Министерство цифрового развития, связи и массовых коммуникаций РФ

ордена Трудового Красного Знамени федеральное государственное

бюджетное образовательное учреждение высшего образования

«МОСКОВСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ СВЯЗИ И

ИНФОРМАТИКИ»

Кафедра «Информатика»

Лабораторные работы

по дисциплине «Алгоритмы и алгоритмические языки»

Вариант №21

Выполнил:

студент группы БФИ2204

Петрушин Е.И.

Проверил:

доц. кафедры «Информатика»

к. п. н. Гуриков С. Р.

Москва, 2023 г.

**Лабораторная работа №1**

**«Программирование арифметических выражений на языке Visual C# с использованием методов»**

**Задание**

**Разработать программу для расчета арифметического выражения**

**Для того, чтобы выполнить задание, необходимо разработать следующие** **методы:**

1. Метод public static double Vvod(TextBox t), предназначенный для конвертации строкового значения в числовое.
2. Метод public static void Vivod(TextBox t, double b), предназначенный для вывода данных в текстовое меню.
3. Метод public static double Raschet(double a, double b , double x, double y), предназначенный для возвращающего нахождения значения выражения по исходным данным.
4. Метод public static void Raschet\_2(double a, double b, double x, double y, out double f), предназначен для значения с помощью модификации out.
5. Метод public static void Raschet\_3(double a, double b, double x, double y, ref double f), предназначенный для возвращающего значения с помощью модификатора ref.

Все приведенные методы должны быть размещены в DLL-библиотеке.

**В основной части программы следует последовательно осуществить**

**вызов ранее написанных методов.**

**Перечень блок-схем**

На рисунке 1 изображен алгоритм метода, предназначенного для конвертации строкового значения в числовое.

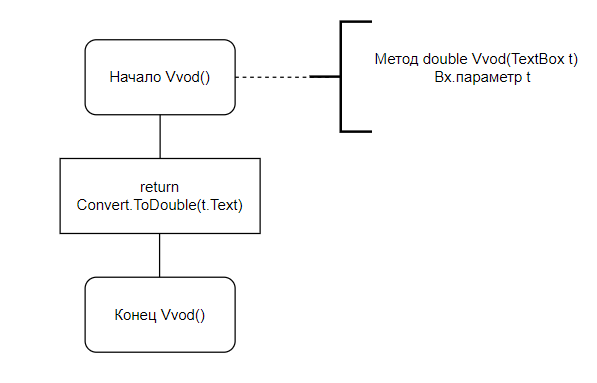


Рисунок 1 – Алгоритм метода, конвертирующего строковое значение в числовое

На рисунке 2 изображен алгоритм метода, предназначенного для вывода данных в текстовое меню.

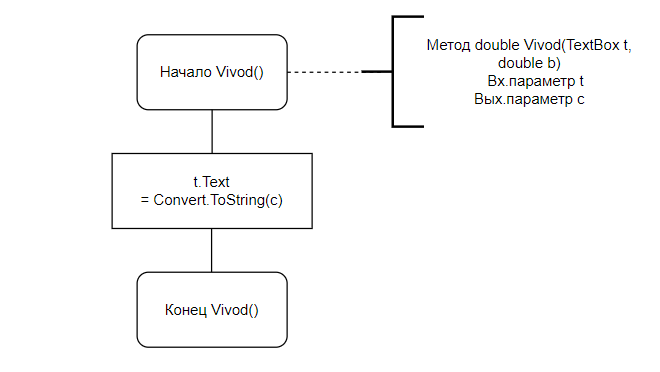


Рисунок 2 - Алгоритм метода, выводящего данные в текстовое меню.

**Ниже представлены:**

**Алгоритм метода, возвращающего значение с помощью оператора return**

**Алгоритм метода, возвращающего значение с помощью модификатора**

**out;**

**Алгоритм метода, возвращающего значение с помощью модификатора**

**ref.**

На рисунке 3 изображен алгоритм метода, предназначенного для нахождения значения выражения по исходным данным.

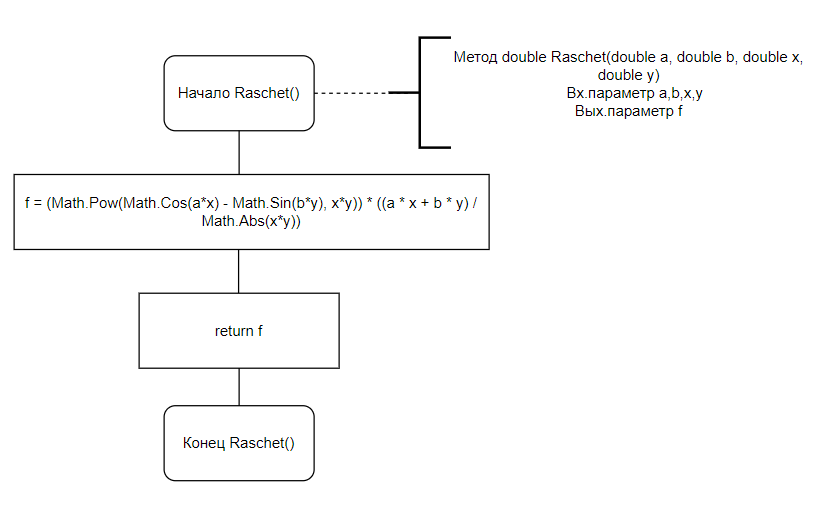


Рисунок 3 - Алгоритм метода, находящий значение выражения по исходным данным

На рисунке 4 изображен алгоритм метода, предназначенного для возвращения значения с помощью модификатора out.

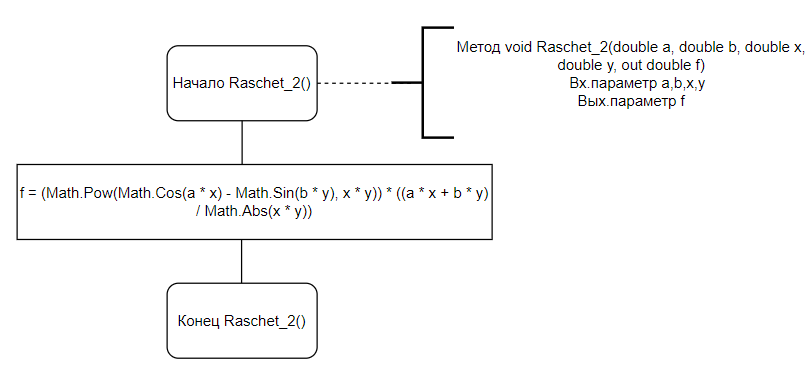


Рисунок 4 – Алгоритм метода, возвращающего значение с помощью модификатора out

На рисунке 5 изображен алгоритм метода, предназначенного для возвращения значения с помощью модификатора ref.

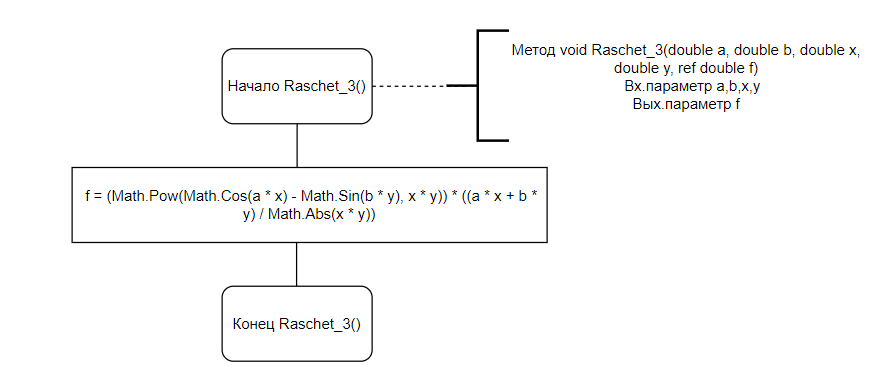


Рисунок 5 – Алгоритм метода, возвращающего значение с помощью модификатора ref

На рисунке 6 изображен алгоритм событийной кнопки.

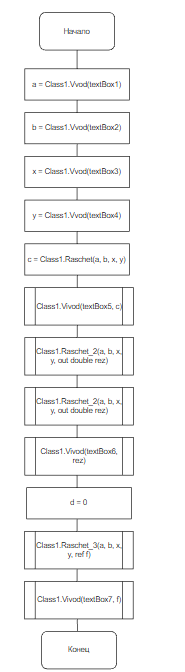


Рисунок 6 – Алгоритм событийной кнопки

**Содержание DLL-библиотеки**

using System;

using System.Windows.Forms;

namespace Rabota\_lab

{

public class Class1

{

public static double Vvod(TextBox t)

{

return Convert.ToDouble(t.Text);

}

public static void Vivod(TextBox t, double c)

{

t.Text = Convert.ToString(c);

}

public static double Raschet(double a, double b, double x, double y)

{

double f = (Math.Pow(Math.Cos(a\*x) - Math.Sin(b\*y), x\*y)) \* ((a \* x + b \* y) / Math.Abs(x\*y));

return f;

}

public static void Raschet\_2(double a, double b, double x, double y, out double f)

{

f = (Math.Pow(Math.Cos(a \* x) - Math.Sin(b \* y), x \* y)) \* ((a \* x + b \* y) / Math.Abs(x \* y));

}

public static void Raschet\_3(double a, double b, double x, double y, ref double f)

{

f = (Math.Pow(Math.Cos(a \* x) - Math.Sin(b \* y), x \* y)) \* ((a \* x + b \* y) / Math.Abs(x \* y));

}

}

}

**Содержание основной части программы**

using System;

using System.Windows.Forms;

using Rabota\_lab;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace WindowsFormsApp3

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void button1\_Click\_1(object sender, EventArgs e)

{

double a = Class1.Vvod(textBox1);

double b = Class1.Vvod(textBox2);

double x = Class1.Vvod(textBox3);

double y = Class1.Vvod(textBox4);

double c = Class1.Raschet(a, b, x, y);

Class1.Vivod(textBox5, c);

Class1.Raschet\_2(a, b, x, y, out double rez);

Class1.Vivod(textBox6, rez);

double f = 0;

Class1.Raschet\_3(a, b, x, y, ref f);

Class1.Vivod(textBox7, f);

}

}

}

**Результаты выполнения программы**

На рисунке 7 представлены результаты выполненной программы, вычисляющей значение выражения по исходным данным.

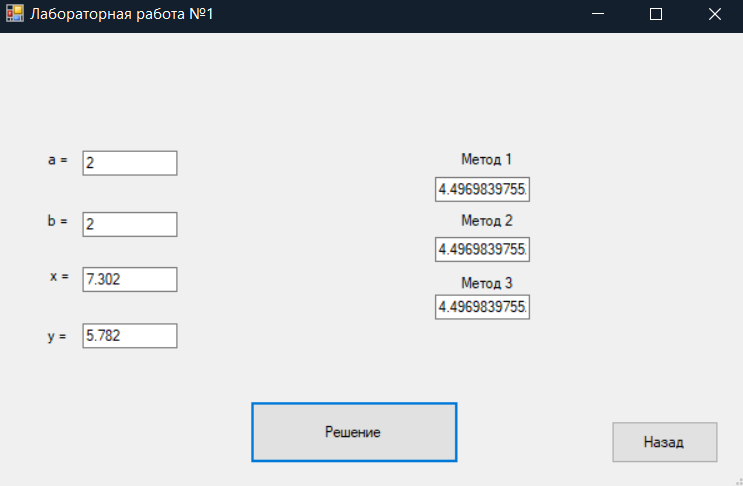


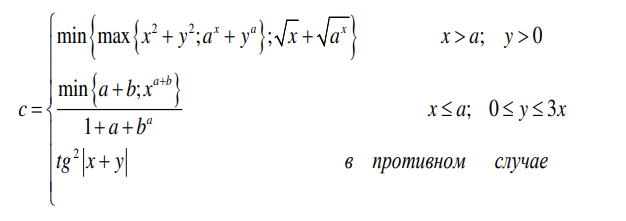
Рисунок 7 – Результаты выполненной программы

**Лабораторная работа №2**

**«Программирование алгоритмов разветвляющихся структур»**

**Задание**

**Разработать программу для расчета функции**



**Для того, чтобы выполнить задание, необходимо разработать следующие** **методы:**

1. Метод public static double Vvod(TextBox t), предназначенный для конвертации строкового значения в числовое.
2. Метод public static void Vivod(TextBox t, double b), предназначенный для вывода данных в текстовое меню.
3. Метод public static double Rashet\_function(double x, double y, double a, double b), предназначенный для установления значения функции.

Все приведенные методы должны быть размещены в DLL-библиотеке.

**В основной части программы следует последовательно осуществить**

**вызов ранее написанных методов.**

**Перечень блок-схем**

На рисунке 1 изображен алгоритм метода, предназначенного для конвертации строкового значения в числовое.

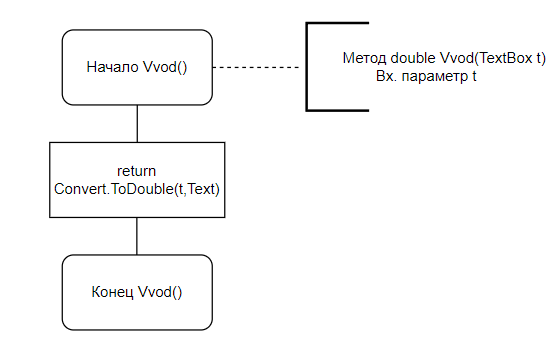


Рисунок 1 – Алгоритм метода, который конвертирует строковое значение в числовое

На рисунке 2 изображен алгоритм метода, назначенного для вывода данных в текстовое меню.

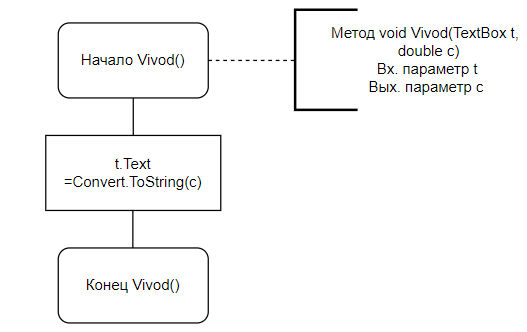


Рисунок 2 - Алгоритм метода, выводящего данные в текстовое меню.

На рисунке 3 изображен алгоритм метода, предназначенного для установления значения функции и результата с помощью оператора return.

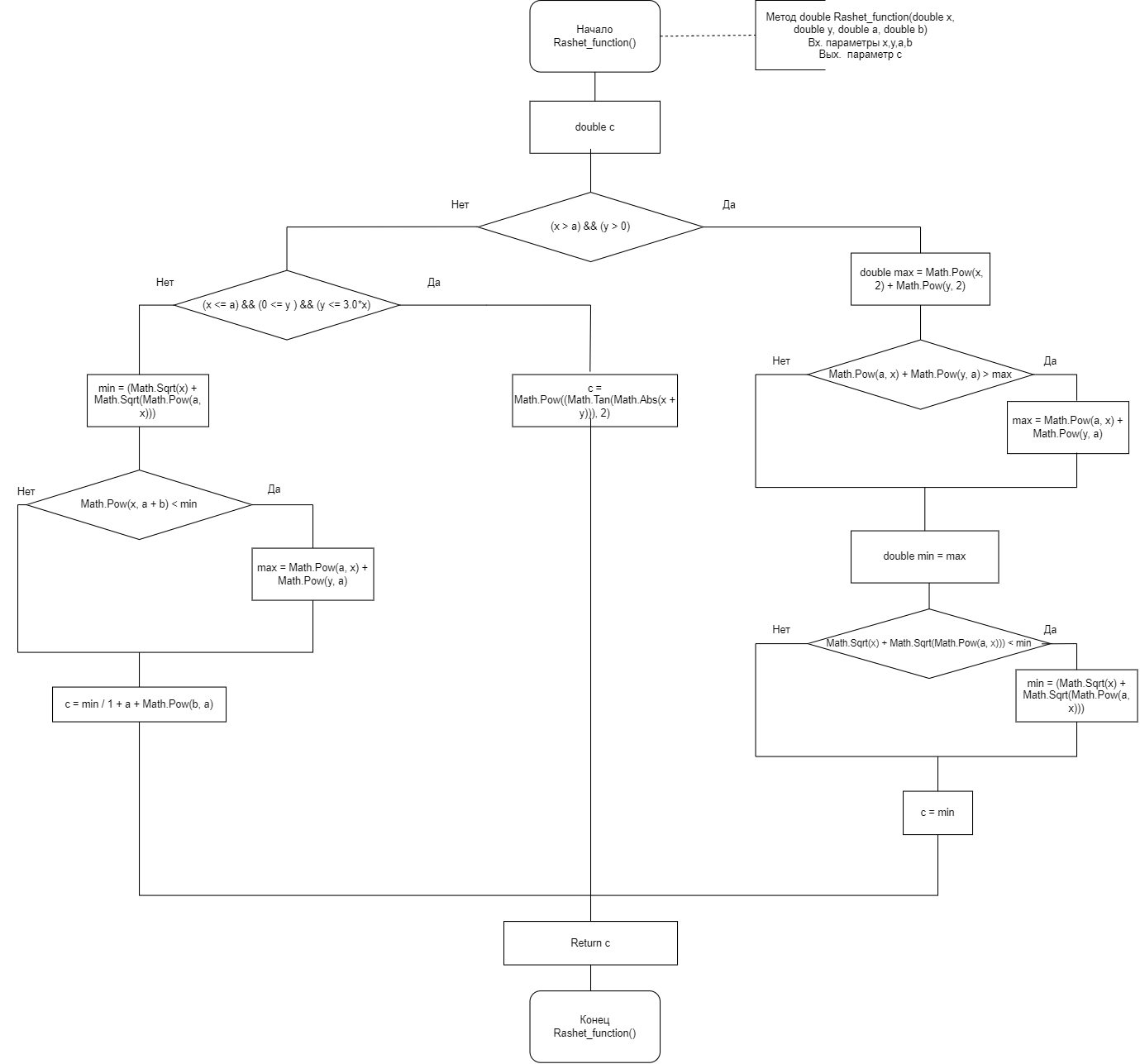


Рисунок 3 - Алгоритм метода, устанавливающий значение функции и результата оператором return

На рисунке 4 изображен алгоритм событийной кнопки.

