Cho đối tượng Sách gồm những thuộc tính được mô tả như sau:

- Mã sách (kiểu số nguyên)
- Tên sách (kiểu chuỗi)
- Giá tiền (kiểu số thực)

Câu 1: Định nghĩa lớp mô tả đối tượng Sách?

Câu 2: Viết các phương thức chuyển đổi tượng Sách thành mảng các byte và ngược lại?

Câu 3: Viết chương trình (Console App) Client – Server, trong đó:

- Phía Client cho phép người dùng nhập thông tin sách và gửi đến Server.
- Phía Server sẽ xuất thông tin và số byte dữ liệu của sách nhận được.
- Khi nhập xong thông tin sách hỏi người dùng có tiếp tục nhập không, nếu trả lời "khong" thì dừng nhập.

Câu 4: Ghi thông tin tất cả Sách nhận được từ Client vào file "ThongTinSach.txt"

Client

```
using De1.Class;
using System;
using System.Net.Sockets;
using System.Collections.Generic;

namespace De1
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.OutputEncoding = System.Text.Encoding.UTF8;

            TcpClient client;
            try
            {
                client = new TcpClient("127.0.0.1", 6000);
            }
            catch (SocketException)
            {
                Console.WriteLine("Khong ket noi duoc voi server");
                return;
            }

            book itemBook;
            List<book> listBook = new List<book>();

            string kq = "";
            do
            {
                itemBook = new book();
                itemBook.addBookInfo();
                listBook.Add(itemBook);

                Console.Write("\n>>> Bạn có muốn tiếp tục nhập ko? >>> ");
                kq = Console.ReadLine().Trim();
            } while (kq != "khong");
```

```
            byte[] listSize = new byte[2];
            listSize = BitConverter.GetBytes(listBook.Count);

            NetworkStream ns = client.GetStream();
            ns.Write(listSize, 0, 2);

            foreach (var item in listBook)
            {
                byte[] data = item.GetBytes();
                int size = item.size;
                byte[] packetSize = new byte[2];
                Console.WriteLine("Kich thuoc goi tin = {0}", size);
                packetSize = BitConverter.GetBytes(size);
                ns.Write(packetSize, 0, 2);
                ns.Write(data, 0, size);
                ns.Flush();
            }
            Console.WriteLine(">>> Đã gửi thông tin sách đến server!");
            Console.WriteLine(">>> Tổng số gói tin đã gửi: {0}", listBook.Count);
            ns.Close();

            Console.ReadKey();
            client.Close();

        }


    }
}
Class book.cs của Client
using System;
using System.Collections.Generic;
using System.IO;
using System.Text;

namespace De1.Class
{
    public class book
    {
        public int bookID;
        private int nameSize;
        public string name;
        public double price;
        public int size;

        public byte[] GetBytes()
        {
            byte[] data = new byte[1024];
            int place = 0;
            Buffer.BlockCopy(BitConverter.GetBytes(bookID), 0, data, place, 4);
            place += 4;
            Buffer.BlockCopy(BitConverter.GetBytes(name.Length), 0, data, place, 4);
            place += 4;
            Buffer.BlockCopy(Encoding.ASCII.GetBytes(name), 0, data, place, name.Length);
            place += name.Length;
            Buffer.BlockCopy(BitConverter.GetBytes(price), 0, data, place, 8);
            place += 8;
            size = place;
            return data;
        }

        public book()
        {
```

```
        }

        public book(byte[] data)
        {
            int place = 0;
            bookID = BitConverter.ToInt32(data, place);
            place += 4;
            nameSize = BitConverter.ToInt32(data, place);
            place += 4;
            name = Encoding.ASCII.GetString(data, place, nameSize);
            place = place + nameSize;
            price = BitConverter.ToDouble(data, place);
        }

        public void addBookInfo()
        {
            Console.WriteLine(">>> Mời nhập thông tin sách");
            Console.Write("+ Nhập mã sách: ");
            this.bookID = Int32.Parse(Console.ReadLine().Trim());
            Console.Write("+ Nhập tên sách: ");
            this.name = Console.ReadLine().Trim();
            Console.Write("+ Nhập giá sách: ");
            this.price = Double.Parse(Console.ReadLine().Trim());
        }

        public void addToFile(string filename, int packSize)
        {
            File.AppendAllText(filename, packSize.ToString() + ",");
            File.AppendAllText(filename, this.bookID.ToString() + ",");
            File.AppendAllText(filename, this.name + ",");
            File.AppendAllText(filename, this.price.ToString());
            File.AppendAllText(filename, "\n");
        }

    }
}
```

Server

```
using Server.Class;
using System;
using System.IO;
using System.Net;
using System.Net.Sockets;

namespace Server
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.OutputEncoding = System.Text.Encoding.UTF8;
            Console.WriteLine("Đang chờ client gửi thông tin...");

            TcpListener server = new TcpListener(IPAddress.Any, 6000);
            server.Start();
            TcpClient client = server.AcceptTcpClient();

            NetworkStream ns = client.GetStream();

            byte[] total = new byte[2];
            int recv = ns.Read(total, 0, 2);
            int totalItem = BitConverter.ToInt16(total, 0);
```

```
            Console.WriteLine("Số gói tin nhận được: {0}", totalItem);

            string filename = "ThongTinSach.txt";
            File.WriteAllText(filename, "");

            Console.WriteLine(">>> Kết quả");
            for (int i = 0; i < totalItem; i++)
            {
                Console.WriteLine("+ Thông tin sách thứ {0}:", i + 1);

                byte[] data = new byte[1024];
                byte[] size = new byte[2];
                recv = ns.Read(size, 0, 2);
                int packSize = BitConverter.ToInt16(size, 0);

                Console.WriteLine("Kích thước gói tin: {0}", packSize);
                recv = ns.Read(data, 0, packSize);

                book itemBook = new book(data);
                Console.WriteLine("Mã sách: {0}", itemBook.bookID);
                Console.WriteLine("Tên sách: {0}", itemBook.name);
                Console.WriteLine("Giá tiền: {0}\n", itemBook.price);

                itemBook.addToFile(filename, packSize);
            }

            Console.WriteLine("Đã ghi tất cả thông tin sách vào file ThongTinSach.txt!");
            Console.ReadKey();
            ns.Close();
            client.Close();
            server.Stop();
        }
    }
}
```

