知识点

1. 创建列表

2. 读取元素

3. 遍历列表

4. 增加和删除列表元素

5. 常用函数

1.列表

Python内置的一种数据类型是列表(list)。list是一种最具灵活性的有序集合对象类型，可以随时添加和删除其中的元素。

1.列表

1.1创建列表

通常使用左右方括号（即：[和]）将数据元素包裹起来创建一个列表，如下所示。

```
>>> List1=[1,2,3,4,5]
>>> List2=["a","b","c","d"]
```

其中列表list1中包含5个元素，分别是1、2、3、4、5，list1为列表名。这种创建列表的方式适用于对于列表中元素个数及其数值已知时。

1.列表

1.1创建列表

当遇到如将一个元组（参考3.2节）转换为列表时，则需要使用另外一种方法创建列表—调用list(tuple)函数，该函数返回一个包含tuple中所有元素的列表。

注：直接调用不带参的list()函数时，将返回一个空列表，即：[ ]。

1.列表

1.1创建列表

列表中的元素的数据类型可以各不相同，如int，string类型，甚至可以是一个列表类型，如在下例中[10,20]为一个list类型，它作为list3的一个元素存在于list3中。

```
>>> List3=['marry',2.0,5,[10,20]]
>>> List3
['marry', 2.0, 5, [10, 20]]
```

1.列表

1.1创建列表

如图3-1所示，列表的下标是从0开始，List3列表的第一个元素是'marry'，用L[0]可以表示L的第一个元素，第二个元素是2.0，用L[1]表示，以此类推，第四个元素是一个列表，即[10, 20]。

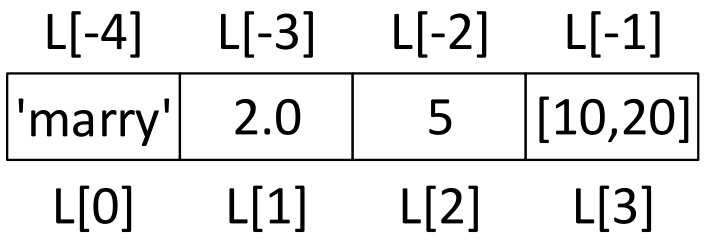

图3-1　元组索引位置

1.列表

1.2读取元素

元素下标表示该元素在list中的位置。注意list中元素下标是从0开始的，如第n个元素下标为n-1。但当读取元素传入的元素下标超出list集合的大小时将会报“元素下标超出范围”的错误。

1.列表

1.2读取元素

由于list2的长度为4，在取第5个元素时，list2中元素的最大下标为3<5，因此出现“list index out of range”的错误。

```
>>> List2=["a","b","c","d"]
>>> List2[0]  #访问列表的第一个元素
'a'
>>> List2[1]  #访问列表的第二个元素
'b'
>>> List2[5]  #超出列表元素下标，报错
Traceback (most recent call last):
  File "<pyshell#46>", line 1, in
<module>
    List2[5]
IndexError: list index out of range
```

1.列表

1.2读取元素

除了正向取list中的元素外，也可以逆向去取，用元素下标-1表示最后一个元素，-2表示倒数第二个元素，同样注意不能超出元组个数的界限，例如：。

```
>>> List2[-1]
'd'
>>> List2[-2]
'c'
>>> List2[-3]
'b'
>>> List2[-5]  #超出列表元素下标，报错
Traceback (most recent call last):
  File "<pyshell#50>", line 1, in
<module>
    List2[-5]
IndexError: list index out of range
```

1.列表

1.3遍历列表

遍历一个列表元素的最常见方式是使用for循环，常见的有以下两种方式。

```
>>> for cheese in cheeses:
        print(cheese)
```

第一种遍历方法隐藏了列表cheeses的长度，操作较为便利。

1.列表

1.3遍历列表

第二种遍历方法则使用len()函数计算出列表numbers的长度后进行遍历操作，其中range()函数返回的是从0到numbers长度的数值序列。

```
>>> for i in range(len(numbers)):
        numbers[i]=numbers[i] *2
```

