МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Тульский государственный университет»

Институт прикладной математики и компьютерных наук

Кафедра вычислительной техники

Курсовая работа по дисциплине

ЧИСЛЕННЫЕ МЕТОДЫ

на тему: ЧИСЛЕННОЕ РЕШЕНИЕ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ

Студент группы 220681 Пахоменкова М.И. \_\_\_\_\_\_\_\_\_

(Ф.И.О.) (подпись, дата)

Руководитель работы к.т.н., доцент каф. ВТ Волошко А.Г. \_\_\_\_\_\_\_\_\_

(Ф.И.О., должность) (подпись, дата)

Комиссия: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Тула 2020

МИНОБРНАУКИ РОССИИ

ФГБОУ ВО «Тульский государственный университет»

Институт прикладной математики и компьютерных наук

Кафедра «Вычислительная техника»

**Задание**

На курсовую работу по дисциплине «Численные методы»

студентке группы 220681 Пахоменковой Марии Игоревны

Тема работы:

«Методы решения дифференциальных уравнений»

Входные данные Вариант №16:

задача: решение дифференциальных уравнений методом Эйлера и методом Адамса;

уравнение: на отрезке [a,b], c шагом h;

приложение №1: приложение для решения численными методами дифференциальных уравнений.

Задание получил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ «20» февраля 2020 г.

(подпись студента)

Срок представления задания «23» июня 2020 г.

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись) (расшифровка подписи)

«\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_г.

К защите. Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись) (расшифровка подписи)

Замечания руководителя \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_«\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_г.

*При защите курсового проекта (работы) наличие рецензии обязательно.*

Содержание

[ВВЕДЕНИЕ 3](#_Toc42187143)

[1. Цели и задачи работы 4](#_Toc42187144)

[2. Математическое описание методов 5](#_Toc42187145)

[2.1 Метод Эйлера 5](#_Toc42187146)

[2. Метод Адамса 7](#_Toc42187147)

[3. Приложение на базе WinForms/C# 9](#_Toc42187148)

[3.1 Программные средства разработки 9](#_Toc42187149)

[3.2 Логическое проектирование 11](#_Toc42187150)

[4. Описание входных и выходных данных 14](#_Toc42187151)

[5. Алгоритмы решения дифференциальных уравнений 15](#_Toc42187152)

[5.1 Метод Эйлера 15](#_Toc42187153)

[5.2 Метод Адамса 15](#_Toc42187154)

[6. Тестирование программы 18](#_Toc42187155)

[7. Проверка корректности работы программы 21](#_Toc42187156)

[8. Сравнительный анализ методов решения дифференциальных уравнений по критериям точности, вычислительной сложности. 24](#_Toc42187157)

[8.1 Погрешность метода Эйлера 24](#_Toc42187158)

[8.2 Погрешность метода Адамса 24](#_Toc42187159)

[8.3 Сравнительный анализ 25](#_Toc42187160)

[ЗАКЛЮЧЕНИЕ 27](#_Toc42187161)

[СПИСОК ЛИТЕРАТУРЫ 28](#_Toc42187162)

[ПРИЛОЖЕНИЕ 29](#_Toc42187163)

# **ВВЕДЕНИЕ**

Математическое моделирование задач механики, физики и других отраслей науки и техники сводятся к дифференциальным уравнениям. В связи с этим решение дифференциальных уравнений является одной из важнейших математических задач. В вычислительной математике изучаются численные методы решения дифференциальных уравнений, которые особенно эффективны в сочетании с использованием вычислительной техники. Дифференциальные уравнения делятся на две категории в зависимости от числа переменных: обыкновенные дифференциальные уравнения, содержащие одну независимую переменную, и уравнения с частными производными, содержащие несколько независимых переменных. Численные методы решения дифференциальных уравнений в настоящее время являются основным инструментом при исследовании научно-технических задач, описываемых дифференциальными уравнениями.

К численным методам решения обыкновенных дифференциальных уравнений относятся метод Эйлера, метод Рунге-Кутта и метод Адамса.

# **Цели и задачи работы**

При выполнении курсовой работы необходимо создать приложение, которое решает дифференциальное уравнение и сравнить методы по вычислительной эффективности. Приложение решает одну и ту же задачу методом Эйлера и методом Адамса.

Вариант задания №16:

* уравнение: на отрезке [a,b], c шагом h;
* приложение №1 – приложение для решения численными методами дифференциальных уравнений;

В процессе выполнения курсовой работы необходимо получить навык:

- анализа различных численных методов решения дифференциальных уравнений;

- разработки программных средств для решения численными методами дифференциальных уравнений.

# **Математическое описание методов**

## **Метод Эйлера**

Решить дифференциальное уравнение  численным методом - это значит для заданной последовательности аргументов  и числа , не определяя функцию *у = F(x),* найти такие значения  что  (*i = 1,2,...,n*) и .

Таким образом, численные методы позволяют вместо нахождения функции *у = F(x)* получить таблицу значений этой функции для заданной последовательности аргументов. Величина  называется *шагом интегрирования*. Рассмотрим некоторые из численных методов.

Метод Эйлера является сравнительно грубым и применяется в основном для ориентировочных расчетов.

Пусть дано дифференциальное уравнение первого порядка с начальным условием . Требуется найти решение на отрезке [а, b].

Разобьем отрезок [а,b] на n равных частей и получим последова­тельность , где  (*i = 1, 2,..., n*), a  - шаг интегрирования.

Выберем *k*-й участок  и проинтегрируем уравнение:



Тогда формула примет вид:



Обозначив,  т.е. , получим



Продолжая этот процесс и каждый раз принимая подынтегральную функцию на соответствующем участке постоянной и равной ее значению в начале участка, получим таблицу решений дифференциального уравнения на заданном отрезке [а, b].

Также метод Эйлера может быть применен к решению систем дифференциальных уравнений и дифференциальных уравнений высших порядков. Однако в последнем случае дифференциальные уравнения должны быть приведены к системе дифференциальных уравнений первого порядка.

## **Метод Адамса**

В том случае, когда правая часть уравнения сложное аналитическое выражение, для решения такого уравнения применяется метод Адамса, который не требует многократного подсчета правой части уравнения.

Пусть дано дифференциальное уравнение , с начальным условием , . Требуемся найти решение этого уравнения на отрезке [a.b]. Разобьем отрезок [a,b] на n равных частей точками    
(*i = 1, 2,..., n*), a  – проинтегрируем дифференциальное уравнение). Выберем участок  и проинтегрируем дифференциальное уравнение; тогда получим, или.

Для нахождения производной воспользуемся второй интерполяцион­ной формулой Ньютона (ограничиваясь при этом разностями третьего по­рядка):

.

или

.

Подставляя выражение для  из формулы в соотношение и учитывая, что , имеем



Обозначим в дальнейшем  (*i = 0,1,2,…,n*). Тогда для любой разности имеем  и .

По формуле  получаем решение уравнения. Формула носит название экстраполяционной формулы Адамса.

Для начала процесса нужны четыре начальных значения  - так называемый начальный отрезок, который может быть найден, исходя из начального условия с использованием одного из известных методов. Обычно начальный отрезок решения находится методом Рунге-Кутта.

Зная  можно определить ;; ; .

Далее составляется таблица разностей величины *q* (табл. 1).

