Отчет лабораторной работы № 1.

Создание простого клиент-серверного приложения «Эхо».

ВАРИАНТ 2

Задание. Создать клиент-серверное приложение (простейший эхо-сервер + клиент), работающее по протоколу TCP, с помощью **сокетов**.

Описание работы

Описание работы сервера и клиента

Серверная часть:

1. Инициализирует сокеты Windows (WSAStartup).
2. Создает сокет для сервера (SOCK_STREAM).
3. Привязывает сокет к IP-адресу и порту.
4. Переводит сокет в режим прослушивания (listen).
5. Принимает входящие соединения от клиентов (accept).
6. Получает сообщения от клиента (recv) и отправляет ответ (send).
7. Закрывает соединение (shutdown, closesocket).

Клиентская часть:

1. Инициализирует сокеты Windows (WSAStartup).
2. Создает сокет для клиента (SOCK_STREAM).
3. Устанавливает соединение с сервером (connect).
4. Отправляет сообщение серверу (send).
5. Получает ответ от сервера (recv).
6. Закрывает соединение (shutdown, closesocket).

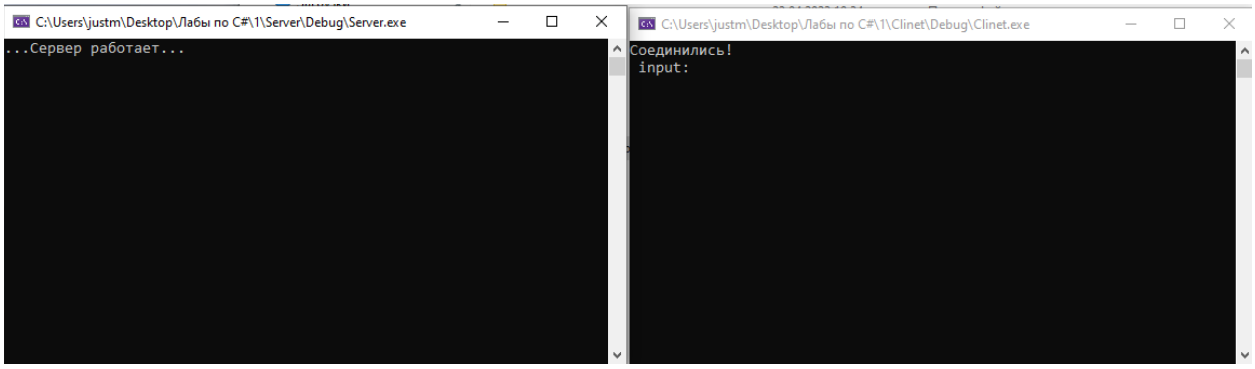
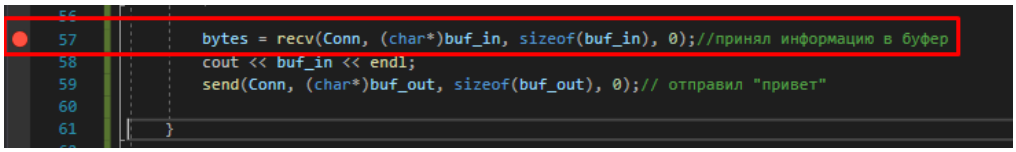


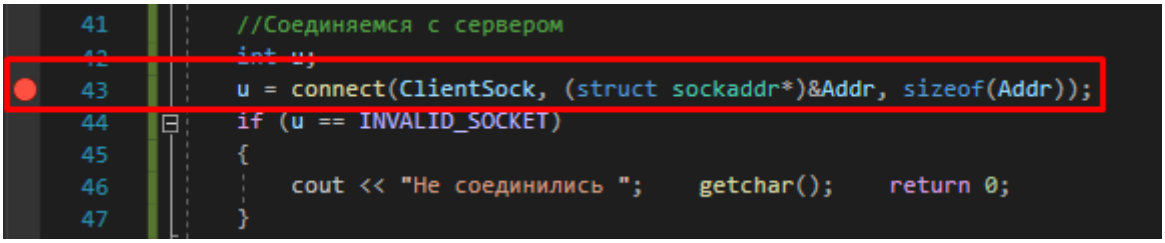
Рисунок 1 – Результаты запуска сервера и клиента

Отладка проверки работы Сервера и Клиента.

