实验一 利用SOCKET编写聊天程序

郭坤昌 2012522

一、 聊天程序流程

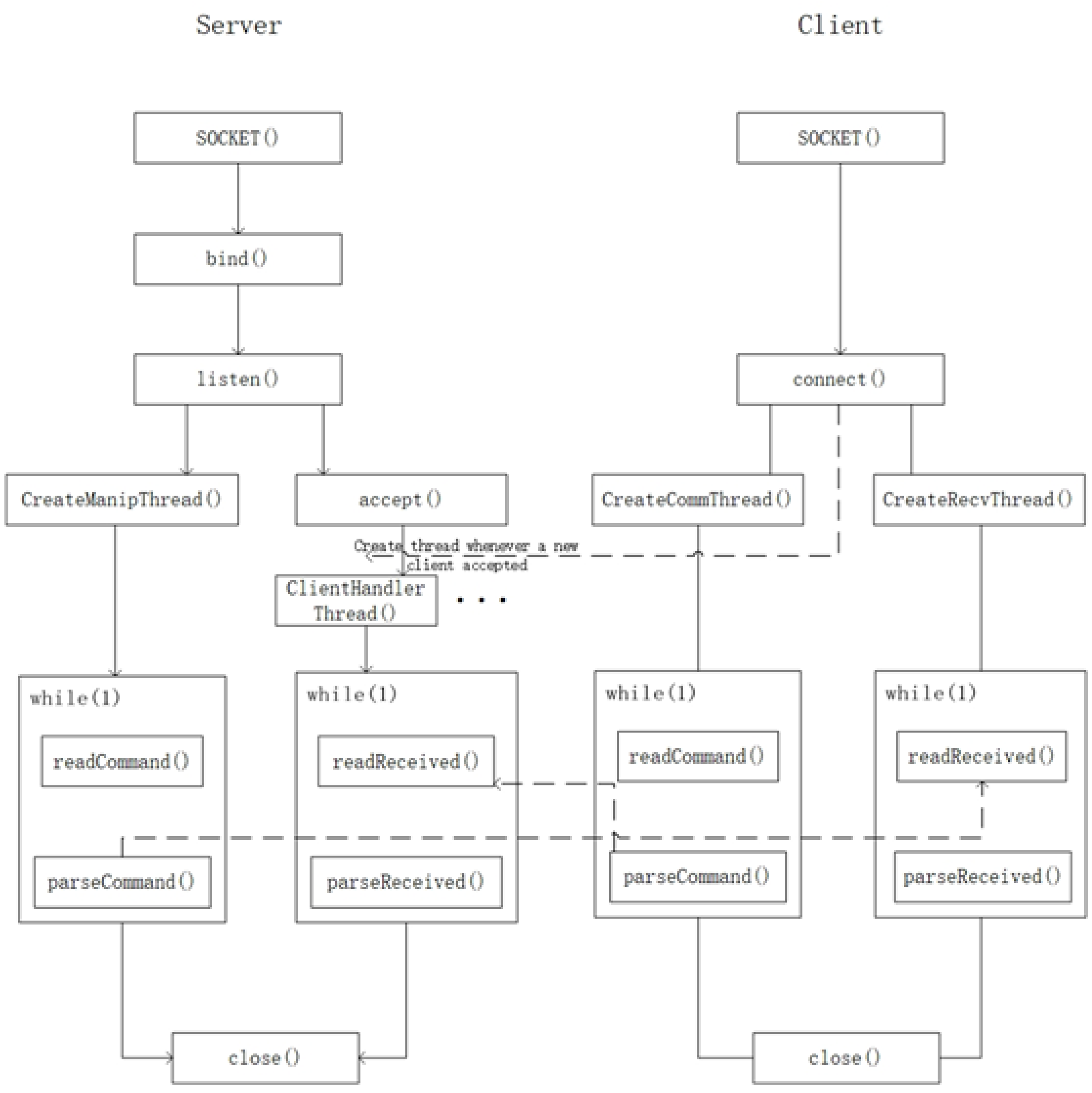
聊天程序的整体运行思路为：以服务器为核心，对各客户端进行管理调度；客户端直接与服务器交互，客户端彼此之间不直接通信。

对服务器：加载链接库，创建套接字，将套接字绑定到指定端口，监听客户端的连接。每当有新的客户端连接时，创建新的客户端处理线程，读取客户端传送来的指令，进行处理。服务器在接收的同时也能进行指令输入，通过创建新的写入线程实现。

对客户端，加载链接库，创建套接字，将客户端连接到服务器，将输入的指令进行处理，发送到服务器。在写入的同时，客户端也能进行任意时刻读取并处理服务器发送来的指令，通过创建新的接收线程实现。

聊天程序的整体运行思路为：以服务器为核心，对各客户端进行管理调度；客户端直接与服务器交互，客户端彼此之间不直接通信。

程序流程如下图所示。



二、 协议设计实现

协议由语法、语义、时序组成。程序的一个重点在于如何对指令进行语法解析，并根据得到的语义，按一定顺序流程执行相应动作。

对服务器和客户端，均有主动写入指令和被动接收指令两种需要，客户端和服务器略有区别。

对于客户端的写入和接收，按如下方式定义指令头（指令的前4个字节），用以区分指令的类型和动作。注释部分。具体指令格式和释义如下表。

指令类型	指令头	指令格式	指令含义
COMM_QUIT	"QUIT"	QUIT	退出
COMM_HELP	"HELP"	HELP	显示帮助
COMM_NAME	"NAME"	NAME [new name]	重命名
COMM_SEND	"SEND"	SEND [message] TO [receive id]	向指定ID用户发送信息
COMM_BROA	"BROA"	BROA [message]	广播信息
RECV_IDEN	"IDEN"	IDEN ID:[id]	接收确认当前用户ID
RECV_SEND	"SEND"	SEND [message] FROM [send id]	接收私聊信息
RECV_BROA	"BROA"	BROA [message] FROM [send id]	接收广播信息

