10_Python文件读写和目录操作

- 1 I/O操作
  - 1.1 打开文件
    - 1.1.1 方法一：先打开后关闭
      - 1.1.1.1 file=open(filename[,mode[buffering]])
      - 1.1.1.2 指定编码方式打开文件
      - 1.1.1.3 记得关闭文件
    - 1.1.2 方法二：with语句
  - 1.2 写入文件
    - 1.2.1 file.write()
    - 1.2.2 file.writelines()
  - 1.3 读取文件
    - 1.3.1 file.read(size)
    - 1.3.2 file.seek(offset,[,whence])
    - 1.3.3 file.readline()
    - 1.3.4 file.readlines()
- 2 文件的重命名、删除（了解）
  - 2.1 文件重命名
  - 2.2 删除文件
  - 2.3 文件夹的相关操作
    - 2.3.1 获取当前目录
    - 2.3.2 创建文件夹
    - 2.3.3 获取目录列表
    - 2.3.4 删除文件夹
  - 2.4 os方法大全

In [7]:
```
#设置全部行输出
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"
```

1 I/O操作

1.1 打开文件

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)

In [ ]:
```
import os

os.getcwd() #返回当前工作目录
```

In [ ]:
```
%pwd
```

In [129]:
```
os.chdir("C:\\Users\\Administrator\\Desktop\\IO")  #临时改变当前工作目录到指定的路径（本次有
```

1.1.1 方法一：先打开后关闭

1.1.1.1 file=open(filename[,mode[buffering]])

file:被创建的文件对象

filename:要创建或打开文件的文件名称，需要使用单引号或者双引号括起来。如果要打开的文件和当前文件在同一目录下，直接写文件名即可。

mode:访问模式，访问方式字符功能见下表,默认为r

buffering:0表示不缓存，1表示缓存（默认），大于1表示缓冲区大小

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。
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。

1.1.1.2 指定编码方式打开文件

In [12]:
```
1  file=open("poetry.txt","r",encoding='utf-8')
2  hehe=file.read()
3  print(hehe)
```

```
轻轻的我走了，
正如我轻轻的来；
我轻轻的招手，
作别西天的云彩。

那河畔的金柳，
是夕阳中的新娘；
波光里的艳影，
在我的心头荡漾。
```

如果**poetry.txt**不在默认的当前工作目录，怎么办？比如在**E**盘根目录下：

In [5]:
```
1  file=open(r"E:\poetry.txt","r",encoding='utf-8')  #文件名添加上其对应的路径即可
2  hehe=file.read()
3  print(hehe)
```

```
软泥上的青荇，
油油的在水底招摇；
在康河的柔波里，
我甘心做一条水草！
```

1.1.1.3 记得关闭文件

In [13]:
```
1  file.close()
```

如果不关闭文件:

1. 会导致内存始终被占用，得不到释放

2. 会导致文件被占用中, 无法删除, 剪切等操作m

In [14]:
```
file=open("poetry.txt","r",encoding='utf-8')
hehe=file.read()

file.close()

print(hehe)
```

轻轻的我走了，
正如我轻轻的来；
我轻轻的招手，
作别西天的云彩。

那河畔的金柳，
是夕阳中的新娘；
波光里的艳影，
在我的心头荡漾。

1.1.2 方法二：with语句

如果打开的文件抛出异常，文件将会无法关闭，为了避免由于忘记关闭文件导致的各种问题，可以使用with语句。

with语句执行完毕后，关闭文件，无论是否抛出异常。

In [15]:
```
with open("poetry.txt","r",encoding='utf-8') as file:
    hehe=file.read()
    hehe
    print(hehe)

# file=open("poetry.txt","r",encoding='utf-8')
# hehe=file.read()
# file.close()
# print(hehe)
```

Out[15]: '轻轻的我走了，\n正如我轻轻的来；\n我轻轻的招手，\n作别西天的云彩。\n\n那河畔的金柳，\n是夕阳中的新娘；\n波光里的艳影，\n在我的心头荡漾。\n'

