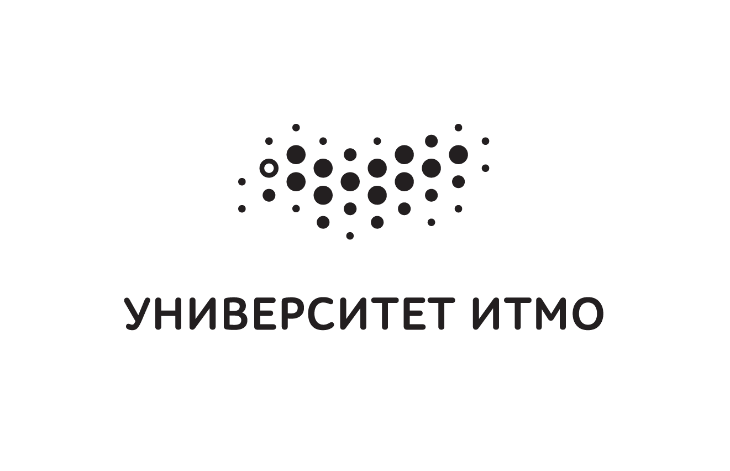
Санкт-Петербургский Национальный Исследовательский Университет Информационных Технологий, Механики и Оптики

ФКТиУ, кафедра Вычислительной техники

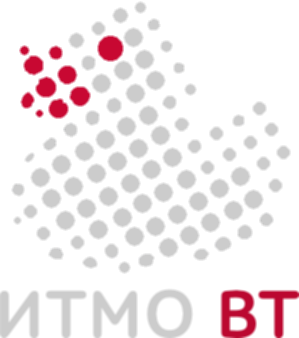


(Метод Рунге-Кутты четвёртого порядка)

Лабораторная работа №4

по дисциплине

«Вычислительная математика»

Выполнил: Студент группы P3210

Заводов Андрей Павлович

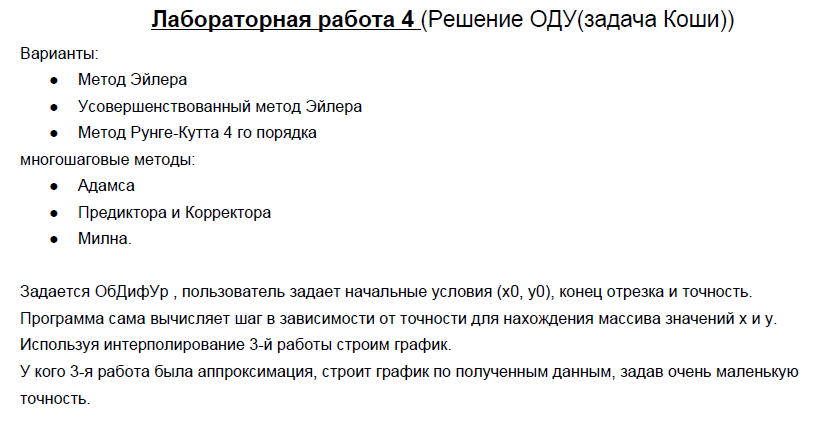
Преподаватель: ассистент кафедры ИПМ

Петрова Мария Максимовна

Санкт-Петербург

2017 г.

**Задание (Метод Рунге-Кутты четвёртого порядка)**



**Листинг**

(Pragram.cs) (методы для отрисовки)

public void DrawGraph()

{

//получение контекста и установка параметров

graph = panel1.CreateGraphics();

Pen dashed\_pen = new Pen(Brushes.DarkGray, 1);

dashed\_pen.DashStyle = System.Drawing.Drawing2D.DashStyle.Dash;

// получаем масштаб

scale = trackBar1.Value;

graph.FillRectangle(new SolidBrush(Color.Black), X0Y0.X, X0Y0.Y, 3, 3);

//Отрисовка делений

drawSplits(scale);

//отрисовка графика

if (checkBox1.Checked)

try

{

pointsTrue = GetTrueFunc(scale);

//отрисовка эталоного графика sin (x)

Pen line\_true = new Pen(Color.Green);

