

# МЕТОДИ

*Метод - це іменований блок коду, який виконує деякі дії*

```
1 [модификаторы] тип_возвращаемого_значения название_метода ([параметры])
2 {
3     // тело метода
4 }
```

Загальна форма	Приклад.
<div>заголовок функції</div> <pre>static &lt;тип результату&gt; &lt;ім'я функції&gt;(&lt;список форм. парам.&gt;) {     //тіло функції }</pre>	<p>Знайти максимальне з трьох цілих чисел</p> <pre>static int Max(int c1,int c2,int c3) {     int m = c1;     if (c2 &gt; m)         m = c2;     if (c3 &gt; m)         m = c3;     return m; }</pre>

```
1 static void Main(string[] args)
2 {
3
4 }
```

```
1 using System;
2
3 namespace HelloApp
4 {
5     class Program
6     {
7         static void Main(string[] args)
8         {
9
10        }
11
12        static void SayHello()
13        {
14            Console.WriteLine("Hello");
15        }
16        static void SayGoodbye()
17        {
18            Console.WriteLine("GoodBye");
19        }
20    }
21 }
```

# ВИКЛИК МЕТОДІВ

1 | `название_метода (значения_для_параметров_метода);`

```
class Program
{
    static void Main(string[] args)
    {
        SayHello();
        SayGoodbye();

        Console.ReadKey();
    }

    static void SayHello()
    {
        Console.WriteLine("Hello");
    }

    static void SayGoodbye()
    {
        Console.WriteLine("GoodBye");
    }
}
```

Hello  
GoodBye

# ПОВЕРНЕННЯ ЗНАЧЕННЯ

1 | `return возвращаемое значение;`

```
1 | static string GetHello()
2 | {
3 |     return "Hello";
4 | }
5 | static int GetSum()
6 | {
7 |     int x = 2;
8 |     int y = 3;
9 |     int z = x + y;
10 |    return z;
11 | }
```

```
1 | static int GetSum()
2 | {
3 |     int x = 2;
4 |     int y = 3;
5 |     return x + y;
6 | }
```

```
1 | static int GetSum()
2 | {
3 |     int x = 2;
4 |     int y = 3;
5 |     return "5"; // c
6 | }
```

```
static void Main(string[] args)
{
    string message = GetHello();
    int sum = GetSum();

    Console.WriteLine(message); // Hello
    Console.WriteLine(sum);    // 5

    Console.ReadKey();
}

static string GetHello()
{
    return "Hello";
}

static int GetSum()
{
    int x = 2;
    int y = 3;
    return x + y;
}
```

```
1 | static string GetHello()
2 | {
3 |     Console.WriteLine("Hello");
4 | }
```

## ВИХІД З МЕТОДУ

```
1 static string GetHello()
2 {
3     return "Hello";
4     Console.WriteLine("After return");
5 }
```

```
1 static void SayHello()
2 {
3     int hour = 23;
4     if(hour > 22)
5     {
6         return;
7     }
8     else
9     {
10        Console.WriteLine("Hello");
11    }
12 }
```

### Скорочений запис методів

```
1 static string GetHello()
2 {
3     return "hello";
4 }
```

```
1 static string GetHello() => "hello";
```

```
1 static void SayHello() => Console.WriteLine("Hello");
```

## ПАРАМЕТРИ МЕТОДІВ

```
1 static int Sum(int x, int y)
2 {
3     return x + y;
4 }
```

```
1 class Program
2 {
3     static void Main(string[] args)
4     {
5         int result = Sum(10, 15);
6         Console.WriteLine(result); // 25
7
8         Console.ReadKey();
9     }
10    static int Sum(int x, int y)
11    {
12        return x + y;
13    }
14 }
```

При виклику методу *Sum* значення передаються параметрам по позиції. Число 10 передається параметру *x*, а число 15 - параметру *y*. Числа 10, 15 називаються аргументами

