Python《从入门到实践》笔记

#print函数参数表：

print(\*objects, sep=’ ‘, end=’\n’)

关键在于sep和end：

sep：不同object之间的分隔符，默认为空格‘ ’.

end：最后一个object后面的结束符，默认为换行“\n”.

eg:

print(‘a’,’b’,’c’,sep=’-’,end=’\_’) 输出 a-b-c\_

#字符串str：不可变类型，仅返回新字符串，不修改原字符串。

<str>.rstrip()字符串删除右端的空白 lstrip左端的空白 strip两端的空白

<str1>.removeprefix(<str2>)字符串移除前缀

<str>.title()让首字母大写 lower全部小写 upper全部大写

~~<str1>.append(<str2>)~~字符串不可变，没有append方法

<str>.split(<sep>)根据字符将字符串切割成一个列表（注：’’也算列表元素）

<str>.count(<substr>)子串出现次数

<str>.replace(<old>,<new>)以旧换新

<str>.center(width[,fillchar])字符串居中，可选择填充符号

<str>.strip(chars=’ ’)删掉左侧和右侧chars列出的元素,无参数为删掉两边空格

<str>.join(<sep>)在每个字符之间插一个sep

n\*<str>或<str>\*n 返回n个字符串拼接的大字符串

chu(<unicode>)编码转字符 ord(<char>)字符转编码

#字符串格式化：

老版本格式化：”...{0:...}...”.format(<val>)

新版本格式化（f-string）: **f**”...{<val>:...}...”

以下皆采用新版本格式化。

格式： <content>:<填充><对齐><宽度> （注：对齐：<左>右^居中）

<number>:<,(千位分隔符)><.精度f><类型>

类型：

b:二进制

c:字符（unicode）

d:十进制

o:八进制

x:十六进制

X:大写的十六进制

e:小e的科学计数法

E:大E的科学计数法

f:以通常的非科学计数法形式表示

%:百分数形式

#列表list：

l=[]创建列表

list(<other>)创建列表

<list>.append(<val>)追加（只有列表有append方法）

<list>.insert(<Index>,<val>)在索引前面插入

del <list>[Index]删除索引位置元素，找不到抛出异常ValueError

<list>.pop(<Index>=-1)返回并去除索引位置元素(默认尾部)，找不到抛出异常ValueError

<list>.remove(<val>)删除指定元素

<list>.reverse()反转

len(<list>)大小

numbers=list(<other list>)拷贝

min(<list>)最小值 max最大值 sum求和

<list>[a:b] 切片：返回[a,b)的列表，可以写成[:b] [a:]的形式

<list>[a:b:c] 对[a,b)以c为步长切片

<list>[::-1] 翻转

<list>.index(x) 第一次出现x的索引，找不到会异常ValueError

<list>.index(x,i,j) 从i到j第一次出现x的索引，找不到会异常ValueError

<list>[i:j]=<otherList>将列表[i,j)部分替换为另一个列表

range(a,b)返回一个[a,b)的列表

range(a,b,c)返回一个[a,b)的步长为c的列表[a,a+c,a+2\*c,...]

#列表推导式：

<expression> for <item> in <iterable> (if <condition>，可选)

理解：顺序为2-3-1

如



等价于

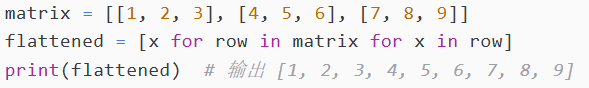


再比如

lst = [x for x in range(10) if x % 2 == 0]

注：1、里面expression可以是一个普通的计算式，也可以是三元运算符。

2、中间的for还可以是嵌套for，如：



3.可以利用列表推导式写一个矩阵，如写一个全是val的Row\*Col矩阵：

matrix=[[<val>]\*Col for \_ in range(Row)]

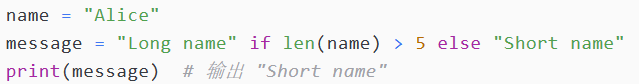
也可以用

matrix=[[<val> for i in range(Col)] for j in range(Row)]

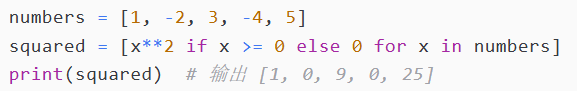
#条件表达式（三元操作符）:

<value\_if\_true> if <condition> else <value\_if\_false>

eg1:



eg2:

、

#Python的深浅拷贝：

浅拷贝: <list>[:]整个列表,也可以用<list>.copy()

深拷贝：可以import copy , <deepcopied>=copy.deepcopy(<original>)

Python的深拷贝和cpp的浅拷贝一样，生成一个**完全独立的副本**，和原列表互不影响；

Python的浅拷贝则是相当于cpp引用（但是不完全是）：

修改新对象中的**非嵌套的不可变对象**（如整数、字符串等）不会影响原对象。

修改新对象中的**可变对象**（如嵌套列表、字典等）会影响原对象，因为它们是同一个对象的引用。

eg: orig=[1,2,3,[4,5]],shacp=orig[:],则修改shacp[0]的时候不修改orig,

而修改shacp[3]则会影响原对象，因为浅拷贝将orig的**引用拷贝**给了shacp。

#元组tuple：没什么好讲的

#字典dict：

d={}创建字典（**不是set！**）

dict(<other>)创建字典

del <dict>[<key>]删除key以及对应的value，如果找不到抛出异常KeyError

<dict>.get(<key>,’No message’)返回对应值，如果键不存在就返回右边的值

<dict>.pop(<key>)返回并去除对应值，如果键不存在抛出异常KeyError

<dict>.pop(<key>,’No message’)返回并去除对应值，如果键不存在就返回右边的值

for循环遍历字典：

for k,v in <dict>.items():... items所有键值对,keys所有键,values所有值

<dict>.items()返回所有键值对的列表,keys所有键,values所有值

<dict>.popitem()随机返回并去除一个键值对，以**元组**形式返回

<dict>.update(<other>)将other的键值对更新到字典中（有则修改，无则添加）

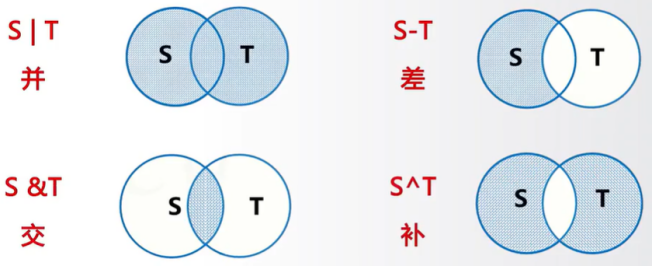
#集合set:

1.使用set(<iterable>)方法构造集合的时候，集合会对other进行**去重**，如：

S=set(“pypy123”) S={‘1’,’2’,’3’,’p’,’y’}

注：不是S={“pypy123”} !参数是一个序列，而不是一个元素！

2.集合间操作：



集合间对比：可以使用<=,<,>=,>,=等比较关系。

3.<set>.add(x)添加

<set>.discard(x)移除，如果找不到不报错

<set>.remove(x)移除，如果找不到抛出异常KeyError

<set>.clear()清空

<set>.pop()随机返回并去除一个元素，若空则异常KeyError

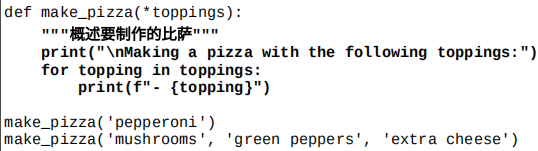
<set>.copy()拷贝

len(<set>)元素个数

#未知数量参数的传递：

1.传递未知数量的位置参数，可以采用\*<args>，参数被传递到一个**元组**中。

如：

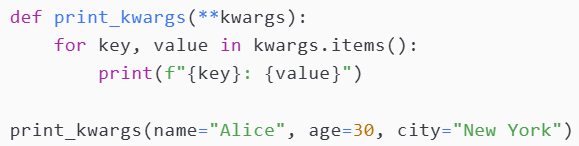


还可以将**列表或元组**解包(**\***)之后传递给未知数量形参，如：



2.传递未知数量的关键字参数（**键值对**），可以采用\*\*<kwargs>，参数被传递到一个**字典**中。

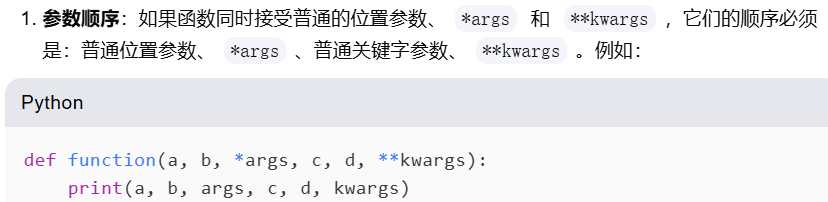
如：



也可以把一个**字典**解包(**\*\***)之后传递给函数：



注意：使用上述两种未知数量参数的顺序：



#Python算法处理函数：

(已知numbers=[1,2,5,3,6])

1.all（且）:对所有元素进行判断，都满足则返回True，反之返回False，

any（或）:对所有元素进行判断，有一个满足则返回True，反之返回False，

如result=all(x>0 for x in numbers)--->True

2.map(<func>,\*<iterable>)对所有元素进行处理，返回处理后的同类型容器**迭代器**，

如**list**(map(lambda x:x\*2,numbers))--->[2,4,10,6,12]

可放入多个可迭代对象。

3.<iterable>.sort(key=None,reverse=False)排序，**直接修改原序列**，默认升序，

key：排序的依据 reverse：False为升序，True为降序

eg:

people = [('Alice', 25), ('Bob', 20), ('Charlie', 30)]

people.sort(key=lambda x: x[1])

print(people)

# 输出[('Bob', 20), ('Alice', 25), ('Charlie', 30)]

<iterable>.sorted(key=None,reverse=False)排序，返回一个**可迭代对象（而不是迭代器）。**

4.filter(<**Pred**>,<iterable>)过滤，根据谓词返回的True决定保留，False丢弃，返回**迭代器**，

如**list**(filter(lambda x:x%2==0,numbers))--->[2,6]

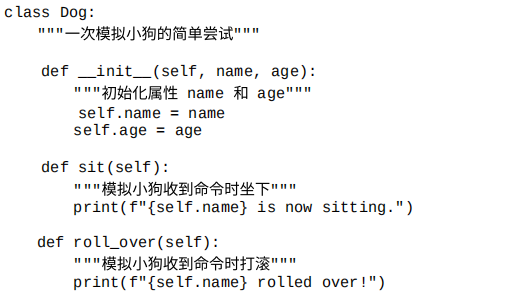
5.<sep>.join(<iterable>)将字符串序列连接sep连接成新字符串，必须要是字符串的序列，

eg:

str\_seq=[‘a’,’b’,’c’]

conc\_str=’\_’.join(str\_seq)--->’a\_b\_c’

#类：



注：

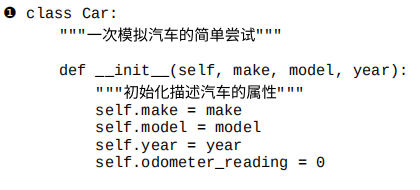
1.类的任何函数都要在参数表前写一个self.

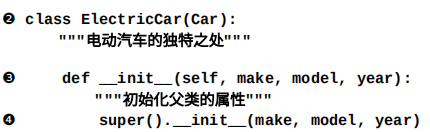
2.构造方法\_\_init\_\_(self,<inline args...>)

第二个参数及以后是类的成员变量初始值（不需要在类中显示声明）.

3.类的初始化方法：dog\_fact=Dog(‘Baffer’,18)

#继承：





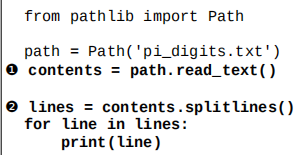
注：

1.继承写法：class <Derived>(<Base>).

2.super()函数：调用父类的初始化方法.

