# 파이썬 프로그래밍

## 6. 함수1

00

# Contents

## ❖ 수업 목표

- 함수를 정의하고 호출할 수 있다.
- 함수의 매개변수와 인수를 사용할 수 있다.
- return을 활용할 수 있다.
- 함수의 호출 순서를 설명할 수 있다.

## ❖ 세부 목표

00

# 1. 함수 정의

## ❖ 함수

- 반복되는 코드 모아 이름 붙인 것
  - 특정 목적의 작업을 수행하기 위해 독립적으로 설계된 코드의 집합

## ❖ 함수 사용 목적

- 모듈성(재사용성)
  - 함수는 한번 정의하면 프로그램 내 여러 위치에서 호출하여 사용 가능
- 가독성
  - 코드를 기능 단위로 분리하면, 어떤 역할을 하는지 이해가 용이
- 유지보수성
  - 기능 수정 시, 정의된 함수만 수정하면 호출되는 여러 위치에서 동일하게 동작
- 추상성
  - 함수 내부의 복잡한 처리 과정을 모두 이해할 필요 없이 원하는 작업 수행 가능

00

# 1. 함수 정의

## ❖ 함수의 종류

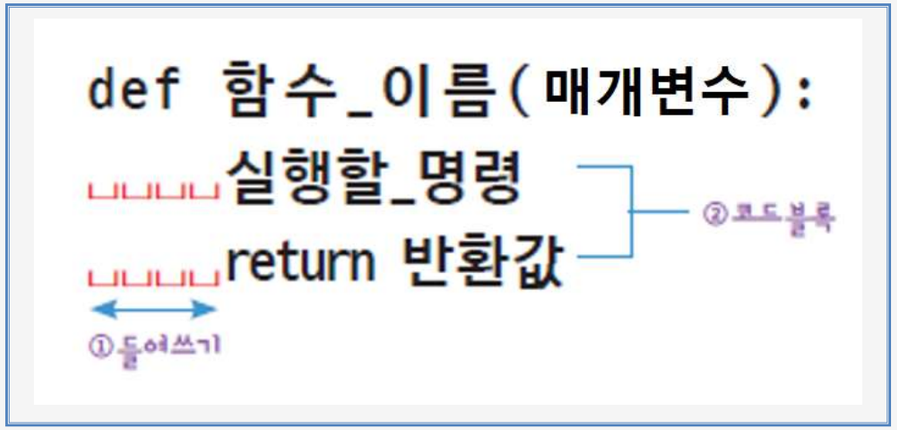
- **내장(Built-in) 함수**
  - 파이썬에 포함되어 있는 함수
  - print()
- **모듈(Module)의 함수**
  - 모듈 : 비슷한 함수끼리 모아둔 파일
- **사용자 정의 함수**
  - 직접 만들어 사용하는 함수

00

# 1. 함수 정의

## ❖ 함수의 기본 구조

- def 키워드 사용
- 괄호 작성 필수(매개변수는 옵션)
- 매개변수 (Parameter)
  - 함수에 전달하는 입력 값
- 반환값
  - 함수 실행 결과로 돌려주는 값




코드 8-2 함수를 사용해 문자열을 출력하는 코드

```
01:  def my_func():
02:      print('토끼야 안녕!')
03:
04:  my_func()
```

↳ 실행화면

```
토끼야 안녕!
```

00

# 2. 함수 호출

## ❖ 함수 호출 형식

- **함수 호출** : 정의된 함수의 기능을 사용
- 함수 호출 방법 : 함수명을 사용할 인수와 함께 사용
  - 일반적으로 인수는 매개변수가 정의된 경우에 사용 가능

- **인수** (Argument)
  - 함수에 전달하는 입력 값

함수 정의 시 매개변수 = 함수 호출 시 인수

# 2. 함수 호출

## ❖ 함수 호출 형식

- 함수가 호출되면 인수 값은 매개변수에 대입되어 함수 기능을 수행
- 함수 정의의 매개변수와 함수 호출의 인수는 1대 1 대입 관계

**syntax : 함수 정의와 함수 호출 형식**

```
def 함수 이름(매개변수):
        명령어
        명령어
        [return 반환값]

함수 이름(인수)
```

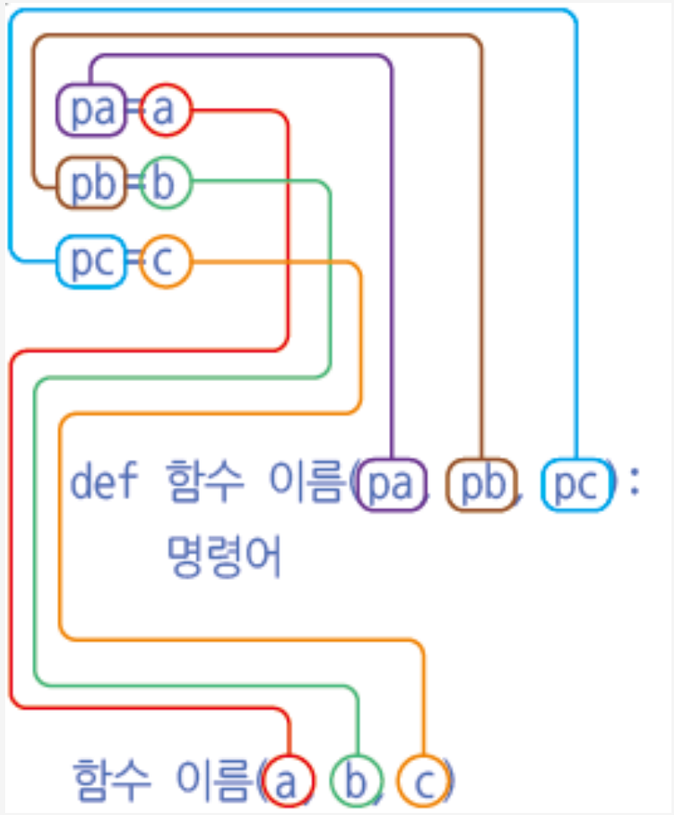
# 2. 함수 호출

## ❖ 함수 호출 형식

- 다음 3개 데이터를 인수로 사용할 경우 매개변수도 3개 필요
- 각 요소는 쉼표(, comma)로 구분

```
a=10
b=10.7
c="파이썬"
```

