“小帅，老师给了我一个含有全班同学姓名的文件，让我将全班同学的姓名工整地打印出来。除了手工一个字一个字地录入以外，有什么好办法能够快速完成这项任务呢？”

“小萌，我们学习过一维数据，想让全班同学的名字工整地排列，我们可以读取文件，然后在一维的列表中用程序控制读写，轻松完成这项任务！”

5.1　一维数据的表示

Python 的组合数据类型可以很好地表示一维数据。如果一维数据存在顺序关系，那么用列表表示就是一种最为合理的选择。例如，列表中三个元素的值分别代表小明、小红、小丽的一分钟跳绳次数：skip = [120,135,112]。这时列表中的值与不同的同学一一对应，是有次序的。可以通过 for 循环遍历其中的数据，实现对每个数据的处理。

```
skip = [120, 135, 112]
print("小明、小红、小丽的跳绳成绩依次为（次/分）:")
for record in skip:
    print(record)
print("其中最高记录为{}次/分".format(max(skip)))  # 用max()统计
                                                  # 最大值
```

无序的一维数据则适合用集合类型来表示，因为 Python 的集合类型可以自动去除重复数据，不支持索引、切片等与先后顺序相关的操作，这样的特点恰恰符合无序数据的特征。例如，两组同学分别记录了自己在动物园中参观过的动物名称，最后列举大家一共参观过哪些动物，统计参观过的动物种类数量。

运行结果如下。

```
group1 = {'非洲鸵鸟','亚洲象','美洲狮','长颈鹿','孔雀','金丝猴'}
group2 = {'大熊猫','孔雀','马来熊','亚洲象','长臂猿','金丝猴'}
group_t = group1 | group2       # 集合的并运算
print("两组同学参观的动物有:")
for a in group_t:
    print(a,end="*")
print("\n两组同学共参观了{}种动物".format(len(group_t)))
```

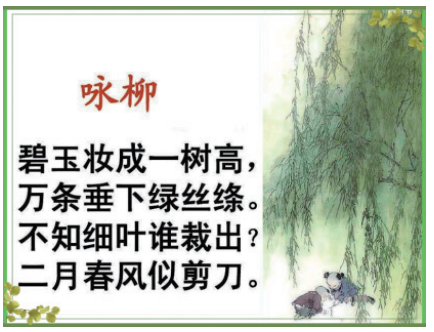
```
两组同学参观的动物有:
大熊猫*亚洲象*非洲鸵鸟*美洲狮*马来熊*长臂猿*孔雀*长颈鹿*金丝猴*
两组同学共参观了9种动物
```

“除了列表和集合，其他数据类型能不能表示一维数据呢？”

“其实元组、字符串等用来表示一维数据也是可以的，但是元组不可修改，字符串中每个元素都是字符，这些局限在一维数据的计算中可能会造成编程的困难。”

练一练

【问题 5-1】 语文课将要学习唐诗——《咏柳》，这首诗中，同学们学习过的汉字有“柳、玉、成、一、树、万、下、绿、不、叶、出、二、月、春、风、刀”。请编写程序，为同学们列出这首唐诗中将要学习的生字有哪些。



5.2　一维数据的存储

存储对于数据处理而言，是首先要做好的操作。一维数据是较为简单的数据组织形式，可以使用多种存储方式实现一维数据的存储。在硬盘、U 盘等存储设备中以文件形式存储一维数据，是常见的存储形式。

以文件作为读取数据的“源头”，或者写入数据的“目的地”，可以使数据保存在硬盘、U 盘等存储设备中，即便程序运行完毕，甚至关闭计算机，数据都不会丢失，这也称为“永久存储”。

