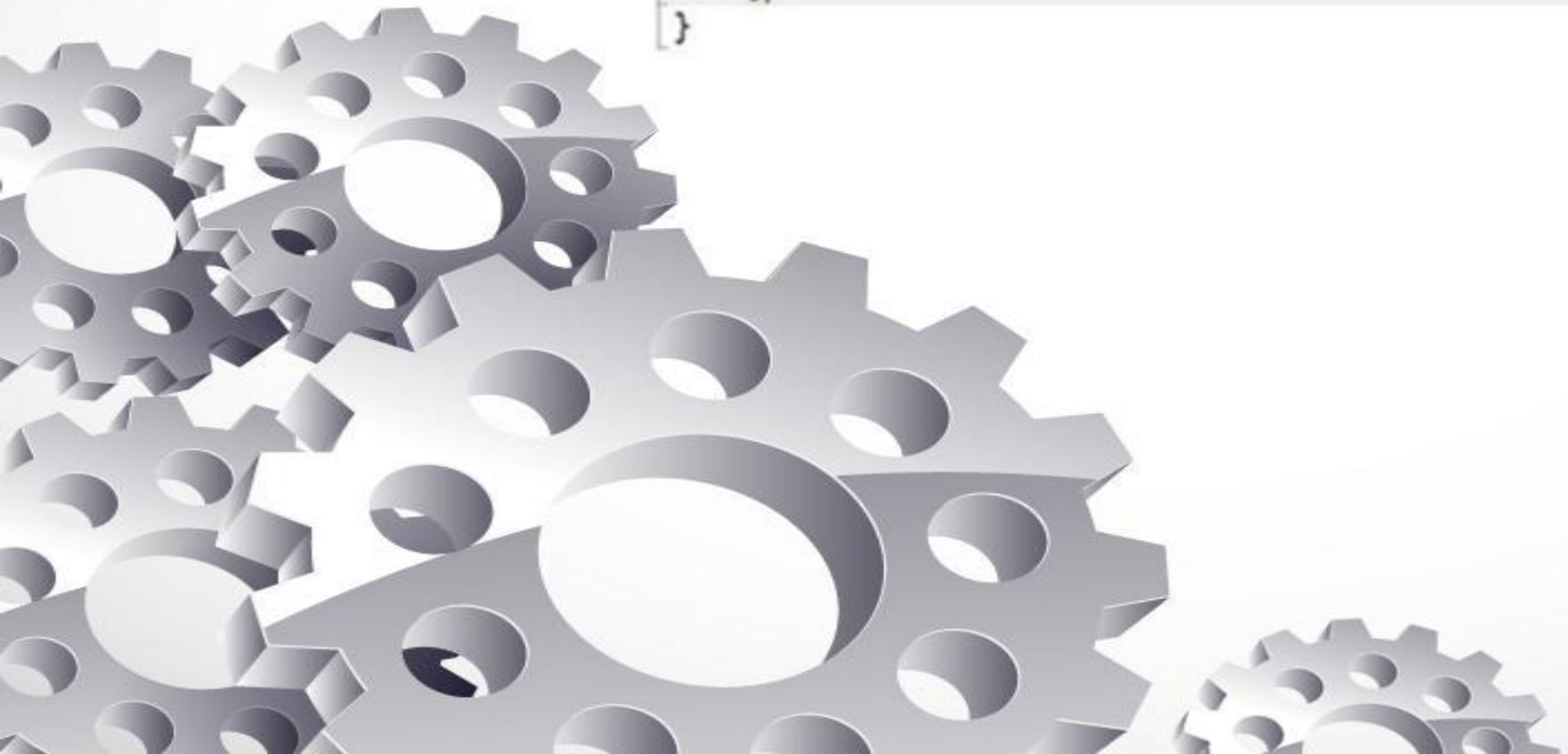
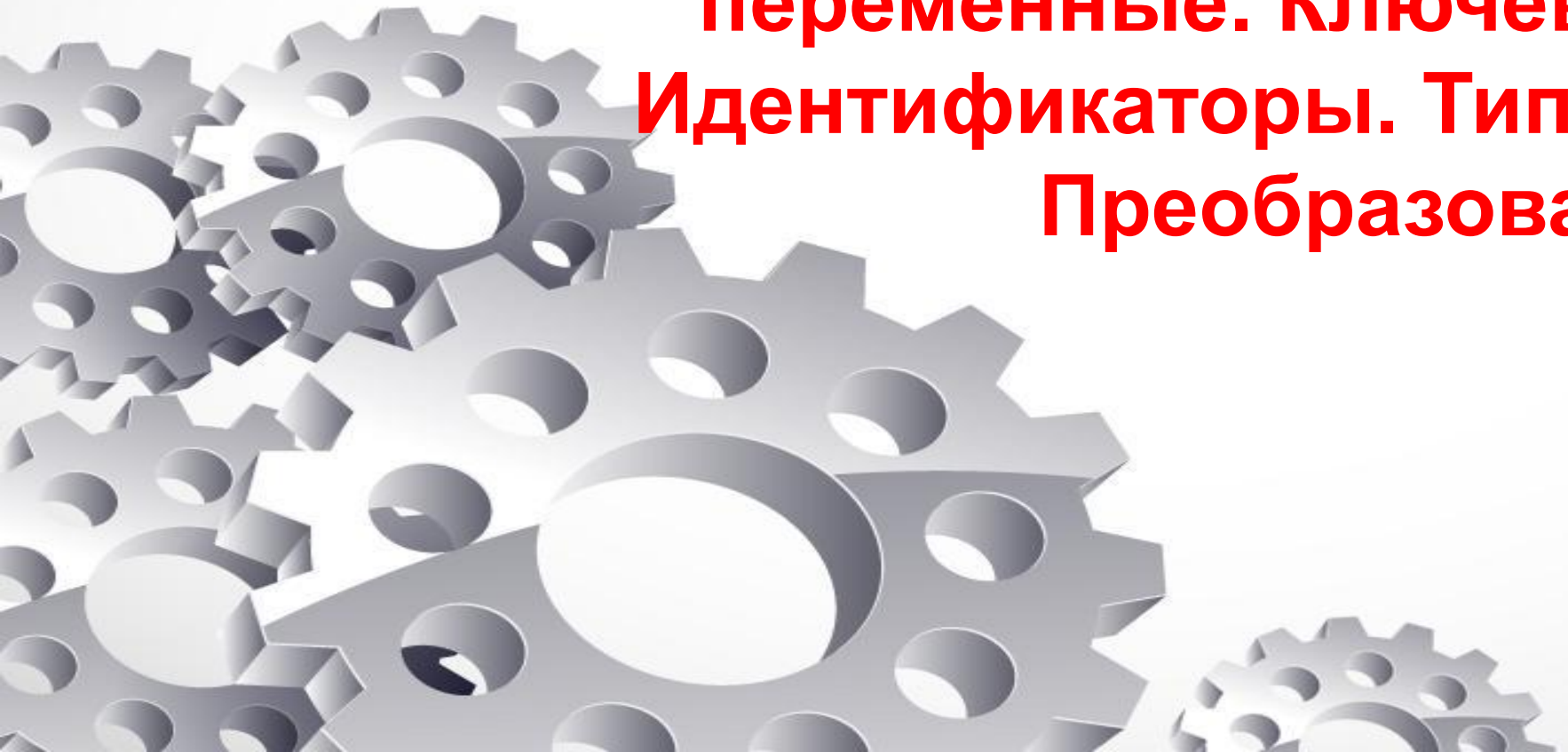
```
using System;

namespace ConsoleApplication2
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Hello, World!");
        }
    }
}
```

