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-204-52-00

Сокованов Игорь Николаевич

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2022

1. Цель работы: освоить принципы работы в графическом режиме; получить базовые навыки взаимодействия с графическими примитивами.
2. Задание:

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. Код программы

**uses** graphABC;

**var**

vvod: byte;

m1, m2: real;

**var**

n: integer;

**function** vibor(**var** v: byte): byte; //Для выбора ответа

**begin**

writeln('Да - 1');

writeln('Нет - 0');

Write('Выберите вариант: ');

readln(v);

vibor := v;

writeln;

**end**;

**function** fun(**var** x: real): real; //Функция основная

**begin**

**var** fo: real;

fo := 2 \* power(x, 3) + power(x, 2) + (-4) \* x + 15;

fun := fo;

**end**;

**function** fun1(**var** x: real): real; //Функция первообразная

**begin**

**var** fo: real;

fo := 1 / 2 \* power(x, 4) + 1 / 3 \* power(x, 3) - 2 \* power(x, 2) + 15 \* x;

fun1 := fo;

**end**;

//Печать качества Гошандрийского фонда

**function** graph(**var** s1, s2: real): integer; // график

**var**

x, mx, my: real;

a, b, x0, y0, i: integer;

**begin**

MaximizeWindow;

clearwindow;

setpencolor(clblack);

a := -5;

b := 100;

x0 := windowwidth **div** 2;

y0 := windowheight **div** 2;

mx := m1;

my := m2;

line(0, y0, windowwidth, y0);

line(x0, 0, x0, windowheight);

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

**begin**

line(x0 + round(i \* mx), y0 - 3, x0 + round(i \* mx), y0 + 3);

line(x0 - round(i \* mx), y0 - 3, x0 - round(i \* mx), y0 + 3);

line(x0 - 3, y0 + round(i \* my), x0 + 3, y0 + round(i \* my));

line(x0 - 3, y0 - round(i \* my), x0 + 3, y0 - round(i \* my));

textout(x0 + round(i \* mx), y0 + 10, inttostr(i));

textout(x0 - round(i \* mx), y0 + 5, inttostr(-i));

textout(x0 - 25, y0 - round(i \* my), inttostr(i));

textout(x0 - 20, y0 + round(i \* my), inttostr(-i));

**end**;

textout(x0 + 5, y0 + 10, '0');

textout(windowwidth - 10, y0 - 15, 'X');

textout(x0 + 5, 10, 'Y');

x := a;

setpencolor(clblue);

line(x0 + round(s1 \* mx), 0, x0 + round(s1 \* mx), windowheight);

setpencolor(clgreen);

line(x0 + round(s2 \* mx), 0, x0 + round(s2 \* mx), windowheight);

**var** l, w, k, e: real;

l := round((x0 + round(s2 \* mx) - x0 - round(s1 \* mx)) / n);

k := l;

w := 0;

x := a;

**while** x <= b **do**

**begin**

**if** (x0 + round(s1 \* mx)) = (x0 + round(x \* mx)) **then**

**begin**

**if** y0 - round(fun(x) \* my) < y0 **then**

**begin**

setpencolor(clblue);

Rectangle(x0 + round(s1 \* mx), y0 - round(fun(x) \* my), x0 + round(s1 \* mx + l), y0);

**end**;

**end**;

**if** (x0 + round(s1 \* mx + l)) = (x0 + round(x \* mx)) **then**

**begin**

**if** y0 - round(fun(x) \* my) < y0 **then**

**begin**

setpencolor(clblue);

**for var** v := x0 + round(s1 \* mx - l) **to** x0 + round(s2 \* mx) **do**

Rectangle(x0 + round(s1 \* mx + l), y0 - round(fun(x) \* my), x0 + round(s1 \* mx + (k + l)), y0);

**end**;

l := l + k;

w := w + 1;

**if** (w + 1) = n **then**

**break**;

**end**;

x := x + 0.001;

**end**;

x := a;

**while** x <= b **do**

**begin**

setpixel(x0 + round(x \* mx), y0 - round(fun(x) \* my), clred);

x := x + 0.001;

**end**;

**end**;

**function** graphDOP(**var** s1, s2: real): integer;

**var**

con: integer;

s: string;

**begin**

con := 0;

**repeat**

s := 'Хотите ввести маштаб по x(сейчас стоит ' + m1 + ' маштаб)';

textout(10, 10, s);

textout(10, 30, 'или же');

s := 'Хотите ввести маштаб по у(сейчас стоит ' + m2 + ' маштаб)';

textout(10, 50, s);

textout(10, 70, '1 - Изменить маштаб по x');

textout(10, 90, '2 - Изменить маштаб по y');

textout(10, 110, '0 - Продолжить');

read(vvod);

