Министерство образования и науки Кыргызской Республики

**Кыргызский Государственный Технический Университет**

**им. И. Раззакова**

**Факультет информационных технологий**

**ПРОГРАММНАЯ ИНЖЕНЕРИЯ**

**Бакалаврский курс**

**Основы разработки и анализа требований к ПО**

**Отчет**

**по лабораторной работе №1**

**«Bisection Method»**

Выполнил: Байгазиев Санжар

Группа ПИ-2-15

4 курс

Преподаватель: кандидат технических наук, профессор

Тен Иосиф Григорьевич

Бишкек 2018

Содержание

[Глава 1: Описание проблемы 3](#_Toc525208502)

[Глава 2: Описание Bisection Method 3](#_Toc525208503)

[Глава 3: Спецификация требований к программному обеспечению 4](#_Toc525208504)

[3.1. Наименование программной разработки 4](#_Toc525208505)

[3.2. Описание бизнес-процесса и анализ расхождения 4](#_Toc525208506)

[3.3. Бизнес-цель разработки программного обеспечения 4](#_Toc525208507)

[3.4. Пользовательская история разработки программного обеспечения 4](#_Toc525208508)

[3.5. Функциональные требования к разработке программного обеспечения 5](#_Toc525208509)

[3.6. Нефункциональные требования на разработку программного обеспечения 6](#_Toc525208510)

[3.7. Ограничения на разработку программного обеспечения 7](#_Toc525208511)

[Глава 4: Стадии проектирования системы для поиска корня нелинейного уравнения f(x)=0, реализующей Bisection Method: 8](#_Toc525208512)

[Глава 5: Дизайн интерфейсной формы системы, реализующей Bisection method: 12](#_Toc525208513)

[Глава 6: Стадии конструирования системы для поиска корня нелинейного уравнения f(x)=0, реализующей Bisection Method: 13](#_Toc525208514)

[Stage №.1: Код программы на C#, ассоцированный с интерфейсной формой “-Form1.cs” 13](#_Toc525208515)

[Stage No.2: Код программы на C#, ассоцированный с классом “BisectionMethod.cs” , который реализует логику Bisection method по поиску корня нелинейного уравнения и составляет отдельный модуль в виде Public Class “BisectionMethod”: 18](#_Toc525208516)

[Stage No.3: Код программы на C#, ассоцированный с ***программным модулем*** “BisectionMethod.cs” , который реализует логику Bisection method по поиску корня нелинейного уравнения и показывает код по управлению Progress Bar: 20](#_Toc525208517)

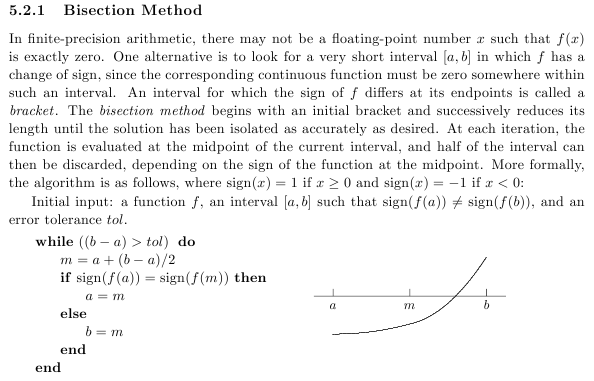
[Stage No.4: Подключение библиотечной программы “ Extreme.Numerics.Net40.dll” к программному проекту для выполнения функции парсинга 22](#_Toc525208518)

[Результат работы программы, реализующей Bisection method: 24](#_Toc525208519)

# Глава 1: Описание проблемы

Найти корень произвольного нелинейного уравнения – ноль нелинейной функции f(x) с заданной допустимой погрешностью Tolerance не более величины 1E–28 методом деления отрезка пополам (Bisection Method). Нелинейная функция f(x) имеет произвольный аналитический вид, составленный из математических функций (полиномов различных степеней, тригонометрических – sin(x), cos(x), exp(x), ln(x), log(x) и. т. д.), которая имеет математический смысл, и для которой существует хотя бы одно решение задачи. Описание метода деления отрезка пополам – Bisection Method – приведено по книге “SCIENTIFIC COMPUTING. An Introductory Survey. Michael T. Heath. University of Illinois at Urbana-Champaign. 1997 by The McGraw-Hill Companies. ISBN 0-07-027684-6”, стр. 154.

# Глава 2: Описание Bisection Method



[GoTo Contents](#Contents)

# Глава 3: Спецификация требований к программному обеспечению

## 3.1. Наименование программной разработки

Система поиска корней нелинейного уравнения методом деления отрезка пополам

## 3.2. Описание бизнес-процесса и анализ расхождения

При решении различных бизнес-задач отделу исследований (users, clients, customers) требуется находить корень произвольного нелинейного уравнения с высокой точностью. Для решения таких задач в этом отделе имеется 5 ставок специалистов. Два из них занимаются формализацией задачи и определением/нахождением аналитического вида нелинейной функции, описывающей, как можно точнее, модель поведения реальной системы, а остальные занимаются решением задачи – поиском корня нелинейного уравнения для найденной нелинейной функции. При больших трудозатратах (заняты три специалиста высокой квалификации), не всегда обеспечивается приемлемая и единообразная/унифицированная точность решения задачи. Бизнесу желательно было бы гарантированно обеспечить любую допустимую погрешность решения задачи при любых видах нелинейной функции f(x) с минимальными трудозатратами.

Таким образом в отделе для достижения цели проводится следующие действия

1. формализуется задачи
2. определяется аналитический вид нелинейной функции
3. находят корня нелинейной функции

Для решения таких задач в отделе заняты три специалиста высокой квалификации и не всегда обеспечивается приемлемая точность решения задачи. После автоматизации, деятельность отдела существенно изменится, так как нахождением корня не линейного уравнения займется программа. Программа обеспечит любую допустимую погрешность до 1e-28 для решения задачи при любых видах нелинейной функции за короткое время.

## 3.3. Бизнес-цель разработки программного обеспечения

Минимизировать затраты времени на поиск корня нелинейного уравнения ***до одной минуты*** при максимально высокой точности решения – допустимая погрешность решения должна быть ***не менее 1e-28 (т.е., 0.000000000000000000000000001)***, что позволит сократить ***две ставки*** специалиста в отделе исследований, поскольку теперь с такой задачей может справиться уже один специалист с помощью разрабатываемой системы.

## 3.4. Пользовательская история разработки программного обеспечения

***Пользователь:***

* должен вводить аналитическое выражение для требуемой нелинейной функции ***f(x)***;
* значения концов интеравала ***[a, b]***;
* значение допустимой погрешности ***Tolerance*** решения задачи;
* допустимое максимальное количество ***k\_max*** итераций
* допустимое максимальное количество ***max\_time*** времени

***Программа:***

* должна провести parsing – синтаксический анализ выражения введенной нелинейной функции и преобразовать аналитическую запись нелинейной функции в виде символов в вычисляемую процедуру в виде подпрограммы-функции;
* должна ввести символьные значения концов интервала ***[a, b]*** и преобразовать их в числовые значения концов интеравала;
* должна найти решение задачи – корень нелинейного уравнения ***f(x)=0*** или ноль нелинейной функции ***f(x)*** – по алгоритму Bisection Method;
* должна выводить найденное значение ***X\**** корня заданного нелинейного уравнения;
* должна выводить значение заданной нелинейной функции ***f(x\*)*** в этой точке;
* должна выводить общее затраченное время (в миллисекундах) на поиск решения ***Elapsed Time*** для оценки производительности программы;
* должна выводить затраченное на поиск решения с заданной погрешностью количество итераций ***Amount Of Iterations***;
* должна выводить достигнутое значение абсолютной ошибки решения ***Abs(b–a)***.

## 3.5. Функциональные требования к разработке программного обеспечения

* Система должна использовать ***Bisection Method для поиска корней*** нелинейного уравнения;
  + Система должна искать решение задачи по следующему алгоритму (Bisection algorithm):
  + Система должна обеспечивать ввод любых начальных числовых значений концов интервала ***[a; b]***;
  + Система должна распознавать ситуации , которые недопустимы для корректного применения метода – Bisection Method – поиска корней нелинейного уравнения, и сообщать пользователю о возникновении подобной ситуации;
  + Система должна обеспечить возможность продолжения поиска решения задачи с последних достигнутых на шаге ***k\_max*** концов интервала ***[a(k\_max), b(k\_max)]***, приняв их за начальные значения концов интервала {т.е., приняв a(0)=a(k\_max) и b(0)=b(k\_max) система должна заново выполнить не менее ***k\_max*** итераций}, если за данное количество итераций ***k\_max*** не достигнуто решение с требуемой погрешностью ***Tolerance***;
* Система должна обеспечивать возможность ввода исходных данных имеющих следующие особенности:
  + Система должна обеспечивать возможность ввода ***аналитического выражения*** для любой нелинейной функции;
  + Система должна проводить ***parsing***\*\* – синтаксический анализ аналитического выражения нелинейной функции f(x) – и автоматически конструировать подпрограмму для вычисления значения этой функции для любого значения аргумента ***x*** этой функции;
  + Система должна обеспечить ввод любого целого числового значения в качестве максимального допустимого количества итераций ***k\_max***;
  + Система должна обеспечевать возможность ввода любых числовых значений допустимой погрешности ***Tolerance*** как в формате с фиксированной точностью “0.0000000000000000000000000001”, так и в экпоненциальном формате “1e-28”;
  + Система должна быть способна проверять валидность введенного аналитического выражения в качестве функции f(x) и сообщать пользователю об ошибке в аналитическом выражении функции;
  + Система должна быть способна проверять валидность формата введенных исходных числовых данных и сообщать пользователю об ошибке в формате исходных данных;
* Система должна обеспечить вывод следующих результатов решения задачи:
  + Система должна обеспечить вывод результата решения задачи в виде числа x\* – корня нелинейного уравнения или нуля нелинейной функции;
  + Система должна обеспечить вывод значения функции в этой точке f(x\*);
  + Система должна обеспечить вывод количества итераций, которые потребовались системе, чтобы найти решение задачи с требуемой погрешностью в качестве критерия производительности системы;
  + Система должна обеспечить вывод достигнутого значения погрешности решения задачи в качестве оценки точности этого метода (Bisection method);
  + Система должна обеспечить невозможность какого-либо ***исправления вручную*** выходных данных системы;
* Система должна обеспечить ***оценку производительности программы*** при решении задачи заданным методом (***Bisection Method***) и реализации программы на языке ***C# в среде Visual Studio 2017*** путем измерения и вывода времени, затрачиваемой на поиск решения с заданной погрешностью;

