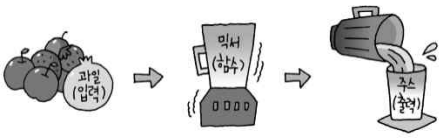
함수

- 함수



- 문법

```
def 함수명(매개변수):
    <수행할 문장1>
    <수행할 문장2>
     ...
```

- 예

```
def add(a, b):
    c = a + b
    return c
```

```
a = 3
b = 4
result = add(a, b)
print(result)
```


함수

- 매개변수와 인수
  - _____ : 함수에 전달된 값을 저장하는 변수
  - _____ : 함수를 호출할 때 전달하는 값
  - 예:

```
def add(a, b):      # a, b는 매개변수? 인수?
    return a+b

c = add(3, 4)       # 3, 4는 매개변수? 인수?
print(c)
```

함수

- 함수 유형 : 입력값 → 함수 → 리턴값

	리턴값O	리턴값X
입력값O	1	3
입력값X	2	4

- 유형 1 : 입력값과 리턴값이 모두 있는 경우

```
def 함수이름(매개변수1, 매개변수2, …):
  <수행할 문장 >
   ...
  return 리턴값
```

```
def add(a, b):
   c = a + b
   return c
```

```
리턴값 받을 변수 = 함수이름 (인수1, 인수2, …)
```

```
result = add(3, 4)
print(result)
```


- 유형 2 : 리턴값만 있는 경우

```
def 함수이름():
   <수행할 문장 >
    ...
   return 리턴값
```

```
def say():
    return 'Hi'
```

```
리턴값을 받을 변수 = 함수이름 ()
```

```
a = say()
print(a)
```

- 유형 3 : 입력값만 있는 경우

```
def 함수이름(매개변수1,…):
   <수행할 문장 >
    ...
```

```
def add(a, b):
   print(a+b)
```

```
함수이름 (인수1, …)
```

```
add(3, 4)
```

- 유형 4 : 입력값도 리턴값도 없는 함수

```
def 함수이름():
   <수행할 문장 >
    ...
```

```
def say():
    print('Hi')
```

```
함수이름 ()
```

```
a = say()
```

함수

- 매개변수 지정하여 호출하기
  - 매개변수를 지정하면 순서에 상관없이 사용할 수 있다는 장점
  - 예:

```
def sub(a, b):
    return a - b
```

```
result = sub(7, 3)
print(result)
```

```
result = sub(a=7, b=3)
print(result)
```

```
result = sub(b=3, a=7)
print(result)
```


함수

- 입력값의 개수를 모를 때
  - *매개변수로 작성하면 입력값을 전부 모아서 튜플로 만들어줌

```
def 함수명(*매개변수):
    <수행할 문장1>
    <수행할 문장2>
     ...
```

  - 예:

```
def add_many(*args):
    result = 0
    for i in args:
        result = result + i
    return result
```

```
result = add_many(1,2)
print(result)
```

```
result = add_many(1,2,3)
print(result)
```

함수

- 입력값의 개수를 모를 때
  - 예:

```
def add_mul(choice, *args):
    if choice == "add":
        result = 0
        for i in args:
            result = result + i
    elif choice == "mul":
        result = 1
        for i in args:
            result = result * i
    return result
```

```
result = add_mul('add', 1,2)
print(result)
```

```
result = add_mul('mul', 1,2,3)
print(result)
```


함수

- 키워드 매개변수 kwargs(keyword argrments)
  - **매개변수 : 매개변수 kwargs 는 딕셔너리가 되고 모든 key=value 형태의 입력값이 그 딕셔너리에 저장됨
  - 예:

```
def print_kwargs(**kwargs):
    print(kwargs)
```

```
print_kwargs(a=1)
```

```
print_kwargs(name='foo', age=3)
```

함수

- 함수의 리턴값은 언제나 하나이다
  - 예:

```
def add_mul(a,b):
    return a+b, a*b
```

```
result = add_mul(3,4)
print(result)
```

```
result1, result2 = add_mul(3,4)
print(result1, result2)
```

```
def add_mul(a,b):
    return a+b
    return a*b
```


함수

- 매개변수에 초기값 미리 설정하기
  - 주의!! 초기화시키고 싶은 매개변수는 항상 뒤쪽에 놓아야 함
  - 예

```
def say_myself(name, dept, year=2):
    print("이름 : %s, 학과 : %s , 학년 : %d " % (name, dept, year))
```

```
say_myself("홍길동", "소프트웨어공학과")
say_myself("김철수", "소프트웨어공학과", 3)
```

  - 예

```
def say_myself(name, year=2, dept):
    print("이름 : %s, 학과 : %s , 학년 : %d " % (name, dept, year))
```

```
say_myself("홍길동", "소프트웨어공학과")
```

함수

- 함수 안에서 선언한 변수의 효력 범위
  - 예

```
a = 1
def vartest(a):
    a = a +1

vartest(a)
print(a)
```

```
def vartest(a):
    a = a +1

vartest(a)
print(a)
```

함수 안에서 선언한 매개변수는 함수 안에서만 사용될 뿐 함수 밖에서는 사용되지 않는다!!

함수

- 함수 안에서 함수 밖의 변수를 변경하는 방법
  - 1. return 사용

```
a = 1
def vartest(a):
    a = a +1
    return a

a = vartest(a)
print(a)
```

  - 2. global 명령어

```
a = 1
def vartest():
    global a
    a = a+1

vartest()
print(a)
```

global 명령어는 사용하지 않는 것이 좋다.
왜냐하면 함수는 독립적으로 존재하는 것이 좋기 때문이다.
외부 변수에 종속적인 함수는 그다지 좋은 함수가 아니다.

함수

- lambda
  - 보통 함수를 한줄로 간결하게 만들 때 사용한다.
  - 형식 함수명 = lambda 매개변수 1, 매개변수 2, … : 매개변수를 이용한 표현식
  - 예

```
add = lambda a, b: a+b

result = add(3, 4)
print(result)
```

```
def add(a, b):
    return a+b

result = add(3, 4)
print(result)
```


- 예1 : 정수 입력받아 짝수/홀수 판단하여 출력

```
def chkEvenOdd(a):




num = int(input("숫자 : "))
chkEvenOdd(num)
```

```
숫자 : 4
숫자 4은 짝수
숫자 : 5
숫자 5은 홀수
```

- 예2 : 정수 입력받아 짝수/홀수 판단하여 출력

```
def chkEvenOdd(a):




num = int(input("숫자 : "))
result = chkEvenOdd(num)
print("숫자 %d은 %s" % (num, result))
```

```
숫자 : 4
숫자 4은 짝수
숫자 : 5
숫자 5은 홀수
```

- 예3 : 반지름(정수형)을 입력받아 원의 둘레 구해 출력

```
def calc_circle(r):



a = int(input("반지름 : "))
result = calc_circle(a)
print("반지름 : %d, 둘레 : %5.2f" % (a, result))
```

```
반지름 : 3
반지름 : 3, 둘레 : 18.84
```

- 예4 : 반지름(정수형)을 입력받아 원의 둘레와 넓이 구해 출력

```
def calc_circle(r):



a = int(input("반지름 : "))
calc_circle(a)
```

```
반지름 : 3
반지름 : 3, 둘레 : 18.84, 넓이 : 28.26
```


- 예5 : 리스트에 있는 값을 짝수와 홀수 갯수 세어 출력

```
def chkEvenOdd(a):









    print("리스트 : ", a)
    print("짝수 %d 개, 홀수 %d 개" % (cnt_even, cnt_odd))


num = [1,2,3,4,5]
chkEvenOdd(num)
```

```
리스트 :  [1, 2, 3, 4, 5]
짝수 2 개, 홀수 3 개
```