#使用pathlib库的文件处理：

（1）文件读取



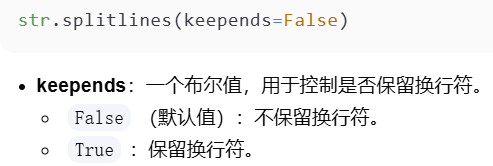
1.Path(<path>) 读取路径字符串返回一个路径类Path。

2.<Path>.read\_text()读取文件，返回冗长的字符串。

注：在显示文件路径的时候系统采用\，但是写路径字符串应该使用/。

<Path>.read\_text(encoding=’utf-8’)按照UTF-8编码读取。

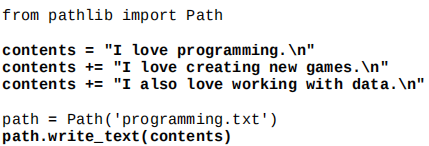
3.<str>.splitlines()将字符串按照’\n’切割，返回一个字符串的列表。



保留换行符的形式：

4.文件找不到：系统发送一个FileNotFoundError异常，可以选择pass。

（2）文件写入：



write\_text(<Path>)方法在文件中写入所有的content。

注意：如果Path路径的文件有内容，会被抹除重写。

（3）与内置的文件处理的转换：

with <Path>.open(‘a’) as f:

f.write(<content>)

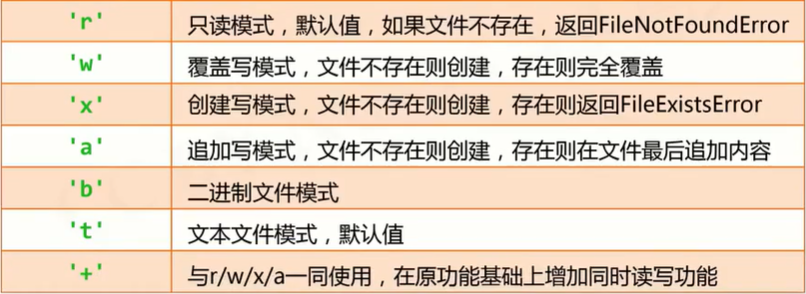
可以打开追加模式。

#普通的文件处理：

f=open(<path>,mode=’r’)打开路径的文件，（可以指定encoding=’utf-8’）

也可以使用with open(<path>,mode=’r’) as f:...

模式：



<f>.read(size=-1)读入全部，有参则读入size个字

<f>.readline(size=-1)读入一行内容，有参则读入这一行的前size个字

<f>.readlines(hint=-1)读入所有行，每一行为元素，形成一个列表，有参则读入前hint行

for line in <f>: 对文件进行逐行处理

<f>.write(s)写

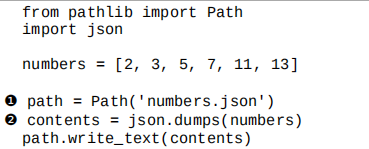
<f>.writelines(<strList>)将字符串列表写入一行

<f>.seek(offset)移动文件操作指针，0 -开头，1 -不动，2 -结尾

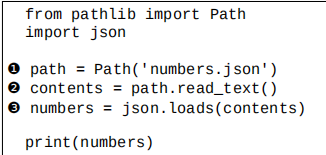
<f>.close()关闭

#json数据存储：

保存：



读取：

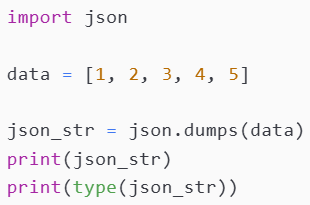


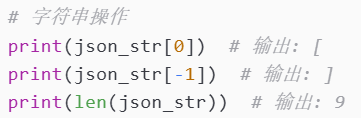
1.json.**dumps**(<content>)读取一个内容，返回一个**JSON字符串**。

对JSON字符串的理解：和普通字符串的类型是一样的，

和控制台输出的样子一模一样（带括号），但是是字符串。

eg：





2.json.**loads**(<json-str>)读取一个Json字符串，返回内容。

注：json文件的Path读取返回的就是一个JSON字符串。

3.if <Path>.exists():判断文件是否存在。

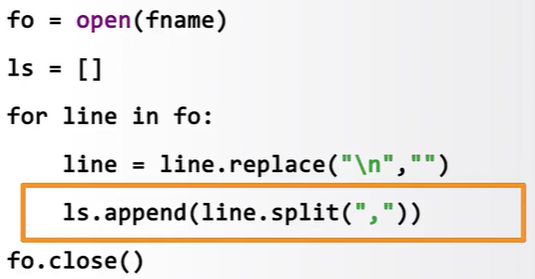
4.json相比txt多一个步骤：将要保存的数据使用json加载成json字符串