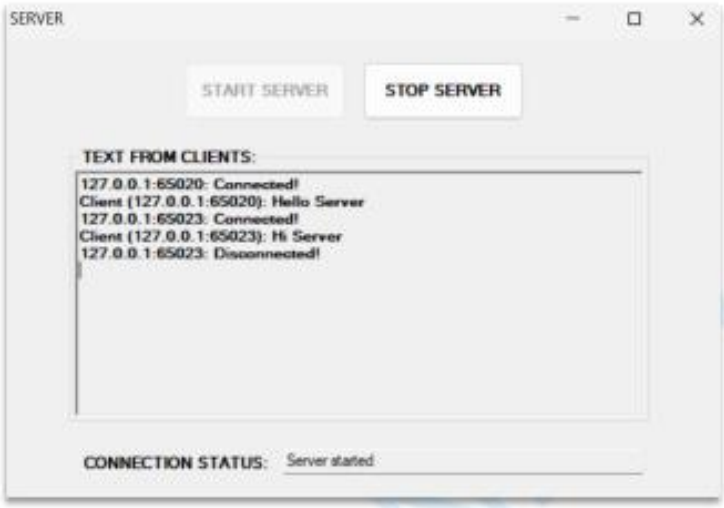
Class book.cs của Server

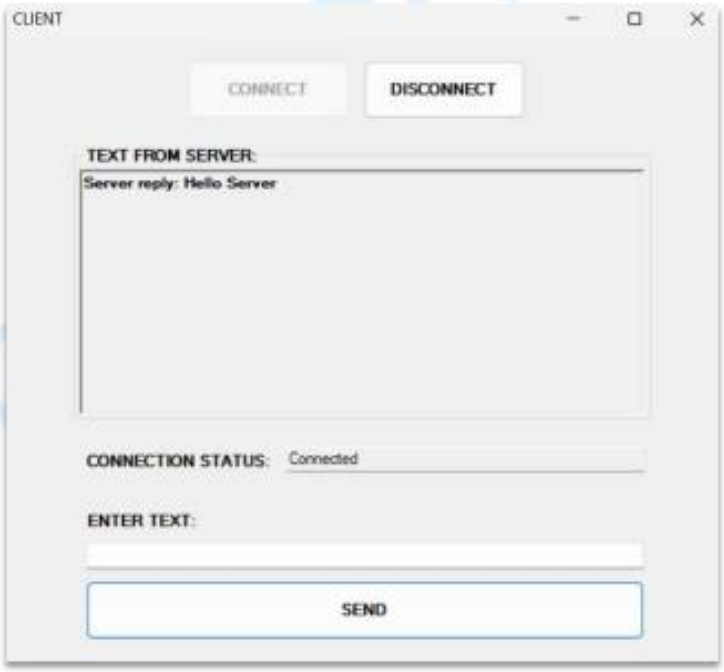
```
using System;
using System.Collections.Generic;
using System.IO;
using System.Text;

namespace Server.Class
{
    public class book
    {
        public int bookID;
        private int nameSize;
        public string name;
        public double price;
        public int size;

        public byte[] GetBytes()
        {
            byte[] data = new byte[1024];
            int place = 0;
            Buffer.BlockCopy(BitConverter.GetBytes(bookID), 0, data, place, 4);
            place += 4;
            Buffer.BlockCopy(BitConverter.GetBytes(name.Length), 0, data, place, 4);
            place += 4;
            Buffer.BlockCopy(Encoding.ASCII.GetBytes(name), 0, data, place, name.Length);
            place += name.Length;
            Buffer.BlockCopy(BitConverter.GetBytes(price), 0, data, place, 8);
            place += 8;
            size = place;
            return data;
```

```
        }

        public book()
        {

        }

        public book(byte[] data)
        {
            int place = 0;
            bookID = BitConverter.ToInt32(data, place);
            place += 4;
            nameSize = BitConverter.ToInt32(data, place);
            place += 4;
            name = Encoding.ASCII.GetString(data, place, nameSize);
            place = place + nameSize;
            price = BitConverter.ToDouble(data, place);
        }

        public void addBookInfo()
        {
            Console.WriteLine(">>> Mời nhập thông tin sách");
            Console.Write("+ Nhập mã sách: ");
            this.bookID = Int32.Parse(Console.ReadLine().Trim());
            Console.Write("+ Nhập tên sách: ");
            this.name = Console.ReadLine().Trim();
            Console.Write("+ Nhập giá sách: ");
            this.price = Double.Parse(Console.ReadLine().Trim());
        }

        public void addToFile(string filename, int packSize)
        {
            File.AppendAllText(filename, packSize.ToString() + ",");
            File.AppendAllText(filename, this.bookID.ToString() + ",");
            File.AppendAllText(filename, this.name + ",");
            File.AppendAllText(filename, this.price.ToString());
            File.AppendAllText(filename, "\n");
        }

    }
}
```

