**Министерство образования и науки Кыргызской Республики**

**Кыргызский Государственный Технический Университет**

**им. И. Раззакова**

**Кафедра программного обеспечения компьютерных систем**

**БАКАЛАВРСКАЯ УЧЕБНАЯ ПРОГРАММА**

**НАПРАВЛЕНИЕ—710400 «ПРОГРАММНАЯ ИНЖЕНЕРИЯ»**

**Дисциплина «Методы оптимизации»**

ОТЧЕТ

**НА ПРАКТИЧЕСКИЕ РАБОТЫ**

по дисциплине

«Технология командной разработки»

Выполнил: ст. гр. ПИ 1-13

Ким А. Темиров А.

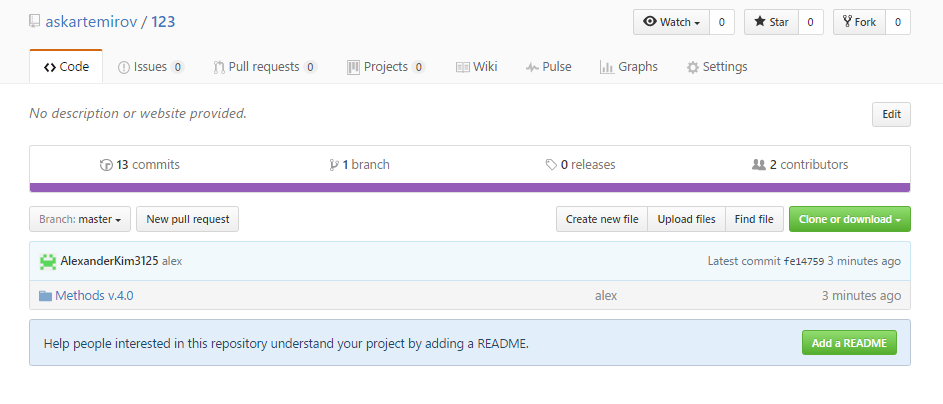
Проверила: Саабаева К.К.

**Бишкек 2016**

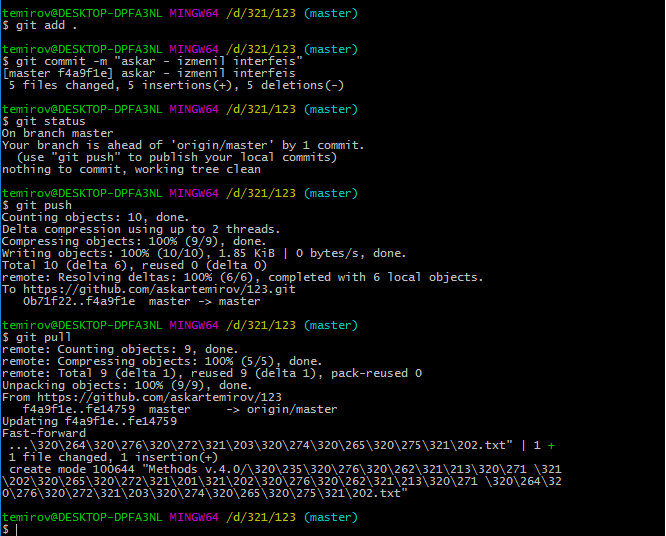
**Работа с GIT.**

Для разработки проекта “Методы оптимизации” будем использовать распределенную систему контроля версий “GIT”, это требуется для командной разработки ПО.

Мы создали на сайте <https://github.com/>, репозиторий с именем “123”, далее начинаем работать с Git bash(командная строка).

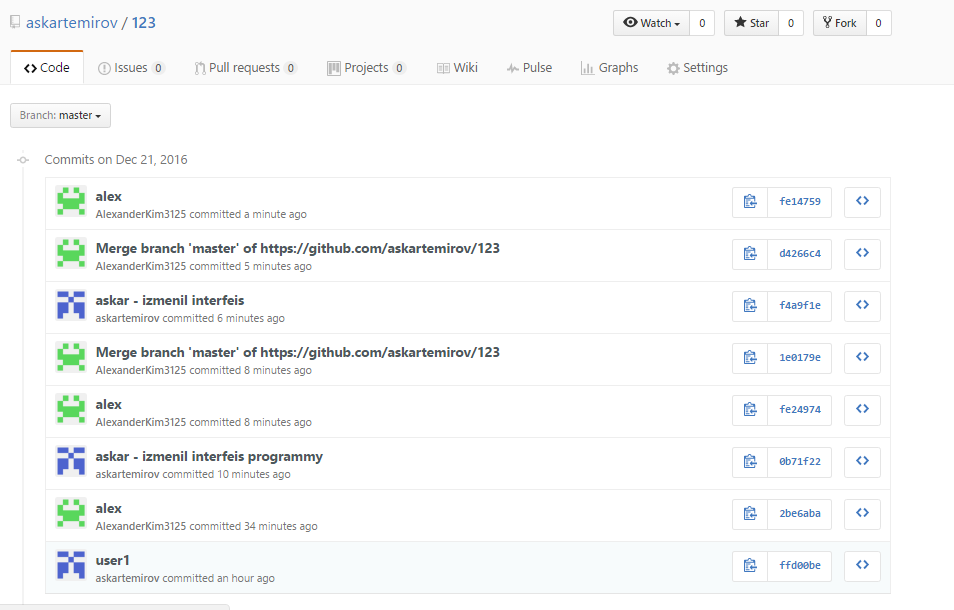


Отправка и получение файлов через <https://github.com/>



**Список изменений**

Изменения в проекте, кто в какое время из разработчиков изменил проект.



## Практическая работа №1

## ЧАСТЬ №1

«Разработка ПО для поиска минимума ***нелинейной унимодальной функции***

на основе итерационного метода: ***Even Search Method***

**Что дано:**

* ***Спецификация проблемы*** поиска минимума нелинейной функции f(x);
* ***Спецификация*** итерационного метода (***Even Search Method***) для нахождения минимума ***нелинейной унимодальной функции***
* ***Блок-схема*** Even Search method;
* ***Интерфейсная форма*** системы поиска минимума ***нелинейной унимодальной функции***, реализующей итерационный метод — ***Even Search Method***;
* ***Тесты*** для проверки ПО.

**Что требуется:**

* Разработать ***проект*** ПО для поиска минимума произвольной ***нелинейной унимодальной функции*** ***f(x)*** для произвольной заданной допустимой погрешности;
* ПО для поиска минимума произвольной нелинейной функции f(x) для произвольной заданной допустимой погрешности;
* Провести валидацию системы – доказать идентичность результатов решения задач с помощью разработанного ПО заданным тестам.
* Сконструировать систему, реализующую итерационный метод Even Search Method на основе использования парсера;

Разработать ***код***

## ЧАСТЬ №2

### Раздел №1

**Наименование работы –** Нахождение минимума нелинейной функции *методом равномерного поиска*

### Раздел №2

**СПЕЦИФИКАЦИЯ ПРОБЛЕМЫ №2: Нахождение МИНИМУМА нелинейной функции методом равномерного поиска**

* Найти минимум произвольной нелинейной функции

с заданной допустимой погрешностью ***Tolerance*** методом равномерного поиска. Нелинейная функция 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.), который имеет математический смысл, и для которой существует хотя бы одно решение задачи.

### Раздел №3

**СПЕЦИФИКАЦИЯ (Описание) метода равномерного поиска**

Метод относится к пассивным стратегиям. Задается количество интервалов N, на которое разбивается исходный интервал L0 = [a0, b0]. Вычисления производятся в N +1 равноотстоящих друг от друга точках. Путем сравнения величин f(xi), i = 0,1,…,N находится точка xk, в которой значение функции наименьшее. Искомая точка минимума считается заключенной в интервале [xk-1, xk+1].

**Описание алгоритма равномерного поиска полностью:**

Предположим, что нам задана задача максимизации целевой функции типа , то метод равномерного поиска имеет формулу (для решения максимизации):

if  */\*Нахождение максимума функции методом равномерного поиска\*/*

then 

else 



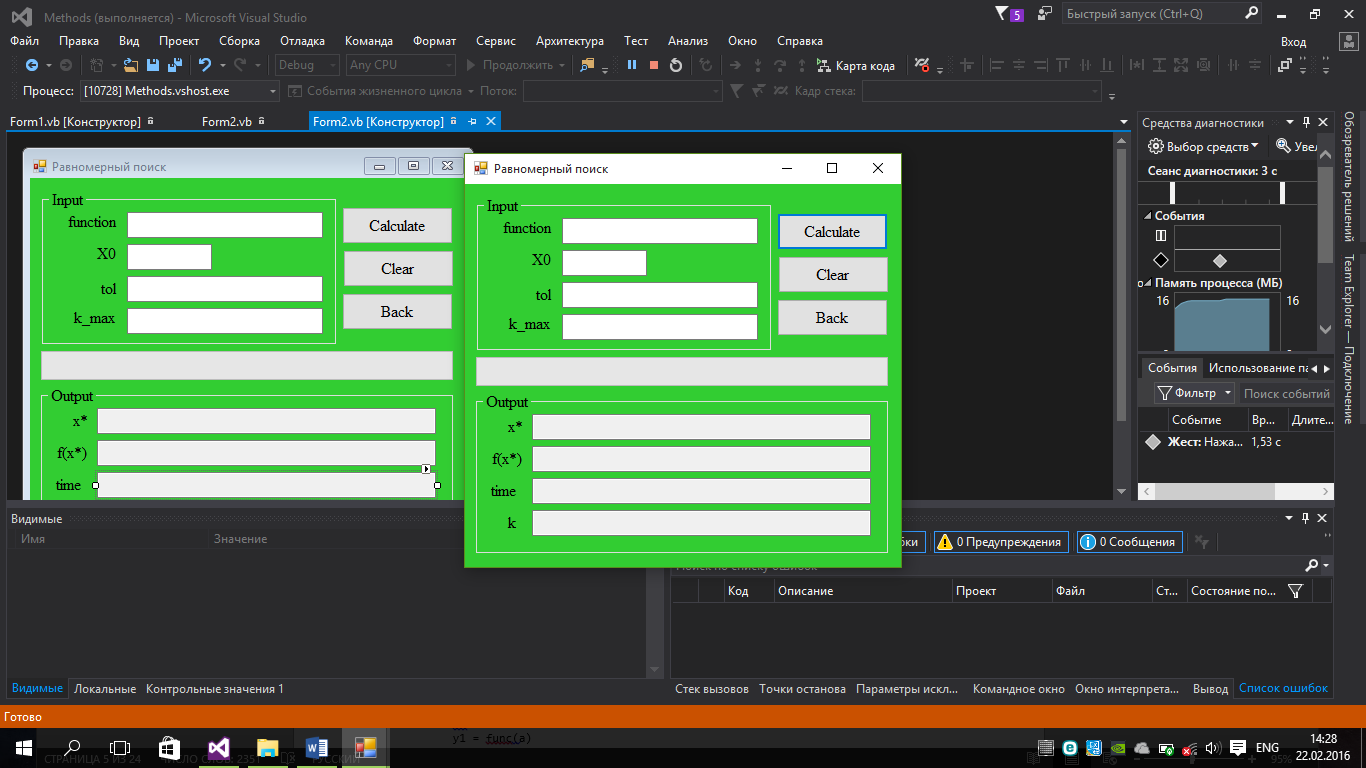
### Раздел №4

**Стадии *проектирования* системы для поиска минимума нелинейной функции , реализующей метод равномерного поиска:**

Stage No.1: Разработка блок-схемы метода равномерного поиска



Stage No.2: Проектирование интерфейса системы, реализующей метод равномерного поиска



