实用Python程序设计

郭 炜

微信公众号

微博：http://weibo.com/guoweiofpku

学会程序和算法，走遍天下都不怕!

Python 组合数据类型(二)
字典和集合

字典的基本概念

泰国普吉岛

字典

- 字典的每个元素是由“键：值”两部分组成，可以根据“键”进行快速查找
- **格式：** `d = {key1 : value1, key2 : value2..... }`
- 字典元素的值是可赋值的，因此也是指针
- 所有元素的键都不相同
- 键必须是不可变的数据类型，比如字符串、整数、小数、元组。
- 一个元组如果包含可变数据类型，也不能作为集合的元素
- 列表、集合、字典等可变的数据类型，不可作为字典元素的键。

字典

```
dt = {'Jack':18,'Mike':19, 128:37, (1,2):[4,5] }
print(dt['Jack'])              #>>18  键为'Jack'的元素值是18
print(dt[128])                 #>>37  键为128的元素值是37
print(dt[(1,2)])               #>>[4, 5]
print(dt['c'])        #不存在键为'c'的元素，产生异常，导致运行时错误
dt['Mike'] = 'ok'           #将键为 'Mike' 的元素的值改为 'ok'
```

字典

```
dt['School'] = "Pku"      #添加键为 'School'的元素，其值为'Pku'
print(dt)
#>>{128: 37, (1, 2): [4, 5], 'Jack': 18, 'Mike': 'ok',
   'School': 'Pku'}
del dt['Mike']            #删除键为'Mike'的元素
print(dt)
#>>{128: 37, (1, 2): [4, 5], 'Jack': 18, 'School': 'Pku'}
scope={}                  #空字典
scope['a'] = 3            #添加元素 'a':3
scope['b'] = 4            #添加元素 'b':4
```

字典

```
print(scope)          #>>{'a': 3, 'b': 4}
print('b' in scope) #>>True          判断是否有元素键为'b'
scope['k'] = scope.get('k',0) + 1
#get(key,v) :如果键key存在，则返回键为key的元素的值，否则返回v
print(scope['k'])     #>>1
scope['k'] = scope.get('k',0) + 1
print(scope['k'])     #>>2
```

字典的键不可重复

- 字典的键不可重复，指的是字典的键的内容不能一样

```
a = (1,2,3)
b = (1,2,3)
d = {a:60,b:70,(1,2,3):80, (1,2,3):50}
#d中实际上只有一个元素
print(d[a])              #>>50
print(d[b])     #>>50
print(d[(1,2,3)])      #>>50
for x in d.keys():
        print(x)         #此循环只输出一个 (1, 2, 3)
```

