Manual Técnico

Clase analizador:

La clase analizador es la que se encarga de poder realizar todo el análisis léxico del archivo de entrada, dentro de esta clase se encuentra los siguientes métodos:

- **public List<Token> Escaner(String entrada**): es el método encargado que recibe como parámetro el texto de entrada, dentro de este método se maneja los estados de transición del autómata y se verifica que todos los caracteres corresponda a un token.
- **public void AgregarToken(Token.TipoToken tipo**): método que se encarga de agregar los lexemas validos a un lista de tokens.
- **public Boolean PalabraReservada(String entrada):** método que se encarga de verificar las palabras reservadas que existen en lenguaje.
- **public List<Canciones> GenerarPlaylist(List<Token> entrada1):** método que se encarga de realizar la búsqueda de las canciones que están definidas dentro del bloque playlist del archivo de entrada. Estas se van agregando a una list de canciones para posteriormente usarlas.
- **public int AgregarCancionesR(List<Token> token,int posicion,int repeticion):** método que se encarga del manejo de todas las canciones que se encuentra dentro del bloque repetición
- **public int CancionesSimples(List<Token> token, int posicion, int repeticion**): método que se encarga del manejo de las canciones que no están dentro de un bloque repetición.

Clase Token:

La clase token es utilizada para el manejo de todos los token definidos dentro del lenguaje posee los siguientes métodos:

- **public String getLexema**(): devuelve el lexema guardado dentro de un objeto token
- **public int getFila():** devuelve el número de fila en que se encuentra un token especifico.
- **public int getColumna():** devulve el número de la columna en que termina un token especifico.
- **public string getTipo():** devuelve el tipo de token al que pertenece un objeto token(cadena, palabra reservada, etc).

Clase canciones:

La clase canción es utilizada para guardar los datos de una canción buscándola dentro de la lista de token. La clase canción posee los siguientes atributos nombre, artista, álbum, año, duración, url, repetición, playlist. Dentro de esta clase se encuentra los siguientes métodos.

Nombre: define el nombre de la canción.
Artista: define el nombre del artista al que pertenece la canción.
Álbum: define el álbum al que pertenece la canción.
Año: define el año en que salió la canción.
Duración: define el tiempo que tarda una canción.
url: define la ruta donde se encuentra el archivo .mp3 de la canción
playlist: define el nombre de la playList a la que pertenece la canción.

Métodos: los siguientes métodos devuelven el atributo de un objeto canción.

- public String getNombre()
- public String getArtista()
- public String getAlbum()
- public String getAño()
- public String getDuracion()
- public String getUrl()
- public int getRepes()
- public String getPlay()

Formulario Form1
Contiene la interfaz gráfica de la aplicación.

menuStrip1 contiene las herramientas de la cinta superior de la interfaz
Explorador OpenFileDialog utilizado para permitirle al usuario poder abrir un archivo desde cualquier dirección
Guardar SaveFileDialog usado para que el usuario pueda guardar un archivo ya existente verificando que sea válido o un archivo nuevo.
Guardar Como SaveFileDialog usado para guardar una archivo ya existen con otro nombre y nueva dirección
Texto RichTextBox que le permite al usuario realizar la edición del texto

Métodos:

- **public void Pintar(String entrada)** método encargado de pintar aquellos token validos dentro del editor de texto, recibe como parámetro la lista de tokens que se genero

Eventos:

- **private void analizarToolStripMenuItem_Click(object sender, EventArgs e)** evento que se encarga de correr el elemento explorador y así mismo extrae el contenido del elemento areaTra para poder analizarlo
- **private void abrirToolStripMenuItem_Click(object sender, EventArgs e)** Evento que se encarga de llamar al elemento explorador

- **private void guardarToolStripMenuItem_Click(object sender, EventArgs e)** Evento que se encarga de llamar al elemento Guardar
- **private void guardarComoToolStripMenuItem_Click(object sender, EventArgs e)** Evento que se encarga de llamar al elemento GuardarComo.
- **private void acercaDeToolStripMenuItem_Click(object sender, EventArgs e)** Evento que abre una venta, donde se muestran los datos del creador de la aplicación.
- **private void salirToolStripMenuItem_Click(object sender, EventArgs e)** Evento que se encarga de finalizar la aplicación.

Formulario Form2:

Formulario en donde se cargan los token validos después de realizar el análisis léxico.

Elementos:
Tablatoken: un elemento de tipo Tabla

Eventos:
- **private void Tokens_Load(object sender, EventArgs e):** evento que se realiza para cargar los tokens dentro de la tabla tablatoken

Formulario TablaErrores:

Formulario donde se cargan todos los errores léxicos encontrados durante el análisis.

Elementos:
TablaError: un elemento de tipo Tabla

Eventos:
- **private void Tokens_Load(object sender, EventArgs e):** evento que se realiza para cargar los errores léxicos dentro de la tabla tablaError.

Formulario Reproductor:

Formulario que se encarga de todo del funcionamiento del reproductor, este formulario tiene los siguientes métodos, eventos, elementos, etc.

Elementos:

- MisCanciones: elemento de tipo ListView, que es utilizado para poder cargar todas las canciones encontradas.
- MiReproductor: elemento que pertenece a la clase Windows Media, que es utilizada para poder reproducir las canciones.

