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 по практической работе №2**

**«Разработка ПО для поиска решения одномерной задачи оптимизации**

**на основе *Even Search* method of optimization – *Метод равномерного поиска*»**

по дисциплине

«МЕТОДЫ ОПТИМИЗАЦИИ»

Выполнил: Жанболот уулу Аскабек

Группа: ПИ-2-20

Проверил: Тен И.Г.

Бишкек 2022

**Практическая работа №2**

«Разработка ПО для поиска решения одномерной задачи оптимизации

на основе ***Even Search*** method of optimization – ***Метод равномерного поиска***»

**Дано:**

* Задается аналитическое выражение для произвольной целевой функции f(x);
* Спецификация метода оптимизации (***Even Search Method***) для нахождения оптимального значения решающей переменной;
* Структура интерфейсной формы, реализующая итерационный метод — ***Even Search Method***.

**Что требуется:**

* Разработать проект ПО для поиска **решения задач оптимизации** для произвольной заданной допустимой погрешности;
* Написать ***код*** ПО для поиска **решения задач оптимизации** для произвольной заданной допустимой погрешности;
* Доказать, что найдено **оптимальное**решение с погрешностью решения не более заданной допустимой погрешности.

**Раздел №1**

**Наименование работы –** поиск решения одномерной задачи оптимизации методом Равномерного поиска (ESM)

**Раздел №2** СПЕЦИФИКАЦИЯ ПРОБЛЕМЫ №1: Поиск решения одномерной задачи оптимизации методом Равномерного поиска.

Найти решение задачи оптимизации для произвольной заданной допустимой погрешности. Метод имеет постоянный размер шага h, который не зависит от номера итерации или иначе этот метод можно назвать алгоритмом с фиксированным шагом поиска. Таким образом, метод равномерного поиска генерирует стационарный одношаговый итерационный процесс.

**Раздел №3**

**Спецификация(описание) метода**

To solve the one-dimensional optimization problem 𝐹𝑖𝑛𝑑 𝑒𝑥𝑡𝑟 𝑓(𝑥)𝑓𝑜𝑟 𝑥 ∈ 𝑅 1 using Even Search Method and given an initial approximation x(0) of the searching variable value which obey the inequality x(0) < x\*, and given step size H which obeys the inequality H < tol , where x\* – is an unknown solution of the optimization problem; H – is a generic parameter of the algorithm; tol – is a required error tolerance.

**Initial input:** a function 𝑓(𝑥), an initial approximation x(0) which obey the inequality x(0) < x\*, step size H and an error tolerance 𝑡𝑜𝑙.

**1.1: Mathematical description of the algorithm:  
Case (i):** IF f(x(k))>=f(x(k-1)) THEN x(k+1)=x(k); ELSE x(k+1)=x(k)+H; For all k=0,1,2,…;

Where k – is a number of iteration; x(k) – is an approximate value of the searching variable x on the k-th iteration.

**Note:** It is the algorithm of the Even Search Method to find a **minimum** of a given objective function.

Figure 1a: Flow-chart of the Even Search Method (find minimum) is shown below.

**Case (ii):** IF f(x(k)) <=f(x(k-1)) THEN x(k+1) =x(k); ELSE x(k+1) =x(k)+H; For all k=0,1,2…,

Where k – is a number of iteration; x(k) – is an approximate value of the searching variable x on the k-th iteration.

**Note:** It is the algorithm of the Even Search Method to find a **maximum** of a given objective function.

Figure 1b: Flow-chart of the Even Search Method (find maximum) is shown below.

**1.2 Computational description of the algorithm:**

Input {f(x); X0; Epsilon; H; k\_max}

YF0 = f(X0);

Time\_0 = Timer;

X1 = X0 + H;

YF1 = f(X1);

k = 0;

Cond = 0;

k\_max0 = k\_max;

CondKmax = 0;

DO

K = k + 1;

IF k = k\_max0 Then

Increase k\_max?

Reply = Yes/No;

IF Reply = Yes Then

K\_max0 = k\_max0 + k\_max;

ELSE

CondKmax = 1;

END IF

END IF

IF YF1 <= YF0:

IF k = 1:

Cond = 1;

END IF

X1 = X0;

YF1 = YF0;

ELSE:

X0 = X1;

YF0 = YF1;

X1 = X0 + H;

YF1 = f(X1);

END IF

ElapsedTime = Time – Timer\_0

WHILE Cond = 0 & Abs(X1 – X0) > Epsilon & CondMax = 0

Output {X1; YF1; Abs(X1 – X0); k; ElapsedTime; Cond}

**Note:** It is the algorithm of the Even Search Method to find a **minimum** of a given objective function.

Input {f(x); X0; Epsilon; H; k\_max}

YF0 = f(X0);

Time\_0 = Timer;

X1 = X0 + H;

YF1 = f(X1);

k = 0;

Cond = 0;

k\_max0 = k\_max;

CondKmax = 0;

DO

K = k + 1;

IF k = k\_max0 Then

Increase k\_max?

Reply = Yes/No;

IF Reply = Yes Then

K\_max0 = k\_max0 + k\_max;

ELSE

CondKmax = 1;

END IF

END IF

IF YF1 >= YF0:

IF k = 1:

Cond = 1;

END IF

X1 = X0;

YF1 = YF0;

ELSE:

X0 = X1;

YF0 = YF1;

X1 = X0 + H;

YF1 = f(X1);

END IF

ElapsedTime = Time – Timer\_0

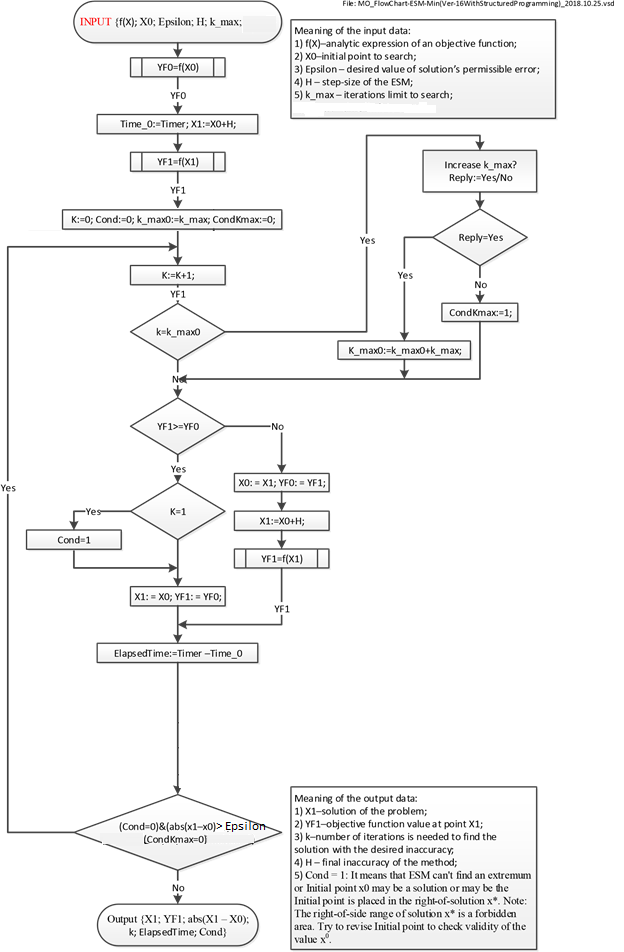
WHILE Cond = 0 & Abs(X1 – X0) > Epsilon & CondMax = 0

Output {X1; YF1; Abs(X1 – X0); k; ElapsedTime; Cond}

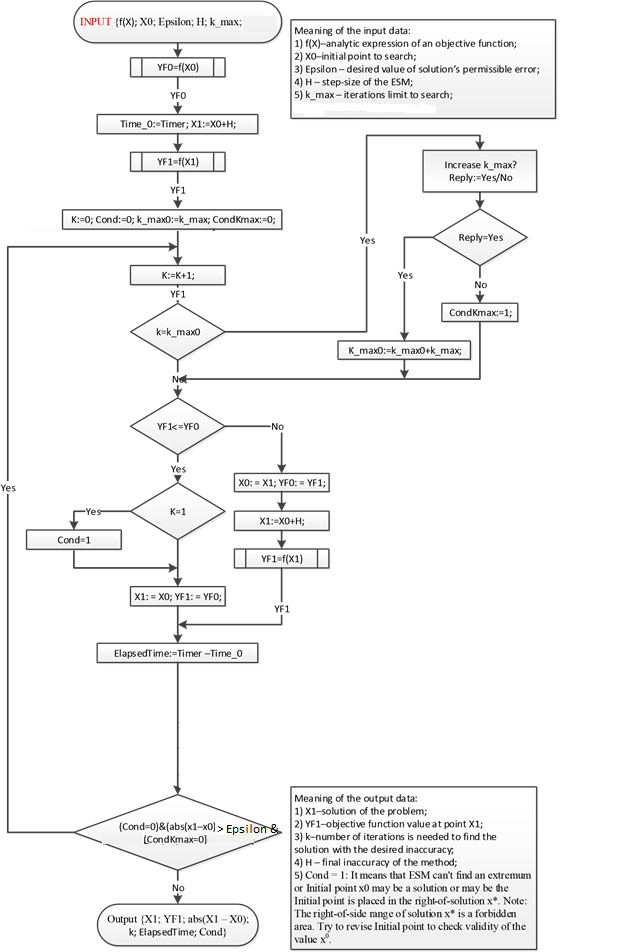
**Note:** It is the algorithm of the Even Search Method to find a **maximum** of a given objective function.

