**МИНОБРНАУКИ РОССИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕНЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.Шухова»**

**(БГТУ им. В.Г.Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа №

По дисциплине: Системное моделирование

Тема: “ Движение механических систем”

Выполнил: Руденький А.О.

Проверил: Полунин А.И.

Белгород 2019

**Цель работы:** Получение теоретических знаний о принципах моделирования физических систем с помощью метода сил и моментов.

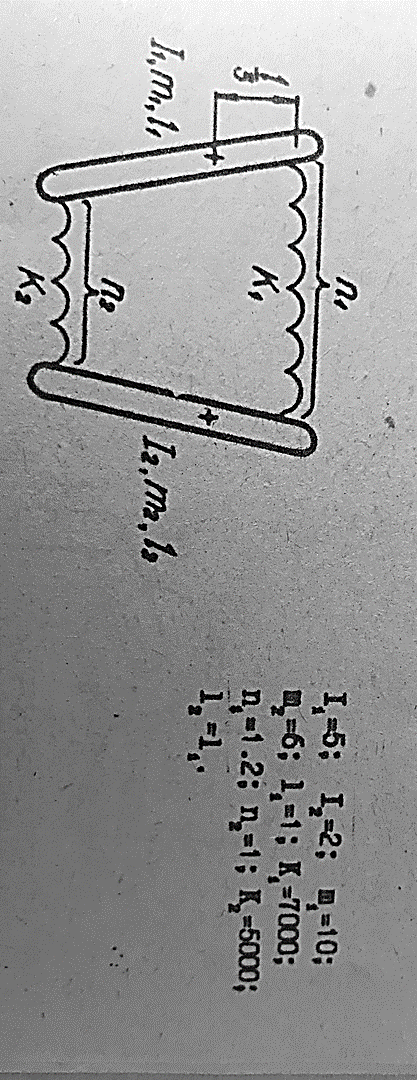
**Задания к лабораторной работе**

1. Изучить физическую систему, соответствующую варианту.

2. Составить дифференциальные уравнения, описывающих движение системы.

3. Запрограммировать численный метод Рунге-Кутта 4 порядка и построить графики.

**Задание.**

[](https://vk.com/photo184073163_456240017)

В данной вариантом системе можно наблюдать две степени свободы. Уравнения для левой и правой балок написаны в угловой системе координат с использованием метода сил и моментов:

Уравнения моментов для левой (первой) балки:

момент силы тяжести: сила тяжести:

момент первой (верхней) пружины: сила первой пружины:

момент второй (нижней) пружины: сила второй пружины:

Уравнения моментов для правой (второй) балки:

момент силы тяжести: сила тяжести:

момент первой (верхней) пружины: сила первой пружины:

момент второй (нижней) пружины: сила второй пружины:

Окончательные уравнения физической системы соответствующего варианта:

Рунге-кутт для системы из 4-ёх уравнений

Код программы:

using LiveCharts;

using LiveCharts.Wpf;

using System;

using System.Collections.Generic;

using System.Threading;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Input;

using System.Windows.Media;

using LiveCharts.Charts;

namespace WpfApp6

{

public partial class MainWindow : Window

{

bool paused=true;

struct Result

{

public double fi1, fi2, w1, w2;

}

double f1(double t, double fi1, double fi2)

{

return (l \* (-3 \* gravity \* m1 \* fi1 + 8 \* k2 \* l \* (fi1 - fi2) + 2 \* k1 \* l \* (-fi1 + fi2))) / (18 \* y1);

}

double f2(double t, double fi1, double fi2)

{

return (l \* (8 \* k2 \* l \* (fi1 - fi2) - 3 \* gravity \* m2 \* fi2 + 2 \* k1 \* l \* (-fi1 + fi2))) / (18 \* y2);

}

double f3(double t, double w1)

{

return w1;

}

double f4(double t, double w2)

{

return w2;

}

Result RungeKutt(double t, double fi1, double fi2, double w1, double w2, double h) //Рунге кутт для системы из 4-ёх уравнений

