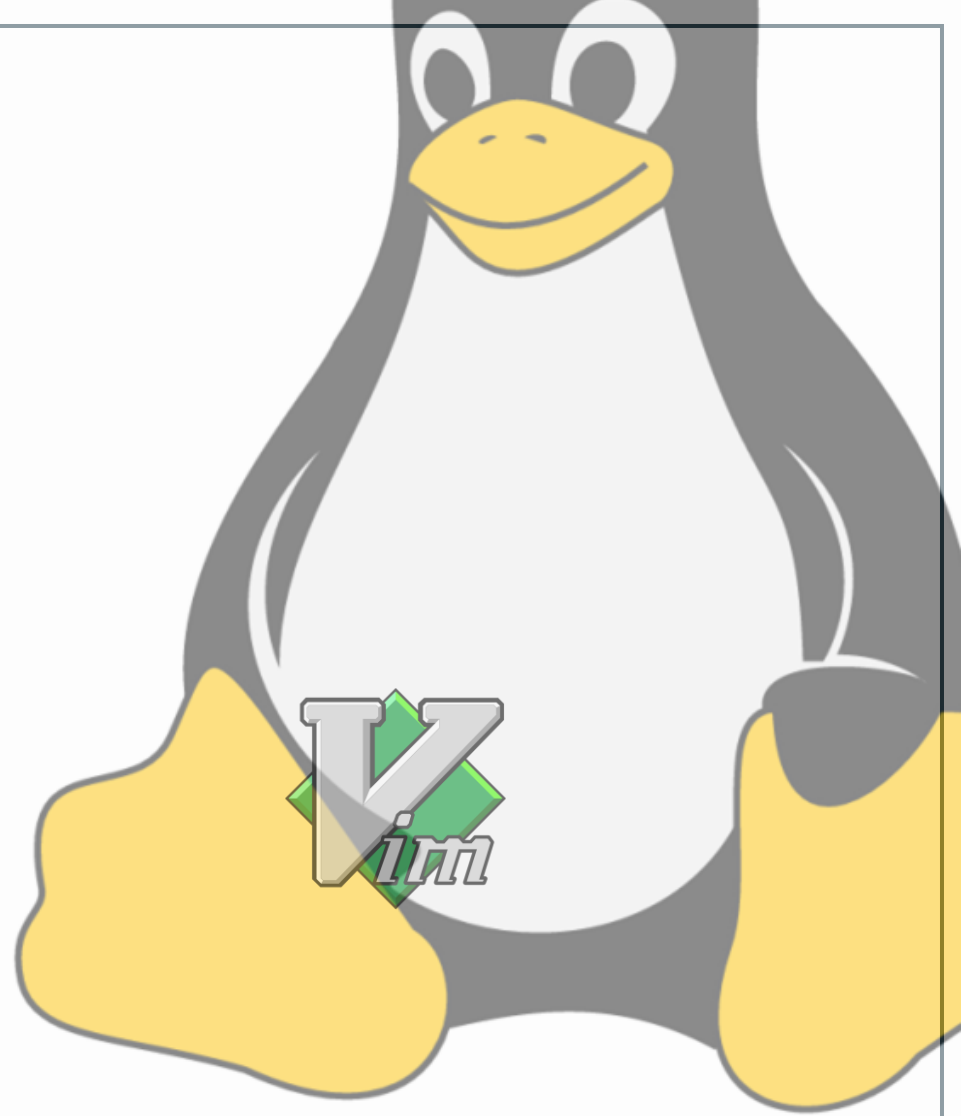





# EDA Tool [Cadence / Synopsys] 및 Linux 소프트웨어 개발자를 위한 Vi 사용자 간단 메뉴얼



# Vi Editor 관련 교육

## 초보자를 위한 온라인 웹사이트 및 튜토리얼

### 1. [openvim.com](https://openvim.com)

- 인터랙티브한 웹 기반 Vim 튜토리얼
- 기본 커맨드를 직접 타이핑하면서 학습 가능
-  강력 추천

### 2. [Vim Adventures](#)

- 게임 형태로 Vim 키맵을 익힐 수 있는 사이트
- 첫 단계는 무료, 이후 유료
- 재밌게 익히고 싶은 분에게 적합

### 3. [\[Vim Tutor Mode in Vim itself\]](#)

- 터미널에 `vimtutor` 입력하면 나오는 공식 튜토리얼
- 시스템에 Vim이 설치되어 있다면 누구나 접근 가능
- 실습 중심으로 매우 유익

### 4. [Learn Vim Progressively](#)

- GitHub 기반의 매우 잘 정리된 입문서 (영문)
- 커맨드별 예시, 실습 방법, 학습 루트 제공

## 문서 / 블로그 튜토리얼

### • Vim 기초 정리 블로그

- <https://riptutorial.com/ko/vim>
- 각 커맨드와 용례를 정리한 간단 명료한 튜토리얼

### • 효율적인 Vim 사용법 요약 (영문)

- <https://danielmiessler.com/study/vim/>
- 자주 쓰는 명령어 위주로 정리됨

## 추천 학습 순서

- vimtutor로 기초 실습
- openvim.com으로 반복 연습
- Vim Adventures로 키맵 감각 익히기
- GitHub 튜토리얼 문서 읽기
- 실제 코드/문서 편집에 Vim 사용 적용

