

LAB 3

I. Mục đích	3
II. Yêu cầu	3
III. Hướng dẫn	3
III.1. Xây dựng chương trình UDP Client – Server đơn giản.....	3
III.1.1. Hướng dẫn lập trình UDP Server	3
III.1.2. Hướng dẫn lập trình UDP Client	4
III.1.3. Test chương trình:.....	4
III.1.4. Trả lời câu hỏi:	5
III.2. Bài tập	5
III.3. Cải tiến chương trình UDP client-server để có thể gửi và nhận dữ liệu liên tục	6
III.3.1. Hướng dẫn lập trình UDP Client	6
III.3.2. Hướng dẫn lập trình UDP Server	6
III.3.3. Test chương trình.....	6
III.3.4. Trả lời câu hỏi:	7
III.4. Sử dụng phương thức Connect ở client để thiết lập kết nối trước với server	8
III.4.1. Hướng dẫn lập trình.....	8
III.4.2. Test chương trình.....	8
III.4.3. Trả lời câu hỏi:	9
III.5. Kiểm tra khả năng phân biệt biên thông điệp của giao thức UDP	10
III.5.1. Hướng dẫn lập trình UDP Client	10
III.5.2. Hướng dẫn lập trình UDP Client	10
III.5.3. Test chương trình.....	10

III.5.4. Trả lời câu hỏi:	11
III.6. Ngăn cản mất dữ liệu khi lập trình mạng sử dụng giao thức UDP	12
III.6.1. Trả lời câu hỏi:	12
III.6.2. Hướng dẫn lập trình để ngăn cản mất dữ liệu	12
III.6.3. Test chương trình.....	13
III.6.4. Trả lời câu hỏi:	14
III.7. Ngăn cản mất gói tin khi lập trình mạng sử dụng giao thức UDP.....	14
III.7.1. Hướng dẫn lập trình để ngăn cản mất gói tin	14
III.7.2. Test chương trình.....	17
III.7.3. Trả lời câu hỏi:	18

I. Mục đích

Lập trình client – server sử dụng giao thức UDP

Nắm được các lỗi thường xảy ra khi lập trình theo giao thức UDP và cách hạn chế các lỗi này

II. Yêu cầu

1) Xây dựng chương trình UDP Client – Server đơn giản (xem hướng dẫn)

III. Hướng dẫn

III.1. Xây dựng chương trình UDP Client – Server đơn giản

III.1.1. Hướng dẫn lập trình UDP Server

Để lập trình socket ta sử dụng 2 namespace:

```
using System.Net;  
using System.Net.Sockets;
```

Tạo Server EndPoint, EndPoint này sẽ tham chiếu đến địa chỉ IP và Port của Server:

```
EndPoint serverEndPoint = new EndPoint(IPAddress.Parse("127.0.0.1"), 5000);
```

Tạo Server Socket, Socket này dùng để trao đổi dữ liệu với client

```
Socket serverSocket = new Socket(AddressFamily.InterNetwork, SocketType.Dgram,  
ProtocolType.Udp);
```

Chương trình UDP Server khác với chương trình TCP Server ở chỗ nó không lắng nghe kết nối, trên socket ta chỉ việc Bind nó với Server EndPoint

```
serverSocket.Bind(serverEndPoint);
```

Khi client kết nối tới nó sẽ hiển thị thông tin của client đang kết nối đến:

```
Console.WriteLine(remote.ToString());
```

Để nhận dữ liệu từ client gửi lên ta dùng hàm `ReceiveFrom` với chú ý `EndPoint` chứa thông tin của client kết nối đến phải được truyền tham chiếu

```
serverSocket.ReceiveFrom(buff, 0, buff.Length, SocketFlags.None, ref remote);
```

III.1.2. Hướng dẫn lập trình UDP Client

Tạo Server Socket, Socket này sẽ được dùng để gửi dữ liệu tới Server

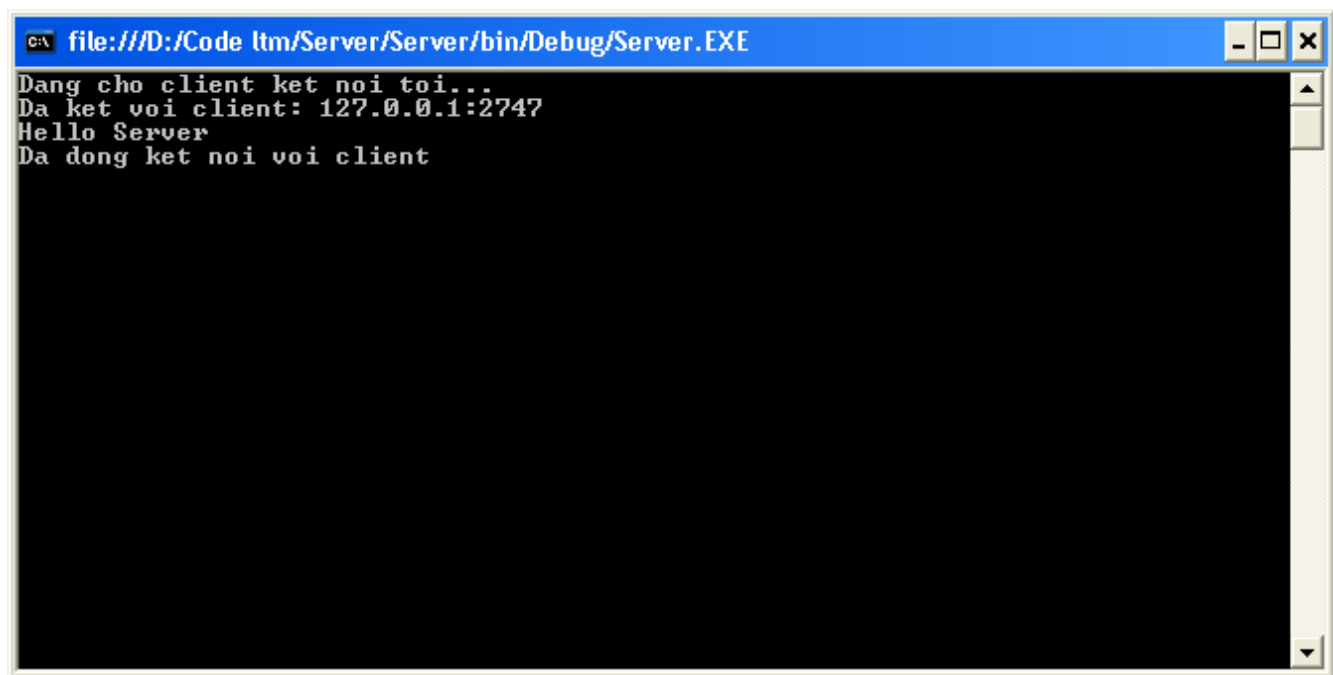
```
Socket server = new Socket(AddressFamily.InterNetwork, SocketType.Dgram,  
ProtocolType.Udp);
```

