

***Министерство образования Республики Беларусь***  
***Учреждение Образования***  
***«Брестский Государственный Технический Университет»***  
***Кафедра ИИТ***

**Лабораторная работа №4**  
**По дисциплине АПОЭВМ за VI семестр**  
**Тема: «Программирование сетевых приложений на базе WINSOCK2»**

**Выполнил:**

Студент 3-го курса

Группы АС-56

Ровенский Д.В.

**Проверил:**

Булей Е.В.

Брест 2022

Цель работы: изучить основы программирования сетевых приложений Windows на базе библиотеки WINSOCK2.H; приобрести навыки по практическому использованию библиотеки для реализации сетевых приложений в среде C++ на базе протоколов TCP и UDP.

## Ход работы

### Вариант 1

#### Задание

Функция, выполняемая TCP-сервером:

Отсылка клиенту содержимого текстового файла <имя файла> в случае приема сервером в потоке символов команды load fname.txt, <имя файла> – имя некоторого текстового файла, находящегося в каталоге сервера. В случае, если запрашиваемый файл отсутствует в каталоге сервера, сервер должен отослать сообщение об этом и разорвать соединение.

Server.cpp:

```
//Сервер
#define _WINSOCK_DEPRECATED_NO_WARNINGS
#include <iostream>
#include <cstdio>
#include <string>
#include <fstream>
#include <winsock2.h>
#pragma comment(lib, "WS2_32.lib")
using namespace std;

DWORD WINAPI serverReceive(LPVOID lpParam) { //Получение данных от клиента
    char buffer[1024] = { 0 }; //Буфер для данных
    SOCKET client = *(SOCKET*)lpParam; //Сокет для клиента
    while (true) { //Цикл работы сервера
        if (recv(client, buffer, sizeof(buffer), 0) == SOCKET_ERROR) {
            //Если не удалось получить данные буфера, сообщить об ошибке и выйти
            cout << "recv function failed with error " << WSAGetLastError() << endl;
            return -1;
        }
        if (strcmp(buffer, "exit\n") == 0) { //Если клиент отсоединился
            cout << "Client Disconnected." << endl;
            break;
        }
        string command = "";
        for (int i = 0; i < strlen(buffer); ++i) {
            if (buffer[i] == ' ') break;
            command += buffer[i];
        }
        if (command == "load") {
            cout << "Client print load!!!!!!" << endl;
            string filename = "";
```

```

        for (int i = command.length()+1; i < strlen(buffer); ++i) {
            if (buffer[i] == '\n') break;
            filename += buffer[i];
        }
        cout << "Name of file: " << filename << endl;
        ifstream file;
        file.open(filename, ios::in | ios::binary);
        if (!file.is_open()) {
            char err[] = "Can't open file\nConnection close";
            cout << err << endl;
            send(client, err, sizeof(err), 0);
            break;
        }
        else {
            file.seekg(0, file.end);
            long double size = file.tellg();
            file.seekg(0, file.beg);
            char* text = new char[size];
            file.read(text, size);

            send(client, text, size, 0);
        }
    }
    else cout << "Client: " << buffer << endl; //Иначе вывести сообщение от
клиента из буфера
    memset(buffer, 0, sizeof(buffer)); //Очистить буфер
}
return 1;
}

DWORD WINAPI serverSend(LPVOID lpParam) { //Отправка данных клиенту
    char buffer[1024] = { 0 };
    SOCKET client = *(SOCKET*)lpParam;
    while (true) {
        fgets(buffer, 1024, stdin);
        if (send(client, buffer, sizeof(buffer), 0) == SOCKET_ERROR) {
            cout << "send failed with error " << WSAGetLastError() << endl;
            return -1;
        }
        if (strcmp(buffer, "exit\n") == 0) {
            cout << "Thank you for using the application" << endl;
            break;
        }
    }
    return 1;
}

int main() {
    setlocale(LC_ALL, "rus");
    WSADATA WSAData; //Данные
    SOCKET server, client; //Сокеты сервера и клиента
    SOCKADDR_IN serverAddr, clientAddr; //Адреса сокетов
    WSStartup(MAKEWORD(2, 0), &WSAData);
    server = socket(AF_INET, SOCK_STREAM, 0); //Создали сервер
    if (server == INVALID_SOCKET) {
        cout << "Socket creation failed with error:" << WSAGetLastError() << endl;
        return -1;
    }
    serverAddr.sin_addr.s_addr = INADDR_ANY;
    serverAddr.sin_family = AF_INET;
    serverAddr.sin_port = htons(5555);

