

МИНОБРНАУКИ РОССИИ
федеральное государственное автономное образовательное учреждение
высшего образования
«Санкт-Петербургский политехнический университет Петра Великого»
(ФГАОУ ВО «СПбПУ»)
Университетский политехнический колледж

Утверждаю
Зам. директора по УМР
Е.Г.Конакина _____
«__» ____ 2017 г.

ОТЧЕТ
по производственной практике

по профессиональному модулю ПМ.01 «Разработка программных модулей
программного обеспечения для компьютерных систем»
ПП.01.02 «Прикладное программирование»
(код и наименование)

Специальность 09.02.03 «Программирование в компьютерных системах»
(код и наименование специальности)

Студент II курса 22928/2 группы
Тимушев Фёдор Алексеевич
(Фамилия, имя, отчество)

Место прохождения практики
Вычислительный центр Университетского политехнического колледжа,
пр. Энгельса д.23
(наименование и адрес организации)

Период прохождения практики
с «20» февраля 2017 г. по «11» марта 2017 г.

Руководитель(и) практики:

(подпись)

Девятко Н.С.
(Ф.И.О.)

Итоговая оценка по практике

Санкт-Петербург
2017

Утверждаю
Зам. директора по УМР
Е.Г.Конакина _____
«___» _____ 2017 г.

Задание на учебную практику

по профессиональному модулю ПМ.01 «Разработка программных модулей
программного обеспечения для компьютерных систем»

ПП.01.02 «Прикладное программирование»

(код и наименование)

Специальность 09.02.03 «Программирование в компьютерных системах»

(код и наименование специальности)

Студент II курса 22928/2 группы

Тимушев Фёдор Алексеевич

(Фамилия, имя, отчество)

Место прохождения практики

Вычислительный центр Университетского политехнического колледжа,
пр. Энгельса д.23

(наименование и адрес организации)

Период прохождения практики

с «20» февраля 2017 г. по «11» марта 2017 г.

Виды работ, обязательные для выполнения

Разработка спецификаций отдельных компонент

Проектирование программного обеспечения на уровне модулей

Создание модулей

Отладка и тестирование модулей

Разработка технической документации с использованием инструментальных средств

Индивидуальное задание

ВАРИАНТ 5, 6, 7, 26

Задание выдал «20» февраля 2017 г.

(подпись)

Девятко Н.С.
(Ф.И.О.)

Тема 1: Вычисление функции с помощью разложения в ряд

Задание 1 (5 баллов), задание 1 и 2 (10 баллов)

Задание 1

Необходимо разработать программу, вычисляющую значение функции с помощью разложения в ряд. Сумма ряда вычисляется при помощи цикла с неизвестным числом повторений, так как требуется найти значение с заданной точностью (точность вводится с клавиатуры). Сходящийся числовой ряд будет достигать искомого значения при большом количестве суммируемых членов ряда. При этом разность между соседними членами последовательности станет очень мала. Будем считать, что необходимая точность вычислений достигнута, если разность между соседними элементами ряда меньше заданной точности (условие выхода из цикла). Поэтому на каждом проходе цикла нужно запоминать предыдущий член ряда. Это удобно, так как следующий член ряда вычисляется всегда через предыдущий (не используйте возведение в степень, вместо этого будет простое умножение).

Для проверки полученного результата выведите значение указанной функции (левую часть выражения), вычисленную в цикле сумму ряда (правую часть выражения), а также количество просуммированных членов ряда.

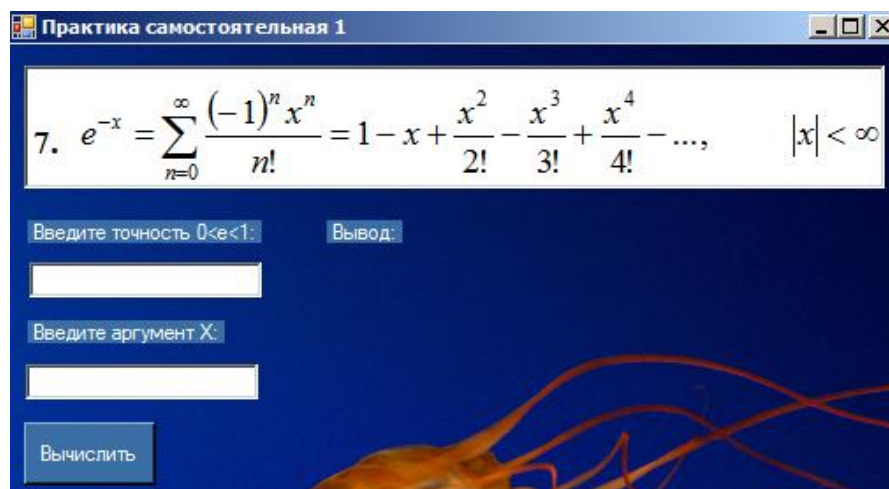
Задание 2

Добавьте проверку:

- ✓ правильности ввода данных при нажатии клавиш клавиатуры в поля редактирования (используйте событие KeyPress);
- ✓ диапазона вводимых данных;
- ✓ контроль пустых полей ввода (заблокируйте кнопку «Вычислить», пока не будут введены все данные).

Вариант № 7:

$$e^{-x} = \sum_{n=0}^{\infty} \frac{(-1)^n x^n}{n!} = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \frac{x^4}{4!} - \dots, \quad |x| < \infty$$



```

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 Практика_самостоятельная_1
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
            button1.Enabled = false;
        }
        private void textBox1_KeyPress(object sender, KeyPressEventArgs e) //проверка на ввод - только
числа и запятой для 1ого текстового
        {
            if ((e.KeyChar <= 47 || e.KeyChar >= 58) && e.KeyChar != 8 && e.KeyChar != '\b') e.Handled
= true;
            if (e.KeyChar == ',')
            {
                e.Handled = false;
            }
            if (textBox1.Text.Length >= 1 && textBox2.Text.Length >= 1) //проверка на наличие символов
в текстовых
                button1.Enabled = true;
            else
                button1.Enabled = false;
        }
        private void textBox2_KeyPress(object sender, KeyPressEventArgs e) //проверка на ввод - только
числа и запятой для 2ого текстового
        {
            if ((e.KeyChar <= 47 || e.KeyChar >= 58) && e.KeyChar != 8 && e.KeyChar != '\b') e.Handled
= true;
            if (e.KeyChar == ',')
            {
                e.Handled = false;
            }
            if (textBox1.Text.Length > 0 && textBox2.Text.Length > 0)
                button1.Enabled = true;
            else
                button1.Enabled = false;
        }
        private void button1_Click(object sender, EventArgs e)
        {
            int k = 0;
            double num1 = Convert.ToDouble(textBox1.Text);
            double x = Convert.ToDouble(textBox2.Text); // конвертация в дابل введенных переменных
            if (num1 <= 0 || num1 >= 1) //проверка на ввод диапазона
                MessageBox.Show("Введите правильный диапазон!");
            else
            {
                // num1 = 0
                double current = 1; // назначение первого элемента
                // частичная сумма
                double sum = current;
                for (int n = 1; ; n++)
                {
                    k++;
                    current *= -x / n;
                    sum += current;
                    if (Math.Abs(current) < num1) // возвращаем значение числа при условии, что оно <
первой переменной
                        break;
                }
                label3.Text = "Вывод = " + Math.Round(sum, 3) + "\nКол-во членов суммы: " + k;
            }
        }
    }
}

```

Практика самостоятельная 1

$$7. e^{-x} = \sum_{n=0}^{\infty} \frac{(-1)^n x^n}{n!} = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \frac{x^4}{4!} - \dots, \quad |x| < \infty$$

Введите точность $0 < \epsilon < 1$:

Вывод = 0,05
Кол-во членов суммы: 14

Введите аргумент X:

Практика самостоятельная 1

$$7. e^{-x} = \sum_{n=0}^{\infty} \frac{(-1)^n x^n}{n!} = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \frac{x^4}{4!} - \dots, \quad |x| < \infty$$

Введите точность $0 < \epsilon < 1$:

Вывод = 0,05
Кол-во членов суммы: 14

Введите аргумент X:

Введите правильный диапазон!

Тема 2: «Использование подпрограмм для обработки простых типов данных»

Задание 1 (5 баллов), задание 1 и 2 (10 баллов)

Требуется разработать программу, которая демонстрирует создание и использование собственной функции. Функция должна получать исходные данные через параметры, вычислять по заданию результат и передавать его в место вызова.