- 각각의 매개변수명을
  pa, pb, pc로 하고
  인수와 매개변수의 관계를
  대입 연산자로 표현



# 3. 매개변수와 함수 호출

## ❖ 매개변수가 없는 함수 호출

```
# 함수 정의하기
def fhello(): # -----> ①
    print("매개변수 없는 함수 호출하기")

# 함수 호출하기
fhello() # ---------> ②
```

〈화면 출력〉

```
매개변수 없는 함수 호출하기
```

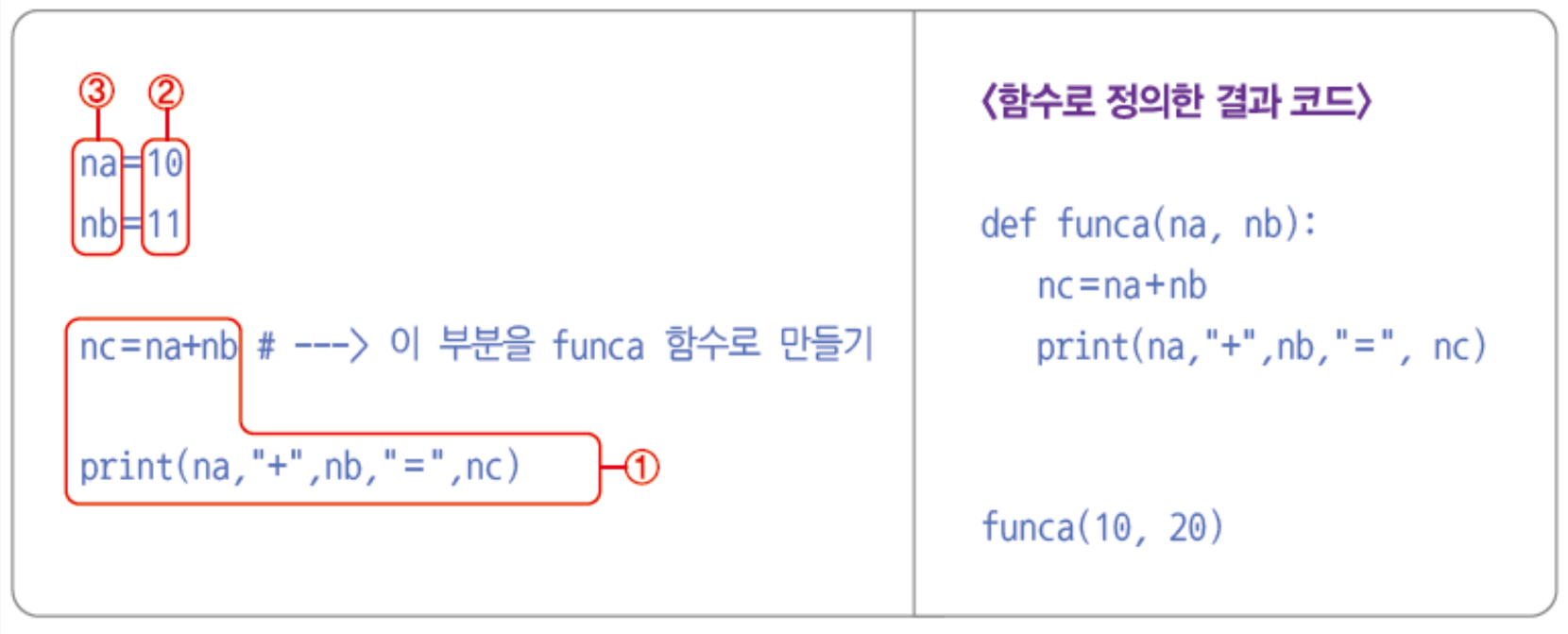
- ① fhello 함수를 정의에 매개변수 없음
- ② fhello 함수를 호출 시 인수 사용 불가 (단, 괄호는 있어야 함)

# 3. 매개변수와 함수 호출

## ❖ 매개변수가 있는 함수 호출

### ▪ 함수로 바꾸는 코드

- 매개변수와 인수의 관계 이해를 위해 연산 프로그램을 함수로 변환하기



# 2. 함수 호출

## ❖ 함수 정의 및 호출 예시

- **add() 함수**
  - 두 개의 숫자를 입력으로 받음

코드 8-3 함수를 사용해 두 개의 숫자를 더하는 코드

```
01: def add(num1, num2):
02:     return num1 + num2
03:
04: print(add(2, 3))
```

