Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**

**ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

|  |
| --- |
| Институт информационных технологий и анализа данных |
| наименование института |

**ОТЧЁТ**

К лабораторной работе №3 по дисциплине:

|  |
| --- |
| Технология программирования |
| Свой тип |

наименование темы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Выполнил студент группы: | ИСТб-20 |  |  |  |  |
|  | шифр группы |  | подпись |  | Фамилия И.О. |
| Проверил: |  |  |  |  | З.А. Бахвалова |
|  | должность |  | подпись |  | Фамилия И.О. |

Иркутск 2021 г.

**Оглавление**

[Свой тип 1](#_Toc85548417)

[Постановка задачи 3](#_Toc85548418)

[Таблица классов 3](#_Toc85548419)

[Таблица методов 3](#_Toc85548420)

[Таблица спецификаций 4](#_Toc85548421)

[Таблица тестов 5](#_Toc85548422)

[Код программы 5](#_Toc85548423)

[Результаты тестов 5](#_Toc85548424)

### Постановка задачи

**Задача №6.**

Число, заданное в системе счисления (двоичная, восьмеричная, десятичная шестнадцатеричная)

* сложение
* вычитание
* умножение
* сравнение
* вывод значения в любой системе счисления

### Таблица спецификаций

### Класс Number

Таблица 1 – Таблица спецификация класса Number

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Назначение | Тип | Диапазон |
| class Number – класс числа | | | |
| \_value | Тип кино | private readonly string | — |
| \_system | Наименование | private readonly byte | [2,8,10,16] |

Таблица 2 – Таблица методов класса Number

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Назначение | Тип | Параметры |
| operator + | Перегрузка оператора сложения | public static Number | Number first, Number second |
| operator - | Перегрузка оператора вычитания | public static Number | Number first, Number second |
| operator \* | Перегрузка оператора умножения | public static Number | Number first, Number second |
| operator / | Перегрузка оператора деления | public static Number | Number first, Number second |
| operator > | Перегрузка оператора больше | public static Number | Number first, Number second |
| operator < | Перегрузка оператора меньше | public static Number | Number first, Number second |
| operator >= | Перегрузка оператора больше или равно | public static Number | Number first, Number second |
| operator <= | Перегрузка оператора меньше или равно | public static Number | Number first, Number second |
| operator == | Перегрузка оператора равенства | public static Number | Number first, Number second |
| operator != | Перегрузка оператора не равенства | public static Number | Number first, Number second |
| ConvertFromDecimal | Перевод из десятичной системы | public string | byte system |

