

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «МОСКОВСКИЙ АВИАЦИОННЫЙ
ИНСТИТУТ (НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)»**

Журнал по практике

Студент Гордеев Никита Максимович

Институт №3 “Системы управления, информатика и электроэнергетика”

Кафедра №304 “Вычислительные машины, системы и сети”

Учебная группа М30-107Б

Направление подготовки (специальность) 09.03.01 “Информатика и вычислительная техника”
(шифр)(название направления, специальности)

Вид практики учебная

(учебная, производственная, преддипломная или другой вид практики)

Руководитель практики от МАИ

Чечиков Юрий Борисович / _____ / “ ____ ” _____ 2019 г.

(фамилия, имя, отчество)

(подпись)

(дата)

Студент

Гордеев Никита Максимович / _____ / “ ____ ” _____ 2019 г.

(фамилия, имя, отчество)

(подпись)

(дата)

Москва 2019 г.

1. Место и сроки проведения практики

Наименование предприятия: Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский авиационный институт (национальный исследовательский университет)»

Название структурного подразделения: Кафедра 304

Сроки проведения практики:

-дата начала практики: 09.02.19

-дата окончания практики: 05.07.19

2. Инструктаж по технике безопасности

_____/_____/ “ ____ ” _____ 20 ____ г.
(подпись проводившего) (дата проведения)

3. Индивидуальное задание студенту

Разработать программы на языке С для решения задач с интерфейсом для выбора варианта ввода исходных данных и вывода результатов:

Задача 1 - разработка программы, реализующей численные методы решения задач алгебры, дифференциальных уравнений, системы линейных уравнений.

Задача 2 - разработка программы, выполняющей формирование, преобразование и анализ элементов одномерных массивов и матриц.

Задача 3 - разработка программы, выполняющей обработку текста, поиск, сортировку, переформатирование текста, определение статистических характеристик.

Для каждого задания обеспечить:

- 1) выбор ввода исходных данных из файла или с клавиатуры,
- 2) выбор вывода результатов на экран или в файл.

В отчете для каждого задания необходимо привести:

- 1) постановку задачи.
- 2) описание используемого математического метода,
- 3) входные данные,
- 4) выходные данные,
- 5) словесное описание алгоритма решения задачи,
- 6) структурную схему алгоритма,
- 7) описание интерфейса программы,
- 8) текст программы с комментариями,
- 9) результаты проверки функционирования программы,
- 10) выводы.

Формируемые в ходе выполнения задания компетенции:

№ пп	Шифр	Компетенция
1	ПСК 6	Владеет современными технологиями разработки и тестирования программного обеспечения
2	ПСК 8	Способен и умеет применять методологию системного анализа при решении слабоструктурированных задач

Достигаемые в ходе выполнения задания результаты освоения:

№ пп	Шифр	Результат освоения
1		
2		

4. План выполнения индивидуального задания

План самостоятельной работы студента

Дата	Содержание или наименование проделанной работы	Продолжитель- ность, часы	Компетенция	Подпись руководителя практики
11.02	Разработка структурной схемы алгоритма для решения задачи 1.	5		
18.02	Проверка структурной схемы алгоритма для решения задачи 1 руководителем практики.	3		
20.02	Разработка программы 1 для решения задачи 1. Ввод и вывод данных с клавиатуры в интерактивном режиме.	9		
25.02	Проверка программы 1 руководителем практики.	3		
27.02	Отладка и тестирование.	7		
04.03	Проверка результата работы программы 1 руководителем практики	3		
12.03	Разработка структурной схемы алгоритма для решения задачи 2.	5		
18.03	Проверка структурной схемы алгоритма для решения задачи 2 руководителем практики.	3		
26.03	Разработка программы 2 для решения задачи 2. Ввод и вывод данных с клавиатуры в интерактивном режиме.	9		
01.04	Проверка программы 2 руководителем практики.	3		
09.04	Отладка и тестирование.	7		
15.04	Проверка результата работы программы 2 руководителем практики	3		
23.04	Разработка структурной схемы алгоритма для решения задачи 3.	5		
29.04	Проверка структурной схемы алгоритма для решения задачи 3 руководителем практики.	3		
07.05	Разработка программы для решения задачи 3. Ввод и вывод данных с клавиатуры в интерактивном режиме.	9		
13.05	Проверка программы 3 руководителем практики.	3		
15.05	Отладка и тестирование.	7		
20.05	Проверка результата работы программы 3 руководителем практики	3		
28.05	Оформление предварительного отчета по практике. Первая проверка отчета руководителем практики.	3		
	Итого:	93		

Гордеев Н.М./_____/_____
(подпись студента-практиканта)

“____” _____ 2019г.
(дата)

Чечиков Ю.Б./_____/_____
(подпись руководителя практики от МАИ)

“____” _____ 2019г.
(дата)

План работ практиканта по месту прохождения практики

Дата	Содержание или наименование проделанной работы	Подразделение	Продолжи тельность, часы	Компетенци я	Подпись руководителя практики
1.07	Корректировка отчета по практике. Проверка отчета руководителем практики.	Кафедра 304	3	ПСК 6 ПСК 8	
2.07	Подготовка презентации для выступления на защите. Проверка презентации руководителем практики.	Кафедра 304	4	ПСК 6 ПСК 8	
03.07	Защита отчетов по практике.	Кафедра 304	4	ПСК 6 ПСК 8	
04.07	Защита отчетов по практике.	Кафедра 304	4	ПСК 6 ПСК 8	
			Итого: 15		

Гордеев Н.М./_____ /
(подпись студента-практиканта)

“ ____ ” _____ 2019г.
(дата составления)

Чечиков Ю.Б./_____ /
(подпись руководителя практики от МАИ)

“ ____ ” _____ 2019г.
(дата составления)

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «МОСКОВСКИЙ АВИАЦИОННЫЙ
ИНСТИТУТ (НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)»**

Отчет о прохождении практики

Студента Гордеев Никита Максимович

Институт №3 “Системы управления, информатика и электроэнергетика”

Кафедра №304 “Вычислительные машины, системы и сети”

Учебная группа М30-107Б

Направление подготовки (специальность) 09.03.01 “Информатика и вычислительная техника”
(шифр)(название направления, специальности)

Вид практики учебная
(учебная, производственная, преддипломная или другой вид практики)

Руководитель практики от МАИ

Чечиков Юрий Борисович
(фамилия, имя, отчество) (подпись)

Наименование предприятия: Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский авиационный институт (национальный исследовательский университет)»

Название структурного подразделения (отдел, лаборатория): Кафедра 304

Студент

Гордеев Никита Максимович / _____ / “ ____ ” _____ 2019 г.
(фамилия, имя, отчество) (подпись) (дата)

Москва 2019 г.

Московский авиационный институт
(национальный исследовательский университет)

Институт №3. «Системы управления, информатика и электроэнергетика».

Кафедра №304 «Вычислительные машины, системы и сети»

Отчёт практической работе № 1

Выполнил:

М30-107Б-18

Гордеев Н. М.

Проверил:

Чечиков Ю.Б.

Москва 2019

Содержание

- 1) Задание
- 2) Блок-схема алгоритма
- 3) Описание функций
- 5) Текст работающей программы
- 6) Графики функций
- 7) Вывод

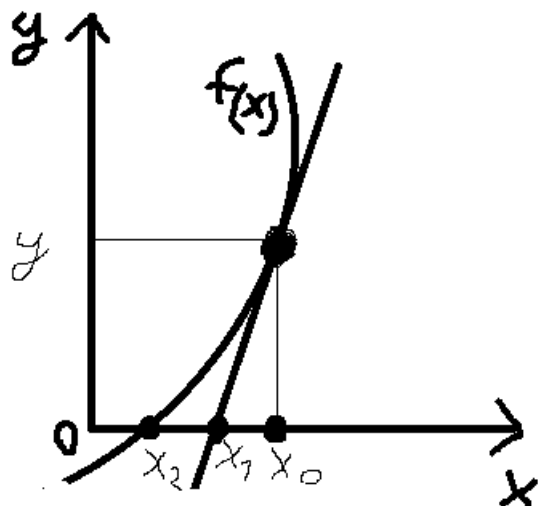
1) Пишу уравнение касательной
 $y = f(x_0) + f'(x_0)(x_1 - x_0)$

2) Из равенства значения f -ии (y) и значения f -ии уравнения касательной ($f(x_0) + f'(x_0)(x_1 - x_0)$) в точке касания следует что если разделить $\frac{y}{f'(x_0)}$ получим

$$\frac{y}{f'(x_0)} = \frac{f(x_0)}{f'(x_0)} + \frac{f'(x_0)(x_1 - x_0)}{f'(x_0)} = \frac{f(x_0)}{f'(x_0)} + (x_1 - x_0)$$

3) Выразив x_1 получим

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} \quad 1.$$

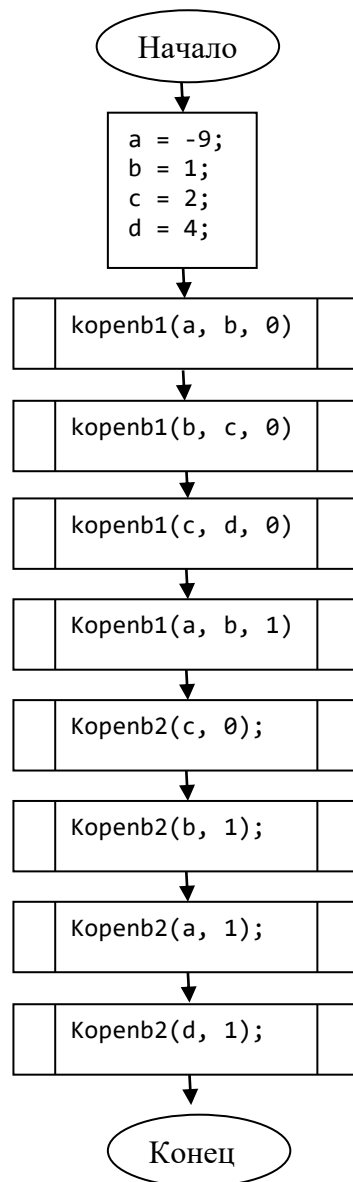


1) Если проанализировать уравнение 1, заметим, что при x_0 достаточно близком к x_2 , x_1 будет ближе к x_2 чем x_0 . Следовательно, подставляя в уравнение 1 вместо x_0 x_1 мы будем приближаться к x_2 (см рисунок.)

Нетрудно заметить, что если взять x_0 немного дальше от корня, то касательная может быть перпендикулярна оси ox или вообще уводить от него, поэтому условие близости x_0 к корню уравнения является обязательным.

