day06 : 列表

笔记本:	Python基础		
创建时间:	2018/6/7 15:22	**更新时间:**	2018/7/2 19:35
作者:	liuchang_0412@163.com		

day06 笔记

1.列表 list:

定义：列表是由一系列特定元素组成的，元素之间可能没有任何关联，但他们之间有先后顺序关系

列表可以改变各个元素的值

列表是一种容器

①创建空列表:

```
L = [ ]     #空列表
L = list() #空列表
```

②创建非空列表:

```
L = [1 , 2 , 3 , 4]
L = ["Beijing" , "Shanghai" , "Shenzhen"]
L = [1, "two" , 3.0 , 'four']
L = [1, 2, [3.1 , 3.2] , 4]
```

③列表的生成函数list()

list( ) 生成一个空列表，等同于[ ]

list(iterable) 用可迭代对象初始化一个列表

```
L = list( "hello" )                    # L--> ['h', 'e', 'l', 'l', 'o']
i  = range(0,10)  L = list(i)          # L--> [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

2.列表的运算

①算术运算： + , += , * , *=

\+ 加号运算符用于拼接列表

例：x=[1, 2 , 3]   y=[4 , 5 , 6]   z=x+y   #z--> [1 , 2 , 3 , 4 , 5 , 6]

+= 运算符用于原列表与右侧列表拼接生成新的列表

例：x=[1, 2 , 3]   y=[4 , 5 , 6]   x+=y   #x--> [1 , 2 , 3 , 4 , 5 , 6]

* 运算符用于生成重复的列表

例：x=[1 , 2] * 3   #x-->[1 , 2 , 1 , 2 , 1 , 2]

*= 运算符用于生成重复的列表，并改变变量的绑定

例：x=[1,2]   x*=3   #x-->[1 , 2 , 1 , 2 , 1 , 2]

②列表的关系运算： > , >= , < , <= , == , !=

例： x=[1,2,3]   y=[2,3,4]

```
x != y    #True
```

```
x==y     #False
x > y     #False  列表中的每个元素按序进行比较
x < y     #True
[1, "two"] < ["two" , 1]     #报错，类型错误
```

③列表的 in / not in 运算符：判断一个元素是否在列表内

例：x=[1, 'two' , 3.0 , "four"]

```
1 in x    #True
3 in x    #True
```

3.列表的基本操作：

①索引操作 index，列表索引规则等同于字符串的索引

格式：列表[index]

列表是可变的，可以通过索引赋值改变列表的元素

例：x = [1, 2, 3, 4]   x[2] = 3.14   # x --> [1, 2, 3.14, 4]

②切片 slice，列表切边规则等同于字符串切片规则

格式：列表[起始值 : 结束值 : 步长]

例：x = [1 ,2 ,3 ,4 ,5]   y = x[1 : : 2]   # y--> [2, 4]

③切片赋值：切片赋值可以改变原列表的排列，及插入、删除数据。

列表中可以用切片改变对应元素的值。

例：#用多个值来代替一个值

```
L = [2 , 3 , 4]      L[0:1]=[1.1 , 2.2 ]      #L --> [1.1 , 2.2 , 3, 4]
#用多个值来代替多个值
L[2: ] = [3.3, 4.4 , 5.5]                      #L --> [1.1 ,2.2 ,3.3 ,4.4 ,5.5]
#完全替换
L[ : ] = [3 ,4 ]                               #L --> [3 , 4]
#在列表中插入
L[ 1:1 ] = [3.1 , 3.2]                         #L -->[ 3 , 3.1 , 3.2 , 4]
#等位替换，对于步长大于1的切片赋值，可能会出现赋值错误问题
L=[1,2,3,4,5,6]
L[0 : : 2]= [1.1 , 3.3 , 5.5]                  # L--> [1.1, 2, 3.3, 4, 5.5, 6]
```

#切片赋值是改变原列表，不会生成新列表

例：L=[1, 2, 3, 4, 5, 6]

