Министерство цифрового развития, связи и массовых коммуникаций

ордена трудового красного знамени

федеральное государственное бюджетное образовательное

учреждение высшего образования

«Московский технический университет связи и информатики»

Кафедра «Информатика»

Лабораторные работы

по дисциплине

«Алгоритмизация и программирование»

Выполнил:

студент гр.БЭИ2102

Ахтямов Б. Р.

Проверил:

Научный руководитель

доцент кафедры “Информатика”

Юсков И. О.

Москва 2022

**Лабораторная работа №1**

**«Запись одномерных массивов в PDF-документ. Запись одномерных массивов в редактор Блокнот»**

1. **Постановка задачи**

Дан динамический целочисленный массив. Опишите алгоритм, позволяющий найти и вывести наименьшую нечётную сумму двух соседних элементов массива.

Разработайте метод, который формирует новый массив, из тех элементов исходного массива, значения которых больше найденного числа.

Кроме того, необходимо реализовать алгоритм, позволяющий найти и вывести сумму всех трёхзначных чисел, десятичная запись которых оканчивается на 9, но не на 99, если их нет вывести -1. После отсортировать массив двумя видами сортировок простой вставкой и простым обменом.

Для выполнения задания следует разобрать нижеперечисленные методы:

* Метод ***Zapis\_v\_PDF(int lenght, int[] mas, int count, int[] rezmas***, предназначенный для вывода массивов в PDF-документ;
* Метод ***Zapis\_v\_bloknot(int length, int[] mas, int count, int[] rezmas)***, предназначенный для записи массивов в редактор Блокнот;
* Метод ***Vvod()***, предназначенный для ввода данных;
* Метод ***Vivod()***, предназначенный для вывода данных;
* Метод ***arrayGenerate()***, предназначенный для создания массива, состоящего из случайного набора чисел;
* Метод ***output\_mas()***, предназначенный для вывода массива в таблицу;
* Метод ***set\_mas()***, формирующий новый массив, из тех элементов исходного массива, которые больше наименьшей нечетной суммы соседних элементов;
* Метод ***poisk()***, созданный для поиска пары соседних чисел с наименьшей нечётной суммой;
* Метод ***VivodDGV***(), созданный для вывода значений больших наименьшей суммы с нечётным значением;
* Метод ***InputInt()***, выводит число из ***textbox***;
* Метод ***Sort\_prost***, сортирующая простым методом;
* Метод ***Sort\_obmen***, сортирующая обменом;
* Метод ***Zad\_3***, выполняющая задание, описанное выше;
* Метод ***Zad\_3***, выполняющая задание, описанное выше;

1. **Разработка проекта**
   1. Разработка графического интерфейса

Интерфейс пользователя представлена на рисунке 1.

