МИНИСТЕРСТВО науки и высшего ОБРАЗОВАНИЯ РОссИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(национальный исследовательский университет)»

Институт №3 «Системы управления, информатика и электроэнергетика»

Кафедра № 304 «Вычислительные машины, системы и сети»

Программирование

Отчет по лабораторной работе № 3

Структуры

Выполнили студенты группы M3О-111Б-21

Багиров Эльдар

Нуриев Наиль

Проверила доцент, к.т.н., Дмитриева Е.А.

Москва 2022 г.

Оглавление

[Задание 3](#_Toc101373486)

[Структурные схемы алгоритмов 4](#_Toc101373487)

[Функция main 4](#_Toc101373488)

[Функция Errors 5](#_Toc101373489)

[Функция memory\_for\_matrix 6](#_Toc101373490)

[Функция input\_matrix 7](#_Toc101373491)

[Функция matrix\_from\_file 8](#_Toc101373492)

[Функция print\_matrix 9](#_Toc101373493)

[Функция maxim 10](#_Toc101373494)

[Функция product 11](#_Toc101373495)

[Код программы 12](#_Toc101373496)

[Тестирование программы 15](#_Toc101373497)

[Тестирование некорректных режимов 15](#_Toc101373498)

[Тестирование корректных режимов 17](#_Toc101373499)

[Вывод 20](#_Toc101373500)

# Задание

Кафедра: 302 Курс: ИНФОРМАТИКА

Практика

**ВАРИАНТ №3**

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

**ТУ-154М Б-3726 131 12.1**

марка ЛА бортовой количество груз

номер пассажиров

1) подготовить программу, сортирующую записи с использованием индексной сортировки линейным методом в порядке возрастания числовой части бортовых номеров; определить суммарное количество пассажиров, вес грузов; результат печатать в виде таблицы;

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

Чтение данных из 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.

