

# Lecture #2. 파이썬 기초 (2)

2D 게임 프로그래밍

이대현 교수



한국공학대학교  
TECH UNIVERSITY OF KOREA

# Tuple

- 여러 개의 값을 동시에 관리. 리스트와 유사.
- 일단 한번 만들어지면 그 이후에는 값을 바꿀 수는 없음. ==> 프로그램 중 변경이 되지 않는 값들의 모음이 필요할 때 사용하면 됨.

```
>>> t1 = (1,2,3,4,5,6,7,8,9,10)
>>> type(t1)
<class 'tuple'>
>>> t2 = (1, )
>>> t3 = ()
>>> t4 = 1,2,3,4
>>> t5 = 1,
>>> t6 = (1, 'a', 'park', 3.14, [1,2,3], (4,5,6))
>>>
>>> t1[::-1]
(10, 9, 8, 7, 6, 5, 4, 3, 2, 1)
>>> t1 + t6
(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 1, 'a', 'park', 3.14, [1, 2, 3], (4, 5, 6))
>>> t1 * 2
(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
```

# set

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

```
>>> s1 = {1,2,3,4,5}
>>> type(s1)
<class 'set'>
>>> s2 = {1,1,2,2,3,3,4,4,5,5}
>>> s2
{1, 2, 3, 4, 5}
>>> l1 = [1,2,2,2,2,3,3,3,5,5,5]
>>> s3 = set(l1)
>>> s3
{1, 2, 3, 5}
```

```
>>> s4 = {3,4,5,6,7,8,9}
>>> s1 + s4
Traceback (most recent call last):
  File "<input>", line 1, in <module>
TypeError: unsupported operand type(s) for +: 'set' and 'set'
>>> s1 | s4
{1, 2, 3, 4, 5, 6, 7, 8, 9}
>>> s1 & s4
{3, 4, 5}
>>> s1 - s4
{1, 2}
>>> s4 - s1
{6, 7, 8, 9}
>>> s1.add(8)
>>> s1
{1, 2, 3, 4, 5, 8}
>>> s4.remove(9)
>>> s4
{3, 4, 5, 6, 7, 8}
```

# Complex Data Type

## ▪ List – list

- 순서가 있는, 중복을 허용하는 데이터들의 집합.
- 원하는 데이터를 찾기 위해, 순서 index 를 이용.

[ val1, val2, ... ]

## ▪ Dictionary – dict

- 검색을 위한 키를 갖는 데이터들의 집합
- key – value 쌍 들의 집합

{ key1: val1, key2: val2, ... }

## ▪ Tuple – tuple

- 순서가 있는, 중복을 허용하는 데이터들의 집합
- 다만, 데이터값을 변경하는 것은 불가

( val1, val2, ... )

## ▪ Set – set

- 중복을 허용하지 않는, 순서에 상관없는 데이터들의 집합

{ val1, val2, ... }

# Turtle 모듈

---

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



펜을 물고 있는 거북이

# 모듈의 사용 문법


---

모듈을 사용하기 위해 수입(import)함.