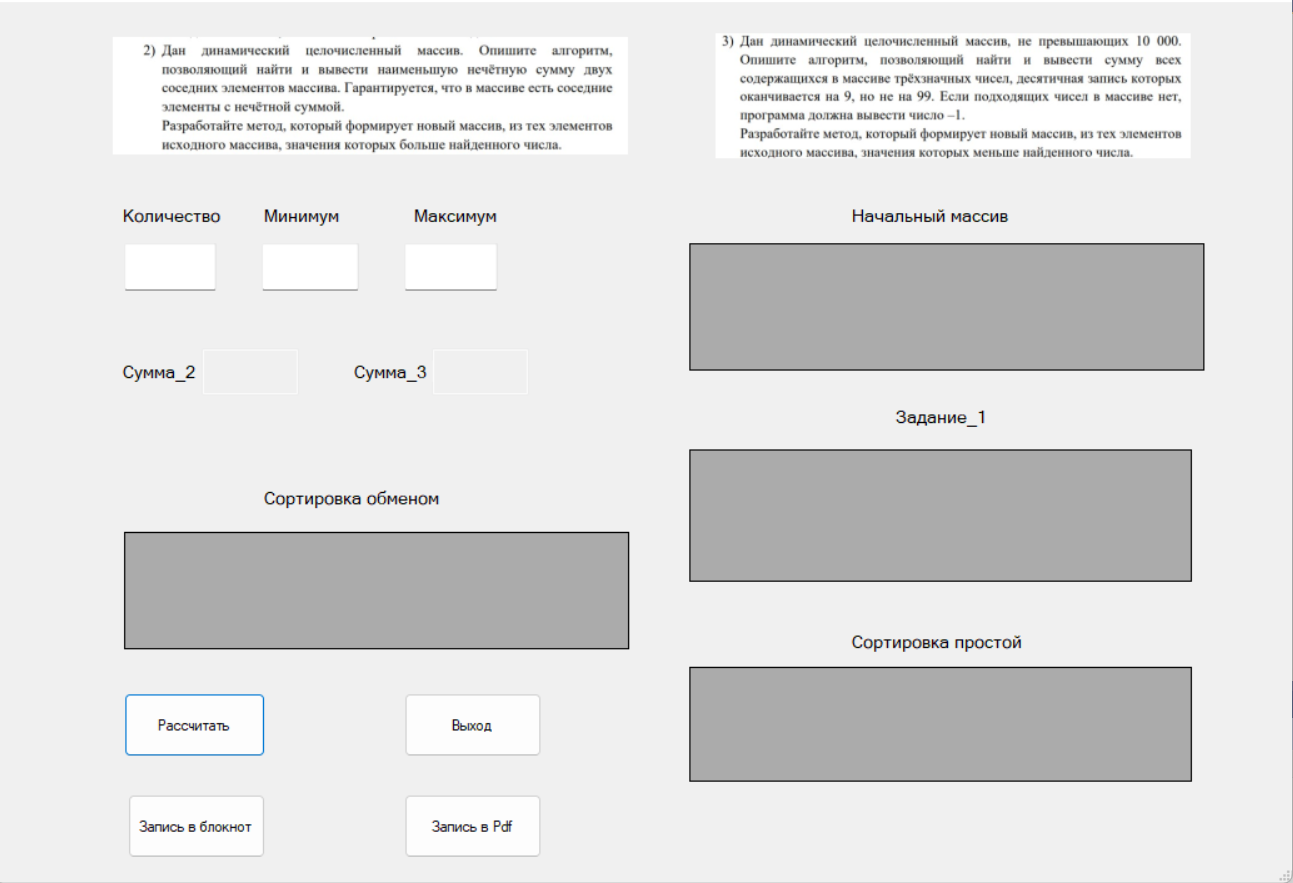


Рисунок 1 - Интерфейс пользователя

* 1. Разработка схемы алгоритмов

Блок-схемы событийной части проекта представлен на рисунках 2 - 5.

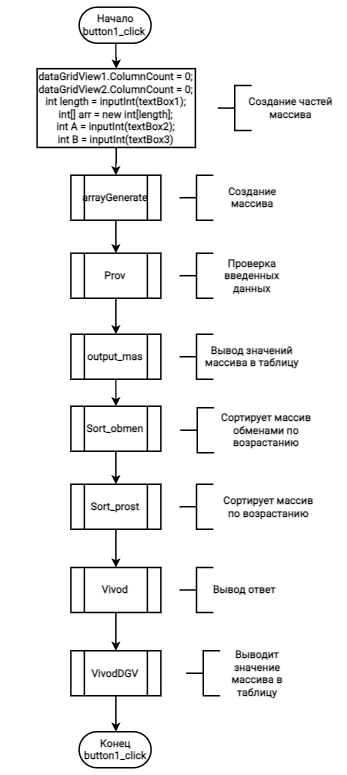


Рисунок 2 - Схема алгоритмов процедур **button1\_Click**

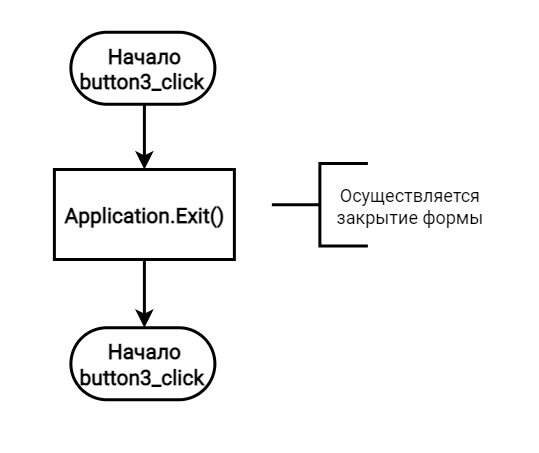
****

Рисунок 3 - Схема алгоритмов процедур **button3\_Click**

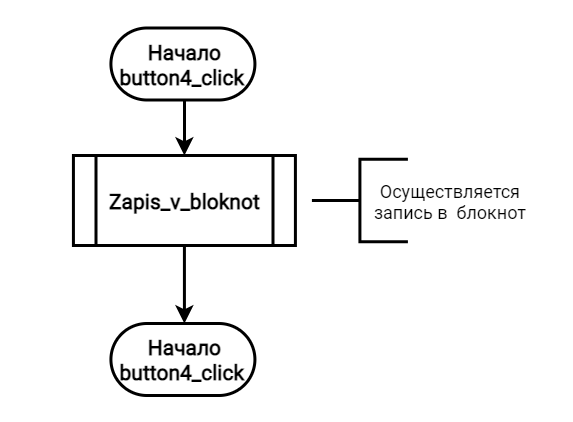


Рисунок 4 - Схема алгоритмов процедур **button4\_Click**

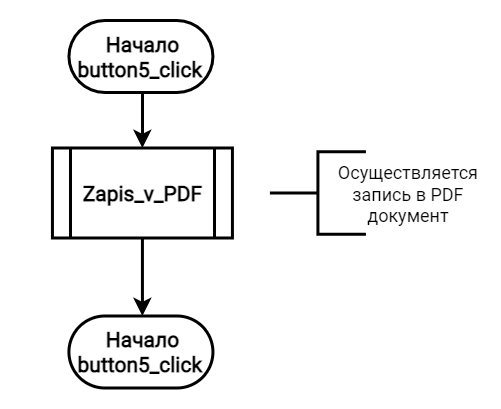
****

Рисунок 5 - Схема алгоритмов процедур **button5\_Click**

Проверка ввода данных будет применяться на все элементы управления ***Textbox***.