# ПАРАМЕТРИ МЕТОДІВ

```
{
    static void Main(string[] args)
    {
        int a = 25;
        int b = 35;
        int result = Sum(a, b);
        Console.WriteLine(result); // 60

        result = Sum(b, 45);
        Console.WriteLine(result); // 80

        result = Sum(a + b + 12, 18); // "a + b + 12" представляє значення параметра x
        Console.WriteLine(result); // 90

        Console.ReadKey();
    }
    static int Sum(int x, int y)
    {
        return x + y;
    }
}
```

```
class Program
{
    static void Main(string[] args)
    {
        Display("Tom", 24); // Name: Tom Age: 24

        Console.ReadKey();
    }
    static void Display(string name, int age)
    {
        Console.WriteLine($"Name: {name} Age: {age}");
    }
}
```

```
class Program
{
    static void Main(string[] args)
    {
        int a;
        int b = 9;
        Sum(a, b); // Ошибка - переменной a не присвоено значение

        Console.ReadKey();
    }
    static int Sum(int x, int y)
    {
        return x + y;
    }
}
```

1 | Display(45, "Bob"); // Ошибка! несоответствие значений типам параметров

# ПАРАМЕТРИ МЕТОДІВ

## Необов'язкові параметри

```
1 static int OptionalParam(int x, int y, int z=5, int s=4)
2 {
3     return x + y + z + s;
4 }
```

```
1 static void Main(string[] args)
2 {
3     OptionalParam(2, 3);
4
5     OptionalParam(2,3,10);
6
7     Console.ReadKey();
8 }
```

## Іменовані параметри

```
1 static int OptionalParam(int x, int y, int z=5, int s=4)
2 {
3     return x + y + z + s;
4 }
5 static void Main(string[] args)
6 {
7     OptionalParam(x:2, y:3);
8
9     //Необязательный параметр z использует значение по умолчанию
10    OptionalParam(y:2, x:3, s:10);
11
12    Console.ReadKey();
13 }
```

## Без функцій

$$S = \underbrace{\max\{a, -b, 3\}}_{\text{max 1}} * \underbrace{\max\{a, 2b, c\}}_{\text{max 2}} - \underbrace{\max\{-a, b, 7\}}_{\text{max 3}} = \text{max 1} * \text{max 2} - \text{max 3}$$

```
class Program
{
```

```
    static void Main(string[] args)
    {
        int a,b,c;
        Console.Write("a=");
        a = int.Parse(Console.ReadLine());
        Console.Write("b=");
        b = int.Parse(Console.ReadLine());
        Console.Write("c=");
        c = int.Parse(Console.ReadLine());
```

```
        int max1 = a;
        if (-b > max1)
            max1 = -b;
        if (3 > max1)
            max1 = 3;
```

```
        int max2 = a;
        if (2*b > max2)
            max2 = 2*b;
        if (c > max1)
            max2 = c;
```

```
        int max3 = -a;
        if (b > max3)
            max3 = b;
        if (7 > max3)
            max3 = 7;
```

```
        int S = max1 * max2 - max3;
        Console.WriteLine("S={0}",S);
        Console.ReadKey();
    }
```

```
}
```

```
class Program
{
```

```
    static int Max(int c1, int c2, int c3)
    {
        int m = c1;
        if (c2 > m)
            m = c2;
        if (c3 > m)
            m = c3;
        return m;
    }
```

```
    static void Main(string[] args)
    {
        int a,b,c;
        Console.Write("a=");
        a = int.Parse(Console.ReadLine());
        Console.Write("b=");
        b = int.Parse(Console.ReadLine());
        Console.Write("c=");
        c = int.Parse(Console.ReadLine());
```

```
        int max1 = Max(a, -b, 3);
```

```
        int max2 = Max(a, 2*b, c);
```

```
        int max3 = Max(-a, b, 7);
```

```
        int S = max1 * max2 - max3;
        Console.WriteLine("S={0}",S);
        Console.ReadKey();
    }
```

```
}
```

## з функціями

# ПЕРЕДАЧА ПАРАМЕТРІВ ЗА ПОСИЛАННЯМ І ЗНАЧЕННЯМ

## Передача параметрів за значенням

```
1 class Program
2 {
3     static void Main(string[] args)
4     {
5         Sum(10, 15);           // параметри передаються по значенню
6         Console.ReadKey();
7     }
8     static int Sum(int x, int y)
9     {
10         return x + y;
11     }
12 }
```

