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

Яковлев Александр Станиславович

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2024

**Цель работы:** освоить принципы роботы с графическим режимом; получить базовые навыки взаимодействия с графическими примитивами.

**Формулировка задания:**

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 интегралов.

**Описание алгоритма:** за основу кода была взята программа из ДКР №3, поэтому прежние опции и функции до сих пор работают. Из нового можно выделить встроенную библиотеку GraphABC, которая помогает визуализировать работу функции с помощью встроенных функций.

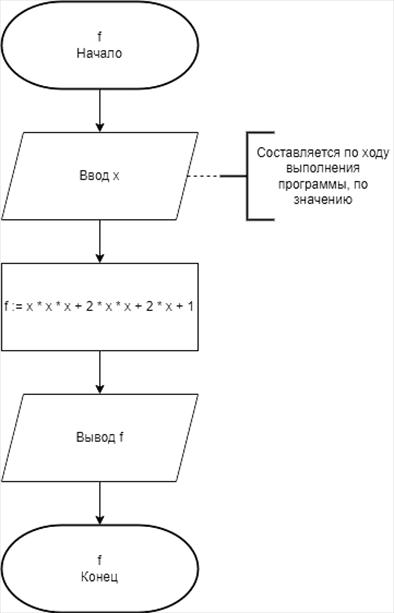


Рисунок 1 – схема алгоритма для функции f

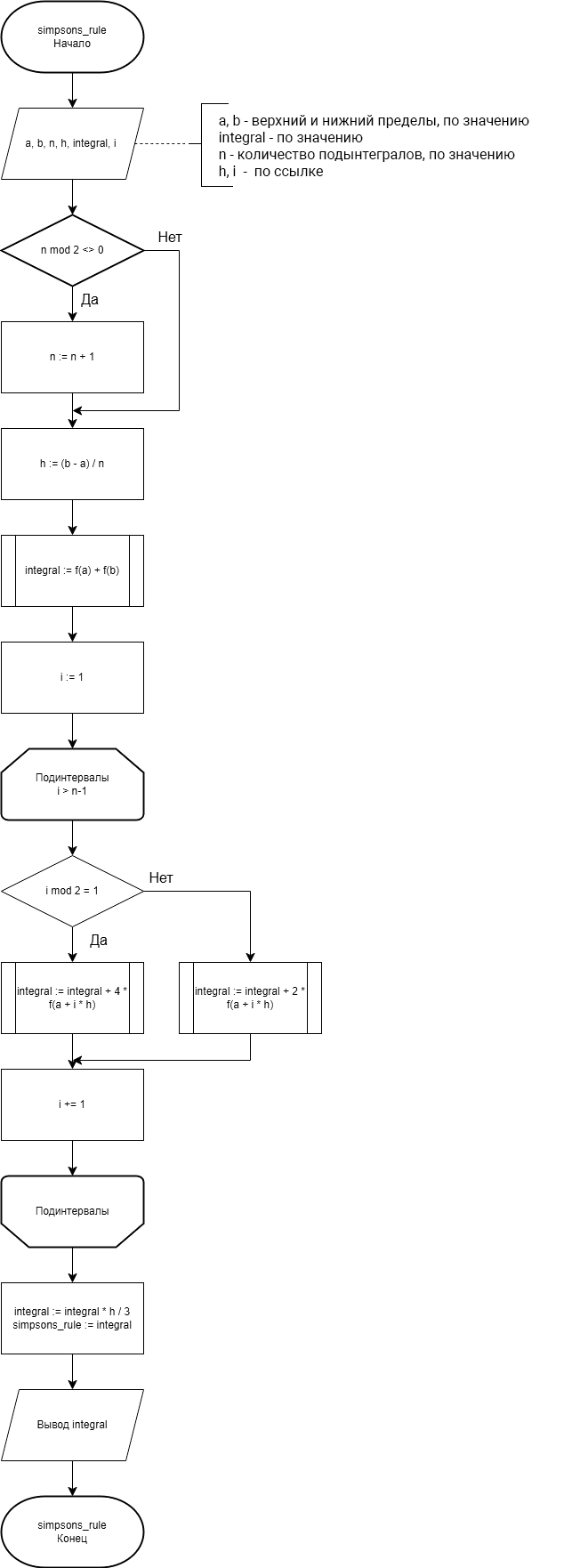


Рисунок 2 – схема алгоритмов для функции simpson\_rule

