Immutable

(정수, 실수, 문자열, 튜플)

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

vartest(a)
print(a)
```

Mutable

(리스트, 딕셔너리, 집합)

```
b = [1,2,3]
def vartest2(b):
    b = b.append(4)

vartest2(b)
print(b)
```

점프 투 파이썬

05장 파이썬 날개달기

지은이: 박응용

강의: 조코딩

클래스,
모듈, 패키지,
예외 처리,
내장 함수, 외장 함수

클래스

반복되는
변수 & 메서드(함수)를
미리 정해놓은 틀(설계도)

05-1 파이썬 프로그래밍의 핵심, 클래스

클래스는 도대체 왜 필요한가?

```
result = 0
def add(num):
    global result
    result += num
    return result

print(add(3))
print(add(4))
```

05-1 파이썬 프로그래밍의 핵심, 클래스

두 개의 계산기

```
result1 = 0
result2 = 0

def add1(num):
    global result1
    result1 += num
    return result1

def add2(num):
    global result2
    result2 += num
    return result2

print(add1(3))
print(add1(4))
print(add2(3))
print(add2(7))
```

05-1 파이썬 프로그래밍의 핵심, 클래스

클래스를 이용한 계산기

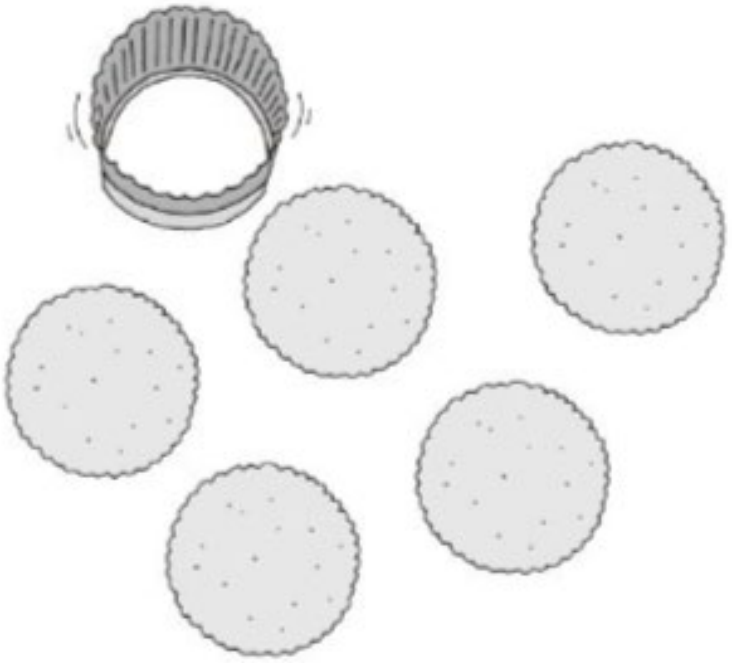
```
class Calculator:
    def __init__(self):
        self.result = 0

    def add(self, num):
        self.result += num
        return self.result

cal1 = Calculator()
cal2 = Calculator()