ĐỀ ÔN THI MÔN LẬP TRÌNH MẠNG

Đề số 2

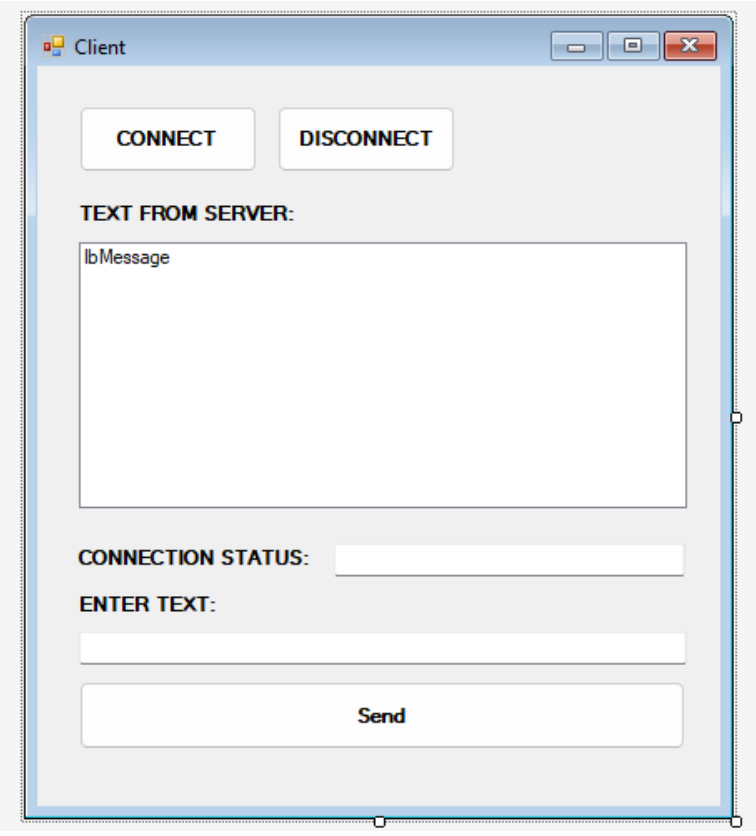




Thiết kế giao diện Form và thực hiện các chức năng sau:

1. Các chức năng phía **Server**:
   - Nút **START SERVER**: bắt đầu lắng nghe, chờ Client kết nối
   - Nút **STOP SERVER**: dừng Socket phía Server
   - **TEXT FROM CLIENTS** (RichTextBox): nhận và hiển thị tin nhắn từ Client
   - **CONNECTION STATUS**: hiển thị trạng thái Server
2. Các chức năng phía **Client**:
   - Nút **CONNECT**:
   - Nút **DISCONNECT**:
   - **TEXT FROM SERVER** (RichTextBox): nhận và hiển thị tin nhắn từ Server
   - Nút **SEND**: gửi tin nhắn trong TextBox tới Server
   - **CONNECTION STATUS**: hiển thị trạng thái Client
3. Khi Client gửi tin nhắn tới Server, Server sẽ gửi lại tin nhắn đó về Client
4. Chương trình cho phép nhiều Client kết nối đồng thời
5. Xử lý phát sinh lỗi khi tắt Server hoặc Client ngắt kết nối

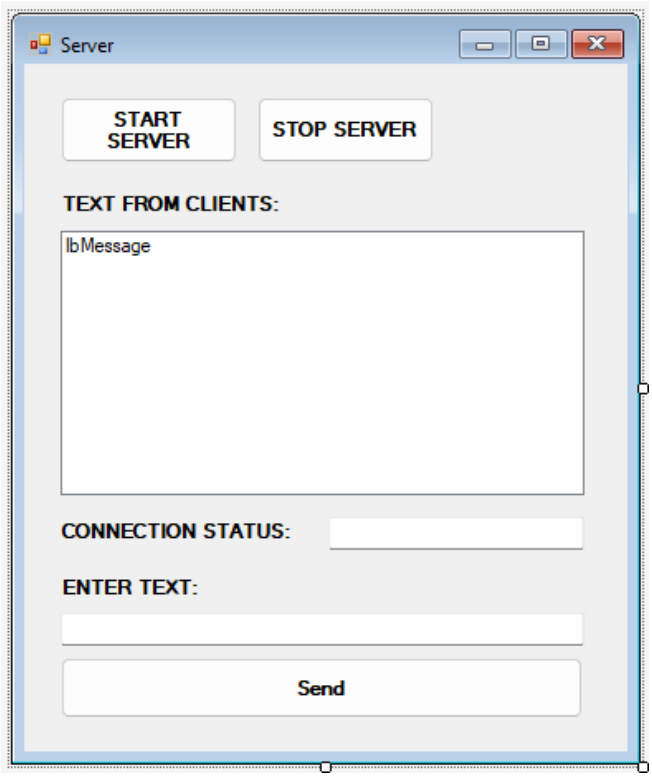
Client



```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Net.Sockets;
using System.Net;
using System.Text;
using System.Threading;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO;

namespace client
{
    public partial class client : Form
    {
        public client()
        {
            InitializeComponent();
            Control.CheckForIllegalCrossThreadCalls = false;

        }
        IPEndPoint ipe;
        TcpClient tcpClient;
        Stream stream;

        private void btnSend_Click(object sender, EventArgs e)
        {
            Send();
        }

        private void Connect()
        {
            ipe = new IPEndPoint(IPAddress.Parse("127.0.0.1"), 6000);
            tcpClient = new TcpClient();
            tcpClient.Connect(ipe);
            stream = tcpClient.GetStream();
            Thread recv = new Thread(Receive);
            recv.IsBackground = true;
            recv.Start();
        }

        private void Send()
        {
            byte[] data = Encoding.UTF8.GetBytes(txtMessage.Text);
            stream.Write(data, 0, data.Length);
            AddMessageToListBox(txtMessage.Text, "Client");
            txtMessage.Text = "";

        }

        private void Receive()
        {
            while (true)
            {
                try
                {
                    byte[] recv = new byte[1024];
                    stream.Read(recv, 0, recv.Length);
                    string str = Encoding.UTF8.GetString(recv);
                    AddMessageToListBox(str, "Server");
                }
                catch (Exception)
```

```
                {
                    AddMessageToListBox("Server đã ngắt kết nối", "Client");
                    btnConnect.Enabled = true;
                    break;
                }
            }
        }

        private void AddMessageToListBox(string msg, string type)
        {
            lbMessage.Items.Add(type + ": " + msg + Environment.NewLine);
        }

        private void btnConnect_Click(object sender, EventArgs e)
        {
            Connect();
            txtStatus.Text = "Connected";
            btnConnect.Enabled = false;
            btnDisconnect.Enabled = true;
            AddMessageToListBox("Đã kết nối tới server!", "Client");

        }

        private void btnDisconnect_Click(object sender, EventArgs e)
        {
            tcpClient.Close();
            txtStatus.Text = "Disconnected";
            btnConnect.Enabled = true;
            btnDisconnect.Enabled = false;
            AddMessageToListBox("Đã ngắt kết nối!", "Client");
        }
    }
}
```

