Python语言程序设计
Design and Programming of The Python Language

主讲教师：张小东
联系方式：z_xiaodong7134@163.com
答疑地点：宋健研究院514

第5章 网络程序设计

主要内容

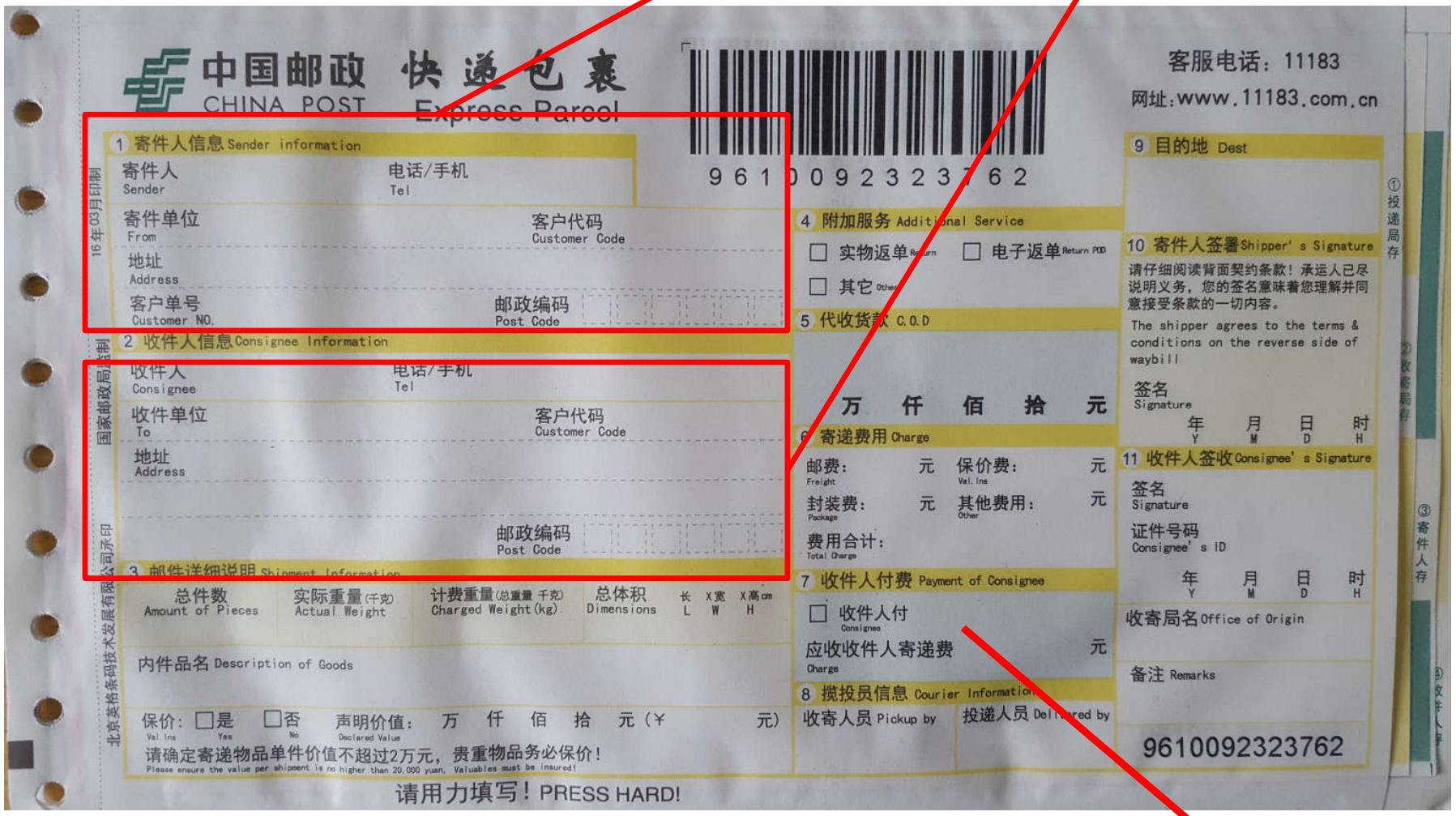
- 网络通信模型
- **Socket**编程
- **Internet**编程
- 应用实例

