nsd1904_py01_day04

shutil模块

用于实现文件复制等操作

```
>>> import shutil
>>> f1 = open('/bin/touch', 'rb')
>>> f2 = open('/tmp/tch', 'wb')
>>> shutil.copyfileobj(f1, f2)
>>> f1.close()
>>> f2.close()

# 拷贝文件和权限
>>> shutil.copy('/bin/mkdir', '/tmp/')
# cp -r /etc/security /tmp/anquan
>>> shutil.copytree('/etc/security', '/tmp/anquan')
# mv /tmp/anquan /var/tmp/
>>> shutil.move('/tmp/anquan', '/var/tmp')
# rm -rf /var/tmp/anquan
>>> shutil.rmtree('/var/tmp/anquan')
# chown bob.bob /tmp/tch
>>> shutil.chown('/tmp/tch', user='bob', group='bob')
```

subprocess模块

该模块用于执行系统命令

```
# 在shell环境下执行系统命令
>>> subprocess.run('ls ~', shell=True)
# 将输出保存到stdout，将错误保存到stderr
>>> result = subprocess.run('id root; id john', shell=True, stdout=subprocess.PIPE,
stderr=subprocess.PIPE)
# 查看状态码和输出、错误
>>> result.returncode   # 就是shell中的$?
1
>>> result.stdout
b'uid=0(root) gid=0(root) \xe7\xbb\x84=0(root)\n'
>>> result.stderr
b'id: john: no such user\n'
# decode将bytes类型转换为str类型
>>> result.stdout.decode()
'uid=0(root) gid=0(root) 组=0(root)\n'
```

语法风格

```
# 链式多重赋值
>>> x = y = 10
# 多元赋值
>>> a, b = 10, 20
>>> c, d = 'ab'
>>> e, f = (100, 200)
>>> g, h = ['tom', 'jerry']
# 交换a和b的值
>>> a, b = b, a
```

关键字

```
>>> import keyword
>>> keyword.kwlist
['False', 'None', 'True', 'and', 'as', 'assert', 'break', 'class', 'continue', 'def',
'del', 'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if', 'import', 'in',
'is', 'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with',
'yield']
```

内建：https://docs.python.org/zh-cn/3/library/index.html

- 不是关键字
- 可以重新赋值

模块布局

```
#!/usr/bin/python      # 解释器位置
"文档字行串"

import time            # 模块导入
import os

all_chs = 'abcd'       # 全局变量定义
debug = True

class MyClass:         # 类定义
    pass

def func1():           # 函数定义
    pass

if __name__ == '__main__':
    mc = MyClass()
```

编程思路

例：创建文件

1. 发呆。思考程序的运行方式（交互？非交互？），程序提示什么？用户回什么？

```
# python mktxtfile.py
filename: /etc/hosts
/etc/hosts 文件已存在，请重试。
filename: /etc
/etc 文件已存在，请重试。
filename: /tmp/abc.txt
请输入内容，单独一行为end表示结束。
> hello world!
> the end
> how are you?
> end
# ls /tmp/abc.txt
/tmp/abc.txt
# cat /tmp/abc.txt
hello world!
the end
how are you?
```

2. 分析程序有哪些功能，将这些功能编写成功能函数

```
def get_fname():
    "用于获取文件名"

def get_content():
    "用于获取文件内容"

def wfile(fname, content):
    "将content内容写到文件fname中"
```

3. 思考函数调用顺序，写到程序主体代码中

```
def get_fname():
    "用于获取文件名"

def get_content():
    "用于获取文件内容"

def wfile(fname, content):
    "将content内容写到文件fname中"

if __name__ == '__main__':
    fname = get_fname()
    content = get_content()
    wfile(fname, content)
```

4. 编写每个函数具体的代码

判断某一文件是否存在

```
>>> import os
>>> os.path.exists('/etc/')
True
>>> os.path.exists('/etc/hosts')
True
>>> os.path.exists('/tmp/abc.txt')
False
```

序列对象

- 包括字符串、列表、元组

```
>>> str(100)
'100'
>>> list('abcd')   # 将字符串转为列表
['a', 'b', 'c', 'd']
>>> list(('tom', 'jerry'))  # 将元组转为列表
['tom', 'jerry']
>>> tuple('abcd')   # 把字符串转为元组
('a', 'b', 'c', 'd')
>>> tuple(['bob', 'alice'])   # 把列表转为元组
('bob', 'alice')

# reversed用于翻转序列对象
>>> from random import randint
>>> nums = [randint(1, 100) for i in range(10)]
>>> nums
[81, 23, 2, 70, 16, 96, 85, 98, 5, 49]
>>> list(reversed(nums))
[49, 5, 98, 85, 96, 16, 70, 2, 23, 81]
>>> for i in reversed(nums):
...   print(i)

# sorted用于排序
>>> sorted(nums)
[2, 5, 16, 23, 49, 70, 81, 85, 96, 98]
```