**Раздел № 4 Стадия проектирования системы для поиска решения одномерной задачи оптимизации метод Равномерного поиска**

**Flow-chart of the EVEN SEARCH METHOD to find out minimum of an objective function**

****

**Flow-chart of the EVEN SEARCH METHOD to find out maximum of an objective function**



Функциональные требования:

1) Программа должна производить поиск экстремума произвольной нелинейной функции, с заданной погрешностью.

2) В программе должно быть предоставлено доказательство валидности найденного решения.

3) Программа должна определять траекторию к определенному excel-file “MO\_LookingForOnePoint.xlsx”.

4) Программа должна открывать лист “Russian” excel файла “MO\_LookingForOnePoint.xlsx”, и вставлять целевую функцию, левую и правую границы в определенные ячейки листа.

5) Программа должна вычислять значение целевой функции f(x), для любого значения аргумента x.

6) В программе должна производиться проверка валидности введеных данных.

7) В программе должна быть реализована EvenSearchMethod для поиска экстремума функции.

8) В программе при вводе начальной точкой которая является экстремиуом или которая находится справа от экстремума, должно выводится сообщение и предложение открыть excel файл для просмотра графика.

**Раздел №5**

**ПРОЕКТИРОВАНИЕ По**: Документирование этапов проектирования интерфейсной формы системы, реализующей метод равномерного поиска – для поиска решения одномерной задачи оптимизации

Stage No.3: Документирование процесса задания свойств элементов интерфейсной формы системы, выбранной для внедрения и реализующей Even Search Method

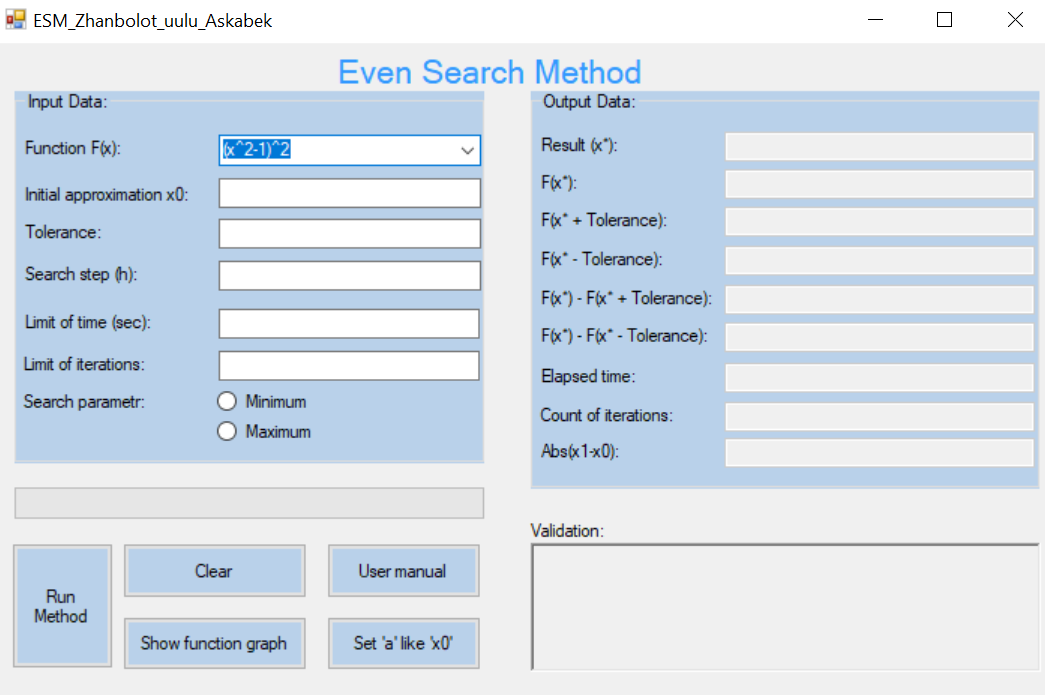


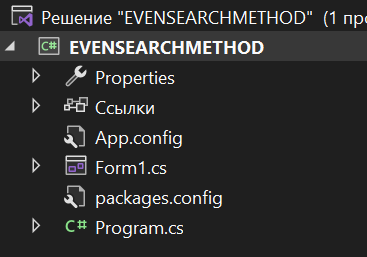
Table 2: Settings for control properties of the software system implementing Even Search method (the table is intended to document the activity of a student to settings control properties of the interface form – the table is ***mandatory***)

|  |  |  |  |
| --- | --- | --- | --- |
| **Number of control** | **Control** | **Property** | **Setting** |
| 1 | Label1 | Appearance (Text) | Pocket Search Method |
| Design (Name) | label1 |
| 2 | GroupBox1 | Appearance (Text) | Input Data |
| Design (Name) | groupBox1 |
| 3 | Label2 | Appearance (Text) | Function F(x) |
| Design (Name) | label2 |
| 4 | ComboBox1 | Appearance (Text) | (x^2-1)^2 |
| The rest (Items) | Collection |
| Items (Collection) | (x-1)\*(x-2)^2  x^2-4\*sin(x)  x^3-5\*x^2+x+5  (x-2)^2-ln(x)  (x-2)^2-log(x) |
| Design (Name) | Function |
| 5 | Label3 | Appearance (Text) | Initial approximation x0 |
| Design (Name) | label3 |
| 6 | TextBox1 | Appearance (Text) |  |
| Design (Name) | InitialApproximation |
| 7 | Label4 | Appearance (Text) | Tolerance |
| Design (Name) | label4 |
| 8 | TextBox2 | Appearance (Text) |  |
| Design (Name) | Tolerance |
| 9 | Label5 | Appearance (Text) | Limit of time: |
| Design (Name) | label6 |
| 10 | TextBox3 | Appearance (Text) |  |
| Design (Name) | limitOfTime |
| 11 | Label6 | Appearance (Text) | Limit of iterations: |
| Design (Name) | label5 |
| 12 | TextBox4 | Appearance (Text) |  |
| Design (Name) | limitOfIterations |
| 13 | ProgressBar1 | Behavior (Visible) | true |
| Design (Name) | progressBar1 |
| 14 | GroupBox2 | Appearance (Text) | Output Data |
| Design (Name) | groupBox2 |
| 15 | Button1 | Appearance (Text) | User manual |
| Design (Name) | button5 |
| 16 | RadioButton1 | Appearance (Text) | minimum |
| Design (Name) | Minimum |
| 17 | RadioButton2 | Appearance (Text) | maximum |
| Design (Name) | Maximum |
| 18 | Button2 | Appearance (Text) | Run Method |
| Design (Name) | button1 |
| 19 | Button3 | Appearance (Text) | Show function graph |
| Design (Name) | FunctionGraph |
| 20 | Button4 | Appearance (Text) | Clear |
| Design (Name) | button3 |
| 21 | Button5 | Appearance (Text) | Set ‘a’ like ‘x0’ |
| Design (Name) | button2 |
| 22 | GroupBox3 | Appearance (Text) | Output Data |
| Design (Name) | groupBox2 |
| 23 | Label7 | Appearance (Text) | Result (x\*): |
| Design (Name) | label12 |
| 24 | TextBox5/ Behavior  (ReadOnly) | ReadOnly | true |
| Design (Name) | ResultX |
| 25 | Label8 | Appearance (Text) | Elapsed time: |
| Design (Name) | label9 |
| 26 | TextBox6/ Behavior  (ReadOnly) | ReadOnly | true |
| Design (Name) | elapsedTime |
| 27 | Label9 | Appearance (Text) | Count of iterations: |
| Design (Name) | label10 |
| 28 | TextBox7/ Behavior  (ReadOnly) | ReadOnly | true |
| Design (Name) | countofiterations |
| 29 | Label10 | Appearance (Text) | F(x\*): |
| Design (Name) | label13 |
| 30 | TextBox8/ Behavior  (ReadOnly) | ReadOnly | true |
| Design (Name) | fx |
| 31 | Label11 | Appearance (Text) | F(X\*+Tolerance): |
| Design (Name) | label16 |
| 32 | TextBox9/ Behavior  (ReadOnly) | ReadOnly | true |
| Design (Name) | fxplustolerance |
| 33 | Labe112 | Appearance (Text) | F(X\*-Tolerance): |
| Design (Name) | label15 |
| 34 | TextBox10/ Behavior  (ReadOnly) | ReadOnly | true |
| Design (Name) | fxminustolerance |
| 35 | Label13 | Appearance (Text) | Validation |
| Design (Name) | label17 |
| 36 | RichTextBox1/ Behavior  (ReadOnly) | Appearance  (Text) |  |
| 38 | Label14 | Appearance (Text) | F(X\*)-F(X\*+Tolerance) |
| Design (Name) | label11 |
| 39 | TextBox13/ Behavior  (ReadOnly) | ReadOnly | true |
|  |  | Design (Name) | fxminusplustolerance |
| 40 | Label15 | Appearance (Text) | F(X\*)-F(X\*-Tolerance) |
|  |  | Design (Name) | label8 |
| 41 | TextBox12/ Behavior  (ReadOnly) | ReadOnly | true |
|  |  | Design (Name) | fxminusminustolerance |
| 42 | TextBox13/Behavior  (ReadOnly) | ReadOnly | true |
|  |  | Design (Name) | absError |
| 43 | Label16 | Appearance (Text) | Abs(x1-x0): |
|  |  | Design (Name) | label21 |
| 44 | TextBox14/Behavior  (ReadOnly) | ReadOnly | true |
|  |  | Design (Name) | resultSearchStep |
| 45 | Label17 | Appearance (Text) | Search step(k): |
|  |  | Design (Name) | label20 |
| 46 | Label18 | Appearance (Text) | Search step (h): |
|  |  | Design (Name) | label7 |
| 47 | TextBox15 | Appearance (Text) |  |
|  |  | Design (Name) | SearchStep |

