Министерство науки и высшего образования Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ "МЭИ"

Институт информационных и вычислительных технологий

Кафедра математического и компьютерного моделирования

Сетевые технологии
Лабораторная работа №3
«Клиент-серверная система на основе классов TcpClient и TcpListener языка C#»

Студент: Симаков А.М.
Группа: А-16-20
Преподаватель: Князев А.В.

Москва 2024

1 Задание

Разработать клиент-серверную систему, реализующую заданный протокол обмена данными.

Программа должна быть разработана в среде Visual Studio на языке C# с использованием классов TcpClient и TcpListener.

Клиент должен иметь удобный интерфейс (меню, окна и т.д.).

Настройки клиента должны включать задание удалённого адреса и порта.

Клиент должен реализовать дополнительные настройки, обеспечивающие гибкость выполнения соответствующего задания.

Сервер должен иметь удобный интерфейс (меню, окна и т.д.).

Сервер должен вести учёт соединений.

Настройки сервера должны включать задание прослушиваемого порта, имя журнала учёта, рабочую директорию.

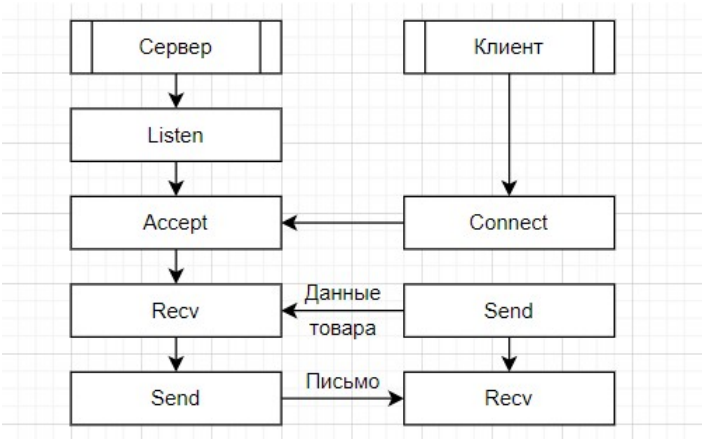
Сервер должен реализовать дополнительные настройки, обеспечивающие гибкость выполнения соответствующего задания.

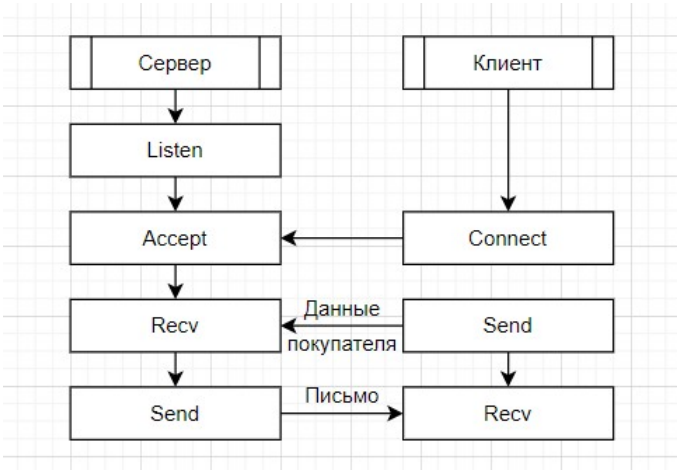
Отчёт по лабораторной работе должен содержать:

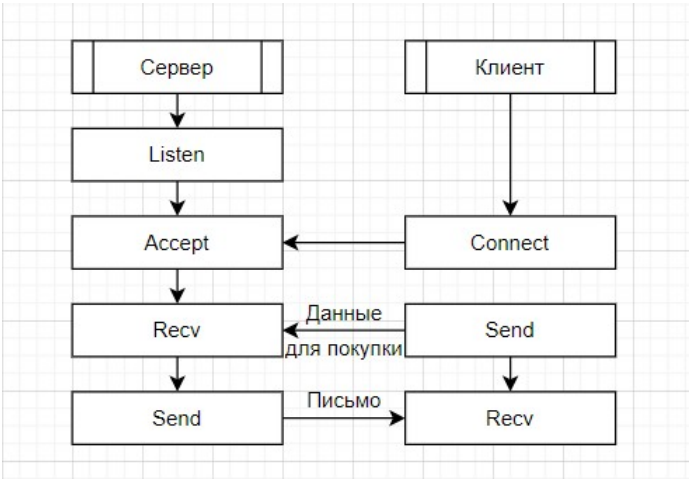
- Титульный лист
- Задание на работу
- Схему, отображающую протокол работы программы
- Тесты
- Распечатки экранов при работе программы
- Листинг программы

