**Introducción:** En la segunda fase del proyecto de la asignatura Teoría de lenguajes formales se procede a la construcción del analizador léxico, en este documento se encontrarán las actividades realizadas antes, durante y después de construir dicho analizador, tales como identificación de tokens, desarrollo de autómatas, estrategias, conclusiones, etc.

**Actividades:**

1. Identificación de Tokens: Los tokens encontrados en el lenguaje proporcionado por la docente fueron los siguientes:
   1. Palabra reservada.
   2. Identificador.
   3. Cadena.
   4. Comentario.
   5. Entero.
   6. Simbolo.
   7. Terminal de línea.
   8. Separador.
2. Definición de los Tokens con expresiones regulares: Para la especificación de los tokens del lenguaje, se utilizaron las siguientes convenciones:

L: Letra mayúscula o minúscula.

M: Letra mayúscula.

m: Letra minúscula.

D: Digito; 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

\_: La raya baja o underscore.

S: Cualquier símbolo excepto los delimitadores de cadena.

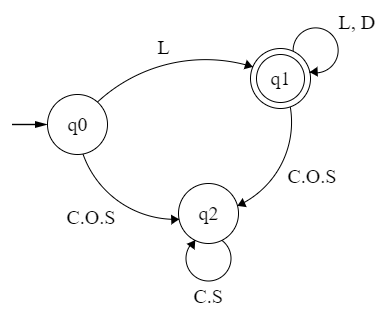
A continuación se muestran las expresiones regulares para cada uno de los tokens identificados anteriormente. Los paréntesis y los corchetes son solo agrupadores, mientras las llaves y las comillas hacen parte de los tokens.

* 1. No se encuentra en este documento la expresión regular para el token palabra reservada ya que puede haber confusión con el token identificador.
  2. L(D U L)\*
  3. “ S\* “
  4. - - S\*
  5. D+
  6. ({) U (})
  7. .
  8. ,

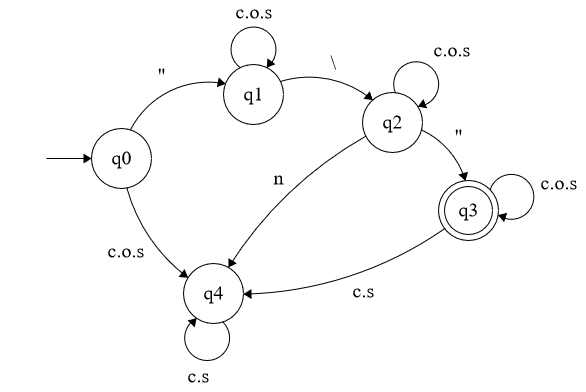
1. Elaboración de diagramas de transición: Los diagramas mostrados a continuación son los Autómatas finitos deterministas (AFD) para cada uno de los tokens.

AFD Palabra reservada: No se agrega el AFD para el token Palabra reservada ya que puede haber confusión con el AFD del token identificador.

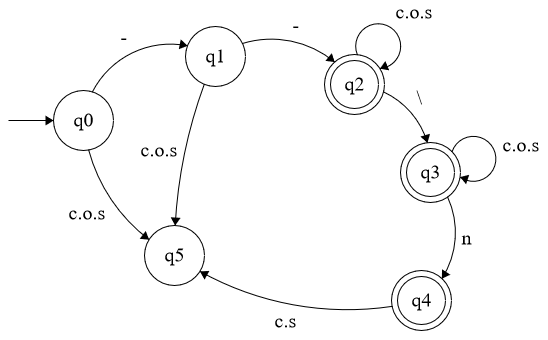
AFD Identificador:



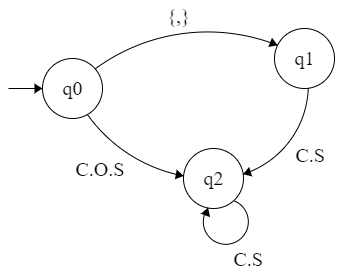
AFD Cadena:



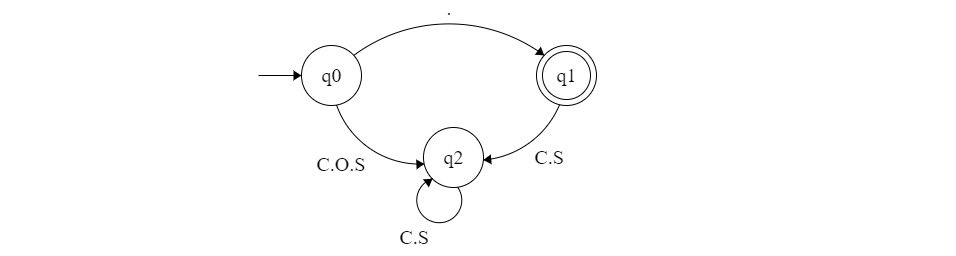
AFD Comentario:



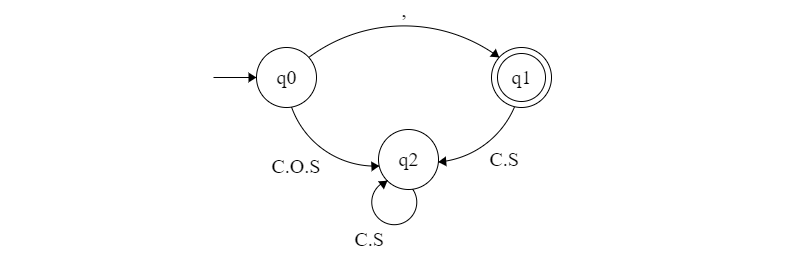
AFD Símbolo:



AFD Terminal de línea:



AFD Separador:



**Implementación del analizador léxico:** Después de realizados las actividades anteriores se procedió a la implementación del analizador en cuestión, el primer paso fue la creación de la interfaz gráfica la cual es un ventana simple que cuenta con una barra de menú, dos áreas de texto y una tabla.

La barra de menús tiene dos menús principales: Archivo y funciones. El menú archivo tiene 4 submenús los cuales son Abrir, Guardar, Guardar como y Salir que como sus nombres indican los 3 primeros sirven para operar sobre los archivos SSQL y el último cierra la aplicación.

En la primer área de texto se ingresa el código SSQL o se visualiza y modifica si ha sido cargado desde la función abrir.

En la segunda área de texto se visualizarán todos los errores encontrados en el código SSQL después de haber sido analizado.

En la tabla que cuenta con 3 columnas: Token, Lexema, Línea, Se visualizan cada uno de los tokens encontrados en el código SSQL con su correspondiente Lexema y la línea en que se encuentra.

Para realizar la interfaz gráfica se hizo uso de los paquetes awt y swing que proporciona java.

Luego de finalizada la parte gráfica se prosiguió a implementar la lógica de la aplicación. La aplicación cuenta con 4 clases lógicas de las cuales 2 fueron proporcionadas por la docente las cuales son AnalizadorLexico.java y Token.java, estas dos clases fueron modificadas para que fueran funcionales con el lenguaje SSQL.

La siguiente clase con la que cuenta la aplicación es FileHandler.java la cual se usa para operar sobre los archivos, bien sea para cargarlos, guardarlos, modificarlos, etc.

Por ultimo está la clase Launcher.java que básicamente ejecuta la aplicación.

**Dificultades durante el desarrollo:** Durante el desarrollo del analizador léxico resultaron algunas preguntas que con ayuda de internet y de la docente de la asignatura fueron resueltas a continuación se señalan algunas de estas dificultades.

La primera duda surgió en el momento que se estaba realizando el método para obtener palabras reservadas y es que no se tenía claro cómo se podría diferenciar los identificadores de las palabras reservadas, la solución fue sencilla, se implementó una lista con todas las palabras reservadas con las cuales cuenta el lenguaje SSQL y cuando se encontraba una palabra sin espacios se buscaba dentro de la lista.

Con la solución anterior surgió un problema y es que algunas de las palabras reservadas pueden contener espacios y esto era un problema, para solucionar esto cada vez que se encontraba una palabra sin espacios se buscaba como prefijo dentro de las palabras reservadas, en caso de ser prefijo se seguía buscando otra palabra, y se hacia el mismo proceso hasta que fuera la palabra acumulada fuera prefijo propio de alguna palabra reservada o hasta que no fuera prefijo. Si no era prefijo se retornaba un nulo.

Otro problema que se considera el más complicado de solucionar se hizo notar cuando se estaba realizando la interfaz gráfica y fue la implementación de la numeración del área de texto donde se encuentra el código SSQL, pues la clase JTextArea no cuenta de forma nativa con esta funcionalidad. Para solucionar esto se recurrió a buscar en Github algún tipo de proyecto donde ya hayan solucionado dicho problema, y se encontró una clase llamada TextLineNumbers.java la cual solucionaba directamente el problema, solo se necesitaba copiar la clase al paquete donde se desarrollaba la interfaz y agregar al scroll que contiene el área de texto un RowHeaderView el cual es una instancia de la clase TextLineNumbers.

**Conclusiones:**

Con el desarrollo de esta fase del proyecto se llegó a la comprensión del uso y aplicación de los autómatas finitos deterministas (AFD) y no deterministas (AFN).

Se destacó la importancia del uso de autómatas para definir lenguajes regulares y conocer cómo a través de estos se facilita la comprensión y análisis de los distintos lenguajes planteados.

Se hizo uso de los autómatas finitos para la representación de los tokens del lenguaje de programación proporcionado.

Así mismo, por medio de esta práctica se comprendió cómo trabajan los diversos lenguajes de programación usando los analizadores léxicos los cuales permiten una posterior traducción de sus tokens para la lectura de la máquina.

Se hizo uso de los autómatas para el análisis de cada uno de los lexemas dados.

**Recursos:** Los recursos necesarios para el desarrollo del analizador léxico los brindo la docente de la asignatura aun así como se dijo anteriormente se necesitó de una clase en especial para la numeración del código, dicha clase puede ser descargada del siguiente enlace: <https://github.com/stevenremot/RaSKlett/blob/master/src/graphicInterface/TextLineNumbers.java>