学习目标：

1. 了解3类基本组合数据类型
2. 理解列表概念并掌握Python中列表的使用
3. 理解字典概念并掌握Python中字典的使用
4. 运用列表管理采集的信息，构建数据结构
5. 运用字典处理复杂的数据信息
6. 运用组合数据类型进行文本词频统计

第六章 组合数据类型

目录

CONTENTS

6.1 组合数据类型概述

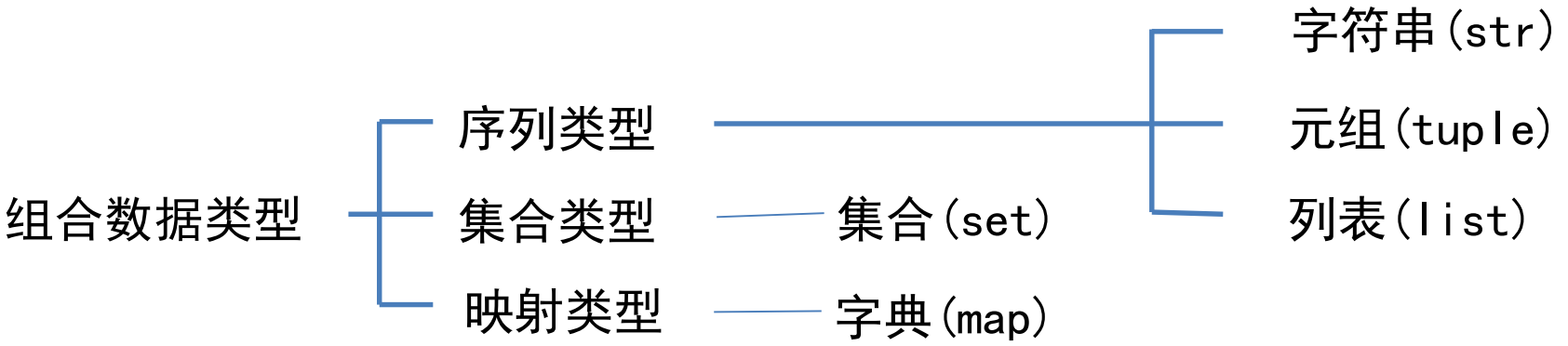
给定一组单词{python，data，function，list，loop}，统计每个单词的长度
是使用word1,word2,word3,word4,word5还是用word[1]-word[5]好？
后者就是组合数据类型。