Для этого сначала запустим сервер в режиме отладки и установим точку останова на строке:

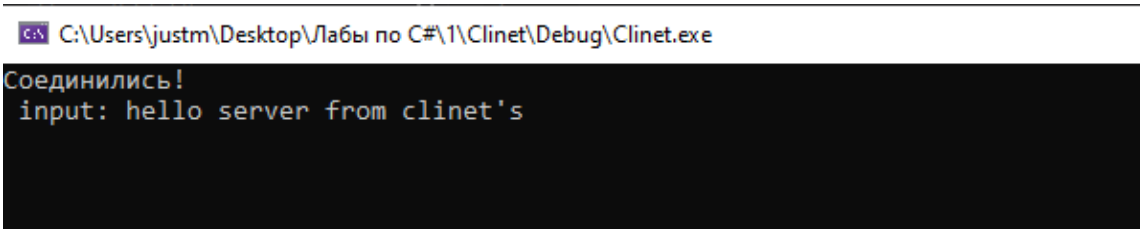


Затем запустим клиента в режиме отладки и установим точку останова на строке:

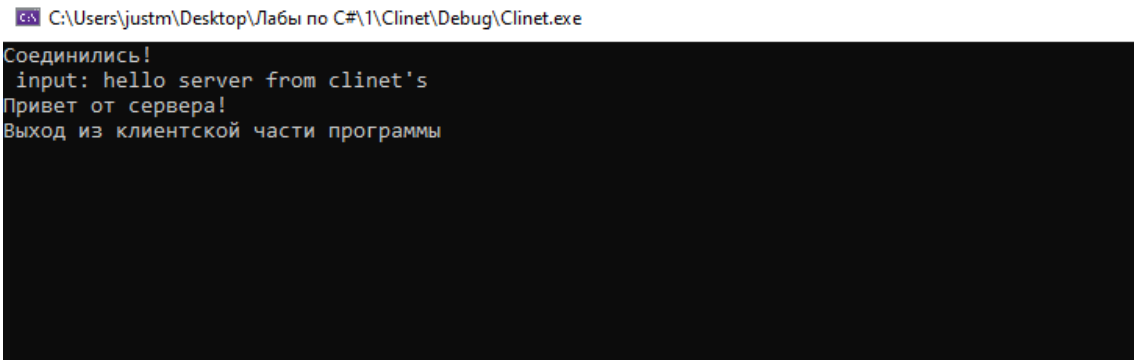


Отладка кода Клиента

После подключения к серверу и прохождения отладки по (F11) gпереходим к строке ***cin >> buf;*** к вводу строки для отправки на сервер, после вводим сообщение для сервера и отправляем ему



После выполнения отладки на сервере и на клиенте получаем на клиенте сообщение от сервера:



Исходный код клиента:

```
using namespace std;

#include <stdio.h>
#include <winsock2.h>
#include <conio.h>
#include <iostream>
#include <Ws2tcpip.h>
#pragma comment(lib, "Ws2_32.lib")
#define  SERVER_PORT 3817

int main(int argc, char* argv[])
{
	setlocale(LC_CTYPE, "Russian"); // включаем русскую кодировку
	SOCKET ClientSock = INVALID_SOCKET;
	WSADATA WSStartData; /* Структура данных библиотеки сокета. */
	CHAR DefaultIPAddr[] = "127.0.0.1"; /* Локальная система. */
	struct sockaddr_in Addr; /* Адрес сокета  */

	WSADATA wsaData;
	int iResult = WSAStartup(MAKEWORD(2, 2), &wsaData);
	if (iResult < 0)
	{
		cout << endl << "ошибка" << endl;
		getchar();
		exit(0);
	}

	// Создали сокет
	ClientSock = socket(AF_INET, SOCK_STREAM, 0);
	// Задаем адрес сервера
	memset(&Addr, 0, sizeof(Addr));
	Addr.sin_family = AF_INET;
	inet_pton(AF_INET, "127.0.0.1", &Addr.sin_addr);
	Addr.sin_port = htons(SERVER_PORT);

	//  HOSTENT* hst ;
	char hostname[] = "                ";
	DWORD size = 20;
	char* localIP;

	//Соединяемся с сервером
	int u;
	u = connect(ClientSock, (struct sockaddr*)&Addr, sizeof(Addr));
	if (u == INVALID_SOCKET)
	{
		cout << "Не соединились ";    getchar();    return 0;
	}
	else cout << "Соединились!\n ";
	int bytes = 0;
	char rez[34] = "";
	char buf[20] = "начало";
	int len;

	cout << "input: ";
```

```
	cin >> buf;
	len = sizeof(buf);
	//Послали сообщение
	send(ClientSock, (char*)buf, len, 0);
	//Получили сообщение
	bytes = recv(ClientSock, (char*)rez, sizeof(rez), 0);
	if (bytes == -1) cout << "результат с ошбкой";
	else cout << rez;


	shutdown(ClientSock, 2); /* Запретить посылку и прием сообщений. */
	closesocket(ClientSock);//закрыть сокет
	WSACleanup();//закрыть библиотеку
	cout << endl << "Выход из клиентской части программы" << endl;
	getchar();
	return 0;
}
```

Исходный код сервера:

```
using namespace std;

#include <stdio.h>
#include <winsock2.h>
#include <conio.h>
#include <iostream>
#include <conio.h>
#include <Ws2tcpip.h>
#pragma comment(lib, "Ws2_32.lib")
#define SERVER_PORT 3817


int main(int argc, char* argv[])
{
	setlocale(LC_CTYPE, "Russian"); // включаем русскую кодировку
	struct sockaddr_in SrvAddr;	// Адресная структура сервера
	struct sockaddr_in ConnAddr;  // Адресная структура клиента
	SOCKET SrvSock, Conn;

	WSADATA wsaData;  // для работы Windows с сокетами
	int iResult = WSAStartup(MAKEWORD(2, 2), &wsaData);
	if (iResult < 0)
	{
		cout << "ошибка" << endl;
		getchar();
		exit(0);
	}


	SrvSock = socket(AF_INET, SOCK_STREAM, 0);//Создаем  сокет

	//Задаем адрес сервера
	SrvAddr.sin_family = AF_INET;
	//SrvAddr.sin_addr.s_addr = inet_addr("127.0.0.1");
	inet_pton(AF_INET, "127.0.0.1", &SrvAddr.sin_addr);
```

```
	SrvAddr.sin_port = htons(3817);
	//Настраиваем сокет
	bind(SrvSock, (sockaddr*)&SrvAddr, sizeof SrvAddr);
	//Слушаем
	listen(SrvSock, 5);

	printf("...Сервер работает...\n");
	int AddrLen = sizeof(ConnAddr);
	char buf_in[20];
	char exit[20] = "exit";
	char buf_out[34] = "Привет от сервера!";
	int nsize;
	int bytes = 0;
	bool stop = false;
	//Ждем очередного клиента
	while (1)
	{
		Conn = accept(SrvSock, (struct sockaddr*)&ConnAddr, &AddrLen);  //
сервер зависает в режиме ожидания
		 //hst = gethostbyaddr((char *)&ConnAddr. sin_addr.s_addr, 4,
AF_INET);
		 //cout<<"Подключился " << inet_ntoa(ConnAddr.sin_addr)<<endl;

		bytes = recv(Conn, (char*)buf_in, sizeof(buf_in), 0);//принял
информацию в буфер
		cout << buf_in << endl;
		send(Conn, (char*)buf_out, sizeof(buf_out), 0);// отправил "привет"

	}

	shutdown(Conn, 2);
	closesocket(Conn);

	cout << endl << "Выход из серверной части программы" << endl;
	_getch();
	return 0;
}
```