**Раздел №6**

**Стадии *конструирования* программного обеспечения для поиска решения одномерной задачи оптимизации, реализующей Even Search Method:**

Stage №.4: Код программы на C# Windows Forms(.NET Framework), ***ассоцированный с интерфейсной формой*** “Form1.cs”, который ***реализует функции ввода и вывода данных***, ***реализует логику*** Pocket Search метода и составляет Public Class “Form1”



using System;

using System.Text;

using Microsoft.Office.Interop.Excel;

using System.Windows.Forms;

using Application = Microsoft.Office.Interop.Excel.Application;

using TextBox = System.Windows.Forms.TextBox;

using MessageBox = System.Windows.Forms.MessageBox;

using aziretParser;

using System.Diagnostics;

using System.Drawing;

namespace POCKETSEARCHMETHOD

{

public partial class Form1 : Form

{

private const string nameOfExcel = @"\Zhanbolot\_uulu\_Askabek\_LookingForOnePoint.xlsm";

string inputFuncFX = "";

decimal x0 = 0;

decimal x1 = 0;

decimal f0;

decimal f1;

decimal e\_tol = 0;

decimal h = 0;

int k\_max = 0;

decimal t\_max = 0;

bool b = false;

Application xls;

Workbook book = null;

Worksheet sheet = null;

public Form1()

{

InitializeComponent();

xls = new Application();

}

public int getSign(decimal number)

{

if (number < 0)

{

return -1;

}

else

{

return 1;

}

}

public void OpenExcel()

{

if (!checkFunction(1)) return;

string function;

decimal startPoint;

try

{

if (book == null)

{

book = xls.Workbooks.Open(System.IO.Directory.GetCurrentDirectory() + nameOfExcel);

}

if (sheet == null)

{

sheet = book.Sheets["Russian"];

sheet.Activate();

}

xls.Visible = true;

function = Function.Text;

if (InitialApproximation.Text != "" && InitialApproximation.Text != "-" && InitialApproximation.Text != "+" && InitialApproximation.Text != ".")

{

startPoint = Decimal.Parse(InitialApproximation.Text);

}

else

{

startPoint = 1;

}

sheet.Cells[4, 9] = startPoint;

sheet.Cells[2, 1] = "f(x)=" + Function.Text;

}

catch

{

book = xls.Workbooks.Open(System.IO.Directory.GetCurrentDirectory() + nameOfExcel);

sheet = book.Sheets["Russian"];

sheet.Activate();

xls.Visible = true;

function = Function.Text;

if (InitialApproximation.Text != "" && InitialApproximation.Text != "-" && InitialApproximation.Text != "+" && InitialApproximation.Text != ".")

{

startPoint = Decimal.Parse(InitialApproximation.Text);

}

else

{

startPoint = 1;

}

sheet.Cells[4, 9] = startPoint;

sheet.Cells[2, 1] = "f(x)=" + Function.Text;

}

StringBuilder builder = new StringBuilder(function);

builder.Replace("exp", ":");

builder.Replace("x", "D4");

builder.Replace(":", "exp");

function = builder.ToString();

sheet.Range["E4:E10003"].Value = "=" + function;

}

private bool parseTry(TextBox t, String type)

{

try

{

if (type == "Decimal")

Decimal.Parse(t.Text, System.Globalization.NumberStyles.Float);

else if (type == "Integer")

int.Parse(t.Text);

return true;

}

catch

{

return false;

}

}

private void Clean(Control control)

{

foreach (var element in control.Controls)

{

switch (element.GetType().Name)

{

case "TextBox":

((TextBox)element).Text = String.Empty;

break;

case "RadioButton":

((RadioButton)element).Checked = false;

break;

case "RichTextBox":

((RichTextBox)element).Text = String.Empty;

break;

case "ProgressBar":

((System.Windows.Forms.ProgressBar)element).Value = 0;

break;

case "GroupBox":

Clean((Control)element);

break;

default:

break;

}

}

}

private bool IsOKForDecimalTextBox(char theCharacter, TextBox theTextBox, bool positive)

{

if (!char.IsControl(theCharacter) && !char.IsDigit(theCharacter) && (theCharacter != ',')

&& (theCharacter != '-') && (theCharacter != '+') && (theCharacter != 'E') && (theCharacter != 'e'))

{

return false;

}

if(positive && theCharacter == '-' && (theTextBox.Text.IndexOf('E') == -1 && theTextBox.Text.IndexOf('e') == -1))

{

return false;

}

if (theCharacter == ',' && theTextBox.Text.IndexOf(',') > -1)

{

return false;

}

if (theCharacter == 'e' && (theTextBox.Text.IndexOf('e') > -1 || theTextBox.Text.IndexOf('E') > -1))

{

return false;

}

if (theCharacter == 'E' && (theTextBox.Text.IndexOf('E') > -1 || theTextBox.Text.IndexOf('e') > -1))

{

return false;

}

if (theCharacter == '-' && (theTextBox.Text.IndexOf('-') > -1 || theTextBox.Text.IndexOf('+') > -1))

{

return false;

}

if (theCharacter == '+' && (theTextBox.Text.IndexOf('+') > -1 || theTextBox.Text.IndexOf('-') > -1))

{

return false;

}

if (((theCharacter == '-') || (theCharacter == '+')) && (theTextBox.SelectionStart != 0 && (theTextBox.Text.IndexOf('E') == -1 && theTextBox.Text.IndexOf('e') == -1)))

{

return false;

}

if ((char.IsDigit(theCharacter) || (theCharacter == ',')) && ((theTextBox.Text.IndexOf('-') > -1)

|| (theTextBox.Text.IndexOf('+') > -1)) && theTextBox.SelectionStart == 0)

{

return false;

}

return true;

}

public decimal F(decimal x)

{

decimal result;

result = aziretParser.Computer.Compute(inputFuncFX, x);

return result;

}

private void button4\_Click(object sender, EventArgs e)

{

OpenExcel();

}

private void button3\_Click(object sender, EventArgs e)

{

Clean(this);

}

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

{

if (e.KeyChar == '.')

{

e.KeyChar = ',';

}

e.Handled = !IsOKForDecimalTextBox(e.KeyChar, InitialApproximation, false);

}

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

{

if (e.KeyChar == '.')

{

e.KeyChar = ',';

}

e.Handled = !IsOKForDecimalTextBox(e.KeyChar, Tolerance, true);

}

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

{

if (e.KeyChar == '.')