Gửi câu chào lên server, câu chào này sẽ được đặt trong mảng `buff`

```
server.SendTo(buff, buff.Length, SocketFlags.None, serverEndPoint);
```

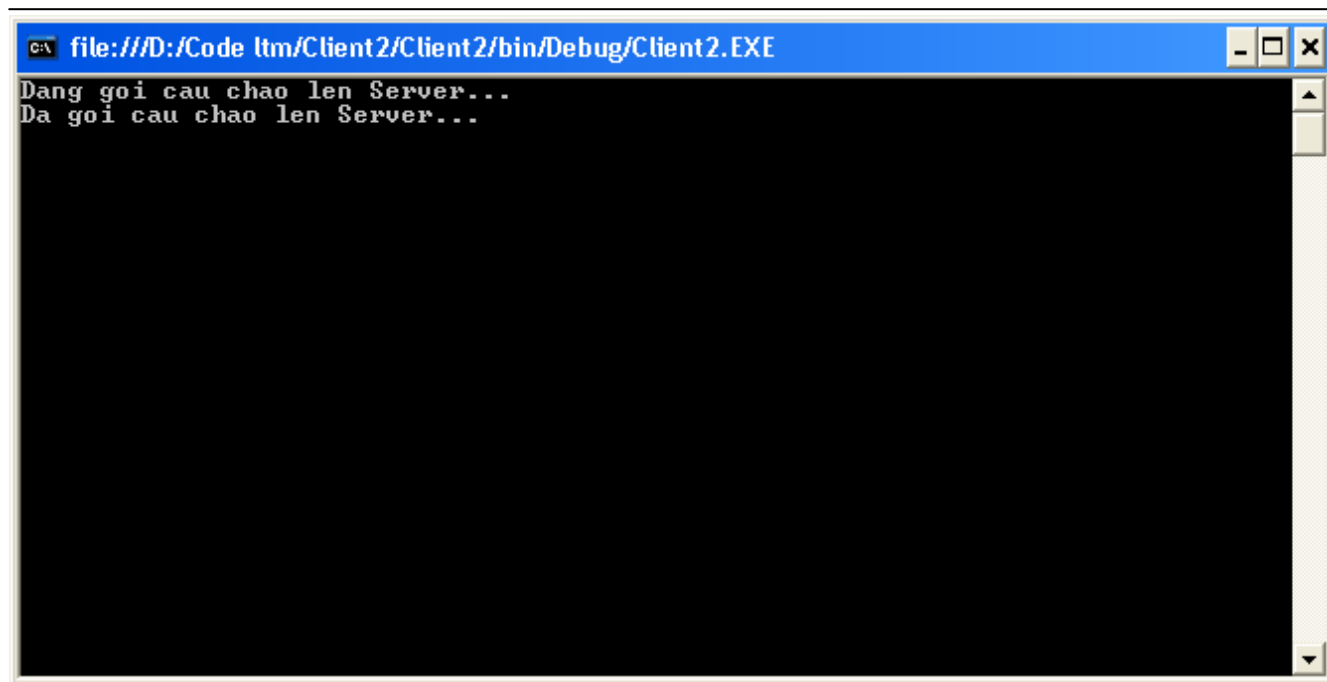
III.1.3. Test chương trình:

Chạy chương trình Server:

A screenshot of a Windows command prompt window. The title bar shows the file path: "file:///D:/Code Itm/Server/Server/bin/Debug/Server.EXE". The window contains the following text:

```
Dang cho client ket noi toi...  
Da ket voi client: 127.0.0.1:2747  
Hello Server  
Da dong ket noi voi client
```

Chạy chương trình ở client:



III.1.4. Trả lời câu hỏi:

- 1) Như hình trên port 2747 ở đâu ra

.....
.....

- 2) Có phải lúc nào client cũng mở port 2747 để kết nối với Server không ?

.....
.....

- 3) Tại sao khi lập trình mạng dùng giao thức UDP thì client phải gửi câu chào lên server trước?

.....
.....

III.2. Bài tập

- 1) Cải tiến chương trình để ở client gõ “exit” thì đóng client, khi client gõ “exit all” thì đóng cả client và server.

III.3. Cải tiến chương trình UDP client-server để có thể gửi và nhận dữ liệu liên tục

III.3.1. Hướng dẫn lập trình UDP Client

Dùng vòng lặp vô hạn để khi người dùng nhập dữ liệu xong thì sẽ gửi lên server và chờ server gửi kết quả trả về, lấy kết quả đó hiển thị lên màn hình

```
while (true)
{
    str = Console.ReadLine();
    buff = Encoding.ASCII.GetBytes(str);
    serverSocket.SendTo(buff, 0, buff.Length, SocketFlags.None, remote);
    byteReceive = serverSocket.ReceiveFrom(buff, 0, buff.Length, SocketFlags.None,
ref remote);
    str = Encoding.ASCII.GetString(buff, 0, byteReceive);
    Console.WriteLine(str);
}
```

