

강의 1-1  
파이썬의 소개

강영민  
동명대학교 게임공학과



## 1.4 프로그래밍이란 무엇인가

- **프로그램**program

- 컴퓨터가 해야할 일을 미리 기록해 놓은 작업 지시서 같은 것
- 우리가 사용하는 파워포인트나 카카오톡과 같은 것들이 모두 프로그램

- **프로그래밍**programming

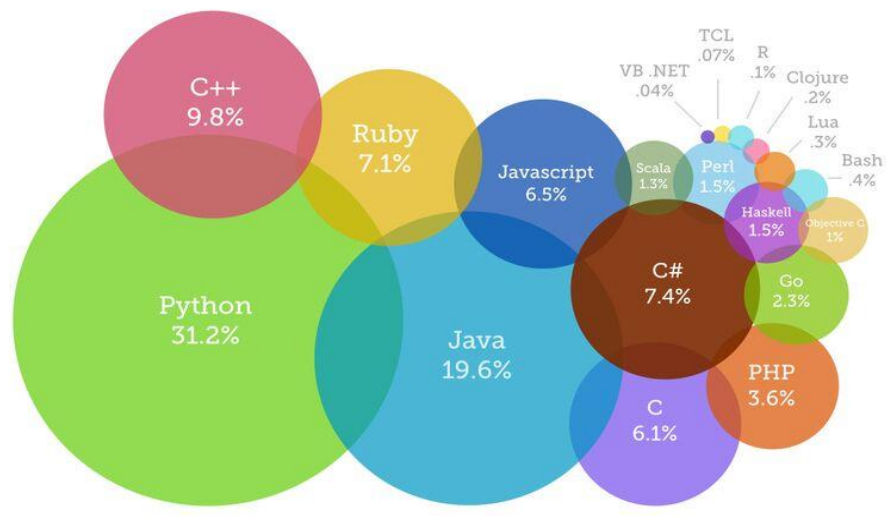
- 하나 이상의 **명령어**instruction들을 입력하여 프로그램을 작성하는 과정
- 다른 표현으로 **코딩**coding이라고도 한다.

- **프로그래머**programmer

- 프로그램을 작성하는 사람
- 컴퓨터에 명령을 내리는 명령어를 작성할 수 있어야 한다.

# 파이썬 밖에 없을까

- 프로그래밍 언어도 많은 종류가 있고 프로그래머들은 각자에게 맞는 언어를 골라 사용한다.
- 대표적인 언어는 '파이썬', '자바', 'C', 'C++', 'JavaScript' 같은 것들이다. 이 프로그래밍 언어들은 고유한 문법 체계를 가지고 있다.



2019년 가장 트렌디한 프로그래밍 언어



제가 파이썬의  
창시자 입니다!

- **파이썬**Python은 **귀도 반 로섬**Guido van Rossum이 1991년에 개발한 대화형 프로그래밍 언어인데 최근 많은 인기를 얻고 있다.
- 가장 큰 이유는 생산성이 뛰어나기 때문이다. 파이썬을 이용하면 간결하면서도 효율적인 프로그램을 빠르게 작성할 수 있다.
- 파이썬은 오픈 소스이어서 무료이고 패키지들이 계속 추가되고 있어서 매일 진화하는 언어이기도 하다.



