1. **Python对文件夹的操作**

**第一部分**

**一、判断某个文件夹是否存在**

方法:os.path.exists(path) 判断一个目录是否存在,如果存在，则返回True，如果不存在，返回False.

import os

print(os.path.exists(r"c:\a"))

**二、创建一个文件夹**

方法1：os.mkdir(path) 创建目录

方法2：os.makedirs(path) 多层创建目录

这两个函数之间最大的区别是当父目录不存在的时候os.mkdir(path)不会创建，os.makedirs(path)则会创建父目录。

需要注意的是，上面两种方法，如果目录已经存在，则不能再创建，否则报错。

#coding=utf-8

import os

if not os.path.exists(r"c:\d\f\e"):

os.makedirs(r"c:\d\f\e")

else:

print("该目录已经存在")

**三、遍历文件夹**

使用 os.walk遍历文件夹，这个方法返回的是一个三元tupple(dirpath, dirnames, filenames),

其中第一个为起始路径，第二个为起始路径下的文件夹,第三个是起始路径下的文件。  
dirpath是一个string，代表目录的路径,

dirnames是一个list，包含了dirpath下所有子目录的名字,

filenames是一个list，包含了非目录文件的名字.这些名字不包含路径信息,如果需要得到全路径,需要使用 os.path.join(dirpath, name).

import os

for i in os.walk(r"Q:/d"):

print(i)

执行结果:

('Q:/d', [], ['d.py', 'data.pkl', 'e.py'])

1. **删除文件和文件夹**

import os

1、删除文件：   
os.remove()

2、删除空目录：   
os.rmdir()

3、递归删除空目录：   
os.removedirs()

例子：

import os

print os.remove("file2.txt")

import os

#只删除e空文件夹

os.rmdir(r"c:\d\f\e")

#如果d和f都是空文件夹,都删除

os.removedirs(r"c:\d\f")

4、[python删除一个非空文件](http://www.cnblogs.com/xiaowuyi/archive/2012/05/04/2482113.html)

#coding=utf-8

import shutil

shutil.rmtree(r"Q:/d")

5、自己编写代码，删除一个非空文件夹

import os

for root, dirs, files in os.walk(r'Q:/d', topdown=False):

for name in files:

os.remove(os.path.join(root, name))

for name in dirs:

os.rmdir(os.path.join(root, name))

**第二部分**

1. **综合案例**
2. 统计某个文件夹下所有.txt的文件的格式,包括子目录

方法1：

import os

def getFilename\_extension(filename):

return filename[filename.rfind('.')+1:]

i=0

for roots ,folder ,fileName in os.walk(r"Q:\d",topdown=False):

for file2 in fileName:

if getFilename\_extension(file2)=='txt’:

i=i+1

print(file2)

print(i)

topdown=False：自下而上的遍历文件;默认为自上而下

方法2:glob模块提供了一个函数用于从目录通配符搜索中生成文件列表:

例子：

import glob

#获取指定目录下的所有图片

print glob.glob(r"E:\Picture\\*\\*.jpg")

**import** glob  
path1=**r'Q:/d/'**fileName=**r'/\*.txt'**listAll=[]  
listFiles=glob.glob(path1+fileName)  
**while** len(listFiles):  
 listAll.extend(listFiles)  
 path1=path1+**"/\*"** listFiles = glob.glob(path1 + fileName)  
print(listAll)

**第三部分**

方法3：

递归算法：自己调用自己的函数叫做递归函数：

斐波那契数列：

1、1、2、3、5、8、13、21、34

**def** feibo(n):  
 **if** n==1:  
 **return** 1  
 **elif** n==2:  
 **return** 1  
 **else**:  
 **return** feibo(n-2)+feibo(n-1)  
print(feibo(10))

知识点：

1. os.listdir() 方法用于返回指定的文件夹包含的文件或文件夹的名字的列表。
2. os.path.join：将多个路径组合后返回

3、os.path.isdir()函数来判断路径是否为目录。

思路：

1. 通过os.listdir()获取该目录下所有的目录和文件的列表

def get\_py(path,l):

fileList = os.listdir(path)

3、遍历列表，判断是否是目录，如果是目录，继续获取该目录下所有的目录和文件的列表，

如果是文件，则加入用于存储py文件的列表。

for filename in fileList:

#获取path与filename组合后的路径

pathTmp = os.path.join(path,filename)

print( pathTmp)

#如果是目录

if os.path.isdir(pathTmp):

get\_py(pathTmp,l)

#则递归查找

elif filename[-4:]=='.txt':

#不是目录,把文件加入列表

l.append(pathTmp)

完整的源代码：

#coding=utf-8

import os

import os.path

#获取指定目录及其子目录下的py文件路径说明：用于存储找到的py文件路径 get\_py 函数，递归查找并存储py文件路径于l。

l = []

def get\_py(path,l):

fileList = os.listdir(path)

print( fileList)

#获取path目录下所有文件

for filename in fileList:

#获取path与filename组合后的路径

pathTmp = os.path.join(path,filename)

print( pathTmp)

#如果是目录

if os.path.isdir(pathTmp):

get\_py(pathTmp,l)

#则递归查找

elif filename[-4:]=='.txt':

#不是目录,把文件加入列表

l.append(pathTmp)

path = input('请输入路径:').strip()

get\_py(path,l)

print('在%s目录及其子目录下找到%d个py文件\n分别为：\n'%(path,len(l)))

for filepath in l:

print(filepath+'\n')