파이썬 심화 / 가상환경

5번째 세션

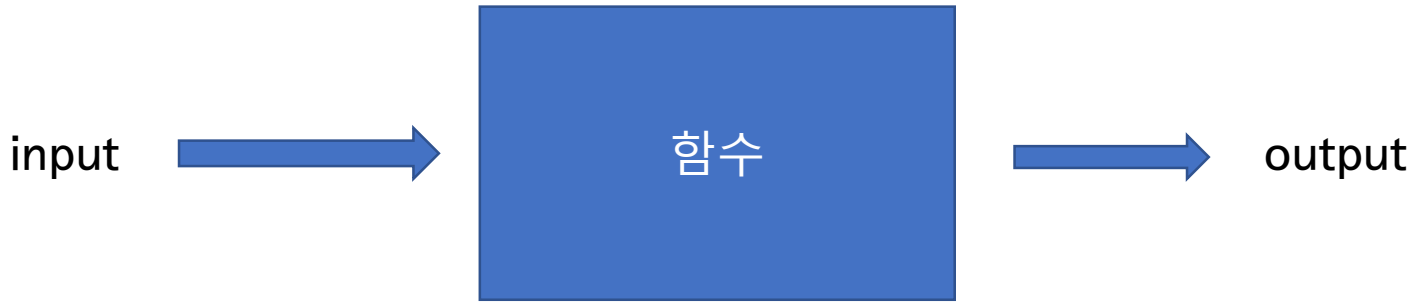
NEXT X LIKELION

박지환

함수란?

함수(function)

하나의 특별한 목적의 작업을 수행하기 위해 독립적으로 설계된 코드의 집합



함수란?

왜 써요?

반복적인 프로그래밍을 피할 수 있습니다.

특정 작업을 여러 번 반복해야 할 때 :

해당 작업을 수행하는 함수를 작성
-> 작업이 필요할 때마다 함수 호출

함수란?

왜 써요?

프로그램 흐름을 일목요연하게 볼 수 있다

like 분업

-> 오류가 어디서 나는지도 바로 파악할 수 있다.

함수 만들기

```
def 함수명(매개변수):
    <수행할 문장1 >
    <수행할 문장2 >
    ...
    return 결과값
```

* return을 만나면 함수가 종료!

함수 형태

1. 입력값 o, 결과값 o
2. 입력값 o, 결과값 x
3. 입력값 x, 결과값 o
4. 입력값 x, 결과값 x

함수형태

입력값 O, 결과값 O

```
def add(a, b):
    return a+b
```

함수형태

입력값 O, 결과값 X

```
def show_add(a, b):
    print(a+b)
```

함수형태

입력값 x, 결과값 o

```
def get_number():
    return 4
```

함수형태

입력값 x, 결과값 x

```
def say_hello():
    print('hello')
```

함수 실행

지금까지 함수를 만들었어요.

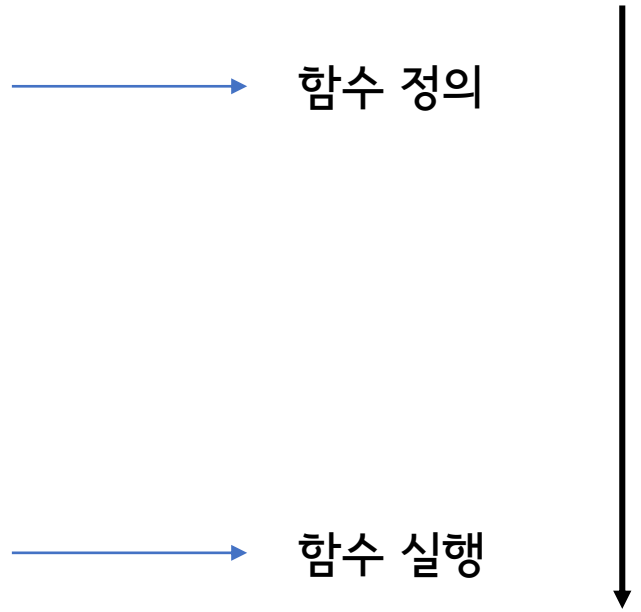
어떻게 실행시키죠?

함수명(인자)

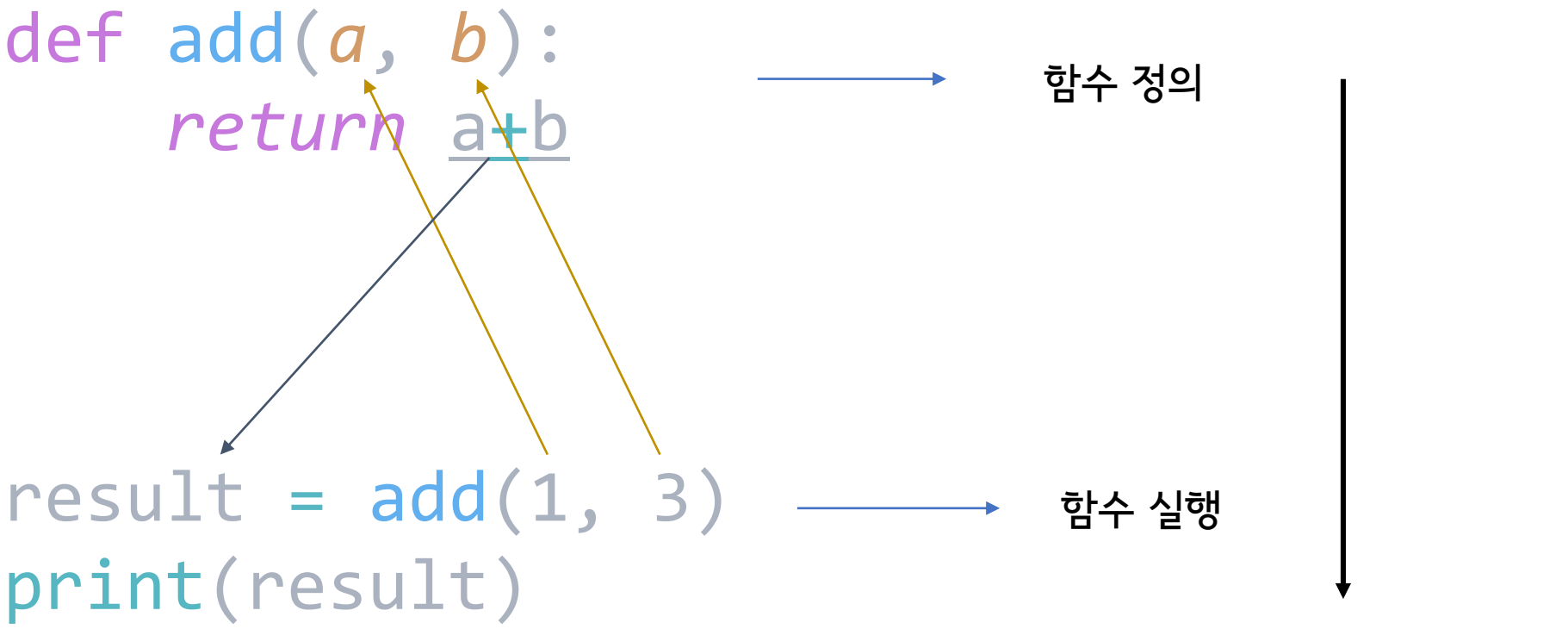
함수 실행

```
def say_hello():
    print('hello')

say_hello()
```



