vi 편집기 사용법

[1] 시작

셸상에서 vi [파일명] (vim [파일명])를 입력하면 새로운 문서를 편집할 수 있는 환경이 만들어 집니다.

예) >vi
기존의 문서를 편집하고자 할 때에는 파일의 이름을 구체적으로 명시합니다.
예) >vi file1
지정한 파일의 내용을 읽기 전용으로 열어서 볼 때는 다음과 같이 명시합니다.
예) >vi -R file

[2] vi의 세가지 모드

vi명령어는 어떻게 구성되어 있을까요?
vi명령어는 다음과 같이 입력 모드, 명령 모드, 콜론 모드(ex 모드) 크게 세가지로 분리됩니다.
① 입력 모드 - i, a, o, I, A, O를 누른 후 텍스트를 입력할 수 있는 상태
② 명령 모드(Esc모드) - ESC키를 누른 상태
③ 콜론 모드(Ex모드) - ESC키를 누르고, :(콜론)을 입력한 상태

그리고, 이러한 기본적인 모드들을 다른 편집기의 활용과 비교해보자면, 다음과 같습니다.
① 입력 모드 - 다른 편집기에서 타이핑을 하여 파일의 내용을 입력하는 과정
② 명령 모드 - 다른 편집기의 편집(Edit) 메뉴에서 제공하는 복사(Copy), 붙이기(Paste),삭제(Delete) 등의 편집 기능의 활용
③ 콜론 모드(ex 모드) - 다른 편집기의 파일(File) 메뉴에서 수행하는 열기(Open), 저장 (Save), 다른 이름으로 저장(Save as) 등의 명령 수행
vi는 실행될 때 명령 모드에서 시작하고, 실행을 종료할 때에는 콜론 모드에서 종료 명령을 수행합니다. 또한 vi는 대문자와 소문자 구분을 확실히 해두어야 한다.

(1) 입력 모드
글자를 입력하기 위해서는 입력모드로 가야하는데, 'i'를 입력하면, 하단에 --INSERT--가 나오면서, 글자를 입력할 수 있는 입력모드로 바뀝니다.
입력 모드로 들어가는 대표적인 방법은 i 키를 누르는 것이지만, 이외에도 몇 가지 방법이 더 있습니다.

- i : Insert, 현재 커서의 위치에 글자를 삽입
- I : Insert, 커서가 있는 줄(line)의 맨 앞에 글자를 삽입
- a : Append, 현재 커서 위치의 다음 칸에 글자를 추가
- A : Append, 커서가 있는 줄(line)의 맨 뒤에 글자를 추가
- o : Open line, 현재의 줄 다음에 새로운 줄을 삽입
- O: Open line, 현재의 줄 앞에 새로운 줄을 삽입

(2) 명령 모드
입력 모드에서 명령 모드로 다시 전환하려면 Esc 키를 누르기만 하면 됩니다. ESC 키를 누르고 문자를 입력하려고 하면 비프음만 날 뿐 입력은 되지 않을 것입니다.
리눅스에서는 명령모드가 따로 있습니다. 명령모드는 편집모드라고도 하는데, 글을 입력시키는 방법이 아닌 수정과 편집을 할 수 있는 상태임을 기억해 두어야 합니다. 즉, 명령모드에서는 복사,삭제, 붙이기. . . 등의 작업이 이루어 집니다. 또한, 입력모드를 통해 삽입 등 잘못된 명령을 내렸을 때 'u'라는 명령어를 통해서 Undo, 즉 복구가 가능하게 할 수 있습니다.