"Hello, world!" на C#



Введение в C#. Константы и переменные. Ключевые слова. Идентификаторы. Типы данных. Преобразование типов



Введение

.NET Framework – это платформа, созданная Microsoft для разработки приложений. Microsoft Visual Studio — линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств.

Язык программирования C#

Язык программирования C# был разработан Microsoft на базе языков C и C++ специально для работы с платформой .NET.

На языке C# можно писать приложения:

- Windows-приложения (например, Microsoft Office);
- Web-приложения;
- Web-службы.

Комментарии

В C# различают:

- многострочные комментарии

/*многострочный комментарий*/

- однострочные комментарии

//однострочный комментарий

- Специальный комментарий: извлекает текст из комментария и создает специальный текстовый файл. Эти комментарии должны быть составлены по правилам XML-документации

- ///специальный комментарий

Простая программа на C#

```
//Подключение пространства имен System
using System;

namespace ConsoleApplication1//Название пространства имен
{//начало пространства имен
    class Program  //название класса
    {//начало класса
        static void Main(string[] args)
        {//начало программы
            Console.WriteLine("Hello, world!");//Тело программы
        } //конец программы
    } //конец класса
} //конец пространства имен
```

Простейшая программа на C# (консольная)

```
using System;
class MyClass
{
    static void Main()
    {
        //Вывод данных в консольном режиме
        Console.WriteLine("Hello, world!");
        Console.ReadKey();//Пауза в конце программы
    }
}
```

Идентификаторы

Имена, или идентификаторы, служат для того чтобы обращаться к программным объектам и различать их.

