**Оглавление**

[1. Вычисление значений выражений. 3](#_Toc179328260)

[1.1.Задание 3](#_Toc179328261)

[1.2.Код программы 3](#_Toc179328262)

[1.3.Тестирование 4](#_Toc179328263)

[2. Вычисление значения функции, заданной её графиком 5](#_Toc179328264)

[2.1.Задание 5](#_Toc179328265)

[2.2.Код программы 5](#_Toc179328266)

[2.3.Тестирование 6](#_Toc179328267)

[3. Проверка принадлежности точки заданной области на плоскости 8](#_Toc179328268)

[3.1.Задание 8](#_Toc179328269)

[3.2.Код программы 9](#_Toc179328270)

[3.3.Тестирование 10](#_Toc179328271)

[4. Вывод таблицы значений функции, заданной её графиком 11](#_Toc179328272)

[4.1.Задание 11](#_Toc179328273)

[4.2.Код приложения 11](#_Toc179328274)

[4.3.Тестирование 13](#_Toc179328275)

[5. Проверка серии «выстрелов по мишени» 13](#_Toc179328276)

[5.1.Задание 13](#_Toc179328277)

[5.2.Код программы 14](#_Toc179328278)

[5.3.Тестирование 16](#_Toc179328279)

[6. Вычисление суммы ряда с заданной точностью 16](#_Toc179328280)

[6.1.Задание 16](#_Toc179328281)

[6.2.Код программы 17](#_Toc179328282)

[6.3.Тестирование 18](#_Toc179328283)

[7. Вывод таблицы значений функции, вычисляемой с помощью ряда Тейлора 19](#_Toc179328284)

[7.1. Задание 19](#_Toc179328285)

[7.2.Код приложения 20](#_Toc179328286)

[7.3.Тестирование 21](#_Toc179328287)

# 

# 1. Табулирование функции.

1.1.Задание:Функция y=f(x) задана графически (см. задание № 1). Для определения значений аргумента X задан отрезок [A, B] и целое число N.

Необходимо:

1. Решить задачу табуляции функции на множестве точек, полученных делением заданного отрезка [A, B] на N равных частей. Для каждой из точек, координаты которых определяют значения аргумента X, вычисляется значение функции Y.
2. Результаты сохранить в виде двумерного массива D (Data) с двумя столбцами, каждая строка которого соответствует одной паре найденных значений {X, Y}.
3. Используя сформированный массив D, вывести на экран таблицу, состоящую из трёх столбцов:
4. порядковый номер точки;
5. аргумент функции X;
6. значение функции Y.

Решение задачи реализовать с помощью отдельных подпрограмм с параметрами:

1. для формирования двумерного массива D;
2. для вывода на экран таблицы с использованием известного массива D.

Для определения значения функции целесообразно использовать готовую подпрограмму (см. задание № 2).

***Исходные данные:***

а – начало отрезка, b – конец отрезка, N – кол-во делений отрезка.

***Результаты:***

Таблица состоящая из 3х столбцов:

1. порядковый номер точки;
2. аргумент функции X;
3. значение функции Y.

## 1.2. Требование к приложению

-Ввод исходных данных в поля для ввода(value\_a, value\_b, value\_N).

-Визуализация математической постановки задачи при помощи графика.

- Выполнение Необходимых расчётов для вычисления значений по формулам для F(x) после нажатия на кнопку(Play).

- Вывод и интерпретация решения задачи в таблицу (dataGridView1)

