

Основы программирования

# Синтаксис языка C#

Поразрядные операции

# Побитовые операторы и операторы сдвига

Действуют непосредственно на разряды своих операндов целочисленных или символьных типов.

- Побитовые операторы
- Операторы сдвига

Предназначены для тестирования, установки или сдвига битов (разрядов).

# Побитовые операторы

**&** (логическое умножение)

Умножение производится поразрядно, и если у обоих операндов значения разрядов равно 1, то операция возвращает 1, иначе возвращается число 0.

Например:

```
int x1 = 2;           //010
int y1 = 5;           //101
Console.WriteLine(x1 & y1); // выведет 0
int x2 = 4;           //100
int y2 = 5;           //101
Console.WriteLine(x2 & y2); // выведет 4
```

# Побитовые операторы

| (логическое сложение)

Операция также производится по двоичным разрядам, но возвращается единица, если хотя бы у одного числа в данном разряде имеется единица.

Например:

```
int x1 = 2;           //010
int y1 = 5;           //101
Console.WriteLine(x1 | y1); // выведет 7 - 111
int x2 = 4;           //100
int y2 = 5;           //101
Console.WriteLine(x2 | y2); // выведет 5 - 101
```

# Побитовые операторы

- ^ (логическое исключающее ИЛИ)  
Эту операцию называют XOR, нередко ее применяют для простого шифрования.

Производятся поразрядные операции. Если у нас значения текущего разряда у обоих чисел разные, то возвращается 1, иначе возвращается 0.

# Побитовые операторы

```
int x = 45;
    // Значение, которое надо зашифровать - в
    // двоичной форме 101101

int key = 102;
    //Пусть это будет ключ - в двоичной форме 1100110

int encrypt = x ^ key;
    //Результатом будет число 1001011 или 75

Console.WriteLine("Зашифрованное число: " + encrypt);

int decrypt = encrypt ^ key;
    // Результатом будет исходное число 45

Console.WriteLine("Расшифрованное число: " + decrypt);
```

# Побитовые операторы

- ~ (логическое отрицание или инверсия)  
инвертирует все разряды: если значение разряда равно 1, то оно становится равным нулю, и наоборот.

Например:

```
int x = 12;                // 00001100
Console.WriteLine(~x);     // 11110011 или -13
```

# Побитовые операторы

1101 0011	1101 0011	1101 0011	
1001 1001	1001 1001	1001 1001	1101 0011
&		^	~
_____	_____	_____	_____
1001 0001	1101 1011	0100 1010	0010 1100



# Операторы сдвига

Также производятся над разрядами чисел. Сдвиг может происходить вправо и влево.

- $x \ll y$  - сдвигает число  $x$  влево на  $y$  разрядов.  
Например,  $4 \ll 1$  сдвигает число 4 (которое в двоичном представлении 100) на один разряд влево, то есть в итоге получается 1000 или число 8 в десятичном представлении.
- $x \gg y$  - сдвигает число  $x$  вправо на  $y$  разрядов.  
Например,  $16 \gg 1$  сдвигает число 16 (которое в двоичном представлении 10000) на один разряд вправо, то есть в итоге получается 1000 или число 8 в десятичном представлении.

# Операторы сдвига

Таким образом, если исходное число, которое надо сдвинуть в ту или другую сторону, делится на два, то фактически получается умножение или деление на два.

Поэтому подобную операцию можно использовать вместо непосредственного умножения или деления на два.

# Приоритет поразрядных операций

в порядке убывания приоритета:

- Оператор побитового дополнения  $\sim$
- Операторы сдвига  $\ll$  и  $\gg$
- Оператор логического И  $\&$
- Оператор логического исключающего ИЛИ  $\wedge$
- Оператор логического ИЛИ  $|$