{

e.KeyChar = ',';

}

e.Handled = !IsOKForDecimalTextBox(e.KeyChar, SearchStep, true);

}

private String checkParse()

{

String errorMessage = "";

if (!parseTry(InitialApproximation, "Decimal"))

{

errorMessage += "Invalid value of the field x0 (the starting point of the approximation)! Change the input and perform the calculation!\n\n";

}

else

{

x0 = Decimal.Parse(InitialApproximation.Text, System.Globalization.NumberStyles.Float);

}

if (!parseTry(SearchStep, "Decimal"))

{

errorMessage += "Invalid value of the field search step! Change the input and perform the calculation!\n\n";

}

else

{

h = Decimal.Parse(SearchStep.Text, System.Globalization.NumberStyles.Float);

}

if (parseTry(Tolerance, "Decimal"))

{

e\_tol = Decimal.Parse(Tolerance.Text, System.Globalization.NumberStyles.Float);

}

else

{

errorMessage += "Invalid value of the Tolerance(e) field (entered tolerance)! Change the input and perform the calculation!\n\n";

}

if (!parseTry(LimitOfIterations, "Integer"))

{

errorMessage += "Invalid value of the field limit of iterations! Change the input and perform the calculation!\n\n";

}

else

{

k\_max = Int32.Parse(LimitOfIterations.Text);

}

if (!parseTry(LimitOfTime, "Decimal"))

{

errorMessage += "Invalid value of the field limit of time! Change the input and perform the calculation!\n\n";

}

else

{

t\_max = Decimal.Parse(LimitOfTime.Text, System.Globalization.NumberStyles.Float);

}

return errorMessage;

}

public bool fullCheck()

{

bool check = false;

if (Function.Text == "" || InitialApproximation.Text == "" ||

Tolerance.Text == "" || LimitOfIterations.Text == "" ||

LimitOfTime.Text == "" || SearchStep.Text == "")

{

MessageBox.Show("All fields must be filled in! Enter the missing information and make the calculation!", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

else

{

if (checkParse() != "")

{

MessageBox.Show(checkParse(), "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

else

{

if (isRigth() && checkFunction(x0))

{

check = true;

}

}

}

return check;

}

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

{

if (e.KeyChar == '.')

{

e.KeyChar = ',';

}

e.Handled = !IsOKForDecimalTextBox(e.KeyChar, LimitOfTime, true);

}

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

{

if ((int)e.KeyChar == (int)48 && LimitOfIterations.Text == "")

{

e.Handled = true;

return;

}

e.Handled = !char.IsDigit(e.KeyChar) && !char.IsControl(e.KeyChar);

}

public string getComparisonSign(decimal a, decimal b)

{

if (a > b)

{

return ">";

}

else if (a < b)

{

return "<";

}

else

{

return "=";

}

}

private bool isRigth()

{

bool valid = true;

if (e\_tol <= 0)

{

MessageBox.Show("The value of the tolerance field must be greater than 0! Change the input and perform the calculation!", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

valid = false;

}

if(h <= 0)

{

MessageBox.Show("The value of the search step field must be greater than 0! Change the input and perform the calculation!", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

valid = false;

}

if(h > e\_tol)

{

MessageBox.Show("The value of the search step field must be equal or less than tolerance field! Change input and perform the calculation!", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

valid = false;

}

if (k\_max <= 0)

{

MessageBox.Show("The value of the limit of iterations field must be greater than 0! Change the input and perform the calculation!", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

valid = false;

}

if (t\_max <= 0)

{

MessageBox.Show("The value of the limit of time field must be greater than 0! Change the input and perform the calculation!", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

valid = false;

}

if (!(Maximum.Checked || Minimum.Checked))

{

MessageBox.Show("Please select search option maximum or minimum.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

valid = false;

}

if (valid)

{

return true;

}

return false;

}

private bool checkFunction(decimal x0)

{

inputFuncFX = Function.Text;

if (inputFuncFX == "" || inputFuncFX.IndexOf('x') == -1)

{

MessageBox.Show("The function is entered incorrectly! Change the input and perform the calculation!", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

Clean(this);

return false;

}

try

{

if (inputFuncFX.Contains("log") && x0 <= 0 || inputFuncFX.Contains("ln") && x0 <= 0)

{

MessageBox.Show("If you entered function with 'log' or 'ln' value of X0 must greater than zero!", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

return false;

}

else

{

decimal F1 = F(x0);

return true;

}

}

catch

{

MessageBox.Show("The function or initial approximation is entered incorrectly! Change the input and perform the calculation!", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

Clean(this);

return false;

}

}

public bool MaxOrMin(decimal f0, decimal f1)

{

if (Maximum.Checked)

{

return f0 >= f1;

}

return f0 <= f1;

}

private void button5\_Click(object sender, EventArgs e)

{

DialogResult result = MessageBox.Show("1) Choose a function or write your's on field 'Function'\n" +

"2) Click on the button 'Show function graph'\n" +

"3) In the opened file select the values for a or b,\n" +

"then save the document and return to the program\n" +

"4) If you need 'a' value to insert,\n" +

"click the button 'Set 'a' like 'X0'' or write your's\n" +

"5) Enter tolerance\n" +

"6) Enter search step\n" +

"8) Enter limit of time in sec\n" +

"9) Enter limit of iterations \n" +

"10) Select search parameter\n" +

"Then click the button 'Run Method'.", "Information",

MessageBoxButtons.OK, MessageBoxIcon.Information);

}

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

{

try

{

if (book == null)

{

book = xls.Workbooks.Open(System.IO.Directory.GetCurrentDirectory() + nameOfExcel);

}

if (sheet == null)

{

sheet = book.Sheets["Russian"];

sheet.Activate();

}

book.Save();

InitialApproximation.Text = sheet.Cells[4, 9].Value.ToString();

}

catch

{

book = xls.Workbooks.Open(System.IO.Directory.GetCurrentDirectory() + nameOfExcel);

sheet = book.Sheets["Russian"];

sheet.Activate();

book.Save();

InitialApproximation.Text = sheet.Cells[4, 9].Value.ToString();

}

xls.Visible = false;

book = null;

sheet = null;

}

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

{

inputFuncFX = "";

x0 = 0;

x1 = 0;

f0 = 0;

f1 = 0;

e\_tol = 0;

k\_max = 0;

t\_max = 0;

string extremium;

if (fullCheck())

{

xls.Visible = false;

book = null;

sheet = null;

Stopwatch stopwatch = new Stopwatch();

stopwatch.Start();

Clean(groupBox2);

validation.Text = String.Empty;

progressBar1.Value = 0;

decimal fplusTol;

decimal fminusTol;

f0 = F(x0);

x1 = x0 + e\_tol;

f1 = F(x1);

int k = 0;

if (Maximum.Checked)

{

extremium = "maximizer";

}

else

{

extremium = "minimizer";

}

progressBar1.Value = 0;

while (true)

{

k = k + 1;

progressBar1.Visible = true;

progressBar1.Maximum = (int)(k + 0.00000001);

progressBar1.Value = k;

if (k > k\_max)

{

stopwatch.Stop();

f1 = F(x1);

fminusTol = F(x1 - e\_tol);

fplusTol = F(x1 + e\_tol);

DialogResult result = MessageBox.Show("Iteration limit reached. Do you want to add iterations?",

"Information", MessageBoxButtons.YesNo, MessageBoxIcon.Information);

if (result == DialogResult.Yes)

{

k\_max += k\_max;

LimitOfIterations.Text = k\_max.ToString();

}

else

{

k--;

validation.Text += "Result X\* not found because of limit of iterations = " + k\_max + "." +

"\nSince the following condition is false, namely:" +

"\nSign(f(X\*)-f(X\*+Tolerance)) = " + getSign(f1 - fplusTol) + " and Sign(f(X\*)-f(X\*-Tolerance)) = " + getSign(f1 - fminusTol) + "!" +

"\nResult X\* is not " + extremium + " of the function.";

validation.ForeColor = Color.Red;

FillResult(x1.ToString("F28"), k.ToString(), getError(Tolerance, Math.Abs(x1 - x0)), fminusTol.ToString("F28"), fplusTol.ToString("F28"), f1.ToString("F28"), (f1 - fplusTol).ToString("F28"), (f1 - fminusTol).ToString("F28"));

absError.Text = getError(Tolerance, Math.Abs(x1 - x0));

DialogResult answer = MessageBox.Show("Result X\* not found because of maximum limit of iterations = " + k\_max + "." +

"\nSince the following condition is false, namely:" +

"\nSign(f(X\*)-f(X\*+Tolerance)) = " + getSign(f1 - fplusTol) + " and Sign(f(X\*)-f(X\*-Tolerance)) = " + getSign(f1 - fminusTol) + "!" +

"\nResult X\* is not " + extremium + " of the function." +

"\n\nYou probably entered the values of 'a' and 'b' range incorrectly on Ecxel!" +

"\nSince the program is looking for an extremum only in the range 'a' and 'b'." +

"\nYou need to open the graph and select the correct points [a;b]!" +

"\n\nDo you want to open file?", "Error", MessageBoxButtons.YesNo, MessageBoxIcon.Error);

if (answer == DialogResult.Yes)

{

OpenExcel();

}

break;

}

stopwatch.Start();

}

if (stopwatch.ElapsedMilliseconds >= t\_max \* 1000)

{

stopwatch.Stop();

f1 = F(x1);

fminusTol = F(x1 - e\_tol);

fplusTol = F(x1 + e\_tol);

DialogResult result = MessageBox.Show("Time limit reached. Do you want to add time?",

"Information", MessageBoxButtons.YesNo, MessageBoxIcon.Information);

if (result == DialogResult.Yes)

{

t\_max += t\_max;