单选题 1分

设置

下面哪条语句是不正确的

A dt = {'jack':[1,2], 100:(4,6)}

B {'jack':20, 18:30}[18] = 31

C {'jack':20, 18:30}[100] = 31

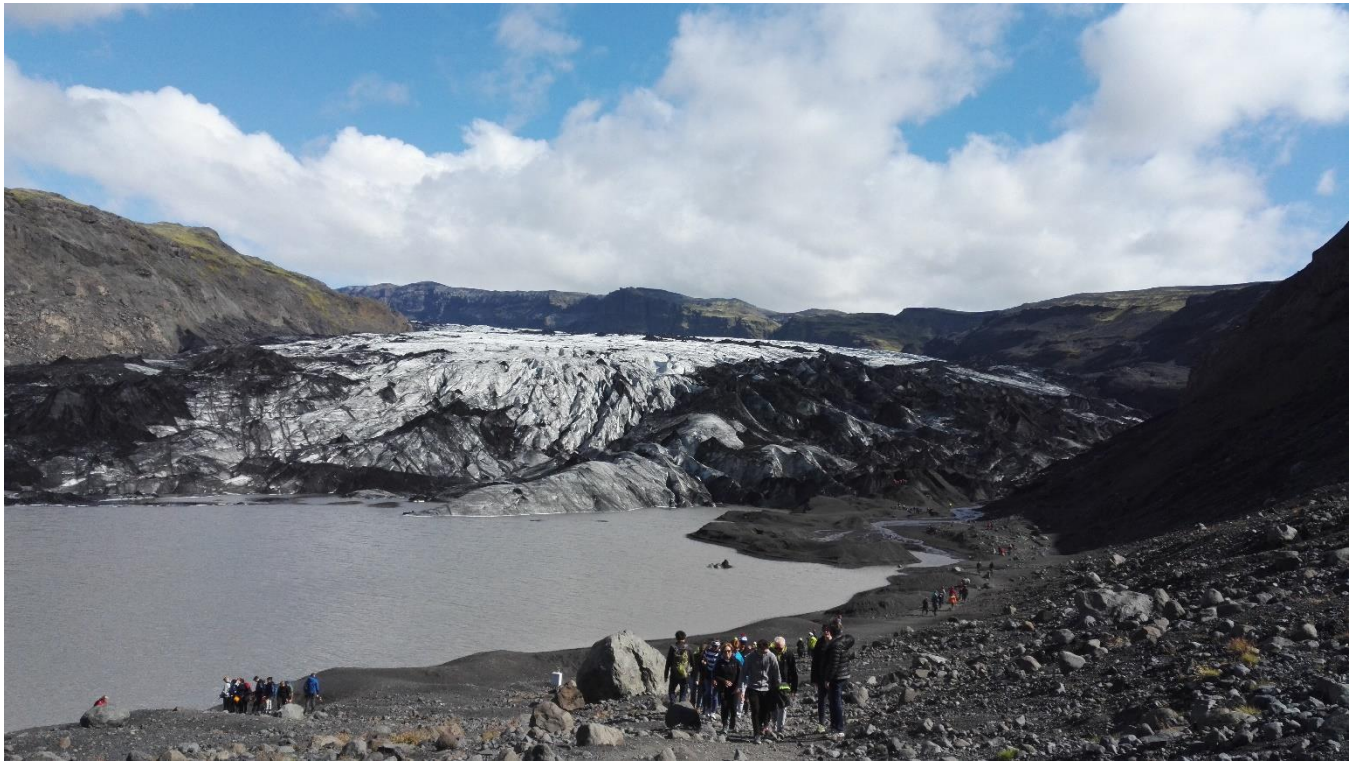
D dt = {[1,2]:3, 'jack':4}

提交

字典的构造

```
items = [('name','Gumby'),('age',42)]
d = dict(items)
print(d)      #>>{'name': 'Gumby', 'age': 42}
d = dict(name='Gumby',age=42,height=1.76)
print(d)      #>>{'height': 1.76, 'name': 'Gumby', 'age': 42}
```

字典相关函数



冰岛索尔黑马冰川

字典的函数

- clear()　清空字典
- keys()　取字典的键的序列
- items()　取字典的元素的序列，可用于遍历字典
- values()　取字典的值序列
- pop(x)　删除键为x的元素，如果不存在，产生异常

上述"序列"，不是 list,tuple或set

- copy()　浅拷贝
- get(k,v)　如果有元素键为k，则返回该元素的值，如果没有，则返回v

字典的函数

```
d={'name': 'Gumby', 'age': 42, 'GPA':3.5}
if 'age' in d.keys():
    print(d['age'])          #>>42
for x in d.items():     #>> ('GPA', 3.5),('age', 42),('name', 'Gumby'),
    print(x,end = ",")
for x in d.items():         #>>GPA,age,name,
    print(x[0],end = ",")
for x in d.items():         #>>3.5,42,Gumby,
    print(x[1], end = ",")
for k,v in  d.items():   #>>GPA 3.5,age 42,name Gumby,
    print (k,v,end = ",")
for x in d.keys():          #>>GPA,age,name,
    print(x,end=",")
```

字典的函数

```
for x in d.values():     #>>3.5,42,Gumby,
    print(x,end=",")
x = {'username':'admin', 1978:[1, 2, 3]}
y = x.copy()
y['username'] = 'mlh'
y[1978].remove(2)    #删除元素2
print(y)     #>>{'username': 'mlh', 1978: [1, 3]}
print(x)     #>>{'username': 'admin', 1978: [1, 3]}
x.pop('username')    #删除键为'username'的元素
print(x)             #>>{1978: [1, 3]}
d.clear()
print(d)     #>>{}
```