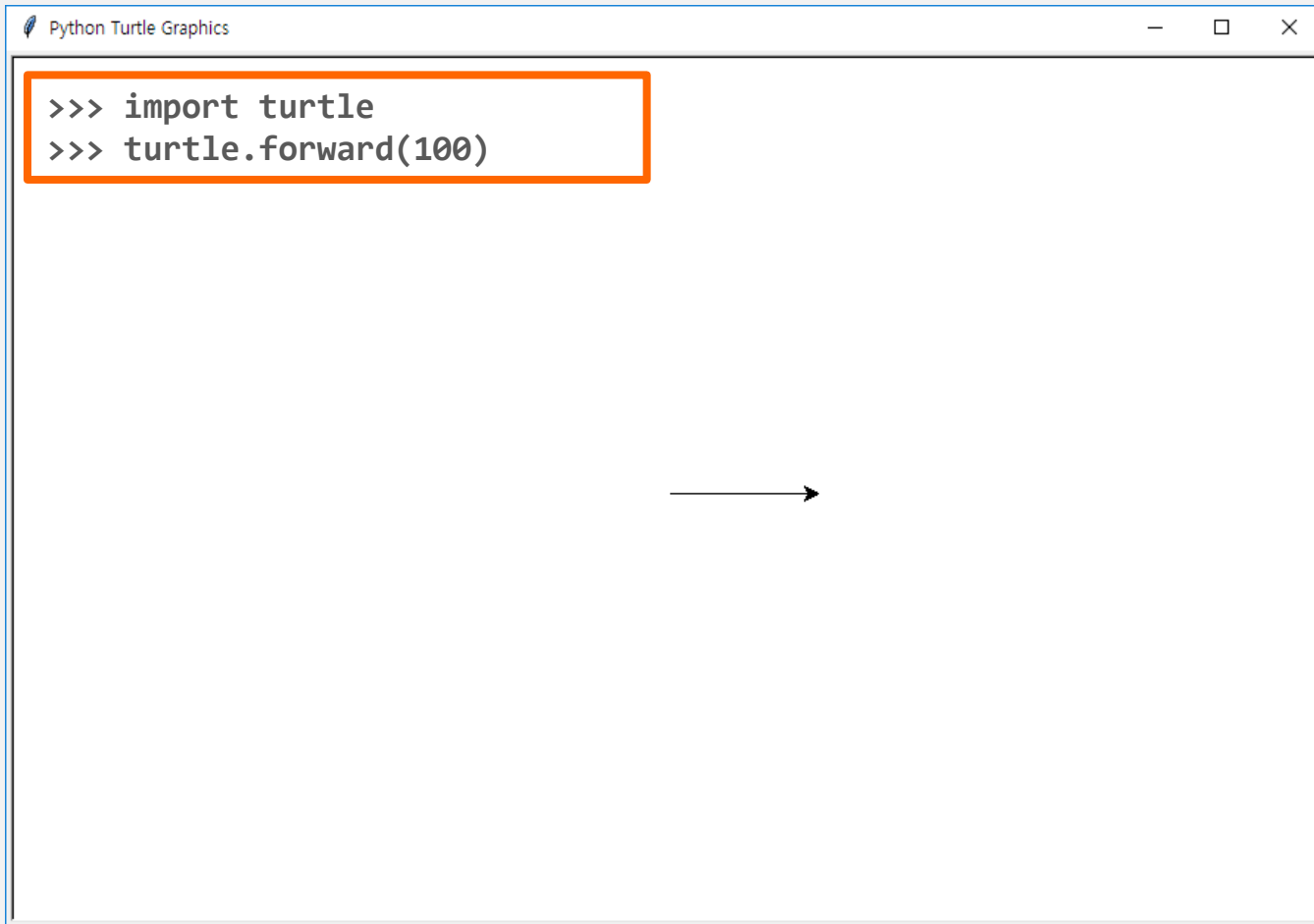


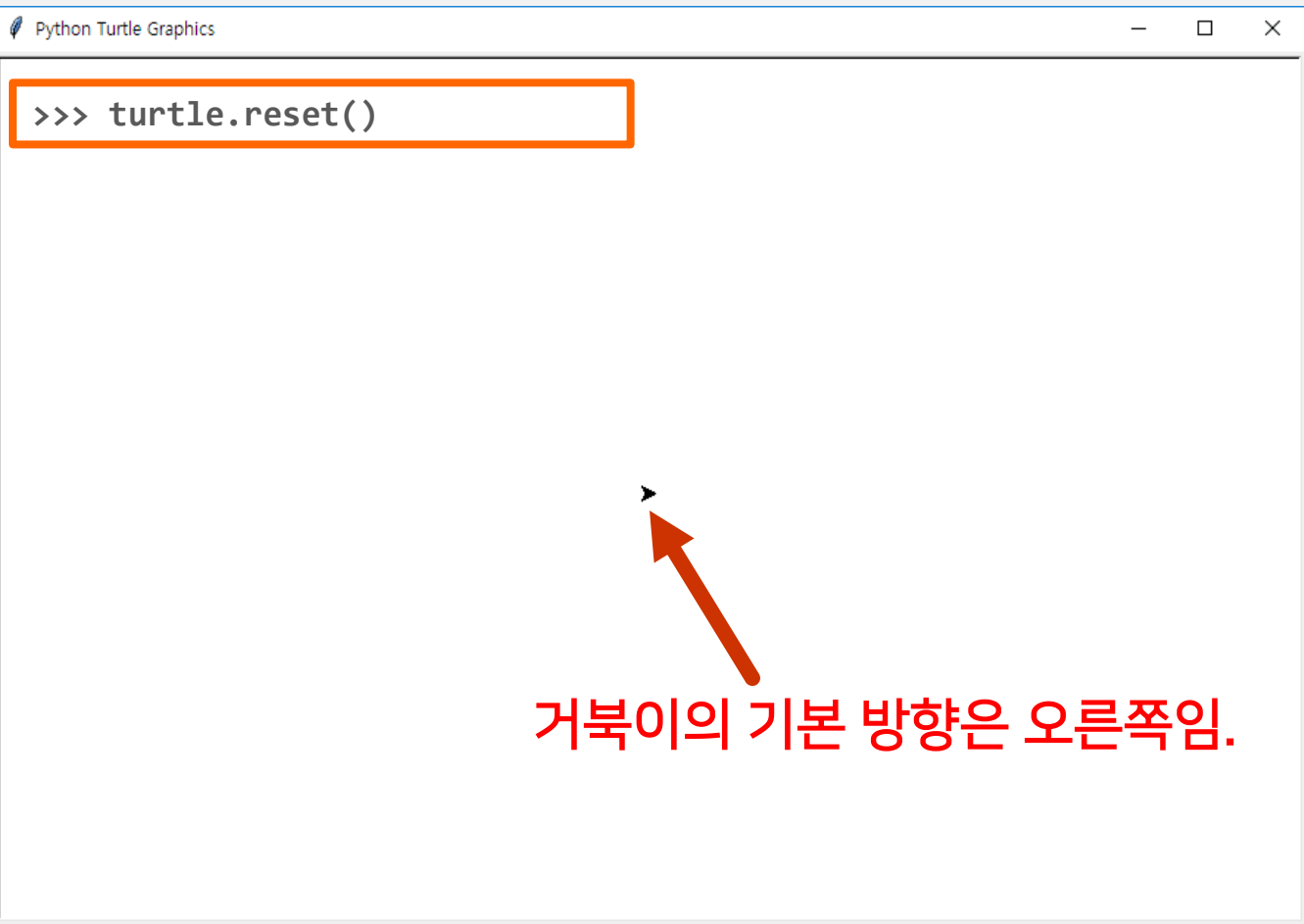
```
import turtle
```

```
turtle.forward(100)
```

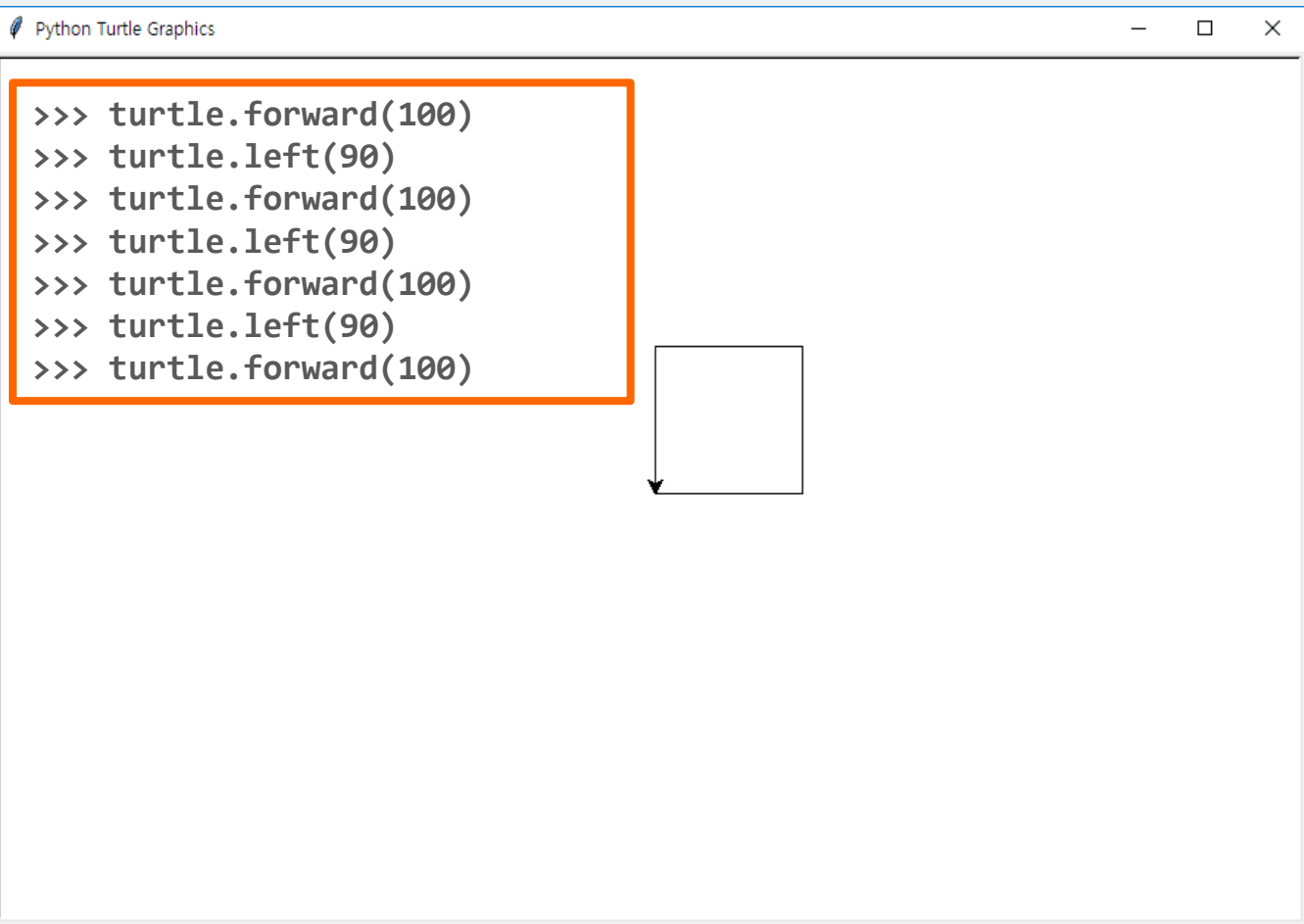


turtle 이 갖고 있는 기능(함수, function)  
을 이용하여, 그림을 그린다.

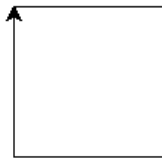








```
>>> turtle.reset()
>>> turtle.forward(100)
>>> turtle.right(90)
>>> turtle.forward(100)
>>> turtle.right(90)
>>> turtle.forward(100)
>>> turtle.right(90)
>>> turtle.forward(100)
```

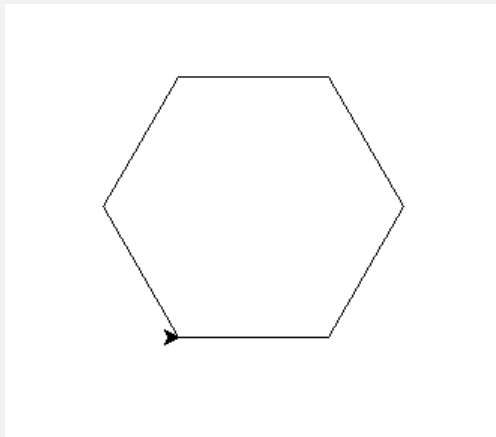


```
>>> turtle.forward(100)  
>>> turtle.left(120)  
>>> turtle.forward(100)  
>>> turtle.left(120)  
>>> turtle.forward(100)
```

