МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний Авіаційний Університет

Факультет кібербезпеки, комп’ютерної та програмної інженерії

Кафедра прикладної математики

Лабораторна робота № 1

На тему:

Дисципліна: «Обчислювальна геометрія та комп´ютерна графіка»

Студента групи 351:

О.Т.Архіпова

Керівник:

асистент кафедри

прикладної математики

А. В. Темніков

Оцінка:

Київ 2023

**Зміст**

**Вступ………………………………………………………………………………………………………………..3**

**Постановка задачі……………………………………………………………………………………………4**

**Розробка програмного забезпечення…………………………………………………………….5**

**Висновок……………………………………………………………………………………………………….12**

**Додаток…………………………………………………………………………………………………………12**

**Вступ**

Формування зображення та різноманітні дії з ним вимагають від користувача відомої математичної грамотності. Геометричні поняття, формули та факти, що відносяться до плоского та тривимірного випадків, грають у завданнях комп'ютерної графіки особливу роль. Принципи аналітичної геометрії у поєднанні з можливостями обчислювальної техніки, що постійно розширюються, є невичерпним джерелом істотних поступів на шляху розвитку комп'ютерної графіки.

На сьогодні комп’ютерна графіка – це наукова область, що має безліч застосувань. Вона широко застосовується в різних сферах діяльності людини: будівництві та архітектурі, промисловості, в комп’ютерних іграх та кіноіндустрії. Крім того графіка знайшла своє місце і в медицині, астрономії, картографії, фотограмметрії і в багатьох інших областях знання.

**Постановка задачі**

Актуальність: широко застосовується у моделюванні та комп’ютерній графіці.

Мета дослідження: потрібно реалізувати інтерактивний метод миттєвої побудови кривих Безьє через алгоритм де Кастельжо.

Об’єкт дослідження: координати опорних точок на площині

Предмет дослідження: криві Безьє других і третіх порядків

Постановка задачі: потрібно реалізувати інтерактивний метод миттєвої побудови кривих Безьє через алгоритм де Кастельжо за допомогою координатів опорних точок на площині.

**Розробка програмного забезпечення**

Криві Безьє використовуються в комп'ютерній графіці для малювання плавних вигинів, CSS-анімації і багато де ще.

Це дуже проста річ, яку варто вивчити один раз, а потім відчувати себе комфортно у світі векторної графіки та просунутих анімацій.

Крива Безьє задається опорними точками.

Їх може бути дві, три, чотири чи більше. Наприклад:

За двома точками:

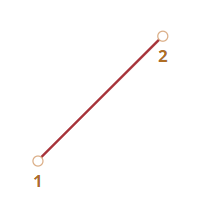


Рис 1

За трьома точками:

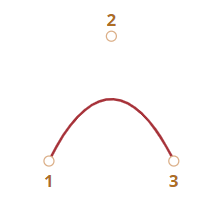


Рис 2

За чотирма точками:

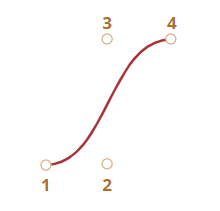


Рис 3

Якщо ви уважно подивіться на ці криві, то «на око» помітите:

Крапки не завжди на кривій. Це абсолютно нормально, як саме будується крива ми розглянемо трохи згодом.

Ступінь кривої дорівнює числу точок мінус один. Для двох точок – це лінійна крива (тобто пряма), для трьох точок – квадратична крива (парабола), для чотирьох – кубічна.

Крива завжди знаходиться всередині опуклої оболонки, утвореної опорними точками:

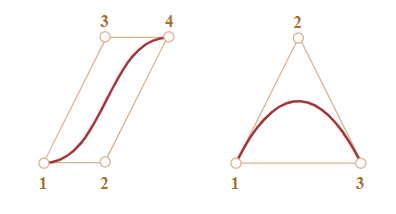


Рис 4

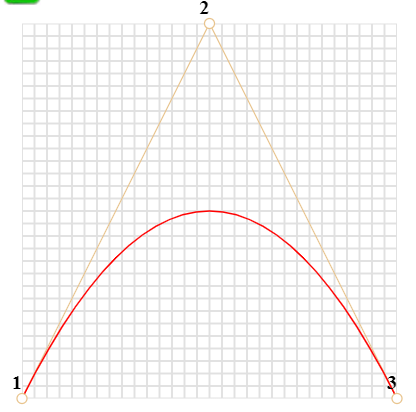


Рис 5

Побудова кривої Безьє з 3 точками за «алгоритмом де Кастельжо»:

1. Малюються опорні точки. У прикладі: 1, 2, 3.
2. Будуються відрізки між опорними точками у наступному порядку 1 → 2 → 3. На малюнку вони коричневі.
3. Параметр t «пробігає» значення від 0 до 1. У прикладі використано крок 0.05, тобто. у циклі 0, 0.05, 0.1, 0.15, ... 0.95, 1.