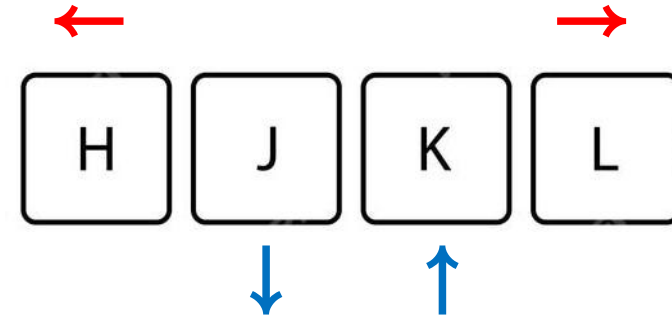
# Contents



- Lesson 1.1: 커서 움직이기
- Lesson 1.2: Vim을 시작하고 끝내기
- Lesson 1.3: 텍스트 편집 - 지우기
- Lesson 1.4: 텍스트 편집 - 삽입 (INSERTION)
- Lesson 1.5: 텍스트 편집 - 추가 (APPENDING)
- Lesson 1.6: 파일 편집
- LESSON 1 요약
- Lesson 2.1: 삭제(DELETION) 명령
- Lesson 2.2: 다른 삭제 명령
- Lesson 2.3: 명령과 적용 대상에 대해
- Lesson 2.4: 대상에 반복 적용하기
- Lesson 2.5: 삭제에 반복 적용하기
- Lesson 2.6: 줄 전체 조작하기
- Lesson 2.7: 취소(UNDO) 명령
- LESSON 2 요약
- Lesson 3.1: 붙이기(PUT) 명령
- Lesson 3.2: 치환(REPLACE) 명령
- Lesson 3.3: 변환(CHANGE) 명령
- Lesson 3.4: c 를 이용한 다른 변환 명령
- LESSON 3 요약
- Lesson 4.1: 위치와 파일의 상태
- Lesson 4.2: 찾기 명령
- Lesson 4.3: 괄호의 짝 찾기
- Lesson 4.4: 치환(SUBSTITUTE) 명령
- LESSON 4 요약
- Lesson 5.1: 외부 명령 실행하는 방법
- Lesson 5.2: 보다 자세한 파일 저장
- Lesson 5.3: 선택적으로 저장하는 명령
- Lesson 5.4: 파일 읽어들이기, 합치기
- LESSON 5 요약
- Lesson 6.1: 새 줄 열기(OPEN) 명령
- Lesson 6.2: 추가(APPEND) 명령
- Lesson 6.3: 치환(REPLACE) 의 다른 버전
- Lesson 6.4: 문자 복사 붙여넣기(COPY AND PASTE)
- Lesson 6.5: 옵션 설정(SET)
- LESSON 6 요약
- LESSON 7.1: 온라인 도움말 명령
- LESSON 7.2: 시작 스크립트 만들기
- Lesson 7.3: 명령어 완성하기
- Lesson 7 요약

# Lesson 1.1: 커서 움직이기

- \*\* 커서를 움직이려면, 표시된 대로 h,j,k,l 키를 누르십시오. \*\*
- ^
- k      힌트: h 키는 왼쪽에 있으며, 왼쪽으로 움직입니다.
- < h      l >      l 키는 오른쪽에 있으며, 오른쪽으로 움직입니다.
- j      j 키는 아래방향 화살표처럼 생겼습니다.
- v



- 1. 익숙해질 때까지 커서를 스크린 상에서 움직여 보십시오.
- 2. 아래 방향키 (j)를 반복입력이 될 때까지 누르고 계십시오.
- 이제 다음 lesson으로 가는 방법을 알게 되었습니다.
- 3. 아래 방향키를 이용하여, Lesson 1.2 로 가십시오.
- 참고: 원하지 않는 무언가가 입력이 되었다면, <ESC>를 눌러서, 명령 모드로 돌아가십시오. 그 후에 원하는 명령을 다시 입력하십시오.
- 참고: 커서키 또한 작동할 것입니다. 하지만 hjkl에 익숙해지면, 커서키보다 훨씬 빠르게 이동할 수 있을 것입니다.

## Lesson 1.2: vim을 시작하고 끝내기

- !! 주의: 아래 있는 단계를 실행하기 전에, 이 lesson 전체를 읽으십시오!!
- 1. <ESC> 키를 눌러서 확실하게 명령 모드로 빠져 나옵니다.
- 2. 다음과 같이 입력합니다: `:q! <ENTER>`
- 이렇게 하면, 바뀐 내용을 \*저장하지 않고\* 편집기를 빠져나갑니다.
- 3. 쉘 프롬프트가 보인다면, 다시 길잡이로 돌아오기 위해 다음과 같이 입력합니다.
- `vimtutor <ENTER>`
- 또는 다음과 같을 수도 있습니다.
- `vim tutor.ko <ENTER>`
- ---> 'vim' 은 vim 편집기로 들어가는 것을 뜻하며, 'tutor.ko'는 편집하려는 파일을 뜻합니다.
- 4. 위에서 이야기한 단계를 기억하였으며, 확신이 서면, 1에서 3까지를 수행하여 편집기를 나갔다가 다시 들어와 보십시오.
- 주의: `:q! <ENTER>` 는 바뀐 내용을 저장하지 않습니다. 이 후 lesson에서 어떻게 편집 내용을 저장하는지 배울 수 있습니다.
- 5. 그 후 커서를 아래로 움직여 Lesson 1.3 으로 가십시오.

## Lesson 1.3: 텍스트 편집 - 지우기

- **\*\* 명령 모드에서 x 를 누르면 커서가 위치한 곳의 글자를 지울 수 있습니다. \*\***
- 1. ----> 로 표시된 곳으로 커서를 옮겨보십시오.
- 2. 오타를 수정하기 위해, 커서를 지울 글자 위로 움직여 보십시오.
- 3. x 키를 눌러서 지워야할 글자를 지우십시오.
- 4. 2에서 4까지를 반복하여 문장이 올바르게 되도록 하여 보십시오.
- ---> The ccow jumpedd ovverr thhe mooon.
- 5. 문장이 정확해졌다면, Lesson 1.4로 가십시오.
- 주의: 이 길잡이를 보면서 외우려고 하지말고, 직접 사용해보면서 익히길 바랍니다.

## Lesson 1.4: 텍스트 편집 - 삽입 (INSERTION)

- **\*\* 명령 모드에서 i 를 누르면 텍스트를 입력할 수 있습니다. \*\***
- 1. 커서를 첫번째 ---> 로 표시된 줄로 움직입니다.
- 2. 첫번째 줄을 두번째 줄과 똑같이 만들것입니다. 텍스트가 들어가야할  
•     곳 다음부터 첫번째 글자 위에 커서를 옮겨 놓습니다.
- 3. i 키를 누른 후, 필요한 내용을 입력합니다.
- 4. 수정한 후에는 <ESC> 를 눌러서 명령 모드로 돌아갑니다.
- 문장을 올바르게 만들기 위해 2에서 4의 과정을 반복합니다.
- ---> There is text misng this .
- ---> There is some text missing from this line.
- 5. 텍스트를 삽입하는 데에 익숙해졌다면, Lesson 1.5로 가십시오.

## Lesson 1.5: 텍스트 편집 - 추가 (APPENDING)

- **\*\* A 를 입력해 텍스트를 추가할 수 있습니다. \*\***
- 1. 커서를 첫번째 ---> 로 표시된 줄로 움직입니다.
- 커서가 문장 내 어디에 있던 상관없습니다.
- 2. A 키를 눌러 필요한 내용을 입력합니다.
- 3. 내용을 모두 입력한 후 <ESC>를 눌러 명령 모드로 돌아갑니다.
- 4. 커서를 두번째 ---> 로 표시된 줄로 움직입니다.
- 문장을 올바르게 만들기 위해 2에서 3의 과정을 반복합니다.
- ---> There is some text missing from th
- There is some text missing from this line.
- ---> There is also some text miss
- There is also some text missing here.
- 
- 5. 텍스트를 추가하는 데 익숙해졌다면, Lesson 1.6으로 가십시오.



