

목차

00. 개요

01. 리눅스의 문서 편집기

02. vi 사용법

03. vi 환경 설정

01 리눅스의 문서 편집기

■ 리눅스 편집기의 종류

- GUI 환경인 그놈에서 제공하는 gedit
- 유닉스에서부터 사용했던 행 편집기(라인 편집기)와 화면 편집기

표 3-1 리눅스의 편집기 종류

구분	종류
행 단위 편집기	ed, ex, sed
화면 단위 편집기	vi, emacs
GUI 편집기	gedit

■ 행 단위 편집기

- ed : 유닉스 초기의 행 편집기로 사용이 불편하여 거의 사용 안 함.
- ex : 행 편집기이지만 단독으로 사용하기보다는 vi에 연결하여 vi를 더욱 강력하게 하는 다양한 기능을 제공
- sed : 스트림 편집기로, 일반 편집기와 달리 지시된 명령에 따라 파일의 내용을 일괄적으로 바꿔서 출력해줌

■ 화면 단위 편집기

- vi : 리눅스에서 일반적으로 사용할 수 있는 화면 편집기
- emacs(이맥스) : 제공하는 기능이 매우 다양하지만 사용법이 어렵고 복잡하여 전문적인 애호가 위주로 사용
 - GNU Emacs는 무료로 배포되며, 별도로 설치해야 함

■ 모드형과 비모드형 편집기

- 모드형
 - 입력 모드와 명령 모드가 구분
 - 입력 모드는 텍스트를 입력할 수 있는 모드이고, 명령 모드는 텍스트를 수정하거나 삭제하고 복사와 붙이기 등 편집을 하는 모드
 - 같은 글자라도 입력 모드에서는 텍스트로 처리하여 입력되고, 명령 모드에서는 텍스트로 입력되는 것이 아니라 편집 명령으로 사용
 - vi는 모드형 편집기
- 비모드형
 - 입력 모드와 명령 모드가 구분되어 있지 않음
 - 편집 기능을 Ctrl이나 Alt 같은 특수 키와 함께 사용
 - 한글과 워드는 비모드형 편집기

표 3-2 모드형과 비모드형 편집기의 비교

구분		모드형(vi)	비모드형(메모장)
입력 모드		텍스트 입력	
명령 모드의 예	복사	yy	Ctrl + c
	붙이기	p	Ctrl + v
	저장	:wq, ZZ	Ctrl + s
모드 전환		i, a, o, Esc	해당 없음