### Раздел №5

**ПРОЕКТИРОВАНИЕ По**: ДОКУМЕНТИРОВАНИЕ ЭТАПОВ ПРОЕКТИРОВАНИЯ ИнтерфейснОЙ формЫ системЫ, реализующЕЙ итерационный метод –мЕТОД РАВНОМЕРНОГО ПОИСКА

|  |  |  |  |
| --- | --- | --- | --- |
| **Number of control** | **Control** | **Property** | **Setting** |
| 1 | Button | Appearance (Text) | Calculate |
| Design (Name) | Button1 |
| 2 | Button | Appearance (Text) | Clear |
| Design (Name) | Button2 |
| 3 | Button | Appearance (Text) | Back |
| Design (Name) | Button3 |
| 4 | GroupBox1 | Appearance (Text) | input |
| Design (Name) | GroupBox1 |
| 5 | Textbox1 | Appearance (Text) |  |
| Design (Name) | Textbox1 |
| 6 | Textbox2 | Appearance (Text) |  |
| Design (Name) | Textbox2 |
| 7 | Textbox3 | Appearance (Text) |  |
| Design (Name) | Textbox3 |
| 8 | Textbox4 | Appearance (Text) |  |
| Design (Name) | Textbox4 |
| 9 | Textbox6 | Appearance (Text) |  |
| Design (Name) | Textbox6 |
| 10 | Textbox7 | Appearance (Text) |  |
| Design (Name) | Textbox7 |
| 11 | Textbox8 | Appearance (Text) |  |
| Design (Name) | Textbox8 |
| 12 | Textbox9 | Appearance (Text) |  |
| Design (Name) | Textbox9 |
| 13 | GroupBox2 | Appearance (Text) | Output |
| Design (Name) | GroupBox2 |
| 14 | Label1 | Appearance (Text) | function |
| Design (Name) | Label1 |
| 15 | Label2 | Appearance (Text) | X0 |
| Design (Name) | Label2 |
| 16 | Label3 | Appearance (Text) | K\_max |
| Design (Name) | Label3 |
| 17 | Label4 | Appearance (Text) | tol |
| Design (Name) | Label4 |
| 18 | Label6 | Appearance (Text) | X\* |
| Design (Name) | Label6 |
| 19 | Label7 | Appearance (Text) | F(x\*) |
| Design (Name) | Label7 |
| 20 | Label8 | Appearance (Text) | Time |
| Design (Name) | Label8 |
| 21 | Label9 | Appearance (Text) | k |
| Design (Name) | Label9 |
| 22 | ProgressBar1 | Appearance (Text) |  |
| Design (Name) | ProgressBar1 |

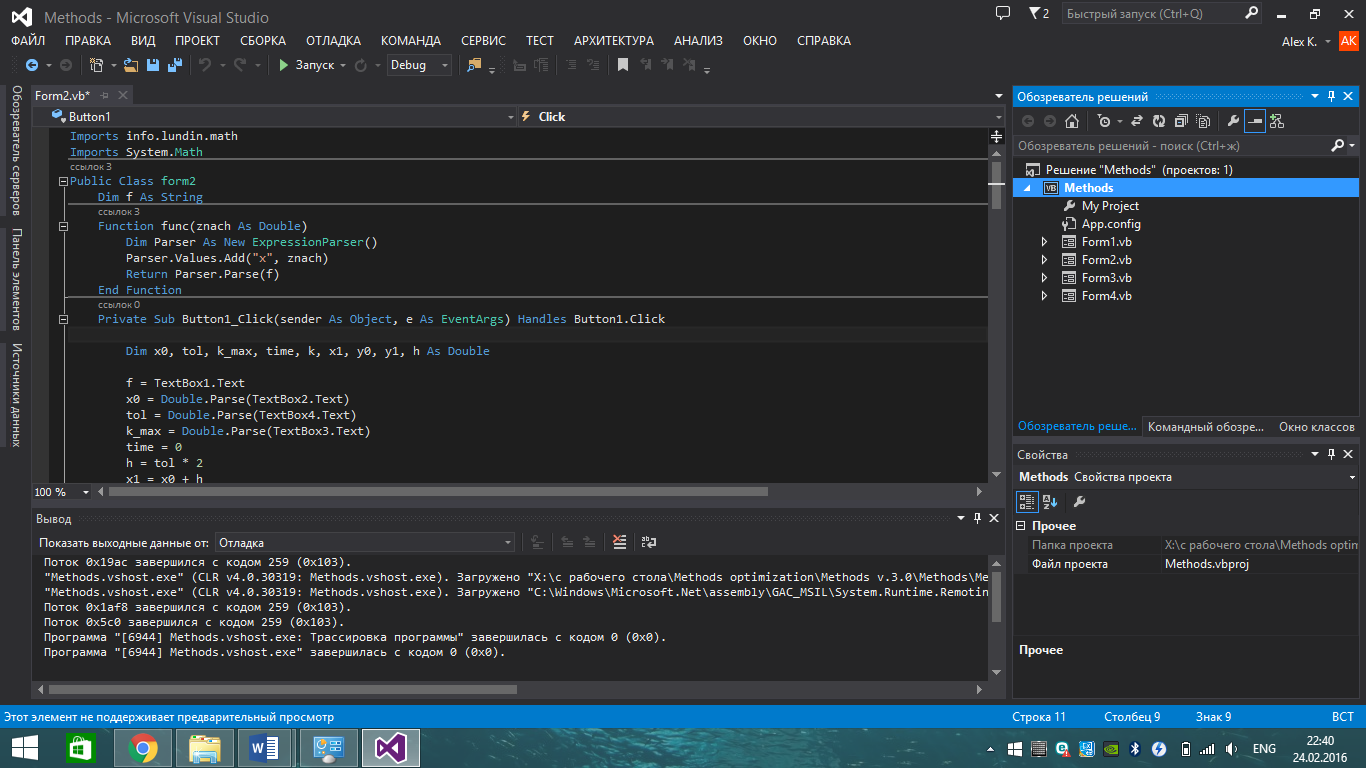
### Раздел №5

**Стадии *конструирования* системы для поиска минимума нелинейной функции , реализующей метод равномерного поиска:**

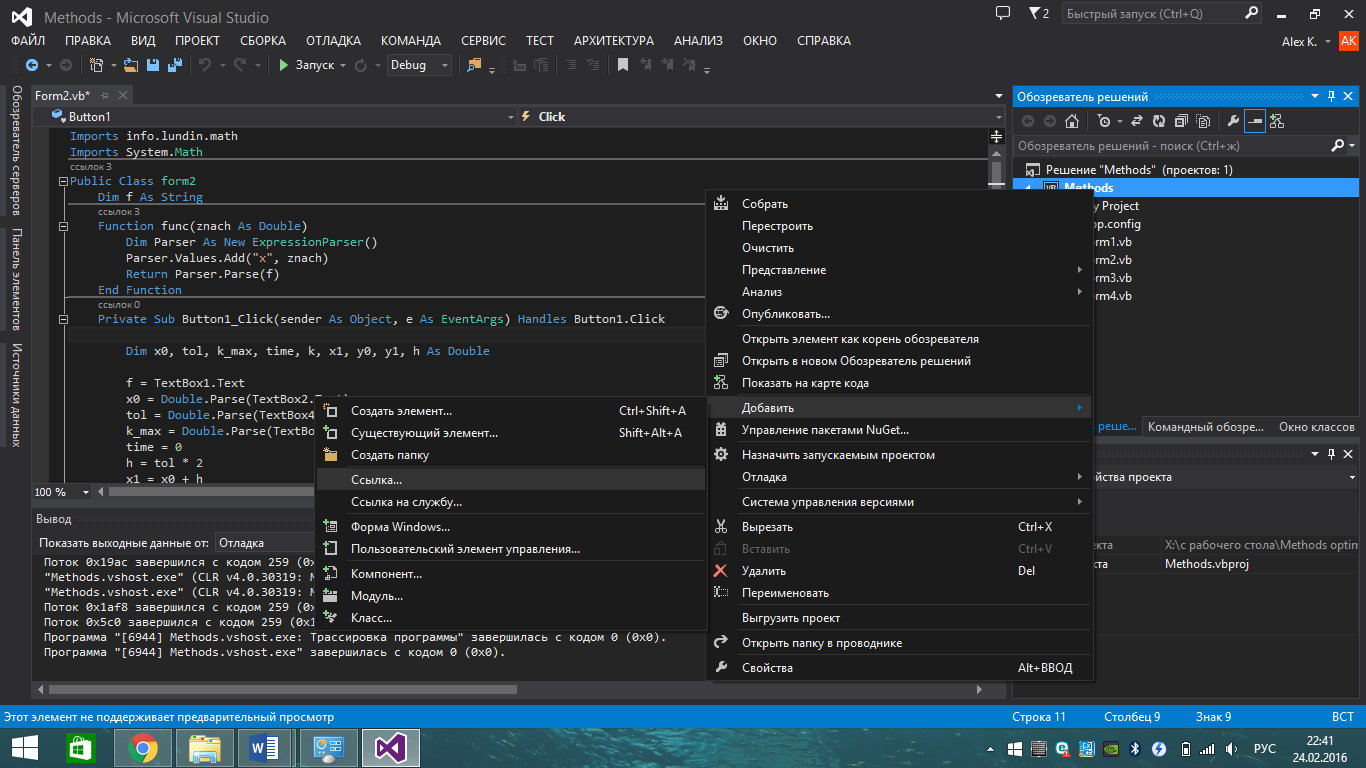
Stage №4: **Подключение библиотечной программы** “info.lundin.math.dll” к программному проекту для выполнения функции парсинга

***Примечание***: *Если вставить приведенные в этом отчете коды программы, то Visual Studio выделит строчки кода листинга программы, в которых есть ссылки на библиотечную функцию «info.lundin.math», как ошибочные. Это связано с тем, что в проект не включена ссылка на эту функцию. Ниже приведена инструкция по включению в проект парсера.*