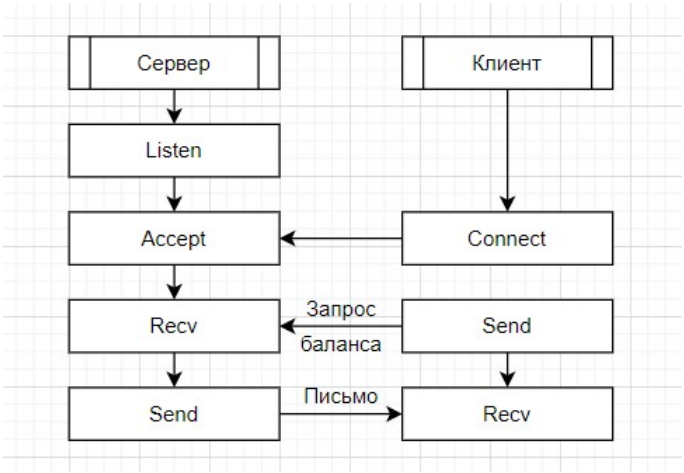
Индивидуальное задание
Модель платёжной системы.
На сервере сначала создаётся база данных товаров: каждая запись содержит код товара, наименование товара, его стоимость и номер счёта продавца. Затем регистрируются покупатели: каждый указывает своё имя, номер своего счёта и сумму денег на этом счёте. Далее покупатели могут посылать на сервер запросы на покупку, указывая своё имя и код товара. Сервер при этом списывает со счёта соответствующую сумму. Покупатель в любой момент может запросить свой баланс. Администратор сервера может получить баланс счетов продавцов.

2 Схема, отображающая протокол работы программы



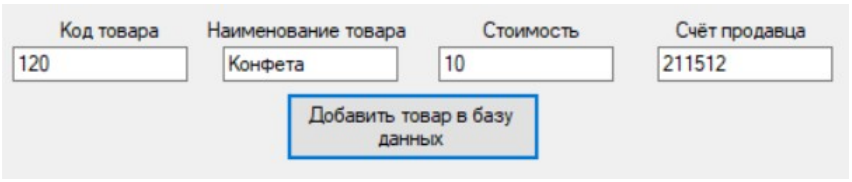






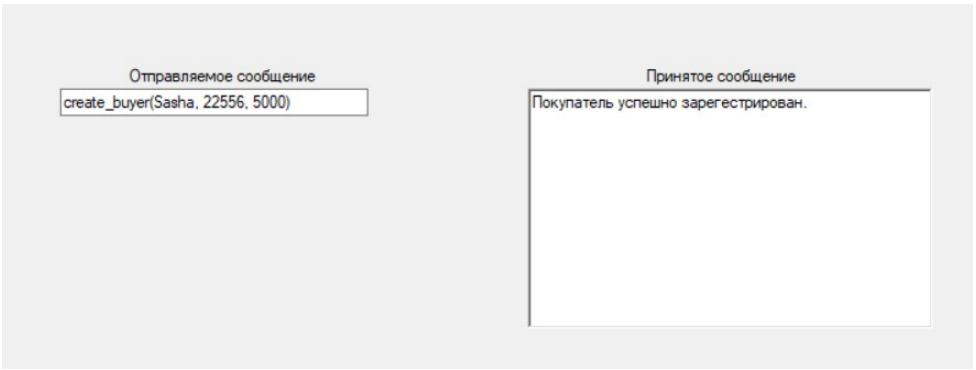
3 Тесты

Создадим товар «Конфета»