Python中的组合数据类型分为**序列类型、集合类型、映射类型**

序列类型：元素的向量，元素之间存在先后顺序，通过序号访问；

集合类型：元素的集合，元素之间无序相同元素在集合中唯一存在；

映射类型：“键-值”数据项的组合，每个元素是一个（key,value）键值对。



6.1 组合数据类型概述

6.1.1 序列类型

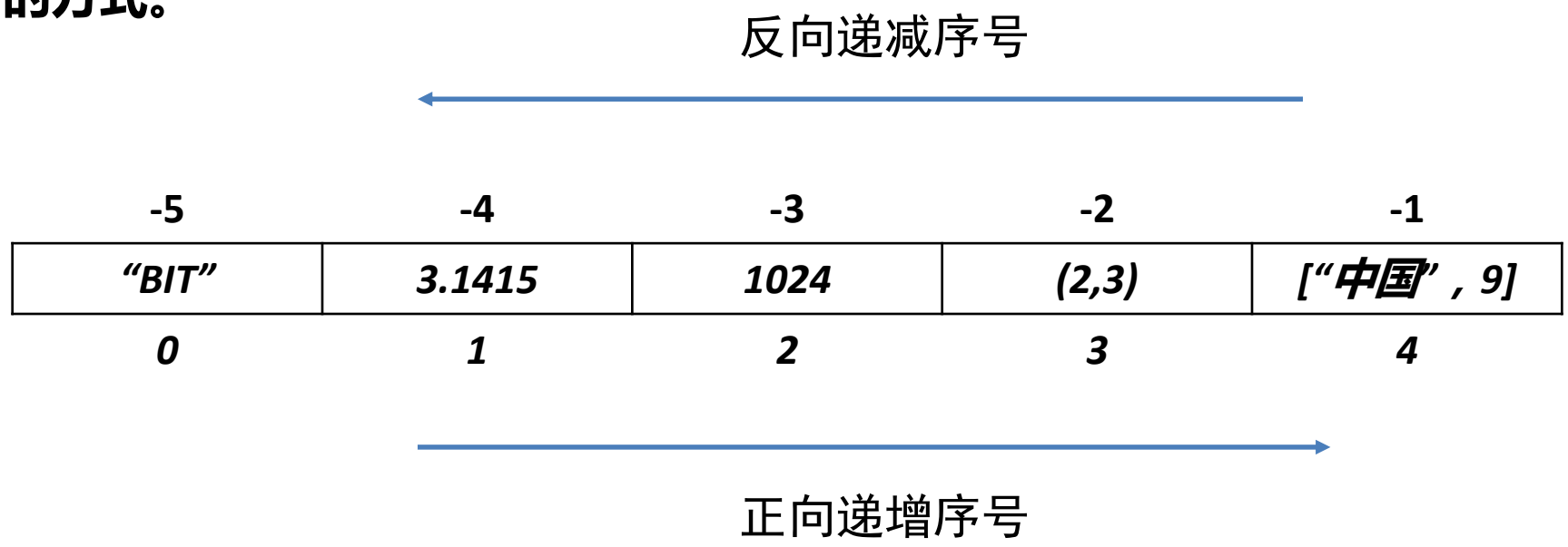
常用的序列类型有字符串str、元组tuple和列表list：

字符串：单个字符的有序组合；

元组：元组生成后是固定的，其中任何数据项不能替换或删除；

列表：是可以改变的序列类型。

所有的序列类型可以通过变量名加标号的形式访问，标号可以采用正向序号和反向序号的方式。



6.1 组合数据类型概述

通用操作符和函数：

操作符或函数	描述
x in s	如果x是s的元素，返回True
x not in s	如果x是s的元素，返回False
s+t	连接s和t
s*n或n*s	将序列s复制n次
s[i]	返回序列s的第i+1个元素
s[i:j]	分片，返回序列s中从i到j-1个元素的子序列
s[i:j:k]	返回包含s第i到j个元素以k为步长的子序列
len(s)	序列s的元素个数
min(s)	序列s中的最小元素
max(s)	序列s中的最大元素
s.index(x[,i[,j]])	序列s中从i开始到j位置中第一次出现元素x的位置
s.count(x)	序列s出现x的总次数

元组类型

实例：

```
>>> creature="cat","dog","tiger","human"
>>> creature
('cat', 'dog', 'tiger', 'human')
>>> color=("red","0x001100","blue",creature)
>>> color
('red', '0x001100', 'blue', ('cat', 'dog', 'tiger', 'human'))
>>> color[2]
'blue'
>>> color[-1][2]
'tiger'
>>>
```

元组需要用逗号将元素分开，可以用圆括号把所有元素括起来，但不是必须的，一个元组可以作为另外一个元组的元素，并采用多级索引访问。

6.1 组合数据类型概述

6.1.2 集合类型

- 集合中的元素不可重复，元素类型只能是固定的数据类型；
- 由于集合是无序组合，没有索引和位置的概念，不能分片，可以动态的增加和删除；
- 集合用大括号表示，可以用赋值语句生成一个集合；
- set(x)函数可以生成集合。

```
>>> s={425,"BIT",(10,"CS"),424}
>>> s
{424, 425, (10, 'CS'), 'BIT'}
>>> t={425,"BIT",(10,"CS"),424,425,"BIT"}
>>> t
{424, 425, (10, 'CS'), 'BIT'}
>>> w=set("apple")
>>> w
{'l', 'a', 'e', 'p'}
>>> v=set(("cat","dog","tiger","human"))
>>> v
{'human', 'dog', 'tiger', 'cat'}
```

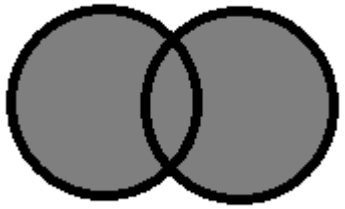
6.1 组合数据类型概述

集合类型的操作符

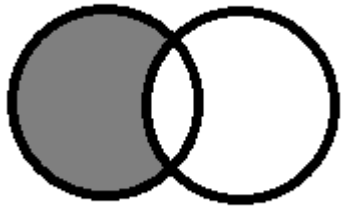
操作符	描述	定义
s-t或s.difference(t)	返回新集合，两集合的差	差集
s-=t或s.diffrence_update(t)	更新集合s,包括两集合的差的元素	
s&t或s.intersection(t)	返回新集合，两集合的交集	交集
s&=t或s.intersection_update(t)	更新集合s，包括两集合的交集	
s^t或s.symmetric_difference(t)	返回新集合，两集合的并集减去两集合的交集	对称差
s^-t或s.symmetric_difference_update(t)	更新集合s，包括两集合的并集减去两集合的交集	
s|t或s.union(t)	返回新集合，两集合的并集	并集
s|=t或s.update(t)	更新集合s，包括两集合的并集	
s<=t或s.issubset(t)	当s是t的子集时，返回True	补集
s>=t或s.isupperset(t)	当s是t的超集时，返回True	

