

Lecture #2. 파이썬 기초 (2)

2D 게임 프로그래밍

이대현 교수



한국공학대학교
TECH UNIVERSITY OF KOREA

List

```
>>> twice = ['momo', 'sana', 'zwi', 'nayun', 'dahyun']
>>> black_pink = ['jisu', 'jeni', 'rose', 'risa']
>>> twice
['momo', 'sana', 'zwi', 'nayun', 'dahyun']
>>> twice.append('jihyo')
>>> twice
['momo', 'sana', 'zwi', 'nayun', 'dahyun', 'jihyo']
>>> twice.sort()
>>> twice
['dahyun', 'jihyo', 'momo', 'nayun', 'sana', 'zwi']
>>> len(twice)
6
>>> unite = twice + black_pink
>>> unite
['dahyun', 'jihyo', 'momo', 'nayun', 'sana', 'zwi', 'jisu', 'jeni', 'rose', 'risa']
>>> unite.remove('momo')
>>> unite
['dahyun', 'jihyo', 'nayun', 'sana', 'zwi', 'jisu', 'jeni', 'rose', 'risa']
```

List 에서 Slice 가 적용됨.


```
>>> unite[0]
'dahyun'
>>> unite[-1]
'risa'
>>> unite[:3]
['dahyun', 'jihyo', 'nayun']
>>> unite[-3:]
['jeni', 'rose', 'risa']
```

Dictionary

```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help

>>> score = { 'momo' : 80, 'zwi' : 85, 'sana' : 98 }
>>> type(score)
<class 'dict'>
>>> score['momo']
80
>>> score['nayun']
Traceback (most recent call last):
  File "<pyshell#3>", line 1, in <module>
    score['nayun']
KeyError: 'nayun'
>>> score['nayun'] = 100
>>> score
{'momo': 80, 'zwi': 85, 'sana': 98, 'nayun': 100}
>>> del score['momo']
>>> score
{'zwi': 85, 'sana': 98, 'nayun': 100}
>>> score.keys()
dict_keys(['zwi', 'sana', 'nayun'])
>>> score.values()
dict_values([85, 98, 100])
>>>
```

Ln: 12 Col: 0

 Python 3.7.0 Shell

File Edit Shell Debug Options Window Help

```
>>> 'zwi' in score
```

```
True
```

```
>>> 'momo' in score
```

```
False
```

```
>>>
```

```
>>> score.clear()
```

```
>>> score
```

```
{}
```

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.

```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
>>>
>>> t1 = (1,2,3)
>>> t2 = (1, )
>>> t3 = ()
>>> t4 = 1,2,3,4
>>> t4
(1, 2, 3, 4)
>>> type(t4)
<class 'tuple'>
>>> t5 = (1, 'a', "park", (1, 2))
>>> t1[1:]
(2, 3)
>>> t1 + t5
(1, 2, 3, 1, 'a', 'park', (1, 2))
>>> t4 * t4
Traceback (most recent call last):
  File "<pyshell#15>", line 1, in <module>
    t4 * t4
TypeError: can't multiply sequence by non-int of type 'tuple'
>>> t4 * 2
(1, 2, 3, 4, 1, 2, 3, 4)
>>> |
```

Ln: 37 Col: 4

set

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

```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
>>> s1 = {1,2,3}
>>> type(s1)
<class 'set'>
>>> s1 = {1,2,2,4}
>>> s1
{1, 2, 4}
>>> l1 = [1,2,2,2,2,3,3,3,3,5,5,5,5,5]
>>> s1 = set(l1)
>>> s1
{1, 2, 3, 5}
>>> s2 = {3,5,6,7}
>>> s1 + s2
Traceback (most recent call last):
  File "<pyshell#36>", line 1, in <module>
    s1 + s2
TypeError: unsupported operand type(s) for +: 'set' and 'set'
>>> s1 | s2
{1, 2, 3, 5, 6, 7}
>>> s1 & s2
{3, 5}
>>> s2 - s1
{6, 7}
>>> s1 - s2
{1, 2}
>>> s1.add(8)
>>> s1
{1, 2, 3, 5, 8}
>>> s2.remove(6)
>>> s2
{3, 5, 7}
```

Ln: 86 Col: 4

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.