LimitOfTime.Text = t\_max.ToString();

}

else

{

validation.Text += "Result X\* not found because of limit of time = " + t\_max + " sec." +

"\nSince the following condition is false, namely:" +

"\nSign(f(X\*)-f(X\*+Tolerance)) = " + getSign(f1 - fplusTol) + " and Sign(f(X\*)-f(X\*-Tolerance)) = " + getSign(f1 - fminusTol) + "!" +

"\nResult X\* is not " + extremium + " of the function.";

validation.ForeColor = Color.Red;

FillResult(x1.ToString("F28"), k.ToString(), getError(Tolerance, Math.Abs(x1 - x0)), fminusTol.ToString("F28"), fplusTol.ToString("F28"), f1.ToString("F28"), (f1 - fplusTol).ToString("F28"), (f1 - fminusTol).ToString("F28"));

absError.Text = getError(Tolerance, Math.Abs(x1 - x0));

DialogResult answer = MessageBox.Show("Result X\* not found because of maximum time limit = " + t\_max + " sec." +

"\nSince the following condition is false, namely:" +

"\nSign(f(X\*)-f(X\*+Tolerance)) = " + getSign(f1 - fplusTol) + " and Sign(f(X\*)-f(X\*-Tolerance)) = " + getSign(f1 - fminusTol) + "!" +

"\nResult X\* is not " + extremium + " of the function." +

"\n\nYou probably entered the values of 'a' and 'b' range incorrectly on Ecxel!" +

"\nSince the program is looking for an extremum only in the range 'a' and 'b'." +

"\nYou need to open the graph and select the correct points [a;b]!" +

"\n\nDo you want to open file?", "Error", MessageBoxButtons.YesNo, MessageBoxIcon.Error);

if (answer == DialogResult.Yes)

{

OpenExcel();

}

break;

}

stopwatch.Start();

}

if (MaxOrMin(f0, f1))

{

x1 = x0;

f1 = f0;

Clean(this);

validation.Text += "The program did not run a single iteration!";

validation.ForeColor = Color.Red;

DialogResult result = MessageBox.Show("The program did not run a single iteration. The starting point is " + extremium + " of function or is to the right of it or search step (h) is too small. You need to open the graph and select the correct point! \n\nDo you want to open file?", "Error", MessageBoxButtons.YesNo, MessageBoxIcon.Error);

if (result == DialogResult.Yes)

{

OpenExcel();

}

return;

}

else

{

x0 = x1;

f0 = f1;

x1 = x0 + h;

f1 = F(x1);

}

fminusTol = F(x1 - e\_tol);

fplusTol = F(x1 + e\_tol);

if (Math.Abs(x1 - x0) != 0)

{

if(extremium == "minimizer")

{

if ((f1 <= fminusTol && f1 <= fplusTol))

{

FillResult(x1.ToString("F28"), k.ToString(), getError(Tolerance, Math.Abs(x1 - x0)), fminusTol.ToString("F28"), fplusTol.ToString("F28"), f1.ToString("F28"), (f1 - fplusTol).ToString("F28"), (f1 - fminusTol).ToString("F28"));

validation.Text += "Since the following condition is true, namely:" +

"\nSign(f(X\*)-f(X\*+Tolerance)) = " + getSign(f1 - fplusTol) + " and Sign(f(X\*)-f(X\*-Tolerance)) = " + getSign(f1 - fminusTol) + "!" +

"\nResult X\* is " + extremium + " of the function. It has been found with the error = " + getError(Tolerance, Math.Abs(x1 - x0)) + ". This is less than or equal to given Tolerance!";

validation.ForeColor = Color.Green;

break;

}

}

else

{

if (f1 >= fminusTol && f1 >= fplusTol)

{

FillResult(x1.ToString("F28"), k.ToString(), getError(Tolerance, Math.Abs(x1 - x0)), fminusTol.ToString("F28"), fplusTol.ToString("F28"), f1.ToString("F28"), (f1 - fplusTol).ToString("F28"), (f1 - fminusTol).ToString("F28"));

validation.Text += "Since the following condition is true, namely:" +

"\nSign(f(X\*)-f(X\*+Tolerance)) = " + getSign(f1 - fplusTol) + " and Sign(f(X\*)-f(X\*-Tolerance)) = " + getSign(f1 - fminusTol) + "!" +

"\nResult X\* is " + extremium + " of the function. It has been found with the error = " + getError(Tolerance, Math.Abs(x1 - x0)) + ". This is less than or equal to given Tolerance!";

validation.ForeColor = Color.Green;

break;

}

}

}

}

stopwatch.Stop();

elapsedtime.Text = stopwatch.ElapsedMilliseconds / 1000.0 + " sec";

timer1.Enabled = true;

timer1.Start();

}

}

public void FillResult(string solution, string iterations, string resultTolerance, string fminustol, string fplustol, string fxvalue, string fminusplus, string fminusminus)

{

ResultX.Text = solution;

countofiterations.Text = iterations;

fxplustolerance.Text = fplustol;

fxminustolerance.Text = fminustol;

fxminusplustolerance.Text = fminusplus;

fxminusminustolerance.Text = fminusminus;

fx.Text = fxvalue;

absError.Text = resultTolerance;

}

public string getError(TextBox tol, decimal error)

{

Console.WriteLine(tol);

if (tol.Text.Contains("E"))

{

return error.ToString("0E0");

}

else if (tol.Text.Contains("e"))

{

return error.ToString("0e0");

}

else

{

return error.ToString();

}

}

private void timer1\_Tick(object sender, EventArgs e)

{

progressBar1.Value = 0;

timer1.Enabled = false;

timer1.Stop();

}

private void Function\_TextChanged(object sender, EventArgs e)

{

Clean(groupBox2);

validation.Text = String.Empty;

}

private void InitialApproximation\_TextChanged(object sender, EventArgs e)

{

Clean(groupBox2);

validation.Text = String.Empty;

}

private void Tolerance\_TextChanged(object sender, EventArgs e)

{

Clean(groupBox2);

validation.Text = String.Empty;

}

private void SearchStep\_TextChanged(object sender, EventArgs e)

{

Clean(groupBox2);

validation.Text = String.Empty;

}

private void ParametrR\_TextChanged(object sender, EventArgs e)

{

Clean(groupBox2);

validation.Text = String.Empty;

}

private void LimitOfTime\_TextChanged(object sender, EventArgs e)

{

Clean(groupBox2);

validation.Text = String.Empty;

}

private void LimitOfIterations\_TextChanged(object sender, EventArgs e)

{

Clean(groupBox2);

validation.Text = String.Empty;

}

private void Maximum\_CheckedChanged(object sender, EventArgs e)

{

Clean(groupBox2);

validation.Text = String.Empty;

}

private void Minimum\_CheckedChanged(object sender, EventArgs e)

{

Clean(groupBox2);

validation.Text = String.Empty;

}

private void Form1\_FormClosed(object sender, FormClosedEventArgs e)

{

xls.Quit();