**case** vvod **of**

1:

**begin**

textout(10, 130, 'Введите маштаб(от 10 до 50)');

readln(m1);

graph(s1, s2);

**end**;

2:

**begin**

textout(10, 130, 'Введите маштаб(от 10 до 50)');

readln(m2);

graph(s1, s2);

**end**;

0: con := 1;

**end**;

**until** con = 1;

graph(s1, s2);

**end**;

**function** predel: integer; //Оснавная функция

**begin**

**var** a, b, h, f, x, S, pog: real;

**var** ss: string;

clearwindow;

textout(10, 10, 'Вычисление площади фигуры, ограниченной кривой 2\*x^3+x^2+(-4)\*x+15 и осью Ох (в положительной части по оси Оу)');

textout(10, 30, 'Введите пределы интегрирования:');

readln(a, b);

ss := a + ' ' + b;

textout(230, 30, ss);

textout(10, 50, 'Количество интервалов разбиения:');

readln(n);

textout(235, 50, n);

h := (b - a) / n;

x := a;

m1 := 20;

m2 := 20;

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

**begin**

f := fun(x);

S := S + f;

x := x + h;

**end**;

m1 := 20;

m2 := 20;

S := S \* h;

S := Round(S \* 1000) / 1000;

ss := 'Ответ ' + S;

textout(10, 70, ss);

textout(10, 90, 'Вывести погрешность?');

textout(10, 110, '1 - Да');

textout(10, 130, '0 - Нет');

read(vvod);

**case** vvod **of**

1:

**begin**

pog := abs((fun1(b) - fun1(a)) - S);

pog := Round(pog \* 1000) / 1000;

ss := 'Погрешность = ' + pog;

textout(10, 150, ss);

textout(10, 170, 'Нажмите enter чтоб продолжить');

readln();

readln;

**end**;

0:

**begin**

textout(10, 150, 'Нажмите enter чтоб продолжить');

readln();

readln;

**end**;

**end**;

graph(a, b);

graphDOP(a, b);

textout(10, 10, 'Начать заново?');

textout(10, 30, '1 - Конечно');

textout(10, 50, '0 - Нет');

read(vvod);

**case** vvod **of**

1: predel;

0: **exit**();

**end**;

predel := 0;

**end**;

**begin** //Главное меню

MaximizeWindow;

Writeln('Вычисление площади фигуры, ограниченной кривой 2\*x^3+x^2+(-4)\*x+15 и осью Ох (в положительной части по оси Оу)');

Writeln('Ввести пределы интегрирования в ручную?');

vvod := vibor(vvod);

**case** vvod **of**

1: predel;

**end**;

**end**.

1. Результат выполнения программы

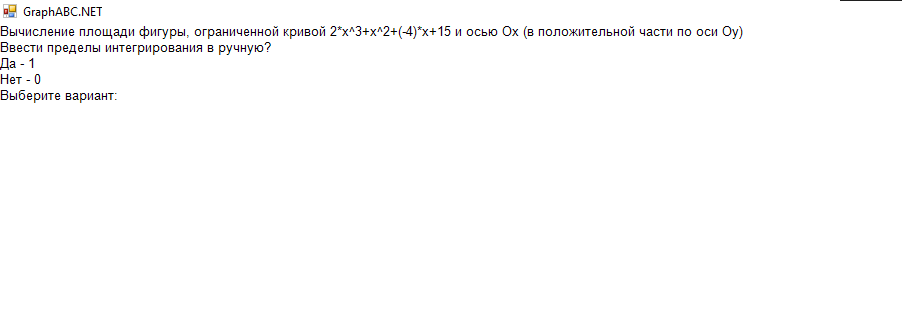


Рисунок 1- Результаты выполнения программы (1/6)

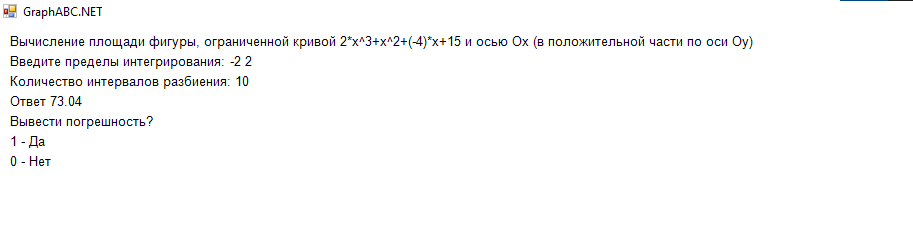


Рисунок 2- Результаты выполнения программы (2/6)

