MANUAL DE USUARIO

PROYECTO #1

Organización de Lenguajes y de Compiladores 1

Javier Alejandro Gutierrez de León

202004765

Guatemala, 6 de marzo de 2022

Objetivos del sistema

Objetivos Generales

✔ Aplicar los conocimientos sobre la fase de análisis léxico y sintáctico de un compilador para construcción de una solución de software que permita generar análisis por medio del método del árbol.

Objetivos Específicos

✔ Reforzar el concepto del método de Árbol de expresiones regulares en Autómatas Finitos Deterministas (AFD).

✔ Reforzar el concepto del método de Thompson de expresiones regulares en Autómatas Finitos No Deterministas (AFND).

✔ Identificar y programar el proceso de reconocimiento de lexemas mediante el uso de Autómatas Finitos Determinista.

Información del Sistema

Se solicita crear un sistema que pueda realizar el metodo del arbol y el metodo de Thompson, con el fin de que los estudiantes puedan verificar sus resultados en hojas de trabajo y examenes cortos, siendo estos ingresados a travez de un archivo de extension ***".exp"*** utilizando un lenguaje en el cual se definan los conjuntos, las expresiones regulares en notacion polaca o prefija y con esto lograr analizar si son validas cadenas que se ingresen en la segunda seccion del archivo de entrada, generando reportes de cada paso realizado para la elaboracion de los metodos por el que se analizan las expresiones regulares, tambien generando un archivo de salida en formato JSON para mostrar si las cadenas son validas para la expresion regular a la que se asigno y ademas se genera un reporte de errores para notificar acerca de errores lexicos y sintacticos que puedan ser detectados a la hora de la ejecucion.

Especificación Técnica

Requisitos de Hardware

- RAM: 128 MB
- Procesador: Mínimo Pentium 2 a 266 MHz
- Espacio en disco: 124 MB para JRE; 2 MB para Java Update; 45MB para programa fuente

Requisitos de software

- Sistema operativo
- Windows 10 (8u51 y superiores)
- Windows 8.x (escritorio)
- Windows 7 SP1
- Windows Vista SP2
- Windows Server 2008 R2 SP1 (64 bits)
- Windows Server 2012 y 2012 R2 (64 bits)
- Oracle Linux 5.5+1
- Oracle Linux 6.x (32 bits), 6.x (64 bits)2
- Oracle Linux 7.x (64 bits)2 (8u20 y superiores)
- Red Hat Enterprise Linux 5.5+1 6.x (32 bits), 6.x (64 bits)2
- Red Hat Enterprise Linux 7.x (64 bits)2 (8u20 y superiores)
- Suse Linux Enterprise Server 10 SP2+, 11.x
- Suse Linux Enterprise Server 12.x (64 bits)2 (8u31 y superiores)
- Ubuntu Linux 12.04 LTS, 13.x
- Ubuntu Linux 14.x (8u25 y superiores)
- Ubuntu Linux 15.04 (8u45 y superiores)
- Ubuntu Linux 15.10 (8u65 y superiores)
- Mac con Intel que ejecuta Mac OS X 10.8.3+, 10.9+
- Exploradores (Para visualizar reporte de errres):
- Google Chrome (en todas las plataformas)
- Mozilla Firefox (en todas las plataformas)
- Internet Explorer (Windows)
- Microsoft Edge (Windows, Android, iOS, Windows 10 móvil)
- Safari (Mac, iOS)
- Opera (Mac, Windows)

