

Python 기본문법

렉스

임기태 강사 (KarL)

프로그래밍이란?

- 프로그래밍 (programming)
 - 프로그램을 만드는 것
- 프로그램 (program)
 - 미리 작성된 것

Pro + Gram = ProGram
미리 + 작성된 것 = 미리 작성된 것

프로그램 = 미리 작성된 것 = 진행 계획

프로그래밍 언어

- 이진 숫자 (binary digit)
 - 0과 1로 이루어진 컴퓨터의 언어
 - 이진 코드 (binary code) : 이진 숫자로 이루어진 코드
- 프로그래밍 언어 (programming language)
 - 사람이 이해하기 쉬운 언어로 프로그램을 만들기 위해 만들어짐
- 소스 코드 (source code)
 - 프로그래밍 언어로 작성한 프로그램
- 코드 실행기
 - 프로그래밍 언어는 컴퓨터가 이해할 수 없으므로
이를 이진 숫자로 변환해 주는 역할

파이썬이란?

- 파이썬 (Python)
 - 1991년 귀도 반 로섬 (Guido van Rossum)이 개발
 - 초보자가 쉽게 배울 수 있는 프로그래밍 언어
- 파이썬의 장점
 - 비전공자도 쉽게 배울 수 있음
 - 다양한 분야에서 활용할 수 있음
 - 대부분의 운영체제에서 동일하게 사용됨
- 파이썬의 단점
 - C언어에 비해 일반적으로 10~350배 느림
 - 최근에는 컴퓨터 성능이 좋아져 연산이 많이 필요한 프로그램이 아니라면 차이 크게 느낄 수 없음

파이썬 설치하기

- 파이썬 설치 프로그램 다운로드

1) 파이썬 공식 홈페이지 (<http://www.python.org>) 접속 – [Downloads]

2) [Download Python 3.9.1] 클릭

The screenshot shows the Python.org homepage. The navigation bar includes links for Python, PEP, Docs, PyPI, Jobs, and Community. The main content area features a dark blue background with a code snippet on the left and a 'Compound Data Types' article on the right. Below this, there are four main sections: 'Get Started', 'Download', 'Docs', and 'Jobs'. The 'Download' section is highlighted, showing the latest Python version (3.9.1) and a link to the download page. The 'Jobs' section mentions a new job board. At the bottom, there are sections for 'Latest News' and 'Upcoming Events'.

Python

python™

About Downloads Documentation Community Success Stories News Events

```
# Python 3: List comprehensions
my_fruits = ["Banana", "Apple", "Lime"]
my_fruit_tuple = tuple(my_fruits)
print(my_fruit_tuple)

# List and the enumerate function
my_fruit_list = ["Banana", "Apple", "Lime"]
for index, fruit in enumerate(my_fruit_list):
    print(index, fruit)
```

Compound Data Types

Lists (present as arrays in other languages) are one of the compound data types that Python understands. Lists can be indexed, sliced and manipulated with other built-in functions. [More about lists in Python 3](#)

Python is a programming language that lets you work quickly and integrate systems more effectively. [Learn More](#)

Get Started
Whether you're new to programming or an experienced developer, it's easy to learn and use Python.
[Start with our Beginner's Guide](#)

Download
Python source code and installers are available for download for all versions!
Latest: Python 3.9.1

Docs
Documentation for Python's standard library, along with tutorials and guides, are available online.
[docs.python.org](#)

Jobs
Looking for work or have a Python-related position that you're trying to hire for? Our [new job board](#) is the place to go.
[jobs.python.org](#)

Latest News [10 More](#)

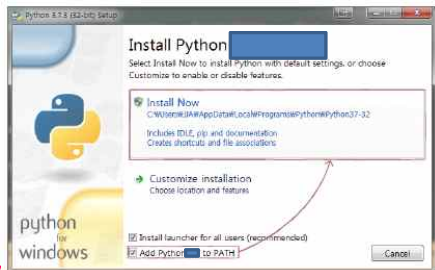
2021-03-03	Python 3.10.0b1 is now available for testing
2021-03-13	Python Software Foundation - January 2021 Newsletter
2021-03-12	2020 in Review
2021-03-04	Python 3.10.0b1 is now available for testing
2020-13-38	Election Before Community Update

Upcoming Events [10 More](#)

2021-02-09	PyCascades 2021
2021-03-18	PyCon Cameroon 2021
2021-03-02	Python Web Conference 2021
2021-04-22	GeoPython 2021
2021-05-12	PyCon UK 2021

- 파이썬 설치하기

- 1) 설치 프로그램 실행 아래 화면에서 [Add Python 3.9.1 to PATH] 체크
- 2) [Install Now] 클릭



여기를 체크하지 않고 설치하면 파이썬이 실행되지 않으므로 꼭 확인합니다.

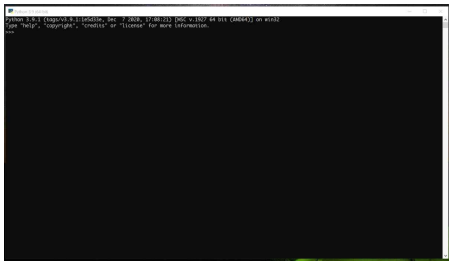
3) 설치 완료 화면이 나타나면 [Close] 클릭

4) 윈도우 [시작] 메뉴에서 [Python 3.9.1] 프로그램 확인



파이썬 : 인터프리터

- 인터프리터 (interpreter)
 - 파이썬으로 작성된 코드를 실행해주는 프로그램
- 파이썬 인터랙티브 셸
 - 파이썬 명령어를 한 줄씩 입력하며 실행결과 볼 수 있는 공간



파이썬 : 인터프리터

- 프롬프트 (prompt)

- >>>

- 코드를 한 줄씩 입력

- 인터랙티브 셸 = 대화형 셸

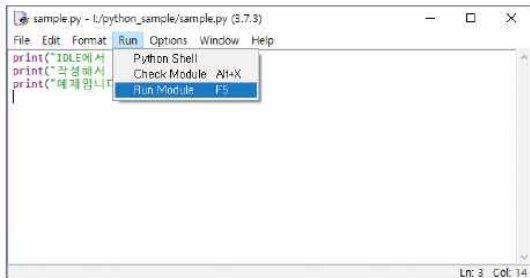
: 컴퓨터와 상호 작용하는 공간이며, 한 마디씩 주고받는 것처럼

대화한다고 하여 대화형 셸로 불리기도 함

```
>>> 10 + 10 Enter → 10 + 10을 입력하니  
20 → 10과 10을 더해 20을 출력합니다.  
  
>>> "Hello" * 3 Enter → Hello라는 문자열을 3번 출력하라는 의미이며,  
'HelloHelloHello' → 'HelloHelloHello'를 출력합니다.  
  
>>>
```

파이썬 : 인터프리터

6) [Run] – [Run Module] 메뉴 선택(혹은 [F5] 단축키)

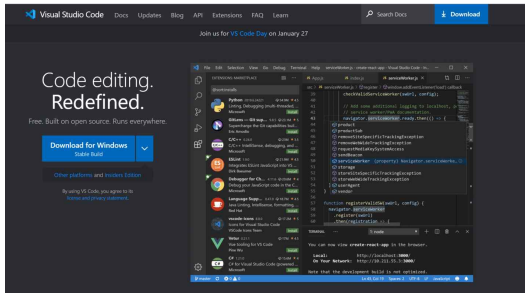


텍스트 에디터 사용하기 : 비주얼 스튜디오 코드

- 비주얼 스튜디오 코드 다운로드해 설치하기

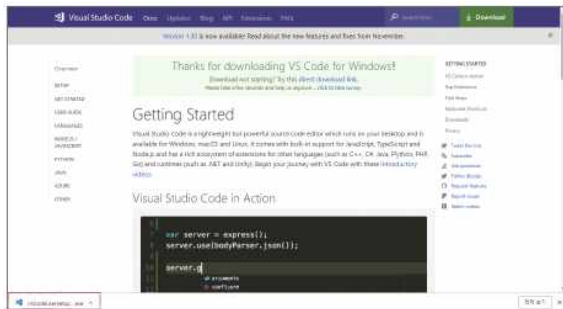
- 홈페이지 (<https://code.visualstudio.com>) 접속

1) [Downloads for Windows] 클릭하여 설치 파일 다운로드



VSCode 설치

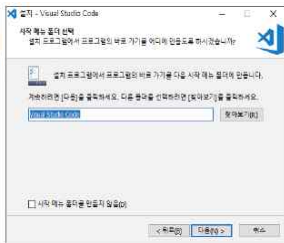
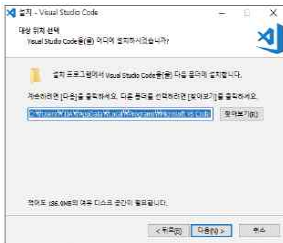
2) 다운로드한 설치 프로그램을 실행



VSCode 설치

5) 설치 폴더 지정 후 [다음] 클릭

6) 시작 메뉴 폴더 이름 지정 후 [다음] 클릭

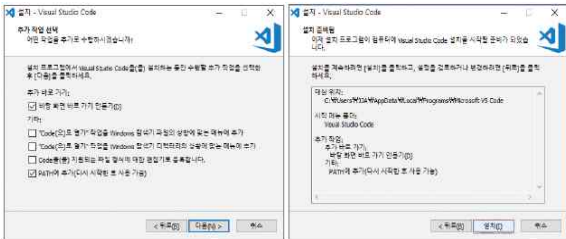


VSCode 설치

7) [바탕 화면 바로 가기 만들기] 체크 후 [다음] 클릭

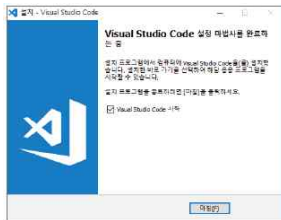
나머지 부분도 작업 시 유용하므로 모두 체크

8) 대상 위치, 시작 메뉴 폴더, 추가 설정 항목 확인 후 [설치] 클릭



VSCode 설치

9) [마침] 클릭. 설치 완료!



VSCode 설치

- 비주얼 스튜디오 코드 한글 언어 팩 설치

- 1) 도구 바에서 [확장] 클릭
- 2) 검색 창에 [korean] 입력
- 3) [Korean Language Pack for Visual Studio Code]의 [Install] 클릭



VSCode 설치

4) 설치 완료 후 [Restart Now] 버튼 클릭하여 재시작



VSCode 설치

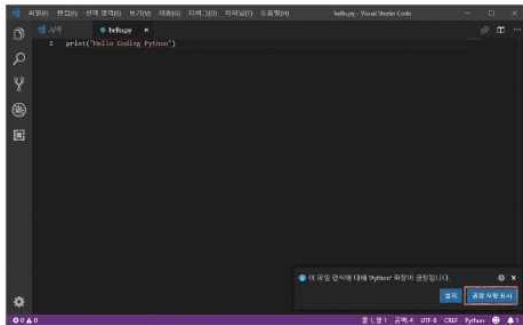
5) 메뉴가 한글로 바뀌었음을 확인



VSCode 설치

- 비주얼 스튜디오 코드에서 코드 작성하고 실행하기

1) 시작 화면에서 [파일] - [새 파일] 메뉴 선택 (단축키 [Ctrl] + [N])



VSCode 설치

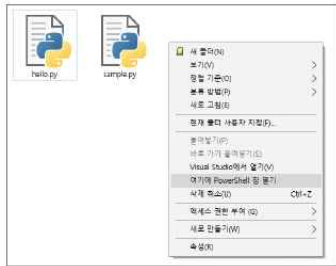
2) [확장] 메뉴에서 [Python] 클릭 - [설치] - [다시 로드]



VSCode 설치

3) 탐색기에서 코드 파일 저장한 폴더로 이동

4) [Shift] 누른 상태로 빈 곳을 마우스 오른쪽 클릭 - [여기에 명령 창 열기]



VSCode 설치

5) 해당 폴더에서 명령 프롬프트가 실행

6) [python hello.py] 입력 후 [Enter] 클릭하면

```
> python hello.py Enter
```

7) Hello Coding Python 출력

```
Hello Coding Python
```

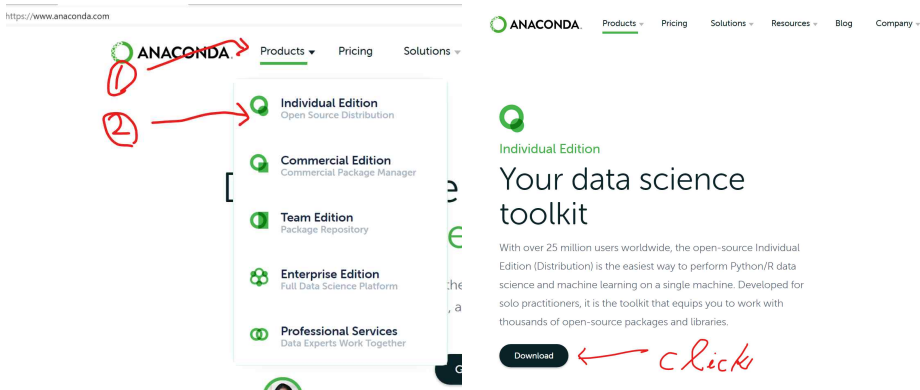
VSCode 설치

- 파이썬을 하려면 파이썬 코드를 입력할 수 있는 **텍스트 에디터**와 파이썬 코드를 실행할 수 있는 도구인 **파이썬 인터프리터**가 필요하다.
- 파이썬은 프롬프트라 불리는 >>>에 코드를 입력하면 바로 실행결과를 볼 수 있는데, 이는 한 마디씩 주고받는 것처럼 대화한다고 하여 **인터랙티브 셸 (대화형 셸)** 이라고 한다.
- 파이썬으로 작성한 파일은 해당 폴더의 명령 프롬프트에서 **python 명령어**로 실행할 수 있다.

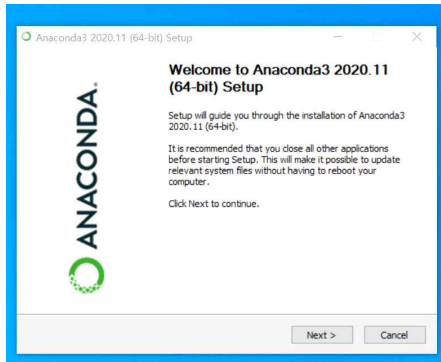
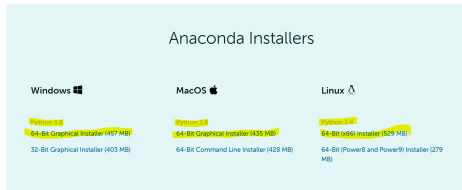
Anaconda 설치하기



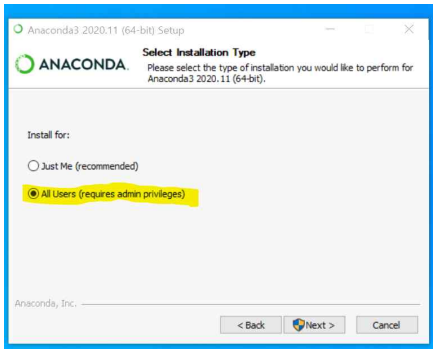
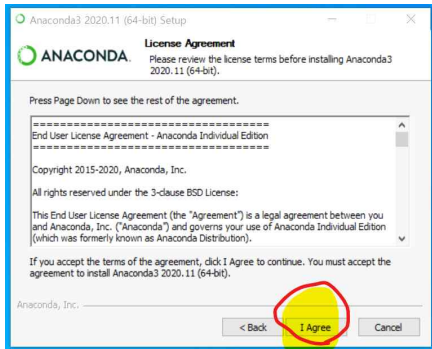
Anaconda 설치하기 <https://www.anaconda.com>



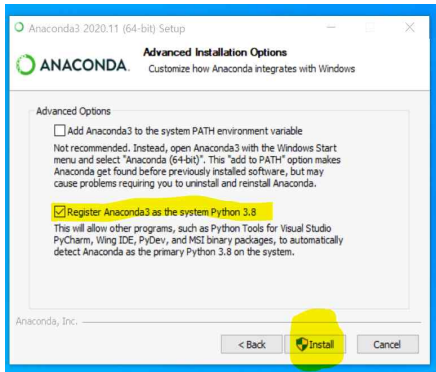
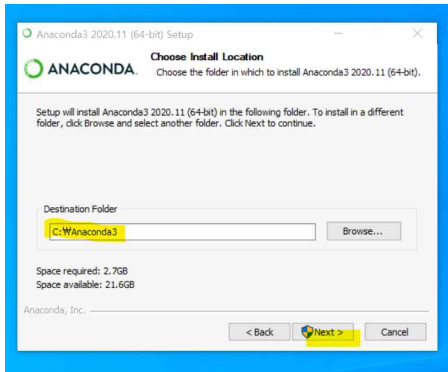
Anaconda 설치하기



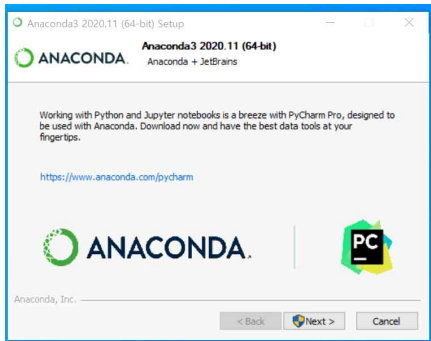
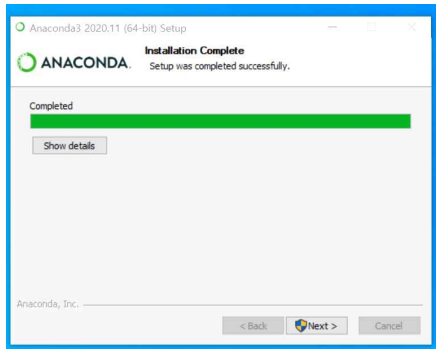
Anaconda 설치하기



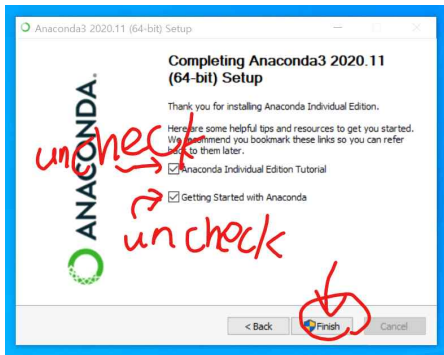
Anaconda 설치하기



Anaconda 설치하기



Anaconda 설치하기



Installation Success

Welcome to Anaconda!

Here are some useful resources to help you get started.

Create your free [Anaconda Nucleus](#) account today to get access to training materials, how-to videos, and expert insights, all free for a limited time to Nucleus members.

[Register for Free](#)

Individual Edition Tutorial

This quick 12-minute tutorial provides an introduction to help you get started using this powerful tool.

[Watch Tutorial](#)

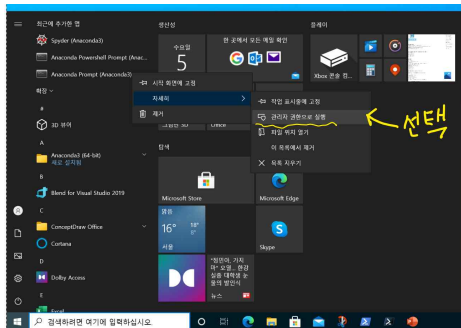
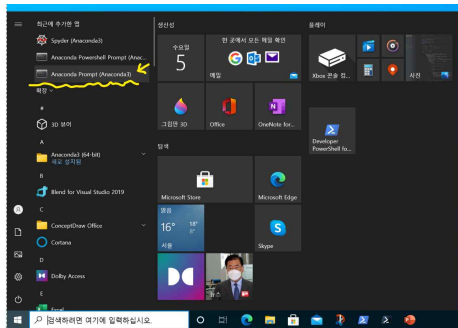
Quick Start Guide

Learn how to use Anaconda Individual Edition, Anaconda Navigator, and conda with cheat sheets, FAQs, and more.

[Learn More](#)

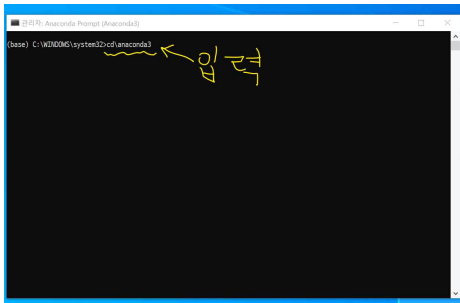
Anaconda 설치하기

마우스 오른쪽 버튼 사용하기



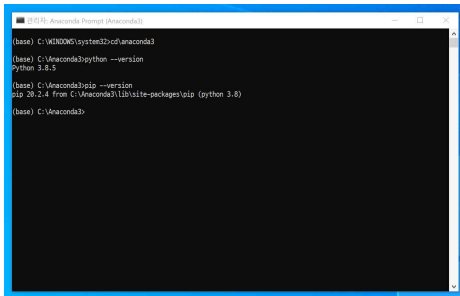
Anaconda 설치하기

버전 체크하기



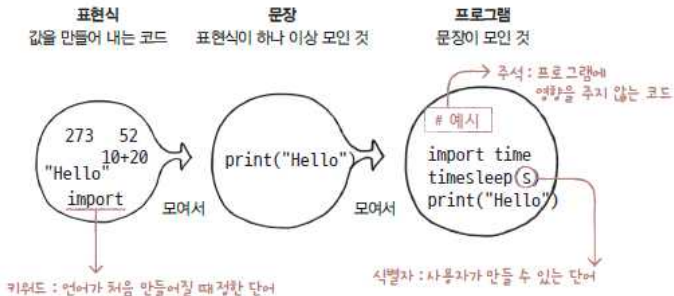
```
(base) C:\WINDOWS\system32>cd\anaconda3
```

A yellow wavy line underlines the path `cd\anaconda3`, and a yellow arrow points to it with the Korean text "입력" (input).



```
(base) C:\WINDOWS\system32>cd\anaconda3
(base) C:\Anaconda3>python --version
Python 3.8.5
(base) C:\Anaconda3>pip --version
pip 20.2.4 from C:\Anaconda3\lib\site-packages\pip (python 3.8)
(base) C:\Anaconda3>
```


프로그램 기초



표현식과 문장

- 표현식 (expression)
 - 파이썬에서 어떠한 값을 만들어내는 간단한 코드
 - 값이란 숫자 수식, 문자열 등이 될 수 있음

```
273
```

```
10 + 20 + 30 * 10
```

```
"Python Programming"
```

```
print("Python Programming")
```

표현식과 문장

- **문장** (statement)

- 표현식이 하나 이상 모일 경우
- 그 자체로 어떠한 값을 만들 수 없으면 문장이 아님

+

-

- **프로그램** (program)

- 문장이 모여서 형성

키워드

- 키워드 (keyword)

- 특별한 의미가 부여된 단어
- 파이썬에서 이미 특정 의미로 사용하기로 예약해 놓은 것
- 프로그래밍 언어에서 이름 정할 때 똑같이 사용할 수 없음

False	None	True	and	as	assert
break	class	continue	def	del	elif
else	except	finally	for	from	global
if	import	in	is	lambda	nonlocal
not	or	pass	raise	return	try
while	with	yield			

- 대소문자 구별

키워드

- 아래 코드로 특정 단어가 파이썬 키워드인지 확인 가능

```
>>> import keyword  
>>> print(keyword.kwlist)
```

```
['False', 'None', 'True', 'and', 'as', 'assert', 'async', 'await', 'break',  
'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for',  
'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or',  
'pass', 'raise', 'return', 'try', 'while', 'with', 'yield']
```

식별자

- 식별자 (identifier)

- 프로그래밍 언어에서 이름 붙일 때 사용하는 단어
- 변수 또는 함수 이름 등으로 사용
- 키워드 사용 불가
- 특수문자는 언더바(_)만 허용
- 숫자로 시작 불가
- 공백 포함 불가
- 알파벳 사용이 관례
- 의미 있는 단어로 할 것

사용 가능한 단어	사용 불가능한 단어	
alpha	break	→ 키워드라서 안 됩니다.
alpha10		
_alpha	273alpha	→ 숫자로 시작해서 안 됩니다.
AlpHa		
ALPHA	has space	→ 공백을 포함해서 안 됩니다.

스네이크 케이스



```
library(snakecase)

to_any_case(
  string = c("YYYY_MM_DD_bla_bla_bla",
             "2019_01-09_bla_bla-bla"),
  sep_out = c("", "", "-", "_"),
  postfix = ".txt")
#> [1] "yyyymmdd-bla_bla_bla.txt"
#> [2] "20190109-bla_bla_bla.txt"
```

식별자

- 스네이크 케이스와 캐멀 케이스

itemlist	loginstatus	characterhp	rotateangle
----------	-------------	-------------	-------------

- 공백이 없어 이해하기 어려움
 - 스네이크 케이스 (snake case) : 언더바(_)를 기호 중간에 붙이기
 - 캐멀 케이스 (camel case) : 단어들의 첫 글자를 대문자로 만들기

식별자에 공백이 없는 경우	단어 사이에 _ 기호를 붙인 경우 (스네이크 케이스)	단어 첫 글자를 대문자로 만든 경우 (캐멀 케이스)
itemlist loginstatus characterhp rotateangle	item_list login_status character_hp rotate_angle	ItemList LoginStatus CharacterHp RotateAngle

- 파이썬에서는 스네이크

식별자

- 식별자가 클래스인지, 변수인지, 함수인지 구분해 봅시다


```
1. print()
2. list()
3. soup.select()
4. math.pi
5. math.e
6. class Animal:
7. BeautifulSoup()
```

주석

- 주석 (comment)

- 프로그램 진행에 영향 주지 않는 코드
- 프로그램 설명 위해 사용
- # 기호를 주석으로 처리하고자 하는 부분 앞에 붙임

```
>>> # 간단히 출력하는 예입니다.
>>> print("Hello! Python Programming...") # 문자열을 출력합니다.
Hello! Python Programming...
```



→ # 기호 뒷부분이 주석 처리됩니다.

연산자와 자료

- 연산자

- 스스로 값이 되는 것이 아닌 값과 값 사이에 무언가 기능 적용할 때 사용

```
>>> 1 + 1
2
>>> 10 - 10
0
```

- 리터럴 (literal)

- 자료 = 어떠한 값 자체

```
1
10
"Hello"
```

출력 print()

- print() 함수

- 출력 기능
- 출력하고 싶은 것들을 괄호 안에 나열

```
print(출력1, 출력2, ...)
```

- 하나만 출력하기

```
>>> print("Hello! Python Programming...")
Hello! Python Programming...
>>> print(52)
52
>>> print(273)
273
```

출력 : print()

- 여러 개 출력하기

```
>>> print(52, 273, "Hello")  
52 273 Hello  
  
>>> print("안녕하세요", "저의", "이름은", "윤인성입니다!")  
안녕하세요 저의 이름은 윤인성입니다!
```

- 줄바꿈하기

```
>>> print()  
→ 빈 줄을 출력합니다.  
  
>>>
```

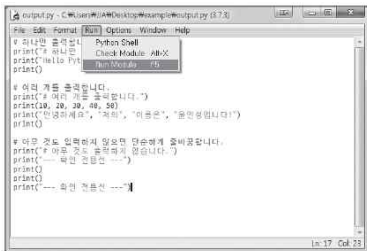
출력 : print()

- 예시 – 기본 출력

```
01  # 하나만 출력합니다.
02  print("# 하나만 출력합니다.")
03  print("Hello Python Programming...!")
04  print()
05
06  # 여러 개를 출력합니다.
07  print("# 여러 개를 출력합니다.")
08  print(10, 20, 30, 40, 50)
09  print("안녕하세요", "저의", "이름은", "윤연성입니다!")
10  print()
11
12  # 아무것도 입력하지 않으면 단순히 줄바꿈합니다.
13  print("# 아무것도 출력하지 않습니다.")
14  print("— 확인 전용선 —")
15  print()
16  print()
17  print("— 확인 전용선 —")
```

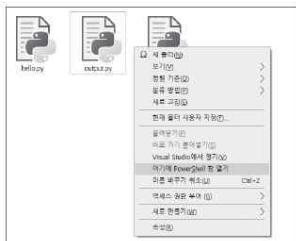
출력 : print()

- 파이썬 IDLE 에디터에서의 실행
 - [Run] – [Run Module] 선택



출력 : print()

- 비주얼 스튜디오 코드에서의 실행
 - 탐색기에서 파일 저장한 폴더로 이동하여 Shift 누른 상태로 마우스 우클릭
 - [여기에 PowerShell 창 열기]



출력 : print()

- 명령 프롬프트 실행되면 python 명령어 사용하여 해당 파일 실행

```
> python output.py
```

```
# 하나만 출력합니다.
```

```
Hello Python Programming...!
```

```
# 여러 개를 출력합니다.
```

```
10 20 30 40 50
```

```
안녕하세요 저의 이름은 윤인성입니다!
```

```
# 아무것도 출력하지 않습니다.
```

```
— 확인 전용선 —
```

```
— 확인 전용선 —
```

파이썬의 특징

1. 직관적이고 쉽다.

아주 간단한 영어 문장을 읽듯이 보고 쉽게 이해할 수 있도록 구성

2. 널리쓰인다.

구글, 아마존, 핀터레스트, 인스타그램, IBM, 디즈니, 야후, 유튜브, 노키아, NASA 등과 네이버, 카카오톡의 주력 언어 중 하나

3. 개발 환경이 좋다.

게임, 인공지능, 수치해석 등 다양한 라이브러리와 커뮤니티 활성화

IDLE 파이썬 셸(Python Shell)

- **파이썬 셸** : 직접 파이썬 명령을 입력하고 엔터 키를 누르면 바로 그 결과가 셸 화면에 출력
- 파이썬 셸을 계산기처럼 사용

```
>>> 10 + 20
```

```
>>> 10 + 20 * 30
```

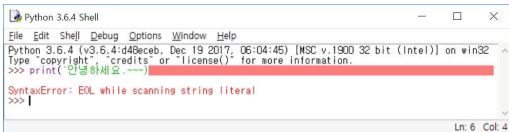
```
>>> 10 * 20 + 30 * 50
```

IDLE 파이썬 셸(Python Shell)

- '안녕하세요~~~'를 화면에 출력

```
>>> print('안녕하세요~~~')  
안녕하세요~~~  
>>>
```

- 오류 메시지 출력



The screenshot shows a window titled 'Python 3.6.4 Shell'. The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The text area contains the following content:

```
Python 3.6.4 (v3.6.4:d48ceeb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32  
Type "copyright", "credits" or "license()" for more information.  
>>> print('안녕하세요~~~')  
SyntaxError: EOL while scanning string literal  
>>> |
```

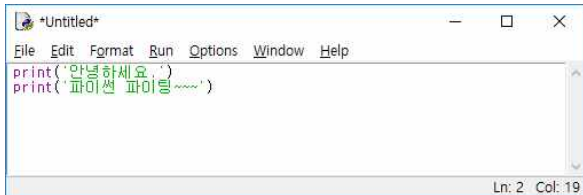
The error message 'SyntaxError: EOL while scanning string literal' is displayed in red. The status bar at the bottom right shows 'Ln: 6 Col: 4'.

텍스트 에디터

- (1) 메모장 : 기능은 뛰어나지 않지만 모든 컴퓨터에 설치되어 있기 때문에 간단한 프로그램 작성하기 편리
- (2) 서브라임 텍스트(Sublime Text) : 유료/무료, 무료 버전으로도 기능이 막강하여 학교와 기업 등에서 많이 사용
- (3) 비주얼 스튜디오(Visual Studio) : 유료/무료, 마이크로소프트사에서 개발한 프로그램으로 C, 자바 등 기존 프로그래머들이 많이 사용, 무료 버전으로도 충분히 사용 가능
- (4) IDLE의 텍스트 에디터 : 앞 절에서 설치한 IDLE 프로그램에 기본으로 내장된 텍스트 에디터로 파이썬 프로그래밍에 최적화

파이썬 프로그램 작성

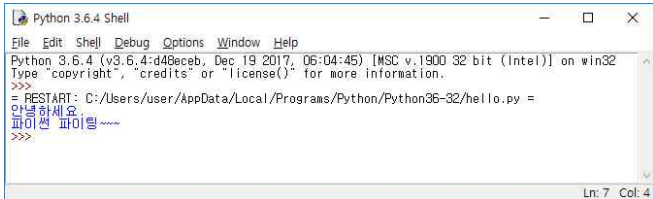
- 텍스트 에디터



- 파일 저장 : hello.py

프로그램 실행하기

- 파이썬 셸에서 단축키 'F5' 누름

A screenshot of a 'Python 3.6.4 Shell' window. The window has a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Window', and 'Help'. The main text area shows the following content: 'Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32', 'Type "copyright", "credits" or "license()" for more information.', '>>>', '= RESTART: C:/Users/user/AppData/Local/Programs/Python/Python36-32/hello.py =', '안녕하세요.', '파이썬 파이팅 ~~~~', and '>>>'. The status bar at the bottom right indicates 'Ln: 7 Col: 4'.

```
Python 3.6.4 Shell
File Edit Shell Debug Options Window Help
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/user/AppData/Local/Programs/Python/Python36-32/hello.py =
안녕하세요.
파이썬 파이팅 ~~~~
>>>
Ln: 7 Col: 4
```

변수란?

- 변수(Variable)는 값을 저장하는 박스
- 변수를 만든다는 것은 숫자나 문자열과 같은 데이터를 저장할 수 있는 공간을 마련하는 것
- 수학의 방정식에서의 $x + y = 3$ 에서 x 와 y 는 어떤 변하는 값을 가지게 되는 변수
- 컴퓨터에서 변수도 수학 변수의 개념과 유사

변수 값의 저장과 출력

```
① >>> a = 5  
② >>> print(a)  
5
```

```
① >>> a = 3  
   >>> b = 7  
② >>> c = a + b  
③ >>> print(c)  
10
```

변수명의 규칙

- 변수명은 영문자 소문자로 시작
- 유효한 변수명 : a, b, x, y, l, j, str, animal, computer, age, sum, type1, type2, num1, num2 ...
- 잘못된 변수명 : 12month, 10rule, 3numer

```
① >>> animal = '사자'
   >>> print(animal)
   사자
② >>> num1 = 7.8
   >>> num2 = 3.57
   >>> print(num1 + num2)
   11.37
③ >>> 12month = '봄'
   SyntaxError: invalid syntax
```

변수명의 규칙

- 변수명은 영문자, 숫자, 밑줄(_)의 조합

```
① >>> x = 3
   >>> y = 7
② >>> font1 = '돋움'
③ >>> my_age = 25
   >>> screen_width = 1024
④ >>> myAge = 18
   >>> screenWidth = 2048
```

변수명의 규칙

- 변수명을 다음과 같이 사용하는 것은 좋지 않음

```
>>> aaa = '돌음'  
>>> xxx = 37  
>>> abc = 10.5
```

- 변수명에 특수문자(&, *, (,), %, \$, #, @, , !), 공백, 한글 사용 금지

```
① >>> email@ = 'test@naver.com'
```

```
SyntaxError: invalid syntax
```

```
② >>> my age = 30
```

```
SyntaxError: invalid syntax
```

정수형 숫자

- 정수형(Integer) : 음수, 0, 양수로 구성된 숫자

```
① >>> 1 + 2 + 3
6
② >>> a = -10 + 10 + (-30 - 40)
>>> print(a)
-70
③ >>> print(10 + 20 + 30)
60
```

실수형 숫자

- 실수형(Floating Point) : -0.37, -33.0, 37.33에서와 같이 소수점을 가진 숫자

① >>> 128.8 + 38 - 222.4764

-55.6764

② >>> a = 2/3

>>> print(a)

0.6666666666666666

③ >>> print('%.2f' % a)

0.67

변수의 형 알아보기

① >>> a = 123

>>> type(a)

<class 'int'>

② >>> b = 123.45

>>> type(b)

<class 'float'>

숫자 연산자

- 사칙 연산자 : 더하기(+), 빼기(-), 곱하기(*), 나누기(/)
- 나머지 연산자(%) : 어떤 수로 나눈 나머지를 계산
- 소수점 절삭 연산자(//) : 소수점 이하를 절삭
- 제곱 연산자(**) : 어떤 수의 제곱

사칙 연산자 : +, -, *, /

```
① >>> a = 10 + 20 * 30
>>> print(a)
610
② >>> (10 + 20) * 30
900
③ >>> b = 10 - 20 / 10
>>> print(b)
8.0
④ >>> type(a)
<class 'int'>
⑤ >>> type(b)
<class 'float'>
```

나머지 연산자 : %

① >>> a = 17 % 5

>>> print(a)

2

② >>> b = 29 % 6

>>> print(b)

5

③ >>> c = a % b

>>> print(c)

2

>>>

소수점 절삭 연산자 : //

```
① >>> 10 / 3  
3.333333333333335
```

```
② >>> 10 // 3  
3
```

제곱 연산자 : **

① >>> 2**3

8

② >>> 3**4

81

숫자 연산자 정리

연산자	설명
+	더하기
-	빼기
*	곱하기
/	나누기
%	나머지 연산
//	나눈 후 소수점 이하 절삭
**	제곱 구하기

문자열

- 문자열(String) : 하나 또는 여러 개의 문자로 구성된 데이터형
- 문자들의 앞과 뒤에 쌍 따옴표(") 또는 단 따옴표(')를 붙임
- "안녕하세요."와
'안녕하세요.'는 동일

문자열의 추출

```
① >>> word = 'apple'
>>> print(word)
apple
② >>> word[0]
'a'
③ >>> word[1]
'p'
④ >>> word[0:3]
'app'
```

※ 인덱스는 1이 아니라 0부터 시작

전화번호는 숫자일까? 문자열일까?

- 컴퓨터에서 숫자란 연산이 적용될 수 있는 수
- 전화번호에다 값을 더하거나 빼거나 하는 연산을 하는 것이 아님
- 전화번호는 문자열로 표현
- '123-1234' 또는 "123-1234"에서와 같이 전화번호 앞 뒤에 따옴표로 감싸야 함
- 주소에서 사용되는 번지수나 동이나 호수도 문자열로 처리

퀴즈

1. 주민등록 번호(XXXXXXXX-XXXXXX)의 데이터 형으로 적합한 것은?

- ① 정수형 ② 문자열 ③ 실수형

2. 다음에 나타난 파이썬 셸 명령의 실행 결과는?

```
>>> a = '우리는 민족중흥의 역사적 사명을 띠고 이 땅에 태어났다.'
```

```
>>> a[3:12]
```

- ① '는 민족중흥의 역사' ② '는 민족중흥의 역'
③ '민족중흥의 역사' ④ ' 민족중흥의 역사'

문자열 연결 연산자 : +

① >>> name = '홍지영'

>>> print(name)

홍지영

② >>> greet = '안녕하세요!'

