Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Вятский государственный университет»

Колледж ВятГУ

**ОТЧЕТ**

**ПО ДОМАШНЕЙ КОНТРОЛЬНОЙ РАБОТЕ №4**

**«РАБОТА В ГРАФИЧЕСКОМ РЕЖИМЕ»**

**ПО ДИСЦИПЛИНЕ «ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ»**

Выполнил: студент учебной группы

ИСПк-203-52-00

Суровцев Александр Сергеевич

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2023

Содержание

1. Цель лабораторной работы
2. Формулировка задания (с вариантом)
3. Описание алгоритма
4. Схема алгоритма с комментариями
5. Код программы
6. Результат выполнения программы
7. Вывод
8. **Цель работы**: освоить принципы работы в графическом режиме, получить базовые навыки взаимодействия с графическими примитивами.
9. **Формулировка задания(Вариант:19)**
10. Дополнить программу, реализованную в ходе предыдущей лабораторной работы режимом визуализации.
11. Предусмотреть возможность вывода кривой, ограничивающей фигуру на координатную плоскость.
12. Реализовать следующие возможности и элементы: масштабирование графика, подписи на осях, вывод информации о задании.
13. Реализовать не менее двух возможностей из представленных: независимое масштабирование по осям, штриховка вычисляемой площади, визуализация численного расчёта интеграла.

**3.Описание алгоритма**

Данный алгоритм состоит из двух процедур. Процедура inf предназначена для выведения информации в модуль clrt. Вторая процедура graph, предназначена для графического отображения графика и выделения площади криволинейной трапеции. Так же задействована процедура left\_method, которая была выполнена ранее. В итоге создано кейс-меню, из которого можно управлять всеми действиями в программе.

