from functools import reduce

from operator import mul

def factorialHOF(n):

return reduce(mul, range(1, n+1), 1)

reduce ->

ex) [1, 2, 3, 4, 5]가 있고 x + y가 인자일 경우,

(x =1, y = 2, x + y = 3)

(x = 3, y = 3, x + y = 6)

(x = 6, y = 4, x + y = 10)

(x = 10, y = 5, x + y = 15)

Closure

def go\_home(fun):

def inner(\*args, \*\*kwargs):

print('집에 가고 싶다.')

fun(\*args, \*\*kwargs)

return inner

@go\_home

def t():

print('a')

다음과 같이 했을 경우, t에 대한 설명은 [<function\_\_main\_\_.go\_home.<locals>.inner(\*args, \*\*kwargs)>]

와 같이 출력됨.

이 경우를 피하기 위해

def go\_home(fun):

@wraps(fun)

def inner(\*args, \*\*kwargs):

print('집에 가고 싶다.')

fun(\*args, \*\*kwargs)

return inner

이와 같이 중첩된함수에 decorator wraps를 추가함.

[<function \_\_main\_\_.t()>]

3단 중첩을 할 경우,

데코레이터 자체에 인자를 주는 방법으로 해결