6.1 组合数据类型概述

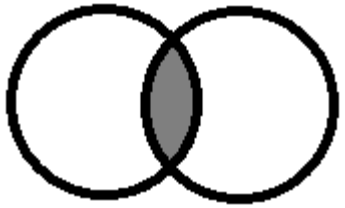
集合类型的操作符



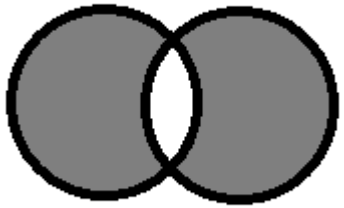
s|t



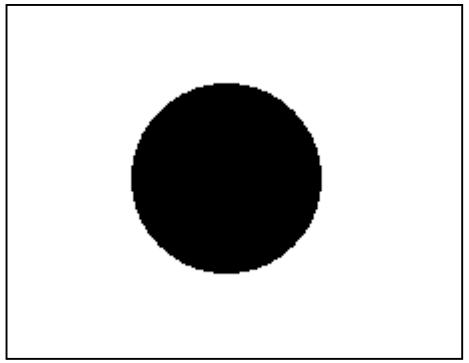
s-t



s&t



s^t



补集

6.1 组合数据类型概述

集合类型的操作函数或方法

操作函数或方法	描述
s.add(x)	如果数据项x不在集合中，将x增加到s
s.clear()	移除s中的所有数据项
s.copy()	返回集合s的一个副本
s.pop()	随机返回s中的一个元素，如果s为空，产生KeyError异常
s.discard(x)	如果x在集合s中，移除，如果不s中，不报错
s.remove(x)	如果x在集合s中，移除，如果不s中，报KeyError异常
s.isdisjoint(t)	如果s和t没有相同元素，返回True
len(s)	返回集合s的元素个数
x in s	如果x 是s的元素，返回True，否则，返回False
x not in s	如果x不是s的元素，返回True，否则，返回False

6.1 组合数据类型概述

6.1.3 映射类型

- 映射类型是“键-值”数据项的组合，一个元素是一个键值对(key，value)，元素之间是无序的。其中键表示一个属性；值是属性的内容，描述对应属性的取值。
- 如：
- （学校名称，城市）为映射类型，则有如下映射实例：

（北京大学，北京）
（清华大学，北京）
（浙江大学，杭州）
（复旦大学，上海）
（中国人民大学，北京）
（南京大学，南京）
……

6.2 列表类型和操作

列表类型的概念

- 列表是包含0个或多个对象的有序序列，但列表的长度和内容是可变的，可以自由的进行数据项进行增加，删除或替换，列表没有长度限制，元素类型可以不同
- 列表也支持成员关系运算符in、长度计算函数len()、分片[]等序列类型的运算，方法和函数。
- 列表用中括号[]表示,也可以通过list()函数将元组或字符串转化成列表。

```
ls1=[425,'BIT',[10,'CS'],425]
print(ls1)
print(ls1[2][-1][0])
ls2=list((425,'BIT',[10,'CS'],425))
print(ls2)
ls3=list('中国是个伟大的国家')
print(ls3)
```

```
[425, 'BIT', [10, 'CS'], 425]
C
[425, 'BIT', [10, 'CS'], 425]
['中', '国', '是', '个', '伟', '大', '的', '国', '家']
```

6.2 列表类型和操作

列表类型的操作

函数或方法	描述	函数或方法	描述
ls[i]=x	替换列表ls第i个数据项为x	ls.append(x)	在列表的最后增加一个元素x
ls[i:j]=lt	用列表lt替换列表ls中第i到j数据项（不包含j项）	ls.clear()	删除列表ls的所有数据项
ls[i:j:k]=lt	用列表lt替换ls中第i到j项以k为步长的数据	ls.copy()	生成新列表，赋值ls中的所有元素
del ls[i:j]	删除ls中i到j项数据，等价于ls[i:j]=[]	ls.insert(i,x)	在列表ls的第i位置增加元素x
del ls[i:j:k]	删除列表ls中i到j以k为步长的数据项	ls.pop(i)	讲列表ls的第i个元素取出并删除该元素
ls+=lt或ls.extend(lt)	将列表lt元素增加到列表ls中	ls.remove(x)	讲列表中出现的第一个元素x删除
ls*=n	更新列表ls，其元素重复n次	ls.reverse(x)	列表ls中的元素反转

6.2 列表类型和操作

列表的操作实例如下：

```
vlist=list(range(5))
print(vlist)
print(len(vlist[2:]))
print(2 in vlist)
vlist[3]="python"
print(vlist)
vlist[1:3]=["bit","computer"]
print(vlist)
```

运行结果：

```
[0, 1, 2, 3, 4]
3
True
[0, 1, 2, 'python', 4]
[0, 'bit', 'computer', 'python', 4]
```

当使用一个列表改变另外一个列表值时，python不要求两个列表长度一样，但遵循“多增少减”的原则，如：

```
>>>vlist[1:3]=[“new_bit”,” new_computer”,123]
>>>vlist
[0, “new_bit”,” new_computer”,123,” python”,4]
>>>vlist[1:3]=[ “fewer” ]
>>>vlist
[0,” fewer”,123,” python”,4]
```

6.2 列表类型和操作

与元组一样，列表可以通过for-in语句对元素进行遍历，基本格式：

```
for <变量> in <列表>
    <语句块>
```

如：

```
for e in vlist:
    print(e,end=“ “)
```

输出结果为：

```
0 fewer 123 python 4
```

6.3 实例9：基本统计值计算

Python的列表数据结构能够支持基本的数据统计应用，本实例求解一组不定长数据的基本统计值：平均值，标准差，中位数

平均值：$m = (\sum_{i}^{n-1} S_i)/n$

标准差：$d = \sqrt{(\sum_{i=0}^{n-1}(S_i - m)^2/(n-1)}$

中位数：S中所有数按照从小到大顺序排列后，处于中间位置的数据，如果n为奇数，则序列中间位置的数据为中位数，如果n为偶数，则中位数表示为最中间两个位置数据的平均数，即$(S_{\frac{n}{2}-1} + S_{\frac{n}{2}})/2$

