python

■ class
클래스안에 구성된 함수는 method라고 함
self → 객체를 알려주는 명령어

```
 # def getdata(self, 매개변수들...)
# a = FourCal()

# a.setdata(4,5)
# print(a.first)
# print(a.second)
# class에 객체를 넘겨줘서 받음

# a = FourCal()
# b = FourCal()
# a.setdata(4,2)
# b.setdata(3,7)
# print(id(a.first))
# print(id(b.first))
```

■ 더하기 기능 만들기

```
# class FourCal:
#     def setdata(self, first, second):
#         self.first = first
#         self.second = second

#     def add(self):
#         result = self.first + self.second
#         return result

# a = FourCal()
# a.setdata(4,2)
# print(a.add())
```

■ 나머지 사칙연산(빼기, 곱하기, 나누기)

```
# class FourCal:
#     def setdata(self, first, second):
#         self.first = first
#         self.second = second
```

```
#       def sub(self):
#           result = self.first - self.second
#           return result

# a = FourCal()
# a.setdata(4,2)
# print(a.sub())

# class FourCal:
#       def setdata(self, first, second):
#           self.first = first
#           self.second = second

#       def mul(self):
#           result = self.first * self.second
#           return result

# a = FourCal()
# a.setdata(4,2)
# print(a.mul())

# class FourCal:
#       def setdata(self, first, second):
#           self.first = first
#           self.second = second

#       def div(self):
#           result = self.first / self.second
#           return result

# a = FourCal()
# a.setdata(4,2)
# print(a.div())
```

■ 객체명 = 빈생성자

```
# class FourCal:
#       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 mul(self):
#         result = self.first * self.second
#         return result
#     def sub(self):
#         result = self.first - self.second
#         return result
#     def div(self):
#         result = self.first / self.second
#         return result

# a = FourCal(4, 2)

# print(a.add())
# print(a.div())
# print(a.sub())
# print(a.mul())

class FourCal:
    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 mul(self):
        result = self.first * self.second
        return result
    def sub(self):
        result = self.first - self.second
        return result
    def div(self):
        result = self.first / self.second
```

```
        return result

■ Class의 상속
# add(), mul(), sub(), div() -> FourCal Class
# class MoreFourCal(FourCal):
#     def pow(self):
#         result = self.first ** self.second
#         return result

# a = MoreFourCal(4,2)
# print(a.pow())
# print(a.add())
# print(a.sub())
# print(a.mul())
# print(a.div())

# Method Overiding
# class SafeFourCal(FourCal):
#     def div(self):
#         if self.second == 0:
#             return  0
#         else:
#             return self.first / self.second
# a = SafeFourCal(4,0)
# print(a.div())

■ Class 변수와 객체변수
# a = FourCal(4,2)
# a.first = 4 <- 객체변수
# a.second = 2 <- 객체변수
# b = FourCal(6,2)
# b.first = 6 <- 객체변수
# b.second = 2 <- 객체변수

class Family:
    lastname = "박"

    def setdata(self, firstname):
        self.firstname = firstname

print(Family.lastname)
```

```
a = Family()
b = Family()

print(a.lastname)
print(b.lastname)

class Family:
    lastname = "박"

    def setdata(self, firstname):
        self.firstname = firstname

print(Family.lastname)

a = Family()
b = Family()
a.setdata("길동")
b.setdata("찬호")

print(a.lastname, a.firstname)
print(b.lastname, b.firstname)
```

■ 패키지 만들기

1. 파일 및 디렉토리 만들기

C:/git_repo/game/__init__.py
C:/git_repo/game/sound/__init__.py
C:/git_repo/game/sound/echo.py
C:/git_repo/game/graphic/__init__.py
C:/git_repo/game/graphic/render.py

2. 각 디렉토리에 __init__.py 파일을 만들어 놓기만 하고 내용은 일단 비워 둔다.

3. echo.py 파일은 다음과 같이 만든다.

```
# echo.py
def echo_test():
    print("echo")
```

4. render.py 파일은 다음과 같이 만든다.

```
# render.py
def render_test():
    print("render")
```

cmd에서
set PYTHONPATH=C:₩git_repo
python 입력
import.game.sound.echo 입력
game.sound.echo.echo_test() 입력
echo 출력이 됨.
※ CMD에서 해야지 오류가 나지 않음

프로그램의 오류
구문오류 : 오타
로직오류 : 구문오류 외의 오류

■ 오류처리 예외문
● try, except문

```
# c = 0
# a = [1,2,3]
# try:
#     c = 10 / 2
#     print(a[1])

# 0으로 나누기 오류
# except ZeroDivisionError:
#     print("0으로 나누기 했습니다.")

# 인덱스 오류
# except IndexError:
#     print(e)
# else:
#     print("오류가 있던없던 실행됨.")

# print("항상 실행됨")

# try:
#     age = int(input('나이를 입력하세요 : '))
# except:
#     print('입력이 정확하지 않습니다.')
# else:
```

```
#       if age <= 18:
#           print('미성년자는 출입금지입니다.')
#       else:
#           print('환영합니다.')
```

■ 오류 일부러 발생시키기

```
class Bird:
    def fly(self):
        raise NotImplementedError

class Eagle(Bird):
    pass

eagle = Eagle()
eagle.fly()
```