(3) 콜론 모드
명령모드에서 콜론 모드로 전환하려면 ':'명령을 실행시키면 됩니다. 콜론 모드는 vi에서 사용할 수 있는 명령어들을 이용하는 곳입니다. 윈도우 환경에서라면 메뉴 바와 같은 역할을 한다고 볼 수 있습니다. 파일을 저장하거나 vi를 종료하는 등의 일을 수행할 수 있습니다. 콜론 모드로 들어가려면 Esc를 누른 후 ":"를 누르면 됩니다. 콜론 모드에 들어가게 되면 화면 아래쪽에 ':' 표시가 나타나게 되며, 여기서 원하는 작업에 해당하는 명령을 입력하고 Enter 키를 치면 됩니다. 아래 그림에서 아래쪽에 ":wq"라는 표시가 눈에 뜨일 것입니다. ":" 이 상태가 바로 앞에서 말한 콜론모드의 형태입니다.

[3] 종료
종료 명령은 앞에서 잠깐 콜론 모드에서 이루어 집니다.
① 편집한 데이터를 저장하고 종료하려면, wq(Write and Quit)를 입력합니다.
예) :wq
② 저장하지 않고, 강제로 종료하려면, q!(Quit!)를 입력합니다.
예) :q!
③ vi를 끝내지 않고, 현재 작업 중인 내용을 저장만 하려면, w를 입력합니다.
예) :w

[4] vi명령어의 간단한 문법
(반복횟수)(편집명령)(반복횟수)(커서이동명령)
현재의 위치에서, 커서이동명령의 내용순서대로, 문서의 임의위치로 이동하는 순간, 편집명령어의 내용이 수행됩니다.

[5] 정규표현식의 특수 기호들
- . : 한글자를 대표하는 기호 (dos 의 ? 와 동일)
- * : 여러개의 문자를 동시에 대표하는 기호
- ^ : 줄의 처음시작
- $: 줄의 맨끝
- % : 처음줄부터 끝줄까지
- \ : 특수한 기호들이 가지는 뜻을 없앨때 사용
- \< : 단어의 시작과 대응합니다
- \> : 단어의 끝과 대응합니다
- [^] : 묶여진 문자를 제외한 아무것이나 대응합니다
- \ : 이어지는 기호를 문자 그대로 해석합니다
- [ ] : 일정한 제한을 두어 글자를 대표하고자 하는 기호
- [a-z] : a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z 를 대표
- [A-Z] : A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X,Y,Z 를 대표
- [AB] : A 또는 B
- p[aeiou]t : pat, pet, pit, pot, put 를 대표
- [0-9] : 0,1,2,3,4,5,6,7,8,9 를 대표

[6] 명령 모드(ESC 모드)에서 사용되는 몇가지 기호
- 다음과 같은 기호는 기본적인 것으로 필수적으로 알아두시기 바랍니다.
- G : 화일의 맨끝으로
- ^ : 현재줄의 맨앞 (빈칸무시)으로
- 0 : 현재줄의 맨앞으로

- $: 현재줄의 맨 뒤로
- % : 짝을 이루는 기호 확인
- w : 다음 단어로
- b : 이전 단어로
- e : 현재 단어의 끝 글자

[7] 커서 움직이기

- 명령모드 상태에서 커서를 마음대로 움직일 수 있습니다.

① 글자 단위 이동

- h : 커서를 한칸 왼쪽으로 이동하는 명령
- j : 커서를 한줄 아래로 이동하는 명령
- k : 커서를 한줄 위로 이동하는 명령
- l : 커서를 한칸 오른쪽으로 이동하는 명령
- Backspace : 커서를 한칸 왼쪽으로 이동하는 명령
- Space : 커서를 한칸 오른쪽으로 이동하는 명령

