19.Python Class

1. Class

1] **Class** 정의

- 파이썬 클래스는 새로운 이름 공간을 지원하는 또 다른 단위
- 파이썬에서는 동적으로 인스턴스 외부에서 인스턴스 멤버를 추가할 수 있음
- 파이썬에서는 클래스와 독립적으로 각 인스턴스를 하나의 이름 공간으로 취급함

2] Class 구조

class 클래스 이름: #헤더(Header)

pass #몸체(Body)

① 인스턴스: 클래스로 부터 만들어낸 객체

② 모듈 vs. 클래스 vs. 인스턴스

- 모듈: 파일 단위로 이름 공간을 구성
- 클래스: 클래스 영역 내에 이름 공간을 구성
- 인스턴스: 인스턴스 영역 내에 이름 공간을 구성
- 클래스가 공장, 붕어빵 틀이면, 인스턴스는 공장물품, 붕어빵

예시 1

```
class S1:
    a = 1
print ("S1.a->", S1.a)
print

S1.b = 2        # 클래스 이름 공간에 새로운 이름의 생성
print ("S1.b->", S1.b)
```

결과

```
------------------------------------------------------------
S1.a-> 1
S1.b-> 2
```

2. 일반/정적/class method

1] **일반 메소드의 정의와 호출**

- 클래스 내부에 메소드 선언 - **def** 키워드 사용
- 일반 함수와 다른 점은 첫번째 인수로 self 사용 (self라는 이름은 관례적)
  - ① self: 인스턴스 객체 자신의 레퍼런스를 지니고 있음
  - ② 각 인스턴스들은 **self**를 이용하여 자신의 이름 공간에 접근
  - ③ 클래스 안 메소드 정의 시 def 키워드 활용
  - ④ 클래스 안에 존재하는 함수들 → 클래스의 메소드
  - ⑤ 클래스의 메소드는 첫 번째 인자에 self 넣음 → 인스턴스에 불려짐
  - ⑥ 첫 번째 인자에 self 들어간 것 → 인스턴스 메소드

2] 정적 메소드

- ① 메소드: 인스턴스 객체와 무관하게 클래스 이름 공간에 존재하는 메소드
- ② 클래스 이름을 이용하여 직접 호출할 수 있는 메소드
- ③ 해당 클래스의 인스턴스를 통해서도 호출 가능
- ④ 장식자(Decorator) @staticmethod 활용

3] class 메소드

① 인스턴스 객체와 무관하게 클래스 이름 공간에 존재하는 메소드
② 클래스 이름을 이용하여 호출하며 첫 인수로 클래스 객체를 자동으로 받는 메소드
③ 해당 클래스의 인스턴스를 통해서도 호출 가능
④ 장식자(Decorator) @classmethod 활용

3. 일반/정적/class method 예시

예시 1

```
class D:
    @staticmethod
    def spam(x, y): # self가 없다.
        print ('static method', x, y)

D.spam(1,2) # 인스턴스 객체 없이 클래스에서 직접 호출
print
d = D()
d.spam(1,2) # 인스턴스 객체를 통해서도 호출 가능.
```

결과

```
------------------------------------------------------------
static method 1 2
static method 1 2
```

예시 2

```
class Var:
    c_mem = 100 # 클래스 멤버 정의
    def f(self):
        self.i_mem = 200 # 인스턴스 멤버 정의
    def g(self):
        print ("self.i_mem->", self.i_mem)
        print ("self.c_mem->", self.c_mem)

 print ("Var.c_mem->", Var.c_mem)  # 클래스 객체를 통하여 클래스 멤버 접근
v1 = Var() # 인스턴스 v1 생성
print ("v1.c_mem->", v1.c_mem)   # 인스턴스를 통하여 클래스 멤버 접근
v1.f()                           # 인스턴스 멤버 i_mem이 생성됨
print ("v1.i_mem->", v1.i_mem )  # 인스턴스 v1을 통하여 인스턴스 멤버 접근
```

결과

```
------------------------------------------------------------
Var.c_mem-> 100
v1.c_mem-> 100
v1.i_mem-> 200
```

1. 생성자와소멸자

1] 생성자 메소드

① __init__: 생성자 메소드

② 객체가 생성될 때 자동으로 불리어지는 메소드

③ self 인자가 정의되어야 함

2] 소멸자 메소드

① 객체가 소멸 (메모리에서 해제)될 때 자동으로 불리어지는 메소드(self 인자가 정의되어야 함_

② 개발자가 특별히 작성하지 않아도 될 메소드(파이썬에서는 메모리나 기타 자원들의 해제가 자동수행)

③ 소멸되기 직전에 del이라는 함수가 수행

3] 생성자 / 소멸자 **메소드**

① __str__: print 예약어나 str() 내장함수 호출에 대응되는 메소드

예시 1

```
from time import ctime, sleep
class Life:
    def __init__(self): # 생성자
        self.birth = ctime() # 현재시간에 대한 문자열을 얻는다.
        print ('Constructor->', self.birth) # 현재 시간 출력
    def __del__(self): # 소멸자
        print ('Destroy->', ctime() ) # 소멸 시간 출력

def test():
    mylife = Life()
    print ('Sleeping for 3 sec')
    sleep(3) #3초간 sleep(block)상태에 있음 (CPU 점유 못함)

test()
```

결과

```
----------------------------------------------------------
Constructor-> Mon Oct 29 17:08:18 2018
Sleeping for 3 sec
Destroy-> Mon Oct 29 17:08:21 2018
```