

# PROGRAMACIÓN EN C#

## 2. ELEMENTOS BÁSICOS DE UN PROGRAMA DE ORDENADOR

1

## 2. ELEMENTOS BÁSICOS DE UN PROGRAMA DE ORDENADOR

### **A.C# Y LOS PROGRAMAS DE ORDENADOR.**

- Utilizamos los lenguajes de programación para comunicarnos con los ordenadores.
- Desde hace años han existido diversos lenguajes de programación que son evolución unos de otros.
- C# es un lenguaje moderno, de uso general y orientado a objetos.
- C# lenguaje creado por Microsoft que funciona en la plataforma .NET.
- Guarda muchas similitudes con lenguajes como C, C++ o Java.



2

2

## 2. ELEMENTOS BÁSICOS DE UN PROGRAMA DE ORDENADOR

### A.C# Y LOS PROGRAMAS DE ORDENADOR.

Ventajas de C#:

- Fácil de aprender, pues es de alto nivel.
- Ampliamente usado para desarrollar:
  - Aplicaciones de ordenador, web, móvil, videojuegos.
  - Gran comunidad y soporte de Microsoft.



3

3

## 2. ELEMENTOS BÁSICOS DE UN PROGRAMA DE ORDENADOR

### B.EL ORDEN EN LOS PROGRAMAS DE ORDENADOR.

- Algoritmo: Conjunto de reglas e instrucciones ordenadas y finitas que permiten llevar a cabo una actividad, resolver un problema o realizar algo.

El algoritmo en términos de programación es el equivalente a una receta y los ingredientes podríamos verlos como los datos o la información que usaremos.

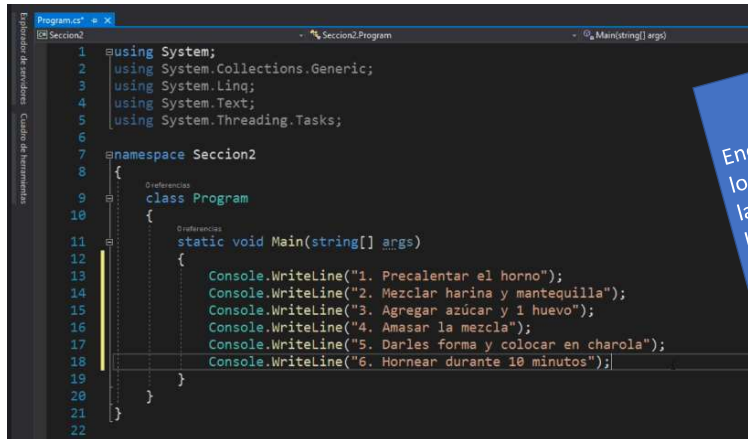


4

4

## 2. ELEMENTOS BÁSICOS DE UN PROGRAMA DE ORDENADOR

### B. EL ORDEN EN LOS PROGRAMAS DE ORDENADOR.



```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace Seccion2
8 {
9     class Program
10     {
11         static void Main(string[] args)
12         {
13             Console.WriteLine("1. Precalentar el horno");
14             Console.WriteLine("2. Mezclar harina y mantequilla");
15             Console.WriteLine("3. Agregar azúcar y 1 huevo");
16             Console.WriteLine("4. Amasar la mezcla");
17             Console.WriteLine("5. Darles forma y colocar en charola");
18             Console.WriteLine("6. Hornear durante 10 minutos");
19         }
20     }
21 }
22
```

Encuentro información sobre los métodos y propiedades de la clase:  
<https://docs.microsoft.com/es-es/dotnet/api/system.console?view=net-5.0>



5

5

## 2. ELEMENTOS BÁSICOS DE UN PROGRAMA DE ORDENADOR

### C. TIPOS DE ERRORES.

- Errores de sintaxis.
  - Son más fáciles de corregir.
  - Escribimos algo mal.
  - Son detectados por el compilador y lo muestra en pantalla de Salida.
- Errores de lógica.
  - Más difíciles de corregir.
  - Se deben a un mal planteamiento de nuestro algoritmo.



6

6

## 2. ELEMENTOS BÁSICOS DE UN PROGRAMA DE ORDENADOR

### D.DIFERENCIAS ENTRE WRITE LINE Y WRITE.

- Método `System.Console.Write("")`;
  - Imprime el texto entrecomillado sin salto de línea.
- Método `System.Console.WriteLine("")`;
  - Imprime el texto entrecomillado con salto de línea.



