Construcción del analizador lexico, sintactico y semantico de un compilador

Integrante:

Nombre y apellido: Pablo Alexis Gaido
Dirección de gmail: pgaido524@alumnos.iua.edu.ar

Introducción:

En el presente informe se explicará con detalle cómo fue realizada la implementación del analizador léxico, semántico y sintáctico de un compilador de lenguaje C escrito en lenguaje Python 3 utilizando la librería ANTLR.

Definiciones:

ANTLR: Es una herramienta de reconocimiento de lenguaje escrita en JAVA que se utiliza para declarar la gramática del lenguaje, denominado "metalenguaje".

Analizador léxico: Consistente en un programa que recibe como entrada el código fuente de otro programa y produce una salida compuesta de tokens o símbolos como (if,for,while) entre otros.

Analizador sintáctico: Es un programa informático que analiza una cadena de símbolos según las reglas de una gramática formal.

Analizador semántico: El analizador semántico utiliza un árbol sintáctico y la información de la tabla de símbolos para comprobar la consistencia semántica del programa fuente con la definición del lenguaje.

Desarrollo:

Analizador léxico: Para implementar el analizador léxico se utilizó una estructura “clave - valor”, por ejemplo : clave → “variable”, valor →”i”.

De esta forma el analizador léxico va recorriendo todo el código fuente buscando caracteres que coincidan con las características de uno o más valores y luego asociarlos a sus respectivas claves, también llamados Tokens .

Analizador sintáctico: El analizador sintáctico fue construido utilizando los tokens declarados por el Analizador léxico como bloques de construcción.

Analiza la ubicación de los tokens y verifica que coincidan con la estructura del lenguaje que se desea compilar y de esta forma ir identificando bloques de código más complejos como: “for(i = 0; i < 10;i++){... }”.

Una vez recorrido todo el código fuente se construye un árbol sintáctico que se estructura asociando todos los bloques de código a nodos, se declara un nodo raíz y de ahí se va ramificando colocando los bloques de código más complejos más cerca de la raíz y los más simples más cerca de las hojas.

Analizador semántico: La tabla de símbolos fue construida utilizando el “listener” que me ofrece la librería ANTLR.

En el listener se va evaluando (en tiempo de ejecución) la coherencia de los tokens almacenados en el árbol sintáctico, tal que cada token esté asociado con información tal como la ubicación, el tipo de datos y el ámbito de cada variable, constante o procedimiento.

Repositorio Github:

URL: https://github.com/Pablo592/DHS-Cursado

Estructura:
Ubicación del proyecto: DHS-Cursado/demo/
Código fuente a ser compilado: DHS-Cursado/input/
Tabla de símbolos: DHS-Cursado/output/
Clases utilizadas: DHS-Cursado/demo/src/main/python/clases.py
Analizador léxico y sintáctico: DHS-Cursado/demo/src/main/python/compiladores.g4
Listener: DHS-Cursado/demo/src/main/python/MiListener.py
Este mismo informe PDF: DHS-Cursado/Informe