МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

**«Нижегородский государственный университет**

**им. Н.И. Лобачевского»**

**Национальный исследовательский университет**

**Кафедра дифференциальных уравнений, математического и численного анализа**

**ЛАБОРАТОРНАЯ РАБОТА**

**«Численное решение задачи Коши для ОДУ.»**

**Выполнил:** студент группы 381706-2

Танский Юрий Игоревич

\_\_\_\_\_\_\_\_\_\_\_ Подпись

**Руководитель:**

Эгамов Альберт Исмаилович

\_\_\_\_\_\_\_\_\_\_\_ Подпись

Нижний Новгород

2020

**Введение**

Наиболее эффективными и часто встречаемыми методами решениями задачи Коши являются методы Рунге - Кутта. Они основаны на аппроксимации искомой функции у(х) в пределах каждого шага многочленом, который получен при помощи разложения функции у(х) в окрестности шага h каждой i-ой точки в ряд Тейлора.  
К классу методов Рунге — Кутты относятся явный метод Эйлера и модифицированный метод Эйлера с пересчётом, которые представляют собой соответственно методы первого и второго порядка точности. Существуют стандартные явные методы третьего порядка точности, не получившие широкого распространения. Наиболее часто используется и реализован в различных математических пакетах (Maple, MathCAD, Maxima) классический метод Рунге — Кутты, имеющий четвёртый порядок точности. При выполнении расчётов с повышенной точностью всё чаще применяются методы пятого и шестого порядков точности. Построение схем более высокого порядка сопряжено с большими вычислительными трудностями.

В данной лабораторной работе в качестве реализуемого дифференциального уравнения второго порядка было взято уравнение маятника с диссипацией 𝑥̈+𝛿𝑥̇+sin𝑥=0.

Для численного решения задачи Коши был выбран метод Рунге-Кутта 4-ого порядка.  
В качестве среды разработки был выбран пакет Visual Studio 2019.  
В качестве языка разработки был выбран C# с поддержкой приложения с пользовательским интерфейсом Windows Forms

**Постановка задачи**

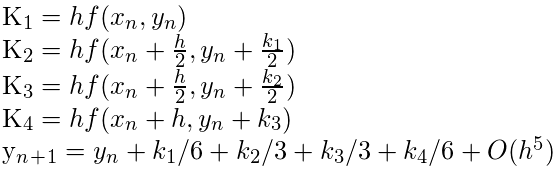
Дано дифференциальное уравнение и начальное условие, то есть поставлена задача Коши:

https://www.bestreferat.ru/images/paper/21/66/9816621.png

Требуется отыскать интегральную кривую, удовлетворяющую поставленной задаче Коши с помощью метода Рунге-Кутты четвертого порядка с автоматическим выбором шага на отрезке. Задачу можно решить аналитически, найдя решение дифференциального уравнения и подставив в него начальное условие, тем самым, отыскав требуемую интегральную кривую. Но для нас интерес представляет решение данной задачи с применением численного метода, а конкретнее – метода Рунге-Кутты 4-го порядка с автоматическим выбором шага, то есть численное решение. Автоматический выбор шага – необходимое условие адекватного поведения программы при резко изменяющихся функциях, задающих интегральную кривую, позволяющее отразить все моменты в поведении интегральной кривой и добиться высокой точности.  
  
Метод Рунге-Кутты находит приблизительное значение y для данного x. Только обыкновенные дифференциальные уравнения первого порядка могут быть решены с помощью метода 4-го порядка Рунге Кутты.

Ниже приведена формула, используемая для вычисления следующего значения yn+1 из предыдущего значения yn. Значения n равны 0, 1, 2, 3, (x — x0) / h. Здесь h — высота шага, а xn+1 = x0 + h

Меньший размер шага означает большую точность.



Формула в основном вычисляет следующее значение yn+1, используя текущее значение yn плюс средневзвешенное значение четырех приращений.

Этот метод является методом четвертого порядка, это означает, что локальная ошибка усечения имеет порядок O (h5), в то время как общая накопленная ошибка составляет порядок O (h4).

