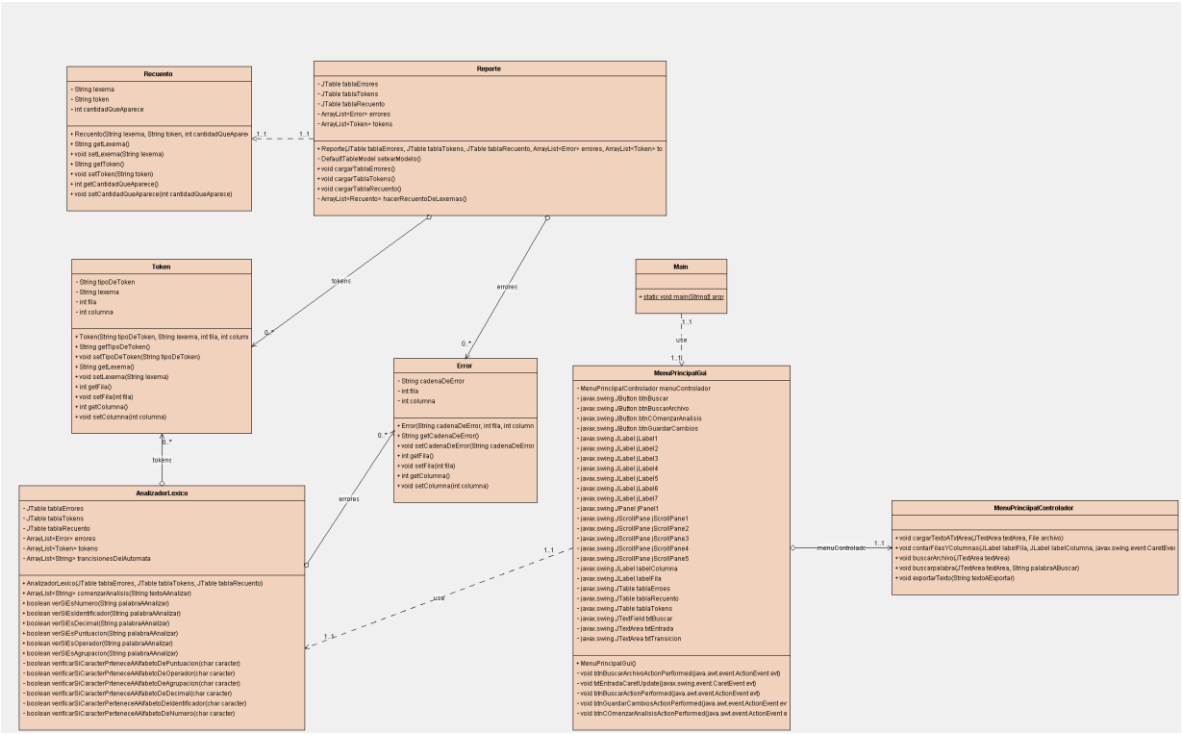
Diagrama de clases



Descripción de cada clase

Clase “AnalizadorLexico”

Esta clase tiene los métodos que identifican un token.

Atributos:

- private JTable tablaErrores;
- private JTable tablaTokens;
- private JTable tablaRecuento;
- private ArrayList<Error> errores = new ArrayList<>();
- private ArrayList<Token> tokens = new ArrayList<>();
- private ArrayList<String> trancisionesDelAutomata = new ArrayList<>();

Métodos:

public AnalizadorLexico(JTable tablaErrores, JTable tablaTokens, JTable tablaRecuento)

- Este es el constructor de la clase.
- Inicializa los atributos de la clase.
- Recibe como parámetros 3 Jtables que serán los que se utilizan para cargar la información de los análisis.

public ArrayList<String> comenzarAnalisis(String textoAAnalizar)

- Devuelve un ArrayList de Strings que corresponderán a todas las transiciones que se hagan dentro el autómata.
- Recibe como parámetro el texto que se analizara.
- Con un for y el método charAt explora cada carácter del String.
- Tiene un if adentro del for que identifica si hay un enter o un espacio en blanco para identificar el final de una palabra y mandar a analizarla. El if tiene un else pues si no se cumple entonces va sumando los caracteres a una palabra.
- Identifica los espacios o los enter para sumar las columnas y las filas correspondientemente.

public boolean verSiEsNumero(String palabraAAnalizar)