В идентификаторе могут использоваться:

- буквы;
- цифры;
- символ подчеркивания.

Прописные и строчные буквы различаются.

Пример: hacker, Hacker и hAcKeR — три разных имени.

Идентификаторы

Первым символом идентификатора может быть буква или знак подчеркивания, но не цифра.

Длина идентификатора не ограничена.

Пробелы внутри имен не допускаются.

В идентификаторах C# разрешается использовать буквы национальных алфавитов.

Пример: Фёкла, сумма и т.д.

Также можно представлять символ с помощью его кода в шестнадцатеричном виде с префиксом \u, например, \u00F2.

Ключевые слова

Ключевые слова — это зарезервированные идентификаторы, которые имеют специальное значение для компилятора. Их можно использовать только в том смысле, в котором они определены.

abstract	as	base	bool	break
byte	case	catch	char	checked
class	const	continue	decimal	default
delegate	do	double	else	enum
event	explicit	extern	false	finally
fixed	float	for	foreach	goto
if	implicit	in	int	interface
internal	is	lock	long	namespace
new	null	object	operator	out
override	params	private	protected	public
readonly	ref	return	sbyte	sealed
short	sizeof	stackalloc	static	string
struct	switch	this	throw	true
try	typeof	uint	ulong	unchecked
unsafe	ushort	using	virtual	void
volatile	while			

В C# константы (литераты) объявляются:

const <тип данных> <имя константы> = <значение константы>;

Например:

```
const int x = 55;
const double pi = 3.14159265358979323846;
```

Константы

Константы в C# бывают:

- логические;
- целые;
- вещественные;
- символьные;
- строковые;
- константа null (ссылка, которая не указывает ни на какой объект).

К константам относят **управляющие последовательности:**

Вид	Наименование
\a	Звуковой сигнал
\b	Удаляет символ перед управляющей последовательностью
\n	Переход на новую строку
\r	Вывод с того места, где стоит управляющая последовательность
\t	Горизонтальная табуляция
\v	Вертикальная табуляция
\\	Обратная косая черта
\'	Апостроф
\"	Кавычка
\0	*Нуль-символ (пробел)*

Переменные

В C# переменные объявляются:

<тип данных> <имя переменной>;

- **Например:**
- int x;
- float a,b;
- int z=10, y=15;

Типы данных

К простым типам относятся типы вроде числовых и булевских значений.

Целочисленные типы	
Тип	Допустимые значения
sbyte	Целые числа от -128 до 127
byte	Целые числа от 0 до 255
short	Целые числа от -32768 до 32767
ushort	Целые числа от 0 до 65535
int	Целые числа от -2147483648 до 2147483647
uint	Целые числа от 0 до 4294967295
long	Целые числа от -9223372036854775808 до -9223372036854775807
ulong	Целые числа от 0 до 18446744073709551615

Типы данных

Типы переменных для хранения чисел с плавающей запятой (вещественные)		
Тип	Приблиз. мин. значение	Приблиз. макс. значение
float	1.5×10^{-45}	3.4×10^{38}
double	5.0×10^{-324}	1.7×10^{308}
decimal	1.0×10^{-28}	7.9×10^{28}

Нечисловые простые типы	
Тип	Допустимые значения
char	Одиночный символ Unicode, хранимый в виде целого числа от 0 до 65535
bool	Булево значение – true либо false
string	Последовательность символов

Математические операции

Операция	Категория	Пример	Результат
+	Бинарная	v1=v2+v3	v1 присваивается значение – результат сложения v2 и v3
-	Бинарная	v1=v2-v3	v1 присваивается значение – результат вычитания v2 и v3
*	Бинарная	v1=v2*v3	v1 присваивается значение – результат умножения v2 и v3
/	Бинарная	v1=v2/v3	v1 присваивается значение – результат деления v2 на v3
%	Бинарная	v1=v2%v3	v1 присваивается значение – остаток от деления v2 на v3
+	Унарная	v1=+v2	v1 присваивается значение v2
-	Унарная	v1=-v2	v1 присваивается значение v2, умноженное на -1

Операции инкремента и декремента

Операция	Категория	Пример	Результат
++	Унарная	v1=++v2	v1 присваивается значение – v2+1. v2 увеличивается на 1
--	Унарная	v1=--v2	v1 присваивается значение – v2-1. v2 уменьшается на 1
++	Унарная	V1=v2++	v1 присваивается значение – v2. v2 увеличивается на 1
--	Унарная	v1=v2--	v1 присваивается значение – v2. v2 уменьшается на 1

