Практическая работа №9

На основе класса, реализованного в практической работа №8, внести изменения и реализовать переопределение стандартных операций сложения, умножения, деления и вычитания.

Продемонстрировать работу реализованных методов.

Заголовки переопределения операций представлены ниже.

метод сложения двух дробей

```
    static public Fraction operator + (Fraction ob1,
Fraction ob2) { . . . }
```

метод вычитания двух дробей

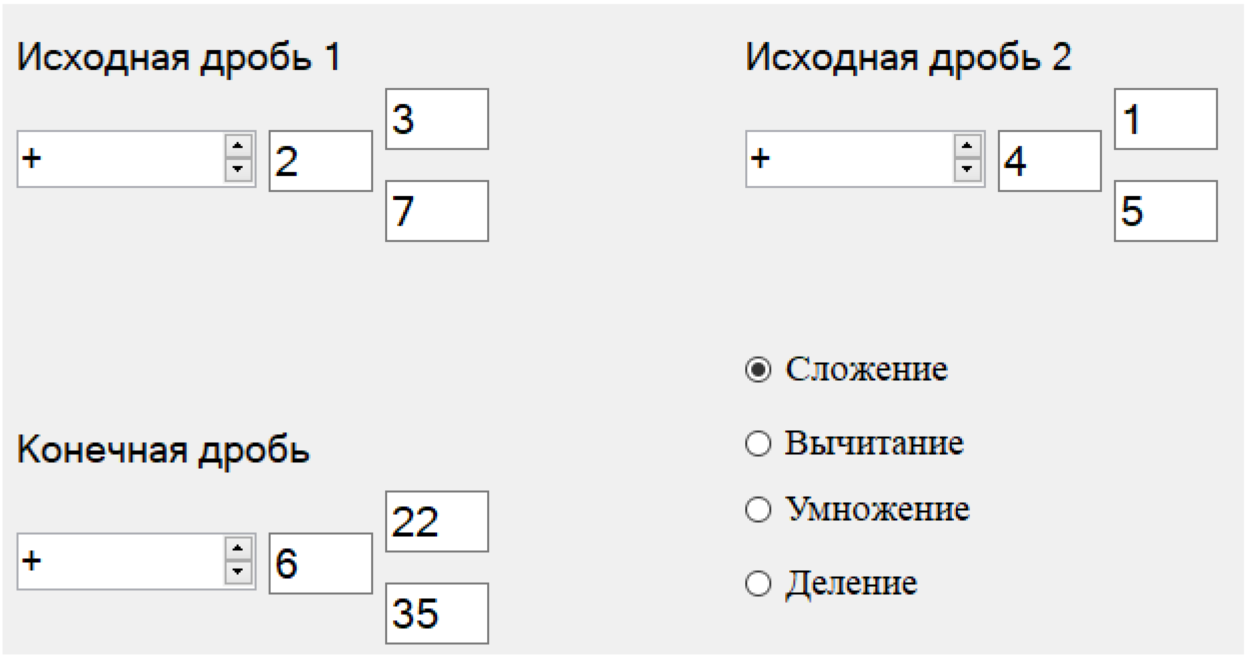
```
    static public Fraction operator - (Fraction ob1,
Fraction ob2) { . . . }
```

метод умножения двух дробей

```
    static public Fraction operator * (Fraction ob1,
Fraction ob2) { . . . }
```

метод деления двух дробей

```
    static public Fraction operator / (Fraction ob1,
Fraction ob2) { . . . }
```


Исходная дробь 1

+ 2 3/7

Исходная дробь 2

+ 4 1/5

[✓ Сложение]