Блок-схема

Main



Описание функций:

1) коренb1

a. Назначение: находит корень уравнения на отрезке методом половинного деления

b. Прототип: `double коренb1(double a, double b, bool r);`

c. Обращение: `коренb1(a, b, 0);`

d. Описание параметров:

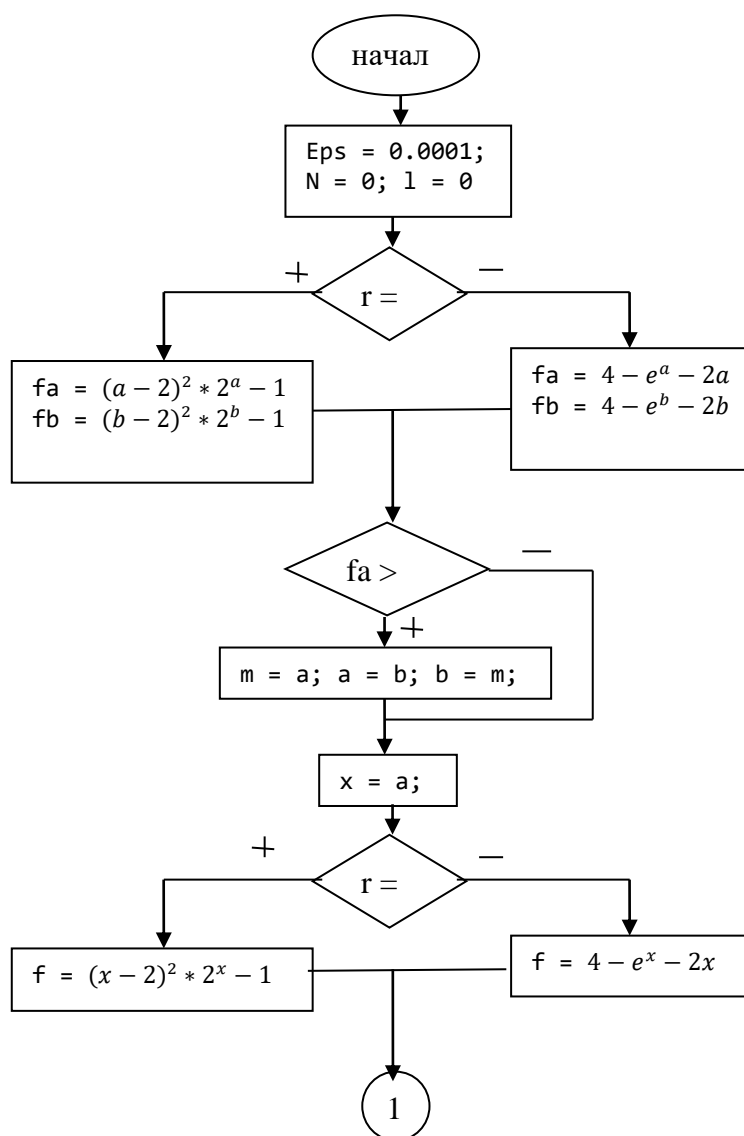
`double a` – начало отрезка,

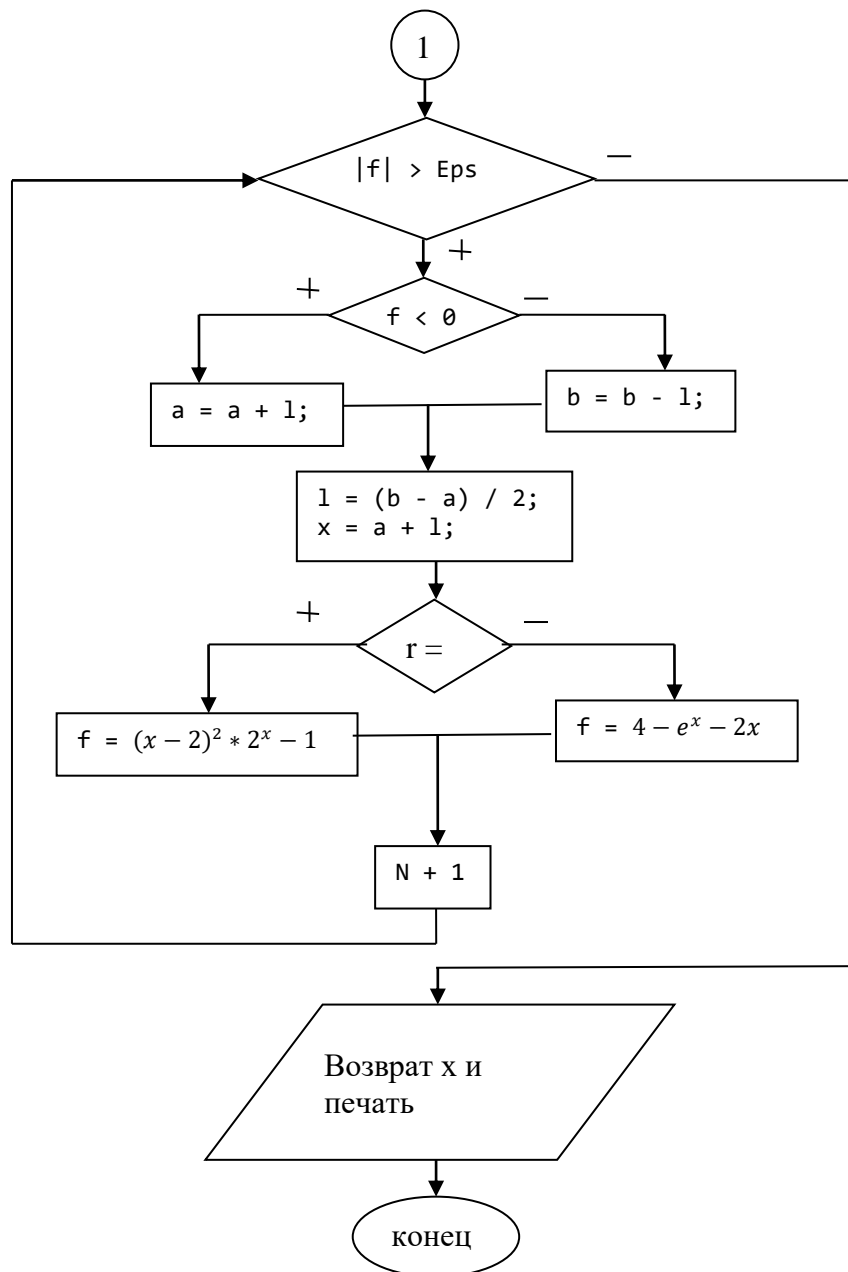
`double b` – конец отрезка,

`bool r` – код уравнения

e. Возвращает `x` – корень уравнения

f. Блок-схема





2) коренb2

a. Назначение: находит корень уравнения на отрезке Ньютона

b. Прототип: `double коренb2(double a, bool r);`

c. Обращение: `коренb2(c, 0);`

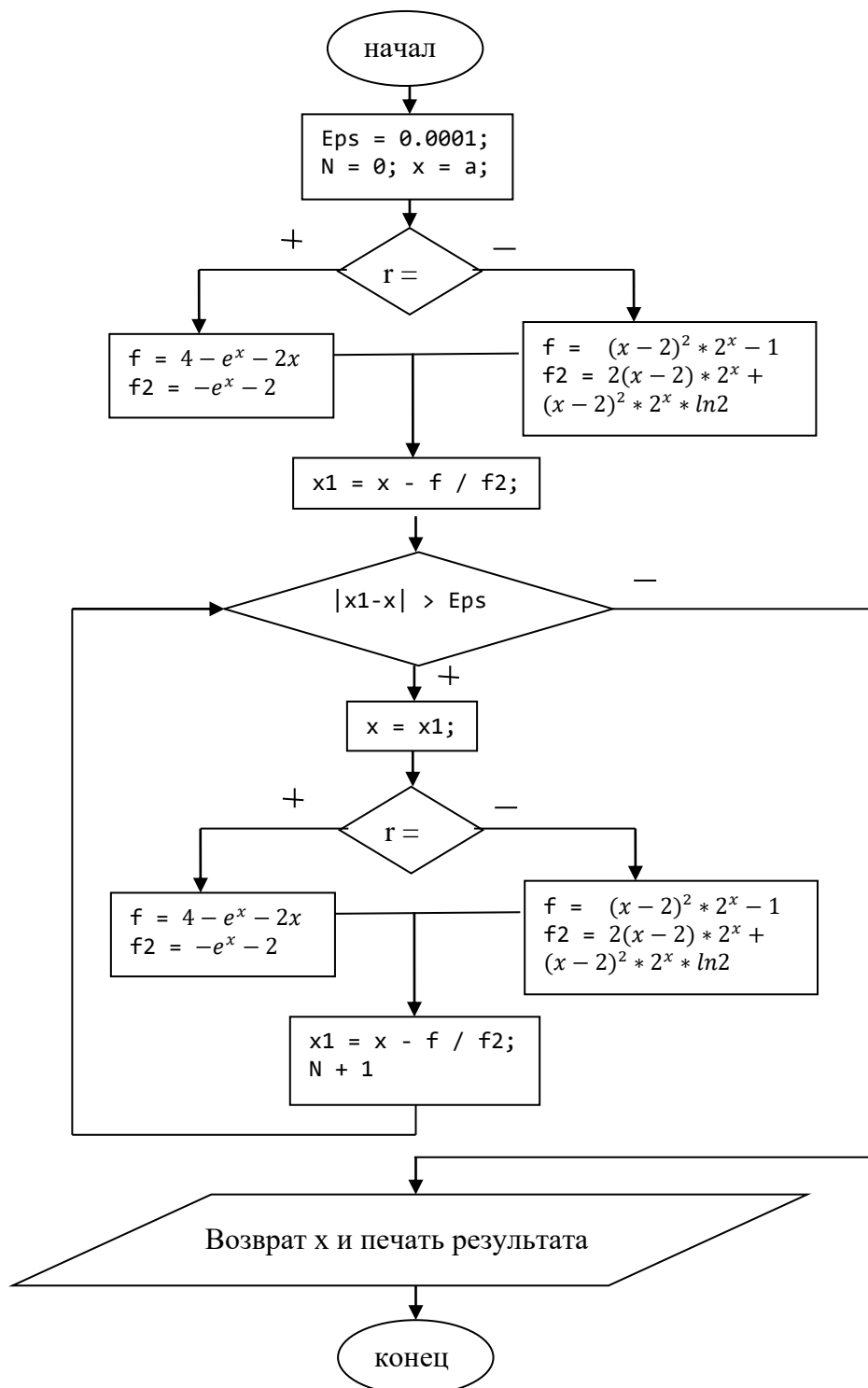
d. Описание параметров:

double a – в близи этого числа ищем корень,

bool r – код уравнения

е. Возвращает x – корень уравнения

ф. Блок-схема $f = 4 - e^x - 2x$



Текст рабочей программы

```

/*****
*          ИНФОРМАТИКА И ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА          *
*****/

*   Probject type: ConsoleApplication                       *
*   Project name:  Laba1                                   *
*   File name:    Laba1.sln                               *
*   Language:     Cpp, MSVS 2017                          *
*   Programmes:   МЗО-107Б-18                             *
*               Гордеев Никита                             *
*   Modified by:  01.03.2019                              *
*   Created:     10.04.2019                               *
*   Comment:     Практика                                 *
*****/

#include "pch.h"
#include <conio.h>
#include <iostream>
#include <iomanip>
#include <ctime>
using namespace std;

double kopenb1(double a, double b, bool r, int &N, int &t);
//double kopenb3(double a, double b, int N);
double kopenb2(double a, bool r, int &N, int &t);
//double kopenb5(double a, int N);
void tabl_3(double x, int N, int t);
void tabl_4(int N, int t);
void tabl_5();

int main() {
    //setlocale(LC_ALL, "Russian");
    int N = 0;                                // Номер итерации
    int t = 0;                                // Номер итерации

    double x = 0;                             // Номер итерации
    // Инициализация переменных
    int a = -9;                                // Граница 1
    int b = 1;                                 // Граница 2
    int c = 2;                                 // Граница 2
}
```

int d = 4;