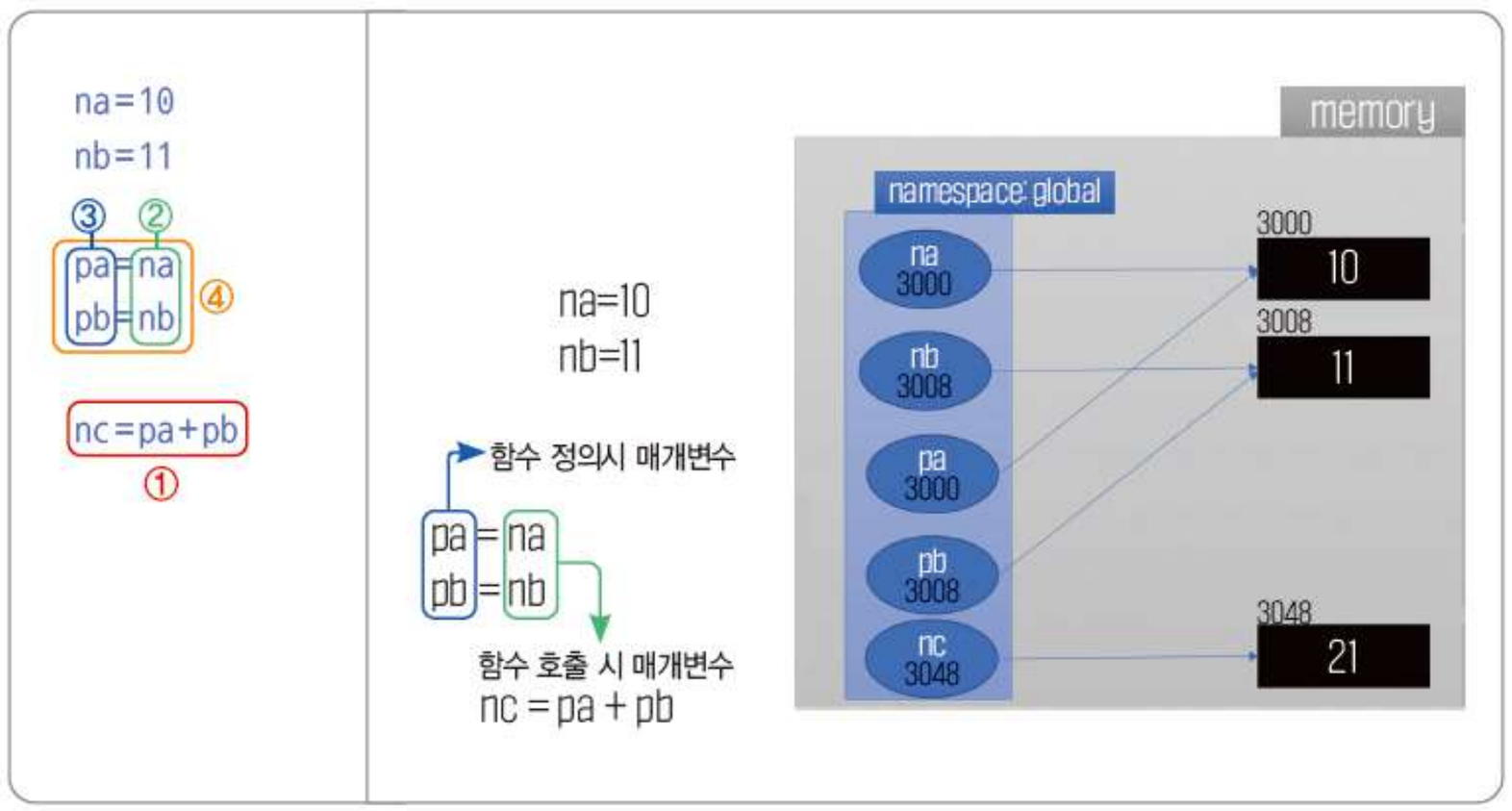
└ 실행화면

```
5
```

# 3. 매개변수와 함수 호출

## ❖ 매개변수가 있는 함수 호출

- 코드를 메인 메모리 관점으로 설명
  - Step1. 함수 정의 전 global 영역에서 각 변수의 기억공간 할당과 각 변수와 명령어가 함수로 정의 될 때 역할에 대한 설명



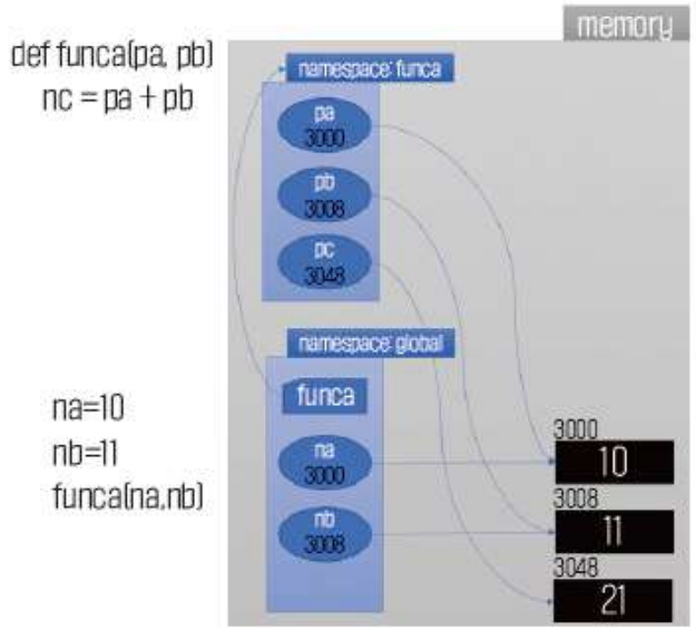
# 3. 매개변수와 함수 호출

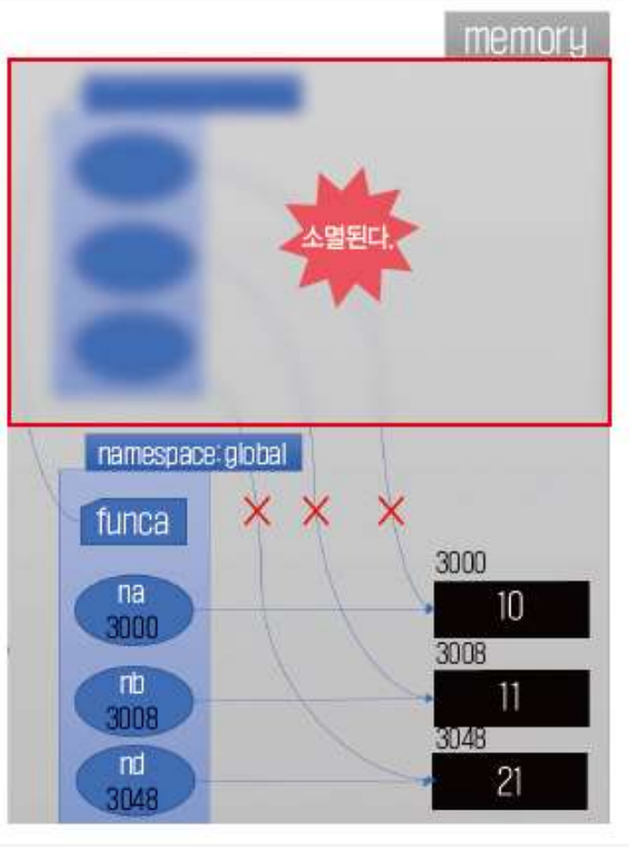
- **코드를 메인 메모리 관점으로 설명**
  - **Step 2. 함수로 정의한 프로그램의 메인 메모리 할당을 보면서 함수 이해하기**

```
def funca(pa, pb): # ------> ②
    nc=pa+pb # ----------> ③

na=10
nb=11

funca(na, nb) # ----------> ①
print(na, "+", nb, "=", nc) # -> ④
```

