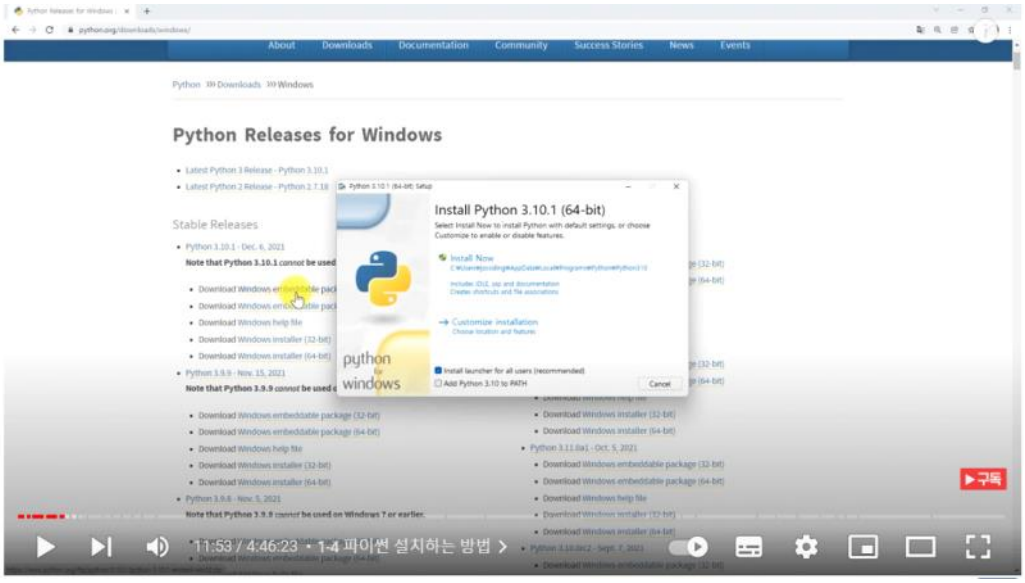
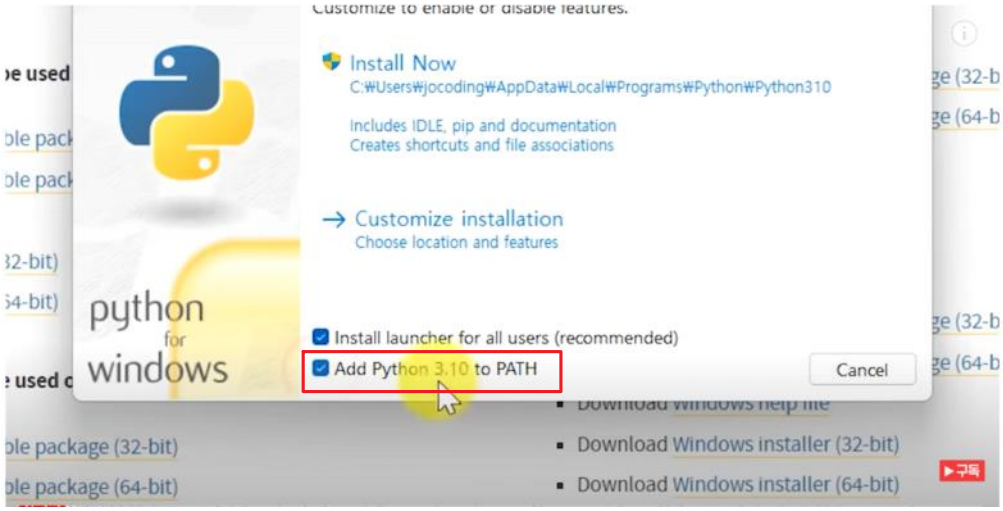
파이썬 설치

2022년 10월 14일 금요일 오후 1:51

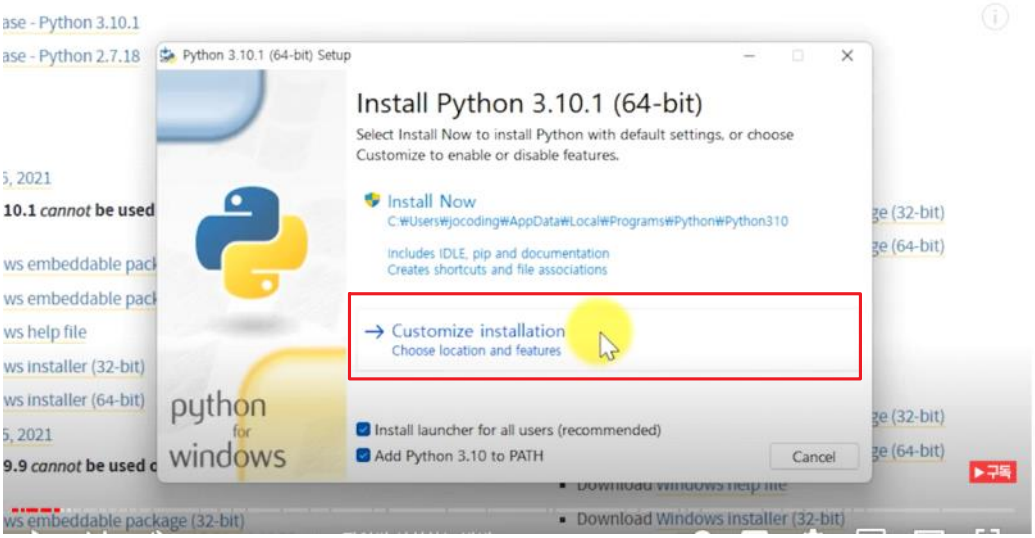
1.python.org에서 자신의 컴퓨터 사양을 확인하고 프로그램을 다운로드 한다.



