

# 파이썬 기본

최석재

lingua@naver.com



# Python 설치

# Python



- 1991년 네덜란드의 귀도 반 로섬에 의해 만들어진 언어
- Python이라는 명칭은 영국의 전설적인 코미디 그룹 Monty Python에서...
- 간결하고, 명시적으로 코딩하는 것을 기본 원칙으로 삼고 있다
- 코드 작성이 매우 쉬우면서도 NASA에서 사용할 정도로 안정적이다
- Python3부터는 유니코드 기반이어서 한글 처리도 잘 된다
- 빅데이터 분석의 전 과정을 수행할 수 있다
  - 데이터 수집, 전처리, 통계분석, 시각화, 머신러닝, 텐서플로

# 아나콘다

- 파이썬은 설치가 어려운 편이다
- 파이썬은 핵심 엔진과 numpy, scipy, pandas 등 기초 패키지들과, 이들을 기초로 발전시킨 패키지들로 구성되어 있다
- 파이썬은 패키지 업그레이드가 이루어질 때마다 큰 변화가 이루어지고 있고, 그때마다 의존 패키지들과의 호환이 이루어지지 않을 때가 많다
- 특히 텐서플로는 파이썬 최신 버전과 잘 호환되지 않는다
- 아나콘다는 이러한 어려움을 해결하기 위하여 파이썬 엔진 및 패키지들을 호환성 문제가 없게 잘 묶어놓은 커다란 패키지이다
- 따라서 아나콘다로 파이썬을 설치하여 버전 호환성 문제를 겪지 않도록 하는 것이 좋다

# 기존 파이썬, 아나콘다, 파이참 삭제

- Python 3.7 ~ 3.9를 포함하고 있는 Anaconda로 설치하는 것을 권장
- 기존 버전이 있으면 이후 사용이 매우 복잡해진다
- 따라서 기존 버전은 모두 삭제하는 것이 좋다
- 삭제 대상에는 파이썬, 아나콘다, 파이참 모두 해당된다
- 제어판을 통해 모두 삭제하고, 특히 .idea, .venv 등의 폴더를 삭제한다
- 아나콘다를 설치한 폴더와 작업한 폴더에 주로 남아 있다
- 가장 좋은 방법은 개인 파일만 남겨두고 컴퓨터를 포맷하는 것이다

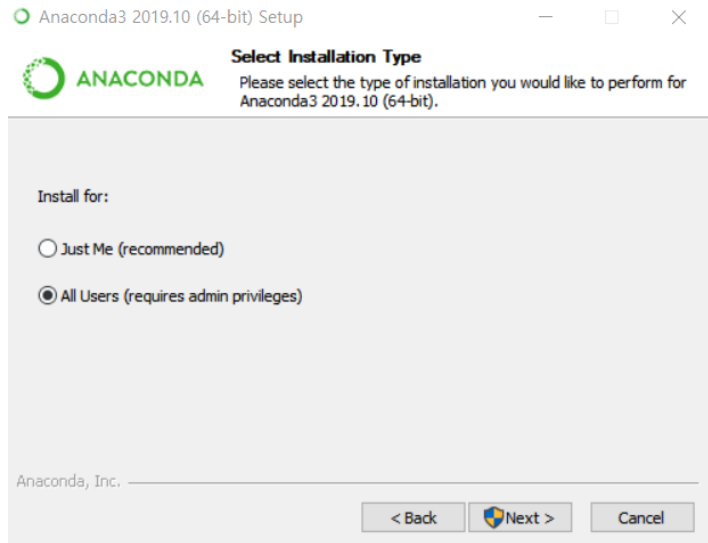
# 아나콘다 설치

<a href="#">Anaconda3-2019.10-Linux-ppc64le.sh</a>	320.3M	2019-10-15 09:26:11	9dd413b0f2d0c68f387541428fe8d565
<a href="#">Anaconda3-2019.10-Linux-x86_64.sh</a>	505.7M	2019-10-15 09:26:05	b77a71c3712b45c8f33c7b2ecade366c
<a href="#">Anaconda3-2019.10-MacOSX-x86_64.pkg</a>	653.5M	2019-10-15 09:27:33	5b051bf25188cd4bdc7794f5bea6886
<a href="#">Anaconda3-2019.10-MacOSX-x86_64.sh</a>	424.2M	2019-10-15 09:27:31	1a56194e89795b7ebbf405b09d9c42d
<a href="#">Anaconda3-2019.10-Windows-x86.exe</a>	409.6M	2019-10-15 09:26:10	0e71632df6a17f625c1103b34f66e8ba
<a href="#">Anaconda3-2019.10-Windows-x86_64.exe</a>	461.5M	2019-10-15 09:27:17	fafcdbf5feb6dc3081bf07cbb8af1dbe

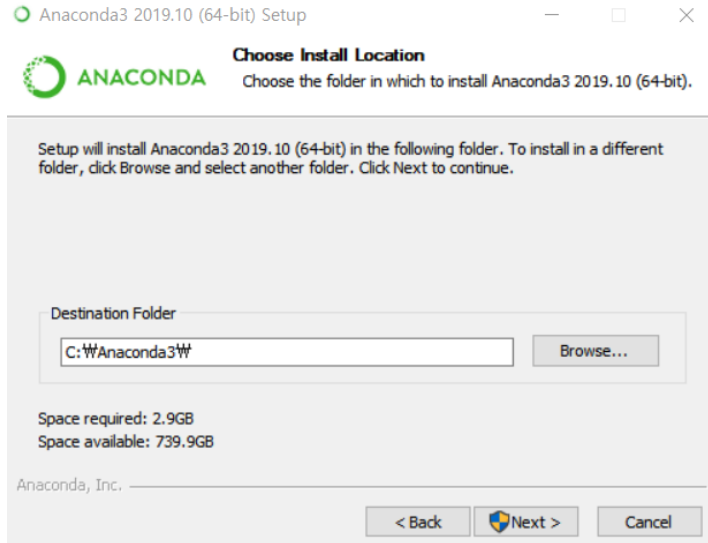
※ 목록의 중간 썸에 있다

- 텐서플로와의 호환을 고려하여 적절한 버전을 설치한다
  - 현재 파이썬 3.9까지 텐서플로와 호환이 잘 되는 것으로 알려져 있으나
  - 구글 Colab은 파이썬 3.7을 이용하며, 많은 사람들이 이 버전을 사용한다
  - 단, 항상 64bit만 텐서플로와 호환된다
- 
- [1]에서 최신 버전을 받고 다운그레이드를 하거나,
  - [2]에서 예전 버전을 찾는 방법이 있다
- 
- [1] <https://www.anaconda.com/products/distribution> 에서 최신 버전을 받는다
  - 설치가 끝나면 관리자 권한으로 Anaconda Prompt를 실행한다
  - conda install python=3.7 로 다운그레이드한다
- 
- [2] <https://repo.anaconda.com/archive/> 에서 2019.10 버전을 받는다 (python 3.7)
  - 윈도우 64 bit의 경우 Anaconda3-2019.10-Windows-x86\_64.exe 를 받는다