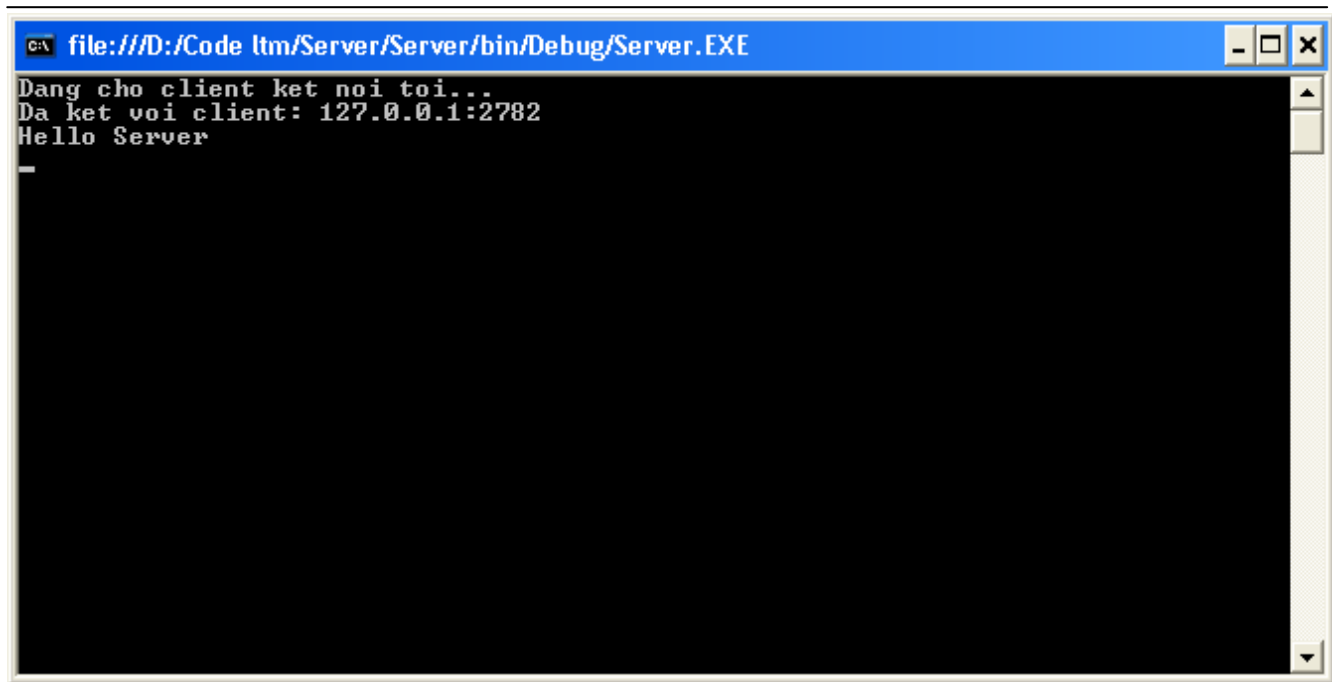
III.3.2. Hướng dẫn lập trình UDP Server

Dùng vòng lặp vô hạn lấy kết quả client gửi lên, chuyển nó thành chuỗi và hiển thị lên màn hình đồng thời gửi lại dữ liệu nhận được về lại cho client

```
while (true)
{
    buff = new byte[1024];
    byteReceive = serverSocket.ReceiveFrom(buff, 0, buff.Length, SocketFlags.None,
ref remote);
    str = Encoding.ASCII.GetString(buff, 0, byteReceive);
    Console.WriteLine(str);
    serverSocket.SendTo(buff, 0, buff.Length, SocketFlags.None, remote);
}
```

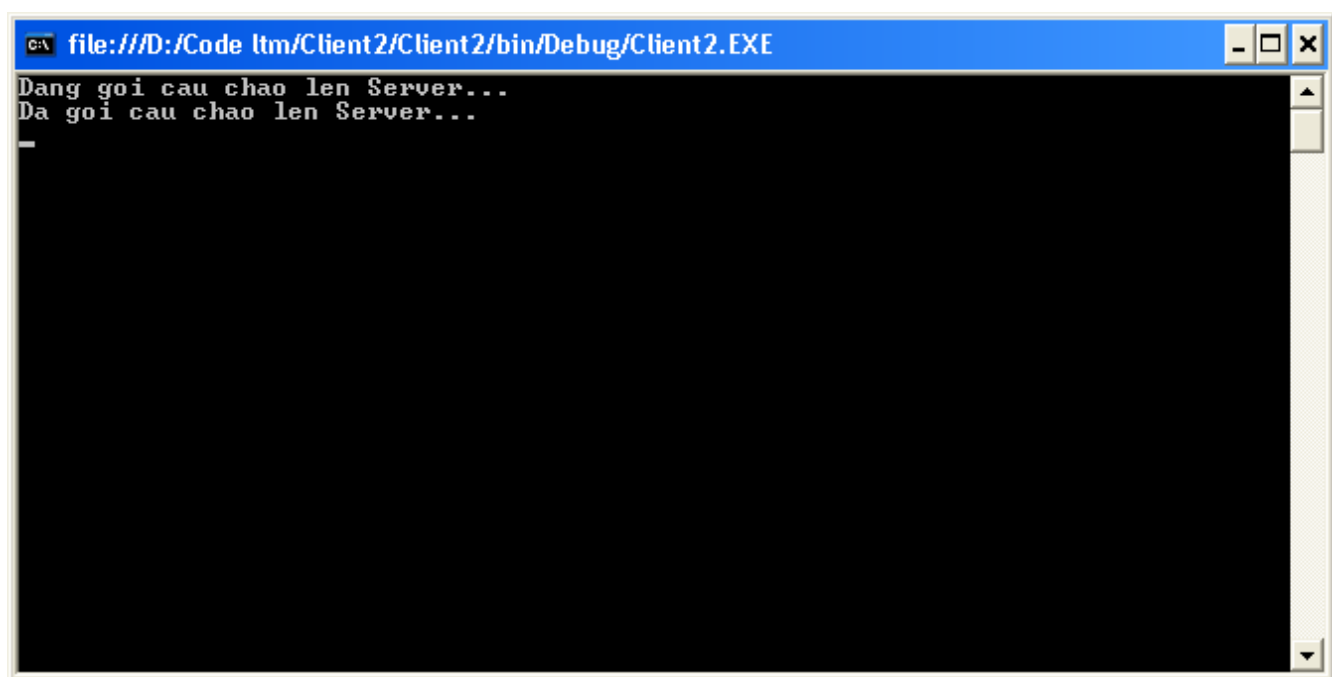
III.3.3. Test chương trình

Chạy chương trình ở server:



```
file:///D:/Code Itm/Server/Server/bin/Debug/Server.EXE
Dang cho client ket noi toi...
Da ket voi client: 127.0.0.1:2782
Hello Server
```

Chạy chương trình ở client:



```
file:///D:/Code Itm/Client2/Client2/bin/Debug/Client2.EXE
Dang goi cau chao len Server...
Da goi cau chao len Server...
```

Nhập thông điệp từ client và gửi lên server, lỗi sẽ xảy ra, hãy sửa lại cho hết lỗi

III.3.4. Trả lời câu hỏi:

- 1) Khi chạy chương trình với đoạn code gửi nhận dữ liệu như trên, lúc chưa nhập dữ liệu cho client để gửi lên thì không xảy ra lỗi nhưng khi nhập dữ liệu để gửi lên server sẽ xảy ra lỗi, vì sao lại xảy ra lỗi này ?