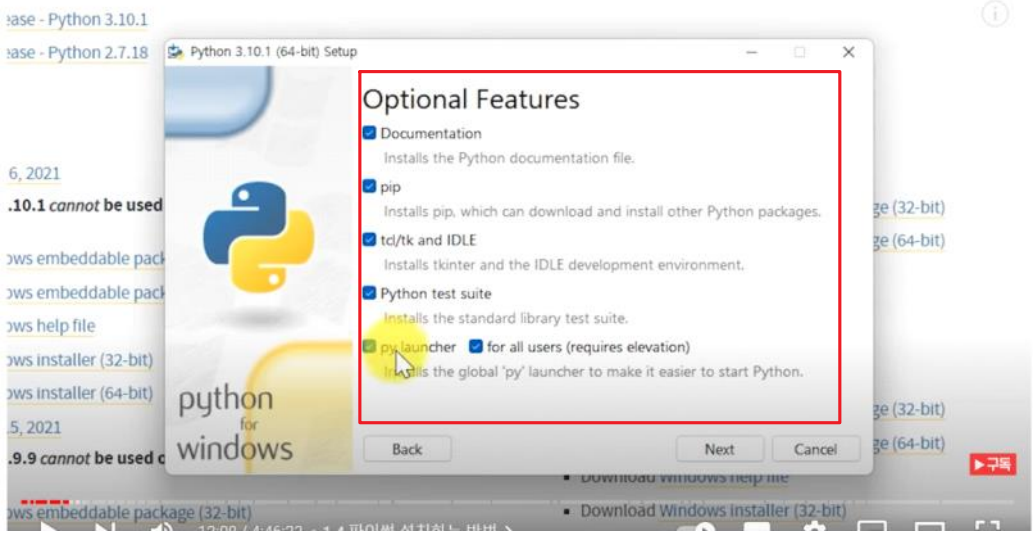
2.install 화면에서 Add python 버전명 to PATH 항목을 체크한다.



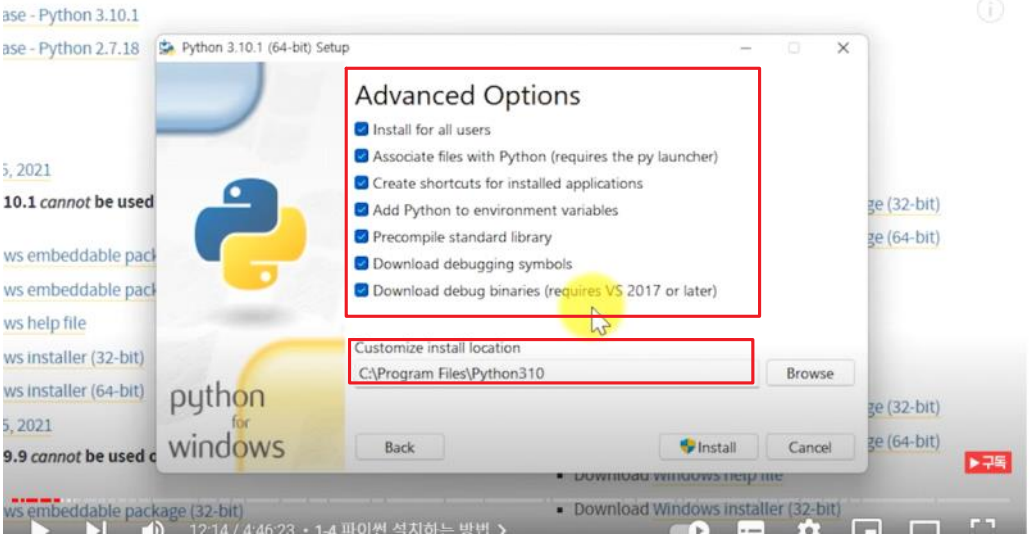
3.->Customize installation 부분을 클릭 한다.



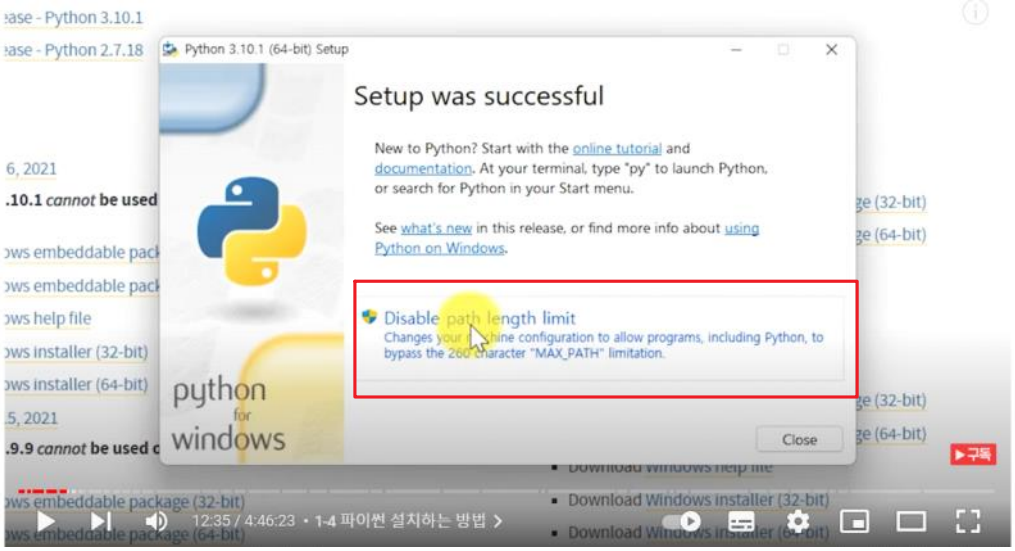
4.optional features 모든 항목을 체크한다.



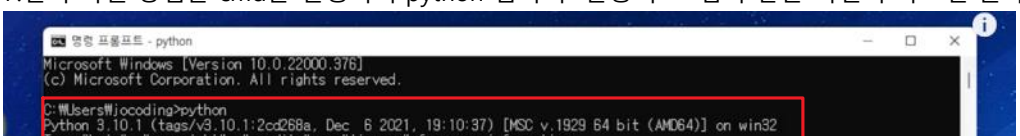
5.advanced options 모든 항목을 체크하고 customize install location위치는 사진과 같은 경로로 설정한다.