Для кожного з цих значень t:

* На кожному з коричневих відрізків береться точка, що знаходиться на відстані пропорційній t від його початку. Так як відрізків два, то й точок дві.
* Наприклад, при t = 0 - точки будуть на початку, при t = 0.25 - на відстані 25% від початку відрізка, при t = 0.5 - 50% (на середині), при t = 1 - в кінці відрізків.
* Ці точки з'єднуються. На малюнку нижче відрізок, що з'єднує їх, зображений синім.

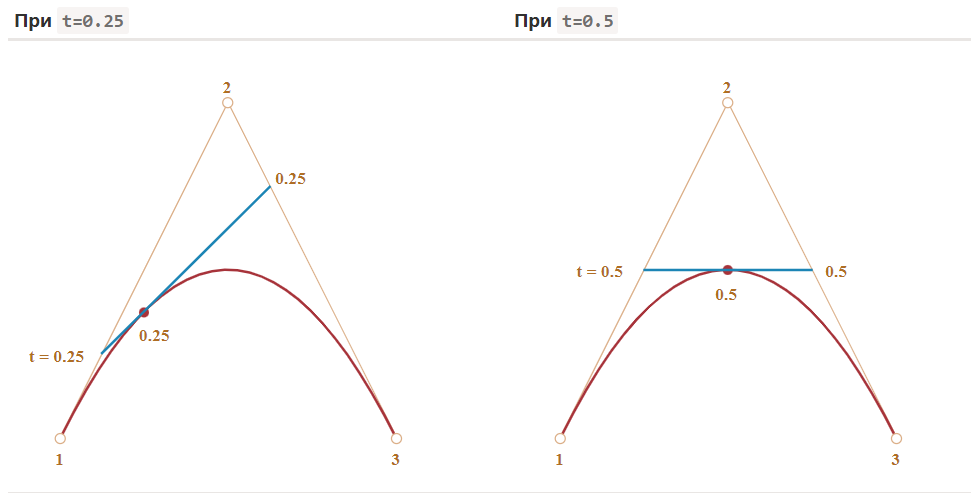


Рис 6

Крива Безьє може бути описана за допомогою математичної формули.

Координати кривої з контрольними точками Pi: перша контрольна точка має координати P1 = (x1, y1), друга: P2 = (x2, y2) і т.д. описуються рівнянням, що залежить від параметра t на відрізку [0,1].

Формула для 2-х точкової кривої:

P = (1-t) P1 + tP2

Для 3 контрольних точок:

P = (1−t)2P1 + 2(1−t)tP2 + t2P3

Для 4 контрольних точок:

P = (1−t)3P1 + 3(1−t)2tP2 +3(1−t)t2P3 + t3P4

Це векторні рівняння. Іншими словами, ми можемо поставити x та y замість P, щоб отримати відповідні координати.

Наприклад, 3-точкова крива утворена точками (x, y), розрахованими як:

x = (1−t)2x1 + 2(1−t)tx2 + t2x3

y = (1−t)2y1 + 2(1−t)ty2 + t2y3

Замість x1, y1, x2, y2, x3, y3 ми повинні помістити координати 3 контрольних точок, а потім при переміщенні від 0 до 1 для кожного значення t ми отримаємо (x, y) кривою.

Використавши цей алгоритм отримаємо:

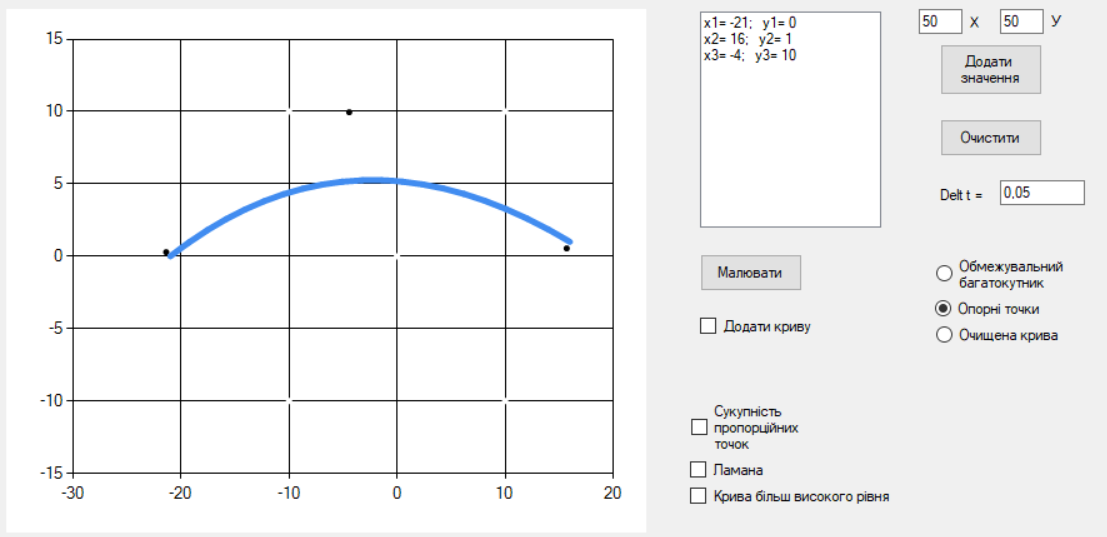


Рис 7

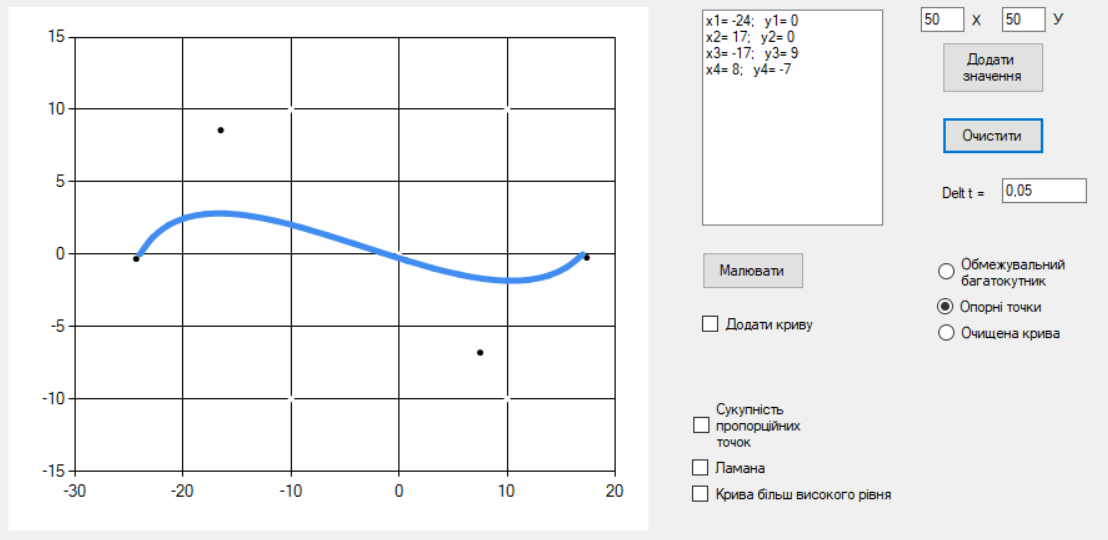


Рис 8

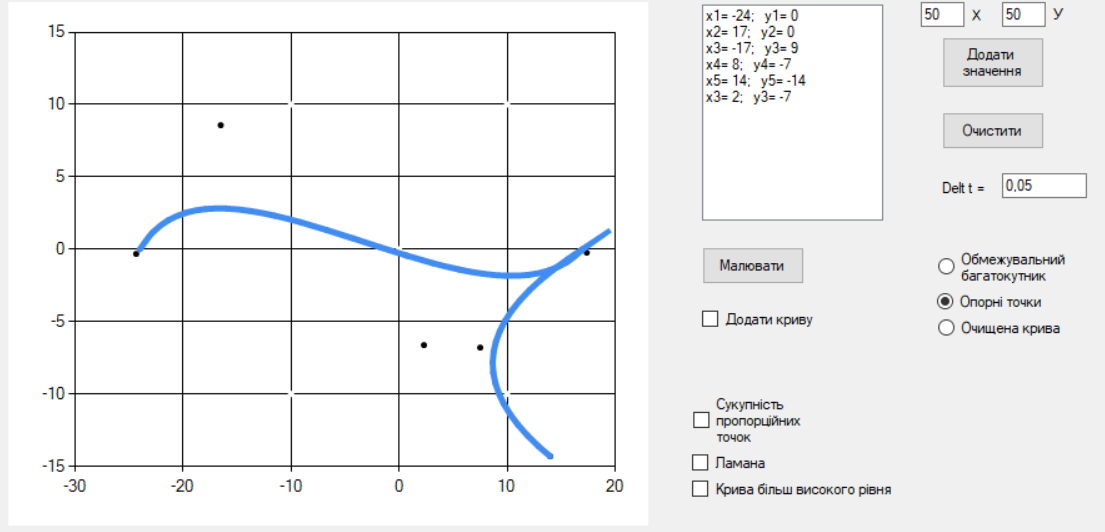


Рис 9

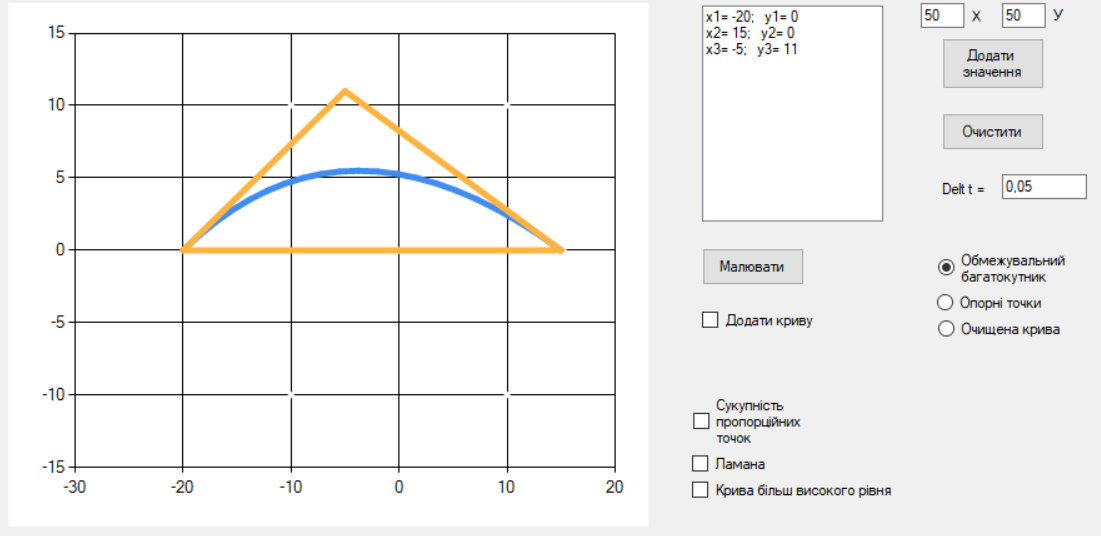


Рис 10

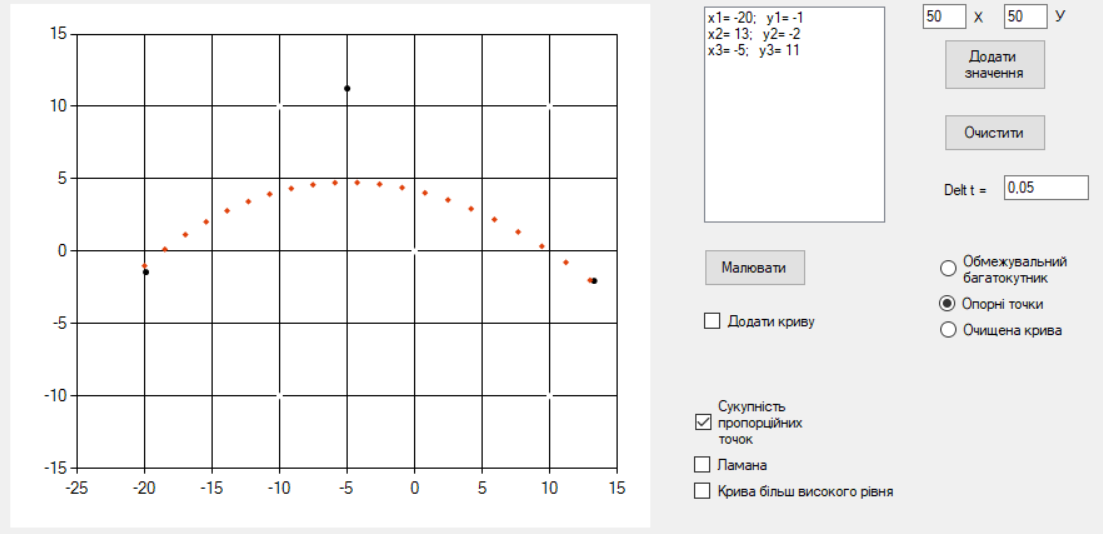


Рис 11

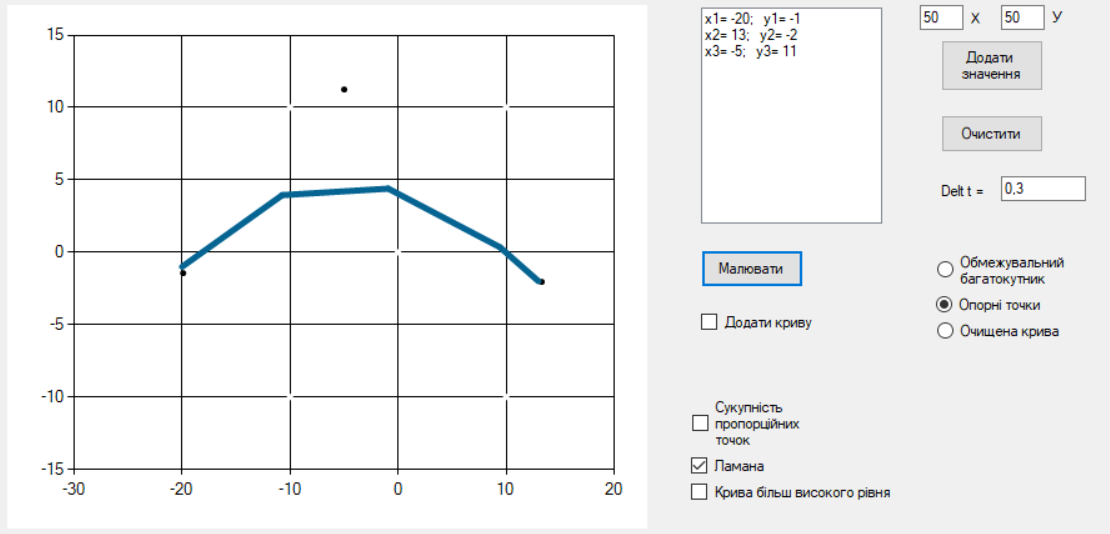


Рис 12

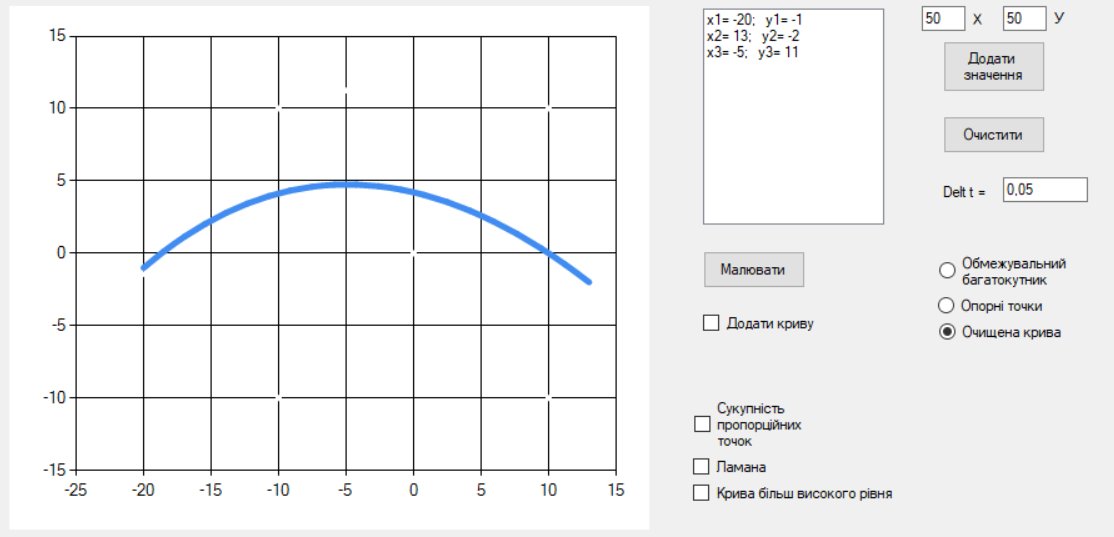


Рис 13

**Висновок**

1. Для реалізації інтерактивного метода миттєвої побудови кривих Безьє через алгоритм де Кастельжо я використав координати опорних точок на площині
2. Для знаходження кривих Безьє других і третіх порядків я використав алгоритм де Кастельжо.
