

## # 오브젝트

파이썬의 모든 부분은 오브젝트로 구성되어 있다.

오브젝트는 타입, 표현 방식, 상호작용 방식으로 구성

▶ 리스트의 경우에는 data와 pointer로 구성

## # OOP에 대해서

파이썬 상에 있는 모든 존재는 전부 객체이다.

데이터를 패키징해서 decomp, abstr를 구현!

분할 정복이 용이하며 코드 재사용도 용이한 구조이다.

## # 클래스를 만드는 과정

```
class Coordinate(object):
    def __init__(self, x, y):
        self.x = x
        self.y = y
```

method to an instance  
tuple  
attributes for object

▶ 여기서 class는 class의 생성을 선언

▶ Coordinate는 클래스의 이름,

▶ (object) 는 상속하고자 하는 다른 class (object는 가장 기본적인 class)

```
class Coordinate(object):
    def __init__(self, x, y):
        self.x = x
        self.y = y
    def distance(self, other):
        x_diff_sq = (self.x - other.x)**2
        y_diff_sq = (self.y - other.y)**2
        return (x_diff_sq + y_diff_sq)**0.5
```

use it to refer to any instance  
another parameter to method  
dot notation

```
c = Coordinate(3,4)
origin = Coordinate(0,0)
print(c.x)
print(origin.x)
```

use the dot to access an attribute  
instance c

```
c = Coordinate(3,4)
zero = Coordinate(0,0)
print(c.distance(zero))
```

```
c = Coordinate(3,4)
zero = Coordinate(0,0)
print(Coordinate.distance(c, zero))
```

인스턴스 간의 상호작용을 관리하는 기능은

\_\_add\_\_, \_\_sub\_\_ 등의 오버라이드로 구현한다.

- 1) \_\_add\_\_(self, other): self + other
- 2) \_\_sub\_\_(self, other): self - other
- 3) \_\_eq\_\_(self, other): self == other
- 4) \_\_lt\_\_(self, other): self < other
- 5) \_\_len\_\_(self): len(self)
- 6) \_\_str\_\_(self): print(self)