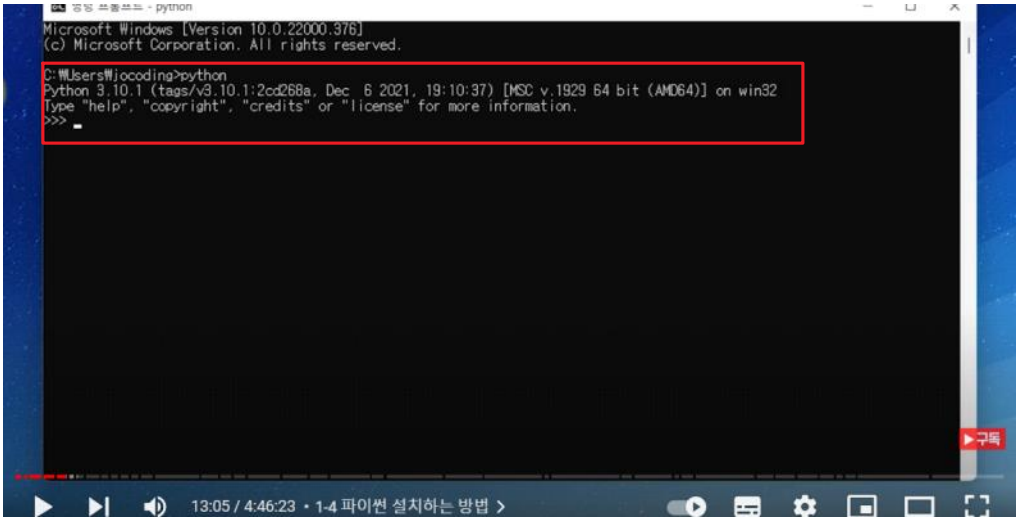


6.파이썬이 설치된 경로가 특정 글자를 넘어가게 되면 오류가 발생하는데 이 제한을 disable path length limit 클릭하여 해제한다.

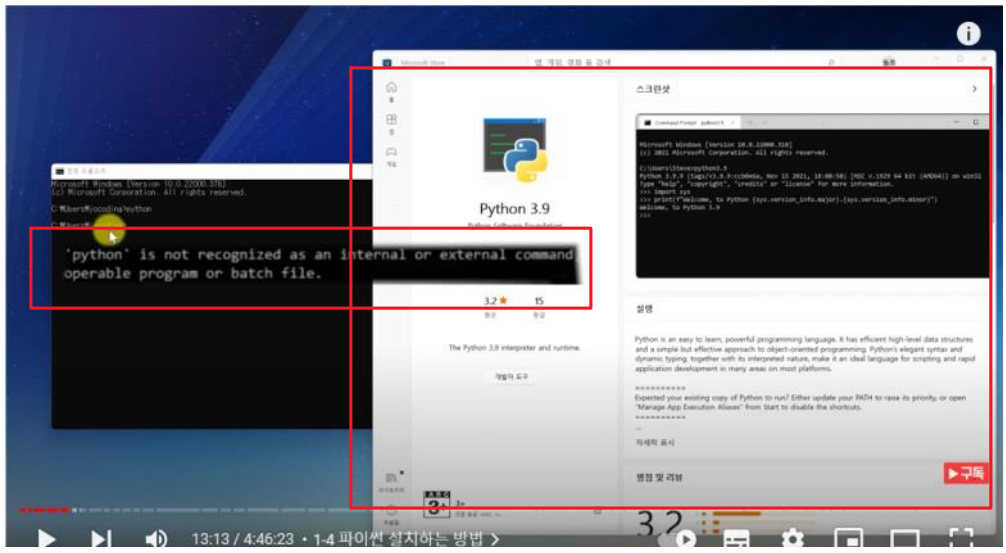


7.설치 확인 방법은 cmd를 실행시켜 python 입력 후 실행 후 그림과 같은 화면이 나오면 된다.