## Lesson 1.6: 파일 편집

- **\*\* :wq 를 이용하여 파일을 저장하고 빠져나갈 수 있습니다. \*\***
- **!! 주의: 아래 있는 단계를 실행하기 전에, 이 lesson 전체를 읽으십시오!!**
- 1. lesson 1.2에서 배웠던 것처럼 :q!로 편집기를 나갈 수 있습니다.
- 만약, 다른 터미널에 접근 가능하다면, 아래의 단계를 다른 터미널에서 해봅니다.
- 2. 쉘 프롬프트에 다음과 같이 입력합니다: vim tutor <ENTER>
- 'vim' 은 vim 에디터 시작을 위한 명령어, 'tutor'는 수정하고자 하는
- 파일의 이름 입니다.
- 3. 앞에서 배웠던 것처럼 텍스트를 삽입하고 지워보세요.
- 4. 다음 명령어를 이용해 파일 수정 부분을 저장하고 빠져나갑니다: :wq <ENTER>
- 5. 만약 1에서 vimtutor를 빠져나갔다가 다시 들어왔다면, 아래로 움직여 요약으로 넘어가도록 합니다.
- 
- 6. 위 모든 단계를 다 읽고 이해한 후에 직접 해보세요.

# LESSON 1 요약

- 1. 커서를 움직일 때에는 화살표 키나 hjkl 키를 이용합니다.
- h (왼쪽)       j (아래)       k (위)       l (오른쪽)
  
- 2. 쉘 프롬프트에서 vim 시작하려면 vim FILENAME <ENTER>
  
- 3. 수정한 내용을 무시한 채로 vim에서 빠져나가려면 <ESC> :q! <ENTER>
- 저장한 후 vim에서 빠져나가려면 <ESC> :wq <ENTER>
  
- 4. 명령 모드에서 커서가 위치한 곳의 글자를 지우려면 x 를 입력합니다.
  
- 5. 명령 모드에서 커서가 위치한 곳에 텍스트를 삽입하려면
- i 를 누른 후 텍스트를 입력하고 <ESC>       커서 앞에 삽입합니다.
- A 를 누른 후 텍스트를 입력하고 <ESC>       문장 뒤에 추가 합니다.
  
- 참고: <ESC>는 명령 모드로 돌아가는 데 쓰며, 원치 않는 명령이나 완전히 입력되지
- 않은 명령을 취소하는 데에도 씁니다.
  
- 그럼 Lesson 2를 시작합시다.

## Lesson 2.1: 삭제(DELETION) 명령

- **\*\* 한 단어를 끝까지 지우려면 dw 라고 치면 됩니다. \*\***
- 1. <ESC> 키를 눌러서 확실하게 명령 모드로 빠져 나옵니다.
- 2. 아래에 ---> 로 표시된 줄 까지 커서를 옮깁니다.
- 3. 지워야할 단어의 처음으로 커서를 옮깁니다.
- 4. dw 라고 쳐서 그 단어를 지웁니다.
- 주의: 위에서 말한대로 하면 화면의 마지막 줄에 dw 라는 글자가 표시됩니다.
- 잘못 쳤다면, <ESC> 를 눌러서 다시 시작하십시오.
- ---> There are a some words fun that don't belong paper in this sentence.
- 5. 3, 4번 과정을 다시 하여 문장을 정확하게 만든 뒤 Lesson 2.2로 가십시오.

## Lesson 2.2: 다른 삭제 명령

- **\*\* d\$ 라고 치면 그 줄 끝까지 지워집니다. \*\***
- 1. <ESC> 키를 눌러서 확실하게 명령 모드로 빠져 나옵니다.
- 2. 아래에 ---> 로 표시된 줄 까지 커서를 옮깁니다.
- 3. 올바른 줄의 끝으로 커서를 옮깁니다. (첫번째로 나오는 . 다음입니다.)
- 4. d\$ 라고 쳐서 줄 끝까지 지웁니다.
- ---> Somebody typed the end of this line twice. end of this line twice.
- 5. 어떤 일이 일어났는지 이해하기 위해 Lesson 2.3 으로 가십시오.

## Lesson 2.3: 명령과 적용 대상에 대해

- 삭제 명령 d의 형식은 다음과 같습니다.
- d 대상
- 여기서:
  - d - 지우는 명령
  - 대상 - 아래에 제시된 대상에 대해 명령을 수행
- 적용 가능한 대상의 종류:
  - w - 커서에서 그 단어의 끝까지 (공백 포함.)
  - e - 커서에서 그 단어의 끝까지 (공백을 포함하지 않음.)
  - \$ - 커서에서 그 줄의 끝까지
- 예를 들어, de 는 커서의 위치부터 해당 단어의 끝까지 지웁니다.
- 참고: 호기심이 있다면, 명령 모드에서 명령 없이 대상을 입력해보십시오.
- 위에서 이야기한 대상의 목록에 따라 커서가 움직이게 됩니다.

## Lesson 2.4: 대상에 반복 적용하기

- **\*\* 대상 이전에 숫자를 넣어주면 그 만큼 반복 됩니다. \*\***
- 1. 아래에 ---> 로 표시된 줄 까지 커서를 옮깁니다.
- 2. 2w 입력하여 커서를 단어 두 개 뒤로 옮깁니다.
- 3. 3e 입력하여 커서를 뒤로 세 번째 단어의 끝으로 옮깁니다.
- 4. 0 (zero) 를 입력하여 문장의 시작부분으로 움직입니다.
- 5. 2에서 3까지를 다른 숫자로 반복해 봅니다.
- ---> This is just a line with words you can move around in.
- 6. Lesson 2.5로 가십시오.

## Lesson 2.5: 삭제에 반복 적용하기

- **\*\* 명령과 숫자를 함께 사용하면 그만큼 반복 수행 됩니다. \*\***
- 위에서 삭제 명령과 대상의 조합과 같이, 대상 이전에 횟수를 넣어 더 많이 삭제 할 수 있습니다:
- d   횟수   대상
- 1. 아래 ---> 표시된 줄에서 커서를 첫번째 대문자 단어로 옮깁니다.
- 2. d2w를 입력하여 두 대문자 단어를 지웁니다.
- 3. 이어지는 대문자 단어들을 1에서 2까지의 단계를 이용해 횟수를 바꾸어 삭제해 봅니다.
- ---> this ABC DE line FGHI JK LMN OP of words is Q RS TUV cleaned up.