```
L2= L
L[ : : 2] = [0.1 , 0.2 , 0.3]     #L=[0.1, 2 , 0.2 , 4 , 0.3 ,6]   L2=[0.1, 2 , 0.2 ,
4 , 0.3 ,6]
```

4.python3 中列表常用的序列函数

len(x)　　返回序列的长度

max(x)　返回序列的最大值的元素
min(x)　返回序列最小值的元素
sum(x)　返回序列所有元素的和
any(x)　真值测试，如果列表中其中一个值为真值则返回 True
all(x)　真值测试，如果列表中所有值都是真值则返回 True

5.python3 中列表的常用方法
见：>>> help(list)
以下L代表列表

#方括号内的内容可以省略
L.index(v [, begin [, end]])　#返回对应元素的第一个索引下标 begin为开始索引，end未结束索引
L.insert(index, obj)　#将某个元素差放到列表中指定的位置
L.count(x)　#返回列表中元素的个数
L.remove(x)　#从列表中删除第一次出现在列表中的值
L.copy( )　#返回此列表的复制（只复制一层，不进行深层复制）
L.append(x)　#在列表尾部添加单个元素
L.extend(list)　#向列表追加另一个列表
L.clear( )　#清空列表，等同于 L[:] = [ ]
L.sort(reverse = False)　#将列表的顺序按照值的小到大顺序进行排列
L.reverse( )　#列表反转
L.pop([index])　#弹出索引对应的元素，如果不加索引，默认弹出最后一个元素

6.列表嵌套：
```
L = [20, 21, 22]
L1 = [10 , L ,30]
L2 = L1.copy
print(L1)　　# L1 --> [10, [20, 21, 22], 30]
print(L2)　　# L2 --> [10, [20, 21, 22], 30]

L1[1][1] = 5　　# 等同于 L[1] = 5
print(L1)　　# L1 --> [10, [20, 5, 22], 30]
print(L2)　　# L2 --> [10, [20, 5, 22], 30]
```

7.复制列表
#深拷贝和浅拷贝：
①浅拷贝（shallow copy）：
L.copy()　　#列表复制函数
L[:]　　#切片复制是浅拷贝

例：见6.中的内容

②深拷贝（deep copy）：将对象逐层复制，复制后的对象完全独立

```
import copy          #导入copy模块
L = [20, 21, 22]
L1 = [10 , L ,30]
L2 = copy.deepcopy(L1)
print(L1)            # L1 --> [10, [20, 21, 22], 30]
print(L2)            # L2 --> [10, [20, 21, 22], 30]
L1[1][1] = 5         # 等同于 L[1] = 5
print(L1)            # L1 --> [10, [20, 5, 22], 30]
print(L2)            # L2 --> [10, [20, 21, 22], 30]
```

8.列表运算符

del 运算符用于删除列表元素

```
例：cities = [ "北京" , "上海" , "深圳" , "天津" ]
cities.remove("深圳")        #删除"深圳"
cities.pop(2)                #删除"深圳"
del cities[2]                #删除"深圳"
```

#练习：输入任意整数，先判断你输入的数是否为质数（只能被1和自身整除的数），如果为质数则加入到列表中，再次输入任意整数后判断，直至输入的数小于等于1为止，最后打印您输入的质数。

9.列表推导式（list comprehension）：列表推导式是用可迭代对象，依次生成列表内元素的方式

语法：

```
[表达式 for 变量 in 可迭代对象] 或
[表达式 for 变量 in 可迭代对象 if 条件表达式]
例：L = [x**2 for x in range(1, 11)]        #生成列表[1,4,9,16,...]
    L = [x**2 for x in range(1, 11, 2)]     #生成列表[1,9,25,...]
    L = [x**2 for x in range(1, 11) if x%2==1]      #生成列表[1,9,25,...]
```

10.列表推导式的嵌套

语法：

[表达式 for 变量1 in 可迭代对象1 (if 条件表达式1) for 变量2 in 可迭代对象2 (if 条件表达式2)]

例：

```
#将列表[2,3,5]中的元素与列表[7,11,13]中的元素分别现成，将得到的元素放于一个列表中
L1 = [ x*y for x in [2,3,5] for y in [7,11,13]]
```

#练习：

1.生成前40个斐波那契数（1,1,2,3,5,8,..），将这些数保存在列表中，打印这些数。

2.完全数（除自身以外的所有的因数之和等于自身，例如 1+2+3=1*2*3=6），求4~5个完全数并打印。