ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Факультет Информационных технологий
Кафедра Информатики и информационных технологий

направление подготовки
09.03.02 «Информационные системы и технологии»

ЛАБОРАТОРНАЯ РАБОТА № 2

Дисциплина: «Распознавание образов в информационных и автоматизированных системах копия 1»

Тема: «Работа с цветами»

Выполнил: студент группы 211-723

Сергеев Станислав Олегович

Дата, подпись_____ _____
(Дата) (Подпись)

Проверил:_____ _____
(Фамилия И.О., степень, звание) **(Оценка)**

Дата, подпись_____ _____
(Дата) (Подпись)

Замечания: _____

_____

_____

Москва

2022

Работа с цветами.

Цель:

Целью данной работы является изучение базовых операций над цветовыми каналами изображений и реализация некоторых фильтров на их основе.

Постановка задачи:

Необходимо разработать приложение Windows Forms, способное осуществлять:

1. загрузку и отображение двух изображений по выбору пользователя;
2. возможность применения базовых операций к загруженным изображениям;
3. возможность применения оконных и комбинированных фильтров к загруженным изображениям.

Листинг программы

```
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 Emgu.CV;
using Emgu.CV.CvEnum;
using Emgu.CV.Structure;
using Emgu.CV.Util;

namespace _2
{
    public partial class Form1 : Form
    {
        private Image<Bgr, byte> sourceImage;
        private Image<Bgr, byte> sourceImage2;//глобальная переменная
        private Image<Bgr, byte> sourceImage3;
        bool isImage2Loaded = false;
        bool isImage1Loaded = false;
        double brightness, contrast, k1, k2;
        byte hue, saturation, value;
        bool notFor9 = true;
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {

        }
```

```
private void statusMinusCnButton()
{
    if (contrast <= 0.1)
        minusContrast.Enabled = false;
    else
        plusContrast.Enabled = true;
}

private void statusPlusCnButton()
{
    if (contrast >= 4.99)
        plusContrast.Enabled = false;
    else
        minusContrast.Enabled = true;
}
private void statusMinusBrButton()
{
    if (brightness <= -255)
        minusBrightness.Enabled = false;
    else
        plusBrightness.Enabled = true;
}

private void statusPlusBrButton()
{
    if (brightness >= 255)
        plusBrightness.Enabled = false;
    else
        minusBrightness.Enabled = true;
}

private void ActiveDisabledButtons()
{
    blueButton.Enabled = true;
    greenButton.Enabled = true;
    redButton.Enabled = true;
    convertFor2.Enabled = true;
    convertFor3.Enabled = true;
    minusBrightness.Enabled = true;
    plusBrightness.Enabled = true;
    minusContrast.Enabled = true;
    plusContrast.Enabled = true;
    huePlus.Enabled = true;
    hueMinus.Enabled = true;
    saturationMinus.Enabled = true;
    saturationPlus.Enabled = true;
    valuePlus.Enabled = true;
    ValueMinus.Enabled = true;
    countOfHue.Text = "0";
    countOfSaturation.Text = "0";
    countOfValue.Text = "0";
    ConvertFor6.Enabled = true;
    convertFor7.Enabled = true;
    convertFor8.Enabled = true;
    matrix00.Enabled = true;
    matrix01.Enabled = true;
    matrix02.Enabled = true;
    matrix10.Enabled = true;
    matrix11.Enabled = true;
    matrix12.Enabled = true;
    matrix20.Enabled = true;
    matrix21.Enabled = true;
    matrix22.Enabled = true;
    convertFor9.Enabled = true;
    Matrix000.Enabled = true;
    Matrix001.Enabled = true;
    Matrix002.Enabled = true;
```

```
    Matrix010.Enabled = true;
    Matrix011.Enabled = true;
    Matrix012.Enabled = true;
    Matrix020.Enabled = true;
    Matrix021.Enabled = true;
    Matrix022.Enabled = true;
    convertFor10.Enabled = true;
}

private void openMenu10()
{
    menu10.Location = new Point(564, 122);
    menu10.Visible = true;
    Back.Visible = true;
    convertFor10.Visible = true;
}
private void closeMenu10()
{
    menu10.Location = new Point(564, 563);
    convertFor10.Visible = false;
}

private void backToMenu()
{
    openMenu1();
    if (menu1.Visible == true)
        closeMenu1();
    if (menu2.Visible == true)
        closeMenu2();
    if (menu3.Visible == true)
        closeMenu3();
    if (menu4.Visible == true)
        closeMenu4();
    if (menu5.Visible == true)
        closeMenu5();
    if (menu6.Visible == true)
        closeMenu6();
    if (menu7.Visible == true)
        closeMenu7();
    if (menu8.Visible == true)
        closeMenu8();
    if (menu9.Visible == true)
        closeMenu9();
    if (menu10.Visible == true)
        closeMenu10();
    menu.Visible = true;
    menu1.Visible = true;
    menu2.Visible = true;
    menu3.Visible = true;
    menu4.Visible = true;
    menu5.Visible = true;
    menu6.Visible = true;
    menu7.Visible = true;
    menu8.Visible = true;
    menu9.Visible = true;
    menu10.Visible = true;
    Back.Visible = false;
}

private void goToMethod(Button but)
{
    menu.Visible = false;
    menu1.Visible = false;
    menu2.Visible = false;
    menu3.Visible = false;
    menu4.Visible = false;
    menu5.Visible = false;
```

```
    menu6.Visible = false;
    menu7.Visible = false;
    menu8.Visible = false;
    menu9.Visible = false;
    menu10.Visible = false;
    Back.Visible = true;
    Back.Visible = true;
    if (but == menu1)
        openMenu1();
    if (but == menu2)
        openMenu2();
    if (but == menu3)
        openMenu3();
    if (but == menu4)
        openMenu4();
    if (but == menu5)
        openMenu5();
    if (but == menu6)
        openMenu6();
    if (but == menu7)
        openMenu7();
    if (but == menu8)
        openMenu8();
    if (but == menu9)
        openMenu9();
    if (but == menu10)
        openMenu10();
}

double Make_Color(double Color)
{

    if (Color > 255)
    {
        return 255;
    }
    else if (Color < 0)
    {
        return 0;
    }
    else
    {
        return Color;
    }
}
    private void openMenu1()
{
    menu1.Location = new Point(564, 122);
    menu1.Visible = true;
    Back.Visible = true;
    ChooseTheColorTextBox.Visible = true;
    blueButton.Visible = true;
    greenButton.Visible = true;
    redButton.Visible = true;
}
private void closeMenu1()
{
    ChooseTheColorTextBox.Visible = false;
    blueButton.Visible = false;
    greenButton.Visible = false;
    redButton.Visible = false;
}
private void openMenu2()
{
    menu2.Location = new Point(564, 122);
    menu2.Visible = true;
    Back.Visible = true;
```

```
    convertFor2.Visible = true;
}
private void closeMenu2()
{
    menu2.Location = new Point(564, 171);
    convertFor2.Visible = false;
}
private void openMenu3()
{
    menu3.Location = new Point(564, 122);
    menu3.Visible = true;
    Back.Visible = true;
    convertFor3.Visible = true;
}
private void closeMenu3()
{
    menu3.Location = new Point(564, 220);
    convertFor3.Visible = false;
}
private void openMenu4()
{
    menu4.Location = new Point(564, 122);
    menu4.Visible = true;
    Back.Visible = true;
    ChangeBrightness.Visible = true;
    changeContrast.Visible = true;
    countOfBrightness.Visible = true;
    countOfContrast.Visible = true;
    plusBrightness.Visible = true;
    minusBrightness.Visible = true;
    plusContrast.Visible = true;
    minusContrast.Visible = true;
}
private void closeMenu4()
{
    menu4.Location = new Point(564, 269);
    ChangeBrightness.Visible = false;
    changeContrast.Visible = false;
    countOfBrightness.Visible = false;
    countOfContrast.Visible = false;
    plusBrightness.Visible = false;
    minusBrightness.Visible = false;
    plusContrast.Visible = false;
    minusContrast.Visible = false;
}

private void openMenu5()
{
    menu5.Size = new Size(238, 35);
    menu5.Location = new Point(597, 1);
    Back.Location = new Point(845, 7);
    loadImage.Location = new Point(350, 2);
    imageBox3.Location = new Point(513, 42);
    menu5.Visible = true;
    Back.Visible = true;
    loadimage2.Visible = true;
    imageBox3.Visible = true;
    chooseTheOperation.Visible = true;
    dopolnenie.Visible = true;
    iskluchenie.Visible = true;
    peresechenie.Visible = true;
    countOfK1.Visible = true;
    countOfK2.Visible = true;
    k1Minus.Visible = true;
    k2Minus.Visible = true;
    k1Plus.Visible = true;
    k2Plus.Visible = true;
```

```
}

private void closeMenu5()
{
    menu5.Location = new Point(564, 318);
    menu5.Size = new Size(285, 43);
    loadImage.Location = new Point(661, 42);
    Back.Location = new Point(661, 87);
    loadimage2.Visible = false;
    imageBox3.Visible = false;
    chooseTheOperation.Visible = false;
    dopolnenie.Visible = false;
    iskluchenie.Visible = false;
    peresechenie.Visible = false;
    countOfK1.Visible = false;
    countOfK2.Visible = false;
    k1Minus.Visible = false;
    k2Minus.Visible = false;
    k1Plus.Visible = false;
    k2Plus.Visible = false;
}

private void openMenu6()
{
    menu6.Location = new Point(564, 122);
    menu6.Visible = true;
    Back.Visible = true;
    setHsvBytes.Visible = true;
    countOfValue.Visible = true;
    countOfHue.Visible = true;
    countOfSaturation.Visible = true;
    hueMinus.Visible = true;
    huePlus.Visible = true;
    saturationMinus.Visible = true;
    saturationPlus.Visible = true;
    valuePlus.Visible = true;
    ValueMinus.Visible = true;
    ConvertFor6.Visible = true;
    changeHue.Visible = true;
    changeSaturation.Visible = true;
    changeValue.Visible = true;
}

private void closeMenu6()
{
    menu6.Location = new Point(564, 367);
    setHsvBytes.Visible = false;
    countOfValue.Visible = false;
    countOfHue.Visible = false;
    countOfSaturation.Visible = false;
    hueMinus.Visible = false;
    huePlus.Visible = false;
    saturationMinus.Visible = false;
    saturationPlus.Visible = false;
    valuePlus.Visible = false;
    ValueMinus.Visible = false;
    ConvertFor6.Visible = false;
    changeHue.Visible = false;
    changeSaturation.Visible = false;
    changeValue.Visible = false;
}

private void openMenu7()
{
    menu7.Location = new Point(564, 122);
    menu7.Visible = true;
    Back.Visible = true;
```

```
    convertFor7.Visible = true;
}

private void closeMenu7()
{
    menu7.Location = new Point(564, 416);
    convertFor7.Visible = false;
}

private void openMenu8()
{
    menu8.Location = new Point(564, 122);
    menu8.Visible = true;
    Back.Visible = true;
    convertFor8.Visible = true;
    matrix00.Visible = true;
    matrix01.Visible = true;
    matrix02.Visible = true;
    matrix10.Visible = true;
    matrix11.Visible = true;
    matrix12.Visible = true;
    matrix20.Visible = true;
    matrix21.Visible = true;
    matrix22.Visible = true;

}

private void closeMenu8()
{
    menu8.Location = new Point(564, 465);
    convertFor8.Visible = false;
    matrix00.Visible = false;
    matrix01.Visible = false;
    matrix02.Visible = false;
    matrix10.Visible = false;
    matrix11.Visible = false;
    matrix12.Visible = false;
    matrix20.Visible = false;
    matrix21.Visible = false;
    matrix22.Visible = false;
}

private void openMenu9()
{
    menu9.Visible = true;
    menu9.Size = new Size(238, 35);
    menu9.Location = new Point(597, 1);
    Back.Location = new Point(845, 7);
    notFor9 = false;
    chooseTheOperation.Location = new Point(156, 498);
    dopolnenie.Location = new Point(42, 542);
    iskluchenie.Location = new Point(168, 542);
    peresechenie.Location = new Point(294, 542);
    loadImage.Location = new Point(350, 2);
    imageBox3.Location = new Point(513, 42);
    loadimage2.Visible = true;
    imageBox3.Visible = true;
    chooseTheOperation.Visible = true;
    dopolnenie.Visible = true;
    iskluchenie.Visible = true;
    peresechenie.Visible = true;
    countOfK1.Visible = true;
    countOfK2.Visible = true;
    k1Minus.Visible = true;
    k2Minus.Visible = true;
    k1Plus.Visible = true;
    k2Plus.Visible = true;
```

```
    ChangeBrightness.Location = new Point(564, 500);
    ChangeBrightness.Visible = true;
    changeContrast.Location = new Point(743, 500);
    changeContrast.Visible = true;
    countOfBrightness.Location = new Point(597, 525);
    countOfBrightness.Visible = true;
    countOfContrast.Location = new Point(774, 525);
    countOfContrast.Visible = true;
    plusBrightness.Location = new Point(643, 525);
    plusBrightness.Visible = true;
    minusBrightness.Location = new Point(564, 525);
    minusBrightness.Visible = true;
    plusContrast.Location = new Point(823, 525);
    plusContrast.Visible = true;
    minusContrast.Location = new Point(742, 525);
    minusContrast.Visible = true;
    convertFor9.Visible = true;
    Matrix000.Visible = true;
    Matrix001.Visible = true;
    Matrix002.Visible = true;
    Matrix010.Visible = true;
    Matrix011.Visible = true;
    Matrix012.Visible = true;
    Matrix020.Visible = true;
    Matrix021.Visible = true;
    Matrix022.Visible = true;

}

private void closeMenu9()
{
    loadimage2.Visible = false;
    imageBox3.Visible = false;
    chooseTheOperation.Visible = false;
    dopolnenie.Visible = false;
    iskluchenie.Visible = false;
    peresechenie.Visible = false;
    countOfK1.Visible = false;
    countOfK2.Visible = false;
    k1Minus.Visible = false;
    k2Minus.Visible = false;
    k1Plus.Visible = false;
    k2Plus.Visible = false;
    menu9.Location = new Point(564, 514);
    menu9.Size = new Size(285, 43);
    notFor9 = true;
    chooseTheOperation.Location = new Point(420, 499);
    dopolnenie.Location = new Point(282, 542);
    iskluchenie.Location = new Point(430, 542);
    peresechenie.Location = new Point(576, 542);
    loadImage.Location = new Point(661, 42);
    Back.Location = new Point(661, 87);
    ChangeBrightness.Location = new Point(564, 181);
    ChangeBrightness.Visible = false;
    changeContrast.Location = new Point(743, 181);
    changeContrast.Visible = false;
    countOfBrightness.Location = new Point(597, 207);
    countOfBrightness.Visible = false;
    countOfContrast.Location = new Point(774, 207);
    countOfContrast.Visible = false;
    plusBrightness.Location = new Point(643, 207);
    plusBrightness.Visible = false;
    minusBrightness.Location = new Point(564, 207);
    minusBrightness.Visible = false;
    plusContrast.Location = new Point(822, 207);
    plusContrast.Visible = false;
    minusContrast.Location = new Point(743, 207);
```

```
    minusContrast.Visible = false;
    convertFor9.Visible = false;
    Matrix000.Visible = false;
    Matrix001.Visible = false;
    Matrix002.Visible = false;
    Matrix010.Visible = false;
    Matrix011.Visible = false;
    Matrix012.Visible = false;
    Matrix020.Visible = false;
    Matrix021.Visible = false;
    Matrix022.Visible = false;
}

private void button1_Click(object sender, EventArgs e)
{
     OpenFileDialog openFileDialog = new OpenFileDialog();
    var result = openFileDialog.ShowDialog(); // открытие диалога выбора файла
    if (result == DialogResult.OK) // открытие выбранного файла
    {
        string fileName = openFileDialog.FileName;
        sourceImage = new Image<Bgr, byte>(fileName);
        imageBox1.Image = sourceImage.Resize(400, 400, Inter.Linear);
        imageBox2.Image = sourceImage.Resize(400, 400, Inter.Linear);
        isImage1Loaded = true;
        ActiveDisabledButtons();
        contrast = 1.0;
        brightness = 0.0;
        k1 = 0.5;
        k2 = 0.5;
        hue = 0;
        value = 0;
        saturation = 0;
        countOfBrightness.Text = brightness.ToString();
        countOfContrast.Text = contrast.ToString();
    }
    if (isImage1Loaded && isImage2Loaded)
    {
        iskluchenie.Enabled = true;
        dopolnenie.Enabled = true;
        peresechenie.Enabled = true;
        k1Minus.Enabled = true;
        k2Minus.Enabled = true;
        k1Plus.Enabled = true;
        k2Plus.Enabled = true;
        countOfK1.Text = k1.ToString();
        countOfK2.Text = k2.ToString();
    }
}

private void blueButton_Click(object sender, EventArgs e)
{
    var destImage = new Image<Bgr, byte>(sourceImage.Size);
    var channel = sourceImage.Split()[0];
    VectorOfMat vm = new VectorOfMat();
    vm.Push(channel[0]);
    CvInvoke.Merge(vm, destImage);
    imageBox2.Image = destImage.Resize(400, 400, Inter.Linear);
}

private void greenButton_Click(object sender, EventArgs e)
{

    var destImage = new Image<Bgr, byte>(sourceImage.Size);
    var channel = sourceImage.Split()[1];
    VectorOfMat vm = new VectorOfMat();
    vm.Push(channel[0]);
    CvInvoke.Merge(vm, destImage);
```

```
            imageBox2.Image = destImage.Resize(400, 400, Inter.Linear);
        }

        private void redButton_Click(object sender, EventArgs e)
        {
            var destImage = new Image<Bgr, byte>(sourceImage.Size);
            var channel = sourceImage.Split()[2];
            VectorOfMat vm = new VectorOfMat();
            vm.Push(channel[0]);
            CvInvoke.Merge(vm, destImage);
            imageBox2.Image = destImage.Resize(400, 400, Inter.Linear);
        }

        private void menu_Click(object sender, EventArgs e)
        {

        }

        private void menu1_Click(object sender, EventArgs e)
        {
            goToMethod(menu1);
        }

        private void Back_Click(object sender, EventArgs e)
        {
            backToMenu();
        }

        private void menu2_Click(object sender, EventArgs e)
        {
            goToMethod(menu2);
        }

        private void convertFor2_Click(object sender, EventArgs e)
        {
            var grayImage = new Image<Gray, byte>(sourceImage.Size);
                for (int x = 0; x < grayImage.Width; x++)
                    for (int y = 0; y < grayImage.Height; y++) // обход по пискелям
                        grayImage.Data[y, x, 0] = Convert.ToByte(0.299 * sourceImage.Data[y, x, 2] + 0.587 * sourceImage.Data[y,
x, 1] + 0.114 * sourceImage.Data[y, x, 0]);
            imageBox2.Image = grayImage.Resize(400, 400, Inter.Linear);
        }

        private void menu3_Click(object sender, EventArgs e)
        {
            goToMethod(menu3);
        }

        private void convertFor3_Click(object sender, EventArgs e)
        {
            var resultImage = new Image<Bgr, byte>(sourceImage.Size);
            bool isFailed = true;
            double k1 = 0.393; double k2 = 0.769; double k3 = 0.189; double k4 = 0.349; double k5 = 0.686; double k6 = 0.168;
double k7 = 0.272; double k8 = 0.534; double k9 = 0.131; double k10 = 0.99;
            while (isFailed)
            {
                try
                {
                    for (int x = 0; x < resultImage.Width; x++)
                        for (int y = 0; y < resultImage.Height; y++) // обход по пискелям
                        {
                            resultImage.Data[y, x, 2] = Convert.ToByte(k1 * sourceImage.Data[y, x, 2] + k2 * sourceImage.Data[y, x,
1] + k3 * sourceImage.Data[y, x, 0]);
                            resultImage.Data[y, x, 1] = Convert.ToByte(k4 * sourceImage.Data[y, x, 2] + k5 * sourceImage.Data[y, x,
1] + k6 * sourceImage.Data[y, x, 0]);
                            resultImage.Data[y, x, 0] = Convert.ToByte(k7 * sourceImage.Data[y, x, 2] + k8 * sourceImage.Data[y, x,
1] + k9 * sourceImage.Data[y, x, 0]);
```

```
                }
                imageBox2.Image = resultImage.Resize(400, 400, Inter.Linear);
                isFailed = false;
            }
            catch (OverflowException)
            {
                k1 *= k10; k2 *= k10; k3 *= k10; k4 *= k10; k5 *= k10; k6 *= k10; k7 *= k10; k8 *= k10; k9 *= k10;
                isFailed = true;
            }
        }
    }

    private void menu4_Click(object sender, EventArgs e)
    {
        goToMethod(menu4);
    }