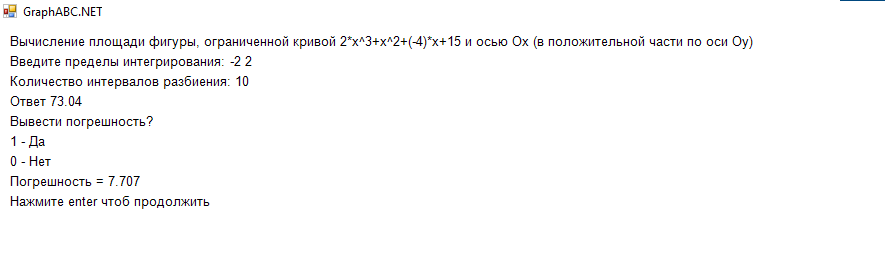


Рисунок 3- Результаты выполнения программы (3/6)

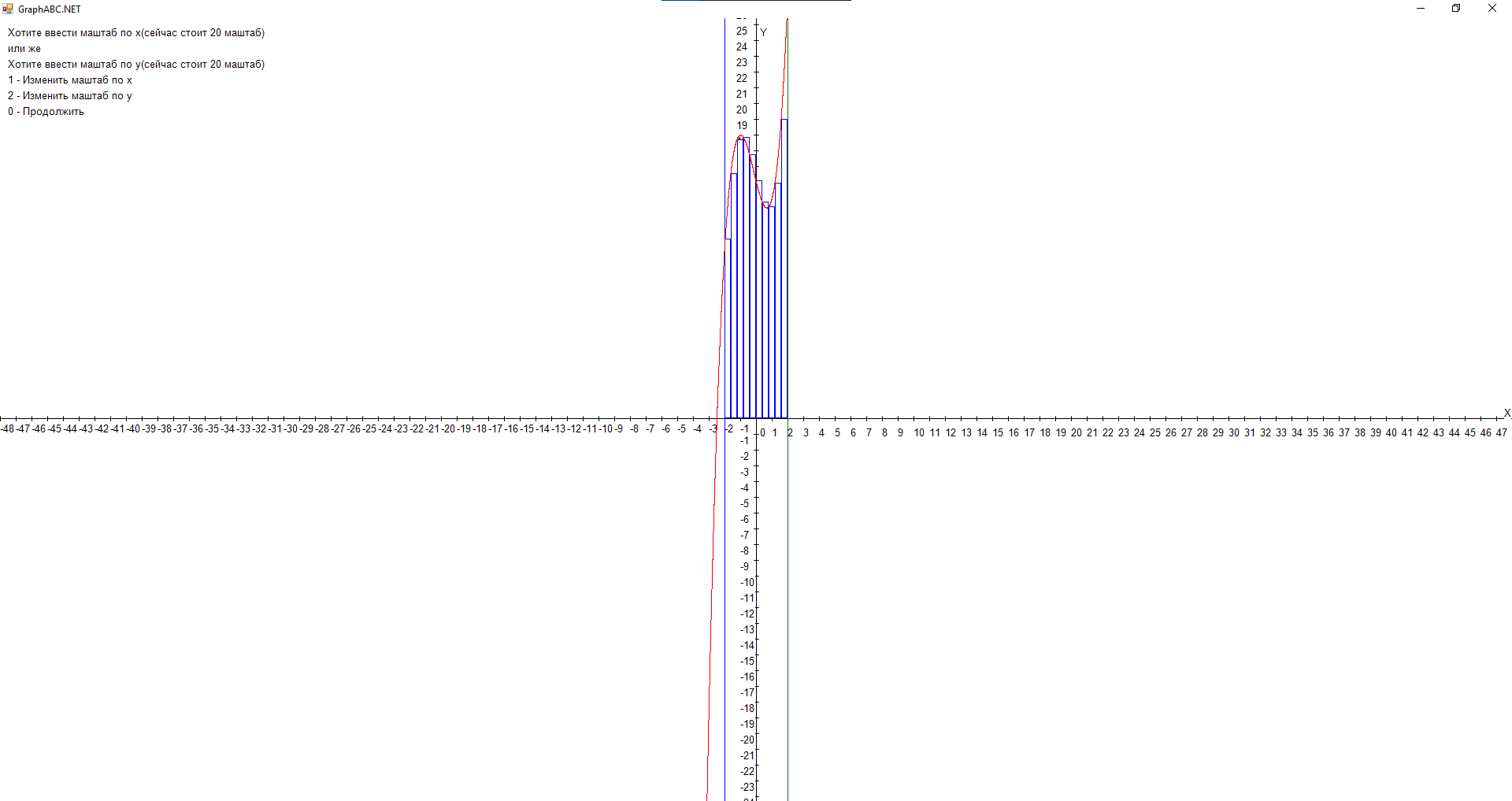


Рисунок 4- Результаты выполнения программы (4/6)

