# 1. 文件操作介绍

## 1.1. 什么是文件

示例如下：



## 1.2. 文件的作用

大家应该听说过一句话：“好记性不如烂笔头”。

不仅人的大脑会遗忘事情，计算机也会如此，比如一个程序在运行过程中用了九牛二虎之力终于计算出了结果，试想一下如果不把这些数据存放起来，相比重启电脑之后，内存的数据全部丢失，这个时候就恢复不了了。

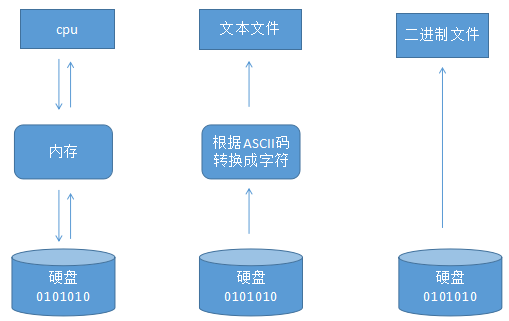
可见，在把数据存储起来有做么大的价值

使用文件的目的：

**就是把一些数据存放起来，可以让程序下一次执行的时候直接使用，加载到内存中，而不必重新制作一份，省时省力。**

## 1.3 文本文件和二进制文件

从文件编码的方式来看，文件可分为ASCII码文件和二进制码文件两种。  
  
　　ASCII文件也称为文本文件，这种文件在磁盘中存放时每个字符对应一个字节，用于存放对应的ASCII码。例如，数5678的存储形式为：  
ASC码： 　00110101 00110110 00110111 00111000  
　　　　　↓ 　　　　↓　　　　↓ 　　　↓  
十进制码： 5　　　　　6　　　　7　　　　8 共占用4个字节。ASCII码文件可在屏幕上**按字符显示**， 例如源程序文件就是ASCII文件，用DOS命令TYPE可显示文件的内容。 由于是**按字符显示**，因此能读懂文件内容。  
  
　　二进制文件是按二进制的编码方式来存放文件的。 例如， 数5678的存储形式为： 00010110 00101110只占二个字节。二进制文件虽然也可在屏幕上显示， 但其内容无法读懂.



# 2. 文件的打开与关闭

想一想：

如果想用word编写一份简历，应该有哪些流程呢？

1. 打开word软件，新建一个word文件
2. 写入个人简历信息
3. 保存文件
4. 关闭word软件

同样，在Python操作文件的整体过程与使用word编写一份简历的过程是很相似的

1. 打开文件，或者新建立一个文件
2. 读/写数据
3. 关闭文件

## 2.1. 打开文件open

在Python，使用open函数，可以打开一个已经存在的文件，或者创建一个新文件open(文件名，访问模式)

示例如下：

f = open('test.txt', 'w')

### 2.1.1. 访问模式说明

| 访问模式 | 说明 |
| --- | --- |
| r | **以只读方式打开文件。文件的指针将会放在文件的开头。这是默认模式。如果文件不存在会崩溃**。 |
| w | 打开一个文件只用于写入。**如果该文件已存在则将其覆盖**。如果该文件不存在，创建新文件。 |
| a | **打开一个文件用于追加**。如果该文件已存在，文件指针将会放在文件的结尾。也就是说，新的内容将会被写入到已有内容之后。如果该文件不存在，创建新文件进行写入。 |
| rb | **以二进制格式打开一个文件用于只读**。文件指针将会放在文件的开头。这是默认模式。 |
| wb | **以二进制格式打开一个文件只用于写入**。如果该文件已存在则将其覆盖。如果该文件不存在，创建新文件。 |
| ab | **以二进制格式打开一个文件用于追加**。如果该文件已存在，文件指针将会放在文件的结尾。也就是说，新的内容将会被写入到已有内容之后。如果该文件不存在，创建新文件进行写入。 |
| r+ | 打开一个文件用于**读写**。文件指针将会放在文件的开头。 |
| w+ | 打开一个文件用于**读写**。如果该文件已存在则将其覆盖。如果该文件不存在，创建新文件。 |
| a+ | 打开一个文件用于**读写**。如果该文件已存在，文件指针将会放在文件的结尾。文件打开时会是追加模式。如果该文件不存在，创建新文件用于读写。 |
| rb+ | 以二进制格式打开一个文件用于**读写**。文件指针将会放在文件的开头。 |
| wb+ | 以二进制格式打开一个文件用于**读写**。如果该文件已存在则将其覆盖。如果该文件不存在，创建新文件。 |
| ab+ | 以二进制格式打开一个文件用于**追加**。如果该文件已存在，文件指针将会放在文件的结尾。如果该文件不存在，创建新文件用于读写。 |

## 2.2. 关闭文件close

close( )

示例如下：

# 新建一个文件，文件名为:test.txt

f = open('test.txt', 'w')

# 关闭这个文件

f.close()

# 3. 文件的读写

## 3.1. 写数据(write)

使用write()可以完成向文件写入数据

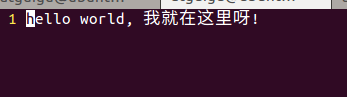
demo:

f = open('test.txt', 'w')

f.write('hello world, 我就在这里呀!')

