AI Programming

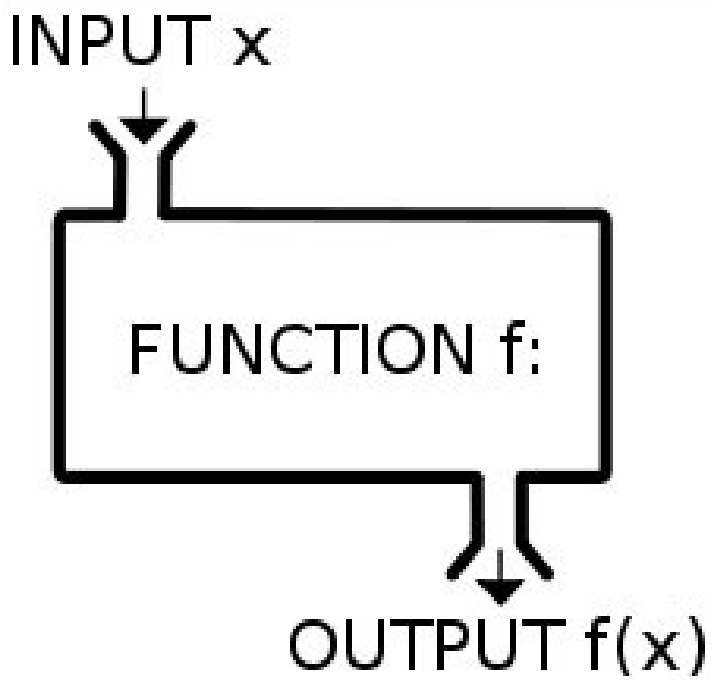
Lecture 12

Preview

- **Ch. 9 함수와 모듈**
  - 9.2 함수 기본
  - 9.3 지역 변수, 전역 변수
  - 9.4 함수의 반환값과 매개변수

9.2 함수 기본

Basics



Basics

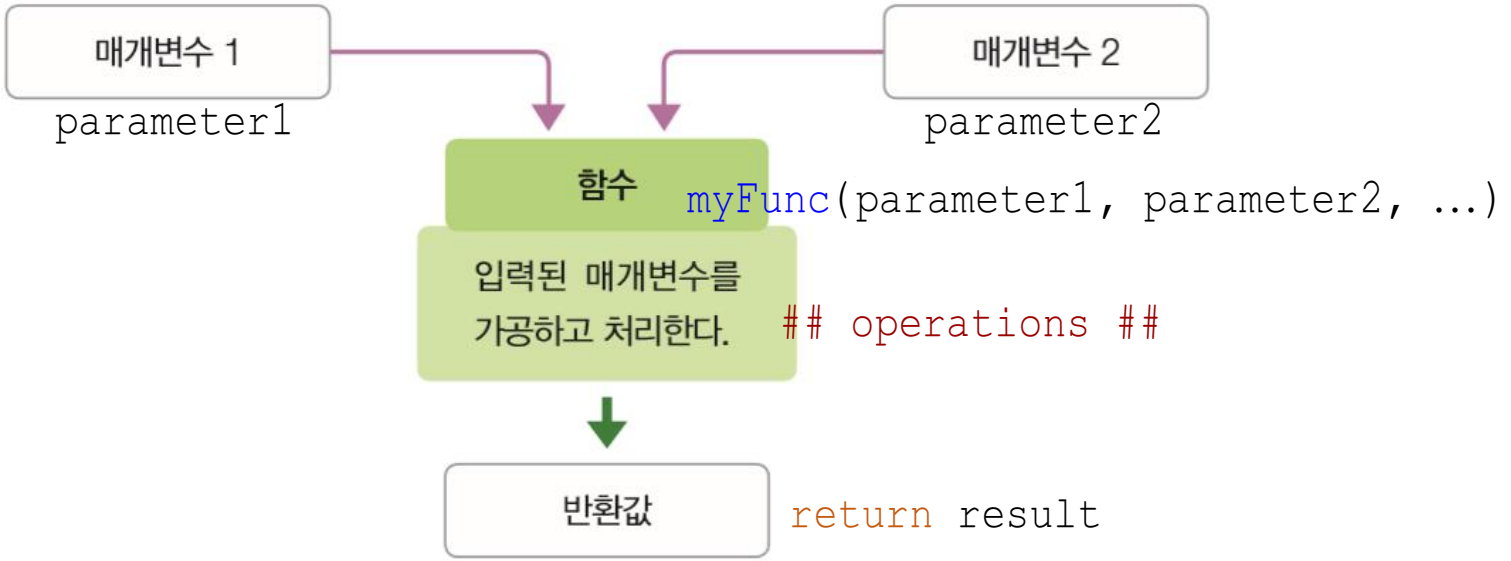
- **Functions**
  - `result = myFunc(parameter)`
  - `myFunc(parameter)`
- **Advantages**
  - Code decoupling
  - Encapsulation (캡슐화)

```
1  ## 변수 선언 부분 ##
2  inStr, outStr = "", ""
3  count, i = 0, 0
4
5  ## 메인 코드 부분 ##
6  inStr = input("문자열을 입력하세요 : ")
7  count = len(inStr)
8
9  for i in range(0, count) :
10      outStr += inStr[count - (i + 1)]
11
12 print("내용을 거꾸로 출력 --> %s" % outStr)
```

