|  |  |
| --- | --- |
|  | /storage/emulated/0/.polarisOffice5/polarisTemp/image1.png |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждениевысшего профессионального образования «Московский технологический университет»  МИРЭА | | |

Институт Информационных Технологий

Кафедра Корпоративных Информационных Систем

**ОТЧЕТ**

по Лабораторной Работе №3

на тему

«Простые типы данных. Ввод и вывод простых типов данных»

по дисциплине

«Процедурное Программирование»

Выполнил студент группы ИКБО-07-17 Акжигитов Р. Р.

Принял Асс. Каф. КИС Алдобаева В. Н.

Выполнено «30» октября 2017 г.

Зачтено «\_\_\_» \_\_\_\_\_\_\_\_\_ 2017 г.

Москва, 2017

**Задание**

Создать приложение - калькулятор, в котором расположены поля ввода 2-х аргументов и набор возможных (предложенных в заданиях) операций над ними. Данные из текстовых полей должны приводятся к нужному типу: числовому или логическому, в зависимости от типа операции. Если в процессе приведения типов возникла ошибка (например, один из аргументов пуст или аргументы различаются по типу). Результат операции вывести в listBox с сохранением истории операций.

**Теоретическая часть**

**Математические функции библиотеки.**

В библиотеке **Math** существует множество полезных математических функций, таких как e в степени x, степень числа, логарифмы, синусы (др. тригонометрические функции), функция округления Round. Также существуют и константы: число е, число Пи.

**Приведение типов.**

Введенные данные пользователем нужно привести к какому-то типу, в нашем случае либо числовой, либо логический. Пользователь может случайно или нет допустить ошибки при вводе, поэтому перед приведением нужно для начала произвести какие-то подготовительные действия, например, для логического типа – это удаление пробелов по краям, приведение текста к нижнему регистру (для более широкого охвата вариантов ввода пользователя), также для логического типа мною было предусмотрена возможность введения данных, как на английском, так и на русском языках. Именно поэтому я и не использовал стандартный вариант bool.Parse, ибо он работает только с английским вариантом.

Числовой тип парсится через стандартное решение double.Parse. Поддерживает ввод, как с точкой, так и без. Также для расширения возможностей различных способов ввода: знаки «запятая» и «точка» программа рассматривает одинаково, как отделение дробной от целой части.

**Неточность типа.**

В работе мы использовали «широкодиапазонный» double, который нам позволяет, в отличие от float, хранить больший диапазон (множество) чисел и причем более точных (больше кол-во знаков (чисел) после запятой соответствуют реальности).

**Исключительные ситуации.**

Существует много вариантов, когда пользователь может ввести некорректные данные, причем требуется определить, в каком поле произошла ошибка, также возможно определить какого типа ошибка. Вот пару возможных ситуаций:

1. Пользователь не ввел данные в одно или в оба поля. **Решение:** вывести сообщение о том, что данные неправильные и не соответствуют нужному типу.
2. Пользователь ввел некорректные данные, которые не являются ни числом, ни логикой, просто строку. **Решение:** вывести сообщение о неправильности введенных данных.
3. Пользователь хотел ввести строку или логику, но ошибся при вводе значения, случайно поставил пробел или лишнюю запятую в числе.
4. Пользователь ввел аргумент поддерживаемый программой (логика или число), однако операция не может его обработать, например, сравнение логики (больше, меньше), сложение, умножение логических данных. **Решение:** вывести сообщение о том, что нужен такой-то тип, а обнаружен другой тип.
5. Пользователь ввел данные различных типов, причем операция может поддерживать их одновременно (как побитовые и логические операции, «принадлежащие» одним и тем же кнопкам), но программа не может понять, какую конкретно операции хочет в результате получить пользователь, ей нужно, чтобы оба типа были одинаковые. **Решение:** вывести сообщение о разных типах, просьбу уточнить данные. «Оба аргумента должны быть либо числовыми, либо логическими».

**Алгоритм решения задачи**

1. Создать форму и создать разметку в конструкторе (все компоненты).
2. Создать функцию проверки ожидаемого типа и введенного типа, чтобы в некоторых операциях выводить исключительные ситуации.
3. Создать функцию получения аргументов из введенных данных.
4. Написать для каждой операции (кнопки) свое событие, реализующее свою операцию.
5. Создать «всегда видимую» историю операций, чтобы новые вычисления не уходили вниз, а появлялись сверху.

