```
Bye!
```

```
^D
```

# 페이지 단위로 파일 내용 보기: more

- 사용법

```
$ more 파일+
```

파일(들)의 내용을 페이지 단위로 화면에 출력한다.

- 예

```
$ more cs1.txt
```

Unix is a multitasking, multi-user computer operating system originally developed in 1969 by a group of AT&T employees at Bell Labs, including Ken Thompson, Dennis Ritchie, Brian Kernighan, Douglas McIlroy, and Joe Ossanna.

...

During the late 1970s and early 1980s, the influence of Unix in academic circles led to large-scale adoption of Unix(particularly of the BSD variant,

--계속--(59%)

# 파일 앞부분보기: head

- 사용법

```
$ head [-n] 파일*
```

파일(들)의 앞부분을 화면에 출력한다. 파일을 지정하지 않으면 표준입력 내용을 대상으로 한다.

- 예

```
$ head -5 cs1.txt
```

Unix is a multitasking, multi-user computer operating system originally developed in 1969 by a group of AT&T employees at Bell Labs, including Ken Thompson, Dennis Ritchie, Brian Kernighan, Douglas McIlroy, and Joe Ossanna.

# 파일 뒷부분보기: tail

- 사용법

```
$ tail [-n] 파일*
```

파일(들)의 뒷부분을 화면에 출력한다. 파일을 지정하지 않으면 표준입력 내용을 대상으로 한다.

- 예

```
$ tail cs1.txt
```

Linux, which is used to power data centers, desktops, mobile phones, and embedded devices such as routers, set-top boxes or e-book readers. Today, in addition to certified Unix systems such as those already mentioned, Unix-like operating systems such as MINIX, Linux, Android, and BSD descendants (FreeBSD, NetBSD, OpenBSD, and DragonFly BSD) are commonly encountered.

The term traditional Unix may be used to describe a Unix or an operating system that has the characteristics of either Version 7 Unix or UNIX System V.

# 단어 세기: wc(word count)

- 사용법

```
$ wc [-lwc] 파일*
```

파일에 저장된 줄(l), 단어(w), 문자(c)의 개수를 세서 출력한다.

파일을 지정하지 않으면 표준입력 내용을 대상으로 한다.

- 예

```
$ wc cs1.txt
38 318 2088 cs1.txt
$ wc -l cs1.txt
38 cs1.txt
$ wc -w cs1.txt
318 cs1.txt
$ wc -c cs1.txt
2088 cs1.txt
```



## 3.6 vi 에디터

# vi 에디터

- vi 에디터

- 기본 텍스트 에디터로 매우 강력한 기능을 가지고 있으나
- 배우는데 상당한 시간과 노력이 필요하다.

```
$ sudo apt install vim
```

\$ vi 파일\*

```

1
vim - 항상된 Vi

판 9.1.16
by Bram Moolenaar et al.
Modified by team+vim@tracker.debian.org
vim은 누구나 소스를 볼 수 있고 공짜로 배포됩니다

vim 개발을 후원해 주세요 !

이에 대한 정보를 보려면      :help sponsor<엔터>      입력

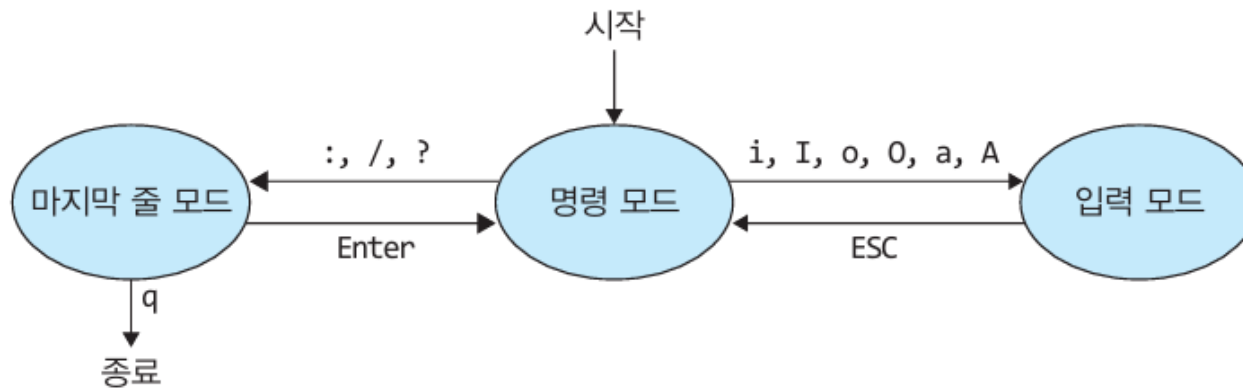
끝내려면                      :q<엔터>              입력
온라인 도움말을 보려면      :help<엔터> 또는 <F1>      입력
판 정보를 보려면            :help version9<엔터>      입력

0,0-1                      모두

