파이썬 설치하기

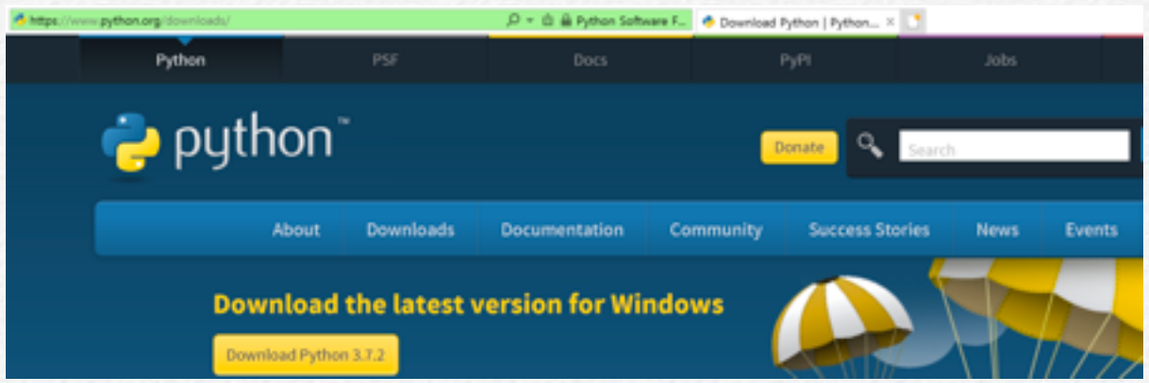
1

Python is a high-level programming language designed to be easy to read and simple to implement. It is open source, which means it is free to use, even for commercial applications. Python can run on Mac, Windows, and Unix systems and has also been ported to Java and .NET virtual machines. (https://techterms.com/definition/python)

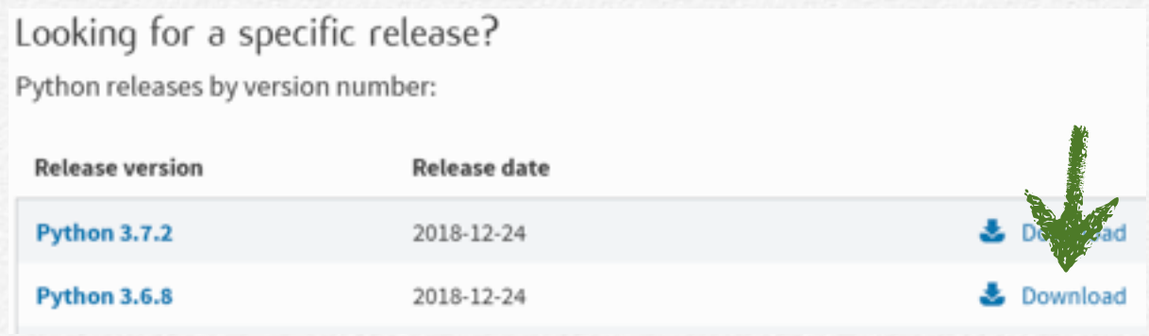
파이썬 설치하기

1) Python 다운로드 사이트에 접속한다.

https://www.python.org/downloads/

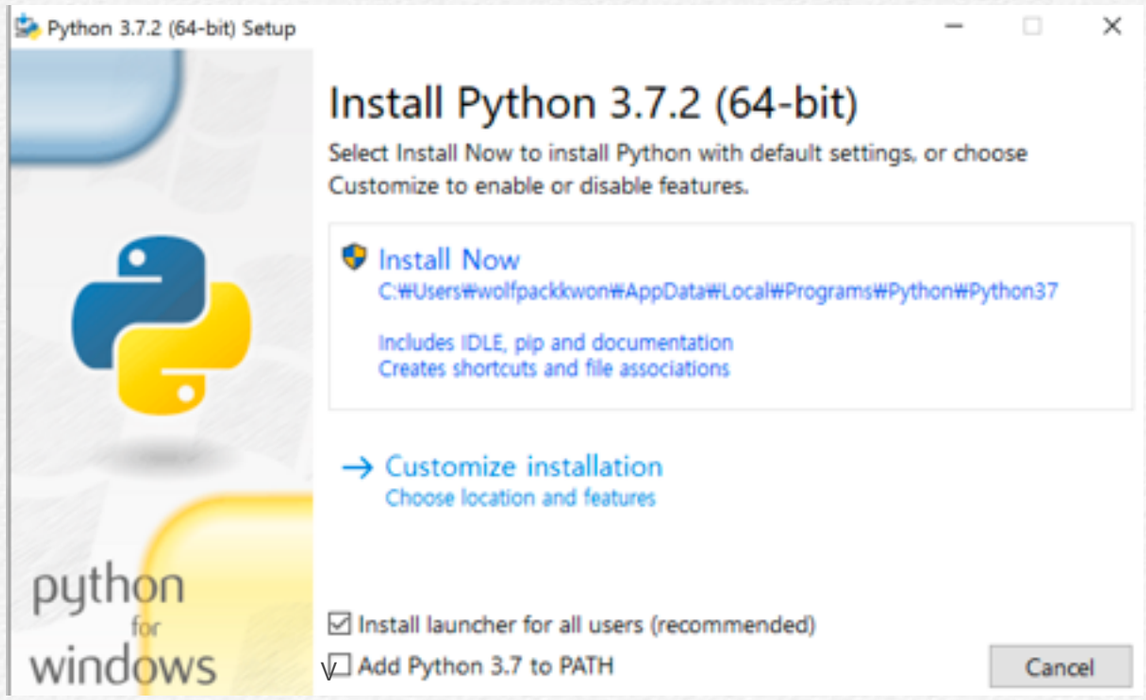


2) OS 시스템에 적합한 파일을 다운로드 한다.



3) 다운로드 된 파일을 실행하여 설치를 시작한다.

4) 설치 화면이 나타나면 Install Now 눌러 설치 시작한다.