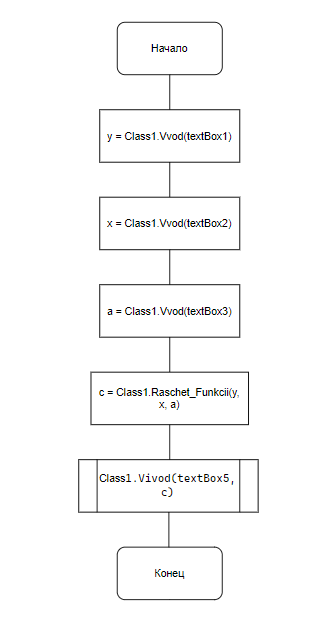


Рисунок 4 – Алгоритм событийной кнопки

**Содержание DLL-библиотеки**

using System;

using System.Runtime.CompilerServices;

using System.Windows.Forms;

namespace Rabota\_lab

{

public class Class1

{

public static double Vvod(TextBox t)

{

return Convert.ToDouble(t.Text);

}

public static void Vivod(TextBox t, double c)

{

t.Text = Convert.ToString(c);

}

public static double Raschet\_function(double x, double y, double a, double b)

{

double c;

if ((x > a) && (y > 0))

{

double max = Math.Pow(x, 2) + Math.Pow(y, 2);

if (Math.Pow(a, x) + Math.Pow(y, a) > max)

{

max = Math.Pow(a, x) + Math.Pow(y, a);

}

double min = max;

if ((Math.Sqrt(x) + Math.Sqrt(Math.Pow(a, x))) < min);

{

min = (Math.Sqrt(x) + Math.Sqrt(Math.Pow(a, x)));

}

c = min;

}

else if ((x <= a) && (0 <= y ) && (y <= 3.0\*x))

{

double min = a + b;

if (Math.Pow(x, a + b) < min)

{

min = Math.Pow(x, a + b);

}

c = min / 1 + a + Math.Pow(b, a);

}

else

{

c = Math.Pow((Math.Tan(Math.Abs(x + y))), 2);

}

return c;

}

}

}

**Содержание основной части программы**

using System;

using System.Windows.Forms;

using Rabota\_lab;

namespace WindowsFormsApp1

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void button1\_Click\_1(object sender, EventArgs e)

{

double a = Class1.Vvod(textBox1);

double b = Class1.Vvod(textBox2);

double x = Class1.Vvod(textBox3);

double y = Class1.Vvod(textBox4);

double c = Class1.Raschet\_function(a, b, x, y);

Class1.Vivod(textBox5, c);

}

}

}

**Результаты выполнения программы**

На рисунках 5-7 представлены результаты выполненной программы,

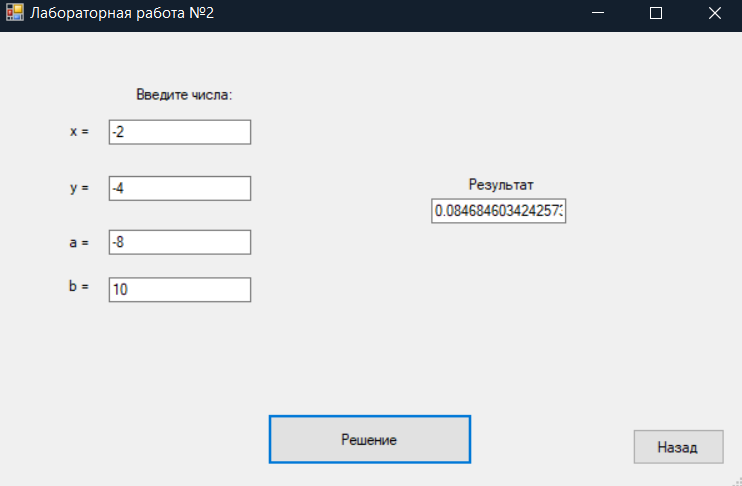


Рисунок 5 – Результат выполненной программы с первым условием

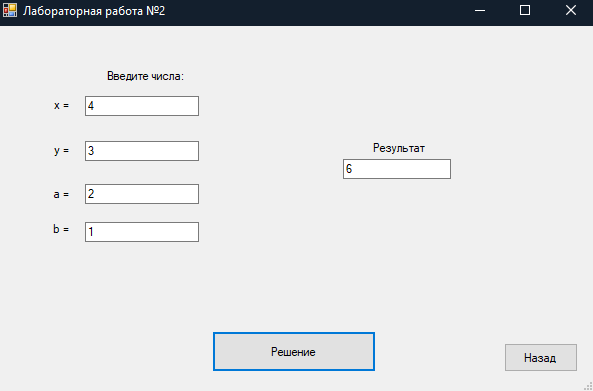


Рисунок 6 – Результат выполненной программы со вторым условием

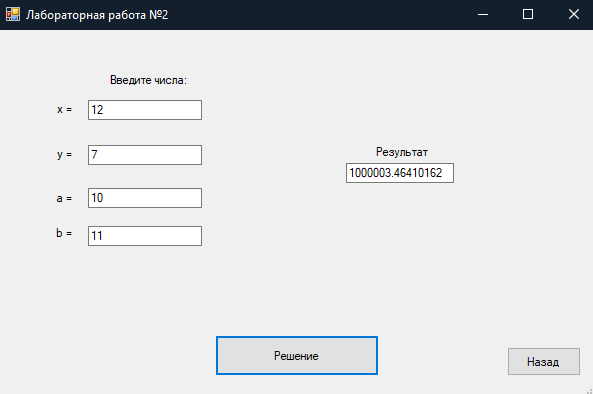


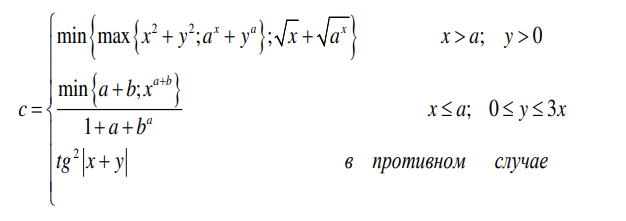
Рисунок 7 – Результат выполненной программы с третьим условием

**Лабораторная работа №3**

**«Табулирование функции»**

**Задание**

**Разработать программу, предназначенную для табулирования функции согласно выданному заданию:**



Вывести сумму, минимум, максимум и произведение из решений функции, а также количество шагов.

**Для того, чтобы выполнить задание, необходимо разработать следующие** **методы:**

1. Метод public static double Vvod(TextBox t), предназначенный для конвертации строкового значения в числовое.
2. Метод public static void Vivod(TextBox t, double b), предназначенный для вывода данных в текстовое меню.
3. Метод public static double Raschet\_function(double x, double y, double a, double b), предназначенный для нахождения значения функции.
4. Метод public static void VivodDGV(double x, double y, DataGridView DGV), предназначенный для отображения данных в табличном формате.
5. Метод public static void Tab(double a1, double b2, double h, double y, double a, double b, DataGridView d), предназначенный для табулирования функции.

Все приведенные методы должны быть размещены в DLL-библиотеке.

**В основной части программы следует последовательно осуществить**

**вызов ранее написанных методов.**

**Перечень блок-схем**

На рисунке 1 изображен алгоритм метода, предназначенного для конвертации строкового значения в числовое.

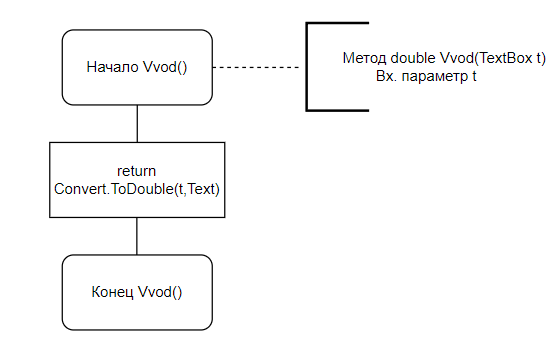


Рисунок 1 – Алгоритм метода, конвертирующего строковое значение в числовое

На рисунке 2 изображен алгоритм метода, предназначенного для вывода данных в текстовое меню.

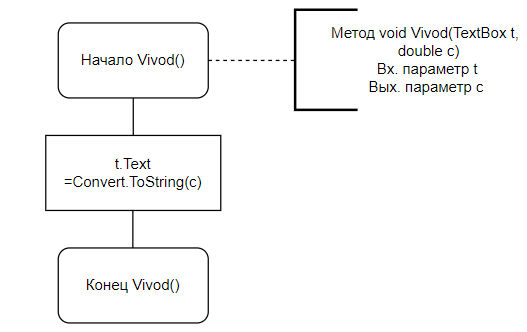


Рисунок 2 - Алгоритм метода, выводящего данные в текстовое меню.

На рисунке 3 изображен алгоритм метода, предназначенного для нахождения значения функции и возращения результата с помощью оператора return.

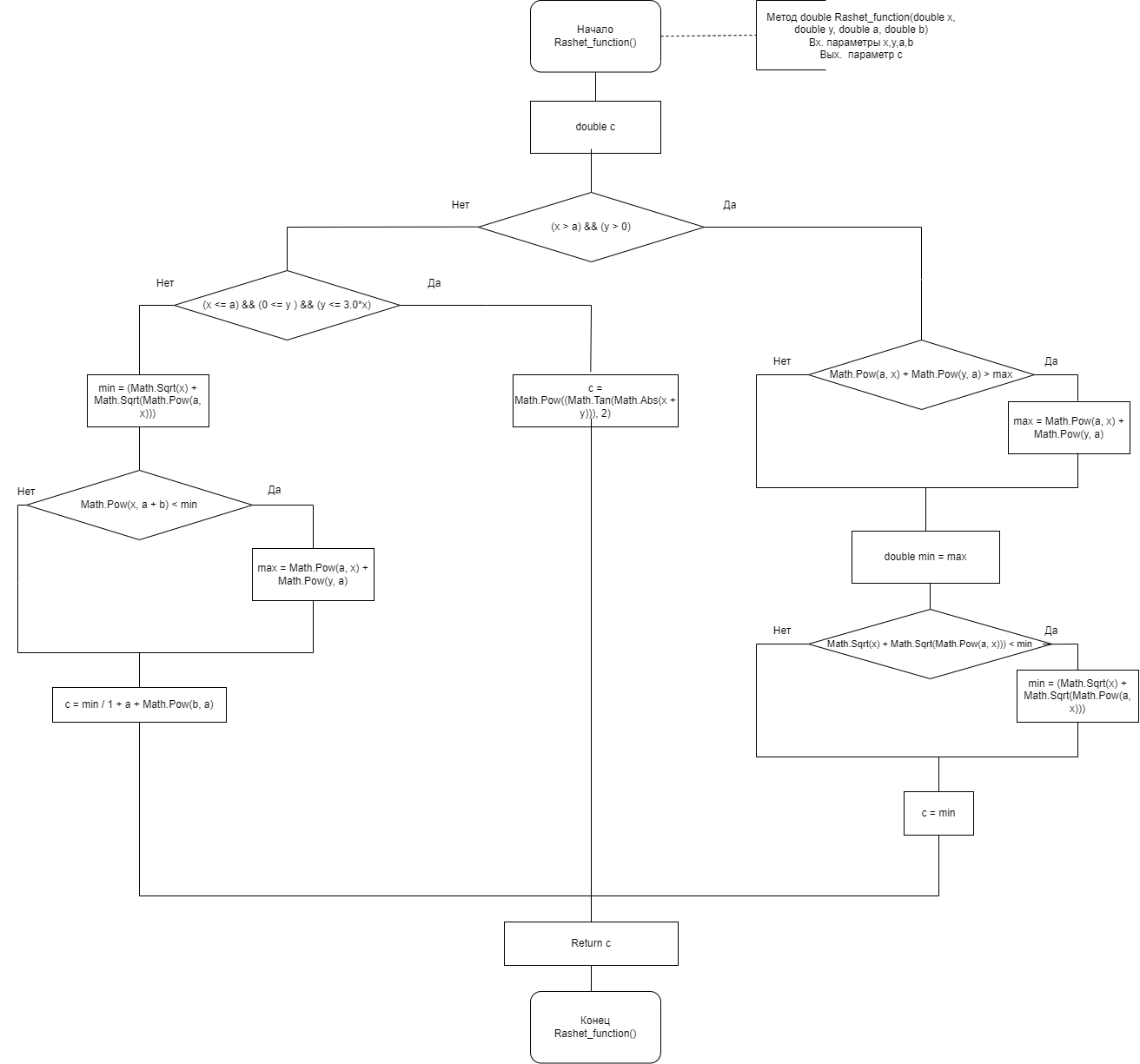


Рисунок 3 - Алгоритм метода, находящий значение функции и возвращающего результат оператором return

На рисунке 4 изображен алгоритм метода, предназначенного для отображения данных в табличном формате.

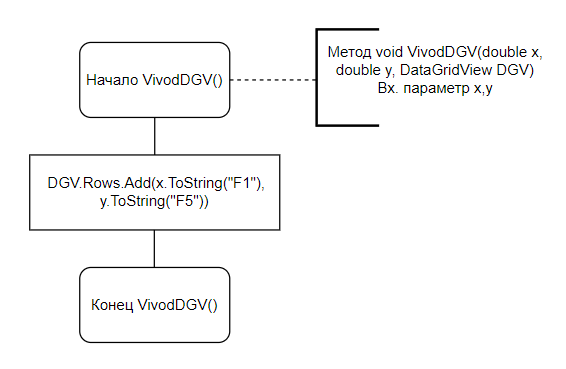


Рисунок 4 – Алгоритм метода, отображающий данные в табличном формате

На рисунке 5 изображен алгоритм метода, предназначенного для табулирования функции.

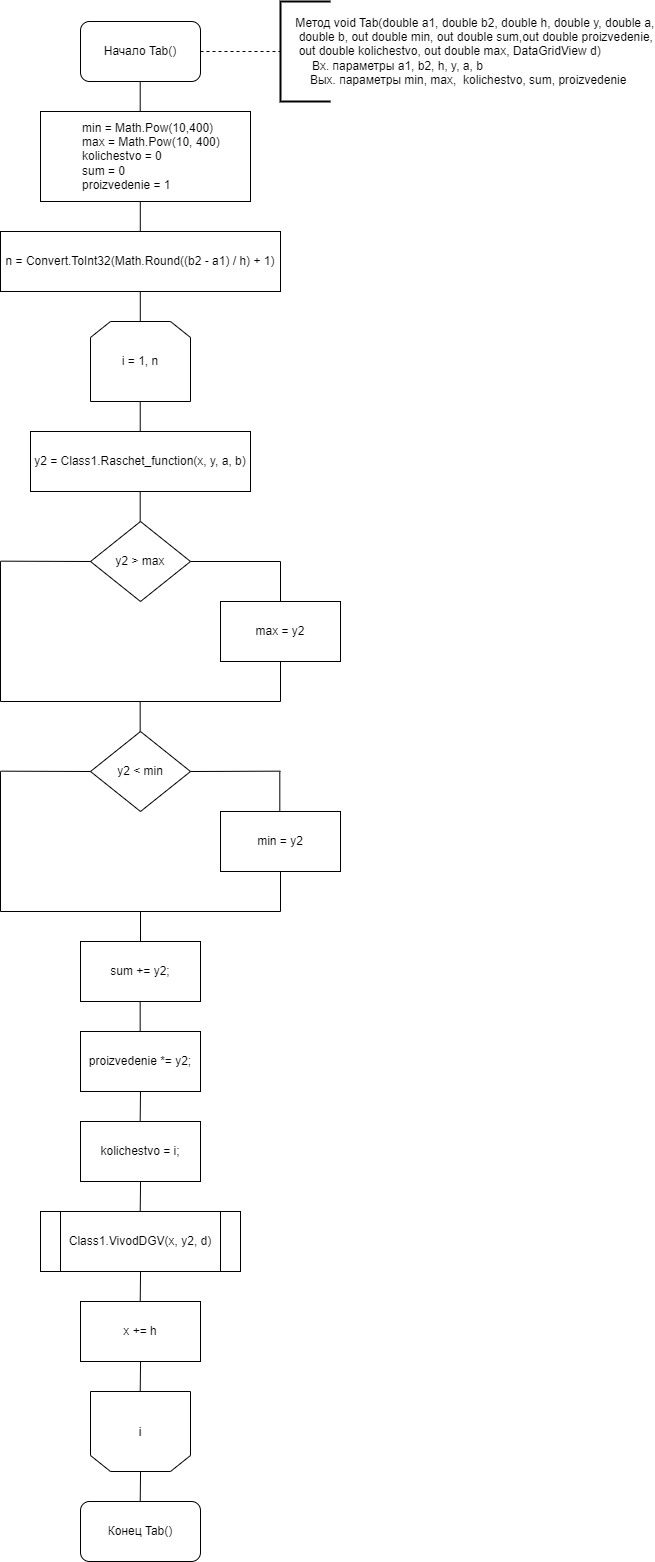


Рисунок 5 – Алгоритм метода, выводящий значения в таблицу

На рисунке 6 изображен алгоритм событийной кнопки.

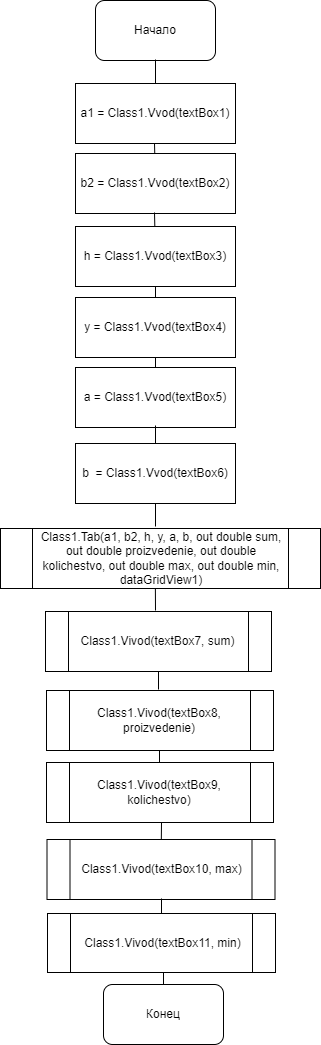


Рисунок 6 – Алгоритм событийной кнопки

**Содержание DLL-библиотеки**

using System;

using System.Windows.Forms;

namespace Rabota\_lab

{

public class Class1

{

public static double Vvod(TextBox t)

{

return Convert.ToDouble(t.Text);

}

public static void Vivod(TextBox t, double c)

{

t.Text = Convert.ToString(c);

}

public static void VivodDGV(double x, double y, DataGridView DGV)

{

DGV.Rows.Add(x.ToString("F1"), y.ToString("F5"));

}

public static double Raschet\_function(double x, double y, double a, double b)

{

double c;

if ((x > a) && (y > 0))

{

double max = Math.Pow(x, 2) + Math.Pow(y, 2);

if (Math.Pow(a, x) + Math.Pow(y, a) > max)

{

max = Math.Pow(a, x) + Math.Pow(y, a);

}

double min = max;

if ((Math.Sqrt(x) + Math.Sqrt(Math.Pow(a, x))) < min);

{

min = (Math.Sqrt(x) + Math.Sqrt(Math.Pow(a, x)));

}

c = min;

}

else if ((x <= a) && (0 <= y ) && (y <= 3.0\*x))

{

double min = a + b;

if (Math.Pow(x, a + b) < min)

{

min = Math.Pow(x, a + b);

}

c = min / 1 + a + Math.Pow(b, a);

}

else

{

c = Math.Pow((Math.Tan(Math.Abs(x + y))), 2);

}

return c;

}

public static void Tab(double a1, double b2, double h, double y, double a, double b, out double min, out double sum, out double proizvedenie, out double kolichestvo, out double max, DataGridView d)

{

double x = a1;

min = Math.Pow(10,400);

max = Math.Pow(10, 400);

kolichestvo = 0;

sum = 0;

proizvedenie = 1;

int n = Convert.ToInt32(Math.Round((b2 - a1) / h) + 1);

for (double i = 1; i <= n; i++)

{

double y2 = Class1.Raschet\_function(x, y, a, b);

if (y2 > max)

{

max = y2;

}

if (y2 < min)

{

min = y2;

}

sum += y2;

proizvedenie \*= y2;

kolichestvo = i;

Class1.VivodDGV(x, y2, d);

x += h;

}

}

}

}

**Содержание основной части программы**

using System;

using System.Windows.Forms;

using Rabota\_lab;

namespace WindowsFormsApp2

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void button1\_Click\_1(object sender, EventArgs e)

{

double a1 = Class1.Vvod(textBox1);

double b2 = Class1.Vvod(textBox2);

double h = Class1.Vvod(textBox3);

double y = Class1.Vvod(textBox4);

double a = Class1.Vvod(textBox5);

double b = Class1.Vvod(textBox6);

Class1.Tab(a1, b2, h, y, a, b, out double sum, out double proizvedenie, out double kolichestvo, out double max, out double min, dataGridView1);

Class1.Vivod(textBox7, sum);

Class1.Vivod(textBox8, proizvedenie);

Class1.Vivod(textBox9, kolichestvo);

Class1.Vivod(textBox10, max);

Class1.Vivod(textBox11, min);

}

}

}

**Результаты выполнения программы**

На рисунке 7 представлен результат выполненной программы, табулирующей функцию согласно выданному заданию.

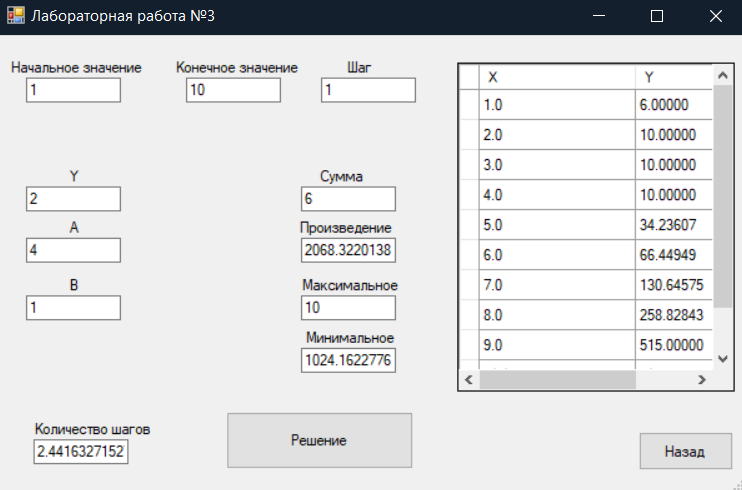


Рисунок 7 – Результат выполненной программы

**Лабораторная работа №4**

**«Создание приложений, использующих итеративные циклические структуры»**

**Задание**

Пусть ; ; k = 1,2.. .Дано действительное число . Найдите первый член , для которого выполнено условие .

**Для того чтобы выполнить задание, необходимо разработать следующие методы:**

1. Метод public static double Vvod(TextBox t), предназначенный для конвертации строкового значения в числовое.
2. Метод public static void Vivod(TextBox t, double b), предназначенный для вывода данных в текстовое меню.
3. Метод public static void VivodDGV(double x, double y, DataGridView DGV) предназначенный для вывода значений в таблицу.
4. Метод public static void LR4\_Tab(double k, DataGridView DGV), предназначенный для вычисления первого члена , рекуррентной формулы , для которого выполнено условие .

Все приведенные методы должны быть размещены в DLL-библиотеке.

**В основной части программы следует последовательно осуществить**

**вызов ранее написанных методов.**

**Перечень блок-схем**

На рисунке 1 изображен алгоритм метода, предназначенного для конвертации строкового значения в числовое.

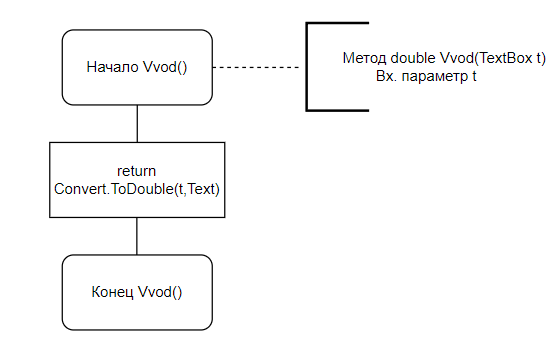


Рисунок 1 – Алгоритм метода, конвертирующего строковое значение в числовое

На рисунке 2 изображен алгоритм метода, предназначенного для вывода данных в текстовое меню.