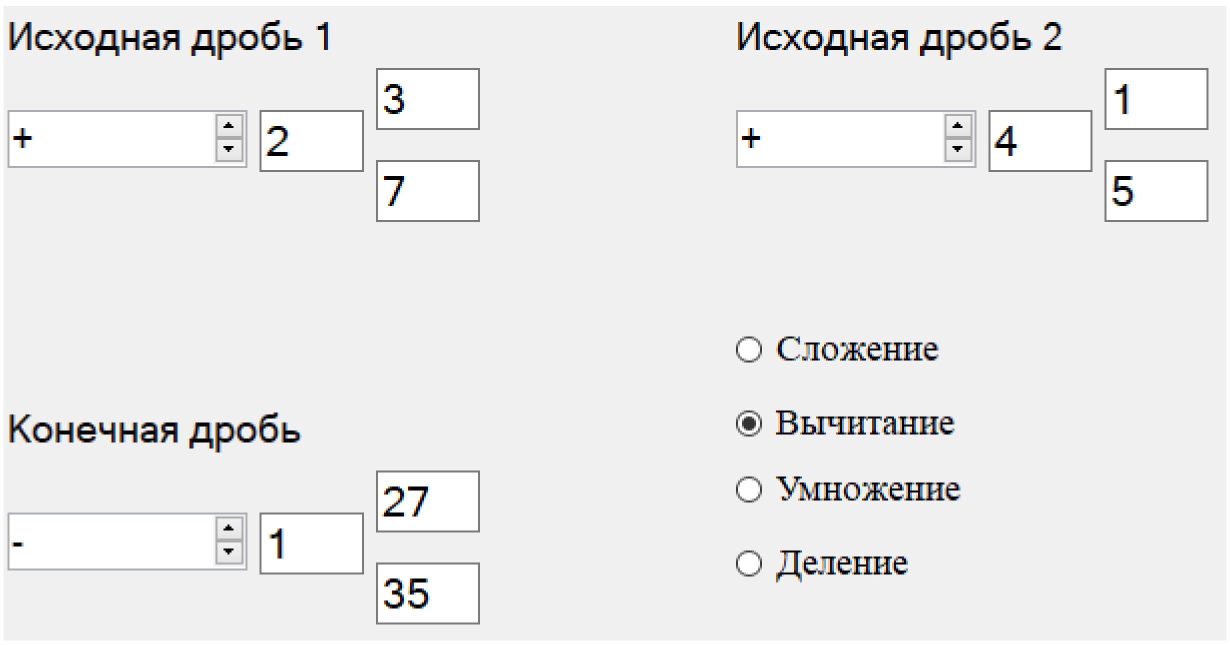
Вычитание

Умножение

Деление

Конечная дробь

+ 6 22/35



Исходная дробь 1

+ 2 $\frac{3}{7}$

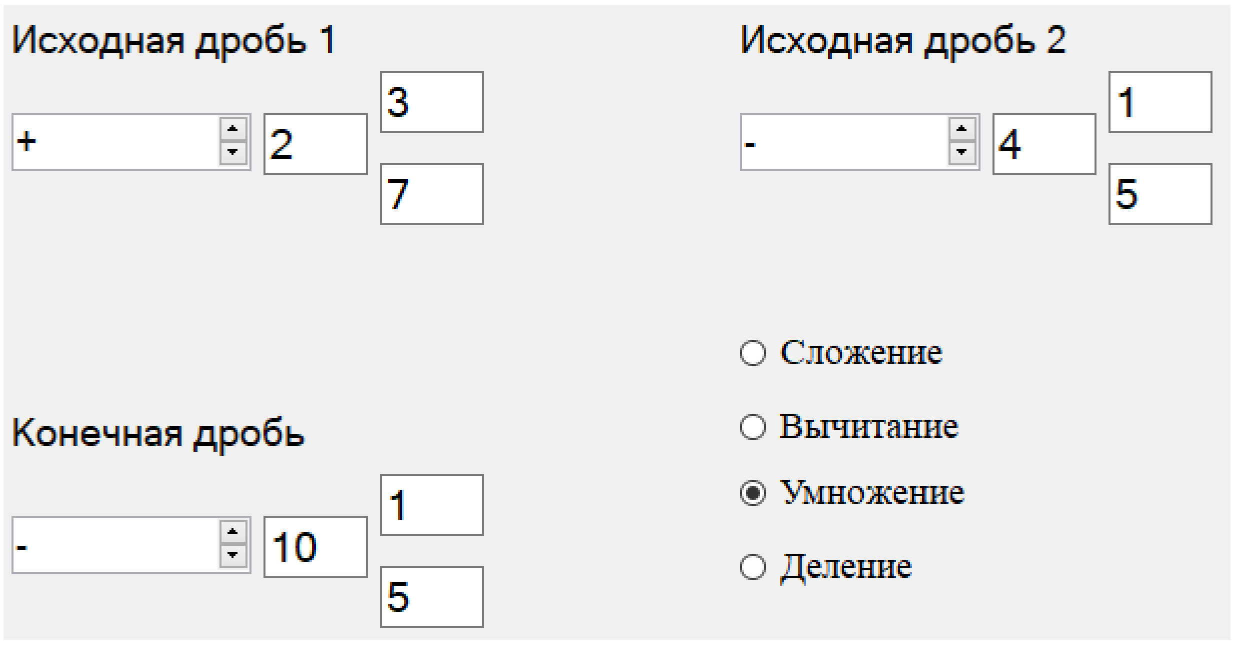
Исходная дробь 2

+ 4 $\frac{1}{5}$

Сложение [✓ Вычитание] Умножение Деление

Конечная дробь

- 1 $\frac{27}{35}$



Исходная дробь 1

+ 2 $\frac{3}{7}$

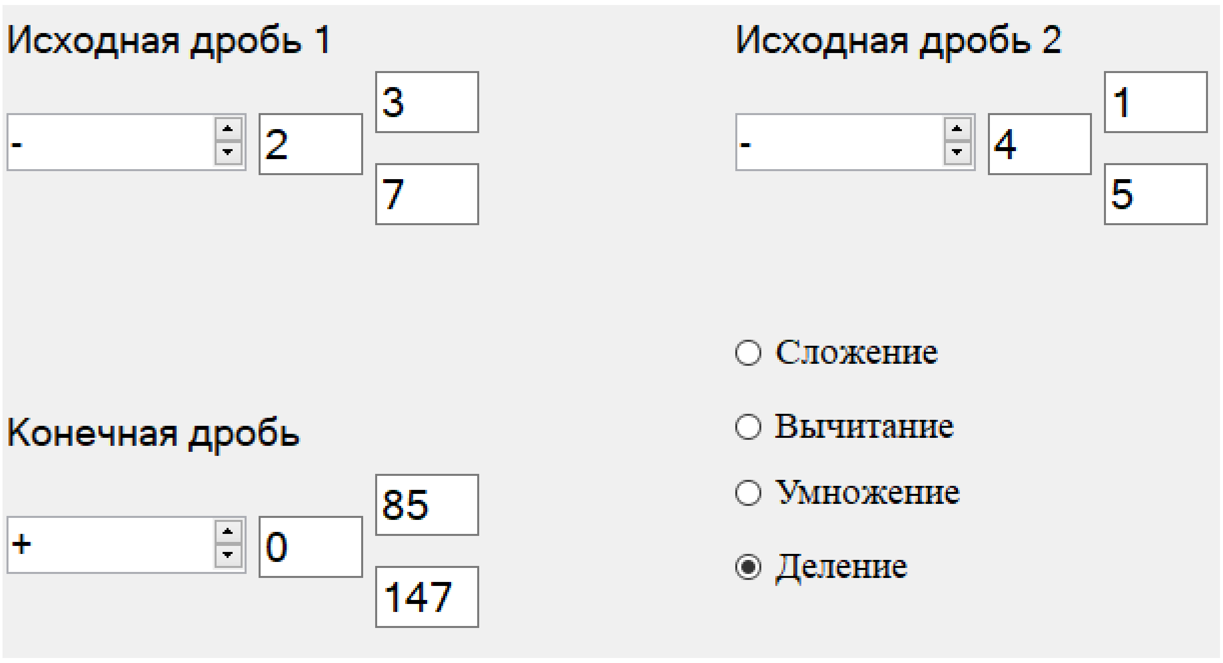
Исходная дробь 2

- 4 $\frac{1}{5}$

Сложение Вычитание [✓ Умножение] Деление

Конечная дробь

- 10 $\frac{1}{5}$


Исходная дробь 1

- 2 3/7

Исходная дробь 2

- 4 1/5

Конечная дробь

+ 0 85/147

○ Сложение

○ Вычитание

○ Умножение

[✓ Деление]

Fraction.cs

```
public class Fraction
    {
        public int sign, integer, numerator, denominator;

        public Fraction(int n_sign, int n_integer, int n_numerator, int
n_denominator)//конструктор класса
        {
            sign =n_sign;
            integer = n_integer;
            numerator = n_numerator;
            denominator = n_denominator;

        }
        public Fraction() {
            sign = 1;
            integer = 0;
            numerator = 0;
            denominator = 1;
        }

        private int NOD(int A, int B)
        {
            while (A != B)
            {
                if (A > B)
                {
                    A = A - B;
                }
                else
                {
                    B = B - A;
                }
            }
            return A;
        }
        public void Reduction()// сокращение дроби
        {
            if (numerator > 0)
            {
                int k = NOD(numerator, denominator);
                if (k != 1)
                {
                    numerator = numerator / k;
                    denominator = denominator / k;
                }
            }
        }

