Московский авиационный институт

(Национальный исследовательский университет)

Институт №3:

*Системы управления, информатика и электроэнергетика*

Кафедра 304:  
«Вычислительные машины, системы и сети»

Отчёт по лабораторной работе №2

По учебной дисциплине «WEB технологии»

Группа: М3О-110Б-23

Выполнил:

*Миронов А.Д.*

Принял:

*Титов Ю.П.*

2024 год Москва

# Задание

1. Разработать программу клиент, работающий на сокетах на выбранном Вами языке программирования. Данный клиент должен хранить лог файл соединений. В лог файле хранится время подключения и адрес сервера, время отправки сообщения и само сообщение, время получения сообщения и само сообщение. При подключении к серверу клиент через определенный промежуток времени (задается разработчиком клиента) отправляет сообщение серверу: ФИО и группа студента, а также ждет от сервера. Данные о сервере необходимо вводить через файл конфигурации. В отчете привести код клиента и один тестовый лог файл.
2. Разработать программу сервер, которая ожидает строку от клиента. Через промежуток времени (моделирование работы сервера), он отправляет в ответ эту же строку, только зеркально отраженную и добавляет к сообщению ФИО студента, написавшего сервер. (пример ответа: Чиволвап Йирю Вотит. Сервер написан Титовы Ю.П. М3О-3ХХБ-ХХ). Сервер должен эмулировать работу путем временной задержки. Отключение клиентов от сервера совершать через определенное время работы клиента. Сервер и клиент должны быть написаны на различных языках программирования. Сервер ведет лог-файл, в котором содержатся Время запуска сервера, Время подсоединения клиента, Время получения сообщения и само сообщение, Время отправки сообщения и само сообщение, Время отключения клиента от сервера. Лог-файл не очищать.
3. Доработать программу сервер для выполнения асинхронных запросов от клиентов. Четные варианты реализуют многопоточный асинхронный сервер, нечетные – многопроцессный. Номер варианта определяется номером по списку. При доработке сервера необходимо создать новый проект, т.е. в результате должно получиться две программы сервера.
4. Запустить 5 клиентов в сети кафедры и 1 сервер (5 студентов). В отчете указать настройки для подключения и лог файлы клиентов и серверов. Проводить тестирование одновременного подключения, т.е. второй, третий и т.д. пользователь подключается к серверу пока первый пользователь еще не получил ответ. Желательно провести тестирование как на однопоточном так и на асинхронном сервере, описать отличия.

# Клиент (C++)

#define **WIN32\_LEAN\_AND\_MEAN**

#include <winsock2.h>

#include <ws2tcpip.h>

#include <windows.h>

#include <iostream>

#include <ctime>

#include <fstream>

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

using namespace std;

const char\* HOST = "localhost";

const char\* PORT = "666";

int main(){

system("chcp");

ofstream fout;

fout.open("ClientLog.txt", ios\_base::app);

time\_t rawtime = time(**NULL**);

struct tm \* timeinfo;

WSADATA wsaData;

ADDRINFO hints;

ADDRINFO\* addrResult = **NULL**;

SOCKET ConnectSocket = **INVALID\_SOCKET**;

const char\* sendBuffer = "Mironov Andrei Dmitrievich";

char recvBuffer[1024];

int result = WSAStartup(**MAKEWORD**(2,2), &wsaData);

if(result!=0){

cout << "WSAStartup failed, result = " << result << endl;

return 1;

}

**ZeroMemory**(&hints, sizeof(hints));

hints.ai\_family = **AF\_INET**;

hints.ai\_socktype = **SOCK\_STREAM**;

hints.ai\_protocol = **IPPROTO\_TCP**;

result = getaddrinfo(HOST, PORT, &hints, &addrResult);

if(result!=0){

cout << "getaddrinfo failed with error: " << result << endl;

WSACleanup();

return 1;

}

ConnectSocket = socket(addrResult->ai\_family, addrResult->ai\_socktype, addrResult->ai\_protocol);

if(ConnectSocket == **INVALID\_SOCKET**){

cout << "Socket creatioin failed" << endl;

freeaddrinfo(addrResult);

WSACleanup();

return 1;

