Дисциплина «Программирование корпоративных систем»
Рабочая тетрадь 1
Основы языка C#. Разработка калькулятора

Теоретический материал

ПЕРЕМЕННЫЕ

Для хранения данных в программе применяются **переменные**.

Переменная представляет именованную область памяти, в которой хранится значение определенного типа. Переменная имеет тип, имя и значение. Тип определяет, какого рода информацию может хранить переменная.

Перед использованием любую переменную надо определить. Синтаксис определения переменной выглядит следующим образом:

```
тип имя_переменной;
int x;
```

ТИПЫ ДАННЫХ

В языке C# есть следующие базовые типы данных:

- **bool**: хранит значение true или false (логические литералы). Представлен системным типом **System.Boolean**
- **byte**: хранит целое число от 0 до 255 и занимает 1 байт. Представлен системным типом **System.Byte**
- **sbyte**: хранит целое число от -128 до 127 и занимает 1 байт. Представлен системным типом **System.SByte**
- **short**: хранит целое число от -32768 до 32767 и занимает 2 байта. Представлен системным типом System.Int16
- **ushort**: хранит целое число от 0 до 65535 и занимает 2 байта. Представлен системным типом **System.UInt16**
- **int**: хранит целое число от -2147483648 до 2147483647 и занимает 4 байта. Представлен системным типом **System.Int32.** Все целочисленные литералы по умолчанию представляют значения типа int:
- **uint**: хранит целое число от 0 до 4294967295 и занимает 4 байта. Представлен системным типом **System.UInt32**
- **long**: хранит целое число от –9 223 372 036 854 775 808 до 9 223 372 036 854 775 807 и занимает 8 байт. Представлен системным типом **System.Int64**
- **ulong**: хранит целое число от 0 до 18 446 744 073 709 551 615 и занимает 8 байт. Представлен системным типом **System.UInt64**
- **float**: хранит число с плавающей точкой от $-3.4*10^{38}$ до $3.4*10^{38}$ и

занимает 4 байта. Представлен системным типом **System.Single**

- **double**: хранит число с плавающей точкой от $\pm 5.0*10^{-324}$ до $\pm 1.7*10^{308}$ и занимает 8 байта. Представлен системным типом **System.Double**
- **decimal**: хранит десятичное дробное число. Если употребляется без десятичной запятой, имеет значение от $\pm 1.0*10^{-28}$ до $\pm 7.9228*10^{28}$, может хранить 28 знаков после запятой и занимает 16 байт. Представлен системным типом **System.Decimal**
- **char**: хранит одиночный символ в кодировке Unicode и занимает 2 байта. Представлен системным типом **System.Char.** Этому типу соответствуют символьные литералы:
- **string**: хранит набор символов Unicode. Представлен системным типом **System.String**. Этому типу соответствуют строковые литералы.
- **object**: может хранить значение любого типа данных и занимает 4 байта на 32-разрядной платформе и 8 байт на 64-разрядной платформе. Представлен системным типом **System.Object**, который является базовым для всех других типов и классов .NET.

КОНСОЛЬНЫЙ ВЫВОД

Для вывода информации на консоль мы уже использовали встроенный метод **Console.WriteLine**. То есть, если мы хотим вывести некоторую информацию на консоль, то нам надо передать ее в метод Console.WriteLine:

```
Console.WriteLine("Добро пожаловать в C#!");
```

Нередко возникает необходимость вывести на консоль в одной строке значения сразу нескольких переменных. В этом случае мы можем использовать прием, который называется **интерполяцией:**

```
1  string name = "Tom";
2  int age = 34;
3  double height = 1.7;
4  Console.WriteLine($"Имя: {name}  Возраст: {age}  Рост: {height}м");
```

Для встраивания отдельных значений в выводимую на консоль строку используются фигурные скобки, в которые заключается встраиваемое значение. Это можем значение переменной ({name}) или более сложное выражение (например, операция сложения {4 + 7}). А перед всей строкой ставится знак доллара $.

При выводе на консоль вместо помещенных в фигурные скобки выражений будут выводиться их значения:

Есть другой способ вывода на консоль сразу нескольких значений:

```
1  string name = "Tom";
2  int age = 34;
```

```
3    double height = 1.7;
4    Console.WriteLine("Имя: {0}  Возраст: {2}  Рост: {1}м", name, height, age);
```

КОНСОЛЬНЫЙ ВВОД

Кроме вывода информации на консоль мы можем получать информацию с консоли. Для этого предназначен метод **Console.ReadLine()**. Он позволяет получить введенную строку.

```
1    Console.Write("Введите свое имя: ");
2    string? name = Console.ReadLine();
3    Console.WriteLine($"Привет {name}");
```

В данном случае все, что вводит пользователь, с помощью метода Console.ReadLine() передается в переменную name.

