UNIVERSIDAD AUTONOMA DE CHIAPAS

CONTADURÍA Y ADMINISTRACIÓN CAMPUS I

LICENCIATURA EN INGENIERIA EN DESARROLLO Y TECNOLOGIAS DE SOFTWARE

LUIS EDUARDO GONZALEZ GUILLEN – 6M – A211397

ACT. 1.1 INVESTIGAR ANALIZADOR LÉXICO Y LENGUAJES REGULARES

COMPILADORES

MTRO LUIS ALFARO GUTIERREZ

19 DE AGOSTO DEL 2023

Funciones de un analizador léxico:

1. **Eliminación de Espacios en Blanco y Comentarios**: El analizador léxico también se encarga de eliminar espacios en blanco innecesarios y comentarios, que no son relevantes para el análisis sintáctico o semántico.
2. **Tokenización**: Divide el flujo de caracteres del código fuente en unidades significativas llamadas tokens. Por ejemplo, en el código `int x = 10;`, los tokens podrían ser `int`, `x`, `=`, `10`, y `;`.
3. **Identificación de Palabras Reservadas y Símbolos**: Reconoce palabras reservadas (como `if`, `else`, `while`, etc.) y símbolos especiales (como `+`, `-`, `*`, etc.) en el lenguaje de programación.
4. **Identificación de Identificadores y Literales**: Reconoce identificadores (nombres de variables, funciones, etc.) y literales (como números y cadenas de texto).
5. **Manejo de Errores**: Detecta errores léxicos, como caracteres no válidos, y puede generar mensajes de error o incluso corregir el error de alguna manera.
6. **Almacenamiento en Tabla de Símbolos**: Algunos analizadores léxicos también almacenan información sobre identificadores en una tabla de símbolos para su uso en fases posteriores del proceso de compilación.
7. **Generación de Tokens**: Produce una salida estructurada, generalmente una lista de tokens, que será la entrada para el analizador sintáctico.
8. **Normalización**: En algunos casos, el analizador léxico también realiza tareas como convertir caracteres a minúsculas o mayúsculas para hacer que el análisis sea insensible al caso, aunque esto depende del lenguaje de programación.
9. **Localización y Anotación**: Además de generar tokens, el analizador léxico a menudo también anota con información adicional, como la ubicación en el código fuente donde se encontró el token, lo que puede ser útil para el manejo de errores y la depuración
10. **Optimización**: En algunos casos, el analizador léxico puede realizar pequeñas optimizaciones, como el reconocimiento de constantes numéricas y su conversión a una forma más eficiente.

Explicar el lema del bombo:

El Lema de Bombeo para lenguajes regulares es una propiedad teórica que todos los lenguajes regulares deben satisfacer. Este lema se utiliza a menudo para demostrar que un determinado lenguaje no es regular. La idea básica del lema es que, dado un lenguaje regular, siempre hay una forma de "bombeo" (es decir, de repetir ciertas partes) de las cadenas en el lenguaje para generar nuevas cadenas que también pertenecen al lenguaje.

Explicar las propiedades de la cerradura:

Los lenguajes regulares tienen varias propiedades de cerradura, lo que significa que al aplicar ciertas operaciones a uno o más lenguajes regulares, el resultado también será un lenguaje regular. Estas propiedades son fundamentales en la teoría de la computación y ayudan a entender y manipular lenguajes regulares

Explicar las propiedades de decisión:

Los lenguajes regulares tienen varias propiedades de decisión que los hacen especialmente manejables desde el punto de vista computacional. Una propiedad de decisión es una pregunta sobre el lenguaje que puede ser respondida de manera algorítmica.

Explicar el proceso de determinación de equivalencia de estados:

La determinación de equivalencias entre estados en un autómata finito y lenguajes regulares es un tema importante en la teoría de la computación. Aquí se abordan dos aspectos: la equivalencia entre estados en un autómata y la equivalencia entre diferentes lenguajes regulares.

Explicar el proceso de minimización de un DFA:

La minimización de un Autómata Finito Determinista (DFA, por sus siglas en inglés) es el proceso de obtener un autómata equivalente con el menor número posible de estados. Un DFA minimizado es útil porque es más eficiente en términos de memoria y tiempo de ejecución. El algoritmo más comúnmente utilizado para la minimización de DFA es el algoritmo de partición por refinamiento, también conocido como algoritmo de minimización de estados de Hopcroft

- Paso 1: Eliminar estados inalcanzables
  Antes de minimizar el DFA, es útil eliminar cualquier estado que no sea alcanzable desde el estado inicial, ya que estos estados no afectan el lenguaje reconocido por el autómata.
- Inicializar la partición
  Inicialmente, se divide el conjunto de estados en dos grupos o clases de equivalencia:
  Estados de aceptación
  Estados no aceptantes
- Refinar la partición
  El objetivo es dividir iterativamente estos grupos en subgrupos hasta que no se puedan dividir más. Dos estados pp y qq en el mismo grupo son separados si hay una transición $\delta(p,a)=r\delta(p,a)=r$ y $\delta(q,a)=s\delta(q,a)=s$ tal que rr y ss ya pertenecen a diferentes grupos. En otras palabras, si hay un símbolo aa en el alfabeto tal que al aplicar aa a pp y qq se llega a estados que ya se sabe que son diferentes, entonces pp y qq también deben ser diferentes.