МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра информационных технологий**

**Отчёт по лабораторной работе №1**

**«Клиент-серверное приложение с общей средой – бинарным файлом»**

Работу выполнил

студент 22 группы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Н.Е. Лучин

Направление подготовки 01.03.02 Прикладная математика и информатика

Проверил

канд. техн. наук, доц.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.А. Полупанов

Краснодар 2022

**Цель работы:** разработать клиент-серверное приложение с общей средой – бинарным файлом.

**Задача:** приложение состоит из двух программ: client.exe и server.exe. При взаимодействии этих программ используются 2 файла. Файл REQUEST.bin заполняет клиент, он пишет в него свой запрос. Файл ANSWER.bin заполняет сервер, он пишет в него свой ответ на запрос клиента.

Взаимодействие клиента и сервера происходит по следующему сценарию:

Клиент дописывает в конец файла REQUEST.bin свой запрос. Сервер в бесконечном цикле проверяет, появились ли в файле REQUEST.bin новые запросы. Для этого он сравнивает предыдущий размер файла с текущим (в начале работы предыдущий и текущий размеры файлов равны 0).

Если эти размеры совпали – нет новых запросов. Иначе – появился новый запрос, в этом случае сервер считывает его из файла REQUEST.bin, обрабатывает и результат записывает в конец файла ANSWER.bin.

Клиент аналогично постоянно проверяет файл ANSWER.bin на наличие новых ответов от сервера.

**Практическая часть:**

Листинг клиента:

#include <iostream>

#include <fstream>

#include <windows.h> *// для Sleep()*

using namespace std;

struct Person

{

    char name[25];

    int height;

    int weight;

} p;

int answer;

long size\_prev;

int main()

{

*// входной и выходной файловые потоки*

    ofstream fR;

    ifstream fA;

*// указатели на строковые литералы путей*

*const* char\* nameR = "REQUEST.bin";

*const* char\* nameA = "ANSWER.bin";

    while (true)

    {

        cout << "Enter request: name, height & weight"

<< endl;

        cin >> p.name >> p.height >> p.weight;

*/\* открываем в бинарном режиме для записи*

*в конец \*/*

        fR.open(nameR, ios::binary | ios::app);

*/\* записываем данные в файл*

*(преобразовываем структуру в указатель на массив символов) \*/*

        fR.write((char\*) &p, sizeof(p));

        fR.close();

*/\* открываем файл, где ожидаем увидеть ответ от сервера \*/*

        fA.open(nameA, ios::binary);

*// указатель в конец*

        fA.seekg(0, ios::end);

*// запомнили размер файла*

        size\_prev = fA.tellg();

*// ждем увеличения размера файла*

        while (size\_prev >= fA.tellg())

        {

            Sleep(100);

*// указатель в конец файла*

            fA.seekg(0, ios::end);

        }

*/\* ставим указатель на начало данных ответа сервера \*/*

        fA.seekg(size\_prev, ios::beg);

*// считываем данные в answer*

        fA.read((char\*) &answer, sizeof(answer));

        fA.close();

*// интерпретация результата*

        switch (answer)

        {

            case 0:

                cout << "Less than normal\n\n";

                break;

            case 1:

                cout << "Normal\n\n";

                break;

            case 2:

                cout << "More than normal\n\n";

                break;

            default:

                cout << "ERROR\n\n";

                break;

        }

    }

*return* 0;

}

Листинг сервера:

#include <iostream>

#include <fstream>

#include <windows.h> *// для Sleep()*

using namespace std;

struct Person

{

    char name[25];

    int height;

    int weight;

} p;

int answer;

long size\_prev;

int main()

{

*// входной и выходной файловые потоки*

    ifstream fR;

    ofstream fA;

*// указатели на строковые литералы (путей)*

*const* char\* nameR = "REQUEST.bin";

*const* char\* nameA = "ANSWER.bin";

    cout << "server is working\n";

*// открытие файла в бинарном режиме для чтения*

    fR.open(nameR, ios::binary);

*// установка указателя в конец файла (0 - смещение)*

    fR.seekg(0, ios::end);

*/\* предыдущий размер файла*

*(инициализируется текущим) \*/*

    size\_prev = fR.tellg();

    fR.close();

    while (true)

