

Manual de Usuario

COMPI1 SECCIÓN B

Diego Alejandro Sierra García

201903969

Introducción



Se desarrolla un programa con múltiples funcionalidades en el Lenguaje de programación Java para procesar expresiones regulares, construir árboles y optimizarlos a un Autómata Finito Determinista. Además podrá evaluar cadenas en base a las expresiones regulares que habían ingresado.

Índice

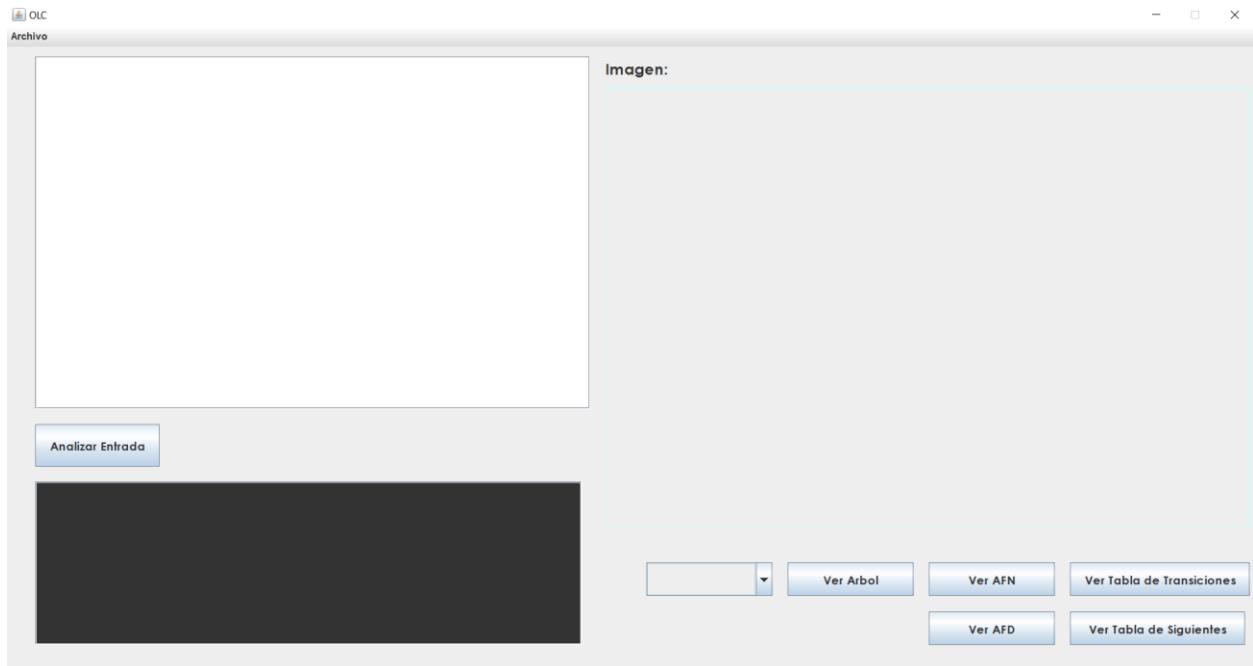
1. Iniciar el programa
2. Funciones básicas
3. Solución de problemas
4. Archivos de salida