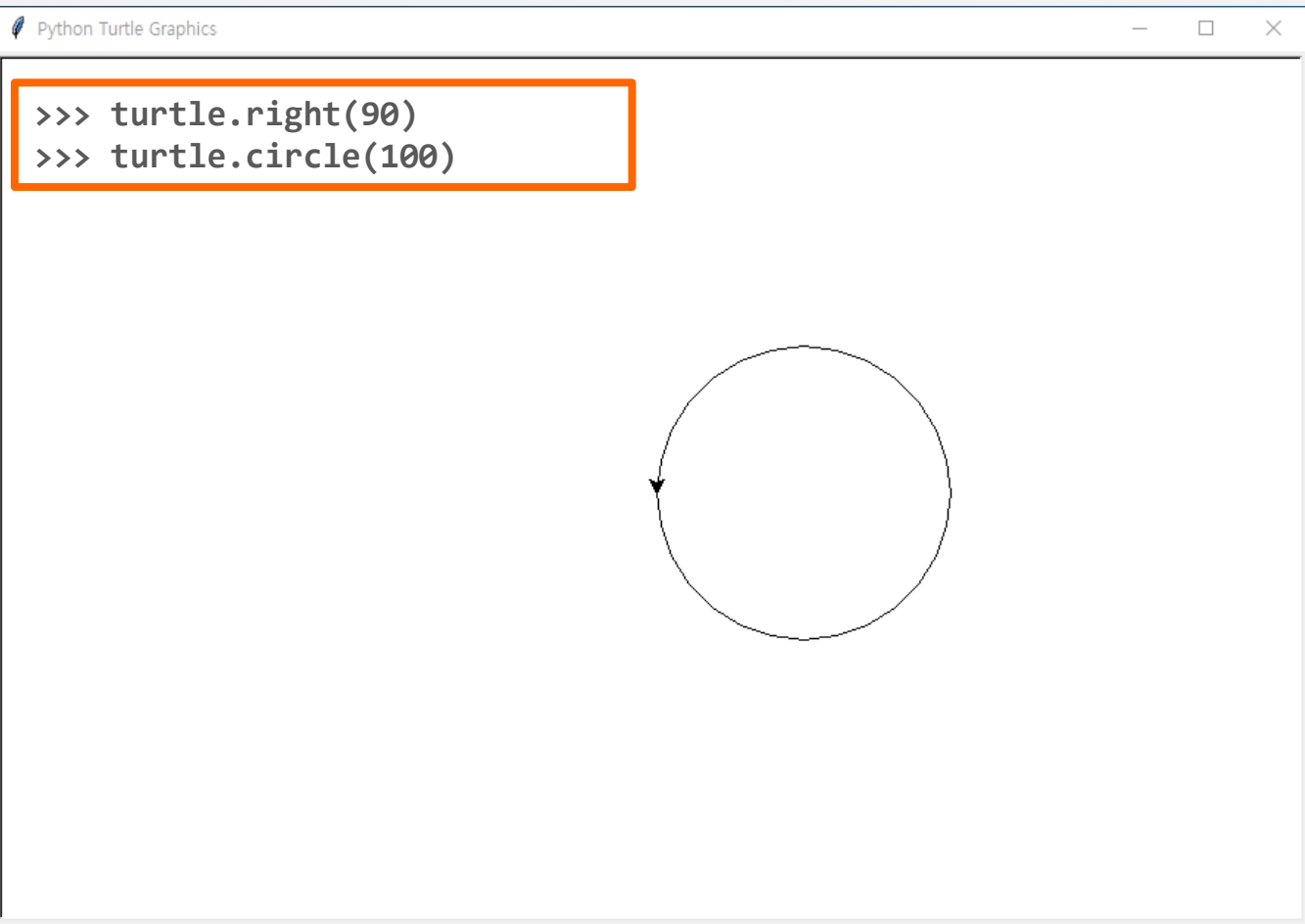


# 퀴즈 #1: 정육각형을 그려보자!

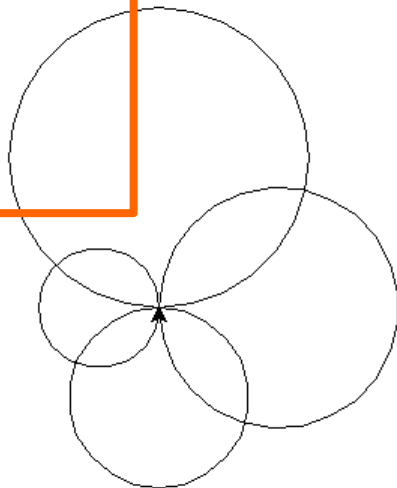
---



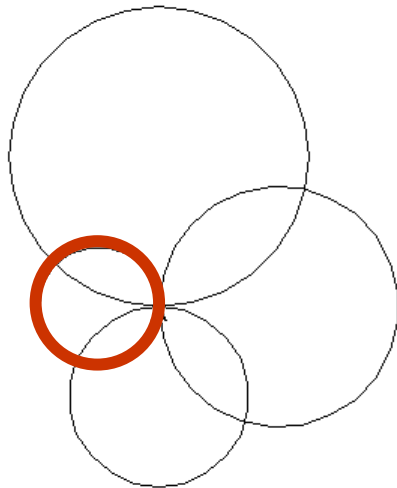




```
>>> turtle.circle(100)
>>> turtle.right(90)
>>> turtle.circle(80)
>>> turtle.right(90)
>>> turtle.circle(60)
>>> turtle.right(90)
>>> turtle.circle(40)
```

