



Objetivo

General

- Desarrollar un analizador léxico eficiente para un lenguajes de programación inspirado en Visual Basic, capaz de reconocer y clasificar correctamente los diferentes componentes léxicos presentes en el código fuente

Específico

- Implementar un analisis lexico para un lenguaje de programación inspirado en Visual Basic
- Implementar lógica y reglas de clasificación para cada tipo de token definido
- Generar una imagen resultante del código fuente

Descripción

La empresa LogiCode lo ha contratado para realizar la primera fase el cual está basado en Visual Basic, el cual es una analizador léxico. El cual consiste en que se escanea el código fuente, y a partir de este código se creará una imagen cuadrícula, donde cada cuadro se va pintar según el orden y tipo de token que fue analizado, más adelante se explicara esta parte.



Identificadores

- Los identificadores deben comenzar con una letra (A-Z o a-z).
- Pueden contener letras, dígitos (0-9) y el carácter de subrayado (_).
- Los identificadores no pueden comenzar con un dígito.



Color del Identificador:

 **#FFD300**

Ejemplos







- userName
- MaxUsers
- MAX_SIZE



Operadores







Los operadores son símbolos que son utilizados para realizar algún tipo de operación aritmética, Racional, Lógica.

Aritméticos




Nombre	Símbolo	Cuadro
Suma	+	 #FF33FF
Resta	-	 #C19A6B
Exponente	^	 #FCD0B4
División	/	 #B4D941
Módulo	Mod	 #D9AB41
Multiplicación	*	 #D80073



Relacionales o Comparación



Nombre	Símbolo	Cuadro
Igual	==	 #6A00FF
Diferente	<>	 #3F2212
Mayor que	>	 #D9D441
Menor que	<	 #D94A41
Mayor o Igual que	>=	 #E3C800
Menor o Igual que	<=	 #F0A30A

Lógicos


Nombre	Símbolo	Cuadro
y	And	 #414ED9
o	Or	 #41D95D
negación	Not	 #A741D9



Asignación






Nombre	Símbolo	Cuadro
Asignación Simple	=	 #41D9D4
Asignación Compuesta	+=, -=, *=, /=	 #FFFFFF

Palabras Reservadas

Nombre	Símbolo	Cuadro
Palabra Reservada	Module End Sub Main Dim As Integer String Boolean Double Char Console.WriteLine Console.ReadLine If Elseif Else Then While Do Loop For To Next Function Return Const	 #60A917



Tipos de Datos

Nombre	Símbolo	Cuadro
Entero	Cualquier número entero: 0, 1, 2, 3,...n	 #1BA1E2
Decimal	2.5, 32.02, 0.001	 #FFFF88
Cadena	"cadena", "Nombre", "123"	 #E51400
Booleano	True, False	 #FA6800
Carácter	'A', '2', '\$'	 #0050EF

Square.Color

Este será un token especial el cual tendrá que realizar una función especial a diferencia de los demás tokens.

La estructura es la siguiente:

Palabra Reservada Especial

Color Asignado

Square.Color(#ffffff, 1, 0)

Fila

Columna




Este token pintará el cuadro según el código de color asignado en el primer parámetro, este tipo de color es hexadecimal. Solo se usará la notación hexadecimal.

La Fila y Columna corresponde a la posición o cuadro que pintará del lienzo, a diferencia de los demás tokens a este se le puede especificar qué cuadro es el que se pintará.

También se podrá aceptar el token especial simple, donde tendrá la misma acción que el anterior, con la diferencia que pintará el cuadro según sea su secuencia dentro del código fuente, y el color que pintará será el especificado en el parámetro, el cual es el siguiente




Square.Color(#4fbad4)

Comentario



Nombre	Símbolo	Cuadro
Comentario	' este es un comentario	 #B3B3B3

Solo se deben aceptar comentarios de una línea.