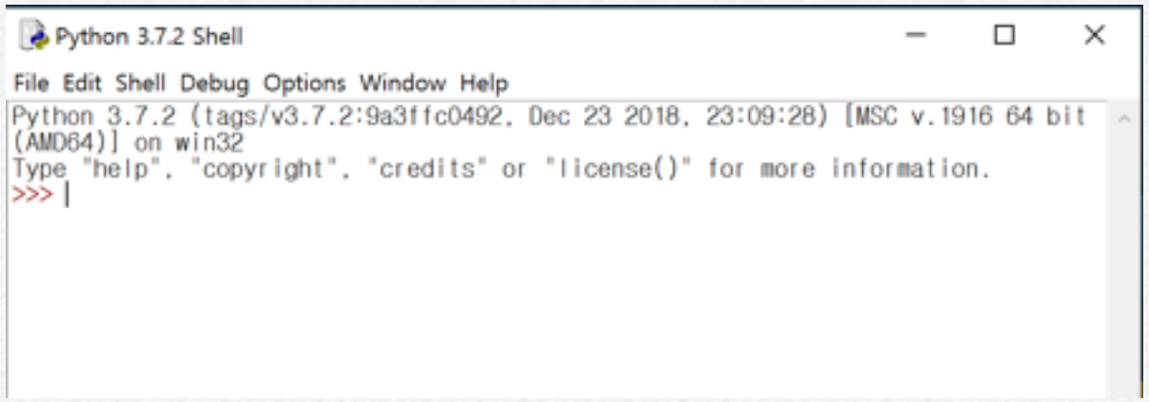


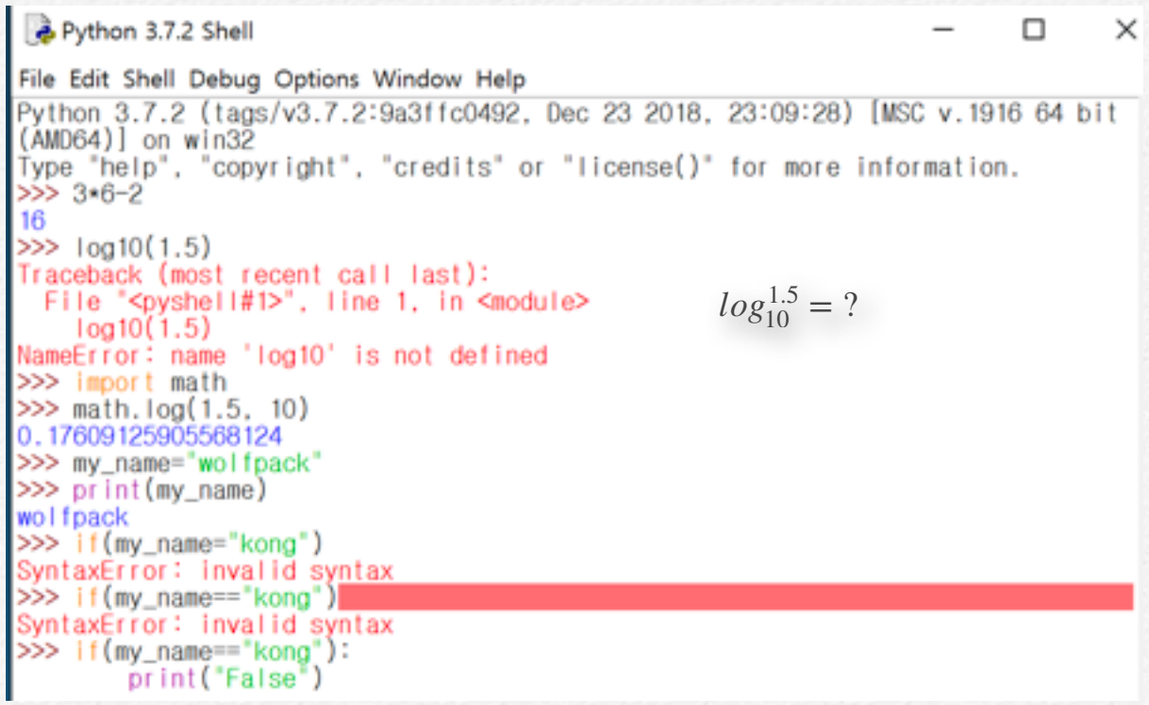
반드시 마지막 옵션 “Add Python * to Path”를 체크하세요. Jupyter notebook 설치 시 엄청난 어려움을 겪게 됩니다.

5) 프로그램 설치가 끝나면 시작화면 앱(프로그램)이 나타난다.

IDLE : Intergrated DevelLopment Envronment

6) IDLE (Python 3.7 64-bit)을 눌러 Python을 실행한다.

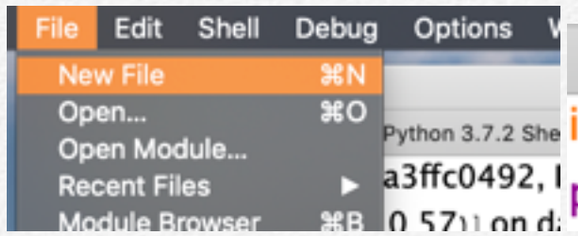


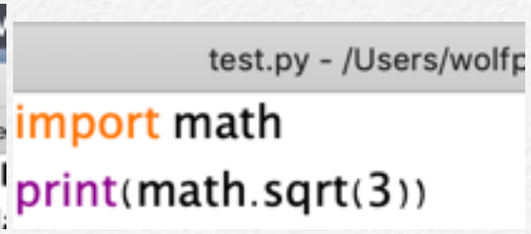


$$log_{10}^{1.5} = ?$$

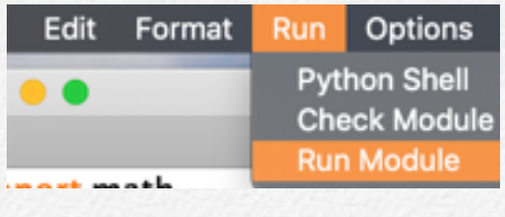
파일에서 하기

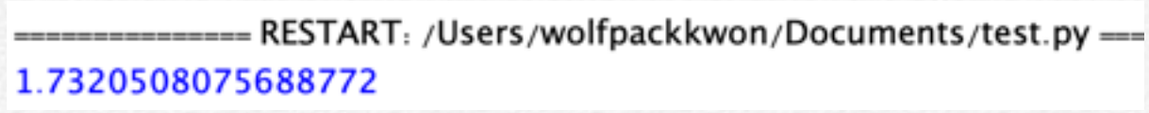
File => New File 새창을 열고 코드를 입력한다.





Run => Run Module 하여 코드를 “test.py”로 저장하면 창에 출력된다.





Shell 형식, 라인 에디터, Interactive mode로 프로그램이 실행된다.

파이썬은 ‘{’와 ‘}’ 기호 없이 들여쓰기로만 영역을 나타내기 때문에 항상 조심해서 들여쓰기해야 한다. IDLE에서 한 칸 공백을 주고(키보드의 스페이스 바를 누르면 됩니다) 명령어 코드를 작성하면 정상적으로 수행되지 않는다. 프로그래밍할 때 ‘Indent’라는 단어가 포함된 오류 메시지가 나타나면 띄어쓰기에 문제가 있는지 코드를 확인 해보기 바란다.

