

## **Функциональные возможности программного комплекса**

### **1.1 Описание программного комплекса**

Программный комплекс лабораторных работ разработанный в рамках моей работы представляет из себя Windows Form приложение, включающее в себя семь лабораторных работ по предмету «Численные методы».

Включены следующие численные методы, распределенные по лабораторным работам:

1. Одношаговые методы решения задачи Коши: метод Эйлера, метод Эйлера-Коши и метод Рунге-Кутты 4-го порядка.
2. Многошаговые методы решения задачи Коши: метод Адамса (явный)
3. Решение жестких систем ОДУ: метод Гира, метод Ракитского (матричной экспоненты).
4. Численное дифференцирование: дифференцирование с помощью сплайнов.
5. Численное интегрирование: формула прямоугольников, формула трапеций, формула Симпсона, формула Гаусса.
6. Приближенное вычисление преобразования Фурье.

В конечном результате должен получится комплекс шести лабораторных работ, по выше описанным численным методам.

Каждая отдельная лабораторная работа содержится в своем классе (lab1 – лабораторная №1, lab2 – лабораторная №2 и т.д.). Все лабораторные работы выполнены в графическом интерфейсе. Исключением являются некоторые из лабораторных, которым требуется изменять исходный код для изменения входных данных. Программа может использоваться, как основа для математических вычислений отдельно взятых приложений.

## 1.2 Описание графического интерфейса

Лабораторная работа №8-9. Обе лабораторные работы исследуют численные методы решения задачи Коши. В восьмой лабораторной работе рассмотрены одношаговые методы, а в девятой многошаговые. Графический интерфейс содержит в себе таблицу вывода значений, которые получаются в процессе решения задачи Коши (рис. 3.1).



Рис. 3.1 – Графический интерфейс 8-9 лабораторной работы

По умолчанию в программе выставлен отрезок исследования от 0 до 3 с шагом 0.1. Метод Эйлера, Эйлера-Коши и Рунге-Кутты 4-го порядка относятся к восьмой лабораторной работе. Метод Адамса – девятая лабораторная работа.

Лабораторная работа №10. Лабораторная работа исследует решение жестких систем ОДУ. Графический интерфейс содержит в себе таблицу вывода значений, которые получаются в процессе решения системы (рис. 3.2).

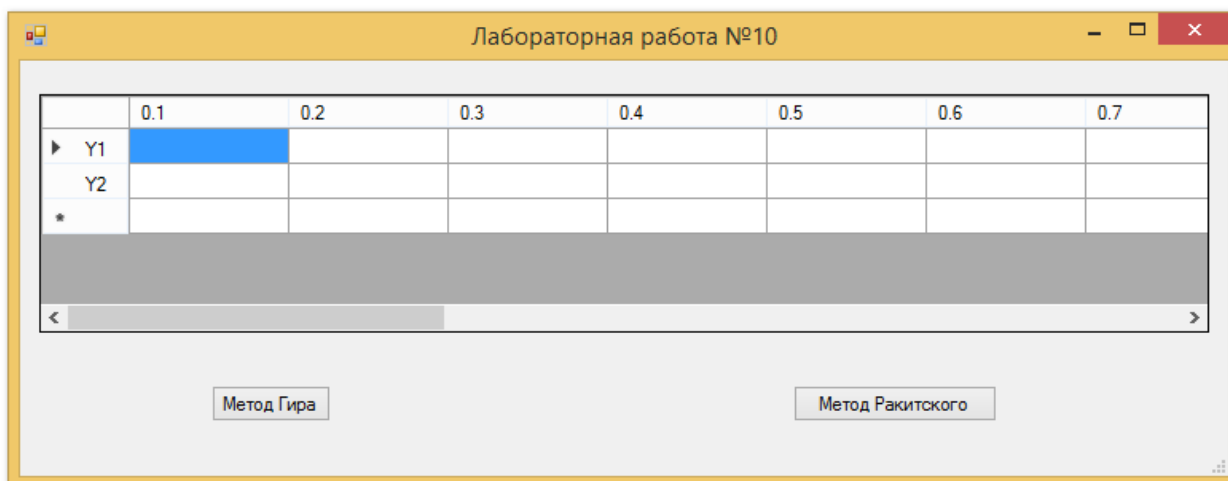


Рис. 3.2 – Графический интерфейс 10 лабораторной работы

По умолчанию в программе выставлен отрезок исследования от 0 до 2 с шагом 0.1.

Лабораторная работа №11. Лабораторная работа исследует численное дифференцирование с помощью сплайнов. Графический интерфейс содержит в себе элемент, который способен отображать график  $X$  от  $Y$ . Так же `radioButton`'ы выбора исследуемой функции (рис. 3.3).

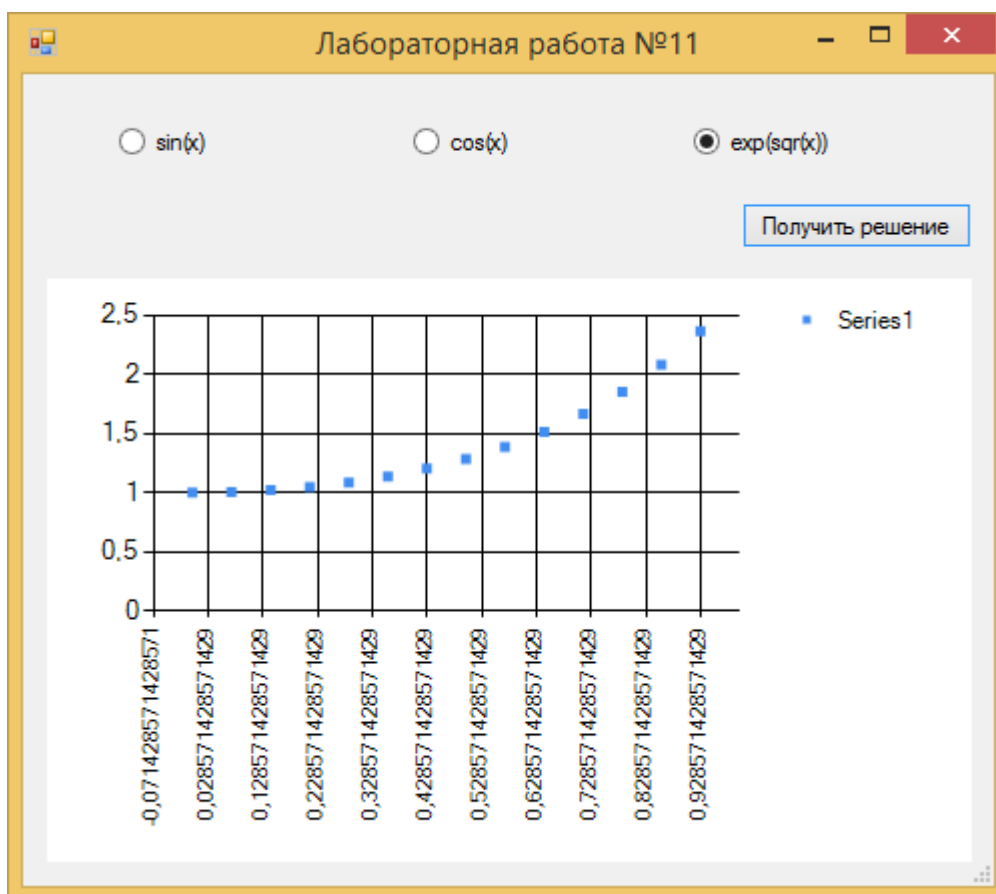


Рис. 3.3 – Графический интерфейс 11 лабораторной работы

По умолчанию выставлен отрезок исследования от 0 до 1.

Лабораторная работа №12. Лабораторная работа исследует численное интегрирование при помощи формулы прямоугольников, трапеций, метода Симпсона и метода Гаусса. Графический интерфейс содержит в себе исследуемую функцию, кнопки вызова решения по каждому методу и текстовое поле для вывода решения (рис. 3.4).

