

计算机网络编程

第15章 POP客户机程序设计

信息工程学院 方徽星

fanghuixing@hotmail.com

大纲

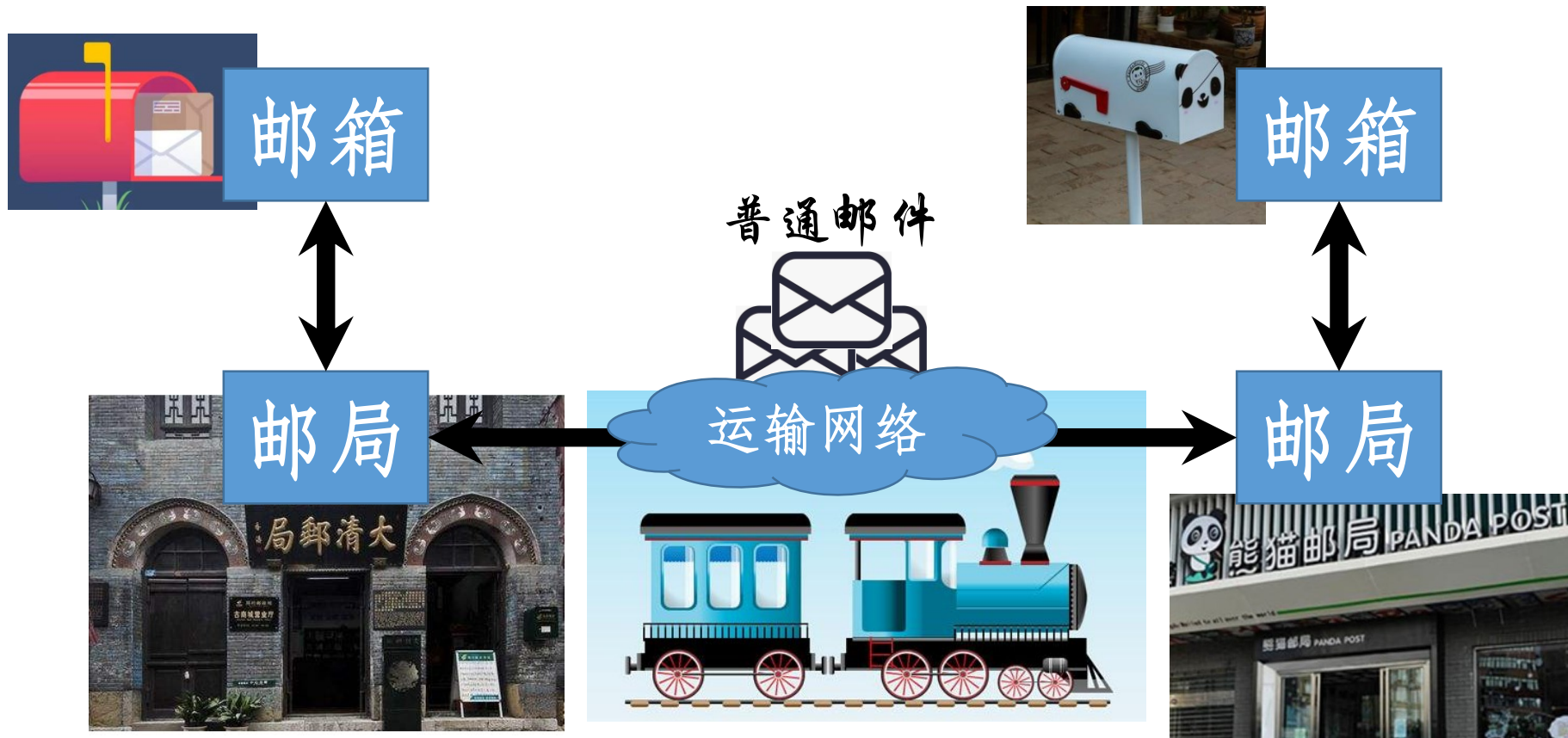
- 设计目的
- 相关知识
- 例题分析

1. 设计目的

- 通过POP客户机程序的设计
 - 了解电子邮件的基本概念与主要功能
 - 掌握应用层服务的设计思路与编程方法

2. 相关知识：电子邮件的基本概念

- 电子邮件服务又称E-mail服务，指用户通过Internet收发电子形式的邮件



2. 相关知识：电子邮件的基本概念

- 电子邮件服务又称E-mail服务，指用户通过Internet收发电子形式的邮件



邮箱



邮件服务器

电子邮件



计算机网络



邮箱



邮件服务器

2. 相关知识： 邮件服务的工作原理

- **电子邮件服务在传输层采用TCP协议**
 - **建立连接**
 - **传输数据**
 - **释放连接**
- **电子邮件服务在应用层采用多种协议**
 - **SMTP (Simple Mail Transfer Protocol)**
 - **POP (Post Office Protocol)**
 - **IMAP (Interactive Mail Access Protocol)**