    private double AddColors(double color1, double color2)
    {
        if (color1 + color2 > 255) return 255;
        else if (color1 + color2 < 0) return 0;
        else return color1 + color2;
    }

    private double MultiplyColors(double color1, double color2)
    {
        if (color1 * color2 > 255) return 255;
        else if (color1 * color2 < 0) return 0;
        else return color1 * color2;

    }

    private void ImageWithBrightnessAndContrast()
    {
        if(notFor9)
        {
            var resultImage = new Image<Bgr, byte>(sourceImage.Size);
            for (int channel = 0; channel < resultImage.NumberOfChannels; channel++) //обход по каналам
                for (int x = 0; x < resultImage.Width; x++)
                    for (int y = 0; y < resultImage.Height; y++) // обход по пискелям
                        resultImage.Data[y, x, channel] = Convert.ToByte(MultiplyColors(AddColors(sourceImage.Data[y, x,
channel], brightness), contrast));
            imageBox2.Image = resultImage.Resize(400, 400, Inter.Linear);
        }
        else
        {
            var resultImage = new Image<Bgr, byte>(sourceImage3.Size);
            for (int channel = 0; channel < resultImage.NumberOfChannels; channel++) //обход по каналам
                for (int x = 0; x < resultImage.Width; x++)
                    for (int y = 0; y < resultImage.Height; y++) // обход по пискелям
                        resultImage.Data[y, x, channel] = Convert.ToByte(MultiplyColors(AddColors(sourceImage3.Data[y, x,
channel], brightness), contrast));
            imageBox2.Image = resultImage.Resize(400, 400, Inter.Linear);
        }
    }


    private void plusBrightness_Click(object sender, EventArgs e)
    {
        brightness += 5.0;
        countOfBrightness.Text = brightness.ToString();
        ImageWithBrightnessAndContrast();
        statusPlusBrButton();
    }

    private void minusBrightness_Click(object sender, EventArgs e)
    {
```

```
    brightness -= 5.0;
    countOfBrightness.Text = brightness.ToString();
    ImageWithBrightnessAndContrast();
    statusMinusBrButton();
}

private void k1Minus_Click(object sender, EventArgs e)
{
    k2Plus_Click(sender, e);
}

private void Isk1Full()
{
    if (k1 == 1)
    {
        k1Plus.Enabled = false;
        k2Minus.Enabled = false;
    }
    else
    {
        k2Plus.Enabled = true;
        k1Minus.Enabled = true;
        k2Minus.Enabled = true;
        k1Plus.Enabled = true;
    }
}

private void Isk2Full()
{
    if (k2 == 1)
    {
        k2Plus.Enabled = false;
        k1Minus.Enabled = false;
    }
    else
    {
        k2Plus.Enabled = true;
        k1Minus.Enabled = true;
        k2Minus.Enabled = true;
        k1Plus.Enabled = true;
    }
}


private void k1Plus_Click(object sender, EventArgs e)
{
    k2Minus_Click(sender, e);
}

private void k2Minus_Click(object sender, EventArgs e)
{
    k1 += 0.1;
    k2 -= 0.1;
    if (k2 > 0.1)
    {
        countOfK1.Text = k1.ToString();
        countOfK2.Text = k2.ToString();
    }
    else
    {
        k1 = 1;
        k2 = 0;
        countOfK1.Text = k1.ToString();
        countOfK2.Text = k2.ToString();
    }
    Isk1Full();
}
```

```
private void k2Plus_Click(object sender, EventArgs e)
{
    k1 -= 0.1;
    k2 += 0.1;
    if (k1 > 0.1)
    {
        countOfK1.Text = k1.ToString();
        countOfK2.Text = k2.ToString();
    }
    else
    {
        k2 = 1;
        k1 = 0;
        countOfK1.Text = k1.ToString();
        countOfK2.Text = k2.ToString();
    }
    Isk2Full();
}

private void loadimage2_Click(object sender, EventArgs e)
{
    OpenFileDialog openFileDialog = new OpenFileDialog();
    var result = openFileDialog.ShowDialog(); // открытие диалога выбора файла
    if (result == DialogResult.OK) // открытие выбранного файла
    {
        string fileName = openFileDialog.FileName;
        sourceImage2 = new Image<Bgr, byte>(fileName);
        imageBox3.Image = sourceImage2.Resize(400, 400, Inter.Linear);
        sourceImage3 = sourceImage2.Resize(400, 400, Inter.Linear);
        isImage2Loaded = true;
        contrast = 1.0;
        brightness = 0.0;
        k1 = 0.5;
        k2 = 0.5;
        countOfBrightness.Text = brightness.ToString();
        countOfContrast.Text = contrast.ToString();
    }
    if (isImage1Loaded && isImage2Loaded)
    {
        countOfK1.Text = k1.ToString();
        countOfK2.Text = k2.ToString();
        iskluchenie.Enabled = true;
        dopolnenie.Enabled = true;
        peresechenie.Enabled = true;
        k1Minus.Enabled = true;
        k2Minus.Enabled = true;
        k1Plus.Enabled = true;
        k2Plus.Enabled = true;
    }
}

private void dopolnenie_Click(object sender, EventArgs e)
{
    var si1 = sourceImage.Resize(400, 400, Inter.Linear);
    var si2 = sourceImage2.Resize(400, 400, Inter.Linear);
    for (int channel = 0; channel < si1.NumberOfChannels; channel++) //обход по каналам
        for (int x = 0; x < si1.Width; x++)
            for (int y = 0; y < si1.Height; y++) // обход по пискелям
                si1.Data[y, x, channel] = Convert.ToByte(AddColors(MultiplyColors((Convert.ToDouble(si1.Data[y, x,
channel])),k1),MultiplyColors((Convert.ToDouble(si2.Data[y, x, channel])),k2)));
    imageBox2.Image = si1.Resize(400, 400, Inter.Linear);
    sourceImage3 = si1;
}

private double SubtractionOfColors(double color1, double color2)
{
```

```
            if (color1 - color2 > 255) return 255;
            else if (color1 - color2 < 0) return 0;
            else return color1 - color2;
        }

        private void iskluchenie_Click(object sender, EventArgs e)
        {
            var si1 = sourceImage.Resize(400, 400, Inter.Linear);
            var si2 = sourceImage2.Resize(400, 400, Inter.Linear);
            for (int channel = 0; channel < si1.NumberOfChannels; channel++) //обход по каналам
                for (int x = 0; x < si1.Width; x++)
                    for (int y = 0; y < si1.Height; y++) // обход по пискелям
                        si1.Data[y, x, channel] =
Convert.ToByte(SubtractionOfColors(MultiplyColors((Convert.ToDouble(si1.Data[y, x, channel])), k1),
MultiplyColors((Convert.ToDouble(si2.Data[y, x, channel])), k2)));
            imageBox2.Image = si1.Resize(400, 400, Inter.Linear);
            sourceImage3 = si1;
        }

        private void peresechenie_Click(object sender, EventArgs e)
        {
            var si1 = sourceImage.Resize(400, 400, Inter.Linear);
            var si2 = sourceImage2.Resize(400, 400, Inter.Linear);
            for (int channel = 0; channel < si1.NumberOfChannels; channel++) //обход по каналам
                for (int x = 0; x < si1.Width; x++)
                    for (int y = 0; y < si1.Height; y++) // обход по пискелям
                        si1.Data[y, x, channel] = Convert.ToByte(Make_Color(si1.Data[y, x, channel] + si2.Data[y, x, channel]));
            imageBox2.Image = si1.Resize(400, 400, Inter.Linear);
            sourceImage3 = si1;
        }

        private void ConvertFor6_Click(object sender, EventArgs e)
        {
            var hsvImage = sourceImage.Convert<Hsv, byte>();
            for (int x = 0; x < hsvImage.Width; x++)
                for (int y = 0; y < hsvImage.Height; y++) // обход по пискелям
                {
                    hsvImage.Data[y, x, 0] = hue;
                    hsvImage.Data[y, x, 1] = saturation;
                    hsvImage.Data[y, x, 0] = value;
                }
            var result = hsvImage.Convert<Bgr, byte>();
            imageBox2.Image = result.Resize(400, 400, Inter.Linear);
        }

        private void huePlus_Click(object sender, EventArgs e)
        {
            if (hue == (byte)255)
                hue = (byte)0;
            else
                hue += (byte)5;
            countOfHue.Text = hue.ToString();
        }

        private void saturationPlus_Click(object sender, EventArgs e)
        {
            if (saturation == (byte)255)
                saturation = (byte)0;
            else
                saturation += (byte)5;
            countOfSaturation.Text = saturation.ToString();
        }

        private void valuePlus_Click(object sender, EventArgs e)
        {
            if (value == (byte)255)
                value = (byte)0;
```

```
        else
            value += (byte)5;
        countOfValue.Text = value.ToString();
    }

    private void ValueMinus_Click(object sender, EventArgs e)
    {
        if (value == (byte)0)
            value = (byte)255;
        else
            value -= (byte)5;
        countOfValue.Text = value.ToString();
    }

    private void saturationMinus_Click(object sender, EventArgs e)
    {
        if (saturation == (byte)0)
            saturation = (byte)255;
        else
            saturation -= (byte)5;
        countOfSaturation.Text = saturation.ToString();
    }

    private void hueMinus_Click(object sender, EventArgs e)
    {
        if (hue == (byte)0)
            hue = (byte)255;
        else
            hue -= (byte)5;
        countOfHue.Text = hue.ToString();
    }

    private void menu7_Click(object sender, EventArgs e)
    {
        goToMethod(menu7);
    }
    Image<Bgr, byte> Add_Image_NoCof(Image<Bgr, byte> Source_Image, Image<Bgr, byte> Add_Image)
    {

        Image<Bgr, byte> Result_Img = new Image<Bgr, byte>(Source_Image.Size);

        for (int x = 0; x < Source_Image.Size.Width; x++)
        {

            for (int y = 0; y < Source_Image.Size.Height; y++)
            {
                Result_Img.Data[y, x, 0] = Convert.ToByte(Make_Color(Source_Image.Data[y, x, 0] + Add_Image.Data[y, x,
0]));
                Result_Img.Data[y, x, 1] = Convert.ToByte(Make_Color(Source_Image.Data[y, x, 1] + Add_Image.Data[y, x,
1]));
                Result_Img.Data[y, x, 2] = Convert.ToByte(Make_Color(Source_Image.Data[y, x, 2] + Add_Image.Data[y, x,
2]));

            }

        }

        return Result_Img.Resize(400, 400, Inter.Linear);

    }
    private void convertFor7_Click(object sender, EventArgs e)
    {

        var resultImage = new Image<Bgr, byte>(sourceImage.Size);
        for (int channel = 0; channel < resultImage.NumberOfChannels; channel++) //обход по каналам
            for (int x = 0; x < resultImage.Width; x++)
                for (int y = 0; y < resultImage.Height; y++)
```

```
{
    List<int> window = new List<int>();
    window.Add((int)sourceImage.Data[y, x, channel]);

    if ((y == 0) && (x == 0))
    {
        window.Add((int)sourceImage.Data[y + 1, x, channel]);
        window.Add((int)sourceImage.Data[y, x + 1, channel]);
        window.Add((int)sourceImage.Data[y + 1, x + 1, channel]);
        window.Add((int)sourceImage.Data[y, x, channel]);
        window.Add((int)sourceImage.Data[y, x, channel]);
        window.Add((int)sourceImage.Data[y, x, channel]);
        window.Add((int)sourceImage.Data[y + 1, x, channel]);
        window.Add((int)sourceImage.Data[y, x + 1, channel]);
        window.Sort();
        resultImage.Data[y, x, channel] = (byte)window[5];
        continue;
    }
    if ((y == resultImage.Height-1) && (x == 0))
    {
        window.Add((int)sourceImage.Data[y, x, channel]);
        window.Add((int)sourceImage.Data[y, x + 1, channel]);
        window.Add((int)sourceImage.Data[y, x + 1, channel]);
        window.Add((int)sourceImage.Data[y - 1, x, channel]);
        window.Add((int)sourceImage.Data[y, x, channel]);
        window.Add((int)sourceImage.Data[y - 1, x, channel]);
        window.Add((int)sourceImage.Data[y, x , channel]);
        window.Add((int)sourceImage.Data[y - 1, x + 1, channel]);
        window.Sort();
        resultImage.Data[y, x, channel] = (byte)window[5];
        continue;
    }
    if ((y == 0) && (x == resultImage.Width-1))
    {
        window.Add((int)sourceImage.Data[y + 1, x, channel]);
        window.Add((int)sourceImage.Data[y, x, channel]);
        window.Add((int)sourceImage.Data[y + 1, x, channel]);
        window.Add((int)sourceImage.Data[y, x, channel]);
        window.Add((int)sourceImage.Data[y, x - 1, channel]);
        window.Add((int)sourceImage.Data[y, x - 1, channel]);
        window.Add((int)sourceImage.Data[y + 1, x - 1, channel]);
        window.Add((int)sourceImage.Data[y, x, channel]);
        window.Sort();
        resultImage.Data[y, x, channel] = (byte)window[5];
        continue;
    }
    if ((y == resultImage.Height-1) && (x == resultImage.Width-1))
    {
        window.Add((int)sourceImage.Data[y, x, channel]);
        window.Add((int)sourceImage.Data[y, x, channel]);
        window.Add((int)sourceImage.Data[y, x, channel]);
        window.Add((int)sourceImage.Data[y - 1, x, channel]);
        window.Add((int)sourceImage.Data[y, x - 1, channel]);
        window.Add((int)sourceImage.Data[y - 1, x - 1, channel]);
        window.Add((int)sourceImage.Data[y, x - 1, channel]);
        window.Add((int)sourceImage.Data[y - 1, x, channel]);
        window.Sort();
        resultImage.Data[y, x, channel] = (byte)window[5];
        continue;
    }
    if ((y == 0))
    {
        window.Add((int)sourceImage.Data[y + 1, x, channel]);
        window.Add((int)sourceImage.Data[y, x + 1, channel]);
        window.Add((int)sourceImage.Data[y + 1, x + 1, channel]);
        window.Add((int)sourceImage.Data[y, x, channel]);
        window.Add((int)sourceImage.Data[y, x - 1, channel]);
```

```
                window.Add((int)sourceImage.Data[y, x - 1, channel]);
                window.Add((int)sourceImage.Data[y + 1, x - 1, channel]);
                window.Add((int)sourceImage.Data[y, x + 1, channel]);
                window.Sort();
                resultImage.Data[y, x, channel] = (byte)window[5];
                continue;
            }
            if ((x == 0))
            {
                window.Add((int)sourceImage.Data[y + 1, x, channel]);
                window.Add((int)sourceImage.Data[y, x + 1, channel]);
                window.Add((int)sourceImage.Data[y + 1, x + 1, channel]);
                window.Add((int)sourceImage.Data[y - 1, x, channel]);
                window.Add((int)sourceImage.Data[y, x, channel]);
                window.Add((int)sourceImage.Data[y - 1, x, channel]);
                window.Add((int)sourceImage.Data[y + 1, x, channel]);
                window.Add((int)sourceImage.Data[y - 1, x + 1, channel]);
                window.Sort();
                resultImage.Data[y, x, channel] = (byte)window[5];
                continue;
            }
            if ((y == resultImage.Height-1) )
            {
                window.Add((int)sourceImage.Data[y, x, channel]);
                window.Add((int)sourceImage.Data[y, x + 1, channel]);
                window.Add((int)sourceImage.Data[y, x + 1, channel]);
                window.Add((int)sourceImage.Data[y - 1, x, channel]);
                window.Add((int)sourceImage.Data[y, x - 1, channel]);
                window.Add((int)sourceImage.Data[y - 1, x - 1, channel]);
                window.Add((int)sourceImage.Data[y, x - 1, channel]);
                window.Add((int)sourceImage.Data[y - 1, x + 1, channel]);
                window.Sort();
                resultImage.Data[y, x, channel] = (byte)window[5];
                continue;
            }
            if (x == resultImage.Width-1)
            {
                window.Add((int)sourceImage.Data[y + 1, x, channel]);
                window.Add((int)sourceImage.Data[y, x, channel]);
                window.Add((int)sourceImage.Data[y + 1, x, channel]);
                window.Add((int)sourceImage.Data[y - 1, x, channel]);
                window.Add((int)sourceImage.Data[y, x - 1, channel]);
                window.Add((int)sourceImage.Data[y - 1, x - 1, channel]);
                window.Add((int)sourceImage.Data[y + 1, x - 1, channel]);
                window.Add((int)sourceImage.Data[y - 1, x, channel]);
                window.Sort();
                resultImage.Data[y, x, channel] = (byte)window[5];
                continue;
            }
            window.Add((int)sourceImage.Data[y + 1, x, channel]);
            window.Add((int)sourceImage.Data[y, x + 1, channel]);
            window.Add((int)sourceImage.Data[y + 1, x + 1, channel]);
            window.Add((int)sourceImage.Data[y - 1, x, channel]);
            window.Add((int)sourceImage.Data[y, x - 1, channel]);
            window.Add((int)sourceImage.Data[y - 1, x - 1, channel]);
            window.Add((int)sourceImage.Data[y + 1, x - 1, channel]);
            window.Add((int)sourceImage.Data[y - 1, x + 1, channel]);
            window.Sort();
            resultImage.Data[y, x, channel] = (byte)window[5];
        }
    imageBox2.Image = resultImage.Resize(400, 400, Inter.Linear);
}

private void menu8_Click(object sender, EventArgs e)
{
    goToMethod(menu8);
}
```

```
private void convertFor8_Click(object sender, EventArgs e)
{
    int[,] Matrix = new int[3, 3];
    Matrix[0, 0] = Decimal.ToInt32(matrix00.Value); Matrix[0, 1] = Decimal.ToInt32(matrix01.Value); Matrix[0, 2] =
Decimal.ToInt32(matrix02.Value);
    Matrix[1, 0] = Decimal.ToInt32(matrix10.Value); Matrix[1, 1] = Decimal.ToInt32(matrix11.Value); Matrix[1, 2] =
Decimal.ToInt32(matrix12.Value);
    Matrix[2, 0] = Decimal.ToInt32(matrix20.Value); Matrix[2, 1] = Decimal.ToInt32(matrix21.Value); Matrix[2, 2] =
Decimal.ToInt32(matrix22.Value);
    Image<Bgr, byte> resultImage = new Image<Bgr, byte>(sourceImage.Size);
    int channel = 0, k = 0, m = 0, Sum_Pix_Mat, Sum_Mat;
    for (int x = 0; x < sourceImage.Size.Width; x++)
    {
        for (int y = 0; y < sourceImage.Size.Height; y++)
        {
            if (x == 0 || y == 0 || x == sourceImage.Size.Width - 1 || y == sourceImage.Size.Height - 1)
            {
                resultImage[y, x] = sourceImage[y, x];
                continue;
            }
            channel = 0; k = 0; m = 0; Sum_Pix_Mat = 0; Sum_Mat = 0;
            for (int i = y - 1; i <= y + 1; i++)
            {
                for (int j = x - 1; j <= x + 1; j++)
                {
                    Sum_Pix_Mat += sourceImage.Data[i, j, channel] * Matrix[k, m];
                    Sum_Mat += Matrix[k, m];
                    m++;
                }
                m = 0;
                k++;
            }
            if (Sum_Mat == 0)
                resultImage.Data[y, x, channel] = Convert.ToByte(Make_Color(Sum_Pix_Mat));
            else
                resultImage.Data[y, x, channel] = Convert.ToByte(Make_Color(Sum_Pix_Mat / Sum_Mat));
            channel = 1; k = 0; m = 0; Sum_Pix_Mat = 0; Sum_Mat = 0;
            for (int i = y - 1; i <= y + 1; i++)
            {
                for (int j = x - 1; j <= x + 1; j++)
                {
                    Sum_Pix_Mat += sourceImage.Data[i, j, channel] * Matrix[k, m];
                    Sum_Mat += Matrix[k, m];
                    m++;
                }
                m = 0;
                k++;
            }
            if (Sum_Mat == 0)
                resultImage.Data[y, x, channel] = Convert.ToByte(Make_Color(Sum_Pix_Mat));
            else
                resultImage.Data[y, x, channel] = Convert.ToByte(Make_Color(Sum_Pix_Mat / Sum_Mat));
            channel = 2; k = 0; m = 0; Sum_Pix_Mat = 0; Sum_Mat = 0;
            for (int i = y - 1; i <= y + 1; i++)
            {
                for (int j = x - 1; j <= x + 1; j++)
                {
                    Sum_Pix_Mat += sourceImage.Data[i, j, channel] * Matrix[k, m];
                    Sum_Mat += Matrix[k, m];
                    m++;
                }
                m = 0;
                k++;
            }
            if (Sum_Mat == 0)
                resultImage.Data[y, x, channel] = Convert.ToByte(Make_Color(Sum_Pix_Mat));
```

