



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт информационных технологий

Кафедра вычислительной техники

ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №11

по дисциплине

«Алгоритмические основы обработки данных»

Выполнил студент группы ИВБО-01-20

Д.А. Манохин

Принял старший преподаватель

Ю.С. Асадова

Практические работы выполнены

«__»_____2021г.

«Зачтено»

«__»_____2021г.

Москва 2021



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт информационных технологий

Кафедра вычислительной техники

Выполнено _____/Д.А. Манохин/

Зачтено _____/Ю.С. Асадова/

Задание на практическую работу №11

Дисциплина: «Алгоритмические основы обработки данных»

Студент Манохин Дмитрий Александрович Шифр 20И2132 Группа ИББО-01-20

1. Тема: «Файлы».

2. Срок сдачи студентом законченной работы: 11.11.2021г.

3. Исходные данные:

Файл. Заказ (номер заказа, ФИО заказчика, товар в заказе, количество товара, сумма заказа).

4. Задание:

Разработать программу, выполняющую следующие функции:

- ввод данных об n объектах из текстового файла в массив структур ($0 < n \leq 50$);
- сортировку массива структур по возрастанию значений одного из полей структуры;
- вывод данных об объектах на экран в упорядоченном по возрастанию виде;
- поиск объекта по значению одного из полей;
- запись упорядоченных данных об объектах в двоичный файл;
- чтение двоичного файла.

5. Содержание отчета:

- титульный лист;
- задание;
- оглавление;
- введение;
- основные разделы отчета;
- заключение;
- список использованных источников;

Руководитель работы

Ю.С. Асадова

_____ «___» _____ 2021г.
подпись

Задание принял к исполнению

Д.А. Манохин

_____ «___» _____ 2021г.
подпись

ОГЛАВЛЕНИЕ

Введение.....	4
Основной раздел.....	5
Заключение	17
Список использованных источников	18

ВВЕДЕНИЕ

В данной практической работе упорядочить объекты из файла и записать их обратно в файл.

Постановка задачи:

Разработать программу, выполняющую следующие функции:

- ввод данных об n объектах из текстового файла в массив структур ($0 < n \leq 50$);
- сортировку массива структур по возрастанию значений одного из полей структуры;
- вывод данных об объектах на экран в упорядоченном по возрастанию виде;
- поиск объекта по значению одного из полей;
- запись упорядоченных данных об объектах в двоичный файл;
- чтение двоичного файла.

Алгоритмы чтения файла, сортировки, поиска, вывода данных об объектах и записи данных в файл оформить в виде функций. Для поиска элемента в упорядоченном массиве использовать бинарный поиск. Текстовый файл создать с помощью любого текстового редактора.

ОСНОВНОЙ РАЗДЕЛ

В начале работы программы пользователь вводит имя файла, из которого будет браться информация для массива из объектов о заказах, если такого файла нет на компьютере, то будет выведена ошибка и произведен досрочный выход из программы.

Если данные были введены правильно, то будет прочитан файл и записаны данные в массив.

Далее сформированный массив будет отсортирован по номеру заказа каждого объекта и можно будет воспользоваться бинарным поиском для нахождения определенного номера заказа, который пользователь вводит с клавиатуры, если заказа с таким номером не существует, то будет выведено соответствующее сообщение.

В конце будет записан отсортированный массив в новый текстовый файл, а также в бинарный файл и выведено сообщение о конце работы программы.

После работы программы и вывода результатов ее работы будет выполнен выход из нее.

Блок – схема алгоритма представлена на рисунках [1.1](#), [1.2](#), [1.3](#), [1.4](#), [1.5](#), [1.6](#), [1.7](#), [1.8](#).

