

## **Estudiantes:**

- **Anderson Saldarriaga castaño**

## **Descripción de proyecto**

El trabajo consistió en construir un programa de computador que permitiera el ingreso de un expresión regular y generar un AFD en base a la expresión entrada.

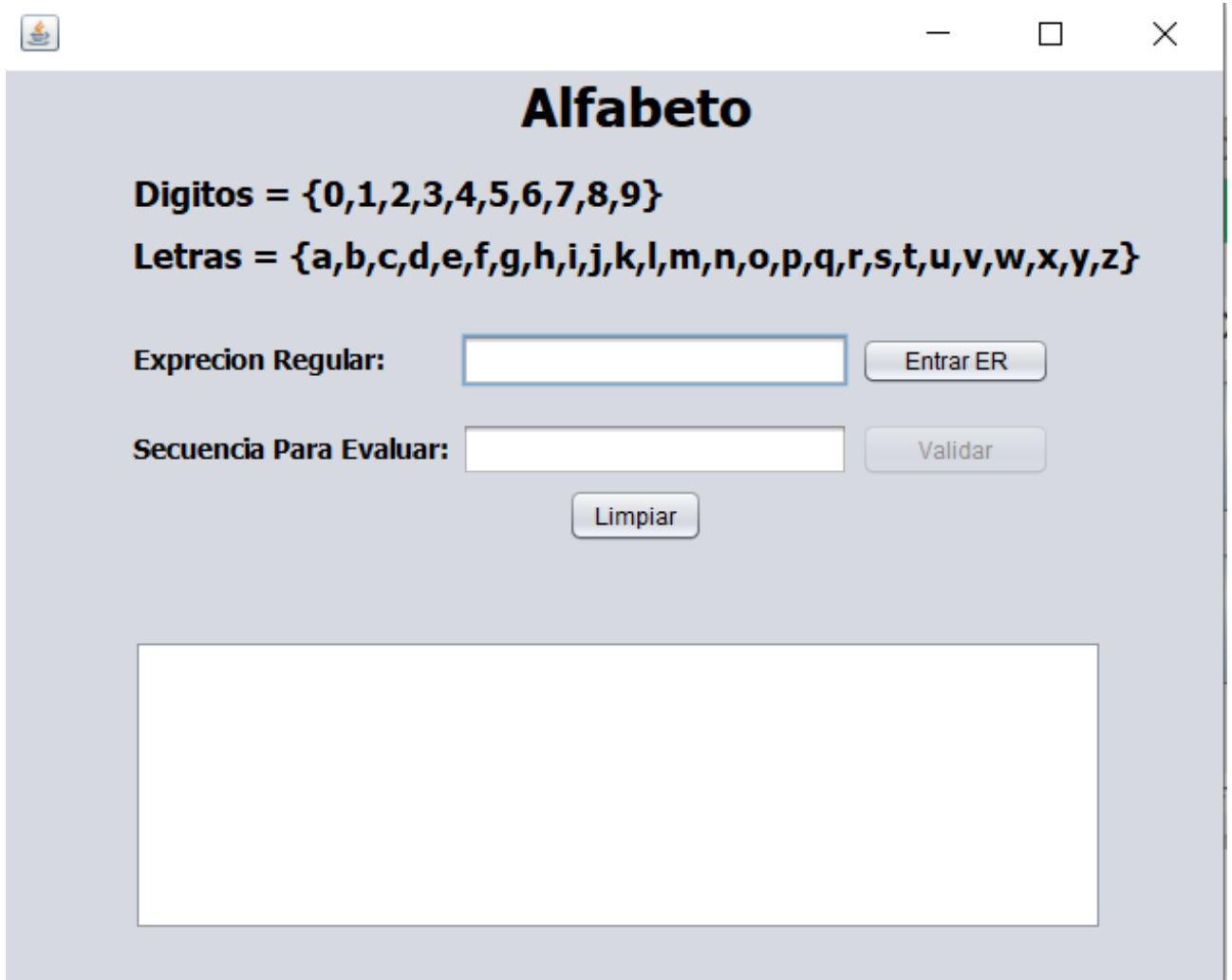
El laboratorio fue realizado en el lenguaje java(Netbeans IDE 8.2), Este posee una interfaz de escritorio realizada con librerías swing.

El proyecto "Practica1TLenguaje" se encuentra dividido en 5 paquetes:

- Estructuras: Contiene la estructura de los datos utilizados en el sistema.
- Análisis: En ella se encuentra la construcción del alfabeto, un analizador de léxico, los tokens utilizados y la expresión regular.
- Algoritmos: Aquí se encuentran los algoritmos de Thompson, la validación entre otros, que son necesarios para la creación de los autómatas.
- Main: Aquí está la clase principal, fue utilizada para las pruebas del proyecto.
- Interfaz: Contiene la interfaz GUI necesaria para la interacción del software con el usuario.

