Федеральное агентство связи

Уральский технический институт связи и информатики (филиал) ФГБОУ ВО "Сибирский государственный университет телекоммуникаций и информатики" в г. Екатеринбурге (УрТИСИ СибГУТИ)

Курсовая работа

По дисциплине: информатика

на тему: **«**Визуализация численных методов.

Решение Обыкновенных дифференциальных уравнений**»**

Выполнил:

студент группы ОЕ-61б

Захаров К.С.

Проверил:

Бикбулатова Н.Г

Екатеринбург, 2017 год

Содержание

[Принцип автоматической обработки информации вычислительным устройством (Принципы фон Неймана) …………………………………………………………..3](#_Toc485175521)-4

[Введение 5](#_Toc485175522)

1.[Постановка Задачи 6](#_Toc485175523)-7

2.[Описание используемых методов 8](#_Toc485175524)

[2.1.Метод Эйлера. 8](#_Toc485175525)-10

2.2[Метод Рунге – Кутта 4-го порядка 11](#_Toc485175526)-13

3.[Блок схемы основных процедур 14](#_Toc485175527)-16

[4.Форма программы 17](#_Toc485175529)-18

5.[Решение в MathCad. 19](#_Toc485175530)

[Библиография. 20](#_Toc485175531)

[Заключение. 21](#_Toc485175532)

[Приложение 22](#_Toc485175533)

# Введение

Существует множество технических систем и технологических процессов, характеристики которых непрерывно меняются со временем **t**. Такие явления обычно подчиняются физическим законам, которые формулируются в виде дифференциальных уравнений.

Дифференциальными называются уравнения, содержащие одну или несколько производных. Лишь очень немногие из них удаётся решить без помощи вычислительной техники. Поэтому численные методы решения дифференциальных уравнений играют важную роль в практике инженерных расчётов.

В зависимости от числа независимых переменных и типа входящих в них производных дифференциальные уравнения делятся на две существенно различные категории: обыкновенные, содержащие одну независимую переменную и производные по ней, и уравнения в частных производных, содержащие несколько независимых переменных и производные по ним, которые называются частными. В этой главе рассматриваются методы решения обыкновенных дифференциальных уравнений.

# 1.Приниип автомитической обработки информации вычислительным устройством

Основным отличием вычислительной машины от таких счётных устройств как счёты, арифмометр, калькулятор, заключается в том, что вся последовательность команд на вычисление предварительно записывается в память вычислительной машины и выполняется последовательно автоматически. Впервые принцип вычислительной машины с автоматическим выполнением команд предложил американский учёный фон Нейман. Он описал основные узлы, которые должна содержать такая машина. Такой принцип получил название фон-неймановской вычислительной машиной. Большинство современных КС, в настоящее время, построено именно по этому принципу.

Машина фон Неймана состояла из памяти, представлявшей собой набор регистров, АЛУ, устройства ввода-вывода и устройства управления (рис. 2.1).



Рисунок 1.1 – Машина Фон Неймана

Устройство ввода передавало команды и данные в АЛУ, откуда они записывались в память. Все команды, совокупность которых называется программой, записываются в память в соседние ячейки по возрастанию их адресов, а данные, которые требуют обработки в ячейки с произвольными адресами. Последняя команда программы – это обязательно команда остановки работы. Каждая команда содержит код операции, которую необходимо выполнить и адреса ячеек, в которых находятся данные, обрабатываемые этой командой. Устройство управления содержит специальный регистр, который называется "Счётчик команд". После загрузки программы и данных в память в счётчик команд записывается адрес первой команды программы. После чего вычислительная машина переходит в режим автоматического выполнения программы.

Устройство управления считывает из памяти содержимое ячейки памяти, адрес которой находится в счётчике команд, и помещает его в специальное устройство – "Регистр команд". Регистр команд хранил команду во время её исполнения. Устройство управления расшифровывает тип операции команды, считывает из памяти данные, адреса которых указаны в команде и приступает к её выполнению. Для каждой команды устройство управления имеет свой алгоритм обработки, который заключается в выработке управляющих сигналов для всех остальных устройств машины. Этот алгоритм мог быть реализован на основе комбинационных логических схем или с помощью специальной внутренней памяти, куда эти алгоритмы были записаны в виде микрокоманд, объединённых в микропрограммы. Выполнение микропрограммы происходит по тому же принципу, что и программы в основной памяти, то есть по принципу фон Неймана. Каждая микрокоманда содержит набор управляющих сигналов для устройств машины. Отметим, что устройства управления выполнением команд процессоров в современных компьютерных системах так же строятся по принципу комбинационных схем или микропрограммных автоматов, в соответствии с чем, делятся на RISC и CISC процессоры, о которых будет рассказано ниже.

