Lecture #0. 파이썬 기초 (1)

2D 게임 프로그래밍

이대현 교수

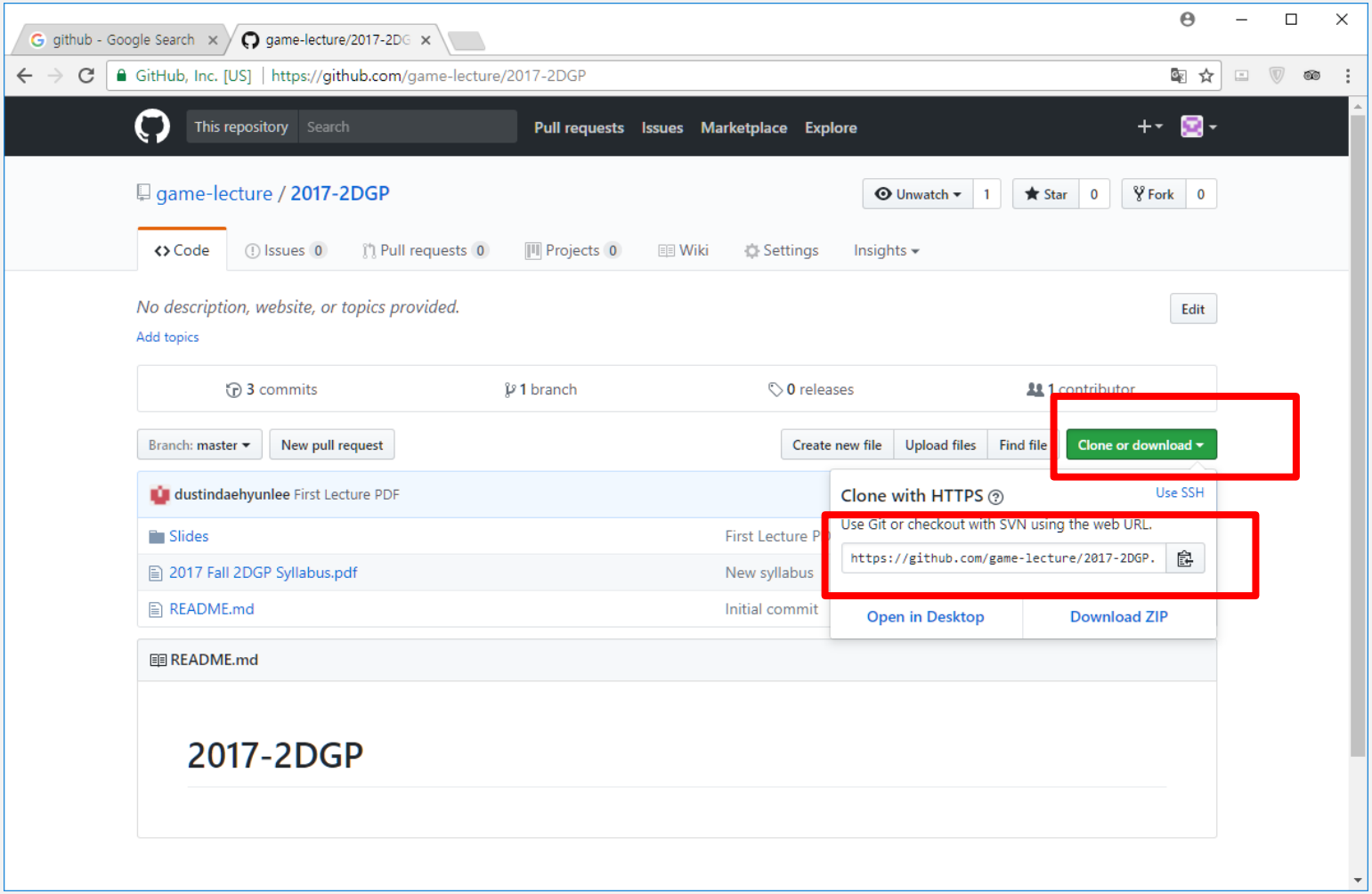
한국산업기술대학교
KOREA POLYTECHNIC UNIVERSITY

```
name = input('Enter file:')
handle = open(name)

counts = dict()
for line in handle:
    words = line.split()
    for word in words:
        counts[word] = counts.get(word,0) + 1

bigcount = None
bigword = None
for word,count in counts.items():
    if bigcount is None or count > bigcount:
        bigword = word
        bigcount = count

print(bigword, bigcount)
```

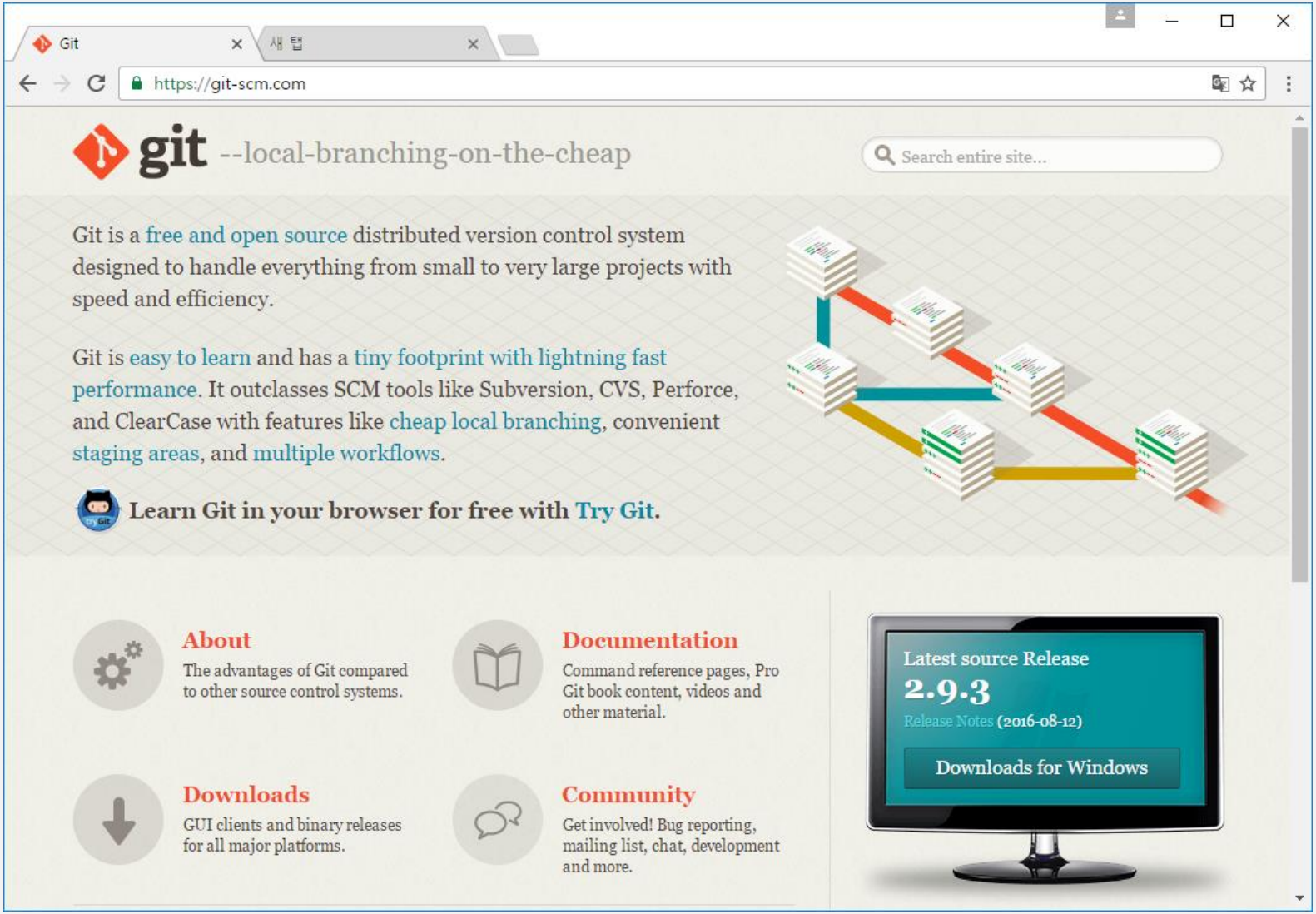
강의및실습자료

2DGP Github 저장소

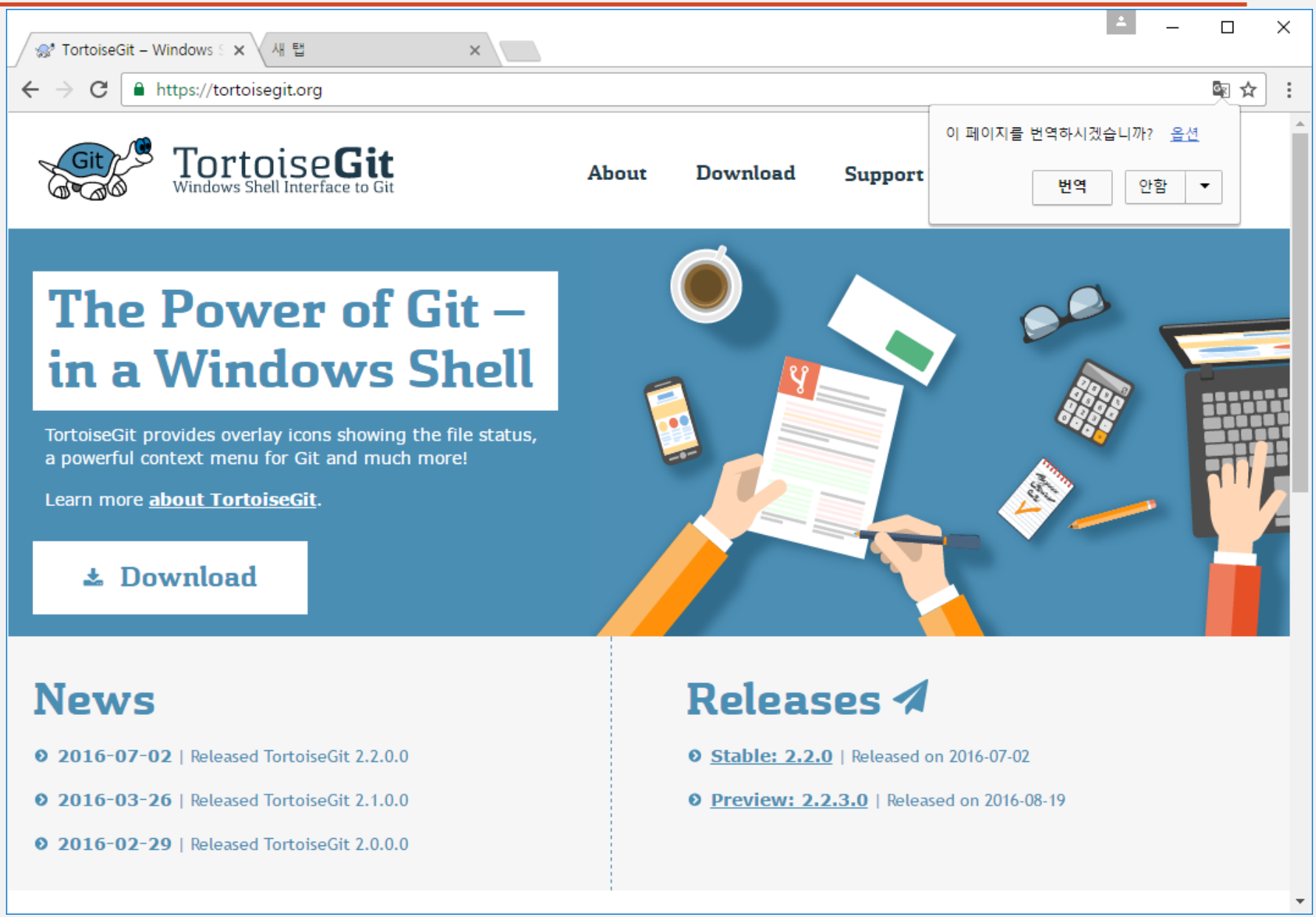
https://github.com/game-lecture/2018-2DGP.git



