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

Генинг Богдана Андреевна

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2023

1. **Цель работы:** освоить принципы работы в графическом режиме; получить базовые навыки взаимодействия с графическими примитивами.
2. **Формулировка задания (Вариант:7)**
3. 1)Дополнить программу, реализованную в ходе предыдущей лабораторной работы, режимом визуализации.

2)Предусмотреть возможность вывода кривой, ограничивающей фигуру, на координатную плоскость.

3)Реализовать следующие возможности и элементы: масштабирование графика, подписи на осях, вывод информации о задании

4)Реализовать не менее двух возможностей из представленных независимое масштабирование по осям, штриховка вычисляемой площади, визуализация численного расчета интеграла.

1. **Описание алгоритма**

Алгоритм начинается с установки параметров для консольного ввода-вывода, очистки экрана и вывода текстовых сообщений, чтобы пользователь ввел значения для системы координат и отрезков по оси X и Y.

Затем пользователю предлагается нажать Enter, чтобы открыть графическое окно. Ожидается ввод от пользователя до тех пор, пока не будет нажата клавиша Enter.

Далее происходит настройка параметров графического окна, установка размеров окна и координатных осей. Затем осуществляется очистка окна.

Вычисляются значения масштаба по осям X и Y, а также координаты нулевой точки. Затем рисуются оси координат и маркировки засечек на осях.

Далее происходит отрисовка графика функции на основе заданных границ и единичных отрезков по осям X и Y.

Наконец, на графике отображаются границы функции и прямоугольники, используемые для вычисления площади. Также выводится информация о графике и заканчивается алгоритм.