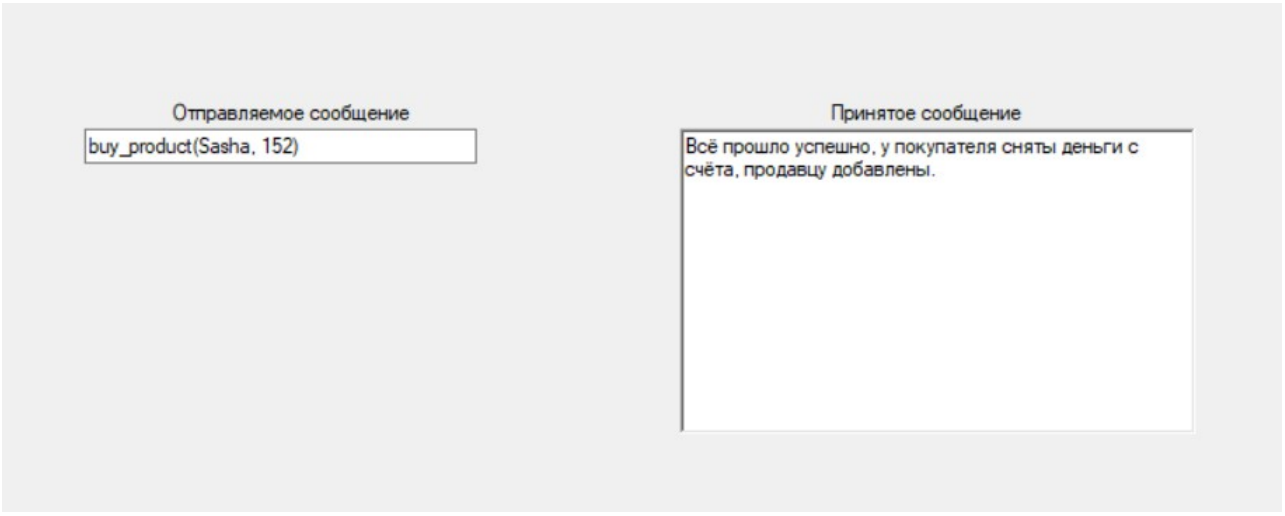

```
415 Молоко 100 421562
152 Яблоко 50 211512
631 Кетчуп 96 421562
125 Шашлык 400 421562
119 Хлеб 25 211512
120 Конфета 10 211512
```

Добавлю покупателя Sasha




```
Alex 21526 4900
Billi 51252 4000
Sasha 22556 5000
```

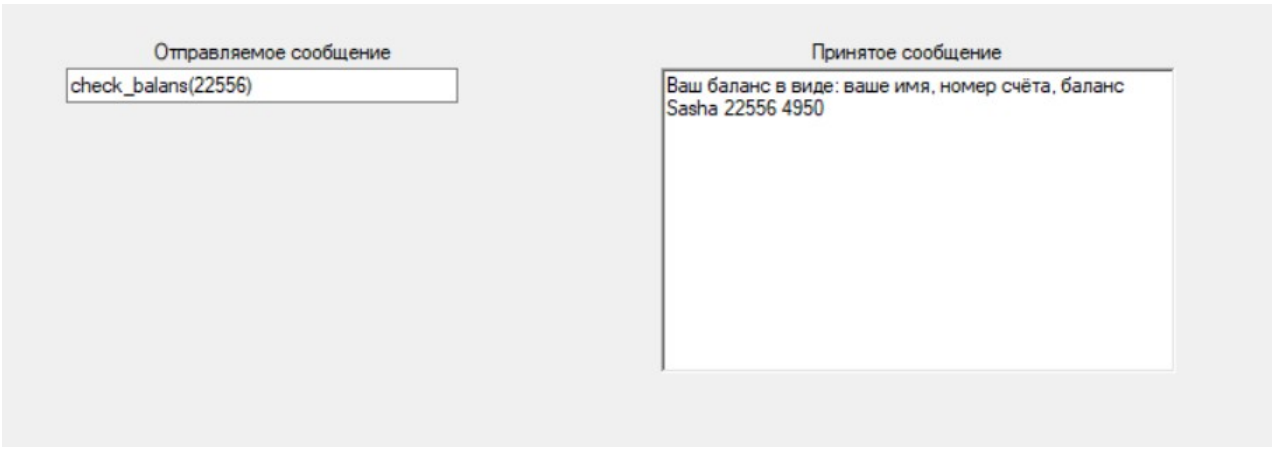
Пусть Sasha купит яблоко (152)



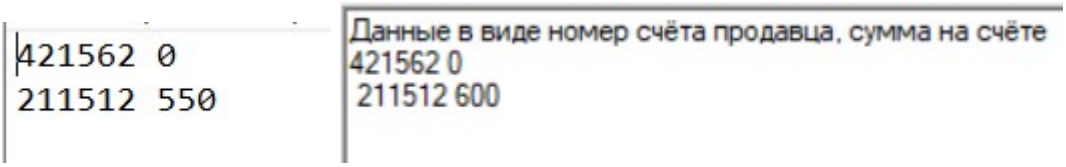
Новые данные покупателей

```
Alex 21526 4900
Billi 51252 4000
Sasha 22556 4950
```

