Лекция 11

Операции присваивания, условная операция. Приоритеты операций и порядок вычислений

Операция присваивания

переменная = выражение;

Механизм выполнения операции присваивания: вычисляется выражение и его результат заносится в память по адресу, который определяется именем переменной, находящейся слева от знака операции. То, что ранее хранилось в этой области памяти, теряется.

```
a = b + c / 2;
x = 1;
x = x + 0.5;
```

Для правого операнда операции присваивания должно существовать неявное преобразование к типу левого операнда.

Операция присваивания

Множественное присваивание выполняет присваивание значений сразу нескольким переменным одновременно:

```
int a, b, c;
a = b = c = 34;
```

Операции присвоения имеют низкий приоритет: вначале будет вычисляться значение правого операнда и только потом будет идти присвоение этого значения левому операнду.

```
int a, b, c;
a = b = c = 34 * 2 / 4;   //  17
```

Операция присваивания

В языке **C#** существуют **сложные (составные, сокращенные) формы операции присваивания**, представляющие собой комбинации операции присваивания и арифметических операторов.

Выражение вида

*переменная = переменная **оператор** выражение*

можно переписать следующим образом:

*переменная **оператор**= выражение*

Оператор x = x+10;

аналогичен оператору x += 10;

Сложные операции присваивания

В **сложных (составных) операциях присваивания** (+=, *=, /= и т п.) при вычислении выражения, стоящего в правой части, используется значение из левой части. Например, при *сложении с присваиванием* ко второму операнду прибавляется первый, и результат записывается в первый операнд, то есть выражение **a += b** является более компактной записью выражения **a = a + b**.

Выражение составного присваивания с точки зрения реализации не эквивалентно простому (первому) присваиванию, так как в первом к переменной приходится обращаться дважды.

Сложные операции присваивания

+=: присваивание после сложения (сложение с присваиванием).

A += B эквивалентно **A = A + B**

-=: присваивание после вычитания (вычитание с присваиванием).

A -= B эквивалентно **A = A - B**

***=**: присваивание после умножения (умножение с присваиванием).

A *= B эквивалентно **A = A * B**

/=: присваивание после деления (деление с присваиванием).

A /= B эквивалентно **A = A / B**

Сложные операции присваивания

%=: присваивание после деления по модулю (деление по модулю с присваиванием).

A %= B эквивалентно **A = A % B**

&=: присваивание после поразрядной конъюнкции (поразрядная конъюнкция с присваиванием).

A &= B эквивалентно **A = A & B**

|=: присваивание после поразрядной дизъюнкции (поразрядная дизъюнкция с присваиванием).

A |= B эквивалентно **A = A | B**

Сложные операции присваивания

^=: присваивание после операции исключающего ИЛИ (исключающее ИЛИ с присваиванием).

A ^= B эквивалентно **A = A ^ B**

<<=: присваивание после сдвига разрядов влево (сдвиг разрядов влево с присваиванием).

A <<= B эквивалентно **A = A << B**

>>=: присваивание после сдвига разрядов вправо (сдвиг разрядов вправо с присваиванием).

A >>= B эквивалентно **A = A >> B**

Сложные операции присваивания

Результатом операции сложного присваивания является значение, записанное в левый операнд.

Операции присваивания *правоассоциативны*, то есть выполняются справа налево, в отличие от большинства других операций (**a = b = c** означает **a = (b = c)**).

У составных операторов присваивания имеются два главных преимущества. Во-первых, они более компактны, чем их "несокращенные" эквиваленты. И во-вторых, они дают более эффективный исполняемый код, поскольку левый операнд этих операторов вычисляется только один раз.

Сложные операции присваивания

```
int a = 10;
a += 10;      // 20
a -= 4;       // 16
a *= 2;       // 32
a /= 8;       // 4
a <<= 4;      // 64
a >>= 2;      // 16
```

Условная операция (тернарный оператор)

Выражение__1 ? Выражение_2: Выражение_3;

Операция "?" выполняется следующим образом: сначала вычисляется *Выражение__1,* которое должно быть типа **bool**. *Если оно истинно,* вычисляется *Выражение_2,* и его значение становится результатом всей операции. Если *Выражение_1* ложно, вычисляется *Выражение_3,* и его значение становится результатом операции. *Выражение__1* и *Выражение_2* могут иметь разные типы. Тип результата зависит от возможности преобразования типов этих выражений.

