**23 ЭЛЕМЕНТЫ УПРАВЛЕНИЯ**

Задание №1. Дана строка, состоящая из групп нулей и единиц. Посчитать количество нулей и единиц.

Листинг программы:

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

{

int ind = listBox1.SelectedIndex;

string str = (string)listBox1.Items[ind];

int leng = str.Length;

int OneAmount = 0;

int ZeroAmount = 0;

int i = 0;

while (i < leng)

{

if (str[i] == '1') {

OneAmount += 1;

}

else if (str[i] == '0')

{

ZeroAmount += 1;

}

i += 1;

}

labelOnes.Text = $"Counting 1: {Convert.ToString(OneAmount)}";

labelZeros.Text = $"Counting 0: {Convert.ToString(ZeroAmount)}";

}

Таблица 23.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 111100000111 | 7, 5 |

Источник: собственная разработка

Анализ результата разработки приложения представлен на рисунке 23.1.

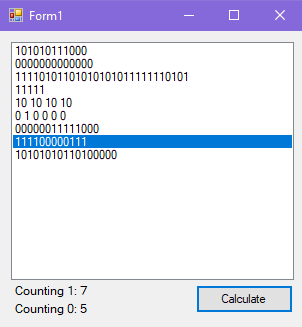


Рисунок 23.1 – Результат работы программы

Источник: собственная разработка

Задание №2. Построить график функции sin2x. Таблицу данных получить путём изменения параметра X с шагом h. Самостоятельно выбрать удобные параметры настройки.

Листинг программы:

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

{

try {

double MinX = double.Parse(textBoxXmin.Text);

double MaxX = double.Parse(textBoxXmax.Text);

double a = double.Parse(textBoxA.Text);

double b = double.Parse(textBoxB.Text);

double Step = double.Parse(textBoxStep.Text);

int count = (int)Math.Ceiling((MaxX - MinX) / Step) + 1;

double[] x = new double[count];

double[] y1 = new double[count];

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

{

x[i] = MinX + Step \* i;

y1[i] = Math.Pow(Math.Sin(x[i]), 2);

}

chart1.ChartAreas[0].AxisX.Minimum = MinX;

chart1.ChartAreas[0].AxisX.Maximum = MaxX;

chart1.ChartAreas[0].AxisX.MajorGrid.Interval = Step;

chart1.Series[0].Points.DataBindXY(x, y1);

}

catch (Exception ex)

{

MessageBox.Show($"An error occured - {ex.Message}", "Error!", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

Таблица 23.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 1, 10, 2, 8, 1 | function graph |

Источник: собственная разработка

Анализ результата разработки приложения представлен на рисунке 23.2.

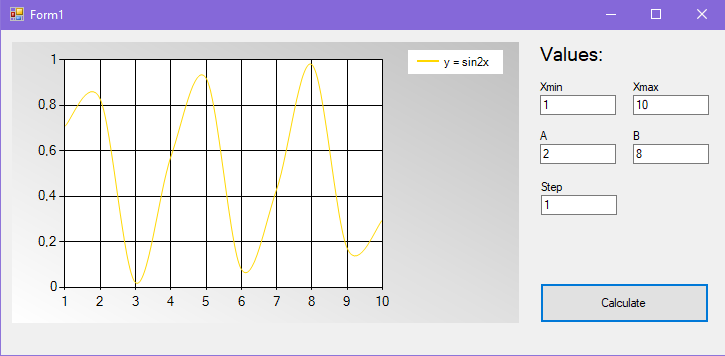


Рисунок 23.2 – Результат работы программы

Источник: собственная разработка

Задание №3. Построить изображение «Домик» при помощи фигур.

