

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования «Рязанский государственный радиотехнический
университет имени В.Ф. Уткина»

Кафедра «ЭВМ»

Отчёт о лабораторной работе № 11

«Файлы»

по курсу

«ОАиООП»

Выполнили:
студенты группы 247
Сидоров А.С.,
Воробьёва М.П.

Проверил:
асс. каф. ЭВМ
Тарасов А.С.,
асс. каф. ЭВМ
Панина И.С.

Рязань, 2023

Цель работы: приобретение навыков работы с файлами.

Вариант №2

Практическая часть

1. Постановка задачи

Описать структуру с именем MARSH, содержащую следующие поля:

- название начального пункта маршрута;
- название конечного пункта маршрута;
- номер маршрута.

Написать программу, в которой происходит ввод с клавиатуры данных в массив, состоящий из восьми элементов типа MARSH, и которая содержит следующие подпрограммы:

- вывод таблицы на экран;
- записи упорядочить по номерам маршрутов;
- вывод отсортированной таблицы на экран;
- вывод на экран информации о маршруте, номер которого введен с клавиатуры. Если таких маршрутов нет, выдать на дисплей соответствующее сообщение;
- сохранить таблицу в файл и прочитать таблицу из файла. Виды файлов: текстовый, бинарный блочным вводом/выводом.

А также оформить вызов подпрограмм в форме меню.

Входные данные: массив из 8 структур MARSH, номер подпрограммы для выполнения

Выходные данные: в зависимости от выбранной подпрограммы.

Рекомендуемый вид экрана во время выполнения программы:

Введите информацию о маршрутах: ...

Меню запросов: ...

2. Математическая модель

Основная программа будет представлять собой цикл до тех пор, пока не будет нажата определённая клавиша. В теле цикла будет выводиться меню запросов, введённое пользователем число будет сравниваться номером запроса.

3. Разработка алгоритма

Блок-схемы программы представлены на рисунке 1.