**1 вариант**

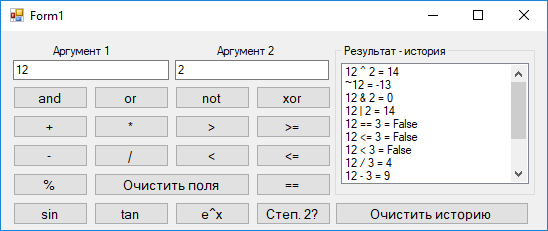
1. является ли число степенью 2

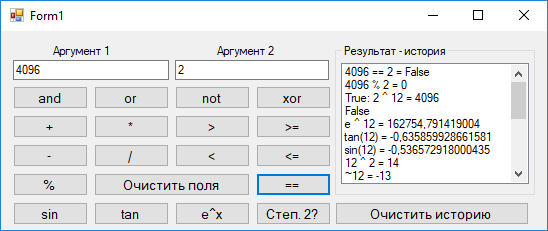
2. sin

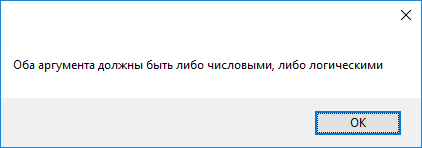
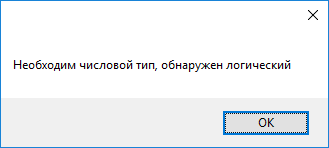
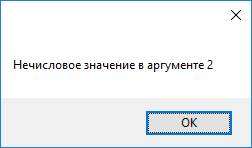
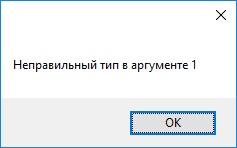
3. tan

4. e^x

**Тестирование**







**Заключение**

Реализовав подобное приложение, я лучше разобрался в том, как обрабатывать исключения и как приводить типы. Узнал больше о неточностях числовых типов.

**Исходный код**

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 laba3

{

public partial class Form1 : Form

{

public double arg1, arg2;

public int argBool1 = -1, argBool2 = -1;

public string CheckIfNumbersOrBool(Tuple<double, double, int, int> arguments, string expected) // -1 - numbers; 0, 1 - bools

{

var args = arguments;

if (args.Item3 == -1 && args.Item4 == -1)

return "numbers";

else if (args.Item3 >= 0 && args.Item4 >= 0)

return "bools";

else

if (expected == "numbers")

{

if (args.Item3 != -1) return "error in 1";

else if (args.Item4 != -1) return "error in 2";

else return "error";

}

else if (expected == "bools")

{

if (args.Item3 < 0) return "error in 1";

else if (args.Item4 < 0) return "error in 2";

else return "error";

}

else return "error";

}

public Tuple<double, double, int, int> getArgs(int param = 2)

{

string text1 = textBox1.Text.Trim().ToLower();

string text2 = textBox2.Text.Trim().ToLower();

if (text1 == "истина" || text1 == "true")

argBool1 = 1;

else if (text1 == "ложь" || text1 == "false")

argBool1 = 0;

else

{

if (!double.TryParse(textBox1.Text.Replace(".", ","), out arg1))

{

MessageBox.Show("Неправильный тип в аргументе 1");

argBool1 = -2;

}

else

argBool1 = -1;

}

if (text2 == "истина" || text2 == "true")

argBool2 = 1;

else if (text2 == "ложь" || text2 == "false")

argBool2 = 0;

else

{

if (!double.TryParse(textBox2.Text.Replace(".", ","), out arg2))

{

if (param == 2)

MessageBox.Show("Неправильный тип в аргументе 2");

argBool2 = -2;

}

else

argBool2 = -1;

}

return Tuple.Create(arg1, arg2, argBool1, argBool2);

}

public Form1()

{

InitializeComponent();

}

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

{

var args = getArgs();

var type = CheckIfNumbersOrBool(args, "numbers");

if (type == "numbers")

listBox1.Items.Insert(0, args.Item1.ToString() + " + " + args.Item2.ToString() + " = " + (args.Item1 + args.Item2).ToString());

else if (type == "bools")

MessageBox.Show("Необходим числовой тип, обнаружен логический");

else if (type.StartsWith("error in"))

MessageBox.Show("Нечисловое значение в аргументе " + (type == "error in 1" ? "1" : "2"));

}

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

{

var args = getArgs();

var type = CheckIfNumbersOrBool(args, "numbers");

if (type == "numbers")

listBox1.Items.Insert(0, args.Item1.ToString() + " - " + args.Item2.ToString() + " = " + (args.Item1 - args.Item2).ToString());

else if (type == "bools")

MessageBox.Show("Необходим числовой тип, обнаружен логический");

else if (type.StartsWith("error in"))

MessageBox.Show("Нечисловое значение в аргументе " + (type == "error in 1" ? "1" : "2"));

