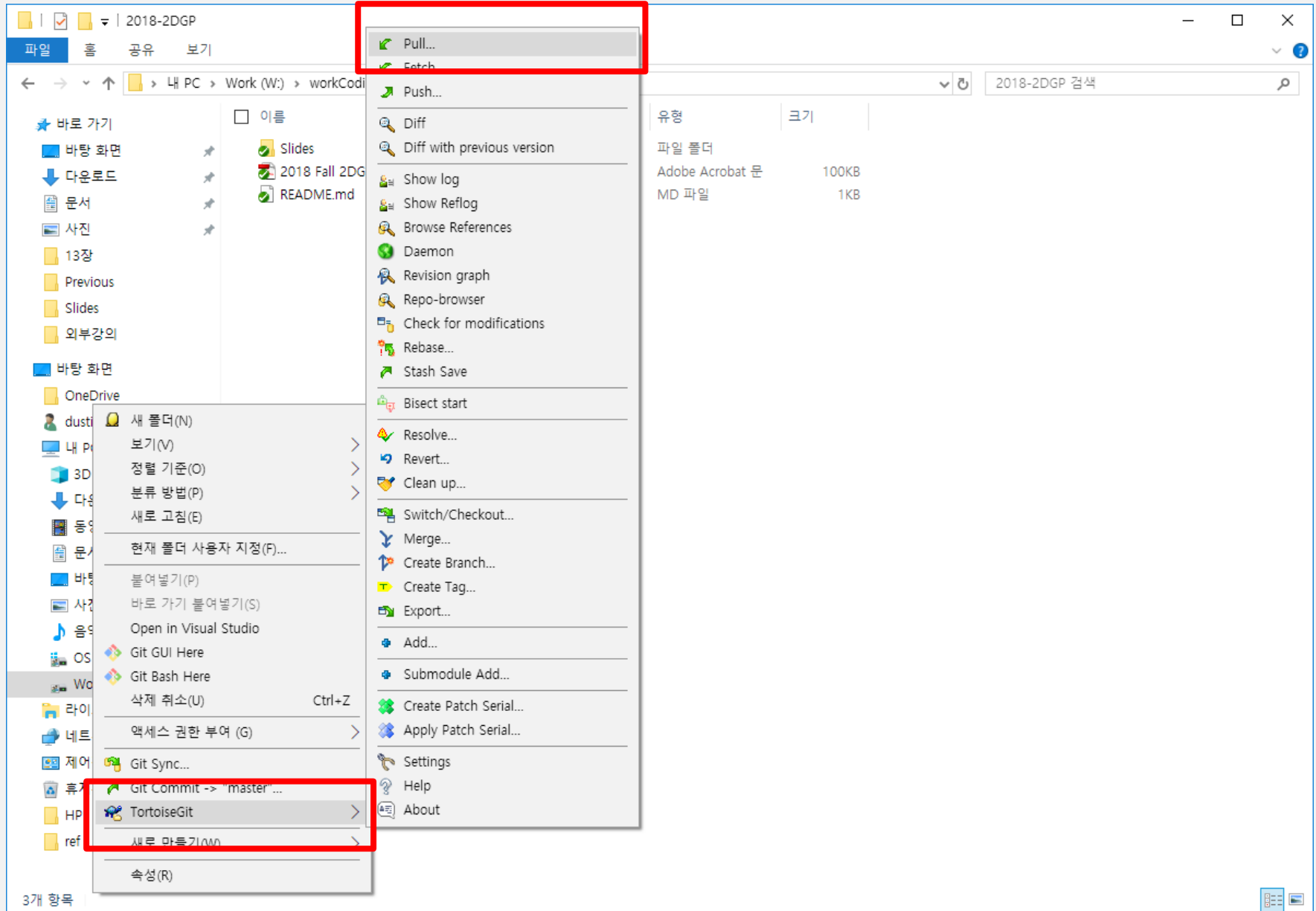
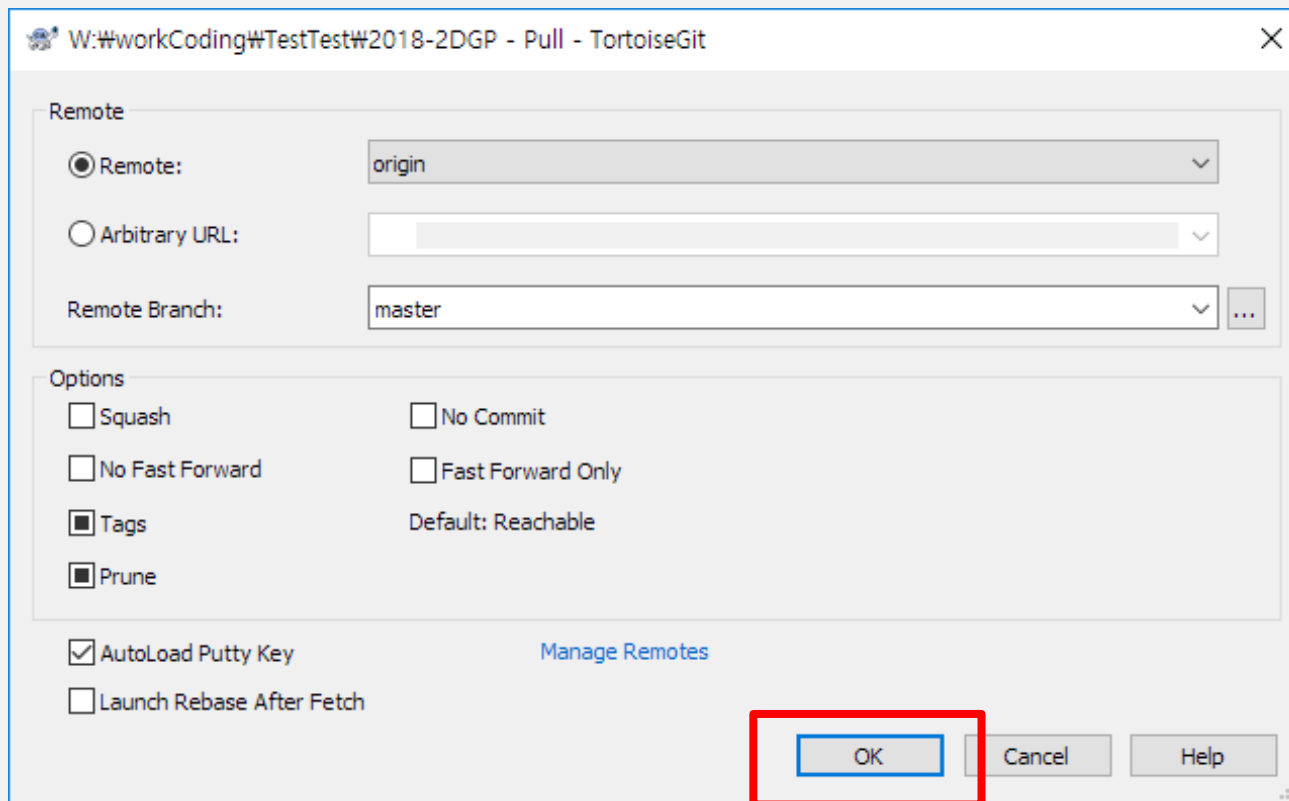


Lecture #1. 파이썬 기초 (2)

Git Pull – 서버에서 업데이트된 내용을 내려받을 때







remote: Compressing objects

```
git.exe pull --progress -v --no-rebase "origin"
```

```
POST git-upload-pack (437 bytes)
```

```
remote: Counting objects: 12, done.
```

```
remote: Compressing objects: 100% (10/10), done.
```

```
remote: Total 12 (delta 1), reused 12 (delta 1), pack-reused 0
```

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

```
46e421a..bb5c854 master -> origin/master
```

```
Updating 46e421a..bb5c854
```

```
Fast-forward
```

Labs/Lab01/bus_fare.py	17 ++++++
Labs/Lab01/draw_circle.py	20 ++++++
Labs/Lab01/drunken_turtle.py	9 ++++++
Labs/Lab01/random_turtle.py	18 ++++++
Labs/Lab01/random_turtle_function.py	11 ++++++
Labs/Lab01/ten_lines.py	20 ++++++
Labs/Lab01/test_age.py	16 ++++++
Labs/Lab01/while_turtle.py	11 ++++++

```
8 files changed, 122 insertions(+)
```

```
create mode 100644 Labs/Lab01/bus_fare.py
```

```
create mode 100644 Labs/Lab01/draw_circle.py
```

```
create mode 100644 Labs/Lab01/drunken_turtle.py
```

```
create mode 100644 Labs/Lab01/random_turtle.py
```

```
create mode 100644 Labs/Lab01/random_turtle_function.py
```

```
create mode 100644 Labs/Lab01/ten_lines.py
```

```
create mode 100644 Labs/Lab01/test_age.py
```

```
create mode 100644 Labs/Lab01/while_turtle.py
```

```
Success (1187 ms @ 2018-09-03 오전 8:55:54)
```

Pulled Diff



Close

Abort

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.



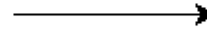
```
import turtle
```

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



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

```
>>> import turtle  
>>> turtle.forward(100)
```

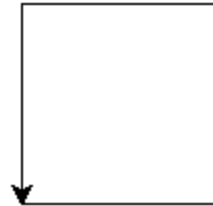


```
>>> turtle.reset()
```

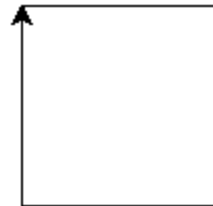


거북이의 기본 방향은 오른쪽임.

