Python程序设计

文件

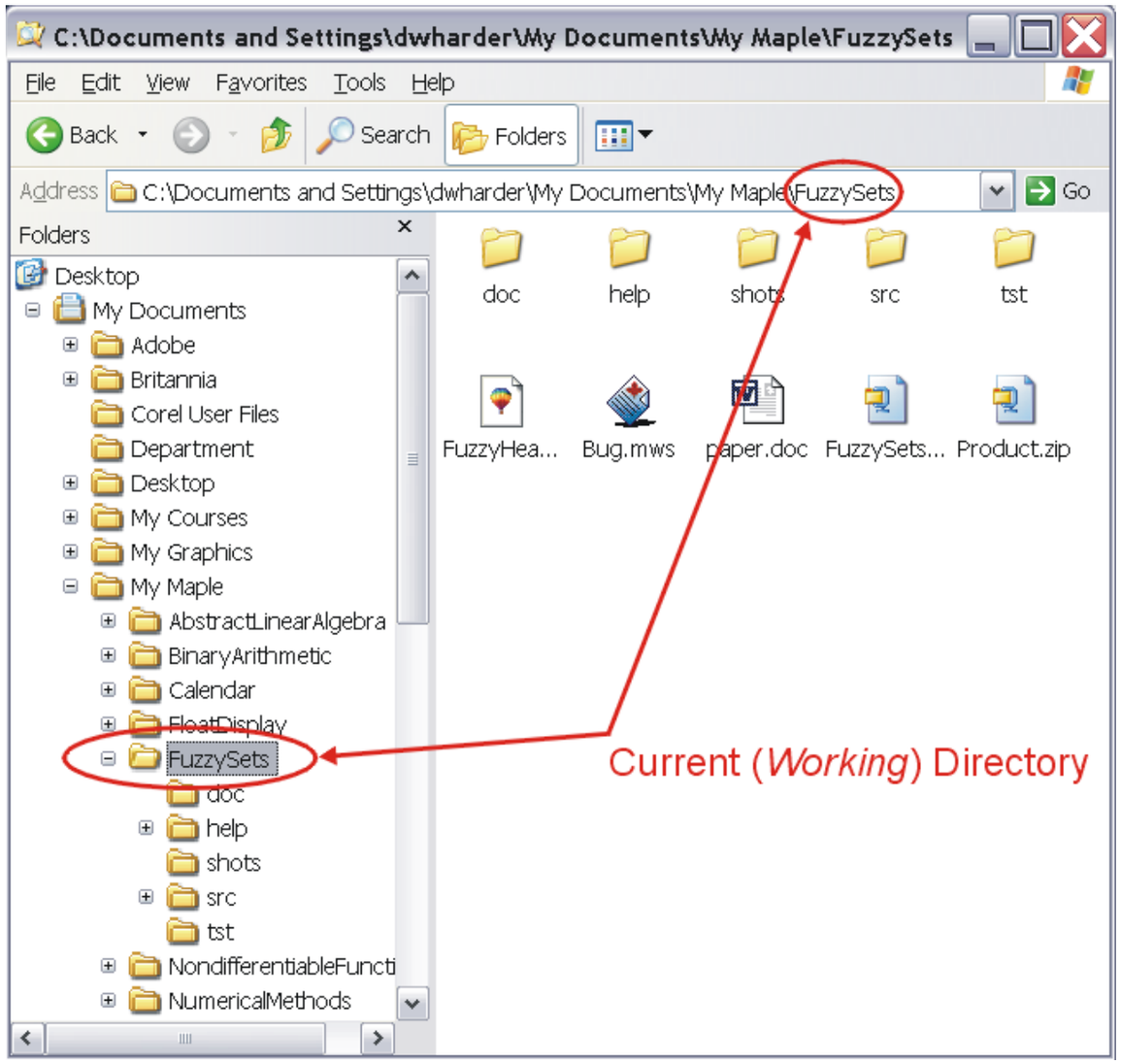
刘安
苏州大学，计算机科学与技术学院

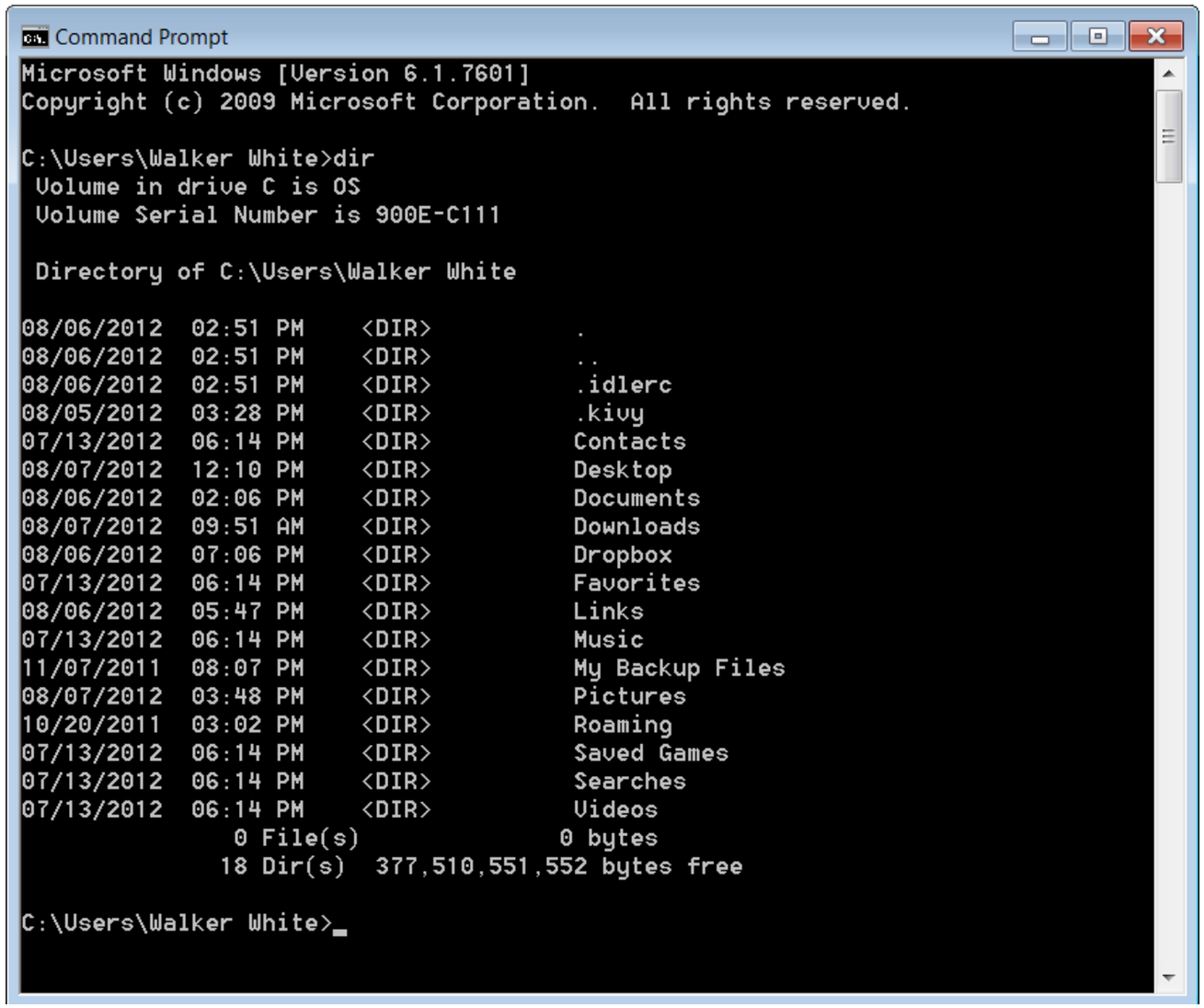
http://web.suda.edu.cn/anliu/

文件路径

- 文件路径指明文件在计算机上的位置
  - Windows - C:\Users\ryan\Documents\report.docx
  - OS X - /Users/ryan/Downloads/googlechrome.dmg
- Windows和OS X中文件夹和文件名不区分大小写，Linux中文件夹和文件名区分大小写
- 根文件夹
  - Windows - C:\ （注意是反斜杠）
  - OS X和Linux中 - / （斜杠）