```

# 명령 모드/입력 모드

- vi 에디터는 명령 모드와 입력 모드가 구분되어 있으며
- 시작하면 명령 모드이다.



- 마지막 줄 모드
  - :set nu // 줄번호 붙이기
  - :syntax on // 구문 강조
  - :wq // 저장하고 끝내기

## vi 내부 명령어

- 원하는 위치로 이동하는 명령
- 입력모드로 전환하는 명령
- 수정 혹은 삭제 명령
- 복사 및 붙이기
- 기타 명령

# 간단한 C 프로그램 작성

## (1) vi test.c (그림 3.6)

test.c 파일을 작성하기 위해 vi 에디터를 시작한다.

```
:set nu           // 줄번호 붙이기
```

```
:syntax on       // 구문 강조
```

## (2) 입력 모드로 전환 (그림 3.7)

i 명령어를 사용하여 입력 모드로 전환하면 좌측하단에 “— 끼 워 넣 기 —” 라고 표시

## (3) 텍스트 입력 (그림 3.8)

이제 입력하고자 하는 텍스트(간단한 C 프로그램)를 입력하고

입력이 끝나면 ESC 키를 쳐서 입력 모드에서 명령 모드로 돌아온다.

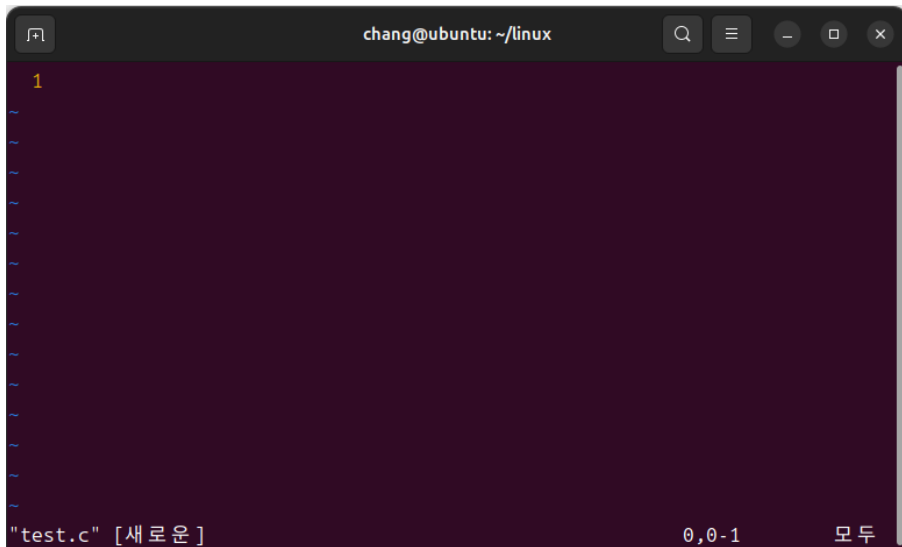
## (4) 텍스트 저장하고 끝내기 (그림 3.9)