6.3 实例9：基本统计值计算

```
#e9.1CallStatitistics.py
from math import sqrt
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（直回车退出）：")
    return nums
def mean(numbers):
    s=0.0
    for num in numbers:
      s=s+num
    return s/len(numbers)
def dev(numbers,mean):
    sdev=0.0
    for num in numbers:
    sdev=sdev+(num-mean)**2
    return sqrt(sdev/(len(numbers)-1))
```

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

getNum():输入系列数值函数；
mean():计算平均数函数；
dev():计算方差函数；
median():计算中位数函数

6.3 实例9：基本统计值计算

```
#e9.1CallStatitistics.py
from math import sqrt
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（直回车退出）：")
    return nums
def mean(numbers):
    s=0.0
    for num in numbers:
        s=s+num
    return s/len(numbers)
def dev(numbers,mean):
    sdev=0.0
    for num in numbers:
        sdev=sdev+(num-mean)**2
    return sqrt(sdev/(len(numbers)-1))
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("平均数：{}，方差：{:.2}，中位数：{}.".format(m,dev(n,m),median(n)))
```

- getNum():输入系列数值函数；
- nums为列表，用于存储输入的系列数值；
- 当输入为非回车时，利用nums.append()函数加入到nums中；
- 通过return nums语句返回生成的列表；

6.3 实例9：基本统计值计算

```
#e9.1CallStatitistics.py
from math import sqrt
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（直回车退出）：")
    return nums
def mean(numbers):
    s=0.0
    for num in numbers:
        s=s+num
    return s/len(numbers)
def dev(numbers,mean):
    sdev=0.0
    for num in numbers:
        sdev=sdev+(num-mean)**2
    return sqrt(sdev/(len(numbers)-1))
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("平均数：{}，方差：{:.2}，中位数：{}.".format(m,dev(n,m),median(n)))
```

- mean():求列表的平均值；
- numbers为形式参数，接受列表n为实际参数；
- s为列表中所有元素的总和；
- 利用return语句返回平均数；

6.3 实例9：基本统计值计算

```
#e9.1CallStatitistics.py
from math import sqrt
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（直回车退出）：")
    return nums
def mean(numbers):
    s=0.0
    for num in numbers:
        s=s+num
    return s/len(numbers)
def dev(numbers,mean):
    sdev=0.0
    for num in numbers:
        sdev=sdev+(num-mean)**2
    return sqrt(sdev/(len(numbers)-1))
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("平均数：{}，方差：{:.2}，中位数：{}.".format(m,dev(n,m),median(n)))
```

- dev():求列表的方差（每个元素偏离平均值的程度）；
- numbers和mean为形式参数，分别对应列表n和列表的平均值m；
- sdev存储$(\sum_{i=0}^{n-1}(S_i-m)^2$；
- sqrt(sdev/(len(numbers)-1)) 等价于

$$\sqrt{(\sum_{i=0}^{n-1}(S_i-m)^2/(n-1)}$$；

6.3 实例9：基本统计值计算

```
#e9.1CallStatitistics.py
from math import sqrt
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（直回车退出）：")
    return nums
def mean(numbers):
    s=0.0
    for num in numbers:
        s=s+num
    return s/len(numbers)
def dev(numbers,mean):
    sdev=0.0
    for num in numbers:
        sdev=sdev+(num-mean)**2
    return sqrt(sdev/(len(numbers)-1))
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("平均数：{}，方差：{:.2}，中位数：{}.".format(m,dev(n,m),median(n)))
```

- median():求列表的中位数；
- Sorted()函数对numbers列表排序；
- 如果列表个数为偶数，中位数值为$(S_{\frac{n}{2}-1}+S_{\frac{n}{2}})/2$，如果为奇数$S_{\frac{n}{2}}$；

实验6.1 有30名同学的成绩存储在列表list1中：67,82,87,80,78,59,46,70,60,66,71,55,42,72,63,65,68,80,67,73,60,89,74,82,74,65,74,63,73,79
统计本班同学的平均分、最高分、最低分、方差。

实验6.2 随机生成有100个字符组成的字符串，统计每个字符出现的次数，counts列表存储26个字符的出现次数。

通用序列操作

https://docs.python.org/zh-cn/3/library/index.html

运算	结果
`x in s`	如果 *s* 中的某项等于 *x* 则结果为 `True`，否则为 `False`
`x not in s`	如果 *s* 中的某项等于 *x* 则结果为 `False`，否则为 `True`
`s + t`	*s* 与 *t* 相拼接
`s * n` 或 `n * s`	相当于 *s* 与自身进行 *n* 次拼接
`s[i]`	*s* 的第 *i* 项，起始为 0
`s[i:j]`	*s* 从 *i* 到 *j* 的切片
`s[i:j:k]`	*s* 从 *i* 到 *j* 步长为 *k* 的切片
`len(s)`	*s* 的长度
`min(s)`	*s* 的最小项
`max(s)`	*s* 的最大项
`s.index(x[, i[, j]])`	*x* 在 *s* 中首次出现项的索引号（索引号在 *i* 或其后且在 *j* 之前）
`s.count(x)`	*x* 在 *s* 中出现的总次数