1. **Схема алгоритма**

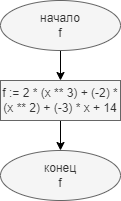
****

Рис 1. Схема алгоритма

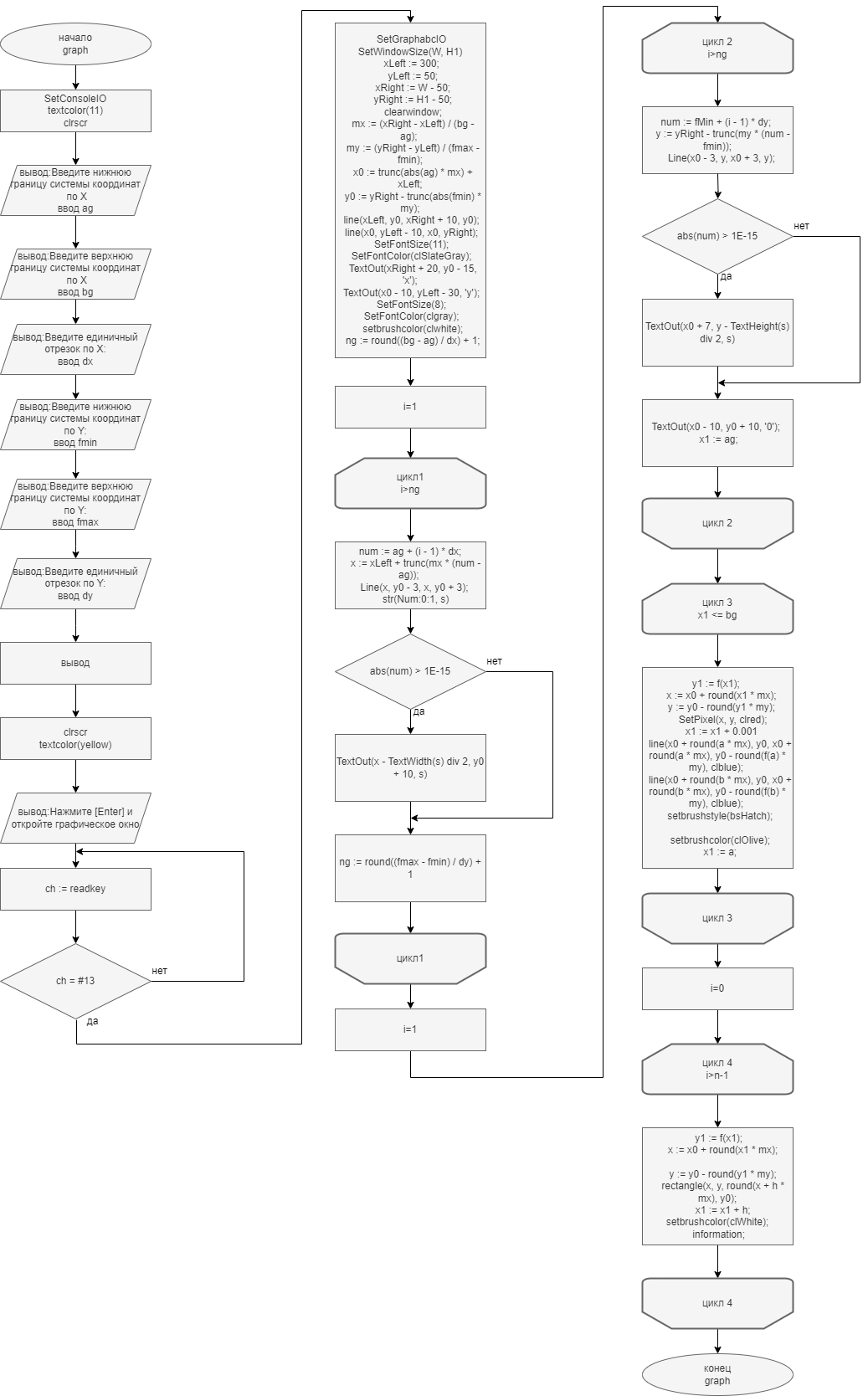


Рис 2. Схема алгоритма

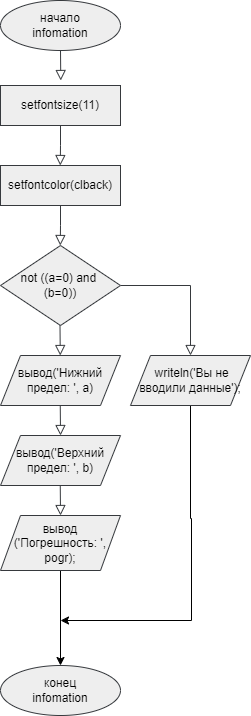
****

Рис 3. Схема алгоритма

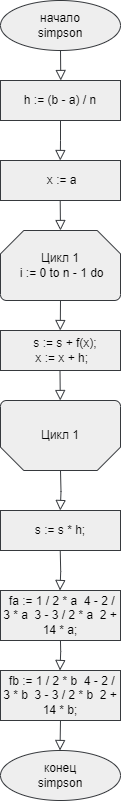
****

Рис 4. Схема алгоритма

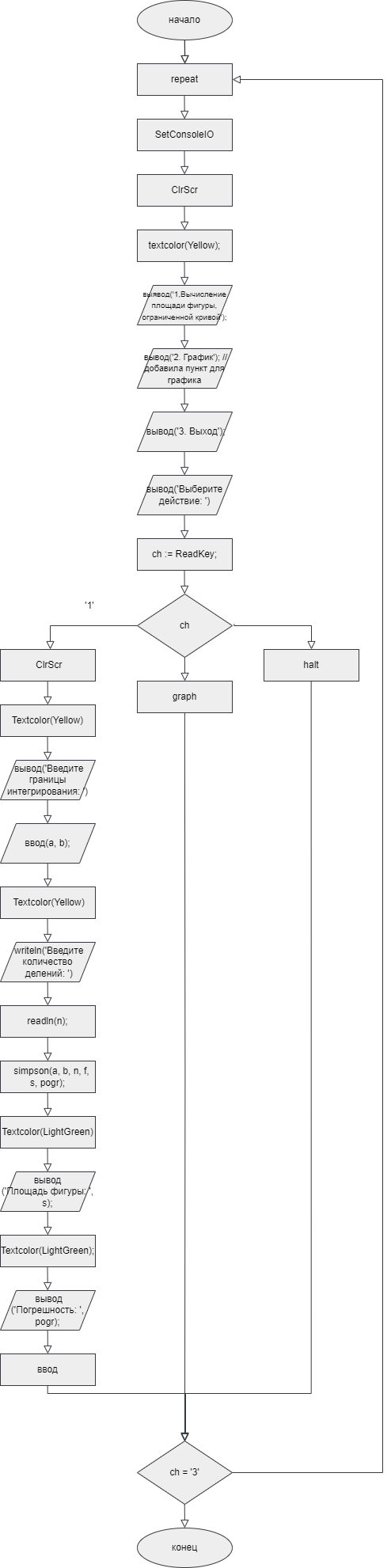
****

Рис 5. Схема алгоритма

1. **Код программы:**

**uses** Crt, GraphABC;

**type**

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

**var**

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

n: Integer;

s, pogr: Real;

ch: char;

x, y: integer;

color: Byte;

