第7章 文件

内容提要

CONTENTS

1.文件概述

文件是一组存储在外部存储介质（磁盘、光盘、U盘等）上的信息集合，可以包含任何数据内容。操作系统是以文件的形式来管理外部存储介质上的文件，并以“文件名.扩展名”的形式标识文件。从文件内容的表现形式看，分为文本文件和二进制文件两种形式。

1.文件概述

01

文本文件是指以ASCII码方式(也称文本方式)存储的文件，更确切地说，英文、数字等字符存储的是ASCII码，而汉字存储的是机内码。

02

二进制文件是按二进制的编码方式来存储文件，没有统一的字符编码，是字节流。

1.文件概述

【例7-1】文本文件和二进制文件内容对比(文件文件 f.txt的内容为“Python语言程序设计”)

```
#exp7-1.py
fp=open("f.txt","r")
print(" 文本文件方式： ",end=' ')
print(fp.readline())
fp.close()
fp=open("f.txt","rb")
print("二进制文件方式： ",end=' ')
print(fp.readline())
fp.close()
```

运行结果：

文本文件方式：Python语言程序设计

二进制文件方式：b'Python\xd3\xef\xd1\xd4\xb3\xcc\xd0\xf2\xc9\xe8\xbc\xc6'

文件操作流程

Python程序对文件的读写过程操作步骤如下：

（1）建立/打开文件：当为了进行读操作而打开文件时，若文件存在则打开，若文件不存在则报错；当为了进行写操作而打开文件时，若文件存在则将其覆盖，若文件不存在则系统会重新创建这个文件。

（2）文件读写：从指定文件读取数据，或将内存中的数据（如变量或序列值）写入文件中。

（3）关闭文件：文件操作结束，务必关闭文件，以取消程序与指定文件间的联系。

文件的打开和关闭

打开格式：

文件对象= open(<文件名>，<打开模式>)

打开模式

- r 模式: 文件必须存在
- w 模式: 文件不存在则创建，文件存在则清空
- a 模式: 追加内容到文件尾
- 与+组合: 可读可写
- 与b组合: 以二进制方法打开，否则以文本文件方式打开

文件的打开和关闭

文件的打开模式

模式	功 能 描 述
r	只读方式打开文本文件，只允许读数据
w	只写方式打开或建立文本文件，只允许写数据
a	追加打开一个文本文件，并在文件结尾添加数据
rb	只读方式打开二进制文件，只允许读数据
wb	只写方式打开或建立二进制文件，只允许写数据
ab	追加打开一个二进制文件，并在文件结尾添加数据
r+	读写方式打开一个文本文件，允许读和写
w+	读写方式打开或建立一个文本文件，允许读和写
a+	读写方式打开一个文本文件，允许读或在文件结尾添加数据
rb+	读写方式打开一个二进制文件，允许读和写
wb+	读写方式打开或建立一个二进制文件，允许读和写
ab+	读写方式打开一个二进制文件，允许读或在文件结尾添加数据

文件的打开和关闭

关闭格式：

文件对象. close()

```
fp=open("D:\\f.txt","r")
fp.close()
```

注：若没有主动关闭，则当程序结束后由系统自动关闭，建议还是在程序中主动关闭。

2.文件的基本操作

■ 文件的读

➢ read()方法

语法格式：str=文件对象.read([size])

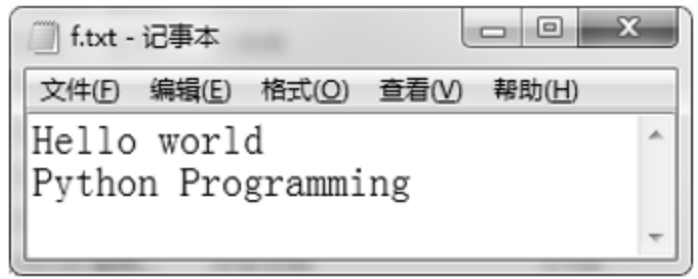
功能：从用读模式（r 或r+）打开的文件中读取指定的字节数，若是文本文件则返回字符串，若是二进制文件，则返回字节流。size为负数或空时，则读取到文件末尾。

```
fp=open("D:\\f.txt","r")
s1=fp.read(5)
```

2.文件的基本操作

■示例-read()方法

```
#exp7-2.py
fp=open("D:\\f.txt","r")
s1=fp.read(5)
print("前5个字符为:",s1)
fp.seek(0)       #指针回到文件开头
s2=fp.read()
print("所有字符为:",s2)
fp.close()
```



运行结果：
前5个字符为: Hello
所有字符为: Hello world
Python Programming

2.文件的基本操作

■文件的读

➢ readline ()方法

语法格式：str=文件对象.readline([size])

功能：从用读模式（r 或r+）打开的文件中从当前位置读取到行末，多用于文本文件。若size缺省或大于本行字符串长度，则读取到本行末尾（含'\n'）。

```
fp=open("D:\\f.txt","r")
s1=fp.readline()
print("首行字符为:",s1)
```

