

# 2D 게임 프로그래밍

- 반드시 카메라를 ON 하고 !
- 입장 이름은 "학번 이름"으로 설정 !
- 미리 수업 git 서버에서 자료를 Pull 해서 준비 !

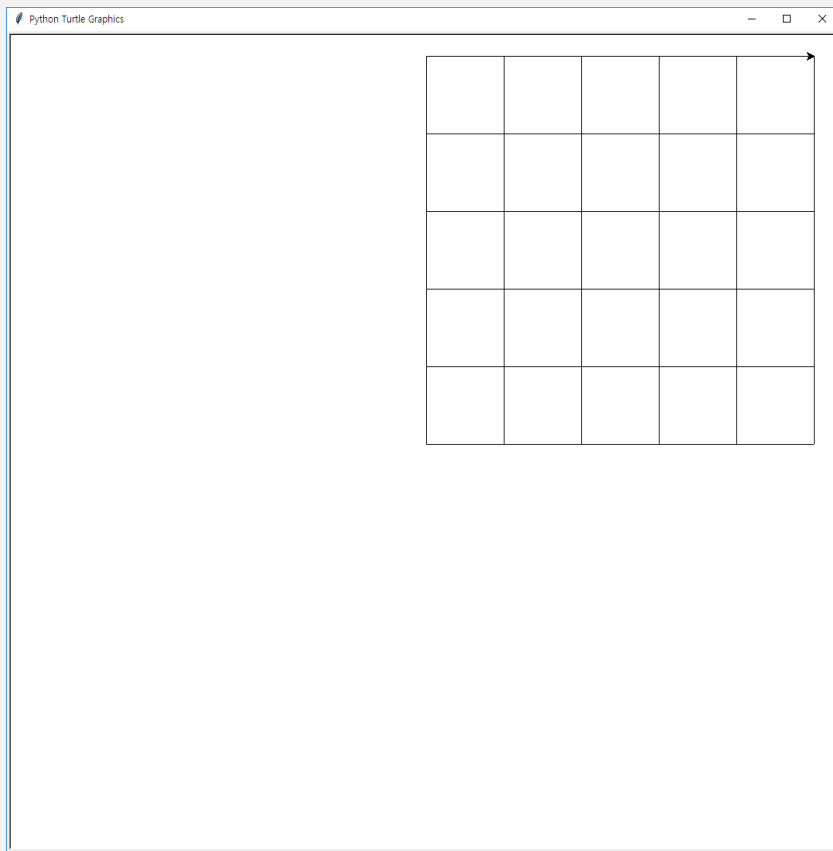
# Lecture #3. 파이썬 기초 (3)

2D 게임 프로그래밍

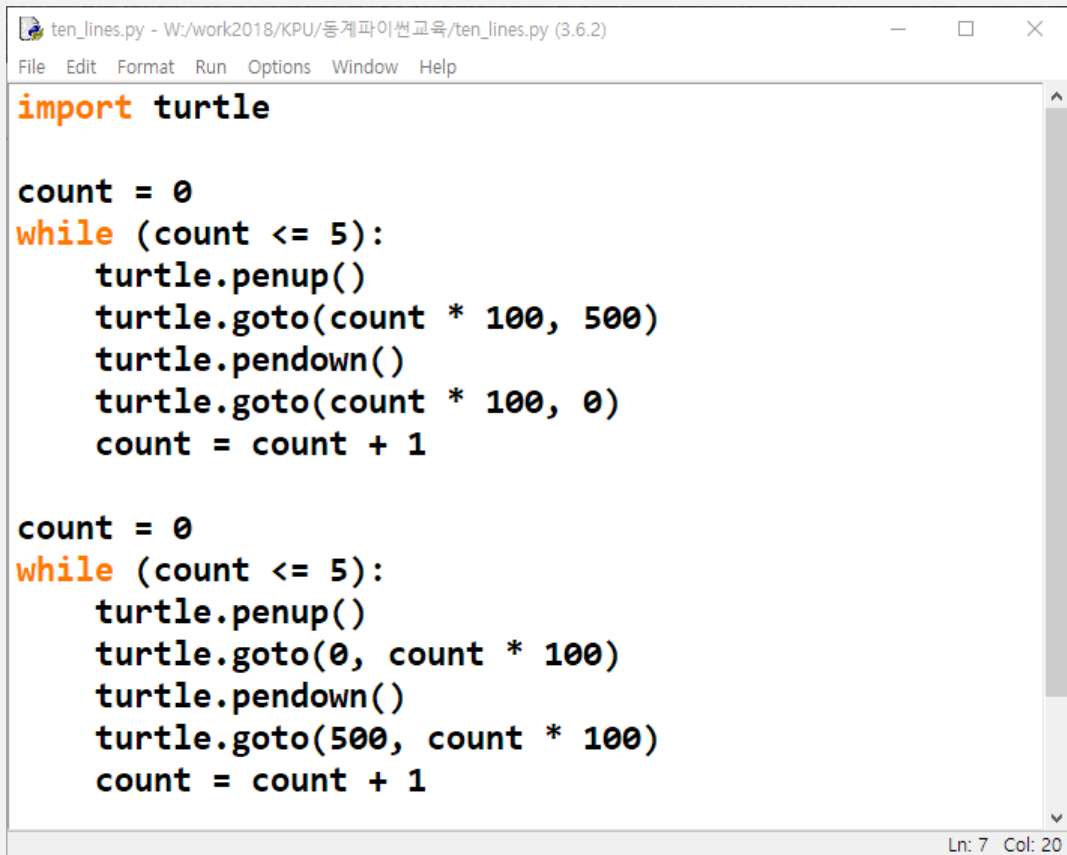
이대현 교수

# 모눈 그리기(길이 500, 간격 100)

---



# 하나의 답안



```
ten_lines.py - W:/work2018/KPU/등계파이썬교육/ten_lines.py (3.6.2)
File Edit Format Run Options Window Help

import turtle

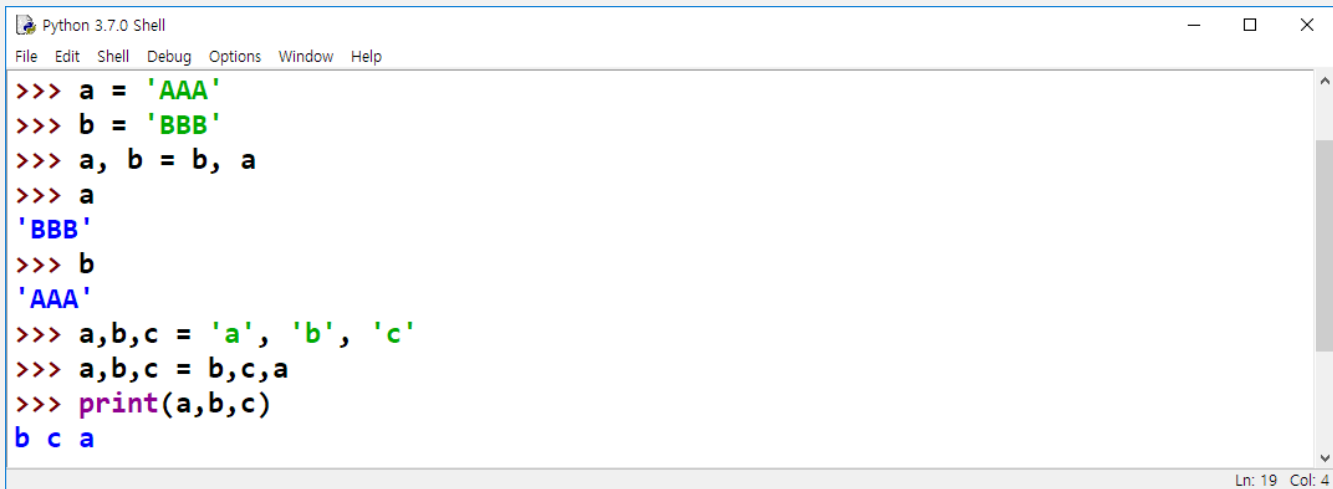
count = 0
while (count <= 5):
    turtle.penup()
    turtle.goto(count * 100, 500)
    turtle.pendown()
    turtle.goto(count * 100, 0)
    count = count + 1

count = 0
while (count <= 5):
    turtle.penup()
    turtle.goto(0, count * 100)
    turtle.pendown()
    turtle.goto(500, count * 100)
    count = count + 1

Ln: 7 Col: 20
```