## Lesson 2.6: 줄 전체 조작하기

- **\*\* dd 라고 치면 줄 전체를 지웁니다. \*\***
- 줄 전체를 지우는 일이 잦기 때문에, Vi를 디자인 한 사람들은, 간단히 d를
- 두번 연달아 치면 한 줄을 지울 수 있도록 하였습니다.
- 1. 커서를 아래 나온 단락의 두번째 줄로 가져가십시오.
- 2. dd 를 입력하여 그 줄을 지우십시오.
- 3. 그런 다음 네번째 줄로 가십시오.
- 4. 2dd 라고 입력하여 두줄을 지웁니다. ( 횃수-명령-대상을 기억하세요. )
- ---> 1) Roses are red,
- ---> 2) Mud is fun,
- ---> 3) Violets are blue,
- ---> 4) I have a car,
- ---> 5) Clocks tell time,
- ---> 6) Sugar is sweet
- ---> 7) And so are you.



## Lesson 2.7: 취소(UNDO) 명령

- **\*\* u 를 누르면 마지막 명령이 취소되며, U 는 줄 전체를 수정합니다. \*\***
- 1. 커서를 ---> 로 표시된 줄로 이동한 후 첫번째 잘못된 부분 위로 옮깁니다.
- 2. x 를 입력하여 첫번째 잘못된 글자를 지웁니다.
- 3. 그럼 이제 u 를 입력하여 마지막으로 수행된 명령을 취소합니다.
- 4. 이번에는 x 명령을 이용하여 그 줄의 모든 에러를 수정해봅시다.
- 5. 대문자 U 를 눌러서 그 줄을 원래 상태로 돌려놓아 보십시오.
- 6. 이번에는 u 를 몇 번 눌러서 U 와 이전 명령을 취소해봅시다.
- 7. CTRL-R (CTRL 키를 누른 상태에서 R을 누르는 것) 을 몇 번 눌러서 명령을 다시 실행해봅시다. (취소한 것을 취소함.)
- ---> Fiix the errors oon thhis line and reeplace them witth undo.
- 8. 이 명령은 매우 유용합니다. 그럼 Lesson 2 요약으로 넘어가도록 합시다.

## LESSON 2 요약

- 1. 커서가 위치한 곳부터 단어의 끝까지 지우려면: `dw`
- 2. 커서가 위치한 곳부터 줄 끝까지 지우려면: `d$`
- 3. 줄 전체를 지우려면: `dd`
- 4. 횟수와 함께 대상을 반복 시키려면: `2w`
- 5. 명령 모드에서 내리는 명령의 형식은 다음과 같습니다:
  - `[횟수] 명령 대상` 또는 `명령 [횟수] 대상`
  - 여기서:
    - 횟수 - 그 명령을 몇 번 반복할 것인가
    - 명령 - 어떤 명령을 내릴 것인가 ( 예를 들어, 삭제인 경우는 `d` )
    - 대상 - 명령이 동작할 대상, 예를 들어 `w` (단어), `$` (줄의 끝) 등.
- 6. 커서를 문장 맨 앞으로 옮기려면: `0`
- 7. 이전 행동을 취소하려면: `u` (소문자 `u`)
- 한 줄에서 수정한 것을 모두 취소하려면: `U` (대문자 `U`)
- 취소한 것을 다시 실행하려면: `CTRL-R`

## Lesson 3.1: 붙이기(PUT) 명령

- **\*\* p** 를 입력하여 마지막으로 지운 내용을 커서 뒤에 붙입니다. **\*\***
- 1. 아래에 있는 문단의 첫 줄로 커서를 움직이십시오.
- 2. **dd** 를 입력하여 그 줄을 지워서 빔의 버퍼에 저장합니다.
- 3. 아까 지운 줄이 가야할 위치의 **\*윗줄로\*** 커서를 옮깁니다.
- 4. 명령 모드에서, **p** 를 입력하여 그 줄을 제대로 된 자리로 옮깁니다.
- 5. 2에서 4를 반복하여 모든 줄의 순서를 바로 잡으십시오.
- d) Can you learn too?
- b) Violets are blue,
- c) Intelligence is learned,
- a) Roses are red,

## Lesson 3.2: 치환(REPLACE) 명령

- **\*\* 커서 아래의 글자 하나를 바꾸려면, r 을 누른 후 바꿀 글자를 입력합니다. \*\***
- 1. 커서를 ---> 로 표시된 첫 줄로 옮깁니다.
- 2. 커서를 잘못된 첫 부분으로 옮깁니다.
- 3. r 을 누른 후, 잘못된 부분을 고쳐 쓸 글자를 입력합니다.
- 4. 2에서 3의 과정을 반복하여, 첫 줄의 오류를 수정하십시오.
- ---> Whan this lime was tuoed in, someone presswd some wrojg keys!
- ---> When this line was typed in, someone pressed some wrong keys!
- 5. Lesson 3.2 로 이동합니다.
- 주의: 외우지 말고, 직접 해보면서 익혀야 한다는 것을 잊지 마십시오.

## Lesson 3.3: 변환(CHANGE) 명령

- **\*\* 한 단어의 전체를 바꾸려면, ce 를 치십시오. \*\***
- 1. 커서를 ---> 로 표시된 첫줄로 옮깁니다.
- 2. 커서를 lubw 에서 u 위에 올려놓습니다.
- 3. ce 라고 명령한 후 단어를 정확하게 수정합니다. (이 경우, 'ine' 를 칩니다.)
- 4. <ESC> 를 누른 후 다음 예러로 갑니다 (수정되어야할 첫 글자로 갑니다.)
- 5. 3에서 4의 과정을 반복하여 첫번째 문장을 두번째 문장과 같도록 만듭니다.
- ---> This lubw has a few wptfd that mrrf changing usf the change command.
- ---> This line has a few words that need changing using the change command.
- ce 는 단어를 치환하는 것 뿐만 아니라, 내용을 삽입할 수 있도록 한다는 것에 유의하십시오.

## Lesson 3.4: c 를 이용한 다른 변환 명령

- **\*\* 변환 명령은 삭제할 때 이용한 대상에 대해 적용할 수 있습니다. \*\***
- 1. 변환 명령은 삭제와 동일한 방식으로 동작합니다. 형식은 다음과 같습니다:
  - [횟수] c 대상 또는 c [횟수] 대상
- 2. 적용 가능한 대상 역시 같습니다. w (단어), \$ (줄의 끝) 등이 있습니다.
- 3. ---> 로 표시된 첫줄로 이동합니다.
- 4. 첫 에러 위로 커서를 옮깁니다.
- 5. c\$ 를 입력하여, 그 줄의 나머지가 두번째 줄처럼 되도록 수정한 후 <ESC> 를 누르십시오.
- ---> The end of this line needs some help to make it like the second.
- ---> The end of this line needs to be corrected using the c\$ command.
- 참고: 입력하는 동안은 백스페이스를 이용할 수 있습니다.