.....

.....

- 2) Khi server chưa bật thì chương trình trên có bị lỗi không? Tạo sao

.....

.....

- 3) Khi đang chạy chương trình tắt client thì chương trình trên có bị lỗi không? Tại sao

.....

.....

III.4. Sử dụng phương thức Connect ở client để thiết lập kết nối trước với server

III.4.1. Hướng dẫn lập trình

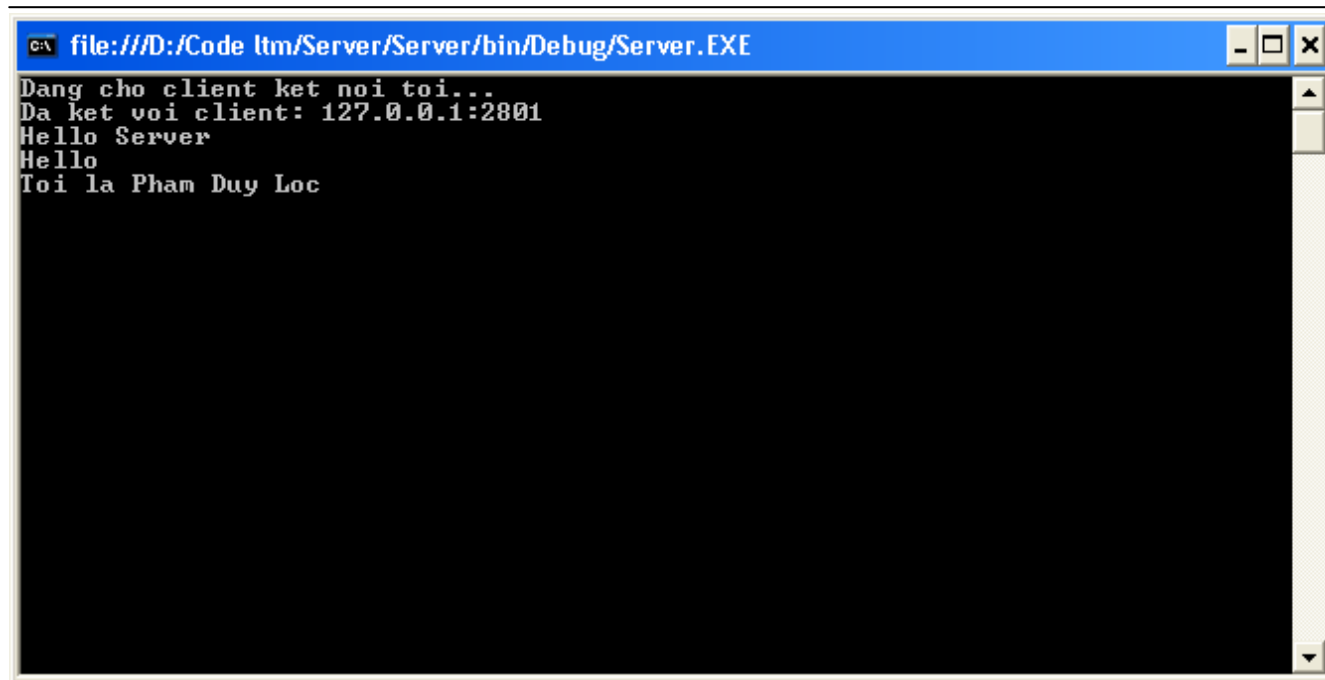
Việc này được thực hiện bằng cách sử dụng phương thức Connect() ở lớp socket

```
serverSocket.Connect(remote);
```

Sau khi gọi phương thức Connect() xong, ta có thể lập trình giống như lập trình với giao thức UDP với client

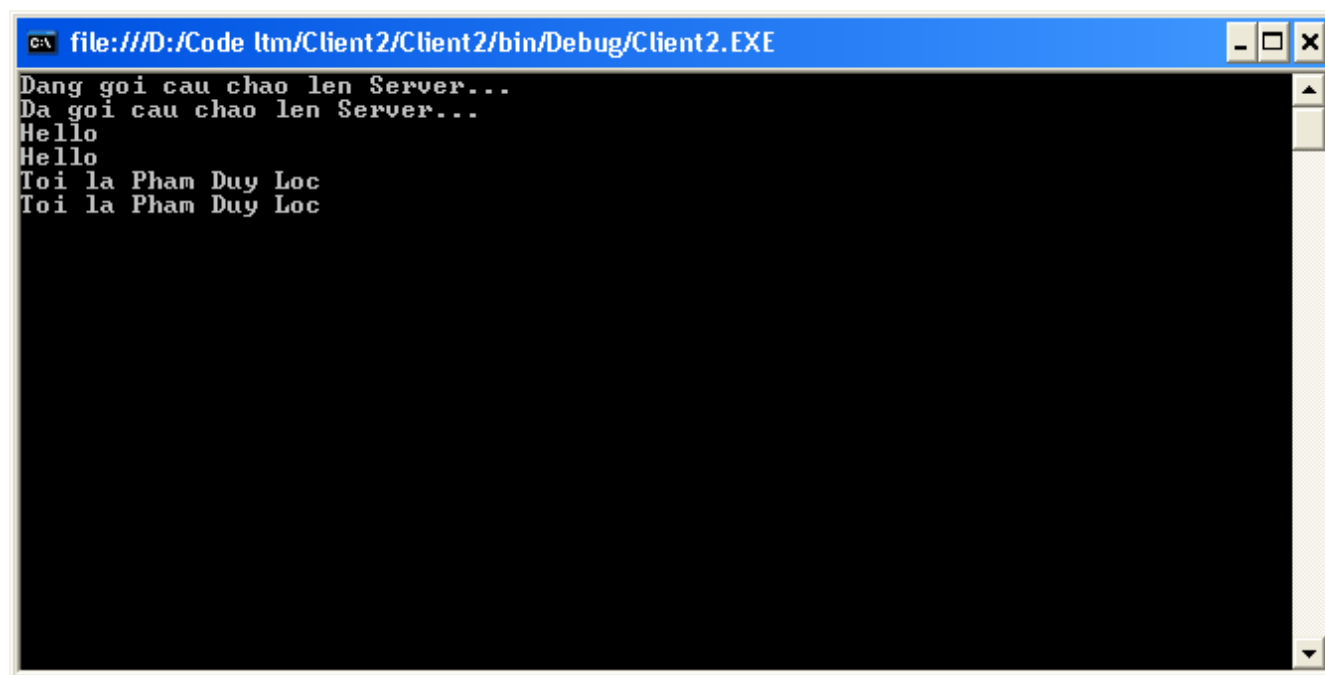
III.4.2. Test chương trình

Chạy chương trình ở server:



```
file:///D:/Code ltm/Server/Server/bin/Debug/Server.EXE
Dang cho client ket noi toi...
Da ket voi client: 127.0.0.1:2801
Hello Server
Hello
Toi la Pham Duy Loc
```