对于服务器的写入和接收，按如下方式定义指令头（指令的前4个字节），用以区分指令的类型和动作。注释部分。具体指令格式和释义如下表。

指令类型	指令头	指令格式	指令含义
COMM_QUIT	"QUIT"	QUIT	退出
COMM_LIST	"LIST"	LIST	列出所有用户ID和名字
COMM_HELP	"HELP"	HELP	显示帮助
COMM_SEND	"SEND"	SEND [message] TO [receive id]	向指定ID用户发送信息
COMM_BROA	"BROA"	BROA [message]	广播信息

指令类型	指令头	指令格式	指令含义
COMM_IDEN	"IDEN"	IDEN ID:[id]	设定特定用户ID
COMM_KICK	"KICK"	KICK [id]	踢出特定用户
RECV_SEND	"SEND"	SEND [message] TO [recv id] FROM [send id]	转发私聊消息
RECV_BROA	"BROA"	BROA [message] FROM [send id]	转发广播信息
RECV_NAME	“NAME”	NAME [id]:[new name]	为用户重命名

三、 模块功能介绍

客户端与服务器程序组成结构类似，如下表。

模块名称	实现功能
Server/Client	如第一部分程序流程中所示。
config	指令头格式，缓冲区大小，客户在线状态
Util	获取时间戳，重写获取特定子串方法

3.1 Server中的核心部分介绍

- 客户端链表的创建和管理

客户端链表由如下数据结构管理，包含了客户端的ID、名称、套接字、状态和指向下一个客户端的指针。

```
class ClientInfo
{
public:
    int id;
    char* name;
    SOCKET* s;
    ClientInfo* nextClient;
    bool status;

    ClientInfo()
    {
        id = -1;
        name = NULL;
        s = NULL;
        nextClient = NULL;
        status = ONLINE;
    }
};

```

客户端完整创建于接收连接成功后，并为其单独创建线程管理对应客户端的指令传递。

```
int Server::startThread()
{
    clientCount = 0;
    clientList = new ClientInfo();
    ClientInfo* currClient = clientList;
    CreateThread(NULL, NULL, (LPTHREAD_START_ROUTINE)manipThread, (LPVOID)this, NULL, NULL);
    while (1)
    {
        // accept a new client
        currClient->s = new SOCKET;
        currClient->nextClient = new ClientInfo();
        int len = sizeof(serverAddress);
        if ((*(currClient->s) = accept(s, (struct sockaddr*)&serverAddress, &len)) == INVALID_SOCKET)
        {
            cout << this->util.currentTime().append(" Failed to accept: ") << WSAGetLastError() << endl;
            return -1;
        }

        // start a new thread for the client
        currClient->id = clientCount;
        struct params p;
        p.server = this;
        p.client = currClient;
        CreateThread(NULL, NULL, (LPTHREAD_START_ROUTINE)clientHandlerThread, (LPVOID)&p, NULL, NULL);

        // adjust client list
        cout << this->util.currentTime().append(" Accepted client ") << currClient->id << endl;
        clientCount++;
        currClient = currClient->nextClient;
    }
    return 0;
}
```

- 客户端线程的参数传递

由于后续对指令的解析和动作需要对服务器和客户端中的变量进行控制，因此在如上创建线程过程中，需要同时将服务器和客户端的指针传递给静态的线程函数，通过结构体`params`实现

```
// 创建线程时的参数传递
struct params p;
p.server = this;
p.client = currClient;
CreateThread(NULL, NULL,
(LPTHREAD_START_ROUTINE)clientHandlerThread, (LPVOID)&p, NULL,
NULL);

// 结构体定义
struct params {
    Server* server;
    ClientInfo* client;
};
```

- 指令解析

指令解析实际上是对特定状态进行反应的过程，整体是通过`if-else`结构对指令头判断进行相应动作。在server中，对接收指令的解析如下。以解析到的发送指令为例，服务器作为中转，需要获取指令的发送者和接收者，并遍历客户端链表向特定客户端发送信息。

```
void Server::parseReceived(char* recv, Server* server)
{
    string recvStr = recv;

    if (recvStr.substr(0, 4) == RECV_SEND)  // SEND [message] TO
[receive id] FROM [send id]
    {
        cout << server->getUtil().currentTime().append("
").append(recvStr) << endl;

        int recvID = atoi(server->getUtil().getSubStr(recvStr,
recvStr.find("TO") + 3, recvStr.find("FROM") - 2).c_str());
        int sendID = atoi(server->getUtil().getSubStr(recvStr,
recvStr.find("FROM") + 5).c_str());
        string newComm = server->getUtil().getSubStr(recvStr, 0,
recvStr.find("TO") - 1).append("FROM ").append(to_string(sendID));
// new:  SEND [message] FROM [send id]
        sendToClient(newComm, findClientByID(recvID, server-
>clientList));
    }
    else if (recvStr.substr(0, 4) == RECV_BROA) // BROA [message]
FROM [send id]
    {
        cout << server->getUtil().currentTime().append("
").append(recvStr) << endl;

        int sendID = atoi(server->getUtil().getSubStr(recvStr,
recvStr.find("FROM") + 5).c_str());
        ClientInfo* currClient = server->clientList;
        while (currClient != NULL)  // forward message to all
clients
        {
            if (currClient->id != sendID && currClient->id >= 0)
            {
                sendToClient(recvStr, currClient->s);
            }
            currClient = currClient->nextClient;
        }
    }
    else if (recvStr.substr(0,4)==RECV_NAME)    // NAME [id]:[new name]
    {
        int id = atoi(server->getUtil().getSubStr(recvStr, recvStr.find(' ') + 1, recvStr.find(':') - 1).c_str());
        string name = server->getUtil().getSubStr(recvStr, recvStr.find(':') + 1);
        ClientInfo* currClient = server->clientList;
        while (currClient != NULL)
        {
            if (currClient->id == id)
            {
                currClient->name = new char[name.length() + 1];
                strcpy_s(currClient->name, name.length() + 1, name.c_str());
                cout << server->getUtil().currentTime().append(" ").append("Client ").append(to_string(id)).append(" changed name to ").append(name) << endl;
                break;
            }
            currClient = currClient->nextClient;
        }
    }
    else
    {
        cout << server->getUtil().currentTime().append(" Invalid command received.\n");
    }
}
```

对输入指令的解析如下：

```
void Server::parseCommand(char* comm, Server* server)
{
    string commStr = comm;
    if (commStr.substr(0, 4) == COMM_QUIT)  // QUIT
    {
        cout << server->getUtil().currentTime().append(" Quited\n");

        closesocket(server->s);
        WSACleanup();
        exit(0);
    }
    else if (commStr.substr(0, 4) == COMM_HELP)
    {
        cout << server->getUtil().currentTime().append(" Help:\n\t");
        cout << helpTxt << endl;
```

```cpp
    }
    else if (commStr.substr(0, 4) == COMM_LIST) // LIST3
    {
        cout << server->getUtil().currentTime().append(" Clients listed: \n");

        ClientInfo* currClient = server->clientList;
        while (currClient && currClient->id != -1)
        {
            cout << "\t Client " << currClient->id << " : " << ((currClient->name == nullptr) ? "NULL" : (currClient->name)) << endl;
            currClient = currClient->nextClient;
        }
    }
    else if (commStr.substr(0, 4) == COMM_SEND) // SEND [message] TO [recv id]
    {
        cout << server->getUtil().currentTime().append(" ").append(commStr) << endl;

        int recvID = atoi(server->getUtil().getSubStr(commStr, commStr.find("TO") + 3).c_str());
        string newComm = server->getUtil().getSubStr(commStr, 0, commStr.find("TO")-1).append("FROM -1");    //new: SEND [message] FROM -1
        sendToClient(newComm, findClientByID(recvID, server->clientList));
    }
    else if (commStr.substr(0,4) == COMM_BROA) // BROA [message]
    {
        cout << server->getUtil().currentTime().append(" ").append(commStr);

        ClientInfo* currClient = server->clientList;
        string newComm = commStr.append(" FROM -1");     // new: BROA [message] FROM -1
        while (currClient != NULL && currClient->id != -1)
        {
            sendToClient(commStr, currClient->s);
            currClient = currClient->nextClient;
        }
    }
    else if (commStr.substr(0, 4) == COMM_KICK) // KICK [id]
    {
        int kickID = atoi(server->getUtil().getSubStr(commStr, commStr.find(' ') + 1).c_str());

        cout << server->getUtil().currentTime().append(" Kicked client ").append(to_string(kickID)) << endl;

        ClientInfo* currClient = server->clientList;
        while (currClient != NULL)
        {
            if (currClient->id == kickID)
```

```
                {
                    currClient->status = OFFLINE;
                    closesocket(*currClient->s);
                    break;
                }
                currClient = currClient->nextClient;
            }
    }
    else
    {
        cout << "Please enter correct command.[HELP]" << endl;
    }
}
```

3.2 Client中的核心部分介绍

- 同时读写

```
int Client::startThread()
{
    char commBuffer[BUFFER_SIZE] = {0};
    CreateThread(NULL, NULL, (LPTHREAD_START_ROUTINE)recvThread,
(LPVOID)this, NULL, NULL); // start
    while (!exitFlag)
    {
        gets_s(commBuffer);
        parseCommand(commBuffer, this);
        /*if (send(s, commBuffer, strlen(commBuffer), 0) ==
SOCKET_ERROR)
        {
            cout << "Send failed: " << WSAGetLastError() << endl;
            break;
        }
        cout << "Data sended." << endl;*/
        memset(commBuffer, 0, BUFFER_SIZE);
    }
    return 0;
}
```

- 指令解析

指令解析部分类似服务器端。对输入指令的解析：

```
int Client::parseCommand(char* comm, Client* client)
{
    string commStr = comm;
    if (commStr.substr(0, 4) == COMM_QUIT)  // QUIT
    {
        cout << client->getUtil().currentTime().append("
Quited.\n");
        client->setExitFlag(1);
    }
    else if (commStr.substr(0, 4) == COMM_SEND) // IN : SEND [message] TO [receive id]      OUT : SEND [message] TO [receive id] FROM [send id]
    {
        int recvID = atoi(client->getUtil().getSubStr(commStr, commStr.find("TO") + 3).c_str());
        string message = client->getUtil().getSubStr(commStr, commStr.find(' ') + 1, commStr.find("TO") - 2);

        cout << client->getUtil().currentTime().append(" ---> ").append(to_string(recvID)).append(" : ").append(message) << endl;

        string newComm = commStr.append(" FROM ").append(to_string(client->getID()));
        char* sendMessage = new char[newComm.length() + 1];
        strcpy_s(sendMessage, newComm.length() + 1, newComm.c_str());
        if (send(client->s, sendMessage, strlen(sendMessage), 0) == SOCKET_ERROR)
        {
            cout << client->getUtil().currentTime().append(" Send failed: ") << WSAGetLastError() << endl;
        }
    }
    else if (commStr.substr(0, 4) == COMM_BROA) // IN : BROA [message]      OUT:  BROA [message] FROM [send id]
    {
        string message = client->getUtil().getSubStr(commStr, commStr.find(' ') + 1);

        cout << client->getUtil().currentTime().append(" ---> ALL : ").append(message) << endl;

        string newComm = commStr.append(" FROM ").append(to_string(client->getID()));
        sendToServer(newComm, &client->s);
    }
    return 0;
}
```

对接收指令的解析:

```
int Client::parseReceived(char* recv, Client* client)
{
    string recvStr = recv;
    if (recvStr.substr(0, 4) == RECV_IDEN)  // IDEN [id]
    {

        client->id = atoi(client->getUtil().getSubStr(recvStr, recvStr.find(' ') + 1).c_str()); // get allocated id from server
        sendToServer(string("").append(COMM_NAME).append(" ").append(to_string(client->id)).append(":").append(client->name), client->getServerSocket());   // send my name to server

```

```cpp
        cout << client->getUtil().currentTime().append(" Accepted. Your ID is ").append(to_string(client->id)) << endl;
    }
    else if (recvStr.substr(0, 4) == RECV_SEND) // SEND [message] FROM [send id]
    {
        int sendID = atoi(client->getUtil().getSubStr(recvStr, recvStr.find("FROM") + 5).c_str());
        string message = client->getUtil().getSubStr(recvStr, recvStr.find(' ') + 1, recvStr.find("FROM") - 2);

        cout << client->getUtil().currentTime().append(" <--- ").append(to_string(sendID)).append(" : ").append(message) << endl;
    }
    else if (recvStr.substr(0, 4) == RECV_BROA) // BROA [message] FROM [send id]
    {
        int sendID = atoi(client->getUtil().getSubStr(recvStr, recvStr.find("FROM") + 5).c_str());
        string message = client->getUtil().getSubStr(recvStr, recvStr.find(' ') + 1, recvStr.find("FROM") - 2);

        cout<<client->getUtil().currentTime().append(" ALL <--- ").append(to_string(sendID)).append(" : ").append(message) << endl;
    }
    return 0;
}
```

3.3 Util中的核心部分介绍

- 获取时间戳

使用`time.h`提供的函数实现。

```cpp
string Util::currentTime()
{
    struct tm t;     // tm结构指针
    time_t now; // 声明time_t类型变量
    time(&now); // 获取系统日期和时间
    localtime_s(&t, &now);  // 获取当地日期和时间

    string s = "";
    if (t.tm_hour < 10) s += "0";
    s += to_string(t.tm_hour);
    s += ":";
    if (t.tm_min < 10) s += "0";
    s += to_string(t.tm_min);
    s += ":";
    if (t.tm_sec < 10) s += "0";
    s += to_string(t.tm_sec);
    return s;
}
```

3.4 Config配置

config.h中配置重要的参数，指令格式等

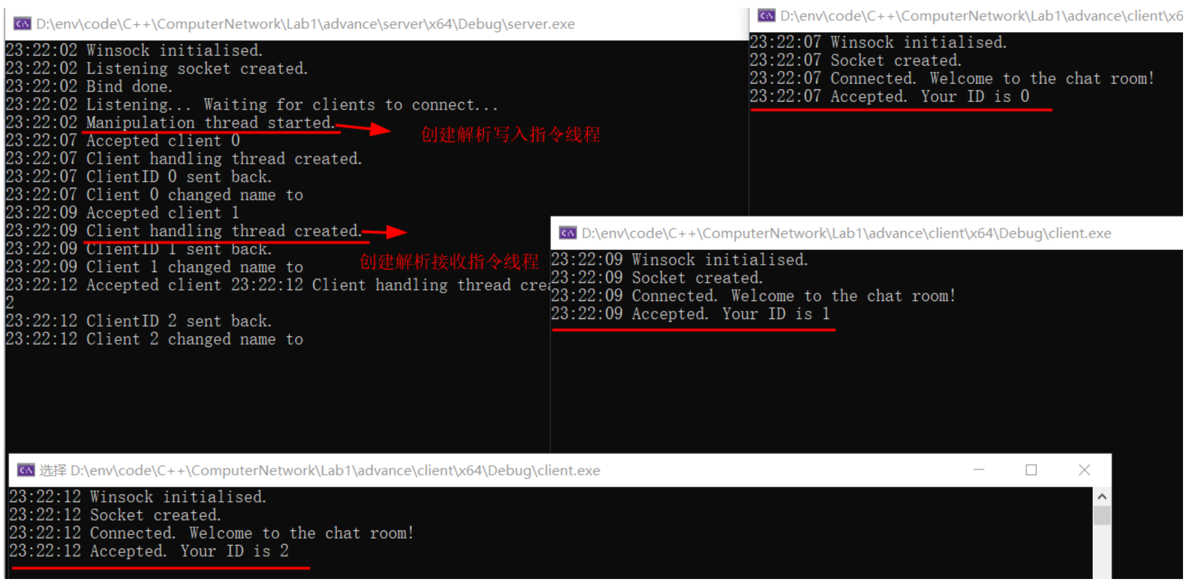
```
#define _WINSOCK_DEPRECATED_NO_WARNINGS 1   // enable deprecated functions
#define SERVERADDRESS "127.0.0.1"
#define PORT 8888
#define BUFFER_SIZE 1024

#define COMMAND_LEN 4

#define ONLINE true
#define OFFLINE false

#define COMM_QUIT "QUIT"    // QUIT
#define COMM_HELP "HELP"    // HELP
#define COMM_IDEN "IDEN"    // IDEN [id]
#define COMM_NAME "NAME"    // NAME [id]:[new name]
#define COMM_SEND "SEND"    // SEND [receive id]:[message]
#define COMM_BROA "BROA"    // BROA [from id]:[message]
#define COMM_LIST "LIST"    // LIST
#define COMM_KICK "KICK"    // KICK [id]

#define RECV_SEND "SEND"    // SEND [id]:[message]
#define RECV_BROA "BROA"    // BROA [id]:[message]
#define RECV_NAME "NAME"    // NAME [id]:[new name]    send id is eliminated cause specific is handling the socket

const char helpTxt[1024] = "This is how you can use s-c chatroom:\n\t QUIT: quit the chatroom\n\t HELP: show this help\n\t IDEN [id]: identify yourself\n\t NAME [id]:[new name]: change your name\n\t SEND [receive id]:[message]: send message to specific user\n\t BROA [from id]:[message]: broadcast message to all users\n\t LIST: list all users\n\t KICK [id]: kick specific user out of the chatroom";