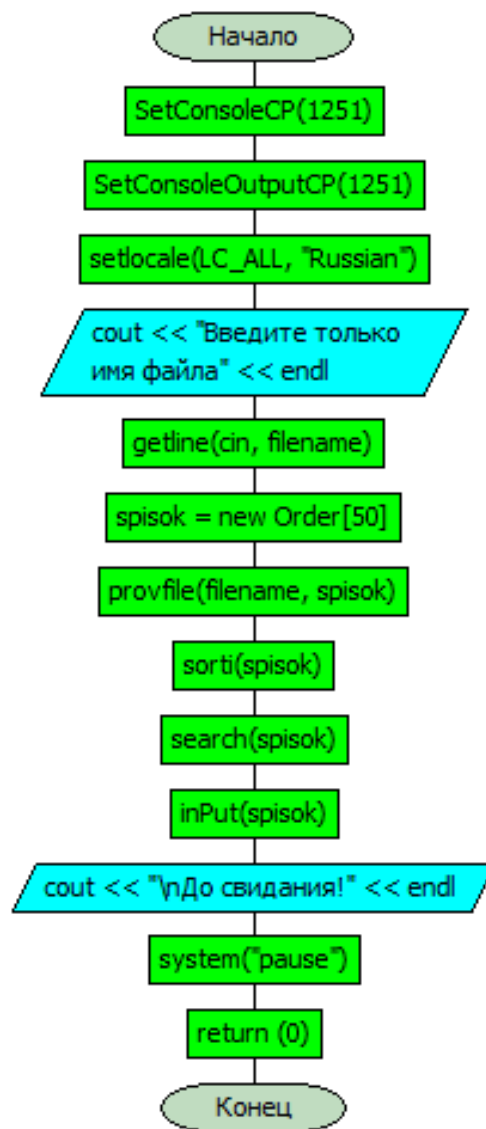


Рисунок 1.1 – Блок-схема алгоритма – функция main

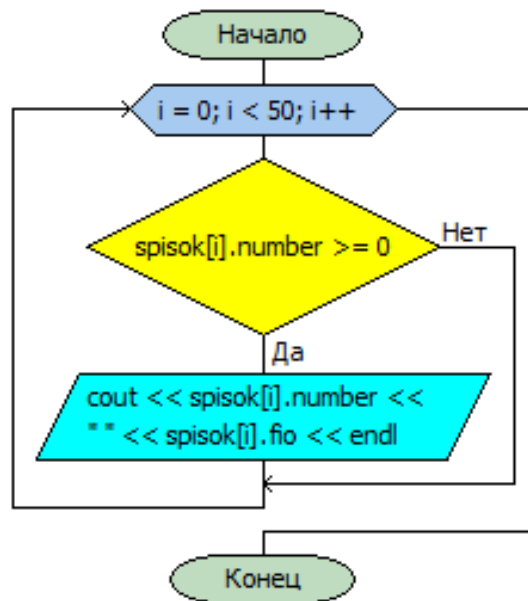


Рисунок 1.2 – Блок-схема алгоритма – функция outPut

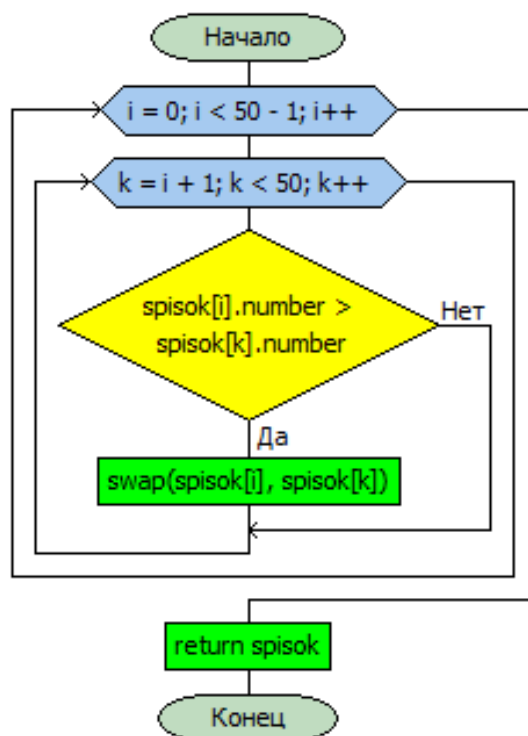


Рисунок 1.3 – Блок-схема алгоритма – функция sorti

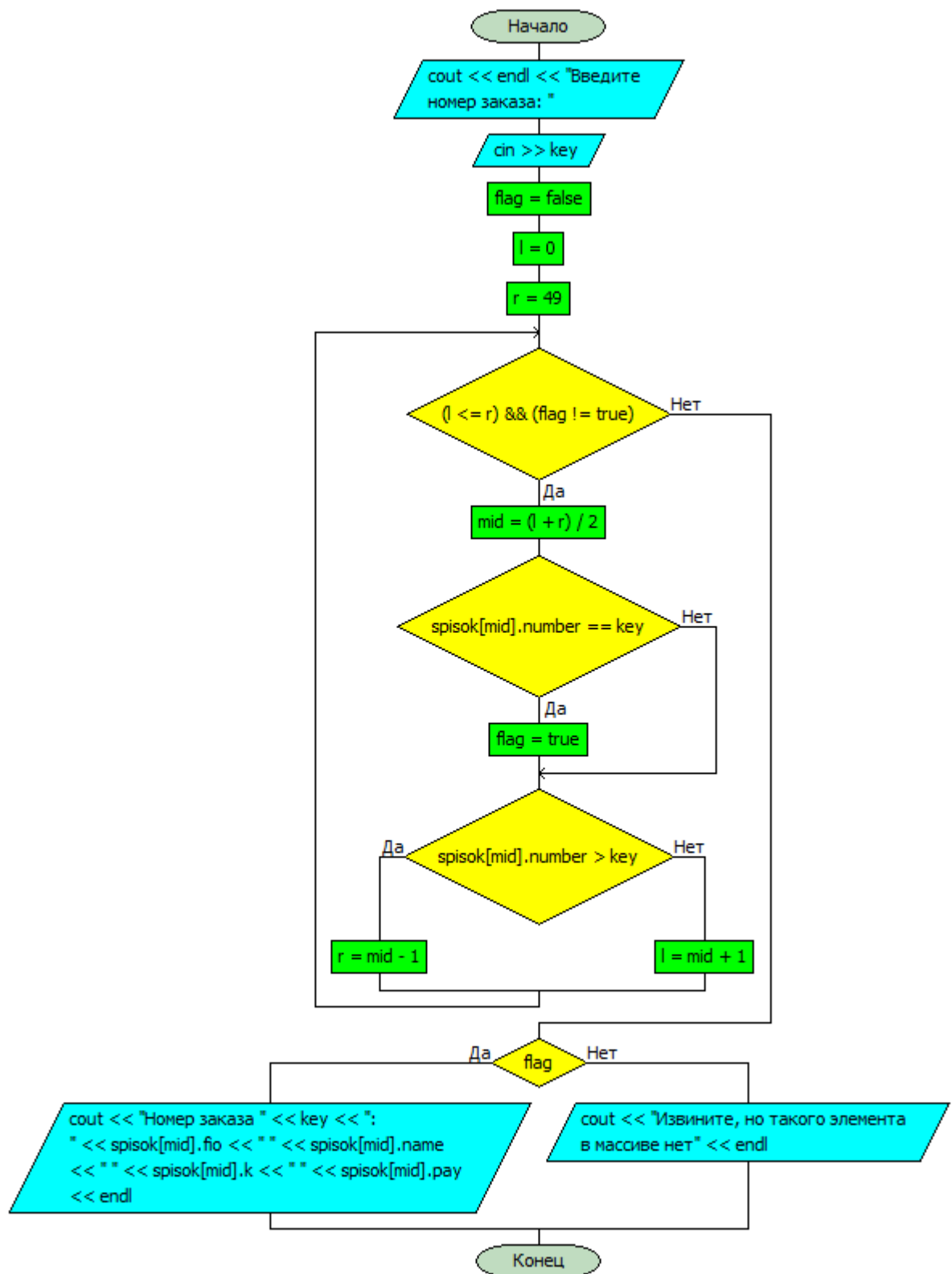


Рисунок 1.4 – Блок-схема алгоритма – функция search

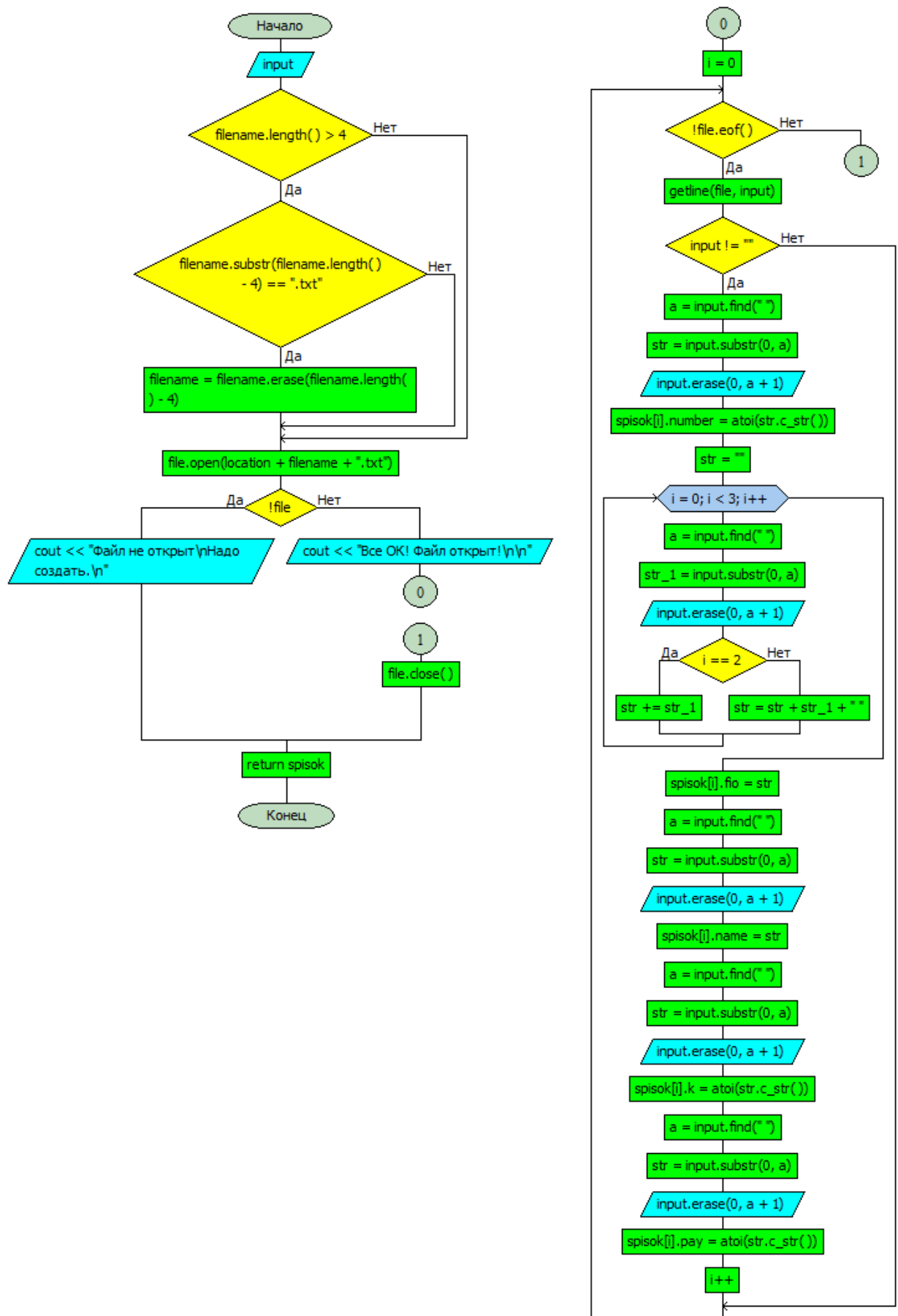


Рисунок 1.5 – Блок-схема алгоритма – функция provfile

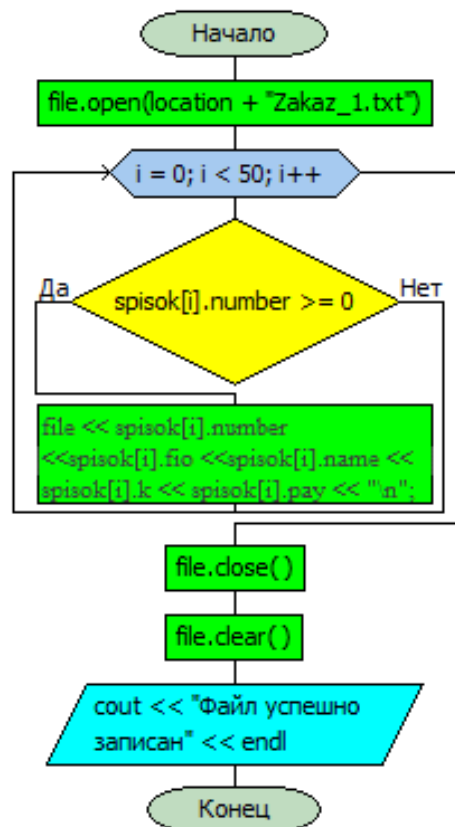


Рисунок 1.6 – Блок-схема алгоритма – функция inPut

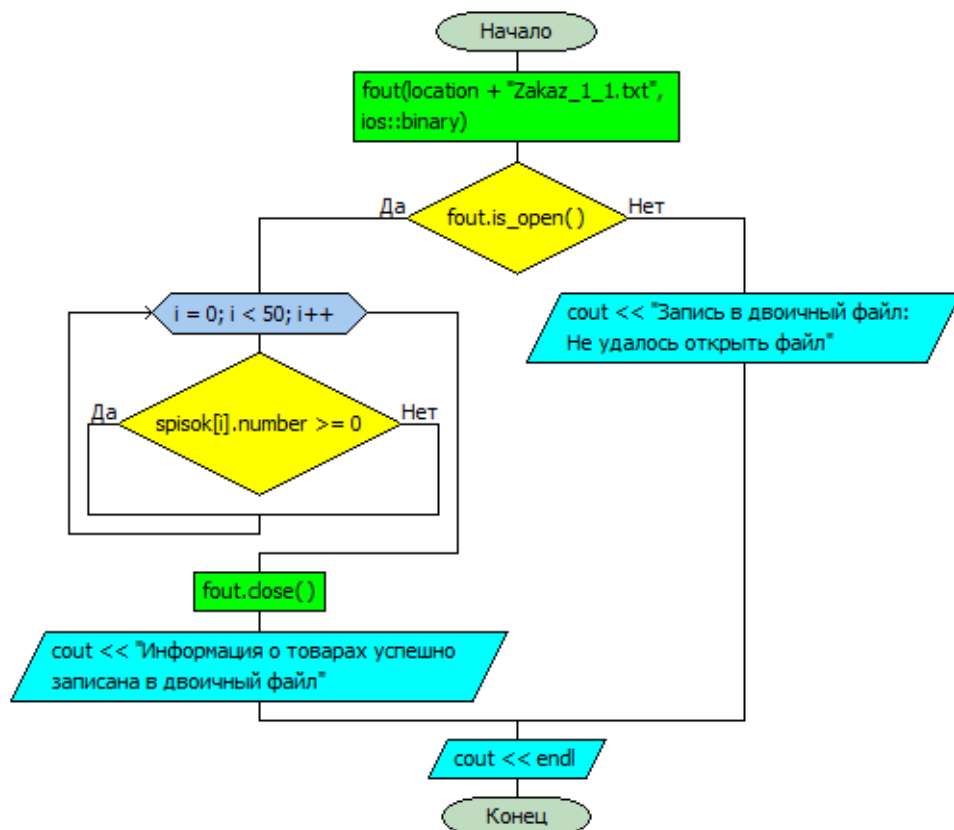


Рисунок 1.7 – Блок-схема алгоритма – функция writeFile

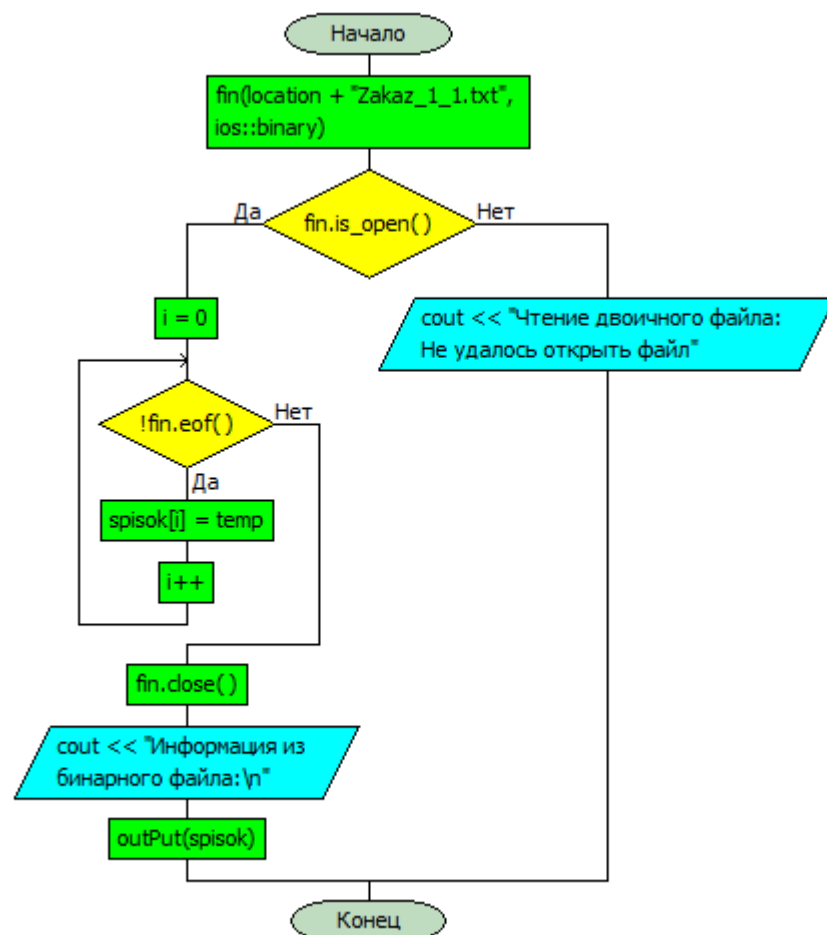


Рисунок 1.8 – Блок-схема алгоритма – функция binaryRead

Исходный код программы представлен в Листингах А.1, А.2, А.3, А.4, А.5, А.6, А.7, А.8, А.9.

Листинг А.1 – Процедура запуска программы

```
#include <iostream>
#include <string>
#include <fstream>
#include <Windows.h>

using namespace std;
string location = "C:\\\\Users\\Dima\\Desktop\\"; //Global Путь!

/*ГЛАВНАЯ ФУНКЦИЯ*/
int main()
{
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);
    setlocale(LC_ALL, "Russian");
    string filename, input;
    cout << "Введите только имя файла" << endl;
    getline(cin, filename);
    Order* spisok = new Order[50];
    provfile(filename, spisok);
    //outPut(spisok);
    sorti(spisok);
    //outPut(spisok);
    search(spisok);
    inPut(spisok);
    cout << "\nДо свидания!" << endl;
    system("pause");
    return (0);
}
```

Листинг А.2 – Процедура создания структуры

```
struct Order {
    int number;
    string fio;
    string name;
    int k;
    int pay;
};
```

Листинг А.3 – Процедура вывода списка объектов

```
void outPut(Order* spisok) {
    for (int i = 0; i < 50; i++) {
        if (spisok[i].number >= 0)
            cout << spisok[i].number << " " << spisok[i].fio << endl;
    }
}
```

Листинг А.4 – Процедура сортировки массива объектов

```
Order* sorti(Order* spisok) {
    for (int i = 0; i < 50 - 1; i++) {
        for (int k = i + 1; k < 50; k++) {
            if (spisok[i].number > spisok[k].number) {