## LESSON 3 요약

- 1. 이미 지운 내용을 되돌리려면, p 를 누르십시오. 이 명령은 커서 \*다음에\*
- 지워진 내용을 붙입니다(PUT). (한 줄을 지운 경우에는 커서 다음 줄에
- 지워진 내용이 붙습니다.)
- 2. 커서 아래의 글자를 치환하려면(REPLACE), r 을 누른 후 원래 글자 대신
- 바꾸어 넣을 글자를 입력합니다.
- 3. 변환 명령(CHANGE)은 커서에서 부터 지정한 대상의 끝까지 바꿀 수 있는
- 명령입니다. 예를 들어, 커서 위치에서 단어의 끝까지 바꾸려면 ce 를
- 입력하면 되며, c\$ 는 줄 끝까지 바꾸는 데 쓰입니다.
- 4. 변환 명령의 형식은 다음과 같습니다:
- [횟수] c 대상 또는 c [횟수] 대상
- 계속해서 다음 Lesson 을 진행합니다.

# Lesson 4.1: 위치와 파일의 상태

- \*\* CTRL-G 를 누르면 파일 내에서의 현재 위치와 파일의 상태를 볼 수 있습니다.
- G 를 누르면 파일 내의 마지막 줄로 이동합니다. \*\*
- 주의: 아래의 단계를 따라하기 전에, 이 Lesson 전체를 먼저 읽으십시오.
- 1. CTRL 키를 누른 상태에서 g 를 누릅니다. 파일 이름과 현재 위치한 줄이 표시된 상태줄이 화면 아래에 표시될 것입니다. 3번째 단계를 위해 그 줄 번호를 기억하고 계십시오.
- 참고: 커서가 화면 오른쪽 하단으로 옮겨진 것을 보인다면, 이는 'ruler' 옵션을 세팅된 경우입니다. (:help 'ruler' 를 참고 하세요.)
- 2. G 를 누르면 파일의 마지막으로 이동합니다.
- gg 를 누르면 파일의 시작 부분으로 이동합니다.
- 3. 아까 기억했던 줄 번호를 입력한 후 G 를 누르십시오. 이렇게 하면 처음에 CTRL-G 를 눌렀던 장소로 되돌아가게 될 것입니다.
- (번호를 입력할 때, 이것은 화면에 표시되지 않습니다.)
- 4. 자신이 생겼다면, 1에서 3까지를 실행해보십시오.



## Lesson 4.2: 찾기 명령

- **\*\* / 를 누른 후 검색할 문구를 입력하십시오. \*\***
- 1. 명령 모드에서 / 를 입력하십시오. : 명령에서와 마찬가지로, 화면 아래에  
• / 와 커서가 표시될 것입니다.
- 2. 'errroor' 라고 친 후 <ENTER> 를 치십시오. 이 단어를 찾으려고 합니다.
- 3. 같은 문구를 다시 찾으려면, 간단히 n 을 입력하십시오.  
• 같은 문구를 반대 방향으로 찾으려면, Shift-N 을 입력하십시오.
- 4. 문구를 역방향으로 찾으려면, / 대신 ? 를 이용하면 됩니다.
- 5. 원래 있던 곳으로 돌아가기 위해서는 CTRL-O 를 이용하면 됩니다. 반복하면 더 이전으로도  
• 갈 수 있습니다. CTRL-I 로 다시 뒤로 갈 수도 있습니다.
- ---> "errroor" is not the way to spell error; errroor is an error.
- 참고: 찾는 중에 파일의 끝에 다다르게 되면, 파일의 처음부터 다시 찾게 됩니다.

## Lesson 4.3: 괄호의 짝 찾기

- **\*\* %** 를 눌러서 **)**, **]**, **}** 의 짝을 찾습니다. **\*\***
- 1. 커서를 **--->** 로 표시된 줄의 **(**, **[**, **{** 중 하나에 가져다 놓습니다.
- 2. **%** 를 입력해 봅시다.
- 3. 커서가 짝이 맞는 괄호로 이동할 것입니다.
- 4. **%** 를 입력하여, 이전 괄호로 되돌아 옵시다.
- 5. 커서를 다른 **(**), **[**], **{** 혹은 **}** 로 움직여 **%** 를 입력해 봅니다.
- **---> This ( is a test line with ('s, ['s ] and {'s } in it. ))**
- 참고: 짝이 맞지 않는 괄호가 있는 프로그램을 디버깅할 때에 매우 유용합니다!

## Lesson 4.4: 치환(SUBSTITUTE) 명령

- **\*\* :s/old/new/g** 하면 'old' 를 'new' 로 치환(SUBSTITUTE)합니다. **\*\***
- 1. 커서를 ---> 로 표시된 줄에 가져다 놓습니다.
- 2. :s/thee/the 를 입력한 후 <ENTER> 를 칩니다. 이 명령은 그 줄에서 처음으로 발견된 것만 바꾼다는 것에 주의하십시오.
- 3. 이번에는 :s/thee/the/g 를 입력합니다. 이는 그 줄 전체(globally)를 치환한다는 것을 의미합니다.
- ---> thee best time to see thee flowers is in thee spring.
- 4. 두 줄 사이의 모든 문자열에 대해 치환하려면 다음과 같이 합니다,
  - :#,#s/old/new/g #,# 는 두 줄의 줄번호를 뜻합니다.
  - :%s/old/new/g 파일 전체에서 발견된 모든 것을 치환하는 경우입니다.
  - :%s/old/new/gc 파일 전체에서 발견된 모든 것을 찾고, 치환할지 안 할지 프롬프트로 명령합니다.

## LESSON 4 요약

- 1. CTRL-G 파일의 상태와 파일 내에서의 현재 위치를 표시합니다.
- G 파일의 끝으로 이동합니다.
- 숫자 G 해당 줄로 이동합니다.
- gg 첫 번째 라인으로 이동합니다.
- 2. / 를 입력한 후 문구를 입력하면 그 문구를 아랫방향으로 찾습니다.
- ? 를 입력한 후 문구를 입력하면 윗방향으로 찾습니다.
- 검색 후, n 을 입력하면 같은 방향으로 다음 문구를 찾으며,
- Shift-N 을 입력하면 반대 방향으로 찾습니다.
- CTRL-O 는 과거의 위치로, CTRL-I는 새로운 위치로 옮겨줍니다.
- 3. 커서가 (, ), [, ], {, } 위에 있을 때에 % 를 입력하면 상응하는 짝을  
찾아갑니다.
- 4. 어떤 줄에 처음 등장하는 old를 new로 바꾸려면       :s/old/new
- 한 줄에 등장하는 모든 old를 new로 바꾸려면       :s/old/new/g
- 두 줄 #,# 사이에서 치환을 하려면       : #, #s/old/new/g
- 파일 내의 모든 문구를 치환하려면       :%s/old/new/g
- 바꿀 때마다 확인을 거치려면 'c'를 붙여서       :%s/old/new/gc