마지막 줄 모드에서 텍스트를 저장하고 끝낸다.

```
:wq
```

# vi 에디터

- vi 에디터 시작

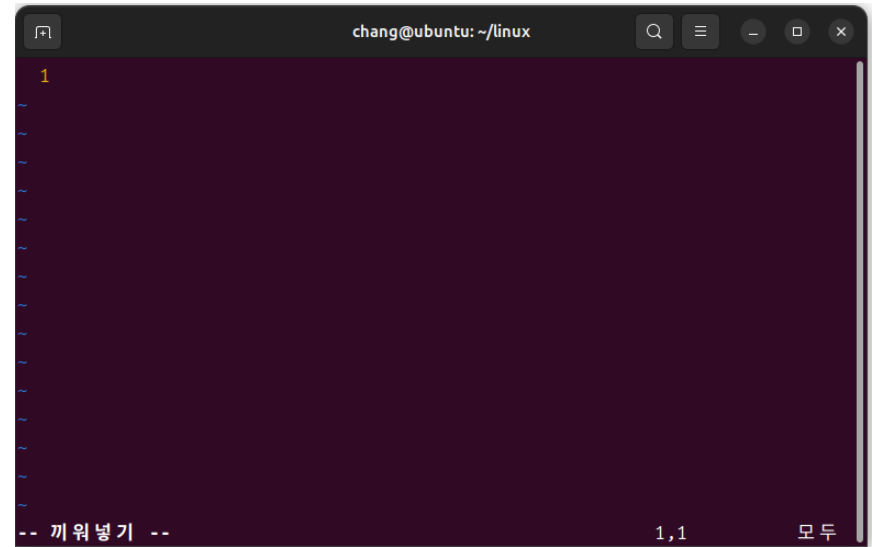


```
chang@ubuntu: ~/linux
```

1

"test.c" [새로운] 0,0-1 모두

- 입력 모드로 전환



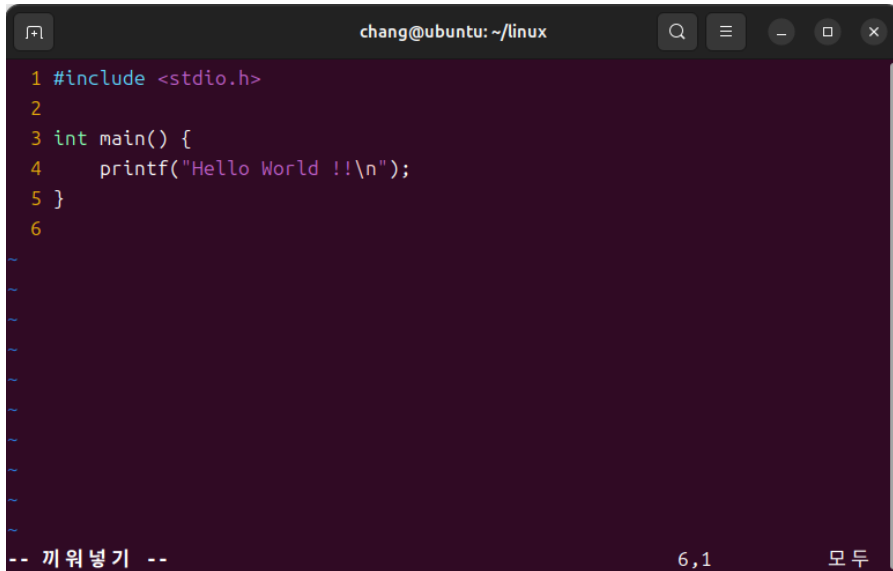
```
chang@ubuntu: ~/linux
```

1

-- 끼워넣기 -- 1,1 모두

# vi 에디터

- 텍스트 입력

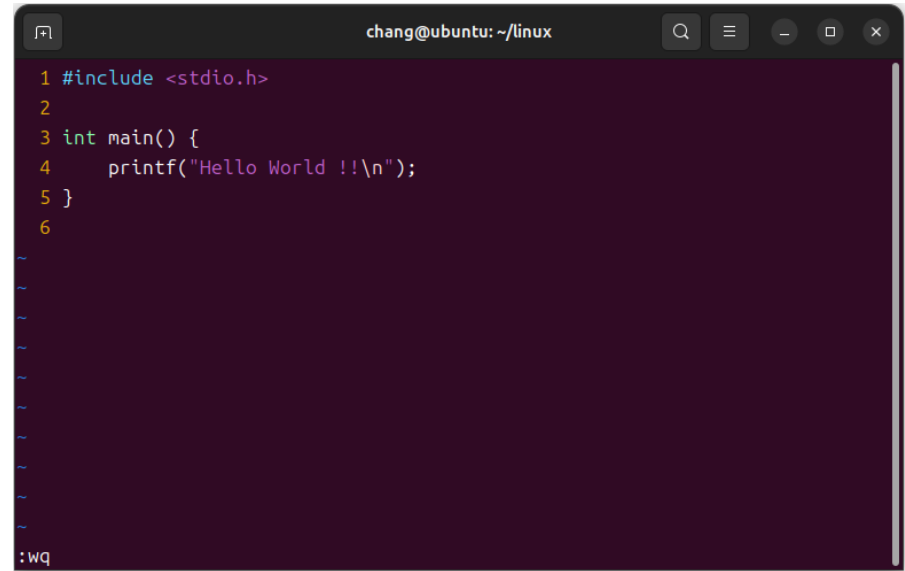


A screenshot of the vi editor interface in insert mode. The window title is 'chang@ubuntu: ~/linux'. The code editor shows a C program with 6 lines: `1 #include <stdio.h>`, `2`, `3 int main() {`, `4 printf("Hello World !!\n");`, `5 }`, and `6`. The status bar at the bottom left shows '-- 끼워넣기 --' (insert mode), and the bottom right shows '6,1' and '모두' (all).

```
1 #include <stdio.h>
2
3 int main() {
4     printf("Hello World !!\n");
5 }
6
```

-- 끼워넣기 -- 6,1 모두

- 저장하고 끝내기