===网络通信模型===

◆ 问题的引入

寄件人信息

收件人信息



包裹内容及其它信息

===网络通信模型===

◆ 问题的引入

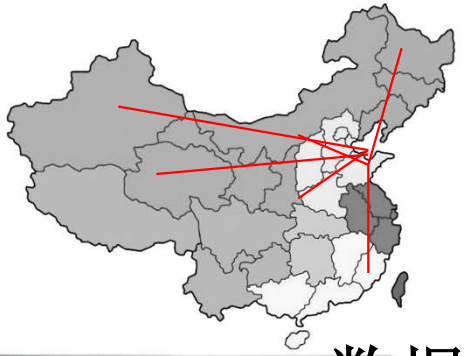
应用

收件人信息

信件

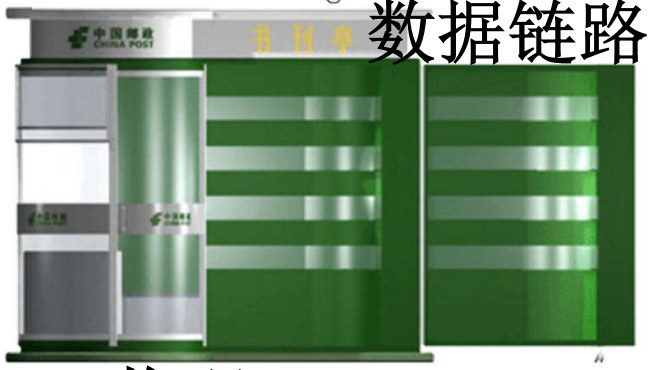
寄件人信息

传输



网络

数据链路



物理

物理

数据链路

网络

Tom

应用

传输

===网络通信模型===

◆ 计算机网络

将地理位置不同且具有独立功能的多台计算机及其外部设备通过通讯线路连接起来，并在网络操作系统、网络管理软件及网络通讯协议的管理和协调下，实现**资源共享和信息传递**的计算机系统。

◆ OSI参考模型

===网络通信模型===

◆ PDU在OSI参考模型中的特定名称

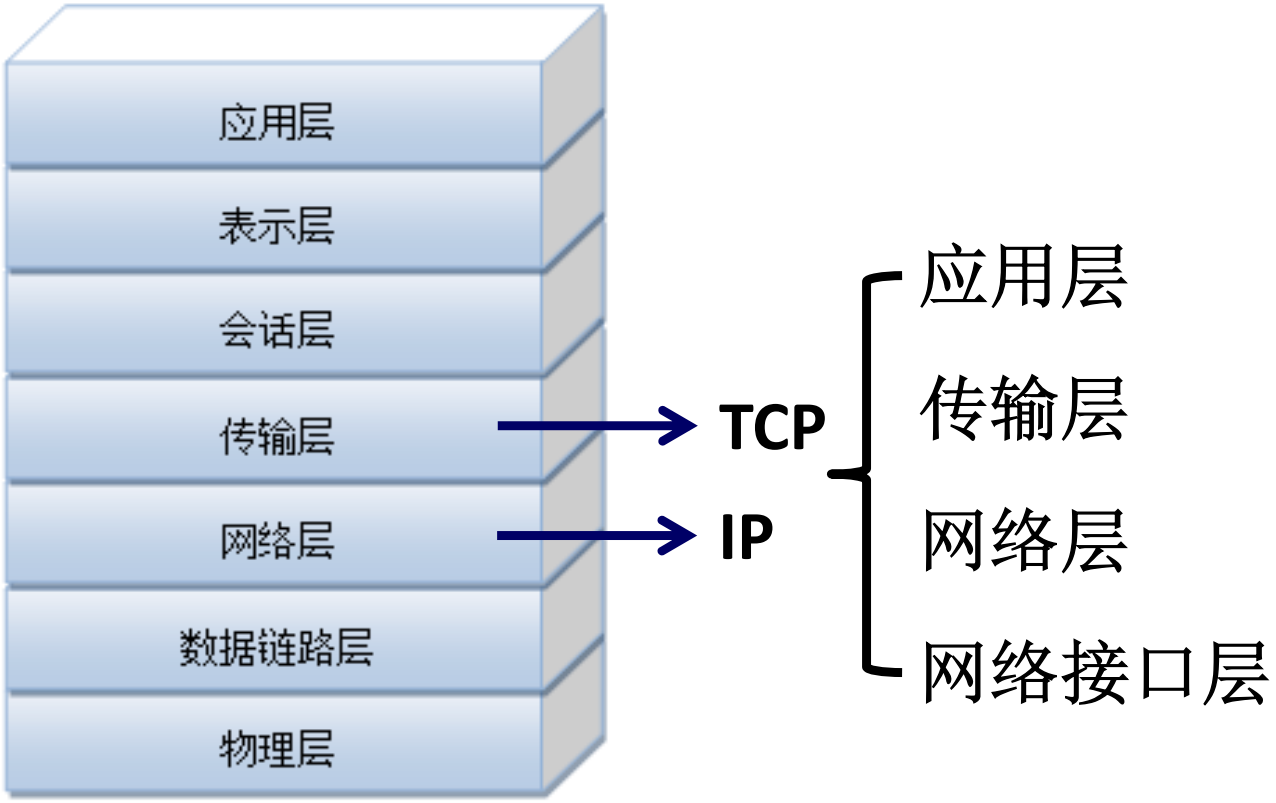
➢ PDU(Protocol Data Unit) 对等协议之间交换的信息单元统称为协议数据单元

OSI参考模型中的层次	PDU的特定名称
传输层	数据段（Segment）
网络层	数据包（Packet）
数据链路层	数据帧（Frame）
物理层	比特（Bit）

===网络通信模型===

◆ TCP/IP协议簇体系结构

➢ TCP/IP是Internet的基础网络通信协议



===网络通信模型===

◆ TCP/IP协议簇体系结构

➢ TCP/IP是Internet的基础网络通信协议

OSI 参考模型		TCP/IP 协议簇	
应用层		应用层	FTP、Telnet SMTP、SNMP、NFS
表示层			
会话层			
传输层		传输层	TCP、UDP
网络层		网络层	IP、ICMP 、ARP、RARP
数据链路层		网络接口层	Ethernet 802.3、Token Ring 802.5、X.25、Frame reley、 HDLC、PPP
物理层		未定义	

