МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ»

(ФГБОУ ВО «ВГТУ», ВГТУ)

Факультет информационных технологий и компьютерной безопасности

(факультет)

Кафедра Систем управления и информационных технологий в строительстве

КУРСОВОЙ ПРОЕКТ

по дисциплине:«Основы программирования и алгоритмизации» .

Тема: «Разработка программы табулирования функции» .

Расчетно-пояснительная записка

Разработал студент И.И. Иванов

Подпись, дата Инициалы, фамилия

Руководитель О.В. Курипта

Подпись, дата Инициалы, фамилия

Члены комиссии

Подпись, дата Инициалы, фамилия

Подпись, дата Инициалы, фамилия

Подпись, дата Инициалы, фамилия

Нормоконтролер О.В. Курипта

Подпись, дата Инициалы, фамилия

Защищена \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

дата

ВОРОНЕЖ

2022

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

(ФГБОУ ВО «ВГТУ», ВГТУ)

КафедраСистем управления и информационных технологий в строительстве

ЗАДАНИЕ

на курсовой проект

по дисциплине: «Основы Программирования и Алгоритмизации»

Тема: «Разработка программы табулирования функции»

..

Студент бИСТ-222 Летников Дионис Михайлович

Группа, фамилия, имя, отчество

Номер варианта 32.

Технические условия: процессор: Intel(R) Celeron(R) CPU G1820 @ 2.70GHz, NVIDIA GT 240, ОС: Windows 10, ОЗУ: 4.00 ГБ 64-разр.

Содержание и объем работы (графические работы, расчеты и прочее):.

стр., иллюстр., табл., .v

Сроки выполнения этапов: анализ и постановка задачи (01.10-15.10); разработка пошаговой детализации программы (16.10-10.11); реализация программы (11.11-01.12); тестирование программы (01.12-10.12); оформление пояснительной записки (02.12-12.12).

.

Срок защиты курсового проекта:

Руководитель О.В. Курипта .

Подпись, дата Инициалы, фамилия

Задание принял студент

Подпись, дата Инициалы, фамилия

Замечания руководителя

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 5](#_Toc121568752)

[1 Постановка задачи 6](#_Toc121568753)

[2 Конструирование программы 7](#_Toc121568754)

[2.1 Пошаговая детализация решения 7](#_Toc121568755)

[2.2 Описание функций 7](#_Toc121568756)

[3 Тестирование программы 9](#_Toc121568757)

[ЗАКЛЮЧЕНИЕ 10](#_Toc121568758)

[СПИСОК ИСРОЛЬЗОВАННЫХ ИСТОЧНИКОВ 11](#_Toc121568759)

[ПРИЛОЖЕНИЕ 12](#_Toc121568760)

# **ВВЕДЕНИЕ**

Целью курсового проекта является разработка программы, позволяющей осуществлять подсчёт функций с конкретно заданным аргументом или с рядом аргументов из заданного промежутка и построение их графиков, и занесение табулирования в файл. А также осуществлять работу с задаваемым пользователем массивом: находить количество отрицательных элементов в строках, содержащих хотя бы один нулевой элемент, и находить седловую точку матрицы.

Задачи:

1. Провести анализ структур данных для реализации подсчётов.
2. Разработать алгоритм программы, позволяющий решать поставленную задачу.
3. Реализовать программный продукт на языке С.

Программа, решающая поставленные задачи, написана на языке С в интегрированной среде разработки «Microsoft Visual Studio 2022».

## 1. Постановка задачи

Согласно заданию нам необходимо разработать программу работы с двумя функциями:

И

Для решения данной задачи необходимо разработать программу по принципу модульного построения, то есть разбить программу на отдельные части, которые будут выполнять какое-то действие. В этом нам поможет синтаксис функций в языке Си.

Основные задачи, которые должна выполнять программа:

1. Считать значение функции Y(x) и V(x) при аргументах, заданных пользователем;
2. Табулировать функции Y(x) и V(x) на промежутке с шагом, которые задаёт пользователь;
3. Выводить таблицу табулированных значений;
4. Рисовать график для табулированных значений;
5. Заносить значения табуляции в файл.
6. Находить количество отрицательных элементов в строках, содержащих хотя бы один нулевой элемент
7. Находить седловую точку матрицы.

Внимательно посмотрев на функции, мы понимаем, что для первой из них необходимо предусмотреть область допустимых значений, то есть первое слагаемое не должно равняться нулю, то есть аргумент не должен принимать значение -1. Это будет необходимо прописать в коде программы. Для второй функции область допустимых значений аргумента принадлежит R, то есть может являться любым числом.

При завершении написании кода, чтобы убедиться в правильности подсчётов на интервале необходимо посмотреть на графики. График первой функции представлен на рисунке 1. График второй функции представлен на рисунке 2.

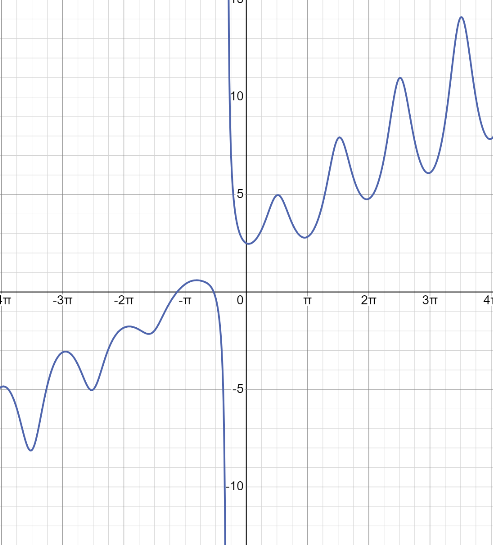


Рисунок 1 – График первой функции.