Математические функции

В C# для работы с математическими функциями необходимо обратиться к классу Math, при этом будут доступны следующие методы (функции):

- Abs(x) – возвращает модуль числа;
- Acos(x) - возвращает угол, косинус которого равен указанному числу;
- Asin(x) - возвращает угол, синус которого равен указанному числу;
- Atan(x) - возвращает угол, тангенс которого равен указанному числу;

- Atan2(x, y) – возвращает угол, тангенс которого равен отношению двух указанных чисел;
- Ceiling(x) - возвращает наименьшее целое число, которое больше или равно заданному десятичному числу;
- Cos(x) - возвращает косинус указанного угла;
- Exp(x) – возвращает значение e, возведенное в указанную степень;
- Floor(x) - возвращает наибольшее целое число, которое меньше или равно указанному десятичному числу;

- Log(x) - возвращает натуральный логарифм (с основанием e) указанного числа;
- Log(x, y) - возвращает логарифм числа x по основанию y;
- Log10(x) – возвращает логарифм с основанием 10 указанного числа;
- Max(x, y) - возвращает большее из двух чисел;
- Min(x, y) - возвращает меньшее из двух чисел;
- Pow(x, y) - возвращает указанное число, возведенное в указанную степень;

- Round(x) - округляет десятичное значение до ближайшего целого;
- Round(x, y) - округляет десятичное значение x до указанного числа дробных разрядов y (y – целое число);
- Sign(x) – возвращает значение, определяющее знак десятичного числа;
- Sin(x) - возвращает синус указанного угла;
- Sqrt(x) - возвращает квадратный корень из указанного числа;
- Tan(x) - возвращает тангенс указанного угла;
- Truncate(x) - вычисляет целую часть заданного числа.

Операторы консольного вывода

- Console.Write(“текст”);
- Console.WriteLine(“текст”);
- Console.WriteLine(a);
- Console.WriteLine(a+b);
- Console.WriteLine(“a=”+a);
- Console.WriteLine(“a и b =”+a+” “+b);
  Console.WriteLine(“a={0}”,a);
- Console.WriteLine(“a={0}, b = {1}”,a, b);

Форматированный вывод

- Console.WriteLine("{0,8:0.00}", 123.4567); // " 123.46"
- Console.WriteLine("{0:0.00}", 123.4); // "123.40“
- Console.WriteLine("{0:0.##}", 123.4567); // "123.46"
- Console.WriteLine("{0:0.##}", 123.4); // "123.4“
- Console.WriteLine("{0:00.0}", 123.4567); // "123.5"
- Console.WriteLine("{0:00.0}", 23.4567); // "23.5"
- Console.WriteLine("{0:00.0}", 3.4567); // "03.5"
- Console.WriteLine("{0:00.0}", -3.4567); // "-03.5"
- Console.WriteLine("{0:#.0}", 123.12345); // "123.1"
- Console.WriteLine("{0:##.0}", 123.12345); // "123.1"

Операторы консольного ввода

Console.Read();

Console.ReadLine();

Console.ReadKey();

Пример:

```
string b;
b= Console.ReadLine();
```

либо

```
string b = Console.ReadLine();
```

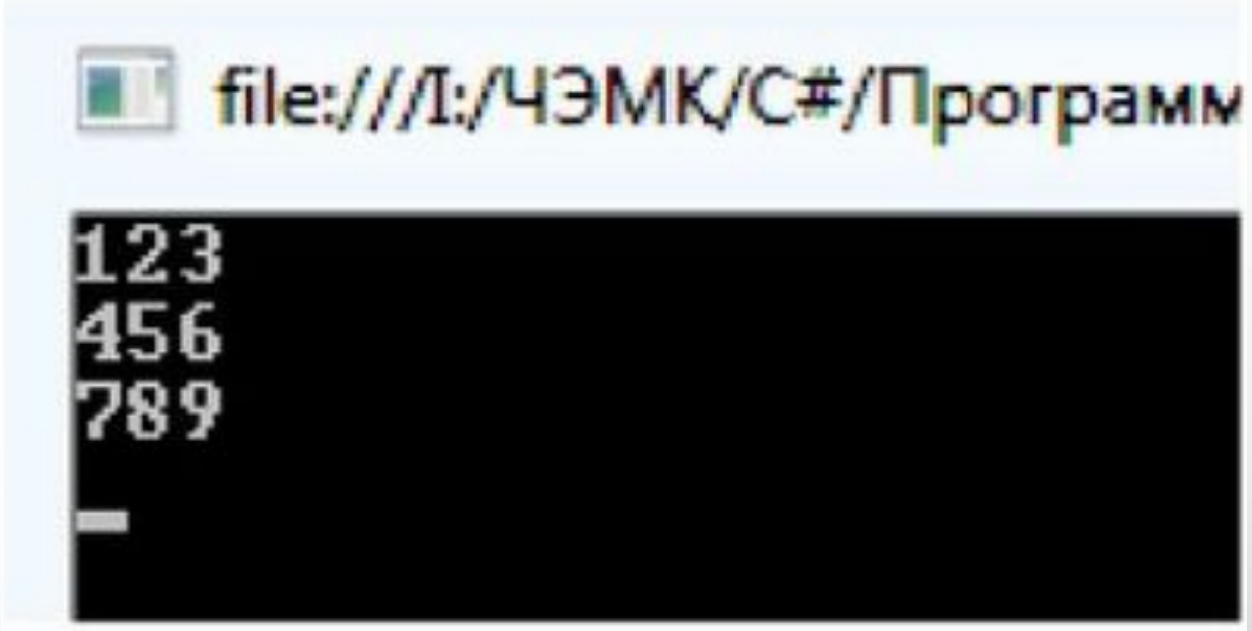
Данная запись возможна только для строкового типа, для других необходимо преобразование типов.