3. Використав мову с# тому, що вона гарно підходить для візуалізації та зображень

**Додаток**

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;

using System.Windows.Forms.DataVisualization.Charting;

using static System.Windows.Forms.VisualStyles.VisualStyleElement.ProgressBar;

namespace lab1

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

Create\_chart\_dots();

radioButton2.Checked = true;

}

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

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

List<double> X2 = new List<double>();

List<double> Y2 = new List<double>();

int count = 1, count1, swap = 0, temp = 0;

void Swap<T>(List<T> list, int i, int j)

{

(list[i], list[j]) = (list[j], list[i]);

}

void Create\_chart\_dots()

{

chart1.Series[0].Color = Color.White;

chart1.Series[0].Points.AddXY(0, 0);

chart1.Series[0].Points.AddXY(10, 10);

chart1.Series[0].Points.AddXY(-10, -10);

chart1.Series[0].Points.AddXY(10, -10);

chart1.Series[0].Points.AddXY(-10, 10);

chart1.Series[0].ToolTip = "X = #VALX, Y = #VALY";

chart1.Series[1].ToolTip = "X = #VALX, Y = #VALY";

chart1.Series[1].BorderWidth = 5;

chart1.ChartAreas[0].AxisX.RoundAxisValues();

chart1.Legends.Clear();

chart1.Series[2].Points.Clear();