字典的深拷贝

```
import copy
x = {'username':'admin', 'machines':['foo', 'bar', 'baz']}
y = copy.deepcopy(x)
y['username'] = 'mlh'
y['machines'].remove('bar')
print(y)
print(x)
=>
{'username': 'mlh', 'machines': ['foo', 'baz']}
{'username': 'admin', 'machines': ['foo', 'bar', 'baz']}
```

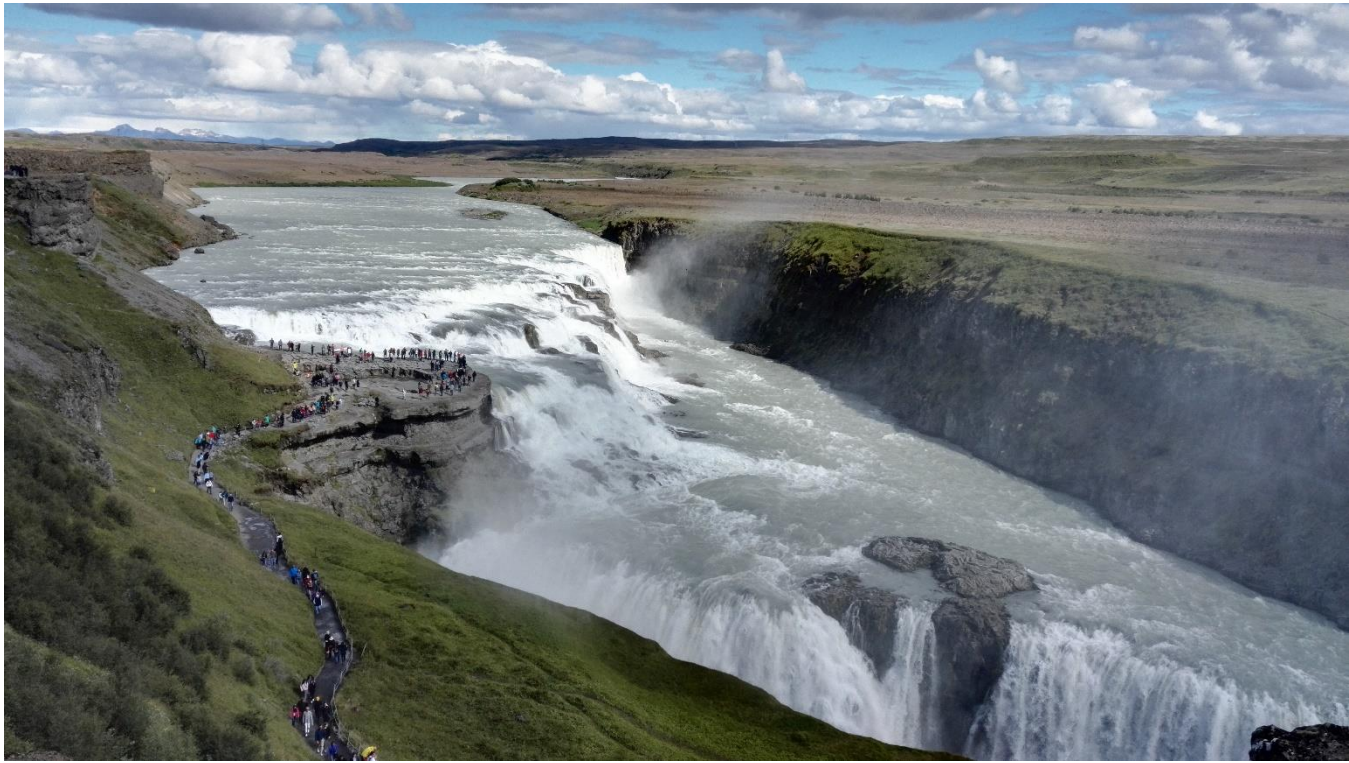
遍历字典

- items

```
x = {'username':'admin', 'machines':['foo', 'bar', 'baz'],
   'Age':15}
for i in x.items():
    print(i[0])
    print(i[1])
```

