

# Лабораторная работа №2

Типы данных. Переменные. Бинарные операции.  
Форматированный вывод.

---

▣ **Тема:**

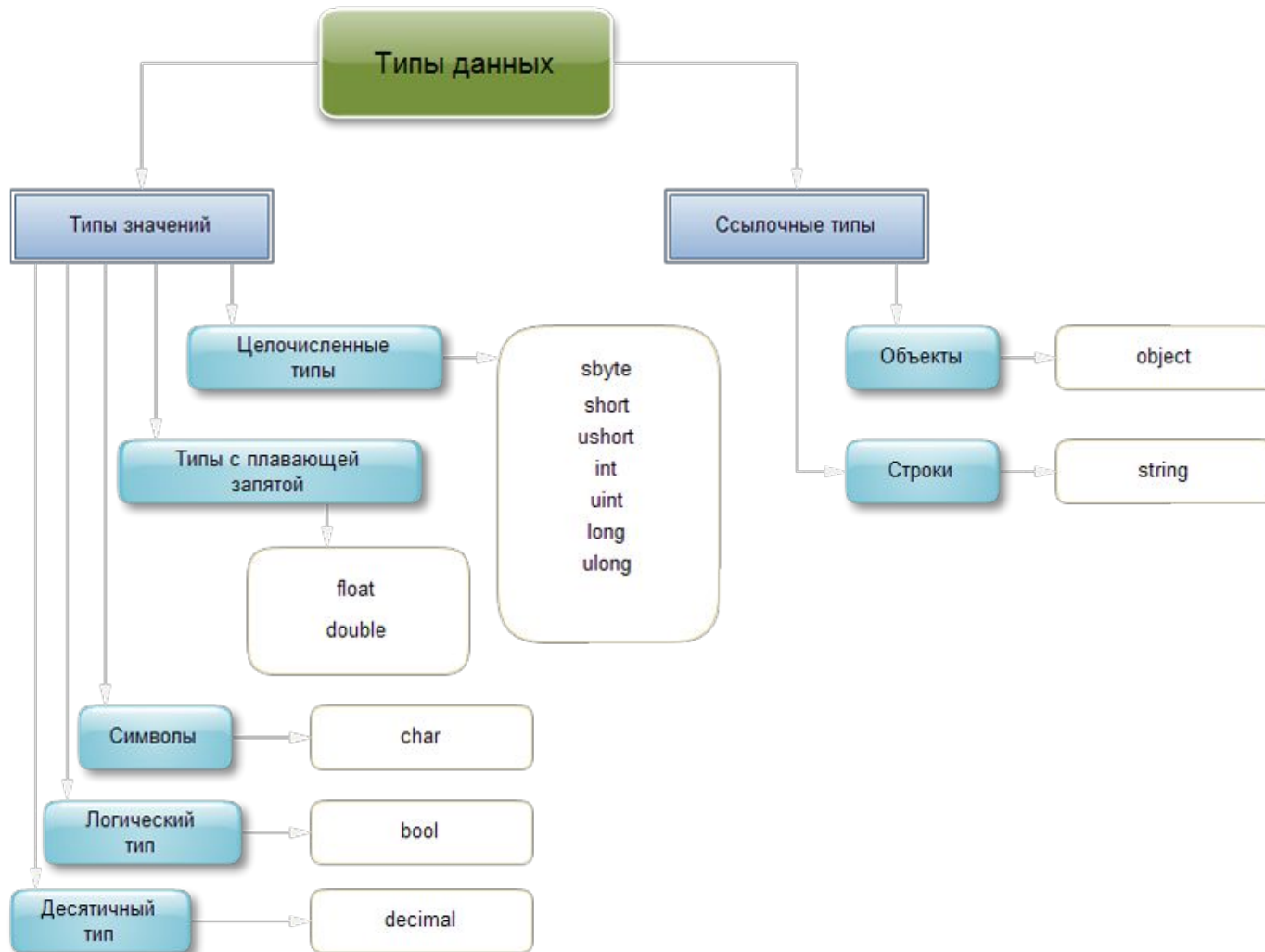
Типы данных. Переменные. Бинарные операции.  
Форматированный вывод.

▣ **Цель работы:**

Ознакомиться с типами данных C#, переменными и операциями над ними.



# Типы данных



# Переменные

---

- ❑ Переменная представляет числовое или строковое значение или объект класса.
- ❑ Значение, хранящееся в переменной, может измениться, однако имя остается прежним.
- ❑ Переменная представляет собой один тип поля.

Следующий код является простым примером объявления целочисленной переменной, присвоения ей значения и последующего присвоения нового значения.

```
int x = 1;  
x = 2;
```



# Переменные

---

- В C# переменные объявляются с определенным типом данных и надписью.
- В C# необходимо указать тип переменной: **int**, **float**, **byte**, **short** или другой из более чем 20 различных типов данных.

