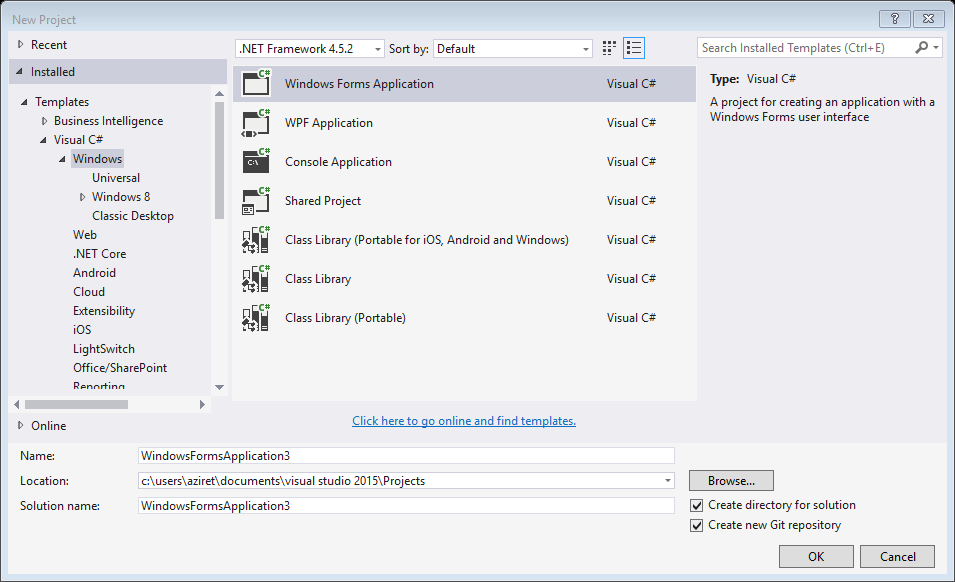
Рассмотрим использование программного модуля на примере создания простой программы:

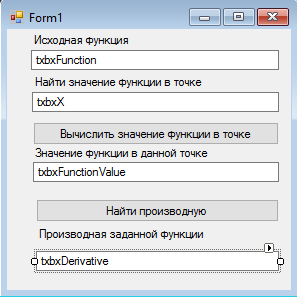
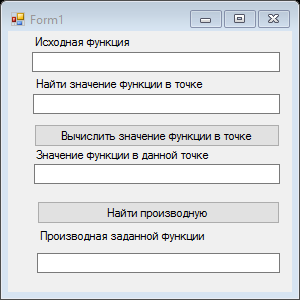
Совершаем ряд следующих действий:

Visual Studio -> New Project -> Visual C# -> Windows Forms Application

Называем проект как вам будет угодно, в моем случае имя по умолчанию WindowsFormsApplication3



Далее настраиваем форму как показано ниже на рисунке

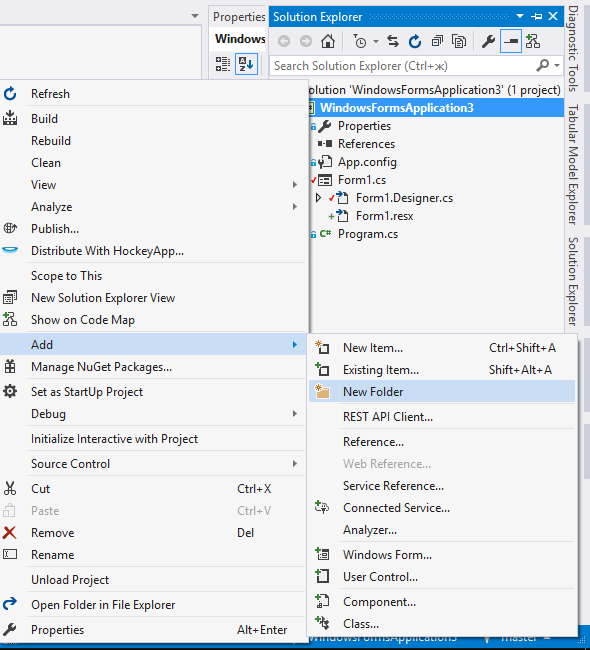


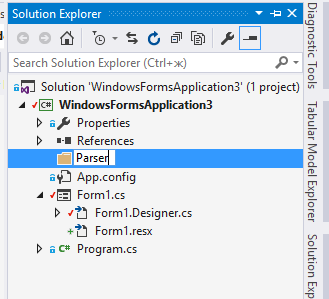
В этой форме находятся следующие элементы управления:

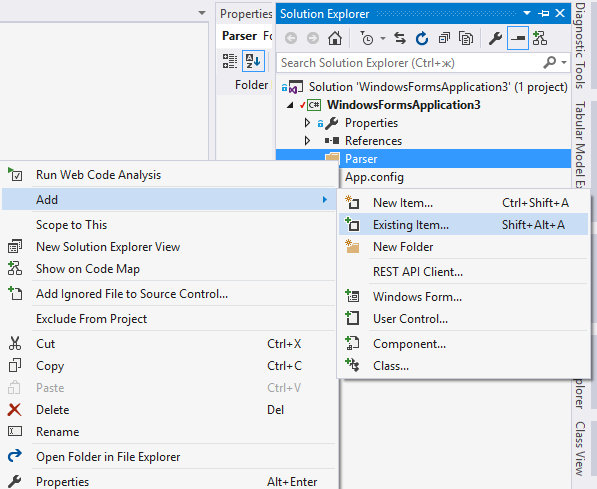
Textbox – в количестве 4 шт., label – в количестве 4 шт., button – в количестве 2 шт.