〈화면 출력〉

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

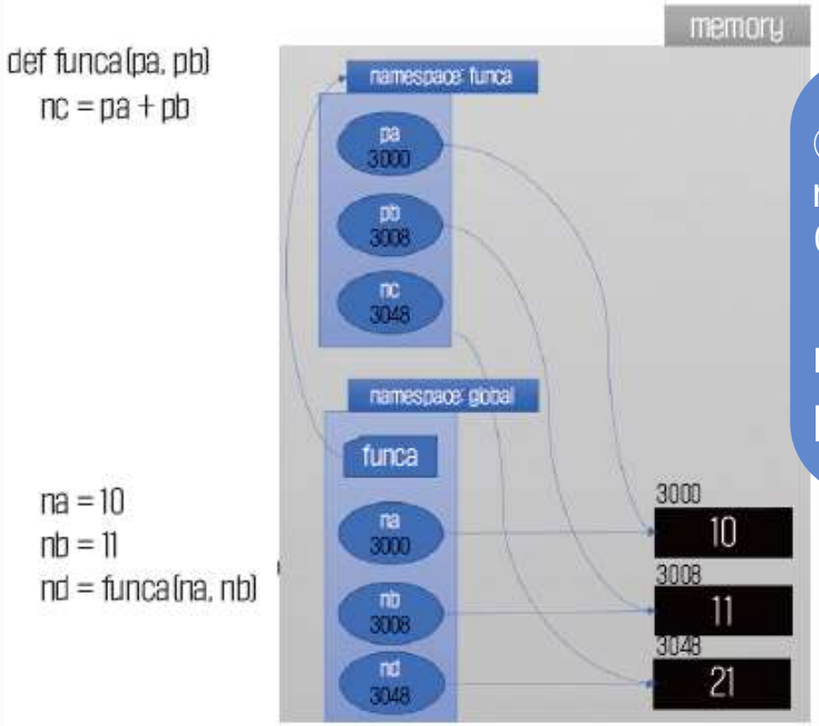


# 3. 매개변수와 함수 호출

- **코드를 메인 메모리 관점으로 설명**
  - **Step 3. 소멸하기 전에 (유산)을 남김. 이때 유산을 반환값을 의미**

```
def funca(pa, pb):
    nc=pa+pb
    return nc # ----------> ⑥

na=10
nb=11

nd=funca(na, nb) # ------> ①
print(na, "+", nb, "=", nd) # -> ⑤
```




⑥ 함수는 소멸 전 작업 결과를 return하여 호출한 프로그램에 남김.
예제에서 return되는 data는 nc 변수값

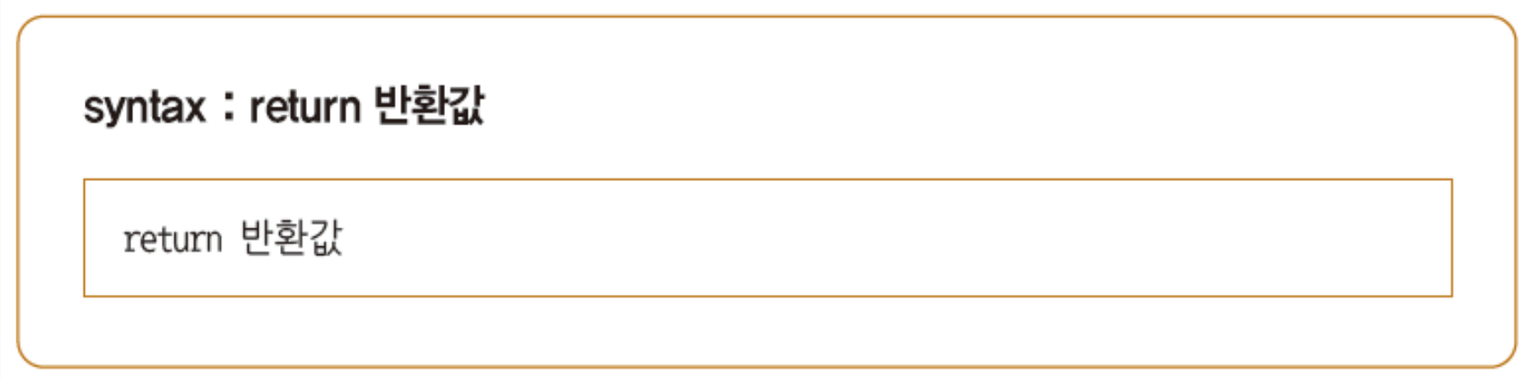
nc

```
nd = fadd(na, nb)  # -------> ①
print(na, "+", nb, " = ", nd)
```

# 4. return 키워드

## ❖ return 키워드

- **반환값**
  - 함수 실행 결과로 돌려주는 값(r_value)




- **함수의 종료를 의미**
  - return 키워드 이후 정의된 함수 코드 블록은 실행되지 않음

# 4. return 키워드

## ❖ 함수 반환값이 없을 때

- ①은 함수를 호출하고 반환값을 l_value에 할당
- 인수가 0인 경우 반환할 값이 없음
- 이 경우 함수는 호출자에게 None 을 반환

```
def fplusminus(arg):
    if arg > 0:
        return "plus"
    elif arg < 0:
        return "minus"

stra=fplusminus(0) # ---> ①
print(stra)
```