*Створити функцію, що повертає середнє арифметичне значення трьох дійсних чисел*

```
class Program
{
    static double Average(double c1, double c2, double c3)
    {
        return (c1 + c2 + c3) / 3;
    }
    static void Main(string[] args)
    {
        double n, m, k;
        Console.WriteLine("n=");
        n = double.Parse(Console.ReadLine());
        Console.WriteLine("m=");
        m = double.Parse(Console.ReadLine());
        Console.WriteLine("k=");
        k = double.Parse(Console.ReadLine());
        double ser= Average(n, m, k);
        Console.WriteLine("Average = {0} ",ser);
        Console.ReadKey();
    }
}
```

# ПЕРЕДАЧА ПАРАМЕТРОВ ЗА ПОСИЛАННЯМ І ЗНАЧЕННЯМ

## Передача параметрів за посиланням і модифікатор ref

```
1 static void Main(string[] args)
2 {
3     int x = 10;
4     int y = 15;
5     Addition(ref x, y); // вызов метода
6     Console.WriteLine(x); // 25
7
8     Console.ReadLine();
9 }
10 // параметр x передается по ссылке
11 static void Addition(ref int x, int y)
12 {
13     x += y;
14 }
```

Начальное значение переменной a = 5

IncrementVal: 6

Переменная a после передачи по значению равна = 5

```
1 class Program
2 {
3     static void Main(string[] args)
4     {
5         int a = 5;
6         Console.WriteLine($"Начальное значение переменной a = {a}");
7
8         //Передача переменных по значению
9         //После выполнения этого кода по-прежнему a = 5, так как мы передали лишь ее копию
10        IncrementVal(a);
11        Console.WriteLine($"Переменная a после передачи по значению равна = {a}");
12        Console.ReadKey();
13    }
14    // передача по значению
15    static void IncrementVal(int x)
16    {
17        x++;
18        Console.WriteLine($"IncrementVal: {x}");
19    }
20 }
```



# ПЕРЕДАЧА ПАРАМЕТРОВ ЗА ПОСИЛАННЯМ І ЗНАЧЕННЯМ

## Передача параметрів за посиланням і модифікатор ref

```
1 class Program
2 {
3     static void Main(string[] args)
4     {
5         int a = 5;
6         Console.WriteLine($"Начальное значение переменной a = {a}");
7         //Передача переменных по ссылке
8         //После выполнения этого кода a = 6, так как мы передали саму переменную
9         IncrementRef(ref a);
10        Console.WriteLine($"Переменная a после передачи ссылке равна = {a}");
11
12        Console.ReadKey();
13    }
14    // передача по ссылке
15    static void IncrementRef(ref int x)
16    {
17        x++;
18        Console.WriteLine($"IncrementRef: {x}");
19    }
20 }
```

*Створити функцію, яка  
більше значення  
зменшує на 2, а менше  
збільшує на 3*

Начальное значение переменной a = 5

IncrementRef: 6

Переменная a после передачи по ссылке равна = 6

```
class Program
{
    static void Fun1(ref int a, ref int b)
    {
        if (a > b)
        {
            a -= 2;
            b += 3;
        }
        else
        {
            b -= 2;
            a += 3;
        }
    }
    static void Main(string[] args)
    {
        int n,m;
        Console.WriteLine("n=");
        n=int.Parse(Console.ReadLine());
        Console.WriteLine("m=");
        m = int.Parse(Console.ReadLine());
        Fun1(ref n,ref m);
        Console.WriteLine("n= {0} m={1}",n,m);
        Console.ReadKey();
    }
}
```

# ВИХІДНІ ПАРАМЕТРИ. МОДИФІКАТОР OUT

```
1 static void Sum(int x, int y, out int a)
2 {
3     a = x + y;
4 }
```

```
1 static void Main(string[] args)
2 {
3     int x = 10;
4
5     int z;
6
7     Sum(x, 15, out z);
8
9     Console.WriteLine(z);
10
11    Console.ReadKey();
12 }
```