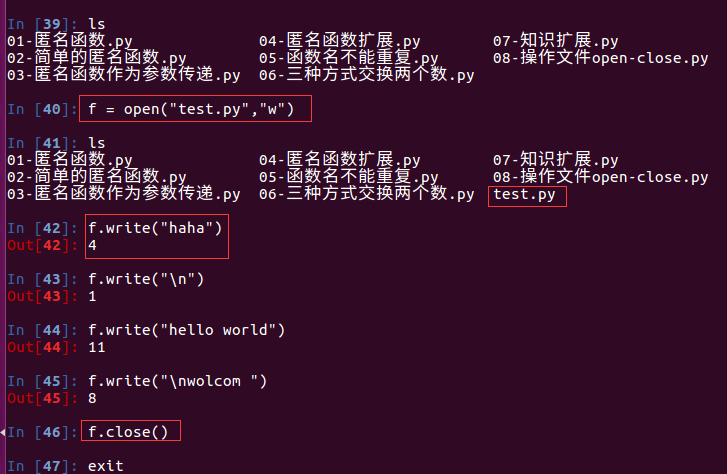
f.close()

运行现象:

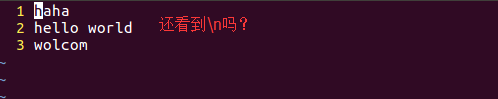


注意：

如果文件不存在那么创建，如果存在那么就先清空，然后写入数据



查看文件是有内容呢：



## 3.2. 读数据(read)

使用read(num)可以从文件中读取数据，num表示要从文件中读取的数据的长度（**单位是字节**），如果没有传入num，那么就表示读取文件中所有的数据

demo:

f = open('test.txt', 'r')

content = f.read(5)

print(content)

print("-"\*30)

content = f.read()

print(content)

f.close()



注意：

如果open是打开一个文件，那么可以不用先打开的模式，即只读 open('test.txt')，如果使用读了多次，那么后面读取的数据是从上次读完后的位置开始的

## 3.3. 读数据（readlines）

就像read没有参数时一样，**readlines可以按照行的方式把整个文件中的内容进行一次性读取**，并且返回的是一个**列表**，其中每一行的数据为一个元素

#coding=utf-8

f = open('test.txt', 'r')

content = f.readlines()

print(type(content))

i=1

for temp in content:

print("%d:%s"%(i, temp))

i+=1

f.close()

运行现象：

## 3.4. 读数据（readline）

#coding=utf-8

f = open('test.txt', 'r')

content = f.readline()

print("1:%s"%content)

content = f.readline()

print("2:%s"%content)

f.close()

## 3.5. 练一练

写两个.py文件，一个是写你想说的一句话，一个是读 取你要说的那句话。

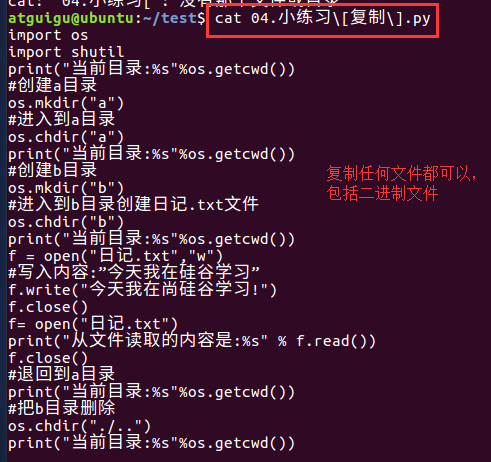
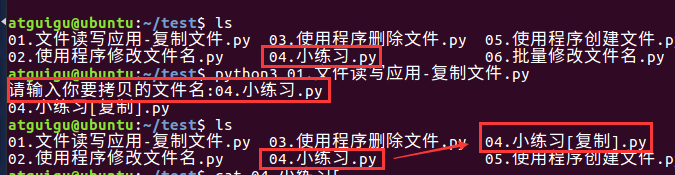
# 4. 文件读写应用-复制文件

