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»

Приближение функций

Лабораторная работа №4

по дисциплине «Вычислительная математики»

Выполнил студент Пшеничный Д. О.

Факультет, группа ФКФН, ПО(аб)-81

Проверил Резак Е.В.

Хабаровск – 2020г.

Задание: Реализовать

- метод наименьших квадратов;

- интерполирование по Лагранжу;

- интерполирование по Ньютону.

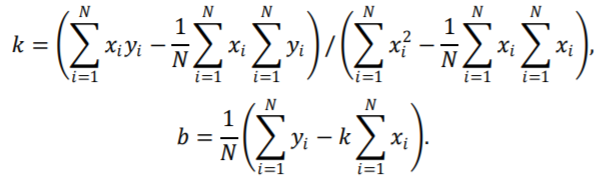
Построить графики

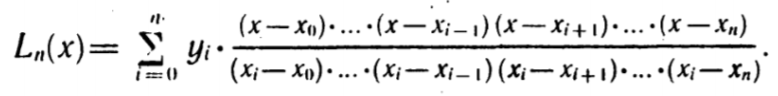
a. прямой, полученной методом наименьших квадратов;

b. интерполяционных полиномов Лагранжа и Ньютона.

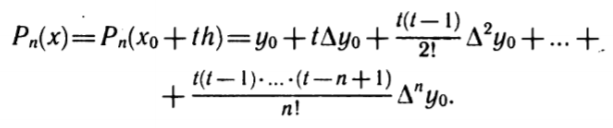
Вариант 16. Функция: 

Отрезок: 0,2:4.

МКД: f(x) = kx + b, где 

Полином Лагранжа: 

Полином Ньютона:



**Листинг**

Form1.cs

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.IO;

using System.Threading;

using System.Globalization;

using System.Windows.Forms.VisualStyles;

namespace Lab4

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

void RefreshForm()

{

panel1.Invalidate();

}

void ClearForm()

{

Main.dots.Clear();

Main.lines.Clear();

Main.curves.Clear();

Main.couples = new double[1, 1];

Main.numOfCouples = 0;

Graphic.scale = 100;

label1.Text = "Импортируйте пары значений";

RefreshForm();

}

double LagrangeCount(double x)

{

double Lx = 0;

for(int i = 0; i < Main.numOfCouples; i++)

{

double pI = Main.couples[i, 1];

for(int j = 0; j < Main.numOfCouples; j++)

{

if (j == i)

continue;

pI = pI \* (x - Main.couples[j, 0]) / (Main.couples[i, 0] - Main.couples[j, 0]);

}

Lx += pI;

}

return Lx;

}

double ThrNewton(double x)

{

double[] temp = new double[Main.numOfCouples];

double[] temp1 = new double[Main.numOfCouples];

for(int i = 0; i < Main.numOfCouples; i++)

{

temp1[i] = 1;

}

for (int i = 0; i < Main.numOfCouples; i++)

{

temp[i] = Main.couples[i, 1];

}

double res = temp[0];

int inter = 0;

int c = Main.numOfCouples - 1;

int count = 1;

for(int i = 1; i < Main.numOfCouples; i++)

{

for(int j = 0; j < c; j++)

{

temp[j] = (temp[j + 1] - temp[j]) / (Main.couples[j + 1 + inter, 0] - Main.couples[j, 0]);

}

inter++;

c--;

for(int j = 0; j < count; j++)

{

temp1[i] \*= (x - Main.couples[j, 0]);

}

count++;

temp1[i] \*= temp[0];

res += temp1[i];

}

//MessageBox.Show("x: " + x.ToString() + " y: " + res.ToString());

return res;

}

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

{

string fileText = "";

Form2 importForm = new Form2();

importForm.Owner = this;

importForm.ShowDialog();

if (importForm.DialogResult == DialogResult.OK)

{

StreamReader file;

try

{

file = new StreamReader(Main.sourcePath);

fileText = file.ReadToEnd();