## 3.6. Нефункциональные требования на разработку программного обеспечения

* Программа в процессе поиска решения должна отображать индикатор состояния процесса выполнения операции ***Progress Bar***, чтобы пользователь понимал что система не зависла, а находится в рабочем состоянии;
* В случае ошибки во введенном аналитическом выражении нелинейной функции f(x) система должна предоставлять достаточно подробную информацию об ошибке, которая будет облегчать поиск источника ошибки;
* Система должна обеспечить возможность очищения текстовых полей ***выходного*** интерфейса от результатов предыдущих вычислений при новом запуске вычислений. При этом данные в текстовых полях ***входного*** интерфейса должны сохраняться. Это позволит пользователю увидеть более наглядно, что получены новые результаты, соответствующие измененным входным данным;
* Сбой системы не должен происходить чаще чем один раз на тысячу запусков программы;
* Система должна обеспечить оценку качества решения задачи заданным методом (***Bisection Method***) путем измерения и вывода количества итераций, потраченных на поиск решения с заданной погрешностью;

## 3.7. Ограничения на разработку программного обеспечения

* Нелинейная функция ***f(x)*** должна быть непрерывной на интервале ***[a; b]***;
* Разработать ПО на основе IDE (integrated development environment – интегрированная среда разработки) – Visual Studio 2017;
* Разработать ПО на языке C# ;
* Система должна решать задачи с допустимой погрешностью до 1e-28;
* Система должна решать задачи за время не более одной минуты;
* Временные затраты на разработку ПО не должны превышать одного месяца;
* Финансовые затраты на разработку ПО и соответсвующую документацию (техническое задание на ПО, руководство пользователя и руководство программиста) не должны превышать $1000;
* Система должна быть разработана для эксплуатации на компъютерах типа пентиум с оперативной памятью не более 2 гигобайт;
* Разработка программы должна начаться не позже 10 сентября 2018 года и система должна быть передана заказчику не позднее 1 октября 2018 года;
* Система должна позволять пользователю решать не менее 100 задач поиска корня нелинейного уравнения за один час для любой нелинейной функции, описываемой в виде аналитического выражения, корректной с точки зрения математики;
* Система должна понимать и обрабатывать любые аналитические выражения для нелинейной функции, которые включают следующие математические символы:

Cтандартных функций

"sqrt", "sin", "cos", "tan",

"atan", "acos", "asin", "acotan",

"exp", "ln", "log",

"sinh", "cosh", "tanh", "abs",

"ceil", "floor", "fac", "sfac", "round", "fpart"

* + Основных вычислительных операций: сложение (+), вычитание (-), умножение (\*), деление (/), остаток от деления (%) и возведение в степень (^);
  + круглых и квадратных скобок любой вложенности;
  + вектора переменных, на которые можно ссылаться, используя запись вида x[N] или x(N), где N - индекс переменной.

# Глава 4: Стадии проектирования системы для поиска корня нелинейного уравнения f(x)=0, реализующей Bisection Method:

Stage No.1: Блок-схема Bisection method

Figure 1: Flowchart of the Bisection Method with a design of the input and output interfaces

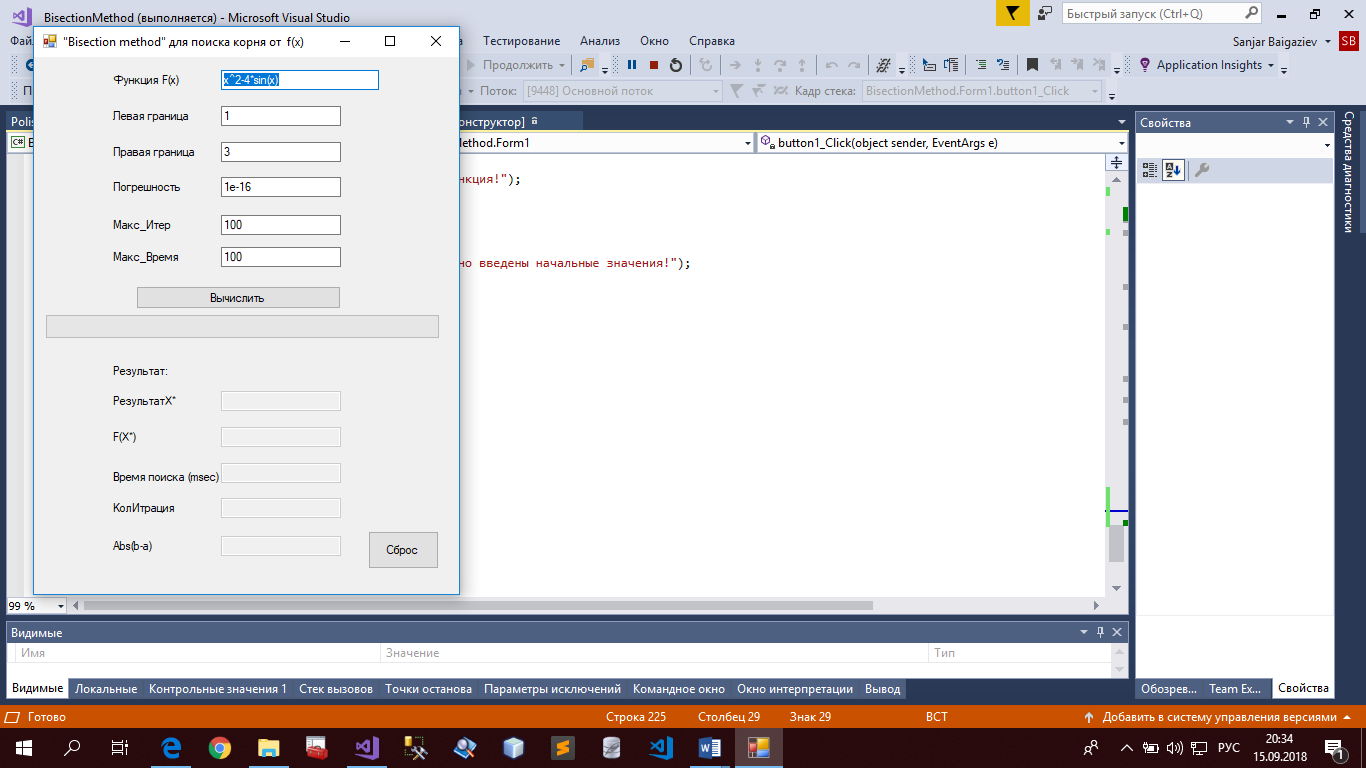
[GoTo Contents](#Contents)

Таблица 1: Настройки для свойств элементов управления программной системой, реализующей метод Bisection

|  |  |  |  |
| --- | --- | --- | --- |
| **Number of control** | **Control** | **Property** | **Setting** |
| 1 | Label1 | Appearance (Text) | Функция F(x) |
| Label1 | Design (Name) | Label1 |
| 2 | Textbox1 | Appearance (Text) | x^2-4\*sin(x) |
| Textbox1 | Design (Name) | Textbox1 |
| 3 | Label2 | Appearance (Text) | Левая граница |
| Label2 | Design (Name) | Label2 |
| 4 | Textbox2 | Appearance (Text) | 1 |
| Textbox2 | Design (Name) | textbox2 |
| 5 | Label3 | Appearance (Text) | Правая граница: |
| Label3 | Design (Name) | Label3 |
| 6 | Textbox3 | Appearance (Text) | 3 |
| Textbox3 | Design (Name) | textbox3 |
| 7 | Label4 | Appearance (Text) | Погрешность: |
| Label4 | Design (Name) | Label4 |
| 8 | Textbox4 | Appearance (Text) | 100 |
| Textbox4 | Design (Name) | 1e-16 |
| 9 | Label5 | Appearance (Text) | Макс\_Итер |
| Label5 | Design (Name) | Label5 |
| 10 | Textbox5 | Appearance (Text) | 100 |
| Textbox5 | Design (Name) | textbox5 |
| 11 | Label11 | Design (Name) | label11 |
| Label11 | Appearance (Text) | Макс\_Время |
| 12 | Button1 | Appearance (Text) | Вычислить |
| Button1 | Design (Name) | button1 |
| 13 | ProgressBar1 | Behavior (Visible) | False |
| ProgressBar1 | Design (Name) | progressBar1 |
| 14 | Label13 | Appearance (Text) | Результат: |
| Label13 | Design (Name) | Label13 |
| 15 | Label10 | Appearance (Text) | РезультатX\* |
| Label10 | Design (Name) | Label10 |
| 16 | Textbox10 | Design (Name) | Textbox10 |
| Textbox10 | Behavior (enabled) | false |
| 17 | Label9 | Appearance (Text) | f(X\*): |
| Label9 | Design (Name) | Label9 |
| 18 | Textbox9 | Behavior (enabled) | false |
| Textbox9 | Design (Name) | textbox9 |
| 19 | Label8 | Appearance (Text) | Время поиска (msec): |
| Label8 | Design (Name) | label8 |
| 20 | Textbox8 | Behavior (enabled) | false |
| Textbox8 | Design (Name) | Textbox8 |
| 21 | Label7 | Appearance (Text) | КолИтрация: |
| Label7 | Design (Name) | label7 |
| 22 | Textbox7 | Behavior (enabled) | false |
| Textbox7 | Design (Name) | textbox7 |
| 23 | Button2 | Appearance (Text) | Сброс: |
| Button2 | Design (Name) | button2 |
| 24 | Label6 | Appearance (Text) | Abs(b-a) |
| Label6 | Design (Name) | Label6 |
| 25 | Textbox6 | Behavior (enabled) | false |
| Textbox6 | Design (Name) | Textbox6 |