Условная операция (тернарный оператор)

```
double yv2; // результат может быть только
            // вещественным в данном примере
int xv;
xv = 10;
yv2 = (xv > 9) ? 100 : 2.0;
Console.WriteLine("т. к. "+ xv+" > 9, то рез-т = " + yv2);
xv = 1;
yv2 = (xv > 9) ? 100 : 2.0;
Console.WriteLine("т. к. " +xv+ " < 9, то рез-т = " + yv2);
```

```
x = 10, y;
if (x>9)   y  =   100;
else y  =  200;
```

Приоритет	**Знак операции**	**Типы операции**	Порядок выполнения
1	( ) [ ] . ->	Выражение	Слева направо
2	- ~ ! * & ++ -- sizeof приведение типов	Унарные	Справа налево
3	* / %	Мультипликативные	Слева направо
4	+ -	Аддитивные	
5	<< >>	Сдвиг	
6	< > <= >=	Отношение	
7	== !=	Отношение (равенство)	
8	&	Поразрядное И	
9	^	Поразрядное исключающее ИЛИ	
10	|	Поразрядное ИЛИ	
11	&&	Логическое И	
12	||	Логическое ИЛИ	
13	? :	Условная	
14	= *= /= %= += -= &= |= >>= <<= ^=	Простое и составное присваивание	Справа налево

Основные поля и статические методы класса Math			
Имя	**Описание**	**Результат**	**Пояснения**
Abs	Модуль	Перегружен	$\lvert x \rvert$ записывается как Abs(x)
Acos	Арккосинус	double	Acos(double x)
Asin	Арксинус	double	Asin(double x)
Atan	Арктангенс	double	Atan2(double x, double y) — угол, тангенс которого есть результат деления y на x
BigMul	Произведение	long	BigMul(int x, int y)
Ceiling	Округление до большего целого	double	Ceiling(double x)
Cos	Косинус	double	Cos(double x)
Cosh	Гиперболический косинус	double	Cosh(double x)
DivRem	Деление и остаток	Перегружен	DivRem(x, y, rem)

Основные поля и статические методы класса Math			
Имя	**Описание**	**Результат**	**Пояснения**
E	База натурального логарифма (число e)	double	2,71828182845905
Exp	Экспонента	double	e *x* записывается как Exp(x)
Floor	Округление до меньшего целого	double	Floor(double x)
IEEERemainder	Остаток от деления	double	IEEERemainder(double x, double y)
Log	Натуральный логарифм	double	$\log_e x$ записывается как Log(x)
Log10	Десятичный логарифм	double	$\log_{10} x$ записывается как Log10(x)
Max	Максимум из двух чисел	Перегружен	Max(x, y)
Min	Минимум из двух чисел	Перегружен	Min(x, y)

Основные поля и статические методы класса Math			
Имя	**Описание**	**Результат**	**Пояснения**
PI	Значение числа π	double	3,14159265358979
Pow	Возведение в степень	double	x^y записывается как Pow(x, y)
Round	Округление	Перегружен	Round(3.1) даст в результате 3 Round (3.8) даст в результате 4
Sign	Знак числа	int	Аргументы перегружены
Sin	Синус	double	Sin(double x)
Sinh	Гиперболический синус	double	Sinh(double x)
Sqrt	Квадратный корень	double	$\sqrt{x}$ записывается как Sqrt(x)
Tan	Тангенс	double	Tan(double x)
Tanh	Гиперболический тангенс	double	Tanh(double x)

Пример 1

```
using System;namespace ConsoleApplication1{
   class Class1     {
     static void Main()        {
        Console.Write( "Введите x: " );
        double x = double.Parse( Console.ReadLine() );
        Console.Write( "Введите y: " );
        double y = double.Parse( Console.ReadLine() );
         Console.WriteLine( "Максимум из x и y : " +
Math.Max(x, y) );
         double z = Math.Pow(Math.Sin(x), 2) +
Math.Pow(Math.Sin(y), 2);
         Console.WriteLine( "Сумма квадратов синусов x и
y : " + z );
 }    }    }
```

Контрольные вопросы

1. Какие форма операции присваивания существуют?
2. Каковы преимущества сложных операция присваивания?
3. Каков порядок действия условной операции?