}

catch

{

MessageBox.Show("Введён неверный путь к файлу!");

return;

}

file.Close();

Main.sourcePath = "";

CultureInfo temp\_culture = Thread.CurrentThread.CurrentCulture;

Thread.CurrentThread.CurrentCulture = CultureInfo.CreateSpecificCulture("en-US");

try

{

ClearForm();

string[] separators = {"\n", " ", "\t"};

string[] splittedText = fileText.Split(separators, StringSplitOptions.RemoveEmptyEntries);

//string text = "";

Main.numOfCouples = int.Parse(splittedText[0]);

Main.couples = new double[Main.numOfCouples, 2];

for (int i = 0; i < Main.numOfCouples; i++)

{

for (int j = 0; j < 2; j++)

{

//MessageBox.Show("+" + splittedText[1 + i \* 2 + j] + "+");

Main.couples[i, j] = double.Parse(splittedText[1 + i \* 2 + j]);

//text += Main.couples[i, j].ToString() + " ";

}

//text += '\n';

}

//MessageBox.Show(text);

}

catch

{

MessageBox.Show("Файл не прочитан");

}

Thread.CurrentThread.CurrentCulture = temp\_culture;

RefreshForm();

Graphic.GetDelta();

Graphic.ImportCouples();

}

}

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

{

string text = Main.numOfCouples.ToString() + " пар:\n";

for (int i = 0; i < Main.numOfCouples; i++)

{

for (int j = 0; j < 2; j++)

{

text += Main.couples[i, j].ToString() + " ";

}

text += '\n';

}

MessageBox.Show(text);

}

private void panel1\_Paint(object sender, PaintEventArgs e)

{

Graphics graphics = panel1.CreateGraphics();

Pen pen = new Pen(Color.Black, 1);

graphics.DrawLine(pen, new Point(0, 250), new Point(500, 250));

graphics.DrawLine(pen, new Point(250, 0), new Point(250, 500));

int x = 250;

int y = 250;

while(x > 0)

{

x -= Graphic.scale;

graphics.DrawLine(pen, new Point(x, 240), new Point(x, 260));

}

x = 250;

while (x < 500)

{

x += Graphic.scale;

graphics.DrawLine(pen, new Point(x, 240), new Point(x, 260));

}

while (y > 0)

{

y -= Graphic.scale;

graphics.DrawLine(pen, new Point(240, y), new Point(260, y));

}

y = 250;

while (y < 500)

{

y += Graphic.scale;

graphics.DrawLine(pen, new Point(240, y), new Point(260, y));

}

foreach (Line i in Main.lines) // Отрисовка линий

{

graphics.DrawLine(new Pen(Color.Red, 2), new Point(i.firstPoint.x, i.firstPoint.y), new Point(i.secondPoint.x, i.secondPoint.y));

}

foreach (Dot i in Main.dots) // Отрисовка точек

{

graphics.DrawRectangle(new Pen(Color.Green, 2), i.x - 1, i.y - 1, 3, 3);

}

foreach (Curve i in Main.curves) // Отрисовка кривых

{

graphics.DrawCurve(new Pen(Color.Red, 1), i.ConvertToPoints());

}

}

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

{

RefreshForm();

}

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

{

if(Main.numOfCouples == 0)

{

MessageBox.Show("Сначала импортируйте пары значений!");

return;

}

double sumX = 0;

double sumY = 0;

double sumXY = 0;

double sumXX = 0;

double k, b;

for(int i = 0; i < Main.numOfCouples; i++)

{

sumX += Main.couples[i, 0];

sumY += Main.couples[i, 1];

sumXY += Main.couples[i, 0] \* Main.couples[i, 1];

sumXX += Main.couples[i, 0] \* Main.couples[i, 0];

}

k = (sumXY - sumX \* sumY / Main.numOfCouples) / (sumXX - sumX \* sumX / Main.numOfCouples);

b = (sumY - k \* sumX) / Main.numOfCouples;

label1.Text = "k: " + k.ToString() + "\nb: " + b.ToString();