Баланс уменьшился на цену яблока
Проверим баланс командой



Проверим балансы продавцов (слева до покупки яблока, справа - после)



4 Программный код

SERVER

```
using System.ComponentModel;
using System.Net;
using System.Net.Sockets;
using System.Text;
using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.IO;
using System.Text;
```

```
using System.Linq;
using System.Windows.Forms;
using System.Threading;
using System.Drawing.Text;
namespace serv
{
    public partial class Form1 : Form
    {
        TcpListener tcpListener;
        //Объект для вторичного потока у сервера.
        Thread ListenThread;
        StreamWriter sw = new StreamWriter("File1.txt", true);
        //!!!!!Пути до файлов.!!!!!
        string path_DB = "C://Users//Пользователь//Desktop//LR3Sanshos//serv//product_database.txt";
        string path_b = "C://Users//Пользователь//Desktop//LR3Sanshos//serv//buyers.txt";
        string path_s = "C://Users//Пользователь//Desktop//LR3Sanshos//serv//sellers_account.txt";
        //Для работы с файлами и директориями.
        DirectoryInfo DirInfo;
        FileInfo FlInfo;
        public Form1()
        {
            InitializeComponent();
            // Открытие текущей директории.
            Server();
        }
        private void Server()
        {
            // 1. Созд. объект socket.
            tcpListener = new TcpListener(IPAddress.Any, 12000);
            ListenThread = new Thread(new ThreadStart(ListenForClients));
            ListenThread.Start();
        }
        private void ListenForClients()
        {
            //Запуск сервера.
            tcpListener.Start();
            while (true)
            {
                // При запросе на соед. метод созд. новый сокет и соед. его с клиентом.
                // Создаем вторичный поток для клиента.
                TcpClient client = tcpListener.AcceptTcpClient(); // Сервер переходит в режим ожидания з
                // Создаём рабочий сокет.
                Thread clientThread = new Thread(new ParameterizedThreadStart(HandleClientComm));
                clientThread.Start(client);
            }
        }
        // Функция вторичного потока для клиента.
        private void HandleClientComm(object client)
        {
            TcpClient tcpClient = (TcpClient)client;
```

```
NetworkStream clientStream = tcpClient.GetStream();
Socket s = tcpClient.Client;
IPEndPoint ep = (IPEndPoint)s.RemoteEndPoint;
IPAddress addr = ep.Address;
string str = addr.ToString();
DateTime dt = DateTime.Now;
string strdt = dt.ToString("G");
str = str + " " + strdt;
sw.WriteLine(str);
sw.Flush();
byte[] message = new byte[4096];
int bytesRead;
while (true)
{
    bytesRead = 0;
    try
    {
        // Блокируется, пока клиент не пошлет сообщение.
        bytesRead = clientStream.Read(message, 0, 4096);
    }
    catch
    {
        // Появляется ошибка на сокете.
        break;
    }
    // Сообщение успешно получено.
    UTF8Encoding encoder = new UTF8Encoding();
    // Преобразует полученные байты в строку.
    string msg = encoder.GetString(message, 0, bytesRead);
    // Посылает ответное сообщение.
    // Echo(msg, encoder, clientStream);
    // Разбор сообщения.
    string[] query = msg.Split('|');
    string msgTypeInQuery = query[0]; ; // Для хранения типа сообщения.
    string msgMain = query[1];
    switch (msgTypeInQuery)
    {
        case "000":
            { // ChangeDir || GetFile.
                string[] comands_vrem = msgMain.Split(' ');
                //string[] comands = msgMain.Split('(');
                string[] comands = msgMain.Split(new char[] { '(' }, 2,
                StringSplitOptions.RemoveEmptyEntries);
                if (comands.Length > 1)
                {
                    comands[1] = comands[1].Remove(comands[1].Length - 1);
                }
                //Добавление нового покупателя
                if (comands[0] == "create_buyer")
                {
```

```
    // Разделяем входную строку по запятой и удаляем лишние пробелы
    string[] parts = comands[1].Split(',').Select(part =>
    part.Trim()).ToArray();
    // Проверяем, что получилось ровно 3 части
    if (parts.Length != 3)
    {
        return;
    }
    // Формируем строку для записи в файл
    string dataToWrite = string.Join(" ", parts);
    // Записываем данные в файл
    using (System.IO.StreamWriter file = new System.IO.StreamWriter(path_b,
    true))
    {
        file.WriteLine(dataToWrite);
    }
    send_mes("Покупатель успешно зарегестрирован.", encoder, clientStream);
}
else
//Прочитать все товары
if (comands[0] == "check_products")
{
    // Читаем все строки из файла
    string fileContent = string.Join("\n", File.ReadAllLines(path_DB));
    send_mes("Все доступные товары в виде: код продукта, наименование,
    цена, номер счёта продавца\n" + fileContent, encoder, clientStream);
}
//Прочитать все товары
if (comands[0] == "check_balans")
{
    // Читаем все строки из файла
    string[] lines = File.ReadAllLines(path_b);
    string mes = "";
    // Проходим по каждой строке
    foreach (string line in lines)
    {
        if (line.Contains(comands[1]))
        {
            mes = line;
        }
    }
    send_mes("Ваш баланс в виде: ваше имя, номер счёта, баланс\n"+ mes,
    encoder, clientStream);
}
else
//Покупка товара
if (comands[0] == "buy_product")
{
    // Разделяем строку на части по запятой
    string[] p = comands[1].Split(',');
```

```
// Присваиваем значения переменным
string name = p[0];