함수 실행



함수 실행

함수 실행은 함수 정의 다음에!

```
say_hello()
```

함수 실행

```
def say_hello():
    print('hello')
```

함수 정의

실행되지 않아요 ㅜㅜ

함수 종류

함수가 종료되는 시점

1. 코드를 다 실행했을 때
2. return을 만날 때

* 함수를 의도적으로 종료시키기 위해 return을 사용하기도 함!

함수 결과값

함수의 결과값은 항상 하나

```
def add_and_mul(a, b):
    return a+b
    return a*b
```

함수 종료

실행 x

함수 결과값

함수의 결과값은 항상 하나

```
def add_and_mul(a, b):
    return a+b, a*b

result = add_and_mul(2, 3)
print(result)
```

(5,6)
=> 결국 튜플 하나를 반환!

함수란?

실습

1. session5 폴더 만들기
2. 폴더 안에 들어가 def_exercise.py 파일 만들기

Q1. 주어진 자연수가 홀수인지 짝수인지 판별해 주는 함수 작성 및 실행

- 인자로 숫자 하나를 받는다
- 해당 숫자가 홀수면 '홀수', 짝수면 '짝수' 출력
- % 연산자 사용

Q2. 인자에 따라 홀수 구구단(1,3,5,7,9단) 또는 짝수 구구단(2,4,6,8단)을 출력하는 함수 작성 및 실행

- 인자로 숫자 하나를 받는다.
- 해당 숫자가 홀수면 홀수 구구단 출력
- 해당 숫자가 짝수면 짝수 구구단 출력
- 두개 숫자 포맷팅 => print("%d * %d" %(1,2))

정답

Q1

```
def is_odd(num):
    if(num % 2 == 0):
        print('짝수')
    else:
        print('홀수')
```

정답

Q2

```
def gugudan_odd_or_even(num):
    if(num % 2 == 0):
        i = 2
        while i < 10:
            for j in range(1, 10):
                print("%d * %d" % (i, j))
            i += 2
    else:
        i = 1
        while i < 10:
            for j in range(1, 10):
                print("%d * %d" % (i, j))
            i += 2
```

실습

Q2

코드가 너무 기네요

함수로 나누어 볼까요?

- 홀수단(1,3,5,7,9단) 출력하는 함수 만들기
- 짝수단(2,4,6,8단) 출력하는 함수 만들기
- gugudan_odd_or_even 함수 안에서 실행

정답

Q2

홀수만 출력하는 gugudan_odd함수

```
def gugudan_odd():
    i = 1
    while i < 10:
        for j in range(1,  10):
            print("%d * %d" % (i, j))
        i += 2
```

정답

Q2

짝수만 출력하는 gugudan_even함수

```
def gugudan_even():
    i = 2
    while i < 10:
        for j in range(1, 10):
            print("%d * %d" % (i, j))
        i += 2
```

정답

Q2

함수 실행하는 gugudan_odd_or__even함수

```
def gugudan_odd_or_even(num):
    if(num % 2 == 0):
        gugudan_even()
    else:
        gugudan_odd()



gugudan_odd_or_even(3)
```

실습

Q2

함수로 합쳐 볼까요?

- 인자에 따라 홀수단 또는 짝수단을 출력하는 함수 만들기
- 인자가 1일 경우 홀수단 출력
- 인자가 2일 경우 짝수단 출력
- gugudan_odd_or_even 함수 안에서 실행

정답

Q2

인자에 따라 출력하는 gugudan함수

```
def gugudan(num):
    i = num
    while i < 10:
        for j in range(1, 10):
            print("%d * %d" % (i, j))
        i += 2
```

정답

Q2

gugudan_odd_or_even 함수

```
def gugudan_odd_or_even(num):
    if(num % 2 == 0):
        gugudan(2)
    else:
        gugudan(1)
```

실습

Q3. 주어진 숫자에 따라 구구단 출력하는 함수 정의 및 실행

- 인자로 숫자 하나를 받는다.
- 해당 숫자만큼 구구단 출력
  - Ex) 인자가 5
    - 1~5단까지 출력
  - Ex) 인자가 8
    - 1~8단까지 출력

실습

Q3

```
def gugudan_input(num):
    i = 1
    while i <= num:
        for j in range(1, 10):
            print("%d*%d" % (i, j))
        i += 1
```