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

{

graph.DrawLine(line\_true, new Point(pointsTrue[i].X, pointsTrue[i].Y),

new Point(pointsTrue[i + 1].X, pointsTrue[i + 1].Y));

}

}

catch (Exception exc) { MessageBox.Show("Ошибка при отрисовке эталонной функции"); }

if (checkBox2.Checked)

try

{

pointsSpl = GetSplimeFunc(scale);

//отрисовка графика по заданным точкам

Pen line\_splime = new Pen(Color.Red);

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

{

graph.DrawLine(line\_splime, new Point(pointsSpl[i].X, pointsSpl[i].Y),

new Point(pointsSpl[i + 1].X, pointsSpl[i + 1].Y));

}

}

catch (Exception exc) { MessageBox.Show("Что-то пошло не так при отрисовке графика.\nСкорее всего вы накосячили со вводом("); }

if (checkBox3.Checked)

try

{

//отрисовка точек

Pen lineB = new Pen(Color.Blue);

int radius = 4;

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

{

graph.DrawEllipse(lineB, X0Y0.X + (float)(x[i] \* 30 \* scale) - radius / 2,

X0Y0.Y - (float)(y[i] \* 10 \* scale) - radius / 2, radius, radius);

}

}

catch (Exception exc){ MessageBox.Show("Что-то пошло не так при отрисовке точек.\nСкорее всего вы накосячили со вводом("); }

}

private List<Point> GetTrueFunc(int scale)

{

List<Point> points = new List<Point>();

for (double i = 0; i < end; i+=accur) {

int X = X0Y0.X + (int)(i\*30\*scale);

int Y = X0Y0.Y - (int)(Runge\_Kutt.Function(i , 0)\*10\*scale);

points.Add(new Point(X, Y));

}

return points;

}

private List<Point> GetSplimeFunc(int scale)

{

// Методом Рунге-Кутты находим точки

double[,] arr = Runge\_Kutt.Runge(x0, y0, end, accur);

//и заполняем ими массив для интерполирования

int n = arr.Length / 2; //так как хранятся и X и Y

x.Clear(); y.Clear();

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

{

x.Add(arr[0, i]);

y.Add(arr[1, i]);

}

//получаем интерполированную функцию

spl.GetSplines(x, y, x.Count);

List<Point> points = new List<Point>();

//по найденной функции ищем точки для построения

for (double i = x0; i < end; i += accur)

{

int X = X0Y0.X + (int)(i \* 30 \* scale);

int Y = X0Y0.Y - (int)(spl.GetInterpolateY(i) \* 10 \* scale);

points.Add(new Point(X, Y));

}

return points;

}

private void panel1\_Resize(object sender, EventArgs e)

{

X0Y0 = new Point(0, panel1.Height / 2);

panel1.Refresh();

DrawGraph();

}

//расчет значения функции

private void button5\_Click(object sender, EventArgs e)

{

double \_x;

double \_y;

if (! Double.TryParse(textBox1.Text,out \_x))

MessageBox.Show("Некорректный ввод");

else

{

spl.GetSplines(x, y, x.Count);

\_y = spl.GetInterpolateY(\_x);

label3.Text = "Y = " + \_y;

Pen lineB = new Pen(Color.Blue);

int radius = 4;

graph.DrawEllipse(lineB, X0Y0.X + (float)(\_x \* 30 \* scale) - radius / 2,

X0Y0.Y - (float)(\_y \* 10 \* scale) - radius / 2, radius, radius);

}

}

/\*\*

\* Отрисовка делений на осях

\* \*/

private void drawSplits(int scale)

{

Pen axis\_pen = new Pen(Brushes.DarkGray, 1);

//Строим ось Х

graph.DrawLine(axis\_pen, new Point(0, X0Y0.Y), new Point(panel1.Width, X0Y0.Y));

graph.DrawString("X", font, new SolidBrush(Color.Black), panel1.Width - 14, X0Y0.Y - 14);

//Строим ось Y

graph.DrawLine(axis\_pen, new Point(X0Y0.X, 0), new Point(X0Y0.X, panel1.Height));

graph.DrawString("Y", font, new SolidBrush(Color.Black), X0Y0.X + 10, 0);

Pen line = new Pen(Color.Gray);

int c = 0;

for (int i = panel1.Height / 2; i > 20; i -= 10 \* scale)

{

graph.DrawLine(line, X0Y0.X, i, X0Y0.X + 4, i);

graph.DrawString("" + c, font1, new SolidBrush(Color.Black), X0Y0.X + 11, i - 5);

c++;

}

c = 0;

for (int i = panel1.Height / 2; i < panel1.Height; i += 10 \* scale)

{

graph.DrawLine(line, X0Y0.X, i, X0Y0.X + 4, i);

graph.DrawString("-" + c, font1, new SolidBrush(Color.Black), X0Y0.X + 11, i - 5);

c++;

}

c = 0;

for (int i = 0; i < panel1.Width - 20; i += 30 \* scale)

{

graph.DrawLine(line, i, X0Y0.Y, i, X0Y0.Y - 7);

graph.DrawString("" + c, font1, new SolidBrush(Color.Black), i , X0Y0.Y - 12);

c++;

}

}

/\*\*

\* Входная точка

\*/

private void draw\_graph\_button\_Click(object sender, EventArgs e)

{

try { x0 = Convert.ToDouble(textBox2.Text); }

catch { MessageBox.Show("Ошибка ввода X0!"); return; }

try { y0 = Convert.ToDouble(textBox3.Text); }

catch { MessageBox.Show("Ошибка ввода Y0!"); return; }

try { end = Convert.ToDouble(textBox4.Text); }

catch { MessageBox.Show("Ошибка ввода x\_end!"); return; }

try { accur = Convert.ToDouble(textBox5.Text); }

catch { MessageBox.Show("Ошибка ввода точности!"); return; }

panel1.Refresh();

DrawGraph();

}

}

}

(Runge-Kutt.cs)

public static class Runge\_Kutt

{

// реализация метода Рунге - Кутта

public static double[,] Runge(double x0, double y0, double end, double accur)

{

double k1 = 0, k2 = 0, k3 = 0, k4 = 0;

double step = accur;

int n = (int)((end - x0) / step) + 1;

double[,] points = new double[2, n];

points[0, 0] = x0;

points[1, 0] = y0;

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

{

k1 = Function(x0, y0);

k2 = Function(x0 + step / 2, y0 + step \* k1 /2);

k3 = Function(x0 + step / 2, y0 + step \* k2 / 2);

k4 = Function(x0 + step, y0 + step \* k3);

x0 += step;

y0 = y0 + step \* (k1 + 2\*k2 + 2\*k3 + k4) / 6;

points[0, i] = x0;

points[1, i] = y0;

}

return points;

}

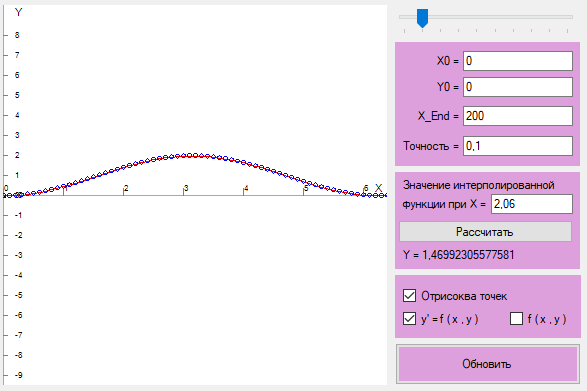
public static double Function(double x, double y)

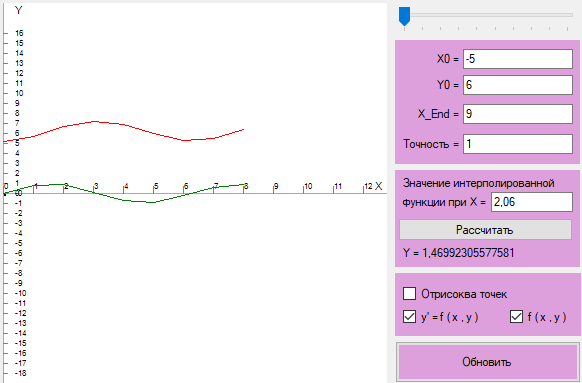
{

return Math.Sin(x);