>>> print(greet)

안녕하세요!

③ >>> print(name + '님 ' + greet)

홍지영님 안녕하세요!

문자열 반복 연산자 : *

```
① >>> a = 'blue' * 5  
>>> print(a)  
  
blueblueblueblueblue  
  
② >>> print('=' * 30)  
  
=====
```

퀴즈

1. 다음의 파이썬 명령에 사용된 연결 연산자 +에 사용된 변수의 형이 달라 오류가 발생한다. 변수 kor과 변수 score의 형은 각각 어떻게 되는가?

```
>>> kor = '국어 성적 : '
```

```
>>> score = 80
```

```
>>> string = kor + score
```

- ① 정수형, 문자열 ② 실수형, 정수형 ③ 정수형, 실수형 ④ 문자열, 정수형

2. 다음 중 문자열을 반복하는 데 사용되는 연산자는?

- ① * ② // ③ % ④ +

퀴즈

3. 다음은 문자열 인덱스를 이용하여 특정 문자를 추출하는 예이다. 프로그램의 실행 결과는?

```
>>> date = '20191025'  
>>> year = date[0:4]  
>>> month = date[4:6]  
>>> day = date[6:]  
>>> date2 = year + '-' + month + '-' + day  
>>> print(date2)
```

- ① 2019/10/25 ② 20191025 ③ 2019 10 25 ④ 2019-10-25

문자열 길이 구하기 : len()

```
>>> message = '안녕하세요!'
```

①

```
>>> str_len = len(message)
```

②

```
>>> print('문자열의 길이 : ' + str(str_len))
```

```
문자열의 길이 : 6
```


키보드로 입력받기 : input()

```
>>> person = input('이름을 입력하세요: ')
```

```
이름을 입력하세요: 강지영
```

```
>>> print(person + '님 안녕하세요~~~')
```

```
강지영님 안녕하세요~~~
```

- Input() 함수

```
input('질문_내용')
```

키보드로 정수 입력받기

① >>> a = input('첫 번째 정수를 입력하세요: ')

첫 번째 정수를 입력하세요: 36

② >>> b = input('두 번째 정수를 입력하세요: ')

두 번째 정수를 입력하세요: 24

③ >>> c = a + b

>>> print(c)

3624

정수형 숫자 36 vs. 문자열 '36'

- 36

정수형 숫자 36은 컴퓨터에서 십진수 36이 이진수로 표현되어 100100와 같은 값을 가짐

- '36'

문자열 '36'은 '3'의 대한 이진 코드 00110011과 '6'에 대한 이진 코드 00110110이 연결된 값인 0011001100110110와 같은 값을 가짐

※ 컴퓨터에서 36과 '36'은 전혀 다른 값임

데이터 형 변환에 사용되는 함수

- **int()**

함수 int()는 실수Floating point나 문자열String을 정수형 숫자로 변환

- **float()**

함수 float()는 정수나 문자열을 실수로 변환

- **str()**

함수 str()은 정수형이나 실수형 숫자를 문자열로 변환

퀴즈

1. 다음에 나타난 파이썬 셸 명령의 실행 결과는?

```
>>> a = input('첫 번째 정수를 입력하세요: ')
```

첫 번째 정수를 입력하세요: 22

```
>>> b = input('두 번째 정수를 입력하세요: ')
```

두 번째 정수를 입력하세요: 33

```
>>> c = a + b
```

```
>>> print(c)
```

① 3322 ② 55 ③ 22 33 ④ 2233

퀴즈

2. 다음에 나타난 파이썬 셸 명령의 실행 결과는?

```
>>> a = input('첫 번째 정수를 입력하세요: ')
```

첫 번째 정수를 입력하세요: 55

```
>>> b = input('두 번째 정수를 입력하세요: ')
```

두 번째 정수를 입력하세요: 60

```
>>> c = int(a) + int(b)
```

```
>>> print(c)
```

① 오류가 발생한다 ② 115 ③ 5560 ④ 6650

화면에 출력하기 : print()

- 컴퓨터 화면에 결과를 출력할 때 print() 함수 사용
- print() 함수를 사용법 4가지
 1. print() 함수의 기본 사용
 2. 파라미터 sep을 사용
 3. 문자열 연결 연산자 +를 사용
 4. 문자열 포맷 코드 %를 사용

print() 함수의 기본 사용법

- print() 함수의 각 항목을 콤마(,)로 구분

print(..., 변수, ..., 수식, ..., 값, ...)

```
① >>> a = 10
>>> print(a)
10
>>> b = 20
② >>> print(a + b)
30
③ >>> print(a + 10, b + 10)
```


sep을 이용한 출력

- 키워드 sep은 'seperator'의 약어로서 항목 사이에 삽입할 문자열을 지정하는 데 사용

```
>>> hp1 = '010'
>>> hp2 = '1234'
>>> hp3 = '5678'
① >>> print(hp1, hp2, hp3, sep='-')
010-1234-5678
```

NULL이란?

- 컴퓨터에 NULL은 값이 없는 것을 의미
- "" 또는 "와 같이 표기
- 0은 정수의 0 값을 의미
- 공백 ' ' 은 따옴표(') 사이에 하나의 공백 문자가 들어가 있음

연결 연산자 +를 이용한 출력

```
① >>> name = input('이름을 입력하세요: ')
    이름을 입력하세요: 홍소영

② >>> age = input('나이를 입력하세요: ')
    나이를 입력하세요: 23

③ >>> print(name + '님의 나이는 ' + age + '세 입니다!')
    홍소영님의 나이는 23세 입니다!
```

문자열 포매팅을 이용한 출력

```
>>> a = 77
```

```
>>> b = '자전거'
```

```
>>> c = 3.3737737
```

```
>>> d = 90
```

```
>>> print('%d, %s, %.2f, %d%%, %6s, %5d' % (a, b, c, d, b, a))
```

①

```
77, 자전거, 3.37, 90%,   자전거,   77
```

문자열 포매팅 코드의 예

포매팅 코드	설명
%d	정수형 숫자
%s	문자열
%.2f	실수형 숫자, .2는 소수점 둘 째 자리까지 나타냄.
%%	% 기호 자체를 나타내는 데 사용함.
%6s	6자리의 문자열
%5d	5자리의 정수형 숫자

퀴즈

1. print() 함수를 이용하여 변수 값을 출력할 때 각 필드를 구분할 때 사용하는 키워드는?