**Руководство программиста**

Для работоспособности программного кода необходимо подключить следующие стандартные библиотеки, включенные в пакет C# VS 2019:  
using System;  
using System.Windows.Forms;

using System.Globalization;  
Дополнительные внешние компоненты в программе не используются.  
Ниже приведен листинг основного метода программы и его описание:

RunRungeKutt() - Основная функция программы, в которой происходят вычисление координат точек фазового портрета. Вызов функции происходит после проверок корректного ввода параметров пользователем.  
x,z – соответствующие массивы для координат. Z – координаты функции, полученные по методу Рунге-Кутты  
h – шаг интегрирования дифференциального уравнения,  
n – количество итераций алгоритма.  
b - конец интервала интегрирования  
private void RunRungeKutt()

{

x = new double[1000000];

z = new double[1000000];

x[0] = a;

z[0] = 0;

if (checkBox1.Checked == true)

{

h = (b - a) / 5000;

}

n = Convert.ToInt32((b - a) / h);

double T1, T2, T3, T4, U1, U2, U3, U4;

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

{

U1 = h \* z[i];

U2 = h \* (z[i] + U1 / 2);

U3 = h \* (z[i] + U2 / 2);

U4 = h \* (z[i] + U3);

T1 = h \* f1(x[i], z[i]);

T2 = h \* f1(x[i] + h / 2, z[i] + T1 / 2);

T3 = h \* f1(x[i] + h / 2, z[i] + T2 / 2);

T4 = h \* f1(x[i] + h, z[i] + T3);

x[i+1] = x[i] + (U1 + 2 \* U2 + 2 \* U3 + U4) / 6;

z[i+1] = z[i] + (T1 + 2 \* T2 + 2 \* T3 + T4) / 6;

}   
ChartGraph(x, z, n);

}

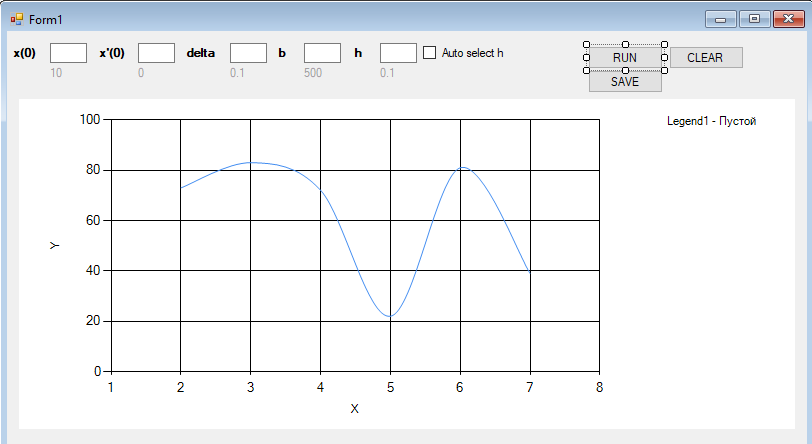
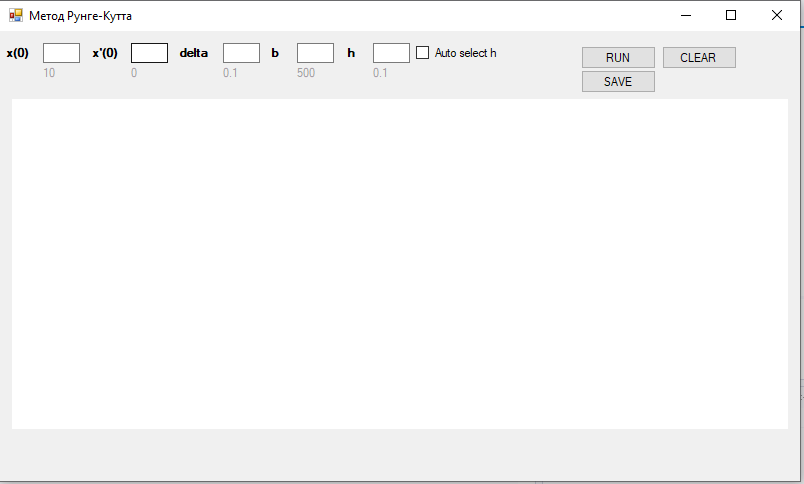
Функция также реализует автоматический подсчет шага алгоритма.  
В данной программе в качестве шага интегрирования по умолчанию был выбран шаг 1/5000 от длины отрезка интегрирования. Подобный шаг был выбран опытным путем, сохраняя быстродействие и оптимальность работы программы.  
После нахождения соответствующих координат, выполняется процедура построения фазового портрета.  
  