## 4.1. 任务描述

需求描述：

输入文件的名字，然后程序自动完成对文件进行拷贝

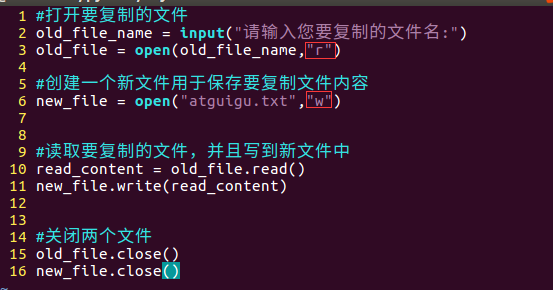
例如，程序运行后提示“请输入你要拷贝的文件名:”,当你输入“test.txt”的时候，会把“test.txt”文件的内容复制一份到文件名叫“test[复制].txt”文件中。



**4.2. 实现步骤分析**

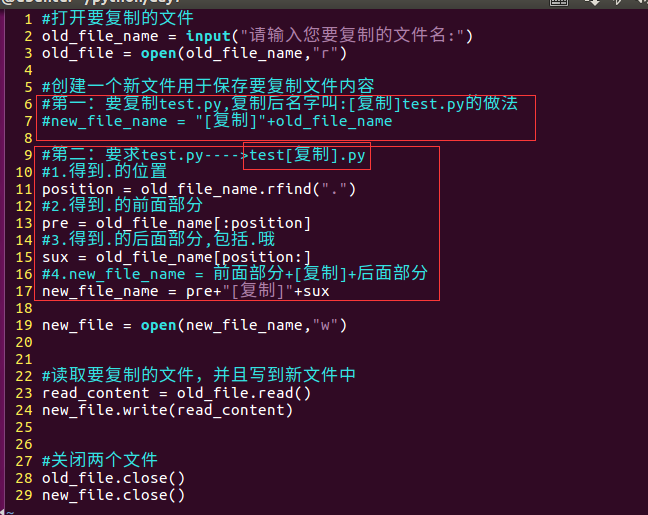
1. **打开要复制的文件**
2. **创建保存文件**
3. **读取要复制文件的数据，并且写到保存文件里面**
4. **关闭两个文件**

## 4.3. 复制简单的实现



## 4.5. 支持xxx[复制].txt格式

前缀（prefix），后缀（suffix）



#coding=utf-8

oldFileName = input("请输入要拷贝的文件名字:")

oldFile = open(oldFileName,'r')

# 如果打开文件if oldFile:

# 提取文件的后缀

fileFlagNum = oldFileName.rfind('.')

if fileFlagNum > 0:

fileFlag = oldFileName[fileFlagNum:]

# 组织新的文件名字

newFileName = oldFileName[:fileFlagNum] + '[复件]' + fileFlag

# 创建新文件

newFile = open(newFileName, 'w')

# 把旧文件中的数据，一行一行的进行复制到新文件中

for lineContent in oldFile.readlines():

newFile.write(lineContent)

# 关闭文件

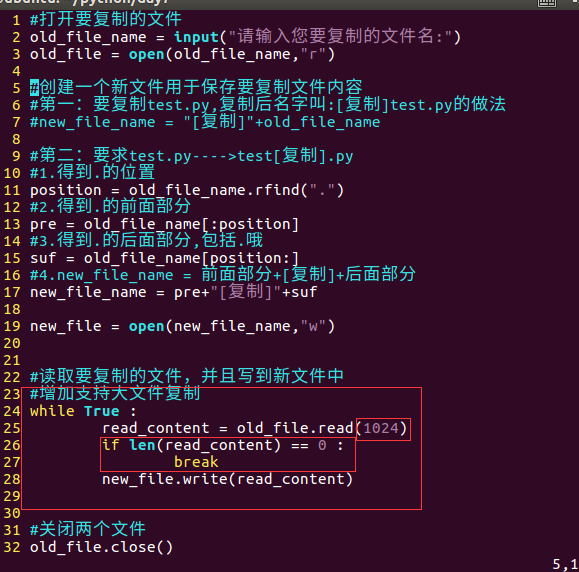
oldFile.close()

newFile.close()

