

따라하며 배우는

파이썬과 데이터 과학



1장 데이터 과학과 파이썬의 세계로

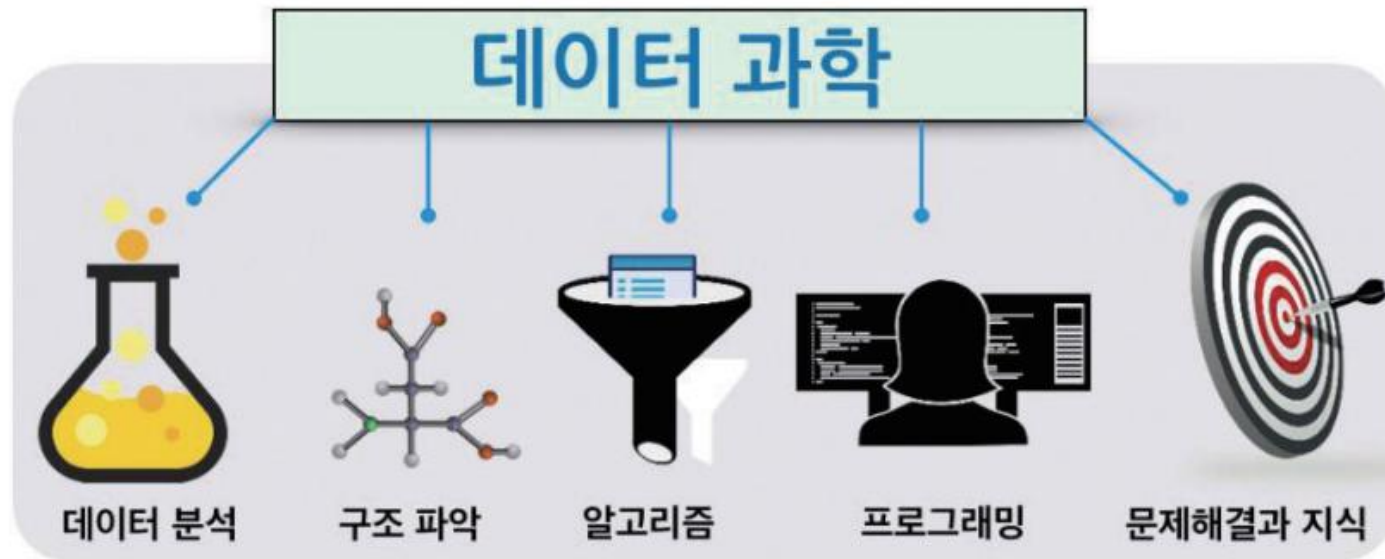


이장에서 배울 것들

- 컴퓨터와 프로그래밍에 대해 이해해 보려고 해요.
- 프로그램은 프로그래머가 프로그래밍을 통해 만들어요.
- 파이썬은 영국 코미디 프로그램의 이름을 딴 프로그래밍 언어예요.
- 최근 파이썬은 인공지능과 데이터 과학 분야에서 엄청난 인기를 얻고 있지요.
- 이런 파이썬을 사용하기 위해 필요한 것들을 살펴볼 것입니다.
- 파이썬 프로그래머가 되기 위해 준비운동을 해 봅시다.
- 자! 시작해 볼까요?

1.1 데이터에 숨겨진 보물 찾기 : 데이터 과학의 세계로 출발

- **데이터 과학** *data science*은 데이터에서 과학적 방법으로 정보나 지식을 추출하는 학문이다.
- 통계학, 컴퓨터 과학, 그리고 데이터가 발생하는 영역과 관련된 학문 분야의 이론과 기술을 융합적으로 사용하여야 한다.

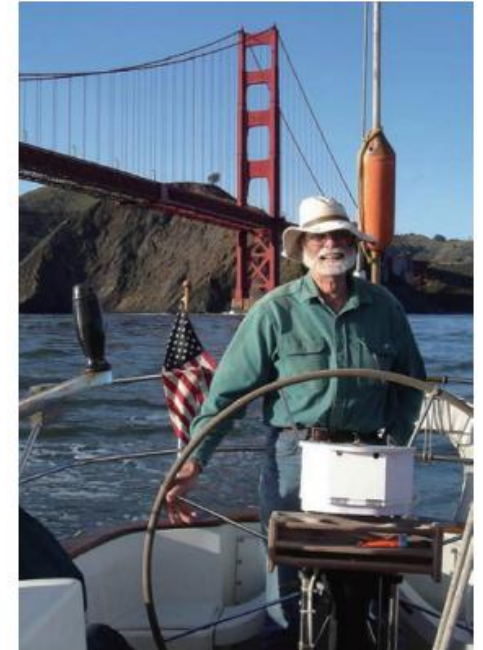


- **파이썬**은 데이터 과학자들을 위한 분석도구이다.
- 프로그래밍을 통한 데이터 처리는 모든 학문에서 중요한 기초가 되었다.
- 뉴턴의 과학 이후 : 실험과학, 이론과학이 축이 됨
- 현재의 과학 : 데이터 중심의 과학



잠깐 - 과학의 네 번째 패러다임: 데이터 과학

1998년도 튜링 상 수상자인 **짐 그레이**Jim Gray 박사는 뉴턴 이후로 실험 과학과 이론 과학이 주류 과학의 두 축이었는데, 최근 수십 년 동안은 컴퓨터를 이용한 계산 과학이 과학 연구의 방법론으로 자리를 잡았다고 분석했다. 그리고 그는 **과학의 네 번째 패러다임**으로 **데이터 중심의 과학**data-intensive science이 등장하고 있다고 보았다. 컴퓨터 과학이 축적한 기술을 바탕으로 여러 분야의 어려운 문제를 해결하는 새로운 변화를 데이터 과학이 만들어 낼 수 있다는 것이 그의 생각이었다. 그는 2007년 자신의 요트를 몰고 바다에 나갔다가 안타깝게도 돌아오지 않았다.



1.2 데이터 과학은 무엇을 다루나



- 데이터를 다루는 이유는 무엇일까? 분석한 데이터는 어디에 쓸까?
- 2010년대 후반 서울시는 심야버스 노선을 조정하기 위해 택시 승하차 정보와 통화량 데이터를 사용했다.
- 서울시는 분석한 자료를 바탕으로 노선을 조정하여 시범 운행한 뒤에, 이것을 바탕으로 노선을 확정하였다.



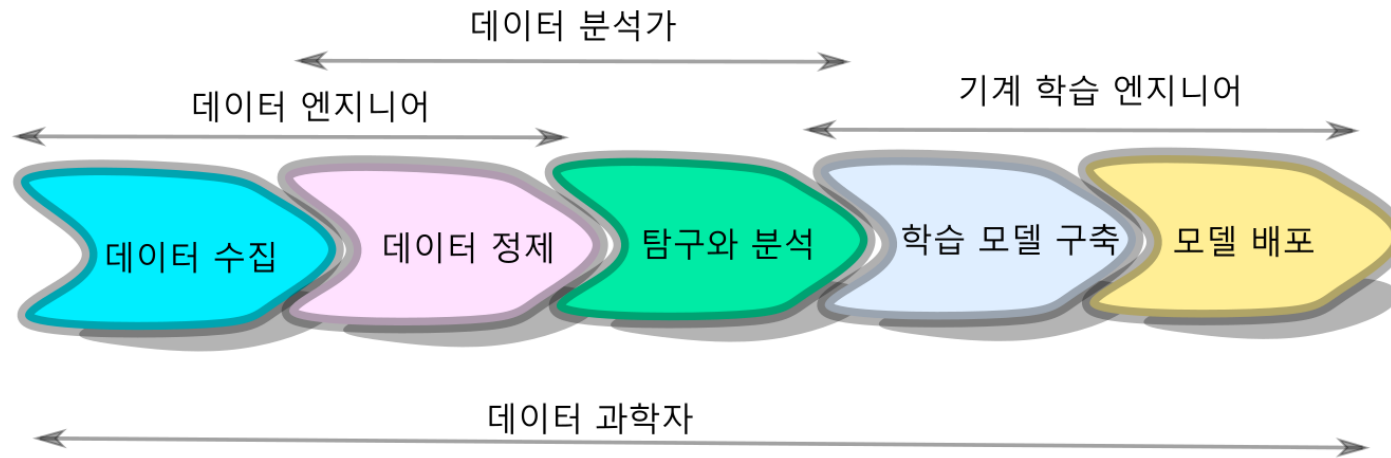
잠깐 - 빅데이터를 활용한 공공 정책 결정의 예

2018년 국토교통부는 중앙부처와 지자체의 정책지원을 위한 빅데이터 분석과제를 완료했다고 발표하였다. 이 분석과제들은 아래와 같다.

- ① (전주시) 교통카드 DB를 활용한 대중교통 현황 빅데이터 분석
- ② (부산광역시) 역세권 토지이용 구상을 위한 빅데이터 활용
- ③ (국토부, LH) 개발공장과 계획입지 간의 상관성 분석
- ④ (소방청) 빅데이터를 활용한 화재대응 취약지역 분석
- ⑤ (식품의약품안전처) HACCP 인증과 식품안전사고 상관분석
- ⑥ (국민건강보험공단) 건강보험 의료 이용지도 구축

1.3 데이터 처리 과정과 관련 전문가의 역할

- 데이터 과학은 데이터를 확보하여 응용 분야에 사용할 수 있는 정보로 만들기 위해 다양한 과정을 수행한다.
- 데이터 과학은 컴퓨터를 이용한 계산 과학의 진보를 바탕으로 데이터의 수집과 분석, 활용과 관련된 모든 이론과 기술을 종합적으로 다루는 분야이다.
- 따라서 데이터 과학자는 데이터 엔지니어, 데이터 분석가, 기계학습 엔지니어의 역량을 모두 갖추고 다양한 문제를 해결해야 한다.



1.3 데이터 처리 과정과 관련 전문가의 역할

- 컴퓨터 하드웨어와 소프트웨어 기술의 급속한 발달에 예전에는 다루기 힘들었던 대규모의 데이터도 빠르게 처리할 수 있게 되었다.
- 컴퓨터를 잘 다룰 수 있다면 데이터를 처리하고 분석하는 일도 더욱 쉽고 효율적으로 바뀔 것이다.
- 특히 문제에 맞춰 **소프트웨어를 개발할 수 있는 프로그래밍 실력**은 이제 데이터 과학의 모든 단계에서 필요한 능력이 되고 있다.



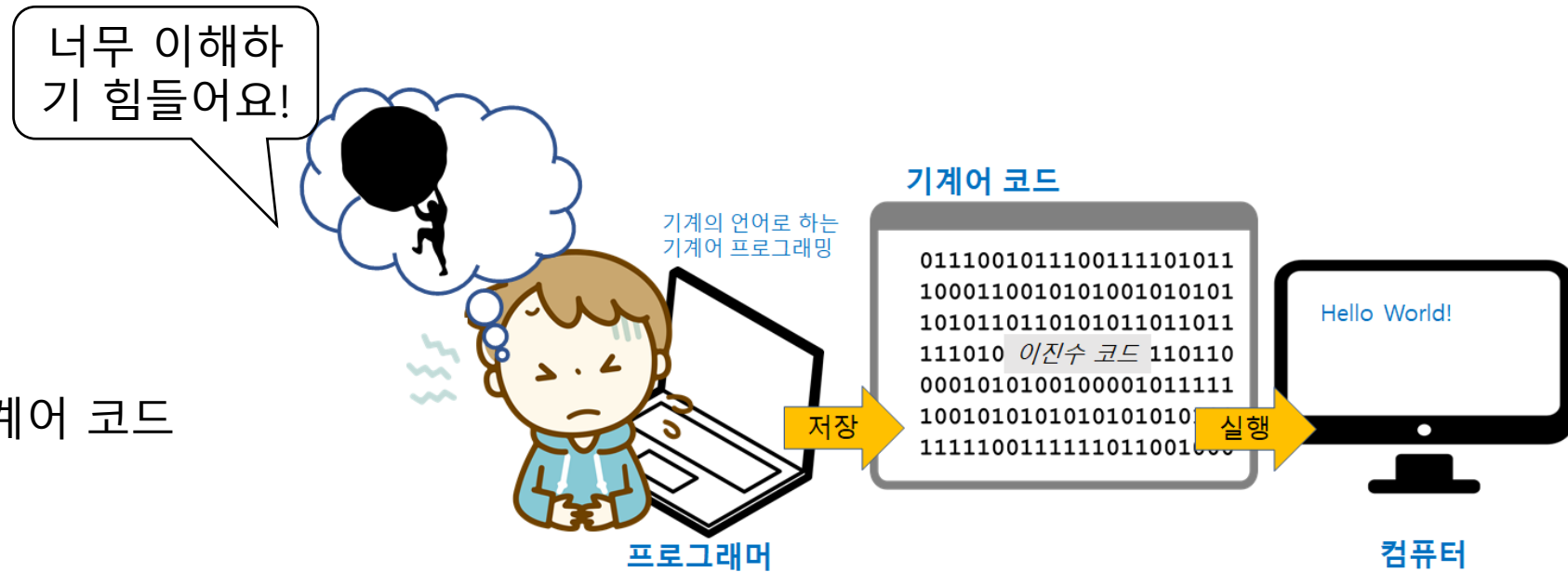
1.4 프로그래밍이란 무엇인가

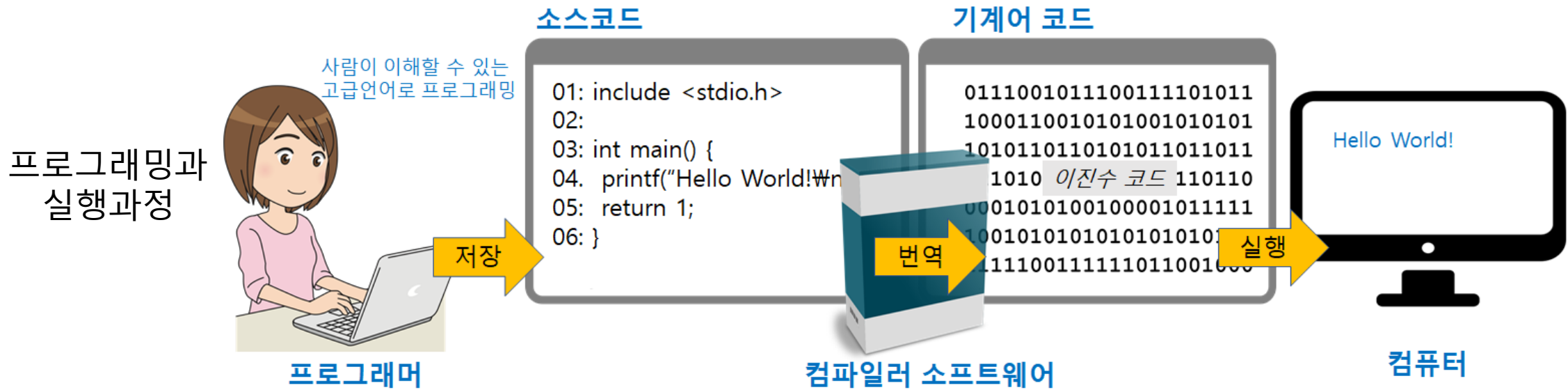
- **프로그램**program
 - 컴퓨터가 해야할 일을 미리 기록해 놓은 작업 지시서 같은 것
 - 우리가 사용하는 파워포인트나 카카오톡과 같은 것들이 모두 프로그램
- **프로그래밍**programming
 - 하나 이상의 **명령어**instruction들을 입력하여 프로그램을 작성하는 과정
 - 다른 표현으로 **코딩**coding이라고도 한다.
- **프로그래머**programmer
 - 프로그램을 작성하는 사람
 - 컴퓨터에 명령을 내리는 명령어를 작성할 수 있어야 한다.

파이썬 코드

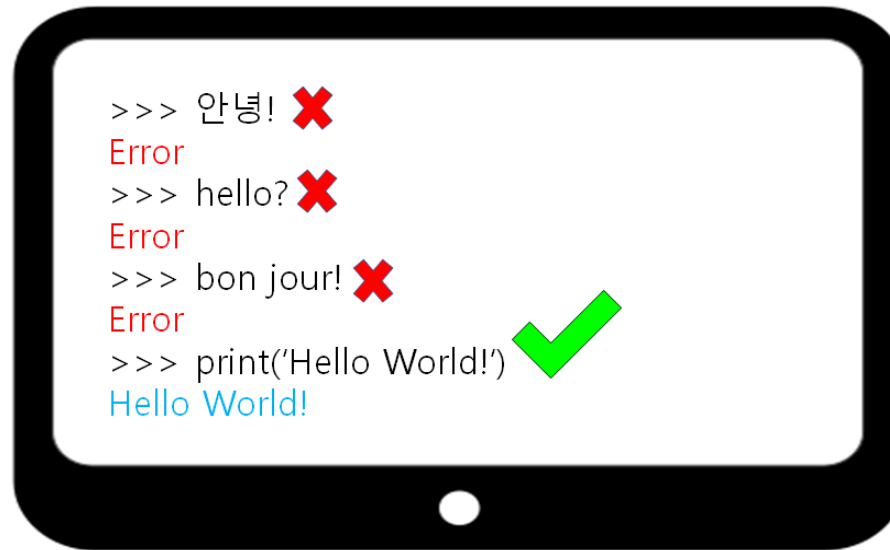


기계어 코드





파이썬 번역기는 엄격한
문법을 가지고 있다.
프로그래밍 문법을 따르지 않으면
코드가 실행되지 않는다



1.5 프로그래밍을 꼭 알아야 하나

- 자연과학이나 공학 분야의 학문과 직업에 종사하는 사람들뿐만 아니라 인문사회계열 역시 이제 컴퓨터 소프트웨어의 활용과 맞춤형 소프트웨어 개발 능력이 필수적인 시대로 바뀌고 있다.
- 지루하고 반복적인 작업은 컴퓨터에게 맡기고 사람은 더욱 창의적인 일에 집중하는 것이 생산성을 높일 수 있다.



1.5 프로그래밍을 꼭 알아야 하나

- 소설을 읽고 이해하는 것은 컴퓨터가 할만한 일이 아니지만, 소설 파일에서 단어의 출현 횟수를 세는 일은 초보적인 일 중에서도 가장 초보적인 일이다.
- 아래의 4줄 가량의 코딩을 통해서 우리는 소설 "모비딕"에 나타나는 모든 단어의 출현 횟수를 금방 계산할 수 있다는 것만 이해하자.

```
from collections import Counter

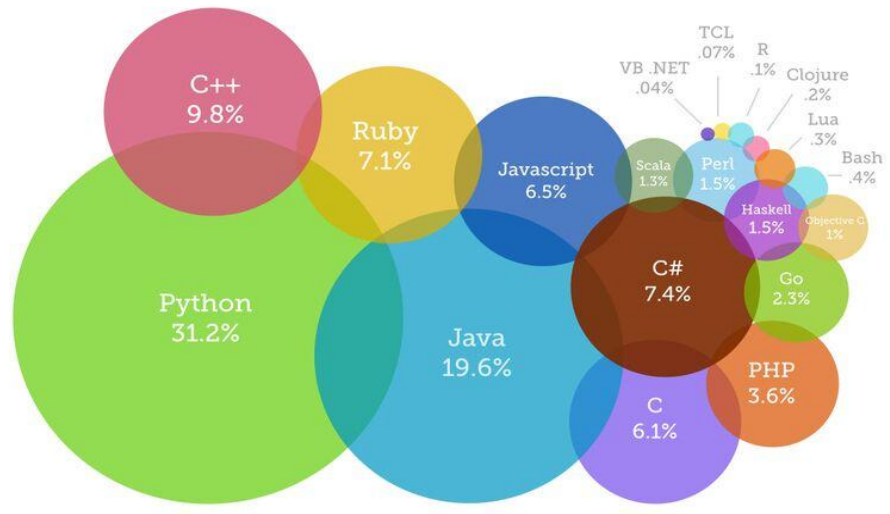
f = open("d:/mobydick.txt", encoding="utf-8")
count = Counter(f.read().split())
print("단어 출현 횟수 :", count)
```

단지 몇 줄의 코딩으로 소설속의 모든 단어의 출현 횟수를 알 수 있다
-> 코딩을 배우면 당신의 능력이 더 커질 것이다.



1.6 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



- 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.

제가 파이썬의
창시자입니다!



1.6 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.



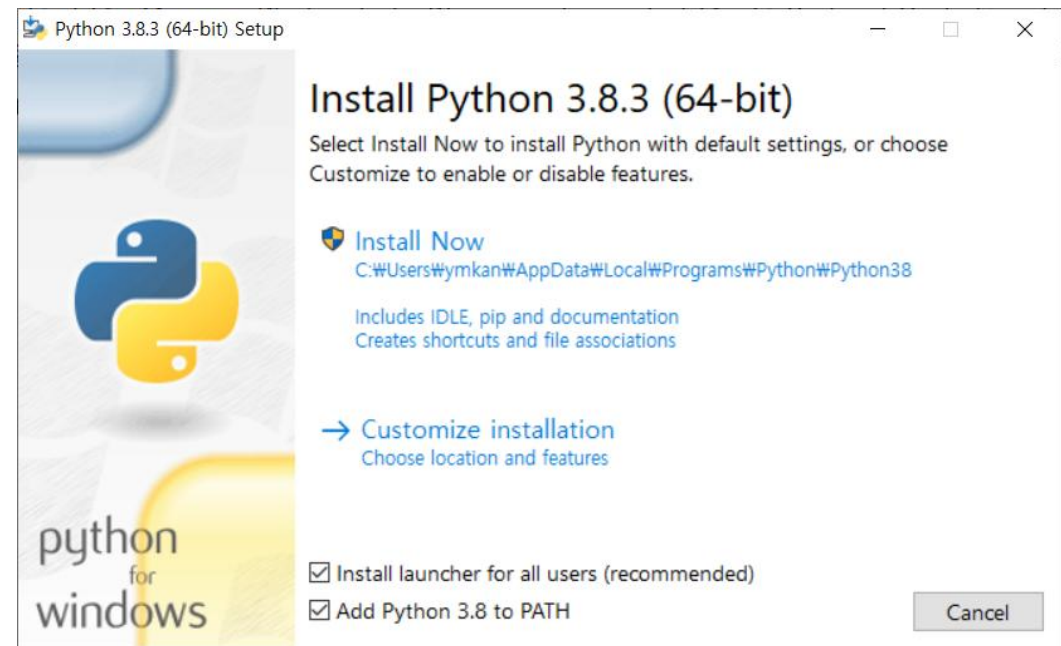
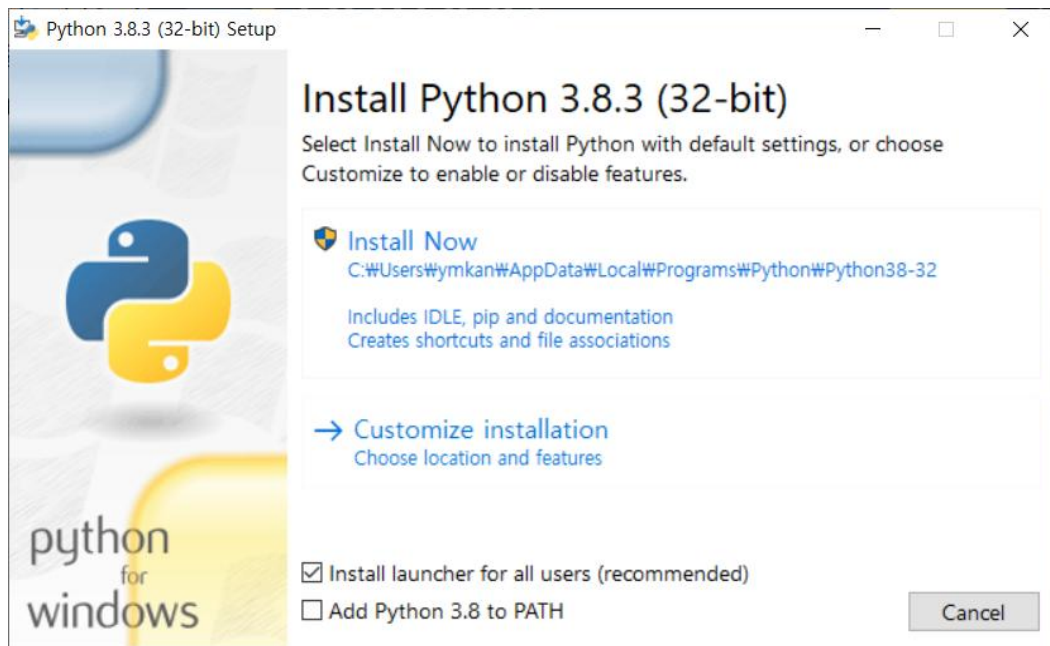
1.7 파이썬 개발도구를 설치해 보자

- 홈페이지 접속
 - <http://www.python.org/>

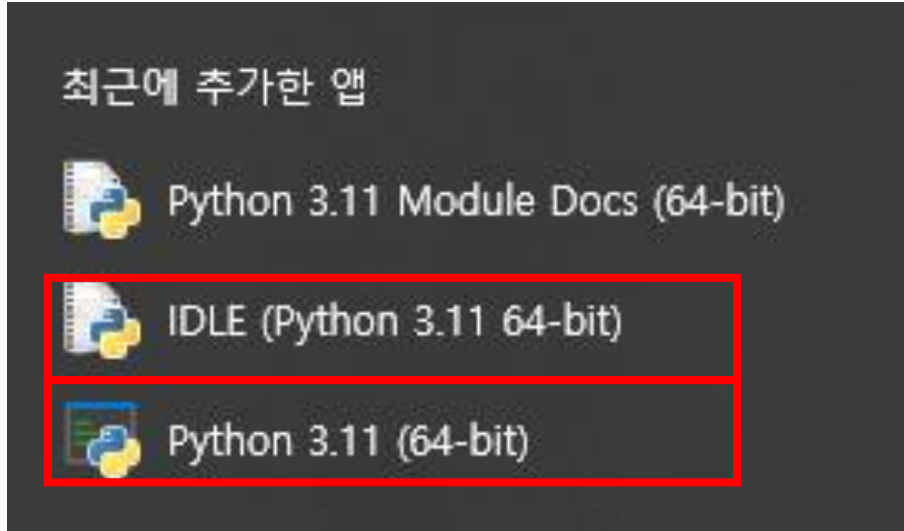


1.7 파이썬 개발도구를 설치해 보자

- python-3-8-x.exe 실행 후(가능한 최신 버전 3.11.4로 설치하세요)
- “Install launcher for all users(recommended)”
- **“Add Python 3.8 to PATH” 선택** [path 자동 설정]

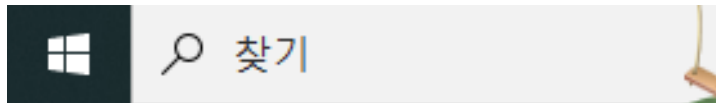


1.7 파이썬 최신버전 설치 확인



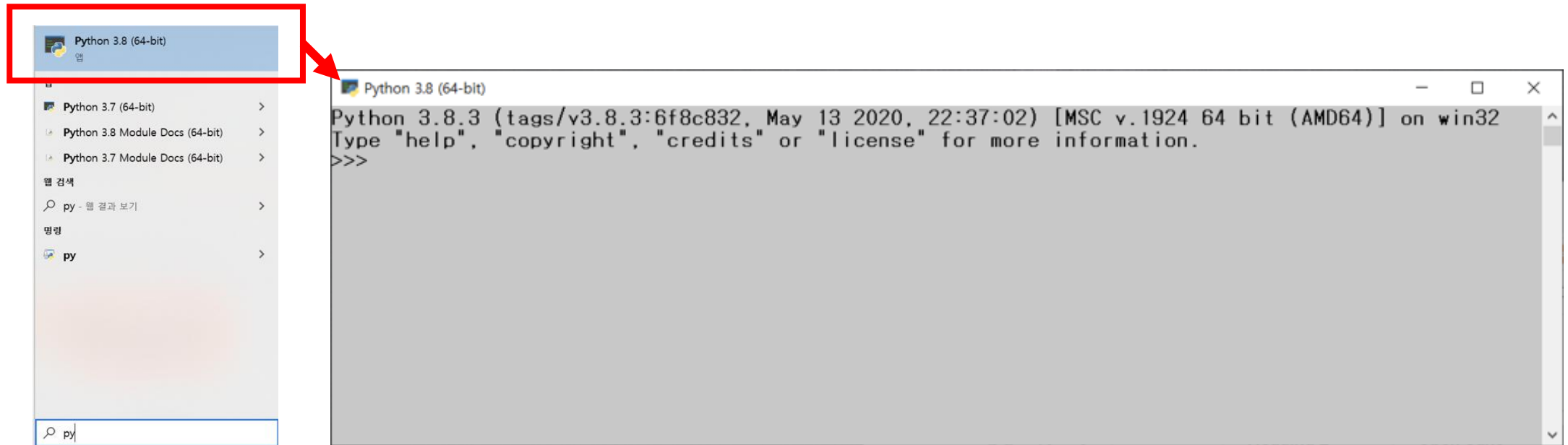
파이썬 실행환경 : IDLE

파이썬 인터프리터



1.8 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.



1.8 파이썬 인터프리터 사용해 보기

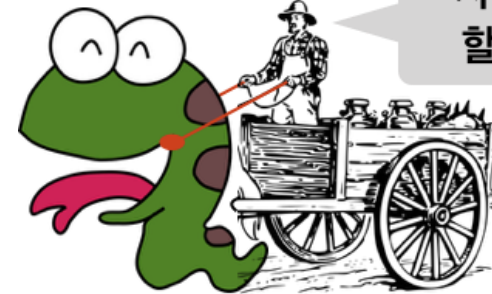
- 사용자의 입력을 받을 수 있는 **프롬프트**prompt에 파이썬 명령어를 준다.(커서가 깜박일 것이다)
- 프롬프트에 `print('Hello Python!!')` 을 입력 후 엔터키를 누르자
- 프롬프트 아래에 Hello Python!!이 출력된다.
- 이런 문자의 모음을 컴퓨터 프로그래밍에서는 **문자열**string이라고 한다.
- `print()`는 파이썬의 **내장함수**built-in function로 괄호 안의 값을 화면에 출력하는 역할을 한다.

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

1.8 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
```

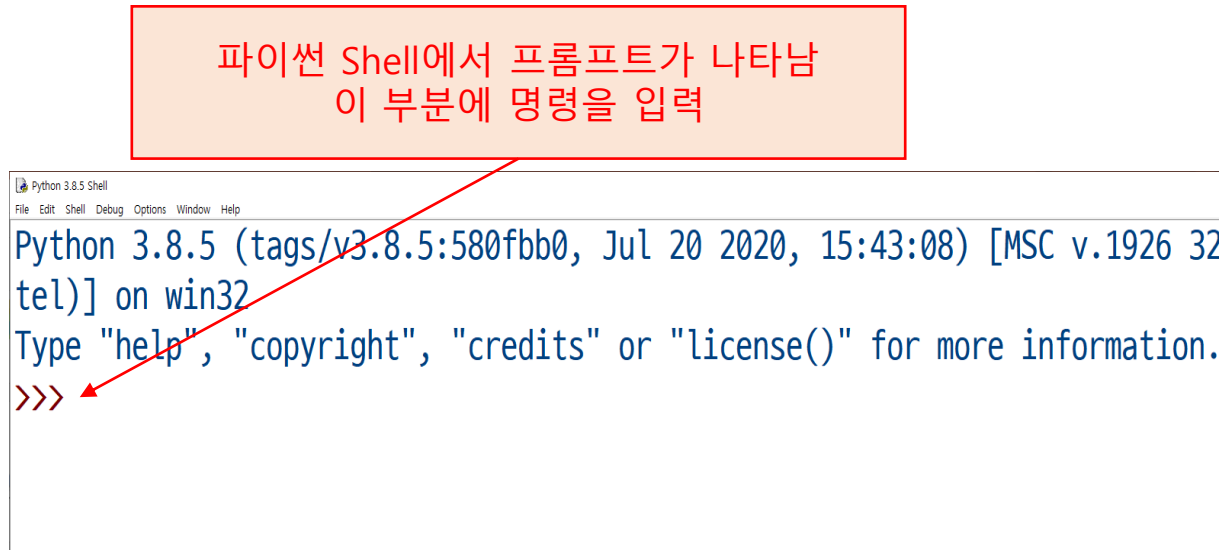
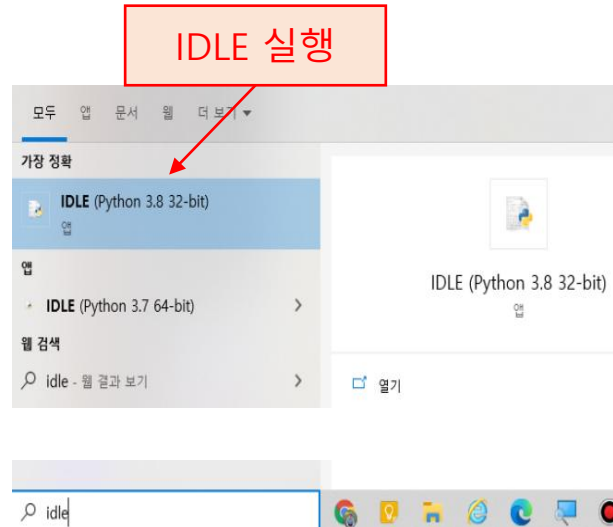


1.9 파이썬 개발도구에서 'Hello World'를 출력 해보자

- 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 code`로 저장해 두는 것이다.
- 소스 코드의 작성과 관리를 돕는 도구를 **통합 개발 도구** `integrated development environment` 혹은 **IDE**라고 한다.

1.9 파이썬 개발도구에서 'Hello World'를 출력해 보자

- 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.



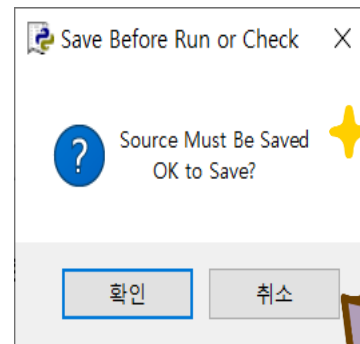
1.9 파이썬 개발도구에서 'Hello World'를 출력 해보자

- 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

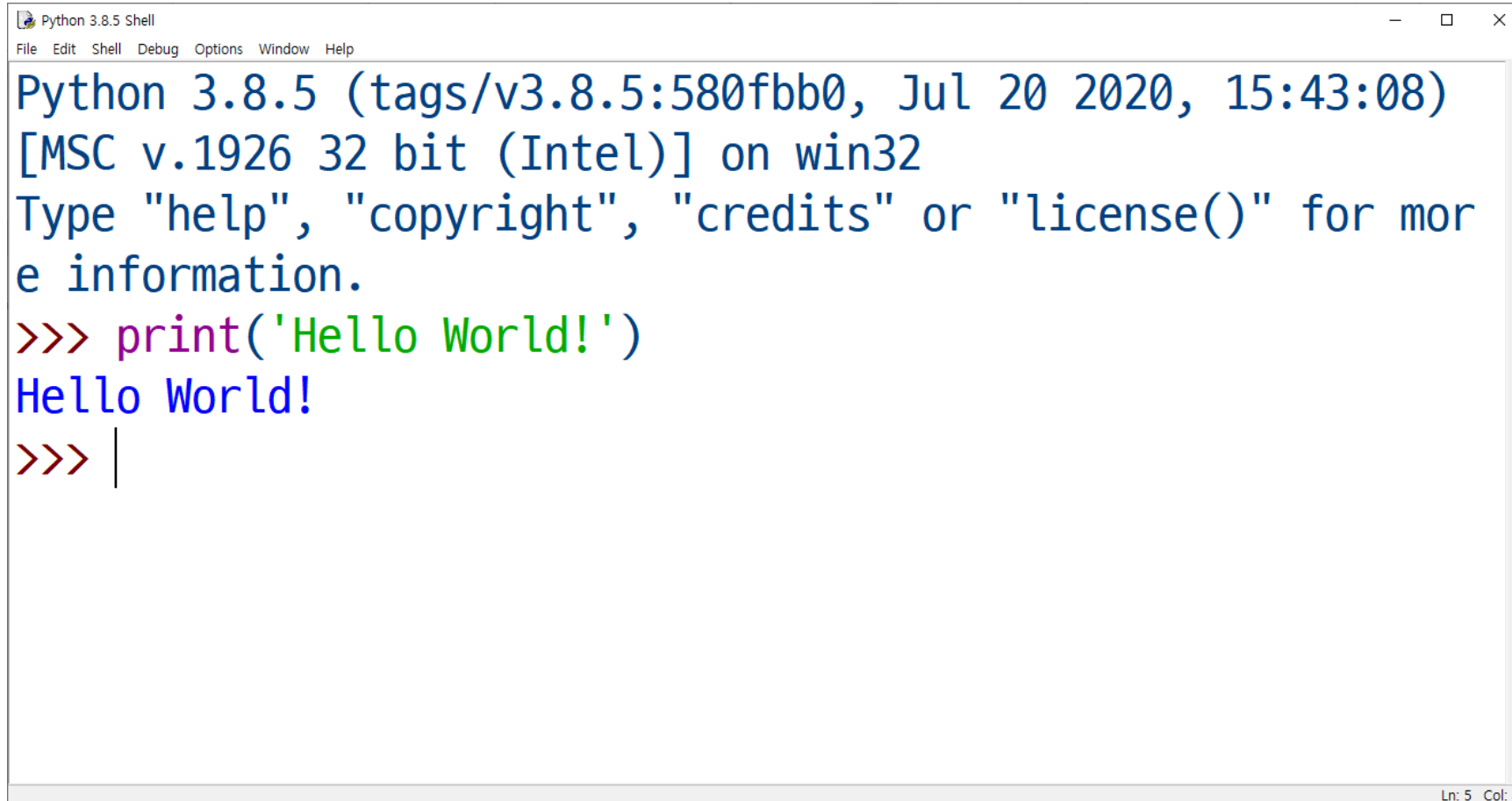
면적 = PI * 반지름 ** 2
print(면적)
```



파이썬 파일을 실행
하기 전에는 반드시
저장을 해야해요

1.9 파이썬 개발도구에서 'Hello World'를 출력 해보자

- 저장이 끝나고 에러없이 수행되면, 파이썬 인터프리터 창에 여러분이 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 more
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 then prompts the user with 'Type "help", "copyright", "credits" or "license()" for more information.'. The user has entered the command '>>> print('Hello World!')' and the output 'Hello World!' is displayed. The prompt '>>>' is followed by a vertical cursor bar. The status bar at the bottom right shows 'Ln: 5 Col: 4'.

1.10 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

1.10 주석으로 이해하기 쉬운 코드를 만들기

- 주석은 컴퓨터를 위한 것이 아니고 프로그램을 읽는 사람을 위한 것이다.
 - 프로그램은 완성된 후에도 유지보수를 계속 해야 한다.
 - 일을 물려 받은 후임 개발자가 소스 코드를 수정하거나 보완하려고 다시 읽을 수 있다.
 - 만약 코드가 복잡하고 오래 전에 만들었다면, 코드를 만든 사람조차도 잘 이해되지 않을 수 있다.
 - 다른 사람이 작성한 코드를 읽는 것은 더욱 어려운 일이다.
- 따라서 개발자는 자신의 프로그램이 무엇을 하려고 하는지 주석으로 만들어서 코드에 붙일 필요가 있는 것이다.

1.10 주석으로 이해하기 쉬운 코드를 만들기

- 여러 줄의 주석 처리를 하는 방법

```
# 여러 줄의 주석을 사용할 때  
# 해시 표시를 문장의 첫 줄에 넣어서  
# 주석을 사용할 수 있다
```

코드에는 이와 같은 많은 주석이 필요합니다. 이렇게 해야 나중에 코드를 이해하기 편합니다.

```
...  
작은따옴표를 이용하여 여러 줄 주석을 만드는 방법이다  
이 방식으로 주석을 만들면  
여러 줄에 걸친 주석을 남길 수 있다  
...
```

주석문의 스타일도 여러가지가 있지요.

```
""""  
혹은 이와 같이 큰따옴표를 이용할 수도 있다  
큰따옴표를 사용해도 작은따옴표를 사용하는 것과 동일하다  
""""
```

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!

1.11 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
```



1.11 무작정 계산부터 해보자

- 이번에는 좀 어려운 계산을 해보자.

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



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



1.11 무작정 계산부터 해보자



잠깐 – 인터프리터 모드(셸 모드)에서의 값 출력

파이썬 대화창에서 계산 결과를 출력할 때 $2+3$ 만 입력하고 엔터키를 눌러도 된다. 다음의 두 가지 문장은 같은 결과를 보여줄 것이다.

```
>>> 2 + 3
```

```
>>> print(2 + 3)
```



도전문제 1.1

파이썬의 인터프리터 모드를 이용하여 다음과 같은 계산을 하여보자

(1) $4*3*2*1$

(2) $1/2$

(3) $300 - 100$

(4) $423 + 1234$

(5) $(1/100) * 1234$

(6) $3.141592 * 12.0 * 12.0$

1.12 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
```

1.12 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 반가워요 반가  
워요 반가워요 반가워요 반가워요 반가워요 반가워요 반가워요
```

1.12 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.

1.12 print() 함수로 원하는 메시지 출력해 보기

- 문자열을 표현할 때 작은따옴표와 큰따옴표 중 아무것이나 사용해도 된다. 딱 한 가지 지켜야 할 것은 각 따옴표로 시작한 문자열은 처음에 쓴 따옴표로 끝내야 한다는 약속이다.
- 줄바꿈을 포함하여 여러 줄에 걸친 문자열을 표현하고 싶을 때도 있다. 이때는 작은따옴표나 큰따옴표 세 개로 문자열을 시작하고 같은 방식으로 닫으면 된다.



잠깐 - 따옴표의 사용법과 여러 줄에 걸친 문자열

문자열을 표현할 때 작은따옴표와 큰따옴표 중 아무것이나 사용해도 된다. 딱 한 가지 지켜야 할 것은 큰따옴표로 시작한 문자열은 큰따옴표로, 작은 따옴표로 시작한 문자열은 작은따옴표로 끝내야 한다는 약속이다. 줄바꿈을 포함하여 여러 줄에 걸친 문자열을 표현하고 싶을 때도 있다. 이때는 작은따옴표나 큰따옴표 세 개로 문자열을 시작하고 같은 방식으로 닫으면 된다.

```
>>> multiline_string = """This is a multiline string  
with "newline" characters  
within the string"""  
>>> print(multiline_string)  
This is a multiline string  
with "newline" characters  
within the string
```

따옴표 3개로 시작해서 끝나는
문자열 안에는 큰따옴표, 작은
따옴표를 모두 사용할 수 있어
요

1.13 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.

1.13 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
```

1.13 파이썬이 정말로 편리한 이유 : 모듈 설치하기

- 미리 만들어진 모듈이 설치된 상태라면 이제 사용하기 위해 불러 오는 것이 필요하다.
- 이를 위해서는 'import'와 함께 모듈 이름을 써 주면 되고, 사용할 때에는 모듈 이름에 점(.)을 찍은 후 모듈 안의 구성요소를 호출해 주면 된다.
- 다음은 현재 스크립트나 대화창에서 모듈을 불러와서 이 모듈에 있는 func라는 이름의 기능을 사용하는 방법이다.

```
import module-name  
module-name.func()
```

1.14 터틀 그래픽으로 눈에 보이는 결과를 만들어 보자

- 파이썬이 초보자에게 좋은 점 중의 하나는 화면에 그림을 그리기가 쉽다는 점이다. 파이썬은 **터틀 그래픽** `turtle graphic`이 지원된다.
- 터틀 그래픽은 2차원 화면에 로봇 거북이가 있다고 가정하고 이 거북이에게 명령을 내려서 그림을 그리는 기능이다.



1.14 터틀 그래픽으로 눈에 보이는 결과를 만들어 보자

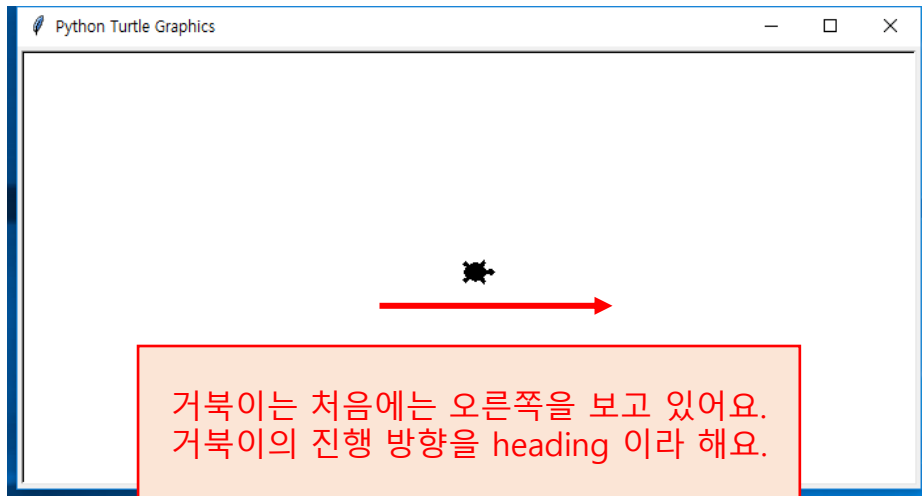
- 터틀 그래픽은 다음과 같이 동작한다.
 1. 터틀 그래픽을 시작하면 종이의 한 가운데 거북이가 나타난다.
 2. 거북이에게 명령을 내리면 거북이가 움직인다. 예를 들어서 "앞으로 전진", "뒤로 후진", "왼쪽으로 방향 전환" 등의 명령을 사용할 수 있다.
 3. 거북이가 움직이면서 종이 위에 그림이 그려진다. 거북이가 펜을 가지고 움직인다고 생각하면 된다.

1.14 터틀 그래픽으로 눈에 보이는 결과를 만들어 보자

- 파이썬 프롬프트에서 다음과 같이 입력하여 보자.

```
>>> import turtle           # turtle 모듈을 사용하기 위해 준비한다
>>> t = turtle.Turtle()     # turtle 모듈에 있는 Turtle 클래스 객체를 t로 생성한다
>>> t.shape('turtle')       # Turtle 클래스 객체인 t의 모양을 거북이 모양으로 설정한다
```

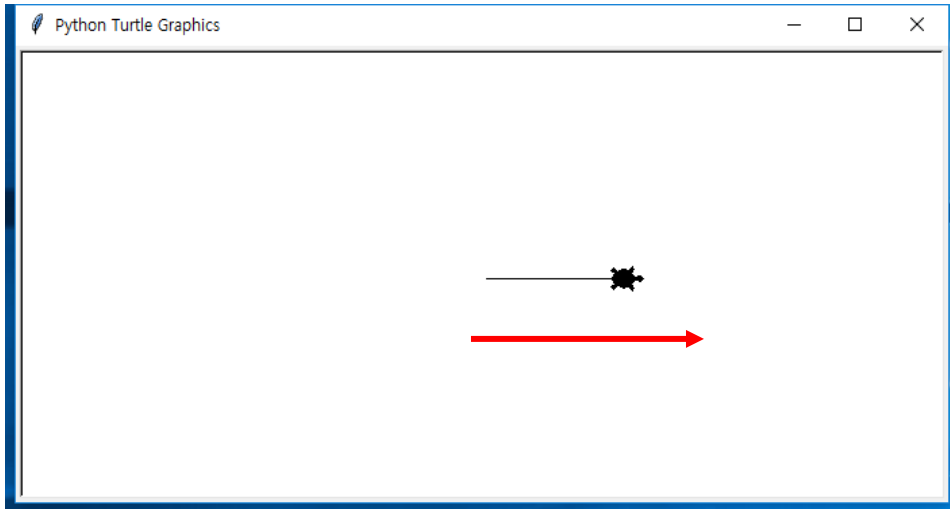
- 위의 3개의 문장을 입력하면 화면에 “Python Turtle Graphics”라는 이름의 캔버스가 나타난다.



1.14 터틀 그래픽으로 눈에 보이는 결과를 만들어 보자

- 지금부터 거북이를 움직이면 캔버스에 그림이 그려진다.
다음과 같은 명령어를 이용하여 거북이를 앞으로 전진시켜보자.

```
>>> t.forward(100)    # 거북이를 전진(forward)시키는 명령어로 100픽셀 만큼 이동
```

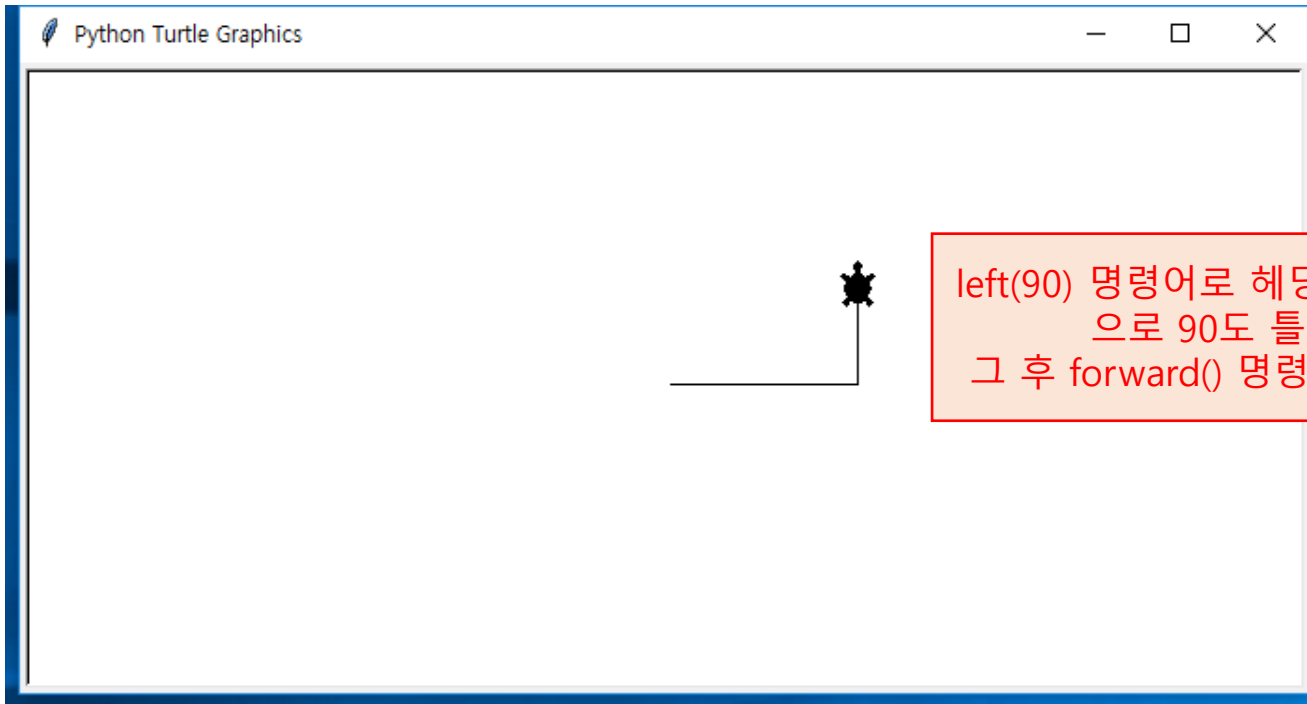


- 거북이가 움직이면서 직선이 그려지는 것을 볼 수 있다. 여기서 100은 100픽셀을 의미한다. 즉 거북이가 앞으로 100 픽셀(pixel)을 움직이면서 직선을 그리게 된다.

1.14 터틀 그래픽으로 눈에 보이는 결과를 만들어 보자

- 이번에는 거북이를 왼쪽으로 회전시킨 후에 전진시켜보자.

```
>>> t.left(90)      # 거북이를 왼쪽으로(left) 90도 회전시키는 명령어
>>> t.forward(50)
```



left(90) 명령어로 헤딩방향을 왼쪽
으로 90도 틀어줌.
그 후 forward() 명령어로 전진함



잠깐 - 픽셀이란 무엇일까

픽셀pixel은 컴퓨터 이미지를 이루는 가장 작은 단위의 점을 뜻한다. 픽셀은 **그림**picture의 **원소**element라는 의미를 가진다. 한자어로는 **화소**畵素라고 부른다. 이 용어는 13장에서 상세히 다룬다.



도전문제 1.2

forward()와 left() 기능을 사용하여 한 변의 길이가 100 픽셀인 도형을 그려보자.

- (1) 거북이를 움직여서 정삼각형을 그려보자(힌트 : 왼쪽으로 120도 회전하도록 하자).
- (2) 거북이를 움직여서 정사각형을 그려보자.
- (3) 거북이를 움직여서 한변의 길이가 50 픽셀인 정육각형을 그려보자. 회전하는 각도를 몇 도로 하여야 하는가?
- (4) 거북이를 10회이상 회전시키고 이동시켜서 임의의 도형을 자유롭게 그려보자.

1.15 오류가 발생할 수 있다

- 프로그래밍 언어를 비교할 때, 파이썬이 무척 관대하고 참을성이 많다고 할 수 있다. 그렇다고 개발자가 아무렇게나 입력하면 안된다. 컴퓨터는 인간과 달라서 상식이 전혀 없고 사정을 봐주지 않는다.
- 프로그래머는 프로그래밍 언어의 문법을 엄격히 지켜야 한다.
우리가 외국인과 영어로 이야기할 때에 문법을 지키지 않으면 의사소통이 "어려운" 정도이지만, 프로그래밍 언어에서 문법을 지키지 않으면 의사소통이 "불가능"하다.

1.15 오류가 발생할 수 있다

- 예를 들어서 다음과 같이 입력해보자.

```
>>> PRINT("Hello World")
Traceback (most recent call last):
  File "<ipython-input-2-ab351b16d57b>", line 1, in <module>
    PRINT("Hello World")
NameError: name 'PRINT' is not defined
```



- 컴퓨터는 위와 같은 문장을 입력하면 빨간색으로 **오류**Error를 출력한다. 문법을 지키라는 이야기이다.
- 오류 메시지를 보면 파이썬이 이해할 수 없는 명령을 입력했다고 말하고 있다. print라고 해야 할 것을 PRINT라고 입력한 것이다.
- 파이썬은 대소문자를 구별하기 때문에 PRINT라고 하면 이것을 어떻게 처리해야 되는지 알지 못한다.

1.15 오류가 발생할 수 있다

- 텍스트를 출력할 때도 따옴표를 생략하면 오류가 발생한다. 이 때 다음과 같이 ^로 오류의 위치와 오류의 원인을 알려준다.

```
>>> print(Good Bye)
File "<ipython-input-4-0389bd3941f5>", line 1
print(Good Bye)
^
SyntaxError: invalid syntax
```

- 오류 메시지를 만나면 당황하지 말고 침착하게 읽어 보자. 오류 메시지는 여러분이 실수한 부분을 찾아 고칠 수 있도록 유용한 정보를 제공해 준다.
- 오류 메시지는 잘못을 지적하는 것이 아니라 여러분을 도와주는 친절한 도우미라고 생각하자.
- 오류 메시지를 꼼꼼히 읽고 오류의 유형들에 익숙해지면 여러분의 코딩 실력도 부쩍 늘게 될 것이다.

오류 메시지를 침
착하게 잘 읽어보
면 **어디에서, 왜**
오류가 발생했는
지 알수 있지요.





잠깐 - 오류 메시지를 즐겁게 읽어보자

오류 메시지를 만나면 당황하지 말고 침착하게 읽어 보자. 오류 메시지는 여러분이 실수한 부분을 찾아 고칠 수 있도록 **유용한 정보를 제공**해 준다. 오류 메시지는 잘못을 지적하는 것이 아니라 여러분을 도와주는 친절한 도우미라고 생각하자. **오류 메시지를 꼼꼼히 읽고** 오류의 유형들에 익숙해지면 여러분의 코딩 실력도 부쩍 늘게 될 것이다.



잠깐 - 대표적인 오류 메시지만 알아도 당신은 프로그래머로 한 걸음 더 나갈 수 있다

SyntaxError: invalid syntax - 파이썬 언어의 약속된 문법 규칙을 지키지 않은 표현이 나타남
IndentationError: expected an indented block - 필요한 들여쓰기를 하지 않은 오류
IndentationError: unexpected indent - 들여쓰기를 하지 않아야 할 곳에서 글을 들여쓴 오류
NameError: name x is not defined - 무언가 가리키는 이름이 사용되었는데 뭔지 알 수 없을때
TypeError: Can't convert ... - 데이터의 종류가 다른 것들이 서로 값을 주고 받을 때

LAB¹⁻¹ 자주 사용하게 될 print() 함수를 연습해 보자

print() 함수를 사용하여 다음과 같이 출력하는 소스를 대화형 모드로 작성해보자. 곱하기 결과를 출력할 때는 실제로 파이썬으로 9*8 연산을 하여서 9*8의 결과를 알려주는 출력을 만들어 보라.

원하는 결과

```
>>> _____
안녕하세요? 여러분
>>> _____
저는 파이썬을 무척 좋아합니다.
>>> _____
9*8은 72 입니다.
>>> _____
안녕히 계세요.
```

파이썬의 대화형 모드를 사용하여 출력문을 작성하여 실행해본다. 문제에서 요구하는 세 번째 결과는 하나의 문자열이 아니라, 문자열과 계산 결과값 등 여러 개의 대상을 출력하는 문제이다. 화면에 여러 가지 값을 출력할 때는 쉼표로 값들을 분리하여 print() 함수로 전달하면 된다. 곱하기 연산을 하려면 9*8과 같이 적어준다. 세 번째 문제는 다음과 같은 방식으로 해결할 수 있을 것이다.

```
>>> print("9*8은", 9*8, "입니다.")
9*8은 72 입니다.
```

LAB¹⁻¹ 자주 사용하게 될 print() 함수를 연습해 보자

```
>>> print("안녕하세요? 여러분")
안녕하세요? 여러분
```

```
>>> print("저는 파이썬을 무척 좋아합니다.")
저는 파이썬을 무척 좋아합니다.
```

```
>>> print("9*8은", 9*8, "입니다.")
9*8은 72 입니다.
```

```
>>> print("안녕히 계세요.")
안녕히 계세요.
```



도전문제 1.3

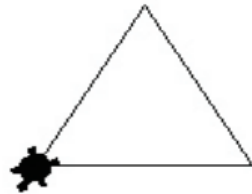
살펴본바와 같이 `print('1 + 1 = ', 1 + 1)` 과 같이 출력문과 연산을 사용하면 화면에 `1 + 1 = 2`가 출력된다. 50과 30 두 값이 있을 경우, 이 값에 대한 사칙 연산을 다음과 같은 결과를 출력하는 프로그램을 작성하시오.

```
50 + 30 = 80
50 - 30 = 20
50 * 30 = 1500
50 / 30 = 1.6666666666666667
```

LAB¹⁻² 터틀 그래픽으로 삼각형을 그려 보자

터틀 그래픽을 이용하여 삼각형을 그려보자. 삼각형의 좌표는 신경 쓰지 말고 모습이 정삼각형이 되도록 거북이를 움직여 보자. 이때 한 변의 길이는 100픽셀로 만들어라.

원하는 결과



삼각형을 그리는데 필요한 명령은 전부 입력하여 실행시키는 것으로 하자. 터틀 그래픽을 사용하려면 다음과 같이 터틀 그래픽 모듈을 불러들여야 한다.

```
>>> import turtle
```

터틀 그래픽을 사용하여서 화면에 그림을 그리려면 캔버스가 필요하다. 캔버스는 다음과 같이 펜을 생성하여야 나타난다.

```
>>> t=turtle.Turtle()
```

거북이를 움직이면 그림이 그려진다.

```
>>> t.forward(100)
```

거북이를 회전시키는 명령은 `left()`와 `right()`이다. `left()`와 `right()`는 입력받은 각도만큼 거북이를 회전시킨다. 삼각형을 그려야 하므로 `t.left`의 괄호안에 들어갈 각도는 120도로 하면 될 것이다.

```
>>> t.left(120)
```

LAB¹⁻² 터틀 그래픽으로 삼각형을 그려 보자

```
>>> import turtle
>>> t = turtle.Turtle()
>>> t.shape("turtle")
>>> t.forward(100)
>>> t.left(120)
>>> t.forward(100)
>>> t.left(120)
>>> t.forward(100)
```



잠깐 - 스크립트 모드에서 터틀 그래픽을 사용할 때는 한 줄 더 추가해야 한다.

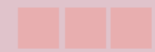
이 실습은 대화창 모드에서 입력하는 것을 기준으로 해답이 제시되었다. 스크립트 모드에서 터틀 그래픽을 사용할 경우에는 스크립트를 다 작성하고 마지막에 터틀 모듈의 `done()` 함수를 불러 화면이 사라지지 않게 유지해야 한다. 즉 스크립트 모드의 해답은 다음과 같다.

```
import turtle
t = turtle.Turtle()
t.shape("turtle")
t.forward(100)
t.left(120)
t.forward(100)
t.left(120)
t.forward(100)
turtle.done() # 주의 : t.done() 이 아님
```




summary

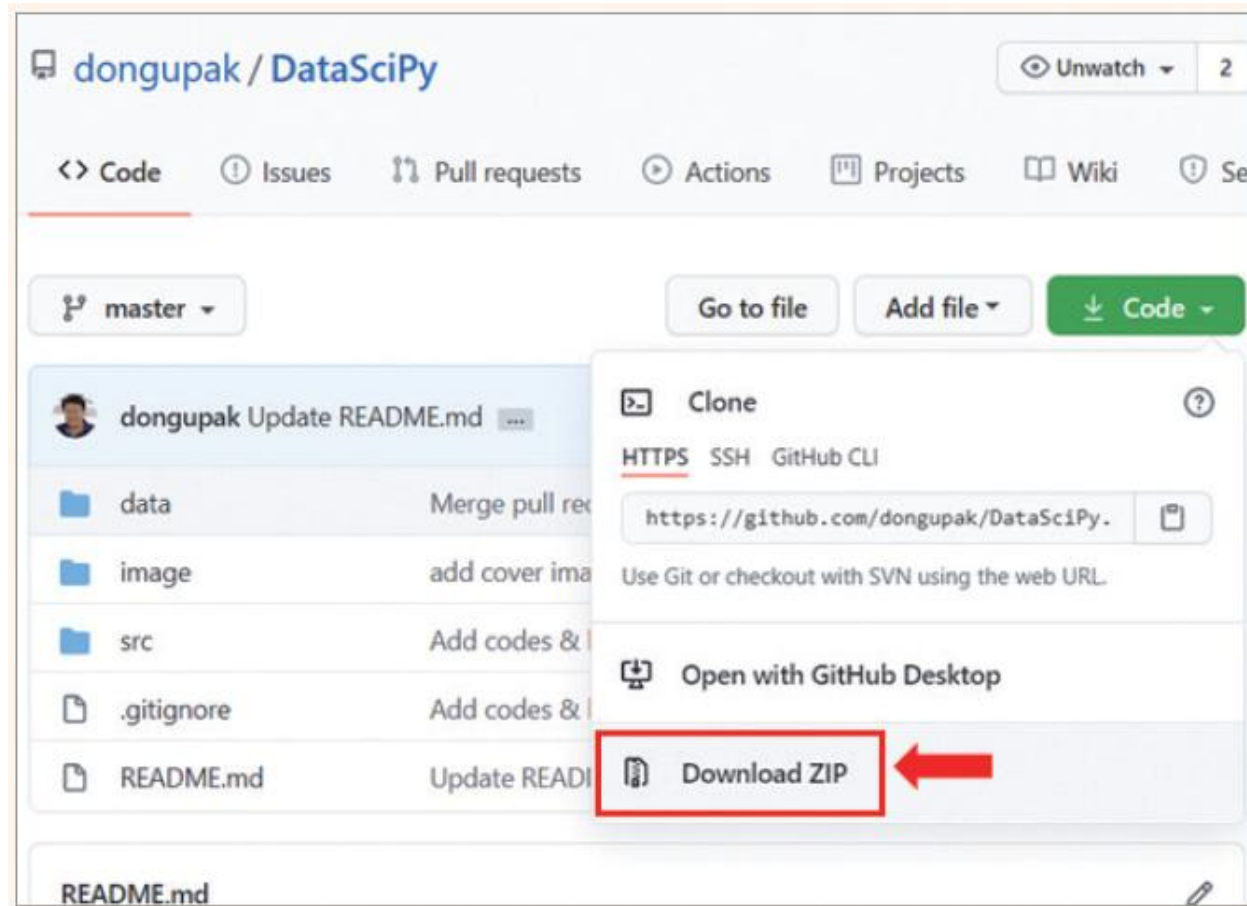
핵심 정리

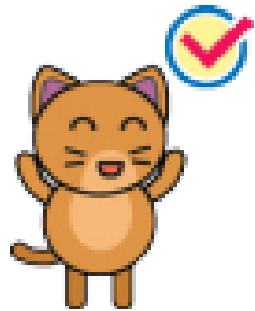


- 프로그램은 컴퓨터에 내리는 명령으로 이루어지는 작업지시서이다.
- 프로그래밍 언어는 컴퓨터가 이해할 수 있는 언어이다.
- 다양한 종류의 프로그래밍 언어가 있고 파이썬도 프로그래밍 언어의 일종이다.
- Python 콘솔에서는 프롬프트 다음에 코드를 입력하고 엔터키를 누르면 코드가 실행된다.
- 산술 계산을 하는 파이썬 연산자에는 $+$, $-$, $*$, $/$ 가 있다.
- `print()`는 화면에 문자열이나 계산 결과를 출력할 수 있다.
- 스크립트 모드를 사용하면 코드를 파일에 저장하였다가 한꺼번에 실행할 수 있다.

이 책의 코드를 다음 웹사이트에서 다운 받으세요

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





Questions?