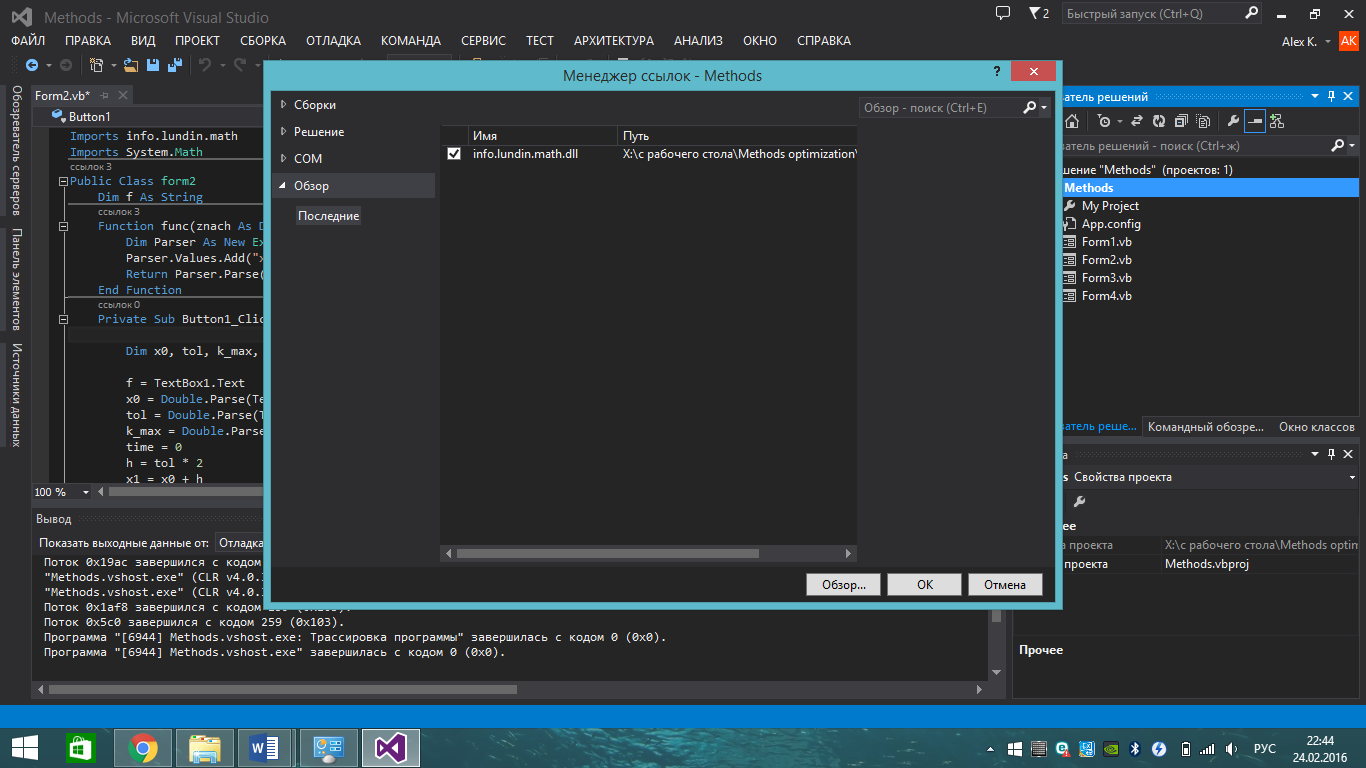
Шаг №3.1: Правой кнопкой мыши открыть контекстное меню на выделенной синим цветом строке:



Шаг №3.2: Щелкнуть мышкой на строчке «Добавить ссылку...»:



Шаг №3.3: В диалоговом окне менеджера ссылок ввести имя файла “info.lundin.math.dll”:



Шаг №3.4: В листинге программы исчезнут все пометки об ошибках в коде программы, связанных с тем, что оператор «Imports info.lundin.math» 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 “info.lundin.math.dll”

Function F(a As Double)

Dim Parser As New ExpressionParser()

Parser.Values.Add("x", a)

Return Parser.Parse(func)

End Function

Stage №.4.1**: Код программы на Visual Basic Метод равномерного поиска**

Imports info.lundin.math

Imports System.Math

Public Class form2

Dim f As String

Function func(znach As Double)

Dim Parser As New ExpressionParser()

Parser.Values.Add("x", znach)

Return Parser.Parse(f)

End Function

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

Dim x0, tol, k\_max, time, k, x1, y0, y1, h As Double

Dim begin As DateTime = Now

Dim bend As DateTime

f = TextBox1.Text

x0 = Double.Parse(TextBox2.Text)

tol = Double.Parse(TextBox4.Text)

k\_max = Double.Parse(TextBox3.Text)

time = 0

h = tol \* 2

x1 = x0 + h

y0 = func(x0)

y1 = func(x1)

k = 0

While (Abs(x1 - x0) > tol And k < k\_max)

k = k + 1

If (y1 > y0) Then

x1 = x0

y1 = y0

Else

x0 = x1

y0 = y1

x1 = x1 + h

y1 = func(x1)

End If

time = time + 0.0001

TextBox8.Text = Convert.ToString(time)

If (ProgressBar1.Value <> 100) Then

ProgressBar1.Value += 1

End If

End While

bend = Now

If (ProgressBar1.Value <> 100) Then

ProgressBar1.Value = 100

End If

TextBox7.Text = y1

TextBox9.Text = k

TextBox6.Text = x1

End Sub

Private Sub Button2\_Click(sender As Object, e As EventArgs) Handles Button2.Click

TextBox1.Text = ""

TextBox2.Text = ""

TextBox3.Text = ""

TextBox4.Text = ""

