

# 프로그래밍 개념, 파이썬 특징

1주차\_01\_01

# 학습목표

2

- ▶ 프로그래밍이 무엇인지 이해
- ▶ 파이썬 활용도가 높은 이유를 파악
- ▶ 파이썬 개발 역사와 파이썬 특징 알기

# 프로그래밍이란 무엇인가?

- ▶ 프로그래밍은
  - ▶ 일련의 명령들(instruction)의 나열로  
계산과정(computation)을 어떻게 행해야 하는지  
기술하는 것이다
  - ▶ 문제 해결하는데 도움을 준다
    - ▶ 문제를 체계적으로 구성하는 능력
    - ▶ 창의적 해결 방법과 과정 제시
    - ▶ 해결책에 대해 정확하며 명료하게 제시
- 프로그램에 대하여 배우는 과정은 문제 해결 능력을  
키울 수 있는 탁월한 기회를 제공한다

# 프로그래밍 언어

- ▶ 고차원 언어(high level language)
  - ▶ C, C++, Perl, Python, and Java
  - ▶ 컴파일러; 실행 파일을 생성하여 실행
- ▶ 인터프리터
  - ▶ 코드를 직접 실행한다
  - ▶ 일괄적으로 컴파일하여 실행화일을 만들지 않고 실행되도록 기능 제공

# 왜 파이썬을 활용하는가?(1/2)

5

- ▶ 파이썬은
  - ▶ 처음 코딩을 배우는 사람도 쉽게 시작할 수 있다
    - ▶ 프로그램을 작성 전에 알아야 하는 것이 적다
    - ▶ 명령어가 영어 일상용어와 유사하다
- ▶ 파이썬은 전산 비전공자도 널리 사용한다
  - ▶ numPy와 SciPy는 과학자들이 주로 사용한다
- ▶ 파이썬은 현대 언어
  - ▶ 웹 어플리케이션에 일반적으로 사용한다
  - ▶ 모바일 앱 개발에도 사용한다, Facebook 앱
  - ▶ 인공지능 개발 언어

# 왜 파이썬을 활용하는가?(2/2)

6

- ▶ 파이썬은
  - ▶ 일반적인 목적으로 인터프리트(Interpret)할 수 있는 고급 프로그래밍 언어
  - ▶ 문법이 간단
  - ▶ 가독성이 좋음
  - ▶ 종합적이고 큰 규모의 표준 라이브러리 제공
  - ▶ 다양한 프로그래밍 패러다임 제공
    - ▶ 객체지향 언어, 명령 지향 언어, 함수형 프로그래밍 방식

# 프로그래밍 언어, 개발자 활용빈도(2022년 12월)

- ▶ TIOBE(measure Your Software Code Quality)
  - ▶ Source <https://www.tiobe.com/tiobe-index/>
  - ▶ The TIOBE Programming Community index is an indicator of the popularity of programming languages. The index is updated once a month.
  - ▶ The ratings are based on the number of skilled engineers world-wide, courses and third party vendors.



# 프로그래밍 언어, 개발자 활용빈도(2022년 12월)

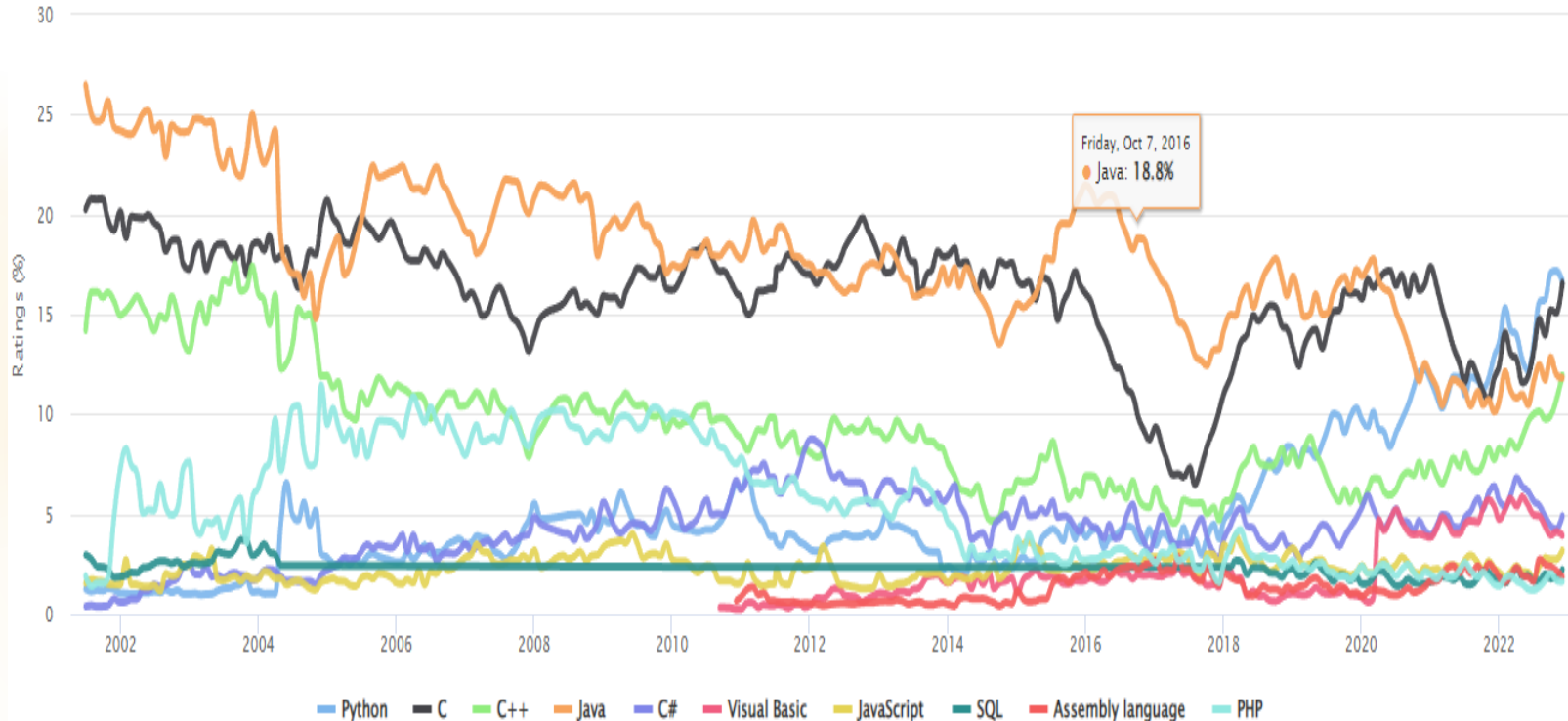
Programming Language	Ratings	Change	Programming Language	Ratings	Change
Python	16.66%	+3.76	R	1.25%	-0.34
C	16.56%	+4.77	Go	1.15%	+0.20
C++	11.94%	+4.21	Classic VB	1.15%	-0.13
Java	11.82%	+1.70	MATLAB	0.95%	+0.03
C#	4.92%	-1.48	Swift	0.91%	-0.86
Visual Basic	3.94%	-1.46	Delphi/Object Pascal	0.85%	-0.30
JavaScript	3.19%	+0.90	Ruby	0.81%	-0.35
SQL	2.22%	+0.43	Perl	0.78%	-0.18
Assembly language	1.87%	-0.38	Objective-C	0.71%	+0.29
PHP	1.62%	+0.12	Rust	0.68%	+0.23



# 프로그래밍 언어, 개발자 활용빈도 변화 추이

TIOBE Programming Community Index

Source: [www.tiobe.com](http://www.tiobe.com)



# 파이썬 개발 역사

10

- ▶ 개발의 시작
  - ▶ 1989년 12월
  - ▶ 네덜란드의 Guido van Rossum at CWI
    - ▶ .. Google, dropbox(from January, 2013)
- ▶ 파이썬 2.0
  - ▶ 2000년 10월 ~ 현재까지 (파이썬 2.7)
- ▶ 파이썬 3.0
  - ▶ 2008년 12월 ~ 현재까지 (파이썬 3.11)
  - ▶ 2.0과 하위 호환성 제공이 안됨



# 파이썬 2.x와 3.x 차이점

11

- ▶ 3.0 버전은 2.x대 버전과 하위호환성을 갖지 않는다
  - ▶ Python 2to3, python2를 python3로 바꿔주는 컨버터
- ▶ 2.x대 버전과의 차이점
  - ▶ 내장자료형의 내부적인 변화
  - ▶ 일부 구형의 구성 요소 제거 또는 조정
  - ▶ 표준 라이브러리 재배포
  - ▶ 한글 변수명 사용 가능
- ▶ Python 3.4 부터 Python 2에 비해 강력한 기능 제공

# 파이썬의 특징 (1/2)

- ▶ 다양한 패러다임을 지닌 프로그래밍 언어
  - ▶ 객체지향 프로그래밍, 구조적 프로그래밍 완벽 지원
- ▶ 고도로 확장 가능하게 설계
  - ▶ 프로그래밍이 가능한 인터페이스를 활용하여 기존의 어플리케이션에 포함 가능
- ▶ 다양한 표준 라이브러리 제공

# 파이썬의 특징 (2/2)

- ▶ 파이썬 언어 개발자의 핵심 목표는
  - ▶ 파이썬을 사용하기 재미있게 만들기
- ▶ 모든 사람들을 위한 컴퓨터 프로그래밍
  - ▶ 쉽고 직관적인 언어
    - ▶ 다른 메이저 경쟁 프로그래밍 언어와 동일한 기능 제공
    - ▶ 오픈소스라서 어떤 사람이라도 개발에 공헌 가능
  - ▶ 명령어가 평이한 영어 문장과 유사
  - ▶ 짧은 개발 시간 안에 일상업무 해결 가능

# 강의 요약

14

- ▶ 프로그래밍이란
  - ▶ 일련의 명령들의 나열로 계산과정을 어떻게 행해야 하는지 기술하는 것
  - ▶ 문제 해결에 도움을 줌
- ▶ 파이썬은
  - ▶ 다양한 패러다임을 지닌 프로그래밍 언어
  - ▶ 고도로 확장 가능하게 설계
  - ▶ 다양한 표준 라이브러리 제공

# 목표 달성 질문

15

- ▶ 프로그래밍을 정의하시오
- ▶ 파이썬이 많이 활용되는 이유를 나열하시오

# 파이썬 활용 분야

1주차\_01\_02



# 학습목표

17

- ▶ 파이썬이 동작하는 플랫폼, 사용되는 분야 알기
- ▶ 파이썬 언어의 특성을 이해
- ▶ 파이썬의 간단한 문법과 기능 알기
- ▶ 파이썬의 활용도 알기

# 동작하는 플랫폼

- ▶ 첫 버전은 매킨토시에서 사용할 목적으로 개발(Mac OS)
- ▶ 다양한 플랫폼 지원 확산
  - ▶ 마이크로소프트 윈도우
  - ▶ 유닉스 계열
    - ▶ 리눅스,
    - ▶ Solaris,
    - ▶ HP-UX,
    - ▶ AIX(IBM), IBM i(IBM power system)
  - ▶ iOS and iPadOS
  - ▶ OS/390: mainframe server OS
  - ▶ VMS: 고성능 서버용 OS
- ▶ 안드로이드 OS에서 사용하려면, Pydroid 3