## 4.7. 支持大文件的复制

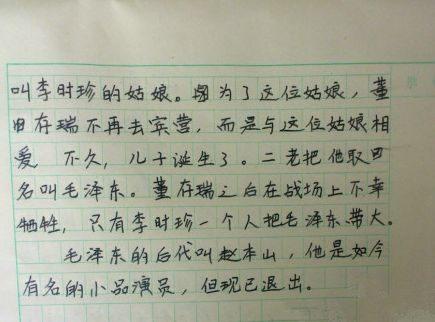
如果一个文件很大，比如5G，试想应该怎样把文件的数据读取到内存然后进行处理呢？





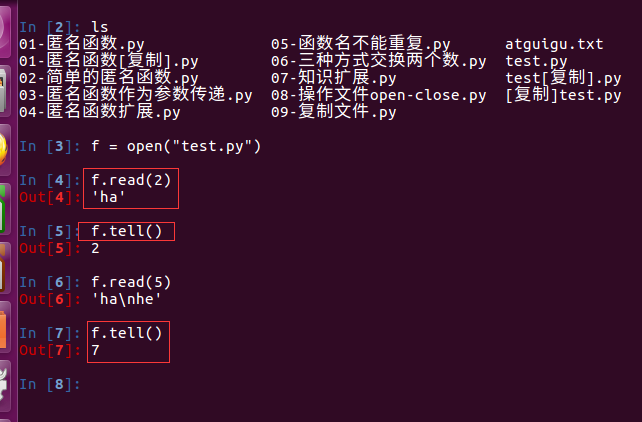
# 5. 文件读写定位

什么是定位？



## **5.1. 获取当前读写的位置tell**

在读写文件的过程中，如果想知道当前的位置，可以使用tell()来获取



# 打开一个已经存在的文件

f = open("test.txt", "r")

str = f.read(3)

print"读取的数据是 : ", str

# 查找当前位置

position = f.tell()

print "当前文件位置 : ", position

str = f.read(3)

print "读取的数据是 : ", str

# 查找当前位置

position = f.tell()

print "当前文件位置 : ", position

f.close()

## 5.2. 定位到某个位置seek

把下面内容拷贝到test.txt文件中

Christmas or Christmas Day is a holiday celebrating the birth of Jesus, the central figure of Christianity. It is traditionally celebrated on December 25 by most Western Christian churches. Although dating to probably as early as [a.d.](https://www.baidu.com/s?wd=a.d.&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1Y3PAf3PH01PHT4m1I-PHm40ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHc3nWbdn10Y" \t "https://zhidao.baidu.com/question/_blank) 200, the feast of Christmas did not become widespread until the Middle Ages. Aspects of celebration may include gift-giving, Christmas trees, display of Nativity sets, church attendance, the Father Christmas/Santa Claus myth, and family gatherings. The word Christmas is derived from Middle English Christemasse. It is a contraction meaning "Christ's mass". The name of the holiday is often shortened to Xmas because Roman letter "X" resembles the Greek letter X, an abbreviation for Christ.

圣诞节是庆祝天主教中心人物[耶稣](https://www.baidu.com/s?wd=%E8%80%B6%E7%A8%A3&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1Y3PAf3PH01PHT4m1I-PHm40ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHc3nWbdn10Y" \t "https://zhidao.baidu.com/question/_blank)诞生的节日，通常西方[天主教堂](https://www.baidu.com/s?wd=%E5%A4%A9%E4%B8%BB%E6%95%99%E5%A0%82&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1Y3PAf3PH01PHT4m1I-PHm40ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHc3nWbdn10Y" \t "https://zhidao.baidu.com/question/_blank)在12月25日庆祝。圣诞节最早可追溯到公元200年，但直到中世纪才开始广泛流传。圣诞节庆祝活动包括赠送礼物、圣诞树、摆[耶稣](https://www.baidu.com/s?wd=%E8%80%B6%E7%A8%A3&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1Y3PAf3PH01PHT4m1I-PHm40ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHc3nWbdn10Y" \t "https://zhidao.baidu.com/question/_blank)系列雕像、参加教堂活动、圣诞老人传说、家庭聚会等。Christmas一词来源于中世纪英语Christemasse，大概的意思是[耶稣](https://www.baidu.com/s?wd=%E8%80%B6%E7%A8%A3&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1Y3PAf3PH01PHT4m1I-PHm40ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHc3nWbdn10Y" \t "https://zhidao.baidu.com/question/_blank)的弥撒。有时候圣诞节被写成Xmas，因为罗马字母X和希腊文Chirst的缩写X很相似。