6.4 字典类型和操作

字典类型的概念

- 在有些比较灵活的信息查询，如查找学生的“C语言成绩”时，我们需要输入学号，而不是输入该成绩在列表中的索引号（下标），这种的查询需要“键值对”，即通过特定的键（如学号），访问值（如C语言成绩）；
- 通过任意键值查找一组数据中信息的过程称为映射，在Python中通过字典实现映射，通过大括号{}建立映射。建立模式如下：
- {<键1>:<值1>，{<键2>:<值2>…,{<键n>:<值n>}
- 由于字典顺序也是一种集合类型，所以键值对之间没有顺序并且不能重复。

```
dcountry={'中国':'北京','美国':'华盛顿','法国':'巴黎'}
print(dcountry)
dict={'one':1,'two':2,'three':3,'four':2}
print(dict)
dict2={'one':1,'two':2,'three':3, 'two':4}
```

输出为：
```
{'中国': '北京', '美国': '华盛顿', '法国': '巴黎'}
{'one': 1, 'two': 2, 'three': 3, 'four': 2}
{'one': 1, 'two': 4, 'three': 3}
```

- 对键值的修改通过中括号和赋值完成,如：dcountry[‘中国’]= ‘大北京’
- 字典对键值的访问格式: <值>=<字典变量>.[<键>]

6.4 字典类型和操作

字典类型的操作

- 字典的创建：利用大括号创建字典，并指定初始值；
- 通过中括号增加新的元素；

如：

dcountry={'中国':'北京','美国':'华盛顿','法国':'巴黎'}

dcountry[‘英国’]=‘伦敦‘

字典的函数和方法如下表：

如果希望keys(),values()等方法返回列表类型，可以采用list()函数转换成列表：

```
>>>dcountry.keys()
dict_keys(['中国', '美国', '法国'])
>>>list(dcountry.values())
['北京', '华盛顿', '巴黎']
```

函数和方法	描述	函数和方法	描述
d.keys()	返回所有的键	d.popitem()	随机取一个键值对，以元组形式返回
d.values()	返回所有的值	d.clear()	删除所有键值对
d.items()	返回所有的键值对	del d.key	删除字典中的某个键值对
d.get(key,default)	键存在则返回对应值，否则返回默认值	key in d	如果键存在，则返回True，否则返回False
d.pop(key,default)	键存在则返回对应值，同时删除键值对，否则返回默认值		

6.4 字典类型和操作

字典类型的操作

与其它组合类型类似，字典可以通过for-in语句对其元素进行遍历：

```
for <变量名> in <字典名>:
        <语句块>
```

为了更好的认识和使用字典，请理解如下一些基本原则：

- 字典是一个键值对的集合，一个键信息只能对应一个值信息；
- 字典中元素以键信息为索引访问；
- 字典长度是可变的，可以通过对键信息赋值实现增加或修改键值对

6.5 模块4：jieba库的使用

jieba库概述

对一段英文，如“China is a great country”，如果想提取其中的单词，只需要使用字符串处理函数split即可（单词之间通过空格或标点符号分割）。
如：

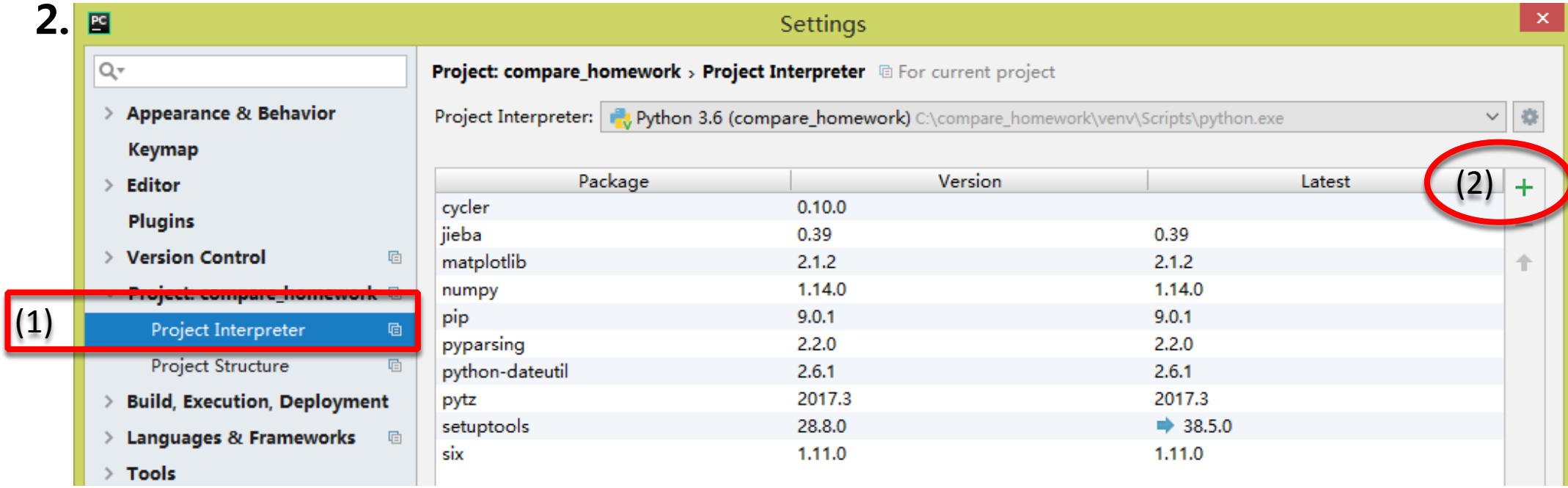
```
>>>"China is a great country“.split()
['China', 'is', 'a', 'great', 'country']
```

