**Universidad de San Carlos de Guatemala**

Facultad de Ingeniería

Organización De Lenguajes Y Compiladores 1

Cat. Ing. Bautista

***Manual de Usuario***

***Proyecto #1***

Jeffry Emanuel Méndez Diaz

Carnet: 20191557

***REGEXIVE***

# DESCRIPCIÓN

El programa Regexive se trata de un software de desarrollado con el fin de brindar herramientas para trabajar de forma efectiva con gramáticas y expresiones regulares en notación polaca o prefija. La aplicación tiene la capacidad de reconocer un archivo de entrada, donde se ingresarán comentarios, conjuntos validos de caracteres, expresiones regulares y evaluaciones de las mismas. En el caso de error se crearán reportes y se harán salidas en consola. Por cada expresión regular que el usuario defina se generan el árbol de la expresión, la tabla siguiente y de transiciones. Y para finalizar se crea un AFD para poder valuar las entradas. Se cuenta con un visualizador de los grafos generados para una mayor utilidad.

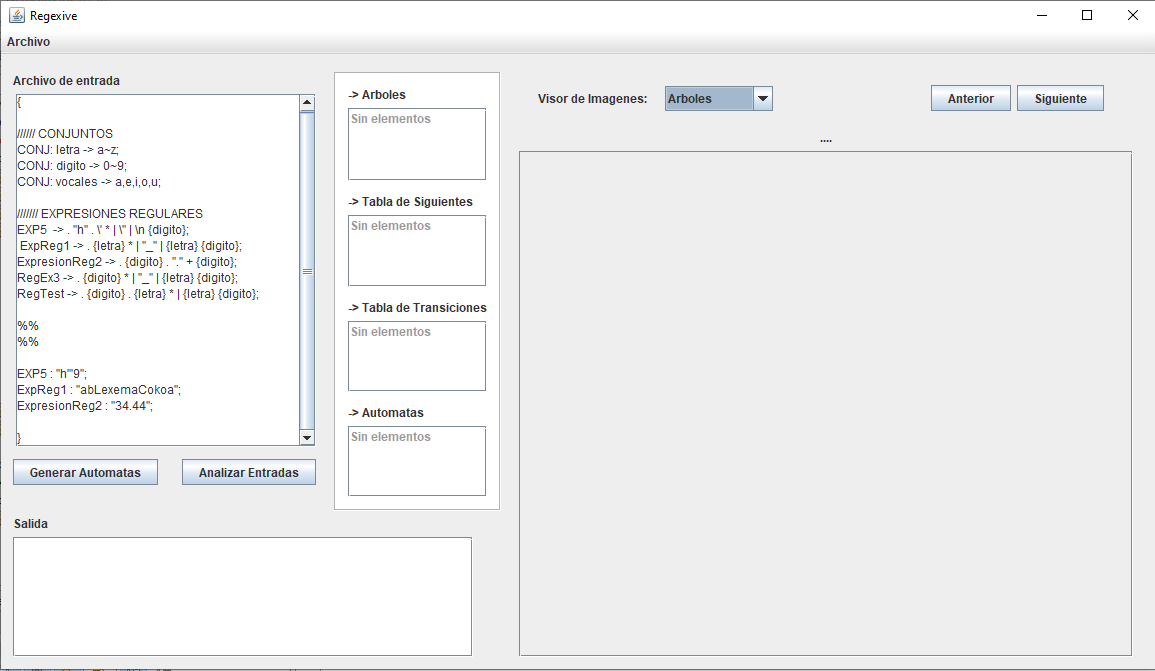
# REQUISITOS SISTEMA

|  |  |
| --- | --- |
| Java | Versión 1.8.0\_281 |
| Java (TM) SE Runtime Environment | Build 1.8.0\_281-b09 |
| Java HotSpot(TM) 64-Bit Server VM | Build 25.281-b09 |
| Sistema Operativo | Windows 10 |

**FUNCIONAMIENTO PROGRAMA**

# PANTALLA INICIAL

Al iniciar el programa se mostrará la siguiente pantalla. Considerar que, en el menú de archivos únicamente se aceptaran los que tengan extensión (.OLC).



# NUEVO ARCHIVO

# Nuevo archivo: Se crea un archivo con numero aleatorio.

Abrir archivo: Permite buscar el archivo en tu directorio.

Guardar: Se guarda el ultimo cambio del archivo actual.

Guardar como: Se guarda el archivo con nombre diferente.

# 

# GRAMATICA

Al iniciar el programa se trae por defecto una gramática que sirve de ejemplo para elaborar sus propios archivos.

