
5주차 : class

# 커리큘럼

1. 변수, 입출력
2. 조건문, 반복문
3. 리스트, 튜플, 세트, 딕셔너리
4. 함수
5. class
6. 알고리즘 입문, 그리디 알고리즘
7. 재귀함수
8. 탐색
9. DP(Dynamic Programming)

# class

✓ 같은 코드를 반복해서 만드는 것은 비효율적 → 미리 만들어 놓기

```
name_1 = '기사'
hp_1 = 50
damage_1 = 8

print('{0} 생성!'.format(name1))
print('체력 {0}'.format(hp1))
print('공격력 {0}'.format(damage1))

name_2 = '궁수'
hp_2 = 35
damage_2 = 12

print('{0} 생성!'.format(name2))
print('체력 {0}'.format(hp2))
print('공격력 {0}'.format(damage2))
```

```
class Unit:
    def __init__(self, name, hp, damage):
        self.name = name
        self.hp = hp
        self.damage = damage
        print('{0} 생성!'.format(name))
        print('체력 {0}'.format(hp))
        print('공격력 {0}'.format(damage))

Knight1 = Unit('기사', 50, 8)
Acher1 = Unit('궁수', 35, 12)
Knight2 = Unit('기사', 50, 8)
```

# 기본 구조

```
class Unit:
    def __init__(self, name, hp, damage):
        self.name = name
        self.hp = hp
        self.damage = damage
        print('{0} 생성!'.format(name))
        print('체력 {0}'.format(hp))
        print('공격력 {0}'.format(damage))

Knight1 = Unit('기사', 50, 8)
Acher1 = Unit('궁수', 35, 12)
Knight2 = Unit('기사', 50, 8)
```

- ✓ __init__ : 생성자
- ✓ Unit : 클래스(class)
- ✓ Knight1, Acher1, ... : 객체(object)
  → 인스턴스
- ✓ name, hp, ... : 멤버(member)
- ✓ **객체 지향 프로그래밍(OOP)**
  ↔ 절차 지향 프로그래밍

# 멤버 변수

```
class Unit:
    def __init__(self, name, hp, damage):
        self.name = name
        self.hp = hp
        self.damage = damage
        print('{0} 생성!'.format(name))
        print('체력 {0}'.format(hp))
        print('공격력 {0}'.format(damage))

Knight1 = Unit('기사', 50, 8)
Wizard1 = Unit('마법사', 30, 0)
print('유닛 이름'.format(Wizard1.name))
Wizard1.stun = True

if Wizard1.stun:
    print('{0}은 스턴 기술을 사용 중'.format(Wizard1.name))
```

# 메소드(method)

```
class MoveUnit:
    def __init__(self, name, hp, damage):
        self.name = name
        self.hp = hp
        self.damage = damage
    def move(self, location):
        print('{0}가 {1}로 이동'.format(self.name, location))
    def attack(self):
        print('{0}가 공격력 {1}로 공격'.format(self.name, self.damage))
    def damaged(self, damage):
        print('{0}가 {1}만큼 데미지를 받음'.format(self.name, damage))
        self.hp -= damage
        if self.hp <= 0:
            print('{0} 소멸'.format(self.name))
```

# pass

- ✓ 아무것도 안하고 일단은 넘어간다는 뜻, 함수 형태만 유지
- ✓ __init__ 이든, 일반 메소드든, 일반 함수든 다 사용가능

```
class upgrade:
    def __init__(self, supply, hp, mp)
        pass

def start():
    pass
```

# 상속(inheritance)

```
class firstclass:
    def __init__(self, a, b):
        self.a = a
        self.b = b
        ...

class secondclass:
    def __init__(self, a, b, ...):
        # firstclass 내용과 겹침
        self.a = a
        self.b = b
        ...
```

```
class firstclass:
    def __init__(self, a, b)
        self.a = a
        self.b = b
        ...

class secondclass(firstclass):
    def __init__(self, a, b, ...):
        firstclass.__init__(self, a, b)
        ...
```

# super

- ✓ 상속할 때 클래스명 대신 사용
- ✓ super는 괄호가 있는 형태이고(super( )), self를 쓰지 않는다.
- ✓ 단, 상속을 여러 개 할때는 제일 먼저 상속한 것만 가지고 온다.

```
class firstclass:
    def __init__(self, a, b)
        self.a = a
        self.b = b
        ...

class secondclass(firstclass):
    def __init__(self, a, b, ...):
        super().__init__(a, b)
        ...
```