}

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

{

var args = getArgs();

var type = CheckIfNumbersOrBool(args, "numbers");

if (type == "numbers")

listBox1.Items.Insert(0, args.Item1.ToString() + " \* " + args.Item2.ToString() + " = " + (args.Item1 \* args.Item2).ToString());

else if (type == "bools")

MessageBox.Show("Необходим числовой тип, обнаружен логический");

else if (type.StartsWith("error in"))

MessageBox.Show("Нечисловое значение в аргументе " + (type == "error in 1" ? "1" : "2"));

}

private void button4\_Click(object sender, EventArgs e)

{

var args = getArgs();

var type = CheckIfNumbersOrBool(args, "numbers");

if (CheckIfNumbersOrBool(args, "numbers") == "numbers")

listBox1.Items.Insert(0, args.Item1.ToString() + " / " + args.Item2.ToString() + " = " + (args.Item1 / args.Item2).ToString());

else if (type == "bools")

MessageBox.Show("Необходим числовой тип, обнаружен логический");

else if (type.StartsWith("error in"))

MessageBox.Show("Нечисловое значение в аргументе " + (type == "error in 1" ? "1" : "2"));

}

private void button5\_Click(object sender, EventArgs e)

{

var args = getArgs();

var type = CheckIfNumbersOrBool(args, "numbers");

if (CheckIfNumbersOrBool(args, "numbers") == "numbers")

listBox1.Items.Insert(0, args.Item1.ToString() + " % " + args.Item2.ToString() + " = " + (args.Item1 % args.Item2).ToString());

else if (type == "bools")

MessageBox.Show("Необходим числовой тип, обнаружен логический");

else if (type.StartsWith("error in"))

MessageBox.Show("Нечисловое значение в аргументе " + (type == "error in 1" ? "1" : "2"));

}

private void button9\_Click(object sender, EventArgs e)

{

var args = getArgs();

var type = CheckIfNumbersOrBool(args, "numbers");

if (CheckIfNumbersOrBool(args, "numbers") == "numbers")

listBox1.Items.Insert(0, args.Item1.ToString() + " > " + args.Item2.ToString() + " = " + (args.Item1 > args.Item2).ToString());

else if (type == "bools")

MessageBox.Show("Необходим числовой тип, обнаружен логический");

else if (type.StartsWith("error in"))

MessageBox.Show("Нечисловое значение в аргументе " + (type == "error in 1" ? "1" : "2"));

}

private void button11\_Click(object sender, EventArgs e)

{

var args = getArgs();

var type = CheckIfNumbersOrBool(args, "numbers");

if (CheckIfNumbersOrBool(args, "numbers") == "numbers")

listBox1.Items.Insert(0, args.Item1.ToString() + " < " + args.Item2.ToString() + " = " + (args.Item1 < args.Item2).ToString());

else if (type == "bools")

MessageBox.Show("Необходим числовой тип, обнаружен логический");

else if (type.StartsWith("error in"))

MessageBox.Show("Нечисловое значение в аргументе " + (type == "error in 1" ? "1" : "2"));

}

private void button12\_Click(object sender, EventArgs e)

{

var args = getArgs();

var type = CheckIfNumbersOrBool(args, "numbers");

if (CheckIfNumbersOrBool(args, "numbers") == "numbers")

listBox1.Items.Insert(0, args.Item1.ToString() + " >= " + args.Item2.ToString() + " = " + (args.Item1 >= args.Item2).ToString());

else if (type == "bools")

MessageBox.Show("Необходим числовой тип, обнаружен логический");

else if (type.StartsWith("error in"))

MessageBox.Show("Нечисловое значение в аргументе " + (type == "error in 1" ? "1" : "2"));

}

private void button13\_Click(object sender, EventArgs e)

{

var args = getArgs();

var type = CheckIfNumbersOrBool(args, "numbers");

if (CheckIfNumbersOrBool(args, "numbers") == "numbers")

listBox1.Items.Insert(0, args.Item1.ToString() + " <= " + args.Item2.ToString() + " = " + (args.Item1 <= args.Item2).ToString());

else if (type == "bools")

MessageBox.Show("Необходим числовой тип, обнаружен логический");

else if (type.StartsWith("error in"))

MessageBox.Show("Нечисловое значение в аргументе " + (type == "error in 1" ? "1" : "2"));

}

private void button14\_Click(object sender, EventArgs e)

{

var args = getArgs();

var type = CheckIfNumbersOrBool(args, "numbers");

if (CheckIfNumbersOrBool(args, "numbers") == "numbers")

listBox1.Items.Insert(0, args.Item1.ToString() + " == " + args.Item2.ToString() + " = " + (args.Item1 == args.Item2).ToString());

else if (type == "bools")

MessageBox.Show("Необходим числовой тип, обнаружен логический");

else if (type.StartsWith("error in"))