// Граница 4

//cout << "\n\n\tПервое уравнение по методу половинного деления\n\n";

cout << "\n\n\tThe first half division equation\n\n";

//cout << "Корень на участке (" << a << " ; " << b << ") равен ";

cout << "Root on the plot (" << a << " ; " << b << ") equals ";

x = kopenb1(a, b, 0, N, t);

cout << x << " and received in " << N << " attempts\n";

cout << "f(x0)= " << (x - 2)*(x - 2)*pow(2, x) << "\t";

cout << "T = " << t << "\n\n";

cout << "Root on the plot (" << b << " ; " << c << ") equals ";

x = kopenb1(b, c, 0, N, t);

cout << x << " and received in " << N << " attempts\n";

cout << "f(x0)= " << (x - 2)*(x - 2)*pow(2, x) << "\t";

cout << "T = " << t << "\n\n";

cout << "Root on the plot (" << c << " ; " << d << ") equals ";

x = kopenb1(c, d, 0, N, t);

cout << x << " and received in " << N << " attempts\n";

cout << "f(x0)= " << (x - 2)*(x - 2)*pow(2, x) << "\t";

cout << "T = " << t << "\n\n";

//cout << "\n\n\tВторое уравнение по методу половинного деления\n\n";

cout << "\n\n\tThe second half division equation\n\n";

cout << "Root on the plot (" << a << " ; " << b << ") equals ";

x = kopenb1(a, b, 1, N, t);

//cout << x << " и получен за " << N << " попыток\n";

cout << x << " and received in " << N << " attempts\n";

cout << "f(x0)= " << 4 - pow(2.718281828459045, x) - 2 * x << "\t";

cout << "T = " << t << "\n\n";

//cout << "\n\n\tПервое уравнение по методу Ньютона \n\n";

cout << "\n\n\tNewton's first equation \n\n";

x = kopenb2(a, 1, N, t);

//cout << "Корень вблизи значения (" << a << ") равен " << x << " и получен за " << N << " попыток\n";

cout << "Root near values (" << a << ") equals " << x << " and received in " << N << " attempts\n";

cout << "f(x0)= " << (x - 2)*(x - 2)*pow(2, x) << "\t";

cout << "T = " << t << "\n\n";

```

x = kopenb2(b, 1, N, t);
cout << "Root near values (" << b << ") equals " << x << " and received in " << N << " attempts\n";
cout << "f(x0)= " << (x - 2)*(x - 2)*pow(2, x) << "\t";
cout << "T = " << t << "\n\n";

x = kopenb2(d, 1, N, t);
cout << "Root near values (" << d << ") equals " << x << " and received in " << N << " attempts\n";
cout << "f(x0)= " << (x - 2)*(x - 2)*pow(2, x) << "\t";
cout << "T = " << t << "\n\n";

//cout << "\n\n\tВторое уравнение по методу Ньютона \n\n";
cout << "\n\n\tSecond Newton's equation \n\n";
x = kopenb2(c, 0, N, t);
cout << "Root near values (" << c << ") equals " << x << " and received in " << N << " attempts\n";
cout << "f(x0)= " << 4 - pow(2.718281828459045, x) - 2 * x << "\t";
cout << "T = " << t << "\n\n\n\n";

cout << "\t\t Half division method" << "\t\t Newton's method\n";
cout << char(218);
cout << setfill(char(196)) << setw(15);
cout << char(194);
cout << setfill(char(196)) << setw(15);
cout << char(194);
cout << setfill(char(196)) << setw(15);
cout << char(194);
cout << setfill(char(196)) << setw(15);
cout << char(194);
cout << setfill(char(196)) << setw(15);
cout << char(191) << endl;
cout << char(179) << "    x    ";
cout << char(179) << "    n    ";
cout << char(179) << "    t    ";
cout << char(179) << "    n    ";
cout << char(179) << "    t    ";
cout << char(179) << endl;
cout << char(195);
cout << setfill(char(196)) << setw(15);
cout << char(197);
cout << setfill(char(196)) << setw(15);
cout << char(197);

```

```

cout << setfill(char(196)) << setw(15);
cout << char(197);
cout << setfill(char(196)) << setw(15);
cout << char(197);
cout << setfill(char(196)) << setw(15);
cout << char(180) << endl;

```

```

tabl_3(kopenb1(a, b, 0, N, t), N, t);
kopenb2(a, 1, N, t);
tabl_4(N, t);
tabl_5();

```

```

tabl_3(kopenb1(b, c, 0, N, t), N, t);
kopenb2(b, 1, N, t);
tabl_4(N, t);
tabl_5();

```

```

tabl_3(kopenb1(c, d, 0, N, t), N, t);
kopenb2(d, 1, N, t);
tabl_4(N, t);
tabl_5();

```

```

tabl_3(kopenb1(a, b, 1, N, t), N, t);
kopenb2(c, 0, N, t);
tabl_4(N, t);
cout << char(192);
cout << setfill(char(196)) << setw(15);
cout << char(193);
cout << setfill(char(196)) << setw(15);
cout << char(193);
cout << setfill(char(196)) << setw(15);
cout << char(193);
cout << setfill(char(196)) << setw(15);
cout << char(193);
cout << setfill(char(196)) << setw(15);
cout << char(217) << endl;

```

```

_getch();
return 0;

```

```

}

```



```

void tabl_3(double x, int N, int t) {
    cout << char(179) << " " << setfill(char(32)) << setw(12) << x << " ";
    cout << char(179) << " " << setfill(char(32)) << setw(12) << N << " ";
    cout << char(179) << " " << setfill(char(32)) << setw(12) << t << " ";
    cout << char(179);
}

void tabl_4(int N, int t) {
    cout << " " << setfill(char(32)) << setw(12) << N << " ";
    cout << char(179) << " " << setfill(char(32)) << setw(12) << t << " ";
    cout << char(179) << endl;
}

void tabl_5() {
    cout << char(195);
    cout << setfill(char(196)) << setw(15);
    cout << char(197);
    cout << setfill(char(196)) << setw(15);
    cout << char(197);
    cout << setfill(char(196)) << setw(15);
    cout << char(197);
    cout << setfill(char(196)) << setw(15);
    cout << char(197);
    cout << setfill(char(196)) << setw(15);
    cout << char(180) << endl;
}

/*****
                                ШАБКА ПЕРВОГО СПОСОБА
*****/

void habkatabl() {
    cout << char(218);
    cout << setfill(char(196)) << setw(15);
    cout << char(194);
    cout << setfill(char(196)) << setw(15);
    cout << char(194);
    cout << setfill(char(196)) << setw(15);
    cout << char(191) << endl;
    cout << char(179) << "   n   ";
    cout << char(179) << "   x   ";
    cout << char(179) << "   f   ";
}

```

```

    cout << char(179) << endl;
    cout << char(195);
    cout << setfill(char(196)) << setw(15);
    cout << char(197);
    cout << setfill(char(196)) << setw(15);
    cout << char(197);
    cout << setfill(char(196)) << setw(15);
    cout << char(180) << endl;
}
/*****

                ШАБКА ВТОРОГО СПОСОБА

*****/

void habkatabl_2() {
    cout << char(218);
    cout << setfill(char(196)) << setw(15);
    cout << char(194);
    cout << setfill(char(196)) << setw(15);
    cout << char(194);
    cout << setfill(char(196)) << setw(15);
    cout << char(194);
    cout << setfill(char(196)) << setw(15);
    cout << char(194);
    cout << setfill(char(196)) << setw(15);

    cout << char(191) << endl;
    cout << char(179) << "   n   ";
    cout << char(179) << "   x   ";
    cout << char(179) << "  f(x)  ";
    cout << char(179) << "  f'(x) ";
    cout << char(179) << " 1-f(x)/f'(x) ";
    cout << char(179) << endl;

    cout << char(195);
    cout << setfill(char(196)) << setw(15);
    cout << char(197);
    cout << setfill(char(196)) << setw(15);
    cout << char(197);
    cout << setfill(char(196)) << setw(15);
    cout << char(197);
    cout << setfill(char(196)) << setw(15);

```

```

    cout << char(197);
    cout << setfill(char(196)) << setw(15);
    cout << char(180) << endl;
}

/*****

ПЕЧАТЬ СЕРЕДИНЫ ТАБЛИЦЫ ПЕРВЫМ СПОСОБОМ

*****/

void tabl(double x, double f, int N) {
    cout << char(179) << " " << setfill(char(32)) << setw(12) << N << " ";
    cout << char(179) << " " << setfill(char(32)) << setw(12) << x << " ";
    cout << char(179) << " " << setfill(char(32)) << setw(12) << f << " ";
    cout << char(179) << endl;

    cout << char(195);
    cout << setfill(char(196)) << setw(15);
    cout << char(197);
    cout << setfill(char(196)) << setw(15);
    cout << char(197);
    cout << setfill(char(196)) << setw(15);
    cout << char(180) << endl;
}

/*****

ПЕЧАТЬ СЕРЕДИНЫ ТАБЛИЦЫ ВТОРЫМ СПОСОБОМ

*****/

void tabl_2(int N, double x, double f, double fx, double fy) {
    cout << char(179) << " " << setfill(char(32)) << setw(12) << N << " ";
    cout << char(179) << " " << setfill(char(32)) << setw(12) << x << " ";
    cout << char(179) << " " << setfill(char(32)) << setw(12) << f << " ";
    cout << char(179) << " " << setfill(char(32)) << setw(12) << fx << " ";
    cout << char(179) << " " << setfill(char(32)) << setw(12) << fy << " ";
    cout << char(179) << endl;

    cout << char(195);
    cout << setfill(char(196)) << setw(15);
    cout << char(197);
    cout << setfill(char(196)) << setw(15);
    cout << char(197);
    cout << setfill(char(196)) << setw(15);
    cout << char(197);
    cout << setfill(char(196)) << setw(15);