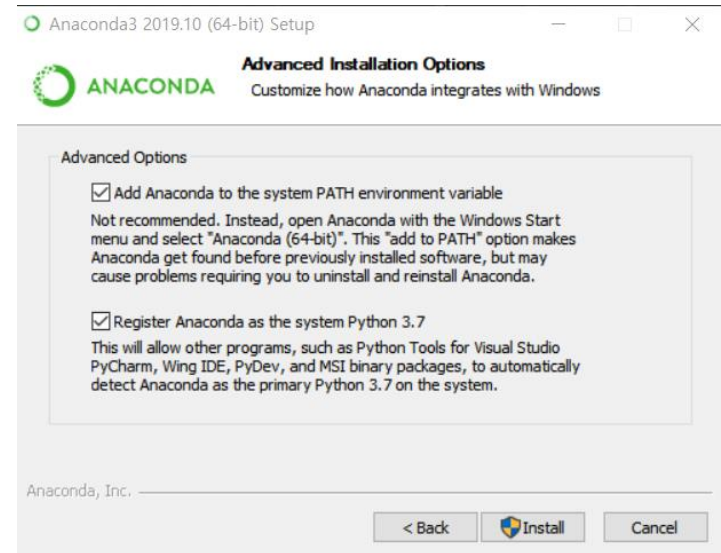
# 설치 옵션



- 관리자 권한으로 실행 파일을 실행한다
- All Users 옵션을 선택한다



기본 경로도 괜찮지만  
C:\Anaconda3\  
로 하면 패키지를 찾을  
때 편리하다



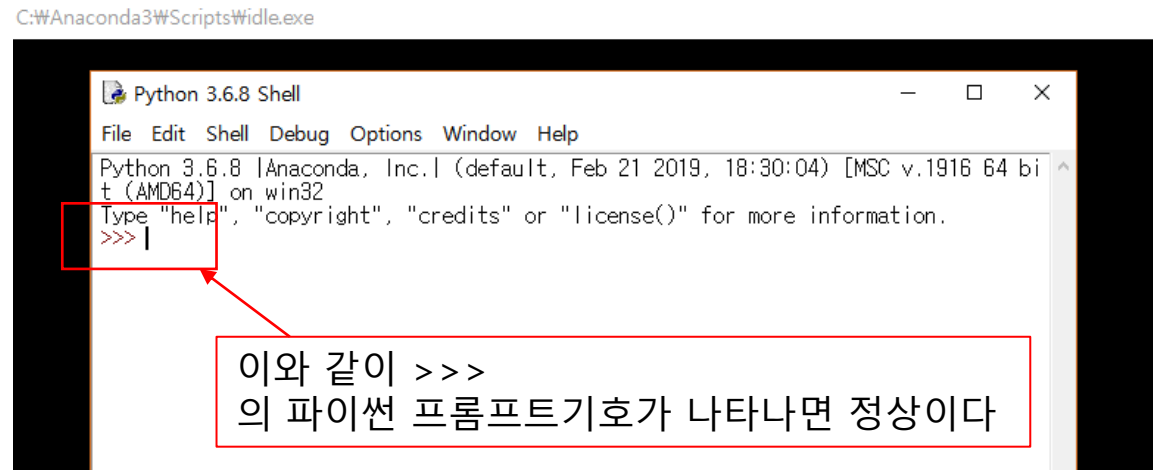
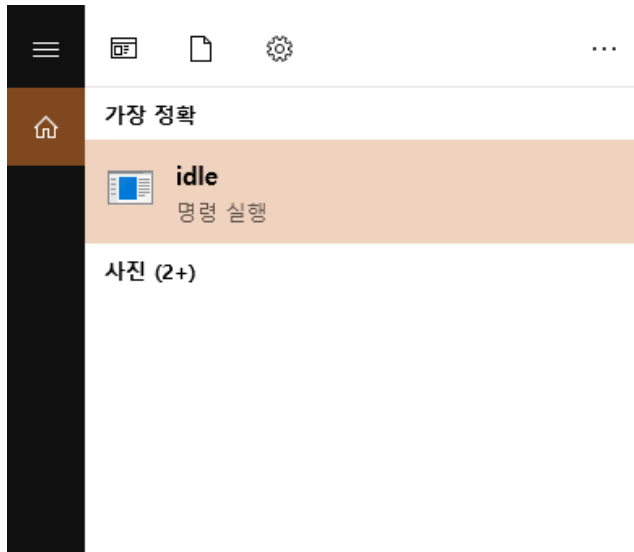
- 두 개의 옵션 모두 선택한다
- 첫 번째 옵션을 선택하면 아나콘다를 제거했다가 다시 설치해야 할 수도 있지만 이 방법이 이후 사용에 더 편리하다
- 첫 번째 옵션은 환경변수에 PATH 설정을 하는 것으로 어디서나 anaconda를 사용할 수 있게 하여 편리하지만, 다른 python 버전이 있을 때는 충돌이 날 수 있다
- 두 번째 옵션은 anaconda를 python 3.7의 대표 시스템으로 등록한다는 뜻이다

코딩 도구



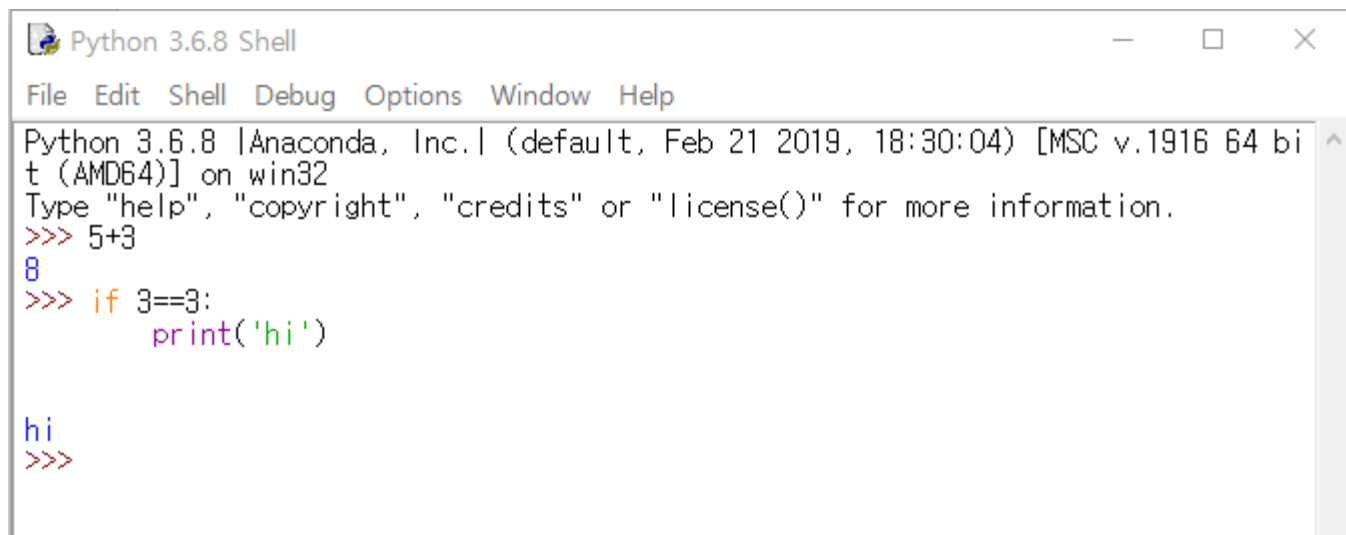
# 1. 파이썬 셸

- 아나콘다 등을 통해 파이썬을 잘 설치했다면 파이썬 셸(shell)을 이용할 수 있다
- 파이썬 셸은 파이썬 코드를 실행하는 가장 기본적인 도구이다
- 검색창에 idle(Integrated Development and Learning Environment)를 입력하여 파이썬 셸을 실행시킨다



