第七章

文件和数据的格式化

本章大纲

1.文件的使用: 文件打开、关闭和读写

2.数据组织的维度：一维数据和二维数据

3.一维数据的处理：表示、存储和处理

4.二维数据的处理：表示、存储和处理

5.采用CSV格式对一二维数据文件的读写

1.文件

文件是存储在辅助存储器上的一组数据序列，可以包含任何数据内容。概念上，文件是数据的集合和抽象。文件包括两种类型：文本文件和二进制文件。

2.文件的类型

1、 文本文件一般由单一特定编码的字符组成，如 UTF-8编码，内容容易统一展示和阅读。

2、 二进制文件直接由比特0和比特1组成，文件内部数据的组织格式与文件用途有关。二进制是信息 按照非字符但特定格式形成的文件，例如，png 格式的图片文件、avi格式的视频文件

3、二进制文件和文本文件最主要的区别在于是否有统一的字符编码。

4、无论文件创建为文本文件或者二进制文件，都可以用“文本文件方式”和“二进制文件方式”打开，但打开后的操作不同。

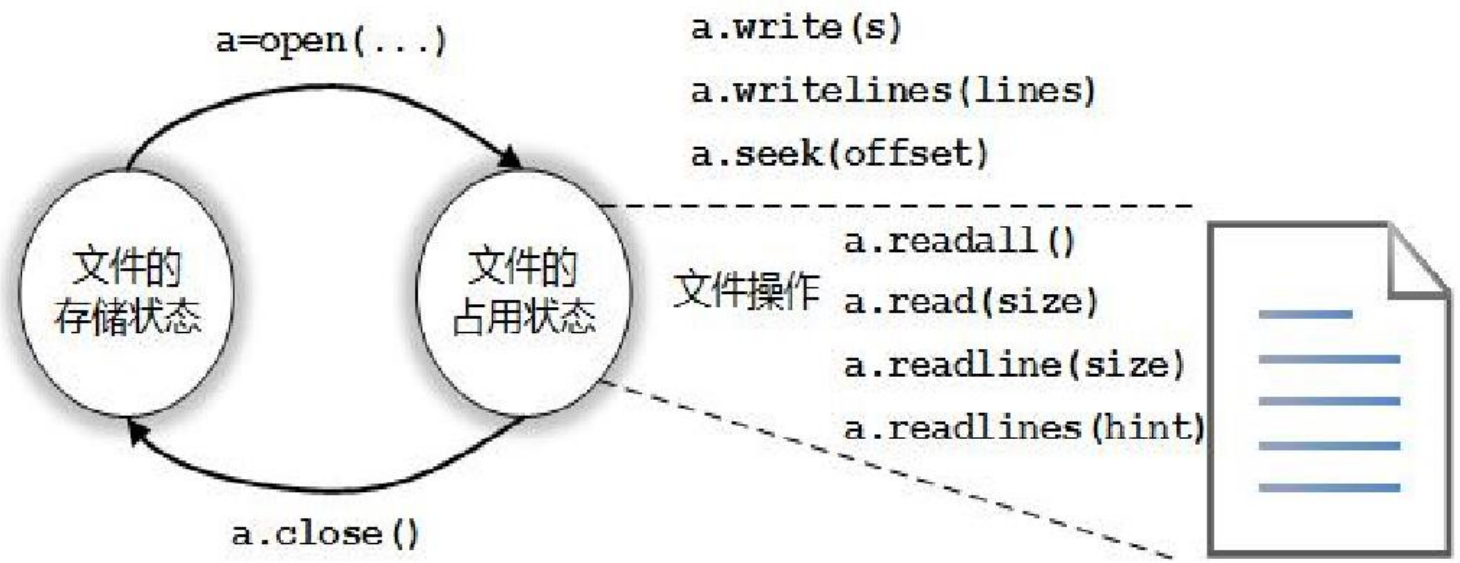
```
f = open('a.txt','rt') #t表示文本方式打开
print(f.readline())
f.close()


f = open('a.txt','rb') #b表示二进制文件方式打开
print(f.readline())
f.close()
```

