

PROGRAMACIÓN EN C#

5. LOS MÉTODOS

1

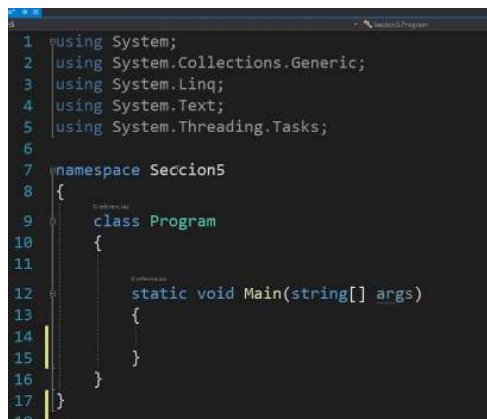
5. LOS MÉTODOS.

A.INTRODUCCIÓN A LOS MÉTODOS.

Un método es un bloque de código que contiene una serie de instrucciones.

La jerarquía en C# es:

Namespace – Clase - Método



```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace Seccion5
8 {
9     class Program
10     {
11         static void Main(string[] args)
12         {
13         }
14     }
15 }
16
17
18
```

2

2

5. LOS MÉTODOS.

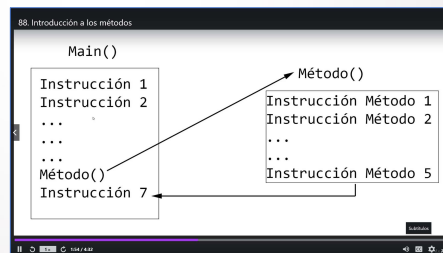
¿Cuándo es necesario crear un método?

Cuando hay cierto código que nosotros repetimos a lo largo del programa.

De esta forma el código que necesitamos que se repita estará dentro del método y cada vez que queramos que se repita bastará con llamar al método.

Ejemplo: llamada al
método

```
Console.WriteLine();
```



3

3

5. LOS MÉTODOS.

B. SINTAXIS DE UN MÉTODO.

Dos tipos de métodos,
estáticos y de instancia.

- Estáticos. se utiliza la palabra reservada `static` para declarar miembros de la clase que no están ligados a ninguna instancia (objeto) de la misma.
- De instancia. Para trabajar con OO.

```
//[modificador][tipo][identificador][parametros]  
Static void Nombre (parametros)  
{  
    //instrucciones  
}
```



4

4

5. LOS MÉTODOS.

Estilos de métodos:

1. Métodos sin parámetros ni tipo.
2. Métodos que devuelven un tipo.
3. Métodos con parámetros.
4. Métodos con parámetros y que devuelven un tipo.



5

5

5. LOS MÉTODOS.

C. ESTILOS DE MÉTODOS. ESTILO 1: MÉTODOS SIN PARÁMETROS, NI TIPO.

```
//[modificador][tipo][identificador][parámetros]
static void Sumar(){
    decimal num1, num2, resultado;
    Console.Write("Mete el primer número: ")
    num1 = Convert.ToDecimal(Console.ReadLine());
    Console.Write("Mete el segundo número: ")
    num2 = Convert.ToDecimal(Console.ReadLine());
    resultado = num1 + num2;
    Console.WriteLine("{0}+{1} = {2}",num1, num2, resultado);
}
```



6

6

5. LOS MÉTODOS.

INVOCACIÓN DE UN MÉTODO SIN PARÁMETROS, NI TIPO.

```
Switch (opción){  
    case 1:  
        Sumar();  
        break;  
    case 2:  
        ....  
}
```



7

7

5. LOS MÉTODOS.

C. ESTILOS DE MÉTODOS. ESTILO 2: MÉTODOS QUE DEVUELVEN UN TIPO.

```
//[modificador][tipo][identificador][parámetros]  
static decimal Restar(){  
    decimal num1, num2, resultado;  
    Console.Write("Mete el primer número: ")  
    num1 = Convert.ToDecimal(Console.ReadLine());  
    Console.Write("Mete el segundo número: ")  
    num2 = Convert.ToDecimal(Console.ReadLine());  
    resultado = num1 - num2;  
    return resultado;  
}
```



8

8

5. LOS MÉTODOS.

INVOCACIÓN DE UN MÉTODO QUE DEVUELVE UN TIPO.

```
Decimal r; //Almacena el valor devuelto por restar

Switch (opción){
    case 1:
        ...
    case 2:
        r = Restar(); //Asignamos a r el valor devuelto por return
        Console.WriteLine("El resultado de restar es: {0}", r); //Mostramos
        el resultado
        break;
    ...
}
```