Basics

- **Syntax**

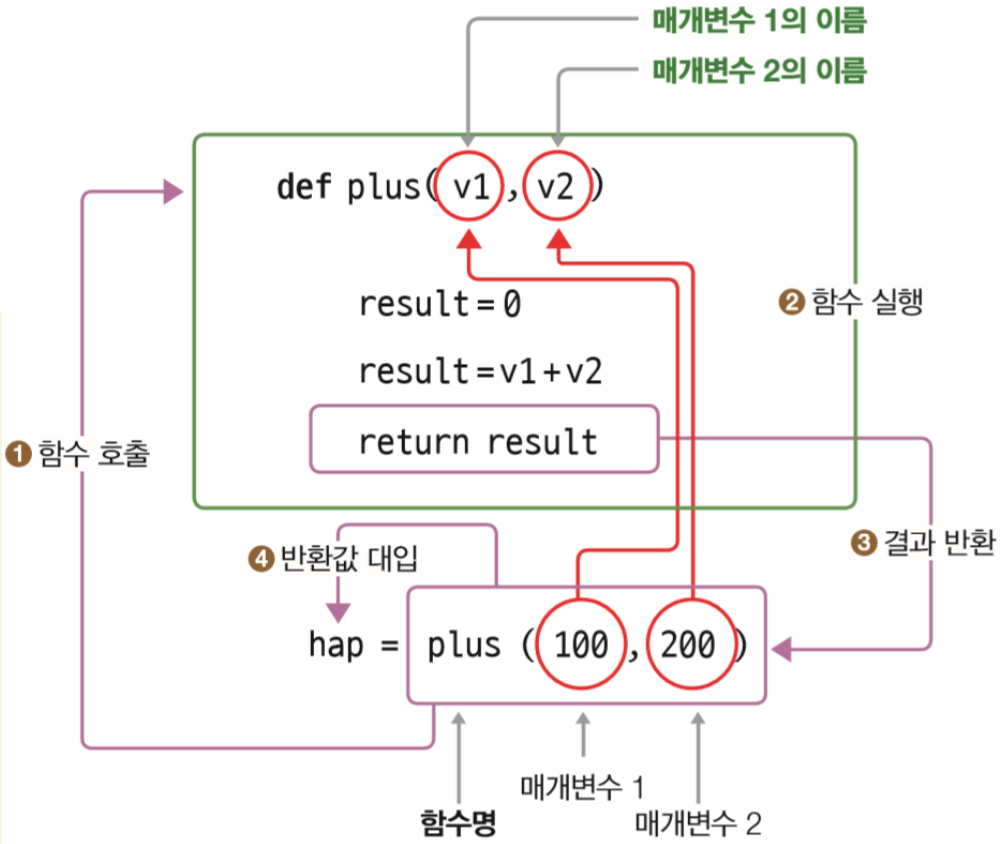
```
def myFunc(parameter1, parameter2, ...):
    ## operations ##
    return result
```



Basics

- **Exercise**

```
1  ## 함수 선언 부분 ##
2  def plus(v1, v2) :
3      result = 0
4      result = v1 + v2
5      return result
6
7  ## 전역 변수 선언 부분 ##
8  hap = 0
9
10 ## 메인 코드 부분 ##
11 hap = plus(100, 200)
12 print("100과 200의 plus() 함수 결과는 %d" % hap)
```



출력 결과

```
100과 200의 plus() 함수 결과는 300
```

Basics

- **Exercise**

```
1  ## 함수 선언 부분 ##
2  def calc(v1, v2, op) :
3      result = 0
4      if op == '+' :
5          result = v1 + v2
6      elif op == '-' :
7          result = v1 - v2
8      elif op == '*' :
9          result = v1 * v2
10     elif op == '/' :
11         result = v1 / v2
12
13     return result
```

```
15 ## 전역 변수 선언 부분 ##
16 res = 0
17 var1, var2, oper = 0, 0, ""
18
19 ## 메인 코드 부분 ##
20 oper = input("계산을 입력하세요(+, -, *, /) : ")
21 var1 = int(input("첫 번째 수를 입력하세요 : "))
22 var2 = int(input("두 번째 수를 입력하세요 : "))
23
24 res = calc(var1, var2, oper)
25
26 print("## 계산기 : %d %s %d = %d" % (var1, oper, var2, res))
```

