Principios y Técnicas de Compiladores – Cursada 2023
Trabajo Práctico Nro. 1

Fecha de Entrega: 08-09-2023

Objetivo

Desarrollar un Analizador Léxico que reconozca los siguientes tokens:

- Identificadores cuyos nombres pueden tener hasta 25 caracteres de longitud. El primer carácter puede ser una letra o "@", y el resto pueden ser letras, dígitos o "@". Los identificadores con longitud mayor serán truncados y esto se informará como Warning. Los identificadores podrán escribirse en minúsculas o mayúsculas.
- Constantes correspondientes al tema particular asignado a cada grupo.
- Operadores aritméticos: "**+**", "**-**" , "*****", "**/**"
- Operador de asignación: ":**=**"
- Comparadores: "**>=**", "**<=**", "**>**", "**<**", "**=**", "**<>**"
- "**(**" , "**)**" , "**,**" y "**;**"
- Cadenas de caracteres correspondientes al tema particular asignado a cada grupo.
- Palabras reservadas (en minúsculas):

  If
  then
  else
  begin
  end
  end_if
  print
  while
  do
  fun
  return

El Analizador Léxico debe eliminar de la entrada (reconocer, pero no informar como tokens al Analizador Sintáctico), los siguientes elementos.

- Comentarios correspondientes al tema particular asignado a cada grupo.
- Caracteres en blanco, tabulaciones y saltos de línea, que pueden aparecer en cualquier lugar de una sentencia.

Analizador Léxico. Especificaciones

a) El Analizador Léxico deberá leer un código fuente, identificando e informando:
   - Tokens detectados en el código fuente. Por ejemplo:
     Palabra reservada **if**
     (
     Identificador **@Var1**
     +
     Constante entera **25**
     Palabra reservada **else**
     etc.
   - Errores léxicos detectados en el código fuente, indicando: nro. de línea y descripción del error. Por ejemplo:
     Línea 24: Constante entera fuera del rango permitido
   - Contenidos de la Tabla de Símbolos.

Se sugiere la implementación de un consumidor de tokens que invoque al Analizador Léxico solicitándole tokens. En el trabajo práctico 2, esta funcionalidad estará a cargo del Analizador Sintáctico.

b) El código fuente **debe ser leído desde un archivo**, cuyo nombre **debe poder ser elegido** por el usuario del compilador.

c) La numeración de las líneas de código debe comenzar en 1. Si se implementa una interfaz que permite mostrar o editar el código fuente, incluir alguna manera de identificar el número de cada línea del código.

d) Para la programación se podrá elegir el lenguaje. Para esta elección, tener en cuenta que el analizador léxico se integrará luego a un Parser (Analizador Sintáctico) generado utilizando una herramienta tipo Yacc. Por lo tanto, es necesario asegurarse la disponibilidad de dicha herramienta para el lenguaje elegido.

e) El Analizador Léxico deberá implementarse mediante una matriz de transición de estados y una matriz de acciones semánticas, de modo que cada cambio de estado y acción semántica asociada, sólo dependa del estado actual y el carácter leído.

f) Implementar una Tabla de Símbolos donde se almacenarán identificadores, constantes, y cadenas de caracteres. Es requisito para la aprobación del trabajo, que la tabla sea implementada con una estructura dinámica.

g) La aplicación deberá mostrar, además de tokens y errores léxicos, los contenidos de La Tabla de Símbolos.

Entrega

La forma de entrega (correo electrónico, medio físico, etc.) se pactará con la cátedra.
El material entregado debe incluir:
- Ejecutable del compilador
- Código fuente completo del compilador
- Casos de prueba
- Informe

Consideraciones

- Debe controlarse que las constantes estén dentro del rango permitido. Si esta condición no se cumple, se debe considerar que la constante no es válida.
- **Para las constantes que pueden llevar signo, la distinción del uso del símbolo '-' como operador aritmético o signo de una constante, se postergará hasta el trabajo práctico Nro. 2.**

Informe:

Debe incluir:
- Temas asignados (esta información deberá repetirse en los informes de los trabajos prácticos subsiguientes).
- Introducción.
- Decisiones de diseño e implementación.
- Diagrama de transición de estados.
- Matriz de transición de estados.
- Descripción del mecanismo empleado para implementar la matriz de transición de estados y la matriz de acciones semánticas.
- Lista de acciones semánticas asociadas a las transiciones del autómata del Analizador Léxico, con una breve descripción de cada una.
- Errores léxicos considerados.

Casos de Prueba

Se debe incluir, como mínimo, ejemplos que contemplen las siguientes alternativas:
(Cuando sea posible, agregar un comentario indicando el comportamiento esperado del compilador)
- Constantes con el primer y último valor dentro del rango.
- Constantes con el primer valor fuera del rango.
- Identificadores de menos y más de 25 caracteres.
- Identificadores con letras, dígitos y @.
- Identificadores bien y mal definidos
- Palabras reservadas escritas en minúsculas y mayúsculas.
- Comentarios bien y mal definidos.
- Cadenas bien y mal definidas.

Temas particulares

Cada grupo de trabajo tendrá asignada una combinación de temas particulares.

Tipos de datos:

Grupo 1

- **Enteros**: Constantes enteras con valores entre -2^{15} y $2^{15} - 1$.
  Se debe incorporar a la lista de palabras reservadas la palabra **integer**
- **Enteros largos sin signo:** Constantes enteras con valores entre 0 y $2^{32} - 1$.
  Se debe incorporar a la lista de palabras reservadas la palabra **ulongint**

Nota: Si la constante está en el rango de los enteros (-2^{15} y $2^{15} - 1$) se considerará que es de tipo **integer**. Cuando sea positiva y esté en el rango 2^{15} y $2^{32} - 1$, será de tipo **ulongint**

Grupo 2

- **Enteros sin signo**: Constantes enteras con valores entre 0 y $2^{16} - 1$.
  Se debe incorporar a la lista de palabras reservadas la palabra **uinteger**
- **Enteros largos:** Constantes enteras con valores entre -2^{31} y $2^{31} - 1$.
  Se debe incorporar a la lista de palabras reservadas la palabra **longint**

Nota: Si la constante está en el rango de los enteros sin signo (0 y $2^{16} - 1$) se considerará que es de tipo **uinteger** Cuando sea negativa o esté en el rango 2^{16} y $2^{31} - 1$, será de tipo **longint**

Conversiones

Grupo 1: Incorporar a la lista de palabras reservadas la palabra **itoul**.

Grupo 2: -.

Comentarios

Grupo 1: Comentarios de 1 línea: Comentarios que comiencen con "**##**" y terminen con el fin de línea.

Grupo 2: Comentarios multilínea: Comentarios que comiencen con "***/**" y terminen con "**/***" (estos comentarios pueden ocupar más de una línea).

Cadenas

Grupo 1: Cadenas multilínea: Cadenas de caracteres que comiencen con "**<<**" y terminen con **">>"** . Estas cadenas pueden ocupar más de una línea. (En la Tabla de símbolos se guardará la cadena sin el salto de línea.

Grupo 2: Cadenas de 1 línea: Cadenas de caracteres que comiencen y terminen con " **"** " (estas cadenas no pueden ocupar más de una línea).