Модифікатор **in** вказує, що через цей параметр буде передаватися в метод за посиланням, однак всередині методу його значення параметра не можна буде змінити. самому методі можна змінити тільки значення параметра **y**, так як параметр **x** вказано з модифікатором **in**

```
1 static void GetData(in int x, int y, out int area, out int perim)
2 {
3     // x = x + 10; нельзя изменить значение параметра x
4     y = y + 10;
5     area = x * y;
6     perim = (x + y) * 2;
7 }
```

```
1 static void Main(string[] args)
2 {
3     int x = 10;
4     int area;
5     int perimetr;
6     GetData(x, 15, out area, out perimetr);
7     Console.WriteLine("Площадь : " + area);
8     Console.WriteLine("Периметр : " + perimetr);
9
10    Console.ReadKey();
11 }
12 static void GetData(int x, int y, out int area, out int perim)
13 {
14     area = x * y;
15     perim = (x + y) * 2;
16 }
```

```
int x = 10;
int area;
int perimetr;
GetData(x, 15, out area, out perimetr);
Console.WriteLine($"Площадь : {area}");
Console.WriteLine($"Периметр : {perimetr}");
```

**Визначення змінних  
безпосередньо при виклику  
методу**

```
int x = 10;
GetData(x, 15, out int area, out int perimetr);
Console.WriteLine($"Площадь : {area}");
Console.WriteLine($"Периметр : {perimetr}");
```

# ВИХІДНІ ПАРАМЕТРИ. МОДИФІКАТОР OUT

*Створити функцію, яка повертає максимальне та мінімальне значення із двох дійсних чисел*

```
class Program
{
    static void MinMax(int c1, int c2, out int max, out int min)
    {
        if (c1 > c2)
        {
            max = c1;
            min = c2;
        }
        else
        {
            max = c2;
            min = c1;
        }
    }
    static void Main(string[] args)
    {
        int n,m;
        Console.WriteLine("n=");
        n=int.Parse(Console.ReadLine());
        Console.WriteLine("m=");
        m = int.Parse(Console.ReadLine());
        int max, min;
        MinMax(n, m, out max, out min);
        Console.WriteLine("max= {0} min={1}",max,min);
        Console.ReadKey();
    }
}
```