第5章 网络编程

主要内容

- 网络通信模型
- **Socket**编程
- **Interne**编程
- 应用实例

◆ Socket基本概念

【定义】

计算机之间进行网络通信的一套程序接口,网络编程的标准

【出版单位】

Berkeley

【作用】

Socket对象是网络通信的基础，相当于一个管道连接了发送端和接收端

【基础协议】

TCP和UDP

===网络通信模型===

➢ 基于UDP协议的网络编程

◆基础函数介绍

（1）socket()：创建一个流式套接字，返回套接字号。

socket(family,type[,protocol])

family: 地址家族, 取值为AF_INET(IPv4)、 AF_INET6(IPv6)。

type: 套接字类型，取值为SOCK_STREAM(TCP)或SOCK_DGRAM(UDP)。

protocol:与特定的地址家族相关的协议。默认值为0。

（2）bind(): 将套接字绑定到一个已知的地址及端口号

bind(address)

（3）sendto(): 发送数据。

sendto(data,(addr, port))

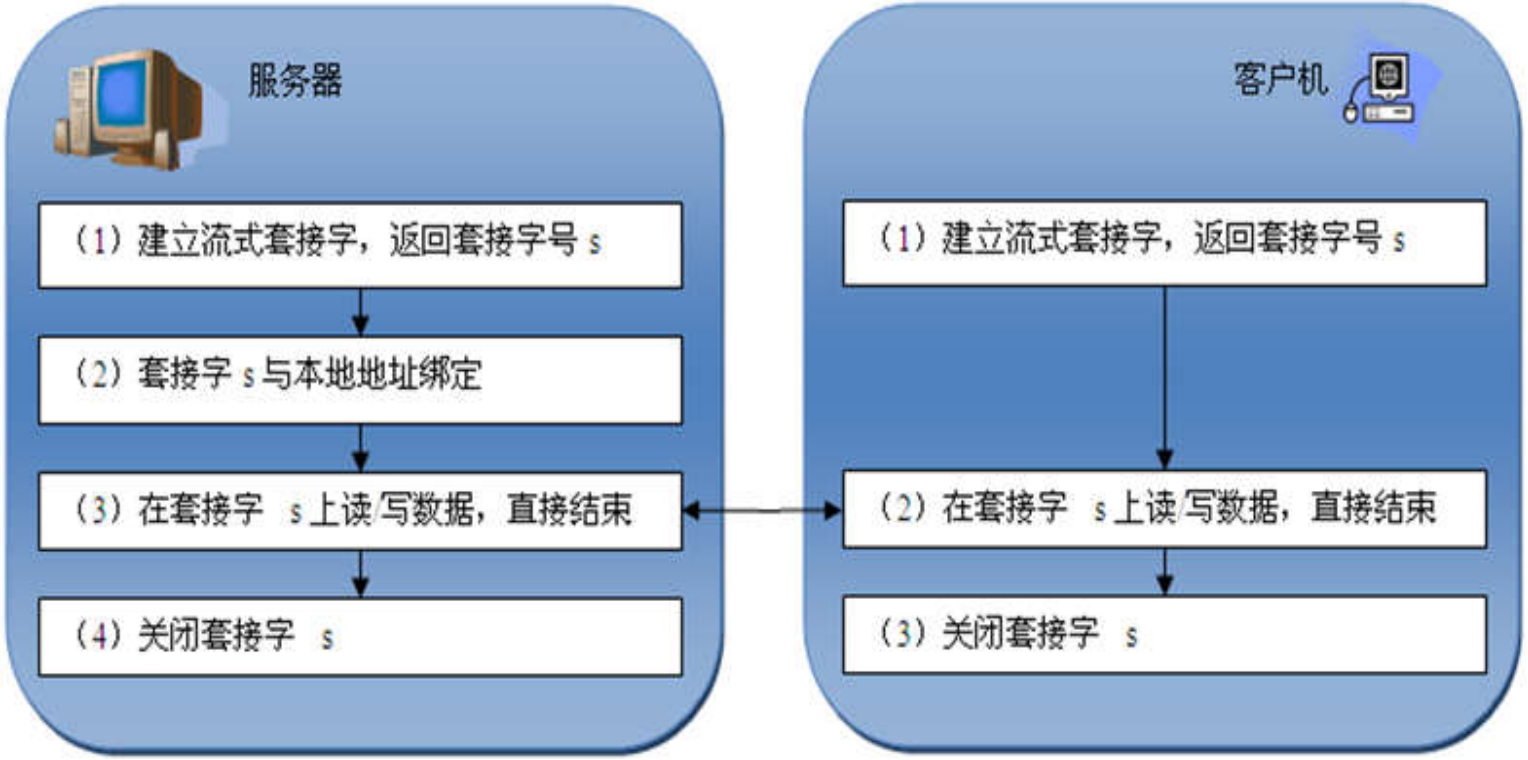
（4）recvfrom()：接收数据。

data, addr = s.recvfrom(bufsize)