采用文本方式读入文件，文件经过编码形成字符串，打印出有含义的字符；采用二进制方式打开文件，文件被解析为字节流

3.文件的打开和关闭

1、Python对文本文件和二进制文件采用统一的操作步骤， 即“打开-操作-关闭”



2、Python通过open()函数打开一个文件，并返回一 个操作这个文件的变量，语法形式如下：

<变量名> = open(<文件路径及文件名>, <打开模式>)

打开模式	含义
‘r’	只读模式，如果文件不存在，返回异常FileNotFoundError，默认值
‘w’	覆盖写模式，文件不存在则创建，存在则完全覆盖源文件
‘x’	创建写模式，文件不存在则创建，存在则返回异常FileExistsError
‘a’	追加写模式，文件不存在则创建，存在则在原文件最后追加内容
‘b’	二进制文件模式
‘t’	文本文件模式，默认值
‘+’	与r/w/x/a一同使用，在原功能基础上增加同时读写功能

3、打开模式使用字符串方式表示，根据字符串定义， 单引号或者双引号均可。上述打开模式中， ‘r’ 、 ‘w’ 、 ‘x’ 、 ‘b’ 可以和 ‘b’ 、 ‘t’ 、 ‘+’ 组合使用，形成既 表达读写又表达文件模式的方式。

4、文件使用结束后要用close()方法关闭，释放文件的使用授权，语法形式如下：

<变量名>.close()

4.文件的读取

1、通过以下几种方式可以读取文件指定的内容。

方法	含义
f.read(size=-1)	从文件中读入整个文件内容。参数可选，如果给出，读入 前size长度的字符串或字节流
f.readline(size = -1)	从文件中读入一行内容。参数可选，如果给出，读入该行 前size长度的字符串或字节流
f.readlines(hint=-1)	从文件中读入所有行，以每行为元素形成一个列表。参数 可选，如果给出，读入hint行
f.seek(offset,whence)	改变当前文件操作指针的位置，whence的值： 0：文件开头；1:当前位置 2: 文件结尾

2、如果文件不大，可以一次性将文件内容读入，保存到程序内部变量中。f.read()是最常用的一次性读入文件的函 数，其结果是一个字符串。

```
>>> f = open('D://a.txt','r',encoding='utf-8')
>>> s = f.read()
>>> print(s)
新年都为有芳华，二月初惊见草芽。
白雪却嫌春色晚，故穿庭树作飞花。
>>> f.close()
```

3、f.readlines()也是一次性读入文件的函数，其结果是一个列表，每个元素是文件的一行。

```
>>> f = open('D://a.txt','r',encoding='utf-8')
>>> ls = f.readlines()
>>> print(ls)
['新年都为有芳华，二月初惊见草芽。\n', '白雪却
嫌春色晚，故穿庭树作飞花。']
>>> f.close()
```

4、f.seek(offset,whence)方法能够移动读取指针的位置。

offset： 开始的偏移量，也就是代表需要移动偏移的字节数，如果是负数表示从倒数第几位开始。

whence：可选，默认值为 0。给 offset 定义一个参数，表示要从哪个位置开始偏移；0 代表从文件开头开始算起，1 代表从当前位置开始算起，2 代表从文件末尾算起。

```
>>> f = open('D://a.txt','r',encoding='utf-8')
>>> s = f.read()
>>> print(s)
新年都为有芳华，二月初惊见草芽。
白雪却嫌春色晚，故穿庭树作飞花。
>>> f.seek(0) #将读取指针重置到文件开头
0
>>> ls = f.readlines()
>>> print(ls)
['新年都为有芳华，二月初惊见草芽。\n', '白雪却
嫌春色晚，故穿庭树作飞花。']
>>> f.close()
```

