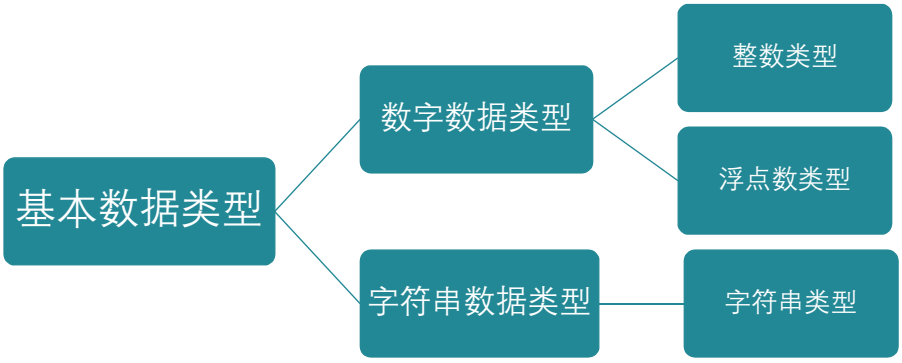
程序设计

Week 9

组合数据类型

Python的数据类型

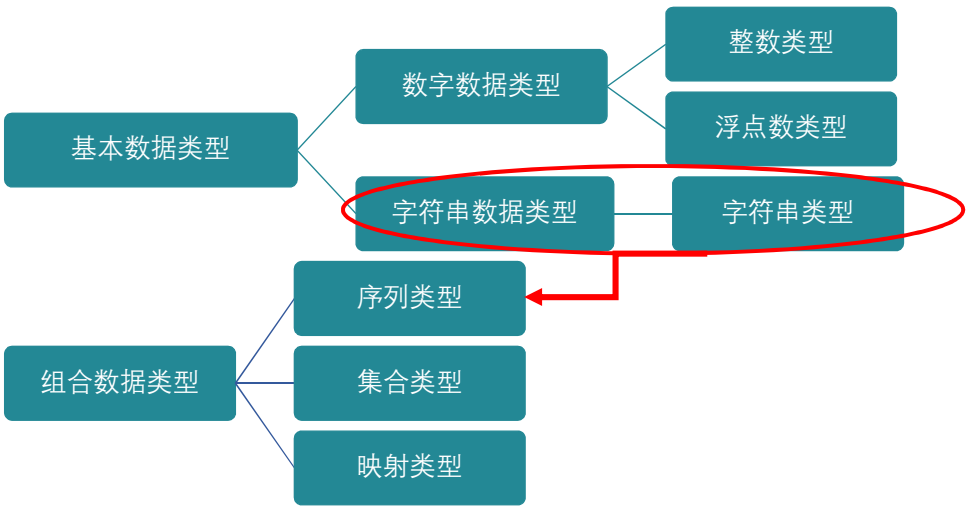


组合数据类型

计算机不仅对单个变量表示的数据进行处理，更多情况，计算机需要对一组数据进行批量处理。一些例子包括：

1. 给定一组单词{python, data, function, list, loop}，计算并输出每个单词的长度；
2. 给定一个学院学生信息，统计一下男女生比例；
3. 一次实验产生了很多组数据，对这些大量数据进行分析；

组合数据类型



组合数据类型

- 序列类型是一个元素向量，元素之间存在先后关系，通过序号访问，元素之间不排他。
- 集合类型是一个元素集合，元素之间无序，相同元素在集合中唯一存在。
- 映射类型是"键-值"数据项的组合，每个元素是一个键值对，表示为(key, value)。

序列类型

序列类型是一维元素向量，元素之间存在先后关系，通过序号访问。

当需要访问序列中某特定值时，只需要通过下标标出即可。

$$\sum_{i=0}^{n-1} S_i$$

序列类型

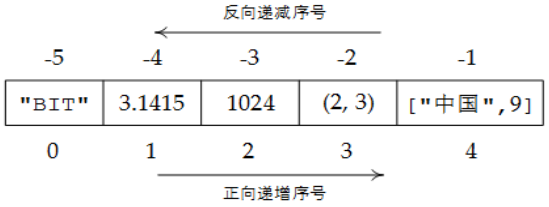
由于元素之间存在顺序关系，所以序列中可以存在相同数值但位置不同的元素。序列类型支持成员关系操作符（in）、长度计算函数（len()）、分片（[]），元素本身也可以是序列类型。

序列类型

Python语言中有很多数据类型都是序列类型，其中比较重要的是：str（字符串）、tuple（元组）和list（列表）。

•元组是包含0个或多个数据项的不可变序列类型。元组生成后是固定的，其中任何数据项不能替换或删除。

•列表则是一个可以修改数据项的序列类型，使用也最灵活



序列类型

序列类型有12个通用的操作符和函数

操作符	描述
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
s + t	连接s和t
s * n 或 n * s	将序列s复制n次
s[i]	索引，返回序列的第i个元素
s[i: j]	分片，返回包含序列s第i到j个元素的子序列（不包含第j个元素）
s[i: j: k]	步骤分片，返回包含序列s第i到j个元素以j为步数的子序列
len(s)	序列s的元素个数（长度）
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中出现x的总次数

列表

列表的定义

- 列表是值的序列。
- 在字符串中，这些值是字符，在列表中，它可以是任何类型。
- 列表中的值称为元素（element），有时也叫列表项（item）。
- 列表用中括号（[]）表示。

列表的定义

```
[10, 20, 30 ,40]

['一月','二月', '三月']

['Spam', 2.0, 5, [10, 20]]
```

列表的创建

- 使用方括号（[]）创建。
- 使用list()创建。

```
>>>ls = [425, "BIT", [10, "CS"], 425]
>>>ls
[425, 'BIT', [10, 'CS'], 425]
>>>ls[2][-1][0]
'C'
>>>list((425, "BIT", [10, "CS"], 425))
[425, 'BIT', [10, 'CS'], 425]
>>>list("中国是一个伟大的国家")
['中', '国', '是', '一', '个', '伟', '大', '的', '国', '家']
>>>list()
[]
```

列表的访问

列表是有序集合，因此要访问列表的任何元素，只需将该元素的位置或索引告诉Python即可。

```
a = ['Spam', 2.0, 5, [10, 20]]
a[1]
```

列表的访问

列表任何整型的表达式都可用做下标；

读写一个不存在的元素，会得到IndexError；

下标是负数，则从列表结尾处反过来数下标访问。

列表的操作

操作符	描述
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
s + t	连接s和t
s * n 或 n * s	将序列s复制n次
s[i]	索引，返回序列的第i个元素
s[i: j]	分片，返回包含序列s第i到j个元素的子序列（不包含第j个元素）
s[i: j: k]	步骤分片，返回包含序列s第i到j个元素以j为步数的子序列
len(s)	序列s的元素个数（长度）
min(s)	序列s中的最小元素
max(s)	序列s中的最大元素

列表的操作

列表是动态的，所以可以添加、修改和删除。

列表的操作——添加

在列表末尾添加，使用.append()方法；

在列表中插入元素，使用insert()方法，需指定新元素的索引和值；

列表的操作——修改

与字符串不同，列表元素的值是可变的：

```
a = ['Spam', 2.0, 5, [10, 20]]
a[1] = 3
```

列表的操作——删除

使用del语句删除元素；

使用pop()方法删除列表末尾的元素，并返回其值；

使用pop()方法删除指定位置的元素；

使用remove()方法，根据值删除元素；

列表的操作——其他方法

使用方法 sort()对列表进行永久性排序；

使用函数 sorted()对列表进行临时排序；

使用reverse()方法倒序打印列表；

使用len()函数，确定列表长度。

列表的操作——其他方法

函数或方法	描述
ls[i] = x	替换列表ls第i数据项为x
ls[i: j] = lt	用列表lt替换列表ls中第i到j项数据（不含第j项，下同）
ls[i: j: k] = lt	用列表lt替换列表ls中第i到j以k为步的数据
del ls[i: j]	删除列表ls第i到j项数据，等价于ls[i: j]=[]
del ls[i: j: k]	删除列表ls第i到j以k为步的数据
ls += lt或ls.extend(lt)	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ls中所有元素
ls.copy()	生成一个新列表，复制ls中所有元素
ls.insert(i, x)	在列表ls第i位置增加元素x
ls.pop(i)	将列表ls中第i项元素取出并删除该元素
ls.remove(x)	将列表中出现的第一个元素x删除
ls.reverse(x)	列表ls中元素反转

列表的高级操作

列表的遍历

如何进行列表的遍历?