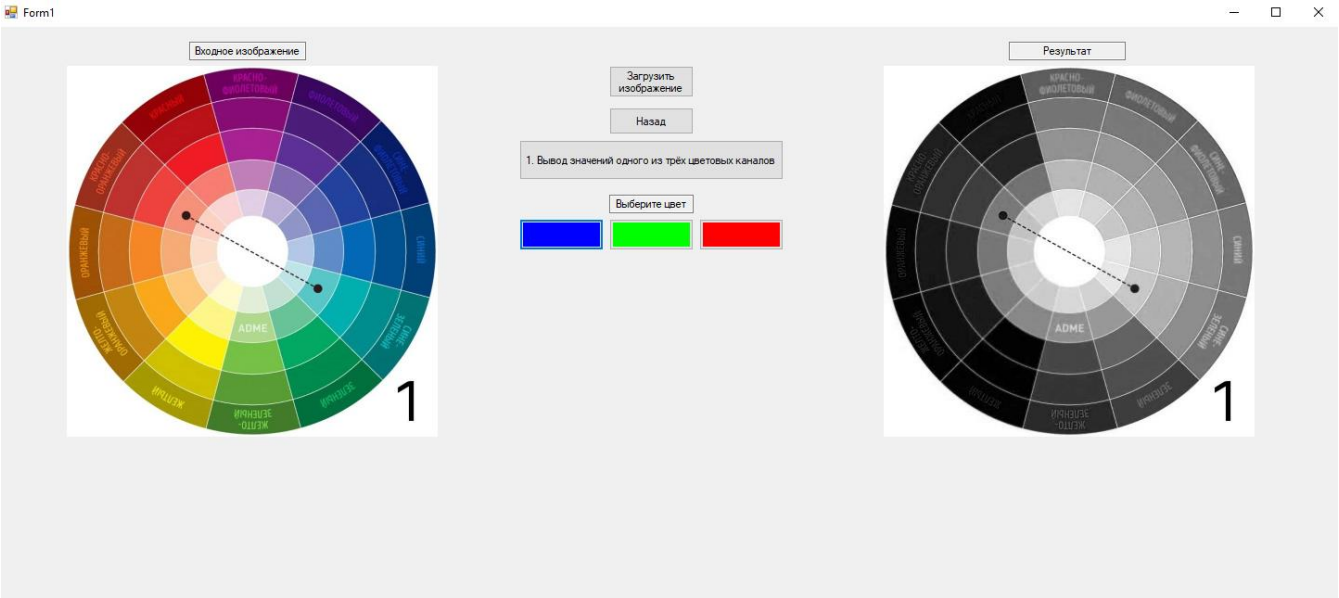
```
                else
                    resultImage.Data[y, x, channel] = Convert.ToByte(Make_Color(Sum_Pix_Mat / Sum_Mat));
            }
        }
        imageBox2.Image = resultImage.Resize(400, 400, Inter.Linear);
    }

    private void menu9_Click(object sender, EventArgs e)
    {
        goToMethod(menu9);
    }

    private void convertFor9_Click(object sender, EventArgs e)
    {
        int[,] Matrix = new int[3, 3];
        Matrix[0, 0] = Decimal.ToInt32(Matrix000.Value); Matrix[0, 1] = Decimal.ToInt32(Matrix001.Value); Matrix[0, 2]
= Decimal.ToInt32(Matrix002.Value);
        Matrix[1, 0] = Decimal.ToInt32(Matrix010.Value); Matrix[1, 1] = Decimal.ToInt32(Matrix011.Value); Matrix[1, 2]
= Decimal.ToInt32(Matrix012.Value);
        Matrix[2, 0] = Decimal.ToInt32(Matrix020.Value); Matrix[2, 1] = Decimal.ToInt32(Matrix021.Value); Matrix[2, 2]
= Decimal.ToInt32(Matrix022.Value);
        Image<Bgr, byte> resultImage = new Image<Bgr, byte>(sourceImage3.Size);
        int channel = 0, k = 0, m = 0, Sum_Pix_Mat, Sum_Mat;
        for (int x = 0; x < sourceImage3.Size.Width; x++)
        {
            for (int y = 0; y < sourceImage3.Size.Height; y++)
            {
                if (x == 0 || y == 0 || x == sourceImage3.Size.Width - 1 || y == sourceImage3.Size.Height - 1)
                {
                    resultImage[y, x] = sourceImage3[y, x];
                    continue;
                }
                channel = 0; k = 0; m = 0; Sum_Pix_Mat = 0; Sum_Mat = 0;
                for (int i = y - 1; i <= y + 1; i++)
                {
                    for (int j = x - 1; j <= x + 1; j++)
                    {
                        Sum_Pix_Mat += sourceImage3.Data[i, j, channel] * Matrix[k, m];
                        Sum_Mat += Matrix[k, m];
                        m++;
                    }
                    m = 0;
                    k++;
                }
                if (Sum_Mat == 0)
                    resultImage.Data[y, x, channel] = Convert.ToByte(Make_Color(Sum_Pix_Mat));
                else
                    resultImage.Data[y, x, channel] = Convert.ToByte(Make_Color(Sum_Pix_Mat / Sum_Mat));
                channel = 1; k = 0; m = 0; Sum_Pix_Mat = 0; Sum_Mat = 0;
                for (int i = y - 1; i <= y + 1; i++)
                {
                    for (int j = x - 1; j <= x + 1; j++)
                    {
                        Sum_Pix_Mat += sourceImage3.Data[i, j, channel] * Matrix[k, m];
                        Sum_Mat += Matrix[k, m];
                        m++;
                    }
                    m = 0;
                    k++;
                }
                if (Sum_Mat == 0)
                    resultImage.Data[y, x, channel] = Convert.ToByte(Make_Color(Sum_Pix_Mat));
                else
                    resultImage.Data[y, x, channel] = Convert.ToByte(Make_Color(Sum_Pix_Mat / Sum_Mat));
                channel = 2; k = 0; m = 0; Sum_Pix_Mat = 0; Sum_Mat = 0;
                for (int i = y - 1; i <= y + 1; i++)
                {
```

