ENTERPRISE LINUX ADMIN GUIDE

VI & VIM

단원목표

- VI 실행
- VI 의 세가지 모드 변경 방법
- 입력모드
- 명령모드
- 최하위행 모드

■ 현재 많이 사용되는 편집기 종류

- vi 편집기(Visual Editor)
  vi 편집기는 유닉스 계열에서 가장 많이 사용되는 편집기이다. 1976년 빌 조이(Bill Joy)가 개발하였다. vi 편집기는 한 화면을 편집하는 비쥬얼 에디터(Visual Editor)이다.

- vim(vi improved)
  브람 무레나르(Bram Moolenaar)가 vi 편집기와 호환되면서 독자적으로 다양한 기능 추가하여 만든 편집기이다. 편집시에 다양한 색상을 이용하여 가시성을 높였으며, 패턴 검색시에 하이라이트 기능을 제공하여 빠른 검색이 가능하게 해준다. (http://www.vim.org)

- emacs 편집기(Editor Macros)
  리처드 스톨만이 개발한 고성능 문서 편집기이다. 매크로 기능이 있는 텍스트 교정 및 편집기이다. 초기 리처드 스톨만에 의해 개발 되었고, 이후에 제임스 고슬링(James Gosling)이 LISP(LISt Processor) 언어에 의한 환경 설정 및 에디터 기능을 확장시킬 수 있는 기능을 포함하여 '고슬링 이맥스'라는 이름으로 배포하였다.

- pico(Pine Composer)
  워싱턴 대학의 Aboil Kasar가 개발한 유닉스 기반의 텍스트 에디터로 pine 이메일 클라이언트 프로그램에 통합되어서 배포되었다. pico의 기본 인터페이스는 윈도우의 메모장(Notepad)와 유사하여 매우 단순한다.

- nano
  pico의 복제 버전이다. nano는 pico의 기능 이외에도 마우스 지원, 자동 들여쓰기, 정규 표현식 검색, 구문 강조 등의 기능을 추가로 제공한다.

VI 실행

```
# vi

# vi filename

# vi -r filename
# vi -L

# vi +38 filename
# vi +/"word" filename
```

1 VI(Visual Editor) 편집기 특징

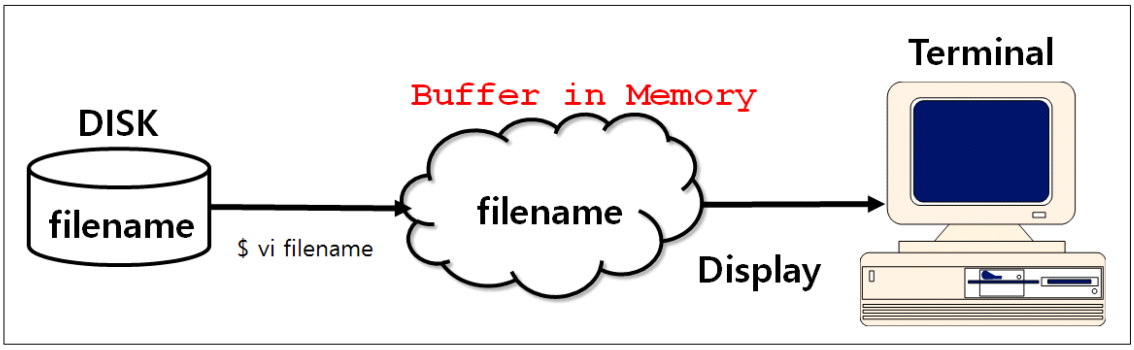
- VI(VIsual editor) -> /bin/vi
- VIM(Visual editor IMproved) -> /usr/bin/vim

```
# ls -l /bin/vi
-rwxr-xr-x 1 root root 581K Jan  9  2013 /bin/vi*
# ls -l /usr/bin/vim
-rwxr-xr-x 1 root root 2.7M Jan  9  2013 /usr/bin/vim*


# vi ~/.bashrc    (# gedit ~/.bashrc)
alias vi='/usr/bin/vim'
# . ~/.bashrc     (# source ~/.bashrc)
```

vi('브이아이'로 부른다)는 Emacs와 함께 Unix 환경에서 가장 많이 쓰이는 문서 편집기이다. 1976년 빌 조이가 초기 BSD 릴리즈에 포함될 편집기로 만들었다. vi라는 이름은 한 줄씩 편집하는 줄단위 편집기가 아니라 한 화면을 편집하는 비주얼 에디터(visual editor)라는 뜻에서 유래했다. 간결하면서도, 강력한 기능으로 열광적인 사용자가 많다.