print(cal1.add(3))
print(cal1.add(4))
print(cal2.add(3))
print(cal2.add(7))
```

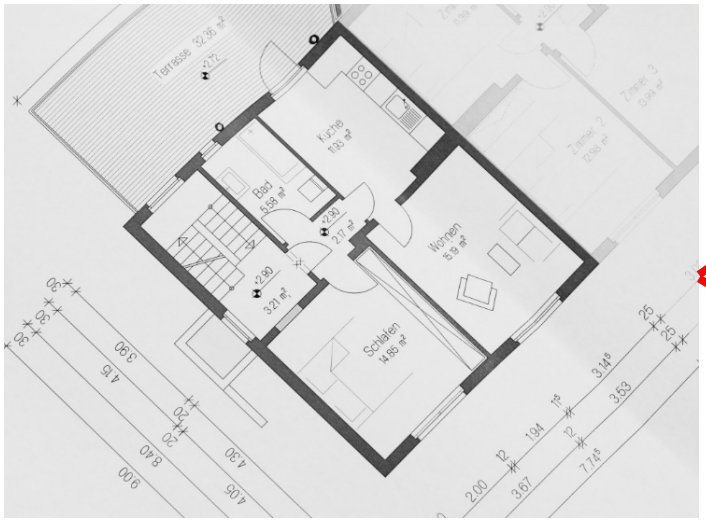
05-1 파이썬 프로그래밍의 핵심, 클래스



과자 틀(클래스)과 이 틀로 찍어 만든 과자(객체)

과자 틀(클래스), 과자(객체)

05-1 파이썬 프로그래밍의 핵심, 클래스

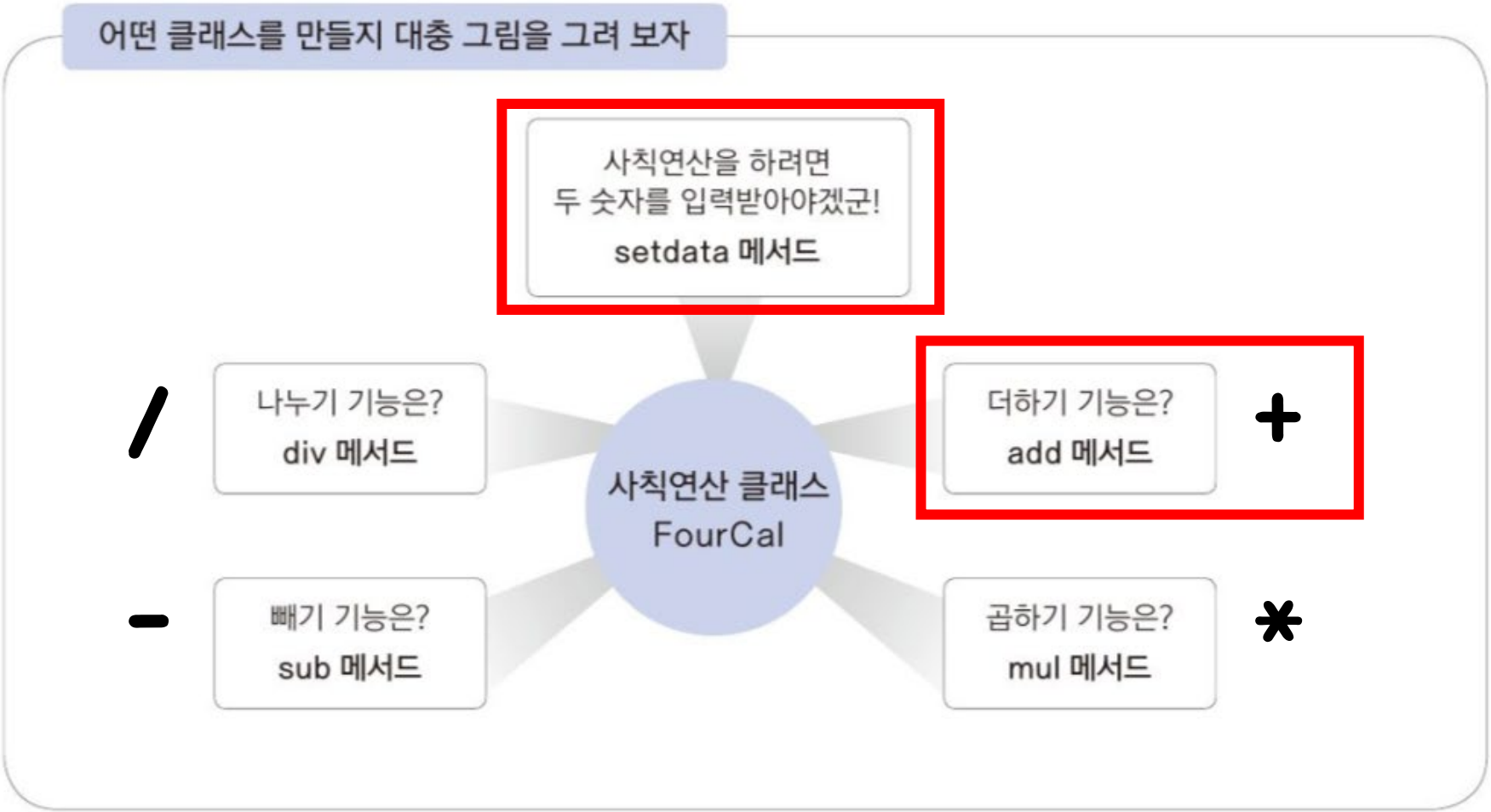


설계 도면(클래스)

건물(객체)

05-1 파이썬 프로그래밍의 핵심, 클래스

사칙연산 클래스 만들기(+,-,*,/)



05-1 파이썬 프로그래밍의 핵심, 클래스

사칙연산 클래스 1