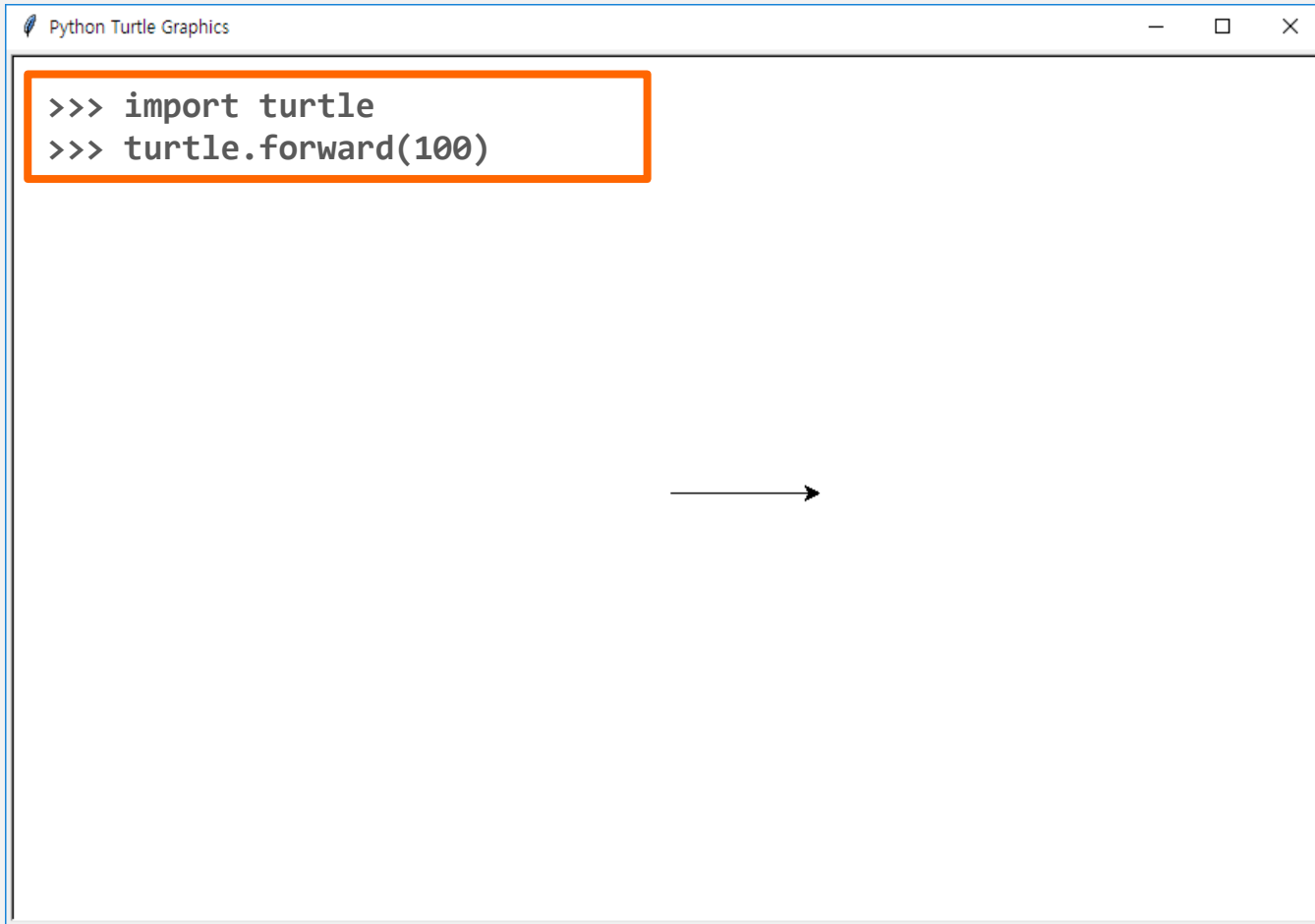


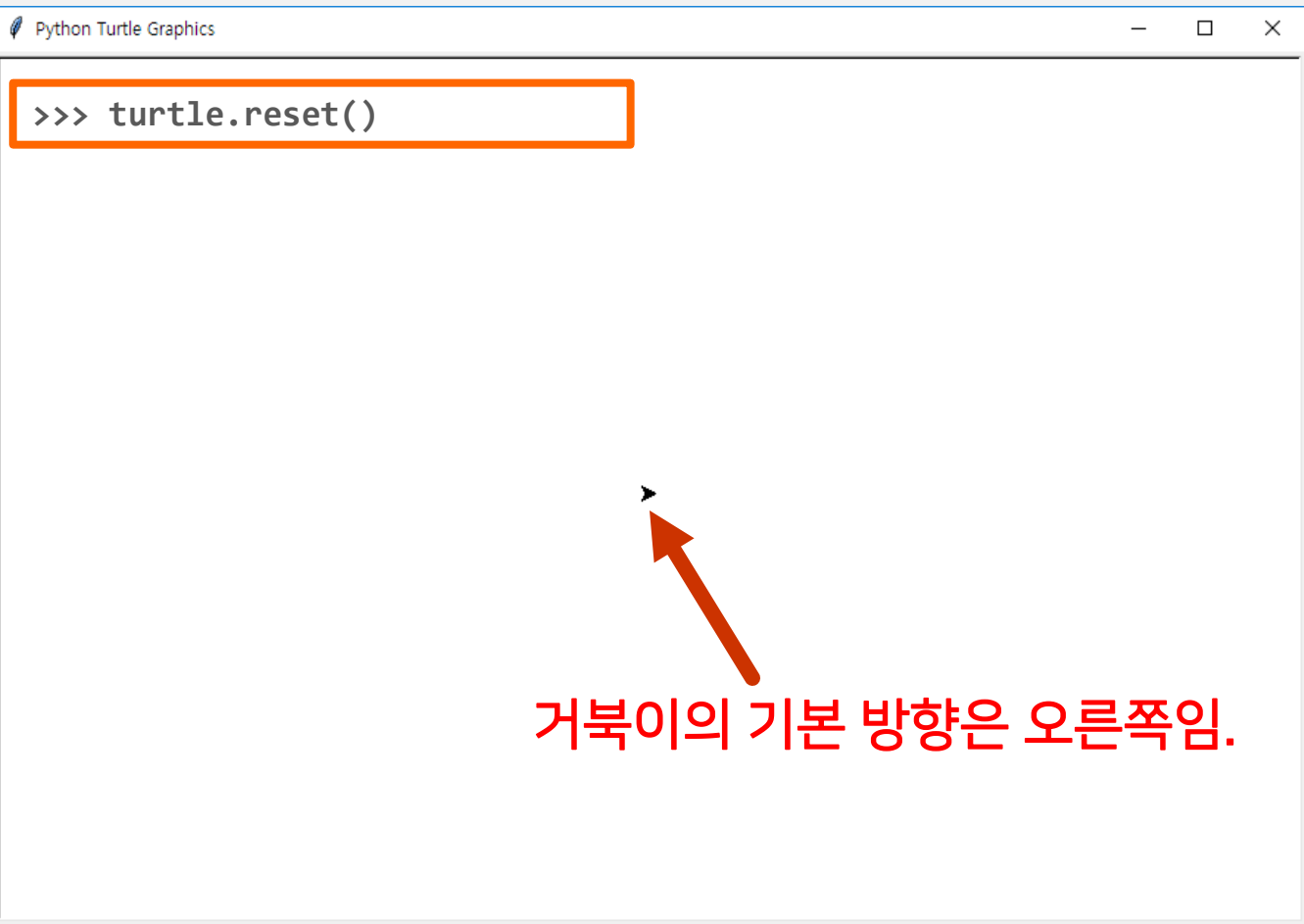
```
import turtle
```

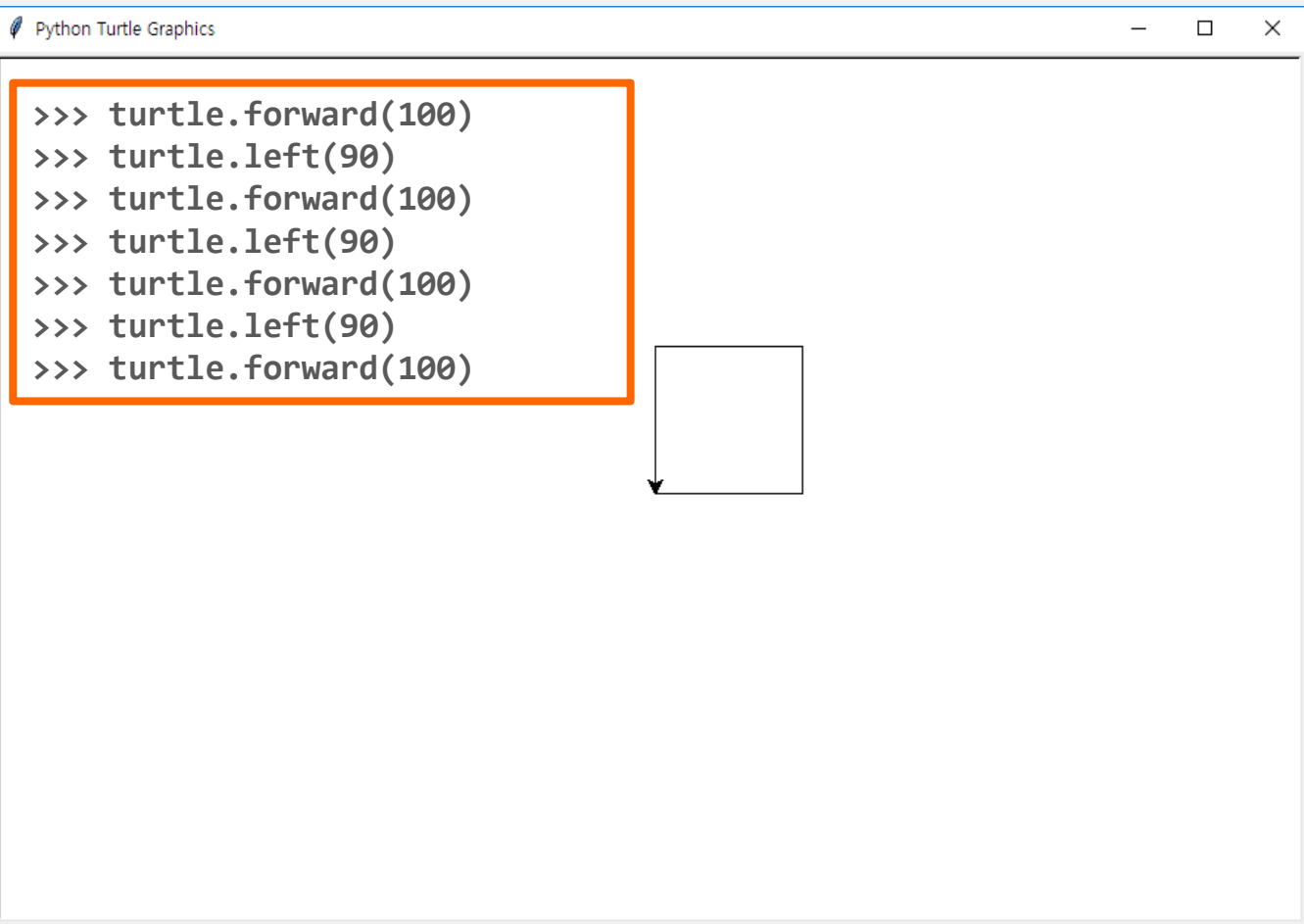
```
turtle.forward(100)
```



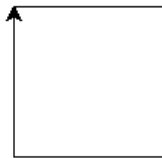
turtle 이 갖고 있는 기능(함수, function)
을 이용하여, 그림을 그린다.







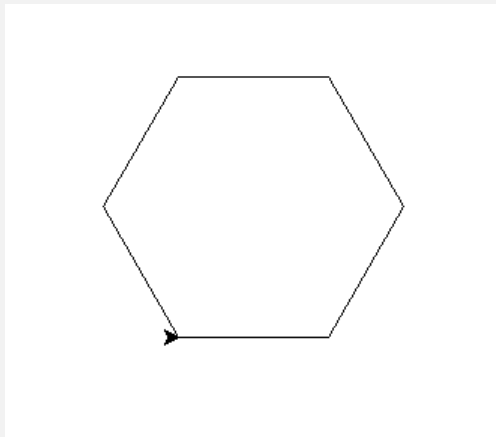
```
>>> turtle.reset()
>>> turtle.forward(100)
>>> turtle.right(90)
>>> turtle.forward(100)
>>> turtle.right(90)
>>> turtle.forward(100)
>>> turtle.right(90)
>>> turtle.forward(100)
```

