

## Taller de Lenguajes I - 2025

Programador Universitario / Licenciatura en Informática / Ingeniería en Informática

Trabajo Práctico Nro 6

### Repositorio.

Crear una carpeta nueva en el disco e inicialice el siguiente repositorio en esa ubicación: <https://tinyurl.com/tl1-2025-tp6>

### Ejercicios propuestos para entrar en contacto con el lenguaje

Vamos a realizar algunos ejercicios para poder conocer un poco mejor el lenguaje, comenzaremos por un proyecto del tipo Consola. Siga los pasos indicados por el docente en clase para crear un proyecto de este tipo.

Configurando el entorno

1. Descargar e instalar el SDK .Net (Version 8) . Puede hacerlo de dos maneras:
  - a. Ingresar a <https://dotnet.microsoft.com/en-us/download> ,descargar e instalar
  - b. Ingresar a la consola de su preferencia (consola integrada de vscode, cmd, Powershell, etc.) y ejecutar la línea de comando **winget install Microsoft.DotNet.SDK.8**
2. Desde la terminal ejecutar comando **dotnet --info** (para ver si está instalado correctamente)

```
Windows PowerShell
PS C:\Users\Javier>
PS C:\Users\Javier> dotnet --info
SDK DE .NET:
Version:      8.0.204
Commit:      c338c7548c
Workload version: 8.0.200-manifests.7d36c14f

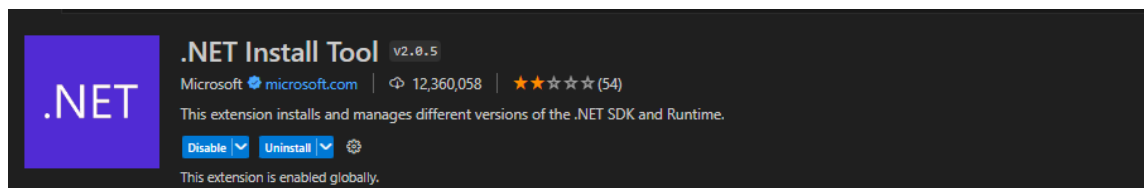
Entorno de tiempo de ejecución:
OS Name:      Windows
OS Version:   10.0.19045
OS Platform:  Windows
RID:          win-x64
Base Path:    C:\Program Files\dotnet\sdk\8.0.204\

Cargas de trabajo de .NET instaladas:
[android]
  Origen de la instalación: VS 17.9.34728.123
  Versión del manifiesto:   34.0.52/8.0.100
  Ruta de acceso del manifiesto: C:\Program Files\dotnet\sdk-manifests\8.0.100\microsoft.net.sdk.android\34.0.52\
  WorkloadManifest.json
  Tipo de instalación:      FileBased

[maui-windows]
  Origen de la instalación: VS 17.9.34728.123
  Versión del manifiesto:   8.0.7/8.0.100
  Ruta de acceso del manifiesto: C:\Program Files\dotnet\sdk-manifests\8.0.100\microsoft.net.sdk.maui\8.0.7\Workl
  oadManifest.json
  Tipo de instalación:      FileBased

[maccatalyst]
```

