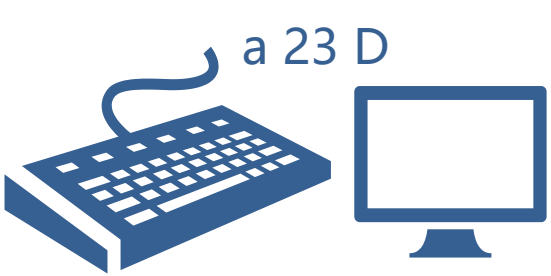
Chap7 Files

第7章 文件

Department of Computer Science and Technology
Department of University Basic Computer Teaching
Nanjing University

程序中的数据



or

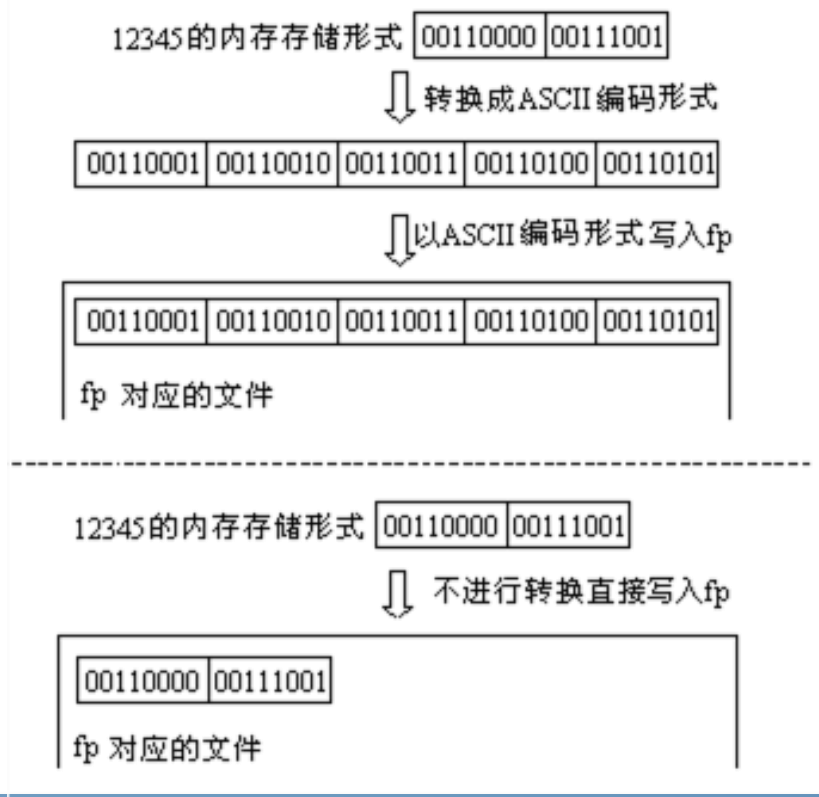
7.1 文件基本概念

文件基本概念

- 文件：存储在某种介质上的信息集合
- 存储：外部介质
- 识别：文件名
- 分类
  - 存取方式：顺序存取，随机存取
  - 文件内容表示方式：二进制文件，文本文件

7.1.1 PYTHON文件系统

二进制文件与文本文件


12345的内存存储形式 00110000 00111001
转换成ASCII编码形式
00110001 00110010 00110011 00110100 00110101
以ASCII编码形式写入fp
00110001 00110010 00110011 00110100 00110101
fp 对应的文件
12345的内存存储形式 00110000 00111001
不进行转换直接写入fp
00110000 00111001
fp 对应的文件

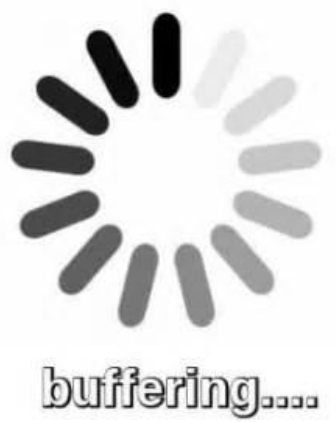
二进制文件与文本文件

- 用二进制形式输出时
  - 可节省外存空间和转换时间
  - 一个字节并不对应一个字符，不能直接输出字符形式。
  - 可读性差，常用于保存中间结果数据和运行程序。

