global 表示将函数变量更改为全局变量，使用方法 global+变量

final\_+变量 局部变量 ，应用于一个函数当中

lambda 表示直接赋值一个函数，省略def赋值过程，使语句更加简便，使用方法 赋值函数=lambda +运算字符+：+运算过程（运算字符包含在其中）

filter 过滤器

**列表返回筛选后的元素**

list(filter(筛选的条件，筛选的对象))

list(filter(lambda x : x%2,range(10)))

map 条件映射,把后面的元素放到前面的条件语句中

语法：list(map(条件过程，元素集合))

list(map(lambda x : x\*2 ,range(10)))

迭代资源占用率比递归低

字典名.keys(): 返回字典中的键

values(): 返回字典中的组

items(): 将大字典拆分为一个小字典

字典名.get 返回字典当中的字符，如果没有，不会报错，并且返回none

字典名.clear() 清空字典

数值 = 赋值数.copy 将数据赋值，数据独立存在

set:创建一个工厂函数

添加元素1在集合中：set1.add(1)

例如 set1=set(集合（1,2,3），字典{1,2,3}，列表【1,2,3】)

set函数不会有相同的数值在里面，并且可以把不同的集合转换为set函数

例如：set2={1,2,3,4,5,4,3,2,1}

set1=set(set2)

print(set1)

{1,2,3,4,5}

remove:移除set里的数值

例如移除set1里面的5，用法: set1.remove(5)

frozenset :不可变的元素集合

用法：set1=frozenset({1,2,3,4,5})