Jupyter Notebook은 오픈 소스 웹 애플리케이션으로 라이브 코드, 인터렉티브 위젯, 시각화, 나레이션, 이미지, 비디오 포함한 문서를 만들고 공유하도록 할 수 있다.
데이터 클리닝과 변형, 수치 시뮬레이션, 통계 모델링, 머신 러닝 등에 사용할 수 있으며 Python, R, Julia, Scala 등 다양한 프로그래밍 언어를 지원한다. 장점은 실시간으로 인터렉티브하게(**대화형**) 데이터를 조작하고 시각화할 수 있다.

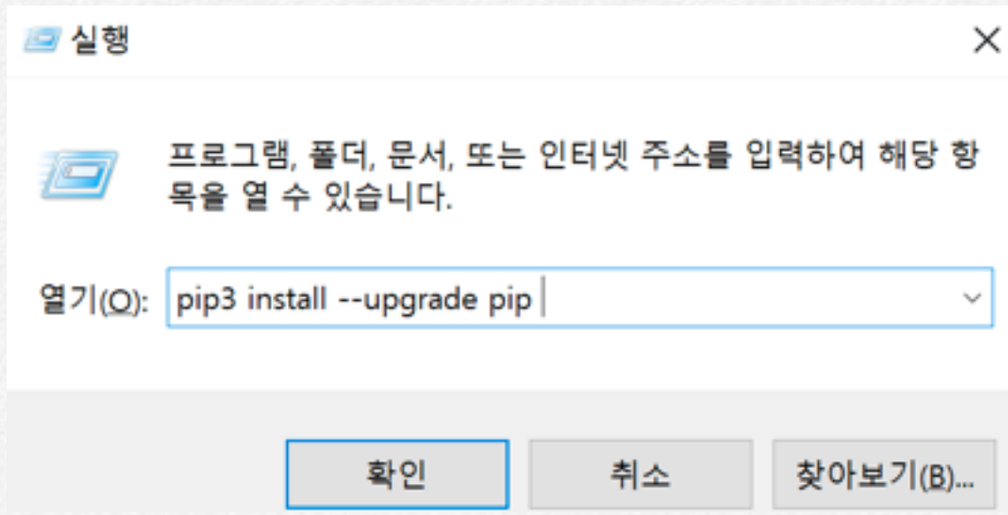
Jupyter Notebook 설치하기

Shell 모드는 프로그램 작성과 디버깅, 실행에 한계를 느낀다 Python을 쉽게 사용할 수 있으려면 Jupyter Notebook을 활용하면 됩니다. 많은 사람들은 Anaconda 활용하여 설치하는 것을 권하지만 굳이 그럴 필요는 없다.

1) Python 설치를 확인한다. (시작메뉴)

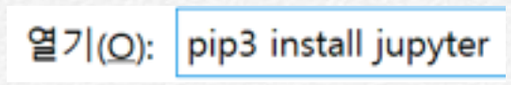
2) Python 업그레이드 확인한다. 윈도우키+R

윈도우키+R 실행 창을 열고 pip3 install --upgrade pip 입력하고 확인 버튼을 누른다.



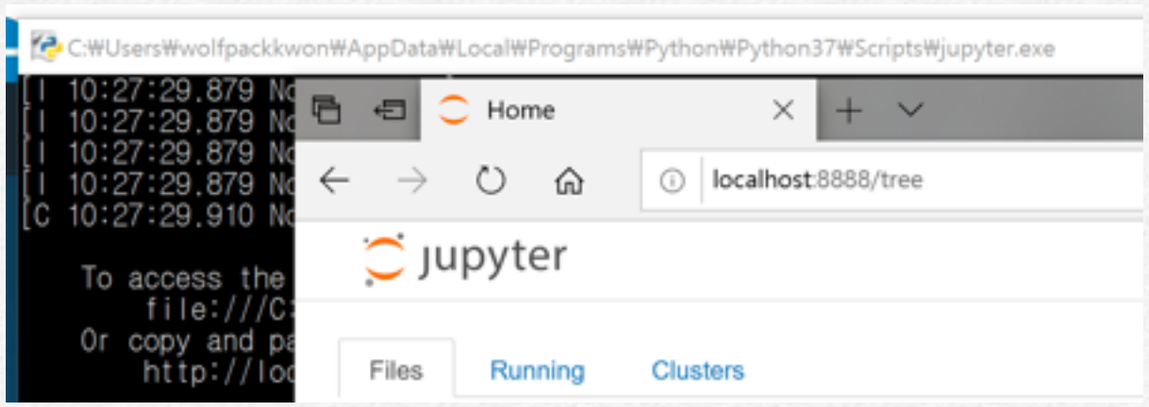
CMD (command) 창이 열리고 Python이 업그레이드 된다.

3) 이제 실행 창을 열고 (윈도우키+R) 주피터를 설치한다



4) 설치가 완료되면 실행 창에서 다음을 실행하면 Jupyter Notebook이 나타난다.

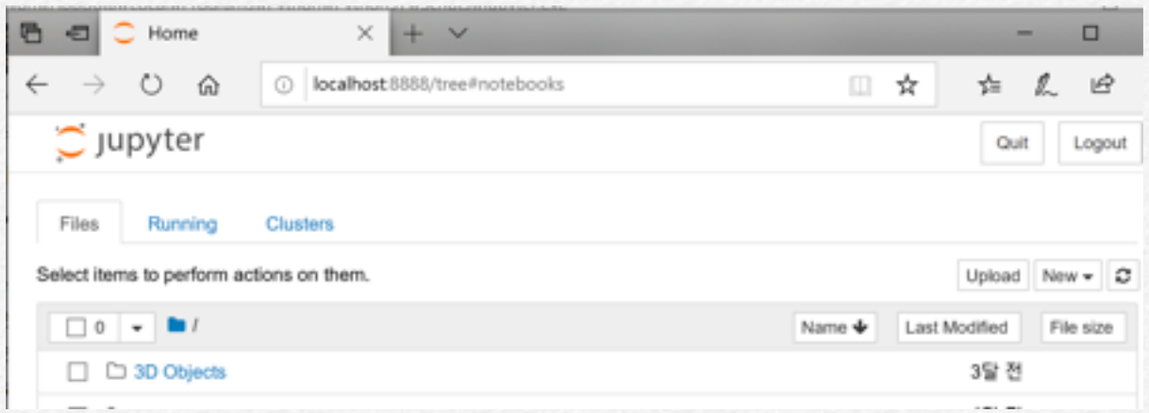
Jupyter는 웹(익스프로러) 상에서 실행된다.