```

```

    cout << char(197);
    cout << setfill(char(196)) << setw(15);
    cout << char(180) << endl;
}

/*****

                                КОНЕЦ ТАБЛИЦЫ ПЕРВЫМ СПОСОБОМ

*****/

void otherktabl(double x, int &N) {
    cout << char(179) << " " << setfill(char(32)) << setw(13) << "Answer:  ";
    cout << char(179) << " N = " << setfill(char(32)) << setw(9) << N;
    cout << char(179) << " X = " << setfill(char(32)) << setw(9) << x;
    cout << char(179) << endl;

    cout << char(192);
    cout << setfill(char(196)) << setw(15);
    cout << char(193);
    cout << setfill(char(196)) << setw(15);
    cout << char(193);
    cout << setfill(char(196)) << setw(15);
    cout << char(217) << endl;
}

/*****

                                КОНЕЦ ТАБЛИЦЫ ВТОРЫМ СПОСОБОМ

*****/

void otherktabl_2(double x, int &N) {
    cout << char(179) << " " << setfill(char(32)) << setw(13) << "Answer:  ";
    cout << char(179) << "    N = " << setfill(char(32)) << setw(5);
    cout << char(179) << setfill(char(32)) << setw(7) << N << "    ";
    cout << char(179) << "    X = " << setfill(char(32)) << setw(5);
    cout << char(179) << setfill(char(32)) << setw(13) << x << " ";
    cout << char(179) << endl;

    cout << char(192);
    cout << setfill(char(196)) << setw(15);
    cout << char(193);
    cout << setfill(char(196)) << setw(15);
    cout << char(193);
    cout << setfill(char(196)) << setw(15);
    cout << char(193);
    cout << setfill(char(196)) << setw(15);

```

```

cout << char(193);
cout << setfill(char(196)) << setw(15);
cout << char(217) << endl;
}

/*****

                ПЕРВЫЙ СПОСОБ

*****/

double kopenb1(double a, double b, bool r, int &N, int &t) {
    int start_time = clock(); // начальное время
    cout << "\n" << start_time << "\n";
    double Eps = 0.0001;      // Точность
    double f;                  // f(x)
    double fa;                 // f(a)
    double fb;                 // f(b)
    double l = 0;              // длина
    N = 0;                     // Номер итерации
    double m;                  // Вспомогательная переменная
    //cout << "Корень на участке (" << a << " ; " << b << ") равен ";
    if (r == 0)
    {
        fa = (a - 2)*(a - 2)*pow(2, a) - 1;
        fb = (b - 2)*(b - 2)*pow(2, b) - 1;
    }
    else
    {
        fa = 4 - pow(2.718281828459045, a) - 2 * a;
        fb = 4 - pow(2.718281828459045, b) - 2 * b;
    }
    //если вункция на участке убывает меняем местами концы отрезка
    if (fa > fb) { m = a; a = b; b = m; }

    double x = a;              // Переменная
    if (r == 0)
        f = (x - 2)*(x - 2)*pow(2, x) - 1;
    else
        f = 4 - pow(2.718281828459045, x) - 2 * x;

    //habkatabl();
    while (abs(f) > Eps)
    {

```

```

        if (f < 0) a = a + l;
        else b = b - l;
        l = (b - a) / 2;
        x = a + l;
        if (r == 0)
            f = (x - 2)*(x - 2)*pow(2, x) - 1;
        else

            f = 4 - pow(2.718281828459045, x) - 2 * x;

        N++;
        //tabl(x, f, N);
    }
    //cout << x << " и получен за " << N << " попыток\n";
    ///otherktabl(x, N);
    //cout << "f(x0)= " << f<<"\n";
    //cout << "\n\n\n";
    int end_time = clock(); // конечное время
    cout << "\n" << end_time << "\n";
    t = end_time - start_time;
    return x;
}

/*****

```

ВТОРОЙ СПОСОБ

```

*****/

double kopenb2(double a, bool r, int &N, int &t) {
    int start_time = clock(); // начальное время
    double Eps = 0.0001;      // Точность
    double f;                  // f(x)
    double x1;                 // f
    double f2;                 // f'(x)
    N = 0;                     // Номер итерации
    double x = a;              // Переменная
    if (r == 0)
    {
        f = 4 - pow(2.718281828459045, x) - 2 * x;
        f2 = -pow(2.718281828459045, x) - 2;
    }
    else
    {
        f = (x - 2)*(x - 2)*pow(2, x) - 1;
    }
}

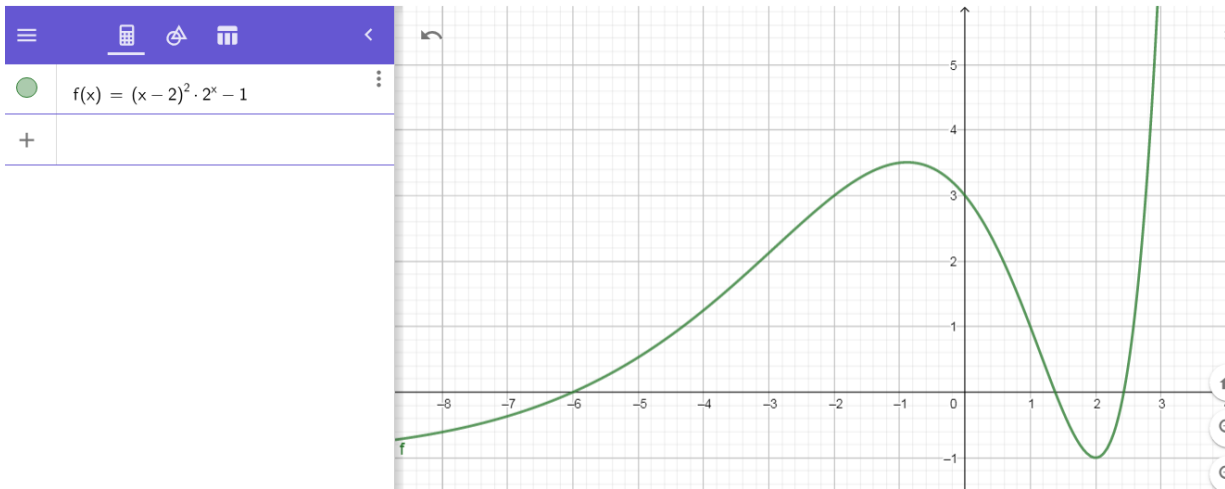
```

```

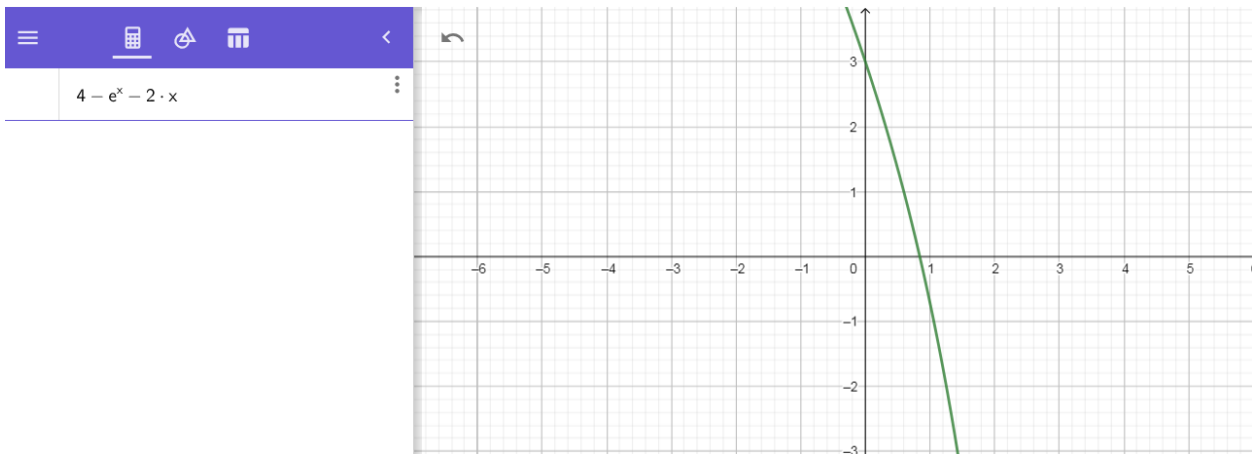
        f2 = 2 * (x - 2)*pow(2, x) + pow(2, x)*log(2)*(x - 2)*(x - 2);
    }
    x1 = x - f / f2;
    //habkatabl_2();
    while (abs(x1-x) > Eps)
    {
        x = x1;
        if (r == 0)
        {
            f = 4 - pow(2.718281828459045, x) - 2 * x;
            f2 = -pow(2.718281828459045, x) - 2;
        }
        else
        {
            f = (x - 2)*(x - 2)*pow(2, x) - 1;
            f2 = 2 * (x - 2)*pow(2, x) + pow(2, x)*log(2)*(x - 2)*(x - 2);
        }
        x1 = x - f / f2;
        N++;
        //tabl_2(N, x, f, f2, x1);
    }
    //otherktabl_2(x, N);
    /*cout << "Корень в близи значения (" << a << ") равен " << x << " и получен за " << N << " попыток\n";
    cout << "f(x0)= " << f << "\n";*/
    //cout << "\n\n\n";
    int end_time = clock(); // конечное время
    t = end_time - start_time;
    return x;
}

```

$$f(x) = (x - 2)^2 \cdot 2^x - 1$$



$$f(x) = 4 - e^x - 2x$$



The first half division equation

Root on the plot (-9 ; 1) equals -6.00012 and received in 14 attempts
 $f(x_0) = 0.999946$ $T = 0$

Root on the plot (1 ; 2) equals 1.38019 and received in 14 attempts
 $f(x_0) = 0.999995$ $T = 0$

Root on the plot (2 ; 4) equals 2.43066 and received in 11 attempts
 $f(x_0) = 0.999954$ $T = 0$

The second half division equation

Root on the plot (-9 ; 1) equals 0.840851 and received in 16 attempts
 $f(x_0) = -4.03103e-05$ $T = 0$

Newton's first equation

Root near values (-9) equals -6 and received in 5 attempts
 $f(x_0) = 1$ $T = 0$

Root near values (1) equals 1.38018 and received in 2 attempts
 $f(x_0) = 1$ $T = 0$

Root near values (4) equals 2.43071 and received in 6 attempts
 $f(x_0) = 1.00018$ $T = 0$

Second Newton's equation

Root near values (2) equals 0.840842 and received in 4 attempts
 $f(x_0) = -1.85493e-07$ $T = 0$

Half division method			Newton's method	
x	n	t	n	t
-6.00012	14	0	5	0
1.38019	14	0	2	0
2.43066	11	0	6	0
0.840851	16	0	4	0

Московский авиационный институт
(национальный исследовательский университет)

Институт №3. «Системы управления, информатика и электроэнергетика».

Кафедра №304 «Вычислительные машины, системы и сети»

Отчёт по практической работе

Выполняли:

М30-107Б-18

Гордеев Н. М.

Проверил:

Чечиков Ю.Б.

