# [python文件流](https://www.cnblogs.com/whych/p/9678249.html)

打开文件

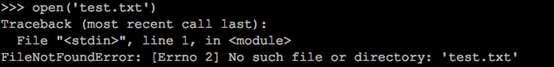
文件的基本方法

迭代文件内容

## 打开文件

打开文件，可以使用自动导入的模块io中的函数open。函数open将文件名作为唯一必不可少的参数，并返回一个文件对象。如果只指定一个文件名，则获得一个可读取的文件对象。

如果当前目录中有此文件则可以打开，如果位于其他位置则需要指定完整的路径，如果指定文件不存在，则如下报错。



**文件模式**

如果需要写入文件，则需要通过指定模式。函数open的参数模式常见有如下：

|  |  |
| --- | --- |
| **模式** | **描述** |
| 'r' | 读取模式（默认） |
| 'w' | 写入模式（文件不存在时创建它） |
| 't' | 文本模式（默认，与其他模式结合使用） |
| 'x' | 独占写模式，新建一个文件，如果该文件已存在则会报错。 |
| 'a' | 附加模式（在既有文件末尾继续写入） |
| 'b' | 二进制模式（与其他模式结合使用） |
| '+' | 打开一个文件进行更新（可读可写，与其他模式结合使用） |

默认模式为'rt'，读取时将自动替换其他行尾字符（'\r','\r\n'），写入时将'\n'替换为系统的默认行尾字符（os.linesep）

要打开一个文本文件进行读写，可使用'r+',但是会将既有内容删除，而'w+'不会。

## 文件的基本方法

读取和写入

管道重定向输出

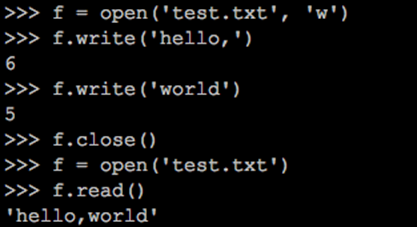
随机存取

读取和写入行

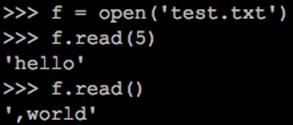
关闭文件

文件最重要的功能就是提供和接收数据。在文本和二进制模式下，基本上分别将str和bytes类用作数据。

**读取和写入：**



使用read读取数据，默认读取全部内容。还可以指定参数，然后读取剩下的全部内容：



**管道重定向输出**

将一个命令的标准输出链接到下个命令的标准输入：

#cat some.txt

you are bad boy

#cat some.py

#!/usr/bin/evn python3

# 计算sys.stdin中包含多少个单词的脚本

import sys

text = sys.stdin.read()

words = text.split()

wordcount = len(words)

print(wordcount)

#cat some.txt | python3 some.py

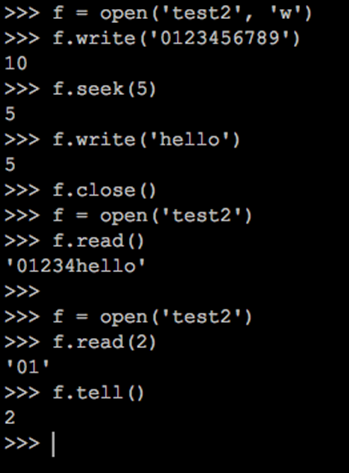
4

**随机存取**

可以使用方法seek和tell。

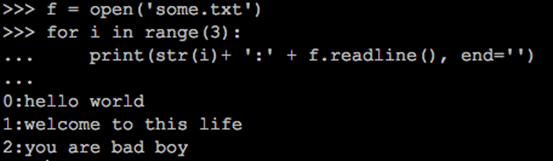
seek(offset,[, whence])将当前位置移到offset（指定字节数）和whence指定的地方（参数whence默认为io.SEEK\_SET(0)，偏移量是相对于文件开头的，而io.SEEK\_SET(1)，相对于当前位置进行移动，io.SEEK\_SET(2)，相对于文件末尾进行移动）

tell()返回当前位于文件的什么位置。



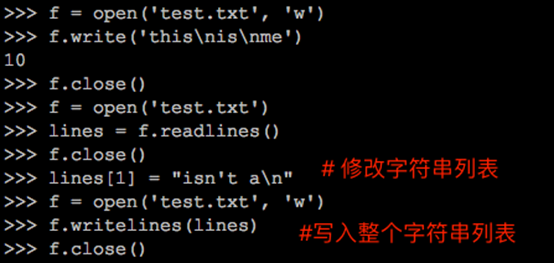
**对行的读取和写入**

可以使用方法readline，不指定参数默认读取一行并返回，指定参数为最多读取多少个字符。要读取文件中所有的行，并**以列表的方式返回**它们，可以使用方法readlines。方法writelines**接受一个字符串列表**写入文件中。



https://img2018.cnblogs.com/blog/1387124/201809/1387124-20180919230304006-580327125.png

# 修改文件



**关闭文件**

避免锁定文件以防止修改，避免用完系统可能指定的文件打开配额。

确保文件关闭，可以使用try/finally语句。

try：

# 将数据写入到文件中

finally：

file.close()

# with语句可以让你打开一个文件并赋值到一个变量，到达该句末尾时，将自动关闭文件，即便出现异常。

with open('somefile.txt') as somefile

　　do\_something(somefile)

## 迭代文件内容

def process(string):

print('Processing:', string)

# 每次迭代一个字符

with open('fileone.txt') as f:

　　while True:

char = f.read(1)

　　if not char: break

　　process(char)

#每次迭代一行字符

with open('fileone.txt') as f:

while True:

line = f.readline()

process(line)

#读取所有内容，使用read

with open('fileone.txt') as f: for char in f.read():

process(char)

#使用readlines

with open('fileone.txt') as f: for char in f.readlines():

process(char)

# 使用fileinput实现延迟迭代，读取实际需要文本的部分

import fileinput

for line in fileinput.input('fileone.txt'):

process(line)

**文件迭代器**

迭代文件

with open('fileone.txt') as f:  
　　for line in f:  
　　process(line)

不将文件对象赋给变量迭代文件

for line in open(fileone.txt):

process(line)

对迭代器的操作也可以对文件做，如list(open(fileone.txt))：

>>> f = open('fileone.txt', 'w')

>>> print('First', 'line', file=f)

>>> print('Second', 'line', file=f)

>>> f.close()

>>> lines = list(open('fileone.txt'))

>>> lines

['First line\n', 'Second line\n']

❄

❄

❄

❄

❄

❄

❄

❄

❄

❄