```

四、 设计时的问题

1. `recv`函数返回接收区中不为0的字节数，若重置时没有将所有字节都置为0，则可能出现返回值大于等于0的情况，造成线程对应的`while`循环判断始终为真，无法正常接收输入。
2. 在服务器中管理客户端成员表，使用链表实现，在创建客户端时，需要初始化其指向下一个客户端的指针，否则出现空指针访问出错
3. `substr`函数的参数由开始位置和长度组成，开始时将结束位置作为了第二个参数，因此出错，重写了新的获取子串方法以解决指令解析的需要。

五、 程序演示

启动服务器和三个客户端，此处预先设定好了服务器地址和端口号。服务器和客户端输出带时间戳的对应日志信息。



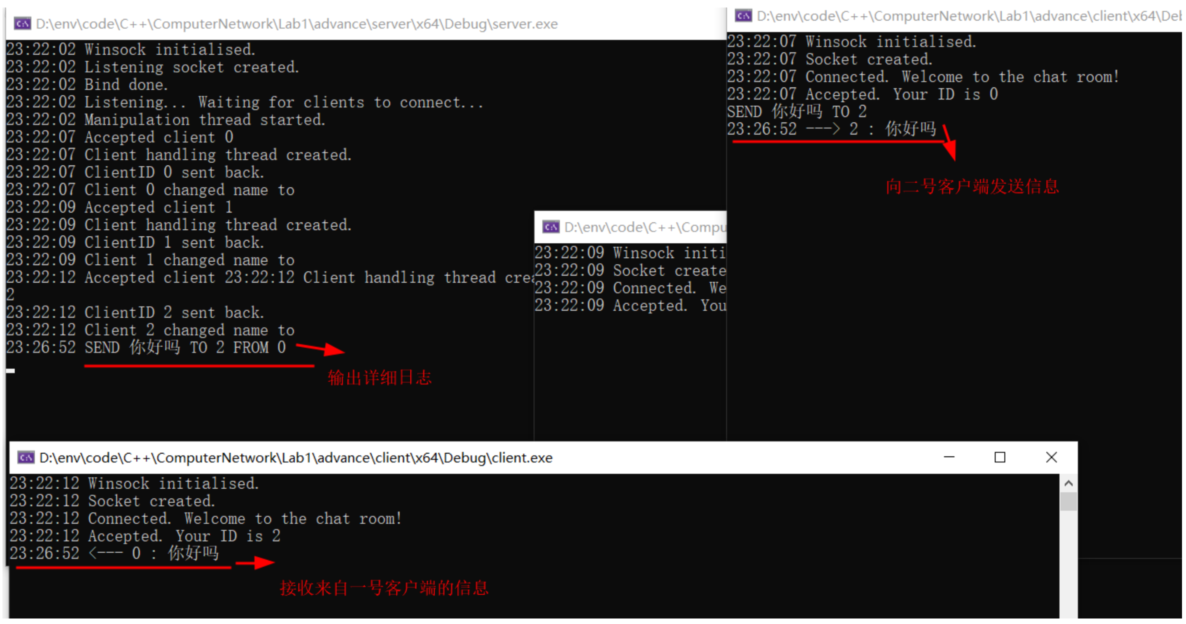
```
D:\env\code\C++\ComputerNetwork\Lab1\advance\server\x64\Debug\server.exe
23:22:02 Winsock initialised.
23:22:02 Listening socket created.
23:22:02 Bind done.
23:22:02 Listening... Waiting for clients to connect...
23:22:02 Manipulation thread started.
23:22:07 Accepted client 0
23:22:07 Client handling thread created.
23:22:07 ClientID 0 sent back.
23:22:07 Client 0 changed name to
23:22:09 Accepted client 1
23:22:09 Client handling thread created.
23:22:09 ClientID 1 sent back.
23:22:09 Client 1 changed name to
23:22:12 Accepted client 23:22:12 Client handling thread cre
2
23:22:12 ClientID 2 sent back.
23:22:12 Client 2 changed name to
```

```
D:\env\code\C++\ComputerNetwork\Lab1\advance\client\x6
23:22:07 Winsock initialised.
23:22:07 Socket created.
23:22:07 Connected. Welcome to the chat room!
23:22:07 Accepted. Your ID is 0
```

```
D:\env\code\C++\ComputerNetwork\Lab1\advance\client\x64\Debug\client.exe
23:22:09 Winsock initialised.
23:22:09 Socket created.
23:22:09 Connected. Welcome to the chat room!
23:22:09 Accepted. Your ID is 1
```

```
选择 D:\env\code\C++\ComputerNetwork\Lab1\advance\client\x64\Debug\client.exe
23:22:12 Winsock initialised.
23:22:12 Socket created.
23:22:12 Connected. Welcome to the chat room!
23:22:12 Accepted. Your ID is 2
```

测试私聊功能，由零号客户端向二号客户端发送信息。



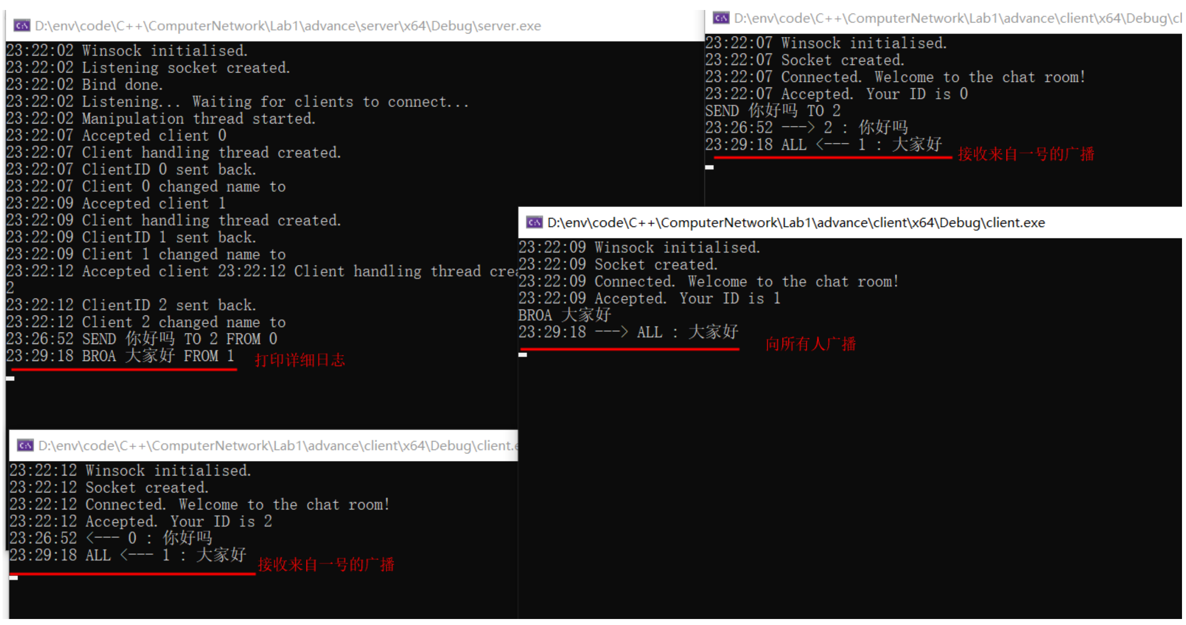
```
D:\env\code\C++\ComputerNetwork\Lab1\advance\server\x64\Debug\server.exe
23:22:02 Winsock initialised.
23:22:02 Listening socket created.
23:22:02 Bind done.
23:22:02 Listening... Waiting for clients to connect...
23:22:02 Manipulation thread started.
23:22:07 Accepted client 0
23:22:07 Client handling thread created.
23:22:07 ClientID 0 sent back.
23:22:07 Client 0 changed name to
23:22:09 Accepted client 1
23:22:09 Client handling thread created.
23:22:09 ClientID 1 sent back.
23:22:09 Client 1 changed name to
23:22:12 Accepted client 23:22:12 Client handling thread cre
2
23:22:12 ClientID 2 sent back.
23:22:12 Client 2 changed name to
23:26:52 SEND 你好吗 TO 2 FROM 0
```

```
D:\env\code\C++\ComputerNetwork\Lab1\advance\client\x64\Del
23:22:07 Winsock initialised.
23:22:07 Socket created.
23:22:07 Connected. Welcome to the chat room!
23:22:07 Accepted. Your ID is 0
SEND 你好吗 TO 2
23:26:52 ---> 2 : 你好吗
```

```
D:\env\code\C++\Compu
23:22:09 Winsock initi
23:22:09 Socket create
23:22:09 Connected. We
23:22:09 Accepted. You
```

```
D:\env\code\C++\ComputerNetwork\Lab1\advance\client\x64\Debug\client.exe
23:22:12 Winsock initialised.
23:22:12 Socket created.
23:22:12 Connected. Welcome to the chat room!
23:22:12 Accepted. Your ID is 2
23:26:52 <--- 0 : 你好吗
```

