1：input和raw\_input的比较

raw\_input会把所有的输入当作原始数据，然后将其放入字符串中

2：长字符串

>>> print '''This is a very log str

... aaaaaaaaaaa

... bbbbbbbbbb

... cccccccccccc'''

3：如果想要打印出\可以使用[\\，使用反斜杠进行转义‘\](\\\\，使用反斜杠进行转义'\\)’

4：原始字符串以r开头，可以在其中输入任何字符，它都会按原样输出

5：原始字符以反斜杠结尾可使用如下方法输出：

>>> print r'C:\Program Foles\foo\bar' '\\'

向上取整

>>> math.ceil(12.3)

13.0

向下取整

>>> math.floor(12.3)

12.0

二：列表和元组

列表与元组的区别：列表可以修改，元组不能修改

分片：开始点的结果包含在列表中，结束点的结果不包含在列表中，当使用一个负数作为步长时，开始点必需大于结束点，对于一个正数步长，序列会从头开始向右提取;对于一个负数步长，序列会从尾开始向左提取。

# 列表的增删改查操作：（列表是有序和可更改的集合。允许重复的成员）

修改操作：

>>> x=[1,1,1]

>>> x[1]=2

>>> x

[1, 2, 1]

>>> name[2:]=list('ar') =========🡺利用分片来修改，将下标为2以后的元素进行修改

>>> name

['P', 'e', 'a', 'r']

删除操作：

>>> names=['a','b','c','d','e','f']

>>> del names[2] ===========》根据下标删除

>>> names

['a', 'b', 'd', 'e', 'f']

-----------------------------------------------------------------

>>> numbers[1:4]=[] ==========》删除空的切片

>>> numbers

[1, 5]

添加操作：

>>> numbers=[1,5]

>>> numbers[1:1]=[2,3,4] =============🡺在第一个元素之后添加2，3，4

>>> numbers

[1, 2, 3, 4, 5]

append方法：将一个对象添加到列表末尾

>>> lst=[1,2,3]

>>> lst.append(4)

>>> lst

[1, 2, 3, 4]

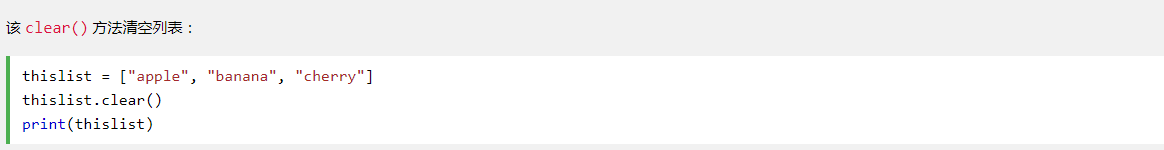
Clear清空列表（此方法只能在python3中使用）

>>> lst=[1,2,3]

>>> lst.clear()

>>> lst

[]



Insert方法

>>> numbers=[1,2,3,4,5,6,7]

>>> numbers.insert(3,'four')

>>> numbers

[1, 2, 3, 'four', 4, 5, 6, 7]

Compare方法

Com(x,y)

x<y：返回负数

x>y：返回正数

x=y：返回零

copy方法：（修改副列表，原始列表保持不变）

>>> a=[1,2,3]

>>> b=a.copy()

>>> b

[1, 2, 3]

>>> b[1]=4

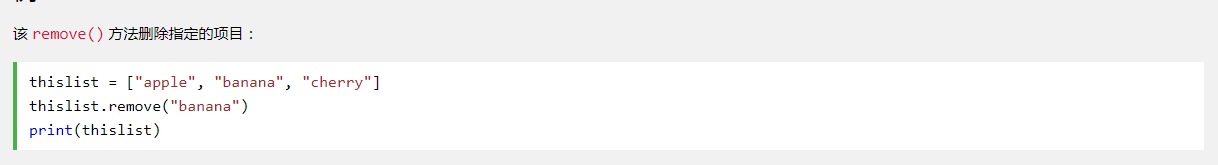
>>> a

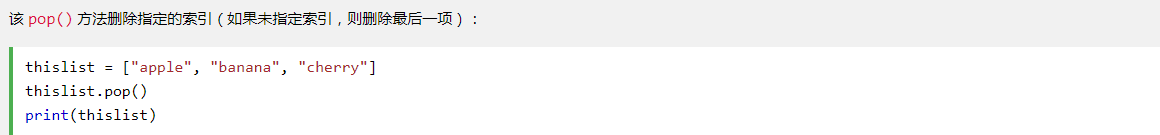
[1, 2, 3]