当前工作目录


C:\Documents and Settings\dwharder\My Documents\My Maple\FuzzySets
File Edit View Favorites Tools Help
Back Search Folders
Address C:\Documents and Settings\dwharder\My Documents\My Maple\FuzzySets
Go
Folders
Desktop
My Documents
Adobe
Britannia
Corel User Files
Department
Desktop
My Courses
My Graphics
My Maple
AbstractLinearAlgebra
BinaryArithmetic
Calendar
FloatDisplay
FuzzySets
doc
help
shots
src
tst
NondifferentiableFuncti
NumericalMethods
doc
help
shots
src
tst
FuzzyHea...
Bug.mws
paper.doc
FuzzySets...
Product.zip
Current (Working) Directory


Command Prompt
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.
C:\Users\Walker White>dir
Volume in drive C is OS
Volume Serial Number is 900E-C111
Directory of C:\Users\Walker White
08/06/2012 02:51 PM <DIR> .
08/06/2012 02:51 PM <DIR> ..
08/06/2012 02:51 PM <DIR> .idlerc
08/05/2012 03:28 PM <DIR> .kivy
07/13/2012 06:14 PM <DIR> Contacts
08/07/2012 12:10 PM <DIR> Desktop
08/06/2012 02:06 PM <DIR> Documents
08/07/2012 09:51 AM <DIR> Downloads
08/06/2012 07:06 PM <DIR> Dropbox
07/13/2012 06:14 PM <DIR> Favorites
08/06/2012 05:47 PM <DIR> Links
07/13/2012 06:14 PM <DIR> Music
11/07/2011 08:07 PM <DIR> My Backup Files
08/07/2012 03:48 PM <DIR> Pictures
10/20/2011 03:02 PM <DIR> Roaming
07/13/2012 06:14 PM <DIR> Saved Games
07/13/2012 06:14 PM <DIR> Searches
07/13/2012 06:14 PM <DIR> Videos
0 File(s) 0 bytes
18 Dir(s) 377,510,551,552 bytes free
C:\Users\Walker White>

当前工作目录

- os.getcwd( ) - 返回表示当前工作目录的字符串

```
In [1]: import os

In [2]: os.getcwd()
Out[2]: 'C:\\Users\\ryan'
```

Windows

```
In [1]: import os

In [2]: os.getcwd()
Out[2]: '/Users/ryan'
```

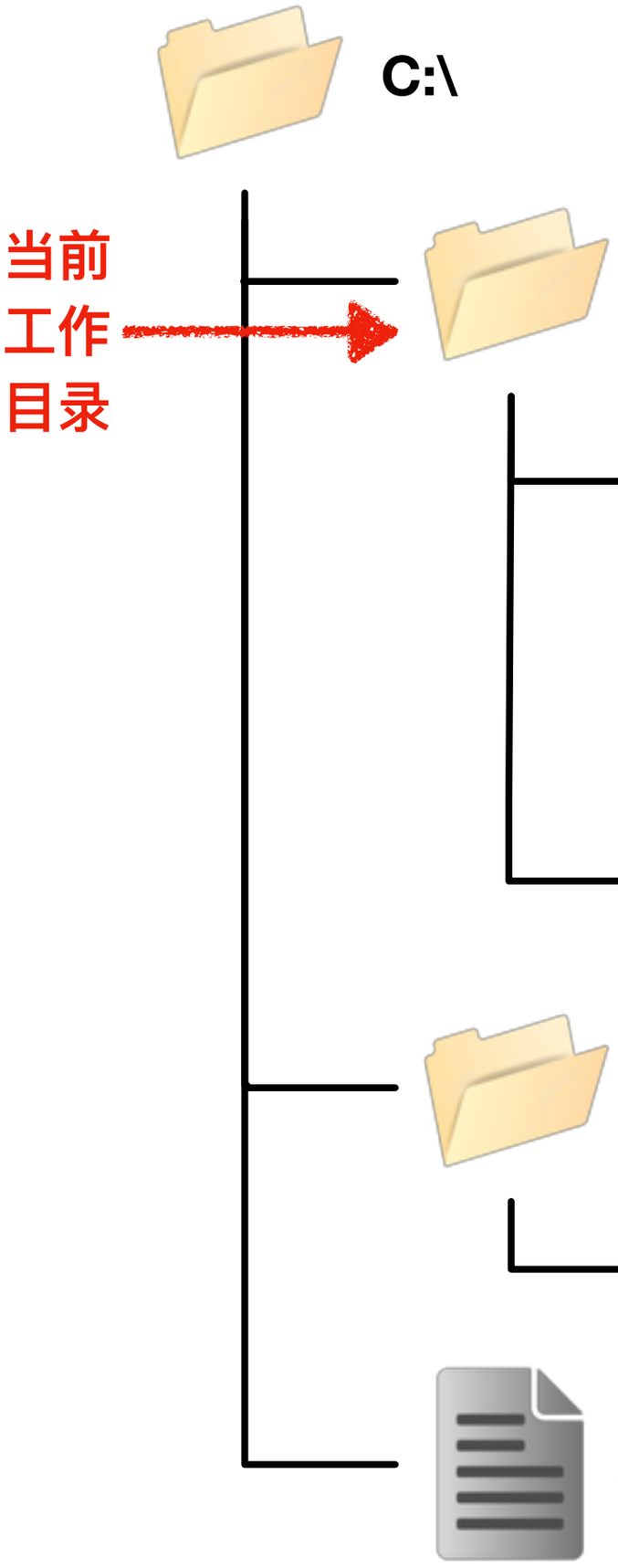
OS X

- Windows中使用反斜杠\作为路径中文件夹之间的分隔符
- OS X和Linux使用斜杠/作为路径中文件夹之间的分隔符

注意：在C:\\Users\\Ryan中，使用转义字符表示反斜杠

绝对路径和相对路径

- 绝对路径：总是从根文件夹开始
- 相对路径：相对于当前工作目录
- 路径中可以使用两个特殊的符号
  - .当前文件夹
  - ..父文件夹



	相对路径	绝对路径
C:\	..\	C:\
bacon	.\	C:\bacon
fizz	.\fizz	C:\bacon\fizz
spam.txt	.\fizz\spam.txt	C:\bacon\fizz\spam.txt
spam.txt	.\spam.txt	C:\bacon\spam.txt
eggs	..\eggs	C:\eggs
spam.txt	..\eggs\spam.txt	C:\eggs\spam.txt
spam.txt	..\spam.txt	C:\spam.txt

相对路径开始处的.\是可选的，比如.\spam.txt和spam.txt是同一个文件

绝对路径和相对路径

- os.path.abspath(path) - 返回参数path的绝对路径
- os.path.isabs(path) - 如果path是绝对路径，返回True

```
>>> os.getcwd() #当前工作目录
'/Users/ryan'
>>>
>>> os.path.abspath('.')
'/Users/ryan'
>>>
>>> os.path.isabs('.')
False
>>>
>>> os.path.isabs('/Users/ryan')
True
```