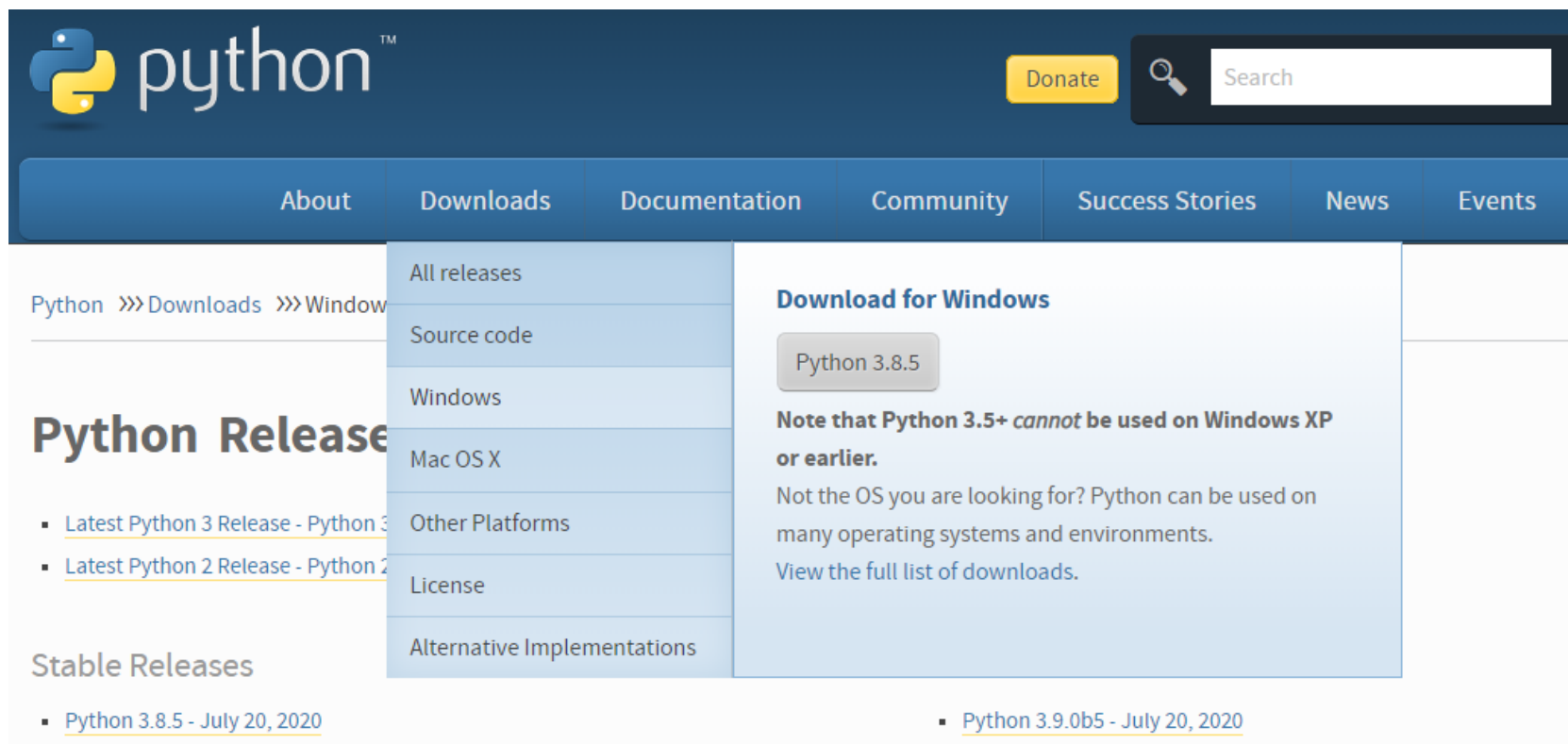
# 파이썬밖에 없을까

- 파이썬은 무엇보다도 초보자의 프로그래밍 입문에 적합한 언어이다. 그 이유는 프로그래머가 한 줄의 문장을 입력하고 엔터키를 치면 명령 해석기인 **인터프리터**interpreter가 이것을 바로 실행한다.
- 파이썬 프로그래머는 자신이 작성한 문장의 결과를 입력 즉시 볼 수 있기 때문에 입문자도 간편하게 프로그램의 실행을 살펴볼 수 있다.



# 파이썬 개발도구를 설치해 보자

- 홈페이지 접속
  - <http://www.python.org/>



# 파이썬 개발도구를 설치해 보자

- python-3-9-x.exe 실행 후(최신 버전으로 설치하세요)
- "Install launcher for all users(recommended)"
- "Add Python 3.9 to PATH" 선택