출력 결과

```
계산을 입력하세요(+, -, * , /) : *
첫 번째 수를 입력하세요 : 7
두 번째 수를 입력하세요 : 8
## 계산기 : 7 * 8 = 56
```

Basics

- 함수 정의 → 메인 코드

```
## 메인 코드 부분 ##
result = get_area(3)
print("반지름이 3인 원의 면적=", result)

## 함수 선언 부분 ##
def get_area(radius):
    area = 3.14*radius**2
    return area
```

```
NameError: name 'get_area' is not defined
```

```
## 함수 선언 부분 ##
def get_area(radius):
    area = 3.14*radius**2
    return area

## 메인 코드 부분 ##
result = get_area(3)
print("반지름이 3인 원의 면적=", result)
```

Basics

- 함수 주석

```
## 함수 선언 부분 ##
def get_area(radius):
    # 원의 면적 계산
    # @param radius: 원의 반지름
    # @return area: 원의 면적
    area = 3.14*radius**2
    return area

## 메인 코드 부분 ##
result = get_area(3)
print("반지름이 3인 원의 면적=", result)
```

Basics

- **함수 이름**
  - 동사 + 명사
    - e.g.) get_area(), getArea()
  - 변수의 이름과 중복되지 않도록 주의

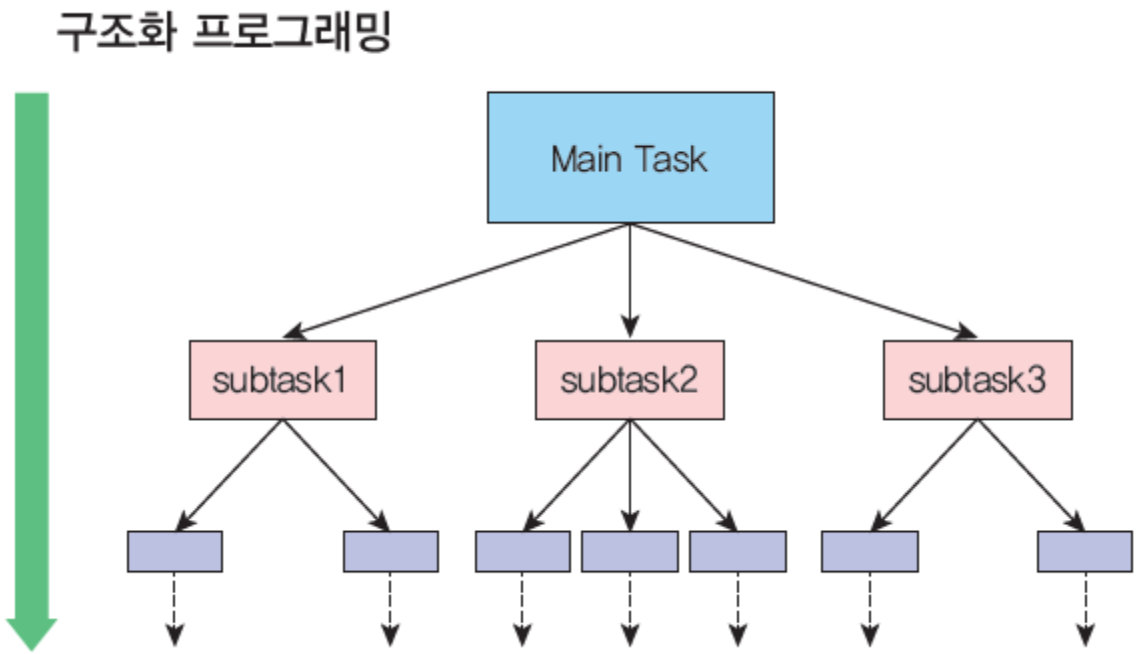
Basics

- 함수형 프로그래밍

```
                    함수 내에 다른
def main() :        함수 호출 가능
    result1 = get_area(3)
    print("반지름이 3인 원의 면적=", result1)

def get_area(radius):
    area = 3.14*radius**2
    return area

main()  메인 코드 부분을 함수 main()으로 대체
```