Server




```
using System;
```

```
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Net;
using System.Net.Sockets;
using System.Text;
using System.Threading;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Demo
{
    public partial class Server : Form
    {
        IPEndPoint ipe;
        Socket client;
        TcpListener tcpListener;
        List<Socket> listClient;

        public Server()
        {
            InitializeComponent();
            Control.CheckForIllegalCrossThreadCalls = false;
            listClient = new List<Socket>();
        }

        private void btnSend_Click(object sender, EventArgs e)
        {

            foreach (var item in listClient)
            {
                Send(item);
            }
            AddMessageToListBox(txtMessage.Text, "Server");
            txtMessage.Text = "";
        }

        private void Connect()
        {
            ipe = new IPEndPoint(IPAddress.Any, 6000);

            tcpListener = new TcpListener(ipe);

            Thread thread = new Thread(() =>
            {
                while (true)
                {
                    tcpListener.Start();
                    try
                    {
                        client = tcpListener.AcceptSocket();
                        Thread recv = new Thread((obj) =>
                        {
                            Receive((Socket)obj);
                        });
                        recv.IsBackground = true;
                        recv.Start(client);
                        listClient.Add(client);
                        string msg = string.Format("Client ({0}) đã kết nôi!",
client.RemoteEndPoint);
                        AddMessageToListBox(msg, "Server");
                    }
                    catch (Exception)
```

```
                    {
                        foreach (var item in listClient)
                        {
                            item.Close();
                        }
                        listClient = new List<Socket>();
                        break;
                    }

                }
            });
            thread.IsBackground = true;
            thread.Start();
        }

        private void Send(Socket client)
        {
            byte[] data = Encoding.UTF8.GetBytes(txtMessage.Text);
            client.Send(data);

        }

        private void SendBackMessageToClient(Socket client, string str)
        {
            byte[] data = Encoding.UTF8.GetBytes(str);
            client.Send(data);
        }

        private void Receive(Object obj)
        {
            while (true)
            {
                try
                {
                    Socket client = obj as Socket;
                    byte[] recv = new byte[1024];
                    int byteReceive = client.Receive(recv, 0, recv.Length,
SocketFlags.None);
                    string str = Encoding.UTF8.GetString(recv, 0, byteReceive);
                    string clientInfo = string.Format("Client ({0})",
client.RemoteEndPoint);
                    if (byteReceive == 0)
                    {
                        listClient.Remove(client);
                        client.Close();
                        AddMessageToListBox(clientInfo + " đã ngắt kết nối!",
"Server");
                        break;
                    }
                    //client.Receive(recv);
                    //string str = Encoding.UTF8.GetString(recv);
                    AddMessageToListBox(str, clientInfo);
                    SendBackMessageToClient(client, str);
                }
                catch (Exception)
                {
                    break;
                }
            }
        }

        private void AddMessageToListBox(string msg, string type)
        {
            lbMessage.Items.Add(type + ": " + msg + Environment.NewLine);
        }
```

```
        private void btnStart_Click(object sender, EventArgs e)
        {
            Connect();
            txtStatus.Text = "Server started";
            btnStart.Enabled = false;
            btnStop.Enabled = true;
            AddMessageToListBox("Đang chờ client kết nối...", "Server");
        }

        private void btnStop_Click(object sender, EventArgs e)
        {
            btnStart.Enabled = true;
            btnStop.Enabled = false;
            tcpListener.Stop();
            txtStatus.Text = "Server stopped";
            AddMessageToListBox("Server đã dừng!", "Server");

        }
    }
}
```

Lab3 bai1 Xây dựng chương trình UDPclient- Server đơn giản

Cliient

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Net;
using System.Net.Sockets;

namespace Lab03_Bai01_Client
{
    internal class Program
    {
        static void Main(string[] args)
        {
            //var serverIP = IPAddress.Parse("127.0.0.1");
            IPEndPoint serverEndPoint = new IPEndPoint(IPAddress.Parse("127.0.0.1"),
5000);
            Socket serverSocket = new Socket(AddressFamily.InterNetwork,
SocketType.Dgram, ProtocolType.Udp);
            string str = "Hello Server";
            byte[] data = Encoding.ASCII.GetBytes(str);

            Console.WriteLine("Dang gui cau chao...");
            serverSocket.SendTo(data, data.Length, SocketFlags.None,
serverEndPoint);
            Console.WriteLine("Da gui cau chao");

            Console.ReadLine();
        }
    }
}
```

Server

```
using System;
```

```
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Net;
using System.Net.Sockets;

namespace Lab03_Bai01_Server
{
    internal class Program
    {
        static void Main(string[] args)
        {
            IPEndPoint serverEndPoint = new IPEndPoint(IPAddress.Parse("127.0.0.1"),
5000);
            Socket serverSocket = new Socket(AddressFamily.InterNetwork,
SocketType.Dgram, ProtocolType.Udp);
            serverSocket.Bind(serverEndPoint);
            Console.WriteLine("Dang cho client ket noi...");

            EndPoint clientEndpoint = new IPEndPoint(IPAddress.Any, 0);

            byte[] buffer = new byte[1024];
            int receivedByte;

            receivedByte = serverSocket.ReceiveFrom(buffer, ref clientEndpoint);

            string dataStr = Encoding.ASCII.GetString(buffer, 0, receivedByte);

            Console.WriteLine("Du lieu tu client: " + dataStr);

