

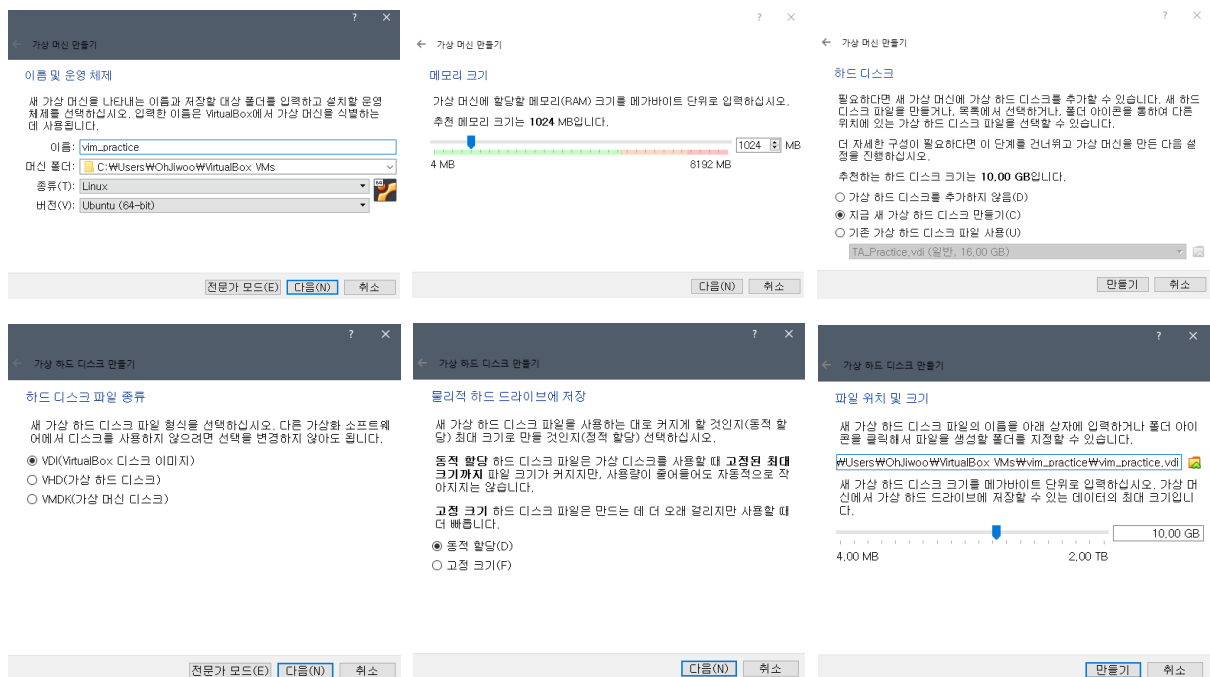
# Linux 텍스트 에디터 Vim 튜토리얼

안녕하세요! 소프트웨어학과 오지우입니다.

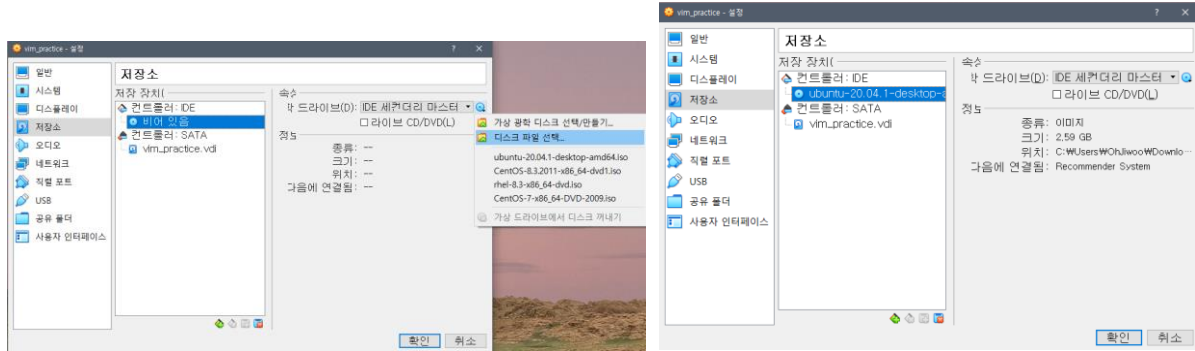
이번 튜토리얼에서는 유명한 오픈소스 소프트웨어 중 하나인 리눅스의 텍스트 편집기 “Vim 에디터”에 대해 알아보겠습니다. 리눅스는 Window, macOS 같은 운영체제입니다. Window에서는 텍스트 편집기로 ‘메모장’을 제공해주고 있는데, 리눅스에서 메모장과 같은 역할을 하는 것이 Vim이라고 이해하면 될 것 같습니다.