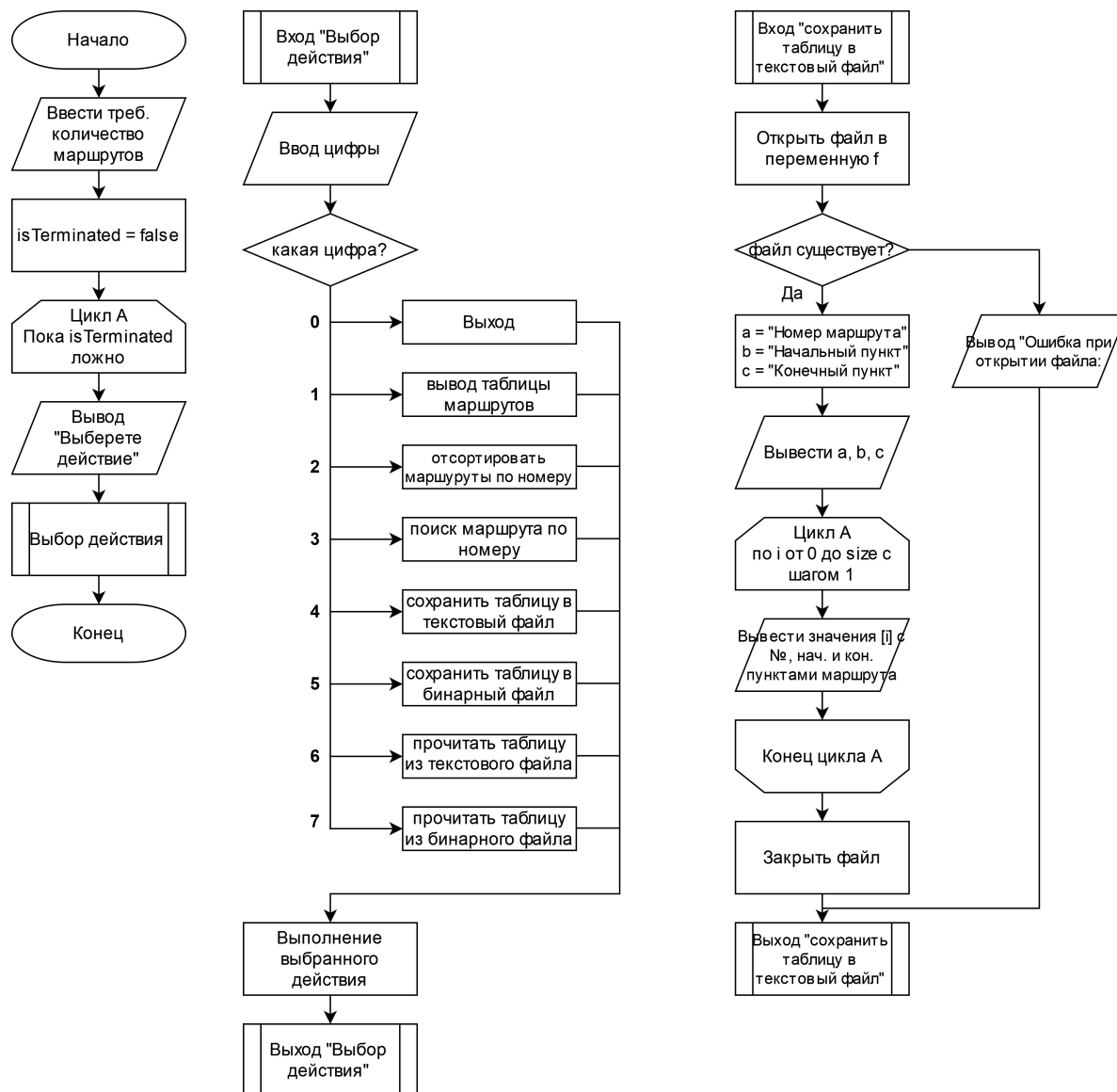


Рисунок 1 – Блок-схемы программы

4. Программирование

Код файла Marsh.h с описанием структуры Marsh:

```
#pragma once
struct Marsh
{
    // Название начального пункта маршрута
    char startName[15];

    // Название конечного пункта маршрута
    char endName[15];

    // Номер маршрута
    int id;

    // Вывести все данные о маршруте в консоль
    void PrintInfo();
};
```

Код файла Marsh.cpp с реализацией структуры Marsh:

```
#include <iostream>
#include "Marsh.h"

void Marsh::PrintInfo()
{
    std::cout << "Номер маршрута: " << id << std::endl;
    std::cout << "Начальный пункт: " << startName << std::endl;
    std::cout << "Конечный пункт: " << endName << std::endl;
}
```

Код файла DataBase.h с описанием структуры DataBase:

```
#pragma once
// Структура, содержащая массив структур Marsh и длину этого массива.
struct DataBase
{
    Marsh* marshes;

    // Длина массива структур
    int size;

    DataBase(int dbSize);
    ~DataBase();

    // Инициализация массива структур пользователем
    void Initialize();

    // Сортировка маршрутов по номеру
    void SortMarshesById();

    // Вывод таблицы маршрутов
    void PrintMarshesTable();

    // Поиск маршрута по номеру
```

```

Marsh SearchById(int toFind);

// Запись таблицы в текстовый файл
void WriteTableIntoTextFile(const char* fileName);

// Запись таблицы в бинарный файл
void WriteTableIntoBinaryFile(const char* fileName);

// Чтение таблицы из текстового файла
void ReadTableFromTextFile(const char* fileName);

// Чтение таблицы из бинарного файла
void ReadTableFromBinaryFile(const char* fileName);
};

```

Код файла DataBase.cpp с реализацией структуры DataBase:

```

#include <iostream>
#include <iomanip>
#include "Marsh.h"
#include "DataBase.h"
using std::cout;
using std::cin;
using std::endl;
using std::setw;

DataBase::DataBase(int dbSize)
{
    marshes = new Marsh[dbSize];
    size = dbSize;
}
DataBase::~DataBase() {
    delete[] marshes;
}

void DataBase::Initialize()
{
    for (int i = 0; i < size; i++)
    {
        cout << "\nМаршрут " << i << endl;
        cout << "Введите название начального пункта маршрута:\n";
        cin >> marshes[i].startName;

        cout << "Введите название конечного пункта маршрута:\n";
        cin >> marshes[i].endName;

        cout << "Введите номер маршрута: ";
        cin >> marshes[i].id;
    }
}

void DataBase::SortMarshesById()
{
    for (int i = 0; i < size - 1; i++)
    {
        for (int j = 0; j < size - i - 1; j++)
        {
            if (marshes[j].id > marshes[j + 1].id)

