Декоратор @property

Раньше в питоне не было записи декораторов через @ и все создавали свойства через функцию-декоратор `property`

После добавления записи через @ стали записывать свойства так:

```
class Cat:
    def __init__(self, name):
        self._name = name                               # имя кошки

    @property
    def name(self):                                     # геттер свойства name
        return self._name

    @name.setter
    def name(self, name):                               # сеттер свойства name
        if isinstance(name, str) and name.isalpha():
            self._name = name
        else:
            raise ValueError('Некорректное имя')

    @name.deleter
    def name(self):                                     # делитер свойства name
        del self._name
```

Изначально мы создаем свойство `name` как геттер, доступное только для чтения

Эквивалент кода выше

```
class Cat:
    def __init__(self, name):
        self._name = name

    def get_name(self):
        return self._name

    name = property(get_name)        # свойство, имеющее только геттер

    def set_name(self, name):
        if isinstance(name, str) and name.isalpha():
            self._name = name
        else:
            raise ValueError('Некорректное имя')

    name = name.setter(set_name)     # свойство, имеющее геттер предыдущего свойства, а также сеттер

    def del_name(self):
        del self._name

    name = name.deleter(del_name)    # свойство, имеющее геттер и сеттер предыдущего свойства, а также делитер
```