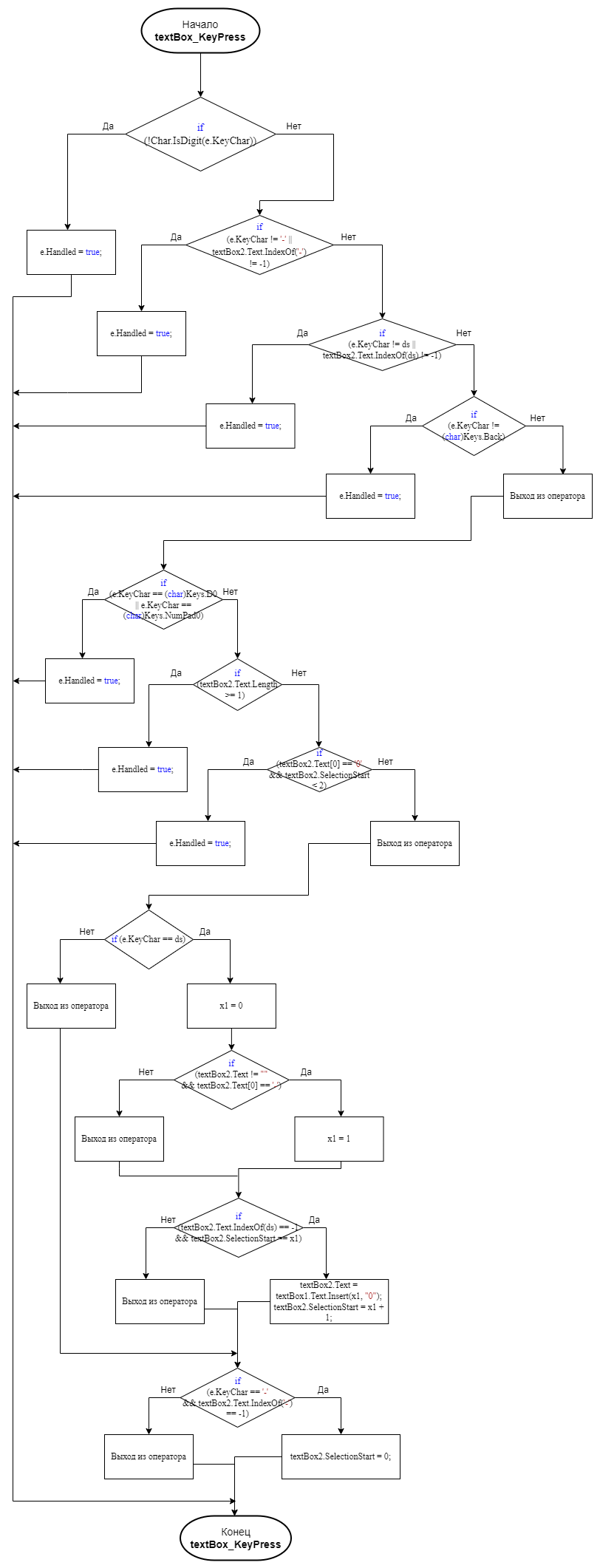


Рисунок 6 - Схема алгоритмов процедур **textbox\_KeyPress**

Блок-схемы функций, подключенных из библиотеки **3sem**, представлены на рисунках 7 – 13.