〈화면 출력〉

```
None
```

# 5. 함수 호출 순서

## ❖ 함수 호출 순서

- 함수는 호출 되기 전에 정의되어야 함
- 함수 호출 예제
  - 실행 시 다음과 같은 오류 발생
    - 함수 정의 전 호출을 시도했기 때문

```
myabs(10)

def myabs(arg):
    if(arg < 0):
        result=arg * -1
    else:
        result=arg
    return result
```

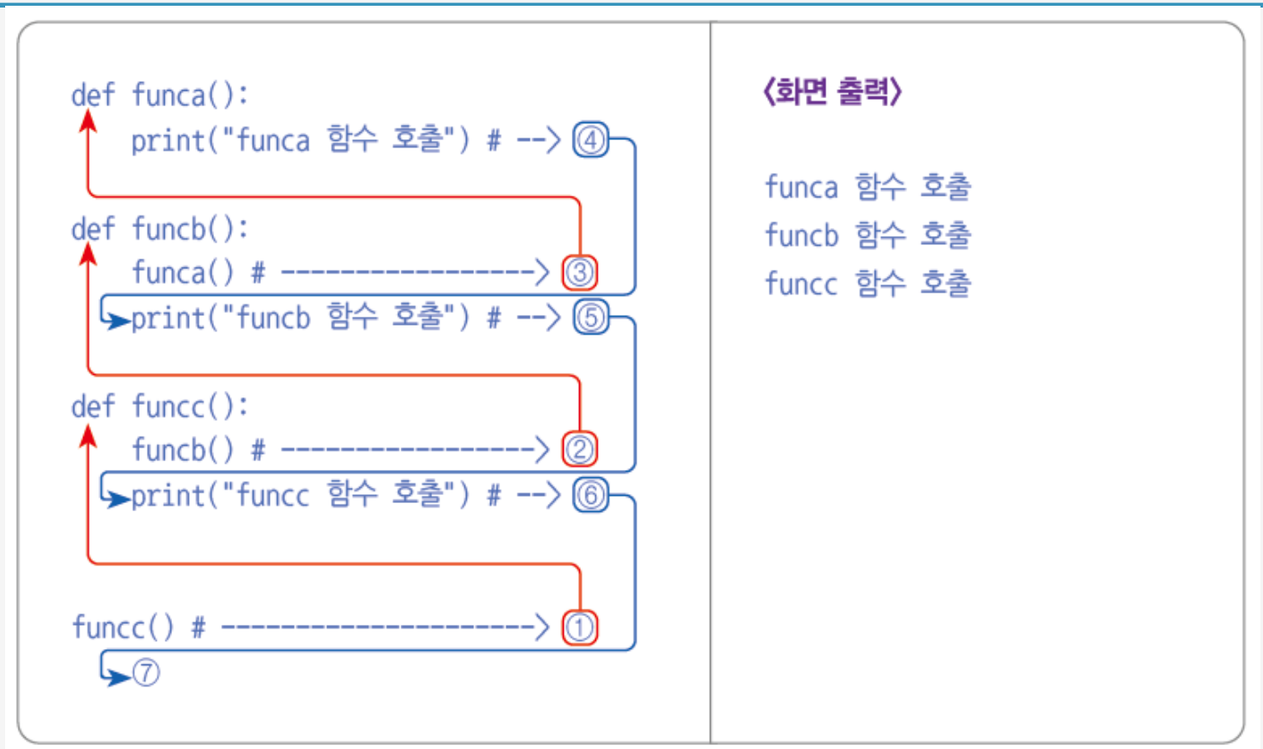
〈화면 출력〉

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

00

# 5. 함수 호출 순서

- 함수 호출 순서 코드




- ① funcc 함수 호출
- funcc 함수에서 ② funcb 함수 호출
- funcb 함수에서 ③ funca 함수 호출
- funca 함수에서 ④의 print 출력 후 제어권이 호출한 함수로 되돌아옴
- ⑤에서 print 화면 출력 후 제어권이 ⑥으로 넘어가고, print 출력 후 제어권이 ⑦로 돌아감
- 더 이상 명령문이 없으므로 프로그램 종료

00

# 5. 함수 호출 순서

- **함수 호출 순서 코드**
  - 두 개의 매개변수를 가진 덧셈을 구하는 함수(fadd)가 결과값을 반환하도록 정의하고, 인수가 10, 20인 fadd 함수를 호출하는 프로그램을 작성

```
def fadd(pa, pb):
    pc=pa+pb
    return pc

na=10
nb=20
nc=fadd(na, nb)
print(na, "+", nb, "결괏값은", nc,"이다.")
```

〈화면 출력〉

```
10+20 결괏값은 30 이다.
```

00

# 7. 연습문제

## ❖ 다음 코드의 출력 결과를 적어보세요

```
01: def welcome():
02:     print('이상한 나라에 오신 것을 환영합니다.')
03:
04: welcome()
```

↳

## ❖ 다음 코드의 출력 결과를 적어보세요

```
01: def welcome(name):
02:     print(name, '님 이상한 나라에 오신 것을 환영합니다.')
03:
04: welcome('앨리스')
05: welcome('도도새')
```

