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:

```
using System;

namespace CalculatorApp
{
  class Program
  {
    static void Main(string[] args)
    {
      double memory = 0;

      Console.WriteLine("Классический калькулятор на C#");
      double currentResult;

      while (true)
      {
        Console.WriteLine("\nВведите первое число (или 'exit' для выхода):");
        string input = Console.ReadLine();
        if (input?.ToLower() == "exit")
          break;

        if (!double.TryParse(input, out currentResult))
        {
          Console.WriteLine("Ошибка: неверный ввод числа.");
          continue;
        }

        while (true)
        {
          Console.WriteLine($"\nТекущее число: {currentResult}");
          Console.WriteLine("Введите операцию (+, -, *, /, %, 1/x, x^2, ^, sqrt, M+, M-, MR, MC, exit):");
          string op = Console.ReadLine()?.ToLower();

          switch (op)
          {
            case "exit":
              return;
            case "c":
              currentResult = 0;
              Console.WriteLine("Текущий результат сброшен.");
              continue;
          }

          double secondOperand = 0;
          bool needSecond = (op == "+" || op == "-" || op == "*" || op == "/" || op == "%" || op == "^");
```

```
if (needSecond)
{
    Console.WriteLine("Введите второе число:");
    string secondInput = Console.ReadLine();
    if (!double.TryParse(secondInput, out secondOperand))
    {
        Console.WriteLine("Ошибка: неверный ввод числа.");
        continue;
    }
}

bool operationPerformed = true;
double result = currentResult;

switch (op)
{
    case "+":
        result = currentResult + secondOperand;
        break;
    case "-":
        result = currentResult - secondOperand;
        break;
    case "*":
        result = currentResult * secondOperand;
        break;
    case "^":
        result = Math.Pow(currentResult, secondOperand);
        break;
    case "/":
        if (secondOperand == 0)
        {
            Console.WriteLine("Ошибка: деление на ноль невозможно.");
            operationPerformed = false;
        }
        else
            result = currentResult / secondOperand;
        break;
    case "%":
        if (secondOperand == 0)
        {
            Console.WriteLine("Ошибка: деление по модулю на ноль невозможно.");
            operationPerformed = false;
        }
        else
            result = currentResult % secondOperand;
        break;
    case "1/x":
        if (currentResult == 0)
        {
            Console.WriteLine("Ошибка: обратное число для 0 не существует.");
            operationPerformed = false;
        }
        else
            result = 1 / currentResult;
        break;
    case "x^2":
        result = currentResult * currentResult;
        break;
    case "sqrt":
        if (currentResult < 0)
        {
            Console.WriteLine("Ошибка: корень квадратный из отрицательного числа невозможен.");
            operationPerformed = false;
        }
        else
            result = Math.Sqrt(currentResult);
```