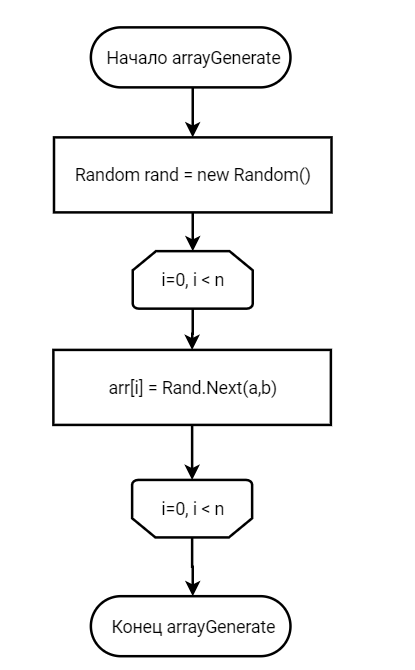


Рисунок 7 - Схема алгоритмов процедур **arrayGenerate()**

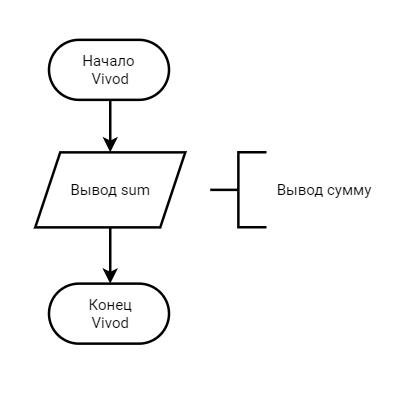


Рисунок 8 - Схема алгоритмов процедур **Vivod()**

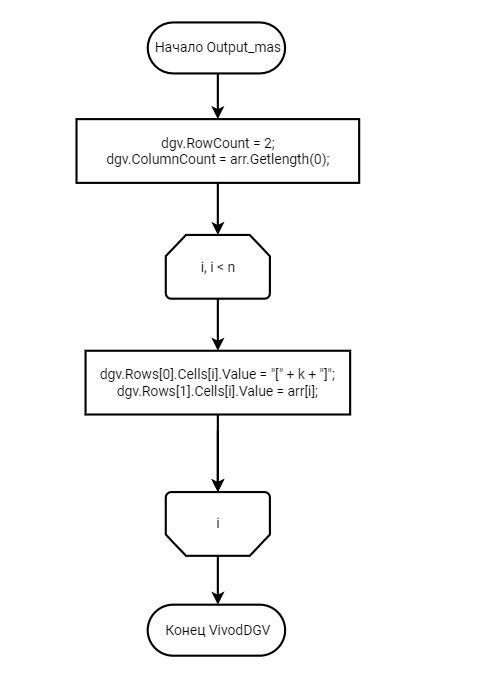
****

Рисунок 9 - Схема алгоритмов процедур **Output\_mas()**

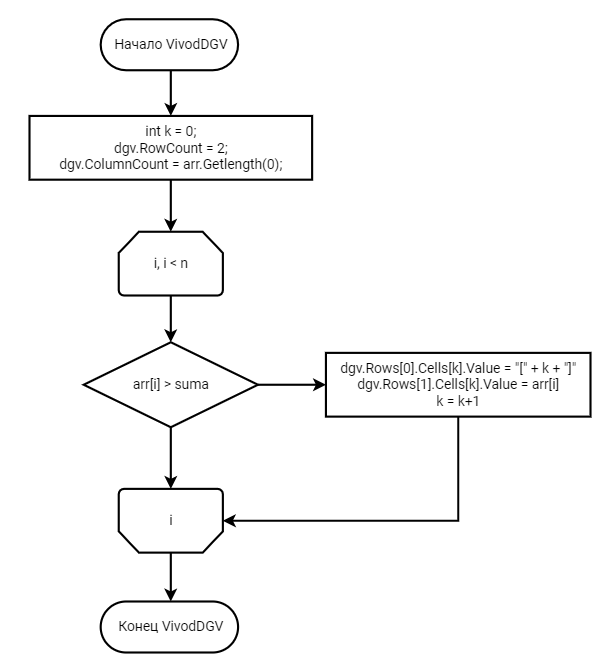
****

Рисунок 10 - Схема алгоритмов процедур **VivodDGV()**

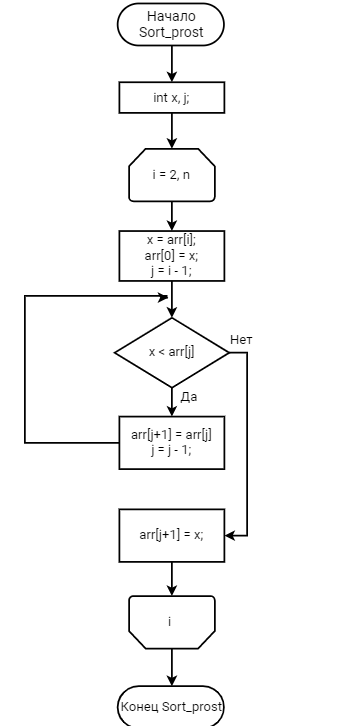
****

Рисунок 11 - Схема алгоритмов процедур **Sort\_prost()**

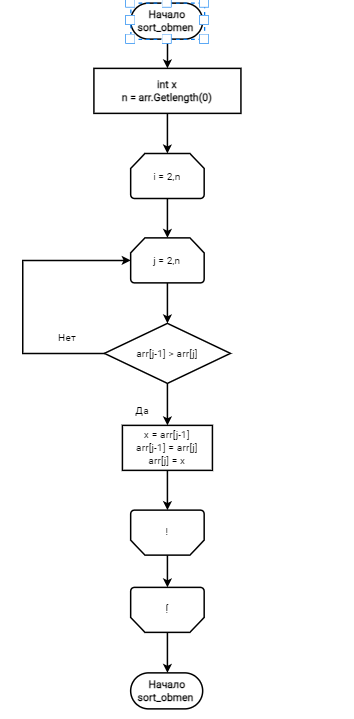
****

Рисунок 12 - Схема алгоритмов процедур **Sort\_obmen()**

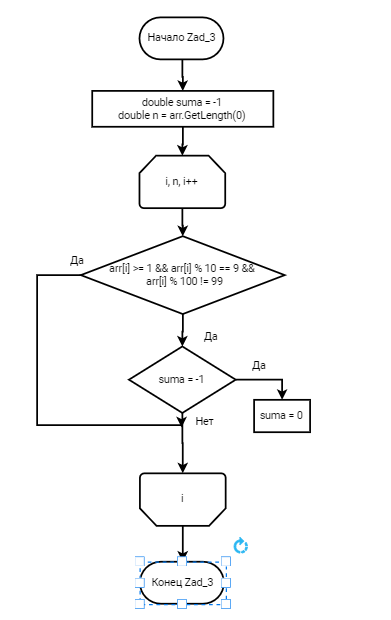
****

Рисунок 13 - Схема алгоритмов процедур **Zad\_3()**

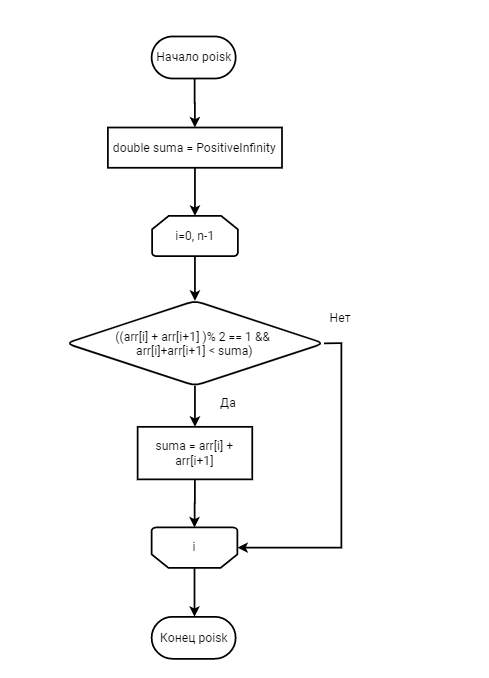


Рисунок 14 - Схема алгоритмов процедур **poisk()**

* 1. Написание программного кода

В основной части проекта подпрограммы-методы вызываются из библиотеки ***3sem*** класса ***Class1***.