- 文本形式输出时
  - 一个字节与一个字符一一对应
  - 便于对字符进行逐个处理，也便于输出字符；
  - 占存储空间较多；
  - 要花费转换时间。

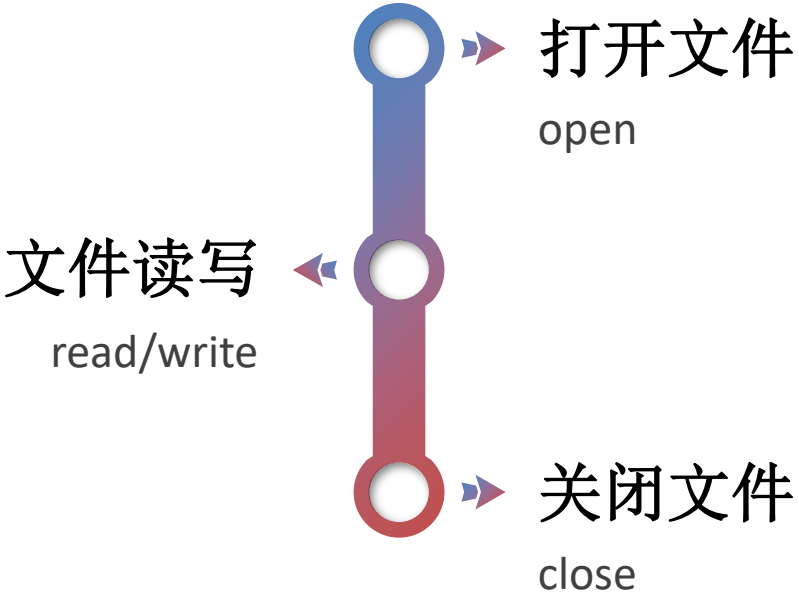
二进制文件与文本文件

- Python中可以处理二进制文件以及文本文件，对二进制文件的操作可以选择是否使用缓冲区
- 缓冲区是内存中的区域，当程序中需要进行频繁的文件读写操作时，使用缓冲区可以减少I/O操作从而提高效率，也方便管理
- 文本文件均使用缓冲区处理



7.1.2 文件的使用过程

文件的使用过程



7.2 文件的打开和关闭

7.2.1 文件的打开

文件的打开

Source

```
>>> f1 = open('d:\\infile.txt')
>>> f2 = open('d:/outfile.txt', 'w')
>>> f3 = open('frecord.csv', 'ab', 0)
```

open()函数返回一个文件（file）对象

file_obj = open(filename, mode='r', buffering=-1)

- mode为可选参数，默认值为r
- buffering也为可选参数，默认值为-1（0代表不缓冲，1或大于1的值表示缓冲一行或指定缓冲区大小）
- 其他常用参数：encoding（指定编码字符集）

open()函数-mode

Mode	Function
r	以读模式打开，文件必须存在
w	以写模式打开，若文件不存在则新建文件，否则清空原内容
x	以写模式打开，若文件已经存在则失败
a	以追加模式打开，若文件存在则向结尾追加内容，否则新建文件
r+	以读写模式打开
w+	以读写模式打开（清空原内容）
a+	以读和追加模式打开
rb	以二进制读模式打开
wb	以二进制写模式打开（参见w）
ab	以二进制追加模式打开（参见a）
rb+	以二进制读写模式打开（参见r+）
wb+	以二进制读写模式打开（参见w+）
ab+	以二进制读写模式打开（参见a+）

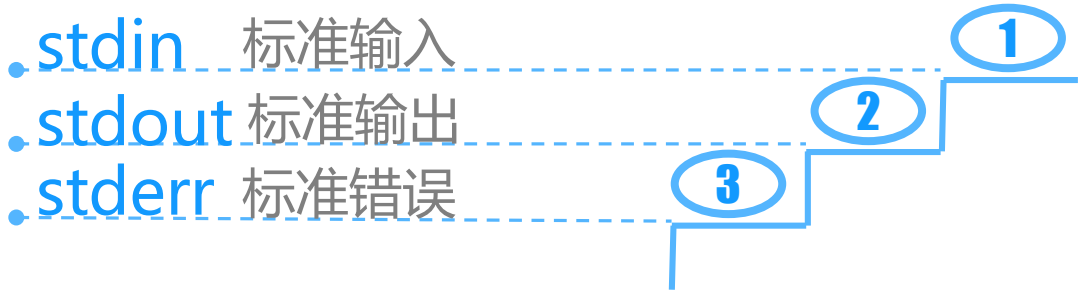
标准文件

- Python把设备抽象成文件，因此键盘和终端（显示器）也对应于文件对象。当一个程序开始运行时，Python自动为这个程序打开3个文件

标准文件	文件对象	对应的设备
标准输入	stdin	键盘
标准输出	stdout	显示器
标准错误	stderr	显示器

标准文件

- 当程序启动后，以下三种标准文件有效



Source

```
>>> newcName = input('Enter the name of new  company: ')
Enter the name of new company: Chaolihai
>>> print(newcName)
Chaolihai
```