5.文件的写入

通过以下方法可以将数据写入指定的文件

方法	含义
f.write(s)	向文件写入一个字符串或字节流
f.writelines(lines)	将一个元素为字符串的列表写入文件

2、f.write(s)向文件写入字符串s，每次写入后，将会记录一个写入指针。该方法可以反复调用，将在写入指针后分批写入内容，直至文件被关闭。

```
f = open('D://c.txt','w')
f.write('新年都为有芳华\n')
f.write('二月初惊见草芽\n')
f.write('白雪却嫌春色晚\n')
f.write('故穿庭树作飞花\n')
f.close()
```

3、f.writelines(lines)直接将列表类型的各元素连 接起来写入文件f。

```
ls = ['新年都为有芳华\n，二月初惊见草芽。\n',
      '白雪却嫌春色晚\n，故穿庭树作飞花。\n']
f = open('D://c.txt','w')
f.writelines(ls)
f.close()
```

数据的组织维度

一组数据在被计算机处理前需要进行一定的组织，表明数据之间的基本关系和逻辑，进而形成“数据的维度”。根据数据的关系不同，数据组织可以分为：一维数据、二维数据和高维数据。

1.一维数据的存储

1、一维数据是最简单的数据组织类型，由于是线性结构，在Python语言中主要采用列表形式表示。

一维数据的文件存储有多种方式，总体思路是采用特殊字符分隔各数据。常用存储方法包括4种。

（1）采用空格分隔元素，例如：北京　上海　天津　重庆

（2）采用逗号分隔元素，例如：北京,上海,天津,重庆

（3）采用换行分隔包括，例如：

北京

上海

天津

重庆

（4）其他特殊符号分隔，以分号分隔为例，例 如：

北京;上海;天津;重庆

1、逗号分割的存储格式叫做CSV格式（Comma-Separated Values，即逗号分隔值），它是一种通用的、相对简单的文件格式，在商业和科学上广泛应用，大部分编辑器都支持直接读入或保存文件为CSV格式

2、列表对象输出为CSV格式文件方法如下，采用字符串的join()方法最为方便。

```
ls = ['北京','上海','天津','重庆']
f = open('city.csv','w')
f.write(','.join(ls)+'\n')
f.close()
```

3、对一维数据进行处理首先需要从CSV格式文件读 入一维数据，并将其表示为列表对象。

```
f = open('city.csv','r')
ls = f.read().strip('\n').split(',')
print(ls)
f.close()
```

2.二维数据的存储

二维数据 →

```
ls = [
    ['学校','报考人数','往年录取人数','理科人数'],
    ['xx实验中学','100','60','60'],
    ['xx中学','150','30','80'],
    ['xx高级中学','200','140','160']
    ]
```

1、二维数据由一维数据组成，用CSV格式文件存储。 CSV文件的每一行是一维数据，整个CSV文件是 一个二维数据。

2、二维列表对象输出为CSV格式文件方法如下，采用遍历循环和字符串的join()方法相结合。

```
ls = [
    ['学校','报考人数','往年录取人数','理科人数'],
    ['xx实验中学','100','60','60'],
    ['xx中学','150','30','80'],
    ['xx高级中学','200','140','160']
    ]
f = open('school.csv','w')
for row in ls:
    f.write(','.join(row)+'\n')
f.close()
```

3、对二维数据进行处理首先需要从CSV格式文件读 入二维数据，并将其表示为二维列表对象。借鉴 一维数据读取方法，从CSV文件读入数据的方法 如下。

```
f = open('school.csv','r')
ls = []
for line in f:
    ls.append(line.strip('\n').split(','))
print(ls)
f.close()
```

4、二维数据处理等同于二维列表的操作，与一维列表不同，二维列表一般需要借助循环遍历实现对每个数据的处理，基本代码格式如下：

```
for row in ls:
    for item in row:
        <对第row行第item列元素进行处理>
```

对二维数据进行格式化输出，打印成表格形状：

```
f = open('school.csv','r')
ls = []
for line in f:
    ls.append(line.strip('\n').split(','))
for row in ls:
    line = ""
    for item in row:
        line += "{:10}\t".format(item)
    print(line)
f.close()
```