数据的存储是为下一步读取、写入及操作数据做准备的，所以在存储中不但要考虑将内容正确地存储下来，还要考虑数据间的合理间隔与区分。通常采用下面几种方式存储一维数据。

1. 以空格作为分隔符的存储方式

数据的存储格式形如：

```
李白 杜甫 白居易 孟浩然 李商隐
```

以空格作为分隔符是一种简单的一维数据存储方式，可以使用一个或多个空格分隔数据，进行存储，数据不必换行。但是如果采用这样的方式存储数据，则要求数据本身不能存在空格，否则将出现数据与分隔符相同，造成数据解析出错的情况。使用其他符号作为分隔符也是类似的，不允许数据内容中含有和分隔符相同的内容。

2. 以逗号作为分隔符的存储方式

通常习惯使用半角逗号“，”作为分隔符，进行一维数据的存储，数据无须换行。半角逗号“，”在程序中常用作表达式或者参数之间的分隔符，在存储中其作用也是相同的。例如：

```
李白，杜甫，白居易，孟浩然，李商隐
```

3. 以其他特殊符号作为分隔符的存储方式

实际上，只要在程序中设置妥当，任何可见的符号、文字都可以作为一维数据的分隔符，例如，“%”“#”等，但是这些符号可能在数据中具有其他含义，容易造成内容和分隔符的冲突，且通用性不强，一般不建议使用。

5.3 一维数据的读写

作为线性展开的一维数据，有时这条“线”可能非常长，即拥有众多的数据。若编程时，数据都是手工录入的，那么编程效率将非常低。对于一维数据，可以采用“读”的方式，从文件中读出，也可以将一维数据写入指定的文件，这样就便于处理蕴含大量数据项的一维数据了。

“读”和“写”是数据处理中最基本的两种操作，其实在数据读、写过程中，还可以进行数据的统计、分析、计算等操作。

一维数据的读写处理就是实现一维数据从存储它的文件中向列表、集合等表示方式转换的过程。

1. 一维数据的读取

如果一维数据以文件的形式存在，则可以通过程序将这些数据读出，并将其转换为列表、集合等数据类型，以便于进行数据的分析和计算。

一维数据的读取可以通过文件内置的读取方法加以实现，如果读取方法返回的是字符串，例如 file.read( )，则在后续处理中要用字符串相应的方法来实现对数据的分析及处理，最后转换为列表或集合等类型。

例如，有如图 5-1 所示的文本文件。

读取上述文件中的文本，并以列表和集合的方式处理为一维数据的代码如下。

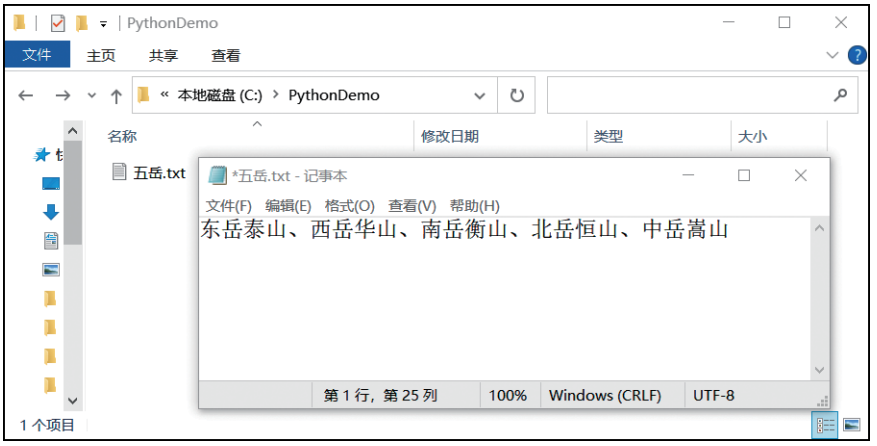


图 5-1　读取的文本信息

```
f = open('C:/PythonDemo/五岳.txt','r',encoding='UTF-8')
tstr = f.read()
tlst = tstr.split('、')   # 以'、'为分隔符，将字串 tstr 拆分为列表
tset = set(tstr.split('、'))   # 将拆分的列表转换为集合
print('列表形式：',tlst)
print('集合形式：',tset)
f.close()
```

运行结果如下。

```
列表形式：['东岳泰山', '西岳华山', '南岳衡山', '北岳恒山', '中岳
          嵩山']
集合形式：{'北岳恒山', '中岳嵩山', '西岳华山', '南岳衡山', '东岳
          泰山'}
```

注意，在打开文件时，如果打开方式为文本方式，则 encoding 参数的编码格式需要与文本文件保存的编码格式一致，否则可能导致文件读取错误。如果文件以二进制形式打开，则不支持 encoding 参数。字符串的 split( ) 方法可以指定字符串的分隔符号，将字符串信息转换为列表。字串的分隔符可以是空格、'#'、'%' 等任意符号。

从程序运行结果可见，集合形式打乱了文件中数据的原有顺序，呈现了一维数据的无序的特点。

【问题 5-2】 如果文本中的数据有多行，将每一行信息都读取并处理成一个列表或者集合，应该如何编程实现？

提示：充分利用文件读取方法及组合数据类型的运算方法。

2. 一维数据的写入

数据的写入操作与读取操作相对应，通过写入可以将一维数据“永久”存储在文件之中。将一维数据写入文件时，可以使用空格以及其他符号作为分隔符分隔数据，以便重新从文件中读取数据时进行识别数据。

在写入数据时，要充分利用字符串、列表、集合等数据类型或序列的内置方法，以灵活处理数据。例如，str.join(iterable) 就是一种在写入一维数据时常用的字符串内置方法，该方法可以将 str 作为分隔符，将组合类型 iterable 中的每个元素进行连接，形成一个字符串，最后将生成的字符串写入文件。例如下列代码：

```
ls = ['长江', '黄河', '黑龙江','雅鲁藏布江']
f = open('C:/PythonDemo/河流.txt', 'w')
f.write(' '.join(ls)) #以空格作为数据的分隔符
f.close()
```

得到的结果如图 5-2 所示。

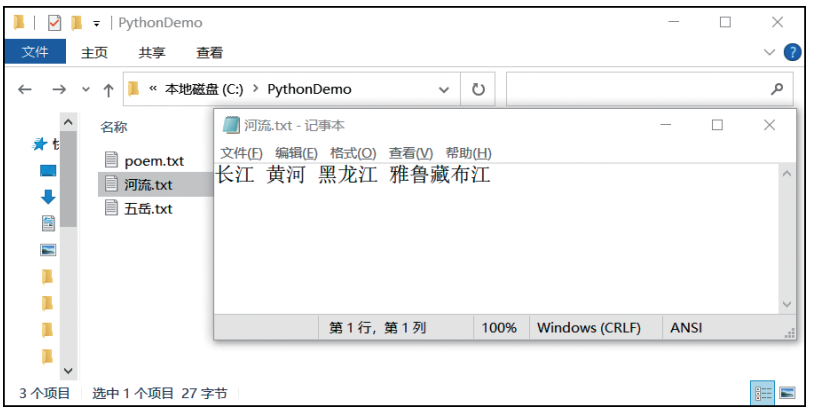


图 5-2 写入后生成的文件

如果要写入的数据来源于有分隔符的字符串，则可以通过字符串内置方法 str.split('character')，以 character 为分隔符，将字符串转换为列表，再参考上述代码写入文件中。例如：

```
>>> str_book ='论语 # 左传 # 孟子 # 春秋 # 礼记 '
>>> list_book = str_book.split('#')
>>> list_book
[' 论语 ', ' 左传 ', ' 孟子 ', ' 春秋 ', ' 礼记 ']
```

“老师，我想把由数字构成的列表用 write( ) 方法写入到文件中，例如 [1,2,3]，为什么程序会出错呢？”