1.列表

1.4替换元素

和字符串不同的是，列表是可变的，可以在列表中指定下标的值对元素进行修改，例如：

```
>>> numbers=[12,13]
>>> numbers[1]=14
>>> numbers
[12, 14]
```

numbers[1]原先为13，当执行 numbers[1]=14时，numbers[1]的值被修改为14。

1.列表

1.5增加元素

方法一：使用“+”将一个新列表附加在原列表的尾部。例如：和字符串不同的是，列表是可变的，可以在列表中指定下标的值对元素进行修改，例如：

```
>>> list=[1]
>>> list=list+['a', 'b']
>>> list
[1, 'a', 'b']
*操作符重复一个列表多次：
>>> [0]*5
[0, 0, 0, 0, 0]
>>> [4,5,6] *4
[4, 5, 6, 4, 5, 6, 4, 5, 6, 4, 5, 6]
```

1.列表

1.5增加元素

方法二：使用append()方法向列表的尾部添加一个新元素。例如：

```
>>> list.append(True)
>>> list
[1, 'a', 'b', True]
```

方法三：使用extend()方法将一个列表添加在原列表的尾部。例如：

```
>>> list.extend(['c',5])
>>> list
[1, 'a', 'b', True, 'c', 5]
```

1.列表

1.5增加元素

方法四：使用insert()方法将一个元素插入到列表的指定位置。该方法有两个参数，第一个参数为插入位置，第二个参数为插入元素。例如：

```
>>> list.insert(0,'x')
>>> list
['x', 1, 'a', 'b', True, 'c', 5]
```

1.列表

1.6检索元素

使用count()方法计算列表中某个元素出现的次数。

```
>>> list=['x','y','a','b',True,'x']
>>> list.count('x')
2
```

使用in运算符检查某个元素是否在列表中。

```
>>> 3 in list
False
>>> 'x' in list
True
```

1.列表

1.6检索元素

使用index()方法返回某个元素在列表中的准确位置，若该元素不在列表中将会出错。值得注意的是，若使用该方法的元素在该列表中存在相同项，则返回显示最小index的位置，如list.index('x')，存在两个'x'，则只显示最小位置。

```
>>> list.index('x')
0
```

1.列表

1.7删除元素

方法一：使用del语句删除某个特定位置的元素。

```
>>> list=['x','y','a','b',True,'x']
>>> del list[1]
>>> list
['x', 'a', 'b', True, 'x']
```

1.列表

1.7删除元素

方法二：使用remove()方法删除某个特定值的元素。remove('x')从list中移除最左边出现的数据项x，如果找不到x就产生ValueError。

```
>>> list=['x','y','a','b',True,'x']
>>> list.remove('x')
>>> list
['y', 'a', 'b',True, 'x']
>>> list.remove('x')
>>> list
['y', 'a', 'b',True]
>>> list.remove('x')
Traceback (most recent call last):
  File "<pyshell#12>", line 1, in <module>
    list.remove('x')
ValueError: list.remove(x): x not in list
```

1.列表

1.7删除元素

方法三：使用pop()方法来弹出（删除）指定位置的元素，缺省参数时弹出最后一个元素。弹出空数组将会报错。

```
>>> list=['x','y','a','b',True,'x']
>>> list.pop()
'x'
>>> list
['x', 'y', 'a', 'b', True]
>>> list.pop(1)
'y'
>>> list
['x', 'a', 'b', True]
>>> list.pop(1)
'a'
```

1.列表

1.7删除元素

接上一页PPT

```
>>> list.pop(1)
'b`
>>> list.pop(1)
True
>>> list.pop()
'x'
>>> list
[]
>>> list.pop()
Traceback (most recent call last):
  File "<pyshell#54>", line 1, in <module>
    list.pop()
IndexError: pop from empty list
```

1.列表

1.8字符串和列表的转化

字符串是字符的序列，而列表是值的序列，但字符的列表和字符串并不相同。若要将一个字符串转化为一个字符的列表，可以使用函数list。

```
>>> s='Micheal'
>>> t=list(s)
>>> t
['M', 'i', 'c', 'h', 'e', 'a', 'l']
```

