函数

Python是**通过引用传递的**，函数参数是通过引用传递的。

•基本形式：

def func1（posArg1，keywordArg1 = 1，..）：

注意：

•关键字参数必须遵循位置参数。

•Python默认情况下不是“惰性求值”，表达式是立即求值的。

•函数调用机制：

1．所有函数都是模块级作用域的本地函数。请参阅“模块”部分。

2．在内部，参数被打包成元组和字典，函数接收元组'args'和字典'kwargs'并在内部

解包。

•'函数是对象'的常见用法：

def func1（ops = [str.strip，user\_ define\_func，..]，..）：

for function in ops：

value = function（value）

**返回值**

•如果在到达函数结束时没有遇到返回语句，则 ***不返回*** 任何值。

•多个值通过一个元组对象返回

return（value1，value2）

value1，value2 = func1（..）

**匿名(又名LAMBDA)函数**

* 什么是匿名函数？

一个由单个语句组成的简单函数。

lambda x ：x \* 2

＃def func1（x）：return x \* 2

•匿名函数的应用:'curring'又名通过局部参数应用从现有函数派生新函数。

ma60 = lambda x ：pd.rolling\_mean（x，60）

**有用函数（用于数据结构**）

1. **Enumerate**返回一个序列（i，value）元组，其中i是当前项的索引。

for i， value in enumerate(collection)：

• 应用程序：创建序列值（假定为唯的一）字典映射到序列中的位置。

1. **Sorted**从任何序列返回新的排序列表

sorted（ [ 2, 1, 3 ] ）=> [ 1, 2, 3]

• 应用程序：

sorted （ set（ 'abc bcd' ） ）=> [' '，'a'，'b'，'c'，'d']

＃返回已排序的唯一字符

1. **Zip**对多个列表、元组或其他序列的元素进行配对，创建一个元组列表:

zip（seq1，seq2）=>

[（'seq1\_1'，'seq2\_1'），（..），..]

•Zip可以采用任意数量的序列。但是，它产生的元素数量由“最短”序列决定。

•应用程序：同时迭代多个序列：

for i，（a，b）in

enumerate（zip（seq1，seq2））：

•Unzip - 另一种思考方法是将行列表转换为列列表。

seq1，seq2 = zip（\* zipOutput）

1. Reversed以相反的顺序迭代序列的元素。

list（reverse（range（10）））\*

\* reversed（）返回迭代器，list（）使其成为一个列表。

控制和流程

1. 'if else'条件的操作符：

检查两个变量是否相同的对象

var1 is var2

检查两个变量是否是不同的对象

var1 is not var2

检查两个变量是否具有相同的值

var1 == var2

**警告**：对复合条件使用'and'，'or'，'not'运算符，而不是&&，||，！。

1. 'for'运算符的常见用法：

遍历集合(即列表或元组)或迭代器

for element in iterator :

如果元素是序列，可以'解包'

for a, b, c in iterator :

3. 'pass' - 无操作语句。用于不采取任何措施的区块。

4. 三元表达式——即不那么冗长的“if else”

•基本形式：

value = true-expr if condition

else false-expr

1. 没有switch / case语句，请使用if / elif代替