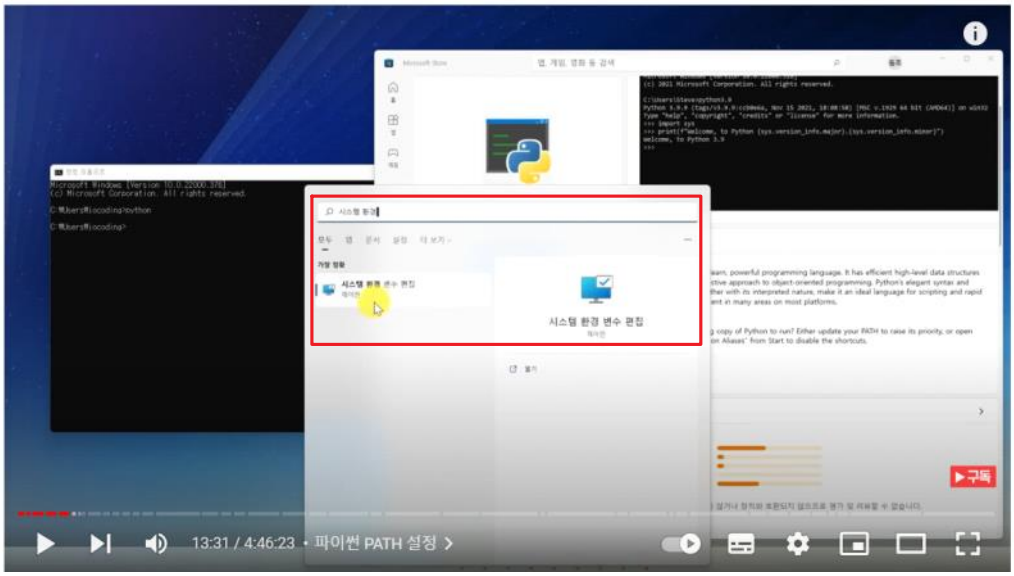




***7번 오류 발생 : python path설정의 문제가 발생하여 오류 메시지가 나오거나 마이크로소프트 스토어가 실행 될 경우**

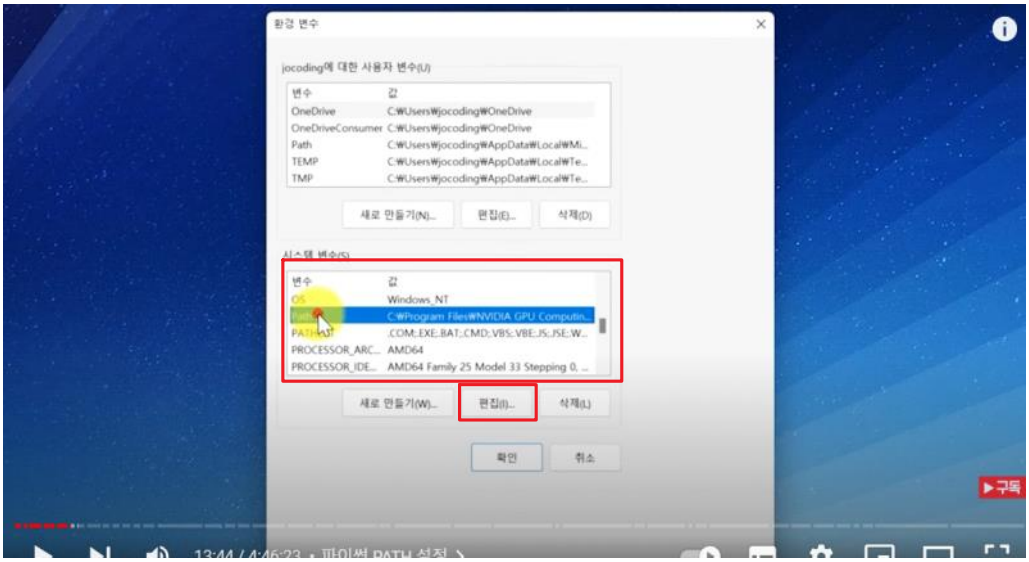


1.시작 -> 시스템 환경 변수 편집 클릭

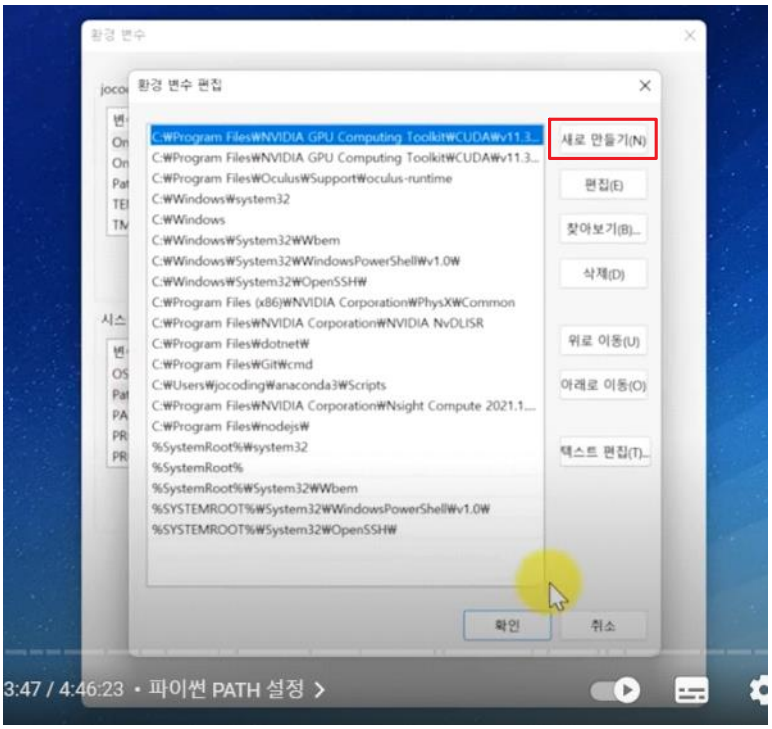


2.시스템 속성 -> 고급 -> 환경 변수(N)