string prod_id = p[1];
int price = 0;
string[] parts;
string seller = "";
string vrem_s;
string vrem_b;
int it = 0;
//Получение цену продукта, и продавца
// Читаем все строки из файла
string[] products = File.ReadAllLines(path_DB);
// Проходим по каждой строке
foreach (string prod in products)
{
    if (prod.Contains(prod_id))
    {
        // Разделяем строку на части по пробелам
        parts = prod.Split(' ');
        price = int.Parse(parts[2]);
        seller = parts[3];
    }
}
//Добавление суммы на счёт продавца
string[] sellers = File.ReadAllLines(path_s);
//Очищаю файл
File.WriteAllText(path_s, string.Empty);
// Проходим по каждой строке, нужному продавцу добавляем деньги
foreach (string se in sellers)
{
    if (se.Contains(seller))
    {
        vrem_s = sellers[it];
        parts = vrem_s.Split(' ');
        parts[1] = (int.Parse(parts[1]) + price).ToString();
        vrem_s = parts[0] + " " + parts[1];
    }
    else
    {
        vrem_s = se;
    }
    it = it + 1;
    File.AppendAllText(path_s, vrem_s + '\n');
}
//Убавление баланса покупателя
string[] buyers = File.ReadAllLines(path_b);
//Очищаю файл
File.WriteAllText(path_b, string.Empty);
it = 0;
foreach (string bu in buyers)
```

```
                        {
                            if (bu.Contains(name))
                            {
                                vrem_b = buyers[it];
                                parts = vrem_b.Split(' ');
                                parts[2] = (int.Parse(parts[2]) - price).ToString();
                                vrem_b = parts[0] + " " + parts[1] + " " + parts[2];
                            }
                            else
                            {
                                vrem_b = bu;
                            }
                            it = it + 1;
                            File.AppendAllText(path_b, vrem_b + '\n');
                        }
                        send_mes("Всё прошло успешно, у покупателя сняты деньги с счёта,
                        продавцу добавлены.", encoder, clientStream);
                    }
                    else
                    {
                        send_mes("Неизвестная команда ", encoder, clientStream);
                    }
                }
                break;
            case "001":
                { // Другие запросы.
                }
                break;
        }
    }
    tcpClient.Close();
} // Конец функции вторичного потока для клиента.
private void send_mes(string msg, UTF8Encoding encoder, NetworkStream clientStream)
{
    // Посылает ответное сообщение.
    byte[] buffer = encoder.GetBytes(msg);
    clientStream.Write(buffer, 0, buffer.Length);
    clientStream.Flush();
}
private void выходToolStripMenuItem_Click(object sender, EventArgs e)
{
    sw.Close();
    Process.GetCurrentProcess().Kill();
    Close();
}
private void журналСоединенийToolStripMenuItem_Click(object sender, EventArgs e)
{
    string str;
    richTextBox1.Clear();
    sw.Close();
```

```
    StreamReader sr = new StreamReader("File1.txt");
    while (!sr.EndOfStream)
    {
        str = sr.ReadLine();
        richTextBox1.AppendText(str + "\n");
    }
    sr.Close();
    sw = new StreamWriter("File1.txt", true);
}
private void label2_Click(object sender, EventArgs e)
{}
//Добавление товара в базу данных
private void button1_Click(object sender, EventArgs e)
{
    // Получаем данные из текстовых полей
    string data1 = textBox1.Text;
    string data2 = textBox2.Text;
    string data3 = textBox3.Text;
    string data4 = textBox4.Text;
    // Проверяем, что все поля заполнены
    if (string.IsNullOrWhiteSpace(data1) || string.IsNullOrWhiteSpace(data2) ||
    string.IsNullOrWhiteSpace(data3) || string.IsNullOrWhiteSpace(data4))
    {
        MessageBox.Show("Пожалуйста, заполните все поля перед сохранением.", "Предупреждение",
        MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }
    // Формируем строку для записи в файл
    string dataToWrite = $"{data1} {data2} {data3} {data4}";
    // Записываем данные в файл
    using (System.IO.StreamWriter file = new System.IO.StreamWriter(path_DB, true))
    {
        file.WriteLine(dataToWrite);
    }
    MessageBox.Show("Данные успешно сохранены в файл.", "Успех", MessageBoxButtons.OK,
    MessageBoxIcon.Information);
}
private void получитьБалансСчётаПродавцовToolStripMenuItem_Click(object sender, EventArgs e)
{
    // Проверяем, существует ли файл
    if (!System.IO.File.Exists(path_s))
    {
        MessageBox.Show("Файл не найден.", "Ошибка",
        MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }
    // Считываем все строки из файла
    string[] lines = System.IO.File.ReadAllLines(path_s);
    // Проверяем, что в файле есть данные
    if (lines.Length == 0)
```

```
            {
                MessageBox.Show("Файл пуст.", "Предупреждение",
                MessageBoxButtons.OK, MessageBoxIcon.Warning);
                return;
            }
            // Объединяем данные из всех строк файла в одну строку
            string allData = string.Join(" ", lines);
            string[] parts = allData.Split(' ');
            for (int i = 1; i < parts.Length; i += 2)
            {
                parts[i] += "\n";
            }
            string replacedString = string.Join(" ", parts);
            // Выводим данные в текстовые поля
            richTextBox1.Text = "Данные в виде номер счёта продавца, сумма на счёте\n" + replacedString;
        }
    }
}
```