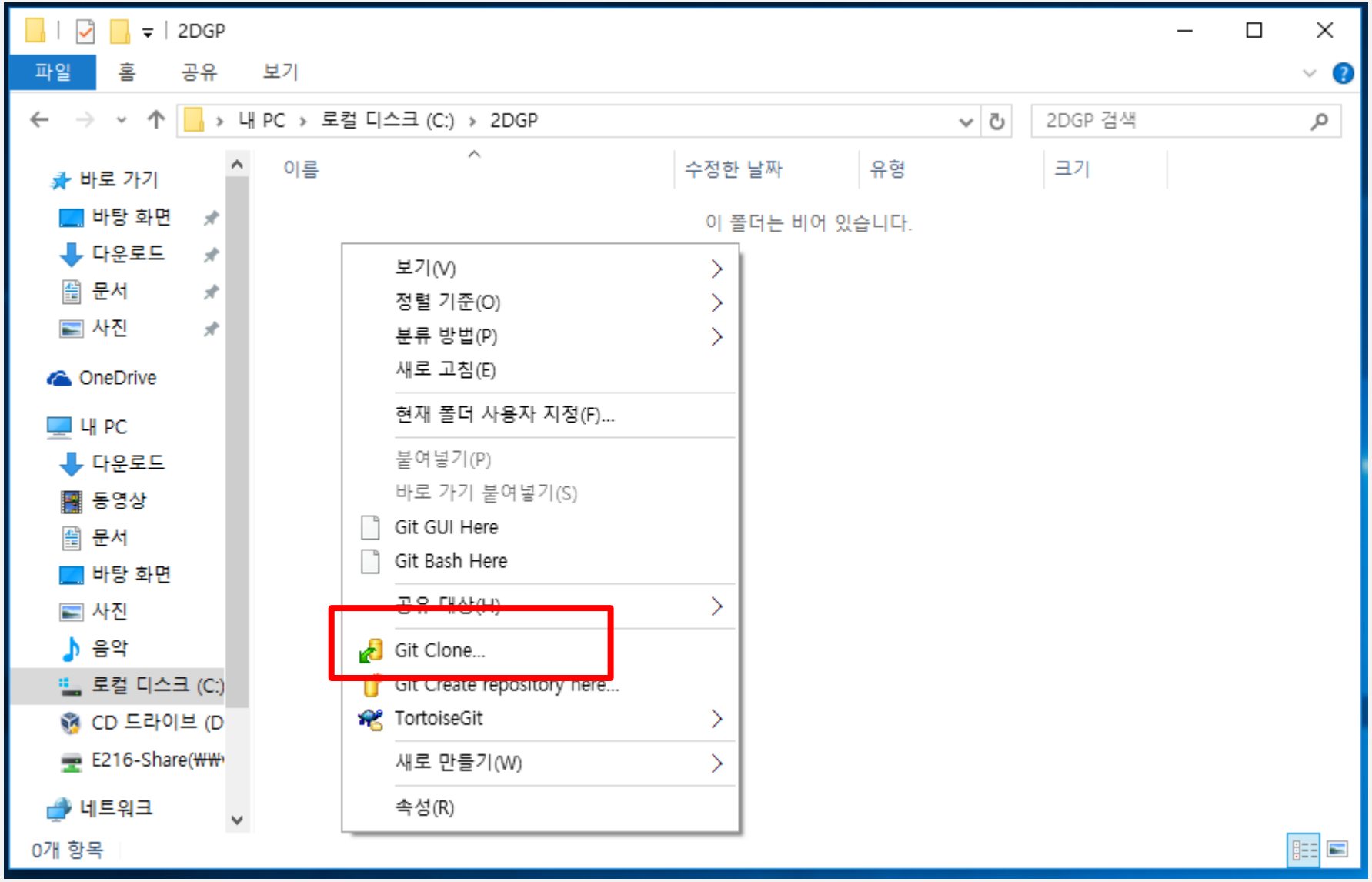
Git 의 설치



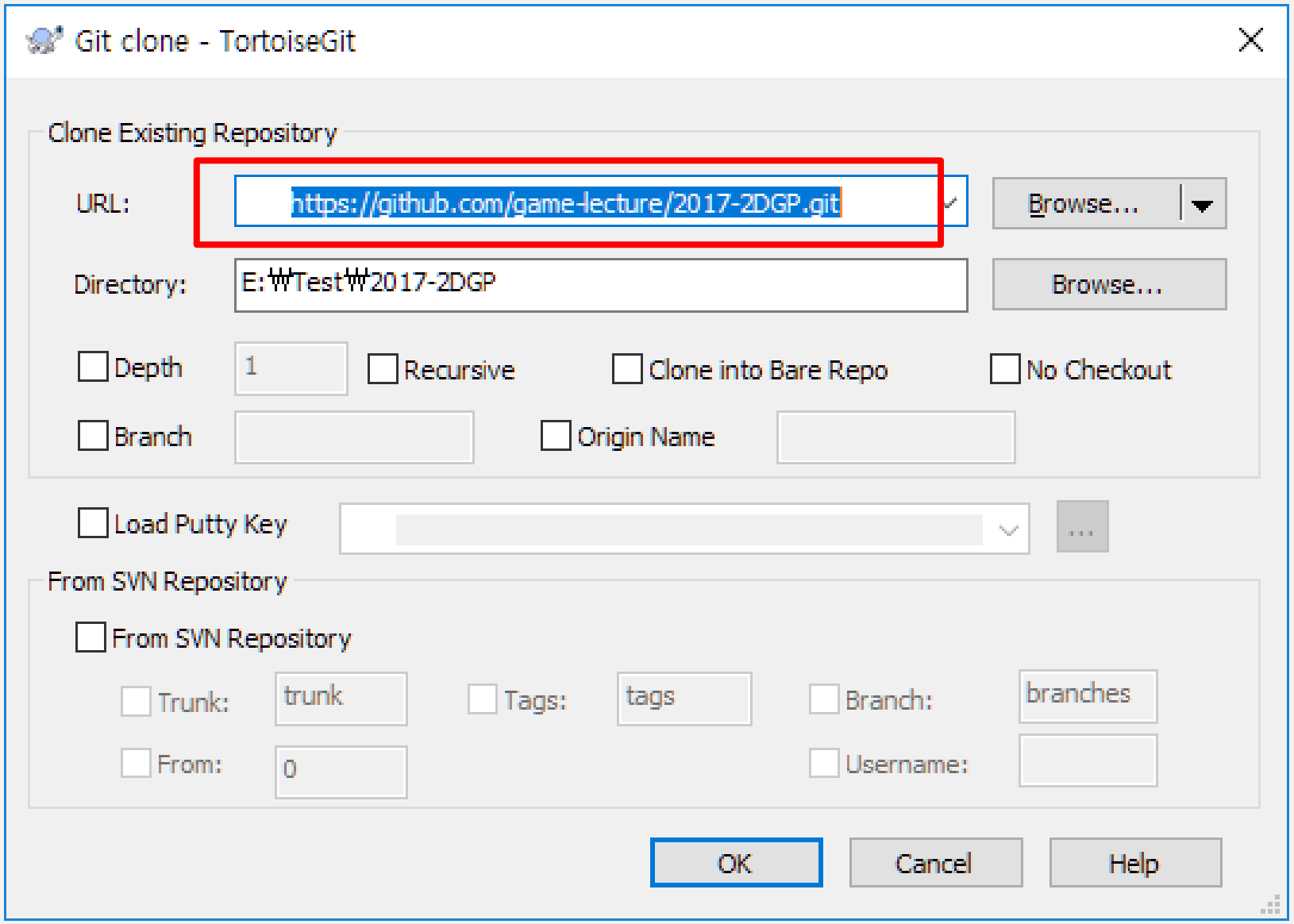
TortoiseGit 의 설치

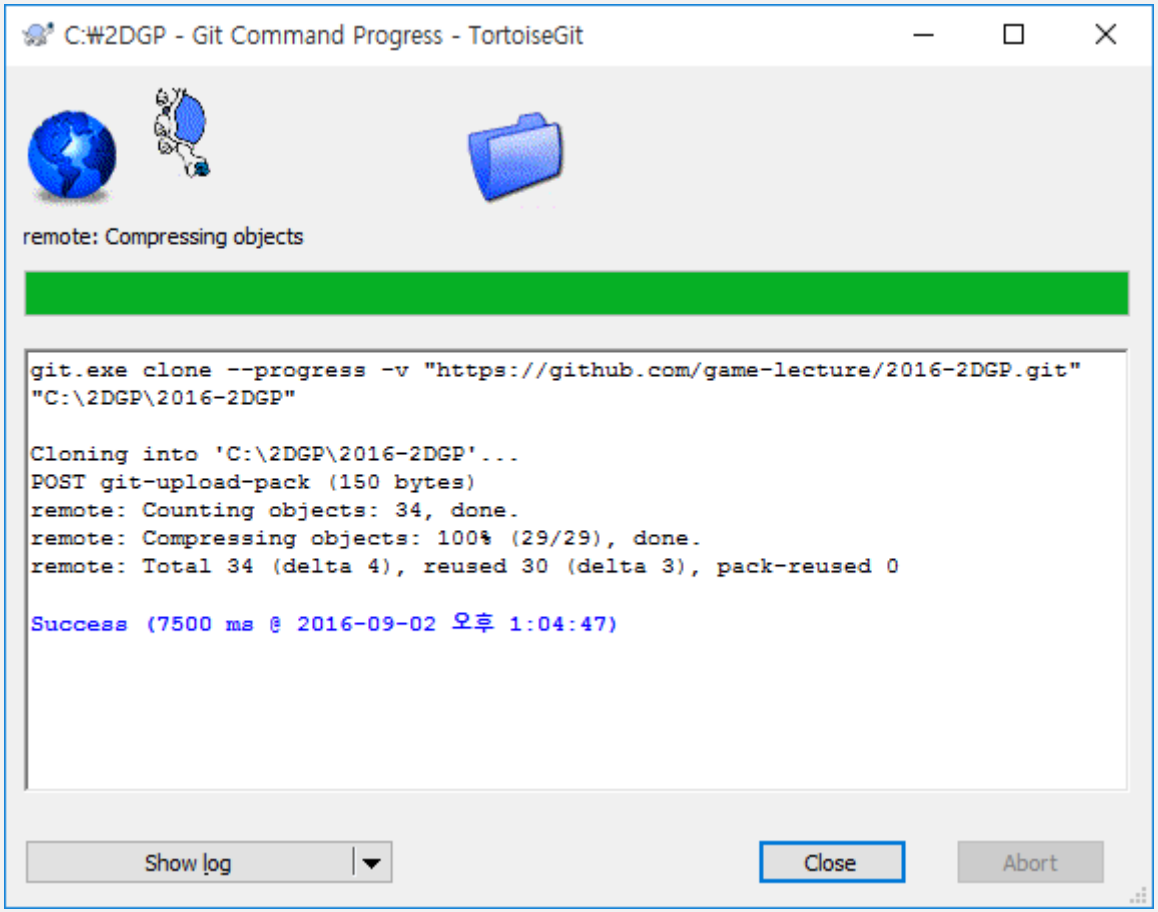


Git Clone



URL: https://github.com/game-lecture/2018-2DGP.git





파이썬의 특징

인간다운 언어이다

프로그래밍이란 컴퓨터에 인간이 생각하는 것을 입력시키는 행위라고 할 수 있다. 앞으로 살펴볼 파이썬 문법들에서도 볼 수 있겠지만 파이썬은 사람이 생각하는 방식을 그대로 표현할 수 있도록 해주는 언어이다. 따라서 프로그래머는 굳이 컴퓨터식 사고 방식으로 프로그래밍을 하려고 애쓸 필요가 없다. 이제 곧 어떤 프로그램을 구상하자마자 생각한대로 쉽게 술술 써내려가는 여러분의 모습에 놀라게 될 것이다. 아래 예문을 보면 이 말이 더 쉽게 이해될 것이다.

```
if 4 in [1,2,3,4]: print ("4가 있습니다")
```

위의 예제는 다음처럼 읽을 수 있다:

"만약 4가 1,2,3,4중에 있으면 "4가 있습니다"를 출력한다."

프로그램을 모르더라도 직관적으로 무엇을 뜻하는지 알 수 있지 않겠는가? 마치 영어문장을 읽는 듯한 착각에 빠져든다.

간결하다

파이썬은 간결하다. 이 간결함은 파이썬을 만든 귀도(Guido)의 의도적인 산물이다. 만약 어떤 언어(Perl?)가 하나의 일을 하기 위한 방법이 100가지라면 파이썬은 가장 좋은 방법 1가지를 선호한다. 이 파이썬의 간결함이란 철학은 소스코드에도 그대로 적용되어 파이썬 프로그래밍을 하는 사람들은 잘 정리되어 있는 소스코드를 볼 수 있게 되었다. 다른 사람들의 소스 코드가 한눈에 들어오기 때문에 이 간결함은 공동 작업에 매우 큰 역할을 하게 되었다. 다음은 파이썬 프로그램의 예제이다:

```
# simple.py
languages = ['python', 'perl', 'c', 'java']

for lang in languages:
    if lang in ['python', 'perl']:
        print("%6s need interpreter" % lang)
    elif lang in ['c', 'java']:
        print("%6s need compiler" % lang)
    else:
        print("should not reach here")
```

프로그래밍이 재밌다

이 부분이 가장 강조하고 싶은 부분이다. 필자에게 파이썬만큼 프로그래밍을 하는 순간을 즐기게 해준 언어는 없었던 것 같다. 파이썬은 다른 것에 신경 쓸 필요 없이 내가 하고자 하는 부분에만 집중할 수 있게 해주기 때문이다. 억지로 만든 프로그램과 즐기면서 만든 프로그램, 과연 어떤 프로그램이 좋을까? 리누즈 토발즈는 재미로 리눅스를 만들었다고 하지 않는가? 파이썬을 배우고 나면 다른 언어로 프로그래밍을 하는 것에 지루함을 느끼게 될 지도 모른다. 조심하자! ^^

개발속도가 빠르다

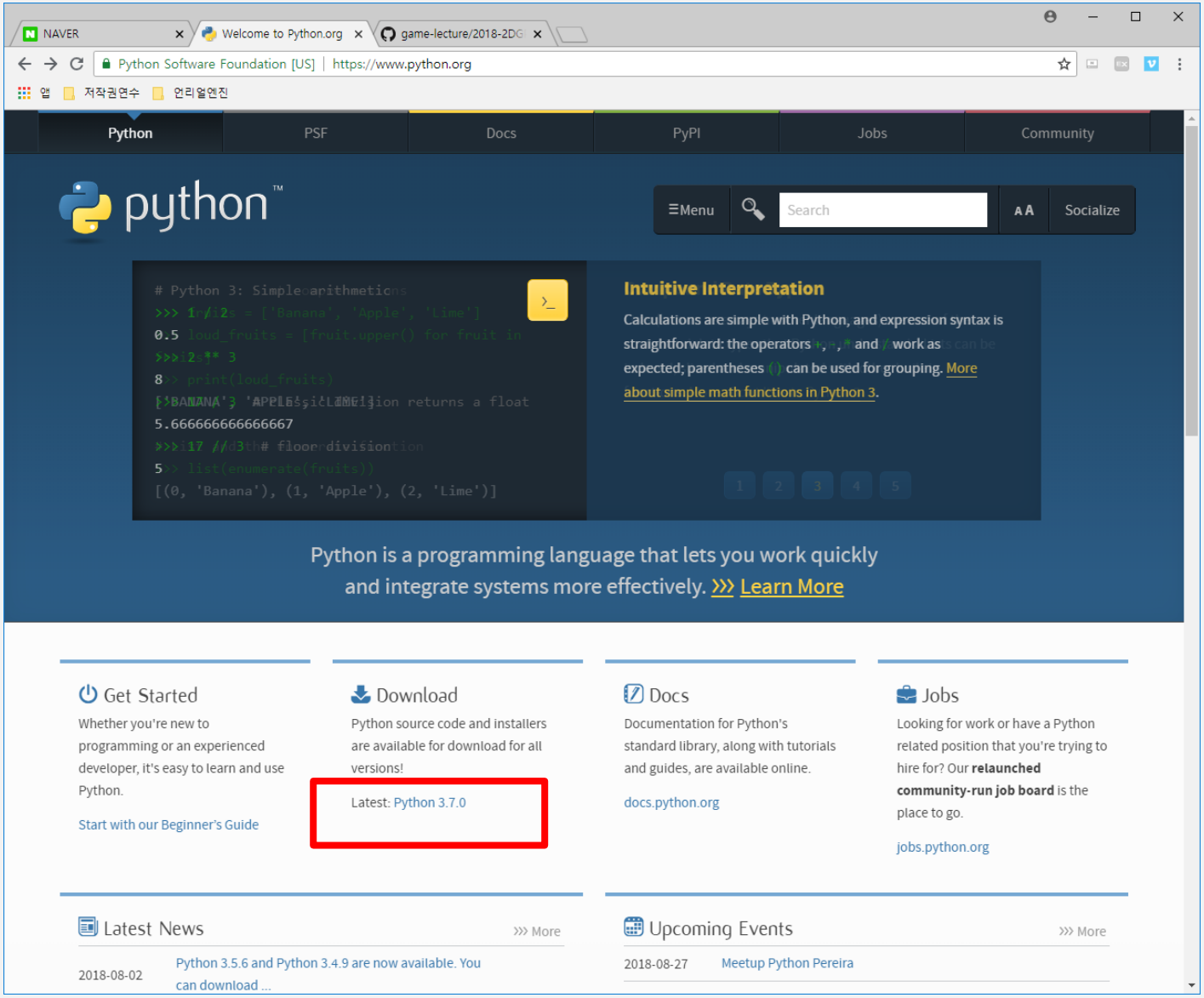
마지막으로 다음의 재미있는 말로 파이썬의 특징을 마무리하려 한다.

Life is too short, You need python.