Конструктор формы имеет следующий вид (рис. 1)  


Рисунок 1. Конструктор формы

Для сохранения возможности запуска программы и избежания конфликтов подключения дополнительных библиотек на различных операционных системах и архитектурах в качестве решения для построения графиков в программе был использован стандартный контейнер Chart, который включен в пакет C# VS 2019.   
В качестве взаимодействия с пользователем используются следующие элементы формы:  
Textbox – для интерактивного ввода пользователем параметров алгоритма  
Checkbox – для переключения пользователем режима расчета шага интегрирования с ручного на автоматический.  
Button – кнопки для интерактивного запуска пользователем функций алгоритма  
Label – информация для пользователя.   
  
Описание всех элементов формы и дополнительные особенности будут приведены в руководстве пользователя.

**Руководство пользователя**

Вход в программу.  
Стартовое окно пользователя (рис.2)

  
Рисунок 2. Стартовое окно пользователя.

При входе в программу пользователю необходимо будет указать параметры алгоритма.  
Пользователь может выбрать параметры алгоритма по умолчанию, которые предоставляет ему разработчик - для этого необходимо нажать на число серого цвета под полем ввода и указанное число подставится автоматически. Или же пользователь может вручную указать параметры.  
x(0), x’(0) – начальные условия  
delta – константа, используемая в диф. уравнении 𝑥̈+𝛿𝑥̇+sin𝑥=0  
h – шаг интегрирования  
b – конец отрезка интегрирования  
Чек бокс “Auto Select” – если установить галочку, то расчет шага интегрирования будет производиться автоматически, также будет недоступно указать шаг вручную.  
Ответственность за корректность входных данных при работе с программой лежит на разработчике, именно поэтому в случае, если пользователь неверно указал один из входных параметров и попытается запустить выполнение метода Рунге-Кутта по кнопке “RUN”, то программа не даст ему это сделать и вызовет соответствующее диалоговое сообщение

(рис. 3). В качестве примера, при вводе намеренно была допущена ошибка для параметра “b”