1. Структурные схемы алгоритмов

## Функция main

Главная (основная – main) функция

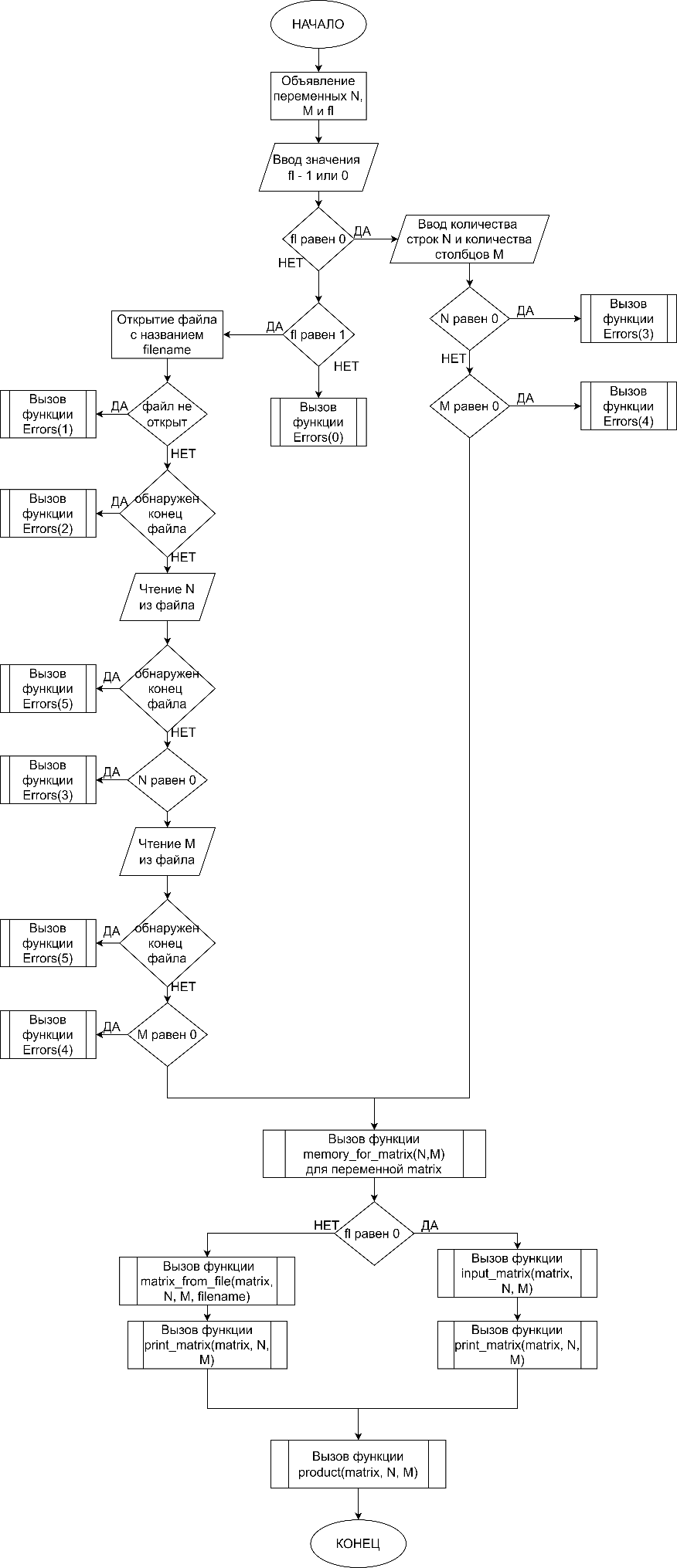


Рисунок 1. Структурная схема алгоритма функции main

## Функция Errors

Функция вывода ошибок по коду

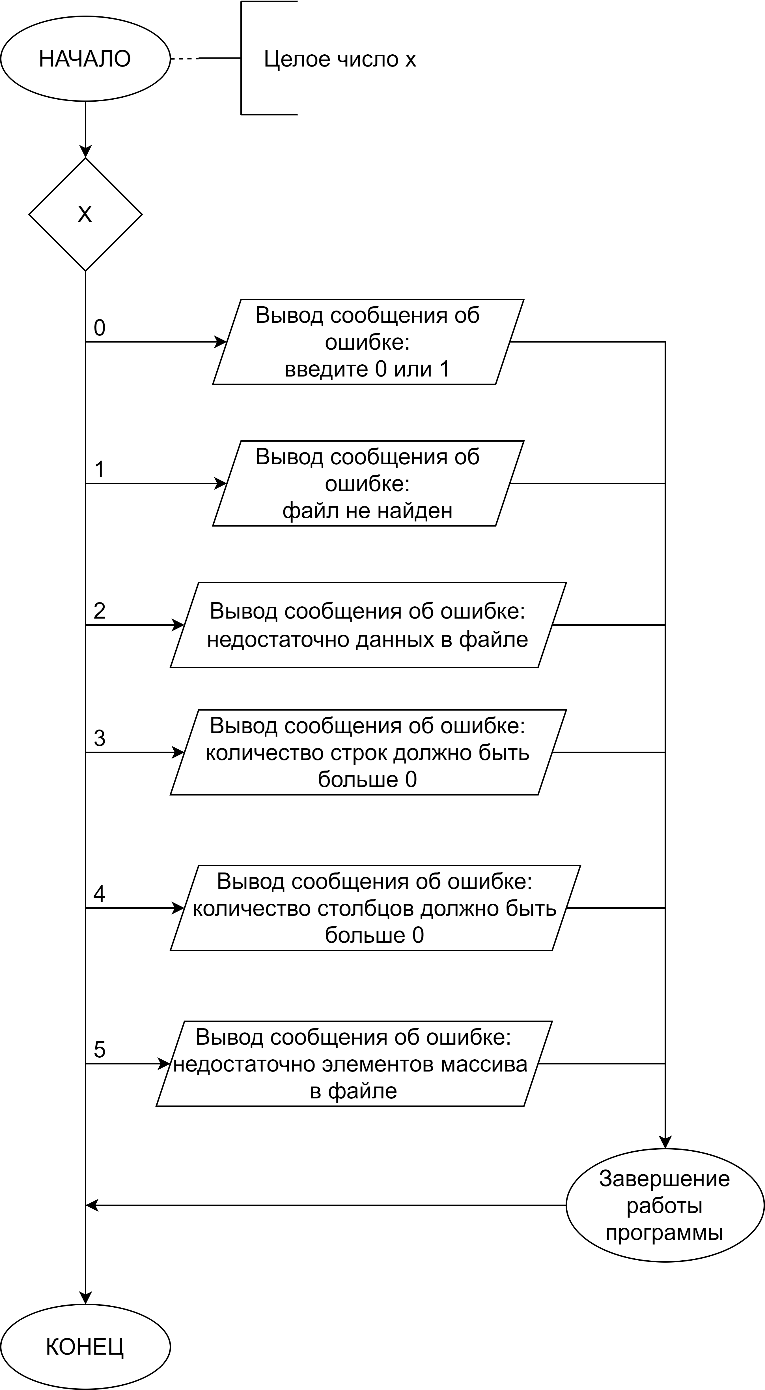


Рисунок 2. Структурная схема алгоритма функции Errors

## Функция memory\_for\_matrix

Функция выделения памяти для матрицы