현재는 오리지널 vi를 사용하는 경우는 거의 없고, 일반적으로 기능을 모방하여 만들어진 클론을 사용하고 있다. 이런 클론 중 많이 쓰이는 것은 기능이 다양한 것을 장점으로 내세우며, 리눅스 배포판에 포함되는 Vim, 그리고, BSD 라이선스로 제공되며 원본 vi의 동작과 호환성으로 정평이 나 있는 nvi, 독자적인 팬층을 확보한 elvis등이 있다.

VI의 특징



vi 편집기가 동작하는 원리를 보면 다음과 같이 버퍼에서 작업을 하게 된다.
그러므로 저장을 시키는 명령어를 입력하지 않는 이상 디스크상에 파일의
내용으로 저장되는 것은 아니다.

명령어 형식	설명
$ vi	새 파일을 편집할 수 있는 화면이 나온다. 여기서 작업을 한후에는 반드시 파일 이름을 지정하여 저장하여 주어야 한다. 예) : w filename (최하위행 모드)
$ vi filename	filename 이름을 가진 파일이 존재하는 경우 해당 파일을 편집하고 filename 이름을 가진 파일이 존재하지 않는 경우 새 파일을 편집할 수 있는 화면 상태가 된다.
$ vi -R filename $ view filename	Readonly, 파일을 Readonly 상태로 열어준다. 중요한 파일, 여러 사람이 동시에 수정가능한 파일을 다룰때 편리하게 사용될수 있다. view 명령어와 동일한 기능을 수행할 수 있다.
$ vi -r filename	Recovery, 이전 vi 편집 작업 중 비정상적으로 작업이 끝난 경우 편집하던 파일 복구시에 사용된다. 이 경우 사용자의 메일로 복구할 파일에 대한 정보가 오게 된다.
$ vi -L	이전 vi 편집 작업 중 비정상적으로 작업이 끝난 경우 복구할 파일들에 대한 전체적인 목록을 볼수 있다.

$ vi +38 filename
$ vi -c 38 filename
$ vi +/"school" filename
	편집작업에 들어갈 때 특별한 명령어를 수행하면서 시작하는 경우 사용한다.
	예) vi +38 filename(38번째 라인부터 시작)
	vi -c 38 filename(위와 같은 의미)
	vi +/"school" filename (school 단어가 있는 라인부터 시작)

```
# vi -L                    /* fedora 사용자가 작업중 파일이 비정상 종료 (작업중 shutdown) */
Swap files found:
   In current directory:
      -- none --
   In directory ~/tmp:
      -- none --
   In directory /var/tmp:
1.    file1.swp
          owned by: fedora   dated: Thu Feb 11 09:47:26 2010
        file name: /test/file1
         modified: YES
        user name: fedora   host name: linux200
      process ID: 24710
   In directory /tmp:
      -- none --
```

```
# ls -l /var/tmp           /* 비 정상 종료된 파일의 목록이 보임 */
total 12
-rw------- 1 fedora fedora 12288 Feb 11 09:47 file1.swp
```

```
# vi -r /var/tmp/file1.swp
linux
```

```
# vi +/"fedora" /etc/passwd
```

```
# vi +30 /etc/passwd
# cat > /test/file1
Linux
Vi Editor
<Ctrl + D>

# su - fedora              /* 사용자 전환 */
$ vi -R /test/file1                /*
Linux
Vi Editor
[본인 이름 입력]
:wq!                       /* 저장 안됨 q! 로 빠져나옴 */
```

(기본 작업 테스트)