Микропрограмма выполнения любой команды обязательно содержит сигналы, изменяющие содержимого счётчика команд на единицу. Таким образом, после завершения выполнения очередной команды, счётчик команд указывал на следующую ячейку памяти, в которой находилась следующая команда программы. Устройство управления читает команду, адрес которой находится в счётчике команд, помещает её в регистр команд и т.д. Этот процесс продолжается до тех пор, пока очередная исполняемая команда не оказывается командой останова исполнения программы. Интересно отметить, что и команды и данные, находящиеся в памяти, представляют собой целочисленные двоичные наборы. Отличить команду от данных устройство управления не может, поэтому если программист забыл закончить программу командой останова, устройство управления читает следующие ячейки памяти, в которых уже нет команд программы, и пытается интерпретировать их как команды.

Особым случаем можно считать команды безусловного или условного перехода, когда требуется выполнить команду, не следующую по порядку за текущей, а отстоящую от данной на какое-то количество адресов. В этом случае команда перехода содержит адрес ячейки, куда требуется передать управление. Этот адрес записывается устройством управления непосредственно в счётчик команд и происходит переход на соответствующую команду программы

# 2.Постановка Задачи

Чтобы решить обыкновенное дифференциальное уравнение, необходимо знать значения зависимой переменной и (или) её производных при некоторых значениях независимой переменной. Если эти дополнительные условия задаются при одном значении независимой переменной, то такая задача называется задачей с начальными условиями, или задачей Коши. Часто в задаче Коши в роли независимой переменной выступает время.

Задачу Коши можно сформулировать следующим образом:

Пусть дано дифференциальное уравнение  и начальное условие y(x0) = y0. Требуется найти функцию y(x), удовлетворяющую как указанному уравнению, так и начальному условию.

Численное решение задачи Коши сводится к табулированию искомой функции. График решения дифференциального уравнения называется интегральной кривой.

Геометрический смысл задачи:

 - тангенс угла наклона касательной к графику решения в точке (x, y) к оси ОХ, - угловой коэффициент (рис.1).



*x*

Рисунок 1.1 – Геометрический смысл задачи Коши.

Существование решения:

Если правая часть f(x,y) непрерывна в некоторой области R, определяемой неравенствами

(x, y)

|x – x0| <a ; |y – y0| <b ,

то существует, по меньшей мере, одно решение y = y(x), определённое в окрестности |x – x0| <h, где h – положительное число.

Это решение единственно, если в R выполнено условие Липшица



где N – некоторая постоянная (константа Липшица), зависящая, в общем случае, от a и b. Если f(x, y)имеет ограниченную производную  в R, то можно положить

N = max|| при 

# 

# 3.Описание используемых методов

Численные методы решения задачи Коши

При использовании численных методов выполняется замена отрезка [x0, X] – области непрерывного изменения аргумента х множеством , состоящего из конечного числа точек x0<x1< … <xn = X– сеткой.

При этом xiназывают узлами сетки.

Во многих методах используются равномерные сетки с шагом



Задача Коши, определённая ранее на непрерывном отрезке [x0, X], заменяется её дискретным аналогом – системой уравнений, решая которую можно последовательно найти значения y1, y2, …, yn – приближённые значения функции в узлах сетки.



Численное решение задачи Коши широко применяется в различных областях науки и техники, и число разработанных для него методов достаточно велико. Эти методы могут быть разделены на следующие группы.

1. Одношаговые методы, в которых для нахождения следующей точки на кривой y = f(x) требуется информация лишь об одном предыдущем шаге. Одношаговыми являются метод Эйлера и методы Рунге – Кутта.
2. Методы прогноза и коррекции (многошаговые), в которых для отыскания следующей точки кривой y = f(x) требуется информация более чем об одной из предыдущих точек. Чтобы получить достаточно точное численное значение, часто прибегают к итерации. К числу таких методов относятся методы Милна, Адамса – Башфорта и Хемминга.
3. Явные методы, в которых функция Ф в выражении (4.1) не зависит от yn+1.
4. Неявные методы, в которых функция Ф зависит от yn+1.