7

7

## 2. ELEMENTOS BÁSICOS DE UN PROGRAMA DE ORDENADOR

### E.VARIABLES.

- Espacios en memoria reservados para almacenar información.
- Los programas trabajan con información y se guarda en variables.

Pasos:

- Declarar variable: tipo + nombre de identificador.
- Reglas para crear el nombre de identificador:
  1. Deben empezar con una letra o guion bajo.
  2. Pueden contener números, excepto en la primera posición.
  3. No pueden llevar acentos.
  4. No pueden llevar signos, excepto guion bajo.
  5. No pueden llevar espacios.
  6. C# SENSIBLE A MAYÚSCULAS Y MINÚSCULAS.



8

8

## 2. ELEMENTOS BÁSICOS DE UN PROGRAMA DE ORDENADOR

### E.VARIABLES.

Normas de nomenclatura para identificadores de nombre de variables (Notación Camello):

- Nombre referente a lo que contiene.
- Empezar por minúscula.
- Segundas palabras por mayúscula.



9

9

## 2. ELEMENTOS BÁSICOS DE UN PROGRAMA DE ORDENADOR

### E. VARIABLES. SISTEMAS DE TIPOS DE C#.

#### - Tipos de valor.

##### - Tipos simples.

- Entero con signo
- Entero sin signo
- Caracteres Unicode.
- Punto flotante binario.
- Punto flotante decimal
- Booleano.

##### - Tipos de enumeración.

##### - Tipos de estructura.

##### - Tipos de valores que aceptan valores NULL.

#### - Tipos de referencia:

- Tipo de clase.
- Tipo de interfaz.
- Tipos de matriz.
- Tipos delegados.

<https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/language-specification/types#simple-types>



10

10

## 2. ELEMENTOS BÁSICOS DE UN PROGRAMA DE ORDENADOR

### E.VARIABLES. DECLARACIÓN.

```
int numeroLibros;  
int librosIngles;  
int librosMatematicas;  
double promedioFinal;  
  
int numeroLibros, librosIngles, librosMatematicas;
```



11

11

## 2. ELEMENTOS BÁSICOS DE UN PROGRAMA DE ORDENADOR

### E.VARIABLES. INICIALIZACIÓN.

Si no damos un valor inicial a la variable C# le dará uno por defecto.

```
//Declaración e inicialización en varias líneas.  
int numeroLibros;  
numeroLibros = 500;  
  
//Declaración e inicialización en una línea.  
int numeroLibros = 500;  
int numeroLibros = 500, librosIngles = 200, librosMatematicas = 100;
```



12

12

## 2. ELEMENTOS BÁSICOS DE UN PROGRAMA DE ORDENADOR

### E.VARIABLES. INICIALIZACIÓN.

```
char salon;  
salon = 'A'; //Asignamos un carácter con comillas simples  
  
string saludo = "Hola"; //Asignamos varios caracteres con comillas  
dobles  
  
String saludo = "Hola"; //Tipo de dato de referencia y clase, puede  
escribirse con mayúscula y minúscula.  
  
bool x = true;  
bool y = false;
```



13

13

## 2. ELEMENTOS BÁSICOS DE UN PROGRAMA DE ORDENADOR

### E.VARIABLES. MOSTRAR VALORES EN LA CONSOLA:

```
double precioCamisa = 100;  
string colorCamisa = "Azul";  
Console.WriteLine("El precio de la camisa es {0} y su color es {1}",  
precioCamisa, colorCamisa);
```



14

14

## 2. ELEMENTOS BÁSICOS DE UN PROGRAMA DE ORDENADOR

### F.COMENTARIOS DE PROGRAMA.

- Sirven para documentar ciertas partes del programa.
- No afecta el código del programa.
- Es ignorado por el compilador

```
//comentario de una línea  
/*bloque de comentario de varias líneas se escribe  
* de esta manera */
```



15

15

## 2. ELEMENTOS BÁSICOS DE UN PROGRAMA DE ORDENADOR

### G. OPERADORES ARITMÉTICOS BINARIOS.

- SUMA DE NÚMEROS

```
int num1 = 8;  
double num2 = 4.5;  
double resultado;  
resultado = num1 + num2;  
Console.WriteLine(resultado);
```