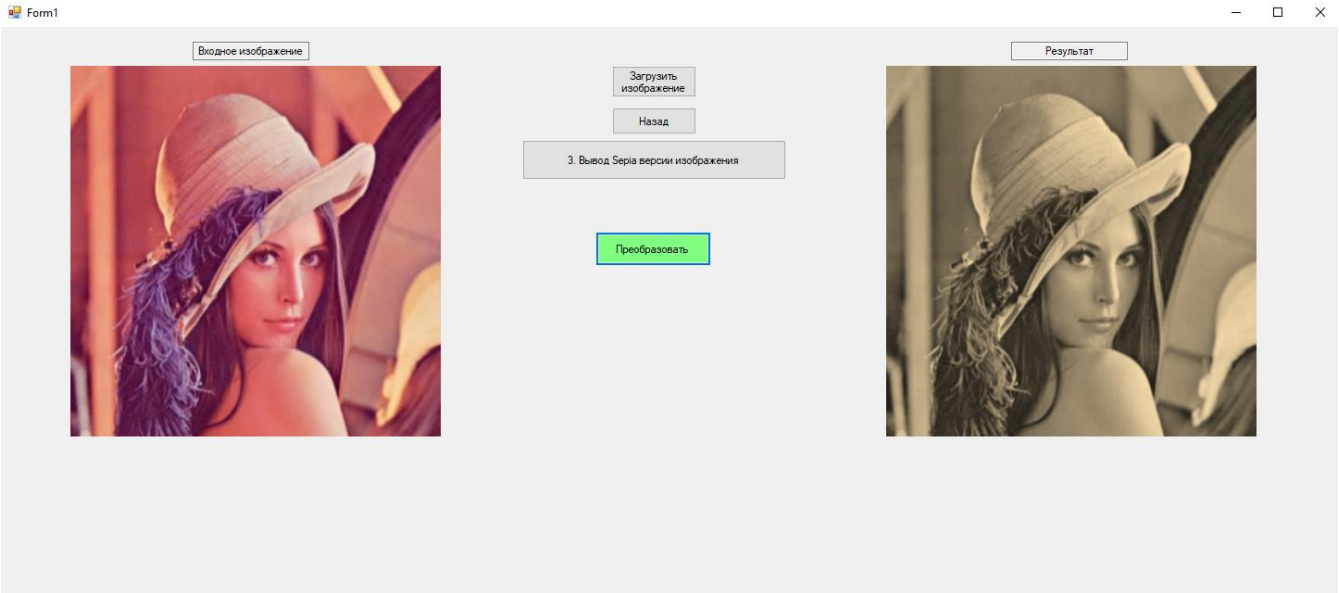
```
                for (int j = x - 1; j <= x + 1; j++)
                {
                    Sum_Pix_Mat += sourceImage3.Data[i, j, channel] * Matrix[k, m];
                    Sum_Mat += Matrix[k, m];
                    m++;
                }
                m = 0;
                k++;
            }
            if (Sum_Mat == 0)
                resultImage.Data[y, x, channel] = Convert.ToByte(Make_Color(Sum_Pix_Mat));
            else
                resultImage.Data[y, x, channel] = Convert.ToByte(Make_Color(Sum_Pix_Mat / Sum_Mat));
        }
    }
    imageBox2.Image = resultImage.Resize(400, 400, Inter.Linear);
}

private void menu10_Click(object sender, EventArgs e)
{
    goToMethod(menu10);
}

private void convertFor10_Click(object sender, EventArgs e)
{
    var grayImage = new Image<Gray, byte>(sourceImage.Size);
    for (int x = 0; x < grayImage.Width; x++)
        for (int y = 0; y < grayImage.Height; y++) // обход по пискелям
            grayImage.Data[y, x, 0] = Convert.ToByte(0.299 * sourceImage.Data[y, x, 2] + 0.587 * sourceImage.Data[y, x,
1] + 0.114 * sourceImage.Data[y, x, 0]);

    var resultImage = grayImage;
    for (int channel = 0; channel < resultImage.NumberOfChannels; channel++) //обход по каналам
        for (int x = 0; x < resultImage.Width; x++)
            for (int y = 0; y < resultImage.Height; y++)
            {
                List<int> window = new List<int>();
                window.Add((int)grayImage.Data[y, x, channel]);

                if ((y == 0) && (x == 0))
                {
                    window.Add((int)grayImage.Data[y + 1, x, channel]);
                    window.Add((int)grayImage.Data[y, x + 1, channel]);
                    window.Add((int)grayImage.Data[y + 1, x + 1, channel]);
                    window.Add((int)grayImage.Data[y, x, channel]);
                    window.Add((int)grayImage.Data[y, x, channel]);
                    window.Add((int)grayImage.Data[y, x, channel]);
                    window.Add((int)grayImage.Data[y + 1, x, channel]);
                    window.Add((int)grayImage.Data[y, x + 1, channel]);
                    window.Sort();
                    resultImage.Data[y, x, channel] = (byte)window[5];
                    continue;
                }
                if ((y == resultImage.Height - 1) && (x == 0))
                {
                    window.Add((int)grayImage.Data[y, x, channel]);
                    window.Add((int)grayImage.Data[y, x + 1, channel]);
                    window.Add((int)grayImage.Data[y, x + 1, channel]);
                    window.Add((int)grayImage.Data[y - 1, x, channel]);
                    window.Add((int)grayImage.Data[y, x, channel]);
                    window.Add((int)grayImage.Data[y - 1, x, channel]);
                    window.Add((int)grayImage.Data[y, x, channel]);
                    window.Add((int)grayImage.Data[y - 1, x + 1, channel]);
                    window.Sort();
                    resultImage.Data[y, x, channel] = (byte)window[5];
                    continue;
                }
```

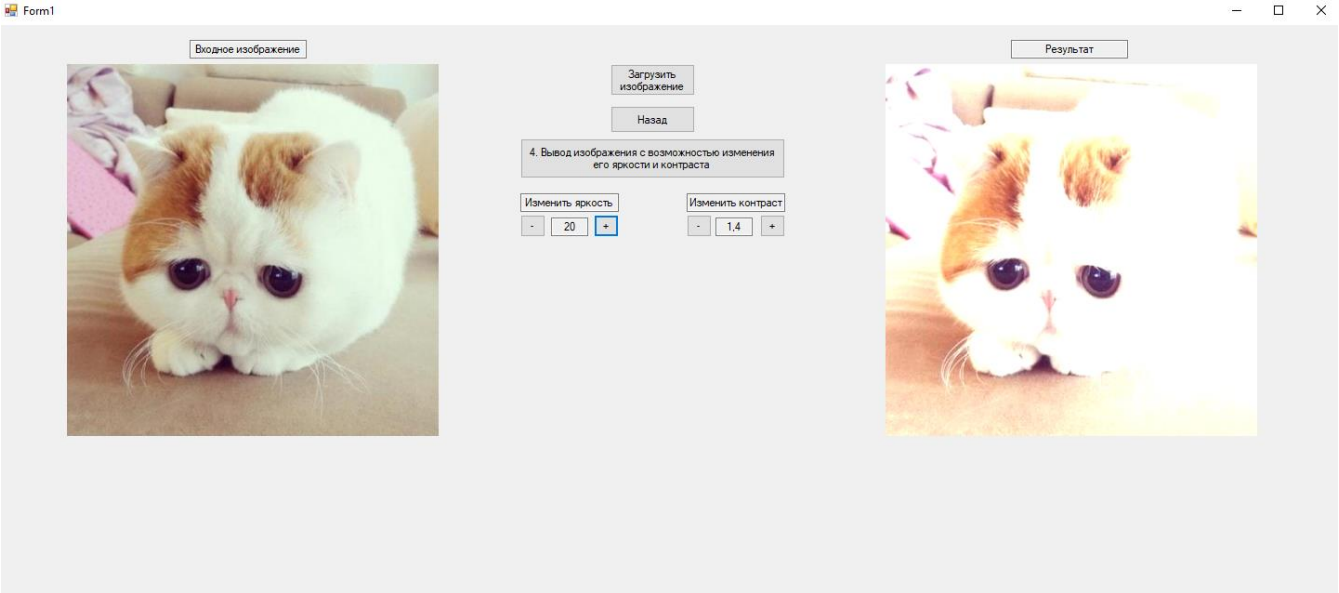
```
if ((y == 0) && (x == resultImage.Width - 1))
{
    window.Add((int)grayImage.Data[y + 1, x, channel]);
    window.Add((int)grayImage.Data[y, x, channel]);
    window.Add((int)grayImage.Data[y + 1, x, channel]);
    window.Add((int)grayImage.Data[y, x, channel]);
    window.Add((int)grayImage.Data[y, x - 1, channel]);
    window.Add((int)grayImage.Data[y, x - 1, channel]);
    window.Add((int)grayImage.Data[y + 1, x - 1, channel]);
    window.Add((int)grayImage.Data[y, x, channel]);
    window.Sort();
    resultImage.Data[y, x, channel] = (byte)window[5];
    continue;
}
if ((y == resultImage.Height - 1) && (x == resultImage.Width - 1))
{
    window.Add((int)grayImage.Data[y, x, channel]);
    window.Add((int)grayImage.Data[y, x, channel]);
    window.Add((int)grayImage.Data[y, x, channel]);
    window.Add((int)grayImage.Data[y - 1, x, channel]);
    window.Add((int)grayImage.Data[y, x - 1, channel]);
    window.Add((int)grayImage.Data[y - 1, x - 1, channel]);
    window.Add((int)grayImage.Data[y, x - 1, channel]);
    window.Add((int)grayImage.Data[y - 1, x, channel]);
    window.Sort();
    resultImage.Data[y, x, channel] = (byte)window[5];
    continue;
}
if ((y == 0))
{
    window.Add((int)grayImage.Data[y + 1, x, channel]);
    window.Add((int)grayImage.Data[y, x + 1, channel]);
    window.Add((int)grayImage.Data[y + 1, x + 1, channel]);
    window.Add((int)grayImage.Data[y, x, channel]);
    window.Add((int)grayImage.Data[y, x - 1, channel]);
    window.Add((int)grayImage.Data[y, x - 1, channel]);
    window.Add((int)grayImage.Data[y + 1, x - 1, channel]);
    window.Add((int)grayImage.Data[y, x + 1, channel]);
    window.Sort();
    resultImage.Data[y, x, channel] = (byte)window[5];
    continue;
}
if ((x == 0))
{
    window.Add((int)grayImage.Data[y + 1, x, channel]);
    window.Add((int)grayImage.Data[y, x + 1, channel]);
    window.Add((int)grayImage.Data[y + 1, x + 1, channel]);
    window.Add((int)grayImage.Data[y - 1, x, channel]);
    window.Add((int)grayImage.Data[y, x, channel]);
    window.Add((int)grayImage.Data[y - 1, x, channel]);
    window.Add((int)grayImage.Data[y + 1, x, channel]);
    window.Add((int)grayImage.Data[y - 1, x + 1, channel]);
    window.Sort();
    resultImage.Data[y, x, channel] = (byte)window[5];
    continue;
}
if ((y == resultImage.Height - 1))
{
    window.Add((int)grayImage.Data[y, x, channel]);
    window.Add((int)grayImage.Data[y, x + 1, channel]);
    window.Add((int)grayImage.Data[y, x + 1, channel]);
    window.Add((int)grayImage.Data[y - 1, x, channel]);
    window.Add((int)grayImage.Data[y, x - 1, channel]);
    window.Add((int)grayImage.Data[y - 1, x - 1, channel]);
    window.Add((int)grayImage.Data[y, x - 1, channel]);
    window.Add((int)grayImage.Data[y - 1, x + 1, channel]);
    window.Sort();
```

