Министерство науки и высшего образования Российской Федерации

федеральное государственное автономное   
образовательное учреждение высшего образования   
«Самарский национальный исследовательский университет   
имени академика С.П. Королева»

Институт информатики и кибернетики

Кафедра технической кибернетики

Отчет по лабораторной работе №4

Дисциплина: «Операционные системы»

Тема «Межпроцессное взаимодействие на основе каналов и  
сокетов в операционной системе Windows»

Выполнила: Иванова А.А.

Группа: 6206-010302D

Самара, 2024

**Задание на лабораторную работу**

Задания выполняются на С++ в операционной системе Windows. В качестве дополнительного задания можно избрать любой другой язык программирования и другую операционную систему. Вариант 0 разбирается в следующем разделе. Следует отметить, что число клиентов зависит от задания. Задания выполняются индивидуально. Нечетный вариант (1, 3, 5…) – реализация через именованные каналы, четный вариант (2, 4, 6…) – реализация через сокеты.

Вариант 4.

На сервере запускается n клиентов. После запуска каждый клиент передает сообщение о подключении на сервер. Если подключиться не удалось – нужно передать ошибку и ее код. Сообщения, полученные от разных клиентов нужно различать. Т.е. клиенты должны быть пронумерованы. После создания клиентов: сервер должен «мониторить» в режиме постоянного опроса, запущены клиенты на данный момент или нет. При отключении очередного клиента требуется выдавать информативное сообщение. Не менее 3 клиентов.

**Текст программы**

Сервер

#include <iostream>

#include <iostream>

#include <ctime>

#include <thread>

#include <vector>

//1 шаг

#include <winsock2.h>

#pragma comment(lib, "ws2\_32.lib")

#pragma warning(disable: 4996)

using namespace std;

int ReadInt() {

int n;

std::cout << "Enter the number of clients > 3: ";

while (!(cin >> n) || (n < 3) || (cin.peek() != '\n') || n <= 0) {

cin.clear();

while (cin.get() != '\n');

std::cout << "Incorrect data. Try again > 3: ";

}

return n;

}

std::vector<SOCKET> clientSockets;

void function(SOCKET clientSocket, int clientCount) {

while (true) {

char buffer[1024];

int bytesReceived = recv(clientSocket, buffer, sizeof(buffer), 0);

if (bytesReceived > 0) {

cout << "Message from the client " << clientCount << ": " << buffer << endl;

}

else {

cout << "The client " << clientCount << " is disconnected" << endl;

if (closesocket(clientSocket) == SOCKET\_ERROR)

cerr << "Failed to terminate connection.\n Error code: " << WSAGetLastError();

break;

}

}

}

int main() {

WORD DLLVersion = MAKEWORD(2, 2); // Версия интерфейса Windows Sockets

WSAData wsaData; //создаем структуру для загрузки

//2 шаг. Проверить, можно ли пользоваться сокетами с помощью функции

int Result = WSAStartup(DLLVersion, &wsaData);

if (Result != 0) { // проверка подключения библиотеки

std::cout << "WSAStartup failed with error: " << Result << endl;

return 1;

}

//Заполняем информацию об адресе сокета

SOCKADDR\_IN addr;

int sizeOfAddr = sizeof(addr);

addr.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

addr.sin\_port = htons(8888);

addr.sin\_family = AF\_INET;

addr.sin\_addr.s\_addr = htonl(INADDR\_ANY);

//3 шаг. Создание сокета

SOCKET sListen = socket(AF\_INET, //семейство протоколов

SOCK\_STREAM, //тип создаваемого сокета (использование потоковых сокетов)

IPPROTO\_TCP); //тип протокола (рекомендуется 0, чтобы ос выбрала протокол)

if (sListen == INVALID\_SOCKET) {

std::cout << "Socket failed with error: " << WSAGetLastError() << endl;

WSACleanup();

return 1;

}

int clientCount = ReadInt();

//соединение сервера с локальным адресом

if (bind(sListen, //связываемый сокет

(SOCKADDR\*)&addr, //структура

sizeof(addr)) == SOCKET\_ERROR) //длина структуры в байтах

{

std::cout << "Error bind %d" << WSAGetLastError() << endl;;

closesocket(sListen);

WSACleanup();

return 1;

}

//переводим сокет в режим ожидания

if (listen(sListen,

SOMAXCONN) == SOCKET\_ERROR) {//количество сообщений в очереди

std::cout << "Listening failed with error: " << WSAGetLastError() << endl;

closesocket(sListen);

WSACleanup();

return 1;