# 다중 대입

---



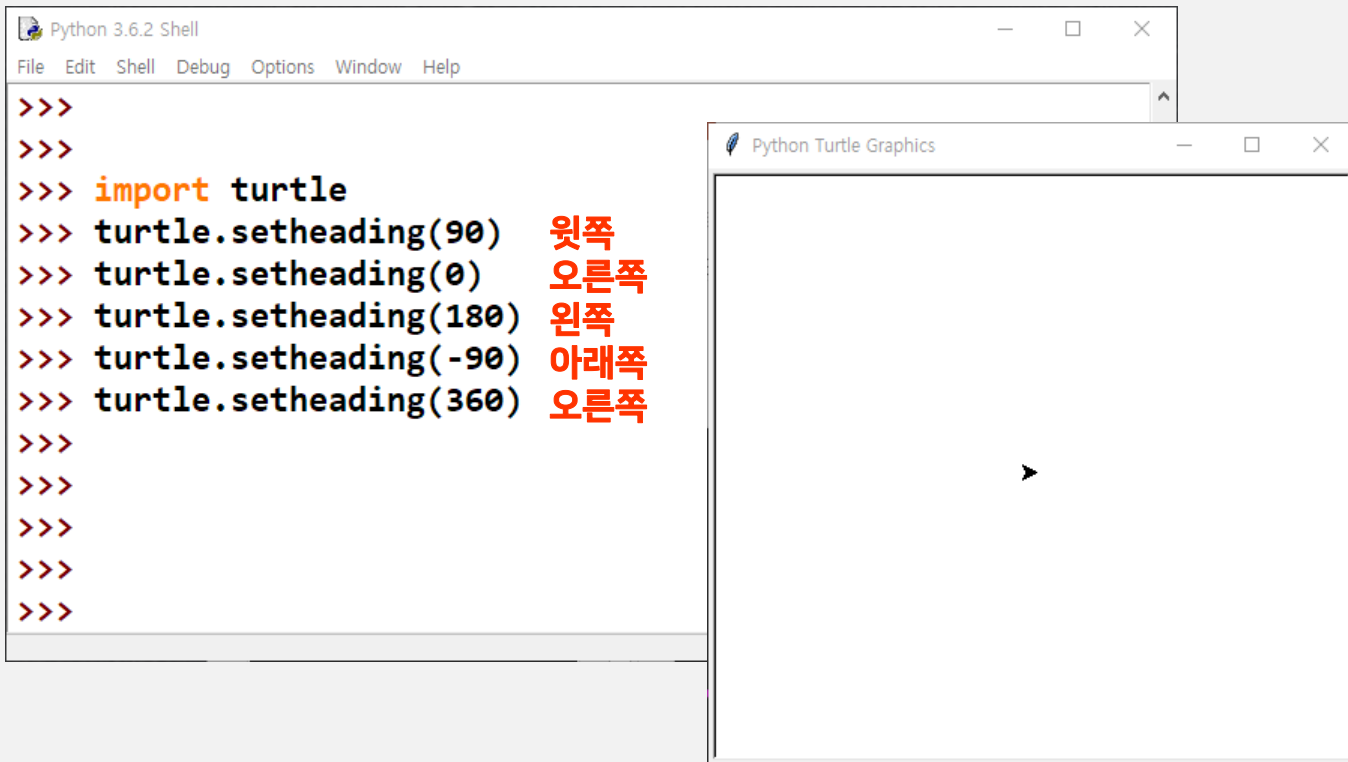
```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help

>>> a = 'AAA'
>>> b = 'BBB'
>>> a, b = b, a
>>> a
'BBB'
>>> b
'AAA'
>>> a,b,c = 'a', 'b', 'c'
>>> a,b,c = b,c,a
>>> print(a,b,c)
b c a
Ln: 19 Col: 4
```

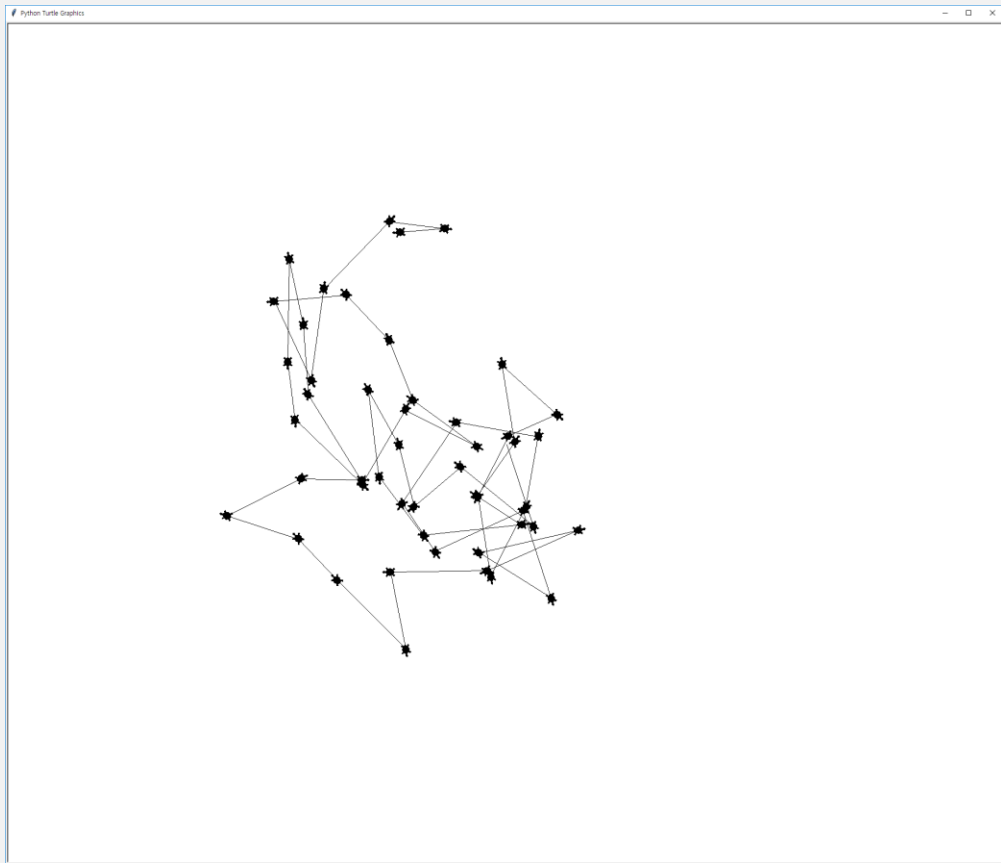
The image shows a Python 3.7.0 Shell window with a menu bar (File, Edit, Shell, Debug, Options, Window, Help). The code demonstrates variable assignment and swapping. First, 'a' is assigned 'AAA' and 'b' is assigned 'BBB'. Then, 'a' and 'b' are swapped using tuple assignment. Next, 'a', 'b', and 'c' are assigned 'a', 'b', and 'c' respectively. Finally, 'a', 'b', and 'c' are swapped using tuple assignment, and the result is printed, showing 'b c a'.