Таблица 1. Таблица разностей величины *q*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| I |  |  |  |  |  |  |  |  |
| (1) | (2) | (3) | (4) | (5) | (6) | (7) | (8) | (9) |
| 0 |  |  | - |  |  |  |  |  |
| 1 |  |  | - |  |  |  |  | - |
| 2 |  |  | - |  |  |  | - | - |
| 3 |  |  |  |  |  | - | - | - |
| 4 |  |  | - | - | - | - | - | - |
| 5 |  | - | - | - | - | - | - | - |
| 6 |  | - | - | - | - | - | - | - |

Метод Адамса заключается в продолжении диагональной таблицы разностей с помощью формулы. Используя числа , которые располагаются в таблице по диагонали, полагая в формуле  
 *n = 3* (известное последнее значение *у* есть ), получаем:

.

Полученное значение  вносят и таблицу и находят . Затем используя значения  и  находят  т.е. получается новая диагональ. По этим данным вычисляют

 .

Таким образом, продолжают таблицу решения, вычисляя правую часть дифференциального уравнения на каждом этапе только один раз.

# **Приложение на базе WinForms/C#**

## **Программные средства разработки**

Для создания процедурного приложения на базе WinForms/C# была выбрана интегрированная среда разработки Microsoft Visual Studio Community 2017, которая предоставляет разработчику удобный интерфейс для проектирования приложений на различных языках программирования, поддерживаемых компанией Microsoft.

Стандартные пространства имен, которые будут использоваться при разработке приложения:

System содержит фундаментальные и базовые классы, определяющие часто используемые типы значений и ссылочных данных, события и обработчики событий, интерфейсы, атрибуты и исключения обработки;

[System.Collections.Generic](https://docs.microsoft.com/ru-ru/dotnet/api/system.collections.generic?view=netframework-4.8) содержит интерфейсы и классы, определяющие универсальные коллекции, которые позволяют пользователям создавать строго типизированные коллекции, обеспечивающие повышенную производительность и безопасность типов по сравнению с неуниверсальными строго типизированными коллекциями.

[System.ComponentModel](https://docs.microsoft.com/ru-ru/dotnet/api/system.componentmodel?view=netframework-4.8) предоставляет классы, используемые для реализации поведения компонентов и элементов управления во время разработки и выполнения. Это пространство имен содержит базовые классы и интерфейсы для реализации атрибутов и преобразователей типов, привязки к источникам данных и лицензирования компонентов.

[System.Data](https://docs.microsoft.com/ru-ru/dotnet/api/system.data?view=netframework-4.8) обеспечивает доступ к классам, представляющим архитектуру ADO.NET. ADO.NET позволяет создавать компоненты, эффективно управляющие данными из нескольких источников данных.

[System.Linq](https://docs.microsoft.com/ru-ru/dotnet/api/system.linq?view=netframework-4.8) содержит классы и интерфейсы, которые поддерживают LINQ.

[System.Text](https://docs.microsoft.com/ru-ru/dotnet/api/system.text?view=netframework-4.8) содержит классы, которые представляют кодировки ASCII и Юникода; абстрактные базовые классы для преобразования блоков знаков в блоки байтов и обратно; вспомогательный класс, который обрабатывает и форматирует объекты [String](https://docs.microsoft.com/ru-ru/dotnet/api/system.string?view=netframework-4.8), не создавая промежуточные экземпляры [String](https://docs.microsoft.com/ru-ru/dotnet/api/system.string?view=netframework-4.8).

System.Threading.Tasks предоставляет типы, которые упрощают работу по написанию параллельного и асинхронного кода.

System.Drawing обеспечивает доступ к базовым функциональным возможностям графического интерфейса;

System.Diagnostics предоставляет классы, позволяющие взаимодействовать с системными процессами, журналами событий и счетчиками производительности;

System.Windows.Forms пространство имен, использующееся для создания приложений Windows, пользующихся преимуществами полного пользовательского интерфейса, предоставляемых в операционной системе Microsoft Windows.

## **Логическое проектирование**

В процессе проектирования приложения WinForms/С# был создан класс Form1, определяющий интерфейс приложения, в котором были реализованы перечисленные функции. Диаграмма класса Form1 процедурного приложения представлена на рис. 1.

Приложение состоит из четырех textBox: left\_textBox (для ввода левой границы промежутка), right\_textBox (для ввода правой границы промежутка), y0\_textBox (для ввода значения y0) и step\_textBox (для ввода шага); из двух dataGridView Euler\_dataGridView и Adams\_dataGridView, которые состоят из трех столбцов: X\_data1 и X\_data2 (для вывода координаты X), Euler\_gridview и Adams\_gridview (для вывода значений по методу Эйлеру и Адамса), time\_Euler\_data1 и time\_Adams\_data2 (время выполнения методов Эйлера и Адамса); из двух кнопок calculate\_button (для вычислений) и clear\_button (для очистки всех полей); из пяти label ( для указаний для пользователей); из одного picturebox1 (для вывода уравнения).

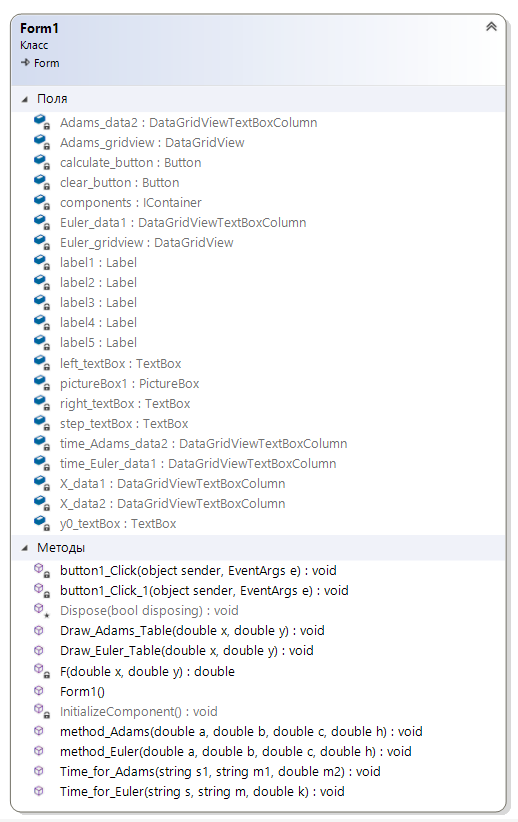


Рисунок 1 – Диаграмма класса Form1

Приложение содержит следующие методы:

private void button1\_Click(object sender, EventArgs e) – метод, с помощью которого по нажатию кнопки производятся вычисления;

private void button1\_Click\_1(object sender, EventArgs e) – метод, с помощью которого по нажатию кнопки очищаются все поля;

public void Draw\_Adams\_Table(double x, double y) – метод, с помощью которого заполняется таблица для метода Адамса;

public void Draw\_Euler\_Table(double x, double y) – метод, с помощью которого заполняется таблица для метода Эйлера;

static double F(double x, double y)- метод, содержащий функцию;

public void method\_Euler(double a, double b, double c, double h) – метод Эйлера;

public void method\_Adams(double a, double b, double c, double h) - метод Адамса;

public void Time\_for\_Adams(string s1, string m1, double m2)- метод, с помощью которого выводится время по методу Адамса;

public void Time\_for\_Euler(string s, string m,double k) - метод, с помощью которого выводится время по методу Эйлера.

# **Описание входных и выходных данных**

Входные данные приложения:

Уравнение –

Метод Эйлера:

public void method\_Euler(double a, double b, double c, double h)

Метод Адамса:

public void method\_Adams(double a, double b, double c, double h)

a – число типа double, левая граница;

b – число типа double, правая граница;

c – число типа double, начальный y0;

h – число типа double, шаг.

Выходные данные:

x – число типа double в DataGridView

y – число типа double в DataGridView;

time – время выполнения метода в DataGridView.

