Федеральное государственное автономное образовательное учреждение высшего образования   
«Пермский национальный исследовательский политехнический университет»

Лабораторная работа   
«Блоковый ввод-вывод»

Выполнил студент группы ИВТ-23-2Б

Муравьев Дмитрий Александрович  
Проверила: доцент кафедры ИТАС

Ольга Андреева Полякова

2023

1. Постановка задачи:

Вариант - 25

Структура "Студент":

- фамилия, имя, отчество;

- дата рождения;

- домашний адрес;

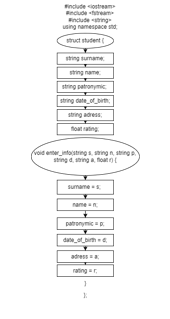
- рейтинг.

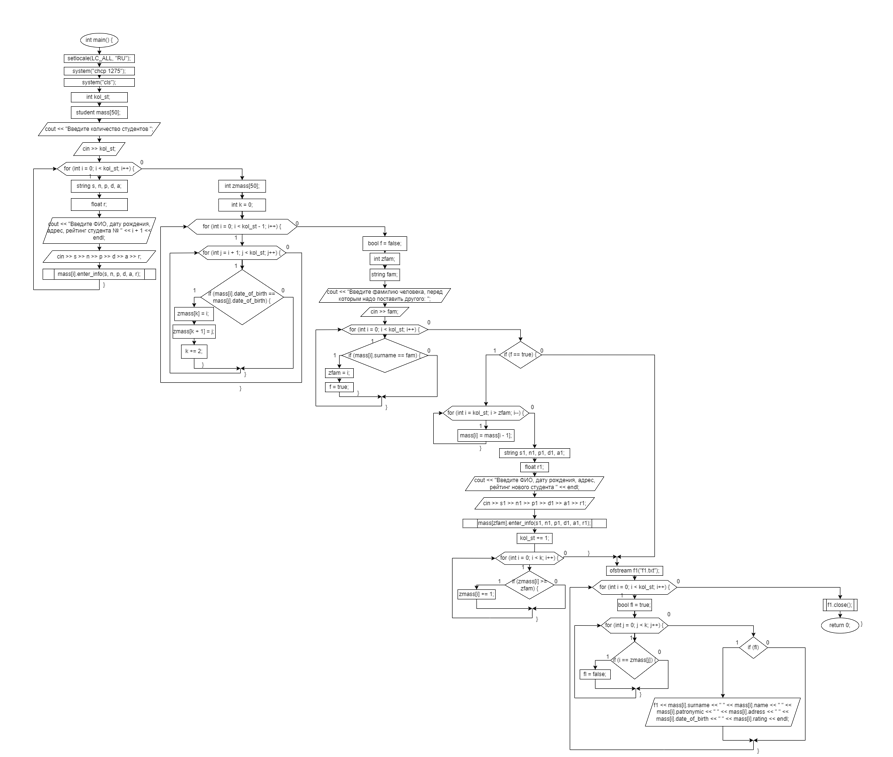
Удалить элементы, у которых даты рождения совпадают, добавить элемент перед элементом с заданной фамилией.

2. Анализ задачи:

* Для работы с данными студентов создаётся структура student, поля которой описывают ФИО, дату рождения, адрес и рейтинг студента.
* Структура содержит метод для инициализации полей элемента.
* Внутри функции main создаётся массив элементов структуры student, обозначается количество студентов. С помощью цикла их данные вводятся с клавиатуры.
* Для хранения индексов элементов, которые нужно удалить из-за совпадения даты рождения, создаётся отдельный массив. Если даты рождения каких-либо студентов совпадают, их индексы помещаются в этот массив.
* Чтобы вставить новый элемент перед элементом с указанной фамилией, остальные элементы сдвигаются вправо на 1 позицию, оставляя место только для нового элемента.
* В файл записываются только те студенты, индексы в массиве у которых не совпадают с запрещёнными, т.е. их даты рождения уникальны.

3. Блок схема





4. Код программы

#include <iostream>

#include <fstream>

#include <string>

#include <windows.h>

using namespace std;

struct student {

string surname; // Фамилия

string name; // Имя

string patronymic; // Отчество

string date\_of\_birth; // Дата рождения

string adress; // Адрес

float rating; // Рейтинг

void enter\_info(string s, string n, string p, string d, string a, float r) {

surname = s; // Функция заполняет поля

name = n; // для одного объекта (студента)

patronymic = p;

date\_of\_birth = d;

adress = a;

rating = r;

}

};

int main() {

setlocale(LC\_ALL, "RU");

system("chcp 1275");

system("cls");

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int kol\_st; // Переменная для количества студентов

student mass[50]; // Массив для 50 студентов

cout << "Введите количество студентов: ";

cin >> kol\_st;

for (int i = 0; i < kol\_st; i++) {

string s, n, p, d, a;

float r; // В цикле вводим данные о студентах

cout << "Введите ФИО, дату рождения, адрес, рейтинг студента № " << i + 1 << endl;

cin >> s >> n >> p >> d >> a >> r;

mass[i].enter\_info(s, n, p, d, a, r);

}

int zmass[50]; // Массив, хранящий индексы элементов, которые нужно удалить

int k = 0; //Количество людей с одной датой рождения - длина запр. массива

for (int i = 0; i < kol\_st - 1; i++) {

for (int j = i + 1; j < kol\_st; j++) {

if (mass[i].date\_of\_birth == mass[j].date\_of\_birth) {

zmass[k] = i; // Цикл сравнивает даты рождения студентов и добавляет

zmass[k + 1] = j; // В массив zmass, который хранит индексы студентов с одной

k += 2; // Датой рождения (все студенты в массиве mass)

}

}

}

bool f = false;

int zfam; // Хранит индекс элемента с указанной фамилией,

string fam; // перед которым нужно вставить ещё один элемент

cout << "Введите фамилию человека, перед которым надо поставить другого: ";

cin >> fam;

for (int i = 0; i < kol\_st; i++) {

if (mass[i].surname == fam) {

zfam = i; // В цикле перебираются все элементы для поиска указанной

f = true; // фамилии, если такая есть, флаг переворачивается

}

}

if (f == true) { // Если f == true, нужно добавить дополнительный элемент

for (int i = kol\_st; i > zfam; i--) {

mass[i] = mass[i - 1]; // В массиве элементы сдвигаются вправо, остаётся место

} // перед элементом с нужной фамилией, его занимает

string s1, n1, p1, d1, a1; // новый элемент, вводимый с клавиатуры

float r1;

cout << "Введите ФИО, дату рождения, адрес, рейтинг нового студента " << endl;

cin >> s1 >> n1 >> p1 >> d1 >> a1 >> r1;

mass[zfam].enter\_info(s1, n1, p1, d1, a1, r1);

kol\_st += 1; // Добавился новый студент

for (int i = 0; i < k; i++) { // k - количество людей с одной датой рождения.

if (zmass[i] >= zfam) { // К индексам элементов для удаления, которые сдвинулись

zmass[i] += 1; // на 1 вправо после добавки нового элемента,

} // прибавляется 1 для сохранения текущих индексов

}

}

// Для каждого элемента проверяется, можно ли его вводить в файл

ofstream f1("f1.txt"); // Включение файла для записи

for (int i = 0; i < kol\_st; i++) {

bool fl = true; // Флаг, нужно ли удалять элемент

for (int j = 0; j < k; j++) {

if (i == zmass[j]) { // Если индекс элемента находится в zmass, флаг

fl = false; // переворачивается, чтобы не пропустить элемент в файл

}

}

if (fl) {

f1 << mass[i].surname << " " << mass[i].name << " " << mass[i].patronymic <<

" " << mass[i].adress << " " << mass[i].date\_of\_birth << " " << mass[i].rating << endl;

}

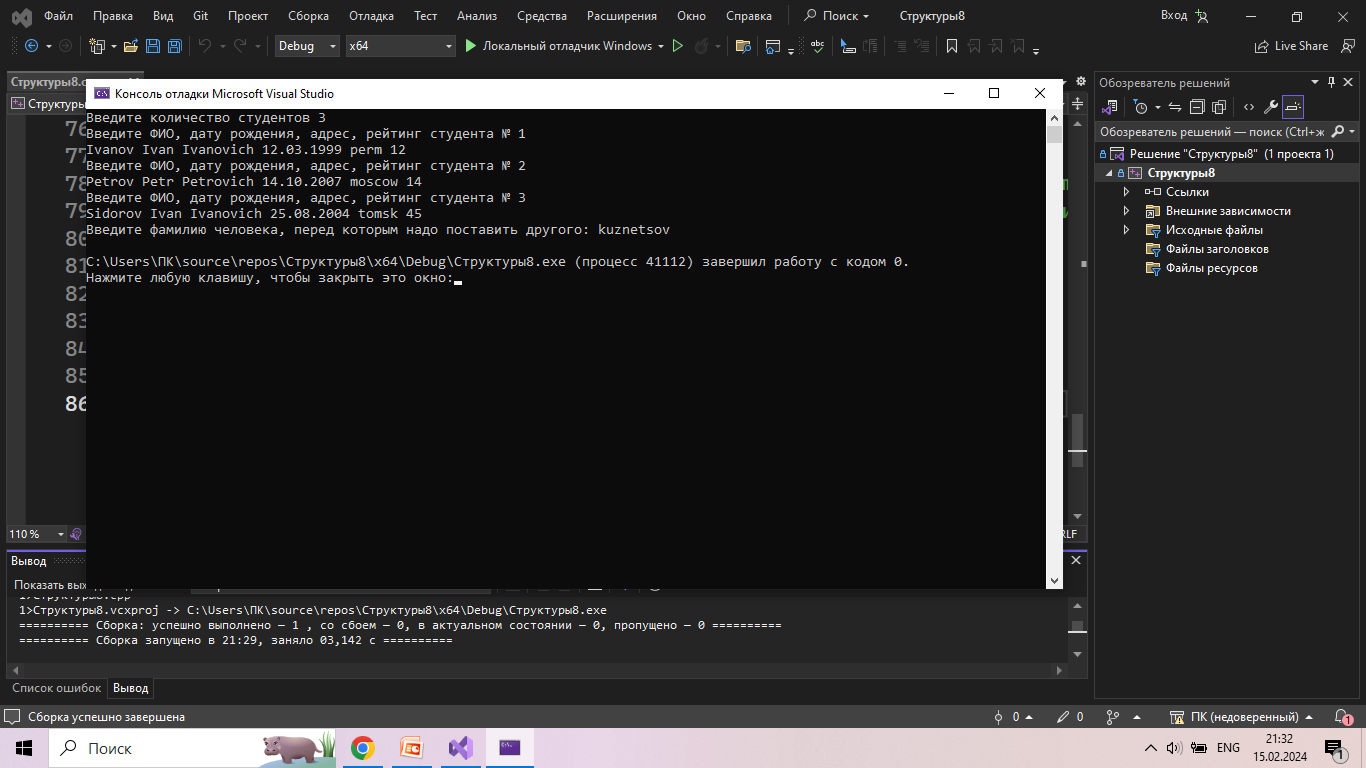
} // Если флаг не перевернулся, элемент записывается в файл

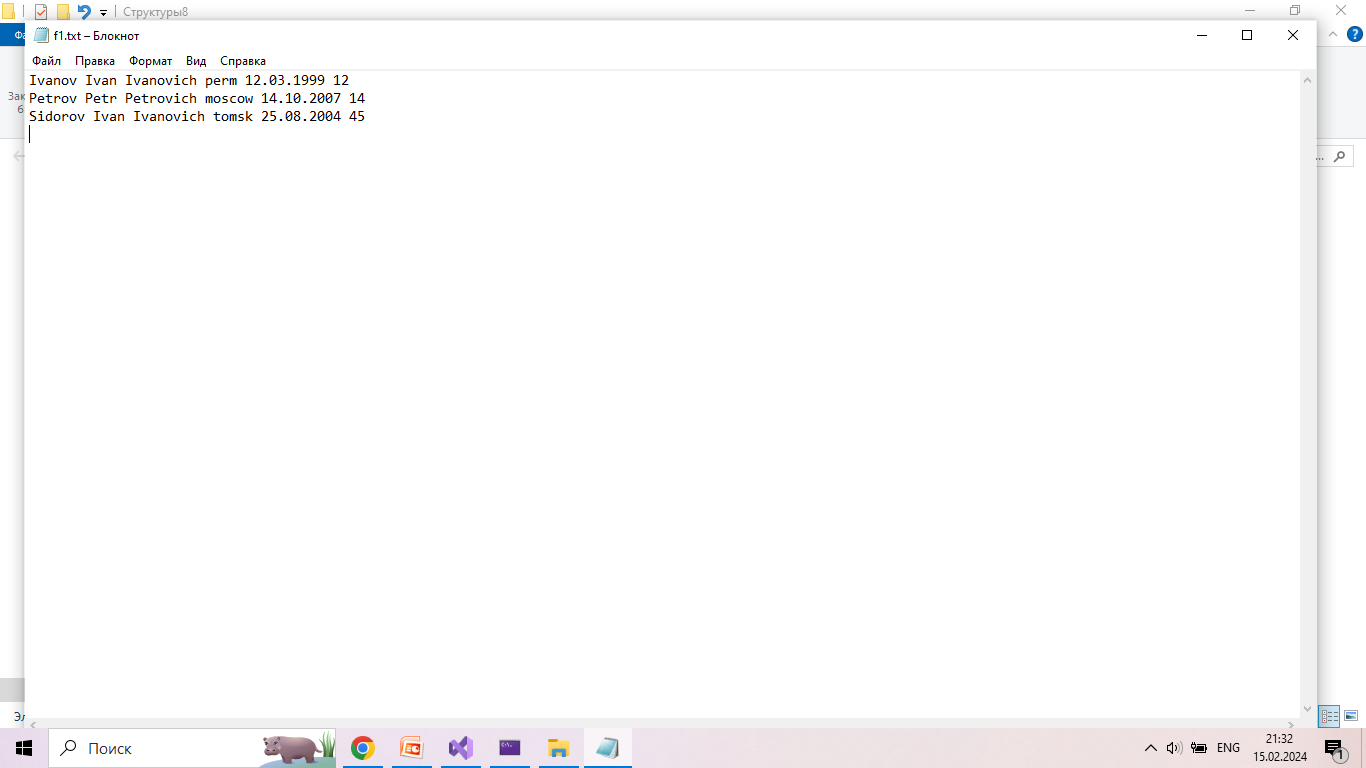
f1.close();

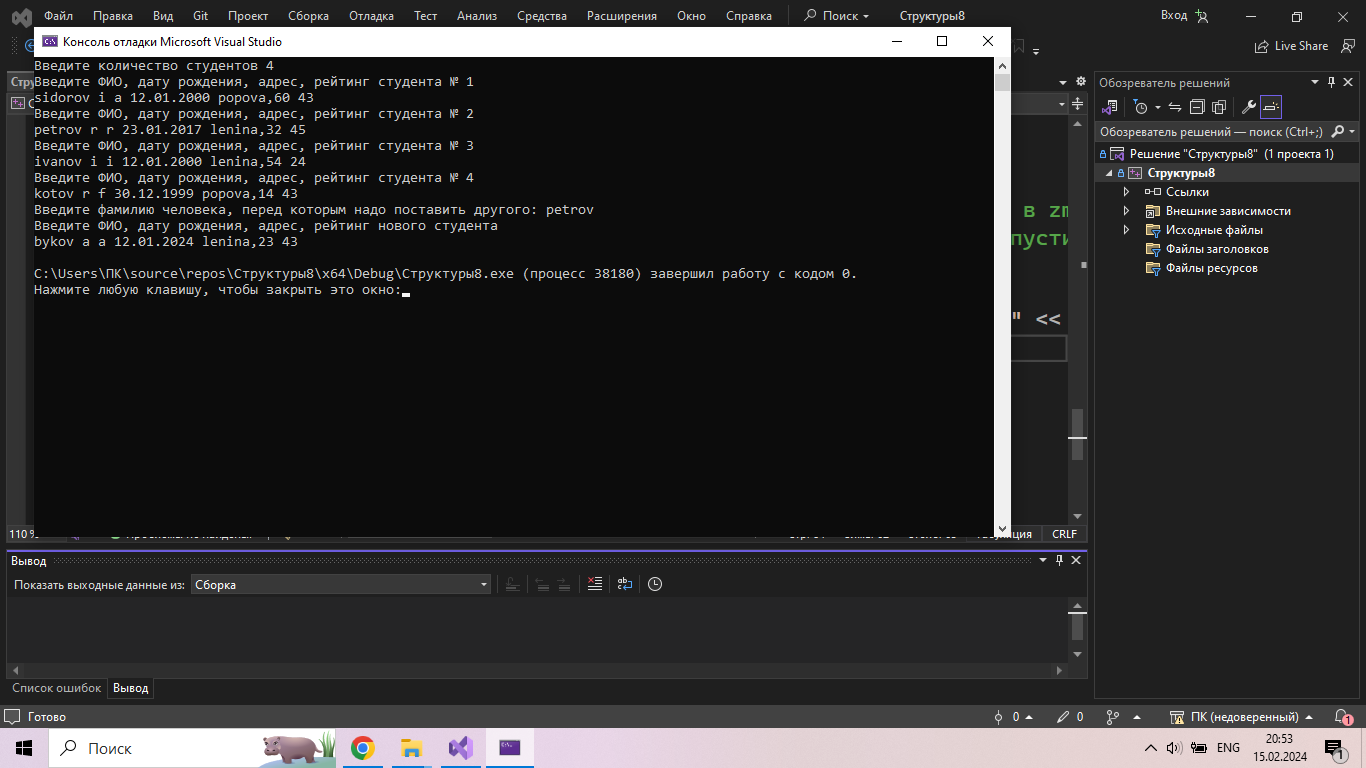
return 0;

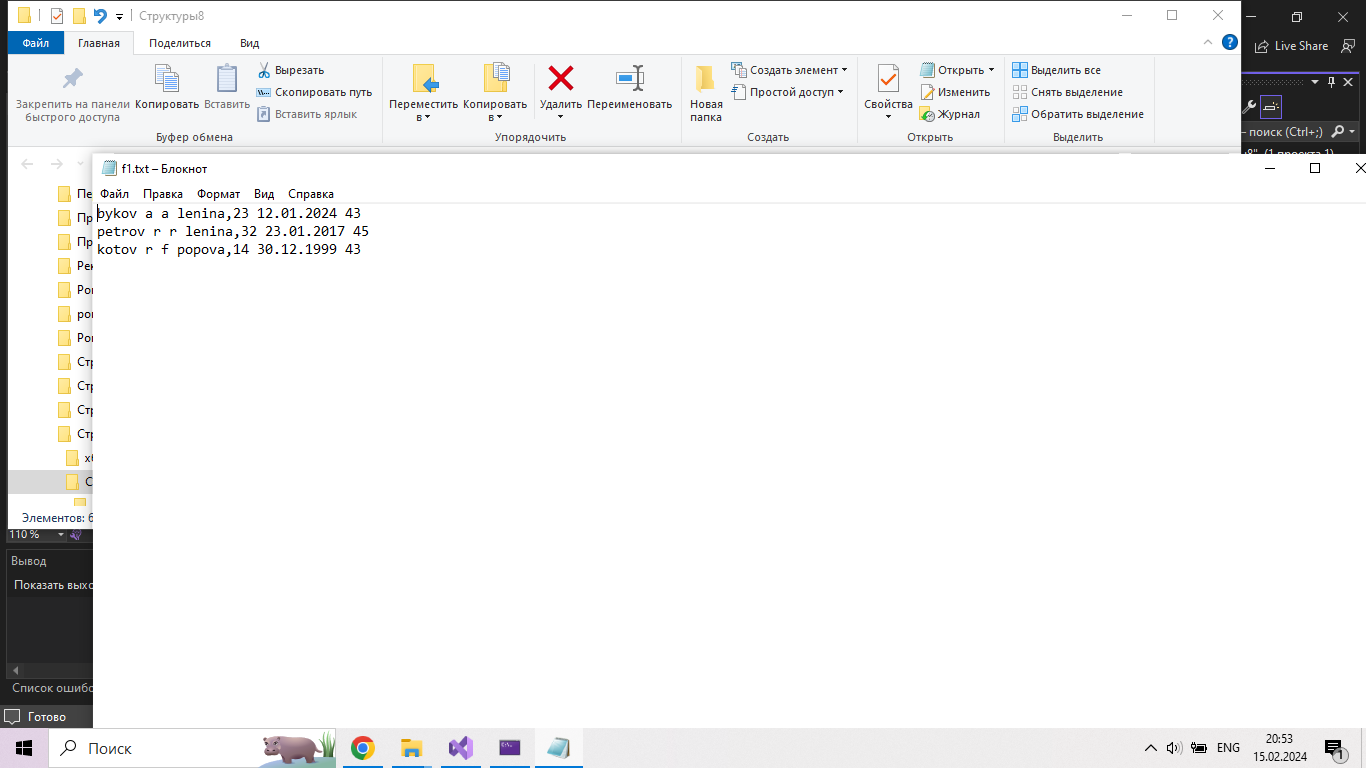
}

5. Вывод программы









6. Вывод

Мну удалось выполнить поставленное задание

7. Github