}

result = connect(ConnectSocket, addrResult->ai\_addr, (int)addrResult->ai\_addrlen);

if(result == **SOCKET\_ERROR**){

fout << "Unable connect to server\n";

cout << "Unable coonect to server\n";

closesocket(ConnectSocket);

ConnectSocket = **INVALID\_SOCKET**;

freeaddrinfo(addrResult);

WSACleanup();

return 1;

}

fout << "Connection time: " << asctime(localtime(&rawtime));

fout << "Connection host: " << HOST << '\n';

Sleep(5000);

rawtime = time(**NULL**);

result = send(ConnectSocket, sendBuffer, (int)strlen(sendBuffer), 0);

if(result == **SOCKET\_ERROR**){

fout << "Send failed, error: " << result << '\n';

cout << "Send failed, error: " << result << endl;

closesocket(ConnectSocket);

WSACleanup();

return 1;

}

fout << "Sending time: " << asctime(localtime(&rawtime));

fout << "Sent message: " << sendBuffer << '\n';

rawtime = time(**NULL**);

cout << "Bytes sent: " << result << endl;

result = shutdown(ConnectSocket, **SD\_SEND**);

if(result == **SOCKET\_ERROR**){

fout << "Shutdown error: " << result << '\n';

cout << "Shutdown error: " << result << endl;

closesocket(ConnectSocket);

WSACleanup();

return 1;

}

do{

**ZeroMemory**(recvBuffer, sizeof(recvBuffer));

result = recv(ConnectSocket, recvBuffer, sizeof(recvBuffer), 0);

if(result > 0){

rawtime = time(**NULL**);

fout << "Receive time: " << asctime(localtime(&rawtime));

fout << "Received message: " << recvBuffer << '\n';

cout << "Bytes received " << result << endl;

cout << "Received data: " << recvBuffer << endl;

}

else if(result == 0){

fout << "Connection closed" << '\n';

cout << "Connection closed" << endl;

}

else{

fout << "recv failed with error" << '\n';

cout << "recv failed with error" << endl;

}

}while(result > 0);

closesocket(ConnectSocket);

freeaddrinfo(addrResult);

WSACleanup();

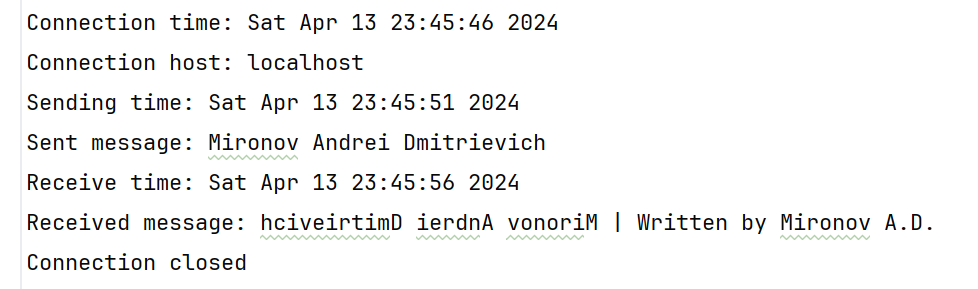
fout << '\n' << endl;

fout.close();

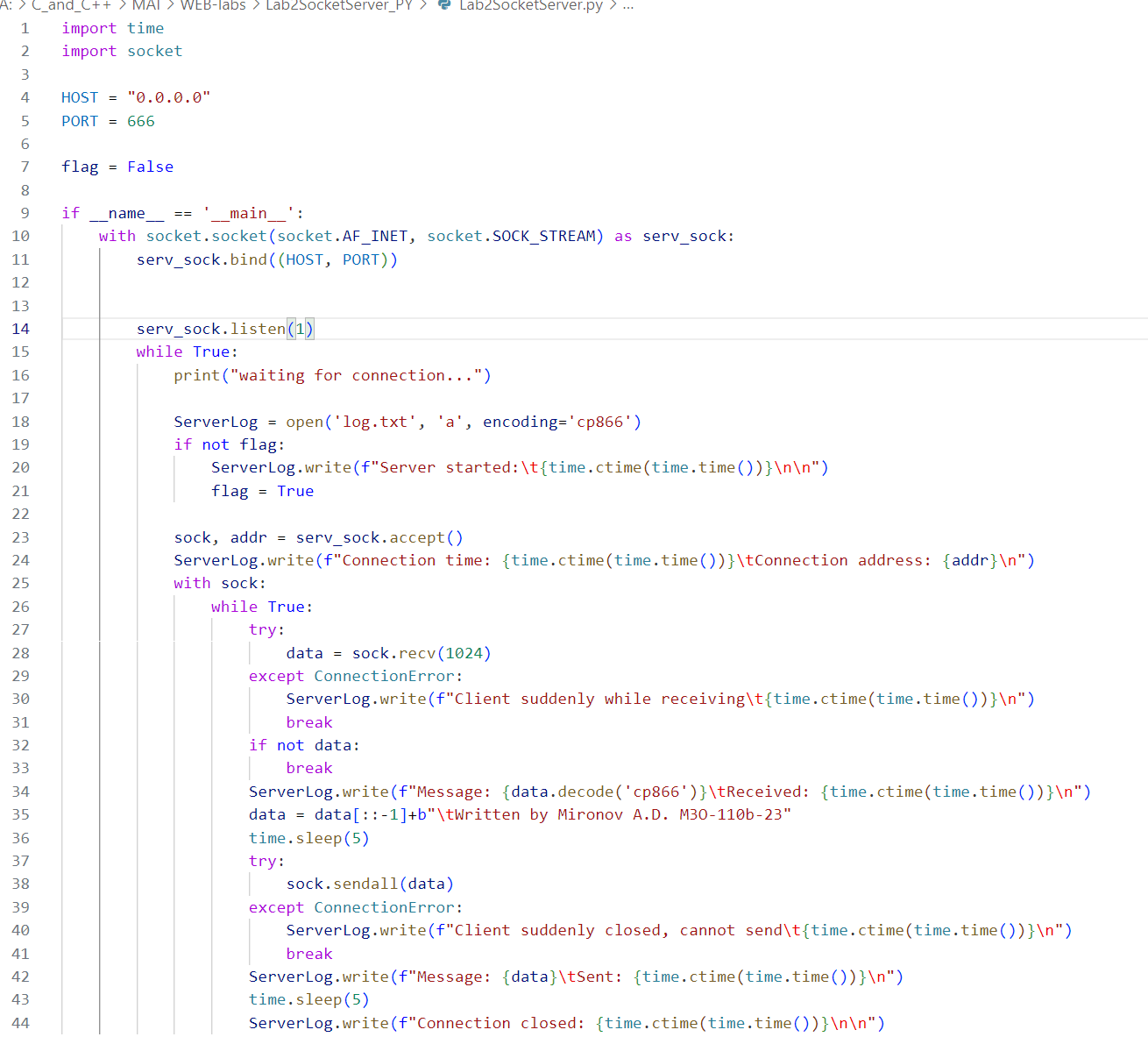
return 0;