# **Алгоритмы решения дифференциальных уравнений**

## **Метод Эйлера**

1. Производится запуск реального времени time\_Euler с помощью функции DateTime.
2. Находится количество отрезков разбиения n по следующей формуле:

,

где a – левая граница промежутка, b – правая граница промежутка, h – шаг.

1. Инициализация и создание массивов X1 и Y1. Массив X1 будет хранить значения X, а массив Y1 будет хранить значения Y, вычисленные по методу Эйлера. Размер массива равен n+1.
2. Инициализация элементов массива. Первый элемент массива X1 равен a, т.е. левой границе промежутка. А первый элемент массива Y1 равен с, т.е. значению y0.
3. С использованием цикла по индексу (i=1) начинается заполнение массива X1 значениями с шагом h, массива Y1 по следующей формуле Эйлера:

1. С использованием цикла по индексу (i=0) заполняется таблица методом Draw\_Euler\_Table(x,y), который на вход принимает значения x и y.
2. Инициализация переменной result\_euler, она будет содержать время работы метода Эйлера. Для этого из настоящего времени работу метода и переводит в миллисекунды с помощью функции TotalMilliseconds. И с помощью метода Time\_for\_Euler(“ ”,” “, result\_euler), который на вход получает пустые строки и значение времени, выводит общее время работы.

Блок-схема метода Эйлера представлена на рисунке 2.

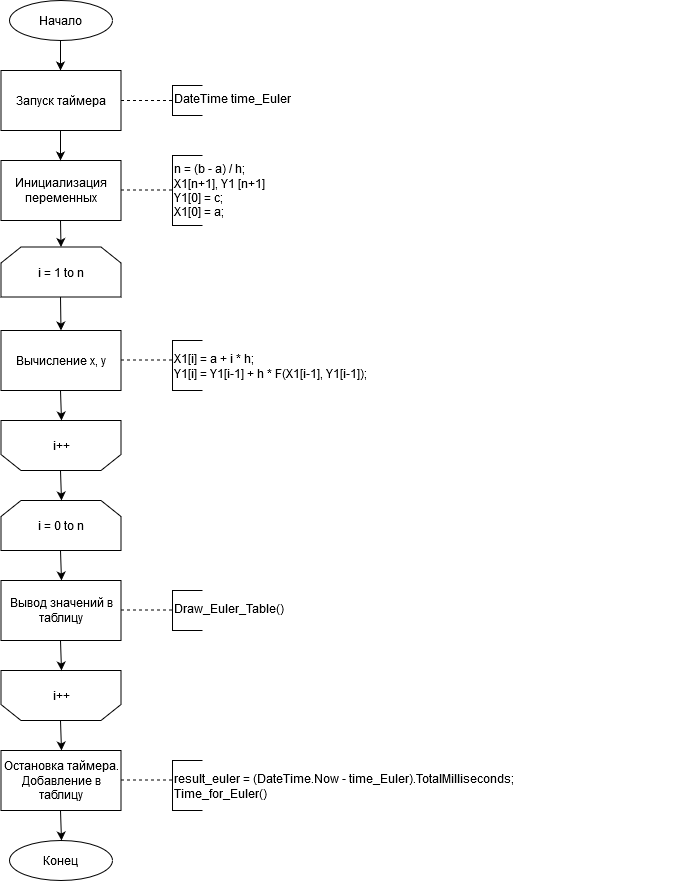


Рисунок 2 – Блок-схема метода Эйлера

## **Метод Адамса**

1. Производится запуск реального времени time\_Adams с помощью функции DateTime.
2. Инициализация и создание двух списков X и Y. Список X будет содержать значения x, а список Y соответственно значения y.
3. Объявление переменных x0 и y0, х0 – левая граница промежутка и равна a, y0 – начальное значение, равное c. Сразу же добавляются первые элементы, в список X переменная x0, а в список Y переменная y0 с помощью функции Add(). И с помощью метода Draw\_Adams\_Table(x,y) в таблице выводятся значения.
4. Объявление переменных k1, k2, k3, k4 – коэффициенты, x\_a и y\_a – x и y.
5. С использованием цикла по индексу (i=0) производится вычисление коэффициентов по следующим формулам (метод Рунге-Кутта):

k1 = f(x1, y1) \* h;

k2 = f(x1 + h / 2, y1 + k1 / 2) \* h;

k3 = f(x1 + h / 2, y1 + k2 / 2) \* h;

k4 = f(x1 + h, y1 + k3) \* h;

И затем производится вычисление первых четырех значений x\_a и y\_a с шагом h по следующей формуле:

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

1. С помощью метода Draw\_Adams\_Table(x,y) в таблице выводятся значния. Так же давляются эти значения в лист X и лист Y соответсвенно.
2. Инициализация переменных x, y и d\_y, x будет равен х\_a, а y будет равен y\_a. И находится количество отрезков разбиения n1 по следующей формуле:

1. С использованием цикла по индексу (i=0) производится вычисление коэффициентов по следующим формулам:

k1 = f(x3, y3) \* h;

k2 = f(x2,y2) \* h;

k3 = f(x1,y1) \* h;

k4 = f(x1, y0) \* h;

Постоянный шаг по x равен h. Производится вычисление постоянного шага по y d\_y по следующей формуле:

d\_y = (55 \* k1 - 59 \* k2 + 37 \* k3 - 9 \* k4) / 24

С помощью метода Draw\_Adams\_Table(x,y) в таблице выводятся значния. С использовнием вложенного цикла по индексу (j=1) фиксируются предыдущие значения.

1. Инициализация переменной result\_adams, она будет содержать время работы метода Адамса. Для этого из настоящего времени работу метода и переводит в миллисекунды с помощью функции TotalMilliseconds. И с помощью метода Time\_for\_Adams(“ ”,” “, result\_adams), который на вход получает пустые строки и значение времени, выводит общее время работы.

Блок-схема метода Адамса представлена на рисунке 3.

