UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de Ingeniería

Laboratorio 1

Esteban Zambrano Garoz – 22119

Construcción de Compiladores

Guatemala, julio 2025

Enclace a repositorio:

https://github.com/EstebanZG999/Compis2_Lab1

Enlace a Video:

https://youtu.be/q3ZL8lDaZsQ

Análisis de Gramatica ANTLR (Minilang.g4)

Al principio del archivo tenemos el encabezado y el nombre de la gramática, el cual declara el nombre de la gramática que se va a utilizar para que todos los archivos que se vayan a generar usen este nombre.

- grammar MiniLang;

A continuación se tiene las reglas del parser y del lexer. Las reglas del parser, como ya sabemos, indican y definen la estructura de lenguajes utilizando expresiones combinando tokens.

- stat: expr NEWLINE # printExpr
- | ID '=' expr NEWLINE # assign
- | NEWLINE # blank
- ;

Cada alternativa termina con un label, que genera una clase de contexto; por ejemplo, el label #Muldiv genera la clase de contexto MulDivContext. Esta parte es útil para implementar los visitors y listeners, ya que se puede especializar el comportamiento según la regla que se esté definiendo.

- #MulDiv

Las reglas del parser definen los patrones de caracteres que van a producir los tokens. Se encuentran operadores como la multiplicación división.

- MUL : '*' ; // define token for multiplication
- DIV : '/' ; // define token for division

Por ejemplo en las reglas está. WS : [\t]+ -> skip ; // toss out whitespace, que se usa para descartar cualquier espacio y tabulador para que estos no pasen al parser.

Algunos detalles importantes que caben destacar es que en las expr, se permite expresar sin precedencia y agrupación implícita de operaciones. También ANTLR elige la primera que case, el orden importa para que se puedan llegar a evitar ambigüedades.

Análisis de Driver.py

Al principio del archivo se tiene los imports, para importar las funciones del parser y lexer.

- from MiniLangLexer import MiniLangLexer
- from MiniLangParser import MiniLangParser

En la funcion principal tenemos que primero se va a leer el archivo de texto que se este pasando linea por linea, para que el lexer pueda leer todos los caracteres que hay en este mismo.

- input_stream = FileStream(argv[1])

Luego tenemos el MinilargLexer que convierte los caracteres en tokens como MUL, ID, DIV.

- lexer = MiniLangLexer(input_stream)

Después se almacenan los tokens en un buffer para que el parser los consuma y pueda realizar el parseo de cada uno de estos tokens.

- stream = CommonTokenStream(lexer

Como siguiente paso, se crea un parser con las configuraciones con la gramática que habíamos definido antes, el cual genera un árbol de parseo.

- parser = MiniLangParser(stream)

Por último, se invoca prog para que se lean tokens hasta que no se cumpla ninguna stat. Si hay errores, ANTLR los envía para que salgan en la terminal, mientras que si no hay errores no sale nada en terminal.