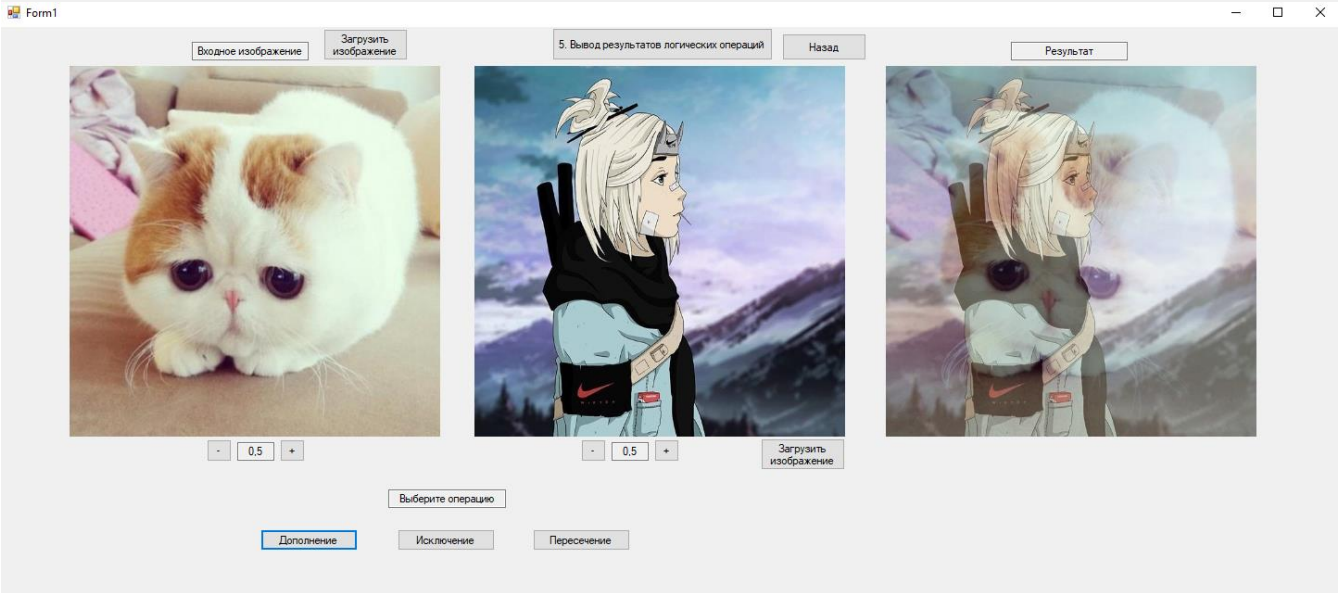
```
                resultImage.Data[y, x, channel] = (byte)window[5];
                continue;
            }
            if (x == resultImage.Width - 1)
            {
                window.Add((int)grayImage.Data[y + 1, x, channel]);
                window.Add((int)grayImage.Data[y, x, channel]);
                window.Add((int)grayImage.Data[y + 1, x, channel]);
                window.Add((int)grayImage.Data[y - 1, x, channel]);
                window.Add((int)grayImage.Data[y, x - 1, channel]);
                window.Add((int)grayImage.Data[y - 1, x - 1, channel]);
                window.Add((int)grayImage.Data[y + 1, x - 1, channel]);
                window.Add((int)grayImage.Data[y - 1, x, channel]);
                window.Sort();
                resultImage.Data[y, x, channel] = (byte)window[5];
                continue;
            }
            window.Add((int)grayImage.Data[y + 1, x, channel]);
            window.Add((int)grayImage.Data[y, x + 1, channel]);
            window.Add((int)grayImage.Data[y + 1, x + 1, channel]);
            window.Add((int)grayImage.Data[y - 1, x, channel]);
            window.Add((int)grayImage.Data[y, x - 1, channel]);
            window.Add((int)grayImage.Data[y - 1, x - 1, channel]);
            window.Add((int)grayImage.Data[y + 1, x - 1, channel]);
            window.Add((int)grayImage.Data[y - 1, x + 1, channel]);
            window.Sort();
            resultImage.Data[y, x, channel] = (byte)window[5];
        }
    var edges = resultImage.Convert<Gray, byte>();
    edges = edges.ThresholdAdaptive(new Gray(100), AdaptiveThresholdType.MeanC, ThresholdType.Binary, 3, new
Gray(0.03));
    var si1 = sourceImage.Resize(400, 400, Inter.Linear);
    var si2 = edges.Convert<Bgr, byte>().Resize(400, 400, Inter.Linear);
    imageBox2.Image = Add_Image_NoCof(sourceImage, edges.Convert<Bgr, byte>());

}

private void menu6_Click(object sender, EventArgs e)
{
    goToMethod(menu6);
}

private void countOfK2_TextChanged(object sender, EventArgs e)
{

}

private void minusContrast_Click(object sender, EventArgs e)
{
    contrast -= 0.1;
    if (contrast > 0.1 )
        countOfContrast.Text = contrast.ToString();
    else
        countOfContrast.Text = "0";
    ImageWithBrightnessAndContrast();
    statusMinusCnButton();
}

private void menu5_Click(object sender, EventArgs e)
{
    goToMethod(menu5);
}

private void plusContrast_Click(object sender, EventArgs e)
{
    contrast += 0.1;
    countOfContrast.Text = contrast.ToString();
```

