



Manual de uso del lenguaje de programación

Este lenguaje de programación fue creado usando C# como lenguaje base para definir los parámetros para que funcione de acuerdo a la gramática puesta. Y se usó ANTLR4 para generar la gramática con la cual funcionará el lenguaje, generar los tokens y el analizador léxico del lenguaje.

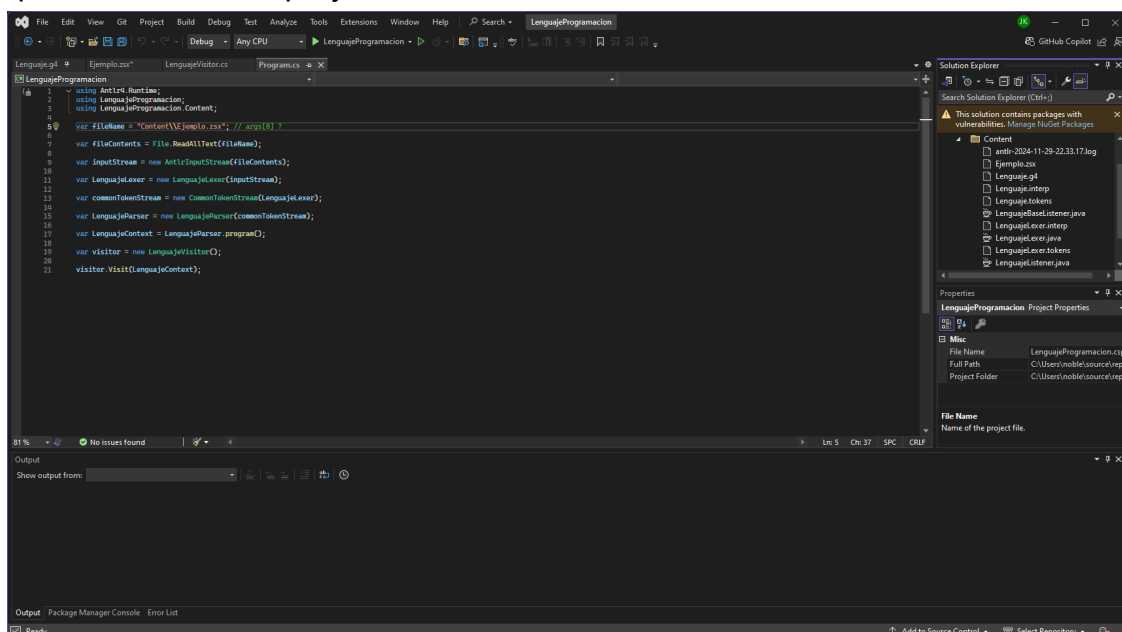
Como usarlo

Clona o descarga el repositorio:

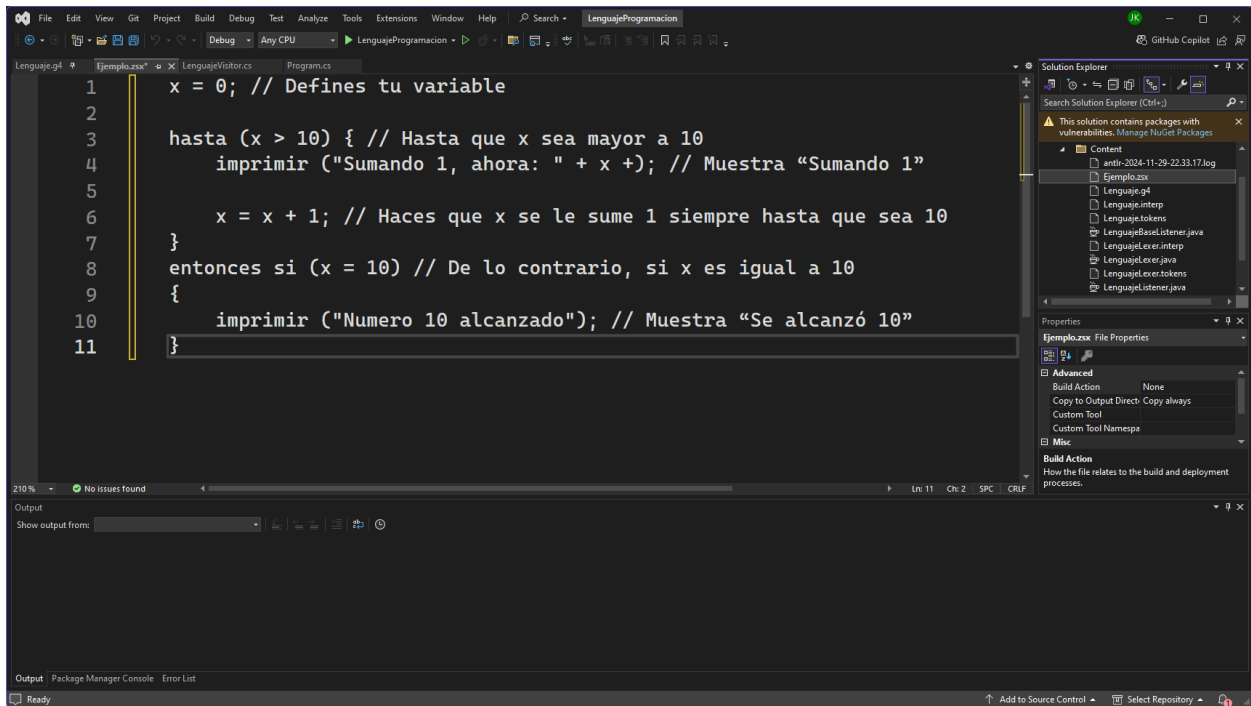
<https://github.com/MidoriWakana/LenguajeProgramacion>

	MidoriWakana	Completo	6349fcf · 47 minutes ago	4 Commits
	.vs	Completo	47 minutes ago	
	LenguajeProgramacion	Completo	47 minutes ago	
	.gitattributes	Initial commit	5 hours ago	
	LICENSE	Initial commit	5 hours ago	
	LenguajeProgramacion.sln	"Completo"	1 hour ago	
	README.md	Initial commit	5 hours ago	

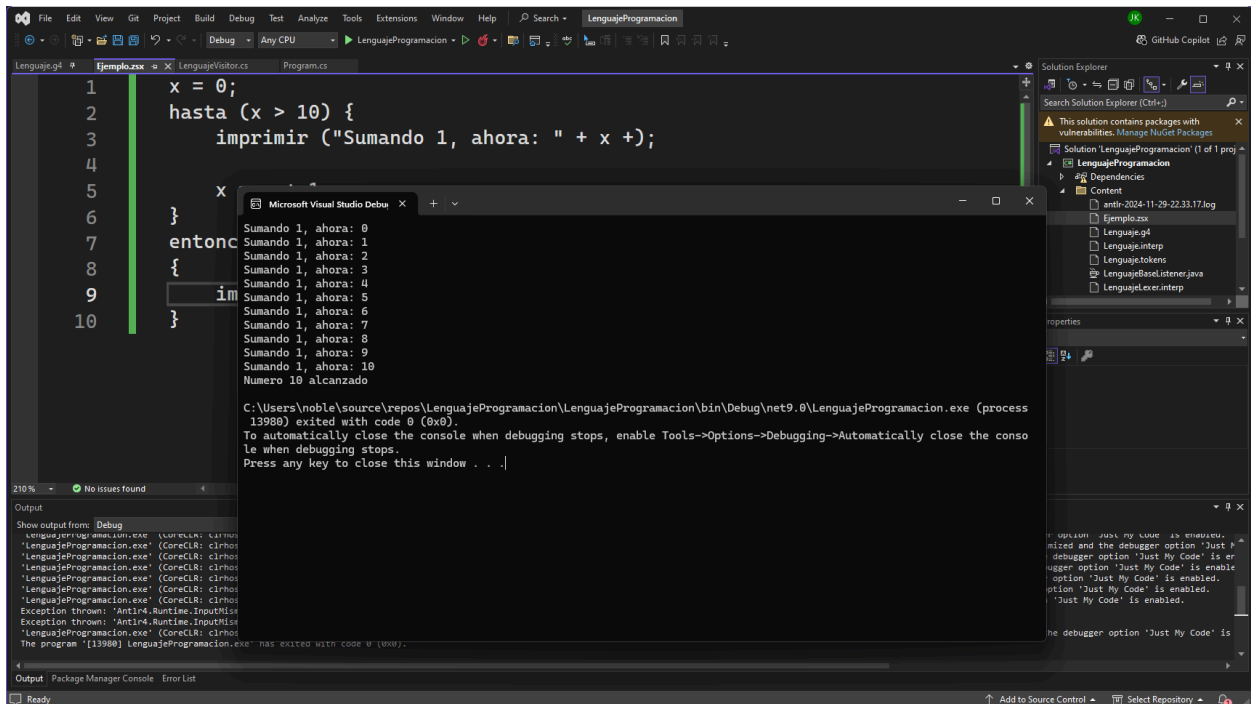
En Visual Studio Community, busca el proyecto y abre el LenguajeProgramacion.sln para poder visualizar el proyecto