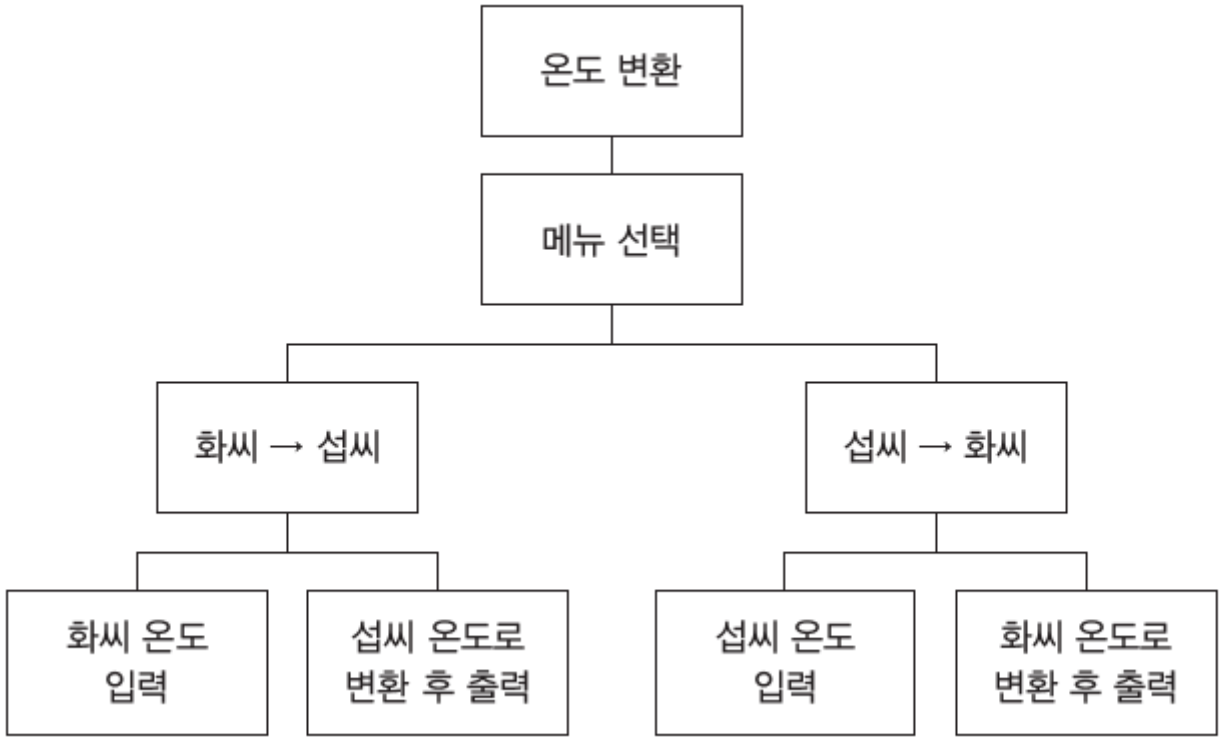
Basics

- 함수형 프로그래밍



Basics

- **Practice: 섭씨-화씨 온도 변환기**



```python
def print_menu() :
    print("1. 섭씨 온도->화씨 온도")
    print("2. 화씨 온도->섭씨 온도")
    print("3. 종료")
    selection = int(input("메뉴를 선택하세요: "))
    return selection

def c2f(c):
    temp = c*9.0/5.0 + 32
    return temp

def f2c(f) :
    temp = (f-32.0)*5.0/9.0
    return temp

def get_f() :
    f = float(input("화씨 온도를 입력하시오: "))
    return f

def get_c() :
    c = float(input("섭씨 온도를 입력하시오: "))
    return c
```

```python
def main() :
    while True:
        sel = print_menu()
        if sel == 1:
            t = c2f( get_c() )
            print(f"화씨 온도 = {t}")
        elif sel == 2:
            t = f2c( get_f() )
            print(f"섭씨 온도 = {t}")
        else :
            break

main()
```