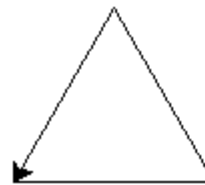

```
>>> turtle.forward(100)
>>> turtle.left(90)
>>> turtle.forward(100)
>>> turtle.left(90)
>>> turtle.forward(100)
>>> turtle.left(90)
>>> turtle.forward(100)
```



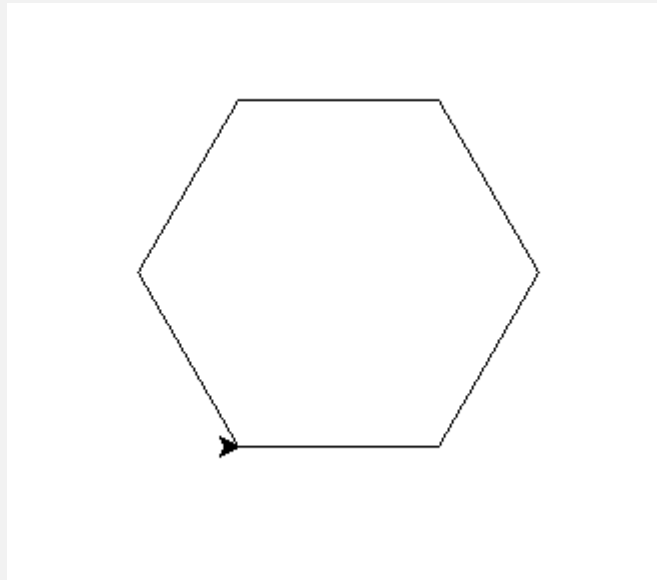
```
>>> turtle.reset()  
>>> turtle.forward(100)  
>>> turtle.right(90)  
>>> turtle.forward(100)  
>>> turtle.right(90)  
>>> turtle.forward(100)  
>>> turtle.right(90)  
>>> turtle.forward(100)
```



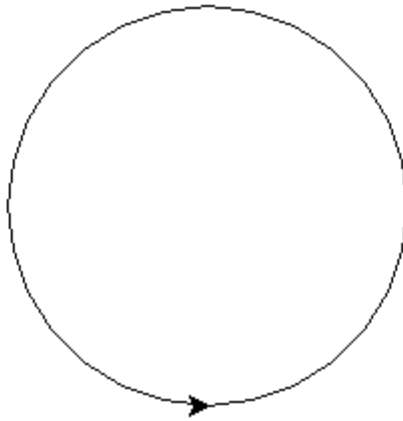
```
>>> turtle.forward(100)
>>> turtle.left(120)
>>> turtle.forward(100)
>>> turtle.left(120)
>>> turtle.forward(100)
```



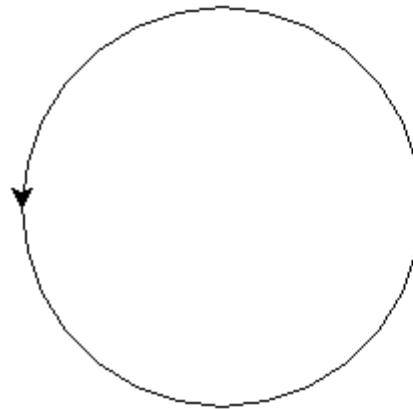
퀴즈 #1: 정육각형을 그려보자 !



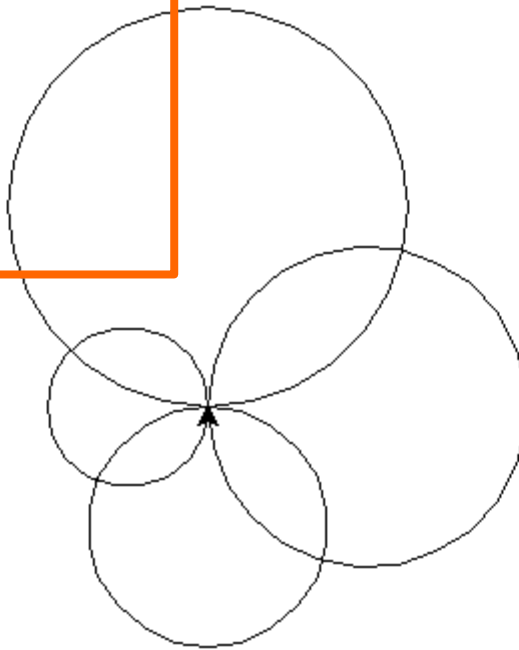
```
>>> turtle.circle(100)
```



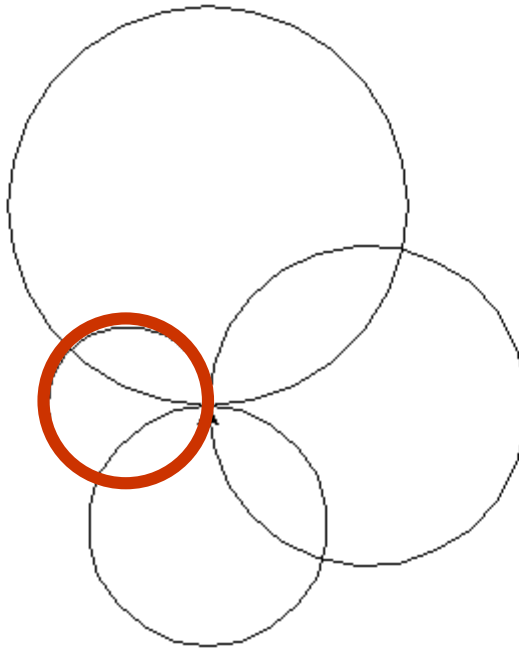
```
>>> turtle.right(90)  
>>> turtle.circle(100)
```



```
>>> 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()
```

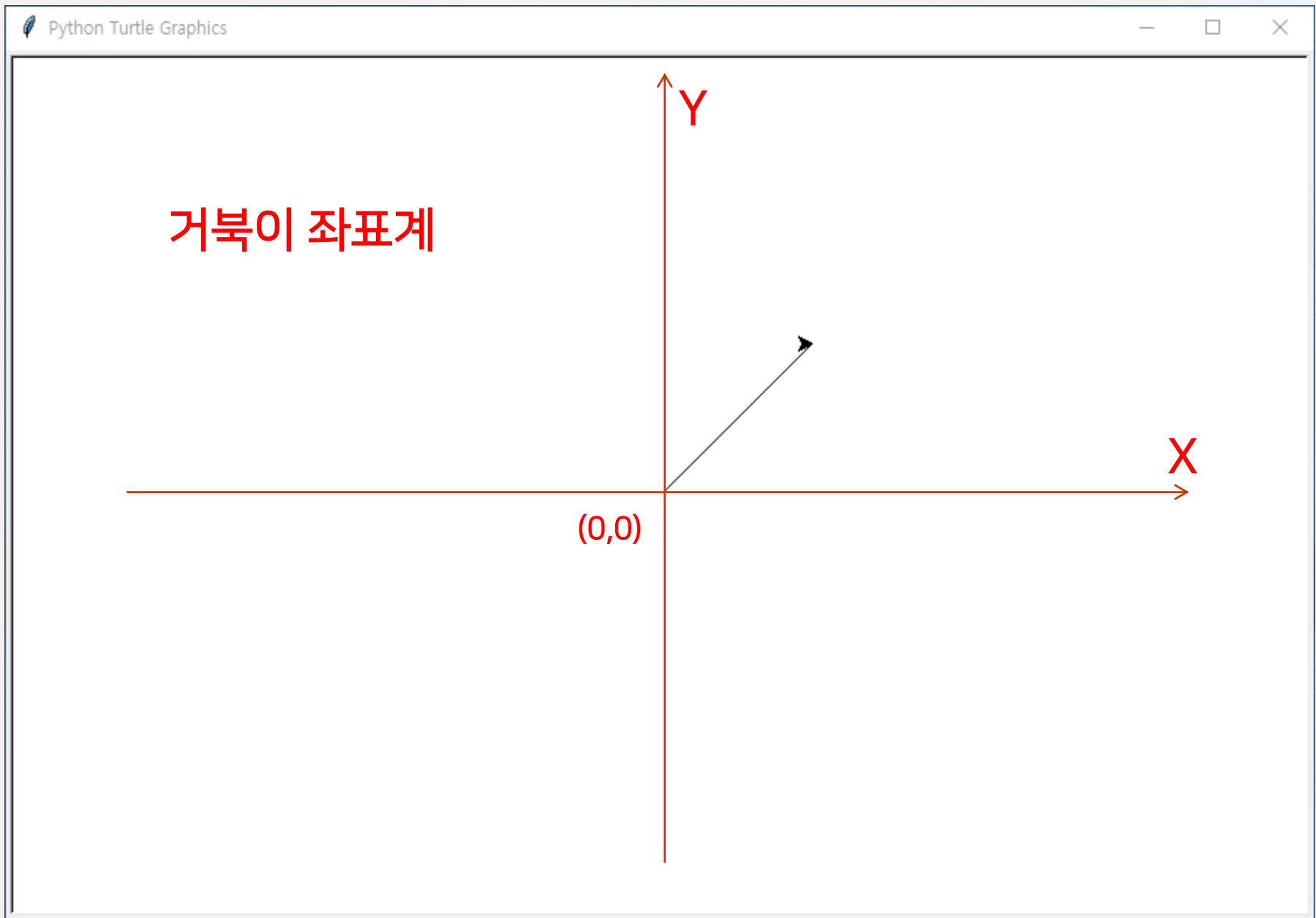


마지막에 그렸던 원이 없어짐.
이전 상태로 되돌아감.