**4. Схема алгоритма с комментариями**

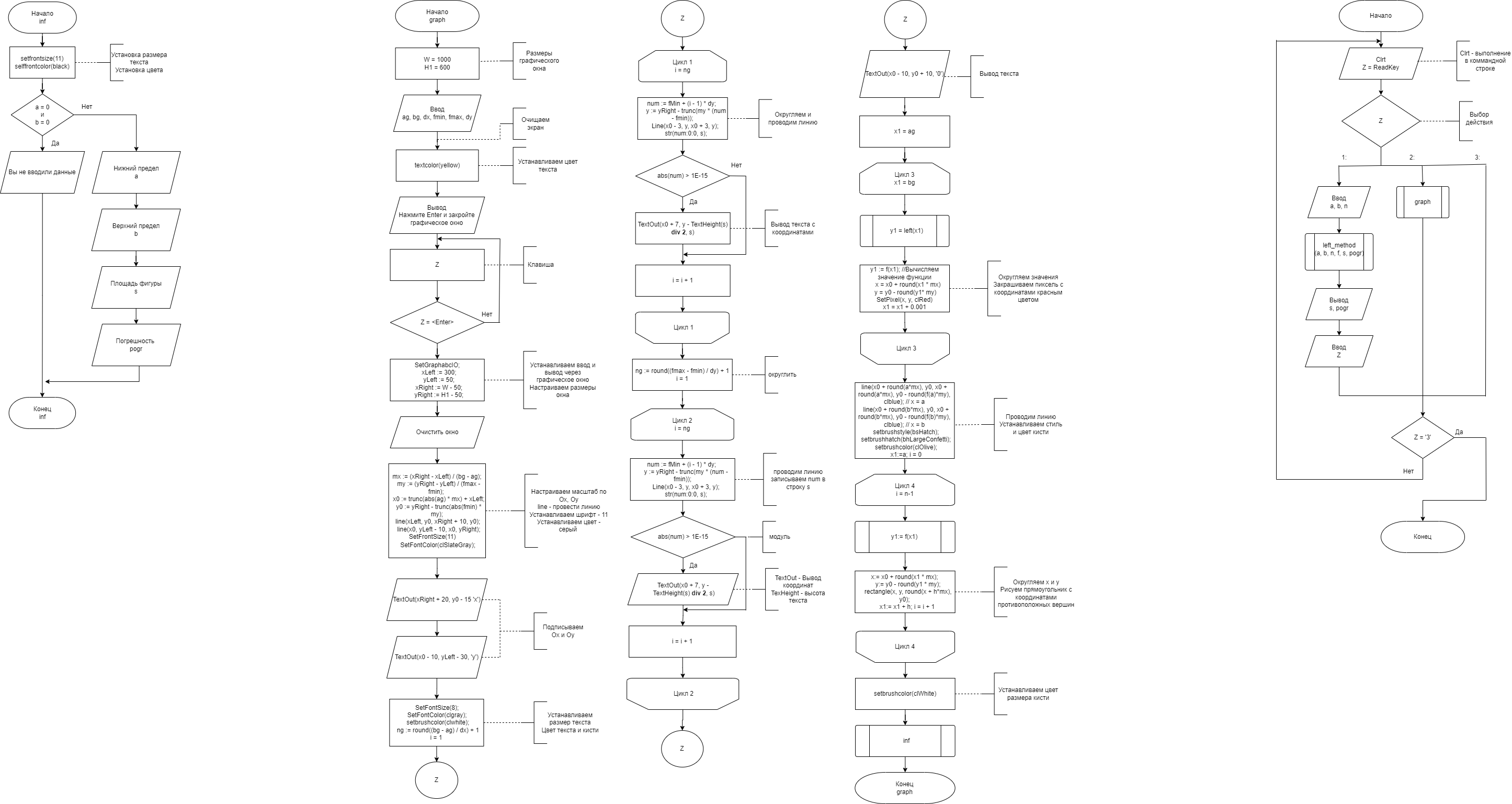


Рисунок 1 – Схема главного алгоритма

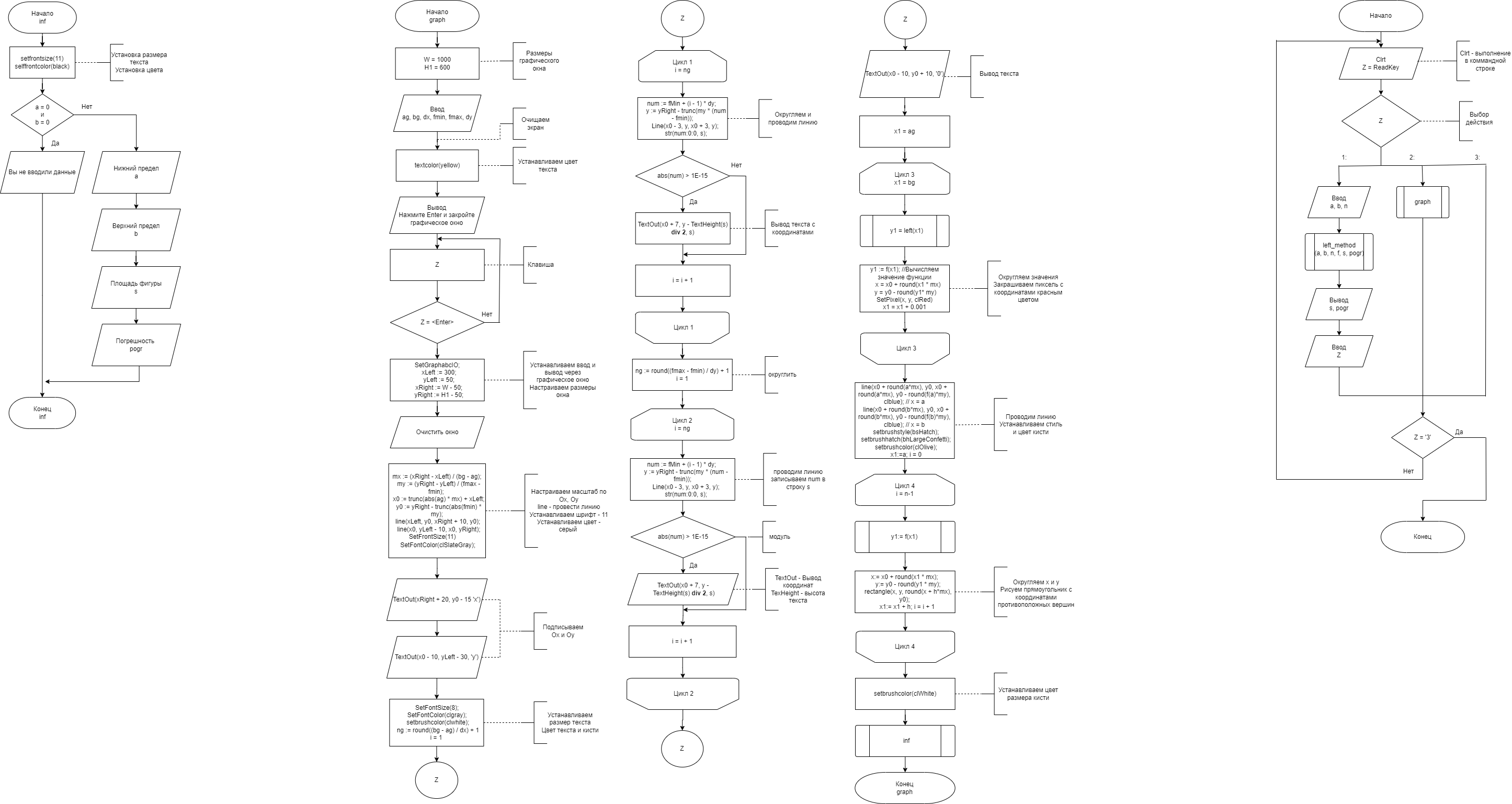


Рисунок 2 – Схема алгоритма graph

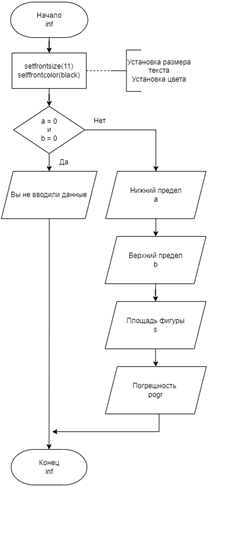


Рисунок 3 – Схема алгоритма inf

**5. Код программы:**

**uses** Crt, GraphABC;

**type**

bam = **function**(x: Real): Real;

**var**

a, b, dx, dy, h: Real;

n: Integer;

s, pogr: Real;

Z: char;

x, y: integer;

color : Byte;

**function** f(x: real): real;

**begin**

f:=(x\*\*3)+(x\*\*2)+(-2)\*x+17;

**end**;

**procedure** left\_method(a, b: Real; n: Integer; func: bam; **var** s, pogr:real);

**var**

x: Real;

i: Integer;

fa, fb: real;

**begin**

h:= (b-a)/n;

x:= a;

**for** i:=0 **to** n-1 **do begin**

s:=s+f(x);

x:=x+h;

**end**;

s:=s\*h;

fa:=(((a\*a\*a\*a)/4)+((a\*a\*a)/3)-(a\*a)-(17\*a));

fb:=(((b\*b\*b\*b)/4)+((b\*b\*b)/3)-(b\*b)-(17\*b));

pogr:=(fb-fa) - s;

**end**;

**procedure** inf;

**begin**

setfontsize(11);

setfontcolor(clblack);