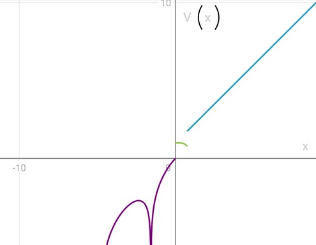


Рисунок 2 – График второй функции.

Все задачи были поставлены. Далее начнётся конструирование самой программы, где наглядно будет показан алгоритм и ветвления программы. Алгоритм и ветвления будут проиллюстрированы с помощью блок-схем.

## 2. Конструирование программы

Для удобства разобьём всю задачу на подзадачи. Это упростит понимание всего объёма работы (рисунок 2.1).

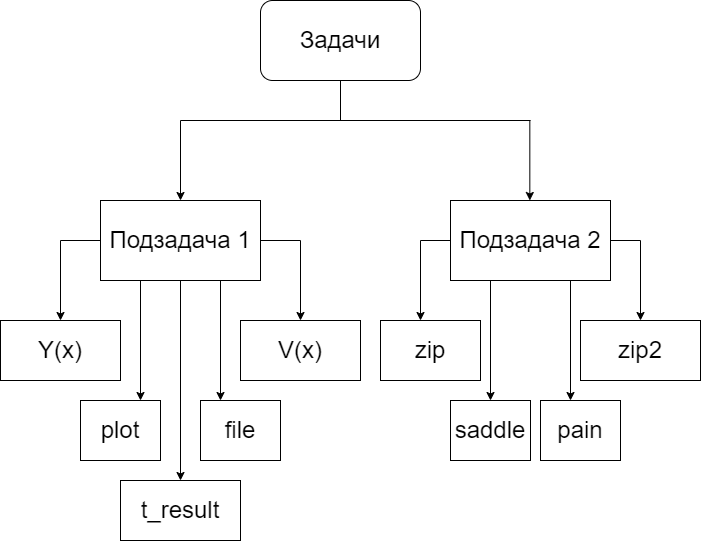


Рисунок 3 – Задачи

## 2.1 Описание функций

В языке C все блоки кода должны находиться в функциях. Для того чтобы код не повторялся, и программа работала эффективно, правильно помещать части кода, отвечающие за какие-либо вычисления в отдельные функции, оставляя в главной функции только ввод данных и их вывод.

Описание написанных функций представлено в таблице 1.

Таблица 1 –Используемые функции.

|  |  |  |
| --- | --- | --- |
| Описание функций | | |
| Название | Назначение | Примечание |
| double sch\_y(double x) | Вычисление функции Y(x) | Возвращает результат вычисления монотонной функции |
| double sch\_v(double x) | Вычисление функции V(x) | Возвращает результат вычисления кусочной функции |
| void t\_result(Tsch f, double xn, double xk, double h) | Рисует таблицу табулированных значений функции |  |
| void plot(double x0, double x1, Tsch f) | Рисует график функции |  |
| void file(Tsch f, double xn, double xk, double h, int z); | Выводит результат табуляции в файл |  |
| int\* zip(int\* array, int n, int m); | Заполнение двумерного массива | Возвращает указатель на заполненный массив |
| int zip2(int\* array, int n, int m); | Выводит двумерный массив | Возвращает 0; |
| void saddle(int\* a, int n, int m); | Находит седловую точку матрицы |  |
| void pain(int\* a, int n, int m); | находит количество отрицательных элементов в строках, содержащих хотя бы один нулевой элемент |  |

Блок-схемы функций представлены на рисунках ниже.

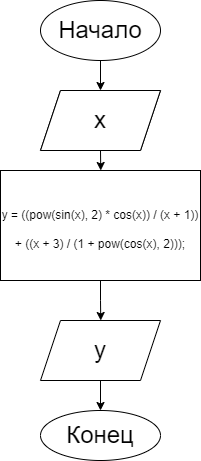


Рисунок 4 – Блок-схема функции подсчёта Y(x)

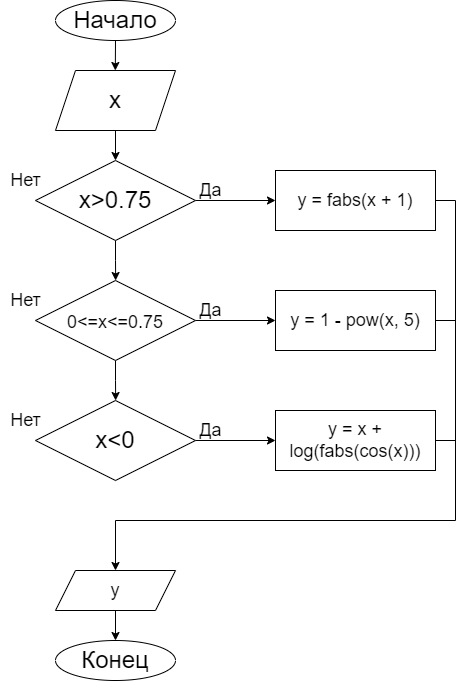


Рисунок 5 – Блок-схема функции подсчёта V(x)

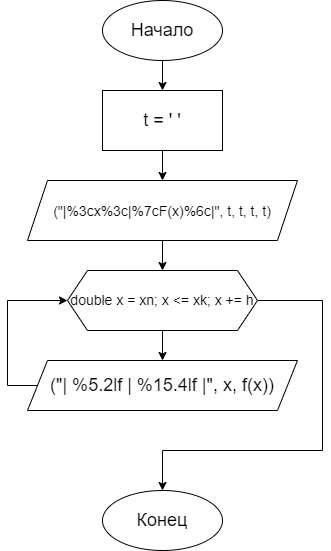


Рисунок 6 – Блок-схема для вывода расчёта табуляции.