/\*MessageBox.Show(sumX.ToString() + " "

+ sumY.ToString() + " "

+ sumXY.ToString() + " "

+ sumXX.ToString() + " k: "

+ k.ToString() + " b: "

+ b.ToString());\*/

Dot dot1 = new Dot(Main.GetMinX() - 1, (Main.GetMinX() - 1) \* k + b);

Dot dot2 = new Dot(Main.GetMaxX() + 1, (Main.GetMaxX() + 1) \* k + b);

Main.lines.Add(new Line(dot1, dot2));

RefreshForm();

}

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

{

ClearForm();

}

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

{

string result = "";

int i = 1;

foreach(Dot x in Main.dots)

{

result += i.ToString() + ": " + x.x.ToString() + "/" + x.y.ToString() + "\n";

i++;

}

MessageBox.Show(result);

}

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

{

if(Main.numOfCouples == 0)

{

MessageBox.Show("Сначала импортируйте пары значений!");

return;

}

Curve curve = new Curve();

for(double i = Main.GetMinX() - 1; i < Main.GetMaxX() + 1; i += 1 / (double)Graphic.scale)

{

curve.dots.Add(new Dot(i, LagrangeCount(i)));

}

Main.curves.Add(curve);

RefreshForm();

}

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

{

if (Main.numOfCouples == 0)

{

MessageBox.Show("Сначала импортируйте пары значений!");

return;

}

Curve curve = new Curve();

for (double i = Main.GetMinX() - 1; i < Main.GetMaxX() + 1; i += 1 / (double)Graphic.scale)

{

Dot dot = new Dot(i, ThrNewton(i));

if (dot.x > 10000)

dot.x = 10000;

if (dot.x < -10000)

dot.x = -10000;

if (dot.y > 10000)

dot.y = 10000;

if (dot.y < -10000)

dot.y = -10000;

curve.dots.Add(dot);

}

Main.curves.Add(curve);

RefreshForm();

}

}

}

Main.cs

using System;

using System.Collections.Generic;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Lab4

{

static class Main

{

public static string sourcePath;

public static int numOfCouples = 0;

public static double[,] couples; // Пары значений couples[\*, 0] - x и couples[\*, 1] - y

public static double convertCoef = 1;

public static List<Dot> dots = new List<Dot>();

public static List<Line> lines = new List<Line>();

public static List<Curve> curves = new List<Curve>();

public static double GetMinX()

{

double res = couples[0, 0];

for(int i = 1; i < numOfCouples; i++)

{

if (couples[i, 0] < res)

res = couples[i, 0];

}

return res;

}

public static double GetMaxX()

{

double res = couples[0, 0];

for (int i = 1; i < numOfCouples; i++)

{

if (couples[i, 0] > res)

res = couples[i, 0];

}

return res;

}

public static int Factorial(int k)

{

if (k == 0)

return 1;

else if (k == 1)

return 1;

else if (k > 0)

{

int result = 1;

for (int i = 1; i <= k; i++)

{

result \*= i;

}

return result;

}

else

return 0;

}

}

class Dot

{

public int x, y;

public Dot(int \_x, int \_y)

{

x = \_x;

y = \_y;

}

public Dot(double \_x, double \_y)

{

\_x = \_x \* Graphic.scale;

if (\_x > 0)

{

\_x = Math.Abs(\_x) + Graphic.SCREEN\_HALF;

}

else

{

\_x += Graphic.SCREEN\_HALF;

}

\_y = \_y \* Graphic.scale;

if (\_y > 0)

{

\_y = Graphic.SCREEN\_HALF - \_y;

}

else

{

\_y = Math.Abs(\_y) + Graphic.SCREEN\_HALF;

}

x = (int)\_x;

y = (int)\_y;

}

}

class Line

{

public Dot firstPoint, secondPoint;

public Line(Dot \_firstPoint, Dot \_secondPoint)