=== Socket编程===

➢ 基于UDP协议的网络编程

【例5-1】编写一个基于UDP通讯程序，实现两台机器之间的信息通讯。



➢ 基于UDP协议的网络编程

```
import socket
s=socket.socket(socket.AF_INET,
          socket.SOCK_DGRAM)
s.bind(('127.0.0.1',5005))
while True:
    data,addr=s.recvfrom(1024)
    if not data:
        print('client has exited!')
        break
    print('receoved:',data,
              ' from ',addr)
s.close()
```

```
import socket
s=socket.socket(socket.AF_INET,
                socket.SOCK_DGRAM)
port=5005
host='127.0.0.1'
while True:
    msg=input()
    if not msg:
        break
    s.sendto(msg.encode("utf-8"),
                                  (host,port))
s.close()
```

=== Socket编程===

```
from socket import *
s=socket(AF_INET,SOCK_DGRAM)
s.bind(('',5005))
try:
    while True:
        data,ipaddr=s.recvfrom(1024)
        if not data or data.decode()=='-1':
            print('break')
            break
        print(data.decode())  #解码
except Exception as e:
    print("发生异常，终止服务！",e) #捕获异常
    s.close()#关闭socket对象
s.close()
```

◆ 基于TCP协议的网络编程

➢ 基础函数介绍

【服务器端】

（1）s.listen(backlog)：开始监听TCP连接,backlog为网络最大连接数。

（2）s.accept():接受TCP连接并返回(conn, address)

【 客户端】

s.connect(address): 连接到address处。Address=(host, port)

【公共】

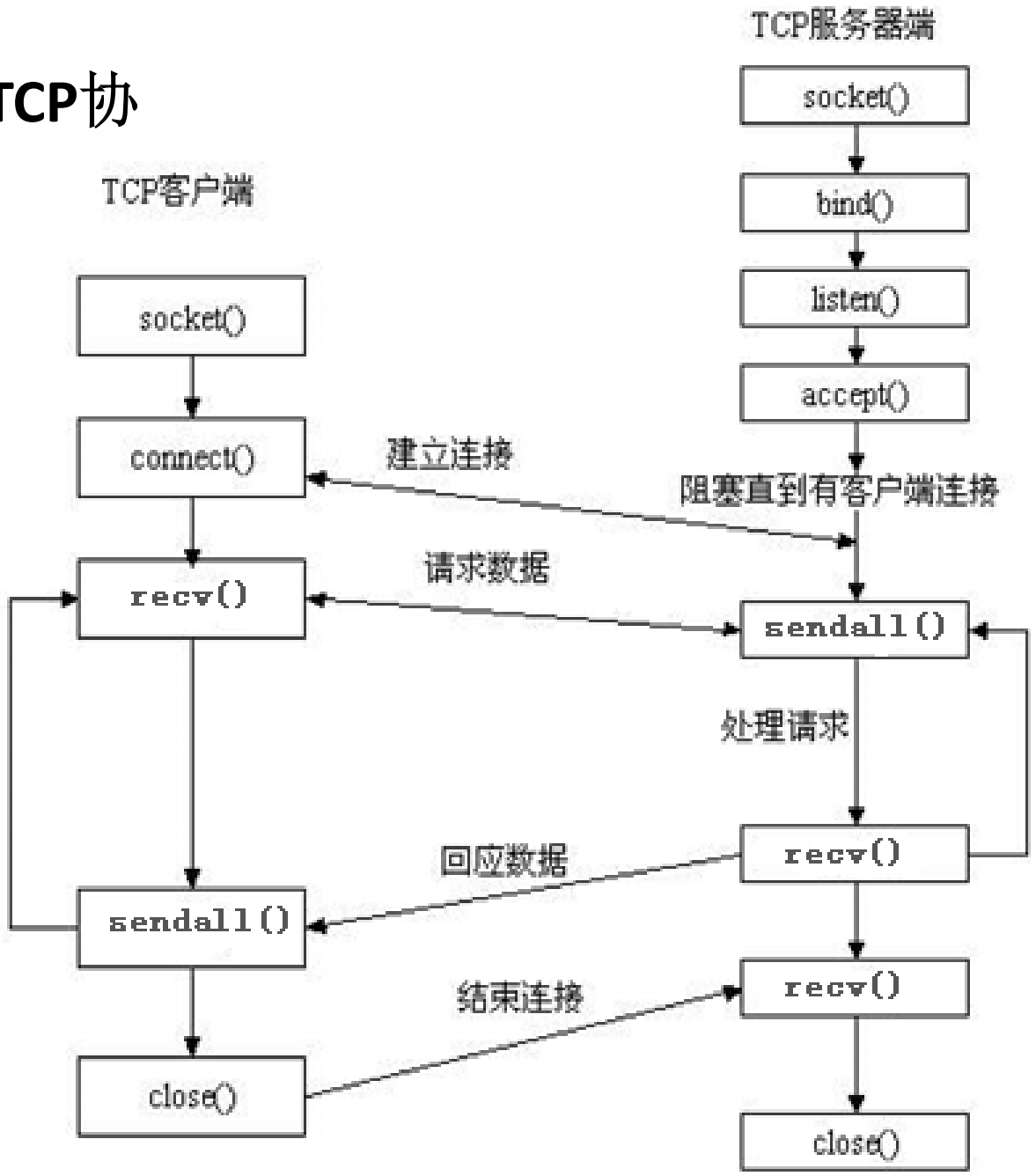
（1）s.recv(bufsize[,flag]):接收TCP数据，以字符串形式返回，bufsize表示接收的最大数据量。