```
class FourCal:
   pass
```

```
a = FourCal()
Print(type(a))
<class '__main__.FourCal'>
```

05-1 파이썬 프로그래밍의 핵심, 클래스

사칙연산 클래스 2

```
a.setdata(4, 2)
```

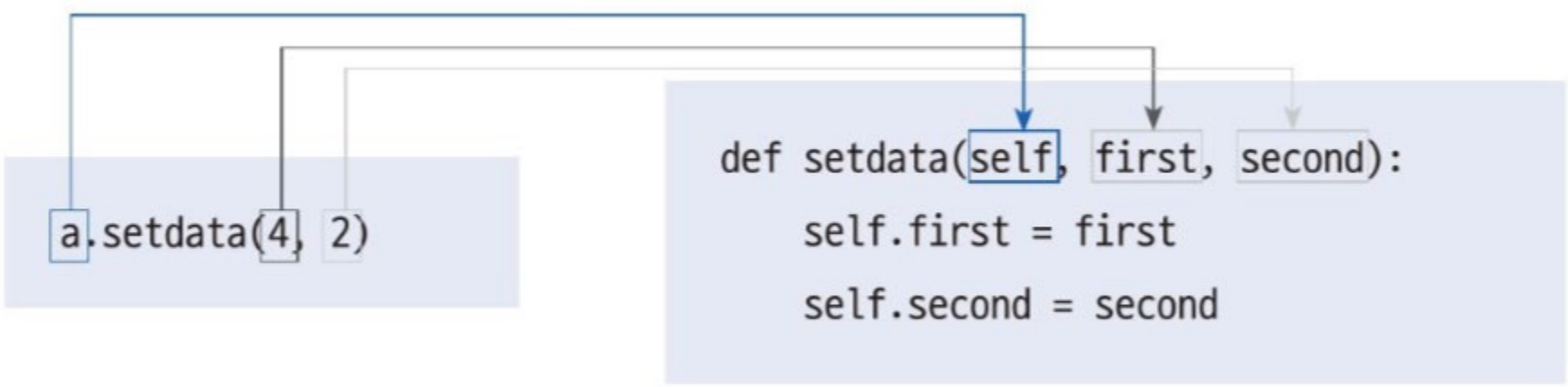
```
class FourCal:
    def setdata(self, first, second):
        self.first = first
        self.second = second

a = FourCal()
a.setdata(4,2)

print(a.first)
print(a.second)
```

05-1 파이썬 프로그래밍의 핵심, 클래스

사칙연산 클래스 2



05-1 파이썬 프로그래밍의 핵심, 클래스

사칙연산 클래스 3

```
class FourCal:
    def setdata(self, first, second):
        self.first = first
        self.second = second
    def add(self):
        result = self.first + self.second
        return result

a = FourCal()
a.setdata(4, 2)
print(a.add())
6
```

05-1 파이썬 프로그래밍의 핵심, 클래스

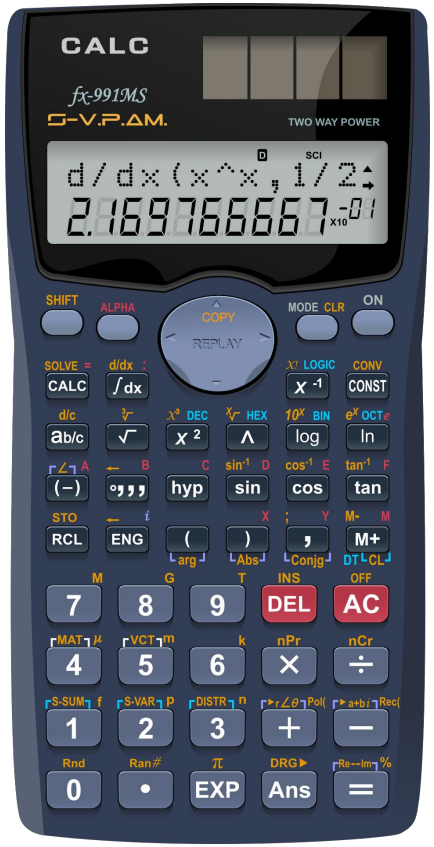
생성자(Constructor)

