Instituto Universitario Aeronáutico

Facultad de Ingeniería

DESARROLLO DE HERRAMIENTAS DE SOFTWARE
TRABAJO PRÁCTICO Nº 1

Alumnos:

Manfredi Angelo

Yanes Kevin

Consigna

Dado un archivo de entrada en lenguaje C, se debe generar como salida el Árbol Sintáctico (ANTLR) correcto. Para lograr esto se debe construir un parser que tenga como mínimo la implementación de los siguientes puntos:

Reconocimiento de un bloque de código, que puede estar en cualquier parte del código fuente, controlando el balance de llaves.

Verificación de:

- declaraciones y asignaciones,
- operaciones aritmético lógicas,
- declaración/llamada a función.

Verificación de las estructuras de control if, for y while.

Además, al finalizar el reconocimiento de la entrada, se deberá mostrar mediante un archivo de texto plano el contenido de la Tabla de Símbolos para cada contexto.

Desarrollo

Conceptos teóricos

ANTLR: Es una herramienta de reconocimiento de lenguaje que opera sobre lenguajes, proporcionando un marco para construir reconocedores (parsers), intérpretes, compiladores y traductores de lenguajes a partir de las descripciones gramaticales de los mismos (conteniendo acciones semánticas a realizarse en varios lenguajes de programación).

Gramática: Una **gramática** proporciona una estructura a un lenguaje de programación, siendo más fácil generar código y detectar errores. Es más fácil ampliar/modificar el lenguaje si está descrito con una **gramática**. La mayor parte de este tema está dedicada a los métodos de análisis sintáctico de uso típico en **compiladores**.

Analizador léxico: Es la primera fase de un compilador, consistente en un programa que recibe como entrada el código fuente de otro programa (secuencia de caracteres) y produce una salida compuesta de *tokens* (componentes léxicos) o símbolos. Estos *tokens* sirven para una posterior etapa del proceso de traducción, siendo la entrada para el analizador sintáctico (en inglés *parser*).

Analizador sintáctico: Analiza una cadena de símbolos según las reglas de una gramática formal. Convierte el texto de entrada en otras estructuras (comúnmente árboles), que son más útiles para el posterior análisis y capturan la jerarquía implícita de la entrada.

Analizador semántico:
El **analizador semántico es** el que emplea la información contenida en los atributos de los componentes léxicos. Recuerda que el sintáctico sólo ve las etiquetas de las categorías. En cuanto a error, suponemos que representa las acciones necesarias para tratar el error.

Procedimiento

Para poder desarrollar la consigna se declaró **reglas gramaticales**, se implementó un analizador **léxico** , un analizador **sintáctico** y un analizador semántico.

Para las reglas gramáticales se tuvo en cuenta que el nombramiento sea intuitivo y simple de entender. Para ver dichas reglas definidas:
https://github.com/angelo59930/compiladores/blob/main/src/main/python/compiladores.g4

Para el analizador léxico creamos el **listener** correspondiente(en nuestro caso lo llamamos “MyListener”) y en este lo que vamos a hacer es tratar los tokens que nos interesa para guardarlos en **una tabla de símbolos.**
La tabla de símbolos es una clase en la cual se encuentra un arreglo de diccionarios/maps en el cual vamos a guardar los distintos **contextos** que posee el archivo que estamos compilando y las **variables que se encuentran declaradas** en cada contexto. Para guardar estas variables vamos a crear más clases como la de **Id**, **Variable** y **Función** la cual nos van a ayudar a poder analizar correctamente el archivo.

Repositorio

- GitHub: https://github.com/angelo59930/compiladores