# Дисциплина «Программирование корпоративных систем» Рабочая тетрадь 1

**Основы языка С#. Разработка калькулятора**

|  |
| --- |
| **Теоретический материал** |
| **ПЕРЕМЕННЫЕ**  Для хранения данных в программе применяются **переменные**.  Переменная представляет именнованную 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\*1038 до 3.4\*1038 и |

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

* **double**: хранит число с плавающей точкой от ±5.0\*10-324 до ±1.7\*10308 и занимает 8 байта. Представлен системным типом **System.Double**
* **decimal**: хранит десятичное дробное число. Если употребляется без десятичной запятой, имеет значение от ±1.0\*10-28 до ±7.9228\*1028, 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:*** | |
|  | Stack<object> stack = new Stack<object>();  Console.Write("Введите число: ");  string? input = Console.ReadLine();  bool numFlag = true; // Флаг на проверку, что ждет калькулятор: число или операцию  double result = Convert.ToDouble(input);  double memory = 0;  static void PrintStack<T>(Stack<T> stack)  {      // Используем массив для временного хранения элементов      T[] items = stack.ToArray(); // ToArray возвращает элементы в порядке от верхушки к низу      Console.WriteLine("Содержимое стека:");      foreach (T item in items)      {          Console.WriteLine(item);      }  }  // Продолжение (основная часть)  while (input != "=")  {      if (stack.Count == 3)      {          double num1, num2;          string? operation;          num2 = Convert.ToDouble(stack.Pop());          operation = Convert.ToString(stack.Pop());          num1 = Convert.ToDouble(stack.Pop());          switch (operation)          {              case "+":                  result = num1 + num2;                  break;              case "-":                  result = num1 - num2;                  break;              case "\*":                  result = num1 \* num2;                  break;              case "/":                  // Проверка деления на ноль                  if (num2 != 0)                  {                      result = num1 / num2;                  }                  else                  {                      Console.WriteLine("Делить на ноль нельзя!");                  }                  break;              case "%":                  result = num1 \* (num2 / 100);                  break;              default:                  Console.WriteLine("Что то не так");                  break;          }          if (!double.IsInfinity(result))          {              stack.Push(result);              Console.WriteLine($"Ответ: {result}");              numFlag = false;              Console.Write("Введите операцию: ");              input = Console.ReadLine();          }          else          {              Console.WriteLine("Слишком большое число. Начните сначала");              Console.Write("Введите число: ");              numFlag = true;              stack.Clear();              input = Console.ReadLine();          }      }      // Калькулятор ждет число      if (numFlag)      {          if (input == "MR")          {              stack.Push(memory);              numFlag = false;          }          else          {              try              {                  stack.Push(Convert.ToDouble(input)); // конвертируем в double для проверки                  // что пользователь действительно ввел число                  numFlag = false;                  if (!(stack.Count == 3))                  {                      Console.Write("Введите операцию: ");                      input = Console.ReadLine();                  }              }              catch              {                  Console.WriteLine("Введите число!");                  input = Console.ReadLine();              }          }      }      // Калькулятор ждет знак      else      {          if (input == "+" || input == "-" || input == "\*" || input == "/" || input == "%")          {              stack.Push(input);              numFlag = true;          }          else if (input == "1/x")          {              // проверка деления на ноль              if (result != 0)              {                  result = 1 / Convert.ToDouble(stack.Pop());                  stack.Push(result);                  Console.WriteLine($"Ответ: {result}");              }              else              {                  Console.WriteLine("Делить на ноль нельзя!");              }          }          else if (input == "x^2")          {              double tempNum = Convert.ToDouble(stack.Pop());              result = tempNum \* tempNum;              stack.Push(result);              Console.WriteLine($"Ответ: {result}");          }          else if (input == "sqrt")          {              result = Math.Sqrt(Convert.ToDouble(stack.Pop()));              stack.Push(result);              Console.WriteLine($"Ответ: {result}");          }          else if (input == "M+")          {              memory = memory + result;          }          else if (input == "M-")          {              memory = memory - result;          }          else          {              Console.WriteLine("Неверный ввод!");          }          if (numFlag)          {              Console.Write("Введите число: ");              input = Console.ReadLine();          }          else          {              Console.Write("Введите операцию: ");              input = Console.ReadLine();          }      }  } |
| ***Ответ:*** | |
|  |  |