파이썬 실행환경 : IDLE

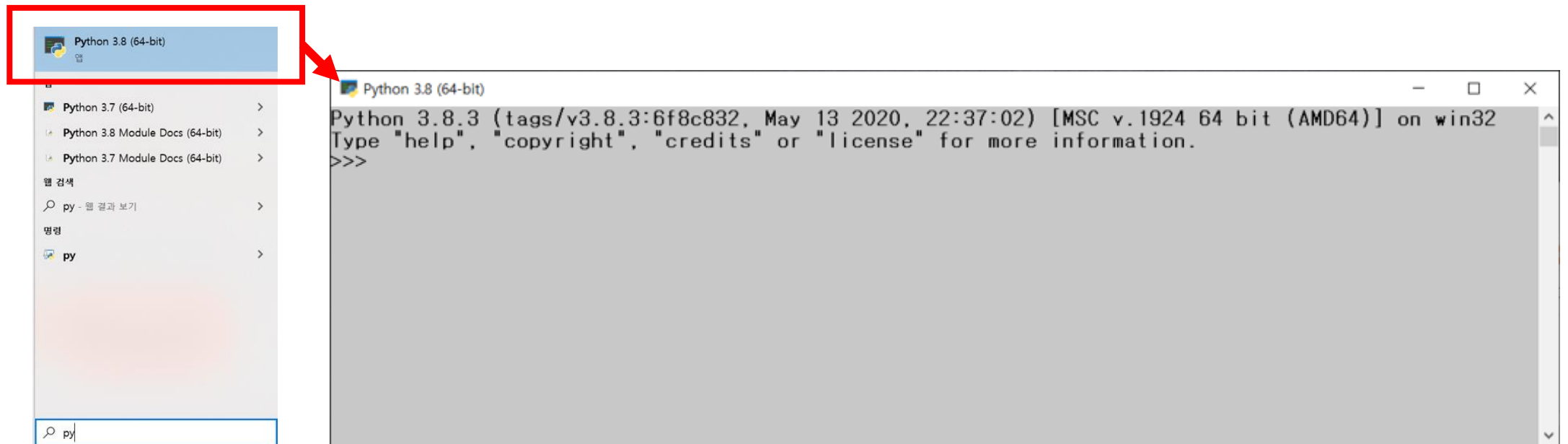
파이썬 인터프리터





# 파이썬 인터프리터 사용해 보기

- 시작 버튼을 눌러 "python"을 검색 후 Python 3.8을 눌러서 실행
- 파이썬 인터프리터는 처음 자신을 소개한다. 화면 첫 줄에는 자신의 **버전**version 등의 정보를 보여주고 있다. 그리고 다음 줄에는 더 많은 정보를 원할 경우 입력할 수 있는 명령들을 보여 주고 있다.



# 파이썬 인터프리터 사용해 보기

- 사용자의 입력을 받을 수 있는 **프롬프트**prompt에 파이썬 명령어를 준다.(커서가 깜박일 것이다)
- 프롬프트에 `print('Hello Python!!')` 을 입력 후 엔터키를 누르자
- 프롬프트 아래에 Hello Python!!이 출력된다.
- 이런 문자의 모음을 컴퓨터 프로그래밍에서는 **문자열**string이라고 한다.
- `print()`는 파이썬의 **내장함수**built-in function로 괄호 안의 값을 화면에 출력하는 역할을 한다.

```
>>> print('Hello Python!!')
Hello Python!!
```

# 파이썬 인터프리터 사용해 보기

- 인터프리터에서 간단한 계산을 할 수도 있다. 그 값을 저장해 놓았다가 출력을 할 수도 있다.

```
>>> 5 + 6
11
>>> 반지름 = 4
>>> 면적 = 3.14 * 반지름 * 반지름
>>> print(면적)
50.24
```