02 vi 사용법

■ vi의 동작 모드

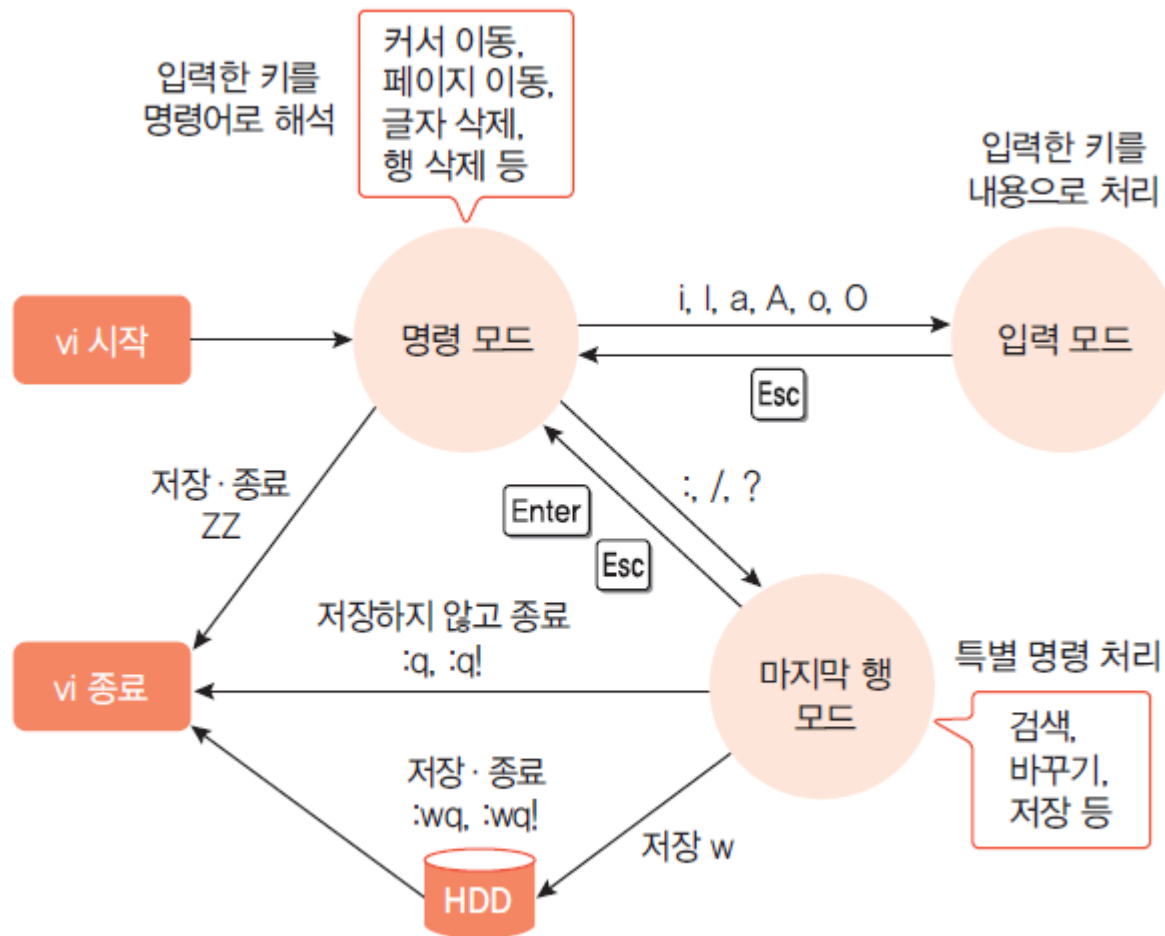


그림 3-2 vi의 모드

■ vi 시작하고 종료하기

vi

- **기능** 지정한 파일을 편집한다. 파일을 지정하지 않으면 빈 파일이 열리고 이 빈 파일의 이름은 별도로 정할 수 있다.
- **형식** vi [파일]

■ vi 시작

- 파일을 지정할 경우: 해당 파일이 있으면 파일의 내용이 보이고, 없는 파일이면 빈 파일이 열린다.

```
user1@myubuntu:~$ vi test.txt
```

→ test.txt 파일이 열린다. 이 파일이 없으면 빈 파일이 열린다.

- 파일을 지정하지 않을 경우: 그냥 빈 파일이 열린다(파일명은 저장할 때 지정 가능)

```
user1@myubuntu:~$ vi
```

→ 빈 파일이 열린다.

그림 3-3 vi의 초기 화면

■ 파일 저장하고 종료하기

- vi 종료
 - 명령모드나 마지막 행 모드에서 저장하고 종료 가능

표 3-3 vi의 저장과 종료 명령 키

모드	명령 키	기능
마지막 행 모드	:q	vi에서 작업한 것이 없을 때 그냥 종료한다.
	:q!	작업한 내용을 저장하지 않고 종료한다.
	:w [파일명]	작업한 내용을 저장만 한다. 파일명을 지정하면 새 파일로 저장한다.
	:wq, :wq!	작업한 내용을 저장하고 vi를 종료한다.
명령 모드	ZZ(Shift+zz)	작업한 내용을 저장하고 vi를 종료한다.

■ 입력 모드로 전환하기




표 3-4 입력 모드 전환 명령 키

명령 키	기능
i	커서 앞에 입력한다(현재 커서 자리에 입력한다).
a	커서 뒤에 입력한다(현재 커서 다음 자리에 입력한다).
o	커서가 위치한 행의 다음 행에 입력한다.
I(대문자 i)	커서가 위치한 행의 첫 칼럼으로 이동하여 입력한다.
A	커서가 위치한 행의 마지막 칼럼으로 이동하여 입력한다.
O	커서가 위치한 행의 이전 행에 입력한다.


■ i 명령 키를 사용해 입력 모드로 전환하기

- vi를 실행한 뒤 명령 모드에서 i 명령 키를 입력하고 나서 다음 내용을 입력

```
user1@user1-pc:~/linux_ex/ch3$ vi test.txt
```

```
ubuntu linux study    →  키를 누르면 다음 행으로 이동한다.  
I like linux    →  키를 누르면 명령 모드로 전환된다.  
~  
~  
~
```

- 입력 모드에서 다시 명령 모드로 전환하기 위해 Esc 키를 누르면 커서가 x 자 위로 이동

```
ubuntu linux study  
I like linux    → 명령 모드로 전환되고 커서가 x로 이동한다.  
~  
~  
~
```

■ i와 a 명령 키의 차이

- 명령 키 i는 커서 앞에, a는 커서 뒤에 입력
- 커서가 마지막 글자인 x 자 위에 있는 상태에서 i를 입력하고 'Space Bar+ubuntu'를 입력한 후 Esc 키를 누르면 명령 모드로 전환되어 마지막 입력 글자인 u 위에 커서가 놓임

```
ubuntu linux study
I like linu ubuntux      → 명령 모드로 전환되고 커서가 u로 이동한다.
~
~
~
```