            Console.ReadLine();

            serverSocket.Close();
        }
    }
}
```

Lab3 bai2 bai tap
Client

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Net;
using System.Net.Sockets;

namespace Lab03_Bai02_Client
{
    internal class Program
    {
        static void Main(string[] args)
        {
            IPEndPoint serverEndpoint = new IPEndPoint(IPAddress.Loopback, 1234);
            Socket serverSocket = new Socket(AddressFamily.InterNetwork,
SocketType.Dgram, ProtocolType.Udp);

            Console.WriteLine("Nhap cau chao: ");
            string str = Console.ReadLine();

            if (str.Equals("exit"))
            {
                serverSocket.Close();
```

```
                Console.WriteLine("Da thoat chuong trinh client");
                Console.ReadLine();
                return;
            }

            byte[] data = Encoding.ASCII.GetBytes(str);

            Console.WriteLine("Dang gui cau chao...");
            serverSocket.SendTo(data, data.Length, SocketFlags.None,
serverEndpoint);
            Console.WriteLine("Da gui cau chao");

            Console.ReadLine();

            serverSocket.Close();
        }
    }
}
```

Server

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Net;
using System.Net.Sockets;

namespace Lab03_Bai02_Server
{
    internal class Program
    {
        static void Main(string[] args)
        {
            IPEndPoint serverEndpoint = new IPEndPoint(IPAddress.Any, 1234);
            Socket serverSocket = new Socket(AddressFamily.InterNetwork,
SocketType.Dgram, ProtocolType.Udp);

            serverSocket.Bind(serverEndpoint);

            Console.WriteLine("Dang cho client ket noi...");

            EndPoint clientEndpoint = new IPEndPoint(IPAddress.Any, 0);

            byte[] buffer = new byte[1024];
            int receivedByte;

            receivedByte = serverSocket.ReceiveFrom(buffer, ref clientEndpoint);

            string dataStr = Encoding.ASCII.GetString(buffer, 0, receivedByte);

            Console.WriteLine("Du lieu tu client: " + dataStr);

            Console.ReadLine();

            serverSocket.Close();
        }
    }
}
```

Lab3 bài3 Cải tiến chương trình UDP client-server để có thể gởi và nhận dữ liệu liên tục

Client

```
using System;
using System.Collections.Generic;
using System.Linq;
```

```
using System.Text;
using System.Threading.Tasks;
using System.Net;
using System.Net.Sockets;

namespace Lab03_Bai03_Client
{
    internal class Program
    {
        static void Main(string[] args)
        {
            IPEndPoint serverEndpoint = new IPEndPoint(IPAddress.Loopback, 1234);
            Socket serverSocket = new Socket(AddressFamily.InterNetwork,
SocketType.Dgram, ProtocolType.Udp);

            while (true)
            {
                Console.Write("Nhap du lieu can gui: ");
                string str = Console.ReadLine();

                if (str.Equals("exit")) break;

                byte[] data = Encoding.ASCII.GetBytes(str);

                serverSocket.SendTo(data, data.Length, SocketFlags.None,
serverEndpoint);
                Console.WriteLine("Da gui cau chao");

                Console.WriteLine();
            }

            Console.WriteLine("Da thoat chuong trinh client");
            Console.ReadLine();
            serverSocket.Close();
        }
    }
}
```

Server

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Net;
using System.Net.Sockets;

namespace Lab03_Bai03_Server
{
    internal class Program
    {
        static void Main(string[] args)
        {
            IPEndPoint serverEndpoint = new IPEndPoint(IPAddress.Any, 1234);
            Socket serverSocket = new Socket(AddressFamily.InterNetwork,
SocketType.Dgram, ProtocolType.Udp);

            serverSocket.Bind(serverEndpoint);

            Console.WriteLine("Dang cho client ket noi...");

            EndPoint clientEndpoint = new IPEndPoint(IPAddress.Any, 0);

            byte[] buffer = new byte[1024];
```

```
            int receivedByte;

            while (true)
            {
                buffer = new byte[1024];
                receivedByte = serverSocket.ReceiveFrom(buffer, ref clientEndpoint);

                string dataStr = Encoding.ASCII.GetString(buffer, 0, receivedByte);

                Console.WriteLine(clientEndpoint + ": " + dataStr);
            }
        }
    }
}
```

Lab3 bài 4 . Sử dụng phương thức Connect ở client để thiết lập kết nối trước với server

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Net;
using System.Net.Sockets;

namespace Lab03_Bai04_Client
{
    internal class Program
    {
        static void Main(string[] args)
        {
            IPEndPoint serverEndpoint = new IPEndPoint(IPAddress.Loopback, 1234);
            Socket serverSocket = new Socket(AddressFamily.InterNetwork,
SocketType.Dgram, ProtocolType.Udp);

            serverSocket.Connect(serverEndpoint);

            if (!serverSocket.Connected)
            {
                Console.WriteLine("Co loi trong qua trinh ket noi");
                Console.ReadLine();
                return;
            }

            while (true)
            {
                Console.Write("Nhap du lieu can gui: ");
                string str = Console.ReadLine();

                if (str.Equals("exit")) break;

                byte[] data = Encoding.ASCII.GetBytes(str);

                serverSocket.Send(data);
                Console.WriteLine("Da gui cau chao");

                Console.WriteLine();
            }

            Console.WriteLine("Da thoat chuong trinh client");
            Console.ReadLine();
            serverSocket.Close();
        }
    }
}
```

Server

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Net;
using System.Net.Sockets;

namespace Lab03_Bai04_Server
{
    internal class Program
    {
        static void Main(string[] args)
        {
            IPEndPoint serverEndpoint = new IPEndPoint(IPAddress.Any, 1234);
            Socket serverSocket = new Socket(AddressFamily.InterNetwork,
SocketType.Dgram, ProtocolType.Udp);

            serverSocket.Bind(serverEndpoint);

            Console.WriteLine("Dang cho client ket noi...");

            EndPoint clientEndpoint = new IPEndPoint(IPAddress.Any, 0);

            byte[] buffer = new byte[1024];
            int receivedByte;

            while (true)
            {
                buffer = new byte[1024];
                receivedByte = serverSocket.ReceiveFrom(buffer, ref clientEndpoint);

                string dataStr = Encoding.ASCII.GetString(buffer, 0, receivedByte);

                Console.WriteLine(clientEndpoint + ": " + dataStr);
            }
        }
    }
}
```