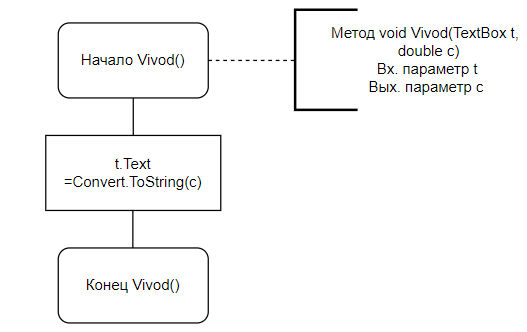


Рисунок 2 - Алгоритм метода, выводящего данные в текстовое меню

На рисунке 3 изображен алгоритм метода, предназначенного для вычисления, предназначенный для вычисления первого члена , рекуррентной формулы , для которого выполнено условие .

.

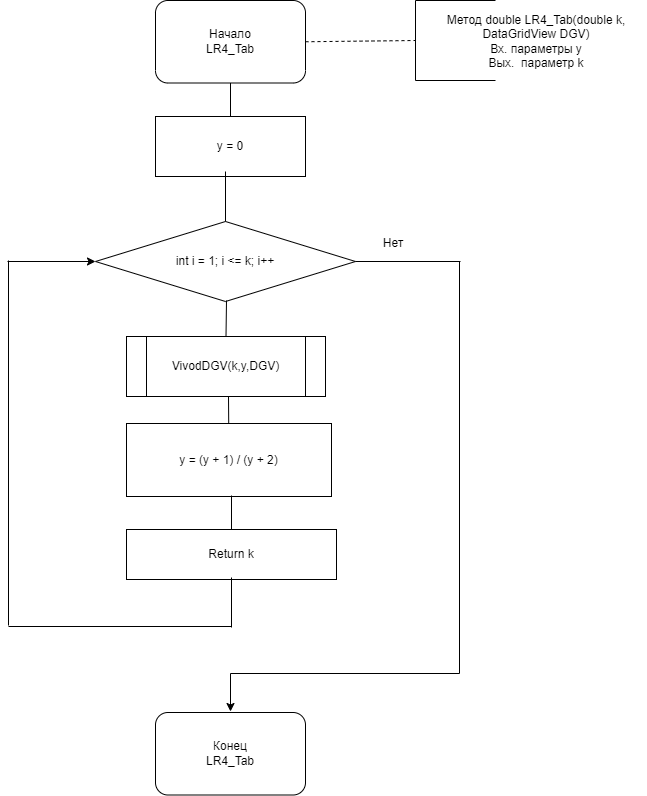


Рисунок 3 - Алгоритм метода, предназначенного для расчета функции

На рисунке 4 изображен алгоритм метода, предназначенного для вывода значений в таблицу.

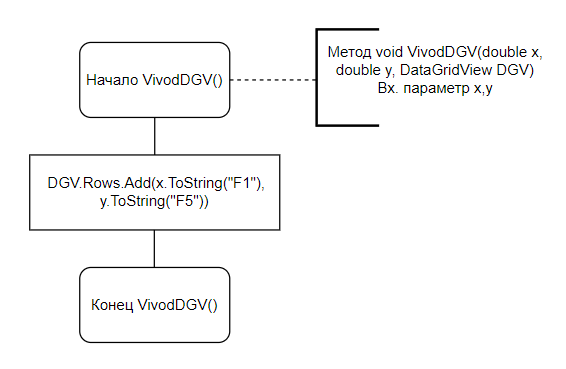


Рисунок 4 – Алгоритм метода, предназначенного для вывода значений в таблицу

На рисунке 5 изображен алгоритм событийной кнопки.

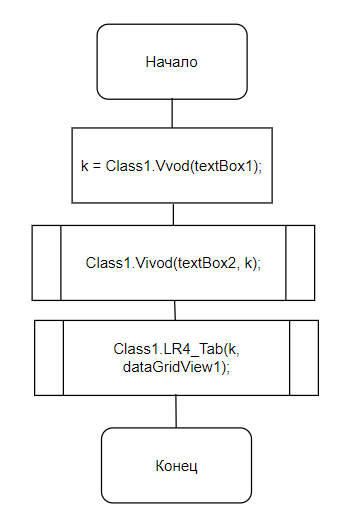


Рисунок 5 – Алгоритм событийной кнопки

**Содержание DLL-библиотеки**

using System;

using System.Windows.Forms;

namespace Rabota\_lab

{

public class Class1

{

public static double Vvod(TextBox t)

{

return Convert.ToDouble(t.Text);

}

public static void Vivod(TextBox t, double c)

{

t.Text = Convert.ToString(c);

}

public static void VivodDGV(double x, double y, DataGridView DGV)

{

DGV.Rows.Add(x.ToString("F1"), y.ToString("F5"));

}

public static double LR4\_Tab(double k, DataGridView DGV)

{

double y = 0;

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

{

VivodDGV(k,y,DGV);

y = (y + 1) / (y + 2);

}

return k;

}

}

}

**Содержание основной части программы**

using Rabota\_lab;

using System;

using System.Windows.Forms;

namespace Forma\_LB

{

public partial class LB4 : Form

{

public LB4()

{

InitializeComponent();

}

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

{

double k = Class1.Vvod(textBox1);

Class1.Vivod(textBox2, k);

Class1.LR4\_Tab(k, dataGridView1);

}

}

}

**Результаты выполнения программы**

На рисунке 6 представлен результат выполнения программы, использующей итеративные циклические структуры.

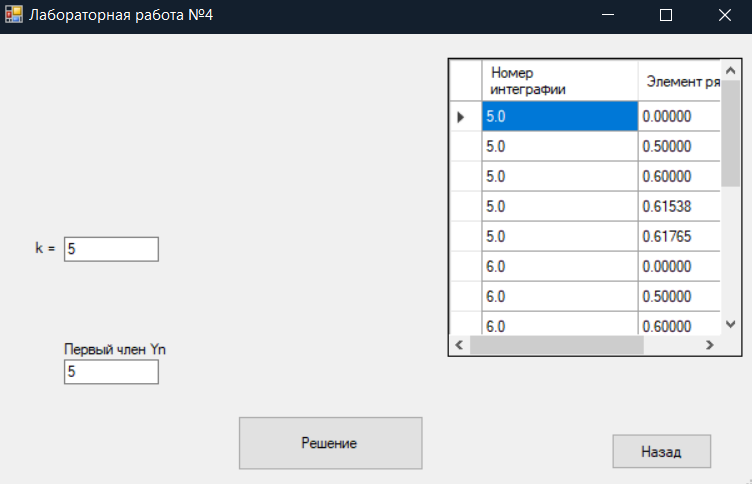


Рисунок 6 – Результат выполнения программы

**Лабораторная работа №5**

**«Работа с одномерными массивами»**

**Задание**

Дан динамический целочисленный массив положительных и отрицательных чисел. Найдите количество пар совпадающих по значению чисел, разработав соответствующий метод. Разработайте метод, который формирует новый массив, из индексов элементов исходного массива, значения которых меньше найденного числа. Для того, чтобы выполнить задание, необходимо разработать следующие методы:

1. Метод public static int InputInt(TextBox t), предназначенный для конвертации строкового значения в числовое.
2. Метод public static void Enter\_mas(int[] mas, int length, int a, int b), предназначенный для генерации массива заданной длины, заполнения элементов значениями в диапазоне (a;b).
3. Метод public static int Task1(int[] arr), предназначенный для нахождения количества пар совпадающих по значению чисел.
4. Метод public static void Output\_mas(int[] mas, int length, DataGridView dgv), предназначенный для вывода массива в таблицу.
5. Метод public static void Set\_mas(int[] mas, int k, out int j, int[] rezmas), предназначенный для формирования нового массива, из индексов элементов исходного массива, значения которых меньше найденного числа.

Данные методы должны быть размещены в заранее разработанной новой DLL библиотеке.

**В основной части программы следует последовательно осуществить**

**вызов ранее написанных методов.**

**Перечень блок-схем**

На рисунке 1 представлен алгоритм метода, конвертирующего строку в число и возвращающий это значение.

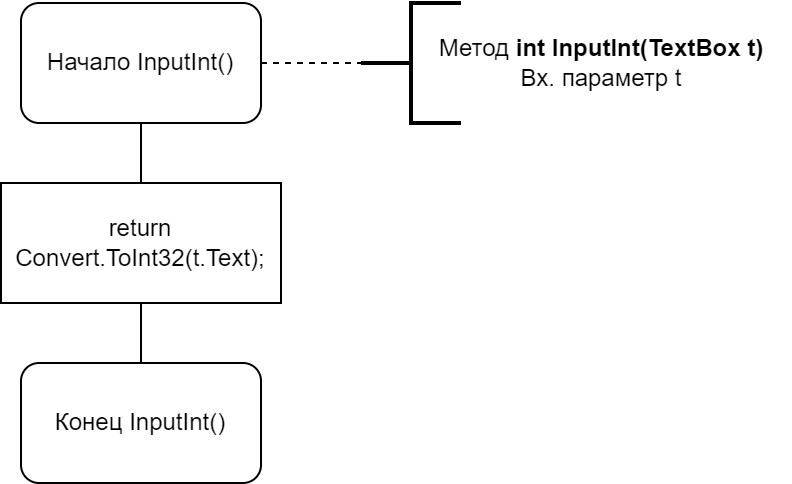


Рисунок 1 – Алгоритм метода, конвертирующего строку в число и возвращающий это значение

На рисунке 2 представлен алгоритм метода, генерирующий массив заданной длины и заполняющий элементы значениями в диапазоне (a;b).

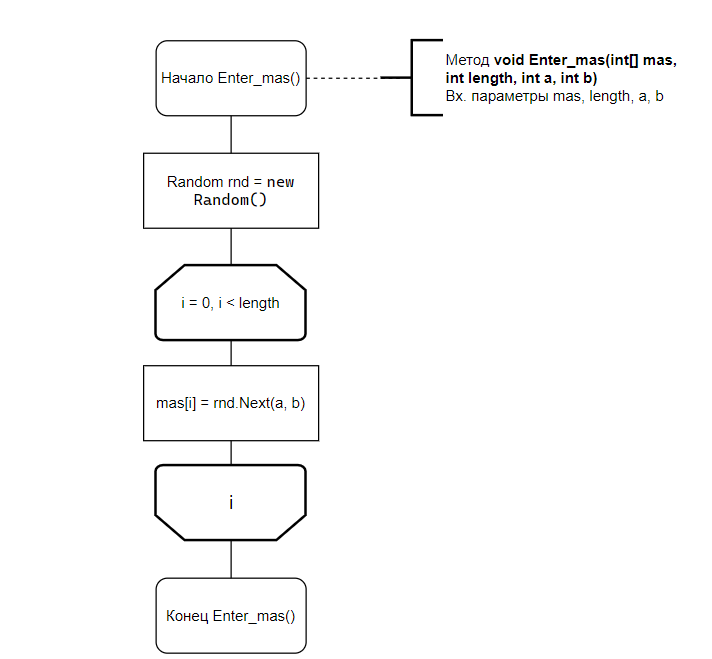


Рисунок 2 - Алгоритм метода, создающий массив заданной длины, заполняя элементами случайного значения в заданном диапазоне

На рисунке 3 представлен алгоритм метода, находящий количество пар совпадающих по значению чисел.

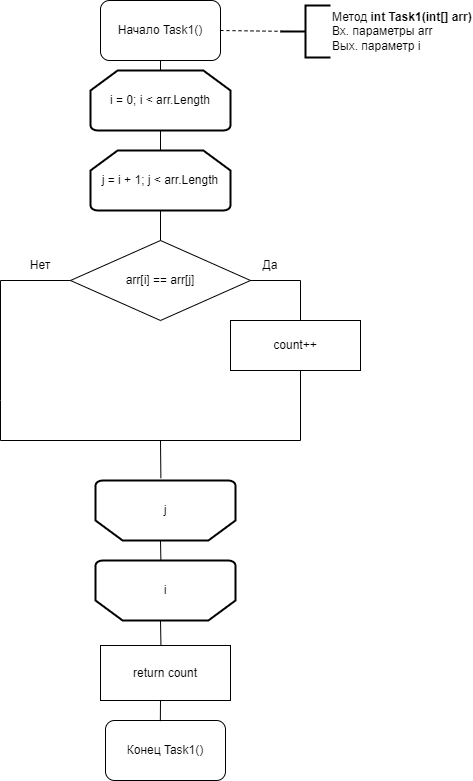


Рисунок 3 – Алгоритм метода, находящий количество пар совпадающих по значению чисел.

На рисунке 4 представлен алгоритм метода, выводящего массив в таблицу.

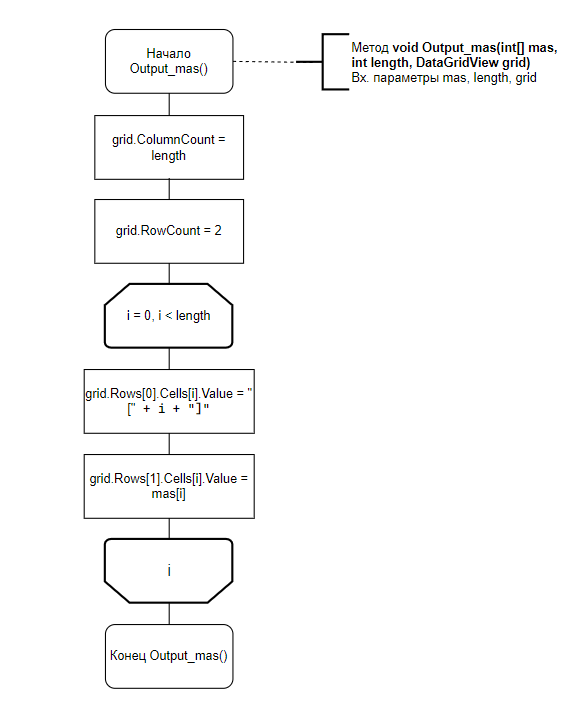


Рисунок 4 - Алгоритм метода, выводящего массив в таблицу.

На рисунке 5 представлен алгоритм метода, формирующий новый массив, из индексов элементов исходного массива, значения которых меньше найденного числа.

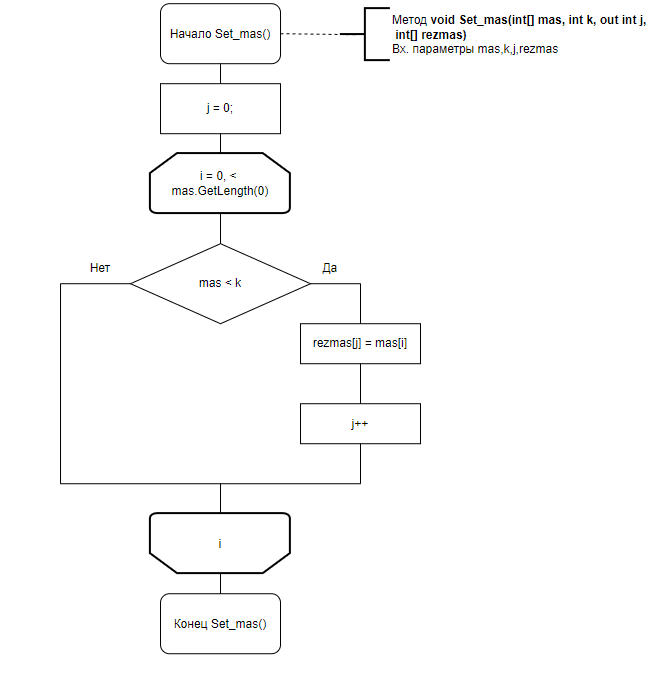


Рисунок 5 - Алгоритм метода, формирующий новый массив, из индексов элементов исходного массива, значения которых меньше найденного числа.

На рисунке 6 представлен алгоритм событийной кнопки.

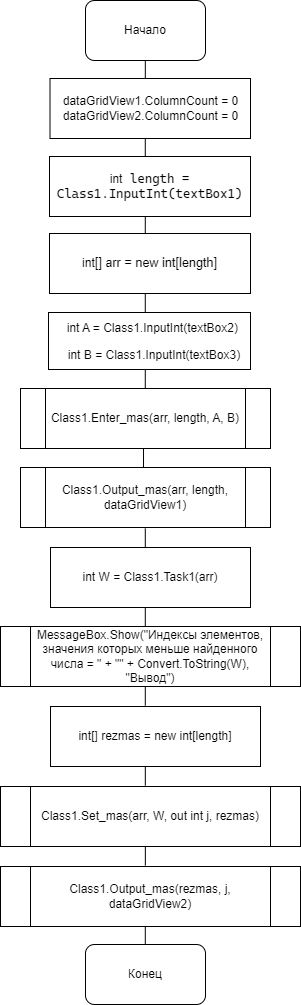


Рисунок 6 – Алгоритм событийной кнопки.

**Содержание DLL-библиотеки**

using System;

using System.Windows.Forms;

namespace Rabota\_Labs

{

public class Class1

{

public static int InputInt(TextBox t)

{

return Convert.ToInt32(t.Text);

}

public static void Enter\_mas(int[] mas, int length, int a, int b)

{

Random rnd = new Random();

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

mas[i] = rnd.Next(a, b);

}

public static int Task1(int[] arr)

{

int count = 0;

for (int i = 0; i < arr.Length; i++)

{

for (int j = i + 1; j < arr.Length; j++)

{

if (arr[i] == arr[j])

{

count++;

}

}

}

return count;

}

public static void Output\_mas(int[] mas, int length, DataGridView dgv)

{

dgv.ColumnCount = length;

dgv.RowCount = 2;

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

{

dgv.Rows[0].Cells[i].Value = "[" + i + "]";

dgv.Rows[1].Cells[i].Value = mas[i];

}

}

public static void Set\_mas(int[] mas, int k, out int j, int[] rezmas)

{

j = 0;

for (int i = 0; i < mas.GetLength(0); i++)

if (mas[i] < k)

{

rezmas[j] = mas[i];

j++;

}

}

}

}

**Содержание основной части программы**

using Rabota\_Labs;

using System;

using System.Windows.Forms;

namespace Forma\_LB

{

public partial class LB5 : Form

{

public LB5()

{

InitializeComponent();

}

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

{

dataGridView1.ColumnCount = 0;

dataGridView2.ColumnCount = 0;

if (textBox1.Text =="")

{

MessageBox.Show("Вы не ввели значение","Вывод", MessageBoxButtons.OK, MessageBoxIcon.Error);

textBox1.Focus();

return;

}

int length = Class1.InputInt(textBox1);

int[] arr = new int[length];

int A = Class1.InputInt(textBox2);

int B = Class1.InputInt(textBox3);

Class1.Enter\_mas(arr, length, A, B);

Class1.Output\_mas(arr, length, dataGridView1);

int W = Class1.Task1(arr);

MessageBox.Show("Индексы элементов, значения которых меньше найденного числа = " + "" + Convert.ToString(W), "Вывод");

int[] rezmas = new int[length];

Class1.Set\_mas(arr, W, out int j, rezmas);

Class1.Output\_mas(rezmas, j, dataGridView2);

}

}

}

\

**Результаты выполнения программы**

На рисунке 7-10 представлены результаты выполнения программы.

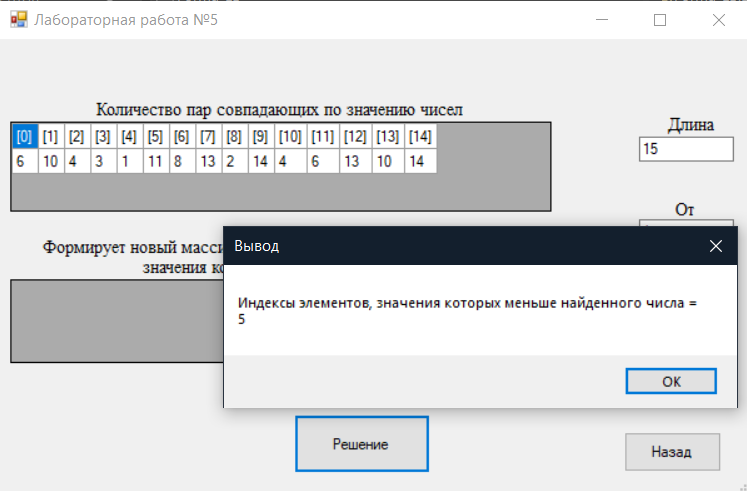


Рисунок 7- Результат №1.

