클래스

클래스를 이용해 프로그래밍하면 데이터와 데이터를 조작하는 함수를 하나의 묶음으로 관리할 수 있으므로 복잡한 프로그램도 더욱 쉽게 작성할 수 있다.
파이썬에서 함수를 정의할 때 def 라는 키워드를 썼던 것처럼 파이썬에서 클래스를 정의하려면 class 라는 키워드를 사용. 클래스를 사용하는 목적이 변수와 함수를 묶어서 하나의 새로운 객체(타입)으로 만드는 것이기 때문에 당연히 클래스에 변수나 함수를 포함시켜 클래스를 정의할 수 있다.

클래스 호출

```
>>> card1 = BusinessCard()
```

클래스에 메서드 추가하기

```
class BusinessCard:
    # 클래스 내부에서 함수를 생성할때는 첫번째는 반드시 self
    def set_info(self, name, email, addr):
        self.name = name
        self.email = email
        self.addr = addr
```

클래스 인스턴스를 통한 메서드 호출

```
>>> member1.set_info("Yuna Kim", "yunakim@naver.com", "Seoul")
>>> print(member1.name)
>>> print(member1.email)
>>> print(member1.addr)

Yuna Kim
yunakim@naver.com
Seoul
```

```
class BusinessCard:

    def __init__(self, name, email, addr):
        self.name = name
```

```
        self.email = email
        self.addr = addr

    def print_info(self):
        print("-----------------------")
        print("Name: ", self.name)
        print("E-mail: ",self.email)
        print("Address: ", self.addr)
        print("-----------------------")

========================
>>> member1 = BusinessCard("Kangsan Lee", "kangsan@naver.com", "USA")
>>> member1.print_info()
-----------------------
Name:  Kangsan Lee
E-mail:  kangsan@naver.com
Address:  USA
-----------------------
```

self 이해하기

Foo 클래스의 func2 메서드는 메서드의 인자가 self뿐이므로 실제 메서드를 호출할때는 인자를 전달할 필요가 없다.
func2 메서드의 첫번째 인자는 self지만 호출할때는 아무것도 전달하지 않는 이유는 첫번째 인자인 self에 대한 값은 파이썬이 자동으로 넘겨주기 때문.

```
class Foo:
    def func1():
        print("function 1")

    def func2(self):
        print("function 2")
============================
>>> f = Foo()
>>> f.func1()   #에러 발생
Traceback (most recent call last):
  File "<pyshell#49>", line 1, in <module>
    f.func1()
TypeError: func1() takes 0 positional arguments but 1 was given

>>> f.func2()   # 에러 발생 안함
function 2
```

self를 좀더 자세히 알아보자
id라는 함수를 이용해 Foo의 func2() 가 어느 메모리에 저장되어있는지 확인

```
class Foo:
def func1():
    print("function 1")

def func2(self):
    print(id(self))
    print("function 2")

=========================
>>> f=Foo()
>>> print("f=Foo() => ", id(f))
f=Foo() =>  54436600

>>> f.func2()
54436600
function 2

>>> f2 = Foo()
>>> print("f2 = Foo() => ", id(f2))
f2 = Foo() =>  54715368
>>> f2.func2()
54715368
function 2
```

```
>>> Foo.func1()     # 객체 생성 없이 클래스 자체를 접근하면 오류가 없다.
function 1
>>> Foo.func2()     # self 위치에 인자를 전달하지 않고 메서드를 호출하면 오류가 빌
Traceback (most recent call last):
  File "<pyshell#58>", line 1, in <module>
    Foo.func2()
TypeError: func2() missing 1 required positional argument: 'self'
```

func2 메서드는 인자를 하나 필요로 하며, 해당 인자는 인스턴스여야 한다.
현재 f3은 새로 생성한 인스턴스를 바인딩하고 있으므로 func2 메서드의 인자로 f3을 전달해주면 된다.

```
>>> f3 = Foo()  # 새로운 객체 생성
>>> print("f3 = Foo() => ",id(f3))
```

```
f3 = Foo() =>  54715704
>>> Foo.func2(f3)   # 객체를 self 매개변수로 전달
54715704
function 2
>>> f3.func2()  # 오류발생 안함
54715704
function 2
```