nsd1903_python1_day04

python官方手册页：https://docs.python.org/zh-cn/3/ -> 标准库参考

shutil模块

主要实现复制、移动等操作

```
import shutil

# 拷贝文件对象的方式，了解
f1 = open('/etc/passwd', 'rb')
f2 = open('/tmp/mima', 'wb')

shutil.copyfileobj(f1, f2)
f1.close()
f2.close()

# 直接拷贝文件
>>> shutil.copyfile('/etc/shadow', '/tmp/sd')
'/tmp/sd'

# 将文件拷贝到目标目录，或指定目标位置及名字，常用
>>> shutil.copy('/etc/hosts', '/tmp/')
'/tmp/hosts'
>>> shutil.copy('/etc/hosts', '/tmp/zhuji.txt')
'/tmp/zhuji.txt'

# copy2相当于是cp -p  ， 常用
>>> shutil.copy2('/etc/hosts', '/tmp/zhj')
'/tmp/zhj'

# cp -r /etc/security /tmp/anquan， 常用
>>> shutil.copytree('/etc/security', '/tmp/anquan')
'/tmp/anquan'

# mv /tmp/anquan /var/tmp/anquan
>>> shutil.move('/tmp/anquan', '/var/tmp/anquan')
'/var/tmp/anquan'

# chown
>>> shutil.chown('/tmp/mima', user='bob', group='bob')

# rm -rf
>>> shutil.rmtree('/var/tmp/anquan')
```

subprocess模块

用于执行系统命令。

```
# 在shell环境中执行命令ls ~
>>> subprocess.run('ls ~', shell=True)

>>> result = subprocess.run('ls abcd', shell=True)
ls: 无法访问abcd: 没有那个文件或目录
>>> result.returncode   # returncode就是$?
2

# 将输出保存到stdout中，将错误保存到stderr中
>>> result = subprocess.run('id root; id john', shell=True, stdout=subprocess.PIPE,
stderr=subprocess.PIPE)
>>> result.stdout
b'uid=0(root) gid=0(root) \xe7\xbb\x84=0(root)\n'
>>> result.stderr
b'id: john: no such user\n'
```

bytes和str的转换

- 字母a对应的10进制数是97，2进制是0b01100001
- 一个字节是8位，一个ASCII字符可以用一个字节表示出来
- 所以bytes类型的数据，一个字节正好能表示成一个ASCII字符时，就显示成字符
- 汉字使用utf8编码，一个汉字需要占3字节。一个字节表示不出来汉字，所以一个汉字就需要使用三个以\x开头的16进制数表示
- str类型的字符串是引号括起来的部分
- bytes类型的字符串，以b''表示

```
# bytes类型转成str类型
>>> result.stdout.decode()
'uid=0(root) gid=0(root) 组=0(root)\n'

>>> hi = '你好tom'
>>> hi.encode()   # str => bytes
b'\xe4\xbd\xa0\xe5\xa5\xbdtom'

>>> data = hi.encode()
>>> data
b'\xe4\xbd\xa0\xe5\xa5\xbdtom'
>>> data.decode()   # bytes => str
'你好tom'
```

python语法风格

```
# 链式多重赋值
>>> a = b = 10
>>> a
10
>>> b
```

```
10
>>> b = 20
>>> a
10

>>> alist = blist = [1, 2, 3]
>>> blist[-1] = 30
>>> blist
[1, 2, 30]
>>> alist
[1, 2, 30]

# 多元赋值
>>> a, b = 10, 20
>>> a
10
>>> b
20
>>> c, d = 'ab'
>>> c
'a'
>>> d
'b'
>>> e, f = [10, 20]
>>> e
10
>>> f
20
>>> m, n = (100, 200)
>>> m
100
>>> n
200

# 其他语言交换变量值的方法
>>> a, b = 10, 20
>>> t = a
>>> a = b
>>> b = t
>>> a
20
>>> b
10

# python交换两个变量值的方法
>>> a, b = b, a
>>> a
10
>>> b
20
```

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：

python创建的一些函数等。

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

python模块布局

```
#!/usr/local/bin/python     # 解释器
"""文档字符串

这是出现在help中的部分"""

import string   # 模块导入
import time

all_chs = string.ascii_letters + digits   # 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()
    func1()
```

编程思路

- 发呆。思考程序的运行方式（交互？非交互？），运行场景
- 思考程序有哪些功能，将这些功能写为函数，写出大体框架
- 编写程序主体。按顺序调用函数
- 编写函数内容

1. 运行方式

```
# python mkfile.py
文件名： /etc/hosts
文件已存在，请重试。
文件名：/etc/
文件已存在，请重试。
文件名：/tmp/abc.txt
请输入内容，输入end结束输入：
```

```
(end to quit)> hello world.
(end to quit)> ni hao.
(end to quit)> exit end
(end to quit)> end
# ls /tmp/abc.txt
abc.txt
# cat /tmp/abc.txt
hello world.
ni hao.
exit end
```

2. 编写功能函数

```
def get_fname():


def get_content():


def wfile(fname, content):
```

3. 程序主体

```
def get_fname():


def get_content():


def wfile(fname, content):


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

4. 编写函数内容

```
import os

def get_fname():
    while True:
        fname = input('文件名: ')
        # os.path.exists(fname) => 文件已存在返回True
        if not os.path.exists(fname):
            break
        print('文件已存在，请重试')

    return fname
```

```
def get_content():
    content = []

    print('请输入内容，输入end结束输入：')
    while True:
        line = input('(end to quit)> ')
        if line == 'end':
            break
        content.append(line + '\n')

    return content

def wfile(fname, content):
    with open(fname, 'w') as fobj:
        fobj.writelines(content)

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

序列对象

```
>>> list('abcd')    # 把字符串转成列表
['a', 'b', 'c', 'd']
>>> list(range(1, 11))
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
>>> list((10, 20, 30))   # 把元组转成列表
[10, 20, 30]

>>> tuple('abcd')
('a', 'b', 'c', 'd')
>>> tuple(range(1, 11))
(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
>>> tuple([10, 20, 30])  # 把列表转成元组
(10, 20, 30)

# 将各种对象转换成字符串
>>> str(100)
'100'
>>> str([10, 20, 30])
'[10, 20, 30]'
>>> str((100, 200, 300))
'(100, 200, 300)'

# 翻转函数
>>> reversed('abcd')
<reversed object at 0x7f8155118ba8>
>>> list(reversed('abcd'))
['d', 'c', 'b', 'a']
>>> for ch in reversed('abcd'):
```

```
...   print(ch)

>>> 'abcd'[::-1]   # 翻转
'dcba'

# 排序
>>> sorted('qwertyu')
['e', 'q', 'r', 't', 'u', 'w', 'y']
>>> sorted([2, 343, 2, 23, 4545, 23, 532])
[2, 2, 23, 23, 343, 532, 4545]
```

字符编码

- 计算机内部存储时，都是2进制的0和1
- 可以提前预定义好一串0/1的组合代表什么字符
- ASCII是美国信息交换标准代码的简称，用7位表示字符
- 欧洲主要采用Latin-1，即ISO-8859-1字符集，共8位
- 中国采用的是gbk / gb2313 / gb18030字符集
- ISO国际标准化组织制定了万国码Unicode，utf8是其中的一种编码方案，它采用变长的编码方案，如果是英文字符，直接用1个字节表示，如果是汉字，用三个字节表示。

```
>>> s1 = '中国'
>>> s1.encode()    # 默认使用utf8编码，显示中国的utf8编码
b'\xe4\xb8\xad\xe5\x9b\xbd'
>>> s1.encode('gbk')   # 明确指明使用的编码方案是gbk
b'\xd6\xd0\xb9\xfa'
```

字符串

字符串格式化

```
# 常用的字符串格式化方法
>>> '%s is %s years old.' % ('tom', 20)
'tom is 20 years old.'
>>> '%s is %d years old.' % ('tom', 20)
'tom is 20 years old.'
>>> '%d is %d years old.' % ('tom', 20)  # 报错
>>> '%10s%10s' % ('tom', 20)  # 第个占位符共10个宽度，不够的用空格补齐
>>> '%-10s%-10s' % ('tom', 20)   # 左对齐
>>> '%-10s%-10s' % ('jerry', 19)

# 不常用，了解
>>> '%f' % (5 / 3)
'1.666667'
>>> '%d' % (5 / 3)
'1'
>>> '%.2f' % (5 / 3)   # 有2位小数
'1.67'
>>> '%5.2f' % (5 / 3)  # 共5个宽度，有2位小数
' 1.67'
```

```
>>> '%c' % 97   # ASCII码97代表的字符
'a'
>>> '%#o' % 10   # 转成8进制
'0o12'
>>> '%#x' % 10   # 转成16进制
'0xa'
>>> '%010d' % 5   # 共10个宽度，不够的用0补齐
'0000000005'
```

使用format方法实现字符串格式化

```
# 使用{}作为占位符
>>> '{} is {} years old.'.format('tom', 20)
'tom is 20 years old.'
>>> '{1} is {0} years old.'.format(20, 'tom')
'tom is 20 years old.'
>>> '{:<10}{:<10}'.format('tom', 20)  # 左对齐
'tom       20        '
>>> '{:>10}{:>10}'.format('tom', 20)   # 右对齐
'       tom        20'
```

原始字符串/真实字符串

```
>>> win_path = 'c:\temp\new'
>>> print(win_path)
c:  emp
ew

>>> win_path = 'c:\\temp\\new'
>>> print(win_path)
c:\temp\new

# 以上写法可以改为原始字符串
>>> wpath = r'c:\temp\new'
>>> print(wpath)
c:\temp\new
>>> wpath
'c:\\temp\\new'
```

字符串方法

```
>>> s1 = 'hello world'
>>> s2 = 'HAO 123'
>>> s3 = 'hao123'
>>> s4 = '7298302'
>>> s5 = '\t  hello world\n'
>>> s1.center(48)  # 居中，总宽度48
'                 hello world                  '
>>> s1.center(48, '*')
'*****************hello world*****************'
>>> s1.ljust(48, '#')
```

```
'hello world########################################'
>>> s1.rjust(48, '#')
'####################################hello world'
>>> s1.upper()
'HELLO WORLD'
>>> s2.lower()
'hao 123'
>>> s5.strip()   # 去除字符串两端的空白字符
'hello world'
>>> s5.rstrip()
'\t  hello world'
>>> s5.lstrip()
'hello world\n'

>>> s1.startswith('h')  # 以h开头吗
True
>>> s1.startswith('he')  # 以he开头吗？
True
>>> s1.endswith('ab')   # 以ab结尾吗？
False
>>> s1.replace('l', 'a')   # 替换l为a
'heaao worad'
>>> s1.replace('ll', 'ab')  # 替换ll为ab
'heabo world'
>>> s1.split()   # 默认以空格作为分隔符切割
['hello', 'world']
>>> 'hello.world.ni.hao'.split('.')   # 将点作为分隔符
['hello', 'world', 'ni', 'hao']
>>> '-'.join(['abc', 'hello', 'hao'])  # 用-拼接
'abc-hello-hao'

>>> from random import choice
>>> from string import ascii_letters, digits
>>> all_chs = ascii_letters + digits
>>> [choice(all_chs)]
['b']
>>> [choice(all_chs) for i in range(8)]
['h', 'n', '4', 'o', 'z', 'A', 'v', 'y']
>>> ''.join([choice(all_chs) for i in range(8)])
'Saqo19u0'

>>> s1
'hello world'
>>> s1.islower()   # 字符串中的所有字母都是小写吗？
True

>>> s2
'HAO 123'
>>> s2.isupper()   # 字符串中的所有字母都是大写吗？
True
>>> s2.isdigit()   # 字符串中的所有字符都是数字吗？
False
>>> s4
```

```
'7298302'
>>> s4.isdigit()
True

>>> 'Hao'.isalpha()   # 所有的字符都是字母吗？
True
>>> 'Hao123'.isalpha()
False
>>> 'Hao123'.isalnum()  # 所有的字符都是字母和数字吗？
True

>>> s4
'7298302'
>>> s4.isdigit()
True
>>>
>>> for ch in s4:
...   if ch not in '0123456789':
...     print(False)
...     break
... else:
...   print(True)
...
True
```