МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

Федеральное государственное бюджетное

образовательное учреждение

высшего образования

«ВЯТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ФАКУЛЬТЕТ КОМПЬЮТЕРНЫХ И ФИЗИКО-МАТЕМАТИЧЕСКИХ НАУК

КАФЕДРА ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ

Отчёт по лабораторной работе №1

по дисциплине «Математические модели в естествознании и методы их исследования»

Исследование динамики популяций в зависимости от

параметров модели и начальных условий

Выполнил: студент группы ФИб-4302-51-00 \_\_\_\_\_\_\_\_\_\_\_ / Савин Д. А./

Проверил: к. б. н., доцент каф. ПМиИ \_\_\_\_\_\_\_\_\_\_\_/ Чупраков П. Г./

Киров 2021

# Цель работы

**Цель работы:** приобретение навыков самостоятельного решения прикладной задачи математического моделирования путем применения технологии вычислительного эксперимента.

**План:**

1. Выписать уравнения динамики популяций для 2 видов в соответствии со своим вариантом (указан в таблице 1).
2. Аппроксимировать полученные уравнения с использованием методов Рунге-Кутты.
3. Варьируя величины параметров модели, провести ряд вычислительных экспериментов.
4. Проанализировать результаты вычислительных экспериментов, сделать выводы.

**Решение**

Пример 1

(1)

(2)

– константы собственной скорости роста видов,

– константы самоограничения численности (внутривидовой конкуренции),

– константы взаимодействия видов.

Изображение выглядит как текст, кроссворд

Автоматически созданное описание

Вариант 9. Хищник-Жертва.

Уравнения Хищник-Жертва:

(1)

(2)

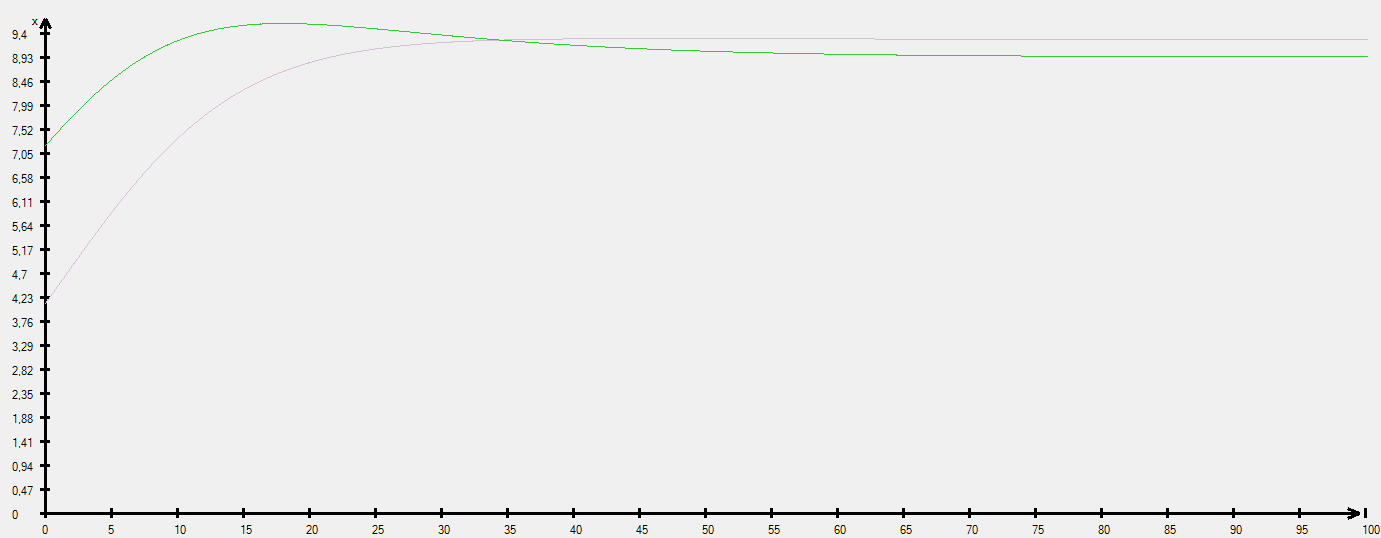
Начальные данные:

Уравнения:

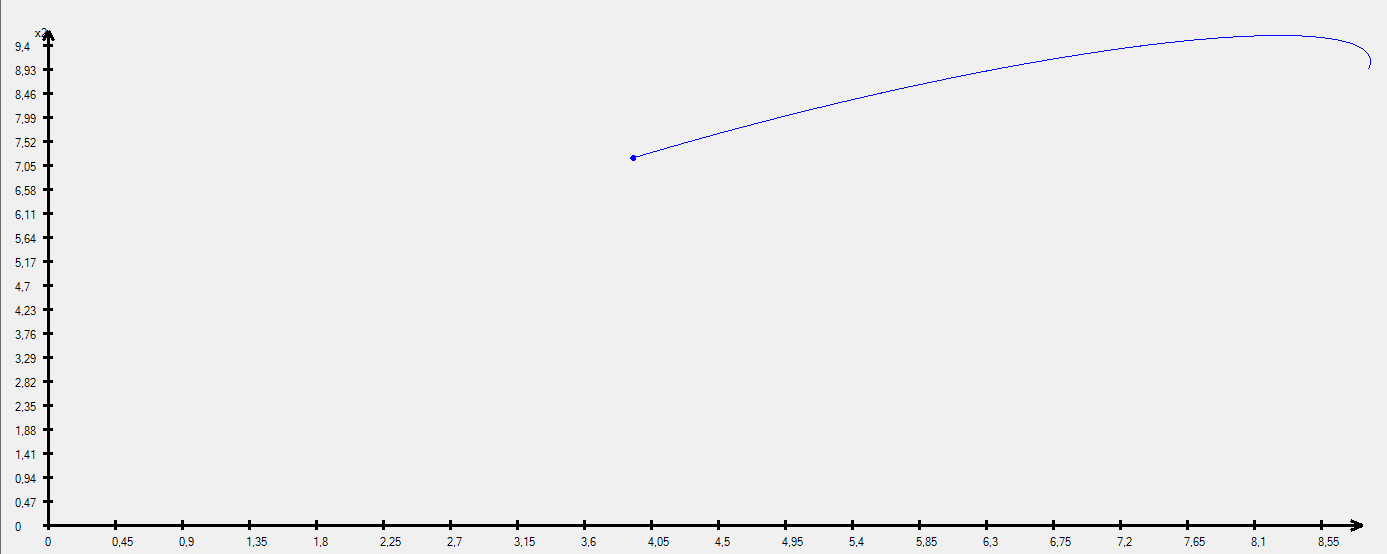
(1)

(2)

Графики:



График, x1=фиолетовый, x2=зеленый, x – тысяч особей, t – годы



Фазовый портрет, круг – начальная точка, x1, x2 – тысяч особей

Результат:

При данных параметрах популяции сначала растут, затем рост одной из популяции снижается и рост прекращается.

Виды сосуществуют.

Пример 2

Начальные данные:

Уравнения:

(1)