[GoTo Contents](#Contents)

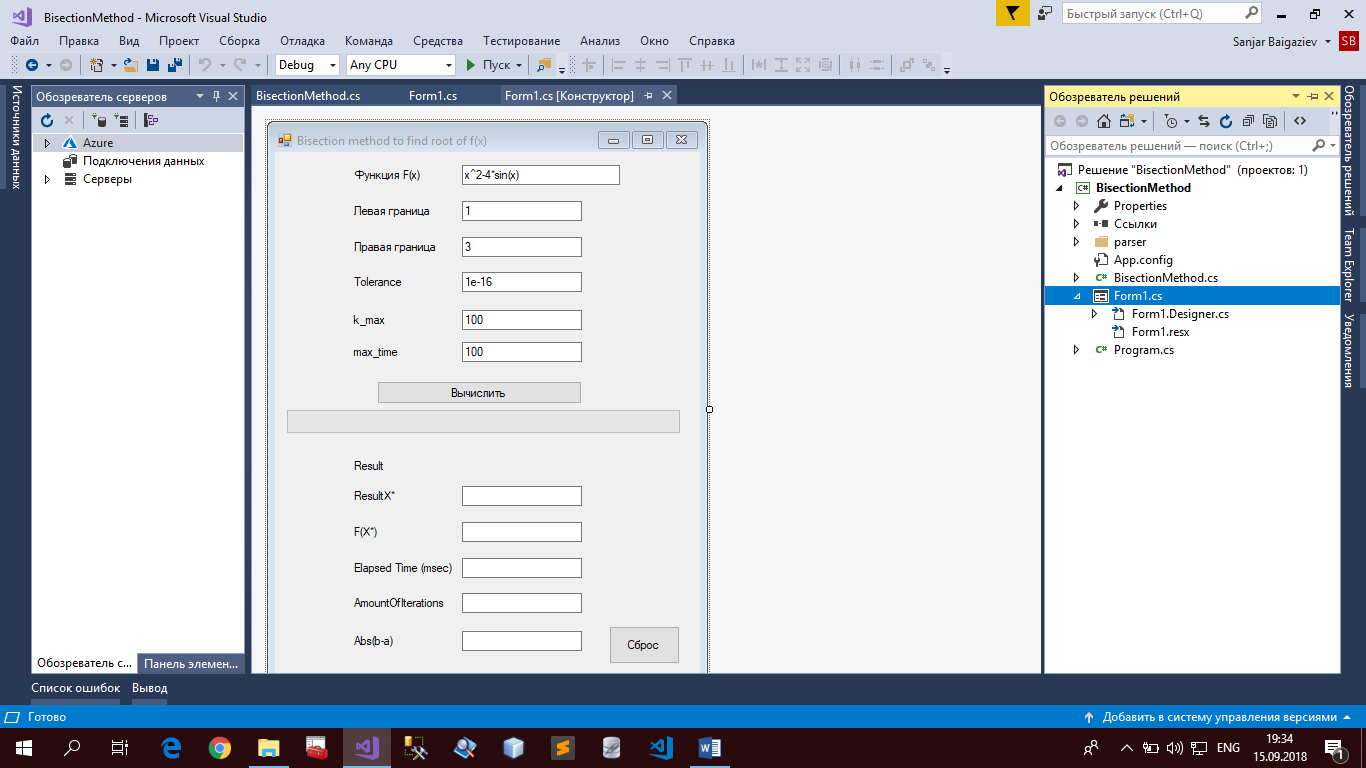
# Глава 5: Дизайн интерфейсной формы системы, реализующей Bisection method:



[GoTo Contents](#Contents)

# Глава 6: Стадии конструирования системы для поиска корня нелинейного уравнения f(x)=0, реализующей Bisection Method:

## Stage №.1: Код программы на C#, ассоцированный с интерфейсной формой “-Form1.cs”



using info.lundin.math;

using System;

using System.Drawing;

using System.Globalization;

using System.Windows.Forms;

namespace BisectionMethod

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

decimal a, b;

string f;

decimal x;

decimal tol;

int k\_max;

int time\_max;

private void button2\_Click(object sender, EventArgs e)

{

textBox6.Text = "";

textBox7.Text = "";

textBox8.Text = "";

textBox9.Text = "";

textBox10.Text = "";

label13.Text = "";

}

private void textBox2\_KeyPress(object sender, KeyPressEventArgs e)

{

if ((e.KeyChar >= '0') && (e.KeyChar <= '9'))

{

return;

}

if (e.KeyChar == '.')

{

e.KeyChar = ',';

}

if (e.KeyChar == ',')

{

if (textBox2.Text.IndexOf(',') != -1)

{

e.Handled = true;

}

return;

}

if (e.KeyChar == '-')

{

if (textBox2.Text.IndexOf('-') != -1)

{

e.Handled = true;

}

return;

}

if (Char.IsControl(e.KeyChar))

{

return;

}

e.Handled = true;

}

private void textBox3\_KeyPress(object sender, KeyPressEventArgs e)

{

if ((e.KeyChar >= '0') && (e.KeyChar <= '9'))

{

return;

}

if (e.KeyChar == '.')

{

e.KeyChar = ',';

}

if (e.KeyChar == ',')

{

if (textBox3.Text.IndexOf(',') != -1)

{

e.Handled = true;

}

return;

}

if (Char.IsControl(e.KeyChar))

{

return;

}

e.Handled = true;

}

private void textBox4\_KeyPress(object sender, KeyPressEventArgs e)

{

if ((e.KeyChar >= '0') && (e.KeyChar <= '9'))

{

return;

}

if (e.KeyChar == 'e')

{

return;

}

if (e.KeyChar == '-')

{

return;

}

if (e.KeyChar == '.')

{

e.KeyChar = ',';

}

if (e.KeyChar == ',')

{

if (textBox4.Text.IndexOf(',') != -1)

{

e.Handled = true;

}

return;

}

if (Char.IsControl(e.KeyChar))

{

return;

}

e.Handled = true;

}

private void textBox5\_KeyPress(object sender, KeyPressEventArgs e)

{

if ((e.KeyChar >= '0') && (e.KeyChar <= '9'))

{

return;

}

if (Char.IsControl(e.KeyChar))

{

return;

}

e.Handled = true;

}

private void textBox11\_KeyPress(object sender, KeyPressEventArgs e)

{

if ((e.KeyChar >= '0') && (e.KeyChar <= '9'))

{

return;

}

if (Char.IsControl(e.KeyChar))

{

return;

}

e.Handled = true;

}

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

{

label12.Text = "";

if (textBox1.Text == "" || textBox2.Text == "" || textBox3.Text == "" || textBox4.Text == "" || textBox5.Text == "" || textBox11.Text == "")

{

MessageBox.Show("Не все данные введены!");

}

else

{

try

{

textBox6.Text = "";

textBox7.Text = "";

textBox8.Text = "";

textBox9.Text = "";

textBox10.Text = "";

decimal tol1;

decimal x\_result;

decimal f\_result;

decimal l\_result;

decimal r\_result;

decimal iter\_result;

decimal time\_result;

decimal tol\_result;

progressBar1.Visible = true;

progressBar1.Maximum = Convert.ToInt32(textBox5.Text) + 1;

BisectionMethod method = new BisectionMethod();

x\_result = method.Bisection\_algo(textBox1, textBox2, textBox3, textBox4, textBox5, textBox11, progressBar1);

tol1 = Decimal.Parse(textBox4.Text, NumberStyles.AllowExponent | NumberStyles.AllowDecimalPoint);

f\_result = method.f\_result;

l\_result = method.l\_result;

r\_result = method.r\_result;

iter\_result = method.iter\_result;

time\_result = method.time\_result;

tol\_result = r\_result - l\_result;

textBox10.Text = x\_result.ToString();

textBox9.Text = f\_result.ToString();

textBox8.Text = time\_result.ToString();

textBox7.Text = iter\_result.ToString();

textBox6.Text = tol\_result.ToString("0e0");

//textBox12.Text = tol\_result.ToString();

if (tol\_result>tol1)

{

label12.ForeColor = Color.Red;

label12.Text = "Результат с заданной точностью не достигнут!";

}

if (iter\_result == 0)

{

label12.ForeColor = Color.Red;

label12.Text = "Ошибка";

}

}

catch (ParserException ex)

{

MessageBox.Show("Неверно введена функция!");

progressBar1.Visible = false;