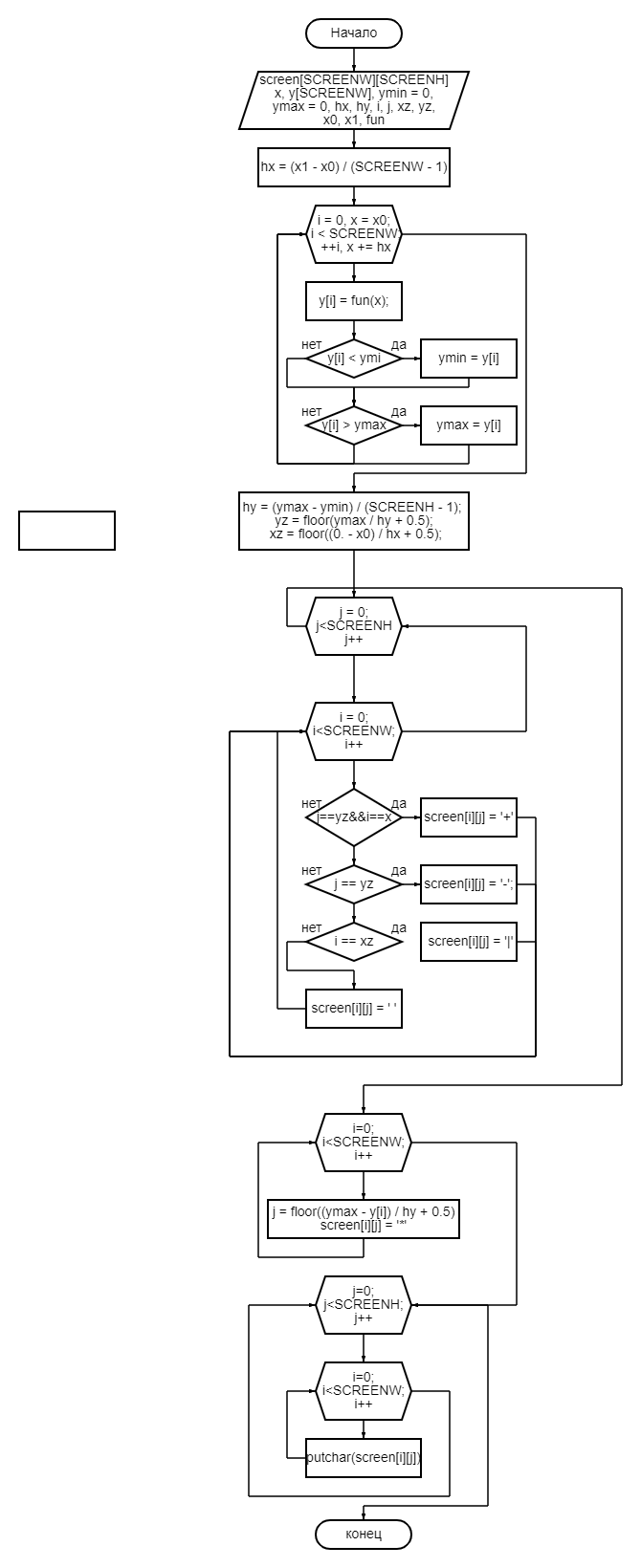


Рисунок 7 – Блок-схема вывода графика.

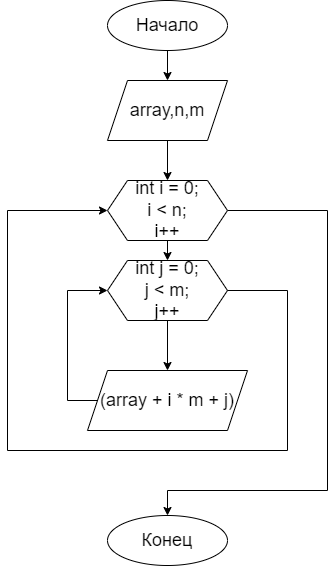


Рисунок 8 – Блок-схема заполнения двумерного массива.

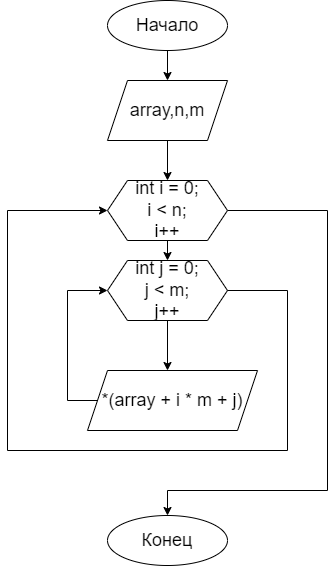


Рисунок 9 – Блок-схема вывода двумерного массива.