# 문법의 특성

19

## ▶ 들여쓰기 이용한 블록 구조

// C code

```
int factorial( int x ) {  
    if (x==0) return 1;  
    else return x * factorial(x-1);  
}
```

들여쓰기 맞지  
않으면 코드가  
미 작동

# python code

```
def factorial(x):  
    if x==0:  
        return 1  
    else:  
        return x * factorial(x-1)
```

콜론으로 괄호를  
대신하거나,  
문법의 끝을  
표시한다

# 파이썬으로 할 수 있는 일(1/3)

20

- ▶ 시스템 유틸리티 제작
- ▶ GUI 프로그래밍(Tkinter)
- ▶ C/C++와의 결합
  - ▶ C/C++ 작성 프로그램을 파이썬에서 사용 가능
- ▶ 웹 프로그래밍
- ▶ 수치 연산 프로그래밍
  - ▶ NumPy 활용
- ▶ 인공지능 프로그래밍

# 파이썬으로 할 수 있는 일(2/3)

21

- ▶ 데이터베이스 프로그래밍
  - ▶ 사이베이스(Sybase), 인포믹스(Infomix), 오라클(Oracle), 마이에스큐엘(MySQL), 포스트그레스큐엘(PostgreSQL) 등의 데이터베이스에 접근할 수 있는 도구 제공
- ▶ 데이터 분석, 사물 인터넷
  - ▶ 판다스(Pandas)라는 모듈을 이용하여 데이터 분석

# 파이썬으로 할 수 있는 일(3/3)

22

- ▶ Web crawling
- ▶ Deep learning 구현(인공지능 개발)
- ▶ 게임
- ▶ 이미지 처리

# 파이썬 활용 기업

23

- ▶ Google
- ▶ Youtube
- ▶ Instagram
- ▶ DropBox
- ▶ Netflix
- ▶ Spotify
- ▶ Quora
- ▶ Astrageneca
- ▶ Exscientia

# 강의 요약 1

24

- ▶ 왜 파이썬을 활용하는가?
  - ▶ 문법은 간단, 가독성이 좋음
  - ▶ 종합적이고 큰 규모의 표준 라이브러리 제공
  - ▶ 다양한 프로그래밍 패러다임 제공
  - ▶ 처음 코딩을 배우는 사람도 쉽게 시작할 수 있음



# 강의 요약 2

25

- ▶ 다양한 플랫폼 지원
  - ▶ 마이크로소프트 윈도우
  - ▶ 매킨토시(맥 OS 9 이전, 맥 OS X 이후 포함)
  - ▶ 각종 유닉스, 리눅스
  - ▶ 팜 OS
  - ▶ 노키아 시리즈 60

# 목표 달성 질문

- ▶ 파이썬의 특성을 나열하시오
- ▶ 파이썬이 동작하는 플랫폼을 나열하시오
- ▶ 파이썬 2.x와 3.x의 차이점을 말해 보시오

# idle 설치, 윈도우용

1주차\_02\_01

# 학습목표

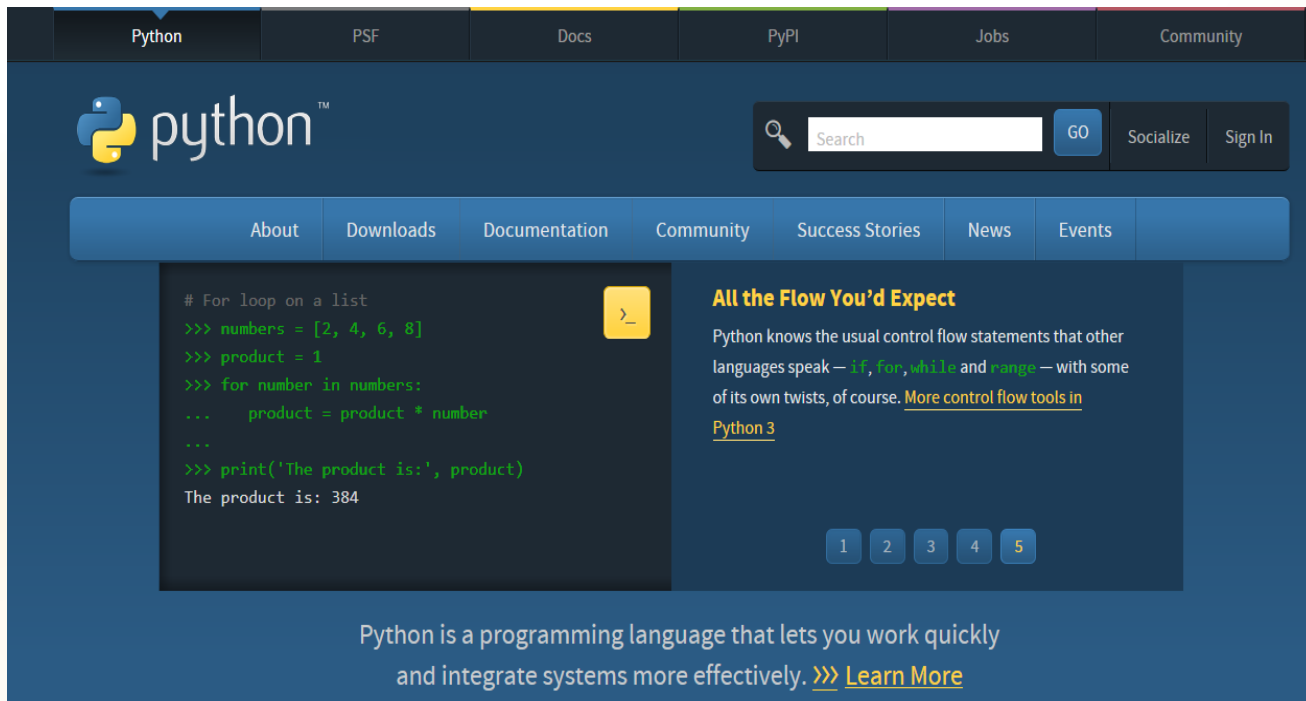
28

- ▶ 윈도우용 파이썬 IDLE를 설치해 보기

# Python.org

29

► <http://www.python.org>



The screenshot shows the Python.org homepage with a dark blue header and navigation bar. The header includes the Python logo and a search bar. The navigation bar has links for Python, PSF, Docs, PyPI, Jobs, and Community. Below the navigation bar, there are links for About, Downloads, Documentation, Community, Success Stories, News, and Events. The main content area features a code snippet on the left and an article titled "All the Flow You'd Expect" on the right. The code snippet demonstrates a for loop on a list. The article text explains that Python knows usual control flow statements like if, for, while, and range, with some twists. A link "More control flow tools in Python 3" is provided. At the bottom, a footer states: "Python is a programming language that lets you work quickly and integrate systems more effectively. >>> [Learn More](#)".

```
# For loop on a list
>>> numbers = [2, 4, 6, 8]
>>> product = 1
>>> for number in numbers:
...     product = product * number
...
>>> print('The product is:', product)
The product is: 384
```

### All the Flow You'd Expect

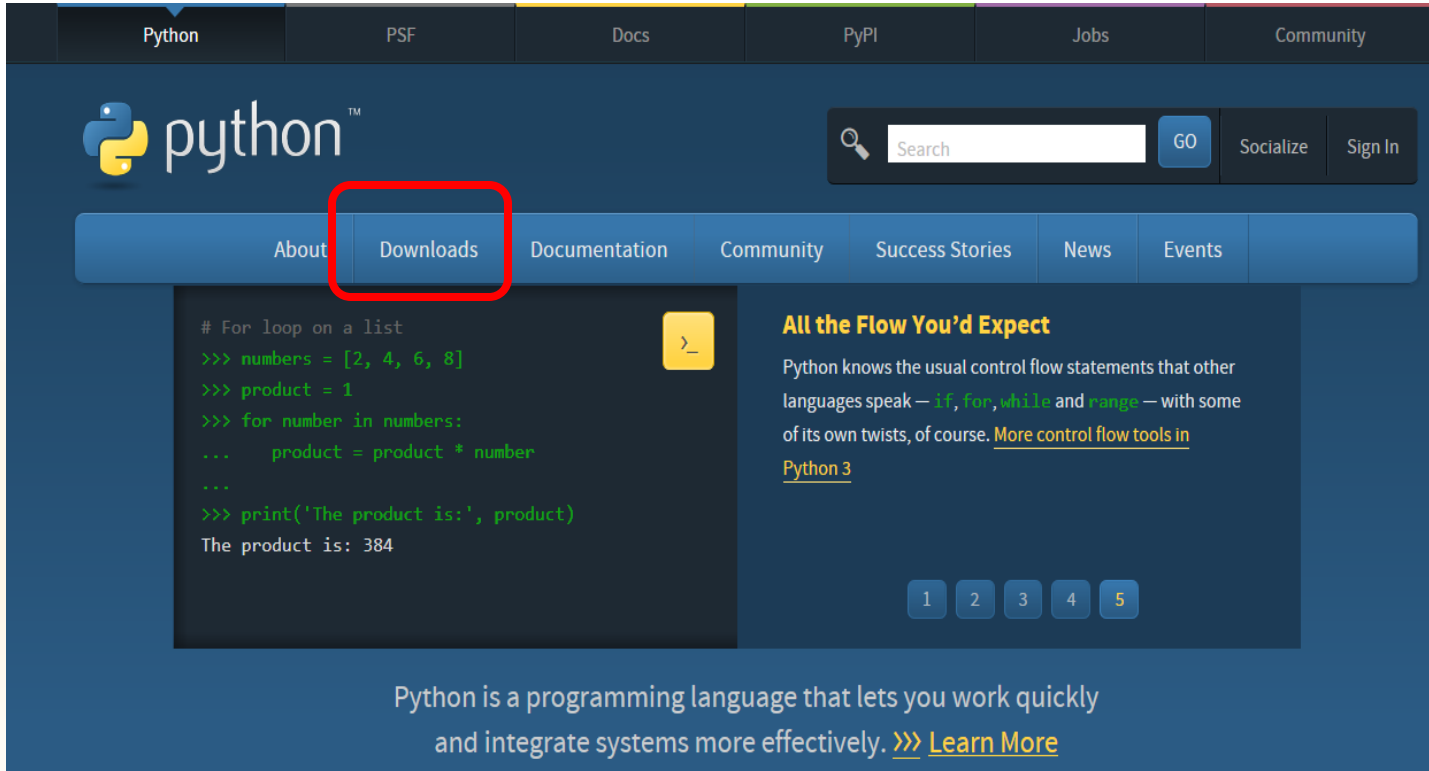
Python knows the usual control flow statements that other languages speak — `if`, `for`, `while` and `range` — with some of its own twists, of course. [More control flow tools in Python 3](#)