- Recibe un String como parámetro que será la palabra que analizaremos.
- Documenta todos los movimientos dentro del autómata.
- Con un for explora cada carácter de la palabra, manda a analizar si el carácter pertenece al alfabeto del autómata número. Si el carácter pertenece al alfabeto entonces se mueven los estados del autómata.
- Si no hubo error agrega las transiciones al array que servirá para generar el reporte de transiciones.
- Al final del for compara si nos encontramos en el estado de aceptación y de ser asi devuelve true de lo contrario devuelve false.

public boolean verSiEsIdentificador(String palabraAAnalizar)

- Recibe un String como parámetro que será la palabra que analizaremos.
- Con un if revisa si el primer carácter de la cadena es una letra si no es entonces acaba el método retornando un false.
- Con un for explora cada carácter de la palabra, manda a analizar si el carácter pertenece al alfabeto del autómata identificador. Si el carácter pertenece al alfabeto entonces se mueven los estados del autómata.
- Si no hubo error agrega las transiciones al array que servirá para generar el reporte de transiciones.
- Devuelve true si al acabar se encuentra en el estado de aceptación del autómata.

public boolean verSiEsDecimal(String palabraAAnalizar)

- Recibe un String como parámetro que será la palabra que analizaremos.
- Con un if revisa si el primer carácter de la cadena es un numero si no es entonces acaba el método retornando un false.
- Con un for explora cada carácter de la palabra, manda a analizar si el carácter pertenece al alfabeto del autómata decimal. Si el carácter pertenece al alfabeto entonces se mueven los estados del autómata.
- Identifica el punto decimal e identifica que no haya más de un punto decimal de ser asi retorna false
- Si no hubo error agrega las transiciones al array que servirá para generar el reporte de transiciones.
- Devuelve true si al acabar se encuentra en el estado de aceptación del autómata.

public boolean verSiEsPuntuacion(String palabraAAnalizar)

- Recibe un String como parámetro que será la palabra que analizaremos.
- Con un for explora cada carácter de la palabra, manda a analizar si el carácter pertenece al alfabeto del autómata Puntuación. Si el carácter pertenece al alfabeto entonces se mueven los estados del autómata.
- Por cada iteración cuenta los chars que ha leído, pues no puede haber más de uno. Si hay más de un char en la cadena entonces devuelve false.
- Si no hubo error agrega las transiciones al array que servirá para generar el reporte de transiciones.
- Devuelve true si al acabar se encuentra en el estado de aceptación del autómata.

public boolean verSiEsOperador(String palabraAAnalizar)

- Recibe un String como parámetro que será la palabra que analizaremos.
- Con un for explora cada carácter de la palabra, manda a analizar si el carácter pertenece al alfabeto del autómata Operador. Si el carácter pertenece al alfabeto entonces se mueven los estados del autómata.
- Por cada iteración cuenta los chars que ha leído, pues no puede haber más de uno. Si hay más de un char en la cadena entonces devuelve false.
- Devuelve true si al acabar se encuentra en el estado de aceptación del autómata.

public boolean verSiEsAgrupacion(String palabraAAnalizar)

- Recibe un String como parámetro que será la palabra que analizaremos.
- Con un for explora cada carácter de la palabra, manda a analizar si el carácter pertenece al alfabeto del autómata Agrupacion. Si el carácter pertenece al alfabeto entonces se mueven los estados del autómata.
- Por cada iteración cuenta los chars que ha leído, pues no puede haber más de uno. Si hay más de un char en la cadena entonces devuelve false.
- Devuelve true si al acabar se encuentra en el estado de aceptación del autómata.

private boolean verificarSiCaracterPrteneceAAlfabetoDePuntuacion(char caracter)

- Retorna true o false dependiendo si caracter == '.' || caracter == ',' || caracter == ';' || caracter == ':'.

private boolean verificarSiCaracterPrteneceAAlfabetoDeOperador(char caracter)

- Retorna true o false dependiendo si caracter == '+' || caracter == '-' || caracter == '*' || caracter == '/' || caracter == '%').

private boolean verificarSiCaracterPrteneceAAlfabetoDeAgrupacion(char caracter)

- Retorna true o false dependiendo si caracter == '(' || caracter == ')' || caracter == '[' || caracter == ']' || caracter == '{' || caracter == '}'.

private boolean verificarSiCaracterPrteneceAAlfabetoDeDecimal(char caracter)

- Retorna true o fale dependiendo si el char enviado se encuentra ente el rango de los codigos ASCII correspondientes a los números del 0 al 9 y el del punto decimal.

private boolean verificarSiCaracterPerteneceAAlfabetoDeIdentificador(char caracter)

- Retorna true o false dependiendo si el char enviado se encuentra en el rango de los códigos ASCII correspondientes a las letras de la ‘‘a” a la “z” minúsculas y mayúsculas.

private boolean verificarSiCaracterPerteneceAAlfabetoDeNumero(char caracter)

- Retorna true o fale dependiendo si el char enviado se encuentra ente el rango de los codigos ASCII correspondientes a los números del 0 al 9.