- a 명령 키로 입력 모드로 전환한 후 'Space Bar+linu'를 입력하고 키를 누르면 커서가 위치한 a 다음부터 글자가 입력되고, Esc키를 입력하면 명령 모드로 전환되면서 마지막 입력 글자인 u 위에 커서가 놓임

```
ubuntu linux study
I like linu ubuntu linux  → 명령 모드로 전환되고 커서가 u로 이동한다.
~
~
~
```

■ o 명령 키를 사용해 입력 모드로 전환하기

- 명령 모드에서 현재 커서가 위치한 그 다음 행에 글자를 입력

```
ubuntu linux study
```

```
I like linu ubuntu linux → 명령 모드에서 o를 누르면 커서가 아랫행으로 이동한다.
```

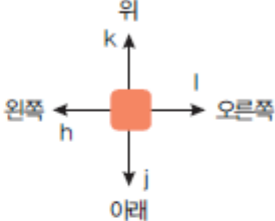

```
o
```

```
~
```

```
~
```

■ 커서 이동하기

표 3-5 커서 이동 명령 키

명령 키	기능	
k	커서를 한 행 위로 이동한다.	
j	커서를 한 행 아래로 이동한다.	
l	커서를 한 글자 오른쪽으로 이동한다.	
h	커서를 한 글자 왼쪽으로 이동한다.	
^ 또는 0	커서를 현재 행의 처음으로 이동한다.	
\$	커서를 현재 행의 마지막으로 이동한다.	
-	커서를 앞 행의 처음으로 이동한다.	
+ 또는 	커서를 다음 행의 처음으로 이동한다.	
H	커서를 화면의 맨 윗행으로 이동한다.	
M	커서를 화면의 중간 행으로 이동한다.	
L	커서를 화면의 맨 아랫행으로 이동한다.	
w	커서를 다음 단어의 첫 글자로 이동한다.	
b	커서를 앞 단어의 첫 글자로 이동한다.	
e	커서를 다음 단어의 마지막 글자로 이동한다.	

02 vi 사용법

1→ Come, have breakfast.
H 2 Look at my hands and my feet.
3 He saw and believed.
4 Who do you say that I am?
5→ I do know him and I keep his word. k
M 6→ You always have the poor with word. e w
- 7→ Forgive and you will be forgiven. \$
^ 8→ He loved his own to the end.
+ 9→ The Lord is with you.
L b j

그림 3-4 커서 이동 명령 키의 예

■ 커서 이동하기

- 초기의 유닉스 vi는 화살표 키로 커서를 이동할 수 없었음
- 리눅스 vi는 화살표 키로도 커서를 이동할 수 있음

표 3-6 기존 vi 명령 키와 화살표 키, 페이지 업·다운 키

명령 키	화살표, 페이지 업·다운 키	명령 키	화살표, 페이지 업·다운 키
k	위 화살표 키(↑)	h	왼쪽 화살표 키(←)
j	아래 화살표 키(↓)	^ 또는 O	Home
l	오른쪽 화살표 키(→)	\$	End

02 vi 사용법

■ 화면 이동하기

- 파일 크기가 터미널의 화면 크기보다 클 경우 화면을 이동시키기 위한 명령

표 3-7 화면 이동 명령 키

기존 명령 키	기능	추가 명령 키
<code>^u</code> (<code>Ctrl</code> + <code>u</code>)	반 화면 위로 이동한다.	
<code>^d</code> (<code>Ctrl</code> + <code>d</code>)	반 화면 아래로 이동한다.	
<code>^b</code> (<code>Ctrl</code> + <code>b</code>)	한 화면 위로 이동한다.	<code>Page Up</code>
<code>^f</code> (<code>Ctrl</code> + <code>f</code>)	한 화면 아래로 이동한다.	<code>Page Down</code>
<code>^y</code> (<code>Ctrl</code> + <code>y</code>)	화면을 한 행만 위로 이동한다.	
<code>^e</code> (<code>Ctrl</code> + <code>e</code>)	화면을 한 행만 아래로 이동한다.	