        public void Incorrect_fraction()//приведение в неправильную дробь
        {
            numerator = integer * denominator + numerator;
            integer = 0;
        }

        public void Correct_fraction()//создание правильной дроби и выделение целой части
        {
            int k = integer;
            integer = numerator / denominator;
            numerator = numerator - integer * denominator;
            integer = integer + k;
        }

        public void Addition(Fraction d)//сложение дробей
        {
            integer = sign* integer + d.sign*d.integer;
            int k1 = sign*numerator * d.denominator + d.sign*d.numerator * denominator;
            denominator = denominator * d.denominator;
            numerator = k1;
            if (integer < 0)
            {
                integer = integer * (-1);
                sign = sign * (-1);
            }
        }
        public void Subtraction(Fraction d)//Вычитание дробей
        {
            Incorrect_fraction();
            d.Incorrect_fraction();
            //integer = sign * integer - d.sign * d.integer;
            int k1 = (sign * numerator) * d.denominator - (d.sign * (d.numerator)) *
    denominator;
            numerator = k1;
            denominator = d.denominator * denominator;
            if (numerator < 0)
            {
                numerator = numerator * (-1);
                sign = sign * (-1);
            }
            Correct_fraction();
            Reduction();
        }


        public void Multiplication(Fraction d)
        {
            Incorrect_fraction();
            d.Incorrect_fraction();
            int t = sign * numerator * d.sign * d.numerator;
            int t2 = denominator * d.denominator;
            numerator = t;
            denominator = t2;
            if (numerator < 0)
            {