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace AnP\_proj

{

public partial class Form9 : Form

{

public static int[] arr;

public static int[] mas;

public static int[] rezmas;

public static int length;

public static int count;

public Form9()

{

InitializeComponent();

}

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

{

dataGridView1.Rows.Clear();

dataGridView2.Rows.Clear();

dataGridView1.ColumnCount = 0;

dataGridView2.ColumnCount = 0;

//vvod parametrov

length = \_3sem.Class1.inputInt(textBox1); //count

if (length <= 0)

{

MessageBox.Show("Ошибка, длина должна быть больше нуля");

length = \_3sem.Class1.inputInt(textBox1); //count

}

//generation and print array

arr = new int[length];

int A = \_3sem.Class1.inputInt(textBox2); //start

int B = \_3sem.Class1.inputInt(textBox3); //stop

if (A > B)

{

MessageBox.Show("Ошибка, верхняя граница не должна превышать нижнюю");

A = \_3sem.Class1.inputInt(textBox2); //start

B = \_3sem.Class1.inputInt(textBox3); //stop

}

\_3sem.Class1.arrayGenerate(ref arr, length, A, B);

//\_3sem.Class1arrayOutput(ref arr, lenght, ref dataGridView1);

\_3sem.Class1.output\_mas(ref dataGridView1, arr);

double suma = \_3sem.Class1.poisk(arr);

\_3sem.Class1.Vivod(textBox4, suma);

mas = new int[length];

\_3sem.Class1.VivodDGV(ref arr, suma, dataGridView2, ref mas);

double suma\_2 = \_3sem.Class1.Zad\_3(arr);

\_3sem.Class1.Vivod(textBox5, suma\_2);

\_3sem.Class1.Sort\_prost(ref arr);

\_3sem.Class1.output\_mas(ref dataGridView3, arr);

\_3sem.Class1.Sort\_obmen(ref arr);

\_3sem.Class1.output\_mas(ref dataGridView4, arr);

}

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

{

Application.Exit();

}

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

{

\_3sem.Class1.Zapis\_v\_bloknot(ref arr, ref mas);

}

private void textBox3\_KeyPress(object sender, KeyPressEventArgs e)

{

\_3sem.Class1.Prof(textBox3, e);

}

private void textBox2\_KeyPress(object sender, KeyPressEventArgs e)

{

\_3sem.Class1.Prof(textBox2, e);

}

private void textBox1\_KeyPress(object sender, KeyPressEventArgs e)

{

\_3sem.Class1.Prof(textBox1, e);

}

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

{

\_3sem.Class1.Zapis\_v\_PDF(length, arr, mas);

}

}

}

Все подпрограммы-методы размещаются в классе ***Class1***.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

using System.Globalization;

using Microsoft.VisualBasic;

using System.Diagnostics;

using iTextSharp.text;

using iTextSharp.text.pdf;

namespace \_3sem

{

public class Class1

{

public static double poisk(int[] arr)

{

double suma = double.PositiveInfinity;

for (int i = 0; i < arr.Length - 1; i++)

{

if ((arr[i] + arr[i + 1]) % 2 == 1)

{

if (arr[i] + arr[i + 1] < suma)

{

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

}

}

}

return suma;

}

int[] mas;

public static double VivodDGV(ref int[] arr, double suma, DataGridView dgv, ref int[] mas)

{

int k = 0;

dgv.RowCount = 2;

dgv.ColumnCount = arr.GetLength(0);

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

{

if (arr[i] > suma)

{

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

dgv.Rows[1].Cells[k].Value = arr[i];

k = k + 1;

mas[k] = arr[i];

}

}

//Zapis\_v\_bloknot(arr, mas);

return 1;

}

public static void arrayGenerate(ref int[] arr, int length, int a, int b)

{

Random rand = new Random();

if (a < b)

{

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

{

arr[i] = rand.Next(a, b);

}

}

}

public static void output\_mas(ref DataGridView dgv, params int[] a)

{

dgv.RowCount = 2;

dgv.ColumnCount = a.GetLength(0);

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

{

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

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

}

}

public static void set\_mas(int k, ref int[] rezmas, ref int j, params int[] mas)

{

j = 0;

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

{

if (mas[i] > k) // if znac more than int

{

rezmas[i] = mas[i];

j++;

}

}

}

public static int inputInt(TextBox t)

{

if (Convert.ToString(t.Text) != "")

{

return Convert.ToInt32(t.Text);

}

else { return 0; }

}

public static void Vivod(TextBox t, double c)

{

t.Text = c.ToString("F");

}

public static void Sort\_prost(ref int[] arr)

{

int x, j;

for (int i = 2; i < arr.GetLength(0); i++)

{

x = arr[i];

arr[0] = x;

j = i - 1;

while (x < arr[j])

{

arr[j + 1] = arr[j];

j = j - 1;

}

arr[j + 1] = x;

}

}

public static void Sort\_obmen(ref int[] arr)

{

int x;

int n = arr.GetLength(0);

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

{

for (int j = n; j < i; j = j - 1)

{

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

{

x = arr[j - 1];

arr[j - 1] = arr[j];

arr[j] = x;

}

}

}

}

public static double Zad\_3(int[] arr)

{

double suma = -1;

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

{

if ((arr[i] / 100 >= 1) && (arr[i] % 10 == 9) && (arr[i] % 100 != 99))

{

if (suma == -1)

{

suma = 0;

}

suma = suma + arr[i];

}

}

return suma;

}

public static void Prof(TextBox textBox1, KeyPressEventArgs e)

{

int x1;

char ds = (char)NumberFormatInfo.CurrentInfo.NumberDecimalSeparator[0];

// Если нажатая клавиша не цифровая.

if (!Char.IsDigit(e.KeyChar))

// Запрет на ввод более одного знака минуса.

if (e.KeyChar != '-' || textBox1.Text.IndexOf('-') != -1)

// Запрет на ввод более одного десятичного разделителя.

if (e.KeyChar != ds || textBox1.Text.IndexOf(ds) != -1)

// Если нажатая клавиша не является клавишей BackSpace.

if (e.KeyChar != (char)Keys.Back)

e.Handled = true; // Запрет ввода

// Запрет ввода повторных нулей в начале числа.

if (e.KeyChar == (char)Keys.D0 || e.KeyChar == (char)Keys.NumPad0)

if (textBox1.Text.Length >= 1) // Если строка не пустая

if (textBox1.Text[0] == '0' && textBox1.SelectionStart < 2)

e.Handled = true; // Запрет ввода

//-незначащий ноль

if (Char.IsDigit(e.KeyChar))

if (textBox1.Text.Length == 1 && textBox1.Text[0] == '0')

textBox1.Paste(",");

// Замена десятичного разделителя в начале числа на "0,".

if (e.KeyChar == ds)

{ // Если введен десятичный разделитель

x1 = 0;

if (textBox1.Text != "" && textBox1.Text[0] == '-') { x1 = 1; }

if (textBox1.Text.IndexOf(ds) == -1 && textBox1.SelectionStart == x1)

{

// Если десятичный разделитель уже есть, то 0 не вставлять.

textBox1.Text = textBox1.Text.Insert(x1, "0"); // Вставка 0

textBox1.SelectionStart = x1 + 1; // Текстовый курсор в конец

}

// Перенос точки ввода минуса в начало (ввод минуса только спереди)

}

if (e.KeyChar == '-' && textBox1.Text.IndexOf('-') == -1)

textBox1.SelectionStart = 0; // Текстовый курсор в начало

//-незначащий ноль

}

public static void Zapis\_v\_PDF(int length, int[] arr, int[] mas)

{

var Document = new Document();

var Zap = PdfWriter.GetInstance(Document, new FileStream(@"C:\Users\Bulat\Desktop\Massive.pdf", FileMode.Create));

Document.Open();

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

var ft = new Font(Shrift, 10, Font.NORMAL, Color.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", ft);

tabl.AddCell(cell);

cell.BackgroundColor = Color.LIGHT\_GRAY;

cell.Colspan = 1;

cell.Border = 15;

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

{

//MessageBox.Show(mas[i].ToString());

if (arr[i] != 0)

{

cell.Phrase = new Phrase(arr[i].ToString(), ft);

tabl.AddCell(cell);

}

}

cell.HorizontalAlignment = Element.ALIGN\_LEFT;

cell.Colspan = 2;

cell.Border = 0;

cell.FixedHeight = 16.0F;

cell.BackgroundColor = Color.WHITE;

cell.Phrase = new Phrase("", ft);

tabl.AddCell(cell);

cell.Phrase = new Phrase("Реузльтирующий массив", ft);

tabl.AddCell(cell);

cell.BackgroundColor = Color.LIGHT\_GRAY;

cell.Colspan = 1;

cell.Border = 15;

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

{

if (mas[i] != 0)

{

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

tabl.AddCell(cell);

}

}

tabl.TotalWidth = Document.PageSize.Width - 400;

tabl.WriteSelectedRows(0, -1, 40, Document.PageSize.Height - 30, Zap.DirectContent);

Document.Close();

Zap.Close();

System.Diagnostics.Process.Start(@"C:\Users\Bulat\Desktop\Massive.pdf");

}

public static void Zapis\_v\_bloknot(ref int[] mas,ref int[] rezmas)

{

StreamWriter rez = File.CreateText("Lr1\_mas.txt");

rez.WriteLine("Исходный массив");

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

{

rez.WriteLine(mas[i]);

}

rez.WriteLine("");

rez.WriteLine("Результирующий массив");

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

{

if (rezmas[i] != 0)

{

rez.WriteLine(rezmas[i]);

}

}

rez.Close();

System.Diagnostics.Process.Start("Lr1\_mas.txt");

}

}

}

1. **Результат выполнения проекта**

Результат работы программы представлен на рисунке 15-17.