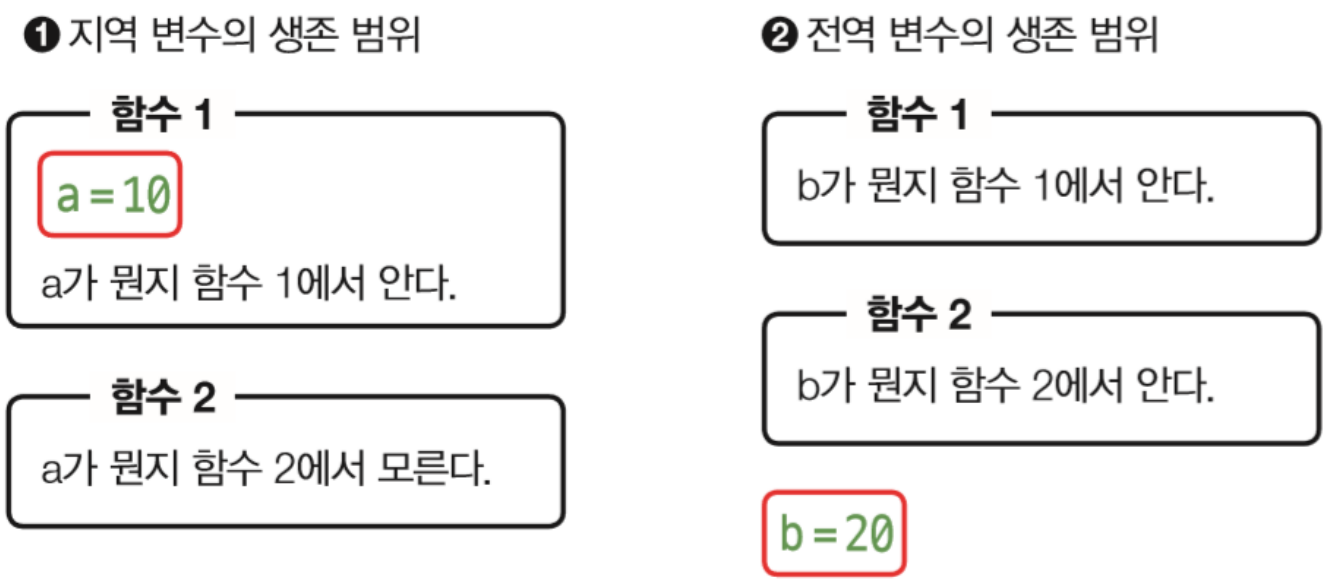
9.3 지역 변수, 전역 변수

Variables

- **Local variables vs. Global variables**
  - Local variables (지역변수)
    - 한정된 지역 (Ex: 함수 내) 에서만 사용
  - Global variables (전역변수)
    - 프로그램 전체에서 사용
    - 함수 내에서 새로운 값 대입 금지
    - `global` 명령어 사용시 대입 가능

Variables

- **Local variables vs. Global variables**



Variables

- **Local vs. Global**

```
def myfunc():
  x = 100
  print(x)

myfunc()
print(x)
```

```
gx = 100

def myfunc():
  print(gx)

myfunc()
print(gx)
```

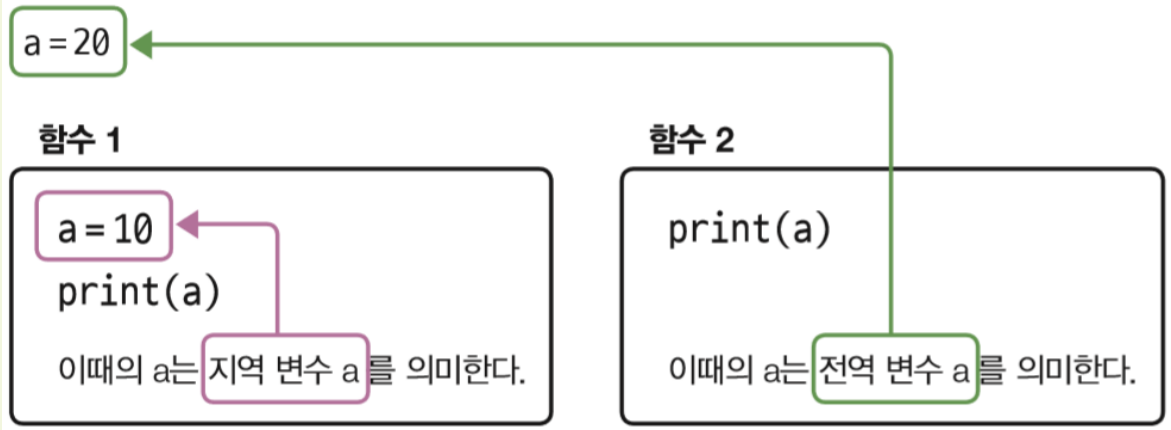
Variables

- **Local variable**
  - 지역 변수는 함수마다 동일한 이름 사용 가능

```
def myfunc() :
    x = 200
    print(x)

def main() :
    x = 100
    print(x)

myfunc()
main()
```

Variables

```
1  ## 함수 선언 부분 ##
2  def func1() :
3      a = 10    # 지역 변수
4      print("func1()에서 a값 %d" % a)
5
6  def func2() :
7      print("func2()에서 a값 %d" % a)
8
9  ## 전역 변수 선언 부분 ##
10 a = 20        # 전역 변수
11
12 ## 메인 코드 부분 ##
13 func1()
14 func2()
```



출력 결과

```
func1()에서 a값 10
func2()에서 a값 20
```

Variables

```
1  ## 함수 선언 부분 ##
2  def func1() :
3      a = 10    # 지역 변수
4      print("func1()에서 a값 %d" % a)
5
6  def func2() :
7      print("func2()에서 a값 %d" % a)
8
9  ## 전역 변수 선언 부분 ##
10 a = 20        # 전역 변수
11
12 ## 메인 코드 부분 ##
13 func1()
14 func2()
```

출력 결과

```
func1()에서 a의 값 10
Traceback (most recent call last):
    File "C:/파이썬코드/09-06.py", line 14, in <module>
        func2()
    File "C:/파이썬코드/09-06.py", line 7, in func2
        print("func2()에서 a값 %d" % a)
NameError: name 'a' is not defined
```