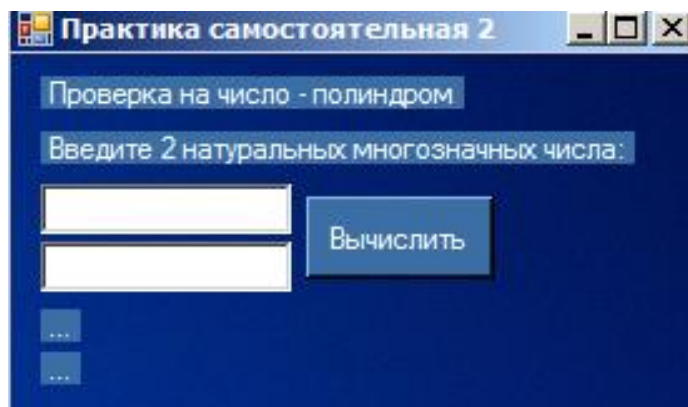
Функция с результатом передает одно вычисленное значение (результат) через оператор `return`. Функция может передавать и несколько результатов через параметры-переменные.

В языке C# для обмена информацией между вызываемой и вызывающей функцией предусмотрено 4 вида параметров:

- ✓ параметры-значения – подставляются при вызове функции, но не изменяются. Обычно используются в качестве входных параметров функции;
- ✓ параметры-переменные – подставляются в виде ссылки со служебным словом `ref`, переданное значение переменной может быть изменено внутри функции;
- ✓ выходные параметры, описываются ключевым словом `out`, используются для передачи результатов;
- ✓ параметры-массивы, описываются с помощью ключевого слова `params`

Вариант 6.

