**Переменная** – это именованная область памяти. В переменную можно записывать данные и считывать. Данные, записанные в переменной, называются **значением** переменной.  
  
**Си-шарп** – язык жесткой типизации. Каждая переменная должна быть определенного типа данных. Ниже, в таблице наведены встроенные типы данных языка Си-шарп:

|  |  |  |
| --- | --- | --- |
| **Тип** | **Область значений** | **Размер** |
|  |  |  |
| sbyte | -128 до 127 | Знаковое 8-бит целое |
|  |  |  |
| byte | 0 до 255 | Беззнаковое 8-бит целое |
|  |  |  |
| char | U+0000 до U+ffff | 16-битовый символ Unicode |
|  |  |  |
| bool | true или false | 1 байт\* |
|  |  |  |
| short | -32768 до 32767 | Знаковое 16-бит целое |
|  |  |  |
| ushort | 0 до 65535 | Беззнаковое 16-бит целое |
|  |  |  |
| int | -2147483648 до 2147483647 | Знаковое 32-бит целое |
|  |  |  |
| uint | 0 до 4294967295 | Беззнаковое 32-бит целое |
|  |  |  |
| long | -9223372036854775808 до 9223372036854775807 | Знаковое 64-бит целое |
|  |  |  |
| ulong | 0 до 18446744073709551615 | Беззнаковое 64-бит целое |
|  |  |  |
| float | ±1,5\*10-45 до ±3,4\*1033 | 4 байта, точность — 7 разрядов |
|  |  |  |
| double | ±5\*10-324 до ±1,7\*10306 | 8 байтов, точность — 16 разрядов |
|  |  |  |
| decimal | (-7,9 \* 1028 до 7,9 \* 1028) / (100–28) | 16 байт, точность — 28 разрядов |
|  |  |  |

\*Здесь нет ошибки. Оперативная память - массив байтов, где каждый байт имеет уникальный адрес. Для bool достаточно одного бита: 0 - false, 1 - true, но минимальная адресуемая сущность - байт, поэтому ненулевой байт считается за истину, нулевой - ложью.  
  
Для того, чтобы использовать переменную, ее сначала нужно объявить:

static void Main(string[] args)  
{  
   int a; // объявляем переменную a типа int  
   a = 5; // записываем в переменную a число 5  
   int b, c; // объявить можно сразу несколько переменных через запятую  
   bool d; // объявляем переменную d типа bool  
   d = true; // записываем в переменную d значение true (истина)  
   long e = 10; // при объявлении переменной можно сразу же задавать ей значение, это называется инициализацией  
   float f = 5.5f; // чтобы записать число с плавающей точкой типа float, нужно после значения добавлять суффикс f.  
   char g = 'g'; // объявление символьной переменой g с ее инициализацией значением символа 'g'  
}

При использовании переменной, в которую не было записано значение, компилятор выдаст ошибку "Use of unassigned local variable [variableName]".

static void Main(string[] args)  
{  
   int a;  
   Console.WriteLine(a); //ошибка  
}

Язык Си-шарп чувствительный к регистру символов. Переменные max и Max это не одно и то же. Не забывайте этого, чтобы не иметь лишних проблем.  
  
Имя переменной должно отображать суть данных, которые она отображает. Не стоит называть переменные ни о чем не говорящими именами типа a, b, c. Используйте английские слова. Высота – height, возраст – age и т. д.  
  
НИКОГДА не используйте кириллические символы в именах переменных.  
  
**Преобразование встроенных типов данных**  
  
Переменные одного типа можно преобразовывать в переменные другого типа. Преобразование бывает явным и неявным. Неявное преобразование выполняет компилятор.  
  
Пример неявного преобразования:

static void Main(string[] args)  
{  
   int a = 35;  
   short b = 10;  
   a = b; // неявное преобразование. Так как int большего размера, чем short – утери данных не будет  
   b = a; // ошибка компиляции, нельзя тип большего размера неявно преобразовать в тип меньшего размера  
}

При явном преобразовании необходимо непосредственно перед переменной, которую вы хотите преобразовать, указать в скобках тип, к которому приводится переменная.  
  