}

std::cout << "Waiting for client to connect..." << endl;

for (int i = 1; i <= clientCount; i++) {

// Запускаем .exe файл

STARTUPINFO si;

PROCESS\_INFORMATION pi;

ZeroMemory(&si, sizeof(si));

si.cb = sizeof(si);

ZeroMemory(&pi, sizeof(pi));

if (CreateProcess(L"C:\\Users\\Asus\\Desktop\\24 инф\\LR4\_OC\_2\\x64\\Debug\\LR4\_OC\_2.exe",

NULL, //аргумент командной строки

NULL, //дескриптор процесса не наследуется

NULL, //дескриптор потока не наследуется

FALSE, //будут ли новому процессу наследоваться дескрипторы

CREATE\_NEW\_CONSOLE, //создает новое окно консоли для нового процесса

NULL, //переменные среду окружения

NULL, //полный путь к текущему рабочему каталогу

&si,//указатель на STARTUPINFO

&pi)) { //указатель на PROCESS\_INFORMATION

//5 шаг. Извлечь запросы на подключение к сокету

SOCKET clientSocket = accept(sListen,//дескриптор, который идентифицирует сокет

nullptr, //необязательный указатель на буфер, который получает адрес подключающегося объекта

nullptr); ///необязательный указатель на целое число, которое содержит длину структуры, на которую указывает параметр addr

if (clientSocket == INVALID\_SOCKET) {

std::cerr << "Accept failed.\\n";

closesocket(sListen);

WSACleanup();

return 1;

}

clientSockets.push\_back(clientSocket);

char msg[256];

sprintf(msg, "%d", i);

//передаем сообщение о номере клиента

send(clientSocket, //дескриптор, идентифицирующий подключенный сокет

msg, //указатель на буфер, содержащий данные для передачи

sizeof(msg), NULL);

std::cout << "New client " << i << " connected!" << endl;

char m[256];

recv(clientSocket, m, sizeof(m), 0);

int a = atoi(m);

thread t(function, clientSocket, a);

t.join();

}

else cout << "Error creating process " << i << endl;

}

std::cout << "Server stop working and close connection";

//закрываем сокет и разрываем соединение

if (closesocket(sListen) == SOCKET\_ERROR)

cerr << "Failed to terminate connection.\n Error code: " << WSAGetLastError();

//отключение библиотеки WinSock2 и освобождения ресурсов перед закрытием программы

WSACleanup();

return 0;

}

Клиент

#include <iostream>

#include <string>

#include <winsock2.h>

#include <cstring>

#pragma comment(lib, "ws2\_32.lib")

#pragma warning(disable: 4996)

using namespace std;

int main()

{

//Загрузка библиотеки

WSAData wsaData; //создаем структуру для загрузки

WORD DLLVersion = MAKEWORD(2, 1); //запрашиваемая версия библиотеки winsock

int Result = WSAStartup(DLLVersion, &wsaData);

if (Result != NO\_ERROR) { //проверка подключения библиотеки

cout << "WSAStartup failed with error: " << Result;

return 1;

}

//сокет для соединения с сервером

SOCKET clientSocket = socket(AF\_INET, SOCK\_STREAM, NULL);

//Заполняем информацию об адресе сокета

SOCKADDR\_IN addr;

addr.sin\_addr.s\_addr = inet\_addr("127.0.0.1"); //адрес

addr.sin\_port = htons(8888); //номер порта

addr.sin\_family = AF\_INET; //семейство протоколов

//4 шаг. Соединение с удаленным узлом

if (connect(clientSocket, //подключаемый сокет

(SOCKADDR\*)&addr, //структура

sizeof(addr)) == SOCKET\_ERROR) { //длина структуры в байтах

cout << "Failed to connect to server" << endl;

closesocket(clientSocket);

WSACleanup();

return 1;

}

char msg[256];

recv(clientSocket, msg, sizeof(msg), NULL); //считали с сервера номер клиента

int a = atoi(msg);

cout << "Hi! I am client " << a << "." << endl;

char m[256];

itoa(a, m, 10);

send(clientSocket, m, sizeof(m), NULL);

cout << "Connected!" << endl;

string str;

cin >> str;

char msg\_buffer[256];

strcpy(msg\_buffer, str.c\_str());

int bytesSent = send(clientSocket, msg\_buffer, sizeof(msg\_buffer), NULL);

if (bytesSent == -1) {

cerr << "Error sending data\n";

}

if (closesocket(clientSocket) == SOCKET\_ERROR)

cerr << "Failed to terminate connection.\n Error code: " << WSAGetLastError();

WSACleanup();

return 0;

}