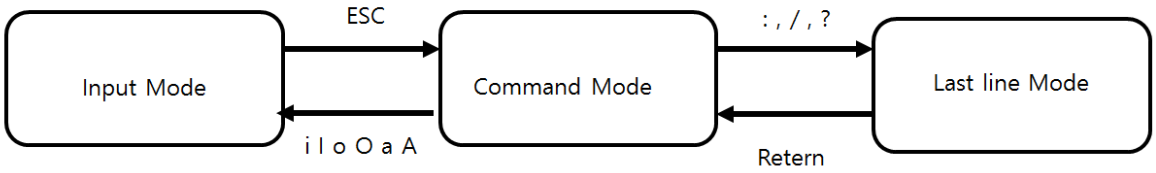
```
# cd /test
# vi filename
```

```
Input : k, j

Input : i

File Contents
============================
hello
hello linux
Welcome To My Server!!!
hello linux
soldesk linux CentOS
Have a Good Time !!

============================
```

```
2 line : linux -> liNux

move -> delete -> insert -> save & quit

(1). move
h, j, k, l
← ↓ ↑ →

(2). delete
x, dd

(3). input
i

(4). save & quit
shift + :

: w!
: wq      /* w: write, q: quit */
: q!
: wq!

(5). verification
[TERM1]
# vi filename

[TERM2]
# cd /test
# cp /etc/services /test
# vi services
```

```
: set number   (: set nu) 파일의 LINE번호를 붙여 표시해준다
: set nonumber (: set nonu)

: 30    (30G)    nG       특정 LINE으로 이동한다
/ftp -> n -> N
```

VI 의 세가지 모드 변경 방법



- **Insert**
  - i : 현재 커서 위치로 부터 입력
  - I : 현재 커서가 있는 행의 처음부터 입력
- **Append**
  - a : 현재 커서 위치 이후부터 입력
  - A : 현재 커커가 있는 행의 마지막 부분부터 입력
- **Open Line**
  - o : 현재 커서가 있는 아래행부터 입력
  - O : 현재 커서가 있는 위행부터 입력

vi 편집기에서는 3가지 모드가 지원된다. 명령행 모드(Comand Mode). 입력행 모드(Input Mode), 최하위행(라인) 모드(Last Line Mode)가 지원이 된다. 명령행 모드는 편집작업 및 기타 명령어를 입력할 수 있는 모드이고 입력행 모드는 입력만 할 수 있는 모드이다. 최하위행(라인) 모드는 저장 및 기타 검색 작업등을 할 수 있는 모드이다.

■ Command Mode(Edit Mode) 키 입력을 명령어로 해석
■ Input Mode(Insert Mode) 키 입력을 파일에 입력
■ Last Line Mode(Ex Mode) ex 명령어를 수행

(1) 입력행 모드(Input Mode)

vi 편집기 실행시 기본 모드는 명령행 모드이다. 명령행 모드에서 입력을 하기 위해서는 입력행 모드로 전환해야 한다. 이런 경우 다음과 같은 문자를 통해 입력 모드로 전환할 수 있다.
• a, A, i, I, o, O

▶ Insert
- i : 현재 커서 위치로부터 입력한다.
- I : 현재 커서가 있는 행의 처음부터 입력한다.

▶ Append
- a : 현재 커서 위치 이후부터 입력한다.
- A : 현재 커서가 있는 행의 마지막부터 입력한다.

▶ Open Line
- o : 현재 커서가 있는 아래행부터 입력한다.
- O : 현재 커서가 있는 위행부터 입력한다.

명령모드

- 이동(move)
  좌우 이동 : (h, l) , (w, b) , (0(^), $)
  상하 이동 : (j, k), (<CTRL + F> , <CTRL + B>), (G, 1G)
- 삭제(delete)
  좌우 삭제 : x, dw , (d0, d$), dd
  상하 삭제 : dd, 3dd, (:1,3d), (dG, d1G)
- 복사/붙이기(copy/paste)
  복사 : yy , Y
  붙이기 : p (:1,3 co 5), (: 1,3 m 5)
- 검색(search)
  /word, n or N
  ?wordm n or N

(2) 명령행 모드(Command Mode)

ⓐ 이동(Move)

■ **좌우 이동** : (h, l), (w, b), (0(^), $)
■ **상하 이동** : (j, k), (<CTRL + F>, <CTRL + B>), (H, L), (G, nG, 1G)

[참고] :5 (=5G), :10 (=10G), 5, 10

- h : 한 문자 왼쪽으로 이동(←)
- l : 한 문자 오른쪽으로 이동(→)
- w : 한 단어 오른쪽으로 이동
- b : 한 단어 왼쪽으로 이동

