Лабораторная работа 03. Основы сетевого программирования в .net. Сокеты.

Решения задач по лаб.работе 03 разместить в одном отдельном решении (solution) на C#.
Использовать локальный git-репозиторий для всего решения.
Для каждой задачи должен быть коммит с подписью “feat: lab03 – task0[N]”, где N – номер задания

Задание 1.

На основе примера (см. ниже) и справочной информации: https://metanit.com/sharp/net/3.2.php
создать простое клиент-серверное приложение на сокетах TCP – эхо сервер.
Проверить его работу для одного клиента.

Пример серверной части:

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

// Сервер
class EchoServer
{
    public static void Main()
    {
        StartServer();
    }

    public static void StartServer()
    {
        // Устанавливаем IP-адрес и порт
        IPAddress ipAddress = IPAddress.Parse("127.0.0.1");
        int port = 8000;

        // Создаем TCP listener
        TcpListener server = new TcpListener(ipAddress, port);

        try
        {
            // Запускаем сервер
            server.Start();
            Console.WriteLine("Сервер запущен на {0}:{1}", ipAddress, port);

            while (true)
            {
                Console.WriteLine("Ожидание подключения...");

                // Принимаем клиента
                TcpClient client = server.AcceptTcpClient();
                Console.WriteLine("Клиент подключен!");

                // Получаем поток для чтения и записи
                NetworkStream stream = client.GetStream();

                try
                {
                    byte[] buffer = new byte[1024];
                    int bytesRead;

                    // Читаем данные от клиента
                    while ((bytesRead = stream.Read(buffer, 0, buffer.Length)) > 0)
                    {
                        string receivedMessage = Encoding.UTF8.GetString(buffer, 0, bytesRead);
                        Console.WriteLine("Получено: {0}", receivedMessage);

                        // Отправляем эхо обратно клиенту
                        byte[] response = Encoding.UTF8.GetBytes(receivedMessage);
                        stream.Write(response, 0, response.Length);
```

```
                    Console.WriteLine("Отправлено обратно: {0}", receivedMessage);
                }
            }
            catch (Exception ex)
            {
                Console.WriteLine("Ошибка при обработке клиента: {0}", ex.Message);
            }
            finally
            {
                // Закрываем соединение с клиентом
                stream.Close();
                client.Close();
                Console.WriteLine("Клиент отключен");
            }
        }
    }
    catch (Exception ex)
    {
        Console.WriteLine("Ошибка сервера: {0}", ex.Message);
    }
    finally
    {
        server.Stop();
    }
  }
}
```

Пример клиентской части:

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

class EchoClient
{
    public static void Main(string[] args)
    {
        try
        {
            // Подключаемся к серверу
            TcpClient client = new TcpClient("127.0.0.1", 8000);
            NetworkStream stream = client.GetStream();

            // Бесконечный цикл для ввода сообщений
            while (true)
            {
                Console.Write("Введите сообщение (или 'exit' для выхода): ");
                string message = Console.ReadLine();

                // Проверяем условие выхода
                if (message.ToLower() == "exit")
                    break;

                // От                // Отправляем сообщение
                byte[] data = Encoding.UTF8.GetBytes(message);
                stream.Write(data, 0, data.Length);
                Console.WriteLine("Отправлено: {0}", message);

                // Читаем ответ от сервера
                byte[] buffer = new byte[1024];
                int bytesRead = stream.Read(buffer, 0, buffer.Length);
                string response = Encoding.UTF8.GetString(buffer, 0, bytesRead);
                Console.WriteLine("Получено от сервера: {0}", response);
            }

            // Закрываем соединение
            stream.Close();
            client.Close();
            Console.WriteLine("Соединение закрыто");
        }
        catch (Exception ex)
```

```
        {
            Console.WriteLine("Ошибка клиента: {0}", ex.Message);
        }
    }
}
```

Задание 2.
Модифицировать приложение таким образом, чтобы сервер обеспечивал решение квадратных уравнений. Клиент должен передавать серверу коэффициенты уравнения, получать его корни (учесть возможность отсутствия действительных корней).
Выводы сообщений о решенных уравнениях должны быть в консолях и клиента и сервера. У сервера вести общий счет количества решенных уравнений.

Задание 3.
Модифицировать решение из задания 2 так, чтобы сервер поддерживал работу с несколькими клиентами одновременно (использовать многопоточность).

Дополнительно сервер в своей консоли при выводе результатов запроса от конкретного клиента должен выводить и уникальный ID клиента (создать его самостоятельно)