

C# tarea inicial

Day Martinez, 2024-1736, viernes 24

1. Declarar variable de los diferentes tipos, asignarles valor e imprimir el valor.

Tipos de Variables en C#

C# es un lenguaje de programación tipado estáticamente, lo que significa que el tipo de cada variable necesita ser definido claramente en el código.

Aquí te presentamos los tipos más comunes:

- **Int**: Para números enteros.
- **Double**: Para números decimales.
- **Char**: Para caracteres individuales.
- **String**: Para secuencias de caracteres o textos.
- **Bool**: Para valores verdaderos o falsos.

Declarar variables en C# es un paso fundamental para el manejo de datos dentro de tus programas. En C#, debes especificar el tipo de la variable y asignarle un nombre según las convenciones del lenguaje.

La sintaxis para declarar una variable en C# es:

tipoDeDato nombreDeVariable;

Donde **tipoDeDato** es uno de los tipos de datos soportados por C# como **int**, **string**, **double**, **bool**, etc., y **nombreDeVariable** es el identificador que escoges para la variable.

- **Inicialización de Variables**

Además de declarar una variable, a menudo querrás inicializarla asignándole un valor inicial. La inicialización se puede hacer en la misma línea de la declaración:

```
1
2  int edad = 25;
3  float precio = 12.20;
4  double distancia = 12345.6789;
5  char inicial = 'J';
6  string nombre = "Ana López";
7  bool esMayorDeEdad = true;
8
```

C# tarea inicial

- **Cómo Imprimir Variables en C#**

Uso del Método Console.WriteLine()

`Console.WriteLine()` es el método más utilizado para imprimir valores en la consola. Este método imprime la información seguido de un salto de línea.

```
1  
2  int edad = 30;  
3  Console.WriteLine(edad);  
4
```

Este código imprimirá el valor de la variable edad en la consola.

Impresión con Cadenas Concatenadas

Puedes concatenar cadenas y variables para formar un mensaje más estructurado.

```
1  
2  string nombre = "Ana";  
3  int edad = 25;  
4  Console.WriteLine("Nombre: " + nombre + ", Edad: " + edad);  
5
```

Este enfoque es directo pero puede volverse menos legible a medida que la concatenación se hace más compleja.

Uso de Interpolación de Cadenas

La interpolación de cadenas, introducida en C# 6, permite insertar variables directamente dentro de una cadena literal, mejorando la legibilidad.

```
1  
2  string nombre = "Carlos";  
3  int edad = 28;  
4  Console.WriteLine($"Nombre: {nombre}, Edad: {edad}");  
5
```

Este método es más limpio y fácil de leer, especialmente con múltiples variables.

C# tarea inicial

Formato de Cadenas con String.Format()

String.Format() permite formatear una cadena en un formato específico, colocando las variables en los lugares indicados por índices.

```
1
2 string nombre = "Laura";
3 int edad = 32;
4 Console.WriteLine(String.Format("Nombre: {0}, Edad: {1}", nombre, edad));
5
```

Esta forma es útil cuando necesitas un control más detallado sobre el formato de salida.

2. **Buscar cómo se declara una constante en C# e imprimir el valor. Probar de cambiar su valor luego y ver que es lo que pasa.**

Sintaxis para Declarar Constantes

```
1
2 const tipo nombreDeLaConstante = valor;
3
```

- **tipo:** Especifica el tipo de datos de la constante
- **nombre:** Es el nombre único que se le da a la constante
- **valor:** Es el valor constante que se asigna a la constante

Ejemplo simple:

```
const double Pi = 3.14159;

Console.WriteLine("El valor de Pi es: " + Pi);
```

Este código declara una constante de tipo double llamada Pi, le asigna un valor aproximado de pi, y nos imprime: `El valor de Pi es: 3.14159`

Pero si luego intento cambiar la constante me da un error de compilación y no me da ningún valor.

