기초 PYTHON 프로그래밍

20. 클래스와 객체

1. 객체 (object)
2. 클래스 (class)
3. 연산자 중복
4. 클래스와 모듈

서강대학교
SOGANG UNIVERSITY

1. 객체 (object)

서강대학교 SOGANG UNIVERSITY

◆ 파이썬에서 모든 데이터는 객체이다.

```
>>> a = 10; b = 3.5; c = 2+10j; d = True
>>> type(a); type(b); type(c); type(d)
<class 'int'>
<class 'float'>
<class 'complex'>
<class 'bool'>
```

```
>>> L = [1,2,3]; type(L)
<class 'list'>
>>> T = (1,2,3); type(T)
<class 'tuple'>
>>> S = 'hello'; type(S)
<class 'str'>
>>> A = {1,2,3}; type(A)
<class 'set'>
>>> D = {1:20, 2:25, 3:22}; type(D)
<class 'dict'>
```

1. 객체 (object)

서강대학교 SOGANG UNIVERSITY

- 파이썬에서 모든 함수는 객체이다.

```
>>> def test():
        print('hello world')

>>> type(test)
<class 'function'>

>>> type(range)
<class 'type'>

>>> type(input)
<class 'builtin_function_or_method'>

>>> type(print)
<class 'builtin_function_or_method'>
```

2. 클래스

서강대학교 SOGANG UNIVERSITY

- 클래스 (class)
  - 객체(object)를 만들기 위한 도구
  - 클래스를 이용하면 현실 세계의 모든 물체들을 객체로 만들 수 있다.
  - 클래스의 구성

속성	객체를 구성하는 데이터
메소드	속성에 대해 어떤 기능을 수행하는 함수
생성자, 소멸자	객체 생성과 소멸 시에 자동 호출되는 특별한 메소드
연산자 중복	연산자(+, – 등) 기호를 이용하여 표현할 수 있도록 함

생성자는 **def __init__(self,…):** 으로 정의함

소멸자는 **def __del__(self,…):** 으로 정의함

2. 클래스

서강대학교 SOGANG UNIVERSITY

◆ 강아지를 클래스를 이용하여 객체로 표현하기

```
class Dog :
    """ 강아지를 이름과 나이로 표현하는 클래스 """
    """ 속성은 강아지 객체를 구성하는 데이터임. """
```

속성

```
    def __init__(self, name, age):
        """ 강아지 객체를 생성하는 생성자 메소드 """
        self.name = name
        self.age = age
```

생성자

```
    def bark(self):
        print(self.name, 'is barking')
```

메소드 (method)

2. 클래스

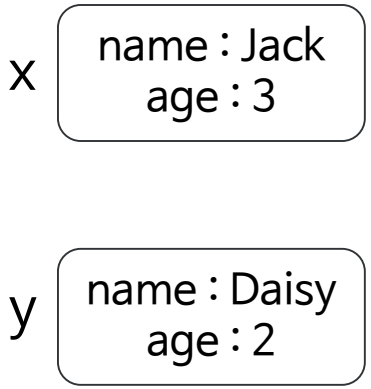
서강대학교 SOGANG UNIVERSITY

```
class Dog :

    def __init__(self, name, age):
        self.name = name
        self.age = age

    def bark(self):
        print(self.name, 'is barking')

x = Dog('Jack', 3)
y = Dog('Daisy', 2)
x.bark()
y.bark()
print(x.name, 'is', x.age, 'years old.')
print(y.name, 'is', y.age, 'years old.')
```




```
Jack is barking
Daisy is barking
Jack is 3 years old.
Daisy is 2 years old.
```

서강대학교 SOGANG UNIVERSITY

◆ 원(circle)을 클래스를 이용하여 객체로 표현하기

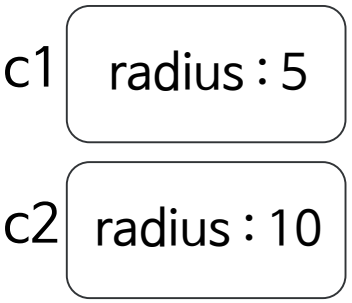
```
class circle :

    def __init__(self, radius):
        self.radius = radius

    def area(self):
        a = 3.14 * pow(self.radius, 2)
        return a

    def perimeter(self):
        p = 2 * 3.14 * self.radius
        return p
```

```
c1 = circle(5)
c2 = circle(10)
```




```
print('c1 area :', c1.area())
print('c1 perimeter : %.2f' % c1.perimeter())

print('c2 area :', c2.area())
print('c2 perimeter : %.2f' % c2.perimeter())
```

```
c1 area : 78.5
c1 perimeter : 31.40
c2 area : 314.0
c2 perimeter : 62.80
```

