**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 3

по дисциплине «Программирование сетевых приложений»

на тему: «**Многопоточное обслуживание клиентов при**

**организации распределённой обработки информации**

**средствами стека протоколов TCP/IP.**

Выполнил: студент гр. ИТП-41

Климанский П.В.

Принял: ассистент

Гуменников Е.Д.

Гомель 2020

**Цель работы**: изучить подходы к использованию протокола *TCP/IP* средствами *WinSocks* c использованием пула потоков.

Задание для 8 варианта: Оптимизация функции методом Фибоначчи.

**Результат выполнения:**

В ходе выполнения лабораторной работы были написаны код клиента и сервера для решения поставленной задачи.

Клиентское приложения представляет собой простое приложение на C++, выполняющее тривиальную задачу: сформировать данные для нахождения экстремумов функции, передать эти данные через TCP сокет на сервер, получить результат и вывести его в консоль.

Приложение-сервер же реализует указанный в задании пул потоков, на основе которого регулируются поступающие с клиентов запросы. Серверное приложение выполняет следующие задачи:

– ожидание запросов от клиента;

– обработка полученных запросов через пул потоков;

– отправка результата обратно на клиент.

Каждый запрос на сервер обрабатывается в отдельном потоке, управлением которым передано пулу потоков.

На рисунке 1 представлено окно консольного приложения клиента.

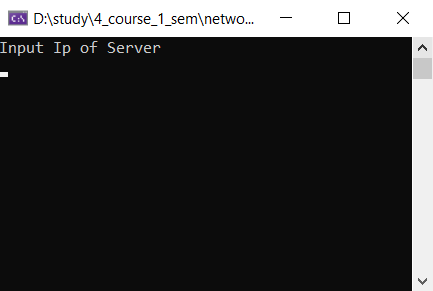
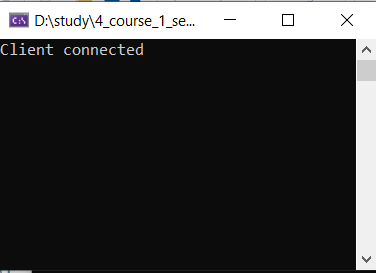


Рисунок 1 – Клиентская часть приложения



На рисунке 2 представлено окно серверного приложения

После ввода начальных условий на клиенте и обработки запроса на сервере, в окне клиента отображается полученный результат, представленный на рисунке 3.

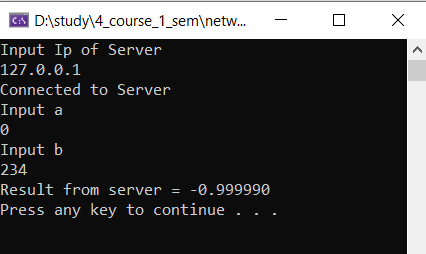


Рисунок 3 – Данные от сервера

**Вывод**: в ходе выполнения лабораторной работы были изучены подходы к обработке множественных запросов на сервер при помощи пула потоков средствами *WinSocks* по протоколу *TCP/IP*.

**ПРИЛОЖЕНИЕ А**

Листинг программы

Файл ***ClientTcp.cpp***:

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

#include <winsock2.h>

#include <windows.h>

#include <iostream>

#include <vector>

#include <sstream>

#pragma warning(disable: 4996)

int main(int argc, char\* argv[])

{

//загрузка версии библиотеки для вызова сетевых функций

WSAData wsaData;

//запрашиваемая версия для библиотеки winsock

WORD DLLVersion = MAKEWORD(2, 1);

//фуекция для загрузки библиотеки

if (WSAStartup(DLLVersion, &wsaData) != 0) {

std::cout << "Error lib\n";

exit(1);

}

char ip\_server[15];

std::cout << "Input Ip of Server\n";

std::cin >> ip\_server;

//std::vector<int> size\_segment(2);

//структура для хранения адреса (для интернет протоколов)

SOCKADDR\_IN socket\_addres;

int sizeof\_socket\_addres = sizeof(socket\_addres);

//для хранения ip-адреса

socket\_addres.sin\_addr.s\_addr = inet\_addr(ip\_server);

//для хранения порта

socket\_addres.sin\_port = htons(5000);

//указание семейства протоколов - af\_inet -сокет интернета

socket\_addres.sin\_family = AF\_INET;

//создаем сокет (параметры: протокол, тип сокета - потоковый или асинхронный

SOCKET client\_Socket = socket(AF\_INET, SOCK\_STREAM, 0);

if (connect(client\_Socket, (SOCKADDR\*)&socket\_addres, sizeof(socket\_addres)) != 0) {

std::cout << "Error connection to Server\n";

return 1;

}

std::cout << "Connected to Server\n";

double a, b;

std::cout << "Input a\n";

std::cin >> a;

std::cout << "Input b\n";

std::cin >> b;

std::stringstream s;

s << a;

s << " ";

s << b;

std::string str = s.str();

const char\* buff2 = str.c\_str();

send(client\_Socket, buff2, std::strlen(buff2) + 1, 0);

double result, s2;

char buff[2048];

recv(client\_Socket, buff, sizeof(double) \* 3 + sizeof(char) \* 2 + 1, 0);

result = strtod(buff, NULL);

printf("Result from server = %f\n", result);

system("pause");

closesocket(client\_Socket);

WSACleanup();

return(0);

}

Файл ***ServerTcp.cpp***:

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

#include <winsock2.h>

#include <windows.h>

#include "ThreadPool.h"

#include <iostream>

#include <vector>

#include <sstream>

#include <chrono>

#pragma warning(disable: 4996)

#define eps 1e-3

//прототип функции

DWORD WINAPI ThreadFunc(LPVOID client\_socket);

SOCKET server\_socket;

double Fun(double x)

{

return (2 \* x \* x \* x - 6 \* x + 3);

}

int F(int n)

{

int f;

int f1 = 1;

int f2 = 1;

int m = 0;

while (m < n - 1)

{

f = f1 + f2;

f1 = f2;

f2 = f;

++m;

}

return f1;

}

double Fib(double a, double b)

{

double x1, x2, \_x, xf1, xf2;

int k = 0;

int N = 0;

double fn1 = 1;

double fn2 = 1;

double fn;

double f = (b - a) / eps;

while (fn1 < f)

{

fn = fn1 + fn2;

fn1 = fn2;

fn2 = fn;

++N;

}

x1 = a + (double)F(N - 2) / F(N) \* (b - a) - (N & 1 ? -1 : 1) \* eps / F(N);

x2 = a + (double)F(N - 1) / F(N) \* (b - a) + (N & 1 ? -1 : 1) \* eps / F(N);

xf1 = Fun(x1);

xf2 = Fun(x2);

P:

++k;

if (xf1 >= xf2)

{

a = x1;

x1 = x2;

xf1 = xf2;

x2 = a + (double)F(N - k - 1) / F(N - k) \* (b - a) + ((N - k) & 1 ? -1 : 1) \* eps / F(N - k);

xf2 = Fun(x2);

}

else

{

b = x2;

x2 = x1;

xf2 = xf1;

x1 = a + (double)F(N - k - 2) / F(N - k) \* (b - a) - ((N - k) & 1 ? -1 : 1) \* eps / F(N - k);

xf1 = Fun(x1);

}

if (fabs(b - a) <= eps)

{

\_x = (a + b) / 2;

double result = Fun(\_x);

return result;

}

else

goto P;

}

DWORD WINAPI ThreadFunc(LPVOID client\_socket)

{

double result, a, b;

SOCKET mySock;

mySock = ((SOCKET\*)client\_socket)[0];

char buff2[2048];

char\* a\_end;

char\* b\_end;

recv(mySock, buff2, sizeof(double) \* 3 + sizeof(char) \* 2 + 1, 0);

a = strtod(buff2, &a\_end);

b = strtod(a\_end, &b\_end);

result = Fib(a, b);

std::stringstream s;

s << result;

std::string str = s.str();

const char\* buff = str.c\_str();

send(mySock, buff, std::strlen(buff) + 1, 0);

return 0;

}

int main(int argc, char\* argv[])

{

//загрузка версии библиотеки для вызова сетевых функций

WSAData wsaData;

//запрашиваемая версия для библиотеки winsock

WORD DLLVersion = MAKEWORD(2, 1);

//функция для загрузки библиотеки

if (WSAStartup(DLLVersion, &wsaData) != 0) {

std::cout << "Error lib\n";

exit(1);

}

//структура для хранения адреса (для интернет протоколов)

SOCKADDR\_IN server\_socket\_addres;

//для хранения ip-адреса

server\_socket\_addres.sin\_addr.s\_addr = 0;

//для хранения порта

server\_socket\_addres.sin\_port = htons(5000);

//указание семейства протоколов - af\_inet -сокет интернета

server\_socket\_addres.sin\_family = AF\_INET;

//создаем сокет (параметры: протокол, тип сокета - потоковый

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

//связываем сокет с его локальным адресом

bind(server\_socket, (SOCKADDR\*)&server\_socket\_addres, sizeof(server\_socket\_addres));

//прослушиваем порт(какой порт + количество клиентов в очереди)

listen(server\_socket, SOMAXCONN);

//принемаем соединение, создавая новый сокет для клиента

SOCKET client\_socket;

SOCKADDR\_IN client\_socket\_addres;

int sizeof\_client\_socket\_addres = sizeof(server\_socket\_addres);

while ((client\_socket = accept(server\_socket, (sockaddr\*)&client\_socket\_addres, &sizeof\_client\_socket\_addres)))

{

if (client\_socket == 0) {

std::cout << "Error with client connection\n";

}

else {

std::cout << "Client connected\n";

}

QueueUserWorkItem(ThreadFunc, &client\_socket, WT\_EXECUTEDEFAULT);

}

closesocket(server\_socket);

system("pause");

WSACleanup();

return(0);

}