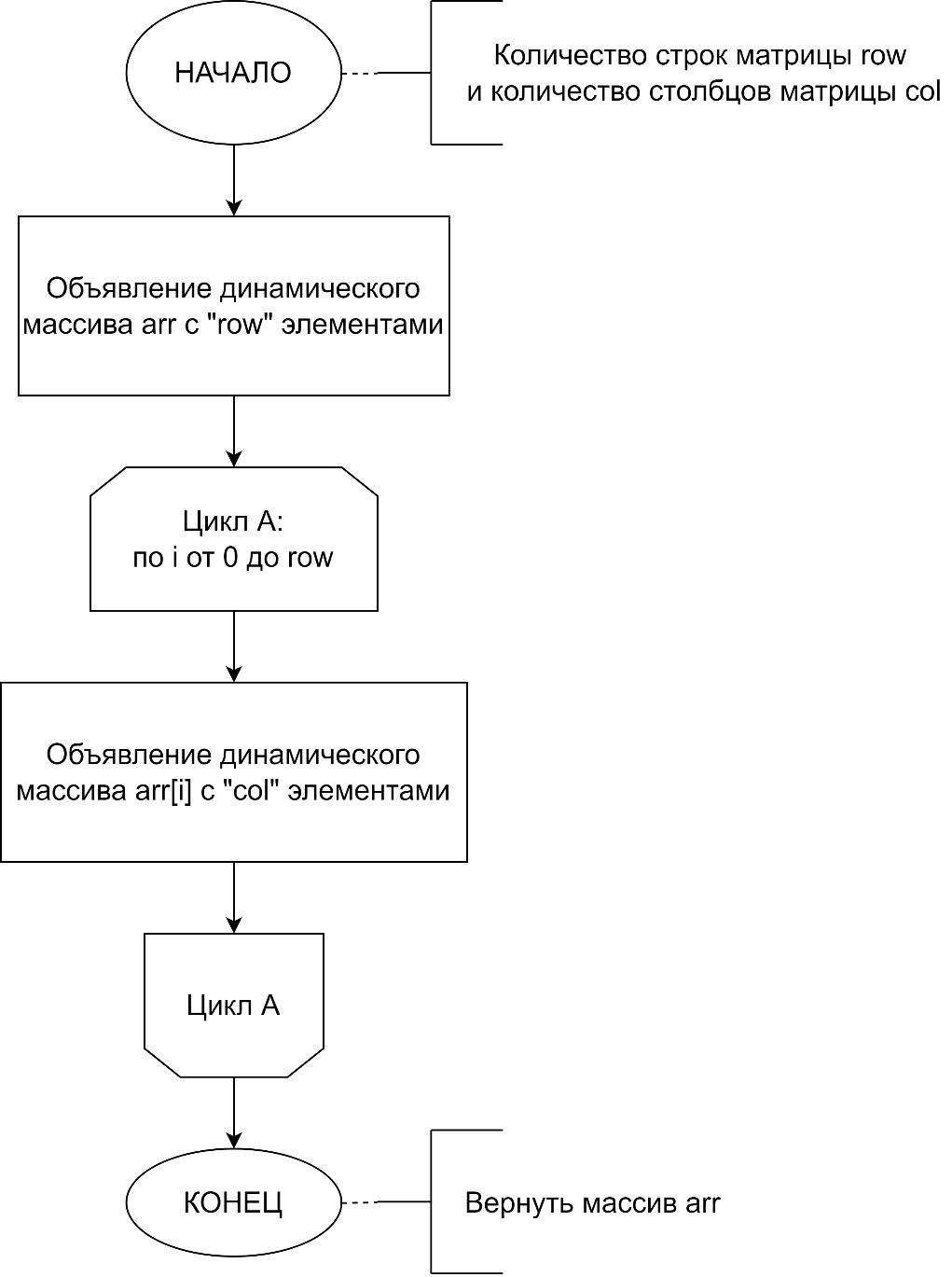


Рисунок 3. Структурная схема алгоритма функции memory\_for\_matrix

## Функция input\_matrix

Ввод матрицы из консоли

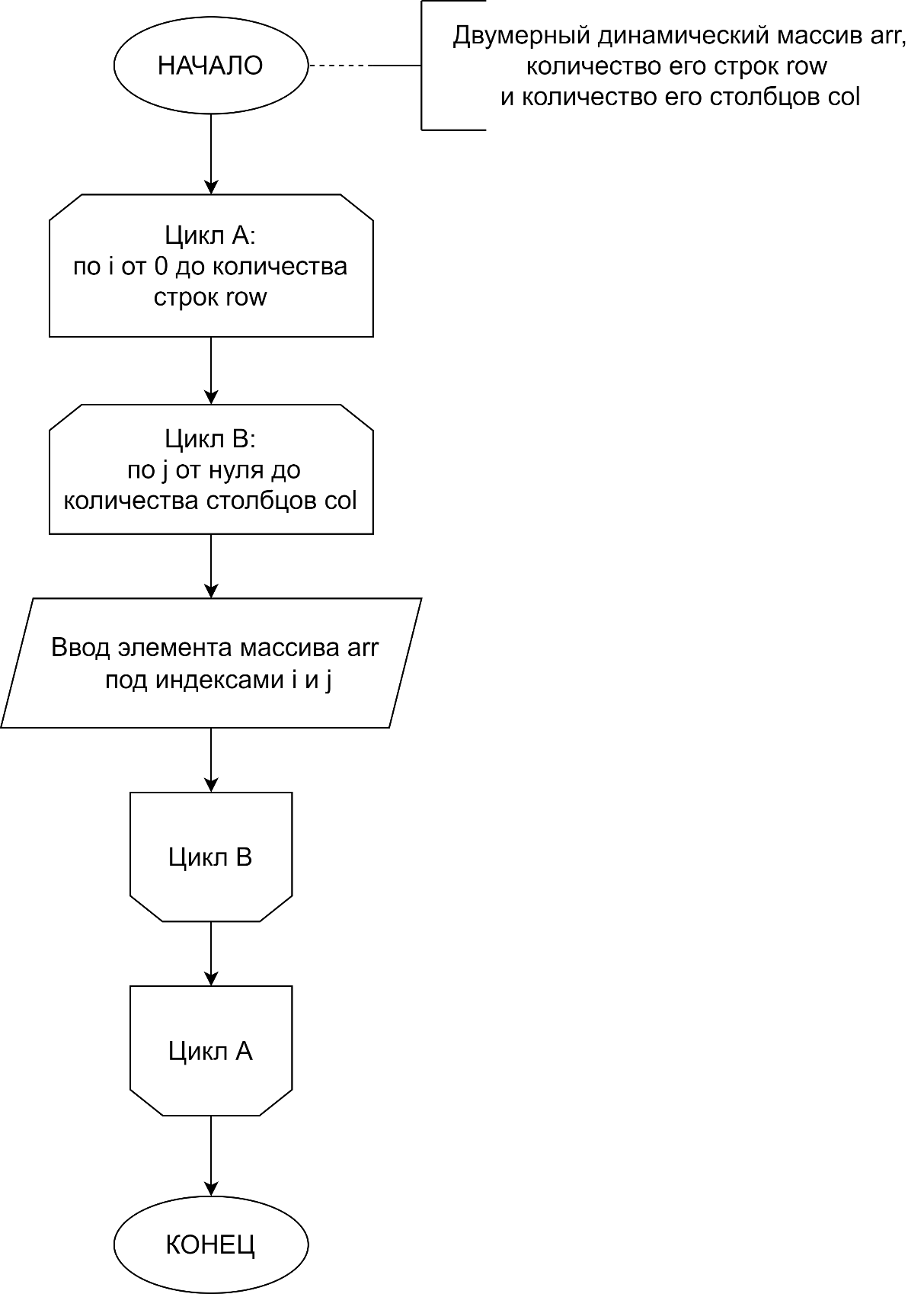


Рисунок 4. Структурная схема алгоритма функции input\_matrix

## Функция matrix\_from\_file

Ввод матрицы из файла

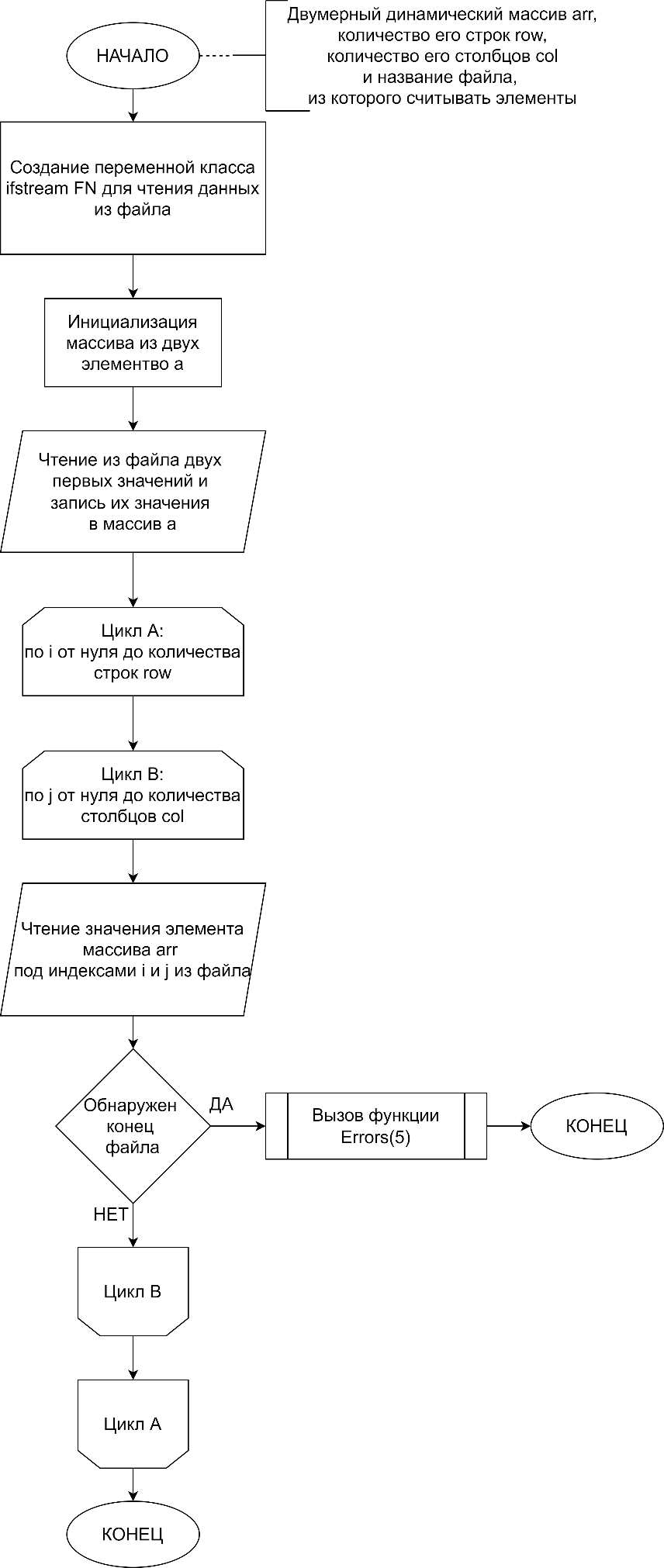