Variables

- global

```
1  ## 함수 선언 부분 ##
2  def func1() :
3      global a    # 이 함수 안에서 a는 전역 변수
4      a = 10
5      print("func1()에서 a값 %d" % a)
6
7  def func2() :
8      print("func2()에서 a값 %d" % a)
9
10 ## 함수 변수 선언 부분 ##
11 a = 20            # 전역 변수
12
13 ## 메인 코드 부분 ##
14 func1()
15 func2()
```

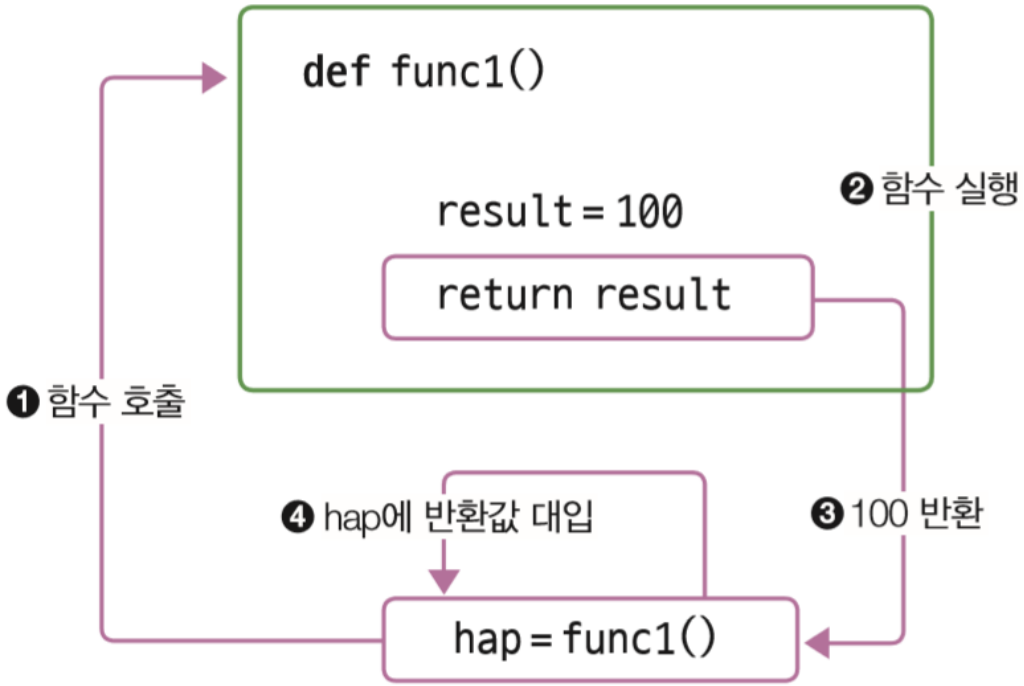
출력 결과

```
func1()에서 a값 10
func2()에서 a값 10
```

