Міністэрства адукацыі Рэспублікі Беларусь

Установа адукацыі

«БЕЛАРУСКІ ДЗЯРЖАЎНЫ УНІВЕРСІТЭТ
ІНФАРМАТЫКІ І РАДЫЁЭЛЕКТРОНІКІ»

Інжынерна-эканамічны факультэт
Кафедра эканамічнай інфарматыкі

ЛАБАРАТОРНАЯ РАБОТА №3

на тэму

«СТВАРЭННЕ ПАРАЛЛЕЛЬНАГА ШМАТПАТОЧНАГА СЕРВЕРА З УСТАЛЯВАННЕМ ЛАГІЧНАГА ЗЛУЧЭННЯ TCP»

Падрыхтаваў ст. гр. 210101
Астроўскі Я. А.

Праверыў выкладчык
Карбіт П. А.

Мінск 2024

Мэта работы

Вывучыць метады стварэння серверных прыкладанняў на аснове усталявання лагічнага злучэння *TCP*, выкарыстоўваючы алгарытм шматпаточнай апрацоўкі запросаў.

Заданне

Распрацаваць дадатак, якое рэалізуе архітэктуру»кліент-сервер". Для гэтага неабходна рэалізаваць паралельны шматструменны сервер з усталяваннем лагічнага злучэння (TCP). Логіку ўзаемадзеяння кліента і сервера рэалізаваць так, як паказана ў варыянце індывідуальнага задання. Прадугледзець магчымасць прагляду, дадання, рэдагавання, выдалення інфармацыі кліентам на сэрвэры.

На серверы захоўваецца спіс гульцоў. Кожны запіс спісу змяшчае наступную інфармацыю пра гульцоў:

ПІІ гульца;

Нумар гульца;

узрост;

Такіх запісаў павінна быць не менш за пяць.

Кліент пасылае на сервер запыт і атрымлівае інфармацыю аб самым маладым гульцу.

Серверная частка

```
#include <iostream>
#pragma comment(lib, "ws2_32.lib")
#include <winsock2.h>
#include <vector>
#include <array>
#include <string>
#include <iomanip>

#pragma warning(disable:4996)

using namespace std;

vector<array<string, 5>> information = { {"Иванов Иван
    Иванович","568010","28","168","86"},
                                                {"Горин
    Геннадий Александрович","875078","31","181","96"},

    {"Жидковский Алексей
    Валерьевич","372304","30","177","100"},
```

```
    {"Достоевский Фкдор
    Михайлович","8106544","25","173","89"},

    {"Агапов Алексей Михайлович","212777","30","178","86"}
    };


DWORD WINAPI ThreadFunc(LPVOID client_socket) {
    SOCKET s2;
    s2 = ((SOCKET*)client_socket)[0];
    int choice_person;
    char buf[2];
    char buf1[100];
    char buf2[100];
    array<string, 5> arr;
    string inf;
    int person = 0;
    int parametr = 0;
    string value;
    while (true) {
      recv(s2, buf, sizeof(buf), 0);
      int choice = atoi(buf);
      switch (choice) {
      case 1: case 3: case 4:
           send(s2, to_string(size(information)).c_str(),
    sizeof(to_string(size(information)).c_str()),
    0);//отправить размер вектора information
           for (int i = 0;i < information.size();i++) {
                inf += to_string(i + 1) + ". " +
    information[i][0] + " (" + information[i][1] + "): " +
    information[i][2] + " лет, " + information[i][3] + "
    см, " + information[i][4] + " кг\n";
           }
           cout << inf << "    " << inf.length() << endl;
           send(s2, inf.c_str(), inf.length() + 1,
    0);//отправка этой строки
           inf.clear();
           if (choice == 3) {
                recv(s2, buf1, sizeof(buf1), 0);
                person = atoi(buf1);
                memset(buf1, 0, sizeof(buf1));
                recv(s2, buf1, sizeof(buf1), 0);
                parametr = atoi(buf1);
                memset(buf1, 0, sizeof(buf1));
                int bute = recv(s2, buf1, sizeof(buf1), 0);
                if (strlen(buf1) != 0) {
                     buf1[bute] = '\0';
                     value = buf1;
                     memset(buf1, 0, sizeof(buf1));
                }
```

```
                information[person - 1][parametr - 1] =
    value;
            }
            if (choice == 4) {
                recv(s2, buf, sizeof(buf), 0);
                person = atoi(buf);
                information.erase(information.cbegin() +
    person - 1);
            }
            break;
      case 2:
            for (int i = 0;i <= 5;i++) {
                int bute = recv(s2, buf2, sizeof(buf2), 0);
                if (bute != 0) {
                    if (strlen(buf2) != 0) {
                        buf2[bute] = '\0';
                        //cout << sizeof(buf2) << buf2 <<
    endl;
                        arr[i - 1] = buf2;
                        memset(buf2, 0, sizeof(buf2));
                    }
                }
                else cout << "error!!";
            }
            information.push_back(arr);
            break;
      case 5:
            person = 0;
            parametr = stoi(information[0][3] +
    information[0][4]);
            for (int i = 1;i < information.size();i++) {
                int t = stoi(information[i][3] +
    information[i][4]);
                if (parametr > t) {
                    parametr = t;
                    person = i;
                }
            }
            inf = information[person][0] + "\n";
            send(s2, inf.c_str(), inf.length() + 1, 0);
            inf.clear();
            break;
      }
      memset(buf, 0, sizeof(buf));
    }
    closesocket(s2);
}

    void main(){
      setlocale(LC_ALL, "rus");
      WORD wVersionRequested;
```

```
    WSADATA wsaData;
    wVersionRequested = MAKEWORD(2, 2);

    if (WSAStartup(wVersionRequested, &wsaData) != 0) {
         cout << "Error" << endl;
         exit(1);

    }
    SOCKET s = socket(AF_INET, SOCK_STREAM, 0);
    sockaddr_in local_addr;
    local_addr.sin_family = AF_INET;
    local_addr.sin_port = htons(1280);
    local_addr.sin_addr.s_addr = 0;

    bind(s, (sockaddr*)&local_addr, sizeof(local_addr));
    int c = listen(s, SOMAXCONN);

    cout << "Server receive ready" << endl;
    cout << endl;
    SOCKET client_socket; // сокет для клиента
    sockaddr_in client_addr; // адрес клиента
    int client_addr_size = sizeof(client_addr);
    //char** information;
    while ((client_socket = accept(s,
(sockaddr*)&client_addr, &client_addr_size))) {
         DWORD thID;// thID идентификатор типа DWORD
         CreateThread(NULL, NULL, ThreadFunc,
&client_socket, NULL, &thID);
         //1. дескриптор потока не может быть
унаследован
         //2. размер стека начального потока (тот же
размер, что и при вызове)
         //3. указатель на функцию выполнения потока
         //4. аргумент для потока
         //5. флаги
         //6. указатель на переменную, к-рая получает
идентификатор потока
         ExitThread(thID);
    }
}
```

Кліентская частка

```
#include <iostream>
#pragma comment(lib, "ws2_32.lib")
#include <winsock2.h>
#include<string>


#pragma warning(disable:4996)
```

```
using namespace std;

string CheckInt() {
    string input;
    bool isNum;
    do {
      isNum = true;
      cout << "\nВвод:  ";
      cin >> input;
      for (int i = 0;input[i] != '\0'; i++) {
           if (input[i] < '0' || input[i] > '9') {
                cout << "Неверный ввод! Пожалуйста,
повторите попытку:" << endl;//
                isNum = false;//условие не выполняется
                break;
           }
      }
    } while (!isNum);
    return input;
}
void SendData(SOCKET s, int choice, bool send_all) {
    SetConsoleCP(1251);
    string fio, num, age, height, weight;
    bool str_input;
    if (choice == 1 || send_all) {
      do {
           str_input = true;
           cout << "Введите ФИО \nВвод:";
           string surname, name, otch;
           cin >> surname >> name >> otch;
           fio = surname + " " + name + " " + otch;
           cout << fio << endl;
           for (int i = 0;fio[i] != '\0'; i++) {
                if (fio[i] >= '0' && fio[i] <= '9') {
                     cout << "ФИО не должно содержать
цифр!" << endl;
                     str_input = false;
                     break;
                }
                if (!(fio[i] >= 'А' && fio[i] <= 'я')) {
                     if (fio[i] == ' ') continue;
                     else {
                          cout << "ФИО должно содержать
только кириллицу!" << endl;//
                          str_input = false;
                          break;
                     }
                }
           }
      } while (!str_input);
```

```
        send(s, fio.c_str(), fio.length() + 1, 0);
    }
    if (choice == 2 || send_all) {
        do {
            cout << "Введите номер игрока\n";
            num = CheckInt();
            if (num.length() != 6) cout << "Номер должен
    содержать 6 цифр!\n";
        } while (num.length() != 6);
        send(s, num.c_str(), num.length() + 1, 0);
    }
    if (choice == 3 || send_all) {
        do {
            cout << "Введите возраст игрока\n";
            age = CheckInt();
            if (stoi(age) < 18 || stoi(age) > 110) cout <<
    "Неверный ввод!\n";
        } while (stoi(age) < 18 || stoi(age) > 110);
        send(s, age.c_str(), age.length() + 1, 0);
    }
    if (choice == 4 || send_all) {
        do {
            cout << "Введите рост игрока\n";
            height = CheckInt();
            if (stoi(height) < 150 || stoi(height) > 240)
    cout << "Неверный ввод!\n";
        } while (stoi(height) < 150 || stoi(height) > 240);
        send(s, height.c_str(), height.length() + 1, 0);
    }
    if (choice == 5 || send_all) {
        do {
            cout << "Введите вес игрока\n";
            weight = CheckInt();
            if (stoi(weight) < 50 || stoi(weight) > 150)
    cout << "Неверный ввод!\n";
        } while (stoi(weight) < 50 || stoi(weight) > 150);
        send(s, weight.c_str(), weight.length() + 1, 0);
    }
    cout << "Данные успешно изменены!" << endl;
}

int main() {
    setlocale(LC_ALL, "Russian");
    SetConsoleCP(1251);
    WORD wVersionRequested;
    WSADATA wsaData;
    wVersionRequested = MAKEWORD(2, 2);
    if (WSAStartup(wVersionRequested, &wsaData) != 0) {
        cout << "Error" << endl;
        exit(1);
```

```
}

SOCKADDR_IN peer;
peer.sin_family = AF_INET;
peer.sin_port = htons(1280);
peer.sin_addr.s_addr = inet_addr("127.0.0.1");

SOCKET s = socket(AF_INET, SOCK_STREAM, 0);
if (connect(s, (struct sockaddr*)&peer, sizeof(peer))
!= 0) {
  cout << "Error" << endl;
}
else cout << "connectred!" << endl;
char buf[500];
string person_choice;
string str_choice;
string parametr_choice;
int int_person_choice = 0;
int choice = 0;
int arr_size = 0;
int int_parametr_choice = 0;
int bute;
char size[30];
do {
  cout << "Выберите действие:" << endl;
  cout << "1. Просмотр игроков\n2. Добавление
данных\n3. Редактирование данных\n4. Удаление
данных\n5. Найти младшего"<< endl;
  do {
       str_choice = CheckInt();
       choice = stoi(str_choice);
       if (choice < 0 || choice > 5) {
            cout << "Неверный ввод! Пожалуйста,
повторите попытку:" << endl;
       }
  } while (choice < 0 || choice > 5);
  switch (choice) {
  case 1: case 3: case 4:
       cout << "номер  ФИО (Номер игрока): Возраст,
Рост, Вес" << endl;
       cout << "-------------------------------------
-----------------------------------------" <<
endl;
       send(s, str_choice.c_str(),
str_choice.length(), 0);
       recv(s, size, sizeof(size), 0);
       arr_size = atoi(size);// размер вектора на
сервере
       bute = recv(s, buf, sizeof(buf), 0);
       if (bute != 0) {
            buf[bute] = '\0';
```

```
            cout << buf << endl;
            memset(buf, 0, sizeof(buf));
        }
        else cout << "error!!";
        if (choice == 3) {
            cout << "\nВыберите игрока, значения
которого хотите изменить";
            do {
                person_choice = CheckInt();
                int_person_choice =
stoi(person_choice);
                if (int_person_choice < 1 ||
int_person_choice > arr_size) {
                    cout << "Введенное значение не
входит в диапазон!" << endl;
                }
            } while (int_person_choice < 1 ||
int_person_choice > arr_size);
            cout << "Выберите параметр, который хотите
изменить:" << endl;
            cout << "1. ФИО\n2. Номер игрока\n3.
Возраст\n4. Рост\n5. Вес" << endl;
            do {
                parametr_choice = CheckInt();
                int_parametr_choice =
stoi(parametr_choice);
                if (int_parametr_choice < 1 ||
int_parametr_choice > 5) {
                    cout << "Введенное значение не
входит в диапазон!" << endl;
                }
            } while (int_parametr_choice < 1 ||
int_parametr_choice > 5);
            send(s, person_choice.c_str(),
sizeof(person_choice.c_str()), 0);
            send(s, parametr_choice.c_str(),
sizeof(parametr_choice.c_str()), 0);
            SendData(s, int_parametr_choice, false);
        }
        if (choice == 4) {
            cout << "\nВыберите игрока, значения
которого хотите изменить";
            do {
                person_choice = CheckInt();
                int_person_choice =
stoi(person_choice);
                if (int_person_choice < 1 ||
int_person_choice > arr_size) {
                    cout << "Введенное значение не
входит в диапазон!" << endl;
                }
```

```
            } while (int_person_choice < 1 ||
    int_person_choice > arr_size);
            send(s, person_choice.c_str(),
    person_choice.length(), 0);
            cout << "Данные успешно удалены!" << endl;
        }
        break;
      case 2:
        send(s, str_choice.c_str(),
    sizeof(str_choice.c_str()), 0);
        SendData(s, 0, true);
        break;
      case 5:
        send(s, str_choice.c_str(),
    sizeof(str_choice.c_str()), 0);
        bute = recv(s, buf, sizeof(buf), 0);
        if (bute != 0) {
            buf[bute] = '\0';
            cout << buf << endl;
            memset(buf, 0, sizeof(buf));
        }
        else cout << "error!!";
        break;
    }
    } while (choice != 0);
    closesocket(s);

    WSACleanup();
    return 0;
}
```