En resumen, la notación es la siguiente:

// -> Comentario

<! -- !> -> Comentario multilínea

CONJ -> Palabra reservada para conjuntos

%%%% -> Divisor de archivo

ID\_expr -> Expresion\_en\_prefijo

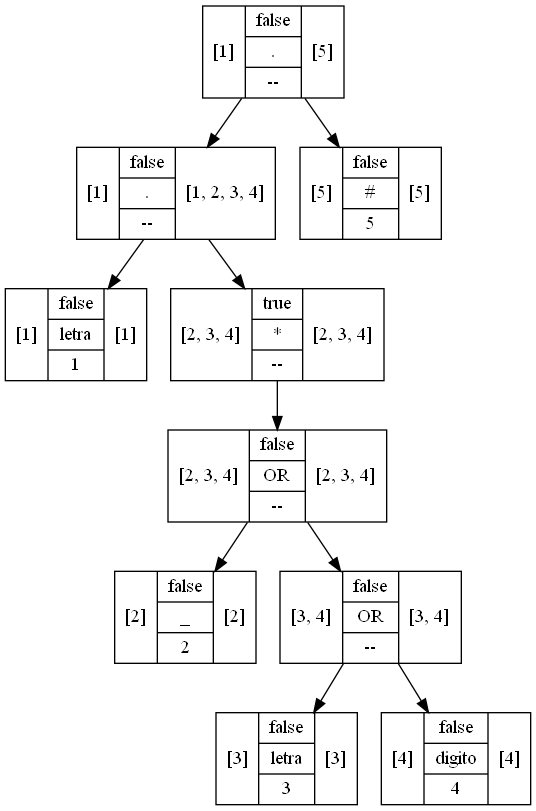
ID\_expr -> Valor\_entrada

**BOTON GENERAR AUTOMATAS**

Si el archivo de entrada no contiene errores irrecuperables, se analizarán todos los elementos añadidos y se generarán los siguientes grafos:

* Árbol binario
* Autómata finito determinista
* Tabla de siguientes
* Tabla de transiciones

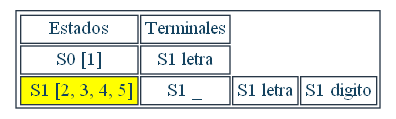
Ejemplo de árbol:



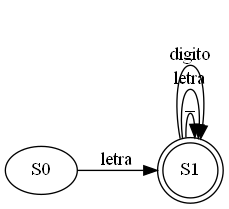
Ejemplo de tabla de siguientes:



Ejemplo de tabla de transiciones:



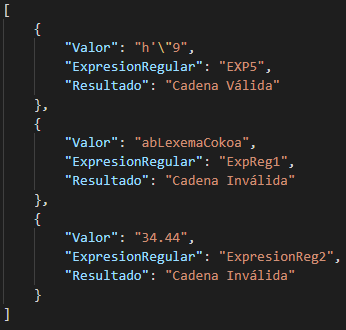
Ejemplo de autómata:



**BOTON ANALIZAR ENTRADAS**

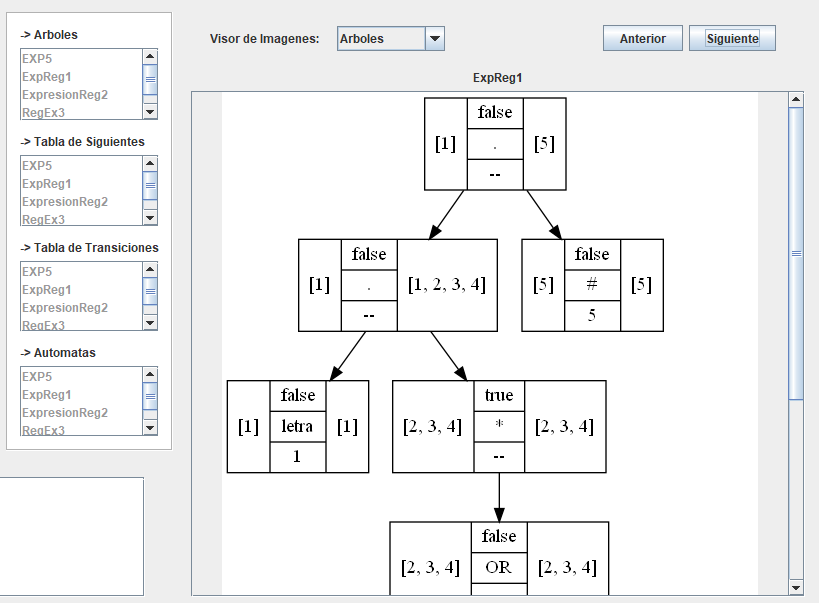
Si ya se generaron los autómatas, se creará un archivo JSON con los resultados de todas las evaluaciones que se añadan.

Ejemplo:



**VISUALIZAR IMÁGENES**

Si ya se generaron todos los grafos, se agregarán al listado de elementos para poder visualizar mejor que elementos fueron creados. Con el seleccionador se puede elegir que carpeta de grafos se desean ver. Y con los botones de anterior y siguiente se puede navegar entre expresiones.



**ERRORES**

Si el archivo de entrada contiene errores léxicos o sintácticos, se mostrarán en la consola y se generara su reporte correspondiente.