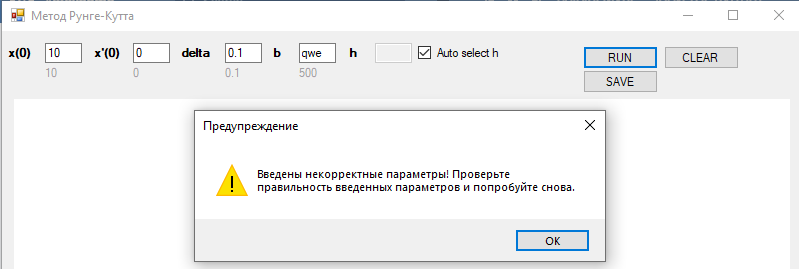
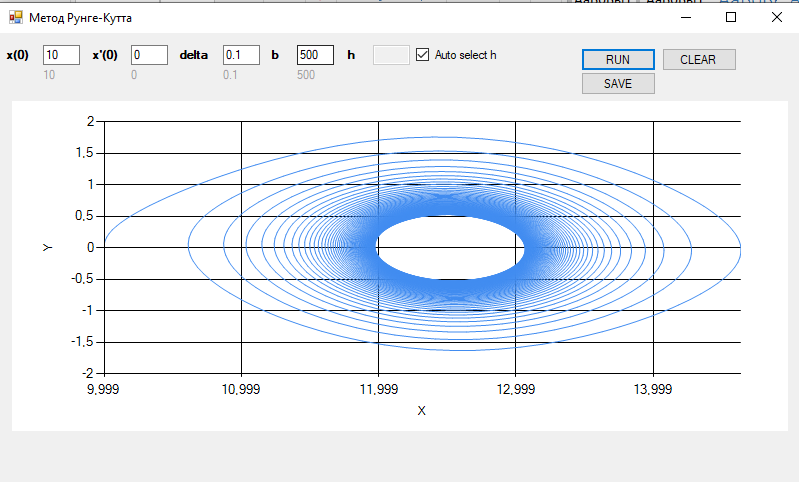
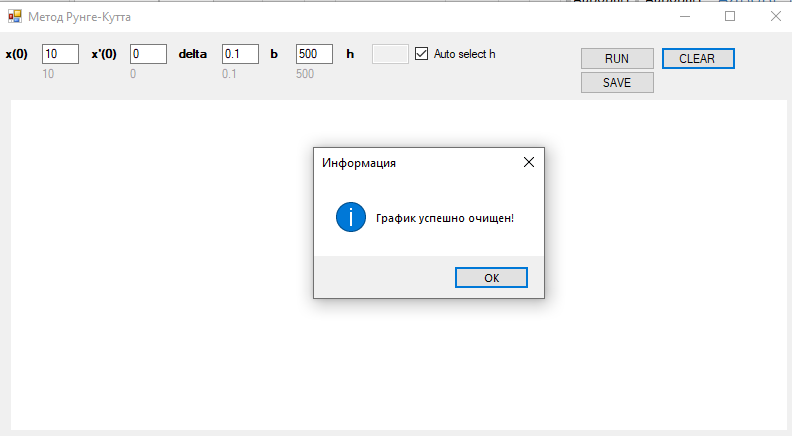


Рисунок 3. Некорректный ввод параметров алгоритма.

После заполнения параметров алгоритма необходимо нажать кнопку “RUN” – программа выполнит реализацию метода Рунге-Кутта для заданных входных данных, а также построит фазовый портрет (рис. 4)

  
Рисунок 4. Фазовый портрет

Стоит обратить внимание, что программа запоминает вводимые пользователем входные данные после запуска алгоритма и отображает их в виде чисел серого оттенка под полем ввода.  
  
В любой момент пользователь может очистить построенный фазовый портрет по кнопке “CLEAR”, а также получить соответствующее диалоговое сообщение (рис. 5)

  
Рисунок 5. Очистка графика.

Пользователь может сохранить вычисленные координаты в текстовый файл, нажав кнопку “SAVE”  
Перед ним откроется диалоговое окно для выбора директории файла. После выбора директории пользователь нажимает “Сохранить”, программа записывает координаты в выбранный файл и выдает соответствующее текстовое сообщение (рис. 6.1, 6.2, 6.3)

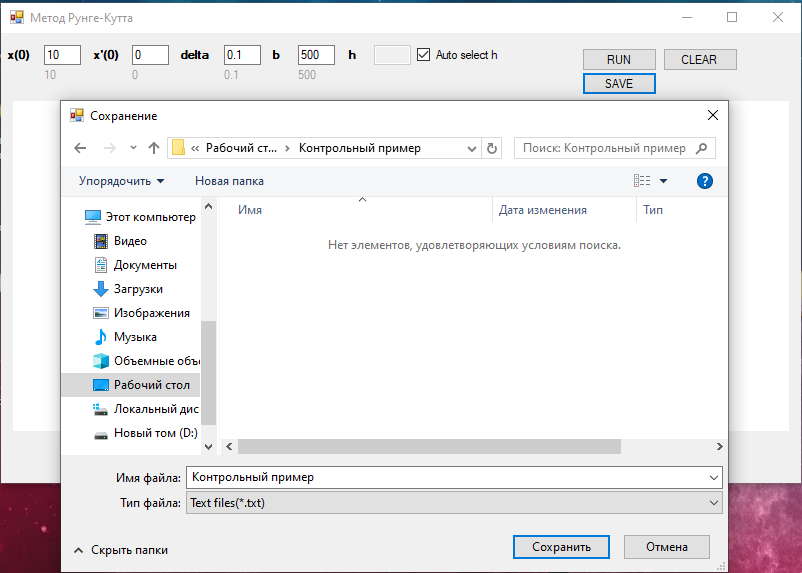


Рисунок 6.1. Выбор директории файла.