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

**begin**

f := 2 \* (x \*\* 3) + (-2) \* (x \*\* 2) + (-3) \* x + 14;

**end**;

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

**var**

x: Real;

i: Integer;

fa, fb: real; //Знаечния первообразной

**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 := 1 / 2 \* a \*\* 4 - 2 / 3 \* a \*\* 3 - 3 / 2 \* a \*\* 2 + 14 \* a; //нахождение значения первой первообразной через производную

fb := 1 / 2 \* b \*\* 4 - 2 / 3 \* b \*\* 3 - 3 / 2 \* b \*\* 2 + 14 \* b; //второй

pogr := (fb - fa) - s; //нахождение пгрешности

**end**;

**procedure** information;

**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;//Размеры графического окна

**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**

ch := readkey;

**until** ch = #13;

SetGraphabcIO;

SetWindowSize(W, H1); //Устанавливаем размеры графического окна

xLeft := 300;

yLeft := 50;

xRight := W - 50;

yRight := H1 - 50;

clearwindow;

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

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

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

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

line(xLeft, y0, xRight + 10, y0); //ось ОХ

line(x0, yLeft - 10, x0, yRight); //ось ОY

SetFontSize(11); //Размер шрифта

SetFontColor(clSlateGray); //Цвет шрифта

TextOut(xRight + 20, y0 - 15, 'х'); //Подписываем ось OX

TextOut(x0 - 10, yLeft - 30, 'у'); //Подписываем ось OY

SetFontSize(8); //Размер шрифта

SetFontColor(clgray); //Цвет шрифта

setbrushcolor(clwhite);

ng := round((bg - ag) / dx) + 1; //количество засечек по ОХ

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

**begin**

num := ag + (i - 1) \* dx; //Координата на оси ОХ

x := xLeft + trunc(mx \* (num - ag)); //Координата num в окне

Line(x, y0 - 3, x, y0 + 3); //рисуем засечки на оси OX

str(Num:0:1, s);

**if** abs(num) > 1E-15 **then** //Исключаем 0 на оси OX

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

**end**;

ng := round((fmax - fmin) / dy) + 1; //количество засечек по ОY

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

**begin**

num := fMin + (i - 1) \* dy; //Координата на оси ОY

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

Line(x0 - 3, y, x0 + 3, y); //рисуем засечки на оси Oy

str(num:0:0, s);

**if** abs(num) > 1E-15 **then** //Исключаем 0 на оси OY

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

**end**;

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

x1 := ag; //Начальное значение аргумента

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

**begin**

y1 := f(x1); //Вычисляем значение функции

x := x0 + round(x1 \* mx); //Координата Х в графическом окне

y := y0 - round(y1 \* my); //Координата Y в графическом окне

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);

setbrushcolor(clOlive);

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);

information;

**end**;

**begin**

**repeat**

SetConsoleIO;

ClrScr;

textcolor(Yellow);

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

writeln('2. График'); //добавила пункт для графика

writeln('3. Выход');

write('Выберите действие: ');

ch := ReadKey;

**case** ch **of**

'1':

**begin**

ClrScr;

Textcolor(Yellow);

writeln('Введите границы интегрирования: ');

readln(a, b);

Textcolor(Yellow);

writeln('Введите количество делений: ');

readln(n);

simpson(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** ch = '3';