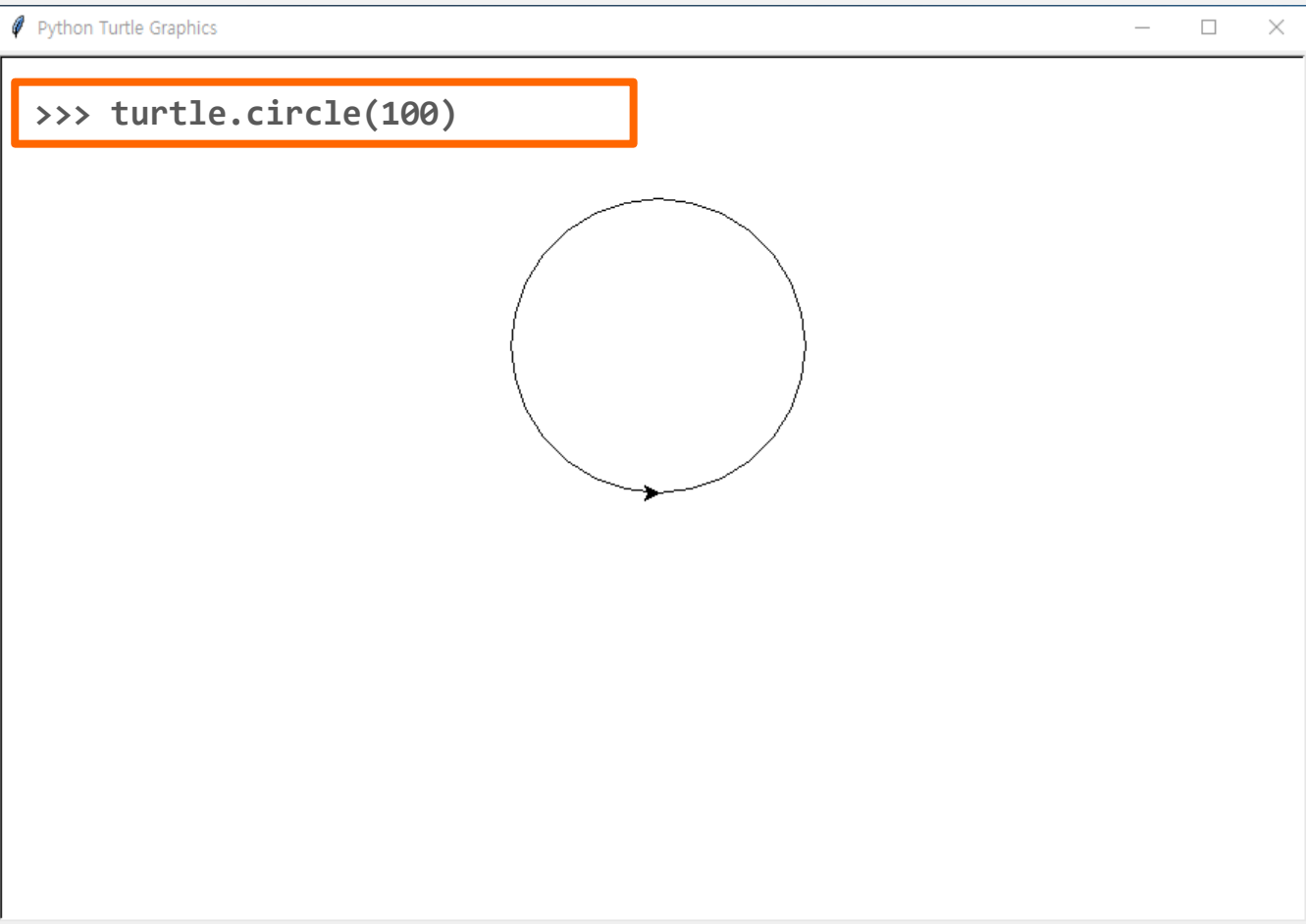


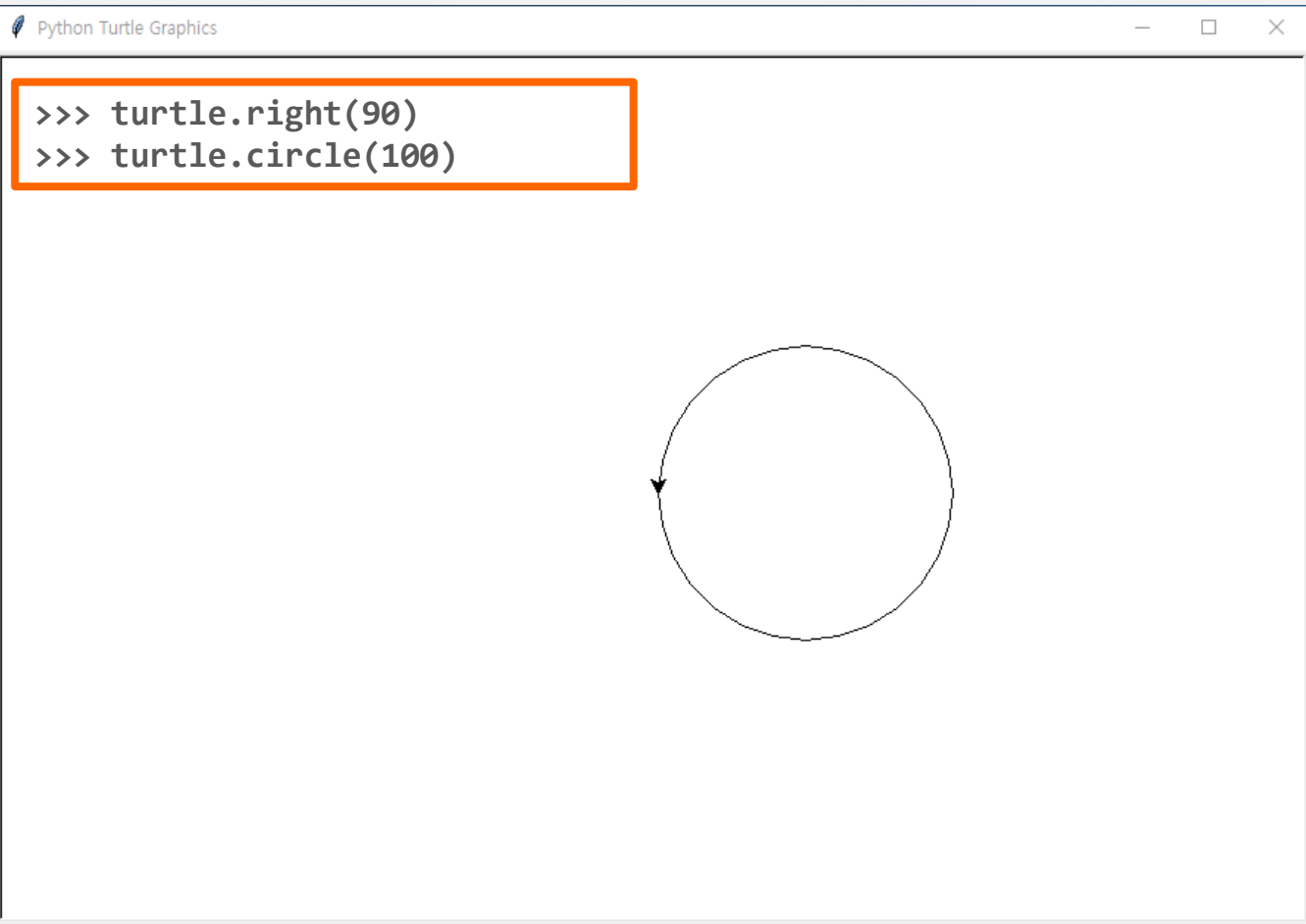
```
>>> turtle.forward(100)
>>> turtle.left(120)
>>> turtle.forward(100)
>>> turtle.left(120)
>>> turtle.forward(100)
```



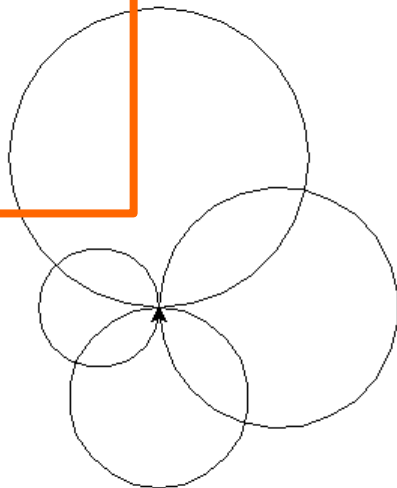
퀴즈 #1: 정육각형을 그려보자!



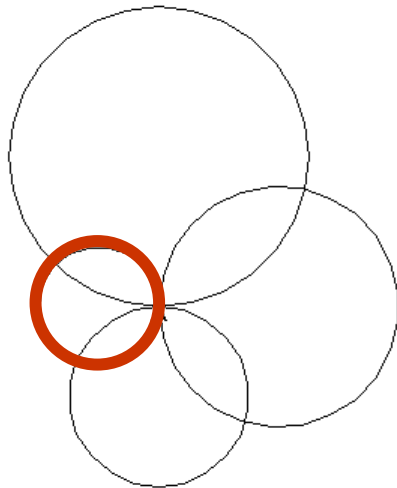




```
>>> turtle.circle(100)
>>> turtle.right(90)
>>> turtle.circle(80)
>>> turtle.right(90)
>>> turtle.circle(60)
>>> turtle.right(90)
>>> turtle.circle(40)
```



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

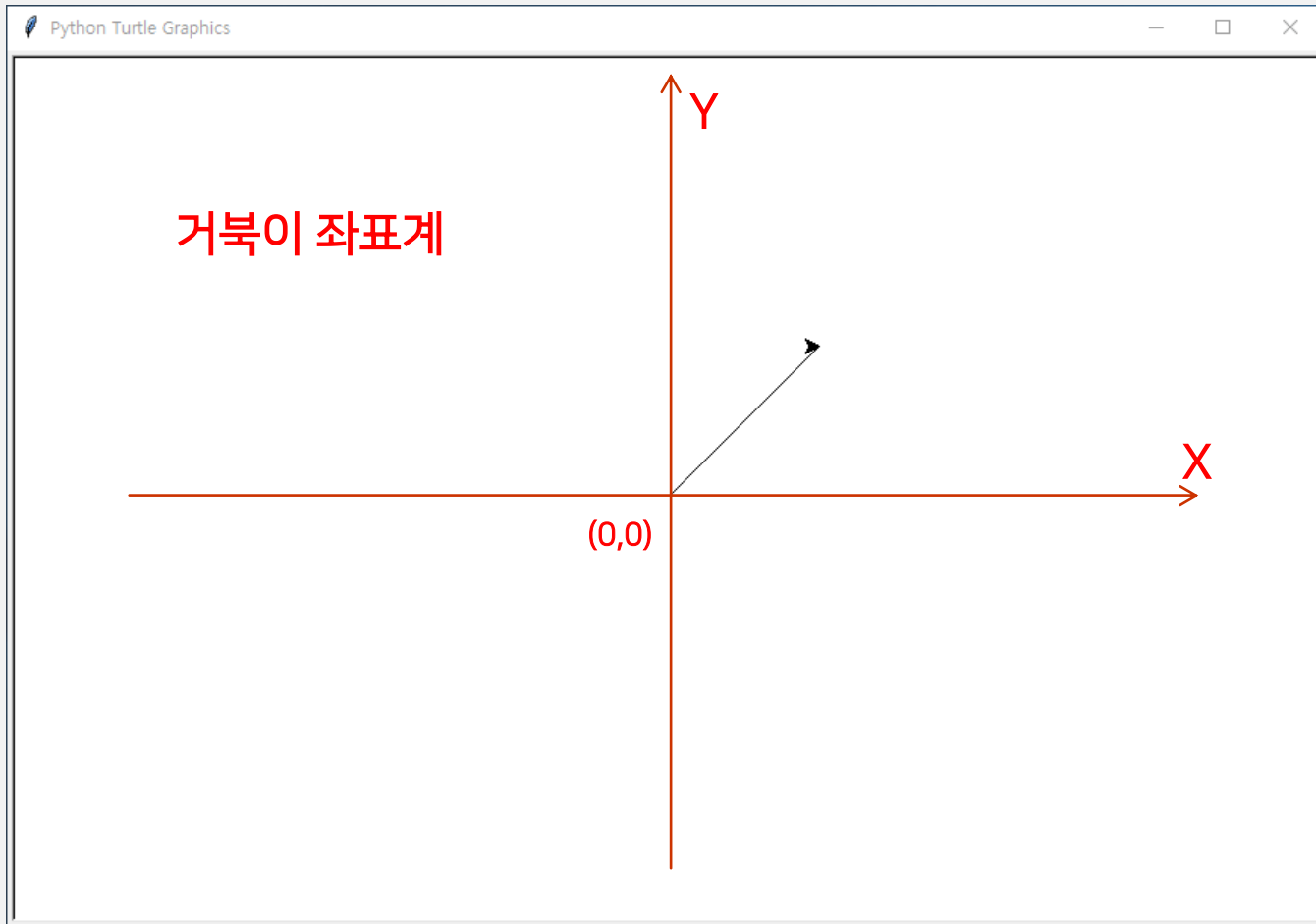


마지막에 그렸던 원이 없어짐.
이전 상태로 되돌아감.

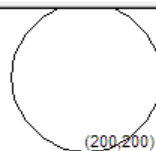
```
>>> turtle.reset()  
>>> turtle.goto(100, 100)
```



(100,100) 으로 이동한다.
거북이의 머리방향은 변함없이 여전히 오른쪽 방향.