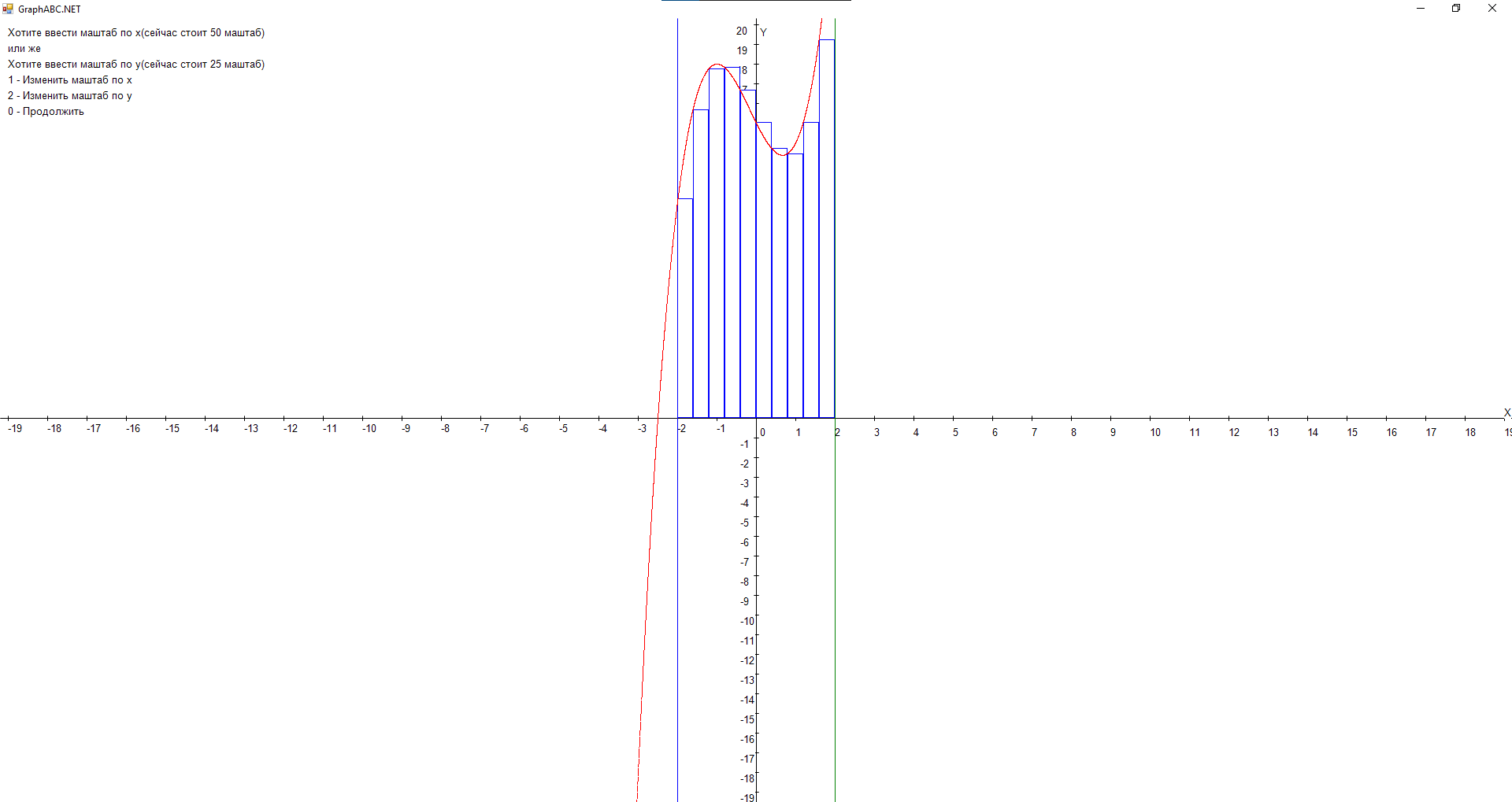


Рисунок 5- Результаты выполнения программы (5/6)