Рисунок 5. Структурная схема алгоритма функции matrix\_from\_file

## Функция print\_matrix

Печать матрицы

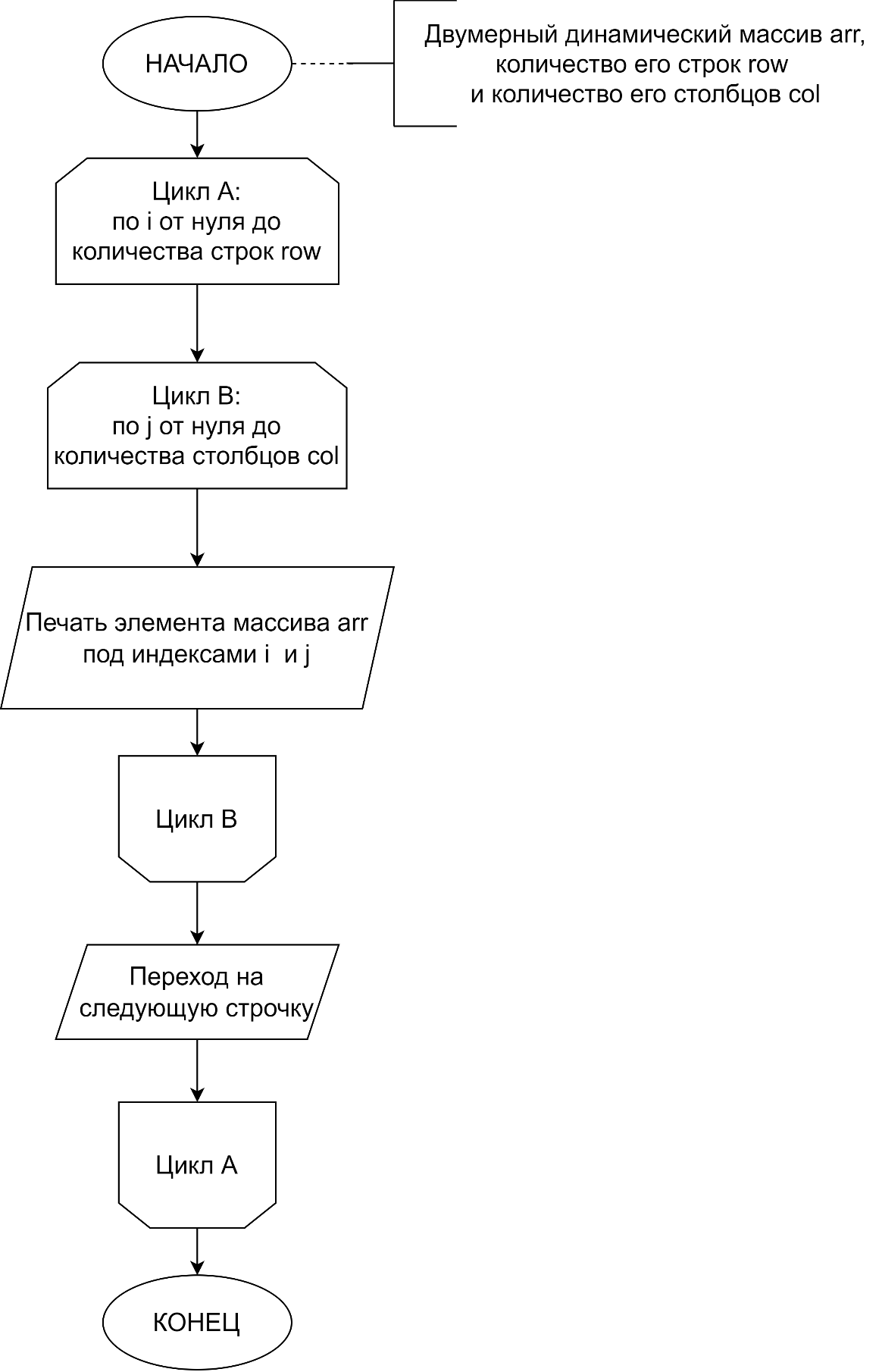


Рисунок 6. Структурная схема алгоритма функции print\_matrix

## Функция maxim

Нахождение максимума среди двух чисел

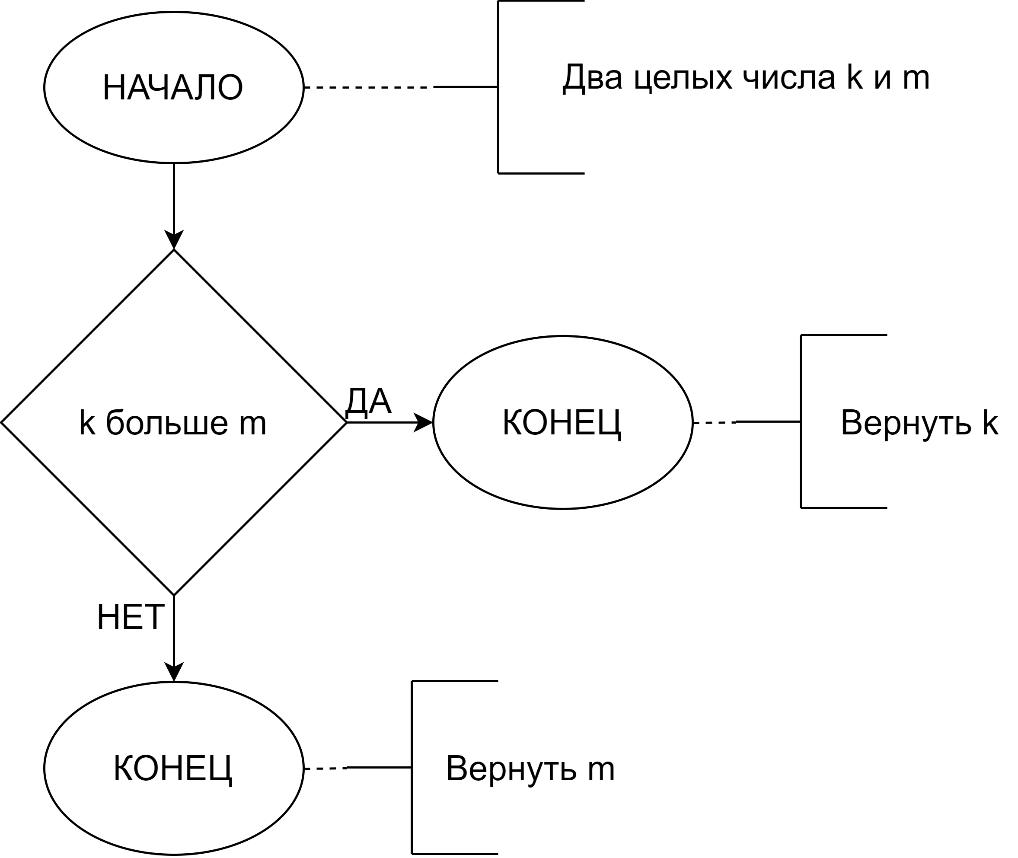


Рисунок 7. Структурная схема алгоритма функции maxim

## Функция product

Нахождение произведения чисел в строках матрицы