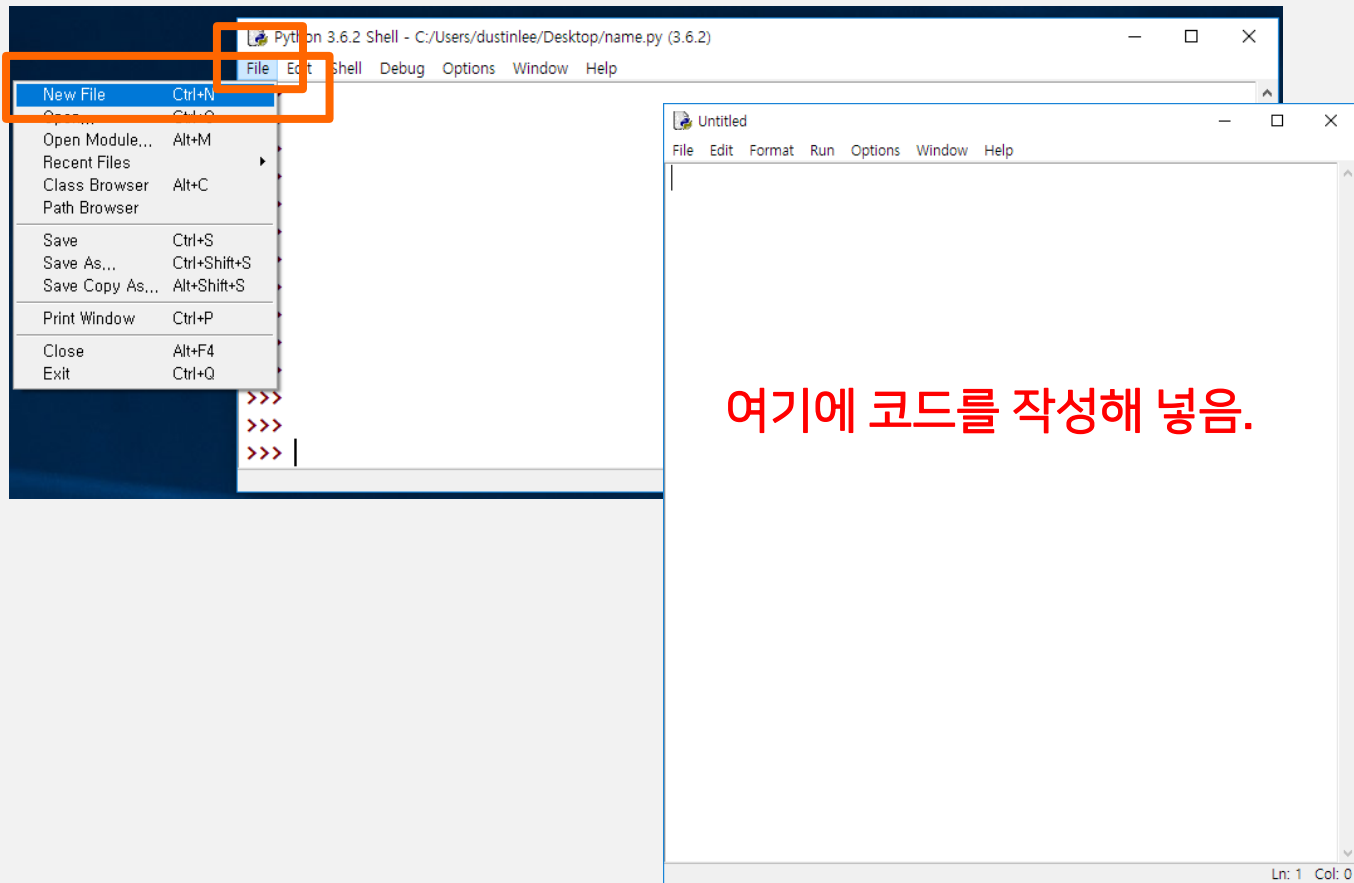


```
>>> turtle.penup()
>>> turtle.goto(200, 200)
>>> turtle.pendown()
>>> turtle.circle(50)
>>> turtle.write("(200,200)")
>>>
>>> turtle.penup()
>>> turtle.goto(-200,-200)
>>> turtle.pendown()
>>> turtle.circle(30)
>>> turtle.write("(-200,-200)")
>>>
>>> turtle.penup()
>>> turtle.home()
>>> turtle.pendown()
>>> turtle.circle(50)
>>> turtle.write("Home")
```

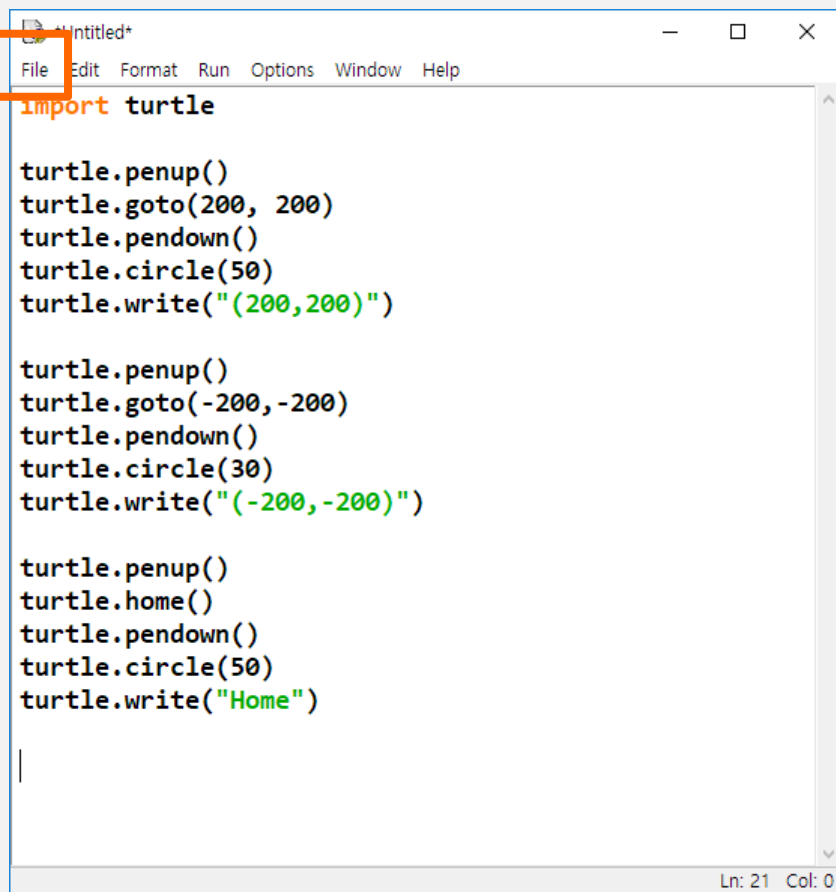


Home

프로그램을 파일로 만들어서 저장



New File	Ctrl+N
Open...	Ctrl+O
Open Module...	Alt+M
Recent Files	
Class Browser	Alt+C
Path Browser	
Save	Ctrl+S
Save As...	Ctrl+Shift+S
Save Copy As...	Alt+Shift+S
Print Window	Ctrl+P
Close	Alt+F4
Exit	Ctrl+Q



The screenshot shows a Python IDE window titled "Untitled*" with a menu open. The menu is located on the left side of the window, and the "Save" option is highlighted. The code in the editor is as follows:

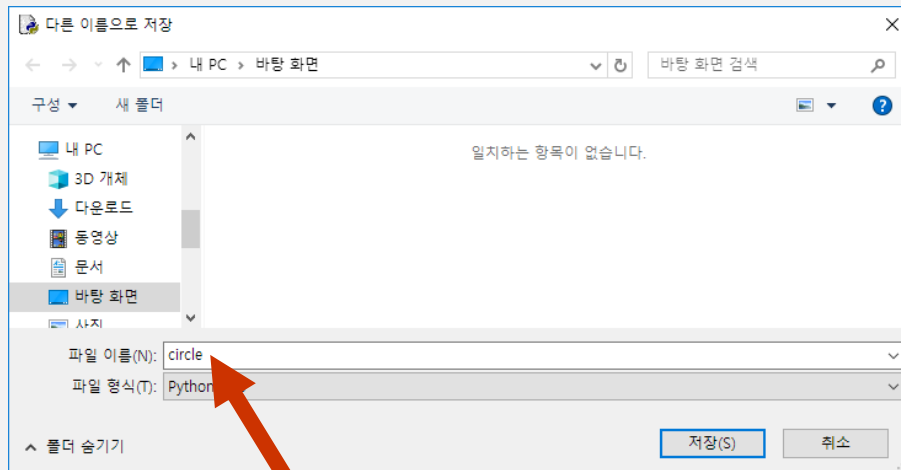
```
import turtle

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

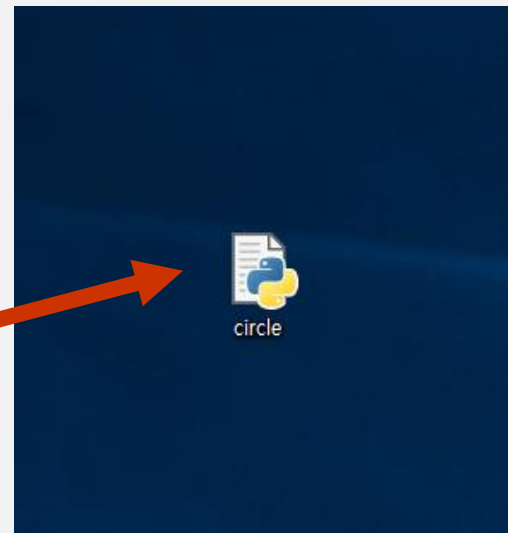
turtle.penup()
turtle.goto(-200,-200)
turtle.pendown()
turtle.circle(30)
turtle.write("(-200,-200)")

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

The status bar at the bottom right of the window shows "Ln: 21 Col: 0".



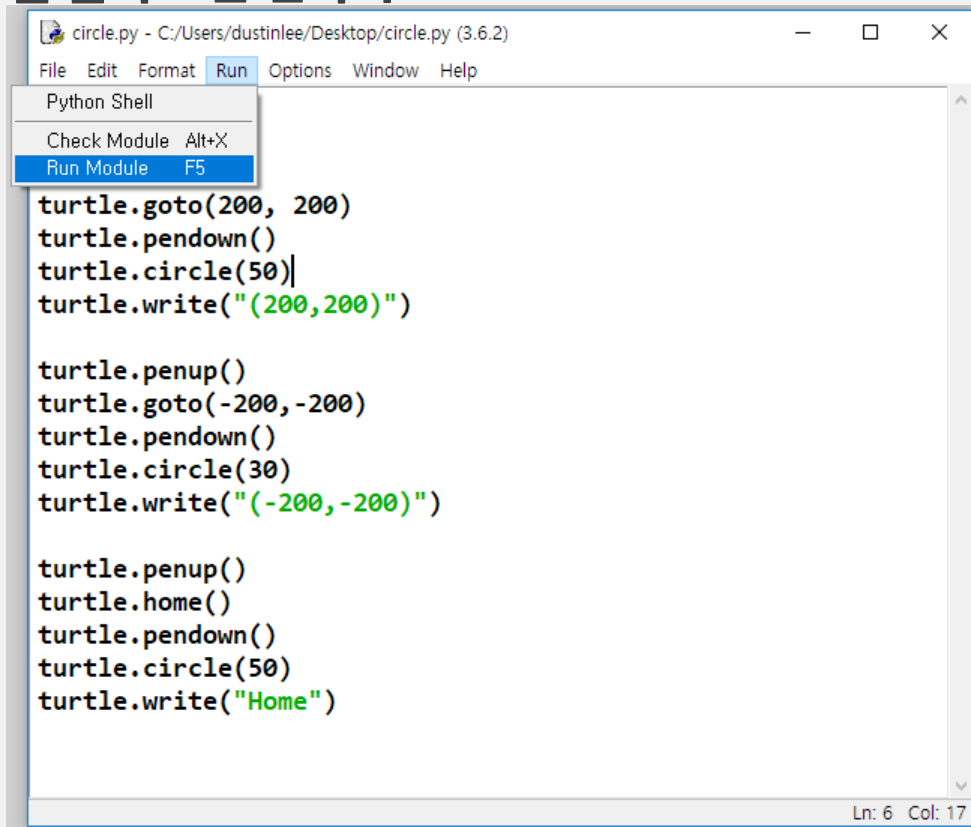
circle이라는 이름으로 바탕
화면에 저장.



바탕화면에 circle.py 라는
이름의 파일이 생성됨.

프로그램 실행 방법 #1

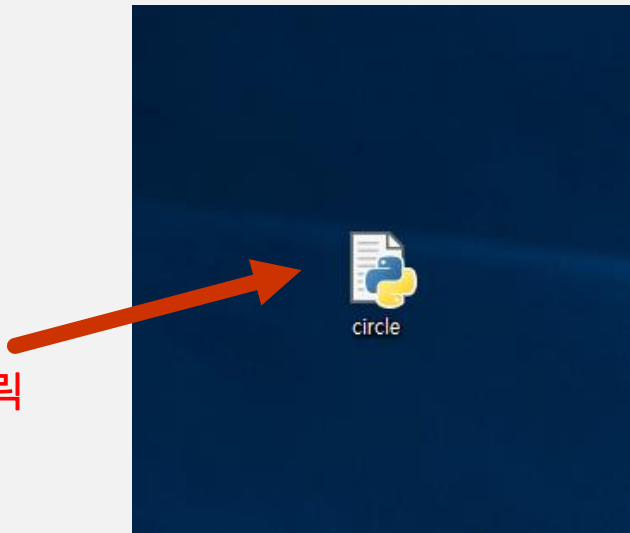
- Run→Run Module 을 클릭 또는 단축기 F5

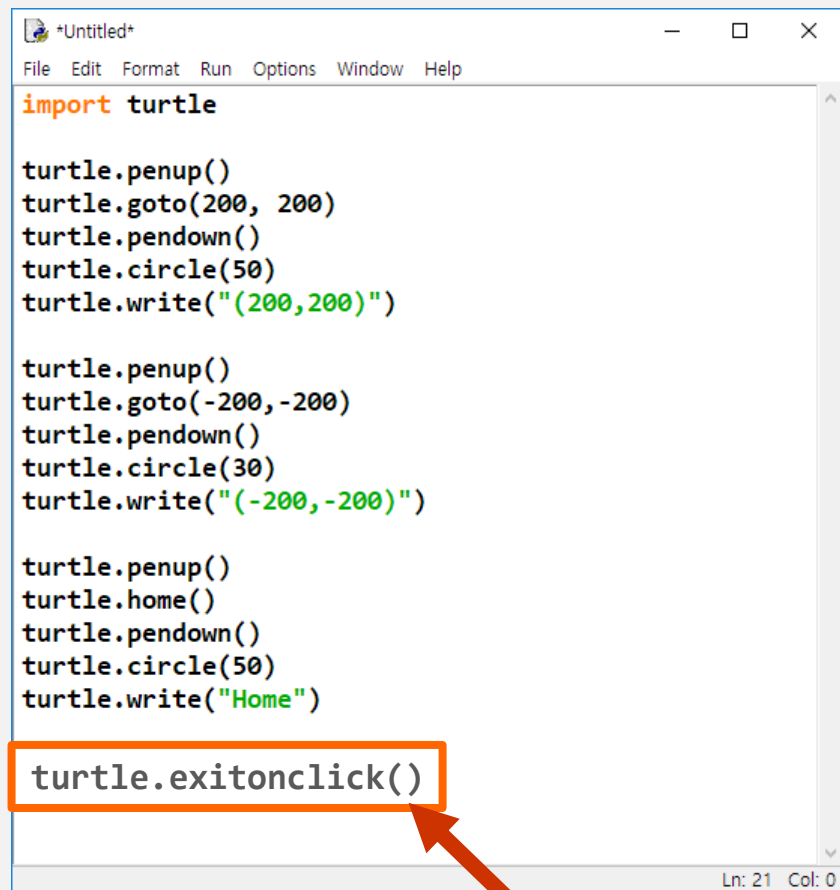


프로그램 실행 방법 #2

- 프로그램 파일을 더블 클릭하여 실행.
- 문제점은?

circle.py 를 더블 클릭





