МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Тихоокеанский государственный университет»

Кафедра «Программное обеспечение вычислительной техники и автоматизированных систем»

Настройка элементов управления и создание интерфейса в среде программирования Visual Studio на языке C#

Практическая работа №4

по дисциплине «Визуальное программирование»

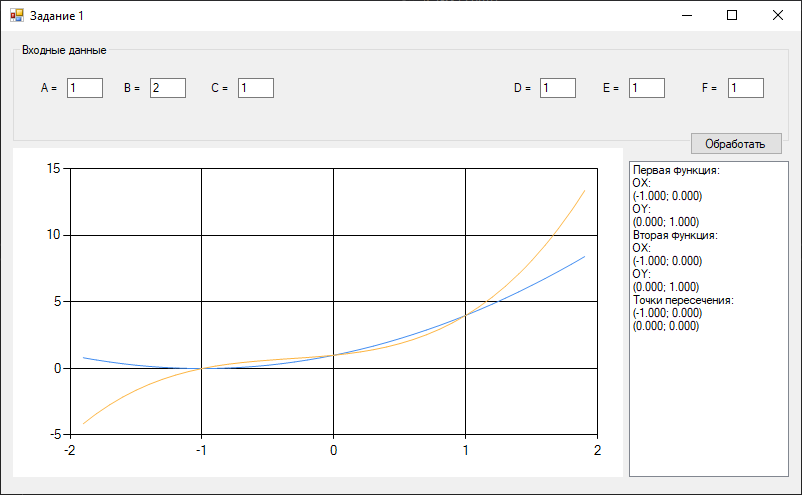
Выполнил студент Пшеничный Д.О.

Факультет, группа ФКФН, ПО(аб)-81

Проверила Резак Е.В.

Хабаровск – 2021г.

Цель: изучить способы настройки элементов управления, научиться создавать пользовательский интерфейс.



Код программы

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Task1

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

}

// Константы для бинпоиска.

const double minX = -1e4, maxX = 1e4;

// Значение кубического полинома в точке.

double f(double a, double b, double c, double d, double x)

{

return a \* x \* x \* x + b \* x \* x + c \* x + d \* d;

}

// Поиск корня на монотонном интервале квадратичного полинома.

double binsearch(double a, double b, double c, double d, double l, double r)

{

// Если нуля нет,

if ((f(a, b, c, d, l) > 0 && f(a, b, c, d, r) > 0) ||

(f(a, b, c, d, l) < 0 && f(a, b, c, d, r) < 0)) // вернем признак того, что корня нет.

return Double.NaN;

// Иначе, в цикле пока интервал не сузился до погрешности,

while (r - l > 1e-6)

{

// Берем середину интервала,

double m = (l + r) / 2;

// В зависимости от значения функции в точке, переносим границы поиска.

if (((f(a, b, c, d, m) > 0) && (f(a, b, c, d, l) > 0)) ||

((f(a, b, c, d, m) < 0) && (f(a, b, c, d, l) < 0)))

l = m;

else

r = m;

}

// Возвращаем ответ.

return l;

}

// Поиск списка корней кубического полинома.

List<double> solve(double a, double b, double c, double d)

{

// Ищем производную, берем дискриминант.

double A = a \* 3, B = b \* 2, C = c, D = B \* B - 4 \* A \* C;

double x1, x2, v;

List<double> answer = new List<double>();

// Рассматриваем случай линейной функции.

if (Math.Abs(A) < 1e-9)

{

// Тогда интервала поиска два.

x1 = -C / B;

v = binsearch(a, b, c, d, x1, maxX);

if (v != Double.NaN && (answer.Count == 0 || Math.Abs(answer.Last() - v) > 1e-9))

answer.Add(v);

return answer;

}

// Если действительных корней у производной нет, ищем корни на всём интервале, т.к. функция монотонна.

if (D < 0)

{

v = binsearch(a, b, c, d, minX, maxX);

if (v != Double.NaN && (answer.Count == 0 || Math.Abs(answer.Last() - v) > 1e-9))

answer.Add(v);

// Случай спаренных действительных корней производной.