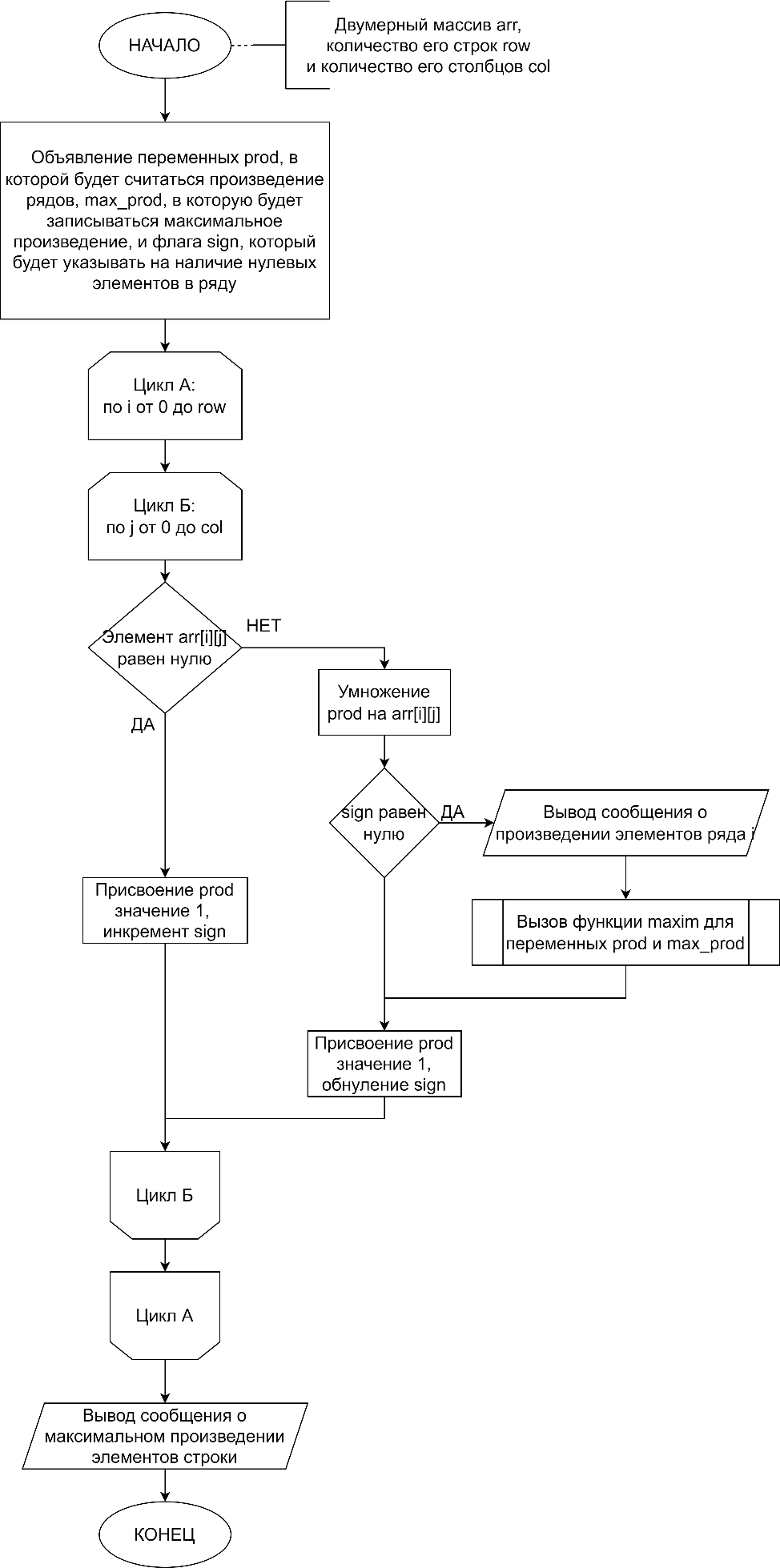


Рисунок 8. Структурная схема алгоритма функции product

# Код программы

#include <iostream>

#include <fstream>

using namespace std;

// ФУНКЦИЯ ДЛЯ ВЫВОДОВ ОШИБКИ ПО КОДАМ

void Errors(int x) {

switch (x) {

case 0:

cout << "Ошибка! Перезапустите программу и введите \"1\" или \"0\"!";

exit(0);

case 1:

cout << "Ошибка! Файл не найден!" << endl;

exit(0);

case 2:

cout << "Ошибка! В файле недостаточно данных!" << endl;

exit(0);

case 3:

cout << "Ошибка! Количество строк должно быть больше нуля!" << endl;

exit(0);

case 4:

cout << "Ошибка!Количество столбцов должно быть больше нуля!" << endl;

exit(0);

case 5:

cout << "Ошибка! В файле недостаточно элементов массива!" << endl;

exit(0);