# 파이썬 쉘에서의 코딩

- 다음을 입력해본다
- `5+3`
- `if 3==3:`  
    `print('hi')`



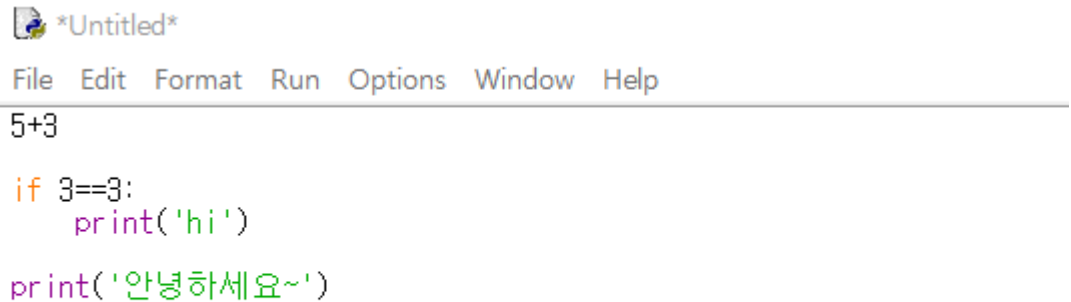
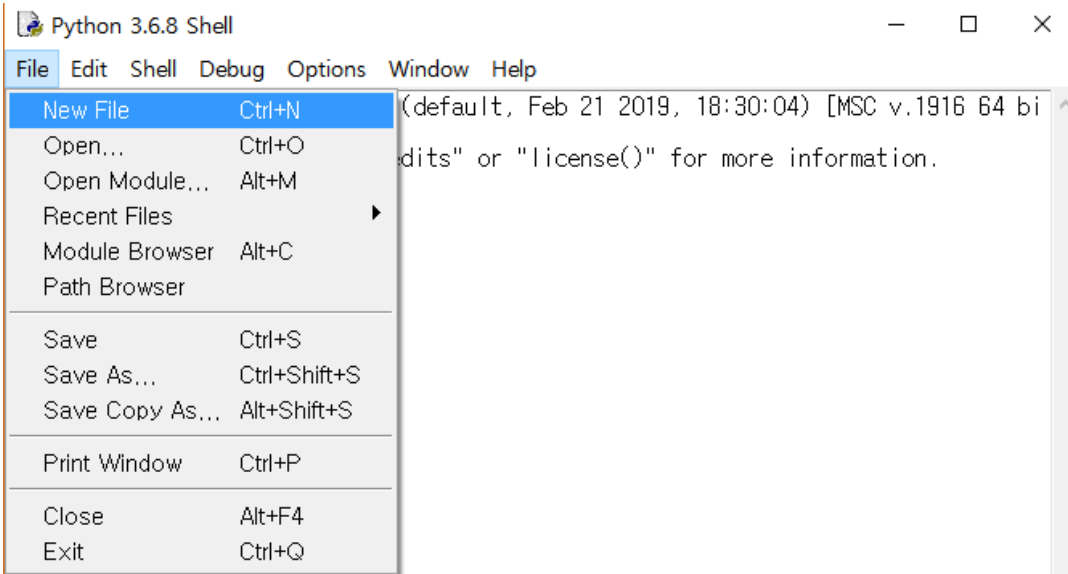
```
Python 3.6.8 Shell
File Edit Shell Debug Options Window Help
Python 3.6.8 |Anaconda, Inc.| (default, Feb 21 2019, 18:30:04) [MSC v.1916 64 bi
t (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 5+3
8
>>> if 3==3:
    print('hi')

hi
>>>
```

- 파이썬 쉘에서의 코딩은 간단하지만 여러 줄의 코드를 작성하기는 불편하다

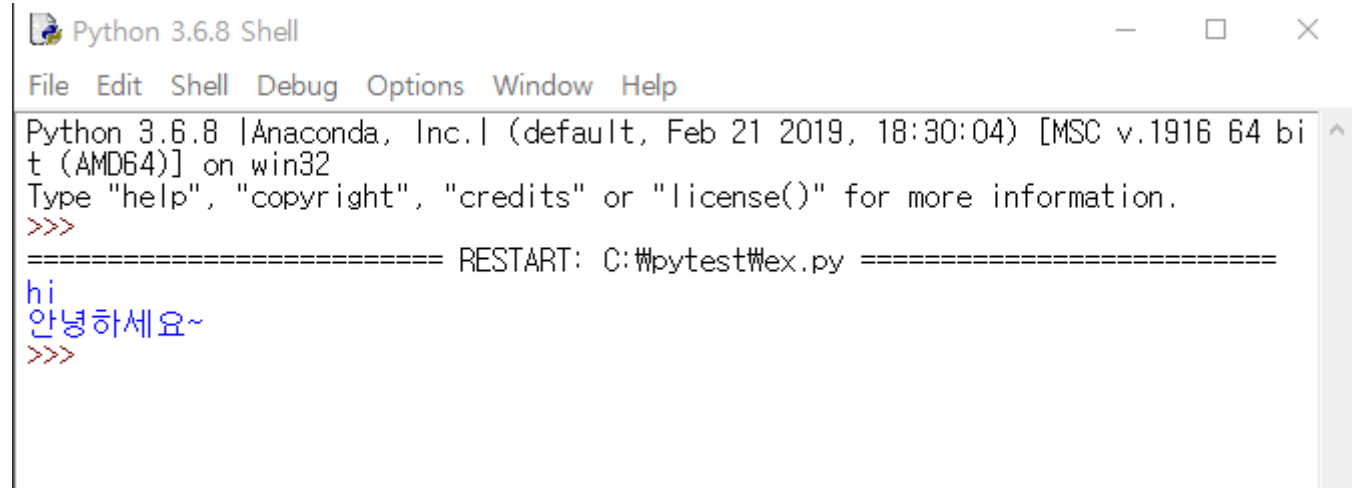
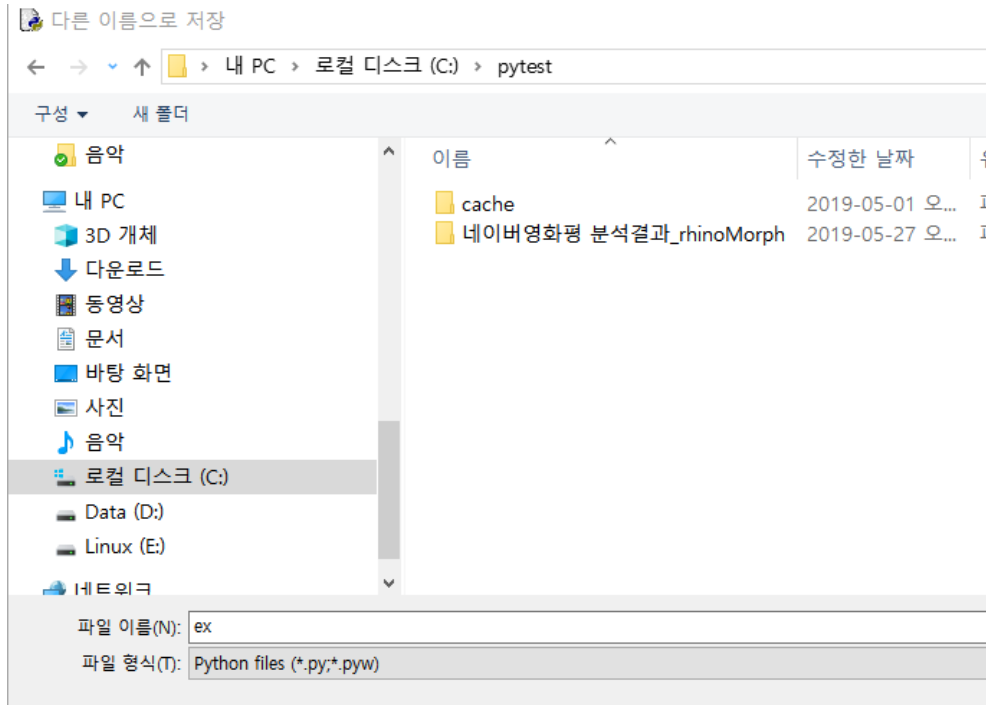
## 2. 메모장 + 파이썬 셸

- 여러 줄의 코드를 쉽게 작성하기 위하여 메모장에 관련 코드를 작성한다
- Shell의 File-New File 을 클릭하여 파이썬 메모장을 띄운다
- 일반 메모장보다는 코드 작성에 편리한 기능을 제공한다
- 그리고 아래와 같이 코드를 작성한다



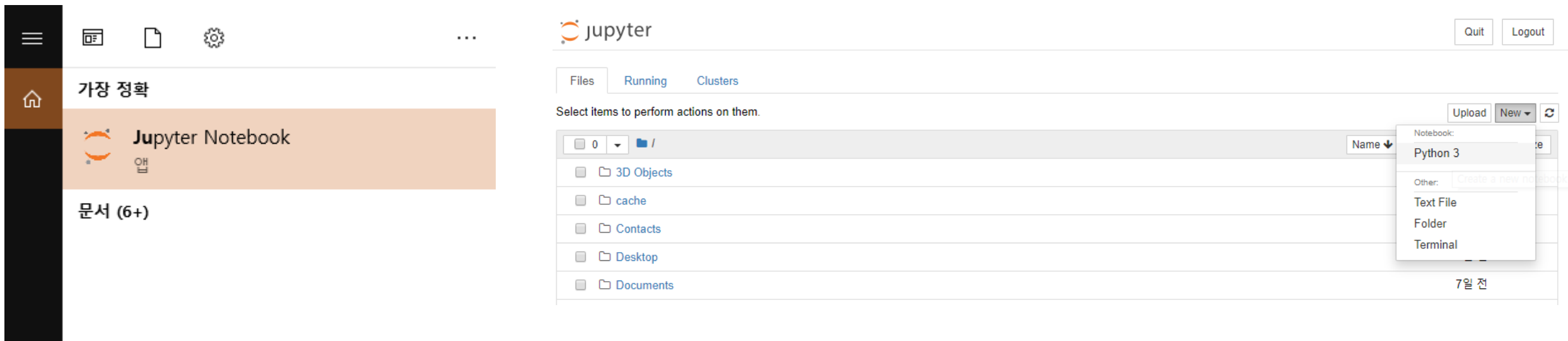
# 저장 및 실행

- Ctrl-S 를 눌러 C:/pytest 폴더에 ex.py라는 이름으로 저장한다
- 열려있는 메모장을 닫고, 위의 파일을 불러온다
- Run – Run Module 또는 F5 키를 눌러 내용을 실행한다