3. En VS Code, abra la carpeta del repositorio en e instale la Extensión C# (.NET INSTALL TOOL)

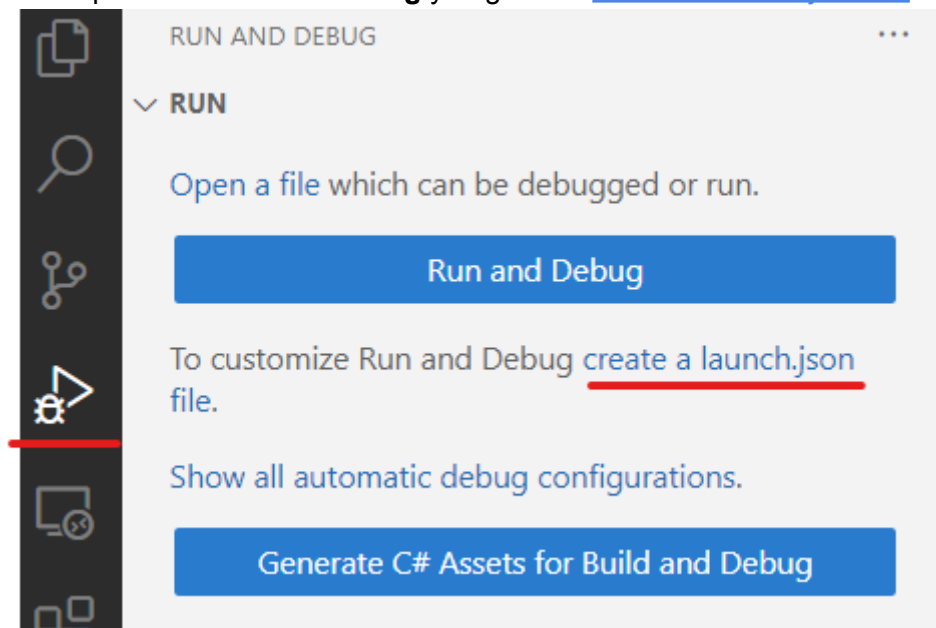


4. Desde la consola integrada de VS Code tipee la siguiente línea de comandos:  
**dotnet new console**

5. Agregue el archivo **.gitignore** correspondiente ingresando a *gitignore.io* e ingresando las palabras clave **Csharp**, **VisualStudioCode** y **DotnetCore**.
6. Abra la consola de vs-code en la carpeta del proyecto (es decir la carpeta donde se descargó el repositorio) y ejecute el comando `dotnet run` para correr el programa. Debería verse en la consola la línea "Hello World!".
7. Abra el archivo `Program.cs` (editar el código) y agregue el siguiente código

```
Console.WriteLine("Hello, World!");  
  
int a;  
int b;  
a=10;  
b=a;  
  
Console.WriteLine("valor de a:"+a);  
Console.WriteLine("valor de b:"+b);
```

8. Ir a la pestaña **Run and Debug** y haga click ["create a launch.json file"](#).



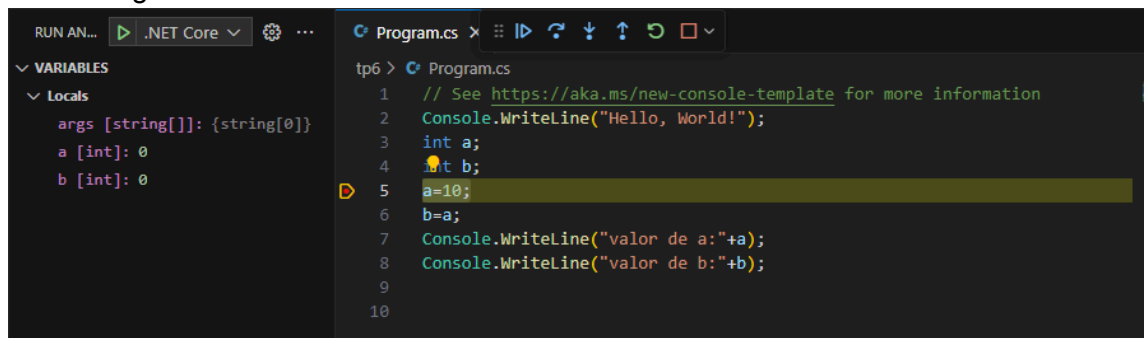
9. Abra el archivo `launch.json` y verifique que la llave **"console"** tenga el valor **"integratedTerminal"** y **NO "internalConsole"**.

```
{  
  "args": [],  
  "cwd": "${workspaceFolder}",  
  // For more information about the 'console' field, see https://aka.ms/VSCode-CS-LaunchJson-Console  
  "console": "integratedTerminal",  
  "stopAtEntry": false  
},
```

10. Seleccione el archivo `Program.cs`, y estando en la línea donde inicializa la variable **a=10** presione **f9** (tiene que aparecer el punto rojo a la izquierda de la línea)

```
3 int a;  
4 int b;  
5 a=10;  
6 b=a;  
7 Console.WriteLine("valor de a:"+a);  
8 Console.WriteLine("valor de b:"+b);  
9  
10
```

11. vaya nuevamente a la pestaña **Run and Debug** y haga clic en el botón “Run and Debug”.



El programa debería ejecutarse y al igual que en el punto 5, debería ejecutarse en la terminal la línea “Hello World!” y podrá ir “debuggando” el código normalmente

**Ejercicio 1.** Construir un programa que permita invertir un número. Verifique que el texto ingresado es de hecho un número y, en caso afirmativo, realice la inversión del número sólo si éste es mayor a 0.