② 줄 단위 이동 - ^ : 빈칸을 무시하고, 커서를 현재 줄의 첫글자로 이동하는 명령

- 0 : 커서를 현재줄의 처음으로 이동하는 명령
- $: 커서를 현재줄의 맨끝으로 이동하는 명령
- % : 짝을 이루는 기호 확인하기
- + : 커서를 다음줄의 처음으로 이동하는 명령
- - : 커서를 이전 줄의 처음으로 이동하는 명령
- Return : 커서를 다음줄의 처음으로 이동하는 명령
- n| : 현재줄의 n 번째 열로 (n은 임의의 숫자)
- H(Home) : 커서를 화면상에 처음줄로 이동하는 명령
- M(Middle) : 커서를 화면상에 중간줄로 이동하는 명령
- L(Last) : 커서를 화면상의 마지막줄로 이동하는 명령
- nH : 화면상의 처음줄로부터 n 줄 밑으로 이동(n은 임의의 숫자)
- nL : 화면상의 마지막줄로부터 n 줄 위로(n은 임의의 숫자)
- g(Go) : 파일의 마지막 줄로 이동하는 명령
- nG : n번째 줄로 건너뛰는 명령
- gg : 맨 마지막줄로
- ngg : n 번째줄로
- n : n 번째줄로

③ 단어 단위 이동 - w(word) : 커서를 다음단어의 첫글자로 이동하는 명령

- b(back) : 커서를 이전단어의 첫글자로 이동하는 명령
- e(end) : 커서를 다음단어의 끝 글자로 이동하는 명령
- E : ?, ! 등 구두점을 무시하고, 현재 단어의 끝으로 이동
-) : 다음 문장의 시작으로 이동하는 명령
- (: 이전 문장의 시작으로 이동하는 명령
- } : 다음 문단의 시작으로 이동하는 명령
- { : 이전 문단의 시작으로 이동하는 명령
-]] : 다음 섹션의 시작으로 이동하는 명령
- [[: 이전 섹션의 시작으로 이동하는 명령

④ 화면단위 이동
- vi에는 스크롤 기능이 없는 대신 화면 단위로 이동하는 명령어들이 있습니다.
- Control + F (Forward) : 한화면 밑으로 이동
- Control + B (Backward) : 한화면 위로 이동
- Control + D (Down) : 반쪽화면 밑으로 이동
- Control + U (Upon) : 반쪽화면 위로 이동
- Control + E : 커서는 현재위치 그대로 화면만 한줄씩 위로 이동
- Control + Y : 커서는 현재위치 그대로 화면만 한줄씩 아래로 이동
- z : 커서의 위치와 함께, 화면상의 맨위로
- nz : n번 라인을 화면상의 맨위로
- z : 커서의 위치와 함께, 화면상의 중간으로
- z- : 커서의 위치와 함께, 화면상의 맨아래로
※ - Control + G : 현재 편집하고 있는 문서의 상태를 알 수있다
- Control + L : 화면 재표시 (글자가 깨졌을 경우; 윈도의 F5와 비슷)
- Control + R(edraw) : 화면 재표시 (글자가 깨졌을경우)

[8] 편집하기
① 복사, 붙이기, 합치기
- y (Yank) : 복사하기
- yy : 한 줄 전체의 내용을 복사한다
- 2yy : 두줄복사
- nyy : 현재 줄 이하로 n개의 줄을 복사한다 (n은 임의의 숫자, 2yy는 2줄 복사)
- ynw : n개의 단어를 복사한다(y2w는 두 개의 단어를 복사)
- yw : 한단어 복사
- y2w : 두단어 복사
- y$: 현재 위치에서 그 줄의 끝까지 복사한다
- y0(y^) : 현재 위치에서 그 줄의 처음까지 복사한다
- yG : 현재 위치에서 파일의 끝까지 복사한다(G는 파일의 마지막 줄)
- Y : 한 줄 전체의 내용을 복사한다 (yy 와 동일)
※ 마지막 명령어의 반복 - . : 마지막에 수행한 명령어를 반복한다.
-2. : 명령어를 2번 반복한다.
- p (Put or Paste) : 붙이기
- p : 버퍼에 저장된 내용을 커서의 오른쪽으로 붙여 넣는다.
- 2p : 아래로(오른쪽으로) 두번 붙이기
- np : n번만큼 p 명령을 반복. 2p라면 버퍼의 내용을 두 번 붙여넣는다
- P : 위로(왼쪽으로) 붙이기
- 2P : 위로(왼쪽으로) 두번 붙이기
- nP : 버퍼에 저장된 내용을 커서의 왼쪽으로 붙여넣는다. 사용법은 p와 같다
- xp : 두 문자를 바꾸는 명령
- deep : 두 단어를 바꾸는 명령
- ddp : 두 줄을 바꾸는 명령
- J(Join) : 여러 줄의 내용을 한 줄로 합친다
- J : 현재줄을 윗줄에 붙인다. (두줄 합치기)
- nJ : n개의 줄을 합쳐 한 줄로 만든다. 커서는 원본 문서의 마지막 줄의 첫번째 위치(합쳐진 줄에서는

중간)에 놓인다. 5J는 5줄로 이루어진 내용을 한 줄로 만든다

② 지우기, 복구하기, 바꾸기 - dnw : n개의 단어를 지운다
- d$: 커서가 있는 위치에서 그 줄의 끝까지 지우기
- D : 커서부터 줄의 끝까지 삭제합니다. (d$ 와 동일)
- d (Delete) : 지우기
- dd : 현재커서가 위치한 줄의 전체를 삭제합니다.
- 2dd : 두줄지우기
- ndd : n줄지우기 (n 은 임의의숫자)
- dw : 한단어 지우기
- d2w : 두단어 지우기
- d0(d^) : 그줄 처음까지 지우기
- dG : 커서가위치한 곳으로 부터 문서 끝까지 지우기
- d move : 커서가 위치한 곳부터 move까지 삭제
- dlG : 커서가 위치한 곳으로부터 편집버퍼의 맨앞까지 삭제.
- :lined : 지정한 줄을 삭제
- :line, lined : 지정한 범위를 삭제
- u (Undo) : 되살리기 명령으로 버퍼에 저장되어 있는 원래의 내용을 복구
- u : 한번복구하기
- 'nu' : (n은 임의의 숫자) 형식으로 사용하며, n 단계까지의 명령을 복구할 수 있다
- 2u : 두번복구하기

- c (Change) : 바꾸기
- cc : 한줄 바꾸기
- 2cc : 두줄 바꾸기
- ncc : n개의 줄 바꾸기(n은 임의의 숫자)
- cw : 한단어 바꾸기
- ce : 공백을 제외하고 한단어를 바꾸기
- c2w : 두단어 바꾸기
- c$: 그 줄 끝까지 바꾸기
- c0(c^) : 현재 위치에서 파일의 끝까지 바꾼다
- c : 그 줄 끝까지 바꾸기 (c$)와 동일
- cmove : 커서부터 move까지 변경
- r (Replace) : 한글자 바꾸기
- 2r : 두글자 바꾸기
※ r 명령어는 insert 모드로 바뀌지 않는다.
- R : 바꾸면서 삽입이 아닌 수정(modify) 모드로 들어간다. 윈도에서 Insert 키를 누르고 수정 상태로 들어가는 것과 같다
- s : 한글자 지우고 insert 모드로 - cl 와 동일
- S : 한줄지우고 insert 모드로 - cc 와 동일
- ~ : 대문자 < - > 소문자 바꾸기 - 영문자에만 해당
※ 지우기와 바꾸기의 차이점은 바꾸기는 명령어 후에 vi 편집모드로 바뀐다.

