1. 组合数据类型

1、集合类型及操作

集合类型定义：

集合是多个元素的无序组合。

集合类型与数学中的集合概念一致。

集合元素之间无序，每个元素唯一，不存在相同元素。

集合元素不可更改，不能是可变数据类型。

集合用大括号{ }表示，元素间用逗号分隔。

建立集合类型用{ }或set（）

建立空集合类型，必须使用set（）。

集合操作符

集合间操作：

S|T 并运算

S-T 差运算

S&T 交运算

S^T 补运算

>=< 比较

6个操作符：

|  |  |
| --- | --- |
| 操作符及应用 | 描述 |
| S|T | 返回一个新集合，包括在集合S和T中的所有元素 |
| S-T | 返回一个新集合，包括在集合S但不在T中的元素 |
| S&T | 返回一个新集合，包括同时在集合S和T中的元素 |
| S^T | 返回一个新集合，包括集合S和T中的非相同元素 |
| S<=T或S<T | 返回True/False，判断S和T的子集关系。 |
| S>=T或S>T | 返回S和T的包含关系 |

4个增强操作符

|  |  |
| --- | --- |
| 操作符及应用 | 描述 |
| S |=T | 更新集合S，包括在集合S和T中的所有元素 |
| S-=T | 更新集合S，包括在集合S但不在T中的元素 |
| S&=T | 更新集合S，包括同时在集合S和T中的元素 |
| S ^=T | 更新集合S，包括集合S和T中的非相同元素 |

集合处理方法

|  |  |
| --- | --- |
| 操作函数或方法 | 描述 |
| S.add(x) | 如果x不在集合S中，将x增加到S |
| S.discard(x) | 移除S中元素x，如果x不在集合S中，不报错 |
| S.remove(x) | 移除S中元素x，如果x不在集合S中，产生keyError异常 |
| S.clear() | 移除S中所有元素 |
| S.pop() | 随机返回S的一个元素，更新S，若S为空产生KeyError异常。 |
| S.copy() | 返回集合S的一个副本 |
| Len(S) | 返回集合S的元素个数 |
| x in S | 判断S中元素x，x在集合S中，返回True，否则返回False |
| x not in S | 判断S中元素x，x不在集合S中，返回True，否则返回False |
| set(x) | 将其他类型变量x转变为集合类型 |

集合类型应用场景：

1、包含关系比较

2、数据去重：集合类型所有元素无重复。

1. 序列类型及操作

序列类型定义

序列是具有先后关系的一组元素。

类似于数学元素序列：S0，S1，Sn-1

元素间由序号引导，通过下标访问序列的特定元素。

序列是一个基类类型。



序号的定义：反向递减序号、正向递增序号

序列处理函数及方法

6个操作符

|  |  |
| --- | --- |
| 操作符及应用 | 描述 |
| x in s | 如果x是序列s的元素。返回True，否则返回False |
| x not in s | 如果x是序列s的元素。返回False，否则返回True |
| s+t | 连接两个序列s和t |
| s\*n或n\*s | 将序列s复制n次 |
| s[i] | 索引，返回s中的第i个元素，i是序列的序号 |
| s[i:j]或s[i:j:k] | 切片，返回序列s中第i到j以k为步长的元素子序列 |

5个函数和方法

|  |  |
| --- | --- |
| 函数和方法啊 | ，描述 |
| len(s) | 返回序列s的长度 |
| min(s) | 返回序列s的最小元素，s中元素需要可比较 |
| max(s) | 返回序列s的最大元素，s中元素需要可比较 |
| s.index(x)或s.index(x,i,j) | 返回序列s从i开始到j位置中第一次出现元素x的位置 |
| s.count(x) | 返回序列s中出现x的总次数 |

