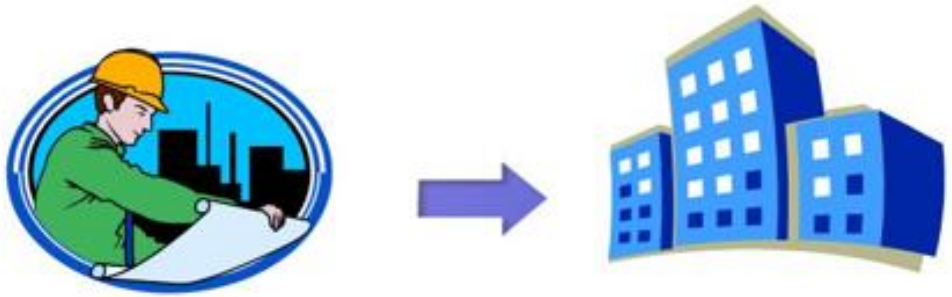
UNISTUDY

파이썬

20강. 클래스와 객체

1. 클래스와 객체

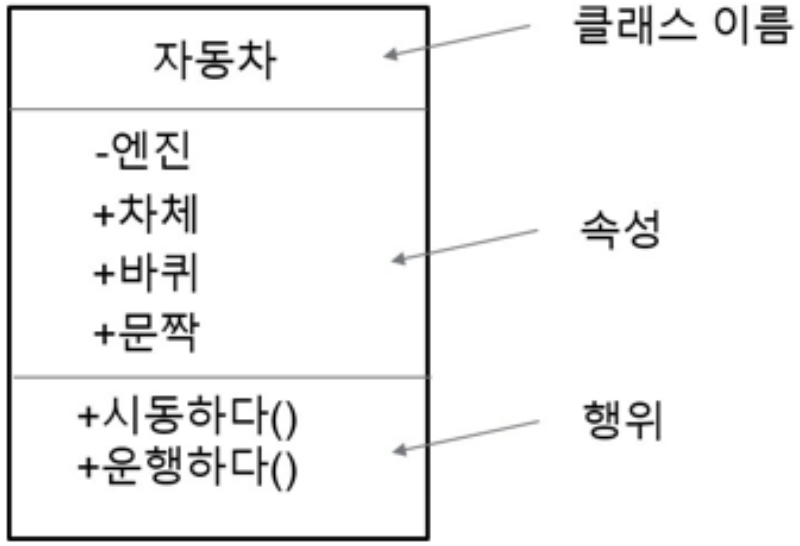
- 클래스와 객체는 객체지향프로그래밍에서 나오는 용어이다. 클래스(Class)는 프로그램을 이용해서 객체를 만들어주는 역할을 하고, 객체(Object)는 클래스에 의해서 만들어지는 결과물(instance)을 말 한다.
- 클래스를 흔히 설계도라고 비유한다.
- 일반적으로 하나의 설계도를 가지고 수많은 건물을 건축할 수도 있고, 자동차를 생산해 낼 수도 있기 때문이다.
- 클래스와 객체의 관계를 설계도와 건물로 비유한 예이다.



클래스(설계도)　　객체(건물)

2. 클래스와 객체의 관계

- 클래스는 속성(Attribute)과 행위(Action)로 구성된다. 다음 [그림] 6-2는 자동차 클래스의 구성요소를 UML(Unified Modeling Language)16) 표기법으로 도식화한 클래스 다이어그램이(Diagram)이다.