3.1.Метод Эйлера

Иногда этот метод называют методом Рунге-Кутта первого порядка точности.

Данный метод одношаговый. Табулирование функции происходит поочередно в каждой точке. Для расчета значения функции в очередном узле необходимо использовать значение функции в одном предыдущем узле.

Пусть дано дифференциальное уравнение первого порядка



с начальным условием

y(x0) = y0.

Выберем шаг h и введём обозначения:

xi = x0 + i.h и yi = y(xi) , где i = 0, 1, 2, …,

xi – узлы сетки,

yi- значение интегральной функции в узлах .

Иллюстрации к решению приведены на рисунке 2.

Проведем прямую АВ через точку (xi,yi) под углом α. При этом

tgα = f(xi,yi) (1).

В соответствии с геометрическим смыслом задачи, прямая АВ является касательной к интегральной функции. Произведем замену точки интегральной функции точкой, лежащей на касательной AB.

Тогда yi+1 = yi+Δy (2).

Из прямоугольного треугольника АВС  (3).

Приравняем правые части (1) и (3). Получим .

Отсюда 

Подставим в это выражение формулу (2), а затем преобразуем его. В результате получаем формулу расчета очередной точки интегральной функции:

 (4).

Метод Эйлера – один из простейших методов численного решения обыкновенных дифференциальных уравнений. Но существенным его недостатком является большая погрешность вычислений. На рисунке 3.1.1 погрешность вычислений для i-го шага обозначена ε. С каждым шагом погрешность вычислений увеличивается.

y

y=y(x)

xi

xi+1

x

0

A

B

α

h

yi

yi+1

ε

Рисунок 3.1.1 – Метод Эйлера

F(x,y)- заданная функция-должна быть описана отдельно.

Входные параметры:

X0,XK- начальное и конечное значения независимой переменной;

Y0-значение y0 из начального условия y(x0)=y0;

N- количество отрезков разбиения;

Выходные параметры:

Y-массив значений искомого решения в узлах сетки

i = 0, … , N-1

Eiler(X0, XK, Y0, N, Y)

h = (XK – X0) / N

x = X0 + i \* h

Yi+1= Yi + h\*F(x,Yi)

end

Рисунок 3.1.2 – Блок-схема процедуры решения дифференциального уравнения методом Эйлера

3.2Метод Рунге – Кутта 4-го порядка

Пусть дано дифференциальное уравнение первого порядка



с начальным условием

*y(x0) = y0*.

Выберем шаг h и введём обозначения:

*xi = x0 + i.h и yi = y(xi) ,* где *i = 0, 1, 2, … .*

Аналогично описанным выше методам производится решение дифференциального уравнения. Отличие состоит в делении шага на 4 части.

Согласно методу Рунге – Кутта четвёртого порядка, последовательные значения *y****i*** искомой функции *y* определяются по формуле:



где

 , *i = 0, 1, 2, …*

а числа *k1(i), k2(i), k3(i), k4(i)* на каждом шаге вычисляются по формулам:



Это явный четырёхэтапный метод четвёртого порядка точности.

Методы Рунге – Кутта легко программируются и обладают значительной точностью и устойчивостью для широкого круга задач.

На рисунке 3.2.1 приведена блок-схема процедуры RUNGE(X0, XK, Y0, N, Y) для решения задачи Коши описанным выше методом Рунге – Кутта.

F(x, y) – заданная функция – должна быть описана отдельно.

Входные параметры:

X0, XK – начальное и конечное значения независимой переменной;

Y0 – значение y0 из начального условия y(x0) = y0;

N – количество отрезков разбиения;

Выходные параметры:

Y – массив значений искомого решения в узлах сетки;

RUNGE4(X0, XK, Y0, N, Y)

h = (XK – X0) / N

i = 0, … , N-1

*x* = X0 + i \* h

K1 = h \* F(x, Yi)

K2 = h \* F(x + h/2, Yi + K1 / 2)

K3 = h \* F(x + h/2, Yi + K2 / 2)

K4 = h \* F(x + h, Yi + K3)

K = (K1 + 2\*K2 + 2\*K3 + K4) / 6

Yi+1 = Yi + K

end

Рисунок 2.2.1 – Блок-схема процедуры RUNGE

Ниже приведена блок-схема алгоритма основной программы для решения задачи Коши и получения результатов с фиксированным количеством отрезков разбиения N. В основной программе происходит обращение к процедуреRUNGE(X0, XK, Y0, N, Y), вычисляющей значения искомой функции yj в точках xj методом Рунге – Кутта.