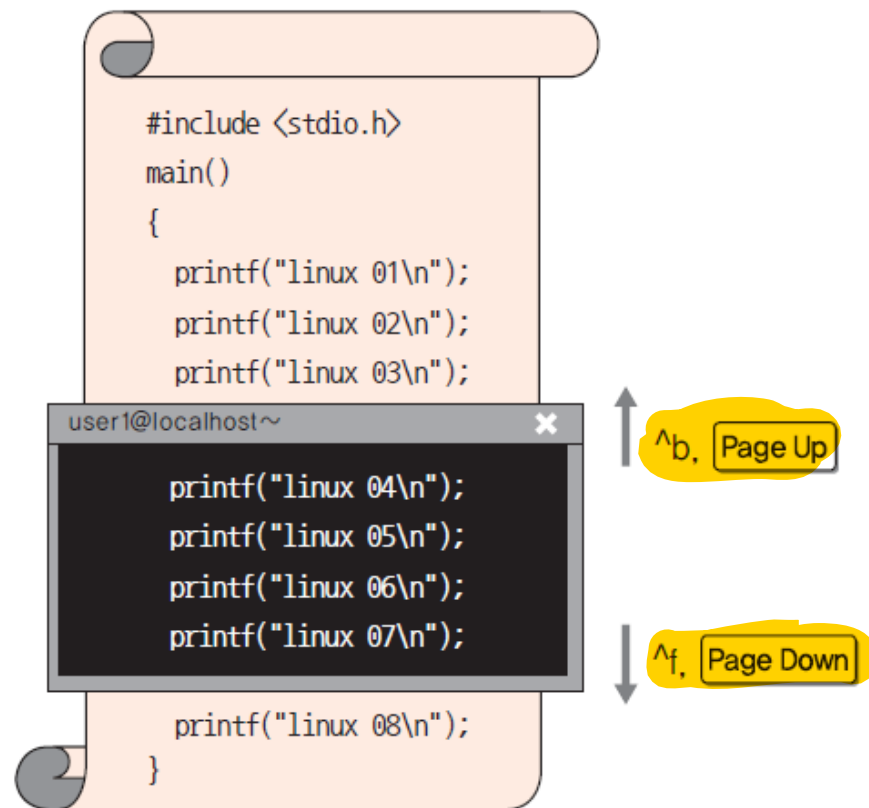


그림 3-5 화면 이동 명령 키의 사용

■ 특정 행으로 바로 이동하기


- 원하는 행으로 커서를 바로 이동
- 예:
 - 50G -> 50행으로 이동
 - :30(enter키) -> 30행으로 이동

표 3-8 특정 행으로 바로 이동하는 명령 키

명령 키	기능
G(Shift+g)	파일의 마지막 행으로 커서가 이동한다.
행 번호G(Shift+g)	지정한 행 번호로 커서가 이동한다.
:행 번호	지정한 행 번호로 커서가 이동한다(마지막 행 모드).
:\$	파일의 마지막 행으로 커서가 이동한다(마지막 행 모드).

■ 내용 수정하기

표 3-9 내용 수정 명령 키

명령 키	기능
r	커서가 위치한 글자를 다른 글자로 수정한다.
cw, #cw	커서 위치부터 현재 단어의 끝까지 수정한다. #에는 수정할 단어의 수를 지정한다. 예를 들면 3cw는 커서 위치부터 세 단어를 수정한다.
s, #s	커서 위치부터  키를 입력할 때까지 수정한다. #에는 수정할 글자의 수를 지정한다. 예를 들면 5s는 커서 위치부터 다섯 글자를 수정한다.
cc	커서가 위치한 행의 내용을 모두 수정한다.
C	커서 위치부터 행의 끝까지 수정한다.

■ 한 글자 수정하기 : r 명령 키

- 수정하려는 글자 위에 커서를 놓은 후 r 명령 키를 먼저 입력하고 바꾸려는 새 글자를 입력
- r 명령 키는 계속 명령 모드를 유지하므로 수정한 후 Esc키를 누를 필요가 없음

```
ubuntu winux study → r 명령 키로 글자를 수정한다(l→w).
```

```
I like linu ubuntu linux
```

```
~
```

■ 단어 수정하기 : cw, #s 명령 키

- 한 단어를 바꿀 때는 cw 명령 키나 #s 명령 키를 사용
- vi에서는 단어를 공백문자나 특수문자로 구별
- 예: 'winux'를 다른 단어로 바꾸려면 cw 명령 키나, 글자 수가 다섯 개이므로 5s로 수정

```
ubuntu winu$ study → cw 명령 키 입력 시 x가 $로 바뀌어 수정할 부분을 나타낸다.
```

```
I like linu ubuntu linux
```

```
~
```

```
ubuntu editorr study → 수정 완료 후 [Esc] 키를 눌러야 명령 모드로 전환된다.
```

```
I like linu ubuntu linux
```

```
~
```

■ 행 단위 수정하기 : C 명령 키

- 커서가 위치한 'r'부터 행의 끝까지 수정하려면 C 명령 키를 사용

```
ubuntu editor r stud$      → 행의 맨 끝에 $가 표시되고 입력 모드로 전환된다.  
I like linu ubuntu linux  
  
~
```

- 수정할 대상의 글자 수와 상관없이 원하는 대로 입력하면 된다. 'r'가 없어졌으므로 'r vi'로 수정하고 Esc 키를 눌러 명령 모드로 전환

```
ubuntu editor vi  
I like linu ubuntu linux  
  
~
```

■ 행 단위 수정하기 : cc 명령 키

- cc 명령 키를 입력하면 현재 행의 모든 내용이 삭제되고, 커서가 행의 처음으로 이동하여 새로운 입력을 기다림

```
I like linu ubuntu linux
~
```

→ 모두 지워지고 행의 처음으로 이동한다. 입력 모드로 전환된다.