2. 相关知识： 邮件服务的工作原理

- SMTP: 将邮件从客户机发送到邮件服务器

- POP

- IMAP



将邮件从邮件服务器
接收到客户机

通常采用POP协议

POP3: FRC1939

SMTP服务器使用熟知端口号25
POP3服务器使用的熟知端口号110

2. 相关知识： 邮件服务的工作原理

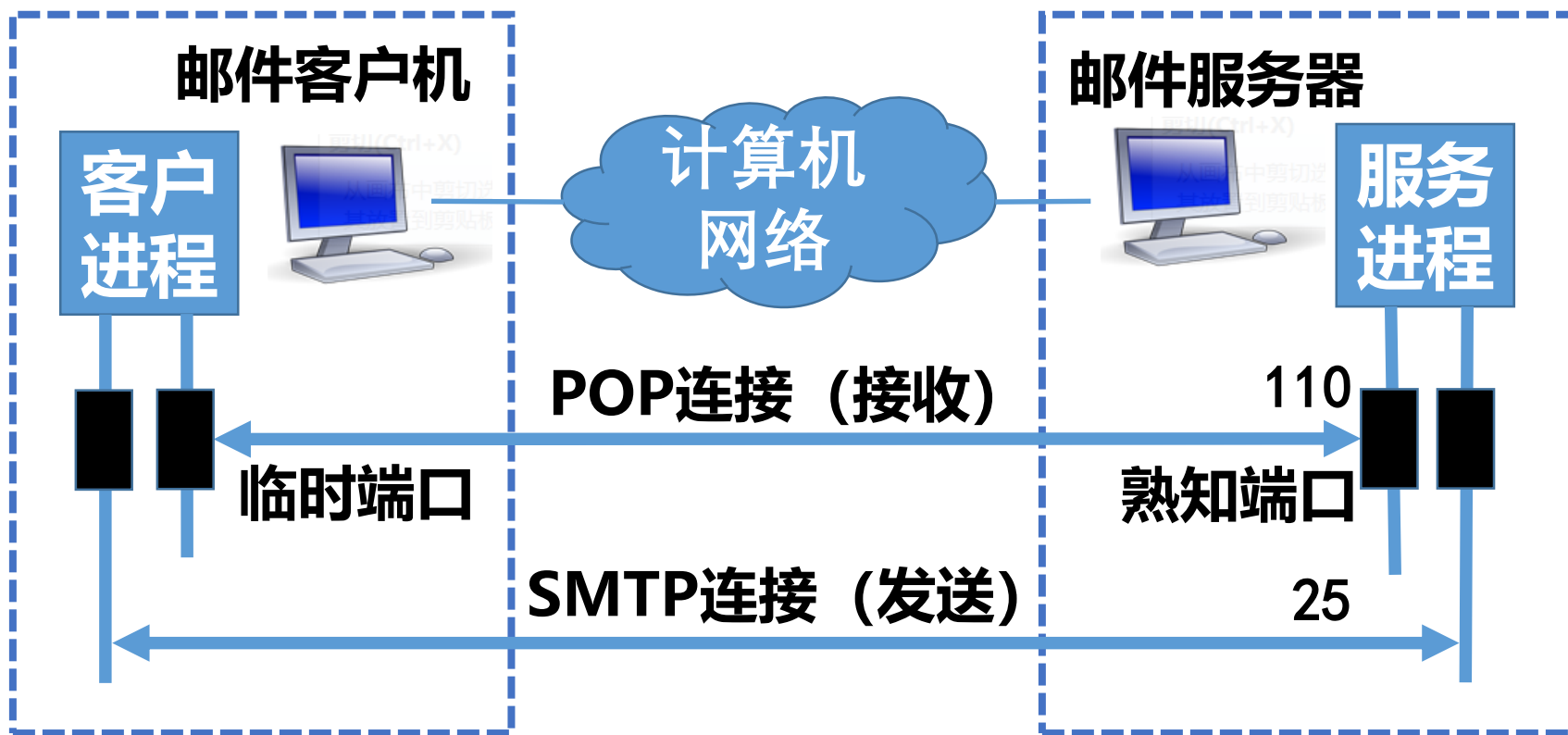
- 电子邮件服务采用客户机/服务器工作模式



- POP3协议工作模式
 - 删除模式：在读取邮件后删除邮件
 - 保留模式：在读取邮件后仍保留在邮箱中

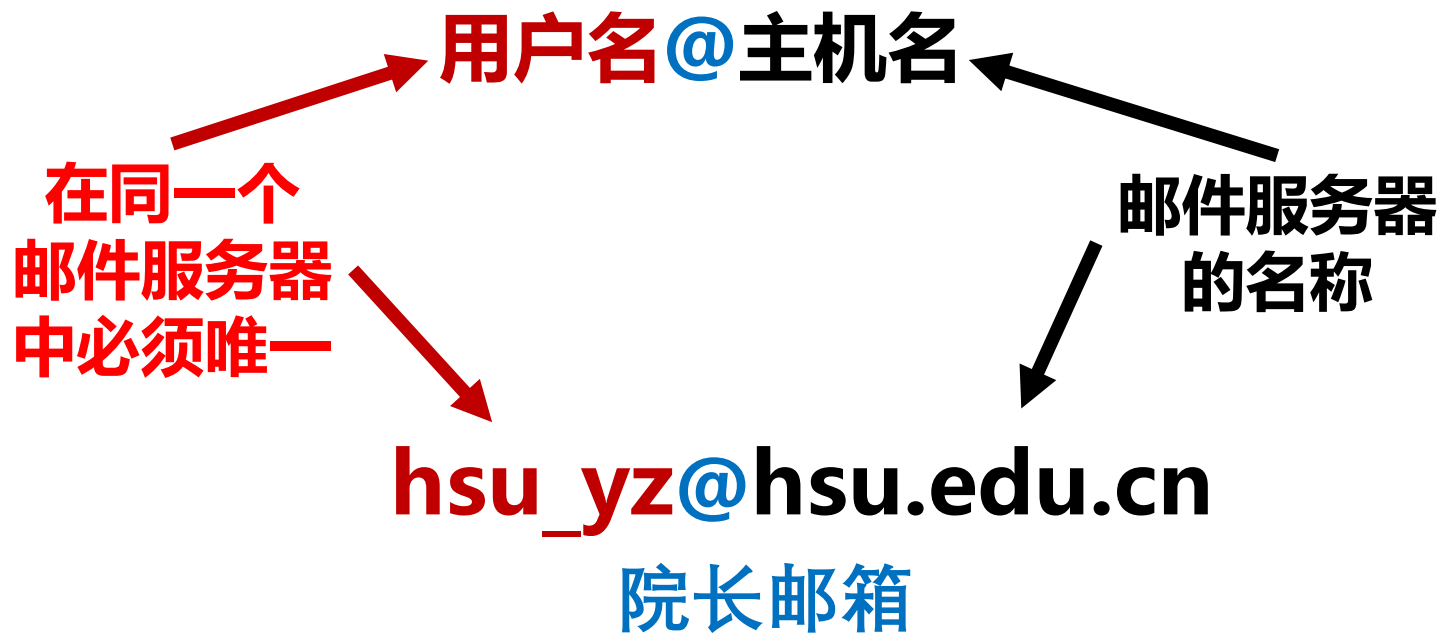
2. 相关知识：邮件服务的工作原理

- 电子邮件服务采用客户机/服务器工作模式

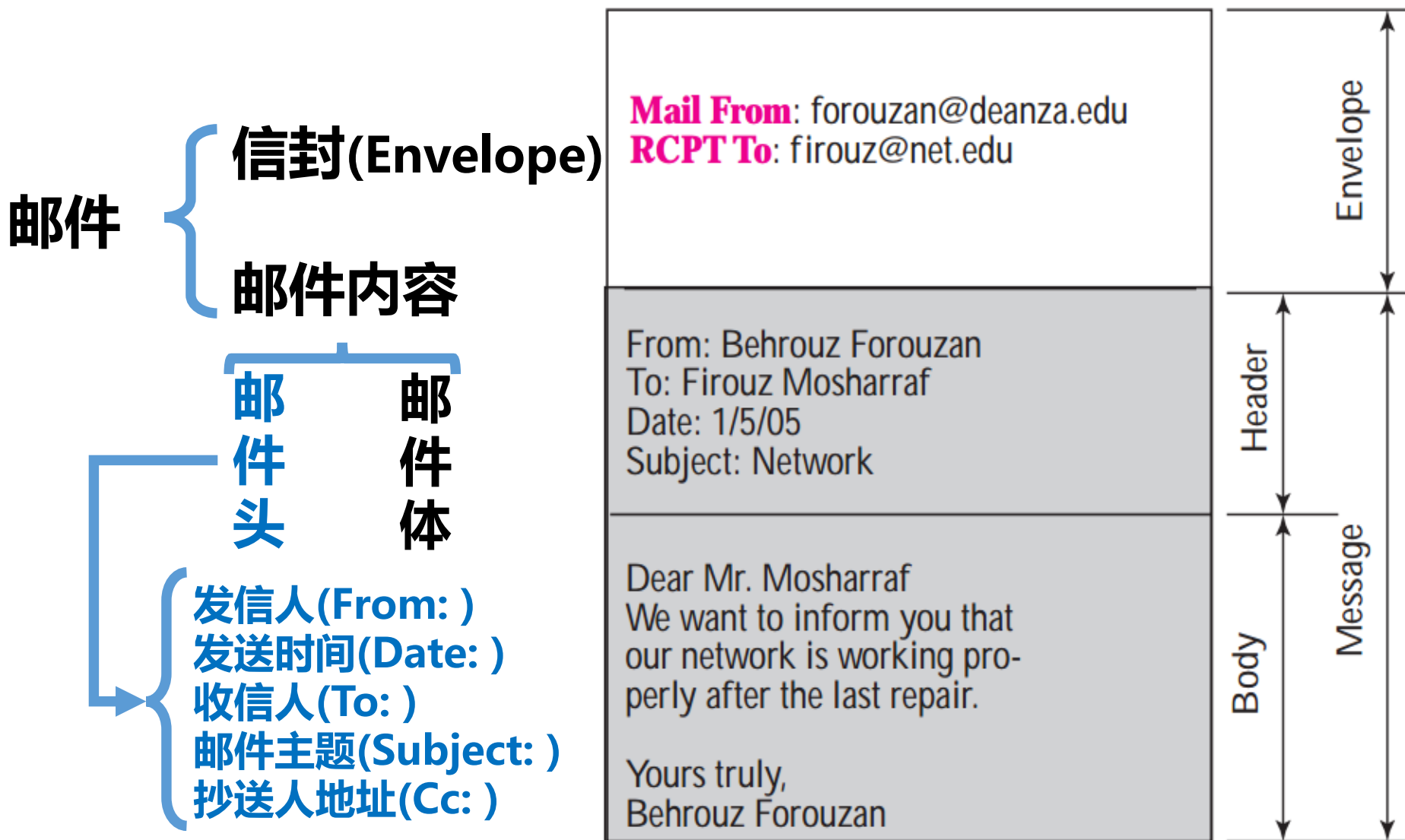


2. 相关知识：邮件地址与邮件格式

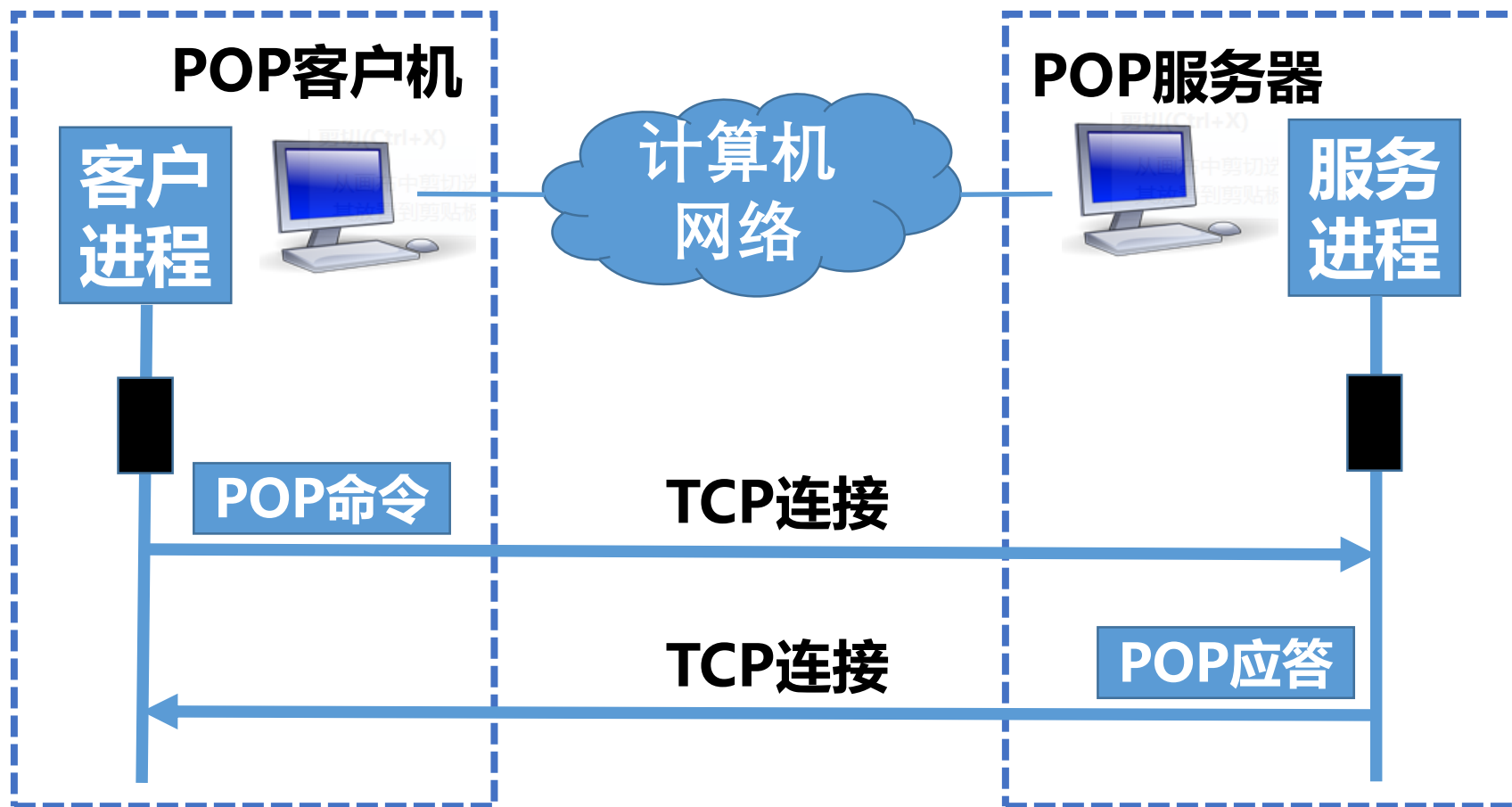
- Raymond Samuel Tomlinson最早提出电子邮件地址，使用@来隔开用户名与邮件服务器地址



2. 相关知识： 邮件地址与邮件格式



2. 相关知识：POP命令与应答



2. 相关知识：POP命令与应答

- POP命令



所有命令由一对回车与换行符表示结束