A screenshot of the vi editor interface in command mode. The window title is 'chang@ubuntu: ~/linux'. The code editor shows the same C program as the previous screenshot. The status bar at the bottom right shows ':wq', indicating the command to save and quit.

```
1 #include <stdio.h>
2
3 int main() {
4     printf("Hello World !!\n");
5 }
6
```

:wq

# 원하는 위치로 이동 명령어

- 커서 이동

h, ←	한 칸 왼쪽
j, ↓	한 칸 아래쪽
k, ↑	한 칸 위쪽
l, →	한 칸 오른쪽
BACKSPACE	왼쪽으로 한 칸
SPACE	오른쪽으로 한 칸
-	이전 줄의 처음
+	다음 줄의 처음
RETURN	다음 줄의 처음
0	현재 줄의 맨 앞
\$	현재 줄의 끝
^	현재 줄의 첫 글자
W	다음단어의 첫 글자
B	이전단어의 첫 글자

- 화면 이동

^F	한 화면 아래로
^B	한 화면 위로
^D	반 화면 아래로
^U	반 화면 위로

- 특정 줄로 이동

nG	n번째 줄로 이동
1G	첫 줄로 이동하기
G	마지막 줄로 이동하기
:n	n번째 줄로 이동

- 탐색(search)

/탐색패턴	forward 탐색
?탐색패턴	backward 탐색



# 입력모드로 전환 명령어

명령어	기능
i	커서 위치 앞에 삽입(insert)
a	커서 위치 뒤에 삽입(append)
I	현재 줄의 앞에 삽입(Insert)
A	현재 줄의 뒤에 삽입(Append)
O	현재 줄의 아래에 전개
O	현재 줄의 위에 전개

```

1 #include <stdio.h>
2 /* hello world 프로그램 */
3
4 int main() {
5     printf("Hello World !!\n");
6 }

