2021.02.26

[객체지향 프로그래밍]

1. 객체와 클래스 생성

```
class 클래스명:                  # 클래스 생성
    함수나 변수들 선언
객체의변수명 = 클래스명( )     # 객체 생성
```

2. 변수와 메소드

(1) 클래스에서 변수와 메소드의 역할

1) 변수: 객체를 이용하여 데이터를 저장한다.
2) 메소드: 객체가 고유한 기능을 갖도록 한다.

(2) 메소드의 유형

1) 인스턴스(객체) 메소드: 첫 매개변수는 **self**여야 하며, 객체의 변수 혹은 메소드에 접근하기 위한 메소드
2) 클래스 메소드: 첫 매개변수에 cls(clazz)를 쓰며, @classmethod를 통해 선언한다.
3) 정적(static) 메소드: 매개변수가 따로 없으며, @staticmethod를 통해 선언한다.

인스턴스(객체) 메소드	클래스 메소드	static 메소드

class Person:
  name = "홍길동"
  gender = "남자"
 	
  def print_info(self):
  print("{}는 {}입니다."
  .format(self.name,
  self.gender))
	class Person:
	  @classmethod
	  def do_(cls):
	  cls.name = "신사임당"
	  cls.gender = "여자"
	  print("{}는 {}입니다"
	  .format(cls.name,
	  cls.gender))
		class Person:
		  @staticmethod
		  def that_():
		
		  print("{}는 {}입니다"
		  .format(Person.name,
		  Person.gender))

3. 생성자(__init__())와 소멸자(__del__())

(1) 생성자함수: **__init__(self,** *매개변수1(option)...***)** / 객체 생성시 자동실행되며, 생성시 필요한 코드를 포함
(2) 소멸자함수: **__del(self)__** / 객체 소멸시 자동 실행되며, 소멸시 필요한 코드를 포함
/ 인스턴스 객체의 레퍼런스 카운트가 0이될 때 소멸한다.

4. 상속과 재정의

(1) 상속: 객체 사용의 한 방법으로, 부모 클래스의 모든 속성을 자식 클래스에게 물려줄 수 있다.
(2) 재정의: 부모 클래스에서 정의한 함수를 자식 클래스에서 다시 정의하는 것
※ **super( )** : 부모 클래스의 멤버를 참조한다.

ex – __str__메서드에서 super( ).__str__( )을 통해 부모 클래스 Person의 __str__메소드 호출

```
class Student(Person):
    def __str__(self):
        return super().__str__() + " 전공:{}".format(self.major)
```