Особенностью метода Console.ReadLine() является то, что он может считать информацию с консоли только в виде строки. Кроме того, возможная ситуация, когда для метода Console.ReadLine не окажется доступных для считывания строк, то есть когда ему нечего считывать, он возвращаает значение **null**, то есть, грубо говоря, фактически отсутствие значения. И чтобы отразить эту ситуацию мы определяем переменную name, в которую получаем ввод с консоли, как переменную типа **string?**. Здесь string указывает, что переменная может хранить значения типа string, то есть строки. А знак вопроса **?** указывает, что переменная также может хранить значение **null**, то есть по сути не иметь никакого значения.

Однако, может возникнуть вопрос, как нам быть, если, допустим, мы хотим ввести возраст в переменную типа int или другую информацию в переменные типа double или decimal? По умолчанию платформа .NET предоставляет ряд методов, которые позволяют преобразовать различные значения к типам int, double и т.д. Некоторые из этих методов:

- **Convert.ToInt32()** (преобразует к типу int)
- **Convert.ToDouble()** (преобразует к типу double)
- **Convert.ToDecimal()** (преобразует к типу decimal)

Задание 1.1

Задача:

Написать программу реализующую функционал классического калькулятора средствами языка C#, предусмотреть реализацию следующих операций:
+, -, *, /,%, 1/x, x^2, корень квадратный из x, M+, M-, MR.

В раздел решения приложить код решения и текстовое описание программного продукта по следующему плану:

1. Функционал;
2. Ограничения;
3. Возможные ошибки.

Решение:

```
class Calculator
{
   static double memory = 0;

   static void Main()
   {
      bool continueCalculations = true;

      while (continueCalculations)
      {
         DisplayMenu();
         string choice = Console.ReadLine();

         switch (choice)
         {
            case "1": Addition(); break;
            case "2": Subtraction(); break;
            case "3": Multiplication(); break;
            case "4": Division(); break;
            case "5": Square(); break;
            case "6": SquareRoot(); break;
            case "7": Fraction(); break;
            case "8": MemoryAdd(); break;
            case "9": MemorySubtract(); break;
            case "10": MemoryRecall(); break;
            case "11": Percentage(); break;
            case "0":
               continueCalculations = false;
               Console.WriteLine("До свидания!");
```

```
                continue;
            default:
                Console.WriteLine("Неверный выбор. Попробуйте снова.");
                break;
        }

        if (continueCalculations)
            continueCalculations = AskToContinue();
    }
}

static void DisplayMenu()
{
    Console.WriteLine("\nВыберите операцию:");
    Console.WriteLine("1. Сложение (+)");
    Console.WriteLine("2. Вычитание (-)");
    Console.WriteLine("3. Умножение (*)");
    Console.WriteLine("4. Деление (/)");
    Console.WriteLine("5. Возведение в квадрат (x^2)");
    Console.WriteLine("6. Квадратный корень (sqrt)");
    Console.WriteLine("7. Обратное число (1/x)");
    Console.WriteLine("8. Добавить в память (M+)");
    Console.WriteLine("9. Вычесть из памяти (M-)");
    Console.WriteLine("10. Вызвать из памяти (MR)");
    Console.WriteLine("11. Процент (%)");
    Console.WriteLine("0. Выход");
}

static bool AskToContinue()
{
    Console.Write("Хотите продолжить? (д/н): ");
    string response = Console.ReadLine();
    if (response == "д" || response == "Д")
    {
        return true;
    }
    else
    {
        return false;
    }
}

static double CheckNumber(string message)
{
```

```
        double result;
        bool isValid;
        do
        {
            Console.Write(message);
            string input = Console.ReadLine();
            isValid = double.TryParse(input, out result);
            if (!isValid)
            {
                string alternativeInput = input.Replace(',', '.');
                isValid = double.TryParse(alternativeInput, out result);
                if (!isValid)
                {
                    alternativeInput = input.Replace('.', ',');
                    isValid = double.TryParse(alternativeInput, out result);
                }
            }
            if (!isValid)
            {
                Console.WriteLine("Некорректный ввод. Пожалуйста, введите
число.");
            }
        } while (!isValid);
        return result;
    }

    static void Addition()
    {
        double a = CheckNumber("Введите первое число: ");
        double b = CheckNumber("Введите второе число: ");
        Console.WriteLine($"Результат: {a} + {b} = {a + b}");
    }

    static void Subtraction()
    {
        double a = CheckNumber("Введите первое число: ");
        double b = CheckNumber("Введите второе число: ");
        Console.WriteLine($"Результат: {a} - {b} = {a - b}");
    }

    static void Multiplication()
    {
        double a = CheckNumber("Введите первое число: ");
        double b = CheckNumber("Введите второе число: ");
```

```
        Console.WriteLine($"Результат: {a} * {b} = {a * b}");
    }


    static void Division()
    {
        double a = CheckNumber("Введите первое число: ");
        double b;
        do
        {
            b = CheckNumber("Введите второе число (не ноль): ");
            if (b == 0)
                Console.WriteLine("Ошибка: деление на ноль недопустимо.");
        } while (b == 0);
        Console.WriteLine($"Результат: {a} / {b} = {a / b}");
    }

    static void Square()
    {
        double x = CheckNumber("Введите число: ");
        Console.WriteLine($"Результат: {x}^2 = {x * x}");
    }

    static void SquareRoot()
    {
        double x;
        do
        {
            x = CheckNumber("Введите неотрицательное число: ");
            if (x < 0)
                Console.WriteLine("Ошибка: нельзя извлечь корень из
отрицательного числа.");
        } while (x < 0);
        Console.WriteLine($"Результат: sqrt({x}) = {Math.Sqrt(x)}");
    }

    static void Fraction()
    {
        double x;
        do
        {
            x = CheckNumber("Введите число (не ноль): ");
            if (x == 0)
                Console.WriteLine("Ошибка: деление на ноль недопустимо.");
```

```
        } while (x == 0);
        Console.WriteLine($"Результат: 1/{x} = {1 / x}");
    }

    static void MemoryAdd()
    {
        double x = CheckNumber("Введите число: ");
        memory += x;
        Console.WriteLine($"Добавлено {x} к памяти. Новое значение:
{memory}");
    }

    static void MemorySubtract()
    {
        double x = CheckNumber("Введите число: ");
        memory -= x;
        Console.WriteLine($"Вычтено {x} из памяти. Новое значение:
{memory}");
    }

    static void MemoryRecall()
    {
        Console.WriteLine($"Значение в памяти: {memory}");
    }

    static void Percentage()
    {
        double number = CheckNumber("Введите число: ");
        double percent = CheckNumber("Введите процент: ");
        double result = (number * percent) / 100;
        Console.WriteLine($"Результат: {percent}% от {number} = {result}");
    }
}
```

