ЛАБОРАТОРНАЯ РАБОТА №2

«ПРОГРАММИРОВАНИЕ ОПЕРАЦИЙ НАД СТРУКТУРАМИ И БИНАРНЫМИ ФАЙЛАМИ»

* 1. **Цель работы**

Изучение структуры Java-программы.

Формирование навыков программирования алгоритмов линейной и разветвляющейся структуры на языке Java.

Исследование особенностей ввода-вывода значений стандартных типов на языке Java.

* 1. **Вариант задания**

Вариант 12

Описать структуру с именем NOTE, содержащую следующие поля:  порядковый номер;  фамилия, имя;  номер телефона;  дата рождения (массив из трех чисел). Написать программу, выполняющую следующие действия с помощью функций:  ввод с клавиатуры данных в файл, состоящий из элементов типа NOTE; записи должны быть упорядочены по датам рождения;  чтение данных из этого файла и вывод их на экран;  корректировку данных в файле по номеру записи;  вывод на экран информации о человеке, номер телефона которого введен с клавиатуры, а если такого нет, выдать на дисплей соответствующее сообщение.

* 1. **Алгоритм программы**

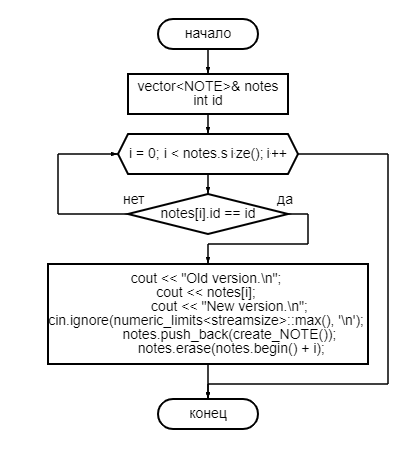
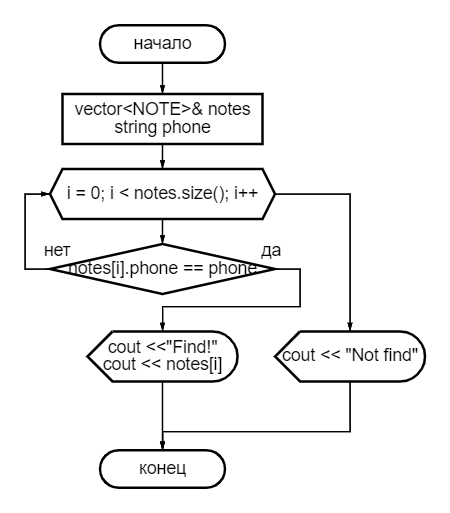


Рисунок 1.2 – Структурная схема работы алгоритма по редактированию записи

Рисунок 1.2 – Структурная схема работы алгоритма по поиску записи по номеру телефона

* 1. **Код программы**

#include <iostream>

#include <string>

#include <algorithm>

#include <vector>

#include <fstream>

#include <regex>

using namespace std;

// структура NOTE

struct NOTE

{

int id;

string name;

string phone;

short date[3];

NOTE(

int id,

string &name,

string &phone,

const short day,

const short month,

const short year) : id(id), name(name), phone(phone), date{day, month, year} {}

short day() const { return date[0]; }

short month() const { return date[1]; }

short year() const { return date[2]; }

// перегрузка операторов сравнения,

// чтобы структуры сравнивались по дате

friend bool operator<(const NOTE &a, const NOTE &b)

{

int a\_days = a.year() \* 365 + a.month() \* 31 + a.day();

int b\_days = b.year() \* 365 + b.month() \* 31 + b.day();

return a\_days < b\_days;

}

friend bool operator>(const NOTE &a, const NOTE &b)

{

int a\_days = a.year() \* 365 + a.month() \* 31 + a.day();

int b\_days = b.year() \* 365 + b.month() \* 31 + b.day();

return a\_days > b\_days;

}

// перегрузка оператора вывода, чтобы печтать информацию о структуре

friend ostream &operator<<(ostream &out, const NOTE &note)

{

out << "ID: " << note.id << "\n";

out << "Name: " << note.name << "\n";

out << "Phone number: " << note.phone << "\n";

out << "Birthday: " << note.day() << "/" << note.month() << "/" << note.year() << "\n";

return out;

}

};

// функция чтение структер из фала

vector<NOTE> read\_NOTES\_from\_file(string filePath)

{

ifstream in(filePath);

string line;

const regex rdelim{","};

vector<NOTE> notes;

if (in.is\_open())

{

while (getline(in, line))

{

// формат записи "id,name,phone,day,month,year"

// разбиваем строку по символам запятой

vector<string> params{

sregex\_token\_iterator(line.begin(), line.end(), rdelim, -1),

sregex\_token\_iterator()};

NOTE note = NOTE(

stoi(params[0]),

params[1],

params[2],

stoi(params[3]),

stoi(params[4]),

stoi(params[5]));

notes.push\_back(note);

}

}

in.close();

return notes;

}

// создание структуры пользователем через ввод из консоли

NOTE create\_NOTE()

{

cout << "Name: ";

string name;

getline(cin, name);

cout << "Phone number: ";

string phone;

getline(cin, phone);

cout << "Day: ";

short day;

cin >> day;

cout << "Month: ";

short month;

cin >> month;

cout << "Year: ";

short year;

cin >> year;

cout << "ID: ";

int id;

cin >> id;

NOTE note = NOTE(id, name, phone, day, month, year);

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout.put('\n');

return note;

}

// открывает файл и сохрнаяет данные о структурах в него

void save\_NOTES\_to\_file(string filePath, vector<NOTE> &notes)

{

// сортировка по дате

std::sort(notes.begin(), notes.end());

std::ofstream out;

out.open(filePath);

if (out.is\_open())

{

for (NOTE note : notes)

{

out << note.id << "," << note.name << "," << note.phone << ','

<< note.day() << "," << note.month() << "," << note.year() << endl;

}

}

}

// функция нахождения NOTE по ID

void edit\_NOTE\_by\_ID(vector<NOTE> &notes, int id)

{

for (int i = 0; i < notes.size(); i++)

{

if (notes[i].id == id)

{

// выводим старую версию

cout << "Old version.\n";

cout << notes[i];

// создаем новую версию

cout << "New version.\n";

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

// добавляем ее

notes.push\_back(create\_NOTE());

// удаляем старую

notes.erase(notes.begin() + i);

return;

}

}

}

// функция поиска по номеру телефона

void find\_NOTE\_by\_phone(vector<NOTE> &notes, string phone)

{

for (int i = 0; i < notes.size(); i++)

{

if (notes[i].phone == phone)

{

cout << "Find!\n";

cout << notes[i];

return;

}

}

cout << "Not find.";

}

// функция отображения всех элементов

void show\_NOTES(vector<NOTE> &notes)

{

for (NOTE note : notes)

{

cout << note;

}

}

int main()

{

// создаем вектор и читаем из файла все элементы

vector<NOTE> notes = read\_NOTES\_from\_file("D:\\text.txt");

// предлагаем пользователю выбор

cout << "Select action:\n"

<< "[1] - Add new NOTE to file\n"

<< "[2] - Show all NOTES from file\n"

<< "[3] - Correct NOTE by ID\n"

<< "[4] - Find NOTE by phone\n";

int select;

cin >> select;

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

// исходя из его выбора, выполняем нужное действие

switch (select)

{

case 1:

notes.push\_back(create\_NOTE());

break;

case 2:

show\_NOTES(notes);

break;

case 3:

cout << "ID: \n";

int id;

cin >> id;

edit\_NOTE\_by\_ID(notes, id);

break;

case 4:

cout << "Phone: \n";

string phone;

getline(cin, phone);

find\_NOTE\_by\_phone(notes, phone);

}

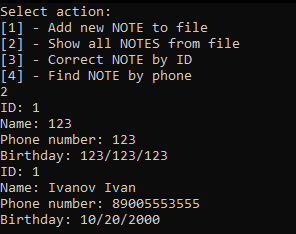
// сохраняем все изменения в файл

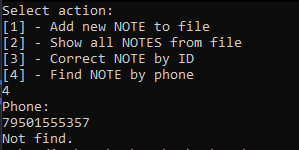
save\_NOTES\_to\_file("D:\\text.txt", notes);

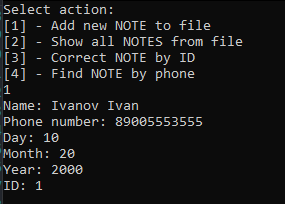
return 0;

}

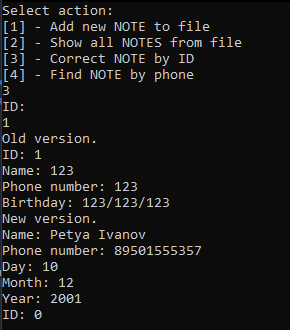
* 1. **Результаты тестирования программы**

В целях тестирования написанной программы были проведены следующие тесты:

Отображение всех записей

Поиск по номер телефона

Добавление новой записи



Изменение записи по ID

**Выводы**

В ходе работы мы познакомились с файловым вводом и выводом в языке программирования С++, научились создавать структуры и делать операции с ними.

Сформировали навыки по поиску и изменению данных.

Так же познакомились с библиотекой С++, которая позволяет работать с файлами.  
Все результаты работы программы отображались в консоли и заносились в файл.