一、re模块

**1、核心函数和方法**

1）match、search、group函数

>>> import re

>>> m = re.search('f..', 'food') #可以匹配在中间的字符

>>> n = re.match('f..', 'seafood') #从字符串开头匹配

>>> print(m, n, sep='\n') #匹配到返回匹配对象，否则返回None

<\_sre.SRE\_Match object; span=(0, 3), match='foo'>

None

>>> m.group() #获得匹配的内容

'foo'

>>> m.group(n) #正则表达式使用了组合()，返回对应组合匹配内容

2）findall、finditer函数

>>> m = re.findall('foo', 'seafood is food')

#返回匹配列表，若使用组合()，匹配到时只返回括号内容

>>> n = re.finditer('foo', 'seafood is food') #返回序列对象

>>> print(m)

['foo', 'foo']

>>> for item in n:

... print(item.group())

foo

foo

1. compile函数

#匹配量大的时候，对正则表达式模式进行编译，返回一个正则对象(提高效率)

>>> patt = re.compile('f..')

>>> m = patt.search('seafood')

>>> m.group()

'foo'

>>> patt.findall('seafood is food')

['foo', 'foo']

1. split方法

>>> mylist = re.split('\.|-', 'hello-world.data')

#以.或-分割字符，返回一个列表

>>> print(mylist)

['hello', 'world', 'data']

5）sub方法

#把字符串所有匹配正则表达式的地方替换成新的字符串

>>> m = re.sub('X', 'Mr.Smith', 'hi,X. Dear X')

>>> print(m)

hi,Mr.Smith. Dear Mr.Smith

**2、案例：分析apache访问日志**

1）统计每个客户端访问apache服务器的次数

2）将统计信息通过字典的方式显示出来

3）分别统计客户端是Firefox和MSIE的访问次数

4）分别使用函数式编程和面向对象编程的方式实现

方法1：使用字典的方式记录次数

import re

def count\_patt(fname, patt):

patt\_dict = {}

cpatt = re.compile(patt)

with open(fname) as fobj:

for line in fobj:

m = cpatt.search(line)

if m:

key = m.group()

patt\_dict[key] = patt\_dict.get(key, 0) + 1

return patt\_dict

if \_\_name\_\_ == '\_\_main\_\_':

fname = 'access\_log'

ip = '^(\d{1,3}\.){3}\d{1,3}'

br = 'Chrome|Firefox|MSIE'

print(count\_patt(fname, ip))

print(count\_patt(fname, br))

方法2：使用OOP编程结合Counter模块按顺序排列

import re

from collections import Counter

class CountPatt:

def \_\_init\_\_(self, fname):

self.fname = fname

def count\_patt(self, patt):

cpatt = re.compile(patt)

patt\_counter = Counter() #创建Counter对象

with open(self.fname) as fobj:

for line in fobj:

m = cpatt.search(line)

if m:

patt\_counter.update([m.group()])

#添加列表数据，次数+1

return patt\_counter

if \_\_name\_\_ == '\_\_main\_\_':

fname = 'access\_log'

ip = '^(\d{1,3}\.){3}\d{1,3}'

br = 'Chrome|Firefox|MSIE'

cp = CountPatt(fname)

print(cp.count\_patt(ip))

print(cp.count\_patt(br))

二、socket模块

**1、C/S架构**

客户端/服务器架构：由服务器硬件及软件提供客户所需的“服务”，客户机使用客户端软件进行访问

**2、socket函数与方法**

**1）关于bytes类型数据**

#注意：所有网络通讯接收的内容都是Bytes类型，'\r\n'结束

>>> hi = 'hello world!'

>>> hello = hi.encode() #将string转换成bytes类型

>>> print(type(hello))

<class 'bytes'>

>>> hi = hello.decode() #将bytes转换成string类型

>>> hello = b'你好' #报错，中文字符不方便直接转为bytes

>>> hi = '你好'

>>> hello = hi.encode()

**2）创建TCP服务器**

import socket

host = '' #空串表示0.0.0.0

port = 1234

addr = (host, port)

s = socket.socket() #1、创建服务器套接字(默认TCP)

s.setsockopt(socket.SOL\_SOCKET, socket.SO\_REUSEADDR, 1)

#设置选项，套接字随时可用，否则系统默认保留60s

s.bind(addr) #2、绑定地址到套接字

s.listen(1) #3、启动监听(1表示允许1个客户端)

while True:

try:

cli\_sock, cli\_addr = s.accept()

#4、等待客户端连接，连接后返回客户端的套接字和地址

except KeyboardInterrupt:

break

while True:

data = cli\_sock.recv(1024) #5、与客户端通信(最多接收1024)

if data.strip() == b'quit':

break

print(data.decode(), end='')

send\_data = input('> ') + '\r\n'

cli\_sock.send(send\_data.encode()) #5、向客户端发送数据

cli\_sock.close()

s.close() #6、关闭套接字

**3）创建TCP客户端**

import socket

sip = '176.121.202.92'

sport = 1234

addr = (sip, sport)

cs = socket.socket() #1、创建客户端套接字

cs.connect(addr) #2、连接服务器

while True:

data = input('> ') + '\r\n'

cs.send(data.encode()) #3、与服务器通信(发送)

if data.strip() == 'quit':

break

recv\_data = cs.recv(1024) #3、与服务器通信(接收)

print(recv\_data.decode(), end='')

cs.close() #4、关闭客户端套接字

**4）创建UDP服务器**

import socket

import time

host = ''

port = 1234

addr = (host, port)

s = socket.socket(type=socket.SOCK\_DGRAM)

#1、创建服务器套接字(需要声明为UDP协议)

s.setsockopt(socket.SOL\_SOCKET, socket.SO\_REUSEADDR, 1)

#设置选项，套接字随时可用，否则系统默认保留60s

s.bind(addr)

#2、绑定服务器套接字

while True:

try:

data, cli\_addr = s.recvfrom(1024) #3、接收数据(返回数据与客地址)

except KeyboardInterrupt:

break

print(data.decode(), end='')

data = "[%s] %s" % (time.strftime('%H:%M:%S'), data.decode())

s.sendto(data.encode(), cli\_addr) #3、发送数据(数据, (host, port))

s.close() #4、关闭套接字

**5）创建UDP客户端**

import socket

sip = '176.121.202.92'

sport = 1234

addr = (sip, sport)

cs = socket.socket(type=socket.SOCK\_DGRAM)

while True:

data = input('> ') + '\r\n'

if data.strip() == 'quit':

break

cs.sendto(data.encode(), addr)

recv\_info = cs.recvfrom(1024)

recv\_data = recv\_info[0]

print(recv\_data.decode(), end='')

cs.close()