МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра информационных технологий**

**Отчёт по лабораторной работе №3**

**«Сетевое клиент-серверное приложение на основе TCP-сокета»**

Работу выполнил

студент 22 группы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Н.Е. Лучин

Направление подготовки 01.03.02 Прикладная математика и информатика

Проверил

канд. техн. наук, доц.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.А. Полупанов

Краснодар 2022

**Цель работы:** реализовать сетевое клиент-серверное приложение на основе TCP-сокета, позволяющее передавать данные произвольных типов (структуру).

**Задача:** приложение состоит из двух программ: клиент и сервер, взаимодействие которых осуществляется с помощью TCP-сокета. Алгоритм взаимодействия следующий: клиент посылает данные через сокет, сервер в бесконечном цикле слушает свой входной порт и при появлении новых сообщений обрабатывает их, а затем посылает клиенту ответное сообщение с результатами обработки.

**Практическая часть:**

Листинг клиента:

#include <iostream>

// для работы с сокетами

#include <winsock2.h>

// вектор для хранения строк

#include <string>

// для работы с Windows API

#include <windows.h>

/\*

линкер скомпонует библиотеку в исполняемый файл

.exe =

объектные файлы (smth.cpp -> smth.o)

+

библиотеки

\*/

#pragma comment(lib, "Ws2\_32.lib")

#define SRV\_HOST "localhost"

#define SRV\_PORT 1234

#define CLNT\_PORT 1235

using namespace std;

/\*

объявление структуры с коэффициентами

для квадратного трехчлена

\*/

struct Coefficients

{

    int a, b, c;

} coeffs;

/\*

объявление структуры с ответом

от сервера

\*/

struct Equation {

    int roots;

    double r1, r2;

} eq;

int main()

{

    // буфер размером 1 кб

    char buff[1024];

    /\*

    инициализация Winsock:

0x0202 - шестнадцатеричный номер версии

интерфейса 2.2

    можно аналогично написать MAKEWORD(2, 2)

    второй аргумент - преобразовываем

    адрес первого элемента буфера в

    указатель на структуру WSADATA, в которую

    будут записаны сведения о конкретной реализации

    интерфейса Winsock

    вместо него можно было бы определить:

    WSADATA wsaData;

    и передать &wsaData

    \*/

    if (WSAStartup(0x0202, (WSADATA\*) &buff[0]))

    {

/\*

если иниц-ия не удалась, вернется ненулевое значение

\*/

cout << "Error WSAStartup\n"

<< WSAGetLastError();

        return -1;

    }

    // объявление сокета

    SOCKET s;

    /\*

    объявление указателя на структуру hostent,

    хранящую информацию об узле

    \*/

    hostent\* hp;

    // объявление структур сокетов для клиента и сервера

    sockaddr\_in clnt\_sin, srv\_sin;

    /\*

    создание сокета для протокола IPv4 (AF\_INET)

    с режимом потокового соединения (SOCK\_STREAM)

    0 - выбор протокола по умолчанию для заданного

    потокового соединения

    (для SOCK\_STREAM будет выбран TCP)

    \*/

    s = socket(AF\_INET, SOCK\_STREAM, 0);

    // используем IPv4

    clnt\_sin.sin\_family = AF\_INET;

    // указание IP-адреса. 0 ~ INADDR\_ANY - любой IP

    clnt\_sin.sin\_addr.s\_addr = 0;

    // htons() преобразует порт в сетевой формат

    clnt\_sin.sin\_port = htons(CLNT\_PORT);

    // привязываем заполненную структуру к сокету

    bind(s, (sockaddr\*) &clnt\_sin, sizeof(clnt\_sin));

    // получаем IP по доменному имени сервера

    hp = gethostbyname(SRV\_HOST);

    // заполняем структуру sockaddr\_in для сервера

    srv\_sin.sin\_port = htons(SRV\_PORT);

    srv\_sin.sin\_family = AF\_INET;

    // заполняем адрес сервера

    ((unsigned long\*) &srv\_sin.sin\_addr)[0] =

        ((unsigned long\*\*) hp -> h\_addr\_list)[0][0];

    connect(s, (sockaddr\*) &srv\_sin, sizeof(srv\_sin));

    while (true)

    {

cout << "Input coeffs: a b c "

<< "(or enter all zeros to exit)"

<< endl;

        cin >> coeffs.a >> coeffs.b >> coeffs.c;

        if (!coeffs.a && !coeffs.b && !coeffs.c)

        {

            send(s, (char\*) "bye", sizeof("bye"), 0);

            closesocket(s);

            cout << "exiting..." << endl;

            return 0;

        }

        // отправка запроса серверу

        send(s, (char\*) &coeffs, sizeof(coeffs), 0);

        /\*

        получение ответа от сервера

        передаем сокет, буфер, размер буфера

        и int flags = 0 - данные будут

        удалены из сокета после приема

        recv() вернет число считанных байтов

        \*/

        int from\_len = recv(s, (char\*) &eq,

sizeof(eq), 0);

        // интерпретация ответа от сервера

        if (eq.roots == 0)

            cout << "There're no roots in real nums"

<< endl;

        else if (eq.roots == 1)

        {

            cout << "1 root: " << eq.r1 << endl;

        }

        else

        {

            cout << "2 roots: " << eq.r1

<< " " << eq.r2 << endl;

        }

        cout << endl;

    }

    return 0;

}

Листинг сервера:

#include <iostream>

#include <winsock2.h>

#include <string>

#include <windows.h>

#include <cmath>

#pragma comment(lib, "Ws2\_32.lib")