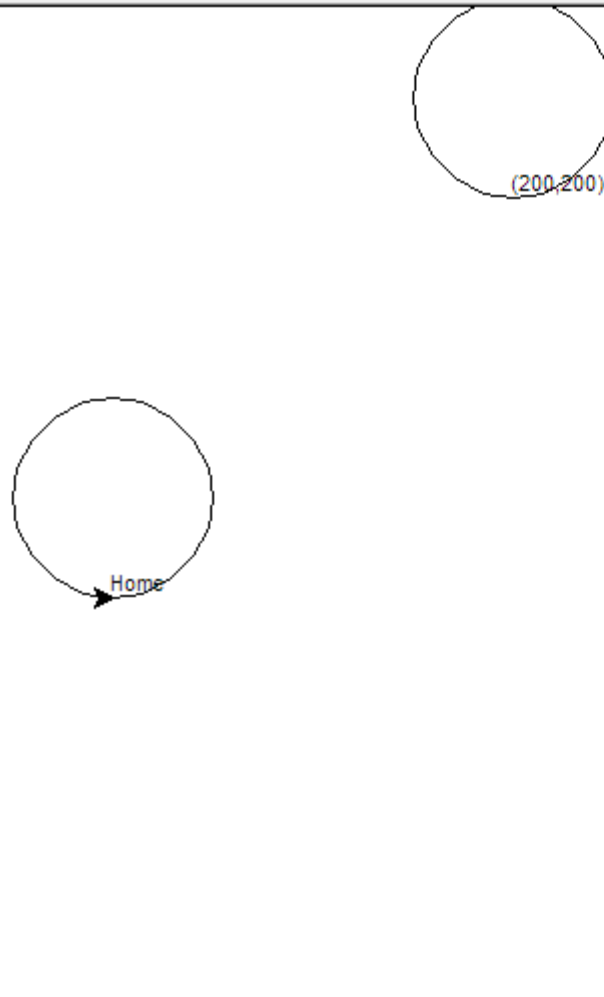

```
>>> turtle.reset()  
>>> turtle.goto(100, 100)
```



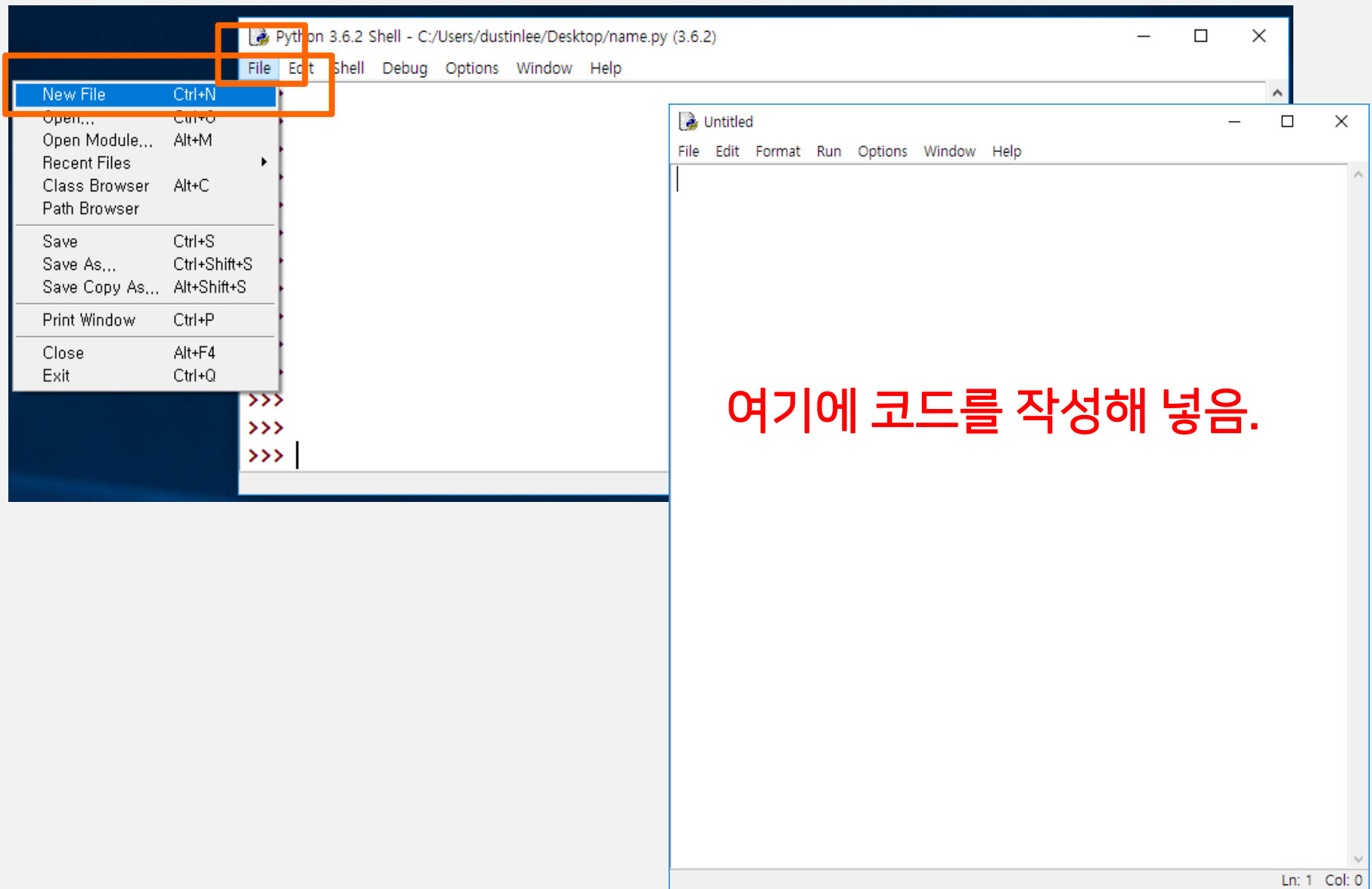
(100,100) 으로 이동한다.
거북이의 머리방향은 변함없이 여전히
오른쪽 방향.



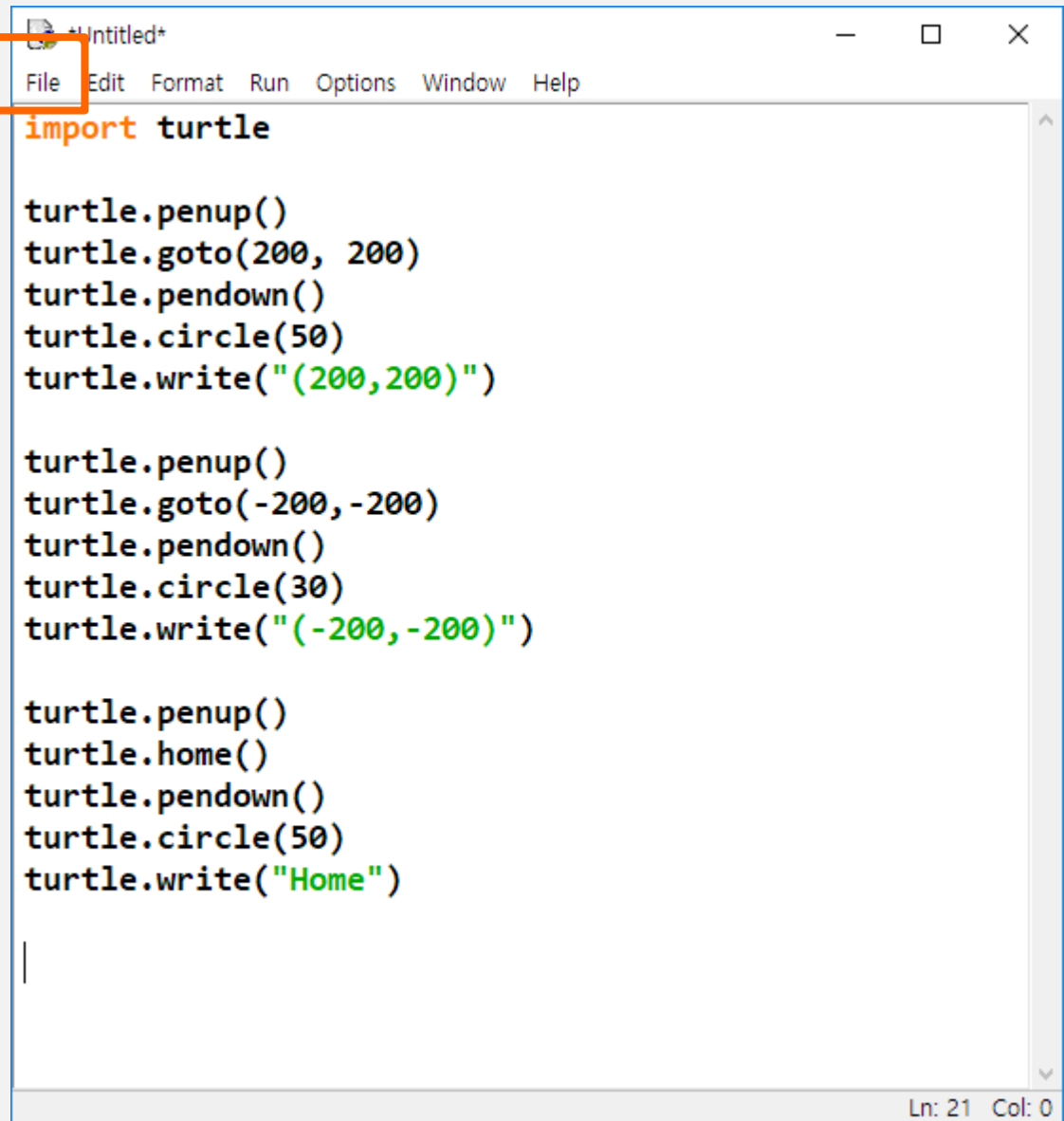
```
>>> turtle.penup()
>>> turtle.goto(200, 200)
>>> turtle.pendown()
>>> turtle.circle(50)
>>> turtle.write("(200,200)")
>>>
>>> turtle.penup()
>>> turtle.goto(-200, -200)
>>> turtle.pendown()
>>> turtle.circle(30)
>>> turtle.write("(-200,-200)")
>>>
>>> turtle.penup()
>>> turtle.home()
>>> turtle.pendown()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



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