Lenguaje de Programación e IDE

Para el desarrollo de este software se utilizó Netbeans 8.2

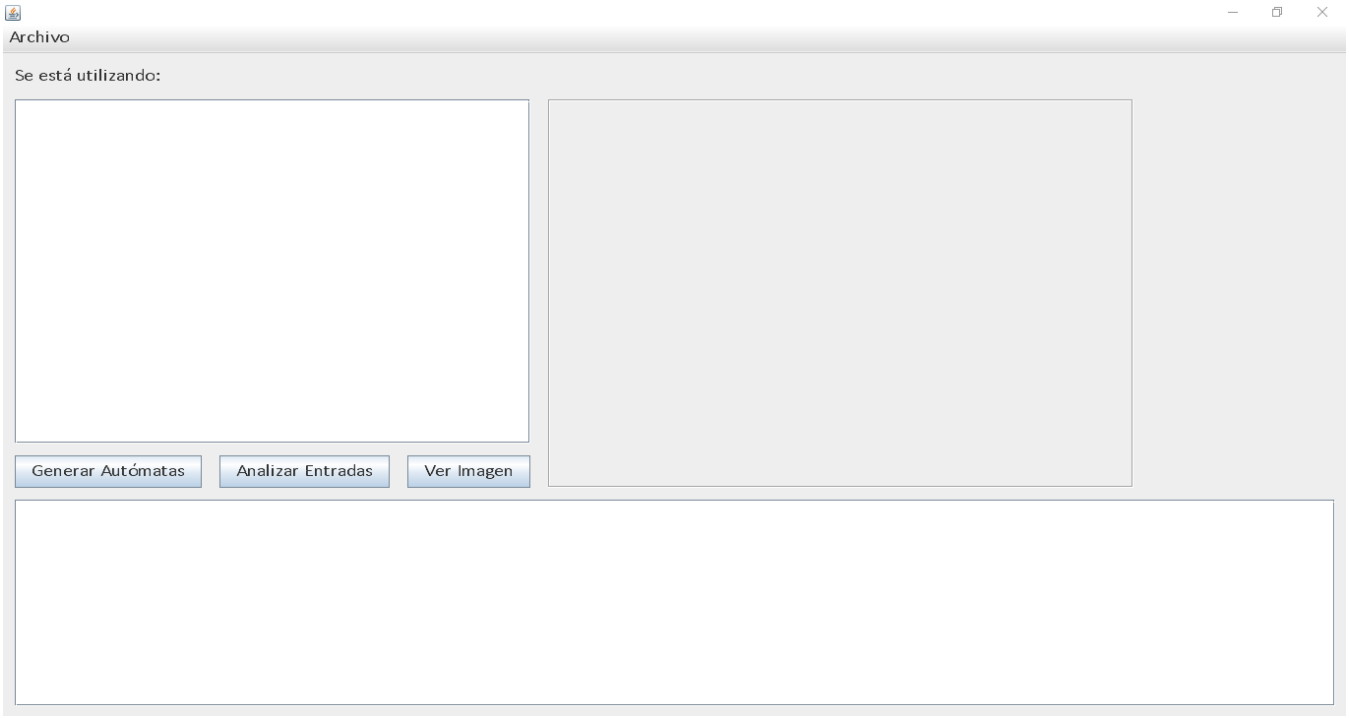
Tecnologías utilizadas

- **Java:** Es un lenguaje de programación de alto nivel , basado en clases y orientado a objetos que está diseñado para tener la menor cantidad posible de dependencias de implementación.
- **HTML5:** lenguaje de marcado para la elaboración de páginas web, con el que se realizaron los reportes de errores lexicos y sintacticos.
- **Boostrap:** Librería para el diseño del reporte HTML.
- **Graphviz:** Es un conjunto de herramientas de software para el diseño de diagramas definido en el lenguaje descriptivo DOT.