（2）s.sendall(bytes[,flag]):完整发送TCP数据。

（3）s.settimeout(timeout):返回当前超时期的值，单位是秒。

===网络通信模型===

【例5-2】建立基于TCP协议的网络通信程序



===网络通信模型===

【例5-2】建立基于TCP协议的网络通信程序

```
from socket import *
from time import ctime
host=''
port=4700
Buf=1024
addr=(host,port)
s=socket(AF_INET,SOCK_STREAM,0)
s.bind(addr)
s.listen(20)
```

```
while True:
    print('等待客户连接...\r\n')
    cs,caddr=s.accept()
    print('...连接来自于:',caddr)
    data='欢迎你的到来！ \r\n'
    cs.sendall(bytes(data,'utf-8'))
    data=cs.recv(Buf).decode('utf-8')
    if not data:
        break
    data='[%s]%s\r\n'%(ctime(),data)
    cs.sendall(bytes(data,'utf-8'))
    print(data)
    cs.close()
s.close
```

=== Socket编程===

【例5-2】建立基于TCP协议的网络通信程序

```
while True:
    ......
    cs,caddr=s.accept()
    ......
    cs.sendall(bytes(data,'utf-8'))
    data=cs.recv(Buf).decode('utf-8')
    cs.sendall(bytes(data,'utf-8'))
    ......
    cs.close()
```

循环接收请求

同步

完成一次交互即断开链接

```
from socket import *
buf=1024
addr=('127.0.0.1', 4700)
cs=socket(AF_INET,SOCK_STREAM,0)
cs.connect(addr)
data=cs.recv(buf).decode('utf-8')
if data:
    print(data)
while True:
    data=input()
    if data:
        cs.sendall(bytes(data,'utf-8'))
        data=cs.recv(buf).decode('utf-8')
        if data:
            print(data)
cs.close()
```

思考

如何提高通信效率？

➢ 资源调度之CPU调度

1. 进程与线程

(1) 进程：资源的分配和调度的一个独立单元
(2) 线程：CPU调度的基本单元
(3) 进程与线程包容关系及资源分配。
(4) 进程与线程的创建与销毁
(4) 线程是轻量级的进程
(5) 线程中执行中的同步和互斥
(6) 线程的管理信息
(7) 进程的管理信息

2. 进程和线程对网络通信影响

【例5-3】基于TCP协议的进程级网络通信程序

```
from socket import *
from multiprocessing import Process
def talk(conn, addr):
    while True:
        try:
            cmsg = conn.recv(1024)
            if not cmsg: break
            conn.send(cmsg.upper())
        except Exception:
            break
    conn.close()
```

```
if __name__ == '__main__':
    print("主进程开始.")
    server = socket()
    ip_port = ("127.0.0.1", 8080)
    server.bind(ip_port)
    server.listen(5)
    while True:
        conn, caddr = server.accept()
        print("连接来自于： ", caddr)
        p = Process(target=talk,
                args=(conn,caddr))
        p.start()
    server.close()
```

=== Socket编程===

【例5-3】基于TCP协议的进程级网络通信程序

```
from socket import *
client = socket()
ip_port = ("127.0.0.1", 8080)
client.connect(ip_port)
while 1:
    inp = input(">>>:").strip()
    if not inp: continue
    client.send(inp.encode("utf-8"))
    smsg = client.recv(1024)
    print("来自服务端的消息:",smsg.decode("utf-8"))

client.close()
```

=== Socket编程===

2. 进程和线程对网络通信影响

【例5-3】基于TCP协议的线程级网络通信程序

```
import socket
import threading
socket_list = []
s = socket.socket()
s.bind(('127.0.0.1', 30000))
s.listen()
def read_client(s):
    try:
        # 接收客户端的数据
        return s.recv(2048).decode('utf-8')
    except:
        print(str(addr) + ' Left!')
        socket_list.remove(s)
```

```
def socket_target(s):
    try:
        while True:
            content = read_client(s)
            if content is None:
                break
            else:
                print(content)
                for client in socket_list:
                    client.send((str(addr)+ ' say: ‘
                        +content).encode('utf-8'))
    except:
        print('Error!')
```

=== Socket编程===

【例5-3】基于TCP协议的线程级网络通信程序

```
while True:
    conn, addr = s.accept()
    socket_list.append(conn)
    print(str(addr) + ' Joined!')
    threading.Thread(target=socket_target, args=(conn,)).start()
```

```
import socket
import threading
s = socket.socket()
s.connect(('127.0.0.1', 30000))
def read_server(s):
    while True:
        content = s.recv(2048).decode('utf-8')
        print(content)
threading.Thread(target=read_server, args=(s,)).start()
while True:
    line = input('')
    if line == 'exit':
        break
    s.send(line.encode('utf-8'))
```