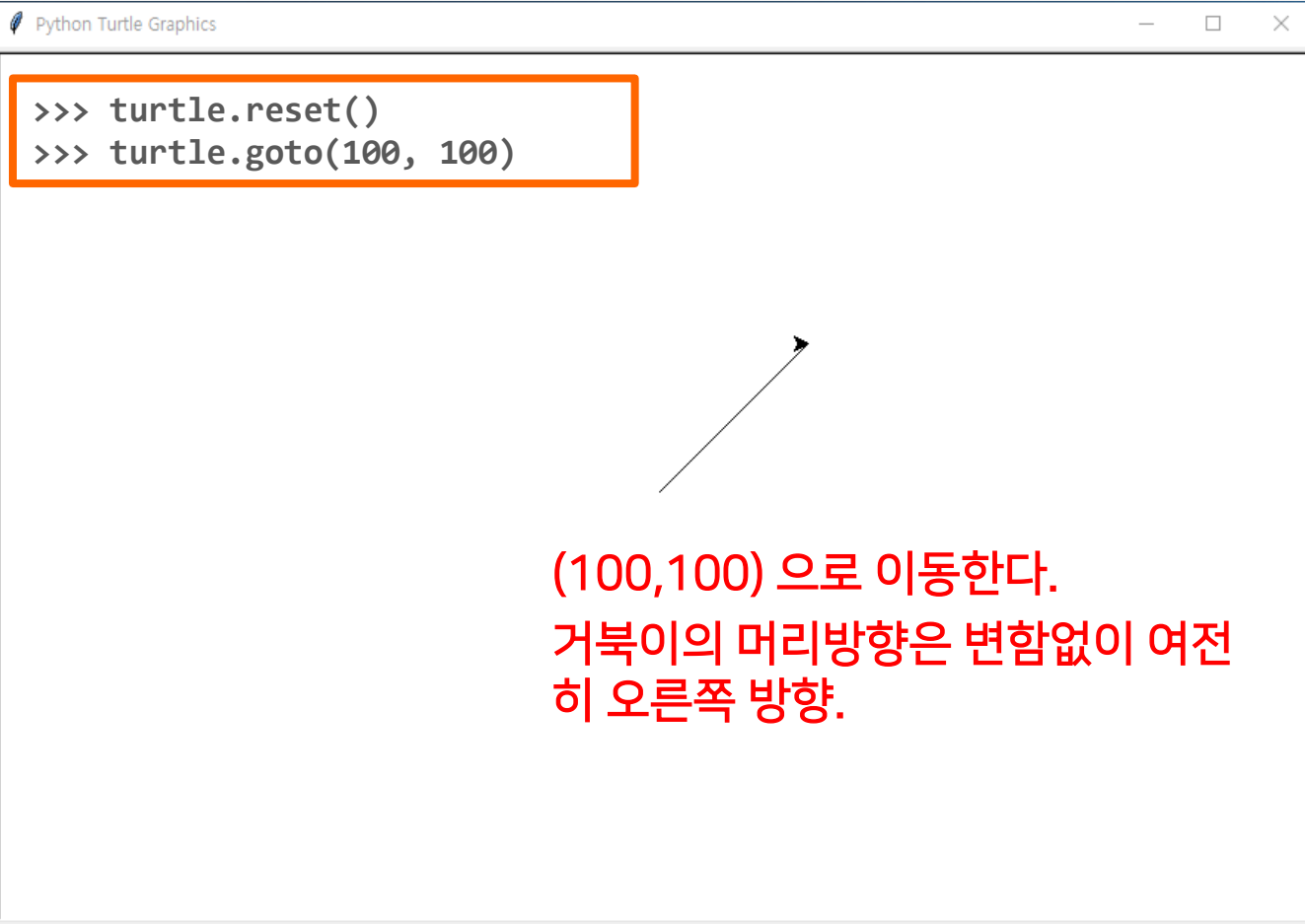


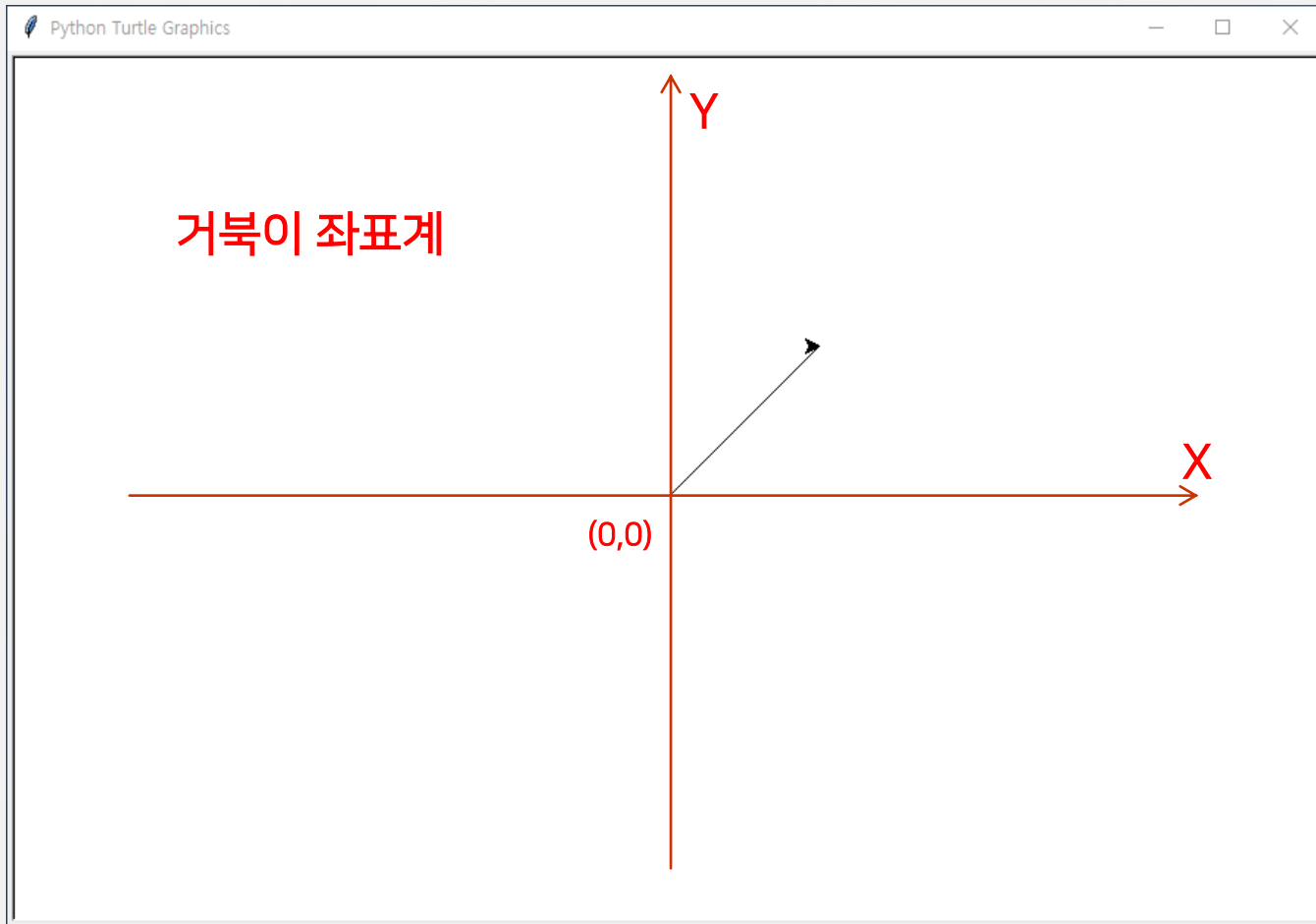
```
>>> turtle.updo()  
>>> turtle.undo()
```



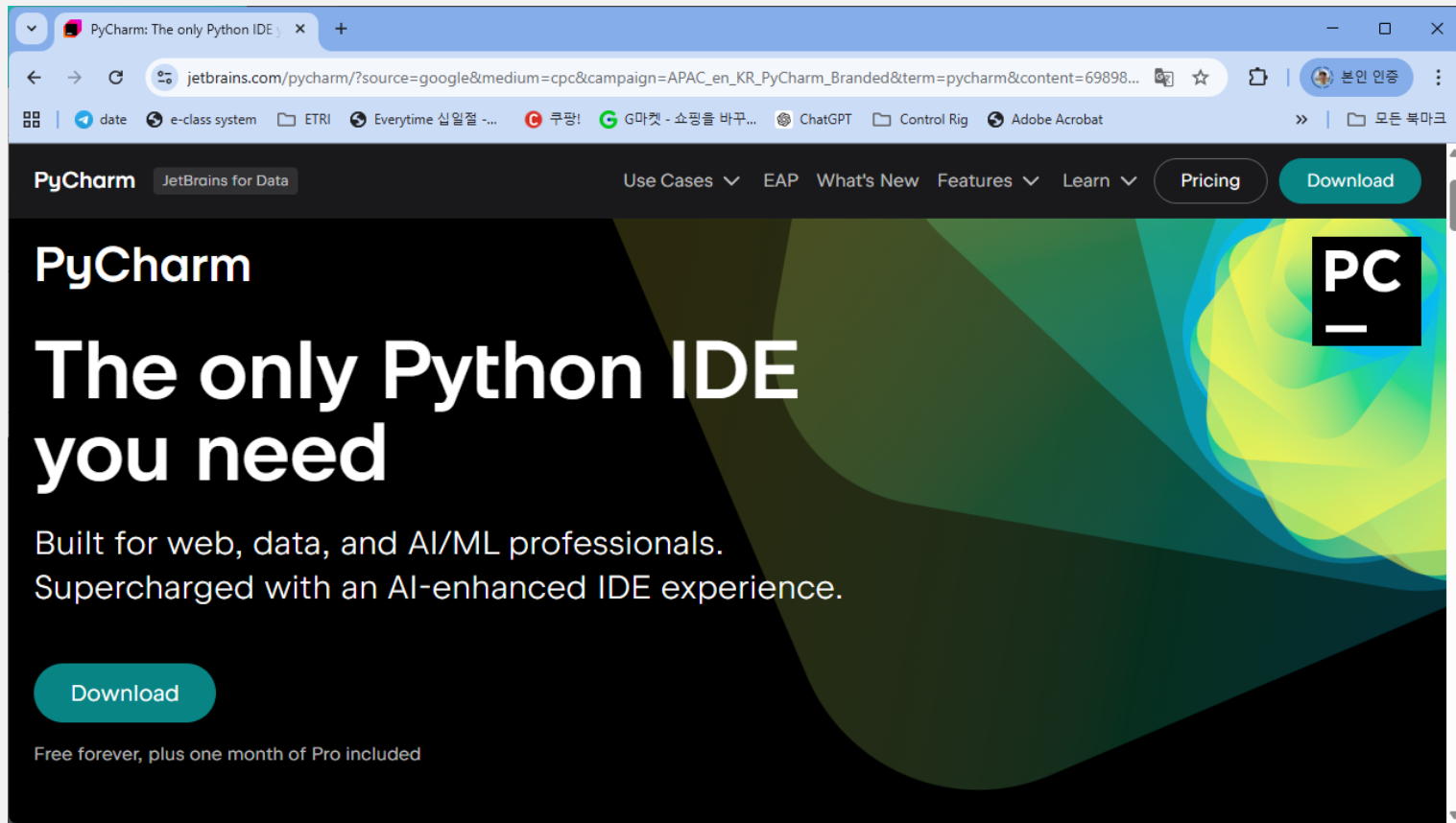
마지막에 그렸던 원이 없어짐.  
이전 상태로 되돌아감.