Iniciar el programa

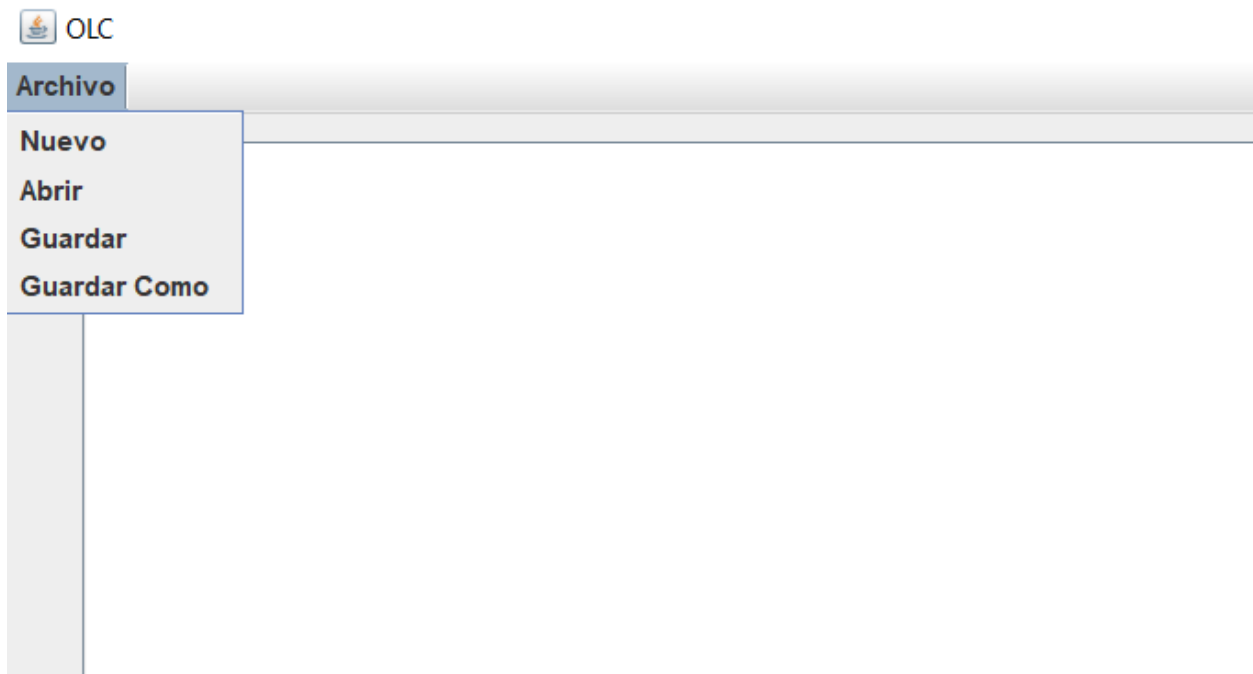
Para iniciar el programa, se debe dar doble click sobre el archivo Compi1-Lenguaje-OLC.jar y tener instalado Java. Debe estar presente la carpeta llamada lib, pero no se debe modificar.

Name	Date modified	Type	Size
 lib	3/6/2021 7:19 PM	File folder	
 Compi1-Lenguaje-OLC.jar	3/6/2021 7:02 PM	Executable Jar File	83

Funciones principales



Este será la vista del programa cuando lo habrá. El programa tiene un cuadro de entrada a la izquierda arriba. Con un cuadro de salida en la izquierda abajo. En la derecha se muestran las imágenes y en la parte de abajo muestran el nombre de las expresiones regulares analizadas.



Se puede seleccionar 4 opciones. La opción nueva abrirá una nueva ventaba limpia. El archivo abrir muestra un cuadro para poder seleccionar un archivo con extensión .olc para poder colocarlo en el cuadro de texto y modificarlo. La opción de guardar, guarda las modificaciones del cuadro de texto en el archivo ya abierto. La opción de guardar como crea un nuevo archivo .olc.



OLC

Archivo

```
{  
  
CONJ: mayus - > A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X,Y,Z;  
CONJ: minus - > a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z;  
CONJ: letra - > a~z;  
CONJ: digito - > 0~9;  
CONJ: posibles - > P, C, O;  
  
PI - > . {posibles} . ? "-" . {digito} . {digito} . {digito} . {letra} . {letra} {letra};  
  
%%  
%%  
  
P1 : "P38df8c"; // malo  
P1 : "P-38df8k"; // malo  
P1 : "K-385ffk"; // malo  
}
```

Analizar Entrada

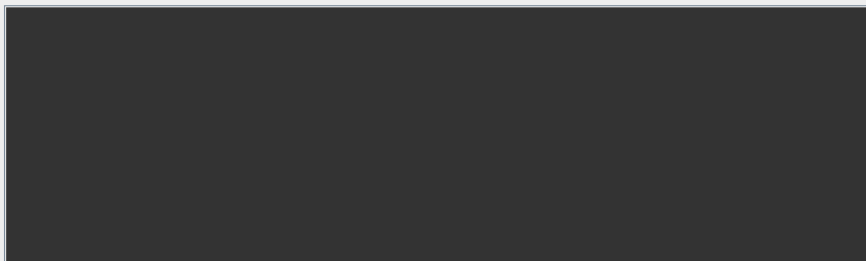
El archivo maneja esta estructura. Comienza con una llave { y termina con }. A la mitad del cuerpo se encuentra un separador %%%%. Previo a este separador se pueden colocar conjuntos o expresiones con un nombre. Después de los separadores se colocan las cadenas que se desean evaluar y con que el nombre de la expresión que la evalúa.

```
{
CONJ: mayus -> A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X,Y,Z;
CONJ: minus -> a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z;
CONJ: letra -> a-z;
CONJ: digito -> 0-9;
CONJ: posibles -> P, C, O;

EXP5 -> .\' . + ||| \n {minus} {mayus} {digito} " " \';
PI -> . {posibles} . ? "-" . {digito} . {digito} . {digito} . {letra} . {letra} {letra};
%%
%%
EXP5 : "\cadena entre comilla simple\''";
P1 : "P-385fde"; // bueno
P1 : "P388fdc"; // bueno
P1 : "P38df8c"; // malo
P1 : "P-38df8k"; // malo
P1 : "K-385ffk"; // malo
}
```