停止的是，今天的一切不愉快。

如果在读写文件的过程中，需要从另外一个位置进行操作的话，可以使用seek()

seek(offset, from)有2个参数

offset:偏移量

from:方向

0:表示文件开头

1:表示当前位置

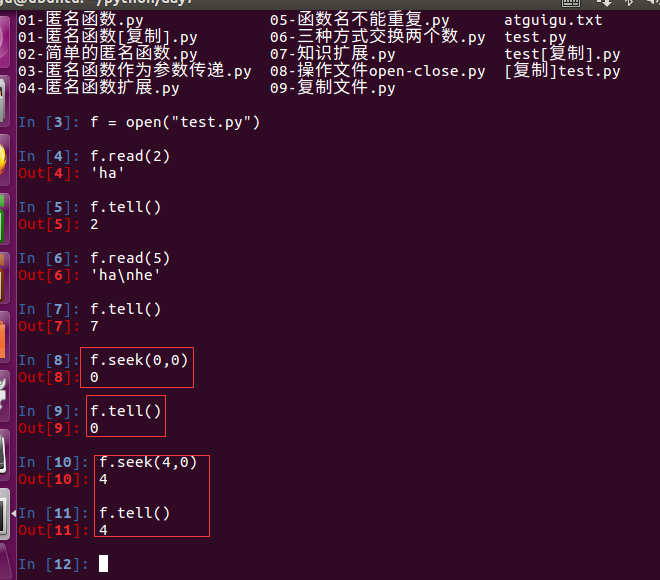
2:表示文件末尾

### 5.2.1. Python2 中seek的使用

定位到文件开头：支持seek(正数,0)；不支持seek(负数,0)

定位到当前位置：支持seek(正数,1)，seek(负数,1)

定位到文件末尾：支持seek(正数,2)，seek(负数,2)





#### 5.2.1.1. 把位置设置为：从文件开头，偏移5个字节

# 打开一个已经存在的文件

f = open("test.txt", "r")

str = f.read(30)

print "读取的数据是 : ", str

# 查找当前位置

position = f.tell()

print "当前文件位置 : ", position

# 重新设置位置

f.seek(5,0)

# 查找当前位置

position = f.tell()

print "当前文件位置 : ", position

f.close()

#### 5.2.1.2. 把位置设置为：离文件末尾，3字节处

# 打开一个已经存在的文件

f = open("test.txt", "r")

# 查找当前位置

position = f.tell()

print "当前文件位置 : ", position

# 重新设置位置

f.seek(-3,2)

# 读取到的数据为：文件最后3个字节数据

str = f.read()

print "读取的数据是 : ", str

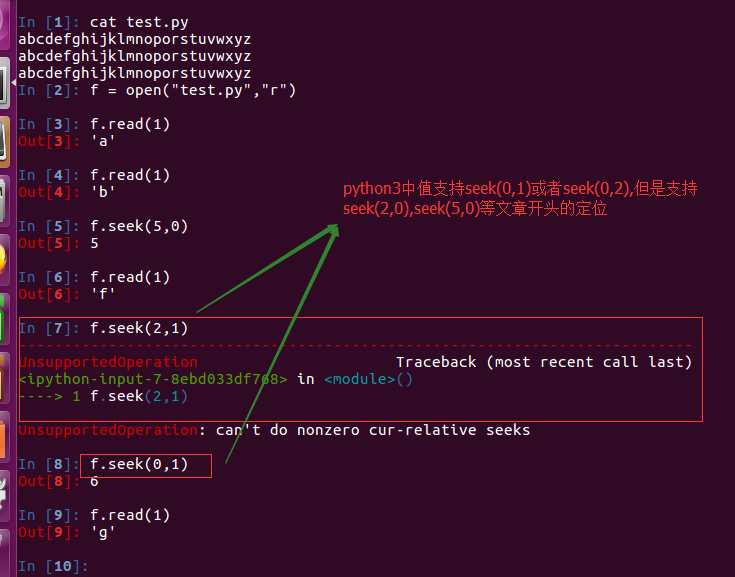
f.close()

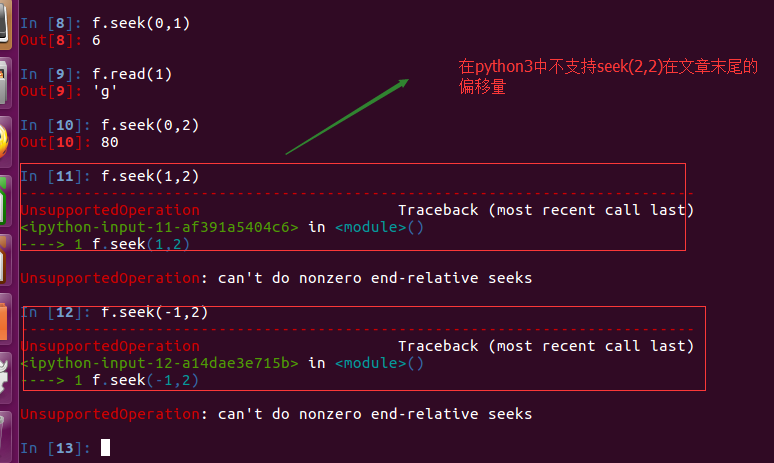
### 5.2.2. Python3中的seek

定位到文件开头：seek(0,0)，seek(3,0);不支持seek(负数,0);