## 1.3. Описание проектируемого UI

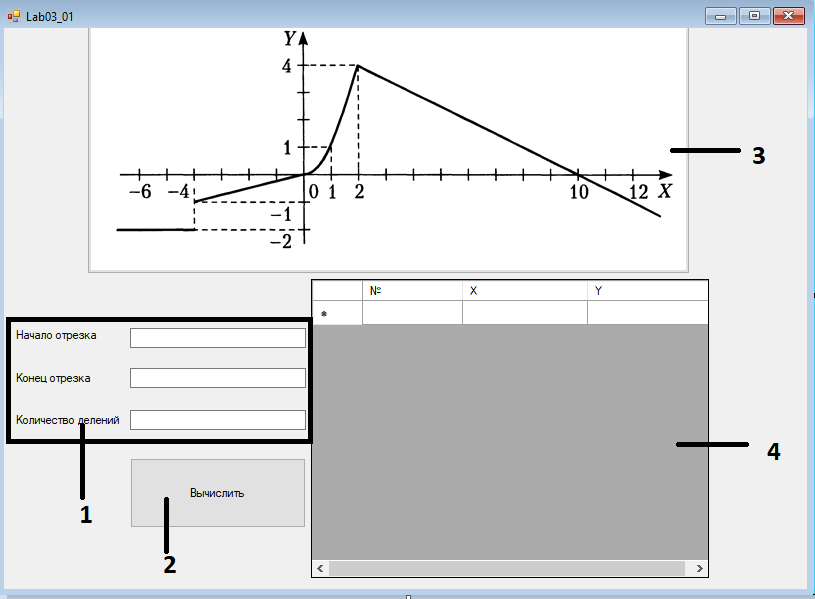


Рисунок 1.1 – UI приложения Lab03\_01

Пользовательский интерфейс представляет собой форму, на которой расположены следующие компоненты:

1. Компонент value\_a, value\_b, value\_N, предназначенные для ввода значений аргумента a, b, N соответственно.
2. Компонент кнопки Play, предназначенный для старта вычислений.
3. Компонент pictureBox для вывода графика функций
4. Компонент dataGridView1 для вывода таблицы полученых результатов.

## 1.4.Код программы:

***Листинг 1 – Код программы Lab03\_01***

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Lab03\_01

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

public void Output(string[,] array, int N)

{

for (int i = 0; i < N + 1; i++)

{

string[] row = {Convert.ToString(i+1), array[i, 0], array[i, 1] };

dataGridView1.Rows.Add(row);

}

}

public void СonclusionNValue(double a, double b, int N)

{

string [,] array = new string[N+1,2];

double x, y;

double AB = b - a;

if (AB <= 0)

{

MessageBox.Show("Введён несуществующий отрезок");

}

else

{

double j = AB / N;

x = a;

for (int i = 0; i <= N; i++)

{

array[i, 0] = Convert.ToString(x);

if (x <= -4)

{

y = -2;

array[i, 1] = Convert.ToString(y);

}

else if (x <= 0)

{

y = x / 4;

array[i, 1] = Convert.ToString(y);

}

else if (x <= 2)

{

y = Math.Pow(x, 2);

array[i, 1] = Convert.ToString(y);

}

else

{

y = (-x + 10) / 2;

array[i, 1] = Convert.ToString(y);

}

x += j;

}

Output(array, N);

}

}

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

{

try

{

double a = Convert.ToDouble(value\_a.Text);

double b = Convert.ToDouble(value\_b.Text);

int N = Convert.ToInt16(value\_N.Text);

//отчистка таблицы, если в ней имеются значения при старте новых вычислений

int rowsCount = dataGridView1.Rows.Count;

if (rowsCount > 1)

{

for (int i = 1; i < rowsCount; i++)

{

dataGridView1.Rows.Remove(dataGridView1.Rows[0]);

}

}

СonclusionNValue(a, b, N);

}

catch

{

MessageBox.Show("Введено НЕ число", "Внимание!");

}

}

}

}

## 1.5.Тестирование:

Контрольный пример 1:

Исходные данные:

a = -10

b = 10

N = 10

Результат работы программы для указанных исходных данных приведён на рисунке 1.2

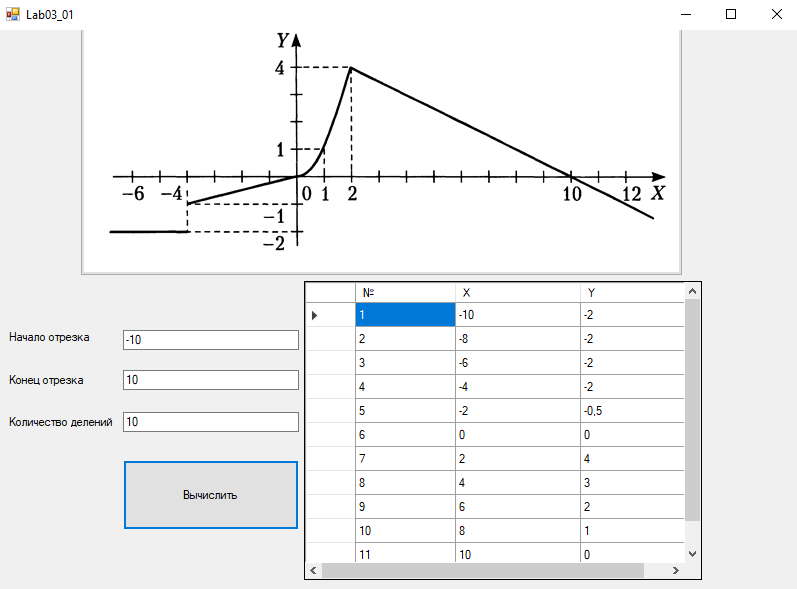


Рисунок 1.2 – Результат работы приложения Lab03\_01

# 2. Определение принадлежности точек заданной области

2.1.Задание:Ограниченная область на плоскости задана графически (см. задание № 1). Задано N точек на плоскости, которые могут принадлежать или не принадлежать заданной области.

Необходимо:

1. Ввести координаты N точек в виде двумерного массива P (Points) с двумя столбцами, каждая строка которого соответствует одной точке с координатами (x, y).
2. Для каждой из N точек, описывающихся строками двумерного массива P, определить, принадлежит ли точка заданной области или нет.
3. Результаты проверки сохранить в одномерном массиве R (Results) значений логического типа («истина» – точка принадлежит, «ложь» – точка не принадлежит). Размер массива и порядок значений соответствует количеству и порядку заданных точек.
4. Используя сформированные массивы P и R, вывести на экран таблицу, состоящую из трёх столбцов:
5. порядковый номер точки;
6. координаты точки в формате «(x, y)»;
7. результат проверки в виде сообщения «принадлежит» или «не принадлежит».

Решение задачи реализовать с помощью отдельных подпрограмм с параметрами:

1. для ввода (заполнения данными) двумерного массива P;
2. для формирования одномерного массива R;
3. для вывода на экран единой таблицы с использованием обоих известных массивов P и R.

Для проверки принадлежности точки области необходимо использовать готовую подпрограмму.

***Исходные данные:***

а – начало отрезка, b – конец отрезка, N – кол-во делений отрезка.

***Результаты:***

Таблица состоящая из 3х столбцов:

1. порядковый номер точки;
2. аргумент функции X;
3. значение функции Y.

## 1.2. Требование к приложению

-Ввод исходных данных в поля для ввода(value\_a, value\_b, value\_N).

-Визуализация математической постановки задачи при помощи графика.

- Выполнение Необходимых расчётов для вычисления значений по формулам для F(x) после нажатия на кнопку(Play).

- Вывод и интерпретация решения задачи в таблицу (dataGridView1)

## 1.3. Описание проектируемого UI

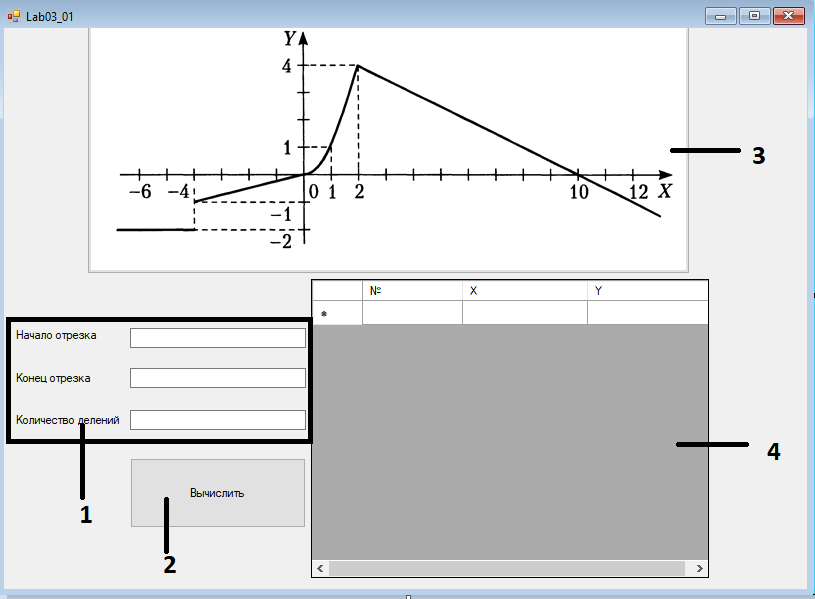


Рисунок 1.1 – UI приложения Lab03\_01

Пользовательский интерфейс представляет собой форму, на которой расположены следующие компоненты:

1. Компонент value\_a, value\_b, value\_N, предназначенные для ввода значений аргумента a, b, N соответственно.
2. Компонент кнопки Play, предназначенный для старта вычислений.
3. Компонент pictureBox для вывода графика функций
4. Компонент dataGridView1 для вывода таблицы полученых результатов.

## 1.4.Код программы:

***Листинг 1 – Код программы Lab03\_01***

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Lab03\_01

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

public void Output(string[,] array, int N)

{

for (int i = 0; i < N + 1; i++)

{

string[] row = {Convert.ToString(i+1), array[i, 0], array[i, 1] };

dataGridView1.Rows.Add(row);

}

}

public void СonclusionNValue(double a, double b, int N)

{

string [,] array = new string[N+1,2];

double x, y;

double AB = b - a;

if (AB <= 0)

{

MessageBox.Show("Введён несуществующий отрезок");

}

else

{

double j = AB / N;

x = a;

for (int i = 0; i <= N; i++)

{

array[i, 0] = Convert.ToString(x);

if (x <= -4)

{

y = -2;

array[i, 1] = Convert.ToString(y);

}

else if (x <= 0)

{

y = x / 4;

array[i, 1] = Convert.ToString(y);

}

else if (x <= 2)

{

y = Math.Pow(x, 2);

array[i, 1] = Convert.ToString(y);

}

else

{

y = (-x + 10) / 2;

array[i, 1] = Convert.ToString(y);

}

x += j;

}

Output(array, N);

}

}

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

{

try

{

double a = Convert.ToDouble(value\_a.Text);

double b = Convert.ToDouble(value\_b.Text);

int N = Convert.ToInt16(value\_N.Text);

//отчистка таблицы, если в ней имеются значения при старте новых вычислений

int rowsCount = dataGridView1.Rows.Count;

if (rowsCount > 1)

{

for (int i = 1; i < rowsCount; i++)

{

dataGridView1.Rows.Remove(dataGridView1.Rows[0]);

}

}

СonclusionNValue(a, b, N);

}

catch

{

MessageBox.Show("Введено НЕ число", "Внимание!");

}

}

}

}

## 1.5.Тестирование:

Контрольный пример 1:

Исходные данные:

a = -10

b = 10

N = 10

Результат работы программы для указанных исходных данных приведён на рисунке 1.2

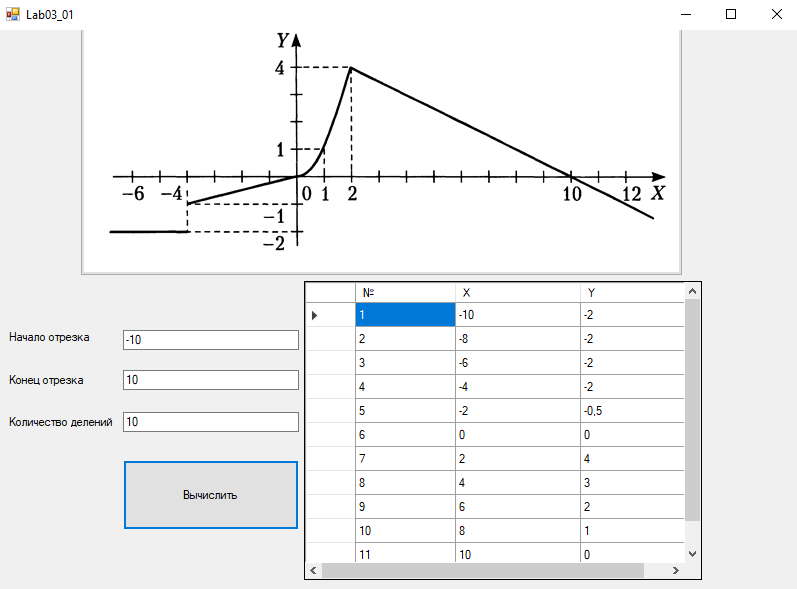


Рисунок 1.2 – Результат работы приложения Lab03\_01