# 거북이의 방향 설정

## `turtle.setheading(각도)`



# 술취한 거북이?



# random 모듈

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

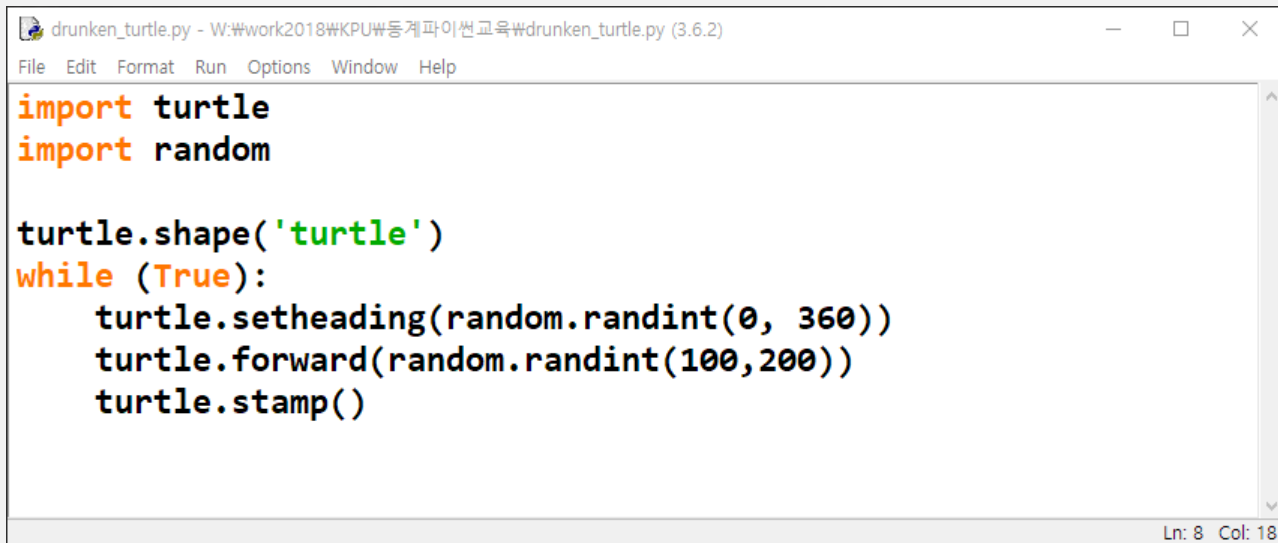
A screenshot of a Python 3.6.2 Shell window. The window has a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Window', and 'Help'. The main area shows a series of commands and their outputs: three empty prompts '>>>', followed by 'import random', then five calls to 'random.randint(1,6)' which return the values 3, 4, 2, 4, and 1 respectively. 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)**



## drunken\_turtle.py



The image shows a screenshot of a Python IDE window titled "drunken\_turtle.py - W:\\work2018\\WKPU\\등계파이썬교육\\drunken\_turtle.py (3.6.2)". The window has a menu bar with "File", "Edit", "Format", "Run", "Options", "Window", and "Help". The code editor contains the following Python code:

```
import turtle
import random

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

The status bar at the bottom right indicates "Ln: 8 Col: 18".

# 문법: for 반복문

---

- 집합적 데이터의 각 요소를 하나씩 꺼내서 반복적으로 처리

```
for 변수 in 리스트(또는 튜플, 문자열):  
    수행할 문장1  
    수행할 문장2  
    ...
```

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

>>> for n in [1,3,4,5]:
    print(n)

1
3
4
5

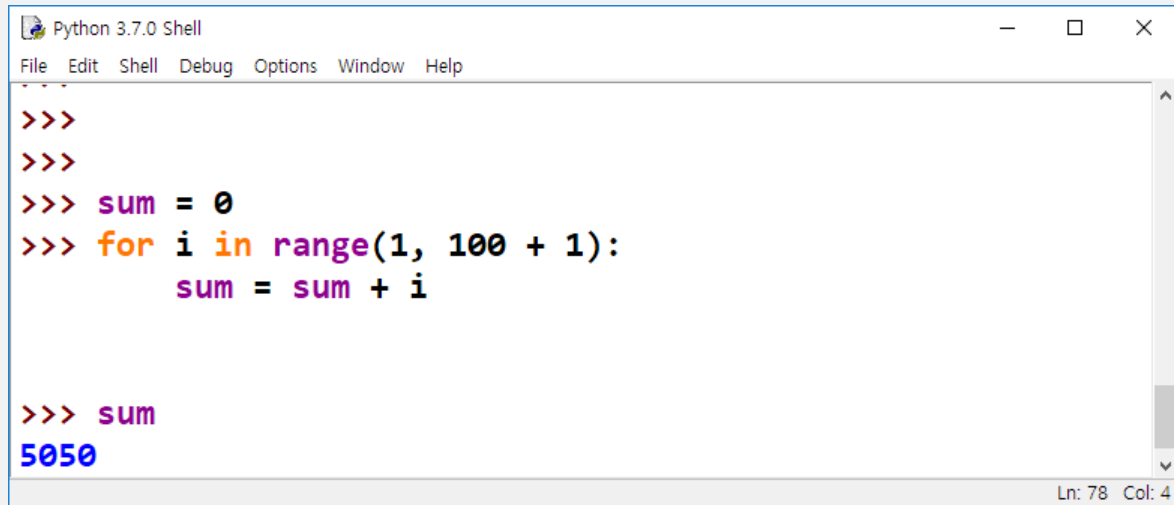
>>> for c in "Lee DAE HYUN":
    print(c)

L
e
e

D
A
E

H
Y
U
N

Ln: 33 Col: 4
```