Исходными данными в данной задаче являются:

X0, XK – начальное и конечное значения независимой переменной;

Y0 – значение y0 из начального условия y(x0) = y0;

N – количество отрезков разбиения.

Результаты работы программы выводятся в виде двух столбцов:

X – массив значений узлов сетки;

Y – массив значений искомого решения в соответствующих узлах сетки

Ввод X0, XK, Y0, N

RUNGE(X0, XK, Y0, N, Y)

h = (XK – X0) / N

i = 0 … N

X = X0 + i \* h

Вывод X, Yi

End

Рисунок 3.2.2 – Блок-схема алгоритма основной программы для решения задачи Коши с фиксированным количеством отрезков разбиения N

# 4.Блок схема основных процедур

Подпрограмма общего решения

X0 ,Y0, Xk , h

N=X0 – Xk/ h

i=0,…,N-1

Xi=X0+i\*h

Эйлер

Рунге-Кутта

4 порядка

График

Эйлер

Yi+1

Yi+1=Yi+h\*f(xi , yi)

i=0,…,N-1

Рунге-Кутта 4 порядка.

i=0,…,N-1

X=X0+i\*h

K1=h\*f(x , y)

K2=h\*f(x+h/2 ;yi+K1/2)

K3=h\*f(x+h/2 ; yi+K2/2)

K4=h\*f(x+h ; yi+K3)

K=(K1+2K2+2K3+K4)/6

Yi+1=yi+K

# 5.Форма программы

Исходная форма:

TextBox1

Form1

TextBox2

Lable1

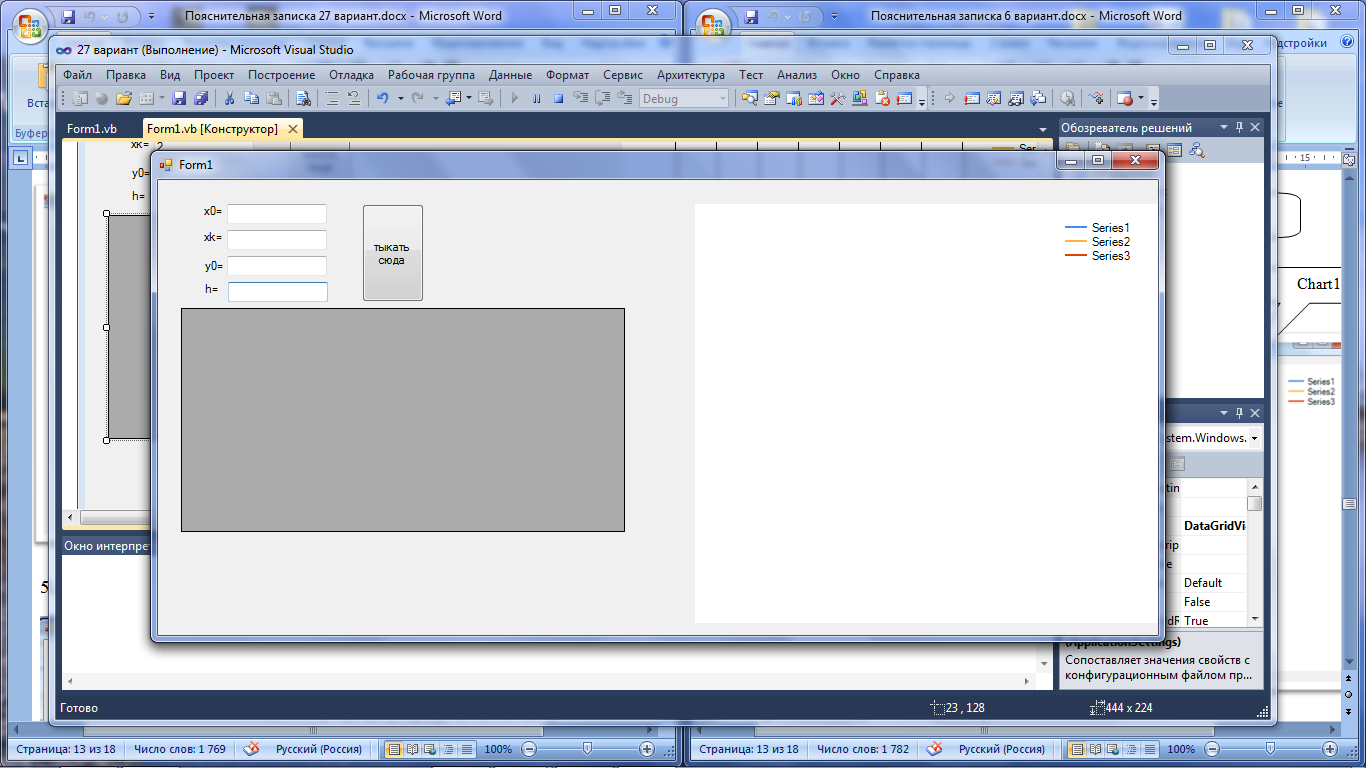
Lable2

TextBox3

Lable4

Chart1

TextBox4



DataGridView1

Button 1

Lable3

Программа реализована на языке Visual Basic studio 2010.

Для организации удобного интерфейса создана Form1. На форме используются следующие элементы:

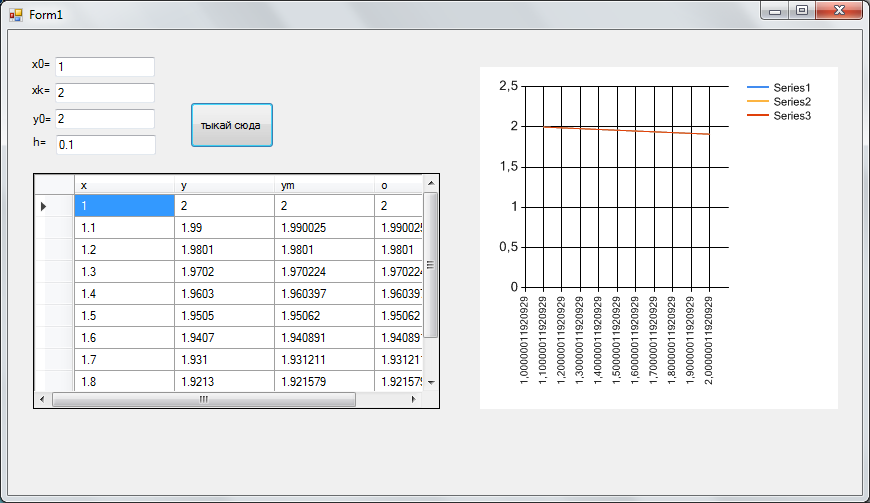
Метка (Label) используется для вывода текста, который не должен изменяться пользователем (например, заголовок какого-либо объекта управления).

Текстовое окно (Text Box) используется для ввода, вывода и редактирования пользователем текстовой (символьной) строки информации.