2. 相关知识：POP命令与应答

主要POP命令

命令名	命令格式	命令用途
USER	USER <username>	登录时需要的用户名
PASS	PASS <password>	登录时需要的密码
STAT	STAT	返回邮箱统计信息，包括邮件数、总字节数
LIST	LIST <msg#>	列出邮箱中的邮件信息 参数msg#是一个可选参数，表示邮件的序号

2. 相关知识： POP命令与应答

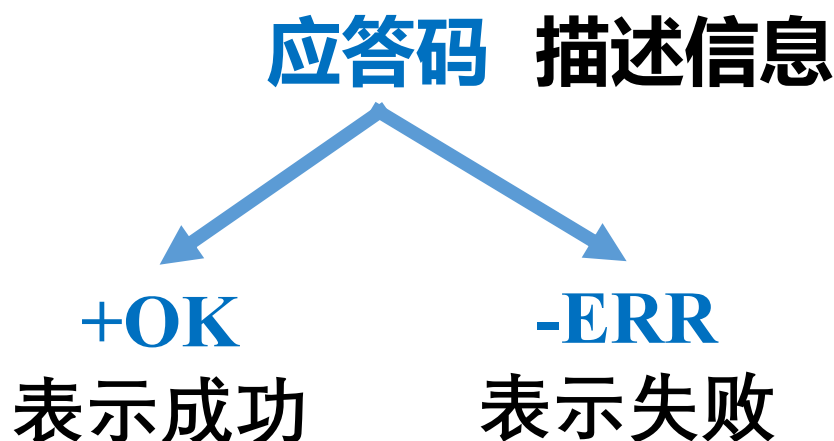
命令名	命令格式	命令用途
RETR	RETR <msg#>	获取某封邮件的内容，参数 msg#表示邮件的序号
DELE	DELE <msg#>	为指定邮件做删除标记，在QUIT时执行
REST	REST	清除所有邮件的删除标记
APOP	APOP <name,digest>	用于替代user和pass命令，它以MD5 数字摘要的形式向POP3邮件服务器提交帐户密码

2. 相关知识： POP命令与应答

命令名	命令格式	命令用途
TOP	TOP <msg#, n>	获取指定邮件的邮件头和邮件体中的前n行内容， 参数msg#表示邮件的序号
NOOP	NOOP	用于检测POP3客户端与POP3服务器的连接情况
QUIT	QUIT	POP3服务器将删除所有设置了删除标记的邮件， 并关闭与POP3客户端程序的网络连接