Тип указывает, помимо всего прочего, точный объем памяти, который следует выделить для хранения значения при выполнении приложения.

```
int answer = 42;  
string greeting = "Hello, World!";  
double bigNumber = 1e100;
```

```
System.Console.WriteLine("{0} {1} {2}", answer, greeting, bigNumber);
```



# Константы

---

- Константа является другим типом поля.
- Она хранит значение, присваиваемое по завершении компиляции программы, и никогда после этого не изменяется.
- Константы объявляются помощью ключевого слова const; их использование способствует повышению удобочитаемости кода.

```
const int speedLimit = 55;  
const double pi = 3.14159265358979323846264338327950;
```



# Константы

---

- Переменная readonly аналогична константе, однако значение ей присваивается при запуске программы.
- Это дает возможность задать значение на основе каких-либо других условий, неизвестных до выполнения программы.
- Однако после первого присваивания значение не может быть снова изменено пока выполняется программа.



# Приведение и преобразование типов

---

- Поскольку в C# тип определяется статически тип во время компиляции, после объявления переменной, она не может быть объявлена вновь или использоваться для хранения значений другого типа, если этот тип не преобразуется в тип переменной.

```
int i;  
i = "Hello"; // Error: "Cannot implicitly convert type 'string' to 'int'"
```





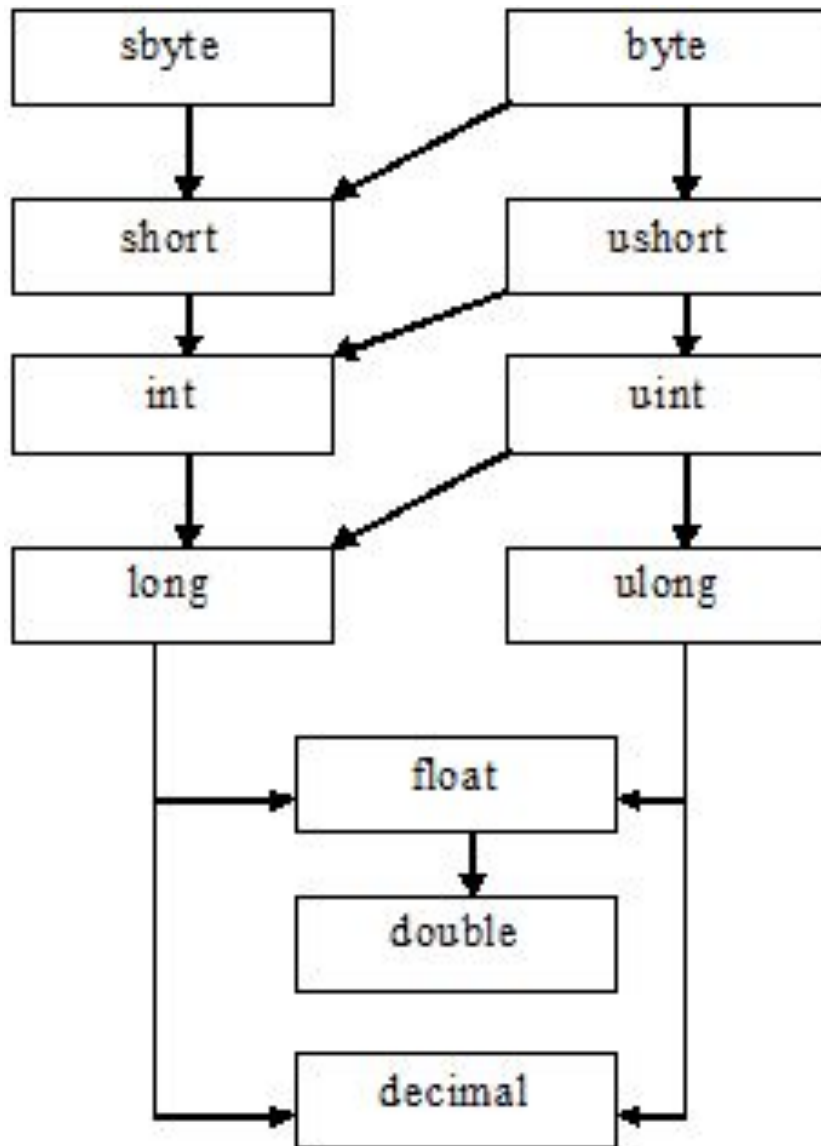
# Неявные преобразование типов

---

- Для встроенных числовых типов неявное преобразование можно выполнить, когда сохраняемое значение может уместиться в переменной без отбрасывания дробной части или округления до ближайшего целого.

```
int num = 2147483647;  
long bigNum = num;
```





## Преобразование типов

На схеме перечислены все арифметические типы.

Стрелками указаны направления, по которым автоматически осуществляется неявное преобразование

(например, из `byte` в `double`).

Любое обратное приведение (против стрелок) должно быть явным!!!