                numerator = numerator * (-1);
                sign = -1;
            }
            else
            {
                sign = 1;
            }

            Correct_fraction();
            Reduction();


        }
        public void Division(Fraction d)//Деление дробей
        {
            Incorrect_fraction();
            d.Incorrect_fraction();
            int t = sign * numerator * d.sign * d.denominator;
            int t2 = d.numerator * denominator;
            numerator = t;
            denominator = t2;
            if (numerator < 0)
            {
                numerator = numerator * (-1);
                sign = -1;
            }
            else
            {
                sign = 1;
            }
            Correct_fraction();
            Reduction();
        }
        //вычитание, умножение деление дописать

        public static Fraction operator +(Fraction b1, Fraction b2)
        {
            Fraction b = new Fraction();
            b1.Incorrect_fraction();
            b2.Incorrect_fraction();
            //integer = sign * integer - d.sign * d.integer;
            int k1 = (b1.sign * b1.numerator) * b2.denominator
                + (b2.sign * (b2.numerator)) * b1.denominator;
            b.numerator = k1;
            b.denominator = b1.denominator * b2.denominator;
            if (b.numerator < 0)
            {
                b.numerator = b.numerator * (-1);
                b.sign = b.sign * (-1);
            }
            b.Correct_fraction();
            b.Reduction();
            return b;
        }
        public static Fraction operator -(Fraction b1, Fraction b2)
        {
            Fraction b = new Fraction();
            b1.Incorrect_fraction();
            b2.Incorrect_fraction();
            //integer = sign * integer - d.sign * d.integer;