Таблица 3 - Таблица спецификаций методов класса Number

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Имя | Назначение | | | Тип | | | Диапазон | | |
| operator + | | | | | | | | | |
| newFirst | Первое число в десятичной системе | | | ulong | | | от 0 до 18 446 744 073 709 551 615. | | |
| newSecond | Второе число в десятичной системе | | | ulong | | | от 0 до 18 446 744 073 709 551 615. | | |
| operator - | | | | | | | | | |
| newFirst | | Первое число в десятичной системе | | | ulong | | | от 0 до 18 446 744 073 709 551 615. | |
| newSecond | | Второе число в десятичной системе | | | ulong | | | от 0 до 18 446 744 073 709 551 615. | |
| operator \* | | | | | | | | | |
| newFirst | | Первое число в десятичной системе | | | ulong | | | от 0 до 18 446 744 073 709 551 615. | |
| newSecond | | Второе число в десятичной системе | | | ulong | | | от 0 до 18 446 744 073 709 551 615. | |
| operator / | | | | | | | | | |
| newFirst | | Первое число в десятичной системе | | | ulong | | | от 0 до 18 446 744 073 709 551 615. | |
| newSecond | | Второе число в десятичной системе | | | ulong | | | от 0 до 18 446 744 073 709 551 615. | |
| operator > | | | | | | | | | |
| newFirst | | Первое число в десятичной системе | | | ulong | | | от 0 до 18 446 744 073 709 551 615. | |
| newSecond | | Второе число в десятичной системе | | | ulong | | | от 0 до 18 446 744 073 709 551 615. | |
| operator < | | | | | | | | | |
| newFirst | | Первое число в десятичной системе | | | ulong | | | от 0 до 18 446 744 073 709 551 615. | |
| newSecond | | Второе число в десятичной системе | | | ulong | | | от 0 до 18 446 744 073 709 551 615. | |
| operator >= | | | | | | | | | |
| newFirst | | Первое число в десятичной системе | | | ulong | | | от 0 до 18 446 744 073 709 551 615. | |
| newSecond | | Второе число в десятичной системе | | | ulong | | | от 0 до 18 446 744 073 709 551 615. | |
| operator <= | | | | | | | | | |
| newFirst | | Первое число в десятичной системе | | | ulong | | | от 0 до 18 446 744 073 709 551 615. | |
| newSecond | | Второе число в десятичной системе | | | ulong | | | от 0 до 18 446 744 073 709 551 615. | |
| operator == | | | | | | | | | |
| newFirst | | Первое число в десятичной системе | | | ulong | | | от 0 до 18 446 744 073 709 551 615. | |
| newSecond | | Второе число в десятичной системе | | | ulong | | | от 0 до 18 446 744 073 709 551 615. | |
| operator != | | | | | | | | | |
| newFirst | | Первое число в десятичной системе | | | ulong | | | от 0 до 18 446 744 073 709 551 615. | |
| newSecond | | Второе число в десятичной системе | | | ulong | | | от 0 до 18 446 744 073 709 551 615. | |
| ConvertFromDecimal | | | | | | | | | |
| value | | | Число в десятичной системе | | | ulong | | | от 0 до 18 446 744 073 709 551 615. |

### Класс Number

Таблица 4 – Таблица спецификация класса Number

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Назначение | Тип | Диапазон |
| class MainForm – класс формы | | | |
| numberMathFirst | Первое число для математический операций | Number | — |
| numberMathSecond | Второе число для математический операций | Number | — |
| numberComparisonFirst | Первое число для операций сравнения | Number | — |
| numberComparisonSecond | Второе число для операций сравнения | Number | — |

Таблица 5 – Таблица методов класса Number

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Назначение | Тип | Параметры |
| MainForm\_Load | Событие загрузки формы | private void | object sender, EventArgs e |
| ComboBoxFirstNumber  \_SelectedIndexChanged | Событие изменения comboBox первого числа для математический операций | private void | object sender, EventArgs e |
| ComboBoxSecondNumber  \_SelectedIndexChanged | Событие изменения comboBox второго числа для математический операций | private void | object sender, EventArgs e |
| MathChanged | Событие изменения comboBox математических операций | private void | object sender, EventArgs e |
| ComboBoxComparisonFirstValue  \_SelectedValueChanged | Событие изменения comboBox первого числа для операций сравнения | private void | object sender, EventArgs e |
| ComboBoxComparisonSecondValue  \_SelectedValueChanged | Событие изменения comboBox второго числа для операций сравнения | private void | object sender, EventArgs e |
| ComparisonChanged | Событие изменения comboBox операций сравнения | private void | object sender, EventArgs e |
| NumberChanged | События изменения TextBox и преобразование числа из десятичной системы в другие | private void | object sender, EventArgs e |
| TextBoxFirstNumber  \_KeyPress | Событие нажатия на кнопку в TextBox первого числа для математических операций | private void | object sender, KeyPressEventArgs e |
| TextBoxSecondNumber  \_KeyPress | Событие нажатия на кнопку в TextBox второго числа для математических операций | private void | object sender, KeyPressEventArgs e |
| TextBoxNumber  \_KeyPress | Событие нажатия на кнопку в TextBox числа для преобразования в другие системы счисления | private void | object sender, KeyPressEventArgs e |
| TextBoxComparisonFirstValue  \_KeyPress | Событие нажатия на кнопку в TextBox первого числа для операций сравнения | private void | object sender, KeyPressEventArgs e |
| TextBoxComparisonSecondValue  \_KeyPress | Событие нажатия на второго в TextBox первого числа для операций сравнения | private void | object sender, KeyPressEventArgs e |
| CheckInput | Проверка символа на соответствие вводу | private static bool | string system, char KeyChar |