ProgressBar1.Value = 0

TextBox8.Text = bend.Subtract(begin).Milliseconds

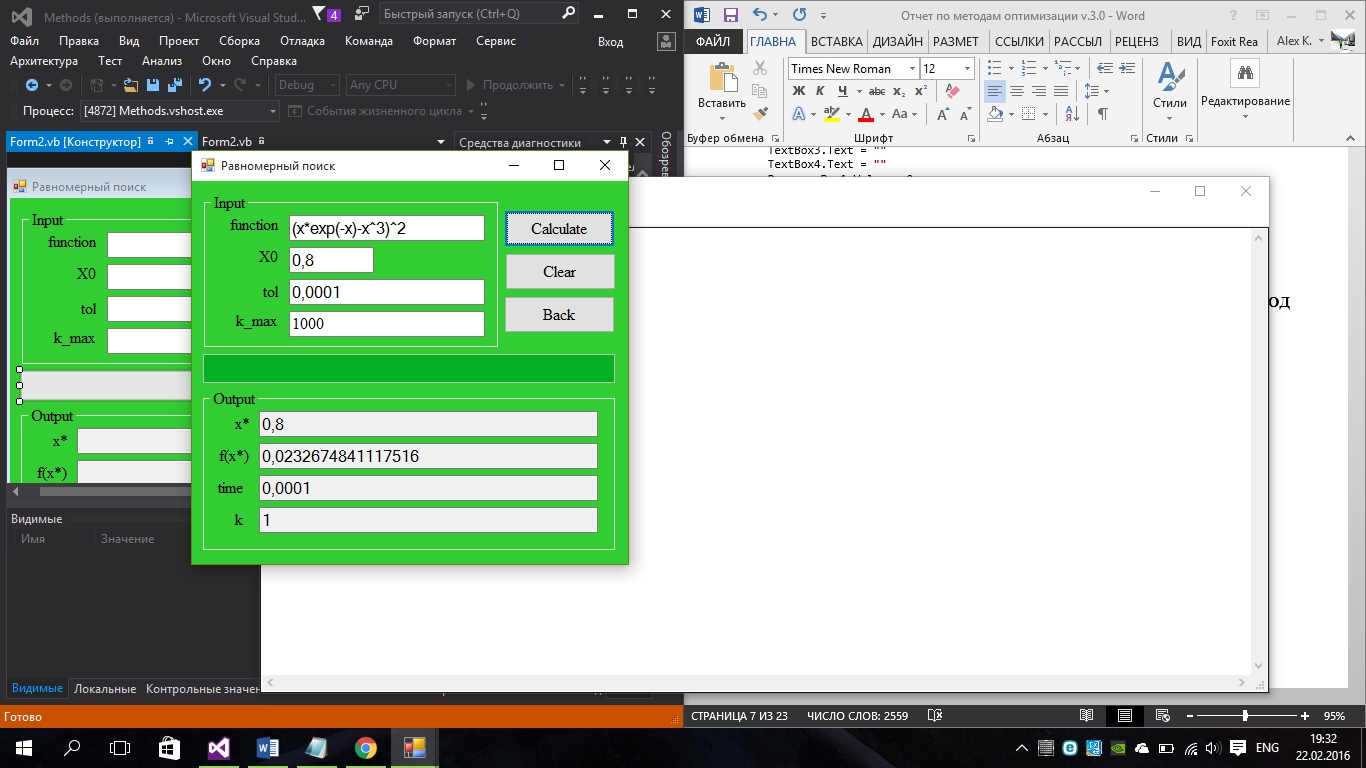
End Sub

End Class

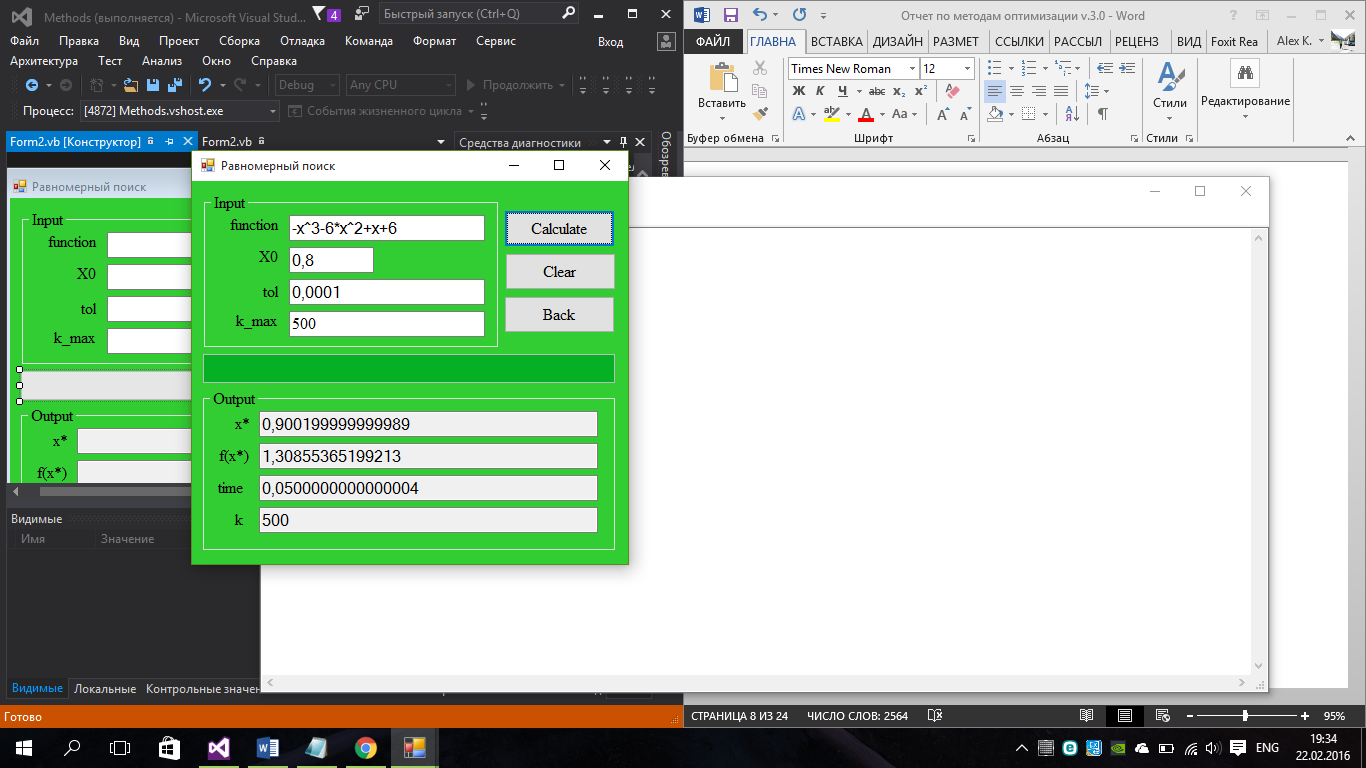
### Раздел №6

**ПРОВЕРКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ, РЕАЛИЗУЮЩЕГО МЕТОД РАВНОМЕРНОГО ПОИСКА**

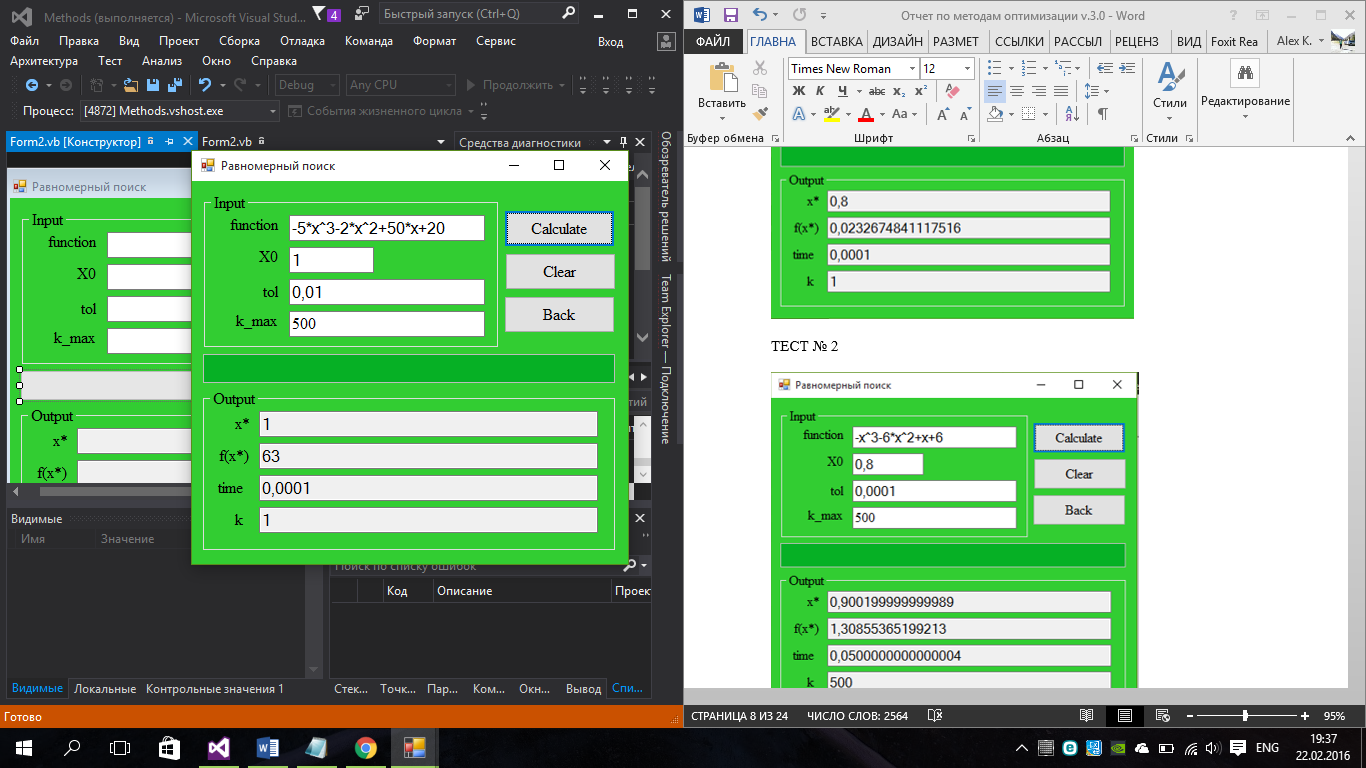
тЕСТ №1



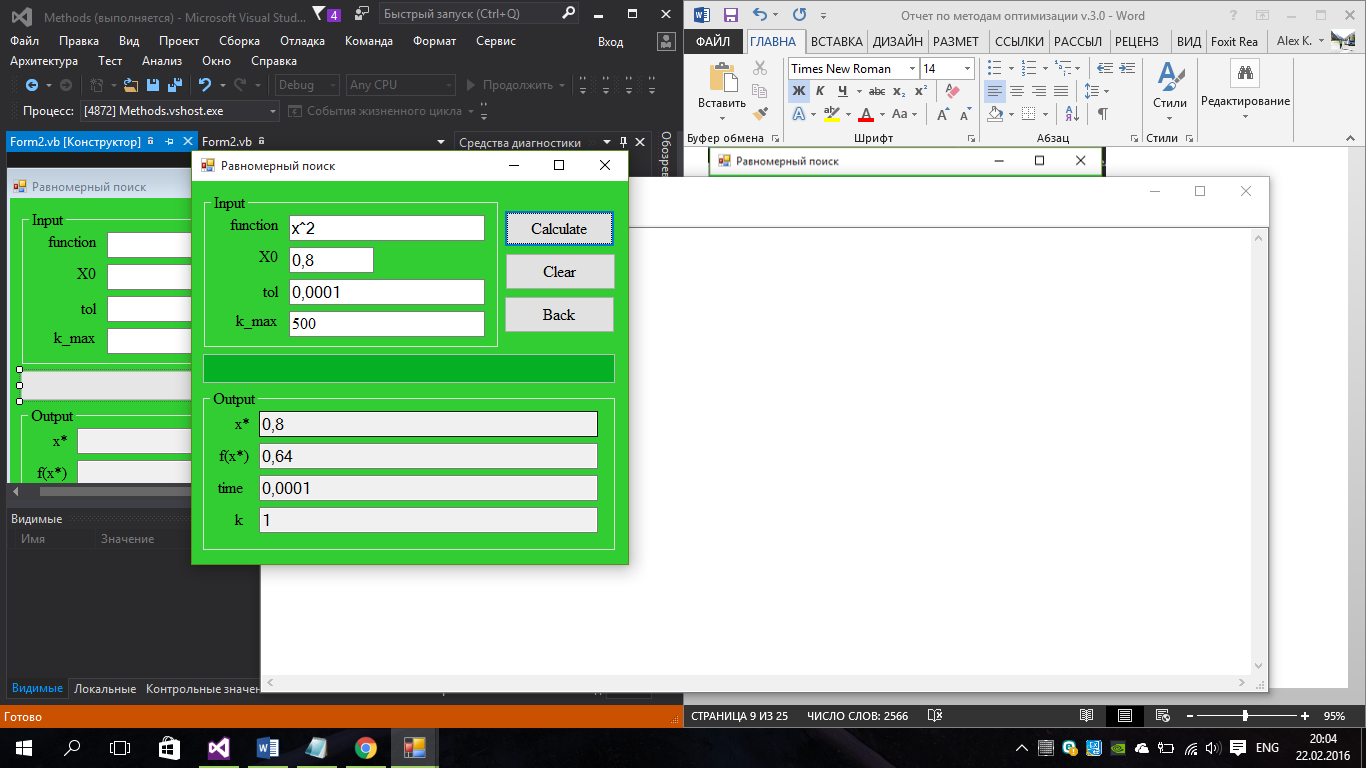
ТЕСТ № 2



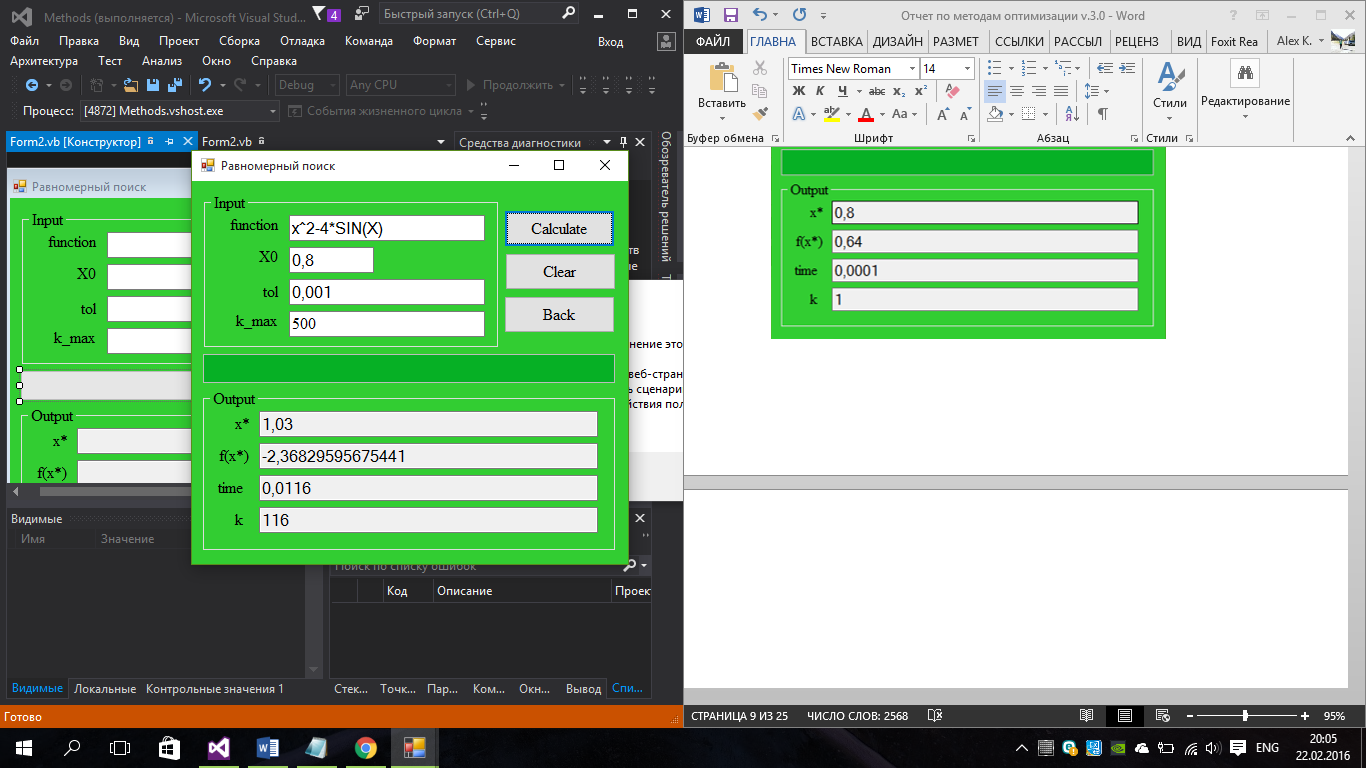
ТЕСТ №3



ТЕСТ №4



ТЕСТ №5



## Практическая работа №2

## ЧАСТЬ №1

«Разработка ПО для поиска ***минимума функций***

на основе *метода поразрядного приближения»*

**Что дано:**

* ***Спецификация проблемы*** поиска минимума произвольной нелинейной функции ;
* ***Спецификация*** метода *поразрядного приближения* для нахождения минимума нелинейной функции;
* ***Блок-схема*** метода *поразрядного приближения*;
* ***Интерфейсная форма*** системы поиска минимума нелинейной функции, реализующей метод *поразрядного приближения;*
* ***Тесты*** для проверки ПО.

**Что требуется:**

* Разработать ***проект*** ПО для решения произвольной оптимизационной задачи с произвольной погрешностью без ограничений методом поразрядного приближения
* Сконструировать систему, реализующую метод поразрядного приближения на основе использования парсера;
* Разработать ***код*** ПО для поиска минимума произвольной нелинейной функции для произвольно заданной допустимой погрешности.
* Провести валидацию системы – доказать идентичность результатов решения задач с помощью разработанного ПО заданным тестам.

## ЧАСТЬ №2

### Раздел №1

**Наименование работы –** Нахождение минимума нелинейной функции *методом поразрядного приближения*

### Раздел №2

**СПЕЦИФИКАЦИЯ ПРОБЛЕМЫ №2: Нахождение МИНИМУМА нелинейной функции методом поразрядного приближения**

* Найти минимум произвольной нелинейной функции

с заданной допустимой погрешностью ***Tolerance*** методом поразрядного приближения. Нелинейная функция 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.), который имеет математический смысл, и для которой существует хотя бы одно решение задачи.

### 

### Раздел №3

**СПЕЦИФИКАЦИЯ (Описание) метода поразрядного приближения**

Метод является усовершенствованным вариантом метода перебора. В этом методе перебор точек интервала неопределенности происходит с шагом h, i = 0, 1, … до тех пор, пока количество итераций не станет больше заданного или h будет меньше чем точность/параметр R.

После этого шаг уменьшается в несколько -R раз, и производится перебор точек в противоположном направлении (с новым шагом) до тех пор, пока значения f(x) не перестанут уменьшаться.

Процедура уменьшения шага и смены направления перебора на противоположное повторяется несколько раз. Поиск прекращается, если текущий шаг дискретизации при последнем проходе алгоритма не превосходит заданной точности.