Lab4

Lab4 cau 2 Chương trình gửi và nhận thông tin sinh viên

Client

Employer

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lab04_Client
{
    class Employee
    {
        public int EmployeeID;
        private int LastNameSize;
        public string LastName;
        private int FirstNameSize;
        public string FirstName;
```

```
        public int YearsService;
        public double Salary;
        public int size;

        public Employee() { }

        public Employee(byte[] data)
        {
            int place = 0;
            EmployeeID = BitConverter.ToInt32(data, place);
            place += 4;
            LastNameSize = BitConverter.ToInt32(data, place);
            place += 4;
            LastName = Encoding.ASCII.GetString(data, place, LastNameSize);
            place += LastNameSize;
            FirstNameSize = BitConverter.ToInt32(data, place);
            place += 4;
            FirstName = Encoding.ASCII.GetString(data, place, FirstNameSize);
            place += FirstNameSize;
            YearsService = BitConverter.ToInt32(data, place);
            place += 4;
            Salary = BitConverter.ToDouble(data, place);
        }

        public byte[] GetBytes()
        {
            byte[] data = new byte[1024];
            int place = 0;
            Buffer.BlockCopy(BitConverter.GetBytes(EmployeeID), 0, data, place, 4);
            place += 4;
            Buffer.BlockCopy(BitConverter.GetBytes(LastName.Length), 0, data, place,
4);
            place += 4;
            Buffer.BlockCopy(Encoding.ASCII.GetBytes(LastName), 0, data, place,
LastName.Length);
            place += LastName.Length;
            Buffer.BlockCopy(BitConverter.GetBytes(FirstName.Length), 0, data,
place, 4);
            place += 4;
            Buffer.BlockCopy(Encoding.ASCII.GetBytes(FirstName), 0, data, place,
FirstName.Length);
            place += FirstName.Length;
            Buffer.BlockCopy(BitConverter.GetBytes(YearsService), 0, data, place,
4);
            place += 4;
            Buffer.BlockCopy(BitConverter.GetBytes(Salary), 0, data, place, 8);
            place += 8;
            size = place;
            return data;
        }
    }
}


using System;
using System.Collections.Generic;
using System.Linq;
using System.Net.Sockets;
using System.Text;
using System.Threading.Tasks;

namespace Lab04_Client
{
    class Program
    {
        static void Main(string[] args)
```

```
        {
            Employee emp1 = new Employee();

            emp1.EmployeeID = 1;
            emp1.LastName = "Nguyen";
            emp1.FirstName = "Van A";
            emp1.YearsService = 12;
            emp1.Salary = 3500000;

            TcpClient client;
            try
            {
                client = new TcpClient("127.0.0.1", 9050);
            }
            catch (SocketException)
            {
                Console.WriteLine("Khong ket noi duoc voi server");
                return;
            }

            NetworkStream ns = client.GetStream();
            byte[] data = emp1.GetBytes();
            int size = emp1.size;
            byte[] packsize = new byte[2];
            Console.WriteLine("Kich thuoc goi tin = {0}", size);
            packsize = BitConverter.GetBytes(size);
            ns.Write(packsize, 0, 2);
            ns.Write(data, 0, size);
            ns.Flush();

            while (true)
            {
                Console.Write("Nhap du lieu de gui: ");
                string txt = Console.ReadLine();
                if (txt.Equals("Khong")) break;
                byte[] sData = Encoding.ASCII.GetBytes(txt);
                ns.Write(sData, 0, sData.Length);
                ns.Flush();
            }

            ns.Close();
            client.Close();
            Console.ReadLine();
        }
    }
}

Server
Employer
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lab04_Server
{
    class Employee
    {
        public int EmployeeID;
        private int LastNameSize;
        public string LastName;
        private int FirstNameSize;
        public string FirstName;
        public int YearsService;
```

```
        public double Salary;
        public int size;

        public Employee() { }

        public Employee(byte[] data)
        {
            int place = 0;
            EmployeeID = BitConverter.ToInt32(data, place);
            place += 4;
            LastNameSize = BitConverter.ToInt32(data, place);
            place += 4;
            LastName = Encoding.ASCII.GetString(data, place, LastNameSize);
            place += LastNameSize;
            FirstNameSize = BitConverter.ToInt32(data, place);
            place += 4;
            FirstName = Encoding.ASCII.GetString(data, place, FirstNameSize);
            place += FirstNameSize;
            YearsService = BitConverter.ToInt32(data, place);
            place += 4;
            Salary = BitConverter.ToDouble(data, place);
        }

        public byte[] GetBytes()
        {
            byte[] data = new byte[1024];
            int place = 0;
            Buffer.BlockCopy(BitConverter.GetBytes(EmployeeID), 0, data, place, 4);
            place += 4;
            Buffer.BlockCopy(BitConverter.GetBytes(LastName.Length), 0, data, place,
4);
            place += 4;
            Buffer.BlockCopy(Encoding.ASCII.GetBytes(LastName), 0, data, place,
LastName.Length);
            place += LastName.Length;
            Buffer.BlockCopy(BitConverter.GetBytes(FirstName.Length), 0, data,
place, 4);
            place += 4;
            Buffer.BlockCopy(Encoding.ASCII.GetBytes(FirstName), 0, data, place,
FirstName.Length);
            place += FirstName.Length;
            Buffer.BlockCopy(BitConverter.GetBytes(YearsService), 0, data, place,
4);
            place += 4;
            Buffer.BlockCopy(BitConverter.GetBytes(Salary), 0, data, place, 8);
            place += 8;
            size = place;
            return data;
        }
    }
}