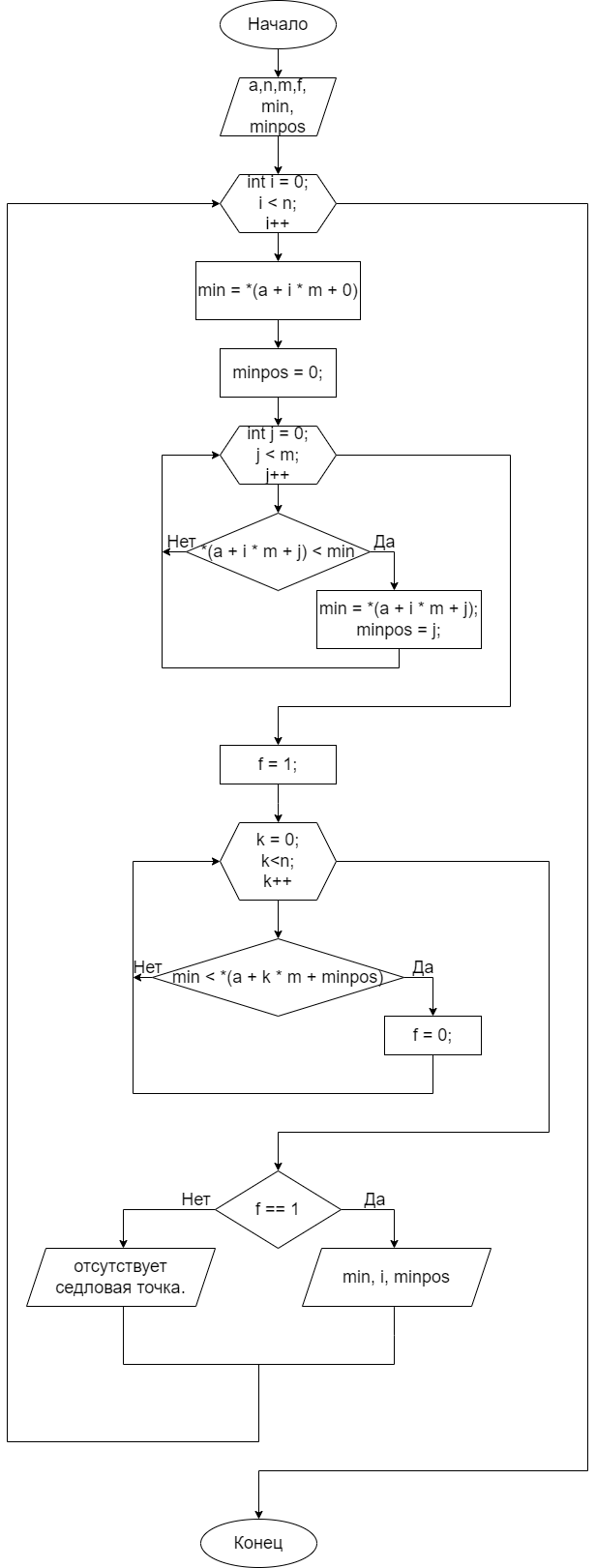


Рисунок 10 – Блок-схема поиска седловой точки.

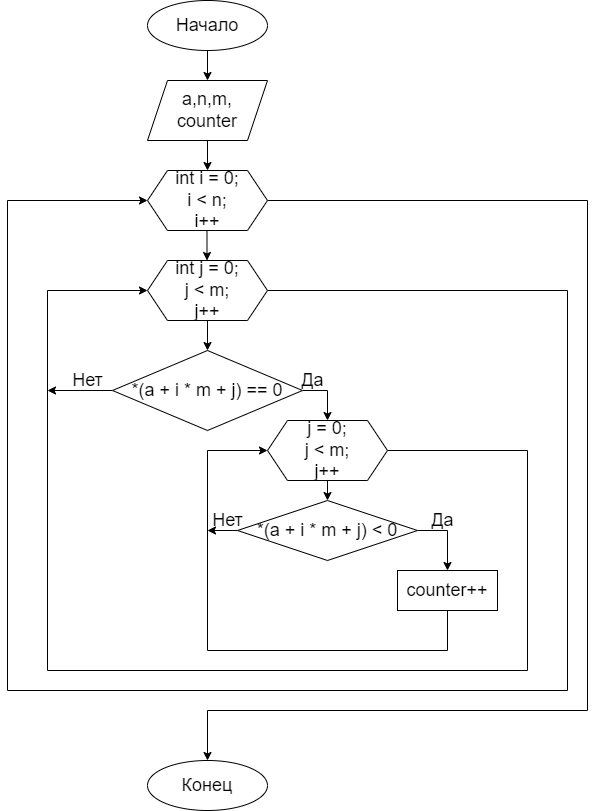


Рисунок 11 – Блок-схема поиска количества отрицательных элементов

Краткое описание переменных, использованных в функции main:

1. das – структура для хранения значений;

2. int r – хранение пользовательского ответа;

3. int n – хранит заданное пользователем количество строк;

4. int m – хранит заданное пользователем количество столбцов;

5. char s1 – хранит левую скобку заданную пользователем;

6. char s2 – хранит правую скобку заданную пользователем;

7. char semicolon – хранит знак точки с запятой при задании интервалов;

8. int\* a – двумерный массив.

**3. Тестирование программы**

Полное тестирование будет проходить с данными, о которых шла речь в разделе 1 «Постановка задачи». При запуске программы нас встречает сама рамка с ФИО студента, группой и меню, где необходимо выбрать действие (Рисунок 12).

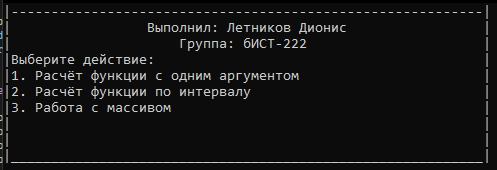


Рисунок 12 – Меню программы.

Далее пользователю необходимо выбрать действие, но мы пойдём по порядку. Выбираю пункт 1, где у нас уточняется для какой функции произвести расчёт, для примера выберем первую, после чего мы должны ввести аргумент (Рисунок 13).

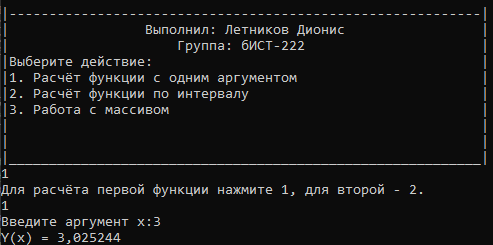


Рисунок 13 – Расчёт функции с аргументом.

Далее работаем с расчётом функции по интервалу. Выбираем пункт 2, далее выбираем первую функцию, задаём интервал и шаг табуляции, затем выводится табуляция. После чего у нас спрашивается хотим ли мы вывести график функции и полученную табуляцию в файл. Результаты представлены на рисунках 14, 15, 16.