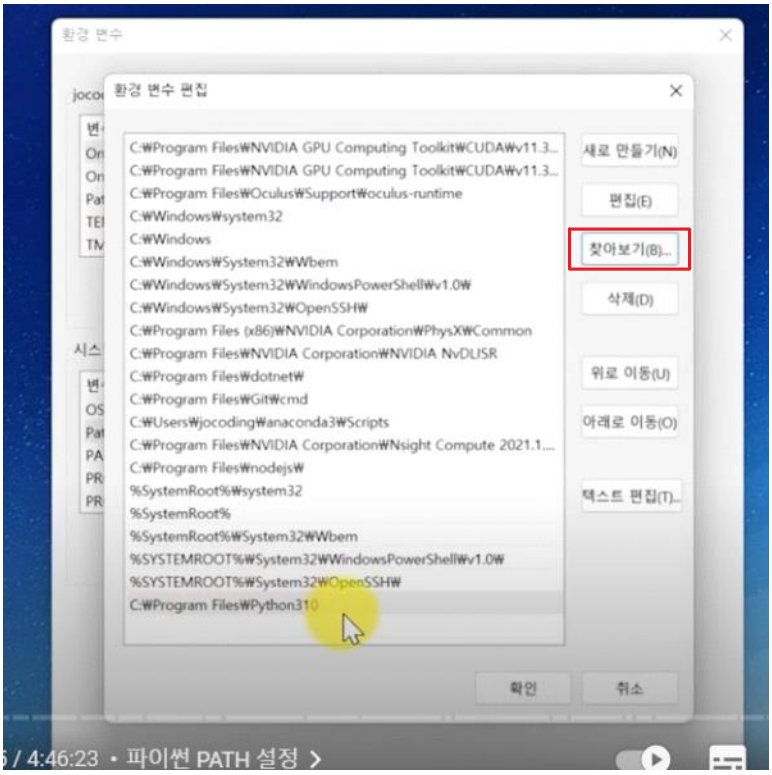
3.시스템 변수(S)부분에서 Path 더블 클릭이나 편집(I) 선택



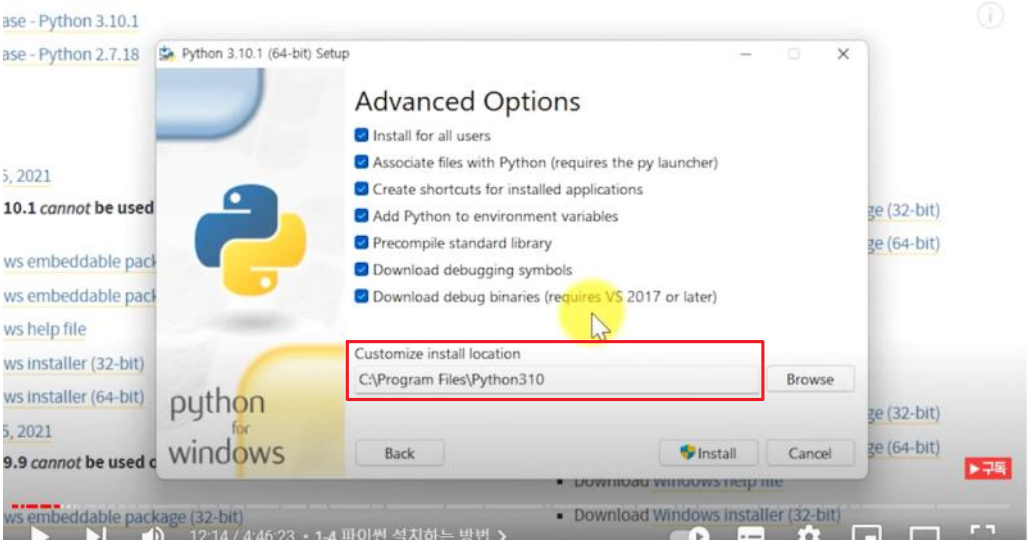
4.새로 만들기(N) 선택



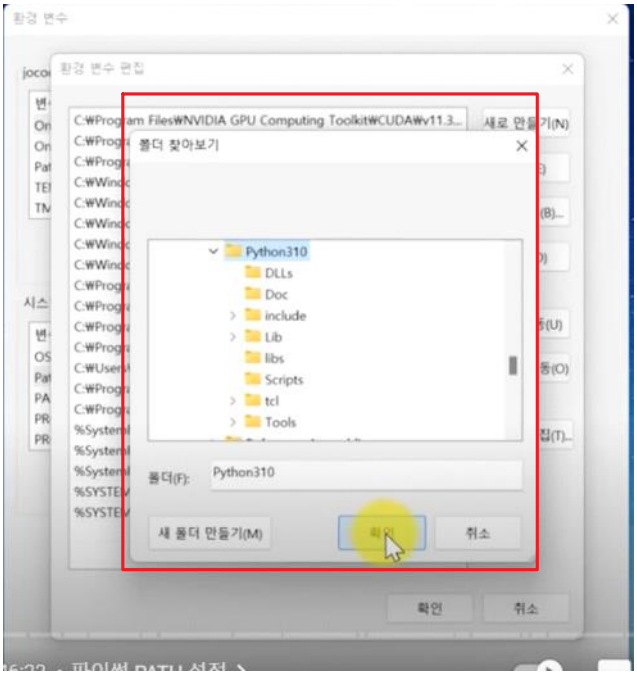
5.찾아보기(B) 선택



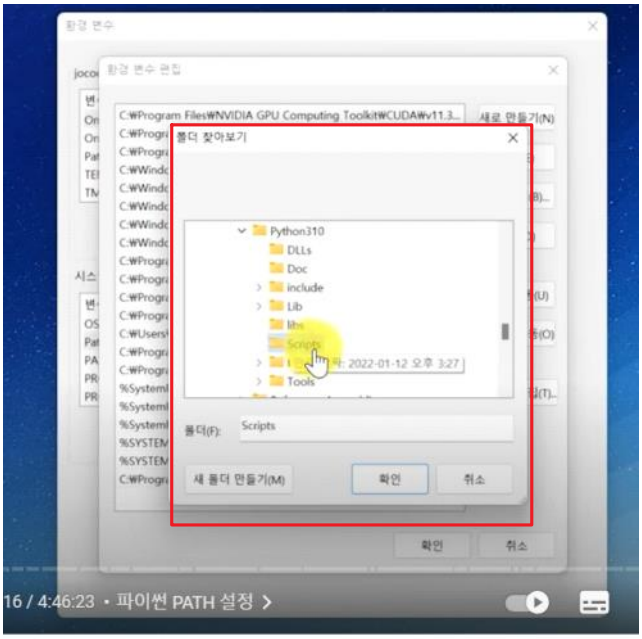
6.customize install location경로를 찾아서



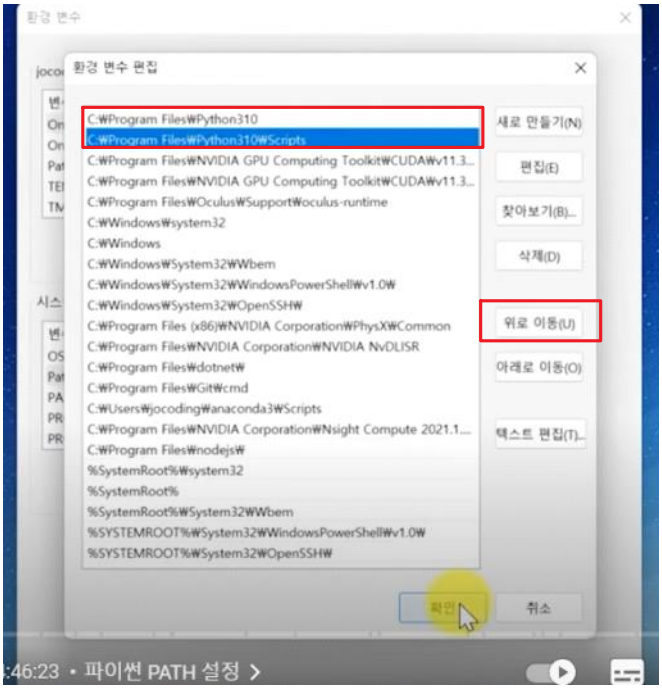
7. 6번의 customize install location경로를 찾아 선택 후 확인 버튼을 누른다.



8.7번과 같은 방법으로C:\Program File\Python310에 있는 Scripts 선택 후 확인 버튼을 누른다.



9.C:\Program File\Python310과 C:\Program File\Python310\Scripts 경로를 위로 이동(U)아이콘을 실행하여 맨 위로 올린다.