- 원하는 내용을 입력하고 Esc 키를 눌러 명령 모드로 전환하면 수정 완료

```
ubuntu editor vi study
I like linu ubuntu linux
~
```

■ 내용 삭제하기

표 3-10 내용 삭제 명령 키

명령 키	기능
x, #x	커서 위치의 글자를 삭제한다. #에는 삭제할 글자 수를 지정한다.
dw, #dw	커서 위치의 단어를 삭제한다. #에는 삭제할 단어 수를 지정한다.
dd, #dd	커서 위치의 행을 삭제한다. #에는 삭제할 행의 수를 지정한다.
D(Shift+d)	커서 위치부터 행의 끝까지 삭제한다.

■ 글자 삭제하기

- x 명령 키로 현재 커서가 놓인 'l' 한 글자만 삭제

```
ubuntu editor vi study  
I like linu ubuntu linux
```

~

```
ubuntu editor vi study  
I like inu ubuntu linux
```

→ x 명령 키로 커서 위치의 l을 삭제한다.

~

- inu를 모두 지우려면 3x(3글자 삭제) 또는 dw(단어 삭제) 명령키 사용: 각 경우에 삭제 후 커서 위치 차이 있음

```
ubuntu editor vi study  
I like ubuntu linux
```

→ dw로 삭제하면 커서가 다음 단어의 첫 글자로 이동한다.

~

```
ubuntu editor vi study  
I like  ubuntu linux
```

→ 3x로 삭제하면 커서가 공백문자에 위치한다.

~

■ 행 삭제하기

- 현재 커서 위치부터 행의 끝까지 삭제하려면 D(Shift+d) 명령 키를 입력

```
ubuntu editor vi study
I like
```

~



- 커서가 위치한 현재 행을 지우려면 dd 명령 키를 입력 : 행 삭제 후 윗행의 첫 컬럼으로 커서 이동

```
ubuntu editor vi study
```

~

■ 명령 취소하기

표 3-11 이전 명령 취소 명령 키

명령 키	기능
 u	명령을 취소한다.
U	해당 행에서 한 모든 명령을 취소한다.
 :e!	마지막으로 저장한 내용 이후의 것을 버리고 새로 작업한다.

- u 명령 키를 입력하면 앞의 예에서 삭제되었던 행이 복구

```
ubuntu editor vi study
```

```
I like
```

```
~
```


■ 복사하기 또는 잘라서 붙이기

표 3-12 복사하기, 잘라내기, 붙이기 명령 키

명령 키	기능
yy, #yy	커서가 위치한 행을 복사한다. #에는 복사할 행의 수를 지정한다.
p	커서가 위치한 행의 아래쪽에 붙인다.
P	커서가 위치한 행의 위쪽에 붙인다.
dd, #dd	커서가 위치한 행을 잘라둔다. 삭제와 같은 기능이다. #에는 잘라낼 행의 수를 지정한다.

■ 복사해서 붙이기 예

- 앞서 저장해둔 test.txt 파일을 vi로 열면 커서는 f에 위치

```
ubuntu editor vi study
I like
~
```

- 현재 커서가 있는 행만 복사하려면 그냥 yy 명령 키만 입력
- 1행과 2행을 함께 복사하려면 2yy를 입력
- 예: 2yy 명령 키로 두 행을 모두 복사한 다음 아랫행으로 이동하여 p 명령 키로 붙이기

```
ubuntu editor vi study
I like
ubuntu editor vi study
I like
~
```

■ 잘라내서 붙이기 예

- dd 명령 키는 삭제뿐만 아니라 잘라내기를 할 때도 사용
- 예: 현재 위치인 3행을 잘라 2행 위에 붙이기 -> dd 키를 입력한 다음 커서를 2행으로 이동하여 p 키를 입력

```
ubuntu editor vi study
ubuntu editor vi study
I like
I like
~
```

■ 마지막 행 모드에서 복사하기, 잘라내기, 붙이기

- 마지막 행 모드에서 행을 복사하거나 잘라낼 때는 범위를 지정해서 할 수 있음

표 3-13 범위 지정 명령 키

명령 키	기능
1, \$ 또는 %	1행부터 마지막 행까지 지정한다.
1, .	1행부터 커서가 있는 행까지 지정한다.
., \$	커서가 있는 행부터 마지막 행까지 지정한다.
., -3	현재 행과 이전 세 행까지(총 네 행) 지정한다.
10, 20	10행부터 20행까지 지정한다.