**end**.

1. **Реультат выполнения программы**

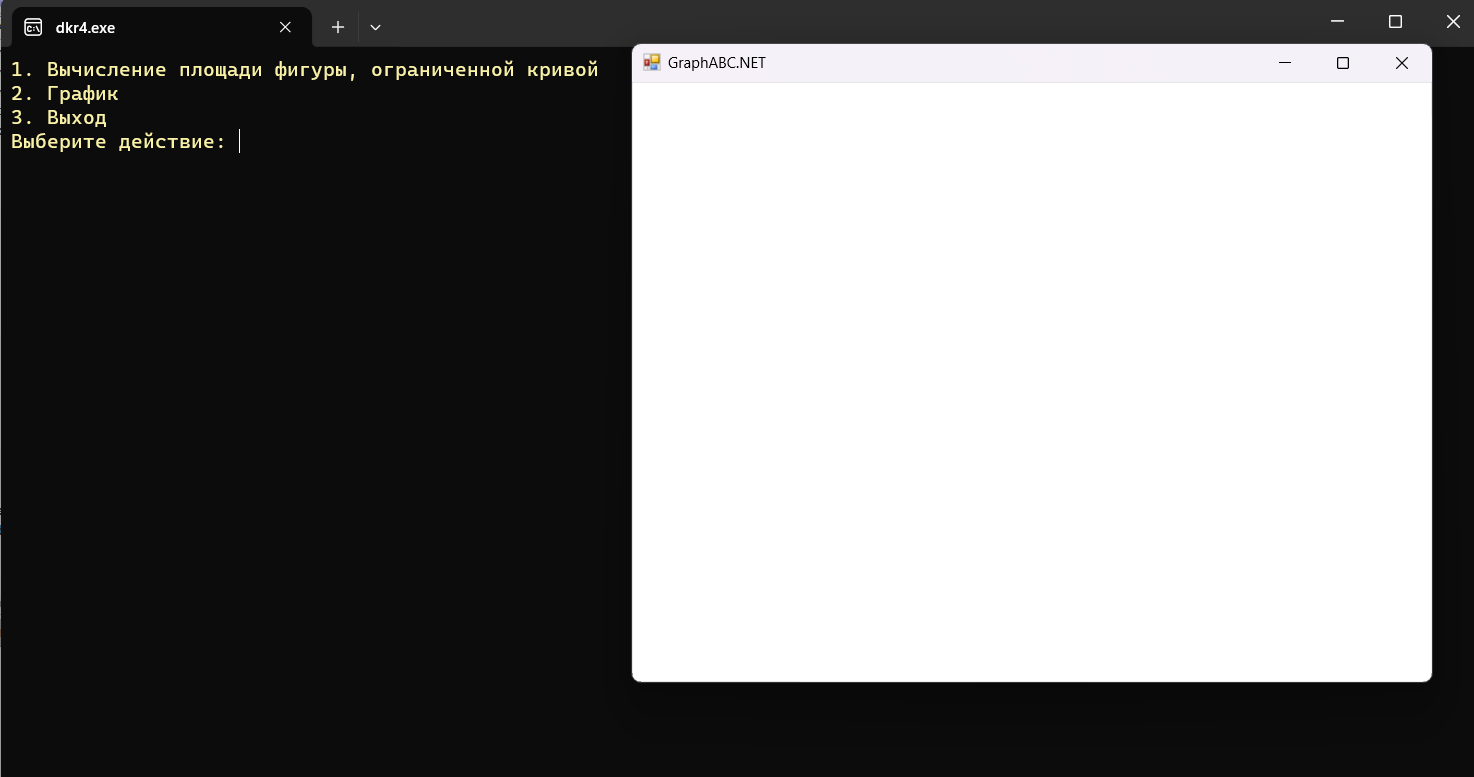


Рис 5. Результат выполнения программы

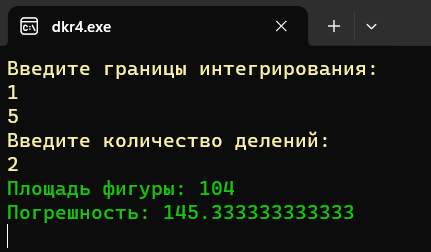


Рис 6. Результат выполнения программы

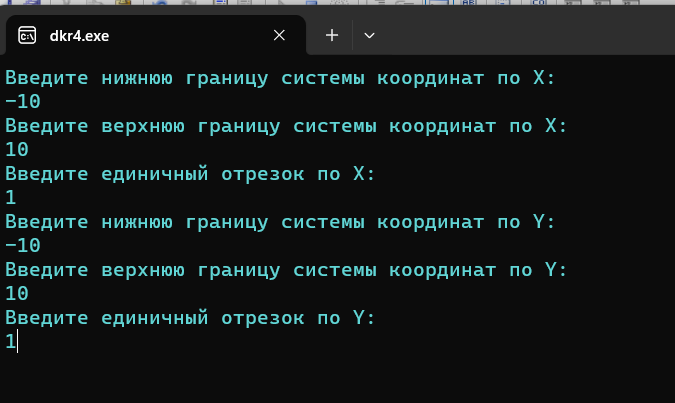


Рис 7. Результат выполнения программы

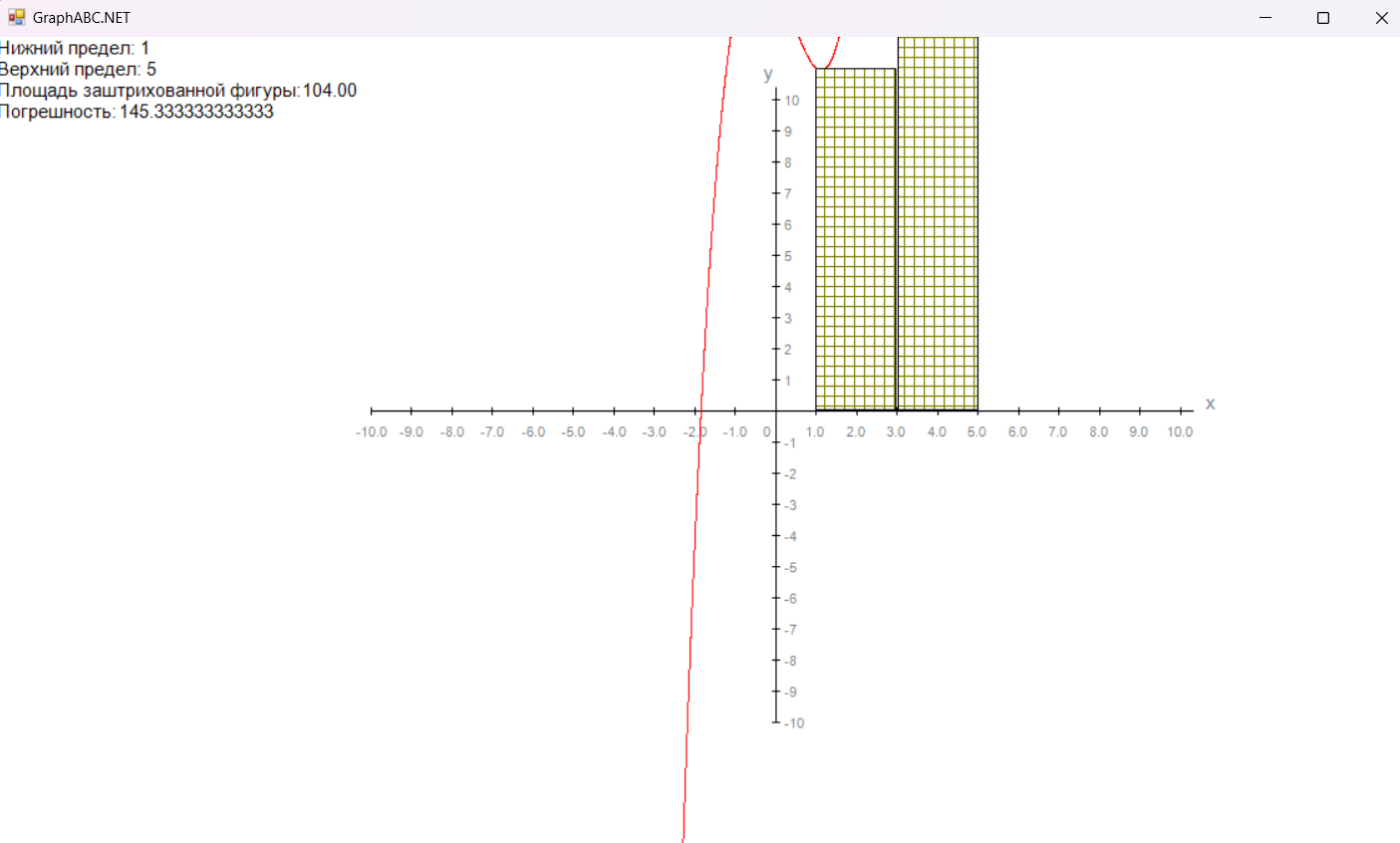


Рис 8. Результат выполнения программы

1. **Вывод**

В ходе данной домашней контрольной работы мы освоили принципы работы в графическом режиме; получили базовые навыки взаимодействия с графическими примитивами.

Мы успешно справились с выполнением задания и были заинтересованы и увлечены процессом изучения языка программирования Pascal. Понимание использования циклов и условий было закреплено, и мы достигли всех поставленных целей. Это дало нам уверенность в наших навыках программирования и мотивацию для дальнейшего развития.