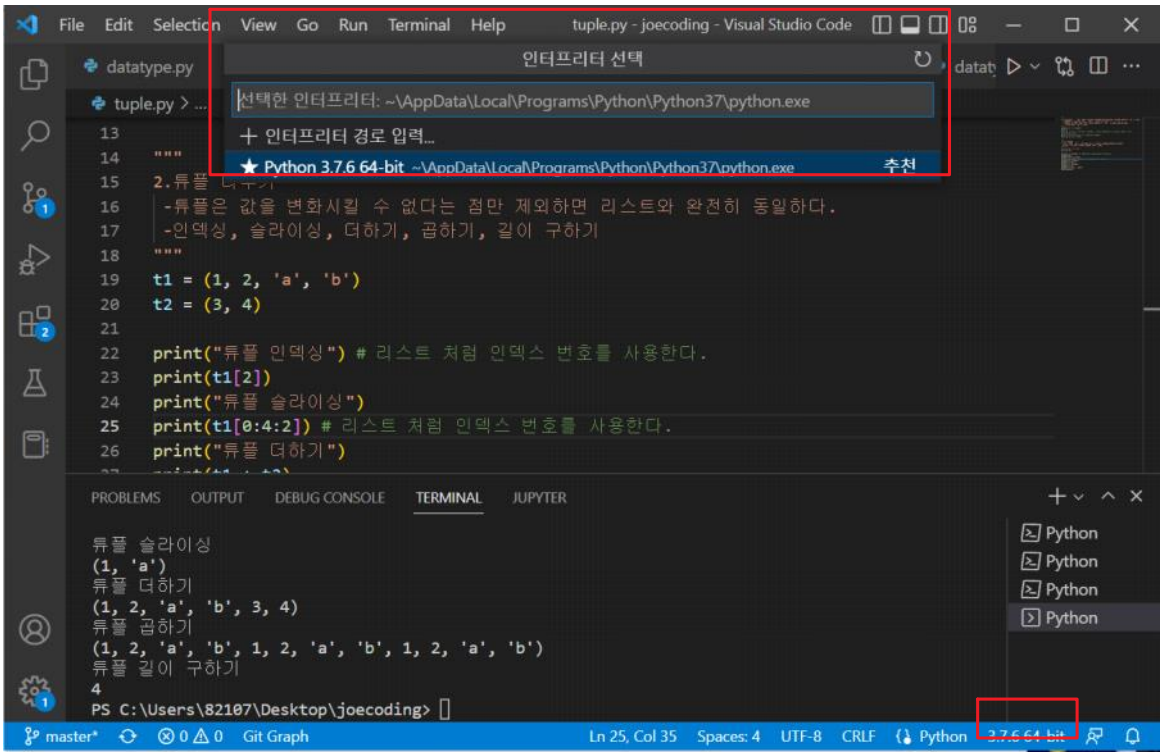

환경 변수
환경 변수 편집
C:₩Program Files₩Python310
C:₩Program Files₩Python310₩Scripts
C:₩Program Files₩NVIDIA GPU Computing Toolkit₩CUDA₩v11.3...
C:₩Program Files₩NVIDIA GPU Computing Toolkit₩CUDA₩v11.3...
C:₩Program Files₩Oculus₩Support₩oculus-runtime
C:₩Windows₩system32
C:₩Windows
C:₩Windows₩System32₩Wbem
C:₩Windows₩System32₩WindowsPowerShell₩v1.0₩
C:₩Windows₩System32₩OpenSSH₩
C:₩Program Files (x86)₩NVIDIA Corporation₩PhysX₩Common
C:₩Program Files₩NVIDIA Corporation₩NVIDIA NvDLISR
C:₩Program Files₩dotnet₩
C:₩Program Files₩Git₩cmd
C:₩Users₩jocoding₩anaconda3₩Scripts
C:₩Program Files₩NVIDIA Corporation₩Nsight Compute 2021.1...
C:₩Program Files₩nodejs₩
%SystemRoot%₩system32
%SystemRoot%
%SystemRoot%₩System32₩Wbem
%SYSTEMROOT%₩System32₩WindowsPowerShell₩v1.0₩
%SYSTEMROOT%₩System32₩OpenSSH₩
새로 만들기(N)
편집(E)
찾아보기(B)...
삭제(D)
위로 이동(U)
아래로 이동(O)
텍스트 편집(T)...
확인
취소
:46:23 • 파이썬 PATH 설정 >

VS CODE 설치 및 설정

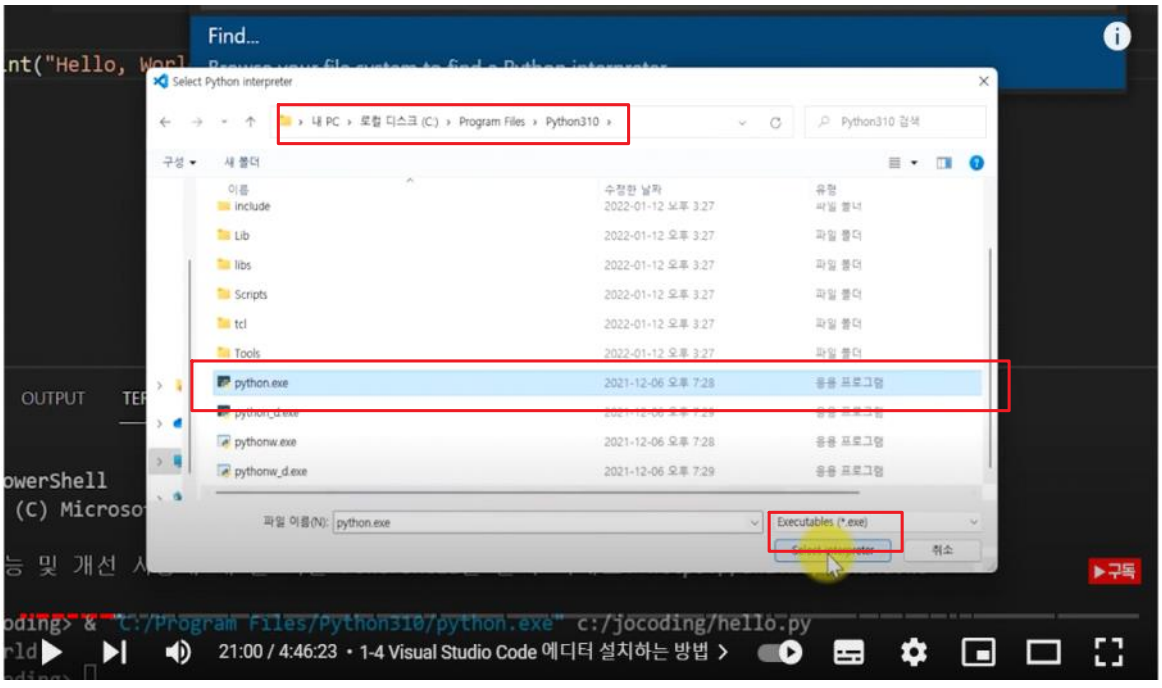
2022년 10월 14일 금요일 오후 2:44

*VS CODE에서 인터프리터 오류 발생시 해결 법

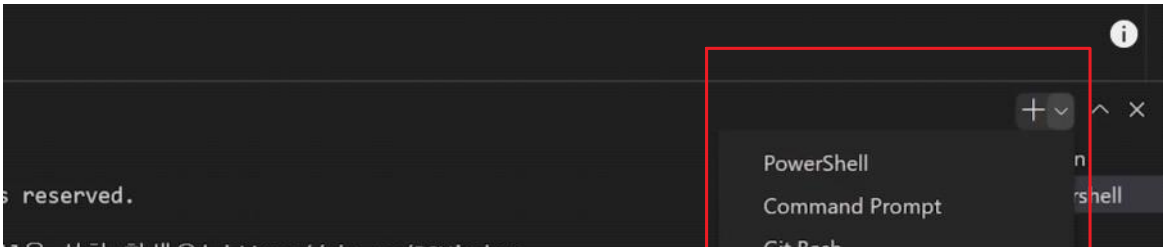
1.화면 아래에서 3.7.6 64-bit(설치한 파이썬 버전)를 선택하면 인퍼프리터 선택 창이 나오는데 거기서 +인터프리터 경로 입력 ... 클릭

