****

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА  
(САМАРСКИЙ УНИВЕРСИТЕТ)»**

Институт (факультет) Информатики и кибернетики

Кафедра Программных систем

**ОТЧЁТ**

**по лабораторной работе**

№8 «Windows-приложения»  
по дисциплине «Основы программирования»

Выполнил Мананников М.А. 6103- 020302D

Проверил Котенева Светлана Эдуардовна

Самара  
2022

ЗАДАНИЕ

Задание 1

Выполнить лабораторную работу 7 как Windows-приложение.

Разработать графический интерфейс пользователя, адекватно отражающий функциональность приложения.

Желательно, чтобы приложение было реализовано в рамках архитектуры «Модель-Вид-Контроллер».

Задание 2

Подготовить отчет о работе.

**КОДПРОГРАММЫ**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Windows.Forms;

namespace WindowsFormsEdit\_1

{

static class Program

{

/// <summary>

/// Главная точка входа для приложения.

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Form1());

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace WindowsFormsEdit\_1

{

class Vectors

{

public static ArrayVector Sum(IVectorable vec1, IVectorable vec2)

{

if (vec1.Length == vec2.Length)

{

ArrayVector result = new ArrayVector();

int sum;

Console.Write("Сумма векторов: ");

for (int i = 0; i < vec1.Length; i++)

{

sum = vec1[i] + vec2[i];

result[i] = sum;

Console.Write(result[i] + " ");

}

return result;

}

else

{

throw new FormatException("Количество координат должно быть одинаково!");

}

}

public static double Scalar(IVectorable vec1, IVectorable vec2)

{

double result = 0;

if (vec1.Length == vec2.Length)

{

Console.Write("Скалярное произведние: ");

for (int i = 0; i < vec1.Length; i++)

{

result += vec1[i] \* vec2[i];

}

}

else

{

throw new FormatException("Количество координат должно быть одинаково!");

}

return result;

}

public static double GetNorm(IVectorable vec)

{

double len = 0;

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

{

len += Math.Pow(vec[i], 2);

}

len = Math.Sqrt(len);

return len;

}

public static LinkedListVector Sum\_1(IVectorable vec1, IVectorable vec2)

{

if (vec1.Length == vec2.Length)

{

LinkedListVector result = new LinkedListVector();

int sum;

Console.Write("Сумма векторов: ");

for (int i = 0; i < vec1.Length; i++)

{

sum = vec1[i] + vec2[i];

result[i] = sum;

Console.Write(result[i] + " ");

}

return result;

}

else

{

throw new FormatException("Количество координат должно быть одинаково!");

}

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace WindowsFormsEdit\_1

{

class LinkedListVector : IVectorable

{

public Node Head;

public class Node

{

public Node link; // null

public int value; // целое число

public Node() // констуктор по умолчанию

{

value = 0;

link = null;

}

public Node Link // свойство

{

get { return link; }

set { link = value; }

}

}

public int Length // свойство для чтения элентов

{

get

{

int i = 0;

Node node = Head;

while (node != null)

{

node = node.link;

i++;

}

return i;

}

}

public LinkedListVector(int len) // конструктор с парамаметром

{

try

{

if (len != 0)

{

Head = new Node();

Node node = Head;

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

{

node.Link = new Node();

node = node.Link;

}

}

}

catch (Exception)

{

Console.WriteLine("Длина не может равняться 0!");

}

}

public LinkedListVector() : this(5) { }

public Node GetElement(int num)

{

Node node = Head;

if (num > -1 && num < Length)

{

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

{

node = node.Link; //добавление ссылки

}

}

else

{

throw new IndexOutOfRangeException(); // принудительный выброс объекта исключения

}

return node;

}

public double GetNorm() // длина вектора (модуль)

{

double result = 0;

Node node = Head;

while (node != null)

{

result += Math.Pow(node.value, 2);

node = node.Link;

}

result = Math.Sqrt(result);

return result;

}

public int this[int num] // индексатор

{

get { return GetElement(num).value; }

set { GetElement(num).value = value; }

}

public override string ToString() // переопределение метода ToString класса Object

{

string str = Length + " ";

Node node = Head;

while (node != null)

{

str += node.value + " ";

node = node.Link;

}

return str;

}

public void AddElenentInBegin(int value)

{

Node newnode = new Node(); // создание нового объекта

newnode.value = value;

newnode.link = Head;

Head = newnode; // переписваивание главного элемента новым элементом

Node node = Head;

while (node != null)

{

Console.WriteLine(node.value);

node = node.link;

}

}

public void AddElementInEnd(int value)

{

Node newnode = new Node(); // создание нового объекта

newnode.value = value;

Node node = Head;

