nsd1907_py01_day03

文件

操作步骤：打开、读写、关闭

读取文本文件

```
(nsd1907) [root@room8pc16 day03]# cp /etc/passwd /tmp/
# 默认以r方式打开，文件不存在则报错
>>> f = open('/tmp/password')  # 报错
>>> f = open('/tmp/passwd')
>>> data = f.read()  # 读取全部内容，保存到data中
>>> print(data)
>>> data = f.read()  # 文件指针已到结尾，再读取将为空串
>>> data
''
>>> f.close()   # 关闭文件

>>> f = open('/tmp/passwd')
>>> f.read(4)  # 读取4字节
'root'
>>> f.read(6)  # 继续向后读取6字节
':x:0:0'
>>> f.readline()   # 继续向后读到行尾
':root:/root:/bin/bash\n'
>>> f.readline()   # 再读一行
'bin:x:1:1:bin:/bin:/sbin/nologin\n'
>>> data = f.readlines()   # readlines将剩余所有行读到列表中
>>> type(data)
<class 'list'>
>>> data
>>> f.close()

# ***最常用的读取文件文件的方法***
>>> f = open('/tmp/passwd')
>>> for line in f:
...   print(line, end='')
>>> f.close()
```

读取非文本文件

```
# 打开图片文件
>>> f = open('/tmp/girl.jpg')
>>> f.read(10)   # 报错，因为python试图将读取的10字节转换成字符，但是失败了
>>> f.close()
```

```
# 打开非文本文件，需要加上b，表示bytes
>>> f = open('/tmp/girl.jpg', 'rb')
>>> f.read(10)
b'\xff\xd8\xff\xe0\x00\x10JFIF'
>>> f.close()

# 文本文件也可以用rb的方式打开
(nsd1907) [root@room8pc16 day03]# echo 'hello达内' > /tmp/hi.txt
>>> f = open('/tmp/hi.txt', 'rb')
>>> f.read()
b'hello\xe8\xbe\xbe\xe5\x86\x85\n'
# 一个英文字符正好占一个字节，就用英文字符本身去显示；但是一汉字在utf8编码中占3字节，没有办法把一个字节表示成字符，所以就用16进制数表示
>>> f.close()

>>> s1 = b'\xe8\xbe\xbe'
>>> s1
b'\xe8\xbe\xbe'
>>> s1.decode()   # 将bytes类型转成str类型
'达'
>>> s2 = '达'
>>> s2.encode()   # 将str类型转成bytes类型
b'\xe8\xbe\xbe'
```

写入文件

```
>>> f = open('/tmp/hi.txt', 'w')  # 将会清空或创建文件
>>> f.write('hello world.\n')   # 写入13字节数据
13
(nsd1907) [root@room8pc16 day03]# cat /tmp/hi.txt    # 空的
>>> f.flush()  # 将缓存数据立即同步到磁盘
(nsd1907) [root@room8pc16 day03]# cat /tmp/hi.txt
hello world.
>>> f.writelines(['2nd line.\n', '3rd line.\n'])
>>> f.close()
(nsd1907) [root@room8pc16 day03]# cat /tmp/hi.txt
hello world.
2nd line.
3rd line.
```

with语句

通过with语句打开文件，当with语句结束时，文件自动关闭

```
>>> with open('/tmp/passwd') as f:
...   f.readline()
...
'root:x:0:0:root:/root:/bin/bash\n'
>>> f.readline()   # 报错，因为文件已经关闭
```

移动文件指针：了解

- tell函数总是返回文件指针距离(偏移)文件开头的字节数
- seek用于移动文件指针，参数中的第一个数字是偏移量，第二个数字是相对位置

```
>>> f = open('/tmp/passwd', 'rb')
>>> f.tell()
0
>>> f.seek(7, 0)  # 文件指针移动到距离开头7个字节处
7
>>> f.read(4)   # 读取4字节
b'0:0:'
>>> f.seek(-5, 2)   # 文件指针移动到文件倒数5个字节处
>>> f.read()
b'bash\n'
>>> f.close()
```

函数

- 函数使用def定义
- 调用函数务必使用()
- 函数一般都需要有一个返回值
    - 返回值使用return关键字指定
    - 没有return默认返回None

```
>>> def add():
...   result = 10 + 20
...
>>> a = add()
>>> print(a)
None

>>> def add2():
...   a = 10
...   b = 20
...   result1 = 10 + 20
...   result2 = 10 * 20
...   return 'hello world'
...
>>> b = add2()
>>> print(b)
hello world

>>> def add3():
...   a = 10
...   b = 20
...   return a + b
...
>>> c = add3()
>>> print(c)
30
```

位置参数

- 在python中，位置参数保存在sys模块的argv列表中

```
# vim position.py
import sys

print(sys.argv)
(nsd1907) [root@room8pc16 day03]# python position.py hao 123
['position.py', 'hao', '123']
```

默认参数

为函数的参数提供了默认值

```
>>> def pstar(n=30):
...   print('*' * n)
...
>>> pstar()
******************************
>>> pstar(40)
****************************************
```

模块

- 一个以.py结尾的python文件就是一个模块
- 文件代码的物理组织形式，模块是逻辑形式
- 将文件名的.py扩展名去除，重到的就是模块名
- 模块名也是一个名称，它也要满足相关的约定

```
# vim star.py
hi = 'Hello World!'

def pstar(n=30):
    print('*' * n)

# python
>>> import star
>>> star.<tab><tab>
star.hi      star.pstar(
>>> star.hi
'Hello World!'
>>> star.pstar()
******************************
>>> star.pstar(50)
**************************************************
```

模块特性

- 每个模块都有一个特殊的变量：__name__
- __name__的值是__main__或模块名
    - 如果文件作为程序直接运行，名为__main__
    - 如果文件作为模块被导入，名为模块名

```
(nsd1907) [root@room8pc16 day03]# echo 'print(__name__)' > foo.py
(nsd1907) [root@room8pc16 day03]# echo 'import foo' > bar.py
(nsd1907) [root@room8pc16 day03]# cat foo.py
print(__name__)
(nsd1907) [root@room8pc16 day03]# cat bar.py
import foo
(nsd1907) [root@room8pc16 day03]# python foo.py
__main__
(nsd1907) [root@room8pc16 day03]# python bar.py
foo
```