Код Основной программы

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Windows.Forms.DataVisualization.Charting;

namespace Temp

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void Button1\_Click(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;

try

{

var inputs = new List<string>();

foreach (var textBox in Controls.OfType<TextBox>())

{

inputs.Add(textBox.Text);

}

// проверка заполнения полей

if (inputs.Any(string.IsNullOrEmpty))

{

MessageBox.Show("Не всі поля заповнені");

}

else

{

var a = Convert.ToDouble(textBox1.Text);

var b = Convert.ToDouble(textBox2.Text);

// проверка нижней и верхней границы

if (a > b)

{

var tmp = a;

a = b;

b = tmp;

}

// підключення усіх графіків

Series graphic1 = chart1.Series["Вариант№14"];

Series graphic2 = chart1.Series["y = log(x)"];

Series graphic3 = chart1.Series["from file"];

// очистка

graphic1.Points.Clear();

graphic2.Points.Clear();

graphic3.Points.Clear();

if (checkBox1.Checked)

{

for (var i = a; i <= b; i += 0.1)

{

var x = i;

var y = Function(x);

graphic1.Points.AddXY(x, y);

}

}

if (checkBox2.Checked)

{

for (var i = a; i <= b; i += 0.1)

{

var x = i;

var y = Log(x);

graphic2.Points.AddXY(x, y);

}

}

if (checkBox3.Checked)

{

// считываем инф из файла в виде массива строк

var fileContent = File.ReadAllLines(@"c:/temp/test.txt");

// разделяем стоку х и у

var stringOfX = fileContent[0];

var stringOfY = fileContent[1];

// разделяем каждую строку через пробелы

var listOfX = stringOfX.Split(' ');

var listOfY = stringOfY.Split(' ');

// преобразование каждого х и у в double

for (var i = 0; i < listOfX.Length; i++)

{

var x = Convert.ToDouble(listOfX[i]);

var y = Convert.ToDouble(listOfY[i]);

graphic3.Points.AddXY(x, y);

}

}

}

}

catch

{

}

Cursor = Cursors.Default;

}

// індивідуальне завдання

public double Function(double x)

{

try

{

if (x == 0.0)

{

return 0;

}

else if (x > -3 && x <= 2)

{

return Math.Pow(x, -2);

}

else if (x > 5 && x < 10)

{

return Math.Pow(Math.Exp(-2 + x), 1 / 5) \* (1 / Math.Sqrt(Math.Pow(x, 2) + x + Math.Log(Math.Abs(x - 3.14))));

}

else

{

return Math.Pow(x, 2);

}

}

catch

{

return 0;

}

}

public double Log(double x)

{

return Math.Log(x);

}

private void ComboBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

// преобразование объекта типа object к типу combobox

var comboBox = (ComboBox)sender;

// достать выбраный елемент

var selectedType = comboBox.SelectedItem.ToString();

// пройтись по всем графикам

foreach (var series in chart1.Series)

{

switch (selectedType)

{

case "Line":

{

series.ChartType = SeriesChartType.Line;

break;

}

case "Point":

{

series.ChartType = SeriesChartType.Point;

break;

}

}

}

}

private void Button2\_Click(object sender, EventArgs e)

{

Application.Exit();

}

// подсоединяем кнопку генерации файла с кодом для файла

private void Button3\_Click(object sender, EventArgs e)

{

try

{

var a = Convert.ToDouble(textBox1.Text);

var b = Convert.ToDouble(textBox2.Text);

var childForm = new Form2

{

A = a,

B = b

};

childForm.Show();

}

catch

{

}

}

// подсоединяем кнопку "Про програму" з кодом

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

{

var about = new AboutProgramm();

about.Show();

}

}

}

Код (О программе)

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Drawing;

using System.Linq;

using System.Reflection;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Temp

{

partial class AboutProgramm : Form

{

public AboutProgramm()

{

InitializeComponent();

this.Text = String.Format("О программе {0}", AssemblyTitle);

this.labelProductName.Text = AssemblyProduct;

this.labelVersion.Text = String.Format("Версия {0}", AssemblyVersion);

this.labelCopyright.Text = AssemblyCopyright;

this.labelCompanyName.Text = AssemblyCompany;

this.textBoxDescription.Text = AssemblyDescription;