}

catch (FormatException ex)

{

MessageBox.Show(ex.Message + "Неверно введены начальные значения!");

}

}

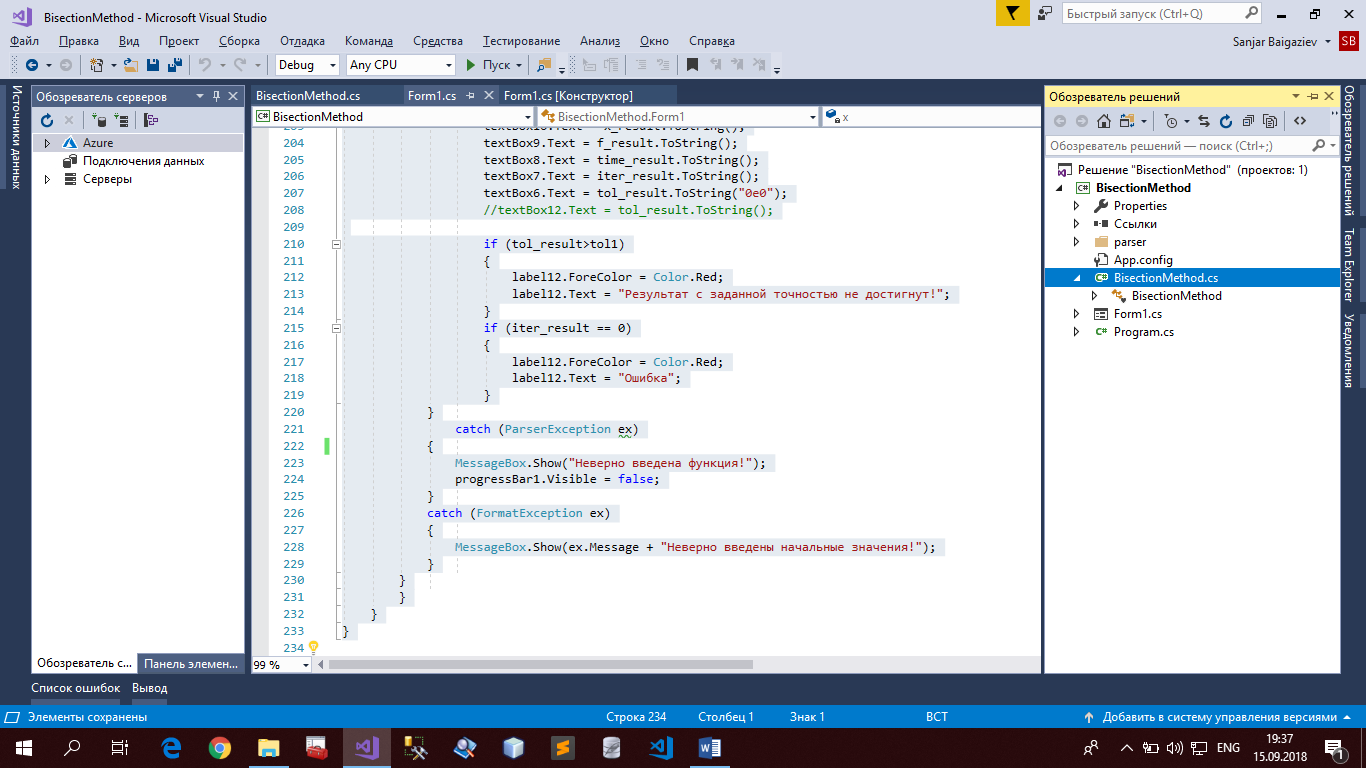
}

}

}

[GoTo Contents](#Contents)

## Stage No.2: Код программы на C#, ассоцированный с классом “BisectionMethod.cs” , который реализует логику Bisection method по поиску корня нелинейного уравнения и составляет отдельный модуль в виде Public Class “BisectionMethod”:



using parserDecimal.Parser;

using System;

using System.Windows.Forms;

using System.Globalization;

using System.Diagnostics;

namespace BisectionMethod

{

class BisectionMethod

{

Computer parser;

public decimal x\_result { get; set; }

public decimal f\_result { get; set; }

public decimal l\_result { get; set; }

public decimal r\_result { get; set; }

public decimal iter\_result { get; set; }

public int time\_result { get; set; }

public decimal tol\_result { get; set; }

Stopwatch stopwatch = new Stopwatch();

public decimal fx(string F,decimal v)

{

parser = new Computer();

return parser.Compute(F, v);

}

public decimal Bisection\_algo(TextBox funct, TextBox L\_text, TextBox R\_text, TextBox tol, TextBox k\_max, TextBox maxt\_time, ProgressBar pb)

{

int iter;

string F;

decimal l;

decimal r;

decimal Tol;

l = Convert.ToDecimal(L\_text.Text);

r = Convert.ToDecimal(R\_text.Text);

Tol = Decimal.Parse(tol.Text, NumberStyles.AllowExponent | NumberStyles.AllowDecimalPoint);

iter = Convert.ToInt32(k\_max.Text);

F = funct.Text;

decimal fa;

decimal fb;

decimal fm;

decimal m;

int max\_iter = Convert.ToInt16(k\_max.Text);

int max\_iter2 = max\_iter;

int max\_time = Convert.ToInt16(maxt\_time.Text);

int max\_time2 = max\_time;

stopwatch.Start();

pb.Value = 1;

fa = fx(F, l);

fb = fx(F, r);

if (Math.Sign(fa) == Math.Sign(fb))

{

MessageBox.Show("Выбранны неправильные границы для вычисления! проверьте границы 'a' и 'b'");

}

else

{

iter = 0;

while ((r - l) > Tol) //проверка на точность

{

if (iter >= max\_iter) //проверка на итерацию

{

stopwatch.Stop();

DialogResult dr = MessageBox.Show("Указанная точность за " + iter + " итераций не достигнута.Продолжить вычисление ? Будет добавлено " + max\_iter + " итераций.", "Продолжить вычисления ? ", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if(dr == DialogResult.No)

{

break;

}

else if (dr == DialogResult.Yes)

{

max\_iter = max\_iter + max\_iter2;

pb.Maximum = max\_iter + 1;

k\_max.Text = (max\_iter + max\_iter2).ToString();

stopwatch.Start();

}

}

if (max\_time<=Convert.ToInt16(stopwatch.ElapsedMilliseconds))

{

stopwatch.Stop();

DialogResult dr = MessageBox.Show("Время вышло. Продолжить вычисление? Будет добавлено " + max\_time2 + " миллисекунд", "Продолжить вычисления?", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if (dr == DialogResult.No)

{

break;

}

else if(dr == DialogResult.Yes)

{

max\_time += max\_time2;

maxt\_time.Text = max\_time.ToString();

stopwatch.Start();

}

}

m = l + (r - l) / 2;

fa = fx(F,l);

fm = fx(F, m);

if (Math.Sign(fa) == Math.Sign(fm))

{

l = m;

pb.Value++;

}

else

{

r = m;

pb.Value += 1;

}

iter++;

}

f\_result = fx(F,l);

l\_result = l;

r\_result = r;

pb.Value = pb.Maximum;

pb.Visible = false;

stopwatch.Stop();

time\_result = Convert.ToInt16(stopwatch.ElapsedMilliseconds);

iter\_result = iter;

}

return l;

}

}

}

[GoTo Contents](#Contents)

## Stage No.3: Код программы на C#, ассоцированный с ***программным модулем*** “BisectionMethod.cs” , который реализует логику Bisection method по поиску корня нелинейного уравнения и показывает код по управлению Progress Bar:

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Здесь в коде реализована логика Bisection Method and Progress Bar**

while ((r - l) > Tol) //проверка на точность

{

if (iter >= max\_iter) //проверка на итерацию

{

stopwatch.Stop();

DialogResult dr = MessageBox.Show("Указанная точность за " + iter + " итераций не достигнута.Продолжить вычисление ? Будет добавлено " + max\_iter + " итераций.", "Продолжить вычисления ? ", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if(dr == DialogResult.No)

{

break;

}

else if (dr == DialogResult.Yes)

{

max\_iter = max\_iter + max\_iter2;

pb.Maximum = max\_iter + 1;

k\_max.Text = (max\_iter + max\_iter2).ToString();

stopwatch.Start();

}

}

if (max\_time<=Convert.ToInt16(stopwatch.ElapsedMilliseconds))

{

stopwatch.Stop();

DialogResult dr = MessageBox.Show("Время вышло. Продолжить вычисление? Будет добавлено " + max\_time2 + " миллисекунд", "Продолжить вычисления?", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if (dr == DialogResult.No)

{

break;

}

else if(dr == DialogResult.Yes)

{

max\_time += max\_time2;

maxt\_time.Text = max\_time.ToString();

stopwatch.Start();

}

}

m = l + (r - l) / 2;

fa = fx(F,l);

fm = fx(F, m);

if (Math.Sign(fa) == Math.Sign(fm))

{

l = m;

pb.Value++;

}

else

{

r = m;

pb.Value += 1;