Chạy chương trình ở client:



```
file:///D:/Code ltm/Client2/Client2/bin/Debug/Client2.EXE
Dang goi cau chao len Server...
Da goi cau chao len Server...
Hello
Hello
Toi la Pham Duy Loc
Toi la Pham Duy Loc
```

III.4.3. Trả lời câu hỏi:

- 1) Khi chạy chương trình mà server chưa được bật thì có hiện tượng gì xảy ra? Tại sao lại có hiện tượng này?

Khi server chưa được bật thì client gửi dữ liệu đi thì không nhận được phản hồi. Vì không có server để nhận dữ liệu từ client và phản hồi lại

.....

2) Khi đang chạy tắt server thì chương trình trên có bị lỗi không? Tạo sao

Chương trình không lỗi. Vì client vẫn có thể gửi dữ liệu lên server nhưng sẽ không được phản hồi

.....

3) Khi đang chạy chương trình tắt client thì chương trình trên có bị lỗi không? Tại sao

Không lỗi vì UDP không cần thiết lập kết nối

.....

III.5. Kiểm tra khả năng phân biệt biên thông điệp của giao thức UDP

III.5.1. Hướng dẫn lập trình UDP Client

Thay đoạn code gửi và nhận thông điệp vô hạn bằng đoạn code gửi 5 thông điệp phân biệt lên server

```
for (int i = 1; i <= 5; i++)
{
    buff = Encoding.ASCII.GetBytes("Thong Diep " + i.ToString());
    serverSocket.SendTo(buff, 0, buff.Length, SocketFlags.None, remote);
}
```

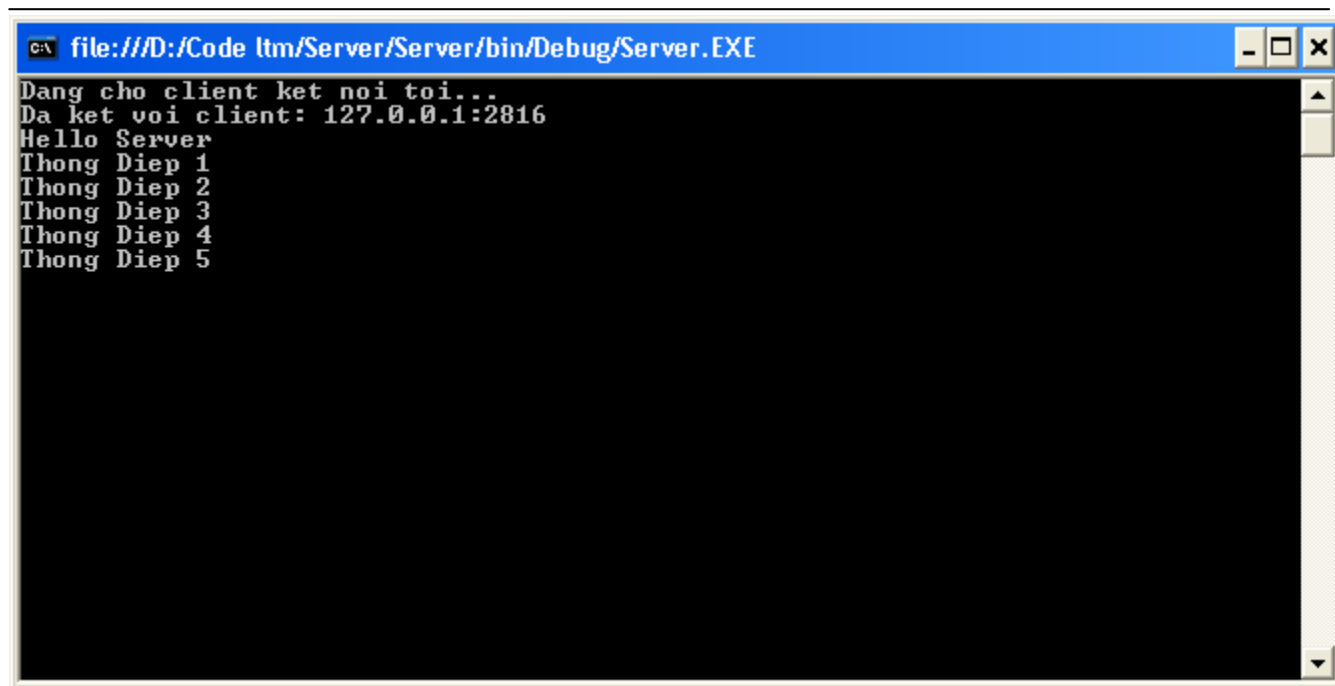
III.5.2. Hướng dẫn lập trình UDP Server

Thay đoạn code gửi và nhận thông điệp vô hạn bằng đoạn code nhận 5 thông điệp trên server

```
for (int i = 1; i <= 5; i++)
{
    byteReceive = serverSocket.ReceiveFrom(buff, 0, buff.Length, SocketFlags.None,
    ref remote);
    str = Encoding.ASCII.GetString(buff, 0, byteReceive);
    Console.WriteLine(str);
}
```

