浮动主题

主题

- 函数
  - 1-函数的定义
    - ·函数以def关键词开头，后接函数名和圆括号()。
      ·函数执行的代码以冒号起始，并且缩进。
      ·return [表达式] 结束函数，选择性地返回一个值给调用方。不带表达式的return相当于返回None。
    - 结构：
      def functionname (parameters):
      "函数文档字符串"
      functionsuite
      return [expression]
  - 2-函数的调用
  - 3-函数文档
  - 4-函数参数
    - 1，位置参数 (positional argument)
      - 结构：
        def functionname(arg1):
        "函数文档字符串"
        functionsuite
        return [expression]
      - arg1 - 位置参数 ，这些参数在调用函数 (call function) 时位置要固定。
    - 2，默认参数 (default argument)
      - 结构：
        def functionname(arg1, arg2=v):
        "函数文档字符串"
        functionsuite
        return [expression]
      - ·arg2 = v - 默认参数 = 默认值，调用函数时，默认参数的值如果没有传入，则被认为是默认值。
        ·默认参数一定要放在位置参数 后面，不然程序会报错。
        ·Python 允许函数调用时参数的顺序与声明时不一致，因为 Python 解释器能够用参数名匹配参数值。
    - 3，可变参数 (variable argument)
      - 结构：
        def functionname(arg1, arg2=v, *args):
        "函数文档字符串"
        functionsuite
        return [expression]
      - ·*args - 可变参数，可以是从零个到任意个，自动组装成元组。
        ·加了星号（*）的变量名会存放所有未命名的变量参数。
    - 4，关键字参数 (keyword argument)
      - 结构：
        def functionname(arg1, arg2=v, *args, **kw):
        "函数文档字符串"
        functionsuite
        return [expression]
      - ·**kw - 关键字参数，可以是从零个到任意个，自动组装成字典。
        可变参数允许传入零个到任意个参数，它们在函数调用时自动组装为一个元组 (tuple)。
        关键字参数允许传入零个到任意个参数，它们在函数内部自动组装为一个字典 (dict)。
    - 5，命名关键字参数 (name keyword argument)
      - 结构：
        def functionname(arg1, arg2=v, args, *, nkw, *kw):
        "函数文档字符串"
        functionsuite
        return [expression]
      - ·*, nkw - 命名关键字参数，用户想要输入的关键字参数，定义方式是在nkw 前面加个分隔符 *。
        ·如果要限制关键字参数的名字，就可以用「命名关键字参数」
        ·使用命名关键字参数时，要特别注意不能缺少参数名。
    - 6，参数组合
      - 在 Python 中定义函数，可以用位置参数、默认参数、可变参数、命名关键字参数和关键字参数，这 5 种参数中的 4 个都可以一起使用，但是注意，参数定义的顺序必须是：

        ·位置参数、默认参数、可变参数和关键字参数。
        ·位置参数、默认参数、命名关键字参数和关键字参数。
        要注意定义可变参数和关键字参数的语法：

        ·*args 是可变参数，args 接收的是一个 tuple
        ·**kw 是关键字参数，kw 接收的是一个 dict
        命名关键字参数是为了限制调用者可以传入的参数名，同时可以提供默认值。定义命名关键字参数不要忘了写分隔符 *，否则定义的是位置参数。
      - 警告：虽然可以组合多达 5 种参数，但不要同时使用太多的组合，否则函数很难懂。
  - 5-函数的返回值
  - 6-变量的作用域
    - ·Python 中，程序的变量并不是在哪个位置都可以访问的，访问权限决定于这个变量是在哪里赋值的。
      ·定义在函数内部的变量拥有局部作用域，该变量称为局部变量。
      ·定义在函数外部的变量拥有全局作用域，该变量称为全局变量。
      ·局部变量只能在其被声明的函数内部访问，而全局变量可以在整个程序范围内访问。
    - 内嵌函数
      - ·是函数式编程的一个重要的语法结构，是一种特殊的内嵌函数。
        ·如果在一个内部函数里对外层非全局作用域的变量进行引用，那么内部函数就被认为是闭包。
    - 闭包
      - ·通过闭包可以访问外层非全局作用域的变量，这个作用域称为 闭包作用域。
    - 递归
      - 如果一个函数在内部调用自身本身，这个函数就是递归函数。