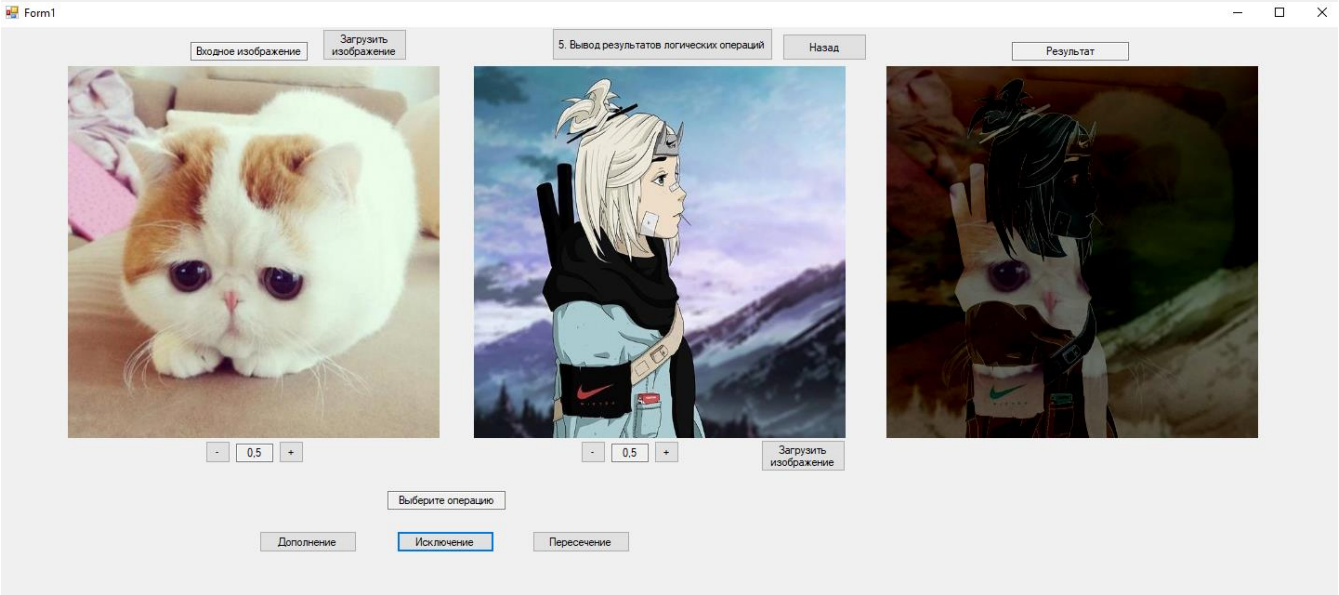
```
            ImageWithBrightnessAndContrast();
            statusPlusCnButton();
        }
    }
}
```

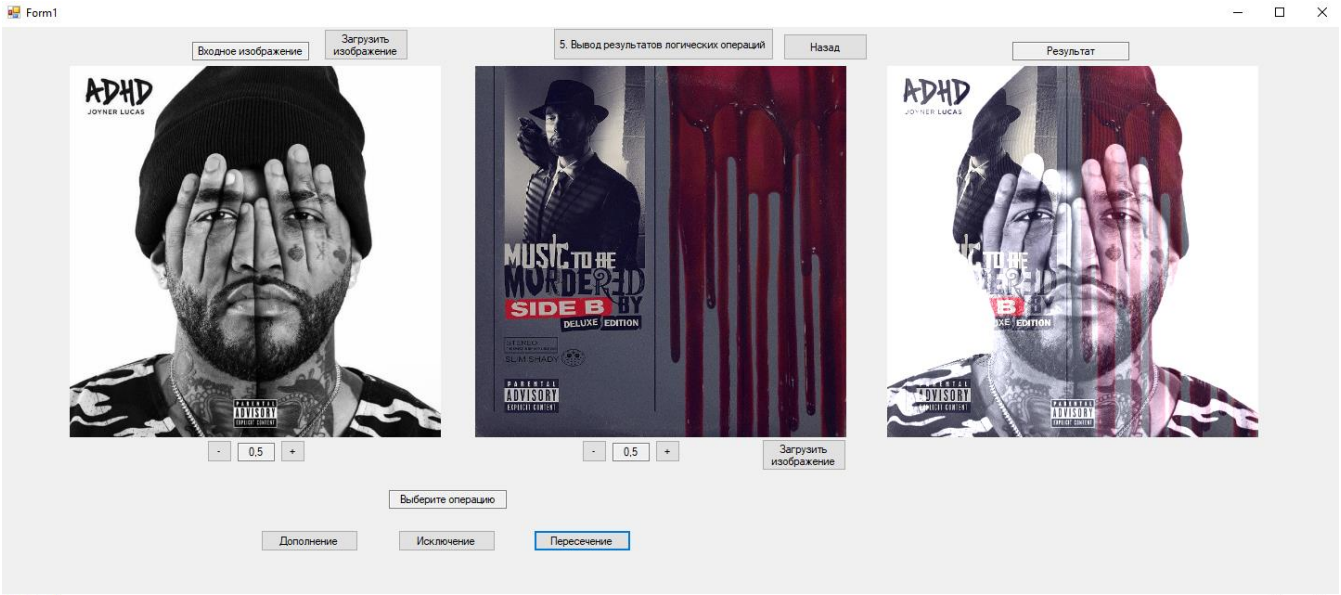


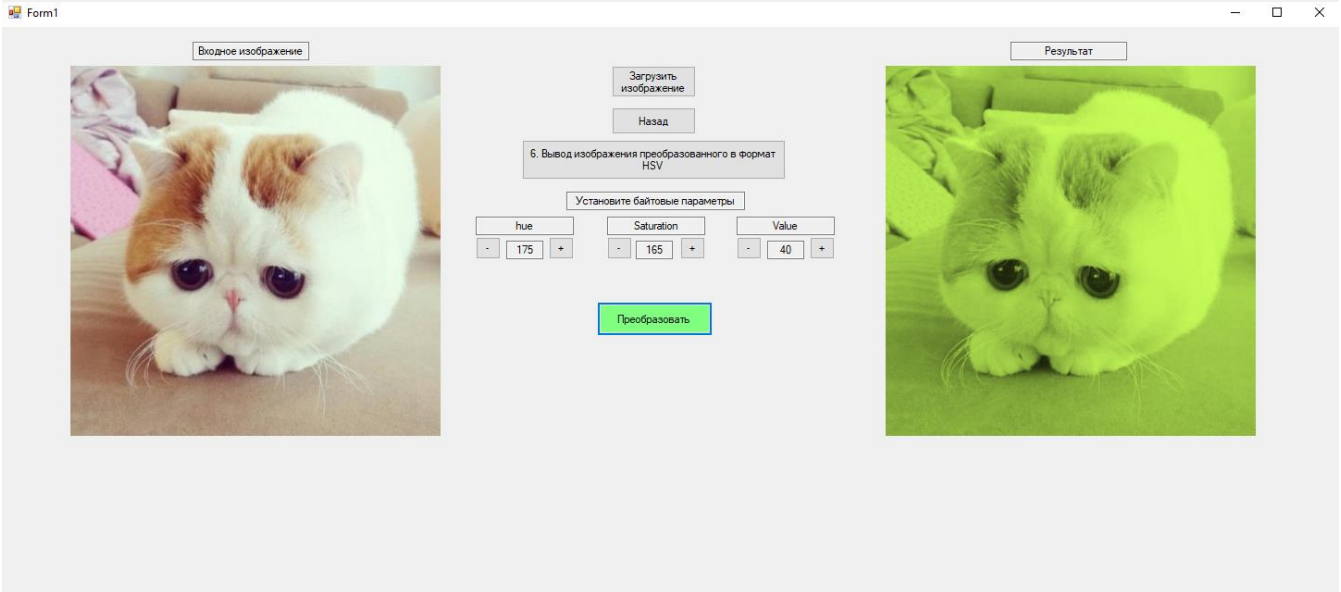


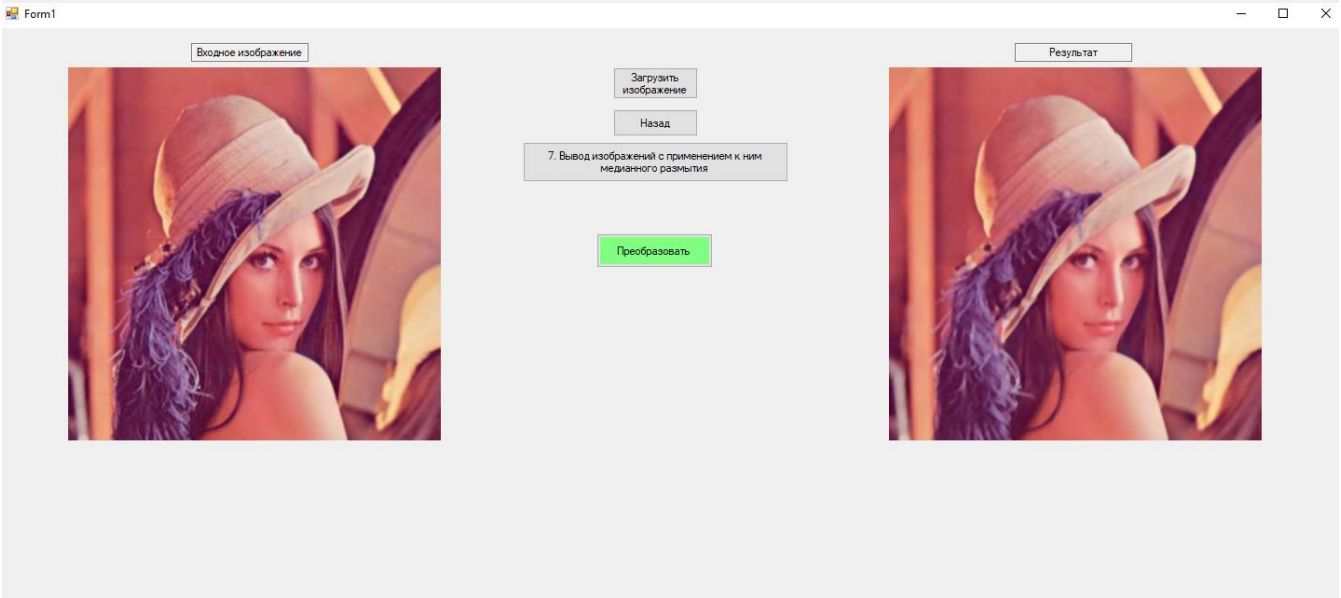

Form1
Входное изображение
Загрузить изображение
Назад
4. Вывод изображения с возможностью изменения его яркости и контраста
Изменить яркость
-
20
+
Изменить контраст
-
1,4
+
Результат

