코딩, Python으로 같이 시작해요

4주차 : 클래스

클래스

<클래스와 객체>

- 객체(object)
  : 1, 2, 3등의 숫자, 리스트, 문자열 등 파이썬에서는 모든 것이 객체

```
>>> type(1)
<class 'int'>
>>> type(1.4)
<class 'float'>
>>> type("abcde")
<class 'str'>
>>> type([1,2,3])
<class 'list'>
```

- 클래스(class)
  : 모든 객체는 일종의 클래스에 속합니다.
  : 숫자 1,2 등은 int 클래스의 인스턴스
  : [1,2,3]은 list 클래스의 인스턴스
  : 클래스를 이용하여 프로그래머가 원하는 새로운 자료형을 만들 수 있다.

- 인스턴스(instance)
  : 인스턴스는 객체와 동일. 보통 객체만 지칭할 때는 객체라는 표현을 사용하고 클래스와 연관지어서 이야기 할 때는 인스턴스라는 표현을 사용합니다.
  ex) Person 클래스의 인스턴스 a를 생성하다. <-> 객체 a를 생성했다.

클래스

클래스를 직접 정의하면 우리가 원하는
새로운 자료형을 만들 수 있구나...!

클래스는 어떻게 만들면 되는데???

클래스의 구조

<클래스의 구조 : 속성과 메서드>

- 속성(attribute)
  : 해당 클래스의 객체들이 갖는 데이터 값들(=변수들)
  : 클래스 안에 있는 변수

- 메서드(method)
  : 해당 클래스의 객체들이 갖는 기능들(=함수들)
  : 클래스 안에 잇는 함수

이처럼, 클래스를 이용하면 데이터(속성)과 데이터를 조작하는 함수(메서드)를 하나의 묶음으로 관리할 수 있습니다.

클래스 : 메서드

사람을 표현하는 자료형을 만들고 싶어서 Person 클래스를 만들기로 했습니다.
Person 클래스에게 밥을 먹는 기능을 만들기 위해 eat라는 메서드를 만들겠습니다. (self는 뭘까요?)

```
class Person:
    def eat(self):
        print("밥을 먹습니다.")
```

인스턴스_이름 = 클래스_이름()
->클래스의 인스턴스를 생성합니다.

인스턴스_이름.메서드()
-> 인스턴스가 가지는 메서드를 사용합니다.

```
>>> kim = Person()
>>> kim.eat()
밥을 먹습니다.
```

```
>>> yun = Person()
>>> yun.eat()
밥을 먹습니다.
```

->하나의 Person 클래스로 수 많은 인스턴스를 계속 만들 수 있습니다.
(클래스=붕어빵 틀, 인스턴스=붕어빵들)

클래스 : self

<self>
- 메서드의 첫 번째 매개변수는 항상 self이어야 합니다. (정적 메서드, 클래스 메서드와 같은 예외적 메서드 제외)
- self는 메서드를 호출하고 있는 인스턴스 자체를 의미합니다.
- 파이썬에서는 인스턴스가 메서드를 호출하면 자동으로 첫 번째 매개변수인 self에게 인스턴스를 인자로 전달해줍니다.

```
class Person:
    def eat(self):
        print("밥을 먹습니다.")
```

```
>>> kim = Person()
>>> kim.eat()
밥을 먹습니다.
```

클래스 : 속성, 생성자

Person 클래스가 이름을 저장하는 속성 name과 체력을 저장하는 속성 energy 을 갖게 만들고 싶습니다.

<생성자가 무엇인지 부터 이해하자!>

- 클래스를 이용해 인스턴스를 만들면 __init__이라는 이름의 메서드가 자동으로 호출됩니다.
- 생성자라고 부르는 이 메서드는 저희가 따로 정의하지 않아도 모든 인스턴스가 자동으로 가지고 있으며, 인스턴스를 생성할 때 딱 한 번 자동으로 호출됩니다.

```
class Person:
    def __init__(self):
        print("생성자가 자동으로 호출되어 인스턴스가 생성되었습니다.")
    def eat(self):
        print("밥을 먹습니다.")
```

```
>>> a = Person()
생성자가 자동으로 호출되어 인스턴스가 생성되었습니다.
```

클래스 : 속성, 생성자

<생성자 안에서 속성 만들기>

- 인스턴스 a가 만들어질 때 호출되는 생성자는 인자로 self, name, energy를 요구
- name과 energy에 “다영”, 78을 인자로 전달.
- self.name과 self.energy가 인스턴스의 속성! (인스턴스 변수라고도 부름.)
- Person 클래스의 인스턴스들은 인스턴스 변수(속성) name, energy와 메서드 eat를 갖는다!
- 인스턴스_이름.속성_이름 -> 이 형식으로 속성/인스턴스 변수에 접근.

```
class Person:
    def __init__(self, name, energy):
        self.name = name
        self.energy = energy

    def eat(self):
        print("밥을 먹습니다.")
```

```
>>> a = Person("다영", 78)
>>> a.__dict__
{'name': '다영', 'energy': 78}
>>> a.name
'다영'
>>> a.energy
78
>>> a.eat()
밥을 먹습니다.
```