Таблица 6 - Таблица спецификаций методов класса Number

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Имя | | Назначение | Тип | Диапазон | |
| ComboBoxFirstNumber\_SelectedIndexChanged | | | | | |
| number | | Значение числа | ulong | от 0 до 18 446 744 073 709 551 615. | |
| system | | Система счисления | ulong | от 0 до 18 446 744 073 709 551 615. | |
| ComboBoxSecondNumber\_SelectedIndexChanged | | | | | |
| number | Значение числа | | ulong | | от 0 до 18 446 744 073 709 551 615. |
| system | Система счисления | | ulong | | от 0 до 18 446 744 073 709 551 615. |
| ComboBoxComparisonFirstValue\_SelectedValueChanged | | | | | |
| number | Значение числа | | ulong | | от 0 до 18 446 744 073 709 551 615. |
| system | Система счисления | | ulong | | от 0 до 18 446 744 073 709 551 615. |
| ComboBoxComparisonSecondValue\_SelectedValueChanged | | | | | |
| number | Значение числа | | ulong | | от 0 до 18 446 744 073 709 551 615. |
| system | Система счисления | | ulong | | от 0 до 18 446 744 073 709 551 615. |
| NumberChanged | | | | | |
| number | | Число в десятичной системе | Number | | — |

### Таблица тестов

Таблица 7 - Таблица тестов

| Номер теста | Цель теста | Входные данные | Выходные данные |
| --- | --- | --- | --- |
| 1 | Тестирование оператора сложения | "1011", 2  "7261", 8 | 3772 |
| 2 | Тестирование оператора вычитания | "1101010", 2  "10110", 2 | 84 |
| 3 | Тестирование оператора умножения | "0011", 2  "11", 16 | 51 |
| 4 | Тестирование оператора сложения | "78", 16  "1001", 2 | 13,333333 |

### Код программы

Код класса Number

using System;

namespace Number\_Systems

{

public class Number

{

private readonly string \_value;

private readonly byte \_system;

public Number()

{

\_value = "0";

\_system = 10;

}

public Number(float value)

{

\_value = value.ToString();

\_system = 10;

}

public Number(string value, byte system)

{

\_value = value;

\_system = system switch

{

2 => 2,

8 => 8,

10 => 10,

16 => 16,

\_ => 10,

};

}

public string Value => \_value;

public byte System => \_system;

public static Number operator +(Number first, Number second)

{

ulong newFirst = Convert.ToUInt64(first.Value, first.System);

ulong newSecond = Convert.ToUInt64(second.Value, second.System);

return new Number((float)newFirst + newSecond);

}

public static Number operator -(Number first, Number second)

{

ulong newFirst = Convert.ToUInt64(first.Value, first.System);

ulong newSecond = Convert.ToUInt64(second.Value, second.System);

return new Number((float)newFirst - newSecond);

}

public static Number operator \*(Number first, Number second)

{

ulong newFirst = Convert.ToUInt64(first.Value, first.System);

ulong newSecond = Convert.ToUInt64(second.Value, second.System);

return new Number((float)newFirst \* newSecond);

}

public static Number operator /(Number first, Number second)

{

ulong newFirst = Convert.ToUInt64(first.Value, first.System);

ulong newSecond = Convert.ToUInt64(second.Value, second.System);

return new Number((float)newFirst / newSecond);

}

public static bool operator ==(Number first, Number second)

{

ulong newFirst = Convert.ToUInt64(first.Value, first.System);

ulong newSecond = Convert.ToUInt64(second.Value, second.System);

return newFirst == newSecond;

}

public static bool operator !=(Number first, Number second)

{

ulong newFirst = Convert.ToUInt64(first.Value, first.System);

ulong newSecond = Convert.ToUInt64(second.Value, second.System);

return newFirst != newSecond;

}

public static bool operator >(Number first, Number second)

{

ulong newFirst = Convert.ToUInt64(first.Value, first.System);

ulong newSecond = Convert.ToUInt64(second.Value, second.System);

return newFirst > newSecond;

}

public static bool operator <(Number first, Number second)