9

5. LOS MÉTODOS.

C. ESTILOS DE MÉTODOS. ESTILO 3: MÉTODOS CON PARÁMETROS.

```
//[modificador][tipo][identificador][parámetros]
static void Multiplicar(decimal num1Pa, decimal num2Pa){
    decimal resultado;
    resultado = num1Pa * num2Pa;
    Console.WriteLine("{0}*{1}={2}", num1Pa, num2Pa, resultado);
}
```



10

10

5. LOS MÉTODOS.

INVOCACIÓN DE UN MÉTODO CON PARÁMETROS.

```
decimal num1Ar, num2Ar; //Argumentos para enviar una copia de su valor al método
switch (opción){
    ...
    case 3:
        Console.Write("Mete el primer número: ")
        num1Ar = Convert.ToDecimal(Console.ReadLine());
        Console.Write("Mete el segundo número: ")
        num2Ar = Convert.ToDecimal(Console.ReadLine());
        Multiplicar(num1Ar, num2Ar);
        break;
    ...
}
```



11

11

5. LOS MÉTODOS.

C. ESTILOS DE MÉTODOS. ESTILO 4: MÉTODOS CON PARÁMETROS Y QUE DEVUELVEN UN TIPO

```
static decimal Dividir (decimal num1Pa, num2Pa){
    decimal resultado;
    if(num2Pa != 0){
        resultado = num1Pa/num2Pa;
    }else{
        Console.WriteLine("No es posible dividir entre 0");
        resultado = 0;
    }
    return resultado;
}
```



12

12

5. LOS MÉTODOS.

INVOCACIÓN DE UN MÉTODO CON PARÁMETROS Y QUE DEVUELVEN UN TIPO.

```
switch (opción){  
    ...  
    case 4:  
        Console.Write("Mete el primer número: ")  
        num1Ar = Convert.ToDecimal(Console.ReadLine());  
        Console.Write("Mete el segundo número: ")  
        num2Ar = Convert.ToDecimal(Console.ReadLine());  
        r=Dividir(num1Ar, num2Ar);  
        Console.WriteLine("El resultado de la división es: {0}",r);  
        break;  
    ...  
}
```



13

13

5. LOS MÉTODOS.

D. OPTIMIZANDO CON MÉTODOS.

```
Console.Write("Mete el primer número: ")  
num1 = Convert.ToDecimal(Console.ReadLine());  
Console.Write("Mete el segundo número: ")  
num2 = Convert.ToDecimal(Console.ReadLine());
```



```
static decimal IngresarNumer(string petición){  
    decimal numero;  
    Console.Write(petición);  
    numero = Convert.ToDecimal(Console.ReadLine());  
    return numero;  
}
```



14

5. LOS MÉTODOS.

E. PASAR POR REFERENCIA VS POR VALOR.

Pasar por valor o pasar por copia.

Estamos enviando una copia del valor de la variable y no como tal la variable.

```
static void main (string[] args){  
    byte numAr = 10;  
    Console.WriteLine(numAr); //imprime 10  
    Prueba(numAr);  
    Console.WriteLine(numAr); //imprime 10  
}  
  
static void Prueba (byte numPa){  
    numPa = 20;  
}
```



15

15

5. LOS MÉTODOS.

E. PASAR POR REFERENCIA VS POR VALOR.

Pasar por referencia

Estamos enviando una referencia de la variable al método. No enviamos la variable como tal sino la ubicación de la variable.

Usamos KeyWord ref

```
static void main (string[] args){  
    byte numAr = 10;  
    Console.WriteLine(numAr); //imprime 10  
    Prueba(ref numAr);  
    Console.WriteLine(numAr); //imprime 20  
}  
  
static void Prueba (ref byte numPa){  
    numPa = 20;  
}
```



16

16

5. LOS MÉTODOS.

F. LA PALABRA CLAVE OUT.

Lo que hacemos en este caso es declarar el argumento en main e inicializarlo en el método Prueba usando un parámetro.

Usamos KeyWord out

```
static void main (string[] args){  
    byte numAr;  
    Prueba(out numAr);  
    Console.WriteLine(numAr); //imprime 20  
}  
  
static void Prueba (out byte numPa){  
    numPa = 20;  
}
```



17

17

5. LOS MÉTODOS.

G. LA PALABRA CLAVE VAR. VARIABLES LOCALES CON ASIGNACIÓN IMPLÍCITA DE TIPOS.

Permite declarar una variable sin necesidad de especificar el tipo.