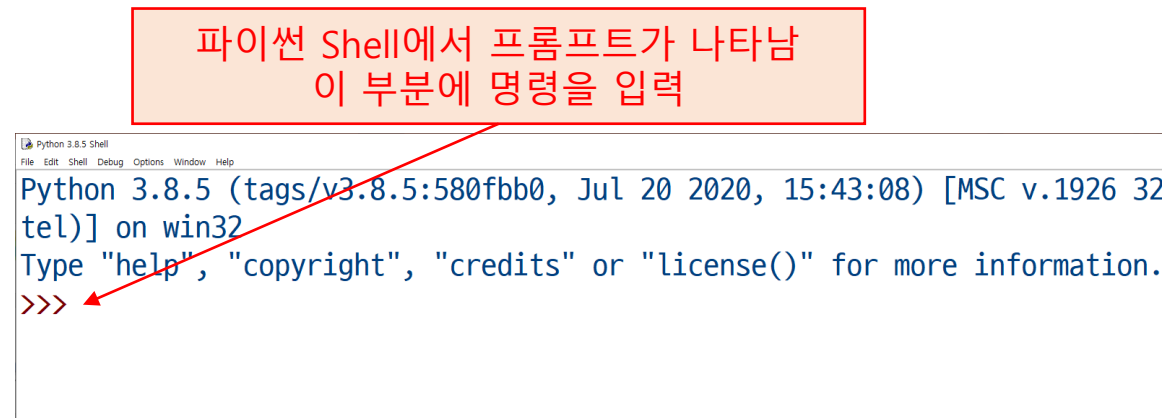


# 파이썬 개발도구에서 'Hello World'를 출력 해보자

- 파이썬 인터프리터로 코딩을 하는 것은 간단하기는 하지만 몇 가지 문제가 있다.
- 잘못 입력한 경우 수정하기가 힘들다. 또 다른 문제는 한 번 일을 시키고 나면, 이 일을 다음에 다시 시키기 어렵다.
- 이런 문제를 피하는 방법은 파이썬에 입력할 명령어들을 모아 하나의 **소스 코드**source code로 저장해 두는 것이다.
- 소스 코드의 작성과 관리를 돕는 도구를 **통합 개발 도구**integrated development environment 혹은 **IDE**라고 한다.

# 파이썬 개발도구에서 'Hello World'를 출력 해보자

- 파이썬을 설치하면 간단한 IDE가 제공되는데, 이 도구의 이름이 IDLE이다.
- IDLE는 Integrated Development and Learning Environment의 약자로 "통합적 개발/학습 환경"이라는 뜻을 가지고 있다.
- 아래와 같이 시작메뉴의 "모든 프로그램"에서 IDLE를 검색하면 된다. 이렇게 IDLE를 찾아 실행하면 기본적으로 파이썬 인터프리터와 동일한 모습이다.