列表的遍历

如何进行列表的遍历?

```
a = ['Spam', 2.0, 5, [10, 20]]
for i in a:
    print(id)
```

创建数值列表

如何使用range()函数创建数值列表?

列表切片遍历操作

[start: stop: step]

如果要遍历列表的部分元素，可在for循环中使用切片。

列表的复制

见代码。

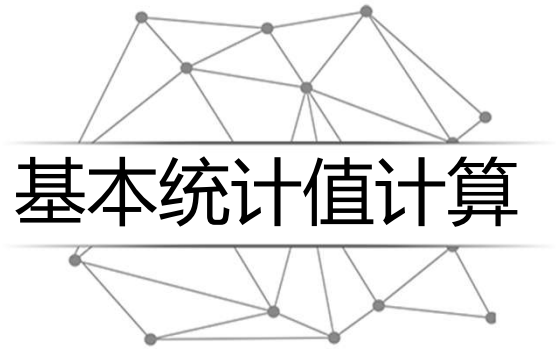
元组

元组

元组是一组值的序列。其中的值可以是任意类型，使用整数索引其位置，因此元组与列表非常相似。而重要的不同之处在于元组的不可变性。

元组的作用

1. 函数多返回值
2. 多变量同步赋值
3. 遍历循环



基本统计值计算

基本统计值的计算

以最简单的统计问题为例，求解一组不定长数据的基本统计值，即平均值、标准差、中位数。

一组数据表示为$S=s_0, s_1, \ldots, s_{n-1}$，其算术平均值、标准差分别表示为：

$$m = (\sum_{i=0}^{n-1} s_i) / n \quad \text{和} \quad d = \sqrt{(\sum_{i=0}^{n-1} (s_i - m)^2) / (n-1)}$$

基本统计值的计算

由于平均数、标准差和中位数是三个不同的计算目标，使用函数方式编写计算程序。

getNum()函数从用户输入获得数据

mean()函数计算平均值

dev()函数计算标准差

median()函数计算中位数

基本统计值的计算

实例代码9.1 e9.1CalStatistics.py

```
#e9.1CalStatistics.py
from math import sqrt
def getNum():        #获取用户输入
    nums = []
    iNumStr = input("请输入数字(直接输入回车退出): ")
    while iNumStr != "":
        nums.append(eval(iNumStr))
        iNumStr = input("请输入数字(直接输入回车退出): ")
    return nums
def mean(numbers):  #计算平均值
    s = 0.0
    for num in numbers:
        s = s + num
    return s / len(numbers)
```

基本统计值的计算

实例代码9.1 e9.1CalStatistics.py

```
def dev(numbers, mean): #计算方差
    sdev = 0.0
    for num in numbers:
        sdev = sdev + (num - mean)**2
    return sqrt(sdev / (len(numbers)-1))
def median(numbers):    #计算中位数
    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()  #主体函数
m =  mean(n)
print("平均值:{},方差:{:.2},中位数:{}.".format(m,\
 dev(n,m),median(n)))
```

基本统计值的计算

```
>>>
请输入数字(直接输入回车退出): 99
请输入数字(直接输入回车退出): 98
请输入数字(直接输入回车退出): 97
请输入数字(直接输入回车退出): 96
请输入数字(直接输入回车退出): 95
请输入数字(直接输入回车退出):
平均值:97.0,方差:1.6,中位数:97.
```

程序先后调用getNum()、mean()、dev()和median()函数。利用函数的模块化设计能够复用代码并增加代码的可读性。每个函数内部都采用了简单的语句。

基本统计值的计算

列表在实现基本数据统计时发挥了重要作用，表现在：

- 列表是一个动态长度的数据结构，可以根据需求增加或减少元素；
- 列表的一系列方法或操作符为计算提供了简单的元素运算手段；
- 列表提供了对每个元素的简单访问方式及所有元素的遍历方式。