定义文件当前位置：支持seek(0,1)；不支持seek(正,1)和seek(负数,1)

定位到文件末尾：支持seek(0,2)；不支持seek(正,2)和seek(负数,2)





## 5.3. 小结

### 5.3.1. 在python2中

seek(offset, from)有2个参数

offset:偏移量

from:方向

0:表示文件开头

1:表示当前位置

2:表示文件末尾

f.seek(0,0),f.seek(2,0)

f.seek(2,1) f.seek(5,1)

f.seek(-2,1)

f.seek(-2,2),f.seek(-5,2)

### 5.3.2. 在python3中

f.seek(0,0),f.seek(2,0)

f.seek(0,1),f.seek(0,2)

下面的都不可用了

f.seek(2,1) f.seek(5,1)

f.seek(-2,2),f.seek(-5,2)

# 6. 使用os模块对文件的重命名、删除

有些时候，需要对文件进行重命名、删除等一些操作，Python的os模块中都有这么功能

## 6.1. 文件重命名rename

os模块中的rename()可以完成对文件的重命名操作

rename(需要修改的文件名, 新的文件名)

import os

os.rename("程序猿自身修养.txt", "程序员自身修养.txt")

## 6.2. 删除文件remove

os模块中的remove()可以完成对文件的删除操作

remove(待删除的文件名)

import os

os.remove("程序员自身修养.txt")

## 6.3. 练一练

1.写一个程序，运行起来的时候，创建一个名为”程序员自我休养.txt”，提示用户输入相关内容，并且查看是否写入内容成功。

2.成功后重命名为：”程序员自我修养.txt”

3.然后再写一个程序删除它

# **7. 使用os模块对文件夹的相关操作**

实际开发中，有时需要用程序的方式对文件夹进行一定的操作，比如创建、删除等

就像对文件操作需要os模块一样，如果要操作文件夹，同样需要os模块

## **7.1. 创建文件夹**

不能创建./张三/test/hehe

import os

os.mkdir("张三")

## **7.2. 创建多层目录**

import os

os.makedirs("张三/李四/王二")

## **7.3. 获取当前目录**

import os

os.**getcwd**()

## **7.4. 改变默认目录**

进入到某个目录，或者是切换目录

**import os**

**os.chdir("../")**

## **7.5. 获取目录列表，包含文件和目录**

import os

os.listdir("./")

## **7.6. 删除文件夹**

删除目录 path，要求path必须是个空目录，否则抛出OSError错误

import os

os.rmdir("张三")

**删除多层目录**

import shutil  
shutil.rmtree(“张三”)

## 7.7. 练习

用程序创建一个a目录，进入a目录，再创建b目录；进入b目录后，创建一文件”日记.txt”，写入内容:”今天我在硅谷学习”，并且读取该文件，打印到终端上。

扩展：退回到a目录下，把当前目录下所有内容列表，并且把b目录删除掉，然后再列出当前目录所有内容。

import os

import shutil

print("当前目录:%s"%os.getcwd())

#创建a目录

os.mkdir("a")

#进入到a目录

os.chdir("a")

print("当前目录:%s"%os.getcwd())

#创建b目录

os.mkdir("b")

#进入到b目录创建日记.txt文件

os.chdir("b")

print("当前目录:%s"%os.getcwd())

f = open("日记.txt","w")

#写入内容:”今天我在硅谷学习”

f.write("今天我在尚硅谷学习!")

f.close()

f= open("日记.txt")

print("从文件读取的内容是:%s" % f.read())

f.close()

#退回到a目录

print("当前目录:%s"%os.getcwd())

#把b目录删除

os.chdir("./..")

print("当前目录:%s"%os.getcwd())

print(os.listdir())

shutil.rmtree("b")

print(os.listdir())

# **8. 应用：批量修改文件名**

## **8.1. 运行过程演示**

写个程序创建目录test，进入test目录，

创建文件如下：

人民的名义-1.avi

人民的名义-2.avi

人民的名义-3.avi

人民的名义-4.avi

人民的名义-5.avi

人民的名义-6.avi

import os

#创建目录test

print(os.getcwd())

os.mkdir("./test")

#进入test目录

os.chdir("./test")

print(os.getcwd())

#创建文件

#人民的名义-1.avi

#人民的名义-2.avi

#人民的名义-3.avi

#人民的名义-4.avi

#人民的名义-5.avi

#人民的名义-6.avi

i = 1

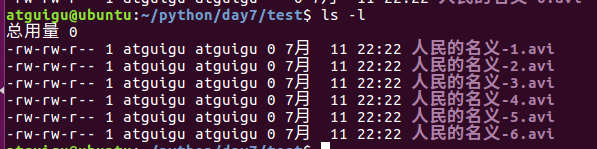
while i <= 6 :

open("人民的名义-%d.avi" %i,"w")

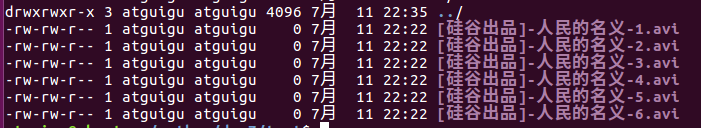
i += 1

print("创建完毕!")

运行程序之前

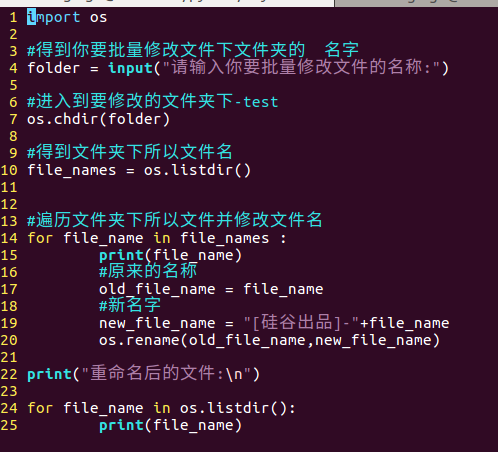


运行程序之后



## **8.2. 参考代码**

参照代码1



参照代码2

#coding=utf-8

# 批量在文件名前加前缀

import os

funFlag = 1 # 1表示添加标志 2表示删除标志

folderName = './renameDir/'

# 获取指定路径的所有文件名字

dirList = os.listdir(folderName)

# 遍历输出所有文件名字

for name in dirList:

print name

if funFlag == 1:

newName = '[硅谷出品]-' + name

elif funFlag == 2:

num = len('[硅谷出品]-')

newName = name[num:]

print newName

os.rename(folderName+name, folderName+newName)

# Python换行符问题

**先知道结果：**

在linux和mac系统上我们读写文本文件使用二进制方式或者文本方式都可以，因为在处理\n都是一样的；那么读写文本文件和二进制文件的时候，可以使用r和w模式或者rb和wb模式。