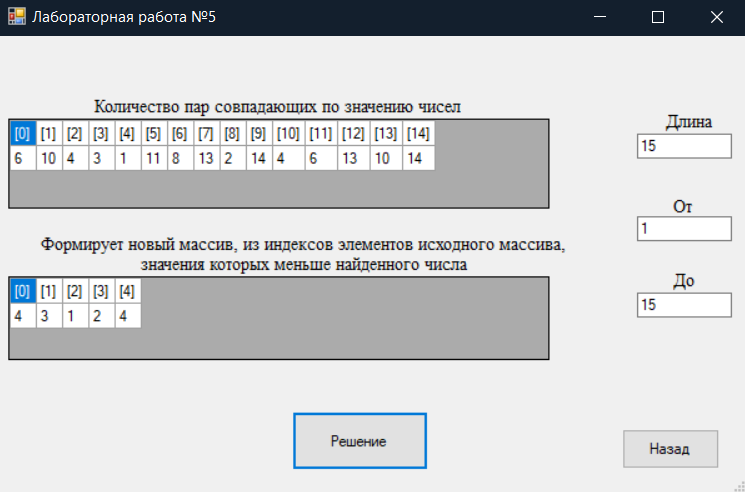


Рисунок 8- Результат №2.

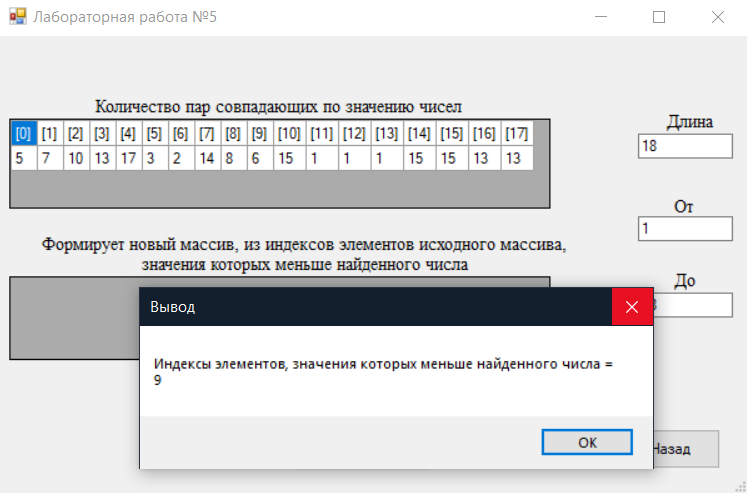


Рисунок 9- Результат №3.

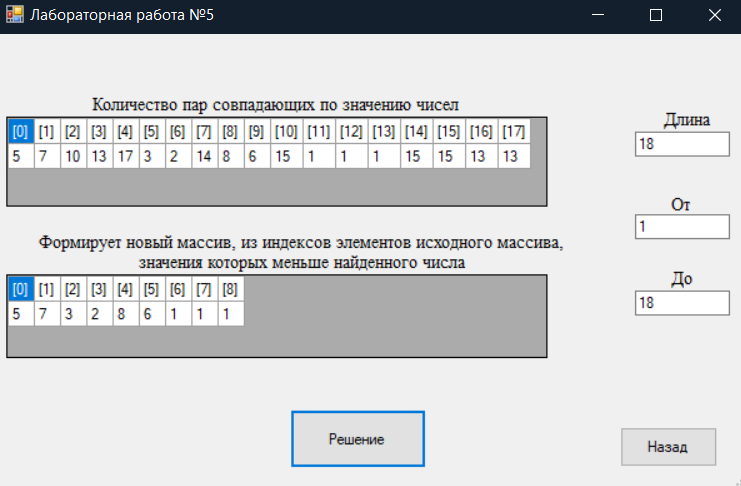


Рисунок 10- Результат №4.

**Лабораторная работа №6**

**«Базовые алгоритмы обработки одномерных массивов и алгоритмы их сортировки»**

**Задание**

Дан динамический целочисленный массив положительных и отрицательных чисел. Найдите количество пар совпадающих по значению чисел, разработав соответствующий метод. Разработайте метод, который формирует новый массив, из индексов элементов исходного массива, значения которых меньше найденного числа.

Необходимо проанализировать одномерный массив на выполнение критерия по нахождению первого отрицательного элемента, первого положительного числа, а также удаление наибольшего элемента массива. Выполнить вставку числа в массив, определить монотонно убывающую последовательность и сделать шейкер-сортировку. Выполнить вывод массива в файл PDF и блокнот.

**Для того, чтобы выполнить задание, необходимо разработать следующие методы:**

1. Метод public static int InputInt(TextBox t), предназначенный для конвертации строкового значения в числовое.

2. Метод public static void Enter\_mas(int[] mas, int length, int a, int b), предназначенный для генерации массива, расположить в DLL-библиотеке.

3. Метод public static void Output\_mas(int[] mas, int lenght, DataGridView DGV), предназначенный для вывода массива в DataGridView, расположить в DLL-библиотеке.

4. Метод public static int Task1(int[] arr), предназначенный для нахождения количества пар совпадающих по значению чисел, расположить в DLL-библиотеке.

5. Метод public static void Set\_mas(int[] mas, int k, out int j, int[] rezmas), предназначенный для вывода результирующего массива, расположить в DLL-библиотеке.

6. Метод public static void Add\_pdf(int[] mas, int length), предназначенный для создания PDF-файла, содержащего таблицу со значениями входного массива, расположить в DLL-библиотеке.

7. Метод public static void ZapisBloknot(int[] mas, int lenght), предназначенный для создания блокнот-файла, содержащего список со значениями входного массива, расположить в DLL-библиотеке.

8. Метод public static int[] Task2(int[] mas, int length), предназначенный для удаления наибольшего элемента массива, расположить в DLL-библиотеке.

9. Метод public static int[] Task3(int[] arr, int length, int m, int k), предназначенный для вставки числа в массив, расположить в DLL-библиотеке.

10. Метод public static int Task4(int[] mas, int length), предназначенный для определения монотонно убывающей последовательности, расположить в DLL-библиотеке.

11. Метод public static int Task5(int[] mas, int length), предназначенный для нахождения первого отрицательного элемента, расположить в DLL-библиотеке.

12. public static int Task7(int[] mas, int length), предназначенный для нахождения первого положительного элемента, расположить в DLL-библиотеке.

13. Метод public static int[] Task14(int[] mas, int length), предназначенный для нахождения шейкер-сортировки, расположить в DLL-библиотеке.

**Перечень блок-схем**

На рисунке 1 представлен алгоритм метода, конвертирующего строку в число и возвращающий это значение.

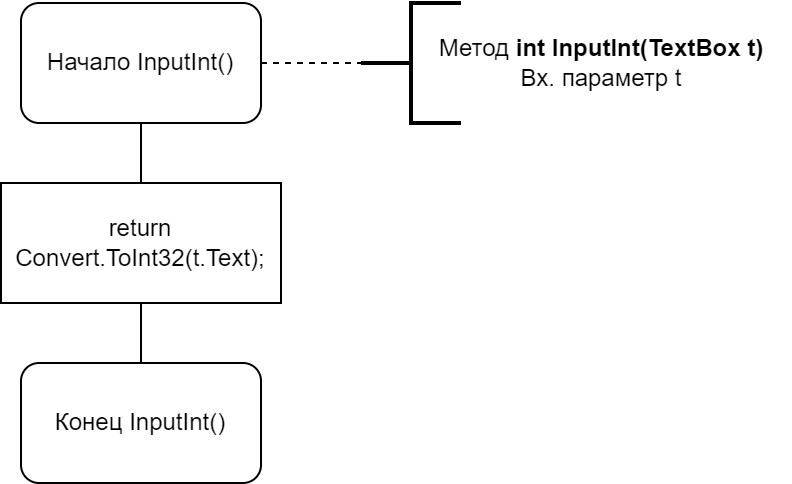


Рисунок 1 – Алгоритм метода, конвертирующего строку в число и возвращающий это значение

На рисунке 2 представлен алгоритм метода, генерирующий массив заданной длины и заполняющий элементы значениями в диапазоне (a;b).

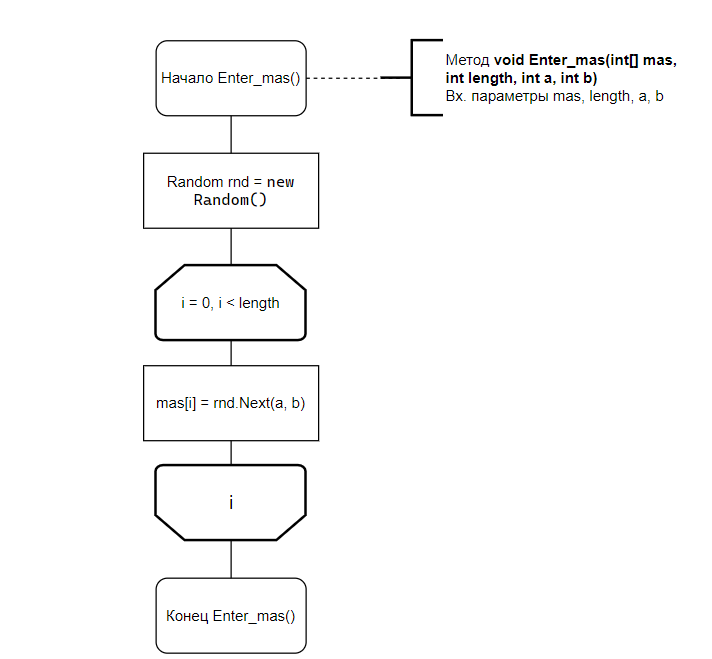


Рисунок 2 - Алгоритм, предназначенный для генерации массива

На рисунке 3 представлен алгоритм метода, выводящего массив в таблицу.

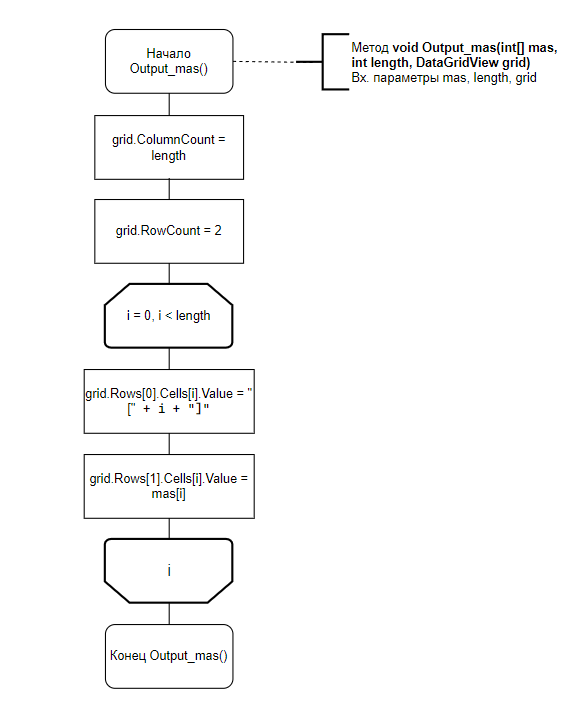


Рисунок 3 - Алгоритм, предназначенный для вывода массива в DataGridView

На рисунке 4 представлен алгоритм метода, находящий количество пар совпадающих по значению чисел.

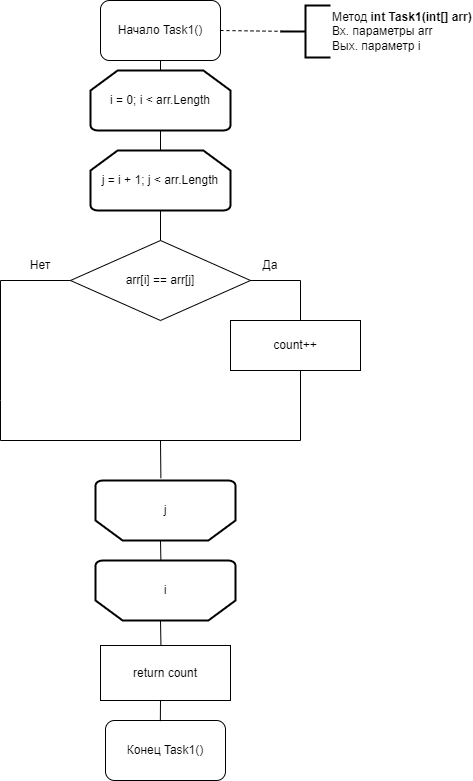


Рисунок 4 – Алгоритм метода, находящий количество пар совпадающих по значению чисел.

На рисунке 5 представлен алгоритм метода, формирующий новый массив, из индексов элементов исходного массива, значения которых меньше найденного числа.

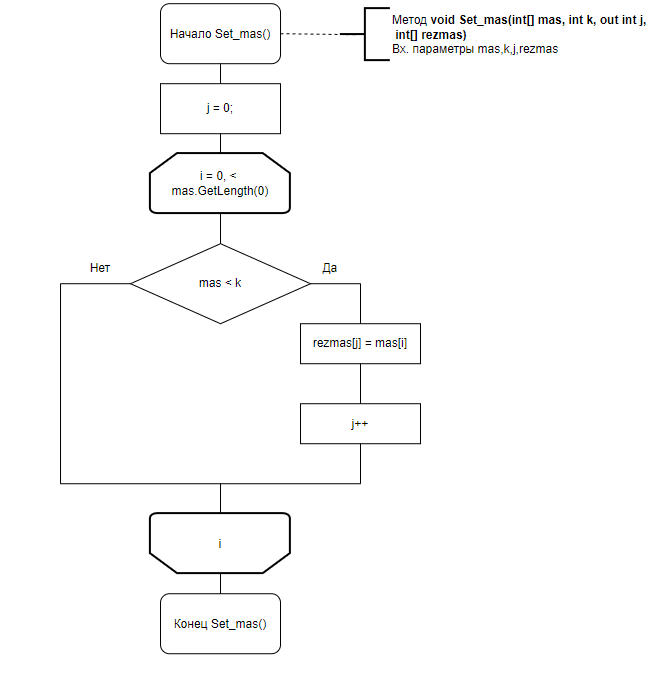


Рисунок 5 - Алгоритм метода, формирующий новый массив, из индексов элементов исходного массива, значения которых меньше найденного числа

На рисунке 6 представлен алгоритм метода, удаляющий наибольший элемент массива.

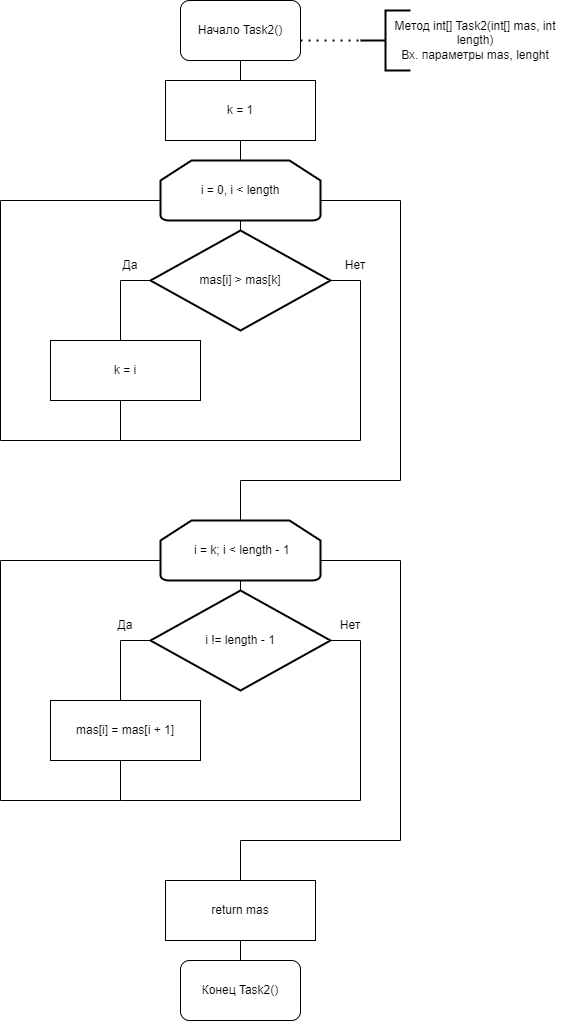


Рисунок 6 - Алгоритм, предназначенный для удаления наибольшего элемента массива

На рисунке 7 представлен алгоритм метода, предназначенный для вставки числа в массив.

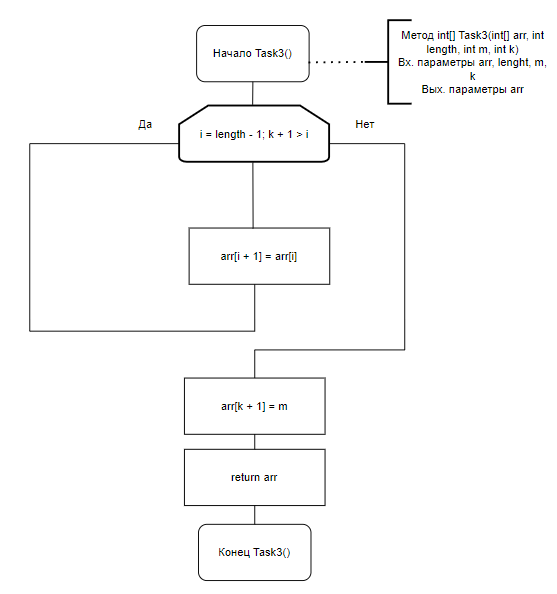


Рисунок 7 - Алгоритм, предназначенный для вставки числа в массив

На рисунке 8 представлен алгоритм метода, для определения монотонно убывающей последовательности.

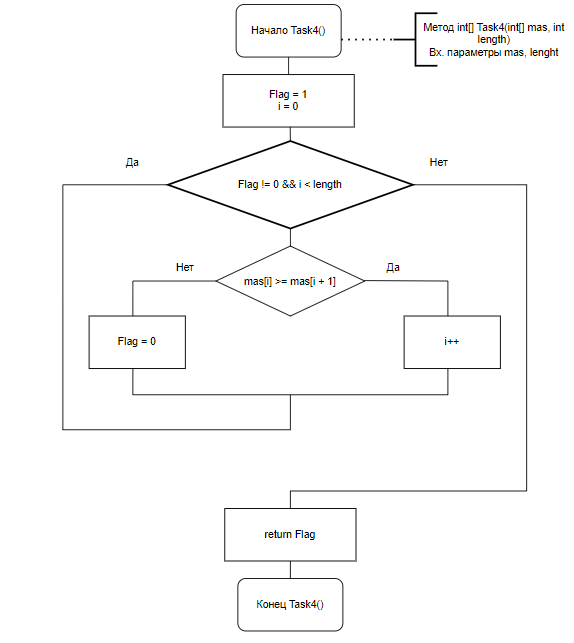


Рисунок 8 - Алгоритм, предназначенный для монотонно убывающей последовательности

На рисунке 9 представлен алгоритм метода, для нахождения первого отрицательного элемента.

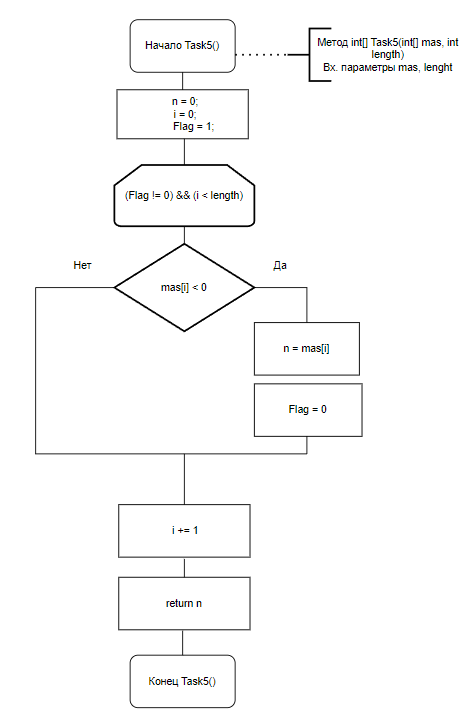


Рисунок 9 - Алгоритм, предназначенный для нахождения первого отрицательного элемента

На рисунке 10 представлен алгоритм метода, для нахождения первого положительного элемента.

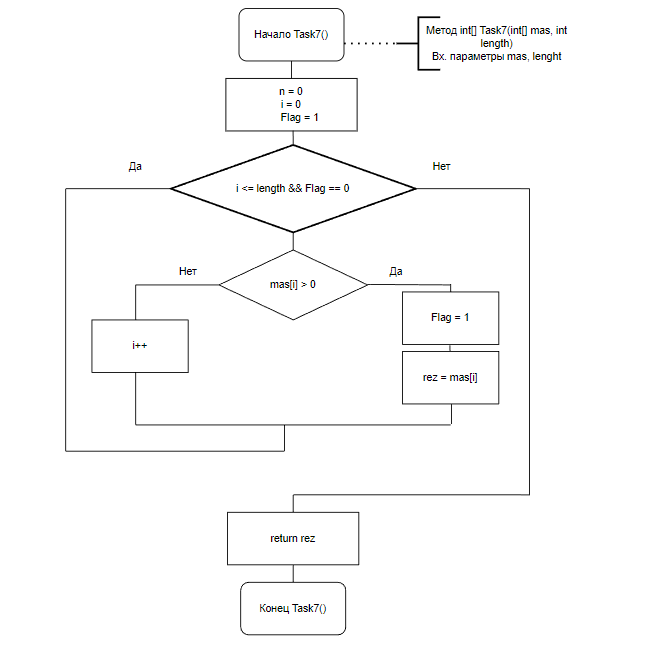


Рисунок 10 - Алгоритм, предназначенный для нахождения первого положительного элемента

На рисунке 11 представлен алгоритм метода, для нахождения шейкер-сортировки.

