

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

Лабораторная работа №4-5

По дисциплине АПОЭВМ за VI семестр

Тема: «Программирование сетевых приложений на базе WINSOCK2 - UDP»

Выполнил:

Студент 3-го курса

Группы АС-56

Климович М.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 протокола UDP. Ход работы

Вариант 5 Задание

Вариант	Ввод символов с отсылкой введенной строки по нажатию на клавишу	Ведение файла протокола событий, включающих: 1) время начала и окончания соединения; 2) передаваемую серверу строку и время передачи строки; 3) принимаемую от сервера строку и время приема строки.	Возможность разрыва соединения при помощи команды: disconnect <адрес> <порт>	1) Задание в программе клиента специальной команды и параметров: подключения к серверу: connect <адрес> <порт> 2) Автоматическое подключение к серверу с заданным по умолчанию адресом при запуске клиента
4□	PgDn□	1), 2), 3)□	-□	2)□

UDPserv.cpp:

```
#define _CRT_SECURE_NO_WARNINGS
#define _WINSOCK_DEPRECATED_NO_WARNINGS
#pragma comment(lib, "Ws2_32.lib")

#include <winsock2.h>
#include <stdio.h>
#include <time.h>

#define BUFLen 512 //Max length of buffer
#define PORT 777 //The port on which to listen for incoming data

int main()
{
    SYSTEMTIME lt;
    SOCKET s;
    struct sockaddr_in server, si_other;
    int slen, recv_len;
    char buf[BUFLen];
    WSADATA wsa;
    slen = sizeof(si_other);

    //Initialise winsock
    printf("\nInitialising Winsock...");
    if (WSAStartup(MAKEWORD(2, 2), &wsa) != 0)
    {
        printf("Failed. Error Code : %d", WSAGetLastError());
        exit(EXIT_FAILURE);
    }
    printf("Initialised.\n");
```

```

//Create a socket
if ((s = socket(AF_INET, SOCK_DGRAM, 0)) == INVALID_SOCKET)
{
    printf("Could not create socket : %d", WSAGetLastError());
}
printf("Socket created.\n");

//Prepare the sockaddr_in structure
server.sin_family = AF_INET; server.sin_addr.s_addr =
INADDR_ANY; server.sin_port = htons(PORT);

//Time
GetLocalTime(&lt);

//Bind
if (bind(s, (struct sockaddr*)&server, sizeof(server)) == SOCKET_ERROR)
{
    printf("Bind failed with error code : %d", WSAGetLastError());
    exit(EXIT_FAILURE);
}
printf("Bind done: %02d.%02d.%02d %02d:%02d:%02d\n", lt.wYear, lt.wMonth, lt.wDay,
lt.wHour, lt.wMinute, lt.wSecond);

//keep listening for data
while (1)
{
    printf("Waiting for data...\n\n");
    fflush(stdout);

    //clear the buffer by filling null, it might have previously received data
    memset(buf, '\0', BUFLen);

    //try to receive some data, this is a blocking call if ((recv_len =
recvfrom(s, buf, BUFLen, 0, (struct sockaddr*)&si_other, &slen)) == SOCKET_ERROR)
    {
        printf("recvfrom() failed with error code : %d", WSAGetLastError());
        exit(EXIT_FAILURE);
    }

    GetLocalTime(&lt);
    //print details of the client/peer and the data received printf("Received
packet from %s:%d\n", inet_ntoa(si_other.sin_addr), ntohs(si_other.sin_port));
    printf("Data: %s %02d.%02d.%02d
%02d:%02d:%02d\n", buf, lt.wYear, lt.wMonth, lt.wDay, lt.wHour, lt.wMinute, lt.wSecond);

    //now reply the client some data
    if (sendto(s, buf, recv_len, 0, (struct sockaddr*)&si_other, slen) == SOCKET_ERROR)
    {
        printf("sendto() failed with error code : %d", WSAGetLastError());
        exit(EXIT_FAILURE);
    }
}

closesocket(s);
WSACleanup();

