文件操作

一、 基本文件操作

1. 创建和打开文件

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

buffering:用于指定读写文件的缓冲模式，0 表示不缓存、1 为缓存、大于 1 则表示缓冲区的大小

mode:指定打开文件的方式，默认的打开方式为 r（只读）

```
#mode参数说明

#文件必须存在
r       #以只读模式打开文件，文件指针位于开头
rb      #以只读方式打开二进制文件(图片、声音)
r+      #打开文件后可读可写，从文件开头覆盖
rb+     #可读可写，从开头覆盖，二进制文件

#文件存在则将其覆盖，否则创建新文件
w       #只写模式
wb      #只写模式打开二进制文件
w+      #打开文件先清楚原来内容，对这个空文件读写
wb+     #二进制打开文件，采用读写模式

#无特殊注意
a       #以追加模式打开文件，如果文件存在则指针放在末尾
        #否则创建新文件用于写入
ab      #以二进制方式打开文件，其余同上
a+      #以读写模式打开文件，存在时指针位于末尾，否则创建文件
ab+     #以二进制方式打开文件，其余同上
```

2. 关闭文件：`file.close()`

3. 打开文件时使用 with 语句

打开文件后要及时将其关闭，如果忘记关闭可能会带来意想不到的问题。另外，如果在打开文件时抛出了异常，那么将导致文件不能及时被关闭。为了更好地避免这类问题发生，可以使用 Python 提供的 with 语句，从而实现在处理文件时，无论是否抛出异常，都能保证 with 语句执行完毕后关闭已经打开的文件。

```
with expression as target:
    with-body
```

4. 写入文件内容

`file.write(string)` 注：1 用这个方法写入时应确保打开模式为 w 或 a 2 写入文件后一定要调用 close()方法关闭文件，否则写入的内容不会保存在文件中。

5. 读取文件

（1）读取指定字符：`file.read([size])` `file.seek(offset[,whence])` 用于将指针指向指定的位置，offset 是偏移量，whence 指定从什么位置开始，0 为开头，1 为当前位置，2 为文件尾

（2）读取一行：`file.readline()` 返回 str

（3）读取多行：`file.readlines()` 返回 str 组成的列表

二、 目录操作

目录也叫文件夹，用于分层保存文件。通过目录可以分门别类的保存和找到想要的文件。在 Python 中，是通过 os 和 os.path 模块来实现的。

1、 os 和 os.path 模块

os 提供的一些操作目录的函数：

```
getcwd()              #返回当前的工作目录
listdir(path)         #返回指定路径下的文件和目录信息
mkdir(path[,mode])    #创建目录
makedirs(path1/path2)#创建多级目录
rmdir(path)           #删除目录
removedirs(path1/path2)#删除多级目录
chdir(path)           #设置为工作目录
walk(top[topdown[onerror]])#遍历目录树，返回元组，包括路径名、目录列表和文件列表
```

os.path 提供的一些操作目录的函数：

```
abspath(path)   #获取文件目录的绝对路径
exists(path)    #判断目录或文件是否存在
join(path,name)#将目录和目录或者文件名连接起来
splitext()      #分离文件名和扩展名
basename(path) #从一个目录中提取文件名
dirname(path)   #从一个文件中提取文件路径，不包括文件名
isdir(path)     #用于判断是否是有效路径
```

2、 路径

(1) 相对路径

学习相对路径前，先了解什么是当前工作目录。当前工作目录就是指当前文件所在的目录。

```
>>> import os
>>> print(os.getcwd())
C:\Users\liuyi\AppData\Local\Programs\Python\Python310
```

相对路径就是依赖于当前工作目录的，如果在当前工作目录下有一个 message.txt 的文件，那么打开这个文件时就可以直接写上文件名，这时采用的就是相对路径，这个文件的实际路径就是工作目录’C:\Users\liuyi\AppData\Local\Programs\Python\Python310’+相对路径’message.txt’。