“这是因为 file.write( ) 方法仅支持将字符串写入文件，所以在将数字等信息写入前，要先转换为字符串类型。请看下面的例子，并尝试自己编程实践一下吧。”

```
ns = [1,2,3,4,5]
f = open("test1.txt","w")
str_ns = str(ns)[1:-1]
# 将列表用 str() 方法转为字符串，并通过切片去掉两边的 "[" 与 "]"
f.write(str_ns)
f.close()
```

【问题 5-3】　小明同学要将彩虹的七种颜色英文词汇写入文件 rainbow.txt 中，并以星号“*”作为分隔符，请在下列代码的画线处填写适当的代码，实现上述功能。

```
colors = 'red,orange,yellow,green,blue,indigo,purple'
colors_list = colors.___(1)___(',')
f = open('rainbow.txt','w')
f.write('*'.___(2)___(colors_list))
f.close()
```

“本单元我们不仅学习了一维数据的表示，而且学习了一维数据的存储及读写等操作。列表和集合是表示一维数据常用的结构，对一维数据的操作往往在列表、集合与文件之间展开，用好文件、列表、集合以及字符串的方法，是在编程中灵活处理一维数据的关键所在。

同学们一定要在学习中做到‘温故而知新’哦。”

习　题

1. 用集合记录水彩盘里的颜色，sc = {'red','green','blue'}，集合 sc 的维度是（　　）。

A. 一维数据　　B. 二维数据　　C. 多维数据　　D. 多维数据

2. 下列选项中，不属于一维数据的是（　　）。

A. [4,5,9,0]

B. "AB98C"

C. ['Monkey',12,34,['Animal',9]]

D. {'h',12,34.5,' 美丽 '}

3. 已知列表 demo_list = [5,1,"talk",'9'],则 demo_list 的维度是(　　)。

A. 一维数据　　B. 二维数据　　C. 多维数据　　D. 多维数据

4. 下列关于一维数据的存储格式的描述，错误的是（　　）。

A. 一维数据可以采用空格分隔方式存储

B. 一维数据可以采用分号“;”分隔方式存储

C. 一维数据不能用特殊符号“@”分隔方式存储

D. 一维数据可以用星号“*”分隔方式存储

5. 下列关于一维数据的描述，正确的是（　　）。

A. 只能用 CSV 文件来存储一维数据

B. 一维数据就是列表或集合

C. “%”是运算符，不能用于分隔一维数据

D. 一维数据都是线性的

6. 关于无序的一维数据，下列描述错误的是（　　）。

A. 无序的一维数据可以用列表类型来表达

B. 无序的一维数据可以用集合类型来表达

C. 无序的一维数据可以用元组类型来表达

D. 无序的一维数据无法用 Python 语言有效表达

7. 若 ls=['1','2','3','4','5']，则表达式“#”.join(ls) 的功能是（　　）。

A. 将列表中所有的元素连接成一个字符串，元素之间用 '#' 作为分隔符

B. 在列表 ls 的每一个元素后面加一个 '#' 号

C. 将 '#' 作为后缀，加到列表 ls 的后面

D. 在列表 ls 的每个元素后面加一个 '#' 号

8. 运行下列代码，输出结果是（　　）。

```
print(" love ".join(["Everyday","Yourself","Python",]))
```

A. Everyday love Yourself

B. Everyday love Python

C. love Yourself love Python

D. Everyday love Yourself love Python

9. 编程实现以下功能。

已知有班级同学名单和某次活动出席同学名单，统计该次活动缺席同学名单，姓名之间用逗号“,”隔开，并保存入名为“abs.txt”的文件中。

例如，有如图 5-3 所示数据（仅为举例，编程时可以自行拟定）

班级名单	小红	晓敏	小刚	小萌	小帅	小兵	大雷	大壮
出席名单	小萌	小兵	大壮	晓敏				

图 5-3　出席活动名单

则创建的 abs.txt 中，缺席名单应该为“小红，小刚，小帅，大雷”，名字次序不做要求。