遍历字典时，在python3.5及以前，顺序不确定。在python 3.6及以后，顺序同元素加入字典的顺序

字典例题



冰岛黄金瀑布

字典例题：统计单词频率

输入

若干行，每行一个单词。

输出

按单词出现次数从高到低打出所有单词。次数相同的，按照字典序从小到大排

输入样例

```
about
send
about
me
```

输出样例

```
2 about
1 me
1 send
```

```
dt = {}
while True:
    try:
        wd = input()
        if wd in dt:          #如果有元素键为wd
            dt[wd] += 1
        else:
            dt[wd] = 1        #加入键为wd的元素，其值是1
    except:
        break   #输入结束后的input()引发异常，跳到这里，再跳出循环
result = []
for x in dt.items():
    result.append(x) #x是个元组，x[0]是单词，x[1]是出现次数
result.sort(key = lambda x:(-x[1],x[0]))
for x in result:
    print(x[1],x[0])
```

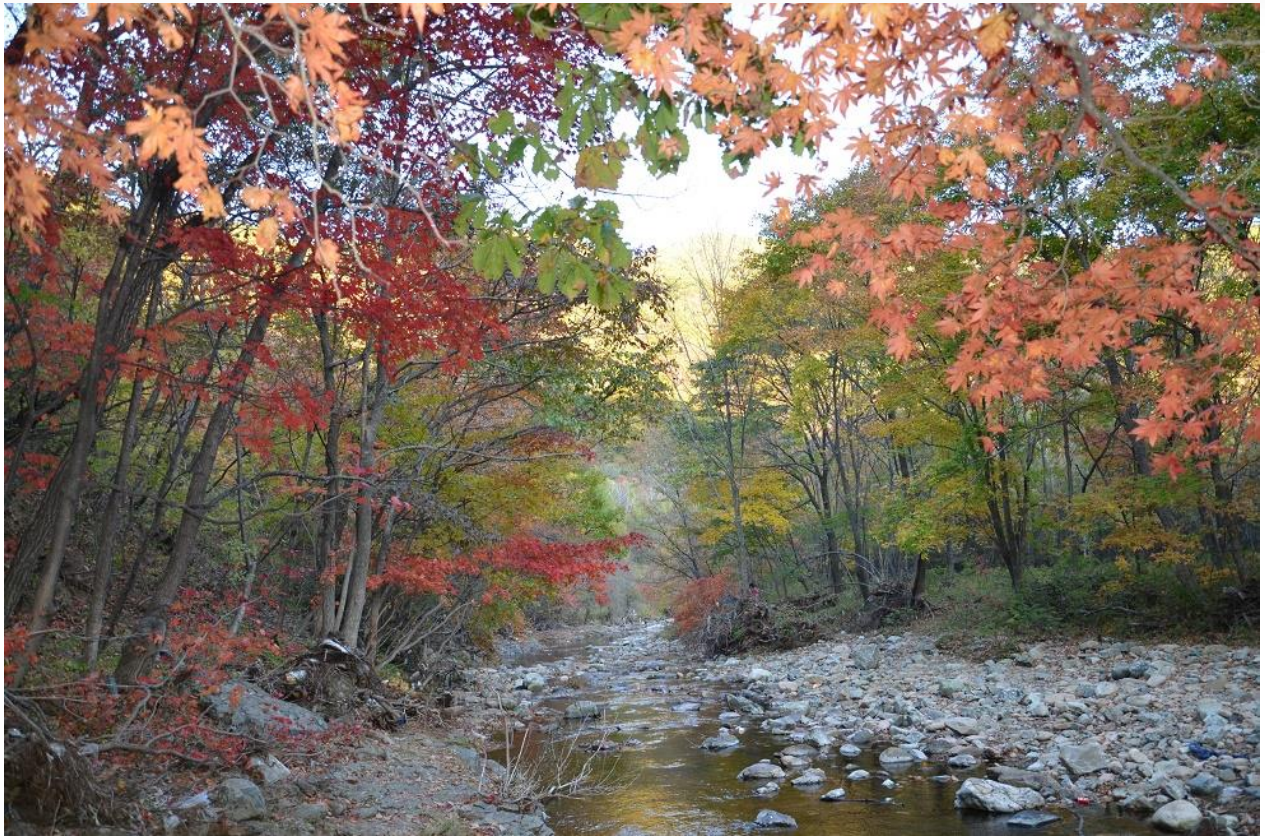
字典例题：统计单词频率

```
if wd in dt:
    dt[wd] += 1
else:
    dt[wd] = 1
```

