

ILERNA

Online

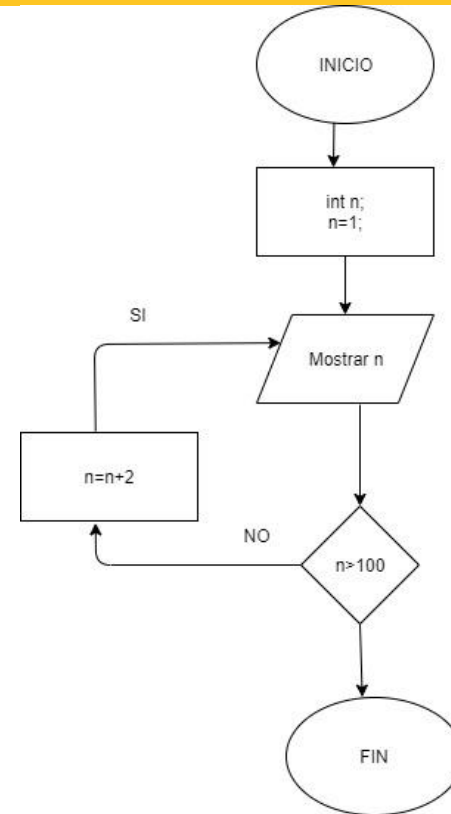
Videotutoría 2: Aprendiendo a programar

Módulo 03A: Programación



RESUMEN Videotutoría 1

Escribir un programa donde se muestre por pantalla los números impares desde el 1 al 99.



Visual Studio 2019: Inicio de un proyecto

Crear un proyecto

Plantillas de proyecto recientes

- Aplicación de consola (.NET Framework) C#

Lenguaje | Plataforma | Tipo de proyecto

- Aplicación de WPF (.NET Framework)
Aplicación cliente de Windows Presentation Foundation.
C# Windows Escritorio
- Biblioteca de clases (.NET Standard)
Proyecto para crear una biblioteca de clases para .NET Standard.
C# Android iOS Linux macOS Windows Biblioteca
- Solución en blanco
Crea una solución vacía sin proyectos.
Otros
- Aplicación de consola (.NET Framework)**
Proyecto para crear una aplicación de línea de comandos.
C# Windows **Consola**
- Aplicación de Windows Forms (.NET Framework)
Proyecto para crear una aplicación con una interfaz de usuario de Windows Forms (WinForms).
C# Windows Escritorio
- Biblioteca de clases (.NET Framework)
Proyecto para crear una biblioteca de clases de C# (.dll).
C# Windows Biblioteca

Atrás | Siguiente

Visual Studio 2019: Mi proyecto

Configure su nuevo proyecto

Aplicación de consola (.NET Framework) C# Windows Consola

Nombre del proyecto

Mi Primer proyecto

Ubicación

C:\Users\jmartin\source\repos

Nombre de la solución ⓘ

Mi Primer proyecto

Colocar la solución y el proyecto en el mismo directorio

Framework

.NET Framework 4.7.2

Atrás Crear

Visual Studio 2019: Estructura de un programa

```
Program.cs [X]
C# Mi Primer proyecto
1  [Lightbulb] using System;           Los using son librerías que se cargan automáticamente. Nos sirve sobretodo para
2  using System.Collections.Generic;   sacar mensajes por pantalla. Y más cosas.
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace Mi_Primer_proyecto
8  {
9      class Program
10     {
11         static void Main(string[] args)
12         {
13             Aquí se programa.
14         }
15     }
16 }
```

Conversiones de datos

- *Implícita*: no es necesario indicar ningún tipo de conversión.
 - P.ej: `int numero = 10; double numero2 = numero + 3;`
- *Explícita (casting)*: Una conversión de tipos es una manera de informar explícitamente al compilador de que se pretende realizar la conversión y se es consciente de que se puede producir pérdida de datos.
 - P.ej: `double x = 12.25; int y; y = (int)x` -> salida: 12
- *Explícita (casting)*: Convierte la representación en forma de cadena

```
Oreferencias
static void Main(string[] args)
{
    int numero;
    Console.WriteLine("Dame un número:");
    numero = Console.ReadLine();

    //numero = int.Parse
    Console.WriteLine(n

Excepciones:
System.IO.IOException
OutOfMemoryException
ArgumentOutOfRangeException
No se puede convertir implícitamente el tipo 'string' en 'int'
```

```
7 namespace Mi_Primer_proyecto
8 {
9     Oreferencias
10    class Program
11    {
12        Oreferencias
13        static void Main(string[] args)
14        {
15            int numero;
16            Console.WriteLine("Dame un número:");
17            numero = int.Parse(Console.ReadLine());
18
19            //double numero2 = numero + 3;
20
21            Console.WriteLine("Salida de mi numero {0}", numero);
22            Console.ReadKey();
23        }
24    }
25 }
```

C:\Users\jmartin\source\rep
Dame un número:
7
Salida de mi numero 7

Formato de cadenas

- Por ejemplo:

```
namespace Mi_Primer_proyecto
{
    0 referencias
    class Program
    {
        0 referencias
        static void Main(string[] args)
        {
            int numero;
            Console.WriteLine("Dame un número:");
            numero = int.Parse(Console.ReadLine());

            //double numero2 = numero + 3;
            Console.WriteLine("Salida de mi numero {0}", numero);
            Console.ReadKey();
        }
    }
}
```

Parámetro = variable

```
C:\Users\jmartin\source\re
Dame un número:
5
Salida de mi numero 5
```

Pondremos entre corchetes todos los parametros que tengamos empezando por el 0: {0} {1} {2}...