1. Разработать функцию, позволяющую распознавать числа-палиндромы («перевертыши»), то есть такие числа, десятичная запись которых одинаково читается слева направо и справа налево.
2. Даны два натуральных многозначных числа. Выяснить, является ли хоть одно из них палиндромом.



```

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 Практика_самостоятельная_2
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
            button1.Enabled = false;
        }
        private void textBox2_KeyPress(object sender, KeyPressEventArgs e)
        {
            if ((e.KeyChar <= 47 || e.KeyChar >= 58) && e.KeyChar != 8 && e.KeyChar != '\b')
e.Handled = true; //проверка на ввод: только числа и запятая
            if (e.KeyChar == ',')
            {
                e.Handled = false;
            }
            if ((textBox1.Text != String.Empty) && (textBox2.Text != String.Empty)) //проверка на
наличие символов в текстовых
                button1.Enabled = true;
            else
                button1.Enabled = false;
        }
        private void textBox1_KeyPress(object sender, KeyPressEventArgs e)
        {
            if ((e.KeyChar <= 47 || e.KeyChar >= 58) && e.KeyChar != 8 && e.KeyChar != '\b')
e.Handled = true; //проверка на ввод: только числа и запятая
            if (e.KeyChar == ',')
            {
                e.Handled = false;
            }
            if ((textBox1.Text != String.Empty) && (textBox2.Text != String.Empty)) //проверка на
наличие символов в текстовых
                button1.Enabled = true;
            else
                button1.Enabled = false;
        }
        private void button1_Click(object sender, EventArgs e)
        {
            string num1 = Convert.ToString(textBox1.Text); //конвертация вводимых символов в целые
числа для 2х переменных
            string num2 = Convert.ToString(textBox2.Text);
            int prov = 0;
            int prov2 = 0;
            for (int i = 0; i < num1.Length / 2; i++)
            {
                if (num1.Substring(i, 1) != (num1.Substring(num1.Length - 1 - i, 1))) //проверка на
отсутствие полиндрома для первого числа
                {
                    label3.Text = "Число " + num1 + " не явл. палиндромом."; //вывод сообщения об
отсутствии полиндрома
                }
                else
                    prov = 1;
            }
            if (prov == 1) //иначе мы сравниваем данные условия с переменной заданной в else в
предыдущей функции "иначе - число полиндром"
                label3.Text = "Число " + num1 + " явл. палиндромом.";
            for (int i = 0; i < num2.Length / 2; i++)
            {
                if (num2.Substring(i, 1) != (num2.Substring(num2.Length - 1 - i, 1))) //проверка на
отсутствие полиндрома для второго числа

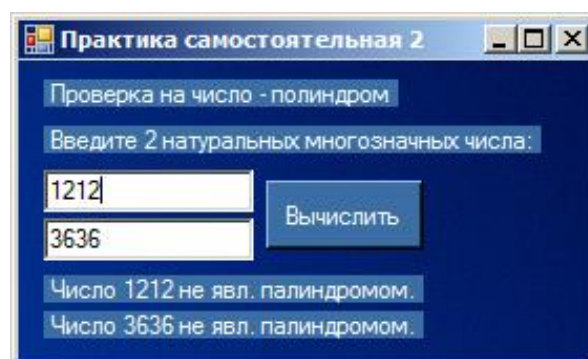
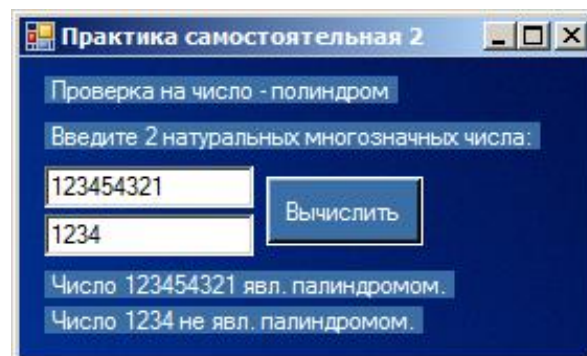
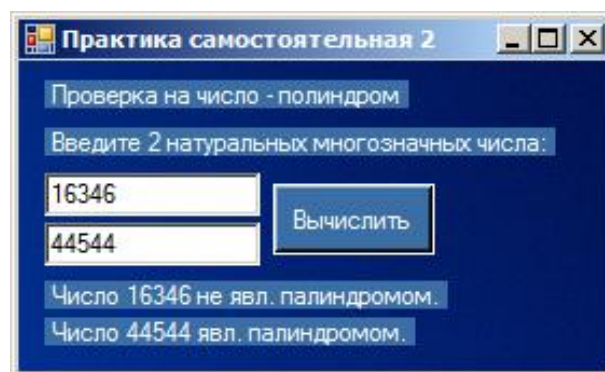
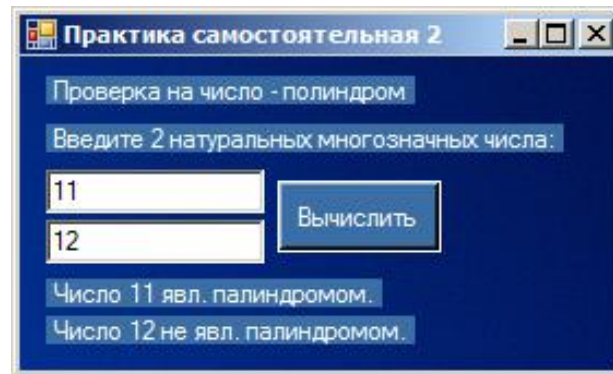
```



```

    {
        label4.Text = "Число " + num2 + " не явл. палиндромом."; //вывод сообщения об
отсутствии палиндрома
    }
    else
        prov2 = 1;
}
if (prov2 == 1) //иначе мы сравниваем данные условия с переменной заданой в else в
предудущей функции "иначе - число палиндром"
    label4.Text = "Число " + num2 + " явл. палиндромом.";
    textBox1.Clear(); //очистение полей ввода
    textBox2.Clear();
}
}
}

```



Тема 3: «Работа с графикой»

Задание 1 (5 баллов), задание 1 и 2 (10 баллов)

Задание 1:

Разработать программу движения фигуры при помощи таймера. Регулируя интервал работы таймера, можно управлять скоростью движения фигуры. Направление движения фигуры задается с помощью изменения координат по горизонтали и (или) по вертикали.

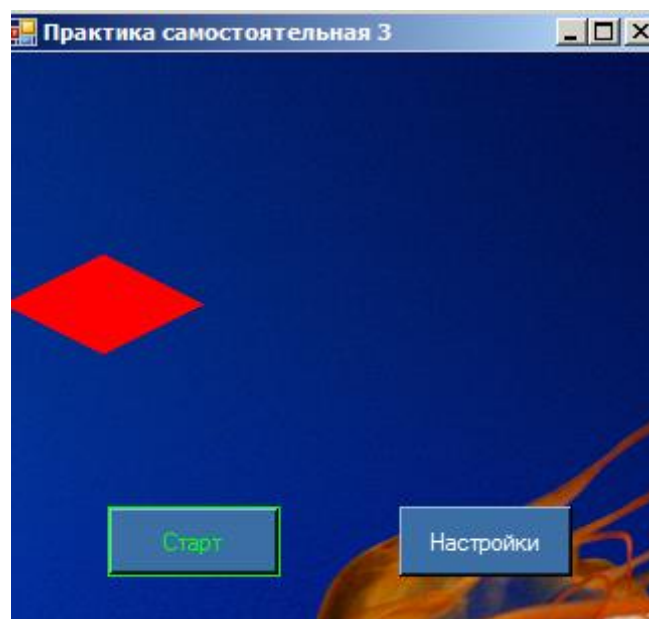
Задание 2:

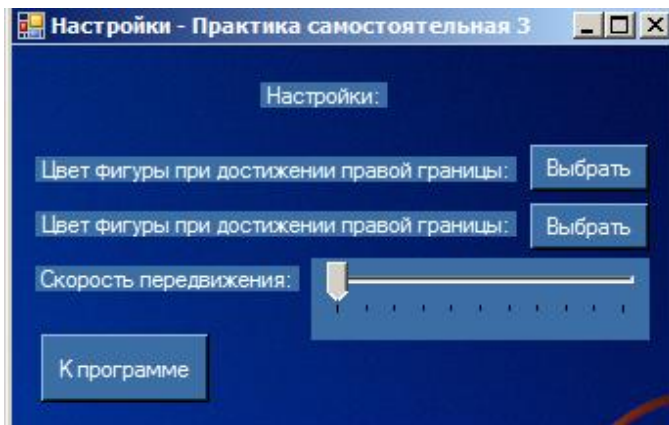
Учесть, что при изменении размеров формы фигура все равно должна двигаться в заданных направлениях в пределах формы.

Добавьте в проект настройку параметров: выбор цвета фигуры с помощью диалогового окна выбора цвета и настройку скорости движения фигуры. Настройка параметров должна происходить с помощью второй формы.

Вариант 26.

Движение ромба по горизонтали, при достижении границы формы ромб меняет цвет и начинает движение в обратном направлении. Окончание работы – нажатие любой клавиши.





```

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 Практика_самостоятельная_3
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        Rectangle rc; //задаем параметры прямоугольной области
        public int sec = 0;
        public int w = 100, h = 50;
        public int x = 0, y = 100;
        public int dx = 5;
        public enum STATUS { Left, Right }; //режимы движения
        public STATUS flag;
        public SolidBrush brush1 = new SolidBrush(Color.Red); // кисть смены 1
        public SolidBrush brush2 = new SolidBrush(Color.Green); // кисть смены 2
        SolidBrush curBrush = new SolidBrush(Color.Red); // кисть основная
        private void button1_Click(object sender, EventArgs e)
        {
            SolidBrush curBrush = new SolidBrush(brush1.Color);
            // кисть основная зависит от выбора цвета для кисти смены 1
            if (timer1.Enabled == false)
            //функции для смены вида кнопок старт/стоп при нажатии на них
            {
                timer1.Enabled = true;
                timer1.Start();
                sec = 0;
                button1.Text = "Стоп";
                button1.ForeColor = Color.Red;
            }
            else
            {
                timer1.Enabled = false;
                timer1.Stop();
                sec = 0;
                button1.Text = "Старт";
                button1.ForeColor = Color.Lime;
            }
        }
        private void button2_Click(object sender, EventArgs e)
        //становление 2ой формы с настройками
        {
            Form2 sett = new Form2();
            sett.Owner = this; //становление владельцем - форма 1
            sett.ShowDialog(); //вызов настроек
        }
        private void timer1_Tick(object sender, EventArgs e)
        {
            Form1 main = this.Owner as Form1; //отмечаем что владелец форма 1
            sec++; // секунды
            rc = new Rectangle(x, y, w, h); // размер прямоугольной области
            this.Invalidate(rc, true); // вызываем прорисовку этой области
            if (flag == STATUS.Left) // движение влево
            {
                x -= dx;
            }
            if (flag == STATUS.Right) // движение вправо
            {
                x += dx;
            }
            if (x >= (this.ClientSize.Width - w)) // если достигли правого края формы