}

private void chart1\_MouseClick(object sender, MouseEventArgs e)

{

double x, y;

chart1.Series[2].Color = Color.Black;

chart1.Series[2].BorderWidth = 5;

if (count < 3)

{

x = chart1.ChartAreas[0].AxisX.PixelPositionToValue(e.X);

y = chart1.ChartAreas[0].AxisY.PixelPositionToValue(e.Y);

X.Add(Math.Round(x));

Y.Add(Math.Round(y));

listBox1.Items.Add("x" + (count).ToString() + "= " + Math.Round(x) + "; y" + (count).ToString() + "= " + Math.Round(y));

count++;

chart1.Series[2].Points.AddXY(x, y);

}

else if(count >= 3 && count < 5 && checkBox1.Checked == false)

{

if(X2.Count == 0) count = 3;

if (count == 4)

{

double x1, y1;

x = chart1.ChartAreas[0].AxisX.PixelPositionToValue(e.X);

y = chart1.ChartAreas[0].AxisY.PixelPositionToValue(e.Y);

int c4 = 0;

for (int i = 0; i < X2.Count; i++)

{

x1 = Math.Round(X2[i]) - Math.Round(x);

y1 = Math.Round(Y2[i]) - Math.Round(y);

if (Math.Abs(x1) <= 1 && Math.Abs(y1) <= 1)

{

X2.RemoveAt(i);

Y2.RemoveAt(i);

//chart1.Series[2].Points.RemoveAt(2 + i);

chart1.Series[2].Points.RemoveAt(chart1.Series[2].Points.Count-1);

count1 = i;

listBox1.Items.RemoveAt(listBox1.Items.Count-1);

c4 = 1;

break;

}

if (i == X2.Count - 1 && c4 == 0)

{

x = chart1.ChartAreas[0].AxisX.PixelPositionToValue(e.X);

y = chart1.ChartAreas[0].AxisY.PixelPositionToValue(e.Y);

if (swap == 1)

{

X2.Add(Math.Round(x));

Y2.Add(Math.Round(y));

Swap<double>(X2, 0, 1);

Swap<double>(Y2, 0, 1);

swap = 0;

}

else

{

X2.Add(Math.Round(x));

Y2.Add(Math.Round(y));

}

listBox1.Items.Add("x" + (count).ToString() + "= " + Math.Round(x) + "; y" + (count).ToString() + "= " + Math.Round(y));

count++;

chart1.Series[2].Points.AddXY(x, y);

if (temp == 0)

{

chart1.Series[1].Points.Clear();

chart1.Series[4].Points.Clear();

chart1.Series[5].Points.Clear();

}

else

{

int k = chart1.Series[1].Points.Count - 1;

do

{

chart1.Series[1].Points.RemoveAt(chart1.Series[1].Points.Count - 1);

k--;

} while (chart1.Series[1].Points[k].XValue != X[0]);

}

Draw();

break;

}

}

}

else

{

x = chart1.ChartAreas[0].AxisX.PixelPositionToValue(e.X);

y = chart1.ChartAreas[0].AxisY.PixelPositionToValue(e.Y);

X2.Add(Math.Round(x));

Y2.Add(Math.Round(y));

listBox1.Items.Add("x" + (count).ToString() + "= " + Math.Round(x) + "; y" + (count).ToString() + "= " + Math.Round(y));

count++;

chart1.Series[2].Points.AddXY(x, y);

if (temp == 0)

{

chart1.Series[1].Points.Clear();

chart1.Series[4].Points.Clear();

chart1.Series[5].Points.Clear();

}

else

{

int k3 = 0;

for(int k2 = 0; k2 < chart1.Series[1].Points.Count; k2++)

{

if(chart1.Series[1].Points[k2].XValue == X[1])

{

k3 = 1;

break;

}

}

if (k3 == 1)

{

int k = chart1.Series[1].Points.Count - 1;

do

{

chart1.Series[1].Points.RemoveAt(chart1.Series[1].Points.Count - 1);

k--;

} while (chart1.Series[1].Points[k].XValue != X[0]);

}

}

Draw();

}

}

else if(count == 5 && checkBox1.Checked == false)

{

double x1, y1;

x = chart1.ChartAreas[0].AxisX.PixelPositionToValue(e.X);

y = chart1.ChartAreas[0].AxisY.PixelPositionToValue(e.Y);

for(int i = 0; i < X2.Count; i++)

{

x1 = Math.Round(X2[i]) - Math.Round(x);

y1 = Math.Round(Y2[i]) - Math.Round(y);

if (Math.Abs(x1) <= 1 && Math.Abs(y1) <= 1)

{

X2.RemoveAt(i);