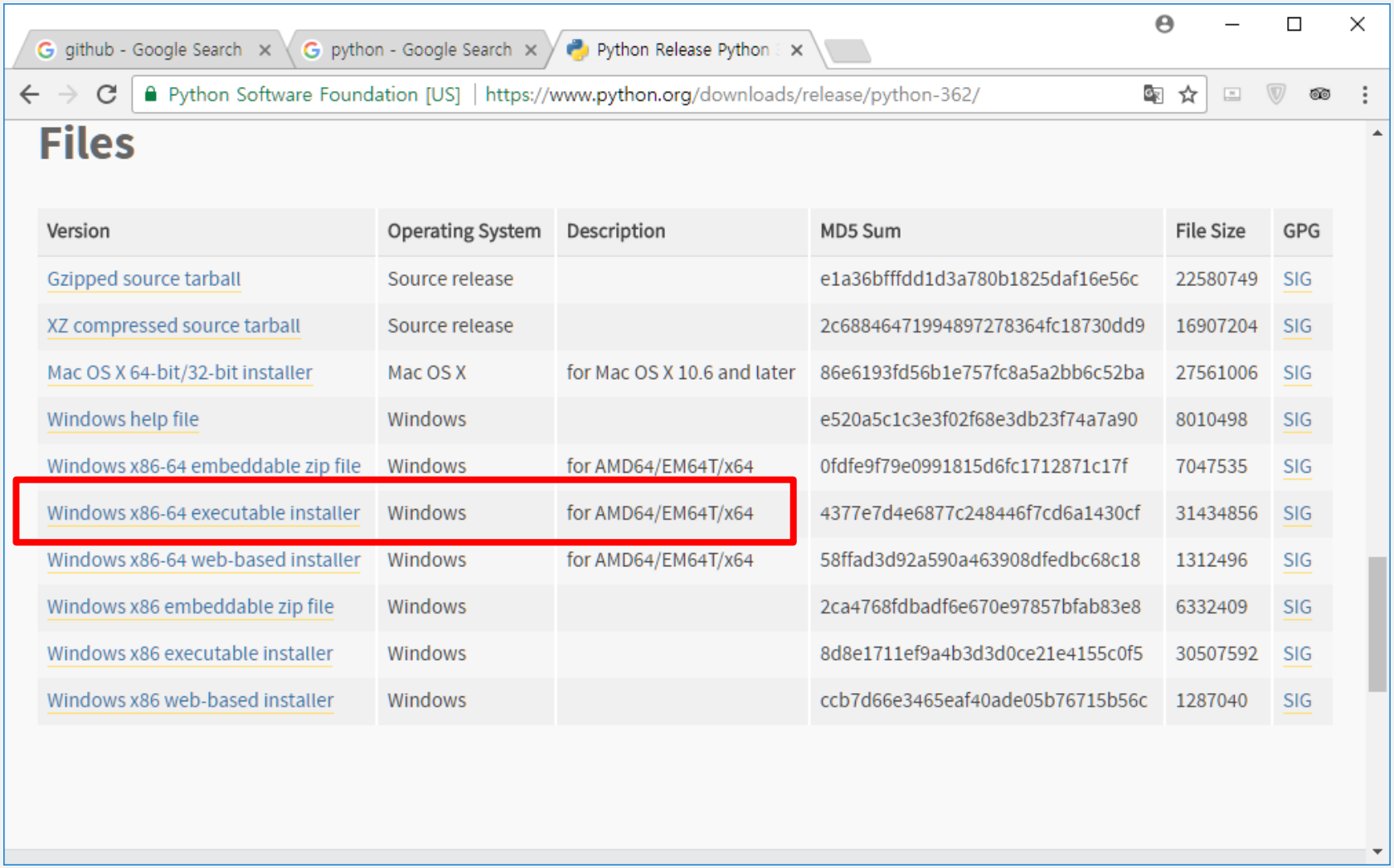
Python Key Words

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

파이썬 홈페이지 – https://www.python.org

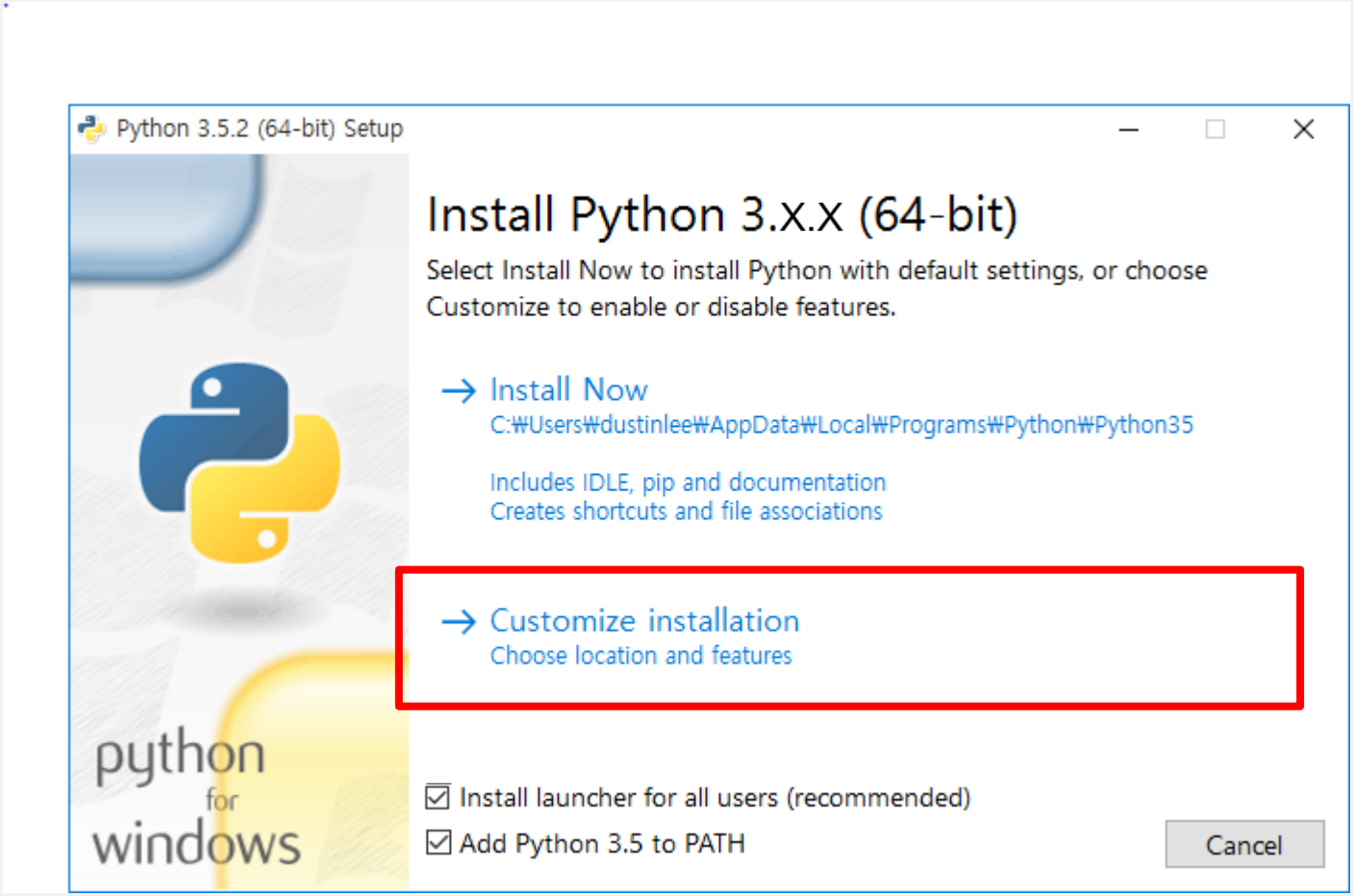


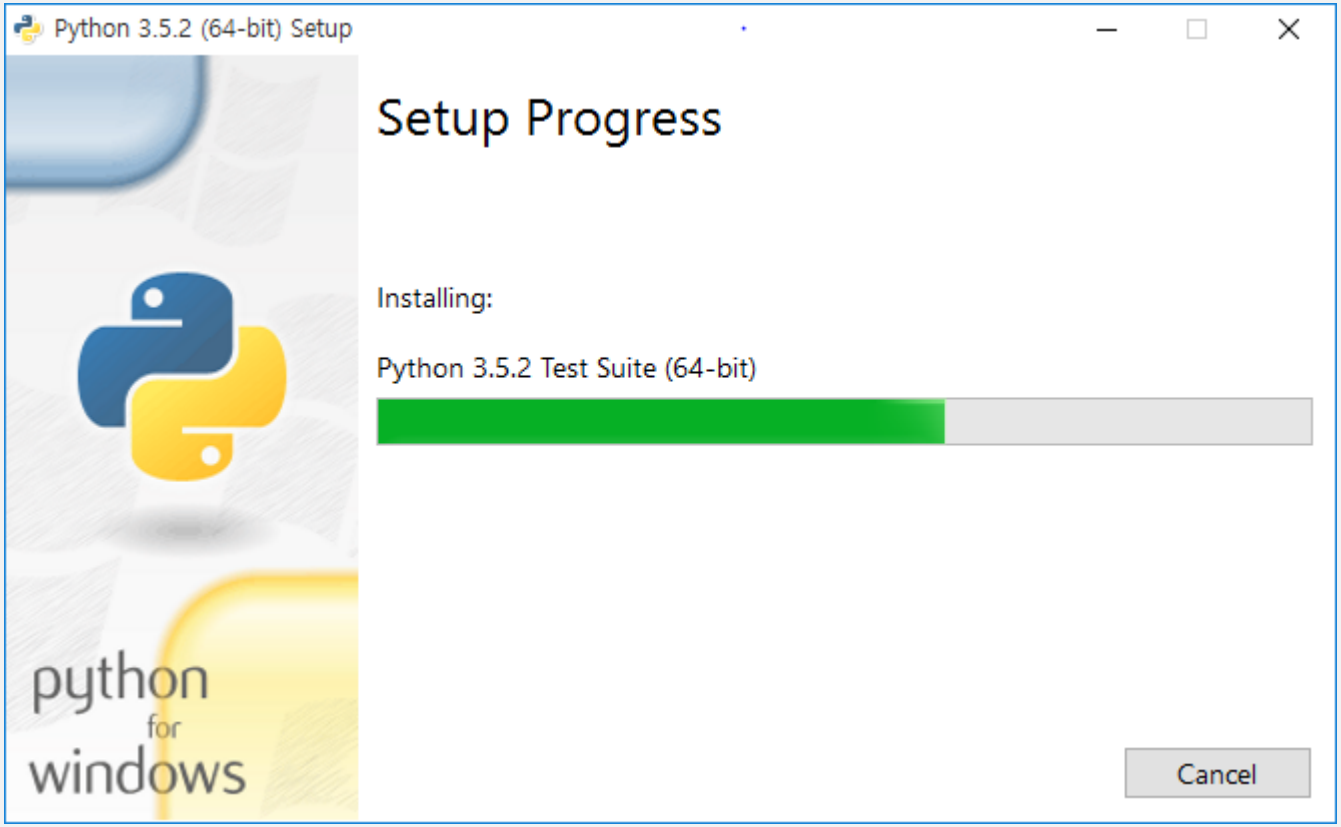
Python 최신 버전 다운로드 – 중요 ! 반드시 64bit version 을 다운로드



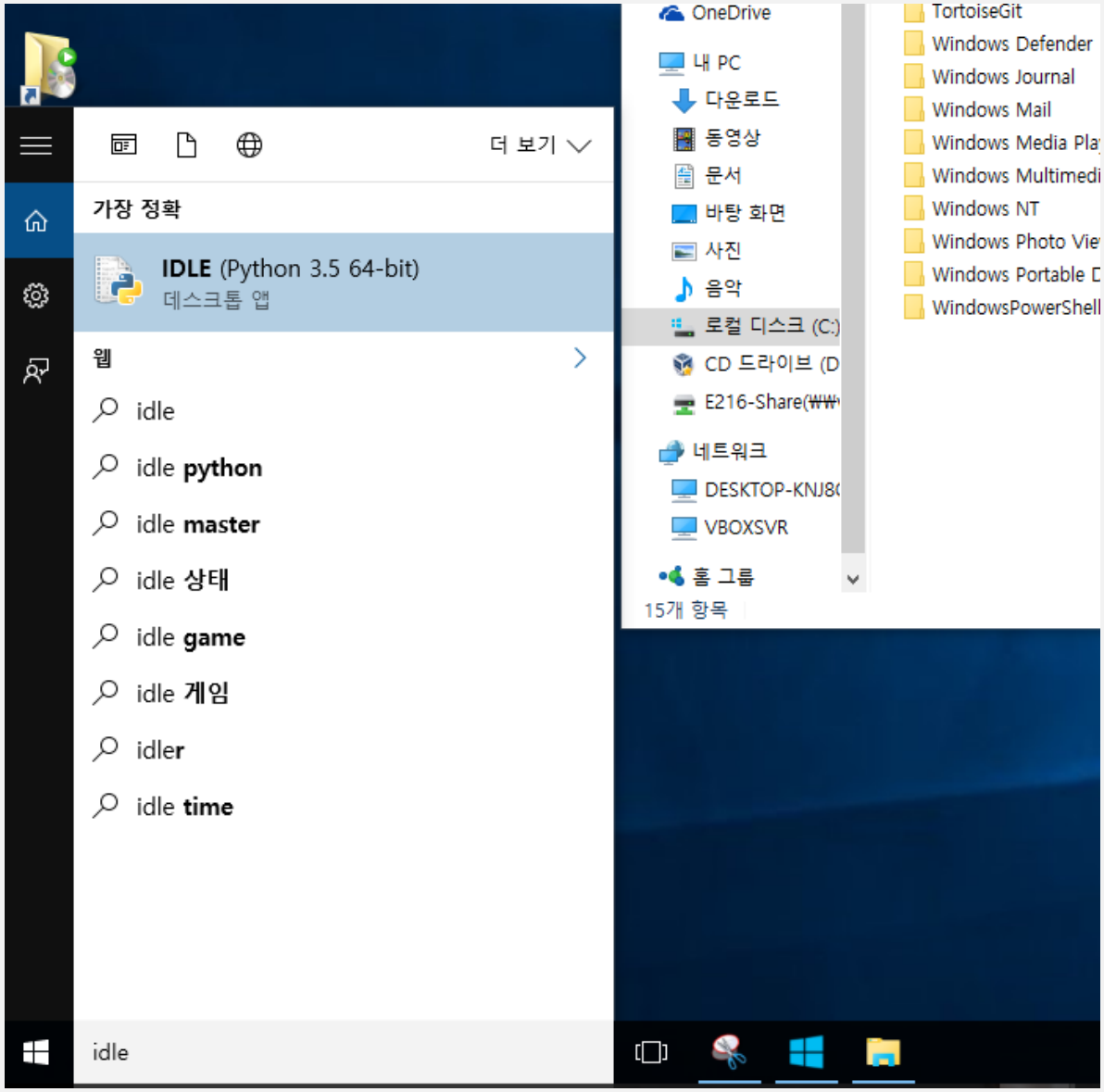
Files

Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		e1a36bfffdd1d3a780b1825daf16e56c	22580749	SIG
XZ compressed source tarball	Source release		2c6884647199489727836 4fc18730dd9	16907204	SIG
Mac OS X 64-bit/32-bit installer	Mac OS X	for Mac OS X 10.6 and later	86e6193fd56b1e757fc8a5a2bb6c52ba	27561006	SIG
Windows help file	Windows		e520a5c1c3e3f02f68e3db23f74a7a90	8010498	SIG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	0fdfe9f79e0991815d6fc1712871c17f	7047535	SIG
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	4377e7d4e6877c248446f7cd6a1430cf	31434856	SIG
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	58ffad3d92a590a463908dfedbc68c18	1312496	SIG
Windows x86 embeddable zip file	Windows		2ca4768fdbadf6e670e97857bfab83e8	6332409	SIG
Windows x86 executable installer	Windows		8d8e1711ef9a4b3d3d0ce21e4155c0f5	30507592	SIG
Windows x86 web-based installer	Windows		ccb7d66e3465eaf40ade05b76715b56c	1287040	SIG



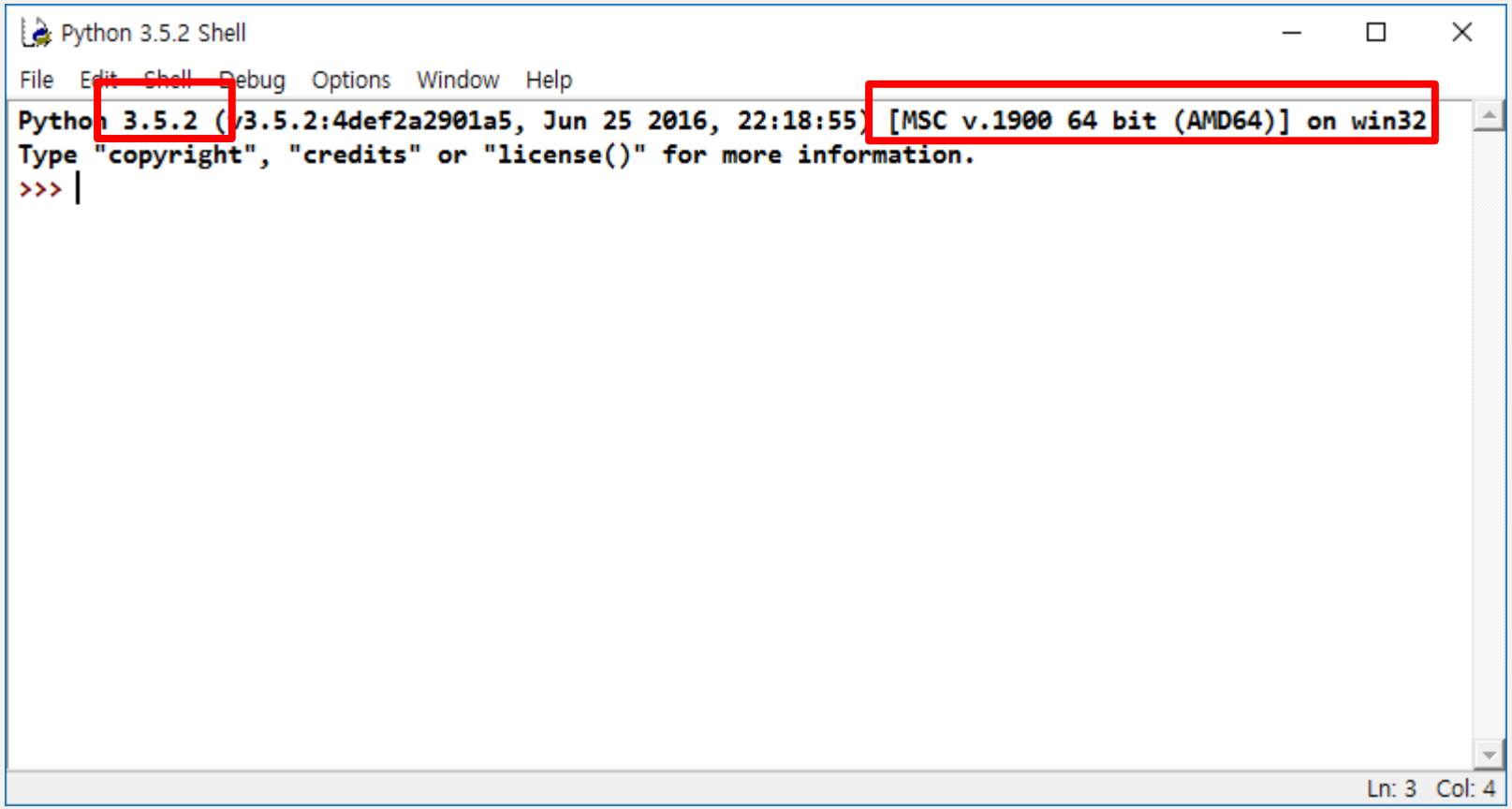


IDLE의 실행



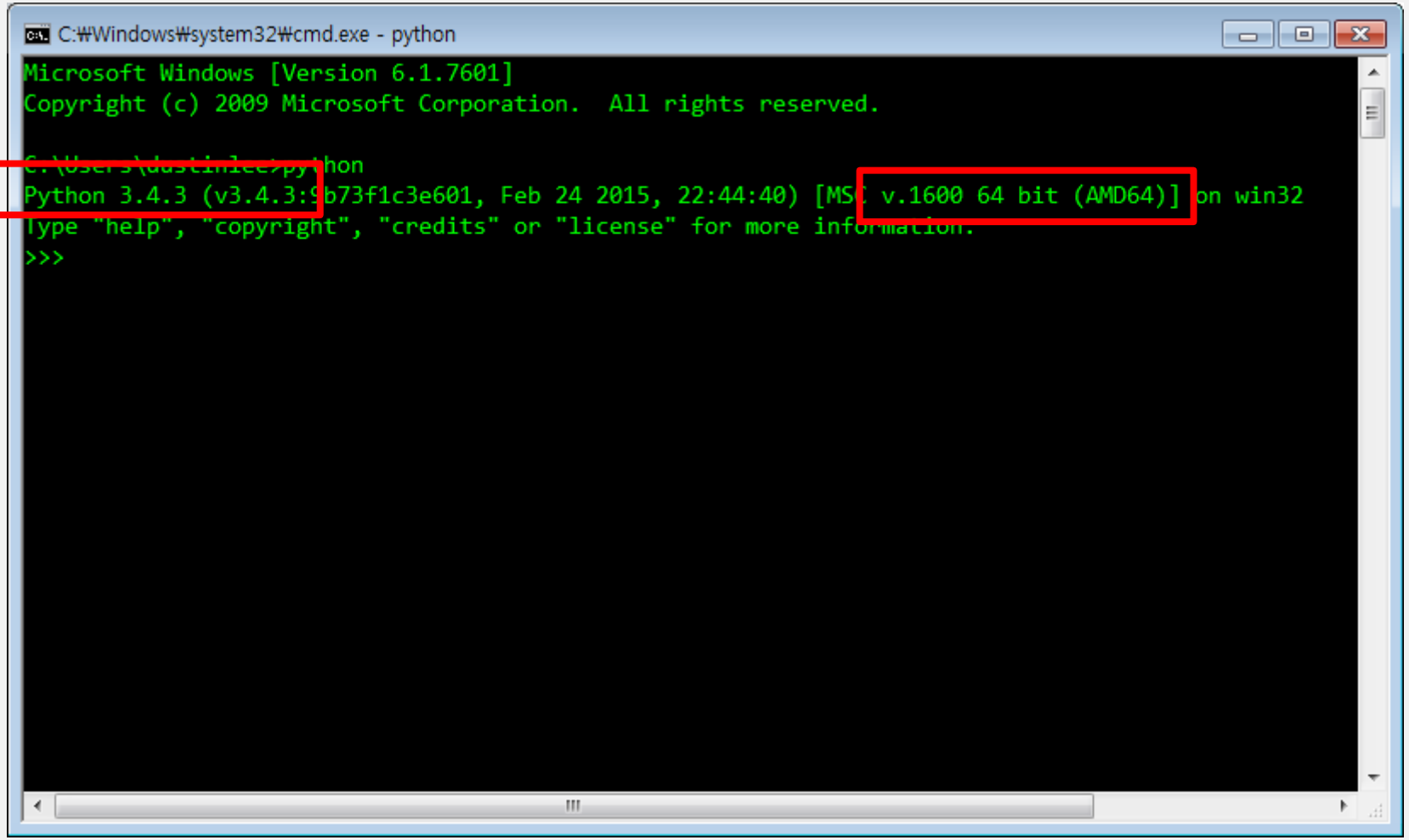
설치 확인 포인트 (1)