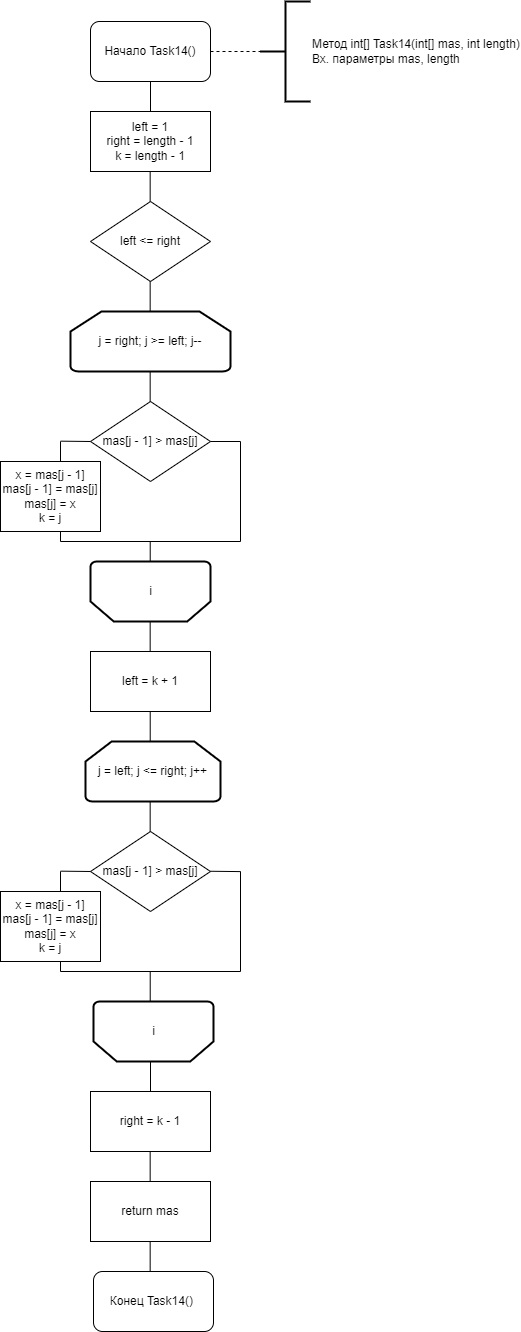


Рисунок 11 - Алгоритм, предназначенный для нахождения шейкер-сортировки

На рисунке 12 представлен алгоритм событийной кнопки.

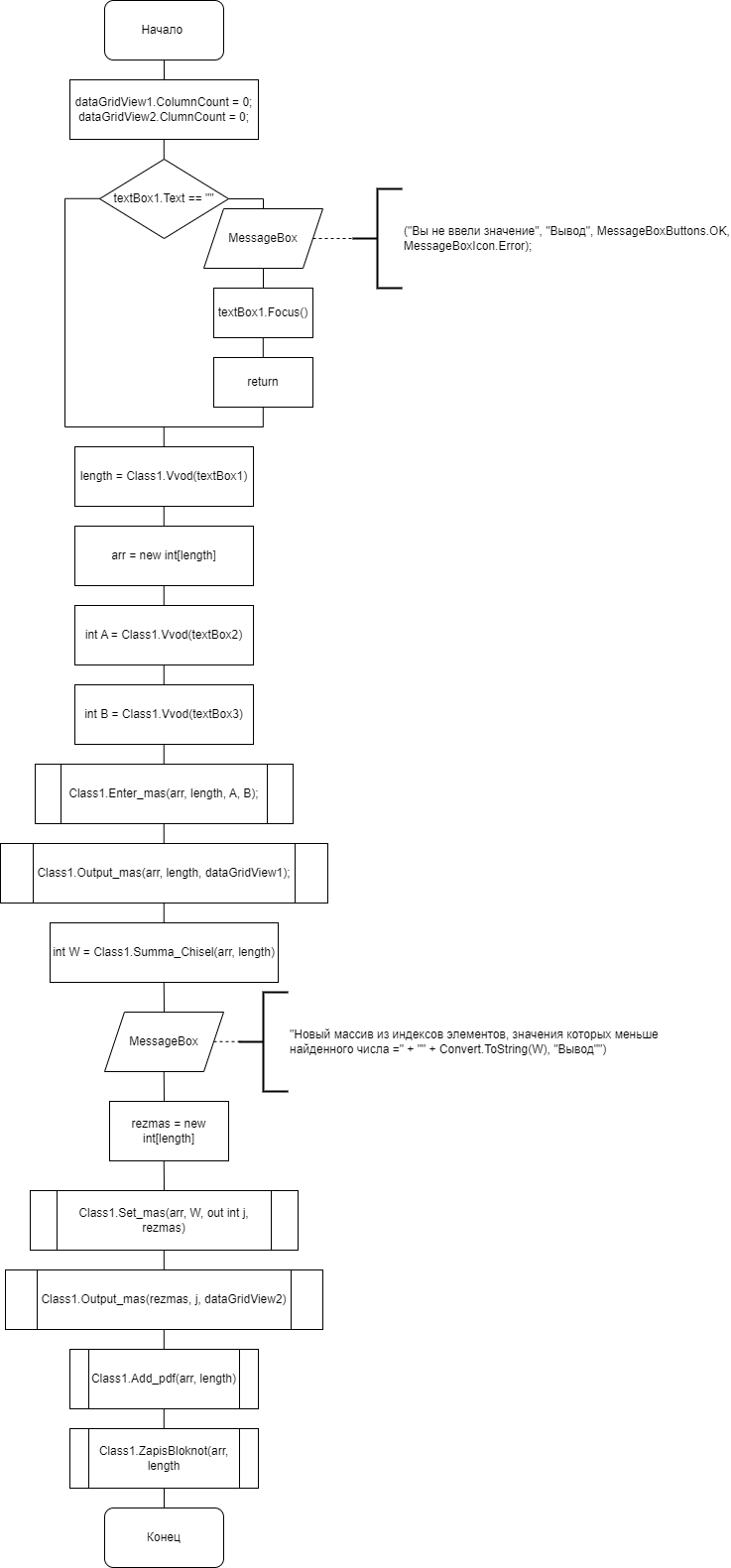


Рисунок 12 – Алгоритм событийной кнопки

**Содержание DLL-библиотеки.**

using System;

using System.Data.OleDb;

using System.IO;

using System.Windows.Forms;

using iTextSharp.text;

using iTextSharp.text.pdf;

using Excel = Microsoft.Office.Interop.Excel;

namespace Rabota\_Labs

{

public class Class1

{

public static int InputInt(TextBox t)

{

return Convert.ToInt32(t.Text);

}

public static void Enter\_mas(int[] mas, int length, int a, int b)

{

Random rnd = new Random();

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

mas[i] = rnd.Next(a, b);

}

public static int Task1(int[] arr)

{

int count = 0;

for (int i = 0; i < arr.Length; i++)

{

for (int j = i + 1; j < arr.Length; j++)

{

if (arr[i] == arr[j])

{

count++;

}

}

}

return count;

}

public static void Output\_mas(int[] mas, int length, DataGridView dgv)

{

dgv.ColumnCount = length;

dgv.RowCount = 2;

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

{

dgv.Rows[0].Cells[i].Value = "[" + i + "]";

dgv.Rows[1].Cells[i].Value = mas[i];

}

}

public static void Set\_mas(int[] mas, int k, out int j, int[] rezmas)

{

j = 0;

for (int i = 0; i < mas.GetLength(0); i++)

if (mas[i] < k)

{

rezmas[j] = mas[i];

j++;

}

}

public static void Add\_pdf(int[] mas, int lenght)

{

var document = new Document();

var zap = PdfWriter.GetInstance(document, new System.IO.FileStream("ZapPDF.pdf", System.IO.FileMode.Create));

document.Open();

var Shrift = BaseFont.CreateFont(@"C:\WINDOWS\Fonts\times.ttf", "CP1251", BaseFont.EMBEDDED);

var font = new Font(Shrift, 10, Font.NORMAL, BaseColor.BLUE);

var tabl = new PdfPTable(1);

var cell = new PdfPCell();

cell.HorizontalAlignment = Element.ALIGN\_LEFT;

cell.Colspan = 2;

cell.Border = 0;

cell.FixedHeight = 16.0F;

cell.Phrase = new Phrase("Исходный массив", font);

tabl.AddCell(cell);

cell.BackgroundColor = BaseColor.WHITE;

cell.Colspan = 1;

cell.Border = 15;

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

{

cell.Phrase = new Phrase(mas[i].ToString(), font);

tabl.AddCell(cell);

}

tabl.TotalWidth = document.PageSize.Width - 500;

tabl.WriteSelectedRows(0, -1, 150, 750, zap.DirectContent);

document.Close();

zap.Close();

System.Diagnostics.Process.Start("msedge.exe", System.IO.Directory.GetCurrentDirectory() + @"\ZapPDF.pdf");

}

public static void ZapisBloknot(int[] mas, int lenght)

{

StreamWriter rez = File.CreateText("Исходный массив.txt");

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

{

rez.WriteLine(mas[i]);

}

rez.Close();

System.Diagnostics.Process.Start("Исходный массив.txt");

}

public static int[] Task2(int[] mas, int length) // Удаление наибольшего элемента массива

{

int k = 1;

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

{

if (mas[i] > mas[k])

k = i;

}

for (int i = k; i < length - 1; i++)

{

if (i != length - 1)

mas[i] = mas[i + 1];

}

return mas;

}

public static int[] Task3(int[] arr, int length, int m, int k) // Втавка числа в массив

{

for (int i = length - 1; k + 1 > i; i--)

{

arr[i + 1] = arr[i];

}

arr[k + 1] = m;

return arr;

}

public static int Task4(int[] mas, int length) // Определение монотонно убывающей последовательности

{

int Flag = 1;

int i = 0;

while (i <= length - 1 && Flag == 1)

{

if (mas[i] >= mas[i + 1])

i++;

else

Flag = 0;

}

return Flag;

}

public static int Task5(int[] mas, int length) // Нахождение первого отрицательного элемента

{

int n = 0;

int i = 0;

int Flag = 1;

while ((Flag != 0) && (i < length))

{

if (mas[i] < 0)

{

n = mas[i];

Flag = 0;

}

i += 1;

}

return n;

}

public static int Task7(int[] mas, int length) // Нахождение первого положительного элемента

{

int i = 0;

int Flag = 0;

int rez = 0;

while ((i <= length) && (Flag == 0))

{

if (mas[i] > 0)

{

Flag = 1;

rez = mas[i];

}

else

i++;

}

return rez;

}

public static int[] Task14(int[] mas, int length) // Шейкер-сортировка

{

int left = 1;

int right = length - 1;

int k = length - 1;

while (left <= right)

{

for (int j = right; j >= left; j--)

{

if (mas[j - 1] > mas[j])

{

int x = mas[j - 1];

mas[j - 1] = mas[j];

mas[j] = x;

k = j;

}

}

left = k + 1;

for (int j = left; j <= right; j++)

{

if (mas[j - 1] > mas[j])

{

int x = mas[j - 1];

mas[j - 1] = mas[j];

mas[j] = x;

k = j;

}

}

right = k - 1;

}

return mas;

}

}

}

**Содержание основной части программы.**

using System;

using Rabota\_Labs;

using System.Windows.Forms;

using iTextSharp.text;

using iTextSharp.text.pdf;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace Forma\_LB

{

public partial class LB6 : Form

{

public LB6()

{

InitializeComponent();

}

private static int[] arr;

private static int length;

private static int k;

private static int m;

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

{

List\_Rabot f = new List\_Rabot();

this.Hide();

f.ShowDialog();

}

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

{

{

dataGridView1.ColumnCount = 0;

dataGridView2.ColumnCount = 0;

if (textBox1.Text == "")

{

MessageBox.Show("Вы не ввели значение", "Вывод", MessageBoxButtons.OK, MessageBoxIcon.Error);

textBox1.Focus();

return;

}

length = Class1.InputInt(textBox1);

arr = new int[length];

int A = Class1.InputInt(textBox2);

int B = Class1.InputInt(textBox3);

Class1.Enter\_mas(arr, length, A, B);

Class1.Output\_mas(arr, length, dataGridView1);

int W = Class1.Task1(arr);

MessageBox.Show(Convert.ToString(W), "Вывод");

int[] rezmas = new int[length];

Class1.Set\_mas(arr, W, out int j, rezmas);

Class1.Output\_mas(rezmas, j, dataGridView2);

Class1.Add\_pdf(arr, length);

Class1.ZapisBloknot(arr, length);

}

}

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

{

int[] array = Class1.Task2(arr, length);

Class1.Output\_mas(array, length - 1, dataGridView2);

}

private void button6\_Click(object sender, EventArgs e)

{

int res = Class1.Task5(arr, length);

if (res != 0)

MessageBox.Show("Результат:" + " " + res, "Вывод");

else

MessageBox.Show("Ошибка,положительное число", "Вывод");

}

private void button7\_Click(object sender, EventArgs e)

{

int res = Class1.Task7(arr, length);

if (res != 0)

MessageBox.Show("Результат:" + " " + res, "Вывод");

else

MessageBox.Show("Ошибка,отрицательное число", "Вывод");

}

private void button8\_Click(object sender, EventArgs e)

{

int[] array = Class1.Task14(arr, length);

Class1.Output\_mas(array, length, dataGridView2);

}

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

{

int flag = Class1.Task4(arr, length);

if (flag == 1)

MessageBox.Show("Убывает монотонно");

else

MessageBox.Show("Не убывает");

}

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

{

int[] array = Class1.Task3(arr, length, m, k);

Class1.Output\_mas(array, length, dataGridView2);

}

}

}

**Результаты выполнения программы**

На рисунке 13-20 представлены результаты выполнения программы

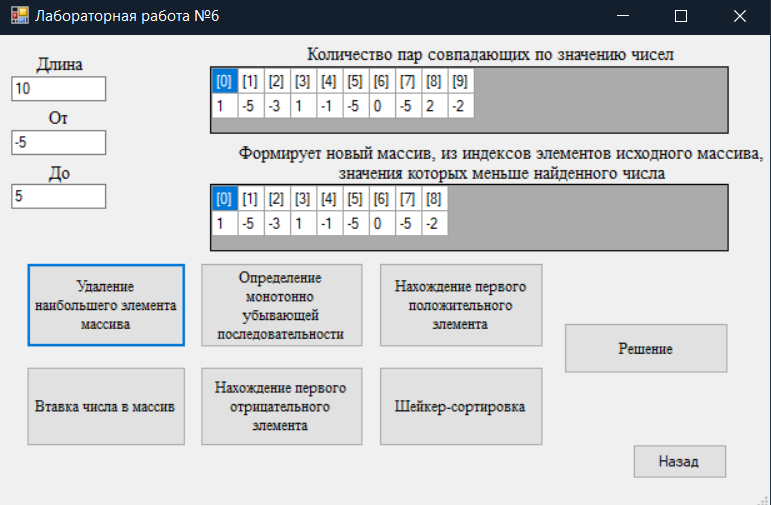


Рисунок 13- Результат №1 Удаление наибольшего элемента массива.

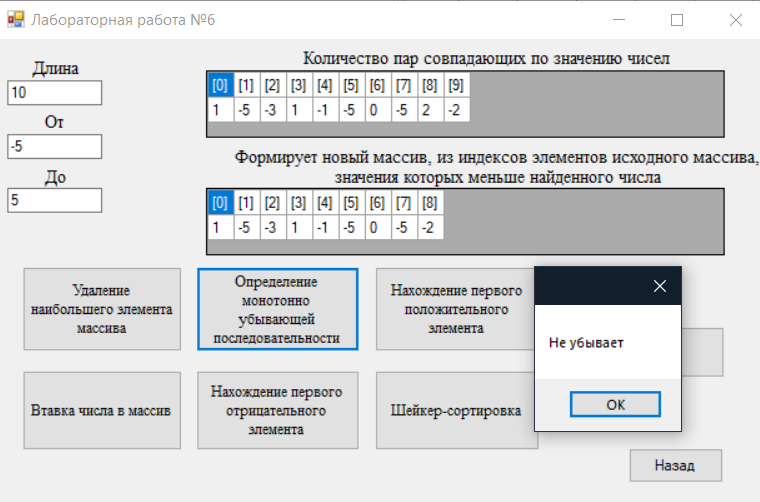


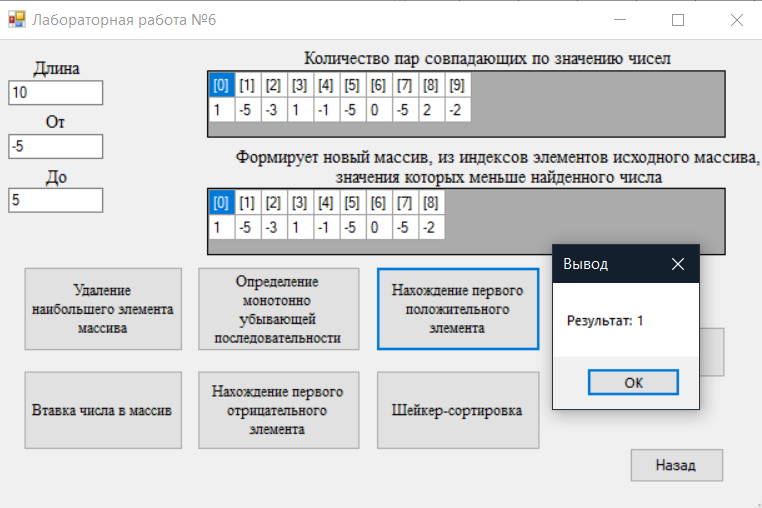
Рисунок 14- Результат №2 Определение монотонно убывающей последовательности.

Рисунок 15- Результат №3 Нахождение первого положительного элемента.

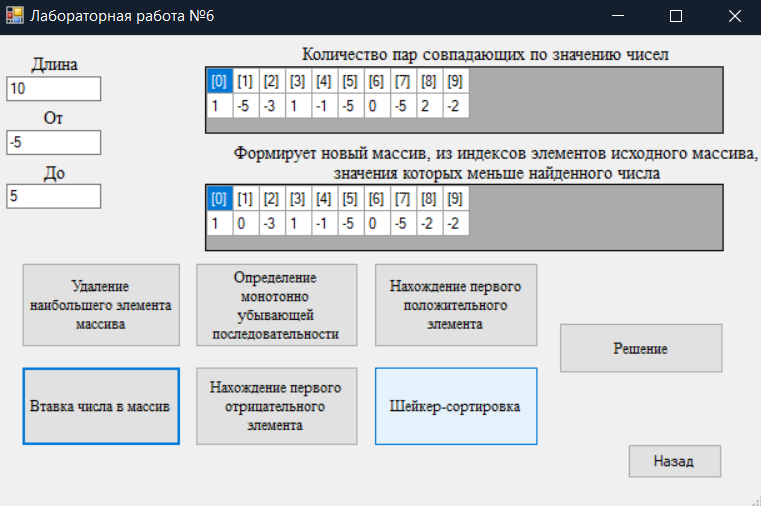


Рисунок 16- Результат №4 Вставка числа в массив.

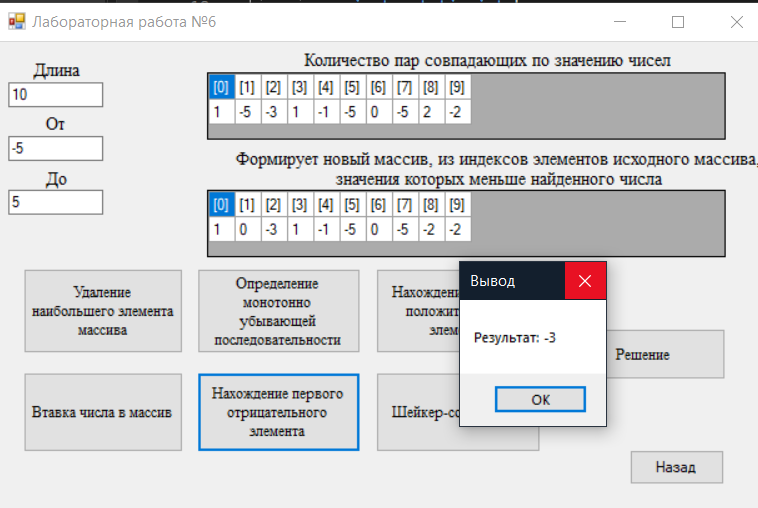


Рисунок 17- Результат №5 Нахождение первого отрицательного элемента

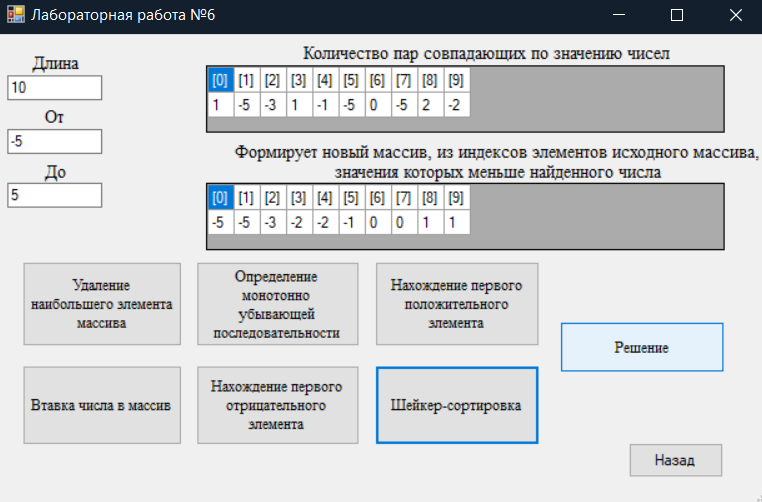


Рисунок 18- Результат №6 Шейкер-сортировка.