元组类型及操作

元组是一种序列类型，一旦创建就不能被修改。

使用小括号（）或tuple()创建，元素间用逗号，分隔

可以使用或不使用小括号。

元组继承序列类型的全部通用操作。

由于元组创建后不能修改，所以没有特殊操作。

列表类型及操作

列表是序列类型的一种扩展，创建后可以随意被修改。

使用方括号[]或list（）创建，元素间用逗号，隔开。

列表中各元素类型可以不同，无长度限制。

|  |  |
| --- | --- |
| 函数或方法 | 描述 |
| ls[i]=x | 替换列表ls第i元素为x |
| ls[i:j:k]=lt | 用列表lt替换ls切片后所对应元素子列表 |
| del ls[i] | 删除列表ls中第i元素 |
| del ls[i:j:k] | 删除列表ls中第i到第j‘以k为步长的元素 |
| ls +=lt | 更新列表ls，将列表lt元素增加到列表ls中 |
| ls \*=n | 更新列表ls，其元素重复n次 |
| ls.append(x) | 在列表ls最后增加一个元素x |
| ls.clear() | 删除列表ls中所有元素 |
| ls.copy() | 生成一个新列表，赋值ls中所有元素 |
| ls.insert(i,x) | 在列表ls的第i位置增加元素x |
| ls.pop(i) | 将列表ls中第i位置元素取出并删除该元素。 |
| ls.remove(x) | 将列表ls中出现的第一个元素x删除 |
| ls.reverse() | 将列表ls中的元素反转 |

定义空列表： lt=[ ]

向lt新增5个元素：lt += [1,2,3,4,5]

修改lt中第2个元素：lt[2]=6

向lt中第2个位置增加一个元素：lt.insert(2,7)

从lt中第1个位置删除一个元素：del lt[1]

删除lt中第1-3位置元素：del lt[1:4]

判断lt中是否包含数字0：0 in lt

向lt新增数字0：lt.append(0)

返回数字0所在lt中的索引：lt.index(0)

lt的长度：len(lt)

lt中最大元素：max(lt)

清空lt：lt.clear()

序列类型应用场景

1. 元组用于元素不改变的应用场景，更多用于固定搭配场景。

列表更加灵活，它是最常用的序列类型

最主要作用：表示一组有序数据，进而操作它们。

1. 元素遍历

for item in ls:

<语句块>

1. 数据保护。

若不希望数据被程序所改变，转换成元组类型。

实例：基本统计值计算

需求：给出一组数，对它们有个概要理解。

例如：总个数、求和、平均值、方差、中位数………

获取用户输入的数字：

#jibentongjizhi.py

def getNum():

nums=[]

iNumStr=input("请输入数字(回车退出):")

while iNumStr !="":

num.append(eval(iNumStr))

iNumStr=input("请输入数字(回车退出):")

return nums

求平均值：

#计算平均值

def mean(numbers)

s=0.0

for num in numbers:

s=s+num

return s/len(numbers)

#计算方差

def dev(numbers,mean):

sdev=0.0

for num in numbers:

sdev=sdev+(num-mean)\*\*2

return pow(sdev/(len(numbers)-1),0.5)

#计算中位数

def median(numbers):

sorted(numbers)

size=len(numbers)

if size % 2==0:

med=(numbers[size//2-1]+numbers[size//2])/2

else:

med=number[size//2]

return med

#jibentongjizhi.py

#获取用户不定长度的输入

def getNum():

nums=[]

iNumStr=input("请输入数字(回车退出):")

while iNumStr !="":

nums.append(eval(iNumStr))

iNumStr=input("请输入数字(回车退出):")

return nums

#计算平均值

def mean(numbers):

s = 0.0

for num in numbers:

s=s + num

return s / len(numbers)

#计算方差

def dev(numbers,mean):

sdev=0.0

for num in numbers:

sdev=sdev + (num - mean)\*\*2

return pow(sdev/(len(numbers)-1),0.5)

有bug

#计算中位数

def median(numbers):

sorted(numbers)

size=len(numbers)

if size % 2==0:

med=(numbers[size//2-1]+numbers[size//2])/2

else:

med=numbers[size//2]

return med

n=getNum()

m=mean(n)

print("平均值:{},方差:{:.2},中位数:{}.".format(m,dev(n,m),median(n)))