- 32bit 로 설치했으면, IDLE 실행될 때, 설치한 버전명 그리고 32bit 로 표시
- 64bit 로 설치했으면, IDLE 실행될 때, 설치한 버전명 그리고 64bit(AMD64)로 표시



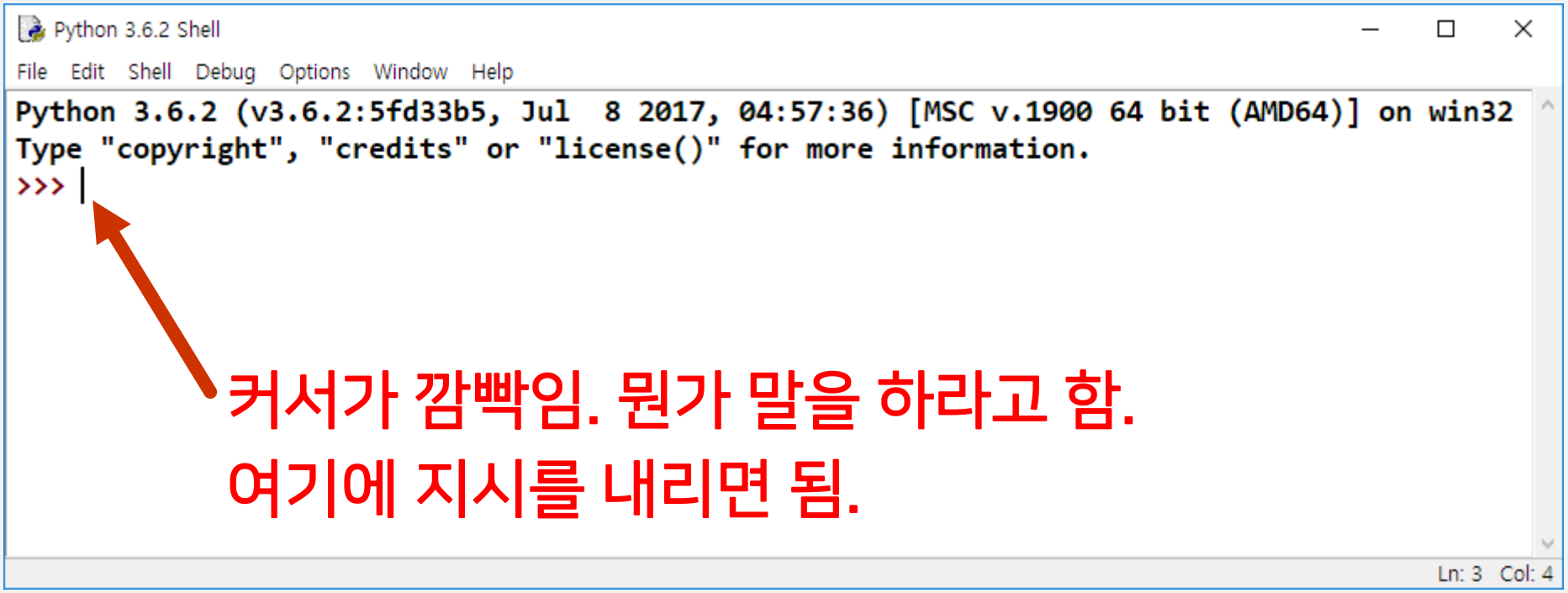
설치 확인 포인트 (2)

- cmd 창을 열어서, python 이라고 치면, 앞서 IDLE 을 실행했을 때와 동일하게 32bit 또는 64 bit가 나와야 함. **중요! 설치한 최신 버전과 동일한 버전으로 표시되어야 함.**

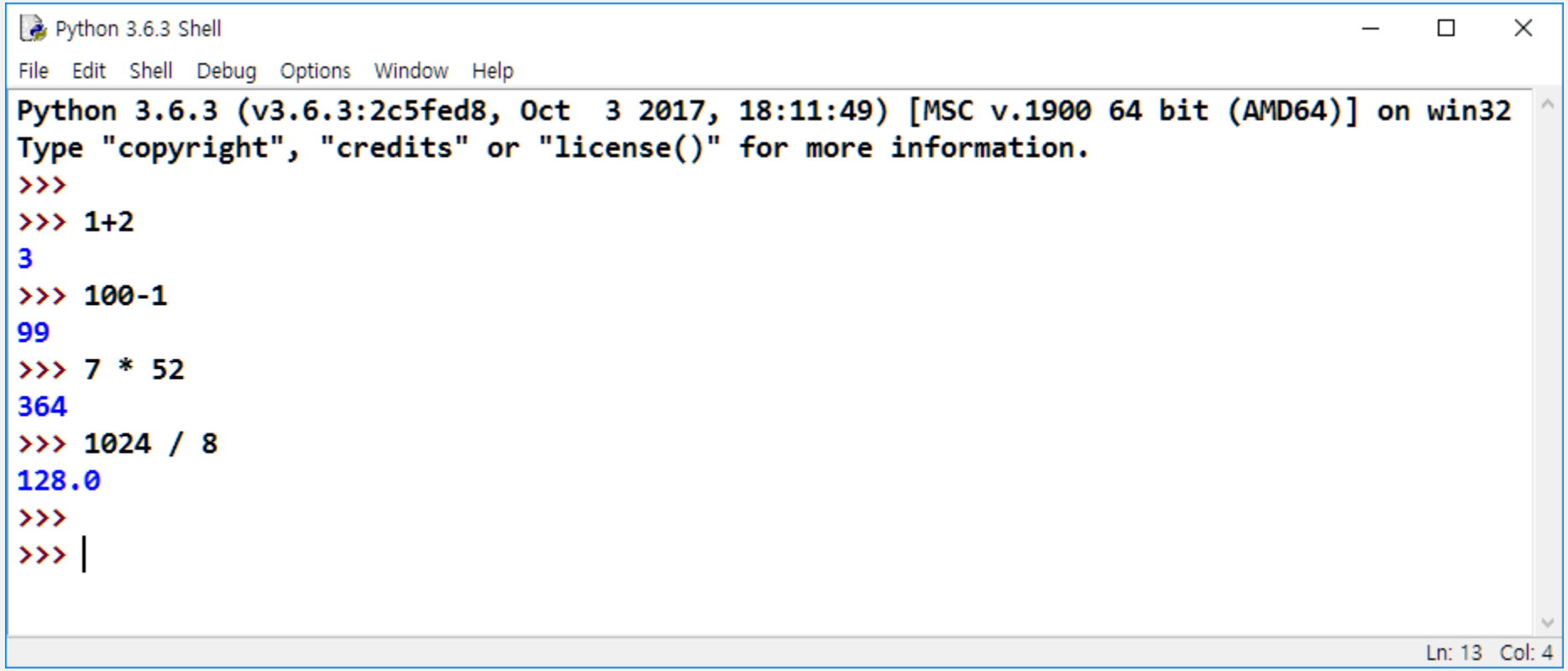


IDLE 실행 화면

- Python 언어로 지시하면, 이를 해석해서 일을 함.



계산을 시켜보자.



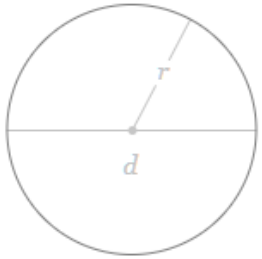
원의 넓이를 구해보자. 반지름이 3미터 이면?

Circle

Solve for area

$$A = \pi r^2$$

r Radius Enter value



Python 3.6.3 Shell

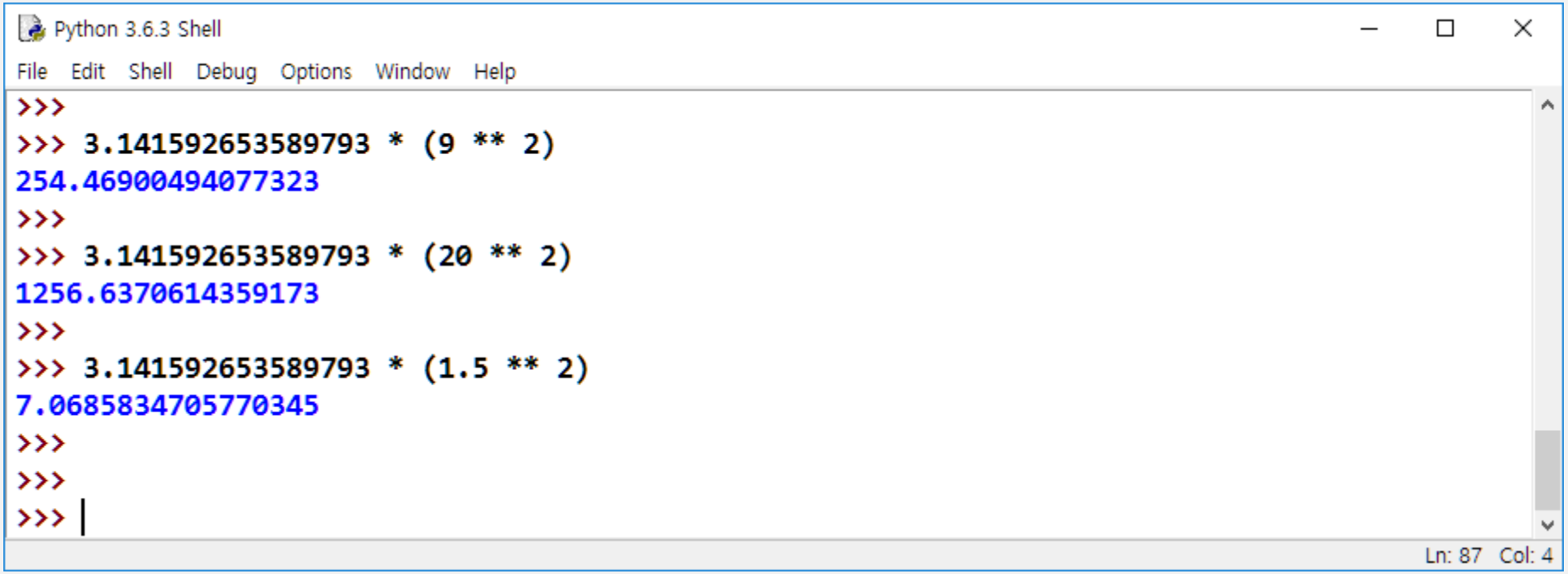
File Edit Shell Debug Options Window Help

```
>>>
>>>
>>>
>>>
>>>
>>> 3.141592653589793 * (3 * 3)
28.274333882308138
>>>
>>> 3.141592653589793 * 3 ** 2
28.274333882308138
>>>
>>>
```

28.274328 평방미터

Ln: 77 Col: 4

반지름이 9미터이면? 20 미터면? 1.5미터면?

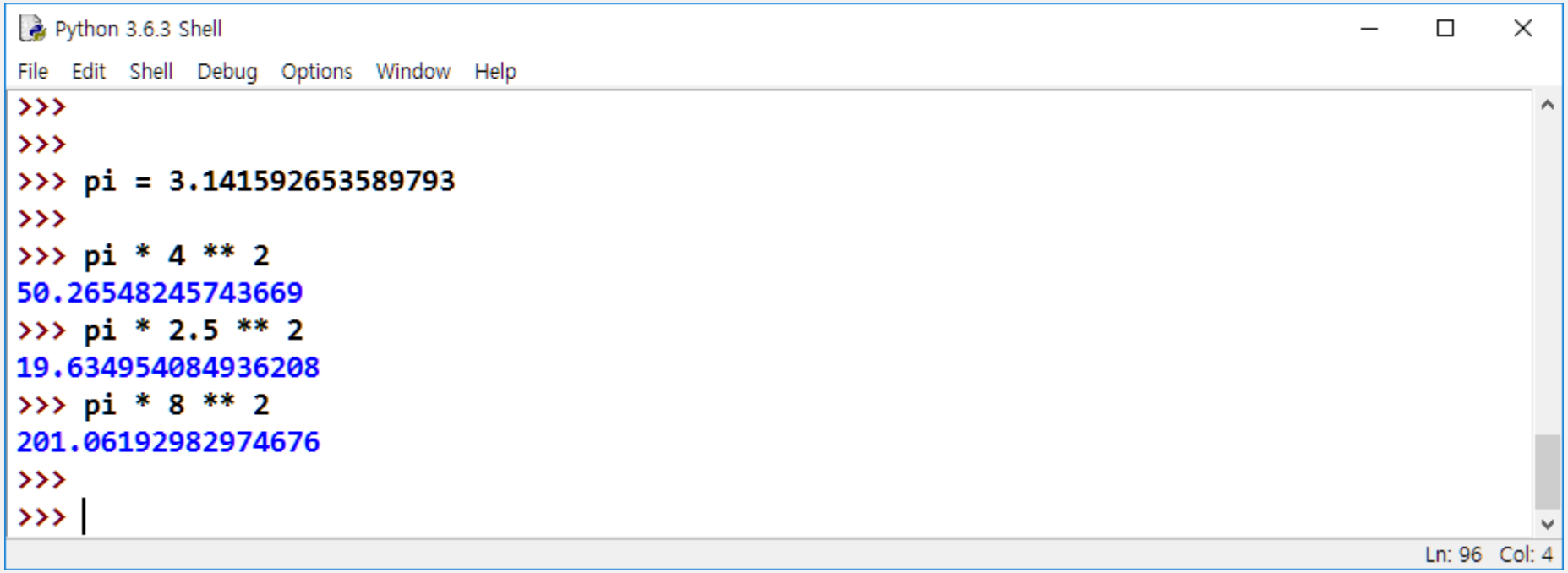


슬슬 귀찮아지기 시작한다.

