第四章 List和Tuple类型

一、List类型

1.Python创建List

Python内置的一种数据类型是列表：list;

list是一种有序的集合，可以随时添加和删除其中的元素。

比如，列出班里所有同学的名字，就可以用一个list表示：

```
L=['Michael', 'Bob', 'Tracy']
print L
```

list是数学意义上的有序集合，也就是说，list中的元素是按照顺序排列的。

list的构造：直接用 [ ] 把list的所有元素都括起来，就是一个list对象。通常，我们会把list赋值给一个变量，这样，就可以通过变量来引用list：

```
classmates = ['Michael', 'Bob', 'Tracy']
print classmates # 打印classmates变量的内容
#输出结果：['Michael', 'Bob', 'Tracy']
```

由于Python是动态语言，所以list中包含的元素并不要求都必须是同一种数据类型，我们完全可以在list中包含各种数据：

```
L = ['Michael', 100, True]
print L
```

一个元素也没有的list，就是空list：

```
empty_list = []
```

练习：假设班里有3名同学：Adam，Lisa和Bart，他们的成绩分别是95.5，85 和 59，请按照 名字, 分数, 名字, 分数... 的顺序按照分数从高到低用一个list表示，然后打印出来

```
L = ['Adma',95.5,'Lisa',85,'Bart',59]
print L
```

2.Python按照索引访问List

由于list是一个有序集合，所以，我们可以用一个list按分数从高到低表示出班里的3个同学：

```
L = ['Adam', 'Lisa', 'Bart']
print L
```

那我们如何从list中获取指定第 N 名的同学呢？方法是通过索引来获取list中的指定元素。

特别注意：索引从 0 开始，也就是说，第一个元素的索引是0，第二个元素的索引是1，以此类推。

因此，要打印第一名同学的名字，用 L[0]：

```
L = ['Adam', 'Lisa', 'Bart']
print L[0]
```

要打印第二名同学的名字，用 L[1]：

```
L = ['Adam', 'Lisa', 'Bart']
print L[1]
```

要打印第三名同学的名字，用 L[2]：

```
L = ['Adam', 'Lisa', 'Bart']
print L[2]
```

要打印第四名同学的名字，用 L[3]：

```
L = ['Adam', 'Lisa', 'Bart']
print L[3]
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
IndexError: list index out of range
```

报错了！IndexError意思就是索引超出了范围，因为上面的list只有3个元素，有效的索引是 0，1，2。

注意：使用索引时，千万注意不要越界。

练习：三名同学的成绩可以用一个list表示： L = [95.5, 85, 59]请按照索引分别打印出第一名、第二名、第三名，同时测试 print L[3]。

```
L = [95.5,85,59]
print L[0]
print L[1]
print L[2]
```

3.Python之倒序访问List

我们还是用一个list按分数从高到低表示出班里的3个同学：

```
L = ['Adam', 'Lisa', 'Bart']
print L
```

这时，老师说，请分数最低的同学站出来。

要写代码完成这个任务，我们可以先数一数这个 list，发现它包含3个元素，因此，最后一个元素的索引是2：

```
L = ['Adam', 'Lisa', 'Bart']
print L[2]
```

更简单的方法?

Bart同学是最后一名，俗称倒数第一，所以，我们可以用 -1 这个索引来表示最后一个元素：

```
L = ['Adam', 'Lisa', 'Bart']
print L[-1]
```

类似的，倒数第二用 -2 表示，倒数第三用 -3 表示，倒数第四用 -4 表示：

```
L = ['Adam', 'Lisa', 'Bart']
print L[-1]
print L[-2]
print L[-3]
print L[-4]
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
IndexError: list index out of range
```

L[-4] 报错了，因为倒数第四不存在，一共只有3个元素。

使用倒序索引时，也要注意不要越界。

练习：三名同学的成绩可以用一个list表示：L = [95.5, 85, 59] 请按照倒序索引分别打印出倒数第一、倒数第二、倒数第三。

```
L = [95.5, 85, 59]
print L[-1]
print L[-2]
print L[-3]
```

4.Python之添加新元素

```
L = ['Adam', 'Lisa', 'Bart']
print L
```

添加新元素Paul:

(1)append()添加到尾部

append()总是把新的元素添加到list的尾部

```
L = ['Adam', 'Lisa', 'Bart']
L.append('Pual')
print L
#['Adam','Lisa','Bart','Pual']
```

(2)insert()添加元素

insert(),接受两个参数，第一个参数是索引导，第二个参数是待添加的新元素

```
L = ['Adam', 'Lisa', 'Bart']
L.insert(0,'Pual')
print L
#['Pual','Adam','Lisa','Bart']
```

L.insert(0,'Pual')的意思是，'Pual'将被添加到索引为0的位置上（也就是第一个），而原来索引为0的Adma同学，都自动向后移一位。

练习：假设新来的一名学生Pual，Pual同学的成绩比Bart好，但是比Lisa差，他应该排到第三名的位置，请用代码实现。

```
L = ['Adam', 'Lisa', 'Bart']
L.insert(2,'Pual')
print L
#方法一
 #L.insert(2,'Paul')
 #print L
#方法二
 #a=Paul
 #L.insert(a>L[1],a<L[-1])
 #print L
#方法三
 #L = ['Adam','Lisa','Paul','Bart']
 #print L
#方法四
 #L.insert(-2,'Paul')
 #print L
```