}

}

// ФУНКЦИЯ ДЛЯ ВЫДЕЛЕНИЯ ПАМЯТИ ПОД МАТРИЦУ

int\*\* memory\_for\_matrix(int row, int col) {

int\*\* arr = new int\* [row];

for (int i = 0; i < row; i++)

arr[i] = new int[col];

return arr;

}

// ФУНКЦИЯ ВВОДА МАТРИЦЫ С РУКИ

void input\_matrix(int\*\* arr, int row, int col) {

for (int i = 0; i < row; i++)

for (int j = 0; j < col; j++)

cin >> arr[i][j];

}

// ФУНКЦИЯ ВВОДА МАТРИЦЫ ИЗ ФАЙЛА

void matrix\_from\_file(int\*\* arr, int row, int col, const string filename) {

ifstream FN;

int a[2];

FN.open(filename);

FN >> a[0] >> a[1];

for (int i = 0; i < row; i++)

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

FN >> arr[i][j];

if (FN.eof())

if (j != col - 1)

Errors(5);

}

}

// ФУНКЦИЯ ПЕЧАТИ МАТРИЦЫ

void print\_matrix(int\*\* arr, int row, int col) {

for (int i = 0; i < row; i++) {

for (int j = 0; j < col; j++)

cout << arr[i][j] << "\t";

cout << endl;

}

}

// ФУНКЦИЯ ДЛЯ НАХОЖДЕНИЯ МАКСИМУМА

long long maxim(long long ber, long long ike) {

if (ber > ike)

return ber;

else

return ike;

}

// ФУНКЦИЯ ДЛЯ ННАХОЖДЕНИЯ ПРОИЗВЕДЕНИЙ СТРОК МАТРИЦЫ ПО ЗАДАНИЮ

void product(int\*\* arr, int row, int col) {

long long prod = 1;

long long max\_prod = 0;

int sign = 0;

for (int i = 0; i < row; i++) {

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

if (arr[i][j] == 0) {

prod = 1;

sign++;

break;

}

/\* prod = prod \* arr[i][j]; \*/

prod = prod \* \*(\*(arr + i) + j);

}

if (sign == 0) {

cout << "Произведение элементов " << i + 1 << " строки: " << prod << endl;

max\_prod = maxim(max\_prod, prod);

}

else {

cout << "Произведение элементов " << i + 1 << " строки: не посчитано - содержится 0" << endl;

}

prod = 1;

sign = 0;

}

if (max\_prod == 0) {

cout << "Максимальное произведение не найдено - все строки содержат 0" << endl << endl;

}

else {

cout << "Максимальное произведение элементов строки: " << max\_prod << endl << endl;

}

}

const string filename = "file.txt";

//const string filename = "test1.txt";

//const string filename = "test2.txt";

//const string filename = "err\_test1.txt";

//const string filename = "err\_test2.txt";

//const string filename = "no\_file.txt";

//const string filename = "hollow\_file.txt";

//const string filename = "err\_test3.txt";

//const string filename = "err\_test4.txt";

int main()

{

setlocale(LC\_ALL, "rus");

system("color F0");

int N, M;

int fl;

cout << "Введите \"0\", если хотите ввести данные с руки, \"1\", если хотите считать их из файла" << endl;

cin >> fl;

ifstream fn;

if (fl == 0) {

cout << "Введите количество строк: ";

cin >> N;

if (N == 0)

Errors(3);

cout << endl << "Введите количество столбцов: ";

cin >> M;

if (M == 0)

Errors(4);

cout << endl;

}

else if (fl == 1) {

fn.open(filename);

if (!fn.is\_open())

Errors(1);

fn >> N;

if (fn.eof())

Errors(2);

if (N == 0)

Errors(3);

fn >> M;

if (M == 0)

Errors(4);

if (fn.eof())

Errors(2);

cout << "Количество строк: " << N << endl;

cout << "Количество столбцов: " << M << endl;

}

else {

Errors(0);

}

// ВЫДЕЛЕНИЕ ПАМЯТИ ПОД МАТРИЦУ

int\*\* matrix = memory\_for\_matrix(N, M);

// ЧТЕНИЕ МАТРИЦЫ С РУКИ ИЛИ ИЗ ФАЙЛА

if (fl == 0) {

cout << "Введите подряд " << N \* M << " целых чисел, разделенных пробелом " << endl;

input\_matrix(matrix, N, M);

cout << "Введенная матрица:" << endl;

print\_matrix(matrix, N, M);

}

else {

matrix\_from\_file(matrix, N, M, filename);

cout << "Полученная матрица:" << endl;

print\_matrix(matrix, N, M);

}

// РАССЧЕТ ПРОИЗВЕДЕНИЙ СТРОК МАТРИЦЫ ПО ЗАДАНИЮ

product(matrix, N, M);

return 0;

}

# Тестирование программы

## Тестирование некорректных режимов

**Тест №1**

Цель теста: проверить работу программы при отсутствии файла

Исходные данные:

Ожидаемый результат: вывод сообщения об ошибке: «Ошибка! Файл не найден!»

Полученный результат:



Вывод: полученный результат совпал с ожидаемым. Тест ошибки не обнаружил.

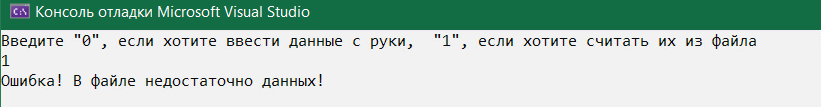
**Тест №2**

Цель теста: проверить работу программы при пустом файле

Исходные данные: (пустой файл)

Ожидаемый результат: вывод сообщения об ошибке: «Ошибка! В файле недостаточно данных!»