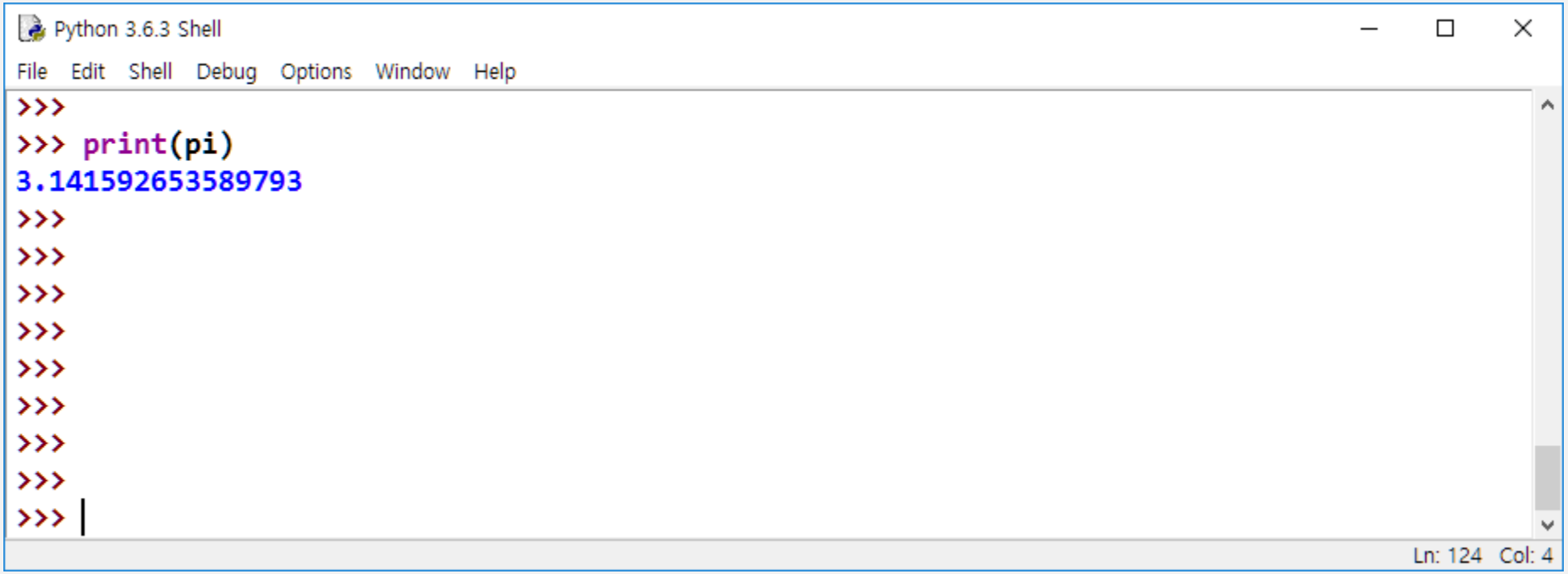
3.141592… 를 어디엔가 기록해놓고, 이걸 재사용하면 좋을 것 같은데….

변수(variable)

- 변수: 값을 저장해놓는 컴퓨터 메모리 안의 공간
- 변수는 해당되는 이름이 있다. 프로그래머가 이름을 지어야 함.
- 이름은 영문자와 숫자를 조합해서 씀. 단, 파이썬의 기본 단어는 쓰면 안됨.
- 사실, 변수의 값은 맘대로 언제든지 바꿀 수 있다.

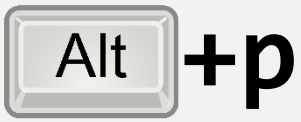


print 함수를 이용하면, 변수의 값을 볼 수 있다.



반지름 변수 r과 면적 변수 area 를 사용한 면적 계산

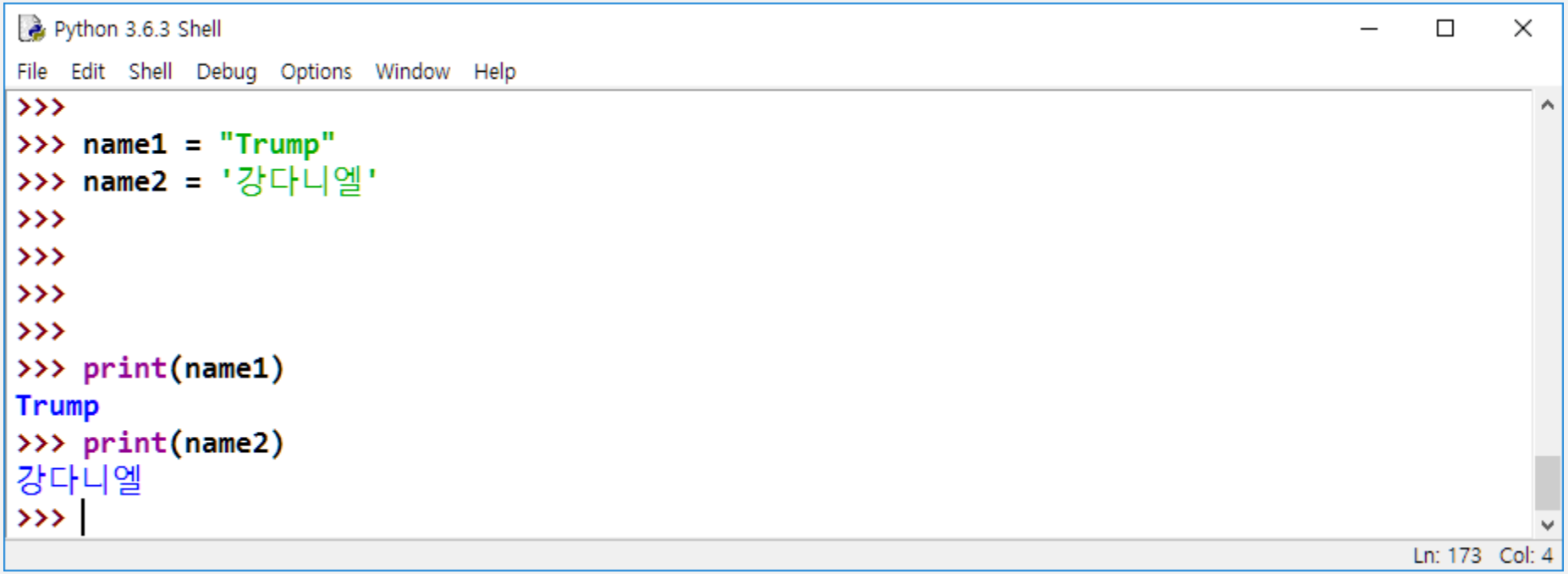
변수값을 바꿔서 사용 : r 에 11을 대입



연산 기호

연산자	연산
+	덧셈
-	뺄셈
*	곱셈
/	나눗셈
**	제곱
%	나머지

변수에는 문자열(string)을 담을 수 있다.



문자열은 문자들의 집함임. 여러 개의 문자들을 나열한 것. 큰따옴표 또는 작은 따옴표로 감쌈. 한글, 영어 상관없음.

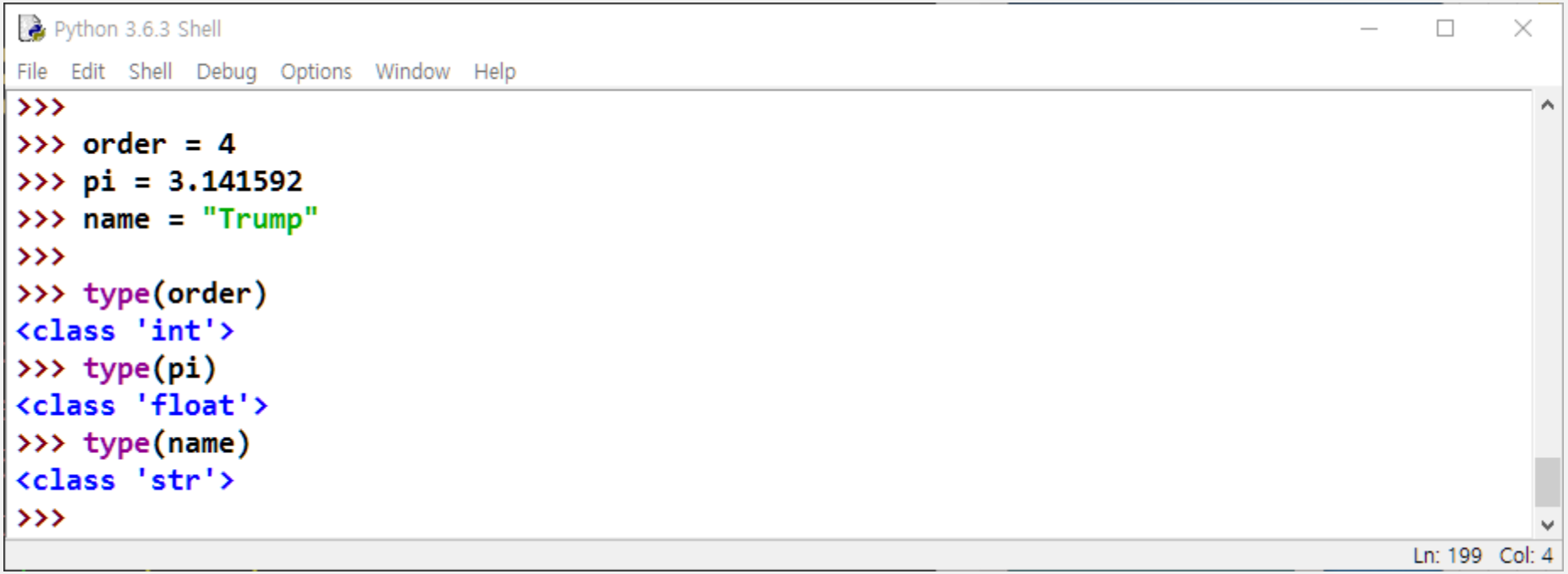
자료형(Type)

- 변수에는 다양한 종류의 정보를 담을 수 있음.
- 어떤 변수의 자료형을 알고 싶으면, type(변수명) 함수를 이용함.

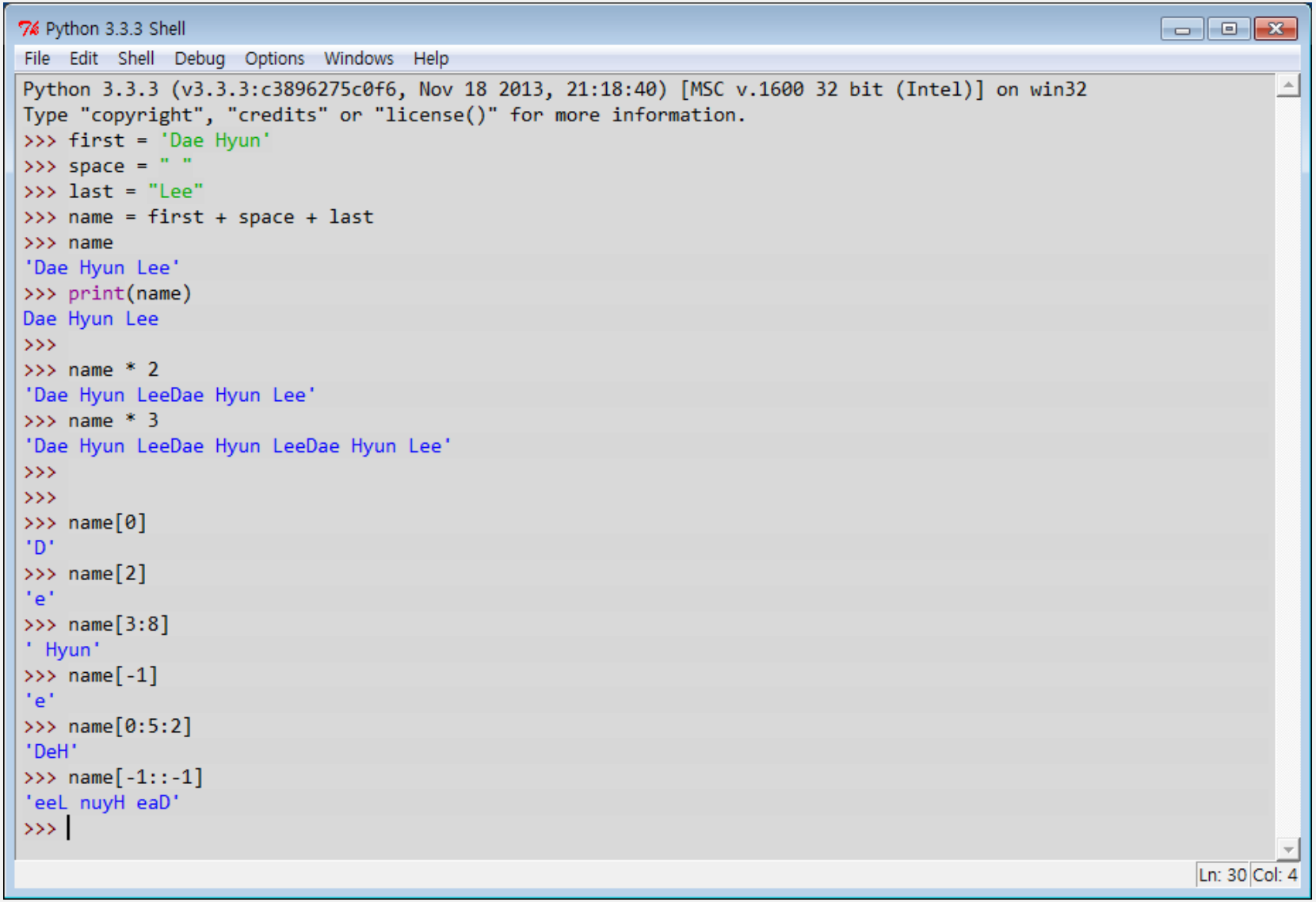
```
order = 4
```
정수형 int

```
pi = 3.141592
```
실수형 float