↳

00

# 7. 연습문제

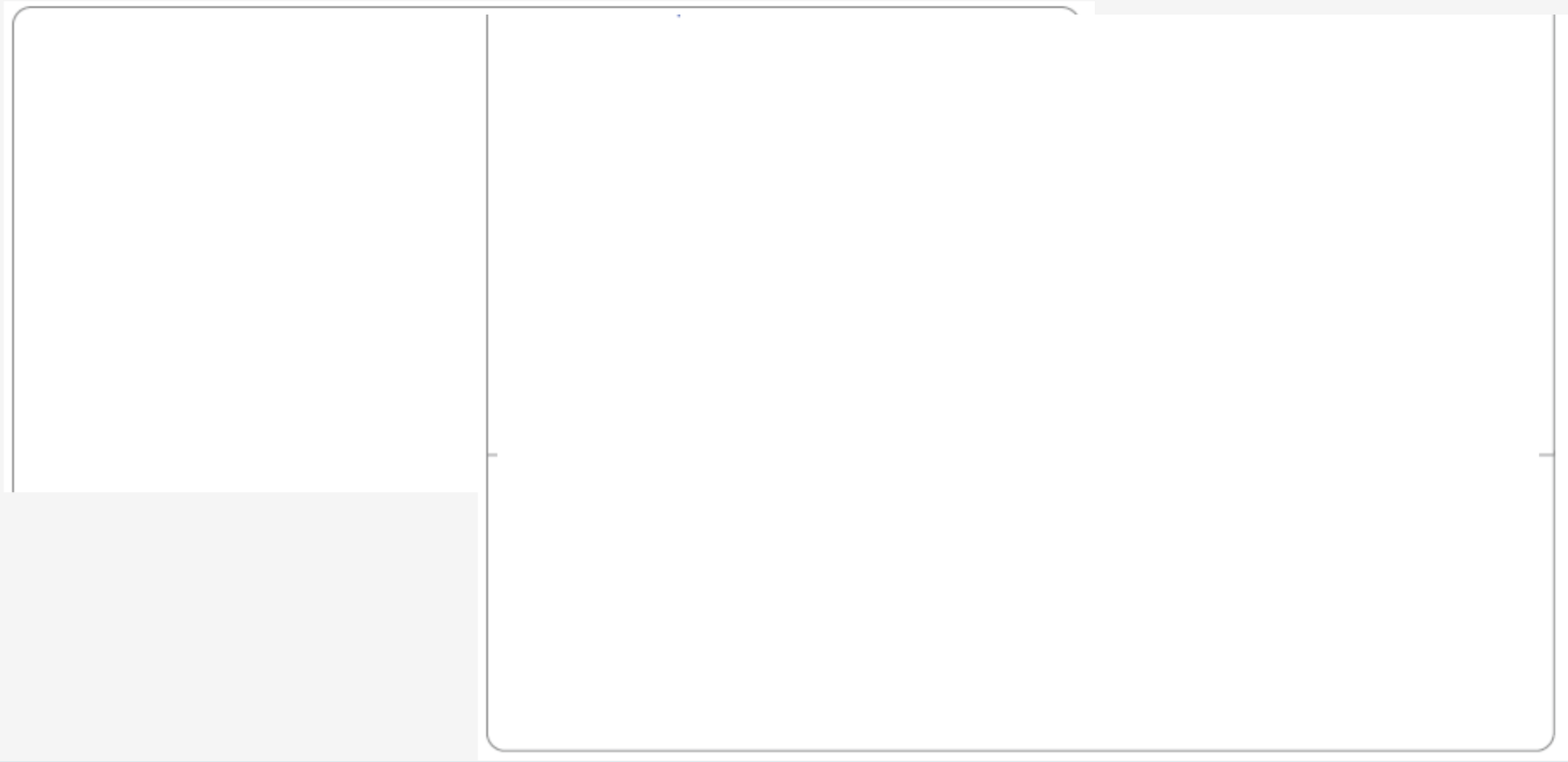
## ❖ 아래와 같은 출력 결과가 나오도록 빈칸을 채워보세요

```
01:  
02:      print('*' * num)
03:
04:  draw_stars(3)
05:  draw_stars(2)
06:  draw_stars(1)
```

```
↳    ***
     **
     *
```

