Министерство образования и науки Кыргызской Республики

Кыргызский Государственный Технический Университет

им. И. Раззакова

Кафедра программного обеспечения компьютерных систем

НАПРАВЛЕНИЕ БАКАЛАВРСКОЙ ПОДГОТОВКИ 710400

ПРОГРАММНАЯ ИНЖЕНЕРИЯ

ОТЧЁТ

ПО ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ

Руководитель: кандидат технических наук, профессор

Тен Иосиф Григорьевич

Выполнила: Дашина Елена

Гр. ПИ-1-14

Бишкек 2017

**СОДЕРЖАНИЕ**

[**Глава 1: Описание проблемы** 2](#_Toc492501193)

[**1.1. Задание:** 2](#_Toc492501194)

[**Глава 2: Описание Bisection Method** 2](#_Toc492501195)

[**Глава 3: Спецификация требований к программному обеспечению** 3](#_Toc492501196)

[**3.1. Наименование программной разработки** 3](#_Toc492501197)

[**3.2. Бизнес-цель разработки программного обеспечения** 3](#_Toc492501198)

[**3.3. Функциональные требования к разработке программного обеспечения** 4](#_Toc492501199)

[**3.4. Нефункциональные требования на разработку программного обеспечения** 5](#_Toc492501200)

[**3.5. Ограничения на разработку программного обеспечения** 6](#_Toc492501201)

[**Глава 4: Образец дизайна интерфейсных форм системы, реализующей Bisection method:** 7](#_Toc492501202)

[**Глава 5: Стадии проектирования системы для поиска корня нелинейного уравнения f(x)=0, реализующей Bisection Method:** 8](#_Toc492501203)

[**5.1: Блок-схема Bisection method** 8](#_Toc492501204)

[**5.2: Проектирование интерфейса** 9](#_Toc492501205)

[**5.3: Документирование процесса** 9](#_Toc492501206)

[**Глава 6: Стадии конструирования системы для поиска корня нелинейного уравнения f(x)=0, реализующей Bisection Method:** 11](#_Toc492501207)

[**6.1: Код программы на C#, ассоциированный с интерфейсной формой** 11](#_Toc492501208)

[**6.2: Код программы на C#, ассоциированный с классом** 12](#_Toc492501209)

[**Глава 7: Результат работы программы, реализующей Bisection method:** 14](#_Toc492501210)

# **Глава 1: Описание проблемы**

Найти корень произвольного нелинейного уравнения – ноль нелинейной функции f(x) с заданной допустимой погрешностью Tolerance не менее 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) и. т. д.), которая имеет математический смысл, и для которой существует хотя бы одно решение задачи.

Проблема данной задачи отражается в затрате времени и денег на дополнительных сотрудников.

# **1.1. Задание:**

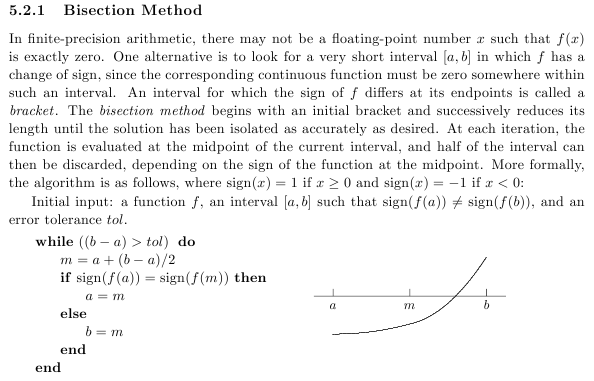
Задача №27:

Изучить тему \_\_ из главы №1 учебника [1, 2].

Разработать программу для определения корня произвольного нелинейного уравнения f(x)=0. Эта программа должна методом Bisection Method найти корень уравнения f(x)=0 с допустимой погрешностью не менее 1E–28. Задание:

1. Разработать входной интерфейс программы.
2. Разработать блок-схему алгоритма вычисления.
3. Разработать выходной интерфейс программы.
4. Продемонстрировать валидность программы.
5. Разработать программу для поиска корней системы нелинейных уравнений.
6. Составить перечень функциональных требований, реализованных в программе.