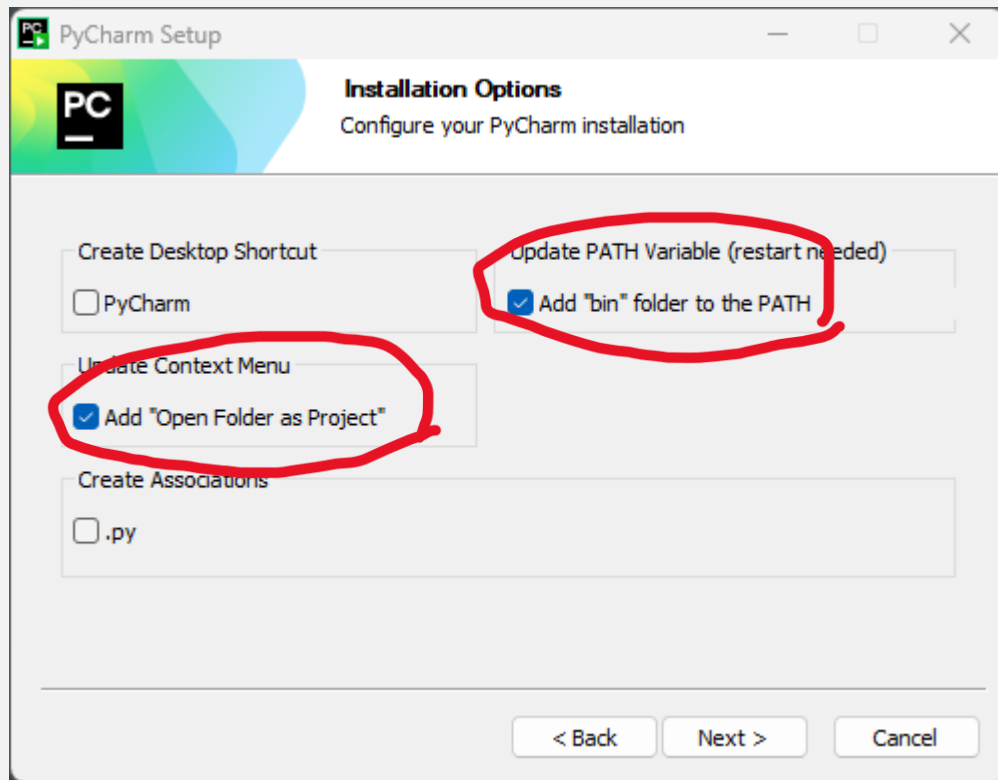




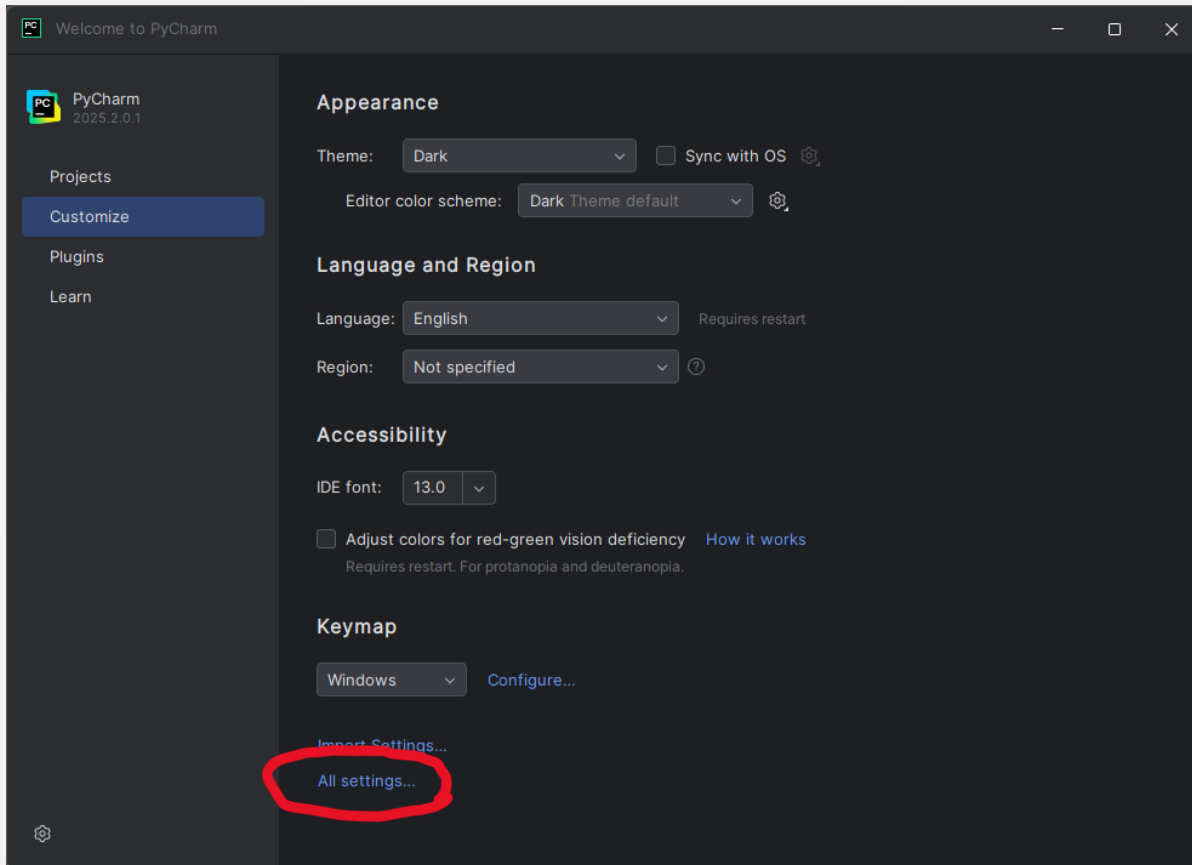


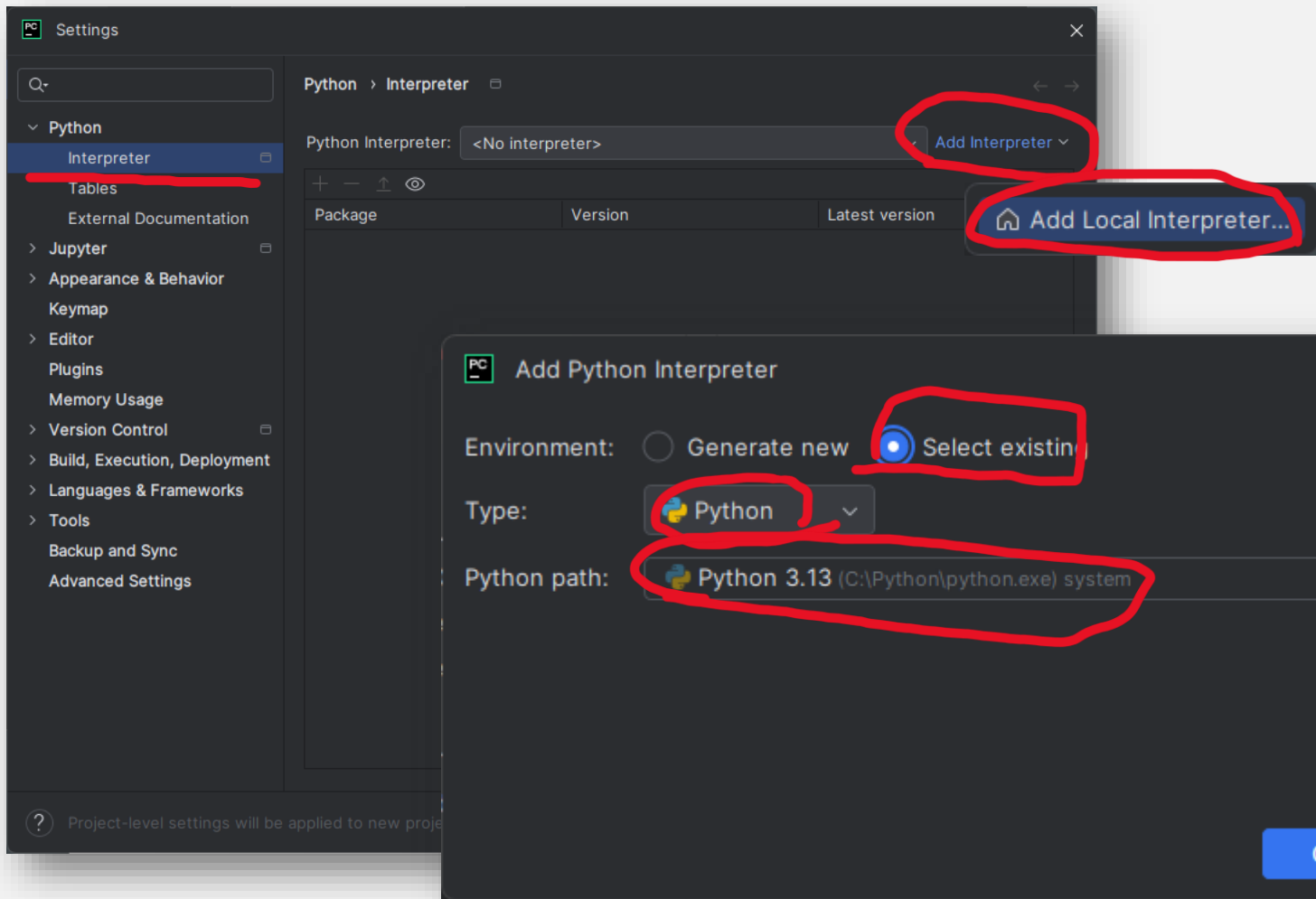
# PyCharm (<https://www.jetbrains.com/pycharm/>)





# 기본 설정 – System Interpreter





```
import turtle

turtle.penup()
turtle.goto(200,200)
turtle.pendown()
turtle.circle(50)
turtle.write("(200,200)")

turtle.penup()
turtle.goto(200,-200)
turtle.pendown()
turtle.circle(30)
turtle.write("(-200,-200)")

turtle.penup()
turtle.home()
turtle.pendown()
turtle.circle(50)
turtle.write("Home")
```



# Commit & Push

7 files committed

first commit

[Share Project on GitHub](#) On GitLab

PC

Share Project On GitHub

✕

Repository name:

PythonProject7

☐ Private

Remote:

origin

Description:

Share by:

Add account ▾

?

Share

Cancel

Log In via GitHub...

Log In with Token...

Log In to GitHub Enterprise...

PC

Log In to GitHub

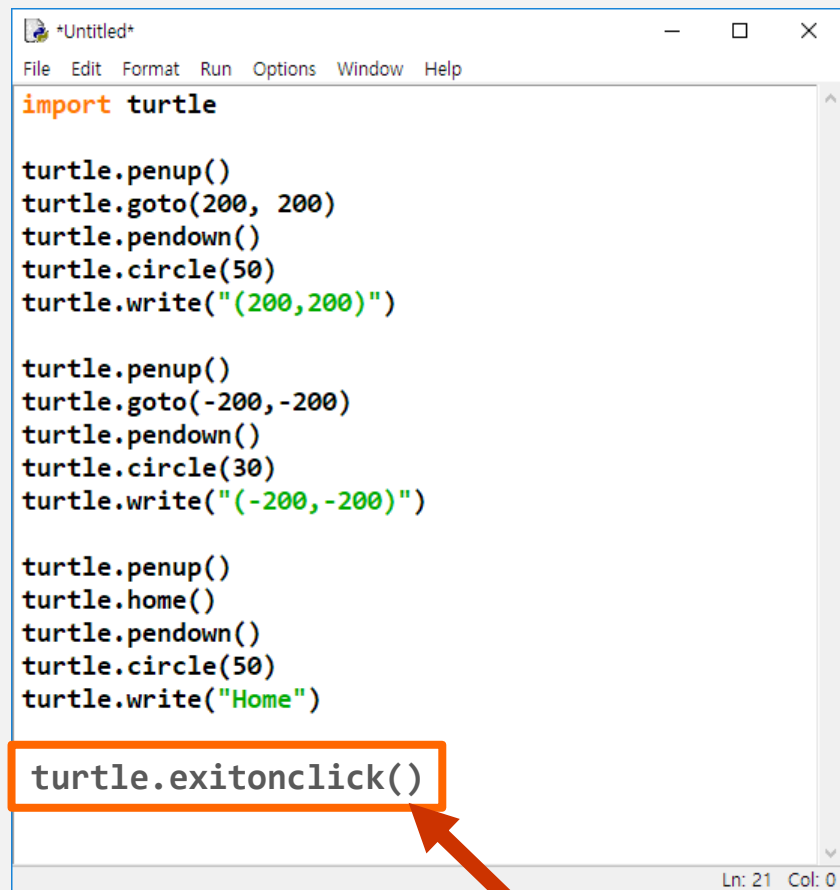
✕

⚙

Logging in...

Cancel





```
*Untitled*
File Edit Format Run Options Window Help

import turtle

turtle.penup()
turtle.goto(200, 200)
turtle.pendown()
turtle.circle(50)
turtle.write("(200,200)")

turtle.penup()
turtle.goto(-200,-200)
turtle.pendown()
turtle.circle(30)
turtle.write("(-200,-200)")

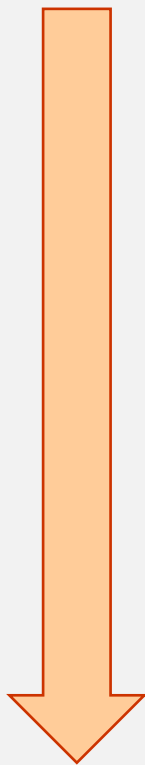
turtle.penup()
turtle.home()
turtle.pendown()
turtle.circle(50)
turtle.write("Home")

turtle.exitonclick()
```

Ln: 21 Col: 0

코드 마지막 부분에 exitonclick() 추가.

# 파이썬 문장은 위에서부터 아래로 차례로 실행



```
circle.py - C:\Users\dustinlee\Desktop\circle.py (3.6.2)
File Edit Format Run Options Window Help

import turtle

turtle.penup()
turtle.goto(200, 200)
turtle.pendown()
turtle.circle(50)
turtle.write("(200,200)")

turtle.penup()
turtle.goto(-200, -200)
turtle.pendown()
turtle.circle(30)
turtle.write("(-200,-200)")

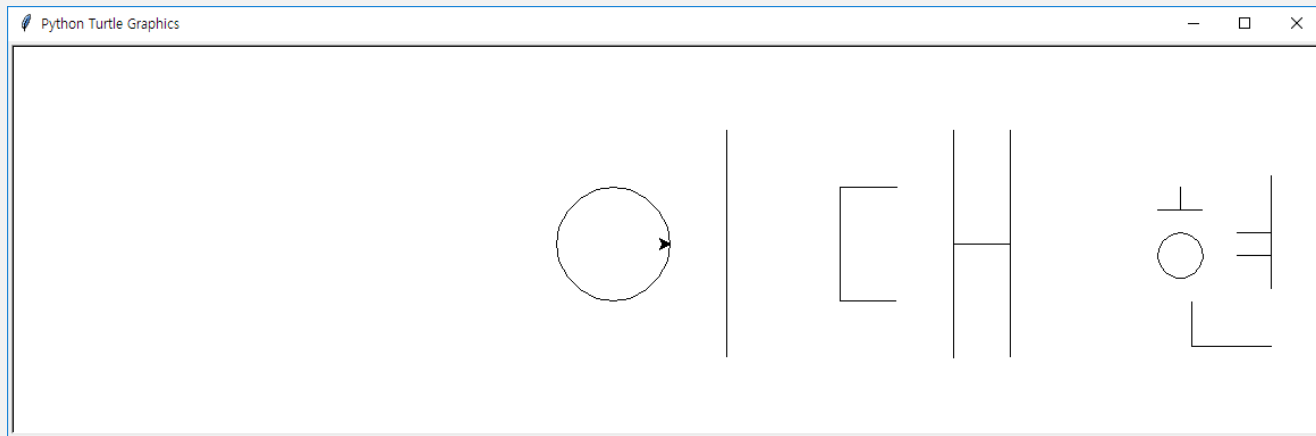
turtle.penup()
turtle.home()
turtle.pendown()
turtle.circle(50)
turtle.write("Home")

turtle.exitonclick()

Ln: 1 Col: 0
```

## 퀴즈 #2. 터틀로 자기 이름 그리기

- `name.py` 로 저장하고, 더블클릭해서 실행.



- 디코 채널에 스크린샷 제출(분반 구별 제출 필수)

# 문법: while 반복문 (Iteration Statement)

---

- 어떤 조건을 만족하는 동안, 계속해서 반복적으로 실행하는 문장.

```
while 조건:
```

```
    문장 1
```

```
    문장 2
```

```
    ...
```

```
else:
```

```
    문장 3
```

```
    문장 4
```

```
    ...
```

```
import turtle
```

```
count = 12
```

```
while count > 0:
```

```
    turtle.forward(100)
```

```
    turtle.left(30)
```

```
    count -= 1
```

```
else:
```

```
    print('count is zero')
```

count가 0 보다 크면 계속해서 반복한다. 뭘?

( turtle을 앞으로 100 이동, 그리고 왼쪽으로 30도 회전, 그리고 count 값 하나 감소)



```
import turtle
```

```
count = 12
```

```
while (count > 0):
```

```
    turtle.forward(100)
```

```
    turtle.left(30)
```

```
    count -= 1
```

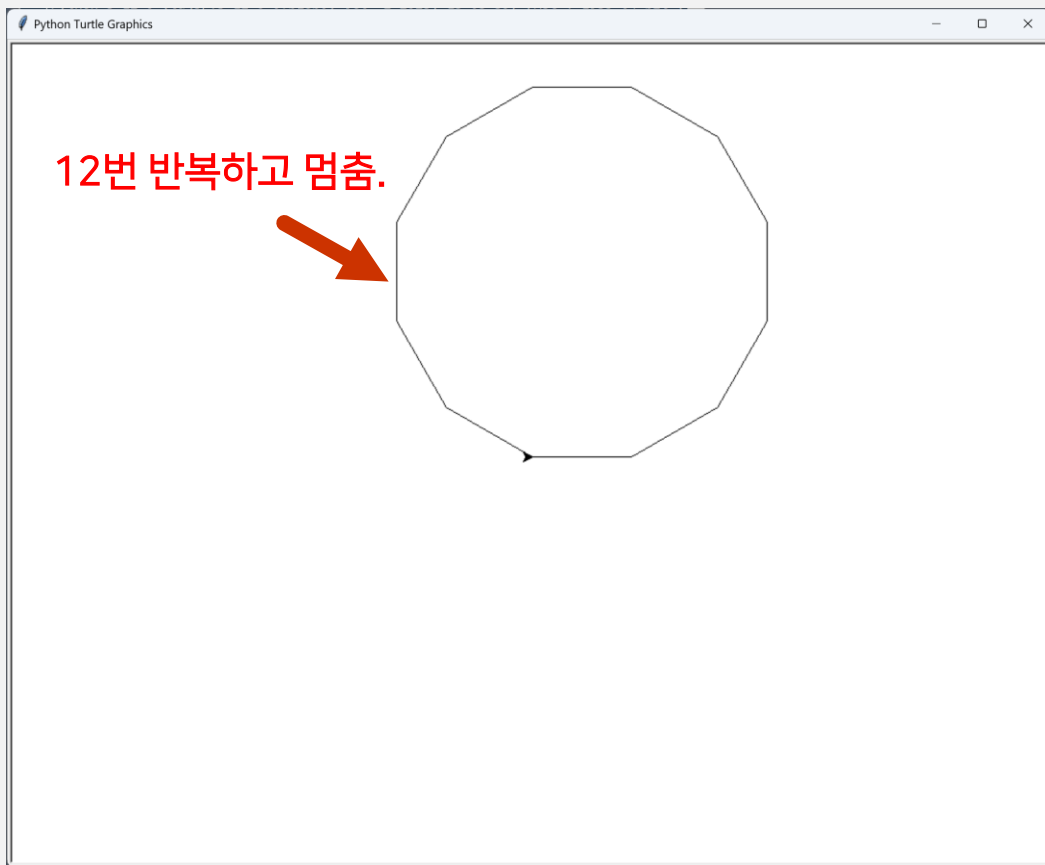
이 조건이 참(True)인 동안



들여쓰기(indentation)

\*\*\* 매우 중요 \*\*\*

여기 블록을 반복적으로 실행한다.



# 문법: 조건문 (Conditional Statement)

- 조건을 검사하여, 그 결과에 따라 처리를 하는 문장

```
import turtle

while True:
    shape = input('Enter Shape: ')
    if shape == 'circle':
        turtle.circle(50)
    elif shape == 'triangle':
        turtle.forward(50); turtle.left(120)
        turtle.forward(50); turtle.left(120)
        turtle.forward(50)
    elif shape == 'quit':
        break
    else:
        print('Wrong Shape')

turtle.bye()
```



```
import turtle
```

```
while True:
```

```
    shape = input('Enter Shape: ')
```

```
    if shape == 'circle':
```

```
        turtle.circle(50)
```

```
    elif shape == 'triangle':
```

```
        turtle.forward(50); turtle.left(120)
```

```
        turtle.forward(50); turtle.left(120)
```

```
        turtle.forward(50)
```

```
    elif shape == 'quit':
```

```
        break
```

```
    else:
```

```
        print('Wrong Shape')
```

```
turtle.bye()
```

이 조건이 참(True)이면,



들여쓰기(indentation)

\*\*\* 매우 중요 \*\*\*

일반적으로 공백4개씩

조건이 참이면, 들여쓰기된 블록을 실행함.

여기 블록에 적힌 대로 실행하라.

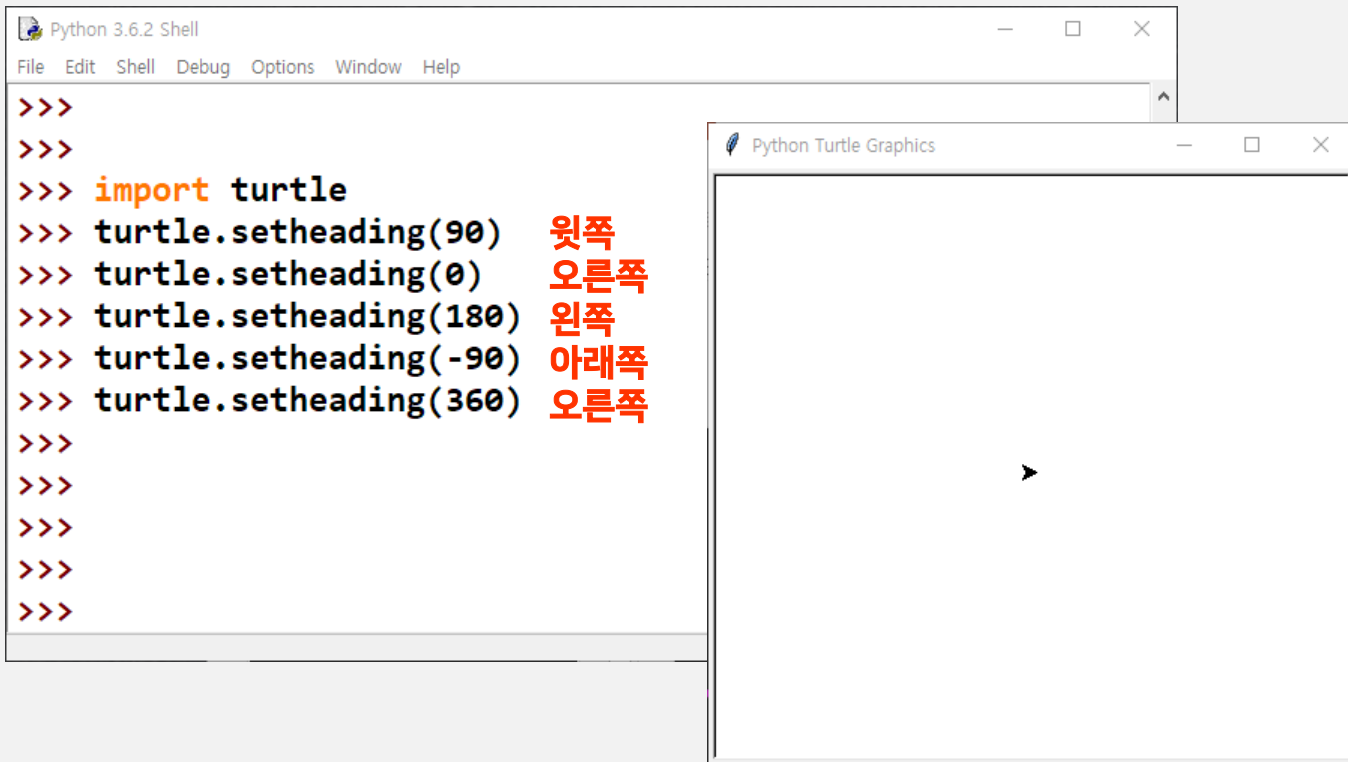
```
shape = input("Enter Shape: ")  
if shape == 'circle':  
    turtle.reset()  
    turtle.circle(50)
```

```
shape = input("Enter Shape: ")  
if shape == 'circle':  
    turtle.reset()  
turtle.circle(50)
```