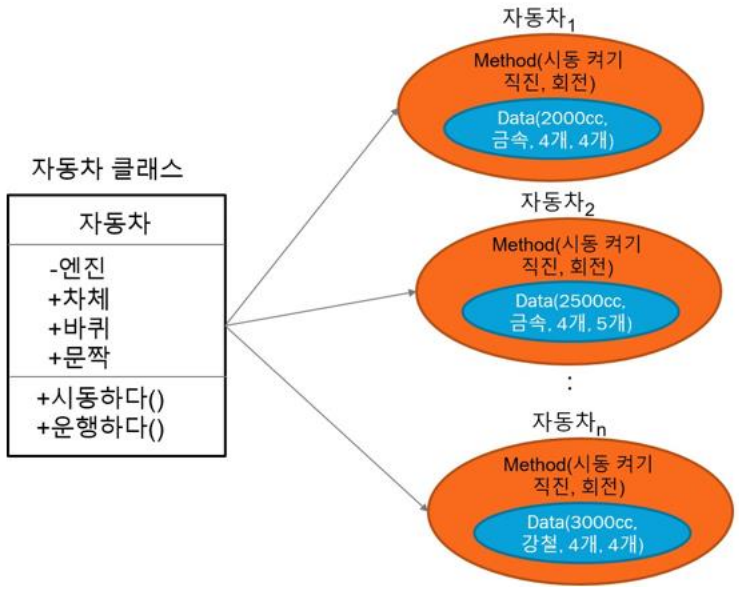


2. 클래스와 객체의 관계

- 클래스의 속성은 자료(data)를 나타내며, 명사 형태로 기술한다. 속성 앞부분의 (-, +)는 외부에서 접 근여부를 나타내는 기호이다.
- - 기호는 외부에서 접근을 차단하여 해당 자료를 은닉(private)화 시키고, +는 외부에서 접근을 허용하도록 공용(pubic)화 시킨다.
- 또한 클래스의 행위는 자료를 연산하거나 조작하는 역할을 하며, 동사 형태로 기술하고, 프로그램에서는 기능을 정의하는 함수로 표현한다.

3. 객체

- 객체는 클래스에 의해서 만들어지는 결과물이다. 클래스의 속성에 실제 자료가 들어가고, 실제 자료를 동작시키는 함수가 하나로 묶여진 단위를 말한다.
- 즉 자료(Data)와 메서드(Method)가 하나로 묶여져 서 만들어진 결과물이다. 클래스와 객체 간의 관계를 나타내고 있다.



3. 객체

- 클래스와 객체는 1 : N의 관계를 갖는다.
- 잘 설계된 자동차 클래스 한 개를 이용하 여 서로 다른 엔진, 차체, 바퀴, 문짝을 가지고 있는 N개의 자동차를 생산한다.
- 이렇게 생산된 N의 자 동차 객체는 4개의 속성에 의해서 만들어진 Data와 2개의 행위에 의해서 만들어진 Method를 하나의 묶음으로 가지고 있다.
- 따라서 각 객체는 Method를 이용하여 자체 Data를 조작하면서 실행된다. 객 체들은 서로 독립적인 메모리 주소를 가지고 있으며, 메시지(Message)를 이용해서 상호작용도 가능하다.

3. 객체

- 함수(Function) vs 메서드(Method)
- 함수(Function) : 독립된 기능을 수행하는 코드(명령문)들의 집합
- 메서드(Method) : 클래스에 포함되어 있는 함수

4. 클래스 구성

- 클래스는 변수와 함수들을 하나로 묶어놓은 집합체이다. 기존 C언어의 구조체와 비슷한 역할을 하는 일종의 자료구조이다.
- 물론 단순한 자료구조의 개념을 넘어서 객체지향의 주요 특징을 구현할 수 있는 메커니즘(Mechanism)을 제공한다.

5. 함수와 클래스

- 함수는 독립적인 기능을 수행할 수 있는 코드의 집합이고, 클래스는 공통 기능을 갖는 함수들을 하나로 묶을 수 있는 단위를 의미한다. 다음은 함수와 클래스를 선언하는 형식이다. 두 형식을 비교하면서 그 차이점이다.

```
함수 형식)
    def 외부함수(인수) :
        변수선언
        def 내부함수(인수) :
            명령문
            return 값
        return 내부함수
```

```
클래스 형식)
    class 클래스명 :
        변수 선언
        def 생성자(인수) :
            명령문
        def 함수명(인수) :
            명령문
```