- ① div ② src ③ split ④ sep

2. 다음에 나타난 파이썬 셸 명령의 실행 결과는?

```
>>> hp1 = '010'
```

```
>>> hp2 = '1234'
```

```
>>> hp3 = '5678'
```

```
>>> print(hp1, hp2, hp3, sep='-')
```

- ① 010/1234/5678 ② 01012345678 ③ 010 1234 5678 ④ 010-1234-5678

퀴즈

3. 다음은 파이썬 셸에서 국어, 영어, 수학 세 과목의 성적을 입력 받아 합계와 평균을 구하는 예이다. 밑줄 친 곳에 들어갈 내용은?

```
>>> kor = input('국어 성적을 입력하세요: ')
국어 성적을 입력하세요: 90
>>> eng = input('영어 성적을 입력하세요: ')
영어 성적을 입력하세요: 80
>>> math = input('수학 성적을 입력하세요: ')
수학 성적을 입력하세요: 100
>>> sum = (1)_____(kor) + (1)_____(eng) + (1)_____(math)
>>> (2)_____ = sum / 3
>>> print('합계 : (3)_____, 평균 : %.2f' % (sum, avg))
합계 : 270, 평균 : 90.00
```

① int, float, sum ② float, avg, sum ③ int, avg, %d ④ int, sum, avg

주석문

- 프로그램을 짤 때 프로그램의 작성자, 작성한 날짜, 프로그램의 기능, 코드에 대한 주석, 즉 설명 글을 다는 데 사용되는 문장
- 파이썬 주석문의 두 가지 방식

주석 기호	설명
#	한 줄의 주석 처리
""" 또는 '''	여러 줄의 주석 처리

주석문 삽입하기

①

```
"""
```

```
print() 함수를 이용한 데이터 출력
```

```
- 작성자 : 황재호
```

```
- 일자 : 2018.4.6
```

②

```
"""
```

③

```
print('안녕하세요.')          # 화면에 '안녕하세요.' 출력
```

④

```
print('파이썬 파이팅~~~')    # 화면에 '파이썬 파이팅~~~' 출력
```

조건문이란?

- 조건문 : 해당 조건에 따라 다른 코드를 실행
- '만약 ~하면 ~ 하다'와 같은 상황에서 사용
- 사용 예
 - 1) 만약 점수가 80점 이상이면 합격이고 80점 미만이면 불합격이다
 - 2) 만약 나이가 65세 이상일 경우에는 입장료가 무료이다
 - 3) 주민번호 앞자리가 1이면 남성이다
 - 4) 비밀번호가 맞으면 로그인된다

If문

- 어떤 수가 양수인지 아닌지를 판단하여 결과를 출력

```
if  $x > 0$ :
```

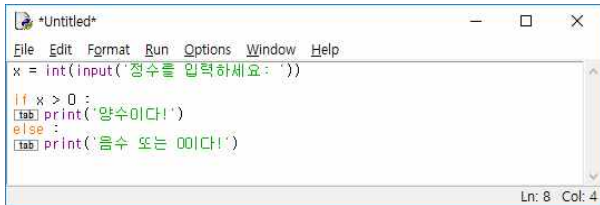
```
    print('양수이다!')
```

```
else :
```

```
    print('음수 또는 0이다!')
```

어떤 수가 양수인지 판별

- IDLE 파이썬 쉘에서 상단 메뉴에 있는 File > New File를 클릭하고 다음의 내용을 작성

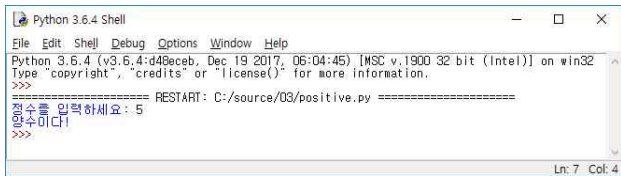


```
*Untitled*  
File Edit Format Run Options Window Help  
x = int(input('정수를 입력하세요: '))  
  
if x > 0 :  
    print('양수이다!')  
else :  
    print('음수 또는 0이다!')
```

Ln: 8 Col: 4

- 저장할 파일명 : positive.py
- 프로그램 실행 : 단축키 F5

positive.py 실행 결과



```
Python 3.6.4 Shell
File Edit Shell Debug Options Window Help
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/source/03/positive.py =====
정수를 입력하세요: 5
양수이다
>>>
```

Ln: 7 Col: 4

If문의 세 가지 유형

if문의 유형	사용 형태
(1) if ~ 구문	만약 이 조건을 만족하면 ~ 해라!
(2) if ~ else ~ 구문	만약 이 조건을 만족하면 ~ 하고, 그렇지 않으면 ~ 해라!
(3) if ~ elif ~ else ~ 구문	만약 조건1을 만족하면 ~ 하고, 조건2를 만족하면 ~하고, ..., 그렇지 않으면 ~ 해라!

비교 연산자

비교 연산자	설명
$a < b$	a는 b보다 작다
$a > b$	a는 b보다 크다
$a == b$	a는 b와 같다
$a != b$	a는 b와 같지 않다
$a \leq b$	a는 b보다 작거나 같다
$a \geq b$	a는 b보다 크거나 같다

비교 연산자 사용 예

① >>> 3 == 3

True

② >>> 8 >= 3

True

③ >>> 8 < 3

False

비교 연산자 사용 예

```
>>> a = 5  
>>> b = 2  
① >>> a > b  
True  
② >>> a == b  
False  
③ >>> a % 2 == 0  
False
```

논리 연산자

논리 연산자	설명
조건1 and 조건2	조건1과 조건2가 둘 다 참이어야 전체 결과가 참
조건1 or 조건2	조건1과 조건2 중 하나만 참이어도 전체 결과가 참
not 조건	조건이 참이면 그 결과는 거짓, 조건이 거짓이면 그 결과는 참

논리 연산자 : and

```
>>> pilgi = 85          # 필기 성적 85점
```

```
>>> silgi = 90          # 실기 성적 90점
```

```
① >>> pilgi >= 80 and silgi >= 80
```

```
True
```

```
② >>> silgi = 70
```

```
③ >>> pilgi >= 80 and silgi >= 80
```

```
False
```

논리 연산자 : or

```
>>> x = 20  
① >>> x %2 == 0 or x%3 == 0  
True  
>>> x = 11  
② >>> x %2 == 0 or x%3 == 0  
False
```

논리 연산자 : not

①

```
>>> x = 10
```

```
>>> not x==10
```

```
False
```

②

```
>>> x = 5
```

```
>>> not x%2==0
```

```
True
```

if~ 구문

if 조건식 :

<문장1, 2, ...>

- 조건식이 참이면 <문장 1, 2,>를 수행
- 조건식이 거짓이면 <문장1, 2, ...>를 수행하지 않음