

CS001 编程零基础 Python语言入门

第六讲

本讲内容

列表的创建和使用

循环处理列表

列表排序

什么是列表

- 单个数据可以放在变量里
- 一组数据可以放在列表里

```
family=['Mom','Dad','Me','Baby']
```

```
luckyNumbers=[2,7,14,26,30]
```

```
Anything=[5,10,'Hello',3.14]    #可以不是同一种类型
```

列表中的每一项叫做元素。中括号作为首尾开始和结束，逗号作为分隔。

创建列表

- `newList=[]`
- 空列表。
- 一开始可能只知道需要一个列表来保存内容，但并不知道内容是什么。
- 有了空列表后，可以向列表增加元素。

向列表增加元素

list1.py

使用列表的成员函数append()

```
#list1.py
friends=[]
print(friends)
friends.append('David')
print(friends)
friends.append('Mary')
print(friends)
```

注意：和内置函数不同，成员函数的使用一般形式为：
列表名.成员函数()

```
[]
['David']
['David', 'Mary']
```

从列表获取元素



按元素的索引号（位置）从列表获取单个元素。

索引位置从0开始编号。

```
#list2.py
letters=['a','b','c','d','e']
print(letters[0])
print(letters[3])
```


列表切片

list3.py

```
#list3.py  
letters=['a','b','c','d','e']  
print(letters[1:3])  
print(letters[:2])  
print(letters[2:])  
print(letters[:])
```

```
['b', 'c']  
['a', 'b']  
['c', 'd', 'e']  
['a', 'b', 'c', 'd', 'e']
```

修改元素

利用赋值直接修改列表某个位置的元素。

位置只能是 $0 \sim N-1$,即不能修改不存在的元素。

```
#list4.py  
letters=['a','b','c','d','e']  
letters[2]='z'  
print(letters)  
letters[5]='z'  
print(letters)
```

```
['a', 'b', 'z', 'd', 'e']
```

```
Traceback (most recent call last):
```

list4.py

向列表增加元素的其它成员函数

`append()`向列表末尾增加一个元素

`extend ()`向列表末尾增加多个元素

`insert()`在列表某个位置增加一个元素

```
#list5.py
letters=['a','b','c','d','e']
letters.extend(['f','g','h'])
print(letters)
letters.insert(2,'z')
print(letters)
```

注意：这是错误的写法

```
letters.extend('f','g','h')
```

```
['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h']
['a', 'b', 'z', 'c', 'd', 'e', 'f', 'g', 'h']
```

从列表删除元素

remove(): 成员函数。从列表删除指定**内容**的元素。

del: 不是成员函数。从列表删除指定**位置**的元素。

```
#list6.py
letters=['a','b','c','d','e']
letters.remove('c')
print(letters)
#letters.remove('f') #wrong
del letters[3]
print(letters)
```

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

从列表删除元素

成员函数`pop()`从列表删除最后一个或者指定位置的元素，并返回该元素。

```
#list7.py
letters=['a','b','c','d','e']
l=letters.pop()
print(letters)
print(l)
l=letters.pop(2)
print(letters)
print(l)
```

```
['a', 'b', 'c', 'd']
e
['a', 'b', 'd']
c
```

搜索列表

list8.py

- **in**: 运算符。判断某元素是否在列表中，返回True/False。
- **index**: 成员函数。返回某元素在列表中的位置。若不存在，则出错。

```
#list8.py
def exist(l, c):
    if c in l:
        return True
    else:
        return False

letters=['a','b','c','d','e']
if exist(letters, 'a'):
    print('Found in ', end='')
    print(letters.index('a'))
else:
    print('Not found.')
```

Found in 0

利用for循环

loop.py

```
#loop.py  
letters=['a','b','c','d','e']  
for letter in letters:  
    print(letter)
```

a
b
c
d
e

利用for循环

average.py

```
#average.py
scores=[90, 89, 95, 100, 85]
sum=0
n=0
for s in scores:
    sum+=s
    n+=1
print('Average:%.2f'%(sum/n))
```

注意：对sum赋值后，在同一个shell里，就不能使用后面讲的sum函数了。

Average:91.80

利用列表的内置函数

average2.py

- `sum()`对列表所有元素求和
- `len()`返回列表中元素的个数

```
#average2.py  
scores=[90, 89, 95, 100, 85]  
average=sum(scores)/len(scores)  
print('Average: %.2f' %average)
```

Average:91.80



使用成员函数`sort()`进行列表排序

- 就像这一讲的大多数函数一样，`sort()`是对列表原地修改，并不是返回一个新列表。

```
#sort1.py
letters=['d','a','e','c','b']
print(letters)
letters.sort()
print(letters)
print(letters.sort())
```

```
['d', 'a', 'e', 'c', 'b']
['a', 'b', 'c', 'd', 'e']
None
```

最后为什么输出None?



使用成员函数**reverse()**进行列表逆序

```
#sort2.py  
letters=['d','a','e','c','b']  
letters.sort()  
print(letters)  
letters.reverse()  
print(letters)
```

```
['a', 'b', 'c', 'd', 'e']  
['e', 'd', 'c', 'b', 'a']
```




使用内置函数**sorted()**得到排序的副本

```
#sort3.py  
letters=['d','a','e','c','b']  
newer=sorted(letters)  
print(letters)  
print(newer)
```

```
['a', 'b', 'c', 'd', 'e']  
['e', 'd', 'c', 'b', 'a']
```


复习阅读

- 课本第7章的7.4~7.6
- 课本第6章的6.3.1

