Вызов метода. Способы передачи параметров

Метод может вызываться в выражениях как операнд или быть вызван как оператор. В качестве оператора может использоваться любой метод – как процедура, так и функция.

Сам **вызов метода**, независимо от того, процедура это или функция, имеет один и тот же **синтаксис**:

```
имя_метода([список_фактических_аргументов])
```

Если это оператор, то вызов завершается точкой с запятой.

Формальный параметр, задаваемый при описании метода, это всегда **имя параметра (идентификатор)**.

Фактический аргумент – это **выражение,** значительно более сложная синтаксическая конструкция.

Точный синтаксис фактического аргумента:

[ref|out]выражение

Между списком формальных и списком фактических аргументов должно выполняться определенное **соответствие по числу, порядку следования, типу и статусу аргументов.**

Если в первом списке n формальных аргументов, то фактических аргументов должно быть не меньше n (соответствие по числу). Каждому i-му формальному аргументу (для всех i от 1 до n-1) ставится в соответствие i-й фактический аргумент. Последнему формальному аргументу при условии, что он объявлен с ключевым словом params, ставятся в соответствие все оставшиеся фактические аргументы (соответствие по порядку).

Если **формальный аргумент объявлен с ключевым словом ref или out, то фактический аргумент должен сопровождаться таким же ключевым словом в точке вызова** (соответствие по статусу).

Если формальный аргумент объявлен с типом T, то выражение, задающее фактический аргумент должно быть согласовано по типу с типом T – допускает преобразование к типу T, совпадает с типом T или является его потомком (соответствие по типу).

Если формальный аргумент является выходным – объявлен с ключевым словом ref или out, то соответствующий фактический аргумент не может быть выражением, поскольку используется в левой части оператора присваивания, так что он должен быть именем, которому можно присвоить значение.

Примеры

Пример 1

Рассмотрим **простейший пример** реализации метода **вычисления суммы двух чисел,** используя разные способы передачи параметров

.

Первый способ

```
class Program
    {
        static void Main(string[] args)
        {
            int a = 2, c = 3;
            Console.WriteLine(p(a, c));
            Console.ReadKey();
        }
        static int p(int a1, int c1)
        {
            int s = a1 + c1;
            return s;

        }
    }
```

В методе p вычисляется сумма двух переменных целого типа. Метод возвращает одно значение s, которое вычисляется в этом методе.

Если метод возвращает значение, то имя переменной, в которую помещается возвращаемое значение, указывается после ключевого слова return, присутствие которого в данном случае обязательно.

Второй способ

Те же вычисления выполняются в методе, не возвращающем значения. Если метод не возвращает значения, он имеет тип void..

```
class Program
    {
        static void Main(string[] args)
        {
            int a = 2, c = 3, x;
             ps(a, c, out x);
            Console.WriteLine(x);
            Console.ReadKey();
         }
        static void ps(int a1, int c1, out int s)
        {
            s = a1 + c1;
        }
    }
```

В списке параметров метода перечислены **входные (a1, c1) и выходной (s)** параметры.

Ключевое слово **out** перед выходным параметром, имеющим тип значения, означает передачу параметра по ссылке, т.е. при обращении к методу на место выходного параметра передается адрес аргумента, фигурирующего в обращении к методу.

В примере передается не значение переменной x, а адрес переменной x. Параметр s не является типом int; он является ссылкой на тип int, в данном случае ссылкой на переменную x.

Поэтому после вызова метода значение переменной x изменяется.

Третий способ

Для передачи по ссылке можно использовать также ключевое слово **ref**, но в этом случае аргумент, передаваемый по ссылке, должен быть инициализирован до обращения к методу (при использовании ключевого слова out это необязательно).

Ниже приводится вариант программы с использованием **ref:**

```
class Program
    {
        static void Main(string[] args)
       {
          int a = 2, c = 3, x = 0;
          //x присвоено фиктивное значение 0
          p(a, c, ref x);
          Console.WriteLine(x);
          Console.ReadKey();
        }
          static void p(int a1, int c1, ref int s)
        {
            s = a1 + c1;
        }

    }
```

Пример 2.
Вычислить число сочетаний из n по m по формуле:
$C = n!/(m!(n-m)!)$ (в программе cnm).

Вычисление факториала оформить в виде метода, возвращающего значение (функции)

```
class Program
    {
        static int fact(int n)
        {
            int f = 1;
            for (int i = 2; i <= n; i++)
            {
                f = f * i;
            }
            return f;
        }
        static void Main()
        {
        int n = 5, m = 3 ; int cnm;
        cnm = fact(n)/ (fact(m) * fact(n - m));
        Console.WriteLine("{0}", cnm);
        Console.ReadKey();
        }

    }
```