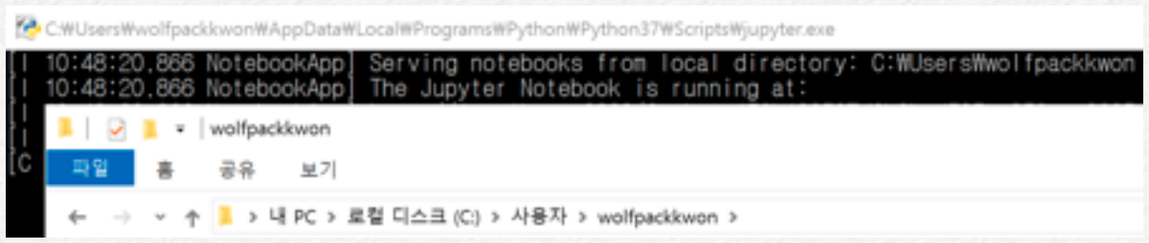
Jupyter 맛보기

Jupyter 초기화면

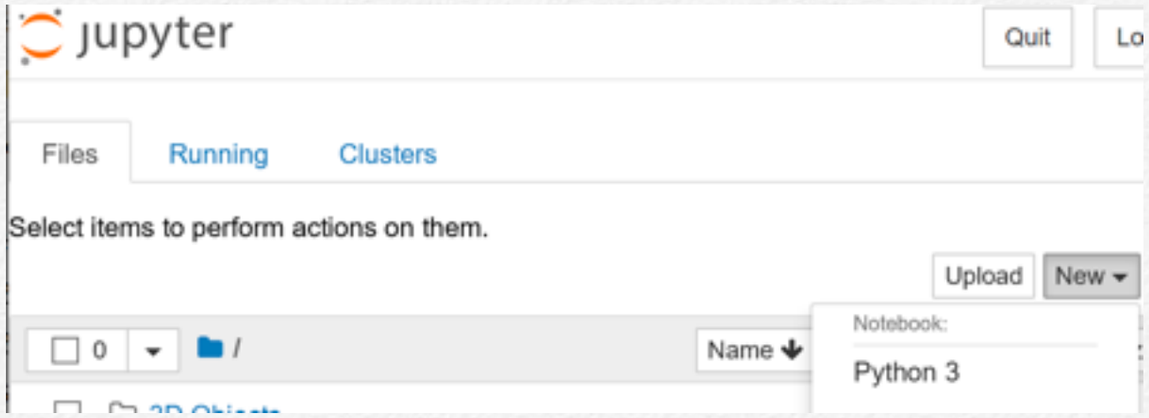
Jupyter 실해하면 커맨드(command, 옛날 도스 창과 동일) 창과 함께 인터넷 익스플로러에 실행된다. 이를 Jupyter Dashboard라 한다.



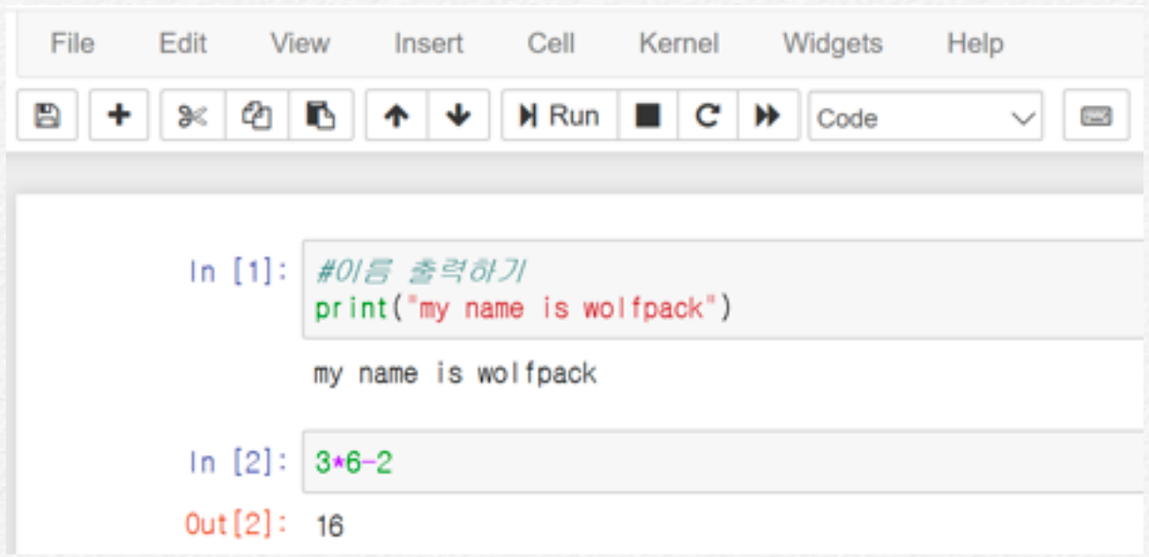
Files은 내 컴퓨터 어디 폴더를 가르키나? 커멘드 창에 “serving notebook from local directory” 정보가 나타난다.



새로운 Python 파일을 작성하기



In [ ]: 에 적절한 코드를 작성하고 Run 을 클릭하면 실행되고 결과가 바로 아래 출력된다. #은 설명문 코드로 실행되지 않으며 코드의 내용을 설명하는데 사용된다. In[숫자]는 코드 일련번호이며, 박스 안의 코드는 한 번에 모두 실행된다.



코드에 오류가 있는 경우 결과 창에 오류 메세지가 출력된다. 동일 코드 창에 수정하거나 다음 코드 창에 완전한 코드를 입력한 후 실행하면 된다.

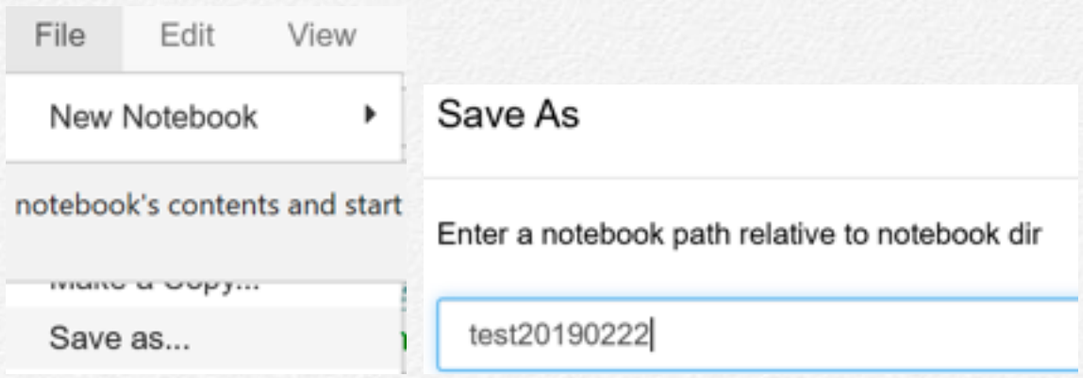
```
In [3]: log(1.5,10)

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-3-908403c7b898> in <module>
----> 1 log(1.5,10)

NameError: name 'log' is not defined
```

```
In [5]: import math
        math.log(1.5,10)
Out[5]: 0.17609125905568124
```