## Lesson 5.1: 외부 명령 실행하는 방법

- **\*\*** **!** 을 입력한 후 실행하려는 명령을 입력하십시오. **\*\***
- 1. 친숙한 명령인 **:** 를 입력하면 커서가 화면 아래로 이동합니다. 명령을 입력할 수 있게 됩니다.
- 2. 이제 **!** (느낌표) 를 입력하십시오. 이렇게 하면 외부 셸 명령을 실행할 수 있습니다.
- 3. 시험삼아 **!** 다음에 **ls** 를 입력한 후 **<ENTER>** 를 쳐보십시오. 셸 프롬프트에서처럼 디렉토리의 목록이 출력될 것입니다. **ls** 가 동작하지 않는다면 **!:dir** 을 시도해 보십시오.
- 참고: 어떤 외부 명령도 이 방법으로 실행할 수 있습니다.
- 참고: 모든 **:** 명령은 **<ENTER>** 를 쳐야 마무리 됩니다.

## Lesson 5.2: 보다 자세한 파일 저장

- **\*\* 수정된 내용을 파일로 저장하려면, :w FILENAME 하십시오. \*\***
- 1. `!:dir` 또는 `!:ls` 를 입력하여 디렉토리의 리스트를 얻어옵니다.
- 위의 명령 후 <ENTER>를 쳐야한다는 것은 이미 알고 있을 것입니다.
- 2. TEST 처럼 존재하지 않는 파일 이름을 하나 고르십시오.
- 3. 이제 `:w TEST` 라고 입력하십시오. (TEST는 당신이 선택한 파일 이름입니다.)
- 4. 이렇게 하면 vim 길잡이 파일 전체를 TEST라는 이름으로 저장합니다.
- 확인하려면, `!:dir` 을 다시 입력하여, 디렉토리를 살펴보십시오.
- 참고: vim을 종료한 후, vim을 다시 실행하여 TEST라는 파일을 열면, 그 파일은 저장했을 때와 완벽히 같은 복사본일 것입니다.
- 5. 이제 그 파일을 지웁시다.
- (MS-DOS에서): `!del TEST`
- (Unix에서): `!rm TEST`

## Lesson 5.3: 선택적으로 저장하는 명령

- \*\* 파일의 일부를 저장하려면, v 대상 :w FILENAME 을 입력합니다. \*\*
- 1. 이 줄로 커서를 가져옵니다.
- 2. v 를 누르고 커서를 아래 다섯번째로 옮깁니다. 이 때, 문자열들이 하이라이트 됨을 주목합니다.
- 3. : 를 누릅니다. 화면 하단에 : '<,>' 가 나타납니다.
- 
- 4. w TEST 를 입력합니다. 여기서 TEST는 파일 이름이며 아직 생성되어 있지 않습니다. <ENTER>를 누르기 전, : '<,>' w TEST 로 입력되었는지 확인 합니다.
- 
- 5. 빔은 선택된 문장들을 TEST 파일에 입력합니다. :!dir 혹은 :!ls를 이용하여 파일이 만들어졌는지 확인하십시오. 아직 삭제하지 마십시오! 다음 레슨에서 이 파일을 사용합니다.
- 
- 참고 : v 를 눌러 비주얼(Visual) 선택을 시작합니다. 커서를 주변으로 움직여 선택 부분을 조절할 수 있습니다. 그리고 명령어를 이용해 해당 문자열을 조작할 수 있습니다. 예를 들어, d 를 이용해 삭제할 수도 있습니다.

## Lesson 5.4: 파일 읽어들이기, 합치기

- **\*\* 어떤 파일의 내용을 삽입하려면, `:r FILENAME` 하십시오 \*\***
- 1. 커서를 이 라인 바로 위로 옮기십시오.
- 
- 주의: 3번째 단계를 실행하면, Lesson 5.3 을 보게 될 것입니다. 그렇게 되면
- 이 lesson으로 다시 내려오십시오.
- 
- 2. 이제 TEST 파일을 읽어들이시다. `:r TEST` 명령을 사용하십시오. TEST 는
- 파일의 이름입니다. 읽어들이는 파일은 커서가 위치한 문장 아래부터 놓이게 됩니다.
- 
- 3. 파일이 읽어들이어진 것을 확인하기 위해, 뒤로 이동해서 기존 버전과 파일에서
- 읽어들이는 버전, 이렇게 Lesson 5.3 이 두번 반복되었음을 확인하십시오.
- 
- 참고: 외부 명령어의 결과값도 읽을 수 있습니다. 예를 들어, `:r !!s` 는 ls 명령어에 대한 결과값을
- 읽어 커서 바로 아래에 합칩니다.



# LESSON 5 요약

- 1. `!:command` 를 이용하여 외부 명령을 실행합니다.
- 유용한 예:
- (MS-DOS)         (Unix)
- `!:dir`         `!:ls`         - 디렉토리의 목록을 보여준다.
- `!:del FILENAME`   `!:rm FILENAME`   - FILENAME이라는 파일을 지운다.
- 2. `:w FILENAME` 하면 현재 빔에서 사용하는 파일을 FILENAME이라는 이름으로
- 디스크에 저장합니다.
- 3. `v` 명령 `:w FILENAME` 은 비주얼 모드에서 선택된 문장들을 파일 FILENAME에 저장합니다.
- 4. `:r FILENAME` 은 디스크에서 FILENAME이라는 파일을 불러들여서 커서 위치
- 뒤에 현재 파일을 집어넣습니다.
- 5. `:r !dir` 는 `dir` 명령어의 결과값을 현재 커서의 위치 아래에 붙입니다.

## Lesson 6.1: 새 줄 열기(OPEN) 명령

- **\*\* o** 를 누르면 커서 아래에 줄을 만들고 편집 모드가 됩니다. **\*\***
- 1. 아래에 ---> 로 표시된 줄로 커서를 옮기십시오.
- 2. o (소문자)를 쳐서 커서 \*아래에\* 줄을 하나 여십시오. 편집 모드가 됩니다.
- 3. ---> 로 표시된 줄을 복사한 후 <ESC> 를 눌러서 편집 모드에서 나오십시오.
- ---> After typing o the cursor is placed on the open line in Insert mode.
- 4. 커서 \*위에\* 줄을 하나 만드려면, 소문자 o 대신 대문자 O 를 치면 됩니다.
- 아래 있는 줄에 대해 이 명령을 내려보십시오.
- ---> Open up a line above this by typing O while the cursor is on this line.