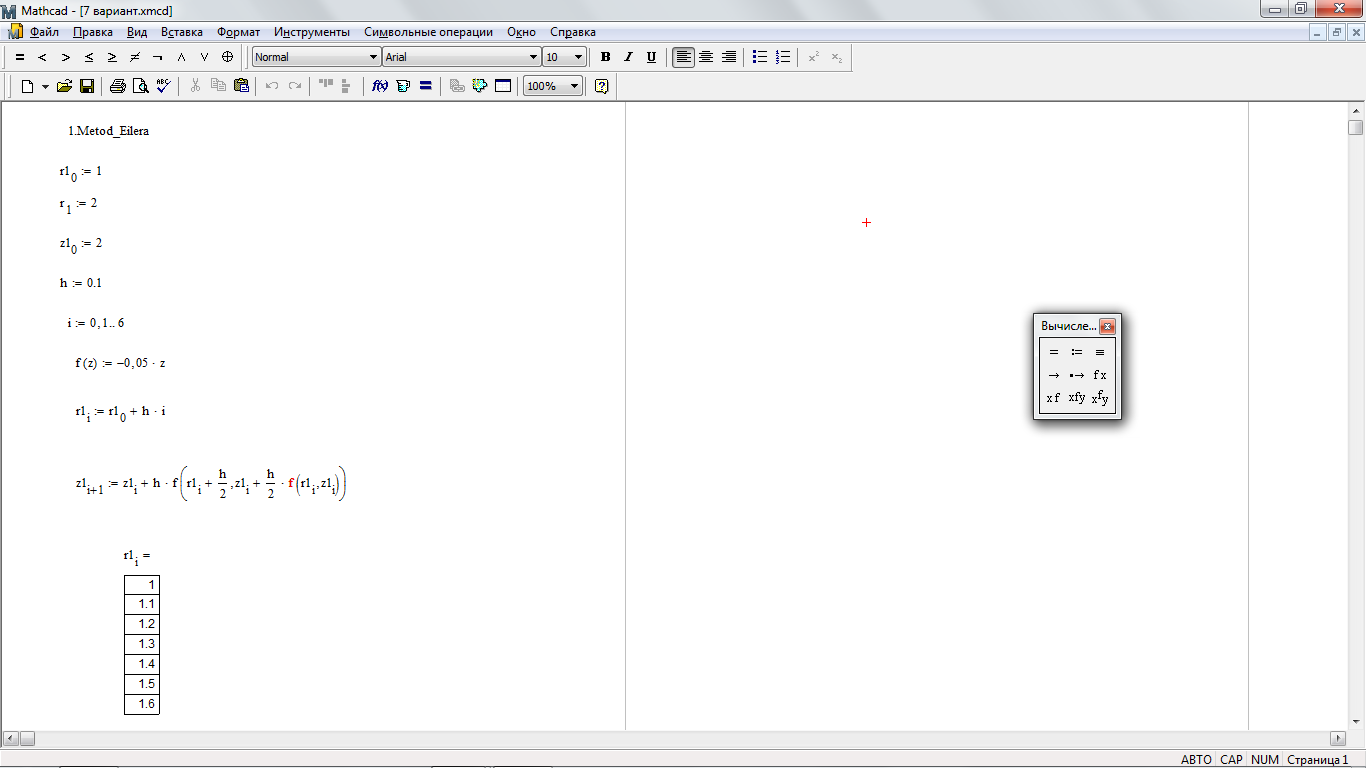
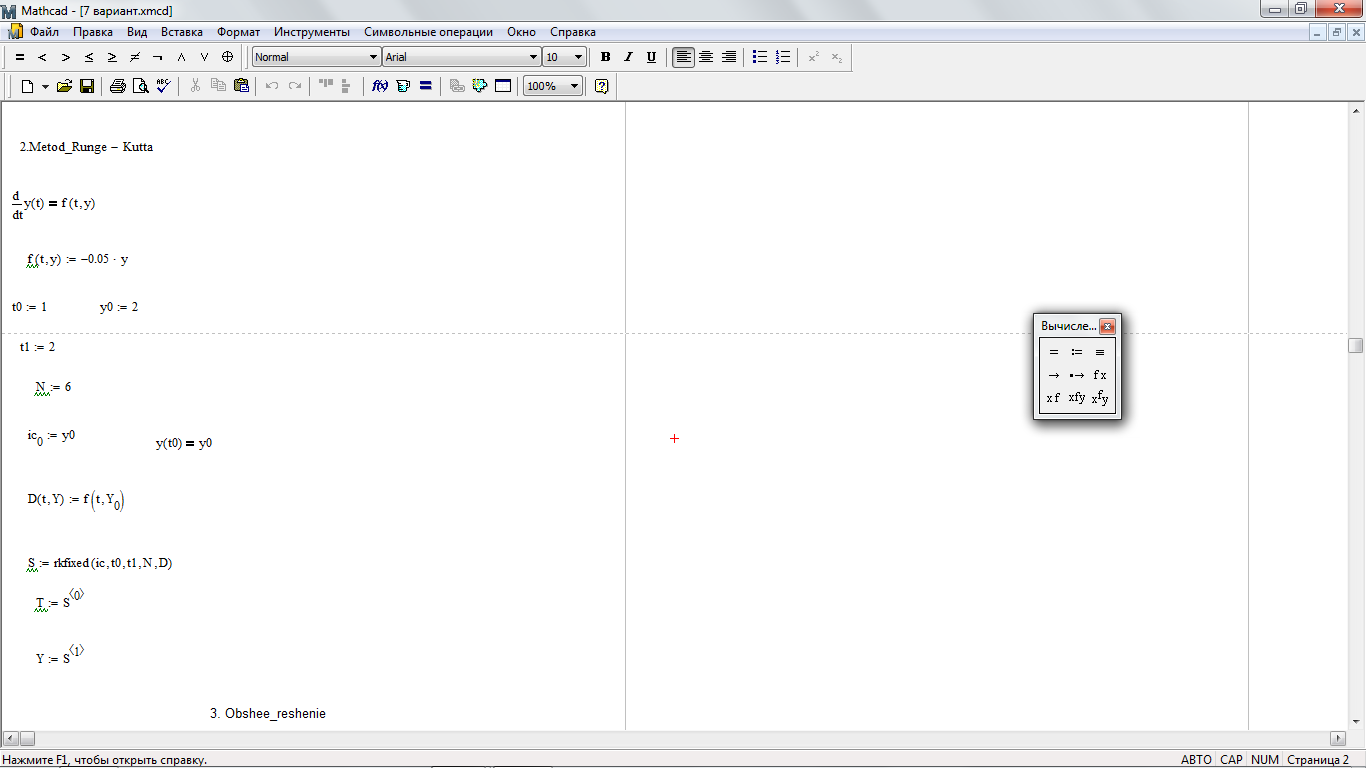
Button (кнопка) - обычно активизирует какую-то операцию\ Изображение или Картинка (Picture Box) используется для показа на форме графических объектов (статических или динамических), которые получаются с помощью графических методов.