- 0(^) : 라인의 처음 문자(라인의 처음)으로 이동
- $: 라인의 마지막으로 이동

- j : 한 문자 아래로 이동(↓), 다음 라인으로 이동
- k : 한 문자 위로 이동(↑), 이전 라인으로 이동

- <CTRL + F> : 다음 페이지로 이동
- <CTRL + B> : 이전 페이지로 이동

- H : 현재 페이지의 가장 첫번째 줄로 이동
- L : 현재 페이지의 가장 마지막 줄로 이동

- 1G : 문서의 첫 번째 라인으로 이동 (gg)
- G : 문서의 마지막 라인으로 이동

ⓑ 삭제(Delete)

■ 좌우 삭제: x, dw, (d0, d$), dd
■ 상하 삭제: dd, 3dd, (:1,3d), (dG, d1G)

[참고] dd(=D), 3dd(=3D)

- x : 현재 커서 한 글자 삭제
- dw : 현재 커서 앞의 한 단어 삭제
- db : 현재 커서 뒤로 한 단어 삭제
- d0/d^ : 현재 커서부터 라인의 처음까지 삭제
- d$: 현재 커서부터 라인의 마지막까지 삭제
- dd : 현재 라인 삭제

- 3dd : 현재 커서 라인을 포함해서 아래로 3개 라인 삭제
- :1,3d : 1번째 라인부터 3번째 라인까지 삭제

- dG : 현재 커서 라인부터 문서 마지막까지 삭제
- d1G : 현재 커서 라인부터 문서 처음까지 삭제

ⓒ Undo
■ u, U

- u : 바로 이전에 상태로 되돌림
- U : 라인전체에 대해 이전 상태로 되돌림

ⓓ Join Line
■ J

- J : 현재라인에 아래 라인 붙이기

ⓔ Replace
■ r, R

- r : 현재 글자를 대치
- R : <ESC>키를 누르기 전까지 연속으로 현재 글자 대치

(3) 최하위행(라인) 모드(Last Line Mode)

ⓐ 복사/붙이기(Copy/Yank & Paste)

■ **yy(=Y), 3yy(3Y), p or P**
■ **:1,3 co 5**
■ **:1,3 m 5**

- yy : 현재 라인 복사(Yank)
- 3yy : 현재 커서 라인 포함해서 하위의 3개의 라인 복사
- :1,3 co 5 : 첫 번째 라인부터 3번째 라인까지 복사하여 5번째 라인 아래에 붙이기
- :1,3 m 5 : 첫 번째 라인부터 3번째 라인까지 5번째 라인 아래에 이동하기

- p : 현재 커서 아래에 붙이기
- P : 현재 커서 위에 붙이기

ⓑ 검색(Search)

■ **/New, n** or **N**
■ **?New n** or **N**

- /New : 현재 커서 라인부터 찾을려는 문자열(예: New) 검색
- ?New : 문서의 마지막 라인부터 찾을려는 문자열(예: New) 검색

- n : n(Next), 정방향다음번째 검색
- N : N(Next), 역방향으로 다음번째 검색

/ 정방향 검색
? 역방향 검색

치환작업

- 검색 바꾸기
  : %s/<검색문자열>/<치환문자열>/g

  : %s/hello/HELLO/g
  : 1,$s/hello/HELLO/g
  : 5,10s/HELLO/hello/g

  : 5,10s/&/ /
  : 5,10s/^   //

ⓒ 검색/바꾸기(Search & Replace)

■ :%s/<찾을문자열>/<바꿀문자열>/g

■ :5,10s/Old/New/g

[참고] "<찾을 문자열>"에는 "정규 표현식"을 사용할 수 있다.

EX) :%s/hello/HELLO/g
  1,$ => :5,10s/HELLO/hello/g
  search(substitution)
  globally => %s/hello/HELLO/ , %s/hello/HELLO/g

```
EX) :5,10s/^/#/                          /* 주석처리 */
    :5,10s/^#//                          /* 주석해제 */

    :5,10s/^/    / (4 blank character)   /* 들여쓰기 */
    :5,10s/^    //                       /* 내어쓰기 */
```

- :%s/hello/HELLO/g : 문서 전체에서 hello를 검색해서 HELLO로 변환
- :1,$s/hello/HELLO/g : 문서 전체에서 hello를 검색해서 HELLO로 변환
- :5,10s/^/#/ : 5번째 라인부터 10번째까지 라인의 처음부분에 '#' 처리