클래스의 필요성

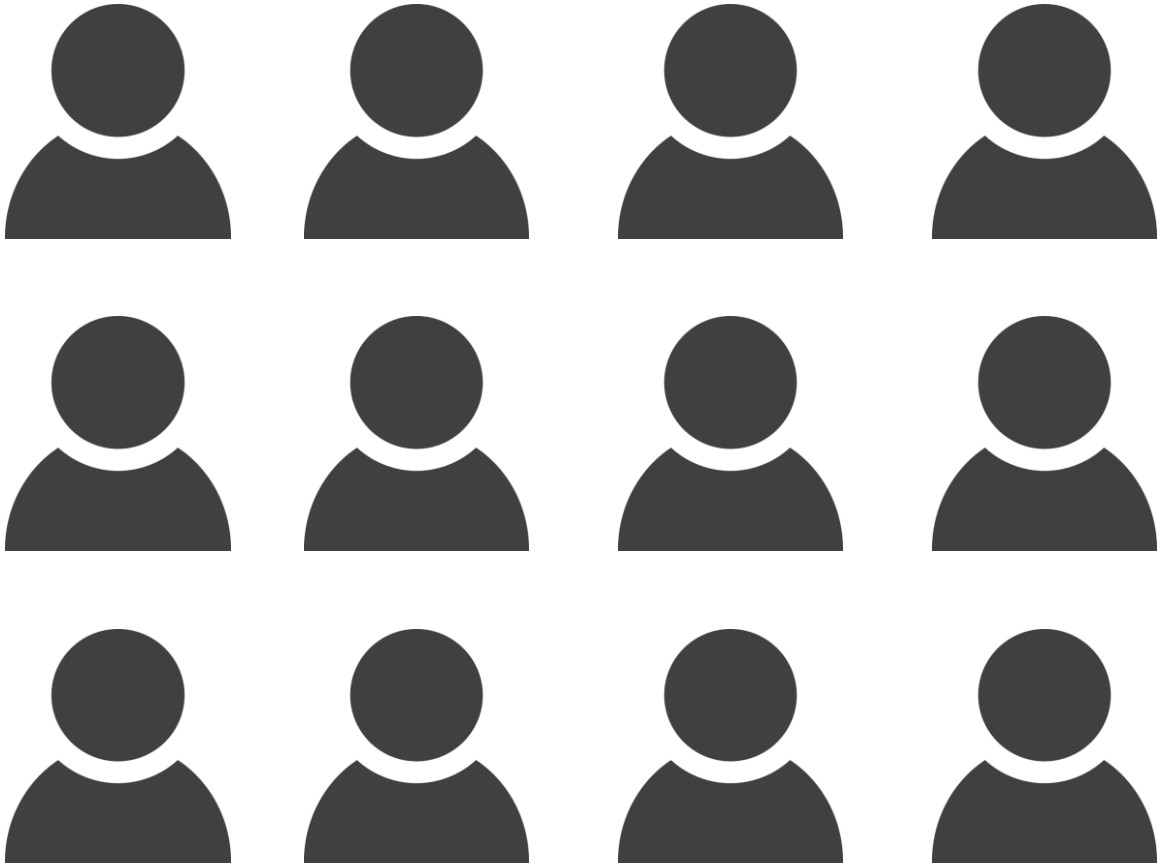
절차만으로 모든 걸 프로그래밍 할 수 있을까?

명령만으로 모든 걸 프로그래밍 할 수 있을까?

함수만으로 모든 걸 프로그래밍 할 수 있을까?

클래스의 필요성

클래스의 필요성



1. 창문 닦기
2. 칠판 닦기
3. 바닥 쓸기

클래스의 필요성

1. 창문 닦기

깨끗한 행주랑 원덱스를 챙겨서 101호 창문 닦아!

깨끗한 행주랑 원덱스를 챙겨서 102호 창문 닦아!

깨끗한 행주랑 원덱스를 챙겨서 103호 창문 닦아!

클래스의 필요성

2. 칠판 닦기

깨끗한 지우개랑 새 분필 챙겨서 101호 칠판 닦아!

깨끗한 지우개랑 새 분필 챙겨서 102호 칠판 닦아!

깨끗한 지우개랑 새 분필 챙겨서 103호 칠판 닦아!

클래스의 필요성

3. 바닥 쓸기

빗자루랑 쓰레받기 챙겨서 101호 바닥 쓸어!

빗자루랑 쓰레받기 챙겨서 102호 바닥 쓸어!

빗자루랑 쓰레받기 챙겨서 103호 바닥 쓸어!

클래스의 필요성

하 일일이 지시하니 힘드네 ㅜㅜ

좀 편하게 시킬 순 없을까..?

역할을 만들자..!