注意：上述函数并不检查路径的有效性

绝对路径和相对路径

- os.path.relpath(path, start = os.curdir) - 返回从start到path的相对路径，start的默认值是当前目录

```
>>> os.path.relpath('/Users/ryan/Downloads',
                    '/Users')
'ryan/Downloads'
>>>
>>> os.path.relpath('/Users/ryan/Downloads',
                    '/Users/ryan/Documents/Temp')
'../../Downloads'
>>>
>>> os.path.relpath('/Users/ryan/Documents/Temp')
'Documents/Temp'
>>>
>>> os.getcwd()
'/Users/ryan'
```

注意：该函数并不检查路径的有效性

目录名称和基本名称

- 基本名称：路径中最后一个分隔符之后的部分
- 目录名称：路径中最后一个分隔符之前的部分
- os.path.split(path) - 将path拆分成目录名称和基本名称

```
>>> os.path.split('/Users/ryan/Downloads')
('/Users/ryan', 'Downloads')
>>>
>>> os.path.split('/Users/ryan/Downloads/')
('/Users/ryan/Downloads', '')
>>>
>>> os.path.split('/Users/ryan/Downloads/zoom.pkg')
('/Users/ryan/Downloads', 'zoom.pkg')
>>>
>>> os.path.split('')
('', '')
```

基本名称可能是一个文件名
也有可能是一个目录名

目录名称和基本名称

- os.path.dirname(path) - 返回path中的目录名称
- os.path.basename(path) - 返回path中的基本名称

```
>>> path = '/Users/ryan/Downloads/zoom.pkg'
>>>
>>> os.path.split(path)
('/Users/ryan/Downloads', 'zoom.pkg')
>>>
>>> os.path.dirname(path)
'/Users/ryan/Downloads'
>>>
>>> os.path.basename(path)
'zoom.pkg'
```

检查路径有效性

- os.path.exists(path) - 如果path所指的文件或文件夹存在，返回True，否则返回False
- os.path.isdir(path) - 如果path是一个存在的文件夹，返回True，否则返回False
- os.path.isfile(path) - 如果path是一个存在的文件，返回True，否则返回False