1 2 3 4 5

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

# Windows, 설치 과정(1/6)

30



The screenshot shows the Python.org homepage. At the top, there is a navigation bar with links for Python, PSF, Docs, PyPI, Jobs, and Community. Below this is a search bar with a magnifying glass icon, a 'GO' button, and links for 'Socialize' and 'Sign In'. A secondary navigation bar contains links for 'About', 'Downloads', 'Documentation', 'Community', 'Success Stories', 'News', and 'Events'. The 'Downloads' link is highlighted with a red rectangle. Below the navigation bar, there is a code snippet on the left and an article titled 'All the Flow You'd Expect' on the right. The code snippet shows a for loop that calculates the product of a list of numbers. The article text discusses Python's control flow statements. At the bottom, there is a footer with the text 'Python is a programming language that lets you work quickly and integrate systems more effectively. >>> [Learn More](#)'.

```
# For loop on a list
>>> numbers = [2, 4, 6, 8]
>>> product = 1
>>> for number in numbers:
...     product = product * number
...
>>> print('The product is:', product)
The product is: 384
```

**All the Flow You'd Expect**

Python knows the usual control flow statements that other languages speak — **if**, **for**, **while** and **range** — with some of its own twists, of course. [More control flow tools in Python 3](#)

1 2 3 4 5

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

# Windows, 설치 과정 (2/6)

31

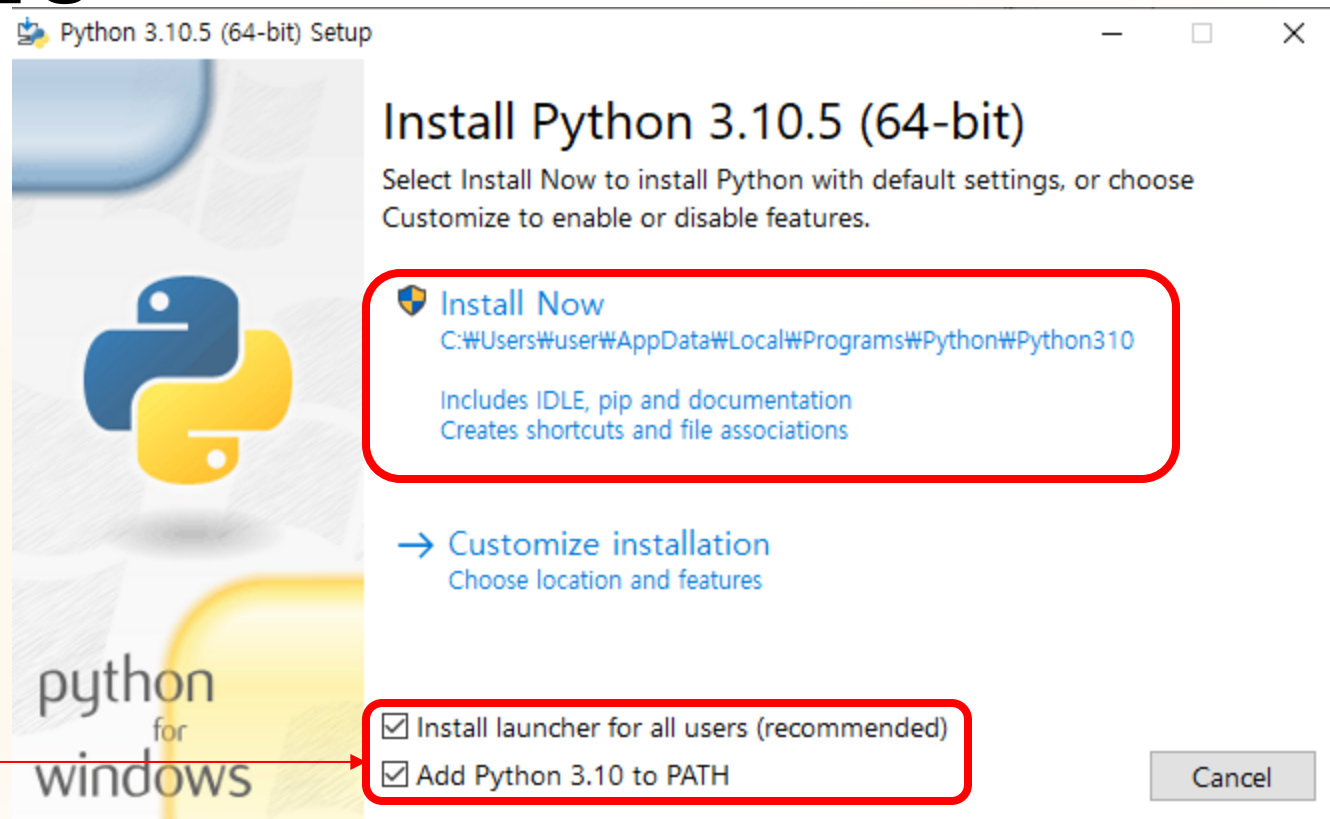
클릭하면 자동으로  
OS에 맞추어서 설치 됨



# Windows, 설치 과정 (3/6)

32

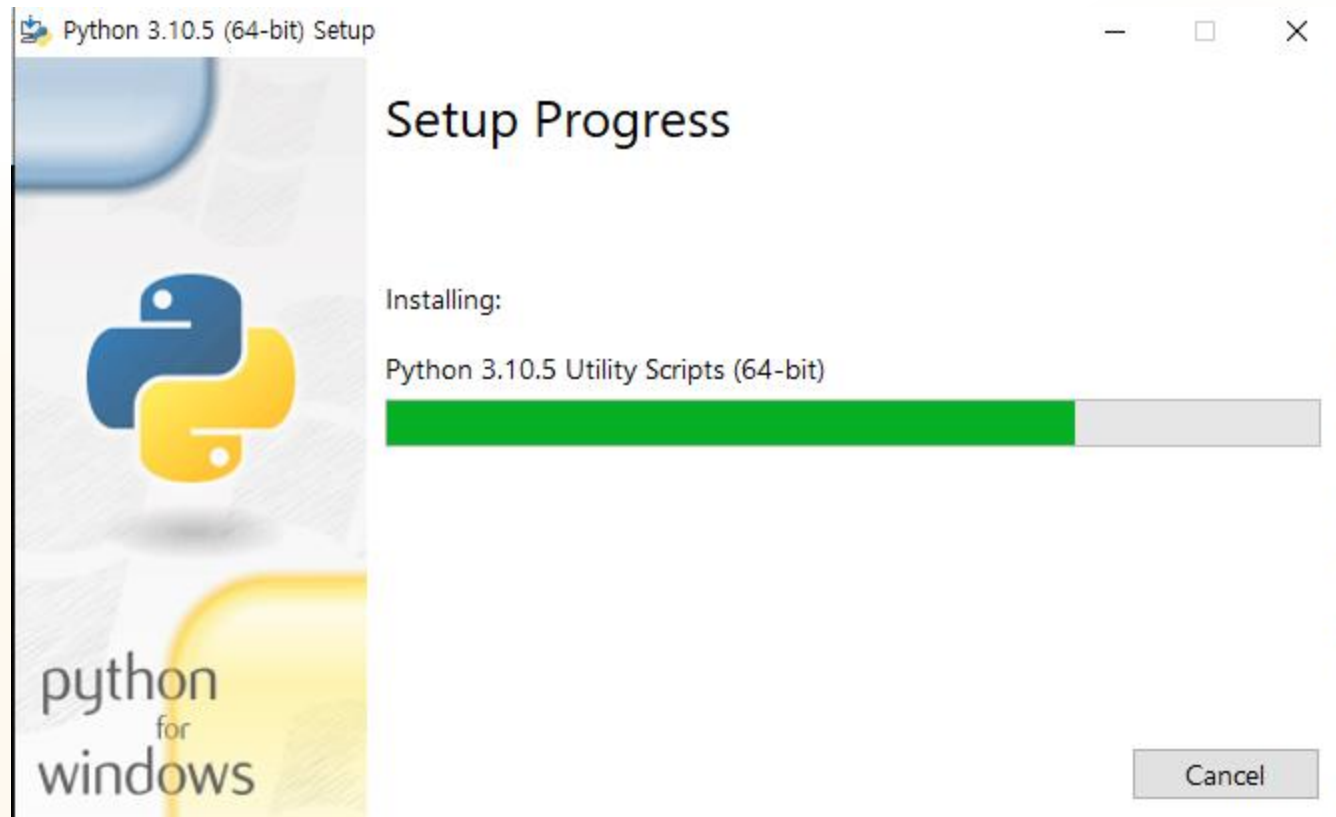
## 1. exe파일 실행





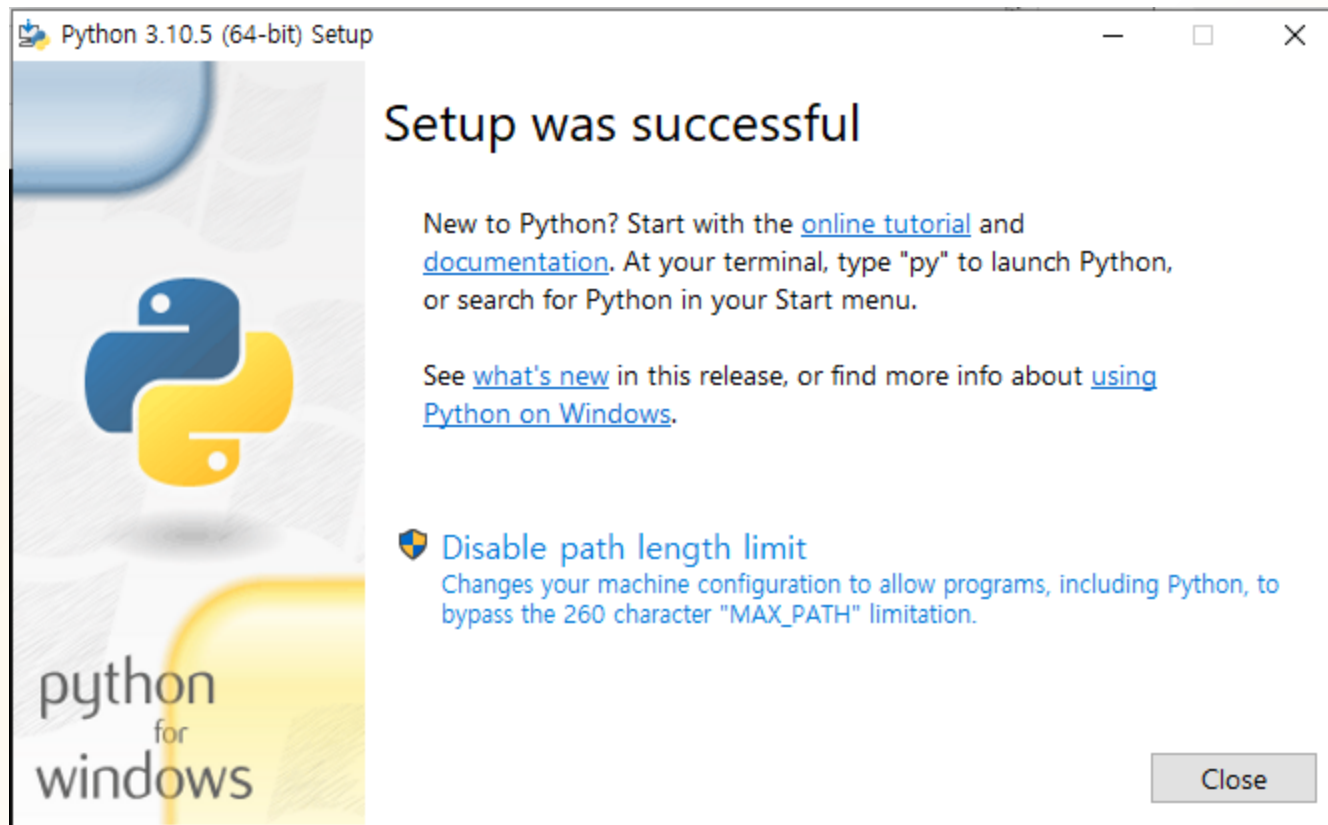
# Windows, 설치 과정 (4/6)