# ПЕРЕДАЧА ПАРАМЕТРІВ ЗА ПОСИЛАННЯМ І ЗНАЧЕННЯМ

Передача за значенням (немає ніяких специфікаторів) (ініціалізація змінної п обов'язкова)	Передача за покажчиком (специфікатор <u>ref</u> ) (ініціалізація змінної п обов'язкова)	Аргумент як вихідне значення (специфікатор <u>out</u> ) (ініціалізація змінної п не обов'язкова)
<pre>class Program {     static void Fun1(int m)     {         m = 1;     }     static void Main(string[] args)     {         int n = 25;         Fun1(n);         Console.WriteLine("n= {0}",n);         Console.ReadKey();     } }</pre>	<pre>class Program {     static void Fun1(ref int m)     {         m = 1;     }     static void Main(string[] args)     {         int n = 25;         Fun1(ref n);         Console.WriteLine("n= {0}",n);         Console.ReadKey();     } }</pre>	<pre>class Program {     static void Fun1(out int m)     {         m = 1;     }     static void Main(string[] args)     {         int n;         Fun1(out n);         Console.WriteLine("n= {0}",n);         Console.ReadKey();     } }</pre>
Вивід програми: n=25	Вивід програми: n=1	Вивід програми: n=1
<b>Змінні в пам'яті ЕОМ:</b> До виклику: <div style="border: 1px dashed black; padding: 5px; margin: 10px auto; width: 100px; text-align: center;">                 Програма n 25             </div>	<b>Змінні в пам'яті ЕОМ:</b> До виклику: <div style="border: 1px dashed black; padding: 5px; margin: 10px auto; width: 100px; text-align: center;">                 Програма n 25             </div>	<b>Змінні в пам'яті ЕОМ:</b> До виклику: <div style="border: 1px dashed black; padding: 5px; margin: 10px auto; width: 100px; text-align: center;">                 Програма n 25             </div>
<b>Під час роботи функції:</b> (для змінної m виділяється нова пам'ять) <div style="display: flex; justify-content: space-around; border: 1px dashed black; padding: 5px; margin: 10px auto; width: 200px;"> <div style="border-right: 1px dashed black; padding: 5px; text-align: center;">                 Програма n 25             </div> <div style="padding: 5px; text-align: center;">                 Функція m 1             </div> </div>	<b>Під час роботи функції:</b> (для змінної m нова пам'ять не виділяється а використовується пам'ять змінної n) <div style="display: flex; justify-content: space-around; border: 1px dashed black; padding: 5px; margin: 10px auto; width: 200px;"> <div style="border-right: 1px dashed black; padding: 5px; text-align: center;">                 Програма n 25             </div> <div style="padding: 5px; text-align: center;">                 Функція m ←→ n             </div> </div>	<b>Під час роботи функції:</b> (для змінної m нова пам'ять не виділяється а використовується пам'ять змінної n) <div style="display: flex; justify-content: space-around; border: 1px dashed black; padding: 5px; margin: 10px auto; width: 200px;"> <div style="border-right: 1px dashed black; padding: 5px; text-align: center;">                 Програма n 25             </div> <div style="padding: 5px; text-align: center;">                 Функція m ←→ n             </div> </div>
<b>Після роботи функції:</b> <div style="border: 1px dashed black; padding: 5px; margin: 10px auto; width: 100px; text-align: center;">                 Програма n 25             </div>	<b>Після роботи функції:</b> <div style="border: 1px dashed black; padding: 5px; margin: 10px auto; width: 100px; text-align: center;">                 Програма n 1             </div>	<b>Після роботи функції:</b> <div style="border: 1px dashed black; padding: 5px; margin: 10px auto; width: 100px; text-align: center;">                 Програма n 1             </div>

# ОБЛАСТЬ ВИДИМОСТИ ЗМІННИХ

```
1  class Program // начало контекста класса
2  {
3      static int a = 9; // переменная уровня класса
4
5      static void Main(string[] args) // начало контекста метода Main
6      {
7          int b = a - 1; // переменная уровня метода
8
9          { // начало контекста блока кода
10
11              int c = b - 1; // переменная уровня блока кода
12
13          } // конец контекста блока кода, переменная c уничтожается
14
15          //так нельзя, переменная c определена в блоке кода
16          //Console.WriteLine(c);
17
18          //так нельзя, переменная d определена в другом методе
19          //Console.WriteLine(d);
20
21          Console.Read();
22
23      } // конец контекста метода Main, переменная b уничтожается
24
25      void Display() // начало контекста метода Display
26      {
27          // переменная a определена в контексте класса, поэтому доступна
28          int d = a + 1;
29
30      } // конец контекста метода Display, переменная d уничтожается
31
32  } // конец контекста класса, переменная a уничтожается
```

# РЕКУРСИВНІ ФУНКЦІЇ

Визначимо метод для знаходження факторіала:

$$n! = 1 * 2 * \dots * n$$

```
1 static int Factorial(int x)
2 {
3     if (x == 0)
4     {
5         return 1;
6     }
7     else
8     {
9         return x * Factorial(x - 1);
10    }
11 }
```

$$x^n = \begin{cases} 1, & \text{якщо } n = 0, \\ \frac{1}{x^{|n|}}, & \text{якщо } n < 0, \\ x \cdot x^{n-1}, & \text{якщо } n > 0. \end{cases}$$

Визначимо метод для  $n$ -го члену послідовності Фібоначчі :  $f(n) = f(n-1) + f(n-2)$

```
1 static int Fibonacci(int n)
2 {
3     if (n == 0 || n == 1)
4     {
5         return n;
6     }
7     else
8     {
9         return Fibonacci(n - 1) + Fibonacci(n - 2);
10    }
11 }
```

```
static double f(double x, int n)
{
    if (n == 0)
        return 1;
    else
        if (n < 0)
            return 1 / f(x, Math.Abs(n));
        else
            return x * f(x, n - 1);
}
```