```

```

        return 0;
    }

```

UDPclient.cpp:

```

#define _WINSOCK_DEPRECATED_NO_WARNINGS
#pragma comment(lib, "Ws2_32.lib")

#include<stdio.h>
#include<winsock2.h>
#include<iostream>
#include<string>
#include <time.h>
#include <conio.h>

#define BUFLen 512 //Max length of buffer using namespace
std;

int main(void)
{
    SYSTEMTIME lt;      struct sockaddr_in
    si_other;    int s,
    slen = sizeof(si_other);
    char buf[BUFLen];
    char message[BUFLen];
    short PORT;
    string IP;

    WSADATA wsa;

    IP = "127.0.0.1";
    const char* SERVER = IP.c_str();
    PORT = 777;

    //Initialise winsock
    printf("\nInitialising Winsock...");
    if (WSAStartup(MAKEWORD(2, 2), &wsa) != 0)
    {
        printf("Failed. Error Code : %d", WSAGetLastError());
        exit(EXIT_FAILURE);
    }
    printf("Initialised.\n");

    //create socket
    if ((s = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP)) == SOCKET_ERROR)
    {
        printf("socket() failed with error code : %d", WSAGetLastError());
        exit(EXIT_FAILURE);
    }
    else {
        GetLocalTime(&lt);
        printf("Connection has been started. Time: %02d.%02d.%02d
%02d:%02d:%02d\n\n", lt.wYear, lt.wMonth, lt.wDay, lt.wHour, lt.wMinute, lt.wSecond);
    }

    //setup address structure
    memset((char*)&si_other, 0, sizeof(si_other));

```

```

si_other.sin_family = AF_INET;    si_other.sin_port =
htons(PORT);                    si_other.sin_addr.S_un.S_addr =
inet_addr(SERVER); //start communication
while (1)
{
    printf("Enter message : ");
    char ch;
    for (int i = 0; i < BUFLen; ++i) {
        ch = _getch();
        if (ch == 81) {
            message[i] = '\0';
            break;
        }
        message[i] = ch;
    }
    cout << ch;
    cout << endl;
    //gets_s(message);

    //send the message
    if (sendto(s, message, strlen(message), 0, (struct sockaddr*)&si_other, slen) ==
    SOCKET_ERROR)
    {
        printf("sendto() failed with error code : %d", WSAGetLastError());
        exit(EXIT_FAILURE);
    }
    else {
        GetLocalTime(&lt);
        printf("Message sent successfully. Time: %02d.%02d.%02d
%02d:%02d:%02d\n", lt.wYear, lt.wMonth, lt.wDay, lt.wHour, lt.wMinute, lt.wSecond);
    }

    //receive a reply and print it
    //clear the buffer by filling null, it might have previously received data
    memset(buf, '\0', BUFLen);
    //try to receive some data, this is a blocking call
    if (recvfrom(s, buf, BUFLen, 0, (struct sockaddr*)&si_other, &slen) == SOCKET_ERROR)
    {
        GetLocalTime(&lt);
        printf("Server closed the connection. Time: %02d.%02d.%02d
%02d:%02d:%02d\n", lt.wYear, lt.wMonth, lt.wDay, lt.wHour, lt.wMinute, lt.wSecond);
        exit(EXIT_FAILURE);
    }
    else {
        GetLocalTime(&lt);
        printf("Message received successfully. Time: %02d.%02d.%02d
%02d:%02d:%02d\n", lt.wYear, lt.wMonth, lt.wDay, lt.wHour, lt.wMinute,
lt.wSecond);
        printf("Received message: ");
        puts(buf);
        cout << endl;
    }

}

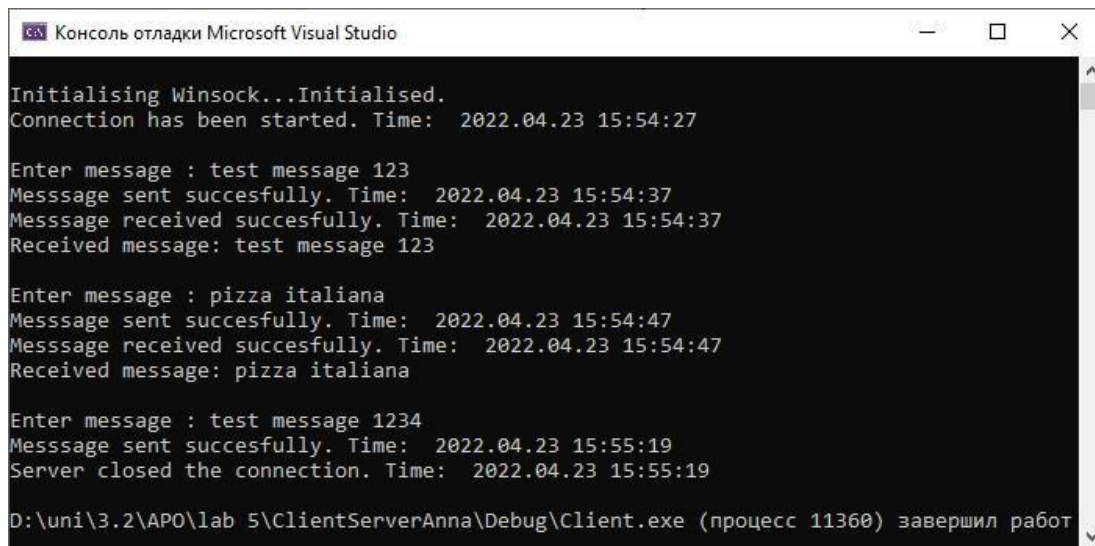
closesocket(s);
WSACleanup();

return 0;
}