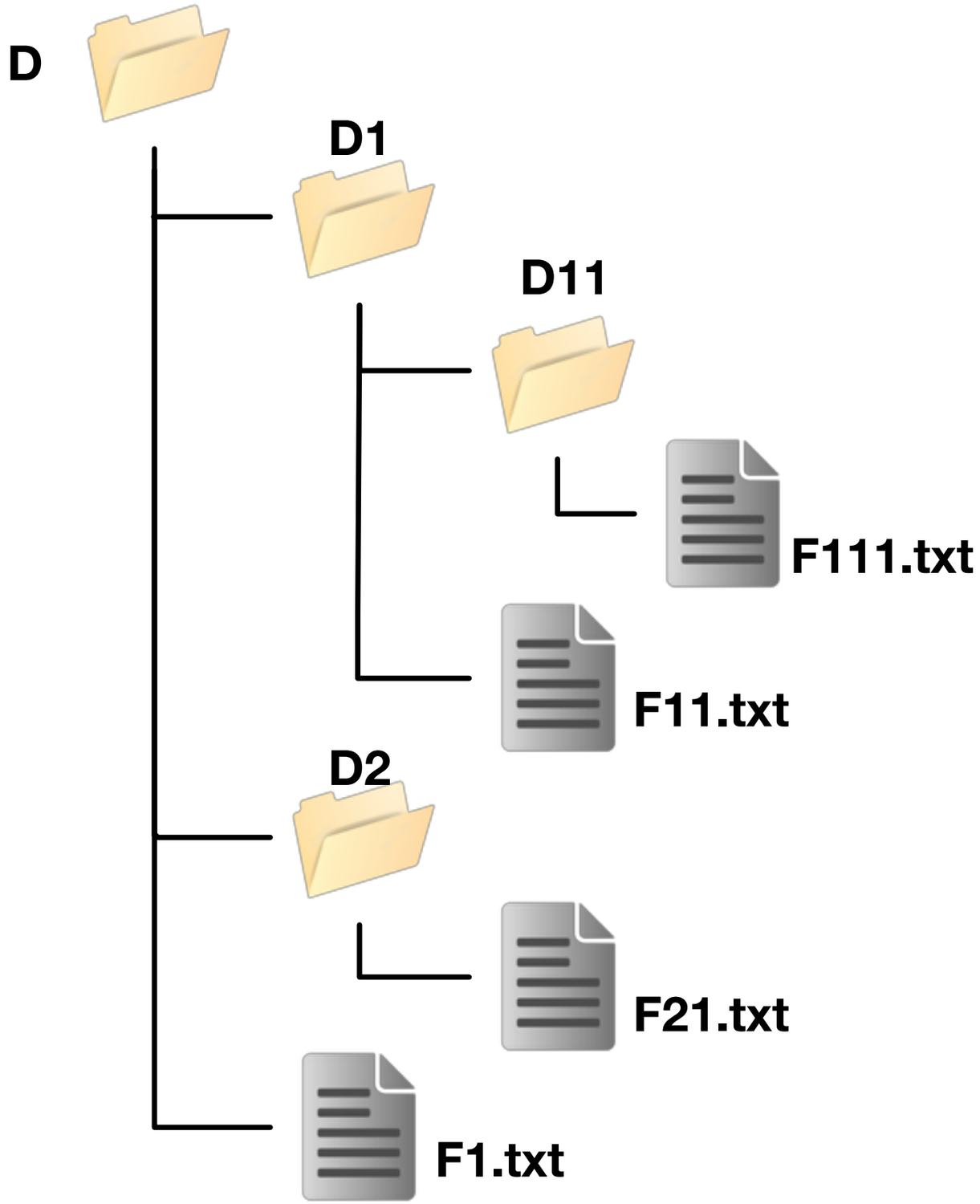
```
>>> os.path.exists('/Users/ryan')
True
>>>
>>> os.path.isdir('/Users/ryan')
True
>>>
>>> os.path.isfile('/Users/ryan/Downloads/zoom.pkg')
True
```

使用os.path.join( )创建路径

- os.path.join(path, *paths) - 将一个或多个path用目录分隔符（os.sep）拼接起来

```
>>> path = '/Users/ryan/Downloads/zoom.pkg'
>>> print(os.getcwd())
/Users/ryan
>>> print(os.sep) #目录分隔符
/
>>> os.path.join(os.getcwd(), 'Downloads', 'zoom.pkg')
'/Users/ryan/Downloads/zoom.pkg'
>>>
>>> os.path.join(*os.path.split(path)) #元组解包
'/Users/ryan/Downloads/zoom.pkg'
```

列出一个文件中所有文件的大小



查看文件大小和文件夹内容

- os.path.getsize(path) - 返回文件/文件夹path的字节数
- os.listdir(path='.') - 返回一个列表，其中包含文件夹path中的所有文件/文件夹名

```
>>> path = '/Users/ryan/Documents/D'
>>> os.listdir(path) #.DS_Store是OSX自动生成的隐藏文件
['.DS_Store', 'F1.txt', 'D1', 'D2']
>>> for x in os.listdir(path):
        x_path = os.path.join(path, x)
        print(x_path, os.path.getsize(x_path))


/Users/ryan/Documents/D/.DS_Store 8196
/Users/ryan/Documents/D/F1.txt 10
/Users/ryan/Documents/D/D1 160
/Users/ryan/Documents/D/D2 96
```

```python
import os

def print_dir(path):
    if os.path.isfile(path):
        print(path, os.path.getsize(path))
    if os.path.isdir(path):
        for x in os.listdir(path):
            x_path = os.path.join(path, x)
            print_dir(x_path)
```

```
>>> print_dir('/Users/ryan/Documents/D')
/Users/ryan/Documents/D/.DS_Store 8196
/Users/ryan/Documents/D/F1.txt 10
/Users/ryan/Documents/D/D1/.DS_Store 6148
/Users/ryan/Documents/D/D1/D11/F111.txt 40
/Users/ryan/Documents/D/D1/F11.txt 50
/Users/ryan/Documents/D/D2/F21.txt 30
```