Signos y Símbolos

Nombre	Símbolo	Cuadro
Paréntesis	()	 #9AD8DB
Llaves	{ }	 #DBD29A
Corchetes	[]	 #DBA49A



Coma	,	 #B79ADB
Punto	.	 #9ADBA6

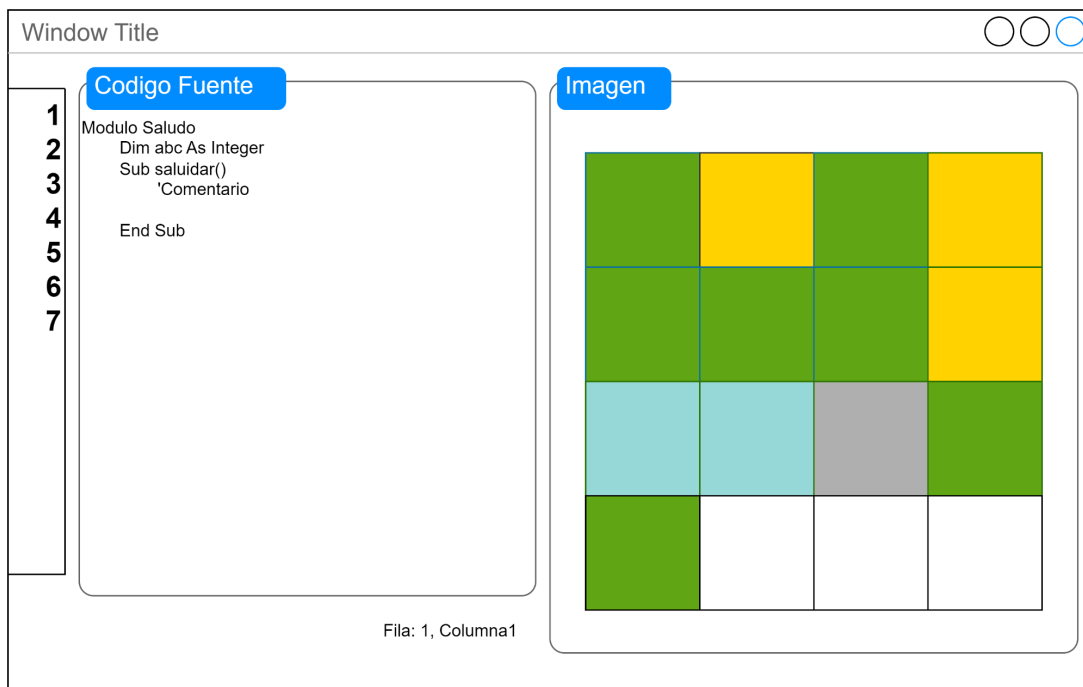
Características

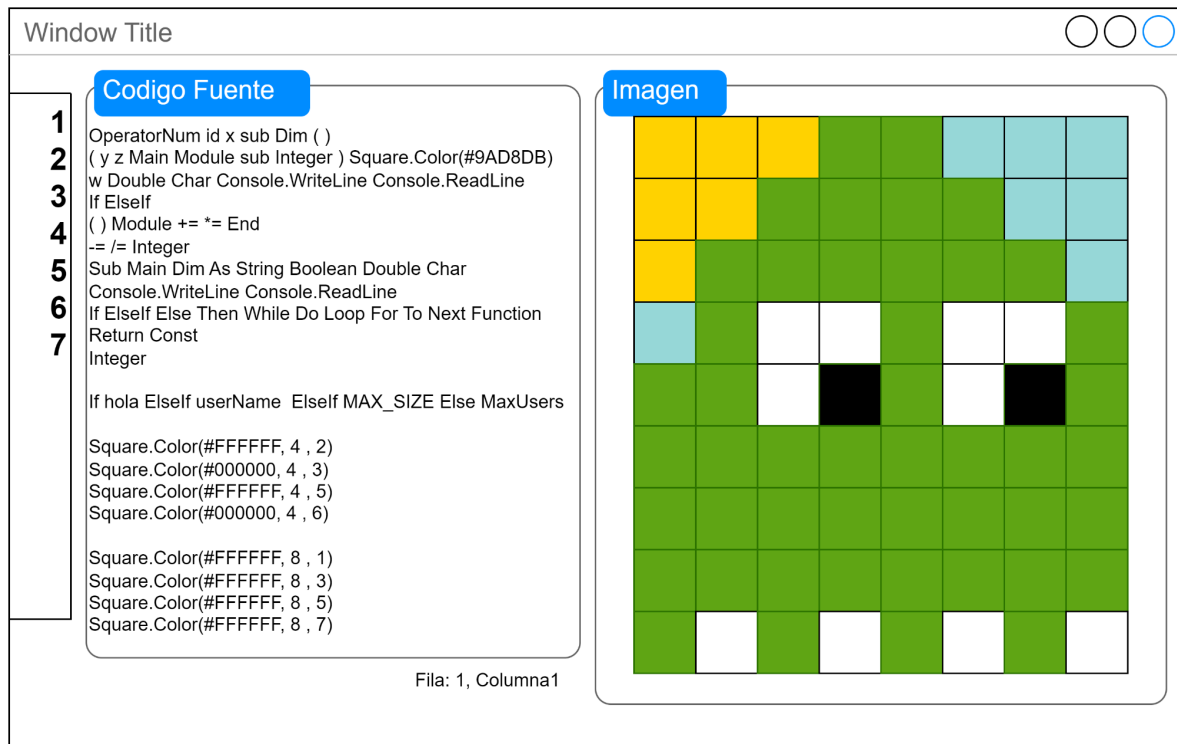
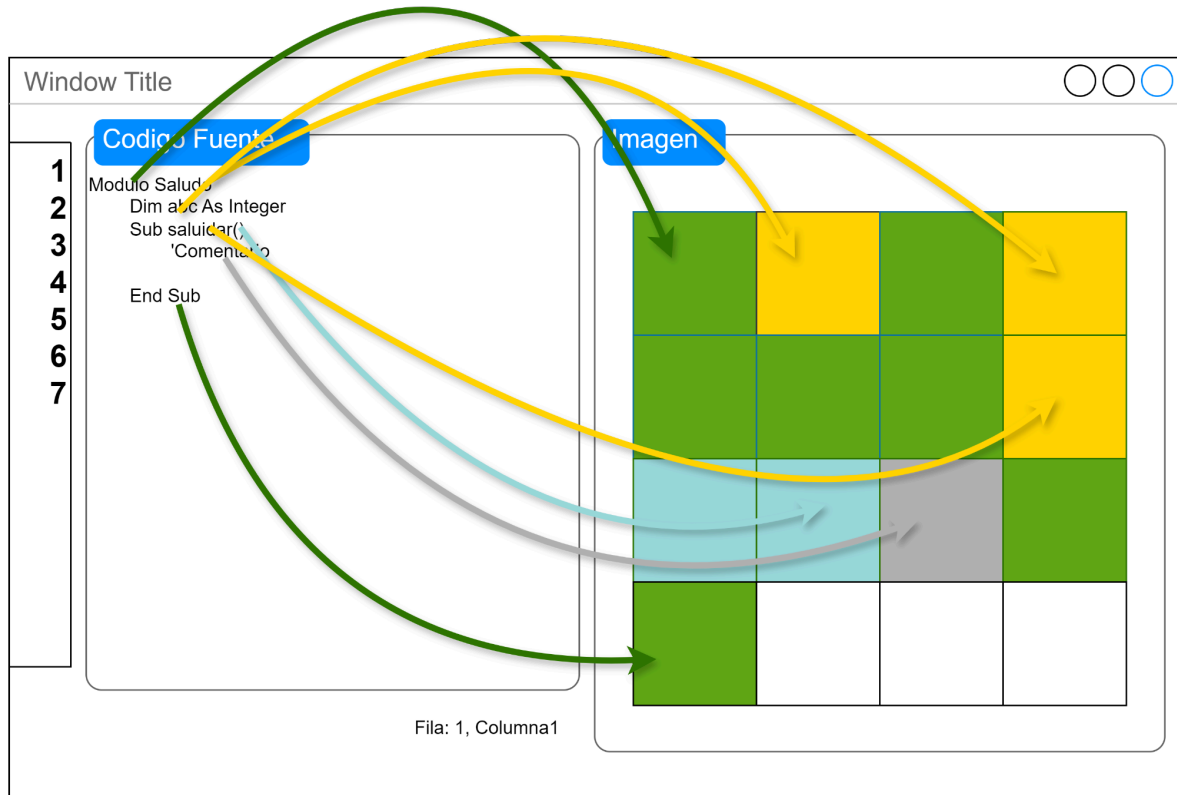
El programa contará con una sección de código fuente donde se ingresa el código de entrada.

También contará con una sección que es donde se generará la imagen, este se irá pintando según el orden con el que se va identificando cada token, y cada token tiene un color correspondiente, por que que el cuadro que le corresponde este se pintara de su color.

Al inicio se debe definir el tamaño de la cuadrícula, cuantas filas y columnas tendrá la imagen pixelada, por lo tanto el número de tokens debe coincidir o ser menor que el número de cuadros que tendrá la imagen.

