

✗ 객체 지향 프로그래밍.

- ~~프로그래밍~~ 프로그램을 실제 시상이 가깝게 모델링하는 기법.

=> 실제 세상에서 일어나는 것처럼 프로그래밍 하는 것.

· 클래스는 속성(클래스 변수 or 인스턴스 변수)과 행위(메소드)를 가지도록 설계됨.

· 특정 클래스를 가진 객체는 어떤 ^①속성(state)을 가지게 되고, 클래스에서 정의한 ^②행위(behavior)를 수행한다.

↳ 메소드

↳ 객체가 어떤 행위를 수행함!

→ ✗ 객체가 어떤 행동을 수행하도록 만들려면, 먼저 해당 객체를 변수에 저장한 후 메소드를 사용해야 한다.

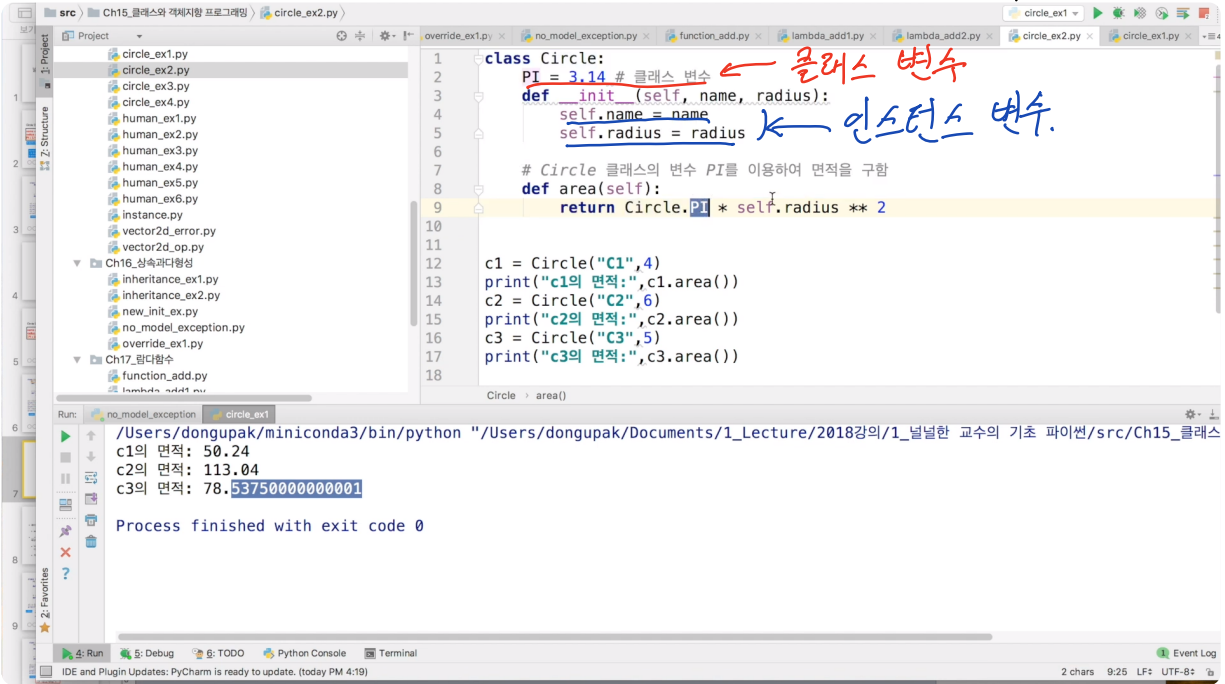
✗ 모든 객체는 특정 class type을 가지고있다!!!

· 클래스 = 객체의 설계도.

· 인스턴스 = 클래스로부터 만들어지는 각각의 객체

※ 클래스 변수와 인스턴스 변수의 차이점:

- 클래스 변수는 모든 인스턴스들이 공유하고 있는 값이 포함되어 있다.



```
1 class Circle:
2     PI = 3.14 # 클래스 변수
3     def __init__(self, name, radius):
4         self.name = name
5         self.radius = radius
6
7     # Circle 클래스의 변수 PI를 이용하여 면적을 구함
8     def area(self):
9         return Circle.PI * self.radius ** 2
10
11
12 c1 = Circle("C1", 4)
13 print("c1의 면적:", c1.area())
14 c2 = Circle("C2", 6)
15 print("c2의 면적:", c2.area())
16 c3 = Circle("C3", 5)
17 print("c3의 면적:", c3.area())
18
```

Run: /Users/dongupak/miniconda3/bin/python "/Users/dongupak/Documents/1_Lecture/2018강의/1_날날한 교수의 기초 파이썬/src/Ch15_클래스
c1의 면적: 50.24
c2의 면적: 113.04
c3의 면적: 78.53750000000001
Process finished with exit code 0

7.1. 클래스(class)와 인스턴스

'김연아'는 실제로 존재하죠? 네, 여러분이 생각하시는 그 김연아 맞아요. ㅎㅎ '김동성'도 실제로 존재하죠? 두 사람 다 실제로 존재하는 사람입니다.

두 사람의 공통점은 무엇일까요? 여러 가지를 들 수 있겠지만 둘 다 '스케이터'라는 공통점을 갖고 있지요.

'스케이터'라는 단 하나의 사람이나 물건이 실제로 존재할까요? 그렇지 않습니다. 하지만 우리는 '스케이트 타는 사람'을 '스케이터'라고 말합니다. 이런 것을 일컫는 말이 클래스(class)입니다. 우리 말로 옮기기는 쉽지 않지만 '부류'라는 의미로 생각하시면 좋을 것 같아요.

다른 예를 들어볼까요?

'사과'는 클래스이고요, '내가 잊어버려 먹은 사과 다섯 개' 중에 두 번째 것'이라고 꼭 적어서 말해주면 실체(instance)로 봐줄만합니다.

'좋은 집'은 실체일까요? 어느 한 집만을 꼭 적어서 '좋은 집'이라고 하기는 힘들 것 같은요. 그럼 '우리 집'은 실체일까요? 그건 실체라고 해도 될 것 같네요. 단, 집을 여러 채 가진 사람이 '우리 집'이라고 말할 때는 정확히 어느 집을 가리키는 것인지 알 수 없겠죠. 프로그램 작성을 위해 클래스를 설계하다보면 이런 애매한 문제를 만날 때도 있지요.

7.2. 변수와 메서드

지난 시간에 클래스와 객체가 무엇인지 잠깐 살펴보았죠? 실제 세계에 존재하는 실체(instance)를 객체(object)라고 하고, 객체들의 공통 점을 간주해서 개념적으로 나타낸 것이 클래스(class)라고 했습니다.

어떤 클래스를 만들려면 ^① 객체가 갖는 성질과 ^② 객체가 하는 행동을 정의해주면 된다고도 했구요. 디아블로2 게임의 아마존이라는 캐릭터를 클래스로 표현해 볼까요?

```
class Amazon:
    strength = 20
    dexterity = 25
    vitality = 20
    energy = 15

    def attack(self):
        return 'Jab!!!'
```

변수

메서드

※ 동일한 class type을 가지고 있는 객체들이 공통된 특징을 가진다면,
공통된 특징의 데이터를 '클래스 변수'에 바인딩 해줘야함.

※ 바인딩 : '어떤 변수가 특정 데이터를 가리킨다!'

< Class와 Object >

- Computer Programming (1960년대.)
- PC의 등장 [IBM XT, 쿼드프로세서] (1980년대)
- 시간이 지남에 따라 프로그램의 양이 엄청 증가함,
↳ 응용프로그램.
- 그러고는 객체 지향이 없었다!!!, 관한 지향만이 존재했음.
⇒ 그래서 여러 사람의 분업화가 불가능했었다.,
생산성도 매우 낮았었다.

- 객체지향형 프로그래밍 : 복잡한 프로그램을 object들의 결합으로 만든다.

↳ ① 여러 모듈을 불러와서

이들을 결합하여 프로그램을 완성함.

② 여러 '함수'들을 결합하여 프로그램을 완성함.

여러 메소드들을 결합하여 특정 클래스가 구현할 수 있는 행동들을 정의함.

- 모든 객체는 '속성'과 '기능'을 가진다.

↑ 클래스 변수 ↑ 메소드.

or 인스턴스 변수

$$\underbrace{a}_{\text{객체}} = \underbrace{3}_{\text{Instance}} \rightsquigarrow \text{Class type int}$$
 Instance : 이름을 지어주지 않은 물체

· 모든 객체는 class type을 가지고 있다.

· 클래스는 type, 객체는 변수.