在window中的换行是\r\n，当我们使用程序以文本方式写入一个\n的时候，默认会帮我们加上\r，这样打开文件的时候才能得到换行的效果；

所有在window上读写文本文件的时候，**建议使用r或者w模式。**

**在读写二进制文件的时候，建议用rb和wb模式。**

## 9.1. 不同操作系统换行符是不一样的

在处理文本数据的时候体现在不同操作系统处理\r\n是不一样的

不同操作系统**换行符**是不一样的

**linux -->\n**

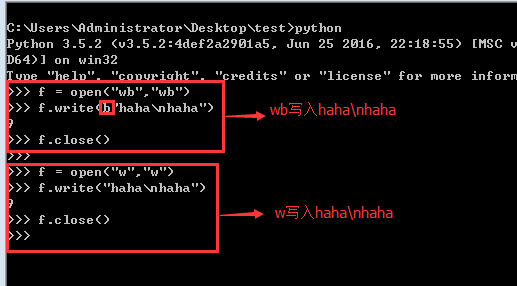
**unix-->\n**

**mac-->老版本(\r)-->\n**

**windows -->\r\n**

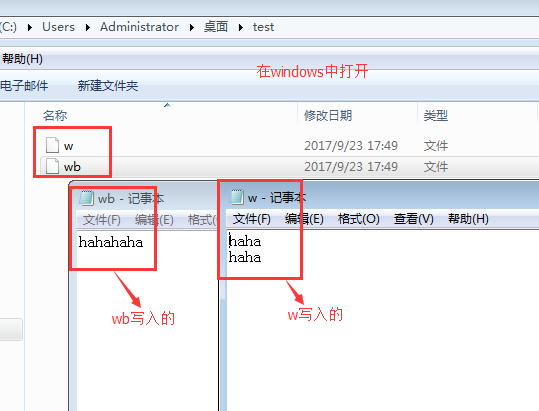
对应换行符windows遇到\r\n才换行显示

## 9.2. 在windos上分别使用wb和w写入内容：haha\nhah

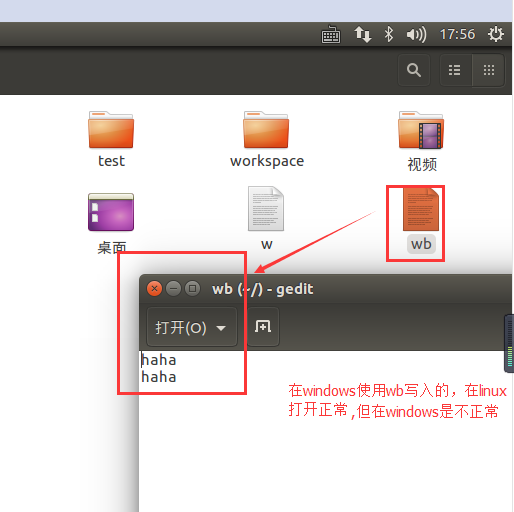


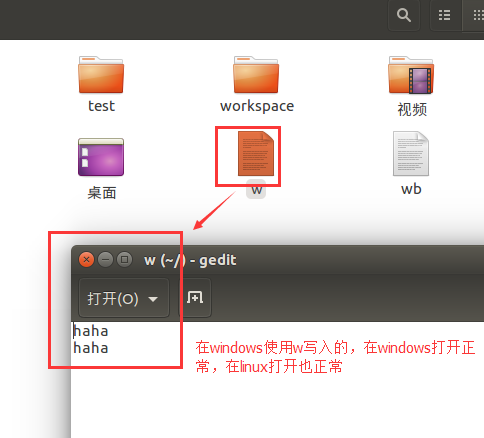
### 9.2.1. 在windows系统中使用记事本打开

使用w写入的正常显示换行；使用wb写入的无法显示换行



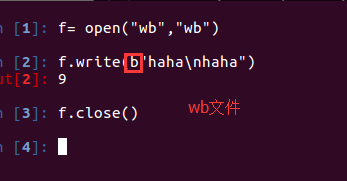
### 9.2.2. 把文件拷贝到linux打开换行都能正常显示



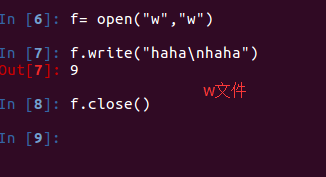


## 9.3. 在linux系统上使用w和wb入的内容：haha\nhaha

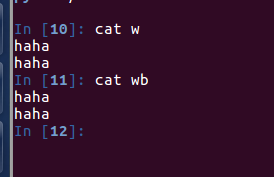
内容包含“haha\nhaha”,如果拷贝到windows上打开，没有显示换行。



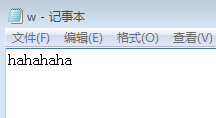
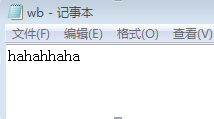
和



### 9.3.1. 在linux系统上都能正常显示

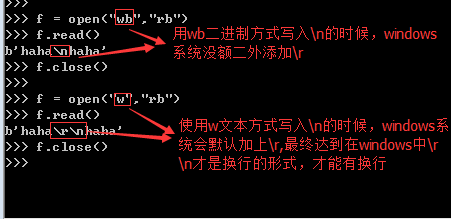


### 9.3.2. 在windows系统中使用记事本打开都不能换行显示



原因是在linux下系统不会默认加上\r

## 9.4. 在windows上都用rb(二进制)方式读取内容会发写入的内容有区别的



当我们去使用记事本打开的时候都，不能显示换行符了。

说明，在windows系统下，当使用“w”,写入‘\n’的时候系统会默认帮我加上‘\r’，变成是‘\r\n’;而使用‘wb’写入的时候不会添加‘\r’。

## 9.4. 小结

在linux和mac系统上我们读写文本文件使用二进制方式或者文本方式都可以，因为在处理\n都是一样的；那么读写文本文件和二进制文件的时候，可以使用r和w模式或者rb和wb模式。

在window中的换行是\r\n，当我们使用文本方式写入一个\n的时候，默认会帮我们加上\r，这样打开文件的时候才能得到换行的效果；

所有在window上读写文本文件的时候，**建议使用r或者w模式。**

**在读写二进制文件的时候，建议用rb和wb模式**

# **10. 作业和总结**

## 1. 综合应用:学生管理系统(文件版)

添加的姓名，能保存到本地，下次重启运行的时候，能直接看到上次保存的。

## 2. 读取一个文件，显示除了以井号(#)开头的行以外的所有行

## 3. 制作一个"密码薄",其可以存储一个网址（例如 www.atguigu.com），和一个密码(例如 123456)，请编写程序完成这个“密码薄”的增删改查功能，并且实现文件存储功能