#define SRV\_PORT 1234

// размер буфера

#define BUF\_SIZE 64

using namespace std;

struct Coefficients

{

    int a, b, c;

} coeffs;

struct Equation {

    int roots;

    double r1, r2;

} eq;

int main() {

    char buff[1024];

    if (WSAStartup(0x0202, (WSADATA\*) &buff[0]))

    {

        cout << "Error WSAStartup\n"

<< WSAGetLastError();

        return -1;

    }

    SOCKET s, s\_new;

    // инициализация массива символов

    /\*

    примеры:

    x has type int[3] and holds 1,2,3:

    int x[] = {1,2,3};

    y has type int[5] and holds 1,2,3,0,0:

    int y[5] = {1,2,3};

    z has type int[4] and holds 1,0,0,0:

    int z[4] = {1};

    w has type int[3] and holds all zeroes:

    int w[3] = {0};

    \*/

    char buf[BUF\_SIZE] = {0};

    // структуры сокетов для сервера и клиента

    sockaddr\_in sin, from\_sin;

    s = socket(AF\_INET, SOCK\_STREAM, 0);

    sin.sin\_family = AF\_INET;

    sin.sin\_addr.s\_addr = 0;

    sin.sin\_port = htons(SRV\_PORT);

    bind(s, (sockaddr\*) &sin, sizeof(sin));

    // создаем очередь запросов длиной 10

    listen(s, 10);

    while (true)

    {

        int from\_len = sizeof(from\_sin);

        /\*

        принимаем соединение из очереди -

        инициализируем сокет s\_new

        \*/

        s\_new = accept(s, (sockaddr\*) &from\_sin,

&from\_len);

        cout << "new connected client!" << endl;

        while (true)

        {

            /\*

получение запроса без очистки данных

(MSG\_PEEK)

\*/

            from\_len = recv(s\_new, (char\*) &coeffs,

sizeof(coeffs), MSG\_PEEK);

            /\*

чистим данные и проверяем на

наличие строки "bye"

\*/

            recv(s\_new, (char\*) &buf, BUF\_SIZE, 0);

            // запишем 0 в конец полученных данных

            buf[from\_len] = 0;

            string msg = (string) buf;

            if (msg == "bye")

                break;

   cout << coeffs.a << " " << coeffs.b

<< " " << coeffs.c << endl;

// считаем дискриминант

            double D = coeffs.b \* coeffs.b –

4 \* coeffs.a \* coeffs.c;

            // просчет веществ-х корней (если они есть)

            if (D > 0)

            {

                eq.roots = 2;

                eq.r1 = (-coeffs.b + sqrt(D)) /

(2 \* coeffs.a);

                eq.r2 = (-coeffs.b - sqrt(D)) /

(2 \* coeffs.a);

            }

            else if (D == 0)

            {

                eq.roots = 1;

                eq.r1 = -coeffs.b / (2 \* coeffs.a);

            }

            else

                eq.roots = 0;

            // отправка ответа

            send(s\_new, (char\*) &eq, sizeof(eq), 0);

        }

        cout << "client is lost" << endl;

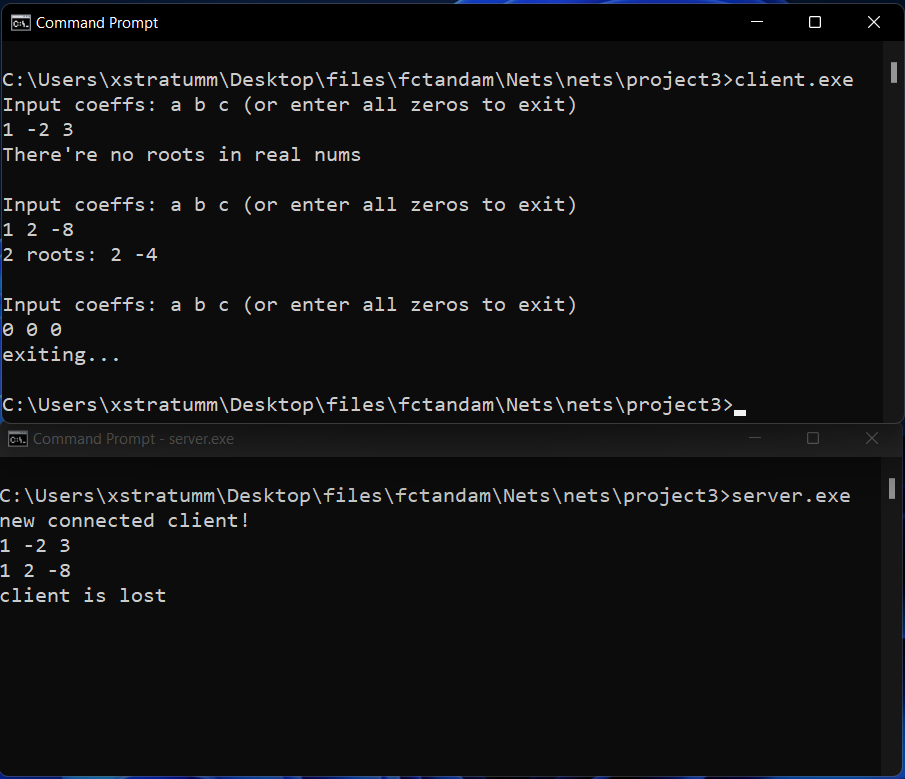
        closesocket(s\_new);

    }

    return 0;

}

Пример работы программ:



*Рисунок 1 – пример работы клиента с сервером*

**Вывод:**былоразработано клиент-серверное приложение на основе TCP-сокетов.