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Net;
using System.Net.Sockets;
using System.IO;

namespace Lab04_Server
{
    class Program
    {
```

```
        static void Main(string[] args)
        {
            byte[] data = new byte[1024];
            TcpListener server = new TcpListener(IPAddress.Any, 9050);
            server.Start();
            TcpClient client = server.AcceptTcpClient();
            NetworkStream ns = client.GetStream();


            byte[] size = new byte[2];
            int recv = ns.Read(size, 0, 2);
            int packsize = BitConverter.ToInt16(size, 0);
            Console.WriteLine("Kich thuoc goi tin = {0}", packsize);
            recv = ns.Read(data, 0, packsize);
            Employee emp1 = new Employee(data);
            Console.WriteLine("emp1.EmployeeID = {0}", emp1.EmployeeID);
            Console.WriteLine("emp1.LastName = {0}", emp1.LastName);
            Console.WriteLine("emp1.FirstName = {0}", emp1.FirstName);
            Console.WriteLine("emp1.YearsService = {0}", emp1.YearsService);
            Console.WriteLine("emp1.Salary = {0}\n", emp1.Salary);

            while (true)
            {
                byte[] rData = new byte[1024];
                recv = ns.Read(rData, 0, 1024);
                var mess = Encoding.ASCII.GetString(rData, 0, recv);
                if (string.IsNullOrEmpty(mess)) break;
                Console.WriteLine("Du lieu nhan: {0}",  mess);

                using (StreamWriter sw = new StreamWriter(@"..\..\..\text.txt"))
                {
                    sw.WriteLine(mess);
                }
            }
            ns.Close();
            client.Close();
            server.Stop();
            Console.ReadLine();
        }
    }
}
```

Lab4 udp client

Employer

```
using System;
using System.Collections.Generic;
using System.Text;

namespace Lab04_UDP_Client
{
    class Employee
    {
        public int EmployeeID;
        private int LastNameSize;
        public string LastName;
        private int FirstNameSize;
        public string FirstName;
        public int YearsService;
        public double Salary;
        public int size;
```

```
        public Employee() { }

        public Employee(byte[] data)
        {
            int place = 0;
            EmployeeID = BitConverter.ToInt32(data, place);
            place += 4;
            LastNameSize = BitConverter.ToInt32(data, place);
            place += 4;
            LastName = Encoding.ASCII.GetString(data, place, LastNameSize);
            place += LastNameSize;
            FirstNameSize = BitConverter.ToInt32(data, place);
            place += 4;
            FirstName = Encoding.ASCII.GetString(data, place, FirstNameSize);
            place += FirstNameSize;
            YearsService = BitConverter.ToInt32(data, place);
            place += 4;
            Salary = BitConverter.ToDouble(data, place);
        }

        public byte[] GetBytes()
        {
            byte[] data = new byte[1024];
            int place = 0;
            Buffer.BlockCopy(BitConverter.GetBytes(EmployeeID), 0, data, place, 4);
            place += 4;
            Buffer.BlockCopy(BitConverter.GetBytes(LastName.Length), 0, data, place,
4);
            place += 4;
            Buffer.BlockCopy(Encoding.ASCII.GetBytes(LastName), 0, data, place,
LastName.Length);
            place += LastName.Length;
            Buffer.BlockCopy(BitConverter.GetBytes(FirstName.Length), 0, data,
place, 4);
            place += 4;
            Buffer.BlockCopy(Encoding.ASCII.GetBytes(FirstName), 0, data, place,
FirstName.Length);
            place += FirstName.Length;
            Buffer.BlockCopy(BitConverter.GetBytes(YearsService), 0, data, place,
4);
            place += 4;
            Buffer.BlockCopy(BitConverter.GetBytes(Salary), 0, data, place, 8);
            place += 8;
            size = place;
            return data;
        }
    }
}


using System;
using System.Net.Sockets;

namespace Lab04_UDP_Client
{
    class Program
    {
        static void Main(string[] args)
        {
            Employee emp1 = new Employee();

            emp1.EmployeeID = 1;
            emp1.LastName = "Nguyen";
            emp1.FirstName = "Van A";
            emp1.YearsService = 12;
```

```
            emp1.Salary = 3500000;

            UdpClient client;
            try
            {
                client = new UdpClient("127.0.0.1", 9050);
            }
            catch (SocketException)
            {
                Console.WriteLine("Khong ket noi duoc voi server");
                return;
            }

            //NetworkStream ns = client.GetStream();
            byte[] data = emp1.GetBytes();
            int size = emp1.size;
            byte[] packsize = new byte[2];
            Console.WriteLine("Kich thuoc goi tin = {0}", size);
            //packsize = BitConverter.GetBytes(size);
            //ns.Write(packsize, 0, 2);
            //ns.Write(data, 0, size);
            //ns.Flush();


            //while (true)
            //{
            //    Console.Write("Nhap du lieu de gui: ");
            //    string txt = Console.ReadLine();
            //    if (txt.Equals("Khong")) break;
            //    byte[] sData = Encoding.ASCII.GetBytes(txt);
            //    ns.Write(sData, 0, sData.Length);
            //    ns.Flush();
            //}

            //ns.Close();
            client.Close();
            Console.ReadLine();
        }
    }
}
```

Udp server
Employer

```
using System;
using System.Collections.Generic;
using System.Text;

namespace Lab04_UDP_Server
{
    class Employee
    {
        public int EmployeeID;
        private int LastNameSize;
        public string LastName;
        private int FirstNameSize;
        public string FirstName;
        public int YearsService;
        public double Salary;
        public int size;
```