Ответ:

1. Функционал
- Сложение (+)
- Вычитание (-)
- Умножение (*)
- Деление (/)
- Процент (%)
- Возведение в квадрат (x^2)
- Квадратный корень (sqrt)
- Обратное число (1/x)

• M+
• M-
• MR

2. Ограничения

• Отсутствует возможность работы со скобками для распределения порядка вычисления

3. Возможные ошибки

• Некорректный ввод числа
• Деление на ноль
• Извлечение квадратного корня из отрицательного числа
• Неверный выбор операции

Реализация одной из операций калькулятора

```
Выберите операцию:
1. Сложение (+)
2. Вычитание (-)
3. Умножение (*)
4. Деление (/)
5. Возведение в квадрат (x^2)
6. Квадратный корень (sqrt)
7. Обратное число (1/x)
8. Добавить в память (M+)
9. Вычесть из памяти (M-)
10. Вызвать из памяти (MR)
11. Процент (%)
0. Выход
5
Введите число: 125
Результат: 125^2 = 15625
```

Реализация M+

```
Выберите операцию:
1. Сложение (+)
2. Вычитание (-)
3. Умножение (*)
4. Деление (/)
5. Возведение в квадрат (x^2)
6. Квадратный корень (sqrt)
7. Обратное число (1/x)
8. Добавить в память (M+)
9. Вычесть из памяти (M-)
10. Вызвать из памяти (MR)
11. Процент (%)
0. Выход
8
Введите число: 544
Добавлено 544 к памяти. Новое значение: 544
```

Реализация MR

```
Выберите операцию:
1. Сложение (+)
2. Вычитание (-)
3. Умножение (*)
4. Деление (/)
5. Возведение в квадрат (x^2)
6. Квадратный корень (sqrt)
7. Обратное число (1/x)
8. Добавить в память (M+)
9. Вычесть из памяти (M-)
10. Вызвать из памяти (MR)
11. Процент (%)
0. Выход
10
Значение в памяти: 544
```