                swap(spisok[i], spisok[k]);
            }
        }
    }
    return spisok;
}
```

Листинг А.5 – Процедура бинарного поиска объекта по номеру заказа

```
void search(Order* spisok) {
    int key;
    cout << endl << "Введите номер заказа: ";
    cin >> key;
    bool flag = false;
    int l = 0; // левая граница
    int r = 49; // правая граница
    int mid;
    while ((l <= r) && (flag != true)) {
        mid = (l + r) / 2;

        if (spisok[mid].number == key)
            flag = true;
        if (spisok[mid].number > key)
            r = mid - 1; // проверяем, какую часть нужно отбросить
        else
            l = mid + 1;
    }

    if (flag)
        cout << "Номер заказа " << key << ": " << spisok[mid].fio << " " <<
        spisok[mid].name << " " << spisok[mid].k << " " << spisok[mid].pay << endl;
    else
        cout << "Извините, но такого элемента в массиве нет" << endl;
}
```

Листинг А.6 – Процедура ввода массива объектов в файл

```
void inPut(Order* spisok) {
    ofstream file;
    file.open(location + "Zakaz_1.txt");
    for (int i = 0; i < 50; i++) {
        if (spisok[i].number >= 0)
            file << spisok[i].number << " " << spisok[i].fio << " " <<
            spisok[i].name << " " << spisok[i].k << " " << spisok[i].pay << "\n";
    }
    file.close();
    file.clear(); // окончили запись файла
    cout << "Файл успешно записан" << endl;
}
```