```

```

        {
            struct Marsh temp = marshes[j];
            marshes[j] = marshes[j + 1];
            marshes[j + 1] = temp;
        }
    }
}

void DataBase::PrintMarshesTable()
{
    const int width = 15;
    // 3 основных столбца + 4 символа '|' + '\0'
    char horizontalLine[width * 3 + 5] =
    {
        "-----"
    };
    // Выравнивание по левой границе
    cout.setf(std::ios::left);

    // Шапка
    cout << horizontalLine << endl;
    cout << "|"
        << setw(width) << "Номер маршрута" << "|"
        << setw(width) << "Начальный пункт" << "|"
        << setw(width) << "Конечный пункт" << "|"
        << endl;
    cout << horizontalLine << endl;

    for (int i = 0; i < size; i++)
    {
        cout << "|"
            << setw(width) << marshes[i].id << "|"
            << setw(width) << marshes[i].startName << "|"
            << setw(width) << marshes[i].endName << "|"
            << endl;

        cout << horizontalLine << endl;
    }
}

Marsh DataBase::SearchById(int toFind)
{
    for (size_t i = 0; i < size; i++)
    {
        if (marshes[i].id == toFind)
        {
            return marshes[i];
        }
    }

    // Маршрут не найден. Предполагается, что
    // маршрутов с отрицательным номером не будет
    return Marsh { "Не найден", "Не найден", -1 };
}

void DataBase::WriteTableIntoTextFile(const char* fileName)
{
    FILE* f = fopen(fileName, "w");
    if (!f)

```

```

{
    perror("Ошибка при открытии файла: ");
}
else
{
    char horizontalLine[15 * 3 + 6] =
    {
        "-----\n"
    };

    // Шапка
    fputs(horizontalLine, f);
    fprintf(f, "%s%-15s%-15s%-15s\n",
            "|", "Номер маршрута",
            "|", "Начальный пункт",
            "|", "Конечный пункт", "|");
    fputs(horizontalLine, f);

    for (int i = 0; i < size; i++)
    {
        fprintf(f, "%s%-15d%-15s%-15s\n",
                "|", marshes[i].id,
                "|", marshes[i].startName,
                "|", marshes[i].endName, "|");
        fputs(horizontalLine, f);
    }

    fclose(f);
}
}

void DataBase::WriteTableIntoBinaryFile(const char* fileName)
{
    FILE* f = fopen(fileName, "wb");
    if (!f)
    {
        perror("Ошибка при открытии файла: ");
    }
    else
    {
        // В файл будут записываться только данные маршрутов
        fwrite(marshes, sizeof(Marsh), size, f);
        fclose(f);
    }
}

void DataBase::ReadTableFromTextFile(const char* fileName)
{
    FILE* f = fopen(fileName, "r");
    if (!f)
    {
        perror("Ошибка при открытии файла: ");
    }
    else
    {
        while (!feof(f))
        {
            char buffer[255]{};
            fgets(buffer, 255, f);
            std::cout << buffer;
        }
    }
}

```

```

        fclose(f);
    }
}

void DataBase::ReadTableFromBinaryFile(const char* fileName)
{
    FILE* f = fopen(fileName, "r");
    if (!f)
    {
        perror("Ошибка при открытии файла: ");
    }
    else
    {
        Marsh* readedMarshes = new Marsh[size];
        fread(readedMarshes, sizeof(Marsh), size, f);

        const int width = 15;

        char horizontalLine[width * 3 + 5] =
        {
            "-----"
        };

        // Шанка
        cout << horizontalLine << endl;
        cout << "|"
            << setw(width) << "Номер маршрута" << "|"
            << setw(width) << "Начальный пункт" << "|"
            << setw(width) << "Конечный пункт" << "|"
            << endl;
        cout << horizontalLine << endl;

        for (int i = 0; i < size; i++)
        {
            cout << "|"
                << setw(width) << readedMarshes[i].id << "|"
                << setw(width) << readedMarshes[i].startName << "|"
                << setw(width) << readedMarshes[i].endName << "|"
                << endl;

            cout << horizontalLine << endl;
        }

        delete[] readedMarshes;
        fclose(f);
    }
}

```

Код основной программы представлен ниже.

```

#include <iostream>
#include "Marsh.h"
#include "DataBase.h"
using std::cout;
using std::cin;
using std::endl;

/*
В лабораторной №10

```


Описать структуру с именем MARSH, содержащую следующие поля:

- название начального пункта маршрута;
- название конечного пункта маршрута;
- номер маршрута.

Написать программу, выполняющую следующие действия:

- ввод с клавиатуры данных в массив, состоящий из восьми элементов типа MARSH;
- вывод таблицы на экран;
- записи упорядочить по номерам маршрутов;
- вывод отсортированной таблицы на экран;
- вывод на экран информации о маршруте, номер которого введен с клавиатуры;
- если таких маршрутов нет, выдать на дисплей соответствующее сообщение.

*/

/*

В лабораторной №11

Доработать:

1. вызов запросов оформить в виде меню;
2. добавить пункты: сохранить таблицу в файл и прочитать таблицу из файла.

Типы файлов:

- текстовый файл;
 - двоичный файл, блочный ввод-вывод.
3. запросы оформить в виде подпрограмм.

*/

```
void FindById(DataBase& db);
```

```
int main()
```

```
{
```

```
    setlocale(LC_ALL, "Russian");
```

```
    cout << "Введите требуемое количество маршрутов: ";
```

```
    int dbSize;
```

```
    cin >> dbSize;
```

```
    struct DataBase db(dbSize);
```

```
    db.Initialize();
```

```
    bool isTerminated = false;
```

```
    while (!isTerminated)
```

```
    {
```

```
        cout << "\nВыберите действие (введите значение):"
```

```
            << "\n 0 - выход;"
```

```
            << "\n 1 - вывод таблицы маршрутов;"
```

```
            << "\n 2 - отсортировать маршруты по номеру;"
```

```
            << "\n 3 - поиск маршрута по номеру;"
```

```
            << "\n 4 - сохранить таблицу в текстовый файл;"
```

```
            << "\n 5 - сохранить таблицу в бинарный файл;"
```

```
            << "\n 6 - прочитать таблицу из текстового файла;"
```

```
            << "\n 7 - прочитать таблицу из бинарного файла.\n";
```

```
        int userChoice;
```

```
        cin >> userChoice;
```

```
        switch (userChoice)
```

```
        {
```

```
            case 0:
```

```
                isTerminated = true;
```

```
                break;
```

```
            case 1:
```

```

        db.PrintMarshesTable();
        break;
    case 2:
        db.SortMarshesById();
        break;
    case 3:
        FindById(db);
        break;
    case 4:
    {
        char filePath[255]{ "H:\\test.txt" };
        //std::cin >> filePath;
        db.WriteTableIntoTextFile(filePath);
        break;
    }
    case 5:
    {
        char filePath[255]{ "H:\\test.bin" };
        //std::cin >> filePath;
        db.WriteTableIntoBinaryFile(filePath);
        break;
    }
    case 6:
    {
        char filePath[255]{ "H:\\test.txt" };
        //std::cin >> filePath;
        db.ReadTableFromTextFile(filePath);
        break;
    }
    case 7:
    {
        char filePath[255]{ "H:\\test.bin" };
        //std::cin >> filePath;
        db.ReadTableFromBinaryFile(filePath);
        break;
    }
    default:
        isTerminated = true;
    }
}
system("pause");
}

void FindById(DataBase& db) {
    cout << "\nВведите номер маршрута для поиска" << endl;
    int toFind;
    cin >> toFind;

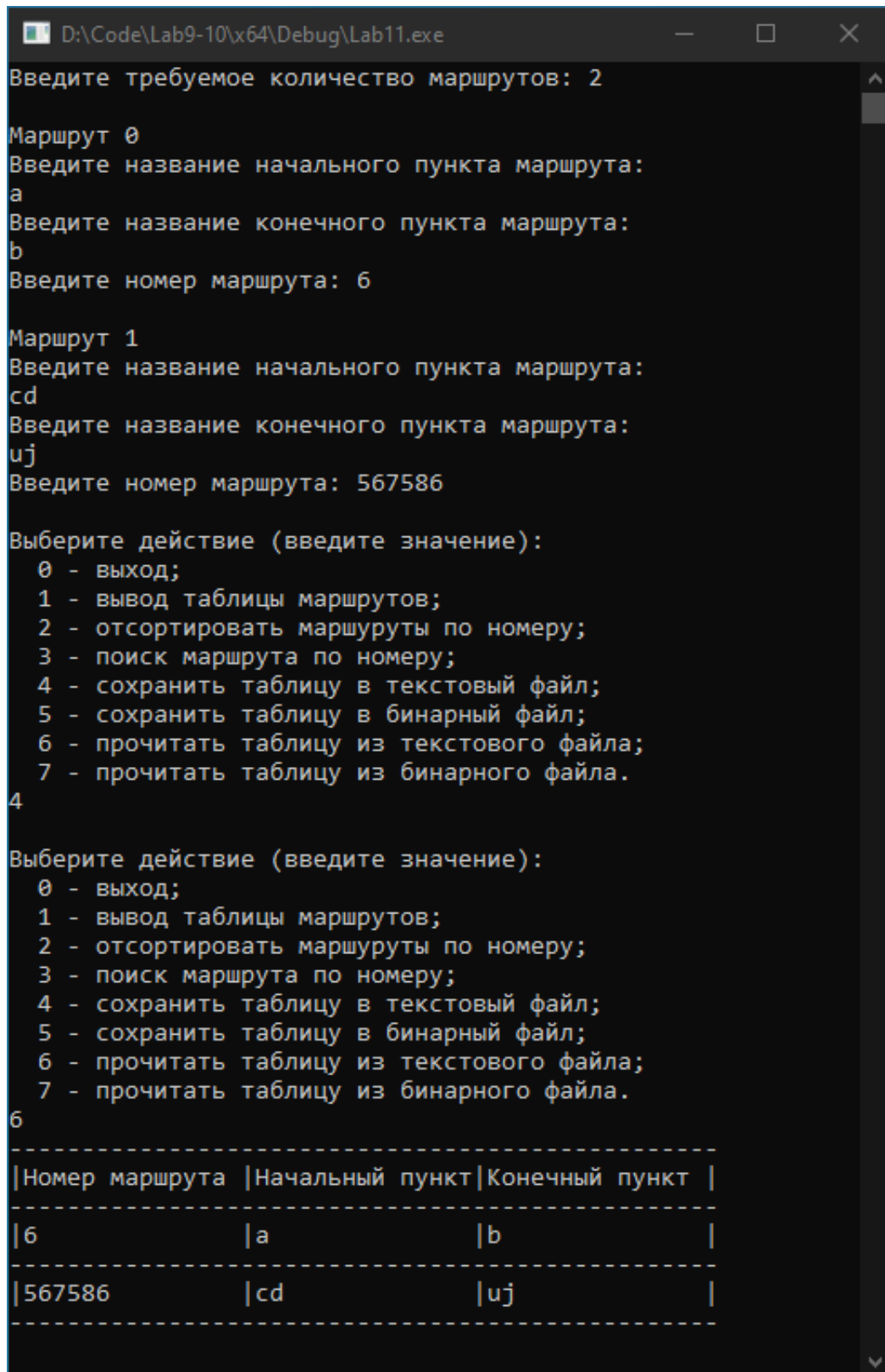
    Marsh found = db.SearchById(toFind);

    if (found.id >= 0)
    {
        found.PrintInfo();
    }
    else
    {
        cout << "Маршрут с номером " << toFind << " не найден" << endl;
    }
}