**if not** ((a = 0) **and** (b = 0)) **then**

**begin**

writeln('Нижний предел: ', a);

writeln('Верхний предел: ', b);

writeln('Площадь заштрихованной фигуры: ', s:0:2);

writeln('Погрешность: ', pogr);

**end**

**else** writeln('Вы не вводили данные');

**end**;

**procedure** graph;

**const**

W = 1000; H1 = 600;

**var**

x0, y0, x, y, xLeft, yLeft, xRight, yRight, ng: integer;

ag, bg, fmin, fmax, x1, y1, mx, my, num: real;

i: byte;

s: string;

**begin**

SetConsoleIO;

textcolor(11);

clrscr;

Writeln('Введите нижнюю границу системы координат по Х: ');

read(ag);

Writeln('Введите верхнюю границу системы координат по Х: ');

read(bg);

Writeln('Введите единичный отрезок по Х: ');

read(dx);

Writeln('Введите нижнюю границу системы координат по Y: ');

read(fmin);

Writeln('Введите верхнюю границу системы координат по Y: ');

read(fmax);

Writeln('Введите единичный отрезок по Y: ');

read(dy);

writeln;

clrscr;

textcolor(yellow);

Writeln('Нажмите [Enter] и откройте графическое окно');

**repeat**

Z := readkey;

**until** Z = #13;

SetGraphabcIO;

