TP 2 – Técnicas de Compilación

UBP UNIVERSIDAD Blas Pascal

- **Carrera:**
  Ingeniería Informática
- **Materia:**
  Técnicas de compilación
- **Docente:**
  Maximiliano A. Eschoyez
- **Integrantes:**
  Arreguez, Germán Rodrigo
  Ventura, Gino

Consigna

Dado un archivo de entrada en lenguaje C, se debe generar como salida el árbol sintáctico (ANTLR) correcto y mostrar por consola o archivo el contenido de la tabla de símbolos de cada contexto. Esto es válido únicamente cuando el archivo de entrada es correcto.
Para esta versión se deberá realizar un control de errores básicos y generar un reporte de lo encontrado para poder analizar archivos con posibles errores de codificación. El reporte de errores podrá realizarse por consola o archivo de texto.
Los errores a detectar deben ser los siguientes:
Errores sintácticos comunes:

- Falta de un punto y coma.
- Falta de apertura de paréntesis.
- Formato incorrecto en lista de declaración de variables.

Errores semánticos comunes:

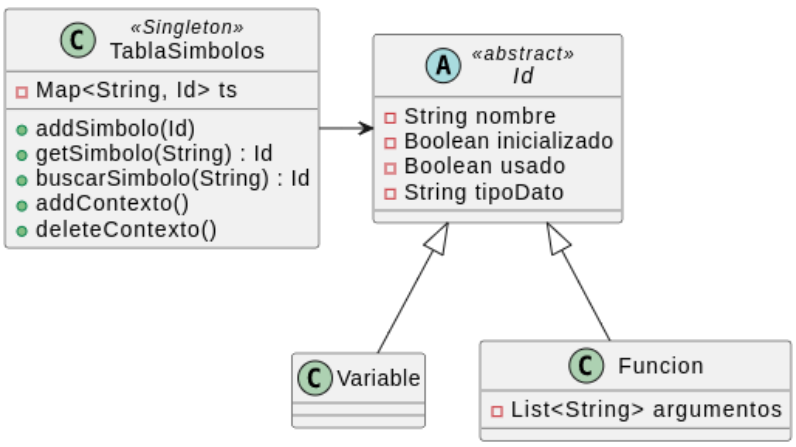
- Doble declaración del mismo identificador.
- Uso de un identificador no declarado.
- Uso de un identificador sin inicializar.
- Identificador declarado pero no usado.
- Tipos de datos incompatibles.

Cada error reportado debe indicarse si es sintáctico o semántico.

Desarrollo

En el trabajo práctico número 1, realizamos el análisis léxico gracias a las expresiones regulares que revisaban la ortografía determinando si eran palabras propias del lenguaje y a esta secuencia de caracteres las convertía en tokens (secuencia de tokens). Luego esta secuencia ingresaba al análisis sintáctico que se encargaba de analizar si los tokens conformaban una estructura correcta (oraciones bien escritas).
Para este trabajo práctico número 2, avanzamos en el proceso y llegamos al análisis semántico, en donde verificamos que todo lo que está escrito tenga algún sentido. Para poder avanzar con este análisis debemos incorporar una Tabla de símbolos en donde vamos a ir guardando los tokens (nombre de funciones y variables). Esta tabla nos va a permitir saber si la variable a analizar está declarada o no, inicializada, usada, también si existe o no en un contexto esperado o está en uno de mayor jerarquía, etc. En el caso de las funciones va a pasar lo mismo, vamos a poder guardar la lista de argumentos de cada una con sus respectivos tipos, el tipo de dato que retornan, cantidad de argumentos que reciben, etc.
Junto con el análisis semántico incorporamos el manejo de errores.

Para desarrollar el trabajo práctico partimos de la base:



Conclusión

Este trabajo práctico nos pareció mucho más desafiante que la primera parte ya que debimos refactorizar casi todas nuestras reglas gramaticales para facilitar la creación de la tabla de símbolos así como agregar y eliminar contextos de la misma utilizando el lenguaje Java.
También nos pareció interesante cómo funcionan los Listener y el manejo de errores personalizados que no tuvimos en cuenta en el TP1.
Ver las clases en video nos ayudó mucho para reforzar lo visto en clases.