{

ulong newFirst = Convert.ToUInt64(first.Value, first.System);

ulong newSecond = Convert.ToUInt64(second.Value, second.System);

return newFirst < newSecond;

}

public static bool operator >=(Number first, Number second)

{

ulong newFirst = Convert.ToUInt64(first.Value, first.System);

ulong newSecond = Convert.ToUInt64(second.Value, second.System);

return newFirst >= newSecond;

}

public static bool operator <=(Number first, Number second)

{

ulong newFirst = Convert.ToUInt64(first.Value, first.System);

ulong newSecond = Convert.ToUInt64(second.Value, second.System);

return newFirst <= newSecond;

}

public string ConvertFromDecimal(byte system)

{

long value = long.Parse(Value);

return Convert.ToString(value, system).ToUpper();

}

}

}

Код формы MainForm

using System;

using System.Windows.Forms;

namespace Number\_Systems

{

public partial class MainForm : Form

{

private Number numberMathFirst = new(0);

private Number numberMathSecond = new (0);

private Number numberComparisonFirst = new(0);

private Number numberComparisonSecond = new(0);

public MainForm()

{

InitializeComponent();

}

//стандартные значния comboBox при запуске

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

{

comboBoxFirstNumber.SelectedItem = "10";

comboBoxSecondNumber.SelectedItem = "10";

comboBoxComparisonFirstValue.SelectedItem = "10";

comboBoxComparisonSecondValue.SelectedItem = "10";

}

//расчёт математических операций

private void ComboBoxFirstNumber\_SelectedIndexChanged(object sender, EventArgs e)

{

long number = Convert.ToInt64(numberMathFirst.Value, numberMathFirst.System);

byte system = byte.Parse(comboBoxFirstNumber.Text);

numberMathFirst = new Number(Convert.ToString(number, system).ToUpper(), system);

textBoxFirstNumber.Text = numberMathFirst.Value;

}

private void ComboBoxSecondNumber\_SelectedIndexChanged(object sender, EventArgs e)

{

long number = Convert.ToInt64(numberMathSecond.Value, numberMathSecond.System);

byte system = byte.Parse(comboBoxSecondNumber.Text);

numberMathSecond = new Number(Convert.ToString(number, system).ToUpper(), system);

textBoxSecondNumber.Text = numberMathSecond.Value;

}

private void MathChanged(object sender, EventArgs e)

{

if (!String.IsNullOrEmpty(textBoxFirstNumber.Text) && !String.IsNullOrEmpty(textBoxSecondNumber.Text))

{

numberMathFirst = new(textBoxFirstNumber.Text, byte.Parse(comboBoxFirstNumber.Text));

numberMathSecond = new(textBoxSecondNumber.Text, byte.Parse(comboBoxSecondNumber.Text));

LabelMathAnswer.Text = comboBoxMathOperation.Text switch

{

"+" => (numberMathFirst + numberMathSecond).Value,

"-" => (numberMathFirst - numberMathSecond).Value,

"\*" => (numberMathFirst \* numberMathSecond).Value,

"/" when comboBoxSecondNumber.SelectedItem.ToString() != "0" => (numberMathFirst / numberMathSecond).Value,

\_ => "",

};

}

else

{

LabelMathAnswer.Text = "";

}

}

//сравнение двух чисел

private void ComboBoxComparisonFirstValue\_SelectedValueChanged(object sender, EventArgs e)

{

long number = Convert.ToInt64(numberComparisonFirst.Value, numberComparisonFirst.System);

byte system = byte.Parse(comboBoxComparisonFirstValue.Text);

numberComparisonFirst = new Number(Convert.ToString(number, system).ToUpper(), system);

textBoxComparisonFirstValue.Text = numberComparisonFirst.Value;

}

private void ComboBoxComparisonSecondValue\_SelectedValueChanged(object sender, EventArgs e)

{

long number = Convert.ToInt64(numberComparisonSecond.Value, numberComparisonSecond.System);

byte system = byte.Parse(comboBoxComparisonSecondValue.Text);

numberComparisonFirst = new Number(Convert.ToString(number, system).ToUpper(), system);

textBoxComparisonSecondValue.Text = numberComparisonSecond.Value;