# **Глава 2: Описание Bisection Method**



# **Глава 3: Спецификация требований к программному обеспечению**

## **3.1. Наименование программной разработки**

Система поиска корней нелинейного уравнения методом деления отрезка пополам (Bisection Method)

## **3.2. Описание бизнес-процесса и анализ расхождения**

При решении различных бизнес-задач отделу исследований (пользователям) 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 нахождением 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 два специалиста высокой квалификации), не всегда обеспечивается приемлемая и 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.

## **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., 0.0000000000000000000000000001)***, 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. Разработка пользовательских требований**

Польза от компьютерной системы, поиска корней нелинейного уравнения методом деления отрезка пополам, является достижение более высокой точности с минимальной затратой времени и исключает надобность в дополнительных сотрудниках. Пользовательскими требованиями к ПО являются следующее:

* Для достижения бизнес-цели разработки программы нужно добиться необходимой точности решения за заданное время и количество итераций.
* Необходимо удостовериться в достижении заданной точности.

**3.5. Пользовательская история разработки программного обеспечения**

***Пользователь:***

* должен вводить нелинейную функцию ***f(x)***;
* значения концов интервала ***[a, b]***;
* значение допустимой погрешности ***Tolerance*** решения задачи;
* допустимое максимальное количество ***k\_max*** итераций;
* время, за которое производится введённое количество итераций.

***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 дать возможность ввести числовые значения концов интервала ***[a, b]*** и преобразовать их в числовые значения концов интервала;
* должна найти решение задачи – корень нелинейного уравнения ***f(x)=0***;
* должна выводить найденное значение ***х*** 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 ***Time(msec)*** для оценки производительности программы;
* должна выводить затраченное на поиск решения с заданной погрешностью количество итераций ***K***;
* должна выводить достигнутое значение абсолютной ошибки решения ***Abs(b–a)*** в формате 0Е0.



## **3.6. 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 обеспечить проверку условия, что левый конец интервала меньше правого конца, то есть a < b;
  + Система должна обеспечивать проверку условия . Выполнение этого условия гарантирует корректность значений, введенных начальных числовых значений концов интервала ***[a; b]***, которые являются необходимым условием для правильной работы Bisection Method;
* 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 обеспечивать возможность ввода любых числовых значений допустимой погрешности Tolerance как в формате с фиксированной точностью “0,000000000000001”, так и в экспоненциальном формате “1e-15”;
  + 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).

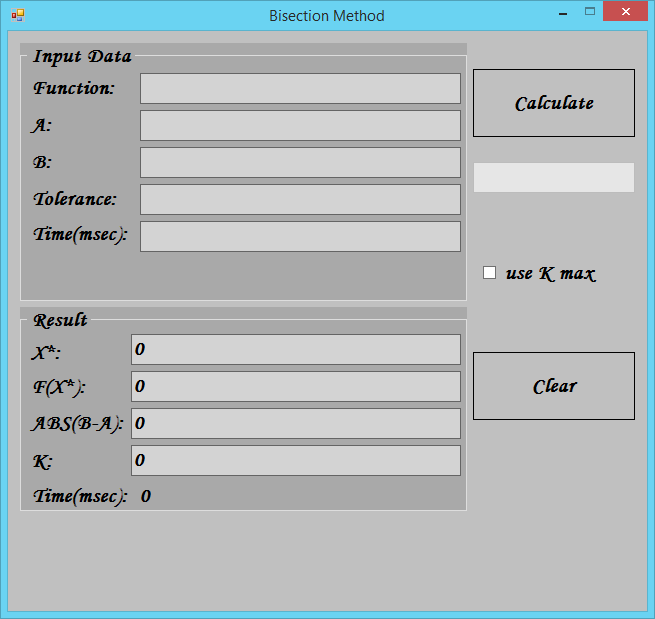
## **3.7. Нефункциональные требования на разработку программного обеспечения**