}

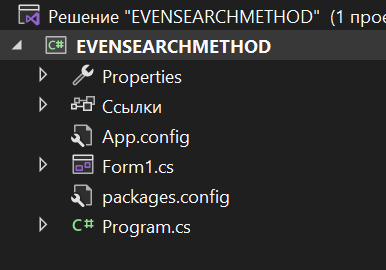
}

}

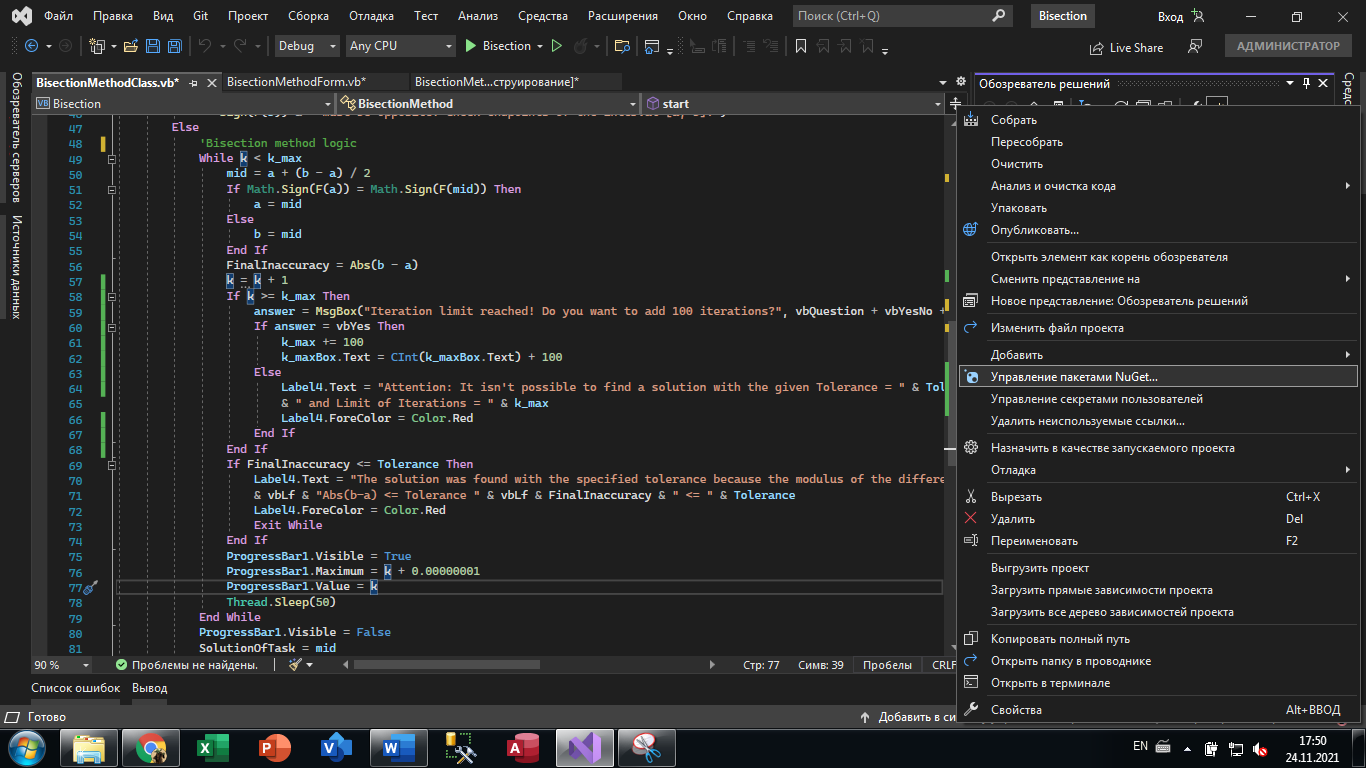
Stage No.5: Подключение библиотечных программ “info.lundin.math.dll” и «Microsoft.Office.Interop.Excel» к программному проекту для выполнения функции парсинга и открытия excel файла

***Примечание***: *Если вставить приведенные в этом отчете коды программы, то Visual Studio выделит строчки кода листинга программы, в которых есть ссылки на библиотечные функции «info.lundin.math» и* «Microsoft.Office.Interop.Excel»*, как ошибочные. Это связано с тем, что в проект не включены ссылки на эти функции. Ниже приведена инструкция по включению в проект библиотек.*

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

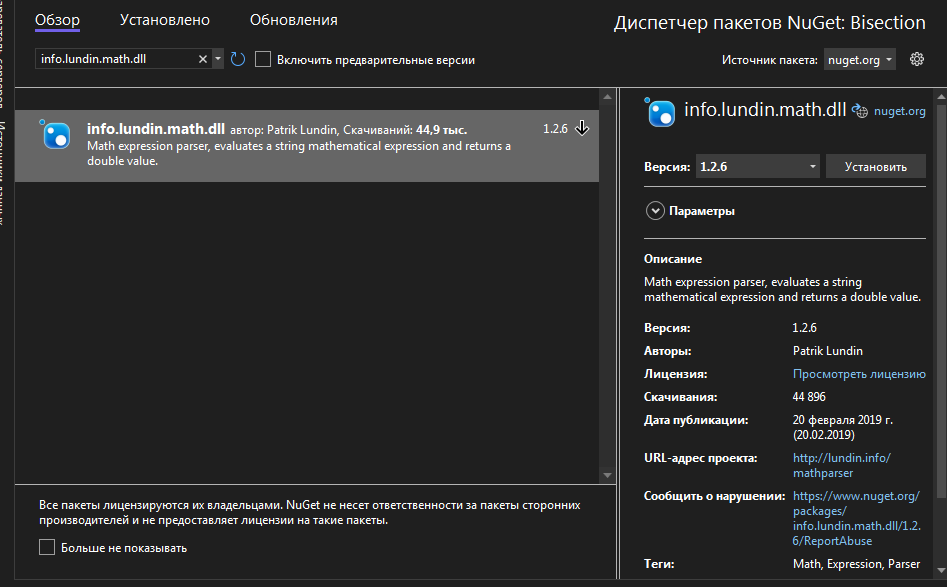


Шаг №3.2: Щелкнуть мышкой на строчке «Управление пакетами NuGet»:

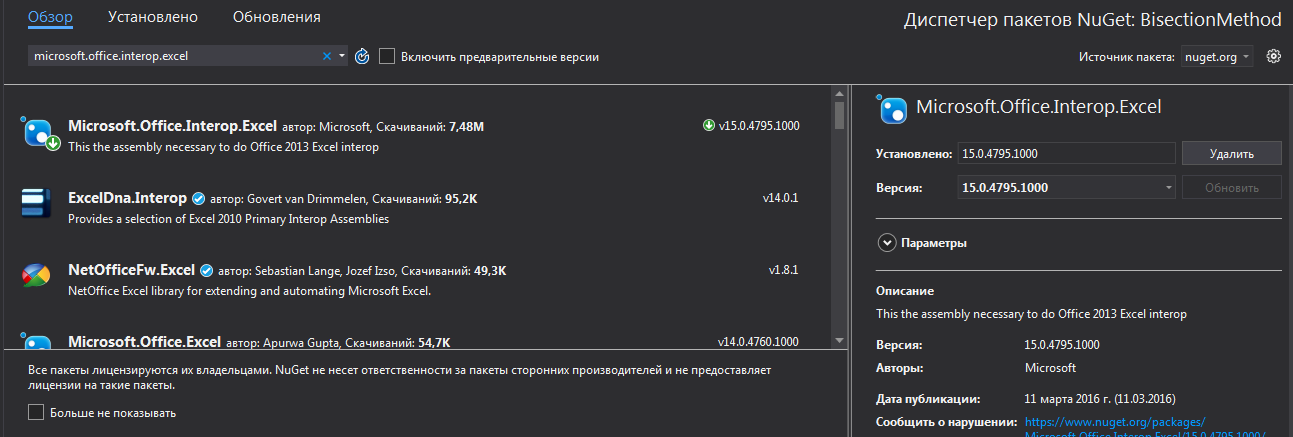


Шаг №3.3: В диалоговом окне в поиске ввести имя файла “info.lundin.math.dll”:

И установить пакет



Шаг №3.4: В диалоговом окне в поиске ввести имя файла “ microsoft.office.interop.excel”:

И установить пакет

Шаг №3.5: В листинге программы исчезнут все пометки об ошибках в коде программы, связанных с тем, что операторы «Imports info.lundin.math» и «Microsoft.Office.Interop.Excel» были неопределены, если нижеприведенные строки кода программы были уже введены до введения ссылки на эти библиотечные функции. 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”\*\*\*\*\*\*\*\*\*\*

ExpressionParser parser = new ExpressionParser();

parser.Values.Add("x", (double)x0);

double F1 = parser.Parse(inputFuncFX);

\*\*\*\*\*\*\*\*\*\*это примеры использования “Microsoft.Office.Interop.Excel”\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public void OpenExcel()

{

Application xls;

Workbook book;

Worksheet sheet;

string func;

double startPoint;

xls = new Application();

book = xls.Workbooks.Open(System.IO.Directory.GetCurrentDirectory() + nameOfExcel);

sheet = book.Sheets["Russian"];

xls.Visible = true;

sheet.Activate();

func = ComboBoxFunction.Text;

startPoint = Double.Parse(TextBoxStartPoint.Text);

sheet.Cells[4, 9] = startPoint;

sheet.Cells[4, 10] = startPoint + 1;

sheet.Cells[2, 1] = "f(x)=" + ComboBoxFunction.Text;

StringBuilder builder = new StringBuilder(func);

builder.Replace("exp", ":");

builder.Replace("x", "D4");

builder.Replace(":", "exp");

func = builder.ToString();