SetWindowSize(W, H1);

xLeft := 300;

yLeft := 50;

xRight := W - 50;

yRight := H1 - 50;

clearwindow;

mx := (xRight - xLeft) / (bg - ag);

my := (yRight - yLeft) / (fmax - fmin);

x0 := trunc(abs(ag) \* mx) + xLeft;

y0 := yRight - trunc(abs(fmin) \* my);

line(xLeft, y0, xRight + 10, y0);

line(x0, yLeft - 10, x0, yRight);

SetFontSize(11);

SetFontColor(clSlateGray);

TextOut(xRight + 20, y0 - 15, 'х');

TextOut(x0 - 10, yLeft - 30, 'у');

SetFontSize(8);

SetFontColor(clgray);

setbrushcolor(clwhite);

ng := round((bg - ag) / dx) + 1;

**for** i := 1 **to** ng **do**

**begin**

num := ag + (i - 1) \* dx;

x := xLeft + trunc(mx \* (num - ag));

Line(x, y0 - 3, x, y0 + 3);

str(Num:0:1, s);

**if** abs(num) > 1E-15 **then**

TextOut(x - TextWidth(s) **div** 2, y0 + 10, s)

**end**;

ng := round((fmax - fmin) / dy) + 1;

**for** i := 1 **to** ng **do**

**begin**

num := fMin + (i - 1) \* dy;

y := yRight - trunc(my \* (num - fmin));

Line(x0 - 3, y, x0 + 3, y);

str(num:0:0, s);

**if** abs(num) > 1E-15 **then**

TextOut(x0 + 7, y - TextHeight(s) **div** 2, s)

**end**;

TextOut(x0 - 10, y0 + 10, '0');

x1 := ag;

**while** x1 <= bg **do**

**begin**

y1 := f(x1);

x := x0 + round(x1 \* mx);

y := y0 - round(y1 \* my);

SetPixel(x, y, clred);

x1 := x1 + 0.001 //Увеличиваем абсциссу

**end**;

line(x0 + round(a\*mx), y0, x0 + round(a\*mx), y0 - round(f(a)\*my), clblue); // х = а

line(x0 + round(b\*mx), y0, x0 + round(b\*mx), y0 - round(f(b)\*my), clblue); // х = b

setbrushstyle(bsHatch);

setbrushhatch(bhLargeConfetti);

setbrushcolor(clTeal);

x1:=a;

**for** i:=0 **to** n-1 **do**

**begin**

y1:= f(x1);

x:= x0 + round(x1 \* mx);

y:= y0 - round(y1 \* my);

rectangle(x, y, round(x + h\*mx), y0);

x1:= x1 + h;

**end**;

setbrushcolor(clWhite);

inf;

**end**;

**begin**

**repeat**

SetConsoleIO;

ClrScr;

textcolor(Blue);

writeln('1. Вычисление площади фигуры, ограниченной кривой');

writeln('2. График');

writeln('3. Выход');

write('Выберите действие: ');

Z := ReadKey;

**case** Z **of**

'1':

**begin**

ClrScr;

Textcolor(Green);

writeln('Введите границы интегрирования: ');

readln(a, b);

Textcolor(Green);

writeln('Введите количество делений: ');

readln(n);

Left\_method(a, b, n, f, s, pogr);

Textcolor(LightGreen);

writeln('Площадь фигуры: ', s);

Textcolor(LightGreen);

writeln('Погрешность: ', pogr);

readln;

**end**;

'2': graph;

'3': halt;