[9] 찾기

- /요 : 현재 위치에서 아래쪽 방향으로 '요'라는 단어를 찾는다
- ?요 : 현재 위치에서 위쪽 방향으로 '요'라는 단어를 찾는다
- / : 단어 찾기를 반복한다(웨에서 아래쪽 방향으로)
- n : 단어 찾기를 반복한다(아래쪽 방향으로)
- ? : 단어 찾기를 반복한다 (아래족에서 위쪽 방향으로)
- N : 단어 찾기를 반복한다 (위쪽 방향으로)
- fx : 현재줄에서 x문자 찾기 - x 는 한개의 글자
- Fx : 현재줄에서 반대방향으로 x문자 찾기 - x 는 한개의 글자
- tx : 현재줄에서 x문자를 찾아서 바로전에 커서놓기
- Tx : 현재줄에서 반대방향으로 x문자를 찾아서 바로후에 커서놓기
- ; : 현재 줄에서 글자 찾기를 반복한다(뒤로)
- ' : 현재 줄에서 글자 찾기를 반복한다(앞으로)

※ 찾기와 편집명령의 응용

- d/simple : simple 이라는 단어가 나올때까지 지우기
- d/^scully : 줄의 맨앞에 scully 라는 단어가 나올때까지 지우기
- y/yahoo : yahoo 라는 단어가 나올때까지 복사하기

[10] 편집모드 지정하기

- i : insert 현재커서위치
- I : 현재커서가 위치한 줄의 맨처음에
- a : append 현재커서위치 바로 다음에
- A : 현재커서가 위치한 줄의 맨끝에
- o : open 현재커서위치 바로 아래줄에
- O : Open 현재커서위치 바로 윗줄에

[11] 표시하기(Marking)

mx : mark 현재의 커서위치를 x 라는 문자로 기억 보이지 않는 북마크

- `x : 기억된 x 위치로 이동
- `` : 이동하기 전의 위치로 (제자리)
- ' ^_ x : 기억된 x 위치의 맨 앞으로 이동
- '' : 이동하기 전 위치의 맨앞으로 이동

[12] 버퍼 이용하기

- "xyy : x 라는 이름의 버퍼에 한줄 복사 하기
- "xp : x 라는 이름의 버퍼에 저장된 내용을 붙이기
- := : 현재 줄번호 보여주기
- :/pattern/ = pattern 이 위치한 줄번호 보여주기

[13] ex 명령어 익히기

vi서 사용하는 ex 명령어는 이전 ex 편집기에서 지원하던 기능들입니다. ex 모드(콜론 모드)에서 사용하는 명령어는 열기(Open), 저장(Save), 다른 이름으로 저장(Save as) 등의 명령을 수행합니다. ex 모드는 Esc 키를 누르고 ':' 키를 입력하여 들어갈 수 있다는 것과 종료시의 명령어에 대해서는 앞에서 설명하였기에 생략하기로 하겠습니다.