Москва 2019

Содержание

1. Задание.....	
2. Блок-схема алгоритма.....	
3. Псевдокод.....	
3. Текст программы.....	
4. Корректные тесты.....	
5. Некорректные тесты.....	
6. Вывод.....	

Кафедра 302
Практика

Курс: ИНФОРМАТИКА II семестр

ВАРИАНТ 7

В процессе функционирования АСУ ВД в файле фиксируются данные о самолетах, совершивших посадку. Каждая запись имеет структуру типа:

376	Б-3726	5.6	44
номер	бортовой	вес	количество
рейса	номер	груза	контейнеров

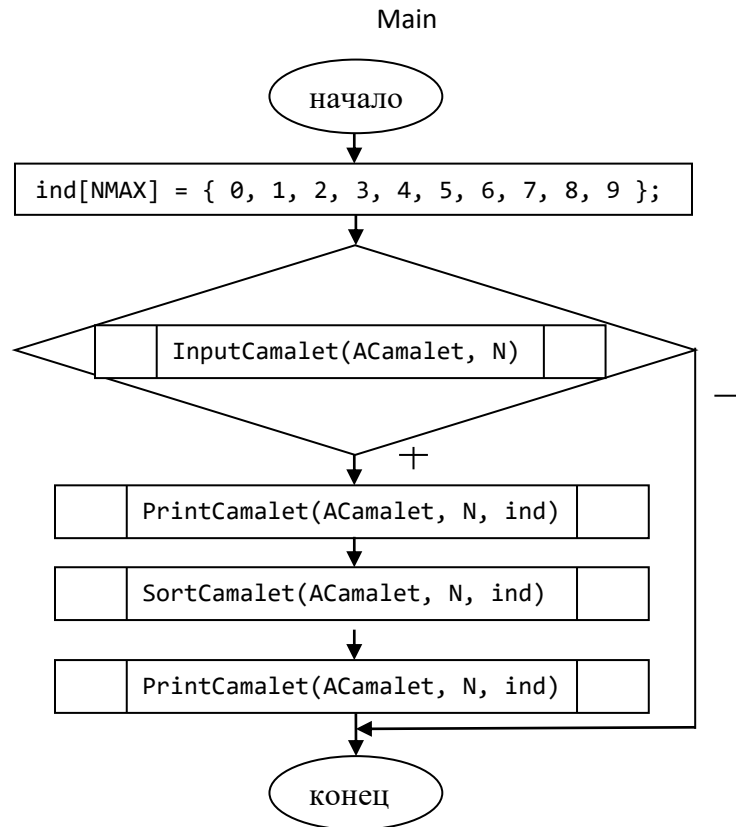
1) подготовить программу, сортирующую записи с использованием индексной сортировки методом выбора в порядке возрастания номеров рейсов; определить суммарное количество контейнеров; результаты печатать в виде таблицы;

2) обеспечить входной контроль номера рейса, бортового номера, веса груза и количества контейнеров, выполнить отладку и тестирование.

Чтение данных их файла производить с использованием функций ввода/вывода языка C++.

Алгоритм должен быть параметризован; обмен данными с подпрограммой должен осуществляться только через параметры; исходные данные хранятся в отдельном файле.

Блок-схема



Описание функций:

1) SharKaTabl

a. Назначение: печатает шапку для таблицы

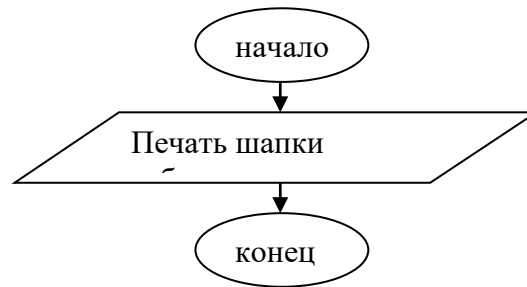
b. Прототип: `void SharKaTabl();`

c. Обращение: `SharKaTabl();`

d. Параметры отсутствуют

e. Ничего не возвращает

f. Блок-схема



2) InputPlane

a. Назначение: чтение данных из файла и нумерация ошибок

b. Прототип: `bool InputPlane(Plane * pPlane, int &N, clyh* clyh)`

c. Обращение: `if (InputPlane(APlane, N, clyh))`

d. Описание параметров:

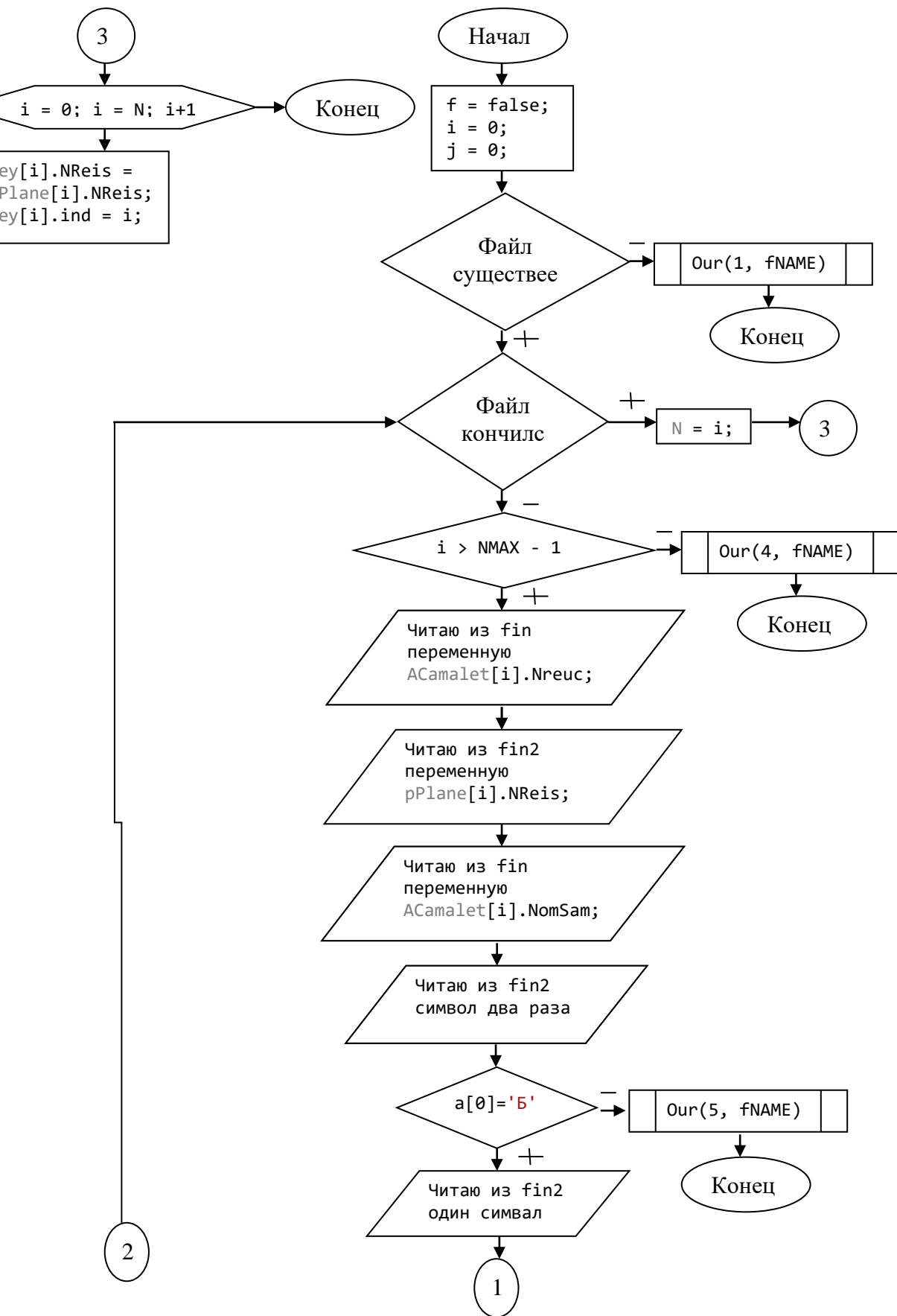
`Plane * pPlane` – структура (массив) самолётов,