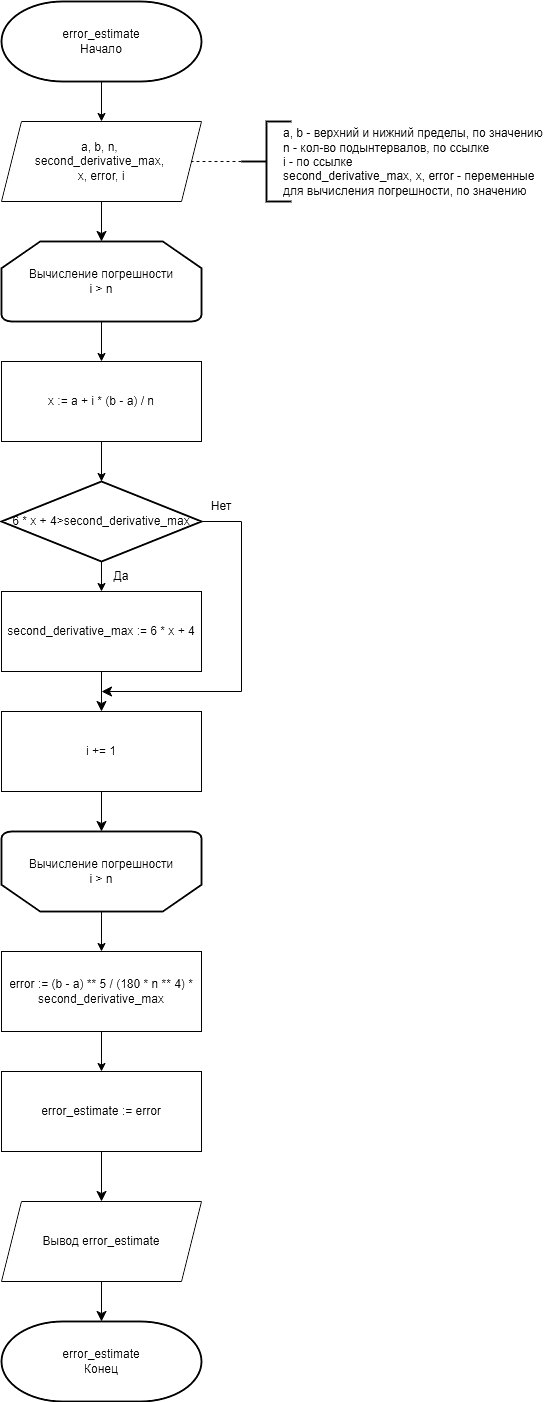


Рисунок 3 – схема алгоритма для функции error\_estimate

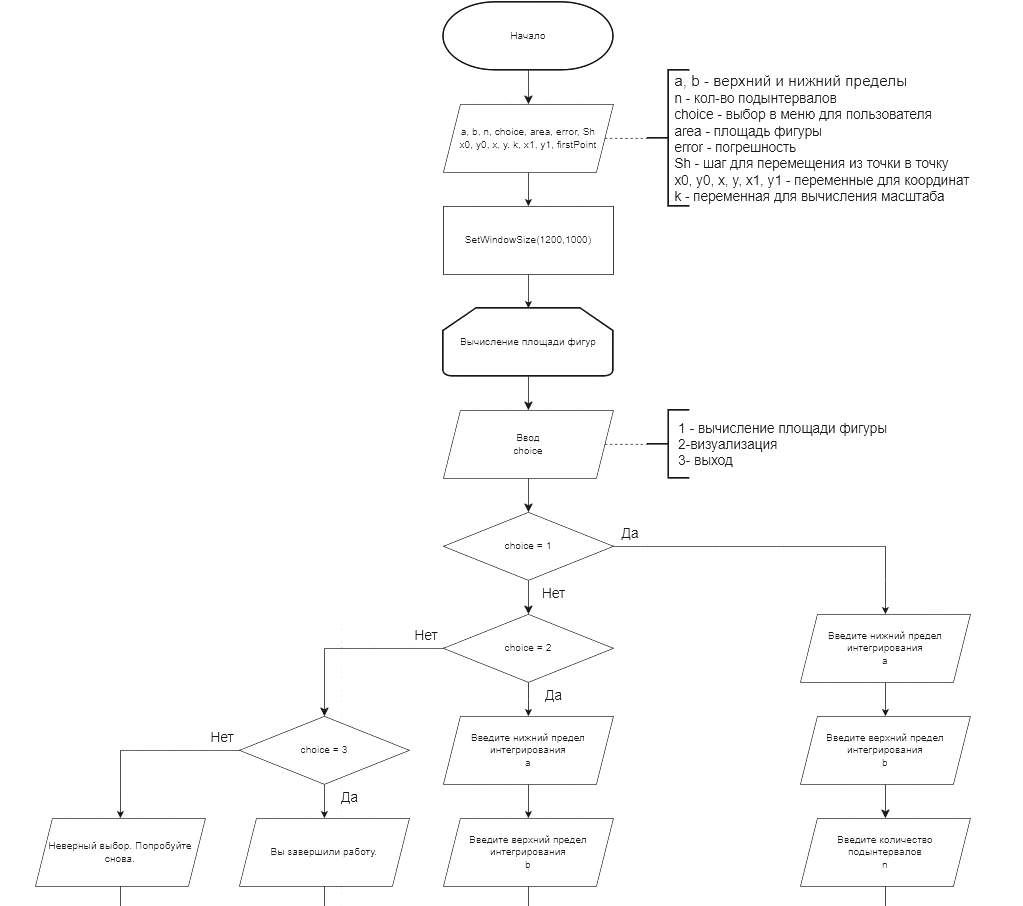


Рисунок 4.1 – схема алгоритмов основной программы

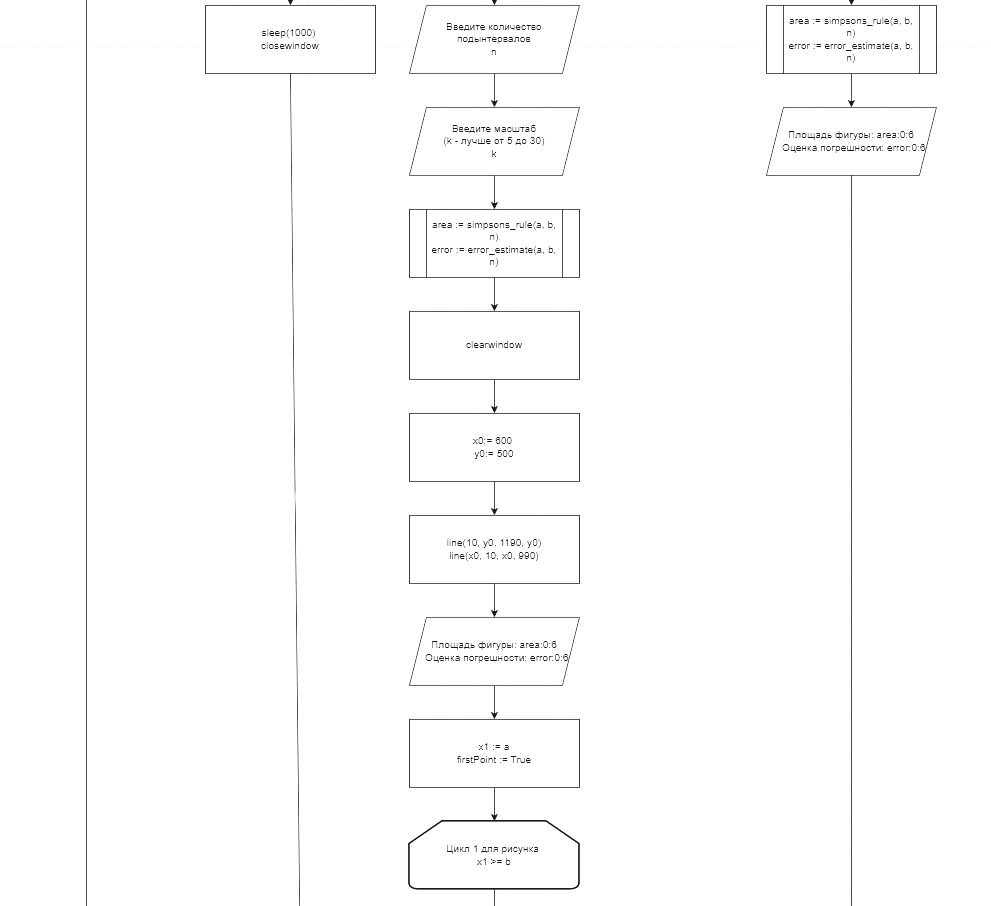


Рисунок 4.2 – схема алгоритмов основной программы

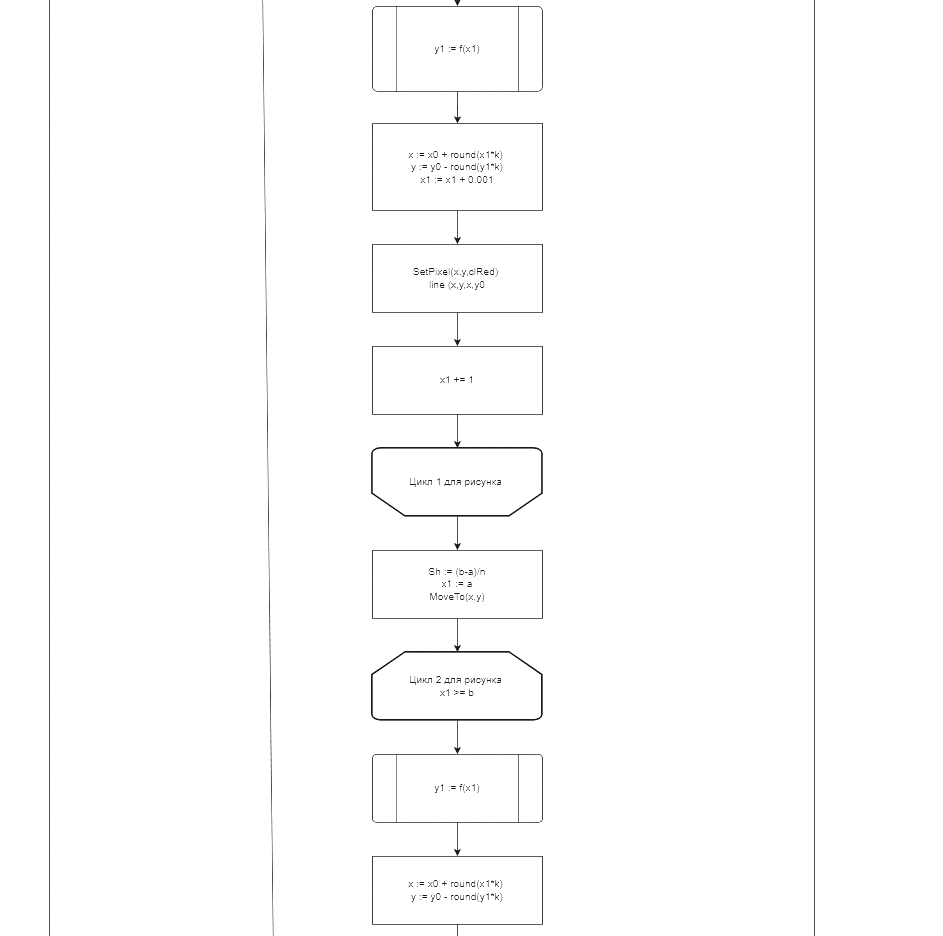


Рисунок 4.3 – схема алгоритмов основной программы

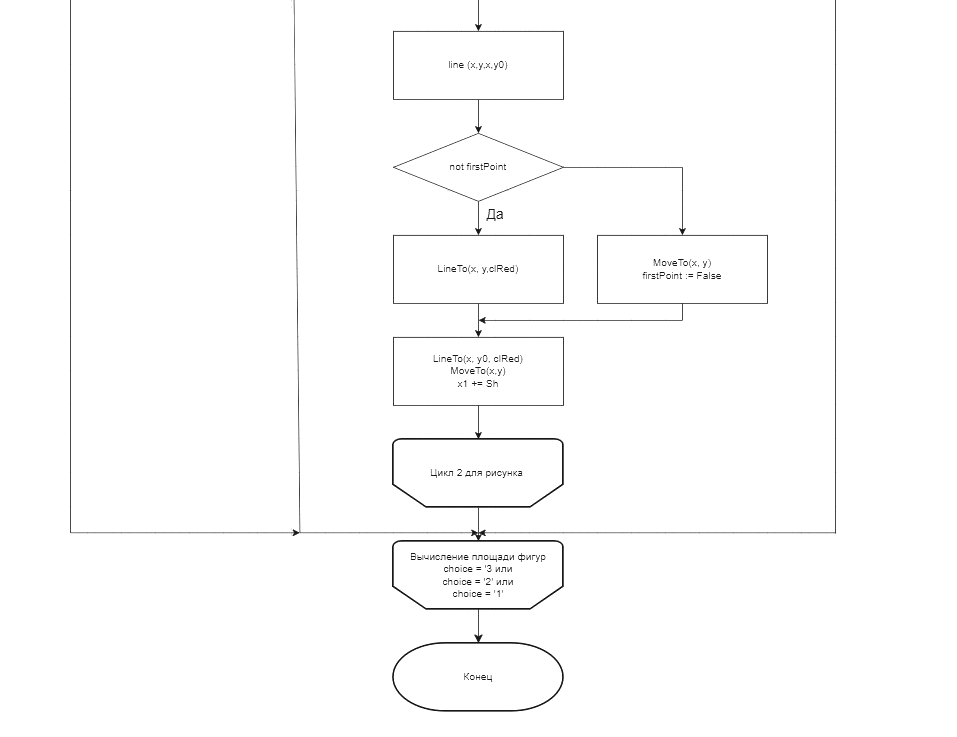


Рисунок 4.4 – схема алгоритмов основной программы

**Код программы:**

**program** one;

**uses** GraphABC;

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

**begin**

f := x \* x \* x + 2 \* x \* x + 3 \* x + 3;

**end**;

**function** simpsons\_rule(a, b: real; n: integer): real;

**var**

h, integral: real;

i: integer;

**begin**

**if** n **mod** 2 <> 0 **then**

n := n + 1; // Если n нечетное, увеличиваем на 1

h := (b - a) / n;

integral := f(a) + f(b);

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

**begin**

**if** i **mod** 2 = 1 **then**

integral := integral + 4 \* f(a + i \* h) // сумма членов с нечетными индексами

**else**

integral := integral + 2 \* f(a + i \* h); // сумма членов с чётными индексами

**end**;

integral := integral \* h / 3;

simpsons\_rule := integral;