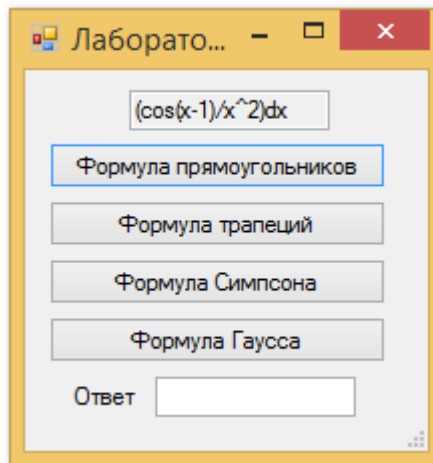


Рис. 3.4 – Графический интерфейс 12 лабораторной работы

По умолчанию отрезок исследования от 0 до 1.

Лабораторная работа №13. Лабораторная работа исследует приближенное вычисление преобразования Фурье. Графический интерфейс содержит в себе таблицу с значениями омега, действительной и мнимой частью (рис. 3.5).

	Омега	Действительная часть	Мнимая часть
▶ 0	-0,314159265358979	0	0
1	0	0,905758175452758	0
2	0,314159265358979	0,883188903385773	-0,159902331512389
3	0,628318530717959	0,818776052565571	-0,304265210885333
4	0,942477796076938	0,72161798631475	-0,420508152978322
5	1,25663706143592	0,604515104518074	-0,501138670637758
6	1,5707963267949	0,481221191298883	-0,544526518373231
7	1,88495559215388	0,36385626184419	-0,554270592170324
8	2,19911485751286	0,261131422412628	-0,53754325059413

Рис. 3.5 – Графический интерфейс 13 лабораторной работы

Рассмотрены графические интерфейсы семи лабораторных работ.

### 1.3 Примеры решения численных методов

Лабораторная работа №8-9. Рассмотрим процесс работы лабораторной на примере решения задачи Коши, указанной на рис. 3.6.

$$\begin{cases} dy_1 / dx = y_2 \\ dy_2 / dx = e^{-xy_1} \end{cases} \quad \begin{cases} y_1(0) = 0 \\ y_2(0) = 0 \end{cases}$$

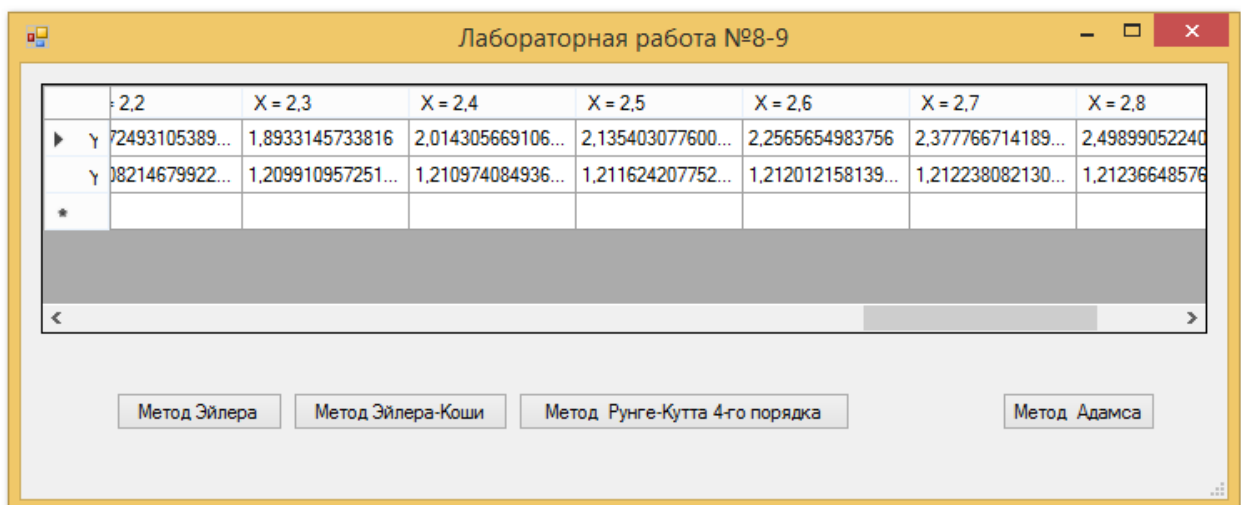
Рис. 3.6 – Задача Коши

Требуется решить на отрезке [0,3] с шагом 0,1 задачу Коши. Введем данные в код программы. Изменим код функций f1 и f2 (рис. 3.7).

```
double SQRN(double a, double b)
{
    if (a != 0) return Math.Exp(b*Math.Log(a));
    return 0;
}
double f1(double x, double y1, double y2)
{
    return y2;
}
double f2(double x, double y1, double y2)
{
    return SQRN(2.71828182846, -x * y1);
}
```

Рис. 3.7 – Содержимое функций

Выберем один из методов решения задачи, например, Эйлера и получим решение (рис. 3.8).



	2,2	X = 2,3	X = 2,4	X = 2,5	X = 2,6	X = 2,7	X = 2,8
► y	72493105389...	1,8933145733816	2,014305669106...	2,135403077600...	2,2565654983756	2,377766714189...	2,49899052240
γ	08214679922...	1,209910957251...	1,210974084936...	1,211624207752...	1,212012158139...	1,212238082130...	1,21236648576
*							

Метод Эйлера    Метод Эйлера-Коши    Метод Рунге-Кутты 4-го порядка    Метод Адамса

Рис. 3.8 – Решение задачи Коши

Все методы выдают результат сопоставимый с погрешностью 0.001.

Лабораторная работа №10. Рассмотрим процесс работы лабораторной на примере решения жесткой системы ОДУ, указанной на рис. 3.9.

$$\begin{cases} y_1' = -11y_1 + 9y_2 \\ y_2' = 9y_1 - 11y_2 \end{cases}$$

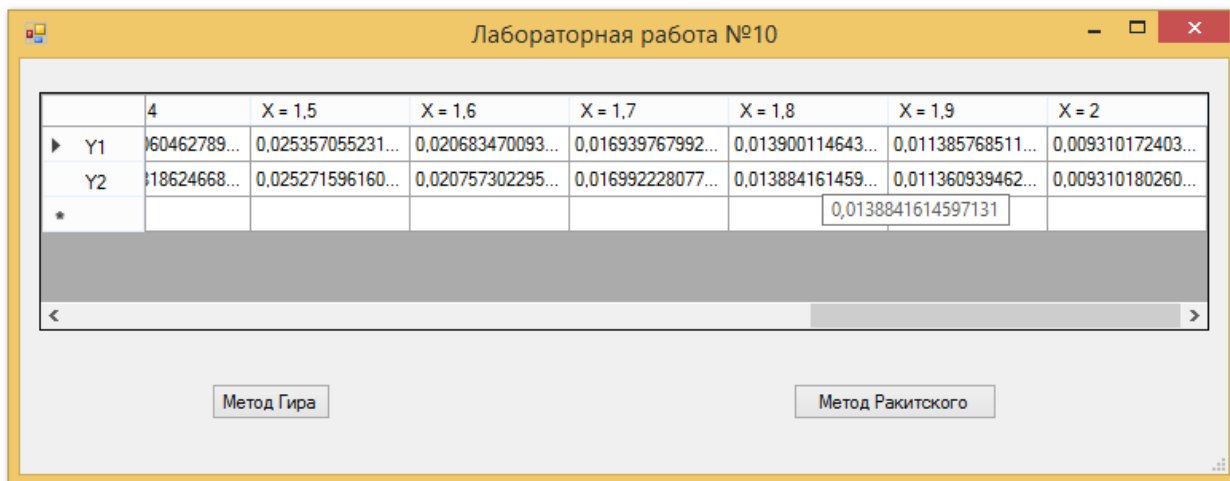
Рис. 3.9 – Жесткая система ОДУ

Введем данные в код программы. Для этого в функциях f1 и f2 введем значения, указанные на рис. 3.10.

```
double f1(double x, double y1, double y2)
{
    return -11 * y1 + 9 * y2;
}
double f2(double x, double y1, double y2)
{
    return 9 * y1 - 11 * y2;
}
```

Рис. 3.10 – Исходные данные задачи

Выберем один из методов решения задачи, например, Гира и получим решение (рис. 3.11).