5. 함수와 클래스

- 왼쪽의 함수 형식에서 def 명령어로 함수명과 인수를 이용하여 선언한다. 외부함수에서는 함수 내에 서 사용할 변수를 선언하고, 내부함수에서는 선언된 변수를 처리하는 명령문을 작성한다. 마지막으로 내부함수를 호출할 수 있도록 return문으로 반환한다.
- 오른쪽의 클래스 형식은 함수 형식과 비슷한 모양을 갖는다. class 명령어 다음에 클래스이름이 온다.
- 함수처럼 외부에서 값을 넘겨받는 인수는 없다.
- 클래스 내부는 객체의 자료가 저장될 변수를 선언 하고, 자료를 처리하는 함수들로 구성된다. 그리고 객체를 생성하기 위해서 '생성자'라는 특수한 함수 를 포함한다. 클래스는 객체를 통해서 함수를 호출하기 때문에 함수들을 반환하는 return문이 없다.

5. 함수와 클래스

chapter06.lecture.step01_class.py

```
# (1) 함수 정의
def calc_func(a, b): # 외부함수
    # 변수 선언 : 자료 저장
    x = a # 10
    y = b # 20

    def plus():  # 내부함수
        p = x + y
        return p

    def minus(): # 내부함수
        m = x - y
        return m
    return plus, minus

# (2) 함수 호출
p, m = calc_func(10, 20)
print('plus =', p())
print('minus=', m())
```

Python Console

```
plus = 30
minus= -10
```

5. 함수와 클래스

```
# (3) 클래스 정의
class calc class :
    # 클래스 변수 : 자료저장
    x = y = 0

    # 생성자 : 객체 생성 + 멤버변수 초기화
    def_init_(self, a, b):
        self.x = a # 10
        self.y = b # 20

    # 클래스 함수
    def plus(self):
        p = self.x + self.y # 변수 연산
        return p

    # 클래스 함수
    def minus(self):
        m = self.x - self.y # 변수 연산
        return m

# (4) 객체 생성
obj = calc class(10, 20)

# (5) 멤버 호출
print('plus = ', obj.plus()) # plus = 30
print('minus =', ob.minus()) # minus = -10
```

```
plus =  30
minus = -10
```

5. 함수와 클래스

- 해설 함수와 클래스 예
- # (1) 함수 정의
- 매개변수 x, y를 대상으로 덧셈과 뺄셈을 수행하는 plus()와 minus()의 내부함수를 포함하는 중첩함 수이다.
- # (2) 함수 호출
- 실인수 10과 20으로 중첩함수를 호출하고, 함수 클로저(내부함수)를 반환받아서 덧셈과 뺄셈 연산 결과를 출력한다.
- # (3) 클래스 정의
- 함수의 매개변수 x, y를 클래스 변수로 선언한다. 생성자를 선언하여 객체를 생성하고, 클래스 변수에 값을 초기화하도록 한다. 덧셈과 뺄셈을 수행하는 plus()와 minus()를 클래스 함수로 정의한다.
- # (4) 객체 생성
- 클래스의 생성자를 이용하여 객체를 생성하고 클래스 변수에 실인수 10과 20을 할당하여 자료를 만든다.
- # (5) 멤버 호출
- 생성된 객체(obj)를 이용하여 클래스에서 정의한 함수를 호출하고, 연산결과를 출력한다.

6. 클래스 구성요소

- 파이썬에서 제공하는 클래스는 일반 변수나 함수와 구분하기 위해서 클래스 선언부(header)에서 선언한 변수는 멤버변수, 클래스 본체(body)에서 정의하는 함수는 메서드 (Method)라고 한다. 멤버변수와 메서드를 간편하게 클래스의 멤버(Member)라고 부른다.
- 그리고 함 수 형태로 선언하지만 ‘ init ’이라는 별도의 이름을 갖는 생성자(Constructor)로 구성된다. 따 라서 클래스는 다음과 같이 멤버와 생성자로 구성되고, 멤버는 자료를 저장하는 멤버변수와 처리 기능 을 갖는 메서드로 구성된다.