CLIENT

```
using System.ComponentModel;
using System.Net;
using System.Net.Sockets;
using System.Text;
using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.IO;
using System.Text;
using System.Linq;
using System.Windows.Forms;
using System.Threading;
using System.Drawing.Text;
namespace client
{
    public partial class Form1 : Form
    {
        TcpClient client = new TcpClient();
        IPEndPoint serverEndPoint = new IPEndPoint(IPAddress.Parse("127.0.0.1"), 12000);
        FileInfo FlInfo;
        public Form1()
        {
            InitializeComponent();
            client.Connect(serverEndPoint);
        }
        private void SendMessage1(string msg)
        {
            NetworkStream clientStream = client.GetStream();
```

```
            UTF8Encoding encoder = new UTF8Encoding();
            msg = msgType.DirAndFile + msg;
            byte[] buffer = encoder.GetBytes(msg);
            clientStream.Write(buffer, 0, buffer.Length);
            clientStream.Flush();
            Byte[] data = new byte[256];
            String responseData = String.Empty;
            data = new byte[256];
            Int32 bytes;
            bytes = clientStream.Read(data, 0, data.Length);
            responseData = System.Text.Encoding.UTF8.GetString(data, 0, bytes);
            richTextBox1.Text = responseData;
        }
        private void выходToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Close();
        }
        private void richTextBox1_TextChanged(object sender, EventArgs e)
        {}
        private void отправитьСообщениеToolStripMenuItem_Click(object sender, EventArgs e)
        {
            string msg;
            msg = "";
            msg = textBox1.Text;
            SendMessage1(msg);
        }
        private void Form1_Load(object sender, EventArgs e)
        {}
    }
    public static class msgType
    {
        public static string DirAndFile = "000|";
    }
}
```