第5章 网络程序设计

主要内容

- 网络通信模型
- **Socket**编程
- **Internet**编程
- 应用实例

➢ Web资源访问

◆ 网络协议

超文本传输协议(HTTP)

◆ 数据访问方式

通过统一资源定位符（URL）请求访问资源

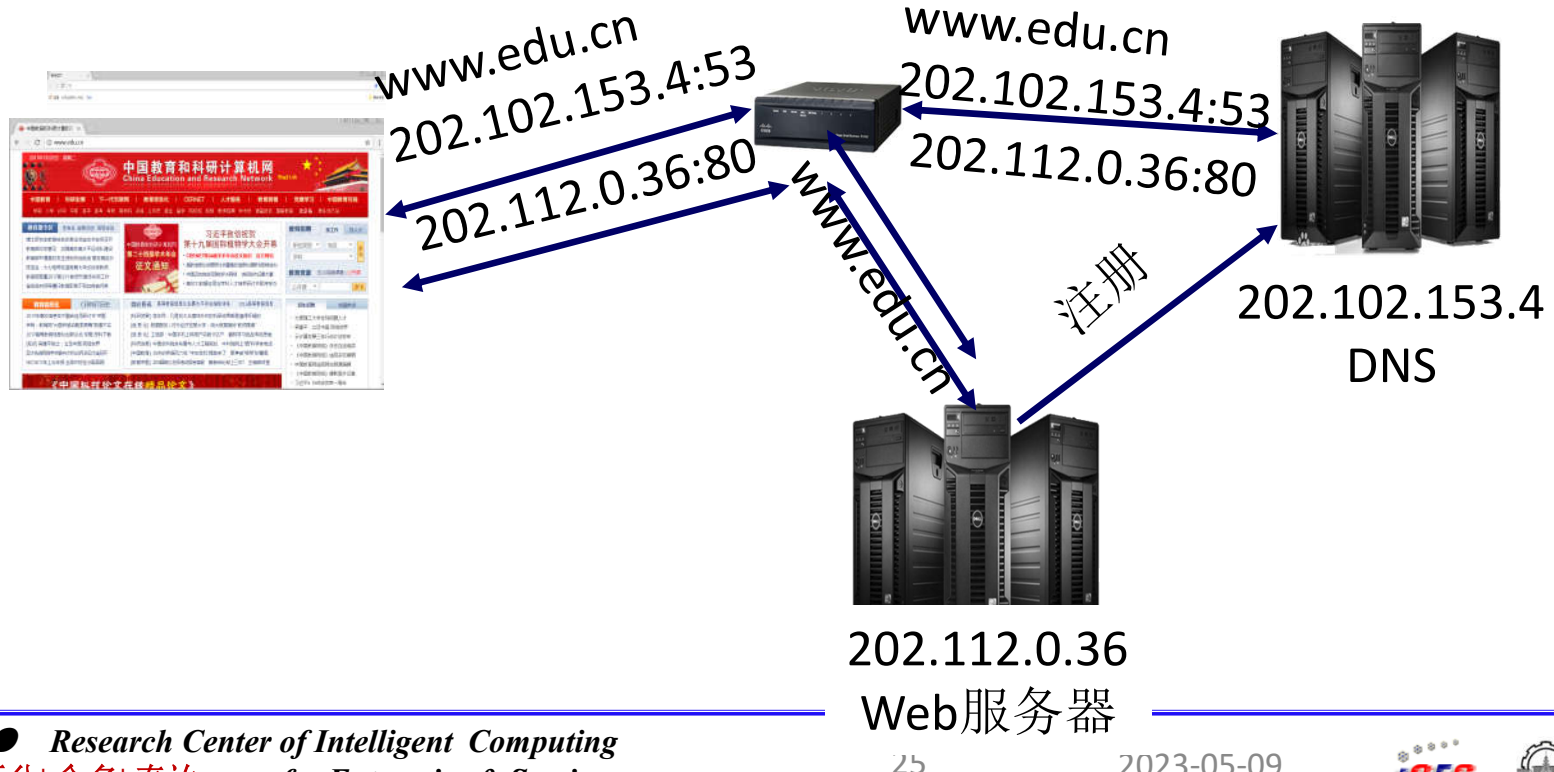
URL:http://www.edu.cn == IP:Port

◆ 主要内容

网页—数据展示

◆ 客户端软件

浏览器



➢ Web资源访问

1. urllib：收集了几个处理URL的模块包

request：打开和读取URL

parse：解析URL

error： urllib.request引发的异常

robotpaser：解析robots.txt文件

（1）urllib.request模块

urlopen()：创建一个表示远程URL的类文件对象

urllib.request.urlopen(url[,data[,proxies]])

表示以post方式提交到url的参数

设置代理

- ✓ info()：返回一个httplib.HTTPMessage对象，表示远程服务器返回的头信息
- ✓ getcode(): 返回HTTP状态码。200，404
- ✓ geturl():返回请求的url

➢ Web资源访问

【例5-4】打开一个网站，显示其头部信息、访问状态、网址及网页上前30行信息。

```
from urllib.request import *
content=urlopen('http://www.edu.cn')
print("http header:",content.info())
print("http status:",content.getcode())
print("url:",content.geturl())
i=0
print('content:')
for line in content.readlines():
    print(line.decode('gb2312'))
    i=i+1
    if i>30:
        break
```

➢ Web资源访问

网址　　本地文件名

urlretrieve(url[,filename[,reporthook[,data]]])

作用：将url资源下载到本地并存成文件

```
from urllib.request import *
content=urlretrieve('http://www.edu.cn','c:\\zxd1.htm')
print(content)
```

Request实例：返回一个Request的对象

urllib.request.Request(key=value[,key=value,...])