```

```

    if (bind(server, (SOCKADDR*)&serverAddr, sizeof(serverAddr)) == SOCKET_ERROR) {
        cout << "Bind function failed with error: " << WSAGetLastError() << endl;
        return -1;
    }

    if (listen(server, 0) == SOCKET_ERROR) { //Если не удалось получить запрос
        cout << "Listen function failed with error:" << WSAGetLastError() << endl;
        return -1;
    }
    cout << "Listening for incoming connections...." << endl;

    char buffer[1024]; //Создать буфер для данных
    int clientAddrSize = sizeof(clientAddr); //Инициализировать адрес клиента
    if ((client = accept(server, (SOCKADDR*)&clientAddr, &clientAddrSize)) !=
INVALID_SOCKET) {
        //Если соединение установлено
        cout << "Client connected!" << endl;
        cout << "Enter \"exit\" to disconnect" << endl;

        DWORD tid; //Идентификатор
        HANDLE t1 = CreateThread(NULL, 0, serverReceive, &client, 0, &tid); //Создание
потока для получения данных
        if (t1 == NULL) { //Ошибка создания потока
            cout << "Thread Creation Error: " << WSAGetLastError() << endl;
        }
        HANDLE t2 = CreateThread(NULL, 0, serverSend, &client, 0, &tid); //Создание
потока для отправки данных
        if (t2 == NULL) {
            cout << "Thread Creation Error: " << WSAGetLastError() << endl;
        }

        WaitForSingleObject(t1, INFINITE);
        WaitForSingleObject(t2, INFINITE);

        closesocket(client); //Закреть сокет
        if (closesocket(server) == SOCKET_ERROR) { //Ошибка закрытия сокета
            cout << "Close socket failed with error: " << WSAGetLastError() << endl;
            return -1;
        }
        WSACleanup();
    }
}

```

Client.cpp:

```

//Клиент
#define _WINSOCK_DEPRECATED_NO_WARNINGS
#include <iostream>
#include <cstdio>
#include <cstring>
#include <winsock2.h>
#pragma comment(lib, "WS2_32.lib")
using namespace std;

DWORD WINAPI clientReceive(LPVOID lpParam) { //Получение данных от сервера
    char buffer[10240] = { 0 };
    SOCKET server = *(SOCKET*)lpParam;
    while (true) {
        if (recv(server, buffer, sizeof(buffer), 0) == SOCKET_ERROR) {

```

```

        cout << "recv function failed with error: " << WSAGetLastError() << endl;

        return -1;
    }
    if (strcmp(buffer, "exit\n") == 0) {
        cout << "Server disconnected." << endl;
        return 1;
    }
    cout << "Server: " << buffer << endl;
    memset(buffer, 0, sizeof(buffer));
}
return 1;
}

DWORD WINAPI clientSend(LPVOID lpParam) { //Отправка данных на сервер
    char buffer[1024] = { 0 };
    SOCKET server = *(SOCKET*)lpParam;
    while (true) {
        fgets(buffer, 1024, stdin);
        if (send(server, buffer, sizeof(buffer), 0) == SOCKET_ERROR) {
            cout << "send failed with error: " << WSAGetLastError() << endl;
            return -1;
        }
        if (strcmp(buffer, "exit") == 0) {
            cout << "Thank you for using the application" << endl;
            break;
        }
    }
    return 1;
}

int main() {
    setlocale(LC_ALL, "rus");
    WSADATA WSAData;
    SOCKET server;
    SOCKADDR_IN addr;
    WSASStartup(MAKEWORD(2, 0), &WSAData);
    if ((server = socket(AF_INET, SOCK_STREAM, 0)) == INVALID_SOCKET) {
        cout << "Socket creation failed with error: " << WSAGetLastError() << endl;
        return -1;
    }

    addr.sin_addr.s_addr = inet_addr("127.0.0.1"); //коннект к серверу
    addr.sin_family = AF_INET;
    addr.sin_port = htons(5555); //порт
    if (connect(server, (SOCKADDR*)&addr, sizeof(addr)) == SOCKET_ERROR) {
        cout << "Server connection failed with error: " << WSAGetLastError() << endl;
        return -1;
    }

    cout << "Connected to server!" << endl;
    cout << " Enter \"exit\" to disconnect" << endl;

    DWORD tid;
    HANDLE t1 = CreateThread(NULL, 0, clientReceive, &server, 0, &tid);
    if (t1 == NULL) cout << "Thread creation error: " << GetLastError();
    HANDLE t2 = CreateThread(NULL, 0, clientSend, &server, 0, &tid);
    if (t2 == NULL) cout << "Thread creation error: " << GetLastError();

    WaitForSingleObject(t1, INFINITE);
    WaitForSingleObject(t2, INFINITE);
}

```

```
        closesocket(server);  
        WSACleanup();  
    }
```

Результат выполнения программы:

Клиент:

```
Connected to server!  
Enter "exit" to disconnect  
load test.txt  
Server: qwerty  
qwertyui  
qwertyuiop  
load ojfjojdgshsojdgnojnndfljfgnbljdfnbjnfdljbndfnkjcfn  
Server: Can't open file  
Connection close
```

Сервер:

```
Listening for incoming connections....  
Client connected!  
Enter "exit" to disconnect  
Client print load!!!!!!  
Name of file: test.txt  
Client print load!!!!!!  
Name of file: ojfjojdgshsojdgnojnndfljfgnbljdfnbjnfdljbndfnkjcfn  
Can't open file  
Connection close
```

Вывод: я изучил основы программирования сетевых приложений Windows на базе библиотеки WINSOCK2.H; приобрел навыки по практическому использованию библиотеки для реализации сетевых приложений в среде C++ на базе протоколов TCP и UDP