}

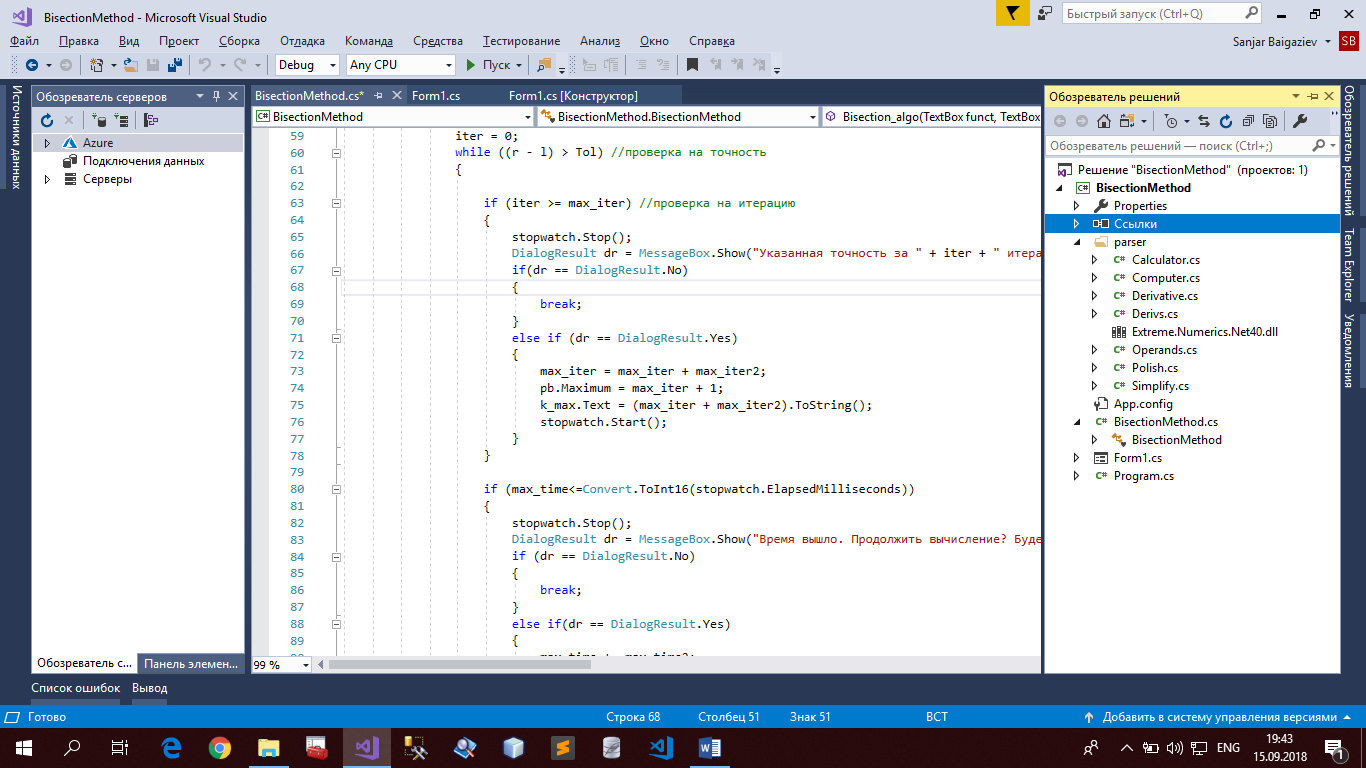
iter++;

}

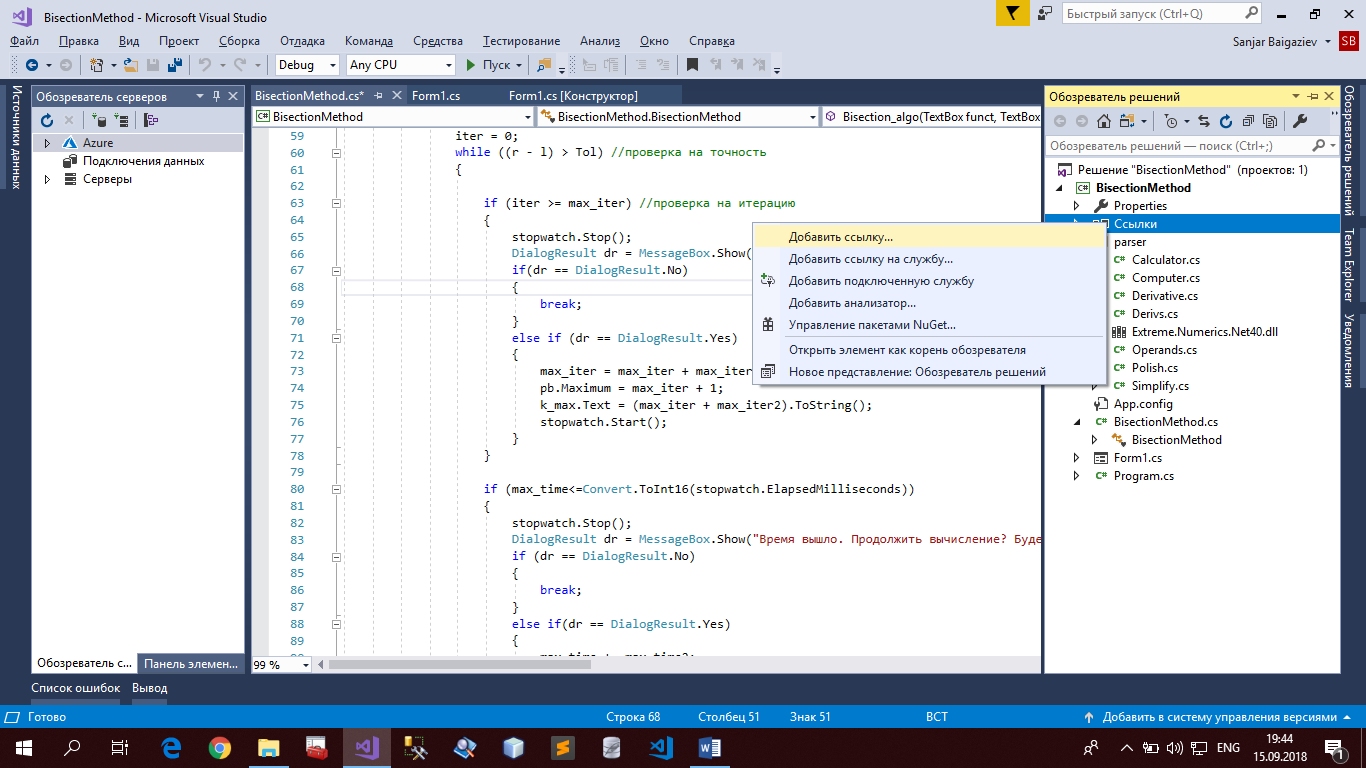
## Stage No.4: Подключение библиотечной программы “ Extreme.Numerics.Net40.dll” к программному проекту для выполнения функции парсинга

***Примечание***: *Если вставить приведенные в этом отчете коды программы, то Visual Studio выделит строки листинга программы, в которых есть ссылки на библиотечную функцию «Extreme.Numerics.Net40.dll», как ошибки. Это связано с тем, что в проект не включена ссылка на эту функцию. Ниже приведена инструкция по включению в проект программного обеспечения парсера “* Extreme.Numerics.Net40.dll*”.*

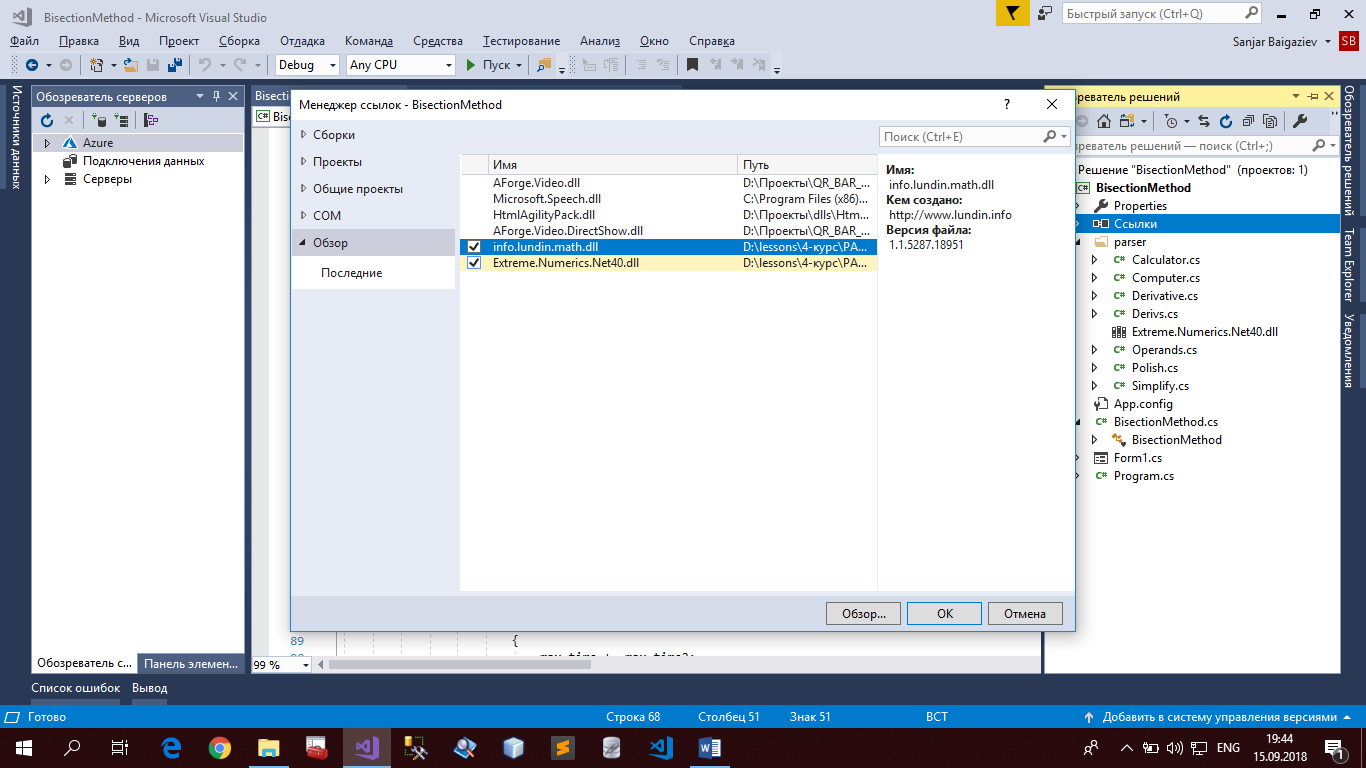
Шаг №4.1: Правой кнопкой мыши открыть контекстное меню на выделенной темным цветом строке:



Шаг №4.2: Щелкнуть мышкой на строчке «Добавить ссылку...»:



Шаг №4.3: В диалоговом окне менеджера ссылок ввести имя файла “ *Extreme.Numerics.Net40.dll* ”:



Шаг №4.4: В листинге программы исчезнут все пометки об ошибках в коде программы, связанных с тем, что оператор «Imports *Extreme.Numerics.Net40.dll*» был неопределен, если нижеприведенные строки кода программы были уже введены до введения ссылки на эту библиотечную функцию. Если же эти строки кода не были до сих пор введены, то теперь можно ввести эти коды, в которых используется функция парсинга, как это показано ниже:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*это примеры использования парсинга “ *Extreme.Numerics.Net40.dll* ”

Computer parser;

public decimal fx(string F,decimal v)

{

parser = new Computer();

return parser.Compute(F, v);

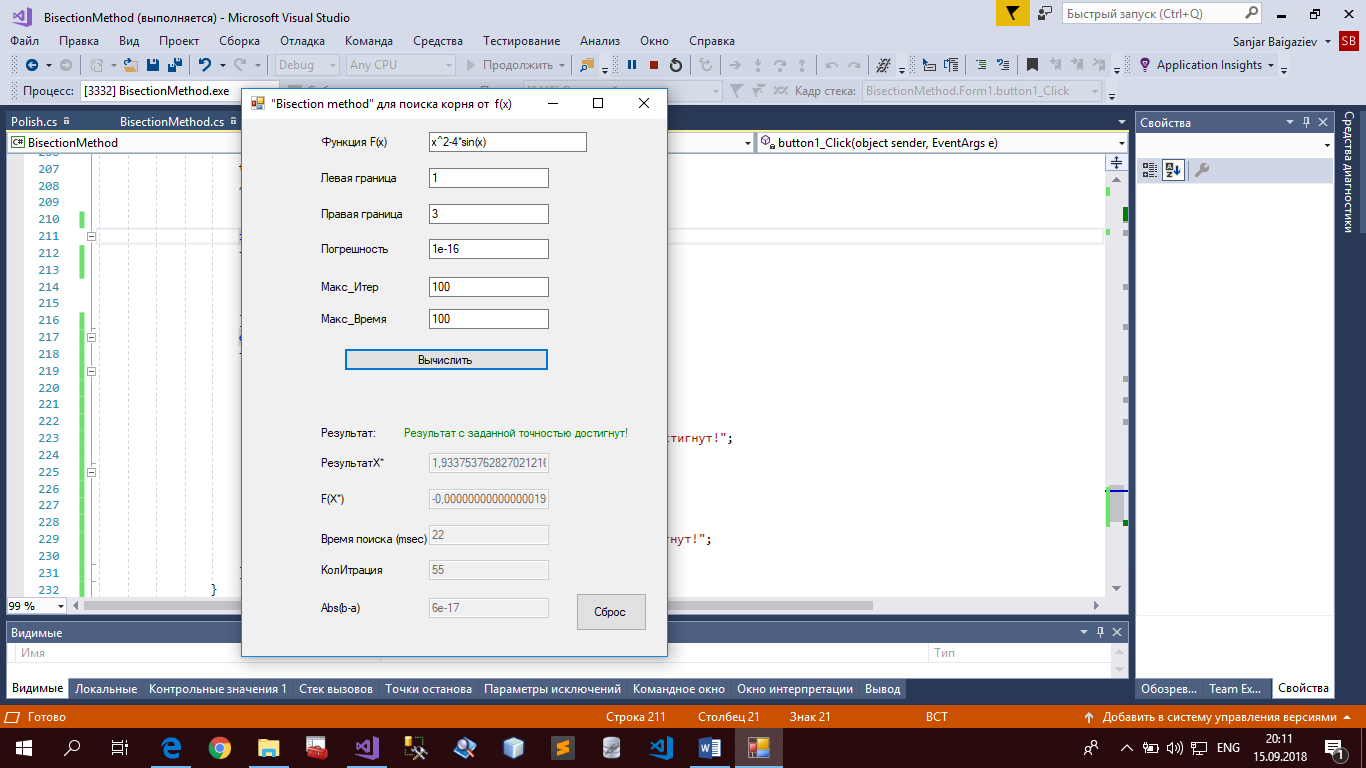
}

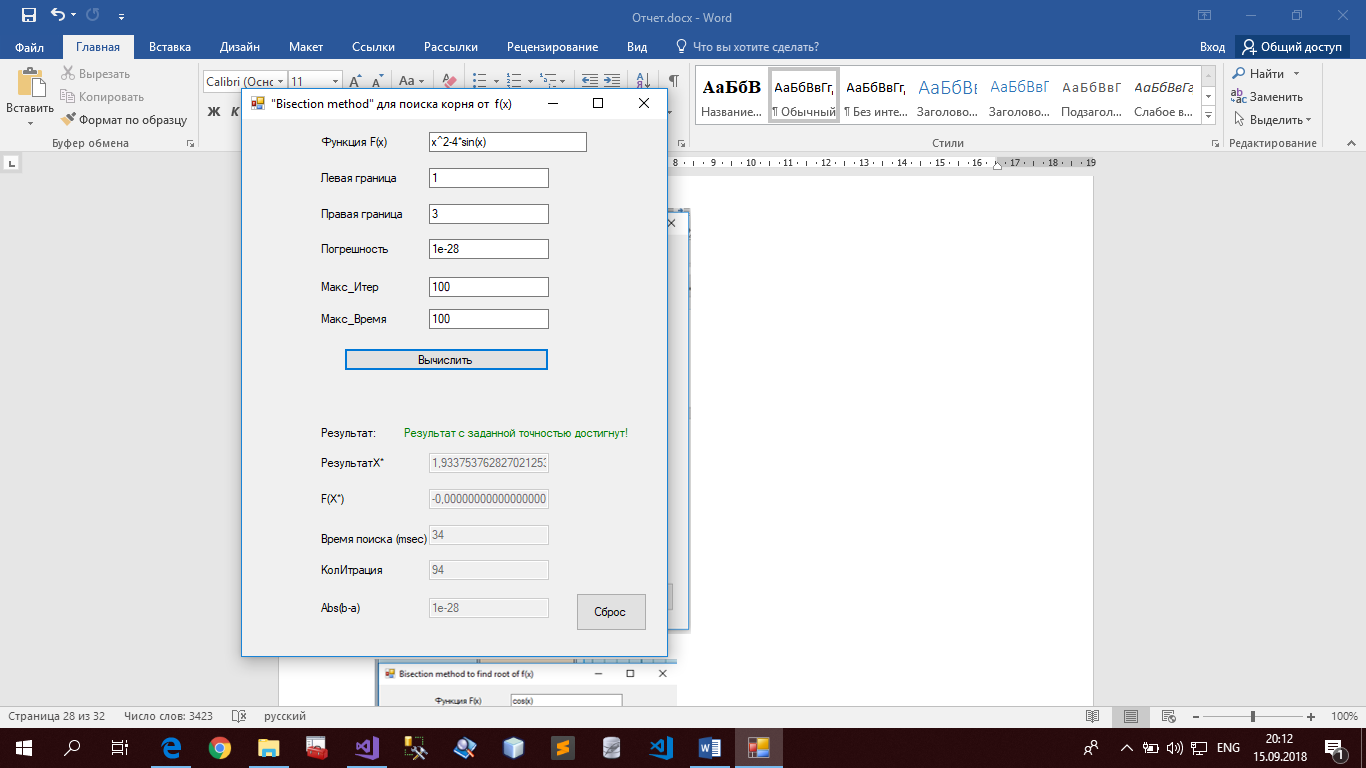
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*это примеры использования парсинга “ *Extreme.Numerics.Net40.dll* ”

[GoTo Contents](#Contents)