```
import turtle

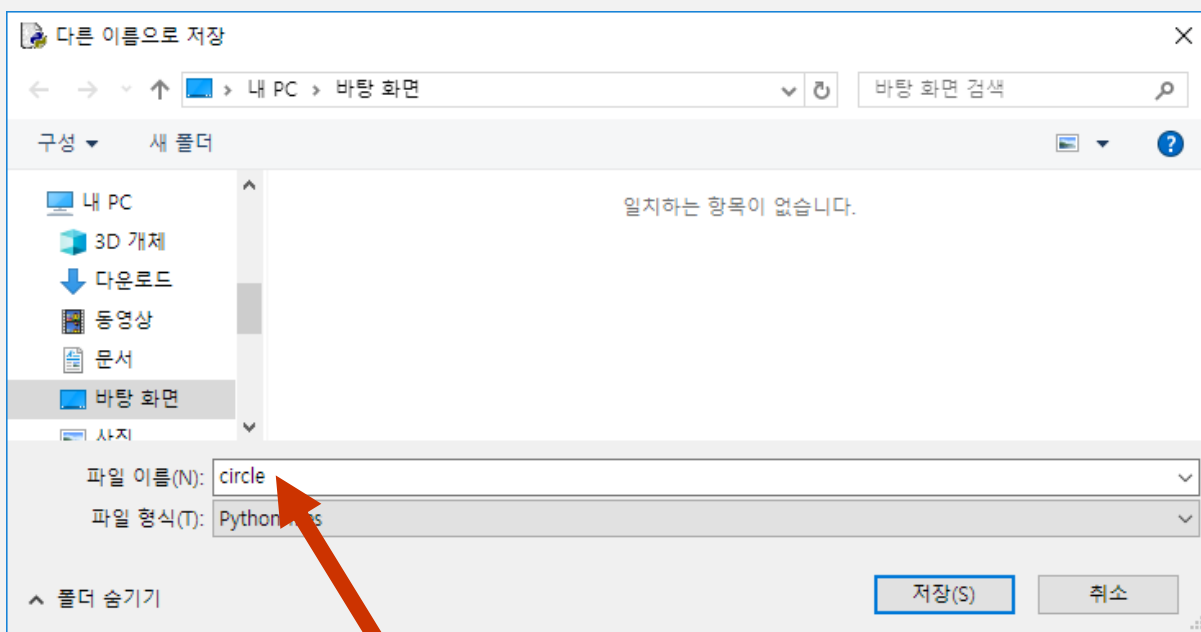
turtle.penup()
turtle.goto(200, 200)
turtle.pendown()
turtle.circle(50)
turtle.write("(200,200)")

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

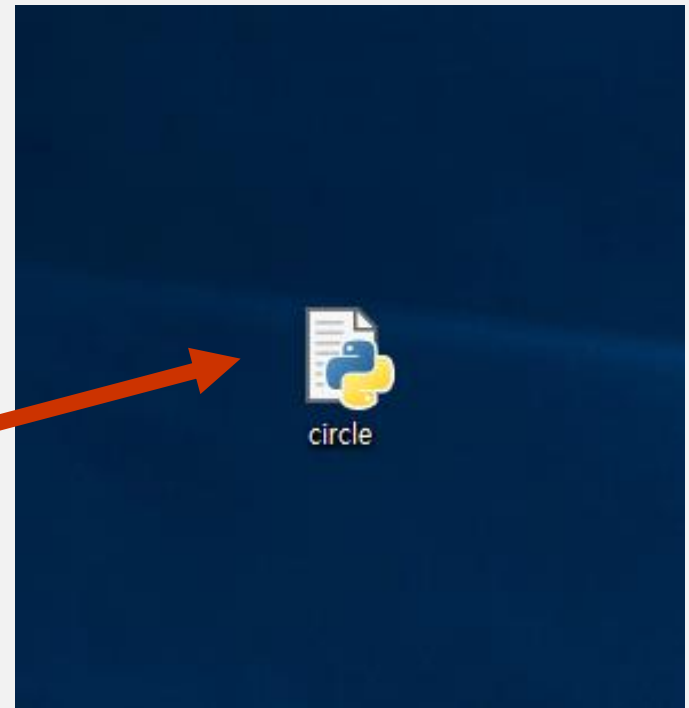
turtle.penup()
turtle.home()
turtle.pendown()
turtle.circle(50)
turtle.write("Home")

|
```

Ln: 21 Col: 0



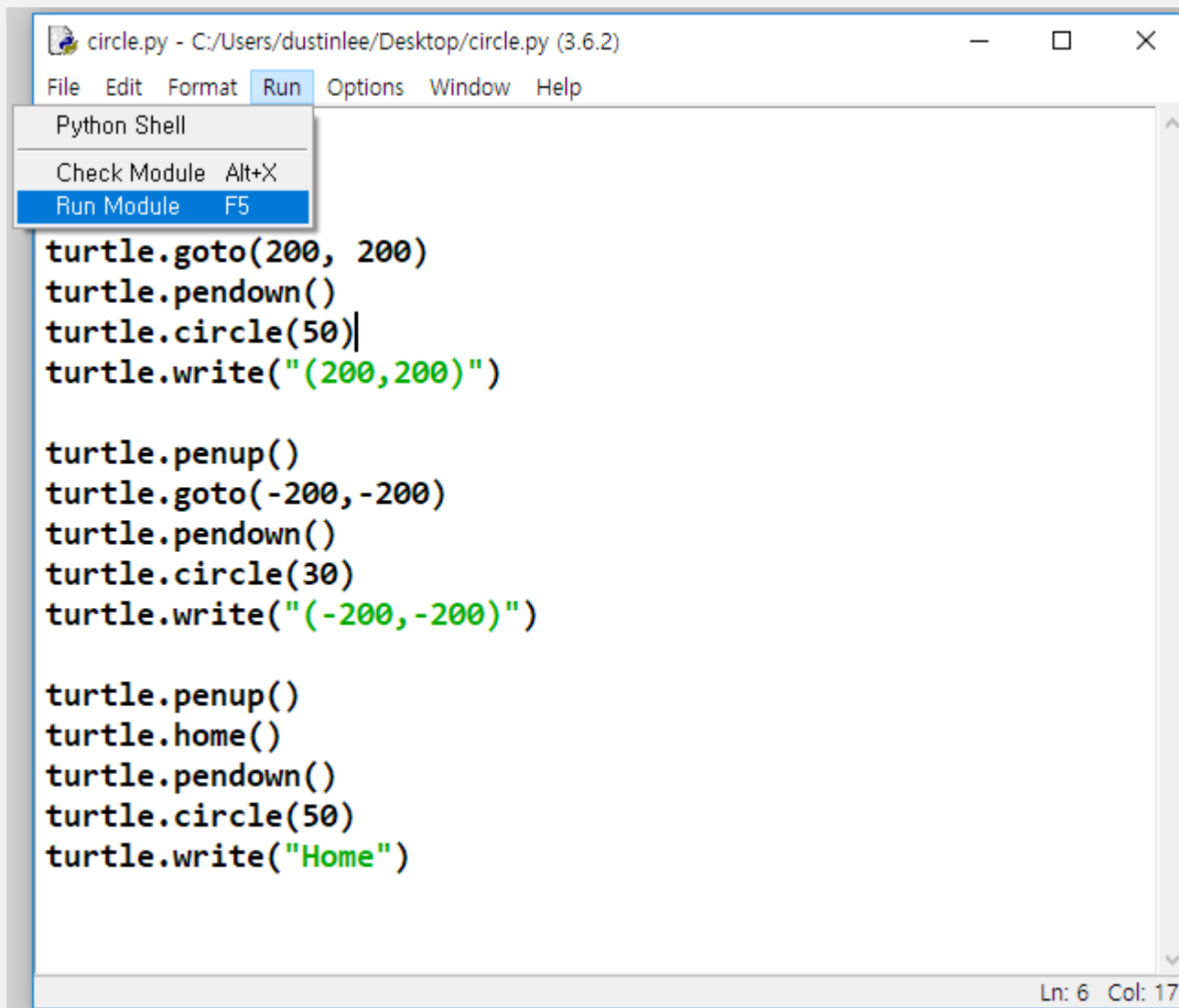
circle이라는 이름으로 바탕
화면에 저장.



바탕화면에 circle.py라는
이름의 파일이 생성됨.

프로그램 실행 방법 #1

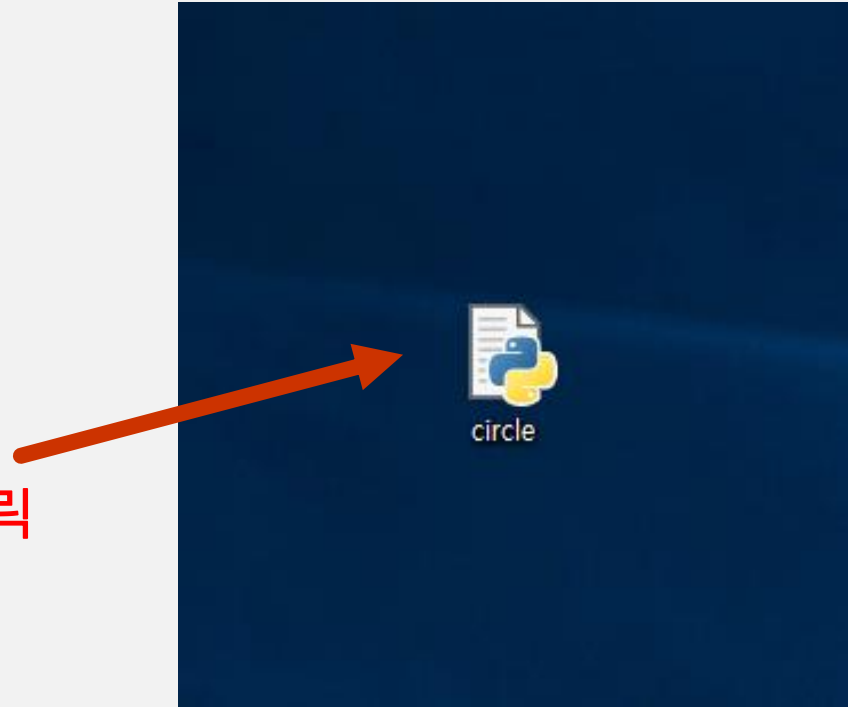
■ Run→Run Module 을 클릭 또는 단축기 F5

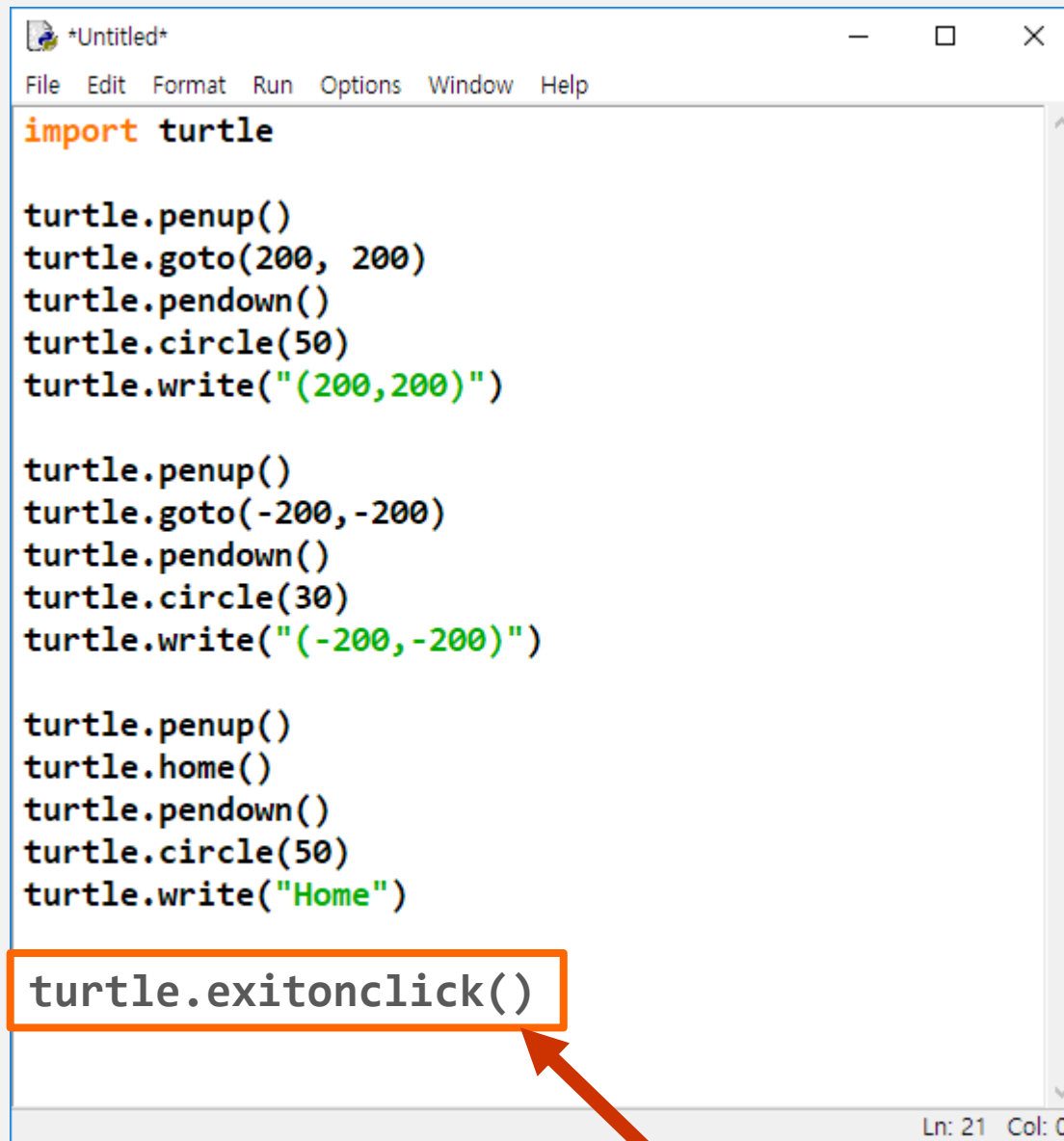


프로그램 실행 방법 #2

- 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")

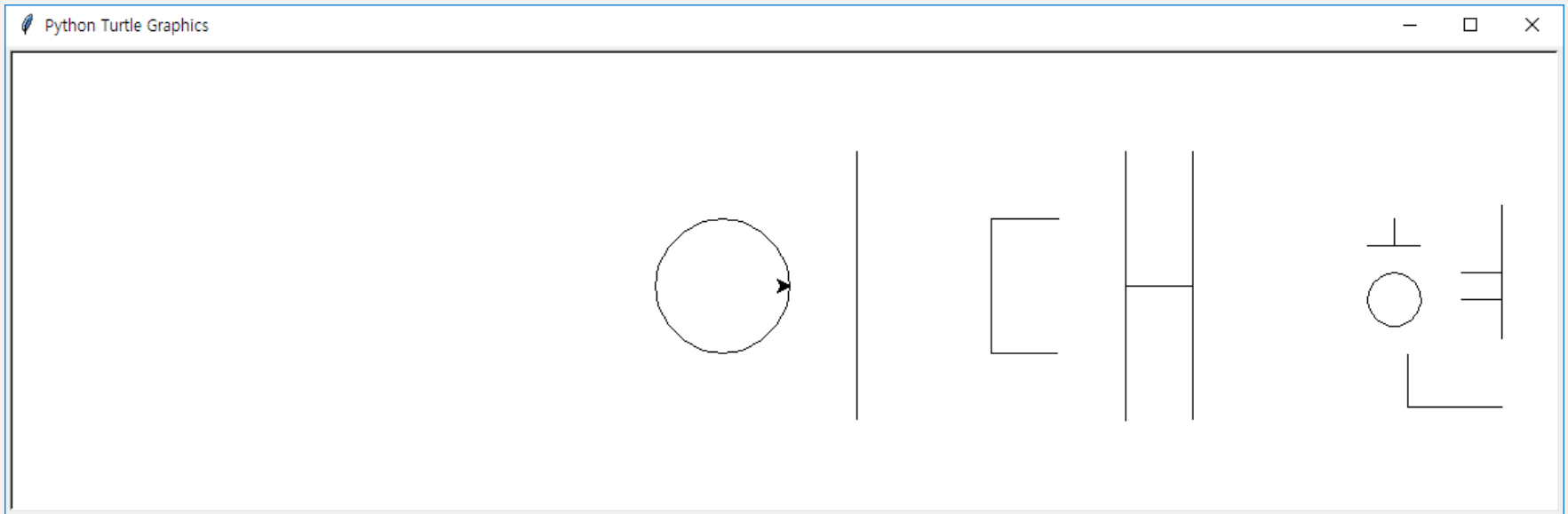
turtle.exitonclick()

Ln: 21 Col: 0
```