（读取时则是将json字符串加载为普通数据）。

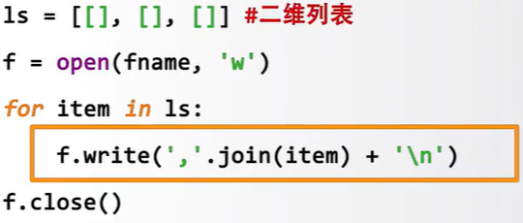
#CSV数据存储：

CSV格式：逗号分隔表示一维，按行分隔表示二维。

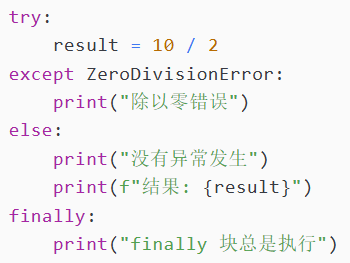
读取：对每一行处理



写入：

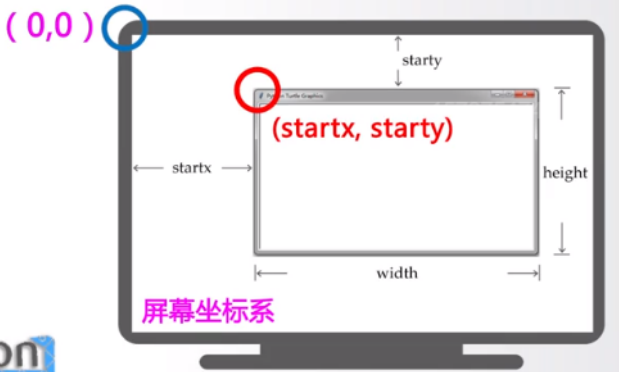


#异常处理：



#turtle库：

窗口坐标以屏幕左上角为原点，海龟绝对坐标以出发点（屏幕正中心）为原点。



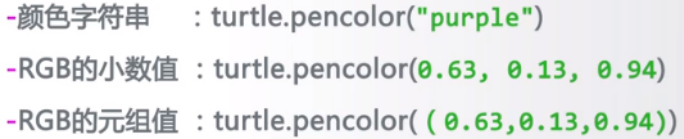
setup(width,height,startx=750,starty=650)初始化窗口(屏幕大概2000\*900)

title(title) 标题

goto(absX,absY)

penup() pendown()

pensize(width) or width(width)



speed(speed) speed=0最快

fd(dist) bk(dist)

circle(radius,angle=360)

seth(absAngle)转向到顺时针度数

left(angle) right(angle)

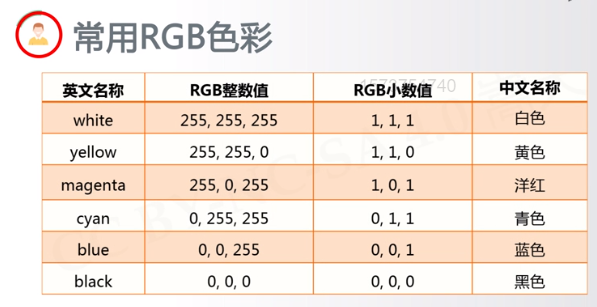
colormode(mode=1.0) mode:1.0-RGB小数模式 255-RGB整数值模式

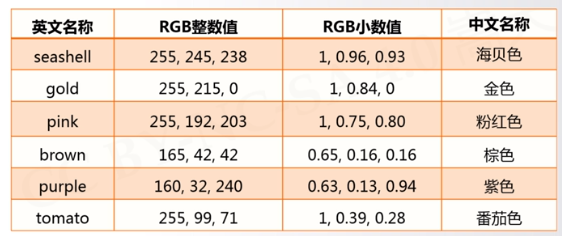
done()保持结束界面

填充：

color/pencolor/fillcolor---begin\_fill()---...---end\_fill()