也可改写为：

```
dt[wd] = dt.get(wd,0) + 1
#若在dt里有键为wd的元素，则get返回其值，否则返回0
```

集 合



本溪洋湖沟

集合(set)的概念和特点

集合(set)的概念同数学上的集合:

- 元素类型可以不同。
- 不会有重复元素。
- 可以增删元素。
- 整数、小数、复数、字符串、元组都可以作为集合的元素。但是列表、字典和集合等可变的数据类型不可作为集合的元素。
- 一个元组如果包含可变数据类型，也不能作为集合的元素
- 集合的作用是快速判断某个东西是否在一堆东西里面(用in)。

集合的构造

```
print(set([]))   #>>set()    集合可由列表转换得到，set([])是空集合
a = {1,2,2,"ok",(1,3)}       #自动去重
print(a)                     #>>{2, 1, 'ok', (1, 3)}
b = (3,4)
c = (3,4)
a = set((1,2,"ok",2,b,c))
for x in a:                  #>>ok 1 2 (3, 4)
    print(x,end = " ")
a = set("abc")               #>>字符串转集合
print(a)                     #>>{'b', 'c', 'a'}
a = set({1:2,'ok':3,(3,4):4})
print(a)                     #>>{1, 'ok', (3, 4)} 只取键
print(a[2])                  #错误，集合元素没有顺序，不能用下标访问
```

集合常用函数

- add(x)　　添加元素x。如果x已经存在，则不添加
- clear()　　清空集合
- copy()　　返回自身的浅拷贝
- remove(x)　　删除元素x。如果不存在元素x，则引发异常
- update(x)　　将序列x中的元素加入到集合

集合运算

a,b是集合：

- x in a　　x是否在集合a中
- a | b　　求a和b的并
- a & b　　求a和b的交
- a – b　　求a和b的差，即在a中而不在b中的元素
- a ^ b　　求a和b的对称差，等价于(a|b) – (a&b)

集合运算

a,b是集合：

- a == b　　a是否元素和b一样
- a != b　　a是否元素和b不一样
- a <= b　　a 是否是b的子集(a有的元素，b都有)
- a < b　　a 是否是b的真子集(a有的元素，b都有，且b还包含a中没有的元素)
- a >= b　　b 是否是a的子集
- a > b　　b 是否是a的真子集

集合示例程序

```
a = set([])                          #a是空集合
b = set([])
a.add(1)                             #添加元素1
a.update([2,3,4])                    #将列表元素添加进a
b.update(['ok',2,3,100])
print(a)                             #>>{1, 2, 3, 4}
print(b)                             #>>{2, 3, 100, 'ok'}
print( a | b)                        #>>{1, 2, 3, 4, 100, 'ok'} 求并
print( a & b )                       #>>{2, 3} 求交
print( a - b)                        #>>{1, 4}    求差
a -= b                               #在a中删除b中有的元素
print(a)                             #>>{1, 4}
```

集合示例程序

```
a ^= {3,4,544}                  #对称差
print(a)                        #>>{544, 1, 3}
a.update("take")
print(a)                        #>>{544, 1, 3, 'e', 'k', 't', 'a'}
print(544 in a)                 #>>True
a.remove(544)                   #删除元素，若元素不存在，会出错
print(a)                        #>> {1, 3, 'a', 'k', 't', 'e'}
a = {1,2,3}
b = {2,3}
print( a > b)                   #>>True  b是a的真子集
print( a >= b)                  #>>True  b是a的子集
print( b < a)                   #>>True  b是a的真子集
```