Листинг программы:

public partial class Form1 : Form

{

static int n = 100;

public Form1() {

InitializeComponent();

}

List<SetBP> list = new List<SetBP>();

int solX = -60, solY = 30, solD = 80;

int dSol = -1; int k = 0;

private void Form1\_Load\_1(object sender, EventArgs e)

{

Random rand = new Random();

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

list.Add(new SetBP(rand, i));

}

private void Form1\_Paint\_1(object sender, PaintEventArgs e)

{

solX += 1;

k += 1;

if (k % 7 == 0) { solY += dSol; k = 0; }

if (solX > 400) dSol = 1;

if (solX > 820) { solX = -60; dSol = -1; solY = 30; }

foreach (SetBP steam in list) steam.DrawHouse(e);

}

}

class SetBP

{

int D = 2, X = 200, Y = 100;

Random rand;

Brush brWhite = new SolidBrush(Color.White);

Brush brRed = new SolidBrush(Color.Red);

Brush brBlack = new SolidBrush(Color.Black);

Brush brBrown = new SolidBrush(Color.Brown);

Brush brGreen = new SolidBrush(Color.Green);

Brush brGreen1 = new SolidBrush(Color.LimeGreen);

Pen penRed = new Pen(Color.Red);

Pen penBlack = new Pen(Color.Black, 4);

Pen penBrown = new Pen(Color.Brown, 3);

public void DrawHouse(PaintEventArgs e)

{

for (int i = 0; i < 2; i++)

{

e.Graphics.FillRectangle(brRed, 50, 200, 200, 200);

e.Graphics.DrawRectangle(penBlack, 50, 200, 200, 200);

e.Graphics.FillRectangle(brBrown, 190, 110, 30, 60);

e.Graphics.DrawRectangle(penBlack, 190, 110, 30, 60);

Point[] p = new Point[3];

p[0].X = 150; p[0].Y = 100;

p[1].X = 10; p[1].Y = 200;

p[2].X = 300; p[2].Y = 200;

e.Graphics.FillPolygon(brBlack, p);

e.Graphics.FillRectangle(brBrown, 120, 300, 50, 100);

e.Graphics.DrawRectangle(penBlack, 120, 300, 50, 100);

e.Graphics.FillRectangle(brWhite, 80, 220, 40, 50);

e.Graphics.DrawRectangle(penBlack, 80, 220, 40, 50);

e.Graphics.DrawLine(penBlack, 100, 220, 100, 270);

e.Graphics.DrawLine(penBlack, 80, 245, 120, 245);

}

}

public SetBP(Random r, int n)

{

rand = r;

}

}

Таблица 23.3 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| figures | result picture |

Источник: собственная разработка

Анализ результата разработки приложения представлен на рисунке 23.3.

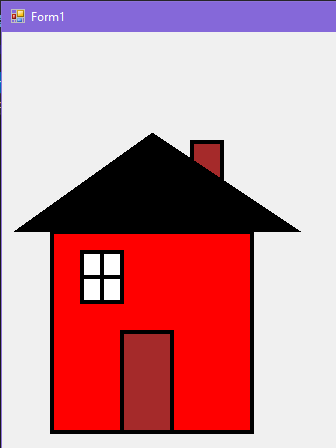


Рисунок 23.3 – Результат работы программы

Источник: собственная разработка

Задание №4. Написать программу с кнопкой и тремя полями ввода. При нажатии на кнопку программа анализирует содержимое первого поля и динамически порождает элемент управления. Если в первом поле ввода содержится буква «К», то на форму добавляется кнопка, если буква «П»-поле ввода, если «М»-метка. Во втором и третьем поле находятся координаты левого верхнего угла будущего элемента управления. При создании будущего элемента настроить его свойства (на ваш выбор (цвет, размер и т.п)). При наведении курсора на элемент управления он должен быть удален с формы.