**Описание алгоритма решения проблемы в виде пошаговой итерационной процедуры**

Шаг 1. Ввести исходные данные: a, b, e.

Шаг 2. Выбрать начальный шаг D = http://konspekta.net/studopediaorg/baza8/1675971933752.files/image017.png . Положить x0 = a. Вычислить f(x0).

Шаг 3. Положить x1 = x0 + D. Вычислить f(x1).

Шаг 4. Сравнить f(x0) и f(x1). Если f(x0) > f(x1), то перейти к шагу 5, иначе – к шагу 6.

Шаг 5. Положить x0 = x1 и f(x0) = f(x1). Проверить условие x0Î (a, b), т. е. a < x0< b. Если условие выполнено, перейти к шагу 3, иначе – к шагу 6.

Шаг 6. Проверка на окончание поиска. Если êDê£e, то вычисления завершить, положив x\* » x0, f(x\* )» f(x0), иначе – перейти к шагу 7.

Шаг 7. Изменение направления и шага поиска. Положить x0 = x1 и f(x0) = f(x1), D = – http://konspekta.net/studopediaorg/baza8/1675971933752.files/image019.png . Перейти к шагу 3.

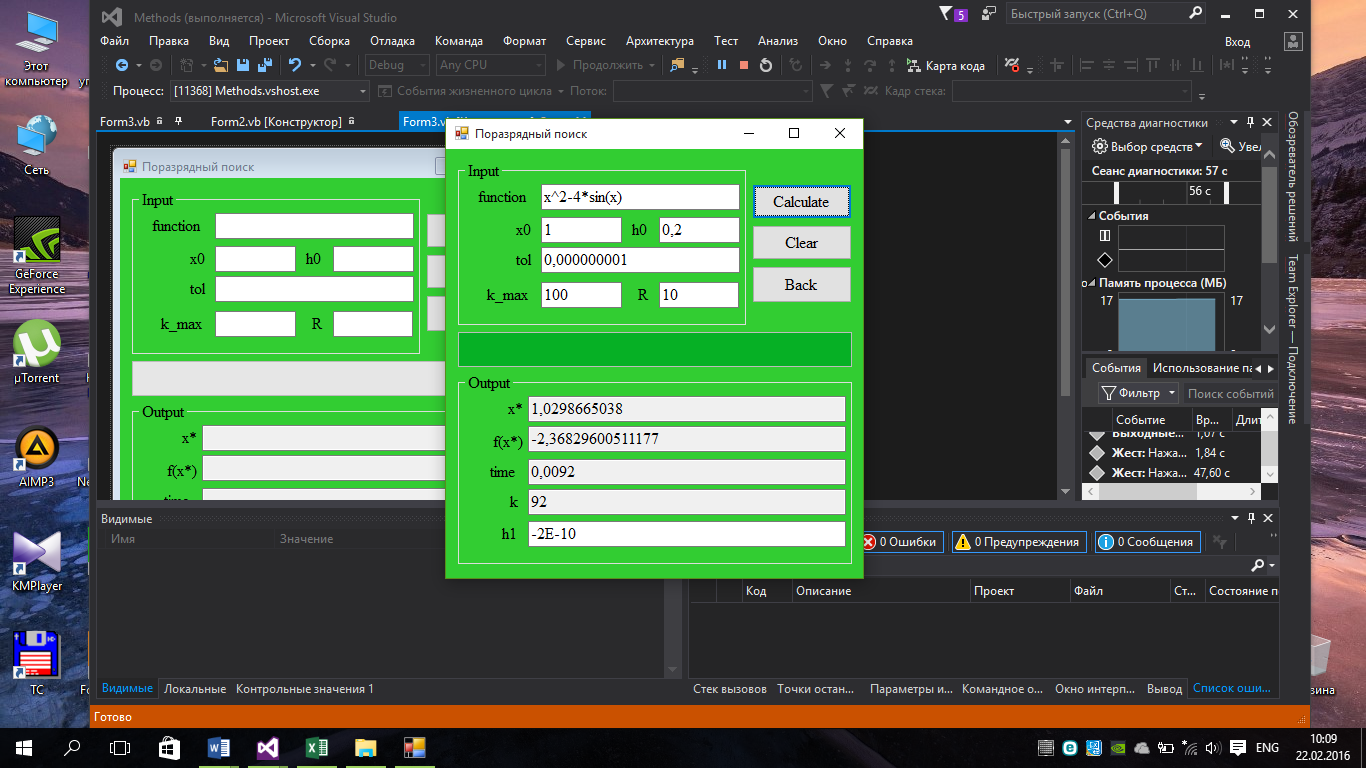
### Раздел №4

**Стадии *проектирования* системы для поиска минимума нелинейной функции , реализующей метод поразрядного приближения:**

Stage No.1: Разработка блок-схемы метода поразрядного приближения



Stage No.2: Проектирование интерфейса системы, реализующей метод поразрядного приближения



### Раздел №5

**Стадии *конструирования* системы для поиска минимума нелинейной функции , реализующей метод поразрядного приближения:**

Stage №.4: **Код программы на Visual Basic Метод поразрядного поиска**

Imports info.lundin.math

Imports System.Math

Public Class form3

Dim f As String

Function func(znach As Double)

Dim Parser As New ExpressionParser()

Parser.Values.Add("x", znach)

Return Parser.Parse(f)

End Function

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

If (TextBox1.Text = "" Or TextBox2.Text = "" Or TextBox3.Text = "" Or TextBox4.Text = "" Or TextBox5.Text = "") Then MsgBox("Неправильный ввод!") : GoTo 10

Dim x0, h0, tol, R, k\_max As Double

Dim x1, h1, y0, y1, k, time As Double

f = TextBox1.Text

x0 = Double.Parse(TextBox2.Text)

h0 = Double.Parse(TextBox3.Text)

tol = Double.Parse(TextBox4.Text)

R = Double.Parse(TextBox5.Text)

k\_max = Double.Parse(TextBox6.Text)

y0 = func(x0)

h1 = h0

x1 = x0 + h1

y1 = func(x1)

k = 0

While (Abs(h1) > tol And k < k\_max)

k = k + 1

If (y1 >= y0) Then

If (Abs(h0) < tol / R) Then

h1 = h0

x1 = x0

y1 = y0

Else3

h1 = -h0 / R

h0 = h1

x0 = x1

y0 = y1

x1 = x0 + h1

y1 = func(x1)

End If

Else

h1 = h0

x0 = x1

y0 = y1

x1 = x0 + h1

y1 = func(x1)

End If

time = time + 0.0001

TextBox9.Text = Convert.ToString(time)

If (ProgressBar1.Value <> 100) Then

ProgressBar1.Value += 1

End If

End While

If (ProgressBar1.Value <> 100) Then

ProgressBar1.Value = 100

End If

ProgressBar1.Value = 100

TextBox7.Text = x1

TextBox8.Text = func(x1)

TextBox10.Text = k

10:

Private Sub Button2\_Click(sender As Object, e As EventArgs) Handles Button2.Click

TextBox1.Text = ""

TextBox2.Text = ""

TextBox3.Text = ""

TextBox4.Text = ""

TextBox5.Text = ""

TextBox6.Text = ""

TextBox7.Text = ""

TextBox8.Text = ""

TextBox9.Text = ""

TextBox10.Text = ""

ProgressBar1.Value = 0

End Sub

Private Sub Button3\_Click(sender As Object, e As EventArgs) Handles Button3.Click

Me.Close()

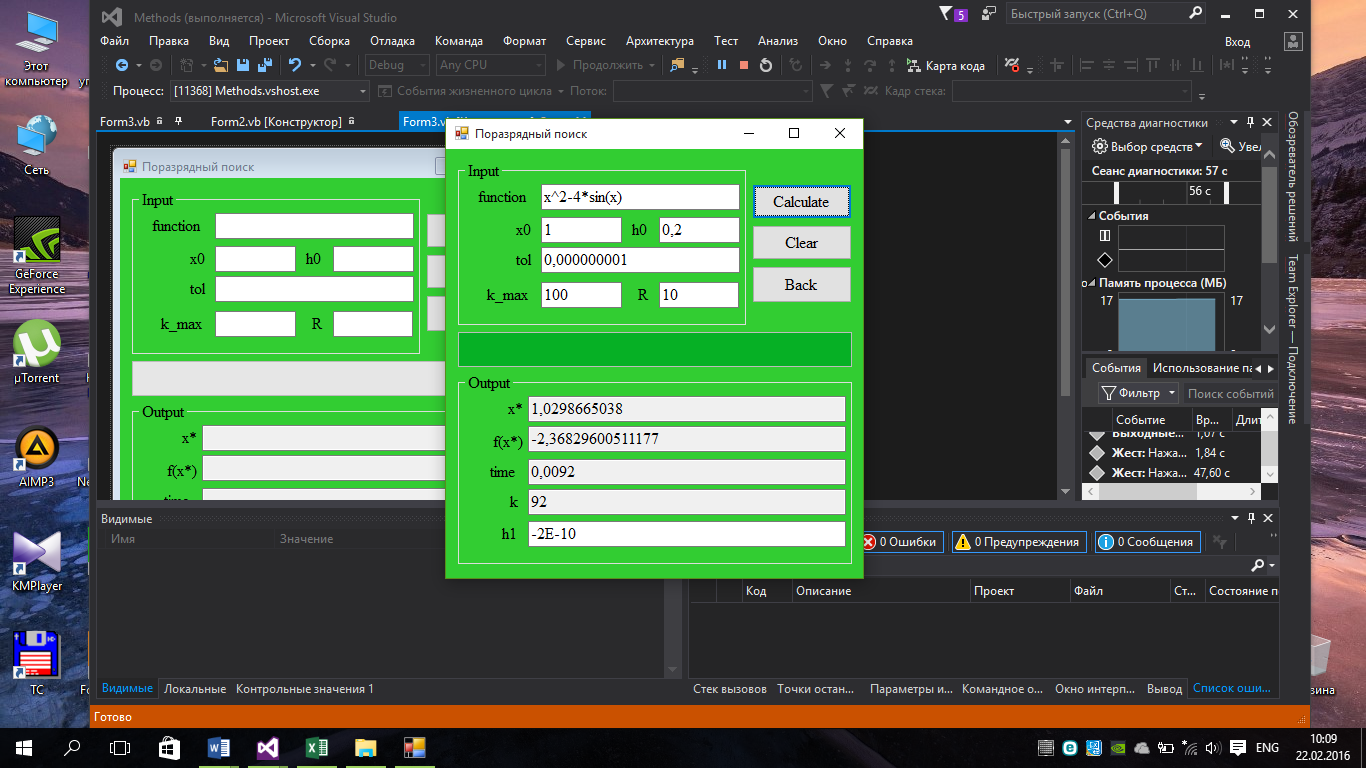
Form1.Show()

End Sub

End Class

### Раздел №6

**ПРОВЕРКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ, РЕАЛИЗУЮЩЕГО МЕТОД ПОРАЗРЯДНОГО ПРИБЛИЖЕНИЯ**



### Раздел №5

**ПРОЕКТИРОВАНИЕ По**: ДОКУМЕНТИРОВАНИЕ ЭТАПОВ ПРОЕКТИРОВАНИЯ ИнтерфейснОЙ формЫ системЫ, реализующЕЙ итерационный метод - МЕТОД ПОРАЗРЯДНОГО ПОИСКА

|  |  |  |  |
| --- | --- | --- | --- |
| **Number of control** | **Control** | **Property** | **Setting** |
| 1 | Button | Appearance (Text) | Calculate |
| Design (Name) | Button1 |
| 2 | Button | Appearance (Text) | Clear |
| Design (Name) | Button2 |
| 3 | Button | Appearance (Text) | Back |
| Design (Name) | Button3 |
| 4 | GroupBox1 | Appearance (Text) | input |
| Design (Name) | GroupBox1 |
| 5 | Textbox1 | Appearance (Text) |  |
| Design (Name) | Textbox1 |
|  |  |  |  |
|  |  |
| 6 | Textbox2 | Appearance (Text) |  |
| Design (Name) | Textbox2 |
| 7 | Textbox3 | Appearance (Text) |  |
| Design (Name) | Textbox3 |
| 8 | Textbox4 | Appearance (Text) |  |
| Design (Name) | Textbox4 |
| 9 | Textbox5 | Appearance (Text) |  |
| Design (Name) | Textbox5 |
| 10 | Textbox6 | Appearance (Text) |  |
| Design (Name) | Textbox6 |
| 11 | Textbox7 | Appearance (Text) |  |
| Design (Name) | Textbox7 |
| 12 | Textbox8 | Appearance (Text) |  |
| Design (Name) | Textbox8 |
| 13 | Textbox9 | Appearance (Text) |  |
| Design (Name) | Textbox9 |
| 14 | Textbox10 | Appearance (Text) |  |
| Design (Name) | Textbox10 |
| 15 | Textbox11 | Appearance (Text) |  |
| Design (Name) | Textbox11 |
| 16 | GroupBox2 | Appearance (Text) | Output |
| Design (Name) | GroupBox2 |
| 17 | Label1 | Appearance (Text) | function |
| Design (Name) | Label1 |
| 18 | Label2 | Appearance (Text) | X0 |
| Design (Name) | Label2 |
| 19 | Label3 | Appearance (Text) | H0 |
| Design (Name) | Label3 |
| 20 | Label4 | Appearance (Text) | tol |
| Design (Name) | Label4 |
| 21 | Label5 | Appearance (Text) | R |
| Design (Name) | Label5 |
| 22 | Label6 | Appearance (Text) | K\_max |
| Design (Name) | Label6 |
| 23 | Label7 | Appearance (Text) | X\* |
| Design (Name) | Label7 |
| 24 | Label8 | Appearance (Text) | F(x\*) |
| Design (Name) | Label8 |
| 25 | Label9 | Appearance (Text) | time |
| Design (Name) | Label9 |
| 26 | Label10 | Appearance (Text) | k |
| Design (Name) | Label10 |
| 27 | Label11 | Appearance (Text) | H1 |
| Design (Name) | Label11 |
| 28 | ProgressBar1 | Appearance (Text) |  |
| Design (Name) | ProgressBar1 |

## Практическая работа №3

## ЧАСТЬ №1

«Разработка ПО для поиска минимума ***нелинейной унимодальной функции***

на основе итерационного метода: ***Golden section method***

**Что дано:**

* ***Спецификация проблемы*** поиска минимума ***нелинейной унимодальной функции*** f(x);
* ***Спецификация*** итерационного метода (***Golden section method***) для нахождения минимума ***нелинейной унимодальной функции***
* ***Блок-схема*** Golden section method;
* ***Интерфейсная форма*** системы поиска минимума ***нелинейной унимодальной функции***, реализующей итерационный метод —***Golden section method***;
* ***Тесты*** для проверки ПО.

**Что требуется:**

* Разработать ***проект*** ПО для поиска минимума произвольной ***нелинейной унимодальной функции*** ***f(x)*** для произвольной заданной допустимой погрешности;
* Сконструировать систему, реализующую итерационный метод Golden section Method на основе использования парсера;
* Разработать ***код*** ПО для поиска минимума произвольной нелинейной функции f(x) для произвольной заданной допустимой погрешности;
* Провести валидацию системы – доказать идентичность результатов решения задач с помощью разработанного ПО заданным тестам.

## ЧАСТЬ №2

### Раздел №1

**Наименование работы –** Нахождение минимума нелинейной функции *методом золотого сечения*

### Раздел №2

**СПЕЦИФИКАЦИЯ ПРОБЛЕМЫ №2: Нахождение МИНИМУМА нелинейной функции методом золотого сечения**

* Найти минимум произвольной нелинейной функции

с заданной допустимой погрешностью ***Tolerance*** методом золотого сечения. Нелинейная функция 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.), который имеет математический смысл, и для которой существует хотя бы одно решение задачи.

### Раздел №3

**СПЕЦИФИКАЦИЯ (Описание) метода золотого сечения**

Метод относится к последовательным стратегиям и является одним из вариантов метода исключения отрезков. Алгоритм опирается на анализ значений функции в двух точках, являющихся точками золотого сечения текущего интервала неопределенности. Исключение отрезка в данном случае выполняется так же, как и в методе дихотомии. При этом с учетом свойств золотого сечения на каждой итерации, кроме первой, требуется только одно новое вычисление функции.

**Описание алгоритма решения проблемы в виде пошаговой итерационной процедуры**

Поиск минимума целевой функции по методу золотого сечения описывается следующей блок-схемой:

Предположим, что интервал  был задан. Тогда алгоритм метода золотого сечения имеет вид:

1. Вычислим две внутренние точки этого интервала:





2) Вычислим значение целевой функции в точках ck и dk: f(ck) и f(dk).