```
        public Employee() { }

        public Employee(byte[] data)
        {
            int place = 0;
            EmployeeID = BitConverter.ToInt32(data, place);
            place += 4;
            LastNameSize = BitConverter.ToInt32(data, place);
            place += 4;
            LastName = Encoding.ASCII.GetString(data, place, LastNameSize);
            place += LastNameSize;
            FirstNameSize = BitConverter.ToInt32(data, place);
            place += 4;
            FirstName = Encoding.ASCII.GetString(data, place, FirstNameSize);
            place += FirstNameSize;
            YearsService = BitConverter.ToInt32(data, place);
            place += 4;
            Salary = BitConverter.ToDouble(data, place);
        }

        public byte[] GetBytes()
        {
            byte[] data = new byte[1024];
            int place = 0;
            Buffer.BlockCopy(BitConverter.GetBytes(EmployeeID), 0, data, place, 4);
            place += 4;
            Buffer.BlockCopy(BitConverter.GetBytes(LastName.Length), 0, data, place,
4);
            place += 4;
            Buffer.BlockCopy(Encoding.ASCII.GetBytes(LastName), 0, data, place,
LastName.Length);
            place += LastName.Length;
            Buffer.BlockCopy(BitConverter.GetBytes(FirstName.Length), 0, data,
place, 4);
            place += 4;
            Buffer.BlockCopy(Encoding.ASCII.GetBytes(FirstName), 0, data, place,
FirstName.Length);
            place += FirstName.Length;
            Buffer.BlockCopy(BitConverter.GetBytes(YearsService), 0, data, place,
4);
            place += 4;
            Buffer.BlockCopy(BitConverter.GetBytes(Salary), 0, data, place, 8);
            place += 8;
            size = place;
            return data;
        }
    }
}


using System;

namespace Lab04_UDP_Server
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Hello World!");
        }
    }
}
```

Lab5

ConsoleLogger

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading;
using System.Threading.Tasks;

namespace TcpThread
{
    class ConsoleLogger : ILogger
    {
        private static Mutex mutex = new Mutex();
        public void writeEntry(ArrayList entry)
        {
            mutex.WaitOne();
            IEnumerator line = entry.GetEnumerator();
            while (line.MoveNext())
                Console.WriteLine(line.Current);
            Console.WriteLine();
            mutex.ReleaseMutex();
        }

        public void writeEntry(String entry)
        {
            mutex.WaitOne();
            Console.WriteLine(entry);
            Console.WriteLine();
            mutex.ReleaseMutex();
        }
    }
}
```

EchoProtocol

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using System.Net.Sockets;
using System.Text;
using System.Threading;
using System.Threading.Tasks;

namespace TcpThread
{
    class EchoProtocol : IProtocol
    {
        public const int BUFFSIZE = 32;
        private Socket clntSock;
        private ILogger logger;

        public EchoProtocol(Socket clntSock, ILogger logger)
        {
            this.clntSock = clntSock;
```

```
            this.logger = logger;
        }

        public void handleClient()
        {
            ArrayList entry = new ArrayList();
            entry.Add("Client address and port = " + clntSock.RemoteEndPoint);
            entry.Add("Thread = " + Thread.CurrentThread.GetHashCode());

            try
            {
                int recvMsgSzie;
                int totalBytesEchoed = 0;
                byte[] recvBuffer = new byte[BUFFSIZE];
                try
                {
                    while ((recvMsgSzie = clntSock.Receive(recvBuffer, 0,
recvBuffer.Length, SocketFlags.None)) > 0)
                    {
                        clntSock.Send(recvBuffer, 0, recvMsgSzie, SocketFlags.None);
                        totalBytesEchoed += recvMsgSzie;
                    }
                }
                catch (SocketException se)
                {
                    entry.Add(se.ErrorCode + ": " + se.Message);
                }
                entry.Add("Client finished; echoed " + totalBytesEchoed + " byte.");
            }
            catch (SocketException se)
            {
                entry.Add(se.ErrorCode + ": " + se.Message);

            }
            clntSock.Close();
            logger.writeEntry(entry);
        }
    }
}
```

FileLogger

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading;
using System.Threading.Tasks;

namespace TcpThread
{
    class FileLogger :ILogger
    {
        private static Mutex mutex = new Mutex();
        private StreamWriter output;

        public FileLogger(String filename)
        {
            output = new StreamWriter(filename, true);
        }
```

```
        public void writeEntry(ArrayList entry)
        {
            mutex.WaitOne();
            IEnumerator line = entry.GetEnumerator();
            while (line.MoveNext())
                output.WriteLine(line.Current);
            output.Flush();
            mutex.ReleaseMutex();
        }

        public void writeEntry(String entry)
        {
            mutex.WaitOne();
            output.WriteLine(entry);
            output.WriteLine();
            output.Flush();
            mutex.ReleaseMutex();
        }
    }
}
```

Ilogger

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace TcpThread
{
    public interface ILogger
    {
        void writeEntry(ArrayList entry);
        void writeEntry(String entry);
    }
}
```

Iprotocol

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace TcpThread
{
    public interface IProtocol
    {
        void handleClient();
    }
}
```

Program

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Net.Sockets;
using System.Text;
using System.Threading;
using System.Threading.Tasks;

namespace TcpThread
{
    class Program
    {
        static void Main(string[] args)
        {
            if (args.Length != 1)
                throw new ArgumentException("Parameter(s): <Port>");
            int serverPort = Int32.Parse(args[0]);

            TcpListener listener = new TcpListener(IPAddress.Any, serverPort);
            ILogger logger = new ConsoleLogger();
            listener.Start();
            for (; ; )
            {
                try
                {
                    Socket client = listener.AcceptSocket();
                    EchoProtocol protocol = new EchoProtocol(client, logger);
                    Thread thread = new Thread(new
ThreadStart(protocol.handleClient));
                    thread.Start();
                    logger.writeEntry("Create and start Thread = " +
thread.GetHashCode());
                }
                catch (System.IO.IOException e)
                {
                    logger.writeEntry("Error: " + e.Message);
                }
            }
        }
    }
}
```