**MIT 코스웨어**

## C4 | 분해, 추상화, 함수

더 복잡하고 많은 기능을 가진 코드를 제작 및 보수하려면,
decomposition, abstraction을 통해 분해.
abstraction을 통해 굳이 안의 기능을 알지 못해도
사용가능하게 만드는 것이 주된 목적
Decomposition을 통해 간단한 장치(함수, 패키지)를 모아
복잡한 기능을 구현하고 재사용하도록 쉽게 보수 가능하게 제작
function이나 class를 통해서 이를 구현할 수 있다.

decomposition과 abstraction의 자세한 설명

function은 reusable한 코드의 조각을 말한다.
function은 call되거나 invoke되기 전까진 작동 X
이름, 파라미터, 주석, 전체, return이 기본적인 요소이다.
function이 불리면 formal paremeter가 value로 된다.
function이 불리면 scope/frame/envi. 가 생성된다.
만약 return이 없으면 function은 None으로 귀결

즉 def func()에서는 작동되지는 않지만, 실제로 func()가 되어야 function이 작동되기 시작함

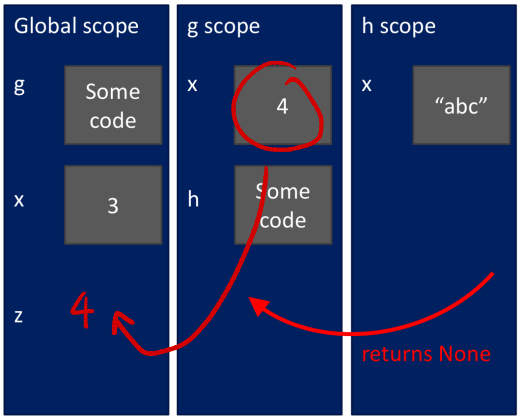
| return | print |
|---|---|
| return은 func. 안에서<br>오직 한 개의 value<br>def를 넘어가면 exec. (X) | print는 func. 밖에서도<br>여러 개의 print value<br>console의 output |

Global Scope와 func() Scope의 프로세스:

```
def g(x):
    def h():
        x = 'abc'
    x = x + 1
    print('g: x =', x)
    h()
    return x

x = 3
z = g(x)
```

① g(x)가 정의된다.
② x=3이 assign
③ g(x)를 call 함으로써 g scope가 생성
④ h를 정의하게된다
⑤ x=x+1이 된다.
∴ g(x)안에서는 x가 없으므로 밖의 x=3을 끌어와서 이를 새변수로 삼는다.
(단, x+=1 같이 외부의 x를 modify하는 process는 불가능)
⑥ 'g: x=4'를 print
⑦ h()가 call되면서 h scope가생성
⑧ h()에서 x = "abc" 이다.
⚠ return이 없으므로 None을 RETURN
⑨ None을 g scope로 전송한다
⑩ g scope None
⑪ g scope에서 x=4가 return되고 이것이 global scope에서 전달된다.




z = 4