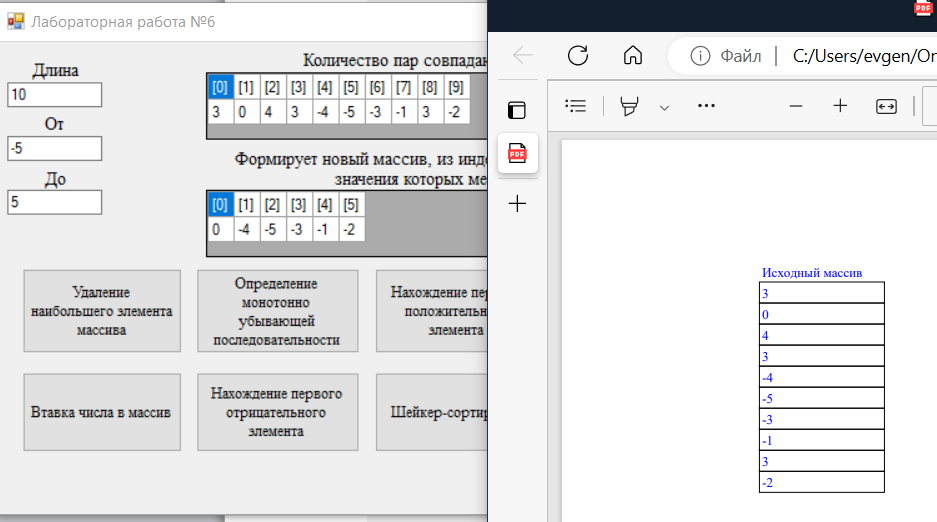


Рисунок 19- Результат №6 Вывод в PDF-файл.

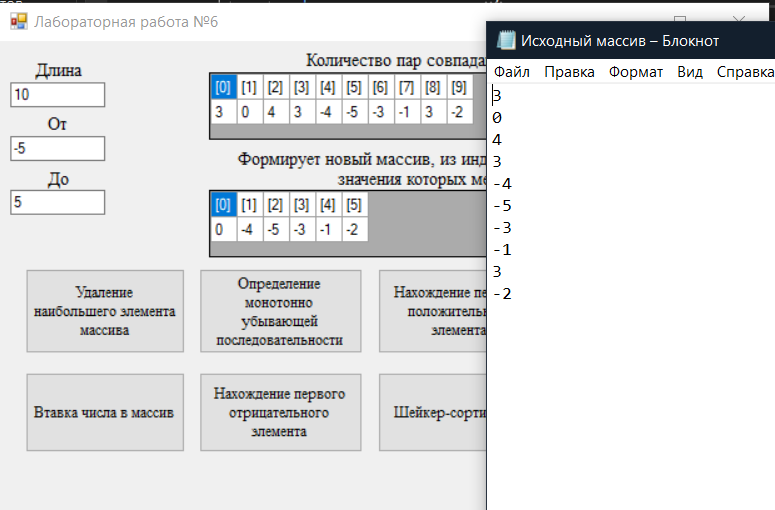


Рисунок 20- Результат №6 Вывод в блокнот

**Лабораторная работа №7**

**«Запись одномерных массивов в программы прикладного назначения»**

**Задание**

Дан динамический целочисленный массив положительных и отрицательных чисел. Найдите количество пар совпадающих по значению чисел, разработав соответствующий метод. Разработайте метод, который формирует новый массив, из индексов элементов исходного массива, значения которых меньше найденного числа.

Необходимо проанализировать одномерный массив на выполнение критерия по нахождению первого отрицательного элемента, первого положительного числа, а также удаление наибольшего элемента массива. Выполнить вставку числа в массив, определить монотонно убывающую последовательность и сделать шейкер-сортировку. Выполнить вывод массива в файл PDF и блокнот.

Кроме того, необходимо выполнить запись одномерных массивов в программы прикладного назначения, а именно: текстовый редактор Microsoft Word, базу данных Microsoft Access, табличный процессор Microsoft Excel. Осуществить написание макроса, предназначенного для созданной таблицы, а также создать кнопку на листе Microsoft Excel, по нажатию на которую, будет осуществляться изменение цвета ячеек таблицы, шрифт и размер текста.

**Для того, чтобы выполнить задание, необходимо разработать следующие методы:**

1. Метод public static int InputInt(TextBox t), предназначенный для конвертации строкового значения в числовое.
2. Метод public static void Enter\_mas(int[] mas, int length, int a, int b), предназначенный для генерации массива, расположить в DLL-библиотеке.
3. Метод public static void Output\_mas(int[] mas, int lenght, DataGridView DGV), предназначенный для вывода массива в DataGridView, расположить в DLL-библиотеке.
4. Метод public static int Task1(int[] arr), предназначенный для нахождения количества пар совпадающих по значению чисел, расположить в DLL-библиотеке.
5. Метод public static void Set\_mas(int[] mas, int k, out int j, int[] rezmas), предназначенный для вывода результирующего массива, расположить в DLL-библиотеке.
6. Метод public static void Add\_pdf(int[] mas, int length), предназначенный для создания PDF-файла, содержащего таблицу со значениями входного массива, расположить в DLL-библиотеке.
7. Метод public static void ZapisBloknot(int[] mas, int lenght), предназначенный для создания блокнот-файла, содержащего список со значениями входного массива, расположить в DLL-библиотеке.
8. Метод public static int[] Task2(int[] mas, int length), предназначенный для удаления наибольшего элемента массива, расположить в DLL-библиотеке.
9. Метод public static int[] Task3(int[] arr, int length, int m, int k), предназначенный для вставки числа в массив, расположить в DLL-библиотеке.
10. Метод public static int Task4(int[] mas, int length), предназначенный для определения монотонно убывающей последовательности, расположить в DLL-библиотеке.
11. Метод public static int Task5(int[] mas, int length), предназначенный для нахождения первого отрицательного элемента, расположить в DLL-библиотеке.
12. public static int Task7(int[] mas, int length), предназначенный для нахождения первого положительного элемента, расположить в DLL-библиотеке.
13. Метод public static int[] Task14(int[] mas, int length), предназначенный для нахождения шейкер-сортировки, расположить в DLL-библиотеке.
14. Метод public static void Zap\_Excel\_Double(int[] mas), предназначенный для создания таблицы в Excel.
15. Метод public static void ZapisWordIsx(int[] mas), предназначенный для записи массива в файл Word.
16. Метод public static void Add1(), предназначенный для создания базы данных.
17. Метод public static void Add\_struct1(), предназначенный для создания таблицы в базе данных.
18. Метод public static void Add\_zap1(int[] mas, int len), предназначенный для добавления массива в таблицу.

Методы должны быть размещены в DLL-библиотеке. В основной части программы следует последовательно осуществить вызов ранее написанных методов.

**Перечень блок схем**

На рисунках 1- 3 представлены блок-схемы событийных кнопок.

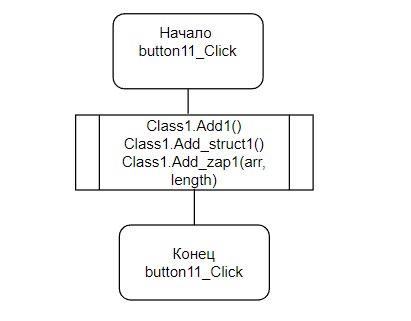


Рисунок 1 – схема алгоритма событийной кнопки

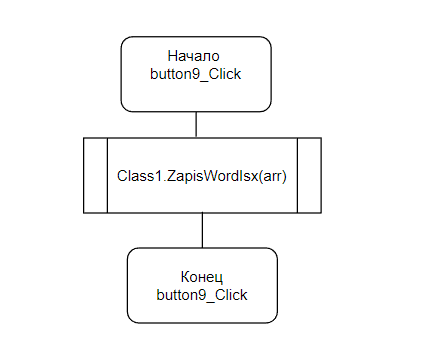


Рисунок 2 – схема алгоритма событийной кнопки

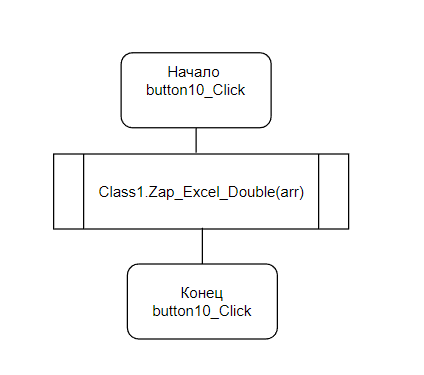


Рисунок 3 – схема алгоритма событийной кнопки

**Содержание DLL-библиотеки:**

using System;

using System.Data.OleDb;

using System.IO;

using System.Windows.Forms;

using iTextSharp.text;

using iTextSharp.text.pdf;

using Excel = Microsoft.Office.Interop.Excel;

namespace Rabota\_Labs

{

public class Class1

{

public static int InputInt(TextBox t)

{

return Convert.ToInt32(t.Text);

}

public static void Enter\_mas(int[] mas, int length, int a, int b)

{

Random rnd = new Random();

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

mas[i] = rnd.Next(a, b);

}

public static int Task1(int[] arr)

{

int count = 0;

for (int i = 0; i < arr.Length; i++)

{

for (int j = i + 1; j < arr.Length; j++)

{

if (arr[i] == arr[j])

{

count++;

}

}

}

return count;

}

public static void Output\_mas(int[] mas, int length, DataGridView dgv)

{

dgv.ColumnCount = length;

dgv.RowCount = 2;

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

{

dgv.Rows[0].Cells[i].Value = "[" + i + "]";

dgv.Rows[1].Cells[i].Value = mas[i];

}

}

public static void Set\_mas(int[] mas, int k, out int j, int[] rezmas)

{

j = 0;

for (int i = 0; i < mas.GetLength(0); i++)

if (mas[i] < k)

{

rezmas[j] = mas[i];

j++;

}

}

public static void Add\_pdf(int[] mas, int lenght)

{

var document = new Document();

var zap = PdfWriter.GetInstance(document, new System.IO.FileStream("ZapPDF.pdf", System.IO.FileMode.Create));

document.Open();

var Shrift = BaseFont.CreateFont(@"C:\WINDOWS\Fonts\times.ttf", "CP1251", BaseFont.EMBEDDED);

var font = new Font(Shrift, 10, Font.NORMAL, BaseColor.BLUE);

var tabl = new PdfPTable(1);

var cell = new PdfPCell();

cell.HorizontalAlignment = Element.ALIGN\_LEFT;

cell.Colspan = 2;

cell.Border = 0;

cell.FixedHeight = 16.0F;

cell.Phrase = new Phrase("Исходный массив", font);

tabl.AddCell(cell);

cell.BackgroundColor = BaseColor.WHITE;

cell.Colspan = 1;

cell.Border = 15;

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

{

cell.Phrase = new Phrase(mas[i].ToString(), font);

tabl.AddCell(cell);

}

tabl.TotalWidth = document.PageSize.Width - 500;

tabl.WriteSelectedRows(0, -1, 150, 750, zap.DirectContent);

document.Close();

zap.Close();

System.Diagnostics.Process.Start("msedge.exe", System.IO.Directory.GetCurrentDirectory() + @"\ZapPDF.pdf");

}

public static void ZapisBloknot(int[] mas, int lenght)

{

StreamWriter rez = File.CreateText("Исходный массив.txt");

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

{

rez.WriteLine(mas[i]);

}

rez.Close();

System.Diagnostics.Process.Start("Исходный массив.txt");

}

public static int[] Task2(int[] mas, int length) // Удаление наибольшего элемента массива

{

int k = 1;

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

{

if (mas[i] > mas[k])

k = i;

}

for (int i = k; i < length - 1; i++)

{

if (i != length - 1)

mas[i] = mas[i + 1];

}

return mas;

}

public static int[] Task3(int[] arr, int length, int m, int k) // Втавка числа в массив

{

for (int i = length - 1; k + 1 > i; i--)

{

arr[i + 1] = arr[i];

}

arr[k + 1] = m;

return arr;

}

public static int Task4(int[] mas, int length) // Определение монотонно убывающей последовательности

{

int Flag = 1;

int i = 0;

while (i <= length - 1 && Flag == 1)

{

if (mas[i] >= mas[i + 1])

i++;

else

Flag = 0;

}

return Flag;

}

public static int Task5(int[] mas, int length) // Нахождение первого отрицательного элемента

{

int n = 0;

int i = 0;

int Flag = 1;

while ((Flag != 0) && (i < length))

{

if (mas[i] < 0)

{

n = mas[i];

Flag = 0;

}

i += 1;

}

return n;

}

public static int Task7(int[] mas, int length) // Нахождение первого положительного элемента

{

int i = 0;

int Flag = 0;

int rez = 0;

while ((i <= length) && (Flag == 0))

{

if (mas[i] > 0)

{

Flag = 1;

rez = mas[i];

}

else

i++;

}

return rez;

}

public static int[] Task14(int[] mas, int length) // Шейкер-сортировка

{

int left = 1;

int right = length - 1;

int k = length - 1;

while (left <= right)

{

for (int j = right; j >= left; j--)

{

if (mas[j - 1] > mas[j])

{

int x = mas[j - 1];

mas[j - 1] = mas[j];

mas[j] = x;

k = j;

}

}

left = k + 1;

for (int j = left; j <= right; j++)

{

if (mas[j - 1] > mas[j])

{

int x = mas[j - 1];

mas[j - 1] = mas[j];

mas[j] = x;

k = j;

}

}

right = k - 1;

}

return mas;

}

public static void ZapisWordIsx(int[] mas) // Запись в текстовый документ Word

{

Microsoft.Office.Interop.Word.Application app = new Microsoft.Office.Interop.Word.Application();

var Wrd = new Microsoft.Office.Interop.Word.Application

{

Visible = true

};

var inf = Type.Missing;

string str;

var Doc = Wrd.Documents.Add(inf);

Wrd.Selection.TypeText("Исходный массив");

object t1 = Microsoft.Office.Interop.Word.WdDefaultTableBehavior.wdWord9TableBehavior;

object t2 = Microsoft.Office.Interop.Word.WdAutoFitBehavior.wdAutoFitContent;

Microsoft.Office.Interop.Word.Table tb1 = Wrd.ActiveDocument.Tables.Add(Wrd.Selection.Range, 2, mas.Length, t1, t2);

for (int i = 0; i < mas.Length; i++)

{

tb1.Cell(1, i + 1).Range.InsertAfter("[" + Convert.ToString(i) + "]");

str = String.Format("{0:f0}", mas[i]);

tb1.Cell(2, i + 1).Range.InsertAfter(str);

}

}

public static void Zap\_Excel\_Double(int[] mas) // Запись в таблицы Excel

{

Excel.Application excelApp = new Excel.Application();

Excel.Workbook workBook;

Excel.Worksheet workSheet;

string cd = Directory.GetCurrentDirectory();

workBook = excelApp.Workbooks.Add(cd + @"\Macros1.xlsm");

workSheet = (Excel.Worksheet)workBook.Worksheets.get\_Item(1);

workSheet.Name = "Исходный Массив";

workSheet.Cells[1, 1] = "Исходный Массив";

for (int i = 0; i < mas.Length; i++)

{

workSheet.Cells[2, i + 1] = "[" + i + "]";

workSheet.Cells[3, i + 1] = mas[i];

}

Excel.Range range1 = workSheet.Range[workSheet.Cells[2, 1],

workSheet.Cells[3, mas.Length]];

range1.Cells.Font.Name = "Times New Roman";

range1.Cells.Font.Size = 14;

range1.Cells.Columns.AutoFit();

range1.Borders.get\_Item(Excel.XlBordersIndex.xlEdgeBottom).LineStyle = Excel.XlLineStyle.xlContinuous;

range1.Borders.get\_Item(Excel.XlBordersIndex.xlEdgeRight).LineStyle = Excel.XlLineStyle.xlContinuous;

range1.Borders.get\_Item(Excel.XlBordersIndex.xlInsideHorizontal).LineStyle = Excel.XlLineStyle.xlContinuous;

range1.Borders.get\_Item(Excel.XlBordersIndex.xlInsideVertical).LineStyle =

Excel.XlLineStyle.xlContinuous;

range1.Borders.get\_Item(Excel.XlBordersIndex.xlEdgeTop).LineStyle =

Excel.XlLineStyle.xlContinuous;

workSheet.Range[("A7")].Select();

excelApp.Visible = true;

excelApp.UserControl = true;

}

public static void Add1() // База данных Access

{

var k = new ADOX.Catalog();

var location = System.Reflection.Assembly.GetExecutingAssembly().Location;

var path = System.IO.Path.GetDirectoryName(location);

try

{

k.Create("Provider=Microsoft.Jet.OLEDB.4.0; Data Source=" +

path.ToString().Replace("\\", "\\\\") + "\\Results.mdb");

}

catch (System.Runtime.InteropServices.COMException exp)

{

MessageBox.Show(exp.Message, "Информация", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

finally

{

k = null;

}

}

public static void Add\_struct1() // Добавление таблицы

{

var location = System.Reflection.Assembly.GetExecutingAssembly().Location;

var path = System.IO.Path.GetDirectoryName(location);

var p = new OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0; Data Source=" + path.ToString().Replace("\\", "\\\\") + "\\Results.mdb");

p.Open();

var c = new OleDbCommand("Create Table [results] ([№ Вопроса] counter, [Результат] char(10))", p);

try

{

c.ExecuteNonQuery();

}

catch (Exception exp)

{

MessageBox.Show(exp.Message, "Информация", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

p.Close();

}

public static void Add\_zap1(int[] mas, int len) // Заполнение таблицы массивом

{

var location = System.Reflection.Assembly.GetExecutingAssembly().Location;

var path = System.IO.Path.GetDirectoryName(location);

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

{

var p = new OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0; Data Source=" +

path.ToString().Replace("\\", "\\\\") + "\\Results.mdb");

p.Open();

var c = new OleDbCommand("INSERT INTO [results] (" + " [Результат]) VALUES('" + mas[i] + "')");

c.Connection = p;

c.ExecuteNonQuery();

p.Close();

}

MessageBox.Show("База данных создана", "Создание БД", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

}

**Содержание основной части программы:**

using System;

using System.IO;

using Rabota\_Labs;

using System.Windows.Forms;

namespace Forma\_LB

{

public partial class LB7 : Form

{

public LB7()

{

InitializeComponent();

}

private static int[] arr;

private static int length;

private static int k;

private static int m;

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

{

List\_Rabot f = new List\_Rabot();

this.Hide();

f.ShowDialog();

}

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

{

dataGridView1.ColumnCount = 0;

dataGridView2.ColumnCount = 0;

if (textBox1.Text == "")

{

MessageBox.Show("Вы не ввели значение", "Вывод", MessageBoxButtons.OK, MessageBoxIcon.Error);

textBox1.Focus();

return;

}

length = Class1.InputInt(textBox1);

arr = new int[length];

int A = Class1.InputInt(textBox2);

int B = Class1.InputInt(textBox3);

Class1.Enter\_mas(arr, length, A, B);

Class1.Output\_mas(arr, length, dataGridView1);

int W = Class1.Task1(arr);

MessageBox.Show(Convert.ToString(W), "Вывод");

int[] rezmas = new int[length];

Class1.Set\_mas(arr, W, out int j, rezmas);

Class1.Output\_mas(rezmas, j, dataGridView2);

Class1.Add\_pdf(arr, length);

Class1.ZapisBloknot(arr, length);

}

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

{

int[] array = Class1.Task2(arr, length);

Class1.Output\_mas(array, length - 1, dataGridView2);

}

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

{

int[] array = Class1.Task3(arr, length, m, k);

Class1.Output\_mas(array, length, dataGridView2);

}

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

{

int flag = Class1.Task4(arr, length);

if (flag == 1)

MessageBox.Show("Убывает монотонно");

else

MessageBox.Show("Не убывает");

}

private void button6\_Click(object sender, EventArgs e)

{

int res = Class1.Task5(arr, length);

if (res != 0)

MessageBox.Show("Результат:" + " " + res, "Вывод");

else

MessageBox.Show("Ошибка,положительное число", "Вывод");

}

private void button7\_Click(object sender, EventArgs e)

{

int res = Class1.Task7(arr, length);

if (res != 0)

MessageBox.Show("Результат:" + " " + res, "Вывод");

else

MessageBox.Show("Ошибка,отрицательное число", "Вывод");

}

private void button8\_Click(object sender, EventArgs e)

{

int[] array = Class1.Task14(arr, length);

Class1.Output\_mas(array, length, dataGridView2);

}

private void button9\_Click(object sender, EventArgs e)

{

Class1.ZapisWordIsx(arr);

}

private void button10\_Click(object sender, EventArgs e)

{

Class1.Zap\_Excel\_Double(arr);

}

private void button11\_Click(object sender, EventArgs e)

{

Class1.Add1();

Class1.Add\_struct1();

Class1.Add\_zap1(arr, length);

}

private void button12\_Click(object sender, EventArgs e)

{

Microsoft.Office.Interop.Excel.Application app = new Microsoft.Office.Interop.Excel.Application();

app.Visible = true;

Microsoft.Office.Interop.Excel.Workbooks books = app.Workbooks;

Microsoft.Office.Interop.Excel.\_Workbook book = null;

string cdir = Directory.GetCurrentDirectory();

book = books.Open(cdir + @"\Macros1.xlsm");

app.Run((object)"Макрос1");

app.ScreenUpdating = true;

}

}

}

**Результаты выполнения программы**

На рисунках 4-7 представлены результаты выполнения программы.