33



# Windows, 설치 과정 (5/6)

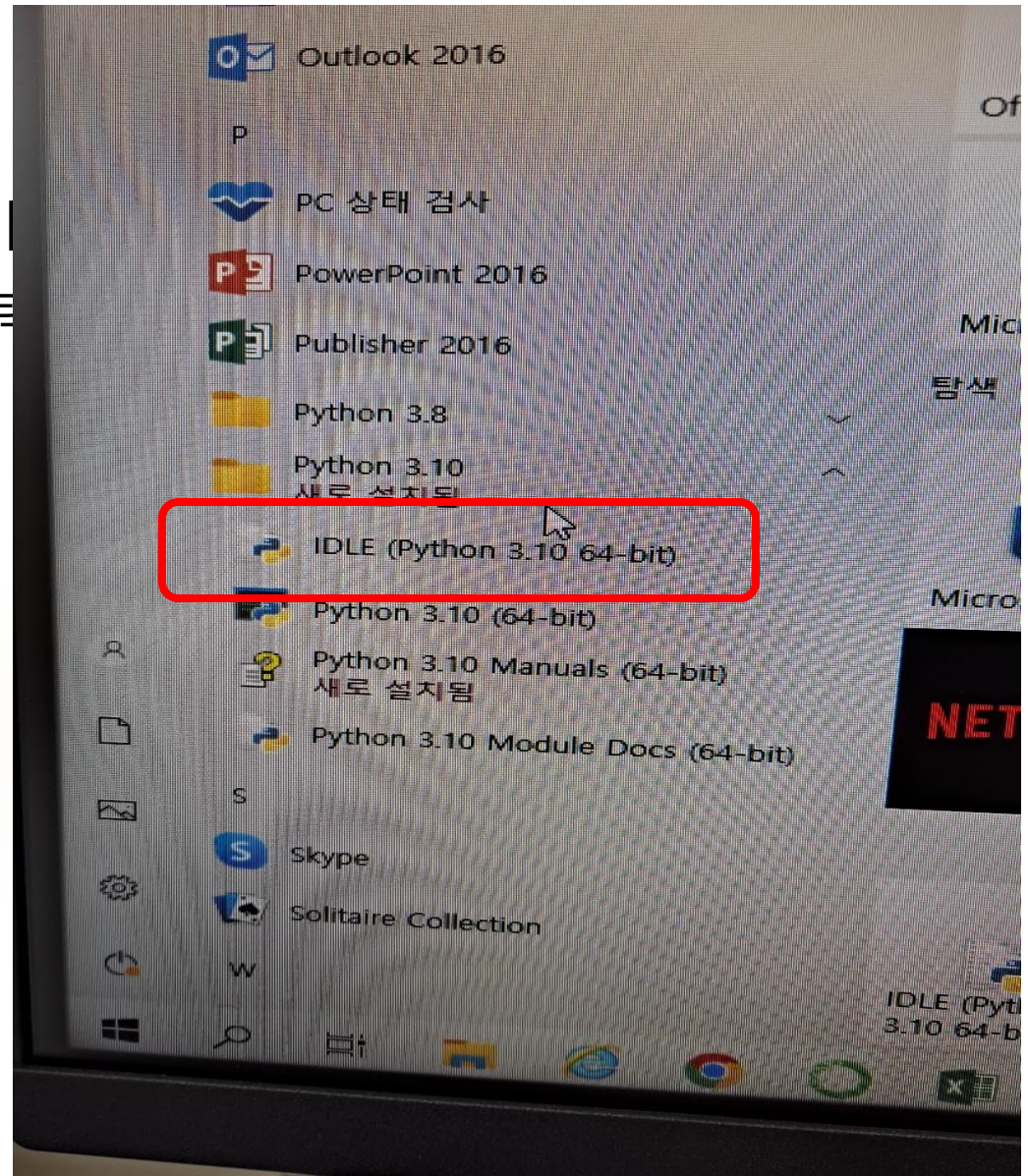
34



# Windows, 설치 과정 (6/6)

35

- ▶ 작업 표시줄에 고정하기
- ▶ 마우스 오른쪽 버튼 선택
- ▶ 작업표시줄 고정 선택



# Idle 실행하기

36

- ▶ 작업 표시줄에서 실행

A screenshot of the IDLE Shell 3.10.5 window. The window has a title bar that says "IDLE Shell 3.10.5" and standard window controls (minimize, maximize, close). Below the title bar is a menu bar with the following items: File, Edit, Shell, Debug, Options, Window, and Help. The main area of the window is a text editor with a white background. It contains the following text: "Python 3.10.5 (tags/v3.10.5:f377153, Jun 6 2022, 16:14:13) [MSC v.1929 64 bit (AMD64)] on win32" followed by "Type 'help', 'copyright', 'credits' or 'license()' for more information." Below this text is a red prompt ">>>" followed by a vertical cursor line.

# 강의 요약

37

- ▶ 파이썬 IDLE 설치
  - ▶ [python.org](https://python.org)

# 목표 달성 질문

38

- ▶ <http://www.python.org> 접속 후 윈도우용 최신 버전은 찾기 쉬웠는가?
- ▶ 설치 후 실행에 문제는 없는가?

# 학습목표

39

- ▶ 맥북용 파이썬 IDLE를 설치해보기



# Mac OS, 설치 과정 (1/6)

40

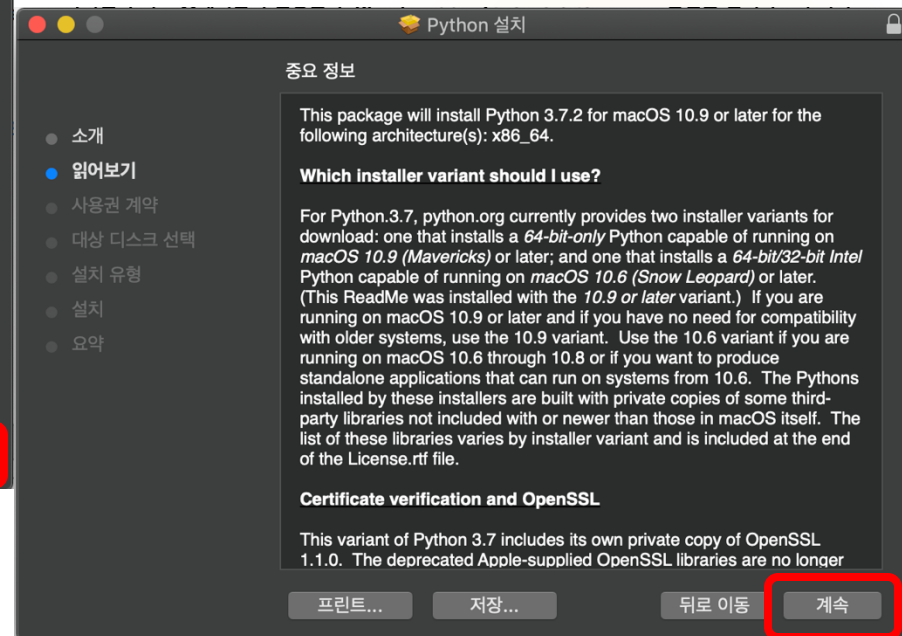
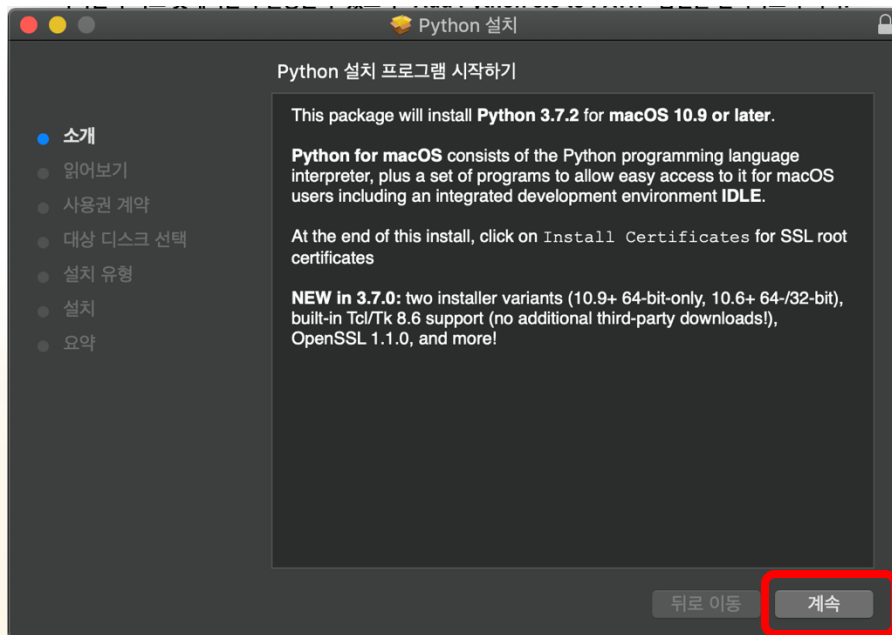
클릭하면 자동으로  
OS에 맞추어서 설치가  
됨





