8주차

python에서 딥러닝까지

Python 기초반

주제 | 반복문 ~ 함수

Contents

Next

반복문

반복문 Mission4-1: 미디어아트1 - Ok

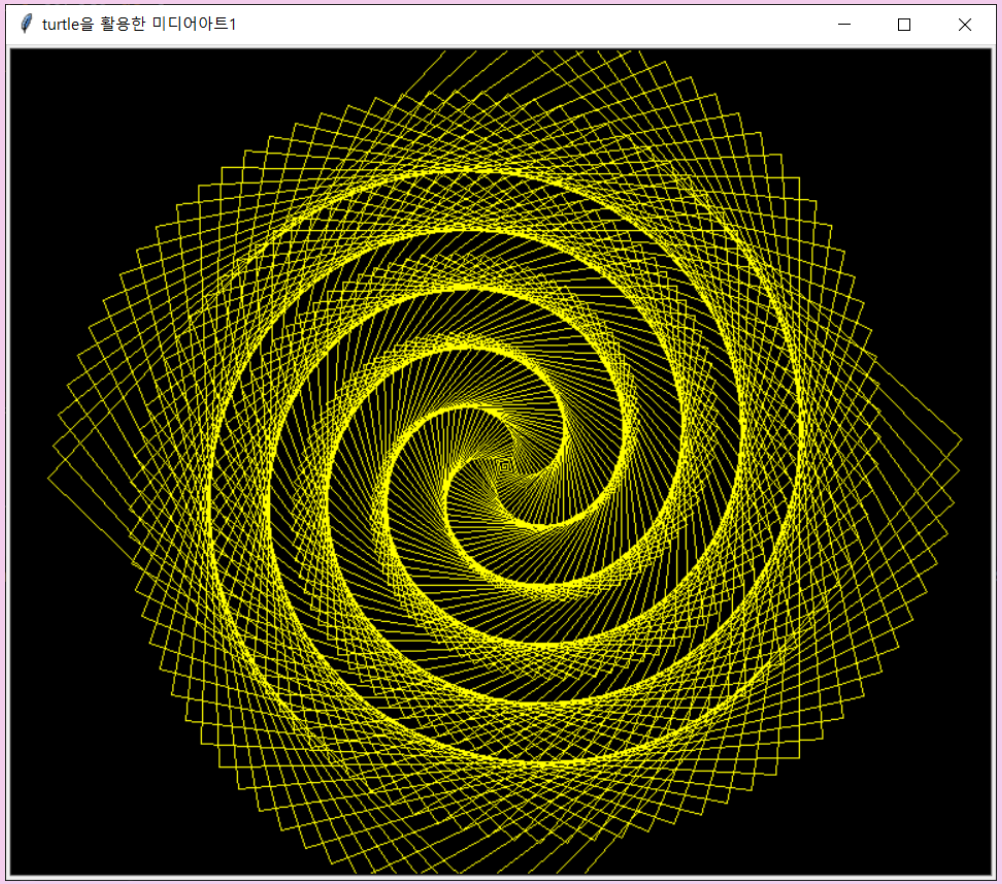
- turtle 명령어에 대한 내용은

  교재 55p, 111p 참고

- for 반복문을 활용해서 다음과 같은 문양을

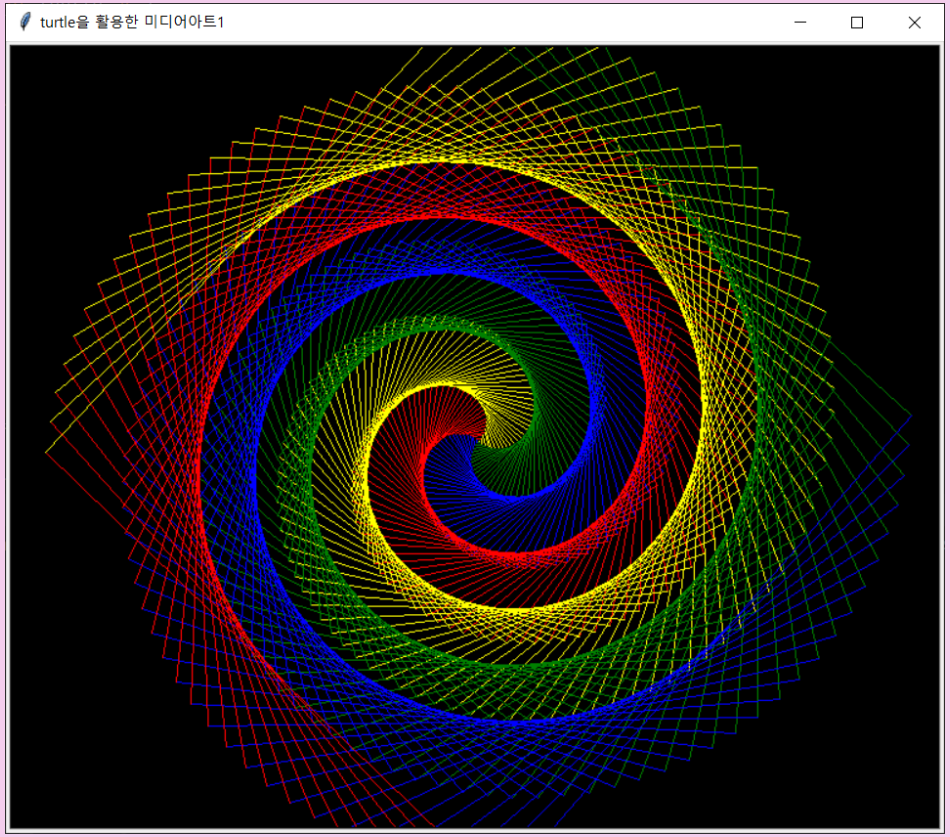
  출력해보자.

  - range를 활용하여 300번 반복할 것.
  - 꺾는 각도를 “89도”로 설정해 줄 것.
  - 사용 명령: t.forward(이동거리), t.right(각도)



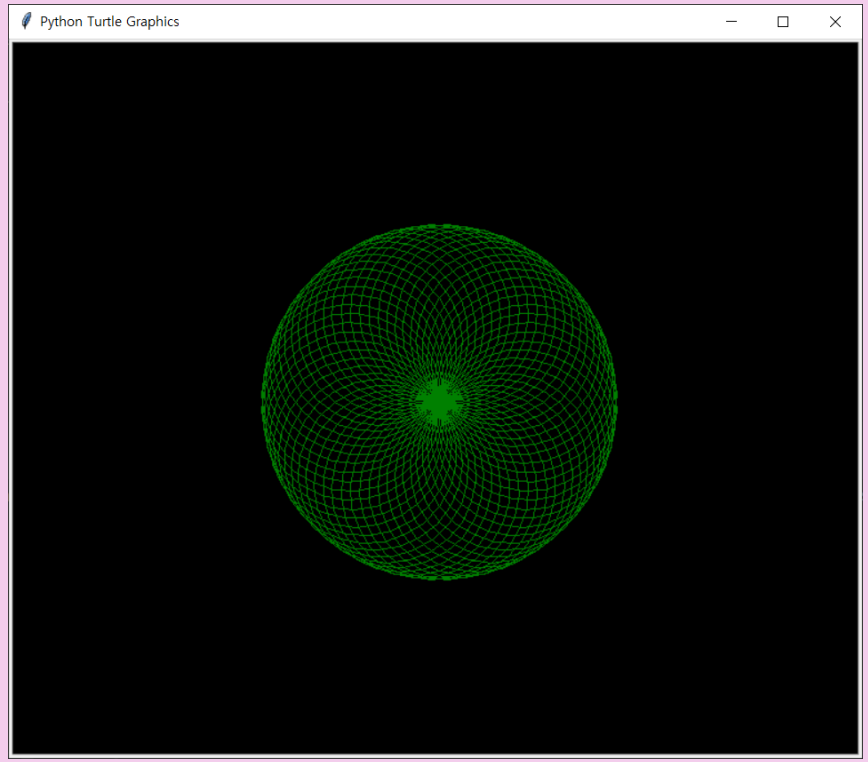
반복문 Mission4-2: 미디어아트1 - Ok

- for 반복문을 활용해서 다음과 같은 문양을 출력해보자.
  - list를 활용하여 ‘red’, ‘yellow’, ‘green’, ‘blue’를 각각 설정해 줄 것.
  - range를 활용하여 300번 반복할 것.
  - 꺾는 각도를 “89도”로 설정해 줄 것.



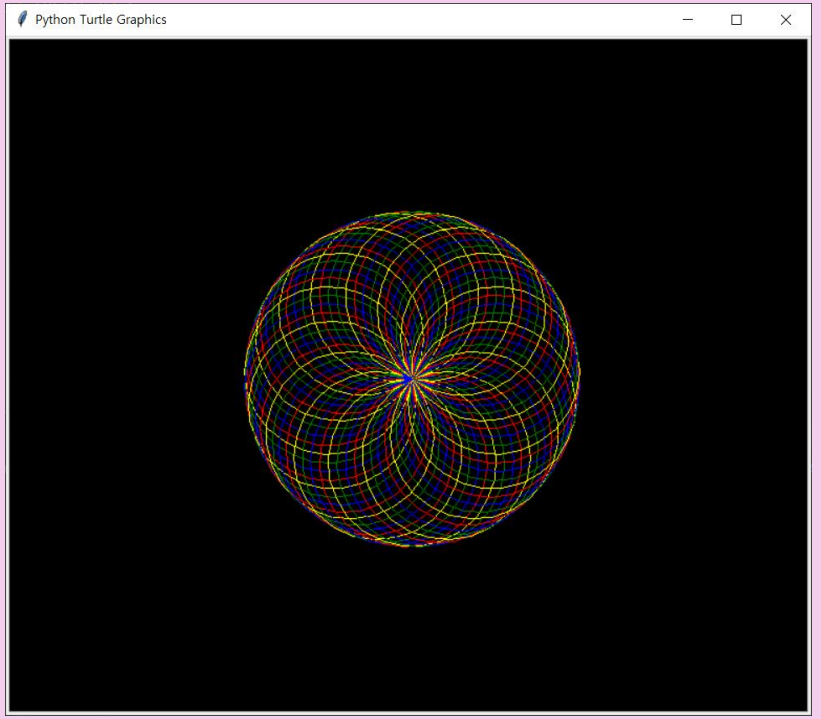
반복문 Mission5: 미디어아트 2 - Ok

- turtle 명령어를 활용하여 원을 반복해서 그리는 프로그램을 작성해보자.
- 활용 명령어
  - t.circle(반지름 길이), 여기서는 80~100 입력
  - t.left(각도)
- n은 50 이상으로 설정할 것.



반복문 Mission5-2: 미디어아트 2 - 0k

- turtle 명령어를 활용하여 원을 반복해서 그리는 프로그램을 작성해보자.
- color = [‘red’, ‘yellow’, ‘green, ‘blue’]로 색상 제어
- 활용 명령어
  - t.circle(반지름 길이), 여기서는 80~100 입력
  - t.left(각도)
- n은 50 이상으로 설정할 것.



Next

반복문 추가문제

이중 for문 (중첩 loop)

- for문 안에 for문을 반복하는 것. (while문도 중첩으로 사용 가능)
- 활용하는 경우: 차원이 있는 구조를 다루는 경우

```
array = [[101,102,103,104,105],
         [201,202,203,204,205],
         [301,302,303,304,305]]

for i in range(3):
    for j in range(5):
        print(array[i][j], end='  ')
    print()
```

이중 for문 연습문제 - with teacher

- **Mission:** 이중 for문을 활용하여 별 모양으로 직각삼각형 만들기
  - 조건1: 이중 for 문을 활용할 것
  - 조건2: 4줄 정도로 끝낼 것

```
C:\Anaconda3\envs\python_deep\python.exe "D:/Google 드라이브/Github_asdfrv20/Python_deep/3rd week mission.py"
*
**
***
****
*****
```

랜덤 함수

- 난수(random number): 무작위 숫자
- 필요 모듈: import random

```
import random
random_num = random.randint(1, 100)
```

- random 명령어
  - random.random( ): 0이상 1미만의 숫자 중에서 무작위 숫자를 돌려주는 함수
  - random.randint(시작숫자,끝나는 숫자의 다음숫자)
    : 정해진 범위에서 하나의 정수를 랜덤으로 반환하는 함수
    - 주의: randrange( )에 들어가는 숫자는 정수이다.

Mission1: Up-Down Game 만들기 - Ok

- **Mission**

  : 랜덤으로 정해진 숫자(1~100)사이 숫자를 맞추는 Up-Down 게임

  - computer가 1~100 사이의 랜덤정수를 하나 발생시킨다.
  - 유저가 임의의 숫자를 입력한다. 이때 입력한 숫자가
    - 랜덤 정수와 같으면 Win!!
    - 랜덤 정수보다 작으면 Up
    - 랜덤 정수보다 크면 Down

```
import random
```

```
random_num = random.randint(1, 100)
```

※ while을 활용하여 무한루프를 만들 것.

Mission2: ASCII 코드를 활용한 슬롯머신 - No

- **Mission: ASCII 코드를 활용한 슬롯머신 만들기**
- 조건
    1. 3개의 random 정수를 발생시킨 후, 이를 ASCII 코드로 변환하여 슬롯머신을 출력하는 프로그램을 작성한다
    2. 시작화면: “1.게임시작 // 2.나가기 >> " 를 출력 후 입력을 받는다
        1. 1. 선택 시, <SLOT MACHINE> 기능 실행
        2. 2. 나가기 실행 시 프로그램 종료
        3. 이외의 숫자 입력 시, “잘못 입력하셨습니다.”를 출력하고 다시 시작화면에서 입력을 받기

Mission2: ASCII 코드를 활용한 슬롯머신

3. <SLOT MACHIN> 기능: “코인을 넣어주세요!(1.코인넣기//2.그만하기)>>” 출력 후 입력받기
   - 1 입력 시, 3개의 랜덤한 정수를 발생시키고 이를 ASCII코드로 전환, 화면에 출력 후. 결과까지 출력 & <SLOT MACHIN> 기능 계속 반복
     - 셋 다 일치할 경우, ‘JACKPOT!!’ 출력)
     - 이외의 경우, ‘아쉽습니다. 다음 기회에 ' 출력
   - 2 입력 시, 메인 화면으로 이동
   - 이외의 입력에 대해, “잘못 입력하셨습니다.” 출력하기

Mission2: ASCII 코드를 활용한 슬롯머신

4. ASCII 코드로 자료형 변환하기

- random.randint()를 활용하여 “33~39”(총 7개) 사이의 정수를 임의로 3개 할당한다. (| “ # $ % & ‘ 로 총 7개)
- 생성된 변수를 print(“%c” %chr(자료형 변활할 변수))로 출력해 준다.
- 이를 슬롯머신과 같은 모양의 틀에 넣어 출력해 줄 것!

Mission2: ASCII 코드를 활용한 슬롯머신

- 출력 화면

```
[Slot Machine GAME]
1.게임시작 // 2.나가기>>1
```

```
1.게임시작 // 2.나가기>>2
게임을 종료합니다:D
```

```
1.게임시작 // 2.나가기>>5
잘못 입력하셨습니다.
```

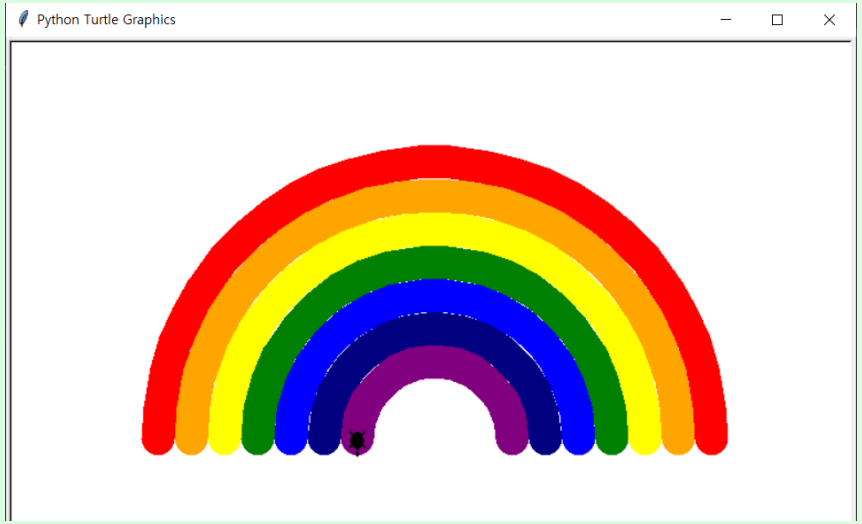
```
<SLOT MACHINE>
코인을 넣어주세요!(1.코인넣기//2.그만하기)>>1
--------------------
|  #  |  !  |  $  |
--------------------
아쉽습니다. 다음 기회에!
```

```
<SLOT MACHINE Start!>
코인을 넣어주세요!(1.코인넣기//2.그만하기)>>5
잘못 입력하셨습니다.

<SLOT MACHINE Start!>
코인을 넣어주세요!(1.코인넣기//2.그만하기)>>
```

```
<SLOT MACHINE Start!>
코인을 넣어주세요!(1.코인넣기//2.그만하기)>>2
슬롯머신 게임을 중단합니다.


[Slot Machine GAME]
1. 게임시작 // 2. 나가기>>
```

Mission3: turtle 모듈로 무지개 그리기

- **Mission: turtle 모듈을 활용하여 무지개 그리기**
  - **rainbow_size=500 // pen_size=30**
  - **rainbow_color = ['red', 'orange', 'yellow', 'green', 'blue', 'navy', 'purple']**
  - **for문 Hint!**
    : 위쪽 보기>펜 들기 > 무지개 시작점으로 이동 > 색 지정 > 펜 내리기 >그리기



Mission3: turtle 모듈로 무지개 그리기

- sethaeding(각도): 거북이가 바라보는 방향을 각도로 설정
- penup( ): 거북이 객체가 움직일 때 선이 남지 않도록 하기 펜을 드는 명령
- setpos(x좌표, y좌표): 거북이 객체를 화면의 (x좌표, y좌표)로 이동시키는 명령
- pencolor(색상명): 거북이 객체가 움직이면서 그리는 선의 색상을 설정하는 명령어(색상명은 문자열)
- pendown( ): 거북이 객체의 움직임에 따라 선이 남도록 펜을 내리는 명령어
- circle(반지름길이, 이동각도): 현재 위치에서 시작하여 입력된 반지름 길이를 가지는 원을 이동각도만큼 움직이도록 하는 명령어

Next

함수 (교재 96p)

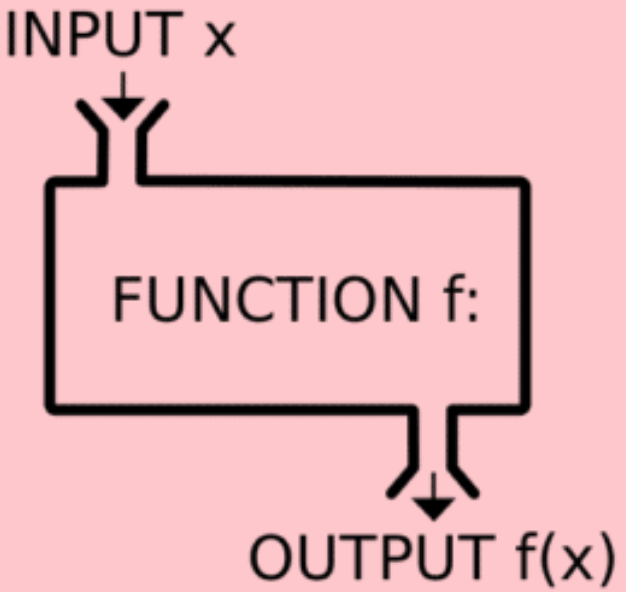
함수(function)의 개념

- 정의

  : 여러 개의 명령어들을 묶어서 한꺼번에 처리할 수 있도록 만든 하나의 명령어 묶음에 이름을 붙인 것.

- 이미 배워 사용하고 있는 함수들

  : **print( ), input( ), turtle.Turtle( ), turtle.setpos( ) ....**



함수의 장점 (함수를 사용하는 이유)

- 중복된 코드를 제어 > 코드가 짧아짐 > 메모리사용량 감소 > 비용 감소
- 유지 관리가 쉬워진다 (유지보수성)
- 함수를 다른 프로그램에서 재사용 가능 (재사용성)
- 코드가 읽기 쉬워진다(가독성)

함수 생성하기

- 함수 선언 문법 :

  def <함수이름> (입력 인자):

      함수 내에서 실행할 문장1 (코드)

      함수 내에서 실행할 문장2 (코드)

      …

      return 반환값

- <함수이름> 뒤에는 반드시 소괄호( )가 있어야 한다.
- 함수내에서 실행할 문장들은 반드시 들여쓰기 (공백 4칸)
- 입력인자가 없으면 생략 가능
- 반환값이 없다면 return 생략 가능
- 반환값이 있다면, 언제나 하나!
  2개 이상일 경우 list나 tuple을 활용한다

함수 Tip: docstring

- **docstring**이란?

  : 정의된 함수 위에 마우스를 올리면 네모난 창과 함께 나오는 함수에 대한 설명

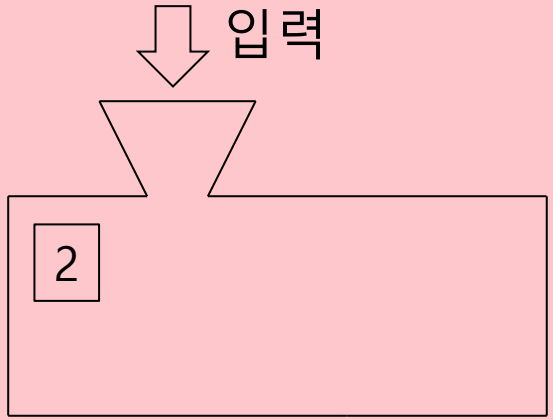
- **docstring** 작성 방법

  : **def** 바로 아래에 ‘’’ ‘’’ **or** “”” “””으로 주석처리를 한 후 그 안에 함수에 대한 설명을 작성한다.

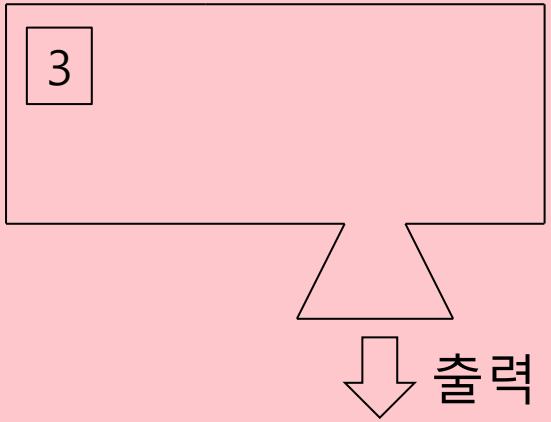
다양한 함수의 종류

입력X, 출력X

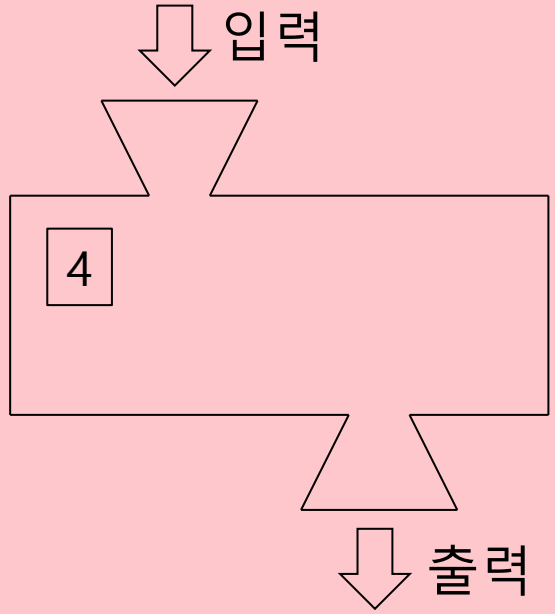
입력O, 출력X



입력X, 출력O



입력O, 출력O



입력과 출력이 모두 없는 함수

- 함수이름()
- 예시
  - turtle.home( )
  - turtle.clear( )
  - Hello world! 코드

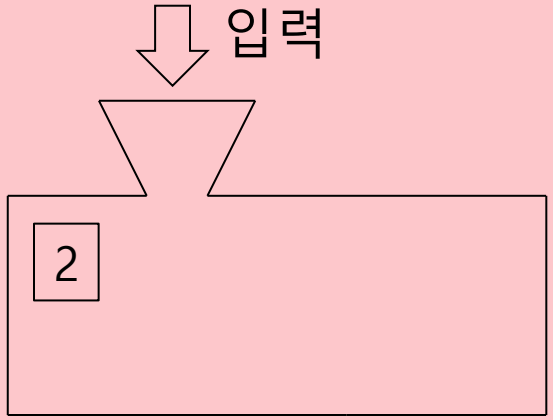

```
def hello_world():
    print("Hello world!")
hello_world()
```

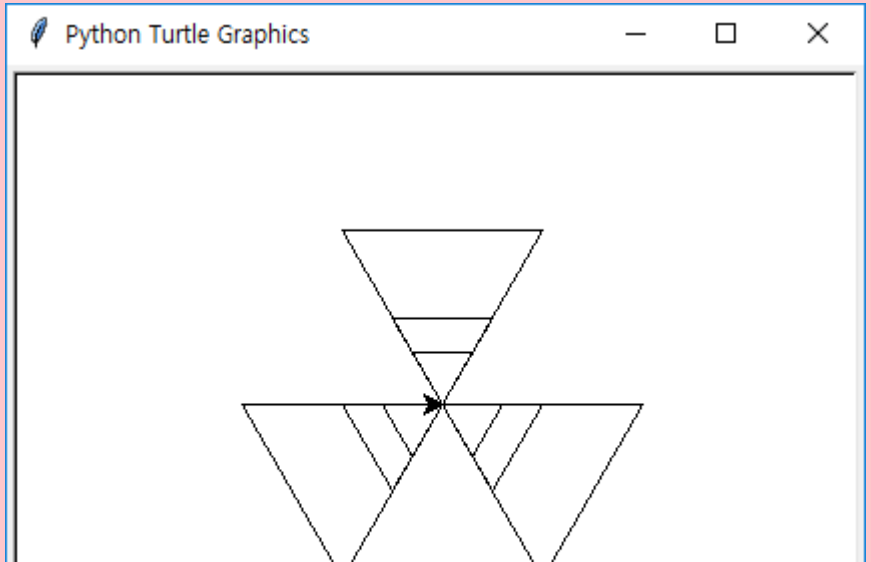
```
C:\Anaconda3\envs\python_deep\python.exe "D:/Google 드라이브/Github_asdfrv20/Python_deep/3rd week mission.py"
Hello world!

Process finished with exit code 0
```

입력만 있고 출력은 없는 함수

- 함수이름(입력인자)
- 예시
  - turtle.forward(이동 거리)
  - turtle.backward(이동거리)
  - turtle.right(각도)
  - turtle.left(각도)

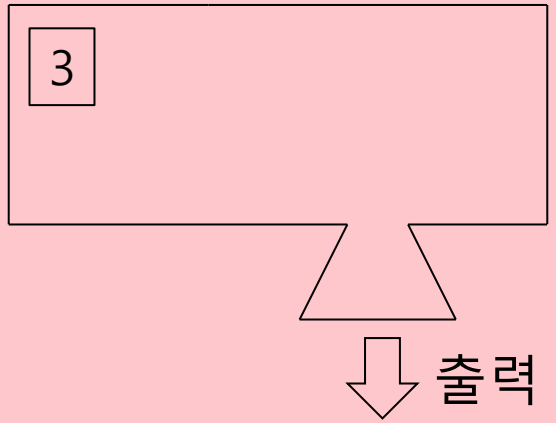
```
def drawTriangle(target, size):
    target.forward(size)
    target.right(120)
    target.forward(size)
    target.right(120)
    target.forward(size)
```





입력은 없고, 출력만 있는 함수

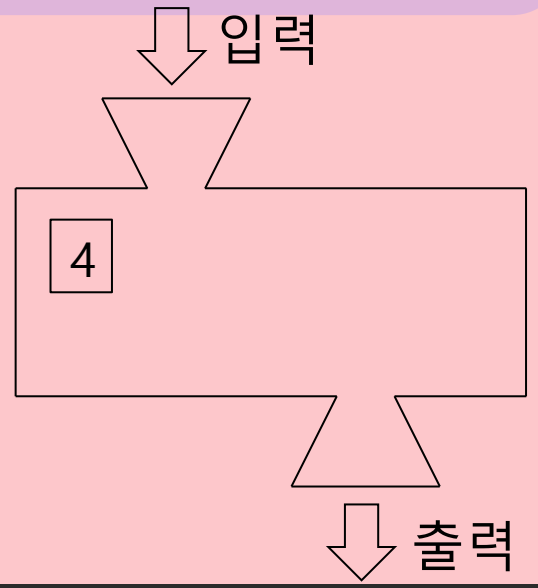
- 변수 = 함수이름()
- 예시
  - turtle.Screen( ): 입력은 없지만 Screen 객체를 반환
  - turtle.Turtle( ): 입력은 없지만 Turtle 객체를 반환




```
def getPi():
    import math      # math 모듈 불러오기
    return math.pi   # math 객체의 데이터 중 pi 반환

pi = getPi()         # getPi 함수 호출하여 반환값을 pi 변수에 저장
print(pi)            # pi 변수 출력
```

입력과 출력이 모두 있는 함수

- 변수 = 함수이름(입력)
- 예시
  - input('안내메세지')
    - 입력: '안내 메세지'
    - 출력: 키보드 입력을 문자열로 반환




```
def max(num1, num2):      # num1과 num2를 입력받는 max 함수 정의
    if (num1 > num2):     # 만약(if) num1이 num2보다 크다면,
        max = num1        # 최대값(max)은 num1이다.
    else:                 # 그렇지 않다면(num1이 num2보다 크지 않다면),
        max = num2        # 최대값(max)은 num2이다.
    return max            # 최대값(max) 반환!

print(max(14,7))          # 14와 7중 최대값 출력
```

함수 연습문제: 종류별 함수 만들기

- 입력X, 출력 X 인 함수

  : 함수 호출 시, 한 변의 길이가 100인 별을 그리는 DrawStart_100( )

- 입력O, 출력 X 인 함수

  : 별 모양 한 변의 길이를 입력하면, 그 길이의 한 변 길이를 가지는 별을 그리는 DrawStart( )

- 입력X, 출력 O 인 함수

  : 1~100까지 랜덤한 정수 1개를 반환하는 getRandomNum( )

- 입력O, 출력 O 인 함수

  : a, b를 입력하면 두 수의 합을 반환하는 add( )

함수 Mission

- **Mission**
  : 앞서 수행한 반복문 Mission4를 활용하여 무지개를 그리는 함수로 만들어 보자
- 조건
  : 입력으로 (거북이 객체, 무지개(반원)크기, 펜의 굵기, x좌표, y좌표)를 입력
- 함수 입력 값
  def draw_rainbow(t, rainbow_size, pen_size, x, y)
  - 거북이 객체: t
  - 무지개의 너비: rainbow_size
  - 펜의 굵기: pen_size
  - 무지개를 그릴 좌표: x,y