```

```

        {
            flag = STATUS.Left; // меняем статус движения на левый и цвет кисти
            curBrush.Color = brush1.Color;
        }
        else
        {
            if (x <= 1) // если достигли левого края формы
            {
                flag = STATUS.Right; // меняем статус движения на правый и цвет кисти
                curBrush.Color = brush2.Color;
            }
            rc = new Rectangle(x, y, w, h); // новая прямоугольная область
            this.Invalidate(rc, true); // снова вызываем прорисовку этой области
        }
        private void Form1_Paint(object sender, PaintEventArgs e) // событие перерисовки формы
        {
            Graphics g = e.Graphics;
            g.FillPolygon(curBrush, new PointF[] { new PointF(x+50, y), new PointF(x + 100, y + 25), new
            PointF(x + 50, y + 50), new PointF(x, y+25) });
            // вызов многоугольника (ромба) на экран
        }
        private void Form1_KeyDown(object sender, KeyEventArgs e)
        {
            this.Close(); // при нажатии любой клавиши - программа закроется
        }
    }
}

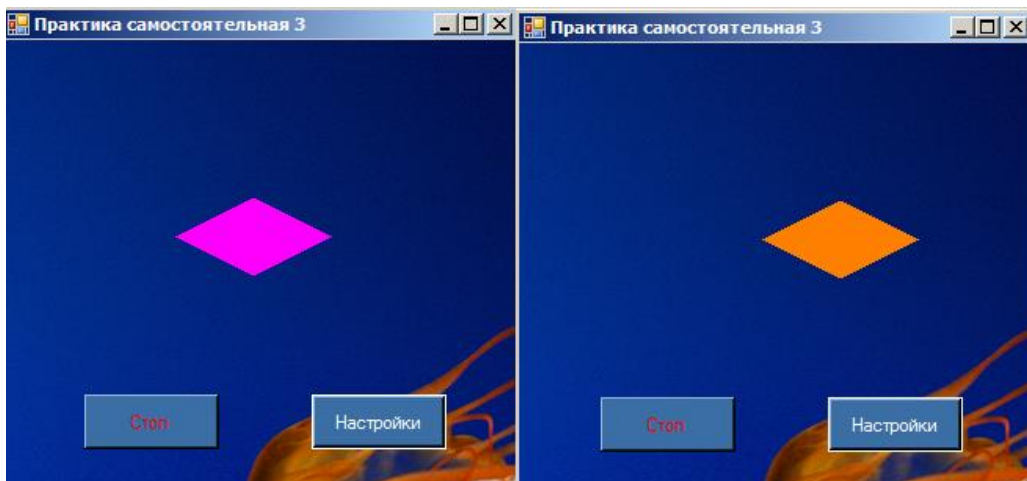
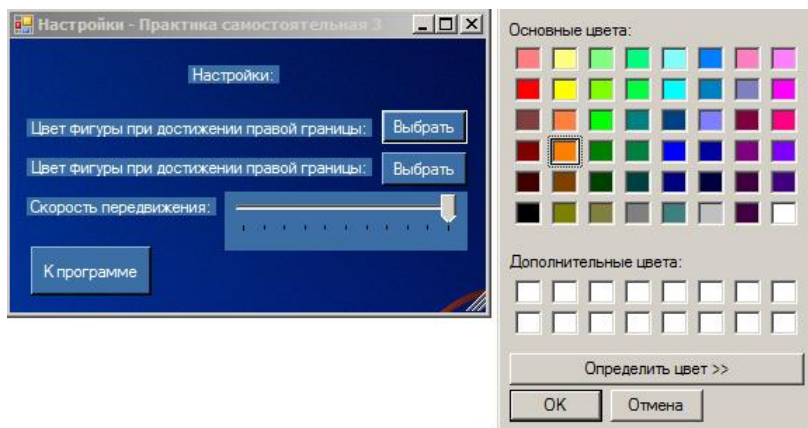
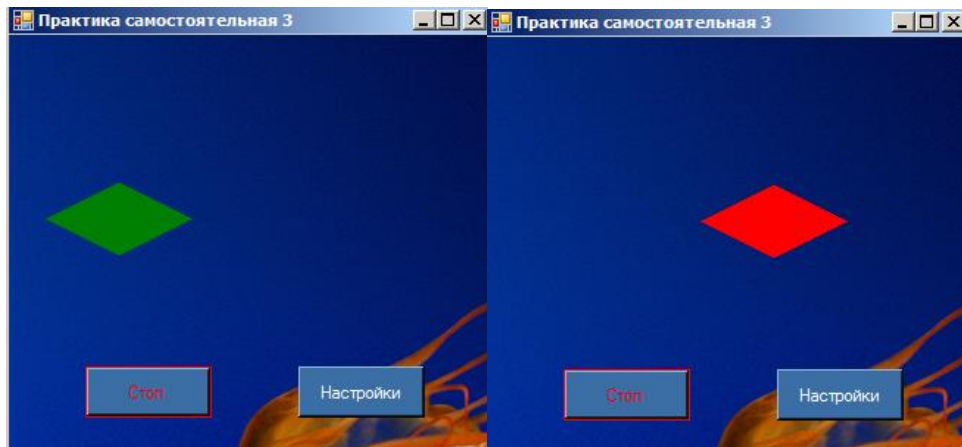
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
namespace Практика_самостоятельная_3
{
    public partial class Form2 : Form
    {
        public Form2()
        {
            InitializeComponent();
            Form1 main = this.Owner as Form1; //становление владельцем формы2 - форма1
        }
        private void button2_Click(object sender, EventArgs e)
        {
            this.Close(); //закрытие настроек
        }
        private void button1_Click(object sender, EventArgs e)
        {
            Form1 main = this.Owner as Form1;
            if (colorDialog1.ShowDialog() == DialogResult.OK)
            //открытие палитры и выбор цвета для 1кисти
                main.brush1.Color = colorDialog1.Color;
        }
        private void button3_Click(object sender, EventArgs e)
        {
            Form1 main = this.Owner as Form1;
            if (colorDialog1.ShowDialog() == DialogResult.OK)
            //открытие палитры и выбор цвета для 2кисти
                main.brush2.Color = colorDialog1.Color;
        }
        private void trackBar1_Scroll(object sender, EventArgs e)
        {
            Form1 main = this.Owner as Form1;
            //установка скоростей движения фигуры в зависимости от положения ползунка на трекбаре
            if (trackBar1.Value < 4)
                main.timer1.Interval = 300; //медленно
            if (trackBar1.Value >= 4 && trackBar1.Value < 7)
                main.timer1.Interval = 100; //средне
        }
    }
}

```

```

        if (trackBar1.Value > 7)
            main.timer1.Interval = 1; //быстро
    }
    private void Form2_KeyDown(object sender, KeyEventArgs e)
    {
        this.Close(); //при нажатии любой клавиши - программа вылетает
    }
}

```



Тема 4: «Обработка одномерных массивов»

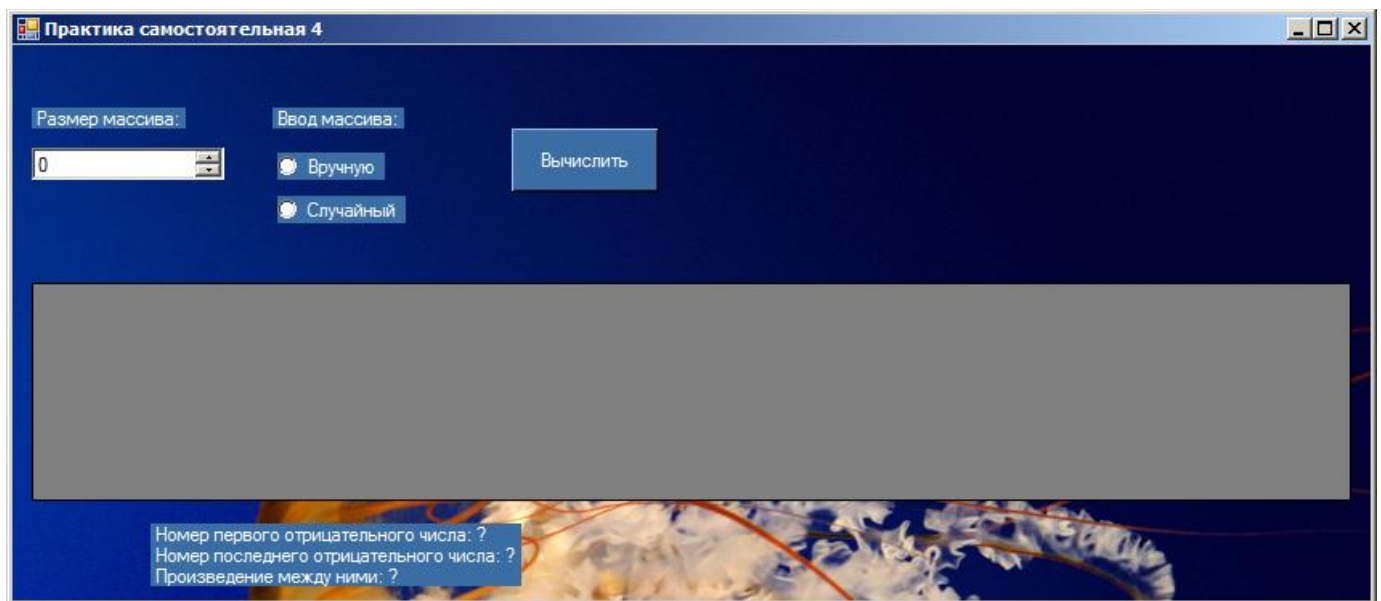
Цель работы: овладеть основными приемами работы с таблицами для эффективной обработки массивов