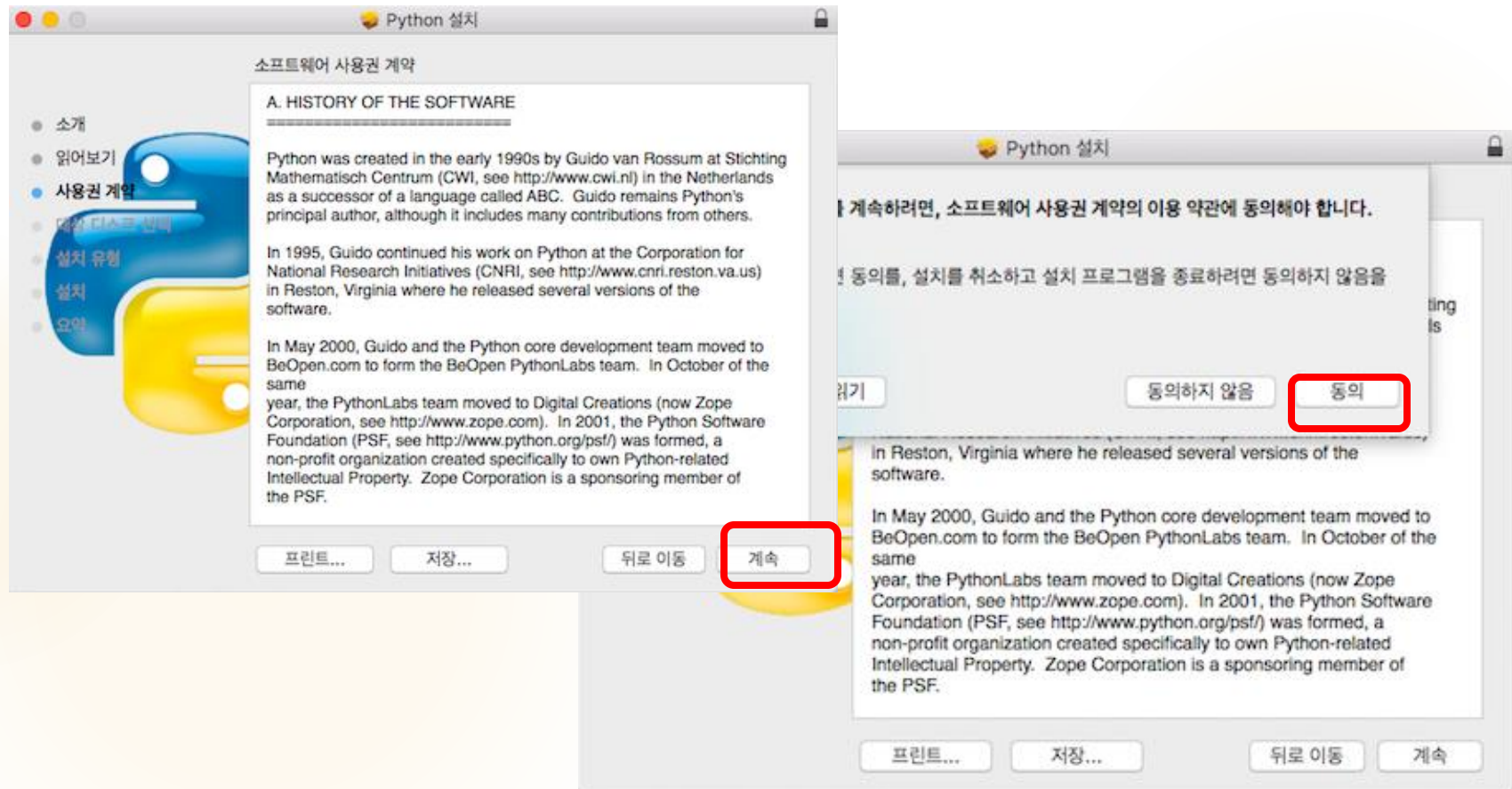
# Mac OS, 설치 과정 (2/6)

41



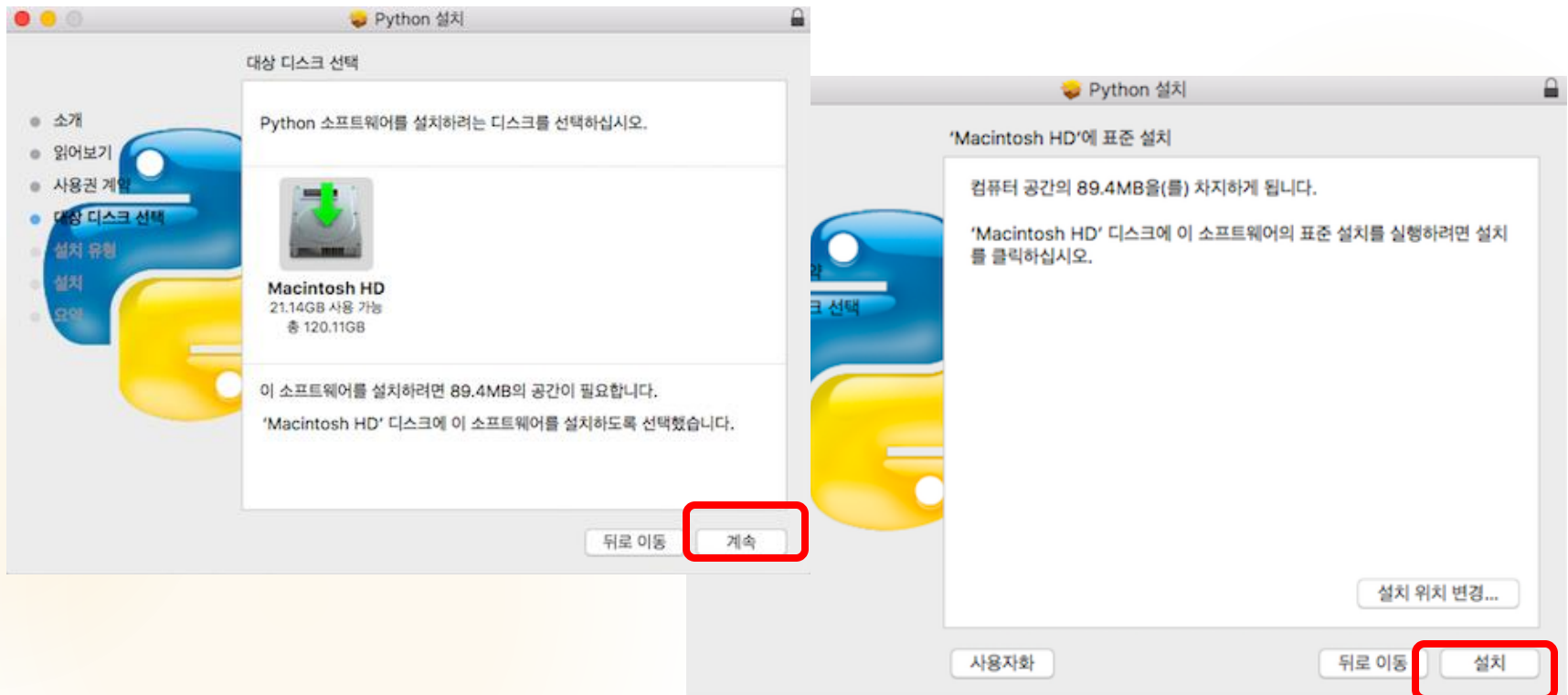
# Mac OS, 설치 과정 (3/6)

42



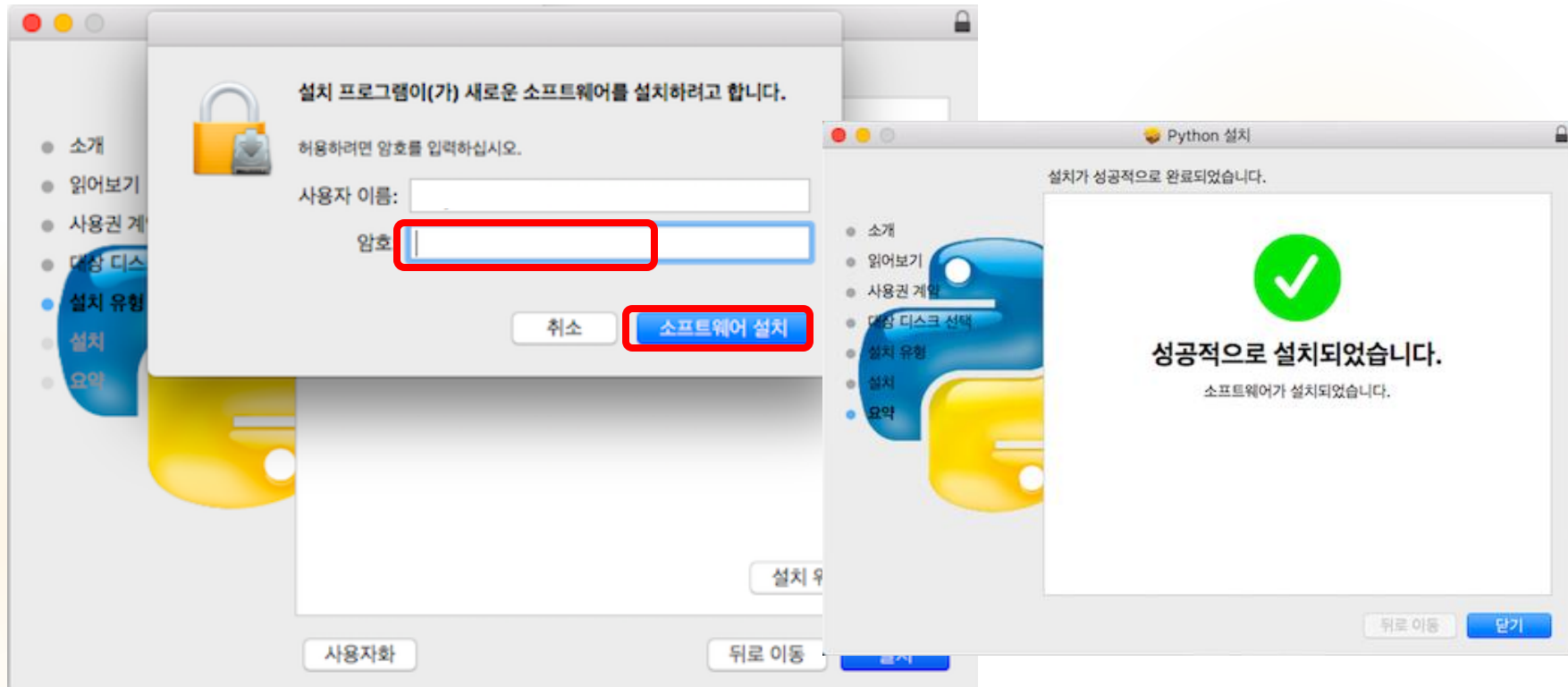
# Mac OS, 설치 과정 (4/6)

43



# Mac OS, 설치 과정 (5/6)

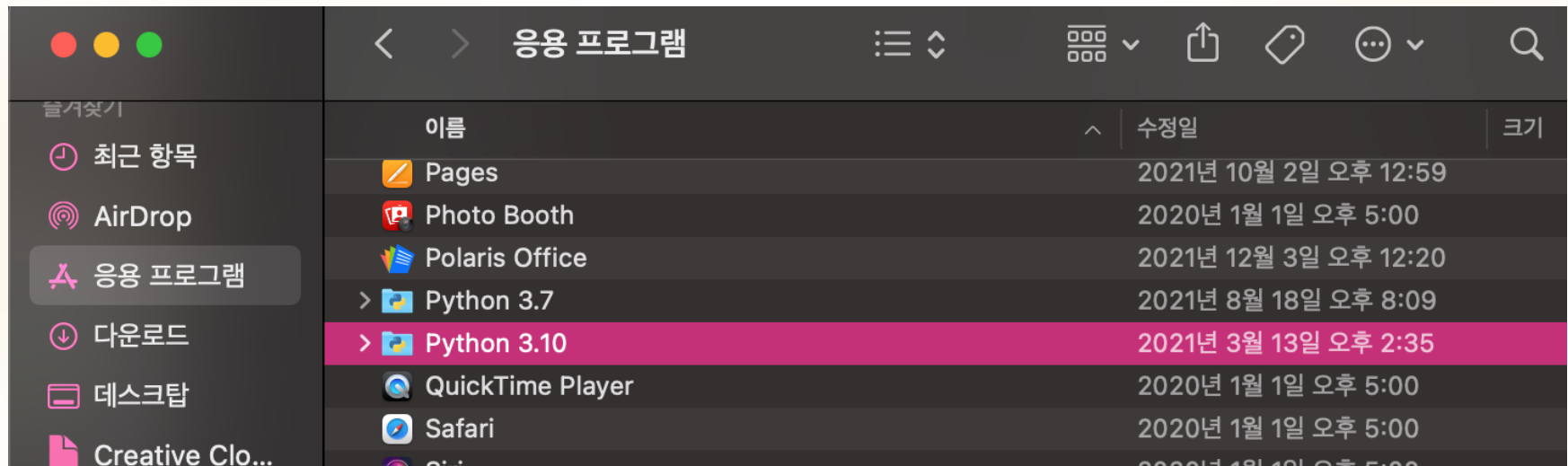
44



# Mac OS, 설치 과정 (6/6)

45

- 실행 경로: Finder – 응용프로그램 – Python 3.10 – IDLE



# 강의 요약

46

- ▶ 파이썬 IDLE 설치
  - ▶ Python.org

# 목표 달성 질문

47

- ▶ <http://www.python.org> 접속 후 맥용 최신 버전은 찾기 쉬웠는가?
- ▶ 설치 후 실행에 문제는 없는가?