- 主题
- 3-从函数到方法（1）
- Lambda表达式
  - 1-匿名函数的定义
    - 在 Python 里有两类函数：

      ·第一类：用 def 关键词定义的正规函数
      ·第二类：用 lambda 关键词定义的匿名函数
    - 结构：
      lambda argument_list: expression
    - 特点：
      ·lambda - 定义匿名函数的关键词。
      ·argument_list - 函数参数，它们可以是位置参数、默认参数、关键字参数，和正规函数里的参数类型一样。
      ·:- 冒号，在函数参数和表达式中间要加个冒号。
      ·expression - 只是一个表达式，输入函数参数，输出一些值。
      注意：

      ·expression 中没有 return 语句，因为 lambda 不需要它来返回，表达式本身结果就是返回值。
      ·匿名函数拥有自己的命名空间，且不能访问自己参数列表之外或全局命名空间里的参数。
  - 2-匿名函数的应用
    - ·非函数式编程
      ·函数式编程
    - 匿名函数 常常应用于函数式编程的高阶函数 (high-order function)中，主要有两种形式：

      ·参数是函数 (filter, map)
      ·返回值是函数 (closure)
      - ·filter(function, iterable) 过滤序列，过滤掉不符合条件的元素，返回一个迭代器对象
        ·map(function, *iterables) 根据提供的函数对指定序列做映射。
- 类与对象
  - 1-对象=属性+方法
    - 对象是类的实例。换句话说，类主要定义对象的结构，然后我们以类为模板创建对象。
      类不但包含方法定义，而且还包含所有实例共享的数据
    - ·封装
      ·继承
      ·多态
  - 2-self是什么?
    - 类的方法与普通的函数只有一个特别的区别 —— 它们必须有一个额外的第一个参数名称（对应于该实例，即该对象本身），按照惯例它的名称是 self。在调用方法时，我们无需明确提供与参数 self 相对应的参数。
  - 3-__init__构造方法
    - __init__(self[, param1, param2...])的魔法方法，该方法在类实例化时会自动调用
  - 4-公有和私有
    - 在 Python 中定义私有变量只需要在变量名或函数名前加上“__”两个下划线，那么这个函数或变量就会为私有的了
  - 5-继承
    - 结构：
      class DerivedClassName(modname.BaseClassName):
      statement-1
      .
      .
      .
      statement-N
    - 一个简单例子：如果子类中定义与父类同名的方法或属性，则会自动覆盖父类对应的方法或属性
    - 调用未绑定的父类方法
    - 使用super函数
    - 多继承（不建议，容易乱）
      - 结构：
        class DerivedClassName(Base1, Base2, Base3):
        statement-1
        .
        .
        .
        statement-N
  - 6-组合
    - 类对象：创建一个类，其实也是一个对象也在内存开辟了一块空间，称为类对象，类对象只有一个。

      class A(object):
      pass

      实例对象：就是通过实例化类创建的对象，称为实例对象，实例对象可以有多个。
  - 7-类、类对象和实例对象
    - 类属性与实例属性
    - 注意：属性与方法名相同，属性会覆盖方法
  - 8-什么是绑定
    - Python 严格要求方法需要有实例才能被调用，这种限制其实就是 Python 所谓的绑定概念。

      Python 对象的数据属性通常存储在名为.__ dict__的字典中，我们可以直接访问__dict__，或利用 Python 的内置函数vars()获取.__ dict__。
  - 9-一些相关的内置函数（BIF）
    - ·issubclass(class, classinfo) 方法用于判断参数 class 是否是类型参数 classinfo 的子类。
      ·一个类被认为是其自身的子类。
      ·classinfo可以是类对象的元组，只要class是其中任何一个候选类的子类，则返回True。
    - ·isinstance(object, classinfo) 方法用于判断一个对象是否是一个已知的类型，类似type()。
      ·type()不会认为子类是一种父类类型，不考虑继承关系。
      ·isinstance()会认为子类是一种父类类型，考虑继承关系。
      如果第一个参数不是对象，则永远返回False。
      如果第二个参数不是类或者由类对象组成的元组，会抛出一个TypeError异常。
    - ·hasattr(object, name)用于判断对象是否包含对应的属性。
    - ·getattr(object, name[, default])用于返回一个对象属性值。
    - ·setattr(object, name, value)对应函数 getattr()，用于设置属性值，该属性不一定是存在的。
    - ·delattr(object, name)用于删除属性。