- 클래스 구성요소 : 멤버(Member) + 생성자(Constructor)
- 멤버 : 멤버변수(자료) + 메서드(기능)

6. 클래스 구성요소

- 결국 클래스는 객체를 생성하는 역할이기 때문에 객체 내에서 공유할 수 있는 자료와 이러한 자료를 처리하는 함수들을 묶어서 객체가 만들어질 수 있도록 정의해 놓은 것을 말한다.

```
class 클래스명 :
    멤버변수                          } header
    …
    def __init__(self) :              } body
        …
    def 메서드명(self) :
        …
```

6. 클래스 구성요소

chapter06.lecture.step01_class.py

```
class Car:
    # (1) 멤버변수
    cc = 0 # 엔진 cc
    door = 0 # 문짝 개수
    carType = None # null


    # (2) 생성자
    def __init__(self, cc, door, carType):
        # 멤버 변수 초기화
        self.cc = cc
        self.door = door
        self.carType = carType # 승용차, SUV


    # (3) 메서드
    def display(self):
        print("자동차는 %d cc이고, 문짝은 %d개, 타입은 %s"
              % (self.cc, self.door, self.carType))



# (4) 객체 생성
car1 = Car(2000, 4, "승용차") # 객체 생성 + 초기화
car2 = Car(3000, 5, "SUV")

# (5) 멤버 호출 : object.member()
car1.display() # car1 멤버 호출
car2.display()  # car2 멤버 호출
```

Python Console

```
자동차는 2000 cc이고, 문짝은 4개,
타입은 승용차
자동차는 3000 cc이고, 문짝은 5개,
타입은 SUV
```

6. 클래스 구성요소

- 해설 클래스 구성요소 예
- # (1) 멤버변수
- Car 클래스의 선언부에 3개의 멤버변수(cc, door, carType)를 선언한다. (초기값은 0과 null이 다.)
- # (2) 생성자
- 생성자는 함수의 선언과 유사하지만 차이점은 이름이 정해져 있고 self 라는 인수를 사용한다는 점이 다. 예문에서 생성자는 def 명령어 다음에 ' init 'this 름과 self 인수를 이용하여 선언된 다. self 인수는 생성자 내에서 클래스의 멤버를 호출하는데 이용된다. self 인수 다음에 오는 3개의 매개변수는 3개의 멤버변수(cc, door, carType)에 값을 초기화하기 위한 인수이다. 매개변수 는 객체가 생성되는 시점에서 외부의 실인수를 받아서 멤버변수에 초기화하는 역할을 한다.
- # (3) 메서드
- 메서드는 함수의 선언과 동일하다. 차이점은 생성자와 마찬가지로 self 라는 정해진 인수를 포함하고 있다는 점이다. 예문에서 display() 메서드는 self를 이용하여 3개의 멤버변수의 내용을 출력하는 역할을 한다.

6. 클래스 구성요소

- # (4) 객체 생성
- 클래스의 생성자를 이용하여 객체를 생성한다. 한 개의 클래스에 의해서 다수의 객체를 만들어 낼 수가 있다. Car 클래스의 생성자를 이용하여 객체를 생성하는 방법은 다음과 같다.
- 참조변수 = Car(실인수)
- Car()는 Car 클래스의 생성자이고, 실인수는 생성자의 매개변수에 전달할 실제값이다. 따라서 Car(2000, 4, "승용차") 형식의 명령문은 3개의 멤버변수에 각각 cc=2000, door=4, carType="승 용차"가 할당되어 car1 객체가 생성된다.
- # (5) 멤버 호출
- 생성된 객체의 참조변수를 이용하여 객체의 멤버를 호출할 수 있다. 참조변수를 이용하여 객체의 멤버를 호출하는 방법은 다음과 같다.
- 참조변수.멤버(멤버변수 or 메서드)
- car1.display()는 car1 참조변수를 이용하여 display() 메서드를 호출하는 명령문이다. 따라서 car1의 객체에 포함된 메서드 호출에 의해서 car1 객체의 자동차 정보가 출력된다. 만약 car2 객체의 정보를 출력하기 위해서는 car2.display() 명령문을 이용한다.

THANK YOU