Операторы консольного ввода

Строку также можно выводить буквально или дословно.

Пример:

```
Console.WriteLine(@”123 456
789”);
```



Преобразование типов

- **Неявное преобразование:** применяется, когда преобразование из типа A в тип B возможно при любых обстоятельствах, а правила выполнения преобразования достаточно просты для того, чтобы доверить их компилятору.
- **Явное преобразование:** применяется, когда преобразование из типа A в тип B возможно только при определенных обстоятельствах или когда правила преобразования довольно сложны и требуют дополнительной обработки.

Неявные преобразования

Неявные числовые преобразования	
Тип	Может быть безопасно преобразован в тип
byte	short, ushort, int, uint, long, ulong, float, double, decimal
sbyte	short, int, long, float, double, decimal
short	int, long, float, double, decimal
ushort	int, uint, long, ulong, float, double, decimal
int	long, float, double, decimal
uint	long, ulong, float, double, decimal
long	float, double, decimal
ulong	float, double, decimal
float	double
char	ushort, int, uint, long, ulong, float, double, decimal

Явные преобразования

- **1 способ:** приведение переменной (целевой тип) исходная переменная

Пример: (float) a;

2 способ: приведение с помощью команд Convert

Команда	Результат
Convert.ToBoolean(val)	val, преобразованное в bool
Convert.ToByte(val)	val, преобразованное в byte
Convert.ToChar(val)	val, преобразованное в char
Convert.ToDecimal(val)	val, преобразованное в decimal
Convert.ToDouble(val)	val, преобразованное в double
Convert.ToInt16(val)	val, преобразованное в short
Convert.ToInt32(val)	val, преобразованное в int
Convert.ToInt64(val)	val, преобразованное в long
Convert.ToSByte(val)	val, преобразованное в sbyte
Convert.ToSingle(val)	val, преобразованное в float
Convert.ToStringval)	val, преобразованное в string
Convert.ToUInt16(val)	val, преобразованное в ushort
Convert.ToUInt32(val)	val, преобразованное в uint
Convert.ToUInt64(val)	val, преобразованное в ulong

Задание 1

Создать новое консольное приложение вывода целого числа и строки.

Задание 2

Создать новое консольное приложение вычисления значения выражения:

$$e = \frac{a + b + c}{d}$$

если:

1) a, b, c, d – вещественные;

2) a, b, c, d – целые числа.

Задание 3

Создать новое консольное приложение вычисления значения выражения:

$$b = a + \cfrac{1}{1 - \cfrac{1}{a + \cfrac{1}{a}}}$$

(переменная a вводится с клавиатуры, для проверки введите a=1)

Самостоятельно

1. Вычислить значение выражения (a и b вводятся с клавиатуры).

$$\sqrt[3]{\frac{a^4 \cdot \sqrt[3]{b^9}}{a^{-2}}}$$

2. Вычислить значение выражения (величина угла вводится с клавиатуры).

$$\left(1+\operatorname{ctg}^2\alpha+\frac{1}{\operatorname{ctg}^2\alpha}\right)\cdot\sin^2\alpha\cdot\cos^2\alpha$$

Самостоятельно

3. Вычислить значение выражения.

$$\log_5 0{,}2 + \log_{0{,}5} 4$$

4. Вычислить значение выражения.

$$log_2(log_2 256) + log_6 2 + log_6 3 + \frac{log_5 729}{log_{0.2} 27}$$