① ex 명령어의 기본형식

(범위지정) (명령어) (명령이 수행될 위치)
:k,l command m
예)
- :1,10 co 50 : 1 줄 부터 10 줄 까지를 50 줄 이후로 복사
- :34,50 d : 34 줄 부터 50 줄 까지 삭제
- :100,150 m 10 : 100 줄 부터 150 줄까지를 10 줄 이후로 옮김
- :.,$ d : 현재줄부터 끝까지 지우기
- :.,+20 co -4 : 현재줄부터 20줄을, 4줄 위에 복사하기
- :-,+ t 0 : 위, 아래로 한줄(총 3줄)씩을, 문서 맨위에 복사하기
- :/pattern/ d : pattern 이 들어있는 줄 지우기
- :/pattern/ -nd : pattern 이 들어있는 줄로부터 n 번째 윗줄 지우기
- :/pattern/ +nd : pattern 이 들어있는 줄로부터 n 번째 아랫줄 지우기
- :/pattern1/,/pattern2/d : pattern1 이 들어있는 줄부터, pattern2 가 들어있 는 줄까지 지우기
- :.,/pattern/ m 23 : 현재줄부터 pattern 이 들어있는 줄까지, 23번줄 이 후로 옮기기
② g 옵션 붙여 문서전체에 적용하기
- :g/리눅스 : 파일 전체에서 '리눅스'가 있는 마지막 줄로 이동한다
- :g/리눅스/ p : 파일 전체에서 '리눅스'가 있는 줄을 보여준다
- :g/리눅스/ nu : 파일 전체에서 '리눅스'가 있는 줄을 번호와 함께 보여준다
- :60,100 g/리눅스/ p : 60~100줄 사이에서 '리눅스'가 있는 줄을 보여준다
- :g/리눅스/d : 문서 전체에서 '리눅스'가 있는 줄을 제거한다
③ 저장 및 종료하기
- :w : 저장하기 (write)
- :q : 종료하기 (quit)
- :wq : 저장하고 종료하기
- :xv저장하고 종료하기 (:wq 와 동일)
- :w! : 강제로 저장하기 (read-only 로 열었을경우)
- :q! : 편집한 내용을 저장하지 않고 종료하기
- :w new_file_name : 새로운 파일이름으로 저장하기
- :w %.new : 현재화일 이름에 .new 를 붙여서 새로운 화일로 저장
- :230,$ w file_name : 230 줄부터 끝줄까지 file_name 으로 저장하기
- :.,600 w file_name : 현재줄부터 600줄까지 file_name 으로 저장하기
- :1,10 w new_file : 1줄부터 10줄까지 new_file 로 저장하기
- :340,$ w >> new_file : 340줄부터 끝줄까지 new_file 에 추가하기
④ 읽기
- :r[ead] filename : 현재위치에 filename 읽어들이기
- :r /usr/local/data : 현재위치에 /usr/local/data 읽어들이기
- :185 r /usr/local/data : 185줄 이후에 /usr/local/data 읽어들이기
- :$r /usr/local/data : 맨끝줄 이후에 /usr/local/data 읽어들이기
- :0 r /usr/local/data : 맨윗줄에 /usr/local/data 읽어들이기
- :/pattern/ r /usr/local/data : pattern 이 존재하는 줄에 /usr/local/data 읽어들이기
⑤ 다중편집하기
vi file1 file2 :file1과 file2라는 두 개의 문서를 읽어들인다. 먼저 읽는 문서는 file1이다
- :args :현재 편집중인 화일목록을 보여준다
v - :n :두 번째 문서(file2)를 편집할 수 있다
- :e # :이후부터는 ':e #' 명령을 사용하여 문서를 번갈아 열며 편집할 수 있다

- :prev[ious] :이전화일로 돌아간다
주의 - 편집중인 화일이 저장되지 않으면 다음 화일로 넘어갈수 없다
* * 새로운 파일 편집하기
- :e file1:새로운 파일(file1)을 읽어들여 편집한다
- e[dit] file_name :새로운 file_name 편집하기
- :e! :현재의 편집중인 내용을 무시하고 가장 최근에 편집한 내용을 다시편집하기
⑥ g 옵션과 바꾸기
- :m,n s/old/new/g - :g/pattern/ s/old/new/g
예)
: 1,5 s/리눅스/linuk/gc :1줄부터 5 줄까지 리눅스를 linuk 로 확인해가면서 바꾼다.
:g/문자 /s/파일/file/g : '문자'가 있는줄만을 찾아서 '파일'을 'file'로 바꾼다.
:% s/버퍼/buffer/g :처음줄부터 마지막줄까지, 버퍼 를 buffer 로 바꾼다.
* * 프로그램 소스 코드에서 괄호 짝 찾기
프로그램이나 HTML 소스에는 수많은 괄호(< >, ,…)가 사용되게 마련입니다. 때문에 편집을 하다 보면 괄호의 짝을 잃어버려 프로그램 오류가 나는 경우도 빈번합니다. 이럴 때 vi는 편리한 기능을 제공합니다. 해당 괄호 위에 커서를 놓은 후 키를 누르면 짝이 되는 괄호의 위치로 이동합니다.