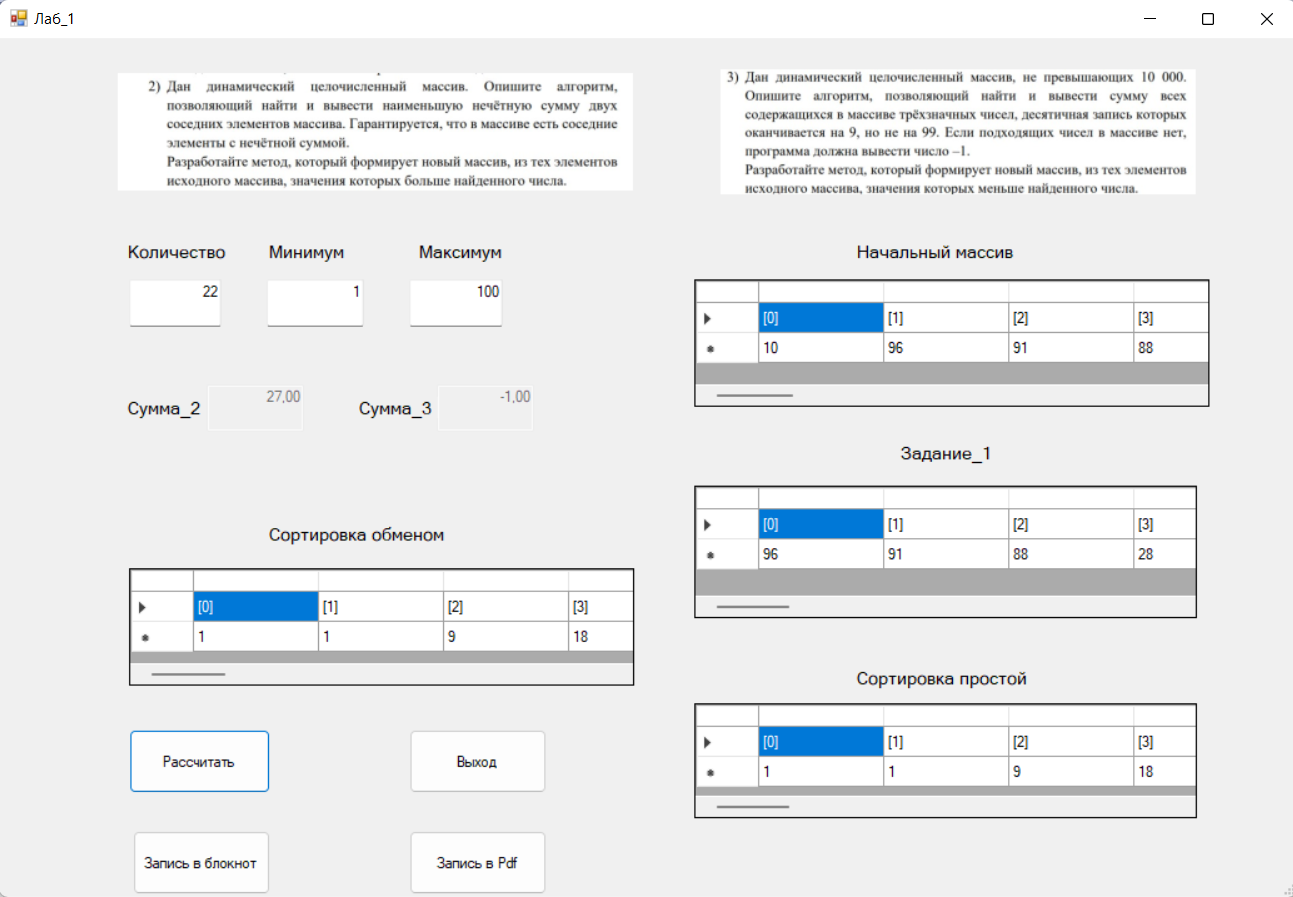


Рисунок 15 – Главная панель проекта

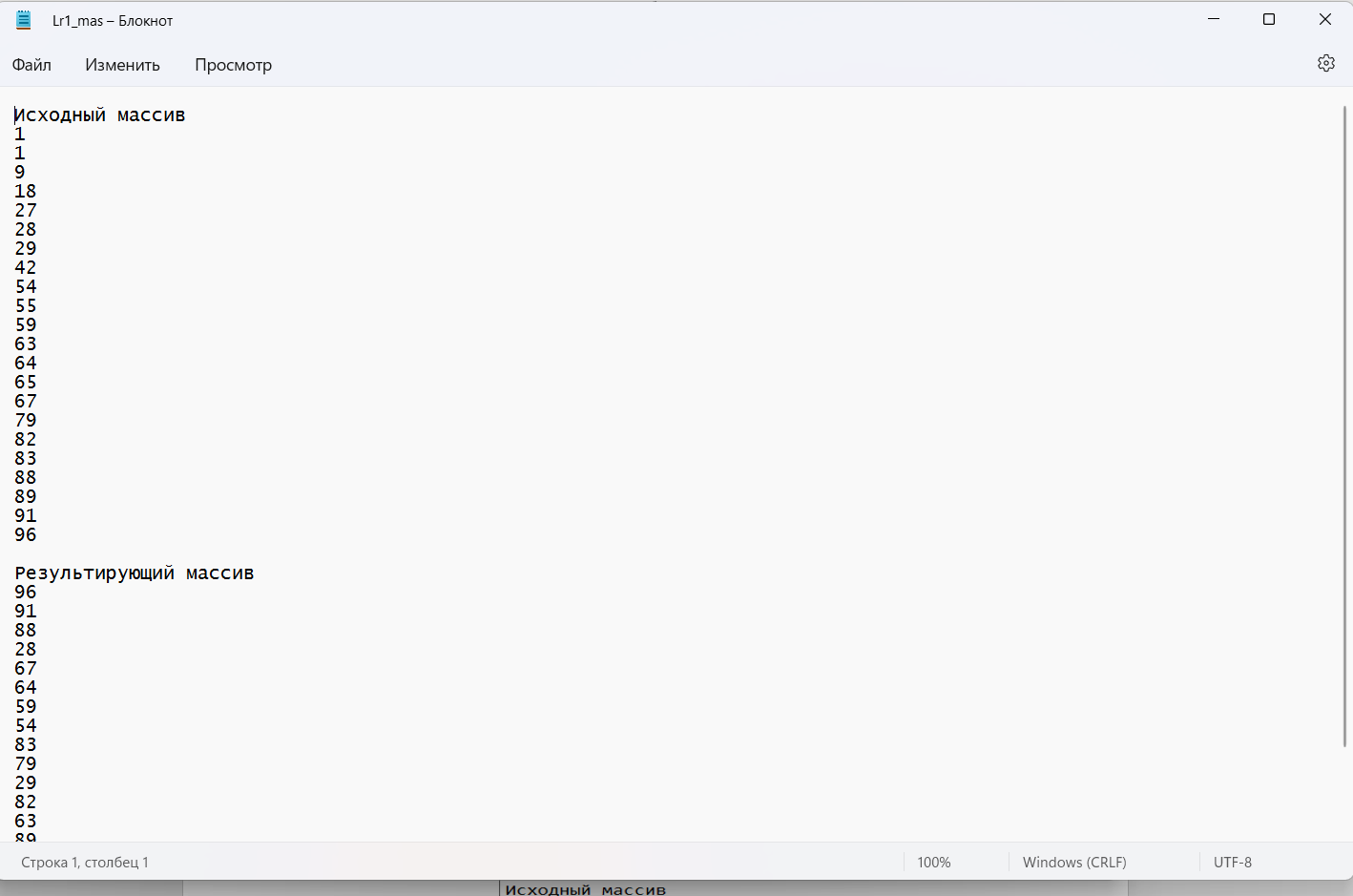


Рисунок 16 - Вывод массивов в PDF-документ

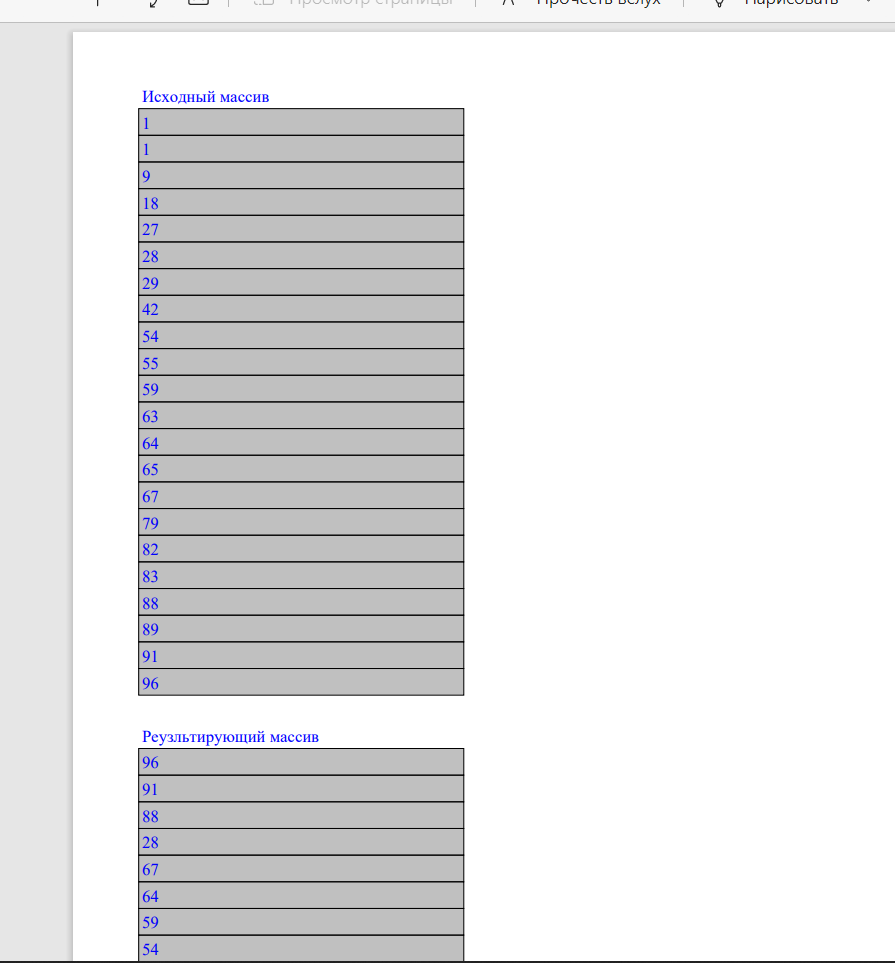


Рисунок 17 - Вывод массивов в редактор Блокнот

Таблица 1 – Работа программы при пограничных значениях

|  |  |  |
| --- | --- | --- |
| Причина | Действие | Примечание |
| A > B | Вывод ошибки через Messagebox | В ошибке написано: «Ошибка, верхняя граница не должна превышать нижнюю» |
| Length = 0 | Вывод ошибки через Messagebox | В ошибке написано: «Ошибка, длина должна быть больше нуля» |