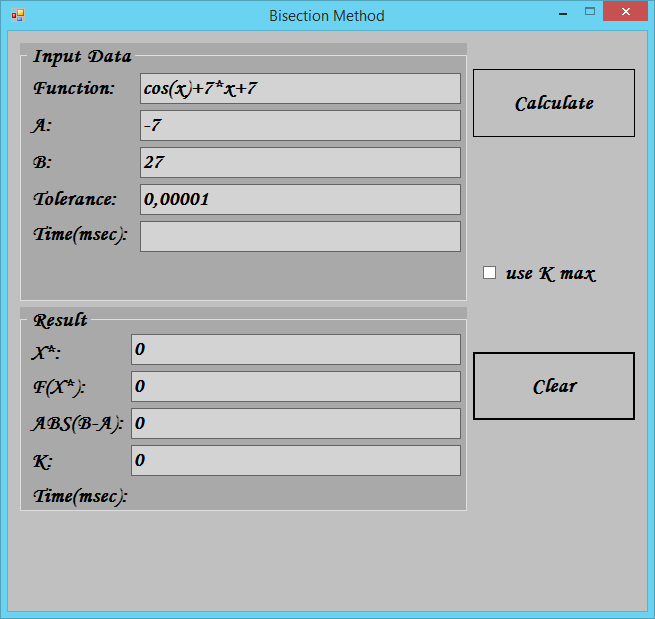
* Программа в процессе поиска решения должна отображать индикатор состояния процесса выполнения операции Progress Bar, чтобы пользователь понимал, что система не зависла, а находится в рабочем состоянии;
* В случае ошибки во введенном аналитическом выражении нелинейной функции f(x) система должна предоставлять 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;

## **3.8. Ограничения на разработку программного обеспечения**

* Нелинейная функция ***f(x)*** должна быть непрерывной на интервале [a; b];
* Разработать ПО на языке C++ / C# / Visual Basic;
* Система должна решать задачи с допустимой погрешностью не более 1e-15;
* 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 соответствующую документацию (техническое задание на ПО, руководство пользователя и руководство программиста) не должны превышать $1000;
* Система должна быть разработана для эксплуатации на компьютерах типа пентиум с оперативной памятью не более 2 гигабайт;
* Разработка программы должна начаться не позже 15 июня 2017 года, и система должна быть передана заказчику не позднее 1июля 2017 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:
  + Стандартных функций *– sin, cos, tg, ctg, arcsin, arccos, arctg, arcctg, sh, ch, th, cth, exp, lоg, ln, sqrt, fac, sinh, cosh, tanh, abs, ceil, floor, sfac, round, fpart.*
  + Основных вычислительных операций: сложение (+), вычитание (-), умножение (\*), деление (/) и возведение в степень (^);
  + круглых и квадратных скобок любой вложенности;
* Система не предназначена для ввода логических и алгоритмически заданных нелинейных функций, потому что понятие «корень нелинейного уравнения» относится к понятию числа, который несовместим с понятием логических значений «ложь-истина» вышеуказанных нелинейных функций.

# **Глава 4: Образец дизайна интерфейсных форм системы, реализующей Bisection method:**





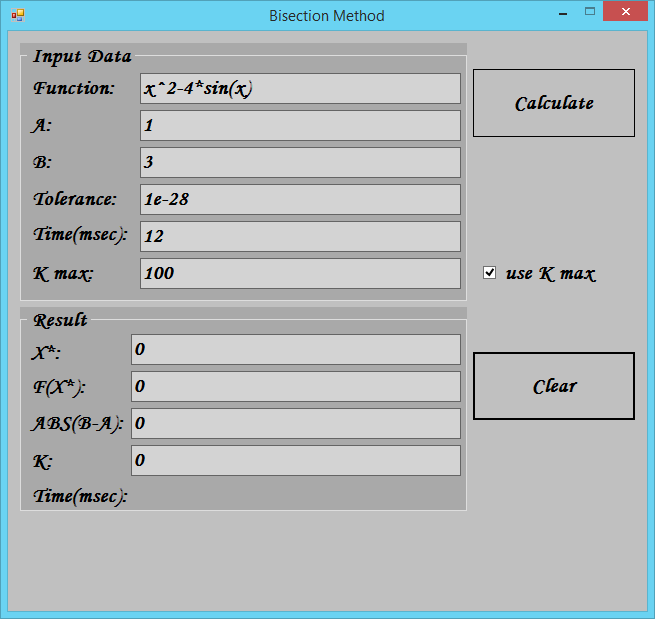
# **Глава 5: Стадии проектирования системы для поиска корня нелинейного уравнения f(x)=0, реализующей Bisection Method:**

