面向对象：利用封装、继承和多态特性，对物件进行描述和对象间的相互关联，对象为基本单位。

类：定义了数据的形式和对数据的操作，是对事物的抽象描述

对象：类的实例

面向对象的特性：

封装：，使用者只需要知道实现了哪些功能，而不需要知道实现的细节

继承：子类通过对父类的继承，只需要实现子类区别于父类的功能

多态：同一个类的不同子类，具有不同的实现功能

常规内部属性/方法：

\_\_slots\_\_：属性的字符串列表，限定类可使用的属性

\_\_init\_\_(self)：创建对象时，进行初始化操作

\*\*self关键字：在类中，通过self调用属性和方法

方法修饰器：

@property：getter，类的属性引用获取

@ATTR.setter：setter。类的属性引用赋值

@staticmethod：静态方法，使用类进行调用的方法

@abstractmethod：抽象方法，无方法实体，用于组成抽象类(不能实例化)。须使用abc模块的ABCMeta/abstractmethod

类与类的三种关系：

1.is-a：继承关系，一个类继承自另一个类

2.has-a：关联关系，一个类的属性为另一个类

3.use-a：依赖关系，一个类的方法使用另一个类作为参数

命名空间(namespace)创建顺序：

1.built-in名称 -- 解释器启动

2.全局变量 -- 模块导入

3.类变量 -- 实例化类

3.局部变量 -- 方法调用

\*\*built-in名称变量是在builtins模块中

\*\*变量引用顺序为创建顺序的反序

\*\*使用for循环和next(iterator) built-in函数，相当于调用generator的\_\_next\_\_()

generator常用函数：

\_\_next\_\_()：generator的下一个item

close()：关闭generator引用

类支持：属性引用和实例化

属性引用：通过类名使用类中的属性(如:变量/方法)

\*\*\_\_doc\_\_变量可用于引用docstring

实例化：以类为模板，创建出实例

\*\*实例会调用包含响应参数的\_\_init\_\_方法

实例包含的属性：变量和方法

同一类的不同实例，不共享变量。但是当变量为mutable类型时，实质上是引用同一个内容空间的

类语法：

class ClassName(BaseClasssName):

statement1

statement2

多点继承：

class ClassName(Base1,Base2,Base3):

statement1

statement2

抽象类定义：

from abc import ABCMeta,abstractmethod

class Abstract\_class(metaclass=ABCMeta):

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

self.\_name = name

#抽象方法，月薪

@abstractmethod

def get\_salary(self):

pass

继承类属性查找规则：

1.从子类查找

2.从父类查找

3.向更上层父类递归查找

python中没有private属性概念，通常使用下划线(\_\_)做前缀的属性为类内部使用属性。

内部方法使用规则：\_\_spam在python语言中，被包装为\_classname\_\_spam，并且使用该包装才能引用该属性

内部属性使用规则：如果使用\_\_开头，子类将不可用该属性，所以以\_为开头表明应注意为内部使用属性，但可正常使用

built-in方法：

isinstance(obj,type)：判断实例obj是否为type类或其子类

issubclass(type1,type2)：判断type1类是否为type2的子类