C# tarea inicial

```
const double Pi = 3.14159;  
  
Console.WriteLine("El valor de Pi es: " + Pi);  
  
Pi = 3.14; //No puedo hacer esto porque Pi es constante
```

3. Declara un entero, incrementarlo, decrementarlo, hacer operaciones con el.

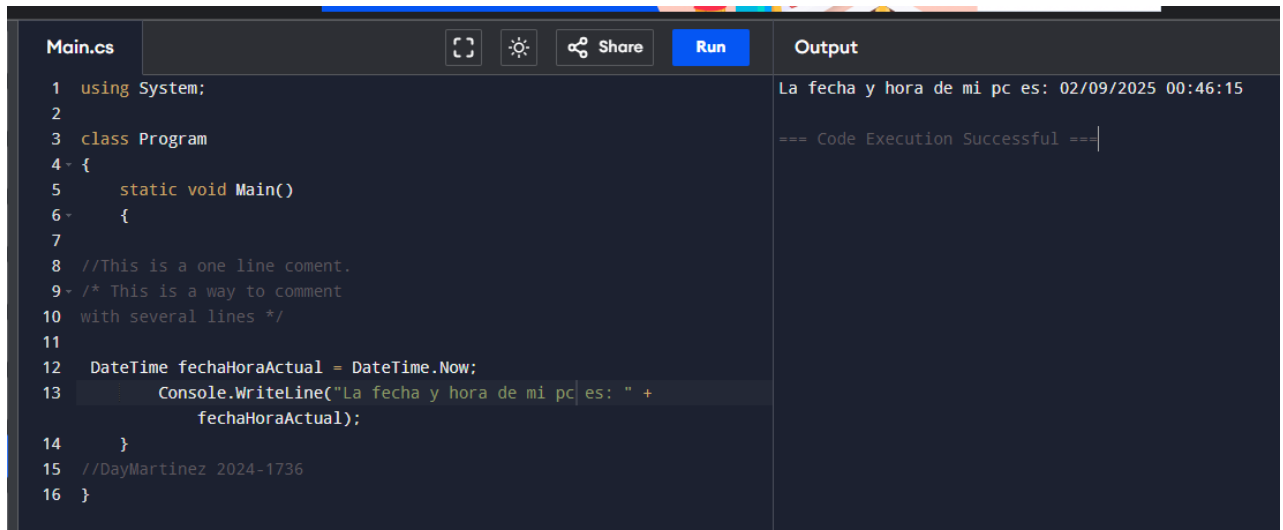
Main.cs	Output
<pre>1 using System; 2 3 class Program 4 { 5 static void Main() 6 { 7 8 int num = 17; 9 num = num + 2; 10 num = num - 4; 11 int result = num * 3; 12 Console.WriteLine("El valor final del entero es: " + result); 13 //DayMartinez 2024-1736 14 } 15 }</pre>	<pre>El valor final del entero es: 45 === Code Execution Successful ===</pre>

4. Declarar un float con valor=10152466.25f. Declara un byte que es igual a 5 + el float.

Main.cs	Output
<pre>1 using System; 2 3 class Program 4 { 5 static void Main() 6 { 7 8 float value1 = 10152466.25f; 9 byte value2 = (byte)(5 + value1); 10 11 Console.WriteLine("El valor del byte es de " + value2); 12 13 14 //DayMartinez 2024-1736 15 } 16 }</pre>	<pre>El valor del byte es de 23 === Code Execution Successful ===</pre>

C# tarea inicial

5. Adjuntar comentario de una y de varias líneas un su código. Imprimir la fecha y hora del sistema.



The screenshot shows a code editor with a file named 'Main.cs'. The code is as follows:

```
1 using System;
2
3 class Program
4 {
5     static void Main()
6     {
7
8         //This is a one line coment.
9         /* This is a way to comment
10        with several lines */
11
12        DateTime fechaHoraActual = DateTime.Now;
13        Console.WriteLine("La fecha y hora de mi pc es: " +
14                           fechaHoraActual);
15
16    }
17 }
```

The 'Output' window on the right shows the result of the program execution:

```
La fecha y hora de mi pc es: 02/09/2025 00:46:15
=== Code Execution Successful ===
```