클래스의 필요성

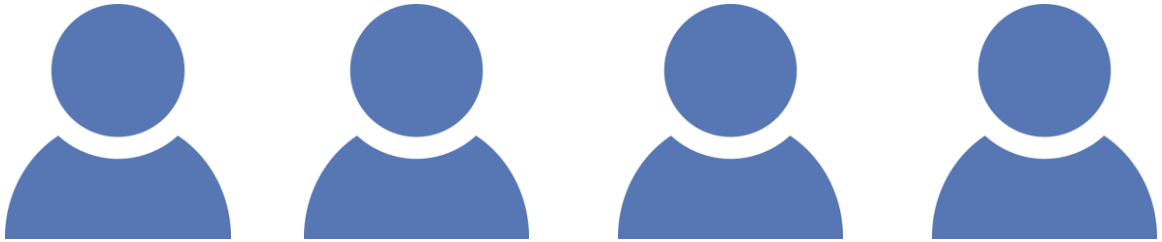

창문 닦기 역할
행주, 원덱스
주어진 교실 창문 닦기


칠판 닦기 역할
지우개, 분필
주어진 교실 칠판 닦기


바닥 쓸기 역할
빗자루, 쓰레받기
주어진 교실 바닥 쓸기

클래스의 필요성

1. 창문 닦기
2. 칠판 닦기
3. 바닥 쓸기



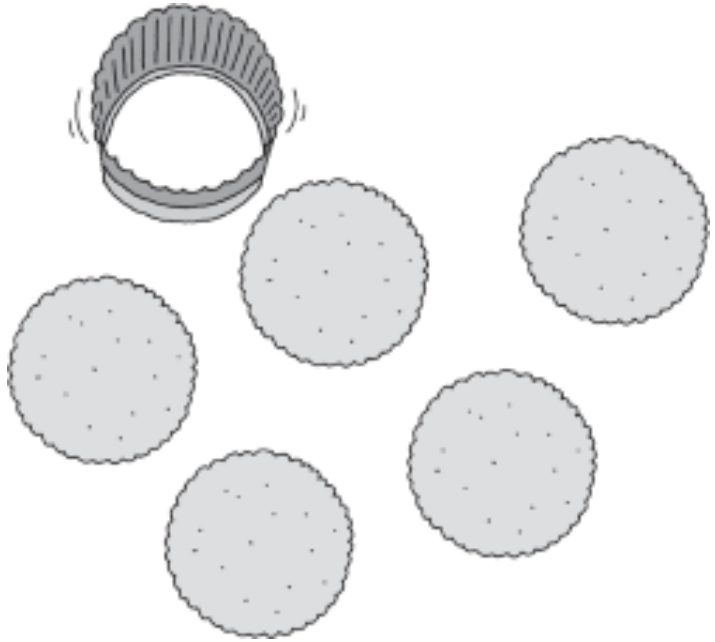
창문 닦기 역할

칠판 닦기 역할

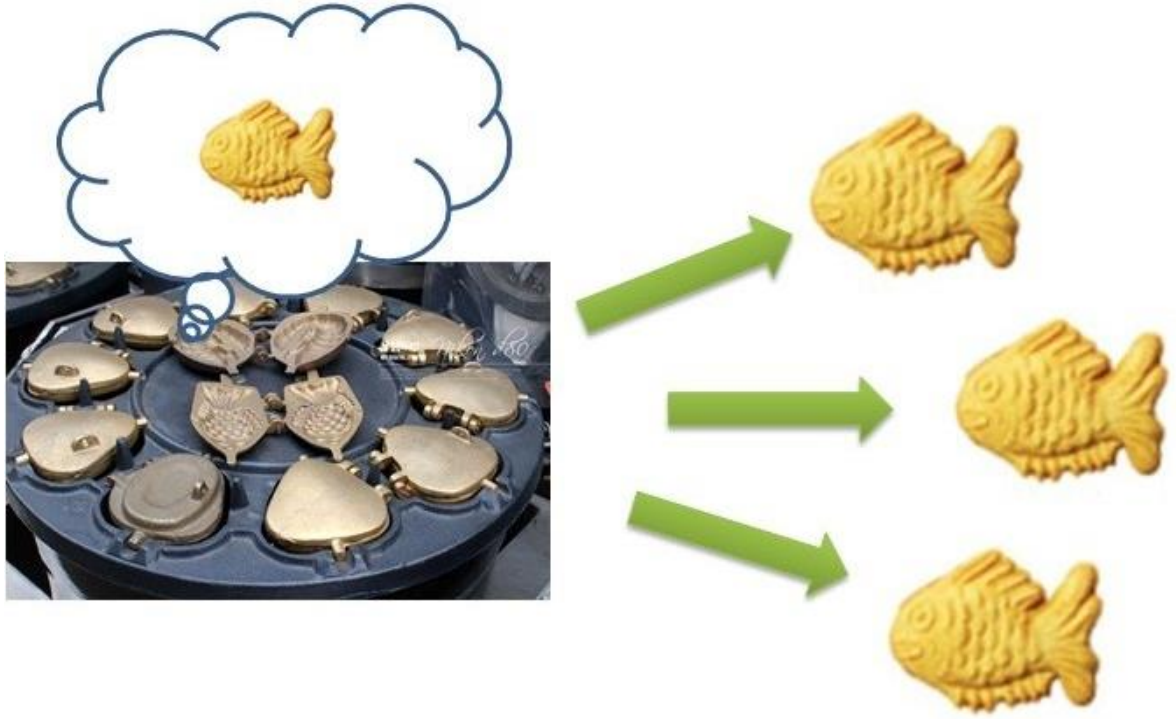
바닥 쓸기 역할

클래스의 필요성

클래스, 객체란?



과자 틀 -> 과자



붕어빵 틀 -> 붕어빵

클래스, 객체란?

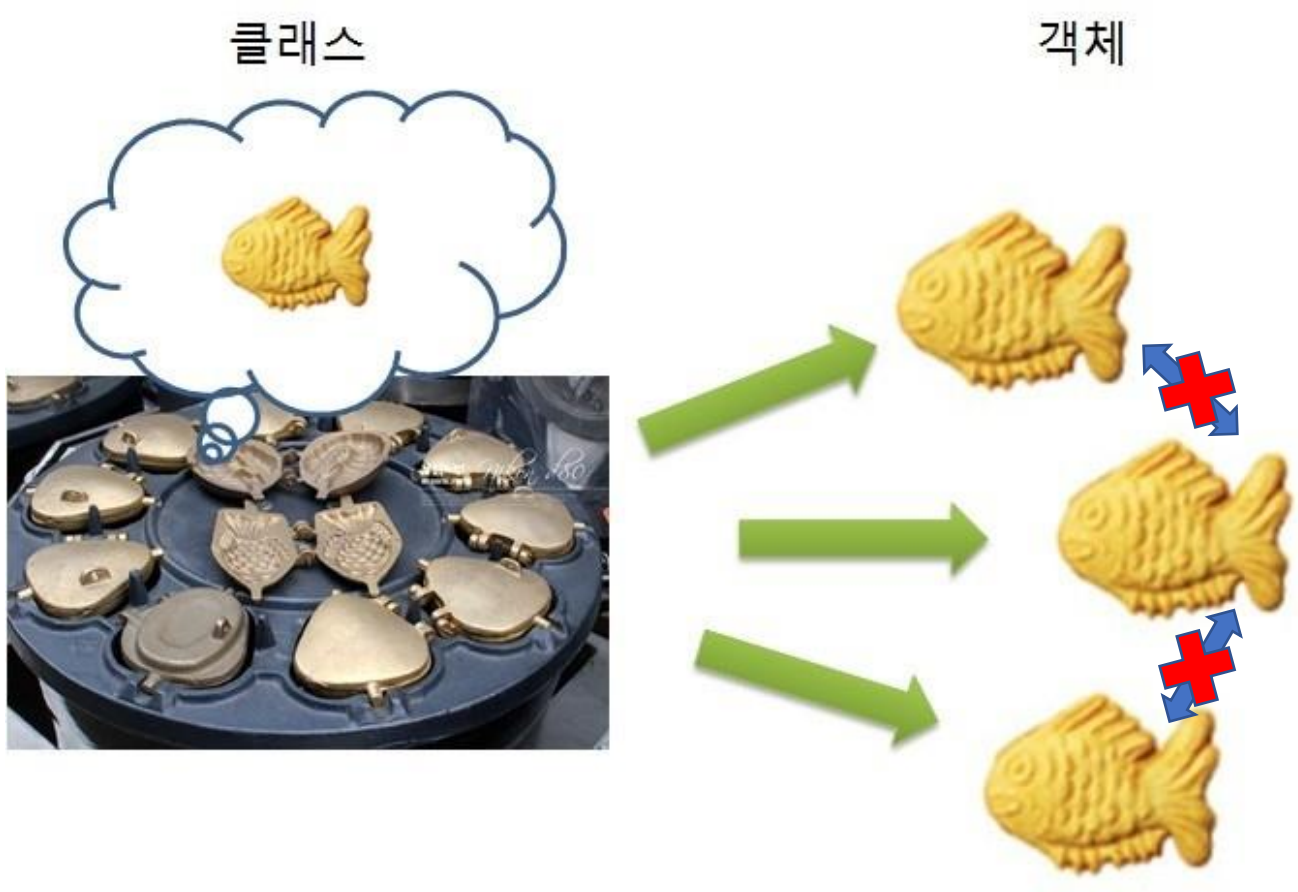
클래스(class)란 똑같은 무엇인가를 계속해서 만들어 낼 수 있는 설계 도면

Like 과자 틀, 붕어빵 틀

객체(object)란 클래스로 만든 피조물

Like 과자, 붕어빵

클래스, 객체란?

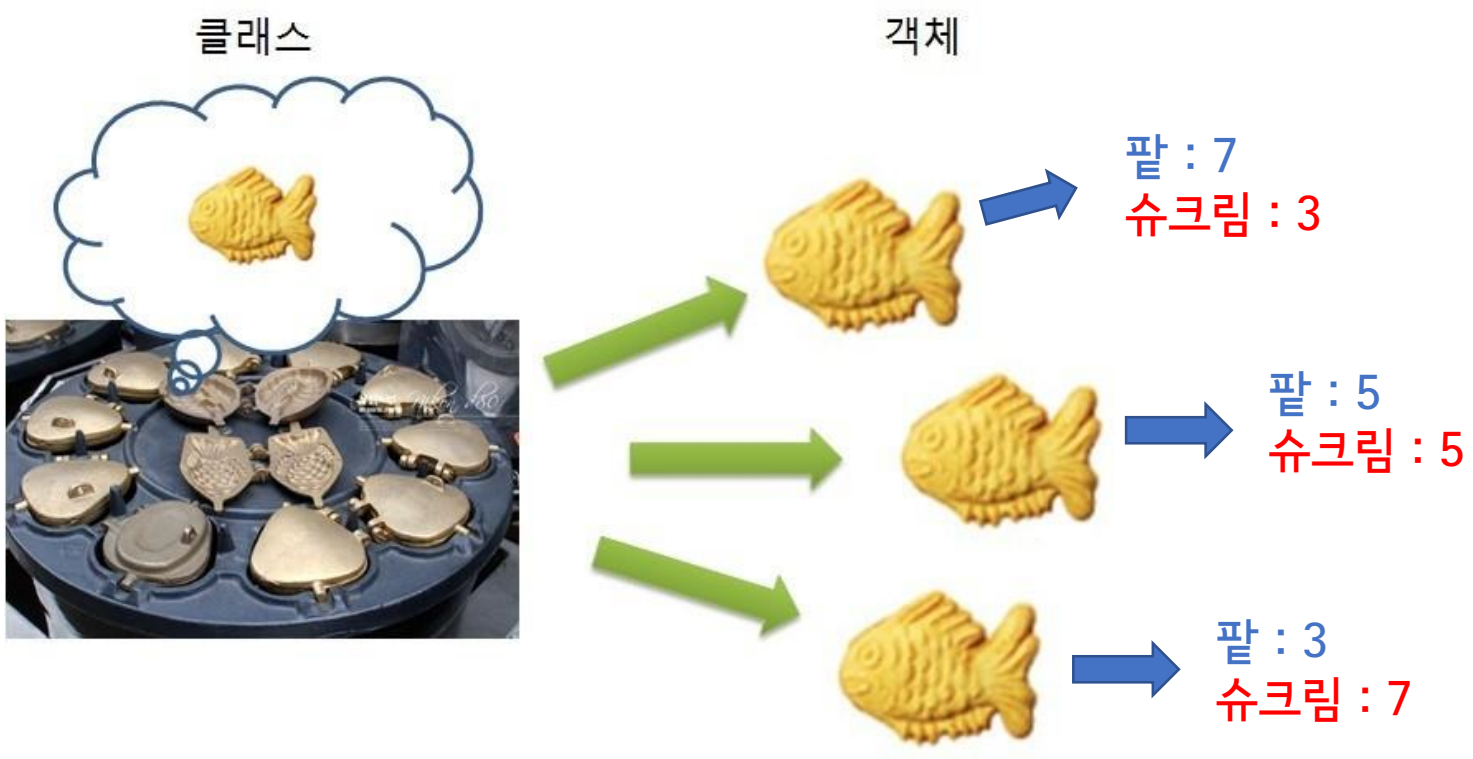


붕어빵끼리는 서로 영향을 주지 않는다

=

클래스로 만든 객체들은 서로 영향을 주지 않는다

클래스, 객체란?



붕어빵마다 재료 비율이 다르다

=

객체마다 고유한 성격을 가진다.

클래스, 객체란?

붕어빵..? 과자..?

먹는거 아닌가요?

모르셔도 됩니다!

Just do it!

클래스 만들기

아주 간단히, 신속히 클래스를 만들어 봅시다..!

계산기 클래스를 통해 계산기를 만들고 실행시켜보아요

클래스 만들기

계산기 클래스를 만들어 봅시다..!

class.py

1. 클래스 만들기

```
class FourCal:
    pass
```

2. 객체 만들기

```
a = FourCal()
```

a는 객체이다. True

a는 FourCal 클래스의 인스턴스이다. True

* pass -> 아무것도 수행하지 않는 문법

클래스 만들기

이제 내용을 넣어볼까요..?

클래스 = 속성 + 메서드

클래스 만들기

클래스 = 속성 + 메서드

속성

누구 계산기인지 저장해야겠죠?

이름(속성) = 박지환(값)

클래스 만들기

클래스 = 속성 + 메서드

메서드 = 객체 안의 함수

그냥 함수라고 생각하시면 됩니다

계산기에 기능을 넣어야겠죠?

클래스 만들기

메서드 만들기

 # class.py

```
class FourCal:
    def add(self, n1, n2):
        result = n1 + n2
        return result


calculator1 = FourCal()
print(calculator1.add(3, 4))
```

→ 7

* 객체를 통해 클래스의 메서드를 호출하려면
calculator1.add처럼 도트(.) 연산자 사용

클래스 만들기

메서드 만들기

self ?

메서드를 실행하는 해당 객체

- 매개변수 self에는 메서드를 호출한 객체가 자동으로 전달된다.

- 모든 메서드의 첫번째 인자는 self를 받는다.

클래스 만들기

메서드 만들기

 # class.py

```
class FourCal:
    def add(self, n1, n2):
        result = n1 + n2
        return result

calculator1 = FourCal()
calculator2 = FourCal()
calculator3 = FourCal()

print(calculator1.add(3, 4))
print(calculator2.add(2, 1))
print(calculator3.add(6, 3))
```


self -> calculator1


self -> calculator2


self -> calculator3

클래스 만들기

속성 지정하기

 # class.py

```
class FourCal:
    def __init__(self, name, age):
        self.name = name
        self.age = age

calculator1 = FourCal("박지환", 25)
print(calculator1.name, calculator1.age)
```

'박지환'

25

클래스 만들기

속성 지정하기

__init__

생성자(constructor)

객체가 생성될 때 자동으로 호출되는 메서드

* __로 시작하는 함수는 호출하는 함수가 아닙니다..!

클래스 만들기

속성 지정, 메서드 작성 완료..!

class.py

```python
class FourCal:
    def __init__(self, name, age):
        self.name = name
        self.age = age

    def add(self, num1, num2):
        return num1+num2
```

클래스 만들기

만들어보죠

실습

1. 멋쟁이사자처럼 폴더 안에 session5 디렉토리 생성
2. class.py 파일 생성
3. FourCal 클래스 만들기
    - 이름, 나이, 학교 속성을 지정하는 __init__ 메서드 작성
    - 사칙연산(더하기, 빼기, 나누기, 곱하기) 메서드 작성
4. FourCal 클래스를 통해 계산기(객체) 만들기
5. 속성, 메서드 결과 출력해보기

클래스 만들기

class.py

```python
class FourCal:
    def __init__(self, name, age, university):
        self.name = name
        self.age = age
        self.university = university
    def add(self, num1, num2):
        return num1+num2
    def sub(self, num1, num2):
        return num1-num2
    def mul(self, num1, num2):
        return num1*num2
    def div(self, num1, num2):
        return num1/num2
```

먼가 부족한데..

클래스 만들기

예외 처리

```
print(calculator1.div(3,0))
```

오류 발생!
=> 프로그램 멈춤
=> 0으로 나눌 때 처리를 해 줘야한다.

```
def div(self, num1, num2):
    if(num2 == 0):
        print('0으로 나눌 수 없습니다')
        return None
    return num1/num2
```

클래스 만들기

최종 정답

class.py

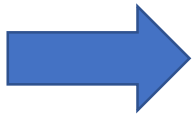
```python
class FourCal:
    def __init__(self, name, age, university):
        self.name = name
        self.age = age
        self.university = university
    def add(self, num1, num2):
        return num1+num2
    def sub(self, num1, num2):
        return num1-num2
    def mul(self, num1, num2):
        return num1*num2
    def div(self, num1, num2):
        if(num2 == 0):
            print('0으로 나눌 수 없습니다')
            return None
        return num1/num2
```

클래스 상속

몰라도 아무 상관 없습니다!

상속 ?

어떤 클래스를 만들 때 다른 클래스의 기능을 물려받을 수 있게 만드는 것



상속 개념을 이용해서 FourCal 클래스에 제곱 기능을 추가해보아요..!

클래스 만들기

몰라도 아무 상관 없습니다!

class 클래스 이름(상속할 클래스 이름)

```
class MoreFourCal(FourCal):
    pass
```

상속 받은 FourCal 클래스의 모든 속성, 메서드 사용 가능

```
computer = MoreFourCal('박지환', 25, '고려대학교')
```

FourCal 클래스의 __init__ 메서드 사용

```
print(computer.name, computer.age, computer.university)

print(computer.add(3,4))
print(computer.sub(3,4))
print(computer.mul(3,4))
print(computer.div(3,4))
```

FourCal 클래스의 add 매서드 사용
FourCal 클래스의 sub 메서드 사용
FourCal 클래스의 mul 메서드 사용
FourCal 클래스의 div 메서드 사용

클래스 상속

메서드 추가하기

상속 받은 후 메서드 추가하기

```
class MoreFourCal(FourCal):
    def pow(self, a, b):
        return a**b
```

MoreFourCal이 가지고 있는 메서드 종류

- __init__
- add
- mul
- div
- sub
- pow

=> 코드는 짧은데 내용은 많다..!

클래스 상속

마지막..!

메서드 오버라이딩(overriding, 덮어쓰기)

부모 클래스(상속한 클래스)에 있는 메서드를 동일한 이름으로 다시 만드는 것

- 부모클래스의 메서드 대신 오버라이딩한 메서드가 호출된다

```
class FourCal:
    def sub(self, num1, num2):
        return num1-num2
```

```
class MoreFourCal(FourCal):
    def sub(self, num1, num2):
        return num2 - num1
```

```
computer = MoreFourCal('박지환', 25, '고려대학교')
print(computer.sub(2,3))
```

클래스

장고 코드 맛보기

그래서 왜 배우는건데??

클래스 만들기

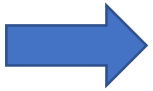
장고 코드 맛보기

데이터베이스
=> 테이블(객체)을 만들고 항목의 내용을 입력(객체)

이름	나이	학교	학년

클래스

박지환	25	고려대학교	3



객체

클래스 만들기

장고 코드 맛보기

붕어빵 만들기 == 데이터베이스에 데이터 저장하기

클래스 만들기

장고 코드 맛보기

```
class Post(models.Model):
    author = models.ForeignKey(settings.AUTH_USER_MODEL, on_delete=models.CASCADE)
    title = models.CharField(max_length=200)
    text = models.TextField()
    created_date = models.DateTimeField(
            default=timezone.now)
    published_date = models.DateTimeField(
            blank=True, null=True)

    def publish(self):
        self.published_date = timezone.now()
        self.save()

    def __str__(self):
        return self.title
```

모듈이란?

모듈 ?

함수나 변수 또는 클래스를 모아 놓은 파일

- 다른 파이썬 프로그램에서 불러와 사용할 수 있게끔 만든 파이썬 파일

- 파이썬 파일은 모두 모듈이다

모듈 사용 방법

1. import 모듈이름

2. from 모듈이름 import 함수, 변수, 클래스

모듈이란?

mod1.py

```python
def add(a, b):
    return a + b

def sub(a, b):
    return a-b
```

main.py

```python
import mod1
# import .mod1

print(mod1.add(1, 2))
print(mod1.sub(1, 2))
```

main.py

```python
from mod1 import add, sub

print(add(1, 2))
print(sub(1, 2))
```