```
*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)")

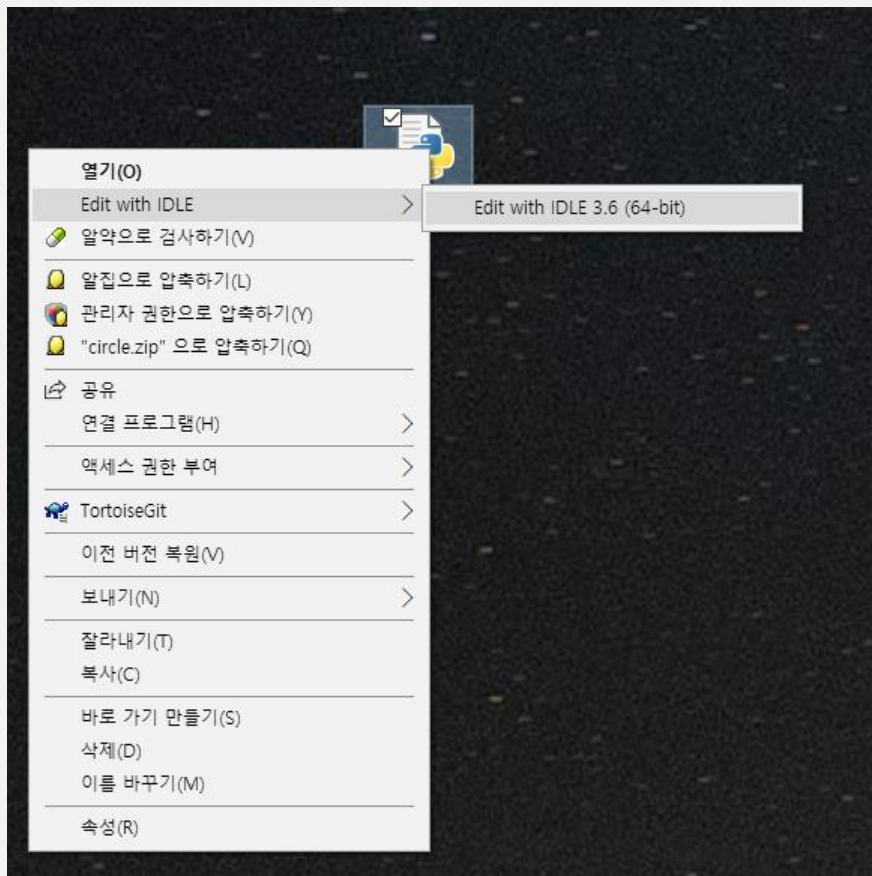
turtle.penup()
turtle.home()
turtle.pendown()
turtle.circle(50)
turtle.write("Home")

turtle.exitonclick()
```

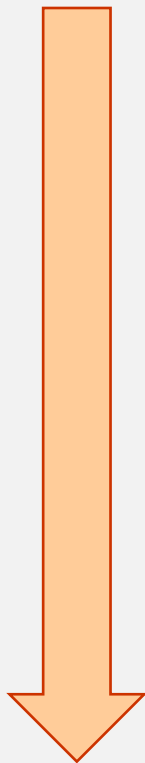
Ln: 21 Col: 0

코드 마지막 부분에 exitonclick() 추가.

마우스 오른쪽 버튼을 클릭하면, 소스코드를 직접 편집 가능.



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)")

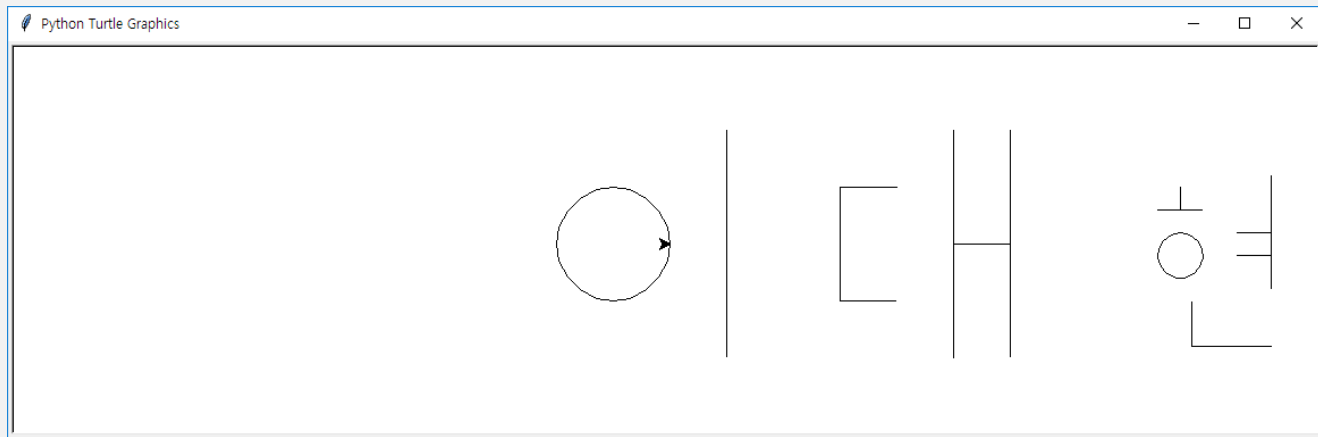
turtle.penup()
turtle.home()
turtle.pendown()
turtle.circle(50)
turtle.write("Home")

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)