2. 클래스

서강대학교 SOGANG UNIVERSITY

◆ list 클래스

```
>>> dir(list)
['__add__', '__class__', '__contains__', '__delattr__',
'__delitem__', '__dir__', '__doc__', '__eq__',
'__format__', '__ge__', '__getattribute__',
'__getitem__', '__gt__', '__hash__', '__iadd__',
'__imul__', '__init__', '__iter__', '__le__', '__len__',
'__lt__', '__mul__', '__ne__', '__new__', '__reduce__',
'__reduce_ex__', '__repr__', '__reversed__',
'__rmul__', '__setattr__', '__setitem__', '__sizeof__',
'__str__', '__subclasshook__', 'append', 'clear',
'copy', 'count', 'extend', 'index', 'insert', 'pop',
'remove', 'reverse', 'sort']
```

```
>>> L1 = [1,3,5]
>>> L2 = list([2,4,6])


>>> L1.append(7)
>>> print(L1)
[1, 3, 5, 7]
>>> L2.pop()
6
>>> print(L2)
[2, 4]
```

3. 연산자 중복

서강대학교 SOGANG UNIVERSITY

◆ int 클래스

```
>>> dir(int)
['__abs__', '__add__', '__and__', '__bool__', '__ceil__',
'__class__', '__delattr__', '__dir__', '__divmod__',
'__doc__', '__eq__', '__float__', '__floor__',
'__floordiv__', '__format__', '__ge__',
'__getattribute__', '__getnewargs__', '__gt__',
'__hash__', '__index__', '__init__', '__int__',
'__invert__', '__le__', '__lshift__', '__lt__', '__mod__',
'__mul__', '__ne__', '__neg__', '__new__', '__or__',
'__pos__', '__pow__', '__radd__', '__rand__',
'__rdivmod__', '__reduce__', '__reduce_ex__',
'__repr__', '__rfloordiv__', '__rlshift__', '__rmod__',
'__rmul__', '__ror__', '__round__', '__rpow__',
'__rrshift__', '__rshift__', '__rsub__', '__rtruediv__',
'__rxor__', '__setattr__', '__sizeof__', '__str__',
'__sub__', '__subclasshook__', '__truediv__',
'__trunc__', '__xor__', 'bit_length', 'conjugate',
'denominator', 'from_bytes', 'imag', 'numerator',
'real', 'to_bytes']
```

```
>>> x = 10
>>> y = int(20)
>>> z = x.__add__(y)   # z = x + y
>>> print(z)
30
>>> p,q,r = 100,100,200
>>> p.__eq__(q)    # p == q
True
>>> p.__gt__(r)      # p > q
False
```

3. 연산자 중복

서강대학교 SOGANG UNIVERSITY

- 수치 연산자 메소드

메소드	연산자
__add__(self, other)	+
__sub__(self, other)	-
__mul__(self, other)	*
__truediv__(self, other)	/
__floordiv__(self, other)	//
__mod__(self, other)	%
__pow__(self, other[, modulo])	**
__gt__(self, other)	>
__ge__(self, other)	>=
__lt__(self, other)	<
__le__(self, other)	<=
__eq__(self, other)	==
__ne__(self, other)	!=

3. 연산자 중복

서강대학교 SOGANG UNIVERSITY

◆ 연산자 중복 예제 : 좌표 클래스 만들기

```
class Point:
    def __init__(self, x=0, y=0):
        self.x = x
        self.y = y

    def get(self):
        return "(" + str(self.x) + "," + str(self.y) + ")"

    def __add__(self, other):
        newX = self.x + other.x
        newY = self.y + other.y
        return Point(newX, newY)
```

```
p1 = Point(2,3)
p2 = Point(4,7)
p3 = p1 + p2
print(p1.get())
print(p2.get())
print(p3.get())
```

```
(2,3)
(4,7)
(6,10)
```

4. 클래스와 모듈

서강대학교 SOGANG UNIVERSITY

- 클래스가 저장된 파일을 모듈로 사용할 수 있다.

```
""" 나의 애완동물 클래스 """

class Dog :
    """ 강아지 클래스 """
    def __init__(self, name, age):
        self.name = name
        self.age = age

    def bark(self):
        print(self.name, 'is barking')

class Cat :
    """ 고양이 클래스 """
    def __init__(self, name, color):
        self.name = name
        self.color = color

    def show_color(self):
        print(self.name, 'is', self.color)
```

pet.py

```
from pet import Dog, Cat

a = Dog('Jack', 3)
b = Dog('Daisy', 2)

c = Cat('Kitty', 'white')
d = Cat('Molly', 'black')

a.bark()
b.bark()

c.show_color()
d.show_color()
```

```
Jack is barking
Daisy is barking
Kitty is white
Molly is black
```