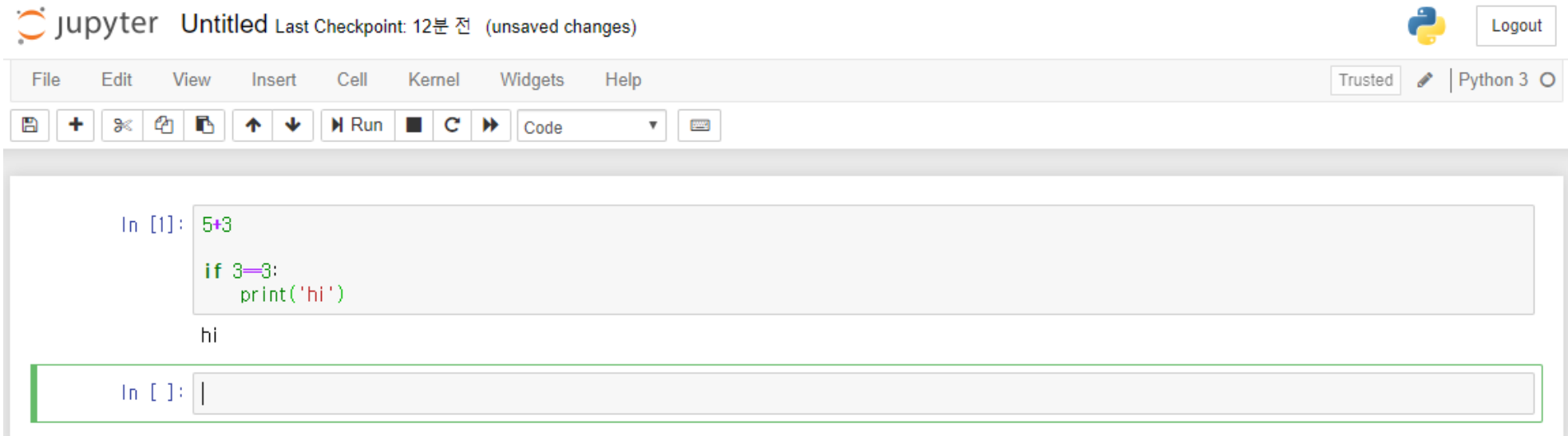
# 3. Jupyter

- 쥬피터 노트북을 이용하여 보다 편리하게 코딩할 수 있다
- 쥬피터 노트북은 Python+α인 IPython을 사용한다
- 쥬피터 노트북은 인터랙티브한 환경을 제시하며
- Markdown 기능으로 코드를 인터넷 웹문서처럼 작성할 수 있다
- Jupyter Notebook 실행 후 New – Python3 를 선택한다



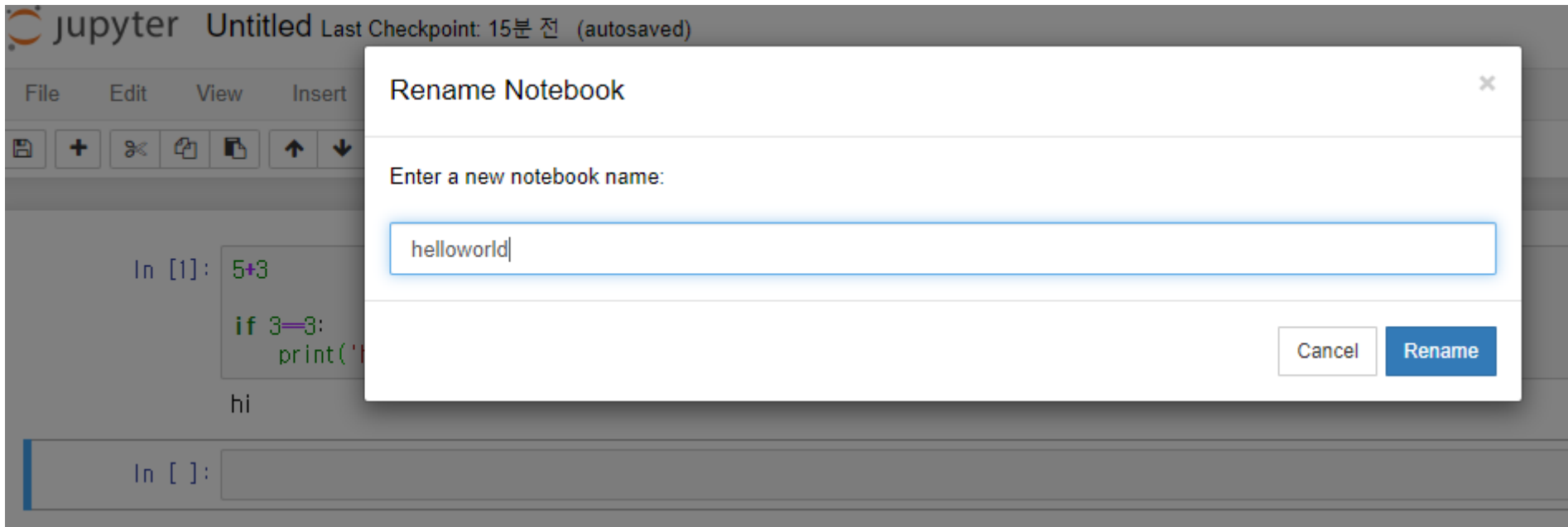
# 실행

- 앞의 코드를 셀 안에 작성한다
- 새로 생긴 셀 안에도 새로운 코드를 작성할 수 있다
- Shift-Enter를 눌러 실행한다



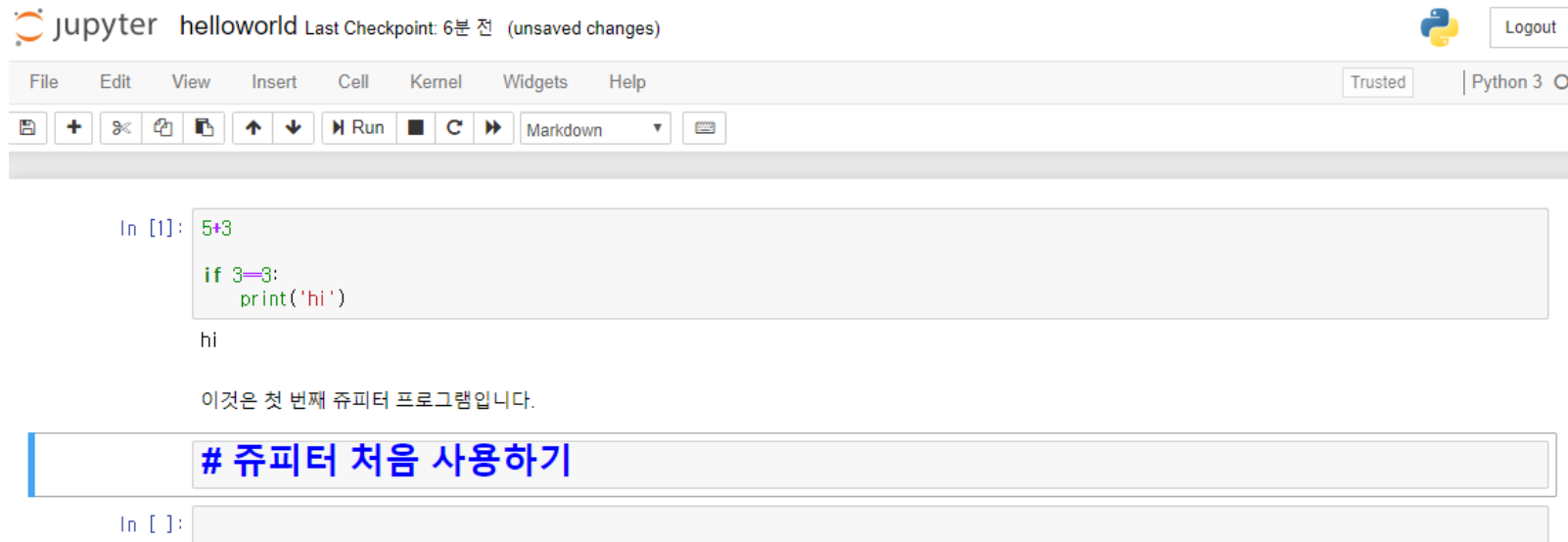
# 저장

- Untitled 부분을 클릭하여 파일의 이름을 'helloworld'라고 변경한다
- File – Save as 기능을 이용할 수도 있다



# 편집

- 아래의 새로 생긴 셀을 클릭한다
- Code 라고 되어 있는 Markdown으로 변경한다
- 적절한 설명글을 남기고 저장한다. #을 누르고 작성하면 제목으로 생성된다
- Shift-Enter를 눌러 실행한다