2. 相关知识：POP命令与应答

- POP应答



所有应答由一对回车与换行符来表示结束

3. 例题分析：设计要求

- **设计要求**

- **基于客户机/服务器工作模式，编写POP客户机程序向服务器发送命令**
- **将服务器返回的应答信息与数据显示在控制台上**
- **只需要实现USER、PASS、STAT、LIST与QUIT命令**

3. 例题分析：设计要求

- 具体要求
 - 命令行程序

PopClient server_addr



POP服务器域名
或IP地址

3. 例题分析：设计要求

- 具体要求
 - 要求将POP服务器的状态显示在控制台上

```
POP>Control Connection...  
应答信息...  
POP>USER: xxxxxxx  
应答信息...  
POP>PASS: xxxxxxxx  
应答信息...  
POP>STAT  
应答信息...  
POP<LIST  
应答信息...  
POP>QUIT  
应答信息...
```

3. 例题分析：关键问题

- 建立数据连接

```
//创建流式套接字
sock = socket(AF_INET, SOCK_STREAM, 0);
//判断IP地址或POP服务器名
if(argv[1]不是IP地址)
    hostent * pHostent = gethostbyname(argv[1]);
//填充服务器的套接字地址
sockaddr_in serveraddr;
serveraddr.sin_family = AF_INET;
serveraddr.sin_port = htons(110); //POP服务器端口
serveraddr.sin_addr.S_un.S_addr = IP地址;
```

3. 例题分析：关键问题

```
//向POP服务器发送建立连接请求
connect( sock, (sockaddr*)&serveraddr,
         sizeof(serveraddr));

//从POP服务器获得应答信息
recv(sock, Respond, MAX_SIZE, 0);

//从应答信息中解析POP应答码
...
```

3. 例题分析：关键问题

- 登录到POP服务器

- 客户机向服务器发送**USER**命令，服务器可能返回的应答是**+OK**或**-ERR**
- 当接收的应答码为**+OK**时，表示用户名正确并且需继续输入密码

3. 例题分析：关键问题

- 登录到POP服务器

- 客户机向服务器发送**PASS**命令，服务器可能返回的应答是**+OK**或**-ERR**
- 当接收到的应答码是**+OK**时，表示用户名和密码均正确并已成功登陆

3. 例题分析： 关键问题

```
//构造标准的USER命令
memcpy(Command, "USER ", strlen("USER "));
memcpy(Command+strlen("USER "),
        m_Account,                //用户名
        strlen(m_Account));

//向POP服务器发送USER命令
send(sock, Command, strlen(Command), 0);
//从POP服务器获得应答信息
recv(sock, Respnd, MAX_SIZE, 0);
```

3. 例题分析：关键问题

```
//解析POP应答码
if (POP应答码== “+OK” )
{
    输出POP服务器的应答信息
    //构造标准的PASS命令
    memcpy (Command, “PASS ”, strlen( “PASS ” ));
    memcpy (Command+strlen( “PASS ” ),
            m_Password,
            strlen(m_Password));
    ...
}
```

3. 例题分析：关键问题

解析POP应答码

```
if (POP应答码 == “+OK” )
```

```
{
```

```
...
```

```
//向POP服务器发送PASS命令
```

```
send(sock, Command, strlen(Command), 0);
```

```
//从POP服务器获得应答信息
```

```
recv(sock, Respond, MAX_SIZE, 0);
```

```
解析应答码
```

```
if (POP应答码 == “+OK” )
```

```
    输出POP服务器应答信息
```

```
}
```

3. 例题分析：关键问题

- 接收邮件列表

- **STAT**命令用来返回邮箱的统计信息
- 客户机向服务器发送**STAT**命令，服务器可能返回的应答是**+OK**或**-ERR**
- 当接收到**+OK**时，应答信息的后两个值依次为**邮件数量**与**字节总数**

3. 例题分析：关键问题

- 接收邮件列表

- **LIST**命令用来返回邮箱中的邮件列表
- 客户机向服务器发送**LIST**命令，服务器可能返回的应答是**+OK**或**-ERR**
- 当接收到**+OK**时，后面接收的邮件列表，直到接收到句点“.”为止

3. 例题分析：关键问题

```
//构造标准的STAT命令  
memcpy(Command, "STAT", strlen("STAT"));  
//向POP服务器发送STAT命令  
send(sock, Command, strlen(Command), 0);  
  
//从POP服务器获得应答信息  
recv(sock, Respond, MAX_SIZE, 0);  
从应答信息中解析POP应答码  
if (POP应答码 == "+OK")  
    输出POP服务器的应答信息
```