Листинг программы:

private void Button1\_Click(object sender, EventArgs e)

{

try

{

string sign = textBoxLetter.Text;

double x = Convert.ToDouble(textBoxX.Text);

double y = Convert.ToDouble(textBoxY.Text);

if (sign == "T")

{

TextBox textBox = new TextBox();

textBox.MouseMove += MouseDownTB;

textBox.Location = new Point(Convert.ToInt32(x), Convert.ToInt32(y));

this.Controls.Add(textBox);}

else if (sign == "B")

{

Button button = new Button();

button.Text = "Button";

button.MouseMove += MouseDownB;

button.Location = new Point(Convert.ToInt32(x), Convert.ToInt32(y));

this.Controls.Add(button);

}

else if (sign == "L")

{

Label label = new Label();

label.Text = "Sign";

label.MouseMove += MouseDownL;

label.Location = new Point(Convert.ToInt32(x), Convert.ToInt32(y));

this.Controls.Add(label);

}

else

{

MessageBox.Show("Control element is missing!", "Warning", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

catch (Exception ex)

{

MessageBox.Show($"An error occored - {ex.Message}", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void MouseDownB(Object sender, MouseEventArgs e)

{

Button B = (Button)sender;

B.Dispose();

}

private void MouseDownL(Object sender, MouseEventArgs e)

{

Label L = (Label)sender;

L.Dispose();

}

private void MouseDownTB(Object sender, MouseEventArgs e)

{

TextBox TB = (TextBox)sender;

TB.Dispose();

}

Таблица 23.4 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| B, 200, 8 | control element |

Источник: собственная разработка

Анализ результата разработки приложения представлен на рисунке 23.4.

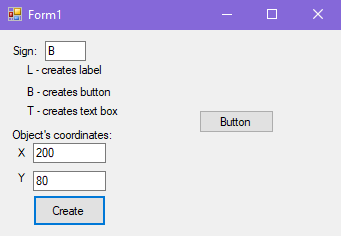


Рисунок 23.4 – Результат работы программы

Источник: собственная разработка

Задание №5. Разработать Windows-приложение, позволяющее пользователю вводить и выводить анкетные данные служащих.

Листинг программы:

private void Form1\_Load(object sender, EventArgs e)

{

YearDropBox.Items.Clear();

var thisYear = DateTime.Now.Year;

for (int i = 1900; i <= thisYear; i++)

{

YearDropBox.Items.Add(i.ToString()); }

}

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

{

File.Delete(@"hr\_info.txt");

string path = @"hr\_info.txt";

SaveInfo(this, path);

}

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

{

ClearFields(this);

}

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

{

Application.Exit();

}

private static void ClearFields(Control parent)

{

foreach (Control c in parent.Controls)

{

if (c is TextBox)

{

((TextBox)c).Clear();

}

if (c.HasChildren)

{

ClearFields(c);

}

if (c is CheckBox)

{

((CheckBox)c).Checked = false;

}

if (c is RadioButton)

{

((RadioButton)c).Checked = false;

}

if (c is ComboBox)

{

((ComboBox)c).Text = "";

}

}

}

private static void SaveInfo(Control parent, string path)

{

foreach (Control c in parent.Controls)

{

if (c is TextBox txt)

{

File.AppendAllText(path, txt.Name + ": " + txt.Text + "\n");

}

if (c.HasChildren)

{

if (!(c is NumericUpDown))

{

SaveInfo(c, path);

}

}

if (c is CheckBox)

{

if (((CheckBox)c).Checked)

{

File.AppendAllText(path, c.Text + ": " + ((CheckBox)c).Checked.ToString() + "\n");

}

}

if (c is RadioButton)

{

if (((RadioButton)c).Checked)

{

File.AppendAllText(path, c.Text + ": " + ((RadioButton)c).Checked.ToString() + "\n");

}

}

if (c is ComboBox)

{

File.AppendAllText(path, c.Name + ": " + ((ComboBox)c).Text + "\n");

}

if (c is NumericUpDown)

{

File.AppendAllText(path, c.Name + ": " + ((NumericUpDown)c).Text + "\n");

}

}

}

Таблица 23.5 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| fields values | hr\_info.txt |

Источник: собственная разработка

Анализ результата разработки приложения представлен на рисунке 23.5.