## 4. PyCharm

- 본격적인 개발에 PyCharm을 많이 사용한다
- 파이참은 프로젝트 관리, 자동완성 기능 등이 돋보이는 편집기이다
- <https://www.jetbrains.com/pycharm/> 에서 Community 버전을 받는다
- 유료 버전인 Professional은 주로 웹과 관련된 기능을 추가한다



### Download PyCharm

Windows

macOS

Linux

#### Professional

For both Scientific and Web Python development. With HTML, JS, and SQL support.

DOWNLOAD

Free trial

#### Community

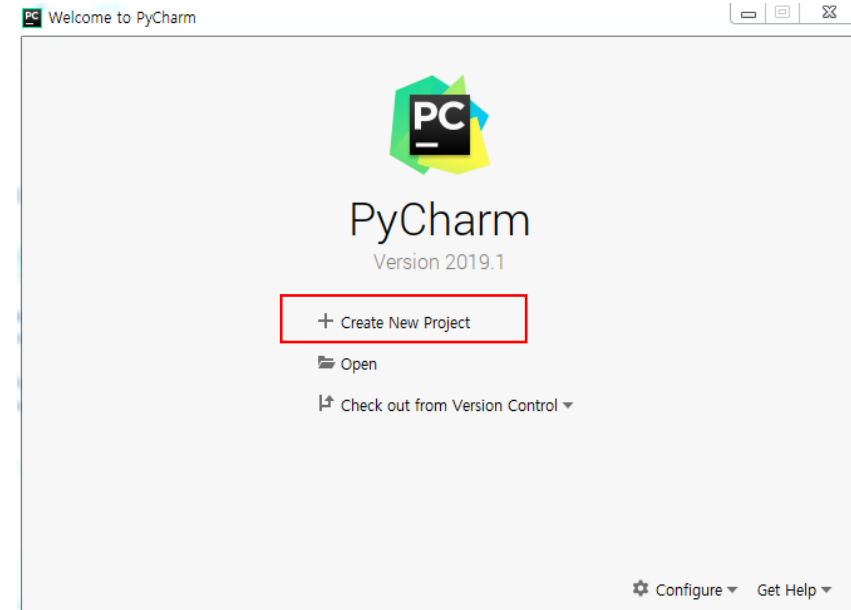
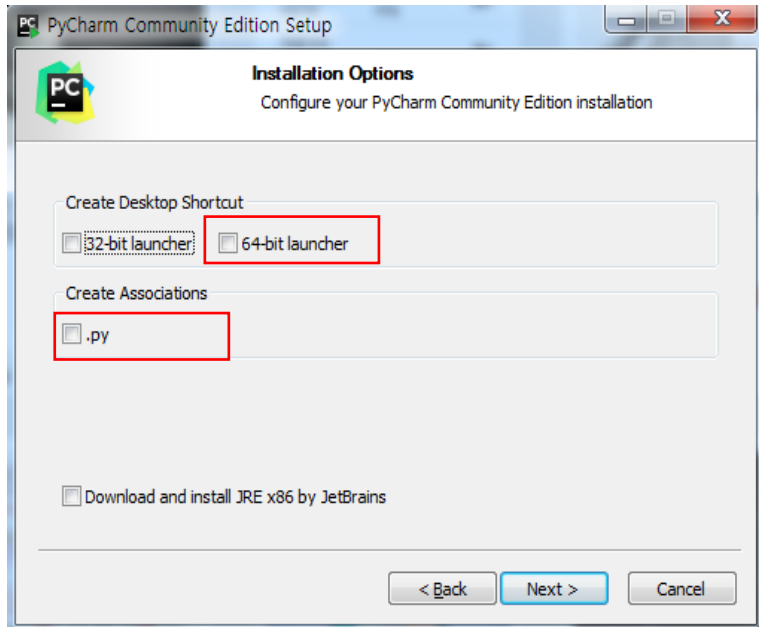
For pure Python development

DOWNLOAD

Free, open-source

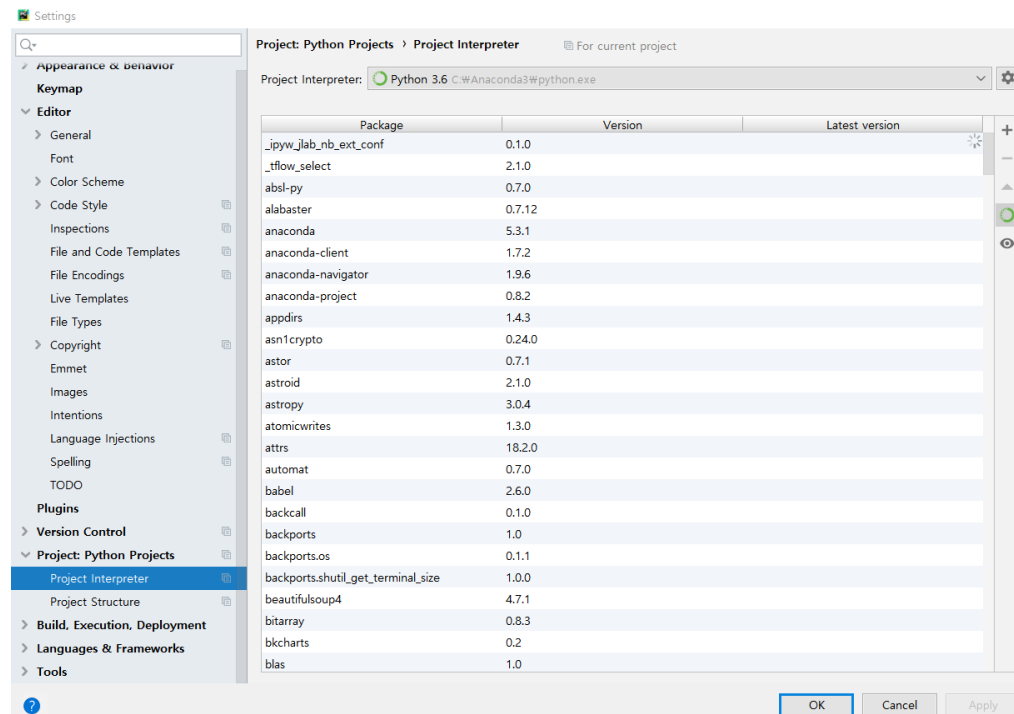
# 프로젝트 설정

- 64-bit launcher를 선택하고, 하단의 .py도 선택한다
- Create New Project 를 누르고 "PythonProjects"라고 입력한다
- 다른 이름도 가능하나, 이 이름이 프로젝트 최상위 경로가 된다



# 파이썬 연결

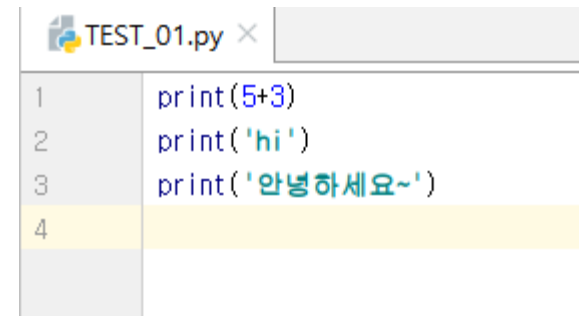
- 실행 뒤, Interpreter setting 에서 파이썬 엔진을 찾아준다
- File – Settings – Project – Project Interpreter에서
- C:\WAnaconda3\python.exe를 선택한다 (경로는 다를 수 있음)