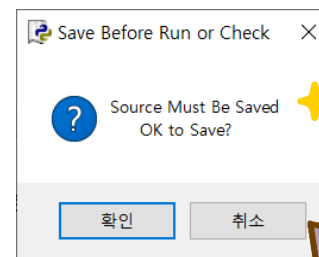
# 파이썬 개발도구에서 'Hello World'를 출력 해보자

- 파이썬 IDLE 프로그램의 "File" 메뉴를 이용하여 "New File"을 선택할 수 있다. 새로운 파일이 생성되고 편집이 가능한 상태가 된다.
- 이 창에 아래 그림처럼 코드를 입력하고 면적의 값을 출력하도록 해 보자. 실행을 시키는 방법은 "Run" 메뉴 아래에 있는 "Run Module"을 선택하면 된다.
- 여러분이 편집한 파일이 저장되지 않았다면 아래 그림의 오른쪽과 같이 먼저 저장하라는 메시지가 뜰 것이다.

```
print('Hello Python!!')

반지름 = 5
PI = 3.141592

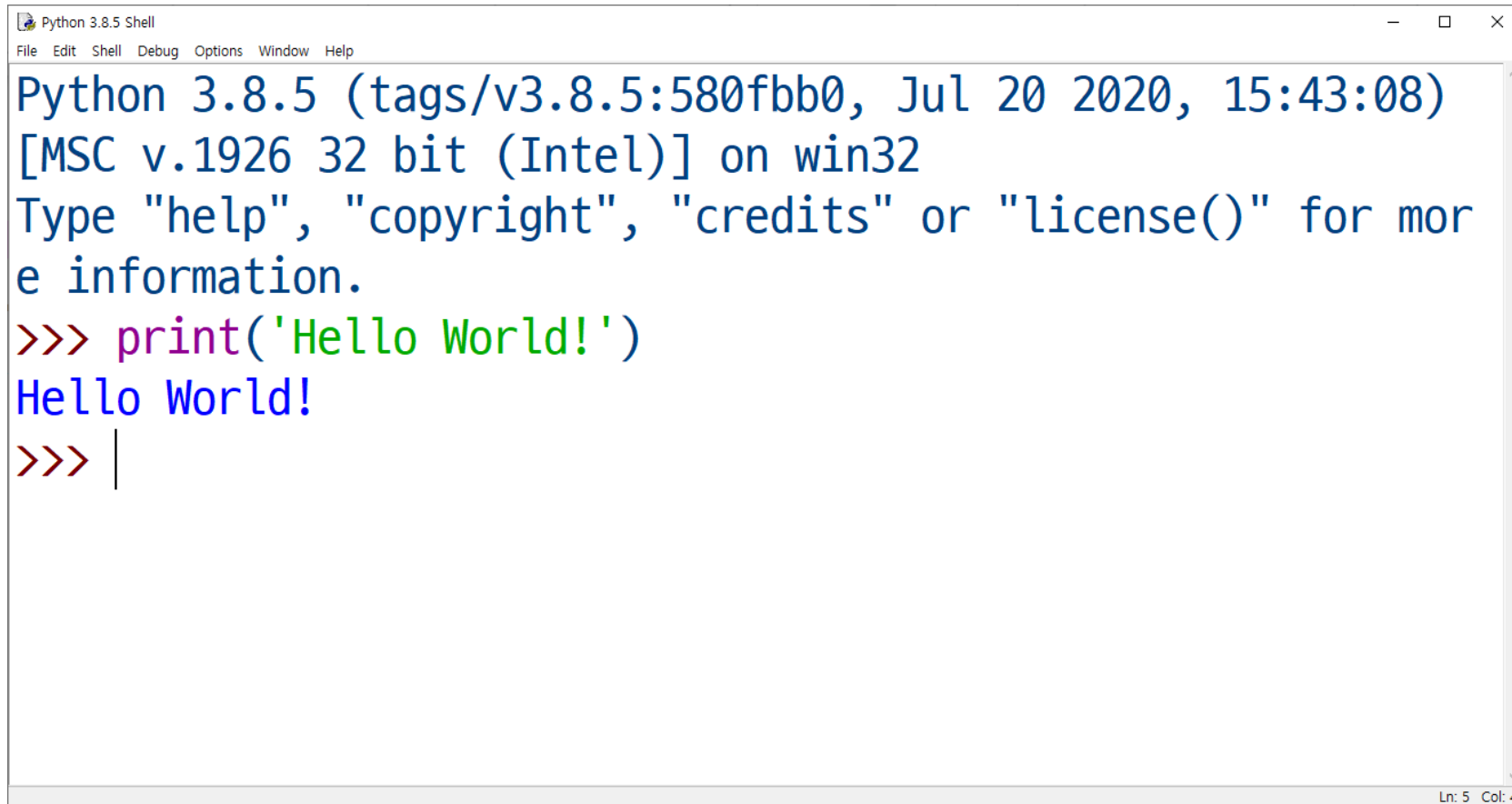
면적 = PI * 반지름 ** 2
print(면적)
```



파이썬 파일을 실행하기 전에는 반드시 **저장**을 해야 해요

# 파이썬 개발도구에서 'Hello World'를 출력 해보자

- 저장이 끝나고 에러없이 수행되면, 파이썬 인터프리터 창에 여러 분이 작성한 코드의 실행 결과가 나타난다.



```
Python 3.8.5 Shell
File Edit Shell Debug Options Window Help
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020, 15:43:08)
[MSC v.1926 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for mor
e information.
>>> print('Hello World!')
Hello World!
>>> |
```

The screenshot shows a Python 3.8.5 Shell window. The title bar reads 'Python 3.8.5 Shell'. The menu bar includes 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Window', and 'Help'. The main text area displays the Python version and build information: 'Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020, 15:43:08) [MSC v.1926 32 bit (Intel)] on win32'. It also shows the prompt 'Type "help", "copyright", "credits" or "license()" for mor e information.' followed by the execution of the command '>>> print('Hello World!')' which results in the output 'Hello World!'. The prompt '>>>' is followed by a vertical bar '|', indicating the cursor is ready for the next command. The status bar at the bottom right shows 'Ln: 5 Col: 4'.