Vim을 사용하기 위해서는 리눅스가 우선 설치되어있어야 하므로 리눅스 배포판 가운데 하나인 Ubuntu를 Virtual Box를 이용하여 설치하겠습니다. <https://www.virtualbox.org/> 다음의 링크로 접속하여 Virtual Box를 설치합니다. Virtual Box는 Window 운영체제를 가진 컴퓨터에서도 리눅스를 사용할 수 있도록 가상의 컴퓨터를 제공해주는 것이라고 보면 됩니다. 하나의 컴퓨터 자원(CPU, 메모리 등)을 함께 공유해서 사용하는 것이죠. 이제 가상머신을 만들어보겠습니다.

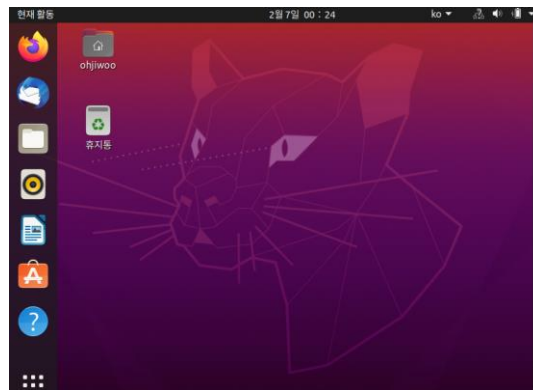
## 1. 가상머신 이름 및 운영체제 종류, 메모리/하드디스크 용량 등을 설정해줍니다.



## 2. 설치된 가상머신에 우분투를 적용하기 위해 <https://ubuntu.com/#download> 다음의 링크로 가서 우분투 iso 파일을 설치한 뒤 [설정]에서 저장소에 적용해줍니다.

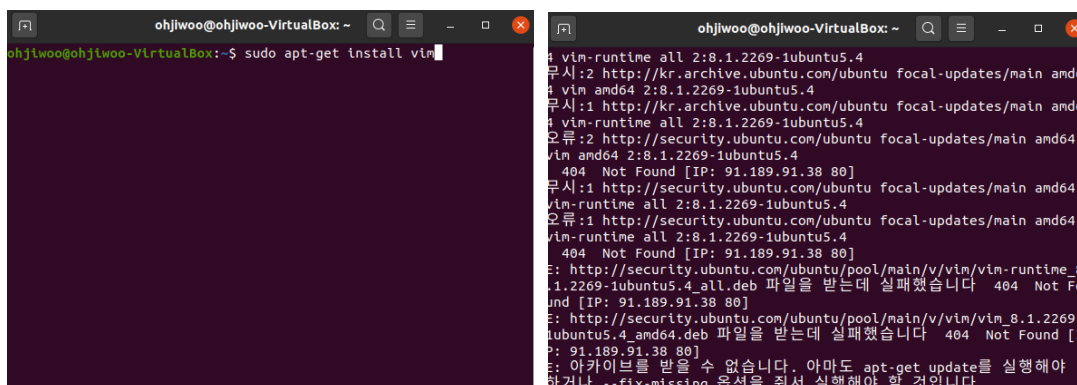


3. 가상머신을 실행하여 설치를 마무리해주시면 됩니다.



설치가 완료된 모습

Vim은 기본적으로 내장되어있어서 우분투를 설치하면 바로 사용이 가능합니다. 하지만 기본적인 Vi 에디터에서는 텍스트를 입력하고 수정하는 데에 불편한 점이 많아 다시 갈아주는 것이 좋습니다.