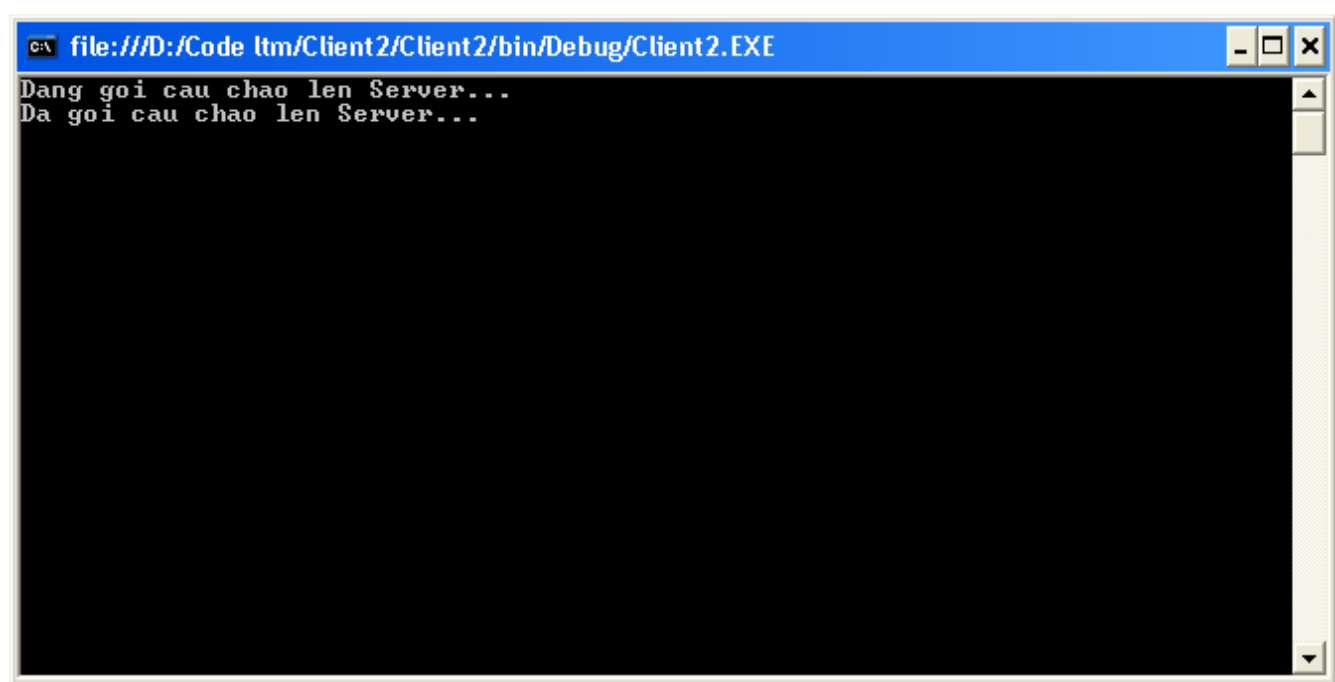
III.5.3. Test chương trình

Chạy chương trình server:



```
file:///D:/Code ltm/Server/Server/bin/Debug/Server.EXE
Dang cho client ket noi toi...
Da ket voi client: 127.0.0.1:2816
Hello Server
Thong Diep 1
Thong Diep 2
Thong Diep 3
Thong Diep 4
Thong Diep 5
```

Chạy chương trình client:



```
file:///D:/Code ltm/Client2/Client2/bin/Debug/Client2.EXE
Dang goi cau chao len Server...
Da goi cau chao len Server...
```

III.5.4. Trả lời câu hỏi:

- 1) Tại sao khi lập trình bằng giao thức UDP các thông điệp được phân biệt với nhau ?

Tại vì trong giao thức UDP khi client gửi dữ liệu từ 1 hàm Send thì Server sẽ nhận lại dữ liệu bằng đúng 1 hàm Recive nên nó có thể phân biệt được các lần gửi tin

III.6. Ngăn cản mất dữ liệu khi lập trình mạng sử dụng giao thức UDP

Trong đoạn code gửi và nhận thông điệp của chương trình UDP Client đơn giản ở trên ta thay đổi kích thước bộ đệm như sau:

```
while (true)
{
    str = Console.ReadLine();
    buff = Encoding.ASCII.GetBytes(str);
    serverSocket.SendTo(buff, 0, buff.Length, SocketFlags.None, remote);
    buff = new byte[10];
    byteReceive = serverSocket.ReceiveFrom(buff, 0, buff.Length, SocketFlags.None,
ref remote);
    str = Encoding.ASCII.GetString(buff, 0, byteReceive);
    Console.WriteLine(str);
}
```

III.6.1. Trả lời câu hỏi:

- 1) Khi gửi dữ liệu với kích thước lớn hơn 10 byte thì có lỗi xảy ra không? Tại sao?

Sẽ sinh ra lỗi ở hàm Recive khi nó nhận dữ liệu lớn hơn 10 byte mà mảng nhận dữ liệu chỉ có 10 byte sẽ có lỗi

III.6.2. Hướng dẫn lập trình để ngăn cản mất dữ liệu

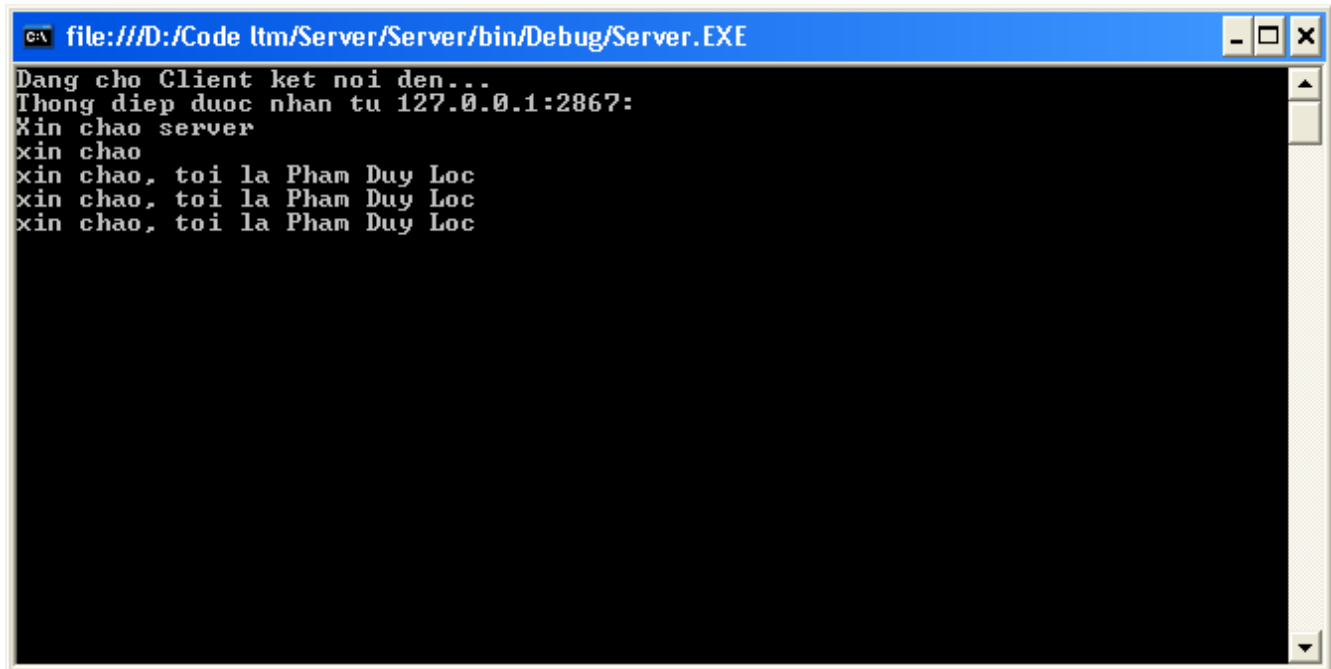
Thay đoạn code gửi và nhận thông điệp vô hạn bằng đoạn code gửi 5 thông điệp phân biệt lên server

```
while (true)
{
    input = Console.ReadLine();
    if (input == "exit")
        break;
    server.SendTo(Encoding.ASCII.GetBytes(input), tmpRemote);
    data = new byte[i];
    try
    {
        recv = server.ReceiveFrom(data, ref tmpRemote);
        stringData = Encoding.ASCII.GetString(data, 0, recv);
        Console.WriteLine(stringData);
    }
    catch (SocketException)
    {
    }
```

```
        Console.WriteLine("Canh bao: du lieu bi mat, hay thu lai");  
        i += 10;  
    }  
}
```