测试广播功能，由一号客户端向所有用户广播消息。



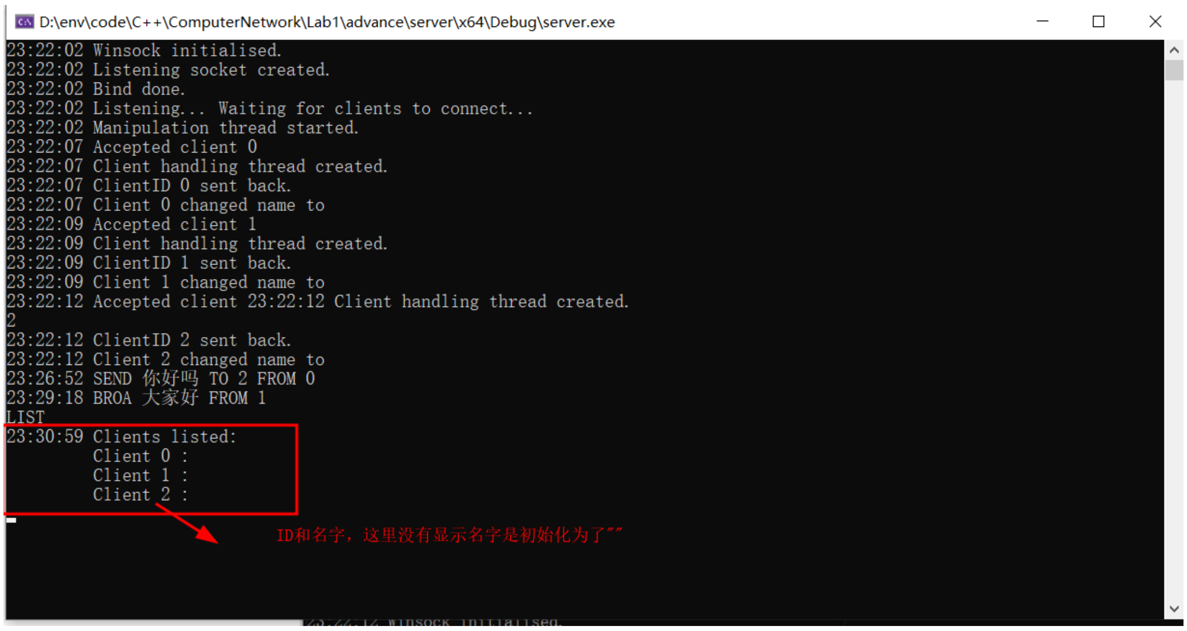
```
D:\env\code\C++\ComputerNetwork\Lab1\advance\server\x64\Debug\server.exe
23:22:02 Winsock initialised.
23:22:02 Listening socket created.
23:22:02 Bind done.
23:22:02 Listening... Waiting for clients to connect...
23:22:02 Manipulation thread started.
23:22:07 Accepted client 0
23:22:07 Client handling thread created.
23:22:07 ClientID 0 sent back.
23:22:07 Client 0 changed name to
23:22:09 Accepted client 1
23:22:09 Client handling thread created.
23:22:09 ClientID 1 sent back.
23:22:09 Client 1 changed name to
23:22:12 Accepted client 23:22:12 Client handling thread cre
2
23:22:12 ClientID 2 sent back.
23:22:12 Client 2 changed name to
23:26:52 SEND 你好吗 TO 2 FROM 0
23:29:18 BROA 大家好 FROM 1
```

```
D:\env\code\C++\ComputerNetwork\Lab1\advance\client\x64\Debug\cl
23:22:07 Winsock initialised.
23:22:07 Socket created.
23:22:07 Connected. Welcome to the chat room!
23:22:07 Accepted. Your ID is 0
SEND 你好吗 TO 2
23:26:52 ---> 2 : 你好吗
23:29:18 ALL <--- 1 : 大家好
```

```
D:\env\code\C++\ComputerNetwork\Lab1\advance\client\x64\Debug\client.exe
23:22:09 Winsock initialised.
23:22:09 Socket created.
23:22:09 Connected. Welcome to the chat room!
23:22:09 Accepted. Your ID is 1
BROA 大家好
23:29:18 ---> ALL : 大家好
```

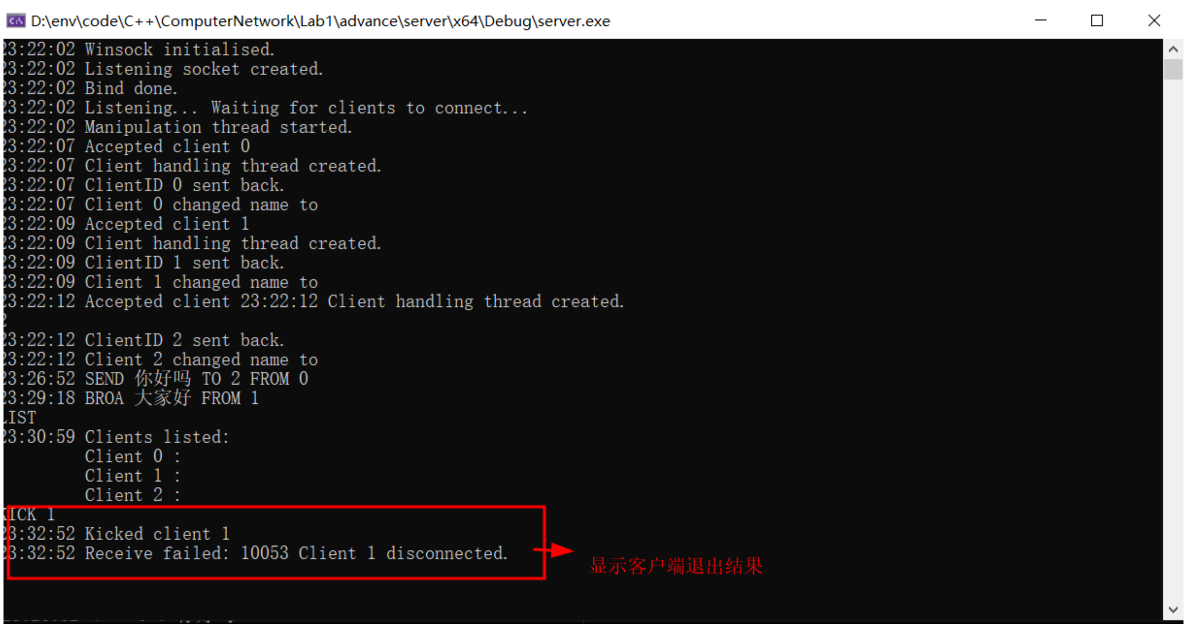
```
D:\env\code\C++\ComputerNetwork\Lab1\advance\client\x64\Debug\client.
23:22:12 Winsock initialised.
23:22:12 Socket created.
23:22:12 Connected. Welcome to the chat room!
23:22:12 Accepted. Your ID is 2
23:26:52 <--- 0 : 你好吗
23:29:18 ALL <--- 1 : 大家好
```