	4	X = 1,5	X = 1,6	X = 1,7	X = 1,8	X = 1,9	X = 2
Y1	60462789...	0,025357055231...	0,020683470093...	0,016939767992...	0,013900114643...	0,011385768511...	0,009310172403...
Y2	18624668...	0,025271596160...	0,020757302295...	0,016992228077...	0,013884161459...	0,011360939462...	0,009310180260...
*						0,0138841614597131	

Рис. 3.11 – Решение задачи Коши

Все методы выдают результат сопоставимый с погрешностью 0.001.

Лабораторная работа №11. Рассмотрим процесс работы лабораторной на примере численного дифференцирования функции, указанной на рис. 3.12.

$$e^{x^2}$$

Рис. 3.12 – Функция, которую требуется продифференцировать

Введем данные в код программы. Для этого в функциях  $f$ ,  $fx$  и  $fx$  введем функцию, первую производную и вторую производную соответственно, указанные на рис. 3.13.

```
double f(double x)
{
    if (radioButton1.Checked) { return Math.Sin(x); }
    if (radioButton2.Checked) { return Math.Cos(x); }
    if (radioButton3.Checked) { return Math.Exp(Math.Pow(x, 2)); }
    return 0;
}
double fx(double x)
{
    if (radioButton1.Checked) { return Math.Cos(x); }
    if (radioButton2.Checked) { return -Math.Sin(x); }
    if (radioButton3.Checked) { return 2 * x * Math.Exp(Math.Pow(x, 2)); }
    return 0;
}
double fxx(double x)
{
    if (radioButton1.Checked) { return -Math.Sin(x); }
    if (radioButton2.Checked) { return -Math.Cos(x); }
    if (radioButton3.Checked) { return 2 * Math.Exp(Math.Pow(x, 2)) * (1 + 2 * x * x); }
    return 0;
}
```

Рис. 3.13 – Исходные данные

Нажмем кнопку «получить решение». В результате получим график производной функции (рис. 3.14).

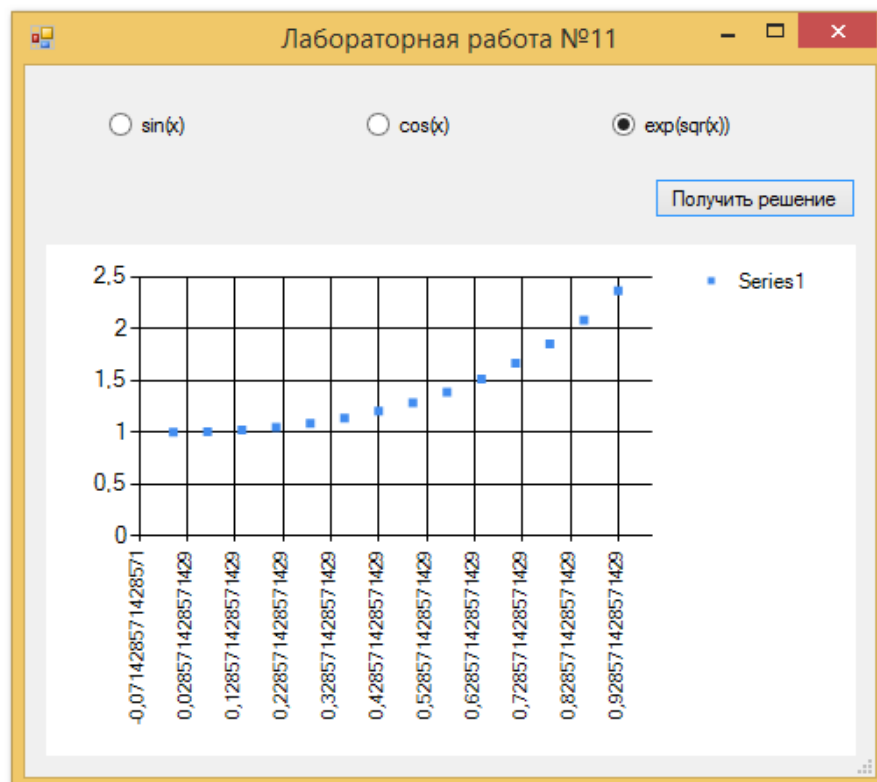


Рис. 3.14 – график производной функции

Лабораторная работа №12. Рассмотрим процесс работы лабораторной на примере численного интегрирования функции, указанной на рис. 3.15

$$\int_0^1 \frac{\cos x - 1}{x^2} dx$$

Рис. 3.15 – Исходный интеграл

Введем данные в код программы. Для этого в функции f введем функцию, как указано на рис. 3.16.

```
double f(double x)
{
    double result = -0.5;
    if (x > 1e-8)
    {
        result = (Math.Cos(x) - 1) / (x * x);
    }
    return result;
}
```

Рис. 3.16 – Исходные данные

Получим решение одним из методов, например прямоугольников, как на рисю 3.17.

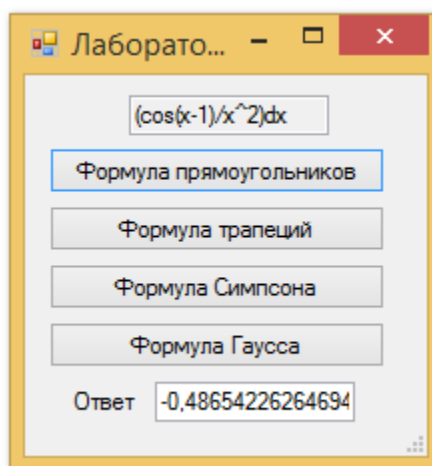


Рис. 3.17 – Решение интеграла

Лабораторная работа №13. Рассмотрим процесс работы лабораторной на примере нахождения приближенного вычисления преобразования Фурье, указанного на рис. 3.18.



**Пример.** Вычислить  $\int_a^b e^{-x^2} e^{i\omega x} dx$ , используя квадратурную формулу с 512 узлами (при  $\omega = \omega_k = \frac{2\pi k}{b-a}$ ,  $k = 0, 1, \dots, n-1, n = 512$ .)

Рис. 3.18 – Задание для решения

Введем данные в код программы. Для этого в конструкторе Lab13 введем функцию, как указано на рис. 3.19.

```
double x = (i - 1) * h;  
ar[i] = Math.Exp(-x * x);  
ai[i] = 0;
```

Рис. 3.19 – Исходные данные

Запустим программу и получим результат на 512 узлах, как и указано в задании (рис. 3.20).

	Омега	Действительная часть	Мнимая часть
504	158,0221104755...	-0,00204693200...	-0,00821457965...
505	158,3362697409...	-0,00282031200...	-0,00798467842...
506	158,6504290062...	-0,00361964347...	-0,00745109456...
507	158,9645882716...	-0,00431450112...	-0,00657237856...
508	159,2787475370...	-0,00474577111...	-0,00537008400...
509	159,5929068023...	-0,00475996145...	-0,00394597378...
510	159,90706606772	-0,00425315119...	-0,00247843705...
511	160,2212253330...	-0,00321088648...	-0,00119319690...
*			

Рис. 3.20 – Решение преобразования Фурье

## Заключение

Разработка программного обеспечения – актуальность этого направления сейчас очень велика. Количество языков программирования и операционных систем, под которые пишут программное обеспечение великое множество, поэтому практикуясь разрабатывать программное обеспечение в рамках бакалаврской работы я обобщаю знания, полученные мною в рамках обучения по специальности «Информатика и вычислительная техника».