# 주석으로 이해하기 쉬운 코드를 만들기

- **주석**comment은 소스 코드에 붙이는 설명글과 같은 것이다. 주석은 프로그램이 하는 일을 설명한다.
- 주석은 프로그램의 실행 결과에 영향을 끼치지 않는다. 그러니 주석이 없어도 프로그램의 실행 결과는 완전히 똑같다.
- 파이썬에서는 #로 시작하면 줄의 끝까지 주석으로 취급한다

```
# 다음 코드는 반지름을 이용하여 원의 면적을 출력하는 코드이다
반지름 = 4                                # 반지름의 값을 저장한다. 이때 공백이 들어가면 안된다
면적 = 3.14 * 반지름 * 반지름            # 반지름의 값을 이용하여 원의 면적을 구한다
print(면적)                              # 면적을 화면에 출력한다
```

파이썬 코드를 설명하는 문장  
실행되지 않음



Energy

$$E = mc^2$$

mass

squared

equals speed of light  
(constant)

myCode.py

```
E = m*c**2 # Energy  
''' m: mass, c: speed of light  
에너지는 질량과 광속으로 표현 가능.  
c는 상수이므로 질량으로 결정 됨 '''
```



주석문을 보니  
이해가 되네요!

# 무작정 계산부터 해보자

- 컴퓨터는 기본적으로 계산하는 기계이다.
- 덧셈, 뺄셈, 곱셈, 나눗셈을 컴퓨터에게 시켜보자. 암산으로도 가능한 계산을 연습 삼아 한 줄씩 입력하고 실행해보자.
- **대화형 모드** `interpreter mode` 모드에서  $2 + 3$ 과 `print(2 + 3)`의 결과는 같다.

```
>>> 2 + 3
5
>>> print(2 + 3)
5
>>> print(2 - 3)
-1
>>> print(2 * 3)
6
>>> print(2 / 3)
0.6666666666666666
```



# 무작정 계산부터 해보자

- 이번에는 좀 어려운 계산을 해보자.

```
>>> print(2345 * 9876 - 5678)
23153542
```



```
>>> print(123456789123456789 * 123456789123456789)
15241578780673678515622620750190521
```



## print() 함수로 원하는 메시지 출력해 보기

- 따옴표로 시작하여 같은 따옴표로 끝나는 문자열을 프롬프트에 입력하면 그 상태 그대로 나타난다. 데이터가 문자열이라는 것을 보여주는 표시이다.
- 하지만 print() 함수 안에 문자열이 있을 경우 따옴표는 나타나지 않는 것에 유의하자. 이때는 print() 함수에 넘겨진 문자열을 출력하라는 명령을 수행한 결과를 보여주는 것이다.

```
>>> 'Hello'           # 문자열 'Hello'
'Hello'
>>> "Hello"           # 문자열 "Hello"는 'Hello'와 동일하다
'Hello'
>>> print('Hello')    # print() 함수안에 문자열이 있을 경우 따옴표는 나타나지 않음
Hello
>>> print("즐거운 " + "파이썬 익히기") # 두 텍스트 데이터를 연결하여 출력함
즐거운 파이썬 익히기
```

## print() 함수로 원하는 메시지 출력해 보기

- 파이썬에서 큰따옴표("...")나 작은따옴표('...')로 둘러싸이면 **텍스트 정보를 담은 문자열**이 된다.
- 문자열에 + 연산자를 이용하여 다른 문자열을 덧붙이면, 두 문자열이 연결된다. 파이썬 문자열에는 다음과 같이 곱셈 기호를 사용하는 것도 가능하다.
- 컴퓨터는 계산도 잘하지만 어떤 것을 반복하는 것에도 소질이 있다.

```
>>> print('반가워요 ' * 20)           # '반가워요'를 20회 반복 출력함  
반가워요 반가워요 반가워요 반가워요 반가워요 반가워요 반가워요 반가워요 반가워요 반가워요 반가워요 반가  
워요 반가워요 반가워요 반가워요 반가워요 반가워요 반가워요 반가워요
```

# print() 함수로 원하는 메시지 출력해 보기

- 문자열과 숫자를 구별하여야 한다.
- 예를 들어서 " 100"은 문자열이고 100은 숫자이다. "100"+"200"을 실행하면 "100200"이 출력된다.  
다음과 같이 100 + 200과 "100"+"200"의 결과가 다르다는 것에 각별히 유의하자.

문자열 덧셈은 두 문자열을 이어 붙여 줍니다.

```
>>> print("100" + "200")    # 문자열 '100', '200'을 연결한다
100200
>>> print(100 + 200)        # 숫자 두 개의 합을 구한다
300
```

숫자 덧셈은 값을 더합니다.

# 파이썬이 정말로 편리한 이유 : 모듈 설치하기

- 파이썬 함수나 변수 또는 클래스들은 별도의 스크립트 파일로 저장하여 불러서 사용하는 것이 편리한데 이렇게 만든 스크립트 파일을 **모듈**module이라고 부른다.
- 파이썬 설치 시에 함께 제공되는 모듈을 **표준 라이브러리**standard library라고 부른다. 다양한 문제를 해결하기 위해서는 이 표준 라이브러리의 기본적인 기능뿐만 아니라 여러 프로그래머와 기관에서 만들어 놓은 라이브러리를 가져다가 활용할 필요가 있다.
- 이 라이브러리는 흔히 **패키지**package라고도 한다. 외부 패키지를 사용하기 위해서는
  - 1) 파이썬 시스템에 pip라는 프로그램을 이용해 패키지를 설치하는 작업
  - 2) 설치된 패키지를 활용을 위해 불러들이는 작업이 필요하다.

# 파이썬이 정말로 편리한 이유 : 모듈 설치하기

- pip는 파이썬의 패키지 관리 소프트웨어로 표준 라이브러리에 포함되지 않은 외부 라이브러리를 설치하도록 도와주는 도구이다.
- pip를 이용하여 설치할 때는 윈도우 컴퓨터의 명령행에서 다음과 같은 명령을 입력한다.

```
C:\> pip install package-name
```

- 예를 들어 numpy라는 패키지를 설치하기 위해서는 아래와 같이 콘솔 명령창에 pip install numpy만 입력하면 된다.



```
명령 프롬프트 - python
C:\Users\user>pip install numpy
Collecting numpy
  Using cached https://files.pythonhosted.org/packages/a8/ce/36f9b4fbc7e675a7c8a3809dd5902e24cecfcdabc006e8a7b2417c2b830a2/numpy-1.17.2-cp37-cp37m-win32.whl
Installing collected packages: numpy
Successfully installed numpy-1.17.2
```