```

-- 끼워넣기 -- 3,1 모두

# 수정 명령어

- 현재 커서를 중심으로 수정

수정 명령어	기능
cw	현재 단어를 삭제하고 삽입 상태로 변경
cc	현재 줄 전체를 삽입에 의해 변경
C	커서의 위치로부터 줄 끝까지 삽입에 의한 변경
r	현재 글자를 r 다음에 입력한 한 글자로 변경 (입력 모드로 바뀌지 않음)
R	입력하는 대로 겹쳐 써서 변경
s	현재 글자를 삭제하고 삽입 상태로 변경

# 대치, 수행취소/재수행

## ● 대치 명령

대치 명령어	기능
:s/문자열패턴/문자열	현재 줄에서 해당되는 첫 번째 문자열 대치
:s/문자열패턴/문자열/g	현재 줄에서 해당되는 모든 문자열 대치
:n,m s/문자열패턴/문자열	지정된 줄 범위(n부터 m까지)에서 각 줄의 첫 번째 해당 문자열 대치
:n,m s/문자열패턴/문자열/g	현재 글자를 r 다음에 입력한 한 글자로 변경 (입력 모드로 바뀌지 않음)

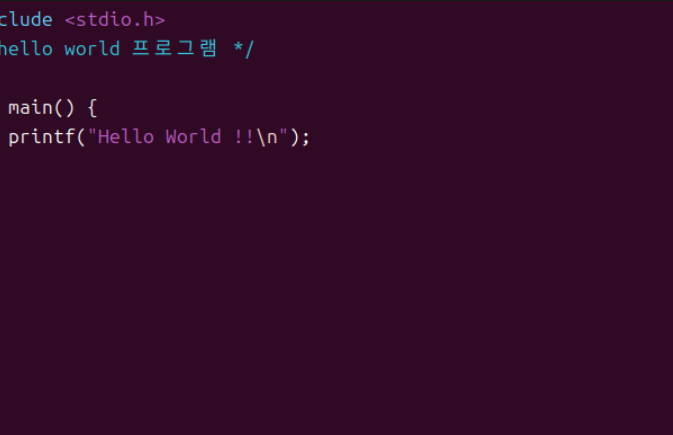
## ● 수행취소/재수행

- u     방금 전 수행 내용 취소(Undo)
- U     현재 줄 수행 내용을 취소
- .     방금 전 수행 내용을 반복(Redo)

# 삭제 명령어

- 현재 커서를 중심으로 삭제

삭제 명령어	기능
x	커서가 있는 문자 지우기
X	커서의 왼쪽 문자 지우기
D	커서부터 줄의 끝까지 지우기
dd	현재 줄의 전체를 지우기
:n,m d	n번째 줄에서 m번째 줄까지 모두 지우기

- 
- The screenshot shows a terminal window with a dark background. The title bar at the top reads "chang@ubuntu: ~/linux". The terminal content is as follows:
- ```
1 #include <stdio.h>
2 /* hello world 프로그램 */
3
4 int main() {
5     printf("Hello World !!\n");
6 }
```
- On the left side of the terminal, there is a vertical column of tilde (~) characters. At the bottom left, the prompt ":wq" is visible.

# 기타

- 다른 파일 편집

- :e 파일이름  
현재 파일 대신에 주어진 파일 열기
- :e#  
이전 파일을 다시 열기

- 줄 번호 붙이기

줄 번호를 붙이거나 없애기

:set number 줄번호 붙이기

:se nu 줄번호 붙이기

:set nonumber 줄번호 없애기

:se non 줄번호 없애기

- 셸 명령어 수행

- 편집기 내에서 셸 명령어 수행
- :!ls
- :!cat test.c

## 핵심 개념

- 리눅스의 디렉터리는 루트로부터 시작하여 계층구조를 이룬다.
- 절대 경로명은 루트 디렉터리부터 시작하고 상대 경로명은 현재 디렉터리부터 시작한다.