}

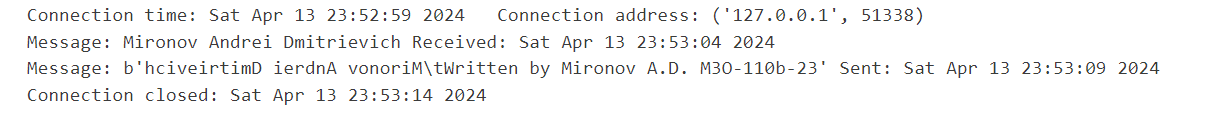
# Лог клиента

**Адрес подключения:** 127.0.0.1:666

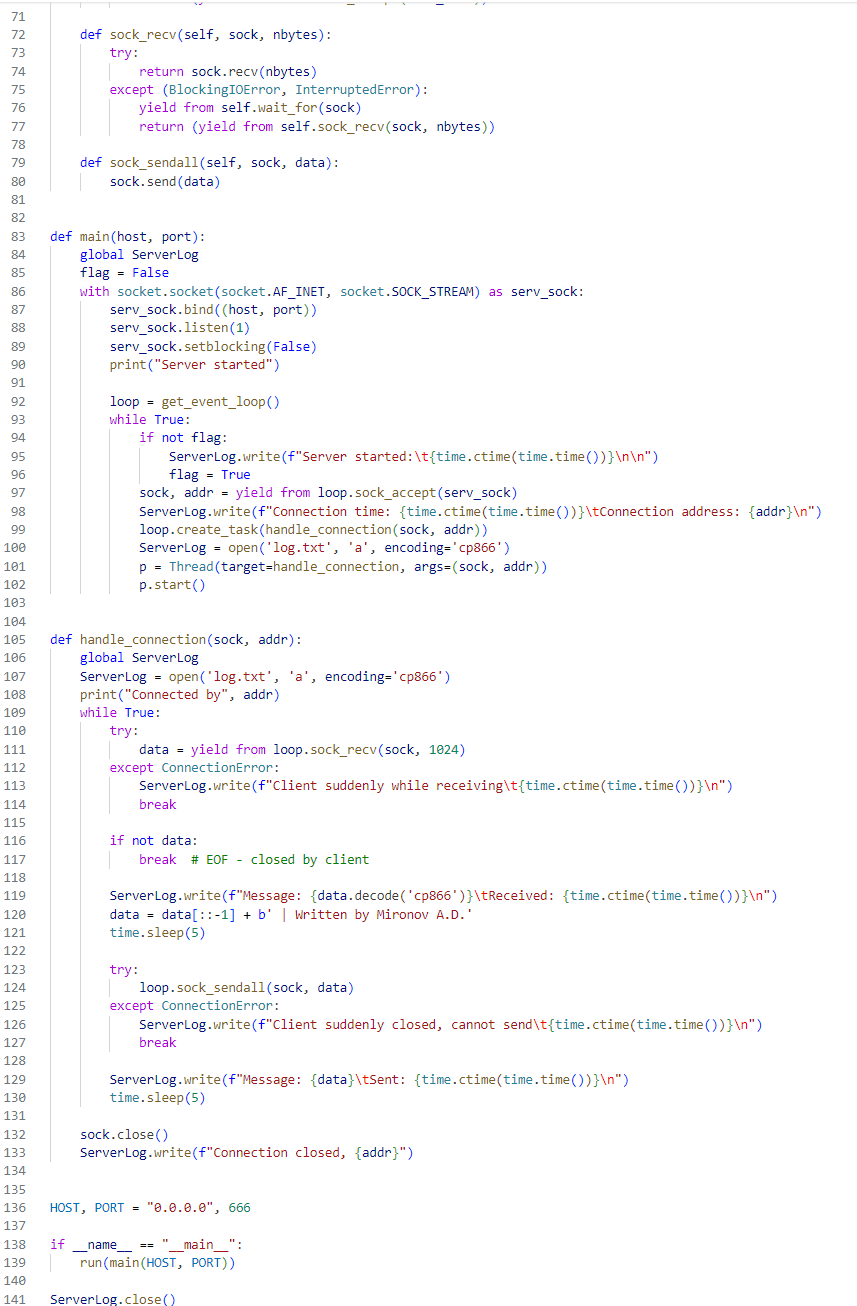
# Сервер (однопоточный) (Python)



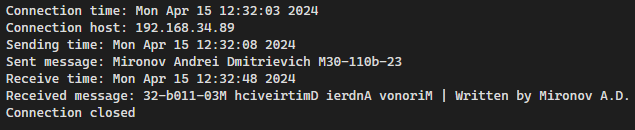
# Лог сервера

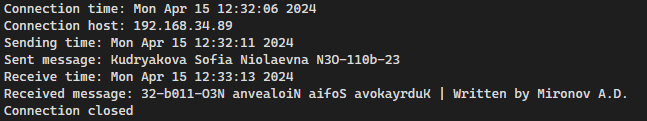


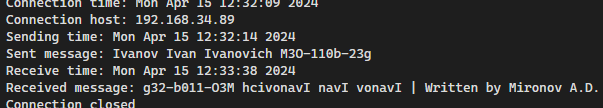
# Сервер (многопоточный) Python

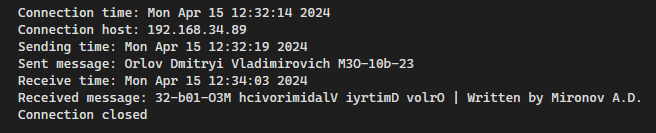


# Лог сервера Логи Клиентов:









# Настройки подключения

**Адрес подключения:** 192.168.34.89:6969