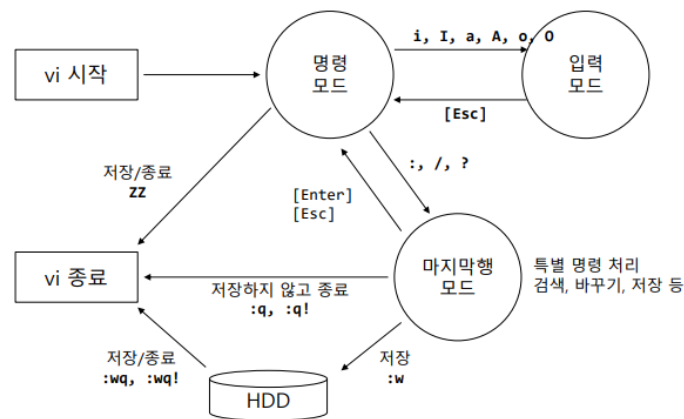
**Sudo apt-get install vim** 명령어로 설치하면 되는데, 다음과 같이 오류가 뜨는 경우가 있을 수 있습니다. 그럴 경우 **sudo apt-get update** 명령어를 실행한 뒤에 다시 시도해보면 정상적으로 설치가 완료되는 것을 확인할 수 있습니다.

Vim에는 명령모드와 입력모드, 마지막 행 모드로 모드가 3가지 있습니다.

처음 Vim에 접속하면 명령모드 상태인데, 방향키로 커서를 이동할 수 있으며 복사, 붙여넣기, 삭제 등의 명령을 수행할 수 있습니다. 명령모드에서 “i”와 같은 명령어를 입력하여 입력모드로 전환할 수 있는데, 입력모드에서는 말 그대로 문자를 입력할 수 있습니다.

Esc 버튼으로 다시 명령모드로 전환할 수 있습니다. 명령모드 상태에서 : (세미콜론)을 입력하면 마지막 행 모드로 전환하여 화면의 하단에서 명령어를 입력할 수 있습니다. 수정 중인 파일을 저장하거나 저장하지 않고 빠져나온다거나, 문자열을 검색하고 특정 문자열을 다른 문자열로 치환하는 등의 명령을 수행할 수 있습니다.

다음은 vim의 전체 동작 구조입니다.



Vim 동작 모드

이제 각 모드 별로 더 자세한 명령어들을 살펴보도록 하겠습니다.

## 1. 명령모드에서의 명령어

명령어 종류	명령어	설명
입력모드로의 전환	i	현재 커서 바로 앞에 삽입
	a	현재 커서 바로 뒤에 삽입
	o	현재 커서가 위치한 행의 다음 행에 삽입
삭제	x, #x	현재 커서가 위치한 곳의 글자 삭제, #은 삭제할 글자 수를 지정
	dw, #dw	현재 커서가 위치한 곳에서부터 한 단어 삭제(띄어쓰기 전까지), #은 삭제할 단어의 수를 지정
	dd, #dd	현재 커서가 위치한 곳의 행 전체 삭제, #은 삭제할 행의 수를 지정(삭제이지만 p 명령어로 복구할 수 있음)
복사	yy, #yy	현재 줄을 버퍼로 복사, #은 복사할 행의 수를 지정
붙여넣기	p	현재 커서가 있는 행의 다음 행에 버퍼 내용 모두 붙여넣기
실행 취소	u	바로 전에 한 명령어 취소

커서 이동	O(숫자 0)	현재 커서가 있는 행의 맨 앞으로 커서 이동
	\$	현재 커서가 있는 행의 맨 뒤로 커서 이동
	G(대문자 g)	현재 파일의 맨 끝으로 커서 이동

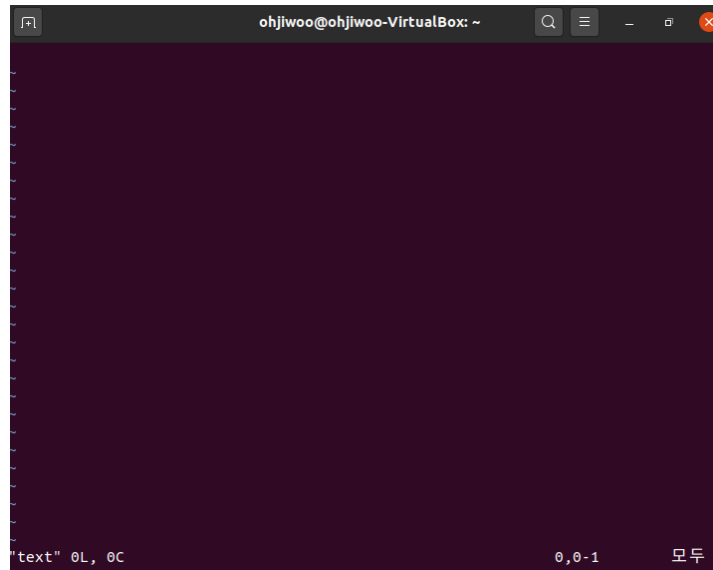
## 2. 마지막행 모드에서의 명령어

명령어 종류	명령어	설명
파일 저장 및 vim 종료	w [파일명]	파일명을 입력하면 입력한 이름으로 파일을 저장, 입력하지 않으면 기존의 파일명으로 저장
	wq, wq!	파일을 저장한 뒤 종료(!가 붙으면 강제로 수행한다는 의미)
	q, q!	파일을 저장하지 않고 종료(!가 붙으면 강제로 수행한다는 의미)
문자열 찾기	/문자열	해당 문자열을 파일의 아래 방향으로 검색(n을 눌러 다음 문자열 탐색, N으로 반대방향으로 탐색가능)
	?문자열	해당 문자열을 파일의 위 방향으로 검색(n을 눌러 다음 문자열 탐색, N으로 반대방향으로 탐색가능)
커서 이동	숫자	해당 라인으로 커서 이동
	\$	파일의 맨 끝 행으로 커서 이동
문자열 치환	s/문자열1/문자열2/, %s/문자열1/문자열2/g	현재 커서가 위치한 행에서 첫 번째로 나오는 문자열1을 문자열2로 치환, %와 g가 앞뒤에 각각 붙으면 파일 전체에 대해서 문자열1을 문자열2로 치환(%에 범위를 지정하면 해당 범위에서만 치환 가능)

### cf) 마지막행 모드에서 범위 지정 방법

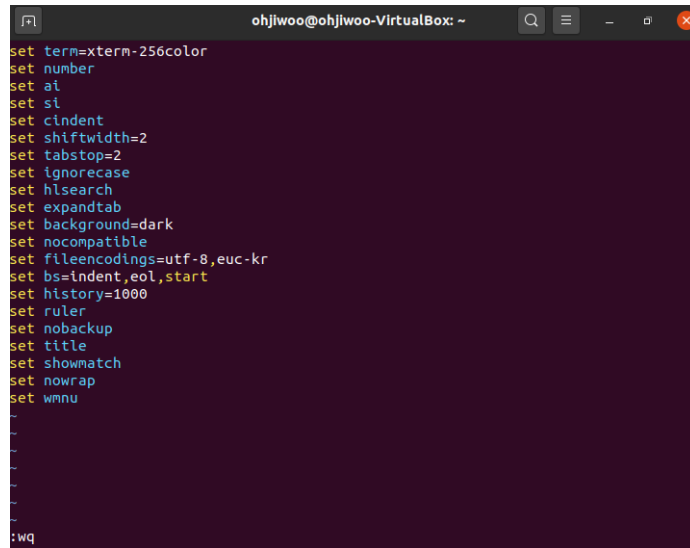
범위 지정 방법	설명
1,\$ 또는 %	1행부터 마지막 행까지 지정
1,.	1행부터 현재 커서가 있는 행까지 지정
.,\$	현재 커서가 있는 행부터 마지막 행까지 지정
10,20	10행부터 20행까지 지정

대략적인 명령어들을 살펴보았는데, 이제는 vim을 텍스트 에디터로 사용하기 편하게 하기 위한 방법을 알아보도록 하겠습니다. Vim에 아무런 설정을 안해주면 다음과 같습니다.