集合例题

输入一些单词，统计不重复的单词一共有多少个。

输入样例:

about

take

about

zoo

take

输出样例:

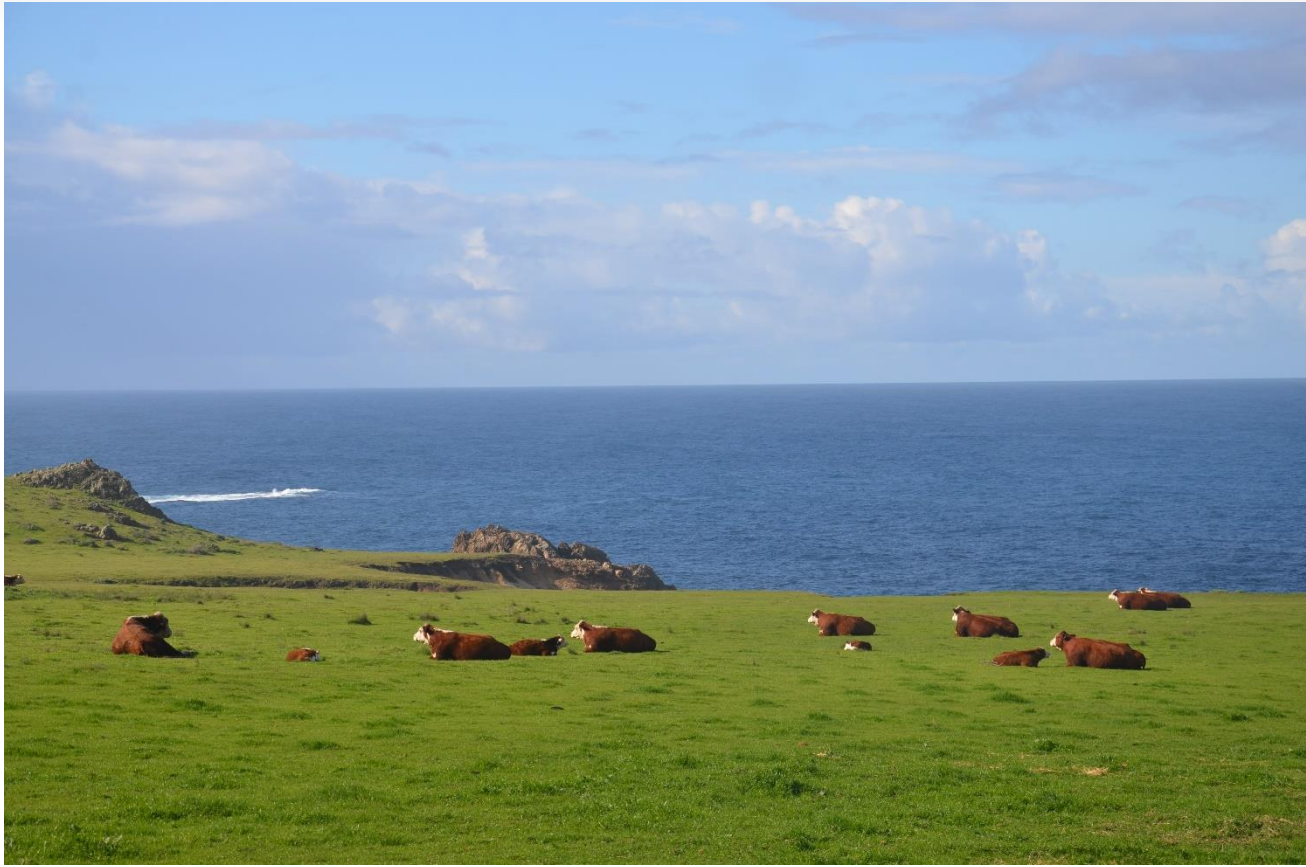
3

集合例题

```
words = set([])
while True:
    try:
        wd = input()
        if not wd in words:
            words.add(wd)
    except:
        break
print(len(words))
```