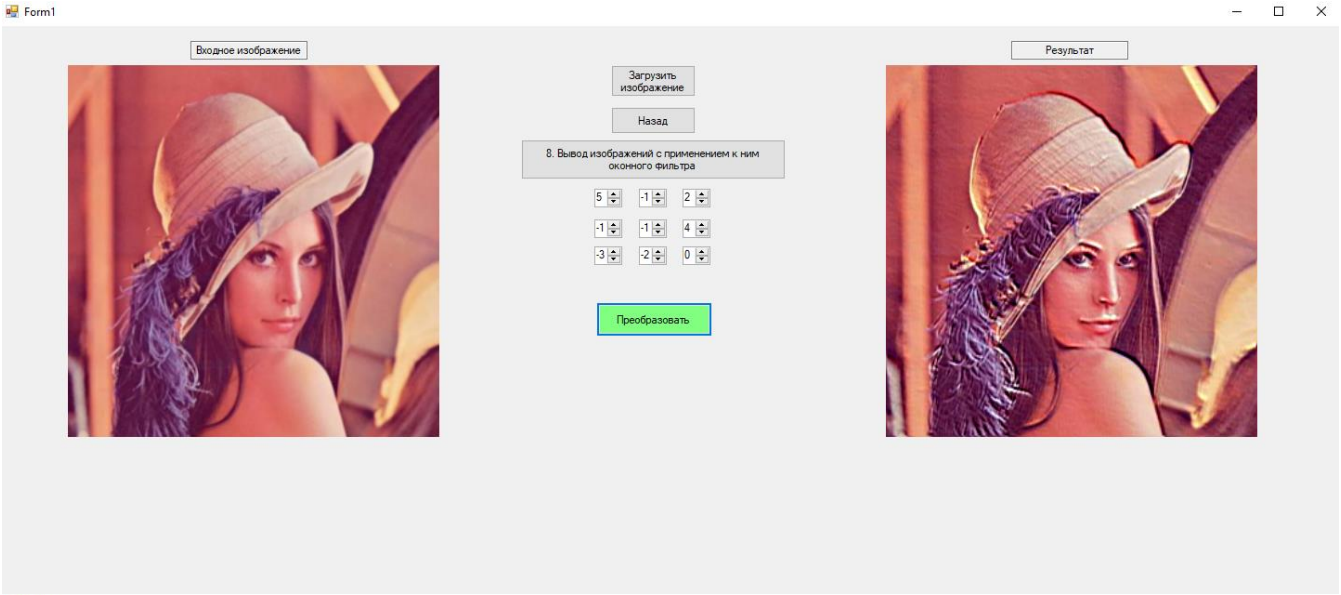

Form1
Входное изображение
Загрузить изображение
5. Вывод результатов логических операций
Назад
Результат
-
0,5
+
-
0,5
+
Загрузить изображение
Выберите операцию
Дополнение
Исключение
Пересечение

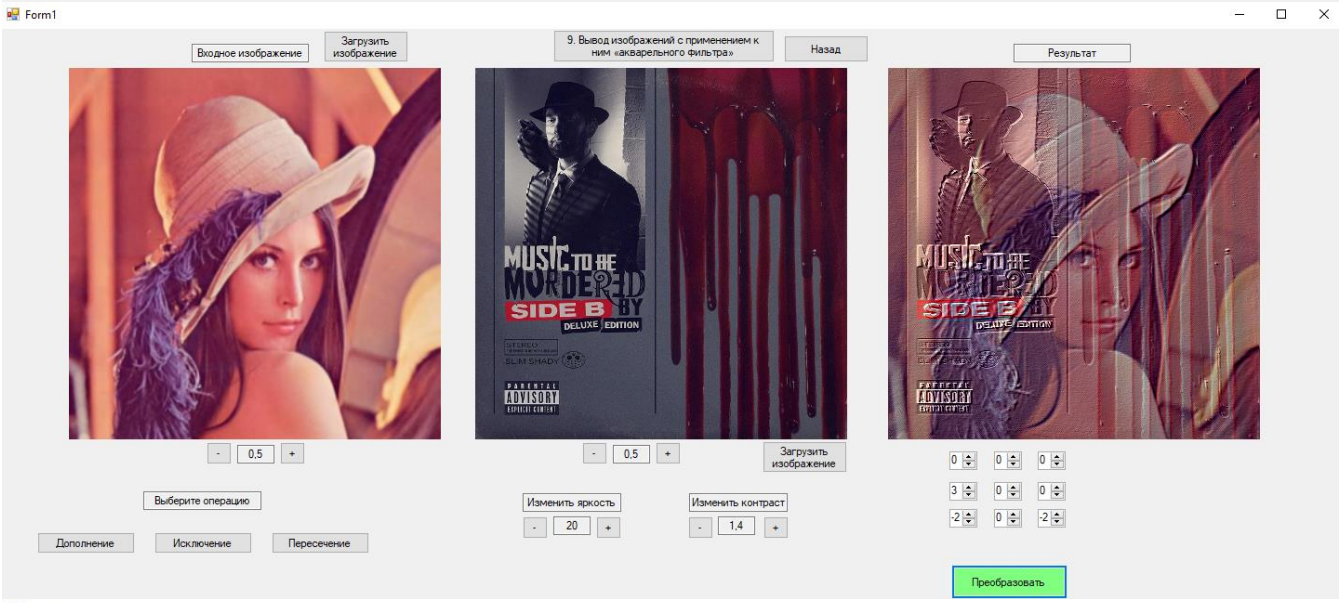

Form1
Входное изображение
Загрузить изображение
5. Вывод результатов логических операций
Назад
Результат
-
0,5
+
-
0,5
+
Загрузить изображение
Выберите операцию
Дополнение
Исключение
Пересечение

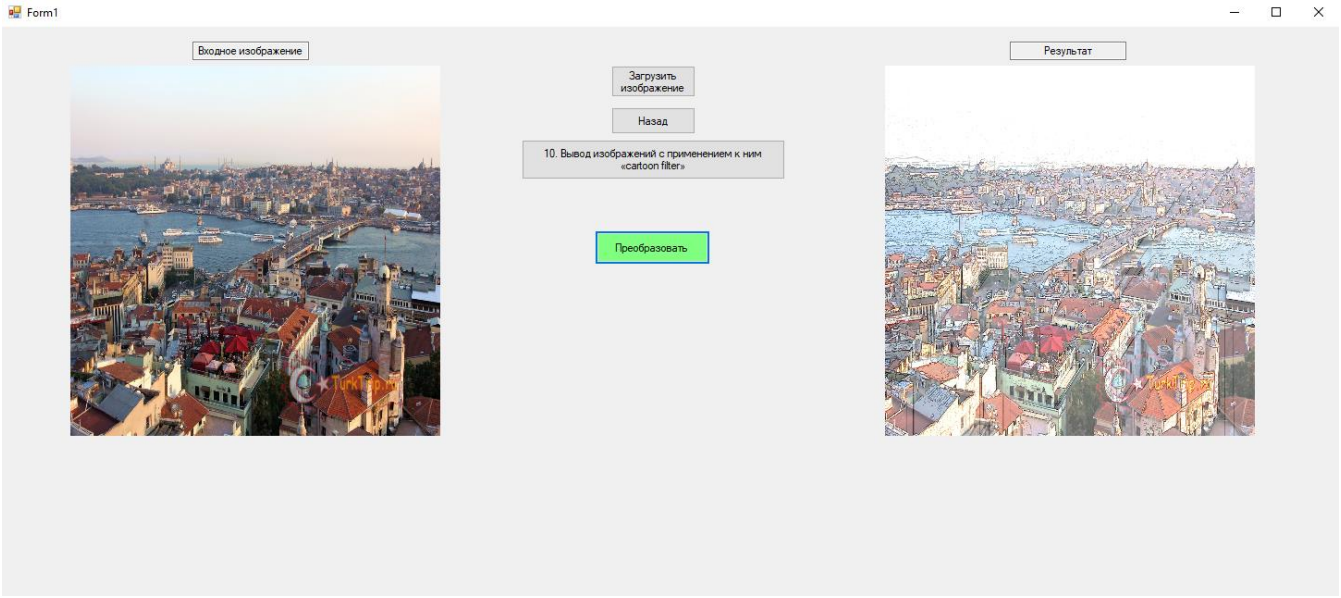

Form1
Входное изображение
Загрузить изображение
5. Вывод результатов логических операций
Назад
Результат
ADHD
JOYNER LUCAS
MUSIC TO BE MURDERED BY
SIDE B
DELUXE EDITION
SLIM SHADY
PARENTAL ADVISORY
EXPLICIT CONTENT
-
0,5
+
Загрузить изображение
Выберите операцию
Дополнение
Исключение
Пересечение


Form1
Входное изображение
Результат
Загрузить изображение
Назад
6. Вывод изображения преобразованного в формат HSV
Установите байтовые параметры
hue
Saturation
Value
-
175
+
-
165
+
-
40
+
Преобразовать


Form1
Входное изображение
Результат
Загрузить изображение
Назад
7. Вывод изображений с применением к ним медианного размытия
Преобразовать


Form1
Входное изображение
Загрузить изображение
Назад
8. Вывод изображений с применением к ним оконного фильтра
5 -1 2
-1 -1 4
-3 -2 0
Преобразовать
Результат


Form1
Входное изображение
Загрузить изображение
9. Вывод изображений с применением к ним «акварельного фильтра»
Назад
Результат
- 0,5 +
Выберите операцию
Дополнение
Исключение
Пересечение
Изменить яркость
- 20 +
Изменить контраст
- 1,4 +
Загрузить изображение
0 0 0
3 0 0
-2 0 -2
Преобразовать


Form1
Входное изображение
Загрузить изображение
Назад
10. Вывод изображений с применением к ним «cartoon filter»
Преобразовать
Результат