}

#region Методы доступа к атрибутам сборки

public string AssemblyTitle

{

get

{

object[] attributes = Assembly.GetExecutingAssembly().GetCustomAttributes(typeof(AssemblyTitleAttribute), false);

if (attributes.Length > 0)

{

AssemblyTitleAttribute titleAttribute = (AssemblyTitleAttribute)attributes[0];

if (titleAttribute.Title != "")

{

return titleAttribute.Title;

}

}

return System.IO.Path.GetFileNameWithoutExtension(Assembly.GetExecutingAssembly().CodeBase);

}

}

public string AssemblyVersion

{

get

{

return Assembly.GetExecutingAssembly().GetName().Version.ToString();

}

}

public string AssemblyDescription

{

get

{

object[] attributes = Assembly.GetExecutingAssembly().GetCustomAttributes(typeof(AssemblyDescriptionAttribute), false);

if (attributes.Length == 0)

{

return "";

}

return ((AssemblyDescriptionAttribute)attributes[0]).Description;

}

}

public string AssemblyProduct

{

get

{

object[] attributes = Assembly.GetExecutingAssembly().GetCustomAttributes(typeof(AssemblyProductAttribute), false);

if (attributes.Length == 0)

{

return "";

}

return ((AssemblyProductAttribute)attributes[0]).Product;

}

}

public string AssemblyCopyright

{

get

{

object[] attributes = Assembly.GetExecutingAssembly().GetCustomAttributes(typeof(AssemblyCopyrightAttribute), false);

if (attributes.Length == 0)

{

return "";

}

return ((AssemblyCopyrightAttribute)attributes[0]).Copyright;

}

}

public string AssemblyCompany

{

get

{

object[] attributes = Assembly.GetExecutingAssembly().GetCustomAttributes(typeof(AssemblyCompanyAttribute), false);

if (attributes.Length == 0)

{

return "";

}

return ((AssemblyCompanyAttribute)attributes[0]).Company;

}

}

#endregion

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

{

Close();

}

}

}

код генерации файла

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 Temp

{

public partial class Form2 : Form

{

public double A;

public double B;

public Form2()

{

InitializeComponent();

}

private void Button1\_Click(object sender, EventArgs e)

{

// взводимсчетчик

var random = new Random();//srand(time(0))

var inputs = new List<string>();

// собираемвсе inputs

foreach (var textBox in Controls.OfType<TextBox>())

{

inputs.Add(textBox.Text);

}

// проверяем на заполненость

if (inputs.Any(string.IsNullOrEmpty))

{

MessageBox.Show("Не всі поля заповнені");

}

else

{

var countOfPoint = Convert.ToInt32(textBox1.Text);

var lowerBorder = Convert.ToInt32(textBox2.Text);

var upperBorder = Convert.ToInt32(textBox3.Text);

if (countOfPoint < 0)

{

countOfPoint = Math.Abs(countOfPoint);

}

if (lowerBorder > upperBorder)

{

var tmp = lowerBorder;

lowerBorder = upperBorder;

upperBorder = tmp;

}

// создать пустые списки чисел для х и у

var listOfX = new List<double>();

var listOfY = new List<int>();

// считаем шаг

var step = (B - A) / (countOfPoint - 1);

// перемещаемся от нижнего значения до верхнего с шагом

for (var i = A; i <= B; i += step)

{

listOfX.Add(i);

// у - рандомное число, сгенерированное на заданом диапазоне

var y = lowerBorder + RandomNumber(random, 0, (upperBorder - lowerBorder));

listOfY.Add(y);

}

// удаляем старый файл

File.Delete(@"c:/temp/test.txt");

// создаем новый

var file = File.CreateText(@"c:/temp/test.txt");

// список х и у, разделенных пробелами

file.WriteLine(string.Join(" ", listOfX));

file.WriteLine(string.Join(" ", listOfY));

// закрыть файл для записи

file.Close();

Close();

}

}

public int RandomNumber(Random random, int min, int max)

{

// рандомное число в заданом диапазоне

return random.Next(min, max);

}

}

}