轻轻的我走了，
正如我轻轻的来；
我轻轻的招手，
作别西天的云彩。

那河畔的金柳，
是夕阳中的新娘；
波光里的艳影，
在我的心头荡漾。

1.2 写入文件

写入文件的步骤：

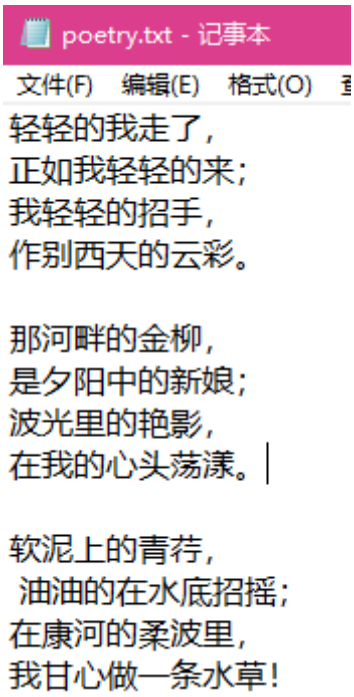
- 1、打开：使用open()方法的时候，打开模式选择为w(重头覆盖写入)或者是a（追加写入）
- 2、写入：

- 如果往文件写入的是字符串，则在with语句缩进内使用.write()
- 如果往文件写入的是字符串列表，则使用.writelines()

1.2.1 file.write()

In [16]:
```
with  open("poetry.txt","a",encoding='utf-8') as file:
    file.write("\n\n软泥上的青荇，\n油油的在水底招摇；\n在康河的柔波里，\n我甘心做一条水草！")
```

Out[16]: 37



In [17]:
```
with open("poetry.txt","r",encoding='utf-8') as file:
    print(file.read())
```

轻轻的我走了，
正如我轻轻的来；
我轻轻的招手，
作别西天的云彩。

那河畔的金柳，
是夕阳中的新娘；
波光里的艳影，
在我的心头荡漾。

软泥上的青荇，
油油的在水底招摇；
在康河的柔波里，
我甘心做一条水草！

用两次**with**语句才能够写入后再查看两个步骤，太麻烦了，有写入和查看的只用一次**with**的方法吗？打开模式改为**"a+"**即可。

In [22]:

```
b="\n\n那榆荫下的一潭，\n不是清泉，是天上虹；\n揉碎在浮藻间，\n沉淀着彩虹似的梦。"

with open("poetry.txt","a+",encoding="utf-8") as file:
    file.write(b)
    file.seek(0)  #必须得写
    print(file.read())
```

Out[22]: 38

Out[22]: 0

轻轻的我走了，
正如我轻轻的来；
我轻轻的招手，
作别西天的云彩。

那河畔的金柳，
是夕阳中的新娘；
波光里的艳影，
在我的心头荡漾。

软泥上的青荇，
油油的在水底招摇；
在康河的柔波里，
我甘心做一条水草！

那榆荫下的一潭，
不是清泉，是天上虹；
揉碎在浮藻间，
沉淀着彩虹似的梦。

1.2.2 file.writelines()

In [26]:
```
b=["\n\n寻梦？撑一支长篙，","\n向青草更青处漫溯；","\n满载一船星辉，","\n在星辉斑斓里放歌。"]

with open("poetry.txt","a+",encoding="utf-8") as file:
    file.writelines(b)
    file.seek(0)  #必须得写
    print(file.read())

```

Out[26]: 0

轻轻的我走了，
正如我轻轻的来；
我轻轻的招手，
作别西天的云彩。

那河畔的金柳，
是夕阳中的新娘；
波光里的艳影，
在我的心头荡漾。

软泥上的青荇，
油油的在水底招摇；
在康河的柔波里，
我甘心做一条水草！

那榆荫下的一潭，
不是清泉，是天上虹；
揉碎在浮藻间，
沉淀着彩虹似的梦。
寻梦？撑一支长篙，
向青草更青处漫溯；

寻梦？撑一支长篙，
向青草更青处漫溯；
满载一船星辉，
在星辉斑斓里放歌。

1.3 读取文件

我们之前就接触过file.read()这语法格式：

1.3.1 file.read(size)

file:文件对象
size:读取的字符串个数

In [27]:
```
with open("poetry.txt","r",encoding='utf-8') as file:
    hehe=file.read(6)
    print(hehe)
```

轻轻的我走了

1.3.2 file.seek(offset,[,whence])

file:文件对象
offset:用于指定移动的字符个数
whence：0表示从文件头开始，1表示从当前位置开始，2表示从文件尾开始。默认为0。

这里需要注意：
在utf-8编码情况下，offset的数值3相当于取一个汉字位置（包括汉字标点符号）
如果是GBK编码，offset数值2相当于取一个汉字字符位置
但是，read()里面参数，一个数值相当于一个汉字字符。