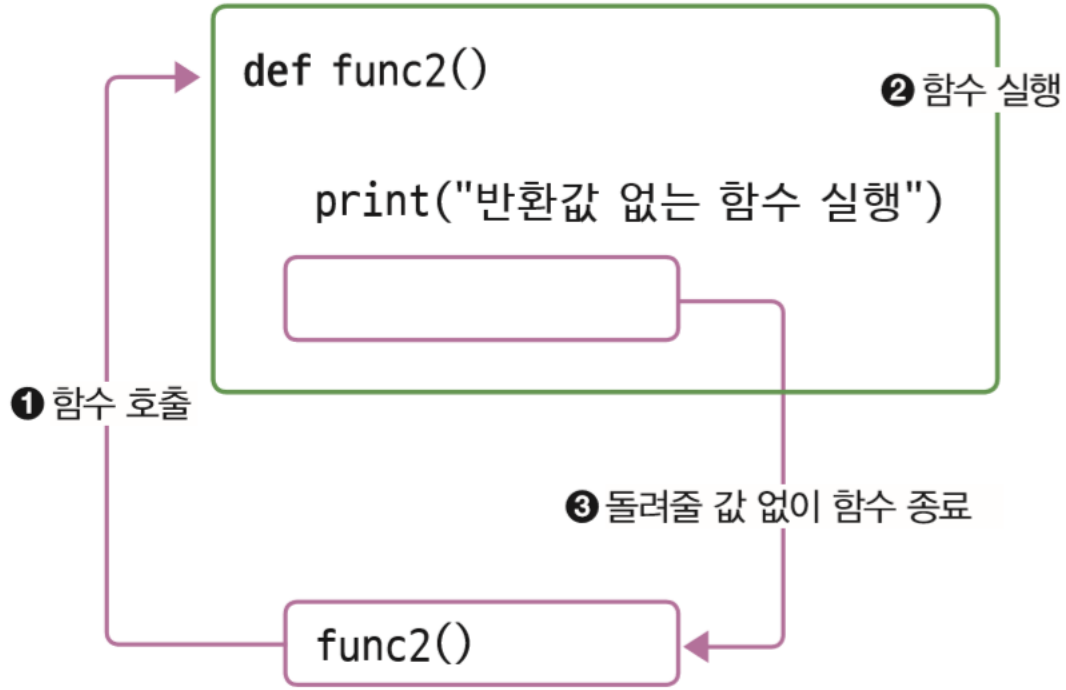
9.4 함수의 반환값과 매개변수

Returns

- **With return**



- **Without return**



Returns

- **Multiple returns**

```
def sub():
    return 1, 2, 3

a, b, c = sub()
print(a, b, c)

x = sub()
print(x)
```

Returns

- **Multiple returns**

```
1  ## 함수 선언 부분 ##
2  def multi(v1, v2) :
3      retList = []        # 반환할 리스트
4      res1 = v1 + v2
5      res2 = v1 - v2
6      retList.append(res1)
7      retList.append(res2)
8      return retList
9
10 ## 전역 변수 선언 부분 ##
11 myList = []
12 hap, sub = 0, 0
```

```
14 ## 메인 코드 부분 ##
15 myList = multi(100, 200)
16 hap = myList[0]
17 sub = myList[1]
18 print("multi()에서 돌려준 값 ==> %d, %d" % (hap, sub))
```

```
hap, sub = multi(100, 200)
```

출력 결과

```
multi()에서 돌려준 값 ==> 300, -100
```

Parameters

- **Default parameters**

```
1  ## 함수 선언 부분 ##
2  def para_func( v1, v2, v3 = 0 ) :
3      result = 0
4      result = v1 + v2 + v3
5      return result
6
7  ## 전역 변수 선언 부분 ##
8  hap = 0
9
10 ## 메인 코드 부분 ##
11 hap = para_func(10, 20)
12 print("매개변수가 2개인 함수를 호출한 결과 ==> %d" % hap)
13 hap = para_func(10, 20, 30)
14 print("매개변수가 3개인 함수를 호출한 결과 ==> %d" % hap)
```

출력 결과

```
매개변수가 2개인 함수를 호출한 결과 ==> 30
매개변수가 3개인 함수를 호출한 결과 ==> 60
```

Parameters

- **Keyword parameters**

```
def sum_range(begin, end, step=1) :          # 매개변수 step이 기본 값을 가짐
    sum = 0
    for n in range(begin, end, step) :
        sum += n
    return sum
```

```
print("sum = ", sum_range(1, 10))          # step은 디폴트 값(1)으로 처리됨
```

```
print("sum = ", sum_range(1, 10, 2))       # 정상 호출. step은 2
```

```
print("sum = ", sum_range(step=3, begin=1, end=10))   # 키워드 인수 사용
```

```
print("game ", end=" ")          # 라인피드가 발생하지 않음(키워드 인수 사용)
```

Parameters

- **Fixed-length parameters**

```
def para2_func( v1, v2 ) :
    result = 0
    result = v1 + v2
    return result
```

```
def para3_func( v1, v2, v3 ) :
    result = 0
    result = v1 + v2 + v3
    return result
```

Parameters

- **Variable-length parameter**

```
def myFunc(*parameter):
    ## operations ##
    return result
```

- `*parameter`
  - Tuple input

```
def myFunc(**parameter):
    ## operations ##
    return result
```

- `**parameter`
  - Dictionary input

Parameters

- **Exercise**

```
## 함수 선언 부분 ##
def para_func ( *para ) :
    result = 0
    for num in para :
        result = result + num

    return result

## 전역 변수 선언 부분 ##
hap = 0

## 메인 코드 부분 ##
hap = para_func(10, 20)
print("매개변수가 2개인 함수를 호출한 결과 ==> %d" % hap)
hap = para_func(10, 20, 30)
print("매개변수가 3개인 함수를 호출한 결과 ==> %d" % hap)
```

