ЛАБОРАТОРНАЯ РАБОТА. ПРЕДОПРЕДЕЛЕННЫЕ ТИПЫ

ДАННЫХ, ПЕРЕМЕННЫЕ, КОНСТАНТЫ

1. Цель и содержание

Цель лабораторной работы: научиться работать с переменными и константами простых типов в C#.

Задачи лабораторной работы:

* научиться объявлять переменные простых типов в языке C#;
* научиться объявлять константы простых типов в языке C#;
* научиться выполнять простейшие действия с переменными и константами.

2. Формируемые компетенции

Лабораторная работа направлена на формирование следующих компетенций:

* способность к проектированию базовых и прикладных

информационных технологий (ПК-11);

* способность разрабатывать средства реализации информационных технологий (методические, информационные, математические, алгоритмические, технические и программные) (ПК-12).

3. Теоретическая часть

Перед выполнением лабораторной работы необходимо изучить материалы лекций. Следует понимать принцип деления типов .NET на типы значений и ссылочные типы.

В данной лабораторной работе необходимо освоить приемы работы с предопределенными типами значений.

2.1 Типы значений C#

Язык C# поддерживает 8 предопределенных целочисленных типов

(таблица 2.1).

Таблица 2.1 – Целочисленные типы C#.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя типа | Тип CTS | Описание | Диапазон  (минимум : максимум) |  |
| sbyte | System.SByte | 8-битное целое со знаком | -128 : 127 |  |
| short | System.Int16 | 16-битное целое со  знаком | -32 768 : 32 767 |  |
| int | System.Int32 | 32-битное целое со  знаком | -2 147 483 648  2 147 483 647 | : |
| long | System.Int64 | 64-битное целое со  знаком | -263 : 263-1 |  |
| byte | System.Byte | 8-битное целое без знака | 0 : 255 |  |
| ushort | System.UInt16 | 16-битное целое без  знака | 0 : 65 535 |  |
| uint | System.UInt32 | 32-битное целое без  знака | 0 : 232-1 |  |
| ulong | System.UInt64 | 64-битное целое без  знака | 0 : 264-1 |  |

Язык C# также поддерживает и типы с плавающей точкой (таблица 2.2).

Таблица 2.2 – Типы с плавающей точкой C#.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя типа | Тип CTS | Описание | Кол-во знаков | Диапазон  (минимум : максимум) |
| float | System.Single | 32-битное с плавающей точкой одинарной  точности | 7 | от 1.51045 до 3.41038 |
| double | System.Double | 64-битное с плавающей точкой двойной  точности | 15/16 | от 5.010324 до 1.710308 |

В таблице 2.3 представлен десятичный тип C#. Данный тип реализован для финансовых операций.

Таблица 2.3 – Десятичный тип C#.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя типа | Тип CTS | Описание | Колво знаков | Диапазон  (минимум : максимум) |
| decimal | System.Decimal | 128-битное с плавающей точкой в десятичной нотации с высокой  точностью | 28 | от 1.01028 до 7.91028 |

Как и во многих языках программирования существует булевский тип (таблица 2.4).

Таблица 2.4 – Булевский тип.

|  |  |  |
| --- | --- | --- |
| Имя типа | Тип CTS | Значения |
| bool | System.Boolean | true или false |

Для хранения одиночных символов в языке C# используется тип char (таблица 2.5)

Таблица 2.5 – Булевский тип.

|  |  |  |
| --- | --- | --- |
| Имя типа | Тип CTS | Значения |
| char | System.Char | Представляет отдельный 16битный (Unicode) символ |

Литералы типа char записываются как одиночные, заключенные в одинарные кавычки символы: ‘F’, ‘w’, ‘ц’, ‘Я’ и т.д.

В переменных типа char можно хранить и специальные символы в виде управляющих последовательностей (таблица 2.6).

Таблица 2.6 – Представление символов в виде управляющих последовательностей.

|  |  |
| --- | --- |
| Управляющая последовательность | Символ |
| \’ | Одиночная кавычка |
| \” | Двойная кавычка |
| \\ | Обратный слэш |
| \0 | Пусто |
| \a | Предупреждение (звуковой сигнал) |
| \b | Забой |
| \f | Подача формы |
| \n | Новая строка |
| \r | Возврат каретки |
| \t | Символ табуляции |
| \v | Вертикальная табуляция |

Если отдельные символы объединены в строку, то необходимо использовать тип string, который отображается на тип CTS – System.String.

2.2 Объявление и инициализация переменных в C#

Синтаксис объявления переменных в C# выглядит следующим образом:

*ТипДанных ИдентификаторПеременной*; Например:



Этот код объявляет переменную типа int с именем a. Компилятор не позволит использовать эту переменную до тех пор, пока она не будет инициализирована (т.е. пока ей не будет присвоено значение).

Для инициализации переменной a необходимо написать следующий код:



Переменную можно инициализировать во время объявления:



или



Синтаксис C# позволяет объявить несколько переменных (и инициализировать их) одного типа в одной синтаксической конструкции.

Например:



В данном примере объявляется 5 переменных типа float, некоторые из них инициализируются в процессе объявления.