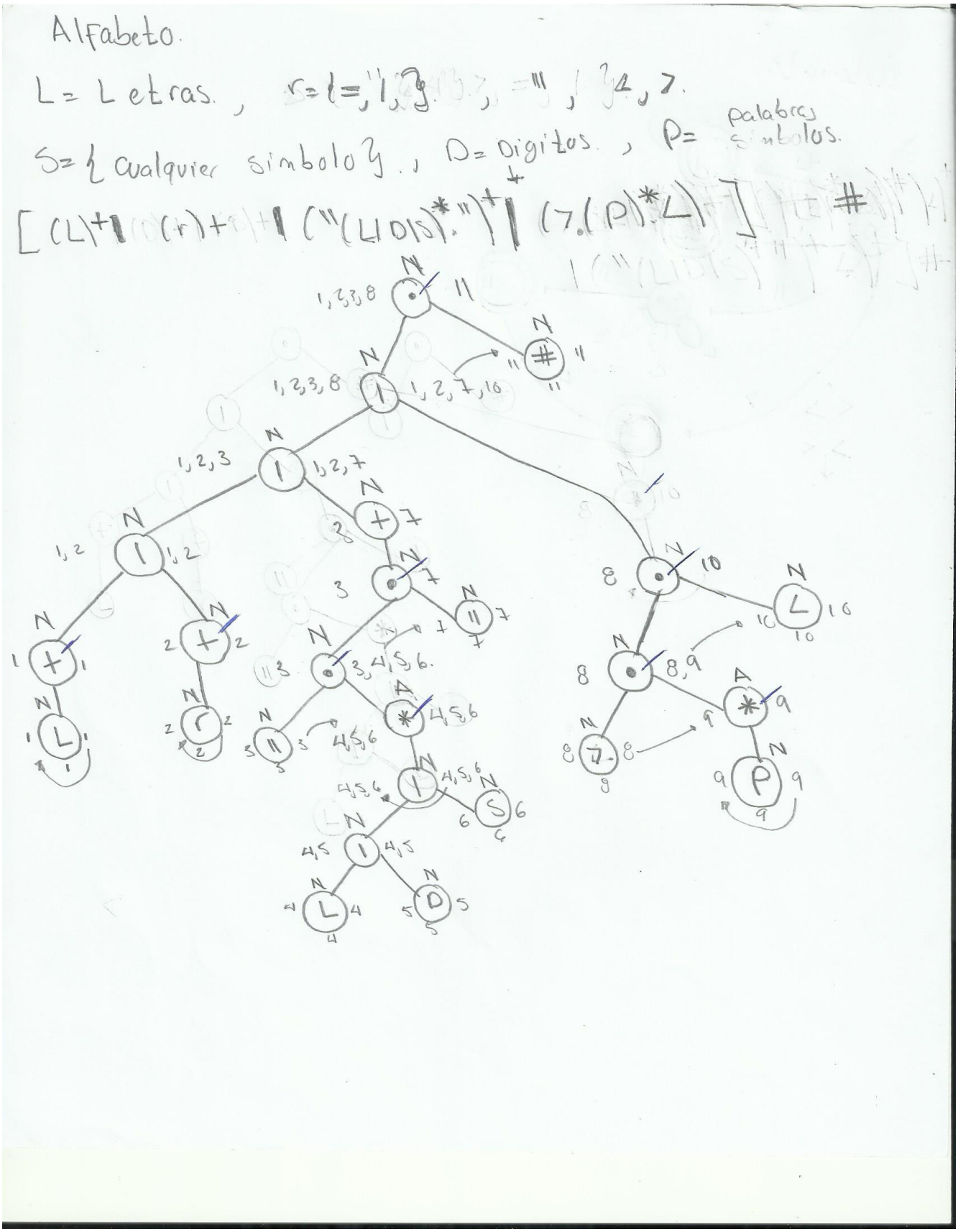
Métodos:

- **public int EncontrarPosicion():** Método que se encarga de encontrar la posicion del elemento del listview que se quiere reproducir, en un caso que no esté seleccionado alguna empezara en la posicon 0.
- **public void Siguiente ():** Método que se encarga de cambiar de canción, cuando la anterior ya ha terminado.

Eventos:

- **private void eliminar_Click(object sender, EventArgs e**): Evento que se encarga de eliminar un elemento del listview, este método se dispara al presionar el botón eliminar.
- **private void MiReproductor_PlayStateChange(object sender, AxWMPLib._WMPOCXEvents_PlayStateChangeEvent e):** Evento que se encarga de verificar el estado del reproductor, en este caso solo se verifica que el medio que se estaba reproduciendo ha llegado a su fin, para ello usamos el valor 8 que nos indica que esta en el estado mencionado anteriormente.
- **private void button1_Click(object sender, EventArgs e**): evento que se encarga de empezar a reproducir las canciones de la playlist.
- **private void Reproductor_Load(object sender, EventArgs e):** evento que se realizar al cargar el formulario, aquí se hace la agregación de las canciones al listview.

CREACION DEL AUTOMATA FINITO DETERMINISTA



NO	Σ	Follow pos
1	L	1, 11
2	r	2, 11
3	11	4, 5, 6, 7
4	L	4, 5, 6, 7
5	D	4, 5, 6, 7
6	S	4, 5, 6, 7
7	11	11
8	>	9, 10
9	P	9, 10
10	<	11
11	#	

Estado	L	r	11	D	S	>	P	<
$S_0 = \{1, 2, 3, 8\}$	S1	S2	S3			S5		S5
*$S_1 = \{4, 11\}$	S1							
*$S_2 = \{2, 11\}$		S2						
$S_3 = \{4, 5, 6, 7\}$	S3		S4	S3	S3			
*$S_4 = \{11\}$			S4					
$S_5 = \{9, 10\}$							S5	S4
$S_6 = \{8, 11\}$								

Autómata.