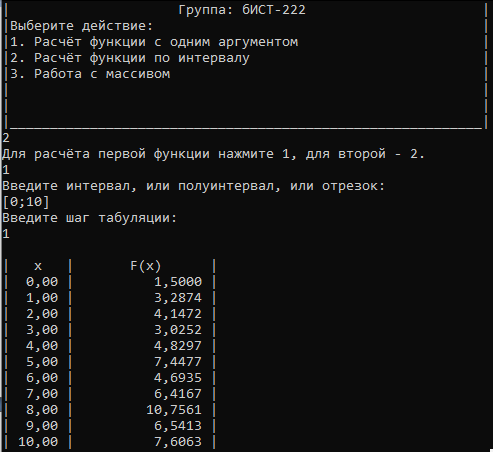


Рисунок 14 – Табуляция первой функции.

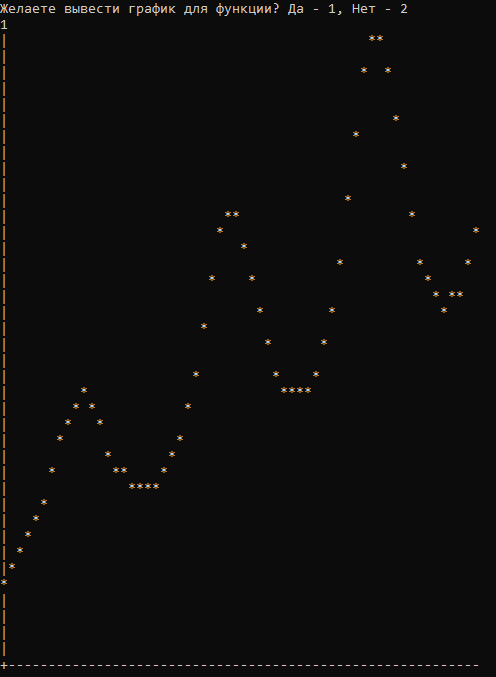


Рисунок 15 – График функции.

Сравнивая с графиком, данным в постановке задачи (Рисунок 1) мы видим, что у нас получился правильный кусок графика, расположенный в первой координатой четверти.

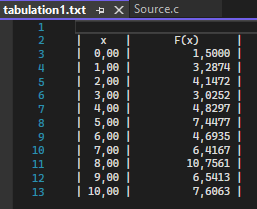


Рисунок 16 – Результат табуляции, выведенный в файл.

Производим те же действия со второй функцией. Результаты приведены на рисунках 17, 18 и 19.

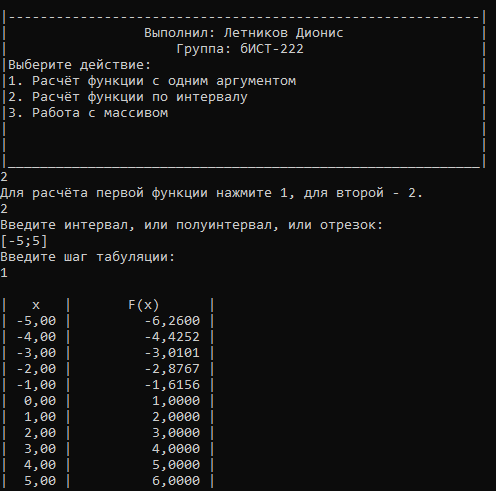


Рисунок 17 – Табуляция второй функции.

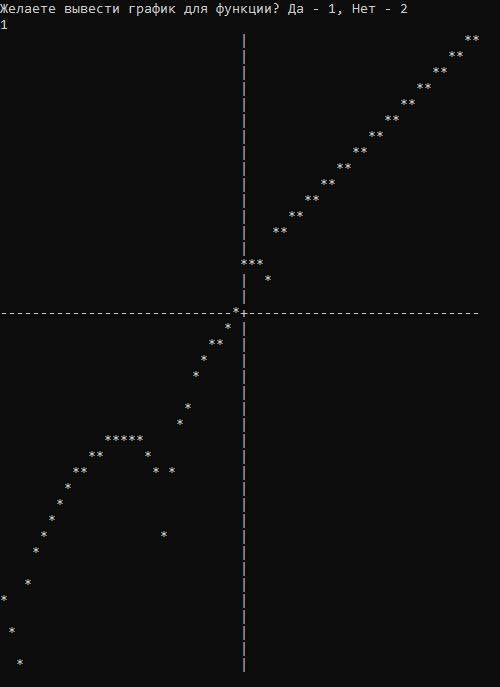


Рисунок 18 – График функции.

Сравнивая с графиком, данным в постановке задачи (Рисунок 2) мы видим, что у нас получился правильный график.

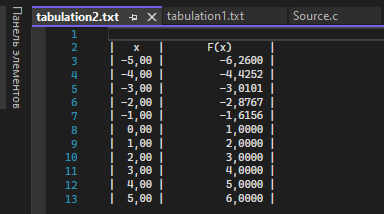


Рисунок 19 – Результат табуляции, выведенный в файл.

Перейдём к работе с двумерным массивом. Выбираем пункт 3, после чего нам необходимо задать количество строк и столбцов и сам массив, затем нам программа выводит количество отрицательных элементов в строках, содержащих хотя бы один нулевой элементи наличие седловой точки матрицы. Результат на рисунке 20.