Edita el archivo Ejemplo.zsx para poder empezar a escribir tu código



Para ejecutarlo, ponlo en Debug y ejecuta el proyecto. La salida se muestra en la ventana de comandos que aparece



Sintaxis

Accion	Definicion
Variables (x, y, z, etc)	Son las variables que se pueden definir, eliges la palabra que desees poner y el contenido de la variable: X = 1; Y = "Hola";
imprimir	Es el comando para que se imprima en pantalla lo que escribas, ya sea texto o número: imprimir ("Hola mundo"); imprimir (x);
hasta	Es un ciclo que sirve para definir hasta cuando se finaliza la condición que se desea poner: hasta (numero = 10);
mientras	Es un ciclo que sirve para definir que el número sea el que se pide, si es el número que se pide, entonces se realizará una acción. De lo contrario, se realizará otra acción con base a lo que se pida: mientras (numero < 10) { imprimir ("Este número no es 10"); }
si - entonces	Sirve para definir una condición, si tu número es tal, entonces el lenguaje realizara una opción, de lo contrario hará otra: si (numero = 10) { imprimir ("Este número es 10"); } entonces si (numero < 10) { imprimir ("Este número no es 10") }
para	Sirve para definir arreglos de números con base a lo que se necesite:

	<pre> para (numero = 0; numero < 10; numero = numero + 1) { imprimir(i); } </pre>
matriz[x]	Sirve para construir matrices y calcular los índices de la matriz.

Operadores

Operadores		
Simbolo	Descripción	Uso
+	Suma	$x + y$
-	Resta	$x - y$
*	Multiplicacion	$x * y$
/	Division	x / y
^	Potencia	$x ^ y$
raiz	Raiz	$x \text{ raiz } x$
=	Igual	$x = y$
>	Mayor	$x > y$
<	Menor	$x < y$
!=	Negacion	$x != y$
>=	Mayor Igual	$x >= y$
<=	Menor Igual	$x <= y$
PI	Pi	Pi / x
E	E	E

Codigos de prueba

Ejemplo de si - entonces:

```
z = 0; // Defines tu variable

si (z < 10) // Si z es menor a 10
{
    imprimir ("No es 10"); // Muestra en pantalla "No es 10"
}
entonces si (z = 10) // De lo contrario, si z es igual a 10
{
    imprimir ("Es 10"); // Muestra en pantalla "Es 10"
}
```

Ejemplo de mientras:

```
y = 0; // Defines tu variable

mientras (y < 10) // Mientras y sea menor a 10
{
    imprimir ("No es suficiente"); // Muestra "No es suficiente"
    imprimir ("Sumando 1: " + y +); // Muestra "Sumando 1: "

    y = y + 1; // Haces que y se le sume 1 siempre hasta que sea 10
}
entonces si (y = 10) // De lo contrario, si y es igual a 10
{
    imprimir ("Se alcanzo el numero 10"); // Muestra "Se alcanzó 10"
}
```

Ejemplo de hasta:

```
x = 0; // Defines tu variable

hasta (x > 10) { // Hasta que x sea mayor a 10
    imprimir ("Sumando 1, ahora: " + x +); // Muestra "Sumando 1"

    x = x + 1; // Haces que x se le sume 1 siempre hasta que sea 10
}
entonces si (x = 10) // De lo contrario, si x es igual a 10
```

```
{  
    imprimir ("Numero 10 alcanzado"); // Muestra "Se alcanzó 10"  
}
```