## **5.1: Блок-схема Bisection method**

Рисунок 1: Блок-схема метода Bisection с конструкцией входных и выходных интерфейсов



**5.2: Проектирование интерфейса** **системы**, реализующей Bisection method и соответсвующих приведенной на рисунке 1 блок-схеме этого метода.



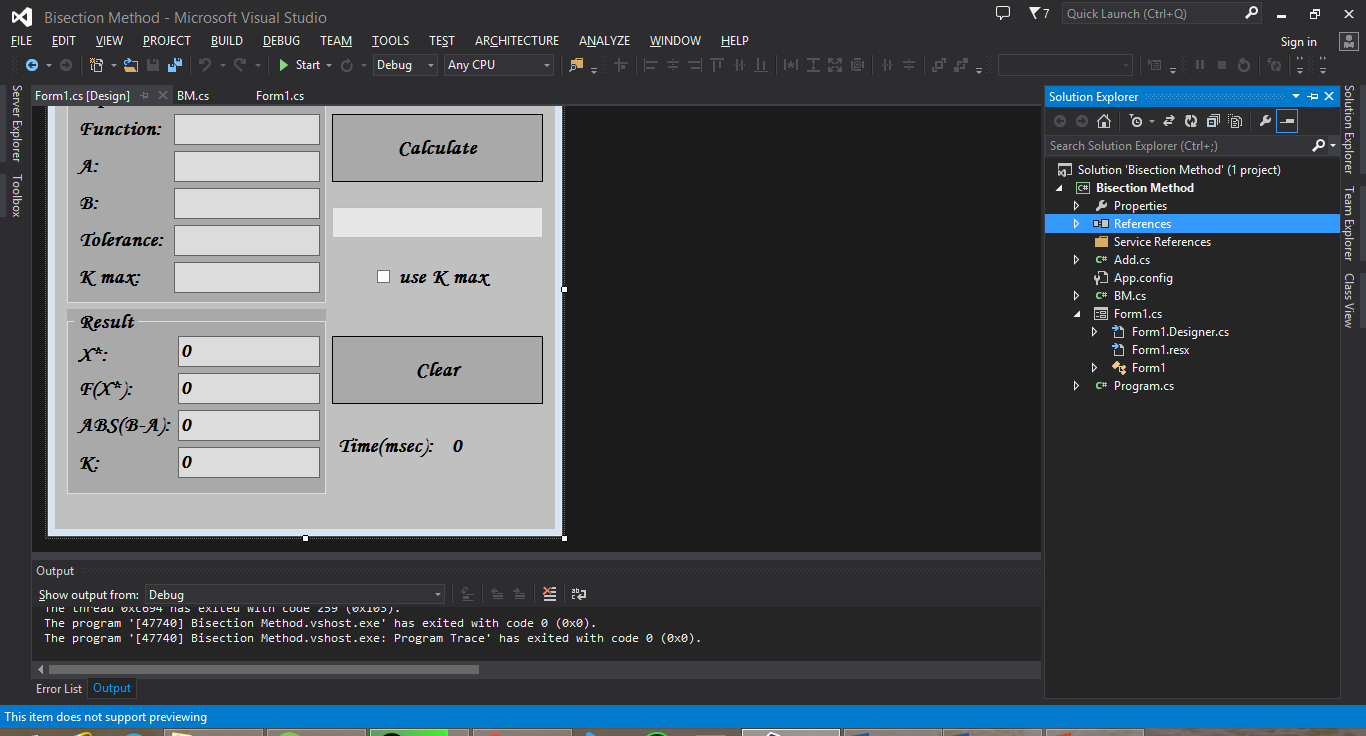
**5.3: Документирование процесса** задания свойств элементов интерфейсной формы системы, выбранной для внедрения и реализующей Bisection method