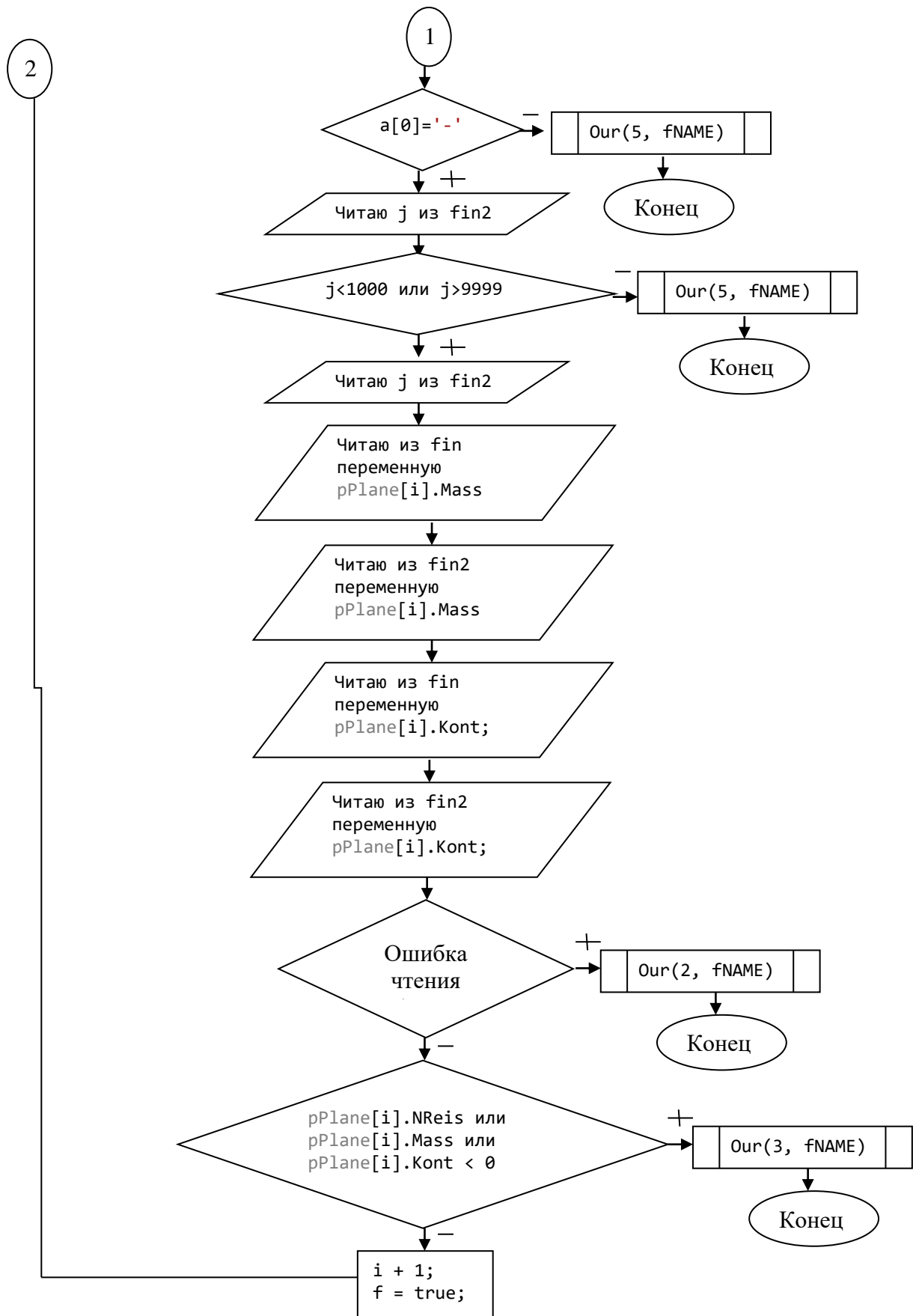
`clyh* clyh` – структура для сортировки

`int &N` – количество элементов структуры (массива) самолётов

e. Возвращает `f` – флаг штатного чтения

f. Блок-схема





a. Назначение: суммирует количество контейнеров

b. Прототип: `int SumPlane(Plane *pPlane, int N);`

c. Обращение: `cout << char(179) << " " << setfill(char(32)) << setw(27)`

`<< SumPlane(pPlane, N) << " " << char(179) << "\n";`

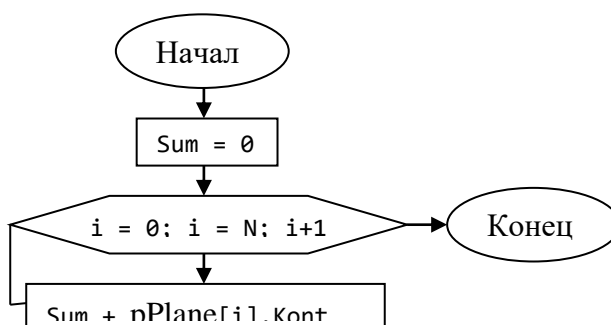
d. Описание параметров:

`Plane * pPlane` – структура (массив) самолётов,

`int N` – количество элементов структуры (массива) самолётов,

e. Возвращает `Sum` – сумму перевозимых контейнеров

f. Блок-схема



4) OtherkTab1

a. Назначение: печать конца таблицы с данными о самолётах и суммы контейнеров

b. Прототип: `void OtherkTab1(Plane *pPlane, int N);`

c. Обращение: `otherkTab1 (pPlane, N);`

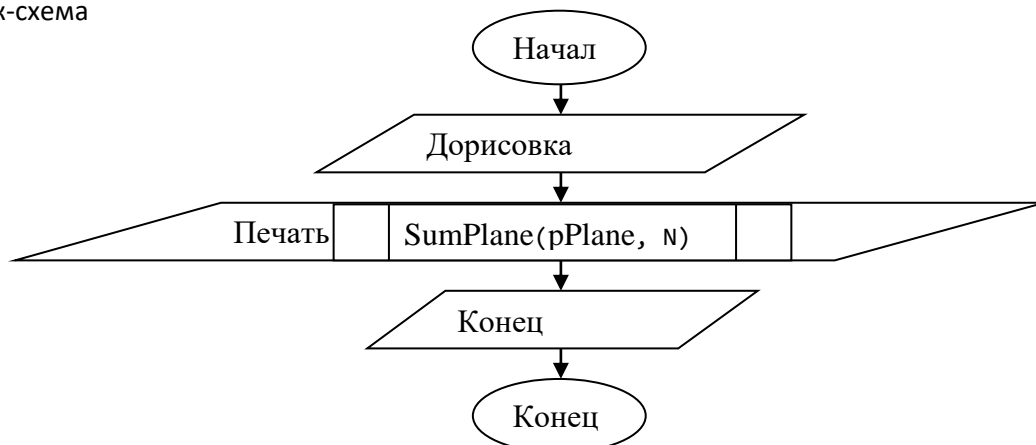
d. Описание параметров:

`Plane * pPlane` – структура (массив) самолётов,

`int N` – количество элементов структуры (массива) самолётов,

e. Ничего не возвращает

f. Блок-схема



5) PrintPlane

a. Назначение: печать таблицы с данными о самолётах

b. Прототип: `void PrintPlane (Plane * pPlane, int N, clyh* clyh);`

c. Обращение: `PrintPlane (APlane, N, clyh);`

d. Описание параметров:

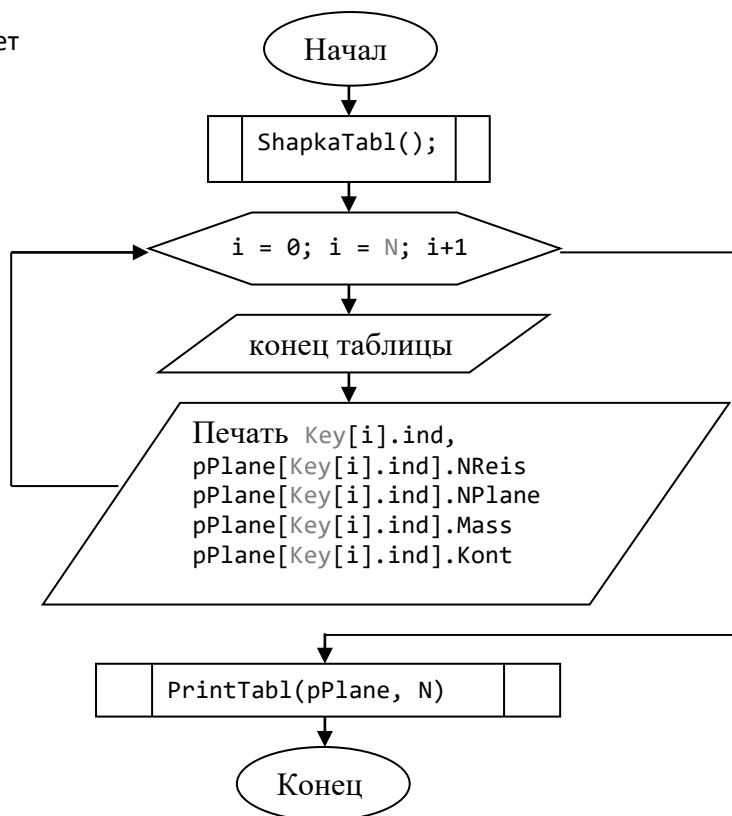
`Plane * pPlane` – структура (массив) самолётов,

`int N` – количество элементов структуры (массива) самолётов,

`clyh* clyh` – структура для сортировки

e. Ничего не возвращает

f. Блок-схема



6) SortPlane

a. Назначение: сортировка структуры с данными о самолётах

b. Прототип: void SortPlane(Plane * pPlane, int N, , clyh* clyh);

c. Обращение: SortPlane(N, , clyh);

d. Описание параметров:

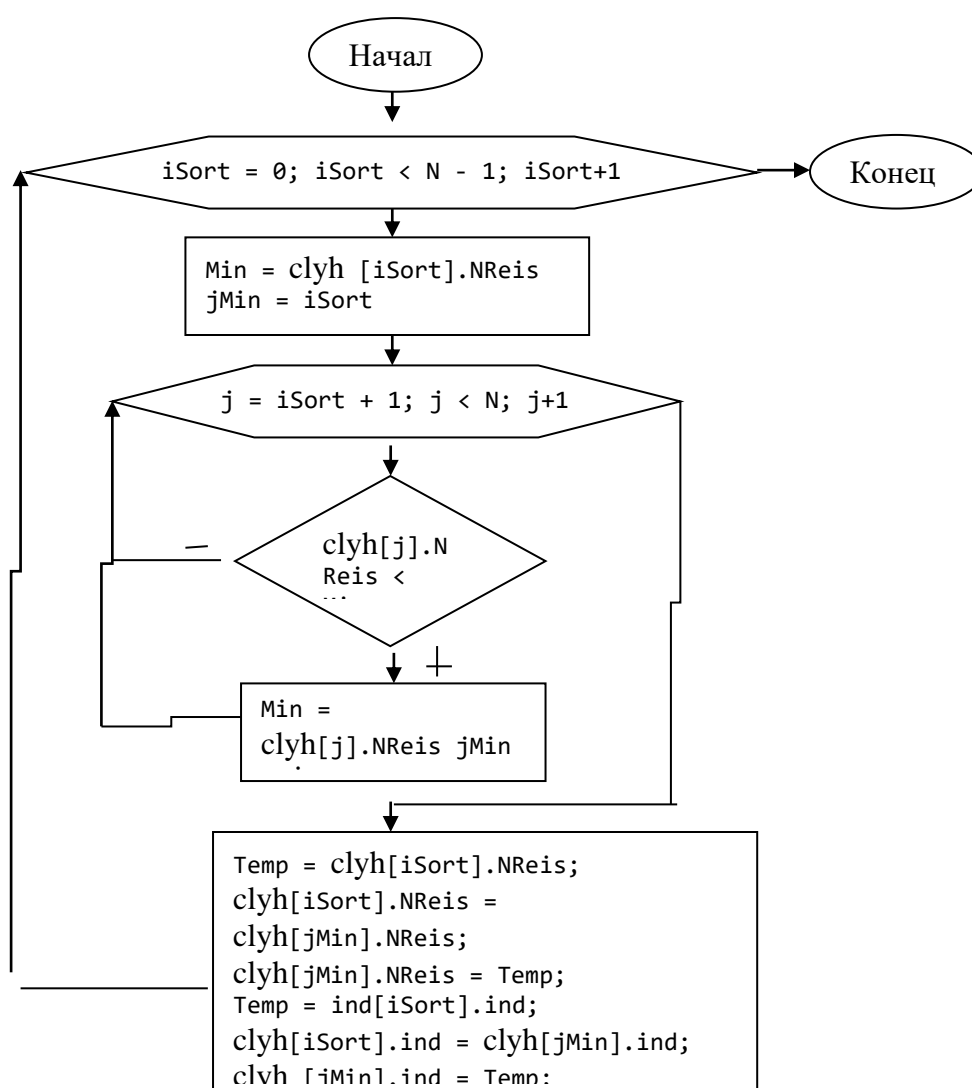
,clyh* clyh – структура номеров самолётов и индексов,

int N – количество элементов структуры (массива) самолётов,

int *ind – упорядоченный индексный массив

e. Ничего не возвращает

f. Блок-схема



7) Our

a. Назначение: сообщения об ошибках

b. Прототип: void void Our(int N_oh, const char FNAME[]);

c. Обращение: Our(1, FNAME);