■ 특수문자를 변경하는 방법
  특수문자는 문자 앞에 역슬래쉬 기호를 추가해야 한다
  /를 $ 로 변경
  :%s/W//W$/g

최하위행 모드

- 저장 & 종료

```
:w
:w filename
:w!

:q
:q!

:wq
:wq!

:!cmd
```

ⓓ Save & Quit

```
■ :w             /* w(write), 현재 파일에 저장 하기 */
■ :w filename    /* 다른 이름으로 저장 하기 */
■ :w!(root use)  /* 현재 파일에 강제적으로 저장 하기 */
■ :w! file       /* 현재까지의 변경사항을 file로 저장 */
■ :3,10w file    /* 3번째 라인부터 10번째 라인까지 file로 저장 */
■ :q             /* q(quit), 편집기 종료 */
■ :q!            /* 저장 안하고 편집기 종료 */
■ :wq            /* 저장하고 편집기 종료 */
■ :wq!(root Use) /* 현재 파일에 강제적으로 저장하고 편집기 종료 */
■ :r file        /* file의 내용을 현재 커서 위치에서 읽어 들임 */
■ :!CMD          /* vi 편집기를 빠져나가지 않은 상태에서 쉘 명령어를 수행 */
```

[참고] 저장하고 빠져나가기(Save & Quit)의 여러가지 방법

```
■ :x
■ :wq
■ ZZ
```

[참고] vi 편집기 실행에 대해서

```
[TERM1] # vi filename
         : wq!
        # cat filename
[TERM2] # vi services
         : q!
        # cat services
```

vi 편집기 환경파일

vi 편집기의 동작하는 기능을 변경하기 위해서는 set 명령어를 사용한다. set 명령어 다음에 all을 사용하면 현재 편집기에 사용가능한 모든 기능변수들에 대한 현재 설정값을 표시한다.

vi 편집기의 기능을 현재 실행되는 편집 화면에서만 변경하기 위해서는 최하위행 모드에서 다음과 같은 방법을 사용한다.

```
# cd /test
# vi filename
```

```
: set all       (Last Line Mode)
: set number    (: set nu)
: set nonumber  (: set nonu)
--- Options ---
  ambiwidth=single  nohidden            nopreserveindent      termencoding=
noautoindent          history=50          prompt              noterse
noautoread          nohlsearch          noreadonly            textauto
noautowrite         noignorecase          remap               notextmode
noautowriteall        iminsert=0          report=2              textwidth=0
  background=light    imsearch=0          scroll=23           notildeop
nobackup            noincsearch           scrolljump=1          timeout
  backupcopy=auto   noinfercase           scrolloff=0           timeoutlen=1000
  backupext=~       noinsertmode        nosecure              nottimeout
  backupskip=/tmp/*   isprint=@,161-255    shell=/bin/bash       ttimeoutlen=-1
nobinary              joinspaces          shellcmdflag=-c       ttybuiltin
nobomb                keywordprg=man      shellquote=           ttyfast
  buflisted         nolazyredraw          shelltemp             ttyscroll=999
  cmdheight=1         lines=48            shellxquote=          ttytype=xterm
  columns=80        nolist              noshiftround          undolevels=1000
nocompatible          listchars=eol:$     shiftwidth=8          updatecount=200
nocopyindent          loadplugins         noshortname           updatetime=4000
  cpoptions=aABceFs   magic               noshowfulltag         verbose=0
  debug=              matchtime=5         noshowmatch           verbosefile=
nodelcombine          maxcombine=2          showmode            novisualbell
  display=            maxmapdepth=1000      sidescroll=0          warn
noedcompatible        maxmem=257666         sidescrolloff=0     noweirdinvert
  encoding=utf-8      maxmemtot=257666    nosmartcase             whichwrap=b,s
  endofline         nomodeline          nosmarttab              wildchar=<Tab>
  equalalways         modelines=5           softtabstop=0         wildcharm=0
  equalprg=           modifiable            startofline         wildignore=
noerrorbells          modified              swapfile            wildmode=full
  esckeys             more                  swapsync=fsync        window=47
noexpandtab           mouse=                switchbuf=            wrap
noexrc                mousemodel=extend     tabstop=8             wrapmargin=0
  fileencoding=       mousetime=500         tagbsearch            wrapscan
  fileformat=unix   nonumber              taglength=0           write
  formatoptions=tcq   operatorfunc=       tagrelative         nowriteany
..... (중략) .....
```

```
: set tabstop=10          /* 탭간격 조정 */
: set nu                  /* 라인 번호 달기 */
: set noshowmode          /* 상태표시행에 모드를 표시하거나 표시하지 않거나를 설정 */
: set directory=/tmp      /* 지정된 기능 변경 */
```