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

|  |  |  |  |
| --- | --- | --- | --- |
| **Number of control** | **Control** | **Property** | **Setting** |
| 0 | groupBox1 | Appearance (Text) | Input Data |
| groupBox1 | Design (Name) | groupBox1 |
| 1 | Label1 | Appearance (Text) | Function: |
| Label1 | Design (Name) | Label1 |
| Textbox1 | Design (Name) | fTB1 |
| 3 | Label2 | Appearance (Text) | A: |
| Label2 | Design (Name) | Label2 |
| Textbox2 | Design (Name) | aTB2 |
| 5 | Label3 | Appearance (Text) | B: |
| Label3 | Design (Name) | Label3 |
| Textbox3 | Design (Name) | bTB3 |
| 7 | Label4 | Appearance (Text) | Tolerance: |
| Label4 | Design (Name) | Label4 |
| Textbox4 | Design (Name) | tTB4 |
| 9 | Label5 | Appearance (Text) | K max: |
| Label5 | Design (Name) | Label5 |
| Textbox5 | Design (Name) | kmTB5 |
| CheckBox1 | Design (Name) | chkKMax |
| CheckBox1 | Appearance (Text) | use K max |
| CheckBox1 | Checked | False |
| 10 | Button1 | Appearance (Text) | Calculate |
| Button1 | Design (Name) | Button1 |
| 11 | groupBox2 | Appearance (Text) | Result |
| groupBox2 | Design (Name) | groupBox2 |
| 12 | ProgressBar1 | Behavior (Visible) | False |
| ProgressBar1 | Design (Name) | ProgressBar1 |
| 13 | Label6 | Appearance (Text) | X\*: |
| Label6 | Design (Name) | Label6 |
| 14 | Textbox6 | Design (Name) | xTB6 |
| Textbox6 | Behavior (ReadOnly) | True |
| 15 | Label7 | Appearance (Text) | f(X\*): |
| Label7 | Design (Name) | Lable7 |
| 16 | Textbox7 | Behavior (ReadOnly) | True |
| Textbox7 | Design (Name) | fxTB7 |
| 17 | Label8 | Appearance (Text) | ABS(B–A): |
| Label8 | Design (Name) | Label8 |
| 18 | Textbox8 | Behavior (ReadOnly) | True |
| Textbox8 | Design (Name) | absTB8 |
| 19 | Label9 | Appearance (Text) | K: |
| Label9 | Design (Name) | Label9 |
| 20 | Textbox9 | Behavior (ReadOnly) | True |
| Textbox9 | Design (Name) | kTB9 |
| 21 | Button2 | Appearance (Text) | Clear |
| Button2 | Design (Name) | Button2 |
| 22 | Label10 | Design (Name) | Label10 |
| 23 | Label11 | Appearance (Text) | 0 |
| Label11 | Design (Name) | Label11 |
| 24 | Label12 | Appearance (Text) | Time(msec): |
| Label12 | Design (Name) | Label12 |
| 25 | Label13 | Design (Name) | Label13 |

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

**6.1: Код программы на C#, ассоциированный с интерфейсной формой** “Form1.cs”, который реализует функцию ввода исходных данных и вывода результатов и составляет отдельный модуль в виде Public Class “Form1”:



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

if (CheckValidG() == false) return;

BM bis = new BM();

string function = fTB1.Text;

decimal a = Decimal.Parse(aTB2.Text);

decimal b = Decimal.Parse(bTB3.Text);

double to = Double.Parse(tTB4.Text);

decimal Tol = (decimal)to;

if (b > a) {

progressBar1.Style = ProgressBarStyle.Marquee;

progressBar1.MarqueeAnimationSpeed = 30;

progressBar1.Show(); //x^2-4\*sin(x)

progressBar1.Style = ProgressBarStyle.Blocks;

progressBar1.Value = progressBar1.Maximum;

int k = 0;

bis.addFunction(function);

Stopwatch stopWatch = new Stopwatch(); int ti = 0;

if (chkKMax.Checked)