3. 例题分析：关键问题

构造标准的LIST命令

//向服务器发送LIST命令

```
send(sock, Command, strlen(Command), 0);
```

//获得应答信息

```
recv(sock, Respnd, MAX_SIZE, 0);
```

解析POP应答码

```
if (POP应答码 == "+OK" )
```

```
{
```

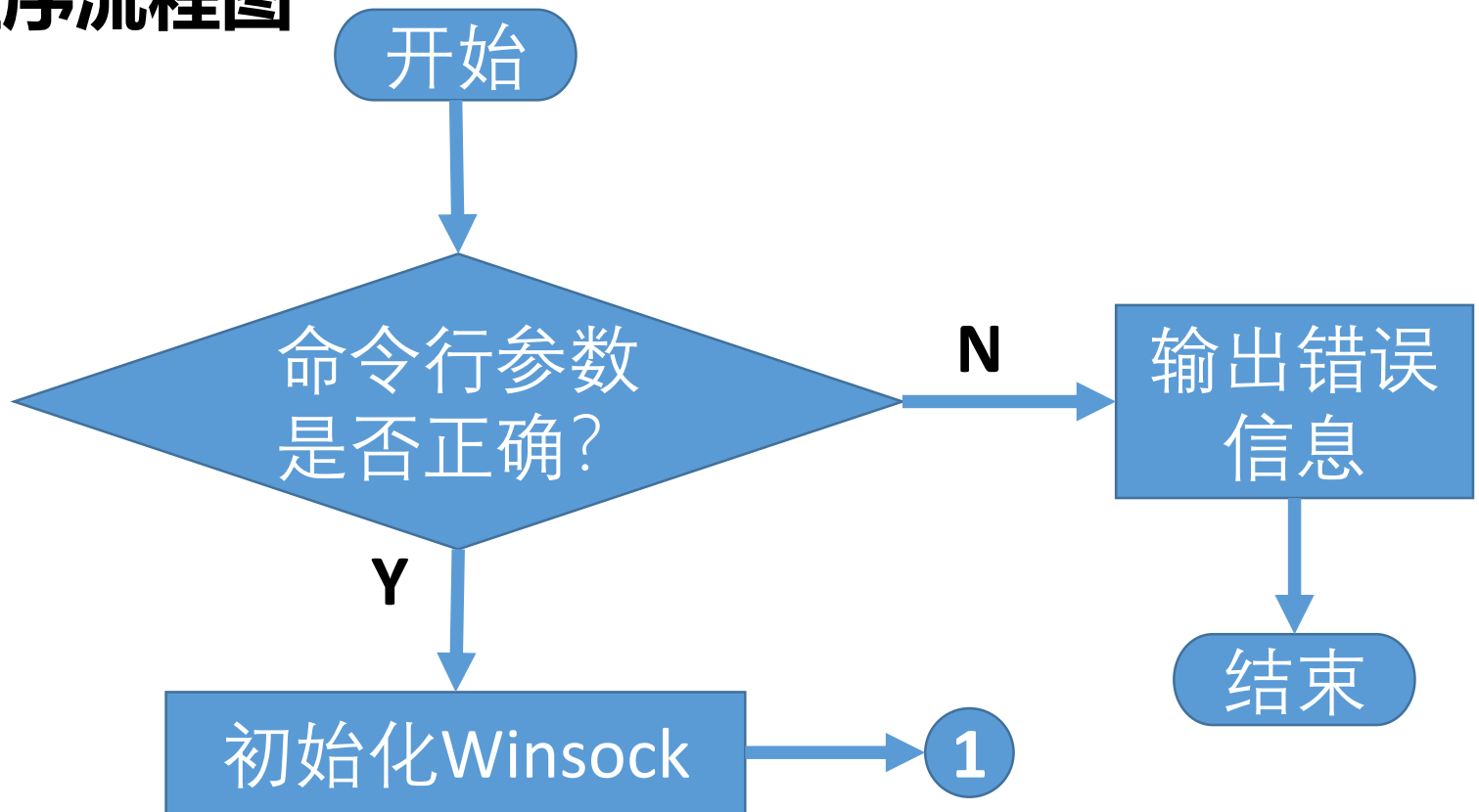
 输出POP服务器的应答信息

 输出接收到的邮件列表

```
}
```

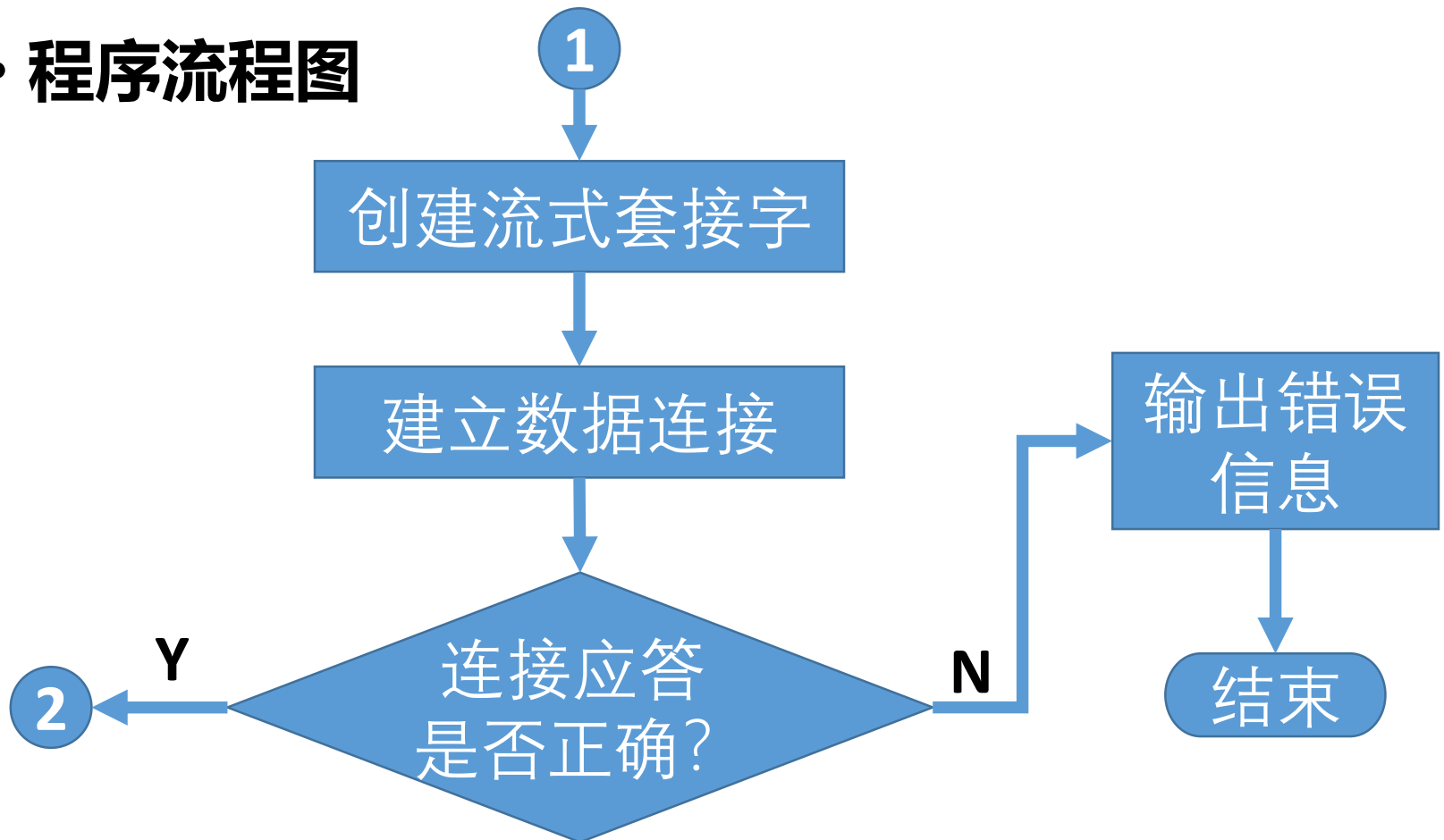
3. 例题分析：关键问题

- 程序流程图



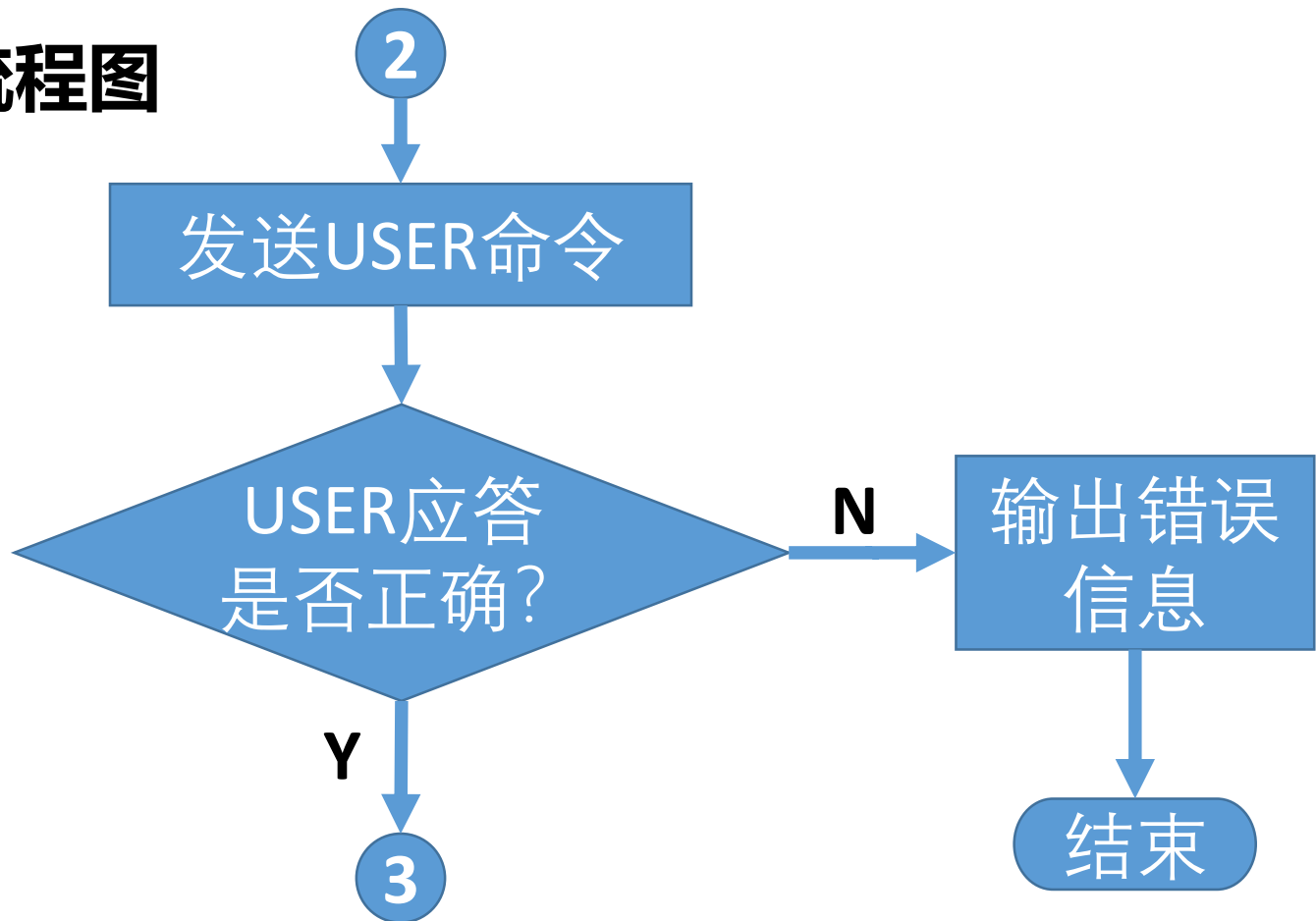