Dejamos que el compilador decida el tipo que le colocará a la variable en función de su contenido.

```
var i = 10; //asignación implícita.  
int i = 10; //asignación explícita.  
var nombre = "Luis"; //siempre debe inicializarse.  
var nombre; //error al no estar inicializada.  
var nombre = "Luis", apellido = "Sanz"; //no puede haber varios declaradores.
```



18

18

5. LOS MÉTODOS.

H. EL TIPO DE DATO TUPLA. DEFINICIÓN.

Nos permiten agrupar varios tipos de datos relacionados.

```
string nombre = "Luis";  
byte edad = 50;  
long numero = 646789867;  
Int dirPostal = 458939
```



ASIGNACIÓN EXPLÍCITA

```
//() Identificador = (valor);  
(string, byte, long, int) persona1 = ("Luis", 50, 646789867, 458939);
```



```
//() Identificador = (valor);  
var persona1 = ("Luis", 50, 646789867, 458939);
```

ASIGNACIÓN IMPLÍCITA

19



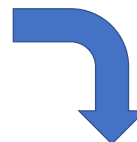
19

5. LOS MÉTODOS.

H. EL TIPO DE DATO TUPLA. IMPRESIÓN

```
var persona1 = ("Luis", 50, 646789867, 458939);  
Console.WriteLine(persona1); //(Luis, 50, 646789867, 458939);  
Console.WriteLine(persona1.Item1); //Luis  
Console.WriteLine(persona1.Item2); //50  
Console.WriteLine(persona1.Item3); //646789867  
Console.WriteLine(persona1.Item4); // 458939
```

ASIGNACIÓN IMPLÍCITA



```
var persona1 = (nombre: "Luis", edad: 50, numero: 646789867, dirPostal: 458939);  
Console.WriteLine(persona1); //(Luis, 50, 646789867, 458939);  
Console.WriteLine(persona1.nombre); //Luis  
Console.WriteLine(persona1.edad); //50  
Console.WriteLine(persona1.numero); //646789867  
Console.WriteLine(persona1.dirPostal); // 458939
```



20

5. LOS MÉTODOS.

H. EL TIPO DE DATO TUPLA. IMPRESIÓN

```
(string, byte, long, int) persona1 = ("Luis", 50, 646789867, 458939);  
Console.WriteLine(persona1); //(Luis, 50, 646789867, 458939);  
Console.WriteLine(persona1.Item1); //Luis  
Console.WriteLine(persona1.Item2); //50  
Console.WriteLine(persona1.Item3); //646789867  
Console.WriteLine(persona1.Item4); // 458939
```

ASIGNACIÓN EXPLÍCITA



```
(string nombre, byte edad, long numero, int dirPostal) persona1 = ("Luis", 50, 646789867, 458939);  
Console.WriteLine(persona1); //(Luis, 50, 646789867, 458939);  
Console.WriteLine(persona1.nombre); //Luis  
Console.WriteLine(persona1.edad); //50  
Console.WriteLine(persona1.numero); //646789867  
Console.WriteLine(persona1.dirPostal); // 458939
```

21

5. LOS MÉTODOS.

I. DEVOLVIENDO MÚLTIPLES VALORES CON RETURN.

```
//variable tupla definida en método main  
(decimal num1, decimal num2, decimal resultado) numeros;  
  
static (decimal, decimal, decimal) Restar(){  
    decimal num1, num2, resultado;  
  
    Console.Write("Mete el primer número: ")  
    num1 = Convert.ToDecimal(Console.ReadLine());  
    Console.Write("Mete el segundo número: ")  
    num2 = Convert.ToDecimal(Console.ReadLine());  
    resultado = num1 - num2;  
  
    return (num1, num2, resultado);  
}
```

```
case 2:  
    numeros= Restar();  
    Console.WriteLine("{0}-{1}={2}", numeros.num1,  
        numeros.num2, numeros.resultado)  
    break;  
....
```

22

TAREA #18

Crear un método para transformar de grados a radianes

Para pasar de grados a radianes hay que multiplicar los grados por $\pi/180$.

Vamos a verlo con unos ejemplos prácticos:

- 120 grados a radianes: $120 \times \pi/180 = 0,66\pi$ radianes
- 30 grados a radianes: $30 \times \pi/180 = 0,16\pi$ radianes
- 360 grados a radianes: $360 \pi/180 = 2\pi$ radianes



23

23

TAREA #19

Crear una aplicación que calcule el área de un círculo, cuadrado o triángulo. Le preguntaremos al usuario a qué figura le quiere calcular el área y dependiendo el caso, ejecutará uno de los 3 métodos.



24

24