# Операция «приведение к типу»

---

- Эта операция используется тогда, когда необходимо преобразовать значение одного типа в значение другого типа.
- Это так называемое явное преобразование типов. Операция задаётся указанием имени типа в круглых скобках.
- Например, `(byte)` – преобразовать в тип байт. Рассмотрим пример:

```
int n = 10;  
double z;  
z = n;  
n = (int) z;
```



```
class Test
{
    static void Main()
    {
        double x = 1234.7;
        int a;
        // Cast double to int.
        a = (int)x;
        System.Console.WriteLine(a);
    }
}
// Output: 1234
```

## Явное преобразование типов

Если преобразование не может быть выполнено без риска потери данных, для компилятора требуется, чтобы пользователь выполнил явное преобразование, которое называется приведением.

Для выполнения приведения заключите тип, в который производится приведение, в скобки перед преобразуемым значением или переменной.



# Строковые преобразования. Класс Convert

---

- Не существует явного преобразования арифметических типов в строки! Операция «приведение к типу» здесь невозможна. Однако, в базовом классе `Object` имеется и определён метод `ToString`.

```
string s;  
int n = 2010;  
s = "Год "+ToString(n);
```



# Строковые преобразования. Класс Convert

---

- Достаточно часто требуется обратное преобразование – строку в число.
- Для таких преобразований предусмотрен специальный класс методов – класс Convert, встроенный в пространство имен System.
- Этот класс содержит 15 статических методов вида To (ToBoolean, ... ToInt64), где Type может принимать значения от Boolean до UInt64.

```
System.DateTime dat = Convert.ToDateTime("15.03.2003");  
Console.WriteLine("Date = {0}", dat);
```

- Результатом вывода будет строка: Date = 15.03.2003 0:00:00
- 



# Форматированный вывод

---

- Строка составного формата состоит из нуля или более фиксированного текста перемежаемых одним или несколькими элементами форматирования.
- Во время выполнения, каждый элемент форматирования заменяется строковым представлением соответствующего аргумента в списке параметров.

```
DateTime dat = new DateTime(2012, 1, 17, 9, 30, 0);  
string city = "Chicago";  
int temp = -16;  
string output = String.Format("At {0} in {1}, the temperature was {2} degrees.",  
                               dat, city, temp);  
Console.WriteLine(output);
```



# Арифметические операции

---

Операция	Назначение	Пример
+	<i>Сложение</i>	$X = x + y;$
-	<i>Вычитание</i>	$X = x - y;$
*	<i>Умножение</i>	$Z = x * y;$
/	<i>Деление</i>	$Z = x / y;$
%	<i>Деление по модулю</i>	$Z = t \% e;$





# Операции отношения

---

- Все операции отношения используются для сравнения значений переменных или выражений.

Операция	Назначение	Пример
$==$	<i>Равно</i>	$I == 0$
$!=$	<i>Не равно</i>	$K != 15$
$>$	<i>Больше, чем</i>	$Z > 15.2$
$<$	<i>Меньше, чем</i>	$Tab < -132.654$
$>=$	<i>Больше или равно</i>	$Z11 >= 0$
$<=$	<i>Меньше или равно</i>	$Y2 <= 10$



# Логические операции

---

Операция	Назначение	Пример
<b>&amp;&amp;</b>	<i>Логическое И</i>	$(R > 2) \ \&\& \ (R < 20)$
<b>  </b>	<i>Логическое ИЛИ</i>	$(L = 12) \    \ (L > 15)$
<b>!</b>	<i>Логическое НЕ</i>	$\text{Tab} \ != \ 13$



## Задания на лабораторную работу №2

---

1. Даны два десятичных числа введенные с клавиатуры. Вычислить их сумму, произведение, разность, частное. Результат вывести на экран с соответствующими пояснениями.
2. Даны два числа, введенные с клавиатуры. Вычислить частное этих чисел. Результат вывести на экран с соответствующими пояснениями.
3. Дано ребро куба, введенное с клавиатуры. Вычислить объем куба и площадь его боковой поверхности.
4. Даны два действительных положительных числа, введенные с клавиатуры. Найти их среднее арифметическое и среднее геометрическое.
5. Даны катеты прямоугольного треугольника, введенные с клавиатуры. Вычислить и вывести на экран длину гипотенузы и площадь.



## Задания на лабораторную работу №2

---

6. Определить и вывести на экран периметр правильного  $n$ -угольника, описанного около окружности радиуса  $r$ . Радиус и количество углов вводится с клавиатуры.
7. Даны три сопротивления включенные параллельно. Вычислить и вывести на экран сопротивление соединения.
8. Даны значения гипотенуза и катета прямоугольного треугольника, вводимые с клавиатуры. Вычислить и вывести на экран значения второго катета и радиуса вписанной окружности.
9. Дана длина окружности, введенной с клавиатуры. Вычислить площадь круга, ограниченного этой окружностью.
10. Треугольник задан размерами углов и радиусом описанной окружности, вводимыми с клавиатуры. Вычислить и вывести на экран значения размеров его сторон.
11. Вычислить и вывести на экран период колебания маятника заданной длиной .



# Отчет по лабораторной работе

---

- Титульный лист
- Задание (со своими данными)
- Теория (описание используемой среды разработки и методов для написания программы)
- Результат выполнения (скриншот лаб.работы)
- Выводы
- Листинг программы.