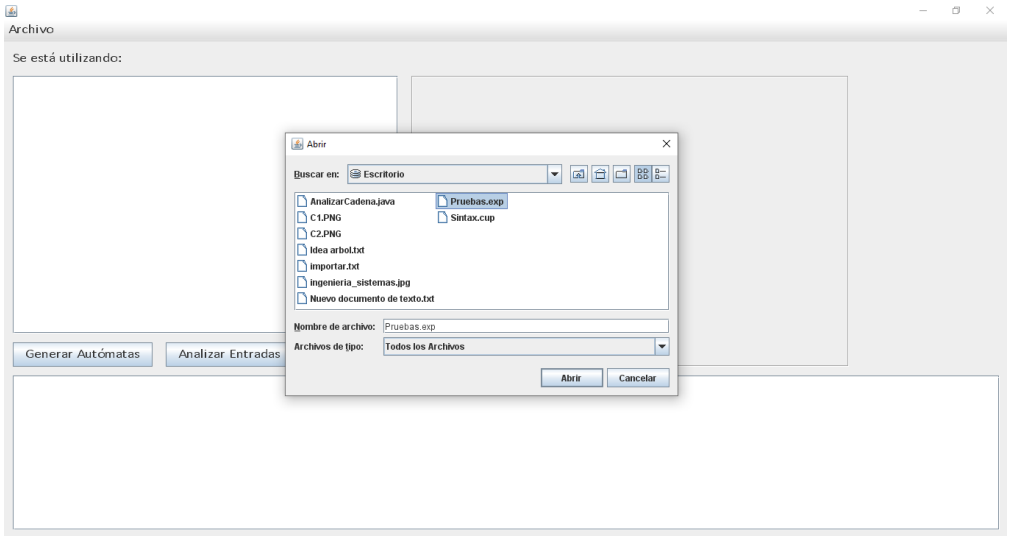
Interfaz Gráfica

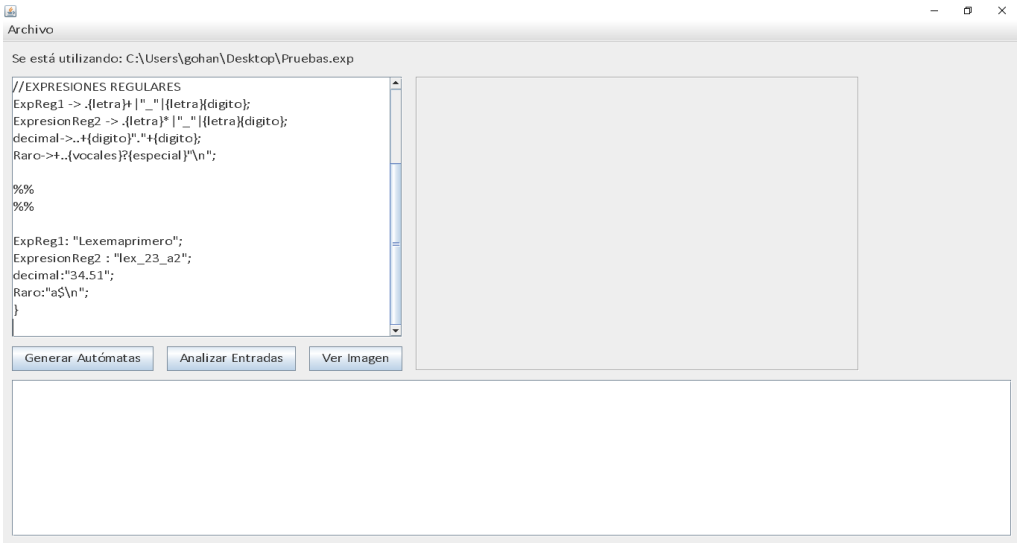
Para comenzar a ejecutar el programa debemos buscar el ejecutable JAR del programa

Al ejecutarse se muestra la siguiente interfaz:



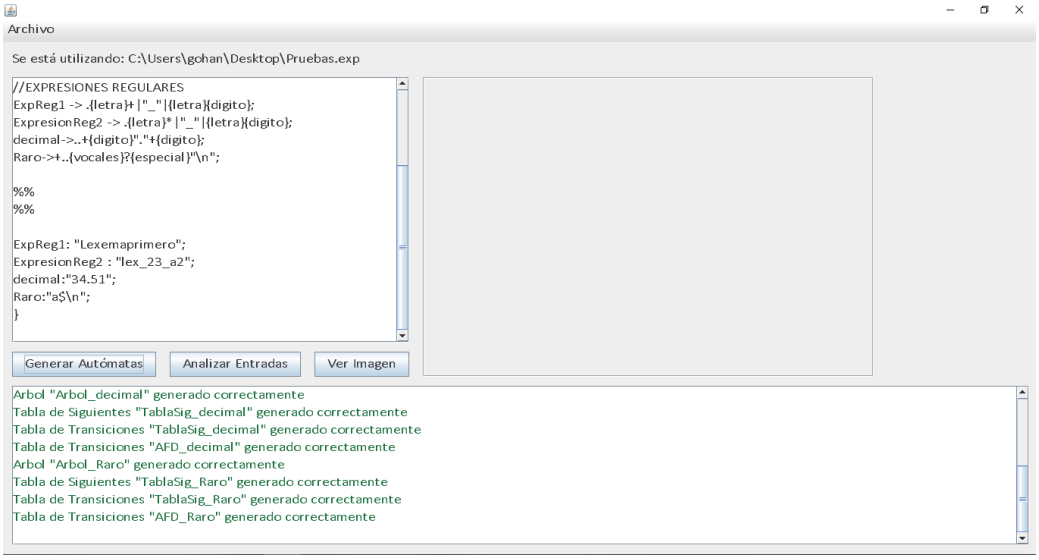
Donde para comenzar a utilizar el programa debemos escribir en el espacio en blanco el lenguaje definido o ya sea cargar un archivo con extension “.exp”.



