Guía Práctica 2 - MII779
Análisis Sintáctico en ANTLR
Desarrollo de Lenguajes Orientados a Objeto y Compiladores Avanzados

Profesor: Ricardo Soto

Ejercicio 1: Implemente el analizador sintáctico para el lenguaje Mile++

- Descargue el archivo `milePP-syntactic.zip` desde `http://www.inf.ucv.cl/~rsoto/cursos/MII7XX/milePP-syntactic.zip`
- Descomprima y copie en su workspace.
- Cree un proyecto seleccionando como fuente del proyecto la carpeta recientemente descomprimida.
- Complete la gramática `MilePPParser.g` con las reglas faltantes (declaración de variables, estructuras de control, expresiones, métodos, etc).
- Compile la gramática.

Ejercicio 2: Comprenda la nueva clase `Tool`

- Diríjase a `src->cl.ucv.inf.mileppcompiler->Tool`. Esta clase permitirá comprobar su analizador sintáctico.
- La clase Tool.java contiene un nuevo método llamado `syntacticTest`. Este método se encarga de invocar a las clases responsables de realizar el análisis léxico y sintáctico. Las instrucciones realizadas en este método son las siguientes:
  - Creación de un objeto lexer a partir del archivo fuente.
  - Creación de un objeto parser utilizando el objeto lexer.
  - Invocación del método `programa`. Note que la compilación de la gramática genera un método para cada regla de la gramática. De esta forma se puede iniciar el proceso de análisis sintáctico llamando al método correspondiente a la primera regla, en este caso `programa`.
  - Creación de un objeto `ASTFrame`. Esta clase permite desplegar en pantalla el AST generado en una interfaz gráfica.

Ejercicio 3: Pruebe el analizador sintáctico

- Ejecute la clase `Tool` utilizando como programa fuente el archivo `examples/test1.mile`. La ejecución debería desplegar el AST generado.
- Explore el AST y verifique que la organización jerárquica de los elementos en el AST sea coherente con la gramática del lenguaje Mile++.
- Cree y ejecute un archivo de test para probar las estructuras de control (`for, if`).

- Cree y ejecute un archivo de test que incluya todos los operadores soportados en Mile++, verifique si las prioridades[1] y el uso de paréntesis se ha especificado en forma correcta.
- Cree y ejecute un archivo de test incluyendo estructuras gramaticales NO pertenecientes al lenguaje Mile++.

Ejercicio 4: Extienda el analizador sintáctico

- Agregue una regla para declarar variables de un mismo tipo en conjunto (ej: `numeric a,b,c;`)
- Agregue una regla para definir accesos (p.ej. `a.b.c()`).

Ejercicio 5: Implemente y pruebe el analizador sintáctico de su lenguaje.

---

[1] Mile++ utiliza las prioridades estándar de los operadores matemáticos.