```
class FourCal:
    def __init__(self, first, second):
        self.first = first
        self.second = second
    def setdata(self, first, second):
        self.first = first
        self.second = second
    def add(self):
        result = self.first + self.second
        return result

a = FourCal()
```

05-1 파이썬 프로그래밍의 핵심, 클래스

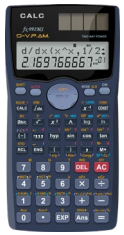
클래스의 상속



05-1 파이썬 프로그래밍의 핵심, 클래스

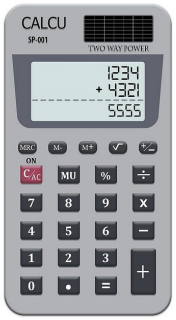
클래스의 상속

FourCal



MoreFourCal1

MoreFourCal2



05-1 파이썬 프로그래밍의 핵심, 클래스

클래스의 상속

```
class MoreFourCal(FourCal):
    pass
```

```
a = MoreFourCal(4, 2)
a.add()
6
```

05-1 파이썬 프로그래밍의 핵심, 클래스

클래스의 상속 2 - 메서드 추가

```
class MoreFourCal(FourCal):
    def pow(self):
        result = self.first ** self.second
        return result
```

```
a = MoreFourCal(4, 2)
a.pow()
16
```

05-1 파이썬 프로그래밍의 핵심, 클래스

메서드 오버라이딩

```
class SafeFourCal(FourCal):
    def div(self):
        if self.second == 0:
            return 0
        else:
            return self.first / self.second
```

05-1 파이썬 프로그래밍의 핵심, 클래스

클래스 변수, 객체 변수

```
class Family:
    lastname = "김"

Family.lastname = "박"
print(Family.lastname)

a = Family()
b = Family()
print(a.lastname)
print(b.lastname)
```

모듈

모듈이란?
= 미리 만들어 놓은 .py 파일
(함수, 변수, 클래스)

05-2 모듈

모듈 만들고 불러 보기

```
# C:\jocoding\mod1.py
def add(a, b):
    return a + b
```

```
cd C:\jocoding
```

```
import mod1
print(mod1.add(3,4))
7
```

05-2 모듈

from 모듈이름 import 모듈함수

```
from mod1 import add
print(add(3,4))
7
```

05-2 모듈

if __name__ == "__main__": 의 의미

```
#mod1.py
def add(a,b):
    return a + b

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

if __name__ == "__main__":
    print(add(1,4))
    print(add(4,2))
```

05-2 모듈

sys.path.append

```
import sys
print(sys.path)
['', 'C:\\Windows\\SYSTEM32\\python35.zip',
'c:\\Python35\\DLLs',
'c:\\Python35\\lib', 'c:\\Python35',
'c:\\Python35\\lib\\site-packages']
```

```
import sys
sys.path.append("C:\\jocoding\\subFolder")
import mod1
print(mod1.add(3,4))
```

패키지

패키지란?

= 모듈 여러 개 모아놓은 것

05-3 패키지

가상의 game 패키지 예

```
game/
    __init__.py
    sound/
        __init__.py
        echo.py
    graphic/
        __init__.py
        render.py
```

05-3 패키지

테스트를 위해 패키지 만들기

```
C:/jocoding/game/__init__.py
C:/jocoding/game/sound/__init__.py
C:/jocoding/game/sound/echo.py
C:/jocoding/game/graphic/__init__.py
C:/jocoding/game/graphic/render.py
```

```
# echo.py
def echo_test():
    print ("echo")
```

```
# render.py
def render_test():
    print ("render")
```

05-3 패키지

패키지 안의 함수 실행하기

```
import game.sound.echo
game.sound.echo.echo_test()
echo
```

```
from game.sound import echo
echo.echo_test()
echo
```

```
from game.sound.echo import echo_test
echo_test()
echo
```

05-3 패키지

__all__

```
from game.sound import *
echo.echo_test()

Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
NameError: name 'echo' is not defined
```

```
# C:/jocoding/game/sound/__init__.py
__all__ = ['echo']
```

05-3 패키지

relative 패키지

```
# render.py
from ..sound.echo import echo_test

def render_test():
    print ("render")
    echo_test()
```

예외처리

오류가 발생했을때
어떻게 할지 정하는 것

```
try:
    #오류가 발생할 수 있는 구문
except Exception as e:
    #오류 발생
else:
    #오류 발생하지 않음
finally:
    #무조건 마지막에 실행
```

05-4 예외처리

오류는 어떤 때 발생하는가?

