Существует два способа передачи параметров в метод в языке C#: **по значению** и **по ссылке**.

### Передача параметров по значению

Наиболее простой способ передачи параметров представляет передача по значению, по сути это обычный способ передачи параметров:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | void Increment(int n)  {  n++;  Console.WriteLine($"Число в методе Increment: {n}");  }    int number = 5;  Console.WriteLine($"Число до метода Increment: {number}");  Increment(number);  Console.WriteLine($"Число после метода Increment: {number}"); |

Консольный вывод:

Число до метода Increment: 5  
Число в методе Increment: 6  
Число после метода Increment: 5

При передаче аргументов параметрам по значению параметр метода получает не саму переменную, а ее копию и далее работает с этой копией независимо от самой переменной.

Так, выше при вызове метод Increment получает копию переменной number и увеличивает значение этой копии. Поэтому в самом методе Increment мы видим, что значение параметра n увеличилось на 1, но после выполнения метода переменная number имеет прежнее значение - 5. То есть изменяется копия, а сама переменная не изменяется.

### Передача параметров по ссылке и модификатор ref

При передаче параметров по ссылке перед параметрами используется модификатор **ref**:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | void Increment(ref int n)  {  n++;  Console.WriteLine($"Число в методе Increment: {n}");  }    int number = 5;  Console.WriteLine($"Число до метода Increment: {number}");  Increment(ref number);  Console.WriteLine($"Число после метода Increment: {number}"); |

Консольный вывод:

Число до метода Increment: 5  
Число в методе Increment: 6  
Число после метода Increment: 6

При передаче значений параметрам по ссылке метод получает адрес переменной в памяти. И, таким образом, если в методе изменяется значение параметра, передаваемого по ссылке, то также изменяется и значение переменной, которая передается на его место..

Так, в метод Increment передается ссылка на саму переменную number в памяти. И если значение параметра n в Increment изменяется, то это приводит и к изменению переменной number, так как и параметр n и переменная number указывают на один и тот же адрес в памяти.

Обратите внимание, что модификатор ref указывается как перед параметром при объявлении метода, так и при вызове метода перед аргументом, который передается параметру.

### Выходные параметры. Модификатор out

Выше мы использовали входные параметры. Но параметры могут быть также выходными. Чтобы сделать параметр выходным, перед ним ставится модификатор **out**:

|  |  |
| --- | --- |
| 1  2  3  4 | void Sum(int x, int y, out int result)  {  result = x + y;  } |

Здесь результат возвращается не через оператор return, а через выходной параметр result. Использование в программе:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | void Sum(int x, int y, out int result)  {  result = x + y;  }    int number;    Sum(10, 15, out number);    Console.WriteLine(number); // 25 |

Причем, как и в случае с **ref** ключевое слово **out** используется как при определении метода, так и при его вызове.

Также обратите внимание, что методы, использующие такие параметры, обязательно должны присваивать им определенное значение. То есть следующий код будет недопустим, так как в нем для out-параметра не указано никакого значения:

|  |  |
| --- | --- |
| 1  2  3  4 | void Sum(int x, int y, out int result)  {  Console.WriteLine(x + y);  } |

Прелесть использования подобных параметров состоит в том, что по сути мы можем вернуть из метода не одно значение, а несколько. Например:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | void GetRectangleData(int width, int height, out int rectArea, out int rectPerimetr)  {  rectArea = width \* height; // площадь прямоугольника - произведение ширины на высоту  rectPerimetr = (width + height) \* 2; // периметр прямоугольника - сумма длин всех сторон  }    int area;  int perimetr;    GetRectangleData(10, 20, out area, out perimetr);    Console.WriteLine($"Площадь прямоугольника: {area}"); // 200  Console.WriteLine($"Периметр прямоугольника: {perimetr}"); // 60 |

Здесь у нас есть метод GetRectangleData, который получает ширину и высоту прямоугольника (параметры width и height). А два выходных параметра мы используем для подсчета площади и периметра прямоугольника.

При этом можно определять переменные, которые передаются **out**-параметрам в непосредственно при вызове метода. То есть мы можем сократить предыдущий пример следующим образом:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | void GetRectangleData(int width, int height, out int rectArea, out int rectPerimetr)  {  rectArea = width \* height;  rectPerimetr = (width + height) \* 2;  }    GetRectangleData(10, 20, out int area, out int perimetr);    Console.WriteLine($"Площадь прямоугольника: {area}"); // 200  Console.WriteLine($"Периметр прямоугольника: {perimetr}"); // 60 |

При этом, если нам неизвестен тип значений, которые будут присвоены параметрам, то мы можем для их определения использовать оператор **var**:

|  |  |
| --- | --- |
| 1  2  3  4 | GetRectangleData(10, 20, out var area, out var perimetr);    Console.WriteLine($"Площадь прямоугольника: {area}"); // 200  Console.WriteLine($"Периметр прямоугольника: {perimetr}"); // 60 |

### Входные параметры. Модификатор in

Кроме выходных параметров с модификатором out метод может использовать входные параметры с модификатором **in**. Модификатор **in** указывает, что данный параметр будет передаваться в метод по ссылке, однако внутри метода его значение параметра нельзя будет изменить. Например, возьмем следующий метод:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | void GetRectangleData(in int width, in int height, out int rectArea, out int rectPerimetr)  {  //width = 25; // нельзя изменить, так как width - входной параметр  rectArea = width \* height;  rectPerimetr = (width + height) \* 2;  }    int w = 10;  int h = 20;  GetRectangleData(w, h, out var area, out var perimetr);    Console.WriteLine($"Площадь прямоугольника: {area}"); // 200  Console.WriteLine($"Периметр прямоугольника: {perimetr}"); // 60 |

В данном случае через входные параметры width и height в метод передаются значения, но в самом методе мы не можем изменить значения этих параметров, так как они определены с модификатором **in**.

Передача по ссылке в некоторых случаях может увеличить произодительность, а использование оператора **in** гарантирует, что значения переменных, которые передаются параметрам, нельзя будет изменить в этом методе.