hideturtle()隐藏海龟





#random库：

seed(num)产生随机数种子

random()生成一个[0,1)的随机小数

uniform(a,b)生成一个[a,b]之间的随机小数

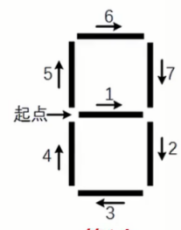
randint(a,b)生成一个[a,b]的随机整数

randrange(m,n[,k])生成一个[m,n)之间以k为步长的随机整数

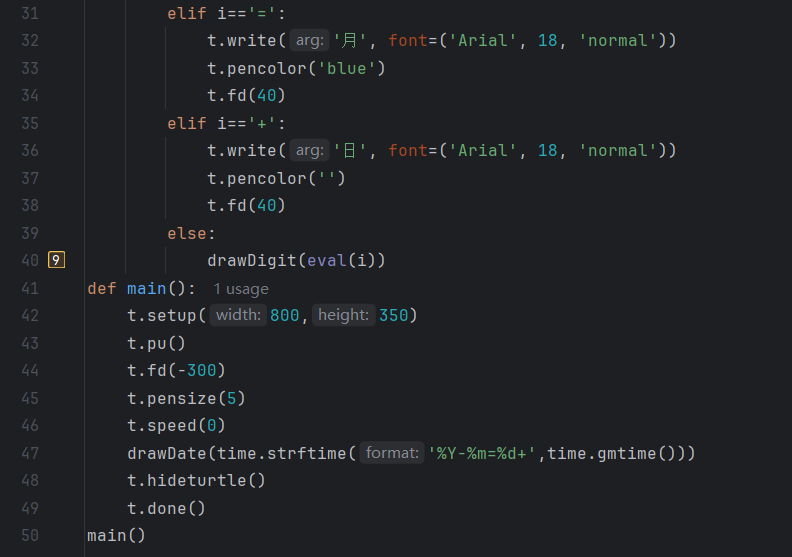
choice(<iterable>)从序列中随机抽取一个元素（<iterable>可以是列表，元组，字符串等）

shuffle(<iterable>)返回序列元素打乱后的序列

#七段数码管绘制（turtle库）:







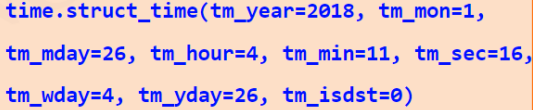
#time库：

四种时间类型：

<1>内部时间浮点数:1516939876.6022282

<2>易读时间:’Fri Jan 26 12:11:16 2018’

<3>时间结构体:



<4>格式化模版字符串:格式化前’%Y-%m-%d %H:%M:%S’---->格式化后’2025-01-26 12:55:20’

%Y年 %m月 %d日 %H时（24h） %M分 %S秒

%B月份名称 %b月份名称缩写 %A星期 %a星期缩写

%I时（12h） %p上/下午

time库中的函数：

获取时间：

time()

ctime()

gmtime()

时间格式化:

strftime(格式化前，结构体) 将提供的时间结构体格式化

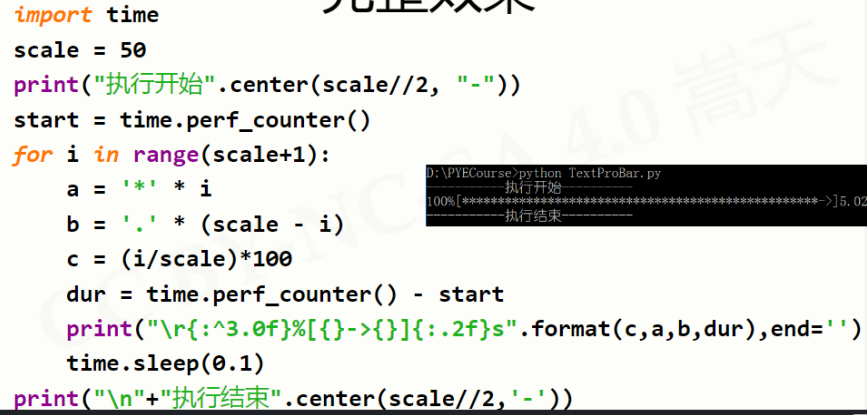
strptime(格式化后，格式化前) 将 格式化后的字符串 按照 格式化前的模版 转变为 时间结构体