**end**;

**function** error\_estimate(a, b: real; n: integer): real;

**var**

second\_derivative\_max, x, error: real;

i: integer;

**begin**

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

**begin**

x := a + i \* (b - a) / n;

**if** (6 \* x + 4) > second\_derivative\_max **then** // первообразная 2-ого порядка от начальной функции

second\_derivative\_max := 6 \* x + 4;

**end**;

error := (b - a) \*\* 5 / (180 \* n \*\* 4) \* second\_derivative\_max; // Формула для вычисления погрешности

error\_estimate := error;

**end**;

**var**

a, b, Sh: real;

n, x0, y0, x, y: integer;

choice: char;

area, error, k, x1, y1: real;

firstPoint: boolean;

**begin**

SetWindowSize(1000,500);

**repeat**

writeln('Меню:');

writeln('1. вычислить площадь фигуры');

writeln('2. визуализация');

writeln('3. выход');

readln(choice);

**case** choice **of** '1':

**begin**

writeln('Введите нижний предел интегрирования (a): ');

readln(a);

writeln('Введите верхний предел интегрирования (b): ');

readln(b);

writeln('Введите количество подынтервалов: ');

readln(n);

area := simpsons\_rule(a, b, n);

error := error\_estimate(a, b, n);

writeln(' площадь фигуры: ', area:0:6);

writeln(' оценка погрешности: ', error:0:6);

**end**;

'2':

**begin**

writeln('введите нижний предел интегрирования (a).');

readln(a);

writeln('введите верхний предел интегрирования (b).');

readln(b);

writeln('введите количество подынтервалов.');

readln(n);

writeln('введите масштаб (k-лучше от 5 до 30).');

readln(k);

area := simpsons\_rule(a, b, n);

error := error\_estimate(a, b, n);

clearwindow;

x0:= 600;

y0:= 500;

line(10, y0, 1190, y0);

line(x0, 10, x0, 990);

writeln('площадь фигуры: ', area:0:6);

writeln('оценка погрешности: ', error:0:6);

x1 := a;

firstPoint := True;

**while** x1 < b **do**

**begin**

y1 := f(x1);

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

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

SetPixel(x,y,clFuchsia);

line (x,y,x,y0);

x1 := x1 + 0.001;

**end**;

Sh := (b-a)/n;

x1 := a;

MoveTo(x,y);

**while** x1 < b **do**

**begin**

y1 := f(x1);

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

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

line (x,y,x,y0);

**if not** firstPoint **then**

**begin**

LineTo(x, y,clFuchsia);

**end**

**else**

**begin**

MoveTo(x, y);

firstPoint := False

**end**;

LineTo(x, y0, clFuchsia);

MoveTo(x,y);

x1 += Sh;

**end**;

**end**;

'3':

**begin**

writeln('вы завершили работу.');

sleep(1000);

closewindow;

**end**

**else**

writeln('неверный выбор. Попробуйте снова.');