[참고] VIM 편집기의 환경 설정 파일

■ 사용자 정의 $HOME/.vimrc 파일 생성

```
# vi ~/.vimrc   (# vi ~/.exrc)
```

```
set number
```

```
# vi /etc/passwd
```

```
1 root:x:0:0:root:/root:/bin/bash
2 bin:x:1:1:bin:/bin:/sbin/nologin
3 daemon:x:2:2:daemon:/sbin:/sbin/nologin
4 adm:x:3:4:adm:/var/adm:/sbin/nologin
```

■ $HOME/.vimrc 파일의 샘플 파일 생성

```
# cp /usr/share/vim/vim70/vimrc_example.vim ~/.vimrc
```

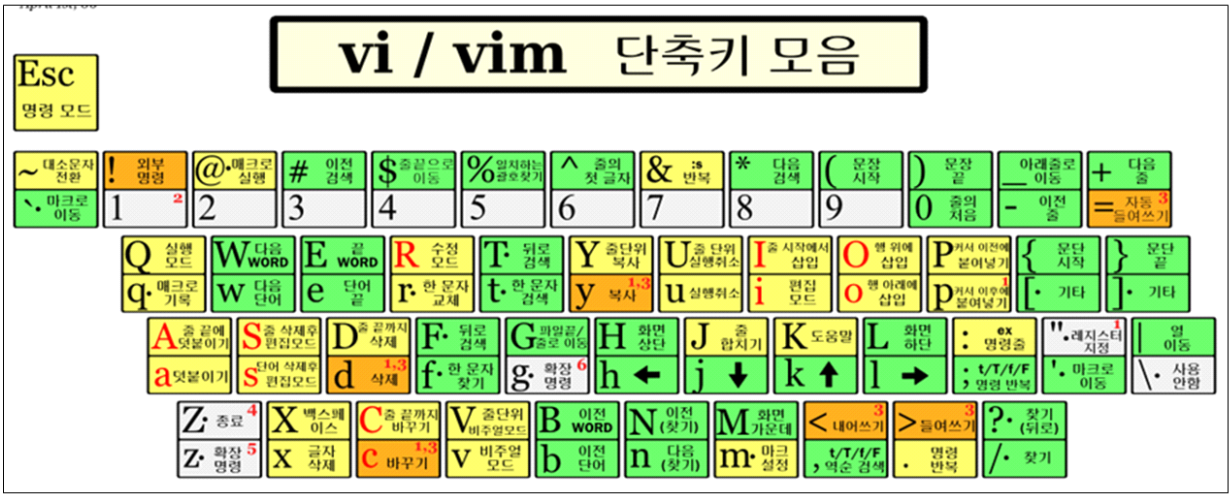
■ VIM 편집기 참고 파일(VIM 편집기 사용법 문서)

```
# vi /usr/share/vim/vim70/tutor/tutor.ko.ufs-8
```

■ VIM 편집기의 Plugin 모음들

http://saelly.tistory.com/296
http://ethanschoonover.com/solarized

VI 단축키



커서의 이동	
h	왼쪽으로 이동
j	아래로 이동
k	위로 이동
l	오른쪽으로 이동
w	한단어 오른쪽으로이동
b	한단어 왼쪽으로 이동
Return	한행아래로 이동
Backspace	한문자 왼쪽으로 이동
Spacebar	한문자 오른쪽으로 이동
H	화면의 맨위로 이동
M	화면의 중간으로 이동
L	화면의 맨 아래로 이동
^F	한화면 앞으로 이동
^D	반화면 앞으로 이동
^B	한화면 뒤로 이동
^U	반화면 뒤로 이동
삽입명령어	
a	**커서 오른쪽에 문자삽입**
A	**커서 오른쪽, 행의끝에 문자삽입**
i	**커서 왼쪽에 문자삽입**
I	**커서왼쪽, 행의 처음에 문자삽입**
o	**커서 아래에 행 삽입**
O	**커서 위에 행 삽입**
<Esc>	작업완료후 반드시 입력
텍스트 변경	
cw	단어 변경
cc	행 변경
C	커서 오른쪽의 행 변경