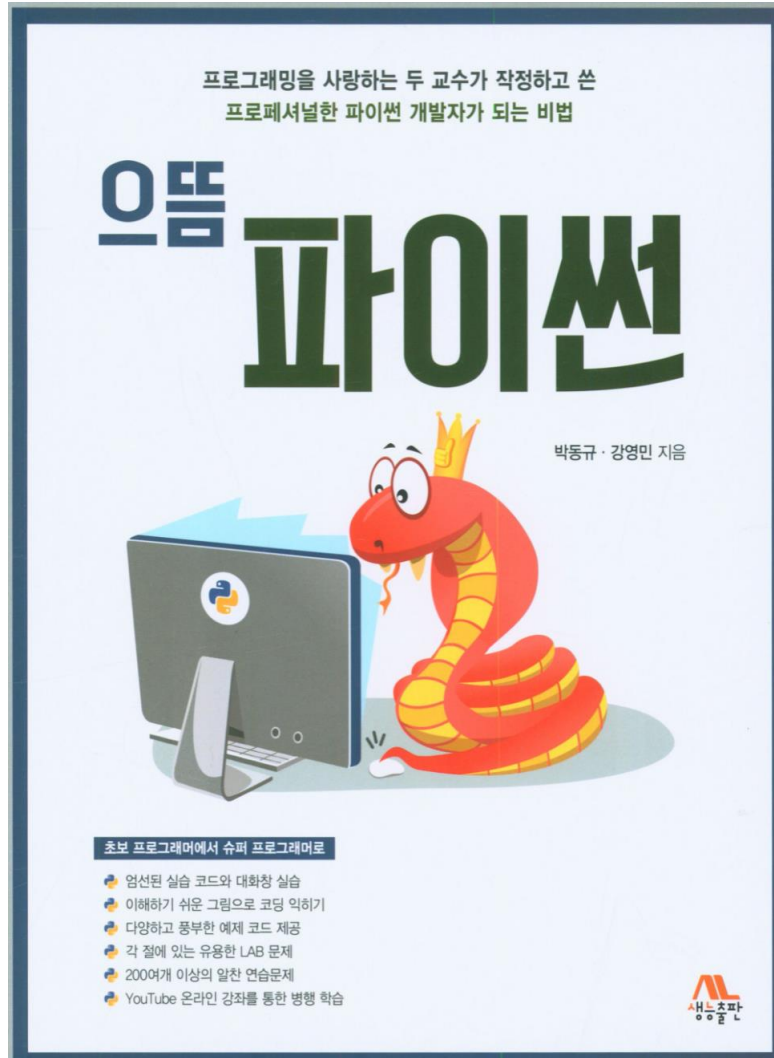


## 1.13 파이썬이 정말로 편리한 이유 : 모듈 설치하기

- 우리가 이 책에서 사용할 패키지는 다음과 같다.
  - numpy, matplotlib, pandas, scikit-learn, seaborn, opencv-python
- 위에서 다룬 모듈을 설치하기 위해서는 다음과 같이 pip를 이용하여 모듈 설치를 하기만 하면 된다.
  - 심표는 사용하지 않는다.

```
C:\> pip install numpy matplotlib pandas scikit-learn seaborn opencv-python
```

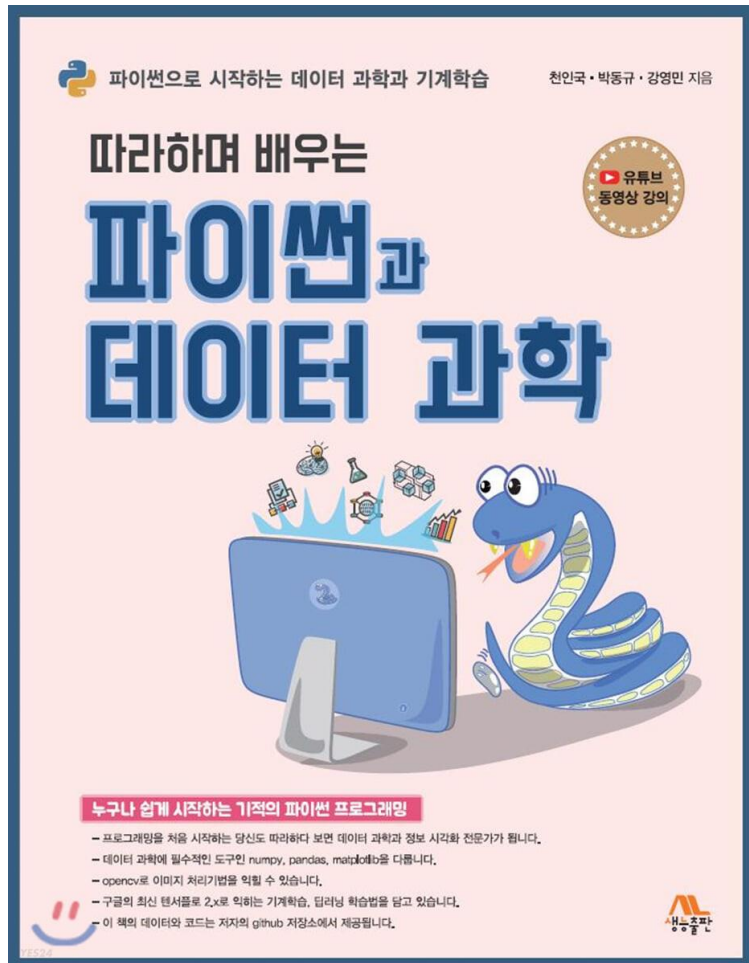
# 파이썬 학습을 위한 사이트



으뜸 파이썬  
박동규, 강영민 저 | 생능출판사 | 2020년 02월 17일

<https://github.com/dongupak/Prime-Python>

# 파이썬 학습을 위한 사이트



따라하며 배우는 파이썬과 데이터 과학  
천인국, 박동규, 강영민 저 | 생능출판사 | 2020년 12월 28일

<https://github.com/dongupak/DataSciPy>

시작해 봅시다