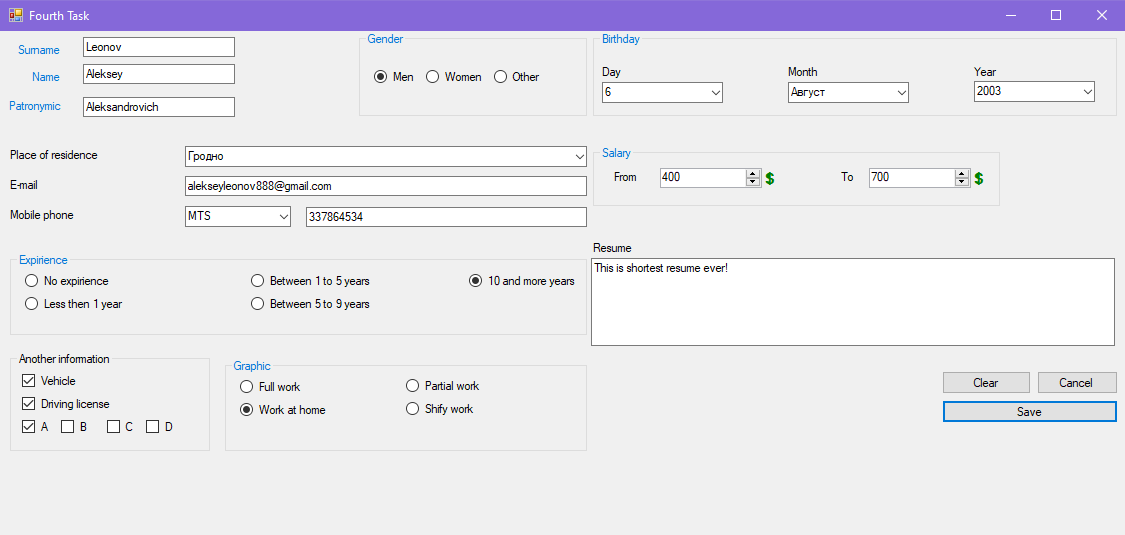


Рисунок 23.5 – Результат работы программы

Источник: собственная разработка

Задание №6. Разместите на форме четыре кнопки (Button). Напишите для каждой обработчик события, который будет менять размеры и местоположение на окне других кнопок.

Листинг программы:

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

{

button2.Size = new System.Drawing.Size(100, 30);

button2.Location = new Point(162, 49);

button3.Size = new System.Drawing.Size(70, 50);

button3.Location = new Point(182, 125);

button4.Size = new System.Drawing.Size(80, 20);

button4.Location = new Point(24, 149); }

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

{

button1.Size = new System.Drawing.Size(75, 23);

button1.Location = new Point(210, 108);

button3.Size = new System.Drawing.Size(85, 33);

button3.Location = new Point(94, 157);

button4.Size = new System.Drawing.Size(95, 43);

button4.Location = new Point(282, 170);

}

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

{

button1.Size = new System.Drawing.Size(30, 50);

button1.Location = new Point(25, 27);

button2.Size = new System.Drawing.Size(30, 30);

button2.Location = new Point(256, 123);

button4.Size = new System.Drawing.Size(50, 20);

button4.Location = new Point(142, 96);

}

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

{

button1.Size = new System.Drawing.Size(100, 50);

button1.Location = new Point(36, 228);

button2.Size = new System.Drawing.Size(88, 38);

button2.Location = new Point(46, 141);

button3.Size = new System.Drawing.Size(65, 20);

button3.Location = new Point(142, 59);

}

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

{

button1.Size = new System.Drawing.Size(75, 23);

button1.Location = new Point(78, 79);

button2.Size = new System.Drawing.Size(75, 23);

button2.Location = new Point(210, 79);

button3.Size = new System.Drawing.Size(75, 23);

button3.Location = new Point(78, 199);

button4.Size = new System.Drawing.Size(75, 23);

button4.Location = new Point(210, 199);

}

Таблица 23.6 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Click | Buttons positions |

Источник: собственная разработка

Анализ результата разработки приложения представлен на рисунке 23.6.

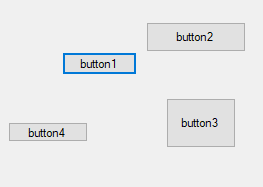


Рисунок 23.6 – Результат работы программы

Источник: собственная разработка

Задание №7. Разместите на форме четыре кнопки (Button). Сделайте на кнопках следующие надписи: «красный», «зеленый», «синий», «желтый». Создайте четыре обработчика события нажатия на данные кнопки, которые будут менять цвет формы в соответствии с текстом на кнопках.

Листинг программы:

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

{

this.BackColor = System.Drawing.Color.Red;

}

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

{

this.BackColor = System.Drawing.Color.Green; }

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

{

this.BackColor = System.Drawing.Color.Blue;

}

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

{

this.BackColor = System.Drawing.Color.Yellow;

}

Таблица 23.7 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Click | Color change |

Источник: собственная разработка

Анализ результата разработки приложения представлен на рисунке 23.7.



Рисунок 23.7 – Результат работы программы

Источник: собственная разработка

Задание №1. Составить программу вычисления для заданных значений x = 2.444, y = 0.869 \* 10-2, z = -0.13 \* 103 арифметического выражения:



Листинг программы:

private void Form1\_Load(object sender, EventArgs e)

{

textBox1.Text = 2.444.ToString();

textBox2.Text = (0.869 \* Math.Pow(10, -2)).ToString();

textBox3.Text = (-0.13 \* Math.Pow(10, 3)).ToString();

}

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

{

double constantE = 2.7182;

textBox4.Text = "Laboratory work 23, Task 7";

double x = double.Parse(textBox1.Text);

textBox4.Text += Environment.NewLine + "X = " + x.ToString();

double y = double.Parse(textBox2.Text);

textBox4.Text += Environment.NewLine + "Y = " + y.ToString();

double z = double.Parse(textBox3.Text);

textBox4.Text += Environment.NewLine + "Z = " + z.ToString();

double h = Math.Round((Math.Pow(x, y + 1) + Math.Pow(constantE, y - 1)) / (1 + x \* Math.Abs(y - Math.Tan(z))) \* (1 + Math.Abs(y - x)) + (Math.Pow(Math.Abs(y - x), 2) / 2) - (Math.Pow(Math.Abs(y - x), 3) / 3), 4);

textBox4.Text += Environment.NewLine +

"Result h: " + h.ToString();

}

Таблица 23.8 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 2.444, 0.00869, -130 | -0.4987 |

Источник: собственная разработка

Анализ результата разработки приложения представлен на рисунке 23.8.

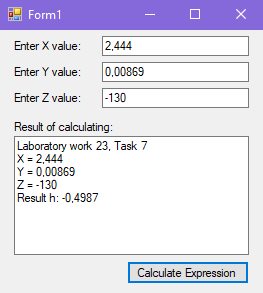


Рисунок 23.8 – Результат работы программы

Источник: собственная разработка

Задание №9. Составить программу вычисления для заданных значений x = 16.55 \* 10-3, y = -2.75, z = 0.15 арифметического выражения:



Листинг программы:

private void Form1\_Load(object sender, EventArgs e)

{

textBox1.Text = (16.55 \* Math.Pow(10, -3)).ToString();

textBox2.Text = (-2.75).ToString();

textBox3.Text = 0.15.ToString();

}

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

double constantE = 2.7182;

textBox4.Text = "Laboratory work 23, Task 9";

double x = double.Parse(textBox1.Text);

textBox4.Text += Environment.NewLine + "X = " + x.ToString();

double y = double.Parse(textBox2.Text);

textBox4.Text += Environment.NewLine + "Y = " + y.ToString();

double z = double.Parse(textBox3.Text);

textBox4.Text += Environment.NewLine + "Z = " + z.ToString();

double B = Math.Round(Math.Sqrt(10 \* (Math.Pow(Math.Sqrt(x), 1 / 3) + Math.Pow(x, y + 2))) \* (Math.Pow(Math.Asin(z), 2) - Math.Abs(x - y)), 4);

textBox4.Text += Environment.NewLine +

"Result w: " + B.ToString();

}

Таблица 23.9 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 0.01655, -2.75, 0.15 | -41.3153 |

Источник: собственная разработка

Анализ результата разработки приложения представлен на рисунке 23.9.

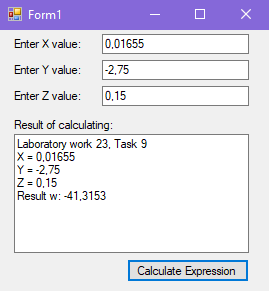


Рисунок 23.9 – Результат работы программы

Источник: собственная разработка

Задание №10. Массив задан датчиком случайных чисел на интервале [-33, 66]. Найти наименьший нечетный элемент.

Листинг программы:

public partial class Form1 : Form

{

private int[] Mas = new int[20];

private int min = 99;

public Form1()

{

InitializeComponent(); }

private void Button1\_Click(object sender, EventArgs e)

{

listBoxArr.Items.Clear();

Random rnd = new Random();

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

{

Mas[i] = rnd.Next(-33, 66);

listBoxArr.Items.Add("Mas[" + i.ToString() + "] = " + Mas[i].ToString());

}

}

private void Button2\_Click(object sender, EventArgs e)

{

listBoxArrResult.Items.Clear();

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

{

if (min > Mas[i])

{

min = Mas[i];

}

}

listBoxArrResult.Items.Add(min.ToString());

}

}

Таблица 23.10 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Array | -31 |

Источник: собственная разработка

Анализ результата разработки приложения представлен на рисунке 23.10.

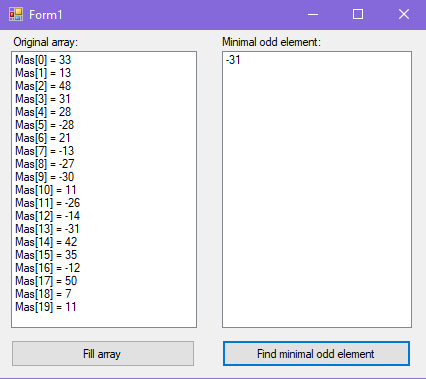


Рисунок 23.10 – Результат работы программы

Источник: собственная разработка

Задание №11. Дана матрица A(3,4). Найти наименьший элемент в каждой строке матрицы. Вывести исходную матрицу и результаты вычислений.

Листинг программы:

private void Button1\_Click(object sender, EventArgs e)

{

dataGridView1.RowCount = 3;

dataGridView1.ColumnCount = 4;

int[,] matrix = new int[3, 4];

int i, j;

Random random = new Random();

for (i = 0; i < 3; i++)

{

for (j = 0; j < 4; j++)

{

matrix[i, j] = random.Next(-6, 10); }

}

for (i = 0; i < 3; i++)

{

for (j = 0; j < 4; j++)

{

dataGridView1.Rows[i].Cells[j].Value = matrix[i, j].ToString();

}

}

for (i = 0; i < 3; i++)

{

int min = 100;

for (j = 0; j < 4; j++)

{

if (matrix[i, j] < min)

{

min = matrix[i, j];

}

}

textBoxResult.Text += min.ToString() + " - Min element in string" + Environment.NewLine;

}

}

Таблица 23.11 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Matrix[3, 4] | -6, -1, -4 |

Источник: собственная разработка

Анализ результата разработки приложения представлен на рисунке 23.11.

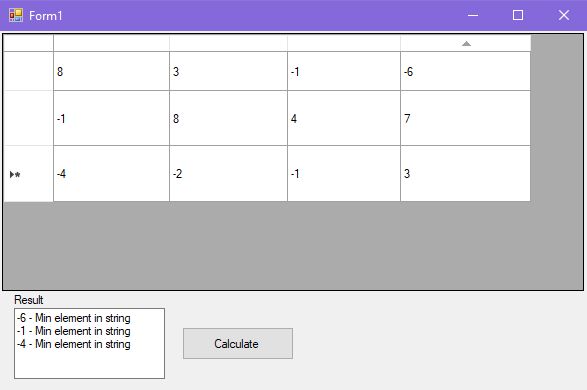


Рисунок 23.11 – Результат работы программы

Источник: собственная разработка