# 메소드 오버라이팅

```
class firstclass:
    def hello(self):
        print(‘Hello’)

class secondclass(firstclass):
    def hello(self):
        print(‘Hello world’)

py = secondclass()
py.hello()
```

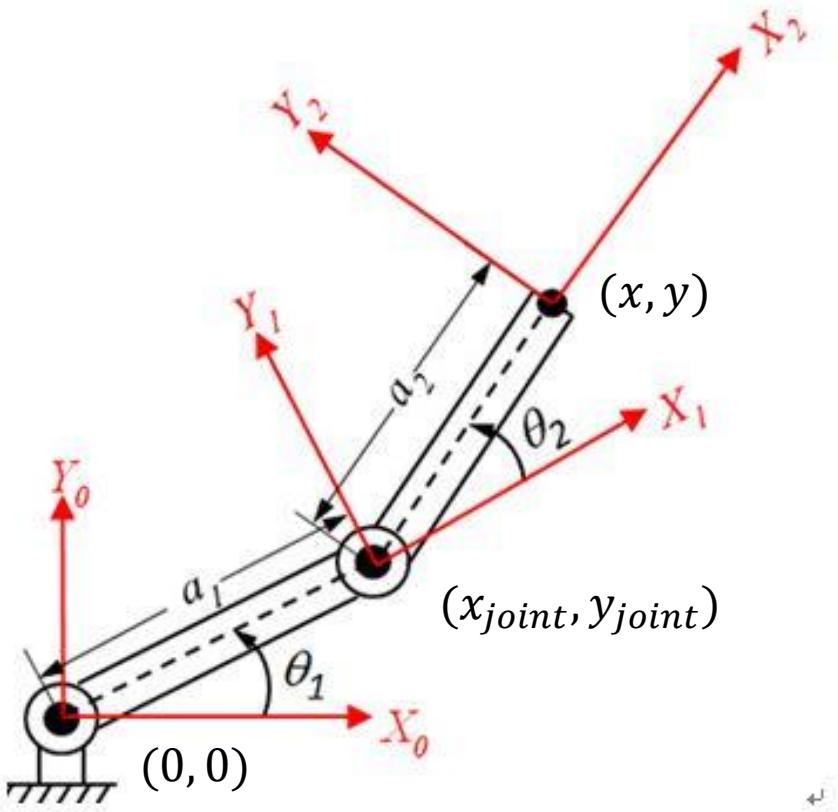
```
class firstclass:
    def hello(self):
        print(‘Hello’)

class secondclass(firstclass):
    def hello(self):
        super().hello()
        print(‘world’)

py = secondclass()
py.hello()
```


과제

# 5-1




링크가 2개인 로봇의 말단부 위치 $(x, y)$에 대해서 각 링크의 회전각은 아래와 같은 식을 만족합니다.

$$\theta_1 = \operatorname{atan2}(y, x) \pm \arccos \frac{x^2+y^2+a_1^2-a_2^2}{2a_1\sqrt{x^2+y^2}}$$

$$\theta_2 = \mp \arccos \frac{x^2 + y^2 - a_1^2 - a_2^2}{2a_1 a_2}$$

(복부호 동순)

이 때, Joint의 위치를 구하기 바랍니다.
단, 평면상에서 링크가 2개인 로봇의 Joint의 위치는 총 2개가 나옵니다.

# 5-1 (이어서)

$$\operatorname{atan2}(y,x) = \begin{cases} \arctan\left(\frac{y}{x}\right) & (x > 0) \\ \arctan\left(\frac{y}{x}\right) + \pi & (x < 0 \; and \; y \geq 0) \\ \arctan\left(\frac{y}{x}\right) - \pi & (x < 0 \; and \; y < 0) \\ +\frac{\pi}{2} & (x = 0 \; and \; y > 0) \\ -\frac{\pi}{2} & (x = 0 \; and \; y < 0) \\ undefined & (x = 0 \; and \; y = 0) \end{cases}$$

| 입력 예시: $a_1, a_2, x, y$ 입력 | 출력 예시 |
| --- | --- |
| 4 6 3 5 | [-2.654205214895146, 2.992523128937087]<br>[3.889499332542204 3, -0.9336995995253227] |