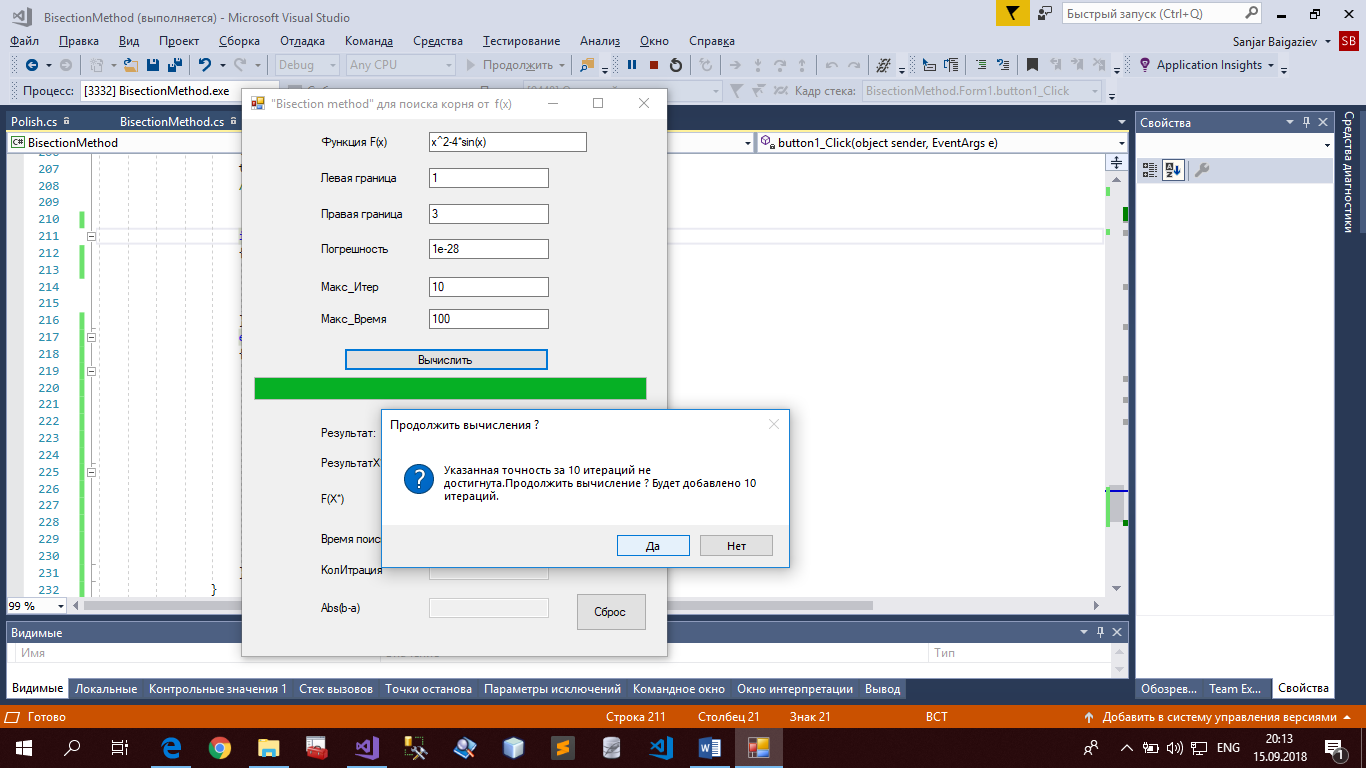
# Результат работы программы, реализующей Bisection method:

1. Проверка программы на разные точности

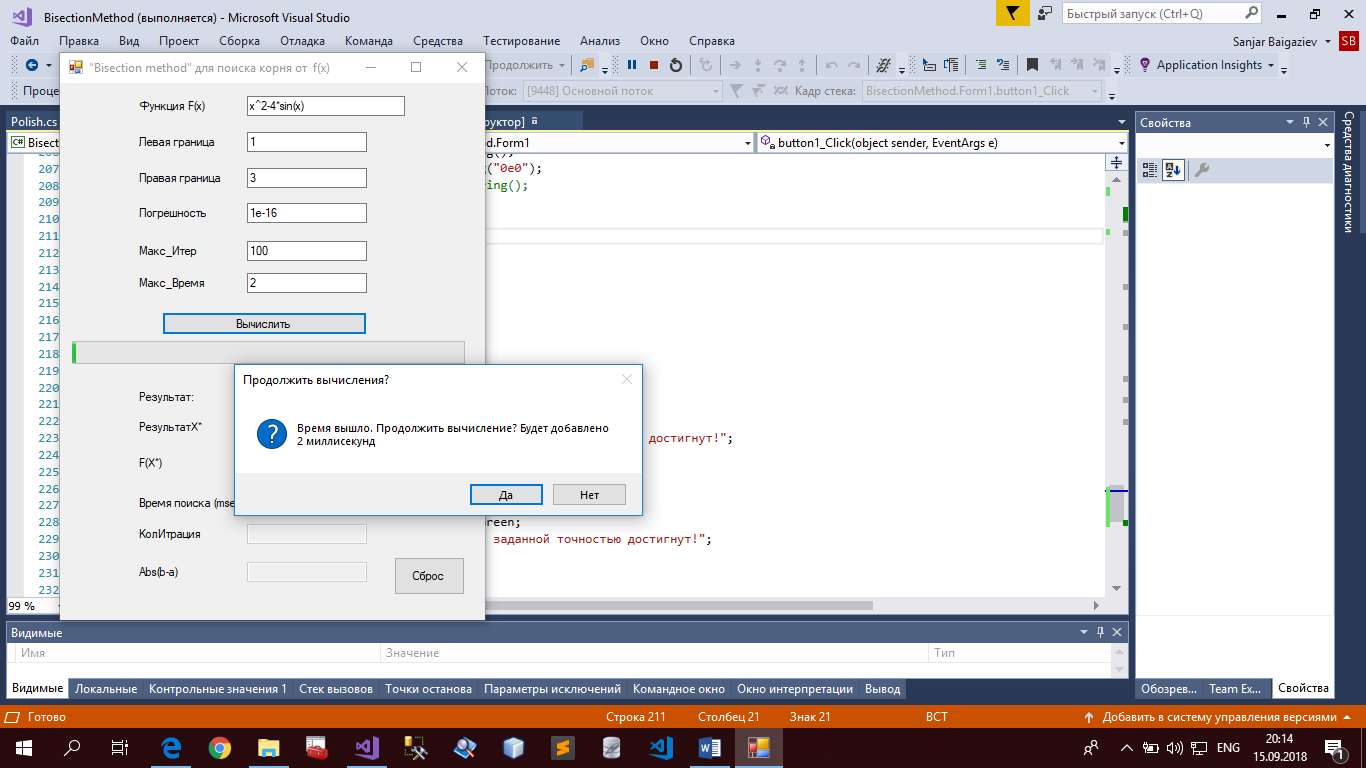




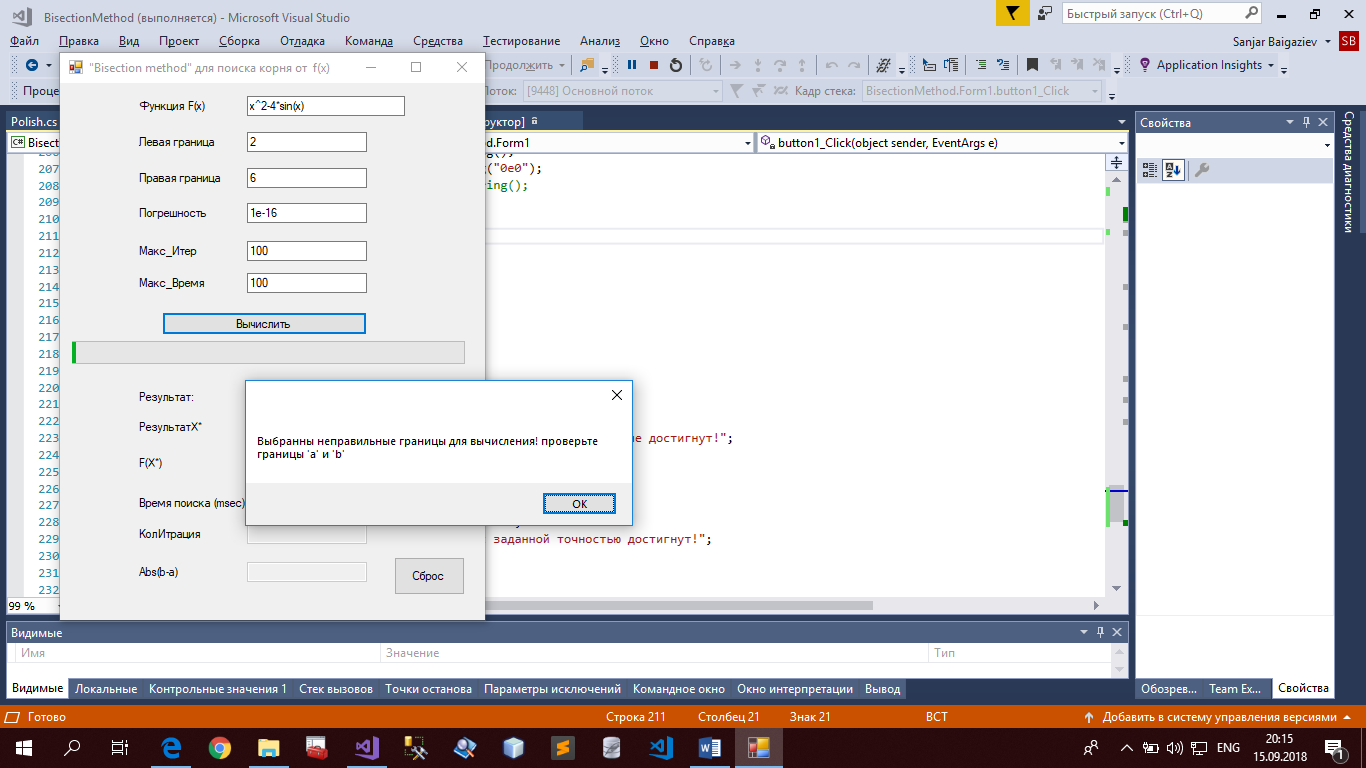
1. Остановка программы по лимиту итераций