然而，对于“中国是一个伟大的国家”,获得其中的单词十分困难（中文单词之间缺少分隔符）；
为解决这个问题，Python引入了jieba三方库。

6.5 模块4：jieba库的使用

jieba库概述

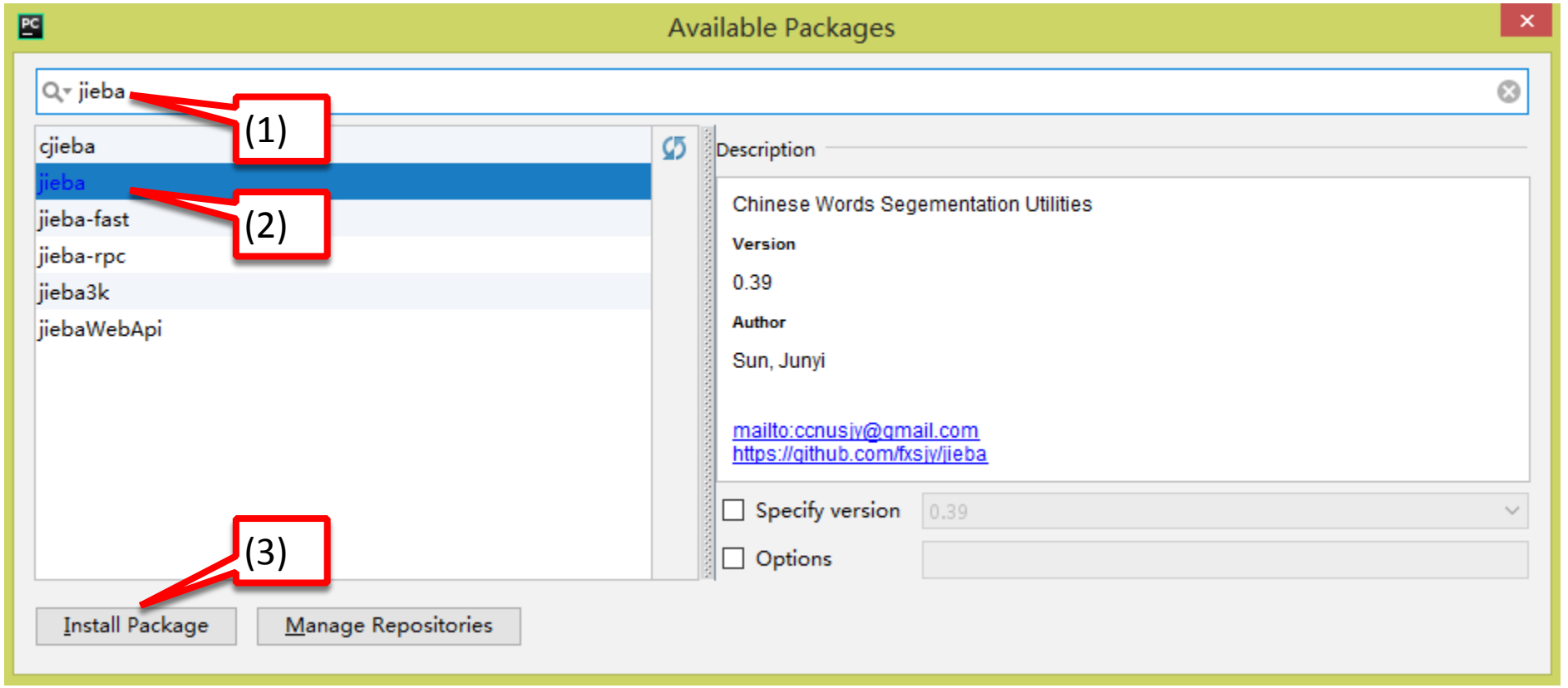
第三方库在pycharm中加载的方法：

1. File→Settings...
2. 

6.5 模块4：jieba库的使用

jieba库概述

3.输入jieba，选择jieba，单击install package,进行下载和加载



4.安装成功后，就可以用import jieba引入第三方库jieba，并使用jieba库的函数

6.5 模块4：jieba库的使用

jieba库解析

Jieba库支持3种分词模式：

- 精确模式：将句子最精确的切开，适合文本分析；
- 全模式：把句子中所有可以成词的词语都扫描出来，速度快，但不能消除歧义
- 搜索引擎模式：在精确模式的基础上，对长词再次切分，提高召回率，适合搜索引擎分词。