# idle 사용해 보기

1주차\_02\_03



# 학습목표

49

- ▶ idle 설치 후 사용해보기
- ▶ 상호작용 방식으로 사용하기
- ▶ 스크립트 모드로 사용하기

# Idle 실행하기

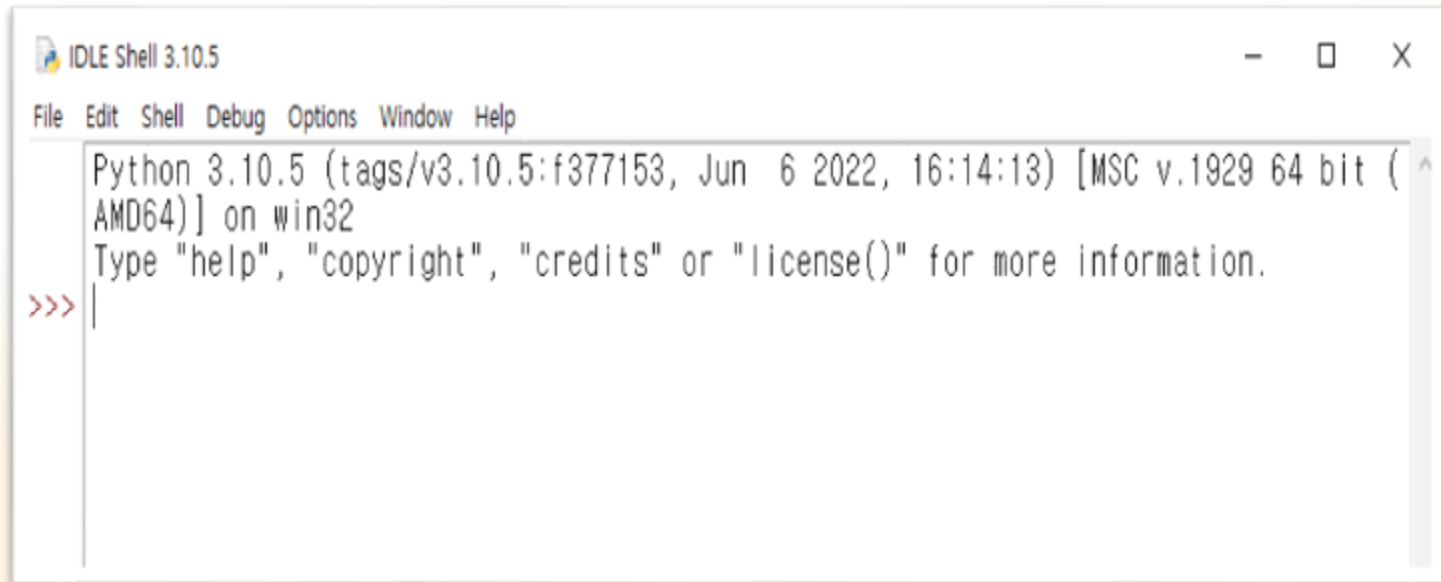
50

- ▶ 작업 표시줄에서 실행



# 실행해 보기, IDLE

51

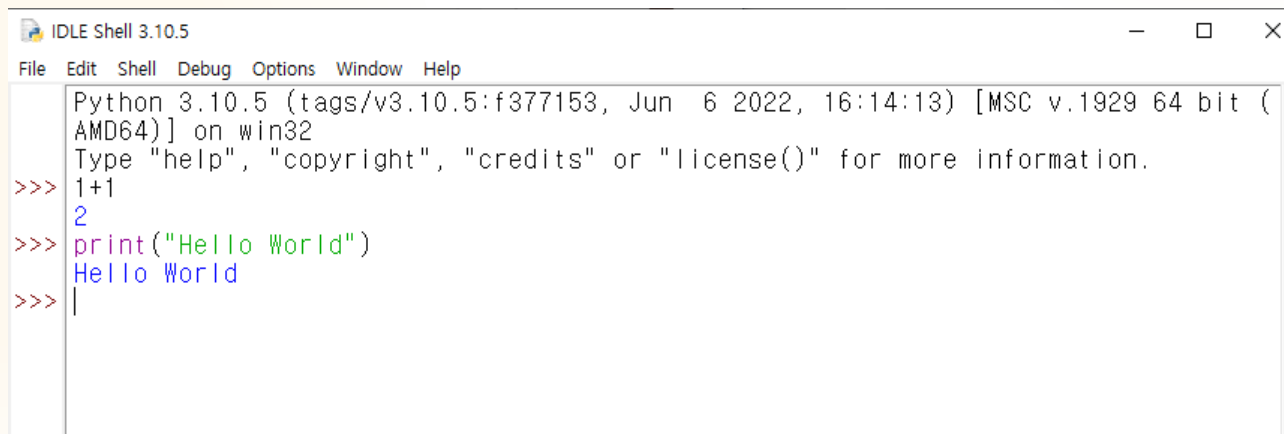


```
Python 3.10.5 (tags/v3.10.5:f377153, Jun 6 2022, 16:14:13) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> |
```

- ▶ Integrated DeveLopment Environment for Python
- ▶ the Python IDE built with the Tkinter GUI toolkit
  - ▶ 파이썬으로 만들어졌으며, Tkinter GUI 지원
  - ▶ 윈도우와 유닉스에서 사용 가능
  - ▶ 여러 개의 작업 가능하고, 입력 취소 기능 제공
  - ▶ 간단한 디버깅 기능 제공

# IDLE실행 – 상호작용 방식

- ▶ 파이썬 인터프리터 사용
  - ▶ interactive mode
    - ▶ `>>> 1 + 1`
    - ▶ `2`
    - ▶ `>>> print("Hello World!")`
    - ▶ `Hello World!`

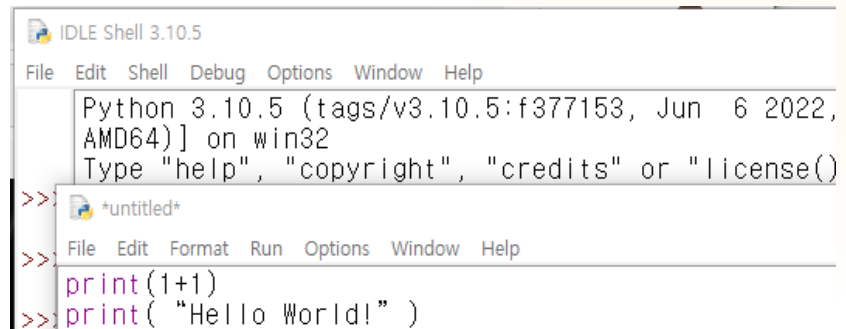


```
IDLE Shell 3.10.5
File Edit Shell Debug Options Window Help
Python 3.10.5 (tags/v3.10.5:f377153, Jun 6 2022, 16:14:13) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 1+1
2
>>> print("Hello World")
Hello World
>>> |
```

# IDLE실행 – 스크립트 방식(1/3)

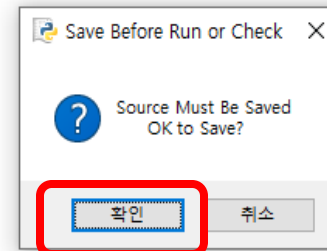
54

- ▶ 파이썬 스크립트는 .py로 끝나는 파일로 저장
- ▶ “File”을 클릭 후 – “New file” 클릭
- ▶ 키보드로 입력
  - ▶ `print(1+1)`
  - ▶ `print("Hello World!")`
- ▶ 저장 후 실행(F5)



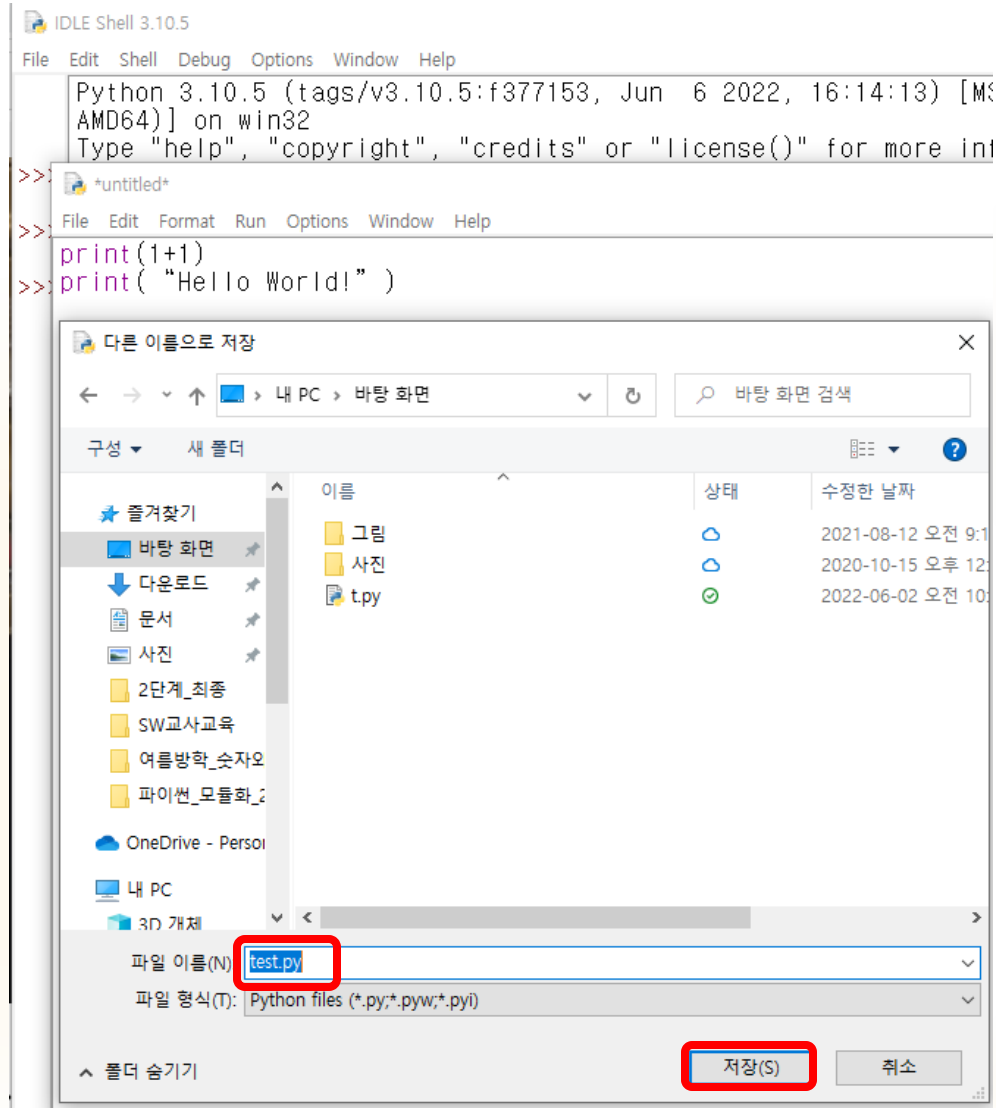
The screenshot shows the IDLE Shell 3.10.5 window. The top menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The main text area displays the Python 3.10.5 version information and a prompt. Below this, a new file editor window titled '\*untitled\*' is open, showing the same menu bar (File, Edit, Format, Run, Options, Window, Help) and the Python prompt. The code entered in the editor is:

```
>>> print(1+1)
>>> print("Hello World!")
```