            int k1 = (b1.sign * b1.numerator) * b2.denominator - (b2.sign *
(b2.numerator)) * b1.denominator;
            b.numerator = k1;
            b.denominator = b1.denominator * b2.denominator;
            if (b.numerator < 0)
            {
                b.numerator = b.numerator * (-1);
                b.sign = b.sign * (-1);
            }
            b.Correct_fraction();
            b.Reduction();
            return b;
        }
        public static Fraction operator *(Fraction b1, Fraction b2)
        {
            Fraction b = new Fraction();
            b1.Incorrect_fraction();
            b2.Incorrect_fraction();
            int t = b1.sign * b1.numerator * b2.sign * b2.numerator;
            int t2 = b1.denominator * b2.denominator;
            b.numerator = t;
            b.denominator = t2;
            if (b.numerator < 0)
            {
                b.numerator = b.numerator * (-1);
                b.sign = -1;
            }
            else
            {
                b.sign = 1;
            }

            b.Correct_fraction();
            b.Reduction();
            return b;
        }
        public static Fraction operator /(Fraction b1, Fraction b2)
        {
            Fraction b = new Fraction();
            b1.Incorrect_fraction();
            b2.Incorrect_fraction();
            int t = b1.sign * b1.numerator * b2.sign * b2.denominator;
            int t2 = b2.numerator*b1.denominator;
            b.numerator = t;
            b.denominator = t2;
            if (b.numerator < 0)
            {
                b.numerator = b.numerator * (-1);
                b.sign = -1;
            }
            else
            {
                b.sign = 1;
            }

            b.Correct_fraction();
            b.Reduction();
            return b;
        }
    }
```