클래스 : __dict__

<__dict__>

- 인스턴스와 클래스는 __dict__이라는 속성을 갖습니다.(저희가 따로 만들어주지 않아도요!)
- 클래스가 어떤 메서드를 갖는지, 인스턴스가 어떤 속성을 갖고 있는지를 확인할 수 있습니다.

```
>>> potter = Person("Harry", 88)
>>> house_elf = Person("Doby", 79)
>>> potter.__dict__
{'name': 'Harry', 'energy': 88}
>>> house_elf.__dict__
{'name': 'Doby', 'energy': 79}

>>> Person.__dict__
mappingproxy({'__module__': '__main__', '__init__': <function Person.__init__ at
 0x03178BB0>, 'eat': <function Person.eat at 0x03178BF8>, '__dict__': <attribute
 '__dict__' of 'Person' objects>, '__weakref__': <attribute '__weakref__' of 'Pe
rson' objects>, '__doc__': None})
```

클래스 : 메서드와 속성의 관계

메서드는 인스턴스의 속성을 조작할 수 있기 때문에 유요한건데요.
따라서 아까 저희가 만든 eat 메서드는 아무 쓸모가 없는 메서드입니다.
밥을 먹으면 인스턴스의 energy 속성이 10 만큼 증가하게 만들어서 쓸모 있는 메서드로 발전 시켜 봅시다.

```
class Person:
    def __init__(self, name, energy):
        self.name = name
        self.energy = energy

    def eat(self):
        print("밥을 먹어 기운이 납니다.")
        self.energy += 10
        print("{0}님의 현재 에너지 : {1}".format(self.name, self.energy))
```

```
>>> kim =Person("Doby", 45)
>>> kim.eat()
밥을 먹어 기운이 납니다.
Doby님의 현재 에너지 : 55
```

클래스 실습

- 은행 계좌를 만드는 Account 클래스를 만들어 봅시다.
- 속성 : 계좌주 이름을 저장하는 name, 잔액을 저장하는 money
- 메서드 :돈을 입금하는 deposit 메서드.

```
class Account:
    def __init__(self, name, money):
        self.name = name
        self.money = money
```

```
>>> myAccount = Account("윤다영", 34000)
>>> myAccount.__dict__
{'name': '윤다영', 'money': 40000}
>>> myAccount.name
'윤다영'
>>> myAccount.money
40000
```

클래스 실습

```
class Account:
    def __init__(self, name, money):
        self.name = name
        self.money = money

    def deposit(self, amount):
        self.money += amount
        print("-----{0}님의 계좌-----".format(self.name))
        print("{0}원을 입급했습니다.".format(amount))
        print("잔액 : {0}".format(self.money))
```

```
>>> myAccount.deposit(6000)
-----윤다영님의 계좌-----
6000원을 입급했습니다.
잔액 : 40000
```

과제 1

- week4-1.py 로 저장
- 아래의 Person 클래스에 age 속성을 추가하세요.(속성은 생성자 안에서 만들어야 하겠지요?)
- Person 클래스에 자기소개를 하는 greeting 메서드를 추가하세요.
  (자기 이름과 나이를 말해야 합니다.)
- 다음과 같이 인스턴스를 만들었을 때 속성과 메서드를 정상적으로 사용할 수 있으면 됩니다.

```
>>> a = Person("yun", 78, 22)
>>> a.__dict__
{'name': 'yun', 'energy': 78, 'age': 22}
>>> a.eat()
밥을 먹어 기운이 납니다.
yun님의 현재 에너지 : 88
>>> a.greeting()
안녕하세요, 저는 yun입니다. 22살 입니다.
```

```
class Person:
    def __init__(self, name, energy):
        self.name = name
        self.energy = energy

    def eat(self):
        print("밥을 먹어 기운이 납니다.")
        self.energy += 10
        print("{0}님의 현재 에너지 : {1}".format(self.name, self.energy))
```

과제 2

- week4-2.py 로 저장
- (실습에서 만들었던)Account 클래스에 돈을 출금하는 withdraw 메서드를 추가하세요.
- 출금하려는 액수가 남은 잔액(money 속성 -> self.money)보다 크면 “잔액이 부족합니다.”라고 문구를 띄우고 잔액을 알려주세요.
- 잔액이 충분하면 self.money에서 해당 액수를 차감하고 남은 잔액을 알려주세요.

```
>>> a = Account("윤다영", 3000)
>>> a.withdraw(10000)
-----윤다영님의 계좌-----
잔액이 부족합니다.
잔액 : 3000
>>> a.deposit(8000)
-----윤다영님의 계좌-----
8000원을 입급했습니다.
잔액 : 11000
>>> a.withdraw(10000)
-----윤다영님의 계좌-----
10000를 출금했습니다.
잔액 : 1000
```

과제 제출 안내

과제 제출은 다음 주 월요일 자정 까지 입니다!

제출 이메일 : ektmf7890@hanyang.ac.kr