## Interfaz Gráfica

El software solo cuenta con una ventana, en esta se muestra el alfabeto que se usará. Básicamente el programa soporta todos los números del 0 al 9 y el abecedario con excepción de la "ñ". Se añadió el alfabeto válido en la ventana, para que el usuario tenga en cuenta que símbolos puede usar.



The image shows a graphical user interface window titled "Alfabeto". The window has a standard title bar with a minimize button, a maximize button, and a close button. The main content area has a light gray background. At the top, the title "Alfabeto" is displayed in a large, bold, black font. Below the title, there are two lines of text: "Digitos = {0,1,2,3,4,5,6,7,8,9}" and "Letras = {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}". Below these, there are two input fields. The first is labeled "Expresion Regular:" and has a button "Entrar ER" to its right. The second is labeled "Secuencia Para Evaluar:" and has a button "Validar" to its right. Below the "Validar" button is a button labeled "Limpiar". At the bottom of the window is a large, empty rectangular box.


El usuario tendrá que ingresar una expresión regular con el alfabeto dado en el cajón de texto al frente de “Expresión Regular:”

Luego de ingresar deberá dar clic en el botón “Entrar ER”. Esto hará que el Software cree un autómata finito determinista que se imprimirá con sus respectivos estados de aceptación en el panel de texto que se encuentra en la parte inferior de la ventana.

Esta acción habilitará el botón “Validar” que se explicará más adelante.

**Importante:** Las ER que se usa soportan los siguientes operadores:

- Unión ( | )
- Cerradura positiva ( + )
- Cerradura Kleene ( \* )
- Agrupaciones (se usan paréntesis)
- Concatenación (no se necesita carácter) **ejemplo de concatenación: 10**

—□×

## Alfabeto

**Digitos = {0,1,2,3,4,5,6,7,8,9}**

**Letras = {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}**

**Expresion Regular:**

**Secuencia Para Evaluar:**

**Automata Creado**

Automata Finito

(0 1 4)i --> (0 1 4)i{0} --> (2 3)f{1}

(2 3)f --> (2 3)f{0} --> (0 1 4)i{1}

Estados De Aceptacion

[(2 3)f]

Dado el caso en que el usuario ingrese erróneamente la ER el programa hará que aparezca una ventana dando a conocer el problema, ya sea la falta de un paréntesis u otro error de sintaxis.

A Continuación se mostrará un ejemplo de dicha situación. la ER que es **(0|10\*110\*** Como se muestra le hace falta un paréntesis de cierre y hasta que el usuario no corrija dicho problema no se creará el AF.

## Alfabeto

**Digitos = {0,1,2,3,4,5,6,7,8,9}**

**Letras = {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}**

**Expresion Regular:**

**Secuencia Para E**

✕




Error de sintaxis

Caracter:

Mensaje : Falta parentesis de cierre

Ahora con el AF creado podemos evaluar una secuencia. Ahora escribiremos la cadena de texto que queramos evaluar y luego daremos en el botón "Validar". Esto hará que nos aparezca un mensaje si la cadena fue o no fue aceptada por el AF.

—□×

## Alfabeto

**Digitos = {0,1,2,3,4,5,6,7,8,9}**

**Letras = {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}**

**Expresion Regular:**

**Secuencia Para Evaluar:**

**La cadena es ACEPTADA.**

Automata Finito

$(0\ 1\ 4)i \rightarrow (0\ 1\ 4)i\{0\} \rightarrow (2\ 3)f\{1\}$

$(2\ 3)f \rightarrow (2\ 3)f\{0\} \rightarrow (0\ 1\ 4)i\{1\}$

Estados De Aceptacion

$[(2\ 3)f]$

Se podrá seguir validando infinitamente cadenas de texto, pero si el usuario quiere crear otro AF, tendrá que dar en el botón “Limpiar” esto habilita que se pueda escribir otra expresión regular. volviendo al inicio de cómo se ejecutó el programa.



The image shows a Java Swing window titled "Alfabeto". The window has a standard title bar with a minimize button, a maximize button (disabled), and a close button. The main content area has a light gray background. At the top, the title "Alfabeto" is displayed in a large, bold, black font. Below the title, there are two sets of definitions: "Digitos = {0,1,2,3,4,5,6,7,8,9}" and "Letras = {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}", both in bold black font. Below these definitions, there are two input fields. The first is labeled "Expresion Regular:" and has a text input field next to it. To the right of this input field is a button labeled "Entrar ER". The second is labeled "Secuencia Para Evaluar:" and has a text input field next to it. To the right of this input field is a button labeled "Validar". Below these two input fields, there is a button labeled "Limpiar". At the bottom of the window, there is a large, empty rectangular area, likely intended for displaying the results of the validation.

**Alfabeto**

**Digitos = {0,1,2,3,4,5,6,7,8,9}**

**Letras = {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}**

Expresion Regular:

Secuencia Para Evaluar: