

■ 10장 함수

function

*def _____ :
 {*

이 장의 내용

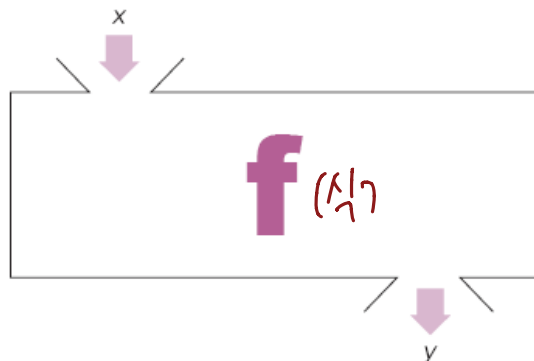
- 10.1 함수 정의
- 10.2 지역 변수와 전역 변수 *local variable/ global variable*
- 10.3 함수 예제 *Scope rule*
- 10.4 재귀 함수 *recursive*
- 10.5 모듈

10.1 함수 정의

함수란 무엇인가?

■ 함수 f

- 값 x 를 받아서 어떤 계산을 수행하여 결과값 $y = f(x)$ 를 돌려준다.



■ 예: 10% 할인 가격 계산 함수

- 함수 정의 $f(x) = x - x * 0.1$
- 함수 사용(호출) $f(48000) = 43200$

함수 정의 및 호출

definition

- **def 키워드**를 이용해서 새로운 함수 정의
 - 함수 이름 *값이 반환될 수 있음 (return)*
 - 매개변수 **parameter** : 함수가 값을 받기 위한 변수
 - 함수본체 **function body** : 이 함수가 수행할 문장들

def 함수이름(매개변수):

함수본체

↖ 값이 전달됨

- 함수 호출

- 정의된 함수 내의 코드를 실행한다.
- 인자: 함수를 호출할 때 전달하는 값

함수이름(인자)

↖ 값이 전달됨

예: 함수 정의 및 호출

프로그램 10.1 10% 할인 가격 계산 함수



10% 할인 가격 계산 함수

```
def salePrice(price):  
    result = price - price * 0.1  
    return result
```

↪ result값을 함수이름에 저장하는 개념.

실행 결과

```
>>> salePrice(48000)  
43200.0
```

인자

salePrice(48000)

def salePrice(price):

result = price - price * 0.1

return result

매개변수

(modulize)

예: 할인율을 받은 할인 가격 계산 함수

■ 매개변수

□ 가격 : price

□ 할인율 : rate (0.1 감소율)

프로그램 10.2 할인율을 받는 할인 가격 계산 함수



가격과 할인율을 매개변수로 받는 할인 가격 계산 함수

```
def salePrice(price, rate):  
    result = price * (1 - rate/100)  
    return result
```

실행 결과

```
>>> salePrice(48000, 30)  
33600.0
```

결과값을 반환하지 않는 함수

- 수학에서의 함수는 반드시 결과값을 반환하여야 한다.
- 파이썬 언어에서는 함수가 결과값을 반환하지 않아도 된다.

↳ return이 필요X

프로그램 10.3 결과값을 출력하는 함수



가격과 할인율을 매개변수로 받아 할인 가격을 출력하는 함수

```
def salePrice(price, rate):  
    result = price * (1 - rate/100)  
    print("할인 가격:", result)
```

실행 결과

```
>>> salePrice(48000, 30)
```

```
할인 가격: 33600.0
```


10.2 지역 변수와 전역 변수

local variable

*global
variable*

지역 변수 local variable

- 함수 내에서 선언된 변수는 함수 내에서만 사용 가능
 - 변수 `result`, 매개변수 `price`
- 유효범위 **scope** *"scope rule"*
 - 선언된 변수가 사용될 수 있는 범위
 - 지역 변수는 그 유효범위가 함수 내부
- 함수 밖에서 사용 오류

```
>>> salePrice(48000, 30)
33600.0
>>> print(result)
Traceback (most recent call last):
  File "<pyshell#33>", line 1, in <module>
    print result
NameError: name 'result' is not defined
```

소프트웨어의 이해



전역 변수 global variable

- 함수 밖에서 선언된 전역 변수
 - 함수 밖에서만 아니라 함수 내에서도 사용 가능.

프로그램 10.4 할인율을 전역 변수로 사용한 할인 가격 계산 함수

할인 가격 계산 함수를 활용하는 프로그램

rate = 20

→ global

할인율을 전역 변수로 사용한 할인 가격 계산 함수

def salePrice(price):

result = price * (1 - rate/100)

return result

original = int(input("가격을 입력하세요:"))

print("원래 가격:", original)

print(rate, "% 할인 가격:", salePrice(original))

rate = 30

print(rate, "% 할인 가격:", salePrice(original))

소프트웨어의 이해

실행 결과

가격을 입력하세요: 48000

원래 가격: 48000

20 % 할인 가격: 38400.0

30 % 할인 가격: 33600.0



~~주의사항~~: 전역 변수 사용

read(o)

write(X)

- 함수 내에서 전역 변수 사용은 가능하지만 수정은 불가!
 - 함수 내에서 대입문을 사용하여 전역 변수 수정을 시도하면
 - 자동적으로 그 이름에 해당하는 새로운 지역 변수를 만든다.

프로그램 10.5 전역 변수 값 수정 시도



전역 변수 값의 수정을 시도하는 프로그램

```
rate = 20 → global
def salePrice(price):
    rate = 30 → local ↘
    result = price * (1 - rate/100)
    return result
print(salePrice(50000))
print(rate)
```

실행 결과

35000.0

→ local

20

→ global



전역 변수 사용 오류

프로그램 10.6 전역 변수 사용 오류



전역 변수 사용 오류 프로그램

```
rate = 20
```

```
def salePrice(price):
```

```
    rate = rate + 10
```

```
    result = price * (1 - rate/100)
```

```
    return result
```

```
print(salePrice(50000))
```

local로 받아들이

최대 설정을 하지 않아서 무한대

실행 결과

Traceback (most recent call last):

...

```
rate = rate + 10
```



UnboundLocalError: local variable 'rate' referenced before assignment

함수 내에서 전역 변수 수정

- 함수 내에서 전역 변수 값을 수정하려면
 - 전역 변수를 먼저 **global**로 선언해야 한다

프로그램 10.7 전역 변수 값 수정



전역 변수 값을 변경하는 프로그램 2

✓ rate = 20

```
def salePrice(price) :
```

```
    global rate
```

```
    rate = 30
```

```
    # 혹은 rate = rate + 10
```

```
    result = price * (1 - rate/100)
```

```
    return result
```

```
print(salePrice(50000))
```

```
print(rate)
```

실행 결과

35000.0

30

global로 선언했으므로 ✓



10.3 함수 예제

삼각형 면적 계산 함수

■ 헤론의 공식

$$s = (a + b + c) / 2$$

$$\text{삼각형 면적} = \sqrt{s * (s-a) * (s-b) * (s-c)}$$

프로그램 10.8 헤론의 공식을 이용한 삼각형 면적 계산



`import math` 제공된 함수를 사용하기

헤론의 공식을 이용한 삼각형 면적 계산 함수

`def heron(a, b, c):`

`s = (a + b + c) / 2`
`area = math.sqrt(s * (s-a) * (s-b) * (s-c))`
`return area`

square root(√)

단, a, b, c가 △의 조건을 만족해야함

$$\begin{pmatrix} a+b > c \\ b+c > a \\ c+a > b \end{pmatrix}$$

실행 결과

`>>> heron(3,4,5)`

6.0



평균과 중앙값 계산 함수

프로그램 10.9 평균과 중간값 계산 함수



평균 계산 함수

```
def mean(x):  
    return sum(x) / len(x)
```

← 내장함수
↓

중간값 계산 함수

```
def median(x):  
    n = len(x)  
    x.sort()  
    mid = n // 2  
    if n % 2 == 1:  
        return x[mid]  
    else:  
        low = mid - 1  
        high = mid  
        return (x[low] + x[high])/2
```

→ 짝수 나누기



실행 결과

실행 결과

```
>>> incomes = [8800, 3500, 5600, 7500, 3900, 6000, 5200, 4100, 9000,
                6500]

>>> mean(incomes)
6010.0

>>> median(incomes)
5800.0
```

체질량지수 계산 함수

프로그램 10.10 체질량지수 계산 함수



키와 몸무게 리스트를 받아 체질량지수 리스트를 계산하는 함수

```
def bmi(height_weight_list):  
    bmi_list = []  
    for h, w in height_weight_list:  
        bmi = w / (h/100 * h/100)  
        bmi_list.append(bmi)  
    return bmi_list
```

```
incomes = [8800, 3500, 5600, 7500, 3900, 6000, 5200, 4100, 9000, 6500]  
print(mean(incomes))  
print(median(incomes))  
print(mean(incomes[0:5]))  
print(median(incomes[0:5]))
```

인덱스 후 평균 계산

실행 결과

투입은 리스트로 저장

```
>>> list1 = [(160,52), (162,65), (170,60), (157,50), (165,48)]
```

```
>>> result = bmi(list1)
```

```
>>> print(result)
```

```
[20.3125, 24.767565919829295, 20.761245674740483, 20.28479857195018,  
17.63085399449036]
```

파일 내의 단어 출현 횟수 계산 함수

def word_count(filename):

f = open(filename) *local*

✓ counts = {}

for line in f: *line 단위로 꺼냄 (\n)*

list = line.split() *word 단위로 쪼개냄*

for word in list:

if word in counts:

counts[word] += 1 *있으면 +1 (word를 인덱스로 봄)*

else:

counts[word] = 1 *없으면 새로 추가*

return counts

reply = 'y'

while reply == 'y':

filename = input("검색할 파일 이름을 입력하세요: ")

wc = word_count(filename)

for word in sorted(wc):

print(word, wc[word])

키 카운트값

reply = input("계속하겠습니까(y/n) ")

다른 변수임

global

실행 결과

실행 결과

검색할 파일 이름을 입력하세요: `raiseup.txt`

But 1

Each 1

I 9

Sometimes, 1

...

with 2

without 1

wonder, 1

you 2

your 1

계속하겠습니까(y/n)

10.4 재귀 함수

recursive

: 자기자신을 부르는 것

재귀 함수 recursive function

■ 함수를 정의하는데 자기 자신을 사용하여 정의하는 함수

□ 예: 팩토리얼

$$n! = n * (n-1) * \dots * 1$$

□ 재귀적 정의

$$1! = 1$$

$$n! = n * (n-1)!$$

$$3! = 3 * 2! = 3 * 2 * 1! = 3 * 2 * 1 = 6$$

□ 재귀 함수

$$\text{fact}(1) = 1$$

$$\text{fact}(n) = n * \text{fact}(n-1)$$

$$\text{fact}(3) = 3 * \text{fact}(2) = 3 * 2 * \text{fact}(1) = 3 * 2 * 1 = 6$$

팩토리얼 함수

프로그램 10.12 팩토리얼 함수



팩토리얼 함수

```
def fact(n):  
    if n == 1:  
        return 1  
    else:  
        return n * fact(n-1)
```

실행 결과

```
>>> fact(20)  
2432902008176640000
```



예: 리스트에서 값 검색 함수

- 리스트에서 순차적으로 값을 검색한다.
 - 첫 번째 원소와 비교하여 찾으면 **True** 리턴
 - 그렇지 않으면 나머지 원소에 대해서 재귀 호출

프로그램 10.13 재귀를 이용한 검색 함수 구현



재귀를 이용한 값 검색 함수

```
def search(lst, key):
```

```
    if not lst:                # 빈 리스트인 경우
```

```
        return False
```

```
    elif lst[0] == key:        # 찾고자 하는 원소를 찾은 경우
```

```
        return True
```

```
    else:                      # 리스트 나머지 부분에 대해서 검색
```

```
        return search(lst[1:], key)
```

실행 결과

```
>>> search([35, 28, 30, 29, 33, 31, 30], 30)
```

```
True
```



예: 리스트 내의 값의 개수 계산 함수

- 리스트에서 순차적으로 값을 검색하여 개수를 계산한다.
 - 첫 번째 원소와 비교하고, 나머지 원소에 대해서 재귀 호출
 - 총갯수 = 재귀 호출에서 찾은 개수(n) + 1 혹은 0

재귀를 이용한 값의 개수 검색 함수

```
def searchn(lst, key):
```

```
    if not lst: → 빈 리스트인 경우
```

```
        return 0
```

```
    elif lst[0] == key: 찾은 경우
```

```
        n = searchn(lst[1:], key)
```

```
        return n+1
```

```
    else:
```

```
        n = searchn(lst[1:], key)
```

```
        return n
```

실행 결과

```
>>> searchn([35, 28, 30, 29, 33, 31, 30], 30)
```

2



10.5 모듈

모듈 사용하기

- 표준 라이브러리 모듈
 - random 모듈: 난수 생성
 - time 모듈: 시간 날짜 정보

프로그램 10.14 모듈 사용하기



```
>>> import time
```

```
>>> print(time.asctime())
```

```
Mon Aug 29 10:59:08 2016
```

```
>>> print("안녕하세요")
```

```
안녕하세요
```

```
>>> time.sleep(2)
```

```
>>> print("반갑습니다")
```

```
반갑습니다
```

```
>>> import random
```

```
>>> print(random.randint(0,1000))
```

```
352
```

현재 컴퓨터의 시간


← 안의 숫자만큼 프로그램의 멈춤

모듈 사용

■ 모듈 내의 함수 import

`from 모듈이름 import 함수이름`

■ 예



```
>>> from random import randint
>>> print(randint(0,1000))
831
```

모듈 만들기

■ 모듈

✓ ↗ 다음시간(개념)

□ 함수나 변수 또는 클래스 등을 모아 놓은 파이썬 파일(.py)

mod1.py

다음에 이걸 import 하네.

```
# 평균 계산 함수
def mean(x):
    return sum(x) / len(x)

# 중간값 계산 함수
def median(x):
    n = len(x)
    x.sort()
    mid = n // 2
    if n % 2 == 1:
        return x[mid]
    else:
        low = mid - 1
        high = mid
        return (x[low] + x[high]) / 2
```



만든 모듈 사용하기

- ① 모듈 파일을 저장한 디렉터리로 이동 후에 파이썬 실행

```
C:\Users\...> cd C:\Python\Module
```

프롬프트창에서

```
C:\Python\Module> python
```

```
>>> import mod1
```

```
>>> mod1.median([1,3,5,7,9])
```

```
5
```

```
c:\WTemp>python
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 23:03:10) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> import mod1
>>> mod1.median([1,3,5,7,9,11])
6.0
```

- ② sys.path.append 사용하기

```
>>> import sys
```

```
>>> sys.path.append("C:/Python/Module")
```

```
>>> import mod1
```

```
>>> mod1.median([1,3,5,7,9])
```

```
5
```

```
>>> import mod1
Traceback (most recent call last):
  File "<pyshell#0>", line 1, in <module>
    import mod1
ModuleNotFoundError: No module named 'mod1'
>>> import sys
>>> sys.path.append("C:WTemp")
>>> sys.path
['', 'C:WUsersWWmoonWWAppDataWWLocalWWProgramsWWPythonWWPython38WWLibWWidlelib',
 'C:WUsersWWmoonWWAppDataWWLocalWWProgramsWWPythonWWPython38WWpython38.zip',
 'C:WUsersWWmoonWWAppDataWWLocalWWProgramsWWPythonWWPython38WWDLLs', 'C:WUsersWW
moonWWAppDataWWLocalWWProgramsWWPythonWWPython38WWlib', 'C:WUsersWWmoonWWAppDat
aWWLocalWWProgramsWWPythonWWPython38', 'C:WUsersWWmoonWWAppDataWWLocalWWProgram
sWWPythonWWPython38WWLibWWsite-packages', 'C:WTemp']
>>> import mod1
>>> mod1.median([1,3,4,5,6,7])
4.5
```

파일



만든 모듈 사용하기



환경 변수 PYTHONPATH 사용하기

```
C:\Users\...> set PYTHONPATH=C:\Python\Module  
C:\Users\...> python
```

```
>>> import mod1  
>>> mod1.median([1,3,5,7,9])  
5
```

```
Microsoft Windows [Version 10.0.17134.1304]  
(c) 2018 Microsoft Corporation. All rights reserved.  
  
C:\Users\Wmoon>set pythonpath=c:WTemp  
  
C:\Users\Wmoon>python  
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 23:03:10) [MSC v.1916 64 bit (AMD64)] on  
Type "help", "copyright", "credits" or "license" for more information.  
>>> import mod1  
>>> _
```



Key Point

Key Point

- `def` 키워드를 이용해서 새로운 함수를 정의(function definition)할 수 있다.
- 정의된 함수를 호출하여 실행하는 것을 함수 호출(function call)이라고 한다.
- 함수 내에서 선언된 변수는 함수 내에서만 사용될 수 있으며 이러한 변수를 지역 변수(local variable)라고 한다.
- 어떤 함수는 함수를 정의하는데 자기 자신을 사용하여 정의할 수 있는데 이러한 함수를 재귀함수(recursive function)라고 한다

프로그래밍 실습

▶ 프로그래밍 실습 1

- 학생들의 점수를 입력받아 평균과 중간값을 출력하시오. 프로그램 10.8의 `mean` 함수와 `median` 함수를 사용하시오.

▶ 프로그래밍 실습 2

- **search** 함수를 수정하여 찾고자 하는 값의 리스트 내의 인덱스를 계산하는 함수를 작성하시오. 만약 찾고자 하는 값이 첫 번째 원소와 같으면 해당 값을 찾았으므로 인덱스 **0**를 반환한다. 그렇지 않으면 나머지 리스트(**lst[1:]**)에 대해서 검색하도록 이 함수를 재귀 호출하고 이 호출이 반환한 인덱스에 **1**을 더하여 반환하면 된다. 이렇게 하여 리스트 내의 몇 번째 원소에서 해당 값을 찾았는지 계산한다.