Clase “Error”

Esta clase es un modelo de cómo se representara un Error léxico dentro de la aplicación.

Atributos:

- private String cadenaDeError.
- private int fila.
- private int columna.

Métodos:

- Getters y Setters de todos los atributos.
- Constructor de la clase que inicializa todos los atributos.

Clase “Token”

Esta clase es un modelo de cómo se representara un Token dentro de la aplicación.

Atributos:

- private String tipoDeToken.
- private String lexeme.
- private int fila.
- private int columna.

Métodos:

- Getters y Setters de todos los atributos.
- Constructor de la clase que inicializa todos los atributos.

Clase “Recuento”

Esta clase es un modelo de cómo se representara un Recuento dentro de la aplicación.

Atributos:

- String lexema.
- String token.
- int cantidadQueAparece.

Métodos:

- Getters y Setters de todos los atributos.
- Constructor de la clase que inicializa todos los atributos.

Clase “Reporte”

Esta clase es la encargada de cargar los arrayList a las tablas del frontend para generar reportes.

Atributos:

- private JTable tablaErrores.
- private JTable tablaTokens.
- private JTable tablaRecuento.
- private ArrayList<Error> errores.
- private ArrayList<Token> tokens.

Metodos:

- Constructor de la clase que inicializa todos los atributos.

private DefaultTableModel setearModelo()

Este método le da un modelo a una tabla, hace que la tabla no sea editable.

public void cargarTablaErrores()

- Establece el texto de las columnas de una tabla.
- Recorre todo los elementos de arrayList de Errores. Luego los agrega a una tupla de la tabla errores.

public void cargarTablaTokens()

- Establece el texto de las columnas de una tabla.
- Recorre todo los elementos de arrayList de Tokens. Luego los agrega a una tupla de la tabla tokens.

public void cargarTablaRecuento()

- Establece el texto de las columnas de una tabla.
- Recorre todo los elementos de arrayList de Recuento. Luego los agrega a una tupla de la tabla recuento.

private ArrayList<Recuento> hacerRecuentoDeLexemas()

- Crea un arrayList de Recuento.
- Con un for recorre cada elemento de el array de Tokens.
- Con un for anidado recorre cada elemento del arrayList de Recuento y verifica si se trata del mismo token, de ser asi entonces suma uno en existencias de token.
- Verifica con una bandera si el elemento existe en la tabla de recuentos de no ser así entonces agrega el token al array de recuento.

Clase “MenuPrincipalControlador”

Es la encargada de regular las acciones que se hagan en el frontEnd y que no sean el análisis léxico como tal.

Metodos:

public void cargarTextoATxtArea(JTextArea textArea, File archivo)

- Con un while lee todas las líneas del archivo que tiene como parámetro.

- Por cada iteración del while agrega una línea nueva a un String.
- Al terminar el while agrega todo el string generado al contenido del TextArea.

public void contarFilasYColumnas(JLabel labelFila, JLabel labelColumna, javax.swing.event.CaretEvent evt)

- Obtiene la posición del cursor dentro del TextArea.
- Sette el texto de las label con las filas y columnas obtenidas.

public void buscarArchivo(JTextArea textArea)

- Crea un file chooser y crea un obketo File a partir de la elección del usuario.
- Manda a cargar el texto del archivo al TextArea.

public void buscarpalabra(JTextArea textArea, String palabraABuscar)

- Crea un objeto DefaultHighlightPainter que sera el que pintara los textos que coincidan con la palabraABuscar.
- Crea un objeto Highlighter para borrar todos los marca textos del texto del JTextArea.
- Con un for explora carácter por carácter el texto del JTextArea.
- Detecta cuando se hay un enter o un espacio pues eso indica el fin de una palabra, compara si la palabra formada es iguala la palabra a buscar y de ser asi la resalta, reinicia la palabra formada y adelanta una unidad el inicio de una palabra.
- Si no hay espacio o enter entonces el else suma a un String los caracteres.

public void exportarTexto(String textoAExportar)

- Crea un JFileChooser y guarda un archivo con la ubicación de que el usuario eligio.