（4）对url进行编码、解码的有用方法

urllib.quote(string[,safe]), urllib.unquote(string)

urllib.quote_plus(string[,safe]),

urllib.unquote_plus(string),

➢ Web资源访问

（2）向Web服务器传递参数

如http://search.jd.com/Search?keyword=平板电脑&enc=utf-8

```
import urllib.parse
import urllib.request
url='http://search.jd.com/Search'
values={'enc':'utf-8','keyword':'平板电脑'}
data=urllib.parse.urlencode(values)
kv = {'user-agent':'Mozilla/5.0'}
req = urllib.request.Request(url= url+'?'+data, headers=kv)
content=urllib.request.urlopen(req)
i=0
for line in content:
    print(line.decode('utf-8'))
    i=i+1
    if i>60:
        break
content=urllib.request.urlretrieve(url+'?'+data,'zxd2.htm')
```

➢ 邮件客户端编程

◆POP3(Post Office Protocol 3):负责从邮件服务器中检索电子邮件。支持多用户互联网邮件扩展，允许在电子邮件上附带二进制文件。因特网电子邮件的第一个离线协议标准。

◆POP3 类：poplib

POP3(host, post=POP3_PORT[,timeout]): 构造方法，连接host邮件服务器。POP3_PORT=110

user(username): 发送邮件的账号(登录用户名)

pass_(password): 发送用户密码

getwelcome(): 返回欢迎信息

stat():返回邮箱状态，结果为元组(邮件数量，邮件字节数)

list():返回邮件列表，结果是一个三元组

dele(i):设置邮件删除标志，调用quit()时从服务器删除

quit(): 注销，释放连接

➢ 邮件客户端编程

【例5-5】从mail.hit.edu.cn

```
import poplib
import getpass
user='zxd@hit.edu.cn'
pwd=getpass.getpass('密码:')
host='mail.hit.edu.cn'
M=poplib.POP3(host)
M.user(user)
M.pass_(pwd)
mboxstat=M.stat()
print("邮件数量",mboxstat[0])
mylist=M.list()
for i in range(3):
    print(mylist[i])
numMail=len(M.list()[0])
print("邮件数量",numMail)
k=75
mailx=M.retr(k)[1]
i=0
for line in mailx:
    print(line)
    i=i+1
    if i>=5:
        break
M.quit()
```

返回邮件信息

=== Interne编程===

➢ 邮件客户端编程

【例5-6】从接收mail升级

题目

发件人

收件人

附件

邮件内容

编码

文件存储

【例5-6】从mail.hit.edu.cn升级版

题目\发件人\收件人

```
import poplib
from email.parser import Parser
from email.header import decode_header
from email.utils import parseaddr
def decode_str(s):
    value,charset=decode_header(s)[0]
    if charset:
        value=value.decode(charset)
    return value
```

```
def print_info(msg):
    for header in ['From','To','Subject']:
        value=msg.get(header,'')
        if value:
            if header=='Subject':
                value=decode_str(value)
            else:
                hdr,addr=parseaddr(value)
                name=decode_str(addr)
                value=name+'<'+addr+'>'
            print(header+':'+value)
```

编码

```
def guess_charset(msg):
    charset=msg.get_charset()
    if charset is None:
        content_type=msg.get('Content-Type','').lower()
        pos=content_type.find('charset=')
        if pos>=0:
            charset=content_type[pos+8:].strip()
    return charset
```

【例5-6】从mail.hit.edu.cn升级版

```
i=0
for part in msg.walk():
    filename=part.get_filename()
    content_type=part.get_content_type()
    print('====',content_type,'====')
    charset=guess_charset(part)
    if filename:
        filename=decode_str(filename)
        data=part.get_payload(decode=True)
        if filename!=None or filename!='':
            print('Accessory:'+filename)
            savefile(filename,data,mypath)
    else:
        email_content_type=''
        content=''
        if content_type=='text/plain':
            email_content_type='text'
        elif content_type=='text/html':
            email_content_type='html'
        if charset:
            content=part.get_payload(decode=
                True).decode(charset)
        print(email_content_type+' '+content)
```

```
email='zxd@hit.edu.cn'
pwd=input("input password")
pop3_server='mail.hit.edu.cn'
mypath='D://pythonemail/'
server = poplib.POP3(pop3_server,110)
print(server.getwelcome())
server.user(email)
server.pass_(pwd)
print('Message:%s. Size:%s'%server.stat())
resp,mails,objects=server.list()
#index=len(mails)
resp,lines,octets=server.retr(76)
lists=[]
for e in lines:
    lists.append(e.decode())
msg_content='\r\n'.join(lists)
msg=Parser().parsestr(msg_content)
print_info(msg)
server.quit()
```

➢ 邮件客户端编程

◆ 使用SMTP协议发送邮件

SMTP(host=‘’,port=0,local_h=None[,timeout],src_addr=None),构造函数，其主要参数**host**是发送邮件的服务器

login(user, password):登录服务器

sendmail(from_addr, to_addr, msg, mail_option=[],
rept_option=[]): 发送邮件

send_message(msg, from_addr, to_addr): 以邮件对象的形式发送邮件。

quit():注销，释放连接

=== Interne编程===

【例5-7】编写一发送邮件的程序

```
import smtplib
import email
def prompt(prompt):
    return input(prompt).strip()
fromaddr=prompt("From:")
toaddrs=prompt("To:")
subject=prompt("Subject:")
print("Enter message, end with continusly enter:")
text=""
while True:
    try:
        line=input()
    except EOFError:
            break
    if len(line)==0:
        break
    text=text+line+"\r\n"
```

➢ 邮件客户端编程

【例5-7】编写一发送邮件的程序

```
msg=email.message_from_string(text)
msg['Subject']=subject
msg['From']=fromaddr
msg['To']=toaddrs
msg.set_charset('GB2312')
server=smtplib.SMTP('123.125.50.133')
user=fromaddr
pwd=input('请输入邮箱密码')
server.login(user,pwd)
server.send_message(msg,fromaddr,toaddrs)
print("Send successfully")
server.quit()
```

第5章 网络程序设计

主要内容

- 网络通信模型
- **Socket**编程
- **Internet**编程
- 应用实例

===应用实例===

➢ 服务器与客户端的交互计算

【例5-8】使用基于TCP的socket套接字模块编写一个网络通信程序，实现客户与服务器的交互计算。其中，客户端将三角形的三边长发给服务器，服务器计算出三角形的面积再把结果返回给客户端。

【问题分析】

（1）建立服务器socket对象，等待用户发送请求

（2）服务器接收到客户端数据，调用函数进行三角形面积计算，并把计算结果返回给客户端

（3）建立客户端socket对象，收集用户输入信息，发送给服务器端

===应用实例===

【例5-8】基于TCP的socket编写一个网络通信程序

```
from socket import *
from time import ctime
from math import *
def triangle(a,b,c):
    s=(a+b+c)/2
    return sqrt(s*(s-a)*(s-b)*(s-c))
addr=('',5005)
ss=socket(AF_INET,SOCK_STREAM,0)
ss.bind(addr)
ss.listen(20)
```

===应用实例===

【例5-7】基于TCP的socket编写一个网络通信程序

```
while True:
    print('等待客户连接...\r\n');
    cs,caddr=ss.accept()
    print('连接来自于：',caddr)
    data='欢迎你的到来！ \r\n'
    cs.sendall(bytes(data,'utf-8'))
    data=cs.recv(1024).decode('utf-8')
    if not data:
        break
    print('三角形三边长为:',data)
    sides=data.split(',')
    cs.sendall(bytes(''+str(triangle(float(sides[0]),
                    float(sides[1]),float(sides[2]))),'utf-8'))
    cs.close()
ss.close()
```

```
from socket import *
s=socket(AF_INET,SOCK_STREAM,0)
s.connect(('127.0.0.1',5005))
data=s.recv(1024).decode('utf-8')
if data:
    print(data)
a=input("请输入边长a:")
b=input("请输入边长b:")
c=input("请输入边长c:")
data=a+','+b+','+c
if data:
    s.sendall(bytes(data,'utf-8'))
    data=s.recv(1024).decode('utf-8')
    if data:
        print('三角形面积为：',data)
s.close()
```

===应用实例===

➢ 服务器与客户端的交互计算

【例5-9】使用urllib模块访问某一个网址并判断得到的网页存在多少个网页链接和图片链接。

【问题分析】

（1）使用input输入任意一个网址

（2）使用urlopen函数打网址，建立应答对象res

（3）使用res的read读取网页资源资源

（4）使用re模块的compile建立网页链接的正则表达式“a href=\”(.+?)\””的对象linkPattern.

（5）使用linkPattern的findall函数获得全部网页链接的列表links

（6）使用links的len函数获得网页链接的数量并显示。

（7）同样建立“imgsrc=\“(.+?)\”对象imagePattern并获得全部链接的images，显示数量

===应用实例===

【例5-9】使用urllib模块访问某一个网址并判断得到的网页存在多少个网页链接和图片链接。

```
import urllib.request
import re
url=input("input url:")
res=urllib.request.urlopen(url)
htm=res.read()
details=htm.decode('utf-8')#gb2312
linkPattern=re.compile("a href=\"(.+?)\"")
links=linkPattern.findall(details)
print("网页链接总数=",len(links))
imagePattern=re.compile("img src=\"(.+?)\"")
images=imagePattern.findall(details)
print("图片总数=",len(images))
```

```
import chardet
#用chardet进行内容分析
chardit1 = chardet.detect(htm)
print(chardit1['encoding'])
```

本章小结

- 网络通信模型
- **Socket**编程
- **Interne**编程
- 应用实例