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

{

node = node.link; // добавление ссылки элементу

}

node.link = newnode; // присваивание ссылки новому элементу

node = Head;

while (node != null)

{

Console.WriteLine(node.value);

node = node.link;

}

}

public void DeleteElementInBegin()

{

Node node = Head;

node = node.link;

Head = node;

while (node != null)

{

Console.WriteLine(node.value);

node = node.link;

}

}

public void DeleteElementInEnd()

{

Node node = Head;

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

{

node = node.link;

}

node.link = null;

node = Head;

while (node != null)

{

Console.WriteLine(node.value);

node = node.link;

}

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace WindowsFormsEdit\_1

{

interface IVectorable

{

int this[int num] { get; set; } // индексатор

int Length { get; } // свойство для чтения

double GetNorm(); // получение модуля вектора

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace WindowsFormsEdit\_1

{

class ArrayVector : IVectorable

{

int[] arr;

public ArrayVector(int len) // конструктор с параметрами

{

arr = new int[len];

}

public ArrayVector() // конструктор без парам (по умолчанию 5)

{

arr = new int[5];

}

public void SetElement(int index, int digit) // метод установки элемента массива

{

arr[index] = digit;

}

public int Length

{

get

{

return arr.Length;

}

}

public int GetElement(int index)

{

return arr[index];

}

public int this[int index] // индексатор для организации доступа к элементам массива, выбрасывающий искл. при некорректном индексе

{

get

{

try

{

return GetElement(index);

}

catch (IndexOutOfRangeException)

{

Console.WriteLine("Ошибка! Выход за границу массива!");

return 0;

}

}

set

{

try

{

SetElement(index, value);

}

catch (IndexOutOfRangeException) // объект исключения

{

Console.WriteLine("Ошибка! Выход за границу массива!");

}

}

}

public double GetNorm() // модуль вектора

{

double modul = 0;

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

{

modul += Math.Pow(arr[i], 2);

}

modul = Math.Sqrt(modul);

return modul;

}

public override string ToString() // переопределение метода ToString()

{

string str = Length + " ";

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

{

str += arr[i] + " ";

}

return str;

}

}

}

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace WindowsFormsEdit\_1

{

public partial class FormLinkedListVector : Form

{

public FormLinkedListVector()

{

InitializeComponent();

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

}

private void textBox3\_TextChanged(object sender, EventArgs e)

{

}

private void textBox2\_TextChanged(object sender, EventArgs e)

{

}

private void textBox4\_TextChanged(object sender, EventArgs e)

{

}

private void textBox9\_TextChanged(object sender, EventArgs e)

{

}

private void textBox7\_TextChanged(object sender, EventArgs e)

{

}

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

{

int count = 0;

int length\_1 = 0;

try

{

length\_1 = Convert.ToInt32(textBox1.Text);

}

catch (Exception)

{

MessageBox.Show("Разменость вектора должна быть целым и положительным числом!");

ClearAll();

count++;

}

if (count == 0 && length\_1 >= 0)

{

LinkedListVector linkedlist1 = new LinkedListVector(length\_1);

string[] array\_1 = textBox3.Text.Split(' ');

try

{

for (int i = 0; i < length\_1; i++)

{

linkedlist1[i] = Convert.ToInt32(array\_1[i]);

}

}

catch (Exception)

{

MessageBox.Show("Некорректный ввод");

}

double modul1 = Vectors.GetNorm(linkedlist1);

textBox7.Text = modul1.ToString();

textBox9.Text = linkedlist1.ToString();

int k = Convert.ToInt32(textBox2.Text);

double k\_1 = linkedlist1[k - 1];

textBox4.Text = Convert.ToString(k\_1);

}

}

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

{

ClearAll();

}

public void ClearAll()

{

textBox1.Clear();

textBox2.Clear();

textBox3.Clear();

textBox4.Clear();

textBox7.Clear();

textBox9.Clear();

}

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

{

Close();

}

private void label1\_Click(object sender, EventArgs e)

{

}

}

}

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace WindowsFormsEdit\_1

{

public partial class FormArrayVector : Form

{

public FormArrayVector()

{

InitializeComponent();

}

private void label1\_Click(object sender, EventArgs e)

{

}

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

{

}

private void textBox9\_TextChanged(object sender, EventArgs e)

{

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

}

private void textBox3\_TextChanged(object sender, EventArgs e)

{

}

private void textBox2\_TextChanged(object sender, EventArgs e)

{

}

private void textBox4\_TextChanged(object sender, EventArgs e)

{

}

private void textBox7\_TextChanged(object sender, EventArgs e)

{

}

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

{

int count = 0;

int length\_1 = 0;

try

{

length\_1 = Convert.ToInt32(textBox1.Text);