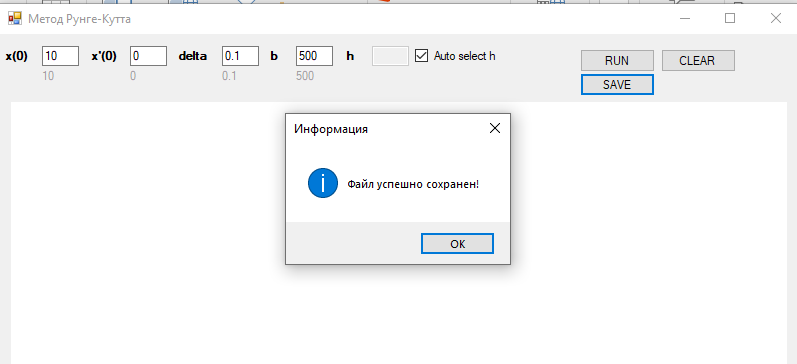
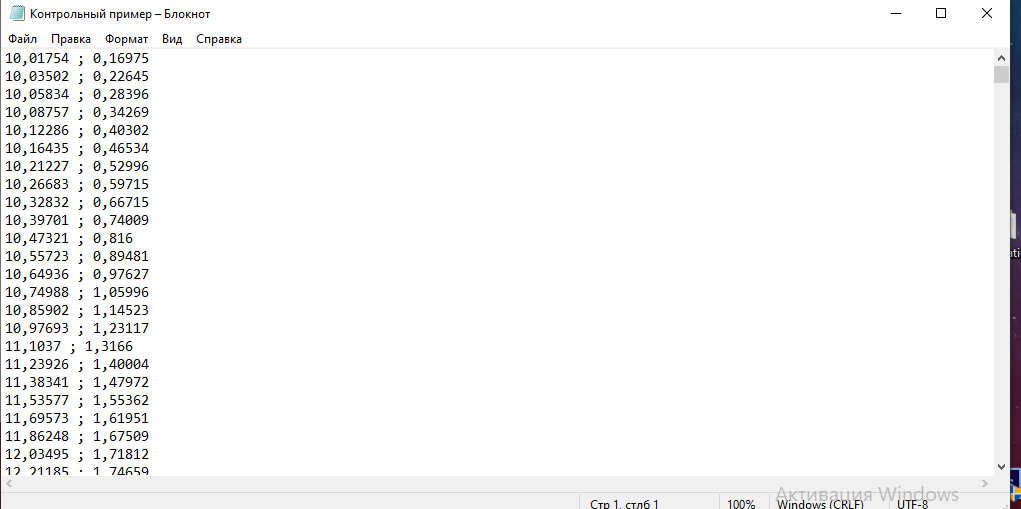


Рисунок 6.2. Информационное сообщение.  
  
Рисунок 6.3 Содержимое файла.

**Заключение**

В данной лабораторной работе в численном виде была решена задача Коши для автономного ОДУ второго порядка с использованием компьютера, программного пакета C#, включенное в среду разработки Visual Studio 2019  
В работе детально рассмотрен метод Рунге-Кутты четвертого порядка с автоматическим выбором длины шага, приведены необходимые теоретические сведения.  
Был реализован интуитивно понятный пользовательский интерфейс с возможными подсказками пользователю.

**Список литературы**

1. Хайрер Э., Нёрсетт С., Ваннер Г. «Решение обыкновенных дифференциальных уравнений. Нежесткие задачи», М., Мир, 1990, 512с.

2. А. А. Самарский “Введение в численные методы”. Издание третье., Москва, “Лань”, 2005, 288с.