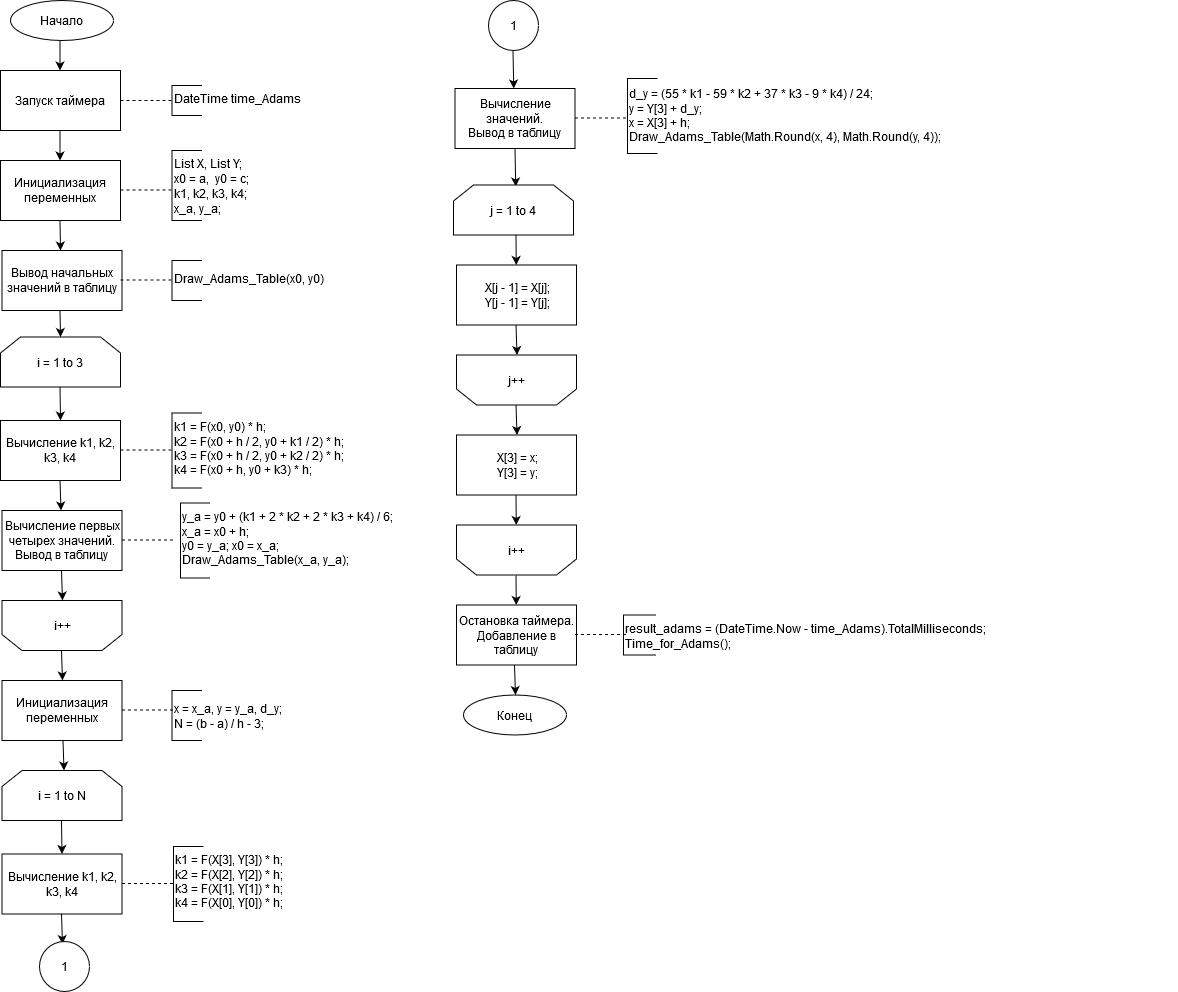


Рисунок 3 – Блок-схема метода Адамса

# **Тестирование программы**

Приложение позволяет вводить границы промежутка, значение y0 и шаг. Так же при нажатии кнопки вычислить осуществляются вычисления, а при нажатии кнопки Очистить – очищаются все поля. Тестирование приложения представлено на рисунках 4 - 7.