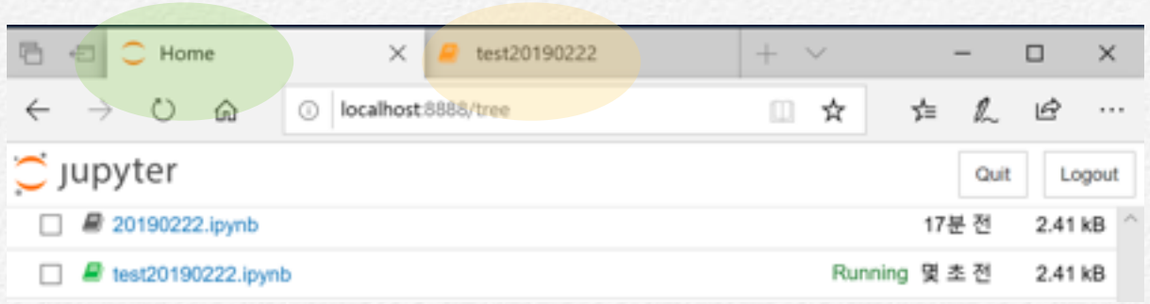
코드 실행 Hot Key :
=> 라인(행) 실행 : Crtl+Enter
=> 전체 In[] 실행 : Alt+Enter

코드 저장하기

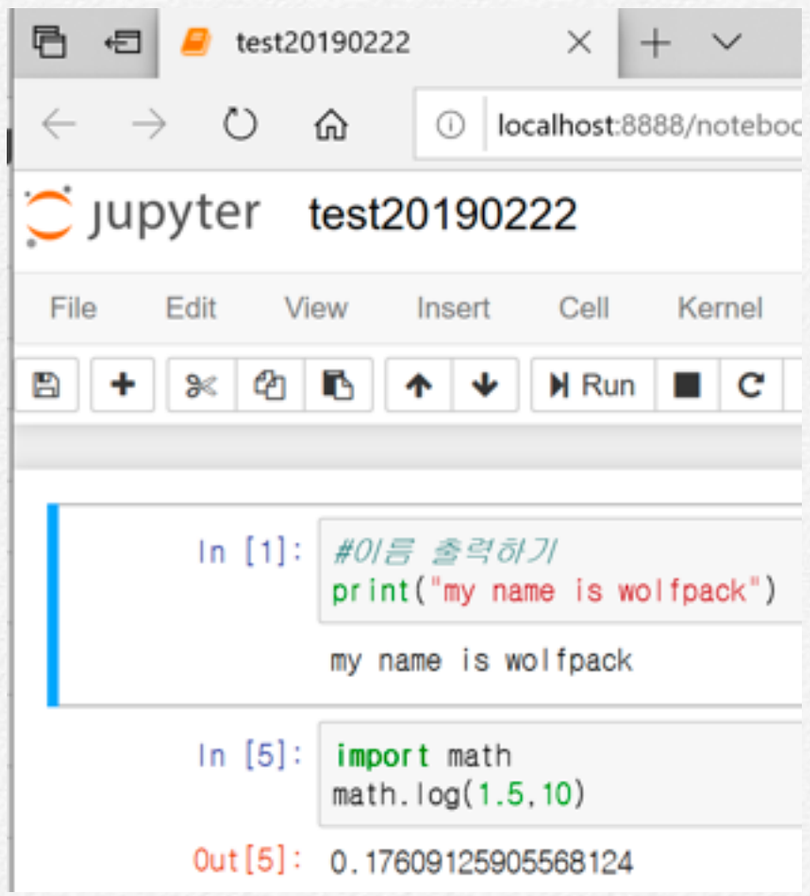
코드를 저장하려면 File -> Save as 선택한 후 Save As 창에 파일이름을 적으면 된다. 저장하지 않더라도 Untitled.(ipnyb)확장자로 매 분 자동 저장되므로 걱정할 필요 없다.(Jupyter 종료 후에도 저장되어 있어 다시 시작하면 다시 사용 가능함)



파일이 저장되는 순간 Untitled 탭이 저장된 파일이름 탭으로 바뀌면서 Home 탭에서 파일이 저장된 것을 확인할 수 있다. (확장자명*.ipynb)



저장된 코드를 수정하기 위하여 불러오려면 Home 탭에서 해당 파일을 클릭하면 새로운 탭에서 자동 업로드 된다.



파이썬 개념

파이썬은 상이한 프로그램 로직을 지원하는 다차원 패러다임 프로그램으로 Object-Oriented Programming (OOP) 객체 지향 프로그램으로 1991년 귀도 반 로섬(Guido van Rossum, 2013년까지는 구글(Google)에서 일했고 최근에는 드롭박스(Dropbox)로 회사를 옮겼습니다)이 최초(자신이 좋아하는 코미디 프로그램인 “Monty Python’s Flying Circus”) 만듬

객체지향이라는 개념이 나타나기 이전의 프로그래밍 방법에서는 프로그램이 어떤 일을 하고나서, 그 다음엔 어떤 일을 하고, 또 그 다음엔 뭘 하라는 식으로 컴퓨터가 해야 할 일을 일련의 순서로 프로그래밍 작업을 하였다.

그런데, 객체지향 프로그래밍에서는 프로그램을 작성할 대상이 되는 실제 세계의 사물(객체)을 그대로 표현하고, 그것들이 어떻게 움직이는지 정해주고 나서야 비로소 그 객체들에게 일을 시킨다. 객체지향 프로그래밍을 잘 사용하면 보다 좋은 프로그램을 빨리 만들 수 있고, 나중에 수정하기도 편해진다고 합니다.

객체 object 개념

객체는 속성 attribute, 행동 behavior으로 구성되어 있다.

앵무새는 객체

- 학명, 나이, 색상은 속성 attributes
- 노래하기, 춤추기는 행동이다.

파이썬의 OOP 재생산 가능 코드에 집중한다. DRY (Don't Repeat Yourself)

객체와 클래스 class

객체는 단순히 데이터 (변수)와 메소드(함수)의 모음입니다. 그리고 클래스는 객체의 청사진입니다.

