1. Encapsulation 의미

- Encapsulation refers to the safe storage of data in an instance.
- Data should be accessed only through instance method
- Data should be safe from changes by external method

-> 하지만 Python에서는 User에게 Encapsulation을 따르도록 강제하지 않는다.
-> Python은 User가 올바른 행동을 할 것이라고 기대한다. (pythonic)

2. @property decorator
-> @ property는 encapsulation을 하도록 도움을 주지만 사람들에게 특정 행동을 하도록 강제하지는 않는다.

ex)

```
class GetSet(object):
    def __init__(self, value):
        self.attrval = value

    @property
    def var(self):
        print(“getting the var attribute”)
        return self.attrval

    @var.setter
    def var(self, value):
        print(“setting the var attribute)
        self.attrval = value
```

-> 실제는 attrval 사용자는 var로 접근한다.
-> 사용자가 getter, method로 접근할 필요 없이 변수 이름으로 접근 가능하도록 한다.

3. Variable Naming

- public attributes or variables: regular_lower_case
- private attributes or variables: _single_leading_underscore (internal use in a module or class)
- private attributes shouldn’t be subclassed: __double_leading_underscore

ex)

```
class GetSet(object):
    instance_count = 0
    __mangled_name = ‘no privacy’

    @property
    def var(self):
        print(“Getting the var attribute)
        return self._attrval    # class 내부에서만 사용하지만 force 하지는 않는다.
```

__mangled_name은 외부에서 접근 불가하다.
-> 접근하려면

```
me = GetSet(5)
me._GetSet__mangled_name    # 이렇게만 접근 가능하다.
```