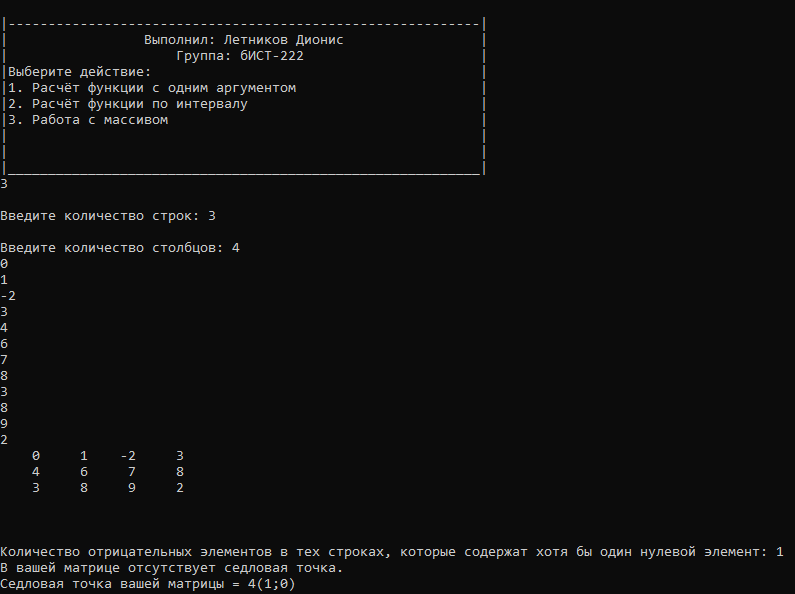
****

Рисунок 20 – Двумерный массив.

# ЗАКЛЮЧЕНИЕ

При выполнении курсового проекта, в среде Microsoft Visual Studio 2022, был разработан программный продукт, позволяющий производить табулирование функций, а также работать с двумерным массивом. Работа над данным курсовым проектом позволила лучше понять язык программирования С. Программа неоднократно тестировалась. В ходе работы были исправлены многие ошибки и недочеты.

В заключении отмечу, что в программе были созданы следующие функции: функция подсчёта с аргументом, функция табулирования, функция рисования графика, функции создания и вывода двумерного массива, функция поиска в этом массиве отрицательных элементов в строках, содержащих хотя бы один нулевой элементи поиск седловой точки матрицы.

# СПИСОК ИСРОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Основы программирования и алгоритмизации: практикум / Курипта О.В., Минакова О.В., Проскурин Д.К, Воронеж, 2015.
2. Процедурное и объектно-ориентированное программирование: учебное пособие / Павловская Т.А. C/C++, 2018.
3. [**Полный справочник по Cи**](http://www.google.com/url?q=http%3A%2F%2Fpaulfertser.info%2Fpolnyy_spravochnik_po_c_gerbert_shildt%2Fmain.htm&sa=D&sntz=1&usg=AFQjCNEorsXMX0sIK3X11rYkcFreadxLqg)**:** спарвочник / Шилдт (Herbert Schildt)
4. Язык Си руководство для начинающих: учебное пособие / M. УЭИТ, С. ПРАТА, Д. МАРТИН
5. Программирование на C и C++: электронный ресурс / http://www.c-cpp.ru/books (Дата обращения: 18.12.2022)
6. С&C++: электронный ресурс / http://natalia.appmat.ru/c&c++/index.php
7. Электронный учебник Си/Си++. От дилетанта до профессионала: электронный ресурс, учебное пособие / http://ermak.cs.nstu.ru/cprog/HTML/index.htm (Дата обращения: 18.12.2022)
8. Visual C ++ в Visual Studio 2015: электронный ресурс / https://docs.microsoft.com/en-us/previous-versions/60k1461a(v=vs.140)?redirectedfrom=MSDN (Дата обращения: 18.12.2022)
9. Справочник по языку C: электронный ресурс / https://docs.microsoft.com/en-us/cpp/c-language/c-language-reference?redirectedfrom=MSDN&view=msvc-160 (Дата обращения: 18.12.2022)
10. Описание языка программирования ANSI C: электронный ресурс / http://www.opennet.ru/docs/RUS/ansi-c/ (Дата обращения: 18.12.2022)

# ПРИЛОЖЕНИЕ A

#include<stdio.h>

#include<locale.h>

#include<math.h>

#include<io.h>

#include<stdlib.h>

//Указатель функции

typedef double (\*Tsch) (double);

struct sad {

double x;

double x1;

double x2;

double sh;

Tsch func;

};

typedef struct sad SAD;

//Прототипы функции

double sch\_y(double);

double sch\_v(double);

void t\_result(Tsch, double, double, double);

void plot(double x0, double x1, Tsch f);

void file(Tsch f, double xn, double xk, double h, int z);

int\* zip(int\* array, int n, int m);

int zip2(int\* array, int n, int m);

void saddle(int\* a, int n, int m);

void pain(int\* a, int n, int m);

void main()

{

setlocale(LC\_ALL, "RUS");

SAD das = {2, -5, 5, 0.5, sch\_y};

int r = -1, n, m;

char s1, s2, semicolon;

int\* a;

while (r!=0)

{

printf("\n|---------------------------------------------------------------------------------------|\n");

printf("| Выполнил: Летников Дионис |\n");

printf("| Группа: бИСТ-222 |\n");

printf("|Выберите действие: |\n");

printf("|1. Расчёт функции с одним аргументом |\n");

printf("|2. Расчёт функции по интервалу |\n");

printf("|3. Работа с массивом |\n");

printf("| |\n");

printf("| |\n");

printf("|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\n");

scanf("%d",&r);

switch (r)

{

case 1:

printf("Для расчёта первой функции нажмите 1, для второй - 2.\n");

scanf("%d", &r);

switch (r)

{

case 1:

printf("Введите аргумент x:");

scanf("%lf", &das.x);

if (das.x == -1) printf("Введено недопустимое значение аргумента");

else printf("Y(x) = %lf", das.func(das.x));

break;

case 2:

printf("Введите аргумент x:");

scanf("%lf", &das.x);

das.func = sch\_v;

printf("V(x) = %lf", das.func(das.x));

break;

default:

printf("Ошибка!");

break;