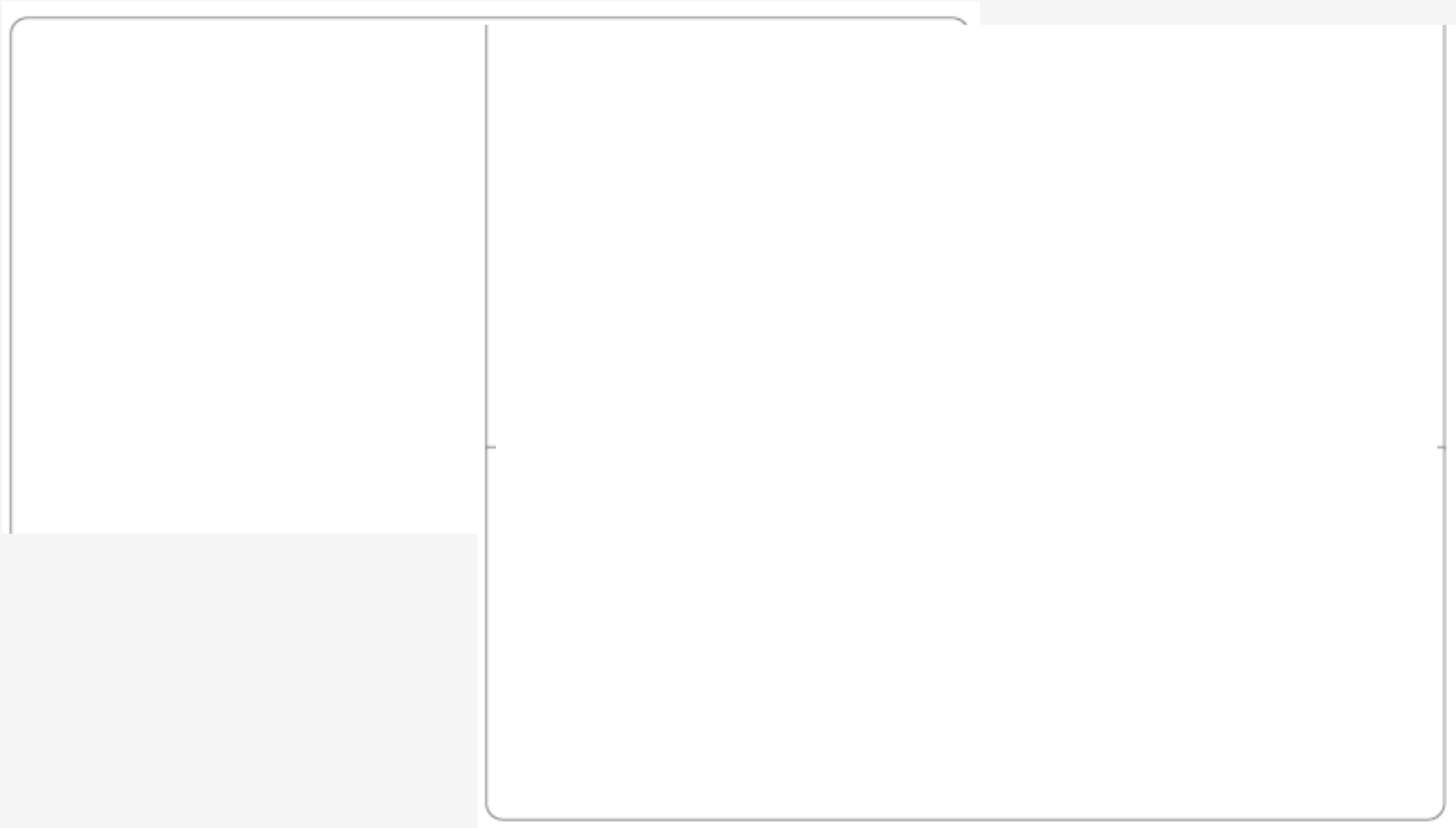
# 7. 연습문제

- 앞서 학습한 덧셈(fadd) 함수 프로그램에 뺄셈(fsub), 곱셈(fmul), 나눗셈(fdiv) 함수를 추가하여 사칙 연산을 하는 4개의 함수가 정의되어 있다. 함수 호출 시 매개변수 값은 100, 3으로 호출해서 각 결과를 반환하여 화면에 출력하도록 fsub, fmul, fdiv 함수를 추가하여 프로그램을 완성하세요.



# 7. 연습문제

- 앞서 작성한 사칙연산 함수 프로그램을 input 표준 입력 함수로 임의의 값을 입력 받아 매개변수 100과 3 대신 사용하도록 프로그램을 변경하세요.



00

# 7. 연습문제

- **문자열을 입력받아 해당 문자열의 길이를 반환하는 함수를 작성하세요.**
  - **즉, 아래 코드에서 len(sta) 대신에 사용할 string_length(stb) 함수를 정의하고 호출하여 동일한 결과가 나오도록 하세요.**

```
sta="python example"
lena=len(sta)
print(lena)
```

```
sta="python example"
def string_length(stb):

    count=0
    for chara in string:
        count=count+1
    return count

lena=string_length(sta)
print(lena)
```