使用os.walk遍历目录

- os.walk(path) - 对于目录path中的每一个文件夹D，生成一个元组(dirpath, dirnames, filenames)，其中字符串dirpath是D的绝对路径，列表dirnames包含D中所有直接子文件夹的名字，列表filenames包含D中所有文件的名字

```
>>> t = os.walk('/Users/ryan/Documents/D')
>>> t
<generator object walk at 0x1050e95d0>
>>> for x in t: print(x)

('/Users/ryan/Documents/D', ['D1', 'D2'], ['.DS_Store', 'F1.txt'])
('/Users/ryan/Documents/D/D1', ['D11'], ['.DS_Store', 'F11.txt'])
('/Users/ryan/Documents/D/D1/D11', [], ['F111.txt'])
('/Users/ryan/Documents/D/D2', [], ['F21.txt'])
```

```python
import os

def print_dir_walk(path):
    for root, dirs, files in os.walk(path):
        for f in files:
            f_path = os.path.join(root, f)
            print(f_path, os.path.getsize(f_path))
```

```
>>> print_dir_walk('/Users/ryan/Documents/D')
/Users/ryan/Documents/D/.DS_Store 8196
/Users/ryan/Documents/D/F1.txt 10
/Users/ryan/Documents/D/D1/.DS_Store 6148
/Users/ryan/Documents/D/D1/F11.txt 50
/Users/ryan/Documents/D/D1/D11/F111.txt 40
/Users/ryan/Documents/D/D2/F21.txt 30
```

文件读写步骤

- 调用open( )函数，返回一个文件对象
- 通过文件对象的相关方法对文件进行读写
- 调用文件对象的close( )方法关闭文件

```
>>> path = '/Users/ryan/Downloads/student.txt'
>>> f = open(path) #打开文件
>>> msg = f.read() #读取文件内容
>>> print(msg)
1011 Alice 80 90 90
1003 Bob 55 85 70
1008 John 95 90 80
1005 Tom 75 70 50
>>> f.close() #关闭文件
```

open函数

- open(file, mode = 'r')：以参数mode指定的模式打开参数file指定的文件，常见的mode如下
  - t：文本文件模式，默认值
  - b：二进制文件模式
  - r：读模式，默认值，不支持写文件
  - w/x/a：都属于写模式，不支持读文件
  - w：如果文件已存在则覆盖已有内容，否则创建新文件
  - x：如果文件已存在则出错，否则创建新文件
  - a：追加模式，在已有文件末尾增加内容，不支持读文件
  - r+/w+/a+：更新模式（支持读写）

文本文件和二进制文件

- 计算机使用二进制，任何数据（文件）都是二进制串
  - *01100001 01100010 01100011*
- 编/解码：将对象转换成二进制串/将二进制串转换成对象
- 文本文件可以使用ASCII编码
  - 读取时：其二进制串的每8个比特转换成一个字符
  - 存储时：每个字符转换成ASCII码（8个比特）
- 不同类型的二进制文件使用不同的编码
  - 文档(doc, pdf, ppt)，压缩文件(zip, rar, iso)
  - 图片(jpg, png, gif)，视频(mp4, mkv, avi)，音频(mp3, wav)

使用open函数

```
>>> tmp_file = '/Users/ryan/Downloads/temp.txt'
>>> f = open(tmp_file, mode='x') #若文件已存在，则抛出异常
>>> f.write('12345') #文本模式下，该函数返回写入的字符数
5
>>> f.close()
>>> f = open(tmp_file, mode='a') #追加写模式
>>> f.write('6789')
4
>>> f.close()
>>> f = open(tmp_file) #默认文本模式和读模式
>>> f.read()
'123456789'
>>> f.close()
```