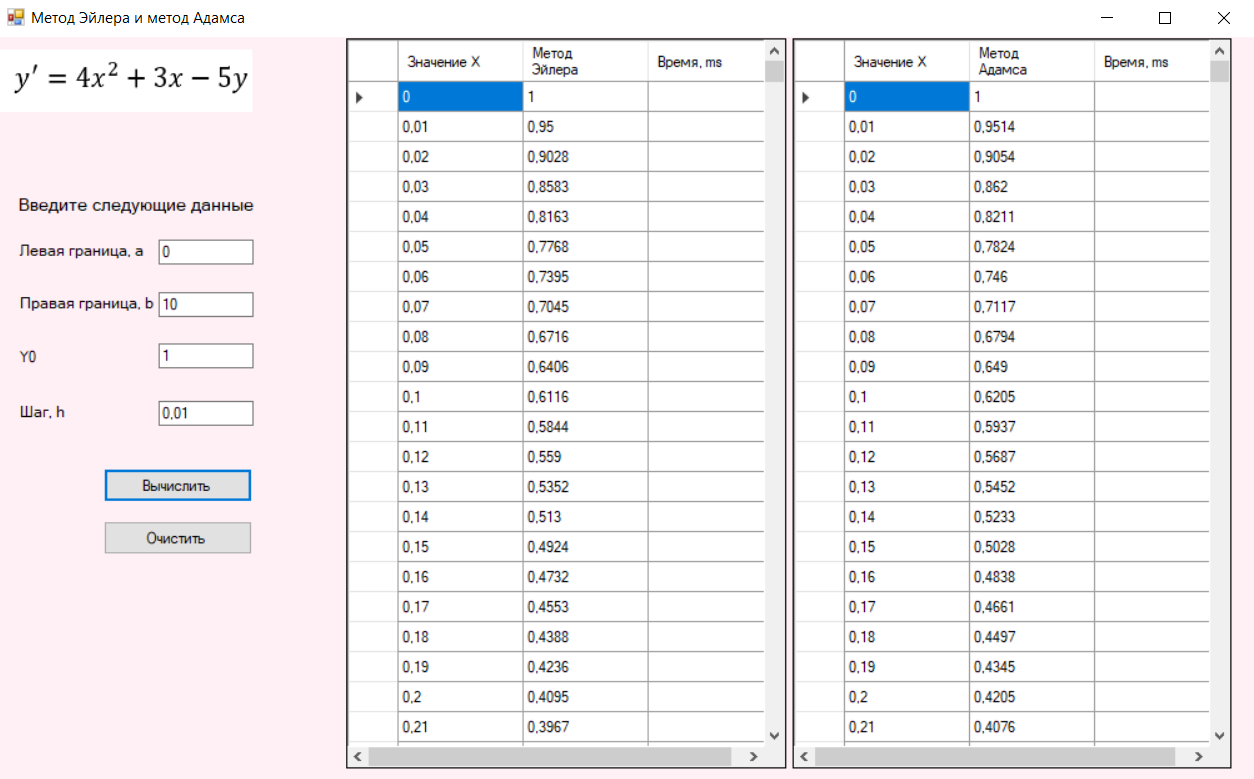


Рисунок 4 – Тестирование программы 1

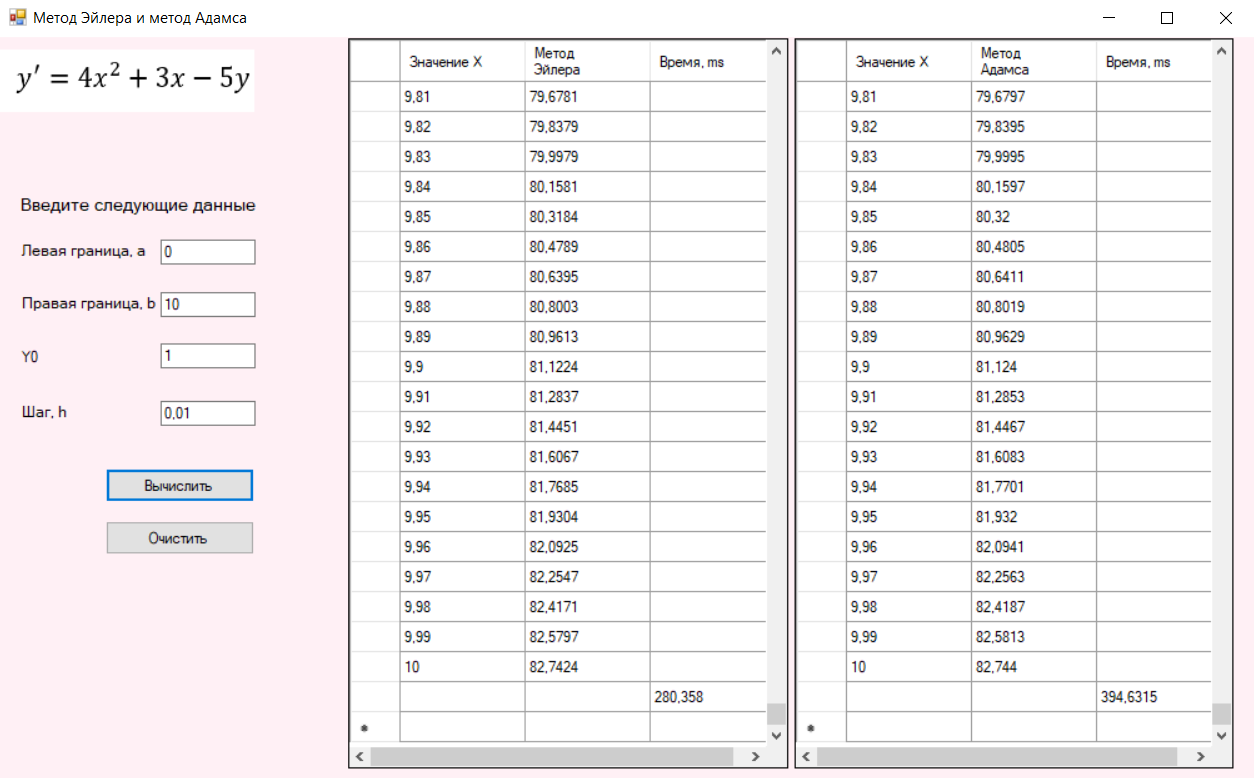


Рисунок 5 – Тестирование программы 1 (продолжение)

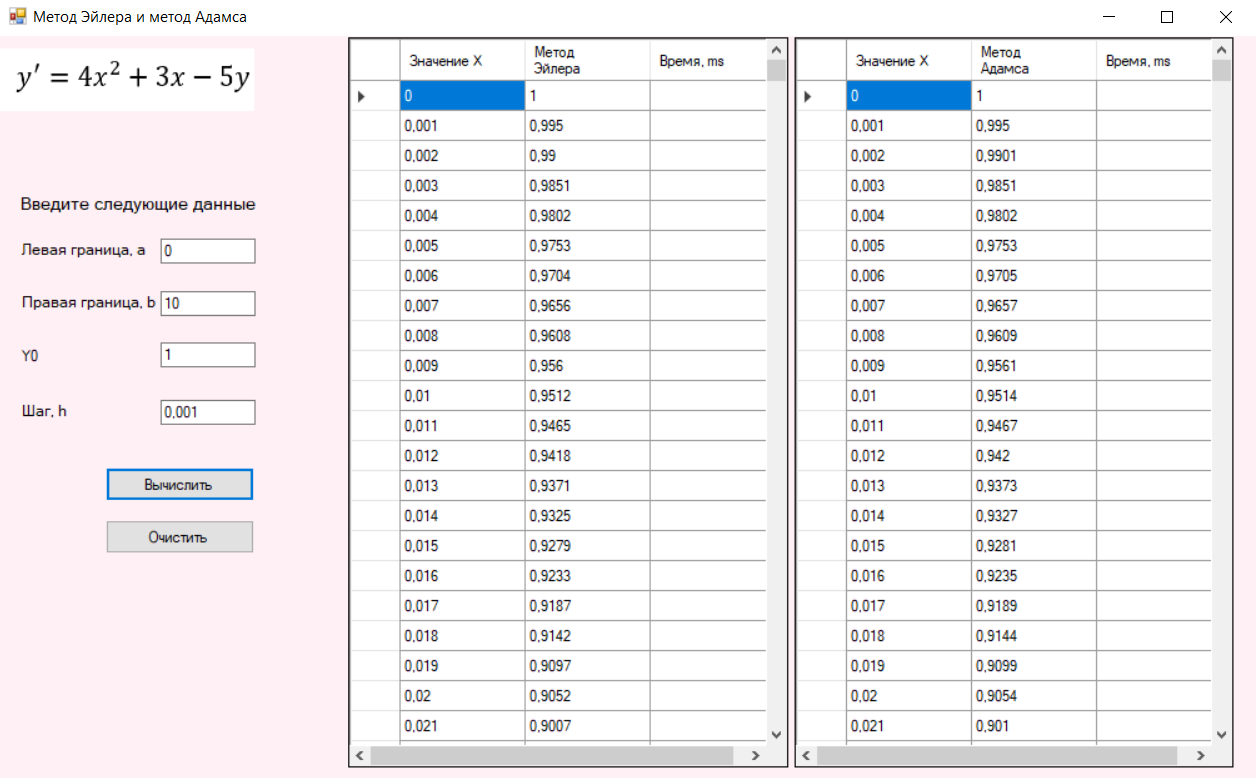


Рисунок 6 – Тестирование программы 2

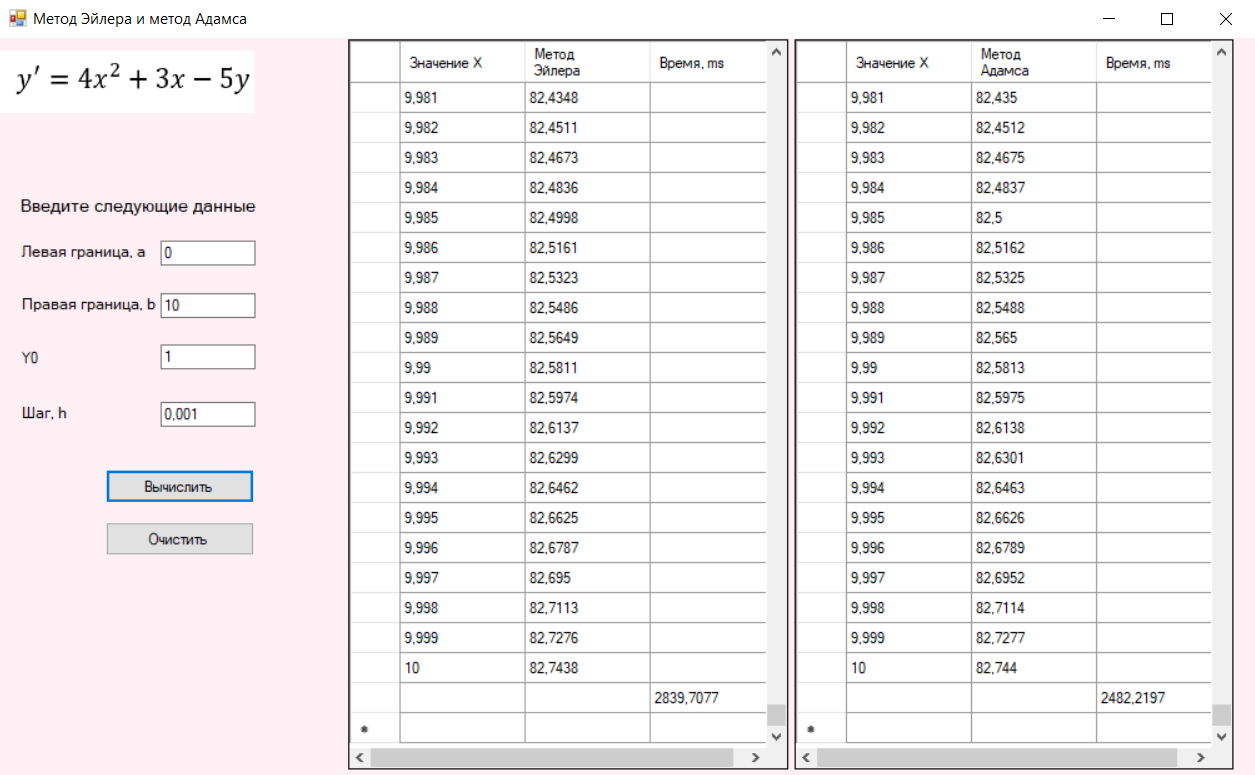


Рисунок 7 – Тестирование программы 2 (продолжение)