Листинг А.7 – Процедура взятия данных об объектах из файла

```
Order* provfile(string filename, Order* spisok) //Существует ли файл?
{
    string input;
    if (filename.length() > 4)
    {
        if (filename.substr(filename.length() - 4) == ".txt")
            filename = filename.erase(filename.length() - 4);
    }
    ifstream file;
    file.open(location + filename + ".txt");
    if (!file)
        cout << "Файл не открыт\nНадо создать.\n";
    else
    {
        cout << "Все ОК! Файл открыт!\n\n";
        int i = 0;
        while (!file.eof())
        {
            getline(file, input);
            if (input != "") {
                int a = input.find(" ");
                string str = input.substr(0, a);
                input.erase(0, a + 1);
                //cout << "number: " << str << endl;
                spisok[i].number = atoi(str.c_str());
                str = "";
                for (int i = 0; i < 3; i++) {
                    a = input.find(" ");
                    string str_1 = input.substr(0, a);
                    input.erase(0, a + 1);
                    if (i == 2)
                        str += str_1;
                    else
                        str = str + str_1 + " ";
                }
                //cout << "fio: " << str << endl;
                spisok[i].fio = str;
                a = input.find(" ");
                str = input.substr(0, a);
                input.erase(0, a + 1);
                //cout << "name: " << str << endl;
                spisok[i].name = str;
                a = input.find(" ");
                str = input.substr(0, a);
                input.erase(0, a + 1);
                //cout << "k: " << str << endl;
                spisok[i].k = atoi(str.c_str());
                a = input.find(" ");
                str = input.substr(0, a);
                input.erase(0, a + 1);
                //cout << "pay: " << str << endl;
                spisok[i].pay = atoi(str.c_str());
                i++;
            }

        }
        file.close();
        return spisok;
    }
    return spisok;
}
```

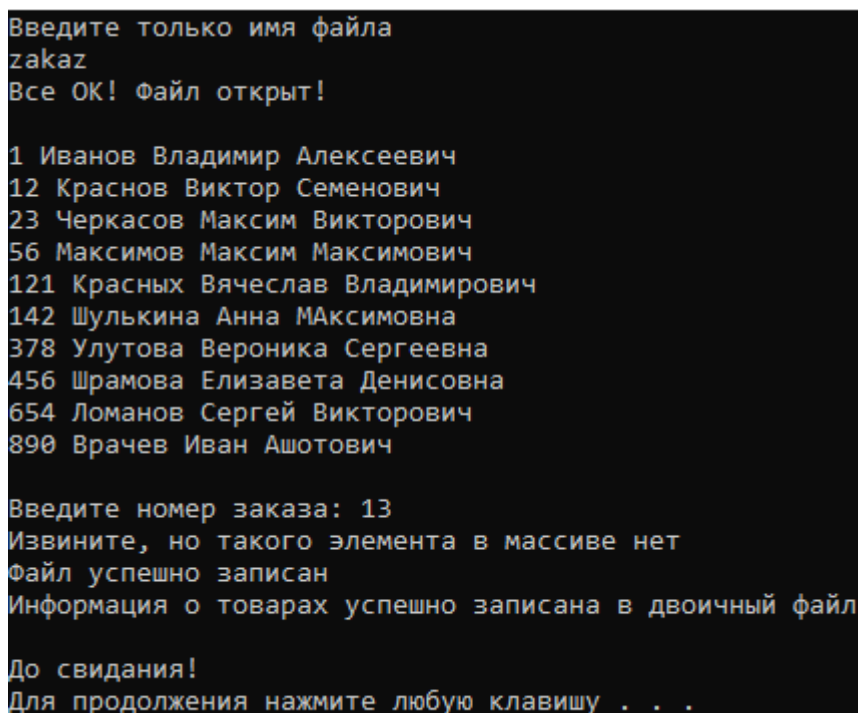
Листинг А.8 – Процедура записи данных в бинарный файл

```
void writeFile(Order* spisok) { //Запись в бинарный файл
    ofstream fout(location + "Zakaz_1_1.txt", ios::binary);
    if (fout.is_open()) {
        for (int i = 0; i < 50; i++) {
            if (spisok[i].number >= 0) {
                fout << spisok[i].number << " " << spisok[i].fio << " " <<
                spisok[i].name << " " << spisok[i].k << " " << spisok[i].pay << endl;
            }
        }
        fout.close();
        cout << "Информация о товарах успешно записана в двоичный файл";
    }
    else cout << "Запись в двоичный файл: Не удалось открыть файл";
    cout << endl;
}
```