2.文件的基本操作

■示例-readline ()方法

```
#exp7-3.y
fp=open("D:\\f.txt","r")
s1=fp.readline()
print("首行字符为:",s1)
s2=fp.readline(6)
print("第二行前6个字符为:",s2)
fp.close()
```

运行结果：

首行字符为：Hello world

第二行前6个字符为：Python

2.文件的基本操作

■ 文件的读

➢ readlines()方法

语法格式：str=文件对象.readlines([size])

功能：从用读模式（r 或r+）打开的文件中从当前位置读取多行数据，多用于文本文件。若size缺省则读取到文件末尾。若size小于本行字符长度，则读取到本行末尾。该方法返回从文件读取的每行内容组成的列表，每行都包括'\n'。

```
fp=open("D:\\f.txt","r")
s1=fp.readlines()
print("文件内容为:",s1)
```

2.文件的基本操作

■ 示例-readlines()方法

```
#exp7-4.py
fp=open("D:\\f.txt","r")
s1=fp.readlines()
print("文件内容为:",s1)
fp.seek(0) #指针回到文件开头
s2=fp.readlines(11) #只读取第一行内容，超过11则第二行内容也被读出
print("文件第一行内容为:",s2)
```

运行结果：

```
文件内容为: ['Hello world\n', 'Python Programming']
文件第一行内容为: ['Hello world\n']
```

2.文件的基本操作

■文件的写

➢ write()方法

语法格式：文件对象.write(字符串)

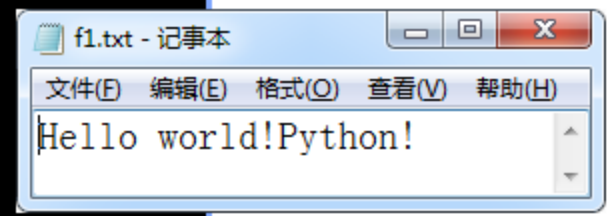
功能：向用写模式（w或w+）打开的文件的当前位置写入字符串。若是文本文件则写入字符串，若是二进制文件，则写入字节流。write()方法返回写入的字符数或字节数。write()方法不在字符串的结尾添加换行符'\n'。

```
fp=open("D:\\f1.txt","w")
len=fp1.write("Hello world!")
print("写入内容1长度为:",len)
```

2.文件的基本操作

■ 示例—write()方法

```
#exp7-5.py
fp1=open("D:\\f1.txt","w")
len1=fp1.write("Hello world!")
fp1.write("Python!")
print("写入内容1长度为:",len1)
fp1.close()
fp2=open("D:\\f2.dat","wb")
len2=fp2.write(bytes([1,2,3]))  #bytes()函数用于将参数转换为字节序列
print("写入内容2长度为:",len2)
fp2.close()
```



运行结果：

```
写入内容1长度为： 12
写入内容2长度为： 3
```

2.文件的基本操作

■文件的写

➢ writelines()方法

语法格式：文件对象.writelines(列表)

功能：向用写模式（w或w+）打开的文件的当前位置写入列表中的所有元素，多用于文本文件。

```
fp = open("D:\\test.txt", "w")
print("文件名为: ", fp.name)
seq = ["Python教程 1\n", "Python教程 2"]
fp.writelines(seq)
```

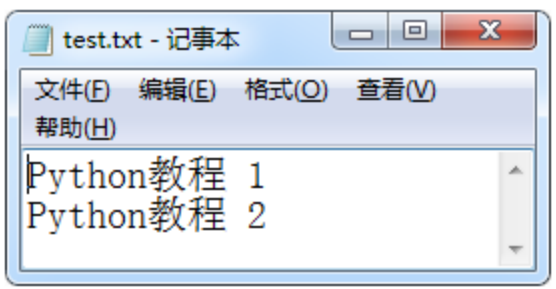
2.文件的基本操作

■示例—writelines()方法

```
#exp7-6.py
fp = open("D:\\test.txt", "w")
print("文件名为: ", fp.name)
seq = ["Python教程 1\n", "Python教程 2"]  #换行时需要指定换行符'\n'
fp.writelines(seq)
fp.close()
```

运行结果：

```
文件名为:  D:\test.txt
```



2.文件的基本操作

■文件的定位

➢ seek ()方法

语法格式：文件对象. seek (offset,whence=0)

功能：参数offset为相对于所指示位置的字节偏移量；whence表示所指示的位置，默认值为0，表示相对于文件开始位置，值为1时，相对于文件读写位置，值为2时，相对于文件结尾。seek ()方法的返回值为当前的读写位置。

```
fp = open("D:\\f.txt", "r")
str1=fp.read()
print("文件内容为:",str1)
fp.seek(7,0)  #以文件开始为基准，向文件尾方向移动7个字节
```

2.文件的基本操作

■ 定位示例—seek ()方法

```
#exp7-7.py
fp = open("D:\\f.txt", "r")
print("文件名为:", fp.name)
str1=fp.read()
print("文件内容为:",str1)
fp.seek(7,0) #以文件开始为基准，向文件尾方向移动7个字节
str2=fp.read()
print("第7个字节后的内容为:",str2)
fp.close()
```