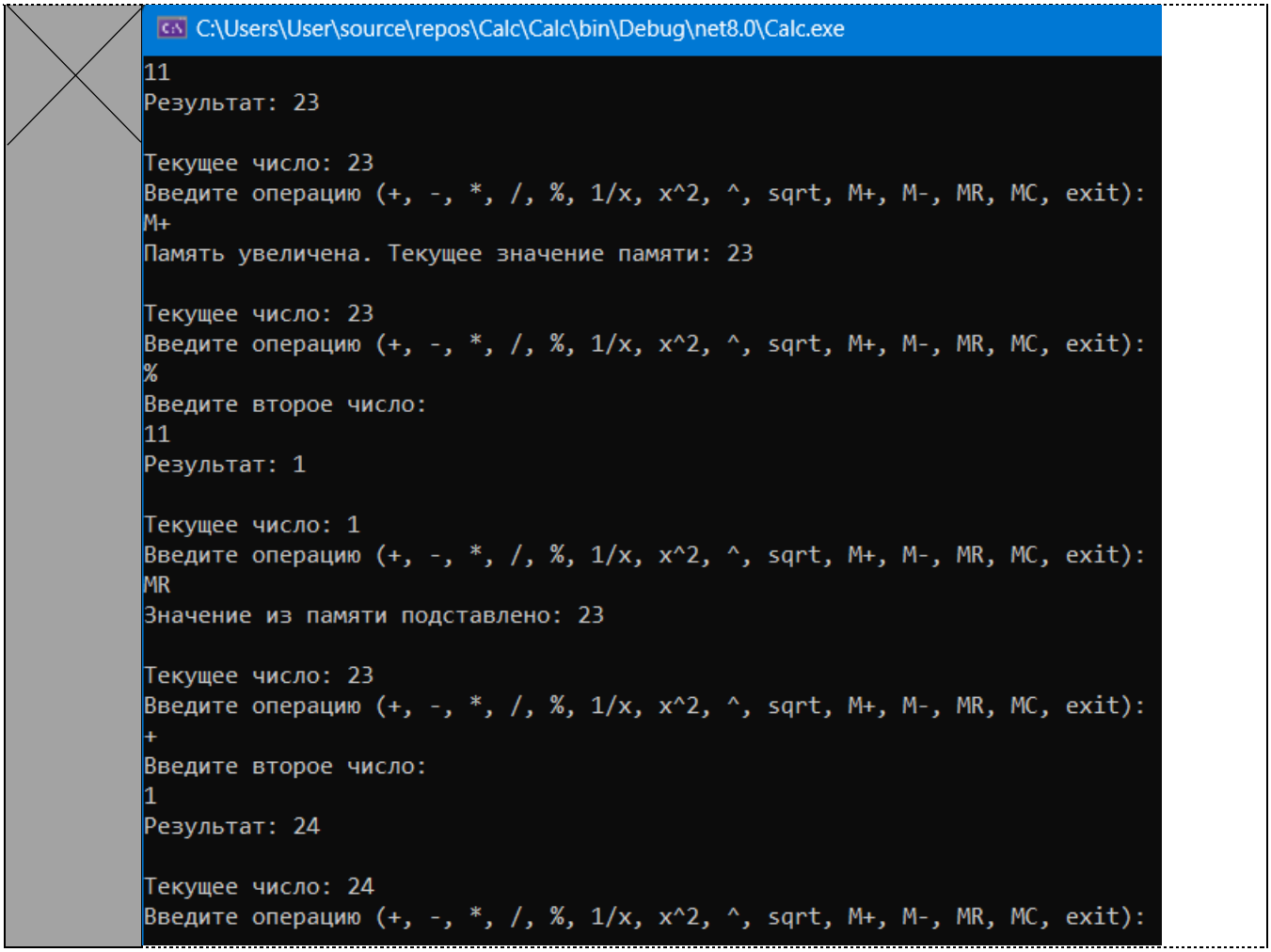
```
                    break;
                case "m+":
                    memory += currentResult;
                    Console.WriteLine($"Память увеличена. Текущее значение памяти: {memory}");
                    operationPerformed = false;
                    break;
                case "m-":
                    memory -= currentResult;
                    Console.WriteLine($"Память уменьшена. Текущее значение памяти: {memory}");
                    operationPerformed = false;
                    break;
                case "mr":
                    currentResult = memory;
                    Console.WriteLine($"Значение из памяти подставлено: {currentResult}");
                    operationPerformed = false;
                    break;
                case "mc":
                    memory = 0;
                    Console.WriteLine("Память очищена.");
                    operationPerformed = false;
                    break;
                default:
                    Console.WriteLine("Ошибка: неизвестная операция.");
                    operationPerformed = false;
                    break;
            }

            if (operationPerformed)
            {
                currentResult = result;
                Console.WriteLine($"Результат: {currentResult}");
            }
        }
    }

    Console.WriteLine("Выход из программы.");
  }
 }
}
```

https://github.com/pkpal-uhobp/workbook_1

Ответ:


C:\Users\User\source\repos\Calc\Calc\bin\Debug\net8.0\Calc.exe
11
Результат: 23
Текущее число: 23
Введите операцию (+, -, *, /, %, 1/x, x^2, ^, sqrt, M+, M-, MR, MC, exit):
M+
Память увеличена. Текущее значение памяти: 23
Текущее число: 23
Введите операцию (+, -, *, /, %, 1/x, x^2, ^, sqrt, M+, M-, MR, MC, exit):
%
Введите второе число:
11
Результат: 1
Текущее число: 1
Введите операцию (+, -, *, /, %, 1/x, x^2, ^, sqrt, M+, M-, MR, MC, exit):
MR
Значение из памяти подставлено: 23
Текущее число: 23
Введите операцию (+, -, *, /, %, 1/x, x^2, ^, sqrt, M+, M-, MR, MC, exit):
+
Введите второе число:
1
Результат: 24
Текущее число: 24
Введите операцию (+, -, *, /, %, 1/x, x^2, ^, sqrt, M+, M-, MR, MC, exit):