}

break;

case 2:

printf("Для расчёта первой функции нажмите 1, для второй - 2.\n");

scanf("%d", &r);

switch (r)

{

case 1:

printf("Введите интервал, или полуинтервал, или отрезок:\n");

scanf(" %c%lf%c%lf%c", &s1, &das.x1, &semicolon, &das.x2, &s2);

printf("Введите шаг табуляции:\n");

scanf("%lf", &das.sh);

if (s1 == '(') das.x1 += das.sh;

if (s2 == ')') das.x2 -= das.sh;

das.func = sch\_y;

t\_result(das.func, das.x1, das.x2, das.sh);

printf("\nЖелаете вывести график для функции? Да - 1, Нет - 2\n");

scanf("%d", &r);

switch (r)

{

case 1:

plot(das.x1, das.x2, das.func);

break;

case 2:

break;

default:

printf("Ошибка!");

break;

}

printf("\nЖелаете занести результаты табуляци в файл? Да - 1, Нет - 2\n");

scanf("%d", &r);

switch (r)

{

case 1:

file(das.func,das.x1,das.x2,das.sh,1);

break;

case 2:

break;

default:

printf("Ошибка!");

break;

}

break;

case 2:

printf("Введите интервал, или полуинтервал, или отрезок:\n");

scanf(" %c%lf%c%lf%c", &s1, &das.x1, &semicolon, &das.x2, &s2);

printf("Введите шаг табуляции:\n");

scanf("%lf", &das.sh);

if (s1 == '(') das.x1 += das.sh;

if (s2 == ')') das.x2 -= das.sh;

das.func = sch\_v;

t\_result(das.func, das.x1, das.x2, das.sh);

printf("\nЖелаете вывести график для функции? Да - 1, Нет - 2\n");

scanf("%d", &r);

switch (r)

{

case 1:

plot(das.x1, das.x2, das.func);

break;

case 2:

break;

default:

printf("Ошибка!");

break;

}

printf("\nЖелаете занести результаты табуляци в файл? Да - 1, Нет - 2\n");

scanf("%d", &r);

switch (r)

{

case 1:

file(das.func, das.x1, das.x2, das.sh, 2);

break;

case 2:

break;

default:

printf("Ошибка!");

break;

}

break;

default:

printf("Ошибка!");

break;

}

break;

case 0:

break;

case 3:

printf("\nВведите количество строк: ");

scanf("%d", &n);

printf("\nВведите количество столбцов: ");

scanf("%d", &m);

a = (int\*)malloc(n \* m \* sizeof(int));

a = zip(a, n, m);

zip2(a, n, m);

pain(a,n,m);

saddle(a,n,m);

break;

default:

printf("Ошибка!");

break;

}

}

}

double sch\_y(double x)

{

double y;

y = ((pow(sin(x), 2) \* cos(x)) / (x + 1)) + ((x + 3) / (1 + pow(cos(x), 2)));

return y;

}

double sch\_v(double x)

{

double y = 0;

if (x > 0.75)

{

y = fabs(x + 1);

}

if (x >= 0 && x <= 0.75)

{

y = 1 - pow(x, 5);

}

if (x < 0)

{

y = x + log(fabs(cos(x)));

}

return y;

}

void t\_result(Tsch f, double xn, double xk, double h)

{

char t = ' ';

printf("\n|%3cx%3c|%7cF(x)%6c|", t, t, t, t);

for (double x = xn; x <= xk; x += h)

{

printf("\n| %5.2lf | %15.4lf |", x, f(x));

}

}

void plot(double x0, double x1, Tsch f)

{

int SCREENW = 60, SCREENH = 40;//размеры поля вывода в символах

char screen[60][40];

double x, y[60];

double ymin = 0, ymax = 0;

double hx, hy;

int i, j;

int xz, yz;

hx = (x1 - x0) / (SCREENW - 1);

for (i = 0, x = x0; i < SCREENW; ++i, x += hx)

{

y[i] = f(x); //расчет значений функции для каждой точки поля вывода графика

if (y[i] < ymin) ymin = y[i];

if (y[i] > ymax) ymax = y[i];

}

hy = (ymax - ymin) / (SCREENH - 1);

yz = (int)floor(ymax / hy + 0.5);

xz = (int)floor((0. - x0) / hx + 0.5);

//построение осей и заполнение массива отображения пробелами

for (j = 0; j < SCREENH; ++j)

for (i = 0; i < SCREENW; ++i)

{

if (j == yz && i == xz) screen[i][j] = '+';

else if (j == yz) screen[i][j] = '-';

else if (i == xz) screen[i][j] = '|';

else screen[i][j] = ' ';

}

//определение положения значения функции на поле вывода

for (i = 0; i < SCREENW; ++i)

{

j = (int)floor((ymax - y[i]) / hy + 0.5);

screen[i][j] = '\*';

}

//печать массива символов

for (j = 0; j < SCREENH; ++j)

{

for (i = 0; i < SCREENW; ++i) putchar(screen[i][j]);

putchar('\n');

}

}

void file(Tsch f, double xn, double xk, double h, int z)

{

if (z == 1)

{

FILE\* result;

char name[20] = "tabulation1.txt";

if ((result = fopen(name, "w")) == NULL)

{

printf("Ошибка открытия файла для записи");

return 0;

}

char t = ' ';

fprintf(result, "\n|%3cx%3c|%7cF(x)%6c|", t, t, t, t);

for (double x = xn; x <= xk; x += h)

{

fprintf(result, "\n| %5.2lf | %15.4lf |", x, f(x));

}

fclose(result);

}

if (z == 2)

{

FILE\* result1;

char name2[20] = "tabulation2.txt";

if ((result1 = fopen(name2, "w")) == NULL)