MessageBox.Show("Нечисловое значение в аргументе " + (type == "error in 1" ? "1" : "2"));

}

private void button6\_Click(object sender, EventArgs e)

{

var args = getArgs();

if (CheckIfNumbersOrBool(args, "bools") == "bools")

{

var bool1 = true ? args.Item3 == 1 : false;

var bool2 = true ? args.Item4 == 1 : false;

listBox1.Items.Insert(0, bool1.ToString() + " && " + bool2.ToString() + " = " + ((bool1 && bool2).ToString()));

}

else if (CheckIfNumbersOrBool(args, "numbers") == "numbers")

listBox1.Items.Insert(0, args.Item1.ToString() + " & " + args.Item2.ToString() + " = " + ((int)args.Item1 & (int)args.Item2).ToString());

else

MessageBox.Show("Оба аргумента должны быть либо числовыми, либо логическими");

}

private void button8\_Click(object sender, EventArgs e)

{

var args = getArgs();

if (CheckIfNumbersOrBool(args, "bools") == "bools")

{

var bool1 = true ? args.Item3 == 1 : false;

var bool2 = true ? args.Item4 == 1 : false;

listBox1.Items.Insert(0, bool1.ToString() + " || " + bool2.ToString() + " = " + (bool1 || bool2).ToString());

}

else if (CheckIfNumbersOrBool(args, "numbers") == "numbers")

listBox1.Items.Insert(0, args.Item1.ToString() + " | " + args.Item2.ToString() + " = " + ((int)args.Item1 | (int)args.Item2).ToString());

else

MessageBox.Show("Оба аргумента должны быть либо числовыми, либо логическими");

}

private void button10\_Click(object sender, EventArgs e)

{

var args = getArgs(1);

if (args.Item3 >= 0)

{

var bool1 = true ? args.Item3 == 1 : false;

listBox1.Items.Insert(0, "!" + bool1.ToString() + " = " + (!bool1).ToString());

}

else if (args.Item3 == -1)

listBox1.Items.Insert(0, "~" + args.Item1.ToString() + " = " + (~(int)args.Item1).ToString());

}

private void button7\_Click(object sender, EventArgs e)

{

var args = getArgs();

if (CheckIfNumbersOrBool(args, "bools") == "bools")

{

var bool1 = true ? args.Item3 == 1 : false;

var bool2 = true ? args.Item4 == 1 : false;

listBox1.Items.Insert(0, bool1.ToString() + " ^ " + bool2.ToString() + " = " + (bool1 ^ bool2).ToString());

}

else if (CheckIfNumbersOrBool(args, "numbers") == "numbers")

listBox1.Items.Insert(0, args.Item1.ToString() + " ^ " + args.Item2.ToString() + " = " + ((int)args.Item1 ^ (int)args.Item2).ToString());

else

MessageBox.Show("Оба аргумента должны быть либо числовыми, либо логическими");

}

private void button17\_Click(object sender, EventArgs e)

{

var args = getArgs(1);

if (args.Item3 == -1)

{

listBox1.Items.Insert(0, "sin(" + args.Item1 + ") = " + Math.Sin(args.Item1).ToString());

}

else

{

MessageBox.Show("Необходим числовой тип, обнаружен логический");

}

}

private void button18\_Click(object sender, EventArgs e)

{

var args = getArgs(1);

if (args.Item3 == -1)

{

listBox1.Items.Insert(0, "tan(" + args.Item1 + ") = " + Math.Tan(args.Item1).ToString());

}

else

{

MessageBox.Show("Необходим числовой тип, обнаружен логический");

}

}

private void button19\_Click(object sender, EventArgs e)

{

var args = getArgs(1);

if (args.Item3 == -1)

{

listBox1.Items.Insert(0, "e ^ " + args.Item1 + " = " + Math.Exp(args.Item1).ToString());

}

else

{

MessageBox.Show("Необходим числовой тип, обнаружен логический");

}

}

private void button20\_Click(object sender, EventArgs e)

{

var args = getArgs(1);

if (args.Item3 == -1)

{

var log2 = Math.Log(args.Item1) / Math.Log(2);

if ((int)log2 == log2)

listBox1.Items.Insert(0, ((int)log2 == log2).ToString() + ": 2 ^ " + log2.ToString() + " = " + args.Item1.ToString());

else

listBox1.Items.Insert(0, ((int)log2 == log2).ToString());

}

else

{

MessageBox.Show("Необходим числовой тип, обнаружен логический");

}

}

private void button15\_Click(object sender, EventArgs e)

{

listBox1.Items.Clear();

}

private void button16\_Click(object sender, EventArgs e)

{

textBox1.Text = "";

textBox2.Text = "";

}

}

}