用列表做，比用集合慢很多很多！单词达到10万，就会非常明显。

程序或算法的时间复杂度



美国加州1号公路

程序或算法的时间复杂度

- 一个程序或算法的时间效率，也称“时间复杂度”，有时简称“复杂度”

程序或算法的时间复杂度

- 一个程序或算法的时间效率，也称“时间复杂度”，有时简称“复杂度”

- 复杂度常用大的字母O和小写字母n来表示，比如$O(n)$, $O(n^2)$等。n代表问题的规模

程序或算法的时间复杂度

●一个程序或算法的时间效率，也称“时间复杂度”，有时简称“复杂度”

●复杂度常用大的字母O和小写字母n来表示，比如O(n)，O(n^2)等。n代表问题的规模。O(X）就表示解决问题的时间和X成正比关系

●时间复杂度是用算法运行过程中，某种时间固定的操作需要被执行的次数和n的关系来度量的。在无序数列中查找某个数，复杂度是O(n)

程序或算法的时间复杂度

●一个程序或算法的时间效率，也称“时间复杂度”，有时简称“复杂度”

●复杂度常用大的字母O和小写字母n来表示，比如O(n)，O(n^2)等。n代表问题的规模，O(X）就表示解决问题的时间和X成正比关系

●时间复杂度是用算法运行过程中，某种时间固定的操作需要被执行的次数和n的关系来度量的。在无序数列中查找某个数，复杂度是O(n)

●计算复杂度的时候，只统计执行次数最多的(n足够大时)那种固定操作的次数。比如某个算法需要执行加法n^2次，除法10000n次，那么就记其复杂度是O(n^2)的。

程序或算法的时间复杂度

- 如果复杂度是多个n的函数之和，则只关心随n的增长增长得最快的那个函数

$O(n^3+n^2) \Rightarrow O(n^3)$
$O(2^n+n^3) \Rightarrow O(2^n)$
$O(n! + 3^n) \Rightarrow O(n!)$

程序或算法的时间复杂度

- 如果复杂度是多个n的函数之和，则只关心随n的增长增长得最快的那个函数

$O(n^3+n^2) \Rightarrow O(n^3)$
$O(2^n+n^3) \Rightarrow O(2^n)$
$O(n! + 3^n) \Rightarrow O(n!)$

- 常数复杂度：O(1)
- 对数复杂度：O(log(n))
- 线性复杂度：O(n)
- 多项式复杂度：$O(n^k)$
- 指数复杂度：$O(a^n)$
- 阶乘复杂度：O(n!)

时间(操作次数)和问题的规模无关

程序或算法的时间复杂度

- 在无序数列中查找某个数(顺序查找)　　O(n)
- 插入排序、选择排序等笨排序方法　O(n^2)
- 快速排序　O(n*log(n))
- 二分查找　O(log(n))

in用于列表和用于字典、集合的区别

a in b

若b是列表，字符串或元组，则该操作时间复杂度O(n)，即时间和b的元素个数成正比

若b是字典或集合，则该操作时间复杂度O(1)，即时间基本就是常数，和b里元素个数无关

因此集合用于需要经常判断某个东西是不是在一堆东西里的情况

此种场合用列表替代集合，容易导致超时!!!!

一些操作的时间复杂度总结

O(1)：集合、字典增删元素，查找元素，以关键字作为下标访问字典元素的值，列表添加元素到末尾（append），列表、字符串、元组根据下标访问元素

O(n)：列表、元组查找元素(in,index)，列表插入元素(insert)、删除元素(remove)

计算出现次数(count)

O(nlog(n))：python自带排序 sort,sorted

O(log(n))：在排好序的列表或元组上进行二分查找（初始的查找区间是整个元组或列表，每次和查找区间中点比较大小，并缩小查找区间到原来的一半。类似于查英语词典）有序就会找得快！