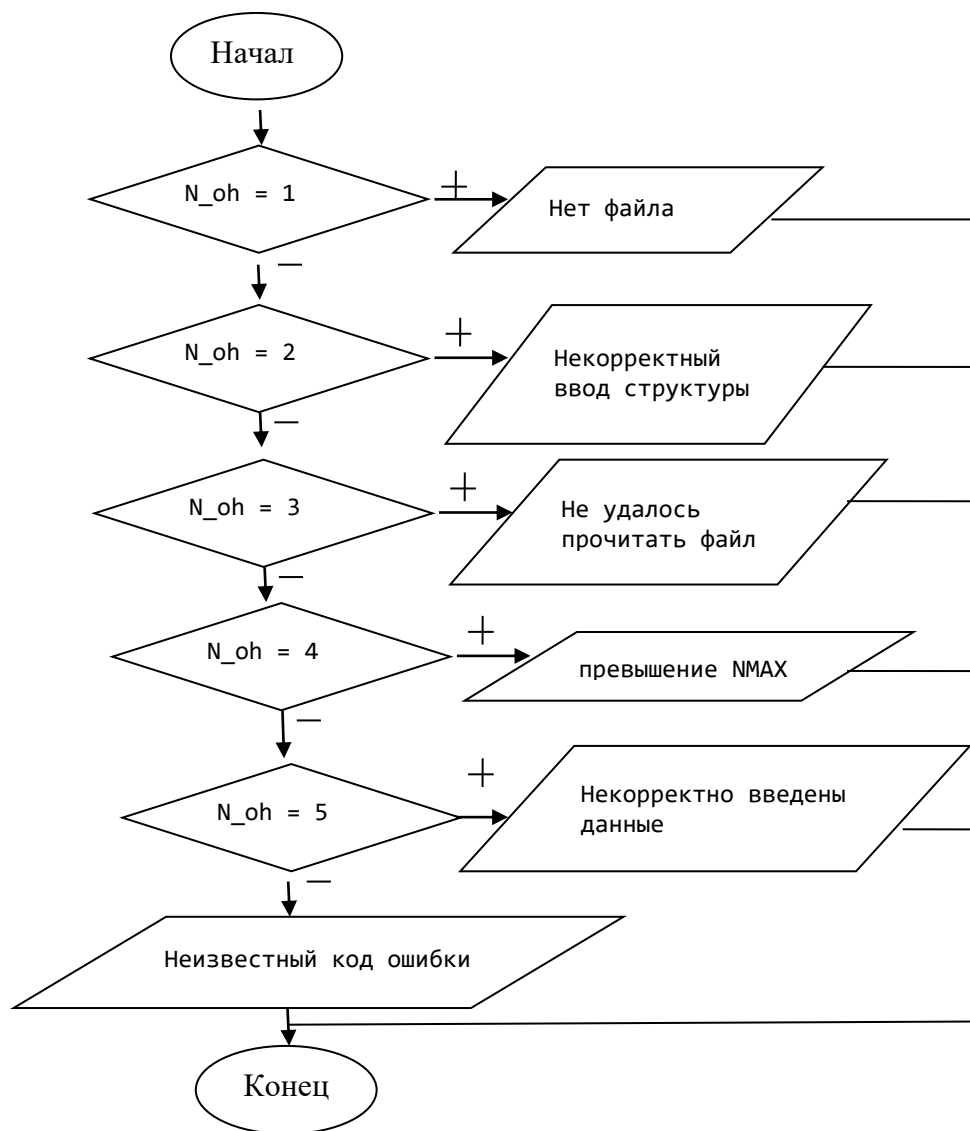
d. Описание параметров:

int N_oh – номер ошибки,

const char FNAME[] – имя файла

e. Ничего не возвращает

f. Блок-схема



```
/*
*
*   ИНФОРМАТИКА И ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА
*
*/
*****
```

```

*   Probject type: ConsoleApplication
*
*   Probject name: 2C_Laba2
*
*   File name: 2C_Laba2.cpp
*
*   Language: Cpp, MSVS 2017
*
*   Programmes: МЗО-107Б-18
*
*   Гордеев Никита
*
*   Онгарбаев Бауржан
*
*   Modified by: 28.04.2019
*
*   Created: 28.04.2019
*
*   Comment: Лабораторная работа №7
*
*****/

#include "pch.h"
#include <iostream>
#include <fstream>
#include <string>
#include <iomanip>
using namespace std;

//const char FNAME[] = "Text0.txt"; //файл не существует
//const char FNAME[] = "Text1.txt"; //буква в графе номер рейса
//const char FNAME[] = "Text2.txt"; //буква в графе вес груза
//const char FNAME[] = "Text3.txt"; //буква в графе количество контейнеров
//const char FNAME[] = "Text4.txt"; //номер рейса не целое число
//const char FNAME[] = "Text5.txt"; //количество контейнеров не целое число
//const char FNAME[] = "Text6.txt"; //привешение NMAX
//const char FNAME[] = "Text7.txt"; //файл пуст
//const char FNAME[] = "Text8.txt"; //номер самолёта слишком длинный
//const char FNAME[] = "Text9.txt"; //номер рейса слишком длинный
//const char FNAME[] = "Text10.txt"; //вес груза слишком большой
//const char FNAME[] = "Text11.txt"; //количество контейнеров слишком большое
//const char FNAME[] = "Text12.txt"; //последняя строка заполнена не полностью
//const char FNAME[] = "Text13.txt"; //отрицательное число в графе вес груза
//const char FNAME[] = "Text14.txt"; //отрицательное число в графе номер рейса
//const char FNAME[] = "Text15.txt"; //отрицательное число в графе количество контейнеров
const char FNAME[] = "Text.txt"; //файл исходных данных
const int NMAX = 10;

struct Plane
{
    int NReis; //номер рейса
    string NPlane; //номер самолёта
    double Mass; //вес груза
    int Kont; //количество контейнеров
};

struct Key
{
    int NReis; //номер рейса
    int ind; //индекс сортировки
};

bool InputPlane(Plane *pPlane, int &N, Key *Key); //ввод структуры из файла
void PrintPlane(Plane *pPlane, int N, Key *Key); //печать структуры
void SortPlane(int N, Key *Key); //сортировка структуры

```

```

void PrintTabl(Plane *pPlane, int N); //печать sum и конец таблицы
void Our(int N_oh, const char FNAME[]); //ошибки
void a() { setlocale(LC_ALL, "C"); } // подключение английского языка
void r() { setlocale(LC_ALL, "Russian"); } //подключение русского языка

int main() {
    system("color F0"); //экран белый, буквы
    черные
    int N;
    Plane APlane[11];
    Key Key[11];
    if (InputPlane(APlane, N, Key)) //ввод из файла
    {
        r();
        cout << "\tДо SortPlane\n";
        PrintPlane(APlane, N, Key); //печать массива
        SortPlane(N, Key); //сортировка
        r();
        cout << "\n\tПосле SortPlane\n";
        PrintPlane(APlane, N, Key); //печать массива
    }
    system("pause"); //пауза
    return 0;
}

void ShapkaTabl() {
    a();
    cout << char(218);
    cout << setfill(char(196)) << setw(4);
    cout << char(194);
    cout << setfill(char(196)) << setw(15);
    cout << char(194);
    cout << setfill(char(196)) << setw(15);
    cout << char(194);
    cout << setfill(char(196)) << setw(15);
    cout << char(194);
    cout << setfill(char(196)) << setw(15);
    cout << char(191) << endl;
    cout << char(179) << " " << char(252) << " ";
    cout << char(179); r(); cout << " № рейса "; a();
    cout << char(179); r(); cout << " бортовой номер"; a();
    cout << char(179); r(); cout << " вес груза "; a();
    cout << char(179); r(); cout << " контейнеры "; a();
    cout << char(179) << endl;
}

bool InputPlane(Plane * pPlane, int &N, Key *Key)
{
    r();
    bool f; //флаг верного чтения файла
    int i = 0; //счетчик числа самолётов

```

```

int j = 0; //счетчик длины графы номер самолёта
char a[1]; //вспомогательный массив
ifstream fin(FNAME); //открытие входного файла
ifstream fin2(FNAME); //открытие входного файла
if (!fin) //не удалось открыть файл
{
    Our(1, FNAME);
    return false;
}
i = 0; //число реально считанных записей
while (!fin.eof())
{
    if (i > NMAX - 1) { //превышение NMAX
        Our(4, FNAME);
        return false;
    }
    fin >> pPlane[i].NReis;
    if ((i == 0) && fin.eof()) //файл пуст
    {
        Our(6, FNAME);
        return false;
    }
    if (fin.fail()) {
        Our(2, FNAME);
        return false;
    }
    fin2 >> pPlane[i].NReis;
    fin >> pPlane[i].NPlane;
    if (fin.fail()) {
        Our(2, FNAME);
        return false;
    }
    fin2.read(a, 1); //уход от пробела
    fin2.read(a, 1);
    if (a[0] != 'Б')
    {
        Our(5, FNAME);
        return false;
    }
    fin2.read(a, 1);
    if (a[0] != '-')
    {
        Our(5, FNAME);
        return false;
    }
    fin2 >> j;
    if ((j < 1000) || (j > 9999))
    {
        Our(5, FNAME);
        return false;
    }
}

```

```

    }
    fin >> pPlane[i].Mass;
    fin2 >> pPlane[i].Mass;
    if (fin.fail()) {
        Our(2, FNAME);
        return false;
    }
    if (fin.eof()) {
        Our(8, FNAME);
        return false;
    }
    fin >> pPlane[i].Kont;
    fin2 >> pPlane[i].Kont;
    if (fin.fail()) {
        Our(2, FNAME);
        return false;
    }
    //введённые элементы меньше 0
    if ((pPlane[i].NReis < 0))
    {
        cout << "Отрицательный номер рейса в строке "<< i+1 << " равен "<< pPlane[i].NReis<<"\n";
        Our(3, FNAME);
        return false;
    }
    if ( (pPlane[i].Mass < 0) )
    {
        cout << "Отрицательная масса в строке " << i+1 << " равна " << pPlane[i].Mass << "\n";
        Our(3, FNAME);
        return false;
    }
    if ((pPlane[i].Kont < 0))
    {
        cout << "Отрицательное количество контейнеров в строке " << i+1 << " равно " << pPlane[i].Kont << "\n";
        Our(3, FNAME);
        return false;
    }
    i++;
    f = true;
} //while

N = i;
fin.close(); //закреть файл
for (int i = 0; i < N; i++)
{
    Key[i].NReis = pPlane[i].NReis;
    Key[i].ind = i;
}
return f;
} //end InputPlane
int SumPlane(Plane *pPlane, int N) {