Листинг А.9 – Процедура взятия данных из бинарного файла

```
void binaryRead(Order* spisok) { //Чтение бинарного файла
    ifstream fin(location + "Zakaz_1_1.txt", ios::binary);
    if (fin.is_open()) {
        int i = 0;
        while (!fin.eof()) {
            Order temp;
            fin >> temp.number >> temp.fio >> temp.name >> temp.k >> temp.pay;
            spisok[i] = temp;
            i++;
        }
        fin.close();
        cout << "Информация из бинарного файла:\n";
        outPut(spisok);
    }
    else cout << "Чтение двоичного файла: Не удалось открыть файл";
}
```

Пример работы программы представлен на рисунках [2.1](#), [2.2](#), [2.3](#), [2.4](#).




```
Введите только имя файла
zakaz
Все ОК! Файл открыт!

1 Иванов Владимир Алексеевич
12 Краснов Виктор Семенович
23 Черкасов Максим Викторович
56 Максимов Максим Максимович
121 Красных Вячеслав Владимирович
142 Шулькина Анна МАксимовна
378 Улутова Вероника Сергеевна
456 Шрамова Елизавета Денисовна
654 Ломанов Сергей Викторович
890 Врачев Иван Ашотович

Введите номер заказа: 13
Извините, но такого элемента в массиве нет
Файл успешно записан
Информация о товарах успешно записана в двоичный файл

До свидания!
Для продолжения нажмите любую клавишу . . .
```


Рисунок 2.1 – Пример работы программы

 zakaz – Блокнот

Файл Правка Формат Вид Справка

```
012 Краснов Виктор Семенович Холодильник 1 12640
654 Ломанов Сергей Викторович Принтер 5 23457
023 Черкасов Максим Викторович Ноутбук 1 23870
142 Шулькина Анна МАксимовна Микроволновка 2 21570
890 Врачев Иван Ашотович Холодильник 12 52342
456 Шрамова Елизавета Денисовна Утюг 1 3999
121 Красных Вячеслав Владимирович Стол 2 12456
378 Улутова Вероника Сергеевна ПК 1 48670
056 Максимов Максим Максимович Телевизор 1 57670
001 Иванов Владимир Алексеевич Колонки 6 123999
```


Рисунок 2.2 – Изначальный файл

 Zakaz_1 – Блокнот

Файл Правка Формат Вид Справка

```
1 Иванов Владимир Алексеевич Колонки 6 123999
12 Краснов Виктор Семенович Холодильник 1 12640
23 Черкасов Максим Викторович Ноутбук 1 23870
56 Максимов Максим Максимович Телевизор 1 57670
121 Красных Вячеслав Владимирович Стол 2 12456
142 Шулькина Анна МАксимовна Микроволновка 2 21570
378 Улутова Вероника Сергеевна ПК 1 48670
456 Шрамова Елизавета Денисовна Утюг 1 3999
654 Ломанов Сергей Викторович Принтер 5 23457
890 Врачев Иван Ашотович Холодильник 12 52342
```

Рисунок 2.3 – Текстовый файл после работы программы

 Zakaz_1_1 – Блокнот

Файл Правка Формат Вид Справка

```
1 Иванов Владимир Алексеевич Колонки 6 123999
12 Краснов Виктор Семенович Холодильник 1 12640
23 Черкасов Максим Викторович Ноутбук 1 23870
56 Максимов Максим Максимович Телевизор 1 57670
121 Красных Вячеслав Владимирович Стол 2 12456
142 Шулькина Анна МАксимовна Микроволновка 2 21570
378 Улутова Вероника Сергеевна ПК 1 48670
456 Шрамова Елизавета Денисовна Утюг 1 3999
654 Ломанов Сергей Викторович Принтер 5 23457
890 Врачев Иван Ашотович Холодильник 12 52342
|
```

Рисунок 2.4 – Бинарный файл после работы программы

ЗАКЛЮЧЕНИЕ

В результате выполнения данной практической работы были закреплены основные знания о работе с массивами, а также их индексами, при помощи функций. Также закреплены знания о работе с файлами и структурами данных. Были закреплены навыки использования основных библиотек языка программирования C++.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Кубенский А.А. Структуры и алгоритмы обработки данных: объектноориентированный подход и реализация на С++ / А.А. Кубенский.— М.: БХВ-Петербург, 2017.— 300 с.
2. Стивен Прата. Язык программирования С++ (С++11). Лекции и упражнения, 6-е издание — М.: Вильямс, 2012. — 1248 с.
3. Седжвик, Р. Алгоритмы на С++ / Р. Седжвик.— М.:Вильямс, 2017.— 1056 с.