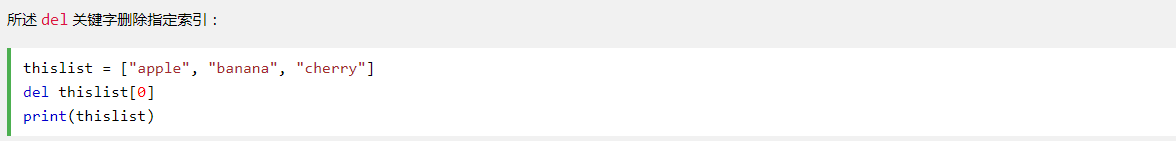
>>> b

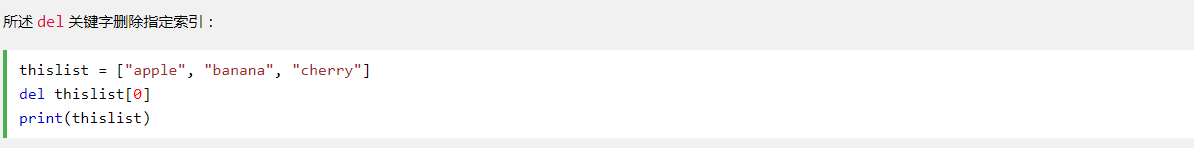
[1, 4, 3]

删除:





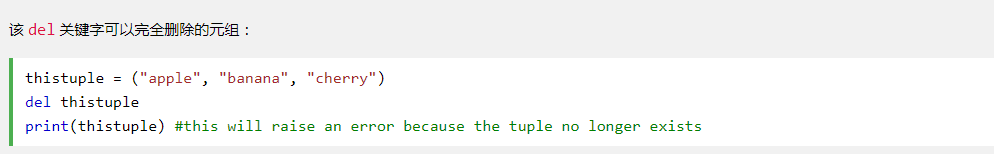




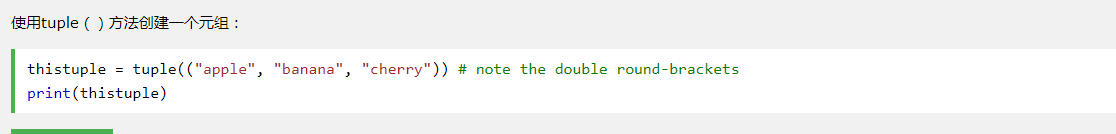
列表的复制：

# 元组（不可修改的序列）

元组的删除操作：（元组是不可**更改的**，因此您无法从中删除项目，但您可以完全删除元组：）



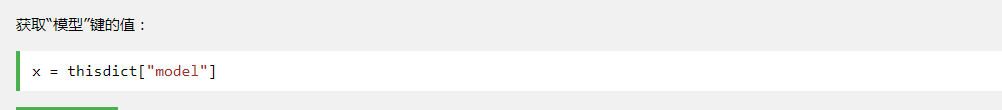
创建：

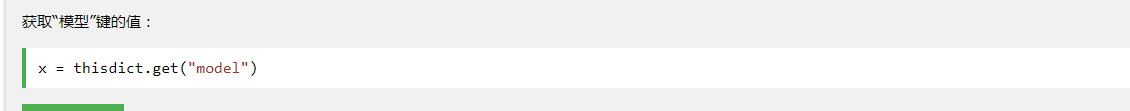


tuple：将一个序列作为参数，将它转换成元组

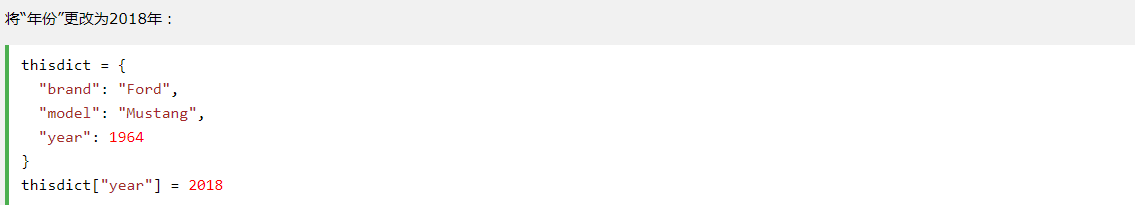
# 字典

字典是一个无序，可变和索引的集合。在Python中，字典用大括号编写，它们具有键和值。





修改：

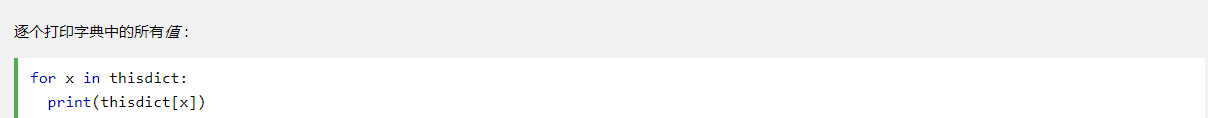


得到字典中的所有键的名称：

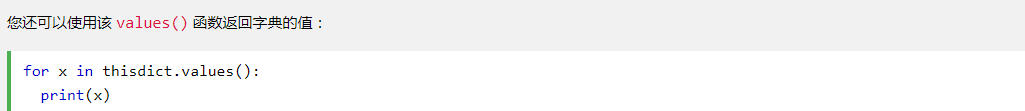


得到字典中的所有值：

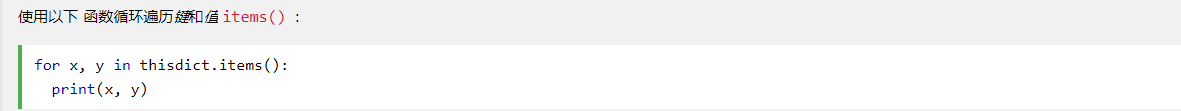
方法一：



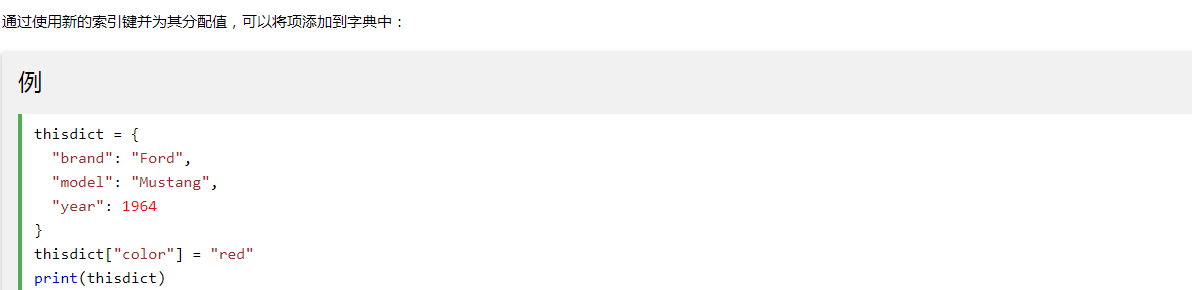
方法二：



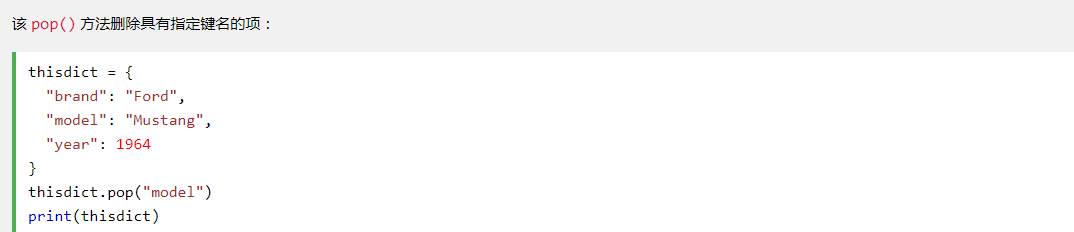
得到键值：

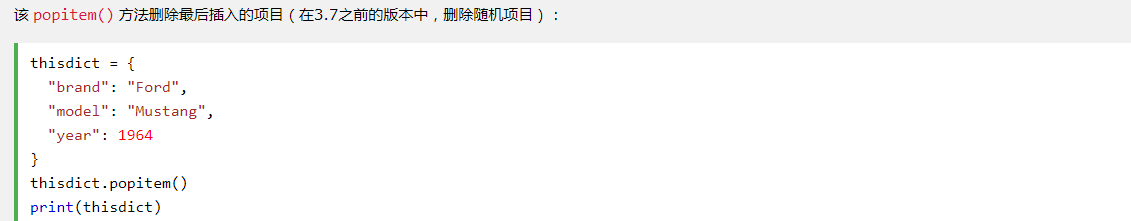


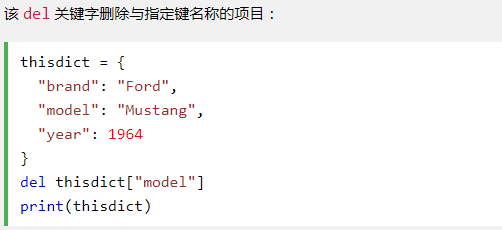
添加：

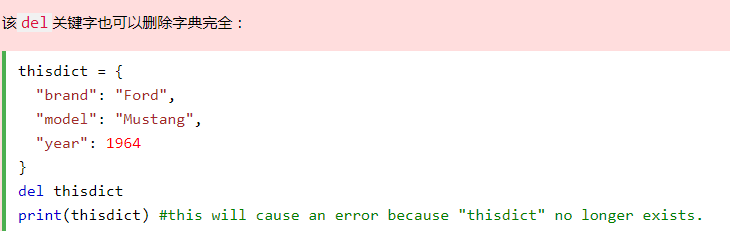


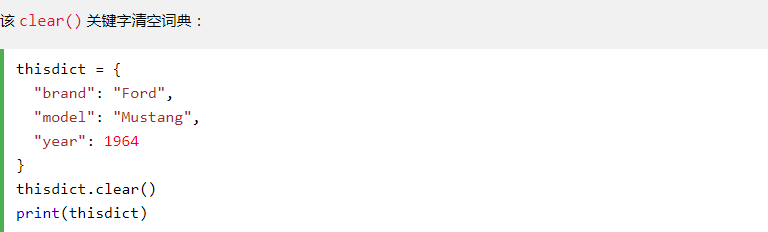
删除：











=================================================================

字段宽度：转换后的值保留的最小字符个数

精度：结果中应该包含的小数位数

>>> '%010.2f' % pi （字符宽度为10，保留2位小数，不够用0填弃）

'0000003.14'

>>> '%-10.2f' % pi （减号用来表示左对齐）

'3.14 '

>>> print('%+5d' % 100)+'\n'+('%+5d' % -100) （加号+用来标示出数字的符号）

+100

-100

将字母转换成标题，使用如下两种方法：

>>> "that's all folks".title()