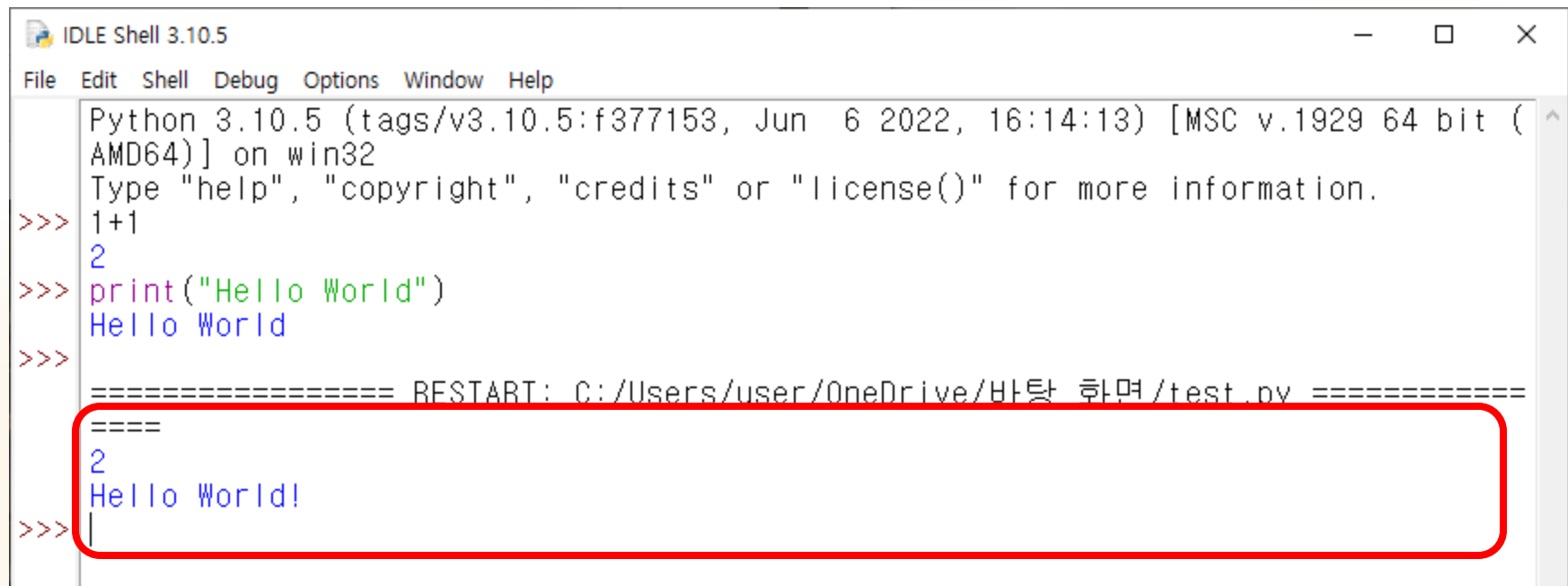
# IDLE실행 - 스크립트 방식(2/3)

55



# IDLE실행 - 스크립트 방식(3/3)

56



```
IDLE Shell 3.10.5
File Edit Shell Debug Options Window Help
Python 3.10.5 (tags/v3.10.5:f377153, Jun 6 2022, 16:14:13) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 1+1
2
>>> print("Hello World")
Hello World
>>>
===== RESTART: C:/Users/user/OneDrive/바탕 화면/test.py =====
>>>
2
Hello World!
>>> |
```



# 사용 예제 (Text)

57

```
milesDriven = input("Enter miles driven:")  
milesDriven = float(milesDriven)  
  
gallonsUsed = input("Enter gallons used:")  
gallonsUsed = float(gallonsUsed)  
  
mpg = milesDriven / gallonsUsed  
print("Miles per gallon:",mpg)
```

# 사용 예제 (move)

58

```
from tkinter import *
tk = Tk()

canvas = Canvas(tk, width=400, height=400)
canvas.pack()

my_t = canvas.create_polygon(10,10,10,60,50,35,fill='pink',
outline='black')

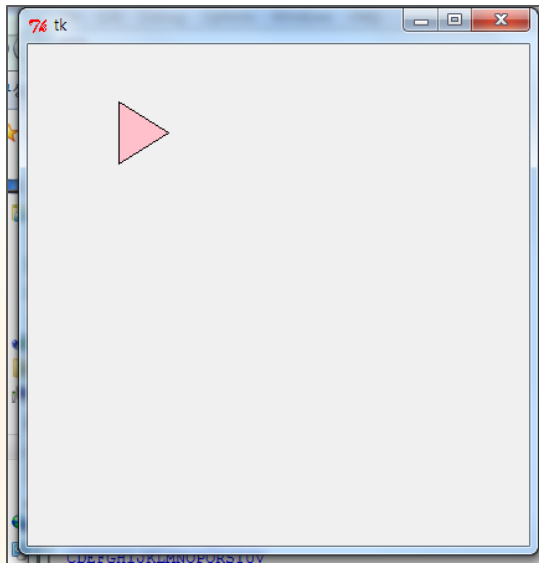
def movetriangle(event):
    if event.keysym=='Up':
        canvas.move(my_t,0,-3)
        canvas.itemconfig(my_t, fill='pink')
    elif event.keysym=='Down':
        canvas.move(my_t,0,3)
        canvas.itemconfig(my_t, fill='grey')
```

```
elif event.keysym=='Left':
    canvas.move(my_t,-3, 0)
    canvas.itemconfig(my_t, fill='lightblue')
else :
    canvas.move(my_t,3,0)
    canvas.itemconfig(my_t, fill='white')

canvas.bind_all('<KeyPress-Up>', movetriangle)
canvas.bind_all('<KeyPress-Down>', movetriangle)
canvas.bind_all('<KeyPress-Left>', movetriangle)
canvas.bind_all('<KeyPress-Right>', movetriangle)
```

# 사용 예제 (move)

59



# 사용 예제 (draw)

60

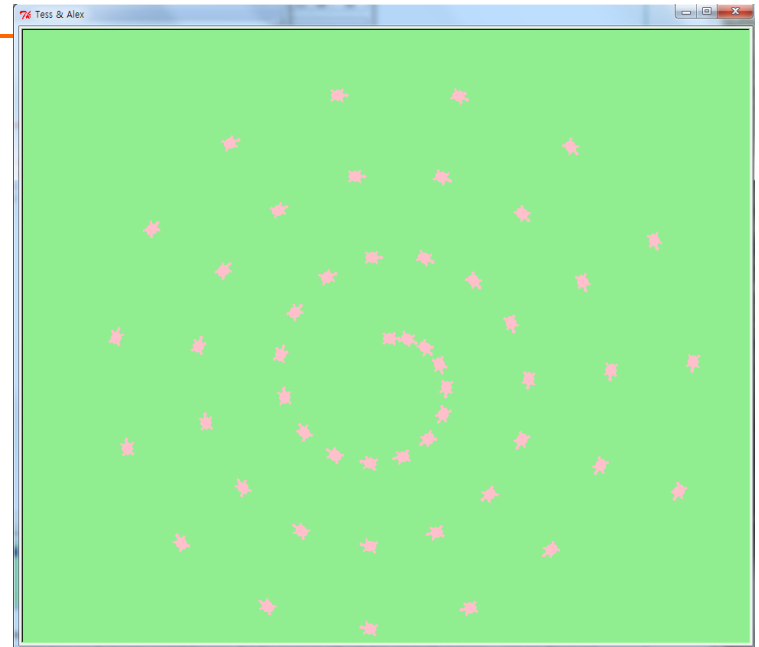
```
import turtle
```

```
wn = turtle.Screen()  
wn.bgcolor("light green")
```

```
t = turtle.Turtle()  
t.shape("turtle")  
t.color("pink")
```

```
t.penup()                # This is new  
size = 20
```

```
for i in range(50):  
    t.stamp()             # Leave an impression on the canvas  
    size = size + 3       # Increase the size on every iteration  
    t.forward(size)       # Move tess along  
    t.right(24)           # ... and turn her
```



# 사용 예제 (web)

61

```
from tkinter import *

def NewFile() :
    print("New File!")
def OpenFile() :
    print("Open File!")
def About() :
    print("This is a simple example of a menu")

root = Tk()
menu = Menu(root)
root.config(menu=menu)
filemenu = Menu(menu)
menu.add_cascade(label="File", menu=filemenu)
filemenu.add_command(label="New", command=NewFile)
filemenu.add_command(label="Open...", command=OpenFile)
filemenu.add_separator()
filemenu.add_command(label="Exit", command=root.quit)
helpmenu = Menu(menu)
menu.add_cascade(label="Help", menu=helpmenu)
helpmenu.add_command(label="About...", command=About)

mainloop()
```

# 사용 예제 (file Input Output)

62

```
inf = open('poem.txt', 'r')
NumWord = []

for i in range(9):
    fline = inf.readline()
    flist = fline.split()
    NumWord.append(len(flist))

print(NumWord)

inf.close()
```

# 강의 요약

63

- ▶ IDLE 처음으로 사용해 보기
  - ▶ 상호작용 모드
  - ▶ 스크립트 모드

# 목표 달성 질문

- ▶ IDLE 상호작용 방식과 스크립트 모드의 차이점을 설명하시오
- ▶ .py 파일을 클릭하면 IDLE와 바로 연결되지 않는다. 어떻게 하여야 연결이 되는지 말해보세요



