第六周

6.1 集合类型及操作

1.集合类型的定义

集合是**多个元素的无序组合**，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。这是因为，若集合元素可以改变，那么万一改变成和其它元素重复的，必然造成错误。

集合用大括号{}表示，元素之间用逗号分隔。建立集合类型有{}或 set()。**如果建立空集合，必须用 set()**。例如：

A={“python”,123,(“python”,123)}#使用{}建立集合

结果是：{‘python’,123,(‘python’,123)}

B=set(“pypy123”)#使用 set()建立集合

结果是{‘1’,’p’,’2’,’3’,’y’}

2.集合操作符

集合间操作主要有：

（1）并：S|T

（2）差：S-T

（3）交：S&T

（4）补：S^T

表 1.1 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 中所有元素
S-T	返回一个新集合，包括在集合 S 但不 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 中的费相同元素
S<=T 或 S<T	返回 True 或 False，判断 S 和 T 的子集关系
S>=T 或 S>T	返回 True 或 False，判断 S 和 T 的包含关系

表 1.2 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 中所有元素
S-=T	更新集合 S，包括在集合 S 但不 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

3.集合处理方法

表 1.3 集合处理方法

操作函数及方法	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 S 中，不报错

S.remove(x)	移除 S 中元素 x，如果 x 不在 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（删除该元素），若 S 为空产生 keyError 异常
S.copy()	返回 S 的一个副本
len(S)	返回集合 S 中元素个数
x in S	判断 S 中元素是否有 x，若有返回 True，否则 False
x not in S	判断 S 中元素是否无 x，若无返回 True，否则 False
set(x)	将其他类型变量 x 转变成集合类型

6.2 序列类型及操作

1.序列类型的定义

序列是具有先后关系的一组元素。它是一维元素向量，元素类型可以不同；类似数学元素序列 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。序列衍生出：字符串类型、元组类型、列表类型。

2.序列处理函数及方法

基本之前都讲过，这里只讲下面几个：

s.index(x)或 s.index(x,i,j)：返回 s 从 i 开始到 j 位置（省略 i 和 j 则是整个序列）中第一次出现 x 的位置。

s.count(x)：是中出现 x 的总次数。

3.元组类型及操作

元组是一种序列类型，但是它**一旦被创建就不能修改**。元组使用小括号()或者 tuple()创建，元素之间用逗号分隔。在使用的时候，可以使用或不使用小括号。

4.列表类型及操作

列表是一种序列类型，但是它创建后**能修改**。它使用方括号[]或 list()创建，元素用逗号分隔。可以使用或不使用小括号。

表 2.1 列表类型操作函数和方法

函数或方法	描述
ls[i]=x	替换列表 ls 第 i 个元素为 x
ls[i:j:k]=lt	用列表 lt 替换 ls 切片后所对应元素字列表
del ls[i]	删除列表 ls 中的第 i 元素
del ls[i:j:k]	很容易懂，不写了
ls+=lt	更新列表 ls，将 lt 加到 ls 后面
ls*=n	更新列表 ls，其元素重复 n 次
ls.append(x)	在 ls 后面增加一个元素 x
ls.insert(i,x)	在 ls 第 i 个位置增加元素 x
ls.remove(x)	将 ls 中出现的第一个 x 删除
ls.reverse()	将 ls 反转

5.序列类型典型应用场景

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，它最常用的

序列类型。

6.3 实例 9：基本统计值计算

例：计算一组数据的总个数、总和、平均数、方差、中位数。

代码：

```
def getNum():
    nums = []
    iNumStr = input("请输入数字")
    while iNumStr !="":
        nums.append(eval(iNumStr))
        iNumStr = input("请输入数字")
    return nums
def getMean(numbers):
    s=0.0
    for num in numbers:
        s += num
    return s/len(numbers)
def dev(numbers,mean):
    sdev=0.0
    for num in numbers:
        sdev = sdev+(num-mean)**2
        return pow(sdev/(len(numbers)-1),0.5)

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=getMean(n)
d=dev(n,m)
med = median(n)
print("平均值是{0}，方差是{1},中位数是{2}".format(m,d,med))
```

6.4 字典类型及操作

1.字典类型定义

字典类型是映射的体现，它通过键值对来体现，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。例如：

d={“China”:”Beijing”,”France”:”Paris”}

输入

d[“China”],

结果是

’Beijing’

如果我们想生成一个空字典，则可以是使用

de={}

直接用大括号只能生成字典！

2.字典处理函数及方法

表 4.1 字典类型操作函数和方法

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，在就是 True，否则 False。k 是索引
d.keys()	返回 d 中所有的键信息
d.values()	返回 d 中所有的值信息
d.items()	返回 d 中所有键值对信息
d.get(k,<default>	键 k 存在，则返回相应值，否则返回<default>值
d.pop(k,<default>)	键 k 存在，则取出（并删除）相应值，否则返回<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()	删除所有键值对
len(d)	返回字典 d 中元素的个数

6.5 模块 5：jieba 库的使用

jieba 是优秀的中文分词第三方库。jieba 分词有三种模式：精确模式、全模式、搜索引擎模式。

精确模式：把文本精确的分开，不存在冗余单词。

全模式：把文本中所有可能的词语都扫描出来，有冗余。

搜索引擎模式：在精确模式基础上，对长词再次切分。

表 5.1 jieba 库常用函数

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粉刺结果，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

6.6 实例 10：文本词频统计

1.Hamlet 词频统计：

```
def getText():
    txt = open("hamlet.txt", "r").read()
    txt = txt.lower()
    for ch in '!"#$%&()*+,-./:;<=>?@[\\]^_‘{|}~':
        txt = txt.replace(ch, " ")     #将文本中特殊字符替换为空格
    return txt

hamletTxt = getText()
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:>5}".format(word, count))
```

2. 《三国演义》人物出场统计（下）（含《三国演义》原文文本）

```
import jieba
excludes = {"将军","却说","荆州","二人","不可","不能","如此"}
txt = open("threekingdoms.txt", "r", encoding='utf-8').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(10):
    word, count = items[i]
    print ("{0:<10}{1:>5}".format(word, count))
```