Form1.cs

```
public partial class Form1 : Form
    {

        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
```

```
{
    domainUpDown1.SelectedIndex = 0;
    domainUpDown2.SelectedIndex = 0;
    domainUpDown3.SelectedIndex = 0;
}


public void Print(Fraction f)
{

    if (f.sign > 0)
    {
        domainUpDown2.SelectedIndex = 0;
    }
    else
    {
        domainUpDown2.SelectedIndex = 1;
    }
    textBox6.Text = f.integer.ToString();
    textBox5.Text = f.numerator.ToString();
    textBox4.Text = f.denominator.ToString();
}

public Fraction ReceiveF()
{
    int sign = 1;
    if (domainUpDown1.SelectedIndex == 1)
    {
        sign = -1;
    }
    else if (domainUpDown1.SelectedIndex == 0)
    {
        sign = 1;
    }

    int integer = Convert.ToInt32(textBox2.Text);
    int numerator = Convert.ToInt32(textBox1.Text);
    int denominator = Convert.ToInt32(textBox3.Text);

    Fraction newf = new Fraction(sign, integer, numerator, denominator);
    return newf;
}

public Fraction ReceiveF2()
{
    int sign2 = 1;
    if (domainUpDown3.SelectedIndex == 1)
    {
        sign2 = -1;
    }
    else if (domainUpDown3.SelectedIndex == 0)
    {
        sign2 = 1;
    }

    int integer2 = Convert.ToInt32(textBox9.Text);
    int numerator2 = Convert.ToInt32(textBox8.Text);
    int denominator2 = Convert.ToInt32(textBox7.Text);

    Fraction newf2 = new Fraction(sign2, integer2, numerator2, denominator2);
    return newf2;
}
```

```
private void button1_Click(object sender, EventArgs e)
{
    Fraction f = ReceiveF();

    f.Reduction();
    Print(f);
}

private void button2_Click(object sender, EventArgs e)
{
    Fraction f = ReceiveF();
    ReceiveF();
    f.Incorrect_fraction();
    Print(f);
}

private void button3_Click(object sender, EventArgs e)
{
    Fraction f = ReceiveF();

    f.Correct_fraction();
    Print(f);
}

private void button5_Click(object sender, EventArgs e)
{
    Fraction f = ReceiveF();
    Fraction f2 = ReceiveF2();
    f.Subtraction(f2);
    Print(f2);
}

private void button6_Click(object sender, EventArgs e)
{
    Fraction f = ReceiveF();
    Fraction f2=ReceiveF2();
    f.Multiplication(f2);
    Print(f2);
}

private void button7_Click(object sender, EventArgs e)
{


        Fraction f1 = ReceiveF();
        Fraction f2 = ReceiveF2();
        Fraction f3 = f1 + f2;
        //Fraction f4 = f1 * f2;
        // Fraction f5 = f1 - f2;
        //Fraction f6 = f1 / f2;
        Print(f3);


}

private void button4_Click(object sender, EventArgs e)
{
    Fraction f = ReceiveF();
    Fraction f2 = ReceiveF2();
    f.Addition(f2);
    Print(f2);
}

private void button8_Click(object sender, EventArgs e)
{
```

```
            Fraction f1 = ReceiveF();
            Fraction f2 = ReceiveF2();
            Fraction f5 = f1 - f2;
            Print(f5);
        }

        private void button9_Click(object sender, EventArgs e)
        {
            Fraction f1 = ReceiveF();
            Fraction f2 = ReceiveF2();
            Fraction f4 = f1 * f2;
            Print(f4);
        }

        private void button10_Click(object sender, EventArgs e)
        {
            Fraction f1 = ReceiveF();
            Fraction f2 = ReceiveF2();
            Fraction f6 = f1 / f2;
            Print(f6);
        }

        private void button11_Click(object sender, EventArgs e)
        {


            Fraction f = ReceiveF();
            Fraction f2 = ReceiveF2();
            f.Division(f2);
            Print(f2);

        }

        private void radioButton1_CheckedChanged(object sender, EventArgs e)
        {
            if (radioButton1.Checked)
            {
                Fraction f1 = ReceiveF();
                Fraction f2 = ReceiveF2();
                Fraction f3 = f1 + f2;
                //Fraction f4 = f1 * f2;
                // Fraction f5 = f1 - f2;
                //Fraction f6 = f1 / f2;
                Print(f3);
            }
        }

        private void radioButton2_CheckedChanged(object sender, EventArgs e)
        {
            if (radioButton2.Checked)
            {
                Fraction f1 = ReceiveF();
                Fraction f2 = ReceiveF2();
                Fraction f5 = f1 - f2;
                Print(f5);
            }
        }

        private void radioButton3_CheckedChanged(object sender, EventArgs e)
        {
            if (radioButton3.Checked)
            {
                Fraction f1 = ReceiveF();
                Fraction f2 = ReceiveF2();
                Fraction f4 = f1 * f2;
                Print(f4);
            }
        }

        private void radioButton4_CheckedChanged(object sender, EventArgs e)
        {
            if (radioButton4.Checked)
            {
                Fraction f1 = ReceiveF();
                Fraction f2 = ReceiveF2();
                Fraction f6 = f1 / f2;
                Print(f6);
            }
        }
    }
```

Ссылка на гитхаб:

https://github.com/Alexandrov911/Practical-9.2022.git