字典类型及操作

字典类型定义：

映射：是一种键（索引）和值（数据）的对应。

字典类型是“映射”的体现。

键值对：键是数据索引的扩展。

字典是键值对的集合，键值对之间无序。

采用大括号{}和dict()创建，键值对用冒号：表示

在字典变量中，通过键获得值

<字典变量>={<键1>:<值1>,…,<键n>:<值n>}

字典处理函数及方法

|  |  |
| --- | --- |
| 函数或方法 | 描述 |
| del d[k] | 删除字典d中键k对应的数据值 |
| k in d | 判断键k是否在字典d中，如果在返回True，否则False |
| d.keys() | 返回字典d中所有的键信息 |
| d.values() | 返回字典d中所有的值信息 |
| d.items() | 返回字典d中所有的键值对信息 |
| d.get(k,<default>) | 键k存在，则返回相应值·，不在则返回<default>值 |
| d.pop(k,<default>) | 键k存在，则取出相应值·，不在则返回<default>值 |
| d.popitem() | 随机从字典d中取出一个键值对，以元组形式返回 |
| d.clear() | 删除所有的键值对 |

定义空字典： d={ }

向d新增2个键值对元素：d[“a”]=1;d[“b”]=2

修改第2个元素：d[“b”]=3

判断字符“C”是否在d的键：”c” in d

计算d的长度：len(d)

清空d：d.clear()

字典类型应用场景：

1. 映射的表达
2. 元素遍历

模块:jieba库的使用

Jieba库基本介绍：

Jieba是优秀的中文分词第三方库，需要额外安装

中文文本需要通过分词获得单个的词语

Jieba库提供三种分词模式，最简单只需掌握一个函数。

（cmd命令行） pip install jieba

Jieba分词依靠中文分词。

利用一个中文词库，确定汉字之间的关联概率。

汉字间概率大的组成词组，形成分词结果。

除了分词，用户还可以添加自定义的词组。

Jieba库使用说明：

三种模式：精确模式、全模式、搜索引擎模式

1、精确模式：把文本精确的切开，不存在冗余单词。最常用的一种单词

2、全模式：把文本中所有可能的词语都扫描出来，有冗余

1. 搜索引擎模式：在精确模式基础上，对长词再次切分。

|  |  |
| --- | --- |
| 函数 | 描述 |
| jieba.lcut(s) | 精确模式，返回一个列表类型的分词结果 |
| jieba.lcut(s,cut\_all=True) | 全模式，返回一个列表类型的分词结果，存在冗余 |
| jieba.lcut\_for\_search(s) | 搜索引擎模式，返回一个列表类型的分词结果，存在冗余 |
| jieba.add\_word(w) | 向分词词典增加新词w |

实例：文本词频统计  
需求：一篇文章，出现了哪些词汇？哪些词出现的最多？

英文文本：Hamet 分析词频

<http://python123.io/resources/pye/hamlet.text>

代码：

#hamlet.py

def getText():

txt=open("hamlet").read()

txt=txt.lower()

for ch in '|"#$%&()\*+,-./:;<=>?@[\\]^\_‘{|}~':

txt=txt.replace(ch," ")

return txt

hamleTxt=getText()

words=hamleTxt.split()

counts={}

for word in words:

counts[word]=counts.get(word,0)+1

items=list(counts.item())

items.sort(key=lambda x:x[i],reverse=True)

print("{0:<10}{1:>5}".format(word,count))

中文文本：《三国演义》分析人物

<http://python123.io/resources/pye/threekingdoms.txt>

例1：

#threekingdoms1.py

import jieba

txt=open("threekingdoms1.txt","r",encoding="utf-8").read()

words=jieba.lcut(txt)

counts={}

for word in words:

if len(word)==1:

continue

else:

counts[word]=counts.get(word,0)+1

items=list(counts.item())

items.sort(key=lambda x:x[1],reverse=True)

for i in range(15):

word,count=items[i]

print("{0:<10}{1:>5}".format(word,count))

例2：修改后

#threekingdoms2.py

import jieba

txt=open("threekingdoms1.txt","r",encoding="utf-8").read()

excludes={"将军","却说","荆州","二人","不可","不能","如此"}

words=jieba.lcut(txt)

counts={}

for word in words:

if len(word)==1:

continue

elif word=="诸葛亮" or word=="孔明曰":

rword="孔明"

elif word=="关公" or word=="云长曰":

rword="关羽"

elif word=="玄德" or word=="玄德曰":

rword="刘备"

elif word=="孟德" or word=="丞相曰":

rword="曹操"

else:

rword=word

counts[rword]=counts.get(rword,0)+1

for word in excludes:

del counts[word]

items=list(counts.items())

items.sort(key=lambda x:x[1],reverse=True)

for i in range(10):

word,count=items[i]

print("{0:<10}{1:>5}".format(word,count))