Python 과학 프로그래밍 기초
6. 함수 (1)

한림대학교 소프트웨어융합대학
박섭형

2021년 1학기

1 함수

1.1 함수 개요

- 함수는 추상화(abstraction)의 기본 방법
  - 추상화: 사용자에게 애플리케이션의 내부 구현을 숨기고 사용법에만 집중하게 하는 과정
- 함수는 호출될 때만 실행되는 코드들의 블록이다
- 함수의 매개 변수 (parameters)를 이용해 데이터를 함수에 전달할 수 있다
- 함수는 데이터를 반환할 수 있다

함수 정의 구문

def function_name(parameter list): ""“docstring”"" statemet(s)

- 대부분의 함수는 return 문으로 끝나지만, 때로는 return 문이 없는 함수도 있다. 이 경우에는 반환하는 값이 없는 것이 아니고 None을 반환한다.

함수 생성

[1]:
```
import turtle
t = turtle.Turtle()


def forward_and_right(distance):
    t.forward(distance)
    t.right(90)

```

```
def draw_a_rectangle(side):
    """변의 길이가 side인 정사각형 그리기
    parameter:
      - side: 정사각형의 한 변의 길이
    """
    for _ in range(4):
        forward_and_right(side)

draw_a_rectangle(100)
draw_a_rectangle(120)



turtle.exitonclick()
```

[2]:
```
help(draw_a_rectangle)
```

```
Help on function draw_a_rectangle in module __main__:

draw_a_rectangle(side)
    변의 길이가 side인 정사각형 그리기
    parameter:
      - side: 정사각형의 한 변의 길이
```

[4]:
```
import turtle
t = turtle.Turtle()

def forward_and_right():
    t.forward(100)
    t.right(90)

forward_and_right()
forward_and_right()
forward_and_right()
forward_and_right()
```

```
t.forward(120)
t.right(90)
t.forward(120)
t.right(90)
t.forward(120)
t.right(90)
t.forward(120)
t.right(90)


turtle.exitonclick()
```

[6]:
```
import turtle
t = turtle.Turtle()

def forward_and_right():
    t.forward(100)
    t.right(90)

def forward_and_right_120():
    t.forward(120)
    t.right(90)

forward_and_right()
forward_and_right()
forward_and_right()
forward_and_right()

forward_and_right_120()
forward_and_right_120()
forward_and_right_120()
forward_and_right_120()

turtle.exitonclick()
```

[10]:
```python
import turtle
t = turtle.Turtle()

def forward_and_right(distance):
    t.forward(distance)
    t.right(90)

forward_and_right(100)
forward_and_right(100)
forward_and_right(100)
forward_and_right(100)

forward_and_right(120)
forward_and_right(120)
forward_and_right(120)
forward_and_right(120)

turtle.exitonclick()
```

[ ]:
```python
import turtle
t = turtle.Turtle()

def forward_and_left(distance):
    t.forward(distance)
    t.right(90)

for _ in range(4):
    forward_and_right(100)

for _ in range(4):
    forward_and_right(120)

turtle.exitonclick()
```

[ ]:
```python
import turtle
t = turtle.Turtle()
```

```
def forward_and_right(distance):
    t.forward(distance)
    t.right(90)

# 길이가 100인 정사각형 그리기
for _ in range(4):
    forward_and_right(100)

# 길이가 200인 정사각형 그리기
for _ in range(4):
    forward_and_right(120)


turtle.exitonclick()
```

[ ]:
```
import turtle
t = turtle.Turtle()

def forward_and_right(distance, rotation=90):
    t.forward(distance)
    t.right(rotation)

def draw_a_rectangle(side):
    for _ in range(4):
        forward_and_right(side)

def draw_a_star_polygon(side, no_of_vertices):
    for _ in range(no_of_vertices):
        forward_and_right(side, 180-180/no_of_vertices)

draw_a_rectangle(100)
draw_a_rectangle(120)

t.penup()
t.goto(150, 0)
```

```
t.pendown()

draw_a_star_polygon(100, 7)

turtle.exitonclick()
```

[ ]:
```
import turtle
t = turtle.Turtle()

def forward_and_right(distance):
    t.forward(distance)
    t.right(90)

def draw_a_rectangle(side):
    for _ in range(4):
        forward_and_right(side)

draw_a_rectangle(100)
draw_a_rectangle(120)

t.penup()
t.goto(150, 0)
t.pendown()

for _ in range(5):
    t.forward(100)
    t.right(180 - 36)

turtle.exitonclick()
```

[ ]:
```
import turtle
t = turtle.Turtle()

def forward_and_right(distance):
    t.forward(distance)
    t.right(90)
```

```python
def draw_a_rectangle(side):
    for _ in range(4):
        forward_and_right(side)

def draw_a_star_polygon(side, no_of_vetices):

    for _ in range(no_of_vetices):
        t.forward(side)
        t.right(180 - 180/no_of_vetices)




draw_a_rectangle(100)
draw_a_rectangle(200)

t.penup()
t.goto(300, 0)
t.pendown()

draw_a_star_polygon(100, 5)

turtle.exitonclick()
```

[ ]:
```python
import turtle
t = turtle.Turtle()

def forward_and_right(distance):
    t.forward(distance)
    t.right(90)

def draw_a_rectangle(side):
    for _ in range(4):
        forward_and_right(side)

def draw_a_star_polygon(side, no_of_vetices):
    for _ in range(no_of_vetices):
        t.forward(side)
        t.right(180 - 180/no_of_vetices)




draw_a_rectangle(100)
draw_a_rectangle(200)

t.penup()
t.goto(300, 0)
t.pendown()

draw_a_star_polygon(100, 5)

turtle.exitonclick()
```

[ ]:

```
import turtle
t = turtle.Turtle()

def forward_and_right(distance, rotation):
    t.forward(distance)
    t.right(rotation)

def draw_a_rectangle(side):
    for _ in range(4):
        forward_and_right(side)

def draw_a_star_polygon(side, no_of_vetices):
    for _ in range(no_of_vetices):
        forward_and_right(side, 180 - 180/no_of_vetices)




draw_a_rectangle(100)
draw_a_rectangle(200)

t.penup()
t.goto(300, 0)
t.pendown()

draw_a_star_polygon(100, 5)

turtle.exitonclick()
```

Positional Parameters and Keyword Parameters

- Positional parameters: 함수가 호출될 때 argument가 parameter의 순서대로 전달됨
- Keyword parameters: 함수가 호출될 때 argument가 'keyword=value" 형태로 전달됨

Argument 전달 방법

`def function(pos1, pos2, /, pos_kwd_1, pos_kws_2, *, kwd1, kwd2):`

- positional 전용 형식: pos1, pos2
- positional 또는 kweyword 형식: pos_kwd_1, pos_kws_2
- keyword 전용 형식: kwd1, kwd2

[11]:
```
def f(a, b, c):
    return 100*a + 10*b +c
print(f(3, 2, 1))
```

```
321
```

[12]:
```
def f(a, b, c, /, real=1, imag=1):
    return 100*a + 10*b +c, real + 1j * imag
print(f(3, 2, 1))
```

```
(321, (1+1j))
```

[13]:
```
print(f(3, 2, 1, 3))
```

```
(321, (3+1j))
```

[14]:
```
print(f(3, 2, 1, 3, 5))
```

```
(321, (3+5j))
```

[15]:
```
print(f(3, 2, 1, 3, imag=5))
```

```
(321, (3+5j))
```

[16]:
```
print(f(3, 2, 1, real=3, imag=5))
```

```
(321, (3+5j))
```

[17]:
```
print(f(3, 2, 1, real=3, 5))
```

```
  File "<ipython-input-17-0f698c1b93d0>", line 1
    print(f(3, 2, 1, real=3, 5))
                            ^

SyntaxError: positional argument follows keyword argument
```

[18]:
```
def f(a, b, c, /, real=1, imag=1, *, init=0):
    return 100*a + 10*b +c, real + 1j * imag, init
print(f(3, 2, 1))
```

```
(321, (1+1j), 0)
```

[19]:
```
print(f(3, 2, 1, 5, 5))
```

```
(321, (5+5j), 0)
```

[20]:
```
print(f(3, 2, 1, 5, 5, 6))
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-20-53a151ff2611> in <module>
----> 1 print(f(3, 2, 1, 5, 5, 6))

TypeError: f() takes from 3 to 5 positional arguments but 6 were given
```

[21]:
```
print(f(3, 2, 1, 5, 5, init=6))
```

```
(321, (5+5j), 6)
```