# 파이썬 파일 설정

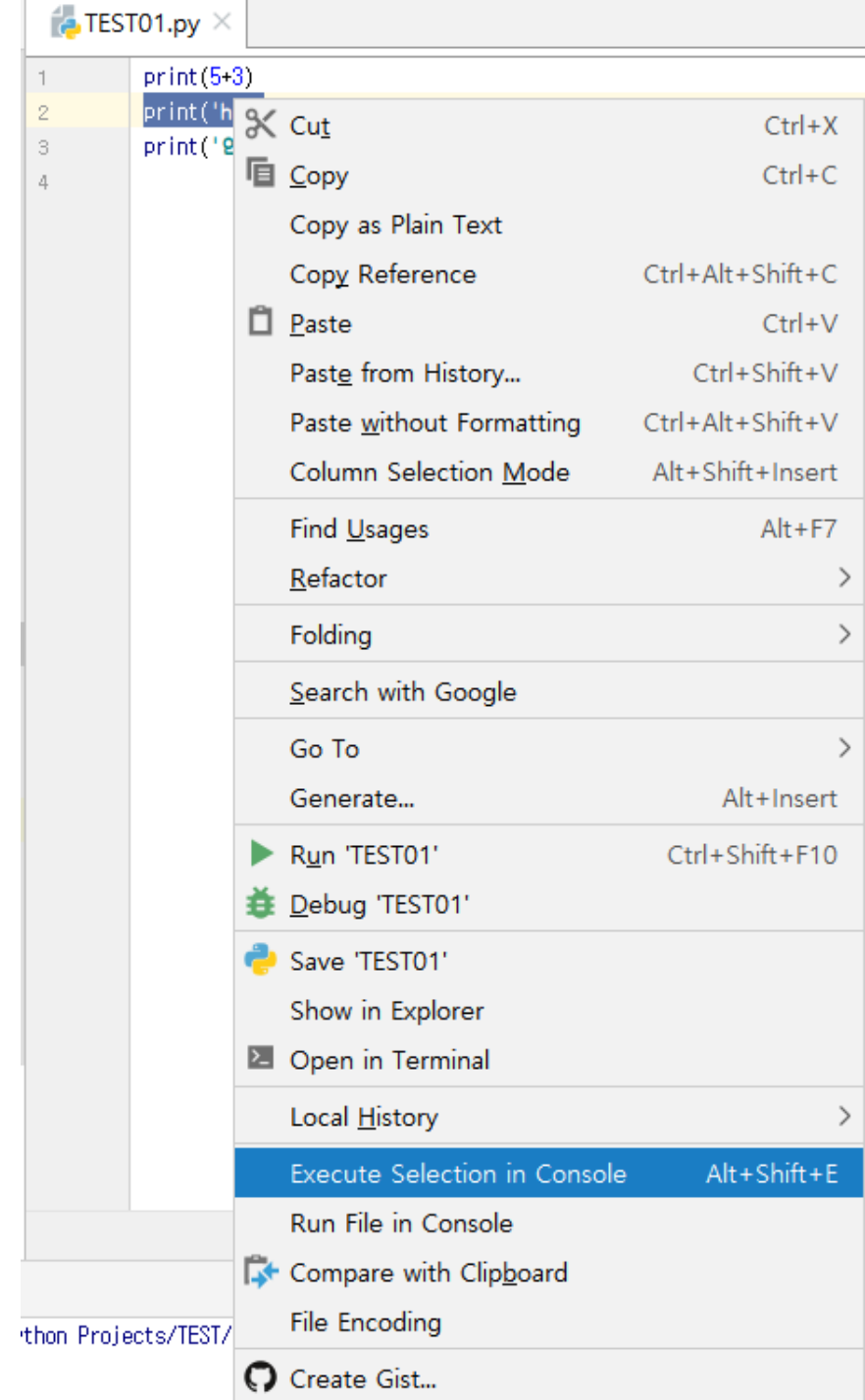
- Python Projects에서 우클릭하여 New-Directory 하여 TEST 폴더를 만든다
- TEST 폴더에서 우클릭하여 New-Python File 하여 TEST01 파일을 만든다
- 아래의 코드를 작성한 후 우클릭 Run 'TEST\_01' 하여 실행한다

```
• print(5+3)  
  print('hi')  
  print('안녕하세요~')
```



# 부분 실행

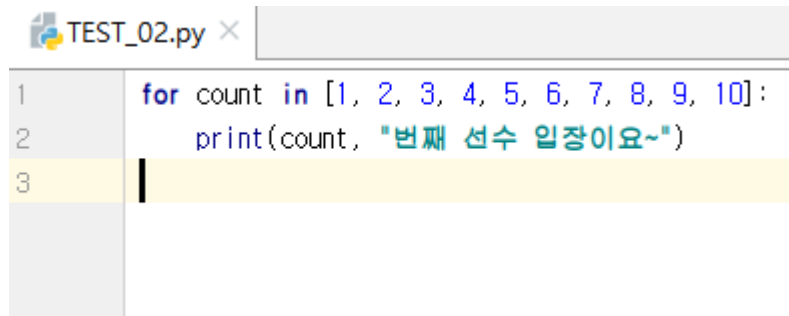
- 또는 원하는 부분을 블록으로 지정한 후,
- 우클릭 – Execute Selection in Console 명령어로
- 부분실행을 할 수도 있다



# 첫 번째 반복문

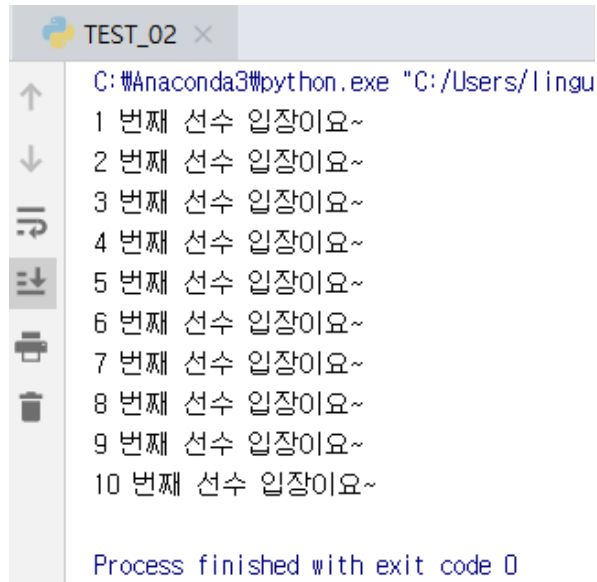
- 컴퓨터는 힘들어하지 않고 같은 일을 빠르게 잘 반복한다
- 반복문을 작성해본다

```
for count in [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]:  
    print(count, "번째 선수 입장이에요~")
```



A screenshot of a code editor window titled 'TEST\_02.py'. It contains a Python for loop that iterates over a list of numbers from 1 to 10. The code is as follows:

```
1 for count in [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]:  
2     print(count, "번째 선수 입장이에요~")  
3
```



A screenshot of a terminal window titled 'TEST\_02'. It shows the command prompt 'C:\#Anaconda3\python.exe "C:/Users/lingu' and the output of the script, which is ten lines of text: '1 번째 선수 입장이에요~' through '10 번째 선수 입장이에요~'. At the bottom, it says 'Process finished with exit code 0'.

```
C:\#Anaconda3\python.exe "C:/Users/lingu  
1 번째 선수 입장이에요~  
2 번째 선수 입장이에요~  
3 번째 선수 입장이에요~  
4 번째 선수 입장이에요~  
5 번째 선수 입장이에요~  
6 번째 선수 입장이에요~  
7 번째 선수 입장이에요~  
8 번째 선수 입장이에요~  
9 번째 선수 입장이에요~  
10 번째 선수 입장이에요~  
  
Process finished with exit code 0
```