## Lesson 6.2: 추가(APPEND) 명령

- \*\* a 를 누르면 커서 \*다음에\* 글을 입력할 수 있습니다. \*\*
- 1. 커서를 ---> 로 표시된 첫번째 줄의 끝으로 옮깁니다.
- 2. e 를 눌러 li 의 끝으로 커서를 옮깁니다.
- 3. 소문자 a 를 커서 아래 글자 \*다음\*에 글을 추가할 수 있습니다.
- 4. 아랫줄과 같이 문장을 완성해 봅니다. <ESC>를 이용해 편집(Insert) 모드를 나갑니다.
- 5. e 를 이용해 다음 고칠 단어로 움직여 3에서 4까지를 반복합니다.
- 참고: 그렇게 하시면 고작 줄의 끝에 추가를 하기 위해 i를 누르고, 커서 아래에 있던 글자를 반복하고, 글을 끼워넣고, <ESC>를 눌러 명령 모드로 돌아와서,
- 커서를 오른쪽으로 옮기고 마지막으로 x까지 눌러야 하는 번거로움을 피하실 수 있습니다.
- 3. 이제 첫 줄을 완성하십시오. 추가 명령은 텍스트가 입력되는 위치 외에는 편집 모드와 완전히 같다는 것을 유념하십시오.
- ---> This li will allow you to pract appendi text to a line.
- ---> This line will allow you to practice appending text to a line.
- 참고: a, i 그리고 A 는 텍스트가 입력되는 위치 외에는 편집 모드와 완전히 같다는 것을 유념하십시오.

## Lesson 6.3: 치환(REPLACE) 의 다른 버전

- \*\* 대문자 R 을 입력하면 하나 이상의 글자를 바꿀 수 있습니다. \*\*
- 1. ---> 로 표시된 첫번째 줄로 움직여 커서를 xxx의 앞으로 옮깁니다.
- 2. R 을 입력한 후, 두번째 줄과 같은 숫자를 입력해 xxx를 치환합니다.
- 3. <ESC> 를 눌러 치환 모드를 빠져나갑니다. 나머지 문장은 그대로 남아 있는지 확인합니다.
- 4. 위 단계들 반복하여 남은 모든 xxx를 치환합니다.
- ---> Adding 123 to xxx gives you xxx.
- ---> Adding 123 to 456 gives you 579.
- 주의: 치환 모드는 편집 모드와 비슷합니다. 하지만 입력된 문자들이 원래 문자들을 삭제하는 점이 다릅니다.

## Lesson 6.4: 문자 복사 붙여넣기(COPY AND PASTE)

- **\*\* y 를 이용해 복사하고 p 로 붙여 넣습니다. \*\***
- 
- 1. ---> 로 표시된 줄로 움직여 커서를 "a)" 뒤로 옮깁니다.
- 
- 2. v 를 눌러 비주얼 모드를 시작하고 "first" 바로 앞까지 커서를 움직입니다.
- 
- 3. y 를 눌러 하이라이트 된 부분을 복사(yank (copy))합니다.
- 
- 4. 커서를 다음 문장의 끝으로 옮깁니다: j\$
- 
- 5. p 를 눌러 문자열을 붙여 넣습니다.(paste) 그리고 second <ESC> 를 입력합니다.
- 
- 6. 비주얼 모드를 이용해 " item."을 선택, y 로 복사, j\$ 으로 다음 문장 끝으로 움직여
- p 로 단어를 붙여 넣습니다.
- 
- ---> a) this is the first item.
- b)
- 
- 참고: y 역시 명령어로 사용 가능합니다. 예를 들어, yw 는 한 단어를 복사합니다.

# Lesson 6.5: 옵션 설정(SET)

- **\*\* 찾기나 바꾸기에서 대소문자 구분을 없애기 위해 옵션을 설정합니다 \*\***
- 1. 다음을 입력하여 'ignore' 를 찾으십시오: `/ignore <ENTER>`
- `n` 키를 이용하여 여러번 반복하십시오.
- 2. 'ic' (대소문자 구별 안함, ignore case) 옵션을 설정하십시오:
- `:set ic`
- 3. `n` 키를 눌러서 'ignore' 를 다시 찾아보십시오.
- 이제 ignore과 IGNORE 모두 검색되는 점을 주목합니다.
- 4. 'hlsearch' 와 'incsearch' 옵션을 설정합니다.
- `:set hls is`
- 5. 찾기 명령을 다시 입력하여, 어떤 일이 일어나는지 확인해 보십시오:
- `/ignore`
- 6. 대소문자 구별을 끄기 위해서는, 다음과 같이 입력합니다:
- `:set noic`
- 참고: 찾은 내용이 강조(HIGHLIGHT)된 것을 없애려면: `:nohlsearch`
- 참고: 만약, 검색 한번에 대해서만 대소문자 구별 세팅을 끄고 싶다면 `Wc` 를 이용할 수 있습니다.
- `: /ignoreWc <ENTER>`

## LESSON 6 요약

- 1. o 를 입력하면 커서 \*아래에\* 한 줄이 열리며, 커서는 편집 모드로 열린 줄 위에 위치하게 됩니다.
- 대문자 O 를 입력하면 커서가 있는 줄의 \*위로\* 새 줄을 열게 됩니다.
- 2. a 를 입력하면 커서 \*다음에\* 글을 입력할 수 있습니다.
- 대문자 A 를 입력하면 자동으로 그 줄의 끝에 글자를 추가하게 됩니다.
- 3. e 를 입력하면 단어의 끝으로 움직입니다.
- 4. y 를 입력하면 복사(yank (copy))를, p 를 입력하면 붙여 넣기가 됩니다.
- 5. 대문자 R 을 입력하면 <ESC> 를 눌러서 나가기 전까지 바꾸기 모드가 됩니다.
- 6. ":set xxx" 를 하면 "xxx" 옵션이 설정됩니다.:
- 'ic' 'ignorecase'            검색시 대소문자 구별을 하지 않습니다.
- 'is' 'incsearch'            검색어에서 부분 검색 결과를 보여줍니다.
- 'hls' 'hlsearch'            검색 결과값을 하이라이트해줍니다.
- 옵션은 전체 이름 혹은 줄인 이름 모두 사용 가능합니다.
- 7. 앞에 "no"를 붙여 옵션을 끌 수 있습니다: :set noic