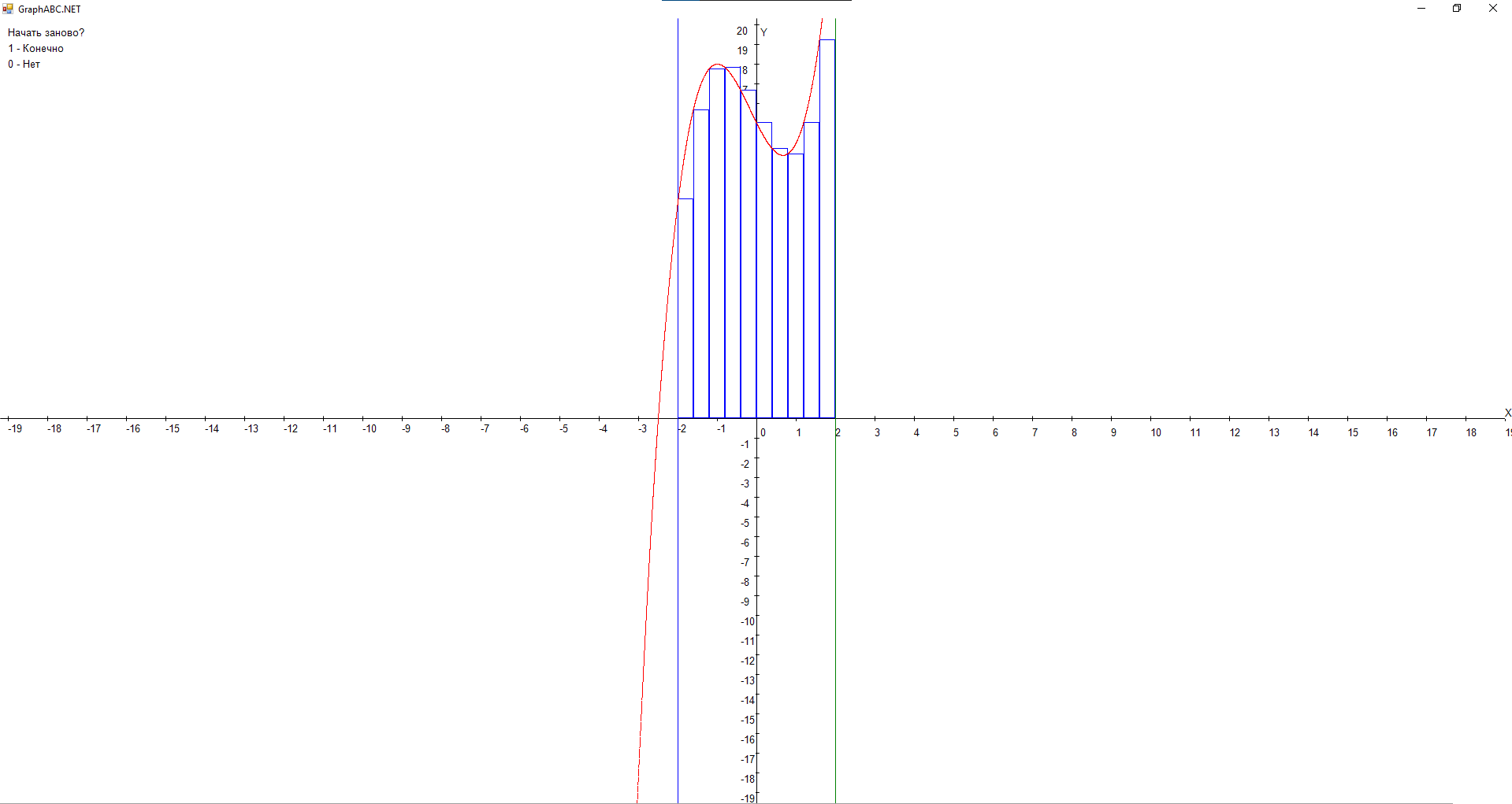


Рисунок 6- Результаты выполнения программы (6/6)

1. Вывод

В работе научился работать в графическом режиме и строить графики.

В программе график строится через пять переменных. Первые две переменные называются x0 и y0 они нужны для нахождение нулевой точки экрана. Из-за того, что экраны бывают разного разрешения поэтому моя программа, находит нулевую точку экрана. Следующие две переменные называются mx и my они нужны для масштаба графика, то есть для нахождения расстояние в пикселях от нулевой точки до единичной. При изменении эти переменных меняется весь график, то есть масштаб графика. Последняя переменная называется b она нужна для того чтоб определить сколько подписей на осях будет. Обычна эта переменная не трогалась так как при изменении её график выглядел либо сжато, либо программа вообще не работала.

В этой программе возникли трудности:

Были трудности с построения графика так как вообще не представлял, как можно было построить график.

Были трудности с построением кривой линии так как опять же не представлял, как можно было построить кривую линию. Но нашёл решение, просто рисовал много плотных точек которые напоминают кривую линию.