На рисунке справа внутри текстовых полей указаны их имена в интерфейсной форме.

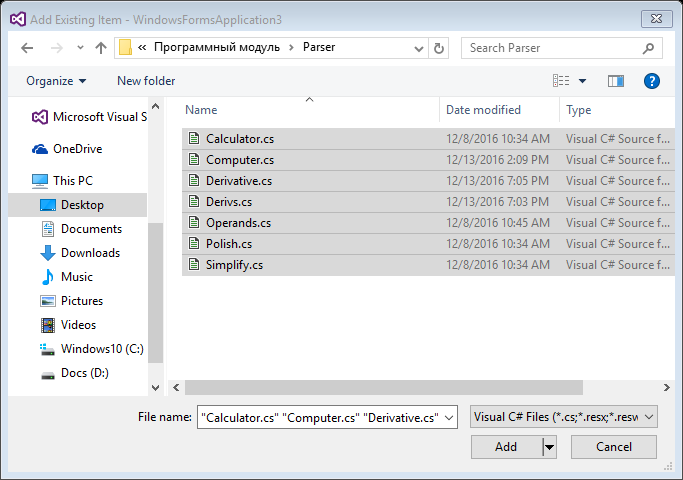
Создаем папку внутри проекта в обозревателе решений и называем Parser:

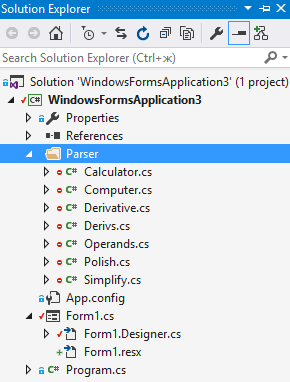




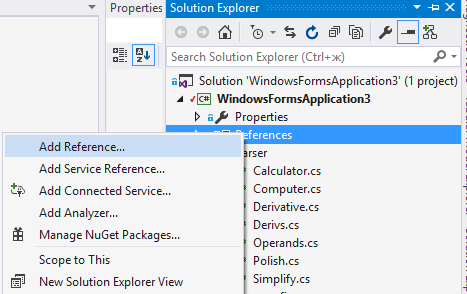


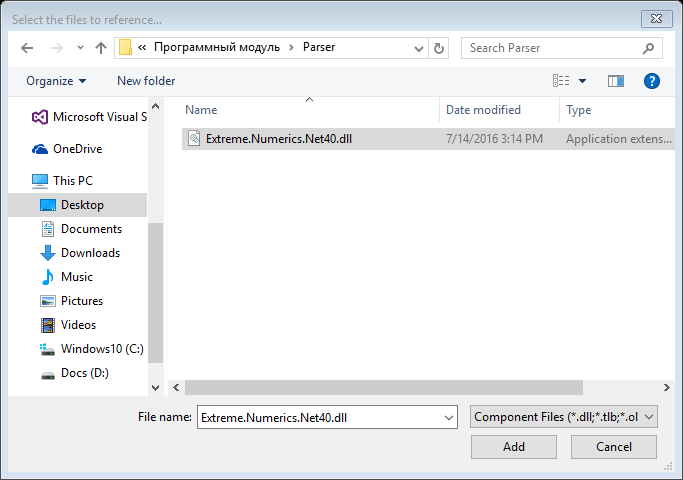
Далее в эту папку добавляем существующие программные модули:





Далее добавляем ссылку на библиотеку Extreme.Numerics.Net40.dll





В итоге получаем готовую к использованию программную логику. Код программы использующий готовый модуль:

using System;

using System.Windows.Forms;

using parserDecimal.Parser;

namespace WindowsFormsApplication3

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

Computer computer = new Computer();

Derivative derivative = new Derivative();

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

{

string function = txbxFunction.Text;

decimal value = decimal.Parse(txbxX.Text);

txbxFunctionValue.Text = computer.Compute(function, value).ToString();

}

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

{

string function = txbxFunction.Text;

txbxDerivative.Text = derivative.ReturnDerivative(function);

}

}

}

Описание кода программы

Модуль для вычисления значения функции:

Computer computer = new Computer();

Модуль для нахождения производной функции:

Derivative derivative = new Derivative();

Вычисление значения функции:

1. string function = txbxFunction.Text;
2. decimal value = decimal.Parse(txbxX.Text);
3. txbxFunctionValue.Text = computer.Compute(function, value).ToString();

в первой строке заносим аналитическое выражение в переменную function

во второй строке заносим значение точки в переменную value, предварительно преобразовав в значение типа decimal

в третей строке получаем значение функции в точке

Вычисление производной функции:

1. string function = txbxFunction.Text;
2. txbxDerivative.Text = derivative.ReturnDerivative(function);

в первой строке заносим аналитическое выражение в переменную function

во второй строке получаем значение производной функции

Пример работы программы:

