Лабораторная работа №6

Тема: Разработка программ с использованием массивов.

Цель: Научится разрабатывать алгоритмы и реализовывать программы по обработке массивов с применением возможностей класса Array.

Выполнение работы:

Пример 1. Работа с одномерными массивами. Дан массив из n действительных чисел. Вычесть из каждого элемента массива среднее значение массива. Распечатать полученный массив.

Возможный вариант решения:

```
using System;
namespace Prim_Mas1
{
class Program
{
public static void Main(string[] args)
{
double []x = new double[5];
int i, n = x.Length;
Console.WriteLine("Задайте {0} вещественных чисел:", n);
for(i = 0; i < n; i++)
{
Console.Write("x[{0}] = ", i);
x[i] = double.Parse(Console.ReadLine());
}
double s = 0;
for(i = 0; i < n; i++)
s += x[i];
s /= n; // это среднее значение
for(i = 0; i < n; i++)
x[i] -= s;
Console.WriteLine("Массив после обработки:");
foreach(double r in x)
Console.WriteLine(r);
Console.Write("Press any key to continue . . . ");
Console.ReadKey(true);
```

```
}
}
}
```

Пример 2. Работа с двумерными массивами. Пронормировать матрицу действительных чисел A[2x3], т.е. каждый элемент матрицы поделить на максимальное по модулю число.

Возможный вариант решения:

```
using System;
namespace Prim_Matr1
{
class Program
{
public static void Main(string[] args)
{
const int n = 2;
const int m = 3;
double [,]a = new double[n, m];
int i, j;
Console.WriteLine("Задайте матрицу A[{0}*{1}]:", n , m);
for(i = 0; i < n; i++)
for(j = 0; j < m; j++)
a[i, j] =
double.Parse(Console.ReadLine());
double max = Math.Abs(a[0, 0]);
for(i = 0; i < n; i++)
for(j = 0; j < m; j++)
if(Math.Abs(a[i, j]) > max)
max = Math.Abs(a[i, j]);
Console.WriteLine("max = {0}", max);
for(i = 0; i < n; i++)
for(j = 0; j < m; j++)
a[i, j] /= max;
Console.WriteLine("Матрица A[{0}*{1}] после нормирования:", n , m);
for(i = 0; i < n; i++)
{
for(j = 0; j < m; j++)
```

```
Console.Write(a[i,j]+"\t ");
Console.WriteLine();
}
Console.Write("Press any key to continue . . . ");
Console.ReadKey(true);
}
}
}
```

Пример 3. Использование методов класса Array с одномерным массивом.

```
using System;
namespace ConsoleApplicationi
{
class Classl
{
static void Main()
{
int[] a = { 24 , 50, 18, 3, 16, -7, 9, -1 };
PrintArray ("Исходный массив:", a );
Console.WriteLine  (Array.IndexOf( a, 18 ) );
Array.Sort(a);
PrintArray( "Упорядоченный массив:", a );
Console.WriteLine (Array.BinarySearch( a, 18) );
}
public static void PrintArray (string header, int[] a )
{
Console.WriteLine (header );
for ( int i = 0; i < a.Length; ++i )
Console.Write ( "\t" + a[i] );
Console.WriteLine();
}
}
}
```

Пример 4. Разработаем класс для работы с одномерным массивом. Создадим в нём индексатор, позволяющий обращаться к элементу массива по индексу.

```
using System;
namespace Test_Indeksator1
{
class Mas
{
private int n;
private double []x;
public Mas()
{
n=1;
x=new double[n];
x[0]=1;
}
public Mas(int n0)
{
n=n0;
if(n<1) n=1;
x=new double[n];
for (int i = 0; i < n; i++) {
x[i]=1+i;
}
}
public double this[int i]
{
set
{
x[i]=value;
}
get
{
return x[i];
}
}
public void print()
```

```
{
Console.WriteLine("Массив:");
for (int i = 0; i < n; i++) {
Console.WriteLine(x[i]);
}
}
}
class Program
{
public static void Main(string[] args)
{
Console.WriteLine("Тестирование работы
индексаторов");
Mas x = new Mas(3);
// Использование индексатора для записи
x[0]=5;
// Использование индексатора для чтения
Console.WriteLine(x[1]);
x.print();
Console.Write("Press any key to continue . .
. ");
Console.ReadKey(true);
}
}
}
```

Пример 5. Разработаем класс для работы с двумерным массивом (матрицей). Создадим в нём индексатор, позволяющий обращаться к элементу матрицы по индексу.

```
using System;
namespace Test_Indeksator2
{
class Matr
{
private int n;
private int m;
private double [,]x;
```

```
public Matr()
{
n=m=1;
x=new double[n,m];
x[0,0]=1;
}
public Matr(int n0, int m0)
{
n=n0;
if(n<1) n=1;
m=m0;
if(m<1) m=1;
x=new double[n,m];
for (int i = 0; i < n; i++) {
for (int j = 0; j < m; j++) {
x[i,j]=(i+1)*10+j+1;
}
}
}
public double this[int i, int j]
{
set
{
x[i,j]=value;
}
get
{
return x[i,j];
}
}
public void print()
{
Console.WriteLine("Матрица:");
for (int i = 0; i < n; i++) {
for (int j = 0; j < m; j++) {
Console.Write(x[i,j].ToString()+" ");
}
```

```
Console.WriteLine();
}
}
}
class Program
{
public static void Main(string[] args)
{
Console.WriteLine("Тестирование работы
индексаторов");
Matr x = new Matr(3,3);
// использование индексатора для записи
x[0,0]=5;
// использование индексатора для чтения
Console.WriteLine(x[1,1]);
x.print();
Console.Write("Press any key to continue . .
. ");
Console.ReadKey(true);
}
}
}
```

Пример 6. Использование формы при решении задач:

1. В одномерном массиве нулевые элементы удалить, положительные элементы расставить по убыванию, отрицательные – по возрастанию. Получить зависимость затрат машинного времени от размера массива.

2. В матрице удалить строки с последними отрицательными элементами, а затем добавить строку из сумм элементов по столбцам.

Проектирование приложения. Выбор, размещение и задание свойств компонентов. Коды обработчиков событий и функций.

1. Запустите VS.

2. Создайте новый проект.

3. Выделите форму, щелкнув на ней левой кнопкой мыши, и в свойство **Text** впишите *МАССИВЫ*

4. Вначале перенесите на форму многостраничную панель – компонент **tabControl1**. Для размещения остальных компонентов приложения

достаточно использовать две страницы компонента – по заданиям 1 и 2 соответственно.

5. Щелкните на компоненте **tabControl1** правой кнопкой мыши и во всплывшем меню используйте команду *Добавить вкладку*. В свойство **Text** первой страницы впишите *массив,* второй – *матрица*.

6. Перенесите на первую страницу (*массив*) компонент **tabControl2** и, как и в **tabControl1**, создайте две страницы, с надписями *тестирование и использование* и *графики* соответственно.

7. На страницу *тестирование и использование* (рисунке 1) перенесите семь меток **Label** (страница *Стандарт*), в свойство **Text** (надпись) которых соответственно впишите *размер массива, начальный, конечный, шаг, диапазон чисел, макс, мин*; пять компонентов ввода целых чисел – **numeric Up Down1** для задания параметров формируемых массивов. В свойствах **Minimum** и **Maximum** укажите необходимый вам диапазон значений.

8. Для разрешения или запрещения вывода на экран исходных и результирующих массивов, а также графиков полученных зависимостей, в правую верхнюю часть страницы перенесите два компонента-индикатора **checkBox**, в свойство **Text** которых впишите соответственно *в таблицу* и *графики*.

9. Ниже индикаторов **checkBox** поместите панель **Label** для вывода в нее сообщений: *Макс значение не м.б. меньше мин значения!, В массиве только нули!, Кол-во сравнений = <число> Кол-во обменов = <число>*. Очистите свойство **Text** у панели

10. Для вывода массивов на экран при тестировании перенесите на страницу компонент **dataGridView1.**

11. Разместите 2 кнопки с надписями соответственно: **Button1 –** *ПУСК*, **Button2 –** *СБРОС*.

12. Для отображения процесса обработки массива ниже компонента **dataGridView1** поместите компонент **progress Bar1**.

13. Далее поместите компонент **statusStrip1.** Щелкните левой клавишей мышки и в выпадающем меню на компоненте и выберите **StatusLabel.** В свойстве **Text** элемента **toolStripStatusLabel1** введите *«Нули удалить, положительные элементы расставить по убыванию, а отрицательные по возрастанию».*

14. Затраты машинного времени на обработку массива оценивают косвенно – по количествам сравнений и обменов. Для вывода на экран зависимостей количеств сравнений и обменов от размера массива на страницу *графики* (рис. 2.2) компонента **tabControl2** перенесите компоненты **chart1** и **chart2.**

15. Задайте свойства компонентов самостоятельно.

16. После проектирования формы будут иметь следующий вид (рисунки 1 – 2):

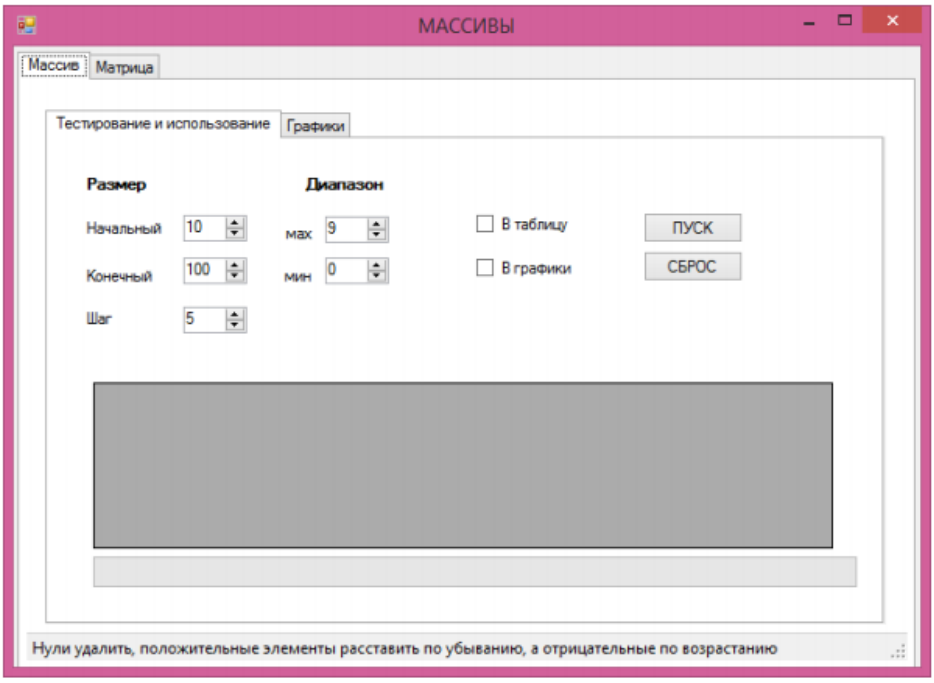

Рисунок 1 - Форма по окончании проектирования (вид 1)

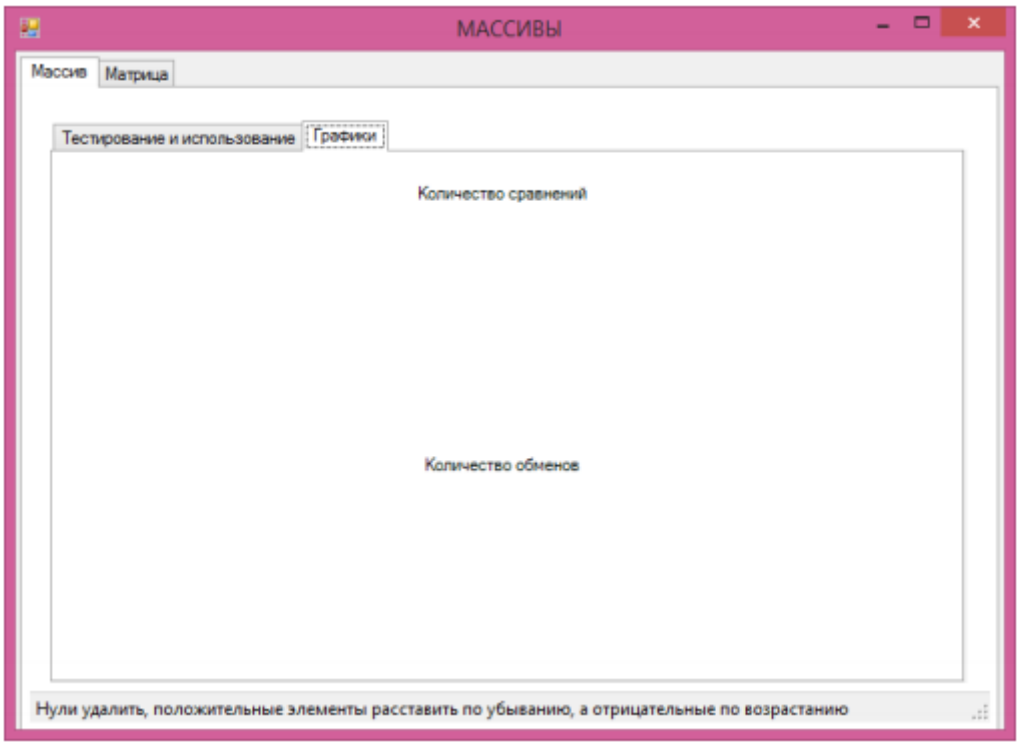

Рисунок 2- Форма по окончании проектирования (вид 2)

17. Перейдем к обработчикам событий – щелчков на кнопках ПУСК1 и СБРОС. В первом обработчике размещен алгоритм формирования и обработки массива по заданию 1, а также вывод сообщений и результатов в виде таблицы и графиков, а во втором – подготовка компонентов к выводу новых сообщений и новых результатов. Перед, после и в обработчике щелчка на кнопке ПУСК напишите:

```
    int i = 0;
private void button1_Click(object sender, EventArgs e)
{
button2.Enabled = false;
label8.Text = "";
if (numericUpDown4.Value < numericUpDown5.Value)
{ label8.Text = "Макс значение не м.б. меньше мин
значения!"; return; }
int count, current = 0;
count = (Convert.ToInt32(numericUpDown2.Value) -
Convert.ToInt32(numericUpDown1.Value)) /
Convert.ToInt32(numericUpDown3.Value) + 1;
for (int n = Convert.ToInt32(numericUpDown1.Value);
```

```
n <= Convert.ToInt32 (numericUpDown2.Value); n +=
Convert.ToInt32 (numericUpDown3.Value))
     {
int[] vptr=new int[n];
Random rand = new Random();
for(int j=0;j<n;j++)
{
vptr[j] = rand. Next(Convert.ToInt32
(numericUpDown5.Value), Convert. ToInt32(numericUpDown4.
Value));
}
if(checkBox1.Checked)
{
dataGridView1.ColumnCount = n+1;
dataGridView1.Rows.Add();
dataGridView1.Rows[i].Cells[0].Value = "Исходный массив";
for (int j = 0; j < n; j++)
{ dataGridView1.Rows[i].Cells[j +
1].Value = vptr[j]; }
i++;
}
sort(vptr,n);
current += 1;
progressBar1.Value = 100 * current / count;
}
}
private void sort(int[] p, int n)
{
int k = 0, sr = 0, obm = 0, m = 0;
for (int j = 0; j < n; j++)
{
if (p[j] == 0) k++;
else p[j - k] = p[j];
}
n -= k;
sr += n;
if (n == 0) { label8.Text = "В массиве одни нули"; return; }
for (m = 0; m < n - 1; m++)
```

```
for (int j = m + 1; j < n; j++)
{
if (p[m] > 0 && p[j] > 0 && p[m] < p[j])
{ swap(ref p[m], ref p[j]); obm++; }
if (p[m] < 0 && p[j] < 0 && p[m] > p[j])
{ swap(ref p[m], ref p[j]); obm++; }
sr += 6;
     }
if (checkBox1.Checked)
{
dataGridView1.AutoResizeColumns();
dataGridView1.Rows.Add();
dataGridView1.Rows[i].Cells[0].Value = "Получен массив";
for (int j = 0; j < n; j++)
{ dataGridView1.Rows[i].Cells[j + 1].Value =
p[j]; }
i++;
}
if (Convert.ToInt32(numericUpDown1.Value) ==
Convert.ToInt32(numericUpDown2.Value))
{ label8.Text = "Количество сравнений=" +
Convert.ToString(sr) + " Количество обменов=" +
Convert.ToString(obm); }
if (checkBox2.Checked)
{
chart1.Series[0].Points.AddXY(n, sr);
chart2.Series[0].Points.AddXY(n, obm);
}
}
void swap(ref int x, ref int y)
{ int z = x; x = y; y = z; }
```

18. В обработчике щелчка на кнопке СБРОС напишите:

```
     private void button1_Click(object sender, EventArgs e)
{
chart1.Series[0].Points.Clear();
chart2.Series[0].Points.Clear();
dataGridView1.Rows.Clear();
dataGridView1.Columns.Clear();
```

```
i = 0;
button2.Enabled = true;
}
```

19. Перенесите на вторую страницу (*матрица*) компонента **tabControl1** (рисунке 3) восемь меток **Label**, в свойство **Text** (надпись) которых впишите значения *размерность, диапазон чисел, макс, мин, исходная матрица, матрица-результат*; три компонента ввода целых чисел – **numeric UpDown** для задания параметров формируемых матриц.

20. Перенесите также две кнопки **Button** с надписями *ПУСК, СБРОС*.

21. Сюда же поместите панель **Label** для вывода в панель сообщений: *Макс не м.б. меньше мин!, В матрице удалены все строки!, В матрице нет удаленных строк!, В матрице удалено <кол-во> строк(и)!*. Очистите свойство **Text** у панели

22. Кроме того, на этой странице разместите две таблицы – компоненты **dataGridView** – для вывода матриц – исходной и матрицы-результата.

23. Далее поместите компонент **statusStrip2.** Щелкните левой клавишей мышки и в выпадающем меню на компоненте и выберите **StatusLabel.** В свойстве **Text** элемента **toolStripStatusLabel2** введите *«Строки с "–" на конце - удалить, а затем добавить строку из сумм по столбцам».*

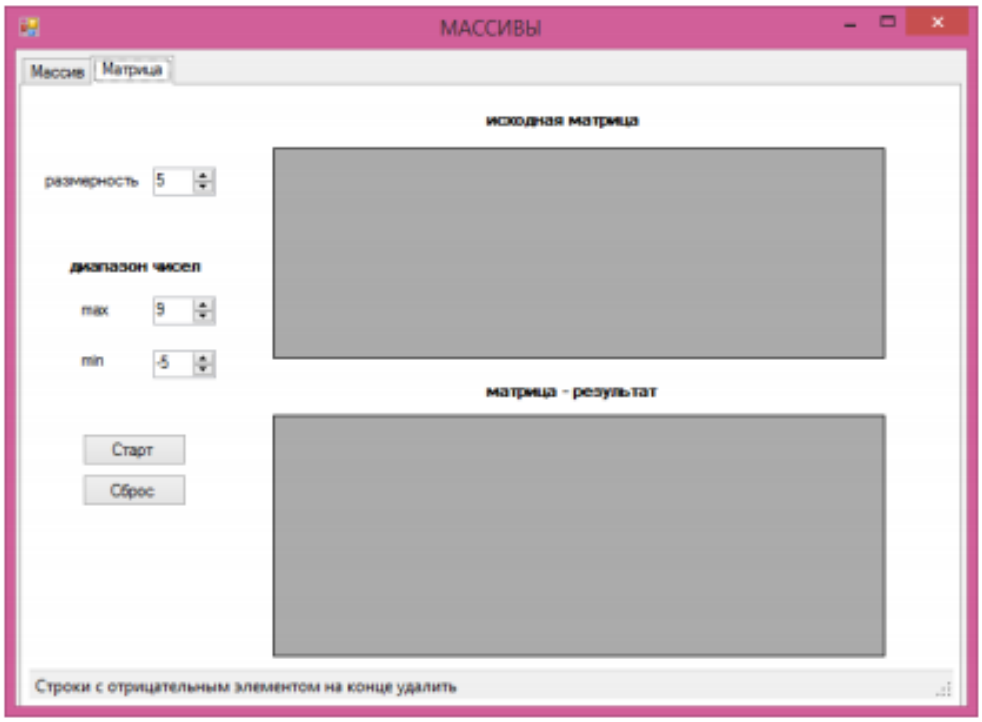

Рисунок 3 - Форма по окончании проектирования (вид 2)

25. В обработчике щелчка на кнопке *ПУСК* находится алгоритм формирования и обработки динамической матрицы согласно заданию 2 с выводом результатов и сообщений, а в обработчике щелчка на кнопке *СБРОС* – зачистка выведенных результатов и сообщений:

```
     int n, m;
private void button3_Click(object sender, EventArgs e)
{
int i, j, k, q;
button3.Enabled = false;
if (numericUpDown8.Value < numericUpDown9.Value)
{ label9.Text = "Макс значение не м.б. меньше
мин значения!"; return; }
n = Convert.ToInt32(numericUpDown6.Value);
m = Convert.ToInt32(numericUpDown6.Value);
int[,] ptr;
ptr = new int[m, n];
Random rand = new Random();
dataGridView2.AutoResizeColumns();
dataGridView2.ColumnCount = n;
for (i = 0; i < n; i++)
{
dataGridView2.Rows.Add();
for (j = 0; j < m; j++)
{
ptr[i,j] =
rand.Next(Convert.ToInt32(numericUpDown9.Value),
Convert.ToInt32(numericUpDown8.Value));
dataGridView2.Rows[i].Cells[j].Value =
ptr[i,j];
}
}
q = 0;
k = 0;
if (ptr[n - 1, m - 1] < 0) k++;
for(q=0;q<n-1;q++)
{
if (ptr[q, m - 1] < 0)
```

```
{
k++;
for (i = q; i < n - 1; i++)
{
     for (j = 0; j < m; j++) ptr[i, j] =
ptr[i + 1, j];
} q
--;
}
if (k + q+1 == n) { break; }
}
if (k == n) { label9.Text = "В матрице удалены все
строки"; return; }
if (k == 0) { label9.Text = "В матрице нет удаленных строк";
return; }
label9.Text = "В матрице удалено " + k + " строк(и)";
for (j = 0; j < m; j++)
{
ptr[n - k, j] = 0;
for (i = 0; i < n - k; i++)
{
ptr[n-k, j] += ptr[i, j];
}
}
dataGridView3.AutoResizeColumns();
dataGridView3.ColumnCount = n;
for (i = 0; i <= n-k; i++)
{
dataGridView3.Rows.Add();
for (j = 0; j < m; j++)
{
dataGridView3.Rows[i].Cells[j].Value =
ptr[i, j];
}
}
}
private void button4_Click(object sender, EventArgs e)
{
```

```
dataGridView2.Rows.Clear();
dataGridView2.Columns.Clear();
dataGridView3.Rows.Clear();
dataGridView3.Columns.Clear();
     button3.Enabled = true;
label9.Text = "";
}
```

26. По окончании проектирования файл LR_1.cpp будет выглядеть так:

```
     using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Windows.Forms.DataVisualization.Charting;
namespace massiv
{
public partial class Form1 : Form
{
public Form1()
{
InitializeComponent();
} i
nt i = 0;
private void button1_Click(object sender, EventArgs e)
{
button2.Enabled = false;
label8.Text = "";
if (numericUpDown4.Value < numericUpDown5.Value)
{ label8.Text = "Макс значение не м.б. меньше мин
значения!"; return; }
int count, current = 0;
count = (Convert.ToInt32(numericUpDown2.Value) -
Convert.ToInt32(numericUpDown1.Value)) /
Convert.ToInt32(numericUpDown3.Value) + 1;
```

```
for (int n =
Convert.ToInt32(numericUpDown1.Value); n <=
Convert.ToInt32(numericUpDown2.Value); n +=
Convert.ToInt32(numericUpDown3.Value))
     {
int[] vptr=new int[n];
Random rand = new Random();
for(int j=0;j<n;j++)
{
vptr[j] =
rand.Next(Convert.ToInt32(numericUpDown5.Value),
Convert.ToInt32(numericUpDown4.Value));
}
if(checkBox1.Checked)
{
dataGridView1.ColumnCount = n+1;
dataGridView1.Rows.Add();
dataGridView1.Rows[i].Cells[0].Value =
"Исходный массив";
for (int j = 0; j < n; j++)
{ dataGridView1.Rows[i].Cells[j +
1].Value = vptr[j]; }
i++;
}
sort(vptr,n);
current += 1;
progressBar1.Value = 100 * current / count;
}
}
private void sort(int[] p, int n)
{
int k = 0, sr = 0, obm = 0, m = 0;
for (int j = 0; j < n; j++)
{
if (p[j] == 0) k++;
else p[j - k] = p[j];
}
n -= k;
```

```
sr += n;
if (n == 0) { label8.Text = "В массиве одни нули"; return; }
for (m = 0; m < n - 1; m++)
for (int j = m + 1; j < n; j++)
{
if (p[m] > 0 && p[j] > 0 && p[m] < p[j])
{ swap(ref p[m], ref p[j]); obm++; }
if (p[m] < 0 && p[j] < 0 && p[m] > p[j])
{ swap(ref p[m], ref p[j]); obm++; }
sr += 6;
}
if (checkBox1.Checked)
{
dataGridView1.AutoResizeColumns();
dataGridView1.Rows.Add();
     dataGridView1.Rows[i].Cells[0].Value = "Получен массив";
for (int j = 0; j < n; j++)
{ dataGridView1.Rows[i].Cells[j + 1].Value =
p[j]; }
i++;
}
if (Convert.ToInt32(numericUpDown1.Value) ==
Convert.ToInt32(numericUpDown2.Value))
{ label8.Text = "Количество сравнений=" +
Convert.ToString(sr) + " Количество обменов=" +
Convert.ToString(obm); }
if (checkBox2.Checked)
{
chart1.Series[0].Points.AddXY(n, sr);
chart2.Series[0].Points.AddXY(n, obm);
}
}
void swap(ref int x, ref int y)
{ int z = x; x = y; y = z; }
private void button2_Click(object sender, EventArgs e)
{
chart1.Series[0].Points.Clear();
chart2.Series[0].Points.Clear();
```

```
dataGridView1.Rows.Clear();
dataGridView1.Columns.Clear();
i = 0;
button2.Enabled = true;
}
int n, m;
private void button3_Click(object sender, EventArgs e)
{
int i, j, k, q;
button3.Enabled = false;
if (numericUpDown8.Value < numericUpDown9.Value)
{ label9.Text = "Макс значение не м.б. меньше
мин значения!"; return; }
n = Convert.ToInt32(numericUpDown6.Value);
m = Convert.ToInt32(numericUpDown6.Value);
int[,] ptr;
     ptr = new int[m, n];
Random rand = new Random();
dataGridView2.AutoResizeColumns();
dataGridView2.ColumnCount = n;
for (i = 0; i < n; i++)
{
dataGridView2.Rows.Add();
for (j = 0; j < m; j++)
{
ptr[i,j] =
rand.Next(Convert.ToInt32(numericUpDown9.Value),
Convert.ToInt32(numericUpDown8.Value));
dataGridView2.Rows[i].Cells[j].Value =
ptr[i,j];
}
}
q = 0;
k = 0;
if (ptr[n - 1, m - 1] < 0) k++;
for(q=0;q<n-1;q++)
{
if (ptr[q, m - 1] < 0)
```

```
{
k++;
for (i = q; i < n - 1; i++)
{
for (j = 0; j < m; j++) ptr[i, j] =
ptr[i + 1, j];
} q
--;
}
if (k + q+1 == n) { break; }
}
if (k == n) { label9.Text = "В матрице удалены все
строки"; return; }
if (k == 0) { label9.Text = "В матрице нет удаленных строк";
return; }
label9.Text = "В матрице удалено " + k + " строк(и)";
for (j = 0; j < m; j++)
{
ptr[n - k, j] = 0;
for (i = 0; i < n - k; i++)
     {
ptr[n-k, j] += ptr[i, j];
}
}
dataGridView3.AutoResizeColumns();
dataGridView3.ColumnCount = n;
for (i = 0; i <= n-k; i++)
{
dataGridView3.Rows.Add();
for (j = 0; j < m; j++)
{
dataGridView3.Rows[i].Cells[j].Value =
ptr[i, j];
}
}
}
private void button4_Click(object sender, EventArgs e)
{
```

```
dataGridView2.Rows.Clear();
dataGridView2.Columns.Clear();
dataGridView3.Rows.Clear();
dataGridView3.Columns.Clear();
button3.Enabled = true;
label9.Text = "";
}
}
}
```

Тестирование и использование приложения

1. Запустите приложение на выполнение, нажав быстрые кнопки *Сохранить все* и *Запуск*.

2. Подготовьте приложение к тестированию задания 1, щелкнув на закладке *массив*, а затем *тестирование и использование* (рисунок 1). Включите индикатор *вывод в таблицу*.

3. Пользуясь *ПУСК* и *СБРОС*, убедитесь в работоспособности приложения с параметрами массива, заданными по умолчанию (рисунок 4).

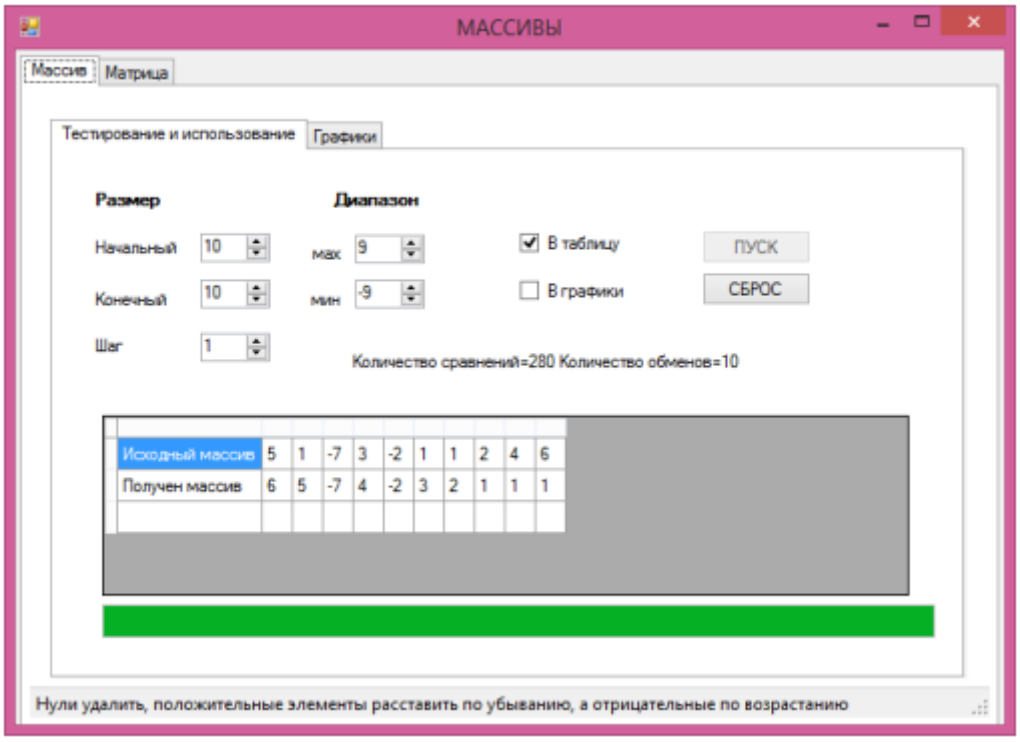

Рисунок 4 - Результаты тестирования по заданию 1

4. Изменяя диапазон значений элементов массива, убедитесь в работоспособности приложения в случаях: а) максимальное значение меньше минимального, б) в массиве только нули, в)

в массиве только положительные элементы, г) в массиве только отрицательные элементы, д) массив состоит из равных по величине элементов (положительных и отрицательных).

5. Перейдите к использованию приложения для построения графиков зависимостей затрат машинного времени от размера массива при разных диапазонах значений элементов массива. Включите индикатор *графики*. При заданных по умолчанию остальных параметрах задайте конечный размер массива – 1000 и нажмите *ПУСК*. По окончании обработки массива получите на вкладе *графики* результат, представленный на рисунок 5.

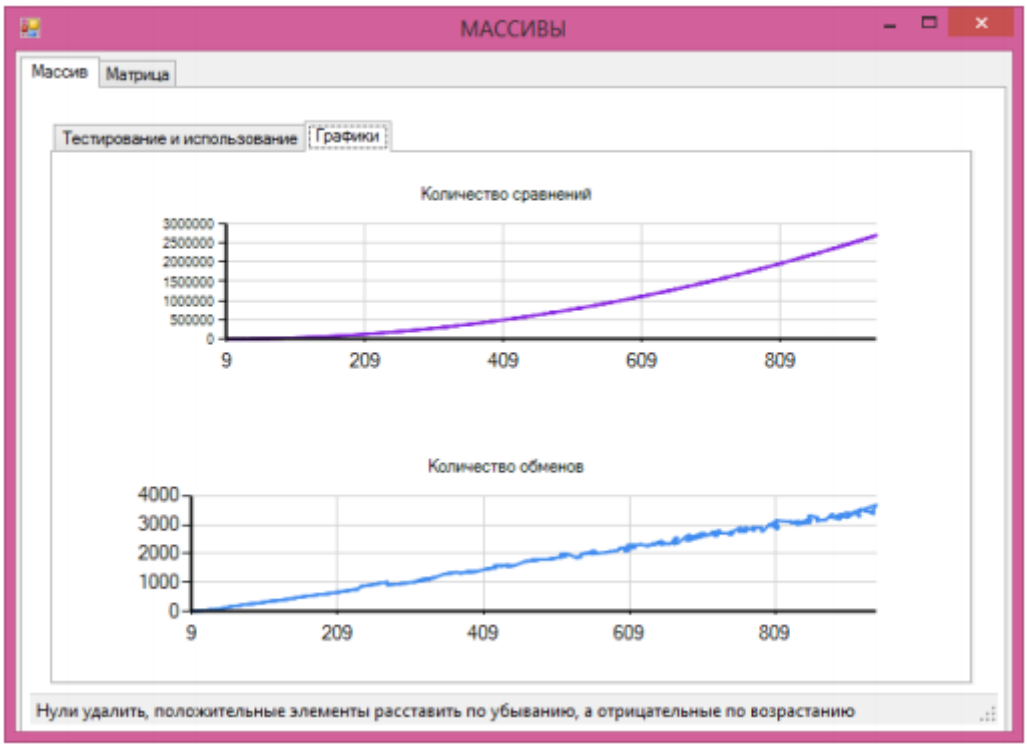

Рисунок 5 - Зависимости затрат машинного времени от размера массива

6. Установите влияние диапазона значений элементов и размера массива на ход указанных зависимостей. В частности, установите параметры диапазона такими, чтобы количество обменов было нулевым при любом размере массива.

7. Подготовьте приложение к тестированию и выполнению задания 2, щелкнув на закладке *матрица* (рисунок 3).

8. Протестируйте приложение для заданных по умолчанию параметров матрицы, щелкая на кнопках *ПУСК* и *СБРОС* (рисунок 6).

9. Изменяя диапазон значений элементов матрицы, убедитесь в работоспособности приложения в случаях: а) максимальное значение меньше минимального, б) удалены все строки, в) нет удаленных строк. Установите влияние диапазона на среднее количество удаленных строк.

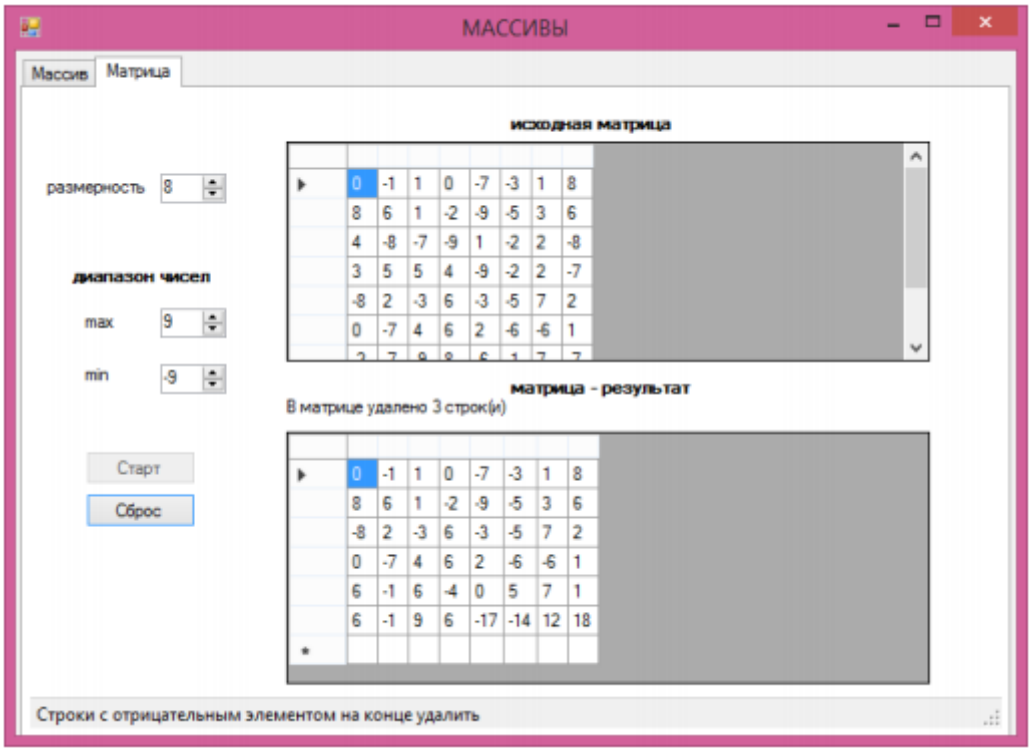

Рисунок 6 - Результаты тестирования по заданию 2

10. Проделайте то же самое, варьируя размерами матрицы.

11. Для завершения работы щелкните на кнопке формы “Закрыть” и выйдите из среды VS.

Варианты индивидуальных заданий

Задание 1. Поиск в массиве.

1. Найти в массиве все числа, составленные из одних и тех же цифр

2. В заданной последовательности целых чисел найти максимально длинную подпоследовательность чисел такую, что каждый последующий элемент подпоследовательности делился нацело на предыдущий.

3. Дан целочисленный массив размера N. Определить максимальное количество его одинаковых элементов.

4. Дан целочисленный массив размера N. Назовем серией группу подряд идущих одинаковых элементов, а длиной серии — количество этих элементов (длина серии может быть равна 1). Вывести строку, содержащую длины всех серий исходного массива.

5. Дан целочисленный массив размера N. Вывести вначале все его четные элементы, а затем — нечетные.

6. Дано N целых чисел. Требуется выбрать из них три таких числа, произведение которых максимально.