{ try { k = Int32.Parse(kmTB5.Text); }

catch { MessageBox.Show("Введите количество итераций!!!"); }

try {

ti = Int32.Parse(textBox1.Text);

bis.setBMvariables(a, b, Tol, k, ti);

bis.BiMeit();

kTB9.Text = bis.getL(); }

catch { MessageBox.Show("Введите время!!!"); } }

else

{ bis.setBMvariables(a, b, Tol);

bis.BiMe();

kTB9.Text = bis.getIterations().ToString(); }

label11.Text = bis.getM();

int tx = Tol.ToString().Length;

try {

if (bis.getX() > 0)

{ xTB6.Text = bis.getX().ToString().Substring(0, tx); }

else

{ xTB6.Text = bis.getX().ToString().Substring(0, tx + 1); } }

catch

{ xTB6.Text = Math.Round(bis.getX(), 28).ToString();

} bis.getR1();

decimal i1 = Math.Abs(bis.getB() - bis.getA());

absTB8.Text = i1.ToString("0E0");

double a1 = Double.Parse(absTB8.Text);

if (a1 > to) { label15.Text = "Погрешность не достигнута, нужно больше итераций!"; }

else { label15.Text = ""; }

fxTB7.Text = bis.getFx().ToString();

label10.Text = bis.getR();

if (label13.Text == bis.getR1())

{ label15.Text = "";

MessageBox.Show("Знаки f(a) и f(b) одинаковы, введите другой интервал"); }

if (label15.Text == bis.getTim())

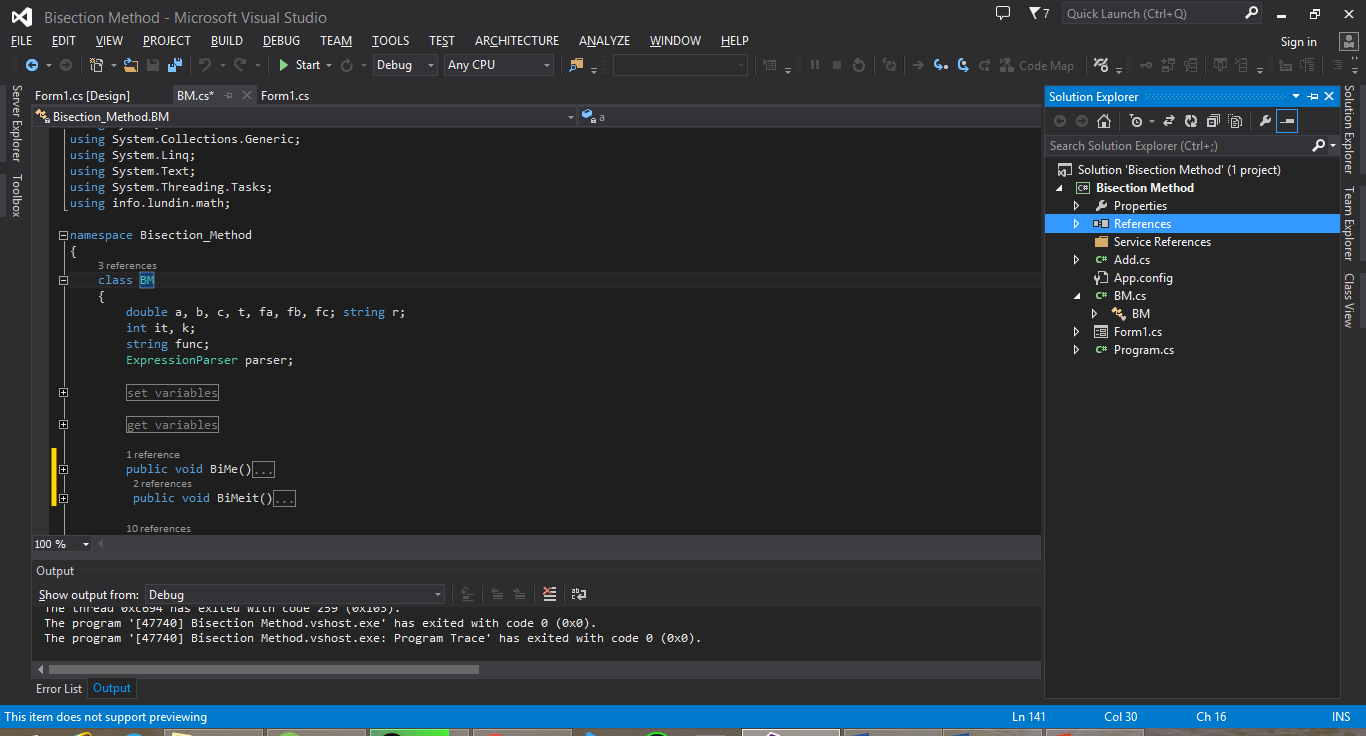
{ MessageBox.Show("Ошибка в функции"); label15.Text = ""; }