Y2.RemoveAt(i);

chart1.Series[2].Points.RemoveAt(2+i);

count1 = i;

listBox1.Items.RemoveAt(2 + i);

count++;

break;

}

}

}

else if(count == 6 && checkBox1.Checked == false)

{

double x1, y1;

x = chart1.ChartAreas[0].AxisX.PixelPositionToValue(e.X);

y = chart1.ChartAreas[0].AxisY.PixelPositionToValue(e.Y);

for (int i = 0; i < X2.Count; i++)

{

x1 = Math.Round(X2[i]) - Math.Round(x);

y1 = Math.Round(Y2[i]) - Math.Round(y);

if (Math.Abs(x1) <= 1 && Math.Abs(y1) <= 1)

{

X2.RemoveAt(i);

Y2.RemoveAt(i);

chart1.Series[2].Points.RemoveAt(2 + i);

count1 = i;

listBox1.Items.RemoveAt(2 + i);

count = 4;

swap = 1;

break;

}

}

if (count == 6)

{

X2.Insert(count1, Math.Round(x));

Y2.Insert(count1, Math.Round(y));

//X3 = cha

DataPoint d = new DataPoint(x, y);

listBox1.Items.Insert(count1 + 2, "x" + (count).ToString() + "= " + Math.Round(x) + "; y" + (count).ToString() + "= " + Math.Round(y));

chart1.Series[2].Points.Insert(count1 + 2, d);

if (temp == 0)

{

chart1.Series[1].Points.Clear();

chart1.Series[4].Points.Clear();

chart1.Series[5].Points.Clear();

}

Draw();

}

}

if(checkBox1.Checked == true)

{

double x1, y1;

int c5 = 0;

if (count == 4 || count == 5 || count == 6)

{

x = chart1.ChartAreas[0].AxisX.PixelPositionToValue(e.X);

y = chart1.ChartAreas[0].AxisY.PixelPositionToValue(e.Y);

for (int i = 0; i < X2.Count; i++)

{

x1 = Math.Round(X2[i]) - Math.Round(x);

y1 = Math.Round(Y2[i]) - Math.Round(y);

if (Math.Abs(x1) > 1 && Math.Abs(y1) > 1)

{

c5++;

}

}

if(c5 != 0)

{

X2.Clear();

Y2.Clear();

X.RemoveAt(0);

Y.RemoveAt(0);

X.Add(Math.Round(x, 3));

Y.Add(Math.Round(y, 3));

listBox1.Items.Add("x" + (count).ToString() + "= " + Math.Round(x) + "; y" + (count).ToString() + "= " + Math.Round(y));

count = 3;

chart1.Series[2].Points.AddXY(x, y);

temp = 1;

checkBox1.Checked = false;

}

}

}

}

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

{

chart1.Series[0].Points.Clear();

chart1.Series[1].Points.Clear();

chart1.Series[2].Points.Clear();

chart1.Series[3].Points.Clear();

chart1.Series[4].Points.Clear();

chart1.Series[5].Points.Clear();

X.Clear();

Y.Clear();

X2.Clear();

Y2.Clear();

count = 1;

listBox1.Items.Clear();

Create\_chart\_dots();

temp = 0;

}

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

{

double z = Convert.ToDouble(tbX.Text);//x

double k = Convert.ToDouble(tbY.Text);//y

chart1.Series[0].Color = Color.White;

chart1.Series[0].Points.AddXY(z, k);

}

void Draw()

{

if(X2.Count == 1)

{

List<double> x\_1 = new List<double>();

List<double> y\_1 = new List<double>();

chart1.Series[4].BorderWidth = 5;

chart1.Series[5].BorderWidth = 5;

double deltt = Convert.ToDouble(tbt.Text);

double x4 = deltt;

int i = 0;

if (checkBox5.Checked) chart1.Series[1].Points.AddXY(X[0], Y[0]);

else if (checkBox6.Checked) chart1.Series[5].Points.AddXY(X[0], Y[0]);

else if (checkBox7.Checked) chart1.Series[4].Points.AddXY(X[0], Y[0]);

else chart1.Series[1].Points.AddXY(X[0], Y[0]);

do

{

x\_1.Add(Math.Pow(1 - x4, 2) \* X[0] + 2 \* (1 - x4) \* x4 \* X2[0] + x4 \* x4 \* X[1]);

y\_1.Add(Math.Pow(1 - x4, 2) \* Y[0] + 2 \* (1 - x4) \* x4 \* Y2[0] + x4 \* x4 \* Y[1]);

//chart1.Series[1].Points.AddXY(x\_1[i], y\_1[i]);

if (checkBox5.Checked) chart1.Series[1].Points.AddXY(x\_1[i], y\_1[i]);

else if (checkBox6.Checked) chart1.Series[5].Points.AddXY(x\_1[i], y\_1[i]);

else if (checkBox7.Checked) chart1.Series[4].Points.AddXY(x\_1[i], y\_1[i]);

else chart1.Series[1].Points.AddXY(x\_1[i], y\_1[i]);

i++;

x4 += deltt;