s	커서가 위치한 문자열 대체
r	커서 위치 문자를 다른 문자로 대체
r-Return	행 분리
J	현재 행과 아래 행 결합
xp	커서 위치 문자와 오른쪽 문자교환
~	문자형(대,소문자) 변경
u	이전 명령 되돌리기
U	행 변경 사항 취소
:u	이전의 최종행 취소
<Ctrl + r>	이전 명령 다시 실행
텍스트 삭제	
x	문자삭제
dw	단어삭제
dd	행 삭제
D	커서 오른쪽 행 삭제
:5,10 d	5-10째 행 삭제
행번호 설정	
:set nu	행번호 표시
:set nonu	행번호 숨기기
행 찾기	
G	파일의 마지막 행으로 가기
12G	파일의 12번째 행으로 가기 (==> :12 동일 한 뜻)
텍스트의 복사 및 이동	
yy	행 yank 또는 복사
Y	행 yank 또는 복사
P	yank 되거나 삭제된 행을 현재 행 위에 삽입
p	yank 되거나 삭제된 행을 현재 행 아래에 삽입
:1,2 co 3	1-2행을 3행 다음으로 복사
:4,5 m 6	4-5행을 6행 다음으로 이동
탐색 및 대체	
/string/	string탐색
?string?	string 역 방향 탐색
n(N)	string의 다음(이전) 계속 탐색
:g/search-string/s//replace-string/gc	각 발생 탐색후 확인하고 대체
:s/str/rep/	현재 행의 str을 rep로 대체
:1,.s/str/rep/	1부터 현재 행의 str을 rep로 전부 대체
:%s/str/rep/g	파일 전체 str을 rep로 전부 대체
화면정리	
:r filename	커서 다음에 파일 삽입
:20 r filename	파일을 20번째 행 다음에 삽입
파일의 저장 및 종료	
:w	변경사항 저장
:w filename	지정한 파일로 저장
:wq	변경사항 저장후 vi종료
:ZZ	변경사항 저장후 vi종료
:q!	변경사항을 저장하지 않고 vi종료 (! 강제의 의미)
:w %.new	현재파일 이름에 .new 를 붙여서 새로운 파일로 저장
:230,$ w filename ant	230 줄부터 끝줄까지 filename으로 저장하기

:.,580 w filename	현재줄부터 580줄까지 filename으로 저장하기
:1,10 w new_filename	1줄부터 10줄까지 new_filename으로 저장하기
:340,$ w >>new_file	340줄부터 끝줄까지 new_file에 추가하기
k,l command m	
: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
:/p1/, /p2/ d	p1 이 들어있는 줄부터, p2 가 들어있는 줄까지 지우기
:.,/pa/ m 23	현재줄부터 pa 이 들어있는 줄까지, 23번줄 이후로 옮기기
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
:/pa/r/usr/local/data	pa 이 존재하는 줄에 /usr/local/data 읽어들이기
다중편집하기	
vi file1 file2 file3 :args	편집중인 파일목록 보여주기
:n[ext]	다음 파일로 넘어가기
:prev[ious]	이전파일로 돌아가기
sc/ESC/g	BX가 있는줄 찾아서 Esc 를 ESC 로 바꾸
:g/editer/ s//editor/g	위와 동일("s/" 다음에 인자가 없어서 윗줄과 같은효과
undo	
u	작업을 취소한다(undo)
U	그 줄에 행해진 작업 모두 취소
.	조금 전에 했던 명령을 반복

[정규문자표]

^	시작하는
$	끝나는
.	.만큼 문자를 포함하는 패턴 (어떤 문자이던 상관 없이)예) r..t => root
*****	모든 예) [a-z]* => 소문자가 인 것
[ ]	패턴이 있으면
[^]	^다음에오는 문자를 제외하고 다음 패턴 예)[^a-m]pattern =>a부터 m까지 제외한 패턴이 있으면