**Nota:** Si observa un subrayado amarillo sugiriendo modificar las variables al tipo “nullable”, abra el archivo de configuración de proyecto de extensión .csproj y elimine la línea que dice `<Nullable>enable</Nullable>`. No olvide guardar el archivo.

**Para los ejercicios 2 y 3, cree dos branches en su repositorio, CalculadoraV1 y CalculadoraV2**

**Ejercicio 2.** Ingrese al branch **CalculadoraV1** y construya un programa que sea una calculadora que permita al usuario realizar las 4 operaciones básicas (Sumar, Restar, Multiplicar y Dividir) a partir de un menú para seleccionar la opción a elegir y que luego pida dos números y que devuelva el resultado de la operación seleccionada. Además una vez que termine de realizar la operación le pregunte si desea realizar otro cálculo.

**Ejercicio 3.** Ingrese al Branch **CalculadoraV2** para implementar las mejoras en la calculadora.. Solicite al usuario un número y muestre por pantalla:

- El valor absoluto de un número
- El cuadrado
- La raíz cuadrada
- El seno
- El Coseno
- La parte entera de un tipo float.

Luego de esto, solicite dos números al usuario y determine:

- El Máximo entre los dos números
- El Mínimo entre los dos números

Para TODOS los casos, no olvide contemplar siempre el caso de que el usuario no ingrese un número válido.

Suba al repositorio github ambas ramas.

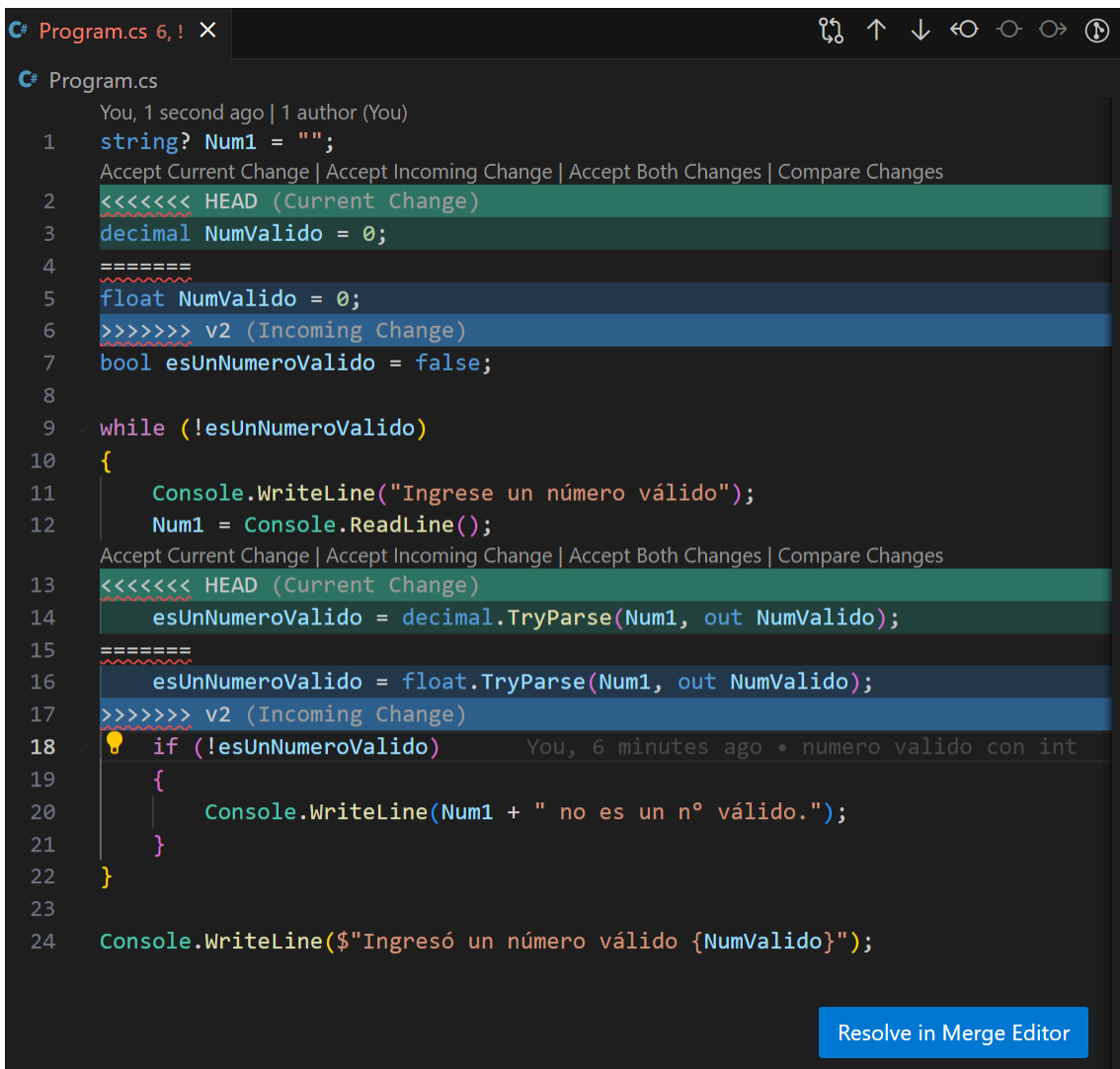
Combine el código del branch **CalculadoraV2** en el branch de **CalculadoraV1** y resuelva los conflictos surgidos.

### Instrucciones para combinar ramas (comando merge)

Ubicándose en el branch CalculadoraV1 ejecute el comando:

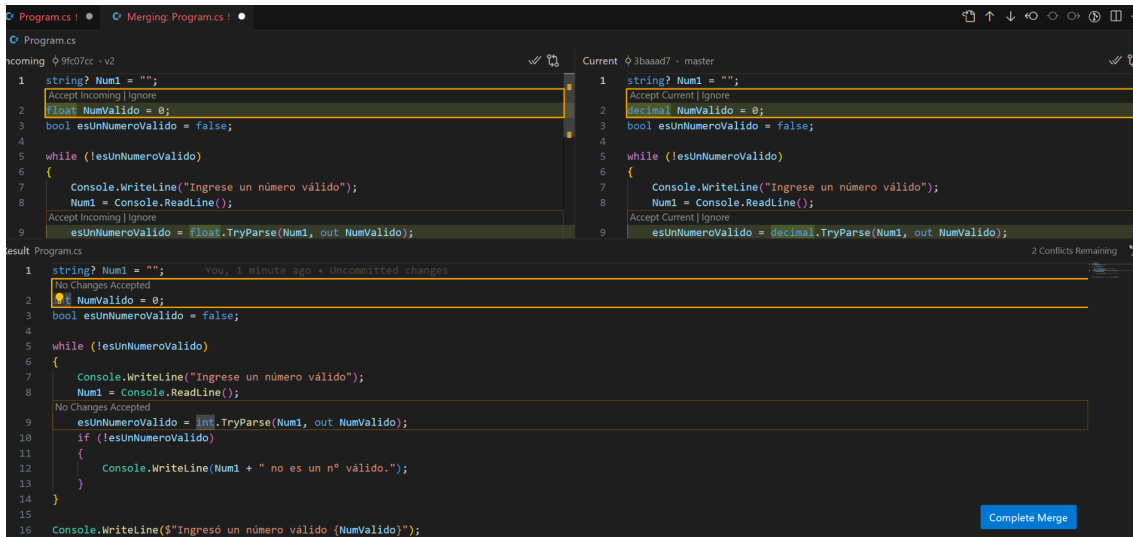
**>> git merge CalculadoraV2**

A continuación se le abrirá una nueva pestaña similar a la de la imagen donde se visualizarán ambas versiones ofreciendo diferentes opciones para combinar el código.



```
Program.cs 6, ! X
Program.cs
You, 1 second ago | 1 author (You)
1 string? Num1 = "";
2 <<<<<<< HEAD (Current Change)
3 decimal NumValido = 0;
4 =====
5 float NumValido = 0;
6 >>>>>>> v2 (Incoming Change)
7 bool esUnNumeroValido = false;
8
9 while (!esUnNumeroValido)
10 {
11     Console.WriteLine("Ingrese un número válido");
12     Num1 = Console.ReadLine();
13 <<<<<<< HEAD (Current Change)
14     esUnNumeroValido = decimal.TryParse(Num1, out NumValido);
15 =====
16     esUnNumeroValido = float.TryParse(Num1, out NumValido);
17 >>>>>>> v2 (Incoming Change)
18 if (!esUnNumeroValido) You, 6 minutes ago • numero valido con int
19 {
20     Console.WriteLine(Num1 + " no es un n° válido.");
21 }
22 }
23
24 Console.WriteLine($"Ingresó un número válido {NumValido}");
Resolve in Merge Editor
```

Si hace click en el botón “*Resolve in Merge editor*”, se abrirá una nueva pestaña donde visualizará las diferencias y como queda resuelto el conflicto con más claridad. Similar a la siguiente imagen.