}

private void ComparisonChanged(object sender, EventArgs e)

{

if (!String.IsNullOrEmpty(textBoxComparisonFirstValue.Text) && !String.IsNullOrEmpty(textBoxComparisonSecondValue.Text))

{

numberComparisonFirst = new(textBoxComparisonFirstValue.Text, byte.Parse(comboBoxComparisonFirstValue.Text));

numberComparisonSecond = new(textBoxComparisonSecondValue.Text, byte.Parse(comboBoxComparisonSecondValue.Text));

labelComparisonAnswer.Text = comboBoxComparison.Text switch

{

"=" => numberComparisonFirst == numberComparisonSecond ? $"{numberComparisonFirst.Value} равно {numberComparisonSecond.Value}" : $"{numberComparisonFirst.Value} не равно {numberComparisonSecond.Value}",

"≠" => numberComparisonFirst != numberComparisonSecond ? $"{numberComparisonFirst.Value} не равно {numberComparisonSecond.Value}" : $"{numberComparisonFirst.Value} равно {numberComparisonSecond.Value}",

"<" => numberComparisonFirst < numberComparisonSecond ? $"{numberComparisonFirst.Value} меньше {numberComparisonSecond.Value}" : $"{numberComparisonFirst.Value} больше или равно {numberComparisonSecond.Value}",

"≤" => numberComparisonFirst <= numberComparisonSecond ? $"{numberComparisonFirst.Value} меньше или равно {numberComparisonSecond.Value}" : $"{numberComparisonFirst.Value} больше {numberComparisonSecond.Value}",

">" => numberComparisonFirst > numberComparisonSecond ? $"{numberComparisonFirst.Value} больше {numberComparisonSecond.Value}" : $"{numberComparisonFirst.Value} меньше или равно {numberComparisonSecond.Value}",

"≥" => numberComparisonFirst >= numberComparisonSecond ? $"{numberComparisonFirst.Value} больше или равно {numberComparisonSecond.Value}" : $"{numberComparisonFirst.Value} Меньше {numberComparisonSecond.Value}",

\_ => "",

};

}

else

{

labelComparisonAnswer.Text = "";

}

}

//преобразование в другие системы счисления

private void NumberChanged(object sender, EventArgs e)

{

if (!String.IsNullOrEmpty(textBoxNumber.Text))

{

Number number = new(textBoxNumber.Text, 10);

labelBinary.Text = $"Двоичная {number.ConvertFromDecimal(2)}";

labelОctal.Text = $"Восмеричная {number.ConvertFromDecimal(8)}";

labeHexadecimal.Text = $"Шестнадцатеричная {number.ConvertFromDecimal(16)}";

}

else

{

labelBinary.Text = "";

labelОctal.Text = "";

labeHexadecimal.Text = "";

}

}

//проверки ввода

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

{

e.Handled = !CheckInput(comboBoxFirstNumber.Text, e.KeyChar);

numberMathFirst = new(textBoxFirstNumber.Text, byte.Parse(comboBoxFirstNumber.Text));

}

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

{

e.Handled = !CheckInput(comboBoxSecondNumber.Text, e.KeyChar);

numberMathSecond = new(textBoxSecondNumber.Text, byte.Parse(comboBoxSecondNumber.Text));

}

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

{

e.Handled = !CheckInput("10", e.KeyChar);

}

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

{

e.Handled = !CheckInput(comboBoxComparisonFirstValue.Text, e.KeyChar);

numberComparisonFirst = new(textBoxComparisonFirstValue.Text, byte.Parse(comboBoxComparisonFirstValue.Text));

}

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

{

e.Handled = !CheckInput(comboBoxComparisonSecondValue.Text, e.KeyChar);

numberComparisonSecond = new(textBoxComparisonSecondValue.Text, byte.Parse(comboBoxComparisonSecondValue.Text));

}

private static bool CheckInput(string system, char KeyChar)