```
f = open("나없는파일", 'r')


Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
FileNotFoundError: [Errno 2] No such file or directory:
'나없는파일'
```

05-4 예외처리

try, except문

```
try:
    ...
except [발생 오류[as 오류 메시지 변수]]:
    ...
```

```
try:
    4 / 0
except ZeroDivisionError as e:
    print(e)
```

05-4 예외처리

try .. else

```
try:
    f = open('foo.txt', 'r')
except FileNotFoundError as e:
    print(str(e))
else:
    data = f.read()
    print(data)
    f.close()
```

05-4 예외처리

try .. finally

```
f = open('foo.txt', 'w')
try:
    # 무언가를 수행한다.
finally:
    f.close()
```

```
f = open('foo.txt', 'w')
try:
    # 무언가를 수행한다.
    data = f.read()
    print(data)
except Exception as e:
    print(e)
finally:
    f.close()
```

05-4 예외처리

여러 개의 오류 처리하기

```
try:
    a = [1,2]
    print(a[3])
    4/0
except ZeroDivisionError:
    print("0으로 나눌 수 없습니다.")
except IndexError:
    print("인덱싱할 수 없습니다.")
```

05-4 예외처리

오류 회피하기

```
try:
    f = open("나없는파일", 'r')
except FileNotFoundError:
    pass
```

05-4 예외처리

오류 일부러 발생시키기

```
class Bird:
    def fly(self):
        raise NotImplementedError
```

```
class Eagle(Bird):
    def fly(self):
        print("very fast")

eagle = Eagle()
eagle.fly()
```

내장함수

파이썬에서 기본적으로
포함하고 있는 함수
ex) print(), type()

05-5 내장함수

abs => 절대값

```
>>> abs(3)
3
>>> abs(-3)
3
>>> abs(-1.2)
1.2
```

05-5 내장함수

all => 모두 참인지 검사

```
>>> all([1, 2, 3])
True
>>> all([1, 2, 3, 0])
False
```

05-5 내장함수

any => 하나라도 참이 있는가?

```
>>> any([1, 2, 3, 0])
True
>>> any([0, ""])
False
```

05-5 내장함수

chr => ASCII 코드를 입력받아 문자 출력

```
>>> chr(97)
'a'
>>> chr(48)
'0'
```

* ASCII(아스키 코드) = 0~127사이의 숫자를 각 문자에 대응

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[ENG OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

05-5 내장함수

dir => 자체적으로 가지고 있는 변수나 함수를 보여줌

```
>>> dir([1, 2, 3])
['append', 'count', 'extend', 'index', 'insert', 'pop',...]
>>> dir({'1':'a'})
['clear', 'copy', 'get', 'has_key', 'items', 'keys',...]
```

05-5 내장함수

divmod => 몫과 나머지를 튜플 형태로 돌려줌

```
>>> divmod(7, 3)
(2, 1)
>>> divmod(1.3, 0.2)
(6.0, 0.09999999999999978)
```

05-5 내장함수

enumerate => 열거하다

```
for i, name in enumerate(['body', 'foo', 'bar']):
    print(i, name)

0 body
1 foo
2 bar
```

05-5 내장함수

eval => 실행 후 결과값을 돌려줌

```
>>> eval('1+2')
3
>>> eval("'hi' + 'a'")
'hia'
>>> eval('divmod(4, 3)')
(1, 1)
```

05-5 내장함수

filter => 함수를 통과하여 참인 것만 돌려줌

```
def positive(x):
    return x > 0

a = list(filter(positive, [1, -3, 2, 0, -5, 6]))
print(a)
```

```
a = list(filter(lambda x: x > 0, [1, -3, 2, 0, -5, 6]))
print(a)
```

05-5 내장함수

id => 주소 값

```
>>> a = 3
>>> id(3)
135072304
>>> id(a)
135072304
>>> b = a
>>> id(b)
135072304
```

05-5 내장함수

input => 사용자 입력 받는 함수

```
>>> a = input()
hi
>>> a
'hi'
>>> b = input("Enter: ")
Enter: hi
```

```
>>> b
'hi'
```

05-5 내장함수

int => 10진수 정수 형태로 변환

```
>>> int('3')
3
>>> int(3.4)
3
```

```
>>> int('11', 2)
3
>>> int('1A', 16)
26
```

05-5 내장함수

len => 길이

```
>>> len("python")
6
>>> len([1,2,3])
3
>>> len((1, 'a'))
2
```

05-5 내장함수

list => 리스트로 변환

```
>>> list("python")
['p', 'y', 't', 'h', 'o', 'n']
>>> list((1,2,3))
[1, 2, 3]
```

```
>>> a = [1, 2, 3]
>>> b = list(a)
>>> b
[1, 2, 3]
```

05-5 내장함수

map => 각 요소가 수행한 결과를 돌려줌

```
def two_times(x): return x*2
a = list(map(two_times, [1, 2, 3, 4]))
print(a)
```

```
a = list(map(lambda a: a*2, [1, 2, 3, 4]))
print(a)
```

05-5 내장함수

max => 최대 값

```
>>> max([1, 2, 3])
3
>>> max("python")
'y'
```

min => 최소 값

```
>>> min([1, 2, 3])
1
>>> min("python")
'h'
```

05-5 내장함수

open

mode	설명
w	쓰기 모드로 파일 열기
r	읽기 모드로 파일 열기
a	추가 모드로 파일 열기
b	바이너리 모드로 파일 열기

```
>>> f = open("binary_file", "rb")
```

05-5 내장함수

pow => 제곱한 결과값 반환

```
>>> pow(2, 4)
16
>>> pow(3, 3)
27
```

05-5 내장함수

range => 범위

```
>>> list(range(5))
[0, 1, 2, 3, 4]
```

```
>>> list(range(5, 10))
[5, 6, 7, 8, 9]
```

```
>>> list(range(1, 10, 2))
[1, 3, 5, 7, 9]
>>> list(range(0, -10, -1))
[0, -1, -2, -3, -4, -5, -6, -7, -8, -9]
```

05-5 내장함수

round => 반올림

```
round(4.6)
>> 5
round(4.2)
>> 4

round(5.678, 2)
>> 5.68
```

05-5 내장함수

sorted => 정렬

```
>>> sorted([3, 1, 2])
[1, 2, 3]
>>> sorted(['a', 'c', 'b'])
['a', 'b', 'c']
>>> sorted("zero")
['e', 'o', 'r', 'z']
>>> sorted((3, 2, 1))
[1, 2, 3]
```

05-5 내장함수

str => 문자열 반환

```
>>> str(3)
'3'
>>> str('hi')
'hi'
>>> str('hi'.upper())
'HI'
```

05-5 내장함수

tuple => 튜블 반환

```
>>> tuple("abc")
('a', 'b', 'c')
>>> tuple([1, 2, 3])
(1, 2, 3)
>>> tuple((1, 2, 3))
(1, 2, 3)
```

05-5 내장함수

type => 타입을 출력

```
>>> type("abc")
<class 'str'>
>>> type([ ])
<class 'list'>
>>> type(open("test", 'w'))
<class '_io.TextIOWrapper'>
```

05-5 내장함수

zip => 자료형을 묶어주는 역할

```
>>> list(zip([1, 2, 3], [4, 5, 6]))
[(1, 4), (2, 5), (3, 6)]
>>> list(zip([1, 2, 3], [4, 5, 6], [7, 8, 9]))
[(1, 4, 7), (2, 5, 8), (3, 6, 9)]
>>> list(zip("abc", "def"))
[('a', 'd'), ('b', 'e'), ('c', 'f')]
```

외장함수

라이브러리 함수,
import 해서 쓰는 것

05-6 외장함수

sys.argv

```
# argv_test.py
import sys
print(sys.argv)
```

```
C:/Python/Mymodules>python argv_test.py you need python
['argv_test.py', 'you', 'need', 'python']
```

05-6 외장함수

pickle

```
import pickle
f = open("test.txt", 'wb')
data = {1: 'python', 2: 'you need'}
pickle.dump(data, f)
f.close()
```

```
import pickle
f = open("test.txt", 'rb')
data = pickle.load(f)
print(data)
{2:'you need', 1:'python'}
```

05-6 외장함수

time

```
import time
print(time.time())
#1970년 1월 1일 0시 0분 0초를 기준으로 지난 시간 초
```

```
print(time.localtime(time.time()))
time.struct_time(tm_year=2013, tm_mon=5, tm_mday=21,
tm_hour=16,
    tm_min=48, tm_sec=42, tm_wday=1, tm_yday=141, tm_isdst=0)
```

05-6 외장함수

time.sleep

```
#sleep1.py
import time
for i in range(5):
    print(i)
    time.sleep(1)
```

05-6 외장함수

random

```
import random
print(random.random())
# 0~1 사이의 난수
```

```
random.randint(1, 10)
6
```

```
data = [1, 2, 3, 4, 5]
random.shuffle(data)
data
[5, 1, 3, 4, 2]
```

```
import random
lotto = sorted(random.sample(range(1,46), 6))
print(lotto)
```

05-6 외장함수

webbrowser

```
import webbrowser
webbrowser.open("http://google.com")
```

```
webbrowser.open_new("http://google.com")
```