(2) 绝对路径

(3) 拼接路径

如果想要将两个或多个路径拼接一起组成新的路径，可以使用 os.path 中 join()方法。

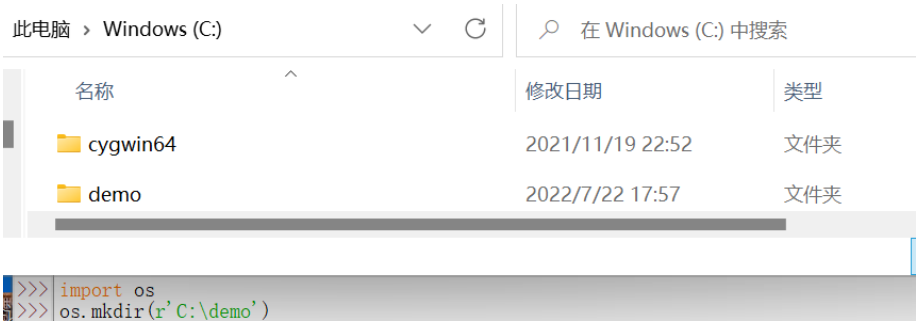
说明：如果要拼接的路径中存在多个绝对路径，那么以从左到右最后一次出现的路径为准，并且该路径之前的参数都将忽略。因此把不同路径拼接时要使用这个函数，这样可以正确处理不同操作系统的路径分隔符

3、 判断目录是否存在：os.path.exists(path)

4、 创建目录

(1) 创建一级目录

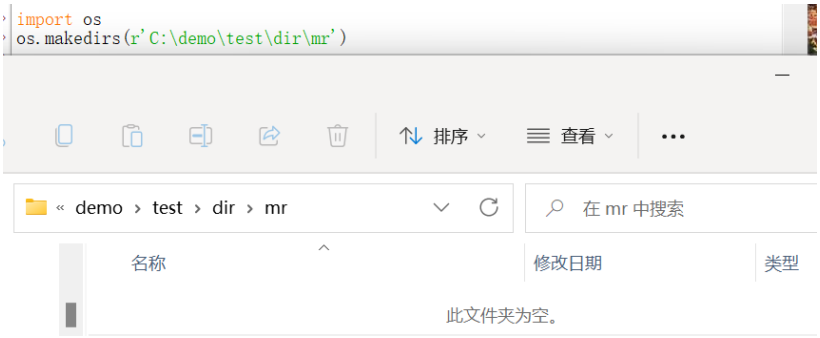
os.mkdir(path)



(2) 创建多级目录

os.makedirs(path)

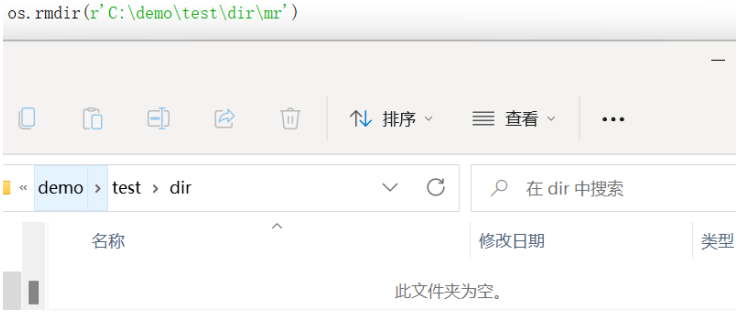
```
import os
os.makedirs(r'C:\demo\test\dir\mr')
```



5、 删除目录

os.rmdir(path)

```
os.rmdir(r'C:\demo\test\dir\mr')
```




```
>>> os.rmdir(r'C:\demo\test')
Traceback (most recent call last):
  File "<pyshell#13>", line 1, in <module>
    os.rmdir(r'C:\demo\test')
OSError: [WinError 145] 目录不是空的。: 'C:\\demo\\test'
```

当目录是非空时，此函数将会报错，这时如果想删除非空目录，应该使用 os.rmtree(path)函数

6、 遍历目录

os.walk(top)

```
>>> import os
>>> tuples=os.walk(r'C:\Users\liuyi\Desktop\软件分析与验证\handout')
>>> for i in tuples:
...     print(i,'\n')
...
...

('C:\\Users\\liuyi\\Desktop\\软件分析与验证\\handout', [], ['10-procedure.handout.pdf', '11-termination.handout.pdf', '2-propositional-logic.handout.pdf', '3-first-order-logic.handout.pdf', '4-theories.handout.pdf', '5-program-semantics.handout.pdf', '6-hoare.handout.pdf', '7-loop.handout.pdf', '8-array.handout.pdf', '9-predicate-transformation.handout.pdf'])
```

三、 高级文件操作

1. 删除文件：os.remove(path)

2. 重命名文件和目录：os.rename(src,dst) src 指定重命名的目录文件，dst 指定重命名后的文件