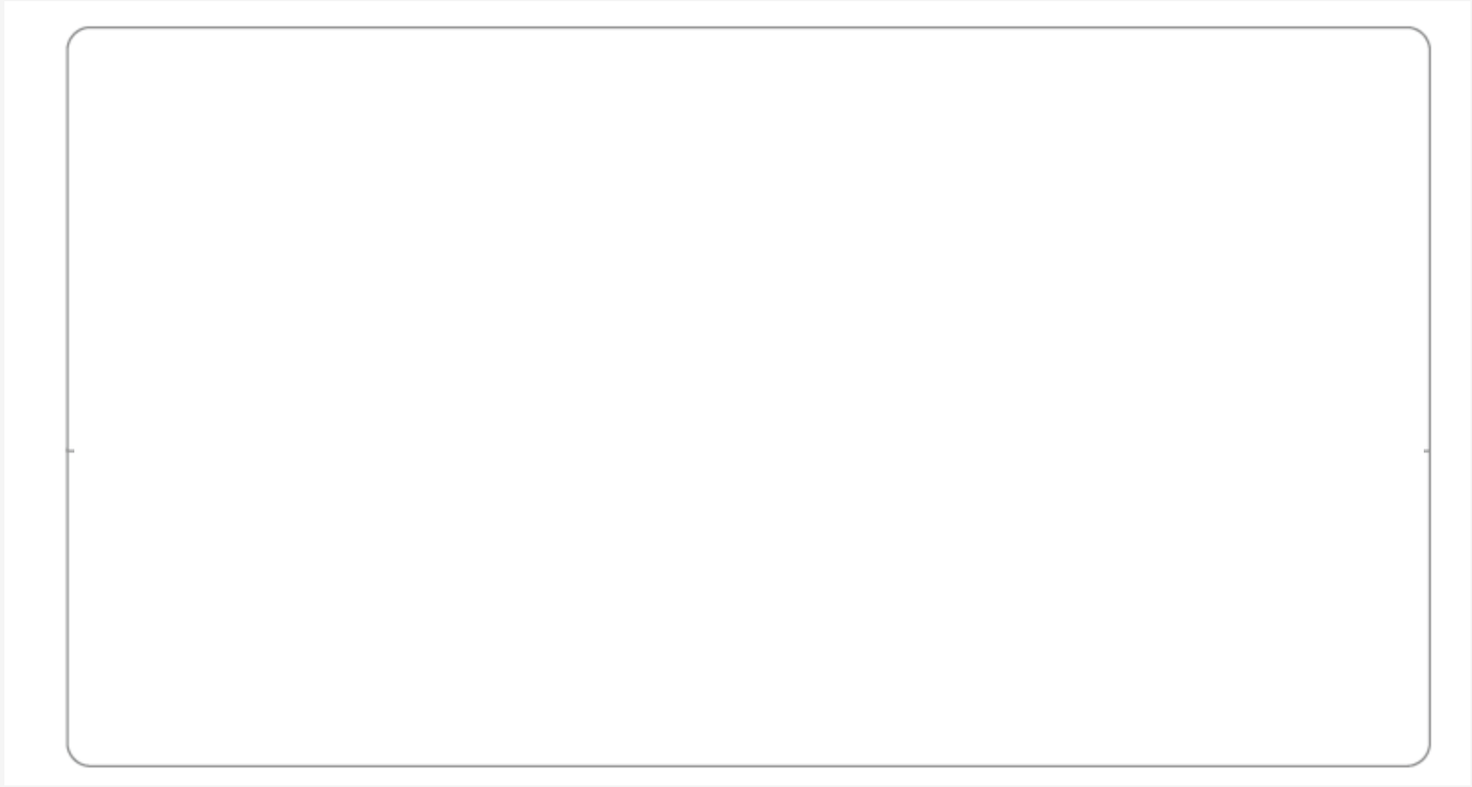
00

# 7. 연습문제

- 2개의 매개변수를 받아 첫 번째 변수를 두 번째 변수로 나눗셈을 하는 fdiv 함수를 작성하세요. (fdiv 함수를 호출 시 인수 2개를 표준 입력 함수 input 로 입력 받을 것.)
  - 단, 임의의 두 수 중 두 번째 매개변수가 0인 경우 “0으로는 나눌수 없다.” 메시지를 출력하고 그렇지 않은 경우는 나눈 결과를 출력하는 프로그램을 작성하세요.

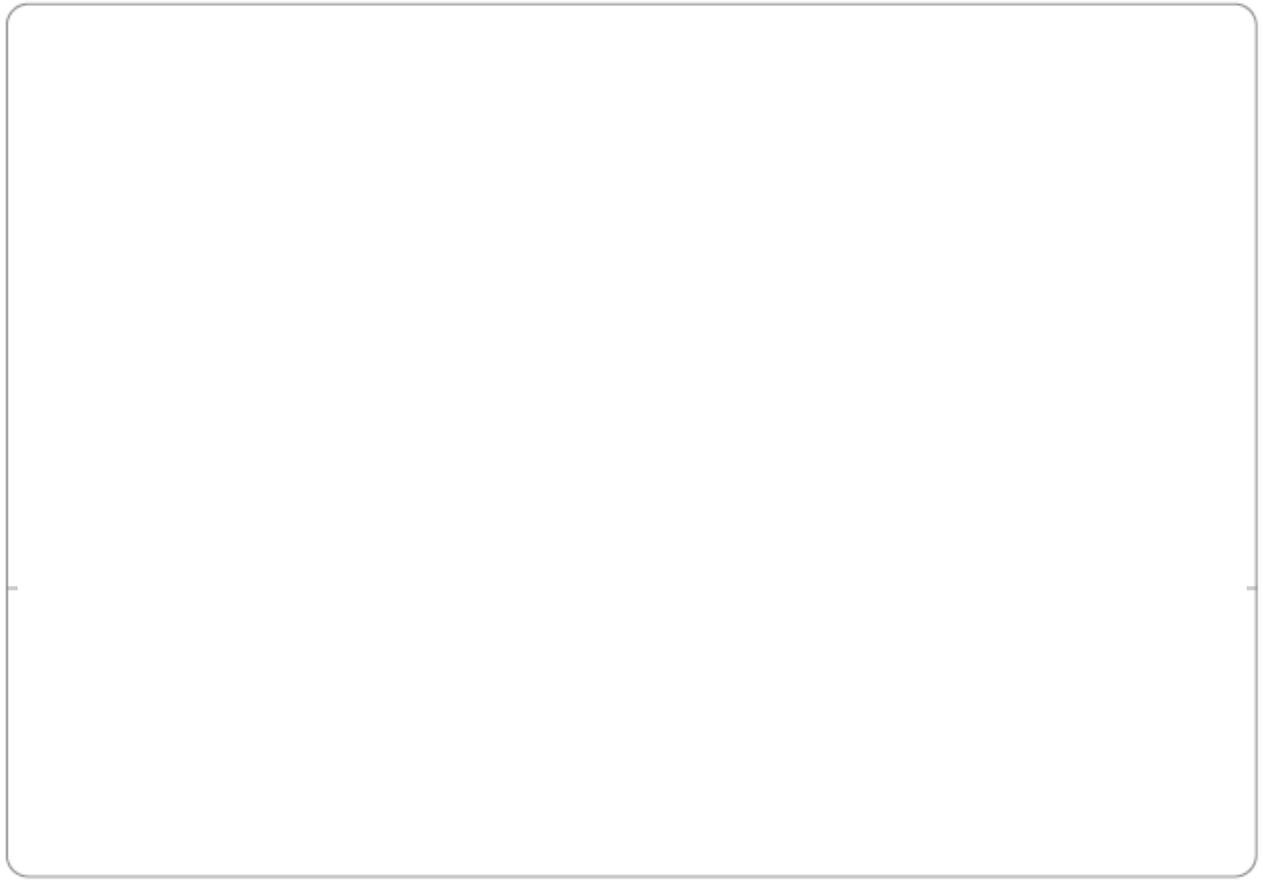
# 7. 연습문제

- 1부터 9까지의 정수 중 하나의 정수를 입력 받아 100까지 배수의 합을 구하는 프로그램을 작성하세요.(for문을 사용할 것.)
  - 배수의 합을 구하고자 하는 정수를 입력하세요.” 라고 화면에 출력하고, 3의 배수인 경우는 3을 입력 받으면 됨.



# 7. 연습문제

- 앞서 작성한 배수의 합을 구하는 프로그램을 수정하여, for 문을 funca 함수로 정의하고 호출하여 결과를 출력하도록 프로그램을 작성하시오.



# Notice

## ❖ 과제

- 1. 함수 정의 및 문법 설명하기
- 2. 함수 정의와 호출 구분하기
- 3. 매개변수 및 반환값 사용한 코드 작성하기
- 4. 코드 함수화 예시 생각하기
- 5. 다양한 매개변수 활용한 코드 작성하기

## ❖ 다음 수업 내용

- 함수
  - SWAP 함수, 전역변수, 지역변수, global 키워드, 디폴트 매개변수, 재귀 함수