

Лабораторная работа №3

Тема работы: Простые типы данных. Ввод и вывод простых типов данных.

Задание:

Часть 1.

Создать приложение, реализующее следующее множество математических операций:

1. Сложение
2. Вычитание
3. Умножение
4. Деление
5. Взятие остатка
6. Логическое и побитовое умножение (AND)
7. Логическое и побитовое сложение (OR)
8. Логическое и побитовое сложение по модулю 2 (XOR)
9. Логическое и побитовое отрицание (NOT)
10. Больше
11. Меньше
12. Больше либо равно
13. Меньше либо равно
14. Равно

Для целочисленных, вещественных и логических данных.

Приложение должно проверять введенные аргументы на корректность и соответствие типу данных. Константе «истина» соответствует текст «true» (без кавычек), константе «ложно» — текст «false» (без кавычек).

В случае некорректного ввода или несоответствия типа данных пользователь должен получить соответствующее сообщение, с указанием конкретной ошибки и аргумента, содержащего ошибку.

Результат вычисления должен вместе с аргументами в виде полного выражения помещаться в конец компонента «спискок». Опционально — предусмотреть функцию очистки истории вычислений.

Часть 2.

Дополнить приложение математическими функциями согласно варианту.

Выполнение:

Для начала необходимо создать пользовательский интерфейс приложения, по аналогии с рисунком. После размещения всех необходимых компонентов (каких именно?) в конструкторе форм, следует определить основные рабочие процедуры. Все вычисления мы будем проводить, пользуясь стандартными математическими функциями библиотеки .NET Framework.

Сперва нам нужно получить от пользователя аргументы. Пользователь заносит их в поля ввода, содержимое которых хранится в системе в виде строки символов. Соответственно, эти строки необходимо преобразовать в переменные соответствующих типов, например, целые. Для этого в стандартной библиотеке, в псевдоклассах числовых типов есть функции Parse: int.Parse() для преобразования строки в целое число, float.Parse() — в вещественное и bool.Parse() — в логическое. Для обратного преобразования служит универсальная функция ToString() соответствующего типа..

Рассмотрим пример (считается, что поля ввода для аргументов не переименовывались, и имеют имена по-умолчанию textBox1 и textBox2):

void button1\_Click(object sender, EventArgs e)

{

int a = int.Parse(textBox1.Text);

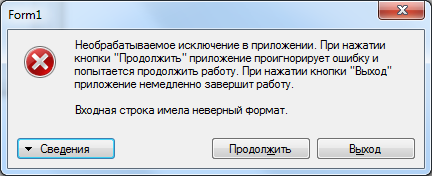
int b = int.Parse(textBox2.Text);

int res = a+b;

MessageBox.Show("Результат: "+res.ToString());

}

Однако, в случае невозможности преобразования эти функции выбрасывают исключительную ситуацию (ошибку), которая сопровождается сообщением следующего вида:



В случае запуска приложения с отладчиком автоматически откроется файл исходного кода на строке, вызвавшей ошибку.

Чтобы избежать подобных ситуаций, в целом пугающих пользователей, необходимо каким-то образом избавиться от этих сообщений и заменить их на более понятные. Для этого существуют 2 подхода: первый, это обработать исключительную ситуацию (об этом мы поговорим позднее), второй, это не допустить ее появления. В данном случае можно легко реализовать 2-й подход с использованием функций с приставкой Try. Эти функции имеют 2 параметра — строку для преобразования и результат преобразования и возвращают логическое значение, удалось преобразование или же нет. Обратите внимание, что обратное преобразование (число в строку) всегда возможно, и функции преобразования чисел в строку не имеют таких «безопасных» версий, так как они не нужны — преобразование «безопасно» всегда. Рассмотрим пример:

void button1\_Click(object sender, EventArgs e)

{

int a, b, res;

if (int.TryParse(textBox1.Text, out a) && int.TryParse(textBox1.Text, out b))

{

res = a+b;

MessageBox.Show("Результат: "+res.ToString());

}

MessageBox.Show("Неверные аргументы", "Ошибка ввода", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

В примере показана обработка неправильного ввода аргументов. Помните, что для успешного завершения операции требуется, чтобы одновременно ОБА аргумента были введены корректно. Также в примере пользователю показывается сообщение о неправильно введенном параметре.

Для хранения параметра в примере использован целочисленный тип int. Это не единственный предопределенный целочисленный тип в C#, другие целочисленные типы и их диапазон значений можно посмотреть в таблице:

|  |  |  |
| --- | --- | --- |
| Тип | Диапазон | Размер |
| **int, Int32** | -2147483648..2147483647 | Знаковый 4 байт (32 бит) |
| **Uint32** | 0..4294967295 | Беззнаковый 4 байт (32 бит) |
| **sbyte, SByte** | -128..127 | Знаковый 1 байт (8 бит) |
| **short, Int16** | -32768..32767 | Знаковый 2 байт (16 бит) |
| **long, Int64** | -263..263-1 | Знаковый 8 байт (64 бит) |
| **byte, Byte** | 0..255 | Беззнаковый 1 байт (8 бит) |
| **UInt16** | 0..65535 | Беззнаковый 2 байт (16 бит) |
| **UInt64** | 0..264-1 | Беззнаковый 8 байт (64 бит) |

Чаще всего используются типы int и UInt32. Также часто используются не указанные в таблице типы IntPtr и UIntPtr. Размеры и диапазон этих типов отличается даны для 32 битного и для 64 битного режима. В 32 битном режиме их размер 32 бит (4 байта) в 64 битном — 64 бит (8 байт).

Помимо целых типов, существуют вещественные — которые хранят числа с плавающей запятой (т.е. нецелые).

|  |  |  |  |
| --- | --- | --- | --- |
| Тип | Диапазон | Кол-во разрядов | Размер (байт) |
| **float, Single** | 1.5 × 10-45 .. 3.4 × 1038 | 7-8 | 4 |
| **double, Double** | 5.0 × 10-324 .. 1.7 × 10308 | 15-16 | 8 |
| **Decimal** | -7,9 × 1028 .. 7,9 × 1028 / (100–28) | 10-20 | 10 |

В данной работе рекомендуется использовать тип Double, т.к. тип Single из-за особенностей обработки вещественных чисел в ЭВМ может давать отличные от калькулятора значения уже в 7-8 знаке после запятой.

Логический тип имеет всего два значения, это «истина» («true») и «ложь» («false»). Основным логическим типом в языке C# является тип System.Boolean (bool). Любое логическое выражение имеет тип System.Boolean (bool). Для проверки логических выражений используется условный оператор if:

if (выражение)

Оператор;

[else

Оператор;]

Ветка else язвляется необязательной.

Пример использования логического выражения:

void button100500\_Click(object sender, EventArgs e)

{

int a = int.Parse(textBox1.Text);

int b = int.Parse(textBox2.Text);

if (a == b)

MessageBox.Show(a.ToString()+" равно "+b.ToString());

else

MessageBox.Show(a.ToString()+" не равно "+b.ToString());

}

1 вариант

1. является ли число степенью 2
2. sin
3. tan
4. e^x

2 вариант

1. является ли число степенью 3
2. cos
3. ctan
4. x^y

3 вариант

1. является ли число степенью 4
2. arctan
3. arcsin
4. корень степени y из x

4 вариант

1. является ли число степенью 5
2. arcctan
3. arccos
4. ln x