label13.Text = bis.getR1(); }

else

{ MessageBox.Show("Должно быть A < B"); }

**6.2: Код программы на C#, ассоциированный с классом** “BM.cs”, который реализует логику Bisection method по поиску корня нелинейного уравнения и составляет отдельный класс в виде Public Class “BM”:



public void BiMe()

{ try {

parser.Values.Add("x", (double)с);

fс = (decimal)parser.Parse(func); }

catch {

tim = ""; }

it = 1;

Stopwatch Watch = new Stopwatch();

Watch.Start();

while (Math.Abs(b - a) > t) {

if (it > 1000000) {

r1 = "Превышено время ожидaния, введите точность меньше";

с = 0;

fс = 0;

a = 0;

b = 0;

it = 1;

break; }

else {

fa = f(a);

fb = f(b);

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

r1 = ""; с = 0;

fс = 0; a = 0;

b = 0; it = 1;

break; }

else {

с = (a + b) / 2;

it++; c1 = с;

fa = f(a); fb = f(b);

fс = f(с);

if (Math.Sign(fa) == Math.Sign(fс))

{ a = с; fa = fс; }

else {

b = с; fb = fс; }

if (fс == 0)

{ с = c1;

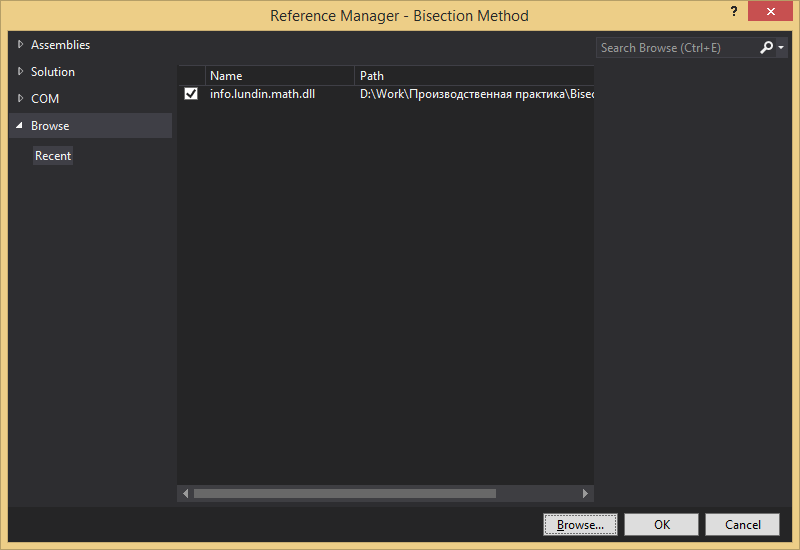
break; } } }

} parser.Values.Clear();

Watch.Stop();

m = Watch.Elapsed.Milliseconds.ToString(); }

**6.3: Подключение библиотечной программы “info.lundin.math.dll” к программному проекту для выполнения функции парсинга:**



***Parsing:***

string func;

ExpressionParser parser;

………………

private decimal f(decimal x)

{

decimal res = 0.0;

try

{

parser.Values["x"].SetValue((double)x);

res = (decimal)parser.Parse(func);

}

catch

{

r1 = "Ошибка в функции";

}

return res;

}

………………

decimal a = Decimal.Parse(aTB2.Text)

………………

# **Глава 7: Результат работы программы, реализующей Bisection method:**