모듈이란?

```
from mod1 import *
```

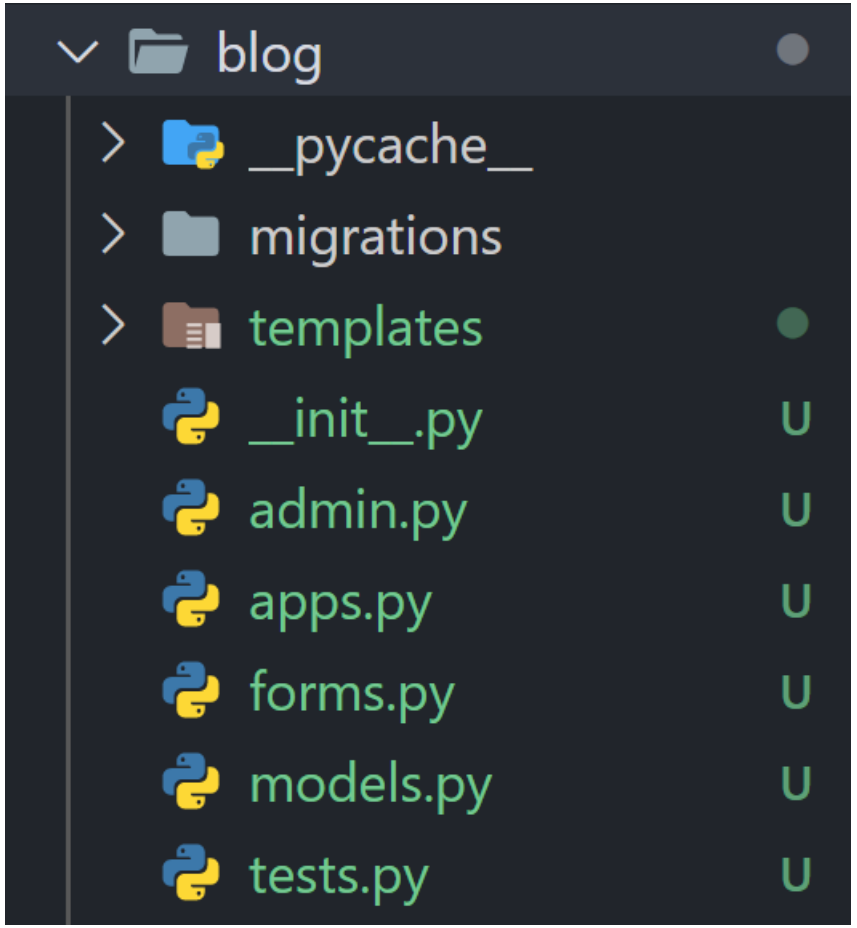
- mod1 파일(모듈)에 있는 모든걸 쓰겠다.

```
if __name__ == "__main__":

    print(add(1, 4))
    print(sub(4, 2))
```

- 다른 파일에서 불러서 사용할 때는 실행되지 않음
  - 직접 파일을 실행할 때만 실행

패키지란?



패키지 ?

디렉터리와 파이썬 모듈

- 도트(.)를 사용하여 파이썬 모듈을 계층적(디렉터리 구조)으로 관리할 수 있게 해준다.

  - 프로젝트라고 생각하면 된다.

패키지란?

__init__.py

해당 디렉토리가 패키지의 일부임을 알려주는 역할

- __init__.py가 없다면 패키지로 인식되지 않는다.
- Python 3.3부터는 없어도 패키지로 인식하지만 호환을 위해 만드는 것이 좋다.

가상환경이란?

pip

PIP

파이썬으로 작성된 패키지 라이브러리를 관리해주는 시스템
-> 남들이 만들어 놓은 코드를 다운받을 때 사용!

파이썬 3.4부터 파이썬을 설치하면 기본적으로 포함되어 있다.

numpy라는 패키지를 설치

```
$ pip(3) install numpy
```

가상환경이란?

가상환경의 필요성

파이썬을 사용하다 보면 pip으로 패키지를 설치하게 되는데 이 패키지들은 파이썬 설치 폴더의 Lib/site-packages 안에 저장됩니다.

-> 컴퓨터에 설치된다.(전역)

numpy 1.0

프로젝트 A

numpy 1.0

프로젝트 B

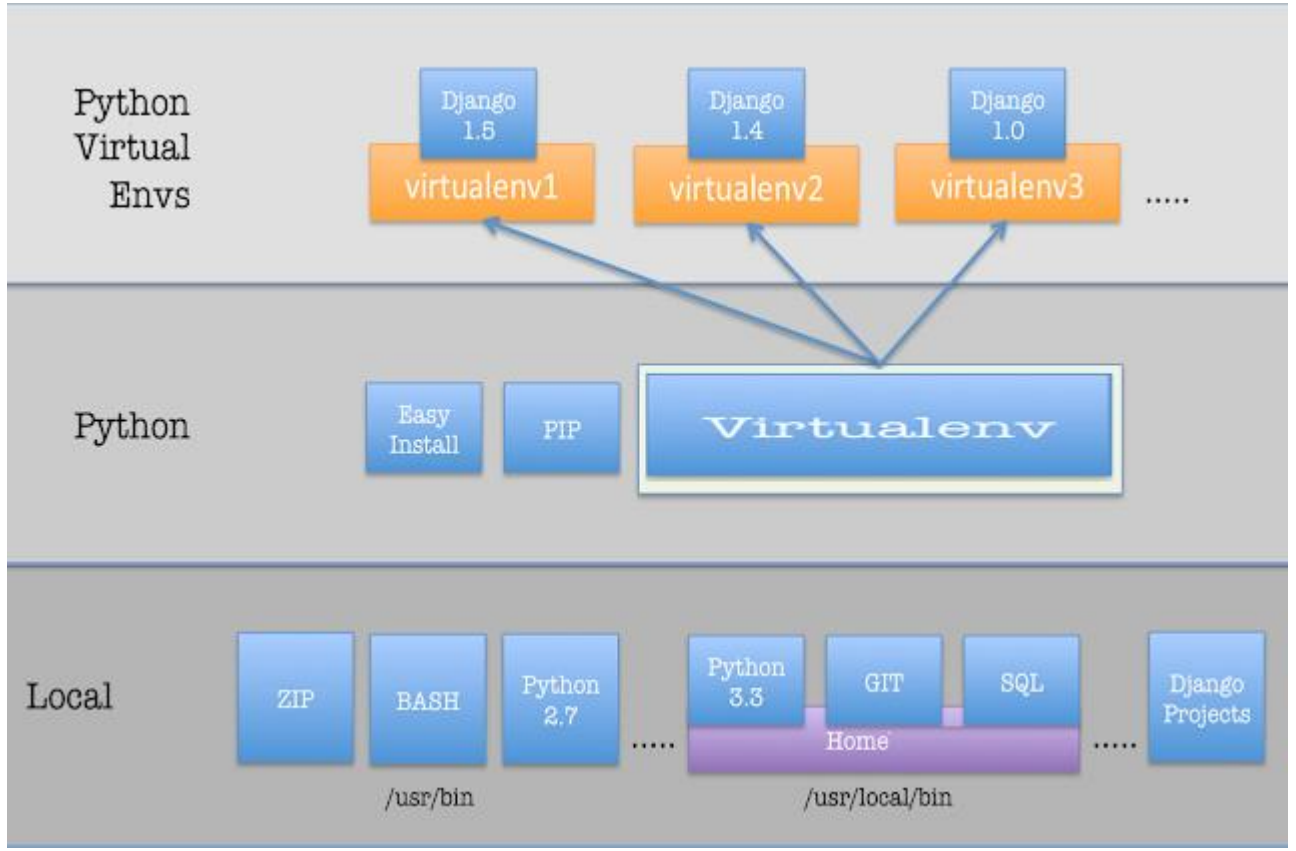
numpy 1.5

가상환경이란?

가상환경

- 독립된 공간 -> 프로젝트별로 개발환경 구축

- 각각의 가상환경은 독립적이어서 서로 다른 가상환경에 설치된 모듈들의 영향을 받지 않는다



가상환경이란?

가상환경 만들어보기

(파이썬의)가상환경

- Venv - Python 3.3 버전 이후 부터 기본모듈에 포함됨
- Virtualenv - Python 2 버전부터 사용해오던 가상환경 라이브러리, Python 3에서도 사용가능
- Conda
- Pyenv
- Pipenv

가상환경이란?

Pipenv?

pipenv

pip + virtualenv

패키지 관리 툴

가상환경

가상환경이란?

Pipenv?

pipenv

장점

- pip와 virtualenv를 따로 쓸 필요가 없다.
- 패키지 관리를 자동으로 해준다.
- 입문자가 사용하기에 쉽다

단점

- 업데이트가 느리다
- 패키지 이름을 오타 냈을 때 깔린다!(??)

가상환경이란?

Pipenv?

pipenv

처음에는 pipenv로 시작

나중에 virtualenv나 pyenv로 바꾸길 추천!

현재 상태 확인

python, pip 버전 확인

Windows

```
$ python --version //3.7.x
$ pip --version //18.x
```

Mac

```
$ python3 --version //3.7.x
$ pip3 --version //18.x
```

Pipenv 설치(컴퓨터에 전역으로!)

Windows

```
$ pip install pipenv
```

Mac

```
$ pip3 install pipenv
```

Pipenv 설치 확인

Windows, Mac

```
$ pipenv --version
```

가상환경 시작하기

pipenv shell

가상환경 만들기

Windows, Mac

$ pipenv shell

반드시 프로젝트 최상위디렉터리에 위치한 상태로 가상환경을 생성해야 한다 !!!!!

1. 해당 폴더에 가상환경이 없다면 가상환경 생성 후 실행
2. 가상환경이 있다면 가상환경 실행

pipenv 실습

1. session5 들어가기

2. 프로젝트 최상위인지 확인

3. pipenv shell

반드시 프로젝트 최상위 디렉터리에 위치한 상태로
가상환경을 생성해야 한다 !!!!!

가상환경 시작하기

pipenv shell

```
cc665@DESKTOP-FADPT9M MINGW64 ~/OneDrive/바 탕  화 면
$ mkdir session5

cc665@DESKTOP-FADPT9M MINGW64 ~/OneDrive/바 탕  화 면
$ cd session5/

cc665@DESKTOP-FADPT9M MINGW64 ~/OneDrive/바 탕  화 면 /session5
$ pipenv shell
Creating a virtualenv for this project▒
Pipfile: C:\Users\cc665\OneDrive\▒▒▒▒ ▒▒\session5\Pipfile
Using c:\users\cc665\appdata\local\programs\python\python37-32\python.exe (3.7.2
) to create virtualenv▒
[====] Creating virtual environment...Already using interpreter c:\users\cc665\a
ppdata\local\programs\python\python37-32\python.exe
Using base prefix 'c:\\users\\cc665\\appdata\\local\\programs\\python\\python37-
32'
New python executable in C:\Users\cc665\.virtualenvs\session5-t_OVEHPs\Scripts\p
ython.exe
Installing setuptools, pip, wheel...
done.

Successfully created virtual environment!
Virtualenv location: C:\Users\cc665\.virtualenvs\session5-t_OVEHPs
Creating a Pipfile for this project▒
Launching subshell in virtual environment▒
```

pipenv 끄기

Windows, Mac

$ exit

control + D

(Pipenv안에) 패키지 설치

$ pipenv install 패키지명

(Pipenv안에) 패키지 제거

$ pipenv uninstall 패키지명

Django 설치

$ pipenv install django==3.0.5

```
cc665@DESKTOP-FADPT9M MINGW64 ~/OneDrive/바 탕  화 면 /session5
$ pipenv install django==3.0.5
Installing django==3.0.5
Adding django to Pipfile's [packages]
Installation Succeeded
Pipfile.lock not found, creating
Locking [dev-packages] dependencies
Locking [packages] dependencies
[ ===] Locking..Success!
Updated Pipfile.lock (9bb2fd)!
Installing dependencies from Pipfile.lock (9bb2fd)
```

pipfile

Pipenv 가상환경이 참고해야할 설정 파일(human readable)

- 설치해야할 패키지, 명령어

pipfile.lock

- 완전히 똑같은 패키지를 설치하기 위한 파일(computer readable)

가상환경

pipfile

$ pipenv install

pipfile, pipfile.lock에 적혀 있는 모든 패키지 설치

- 누군가 프로젝트 세팅을 해서 프로젝트를 올려놓으면 받아서

$ pipenv install

하면 된다!

마무으리 과제

1. 만들어 놓은 FourCal 클래스에 덧셈, 뺄셈, 곱셈, 나눗셈을 연산할 때 마다 어떠한 연산을 몇 번 수행했는지 저장. 연산 횟수 출력하는 메서드 추가

```
cal.ShowCount()
```

```
덧셈 : 1
뺄셈 : 1
곱셈 : 2
나눗셈 : 0
```

마무으리 과제

2.

1) session6 폴더를 만들어서 pipenv 깔기
2) 모듈 설치
   - bs4
   - requests

마무으리

과제

4/23일 세션 전까지 1번 과제 코드 깃헙에 올리기

2번은 양심에^^

(페이스북 모바일 클론 코딩 잘하고 계시죠..?)