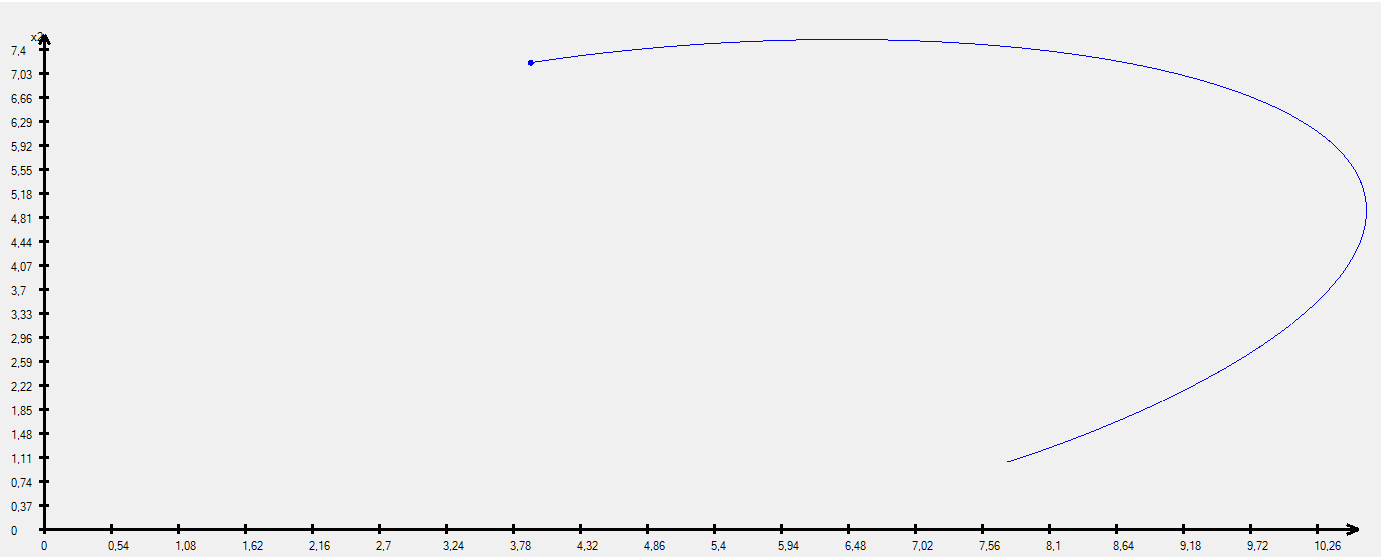
(2)

Графики:

Изображение выглядит как текст, небо, цветной

Автоматически созданное описание

График, x1=фиолетовый, x2=зеленый, x – тысяч особей, t – годы



Фазовый портрет, круг – начальная точка, x1, x2 – тысяч особей

Результат:

Рост популяций сначала растет, но рост одного вида привел к уменьшению численности второго вида, оба вида сосуществуют.

Пример 3

Начальные данные:

Уравнения:

(1)

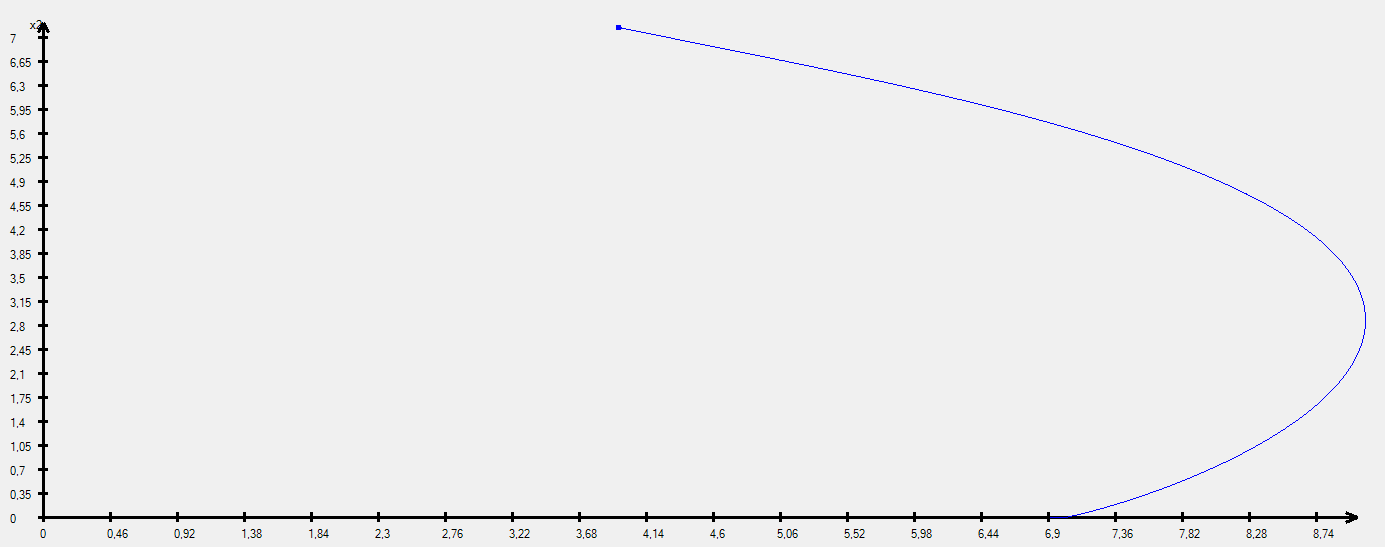
(2)

Графики:

Изображение выглядит как текст, небо, цветной

Автоматически созданное описание

График, x1=фиолетовый, x2=зеленый, x – тысяч особей, t – годы



Фазовый портрет, круг – начальная точка, x1, x2 – тысяч особей

Результат:

Рост первого вида привел к вымиранию второго.

Пример 4

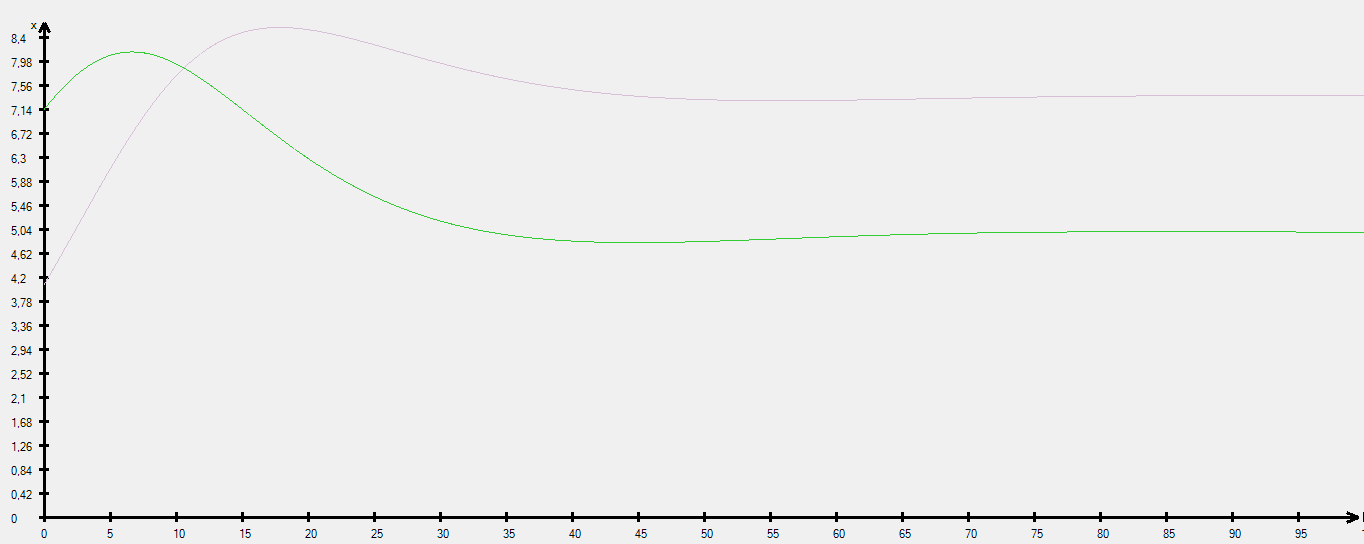
Начальные данные:

Уравнения:

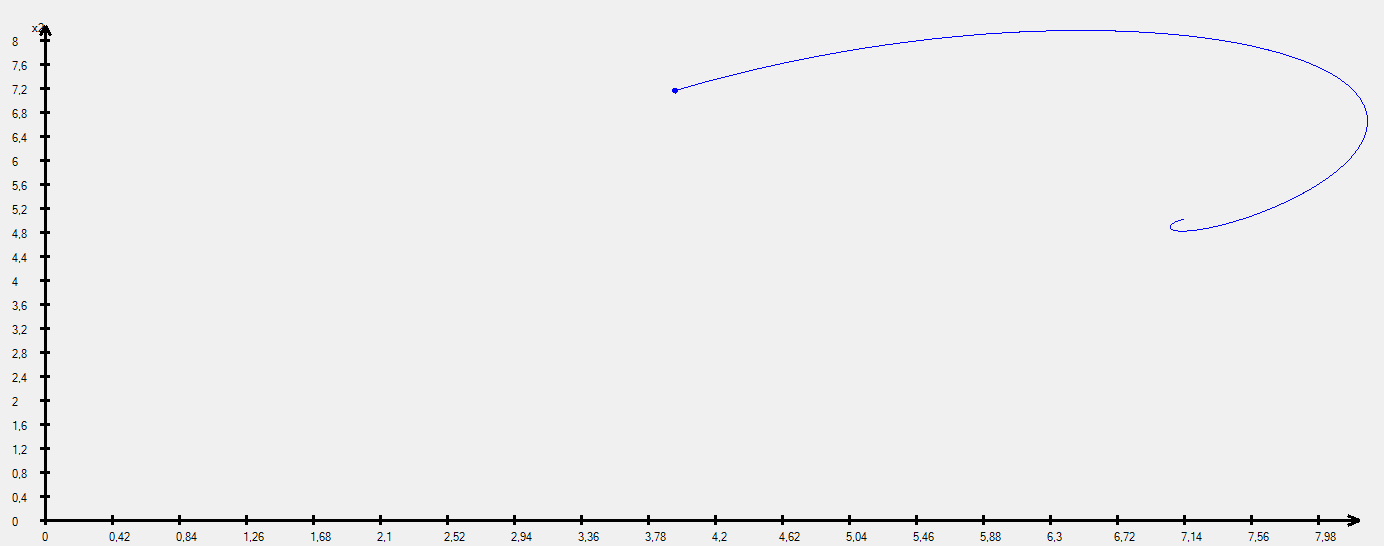
(1)

(2)

Графики:



График, x1=фиолетовый, x2=зеленый, x – тысяч особей, t – годы



Фазовый портрет, круг – начальная точка, x1, x2 – тысяч особей

Результат:

Сначала популяции росли, но из-за роста первой, вторая популяция начала умирать, вследствие чего первая тоже начала умирать, но потом популяции стабилизировались.

**Листинг 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;

namespace Lab\_1

{

public partial class Form1 : Form

{

private double Function0(double x, double[] y)

{

return C[0, 0] \* y[0] + C[0, 1] \* y[0] \* y[1] - C[0, 2] \* y[0] \* y[0];

}

private double Function1(double x, double[] y)

{

return C[1, 0] \* y[1] + C[1, 1] \* y[1] \* y[0] - C[1, 2] \* y[1] \* y[1];

}

double[,] C;

bool next = false;

public Form1()

{

InitializeComponent();

C = new double[2, 3];

}

private void button2\_Click(object sender, EventArgs e)

{

next = true;

button1\_Click(sender, e);

}

private void button1\_Click(object sender, EventArgs e)

{

try

{

double[] populations = { 4, 7 };

double t = 0;

double h = 0.0001;

List<double[]> xValue = new List<double[]>();

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

double maxt = 0, maxx1 = 0, maxx2 = 0;

int X0, Y0;

X0 = 35;

Y0 = pictureBox2.Height - 25;

Stream myStream;

if (openFileDialog1.ShowDialog() == DialogResult.OK)

{

if ((myStream = openFileDialog1.OpenFile()) != null)

{

StreamReader streamReader = new StreamReader(myStream);

string line;

int i = 0;

while (!streamReader.EndOfStream)

{

line = streamReader.ReadLine();

string[] s = line.Split(' ');

for (int j = 0; j < s.Length; j++)

{

C[i, j] = double.Parse(s[j]);

}

i++;

}

streamReader.Close();

}

myStream.Close();

}

richTextBox1.Clear();

richTextBox1.AppendText("1) dx1/dt = " + C[0, 0] + "x1 + " + C[0, 1] + "x1x2 - " + C[0, 2] + "x1x1\n");

richTextBox1.AppendText("2) dx1/dt = " + C[1, 0] + "x2 + " + C[1, 1] + "x2x1 - " + C[1, 2] + "x2x2\n");

for (int i = 0; t < 100; t += 150 \* h, i++)

{

xValue.Add(new double[2]);

xValue[i][0] = populations[0];

xValue[i][1] = populations[1];

maxx1 = Math.Max(maxx1, populations[0]);

maxx2 = Math.Max(maxx2, populations[1]);

tValue.Add(t);

populations = Runge\_kutt4(populations, h, t);

maxt = t;

}

Graphics graph = pictureBox2.CreateGraphics();

Pen myPen = new Pen(Brushes.Black, 3);

Pen myPen1 = new Pen(Brushes.Thistle, 1);

Pen myPen2 = new Pen(Brushes.LimeGreen, 1);

int x = (pictureBox2.Width - 50) / 20;

int y = (pictureBox2.Height - 50) / 20;

int Y = Y0, X = X0;

if (!next)

{

graph.Clear(BackColor);

graph.DrawLine(myPen, X0, Y0, pictureBox2.Width - 25, Y0);

graph.DrawLine(myPen, X0, Y0, X0, 23);

for (int i = 0; i <= 20; i++)

{

graph.DrawLine(myPen, X0 - 5, Y, X0 + 5, Y);

graph.DrawLine(myPen, X, Y0 - 5, X, Y0 + 5);

graph.DrawString((Math.Round(maxt / 20, 2) \* i).ToString(), Font, Brushes.Black, (float)(X - 5), Y0 + 10);

graph.DrawString((Math.Round(Math.Max(maxx1, maxx2) / 20, 2) \* i).ToString(), Font, Brushes.Black, X0 - 35, (float)(Y - 5));

X += x;

Y -= y;

}

graph.DrawString("t", Font, Brushes.Black, (float)(X - 10), Y0 + 5);

graph.DrawString("x", Font, Brushes.Black, X0 - 15, (float)(Y + 5));

graph.DrawLine(myPen, pictureBox2.Width - 35, Y0 + 5, pictureBox2.Width - 23, Y0);

graph.DrawLine(myPen, pictureBox2.Width - 35, Y0 - 5, pictureBox2.Width - 23, Y0);

graph.DrawLine(myPen, X0 - 5, 30, X0, 20);

graph.DrawLine(myPen, X0 + 5, 30, X0, 20);

for (int i = 0; i < tValue.Count - 1; i++)

{

graph.DrawLine(myPen1, X0 + (float)(tValue[i] / maxt \* (pictureBox2.Width - 50)), Y0 - (float)(xValue[i][0] / Math.Max(maxx1, maxx2) \* (pictureBox2.Height - 50)),

X0 + (float)(tValue[i + 1] / maxt \* (pictureBox2.Width - 50)), Y0 - (float)(xValue[i + 1][0] / Math.Max(maxx1, maxx2) \* (pictureBox2.Height - 50)));

graph.DrawLine(myPen2, X0 + (float)(tValue[i] / maxt \* (pictureBox2.Width - 50)), Y0 - (float)(xValue[i][1] / Math.Max(maxx1, maxx2) \* (pictureBox2.Height - 50)),

