UBP UNIVERSIDAD Blas Pascal

Técnica de Compiladores

Trabajo Práctico nº 1

Ferreyra Tomás

UBP UNIVERSIDAD Blas Pascal

Consignas

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.

Ante el primer error léxico o sintáctico el programa deberá terminar.

Resolución

Respondiendo a las consignas solicitadas se procedió a la realización del código. Nos enfrentamos rápidamente ante diferentes incógnitas que se han debido resolver para poder completar la resolución del trabajo:

¿Cómo se debe analizar la estructura del código que debe ser analizado por el compilador que se está construyendo? ¿Cuál es el punto de vista adecuado para aproximar el análisis del código presente?

Tenemos el conocimiento básico mediante el cual se analiza el código y se lo puede descomponer en instrucciones y/o conjunto de instrucciones, y la combinación entre las mismas. Luego, corresponde analizar la estructura del código en busca de patrones mediante los cuales se pueda agrupar cierto conjunto de instrucciones, basados también en el comportamiento que presentan dichas instrucciones.

Agrupando el código mediante dichos patrones, se puede establecer una estructura de análisis de código mediante el cual el compilador puede evaluar correctamente si el texto provisto cumple con los lineamientos léxicos y sintácticos que se deben corresponder con el código de programación del lenguaje C.

Conclusión

De los análisis mencionados concluimos que todo código del lenguaje C se puede estructurar de la siguiente forma:

- En el entorno global, el código aceptado puede estar dentro de alguno de los siguientes patrones:
    - Asignación: variables a las cuales se les asigna un valor
    - Declaraciones: definición de nuevas variables, con sus correspondientes tipos. Las declaraciones pueden ser simples o con asignaciones incluidas
    - Declaración de función: se define el tipo devuelto por la función, el ID que la identifica; y el conjunto (puede ser vacío) de tipos de parámetros que recibe la función, los cuales pueden ir acompañados de un ID de variable o no
    - Definición de función: se respeta la “firma” de la declaración de la función, sin embargo, el conjunto de parámetros que se reciben deben si o si estar acompañados de ID de variable.
- Luego, cualquier otro entorno se encontrará definido dentro de las definiciones de funciones
- Dentro de una definición de función, se presentan los siguientes patrones:
    - Instrucción simple: se distinguen los siguientes tipos
        - Asignación
        - Declaraciones
        - Operaciones aritmético-lógicas: Consisten en lo siguiente:
            - Operaciones aritméticas:
                - Suma, resta, multiplicación, división, módulo
                - Valores propiamente dichos: enteros, caracteres, strings, **llamadas a funciones**
            - Operaciones lógicas: negación, and, or, comparaciones
        - Declaración de función: se pueden declarar funciones dentro de otras funciones, pero la definición de las funciones debe ser siempre en el entorno global del código
    - Estructuras de control:
        - Incluimos las siguientes estructuras de control:
            - Condicional IF/ELSE
                - IF (*argumentos*) *acciones* . Adicionalmente puede tener
                    - ELSE IF (*argumentos*) *acciones*
                    - ELSE *acciones*
            - Bucle WHILE
                - WHILE (*argumentos*) *acciones*
            - Bucle FOR
                - FOR (*condiciones iniciales, argumentos, argumentos*) *acciones*
        - Todas las estructuras de control tienen un conjunto de *acciones* las cuales pueden ser una instrucción simple o un bloque, el cual es un conjunto de instrucciones simples encerradas en un par de llaves (**{}**)
    - Return: instrucción de retorno de función. Tiene un comportamiento similar al de las instrucciones simples, sin embargo, tiene sus particularidades propias, por lo que decidimos separarlas del resto

- Como se puede observar, las llamadas a funciones se encuentran categorizadas dentro del grupo de valores propios. Ésto se debe a que en el lenguaje C, toda llamada a una función devuelve un valor, así sea, el valor VOID.

Referencias

El código del proyecto completo se puede encontrar subido en un repositorio git en GitHub.
Link del repositorio: https://github.com/tomaslicenciado/facultad-TC/tree/main/tp1