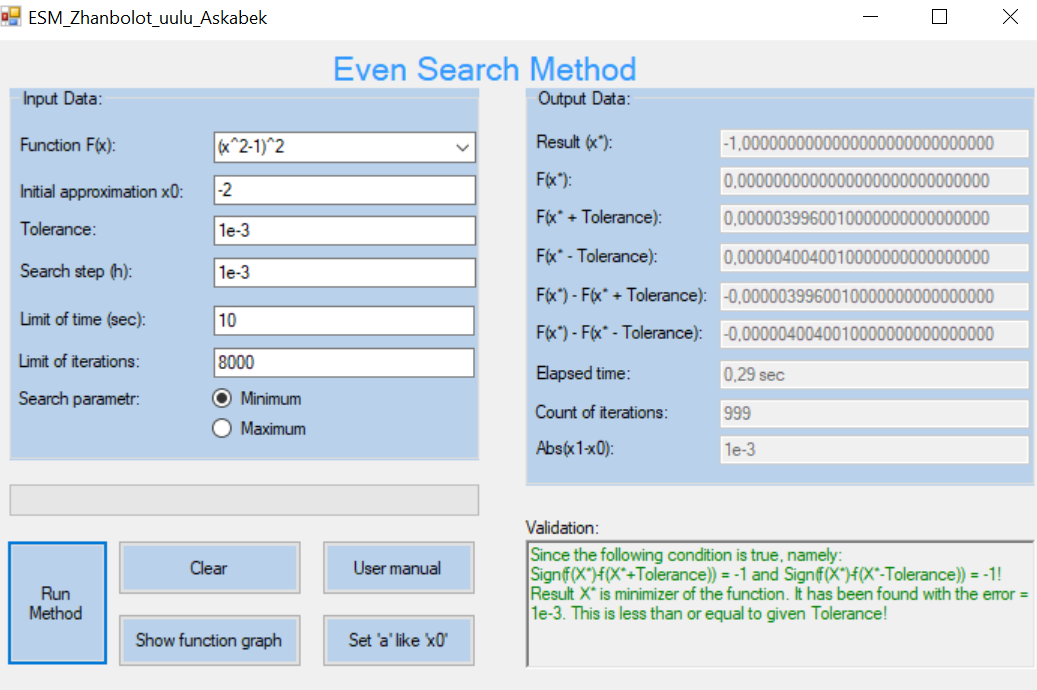
sheet.Range["E4:E10003"].Value = "=" + func;

}

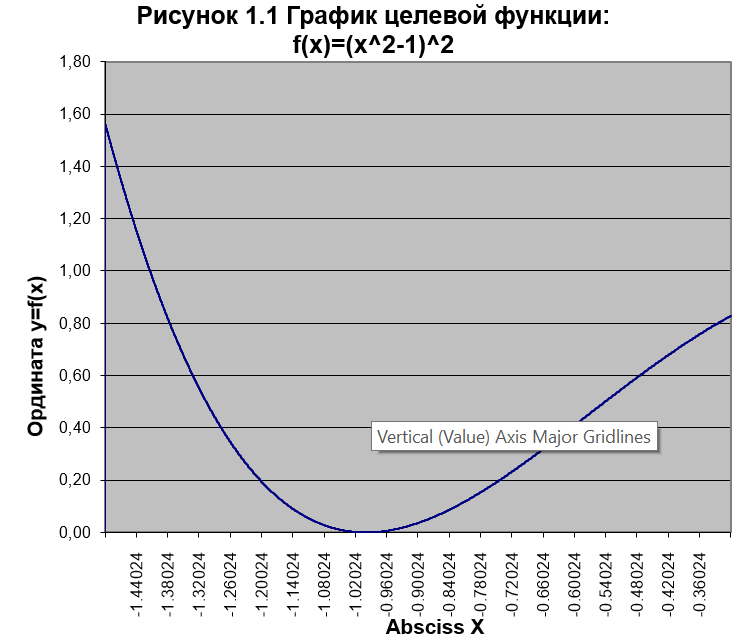
**Раздел 7**

**Тесты для проверки ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ, РЕАЛИЗУЮЩЕГО EVEN SEARCH METHOD**

Тест №1: Цель теста проверка валидности решения задачи минимизации

**

ЗАКЛЮЧЕНИЕ по результатам проведения Теста №1:

**

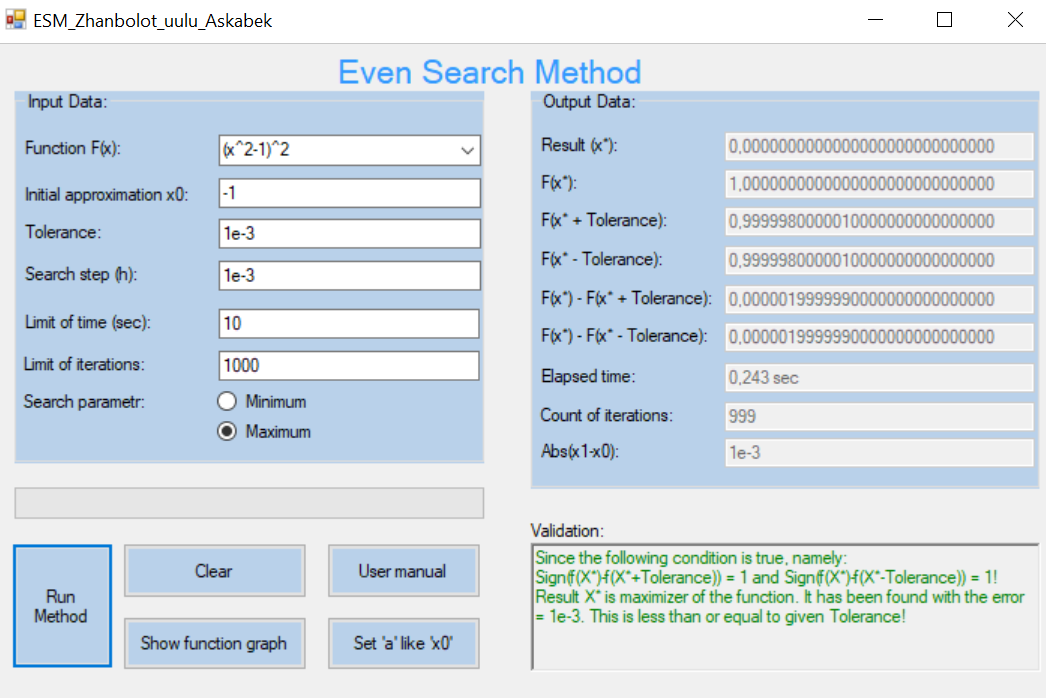
F(x\*+Tolerance)

F(x\*-Tolerance)

F(x\*)

***Screen-shot 7.1*** The graph of the function 𝑓(𝑥) = which visualizes the validation’s sentence such as: the value of the point x\* is a root of the nonlinear equation 𝑓(𝑥 ∗ ) = -1,0000000000000000000000 𝑡ℎ𝑒 𝑜𝑛𝑒 𝑖𝑠 𝑐𝑜𝑚𝑝𝑢𝑡𝑒𝑑 𝑤𝑖𝑡ℎ 𝑡ℎ𝑒 𝒅𝒆𝒔𝒊𝒓𝒂𝒃𝒍𝒆 𝒂𝒄𝒄𝒖𝒓𝒂𝒄𝒚 since the following conditions are true, namely: 𝑠𝑖𝑔𝑛[𝑓(𝑥 ∗ + 𝑇𝑜𝑙𝑒𝑟𝑎𝑛𝑐𝑒)] = -1 𝒂𝒏𝒅 𝑠𝑖𝑔𝑛[𝑓(𝑥 ∗ − 𝑇𝑜𝑙𝑒𝑟𝑎𝑛𝑐𝑒)] = -1

Тест №2: Цель теста проверка валидности решения задачи максимизации



ЗАКЛЮЧЕНИЕ по результатам проведения Теста №2:



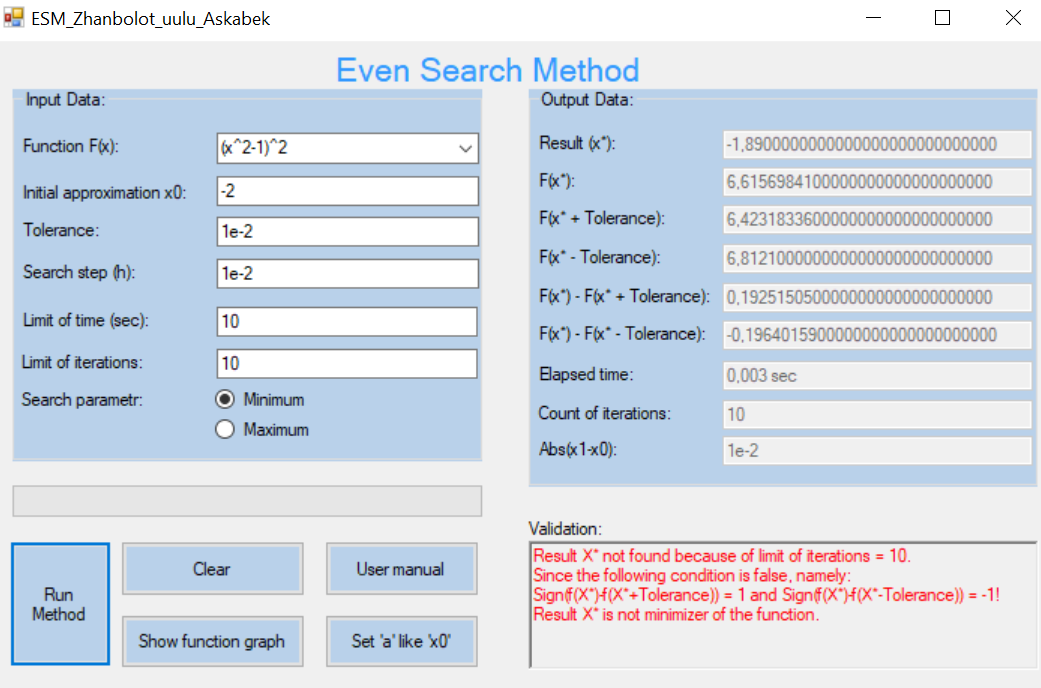
F(x\*+Tolerance)