X0 + (float)(tValue[i + 1] / maxt \* (pictureBox2.Width - 50)), Y0 - (float)(xValue[i + 1][1] / Math.Max(maxx1, maxx2) \* (pictureBox2.Height - 50)));

}

}

if (next)

{

next = false;

graph = pictureBox2.CreateGraphics();

graph.Clear(BackColor);

graph.DrawLine(myPen, X0, Y0, pictureBox2.Width - 25, Y0);

graph.DrawLine(myPen, X0, Y0, X0, 23);

x = (pictureBox2.Width - 30) / 20;

y = (pictureBox2.Height - 50) / 20;

Y = Y0;

X = X0;

for (int i = 0; i <= 20; i++)

{

graph.DrawLine(myPen, X0 - 5, Y, X0 + 5, Y);

graph.DrawLine(myPen, X, Y0 - 5, X, Y0 + 5);

graph.DrawString((Math.Round(maxx1 / 20, 2) \* i).ToString(), Font, Brushes.Black, (float)(X - 5), Y0 + 10);

graph.DrawString((Math.Round(maxx2 / 20, 2) \* i).ToString(), Font, Brushes.Black, X0 - 35, (float)(Y - 5));

X += x;

Y -= y;

}

graph.DrawString("x1", Font, Brushes.Black, (float)(X - 50), Y0 - 5);

graph.DrawString("x2", Font, Brushes.Black, X0 - 15, (float)(Y + 5));

graph.DrawLine(myPen, pictureBox2.Width - 35, Y0 + 5, pictureBox2.Width - 23, Y0);

graph.DrawLine(myPen, pictureBox2.Width - 35, Y0 - 5, pictureBox2.Width - 23, Y0);

graph.DrawLine(myPen, X0 - 5, 30, X0, 20);

graph.DrawLine(myPen, X0 + 5, 30, X0, 20);

myPen1 = new Pen(Brushes.Blue, 1);

graph.FillEllipse(Brushes.Blue, X0 + (float)(xValue[0][0] / maxx1 \* (pictureBox2.Width - 50)) - 3, Y0 - (float)(xValue[0][1] / maxx2 \* (pictureBox2.Height - 50)) - 3, 6, 6);

for (int i = 0; i < xValue.Count - 1; i++)

{

graph.DrawLine(myPen1, X0 + (float)(xValue[i][0] / maxx1 \* (pictureBox2.Width - 50)), Y0 - (float)(xValue[i][1] / maxx2 \* (pictureBox2.Height - 50)),

X0 + (float)(xValue[i + 1][0] / maxx1 \* (pictureBox2.Width - 50)), Y0 - (float)(xValue[i + 1][1] / maxx2 \* (pictureBox2.Height - 50)));

}

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

private double[] Runge\_kutt4(double[] x, double h, double t)

{

double[] results = new double[2];

double k1, k2, k3, k4, m1, m2, m3, m4;

k1 = h \* Function0(t, x);

m1 = h \* Function1(t, x);

k2 = h \* Function0(t + h / 2, new double[2] { x[0] + k1 / 2, x[1] + m1 / 2 });

m2 = h \* Function1(t + h / 2, new double[2] { x[0] + k1 / 2, x[1] + m1 / 2 });

k3 = h \* Function0(t + h / 2, new double[2] { x[0] + k2 / 2, x[1] + m2 / 2 });

m3 = h \* Function0(t + h / 2, new double[2] { x[0] + k2 / 2, x[1] + m2 / 2 });

k4 = h \* Function1(t + h, new double[2] { x[0] + k3, x[1] + m1 });

m4 = h \* Function1(t + h, new double[2] { x[0] + k3, x[1] + m1 });

results[0] = Math.Max(x[0] + (k1 + 2 \* k2 + 2 \* k3 + k4) / 6, 0);

results[1] = Math.Max(x[1] + (m1 + 2 \* m2 + 2 \* m3 + m4) / 6, 0);

return results;

}

}

}