"That'S All Folks"

>>> import string

>>> string.capwords("that's all,folks")

"That's All,folks"

split：将字符串分割成序列

strip：去除两侧空格（不包括内部）

替换操作：

>>> from string import maketrans

>>> table=maketrans('cs','kz')

>>> '\*\*\*spam\*\*hello\*world\*\*\*!!!'.translate(table,'\*')

'zpamhelloworld!!!'

# 字典

使用==运算符来判定两个对象是否相等，使用is判定两者是否等同（同一个对象）

# 条件循环和其它语句

如果能用for循环，就尽量不用while循环

倒序输出99到1之间的所有数字：

>>> range(99,0,-1)

[99, 98, 97, 96, 95, 94, 93, 92, 91, 90, 89, 88, 87, 86, 85, 84, 83, 82, 81, 80, 79, 78, 77, 76, 75, 74, 73, 72, 71, 70, 69, 68, 67, 66, 65, 64, 63, 62, 61, 60, 59, 58, 57, 56, 55, 54, 53, 52, 51, 50, 49, 48, 47, 46, 45, 44, 43, 42, 41, 40, 39, 38, 37, 36, 35, 34, 33, 32, 31, 30, 29, 28, 27, 26, 25, 24, 23, 22, 21, 20, 19, 18, 17, 16, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1]

实现N的阶乘：

>>> def factorial(n):

... if n==1:

... return 1

... else:

... return n\*factorial(n-1)

...

=================================================================

求一个数的幂次方：

>>> def power(x,n):

... result=1

... for i in range(n):

... result\*=x

... return result

# JOSN

1：loads方法：



2：dumps方法：