}

catch (Exception)

{

MessageBox.Show("Разменость вектора должна быть целым и положительным числом!");

ClearAll();

count++;

}

if (count == 0 && length\_1 >= 0)

{

ArrayVector arrayVector1 = new ArrayVector(length\_1);

string[] array1 = textBox3.Text.Split(' ');

try

{

for (int i = 0; i < length\_1; i++)

{

arrayVector1[i] = Convert.ToInt32(array1[i]);

}

}

catch (Exception)

{

MessageBox.Show("Некорректный ввод");

}

double mod1 = Vectors.GetNorm(arrayVector1);

textBox7.Text = mod1.ToString();

textBox9.Text = arrayVector1.ToString();

int k = Convert.ToInt32(textBox2.Text);

double k\_1 = arrayVector1[k - 1];

textBox4.Text = Convert.ToString(k\_1);

}

}

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

{

ClearAll();

}

public void ClearAll()

{

textBox1.Clear();

textBox2.Clear();

textBox3.Clear();

textBox4.Clear();

textBox7.Clear();

textBox9.Clear();

}

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

{

Close();

}

}

}

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace WindowsFormsEdit\_1

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

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

{

FormArrayVector array = new FormArrayVector();

array.ShowDialog();

}

private void label3\_Click(object sender, EventArgs e)

{

}

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

{

FormLinkedListVector list = new FormLinkedListVector();

list.ShowDialog();

}

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

{

Close();

}

}

}

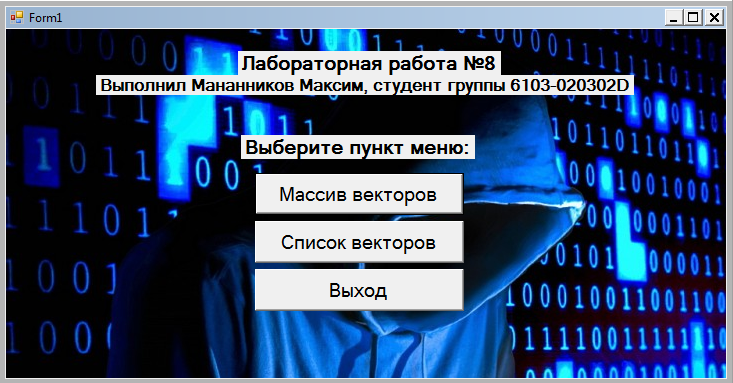


Рисунок 1 - Начало работы программы

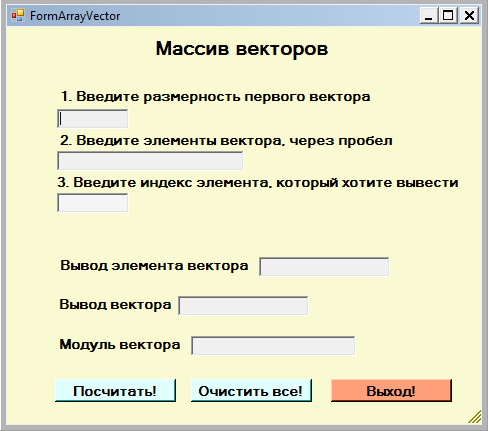


Рисунок 2 - Пример выполнения задания 1

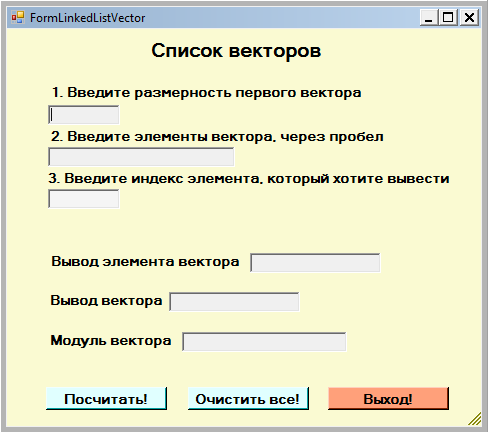


Рисунок 3 - Пример выполнения задания 2

**ВЫВОДЫ**

В лабораторной работе были использованы конструкции языка:

* условия;
* циклы;
* классы;
* конструкторы классов;
* различные методы классов;
* оператор switch;
* делегаты;
* свойства;
* архитектура «Модель-Вид-Контроллер»;
* оператор try-catch.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Павловская Т.А. C#. Программирование на языке высокого уровня. Учебник для вузов [Текст]/Т.А. Павловская. – СПб.: Питер, 2007. – 432 с.
2. Шилдт, Герберт. С# 4.0 : полное руководство / Герберт Шилдт ; [пер. с англ. и ред. И. В. Берштейна]. – Москва [и др.] : Вильямс, 2015. - 1056 с.