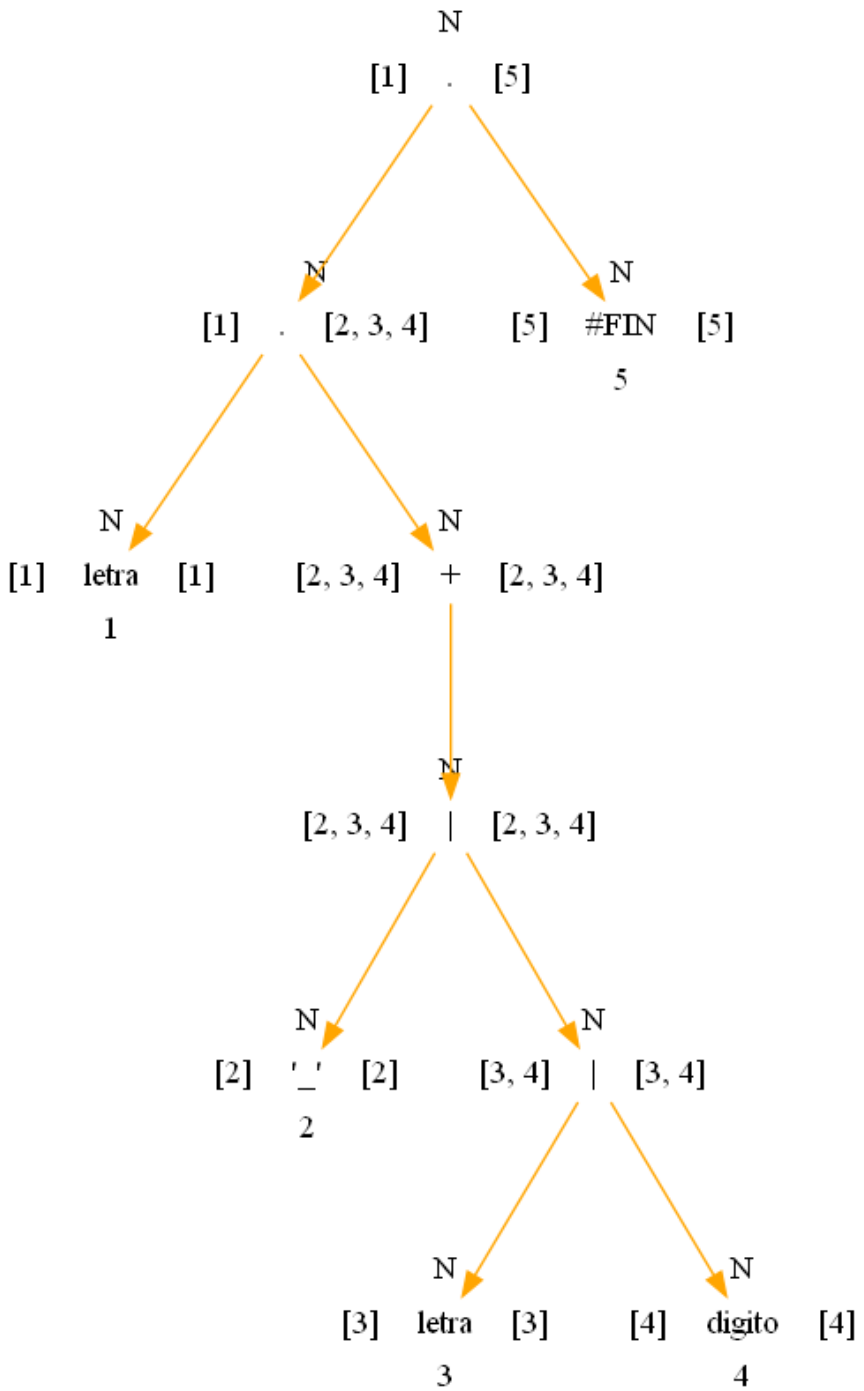


Ademas que si se desea se pueden guardar los cambios que se realicen en el archivo o guardar el que creamos desde 0.