7.2.2 文件的关闭

关闭文件

- **fp.close()**
  - fp为文件对象
  - 切断文件对象与外存储器中文件之间的联系

Source

```
>>> fp = open(r'd:\nfile.txt', 'r')
>>> type(fp)
<class '_io.TextIOWrapper'>
>>> fp.name
'd:\\nfile.txt'
>>> fp.mode
'r'
>>> fp.closed
False
>>> fp.close()
>>> fp.closed
True
```

关闭文件

- 文件使用完后如果不关闭，则当程序运行结束时由系统自动关闭
- 不建议使用系统自动关闭的原因
  - 操作系统允许程序同时打开的文件个数是有限的
  - 写入内容已处理完若缓冲区还未满，缓冲区的内容要等到程序运行结束时由系统自动关闭该文件后才能写出，此时若系统发生非正常情况当前缓冲区中的未写到外存储上的内容就可能丢失掉

7.3 文件的基本操作

文件的基本操作

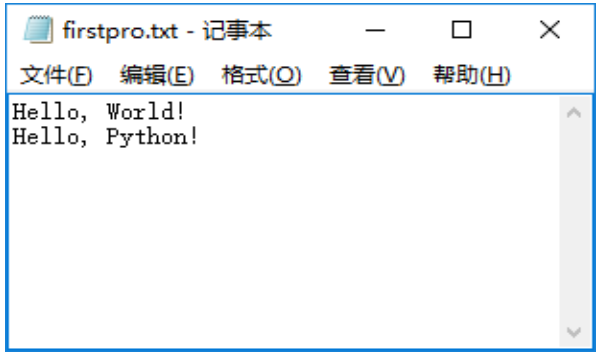
返回值和基本操作

- open()函数返回一个文件（file）对象
- 文件对象可迭代
- 有许多读写相关的方法/函数
  - f.read(), f.write(), f.readline(), f.readlines(), f.writelines()
  - f.seek()

7.3.1 文件的读写

读文件-read()方法

- s = fp.read(size)
  - 从文件当前位置读取size字节数据，若size为负数或空，则读取到文件结束
  - 返回一个字符串（文本文件）或字节流（二进制文件）



Source

```
>>> fp = open(r'd:\firstpro.txt')
>>> s = fp.read(5)
>>> print(s)
Hello
>>> s = fp.read()
>>> s
', World!\nHello, Python!'
>>> fp.close()
```

π中有你的生日吗


pi.txt - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
3. 14159265358979323846264338327950288419716939937510582097494459230781640628620899862803482534211706798214
08651328230664709384460955058223172535940812848111745028410270193852110555964622948954930381964428810975665
59334461284756482337867831652712019091456485669234603486104543266482133936072602491412737245870066063155881
74881520920962829254091715364367892590360011330530548820466521384146951941511609433057270365759591953092186
11738193261179310511854807446237996274956735188575272489122793818301194912983367336244065664308602139494639
52247371907021798609437027705392171762931767523846748184676694051320005681274156235608277857713427577896091
73637178721468440901224953430146549585371050792279689238923542019956112129021960864034418139813629774771309
96051870721134999999837297804995105973173281609631859502445945534690830264252230825334468503526193118817101
00031378387528866587533208381420617177669147303598253490428755468731159562863882353787593751957781857805321
71226806613001927876611195909216420198938095257201065485863278865936153381827968230301952053301852968955773
62259941389124972177528347913151557485724245415069595082953311686172785588907509838175463746493931925506040
09277016711390098488240128583616035637076601047101819429555961989467678374494482553797747268471040475346462
08046684259069491293313677028989152104752162056696602403803815019351125338243003558764024749647326391419927 2
60426992279678235478163600934172164129924586315030286182974555706749838505494588586926995690927210797509 30
29553211653449872027559602364806654991198818347977535663698074265425278625518184175746728909777727938000816
47060016145249192173217214772350141441975568548161361157352552133475741849468438523323907394143334547762416
86251898356948556209921922218427255025425688767179049460165346680498862723279178608578438382796797668145410
09538837863609506806064225125205117392984896084128488626945604241965285022210661186306744278622039194950471
23713786960956364371917287467764657573962413890865832645995813390478027590099465764078951269468398352595709
82582262052248940772671947826848260147699090264013639443745530506820349625245174939965143142980919065925093
72216964615157098583874105978859597729754989301617539284681382668638689427741559918559252459539594310499725
24680845987273644695848653836736222626099124608051243884390451244136549762780797715691435997700129616089441
69486855584840635342207222582648864815845602850601684273945226746767889525213852259956666727823986456596 11
63548862305774564980359363456817432411251507606947945109659690402522288797108931456691368672287489405601015
03308617926808920874760917824938589009714909675985261365549781893129784821682998948722658880485756401427047 7
55513237964145152374623436 4
54285844479526586782105114135473573952311342716610213596953623144295248493718711014576540359027993440374200
73105785390621983874478084788968332144571386875194350643021845319104848100537061468067491927819179739952 0
61419663428754440643745123718192179998391015919561814675142691239748940907186494231961567945208095146550225
23160388193014209376213785595663893778708303906979207734672218256259966150142150306803844773454920260541466