7. Дан массив ненулевых целых чисел размера N. Проверить, чередуются ли в нем четные и нечетные числа. Если чередуются, то вывести 0, если нет, то вывести номер первого элемента, нарушающего закономерность.

8. Дан массив из координат x N точек на прямой. Найти такую точку из данного множества, сумма расстояний от которой до остальных его точек минимальна, и саму эту сумму.

9. Дан целочисленный массив размера N. Преобразовать его, прибавив к четным числам последний элемент. Первый и последний элементы массива не изменять.

10.Дан целочисленный массив A размера 11. Вывести номер первого из тех его элементов A[i], которые удовлетворяют двойному неравенству: A[1] < A[i] < A[10]. Если таких элементов нет, то вывести 0.

Задание 2.

1. В матрице удалить строки, содержащие нули, а затем добавить строку, элементы которой равны произведениям элементов в соответствующих столбцах.

2. В матрице удалить столбцы с нулевыми элементами ниже главной диагонали, а затем добавить столбец, элементы которого равны суммам элементов в соответствующих строках.

3. В матрице удалить столбцы, в которых количество отрицательных элементов превышает заданное, а затем в качестве первого добавить столбец с максимальными элементами по строкам.

4. В матрице удалить строки с нулевыми элементами выше главной диагонали, а затем в качестве третьей добавить строку, элементы которой равны разностям соответствующих элементов первой и второй строк

5. В матрице удалить столбцы с положительными суммами элементов, а затем в качестве первого вставить столбец из минимальных элементов соответствующих строк.

6. В матрице удалить строку с минимальным произведением элементов, а затем в качестве второй добавить строку, элементы которой равны разностям элементов первой и последней строк.

7. В матрице удалить строки, последние элементы которых отрицательны, а затем в качестве первой добавить строку из элементов заданного массива.

8. В матрице удалить первую и последнюю строки, а затем добавить строку из максимальных элементов соответствующих столбцов.

9. В матрице удалить столбцы с отрицательной суммой элементов, а затем добавить столбец из минимальных элементов соответствующих строк.

10. В матрице удалить столбцы с максимальным и минимальным элементами матрицы, а затем на место первого добавить столбец из произведений элементов соответствующих строк.

11. В матрице удалить строки с элементами на главной диагонали, превышающими заданную величину, а затем в качестве первой вставить строку из максимальных элементов соответствующих столбцов.

12. В матрице удалить столбцы с положительными третьими элементами, а затем добавить столбец из элементов заданного массива.

13. Вычислить m-норму матрицы.

14. Вычислить l-норму матрицы.

15. Вычислить k-норму матрицы.

16. В матрице удалить строки с положительными последними элементами, а затем добавить строку из минимальных элементов по соответствующим столбцам.

17. Сформировать массивы из элементов в седловых точках матриц.

Примечание: в седловой точке элемент является минимальным в строке и максимальным в столбце.

18. Приняв за характеристику столбца матрицы сумму модулей его отрицательных нечетных элементов, расположить столбцы матриц в соответствии с ростом характеристик.

19. Выполнить операции сглаживания матриц. Операция сглаживания дает матрицу того же размера, каждый элемент которой получается как среднее арифметическое соседей соответствующего элемента матрицы. Соседями элемента a_{ij} в матрице называют элементы a_{kl} с $i-1 \le k \le i+1, j-1 \le l \le j+1, (k,l) \ne (i,j)$.

20. Элемент матрицы называется локальным минимумом, если он строго меньше всех имеющихся у него соседей. Сформировать массив из количеств локальных минимумов обработанных матриц.

21. Осуществить циклический сдвиг элементов прямоугольных матриц на *n* элементов вправо или вниз (в зависимости от указанного режима); *n* может быть больше количества элементов в строке или столбце матрицы.

22. Осуществить циклический сдвиг элементов квадратных матриц вправо на *k* элементов таким образом: элементы первой строки сдвигаются в последний столбец сверху вниз, из него – в последнюю строку справа налево, из неѐ – в первый столбец снизу вверх, из него – в первую строку; для остальных элементов – аналогично.

23. Сортировать матрицы следующим образом: помещать элементы на диагонали, начиная с главной диагонали, в порядке убывания.

24. В матрице удалить строку с минимальным элементом матрицы, а затем добавить отсортированную по возрастанию предыдущую строку.

25. В матрицах проверять указанную строку на ортогональность остальным строкам.

26. В матрице удалить строку с максимальным по модулю элементом матрицы, а затем в качестве первой добавить строку, элементы которой равны суммам модулей элементов в соответствующих столбцах.

27. Проверять матрицы на попарную ортогональность строк; обработку каждой матрицы завершать выводом списка попарно ортогональных строк.

28. Характеристикой строки матрицы назовѐм сумму еѐ отрицательных четных элементов. Расположить строки в соответствии с убыванием характеристик.

29. В матрице удалить столбец с максимальным элементом матрицы, а затем вставить заданный столбец перед столбцом с минимальным элементом полученной матрицы.

30. В матрицу добавить строку из элементов заданного массива, а затем удалить строки с положительной суммой элементов.

Шкала оценивания индивидуальных заданий

Примеры 1-6	1-5 балла
Примеры 1-6 + Задание 1	1-7 баллов
Примеры 1-6 + Задание 1 + Задание 2	1-10 баллов

Содержание отчета:

1. Номер и тема лабораторной работы.
2. Цель лабораторной работы.
3. Техническое оснащение.
4. Скриншоты выполнения примеров
5. При выполнении индивидуальных заданий в отчет внести изображение кода программы и окно выполнения программы.
6. При выполнении программы с использованием формы, в отчет внести изображением формы с использованными элементами и модифицированной формы, так же окно программы после выполнения.
7. Вывод по лабораторной работе