# 간단한 코드 따라 해 보기

1주차\_03\_01

한 동 대 학 교  
김경미 교수

# 학습목표

66

- ▶ 간단한 코딩 따라 해 보기
- ▶ 상호작용 방식과 스크립트 방식의 코딩 차이를 이해하기
- ▶ print문 활용하기

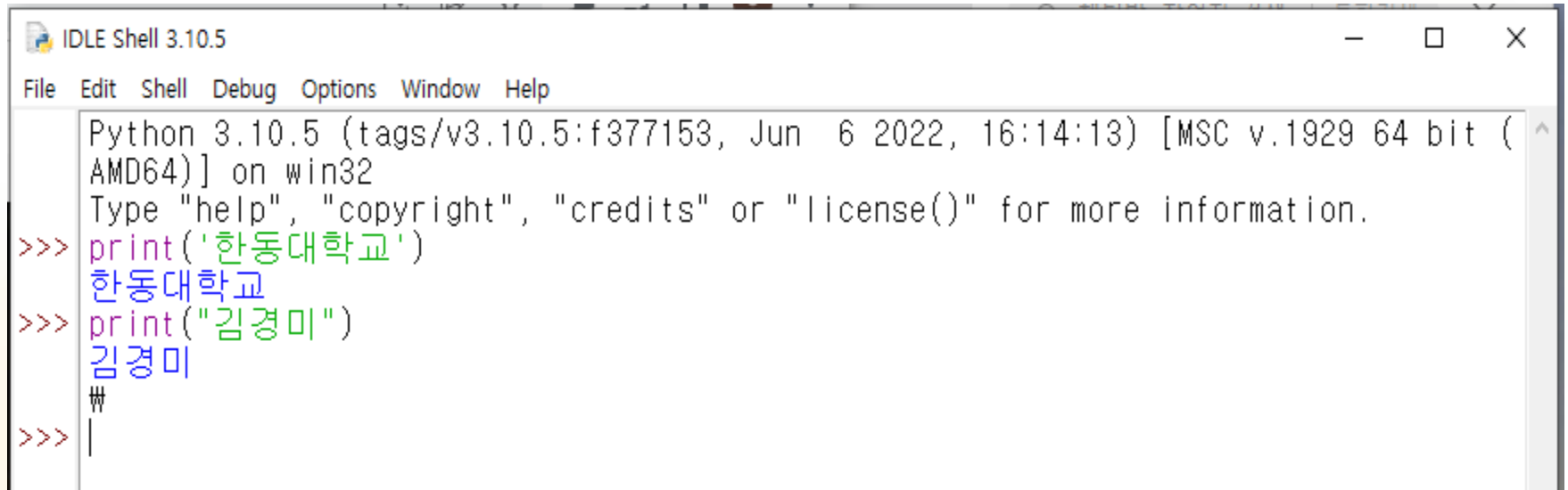
# 연습문제 1

67

- ▶ 상호작용 방식
- ▶ 자신의 소속과 이름을 화면에 출력한다
- ▶ print()를 사용한다

# 연습문제 1, 코드

68



```
Python 3.10.5 (tags/v3.10.5:f377153, Jun 6 2022, 16:14:13) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print('한동대학교')
한동대학교
>>> print("김경미")
김경미
>>> 
```

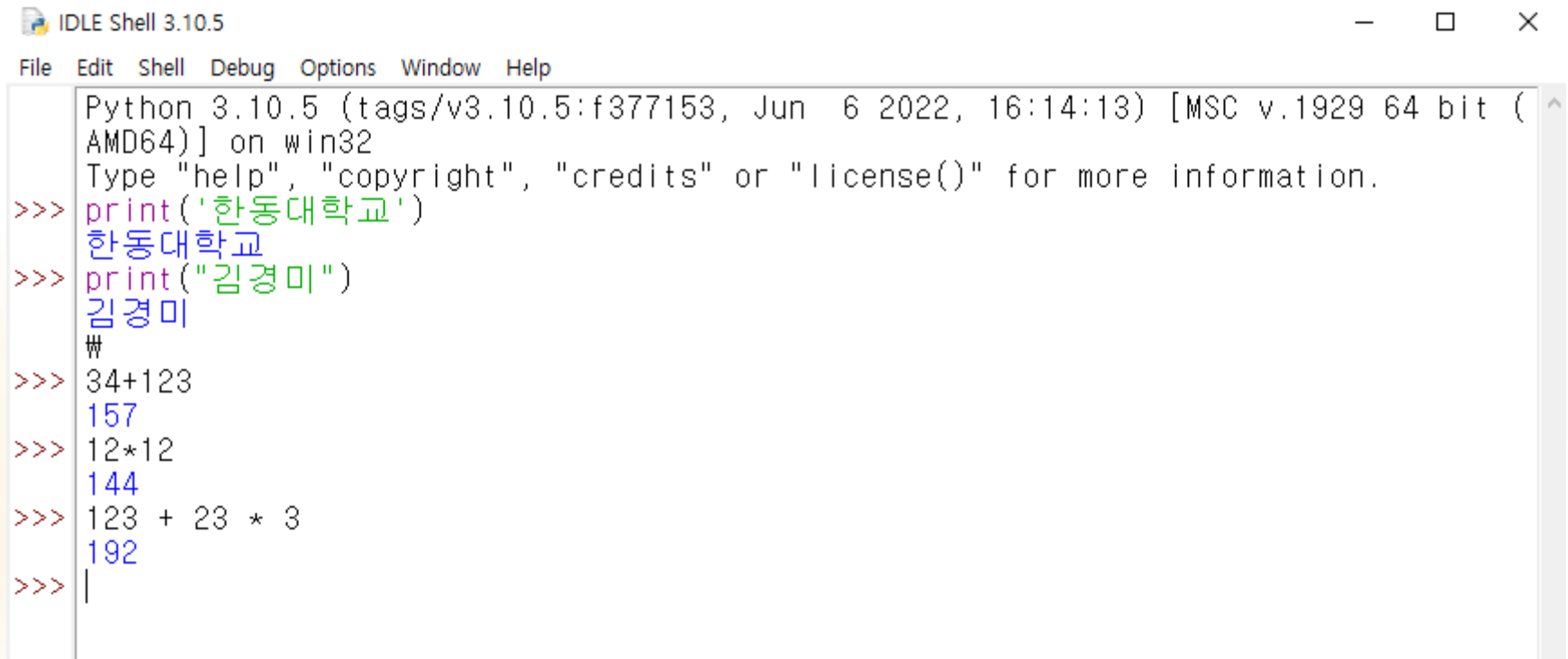
# 연습문제 2

69

- ▶ 상호작용 방식
- ▶ 더하기연산과 곱하기 연산을 시도해 본다
- ▶ 더하기는 '+', 곱하기는 '\*' 기호를 사용한다
- ▶ 상호작용 방식에서  
수식은 `print()`를 사용하지 않아도 결과가 나타난다

# 연습문제 2, 코드

70



```
Python 3.10.5 (tags/v3.10.5:f377153, Jun 6 2022, 16:14:13) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print('한동대학교')
한동대학교
>>> print("김경미")
김경미
>>> 34+123
157
>>> 12*12
144
>>> 123 + 23 * 3
192
>>> |
```

# 연습문제 3

71

- ▶ 스크립트 방식
- ▶ 메뉴 file-New file를 선택한 후,
  - ▶ 연습문제 01, 02 내용을 입력하여
  - ▶ 화일이름 'firstcode.py'로 저장한다
  - ▶ 실행해 본다(F5 key 누르면 실행)
- ▶ 상호작용 방식과 스크립트 방식의 차이가 무엇인가요?

# 연습문제 3, 코드와 결과

72

```
firstcode.py - C:/1_Works/2017Work/KMooc/Exercise Code/firstcode..
File Edit Format Run Options Window Help
print("한동대학교")
print("Kim Kyungmi")
34+123
12*12
123 + 23 * 2
```

```
===== RESTART: C:/1_Works/2017Work/KMooc/Exercise Code/firstcode.py
한동대학교
Kim Kyungmi
>>>
```

```
firstcode.py - C:/1_Works/2017Work/KMooc/Exercise Code/firstcode..
File Edit Format Run Options Window Help
print("한동대학교")
print("Kim Kyungmi")
print(34+123)
print(12*12)
print(123 + 23 * 2)
```

```
===== RESTART: C:/1_Works/2017Work/KMooc/Exercise Code/firstcode.py
한동대학교
Kim Kyungmi
157
144
169
>>> |
```



# 연습문제 4

73

- ▶ 스크립트 방식
- ▶ 다음 코드를 그대로 입력한 후, 실행 해 본다

```
## filename: iftest.py
```

```
age = 25
```

```
if age > 65 :  
    print("senior")  
elif age > 20 :  
    print("adult")  
else :  
    print("young")
```

## 연습문제 4, 코드와 결과

74

```
age = 25
if age > 65 :
    print("senior")
elif age > 20 :
    print("adult")
else :
    print("young")
```

```
===== RESTART: C:\#7_VariousLang\0_python\my_code\conditional.py
adult
>>> |
```

# 연습문제 5

75

- ▶ 연습문제 4에서 age를 코드에서 설정했다면
  - ▶ 이번에는 age를 입력 받는다

```
## filename: iftest2.py
age = int(input("나이를 입력하세요: "))

if age > 65 :
    print("senior")
elif age > 20 :
    print("adult")
else :
    print("young")
```

# 연습문제 5, 코드와 결과

76

```
age = int(input("나이를 입력하세요: "))
```

```
if age > 65 :  
    print("senior")  
elif age > 20 :  
    print("adult")  
else :  
    print("young")  
|
```

```
>>>
```

```
나이를 입력하세요: 17
```

```
young
```

```
>>> =====
```

```
>>>
```

```
나이를 입력하세요: 24
```

```
adult
```

```
>>> |
```

# 연습문제 6

77

- ▶ 스크립트 방식
- ▶ 다음 코드를 그대로 입력한 후, 실행 해 본다

```
## filename: whilettest.py
```

```
step=0
```

```
while step < 1000 :
```

```
    print(step)
```

```
    step += 100
```

# 연습문제 6, 코드와 결과

78

```
loop.py - C:\W7_VariousLang\W0_pyth...
File Edit Format Run Options Window

step=0

while step < 1000 :
    print(step)
    step += 100
```

```
===== RESTART:
0
100
200
300
400
500
600
700
800
900
>>> |
```

# 연습문제 7

- ▶ 구구단 2단을 출력한다
- ▶ 다음 코드를 그대로 입력한 후, 따라해본다
- ▶ num의 숫자를 변경하여 단 수를 변경할 수 있다

```
#fortest.py
```

```
num = 2
```

```
for i in range (1, 10) :  
    print(num, "X", i, "=", num*i)
```

# 연습문제 7 코드와 결과

```
#fortest.py

num = 2

for i in range (1, 10) :
    print(num, "X", i, "=", num*i)
```

```
>>>
2 X 1 = 2
2 X 2 = 4
2 X 3 = 6
2 X 4 = 8
2 X 5 = 10
2 X 6 = 12
2 X 7 = 14
2 X 8 = 16
2 X 9 = 18
>>>
```



- ▶ 간단한 코딩하기 연습
- ▶ 스크립트 방식에서
  - ▶ 수식 연산 결과를 나타내려면 print를 꼭 사용해야 함
  - ▶ 저장한 파일을 실행하면, 결과는 처음 시작한 IDLE 창에 나타남
- ▶ IDLE창에서 코드를 입력하면 색상이 바뀌는 것
  - ▶ print
  - ▶ if
  - ▶ elif
  - ▶ else
  - ▶ while

# 목표 달성 질문

82

- ▶ Idle, jupyter notebook, repl 중 본인이 편한 것을 결정 하셨나요?
- ▶ IDLE에서 상호작용 모드와 스크립트 모드의 차이점을 설명하시오

# 감사합니다

1주차\_03\_01\_ 간단한 코드 따라 해 보기