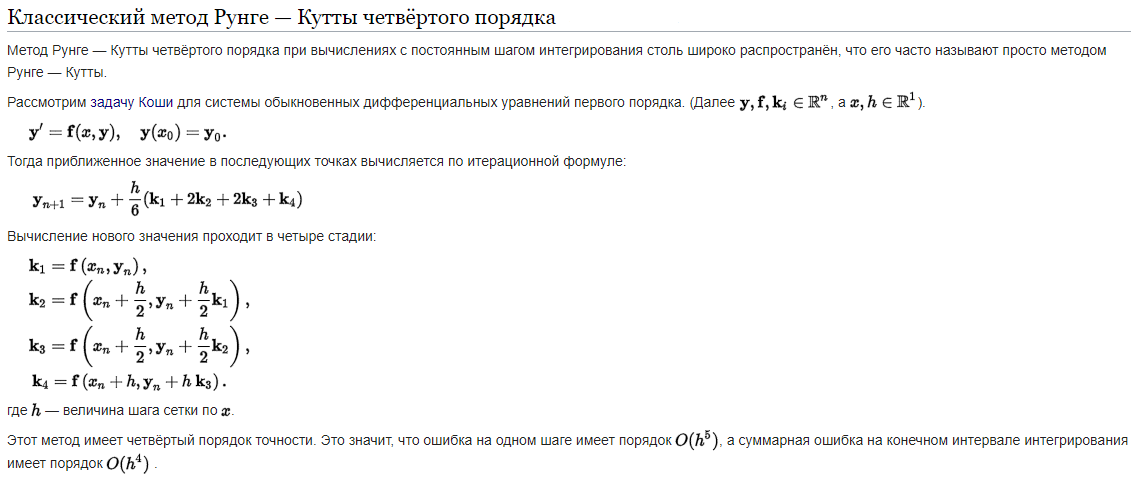
}

**Тестовые данные**

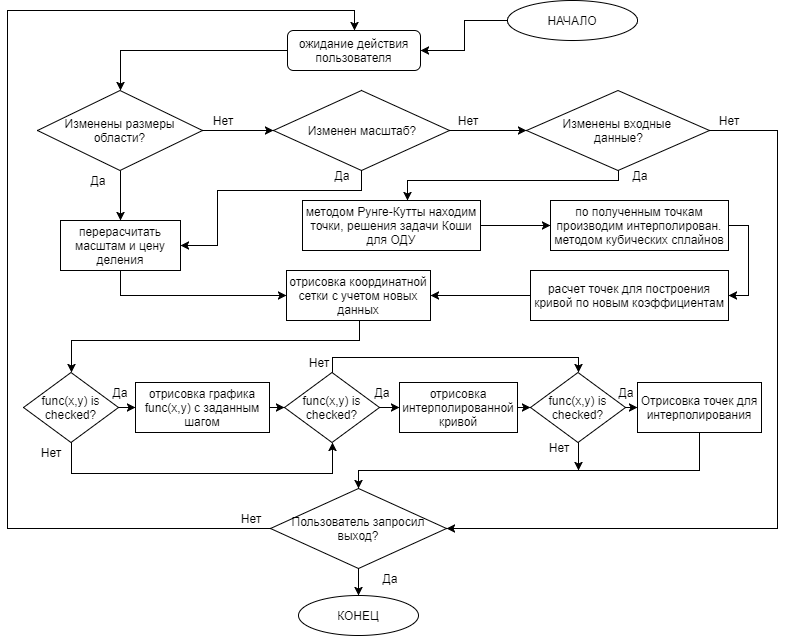




**Теория**



**Блок-схема**



**Вывод**

В ходе выполнения работы был изучен метод Рунге-Кутты для нахождения решения задачи Коши обыкновенных дифференциальных уравнений при заданной точности. А также построен график с использованием материалов 3 лабораторной.