{

firstPoint = \_firstPoint;

secondPoint = \_secondPoint;

}

public Line(int \_x1, int \_y1, int \_x2, int \_y2)

{

firstPoint = new Dot(\_x1, \_y1);

secondPoint = new Dot(\_x2, \_y2);

}

}

class Curve

{

public List<Dot> dots = new List<Dot>();

public Point[] ConvertToPoints()

{

Point[] points = new Point[dots.Count];

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

{

points[i] = new Point(dots[i].x, dots[i].y);

}

return points;

}

}

static class Graphic

{

public static int scale = 100;

public const int SCREEN\_HALF = 250;

public static void GetDelta()

{

double max = 0;

for(int i = 0; i < Main.numOfCouples; i++)

{

for(int j = 0; j < 2; j++)

{

if (Math.Abs(Main.couples[i, j]) > max)

max = Math.Abs(Main.couples[i, j]);

}

}

max = (int)max + 1;

max \*= 2;

scale = (int)(500 / max);

}

public static void ImportCouples()

{

for (int i = 0; i < Main.numOfCouples; i++)

{

Main.dots.Add(new Dot(Main.couples[i, 0], Main.couples[i, 1]));

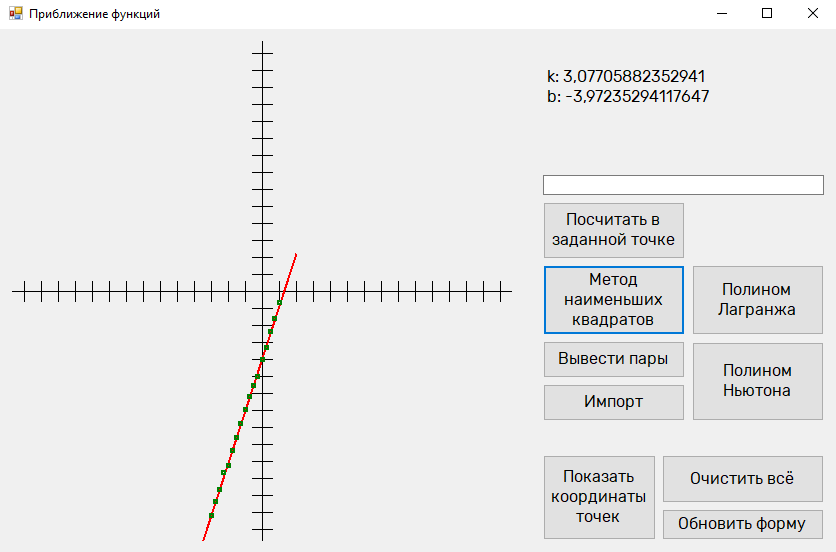
}

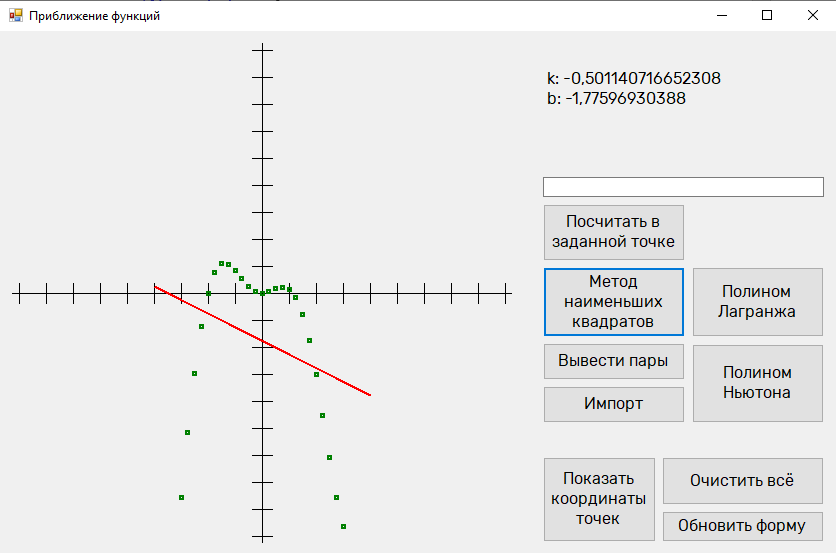
}

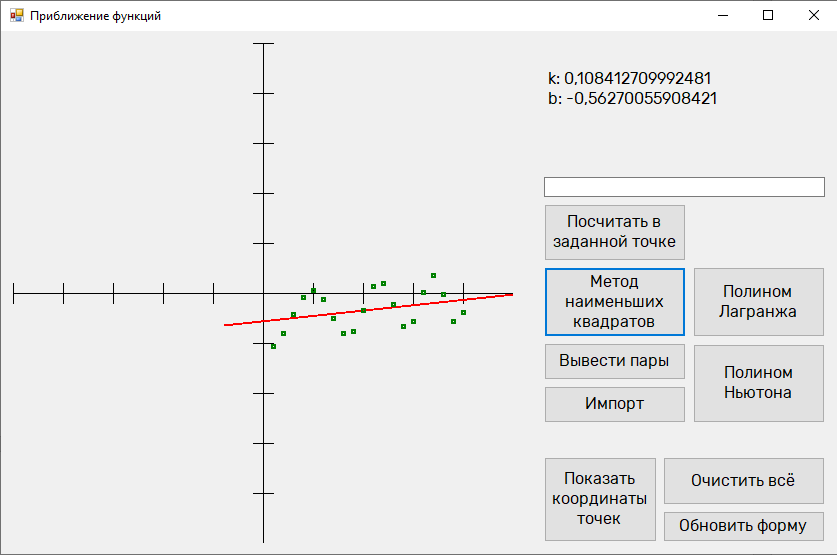
}

}

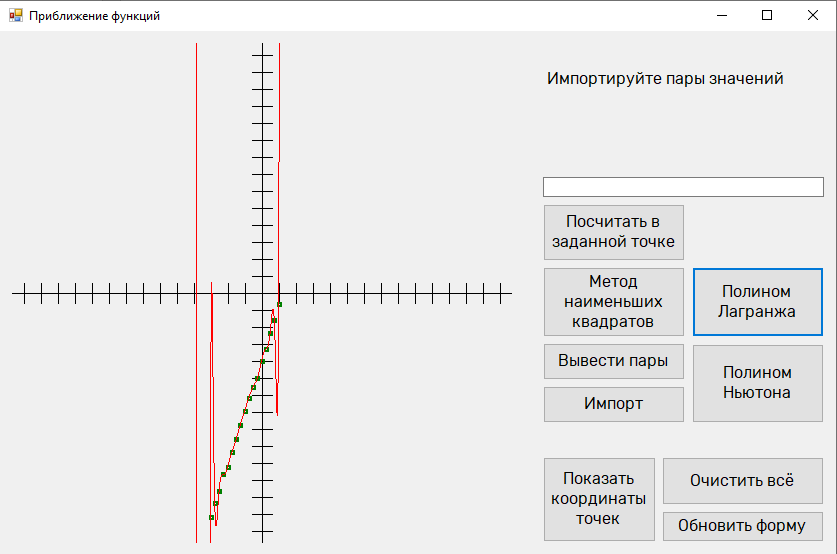
**Результат работы программы**

Результаты вычисления функций методом наименьших квадратов:

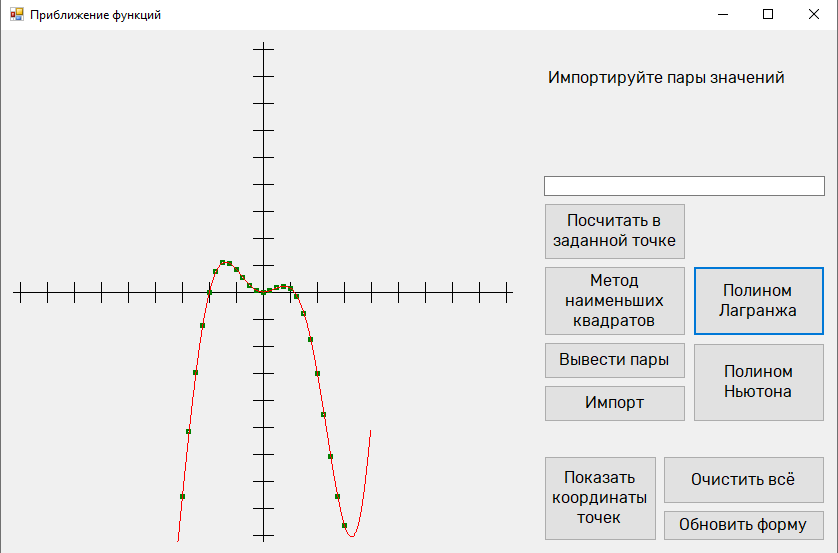
Пример 1.

Пример 2.

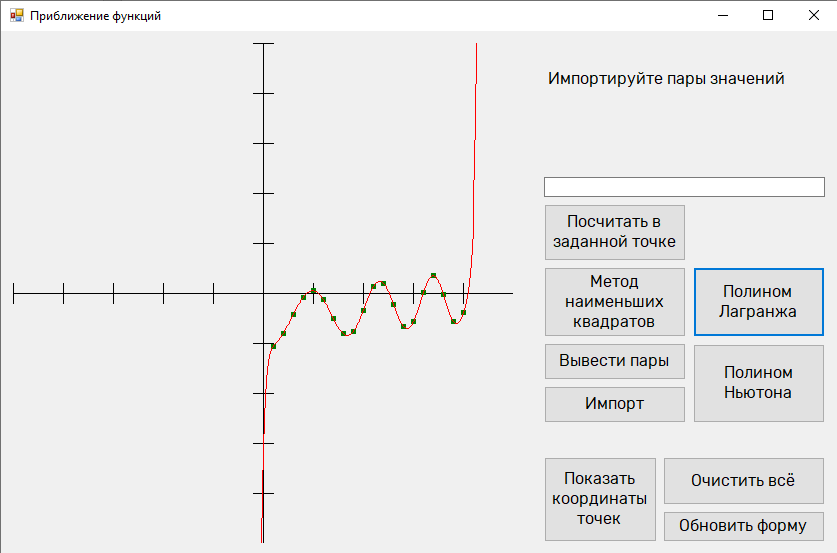
Пример 3.

 Результаты вычисления функций полином Лагранжа:

Пример 1.

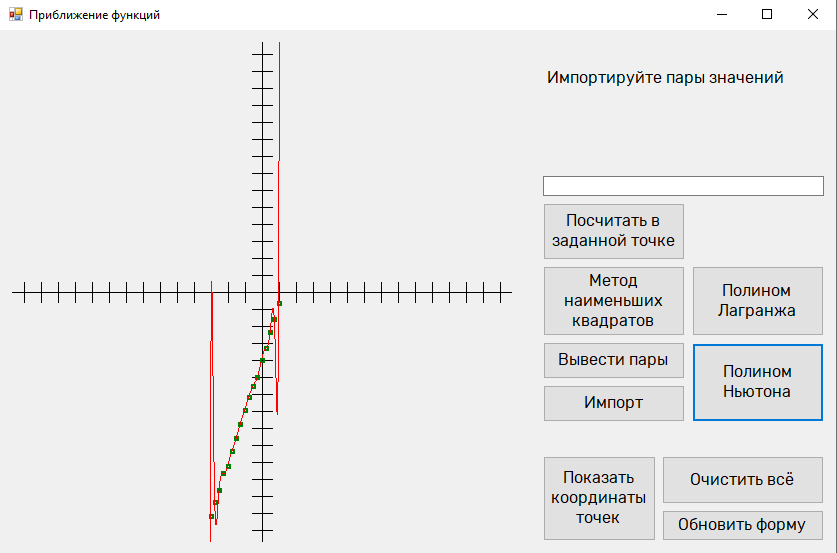


Пример 2.