표 3-14 마지막 행 모드에서의 복사하기, 잘라내기, 붙이기 명령 키

명령 키	기능
:#y	#로 지정한 행을 복사한다. 예를 들면 3y는 세 행을 복사한다.
:<범위>y	범위로 지정한 행을 복사한다. 예를 들면 2,4y는 2~4행을 복사한다.
:#d	#로 지정한 행을 잘라낸다(삭제). 예를 들면 3d는 세 행을 잘라낸다.
:<범위>d	범위로 지정한 행을 잘라낸다(삭제). 예를 들면 1,4d는 1~4행을 잘라낸다.
:pu	현재 행 다음에 버퍼의 내용을 붙인다.
:#pu	#로 지정한 행 다음에 버퍼의 내용을 붙인다. 예를 들면 4pu와 같이 지정한다.

■ 검색하기

- 검색하기 위해 마지막 행으로 이동할 때는 :이 아니라 /이나 ?를 입력

표 3-15 검색 명령 키

명령 키	기능
/문자열	문자열을 아래 방향으로 검색한다.
?문자열	문자열을 위 방향으로 검색한다.
n	원래 찾던 방향으로 다음 문자열을 검색한다.
N	반대 방향으로 다음 문자열을 검색한다.

- 커서가 6행에 있을 때 검색하기 위해 /을 입력하면 마지막행으로 이동
- 검색할 문자열인 'like'를 입력하고 키를 누르면 커서 위치보다 뒤쪽에 위치한 같은 행의 'like'로 커서가 이동

```
I like
I like
ubuntu editor vi study
ubuntu editor vi study
I like
I like
~
(생략)
/like
```

- 계속 'like'를 검색하려고 n을 입력하면, 6행이 파일의 마지막 행이므로 'BOTTOM, continuing at TOP'이라는 메시지를 출력하고 1행의 'like'로 커서가 이동

■ 바꾸기

- 기존의 문자열을 다른 문자열로 바꾸려면 먼저 :을 입력하여 마지막 행 모드로 이동
- 커서 위치의 문자열만 바꿀 수도 있고, 파일 전체나 특정 범위 내에서 해당하는 문자열을 모두 바꿀 수도 있음

표 3-16 바꾸기 명령 키

명령 키	기능
:s/문자열1/문자열2/	커서가 위치한 행에서 첫 번째로 나오는 문자열1을 문자열2로 바꾼다.
:%s/문자열1/문자열2/g	파일 전체에서 모든 문자열1을 문자열2로 바꾼다.
:<범위>s/문자열1/문자열2/	범위 내 모든 각 행에서 첫 번째로 나오는 문자열1을 문자열2로 바꾼다.
:<범위>s/문자열1/문자열2/g	범위 내 모든 행에서 문자열1을 문자열2로 바꾼다.
:<범위>s/문자열1/문자열2/gc	범위 내 모든 행에서 문자열1을 문자열2로 바꿀 때 수정할지 여부를 묻는다.

■ 바꾸기 예

- 현재 커서가 'like'에 있으므로 이 단어를 'LIKE'로 바꾸기 -> :s/like/LIKE/

```
I LIKE
I like
ubuntu editor vi study
ubuntu editor vi study
I like
I like
~
(생략)
:s/like/LIKE/
```

- 범위를 지정하여 문자열 수정 -> 3행과 4행에 있는 'editor'를 'ubuntu'로 바꾸는 명령은 :3,4s/editor/ubuntu/

```
I LIKE
I like
ubuntu ubuntu vi study
ubuntu ubuntu vi study
I like
I like
~
(생략)
:3,4s/editor/ubuntu/
```

■ 바꾸기 예

- 한 행에서 해당 단어를 모두 수정하기
 - 3행에서 :s/ubuntu/UBUNTU/을 수행하고 4행에서 :s/ubuntu/UBUNTU/g를 수행하여 결과의 차이를 비교


```
I LIKE
I like
UBUNTU ubuntu vi study
UBUNTU UBUNTU vi study
I like
I like
~
(생략)
:s/ubuntu/UBUNTU/g
```

- 파일 전체를 바꿀 경우 :%s/like/LIKE/g 또는 :1,\$s/like/LIKE/g를 사용

```
I LIKE
I LIKE
UBUNTU ubuntu vi study
UBUNTU UBUNTU vi study
I LIKE
I LIKE
~
(생략)
3 substitutions on 3 lines
```

■ 파일 읽어오기, 여러 파일 편집하기

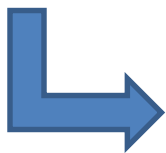
표 3-17 파일 관련 명령 키

명령 키	기능
:r 파일	지정한 파일을 읽어들이어 현재 커서 위치에 삽입한다.
:e 파일	지정한 파일로 전환한다(기존 파일을 :w로 저장한 뒤에 실행해야 한다).
:n 	vi 시작 시 여러 파일을 지정했을 경우 다음 파일로 작업을 이동한다.

■ 다른 파일 읽어오기 예

- 먼저 test.txt 파일을 열고 커서를 4행으로 이동
- :r exec2.txt를 실행하면 exec2.txt 파일의 내용이 test.txt 파일의 4행 다음에 삽입

```
I LIKE  
I LIKE  
UBUNTU ubuntu vi study  
UBUNTU UBUNTU vi study  
I LIKE  
I LIKE  
~
```



```
I LIKE  
I LIKE  
UBUNTU ubuntu vi study  
UBUNTU UBUNTU vi study  
Good afternoon everyone.  
This is a living room.  
My name is Suji Lee.  
This is a living room.  
My name is Sumi Lee.  
Good afternoon everyone.  
I LIKE  
I LIKE  
~
```

■ 파일 편집을 마치고 다른 파일 편집하기

- :e 명령 키는 현재 작업 중인 파일의 작업을 마치고 다른 파일을 편집하려고 할 때 사용
- test.txt 파일 편집을 완료하고 exec.txt 파일 편집으로 바꾸려면 :e exec.txt 사용
- 이때 작업 중이던 파일을 먼저 저장하고 :e 명령을 실행해야 함
- 파일을 저장하지 않고 :e exec.txt를 실행하면 다음과 같은 오류 메시지가 출력

```
I like UBUNTUA linux
I like UBUNTU linux
UBUNTU UBUNTU vi study
UBUNTU UBUNTU vi study
Good afternoon everyone.
This is a living room.
(생략)
~
E37: No write since last change (add ! to override)
```

■ 여러 파일 편집하기

- vi를 시작할 때 파일명을 여러 개 지정
- 파일 작업을 마치고 다음 파일로 이동하려면 :n을 입력

```
user1@myubuntu:~/linux_ex/ch3$ vi test.txt exec.txt exec2.txt
```

■ vi에서 셸 명령 사용하기

표 3-18 셸 명령 실행 명령 키

명령 키	기능
:! 셸 명령	vi 작업을 잠시 중단하고 셸 명령을 실행한다(vi로 돌아오려면 Enter 키를 누른다).
:sh	vi를 잠시 빠져나가서 셸 명령을 실행한다(vi로 돌아오려면 exit 명령을 입력한다).

■ :! 명령 키 사용하기 예

(생략)

~

~

:! ls

user1@myubuntu:~/linux_ex/ch3\$ vi test.txt

exec.txt exec2.txt test.txt

Press ENTER or type command to continue

- vi를 빠져나가거나 하는 번거로움 없이 바로 이용할 수 있다는 장점
- 다시 vi 작업으로 돌아가려면 Enter키 입력

■ :sh 명령 키 사용하기

- ':! 셸 명령'은 한 번에 하나의 셸 명령만 실행
- 실행할 셸 명령이 여러 개라면 :sh로 vi를 잠시 빠져나가서 셸 작업을 수행하고 다시 돌아오는 것이 편리

(생략)

~

~

:sh

```
user1@myubuntu:~/linux_ex/ch3$ ls
exec.txt  exec2.txt  test.txt
user1@myubuntu:~/linux_ex/ch3$ exit
```

- 다시 vi로 돌아가려면 exit를 입력

■ 기타 명령 키

표 3-19 기타 명령 키

명령 키	기능
Ctrl +l(소문자 L)	현재 화면을 다시 출력한다.
Ctrl +g	현재 커서 위치의 행 번호를 마지막 행에 출력한다.
Shift +j(대문자 J)	현재 행과 아래행을 연결하여 한 행으로 만든다.
. (마침표)	바로 직전에 했던 명령을 반복한다.


03 vi 환경 설정

■ vi의 환경 설정 방법

- 사용자 홈 디렉터리에 .exrc 파일로 저장
- 환경 변수 EXTINIT에 지정
- vi의 마지막 행 모드에서 명령으로 설정

■ vi 환경 설정 명령(set)

표 3-20 vi 환경 설정 명령

set 명령과 옵션	기능
set nu	파일 내용의 각 행에 행 번호를 표시한다(보이기만 할 뿐 저장되지는 않는다).
set nonu	행 번호를 감춘다.
set list 	눈에 보이지 않는 특수문자를 표시한다(tab:^\, eol:\$ 등).
set nolist	특수문자를 감춘다.
set showmode	현재 모드를 표시한다.
set noshowmode	현재 모드를 감춘다.
set	set으로 설정한 모든 vi 환경 설정 값을 출력한다.
set all	모든 vi 환경 변수와 현재 값을 출력한다.