# LESSON 7.1: 온라인 도움말 명령

- **\*\* 온라인 도움말 시스템 사용하기 \*\***
- 빔은 폭 넓은 온라인 도움말 시스템을 제공합니다. 도움말을 보려면,
  - 다음 세가지 중 하나를 시도해보십시오:
    - - <HELP> 키를 누른다. (키가 있는 경우)
    - - <F1> 키를 누른다. (키가 있는 경우)
    - - :help <ENTER> 라고 입력한다.
- 도움말 창을 닫으려면 :q <ENTER> 라고 입력하십시오.
- CTRL-W CTRL-W 다른쪽 윈도우로 넘어갑니다.
- :q <ENTER> 도움말 윈도우를 닫습니다.
- ":help" 라는 명령에 인자를 주면 어떤 주제에 관한 도움말을 찾을 수 있습니다.
- 다음 명령을 내려 보십시오. ( <ENTER> 키를 누르는 것을 잊지 마십시오.)
  - :help w
  - :help c\_CTRL-D
  - :help insert-index
  - :help user-manual



## LESSON 7.2: 시작 스크립트 만들기

- **\*\* 빔의 기능 켜기 \*\***
- 빔은 Vi 보다 훨씬 많은 기능을 가지고 있지만, 대부분은 기본적으로 작동하지 않습니다. 더 많은 기능을 써보려면, "vimrc" 라는 파일을 만들어야 합니다.
- 1. "vimrc" 파일을 수정합니다. 이 파일은 사용하는 시스템에 따라 다릅니다:
  - `:e ~/.vimrc`                  Unix의 경우
  - `:e ~/.vimrc`                  MS-Windows의 경우
- 2. 이제 "vimrc"의 예제를 읽어들이십시오:
  - `:r $VIMRUNTIME/vimrc_example.vim`
- 3. 다음과 같이 하여 파일을 저장합니다:
  - `:w`
- 다음 번에 빔을 시작하면, 구문 강조(syntax highlighting)이 사용될 것입니다.
- 모든 원하는 설정을 이 "vimrc" 파일에 넣어둘 수 있습니다.
- 더 자세한 내용은 `:help vimrc-intro`를 참고 하세요.

# 가장 많이 사용하는 설정 값

- " 기본 설정
  - set number " 라인 번호 표시
  - set showcmd " 명령어 입력 시 상태 표시줄에 보여줌
  - set cursorline " 현재 줄 강조
  - set showmatch " 괄호 짝 보여주기
  - set hlsearch " 검색어 하이라이트
  - set incsearch " 입력하면서 검색
  - set ignorecase " 검색 시 대소문자 무시
  - set smartcase " 대문자 포함 시 대소문자 구분
- " 탭 관련
  - set tabstop=4 " 탭은 4칸
  - set shiftwidth=4 " 자동 들여쓰기 간격
  - set expandtab " 탭 대신 스페이스
- " 기타 편의
  - syntax on " 구문 강조
  - filetype plugin indent on " 파일 형식에 따라 자동 설정
  - set autoindent " 자동 들여쓰기
  - set clipboard=unnamedplus " 시스템 클립보드와 공유
- " 백업/스왑 파일 제거
  - set noswapfile
  - set nobackup
  - set nowritebackup
- " 줄 길이 표시
  - set colorcolumn=80
  - highlight ColorColumn ctermbg=darkgray
- " 줄 바꿈 설정
  - set wrap " 긴 줄 줄바꿈 (원하는 경우)

## Lesson 7.3: 명령어 완성하기

- **\*\* CTRL-D 와 <TAB> 이용하여 명령어를 완성할 수 있습니다.\*\***
- 1. 먼저 vim이 호환 모드가 아닌지를 확인합니다: `:set nocp`
- 2. 디렉토리에 파일이 존재하는지 먼저 확인 합니다.: `!!ls` 혹은 `!!dir`
- 3. 다음과 같이 명령어를 입력합니다: `:e`
- 4. CTRL-D 를 누르면 "e"로 시작하는 모든 명령어들을 볼 수 있습니다.
- 
- 5. <TAB> 을 눌러 ":edit" 명령어를 완성해 봅니다.
- 
- 6. 이제 빈칸 하나를 추가한 뒤, 존재하는 파일 이름의 앞 부분을 입력합니다: `:edit FIL`
- 7. <TAB> 을 눌러 파일 이름을 완성 시킵니다.
- 
- 참고: 완성하기는 많은 명령어에서 사용할 수 있습니다. CTRL-D와 <TAB> 만 누르세요!
- 특히, `:help` 에서 유용할 것입니다.

## Lesson 7 요약

- 1. 도움말을 열기 위해 :help 혹은 <F1> 혹은 <Help> 를 누릅니다.
- 
- 2. cmd 에 대한 도움말을 보기 위해서는 :help cmd 를 입력합니다.
- 
- 3. CTRL-W CTRL-W 를 이용해 다른 윈도우로 넘어갑니다.
- 
- 4. :q 로 도움말 윈도우를 빠져나옵니다.
- 
- 5. vimrc 시작 스크립트를 이용해 선호하는 세팅을 유지할 수 있습니다.
- 
- 6. : 명령어를 입력할때, CTRL-D 를 눌러 가능한 명령어들을 볼수 있습니다.
- <TAB> 을 눌러 완성 가능합니다.

# 이것으로 빔 길잡이를 마칩니다.

- 이 길잡이는 빔 편집기에 대한 간략한 개요를 보여주기 위한 의도로 제작되었으며, 이 편집기를 정말 간단히 사용하기에 충분할 뿐입니다.
- 빔에는 이 길잡이와는 비교할 수 없을 만큼 훨씬 많은 명령이 있습니다. 다음 사용자 매뉴얼을 읽으십시오: `":help user-manual"`
- 보다 자세히 읽고 공부하려면, 다음 책을 추천해 드립니다:
- Vim - Vi Improved - by Steve Oualline / 출판사: New Riders
- 이 책은 완전히 빔에 대해서만 다루고 있습니다. 특히 초보자들에게 유용합니다.
- 많은 예제와 그림이 있습니다.
- 다음을 참고하십시오: <https://iccf-holland.org/click5.html>
- 다음 책은 좀 오래된 책으로 빔보다는 Vi에 대해 다루고 있지만, 역시 추천할 만 합니다:
- Learning the Vi Editor - by Linda Lamb / 출판사: O'Reilly & Associates Inc.
- Vi로 하고 싶은 거의 모든 것에 대해 알 수 있는 좋은 책입니다.
- 여섯번째 개정판은 빔에 관한 내용을 포함하고 있습니다.
- 이 길잡이는 Colorado School of Mines의 Michael C. Pierce 와
- Robert K. Ware 가 Colorado State University의 Charles Smith 의 아이디어에 착안하여 썼습니다.
- E-mail: [bware@mines.colorado.edu](mailto:bware@mines.colorado.edu).
- Modified for Vim by Bram Moolenaar.