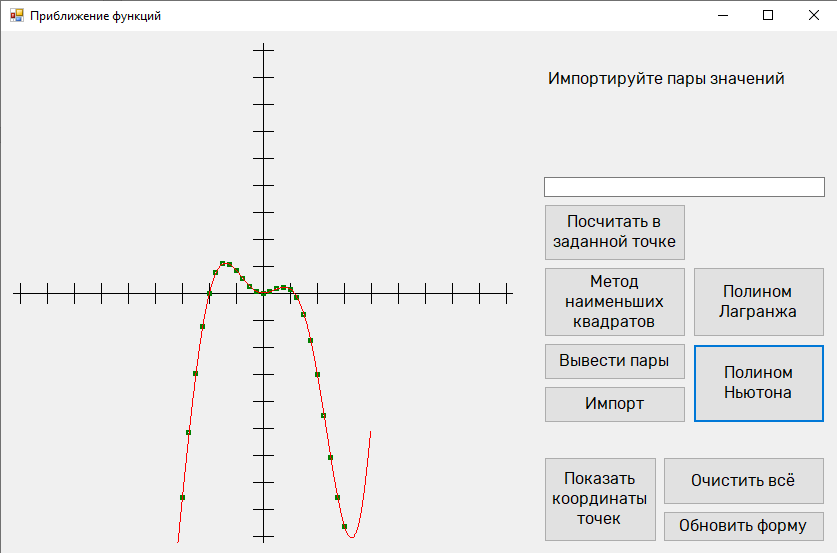


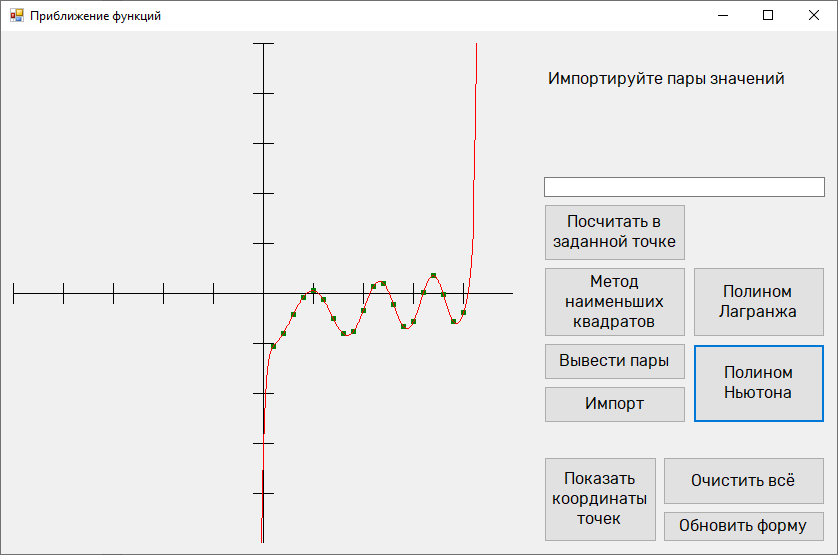
Пример 3.

Результат вычисления функций полином Ньютона:



Пример 1.



****Пример 2.

Пример 3.

Для тестового примера была выбрана точка:

Результаты: значение исходной функции f(-1.375) = -0.23701  
МКД f(-1.375) = 0.5635  
Полином Лагранжа f(-1.375) = -0.23693  
Полином Ньютона f(-1.375) = -0.23693

**Вывод**

Изучено приближение функций. Исходя из теоретической информации, можно отметить, что метод наименьших квадратов необходимо использовать только для линейных функций, тогда как полином Лагранжа и Ньютона одинаково точно и лучше вычисляют более сложные функции. На основе теоретических данных была написана программа, результаты которой совпали с тестовым примером.