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

Максимов Антон Николаевич

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2022

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

2. Задание:

Вариант: 12.

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.

3. Схема алгоритма:

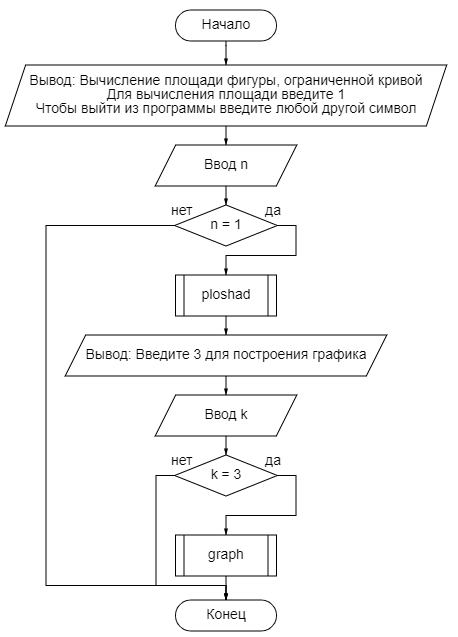


Рисунок 1 – Схема алгоритма (1/4)

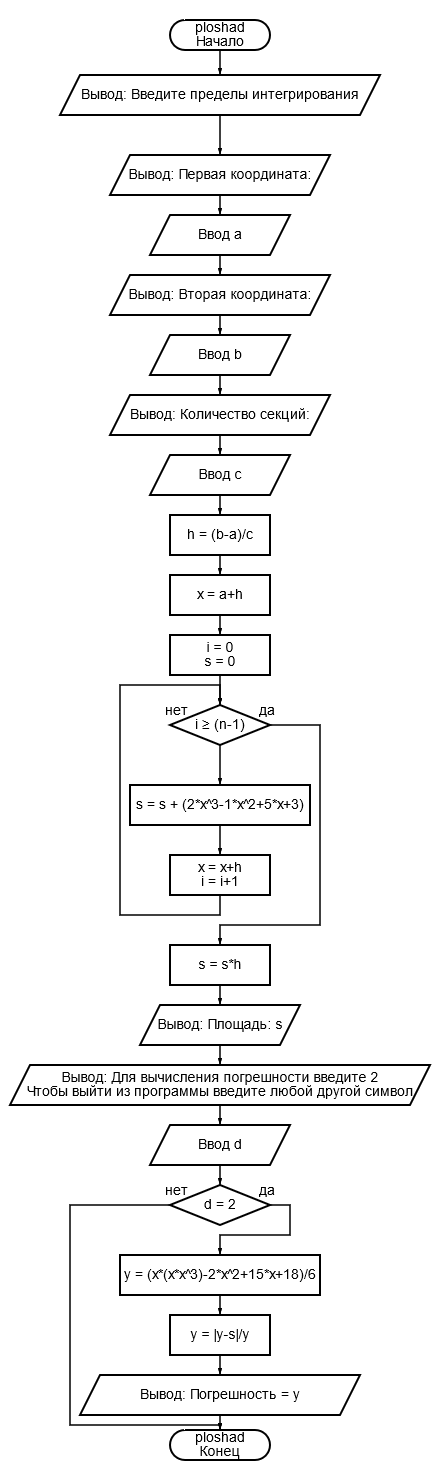


Рисунок 2 – Схема алгоритма (2/4)

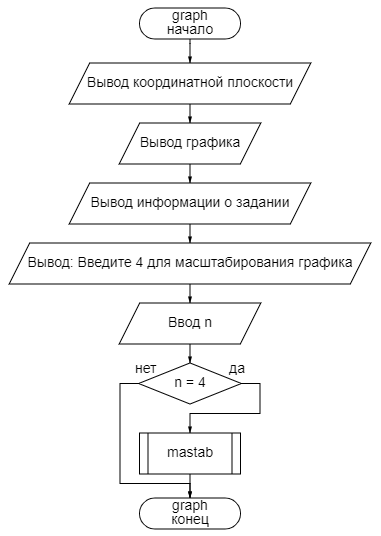


Рисунок 3 – Схема алгоритма (3/4)

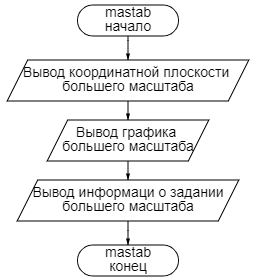


Рисунок 4 – Схема алгоритма (4/4)

4. Код программы:

**uses** graphABC;

**var**

vvod: byte;

m1, m2: real;

**var**

n: integer;