Ejemplos:







Editor

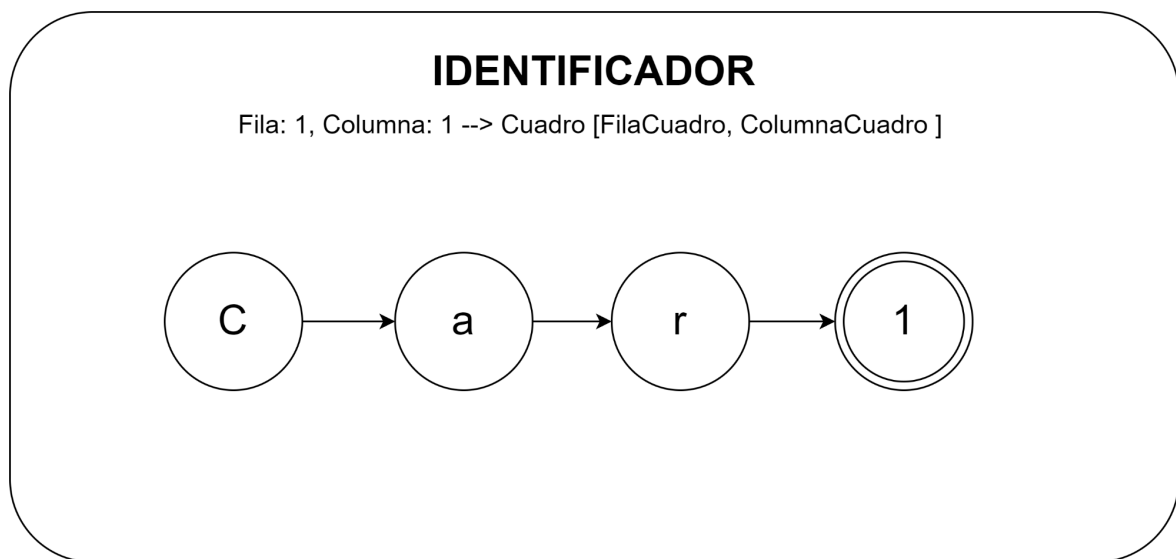
La interfaz de usuario debe incluir un editor de texto que muestre la línea y la columna donde se encuentra el puntero. Este editor debe permitir cargar un archivo de texto o escribir directamente en su área de trabajo

Imagen

Adicional cada cuadro al hacerle clic deberá mostrar en una ventana con la información del token que tiene asignado, la información que debe mostrar es la siguiente:

- Cada token deberá guardar la línea y columna de su ubicación
- Deberá graficar para el token seleccionado un autómata, este autómata corresponde al token, para esto se recomienda usar Graphviz, es decir, debe de graficar el lexema del token seleccionado.
- Debe especificar el tipo de token, la fila y columna del token en el editor de texto y la fila y columna que le corresponde en la cuadrícula o lienzo.

Ejemplo:



También, debe contar con una opción para generar una imagen png o jpg, del lienzo (de la cuadrícula) para poder guardarla.



Reporte

Se deberá de listar los tokens identificados, el reporte debe incluir Token, Lexema, línea, y columna, ejemplo:

Token	Lexema	Línea	Columna	Cuadro
Palabra_Reservada	Module	1	1	Fila:1 Col: 1 Color: ■ #60A917
Identificador	Modulo	1	2	Fila:1 Col: 2 Color: ■ #FFD300
Palabra_Reservada	Dim	2	2	Fila:1 Col: 3 Color: ■ #60A917
Identificador	variable1	2	3	Fila:1 Col: 4 Color: ■ #FFD300



Importante

- Usar lenguaje de programación JAVA
- Práctica obligatoria para tener derecho al siguiente proyecto
- Es válido utilizar algún IDE o cualquier editor de texto
- Las copias obtendrán nota de cero y se notificarán a coordinación
- **No es válido usar regex o algún otro parecido.**
- **No es permitido usar expresiones regulares en el lenguaje de programación.**
- No es válido copy/paste desde internet, si va a usar algo de internet debe entender el código y poder explicarlo.
- Para la imagen del autómata se recomienda usar Graphviz

Entrega

La fecha de entrega es el día 30 de Agosto del 2024 antes de las 23:59 horas por medio de la plataforma Classroom. Los entregables son:

- Código Fuente
- Código compilado (Ejecutable), en la rama principal deberá tener una carpeta con el ejecutable
- Enlace de repositorio
- Manual Técnico, incluyendo automatas de al menos 50% de subprogramas importantes.
- Manual de Usuario.

Calificación

La calificación se realizara el día 30 de Agosto del 2024. El estudiantes debe llevar impresa la hoja de calificación que se publicará en los siguientes días.