Formato de cadenas

- ¿Qué salida produciría este código?

```
class Program
{
    Oreferencias
    static void Main(string[] args)
    {
        int numero;
        Console.WriteLine("Dame un número:");
        numero = int.Parse(Console.ReadLine());

        //double numero2 = numero + 3;

        Console.WriteLine("Salida de mi numero {0}", numero, "hola");
        Console.ReadKey();
    }
}
```



Comentarios en el código

```
class Program
{
    //referencias
    static void Main(string[] args)
    {
        int numero;
        Console.WriteLine("Dame un número:");
        numero = int.Parse(Console.ReadLine());

        /* Esto es un comentario
         * y seguimos comentando
         */

        //double numero2 = numero + 3;

        Console.WriteLine("Salida de mi numero {0}", numero, "hola");
        Console.ReadKey();
    }
}
```

Operadores

Operadores aritméticos

+	Suma aritmética de dos valores.
++	Incremento en 1 del operando.
-	Resta aritmética de dos valores.
--	Decremento en 1 del operando.
*	Multiplicación aritmética de dos valores.
/	División aritmética de dos valores.
%	Obtiene el resto de la división entera.

operador de asignación y suma: $x += y$ es lo mismo que $x = x + y$

Operadores de comparación

>	Mayor.
<	Menor.
>=	Mayor o igual.
<=	Menor o igual.
==	Igual.
!=	desigualdad.
=	Asignación.

Operadores lógicos booleanos

&& AND o Y lógico.

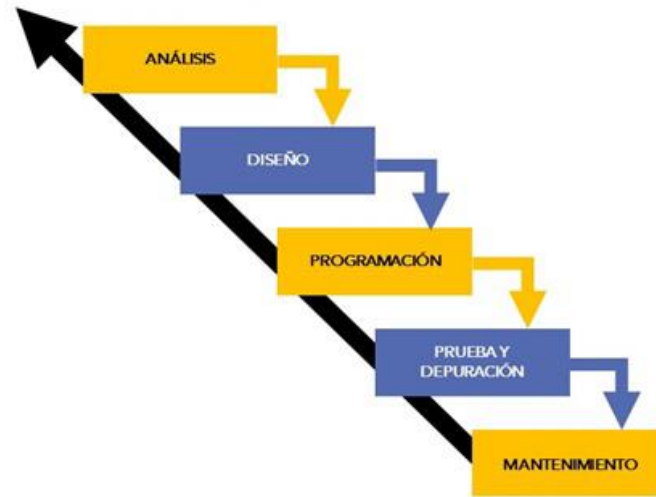
|| OR u O lógico.

! NOT o No lógico.

```
namespace Mi_Primer_proyecto
{
    0 referencias
    class Program
    {
        0 referencias
        static void Main(string[] args)
        {
            Boolean mivariable=false;
            if (!mivariable) { Console.WriteLine("SALIDA 1 "); }

            Console.WriteLine("SALIDA 2");
        }
    }
}
```

Programación estructurada: ciclo de vida



Programación estructurada

GUÍA DIDÁCTICA

- **Edsger Dijkstra** comprobó que todo programa se puede escribir utilizando únicamente tres instrucciones de control:
 - Secuencia de instrucciones.
 - Instrucción condicional. (if/If...else)
 - Iteración o bucle de instrucciones. (While/Do...While/for)

EJERCICIOS

Resuelve estos ejercicios utilizando un IDE en C#:

- Escribir un programa donde sume los primeros diez números naturales. Por definición convencional, se dirá que cualquier miembro del siguiente conjunto, $\mathbb{N} = \{1, 2, 3, 4, \dots\}$, es un número natural
- Escribir un programa donde se lea por teclado diez números y realizar la suma y la media de los mismos.
- Escribir un programa donde se muestre por pantalla los números impares desde el 1 al 99. (++)
- Diseña un programa que calcule la suma, resta, multiplicación y división de dos números introducidos por teclado.

EJERCICIOS (II)

Desarrolla estos ejercicios utilizando secuencias de instrucciones

- Realiza un programa que pida un número, y calcule el perímetro de la circunferencia y el área del círculo con ese radio. Ten en cuenta que el área de un círculo es $a = \text{PI} * r * r$, y el perímetro de la circunferencia es $p = 2 * \text{PI} * r$.
- Realiza un programa que calcule el coste del precio de la gasolina en un viaje. El usuario introducirá los kilómetros a recorrer, el precio de la gasolina y el consumo del coche en litros por cada 100 kilómetros.
- Cree un programa que realiza la transformación de una cantidad de segundos en horas, minutos y segundos. La cantidad de segundos se introduce por teclado.
- Realiza un programa que muestra los resultados de una encuesta en porcentajes. En la encuesta se puede contestar SI, NO o NO SABE-NO CONTESTA. Por teclado únicamente se introducirá el número de respuestas de cada categoría.