Ejemplo de para:

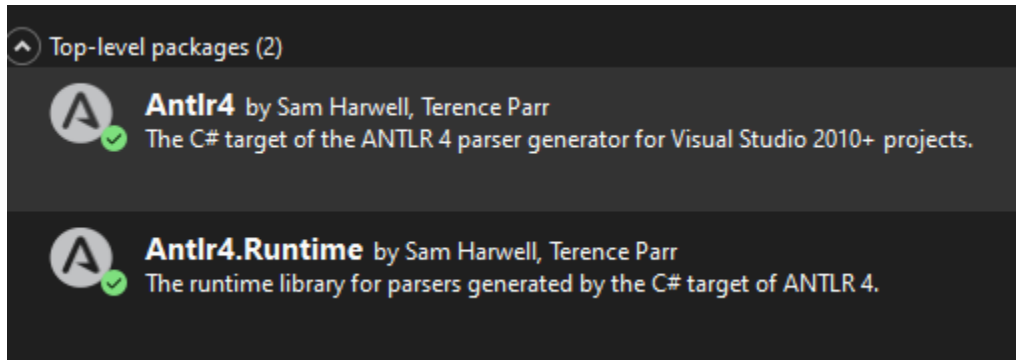
```
i = 0; // Defines tu variable  
  
para (i = 0; i < 10; i = i + 1) // Si i es menor a 10, se agrega 1  
{  
    imprimir(i); // Muestra la lista de números agregados  
}
```

Ejemplo de matriz:

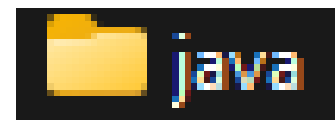
```
miMatriz = [          // Defines tu matriz y se muestra en pantalla  
    [1, 2, 3],  
    [4, 5, 6],  
    [7, 8, 9]  
];  
  
otraMatriz = [        // Defines tu matriz y se muestra en pantalla  
    [1, 2],  
    [4, 5]  
];  
  
imprimir ("Indice: " + miMatriz[0][1]); // Muestra el índice de la  
matriz
```

Recursos utilizados

Para este proyecto, se utilizaron las siguientes librerías. ANTLR4 y ANTLR4 RUNTIME del lado de C#



De lado de Windows, se usó ANTLR4 para crear el analizador léxico, árbol de tokens, etc. Y Java para que ANTLR4 pueda funcionar



Se usó .NET 9 como base del proyecto y el IDE de Visual Studio Community 2022

```
Command Prompt
Execute a .NET SDK command.

sdk-options:
  -d|--diagnostics  Enable diagnostic output.
  -h|--help         Show command line help.
  --info            Display .NET information.
  --list-runtimes   Display the installed runtimes.
  --list-sdks       Display the installed SDKs.
  --version         Display .NET SDK version in use.

SDK commands:
  add               Add a package or reference to a .NET project.
  build             Build a .NET project.
  build-server      Interact with servers started by a build.
  clean             Clean build outputs of a .NET project.
  format            Apply style preferences to a project or solution.
  help              Opens the reference page in a browser for the specified command.
  list              List packages or references of a .NET project.
  msbuild           Run Microsoft Build Engine (MSBuild) commands.
  new               Create a new .NET project or file.
  nuget             Provides additional NuGet commands.
  pack              Create a NuGet package.
  publish           Publish a .NET project for deployment.
  remove            Remove a package or reference from a .NET project.
  restore           Restore dependencies specified in a .NET project.
  run               Build and run a .NET project output.
  sdk               Manage .NET SDK installation.
  sln               Modify Visual Studio solution files.
  store             Store the specified assemblies in the runtime package store.
  test             Run unit tests using the test runner specified in a .NET project.
```

Errores

Este lenguaje de programación puede presentar unos cuantos errores, ya que se hizo en un día. En un futuro se podrían arreglar los errores, puede que algunos errores sean los siguientes:

1. Fallos logicos
2. Falta de gramatica
3. Duplicidad de impresión