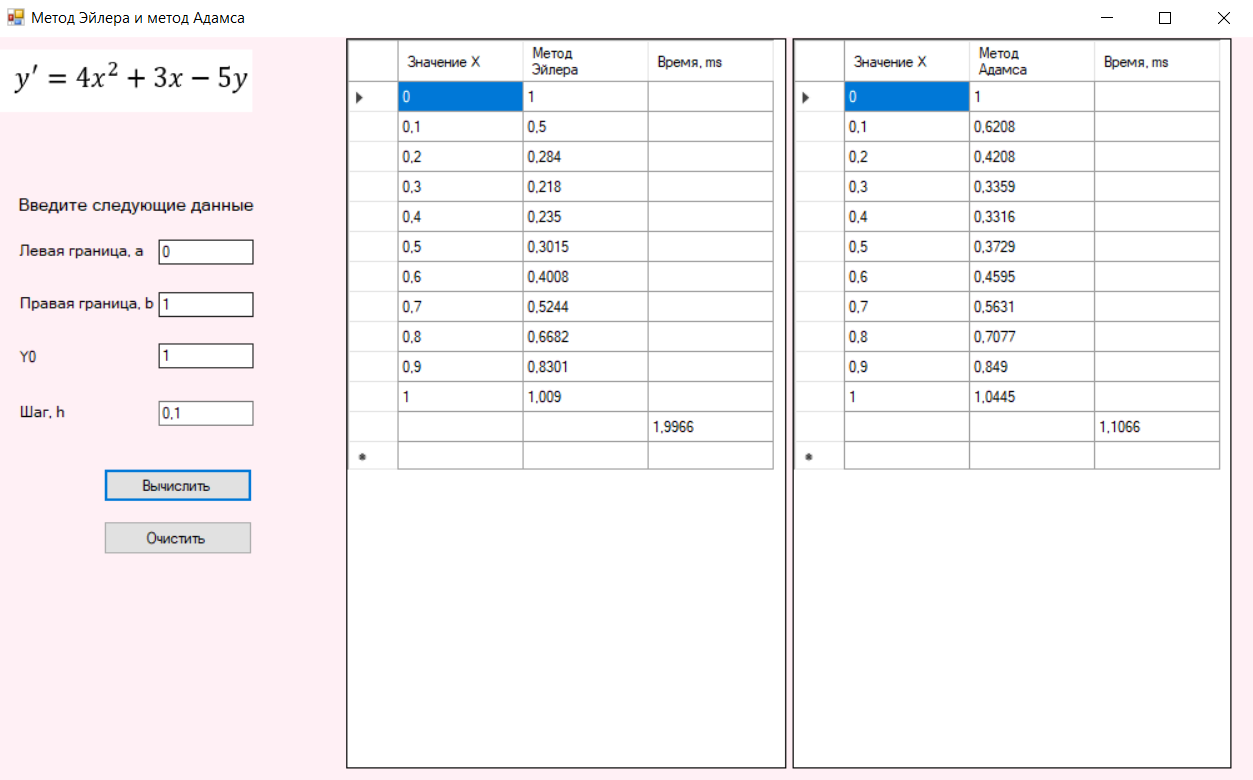


Рисунок 8 – Тестирование программы 3

# **Проверка корректности работы программы**

Приложение состоит из двух методов: метод Эйлера и метод Адамса. Для проверки корректности работы метода Эйлера воспользуемся онлайн калькулятором на сайте [www.mathstools.com](http://www.mathstools.com). Входные данные для проверки будут следующие:

* Шаг h = 0,01;
* Левая граница промежутка x0=0;
* Правая граница промежутка xn=10;
* Начальное значение y0=1;

C результатом вычислений можно ознакомиться на рисунках 7-8.

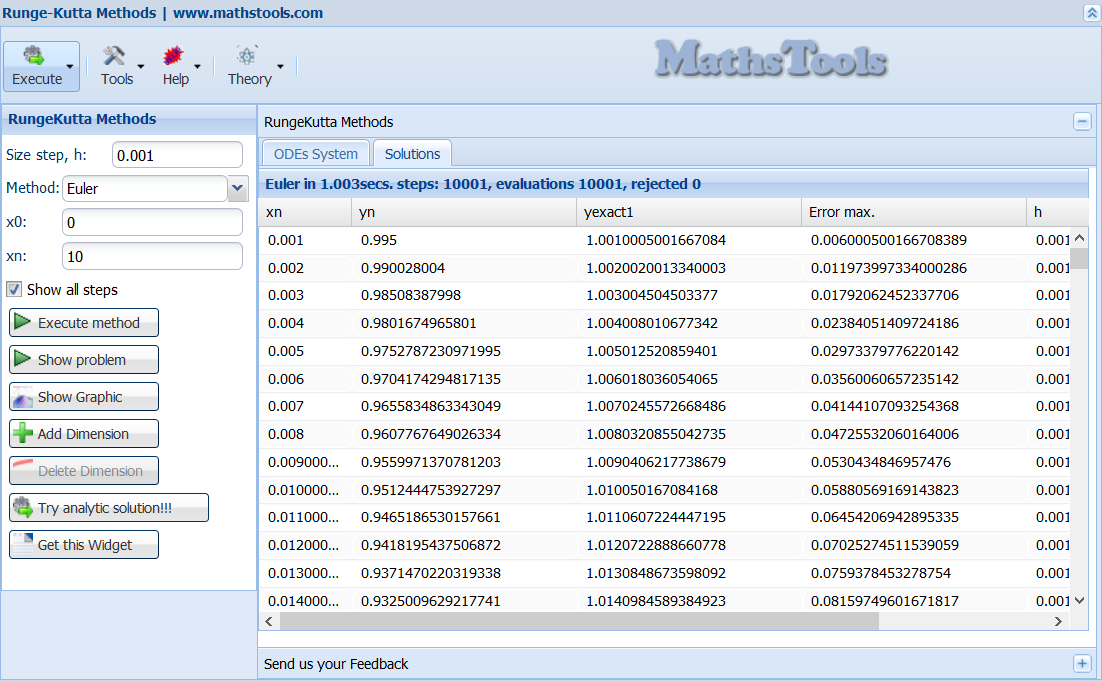


Рисунок 7 –Метод Эйлера с использованием сторонних сервисов

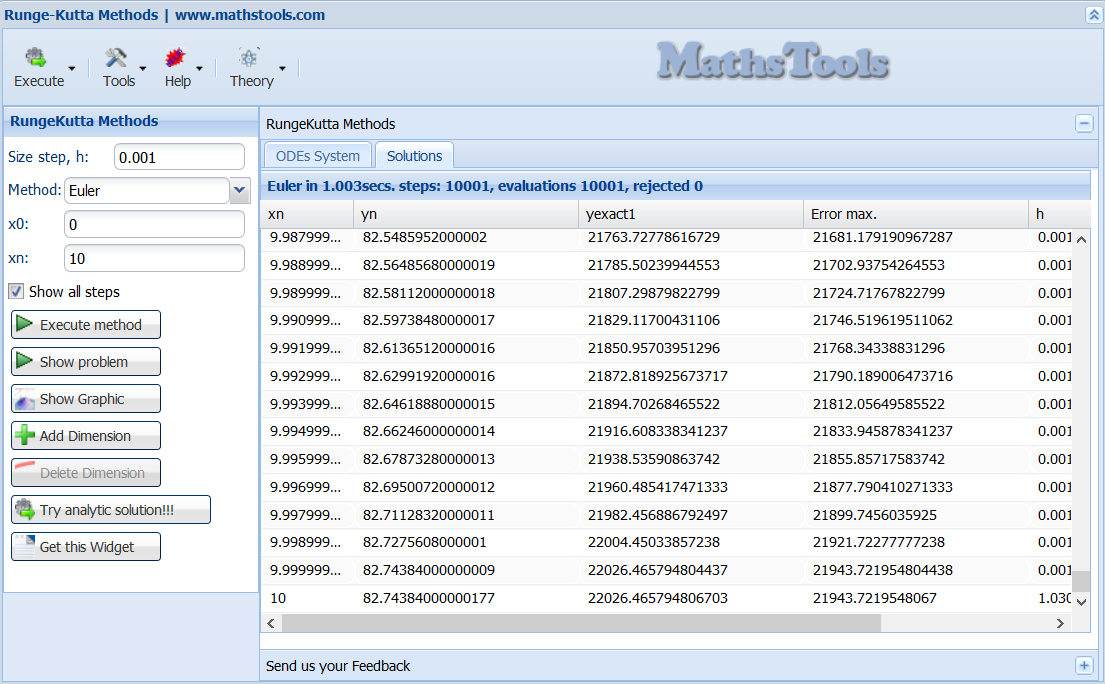


Рисунок 8 – Метод Эйлера с использованием сторонних сервисов, продолжение

Для проверки корректности работы метода Адамса воспользуемся онлайн калькулятором на сайте www.atozmath.com . Входные данные для проверки метода Адамса:

* Шаг h = 0,1;
* Левая граница промежутка x0=0;
* Начальное значение y0=1;
* Найти значение в точке x=0,9;

C результатом вычислений можно ознакомиться на рисунках 9-11.

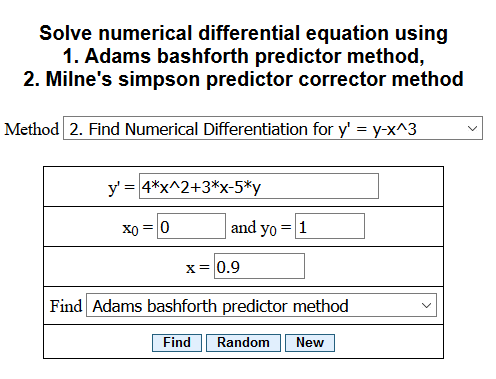


Рисунок 9 – Входные параметры метода Адамса

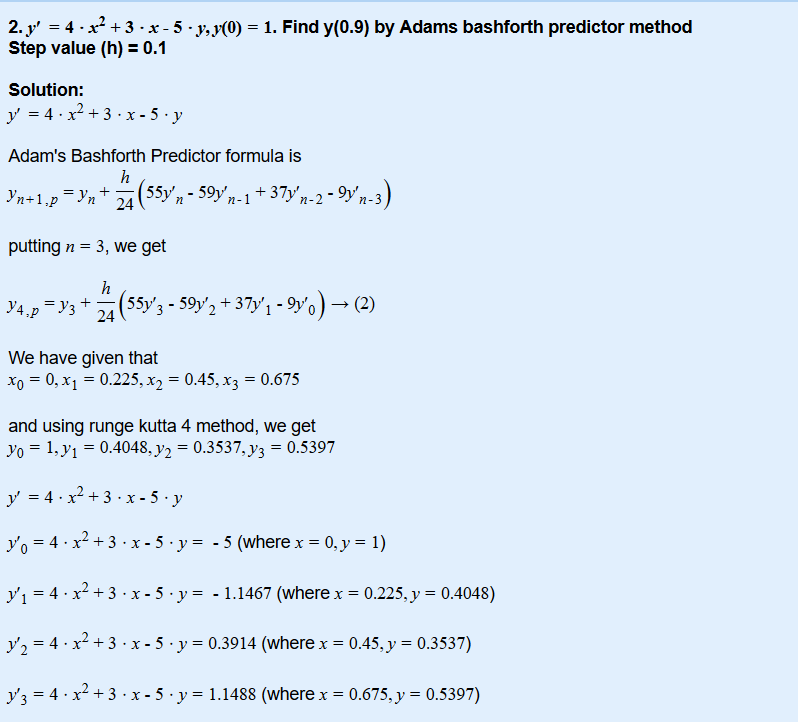


Рисунок 10 - Метод Адамса с использованием сторонних сервисов

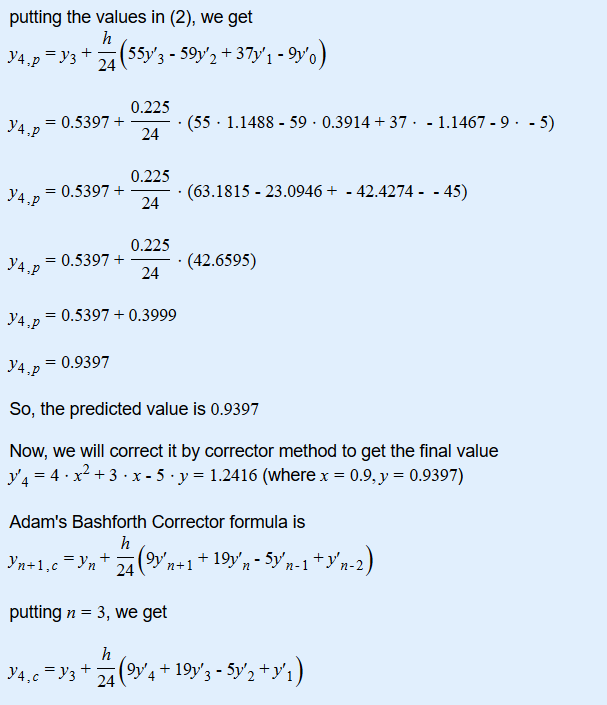


Рисунок 11 – Метод Адамса с использованием сторонних сервисов, продолжение

# **Сравнительный анализ методов решения дифференциальных уравнений по критериям точности, вычислительной сложности.**

## **Погрешность метода Эйлера**

Погрешность на шаге или локальная погрешность — это разность между численным решением после одного шага вычисления y i {\displaystyle y\_{i}} и точным решением в точке x i = x i − 1 + h {\displaystyle x\_{i}=x\_{i-1}+h} . Численное решение задаётся формулой:

Точное решение можно разложить в ряд Тейлора:

y ( x i − 1 + h ) = y ( x i − 1 ) + h y ′ ( x i − 1 ) + O ( h 2 ) . {\displaystyle y(x\_{i-1}+h)=y(x\_{i-1})+hy'(x\_{i-1})+O(h^{2}).} Локальную ошибку L {\displaystyle L} получаем, вычитая из второго равенства первое:

L = y ( x i − 1 + h ) − y i = O ( h 2 ) . {\displaystyle L=y(x\_{i-1}+h)-y\_{i}=O(h^{2}).}

Это справедливо, если y y {\displaystyle y} имеет непрерывную вторую производную. Другим достаточным условием справедливости этой оценки, из которого вытекает предыдущее и которое обычно может быть легко проверено, является непрерывная дифференцируемость f ( x , y ) {\displaystyle f(x,y)} по обоим аргументам.

Погрешность в целом, глобальная или накопленная погрешность — это погрешность в последней точке произвольного конечного отрезка интегрирования уравнения. Для вычисления решения в этой точке требуется S / h {\displaystyle S/h} шагов, где S {\displaystyle S} S длина отрезка. Поэтому глобальная погрешность метода

G = O(G = O ( h 2 S / h ) = O ( h ) {\displaystyle G=O(h^{2}S/h)=O(h)}

Таким образом, метод Эйлера является методом первого порядка — имеет погрешность на шаге O ( h 2 ) {\displaystyle O(h^{2})} и погрешность в целом O ( h ) {\displaystyle O(h)} .