{

double k11 = f1(t, fi1, fi2) \* h;

double k12 = f2(t, fi1, fi2) \* h;

double k13 = f3(t, w1) \* h;

double k14 = f4(t, w2) \* h;

double k21 = f1(t + h / 2, fi1 + k11 / 2, fi2 + k12 / 2) \* h;

double k22 = f2(t + h / 2, fi1 + k11 / 2, fi2 + k12 / 2) \* h;

double k23 = f3(t + h / 2, w1 + k13 / 2) \* h;

double k24 = f4(t + h / 2, w2 + k14 / 2) \* h;

double k31 = f1(t + h / 2, fi1 + k21 / 2, fi2 + k22 / 2) \* h;

double k32 = f2(t + h / 2, fi1 + k21 / 2, fi2 + k22 / 2) \* h;

double k33 = f3(t + h / 2, w1 + k23 / 2) \* h;

double k34 = f4(t + h / 2, w2 + k24 / 2) \* h;

double k41 = f1(t + h, fi1 + k31, fi2 + k32) \* h;

double k42 = f2(t + h, fi1 + k31, fi2 + k32) \* h;

double k43 = f3(t + h, fi1 + k33) \* h;

double k44 = f4(t + h, fi1 + k34) \* h;

Result result;

result.fi1 = fi1 + (k11 + 2 \* k21 + 2 \* k31 + k41) / 6;

result.fi2 = fi2 + (k12 + 2 \* k22 + 2 \* k32 + k42) / 6;

result.w1 = w1 + (k11 + 2 \* k21 + 2 \* k31 + k41) / 6;

result.w2 = w2 + (k12 + 2 \* k22 + 2 \* k32 + k42) / 6;

return result;

}

double gravity = 9.8, l = 0, m1 = 0, m2 = 0, k1 = 0, k2 = 0, y1 = 0, y2 = 0;

private void Canvas\_MouseDown(object sender, MouseButtonEventArgs e)

{

Pause.IsEnabled = true;

Start.IsEnabled = false;

fi1.IsEnabled = false;

fi2.IsEnabled = false;

w1.IsEnabled = false;

w2.IsEnabled = false;

paused = false;

}

private void Canvas\_MouseDown\_1(object sender, MouseButtonEventArgs e)

{

Pause.IsEnabled = false;

Start.IsEnabled = true;

paused = true;

}

string NiceTimeAxis(double t)

{

return (Math.Round(t, 10).ToString());

}

ChartValues<double> container1 = new ChartValues<double>(), container2 = new ChartValues<double>();

List<string> timecontainer = new List<string>();

public MainWindow()

{

InitializeComponent();

mychart.AxisX.Add(new Axis

{

Title = "t, сек.",

Labels = timecontainer,

LabelFormatter = NiceTimeAxis

});

mychart.AxisY.Add(new Axis

{

Title = "Угол, фи",

});

mychart.LegendLocation = LegendLocation.Right;

}

void recompute()

{

if (mychart.Series != null) mychart.Series.Clear();

container1.Clear(); container2.Clear(); timecontainer.Clear();

l = double.Parse(L.Text.Replace(".", ","));

m1 = double.Parse(M1.Text.Replace(".", ","));

m2 = double.Parse(M2.Text.Replace(".", ","));

k1 = double.Parse(K1.Text.Replace(".", ","));

k2 = double.Parse(K2.Text.Replace(".", ","));

y1 = double.Parse(Y1.Text.Replace(".", ","));

y2 = double.Parse(Y2.Text.Replace(".", ","));

Result a;

a.fi1 = double.Parse(fi1.Text.Replace(".", ","));

a.fi2 = double.Parse(fi2.Text.Replace(".", ","));

a.w1 = double.Parse(w1.Text.Replace(".", ","));

a.w2 = double.Parse(w2.Text.Replace(".", ","));

for (double t = 0; t <= double.Parse(T.Text.Replace(".", ",")); t = t + 1)

{

a = RungeKutt(t, a.fi1, a.fi2, a.w1, a.w2, 0.0001);

container1.Add(a.fi1);

container2.Add(a.fi2);

timecontainer.Add(t.ToString());

Console.WriteLine("{1} {2}", t, a.fi1, a.fi2);

}

mychart.Series = new SeriesCollection

{

new LineSeries

{

Title = "Фи1",

Values = container1

},

new LineSeries

{

Title = "Фи2",

Values = container2

}

};

}

private void Button\_Click(object sender, RoutedEventArgs e)

{

Pause.IsEnabled = true;

Start.IsEnabled = false;

fi1.IsEnabled = false;

fi2.IsEnabled = false;

w1.IsEnabled = false;

w2.IsEnabled = false;

do{ recompute(); }while(!paused);

}

}

}