Указания:

Таблица представляет собой специальный компонент, позволяющие отображать данные в виде строк и столбцов. Компонент **dataGridView** позволяет хранить и отображать текстовую и графическую информацию. Но хранение и отображение данных выполняется программистом. Есть возможность подключить источник данных, которые будут представляться в таблице.

Вариант 6

Заполнить одномерный массив целыми числами. Вычислить номер первого отрицательного и последнего отрицательного элемента массива. Вычислить также произведение элементов, стоящих между ними. Если отрицательных нет или только один, то вывести сообщение об этом.



```

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 Практика_самостоятельная_4
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        int[] massiv; //задаем массив
        int n = 0;
        private void button1_Click(object sender, EventArgs e)
        {
            int[] massiv = new int[n]; //создание массива
            for (int i = 0; i < n; i++)
            {
                massiv[i] = Convert.ToInt32(dataGridView1.Rows[0].Cells[i].Value); //конвертация чисел
                в целые и вывод в нашу таблицу
            }
            int imin = -1; //нахождение первого отриц. элемента
            for (int i = 0; i < n; i++)
            {
                if (massiv[i] < 0)
                {
                    imin = i;
                    break;
                }
            }
            int imax = -1; //нахождение последнего отриц. элемента
            for (int i = 0; i < n; i++)
            {
                if (massiv[i] < 0)
                {
                    imax = i;
                }
            }
            label3.Text = "Номер первого отрицательного числа: " + imin + "\nНомер последнего
отрицательного числа: " + imax;
            if (imax - imin > 1) //счет произведения всех чисел между ними
            {
                int proz = 1;
                for (int i = imin + 1; i < imax; i++)
                {
                    proz *= massiv[i];
                }
                label3.Text += "\nПроизведение между ними: " + proz;
            }
            else //вывод сообщения в случае ошибки
            {
                MessageBox.Show("ОШИБКА:\nВозможные причины:\n1.Отрицательный элемент всего
один;\n2.Отрицательных элементов нет;\n3.Между 1ым отрицательным числом и 2ым нету других значений;");
            }
        }
        private void radioButton1_Click(object sender, EventArgs e) //создания свойства click для
радиобаттона, отвечающего за создание случайного массива
        {
            n = (int)this.numericUpDown1.Value;
            bool ronly = radioButton1.Checked; //создание bool отвечающего за ReadOnly
            Random rand2 = new Random();
            dataGridView1.ReadOnly = ronly;
            dataGridView1.ColumnCount = n;

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

```



```

{
    dataGridView1.Columns[i].Name = i.ToString();
    if (ronly)
    {
        dataGridView1.Rows[0].Cells[i].Value = rand2.Next(-100, 100).ToString();
//массив от -100 до 100
    }
    else
    {
        dataGridView1.Rows[0].Cells[i].Value = 0;
    }
}
}
}
}
}

```

Практика самостоятельная 4

Размер массива:

Ввод массива:

☐ Вручную ☐ Случайный

Вычислить

*	21	4	-99	7	-99
---	----	---	-----	---	-----

Номер первого отрицательного числа: 2
 Номер последнего отрицательного числа: 4
 Произведение между ними: 7

Практика самостоятельная 4

Размер массива:

Ввод массива:

☐ Вручную ☐ Случайный

Вычислить

*	49	68	67	52	-82
---	----	----	----	----	-----

Номер первого отрицательного числа: 4
 Номер последнего отрицательного числа: 4

ОШИБКА:

Возможные причины:

- 1.Отрицательный элемент всего один;
- 2.Отрицательных элементов нет;
- 3.Между 1ым отрицательным числом и 2ым нету других значений;

OK