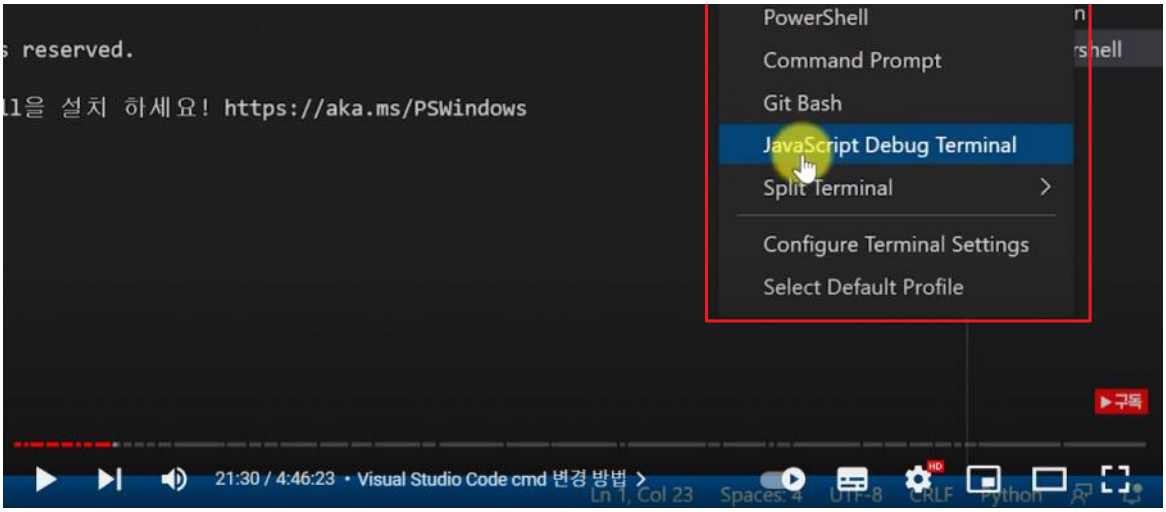


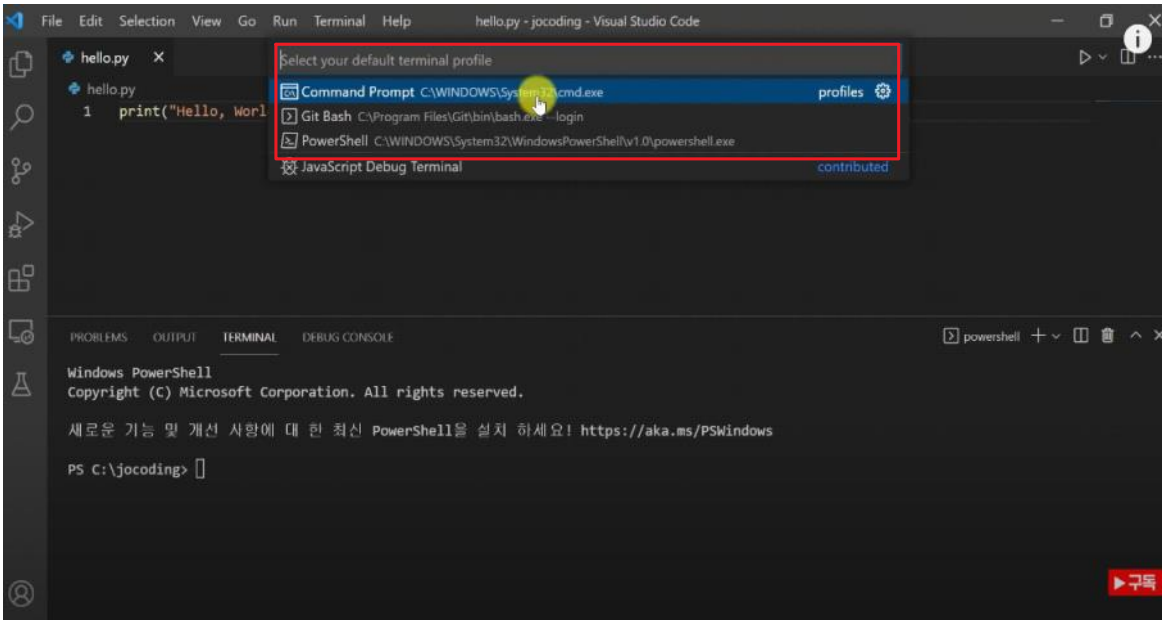
2.설치 한 파이썬 경로로 가서 python.exe 선택 후 select interpreter 클릭



*파이썬 프로그램 실행 프롬프트 변경 방법(원하는 프로그램 실행방법 설정은 Select Default Profile -> Command Prompt 선택한다.)




s reserved.
l을 설치 하세요! https://aka.ms/PSWindows
PowerShell
Command Prompt
Git Bash
JavaScript Debug Terminal
Split Terminal
Configure Terminal Settings
Select Default Profile
21:30 / 4:46:23 • Visual Studio Code cmd 변경 방법
구독


File Edit Selection View Go Run Terminal Help
hello.py - jocoding - Visual Studio Code
hello.py
print("Hello, Worl
Select your default terminal profile
Command Prompt
profiles
Git Bash
login
PowerShell C:\WINDOWS\System32\WindowsPowerShell\v1.0\powershell.exe
JavaScript Debug Terminal
contributed
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
powershell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
새로운 기능 및 개선 사항에 대 한 최신 PowerShell을 설치 하세요! https://aka.ms/PSWindows
PS C:\jocoding>
구독

리스트 자료형

2022년 10월 12일 수요일 오후 9:38

1.컨테이너 자료형의 종류
-리스트, 튜플, 딕셔너리, 집합, 변수

1-1.리스트 : 숫자나 문자 여러 개를 한 곳에 넣을 수 있다.
-리스트 만들기 : 대괄호안에 각 요소값을 쉼표로 구분해 작성 한다.
*리스트명 = [요소1, 요소2, 요소3, ...]

```
a = []
b = [1, 2, 3]
c = ['Life', 'is', 'too', 'short']
d = [1, 2, 'Life', 'is']
e = [1, 2, ['Life', 'is']]
```

리스트는 a처럼 아무것도 포함하지 않아 비어 있는 리스트([ ])일 수도 있고
b처럼 숫자를 요솟값으로 가질 수도 있고
c처럼 문자열을 요솟값으로 가질 수도 있다.
또한 d처럼 숫자와 문자열을 함께 요솟값으로 가질 수도 있으며
e처럼 리스트 자체를 요솟값으로 가질 수도 있다.