5.Python之删除元素

使用pop()删除元素

```
L=['Pual','Adam','Lisa','Bart']
L.pop()
'Pual'
print L
#['Adma','Lisa','Bart']
```

pop()方法总是删掉list的最后一个元素，并且它还返回这个元素，所以我们执行L.pop()后，会打印出'Pual'。

如果Pual在第三位:

必须先定位Pual的位置，由于Paul的索引是2，所以用pop(2)删除:

```
L=['Adam','Lisa','Pual','Bart']
L.pop(2)
'Pual'
print L
#['Adma','Lisa','Bart']
```

练习：L = ['Adam', 'Lisa', 'Paul', 'Bart'] Paul的索引是2，Bart的索引是3，如果我们要把Paul和Bart都删掉，请解释下面的代码为什么不能正确运行：L.pop(2)L.pop(3)怎样调整代码可以把Paul和Bart都正确删除掉?

```
L=['Adam','Lisa','Pual','Bart']
L.pop(2)
L.pop(2)
print L
```

6.Python之替换元素

```
L=['Adam','Lisa','Bart']
```

1）直接用Pual把Bart替换掉

```
L=['Adam','Lisa','Bart']
L[2]='Pual'
print L
```

对list中的某一个索引赋值，就可以直接用新的元素替换掉原来的元素，list包含的元素个数保持不变。

2）Bart还可以用-1做索引，因此，下面的代码也可以完成同样的工作：

```
L=['Adam','Lisa','Bart']
L[-1]='Pual'
print L
```

练习：班里的同学按照分数排名是这样的：L = ['Adam', 'Lisa', 'Bart']但是，在一次考试后，Bart同学意外取得第一，而Adam同学考了倒数第一。请通过对list的索引赋值，生成新的排名。

```
L=['Adam','Lisa','Bart']
a=L[0]
L[0]=L[2]
L[2]=a
print L
```

二、Tuple类型

1.Python之创建tuple

tuple是另一种有序的列表，中文翻译是"元组"。tuple和list非常相似，tuple一旦创建完毕，就不能修改了。

同样是表示班里的同学，用tuple表示如下：

```
t=('Adam','Lisa','Bart')
```

创建tuple和创建list唯一不同的是用（）代替了[]

但是，这个t就不能改变了，tuple没有append()方法，也没有insert()和pop()方法，所以，新的元素没法直接往tuple中添加，元素想退出tuple也不行。

获取tuple元素的方式和list的一模一样，我们可以正常使用t[0],t[-1]等索引方式访问元素，但是不能赋值成别的元素：

```
t=('Adam','Lisa','Bart')
t[0]='Pual'
print t
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
TypeError: 'tuple' object does not support item     assignment
#程序报错
```

练习：创建一个tuple，顺序包含0-9这10个数

```
t=(0,1,2,3,4,5,6,7,8,9)
print t
```

2.Python之创建tuple

tuple和list一样，可以包含0个、1个和任意多个元素。

包含多个元素的 tuple，前面我们已经创建过了。

空tuple:包含0个元素的tuple，直接用()表示

```
t=()
print t
```

创建包含1个元素的 tuple 呢？来试试:

```
t=(1)
print t
```

好像哪里不对！t 不是 tuple ，而是整数1。为什么呢?

因为()既可以表示tuple，又可以作为括号表示运算时的优先级，结果 (1) 被Python解释器计算出结果 1，导致我们得到的不是tuple，而是整数 1。

正是因为用()定义单元素的tuple有歧义，所以 Python 规定，单元素 tuple要多加一个逗号“,”，这样就避免了歧义:

```
t=(1,)
print t
```

Python在打印单元素tuple时，也自动添加了一个“,”，为了更明确地告诉你这是一个tuple。

多元素 tuple 加不加这个额外的“,”效果是一样的:

```
t=(1,2,3,)
print t
#输出结果(1,2,3)
```

练习：请指出右边编辑器中代码为什么没有创建出包含一个学生的 tuple：t = (’Adam’)print t请修改代码，确保 t 是一个tuple。

```
t=('Adam',)
print t
#输出结果('Adam')
```

3.Python之可变的tuple

前面我们看到了tuple一旦创建就不能修改。现在，我们来看一个“可变”的tuple：

```
t=('a','b',['A','B'])
print t)
```

注意到 t 有 3 个元素：'a'，'b'和一个list：['A'，'B']。list作为一个整体是tuple的第3个元素。list对象可以通过 t[2] 拿到：

```
L=t[2]
```

然后，我们把list的两个元素改一改：

```
L[0]='X'
L[1]='Y'
```

再看看tuple的内容：

```
print t
# 输出结果('a','b',['X','Y'])
```

不是说tuple一旦定义后就不可变了吗？怎么现在又变了？

表面上看，tuple的元素确实变了，但其实变的不是 tuple 的元素，而是list的元素。

tuple一开始指向的list并没有改成别的list，所以，tuple所谓的“不变”是说，tuple的每个元素，指向永远不变。即指向'a'，就不能改成指向'b'，指向一个list，就不能改成指向其他对象，但指向的这个list本身是可变的！

理解了“指向不变”后，要创建一个内容也不变的tuple怎么做？那就必须保证tuple的每一个元素本身也不能变。

练习：定义了tuple：t = ('a'，'b'，['A'，'B'])由于 t 包含一个list元素，导致tuple的内容是可变的。能否修改上述代码，让tuple内容不可变？

```
#t=('a','b',['A','B'])
t=('a','b',('A','B'))
print t
```