运行结果：

```
文件名为: D:\f.txt
文件内容为: Python Programming
第7个字节后的内容为: Programming
```

2.文件的基本操作

■文件的定位

➢ tell()方法

语法格式：文件对象. tell ()

功能：返回文件的当前位置（相对于文件开始的位置）。

```
fp = open("D:\\f.txt", "r")
str1=fp.read()
print("文件内容为:",str1)
print("文件指针当前位置:",fp.tell()) #当前位置在文件末尾
fp.seek(0,0)   #回到文件起始位置
str2=fp.read(6)
print("文件指针的当前位置为:",fp.tell()) #读取6个字节内容之后的位置
```

运行结果：

```
文件内容为: Python Programming
文件指针当前位置: 18
文件指针的当前位置为: 6
```

3.文件相关模块

当需要对文件和文件夹进行复杂操作时，就需要Python内置的的os模块和shutil模块。

(1)mkdir()方法

语法格式：os. mkdir (path)

功能：按path指定的路径创建单级目录，若目录存在则抛出异常。

```
#exp7-9.py
import os  #导入os模块
os.mkdir("D:\\abc")
print("指定位置目录创建成功！")
os.mkdir(“python”) #在程序所在目录下创建
print("当前文件夹下目录创建成功！")
```

运行结果：

```
指定位置目录创建成功！
当前文件夹下目录创建成功！
```

3.文件相关模块

(2)makedirs()方法

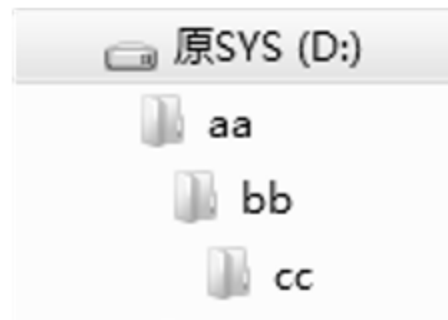
语法格式：os. makedirs (path)

功能：按path指定的路径递归地创建多级目录，若目录存在则抛出异常。

```
#exp7-10.py
import os
os.makedirs("D:\\aa\\bb\\cc")
print("多级目录创建成功！")
```

运行结果：

多级目录创建成功！



3.文件相关模块

(3) listdir()方法

语法格式：os. listdir (path)

功能：列出path指定的路径下所有的文件和目录，结果以列表形式呈现。

```
>>> import os
>>> os.listdir("D:\\")
['$360Section', '$RECYCLE.BIN', '1.txt', '1020.log', '360SANDBOX', '8461.swf', 'aa', 'abc',
'Activator_v1.2.exe', 'ahcre', 'Boot', 'bootmgr', 'coil20_64x64.mat', 'Config.Msi', 'CZWRG',
'Documents and Settings', 'f.txt', 'f1.txt', 'f2.dat', 'KwDownload', 'LenovoDrivers',
'ludashi_5.15.16.1285.exe', 'MSOCache', 'pagefile.sys', 'PerfLogs', 'Program Files', 'Program
Files (x86)', 'ProgramData', 'QMDownload', 'QQPCMgr', 'System Volume Information',
'test.txt', 'Users', 'WeChatSetup.exe', 'Windows', '冒泡法.swf']
```

3.文件相关模块

(4)getcwd()方法

语法格式：os. getcwd ()

功能：显示当前工作目录。

(5)chdir()方法

语法格式：os. chdir (path)

功能：按path指定路径改变当前工作目录。

```
>>> import os
>>> os.chdir("D:\\abc")
>>> os.getcwd()
'D:\\abc'
```

3.文件相关模块

(6)rmdir()方法

语法格式：os. rmdir (path)

功能：按path指定路径删除目录，要求被删除目录需为空目录，否则抛出异常。

```
>>> os.rmdir("D:\\aa\\bb\\cc")  #将删除cc文件夹
```

(7) remove()方法

语法格式：os. remove (file)

功能：删除指定文件，若文件不存在，则抛出异常。

```
>>> os.remove("D:\\abc\\f.txt")  #删除指定文件f.txt
```

3.文件相关模块

(8) rename()方法

语法格式：os. rename (src, dst)

功能：将文件或目录src重命名为dst，若dst存在，则抛出异常。

```
>>> os.chdir("D:\\abc")
>>> os.rename("f.txt","g.txt")  #将文件"f.txt"重命名为"g.txt"
```

3.文件相关模块

(9) copyfile()方法

语法格式：shutil.copyfile(源文件,目标文件)

功能：需要导入shutil模块，将源文件拷贝到目标文件。

```
>>> import shutil
>>> shutil.copyfile("D:\\abc\\g.txt","D:\\abc\\f.txt")
```

小结

- 文件概述:文件有文本文件和二进制文件之分；
- 文件的打开和关闭：介绍了文件打开和关闭方法，重点介绍了文件的打开模式；
- 文件的基本操作：介绍了用于文件读写的方法：read()、readline()、readlines()、write()、writelines()等，还介绍了文件定位方法：seek()和tell()；
- 文件相关模块：介绍了Python内置os模块和shutil模块的用法。