클래스는 객체를 위한 청사진이다.

클래스는 집의 스케치 (원형)라고 생각할 수 있다. 여기에는 바닥, 문, 창문 등에 대한 모든 세부 정보가 포함되어 있습니다. 이러한 설명을 토대로 우리는 집을 짓게 되므로 하우스가 객체이다.

많은 주택은 청사진(설계도)으로부터 만들 수 있기 때문에, 우리는 한 클래스에서 많은 객체를 생성 할 수 있다. 객체는 클래스의 인스턴스라고도 하며, 이 객체를 만드는 프로세스를 인스턴스화 라고 합니다.

클래스 정의

```
class student:              #클래스 정의
    "student duty"  #docstring 으로 클래스 재목
    min_hr=8            #변수 a 값 지정
    def work(self):   #함수 정의
        return "Study Hard" #work 함수 내용
```

- 클래스 내에 정의된 함수를 메소드(method)라 하고 안에 정의된 것을 매개변수라 한다. (self)는 자기 호출 매개변수이다. 그러므로 student 클래스 내 함수(메소드)의 호출은 stdent.work가 된다.
- “def work(self, var1) :”으로 정의된 메소드 호출은 student.work(var1)이다.
- 클래스 내에 정의된 변수나 메소드의 이름은 클래스이름.메소드이름이 된다.
- 클래스 내 변수는 클래스이름.변수명 만으로 사용가능하나 함수는 객체를 정의해야 사용 가능하다.

```
In [10]: print(student.min_hr) #내부 변수 값 출력
         8

In [11]: print(student.work) #함수 내용 출력
         <function student.work at 0x10636a158>

In [12]: print(student.__doc__) #함수 제목 출력
         student duty
```

객체 정의 : 인스턴명=class()

work 메서드는 student라는 클래스가 하는 행동을 정의하고 있다. 다음과 같이 객체를 정의하여 사용하면 된다.

```
In [16]: you=student()
         you.work()
Out[16]: 'Study Hard'
```

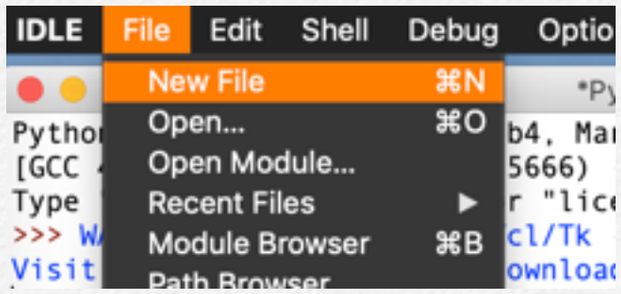
(함수 사용과 유사하다)

모듈 Module 만들기

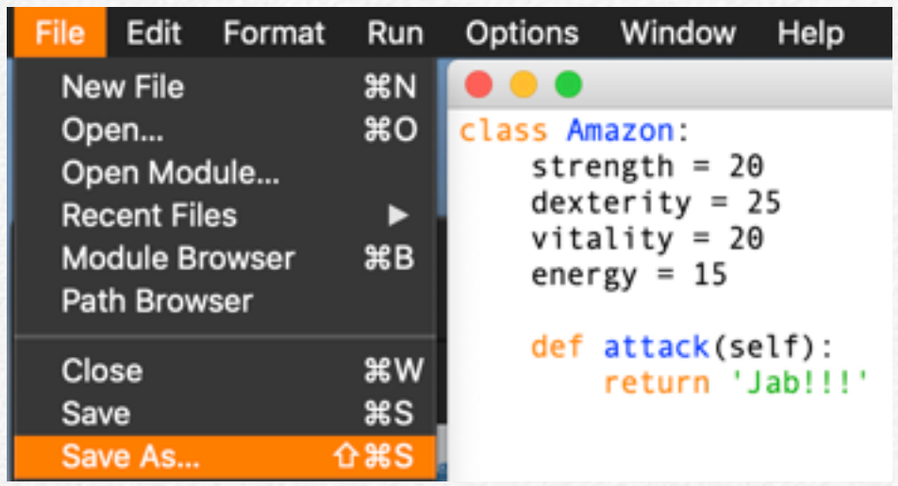
모듈이란 함수나 변수 또는 클래스 들을 모아 놓은 파일이다. 모듈은 다른 파이썬 프로그램에서 불러와 사용할수 있게끔 만들어진 파이썬 파일이라고도 할 수 있다. 우리는 파이썬으로 프로그래밍을 할 때 굉장히 많은 모듈을 사용한다. 다른 사람들이 이미 만들어 놓은 모듈을 사용할 수도 있고 우리가 직접 만들어서 사용할 수도 있다.
모듈 modele 만들기

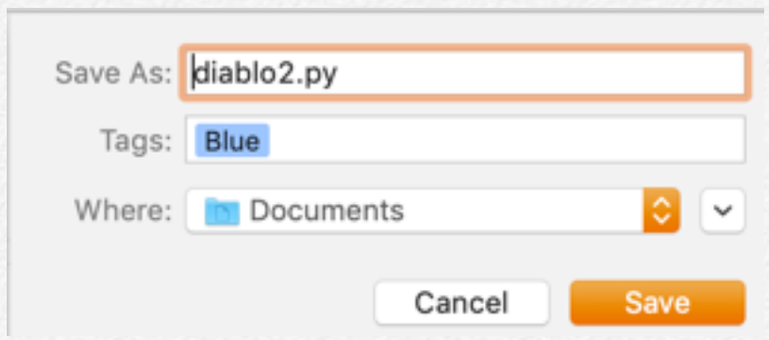
Jupyter에서는 어디서 모듈(패키지와 유사개념)을 만드는지 알 수 없어 파이썬 shell에서 만들고 Jupyter에서 사용하겠다.

(1) 파이썬 Shell(IDLE) 에 접속한 후 File -> New File 을 연다.



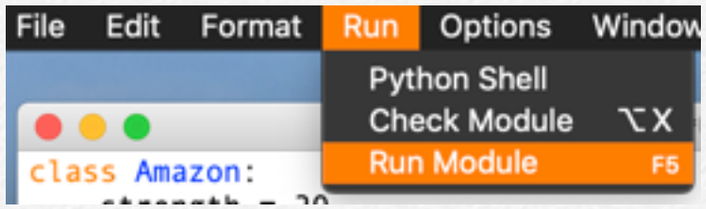
(2) File => Save As… 메뉴를 선택하여 “diablo2.py”로 저장한다.





Documents 폴더에 저장되어 있다. 이것을 기억하기 바란다.