F(x\*-Tolerance)

F(x\*)

***Screen-shot 7.2*** The graph of the function 𝑓(𝑥) = which visualizes the validation’s sentence such as: the value of the point x\* is a root of the nonlinear equation 𝑓(𝑥 ∗ ) = 0,00000000000000000000 𝑡ℎ𝑒 𝑜𝑛𝑒 𝑖𝑠 𝑐𝑜𝑚𝑝𝑢𝑡𝑒𝑑 𝑤𝑖𝑡ℎ 𝑡ℎ𝑒 𝒅𝒆𝒔𝒊𝒓𝒂𝒃𝒍𝒆 𝒂𝒄𝒄𝒖𝒓𝒂𝒄𝒚 since the following conditions are true, namely: 𝑠𝑖𝑔𝑛[𝑓(𝑥 ∗ + 𝑇𝑜𝑙𝑒𝑟𝑎𝑛𝑐𝑒)] = 1 𝒂𝒏𝒅 𝑠𝑖𝑔𝑛[𝑓(𝑥 ∗ − 𝑇𝑜𝑙𝑒𝑟𝑎𝑛𝑐𝑒)] = 1

Тест №3: Цель теста проверка валидности на ограничение итераций



ЗАКЛЮЧЕНИЕ по результатам проведения Теста №3:



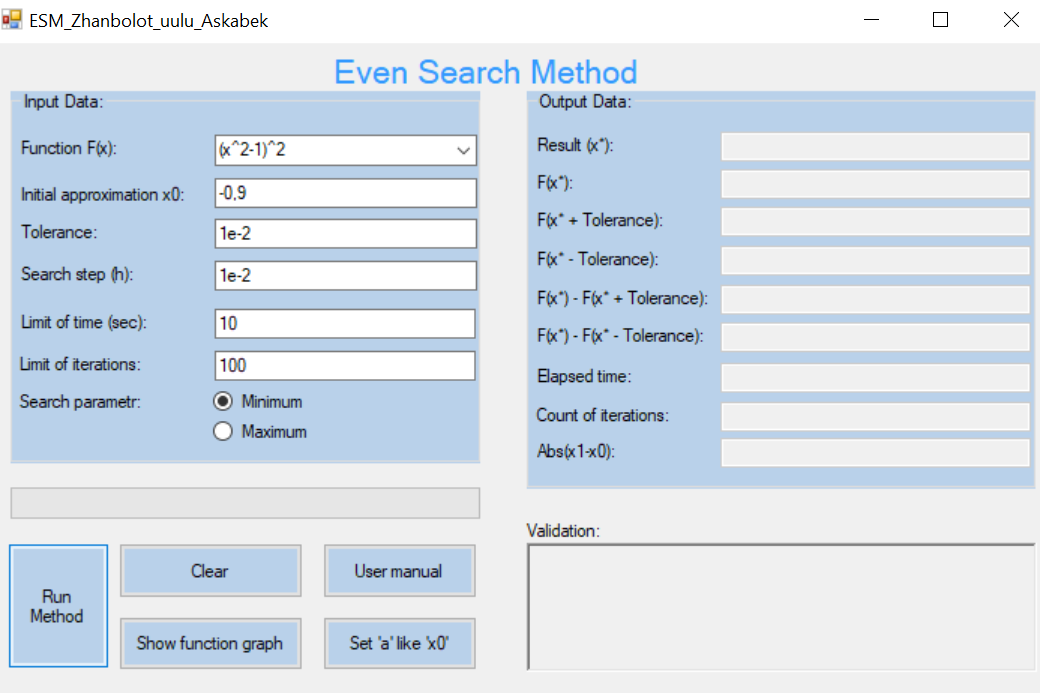
F(x\*-Tolerance)

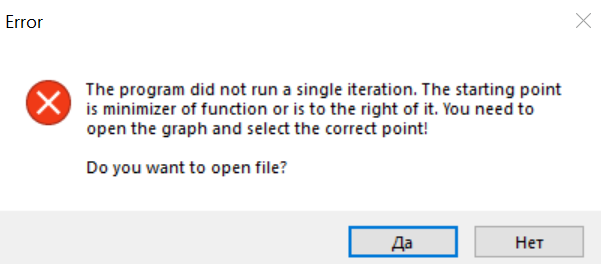
F(x\*+Tolerance)

F(x\*)

***Screen-shot 7.2*** The graph of the function 𝑓(𝑥) = which visualizes the validation’s sentence such as: the value of the point x\* is not a root of the nonlinear equation 𝑓(𝑥∗) = -1,890000000000000000000 since the following conditions are false, namely: 𝑠𝑖𝑔𝑛[𝑓(𝑥 ∗ + 𝑇𝑜𝑙𝑒𝑟𝑎𝑛𝑐𝑒)] = 1 𝒂𝒏𝒅 𝑠𝑖𝑔𝑛[𝑓(𝑥 ∗ − 𝑇𝑜𝑙𝑒𝑟𝑎𝑛𝑐𝑒)] = -1

Тест №4: Цель теста проверка валидности решения задачи на ввод неверной начальной точки



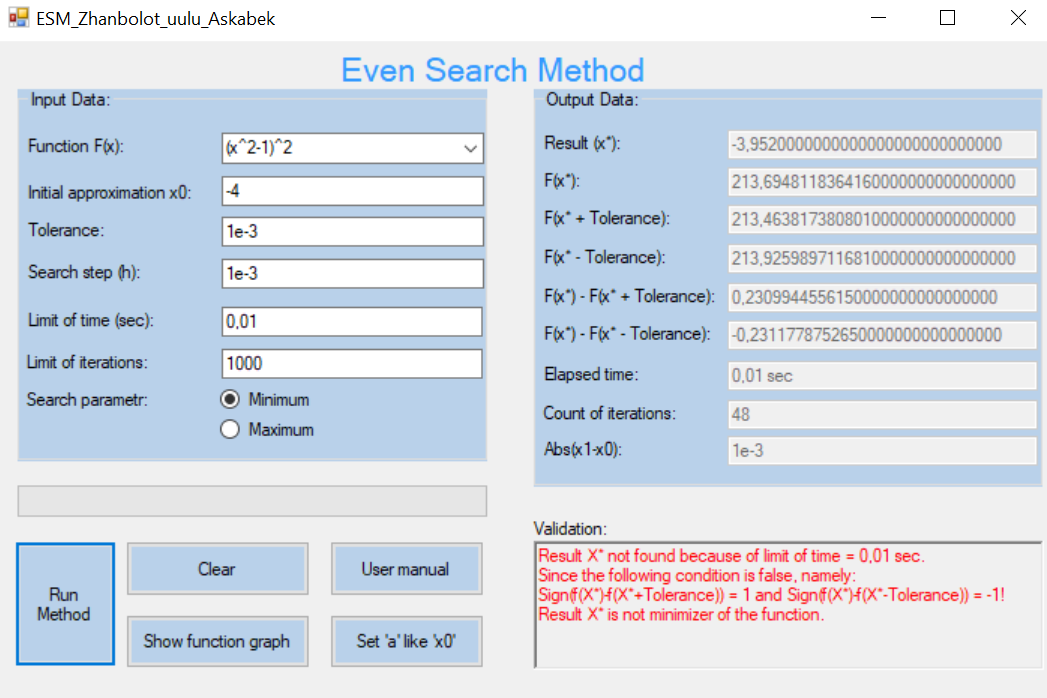


ЗАКЛЮЧЕНИЕ по результатам проведения Теста №4:



The program did not run a single iteration. The starting point is minimizer of function or is to the right of it!

Тест №5: Цель теста проверка валидности на ограничение времени



ЗАКЛЮЧЕНИЕ по результатам проведения Теста №5:



F(x\*-Tolerance)

F(x\*+Tolerance)

F(x\*)

***Screen-shot 7.5*** The graph of the function 𝑓(𝑥) = (𝑥^2 – 1)^2 which visualizes the validation’s sentence such as: the value of the point x\* is not a root of the nonlinear equation 𝑓(𝑥∗) = -3,9520000000000000000000 since the following conditions are false, namely: 𝑠𝑖𝑔𝑛[𝑓(𝑥 ∗ -𝑇𝑜𝑙𝑒𝑟𝑎𝑛𝑐𝑒)] = -1 𝒂𝒏𝒅 𝑠𝑖𝑔𝑛[𝑓(𝑥 ∗ + 𝑇𝑜𝑙𝑒𝑟𝑎𝑛𝑐𝑒)] = 1