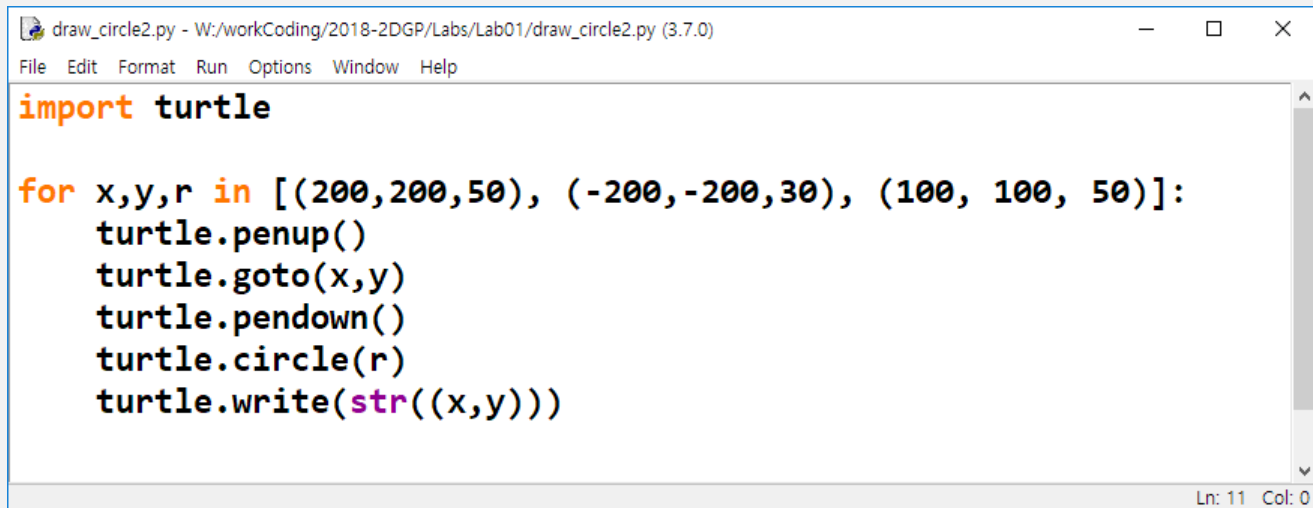


A screenshot of a Python 3.7.0 Shell window. The window has a title bar with the text "Python 3.7.0 Shell" and standard window controls (minimize, maximize, close). Below the title bar is a menu bar with the following items: File, Edit, Shell, Debug, Options, Window, and Help. The main area of the window contains a Python script. The script starts with two empty lines, followed by `sum = 0`, then a `for` loop: `for i in range(1, 100 + 1):` with an indented line `sum = sum + i`. After the loop, there is a line `sum`. The output of the script is `5050`. The status bar at the bottom right of the window shows "Ln: 78 Col: 4".

```
>>>
>>>
>>> sum = 0
>>> for i in range(1, 100 + 1):
        sum = sum + i

>>> sum
5050
```

Ln: 78 Col: 4

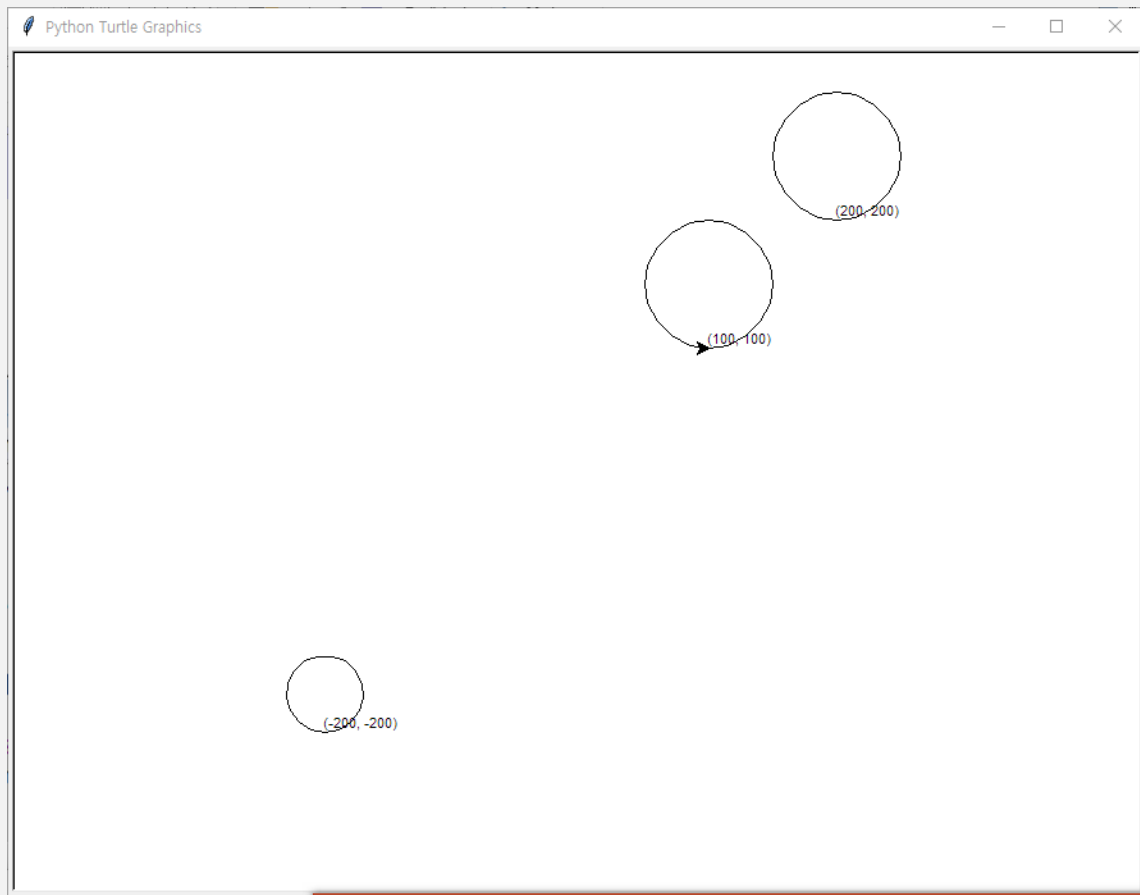


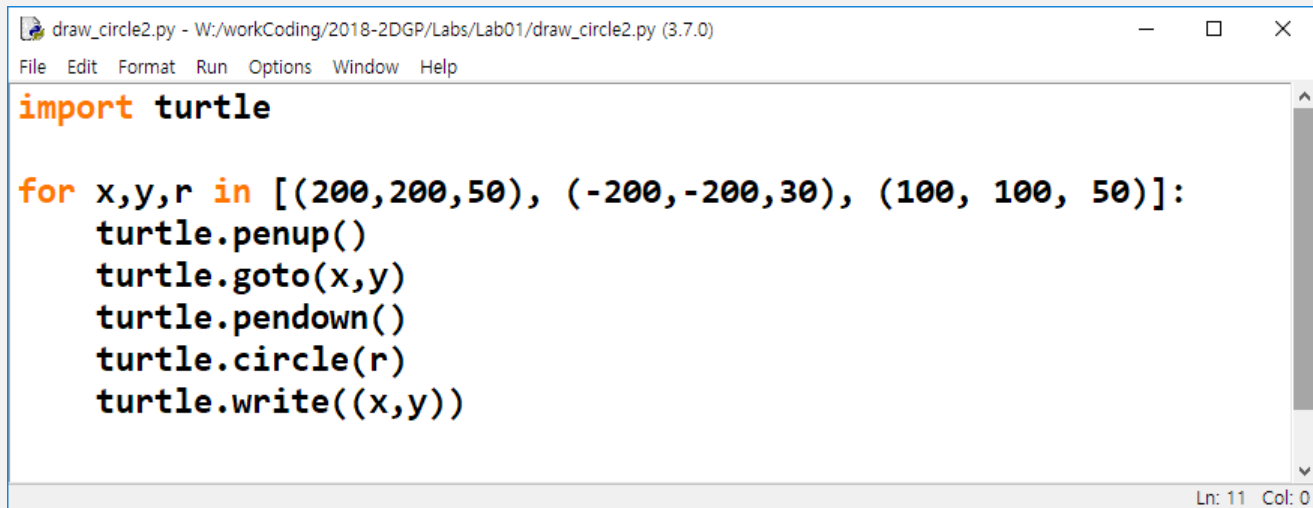
The image shows a screenshot of a Python IDE window titled "draw\_circle2.py - W:/workCoding/2018-2DGP/Labs/Lab01/draw\_circle2.py (3.7.0)". The window has a menu bar with "File", "Edit", "Format", "Run", "Options", "Window", and "Help". The main text area contains the following Python code:

```
import turtle

for x,y,r in [(200,200,50), (-200,-200,30), (100, 100, 50)]:
    turtle.penup()
    turtle.goto(x,y)
    turtle.pendown()
    turtle.circle(r)
    turtle.write(str((x,y)))
```

The status bar at the bottom right indicates "Ln: 11 Col: 0".



A screenshot of a Python IDE window titled "draw\_circle2.py - W:/workCoding/2018-2DGP/Labs/Lab01/draw\_circle2.py (3.7.0)". The window has a menu bar with "File", "Edit", "Format", "Run", "Options", "Window", and "Help". The main text area contains the following Python code:

```
import turtle

for x,y,r in [(200,200,50), (-200,-200,30), (100, 100, 50)]:
    turtle.penup()
    turtle.goto(x,y)
    turtle.pendown()
    turtle.circle(r)
    turtle.write((x,y))
```

A vertical scrollbar is on the right side of the text area. The status bar at the bottom right shows "Ln: 11 Col: 0".

```
draw_circle2.py - W:/workCoding/2018-2DGP/Labs/Lab01/draw_circle2.py (3.7.0)
File Edit Format Run Options Window Help
import turtle

for x,y,r in [(200,200,50), (-200,-200,30), (100, 100, 50)]:
    turtle.penup()
    turtle.goto(x,y)
    turtle.pendown()
    turtle.circle(r)
    turtle.write((x,y))
Ln: 11 Col: 0
```

# 함수(function)

---

- 수학에서 함수는, 어떤 수식을 정의한 것.

$$f(a, b) = a + b$$

$$f(3,4)=?$$

$$f(100,10)=?$$



# 프로그래밍에서 함수(function)란?

---

- 어떤 특정한 일을 처리하는 기능을 모아놓은 것, 수학적인 함수도 구현 가능.
- 일반적으로 라이브러리, 모듈은 여러 개의 함수들로 구성됨.
- 프로그래머는 자기만의 함수를 만들 수 있음.
- 함수의 이름은 그 함수의 기능을 정확히 나타내는 것이 좋음.

```
turtle.forward(100)  
turtle.right(90)  
turtle.undo()
```

# 문법: 함수 정의 - 함수를 만들기

---

```
def 함수명(매개변수):  
    <수행할 문장1>  
    <수행할 문장2>  
    ...
```

# add 함수 만들기

---

a와 b 두개의 값을 받아서,

```
def add(a, b):  
    sum = a + b  
    return sum
```

a와 b를 더해, sum을 계산

sum의 값을 되돌려줌 (return)

함수 정의임을 뜻함.

함수 이름

인수: 외부에서 전달되는 값

```
def add(a, b):  
    sum = a + b  
    return sum
```

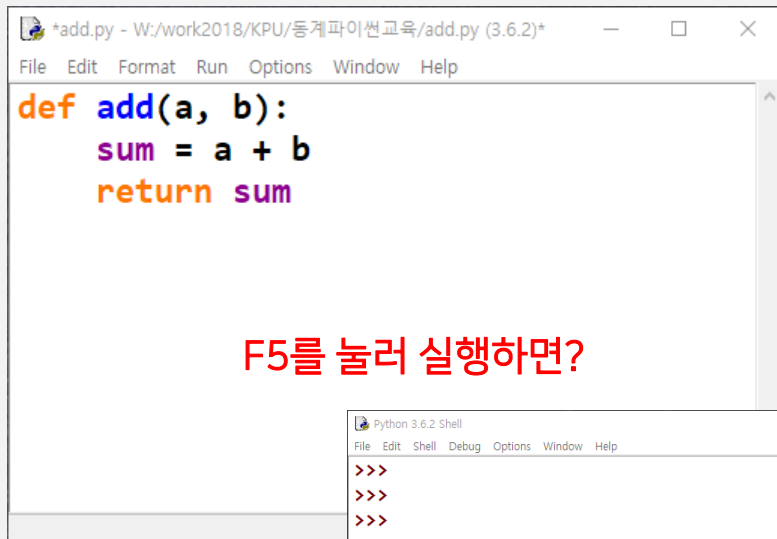
들여쓰기(indentation)

\*\*\* 매우 중요 \*\*\*

함수가 호출될 때 실행됨.

# 함수 정의, 그 자체로는 실행되지 않음.

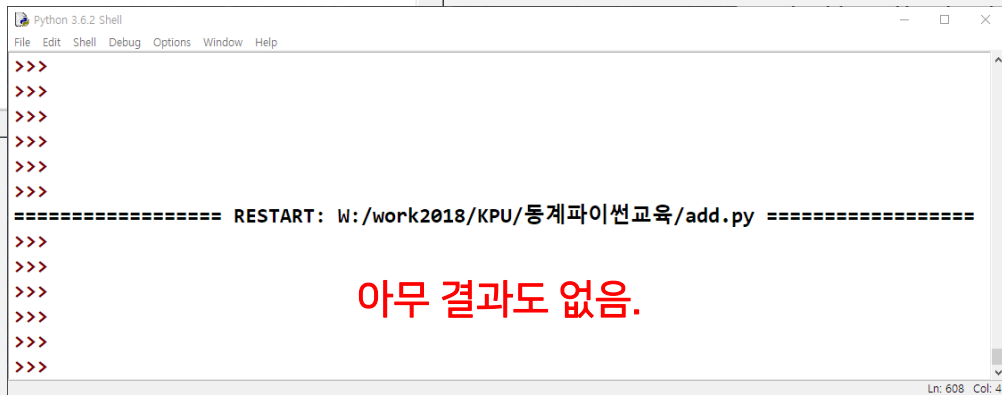
add.py



```
*add.py - W:/work2018/KPU/동계파이썬교육/add.py (3.6.2)*
File Edit Format Run Options Window Help

def add(a, b):
    sum = a + b
    return sum
```

F5를 눌러 실행하면?



```
Python 3.6.2 Shell
File Edit Shell Debug Options Window Help

>>>
>>>
>>>
>>>
>>>
>>>
===== RESTART: W:/work2018/KPU/동계파이썬교육/add.py =====
>>>
>>>
>>>
>>>
>>>
>>>
```

아무 결과도 없음.

# 함수를 실행하려면, 함수 호출을 해야 함.

add.py

```
*add.py - W:/work2018/KPU/동계파이썬교육/add.py (3.6.2)*
File Edit Format Run Options Window Help

def add(a, b):
    sum = a + b
    return sum

result = add(100, 10)
print(result)
```

함수 호출

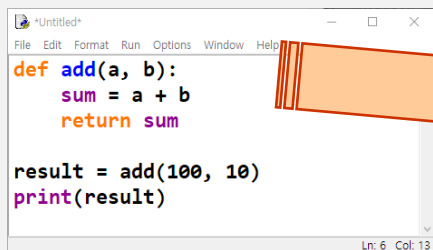
F5를 눌러 실행하면?

```
Python 3.6.2 Shell
File Edit Shell Debug Options Window Help

>>>
>>>
>>>
>>>
>>>
>>>
===== RESTART: W:/work2018/KPU/동계파이썬교육/add.py =====
110
>>>
>>>
>>>
>>>
>>>
```

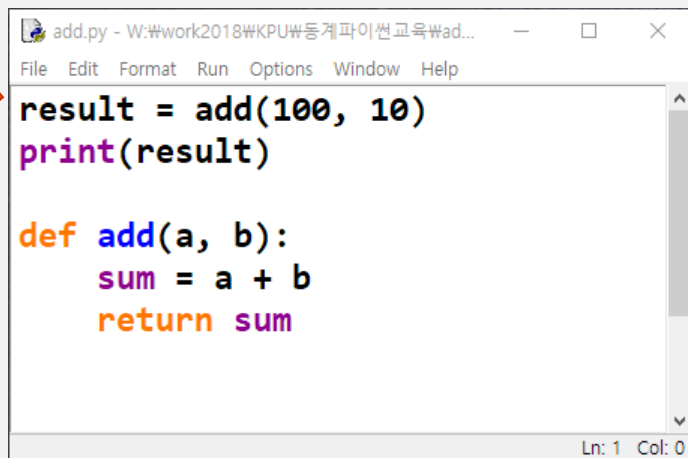
결과가 출력됨.

# 함수를 호출하려면, 함수 정의가 먼저 되어 있어야 함.



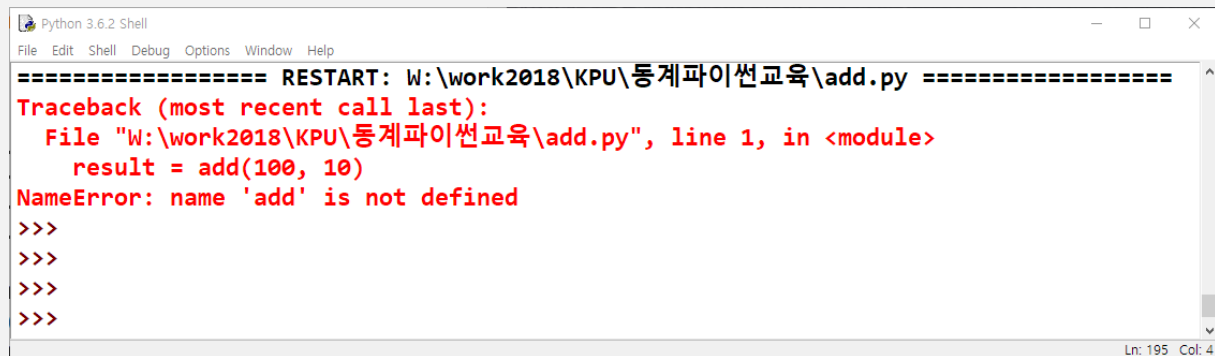
```
def add(a, b):  
    sum = a + b  
    return sum  
  
result = add(100, 10)  
print(result)
```

Ln: 6 Col: 13



```
result = add(100, 10)  
print(result)  
  
def add(a, b):  
    sum = a + b  
    return sum
```

Ln: 1 Col: 0



```
===== RESTART: W:\work2018\KPU\등계파이썬교육\add.py =====  
Traceback (most recent call last):  
  File "W:\work2018\KPU\등계파이썬교육\add.py", line 1, in <module>  
    result = add(100, 10)  
NameError: name 'add' is not defined  
>>>  
>>>  
>>>  
>>>
```

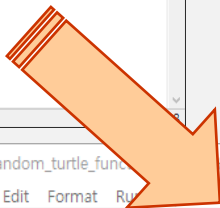
Ln: 195 Col: 4

# 함수는 여러 작업을 모아서 하나로 처리할 수 있게 해 줘.

```
drunken_turtle.py - W:\work2018\KPU\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()
```



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

import turtle
import random

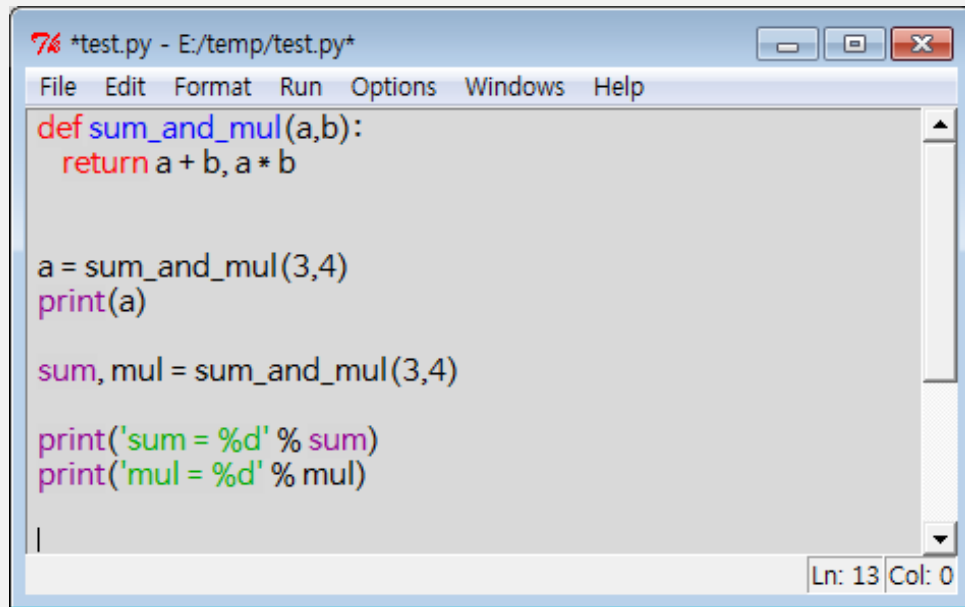
def drunken_move():
    turtle.setheading(random.randint(0, 360))
    turtle.forward(random.randint(100,200))
    turtle.stamp()

turtle.shape('turtle')
while (True):
    drunken_move()

Ln: 7 Col: 18
```



# 여러 개의 return 값 가능

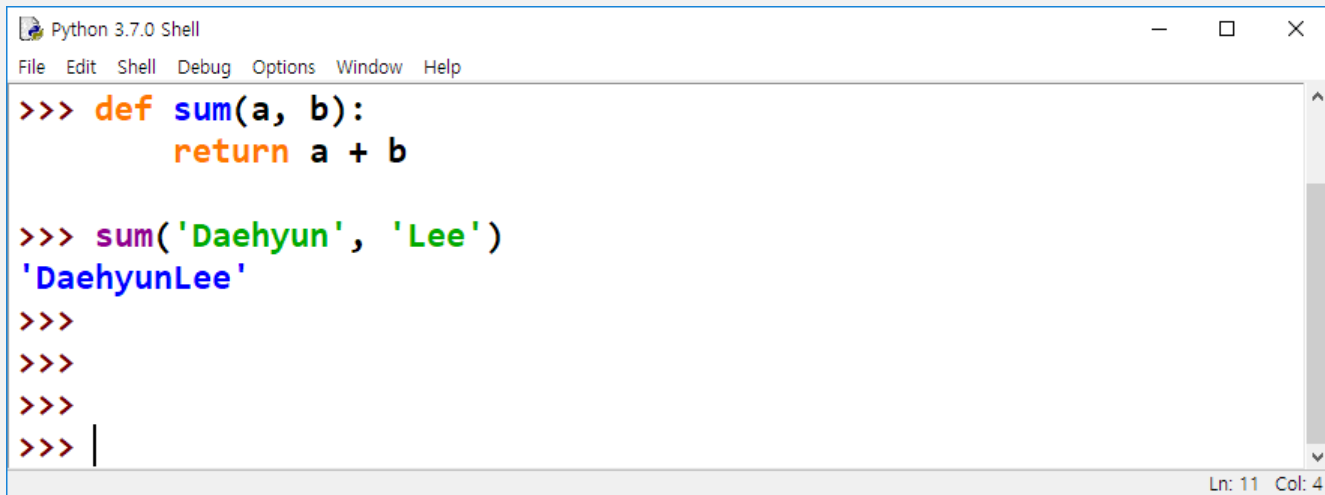


A screenshot of a Python IDE window titled "7% \*test.py - E:/temp/test.py\*". The window has a menu bar with "File", "Edit", "Format", "Run", "Options", "Windows", and "Help". The code editor contains the following Python code:

```
def sum_and_mul(a,b):  
    return a + b, a * b  
  
a = sum_and_mul(3,4)  
print(a)  
  
sum, mul = sum_and_mul(3,4)  
  
print('sum = %d' % sum)  
print('mul = %d' % mul)  
  
|
```

The status bar at the bottom right shows "Ln: 13 Col: 0".

# 인자의 타입에 따라 자동으로 연산 기능이 결정



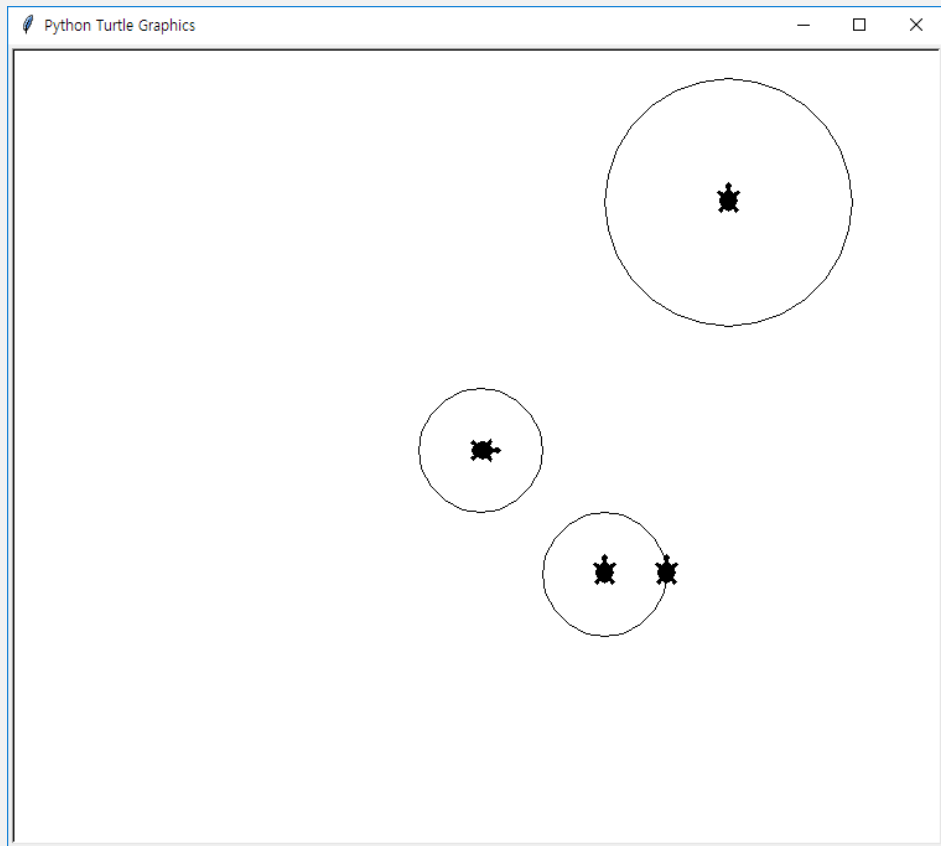
```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help

>>> def sum(a, b):
        return a + b

>>> sum('Daehyun', 'Lee')
'DaehyunLee'
>>>
>>>
>>>
>>> |
```

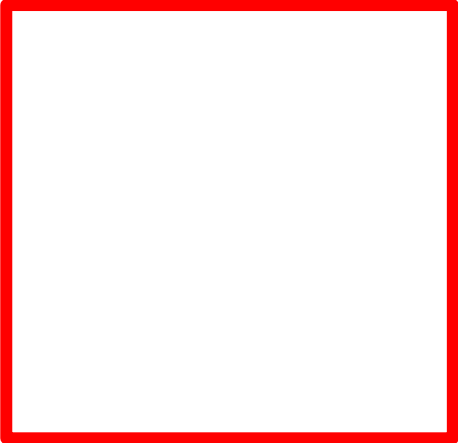
Ln: 11 Col: 4

# Quiz: 지정 위치 중심으로 원 그리는 함수 만들기



```
*draw_circle.py - W:/work2018/KPU/동계파이션교육/draw_circle.py ...
File Edit Format Run Options Window Help

import turtle

def draw_circle(x, y, r):
    

turtle.shape("turtle")
draw_circle(0, 0, 50)
draw_circle(200, 200, 100)
draw_circle(100, -100, 50)

Ln: 15 Col: 0
```

# 거북이를 키 입력을 통해서 조정하기

- `onkey()` 함수를 이용하여, 키 입력에 따라 반응하는 함수를 연결.
- `listen()` 함수를 이용해서, 거북이가 키 입력을 확인할 수 있게 함.

move 라는 이름의 함수가 호출됨.