3) if 

then 

4) Если , то тогда итерационный процесс заканчивается, в противном случае итерации продолжаются

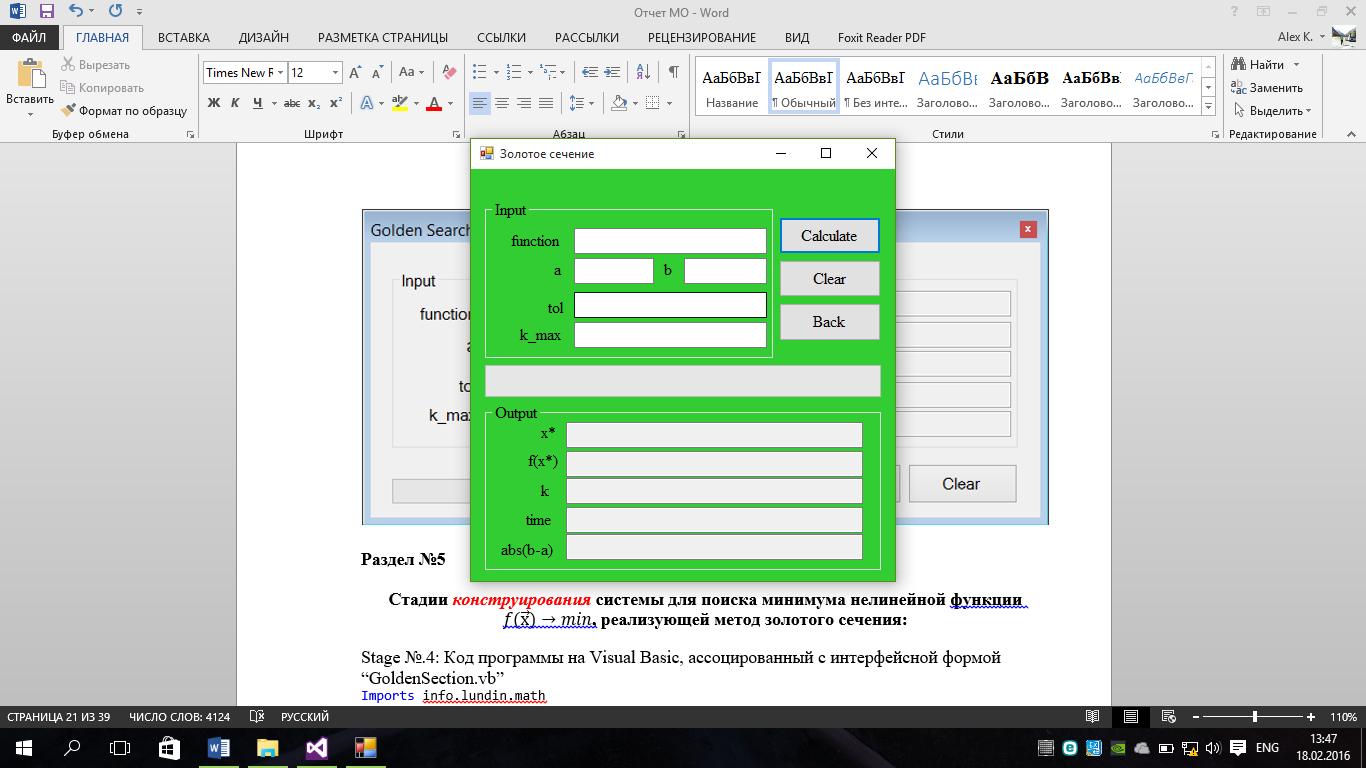
### Раздел №4

**Стадии *проектирования* системы для поиска минимума нелинейной функции , реализующей метод золотого сечения:**

Stage No.1: Разработка блок-схемы метода золотого сечения



Stage No.2: Проектирование интерфейса системы, реализующей метод золотого сечения



### Раздел №5

**Стадии *конструирования* системы для поиска минимума нелинейной функции , реализующей метод золотого сечения:**

Stage №.4: **Код программы на Visual Basic Метод золотого сечения**

Imports info.lundin.math

Imports System.Math

Public Class form3

Dim f As String

Function func(znach As Double)

Dim Parser As New ExpressionParser()

Parser.Values.Add("x", znach)

Return Parser.Parse(f)

End Function

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

If (TextBox1.Text = "" Or TextBox2.Text = "" Or TextBox3.Text = "" Or TextBox4.Text = "" Or TextBox5.Text = "") Then MsgBox("Неправильный ввод!") : GoTo 10

Dim a, b, tol, k\_max, time, k As Double

Dim x1, x2, y1, y2 As Double

f = TextBox1.Text

a = Double.Parse(TextBox2.Text)

b = Double.Parse(TextBox3.Text)

tol = Double.Parse(TextBox4.Text)

k\_max = Double.Parse(TextBox5.Text)

time = 0

k = 0

While (Abs(b - a) > tol And k < k\_max)

k = k + 1

x1 = b - (b - a) / ((1 + Sqrt(5)) / 2)

x2 = a + (b - a) / ((1 + Sqrt(5)) / 2)

y1 = func(x1)

y2 = func(x2)

If (y1 > y2) Then

a = x1

Else

b = x2

End If

time = time + 0.0001

TextBox9.Text = Convert.ToString(time)

If (ProgressBar1.Value <> 100) Then

ProgressBar1.Value += 1

End If

End While

If (ProgressBar1.Value <> 100) Then

ProgressBar1.Value = 100

End If

ProgressBar1.Value = 100

TextBox6.Text = x1

TextBox7.Text = func(x1)

TextBox8.Text = k

TextBox10.Text = Abs(b - a)

10:

End Sub

End Class

### Раздел №5

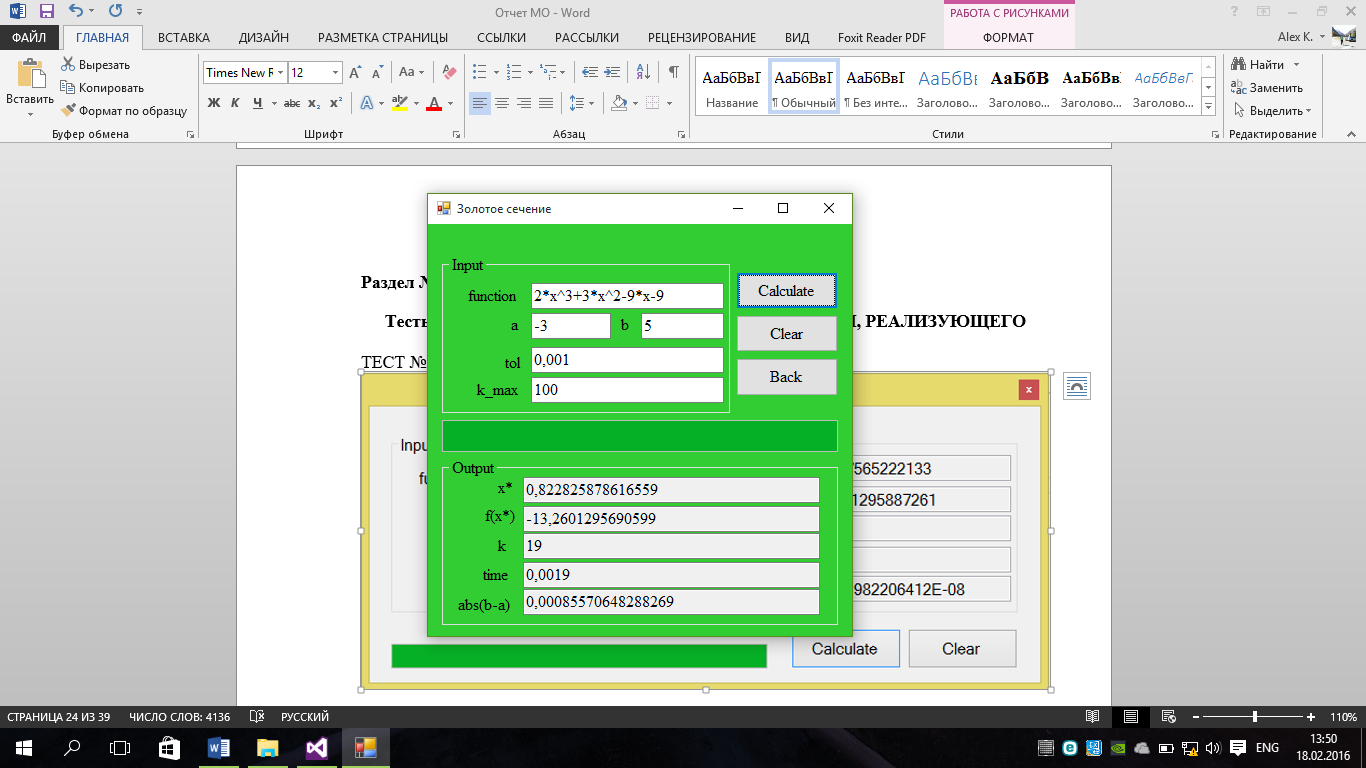
**ПРОЕКТИРОВАНИЕ По**: ДОКУМЕНТИРОВАНИЕ ЭТАПОВ ПРОЕКТИРОВАНИЯ ИнтерфейснОЙ формЫ системЫ, реализующЕЙ итерационный метод –Метод золотого сечения – ДЛЯ поиска корнЕЙ нелинейного уравнения

|  |  |  |  |
| --- | --- | --- | --- |
| **Number of control** | **Control** | **Property** | **Setting** |
| 1 | Button | Appearance (Text) | Calculate |
| Design (Name) | Button1 |
| 2 | Button | Appearance (Text) | Clear |
| Design (Name) | Button2 |
| 3 | Button | Appearance (Text) | Back |
| Design (Name) | Button3 |
| 4 | GroupBox1 | Appearance (Text) | input |
| Design (Name) | GroupBox1 |
| 5 | Textbox1 | Appearance (Text) |  |
| Design (Name) | Textbox1 |
| 6 | Textbox2 | Appearance (Text) |  |
| Design (Name) | Textbox2 |
| 7 | Textbox3 | Appearance (Text) |  |
| Design (Name) | Textbox3 |
| 8 | Textbox4 | Appearance (Text) |  |
| Design (Name) | Textbox4 |
| 9 | Textbox5 | Appearance (Text) |  |
| Design (Name) | Textbox5 |
| 10 | Textbox6 | Appearance (Text) |  |
| Design (Name) | Textbox6 |
| 11 | Textbox7 | Appearance (Text) |  |
| Design (Name) | Textbox7 |
| 12 | Textbox8 | Appearance (Text) |  |
| Design (Name) | Textbox8 |
| 13 | Textbox9 | Appearance (Text) |  |
| Design (Name) | Textbox9 |
| 14 | Textbox10 | Appearance (Text) |  |
| Design (Name) | Textbox10 |
| 15 | GroupBox2 | Appearance (Text) | Output |
| Design (Name) | GroupBox2 |
| 16 | Label1 | Appearance (Text) | function |
| Design (Name) | Label1 |
| 17 | Label2 | Appearance (Text) | a |
| Design (Name) | Label2 |
| 18 | Label3 | Appearance (Text) | b |
| Design (Name) | Label3 |
| 19 | Label4 | Appearance (Text) | tol |
| Design (Name) | Label4 |
| 20 | Label5 | Appearance (Text) | K\_max |
| Design (Name) | Label5 |
| 21 | Label7 | Appearance (Text) | X\* |
| Design (Name) | Label7 |
| 22 | Label8 | Appearance (Text) | F(x\*) |
| Design (Name) | Label8 |
| 23 | Label9 | Appearance (Text) | time |
| Design (Name) | Label9 |
| 24 | Label10 | Appearance (Text) | Abs(b-a) |
| Design (Name) | Label10 |
| 25 | Label11 | Appearance (Text) | H1 |
| Design (Name) | Label11 |