Метод Эйлера является сравнительно грубым и применяется в основном для ориентировочных расчетов.

## **Погрешность метода Адамса**

Когда правая часть уравнения сложное аналитическое выражение применяется метод Адамса, который не требует многократного подсчета правой части уравнения.

Для грубой оценки погрешности применяют принцип Рунге, который состоит в следующем:

1. Находят решение дифференциального уравнения при шаге *h*.
2. Значение шага удваивают и находят решение при шаге *Н = 2h*.

3. Вычисляют погрешность метода по формуле

,

где  - значение приближенного вычисления при двойном шаге *H=2h*;  - значение приближенного вычисления при шаге *h*.

Метод Адамса применяется также и для решения систем дифференциальных уравнений и дифференциальных уравнений *n*-гo порядка.

Погрешность можно представить в более общем виде - .

## **Сравнительный анализ**

Исследования проводились на ЭВМ со следующими характеристиками:

* процессор – Intel Core i5 8250U (Kaby Lake R);
* оперативная память – 8 ГБ DDR4-2400 МГц;
* операционная система – Windows 10;
* видеокарта – NVIDIA GeForce® GTX 1050, 2 ГБ
* встроенная графика - Intel UHD Graphics 620, 64+1632 МБ.

Для проведения сравнительного анализа было сделано более 20 тестирований, в которых задавились разные значения промежутка и точность шага. Результаты тестирования приведены в Таблице 1.

Таблица 1 – Сравнительная таблица

|  |  |  |  |
| --- | --- | --- | --- |
| Шаг | Погрешность | Среднее время выполнения | |
|  | | Метод Эйлера, мс | Метод Адамса, мс |
| 0,1 |  | 2,5026 | 1,9971 |
| 0,01 |  | 29,3686 | 26,2369 |
| 0,001 |  | 353,7897 | 341,8765 |

Можно сделать вывод, что метод Адамас работает быстрее метода Эйлера. Значит, эффективность вычисления метода Адамса выше, чем метода Эйлера.

# **ЗАКЛЮЧЕНИЕ**

В результате выполнения данной работы было разработано приложение для решения численными методами дифференциальных уравнений. Данное приложение вычисляет дифференциальное уравнение методом Эйлера и методом Адамса. Было проведено сравнение, исходя из времени и погрешности. Исследования эффективности приложения показали, что медленным методом оказался метод Эйлера.

Был получен навык анализа различных численных методов решения дифференциальных уравнений, разработки программных средств для решения численными методами дифференциальных уравнений.

# **СПИСОК ЛИТЕРАТУРЫ**

1. ГОСТ 19.106-78 ЕСПД. Требования к программным документам, выполненным печатным способом.
2. ГОСТ 19.401-78 ЕСПД. Текст программы. Требования к содержанию и оформлению.
3. Эйлер Л. Интегральное исчисление. Том 1. — М.: ГИТТЛ. 1956.
4. [Бабенко К. И.](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B1%D0%B5%D0%BD%D0%BA%D0%BE,_%D0%9A%D0%BE%D0%BD%D1%81%D1%82%D0%B0%D0%BD%D1%82%D0%B8%D0%BD_%D0%98%D0%B2%D0%B0%D0%BD%D0%BE%D0%B2%D0%B8%D1%87) Основы численного анализа. — М.: Наука. 1986.
5. Алиева В. Ф., Кувайскова Ю. Е. Исследование эффективности численных методов решения задачи Коши //Математические методы и модели: теория, приложения и роль. – 2016. – С. 94.
6. Мокроусова Т. А., Шаймухаметова Д. В. Численные методы решения обыкновенных дифференциальных уравнений//biological sciences. – 2017.

# **ПРИЛОЖЕНИЕ**

**Form1.сs**

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.Diagnostics;

namespace KR\_CHM

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

static double F(double x, double y)

{

return 4 \* x \* x + 3 \* x - 5 \* y;

}

public void method\_Euler(double a, double b, double c, double h)

{

DateTime time\_Euler = DateTime.Now;

double n = (b - a) / h;

double[] X1 = new double[Convert.ToInt32(n + 1)];

double[] Y1 = new double[Convert.ToInt32(n + 1)];

Y1[0] = c;

X1[0] = a;

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

{

X1[i] = a + i \* h;

Y1[i] = Y1[i-1] + h \* F(X1[i-1], Y1[i-1]);

}

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

{

Draw\_Euler\_Table(Math.Round(X1[i],4), Math.Round(Y1[i],4));

}

double result\_euler = (DateTime.Now - time\_Euler).TotalMilliseconds;

Time\_for\_Euler(" "," ", result\_euler);

}

public void Time\_for\_Euler(string s, string m,double k)

{

Euler\_gridview.Rows.Add(s,m,k);

}

public void Time\_for\_Adams(string s1, string m1, double m2)

{

Adams\_gridview.Rows.Add(s1, m1, m2);

}

public void method\_Adams(double a, double b, double c, double h)

{

DateTime time\_Adams = DateTime.Now;

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

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

double x0 = a;

double y0 = c;

X.Add(x0);

Y.Add(y0);

Draw\_Adams\_Table(x0, y0);

double k1 = 0;

double k2 = 0;

double k3 = 0;

double k4 = 0;

double x\_a = 0;

double y\_a = 0;

for (double i = 0; i < 3; i++)

{

k1 = F(x0, y0) \* h;

k2 = F(x0 + h / 2, y0 + k1 / 2) \* h;

k3 = F(x0 + h / 2, y0 + k2 / 2) \* h;

k4 = F(x0 + h, y0 + k3) \* h;

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

x\_a = x0 + h;

y0 = y\_a;

x0 = x\_a;

Draw\_Adams\_Table(Math.Round(x\_a, 4), Math.Round(y\_a, 4));

X.Add(x0);

Y.Add(y0);

}

double x = x\_a, y = y\_a, d\_y;

double N = (b - a) / h - 3;

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

{

k1 = F(X[3], Y[3]) \* h;

k2 = F(X[2], Y[2]) \* h;

k3 = F(X[1], Y[1]) \* h;

k4 = F(X[0], Y[0]) \* h;

d\_y = (55 \* k1 - 59 \* k2 + 37 \* k3 - 9 \* k4) / 24;

y = Y[3] + d\_y;

x = X[3] + h;

Draw\_Adams\_Table(Math.Round(x, 4), Math.Round(y, 4));

for (int j = 1; j < 4; j++)

{

X[j - 1] = X[j];

Y[j - 1] = Y[j];

}

X[3] = x;

Y[3] = y;

}

double result\_adams = (DateTime.Now - time\_Adams).TotalMilliseconds;

Time\_for\_Adams(" ", " ", result\_adams);

}

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

{

double a = Convert.ToDouble(left\_textBox.Text);

double b = Convert.ToDouble(right\_textBox.Text);

double c = Convert.ToDouble(y0\_textBox.Text);

double h = Convert.ToDouble(step\_textBox.Text);

method\_Euler(a, b, c, h);

method\_Adams(a, b, c, h);

}

public void Draw\_Euler\_Table(double x, double y)

{

Euler\_gridview.Rows.Add(x, y);

}

public void Draw\_Adams\_Table(double x, double y)

{

Adams\_gridview.Rows.Add(x, y);

}

private void button1\_Click\_1(object sender, EventArgs e)

{

Euler\_gridview.Rows.Clear();

Adams\_gridview.Rows.Clear();

left\_textBox.Clear();

right\_textBox.Clear();

step\_textBox.Clear();

y0\_textBox.Clear();

}

}

}