```



```

    int Sum = 0; //сумма контейнеров
    for (int i = 0; i < N; i++)
    {
        Sum += pPlane[i].Kont;
    } //for i
    return Sum;
}

void PrintTabl(Plane *pPlane, int N)
{
    a();
    cout << char(195);
    cout << setfill(char(196)) << setw(4);
    cout << char(193);
    cout << setfill(char(196)) << setw(15);
    cout << char(193);
    cout << setfill(char(196)) << setw(15);
    cout << char(193);
    cout << setfill(char(196)) << setw(15);
    cout << char(193);
    cout << setfill(char(196)) << setw(15);
    cout << char(180) << endl;
    //заполнение
    cout << char(179); r(); cout << " Сумма = "; a(); cout << setfill(char(32)) << setw(25);
    cout << char(32) << " " << setfill(char(32)) << setw(27)
        << SumPlane(pPlane, N) << " " << char(179) << "\n";
    //конец табл
    cout << char(192);
    cout << setfill(char(196)) << setw(34);
    cout << char(196);
    cout << setfill(char(196)) << setw(30);
    cout << char(217) << endl;
}

//печать массива
void PrintPlane(Plane * pPlane, int N, Key *Key)
{
    ShapkaTabl(); //шапка
    int i; //индексная переменная
    //напечатаем и изменим
    for (i = 0; i < N; i++)
    {
        //запись
        cout << char(195);
        cout << setfill(char(196)) << setw(4);
        cout << char(197);
        cout << setfill(char(196)) << setw(15);
        cout << char(197);
        cout << setfill(char(196)) << setw(15);
        cout << char(197);
        cout << setfill(char(196)) << setw(15);
        cout << char(197);
        cout << setfill(char(196)) << setw(15);
    }
}

```

```

        cout << char(180) << endl;
        //заполнение табл
        cout << char(179) << " " << setfill(char(32)) << setw(1) << Key[i].ind + 1 << " ";
        cout << char(179) << " " << setfill(char(32)) << setw(12) << pPlane[Key[i].ind].NReis << " ";
        cout << char(179) << " " << setfill(char(32)) << setw(12);
        r(); cout << pPlane[Key[i].ind].NPlane << " "; a();
        cout << char(179) << " " << setfill(char(32)) << setw(12) << pPlane[Key[i].ind].Mass << " ";
        cout << char(179) << " " << setfill(char(32)) << setw(12) << pPlane[Key[i].ind].Kont << " ";
        cout << char(179) << endl;

    } //for i

    PrintTabl(pPlane, N);          //печать таблицы
} //end

void SortPlane(int N, Key *Key)
{
    int Min;           //минимальный элемент
    int jMin;           //индекс минимального элемента
    int iSort;          //граница отсортированной области
    int j;              //индексная переменная
    int Temp;           //обменная переменная

    for (iSort = 0; iSort < N - 1; iSort++)
    {
        //первый элемент из неупорядоченных назначаем минимальным
        Min = Key[iSort].NReis;           //минимум
        jMin = iSort;                     //его индекс

        //ищем минимальный элемент в оставшейся части массива
        for (j = iSort + 1; j < N; j++)
        {
            if (Key[j].NReis < Min)        //очередной кандидат на минимальный
            {
                //запоминаем минимальный элемент и его номер
                Min = Key[j].NReis;
                jMin = j;
            } //if
        } //for j

        //нашли минимум в неупорядоченной части массива
        //ставим его на место первого в неупорядоченной части массива
        //меняем элементы местами
        Temp = Key[iSort].NReis;
        Key[iSort].NReis = Key[jMin].NReis;
        Key[jMin].NReis = Temp;
        //сортируем индексный массив
        Temp = Key[iSort].ind;
        Key[iSort].ind = Key[jMin].ind;
        Key[jMin].ind = Temp;
    } //for iSort
}

```

```

}

void Our(int N_oh, const char FNAME[])
{
    switch (N_oh)
    {
        case 1: {cout << "\tОшибка! Файл " << FNAME << " не найден\n"; break; }
        case 2: {cout << "\nОшибка! Сбой при чтении файла " << FNAME << endl; break; }
        case 3: {cout << "\nОшибка! Отрицательные значения " << FNAME << "!\n"; break; }
        case 4: {cout << "\nОшибка! Превышение NMAX " << FNAME << "!\n"; break; }
        case 5: {cout << "\nОшибка! Файл " << FNAME << " заполнен неправильно\n"; break; }
        case 6: {cout << "\nОшибка! Файл " << FNAME << " пуст\n"; break; }
        default:
            cout << "Неизвестный код ошибки " << N_oh << endl;
    }
}
}

```

Корректные тесты

Корректный тест №1

Цель: Проверка работы программы, при правильном вводе

Исходные данные:

37 Б-8726 5.6 1
352 Б-3426 4.6 2
23 Б-1426 6.6 3
64 Б-1000 3.6 8
85 Б-8526 3.3 5
46 Б-3846 4.6 6
27 Б-3426 2.6 5
68 Б-3426 5.4 8

Ожидаемый результат:

1) 37164852

2) Сумма 38

Результат программы:

До SortPlane

№	№ рейса	бортовой номер	вес груза	контейнеры
1	37	Б-8726	5.6	1
2	352	Б-3426	4.6	2
3	23	Б-1426	6.6	3
4	64	Б-1000	3.6	8
5	85	Б-8526	3.3	5
6	46	Б-3846	4.6	6
7	27	Б-3426	2.6	5
8	68	Б-3426	5.4	8
Сумма =				38

После SortPlane

№	№ рейса	бортовой номер	вес груза	контейнеры
3	23	Б-1426	6.6	3
7	27	Б-3426	2.6	5
1	37	Б-8726	5.6	1
6	46	Б-3846	4.6	6
4	64	Б-1000	3.6	8
8	68	Б-3426	5.4	8
5	85	Б-8526	3.3	5
2	352	Б-3426	4.6	2
Сумма =				38

Для продолжения нажмите любую клавишу . . .

Вывод: Программа функционирует правильно.

Некорректные тесты

Некорректный тест №1

Цель: Проверка работы программы, если файл пуст


Исходные данные:

<>

Ожидаемый результат:

Файл FNAME пуст

Результат программы:

 C:\Users\potow\source\repos\lab3_variant7\Debug\lab3_variant7.exe

Ошибка! Файл Test_07.txt пуст
Для продолжения нажмите любую клавишу . . .

Вывод: Программа функционирует правильно.

Некорректный тест №2

Цель: Проверка работы программы, если файла не существует


Исходные данные:

<>

Ожидаемый результат:

Файл FNAME не найден

Результат программы:

 C:\Users\potow\source\repos\lab3_variant7\Debug\lab3_variant7.exe

Ошибка! Файл Test_00.txt не найден
Для продолжения нажмите любую клавишу . . .

Вывод: Программа функционирует правильно.

Некорректный тест №3

Цель: Проверка работы программы, если буква в графе номера рейсов

Исходные данные:

376 В-3726 5.6 44

22А О-3124 8.5 31

436 В-6527 241.7 12


541 Т-1243 63.6 34

749 К-3214 24.2 112

Ожидаемый результат:

Ошибка! Сбой при чтении файла FNAME

Результат программы:

 C:\Users\potow\source\repos\lab3_variant7\Debug\lab3_variant7.exe

Ошибка! Сбой при чтении файла Test_01.txt
Для продолжения нажмите любую клавишу . . .

Вывод: Программа функционирует правильно.

Некорректный тест №4

Цель: Проверка работы программы, если буква в графе вес груза

Исходные данные:

376 В-3726 Т 44

223 О-3124 8.5 31

436 В-6527 241.7 12


541 Т-1243 63.6 34

749 К-3214 24.2 112

Ожидаемый результат:

Ошибка! Сбой при чтении файла FNAME

Результат программы:

 C:\Users\potow\source\repos\lab3_variant7\Debug\lab3_variant7.exe

Ошибка! Сбой при чтении файла Test_02.txt
Для продолжения нажмите любую клавишу . . .

Вывод: Программа функционирует правильно.

Некорректный тест №5

Цель: Проверка работы программы, если буква в графе вес груза


Исходные данные:

376 В-3726 Т 44
223 О-3124 8.5 31
436 В-6527 241.7 12
541 Т-1243 63.6 34
749 К-3214 24.2 112

Ожидаемый результат:

Ошибка! Сбой при чтении файла FNAME

Результат программы:

 C:\Users\potow\source\repos\lab3_variant7\Debug\lab3_variant7.exe

Ошибка! Сбой при чтении файла Test_02.txt
Для продолжения нажмите любую клавишу . . .

Вывод: Программа функционирует правильно.

Некорректный тест №6

Цель: Проверка работы программы, если буква в графе количества контейнеров


Исходные данные:

376 В-3726 5.6 44
223 О-3124 8.5 Н
436 В-6527 241.7 12
541 Т-1243 63.6 34
749 К-3214 24.2 112

Ожидаемый результат:

Ошибка! Сбой при чтении файла FNAME

Результат программы:

 C:\Users\potow\source\repos\lab3_variant7\Debug\lab3_variant

Ошибка! Сбой при чтении файла Test_03.txt
Для продолжения нажмите любую клавишу . . .

Вывод: Программа функционирует правильно.

Некорректный тест №7

Цель: Проверка работы программы, если есть превышение NMAX


Исходные данные:

376 В-3726 5.6 44
223 О-3124 8.5 31
436 В-6527 241.7 12
541 Т-1243 63.6 34
749 К-3214 24.2 112
376 В-3726 5.6 44
376 В-3726 5.6 44
376 В-3726 5.6 44
376 В-3726 5.6 44
376 В-3726 5.6 44
376 В-3726 5.6 44

Ожидаемый результат:

Ошибка! Превышение NMAX FNAME!

Результат программы:

 C:\Users\potow\source\repos\lab3_variant7\Debug\lab3_variant7.exe

Ошибка! Превышение NMAX Test_06.txt!

Для продолжения нажмите любую клавишу . . .

Вывод: Программа функционирует правильно.

Некорректный тест №8

Цель: Проверка работы программы, если значения отрицательные

Исходные данные:

-3 В-3726 5.6 44

223 О-3124 8.5 31

436 В-6527 241.7 12


541 Т-1243 63.6 34

749 К-3214 24.2 112

Ожидаемый результат:

Ошибка! Отрицательные значения FNAME!

Результат программы:

 C:\Users\potow\source\repos\lab3_variant7\Debug\lab3_variant7.exe

Ошибка! Отрицательные значения Test_09.txt!

Для продолжения нажмите любую клавишу . . .

Вывод: Программа функционирует правильно.

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ «МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)»**

**О Т З Ы В
РУКОВОДИТЕЛЯ ПРАКТИКИ**

Студент Гордеев Никита Максимович

Институт №3 “Системы управления, информатика и электроэнергетика”

Кафедра №304 “Вычислительные машины, системы и сети”

Учебная группа М30-107Б

Направление подготовки (специальность) 09.03.01 “Информатика и вычислительная техника”
(шифр)(название направления, специальности)

Вид практики учебная

(учебная, производственная, преддипломная или другой вид практики)

Наименование предприятия: Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский авиационный институт (национальный исследовательский университет)»

Название структурного подразделения (отдел, лаборатория): Кафедра 304

План работ выполнен: _____ *(полностью/не полностью)*

Соответствие практики образовательным компетенциям:

ПСК 6, ПСК 8

Руководитель практики от МАИ

Чечиков Юрий Борисович /

(фамилия, имя, отчество) *(подпись)*

Москва 2019 г.