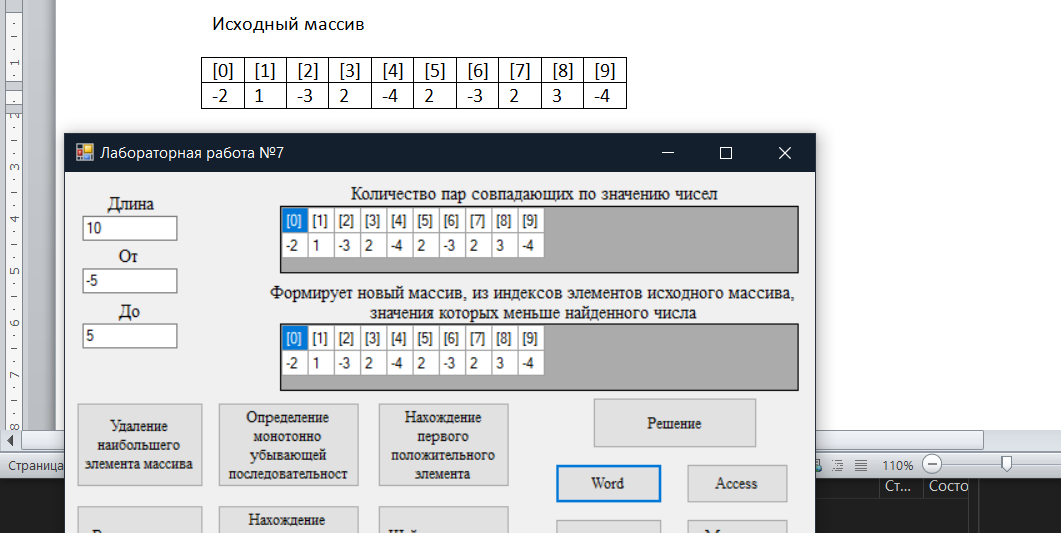


Рисунок 4 – Результат выполнения работы

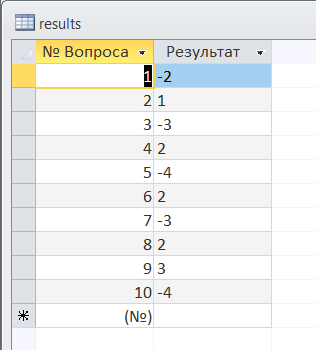


Рисунок 5 – Результат выполнения работы

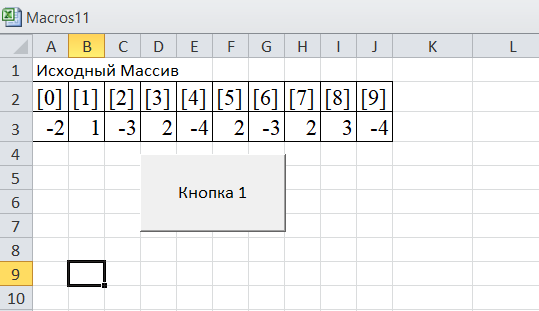


Рисунок 6 – Результат выполнения работы

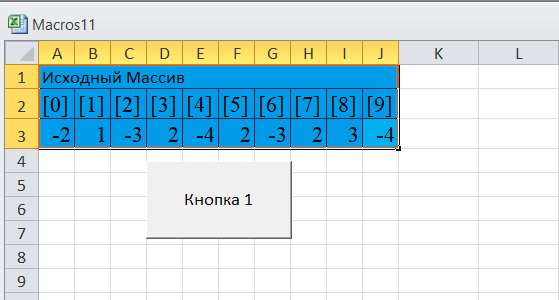


Рисунок 7 – Результат выполнения работы

**Лабораторная работа №8**

**«Работа с двумерными массивами»**

**Задание**

Сформируйте произвольный двумерный массив. Получите и выведите одномерный массив R, где – сумма элементов, предшествующих последнему отрицательному элементу j-го столбца; если все элементы столбца неотрицательны, то примите = -1.

Кроме того, необходимо выполнить запись двумерных массивов в программы прикладного назначения, а именно: текстовый редактор Microsoft Word, базу данных Microsoft Access, табличный процессор Microsoft Excel, текстовый редактор Блокнот.

Осуществить написание макроса, предназначенного для изменения цвета ячеек, а также создать кнопку на листе Microsoft Excel, по нажатию на которую, будет осуществлять заливка выбранных ячеек.

**Для того, чтобы выполнить задание, необходимо разработать следующие методы:**

1. Метод public static int InputInt(TextBox t), предназначенный для конвертации строкового значения в числовое.
2. Метод public static void ArrayGenerate(int[,] mas, int n, int m), предназначенный для заполнения массива случайными числами.
3. Метод public static void mas1(int[] mas, int length, DataGridView dgv), предназначенный для вывода массива в таблицу.
4. Метод public static void Output\_mas(int[,] mas, int n, int m, DataGridView dgv), предназначенный для ввода и вывода данных в текстовые поля.
5. Метод public static int[] Task1(int[,] mas, int n, int m), предназначенный для нахождения суммы элементов, предшествующих последнему отрицательному j-го столбца.
6. Метод public static void ZapisWord(int[,] mas, int[] rezmas, int a, int b, int g), предназначенный для записи массива в файл Word.
7. Метод public static void Add(), предназначенный для создания базы данных.
8. Метод public static void CreateStructBD(int b), предназначенный для структуры базы данных.
9. Метод public static void CommBD(String commandString), предназначенный для команд в базе данных.
10. Метод public static void AddToBD(int[,] mas, int a, int b), предназначенный для заполнения базы данных.
11. Метод public static void ZapisBloknot(int[,] mas, int a, int b), предназначенный для записи массива в блокнот.
12. Метод public static void Zap\_Excel(int[,] mas, int a, int b, int g), предназначенный для записи массива в Excel.

Данные методы должны быть размещены в заранее разработанной новой DLL библиотеке.

**В основной части программы следует последовательно осуществить**

**вызов ранее написанных методов.**

**Перечень блок-схем**

На рисунке 1 представлен алгоритм метода, конвертирующего строку в число и возвращающий это значение.

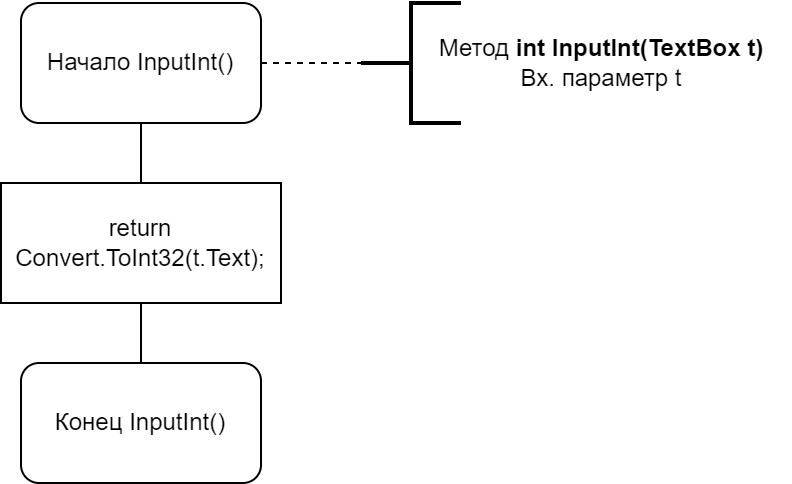


Рисунок 1 – Алгоритм метода, конвертирующего строку в число и возвращающий это значение

На рисунке 2 представлен алгоритм метода, предназначенный для заполнения массива случайными числами.

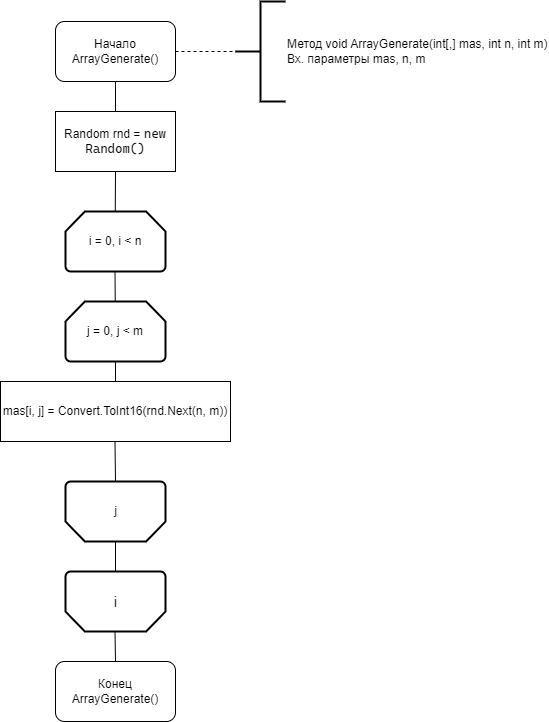


Рисунок 2 - Алгоритм метода, заполняющий массив случайными числами

На рисунке 3 представлен алгоритм метода, который выводит массив в таблицу.

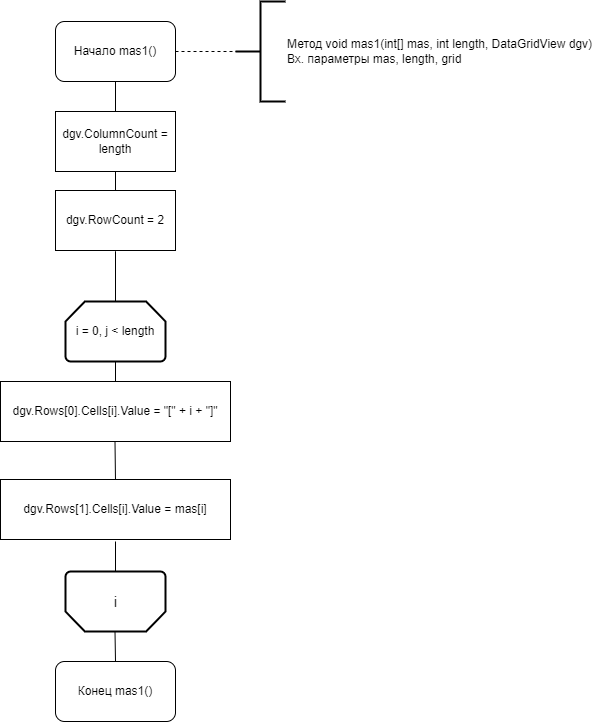


Рисунок 3 – Алгоритм метода, выводящий массив в таблицу

На рисунке 4 представлен алгоритм метода, предназначенный для ввода и вывода данных в текстовые поля.

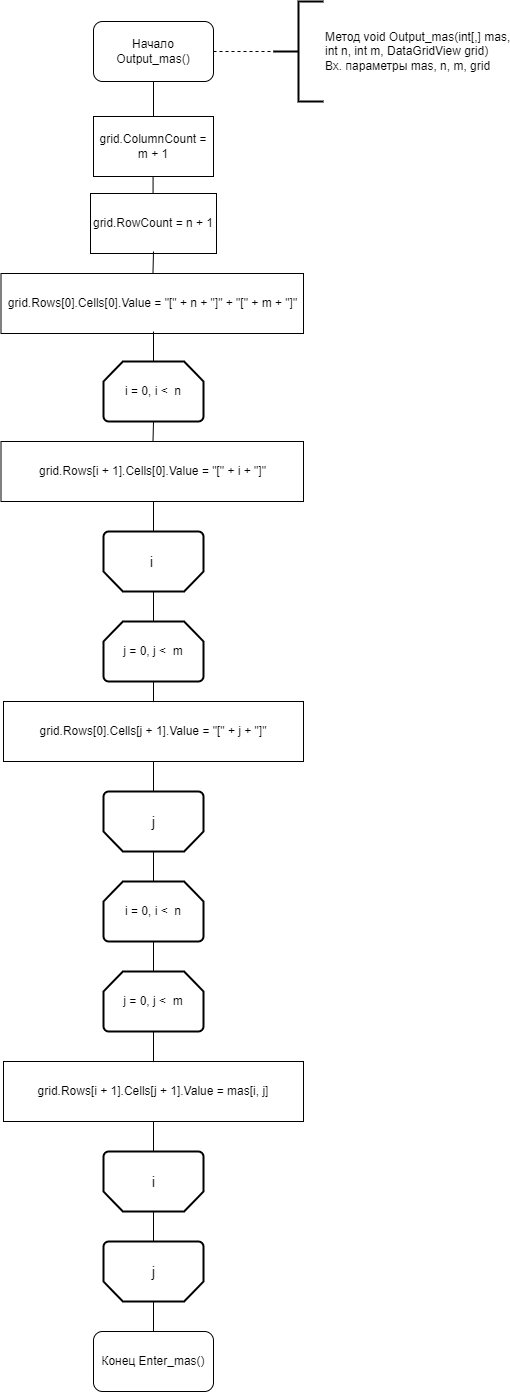


Рисунок 4 - Алгоритм метода, выводящего массив в таблицу

На рисунке 5 представлен алгоритм метода, находящий сумму элементов, предшествующих последнему отрицательному j-го столбца.

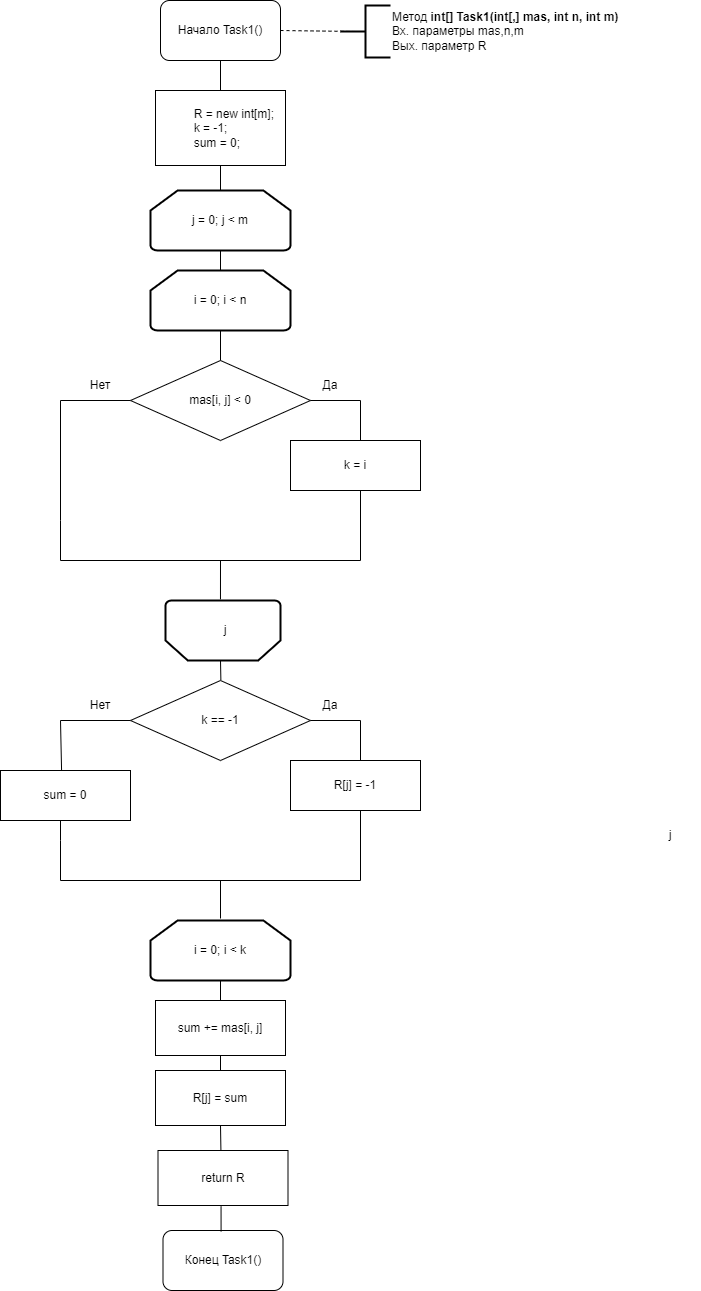


Рисунок 5 - Алгоритм метода, находящий сумму элементов, предшествующих последнему отрицательному j-го столбца

На рисунках 6-7 представлены алгоритмы событийных кнопок.

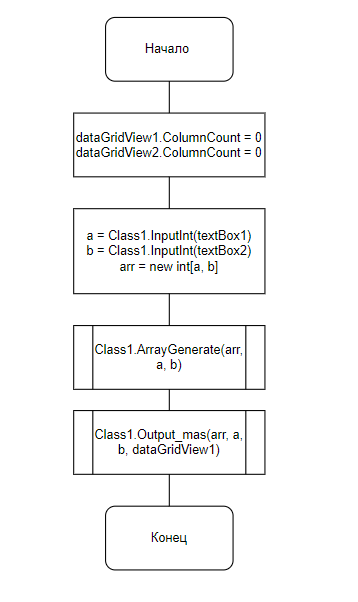


Рисунок 6 – Алгоритм событийной кнопки

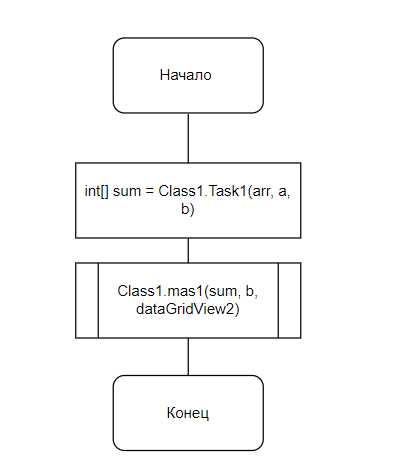


Рисунок 7 – Алгоритм событийной кнопки

**Содержание DLL-библиотеки**

using System;

using System.Data.OleDb;

using System.IO;

using System.Windows.Forms;

using iTextSharp.text;

using iTextSharp.text.pdf;

using Excel = Microsoft.Office.Interop.Excel;

namespace Dll\_Rabota\_Lab

{

public class Class1

{

public static int[] rezmas;

public static int InputInt(TextBox t)

{

return Convert.ToInt32(t.Text);

}

public static void ArrayGenerate(int[,] mas, int n, int m)

{

Random rnd = new Random();

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

for (int j = 0; j < m; j++)

mas[i, j] = Convert.ToInt16(rnd.Next(-50, 100));

}

public static void mas1(int[] mas, int length, DataGridView dgv)

{

dgv.ColumnCount = length;

dgv.RowCount = 2;

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

{

dgv.Rows[0].Cells[i].Value = "[" + i + "]";

dgv.Rows[1].Cells[i].Value = mas[i];

}

}

public static void Output\_mas(int[,] mas, int n, int m, DataGridView dgv)

{

dgv.ColumnCount = m + 1;

dgv.RowCount = n + 1;

dgv.Rows[0].Cells[0].Value = "[" + n + "]" + "[" + m + "]";

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

dgv.Rows[i + 1].Cells[0].Value = "[" + i + "]";

for (int j = 0; j < m; j++)

dgv.Rows[0].Cells[j + 1].Value = "[" + j + "]";

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

for (int j = 0; j < m; j++)

dgv.Rows[i + 1].Cells[j + 1].Value = mas[i, j];

}

public static int[] Task1(int[,] mas, int n, int m)

{

int[] R = new int[m];

int k = -1;

int sum = 0;

for (int j = 0; j < m; j++)

{

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

{

if (mas[i, j] < 0)

{

k = i;

}

}

if (k == -1)

{

R[j] = -1;

}

else

{

sum = 0;

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

{

sum += mas[i, j];

}

R[j] = sum;

}

k = -1;

}

return R;

}

public static void ZapisWord(int[,] mas, int[] rezmas, int a, int b, int g)

{

Microsoft.Office.Interop.Word.Application app = new Microsoft.Office.Interop.Word.Application();

var Wrd = new Microsoft.Office.Interop.Word.Application

{

Visible = true

};

var inf = Type.Missing;

string str;

var Doc = Wrd.Documents.Add(inf);

Wrd.Selection.TypeText("Исходный массив");

Object t1 = Microsoft.Office.Interop.Word.WdDefaultTableBehavior.wdWord9TableBehavior;

Object t2 = Microsoft.Office.Interop.Word.WdAutoFitBehavior.wdAutoFitContent;

Microsoft.Office.Interop.Word.Table tbl = Wrd.ActiveDocument.Tables.Add(Wrd.Selection.Range, a + 1, b + 1, t1, t2);