程序计时器：

perf\_counter() 计时器

sleep(<sleepTime>) 程序拟休眠时间

#进度条：关键在于\r单行动态刷新：



#jieba库（第三方库）：

第三方库的下载：cmd输入：

pip install <library> -i https://pypi.tuna.tsinghua.edu.cn/simple

下载jieba库：

pip install jieba -i https://pypi.tuna.tsinghua.edu.cn/simple

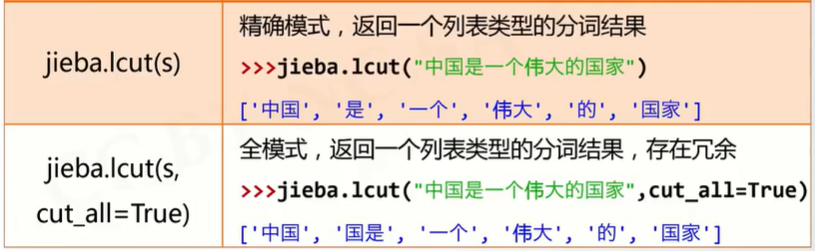
使用清华的镜像源（因为没有外网服务器），如下：

-i https://pypi.tuna.tsinghua.edu.cn/simple

也可以使用中科大的：

-i https://pypi.mirrors.ustc.edu.cn/simple

jieba的使用：





#wordcloud库（第三方库）：

下载wordcloud库：

pip install wordcloud -i https://pypi.tuna.tsinghua.edu.cn/simple some-package

wordcloud库常用用法：

import wordcloud as wc

w=wc.WordCloud

(

width=600,

height=400,

min\_font\_size=10,#最小字体

max\_font\_size=20,#最大字体

font\_step=1,#字号的步长

font\_path=None,#字体的路径，中文可采用’msyh.ttc’(微软雅黑)

max\_words=200,#最大单词数量

stop\_words=set(),#排除的词列表

mask=长方形,#指定词云形状，默认为长方形

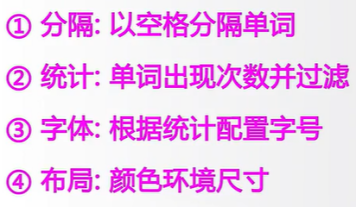
background\_color=’black’,#背景颜色

)创建WordCloud对象

w.generate(<txt>)加载文本

w.to\_file(<png/jpg>)将词云输出为图像文件，.png/.jpg格式

图像生成逻辑：



对于中文语句，则需要结合jieba库，lcut成为词云列表，然后join空格，变成一个大字符串，如下：



mask的特定形状指定：(好像不能用？)

from scipy.misc import imread

msk=imread(“fivestar.png”)

w=wc.WordCloud(mask=msk)

#os库：

（1）os.path子库：都是用\\

import os.path as op

op.abspath(<path>)返回绝对路径

op.normpath(<path>)将/、\等归一化为\\

op.relpath(<path>)返回相对路径

op.dirname(<path>)返回目录路径

op.basename(<path>)返回最后的文件名称

op.exists(<path>)判断文件是否存在

op.getatime(<path>)返回最近一次访问时间（浮点数）（access）

op.getmtime(<path>)返回最近一次修改时间（浮点数）（modify）

op.getctime(<path>)返回文件或目录的创建时间（易读时间）（create）

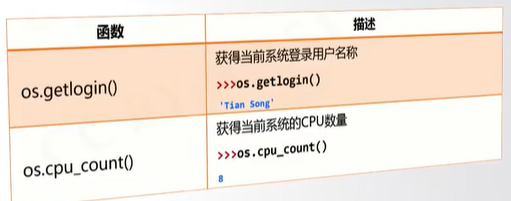
op.getsize(<path>)获得文件大小（B）

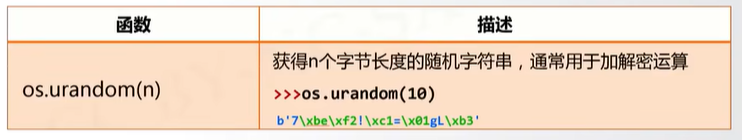
（2）os库：

os.system(<command>)在cmd中输入指令

这个函数使得Python可以调用电脑中的其他程序，成为一个**自动化脚本**。







实例：利用os自动安装第三方库：