Colab 사용하기

# Colab 개요 *Introducing Google Colaboratory*

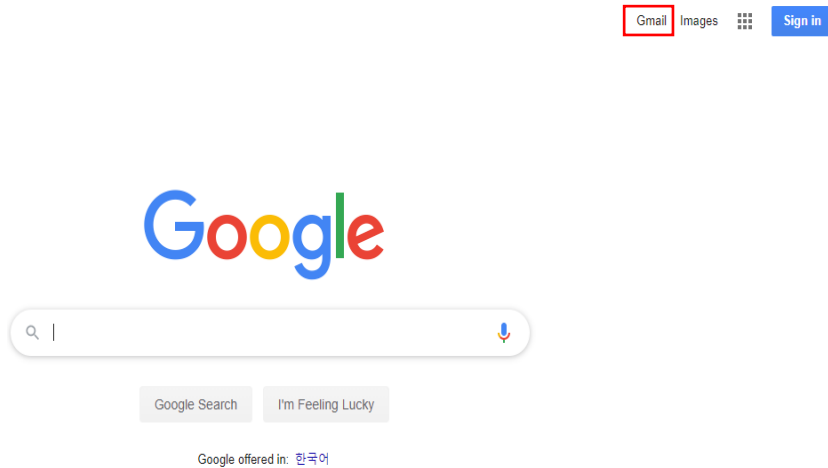
- Google에서 개발한 개발 플랫폼
- Google Drive와 Jupyter Notebook을 결합한 형태이다
- Colab은 Python으로 머신러닝 및 딥러닝을 수행할 준비를 갖추고 있다
- Python, Scikit-learn, Tensorflow, Keras, Matplotlib, ...
- Ubuntu 리눅스를 사용하며, GPU와 TPU까지도 사용이 가능하다