В ходе выполнения бакалаврской работы, на тему «Программирование численных методов. Часть2», было проделано следующее:

1) проведен теоретический обзор по технологии разработки программного обеспечения. Это сделано для расширения кругозора и подготовки мыслительного процесса по проектированию своей системы. В результате стали понятны цели и задачи, которые требуется поставить перед собой;

2) разработанная система подходит для применения в лабораторном комплексе по предмету «Численные методы» на кафедре ПОУТС;

3) в качестве средства разработки системы была выбрана MS Visual Studio, а язык программирования C#;

4) была проведена верификация данных полученных старым исходным кодом и новым, результаты совпали;

5) разработан графический интерфейс, который позволил отказаться от «консоли» и сконцентрировать внимание пользователя на результатах;

6) запрограммированы численные методы с восьмой по тринадцатую лабораторную работу;

7) описаны основные возможности программного обеспечения, такие как примеры решения численных методов и описание графического интерфейса. Это позволит студентам, которые хотят начать работать с данным лабораторным комплексом быстрее разобраться в его основах.

## Список использованных источников

1. Анхимюк, В.Л. Теория автоматического управления [Текст] / В.Л. Анхимюк, О.Ф. Олейко, Н.Н. Михеев. – М.: Дизайн ПРО, 2002. – 352 с.
2. Бесекерский, В.А. Теория систем управления программным обеспечением [Текст] / В.А. Бесекерский, Е.П. Попов. – 4-е изд., перераб. и доп. – СПб.: Профессия, 2003. – 747 с.
3. Гудвин, Г.К. Проектирование программного обеспечения [Текст] / Г.К. Гудвин, С.Ф. Гребе, М.Э. Сальдаго. – пер. с англ. – М.: БИНОМ, Лаборатория знаний, 2004. – 911 с.
4. Брюханов, В.Н. Теория управления программным обеспечением [Текст] : Учеб. для машиностроит. спец. вузов / В.Н. Брюханов, М.Г. Косов, С.П. Протопопов и др.; под ред. Ю.М. Соломенцева. – 3-е изд., стер. – М.: Высш. шк.; 2000. – 268 с.
5. Техносфера [Электронный ресурс] / 2015. – Режим доступа: <http://tekhnosfera.com>, свободный. – Загл. с экрана.
6. ВЕДА программное обеспечение широкого профиля [Электронный ресурс] / 2015. – Режим доступа: <http://www.medical.ua-ru.net>, свободный. – Загл. с экрана.
7. Lib.ru: Журнал «Самиздат» [Электронный ресурс] / 2015. – Режим доступа: <http://samlib.ru>, свободный. – Загл. с экрана.
8. CIT FORUM [Электронный ресурс] / 2015. – Режим доступа: <http://citforum.ru>, свободный. – Загл. с экрана.
9. ОЛЛИ Информационные технологии [Электронный ресурс] / 2015. – Режим доступа: <http://www.olly.ru>, свободный. – Загл. с экрана.
10. TADVISER [Электронный ресурс] / 2015. – Режим доступа: <http://www.tadviser.ru>, свободный. – Загл. с экрана.
11. shportal [Электронный ресурс] / 2015. – Режим доступа: <http://shportal.ru>, свободный. – Загл. с экрана.

# Приложение А

## Исходный код лабораторной работы №8

```
double SQRN(double a, double b)
{
    if (a != 0) return Math.Exp(b*Math.Log(a));
    return 0;
}
double f1(double x, double y1, double y2)
{
    return y2;
}
double f2(double x, double y1, double y2)
{
    return SQRN(2.71828182846, -x * y1);
}
void eiler(double a, double b, int n, int kolfun, double x, MyDelegate[] f,
double[,] y_1)
{
    double t = (b - a) / n;
    x = x + t;
    for (int i = 1; i < n; i++)
    {
        for (int k = 0; k < kolfun; k++)
        {
            y_1[k,i]=y_1[k,i-1]+t*f[k](x,y_1[0,i-1],y_1[1,i-1]);
        }
        dataGridView1.Columns[i].HeaderCell.Value = "X = " + x;
        dataGridView1.Rows[0].Cells[i].Value = y_1[0,i];
        dataGridView1.Rows[1].Cells[i].Value = y_1[1, i];
        x += t;
    }
    dataGridView1.Columns[0].HeaderCell.Value = "X = " + a;
    dataGridView1.Rows[0].Cells[0].Value = y_1[0, 1];
    dataGridView1.Rows[1].Cells[0].Value = y_1[1, 2];
}
void prognos(double a, double b, int n, int kolfun, double x, MyDelegate[] f,
double[,] y_1)
{
    double t = (b - a) / n;
    x = x + t;
    for (int i = 1; i < n; i++)
    {
        for (int k = 0; k < kolfun; k++)
        {
            y_1[k,i]=y_1[k,i-1]+t*f[k](x+0.5*t,y_1[0,i-1]+0.5*t*y_1[0,i-
1],y_1[1,i-1]+0.5*t*y_1[1,i-1]);
        }
        dataGridView1.Columns[i].HeaderCell.Value = "X = " + x;
        dataGridView1.Rows[0].Cells[i].Value = y_1[0, i];
        dataGridView1.Rows[1].Cells[i].Value = y_1[1, i];
        x += t;
    }
    dataGridView1.Columns[0].HeaderCell.Value = "X = " + a;
    dataGridView1.Rows[0].Cells[0].Value = y_1[0, 1];
    dataGridView1.Rows[1].Cells[0].Value = y_1[1, 2];
}
double[,] Runge_Kut(double a, double h, int n, double x, double[,] y_1,
MyDelegate[] FMas)
{
    double[] k = new double[4];
    for (int i = 1; i < n; i++)
    {
```

```

        for (int j = 0; j < 2; j++)
        {
            k[0] = FMas[j](x, y_1[0, i - 1], y_1[1, i - 1]);
            k[1] = FMas[j](x + h / 2, y_1[0, i - 1] + h / 2 * k[1], y_1[1, i - 1]
+ h / 2 * k[1]);
            k[2] = FMas[j](x + h / 2, y_1[0, i - 1] + h / 2 * k[2], y_1[1, i - 1]
+ h / 2 * k[2]);
            k[3] = FMas[j](x + h, y_1[0, i - 1] + h * k[3], y_1[1, i - 1] + h *
k[3]);
            y_1[j, i] = y_1[j, i - 1] + h / 6 * (k[0] + 2 * k[1] + 2 * k[2] +
k[3]);
        }
        x = x + h;
        dataGridView1.Columns[i].HeaderCell.Value = "X = " + x;
        dataGridView1.Rows[0].Cells[i].Value = y_1[0, i];
        dataGridView1.Rows[1].Cells[i].Value = y_1[1, i];
    }
    dataGridView1.Columns[0].HeaderCell.Value = "X = " + a;
    dataGridView1.Rows[0].Cells[0].Value = y_1[0, 1];
    dataGridView1.Rows[1].Cells[0].Value = y_1[1, 2];
    return y_1;
}

```

## Приложение Б

### Исходный код лабораторной работы №9

```
void Adams(double a, double b, double h, double[] NewValues, MyDelegate[] FMas)
{
    double x=a;
    double[,] y_1 = new double[2, 150];
    int n = (int)Math.Round((b-a)/h);
    y_1[0,1]=NewValues[0];
    y_1[1,1]=NewValues[1];
    y_1 = Runge_Kut(a,h, 30, x,y_1,FMas);
    x = 0.3;
    for (int i = 4; i < n; i++)
    {
        for (int j = 0; j < 2; j++)
        {
            NewValues[j]=y_1[j,3]+h/24*(55*FMas[j](x,y_1[0,3],y_1[1,3])-
                59*FMas[j](x-h,y_1[0,2],y_1[1,2])+
                37*FMas[j](x-2*h,y_1[0,1],y_1[1,1])-
                9*FMas[j](x-3*h,y_1[0,0],y_1[1,0]));
        }
        for (int j = 1; j < 4; j++)
        {
            y_1[0,j-1]=y_1[0,j];
            y_1[1,j-1]=y_1[1,j];
        }
        y_1[0,4]=NewValues[0];
        y_1[1,4]=NewValues[1];
        x=x+h;
        dataGridView1.Columns[i].HeaderCell.Value = "X = " + x;
        dataGridView1.Rows[0].Cells[i].Value = y_1[0, i];
        dataGridView1.Rows[1].Cells[i].Value = y_1[1, i];
    }
}
```