读文件-readline()方法

- **s = fp.readline(size=-1)**
  - 从文件当前位置读取本行内size字节数据，若size为默认值或大小超过当前位置到行尾字符长度，则读取到本行结束（包含换行符）
  - 返回读取到的字符串内容

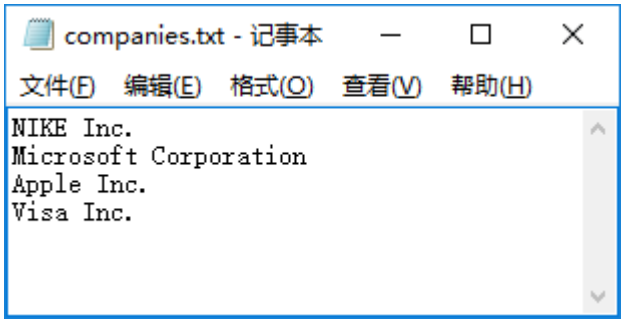
firstpro.txt :
Hello, World!
Hello, Python!

Source

```
>>> fp = open(r'd:\firstpro.txt')
>>> s = fp.readline(20)
>>> s
'Hello, World!\n'
>>> s = fp.readline(2)
>>> s
'We'
>>> s = fp.readline()
>>> s
'lcome'
>>> fp.close()
```

读文件-readlines()方法

- **lines = fp.readlines(hint=-1)**
  - 从文件当前读写位置开始读取需要的字节数，至少为一行；若hint为默认值或负数，则读取从当前位置到文件末尾的所有行（包含换行符）
  - 返回从文件中读出的行组成的列表



Source

```
>>> fp = open(r'd:\companies.txt')
>>> lines = fp.readlines(2)
>>> lines
['NIKE Inc.\n']
>>> lines = fp.readlines()
>>> lines
['Microsoft Corporation\n', 'Apple Inc.\n', ' Visa Inc.']
>>> fp.close()
```

写文件-write()方法

- **fp.write(s)**
  - 向文件中写入数据（字符串或字节流）
  - 返回写入的字符数或字节数

Source

```
>>> fp = open(r'd:\firstpro.txt', 'w')
>>> fp.write("Hello, World!\n")
14
>>> fp.write("Hello, Python!")
14
>>> fp.close()
```

Source

```
>>> f = open(r'd:\firstpro.dat', 'wb')
>>> x = bytes([3, 4, 5])
>>> f.write(x)
>>> f.close()
```

写文件-writelines()方法

- **fp.writelines(*lines*)**
  - 向文件中写入列表数据，多用于文本文件

Source

```
>>> fp = open(r'd:\companies1.txt', 'w')
>>> lines = ['NIKE Inc.\n', 'Microsoft Corporation\n', 'Apple Inc.\n', 'Visa Inc.\n']
>>> fp.writelines(lines)
>>> fp.close()
```

文件读写例子

将文件companies.txt 的字符串前加上序号1、2、3、…后写到另一个文件scompanies.txt中。

File

```
# Filename: prog7-1.py
f1= open('companies.txt')
lines = f1.readlines()
f1.close()
for i in range(len(lines)):
    lines[i] = str(i+1) + ' ' + lines[i]
f2 = open('scompanies.txt', 'w')
f2.writelines(lines)
f2.close()
```

Output:

1 GOOGLE Inc.
2 Microsoft Corporation
3 Apple Inc.
4 Facebook, Inc.

7.3.2 文件的定位

文件的定位-seek()方法

- **fp.seek(*offset , whence=0*)**
  - fp打开的文件必须允许随机访问
  - 在文件中移动文件指针，从*whence*（0表示文件头部，1表示当前位置，2表示文件尾部）偏移*offset*个字节
  - *whence*参数可选，默认值为0
  - 返回当前的读写位置