III.6.3. Test chương trình

Chạy chương trình server:



The screenshot shows a Windows command prompt window with the title bar "file:///D:/Code ltm/Server/Server/bin/Debug/Server.EXE". The window contains the following text:

```
Dang cho Client ket noi den...  
Thong diep duoc nhan tu 127.0.0.1:2867:  
Xin chao server  
xin chao  
xin chao, toi la Pham Duy Loc  
xin chao, toi la Pham Duy Loc  
xin chao, toi la Pham Duy Loc
```

Chạy chương trình client:

```

file:///D:/Code ltm/Client2/Client2/bin/Debug/Client2.EXE
Thông điệp được nhận từ 127.0.0.1:50000:
Hello Client
xin chao
xin chao
xin chao, toi la Pham Duy Loc
Canh bao: du lieu bi mat, hay thu lai
xin chao, toi la Pham Duy Loc
Canh bao: du lieu bi mat, hay thu lai
xin chao, toi la Pham Duy Loc
xin chao, toi la Pham Duy Loc

```

III.6.4. Trả lời câu hỏi:

- 2) Khi dùng phương pháp này các dữ liệu bị mất do kích thước bộ đệm ban đầu nhỏ có lấy lại được không?

Không vì UDP không thiết lập kết nối nên không đảm bảo gói tin truyền đầy đủ và có đến đích hay không

.....

III.7. Ngăn cản mất gói tin khi lập trình mạng sử dụng giao thức UDP

III.7.1. Hướng dẫn lập trình để ngăn cản mất gói tin

Khi gửi dữ liệu bằng giao thức UDP với các chương trình UDP đơn giản ở trên, nếu dữ liệu không đến được tới đích vì một lý do nào đó thì không thể nào biết được gói tin gửi đi đã bị mất.

Để ngăn cản mất gói tin ta dùng phương thức `SetSocketOption()` để thiết lập giá trị `Timeout` để sau một thời gian không nhận được hồi báo thì gửi lại dữ liệu

```
server.SetSocketOption(SocketOptionLevel.Socket, SocketOptionName.ReceiveTimeout, 3000);
```

Và viết lại hàm gửi và nhận dữ liệu:

```
private int SndRcvData(Socket s, byte[] message, EndPoint rmtdevice)
```

```
{  
    int recv;  
    int retry = 0;  
    while (true)  
    {  
        Console.WriteLine("Truyen lai lan thu: #{0}", retry);  
        try  
        {  
            s.SendTo(message, message.Length, SocketFlags.None, rmtdevice);  
            data = new byte[1024];  
            recv = s.ReceiveFrom(data, ref Remote);  
        }  
        catch (SocketException)  
        {  
            recv = 0;  
        }  
        if (recv > 0)  
        {  
            return recv;  
        }  
        else  
        {  
            retry++;  
            if (retry > 4)  
            {  
                return 0;  
            }  
        }  
    }  
}
```