Imagen:

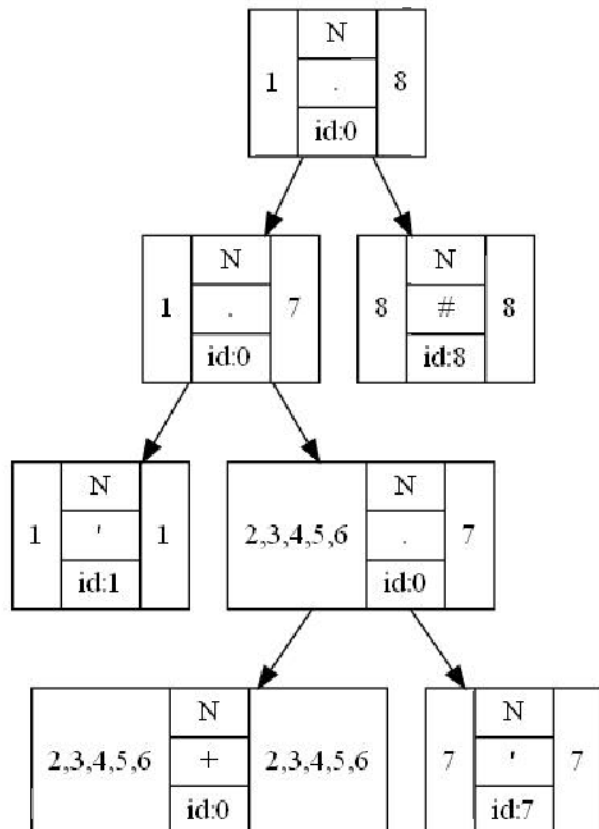
Analizar Entrada



EXP5	▼
EXP5	
PI	

Cuando el archivo esta listo y se respeta la estructura, se le da Analizar entrada. Si no se encuentra ningún error, el cuadro de salida estará vacío y el combobox se llenara con los nombres de la expresiones regulares que se ingresaron.

Arbol: EXP5



EXP5

Ver Arbol

Ver AFN

Ver Tabla de Transiciones

Ver AFD

Ver Tabla de Siguietes

En el lado izquierdo, se mostraran las opciones de análisis de la expresión regular. Se debe seleccionar el nombre y apachar el botón de la imagen que se desea ver. En la parte de arriba se mostrara el nombre del modo de evaluar y la expresión regular.

Manejo de errores

```
{
CONJ: mayus - > A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X,Y,Z;
CONJ: minus - > a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z;
CONJ: letra - > a~z;
CONJ: digito - > 0~9;
CONJ: posibles - > P, C, O☒;










EXP5 - > . \' . + ||| | \n {minus} {mayus} {digito} " " \';
PI - > . {posibles} . ? "-" . {digito} . {digito} . {digito} . {letra} {letra} {letra} ►;
%%
%%
EXP5 : "'cadena entre comilla simple\'';
P☒1 : "P-385fde"; // bueno
P1 : "P388fdc"; // bueno
P1 : "P38df8c"; // malo
P1 : "P-38df8k"; // malo
P1 : "K-385ffk"; // malo
}
```

Analizar Entrada

Error de tipo Lexico en la Linea 8 y columna 1. Componente ☒ no reconocido.
 Error de tipo Tipo Sintactico (Recuperado) en la Linea 89 y columna 11. Componente { no reconocido.
 Error de tipo Lexico en la Linea 11 y columna 1. Componente ► no reconocido.
 Error de tipo Tipo Sintactico (Recuperado) en la Linea 3 y columna 15. Componente P no reconocido.
 Error de tipo Lexico en la Linea 15 y columna 1. Componente ☒ no reconocido.

Si la entrada contenia errores, estos se mostraran en la salida de abajo. Y no se generaran los diferentes reportes de las expresiones regulares.

Archivos de salida

	AFD_201903969	3/7/2021 1:12 PM	File folder	
	AFND_201903969	3/7/2021 1:12 PM	File folder	
	ARBOLES_201903969	3/7/2021 1:12 PM	File folder	
	ERRORES_201903969	3/7/2021 1:12 PM	File folder	
	lib	3/7/2021 12:52 PM	File folder	
	SALIDAS_201903969	3/7/2021 12:52 PM	File folder	
	SIGUIENTES_201903969	3/7/2021 1:12 PM	File folder	
	TRANSICIONES_201903969	3/7/2021 1:12 PM	File folder	
	Compi1-Lenguaje-OLC.jar	3/7/2021 12:52 PM	Executable Jar File	89

Todas las imágenes de análisis se guardarán en las distintas carpetas que se crearan en donde estaba el archivo de ejecución. Los errores también serán guardados en este apartado.