(3) RUN-> Run Modlue을 실행하여 코드를 모듈화 한다.



모듈 사용하기

이제 다시 Jupyter로 돌아가 모듈을 사용해 보자.

```
In [4]: cd
        /Users/wolfpack

In [6]: cd ./Documents
        /Users/wolfpack/Documents

In [7]: import diablo2
```

- Jupyter가 실행된 폴더를 보기 위하여 cd 를 입력한다.
- documents는 현재 폴더 아래 있으므로 cd(change directory) 명령어로 현재 폴더를 변동한다.

import 사용하여 모듈 diablo2를 불러온다. 모듈 내의 함수를 사용하는 경우 모듈명.함수 풀네임을 적어줘야 한다.

```
In [10]: import diablo2

In [11]: king=diablo2.Amazon()
         kong=diablo2.Amazon()
```

```
In [16]: king.strength
Out[16]: 20

In [14]: kong.energy
Out[14]: 15

In [15]: kong.attack()
Out[15]: 'Jab!!!'
```

상속 inherite

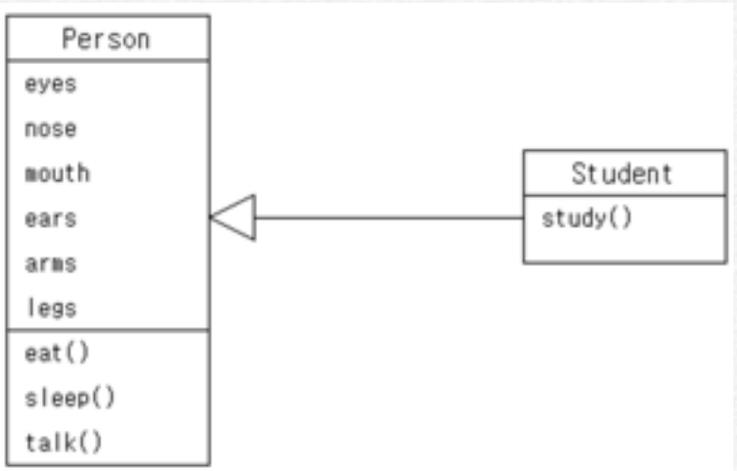
객체지향 프로그래밍의 핵심적인 개념 가운데 하나인 상속(inheritance)입니다. 상속이란 어떤 클래스가 다른 클래스의 성질을 물려받는 것을 말하지요.

어떤 클래스를 만들 때 처음부터 모든 것을 새로 만들 필요 없이, 핵심적인 성질을 갖고 있는 다른 클래스로부터 상속을 받아서 조금만 손을 보면 쓸만한 클래스를 만들 수 있답니다. [이 부분은 https://wikidocs.net/86 강의 정리하였음]

```
In [20]: class Person:
             #변수 값 정의
             eyes = 2
             nose = 1
             mouth = 1
             ears = 2
             arms = 2
             legs = 2
             #함수 정의
             def eat(self):
                 print('얌냠...')

             def sleep(self):
                 print('쿨쿨...')

             def talk(self):
                 print('주절주절...')
```

```
In [22]: class Student(Person):
             def study(self):
                 print('열공열공...')
```

```
class Person:
    #변수 값 정의
    eyes = 2
    nose = 1
    mouth = 1
    ears = 2
    arms = 2
    legs = 2
    #함수 정의
    def eat(self):
        print('얌냠...')

    def sleep(self):
        print('쿨쿨...')

    def talk(self):
        print('주절주절...')

# Person 클래스를 상속받음
class Student(Person):
```



위의 Student 클래스는 Person 클래스를 상속받았습니다. 우리가 Student 클래스를 눈, 코, 입부터 하나하나 다시 만들어 주지 않더라도 Person의 성질들을 모두 물려받아서 갖게 된 것이죠. 우리는 여기에 study라는 메쏘드만 하나 더 써주어서 우아하게 마무리를 했습니다.

굳이 상속을 받지말고 스크립트를 복사해서 붙이면 되지않느냐구요? 물론 그렇게 해도 가능합니다. 하지만 나중에 사람과 학생 클래스에 '옷 색깔'이라든지, '싸우다' 같은 것들을 추가하고 싶어진다면, 그 때마다 사람 클래스와 학생 클래스를 각각 수정해야 되겠지요.

```
In [32]: kong=Person()

In [33]: kong.nose
Out[33]: 1

In [34]: kong.sleep()
         쿨쿨...

In [36]: kong=Student()

In [37]: kong.study()
         열공열공...
```

파이썬 사용자 설명서 및 색인

https://docs.python.org/ko/3.7/contents.html

https://docs.python.org/ko/3/genindex.html

계산기 모듈 만들기 : 삼각형 면적 구하기

정의된 클래스 이름 Area, 함수(매소드) 이름은 cal, 매개변수는 (self, base, height) 3개이지만 실제 self는 클래스 이름 상속하는 역할을 하는 것이므로 실제 매개변수는 2개이다.

In [18] => 클래스 속성이 A에 저장된다.

In [19] => A.변수명.(매개변수 지정) 클래스 내 함수, 설정된 매개변수 값 (base=3, height=7)이 입력되고 print()에 의해 콘솔 출력된다.

```
In [16]: class Area:
             def cal(self,base,height):
                 print("Triangle Area=",base*height/2)

In [18]: A=Area()

In [19]: A.cal(3,7)
         Triangle Area= 10.5
```

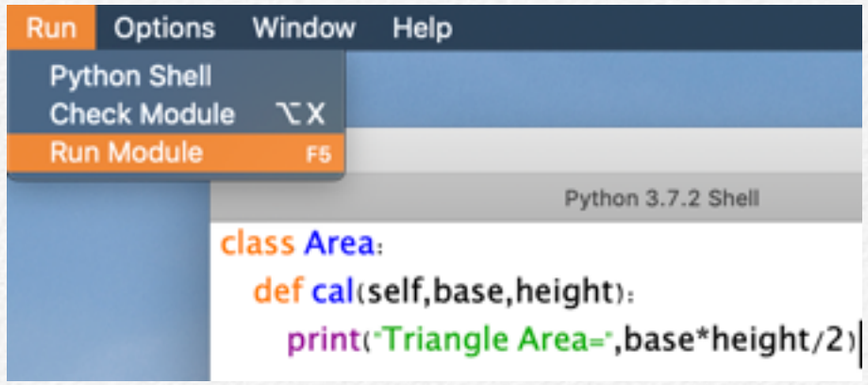
모듈 만들 필요가 없다면 클래스 만들지 말자

클래스를 다시 사용하지 않는 경우에는 모듈을 만들 필요는 없고 그냥 함수만 사용하면 된다.