코드 마지막 부분에 `exitonclick()` 추가.

실습 #1-1(0.5점): 자기 이름 그리기

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



random 모듈

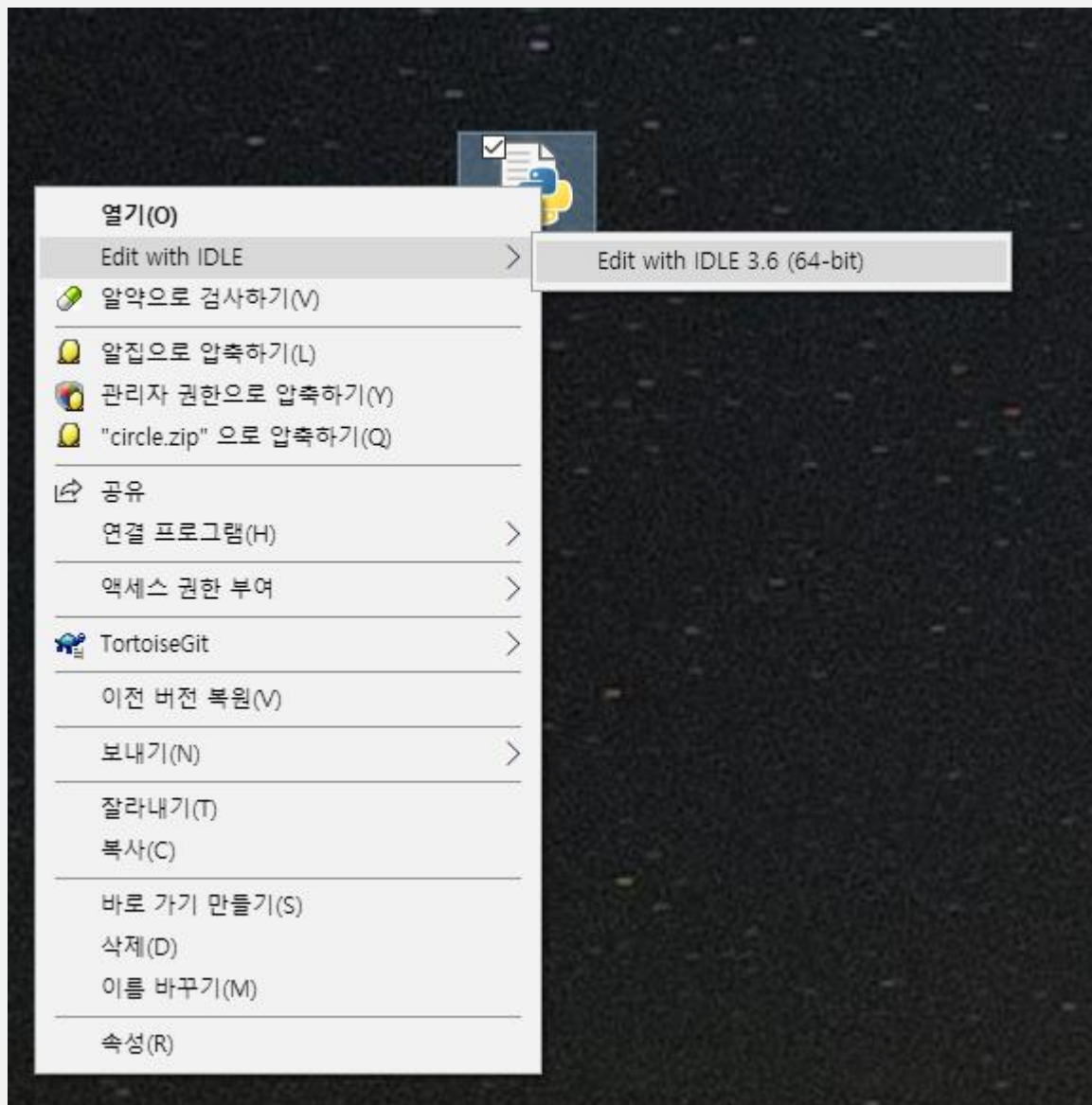
- 주사위를 던지면 어떤 수가 나올까? 무작위로 결정
- 무작위로 어떤 숫자를 뽑아내고자 할 때, random 모듈을 사용하면 된다.