[14] 예기치 않은 시스템 다운후 되살리기
- vi -r :되살릴수 있는 모든 파일의 이름을 보여준다.
- vi -r file :vi에디터를 실행하여 지정한 파일을 되살리는 옵션이다

[15] 디스플레이 제어
^L : 현재의 화면을 다시 불러온다
:set number : 내부의 줄번호로 내용을 불러온다
:set nonumber : 내부의 줄번호로 내용을 되 불러 오지 않게 하는 옵션이다
예) set number 실행시 다음과 같이 줄번호가 나타납니다
- /rexp : 지정된 정규표현식에 대해 앞으로 이동한다.
- / : 이전의 패턴에 대해 앞으로 검색을 반복하는 명령
- ?rexp : 지정된 정규 표현식에 대해 뒤로 이동하는 명령
- ? : 이전의 패턴에 대해 뒤로 검색을 반복하는 명령
- n : : /나 ? 명령에 대해 같은 방향으로 반복하는 명령
- N : : /나 ? 명령에 대해 반대방향으로 반복하는 명령

[16] 줄의 길이조정
- r :Return : 문자를 새로운 라인으로 변경하는 명령
- J :줄을 결합하는 명령
- :set wm=n : 오른쪽으로 n문자 위치에서 자동적으로 줄을 나누는 명령

[17] 편집버퍼를 통한 이동
- ^F : 한 화면 아래로 이동
- ^B : 한화면 위로 이동
- n^F : n 화면 아래로 이동
- n^B : n화면 위로 이동
- ^D : 반화면 아래로 이동
- ^U : 반화면 위로 이동

- n^D : n 줄만큼 아래로 이동
- n^U : n 줄만큼 위로 이동

[18] 쉘 명령실행

중단하고 지정한 셸 명령을 실행
- :!! : vi를 중단하고 이전의 셸 명령을 실행
- :sh : vi를 중단하고 셸을 실행
- :!csh : vi를 중단하고 새로운 C쉘을 실행

[19] 패턴에 의한 치환
- :s/pattern/replace/ : 현재 줄을 치환한다
- :lines/pattern/replace/ : 지정한 줄을 치환한다
- :line,lines/pattern/replace/ : 지정한 범위를 모두 치환한다
- :%s/pattern/replace/ : 모든 줄을 치환한다

[20]데이터 읽기
- :liner file : 파일의 내용을 지정한 줄 다음에 삽입한다
- :r file : 파일의 내용을 현재줄 다음에 삽입한다
- :liner !command : 명령의 결과를 지정한 줄 다음에 삽입한다
- :r !command : 명령의 결과를 현재줄 다음에 삽입한다
- :r !look pattern : 지정한 패턴으로 시작되는 단어를 삽입한다

[21] 편집 중 파일 바꾸기
- :e file : 지정한 파일을 편집한다
- :e! file : 지정한 파일을 편집하며 자동점검을 생략한다

[22] 수정의 취소, 반복
- u : 수정했던 마지막 명령을 취소한다
- U : 현재 줄을 저장한다
- . : 수정했던 마지막 명령을 반복한다

[23] 문자 삭제
- xv : 커서가 있는 문자를 삭제한다
- X : 커서의 왼쪽 문자를 삭제한다

[24]여러줄의 복사및 이동
- : linecotarget : 지정한 줄을 복사하여 대상줄 밑에 삽입한다
- :line,linecotarget : 지정한 범위를 복사하여 대상줄 밑에 삽입한다
- :linemtarget : 지정한 줄로 이동하여 대상줄 밑에 삽입한다
- :line,linemtarget : 지정한 범위로 이동하여 대상줄 밑에 삽입한다

[25] 데이터 처리를 위한 셸 명령
- n!!command : n번 줄에서 명령을 실행한다
- !move command : 커서의 위치로 부터 이동한 곳까지 명령을 실행한다
- !move fmt : 커서의 위치로 부터 이동한곳까지 줄들을 형식에 맞춘다.