Jieba常用的分词函数：

函数	描述
jieba.cuts(s)	精确模式，返回一个可迭代的数据类型
jieba.cut(s,cut_all=True)	全模式，输出文本s中所有可能的单词
jieba.cut_for_search(s)	搜索模式，适合搜索引擎建立索引的分词结果
jieba.lcut(s)	精确模式，返回一个列表类型，建议使用
jieba.lcut(s,cut_all=True)	全模式，返回一个列表类型，建议使用
jieba.lcut_for_search(s)	搜索引擎模式，返回一个列表类型，建议使用
jieba.add_word(w)	向分词词典中增加新词w

6.5 模块4：jieba库的使用

jieba库解析

jieba常用的分词函数的举例：

```
>>> jieba.lcut("中华人民共和国是一个伟大的国家")
['中华人民共和国', '是', '一个', '伟大', '的', '国家']
>>> jieba.lcut("中华人民共和国是一个伟大的国家",cut_all=True)
['中华', '中华人民', '中华人民共和国', '华人', '人民', '人民共和国', '共和', '共和国', '国是', '一个', '伟大', '的', '国家']
>>> jieba.lcut_for_search("中华人民共和国是一个伟大的国家")
['中华', '华人', '人民', '共和', '共和国', '中华人民共和国', '是', '一个', '伟大', '的', '国家']
```

6.6 实例10：文本词频统计

输入：从文件中读取一篇文章
处理：采用字典数据结构统计词语出现频率；
输出：文章中最常出现的10个单词及出现次数。

分析:对于英文文本，因为以空格或标点符号分割，统计算法比较容易，对于中文文本相对麻烦，但可以以jieba进行中文分词。

6.6 实例10：文本词频统计

Hamlet英文词汇统计

1、提取单词：
（1）将文本文件读取到变量中；
（2）将变量的大写字符全部转换为小写字符
（3）将标点符号分隔符替换为空格；
2、对每个单词计数：
单词存入变量word，单词和对应次数存入字典类型counts={}中：
统计已存在的单词次数：counts[word]=counts[word]+1
新出现的单词：counts[word]=1
3、排序：
（1）字典类型无序，所以将字典类型转换为列表类型；
（2）再用sort()方法和lamda函数实现排序；
（3）输出前10位的单词及次数；

6.6 实例10：文本词频统计

Hamlet英文词汇统计

以读的方式打开文件，存入txt中

将所列字符替换为空格

进行单词切分，存入words集合

将字典类型转换为列表类型，存入items

对items排序 Lambda函数：以x为参数，返回x[1], reverse=True

```
def gettxt():
    txt=open("hamlet.txt","r").read()
    txt=txt.lower()
    for ch in '!"#$%^&*()+,_./:;<=>?@[\\]^-{|}':
        txt=txt.replace(ch," ")
    return txt
hamlettxt=gettxt()
words=hamlettxt.split()
counts={}
for word in words:
    counts[word]=counts.get(word,0)+1
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:>15}".format(word,count))
```

运行结果：

the	1137
and	965
to	754
of	667
you	550
a	542
i	541
my	514
hamlet	460
in	436

等价于：

```
if word in counts:
    counts[word]+=1
else:
    counts[word]=1
```

6.6 实例10：文本词频统计

Hamlet英文词汇统计

为排除一些冠词、代词、连接词，采用集合类型构建一个排除词汇库excludes。

定义被排除的单词集合

从counts中删除指定单词

```
excludes={"the","and","of","you","a","i","my","in"}
def gettxt():
    txt=open("hamlet.txt","r").read()
    txt=txt.lower()
    for ch in '!"#$%^&*()+,_./:;<=>?@[\\]^-{|}':
        txt=txt.replace(ch," ")
    return txt
hamlettxt=gettxt()
words=hamlettxt.split()
counts={}
for word in words:
    counts[word]=counts.get(word,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:>15}".format(word,count))
```

运行结果：

to	754
hamlet	460
it	416
that	391
is	340
not	314
lord	309
his	296
this	295
but	269

6.6 实例10：文本词频统计

三国演义人物出场统计

```
# -*- coding: utf-8 -*-
import jieba
import sys
sys.getdefaultencoding()
txt=open("三国演义.txt","r").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.items())
items.sort(key=lambda x:x[1],reverse=True)
for i in range(15):
    word,count=items[i]
    print("{0:<10}{1:>15}".format(word,count))
```

曹操	887
孔明	842
将军	746
却说	644
玄德	578
关公	497
丞相	466
二人	455
不可	422
荆州	405
不能	377
如此	364
玄德曰	354
孔明曰	345
张飞	338

:	<填充>	<对齐>	<宽度>	<,>	<.精度>	<类型>
引导符号	用于填充的单个字符	<左对齐 >右对齐 ^居中	槽设定输出宽度	数字的千位分隔符，适用于整数和浮点数	浮点数的小数部分的精度或字符串的最大输出长度	整数类型：b,c,d,o,x,X 浮点数：e,E,f%

6.6 实例10：文本词频统计

```
# -*- coding: utf-8 -*-
import jieba
import sys
sys.getdefaultencoding()
excludes={"将军","却说","荆州","二人","不可","不能","如此"}
txt=open("三国演义.txt","r").read()
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":
        rword="关羽"
    elif word=="玄德" or word=="玄德曰":
        rword="刘备"
    elif word=="孟德" or word=="丞相":
        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(15):
    word,count=items[i]
    print("{0:<10}{1:>15}".format(word,count))
```

```
曹操          1360
孔明          1323
刘备          1195
关羽           754
张飞           338
```

实验6.1 有30名同学的成绩存储在列表list1中：67,82,87,80,78,59,46,70,60,66,71,55,42,72,63,65,68,80,67,73,60,89,74,82,74,65,74,63,73,79
统计本班同学的平均分、最高分、最低分、方差。

实验6.2 随机生成有100个字符组成的字符串，统计每个字符出现的次数，counts列表存储26个字符的出现次数。

实验6.3 查单词，从word.txt文件中读取文本，存入字典中（英文单词为键，中文意思为值），运行时输入英文单词，显示对应的中文意思。

实验6.4 统计《射雕英雄传》中人物出现次数的排行榜（课下完成）

实验6.2 随机生成有100个字符组成的字符串，统计每个字符出现的次数，counts列表存储26个字符的出现次数。

```
import random
mlist=[]  #生成空列表
for i in range(100):
    rr=random.randint(97,122)
    #生成97到122范围内的随机数
    mlist.append(chr(rr))
    #将随机数转化成对应的字母
counts={}
```

```
for ch in mlist:
    if ch in counts:
        counts[ch]=counts[ch]+1
        #修改已经存在的元素的键值
    else:
        counts[ch]=1 #通过[]对字典增加新元素
items=list(counts.items())
#counts.items()返回所有的键值对，然后转化成列表
items.sort(key=lambda x:x[0])
#以字母的顺序升序排序
for x in items:
    print(x)
```

字典通过如下方式增加新元素：
counts[键]=值

实验6.3 查单词，从word.txt文件中读取文本，存入字典中（英文单词为键，中文意思为值），运行时输入英文单词，显示对应的中文意思。

word.txt

```
what 什么
is 是
your 你的,你们的
name 名字
my 我的
I 我
am 是
in 在...里(内,上)
row (一)排,(一)行
one 一
number 数字,号码
two 二
too 也
three 三
are 是
you 你,你们
yes 是
four 四
five 五
no 不,不是
```

tdicts

what 什么

is 是

your 你的,你们的

……

mlist→items

[what:什么]

[is:是]

[your:你的,你们的]

……

输入单词：if word=x[0]
从列表中查询对应的中文含义:
print(x[1])

实验6.3 查单词，从word.txt文件中读取文本，存入字典中（英文单词为键，中文意思为值），运行时输入英文单词，显示对应的中文意思。

```
# -*- coding: utf-8 -*-
import sys
import jieba
import sys
sys.getdefaultencoding()
#编码方式设置为系统默认方式
mfile=open("word.txt","r",encoding="gb18030",errors='ignore')
txt=mfile.read()
#将文件全部内容读入txt
tdicts=txt.split(“\n”)
#将内容按回车换行进行分割，并存入tdicts
mlist={}
```

```
for dict in tdicts:
    a=dict.split(“ ”)#按空格将中英文分隔开
    mlist[a[0]]=a[1]#存入字典
items=list(mlist.items())#转化为列表类型
while True:
    word = input(“请输入单词：”)
    for x in items:
        if word==x[0]:#如果word等于输入的英文
            print(x[1])#输出中文
    if word==“ ”:#如果输入为空格，结束查询
        break
```

```
C:\Users\Administrator\AppData\Local\Programs\Python\Python36\python.exe D:/课程实验/ex6.4.py
Traceback (most recent call last):
  File "D:/课程实验/ex6.4.py", line 5, in <module>
    txt=open("射雕英雄传.txt","r").read()
UnicodeDecodeError: 'gbk' codec can't decode byte 0xfd in position 1096106: illegal multibyte sequence


Process finished with exit code 1
```

使用python的时候经常会遇到文本的编码与解码问题，其中很常见的一种解码错误如题目所示，下面介绍该错误的解决方法，将‘gbk’换成‘utf-8’也适用。

（1）、首先在打开文本的时候，设置其编码格式，如：open(‘1.txt’,encoding=’gbk’)；

（2）、若（1）不能解决，可能是文本中出现的一些特殊符号超出了gbk的编码范围，可以选择编码范围更广的‘gb18030’，如：open(‘1.txt’,encoding=’gb18030’)；

（3）、若（2）仍不能解决，说明文中出现了连‘gb18030’也无法编码的字符，可以使用‘ignore’属性进行忽略，如：open(‘1.txt’,encoding=’gb18030’，errors=‘ignore’)；

（4）、还有一种常见解决方法为open(‘1.txt’).read().decode(‘gb18030’,’ignore’)

```
# -*- coding: gbk -*-
import sys
import jieba
sys.getdefaultencoding()
txt=open("射雕英雄传.txt","r",encoding="gb18030").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.items())
items.sort(key=lambda x:x[1],reverse=True)
for i in range(50):
    word,count=items[i]
    print("{0:<10},{1:>15}".format(word,count))
```

```
郭靖          ,           2615
黄蓉          ,           1751
甚么          ,           1276
说道          ,           1057
洪七公        ,           1057
欧阳锋        ,           1044
一个          ,           1040
自己          ,            993
师父          ,            875
黄药师        ,            868
心中          ,            778
武功          ,            771
两人          ,            724
咱们          ,            693
```