{

printf("Ошибка открытия файла для записи");

return 0;

}

char t = ' ';

fprintf(result1, "\n|%3cx%3c|%7cF(x)%6c|", t, t, t, t);

for (double x = xn; x <= xk; x += h)

{

fprintf(result1, "\n| %5.2lf | %15.4lf |", x, f(x));

}

fclose(result1);

}

}

int\* zip(int\* array, int n, int m)

{

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

{

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

{

scanf("%d", (array + i \* m + j));

}

}

return array;

}

int zip2(int\* array, int n, int m)

{

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

{

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

{

printf("%5d ", \*(array + i \* m + j));

}

printf("\n");

}

return 0;

}

void saddle(int\* a, int n, int m)

{

int i, j, k, f, min, minpos;

for (i = 0; i < n; i++)

{

min = \*(a + i \* m + 0);

minpos = 0;

for (j = 0; j < m; j++)

{

if (\*(a + i \* m + j) < min)

{

min = \*(a + i \* m + j);

minpos = j;

}

}

f = 1;

for (k = 0; k < n; k++)

{

if (min < \*(a + k \* m + minpos)) // Если минимальное значение текущей строки меньше значения в столбце, то есть число не является седловой точкой

{

f = 0;

break;

}

}

if (f == 1)

{

printf("\nСедловая точка вашей матрицы = %d(%d;%d)", min, i, minpos);

}

else

{

printf("\nВ вашей матрице отсутствует седловая точка.");

}

}

}

void pain(int\* a, int n, int m)

{

int counter = 0;

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

{

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

{

if (\*(a + i \* m + j) == 0)

{

for (j = 0; j < m; j++)

{

if (\*(a + i \* m + j) < 0)

{

counter++;

}

}

}

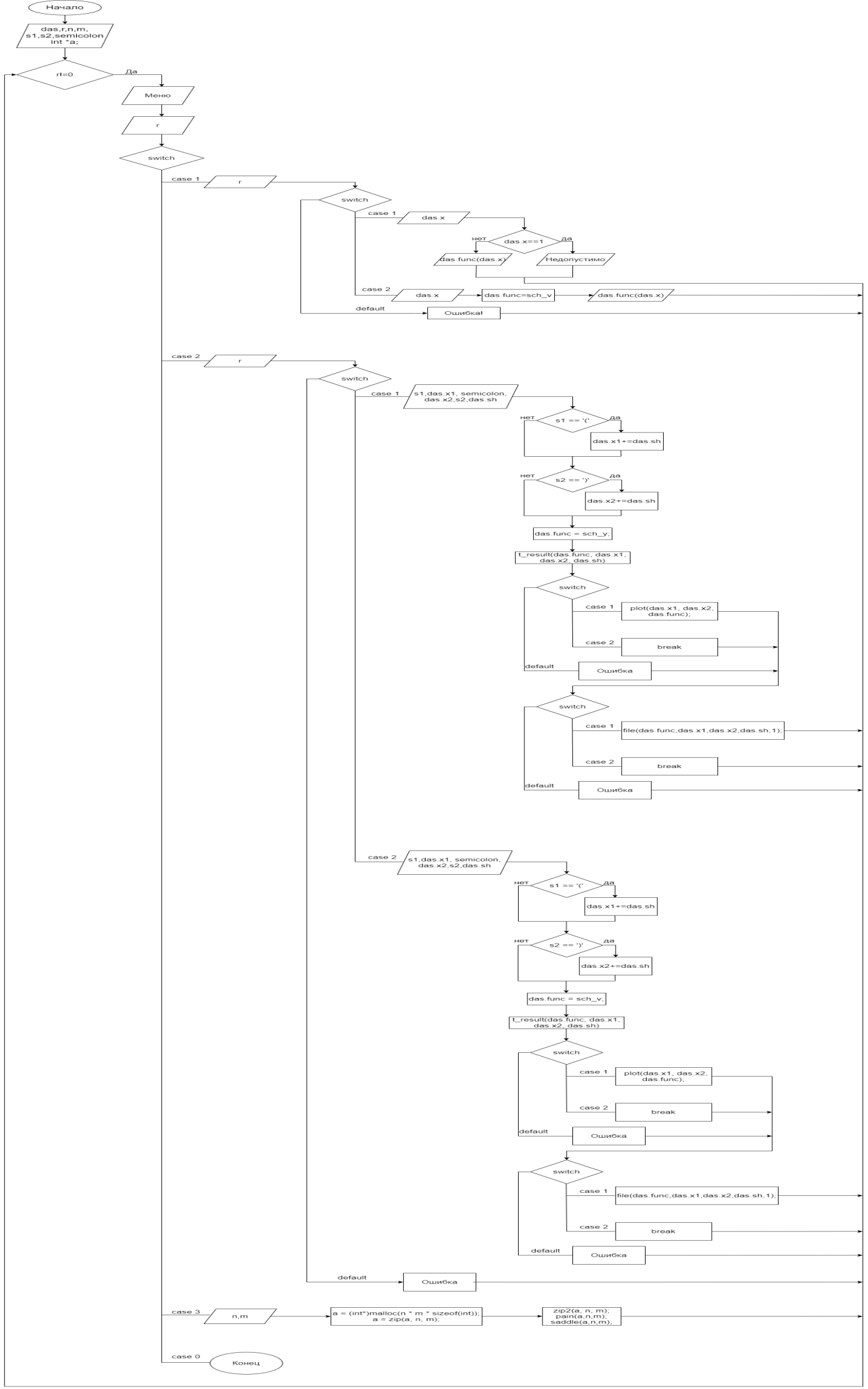
}

printf("\n");

}

printf("Количество отрицательных элементов в тех строках, которые содержат хотя бы один нулевой элемент: %d", counter);

}

****