**end**;

**until** Z = '3';

**end**.

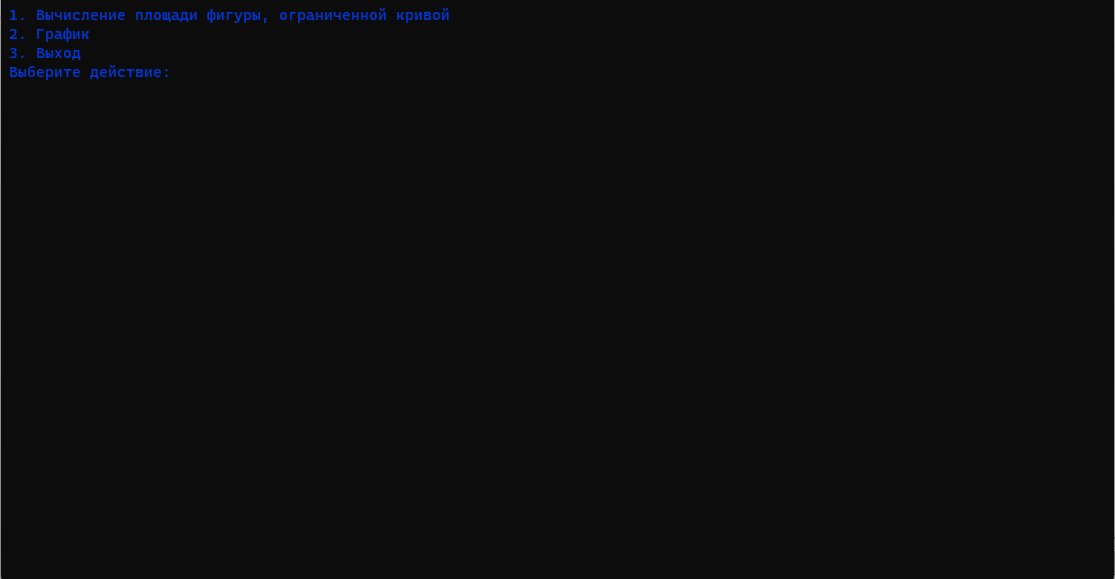


Рисунок 4 - Результат выполнения программы

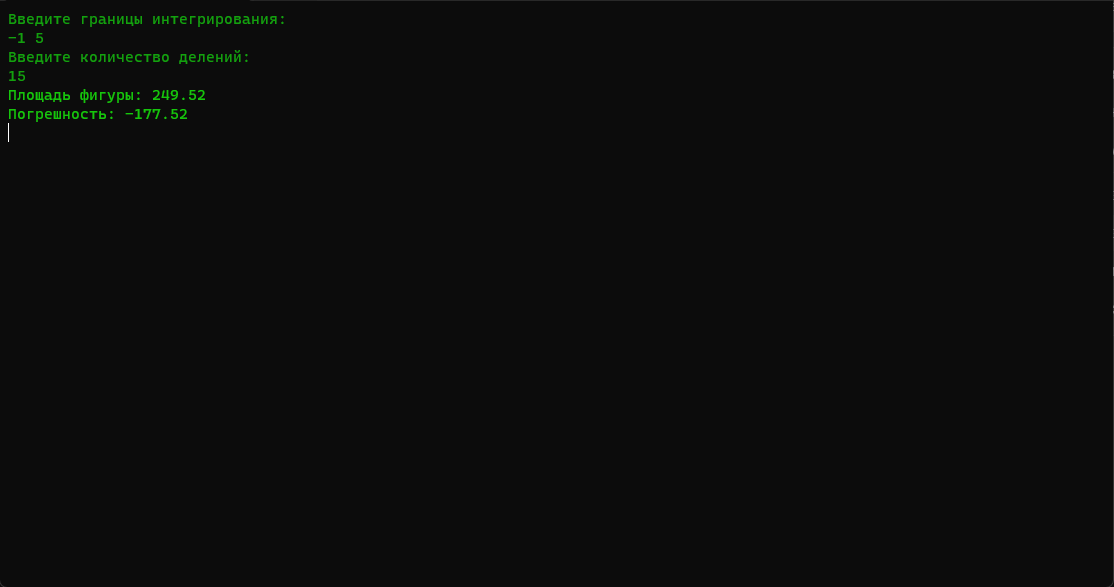


Рисунок 10 – Результат выполнения программы

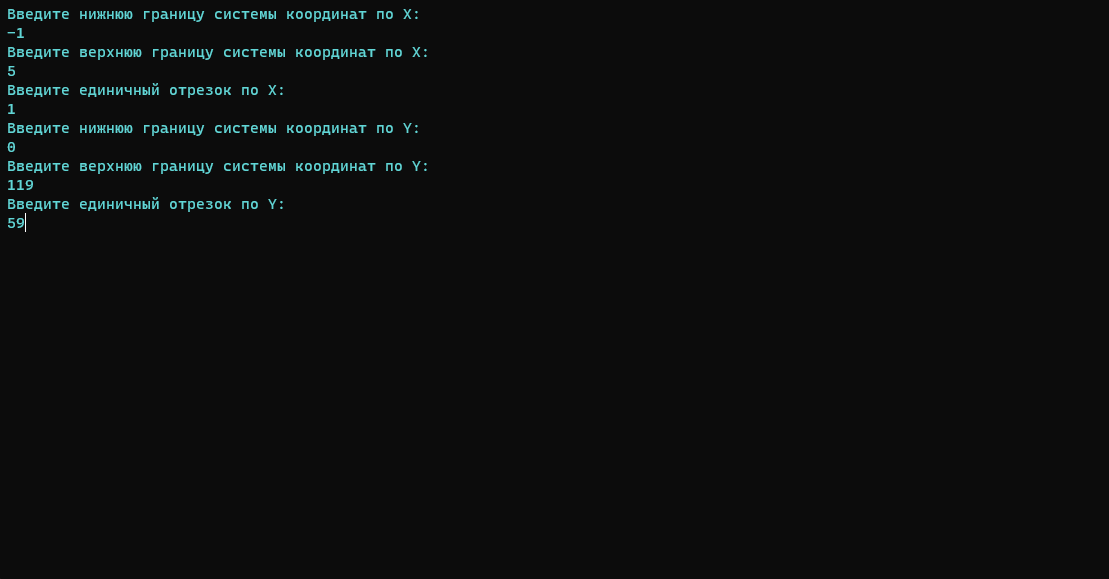


Рисунок 11 – Результат выполнения процедуры inf

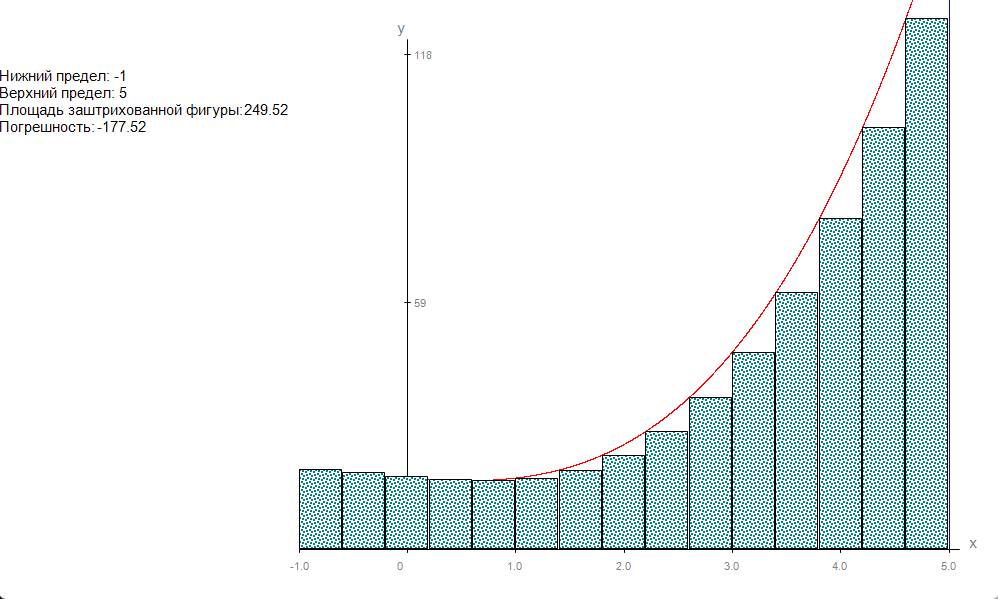


Рисунок 12 – Результат выполнения процедуры graph

**7. Вывод**

Данная работа прошла успешно. В работе было выполнено графическое отображение выполнение высчитывания площади криволинейной трапеции в системе координат.

В работе была написана программа по высчитыванию площади криволинейной трапеции по методу левых прямоугольников, графическое отображение на координатной плоскости, а также управление всеми процессами через кейс меню.

Для создания схемы алгоритмов мы использовали приложение Draw.io. Draw.io — это бесплатный онлайн-сервис, который помогает создавать схемы алгоритмов, прототипы, инфографику и диаграммы любого вида. Чаще всего его используют именно для построения диаграмм, поэтому недавно сервис переименовали в Diagrams.net. Но старое название по-прежнему в ходу. Сервисом пользуются разные IT-специалисты — от бизнес-аналитиков до разработчиков и дизайнеров.