La interfaz proporciona una visualización clara de los cambios aplicados en cada rama, mostrándolos en dos ventanas contiguas, mientras que en la ventana inferior se puede ver como quedará el resultado final del archivo.

Los rectangulos amarillos indican que es lo que cambia en cada archivo, y opciones para aceptar o ignorar dicho cambio.

Cuando esté decidido cómo debe quedar el archivo final, debe presionar el botón azul “Complete merge”

**Importante:** Para concluir el merge, una vez realizado, debe hacer un commit desde la consola para dejar asentados los cambios.

**Nota:** Al resolver los conflictos de merge, puede elegir entre aceptar los cambios de la versión entrante (Incoming), mantener la versión del branch actual (Current) o aceptar ambos cambios. Si desea abortar el merge, puede ingresar el comando `git merge --abort`.

## Trabajando con el tipo string

### Funciones útiles para realizar el ejercicio 4

A continuación se presenta una lista de métodos pertenecientes al objeto *string* para procesar cadenas de texto. Se recomienda investigar el comportamiento de cada uno a medida que los vaya necesitando para resolver el punto 4.

*Conversor de tipo:* ToString()

*Comparador de cadenas:* Compare(), CompareTo() == y !=

*Mayúsculas y minúsculas:* ToUpper() o ToLower()

*Acceso a los caracteres individuales:* SubString(), Replace(), Split() y Trim(),

*Búsqueda y manipulación de una cadena: IndexOf() LastIndexOf, StartsWith y EndsWith. Split()*

#### **Ejercicio 4.**

##### **Responder las siguientes preguntas en el archivo readme.md**

¿String es una tipo por valor o un tipo por referencia?

¿Qué secuencias de escape tiene el tipo string?

¿Qué sucede cuando utiliza el carácter @ y \$ antes de una cadena de texto?

##### **Realizar los siguientes ejercicios**

Dada una cadena (un string) de texto ingresada por el usuario, realice las siguientes tarea:

- Obtener la longitud de la cadena y muestre por pantalla.
- A partir de una segunda cadena ingresada por el usuario, concatene ambas cadenas distintas.
- Extraer una subcadena de la cadena ingresada.
- Utilizando la calculadora creada anteriormente realizar las operaciones de dos números y mostrar por pantalla y mostrar en texto el resultado. Por ejemplo para la suma sería:  
"la suma de " num1 " y de" num2 " es igual a: " resultado.

*Donde num1, num2 y resultados son los sumandos y el resultado de la operación respectivamente.*

*Nota:* Busque el comportamiento del Método *ToString()*;

- Recorrer la cadena de texto con un ciclo Foreach e ir mostrando elemento por elemento en pantalla
- Buscar la ocurrencia de una palabra determinada en la cadena ingresada
- Convierta la cadena a mayúsculas y luego a minúsculas.
- Ingrese una cadena separada por caracteres que usted determine y muestre por pantalla los resultados (Revisar el comportamiento de **split()**)
- Siguiendo con el ejemplo de la calculadora (ejercicio 2) ingrese una ecuación simple como cadena de caracteres y que el sistema lo resuelva. Por ej. ingrese por pantalla "582+2" y que le devuelva la suma de 582 con 2

#### **Ejercicio 5. (optativo)**

##### **Expresiones Regulares**

Busque en diferentes bibliografías que son y cómo funcionan las expresiones regulares.

- ¿Funcionan únicamente en C#?
- ¿En qué casos le parecen útiles? Enuncie al menos 3.
- ¿Cómo se hace uso de estas en C#?

##### **Resuelva las siguiente expresiones regulares**

**Taller de Lenguajes I - 2025**

Programador Universitario / Licenciatura en Informática / Ingeniería en Informática

Trabajo Práctico Nro 6

---

Construir un programa que permita identificar de forma sencilla si la cadena ingresada es una dirección web y otro que una cadena ingresada sea un mail válido.