# 거북이의 방향 설정

## `turtle.setheading(각도)`



# random 모듈

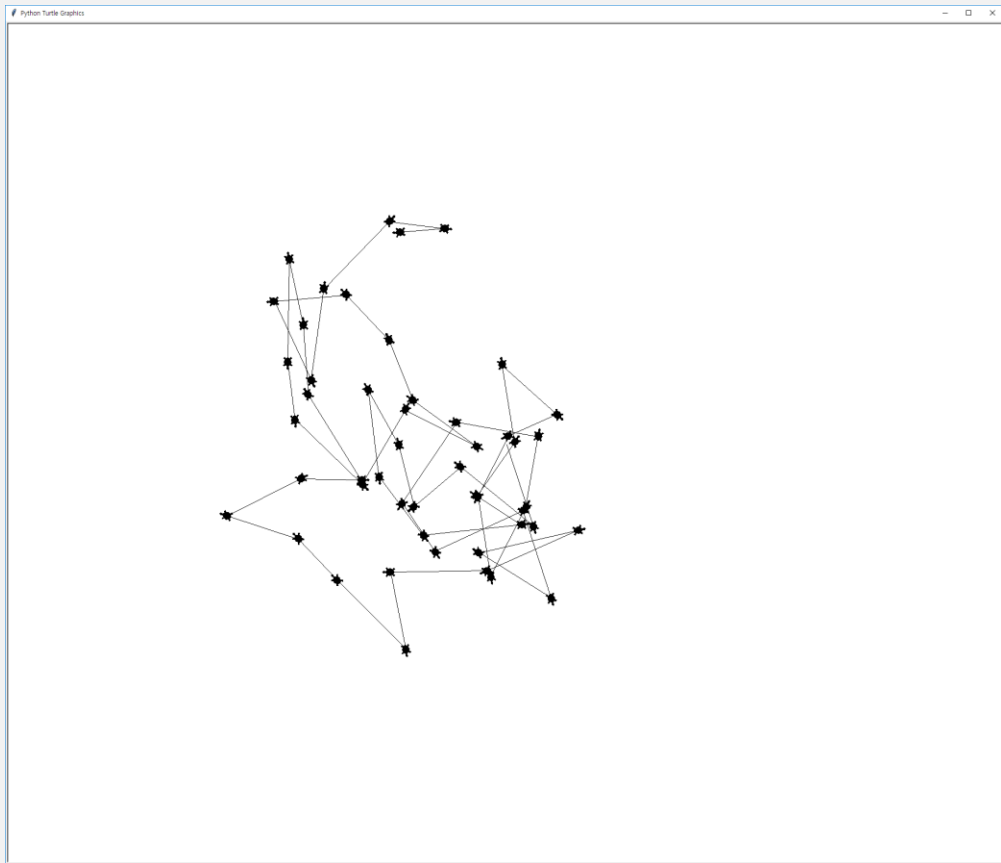
- 주사위를 던지면 어떤 수가 나올까? 무작위로 결정
- 무작위로 어떤 숫자를 뽑아내고자 할 때, random 모듈을 사용하면 된다.



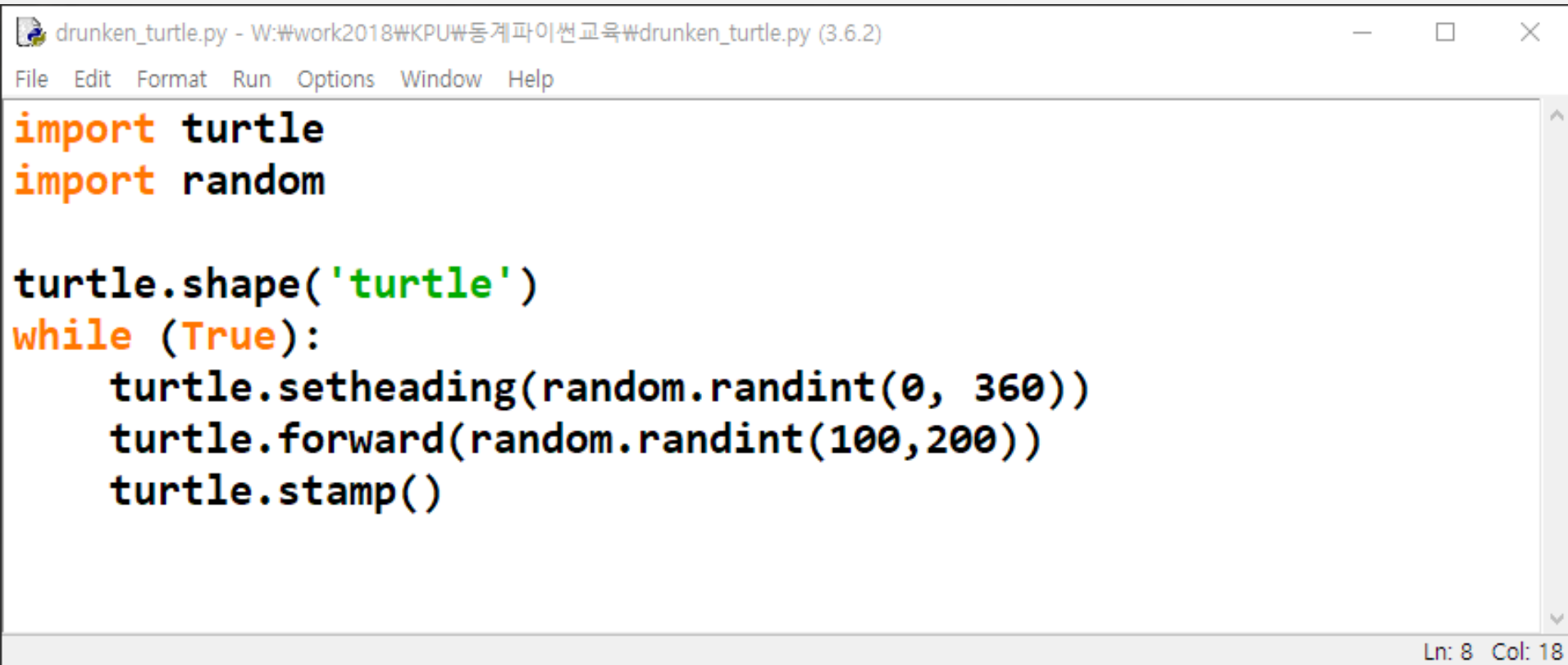
```
Python 3.6.2 Shell
File Edit Shell Debug Options Window Help
>>>
>>> import random
>>> random.randint(1,6)
3
>>> random.randint(1,6)
4
>>> random.randint(1,6)
2
>>> random.randint(1,6)
4
>>> random.randint(1,6)
1
>>>
```

random.randint(시작, 끝)

# 술취한 거북이?



# drunken\_turtle.py



```
drunken_turtle.py - W:\work2018\WKPU\등계파이썬교육\drunken_turtle.py (3.6.2)
File Edit Format Run Options Window Help

import turtle
import random

turtle.shape('turtle')
while (True):
    turtle.setheading(random.randint(0, 360))
    turtle.forward(random.randint(100, 200))
    turtle.stamp()

Ln: 8 Col: 18
```