```
turtle.onkey(move, 'w')
```

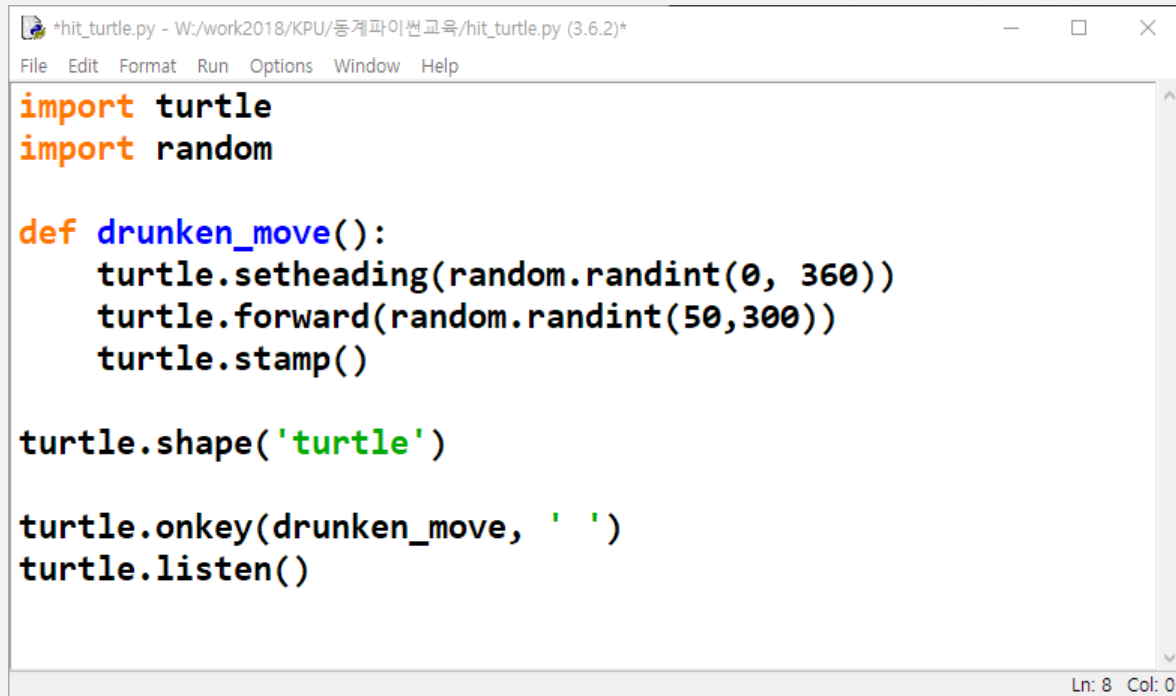
```
turtle.listen()
```

w 키를 누르면,

거북이가 키 입력을 들을 수 있게 함.

기호	뜻
'w'	w 키
'a'	a 키
's'	s 키
'd'	d 키
' '	스페이스 키
'Escape'	ESC 키

# 거북이 채찍질하기



The image shows a screenshot of a Python IDE window titled "\*hit\_turtle.py - W:/work2018/KPU/등계파이썬교육/hit\_turtle.py (3.6.2)\*". The window has a menu bar with "File", "Edit", "Format", "Run", "Options", "Window", and "Help". The code editor contains the following Python code:

```
import turtle
import random

def drunken_move():
    turtle.setheading(random.randint(0, 360))
    turtle.forward(random.randint(50, 300))
    turtle.stamp()

turtle.shape('turtle')

turtle.onkey(drunken_move, ' ')
turtle.listen()
```

The status bar at the bottom right indicates "Ln: 8 Col: 0".

# ESC 키를 누르면 다시 시작

A screenshot of a Python IDE window titled '\*hit\_turtle.py - W:/work2018/KPU/등계파이썬교육/hit\_turtle.py (3.6.2)\*'. The window contains a Python script for a turtle game. The script imports the 'turtle' and 'random' modules. It defines two functions: 'drunken\_move()' which sets a random heading and moves the turtle forward, and 'restart()' which resets the turtle. The turtle's shape is set to 'turtle'. The script then binds the 'drunken\_move' function to the space key and the 'restart' function to the 'Escape' key, and finally calls 'listen()' to start listening for keyboard events. The status bar at the bottom right shows 'Ln: 6 Col: 37'.

```
*hit_turtle.py - W:/work2018/KPU/등계파이썬교육/hit_turtle.py (3.6.2)*
File Edit Format Run Options Window Help

import turtle
import random

def drunken_move():
    turtle.setheading(random.randint(0, 360))
    turtle.forward(random.randint(50, 100))
    turtle.stamp()

def restart():
    turtle.reset()

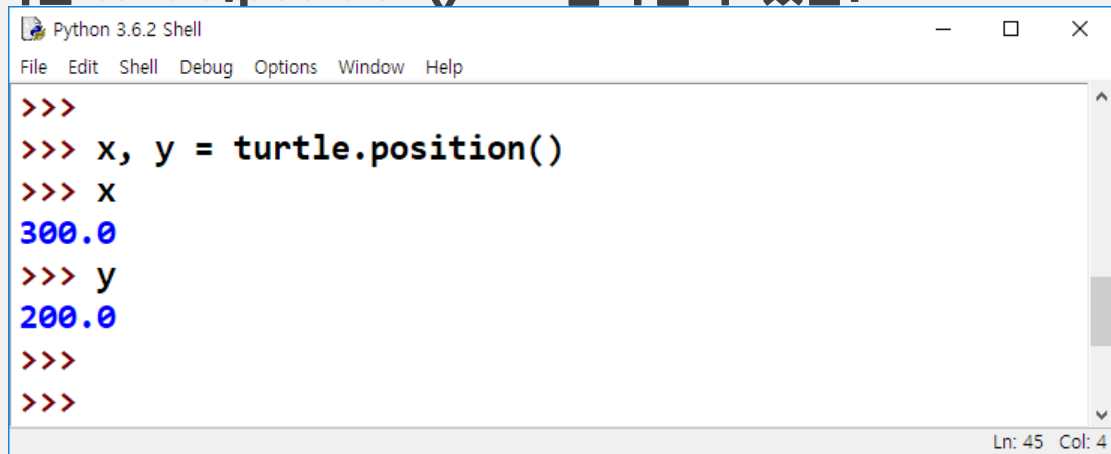
turtle.shape('turtle')

turtle.onkey(drunken_move, ' ')
turtle.onkey(restart, 'Escape')
turtle.listen()

Ln: 6 Col: 37
```

# 심화과제: 원 따먹기 게임

- 무작위 위치에 원이 만들어짐.
- 거북이를 이동시켜서 원에 닿으면, 다시 게임 시작
- global 변수를 쓸 줄 알아야 함.
- 거북이의 현재 위치는 `turtle.position()`으로 알아낼 수 있음.

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 is a text editor showing a Python REPL session. The prompt '>>>' is followed by the command 'x, y = turtle.position()'. The next line shows the variable 'x' being printed with the value '300.0'. The following line shows the variable 'y' being printed with the value '200.0'. The session ends with two more '>>>' prompts. At the bottom right of the window, the status bar shows 'Ln: 45 Col: 4'.

```
Python 3.6.2 Shell
File Edit Shell Debug Options Window Help
>>>
>>> x, y = turtle.position()
>>> x
300.0
>>> y
200.0
>>>
>>>
Ln: 45 Col: 4
```