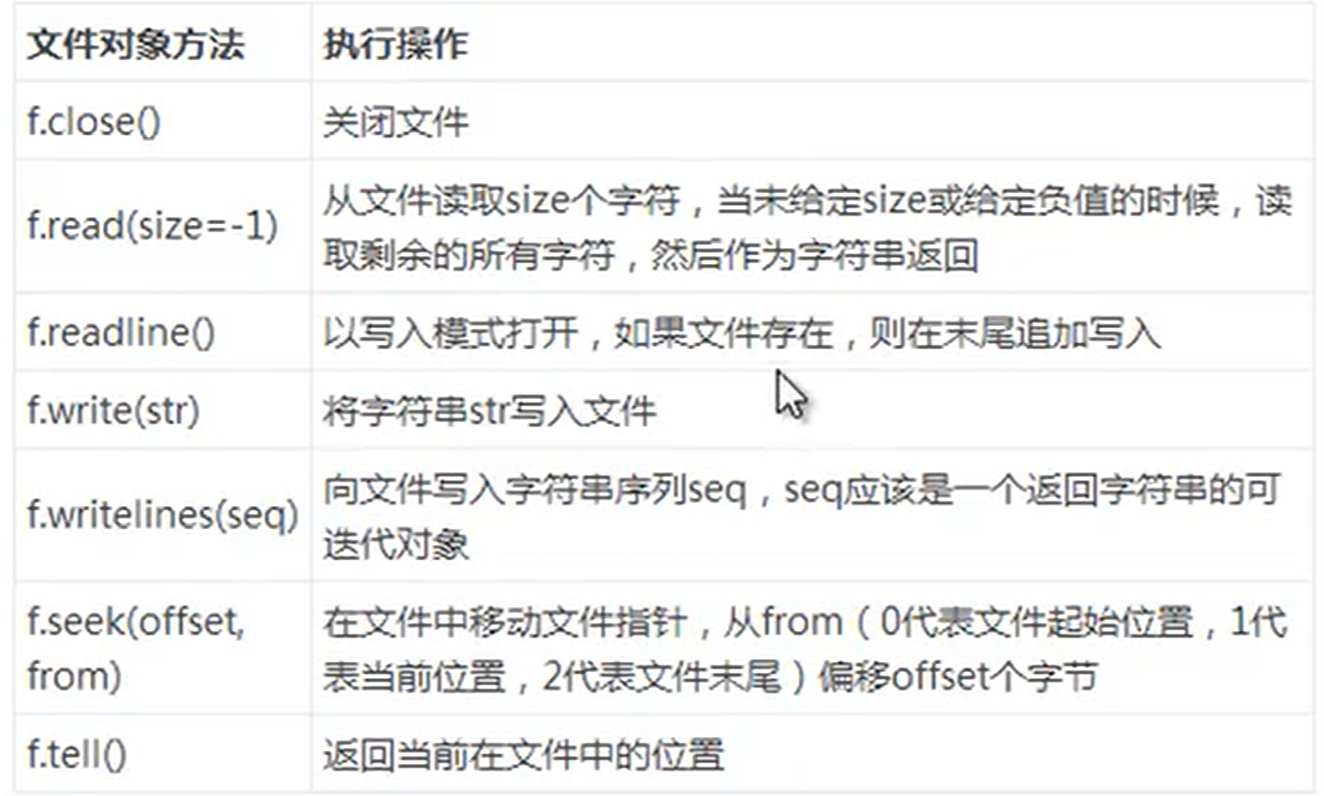
python打开文件的方法

open("文件路径","打开方式")

打开方式有：



使用文件的方法：



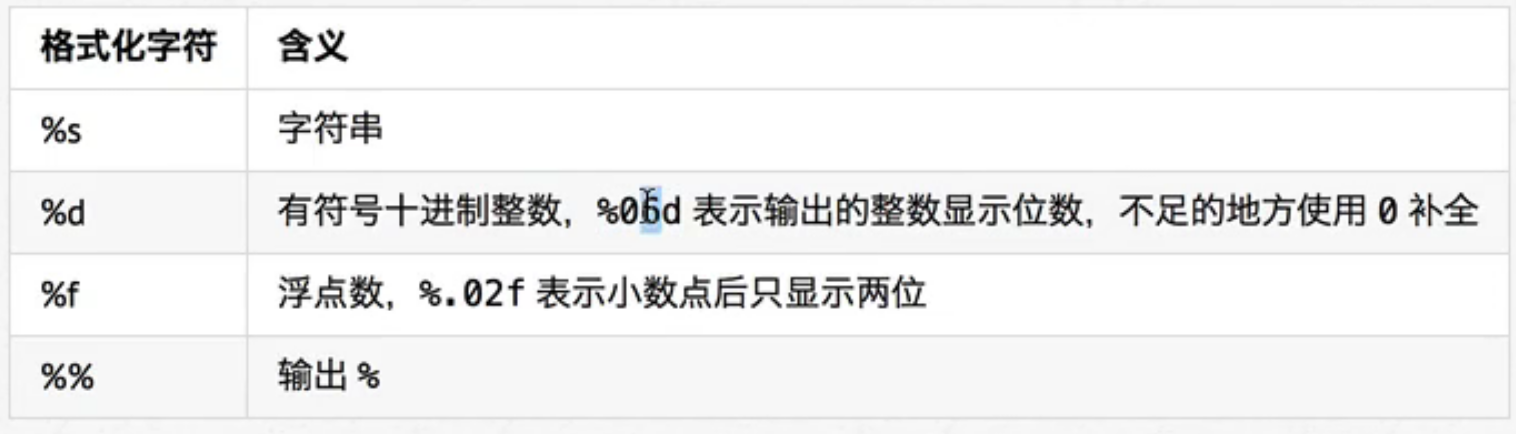
使用完文件需要及时关闭文件

小甲鱼第29章

读取多少数据，光标就停留在数据末尾，下一次提取就不在显示提取出来的数据

f.seek(0,0) 移动光标到文件第一行第一列

格式化输出



代码换行时要与运行结果隔一行

并且需要八个空格的缩进

代码中随机数的处理

Input random

导入random模块

random.randint(a,b) a,b为数值间距

循环

While

break停止循环

continue 跳过本次循环，继续下一次循环

continue 满足条件时，不会运行后面的代码，直接跳转到开头条件判断语句上

注意，在循环中，如果使用continue 关键字

需要确认循环的计数是否修改，否则循环条件会

一直满足，可能会导致死循环

循环嵌套

while循环里面在加一个while循环

**Python 程序运算符**



**函数定义**

定义行一般用 ：row

定义列一般用 ：col

Print(“”) 添加换行操作

**转义符**



\t 协助输出文本时垂直方向保持对齐 一般添加在输出语句中，也可以添加在end=”\t”中

**函数导入**

已经写过的代码不需要重新再写一遍，只需要将文件导入到需要使用文件中，这样可以大大减少程序的复杂性，使程序更加简便

使用方法：import 文件名

文件名 . 需要使用的函数

**函数的创建**

定义函数时不会运行其中的代码，只是起到封装的效果

必须使用函数才能看到程序运行的结果

**函数文档的注释**

在创建函数的下方设置一个多行注释”””注释内容”””，并且函数需要离上一行代码两个回车的距离，调用函数的时候选中函数，按快捷键Ctrl+Q可以看到函数的作用

**形参和实参**

形参：**定义**函数时，小括号中的参数，用来接收参数的，在函数内部作为**变量的使用**

实参：**调用**函数时，小括号中的参数，是用来把数据传递到**函数内部**用的

**函数的嵌套**

**更改需求时，不要轻易修改可以正常运行的代码，可以在原有的代码上添加新的需求代码**

**函数的字符类型**

**整型（int） 浮点型（float） 布尔型（bool）**

**真（True） 假（False）**

**非字符型**

# 列表（list）元组（tuple）字符串（str）字典（dictionary）

**列表的使用方法**

列表删除数据的方法

1. 列表名.pop(要删除的数据号码)默认删除列表最后一个参数
2. 列表名.clean（）删除所有数据
3. 列表名.remove(“要删除的数据”)如果有两个相同的字段，remove 删除第一个出现的字段，后面相同的字段不会被删除
4. del 列表名[列表索引]

del 本质上是从内存中删除变量

删除列表数据建议使用列表自带的删除方式

统计列表长度的关键字len

1. len(列表名)

len 不会统计两个相同的字段名

1. 想知道一个字段在列表中出现了几次

可以使用count 关键字

name = 列表名.count (”字段名”)

print(name) 就可以看到这个字段出现了多少次

列表的排序

**列表的升序**

sort 表示列表的升序

列表名.sort( ) 可以按数字的从小到大排序，也可以按英文字母排序

**列表降序 sort(reverse=True)**

列表名.sort(reverse=True) 可以让列表从大到小排序，也可以按英文字母从后往前排序

**列表逆转 reverse**

列表名.reverse() 将列表原有的排序颠倒过来

**元组的使用方法**

用于储存一串信息，数据之间用 , 隔开，元组用（）定义，元组的索引用 [ ] 定义

元组只包含一个数据时，需要在数据后面添加一个逗号

元组不可以修改

元组取索引：index

元组名.index [已知元组数据] 可以显示数据在元组的索引值

元组名 [索引值] 可以看到索引值对应的数据

统计数据出现在元组当中的次数

元组名.count(元组内的数据) 可以看到数据出现在元组当中出现的次数

**元组和列表之间的转换**

不希望别人修改列表当中的数据时，可以把列表转换为元组

转换方式为：元组名称 = tuple(列表名)

**字典的使用（dictionary）**

字典和列表的区别：列表为有序的集合

字典为无序的集合

字典的组成为：键 key 为索引

值 value 为数据

他们之间用 : 隔开

因为字典是无序的集合，使用print输出字典时显示的顺序通常与定义的顺序不相同

**字典的取值**

在取值的时候，如果取值的键不存在，程序会报错

字典名[键] 可以得到相应字典的值

**字典的增加/修改**

字典名[键] = 值

字典名[需要修改的键] = 修改的值

**字典的删除**

字典名.pop [键]

**合并字典**

原有的字典名 . update (需要合并的字典)

注意:在添加新的字典时如果键在原有的字典当中存在时，添加的新字典会覆盖原有的字典

**字符串的使用**

字符串用来判断



程序开发中一般使用isdecimal 来使用

字符串的查找和替换



使用replace 方法，会返回一个新的字符串

**注意：但是不会修改原有的字符串**

字符串的大小写转换



文本对齐



使用center 方法时，需要在括号内输入数字，表示对齐的间距，空格默认为英文输入法的空格，想要文本对齐可以将输入法中的半角改为全角

去除空白符



拆分和连接