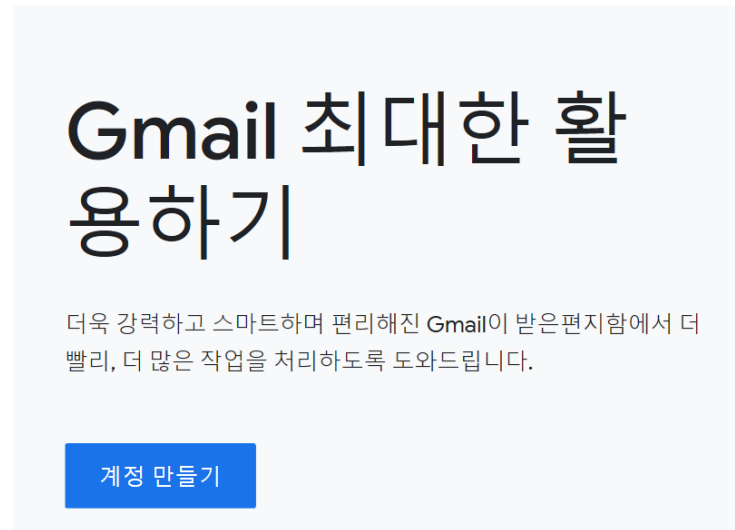
# 구글 계정 만들기

- Colab은 구글 드라이브를 사용하므로 먼저 구글 계정이 있어야 한다

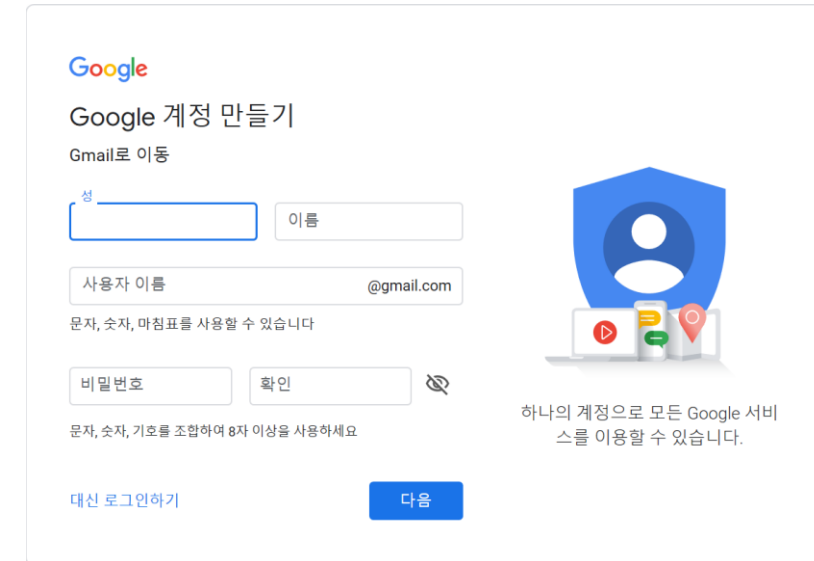
1. Gmail 버튼을 누른다



2. "계정 만들기" 버튼을 누른다

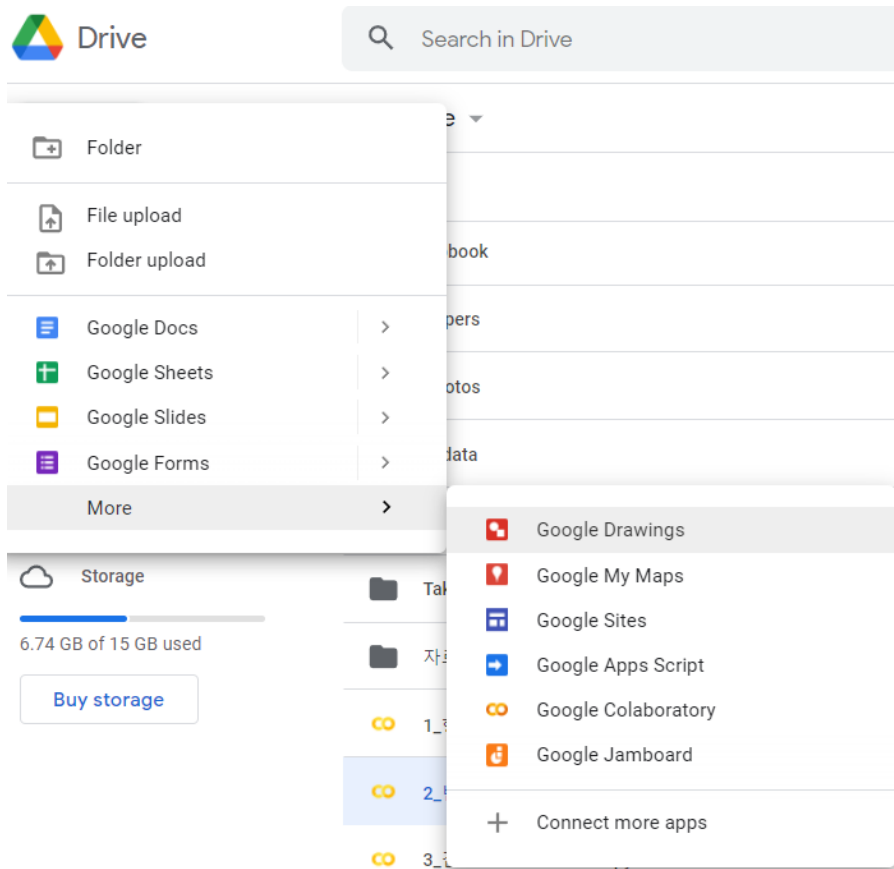


3. 정보를 입력하여 계정 만들기를 완료



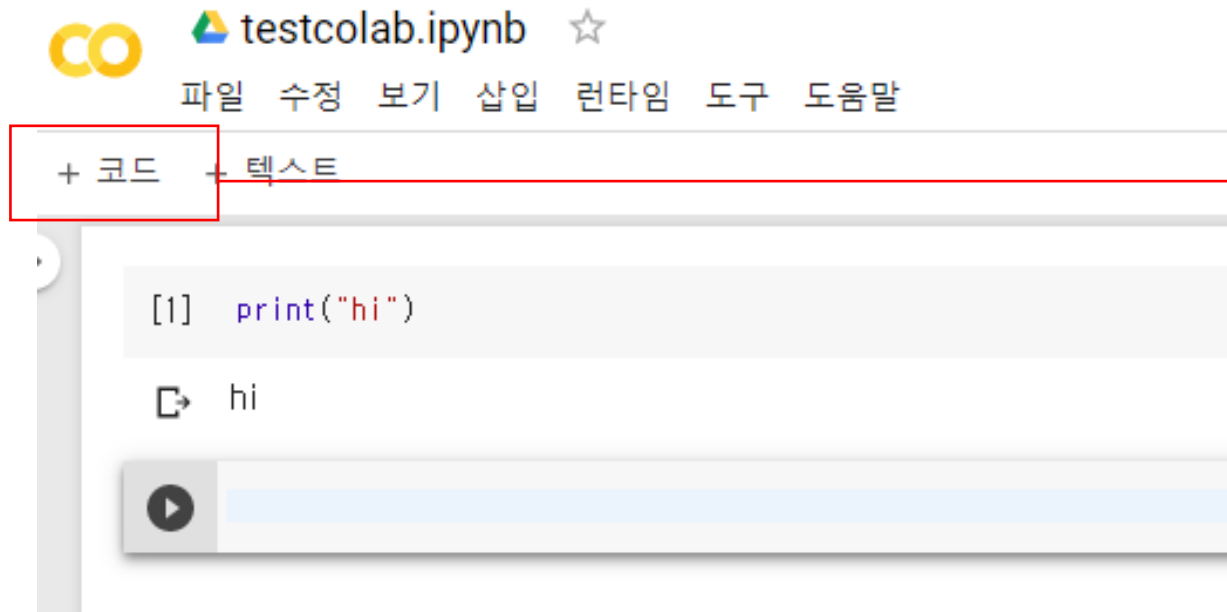
# Colab 검색

- 구글 드라이브에서 New – More – Google Colaboratory



# Colab 시작

- 이름을 적절히 설정하고 간단한 명령어를 넣는다
- `print("hi")`
- Shift-Enter 또는 왼쪽의 진행버튼을 눌러 실행한다



새로운 코드를 넣을 수 있다

# 구글 드라이브와 연결

- 다음의 코드를 입력한 뒤, 절차에 따라 진행한다
- verification code 입력 후, authorization code를 입력한다
- 일정 시간(약 90분) 이후에는 끊기므로, 매번 이 작업을 해주어야 한다

- from google.colab import auth
- auth.authenticate\_user()

} 생략가능

- from google.colab import drive
- drive.mount('/content/gdrive')

※ gdrive 명칭은 바꿀 수 있음 (예: cdrive, mydrive, drive, ...)

최종 화면

```
▶ from google.colab import auth
  auth.authenticate_user()

  from google.colab import drive
  drive.mount('/content/gdrive')
```

Go to this URL in a browser: <https://accounts.google.com/o/oauth2/auth>

Enter your authorization code:  
.....

Mounted at /content/gdrive

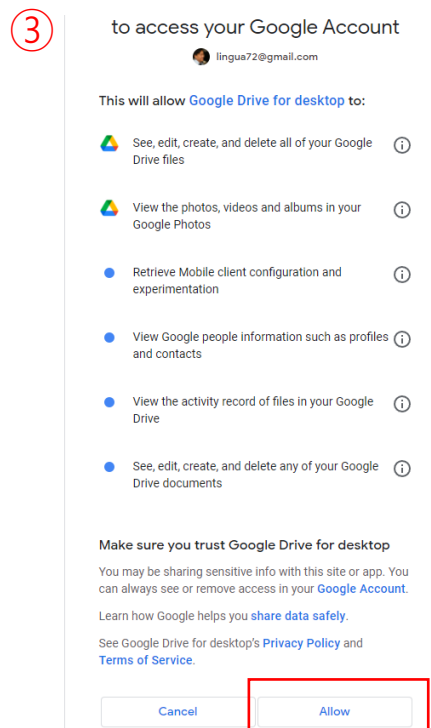
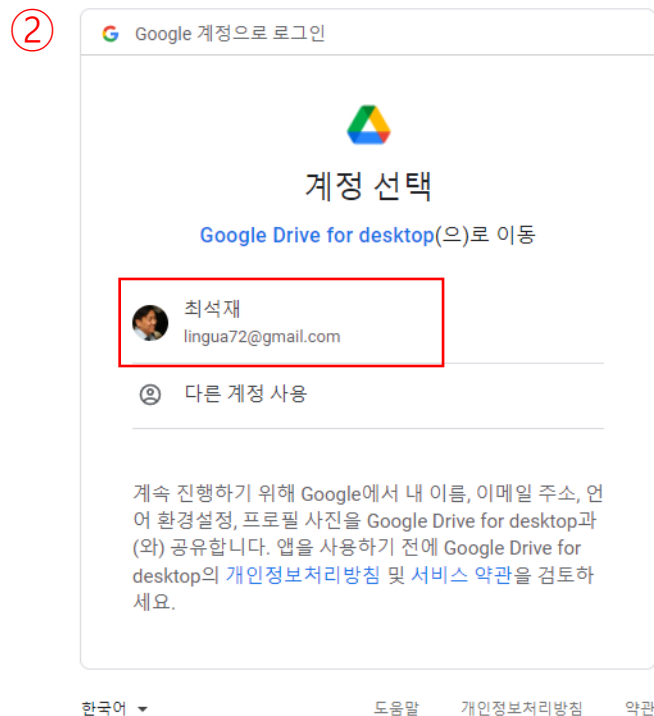
# 인증

## ① 노트북에서 Google Drive 파일에 액세스하도록 허용하시겠습니까?

이 노트북에서 Google Drive 파일에 대한 액세스를 요청합니다. Google Drive에 대한 액세스 권한을 부여하면 노트북에서 실행되는 코드가 Google Drive의 파일을 수정할 수 있게 됩니다. 이 액세스를 허용하기 전에 노트북 코드를 검토하시기 바랍니다.

아니요

Google Drive에 연결



④

```
from google.colab import drive
drive.mount('/content/gdrive')
```

Mounted at /content/gdrive

# 연결 확인

- 구글 드라이브에 있는 파일이 보이는지로 연결을 확인한다
- 구글 드라이브에 pytest\_basic 이라는 폴더를 만든다
- income.csv 파일을 pytest\_basic 폴더에 넣는다 (이미 있음)
- 리눅스 명령어로 pytest\_basic 폴더의 파일 리스트를 확인한다
- `!ls /content/gdrive/MyW Drive/pytest_basic/`

bank.csv	heightweight.csv	myfile2.txt	도시인구.xlsx
boston_housing.csv	income.csv	myfile3.txt	X_test.csv
california_housing.csv	insurance.csv	myfile.txt	X_train.csv
도시인구.csv	iris.csv	presidents_heights.csv	y_train.csv
electricity.CSV	mtcars.csv	sales_result.csv	

# 파일 읽기

- Pandas를 이용하여 파일을 읽는다
- import pandas as pd
- df = pd.read\_csv("/content/gdrive/My Drive/pytest\_basic/income.csv", encoding="cp949")
- print(df)

	age	workclass	education	educationNumber	#
0	39	State-gov	Bachelors	13	
1	50	Self-emp-not-inc	Bachelors	13	
2	38	Private	HS-grad	9	
3	53	Private	11th	7	
4	28	Private	Bachelors	13	
...	...	...	...	...	
32556	27	Private	Assoc-acdm	12	
32557	40	Private	HS-grad	9	
32558	58	Private	HS-grad	9	
32559	22	Private	HS-grad	9	
32560	52	Self-emp-inc	HS-grad	9	

	maritalStatus	occupation	relationship	race	gender	#
0	Never-married	Adm-clerical	Not-in-family	4	Male	
1	Married-civ-spouse	Exec-managerial	Husband	4	Male	
2	Divorced	Handlers-cleaners	Not-in-family	4	Male	

# 경로 변경 후 파일 읽기

- 편리한 접속을 위해 먼저 경로를 변경한 후 파일을 읽는다
- `%cd /content/gdrive/My Drive/pytest_basic/`
- `df = pd.read_csv("income.csv", encoding="cp949")`
- `print(df)`

	age	workclass	education	educationNumber	#
0	39	State-gov	Bachelors	13	
1	50	Self-emp-not-inc	Bachelors	13	
2	38	Private	HS-grad	9	
3	53	Private	11th	7	
4	28	Private	Bachelors	13	
...	...	...	...	...	
32556	27	Private	Assoc-acdm	12	
32557	40	Private	HS-grad	9	
32558	58	Private	HS-grad	9	
32559	22	Private	HS-grad	9	
32560	52	Self-emp-inc	HS-grad	9	

	maritalStatus	occupation	relationship	race	gender	#
0	Never-married	Adm-clerical	Not-in-family	4	Male	
1	Married-civ-spouse	Exec-managerial	Husband	4	Male	
2	Divorced	Handlers-cleaners	Not-in-family	4	Male	