**end**;

**until** (choice = '3') **or** (choice = '2') **or** (choice = '1');

**end**.

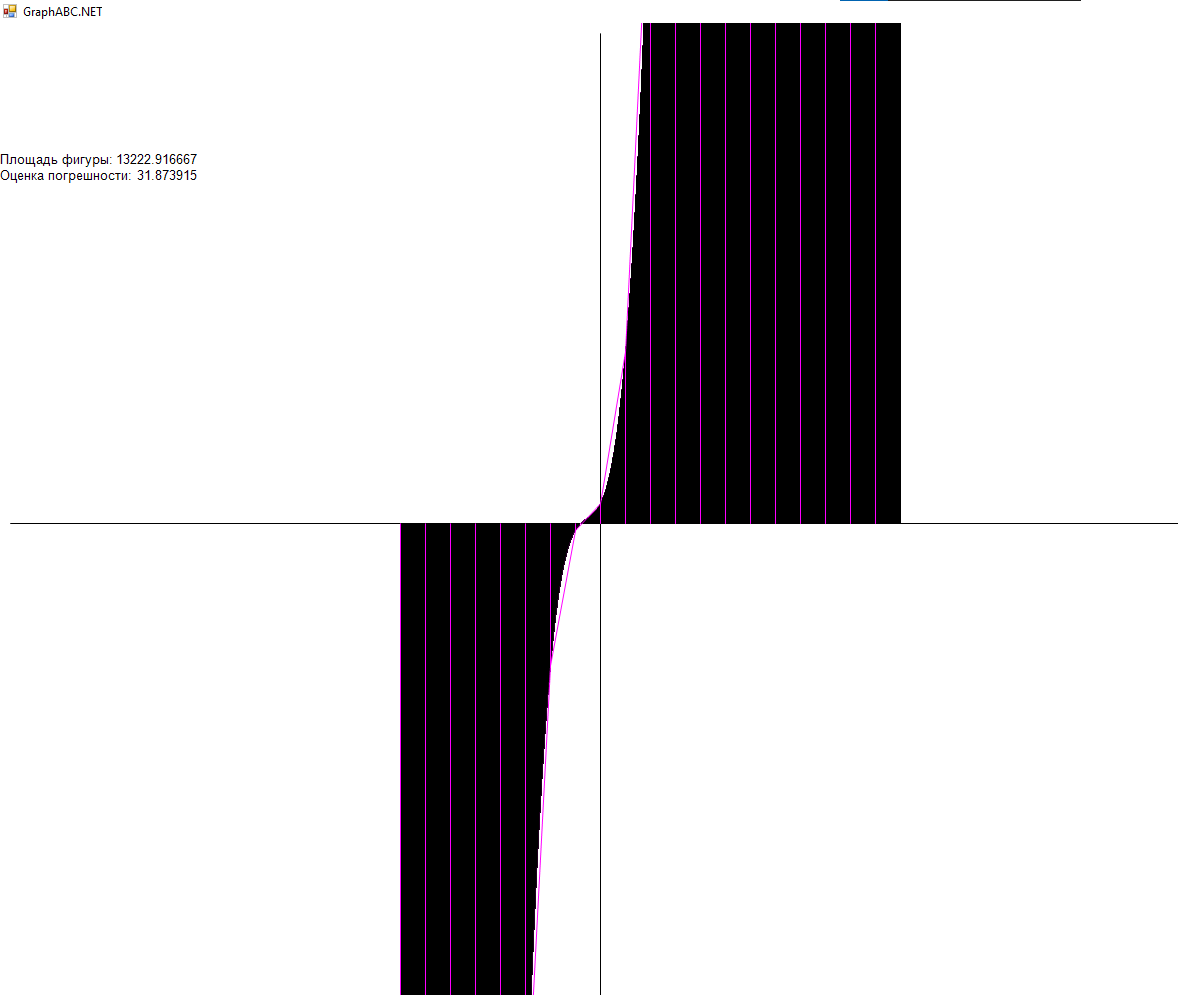


Рисунок 5 – результат выполнения программы

**Вывод:** Работа, которую я выполнял, не сильно отличалась от предыдущего опыта, однако на этот раз мне пришлось уделить ей немного больше времени и внимания.

Этот проект стал отличной возможностью для глубокого погружения в язык программирования Pascal, что позволило мне существенно расширить свои знания и навыки. Особенно мне понравилась работа с библиотекой GraphABC, которая предоставила мне возможность визуализировать работу различных функций.

В итоге, хотя задача не оказалась чрезмерно сложной, она была достаточно увлекательной и интересной, что сделало процесс работы по-настоящему захватывающим. Я получил не только новые знания, но и удовольствие от реализации задуманного.