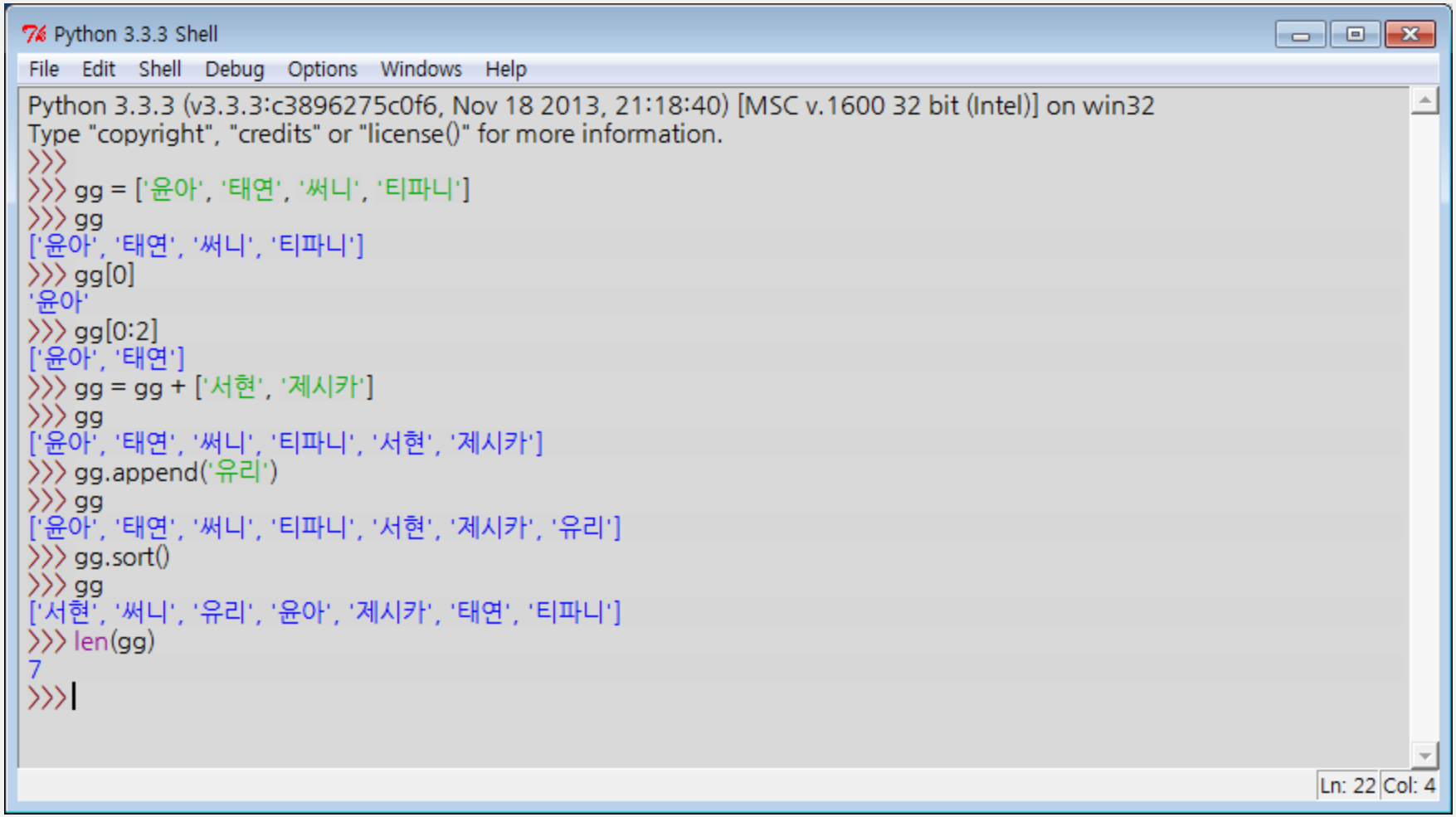
```
name = “Trump”
```
문자열형 str



Str



List



Dictionary

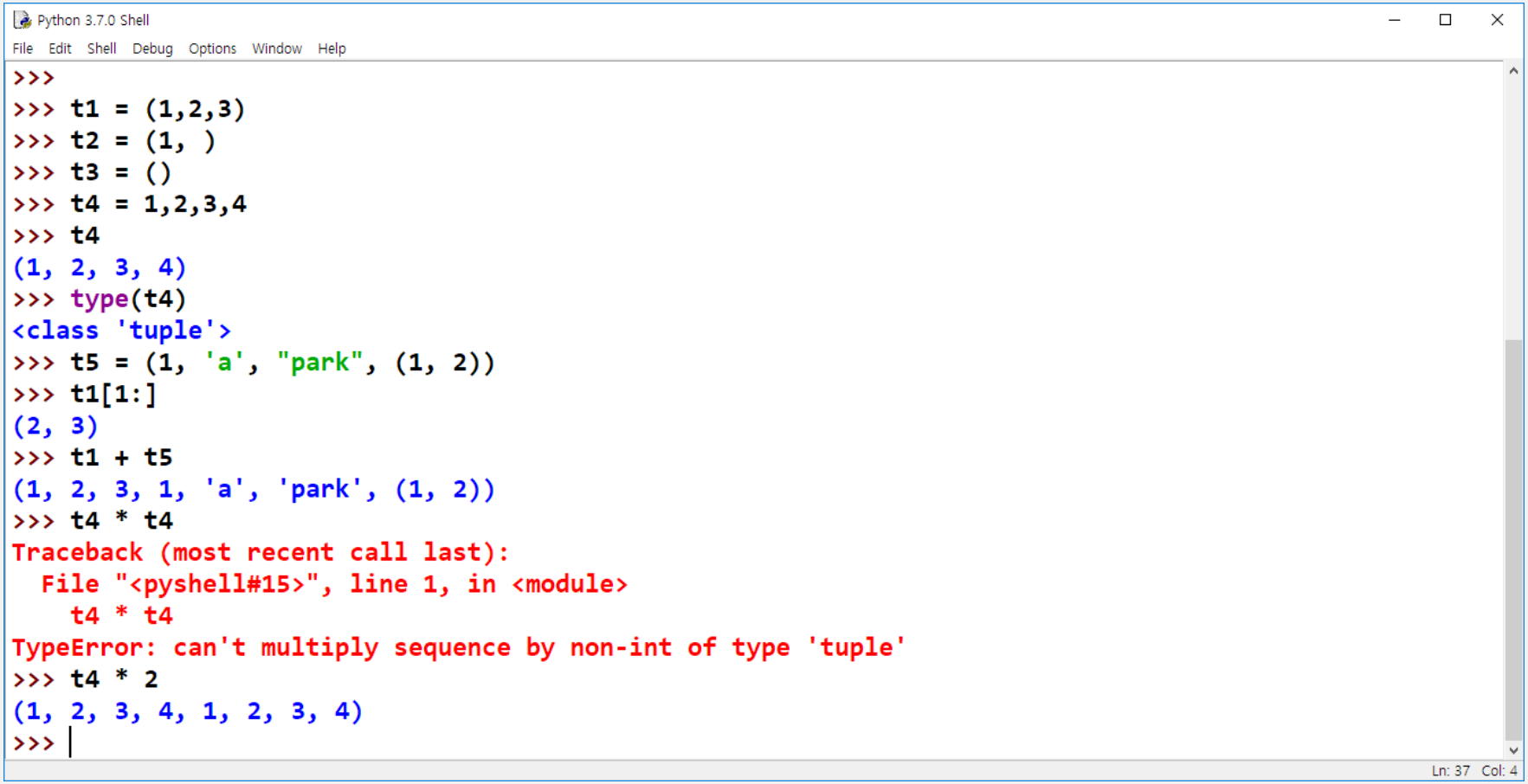
```
>>> phone = {'이대현': '01097881745', '문재인' : '01012345678', '이재용' : '
    01077774949'}

>>> print(phone)
{'이대현' : '01097881745', '이재용': '01112341234', '문재인': '01012345678'}

>>> print(phone['이대현'])
'01097881745'

>>> print(phone.keys())
Dict_keys(['이대현', '이재용', '문재인'])

>>> print(phone.values())
dict_values(['01097881745', '01112341234', '01012345678'])
```

해당 Key가 있는지 조사 (in)

```
>>> a = {'name':'pey', 'phone':'0119993323', 'birth': '1118'}
>>> 'name' in a
True
>>> 'email' in a
False
```

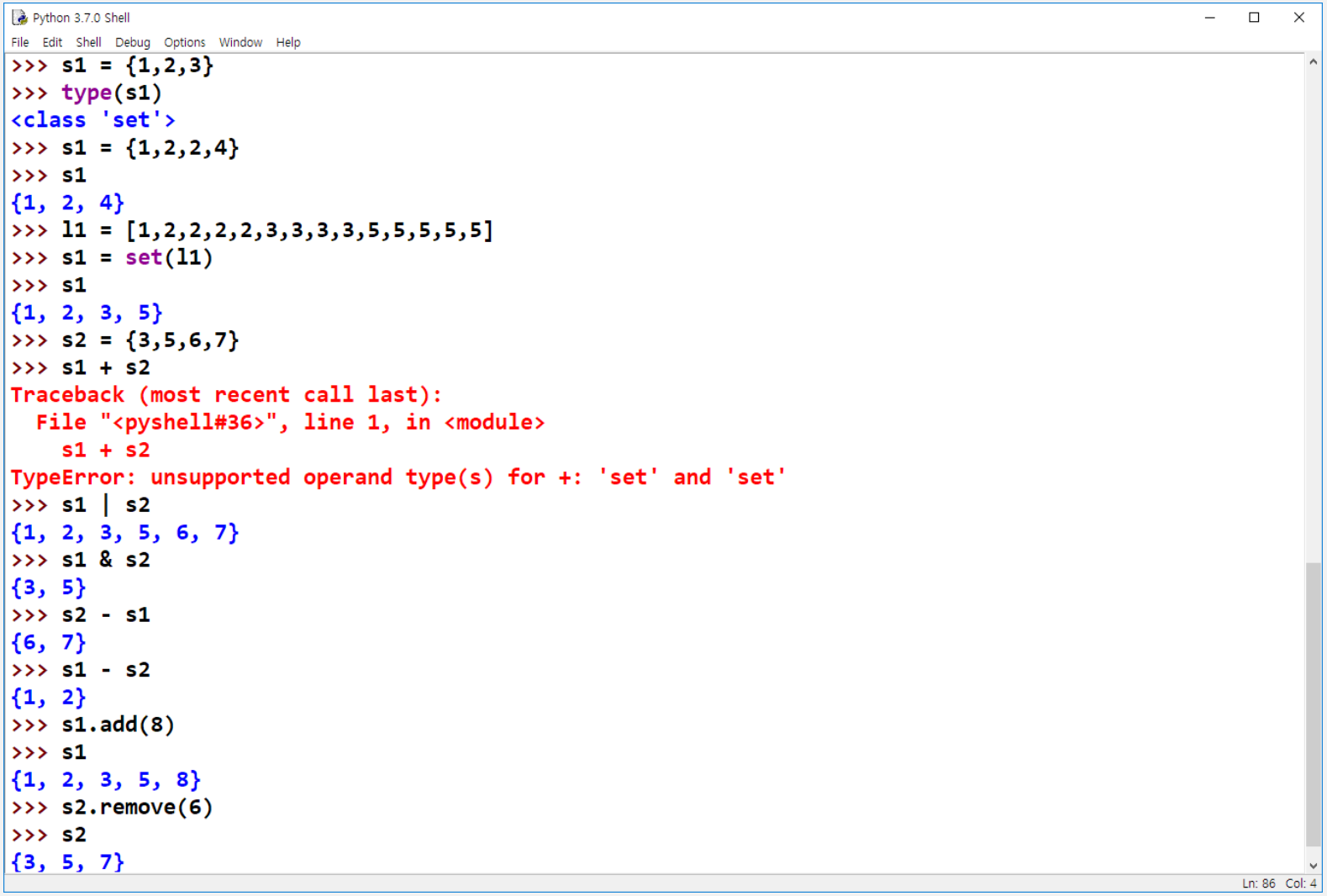
Tuple

- 여러 개의 값을 동시에 관리. 리스트와 유사.
- 하지만, 기본적으로 값을 바꿀 수는 없음. ==> 프로그램 중 변경이 되지 않는 값들의 모음이 필요할 때 사용하면 됨.



set

- 집합 자료형
- 리스트와 달리, 중복을 허용하지 않고, 순서가 없음.



Turtle 모듈

- 펜을 가지고, 화면 위를 다니면서 그림을 그림.
- 전진, 후진, 회전, 원 그리기 등 다양하게 움직이면 그림을 그릴 수 있음.

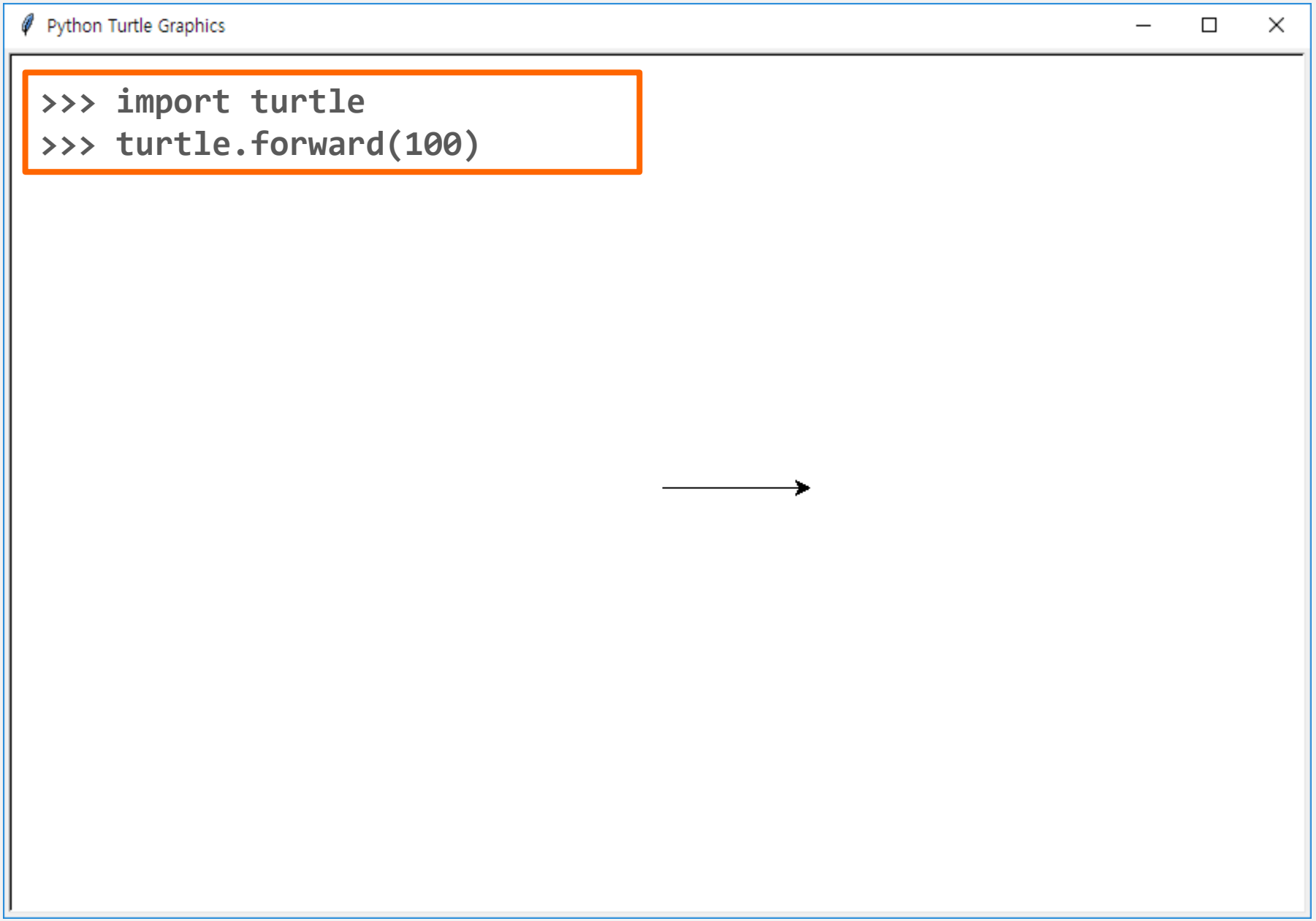
펜을 물고 있는 거북이

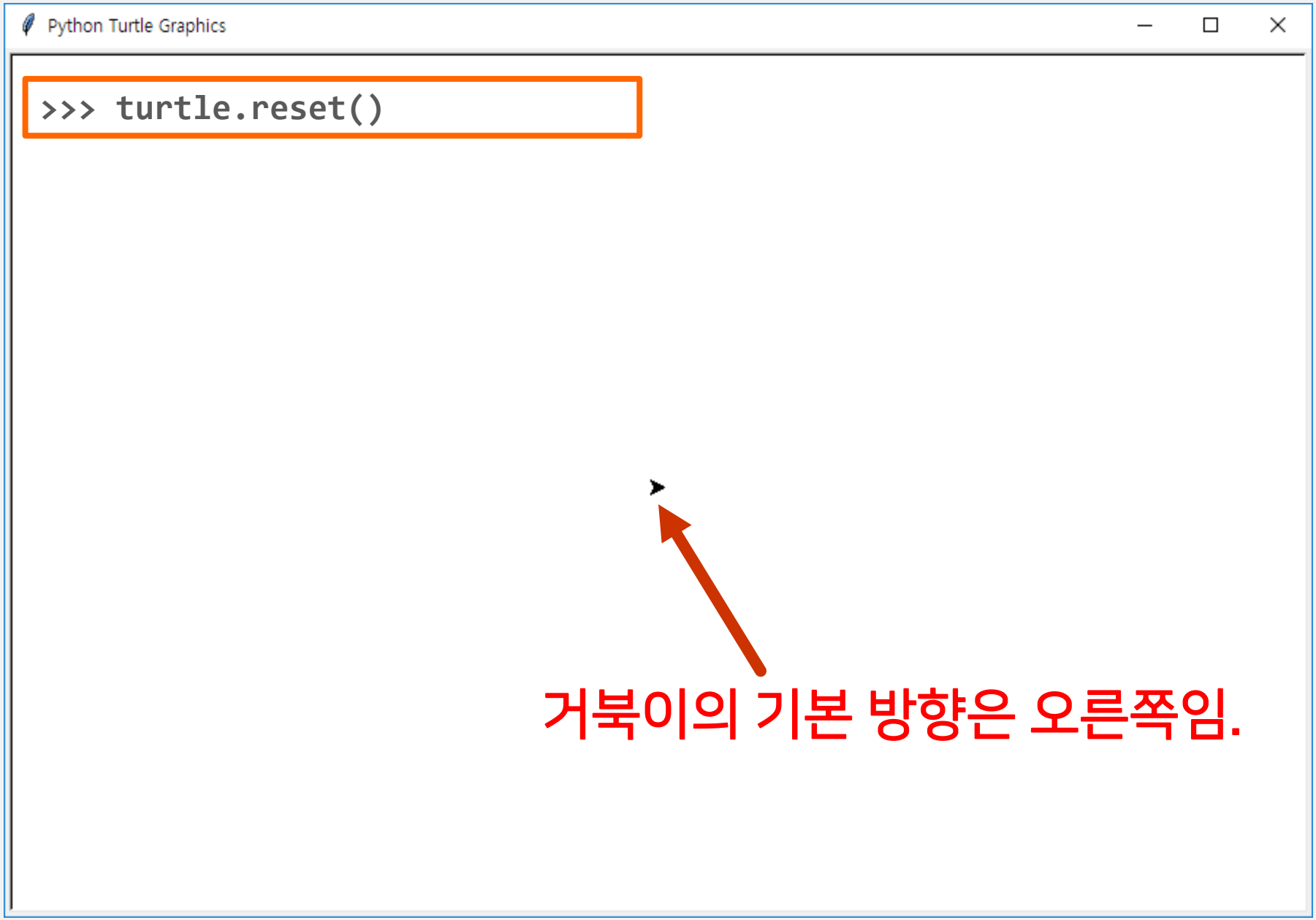
모듈의 사용 문법

모듈을 사용하기 위해 수입(import)함.

```
import turtle

turtle.forward(100)
```

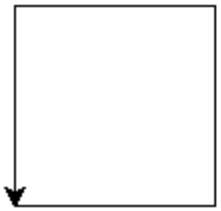
turtle 이 갖고 있는 기능(함수, function)
을 이용하여, 그림을 그린다.