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

tbl.Cell(i + 2, 1).Range.InsertAfter("[" + Convert.ToString(i) + "]");

for (int j = 0; j < b; j++)

tbl.Cell(1, j + 2).Range.InsertAfter("[" + Convert.ToString(j) + "]");

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

for (int j = 0; j < b; j++)

{

str = String.Format("{0:f2}", mas[i, j]);

tbl.Cell(i + 2, j + 2).Range.InsertAfter(str);

}

Object t3 = Microsoft.Office.Interop.Word.WdUnits.wdLine;

Object add\_str = a + 2;

Wrd.Selection.MoveDown(t3, add\_str, inf);

Wrd.Selection.TypeText("Сумма элементов, предшествующая последнему отрицательному j-го столбца:" + g + "\n Результирующий массив");

tbl = Wrd.ActiveDocument.Tables.Add(Wrd.Selection.Range, 2, g, t1, t2);

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

{

tbl.Cell(1, i + 1).Range.InsertAfter("[" + Convert.ToString(i) + "]");

str = String.Format("{0;f2}", rezmas[i]);

tbl.Cell(2, i + 1).Range.InsertAfter(str);

}

}

public static void Add()

{

var k = new ADOX.Catalog();

var location = System.Reflection.Assembly.GetExecutingAssembly().Location;

var path = System.IO.Path.GetDirectoryName(location);

try

{

k.Create("Provider=Microsoft.ACE.OLEDB.12.0; Data Source=" +

path.ToString().Replace("\\", "\\\\") + "\\Results12.accdb");

}

catch (System.Runtime.InteropServices.COMException exp)

{

MessageBox.Show(exp.Message, "Информация", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

finally

{

k = null;

}

}

public static void CreateStructBD(int b)

{

var location = System.Reflection.Assembly.GetExecutingAssembly().Location;

var path = System.IO.Path.GetDirectoryName(location);

var Connect = new OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0; Data Source=" + path.ToString().Replace("\\", "\\\\") + "\\Results12.accdb");

Connect.Open();

String Col = "CREATE TABLE [Results12] ([Rows] counter";

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

Col = Col + ", [" + "Col" + Convert.ToString(i + 1) + "] char(5)";

var Command = new OleDbCommand(Col + ")", Connect);

try

{

Command.ExecuteNonQuery();

MessageBox.Show("База данных создана");

}

catch (System.Runtime.InteropServices.COMException exp)

{

MessageBox.Show(exp.Message);

}

Connect.Close();

}

public static void CommBD(String commandString)

{

var location = System.Reflection.Assembly.GetExecutingAssembly().Location;

var path = System.IO.Path.GetDirectoryName(location);

var Connection = new OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0; Data Source=" + path.ToString().Replace("\\", "\\\\") + "\\Results12.accdb");

Connection.Open();

var Command = new OleDbCommand("" + commandString);

Command.Connection = Connection;

Command.ExecuteNonQuery();

Connection.Close();

}

public static void AddToBD(int[,] mas, int a, int b)

{

String cmdString;

String cmdString2;

var location = System.Reflection.Assembly.GetExecutingAssembly().Location;

var path = System.IO.Path.GetDirectoryName(location);

var Connection = new OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0; Data Source=" + path.ToString().Replace("\\", "\\\\") + "\\Results12.accdb");

Connection.Open();

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

{

cmdString = "INSERT INTO [Results12] ([Rows]";

cmdString2 = ") VALUES (" + "'" + Convert.ToString(i + 1) + "'";

for (int j = 0; j < b; j++)

{

cmdString = cmdString + ", [Col" + Convert.ToString(j + 1) + "]";

cmdString2 = cmdString2 + ", '" + Convert.ToString(mas[i, j]) + "'";

}

cmdString2 = cmdString2 + ")";

CommBD(cmdString + cmdString2);

Connection.Close();

}

MessageBox.Show("Структура таблицы", "Вывод", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

public static void ZapisBloknot(int[,] mas, int a, int b)

{

StreamWriter rez = File.CreateText("Исходный массив.txt");

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

{

for (int j = 0; j < b; j++)

{

rez.Write(mas[i, j] + "\t");

}

rez.WriteLine("\n");

}

rez.Close();

System.Diagnostics.Process.Start("Исходный массив.txt");

}

public static void Zap\_Excel(int[,] mas, int a, int b, int g)

{

Excel.Application excelApp = new Excel.Application();

Excel.Workbook workBook;

Excel.Worksheet workSheet;

string cd = Directory.GetCurrentDirectory();

workBook = excelApp.Workbooks.Add(cd + @"\Macros2.xlsm");

workSheet = (Excel.Worksheet)workBook.Worksheets.get\_Item(1);

workSheet.Name = "Массив исходный";

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

{

for (int j = 0; j < b; j++)

{

workSheet.Cells[i + 1, j + 1] = mas[i, j];

}

}

workSheet.Cells[9, 1] = "Результирующий массив";

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

{

workSheet.Cells[10, i + 1] = "[" + i + "]";

workSheet.Cells[11, i + 1] = rezmas[i];

}

workSheet.Range[("A15")].Select();

excelApp.Visible = true;

excelApp.UserControl = true;

}

}

}

}

**Содержание основной части программы**

using System;

using System.Windows.Forms;

using Dll\_Rabota\_Lab;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace Forma\_LB

{

public partial class LB8 : Form

{

public LB8()

{

InitializeComponent();

}

private static int[,] arr;

private static int a;

private static int b;

private static int g;

private static int [] rezmas;

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

{

dataGridView1.ColumnCount = 0;

dataGridView2.ColumnCount = 0;

if (textBox1.Text == "")

{

MessageBox.Show("Вы не ввели значение", "Вывод", MessageBoxButtons.OK, MessageBoxIcon.Error);

textBox1.Focus();

return;

}

a = Class1.InputInt(textBox1);

b = Class1.InputInt(textBox2);

arr = new int[a, b];

Class1.ArrayGenerate(arr, a, b);

Class1.Output\_mas(arr, a, b, dataGridView1);

}

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

{

List\_Rabot f = new List\_Rabot();

this.Hide();

f.ShowDialog();

}

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

{

int[] sum = Class1.Task1(arr, a, b);

Class1.mas1(sum, b, dataGridView2);

}

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

{

Class1.ZapisWord(arr, rezmas, a, b, g);

}

private void button6\_Click(object sender, EventArgs e)

{

Class1.Add();

Class1.CreateStructBD(b);

Class1.AddToBD(arr, a,b);

}

private void button7\_Click(object sender, EventArgs e)

{

Class1.ZapisBloknot(arr, a, b);

}

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

{

Class1.Zap\_Excel(arr, a, b,g);

}

}

}

**Результаты выполнения программы**

На рисунке 8-12 представлены результаты выполнения программы.

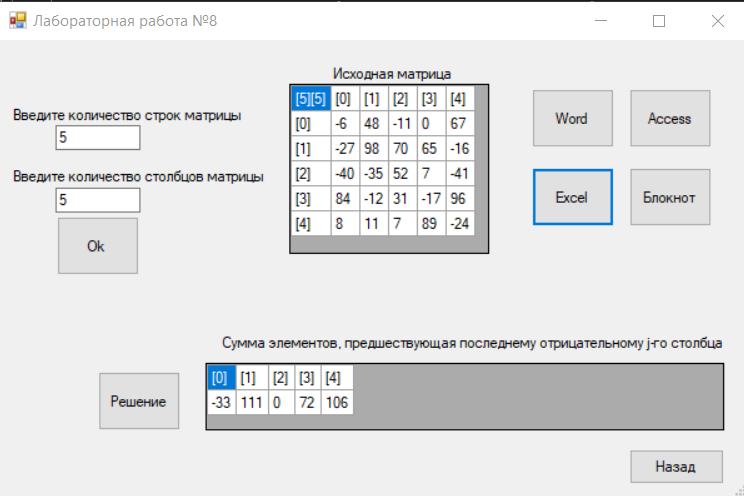


Рисунок 8- Результат выполнения работы

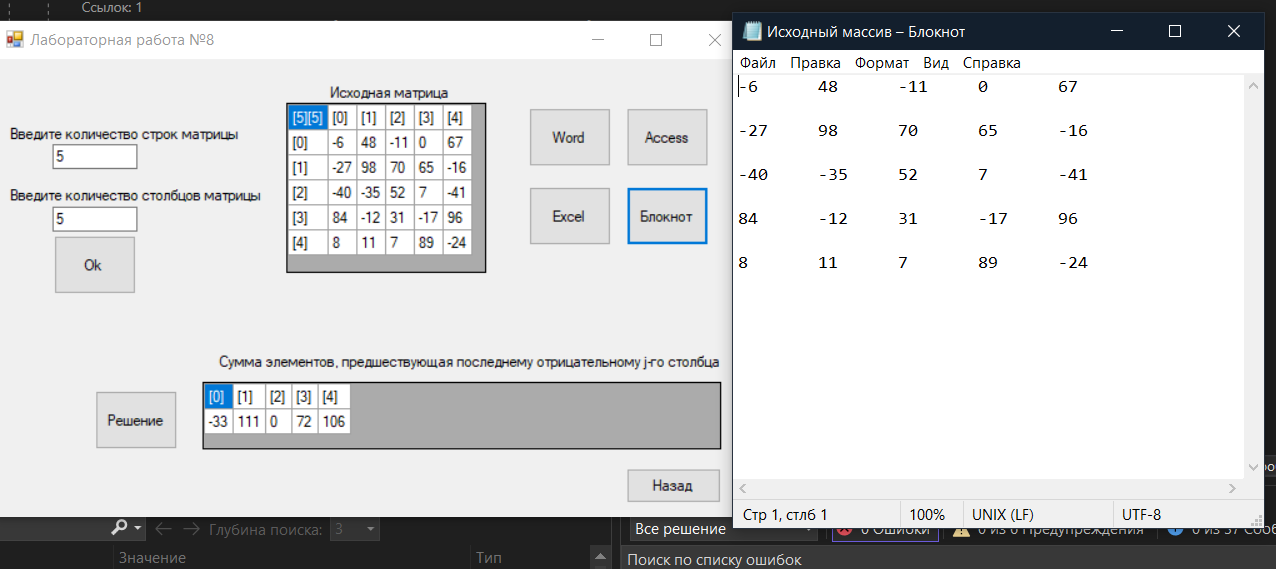


Рисунок 9- Результат выполнения работы

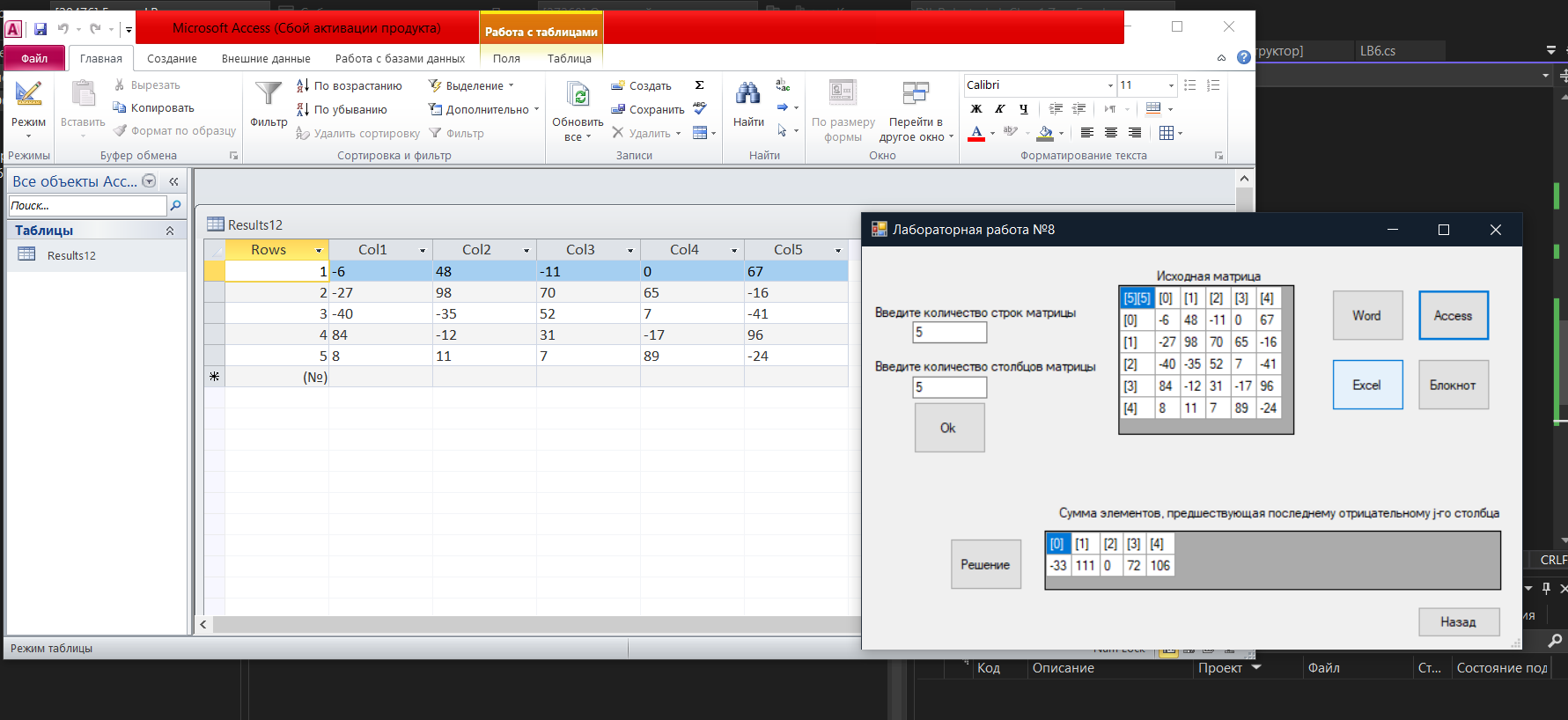


Рисунок 10- Результат выполнения работы

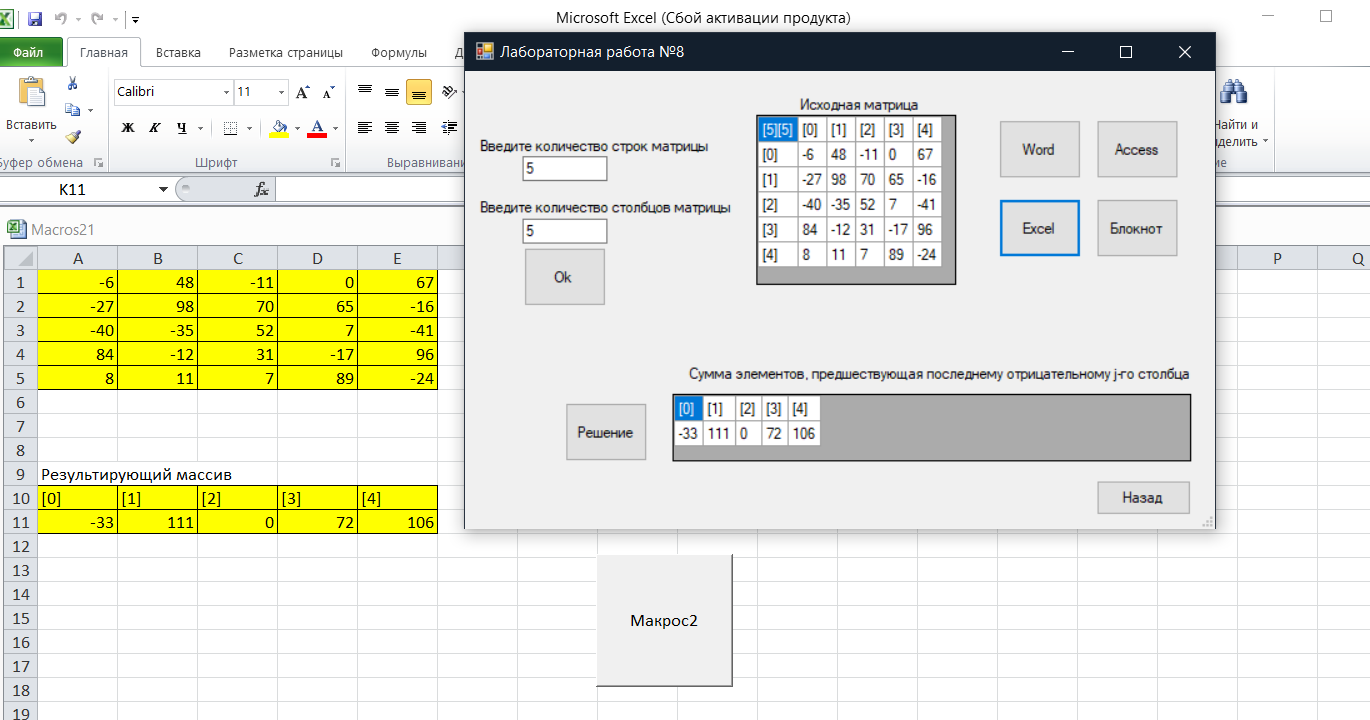


Рисунок 11- Результат выполнения работы

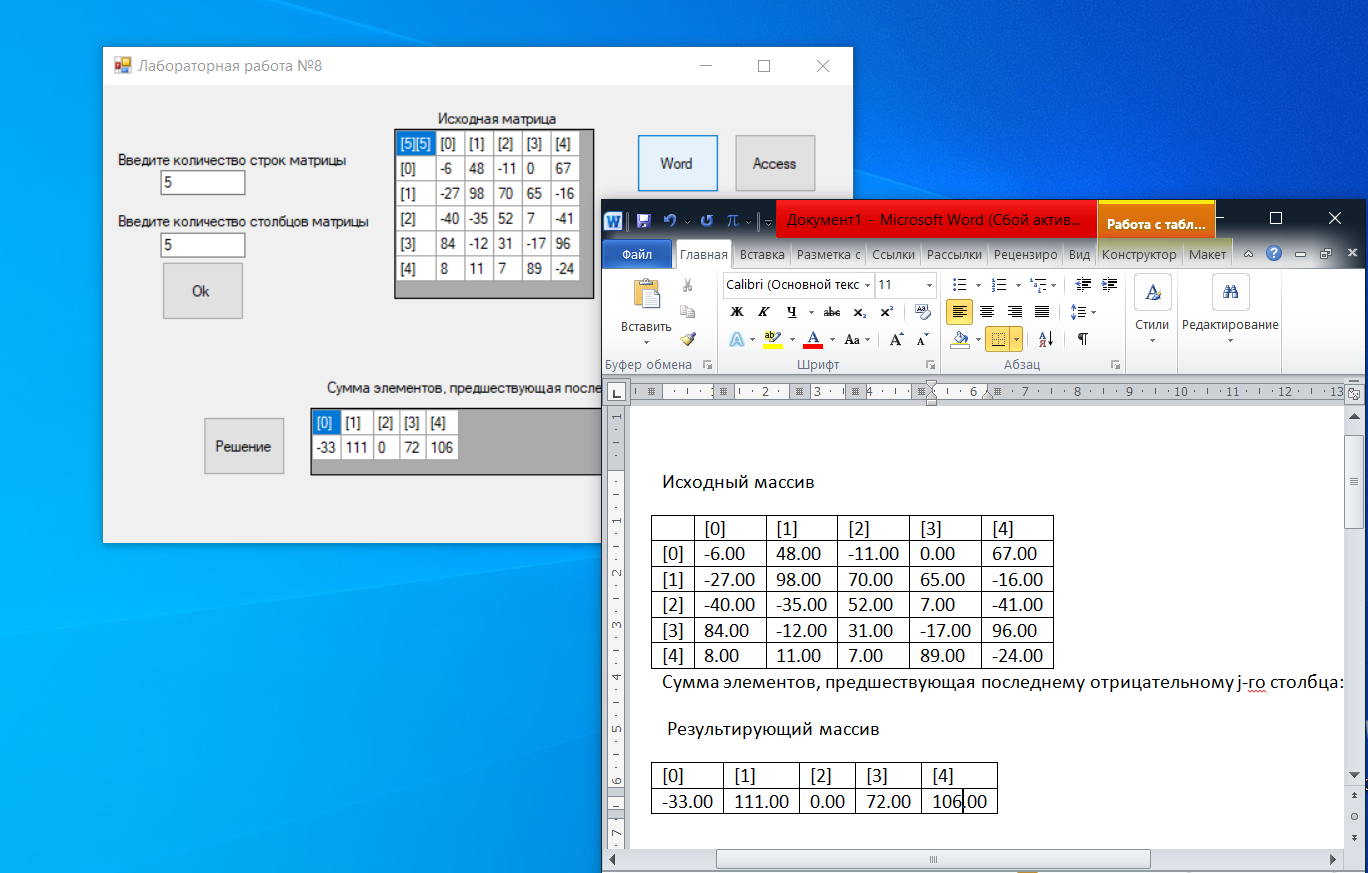


Рисунок 12- Результат выполнения работы

**Список используемых источников**

1. Гуриков С. Р. Введение в программирование на языке Visual C#: учебное пособие / С.Р. Гуриков. — Москва : ФОРУМ : ИНФРА-М, 2020. — 447 с.
2. ГОСТ 7.32-2017, Москва.; Стандартинформ 2017.; – 26 с.