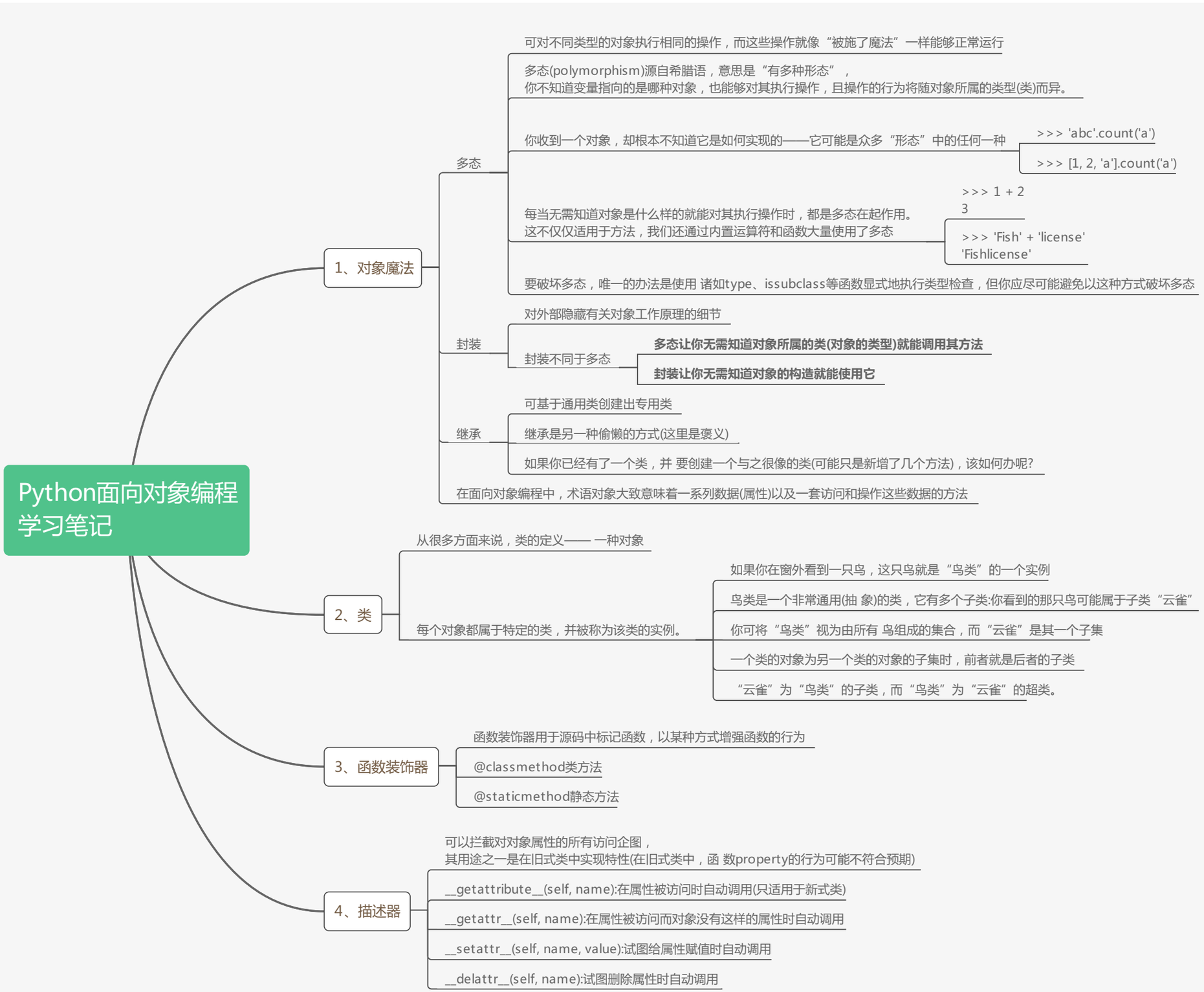
Python面向对象编程学习笔记
1、对象魔法
多态
可对不同类型的对象执行相同的操作，而这些操作就像“被施了魔法”一样能够正常运行
多态(polymorphism)源自希腊语，意思是“有多种形态”，
你不知道变量指向的是哪种对象，也能够对其执行操作，且操作的行为将随对象所属的类型(类)而异。
你收到一个对象，却根本不知道它是如何实现的——它可能是众多“形态”中的任何一种
>>> 'abc'.count('a')
>>> [1, 2, 'a'].count('a')
每当无需知道对象是什么样的就能对其执行操作时，都是多态在起作用。
这不仅仅适用于方法，我们还通过内置运算符和函数大量使用了多态
>>> 1 + 2
3
>>> 'Fish' + 'license'
'Fishlicense'
要破坏多态，唯一的办法是使用 诸如type、issubclass等函数显式地执行类型检查，但你应尽可能避免以这种方式破坏多态
封装
对外部隐藏有关对象工作原理的细节
封装不同于多态
多态让你无需知道对象所属的类(对象的类型)就能调用其方法
封装让你无需知道对象的构造就能使用它
继承
可基于通用类创建出专用类
继承是另一种偷懒的方式(这里是褒义)
如果你已经有了一个类，并 要创建一个与之很像的类(可能只是新增了几个方法)，该如何办呢?
在面向对象编程中，术语对象大致意味着一系列数据(属性)以及一套访问和操作这些数据的方法
2、类
从很多方面来说，类的定义—— 一种对象
每个对象都属于特定的类，并被称为该类的实例。
如果你在窗外看到一只鸟，这只鸟就是“鸟类”的一个实例
鸟类是一个非常通用(抽 象)的类，它有多个子类:你看到的那只鸟可能属于子类“云雀”
你可将“鸟类”视为由所有 鸟组成的集合，而“云雀”是其一个子集
一个类的对象为另一个类的对象的子集时，前者就是后者的子类
“云雀”为“鸟类”的子类，而“鸟类”为“云雀”的超类。
3、函数装饰器
函数装饰器用于源码中标记函数，以某种方式增强函数的行为
@classmethod类方法
@staticmethod静态方法
4、描述器
可以拦截对对象属性的所有访问企图，
其用途之一是在旧式类中实现特性(在旧式类中，函 数property的行为可能不符合预期)
__getattribute__(self, name):在属性被访问时自动调用(只适用于新式类)
__getattr__(self, name):在属性被访问而对象没有这样的属性时自动调用
__setattr__(self, name, value):试图给属性赋值时自动调用
__delattr__(self, name):试图删除属性时自动调用