{

return system switch

{

"2" => KeyChar == '0' || KeyChar == '1' || Char.IsControl(KeyChar),

"8" => KeyChar == '0' ||

KeyChar == '1' ||

KeyChar == '2' ||

KeyChar == '3' ||

KeyChar == '4' ||

KeyChar == '5' ||

KeyChar == '6' ||

KeyChar == '7' ||

Char.IsControl(KeyChar),

"10" => Char.IsNumber(KeyChar) || Char.IsControl(KeyChar),

"16" => KeyChar == 'A' ||

KeyChar == 'B' ||

KeyChar == 'C' ||

KeyChar == 'D' ||

KeyChar == 'E' ||

KeyChar == 'F' ||

KeyChar == 'a' ||

KeyChar == 'b' ||

KeyChar == 'c' ||

KeyChar == 'd' ||

KeyChar == 'e' ||

KeyChar == 'f' ||

Char.IsNumber(KeyChar) || Char.IsControl(KeyChar),

\_ => false,

};

}

}

}

Код автотестов

using Microsoft.VisualStudio.TestTools.UnitTesting;

using Number\_Systems;

namespace TestProject

{

[TestClass]

public class UnitTest1

{

//тесты оператора сложения

[TestMethod]

public void TestOperatorPlus1()

{

Number number = new Number("1011", 2) + new Number("7261", 8);

Assert.AreEqual("3772", number.Value);

}

[TestMethod]

public void TestOperatorPlus2()

{

Number number = new Number(63) + new Number("D8", 16);

Assert.AreEqual("279", number.Value);

}

//тесты оператора вычитания

[TestMethod]

public void TestOperatorMinus1()

{

Number number = new Number("1101010", 2) - new Number("10110", 2);

Assert.AreEqual("84", number.Value);

}

[TestMethod]

public void TestOperatorMinus2()

{

Number number = new Number("D1", 16) - new Number("473", 8);

Assert.AreEqual("-106", number.Value);

}

//тесты оператора умножения

[TestMethod]

public void TestOperatorMultiply1()

{

Number number = new Number("0011", 2) \* new Number("11", 16);

Assert.AreEqual("51", number.Value);

}

[TestMethod]

public void TestOperatorMultiply2()

{

Number number = new Number("26", 16) \* new Number("10", 8);

Assert.AreEqual("304", number.Value);

}

//тесты оператора деления

[TestMethod]

public void TestOperatorDivide1()

{

Number number = new Number("78", 16) / new Number("1001", 2);

Assert.AreEqual("13,333333", number.Value);

}

[TestMethod]

public void TestOperatorDivide2()

{

Number number = new Number("210", 10) / new Number("44", 8);

Assert.AreEqual("5,8333335", number.Value);

}

}

}

Ссылка на код в GitHub:

### Результаты тестов

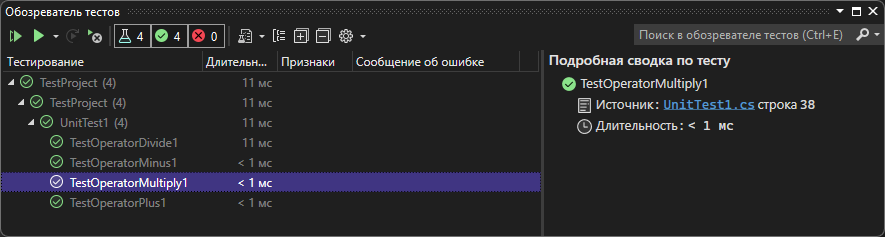


Рисунок 1 – Результат автотестов

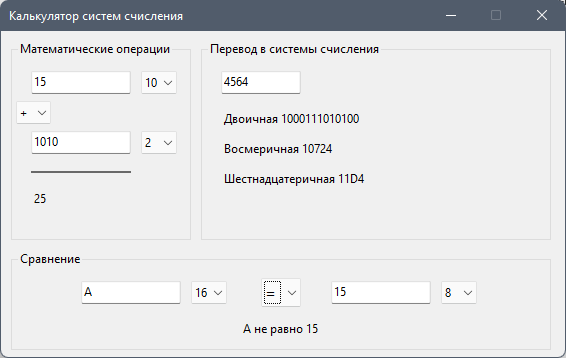


Рисунок 2 – Внешний вид программы