1.列表

1.8字符串和列表的转化

由于list是内置函数的名称，所以应当尽量避免使用它作为变量名称。list函数会将字符串拆成单个的字母。如果想要将字符串拆成单词，可以使用split方法。

```
>>> s='you are so beautiful'
>>> t=s.split()
>>> t
['you', 'are', 'so', 'beautiful']
```

1.列表

1.8字符串和列表的转化

但是下面的例子却失败了，输出了整个列表。

```
>>> u='www.studyPython.com.cn'
>>> d=u.split()
>>> d
['www.studyPython.com.cn']
```

这时候就要用split接受一个可选的形参，作为分隔符，用于指定用哪个字符来分割单词。例如上面的例子可以用一个“.”作为分隔符，如下所示。

```
>>> u='www.studyPython.com.cn'
>>> c=u.split('.')
>>> c
['www', 'studyPython', 'com', 'cn']
```

1.列表

1.8字符串和列表的转化

join是split的逆操作。它接收字符串列表，并拼接每个元素。join是字符串的方法，所以必须在分隔符上调用它，并传入列表作为实参。

```
>>> t=['you','are','so','beautifil']
>>> s=' '.join(t)
>>> s
'you are so beautifil'
```

1.列表

1.8列表的常用函数

cmp()

- **格式**：cmp(列表1,列表2)。
- **功能**：对两个列表进行比较，若第一个列表大于第二个，则结果为1，相反则为-1，元素完全相同则结果为0。

```
>>> list1=[123,'xyz']
>>> list2=[123,'abc']
>>> cmp(list1,list2)
1
>>> cmp(list2,list1)
-1
>>> list2=list1
>>> cmp(list1,list2)
0
```

1.列表

1.8列表的常用函数

len()

- **格式：**len(列表)。
- **功能：**返回列表中的元素个数。

```
>>> len(list1)
   2
```

1.列表

1.8列表的常用函数

max()和min()

- **格式**：max(列表)　min(列表)。
- **功能**：返回列表中的最大或最小元素。

```
>>> str_l=['abc','xyz','123']
>>> num_l=[123,456,222]
>>> max(str_l)
'xyz'
>>> min(str_l)
'123'
>>> max(num_l)
456
>>> min(num_l)
123
```

1.列表

1.8列表的常用函数

sorted()和resersed()

- **格式**：sorted(列表) reversed(列表)。
- **功能**：前者的功能是对列表进行排序，默认是按升序排序，还可在列表的后面增加一个reverse参数，其等于True则表示按降序排序；后者的功能是对列表进行逆序。

```
>>> list=[1,4,3,6,9,0,2]
>>> for x in reversed(list):
                print x,

2 0 9 6 3 4 1
>>> sorted(list)
[0, 1, 2, 3, 4, 6, 9]
>>> sorted(list,reverse=True)
[9, 6, 4, 3, 2, 1, 0]
```

1.列表

1.8列表的常用函数

sum()

- **格式：**sum(列表)。
- **功能：**对数值型列表的元素进行求和运算，对非数值型列表运算则出错。

```
>>> sum(list)
25
>>> sum(str_l)
Traceback (most recent call last):
  File "<pyshell#26>", line 1, in <module>
    sum(str_l)
TypeError: unsupported operand type(s) for +: 'int' and 'str
```

代码练习题

已知列表name_list = [“zhangsan”, ”lisi”, ”wangwu”]，请做出如下修改：
1、修改’lisi’为’李四’；
2、在列表末尾追加数据’王小二‘；
3、删除’wangsu’。

答案：

```
name_list[1] = "李四"
name_list.append("王小二")
name_list.remove("wangwu")
```

代码练习题

请统计列表name_list中'张三'出现的次数。

```
name_list = ["张三", "李四", "王五", "王小二", "张三"]
```

答案：

```
count = name_list.count("张三")
print("张三出现了 %d 次" % count)
```