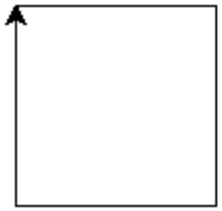
Python Turtle Graphics

```
>>> turtle.forward(100)
>>> turtle.left(90)
>>> turtle.forward(100)
>>> turtle.left(90)
>>> turtle.forward(100)
>>> turtle.left(90)
>>> turtle.forward(100)
```



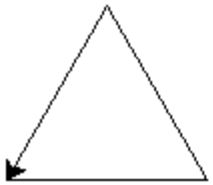
Python Turtle Graphics

```
>>> turtle.reset()
>>> turtle.forward(100)
>>> turtle.right(90)
>>> turtle.forward(100)
>>> turtle.right(90)
>>> turtle.forward(100)
>>> turtle.right(90)
>>> turtle.forward(100)
```

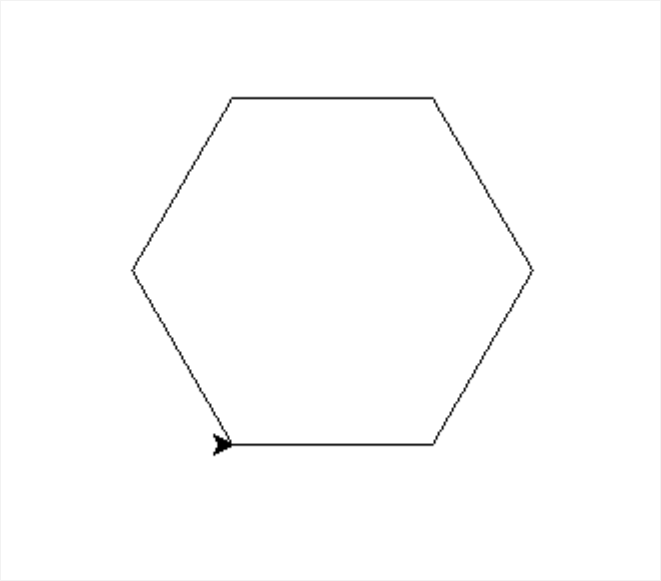


Python Turtle Graphics

```
>>> turtle.forward(100)
>>> turtle.left(120)
>>> turtle.forward(100)
>>> turtle.left(120)
>>> turtle.forward(100)
```

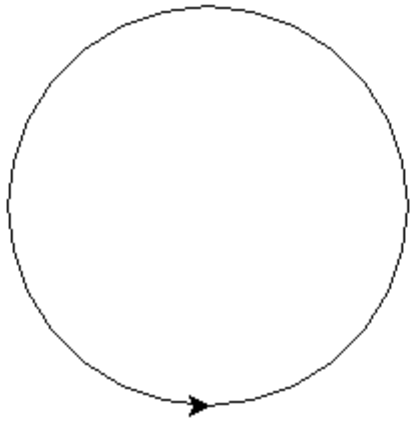


퀴즈 #1: 정육각형을 그려보자 !



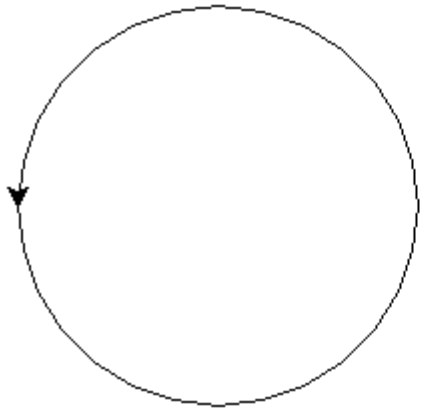
Python Turtle Graphics

```
>>> turtle.circle(100)
```



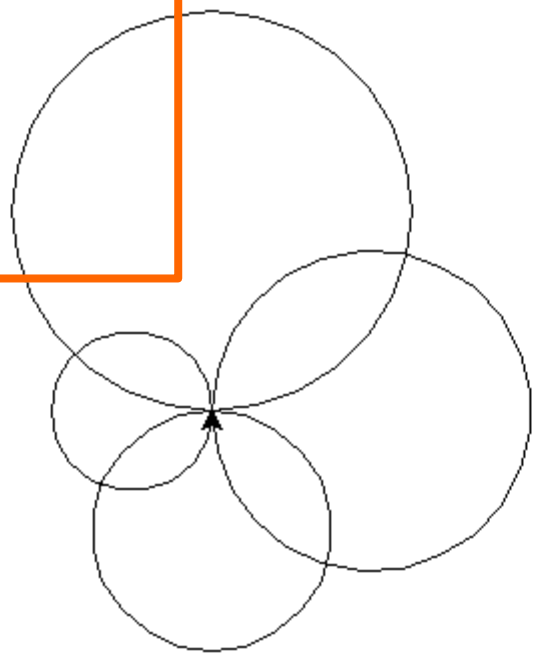
Python Turtle Graphics

```
>>> turtle.right(90)
>>> turtle.circle(100)
```



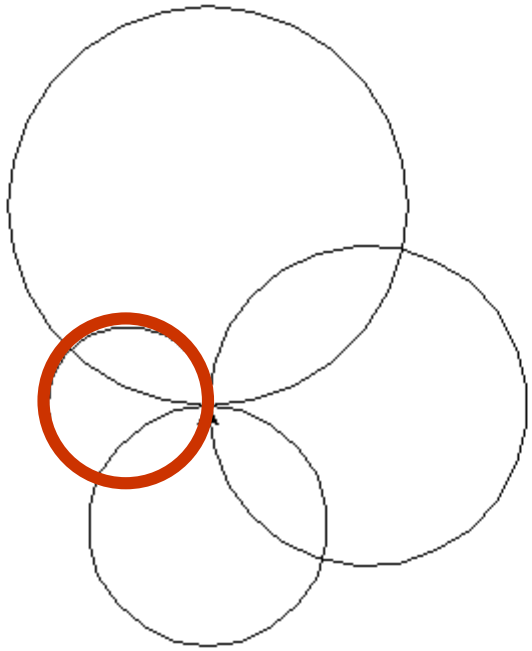
Python Turtle Graphics

```
>>> turtle.circle(100)
>>> turtle.right(90)
>>> turtle.circle(80)
>>> turtle.right(90)
>>> turtle.circle(60)
>>> turtle.right(90)
>>> turtle.circle(40)
```



Python Turtle Graphics

```
>>> turtle.updo()
>>> turtle.undo()
```



마지막에 그렸던 원이 없어짐.

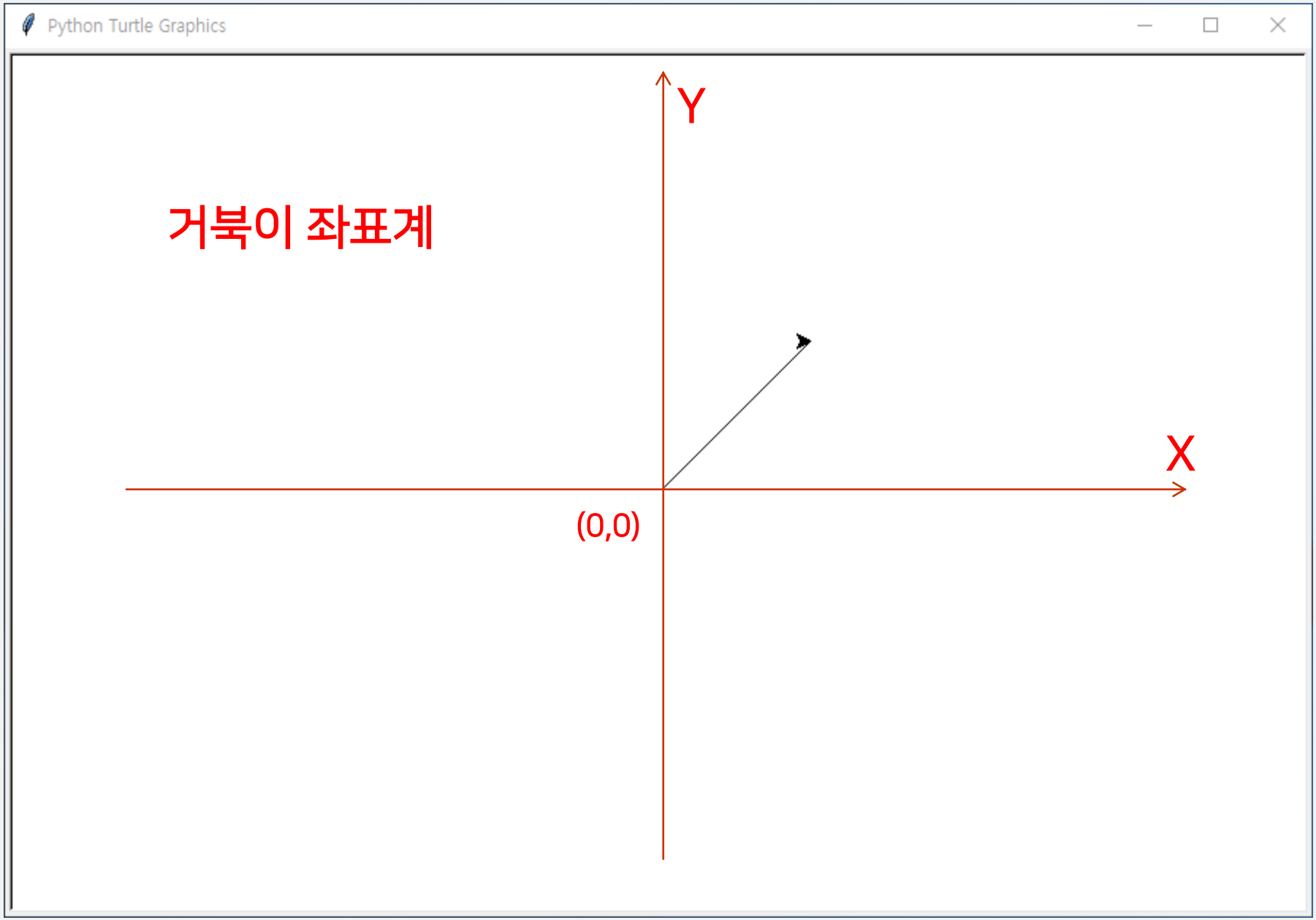
이전 상태로 되돌아감.

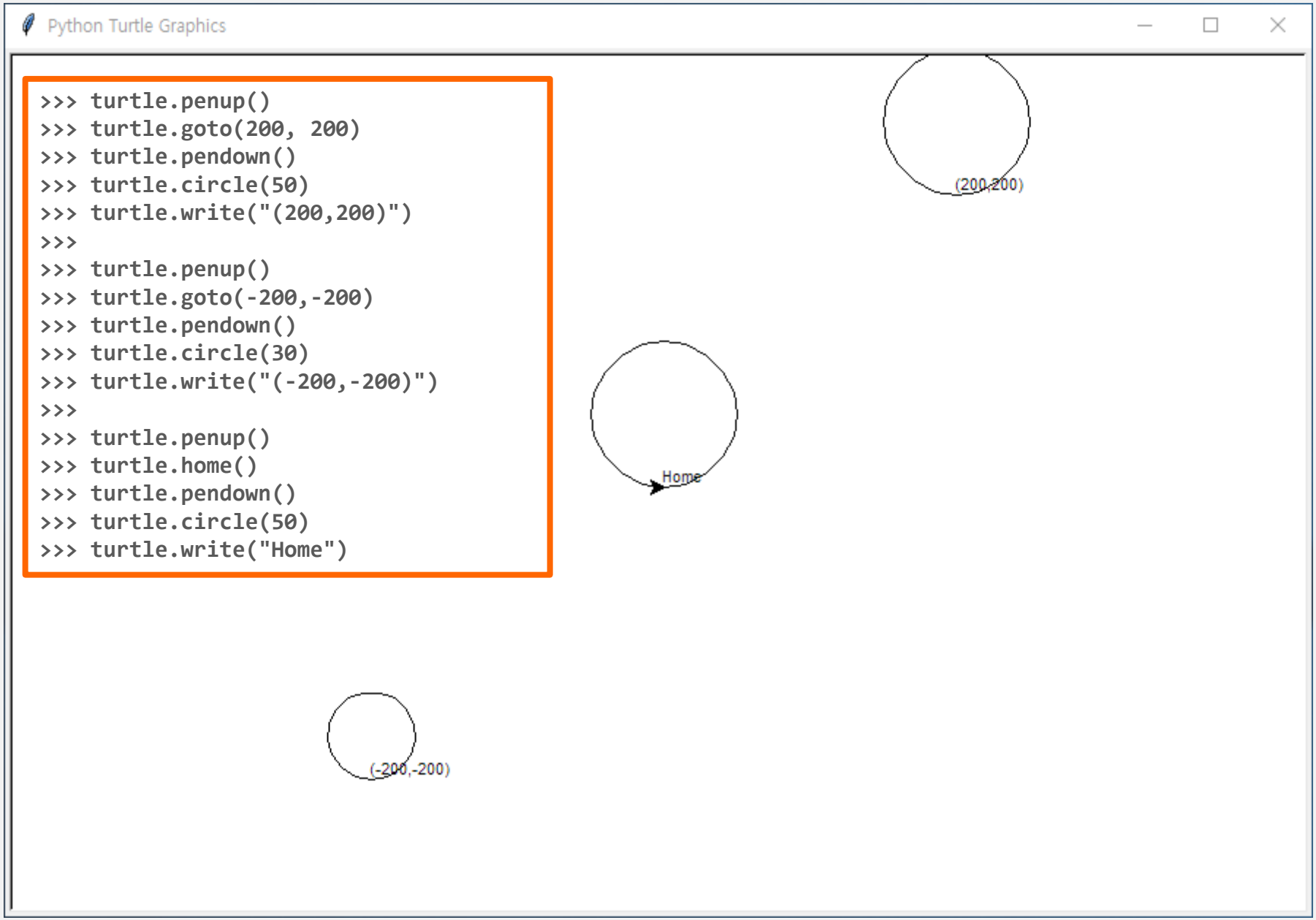
Python Turtle Graphics

```
>>> turtle.reset()
>>> turtle.goto(100, 100)
```

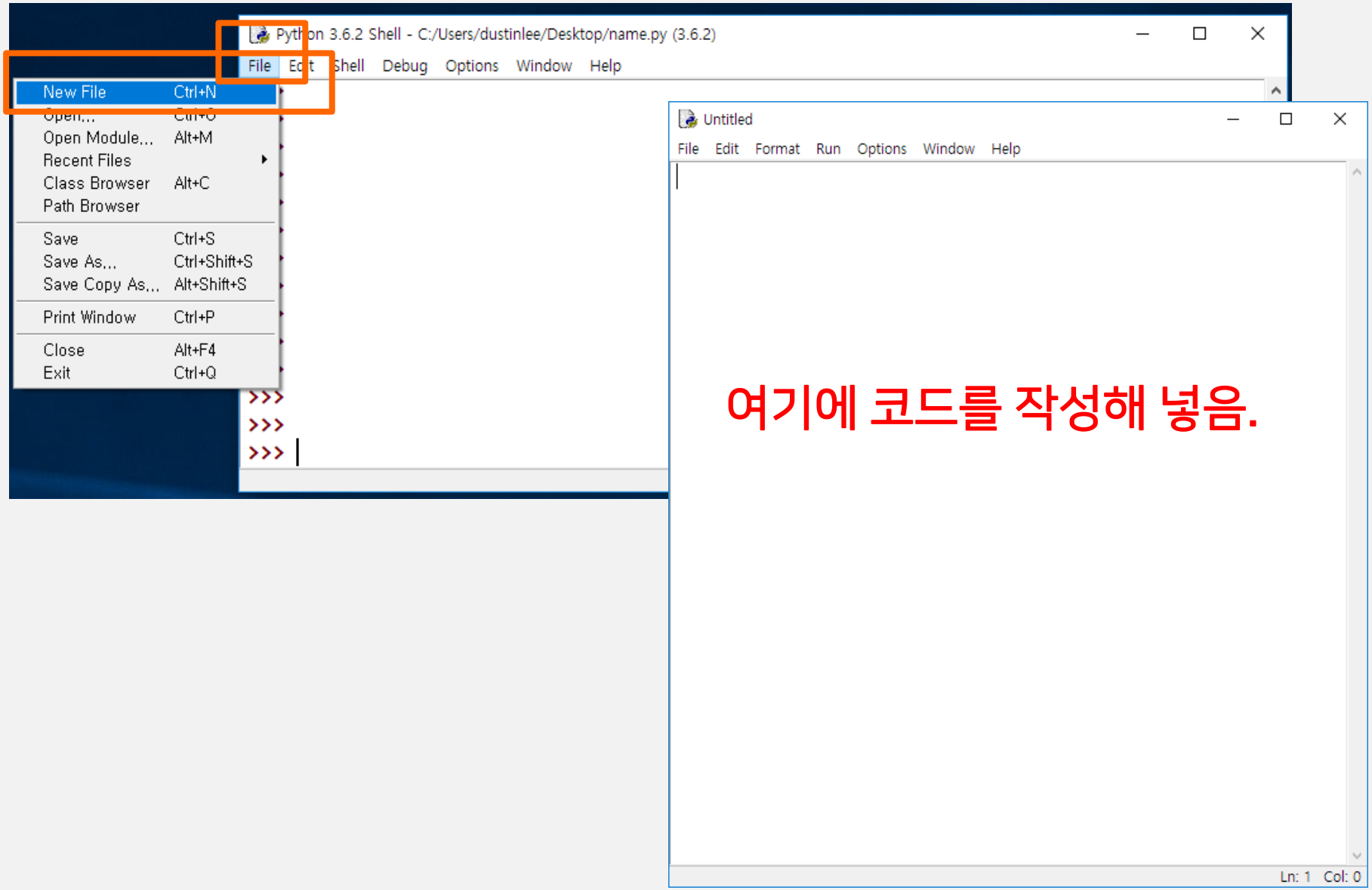
(100,100) 으로 이동한다.