Пример явного преобразования:

static void Main(string[] args)  
{  
   int a = 35000;  
   short b = 10;  
   b = (short) a; // в этом случае уже ошибки не будет. Так как максимальное значение типа short 32767, здесь будет утеря данных.  
}

**Константы**  
  
**Константа** – это переменная, значение которой нельзя изменить. Константы используются для гарантирования того, что данные в этой переменной не изменятся.  
Для того, чтобы объявить константу, перед обычным объявлением переменной нужно добавить ключевое слово const:

static void Main(string[] args)  
{  
   const int months = 12; // объявление константы  
   months = 13; // ошибка компиляции  
}

При объявлении константы она должна обязательно быть проинициализирована значением.  
Константы также делают ваш код более красивым, читаемым.

static void Main(string[] args)  
{  
   const int months = 12;  
   const int monthSalary = 1024;  
   int yearSalary = monthSalary \* months;  
}

Гораздо понятнее чем:

static void Main(string[] args)  
{  
   int yearSalary = 12 \* 1024;  
}

Константы могут быть двух типов: простые литералы и строчные:

static void Main(string[] args)  
{  
   Console.WriteLine(100); // 100 есть 100 и этого не изменить, это константа, а точнее числовой литерал  
   Console.WriteLine("Hello!"); // строка “Hello!” является строчным литералом  
}

Здесь стоит отличать константы от переменных-констант, последние имеют имя, как в примере с месяцами и зарплатой.  
  
**Ключевое слово var**  
  
Начиная с версии C# 3.0 в язык было добавлено ключевое слово **var**, которое позволяет создавать переменные без явного указания типа данных. Тип данных такой переменной определяет компилятор по контексту инициализации.

static void Main(string[] args)  
{  
   var number = 5; // number будет типа int  
   var text ="some text"; // text будет типа string  
   var number2 = 0.5; // number2 будет типа double  
}

var сохраняет принцип строгой типизации в Си-шарп. Это означает, что после того, как для переменной уже был определен тип, в нее нельзя записать данные другого типа:

static void Main(string[] args)  
{  
   var number = 5;  
   number = "some text"; // ошибка, number определен как int  
}

Ключевое слово var следует использовать в первую очередь с LINQ выражениями (при работе с базами данных)

static void Main(string[] args)  
{  
  var query = from s in bdContext.Students select s;  
}

О LINQ мы будем говорить позже.  
  
Ключевое слово var имеет ограничения по его использованию - var не может быть в качестве:  
  
- *поля класса*  
- *аргумента функции*  
- *возвращаемого типа функции*  
- *переменной, которой присваивается null*  
  
Нововведение var является достаточно противоречивым среди разработчиков на C#, некоторые используют его где только возможно, другие его избегают (код становится плохо читаемым).  
  
**Ссылочные типы**  
  
Все типы данных, о которых мы говорили выше, являются структурными. Также существуют ссылочные типы. Из базовых типов к ссылочным относятся object и string.  
Тип object является базовым для всех остальных типов данных. Типу string соответствует строка символов Unicode.  
  
Пример использования типа [string](http://mycsharp.ru/post/16/2013_05_26_rabota_so_strokami_v_si-sharp_klass_string.html).

static void Main(string[] args)  
{  
   string hello = "Hello!";  
   Console.WriteLine(hello);  
}

Структурные типы данных в Си-шарп хранятся в стеке. Для этих данных зарезервирована область в стеке.  
  
**Стек** — это структура данных, которая сохраняет элементы по принципу «последним пришёл — первым вышел» . Примером из жизни служит стопка тарелок. Скорость работы со стеком гораздо выше, чем с оперативной памятью, следовательно, использование стека повышает скорость работы программы.  
  
Ссылочные типы хранятся в куче.  
  
**Куча** — это область динамической памяти, которая выделяется приложению для хранения данных (например объектов). Доступ к данным в куче осуществляется медленнее, чем к стеку. Переменные ссылочных типов хранят ссылки на данные.  
  
К ссылочным типам относятся классы, интерфейсы, делегаты и массивы. Больше об этом читайте в уроке [Ссылочные типы и типы значений в Си-шарп](http://mycsharp.ru/post/46/2014_08_26_ssylochnye_tipy_i_tipy_znachenij_v_si-sharp_klyuchevye_slova_ref_i_out.html).

**Практическое задание**  
  
Создайте новый проект, объявите несколько переменных различных типов, примените явное и неявное преобразование. Создайте константную переменную, попробуйте изменить ее значение.