```

Результат выполнения программы:

Клиент:



```
Консоль отладки Microsoft Visual Studio

Initialising Winsock...Initialised.
Connection has been started. Time: 2022.04.23 15:54:27

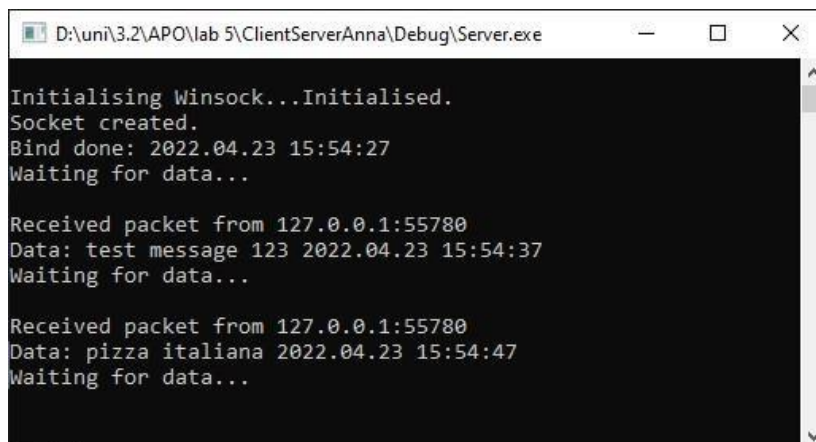
Enter message : test message 123
Message sent succesfully. Time: 2022.04.23 15:54:37
Messsage received succesfully. Time: 2022.04.23 15:54:37
Received message: test message 123

Enter message : pizza italiana
Message sent succesfully. Time: 2022.04.23 15:54:47
Messsage received succesfully. Time: 2022.04.23 15:54:47
Received message: pizza italiana

Enter message : test message 1234
Message sent succesfully. Time: 2022.04.23 15:55:19
Server closed the connection. Time: 2022.04.23 15:55:19

D:\uni\3.2\APO\lab 5\ClientServerAnna\Debug\Client.exe (процесс 11360) завершил работ
```

Сервер:



```
D:\uni\3.2\APO\lab 5\ClientServerAnna\Debug\Server.exe

Initialising Winsock...Initialised.
Socket created.
Bind done: 2022.04.23 15:54:27
Waiting for data...

Received packet from 127.0.0.1:55780
Data: test message 123 2022.04.23 15:54:37
Waiting for data...

Received packet from 127.0.0.1:55780
Data: pizza italiana 2022.04.23 15:54:47
Waiting for data...
```

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 протокола UDP.