字符编码

- 美国常用的字符集是ASCII
- 西欧常用的字符集是iso-8859-1，也叫latin-1
- 中国大陆常用字符集是gbk / gb18030 / gb2312
- 世界各地都有自己的字符集，不利于信息交换。所以ISO创造了万国码unicode，它是一个非常大的字符编码方案，把全世界所有的文字都包含在内，每个字符都有唯一0/1组合。
- UTF8是unicode的一种编码方案，它采用变长编码方式。一个英文字符用1个字节表示，一个汉字用3个字节表示。
- str类型，是字符类型。一个英文字母是一个字符，一个汉字也是一个字符（UTF8）。
- btyes类型，是字节类型。一个英文字母是一个字节，一个汉字是3个字节（UTF8）。

```
>>> s = '中国'
>>> len(s)
2
>>> s.encode()   # 默认转为utf8的bytes，一个汉字3字节
b'\xe4\xb8\xad\xe5\x9b\xbd'
>>> s.encode('gb2312')   # 指定转成gb2312的bytes，一个汉字占2字节
b'\xd6\xd0\xb9\xfa'
>>> b = s.encode()
>>> b
b'\xe4\xb8\xad\xe5\x9b\xbd'
>>> b.decode()   # 将utf8编码的bytes转成str
'中国'
```

字符串格式化

```
>>> '%s is %s years old' % ('tom', 20)
'tom is 20 years old'
>>> '%s is %d years old' % ('tom', 20)  # 数字可用%d
'tom is 20 years old'
>>> '%8s%8s' % ('name', 'age')  # 每段各占8个字符的宽度
'    name     age'
>>> '%-8s%-8s' % ('name', 'age')   # 左对齐
'name    age     '

>>> '%f' % (5 / 3)
'1.666667'
>>> '%.2f' % (5 / 3)   # 小数位有2位
'1.67'
>>> '%6.2f' % (5 / 3)   # 总宽度为6，小数位占2位
'  1.67'
>>> '%e' % (1000000)   # 科学计数法
'1.000000e+06'
>>> '%#x' % 10   # 以16进制表示
'0xa'
>>> '%#o' % 10   # 以8进制表示
'0o12'
```

字符串格式化还可以使用format方法。

```
>>> '{} is {} years old'.format('bob', 20)
'bob is 20 years old'
>>> '{} is {} years old'.format(20, 'bob')
'20 is bob years old'
>>> '{1} is {0} years old'.format(20, 'bob')
'bob is 20 years old'
```

原始字符串/真实字符串

```
>>> win_path = 'c:\temp'
>>> print(win_path)
c:      emp
>>> win_path = 'c:\\temp'
>>> print(win_path)
c:\temp
>>> wpath = r'c:\temp'
>>> print(wpath)
c:\temp
>>> wpath
'c:\\temp'
```

字符串的方法

```
# strip去除字符串两端空白字符
>>> '  \thello   \n'.strip()
'hello'
# lstrip去除字符串左端空白字符
>>> '  \thello   \n'.lstrip()
'hello   \n'
# rstrip去除字符串右端空白字符
>>> '  \thello   \n'.rstrip()
'  \thello'
>>> 'hello'.center(30)   # 居中
'            hello             '
>>> 'hello'.center(30, '#')  # 居中，用#填充，而不是空格
'############hello#############'
>>> 'hello'.rjust(30, '#')   # 右对齐
'########################hello'
>>> 'hello'.ljust(30, '#')   # 左对齐
'hello########################'
>>> 'hello'.upper()   # 字母转成大写
'HELLO'
>>> 'Hao123'.lower()   # 字母转成小写
'hao123'
>>> 'hao123'.islower()   # 字母都是小写的吗？
True
>>> 'hao123'.isupper()   # 字母都是大写的吗？
False
>>> 'hao123'.isdigit()   # 所有的字符都是数字吗？
False
>>> '123'.isdigit()
True
>>> 'hello'.startswith('h')   # 字符串以h开头吗？
True
>>> 'hello'.startswith('he')   # 字符串以he开头吗？
True
>>> 'hello'.endswith('abc')   # 字符串以abc结尾吗？
False
>>> 'hello'.replace('e', 'ab')   # 把e替换为ab
'hablo'
```

```
>>> 'hello.tar.gz'.split('.')   # 用点切分字符串
['hello', 'tar', 'gz']

# 自行判断某一字符串是不是全由数字组成
>>> s1 = '1234'
>>> for c in s1:
...   if c not in '0123456789':
...     print(False)
...     break
... else:
...   print(True)
...
True

>>> s1 = '123a456'
>>> for c in s1:
...   if c not in '0123456789':
...     print(False)
...     break
... else:
...   print(True)
...
False
```