测试列出客户端功能。



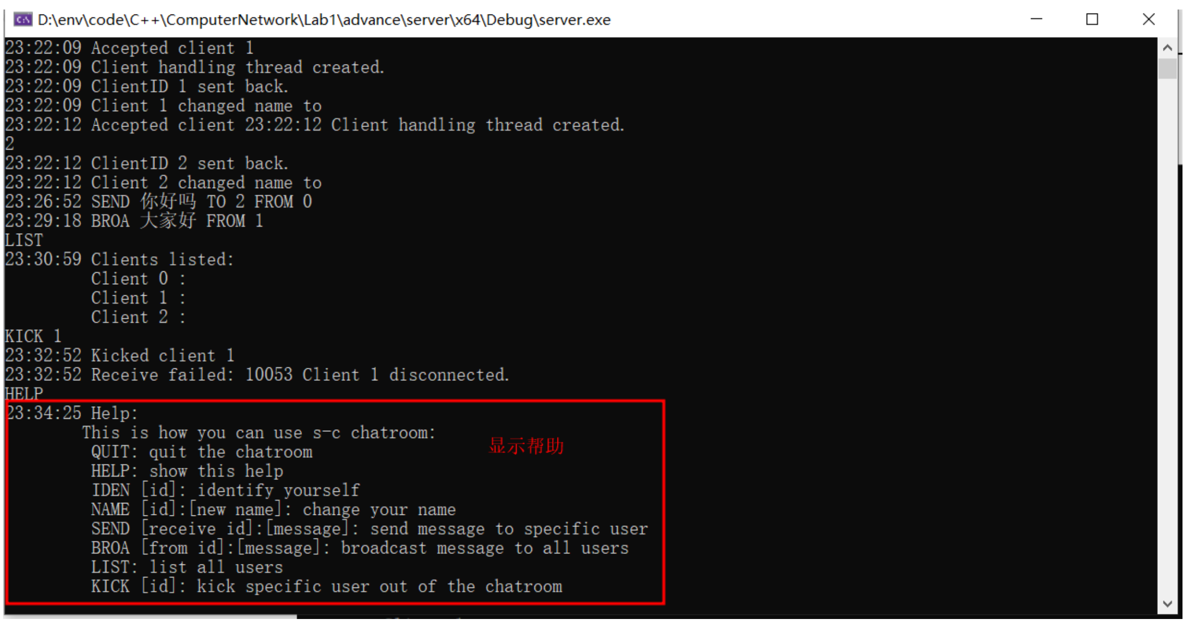
```
D:\env\code\C++\ComputerNetwork\Lab1\advance\server\x64\Debug\server.exe
23:22:02 Winsock initialised.
23:22:02 Listening socket created.
23:22:02 Bind done.
23:22:02 Listening... Waiting for clients to connect...
23:22:02 Manipulation thread started.
23:22:07 Accepted client 0
23:22:07 Client handling thread created.
23:22:07 ClientID 0 sent back.
23:22:07 Client 0 changed name to
23:22:09 Accepted client 1
23:22:09 Client handling thread created.
23:22:09 ClientID 1 sent back.
23:22:09 Client 1 changed name to
23:22:12 Accepted client 23:22:12 Client handling thread created.
2
23:22:12 ClientID 2 sent back.
23:22:12 Client 2 changed name to
23:26:52 SEND 你好吗 TO 2 FROM 0
23:29:18 BROA 大家好 FROM 1
LIST
23:30:59 Clients listed:
         Client 0 :
         Client 1 :
         Client 2 :
```

测试踢人功能。



```
D:\env\code\C++\ComputerNetwork\Lab1\advance\server\x64\Debug\server.exe
3:22:02 Winsock initialised.
3:22:02 Listening socket created.
3:22:02 Bind done.
3:22:02 Listening... Waiting for clients to connect...
3:22:02 Manipulation thread started.
3:22:07 Accepted client 0
3:22:07 Client handling thread created.
3:22:07 ClientID 0 sent back.
3:22:07 Client 0 changed name to
3:22:09 Accepted client 1
3:22:09 Client handling thread created.
3:22:09 ClientID 1 sent back.
3:22:09 Client 1 changed name to
3:22:12 Accepted client 23:22:12 Client handling thread created.

3:22:12 ClientID 2 sent back.
3:22:12 Client 2 changed name to
3:26:52 SEND 你好吗 TO 2 FROM 0
3:29:18 BROA 大家好 FROM 1
IST
3:30:59 Clients listed:
        Client 0 :
        Client 1 :
        Client 2 :
ICK 1
3:32:52 Kicked client 1
3:32:52 Receive failed: 10053 Client 1 disconnected.
```

测试显示帮助功能



```
D:\env\code\C++\ComputerNetwork\Lab1\advance\server\x64\Debug\server.exe
23:22:09 Accepted client 1
23:22:09 Client handling thread created.
23:22:09 ClientID 1 sent back.
23:22:09 Client 1 changed name to
23:22:12 Accepted client 23:22:12 Client handling thread created.
2
23:22:12 ClientID 2 sent back.
23:22:12 Client 2 changed name to
23:26:52 SEND 你好吗 TO 2 FROM 0
23:29:18 BROA 大家好 FROM 1
LIST
23:30:59 Clients listed:
         Client 0 :
         Client 1 :
         Client 2 :
KICK 1
23:32:52 Kicked client 1
23:32:52 Receive failed: 10053 Client 1 disconnected.
HELP
23:34:25 Help:
        This is how you can use s-c chatroom:
         QUIT: quit the chatroom
         HELP: show this help
         IDEN [id]: identify yourself
         NAME [id]:[new name]: change your name
         SEND [receive id]:[message]: send message to specific user
         BROA [from id]:[message]: broadcast message to all users
         LIST: list all users
         KICK [id]: kick specific user out of the chatroom
```