■ vi 내에서 명령으로 설정하기

- 행 번호 표시하기 -> :set nu
 - 행 번호는 사용자의 편의를 위해 보이는 것으로 파일에 저장되지는 않음
 - :set nonu를 입력하면 행 번호가 없어짐

파일: exec2.txt	set nu 실행 후
Good afternoon everyone. This is a living room. My name is Suji Lee. This is a living room. My name is Sumi Lee. Good afternoon everyone. ~ :set nu	1 Good afternoon everyone. 2 This is a living room. 3 My name is Suji Lee. 4 This is a living room. 5 My name is Sumi Lee. 6 Good afternoon everyone. ~ :set nu

■ 특수문자 표시하기

- 행의 끝이나 탭 같은 특수 문자는 vi에서 보이지 않음
- 이런 특수 문자를 보려면 :set list 명령 입력
 - \$는 행의 끝, ^I(+대문자 i)는 탭을 표시

파일: exec2.txt	set list 실행 후
Good afternoon everyone,	Good afternoon everyone,\$
This is a living room,	This is a living room,\$
My name is Suji Lee,	My name is ^ISuji Lee,\$
This is a living room,	This is a living room,\$
My name is Sumi Lee,	My name is Sumi Lee,\$
Good afternoon everyone,	Good afternoon everyone,\$
~	~
:set list	:set list

03 vi 환경 설정

■ 환경 설정 값 표시하기 -> :set

```
~
:set
--- Options ---
  scroll=11          ttyfast
  fileencodings=ucs-bom,utf-8,default,latin1
  runtimepath=~/.vim,/var/lib/vim/addons,/usr/share/vim/vimfiles,/usr/share/vim/
vim80,/usr/share/vim/vimfiles/after,/var/lib/vim/addons/after,~/.vim/after
Press ENTER or type command to continue
```

03 vi 환경 설정

■ 모든 환경 변수 표시 -> :set all

```
:set all
  highlight=8:SpecialKey,~,EndOfBuffer,@:NonText,d:Directory,e:ErrorMsg,i:IncSearch,l:Search,m:MoreMsg,M:ModeMsg,n:LineNr,N:CursorLineNr,r:Question,s:StatusLine,S:StatusLineNC,c:VertSplit,t:Title,v:Visual,V:VisualNOS,w:WarningMsg,W:WildMenu,f:Folded,F:FoldColumn,A:DiffAdd,C:DiffChange,D:DiffDelete,T:DiffText,>:SignColumn,-:Conceal,B:SpellBad,P:SpellCap,R:SpellRare,L:SpellLocal,+:Pmenu,=:PmenuSel,x:PmenuSbar,X:PmenuThumb,*:TabLine,#:TabLineSel,_:TabLineFill,! :CursorColumn,:CursorLine,o:ColorColumn
  isfname=@,48-57,/.,-,_,+,,#,$,%,&,~.=
  isident=@,48-57,_,192-255
  iskeyword=@,48-57,_,
  matchpairs=(:),{:},[:]
  maxmempattern=1000
  nrformats=bin,octal,hex

packpath=~/.vim,/usr/share/vim/vimfiles,/usr/share/vim/vim80,/usr/share/vim/vimfiles/after,~/.vim/after
  paragraphs=IPLPPPQPP TPHLIPpLpItpplpipbp
  path=.,/usr/include,,

runtimepath=~/.vim,/var/lib/vim/addons,/usr/share/vim/vimfiles,/usr/share/vim/vim80,/usr/share/vim/vimfiles/after,/var/lib/vim/addons/after,~/.vim/after
  sections=SHNHH HUnhsh
  selection=inclusive
  shellredir=%s 2>&1
  suffixes=.bak,~,.,o,.h,.info,.swp,.obj
Press ENTER or type command to continue
```

03 vi 환경 설정

■ .exrc 파일에 설정하기

- 사용자 홈 디렉터리에 .exrc 파일로 저장
- 기본적으로 없는 파일이므로 사용자가 만들어야 함
- 파일에는 set 명령과 옵션만 지정
- 이 파일이 있을 경우 vi를 시작할 때마다 확인하므로 모든 파일에 동일하게 적용 가능

```
set nu  
set list  
set showmode
```

■ EXINIT 환경 변수에 설정하기

- vi 환경 설정은 다음과 같이 셸의 환경 변수인 EXINIT에도 가능

```
user1@myubuntu:~/linux_ex/ch3$ EXINIT='set nu list'  
user1@myubuntu:~/linux_ex/ch3$ export EXINIT  
user1@myubuntu:~/linux_ex/ch3$
```