In [28]:
```
with open("poetry.txt","r",encoding='utf-8') as file:
    file.seek(23)       #文件开头偏移7个汉字，7*3=21，加上“\n”就是23
    hehe=file.read(7)
    print(hehe)
```

Out[28]: 23

正如我轻轻的来

1.3.3 file.readline()

In [29]:
```
with open("poetry.txt","r",encoding='utf-8') as file:
    hehe=file.readline().strip()
    print(hehe)
    #打开同一个文件后，读取第一行字符，继续读取的话，会从下一行开始
    hehe=file.readline().strip()
    print(hehe)
```

轻轻的我走了，
正如我轻轻的来；

In [30]:
```
with open("poetry.txt","r",encoding='utf-8') as file:
    for i in range(4):
        hehe=file.readline().strip()
        print(hehe)
```

轻轻的我走了，
正如我轻轻的来；
我轻轻的招手，
作别西天的云彩。

1.3.4 file.readlines()

需要注意的是，如果使用file.readlines()读取全部行，返回的一个字符串列表，每个元素作为文件的一行内容。

因此，可以用for循环逐行读取字符串列表。

In [40]:
```
with open("poetry.txt","r",encoding='utf-8') as file:
    hehe=file.readlines()
    for i in hehe:
        i=i.strip()
        print(i)
```

```
轻轻的我走了，
正如我轻轻的来；
我轻轻的招手，
作别西天的云彩。

那河畔的金柳，
是夕阳中的新娘；
波光里的艳影，
在我的心头荡漾。

软泥上的青荇，
油油的在水底招摇；
在康河的柔波里，
我甘心做一条水草！

那榆荫下的一潭，
不是清泉，是天上虹；
揉碎在浮藻间，
沉淀着彩虹似的梦。
```

In [41]:
```
hehe
```

Out[41]:
```
['轻轻的我走了，\n',
 '正如我轻轻的来；\n',
 '我轻轻的招手，\n',
 '作别西天的云彩。\n',
 '\n',
 '那河畔的金柳，\n',
 '是夕阳中的新娘；\n',
 '波光里的艳影，\n',
 '在我的心头荡漾。\n',
 '\n',
 '软泥上的青荇，\n',
 '油油的在水底招摇；\n',
 '在康河的柔波里，\n',
 '我甘心做一条水草！\n',
 '\n',
 '那榆荫下的一潭，\n',
 '不是清泉，是天上虹；\n',
 '揉碎在浮藻间，\n',
 '沉淀着彩虹似的梦。']
```

2 文件的重命名、删除（了解）

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

2.1 文件重命名

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)

In [126]:
```
1 import os
```

In [51]:
```
1 print(dir(os))
```

```
['F_OK', 'MutableMapping', 'O_APPEND', 'O_BINARY', 'O_CREAT', 'O_EXCL', 'O_NOINHERIT', 'O_RAN
DOM', 'O_RDONLY', 'O_RDWR', 'O_SEQUENTIAL', 'O_SHORT_LIVED', 'O_TEMPORARY', 'O_TEXT', 'O_TRUN
C', 'O_WRONLY', 'P_DETACH', 'P_NOWAIT', 'P_NOWAITO', 'P_OVERLAY', 'P_WAIT', 'R_OK', 'SEEK_CU
R', 'SEEK_END', 'SEEK_SET', 'TMP_MAX', 'W_OK', 'X_OK', '_DummyDirEntry', '_Environ', '__all_
_', '__builtins__', '__cached__', '__doc__', '__file__', '__loader__', '__name__', '__package
__', '__spec__', '_dummy_scandir', '_execvpe', '_exists', '_exit', '_get_exports_list', '_put
env', '_unsetenv', '_wrap_close', 'abort', 'access', 'altsep', 'chdir', 'chmod', 'close', 'cl
oserange', 'cpu_count', 'curdir', 'defpath', 'device_encoding', 'devnull', 'dup', 'dup2', 'en
viron', 'errno', 'error', 'execl', 'execle', 'execlp', 'execlpe', 'execv', 'execve', 'execv
p', 'execvpe', 'extsep', 'fdopen', 'fsdecode', 'fsencode', 'fstat', 'fsync', 'ftruncate', 'ge
t_exec_path', 'get_handle_inheritable', 'get_inheritable', 'get_terminal_size', 'getcwd', 'ge
tcwdb', 'getenv', 'getlogin', 'getpid', 'getppid', 'isatty', 'kill', 'linesep', 'link', 'list
dir', 'lseek', 'lstat', 'makedirs', 'mkdir', 'name', 'open', 'pardir', 'path', 'pathsep', 'pi
pe', 'popen', 'putenv', 'read', 'readlink', 'remove', 'removedirs', 'rename', 'renames', 'rep
lace', 'rmdir', 'scandir', 'sep', 'set_handle_inheritable', 'set_inheritable', 'spawnl', 'spa
wnle', 'spawnv', 'spawnve', 'st', 'startfile', 'stat', 'stat_float_times', 'stat_result', 'st
atvfs_result', 'strerror', 'supports_bytes_environ', 'supports_dir_fd', 'supports_effective_i
ds', 'supports_fd', 'supports_follow_symlinks', 'symlink', 'sys', 'system', 'terminal_size',
 'times', 'times_result', 'truncate', 'umask', 'uname_result', 'unlink', 'urandom', 'utime',
 'waitpid', 'walk', 'write']
```

如果想将当前工作目录的"poetry.txt"文件名改为"poetry2.txt":

In [130]:
```
1 os.rename("poetry.txt", 'poetry2.txt')
```

2.2 删除文件

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)

删除当前工作目录的poetry2.txt:

In [131]:
```
1 os.remove('poetry2.txt')
```

2.3 文件夹的相关操作

2.3.1 获取当前目录

In [140]:
```
1 import os
2 os.getcwd()
```

Out[140]: 'C:\\Users\\Administrator\\Desktop\\IO'

2.3.2 创建文件夹

In [132]:
```
os.getcwd()   #查看当前工作路径（get current working directory）
```

Out[132]: 'C:\\Users\\Administrator\\Desktop\\IO'

In [136]:
```
os.mkdir('我的Python编程基础笔记')  #make directory
os.mkdir('我的Python数据清洗笔记')  #make directory
```

2.3.3 获取目录列表

In [141]:
```
import os
os.listdir()
```

Out[141]: ['我的Python数据清洗笔记', '我的Python编程基础笔记']

In [142]:
```
for i in os.listdir():
    print(i)
```

```
我的Python数据清洗笔记
我的Python编程基础笔记
```

2.3.4 删除文件夹

In [143]:
```
os.rmdir("我的Python编程基础笔记")
```

2.4 os方法大全

import os

os方法	作用
os.getcwd()	获得当前工作目录
os.chdir("dirname")	改变当前脚本的工作路径，相当于shell下的cd
os.curdir	返回当前目录'.'
os.pardir	获取当前目录的父目录字符串名'..'
os.makedirs('dirname1/dirname2')	可生成多层递归目录
os.removedirs('dirname1/dirname2')	若目录为空，则删除，并递归到上一级目录，如若也为空，则删除，依此类推
os.mkdir('test4')	生成单级目录；相当于shell中mkdir dirname
os.rmdir('test4')	删除单级空目录，若目录不为空则无法删除，报错；相当于shell中rmdir dirname
os.listdir('/pythonStudy/s12/test')	列出指定目录下的所有文件和子目录，包括隐藏文件，并以列表方式打印
os.remove('log.log')	删除一个指定的文件
os.rename("oldname","newname")	重命名文件/目录)
os.stat('/pythonStudy/s12/test')	获取文件/目录信息
os.pathsep	输出用于分割文件路径的字符串';'
os.name	输出字符串指示当前使用平台。win->'nt'; Linux->'posix'

os方法	作用
os.system(command='bash')	运行shell命令，直接显示
os.environ	获得系统的环境变量
os.path.abspath('/pythonStudy/s12/test')	返回path规范化的绝对路径
os.path.split('/pythonStudy/s12/test')	将path分割成目录和文件名二元组返回
os.path.dirname('/pythonStudy/s12/test')	| 返回path的目录。其实就是os.path.split(path)的第一个元素
os.path.basename('/pythonStudy/s12/test')	返回path最后的文件名。如果path以 / 或\结尾，那么就会返回空值。即os.path.split(path)的第二个元素
os.path.exists('test')	判断path是否存在
os.path.isabs('/pythonStudy/s12/test')	如果path是绝对路径，返回True
os.path.isfile('test')	如果path是一个存在的文件，返回True。否则返回False
os.path.isdir('/pythonStudy/s12/test')	如果path是一个存在的目录，则返回True。否则返回False
os.path.getatime('/pythonStudy/s12/test')	返回path所指向的文件或者目录的最后存取时间
os.path.getmtime('/pythonStudy/s12/test')	返回path所指向的文件或者目录的最后修改时间