}

else if (Math.Abs(D) < 1e-9)

{

x1 = -(B / 2 \* A);

// Два интервала, на которые разбивает этот корень, ищем корень на обоих.

v = binsearch(a, b, c, d, minX, x1);

if (v != Double.NaN && (answer.Count == 0 || Math.Abs(answer.Last() - v) > 1e-9))

answer.Add(v);

v = binsearch(a, b, c, d, x1, maxX);

if (v != Double.NaN && (answer.Count == 0 || Math.Abs(answer.Last() - v) > 1e-9))

answer.Add(v);

}

else

{

// Случай различных корней.

double sqrtD = Math.Sqrt(d);

x1 = (-B + sqrtD) / (2 \* A);

x2 = (-B - sqrtD) / (2 \* A);

// Ищем на каждом из трёх отрезков.

v = binsearch(a, b, c, d, minX, x1);

if (v != Double.NaN && (answer.Count == 0 || Math.Abs(answer.Last() - v) > 1e-9))

answer.Add(v);

v = binsearch(a, b, c, d, x1, x2);

if (v != Double.NaN && (answer.Count == 0 || Math.Abs(answer.Last() - v) > 1e-9))

answer.Add(v);

v = binsearch(a, b, c, d, x2, maxX);

if (v != Double.NaN && (answer.Count == 0 || Math.Abs(answer.Last() - v) > 1e-9))

answer.Add(v);

}

return answer;

}

private void btWork\_Click(object sender, EventArgs e)

{

// Считываем данные.

double A = Convert.ToDouble(tbA.Text),

B = Convert.ToDouble(tbB.Text),

C = Convert.ToDouble(tbC.Text),

D = Convert.ToDouble(tbD.Text),

E = Convert.ToDouble(tbE.Text),

F = Convert.ToDouble(tbF.Text);

// Рисуем график.

ch.Series[0].Points.Clear();

ch.Series[1].Points.Clear();

for (double x = -1.9; x <= 2; x += 0.1)

{

ch.Series[0].Points.AddXY(x, A \* x \* x + B \* x + C);

ch.Series[1].Points.AddXY(x, x \* x \* x + D \* x \* x + E \* x + F);

}

// Выводим данные пользуясь описанными выше функциями.

List<double> s1, s2, s3;

lb.Items.Clear();

lb.Items.Add("Первая функция:");

lb.Items.Add("OX:");

s1 = solve(0, A, B, C);

for (int i = 0; i < s1.Count; i++)

lb.Items.Add("(" + s1[i].ToString("f3") + "; " + f(0, A, B, C, s1[i]).ToString("f3") + ")");

lb.Items.Add("OY:");

lb.Items.Add("(" + 0.ToString("f3") + "; " + f(0, A, B, C, 0).ToString("f3") + ")");

lb.Items.Add("Вторая функция:");

lb.Items.Add("OX:");

s2 = solve(1, D, E, F);

for (int i = 0; i < s2.Count; i++)

lb.Items.Add("(" + s2[i].ToString("f3") + "; " + f(1, D, E, F, s2[i]).ToString("f3") + ")");

lb.Items.Add("OY:");

lb.Items.Add("(" + 0.ToString("f3") + "; " + f(1, D, E, F, 0).ToString("f3") + ")");

lb.Items.Add("Точки пересечения:");

s3 = solve(1, D - A, E - B, F - C);

for (int i = 0; i < s3.Count; i++)

lb.Items.Add("(" + s3[i].ToString("f3") + "; " + f(1, D - A, E - B, F - C, s3[i]).ToString("f3") + ")");

}

}

}

Вывод: научились настраивать форму Windows с помощью различных методов.