Полученный результат:



Вывод: полученный результат совпал с ожидаемым. Тест ошибки не обнаружил.

**Тест №3**

Цель теста: проверить работу программы при отсутствии некоторых данных

Исходные данные: 1

Ожидаемый результат: вывод сообщения об ошибке: «Ошибка! В файле недостаточно данных!»

Полученный результат:



Вывод: полученный результат совпал с ожидаемым. Тест ошибки не обнаружил.

**Тест №4**

Цель теста: проверить работу программы при нулевом количестве строк

Исходные данные: 0 5

Ожидаемый результат: вывод сообщения об ошибке: «Ошибка! Количество строк должно быть больше нуля!»

Полученный результат:



Вывод: полученный результат совпал с ожидаемым. Тест ошибки не обнаружил.

**Тест №5**

Цель теста: проверить работу программы при нулевом количестве столбцов

Исходные данные: 6 0

Ожидаемый результат: вывод сообщения об ошибке: «Ошибка!Количество столбцов должно быть больше нуля!»

Полученный результат:



Вывод: полученный результат совпал с ожидаемым. Тест ошибки не обнаружил.

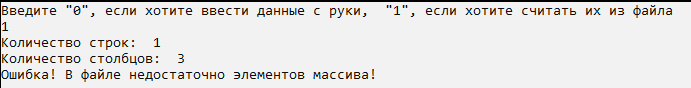
**Тест №6**

Цель теста: проверить работу программы при недостаточном количестве элементов массива

Исходные данные: 1 3 2 3

Ожидаемый результат: вывод сообщения об ошибке: «Ошибка! В файле недостаточно элементов массива!»

Полученный результат:



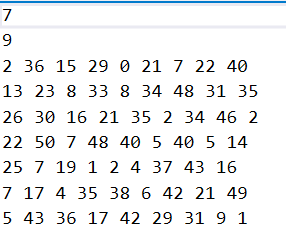
Вывод: полученный результат совпал с ожидаемым. Тест ошибки не обнаружил.

## Тестирование корректных режимов

**Тест №1**

Цель теста: проверить работу программы в корректной области исходных данных

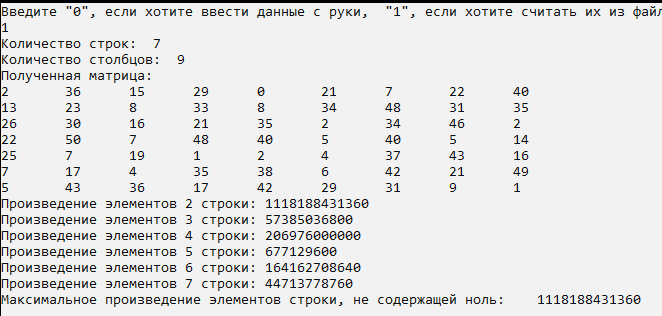
Исходные данные:



Ожидаемый результат:

Максимальное произведение элементов строки, не содержащей ноль: 1118188431360

Полученный результат:

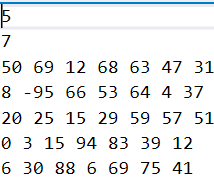


Вывод: полученный результат совпал с ожидаемым. Тест ошибки не обнаружил.

**Тест №2**

Цель теста: проверить работу программы в корректной области исходных данных

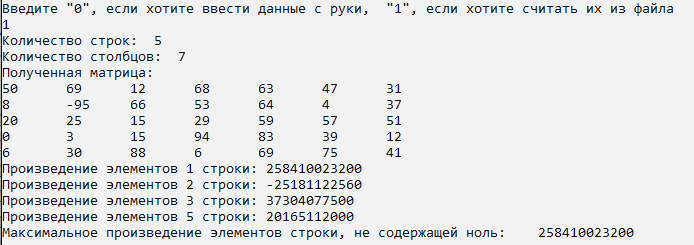
Исходные данные:



Ожидаемый результат:

Максимальное произведение элементов строки, не содержащей ноль: 258410023200

Полученный результат:



Вывод: полученный результат совпал с ожидаемым. Тест ошибки не обнаружил.

**Тест №3**

Цель теста: проверить работу программы в корректной области исходных данных

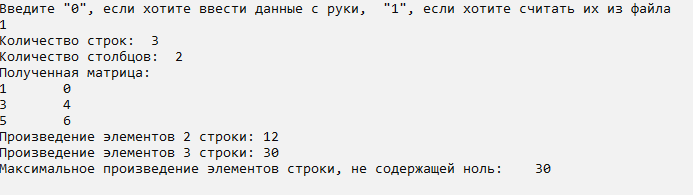
Исходные данные:



Ожидаемый результат:

Максимальное произведение элементов строки, не содержащей ноль: 30

Полученный результат:



Вывод: полученный результат совпал с ожидаемым. Тест ошибки не обнаружил.

# Вывод

Был изучен параметризированный алгоритм работы с двумерным массивом, ввод данных в двумерный массив из консоли и из файла. Были составлены структурные схемы алгоритмов функций для дальнейшего написания кода. В результате проведения тестов ошибок в программе не обнаружено. Разработка программы завершена на основании того, что полученные результаты совпали с ожидаемыми, набор тестов считаем полным.