**function** vibor(**var** v: byte): byte;

**begin**

readln(v);

vibor := v;

writeln;

**end**;

**function** fun(**var** x: real): real;

**begin**

**var** fo: real;

fo := (exp(ln(x)\*3))-1\*(exp(ln(x)\*2))+5\*x+3;

fun := fo;

**end**;

**function** fun1(**var** x: real): real;

**begin**

**var** fo: real;

fo := (x\*(x\*(exp(ln(x)\*3))-2\*(exp(ln(x)\*2))+15\*x+18))/6;

fun1 := fo;

**end**;

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

**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(clyellow);

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(clpurple);

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(clorange);

**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), clgreen);

x := x + 0.001;

**end**;

**end**;

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

**var**

con: integer;

s: string;

**begin**

con := 0;

**repeat**

s := 'Масштаб по X - ' + m1;

textout(10, 10, s);

s := 'Масштаб по Y - '+ m2;

textout(10, 30, s);

textout(10, 50, 'Введите 1 для изменения масштабирования по оси X');

textout(10, 70, 'Введите 2 для изменения масштабирования по оси Y');

textout(10, 90, 'Введите 0 для завершения программы');

read(vvod);

**case** vvod **of**

1:

**begin**

textout(10, 110, 'Укажите коэффицент масшатабирования от 10 до 50');

readln(m1);

graph(s1, s2);

**end**;

2:

**begin**

textout(10, 130, 'Укажите коэффицент масшатабирования от 10 до 50');

readln(m2);

graph(s1, s2);

**end**;

0: con := 1;

**end**;

**until** con = 1;

**exit**();

**end**;

**function** predel: integer;

**begin**

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

**var** ss: string;

writeln('Введите пределы интегрирования');

writeln('Первая координата:');

readln(a);

writeln('Вторая координата:');

readln(b);

writeln('Колличество секций:');

readln(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;

writeln('Площадь - ', S);

writeln('Введите 1 для вывода погрешности, 2 для вывода графика функции или 0 для завершения программы');

read(vvod);

**case** vvod **of**

1:

**begin**

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

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

ss := 'Погрешность = ' + pog;

writeln('Погрешность вычисления - ', pog);

writeln('Введите 1 для вывода графика функции, для завершения программы 0');

read(vvod);

**case** vvod **of**

1:graph(a,b);

0: **exit**();

**end**;

**end**;

2:graph(a,b);

0:**exit**();

**end**;

graphDOP(a, b);

**end**;

**begin**

setwindowsize(800,600);

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

Writeln('Для вычисления площади введите 1, для завершения программы 0');

vvod := vibor(vvod);

**case** vvod **of**

1: predel;

**end**;

**end**.

5. Результат выполнения программы:

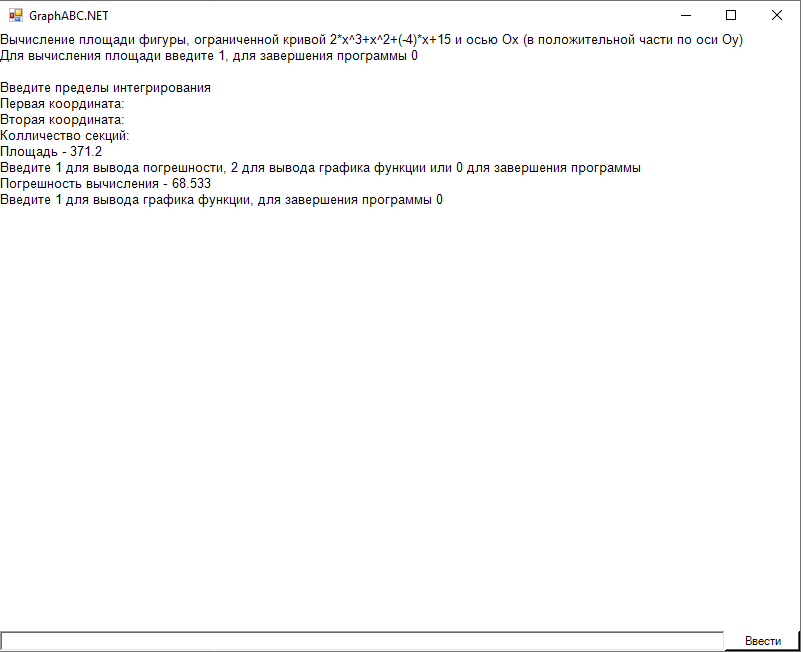


Рисунок 3 – Результат выполнения программы (1/2)

