面向对象编程（面向对象程序设计）是一种计算机编程架构

1. 'object'是所有Python类型的基类型

2.Python中的所有类型（数字，字符串，函数，类，模块等）都是一个对象，每个对象都有一个'类型'。对象变量是指向其在内存中的位置的指针。

3.所有对象都是引用计数

|  |
| --- |
| sys.getrefcount(5)=>x |
| a=5,b=a这将创建对等号右侧对象的引用，因此A和B都指向5 |
| sys.getrefcount(5) => x + 2 |
| del(a); sys.getrefcount(5) => x + 1 |

4.class基本形式

|  |
| --- |
| class MyObject(object):  # 'self'相当于Java / C ++中的'this'  def \_\_init\_\_(self, name):  self.name = name  def memberFunc1(self, arg1):  ..  @staticmethod  def classFunc2(arg1):  ..  obj1=MyObject('name1') obj1.memberFunc1('a') MyObject.classFunc2('b') |

1. 有用的交互命令

dir(variable1) # 列出该对象的所有方法

通用的字符串操作

异常处理

基本形式：

|  |
| --- |
| try:  ..  except ValueError as e:  print e  except (TypeError, AnotherError):  ..  except:  ..  finally:  ..#清理，例如关闭数据库。 |

手动引发异常：

|  |
| --- |
| raise AssertionError #断言失败  raise SystemExit #请求程序退出  raise RuntimeError('Error message : ..') |

列表、集合和字典的压缩

使代码更容易读写的语法糖

1. 列表生成式

通过一个简洁的表达式，过滤集合中的元素，并将通过过滤的元素形成一个新列表。

基本形式：

|  |
| --- |
| [expr for val in collection if condition] |

另一种形式:

|  |
| --- |
| result = []  for val in collection:  if condition:  result.append(expr) |

过滤条件可以省略，只留下表达式。

1. 字典生成式

基本形式：

|  |
| --- |
| {key-expr : value-expr for value in collection if condition} |

1. 集合生成式

基本形式:与列表生成式相同，除了用{}代替[ ]

4.嵌套列表生成式

基本形式：

|  |
| --- |
| [expr for val in collection for innerVal in val if condition] |

|  |  |
| --- | --- |
| 使用Separator连接List/Tuple | ', '.join([ 'v1', 'v2', 'v3']) => 'v1, v2, v3' |
| 格式化字符串 | string1 = 'My name is {0} {name}'  newString1=string1.format('Sean', name = 'Chen') |
| 拆分字符串 | sep = '-'  stringList1 = string1.split(sep) |
| 获取子串 | start = 1  string1[start:8] |
| 使用零填充字符串 | month='5'  month.zfill(2) => '05'  month = '12'  month.zfill(2) => '12' |