1-2.리스트 인덱싱과 슬라이싱
-리스트도 문자열처럼 인덱싱과 슬라이싱이 가능하다.

1-3.리스트 더하기(+)
-리스트 사이에서 + 기호는 2개의 리스트를 합치는 기능을 한다.

1-4.리스트 반복하기(*)
-리스트가 * 숫자만큼 반복되어 새로운 리스트를 만들어낸다.

```
a = [1, 2, 3]
b = [4, 5, 6]
print("리스트 더하기 : {}".format(a + b))
print(f"리스트 반복하기는 : {a * 3}")
```

리스트 더하기 : [1, 2, 3, 4, 5, 6]
리스트 반복하기는 : [1, 2, 3, 1, 2, 3, 1, 2, 3]

1-5.리스트 관련 함수들
-리스트에 요소 추가
1).append()
2).insert()
3).extend()

```
print("리스트에 요소 추가하기 append()") # 매개변수로 요소를 입력한다. 맨 마지막에 요소 값이 추가된다.
b.append(7)
print(b)
print("리스트에 요소 추가하기 insert()") # 매개변수로 원하는 위치의 인덱스와 값을 입력한다.
a.insert(3, 348)
print(a)
print("리스트에 요소 추가하기 extend()") # 매개변수로 대괄호(리스트)를 이용하여 원하는 갯수 만큼 요소를 추가한다.
b.extend(["안녕", 452])
print(b)
print()
```

-리스트 요소 제거

1).del 리스트이름[요소]

2).remove()

3).pop()

4).clear()

5) a[0:2] = [] # 원하는 범위를 빈 리스트로 대입한다.

```
print("리스트에 요소 제거하기 del 리스트[인덱스]") # 원하는 리스트의 인덱스 번호를 이용해 요소를 제거한다.
del a[2]
print(a)
print("리스트에 요소 제거하기 remove()") # 매개변수로 제거하고자 하는 요소의 값을 적는다.
b.remove(6)
print(b)
print("리스트에 요소 제거하기 pop()") # 매개변수로 원하는 리스트의 인덱스 번호를 지정하지 않으면 맨 끝에 있는 요소만 제거된다.
a.pop(0)
print(a)
print("리스트에 요소 제거하기 clear()") # 리스트 안에 있는 요소 전체가 제거된다.
print(a.clear())
print()
```

-리스트 정렬

1).sort()

2).reverse()

```
print("리스트 정렬 sort()")
a = ['a', 'c', 'b']
a.sort()
print(a)
print("리스트 뒤에서 부터 정렬 reverse()") # 역순으로 정렬하는 것이 아니라 그저 현재의 리스트를 그대로 거꾸로 뒤집는다.
c = [1, 4, 3, 2]
c.reverse()
print(c)
print("리스트 역정렬 방법") # sort(reverse=True)를 이용한다.
c.sort(reverse=True)
print(c)
```

-리스트 요소 찾기

1).index() # 찾고자 하는 요소를 매개변수로 지정한다 결과값은 요소의 인덱스번호를 돌려준다.

```
print("리스트 요소 찾기 index()")
print(c.index(3)) # 찾고자 하는 요소를 매개변수로 지정한다 결과값은 요소의 인덱스번호를 돌려준다.
```

-요소 개수 세기

1).count() # 요소를 매개변수로 지정하면 결과값으로 리스트 안에 요소가 몇 개 있는지 그 개수를 알려준다.

```
f = [1, 2, 3, 1, 1, 1]
print("요소 개수 세기 count()")
print(f.count(1)) # 요소를 매개변수로 지정하면 결과값으로 리스트 안에 요소가 몇 개 있는지 그 개수를 알려준다.
```

튜플 자료형

2022년 10월 14일 금요일 오전 9:34

1.튜플(tuple)은 몇 가지 점을 제외하곤 리스트와 거의 비슷하며 리스트와 다른 점은 다음과 같다.
-리스트는 [ ]으로 둘러싸지만 튜플은 ( )으로 둘러싼다.
-리스트는 그 값의 생성, 삭제, 수정이 가능하지만 튜플은 그 값을 바꿀 수 없다.
*튜플의 모습은 다음과 같다.

```
>>> t1 = ()
>>> t2 = (1,) # 단지 1개의 요소만을 가질 때는 요소 뒤에 콤마(,)를 반드시 붙여야 한다
>>> t3 = (1, 2, 3)
>>> t4 = 1, 2, 3 # 괄호( )를 생략해도 무방하다
>>> t5 = ('a', 'b', ('ab', 'cd'))
```

2.튜플 다루기
-튜플은 값을 변화시킬 수 없다는 점만 제외하면 리스트와 완전히 동일하다.

-인덱싱하기

```
>>> t1 = (1, 2, 'a', 'b')  # 리스트 처럼 인덱스 번호를 사용한다.
>>> t1[0]
    1
>>> t1[3]
   'b'
```

문자열, 리스트와 마찬가지로 t1[0], t1[3]처럼 인덱싱이 가능하다.

-슬라이싱하기

```
>>> t1 = (1, 2, 'a', 'b')
>>> t1[1:] # 리스트 처럼 인덱스 번호를 사용한다.
   (2, 'a', 'b')
```

*t1[1]부터 튜플의 마지막 요소까지 슬라이싱하는 예이다.

-튜플 더하기

```
>>> t1 = (1, 2, 'a', 'b')
>>> t2 = (3, 4)
>>> t1 + t2
  (1, 2, 'a', 'b', 3, 4)
```

튜플을 더하는 방법을 보여 주는 예이다.

-튜플 곱하기

```
>>> t2 = (3, 4)
>>> t2 * 3
```

```
    (3, 4, 3, 4, 3, 4)
```

튜플의 곱하기(반복) 예를 보여 준다.

튜플 길이 구하기

```
>>> t1 = (1, 2, 'a', 'b')
>>> len(t1)
    4
```