2.3 Объявление и инициализация констант в C#

Константа – это переменная, значение которой не меняется за время выполнения программы. Для объявления константы необходимо воспользоваться ключевым словом const. Например:



Очевидно, что при таком объявлении, поменять значения simv и pi в дальнейшем будет нельзя.

1. Оборудование и материалы

Для выполнения лабораторной работы рекомендуется использовать персональный компьютер со следующими характеристиками: 64-разрядный (x64) процессор с тактовой частотой 1 ГГц и выше, оперативная память – 1 Гб и выше, свободное дисковое пространство – не менее 1 Гб, графическое устройство DirectX 9. Программное обеспечение: операционная система WINDOWS 7 и выше, Microsoft Visual Studio 2012 и выше.

1. Методика и порядок выполнения работы

1. Создайте консольное приложение в соответствии с алгоритмом описанном в лабораторной работе №1.
2. Изучите материал в разделе «Теоретическое обоснование» данной лабораторной работы.
3. Модифицируйте исходный файл как показано на рис. 2.1.

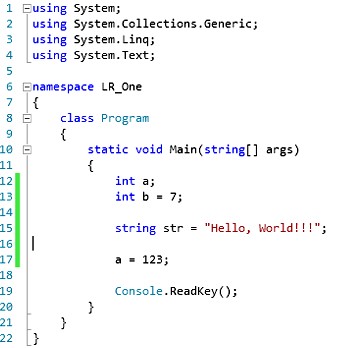


Рисунок 2.1 – Объявление переменных в C#.

1. Рассмотрим приведенный пример подробнее:
   1. В строке 12 объявляется переменная типа int с именем a.
   2. В строке 13 объявляется переменная типа int с именем b, причем при объявлении для нее устанавливается начальное значение, равное 7.
   3. В строке 15 объявляется переменная типа string с именем str и инициализируется строковым значением «Hello, World!!!».
   4. В строке 17 переменной a присваивается целочисленное значение 123.
2. Запустите приложение на выполнение. У вас должно появиться пустое окно консольного приложения. Очевидно, что исходный код, представленный на рис. 2.1 не предполагает вывода какой-либо информации на экран.
3. Для вывода информации на экран воспользуемся функцией Console.WriteLine, изученной в лабораторной работе 1. Добавим в исходный

файл следующие строки (строки 19 – 21 на рис. 2.2).



Рисунок 2.2 – Добавление строк кода для вывода значений объявленных переменных на экран.

1. В примере на рис. 2.2 используется простой вывод, то есть имя переменной просто передается в качестве параметра функции

Console.WriteLine.

1. Для выполнения форматного вывода (изменения формата представления выводимой информации) необходимо реализовать вывод в следующем виде (рис. 2.3 изменены строки 19-21):

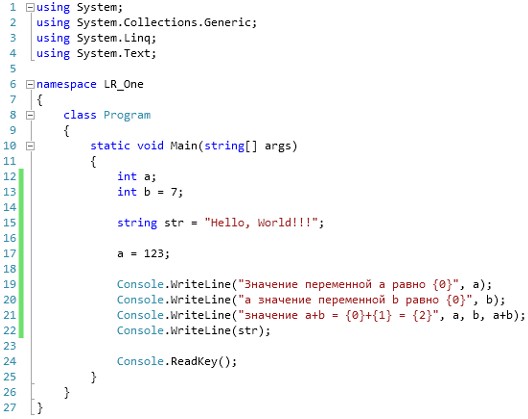
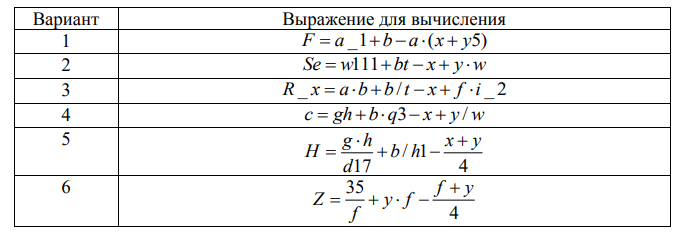


Рисунок 2.3 – Вывод информации с использованием форматных строк.

1. Внимательно изучите код полученной программы, Затем выполните индивидуальное задание.

**Индивидуальное задание.**

Объявите требуемые переменные, присвойте им начальные значения (определите самостоятельно, значения какого типа могут принимать переменные), выведите на экран с использованием форматной строки значения переменных и результат вычисления выражения в соответствие с вариантом:



7. Содержание отчета и его форма

Отчет по лабораторной работе должен содержать:

1. Номер и название лабораторной работы.
2. Цели лабораторной работы.
3. Ответы на контрольные вопросы.
4. Экранные формы и листинг программного кода, показывающие порядок выполнения лабораторной работы, и результаты, полученные в ходе её выполнения.

Отчет о выполнении лабораторной работы в письменном виде сдается преподавателю.

8. Контрольные вопросы

1. Что такое переменная? Как объявляется переменная?
2. Как объявляется константа? Чем константа отличается от переменной?
3. Какие типы значений применяются C#?
4. Чем тип char отличается от типа string?
5. Как производится инициализация переменных? Как производится инициализация констант?
6. Что такое управляющие последовательности?