文本模式下读文件

- 文本文件：通过换行符\n分隔的若干行文本
- read(size = -1)：从文件当前位置读size个字符，如果size等于-1，读取文件的所有内容

```
>>> f = open(tmp_file) #默认文本模式和读模式
>>> f.read(1)
'1'
>>> f.read(2)
'23'
>>> f.read(-1)
'456789'
>>> f.read(1)
''
```

文本模式下读文件

- readline(size = -1)：如果没有设置size的大小，从文件当前位置读一行字符，否则，最多读取size个字符

```
>>> stu_file = '/Users/ryan/Downloads/student.txt'
>>> f = open(stu_file)
>>> f.read()
'1011 Alice 80 90 90\n1003 Bob 55 85 70\n1008 John 95 90 80\n1005 Tom 75 70 50'
>>> g = open(stu_file)
>>> g.readline()
'1011 Alice 80 90 90\n'
>>> g.readline(4)
'1003'
>>> g.readline()
' Bob 55 85 70\n'
```

文本模式下读文件

- readlines( )：从文件当前位置以行为单位读取所有内容，返回一个包含所有行的列表（每个元素对应一行）

```
>>> stu_file = '/Users/ryan/Downloads/student.txt'
>>> f = open(stu_file)
>>> f.readlines()
['1011 Alice 80 90 90\n', '1003 Bob 55 85 70\n', '100
8 John 95 90 80\n', '1005 Tom 75 70 50']
>>> g = open(stu_file)
>>> g.readline()
'1011 Alice 80 90 90\n'
>>> g.readlines()
['1003 Bob 55 85 70\n', '1008 John 95 90 80\n', '1005
Tom 75 70 50']
```

文本模式下写文件

- write(text)：将字符串text写入文件当前位置
- writelines(list_str)：将列表list_str中所有字符串写入文件
- 上述方法都不会在写入的字符串后面添加\n

```
>>> prime_file = '/Users/ryan/Downloads/prime.txt'
>>> f = open(prime_file, 'w')
>>> f.write('2')
1
>>> f.writelines(['3', '5', '7'])
>>> f.close()
>>> f = open(prime_file)
>>> f.readlines()
['2357']
```

open函数返回的文件对象是可迭代的

- 注意第三行代码中的列表推导利用了文件对象的可迭代性

```
>>> back_path = '/Users/ryan/Downloads/backup.txt'
>>> back_file = open(back_path, 'w')
>>> list_str = [line for line in open(stu_file)]
>>> back_file.writelines(list_str)
>>> back_file.close()
>>> back_file = open(back_path)
>>> back_file.read()
'1011 Alice 80 90 90\n1003 Bob 55 85 70\n1008 John 95
90 80\n1005 Tom 75 70 50'
```

利用pickle模块读写python对象

- dump(obj, file)：将对象obj的二进制表示写入文件file中
- load(file)：从文件file中读取对象

```
>>> import pickle
>>> f = open('/Users/ryan/Downloads/obj.dat', 'wb')
>>> pickle.dump([1, 2, 3], f) #将列表写入文件
>>> pickle.dump('python', f) #将字符串写入文件
>>> pickle.dump(3.141592, f) #将浮点数写入文件
>>> pickle.dump({'the':3, 'what':5}, f) #将字典写入文件
>>> f.close()
```

利用pickle模块读写python对象

- load(file)：从文件file中读取对象

```
>>> f = open('/Users/ryan/Downloads/obj.dat', 'rb')
>>> a = pickle.load(f)
>>> b = pickle.load(f)
>>> c = pickle.load(f)
>>> d = pickle.load(f)
>>> print(type(a), a)
<class 'list'> [1, 2, 3]
>>> print(type(b), b)
<class 'str'> python
>>> print(type(c), c)
<class 'float'> 3.141592
>>> print(type(d), d)
<class 'dict'> {'the': 3, 'what': 5}
```

当文件到达末尾时调用load函数，会抛出EOFError异常