文件的定位-seek()方法

Source

```
>>> fp = open('testseek.dat', 'wb+')
>>> fp.write(b'Hello,word!')
11
>>> fp.seek(0)
0
>>> s = fp.read(5)
>>> s
b'Hello'
>>> fp.seek(-5, 2)
6
>>> s = fp.read()
```

Source

```
>>> s
b'word!'
>>> fp.seek(3, 0)
3
>>> fp.read(3)
b'lo,'
>>> fp.seek(2, 1)
8
>>> fp.read(3)
b'rd!'
```

文件读写例子改写

将文件companies.txt 的字符串前加上序号1、2、3、…。


```
# Filename: prog7-2.py
f= open('companies.txt', 'r+')
lines = f.readlines()
for i in range(len(lines)):
    lines[i] = str(i+1) + ' ' + lines[i]
f.seek(0)
f.writelines(lines)
f.close()
```

Output:

1 GOOGLE Inc.
2 Microsoft Corporation
3 Apple Inc.
4 Facebook, Inc.

文件的定位-tell()方法

- **fp.tell()**
  - 返回文件的当前读写位置

Source

```
>>> fp = open('testseek.dat', 'rb+')
>>> fp.tell()
0
>>> fp.read(5)
b'Hello'
>>> fp.tell()
5
>>> fp.close()
```

7.3.3 文件的其他操作

文件的其他方法及属性

方法	功能
f.flush()	将写缓冲区的数据写入文件
f.truncate(size=None)	将文件截取为给定大小的字节（如果未指定大小，则为当前文件读写位置）。当前文件读写位置没有改变
f.closed	文件关闭属性，当文件关闭时为True，否则为False
f.fileno()	返回文件描述符（整数）
f.readable()	判断文件是否可读，是返回True，否返回False
f.writable()	判断文件是否可写，是返回True，否返回False
f.seekable()	判断文件是否支持随机访问，是返回True，否返回False
f.isatty()	判断文件是否交互（如连接到一个终端设备），是返回True，否返回False

文件异常处理

Source

```
with open('companies.txt', 'r+') as f:
    lines = f.readlines()
    for i in range(len(lines)):
        lines[i] = str(i+1) + ' ' + lines[i]
    f.seek(0)
    f.writelines(lines)
```

文件综合例

读取成绩文件score.txt，分别计算语文、数学、英语的最高分、最低分、平均分并输出到显示终端上，同时在score.txt添加一列计算出每位同学的总成绩。

```
姓名,语文,数学,英语
陈纯,88,87,85
方小磊,93,88,90
王妤,82,99,96
彭子晖,97,94,84
丁海斌,97,94,76
```

⟹

```
姓名,语文,数学,英语,总分
陈纯,88,87,85,260
方小磊,93,88,90,271
王妤,82,99,96,277
彭子晖,97,94,84,275
丁海斌,97,94,76,267
```

文件综合例

File

```
# Filename: prog7-3.py
def f(listdata, n, flag = 'max'):
    m = listdata[0][n]
    if flag == 'max':
        for item in listdata:
            if item[n] > m:
                m = item[n]
    elif flag == 'min':
        for item in listdata:
            if item[n] < m:
                m = item[n]
    elif flag == 'ave':
        m = 0
        for item in listdata:
                m += item[n]
        m /= len(listdata)
    return m
```

File

```
def stas(x, field):
    print('     {:5s}{:5s}{:5s}'.format(field[1],field[2],field[3]))
    print('最高分:{:5d}{:5d}{:5d}'.format(f(x, 1), f(x, 2), f(x, 3)))
    print('最低分:{:5d}{:5d}{:5d}'.format(f(x, 1, 'min'), f(x, 2,
                'min'), f(x, 3, 'min')))
    print('平均分:{:5.1f}{:5.1f}{:5.1f}'.format(f(x, 1, 'ave'), f(x,
                2,'ave'),f(x, 3, 'ave')))
```

文件综合例


```
if __name__ == "__main__":
    with open("score.txt", 'r+') as fp:
        lines = fp.readlines()
        field = lines.pop(0).split(',')
        x = []
        for eachline in lines:
            t = eachline.split(',')
            for j in range(1, len(t)):
                t[j] = int(t[j])
            x.append(t)
        stas(x, field)
        for i in range(len(x)):
            lines[i] = lines[i].strip() + ',' + str(x[i][1] + x[i][2] + x[i][3]) + '\n'
        field[-1] = field[-1].strip()
        field.append('总分\n')
        lines.insert(0, ','.join(field))
        print(lines)
        fp.seek(0)
        fp.writelines(lines)
```

小结

- 文件的打开
- 文件打开模式与文件类型
- 文件的关闭
- 文件的读写
- 文件定位