파이썬은 연산자 중복이 됨. -?????????????????

클래스 첫글자는 관례상 대문자로 시작

기본적으로 클래스 내에 포함 되는 함수 – 메서드,

생성자, 소멸자, 클래스 매서드가 있음 \_\_init\_\_ , \_\_del\_\_

그리고 매서드는 self가 반드시 들어가야함

해당 클래스 소속 매서드

인스턴스 생성 시 클래스 명과 괄호 꼭 써줘야함

클래스 명으로 호출하는걸 언바운드 매서드 호출, 인스턴스 명으로 호출하는걸 바운드 매서드 콜

언바운드 매서드 콜 할 때는 self를 써줘야함, self 값이 자동 전달 되지 않음 인스턴스 정보를 무조건 첫번째 인자를 전달 해줘야함

생성자, 소멸자는 상속이 아닌 경우 명시적으로 호출 안함

Instance 변수는 통상 생성자에서 정의. 초기화

소멸자 - 파이썬에 메모리 자동 관리 기능이 있어서 소멸자는 잘 쓰지 않음

객체간에 message를 전달해서 작동하게끔 하는게 객체지향 프로그래밍 Instance.Method()

함수를 호출하는거랑 상수 처리랑 속도 차이가 있음. SUBJECT=3 , len(subject)

class: Object를 생성하기 위한 template. Field(변수)+method(함수) / Attribtue(속성)+Behavior(행위) 속성, 변수는 Class(Object)의 상태, 함수, 행위는 class(object)의 상태 변화

Instance: class에 의해 Memory에 생성된 Object

클래스 멤버(변수)와 인스턴스 맴버(변수)는

s=Student()

student class의 instance를 생성해서 reference를 s에 저장

student.name=’hong’

파이썬은 데이터가 치환 될 때 데이터가 만들어짐

클래스 멤버도 self 붙여줘야함

다른 두 인스턴스끼리 클래스멤버는 공유하짐나 인스턴스 멤버는 공유 안함

>>> v1.c\_mem=200

>>> print(v2.g())

(1000, 100)

>>> v1.c\_mem

200

>>> v1.c\_mem

200

>>> v2.c\_mem

100

>>> Var.c\_mem

100

v1(instance)안에 새로 c\_mem이 생성됨

>>> Var.c\_mem=50

>>> print(v2.g())

(1000, 50)

>>> print(v1.g())

(500, 200)

클래스 멤버에 접근 할 때는 클래스명으로 접근하는게 정확함

슬롯에다가 변수명을 나열하면 그 변수명 외에는 추가가 안됨

클래스에 정의한 이름 이외의 속성을 만들 수 없게 하려고

class 구조

-black box 구조: 캡슐화(encapsulation):관련 있는 내용 묶기

정보은닉(Information hiding):정보를 외부로부터 숨기기

-white box 구조

black box 구조는 java?

white box는? 파이썬?????????

필드 추가를 막겠다

역할에 따른 class 구분

-Data 자체를 표현 하는 class: entity class 문제영역에서 관리되어야 하는 데이터를 표현하는 클래스

-class 관리 역할을 하는 class: control class class가? Instance가? class가 여러 개 일 때 관리하는 역할. 리스트 관리 entity class에서 생성하면 control class에서 관리

-사용자와의 Interface를 위한 class: Boundary(view) class

변수명 앞에+는 외부에서 접근가능 하다는 의미

class Calculator:#Entitiy

def \_\_init\_\_( self ):

self.result = 0

self.op = { 1:'+', 2:'-', 3:'\*', 4:'/' }

def calculate( self, number1, number2, op\_select ):

self.result = eval( number1 + self.op[op\_select] + number2 )

return self.result

class ControlCalculator:

def \_\_init\_\_(self):

self.calculator=Calculator()

def calculate(self,number1,number2,op\_select):

self.result=self.calculator.calculate(number1,number2,op\_select)

return self.result

controlcalculator가 calculator한개 생성 1:1관계.

클래스 생성 원칙

단일 책임의 원칙 – 하나의 클래스는 책임 하나만