- SUMA DE CADENAS O CONCATENAR

```
string saludo = "Hola", nombre = "Hugo";  
Console.WriteLine(saludo + " " + nombre);
```



16

16



## 2. ELEMENTOS BÁSICOS DE UN PROGRAMA DE ORDENADOR

### G. OPERADORES ARITMÉTICOS BINARIOS.

#### - RESTA

```
Console.WriteLine(50 - 8);
```

#### - MULTIPLICACIÓN

```
Console.WriteLine(2 * 2);
```



17

17

## 2. ELEMENTOS BÁSICOS DE UN PROGRAMA DE ORDENADOR

### G. OPERADORES ARITMÉTICOS BINARIOS.

#### - DIVISIÓN

```
//Resultado da 6, no 6.5  
Console.WriteLine(13 / 2);  
//Resultado da 6.5  
Console.WriteLine(13 / 2.0);  
//Precisión de tipo flotante de 6 a 9 dígitos.  
Console.WriteLine(16.8f / 4.1f);  
//Precisión tipo double de 15 a 17 dígitos.  
Console.WriteLine(16.1d / 4.1d);  
//Precisión tipo decimal de 28 a 29 dígitos.  
Console.WriteLine (16.1m / 4.1m);
```



18

18

## 2. ELEMENTOS BÁSICOS DE UN PROGRAMA DE ORDENADOR

### G. OPERADORES ARITMÉTICOS BINARIOS.

- RESTO, RESÍDUO O MÓDULO.

```
Console.WriteLine(5 % 2); //1  
Console.WriteLine(5.9 % 3.1); //2.8
```



19

19

## 2. ELEMENTOS BÁSICOS DE UN PROGRAMA DE ORDENADOR

### H. JERARQUÍA DE OPERACIONES.

- Operadores de incremento `x++` y decremento `x--` posfijos.
- Operadores de incremento `++x` y decremento `--x` prefijos, y operadores unarios `+` y `-`.
- Operadores de multiplicación `*`, `/` y `%`.
- Operadores de suma adición `+` y `-`.

Los operadores con el mismo nivel de prioridad se evalúan de izquierda a derecha.

Use los paréntesis, `()`, para cambiar el orden de evaluación impuesto por la prioridad y la asociatividad de operadores.

```
https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/operators/arithmetic-operators
```



20

20

## 2. ELEMENTOS BÁSICOS DE UN PROGRAMA DE ORDENADOR

### I. MÉTODO READLINE().

- Pedirle datos al usuario.

```
string nombre;  
Console.Write("Dime tu nombre: ");  
nombre = Console.ReadLine();  
Console.WriteLine("Hola {0}, un placer", nombre);
```



21

21

## 2. ELEMENTOS BÁSICOS DE UN PROGRAMA DE ORDENADOR

### J. CONVIRTIENDO UNA CADENA EN TIPO NUMÉRICO.

- Las cadenas que nos da el usuario por teclado debemos convertirlas para poder operar con ellas.

```
string entrada;  
int num1 = 5, num2, resultado;  
  
Console.Write("Dame un número para sumarlo: ");  
entrada = Console.ReadLine();  
  
num2 = Convert.ToInt32(entrada); //Convertimos cadena a int  
resultado = num1 + num2;  
  
Console.WriteLine("El resultado es {0}: ", resultado);
```



22

22

## 2. ELEMENTOS BÁSICOS DE UN PROGRAMA DE ORDENADOR

### J. CONVIRTIENDO UNA CADENA EN TIPO NUMÉRICO.

- Las cadenas que nos da el usuario por teclado debemos convertirlas para poder operar con ellas.

```
string entrada;  
int num1 = 5, num2, resultado;  
  
Console.Write("Dame un número para sumarlo: ");  
entrada = Console.ReadLine();  
  
num2 = Int32.Parse(entrada); //Convertimos cadena a int  
  
resultado = num1 + num2;  
Console.WriteLine("El resultado es {0}: ", resultado);
```

La diferencia entre usar la clase Convert y el método Parse() es... Convert devuelve un valor de cero en caso de que haya un error, y Parse genera una excepción deteniendo la ejecución.



23