    {

*// снова открываем и ставим указатель в конец*

        fR.open(nameR, ios::binary);

        fR.seekg(0, ios::end);

*/\* пока в файле не появился запрос*

*(размер файла такой же или уменьшился) \*/*

        while (size\_prev >= fR.tellg())

        {

            Sleep(100);

*// указатель в конец файла*

            fR.seekg(0, ios::end);

        } *// вышли из цикла => появился запрос*

*/\* сместим указатель на место перед новой записью,* *т.е. после 'size\_prev' байт от начала файла \*/*

        fR.seekg(size\_prev, ios::beg);

*/\* &p - получить адрес 'p' в памяти*

*(char\*) - преобразовать его в указатель на char*

*первый параметр read() - куда считываем, т.е. считаем в структуру p, к которой обратимся через указатель*

*sizeof() - размер в байтах*

*второй параметр read() - сколько байт считать \*/*

        fR.read((char\*) &p, sizeof(p));

*/\* read сместит указатель.*

*мы обновим size\_prev -*

*поставим на конец записанных данных \*/*

        size\_prev = fR.tellg();

        fR.close();

cout << p.name << " " << p.height << " "

<< p.weight << endl;

double BMI = p.weight / (0.01 \* p.height)

/ (0.01 \* p.height);

        if (18.5 <= BMI && BMI < 25) answer = 1;

        if (18.5 > BMI) answer = 0;

        if (25 <= BMI) answer = 2;

*/\* записываем ответ для клиента –*

*рассчитанный BMI*

*открываем файл в бинарном режиме для записи в конец \*/*

        fA.open(nameA, ios::binary | ios::app);

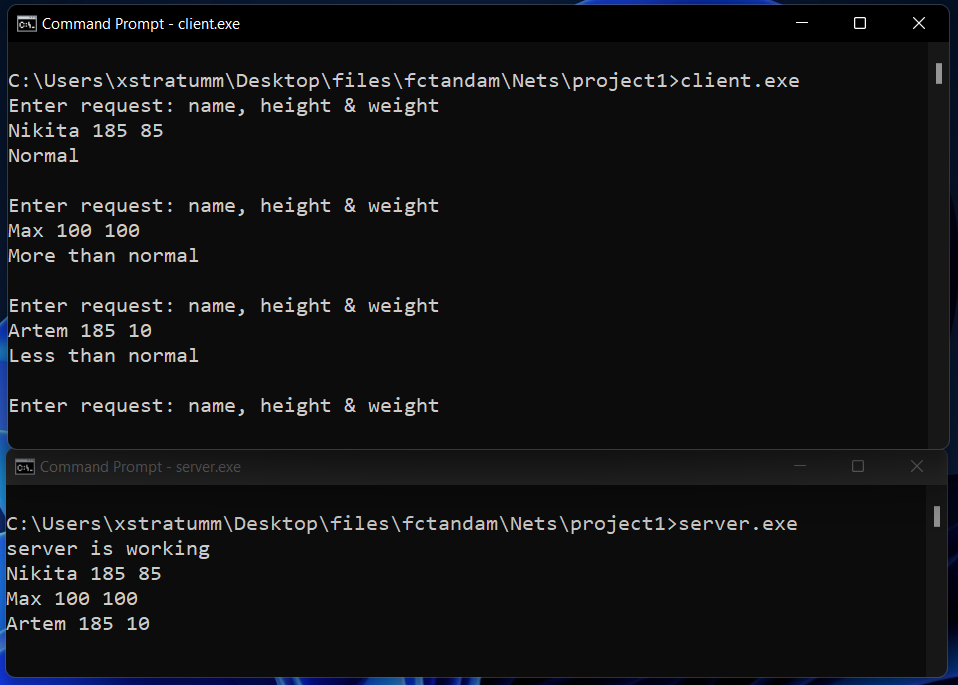
        fA.write((char\*) &answer, sizeof(answer));

        fA.close();

    }

}

Пример работы программ:



*Рисунок 1 – пример работы клиента с сервером*

**Вывод:**былоразработано клиент-серверное приложение с общей средой – бинарным файлом.