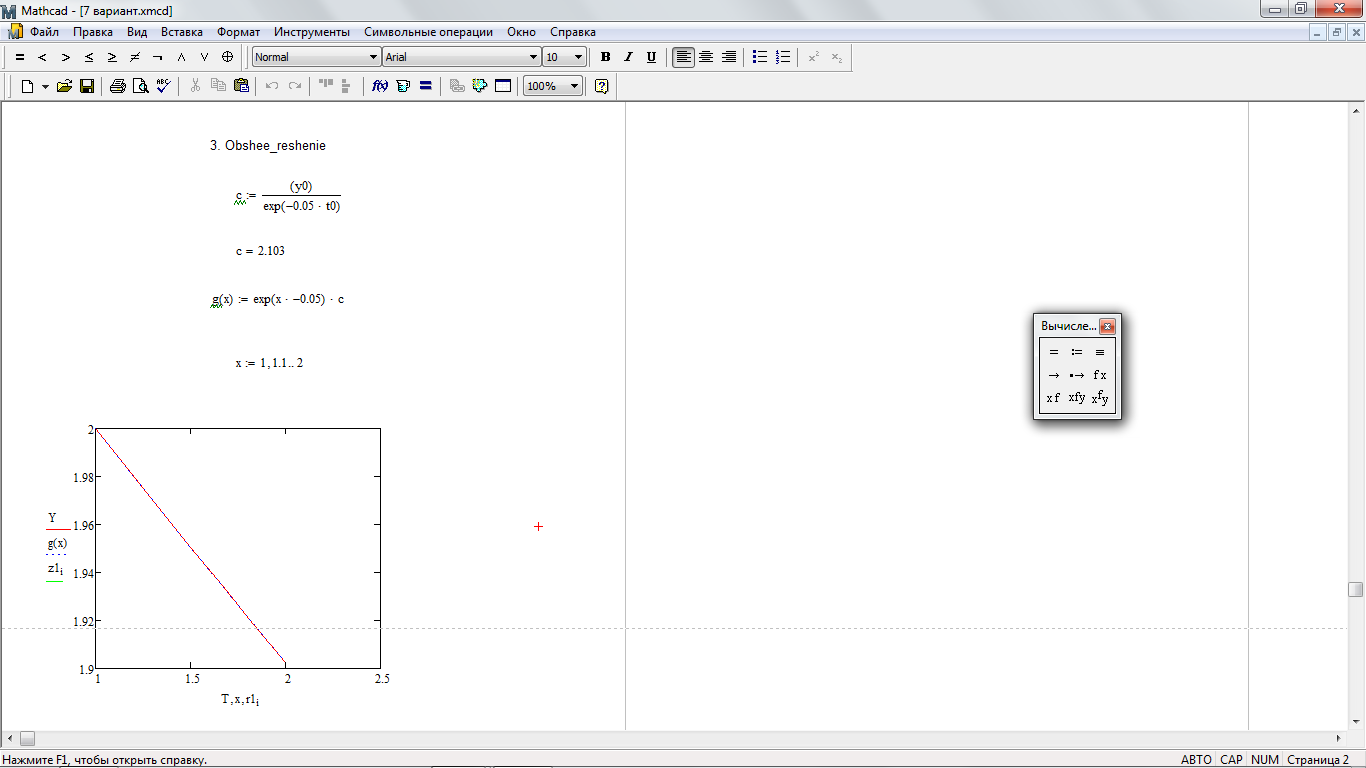
DataGridView1 – вывод таблицы

Конечная форма:



# 6.Решение в MathCad.





# Библиография.

1. Информатика. Базовый курс : учебное пособие для вузов / под ред. С. В. Симоновича .- 3-е изд.- СПб. : Питер, 2012. – (Стандарт третьего поколения)
2. Информатика. Базовый курс: учебник для вузов : для бакалавров и специалистов/ под ред. С. В. Симоновича. – 3-е изд. – СПб.:Питер, 2011. - (Стандарт третьего поколения).
3. [Макарова Н., Волков В. Информатика: Учебник для вузов. Стандарт третьего поколения. — СПб. : Питер, 2011 г. — 576 с. — Электронное издание. — Гриф УМО Учебник.](http://ibooks.ru/reading.php?productid=23133)
4. [Шапорев С. Информатика. Теоретический курс и практические занятия. — СПб. : БХВ-Петербург, 2010 г. — 480 с. — Электронное издание. — Гриф НМС по математике.](http://ibooks.ru/reading.php?productid=18483)

# Заключение.

В итоге , я изучил основы системы программирования Microsoft Visual Basic и приобрел начальные навыки в разработке программного обеспечения для операционных систем Windows.

Решив дифференциальное уравнение двумя способами, я пришел к выводу, что лучше производить вычисления методом Рунге-Кутта четвертого порядка, так как он во много раз точнее метода Эйлера модифицированного, его график сливается с общим решением (что мы видим на конечной форме).

Дифференциальные уравнения могут быть настолько сложными, что их невозможно решить вручную. Поэтому вычисление на компьютере является оптимальным решением.

# Приложение А

Листинг программы на языке VisualBasic:

Public Class Form1

Private x0, y0, xk, h As Single

Private n, m As Integer

Private k1, k2, k3, k4, k As Single

Dim x() As Single

Dim y1() As Single

Dim y2() As Single

Dim o() As Single

Function F(ByVal z As Single, ByVal t As Single) As Single

F = (Math.E ^ 2 + t \* z) / z

End Function

Function c(ByVal z As Single, ByVal t As Single) As Single

c = (t - Math.Exp(z) \* Math.Log(z)) / Math.Exp(z)

End Function

Private Sub Button1\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

x0 = Val(TextBox1.Text)

y0 = Val(TextBox2.Text)

xk = Val(TextBox3.Text)

h = Val(TextBox4.Text)

n = (xk - x0) / h

ReDim x(n)

ReDim y1(n)

ReDim y2(n)

ReDim o(n)

y1(0) = y0

y2(0) = y0

DataGridView1.ColumnCount = 4

DataGridView1.RowCount = n + 1

DataGridView1.Columns(0).Name = "x"

DataGridView1.Columns(1).Name = "y"

DataGridView1.Columns(2).Name = "ym"

DataGridView1.Columns(3).Name = "o"

For m As Integer = 0 To n

x(m) = x0 + (m) \* h

o(m) = Math.Exp(x(m)) \* (Math.Log(x(m)) + c(x0, y0))

DataGridView1.Item(0, m).Value = Str(x(m))

DataGridView1.Item(3, m).Value = Str(o(m))

Next m

For m As Integer = 0 To n - 1

x(m) = x0 + m \* h

y1(m + 1) = Math.Round(y1(m) + h \* F(x(m), y1(m)), 4)

k1 = h \* F(x(m), y2(m))

k2 = h \* F(x(m) + (h / 2), y2(m) + (k1 / 2))

k3 = h \* F(x(m) + (h / 2), y2(m) + (k2 / 2))

k4 = h \* F(x(m) + h, y2(m) + k3)

k = (k1 + 2 \* k2 + 2 \* k3 + k4) / 6

y2(m + 1) = y2(m) + k

DataGridView1.Item(1, m).Value = Str(y1(m))

DataGridView1.Item(2, m).Value = Str(y2(m))

Next m

For m As Integer = 0 To n - 1

Chart1.Series("Series1").Points.AddXY(x(m + 1), o(m))

Chart1.Series("Series2").Points.AddXY(x(m + 1), y1(m))

Chart1.Series("Series3").Points.AddXY(x(m + 1), y2(m))

Next m

End Sub

Private Sub Chart1\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Chart1.Click

End Sub

End Class