Viết lại class điều khiển việc gửi và nhận dữ liệu ngăn cản mất gói tin

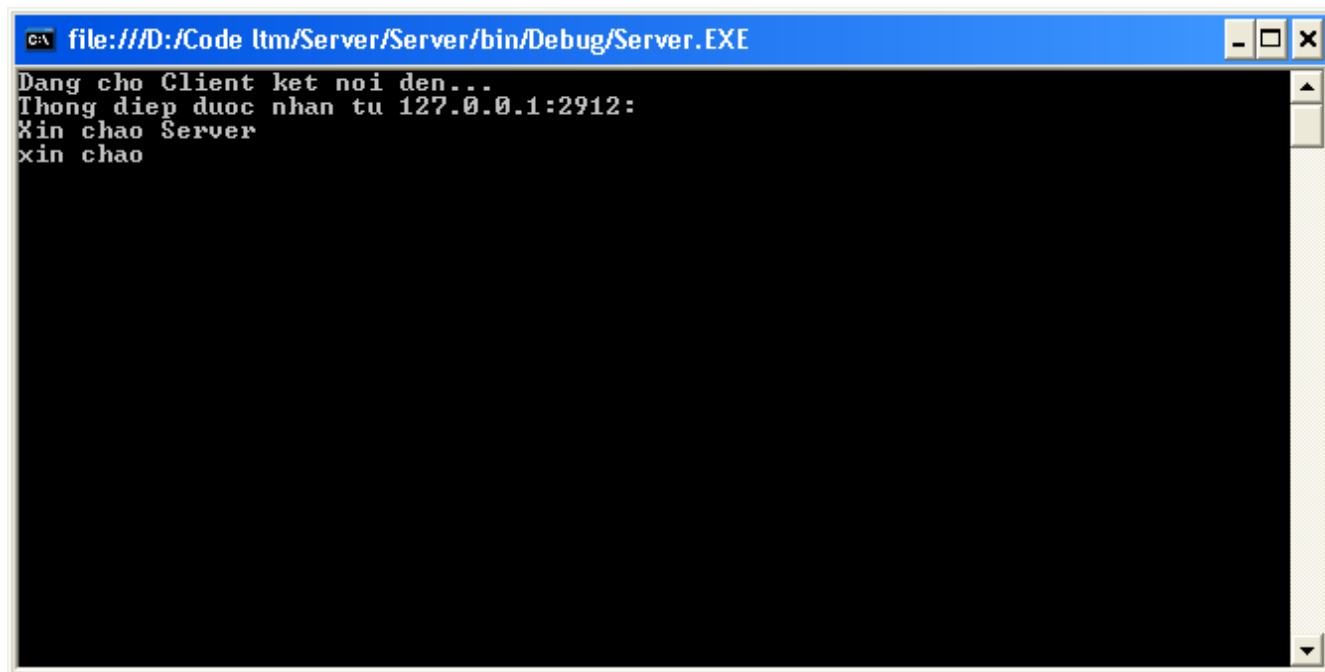
```
public RetryUdpClient()  
{  
    string input, stringData;  
    int recv;  
    IPEndPoint ipep = new IPEndPoint(IPAddress.Parse("127.0.0.1"), 5000);  
    Socket server = new Socket(AddressFamily.InterNetwork,
```

```
        SocketType.Dgram, ProtocolType.Udp);
int sockopt = (int)server.GetSocketOption(SocketOptionLevel.Socket,
    SocketOptionName.ReceiveTimeout);
Console.WriteLine("Gia tri timeout mac dinh: {0}", sockopt);
server.SetSocketOption(SocketOptionLevel.Socket,
    SocketOptionName.ReceiveTimeout, 3000);
sockopt = (int)server.GetSocketOption(SocketOptionLevel.Socket,
    SocketOptionName.ReceiveTimeout);
Console.WriteLine("Gia tri timeout moi: {0}", sockopt);
string welcome = "Xin chao Server";
data = Encoding.ASCII.GetBytes(welcome);
recv = SndRcvData(server, data, ipep);
if (recv > 0)
{
    stringData = Encoding.ASCII.GetString(data, 0, recv);
    Console.WriteLine(stringData);
}
else
{
    Console.WriteLine("Khong the lien lac voi thiet bi o xa");
    return;
}
while (true)
{
    input = Console.ReadLine();
    if (input == "exit")
        break;
    recv = SndRcvData(server, Encoding.ASCII.GetBytes(input), ipep);
    if (recv > 0)
    {
        stringData = Encoding.ASCII.GetString(data, 0, recv);
        Console.WriteLine(stringData);
    }
    else
        Console.WriteLine("Khong nhan duoc cau tra loi");
}
Console.WriteLine("Dang dong client");
server.Close();
}
```

III.7.2. Test chương trình

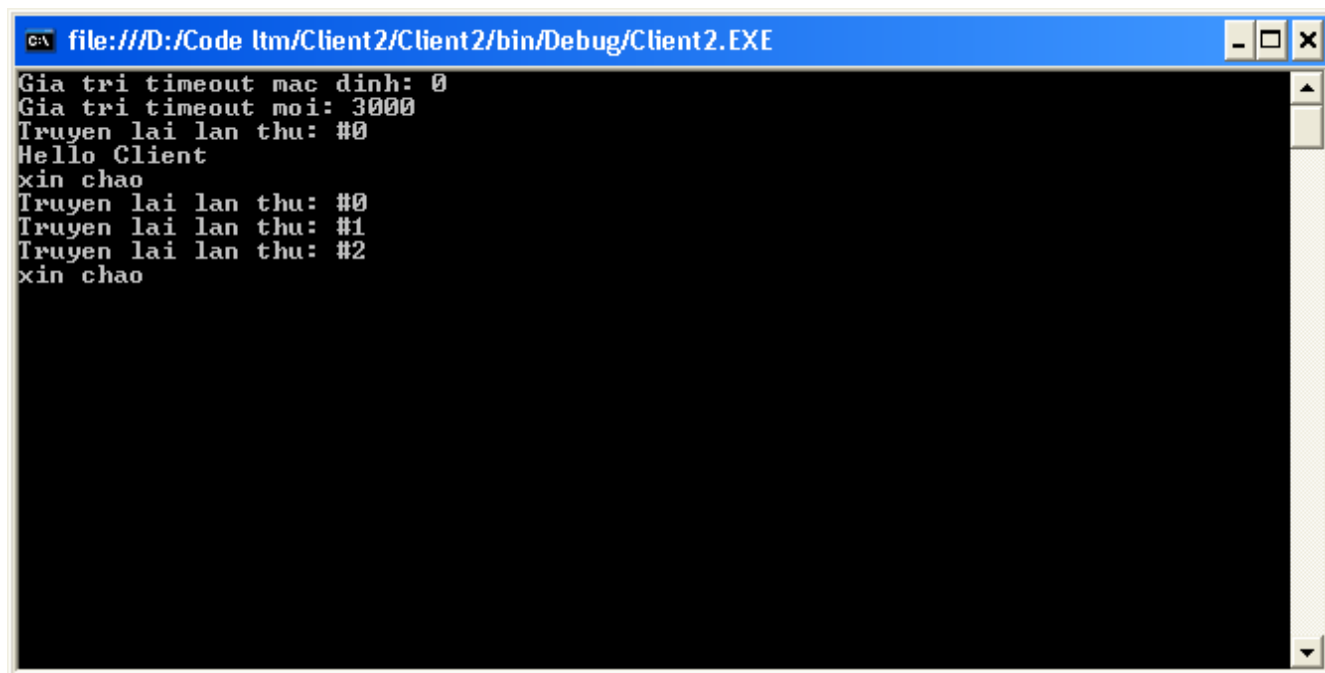
Để test chương trình ta dùng hàm Sleep() trong vòng lặp While ở server để giả lập lỗi

Chạy chương trình server:



```
file:///D:/Code ltm/Server/Server/bin/Debug/Server.EXE
Dang cho Client ket noi den...
Thong diep duoc nhan tu 127.0.0.1:2912:
Xin chao Server
xin chao
```

Chạy chương trình client:



```
file:///D:/Code ltm/Client2/Client2/bin/Debug/Client2.EXE
Gia tri timeout mac dinh: 0
Gia tri timeout moi: 3000
Truyen lai lan thu: #0
Hello Client
xin chao
Truyen lai lan thu: #0
Truyen lai lan thu: #1
Truyen lai lan thu: #2
xin chao
```

III.7.3. Trả lời câu hỏi:

- 3) Sử dụng phương pháp này có ngăn cản triệt để mất dữ liệu không? Nếu không hãy đề xuất phương pháp khác

.....

.....