Parameters

- **Exercise**

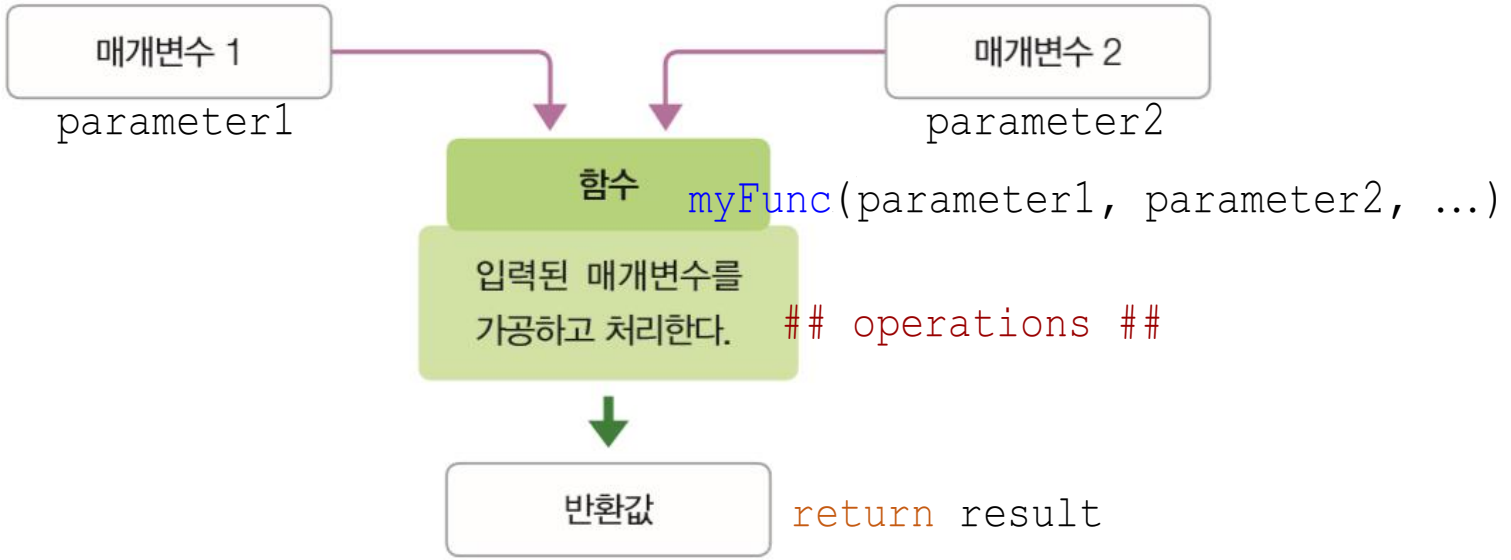
```
def dic_func(**para) :
    for k in para.keys() :
        print("%s  --> %d명입니다." % (k, para[k]))

dic_func(트와이스 = 9, 소녀시대 = 7, 걸스데이 = 4, 블랙핑크 = 4)
```

출력 결과

```
트와이스  --> 9명입니다.
소녀시대  --> 7명입니다.
걸스데이  --> 4명입니다.
블랙핑크  --> 4명입니다.
```

Summary

```
def myFunc(parameter1, parameter2, ...):
    ## operations ##
    return result
```



Assignment 10

- 연락처 프로그램 (Assignment 9)의 함수형 프로그래밍

```
def main():

    while True:
        sel = display_menu()

        if sel == 1:
            insert()
        elif sel == 2:
            delete()
        elif sel == 3:
            search()
        elif sel == 4:
            for key in address_book.keys():
                print_address(key)
        else:
            break
```

```
def insert():
    # 연락처 삽입 함수
    # get_contact() 함수 사용

def delete():
    # 연락처 삭제 함수
    # get_name(), is_key() 사용
    # print_error() 사용

def search():
    # 연락처 탐색 함수
    # get_name(), is_key() 사용
    # print_address(), print_error() 사용
```

Assignment 10

- **함수 기능**
  - display_menu()
    - 메뉴 출력
    - return: 사용자 입력 변수 sel
  - get_contact()
    - return: 사용자 입력 변수 name, number
    - 함수 get_name() 및 get_number()를 사용할 것 (input() 사용 금지)
  - get_name()
    - return: 사용자 입력 변수 name
  - get_number()
    - return: 사용자 입력 변수 number

1. 연락처 추가
2. 연락처 삭제
3. 연락처 검색
4. 연락처 출력
5. 종료
메뉴 항목을 선택하시오: 1
이름: 전요한
전화번호: 010-1111-4444

Assignment 10

- **함수 기능**
  - is_key(key)
    - parameter: key (이름)
    - return: True or False
      - True: key가 address_book에 존재할 경우
      - False: key가 address_book에 없을 경우
  - print_error()
    - 오류 메시지 출력
  - print_address(key)
    - key (이름)과 그에 해당하는 value (전화번호) 출력

```
1. 연락처 추가
2. 연락처 삭제
3. 연락처 검색
4. 연락처 출력
5. 종료
메뉴 항목을 선택하시오: 3
이름: 변기태
Error: 주소록에 없는 이름!
```

```
1. 연락처 추가
2. 연락처 삭제
3. 연락처 검색
4. 연락처 출력
5. 종료
메뉴 항목을 선택하시오: 4
전요한의 전화번호: 010-1111-4444
강인구의 전화번호: 010-2222-3333
```