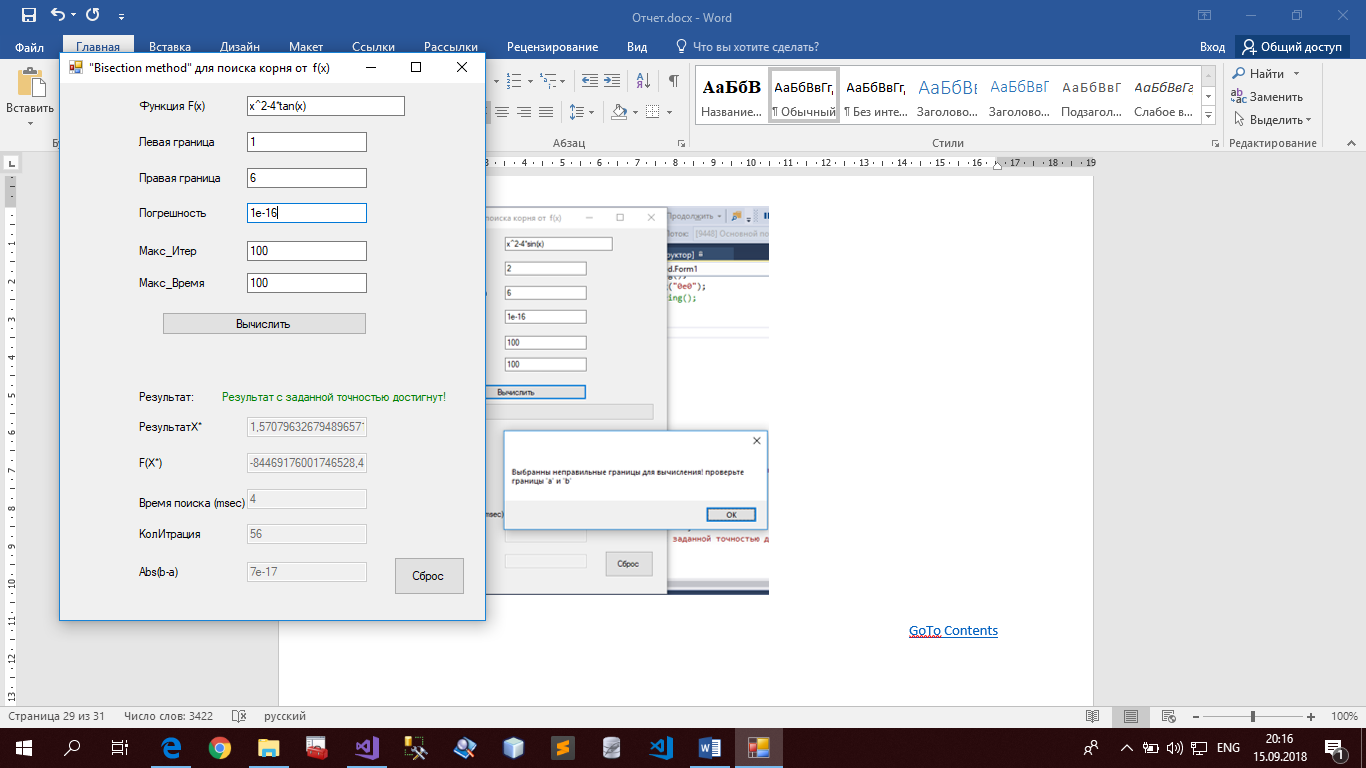


1. Остановка программы по лимиту времену

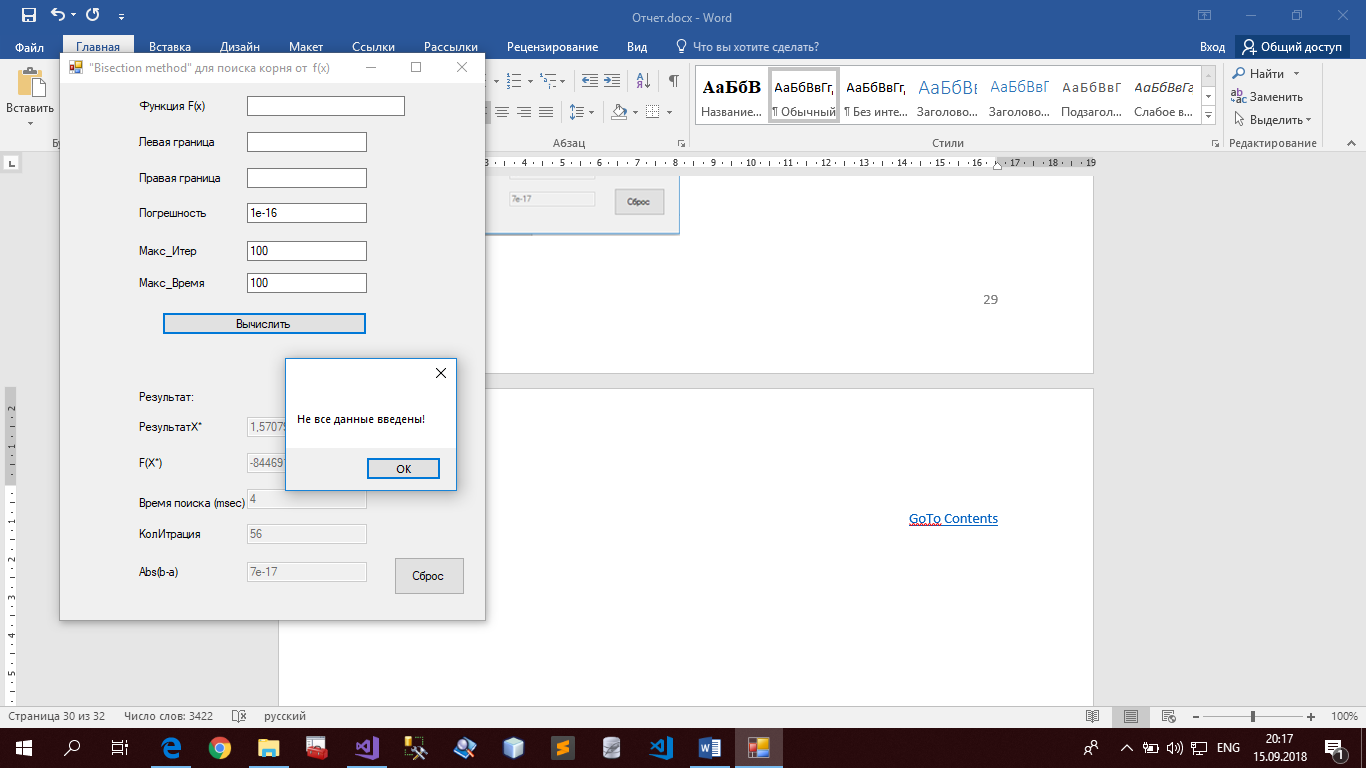


1. Не правильно выбраны границы поиска





1. Некоторые поля не заполнены



[GoTo Contents](#Contents)

**References**

1. [\\192.168.8.125\Buffer\ТенИГ\_ОсновыРАТПО(ПИ-15)\_2018.09.04\TenIG\_ОснРАТПО(Пособие-2a\_ПримерИсходногоТЗнаBisectionM\_Ver-9\_11pages)\_2016.09.02.docx](\\\\192.168.8.125\\Buffer\\ТенИГ_ОсновыРАТПО(ПИ-15)_2018.09.04\\TenIG_ОснРАТПО(Пособие-2a_ПримерИсходногоТЗнаBisectionM_Ver-9_11pages)_2016.09.02.docx)
2. [\\192.168.8.125\Buffer\ТенИГ\_ОсновыРАТПО(ПИ-15)\_2018.09.04\TenIG\_ОснРАТПО(Пособие-2b\_ПримерФинТЗнаBM\_Ver-16\_38pages)\_2017.09.04.docx](file:///\\192.168.8.125\Buffer\ТенИГ_ОсновыРАТПО(ПИ-15)_2018.09.04\TenIG_ОснРАТПО(Пособие-2a_ПримерИсходногоТЗнаBisectionM_Ver-9_11pages)_2016.09.02.docx)
3. [\\192.168.8.125\Buffer\ТенИГ\_ОсновыРАТПО(ПИ-15)\_2018.09.04\TenIG\_ОснРАТПО(Пособие-3\_ПримерСтадийПроект\_Ver-7\_29стр)\_2016.09.01.docx](file:///\\192.168.8.125\Buffer\ТенИГ_ОсновыРАТПО(ПИ-15)_2018.09.04\TenIG_ОснРАТПО(Пособие-2a_ПримерИсходногоТЗнаBisectionM_Ver-9_11pages)_2016.09.02.docx)
4. [\\192.168.8.125\Buffer\ТенИГ\_ОсновыРАТПО(ПИ-15)\_2018.09.04\TenIG\_ОснРАТПО(Пособие-4\_ОбразецОтчетаПоЛаб-1\_Ver-5\_34pages)\_2017.08.29.docx](file:///\\192.168.8.125\Buffer\ТенИГ_ОсновыРАТПО(ПИ-15)_2018.09.04\TenIG_ОснРАТПО(Пособие-2a_ПримерИсходногоТЗнаBisectionM_Ver-9_11pages)_2016.09.02.docx)
5. [\\192.168.8.125\Buffer\ТенИГ\_ОсновыРАТПО(ПИ-15)\_2018.09.04\SREA\_Проект-2\_Проверка\_ElapseTime\_12pages\_2017.09.30.docx](file:///\\192.168.8.125\Buffer\ТенИГ_ОсновыРАТПО(ПИ-15)_2018.09.04\SREA_Проект-2_Проверка_ElapseTime_12pages_2017.09.30.docx)

[GoTo Contents](#Contents)