A screenshot of a Python 3.6.2 Shell window. The window has a title bar with the text 'Python 3.6.2 Shell' and standard window controls (minimize, maximize, close). Below the title bar is a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Window', and 'Help'. The main area of the window contains a series of Python commands and their outputs. The commands are: '>>>', '>>> import random', '>>> random.randint(1,6)', '>>> random.randint(1,6)', '>>> random.randint(1,6)', and '>>> random.randint(1,6)'. The outputs are: '3', '4', '2', and '4'. The prompt '>>>' is shown at the end of the last line. The status bar at the bottom right shows 'Ln: 519 Col: 0'.

```
>>>
>>> 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)

마우스 오른쪽 버튼을 클릭하면, 소스코드를 직접 편집 가능.



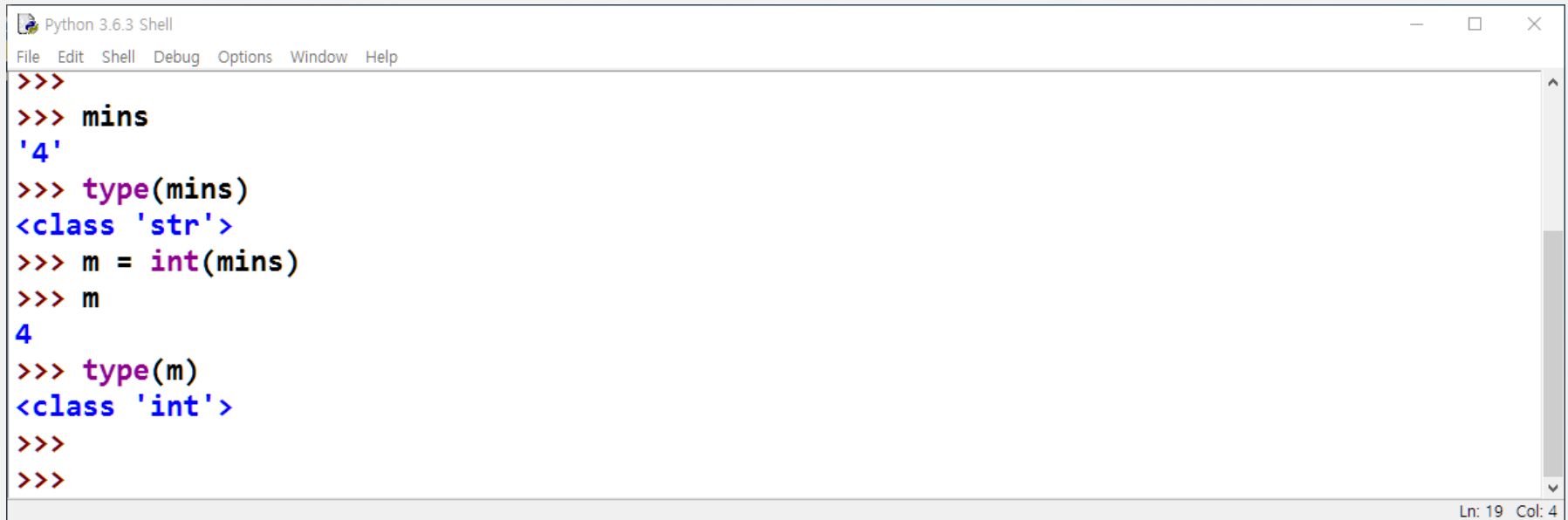
사용자로부터 입력 받기

- input 함수를 이용함.
- 사용자가 입력한 정보가 "문자열"로 되어 넘어옴.

[illegible]

자료형 변환

- mins의 값은 4가 아니고, '4'임. 즉, 정수가 아니고 문자열임.
- 이것을 정수로 바꾸기 위해서는 int() 라는 함수를 사용함.

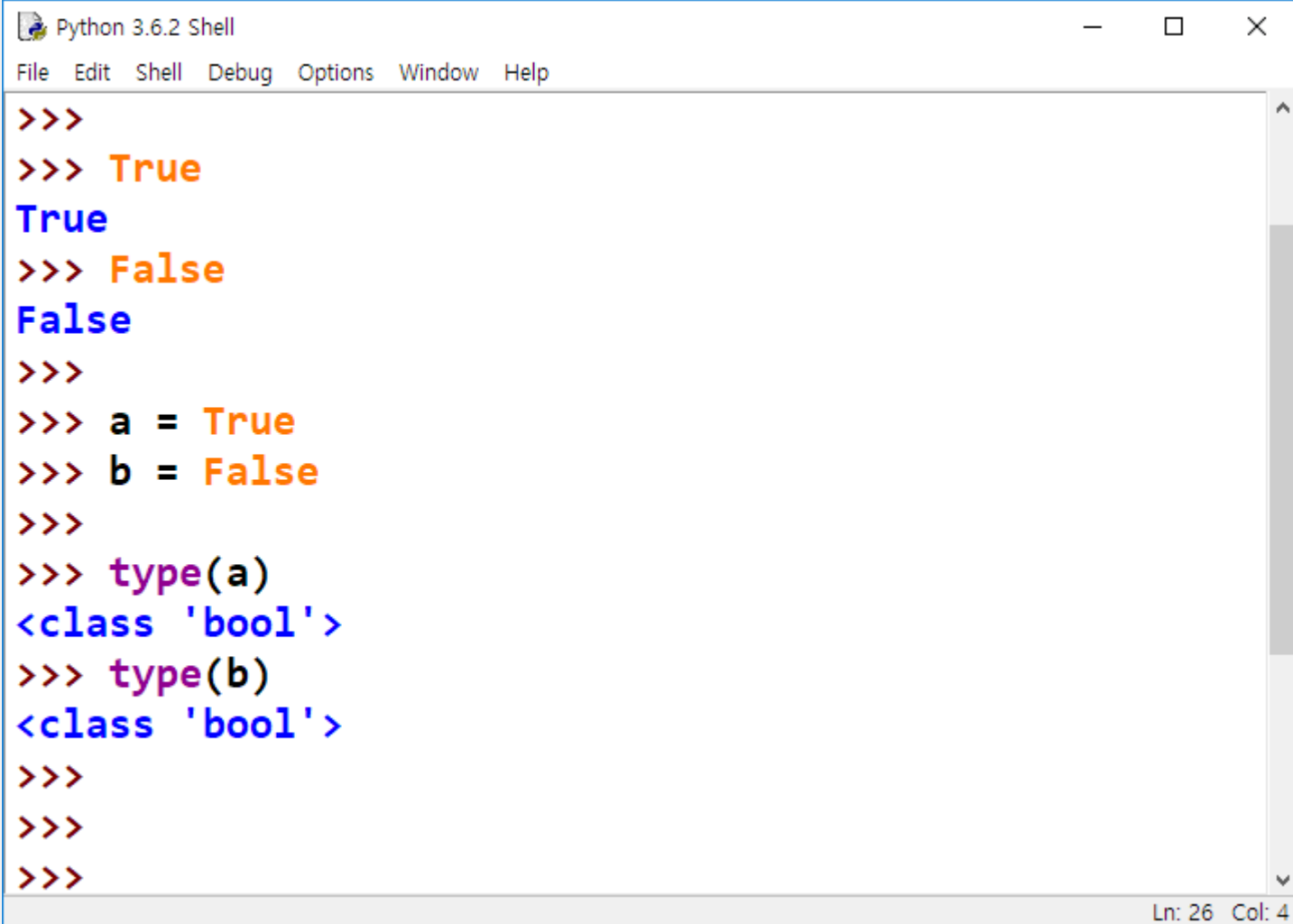


```
Python 3.6.3 Shell
File Edit Shell Debug Options Window Help
>>>
>>> mins
'4'
>>> type(mins)
<class 'str'>
>>> m = int(mins)
>>> m
4
>>> type(m)
<class 'int'>
>>>
>>>
```

Ln: 19 Col: 4

자료형: bool

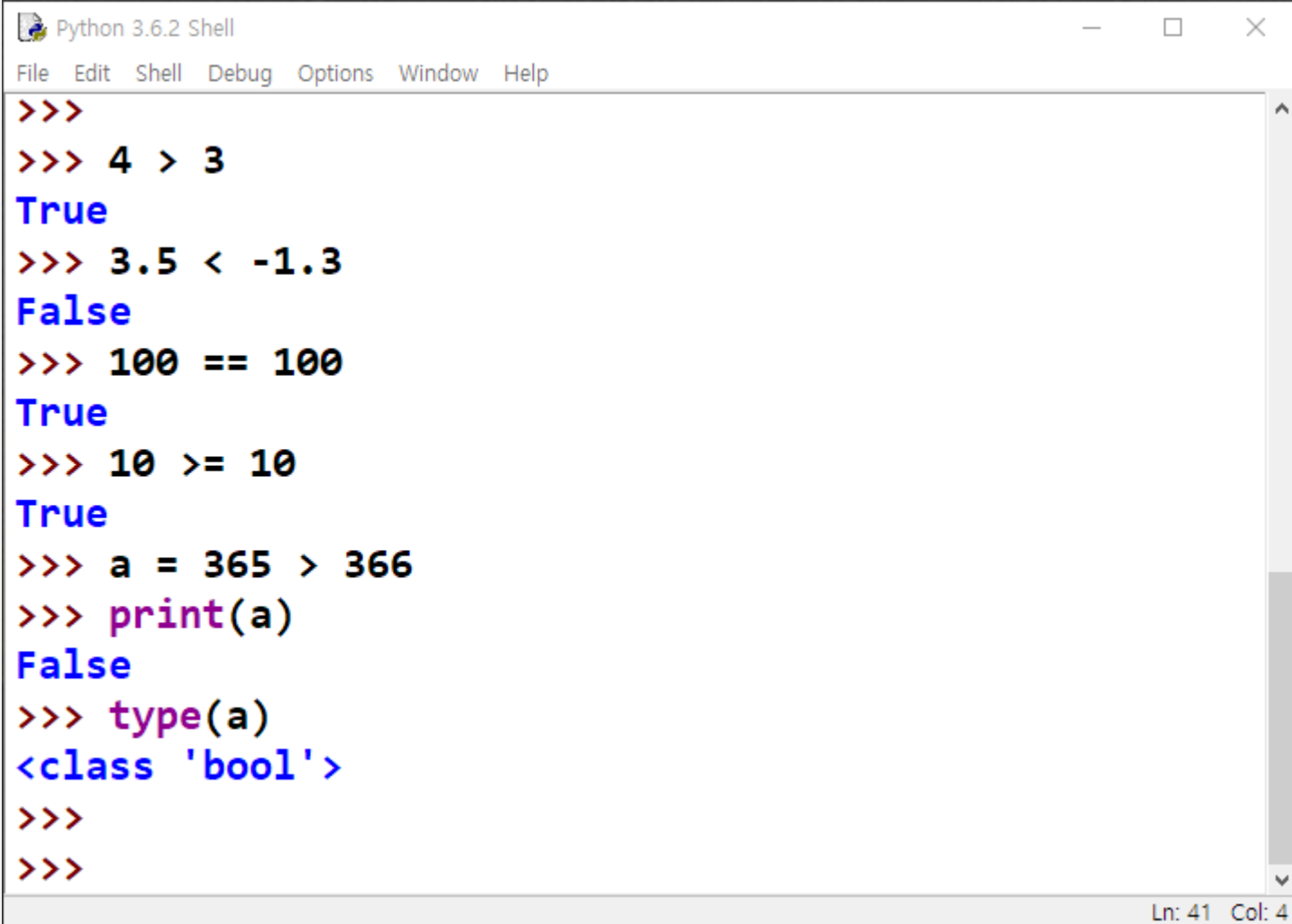
- 참(True), 또는 거짓(False)을 나타내는데 사용되는 자료형

A screenshot of a Python 3.6.2 Shell window. The window has a title bar with the text 'Python 3.6.2 Shell' and standard window controls (minimize, maximize, close). Below the title bar is a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Window', and 'Help'. The main area is a text editor showing a series of Python commands and their outputs. The commands are: '>>>', '>>> True', '>>> False', '>>> a = True', '>>> b = False', '>>> type(a)', and '>>> type(b)'. The outputs are: 'True' and 'False' (in blue), and '<class \'bool\'>' (in blue) for both type() calls. The status bar at the bottom right shows 'Ln: 26 Col: 4'.

```
Python 3.6.2 Shell
File Edit Shell Debug Options Window Help
>>>
>>> True
True
>>> False
False
>>>
>>> a = True
>>> b = False
>>>
>>> type(a)
<class 'bool'>
>>> type(b)
<class 'bool'>
>>>
>>>
>>>
Ln: 26 Col: 4
```

비교 연산(Comparison Operation)

- 두개의 값의 대소, 동일 등을 확인하는 계산.
- 결과는 참(True) 또는 거짓(False)임.

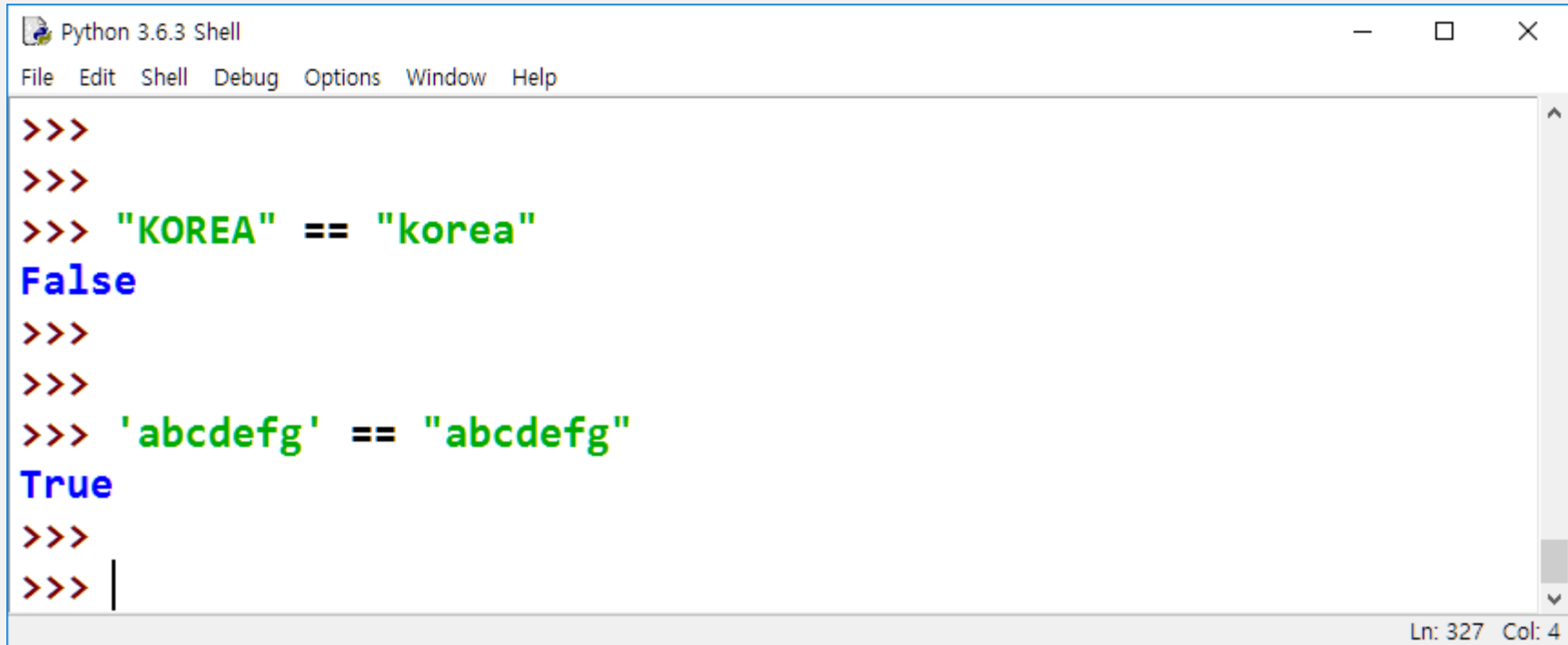
A screenshot of a Python 3.6.2 Shell window. The window has a title bar with the text 'Python 3.6.2 Shell' and standard window controls (minimize, maximize, close). Below the title bar is a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Window', and 'Help'. The main area of the window contains a series of Python commands and their outputs. The commands are: '>>>', '>>> 4 > 3', '>>> 3.5 < -1.3', '>>> 100 == 100', '>>> 10 >= 10', '>>> a = 365 > 366', '>>> print(a)', and '>>> type(a)'. The outputs are: 'True', 'False', 'True', 'True', 'False', and '<class \'bool\'>'. The window also has a status bar at the bottom right showing 'Ln: 41 Col: 4'.

```
Python 3.6.2 Shell
File Edit Shell Debug Options Window Help
>>>
>>> 4 > 3
True
>>> 3.5 < -1.3
False
>>> 100 == 100
True
>>> 10 >= 10
True
>>> a = 365 > 366
>>> print(a)
False
>>> type(a)
<class 'bool'>
>>>
>>>
Ln: 41 Col: 4
```


비교 연산 기호

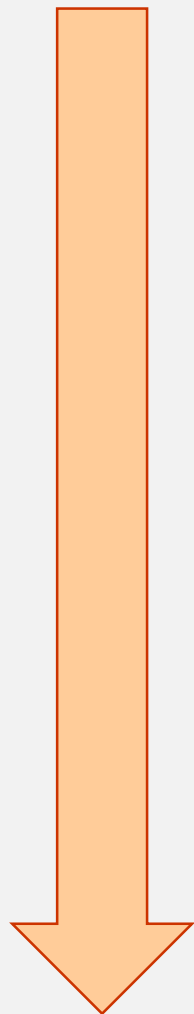
기호	뜻
<	작다
<=	작거나 같다
==	같다
>=	크거나 같다
>	크다
!=	다르다

문자열의 비교

A screenshot of a Python 3.6.3 Shell window. The window has a title bar with the text "Python 3.6.3 Shell" 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 contains a Python REPL session. The prompt is ">>>". The first two prompts are empty. The third prompt is ">>> \"KOREA\" == \"korea\"", followed by the output "False" in blue. The fourth prompt is ">>> \"\"". The fifth prompt is ">>> 'abcdefg' == \"abcdefg\"", followed by the output "True" in blue. The sixth prompt is ">>>". The seventh prompt is ">>> |", where the vertical bar is at the end of the line. A vertical scrollbar is on the right side of the window. At the bottom right of the window, the status bar shows "Ln: 327 Col: 4".

```
Python 3.6.3 Shell
File Edit Shell Debug Options Window Help
>>>
>>>
>>> "KOREA" == "korea"
False
>>>
>>>
>>> 'abcdefg' == "abcdefg"
True
>>>
>>> |
Ln: 327 Col: 4
```

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
```

문법: 조건문 (Conditional Statement)

- 조건을 검사하여, 그 결과에 따라 처리를 하는 문장

```
if (age >= 60):  
    print(age)  
    print("you are very old")
```



만약 age가 60 이상이면, age의 값을 출력하고, 이어서 "you are very old"라는 문자열을 출력하라.
age 가 60보다 작으면? 아무것도 하지 않음.

이 조건이 참(True)이면,

if (age >= 60):

print(age)

print("you are very old")

여기 블록에 적힌 대로 실행하라.