```

5. Тестирование

Результат программы представлен на рисунках 2 и 3.



```
D:\Code\Lab9-10\x64\Debug\Lab11.exe
Введите требуемое количество маршрутов: 2

Маршрут 0
Введите название начального пункта маршрута:
a
Введите название конечного пункта маршрута:
b
Введите номер маршрута: 6

Маршрут 1
Введите название начального пункта маршрута:
cd
Введите название конечного пункта маршрута:
uj
Введите номер маршрута: 567586

Выберите действие (введите значение):
0 - выход;
1 - вывод таблицы маршрутов;
2 - отсортировать маршруты по номеру;
3 - поиск маршрута по номеру;
4 - сохранить таблицу в текстовый файл;
5 - сохранить таблицу в бинарный файл;
6 - прочитать таблицу из текстового файла;
7 - прочитать таблицу из бинарного файла.
4

Выберите действие (введите значение):
0 - выход;
1 - вывод таблицы маршрутов;
2 - отсортировать маршруты по номеру;
3 - поиск маршрута по номеру;
4 - сохранить таблицу в текстовый файл;
5 - сохранить таблицу в бинарный файл;
6 - прочитать таблицу из текстового файла;
7 - прочитать таблицу из бинарного файла.
6

-----
|Номер маршрута |Начальный пункт|Конечный пункт |
|6              |a              |b              |
|567586         |cd             |uj             |
|-----|-----|-----|
```

Рисунок 2 – Результат программы, часть 1

```
D:\Code\Lab9-10\x64\Debug\Lab11.exe
Выберите действие (введите значение):
0 - выход;
1 - вывод таблицы маршрутов;
2 - отсортировать маршруты по номеру;
3 - поиск маршрута по номеру;
4 - сохранить таблицу в текстовый файл;
5 - сохранить таблицу в бинарный файл;
6 - прочитать таблицу из текстового файла;
7 - прочитать таблицу из бинарного файла.
5
Выберите действие (введите значение):
0 - выход;
1 - вывод таблицы маршрутов;
2 - отсортировать маршруты по номеру;
3 - поиск маршрута по номеру;
4 - сохранить таблицу в текстовый файл;
5 - сохранить таблицу в бинарный файл;
6 - прочитать таблицу из текстового файла;
7 - прочитать таблицу из бинарного файла.
7
-----
| Номер маршрута | Начальный пункт | Конечный пункт |
|-----|-----|-----|
|          6 |          a |          b |
|-----|-----|-----|
|      567586 |          cd |          u |
|-----|-----|-----|
```

Рисунок 3 – Результат программы, часть 2

Заключение

В этой лабораторной работе были получены навыки работы с файлами.