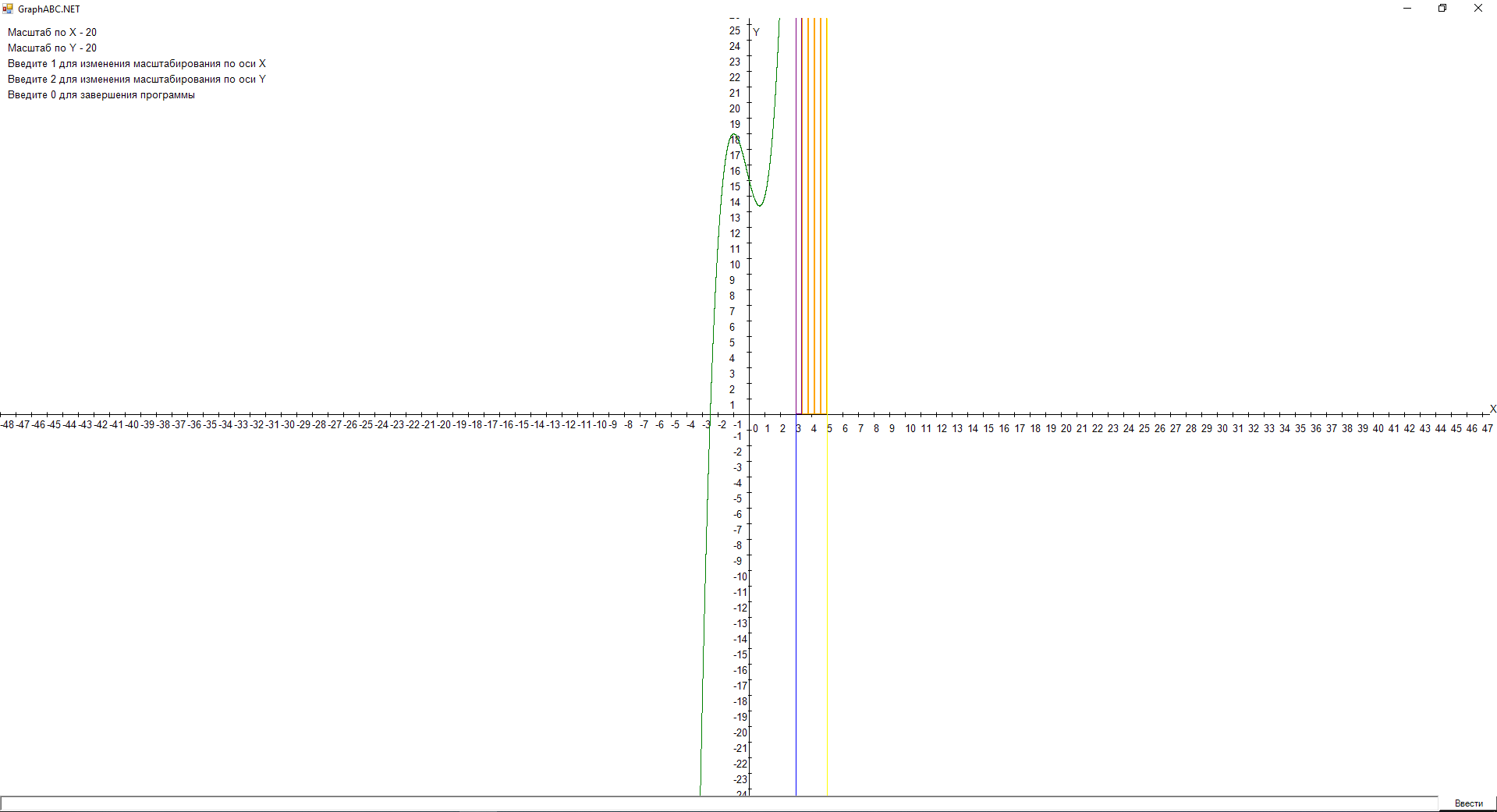


Рисунок 4 – Результат выполнения программы (2/2)

1. Вывод:

Для выполнения задания были использованы функции библиотеки GraphABC языка Pascal.

Для построения координатной плоскости, вывода графика, масштабирования графика, штриховки фигуры, ограничиваемой графиком и вывода подписей были использованы следующие функции: line, moveto, lineto, textout, setpencolor, setfontsize, setfontcolor.

Также были применены функции библиотеки CRT для организации взаимодействия с пользователем посредством case-меню.

К вышеперечисленному можно добавить использование операторов if, циклов for, внутриблочное описание переменных, инициализации переменных при описании, расширенное присваивание, для вывода используется функция Print.

Благодаря всему вышеперечисленному задание было успешно выполнено.