3. 例题分析：关键问题

- 程序流程图



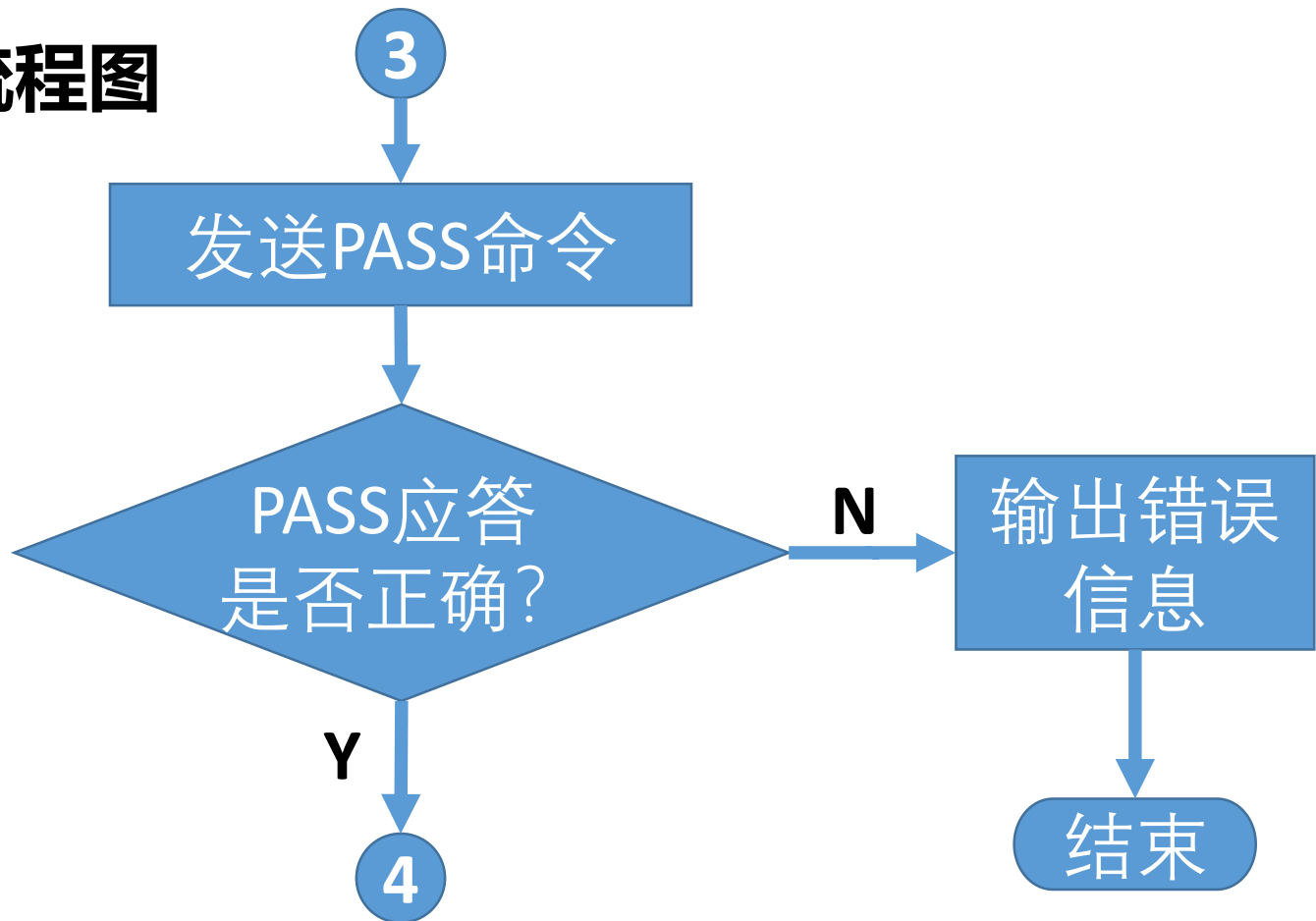
3. 例题分析：关键问题

- 程序流程图



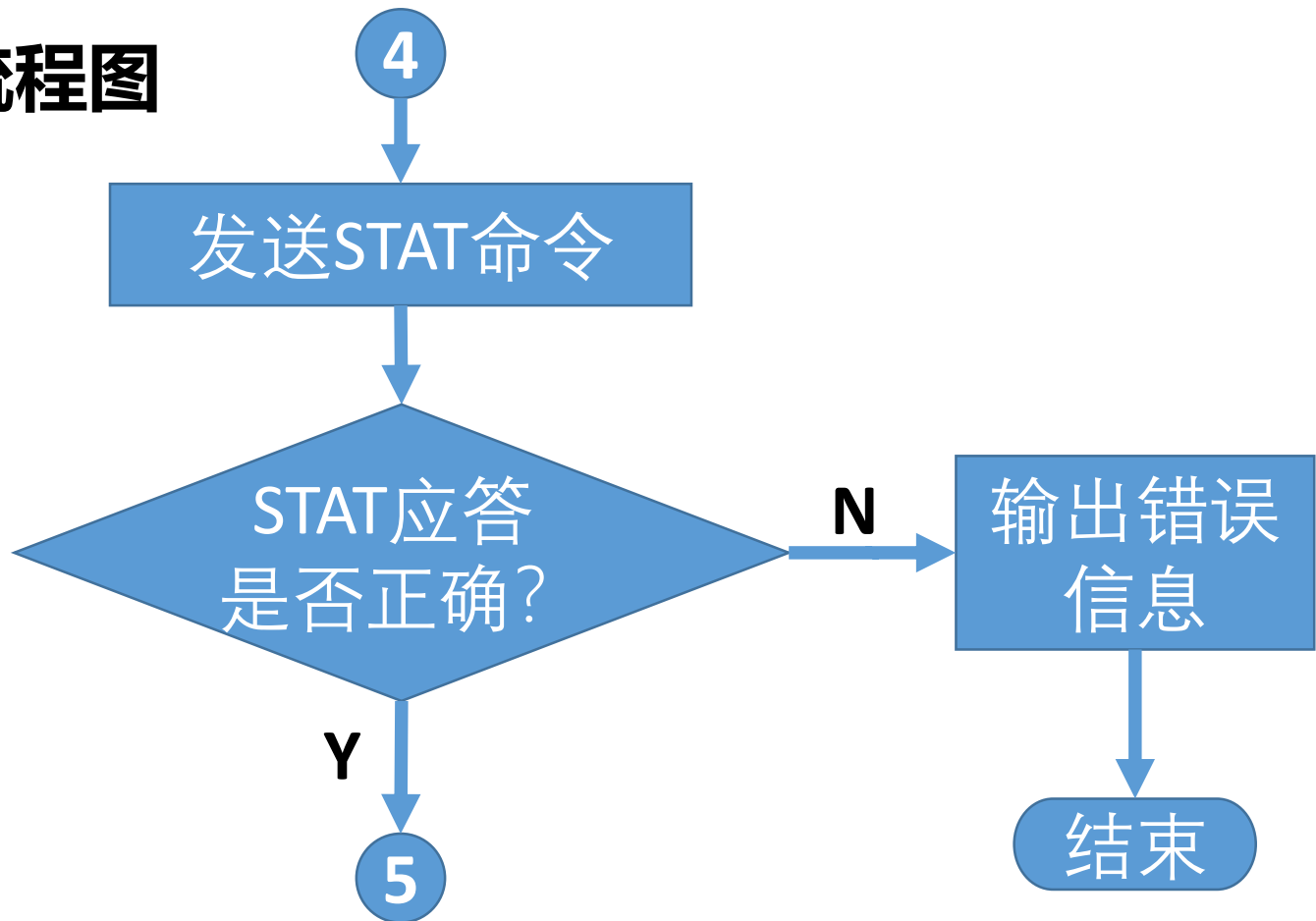
3. 例题分析：关键问题

- 程序流程图



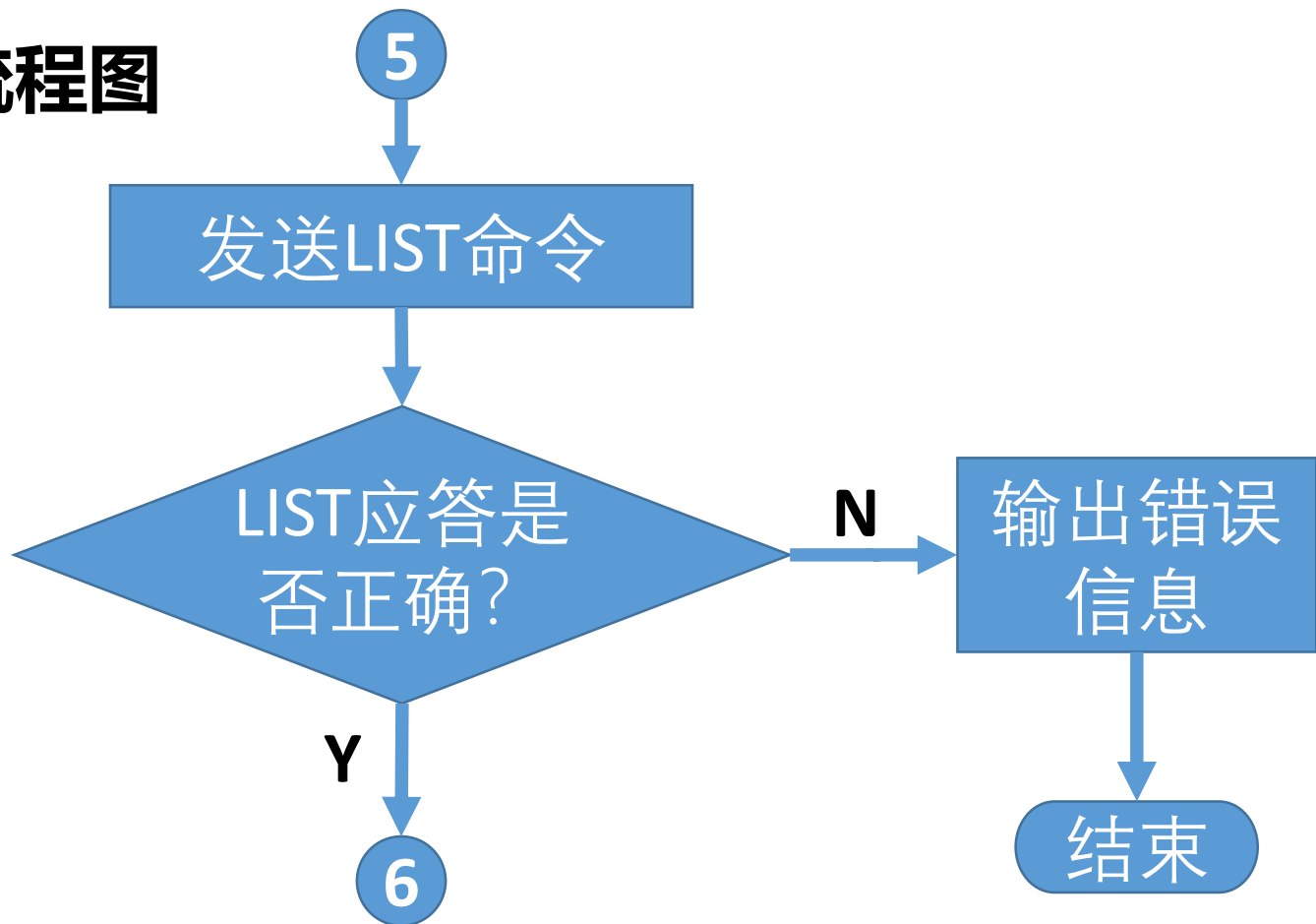
3. 例题分析：关键问题

- 程序流程图



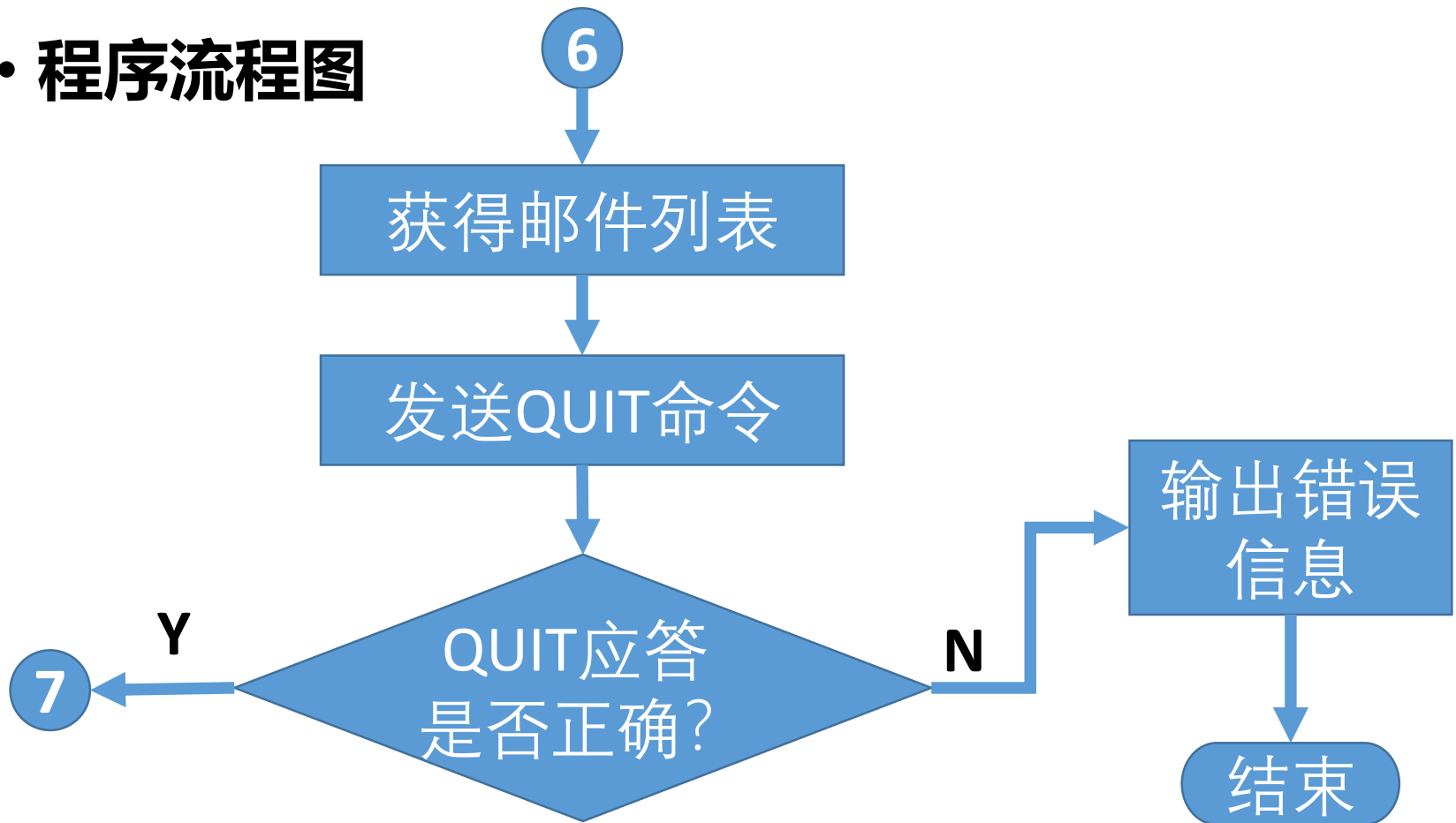
3. 例题分析：关键问题

- 程序流程图



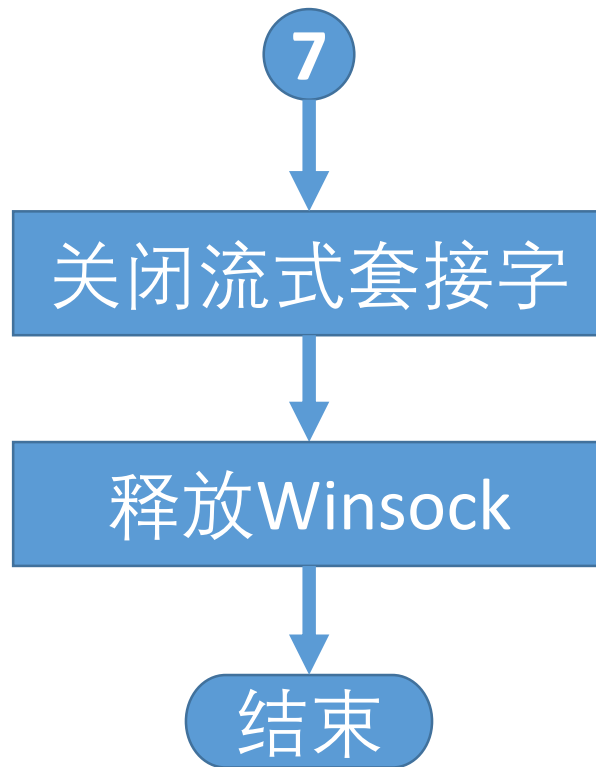
3. 例题分析：关键问题

- 程序流程图