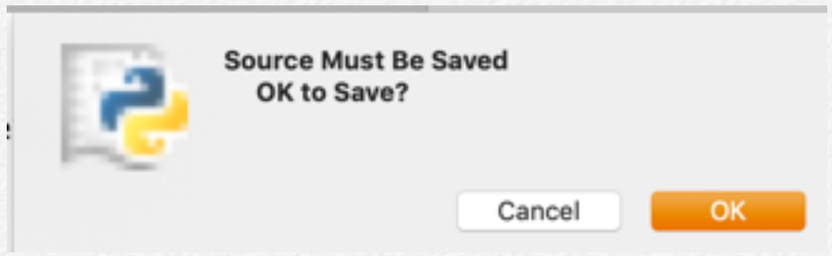
```
In [3]: def cal2(base,height):
            print('Area',base*height/2)

In [4]: cal2(3,7)
        Area 10.5
```

모듈 만들기

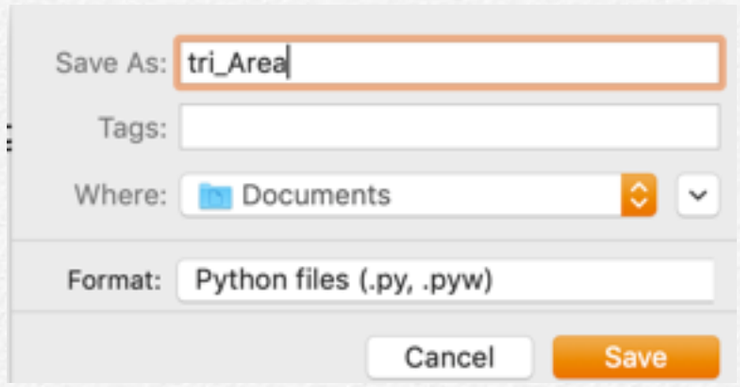
(1) 파이썬 shell에 클래스 코드를 입력하고 모듈 만들기를 한다.



(2) 모듈 만들기 전에 우선 저장이 필요하다.



(3) 폴더 Documnets에 tri_Area 모듈로 저장된다. 확장자는 .py이다.



(4) 모듈 저장과 동시에 저장 내용이 Shell에 나타난다.

```
>>>
========== RESTART: /Users/wolfpackkwon/Documents/tri_Area.py ==
>>>
```

(5) Jupyter 열고 다음 절차에 의해 모듈을 사용한다. Jupyter에서 Dos 명령어 사용이 가능한데, ls(list)는 현재 디렉터리를 보여주는 것이더 cd(change directory)는 폴더 변경이다. “cd ..” 현재 바로 상위 폴더로 이동이다. 폴더 documents에 tri_Area 모듈이 있으므로 그 곳으로 이동한다.

```
In [22]: ls
         Applications/                    Music/
         Applications (Parallels)/        Musics/
         Area.ipynb                       Parallels/
         Box Sync/                        Pictures/
         Desktop/                         Public/
         Documents/                       SASUnivers

In [23]: cd documents
         /Users/wolfpackkwon/Documents
```

(6) 모듈을 import에 의해 불러오고 사용하면 된다.

```
In [25]: import tri_Area

In [26]: x=Area()
         x.cal(3,7)
         Triangle Area= 10.5
```

내장 모듈 사용하기

내장 모듈 module이란?

파이썬은 객체 지향 프로그램인데, 주요 작업을 시행하기 위한 다양한 함수를 모듈(클래스와 객체)화 하여 빌트인(내장) 되어 있다.

(예) 수학 로그 연산 : math

(예) 통계분포함수 모듈 : numpy　　(예) 통계 그리기 함수 : matplotlib

내장 모듈 사용하기

제곱근을 구하는 함수 sqrt()은 모듈 math에 들어 있어 이를 불러온 후 사용해여 한다. 사용할 경우애는 모듈명.함수명을 사용해야 한다.

```
In [108]: sqrt(3)
          ---------------------------------------------
          NameError                                 Trac
          <ipython-input-108-2f71726bed4c> in <module>()
          ----> 1 sqrt(3)

          NameError: name 'sqrt' is not defined

In [109]: import math
          math.sqrt(3)
Out[109]: 1.7320508075688772
```

모듈 이름이 긴 경우 사용하기 어렵다면 “as 약어” 사용하면 된다. 그러면 numpy 대신 ny 만 사용해도 된다.

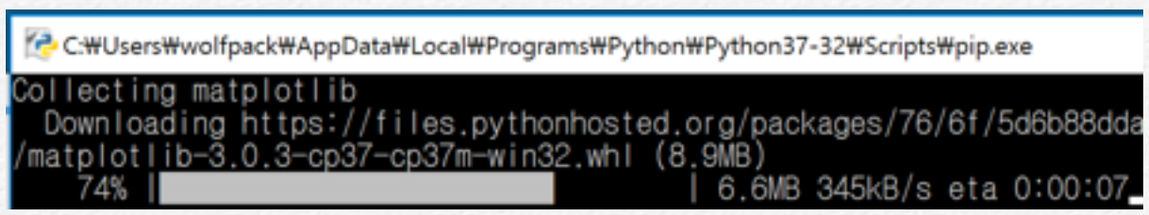
```
In [112]: import numpy as ny
          ny.random.uniform(0,1,10)

Out[112]: array([0.12730179, 0.60379732, 0.37801222,
                 0.99111601, 0.47682653, 0.55845393,
```

내장되지 않은 모듈 파이썬에 설치하기

파이썬 내장되지 않은 모듈(matplotlib)을 사용하려면 다음 오류 메세지가 나타난다.

```
In [107]: import matplotlib as plt

----------------------------------------------------
ModuleNotFoundError                       Traceb
<ipython-input-107-12ca746db26c> in <module>()
----> 1 import matplotlib2 as plt

ModuleNotFoundError: No module named 'matplotlib
```

윈도우에서 윈도우키+R을 눌러 아래 창이 나타나면 pip install matplotlib 입력하고 엔터 키를 치면 모듈이 설치된다.



설치가 완료되면 사용 가능 =>

```
In [114]: import matplotlib as mpl
```

예제 : 정규분포 난수 1,000개 생성 후 확률분포함수 그리기

```
In [129]: import numpy as np
          import matplotlib.pyplot as plt

          # Generate random normally distributed data
          data=np.random.randn(10000)

          # Histogram
          heights,bins = np.histogram(data,bins=50)

          # Normalize
          heights = heights/float(sum(heights))
          binMids=bins[:-1]+np.diff(bins)/2.
          plt.plot(binMids,heights)
```