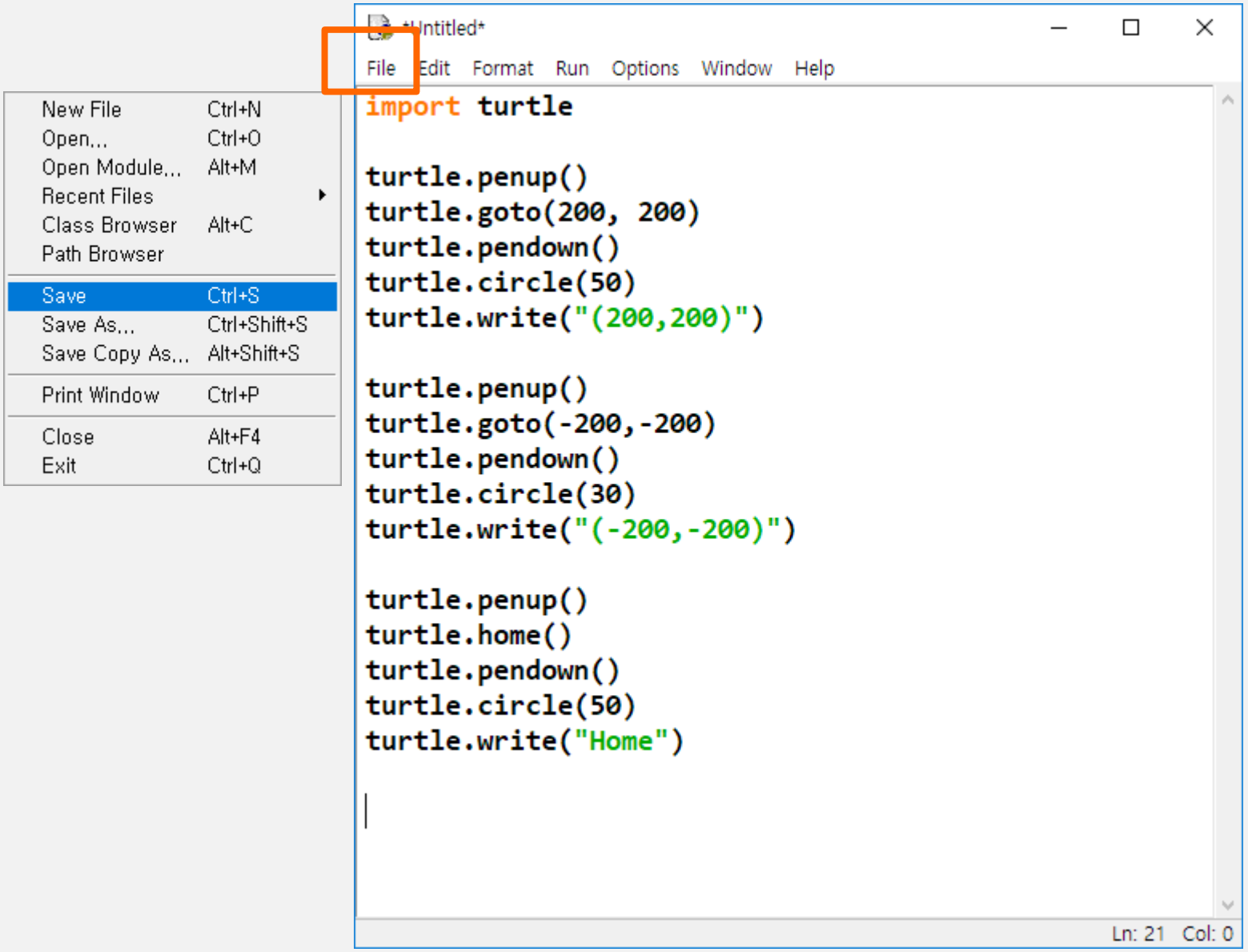
거북이의 머리방향은 변함없이 여전히 오른쪽 방향.

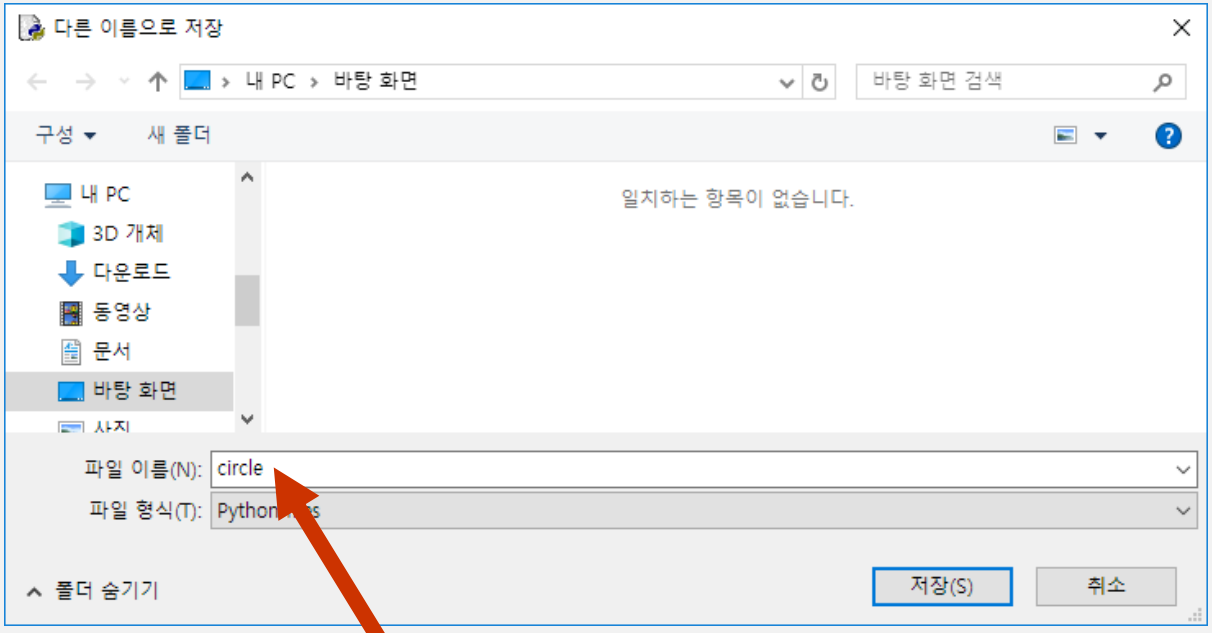




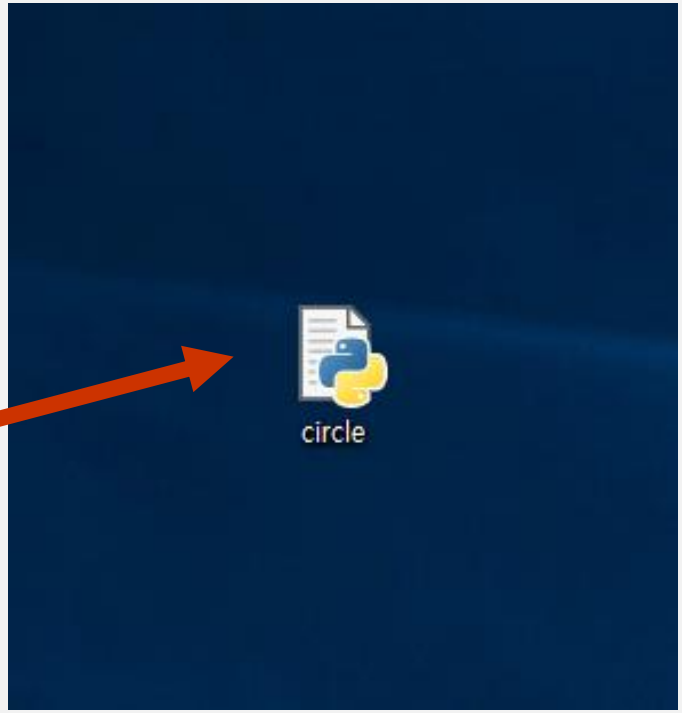
프로그램을 파일로 만들어서 저장







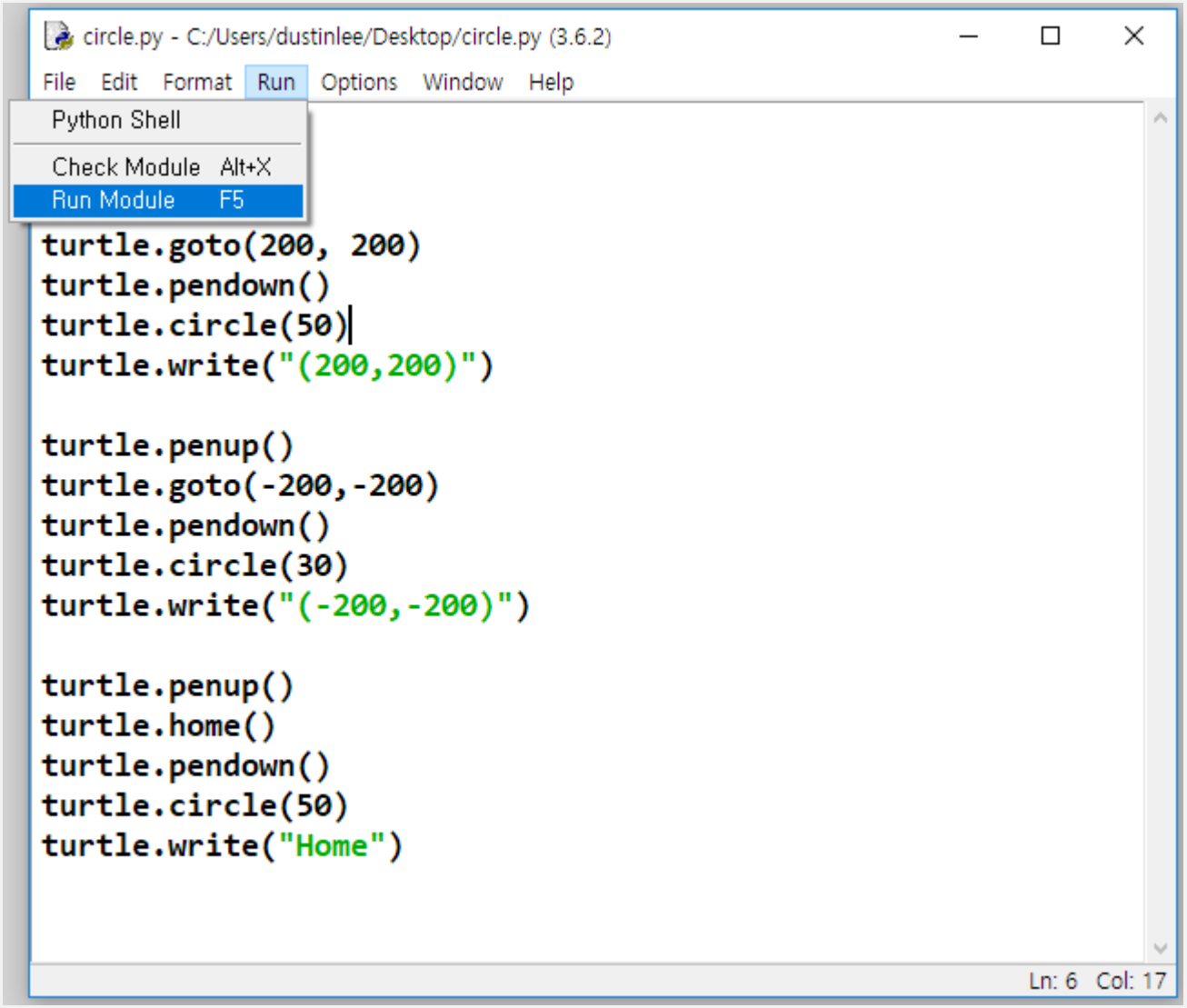
circle 이라는 이름으로 바탕 화면에 저장.



바탕화면에 circle.py 라는 이름의 파일이 생성됨.

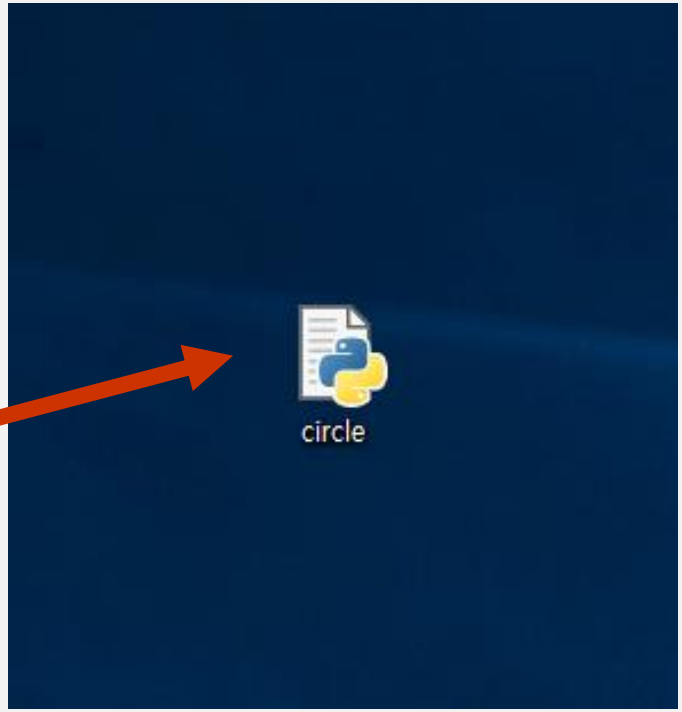
프로그램 실행 방법 #1

- Run→Run Module 을 클릭 또는 단축기 F5

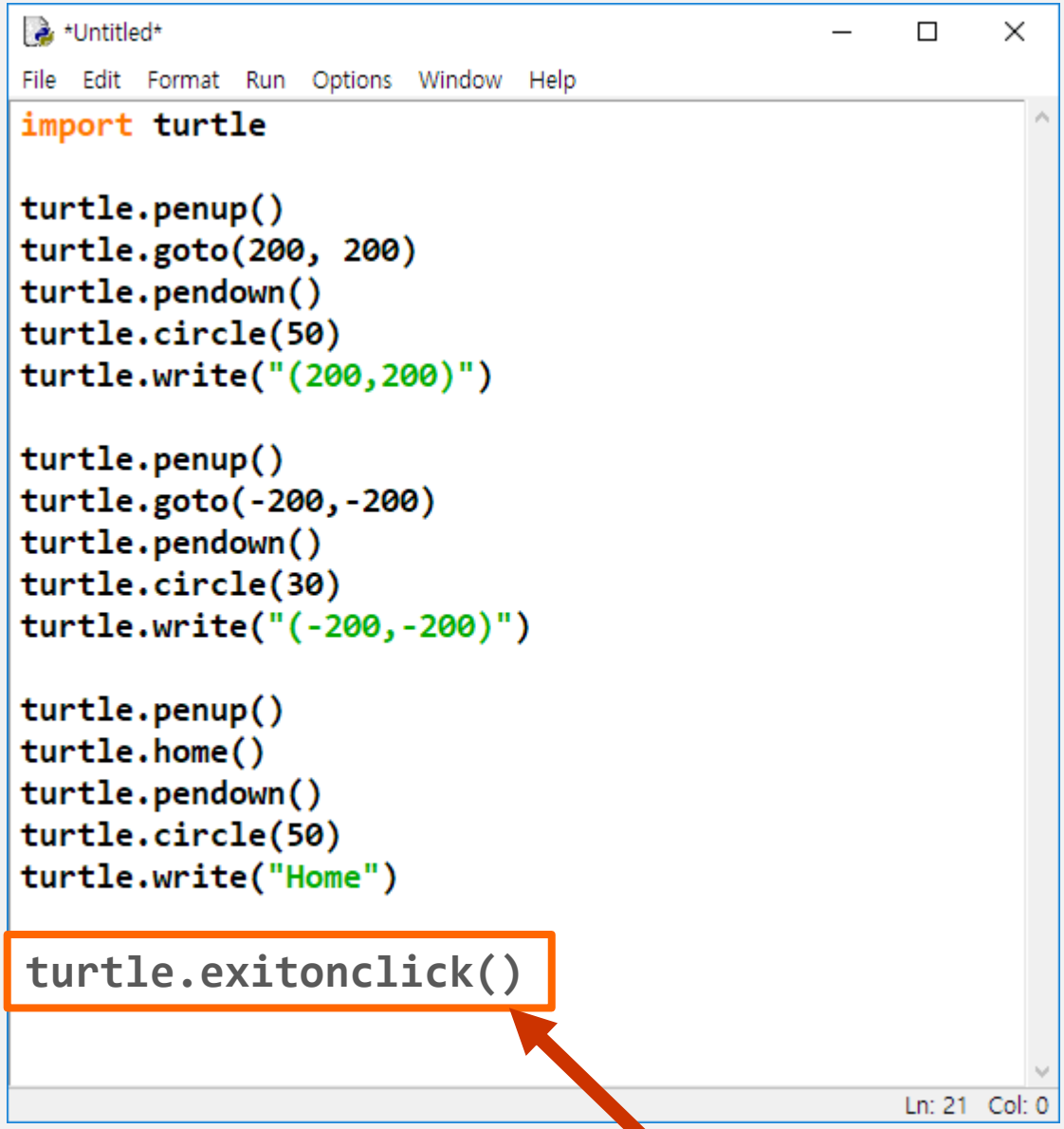


프로그램 실행 방법 #2

- 프로그램 파일을 더블 클릭하여 실행.
- 문제점은?



circle.py 를 더블 클릭



코드 마지막 부분에 exitonclick() 추가.

실습과제 #1: 자기 이름 그리기

- 파일로 작성하여, 바탕화면에 name.py 로 저장하고, 더블클릭해서 실행!
- Tuple 을 이용할 것.