## Приложение В

### Исходный код лабораторной работы №10

```
private delegate double MyDelegate(double x, double y1, double y2);
double[] Runge_Kut(double a, double b, int n, double x, double[] y_1,
MyDelegate[] FMas)
{
    double[] New = new double[2];
    double[] k = new double[4];
    double h=(b-a)/n;
    New[0]=y_1[0];
    New[1]=y_1[1];
    for (int j = 0; j < 2; j++ )
    {
        k[0]=h*FMas[j](x,New[0],New[1]);
        k[1] = h * FMas[j](x + (h / 2.0), New[0] + (1 / 2.0) * k[0], New[1] + (1
/ 2.0) * k[0]);
        k[2] = h * FMas[j](x + h / 2.0, New[0] + (1 / 2.0) * k[1], New[1] + (1 /
2.0) * k[1]);
        k[3]=h*FMas[j](x+h,New[0]+k[2],New[1]+k[2]);
        y_1[j] = y_1[j] + (h / 6.0) * (k[0] + 2 * k[1] + 2 * k[2] + k[3]);
    }
    return y_1;
}
void Gir(double a, double b, int n, double x, double[] y_1, MyDelegate[] FMas)
{
    int z = 0;
    double c1, c2;
    double[,] Fun = new double[2, 4];
    double h = (b - a) / n;
    for (int i = 0; i < 4; i++)
    {
        y_1 = Runge_Kut(a, b, n, x, y_1, FMas);
        x = x + h;
        for (int j = 0; j < 2; j++)
        {
            Fun[j, i] = y_1[j];
        }
    }
    x = a + 0.1;
    for (int i = 0; i < n; i++)
    {
        c1=(-48*Fun[0,3]+36*Fun[0,2]-16*Fun[0,1]+3*Fun[0,0])/1.2;
        c2=(-48*Fun[1,3]+36*Fun[1,2]-16*Fun[1,1]+3*Fun[1,0])/1.2;
        y_1[0]=(-9*c2-(25/1.2+11)*c1)/((25/1.2+11)*(25/1.2+11)-9*9);
        y_1[1]=(9*y_1[0]-c2)/(25/1.2+11);
        z++;
        for (int j = 0; j < 2; j++)
        {
            Fun[j,0]=Fun[j,1];
            Fun[j,1]=Fun[j,2];
            Fun[j,2]=Fun[j,3];
            Fun[j,3]=y_1[j];
        }
        dataGridView1.Columns[z-1].HeaderCell.Value = "X = " + x;
        dataGridView1.Rows[0].Cells[z-1].Value = y_1[0];
        dataGridView1.Rows[1].Cells[z-1].Value = y_1[1];
        x += h;
    }
}
double f1(double x, double y1, double y2)
{
```

```

        return -11 * y1 + 9 * y2;
    }
    double f2(double x, double y1, double y2)
    {
        return 9 * y1 - 11 * y2;
    }
    private void button1_Click(object sender, EventArgs e)
    {
        MyDelegate[] FMas = new MyDelegate[2];
        double[] y_1 = new double[2];
        y_1[0]=1;
        y_1[1]=0;
        FMas[0]=f1;
        FMas[1]=f2;
        Gir(0,2,20,0,y_1,FMas);
    }
    double[,] MulMat(int n, double[,] a, double[,] b)
    {
        double[,] c = new double[n, n];
        for (int i = 0; i < n; i++)
        {
            for (int j = 0; j < n; j++)
            {
                for (int k = 0; k < n; k++)
                {
                    c[i,j]=c[i,j]+a[i,k]*b[k,j];
                }
            }
        }
        return c;
    }
    double[] MatnaVec(int n, double[,] a, double[] b)
    {
        double[] c = new double[n];
        for (int i = 0; i < n; i++)
        {
            for (int j = 0; j < n; j++)
            {
                c[i]=c[i]+a[j,i]*b[j];
            }
        }
        return c;
    }
    double[,] MulConst(int n, double h, double[,] b)
    {
        for (int i = 0; i < n; i++)
        {
            for (int j = 0; j < n; j++)
            {
                b[i, j] = b[i, j] * h;
            }
        }
        return b;
    }
    double[,] MetRak(int n, double h, double[,] a, double[] y, double[,] q, int Ind,
int nts)
    {
        double s = 0;
        double[,] Rabmas1 = new double[n, n];
        //double[,] q = new double[n, n];
        double[] y1 = new double[n], Rabmas2 = new double[n];
        double[] y0 = new double[n];
        if (Ind == 0)
        {

```



```

for (int i = 0; i < n; i++)
{
    for (int j = 0; j < n; j++)
    {
        s += a[i,j]*a[i,j];
    }
}
s=Math.Sqrt(s);
h=0.1/s;
nts = 0;
for (int i = 0; i < n; i++)
{
    for (int j = 0; j < n; j++)
    {
        q[i,j]=0;
    }
}
a = MulConst(n, h, a);
for (int k = 5; k >= 1; k--)
{
    q = MulConst(n, 1 / k, q);
    for (int i = 0; i < n; i++)
    {
        q[i, i] = q[i, i] + 1;
    }
    double[,] temp = MulMat(n, q, a);
    q=temp;
}
for (int i = 0; i < n; i++)
{
    q[i, i] = q[i, i] + 1;
}
for (int i = 0; i < n; i++)
{
    for (int j = 0; j < n; j++)
    {
        y0[j]=0;
    }
    y0[i]=1;
    for (int j = 0; j < Math.Round(0.1/h); j++)
    {
        y1 = MatnaVec(n,q,y0);
        y0=y1;
    }
    for (int j = 0; j < n; j++)
    {
        Rabmas1[j,i]=y0[j];
    }
    double test;
}

q=Rabmas1;
}
if (nts > 0)
{
    Rabmas2 = MatnaVec(n, q, y);
    y=Rabmas2;
}
y_result = y;
h_result = h;
return q;
}
double[] getY0()

```

```

{
    return y_result;
}
double getH()
{
    return h_result;
}
private void button2_Click(object sender, EventArgs e)
{
    int n=2;
    double[,] a1 = new double[n,n];
    double[] y0 = new double[n];
    a1[0,0]=-11; a1[0,1]= 9;
    a1[1,0]= 9; a1[1,1]=-11;
    y0[0]=1; y0[1]=0;
    double x=0;
    double h = 0;
    int nts = 0;
    double[,] a2 = (double[,])a1.Clone();
    double[,] q1 = new double[n,n];
    q1 = MetRak(n, h, a2, y0, q1, 0, nts);
    y0 = getY0();
    h = getH();
    for (nts = 1; nts < 21; nts++)
    {
        a2 = (double[,])a1.Clone();
        q1 = MetRak(n, h, a2, y0, q1, 1, nts);
        y0 = getY0();
        h = getH();
        dataGridView1.Columns[nts - 1].HeaderCell.Value = "X = " + x;
        dataGridView1.Rows[0].Cells[nts - 1].Value = y0[0];
        dataGridView1.Rows[1].Cells[nts - 1].Value = y0[1];
        x=x+0.1;
    }
}

```

## Приложение Г

### Исходный код лабораторной работы №11

```
void Spline3(int n, double[] x, double[] y, double s0, double sn, double[] a, double[] b,
double[] c, double[] d)
{
    double[] f = new double[n];
    double h2 = x[2] - x[1];
    double h3 = x[3] - x[2];
    a[1] = (2 * (h2 + h3)) / h3;
    f[1] = (6 / h3) * (((y[3] - y[2]) / h3) - ((y[2] - y[1]) / h2)) - (h2 * s0) /
h3;
    for (int i = 4; i <= n - 1; i++)
    {
        h2 = x[i - 1] - x[i - 2];
        h3 = x[i] - x[i - 1];
        a[i - 2] = (2 / h3) * (h2 + h3);
        b[i - 2] = h2 / h3;
        f[i - 2] = (6 / h3) * (((y[i] - y[i - 1]) / h3) - ((y[i - 1] - y[i - 2]) /
h2)));
    }
    h2 = x[n - 1] - x[n - 2];
    h3 = x[n] - x[n - 1];
    double p = 2 * (h2 + h3);
    b[1] = h2 / p;
    f[n - 2] = (6 / p) * (((y[n] - y[n - 1]) / h3) - ((y[n - 1] - y[n - 2]) /
h2)) - (h3 * sn) / p;
    d[1] = 1 / a[1]; c[1] = f[1];
    for (int i = 2; i <= n - 3; i++)
    {
        d[i] = 1 / (a[i] - b[i] * d[i - 1]); c[i] = f[i] - b[i] * d[i - 1] * c[i
- 1];
    }
    d[n - 2] = (f[n - 2] - b[1] * d[n - 3] * c[n - 3]) / (1 - b[1] * d[n - 3]);
    for (int i = n - 3; i >= 1; i--)
    {
        d[i] = d[i] * (c[i] - d[i + 1]);
    }
    c[1] = s0; c[n] = sn;
    for (int i = 2; i <= n - 1; i++)
    {
        c[i] = d[i - 1];
    }
    for (int i = 1; i <= n; i++)
    {
        a[i] = 0; b[i] = 0; d[i] = 0;
    }
    for (int i = 2; i <= n; i++)
    {
        h2 = x[i] - x[i - 1]; d[i] = (c[i] - c[i - 1]) / h2;
        b[i] = h2 * c[i] / 2 - (Math.Pow(h2, 2)) * d[i] / 6 + (y[i] - y[i - 1]) /
h2;
        a[i] = y[i];
    }
}

void DifSpline(double[] x, double[] y, double[] z, double xx, int n, double s1,
double s2, bool error)
{
    int i = 1;
    while ((xx >= x[i]) && (i <= n)) { i++; }
    if (i > n) { error = true; return; }
    double hi = x[i] - x[i - 1];
    double t = (xx - x[i - 1]) / hi; //(x[i]-x[i-1]);
}
```

```

        double t1 = 1 - t;
        double t2 = t * t;
        double t12 = t1 * t1;
        s1 = -6 * t1 * t * (y[i - 1] - y[i]) / hi;
        s1 = s1 + t1 * z[i - 1] * (1 - 3 * t) + z[i] * (3 * t - 2) * t;
        s2 = 6 * (y[i - 1] - y[i]) * (2 * t - 1) / hi;
        s2 = s2 + (z[i - 1]) * (6 * t - 4) - z[i] * (2 - 6 * t);
        s2 = s2 / hi;
    }
    private delegate double MyDelegate(double x);
    double f(double x)
    {
        if (radioButton1.Checked) { return Math.Sin(x); }
        if (radioButton2.Checked) { return Math.Cos(x); }
        if (radioButton3.Checked) { return Math.Exp(Math.Pow(x, 2)); }
        return 0;
    }
    double fx(double x)
    {
        if (radioButton1.Checked) { return Math.Cos(x); }
        if (radioButton2.Checked) { return -Math.Sin(x); }
        if (radioButton3.Checked) { return 2 * x * Math.Exp(Math.Pow(x, 2)); }
        return 0;
    }
    double fxx(double x)
    {
        if (radioButton1.Checked) { return -Math.Sin(x); }
        if (radioButton2.Checked) { return -Math.Cos(x); }
        if (radioButton3.Checked) { return 2 * Math.Exp(Math.Pow(x, 2)) * (1 + 2 * x
* x); }
        return 0;
    }
    private void button1_Click(object sender, EventArgs e)
    {
        chart1.Series[0].Points.Clear();
        int m = 15;
        int n = 300;
        double a = 0;
        double b = 1;
        double koefx = 1;
        double xc = -a * koefx;
        double dx = (b - a) / (m - 1);
        double[] x = new double[n];
        double[] ys = new double[n];
        x[0] = a - dx; ys[0] = f(x[0]);
        x[m + 1] = b + dx; ys[m + 1] = f(x[m + 1]);
        x[1] = a; ys[1] = f(a);
        x[m] = b; ys[m] = f(b);
        for (int i = 1; i < m; i++)
        {
            x[i] = x[i - 1] + dx;
            ys[i] = f(x[i]);
        }
        double max = f(a); double min = f(a);
        for (int i = 0; i < m; i++)
        {
            if (max < f(x[i])) { max = f(x[i]); }
            if (min > f(x[i])) { min = f(x[i]); }
        }
        double dt = (chart1.Width - 1) / (b - a);
        double koefy = (chart1.Height - 1) / (max - min);
        koefx = (chart1.Width - 1) / (b - a);
        bool err = false;
        double xx = 0.55;
    }

```

```

        double s1 = 0, s2 = 0;
        double[] z = new double[n], s = new double[n], c = new double[n], d = new
double[n];
        Spline3(m, x, ys, x[0], x[m + 1], z, s, c, d);
        DifSline(x, ys, s, xx, m, s1, s2, err);

        chart1.Series[0].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.FastPoint; // тут сами
поизменяет/повыбирайте тип вывода графика

        for (int i = 1; i < m; i++)
        {
            a=x[i]; b=x[i+1];
            while (a <= b)
            {

                //double xxx = Math.Round(a * koefx + xc);
                //double y = chart1.Height - Math.Round((z[i] + s[i] * (a - x[i]) +
(c[i] / 2) * (a - x[i]) * (a - x[i]) + (d[i] / 6) * (a - x[i]) * (a - x[i]) * (a - x[i])
- min) * koefy);
                double xxx = x[i];
                double y = ys[i];
                chart1.Series[0].Points.AddXY(xxx, y);

                a=a+dt;

            }
        }
    }
}

```

## Приложение Д

### Исходный код лабораторной работы №12

```
private delegate double MyDelegate(double x);
double f(double x)
{
    double result = -0.5;
    if (x > 1e-8)
    {
        result = (Math.Cos(x) - 1) / (x * x);
    }
    return result;
}
double rect(double a, double b, int n)
{
    double h = (b - a) / n;
    double h2 = h / 2.0;
    double s = 0.0;
    for (int j = 0; j < n; j++)
    {
        double x = a + j * h - h2;
        s = s + f(x);
    }
    return s * h;
}
double trap(double a, double b, int n)
{
    double h = (b - a) / n;
    double s = (f(a) + f(b)) * 0.5;
    for (int j = 1; j <= n - 1; j++)
    {
        double x = a + j * h;
        s = s + f(x);
    }
    return s * h;
}
double simps(double a, double b, int n)
{
    int n2 = n * 2;
    int n1 = n2 - 1;
    double h = (b - a) / n2;
    double s = f(a) + f(b);
    for (int j = 0; j < n1; j++)
    {
        double z = 3.0 - Math.Pow((-1), j);
        double x = a + j * h;
        s = s + z * f(x);
    }
    return s * h / 3;
}
double gauss(double a, double b)
{
    double[] ag = new double[] {0.10122854, 0.22238104,
                                0.31370664, 0.36278378,
                                0.36268378, 0.31370664,
                                0.22238104, 0.10122854};
    double[] xg = new double[] {-0.96028986, -0.79666648,
                                -0.52553242, -0.18343464,
                                0.18343464, 0.52553242,
                                0.79666648, 0.96028986};
    double a1 = (b + a) * 0.5;
    double a2 = (b - a) * 0.5;
    double g = 0.0;
```

```

        for (int i = 0; i < 8; i++)
        {
            double x = a1 + a2 * xg[i];
            g = g + ag[i] * f(x);
        }
        return g * a2;
    }

private void button1_Click(object sender, EventArgs e)
{
    double eps = 0.0001;
    int n = 1;
    double a = 0.0;
    double b = 1.0;
    double ts = 0, zz, res = 1e+10;
    do
    {
        n = n * 2;
        zz = res;
        res = rect(a,b,n);
        //Memo1.Lines.Add(IntToStr(n) + ' ' + FloatToStrF(Res,ffExponent,5,5));
        ts = Math.Abs(res - zz)/3;

    } while (ts > eps);
    textBox2.Text = res.ToString();
}

private void button2_Click(object sender, EventArgs e)
{
    double eps = 0.0001;
    int n = 1;
    double a = 0.0;
    double b = 1.0;
    double ts = 0, zz, res = 1e+10;
    do
    {
        n = n * 2;
        zz = res;
        res = trap(a, b, n);
        //Memo1.Lines.Add(IntToStr(n) + ' ' + FloatToStrF(Res,ffExponent,5,5));
        ts = Math.Abs(res - zz) / 3;

    } while (ts > eps);
    textBox2.Text = (res + ts).ToString();
}

private void button3_Click(object sender, EventArgs e)
{
    double eps = 0.0001;
    int n = 1;
    double a = 0.0;
    double b = 1.0;
    double ts = 0, zz, res = 1e+10;
    do
    {
        n = n * 2;
        zz = res;
        res = simps(a, b, n);
        ts = Math.Abs(res - zz) / 15;

    } while (ts > eps);
    textBox2.Text = (res + ts).ToString();
}

```

```
private void button4_Click(object sender, EventArgs e)
{
    double a = 0.0;
    double b = 1.0;
    textBox2.Text = gauss(a,b).ToString();
}
```



## Приложение Е

### Исходный код лабораторной работы №13

```
void ff(double[] xr, double[] xi, double[] yr, double[] yi, int n, int ind)
{
    int k = n;
    int log2 = 0;
    do
    {
        k = k / 2;
        log2 = log2 + 1;
    } while (k >= 2);
    int mm = 1;
    for (int m = 1; m <= log2; m++)
    {
        int m2 = mm * 2;
        double k1 = Math.Truncate(Math.Pow(2, (log2 - m)) - 1);
        int l1 = mm - 1;
        for (k = 1; k <= k1 + 1; k++)
        {
            for (int l = 1; l <= l1 + 1; l++)
            {
                int j = m2 * (k - 1) + l;
                int i = mm * (k - 1) + l;
                double w = Math.PI * (l - 1) / mm;
                double si = ind * Math.Sin(w);
                double co = Math.Cos(w);
                double ni = Math.Truncate(Math.Pow(2, (log2 - 1)) + i);
                double jm = Math.Truncate(j + Math.Pow(2, (m - 1)));
                double xa = xr[(int)ni] * co + xi[(int)ni] * si;
                double xb = xi[(int)ni] * co - xr[(int)ni] * si;
                yr[j] = xr[i] + xa;
                yi[j] = xi[i] + xb;
                yr[(int)jm] = xr[i] - xa;
                yr[(int)jm] = xr[i] - xa;
                yi[(int)jm] = xi[i] - xb;
            }
        }
        for (int it = 1; it <= n; it++)
        {
            xr[it] = yr[it];
            xi[it] = yi[it];
        }
        mm = m2;
    }
    if (ind < 0) { return; }
    for (int i = 1; i <= n; i++)
    {
        xr[i] = xr[i] / n;
        xi[i] = xi[i] / n;
    }
}

void iff(double a, double b, double[] ar, double[] ai, double[] xr, double[] xi,
int n, double eps, int ip)
{
    double h = (b - a)/n;
    double a1, b1, w4, w5, w1, w2, w3;;
    ff(ar, ai, xr, xi, n, ip);
    for (int i = 1; i <= n; i++)
    {
        int k = i - 1;
        double w = k * 2 * Math.PI / (b - a);
        double c = Math.Cos(w * a);
```

```

double s = Math.Sin(w * a);
w = w * h * (b - a);
if (w < eps)
{
    w4 = h - 2 * Math.PI * Math.PI * k * k * Math.Pow(h, 3) / Math.Pow((b
- a), 2);

    w5 = 2 * Math.PI * k * h * h / (b - a);
    a1 = w4 * ar[i] + ip * w5 * ai[i];
    b1 = w4 * ai[i] - ip * w5 * ar[i];
}
else
{
    if (k == 0)
    {
        w1 = 0;
        w3 = 0;
    }
    else
    {
        w1 = Math.Sin(2 * Math.PI * k / n) * (b - a) / (2 * Math.PI * k);
        w2 = 2 * Math.Pow(Math.Sin(Math.PI * k / n), 2);
        w3 = w2 * (b - a) / (2 * Math.PI * k);
    }
    w2 = 2 * Math.Pow(Math.Sin(Math.PI * k / n), 2);
    a1 = w1 * ar[i] + ip * w3 * ai[i];
    b1 = w1 * ai[i] - ip * w3 * ar[i];
}
ar[i] = a1 * c + b1 * s * ip;
ai[i] = b1 * c - a1 * s * ip;
if (ip > 0)
{
    ar[i] = ar[i] * n * n;
    ai[i] = ai[i] * n * n;
}
ar[i] = ar[i] / n;
ai[i] = ai[i] / n;
}
}

```

# Приложение Ж

## Презентационный материал

Федеральное агентство связи  
Федеральное государственное бюджетное образовательное учреждение высшего образования  
«Поволжский государственный университет телекоммуникаций и информатики»

Факультет Заочного Отделения  
Кафедра «ПОУТС»

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

***Программирование численных методов.***

***Часть 2***

*Разработал:*  
студент группы 26П  
Кондрашов И.А.

*Руководитель:*  
ст. преп. каф. ПОУТС  
Малахов С.В.

Самара, 2016

### Актуальность

2

В рамках работы требуется разработать комплекс программ решения численных методов на языке программирования высокого уровня, который изучается студентами ПГУТИ по дисциплине «Численные методы». Комплекс состоит из 6 заданий. Комплекс будет использоваться в замен старому, который предназначен для программирования на языке Pascal

Предметом данной бакалаврской работы является программирование численных методов.

Объектом исследования — задачи линейного

## Введение

3

Эффективное решение крупных естественнонаучных и народнохозяйственных задач сейчас невозможно без применения быстродействующих электронно-вычислительных машин (ЭВМ). В настоящее время выработалась технология исследования сложных проблем, основанная на построении и анализе с помощью ЭВМ математических моделей изучаемого объекта. Такой метод исследования называют вычислительным экспериментом.

## Язык программирования и среда разработки

4

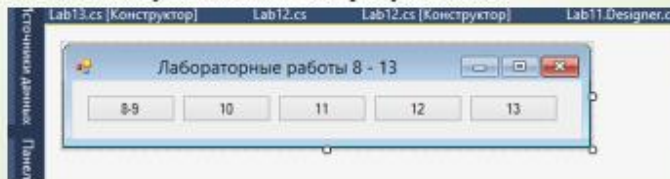


В качестве средств разработки были выбраны, язык программирования **C#** и среда разработки **Microsoft Visual Studio**. Так как являются наиболее распространёнными и используются в университете ПГУТИ

## Требования к графическому интерфейса

5

1. Наличие графического элемента «матрица», для ввода начальных значений.
2. Наличие графических элементов «текстовое поле», для ввода уточняющих значений, таких как «начало отрезка», «конец отрезка» и т.д.
3. Наличие графического элемента «матрица», для вывода результирующих значений численных методов.
4. Наличие графического элемента «кнопка», для вызова начала вычислений того или иного численного метода или уточнений его результатов.