들여쓰기(indentation)

*** 매우 중요 ***

일반적으로 공백4개씩

조건이 참이면, 들여쓰기된 블록을 실행함.

test_age.py

```
test_age.py - C:/Users/dustinlee/Desktop/test_age.py (3.6.3)
File Edit Format Run Options Window Help

age = 10
if (age >= 60):
    print(age)
    print("you are very old")

Ln: 4 Col: 4
```



test_age.py

```
test_age.py - C:/Users/dustinlee/Desktop/test_age.py (3.6.3)
File Edit Format Run Options Window Help

age = 10
if (age >= 60):
    print(age)
print("you are very old")

Ln: 4 Col: 0
```

문법: 조건문 (Conditional Statement) 확장형

```
if (age >= 60):  
    print(age)  
    print("you are very old")  
else:  
    print(age)  
    print("you are young")
```

문법: 조건문 (Conditional Statement) 확장형

```
if (age >= 60):  
    print(age)  
    print("you are very old")  
elif (age <= 20):  
    print(age)  
    print("you are very young")  
else:  
    print(age)  
    print("you are young")
```


버스 요금 계산기를 만들어 보자.

■ 버스 요금

- 미취학 만 7살 미만 : 공짜
- 초등학생 : 450원
- 중고등학생: 720원
- 성인: 1200원
- 65세 이상: 공짜

bus_fare.py

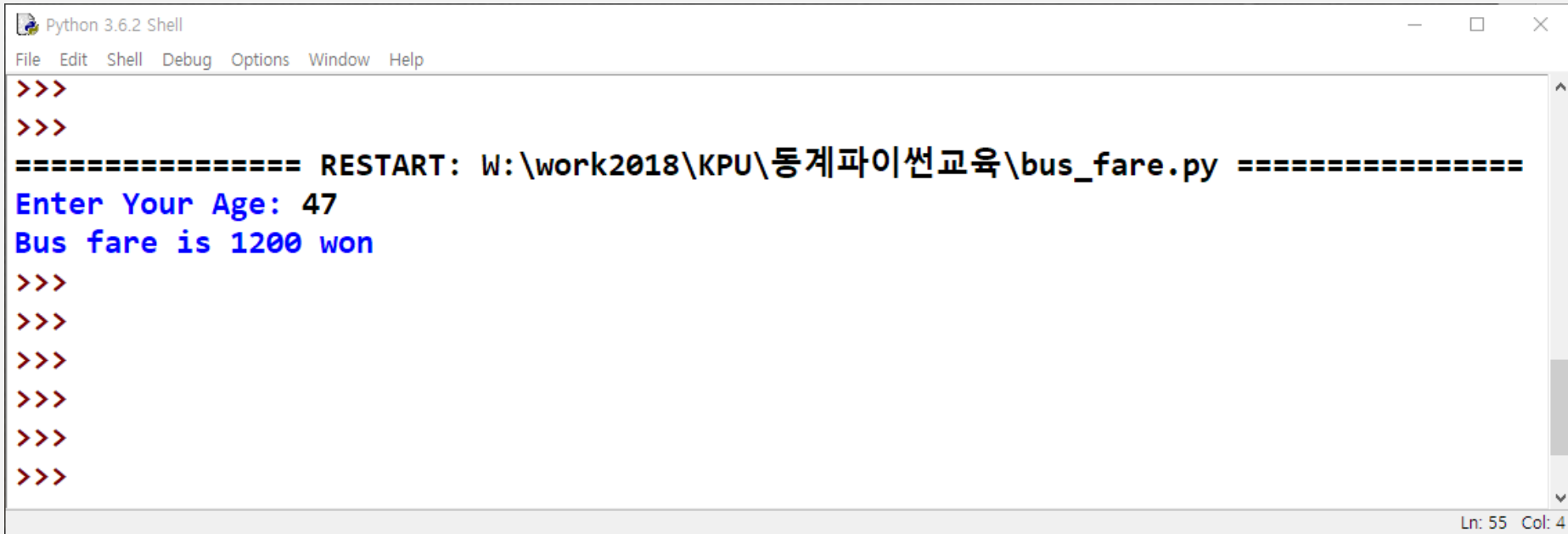
```
bus_fare.py - C:/Users/dustinlee/Desktop/bus_fare.py (3.6.3)
File Edit Format Run Options Window Help

age_str = input("Enter Your Age: ")
age = int(age_str)
if age <= 6:
    print("Bus is free")
elif age <= 12:
    print("Bus fare is 450 won")
elif age <= 18:
    print("Bus fare is 720 won")
elif age <= 64:
    print("Bus fare is 1200 won")
else:
    print("Bus is free")

Ln: 15 Col: 0
```

에디터 화면에서, F5를 눌러서 실행하면,

- IDLE 화면에서, RESTART 가 출력되고, 프로그램 실행이 시작됨.
- 불편한 점은?



The screenshot shows a Python 3.6.2 Shell window with a menu bar (File, Edit, Shell, Debug, Options, Window, Help). The command prompt shows two empty lines, followed by a restart message: "===== RESTART: W:\work2018\KPU\통계파이썬교육\bus_fare.py =====". Below this, the program prompts "Enter Your Age: 47" and outputs "Bus fare is 1200 won". There are several more empty lines. The status bar at the bottom right indicates "Ln: 55 Col: 4".

```
>>>  
>>>  
===== RESTART: W:\work2018\KPU\통계파이썬교육\bus_fare.py =====  
Enter Your Age: 47  
Bus fare is 1200 won  
>>>  
>>>  
>>>  
>>>  
>>>  
>>>  
>>>  
Ln: 55 Col: 4
```

문법: while 반복문 (Iteration Statement)

- 어떤 조건을 만족하는 동안, 계속해서 반복적으로 실행하는 문장.

```
while <조건문>:  
    <수행할 문장1>  
    <수행할 문장2>  
    <수행할 문장3>  
    ...
```

```
import turtle
```

```
count = 10
```

```
while (count > 0):  
    turtle.forward(100)  
    turtle.left(30)  
    count = count - 1
```



count가 0 보다 크면 계속해서 반복한다. 뭘? (turtle을 앞으로 100 이동,
그리고 왼쪽으로 30도 회전, 그리고 count 값 하나 감소)

```
import turtle
```

```
count = 10
```

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

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

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

```
    count = count - 1
```

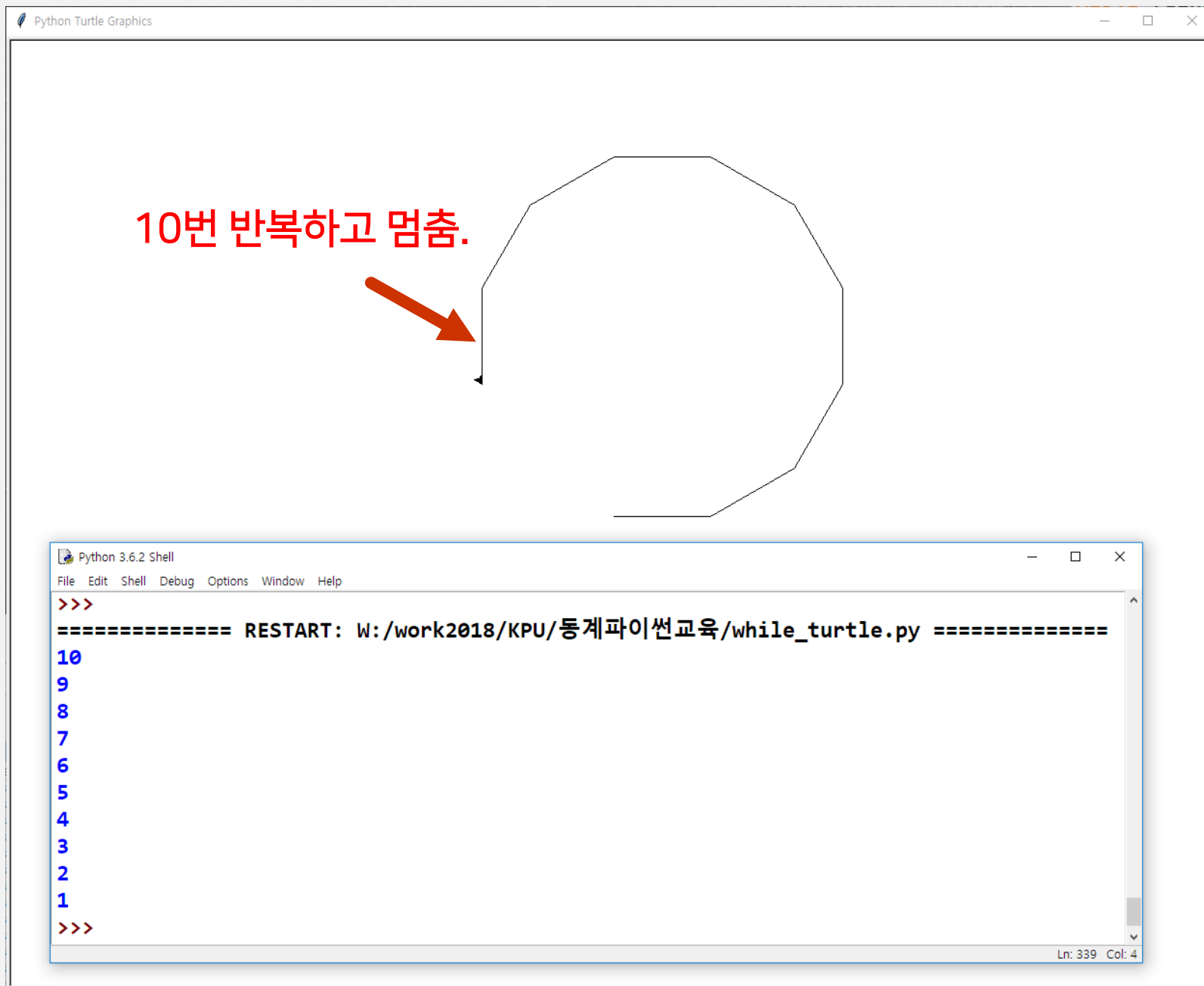
이 조건이 참(True)인 동안



들여쓰기(indentation)

*** 매우 중요 ***

여기 블록을 반복적으로 실행한다.



실습 #1-2(0.5점): 모눈 그리기(길이 500, 간격 100)