} while (x4 <= 1);

if(checkBox5.Checked) chart1.Series[1].Points.AddXY(X[1], Y[1]);

else if(checkBox6.Checked) chart1.Series[5].Points.AddXY(X[1], Y[1]);

else if (checkBox7.Checked) chart1.Series[4].Points.AddXY(X[1], Y[1]);

else chart1.Series[1].Points.AddXY(X[1], Y[1]);

}

else if (X2.Count == 2)

{

//chart1.Series[1].Points.Clear();

List<double> x\_1 = new List<double>();

List<double> y\_1 = new List<double>();

double deltt = Convert.ToDouble(tbt.Text);

double x4 = deltt;

int i = 0;

if (checkBox5.Checked) chart1.Series[1].Points.AddXY(X[0], Y[0]);

else if (checkBox6.Checked) chart1.Series[5].Points.AddXY(X[0], Y[0]);

else if (checkBox7.Checked) chart1.Series[4].Points.AddXY(X[0], Y[0]);

else chart1.Series[1].Points.AddXY(X[0], Y[0]);

do

{

x\_1.Add(Math.Pow(1 - x4, 3) \* X[0] + 3 \* Math.Pow(1 - x4,2) \* x4 \* X2[0] + 3 \* (1 - x4) \* x4 \* x4 \* X2[1] + x4 \* x4 \* x4 \* X[1]);

y\_1.Add(Math.Pow(1 - x4, 3) \* Y[0] + 3 \* Math.Pow(1 - x4, 2) \* x4 \* Y2[0] + 3 \* (1 - x4) \* x4 \* x4 \* Y2[1] + x4 \* x4 \* x4 \* Y[1]);

//chart1.Series[1].Points.AddXY(x\_1[i], y\_1[i]);

if (checkBox5.Checked) chart1.Series[1].Points.AddXY(x\_1[i], y\_1[i]);

else if (checkBox6.Checked) chart1.Series[5].Points.AddXY(x\_1[i], y\_1[i]);

else if (checkBox7.Checked) chart1.Series[4].Points.AddXY(x\_1[i], y\_1[i]);

else chart1.Series[1].Points.AddXY(x\_1[i], y\_1[i]);

i++;

x4 += deltt;

} while (x4 <= 1);

if (checkBox5.Checked) chart1.Series[1].Points.AddXY(X[1], Y[1]);

else if (checkBox6.Checked) chart1.Series[5].Points.AddXY(X[1], Y[1]);

else if (checkBox7.Checked) chart1.Series[4].Points.AddXY(X[1], Y[1]);

else chart1.Series[1].Points.AddXY(X[1], Y[1]);

}

//else MessageBox.Show("Eror");

}

private void checkBox7\_CheckedChanged(object sender, EventArgs e)

{

if (checkBox7.Checked == false) chart1.Series[4].Points.Clear();

}

private void radioButton1\_CheckedChanged(object sender, EventArgs e)

{

if (radioButton1.Checked)

{

chart1.Series[3].BorderWidth = 5;

if (X2.Count == 1)

{

chart1.Series[3].Points.AddXY(X[0], Y[0]);

chart1.Series[3].Points.AddXY(X[1], Y[1]);

chart1.Series[3].Points.AddXY(X2[0], Y2[0]);

chart1.Series[3].Points.AddXY(X[0], Y[0]);

}

else

{

chart1.Series[3].Points.AddXY(X[0], Y[0]);

chart1.Series[3].Points.AddXY(X2[0], Y2[0]);

chart1.Series[3].Points.AddXY(X[1], Y[1]);

chart1.Series[3].Points.AddXY(X2[1], Y2[1]);

chart1.Series[3].Points.AddXY(X[0], Y[0]);

}

chart1.Series[2].Color = Color.White;

}

}

private void radioButton2\_CheckedChanged(object sender, EventArgs e)

{

if (radioButton2.Checked)

{

chart1.Series[2].Color = Color.Black;

chart1.Series[3].Points.Clear();

}

}

private void radioButton3\_CheckedChanged(object sender, EventArgs e)

{

if (radioButton3.Checked)

{

chart1.Series[3].Points.Clear();

chart1.Series[2].Color = Color.White;

}

}

private void checkBox6\_CheckedChanged(object sender, EventArgs e)

{

if(checkBox6.Checked == false) chart1.Series[5].Points.Clear();

}

private void checkBox5\_CheckedChanged(object sender, EventArgs e)

{

if(checkBox5.Checked == false) chart1.Series[1].Points.Clear();

}

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

{

Draw();

}

}

}