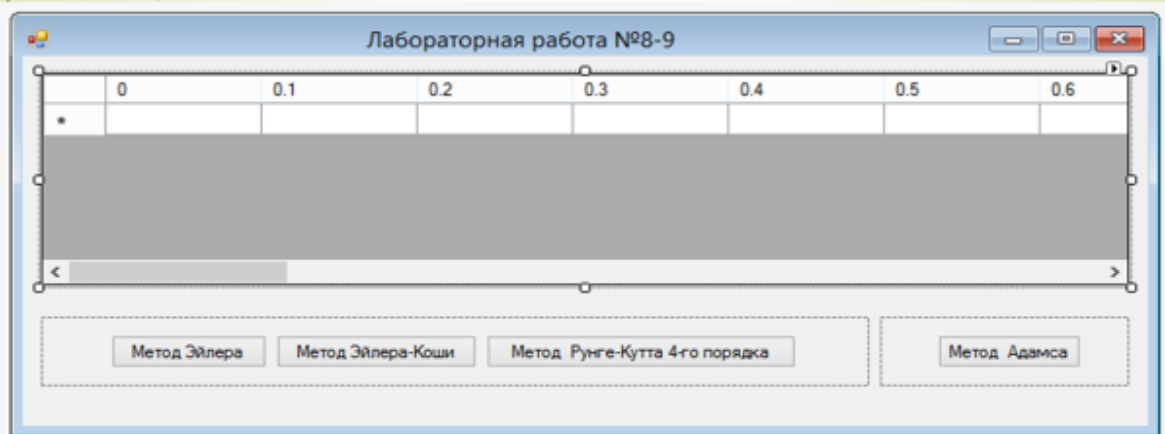
## Решение следующих численных методов

6

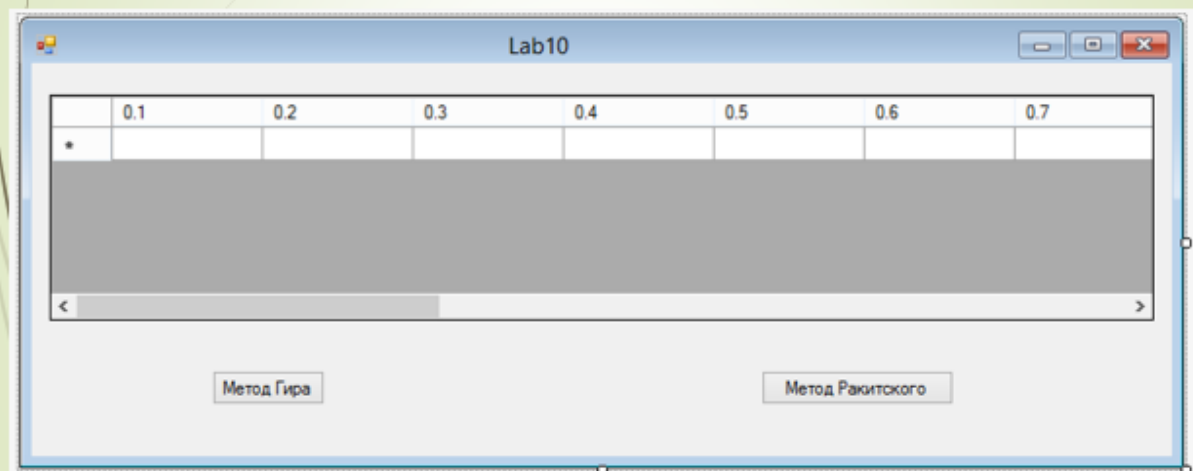
1. Одношаговые методы решения задачи Коши: метод Эйлера, метод Эйлера-Коши и метод Рунге-Кутты 4-го порядка.
2. Многошаговые методы решения задачи Коши: метод Адамса (явный).
3. Решение жестких систем ОДУ: метод Гира, метод Ракитского (матричной экспоненты).
4. Численное дифференцирование: дифференцирование с помощью сплайнов.
5. Численное интегрирование: формула прямоугольников, формула трапеций, формула Симпсона, формула Гаусса.
6. Приближенное вычисление преобразования Фурье.



**Одношаговые методы решения задачи Коши: метод Эйлера, метод Эйлера-Коши и метод Рунге-Кутты 4-го порядка.**  
**Многошаговые методы решения задачи Коши: метод Адамса (явный).**

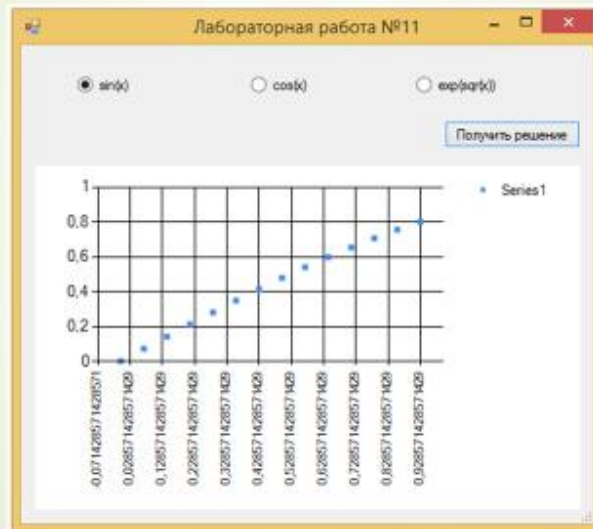


**Решение жестких систем ОДУ: метод Гира, метод Ракитского (матричной экспоненты)**



## Численное дифференцирование: дифференцирование с помощью сплайнов

9



## Численное интегрирование: формула прямоугольников, формула трапеций, формула Симпсона, формула Гаусса

10

Лаборат...

Формула прямоугольников

Формула трапеций

Формула Симпсона

Формула Гаусса

Ответ

## Приближенное вычисление преобразования Фурье

11

Lab13			
	Омега	Действительная часть	Мнимая часть
0	-0.31415926535...	0	0
1	0	0.905758175452...	0
2	0.314159265358...	0.883188903385...	-0.15990233151...
3	0.628318530717...	0.818776052565...	-0.30426521088...
4	0.942477796076...	0.721617986314...	-0.42050815297...

## Исходный код функции Эйлера

12

```
void eiler(double a, double b, int n, int kolfun, double x, MyDelegate[] f, double[,] y_1)
{
    double t = (b - a) / n;
    x = x + t;
    for (int i = 1; i < n; i++)
    {
        for (int k = 0; k < kolfun; k++)
        {
            y_1[k,i]=y_1[k,i-1]+t*f[k](x,y_1[0,i-1],y_1[1,i-1]);
        }
        dataGridView1.Columns[i].HeaderCell.Value = "X = " + x;
        dataGridView1.Rows[0].Cells[i].Value = y_1[0,i];
        dataGridView1.Rows[1].Cells[i].Value = y_1[1, i];
        x += t;
    }
    dataGridView1.Columns[0].HeaderCell.Value = "X = " + a;
    dataGridView1.Rows[0].Cells[0].Value = y_1[0, 1];
    dataGridView1.Rows[1].Cells[0].Value = y_1[1, 2];
}
```



13

## Задача Эйлера-Коши

$$\begin{cases} dy_1/dx = y_2 \\ dy_2/dx = e^{-xy_1} \end{cases} \quad \begin{cases} y_1(0) = 0 \\ y_2(0) = 0 \end{cases}$$

```
double SQRN(double a, double b)
{
    if (a != 0) return Math.Exp(b*Math.Log(a));
    return 0;
}
double f1(double x, double y1, double y2)
{
    return y2;
}
double f2(double x, double y1, double y2)
{
    return SQRN(2.71828182846, -x * y1);
}
```

Содержимое функций

14

## Решение задачи Эйлера-Коши

Лабораторная работа №8-9

	X = 2.2	X = 2.3	X = 2.4	X = 2.5	X = 2.6	X = 2.7	X = 2.8
y	72493105389...	1.8933145733816	2.014305669106...	2.135403077600...	2.2565654983756	2.377766714189...	2.49899052240...
y	8214679922...	1.209910957251...	1.210974084936...	1.211624207752...	1.212012158139...	1.212238082130...	1.21236648576...
*							

Метод Эйлера    Метод Эйлера-Коши    Метод Рунге-Кутты 4-го порядка    Метод Адамса

## Основные результаты и краткие выводы

15

В ходе выполнения бакалаврской работы, было проделано следующее:

- 1) проведен теоретический обзор предметной области.
- 2) разработано ПО, которое подходит для применения в лабораторном комплексе по предмету «Численные методы» на кафедре ПОУТС, состоящий из 6-ти заданий;
- 3) в качестве среды разработки была выбрана MS Visual Studio, а язык программирования C#;
- 4) разработан графический интерфейс, который позволил отказаться от «консоли» и сконцентрировать внимание пользователя на результатах;
- 5) запрограммированы численные методы с первой по седьмую лабораторную работу;
- 6) описаны основные возможности программного обеспечения, такие как примеры решения численных методов и описание графического интерфейса.