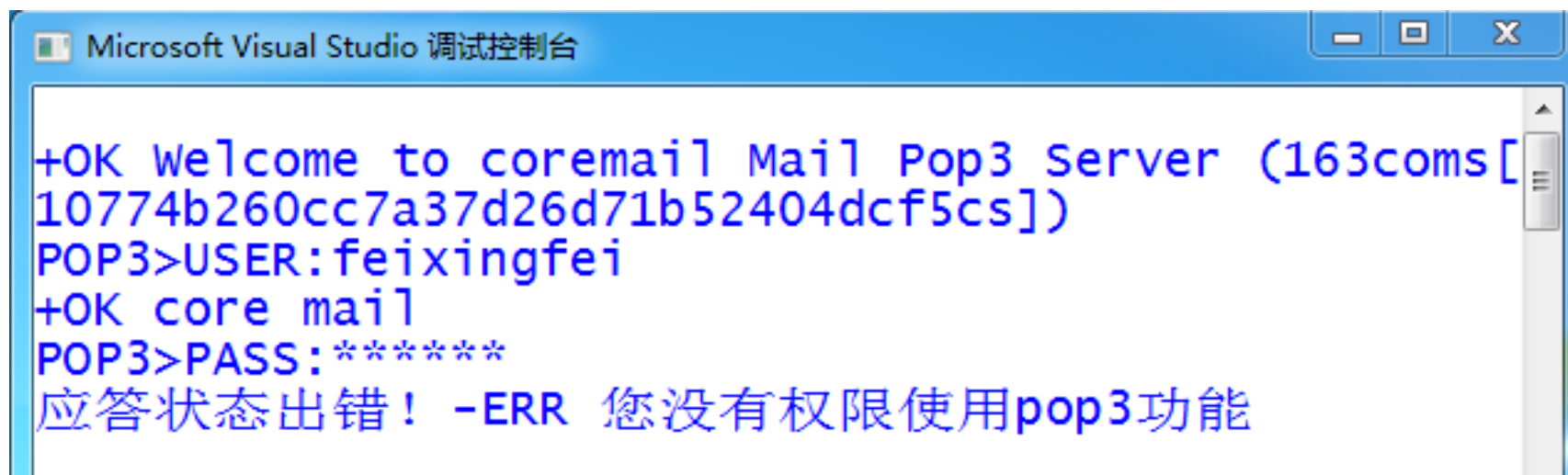
3. 例题分析：关键问题

- 程序流程图



3. 例题分析：关键问题

• 程序演示



```
Microsoft Visual Studio 调试控制台

+OK Welcome to coremail Mail Pop3 Server (163coms[
10774b260cc7a37d26d71b52404dcf5cs])
POP3>USER:feixingfei
+OK core mail
POP3>PASS:*****
应答状态出错！ -ERR 您没有权限使用pop3功能
```


3. 例题分析：关键问题

- 程序演示



```
Microsoft Visual Studio 调试控制台

+OK Welcome to coremail Mail Pop3 Server (163coms[
10774b260cc7a37d26d71b52404dcf5cs])
POP3>USER:feixingfei
+OK core mail
POP3>PASS:*****
+OK 8399 message(s) [1142890645 byte(s)]
POP3>STAT
+OK 8399 1142890645
POP3>LIST:10
+OK 10 7007
POP3>QUIT
+OK core mail
```

本章小结

- **设计目的**

- **了解电子邮件基本概念、主要功能；掌握应用层服务设计思路和编程方法**

- **相关知识**

- **电子邮件基本概念、工作原理、邮件地址、邮件格式和POP命令与应答**

- **例题分析**

- **流式套接字、登录POP、接收邮件列表**