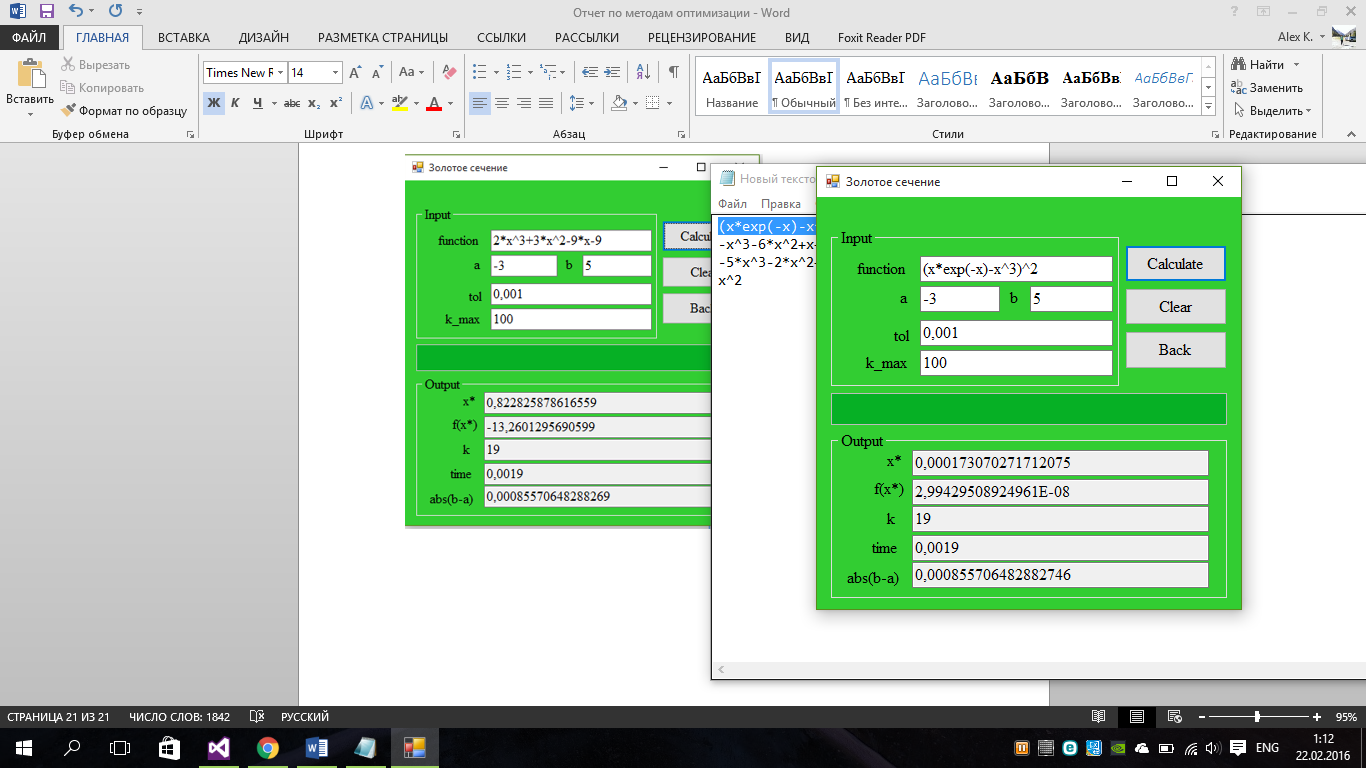
### Раздел №6

**ПРОВЕРКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ, РЕАЛИЗУЮЩЕГО МЕТОД ЗОЛОТОГО СЕЧЕНИЯ**

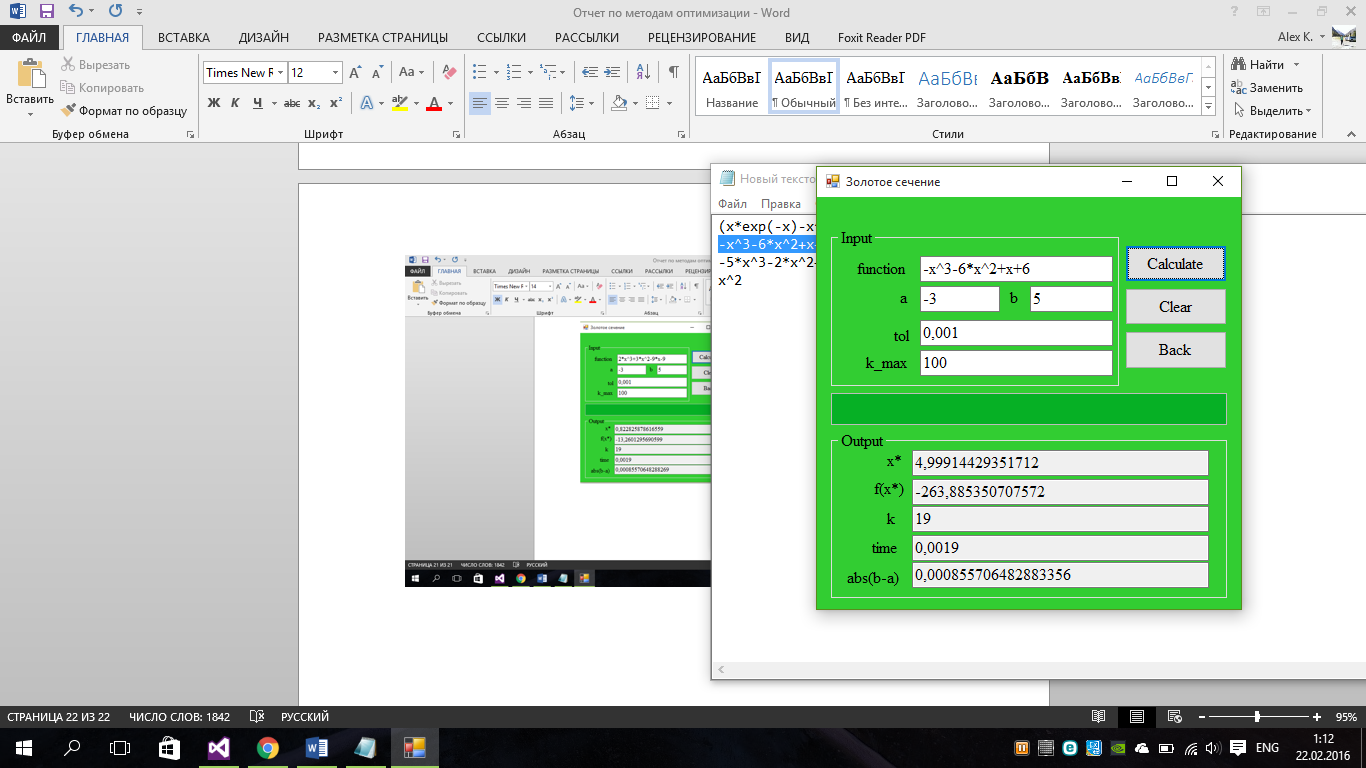
Тест №1:



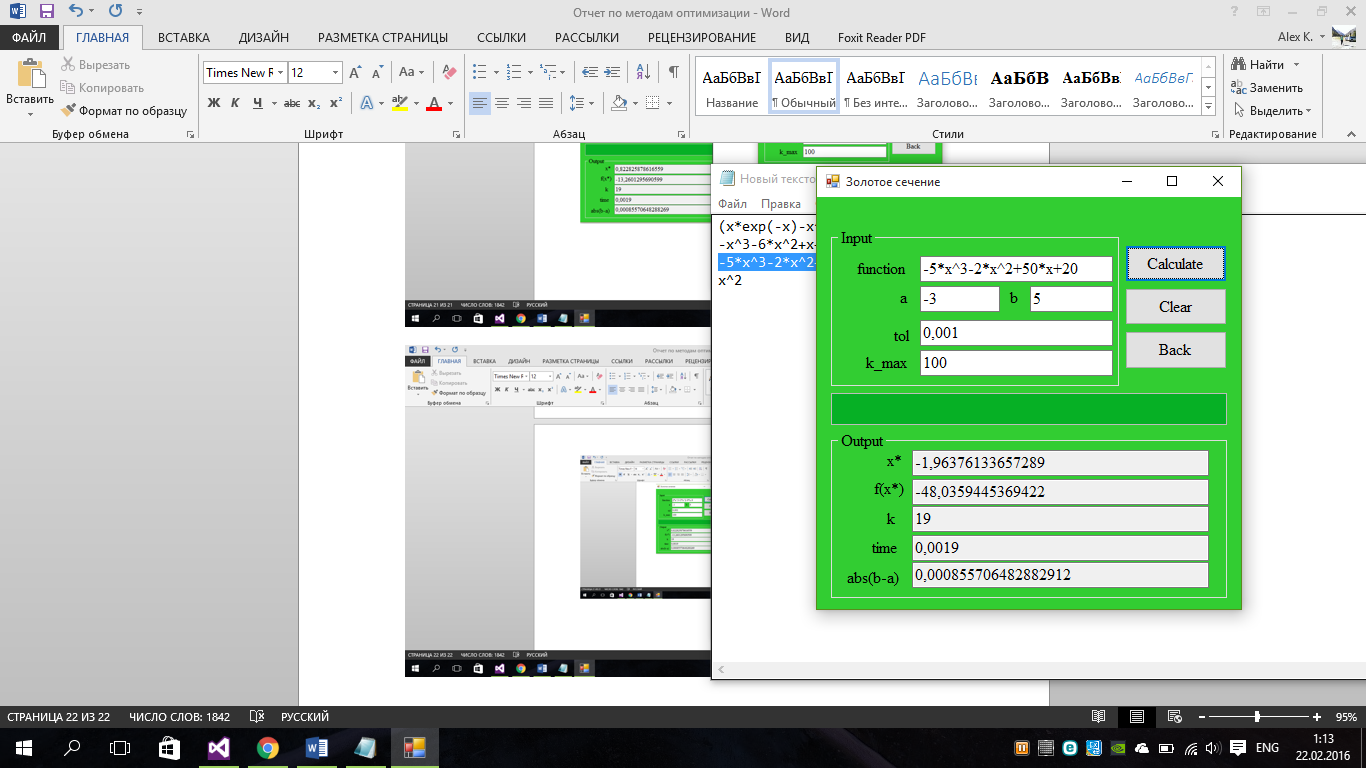
Тест №2:



Тест №3:



Тест №4:



Тест №5:

