

# 클래스와 객체

클래스와 객체를 활용한 객체지향 프로그래밍 하기

# 클래스와 객체

```
In [1]: class Calculator:          선언과 동시에 클래스 정의(생성)
        def setdata(self, first, second):
            self.first = first
            self.second = second
```

```
In [2]: a = Calculator()
```

```
In [3]: a.setdata(5,6)
```

```
In [4]: print (a.first)
        print (a.second)
```

5  
6

정의클래스를 사용하기 위해 객체를 생성함

클래스는 데이터와 메소드로 구성됨(반드시 필요한 것은 아님)



# 클래스와 객체

- 클래스 Calculator, 객체 a 생성 및 객체로 메소드 호출하기
- 같은 클래스의 객체들은 서로 영향 받지 않고 독립적으로 객체 값을 유지

class Calculator:

def setdata(self, first, second):

self.first = first

self.second = second

클래스에 메소드 setdata()를 작성  
매개변수 self 는 setdata()를 호출하는 객체를 의미

a = Calculator()

a.setdata(4, 2)



# 클래스와 객체

- 클래스 Calculator 에 add() 메소드 추가

```
class Calculator:  
    def setdata(self, first, second):  
        self.first = first  
        self.second = second  
    def add(self):  
        result = self.first + self.second  
        return result
```

```
a = Calculator()
```

- 객체 a에 4와 2를 설정
- 객체 a에 설정된 값의 합을 구하기 위해 add() 메소드 호출

```
a.setdata(4, 2)
```

```
a.add()
```

# 클래스와 객체

- 빼기 **sub**, 곱하기 **mul** 및 나누기 메소드 **div** 를 추가함

```
In [19]: class Calculator:
          def setdata(self, first, second):
              self.first = first
              self.second = second
          def add(self):
              result = self.first + self.second
              return result
          def sub(self):
              result = self.first - self.second
              return result
          def mul(self):
              result = self.first * self.second
              return result
          def div(self):
              result = self.first / self.second
              return result
```

```
In [20]: a = Calculator()
```

```
In [22]: a.setdata(4, 2)
```

```
In [24]: a.sub()
```

```
Out[24]: 2
```

```
In [25]: a.mul()
```

```
Out[25]: 8
```

```
In [26]: a.div()
```

```
Out[26]: 2.0
```

# (실습)클래스와 객체

- 실수값 대신 정수값을 출력하는 메소드 `ddiv()` 를 추가한다

# 클래스와 객체

- 객체 a의 setdata() 호출없이 add() 호출

```
In [35]: a = Calculator()
```

```
In [36]: a.add()
```

```
-----  
AttributeError                                Traceback (most recent call last)  
<ipython-input-36-fddc01c87ea9> in <module>  
----> 1 a.add()  
  
<ipython-input-34-4e16001d1307> in add(self)  
      4         self.second = second  
      5     def add(self):  
----> 6         result = self.first + self.second  
      7         return result  
      8     def sub(self):  
  
AttributeError: 'Calculator' object has no attribute 'first'
```

# 클래스와 객체

- 생성자
  - 객체가 생성될 때 자동으로 호출하는 메소드
  - init 앞 뒤로 붙은 \_\_는 언더스코어(\_) 두 개를 붙여서 사용  
\_\_init\_\_

```
class Calculator:  
    def __init__(self, first, second):  
        self.first = first  
        self.second = second
```

객체 생성시 2개의 값이 주어져야 함



# 클래스와 객체

- 객체 a 생성 오류

```
class Calculator:
    def __init__(self, first, second):
        self.first = first
        self.second = second
    def setdata(self, first, second):
        self.first = first
        self.second = second
    def add(self):
        result = self.first + self.second
        return result
    def sub(self):
        result = self.first - self.second
        return result
    def mul(self):
        result = self.first * self.second
        return result
    def div(self):
        result = self.first / self.second
        return result
    def ddiv(self):
        result = self.first // self.second
        return result
```

```
a = Calculator()
```

```
-----
TypeError                                 Traceback (most recent call last)
<ipython-input-38-b3795dc48d5e> in <module>
----> 1 a = Calculator()

TypeError: __init__() missing 2 required positional arguments: 'first' and 'second'
```

# 클래스와 객체

- first와 second에 해당되는 값을 전달하여 객체 a를 생성함

```
a = Calculator(4,2)
```

```
a.first
```

4

---

```
a.second
```

2

- 객체 a 의 add() 메소드를 호출하여 결과를 확인한다

# 클래스와 객체

- 나누기 연산의 오류 원인 찾기

```
a = Calculator(4,0)
```

```
a.first
```

```
4
```

```
a.second
```

```
0
```

```
a.div()
```

```
-----  
ZeroDivisionError                                Traceback (most recent call last)  
<ipython-input-46-656340726662> in <module>  
----> 1 a.div()  
  
<ipython-input-37-ddccbd8fc97b> in div(self)  
    16         return result  
    17     def div(self):  
----> 18         result = self.first / self.second  
    19         return result  
    20     def ddiv(self):  
  
ZeroDivisionError: division by zero
```

# 클래스와 객체

- 나누기 연산의 오류 해결

```
def div(self):  
    if self.second == 0: # 나누는 값이 0인 경우 0을 리턴하도록 수정  
        return 0  
    else:  
        return self.first / self.second
```

```
a = Calculator(4,0)
```

```
a.first
```

```
4
```

```
a.second
```

```
0
```

```
a.div()
```

```
0
```

# 클래스와 객체

- 클래스 변수
  - 클래스의 모든 객체가 해당 변수를 공유함
  - 클래스명.클래스변수 또는 객체.클래스변수 로 사용

```
class Calculator:
    developer = "강윤희"
    def __init__(self, first, second):
        self.first = first
        self.second = second
```

```
a = Calculator(4,0)
```

```
a.developer
```

```
'강윤희'
```

```
Calculator.developer
```

```
'강윤희'
```