Luego de tener el lenguaje en el espacio en blanco procedemos a analizarlo y generar los automatas con el boton “Generar Automatas”, mostrando mensajes depende el resultado:



En este caso que todo esta bien se generan las imágenes utilizando Graphviz. Para el arbol se genera la imagen y se guarda en la carpeta “ARBOLES_202004765”:



ExpReg1: (({letra}.(('_'|({letra}|{digito}))+)).{#FIN})

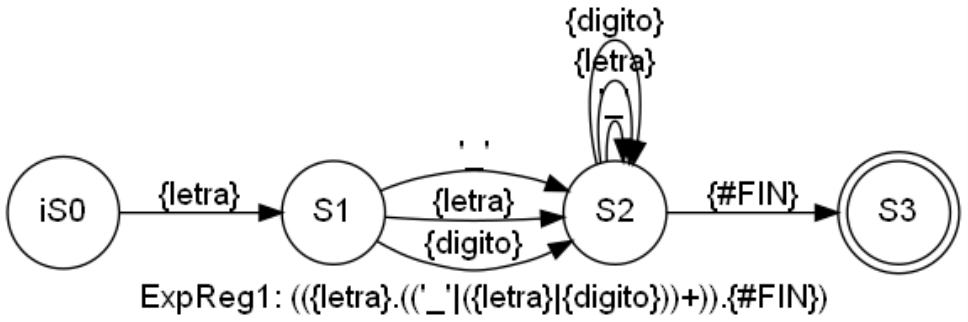
Para la tabla de siguientes se genera la imagen y se guarda en la carpeta “SIGUIENTES_202004765”:

ExpReg1: (({letra}.(("_"|({letra}|{digito}))+)).{#FIN})		
Hoja		Siguientes
{letra}	1	2,3,4
"_"	2	2,3,4,5
{letra}	3	2,3,4,5
{digito}	4	2,3,4,5
{#FIN}	5	-

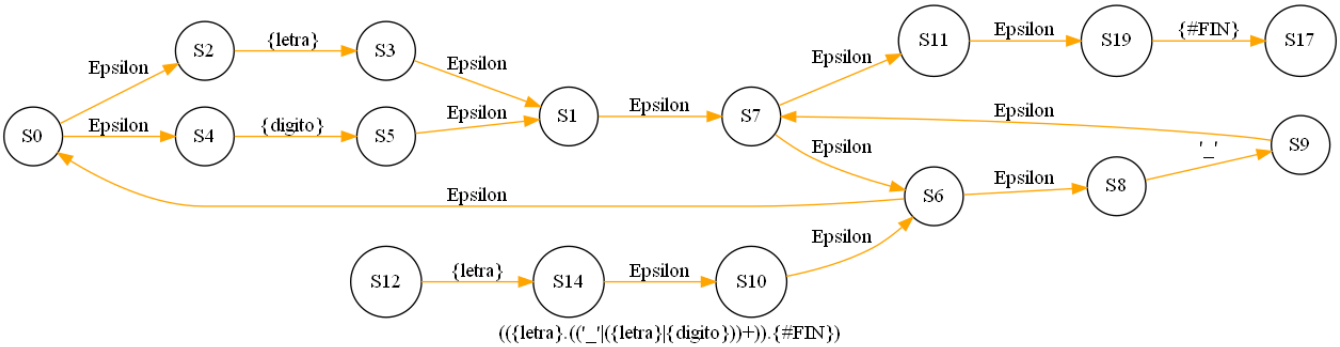
Para la tabla de transiciones se genera la imagen y se guarda en la carpeta “TRANSICIONES_202004765”, en la cual se indica el estado final con un asterisco (*) antes del nombre del estado:

ExpReg1: (({letra}.(("_"|({letra}|{digito}))+)).{#FIN})				
Estado	Terminales			
	{letra}	"_"	{digito}	{#FIN}
S0	S1	-	-	-
S1	S2	S2	S2	-
S2	S2	S2	S2	S3
*S3	-	-	-	-

Para la tabla de transiciones se genera la imagen y se guarda en la carpeta “AFD_202004765”, en la cual se indica el estado inicial con una “i” antes del nombre del estado y con doble circulo en el estado final:



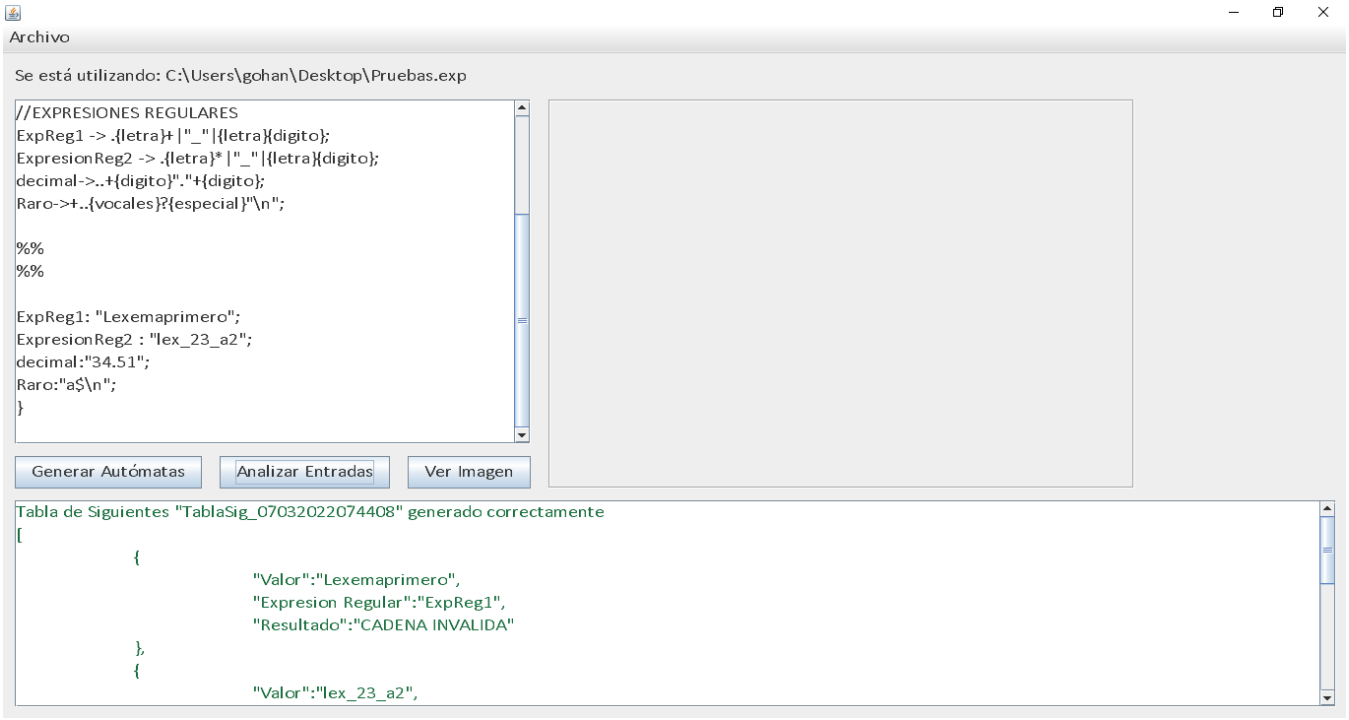
Para la tabla de transiciones se genera la imagen y se guarda en la carpeta “AFND_202004765”:



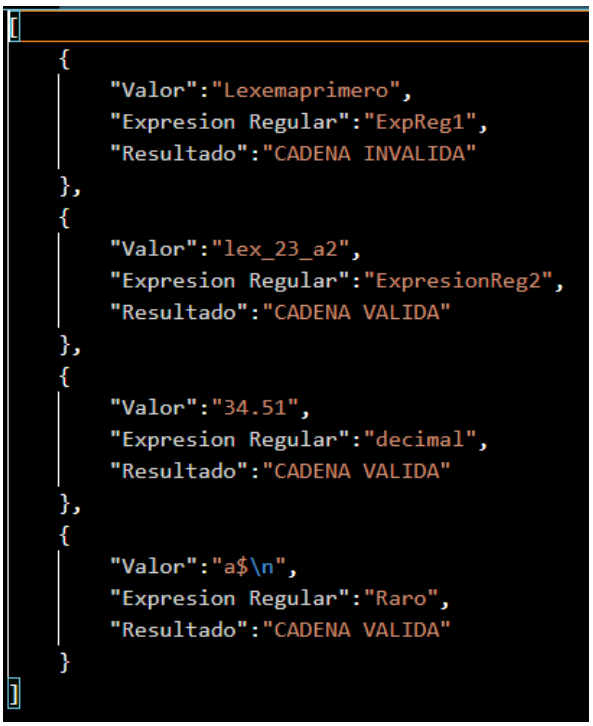
Y en el caso de que se encuentre un error en el lenguaje, se genera un reporte en html en donde se muestra una tabla como la siguiente:

#	Tipo de error	Descripcion	Linea	Columna
1	R Sintactico	Error en "a"	9	162
2	R Sintactico	Error en "-"	17	295

Y para continuar con el proceso se debe analizar las cadenas que se especifican el la segunda parte del lenguaje y se genera un archivo JSON con el resultado de este analisis.



Y al verlo desde el archivo



Y la ultima opcion que se tiene en la interfaz es para poder visualizar las imágenes desde el programa.