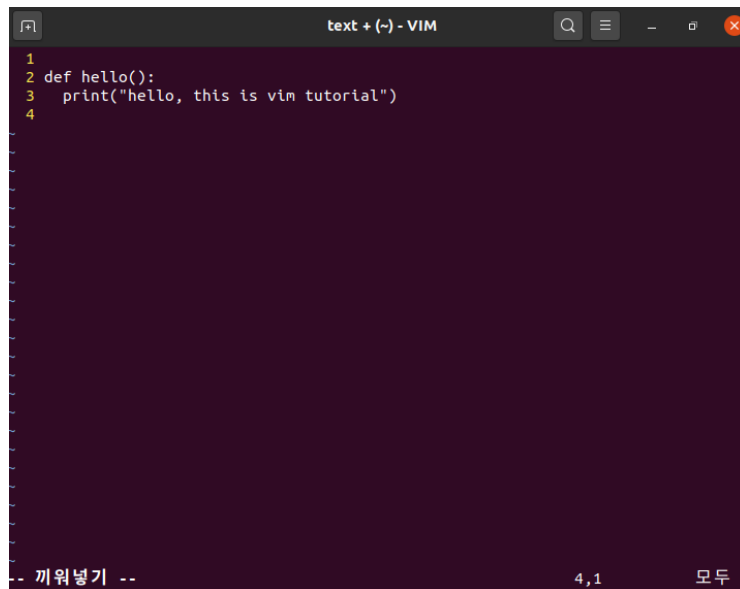


다소 밋밋한 느낌이 들어서 좀 바꿔주도록 하겠습니다. .vimrc 파일을 수정해주면 설정을 해줄 수 있습니다. `vi .vimrc` 명령어로 .vimrc 파일을 열어준 뒤 아래의 내용을 파일에 복사 붙여넣기 해줍니다.

```
set term=xterm-256color
set number " line 표시를 해줍니다.
set ai " auto indent
set si " smart indent
set cindent " c style indent
set shiftwidth=2 " shift 를 2 칸으로 ( >, >>, <, << 등의 명령어)
set tabstop=2 " tab 을 2 칸으로
set ignorecase " 검색시 대소문자 구별하지않음
set hlsearch " 검색시 하이라이트(색상 강조)
set expandtab " tab 대신 띄어쓰기로
set background=dark " 검정배경을 사용할 때, (이 색상에 맞춰 문법 하이라이트 색상이 달라집니다.)
set nocompatible " 방향키로 이동가능
set fileencodings=utf-8,euc-kr " 파일인코딩 형식 지정
set bs=indent,eol,start " backspace 키 사용 가능
set history=1000 " 명령어에 대한 히스토리를 1000 개까지
set ruler " 상태표시줄에 커서의 위치 표시
set nobackup " 백업파일을 만들지 않음
set title " 제목을 표시
set showmatch " 매칭되는 괄호를 보여줌
set nowrap " 자동 줄바꿈 하지 않음
set wmenu " tab 자동완성시 가능한 목록을 보여줌
```

A terminal window titled 'ohjiwoo@ohjiwoo-VirtualBox: ~' showing a list of Vim configuration settings. The settings are listed one by one, each preceded by 'set'. The settings include: term=xterm-256color, number, ai, si, cindent, shiftwidth=2, tabstop=2, ignorecase, hlsearch, expandtab, background=dark, nocompatible, fileencodings=utf-8,euc-kr, bs=indent,eol,start, history=1000, ruler, nobackup, title, showmatch, nowrap, and wmnu. The cursor is at the end of the last line, which is ':wq'.

마지막행 모드에서 wq로 저장한 뒤 빠져나오겠습니다.

A Vim editor window titled 'text + (-) - VIM' showing a Python function definition. The code is: 1 def hello(): 2 print("hello, this is vim tutorial") 3 4. The cursor is at the end of line 4. The status bar at the bottom shows '-- 끼워넣기 --' on the left, '4,1' in the center, and '모두' on the right.

다시 vim을 켜서 파이썬으로 코드를 입력해보았습니다. 우선 행번호가 보이고, 파이썬의 문법에 맞게 자동으로 들여쓰기가 되며, print 함수를 작성할 때 자동으로 여는 괄호와 닫는 괄호를 매칭해줍니다. 이 외에도 설정해준 것들이 적용된 것을 확인할 수 있습니다. 이렇게 하면 IDE를 사용할 때처럼 vim을 이용하여 편리하게 코드를 작성할 수 있을 것입니다.

이상으로 리눅스 텍스트 에디터 Vim의 튜토리얼을 마치겠습니다!