Universidad del Valle de Guatemala
Facultad de Ingeniería
Departamento de Ciencias de la Computación
CC3032 - Construcción de Compiladores

Laboratorio 0
Generación de scanner y parser

A. Familiarizarse con YAPL

Con el objetivo de familiarizar con el lenguaje de programación a compilar durante el curso, se sugiere realizar las siguientes actividades:

1. Leer a detalle los documentos de Especificación del Lenguaje YAPL y Aspectos Léxicos y Sintácticos de YAPL, que se encuentran en los archivos del curso.
2. Dado que YAPL se basa en COOL, utilizar el debugger de COOL (http://dijkstra.eecs.umich.edu/eecs483/cool-debugger/) para escribir un programa simple en YAPL y familiarizarse con sus aspectos léxicos y sintácticos.
3. Familiarizarse con ejemplos de programas escritos en YAPL. Estos ejemplos abarcan la mayoría de los conceptos a evaluar durante la entrega de cada proyecto (ya se encuentran en Canvas)

B. Generación de scanner y parser

Para la generación del analizador léxico y sintáctico de YAPL se recomienda utilizar la herramienta ANTLR (www.antlr.org) (Another Tool for Language Recognition). ANTLR es una herramienta generadora de parsers muy utilizada para la construcción de herramientas de software que se enfoquen en traducir y/o procesar lenguajes. ANTLR construye un parser que puede generar y recorrer árboles de análisis sintáctico [parse trees].

Para utilizar ANTLR, se recomienda leer la documentación respectiva de su uso, junto a la selección del lenguaje de programación a utilizar con dicha herramienta. Existen otras herramientas que son posibles de utilizar de igual forma, tales como CoCo/R, GoldParser, Lex/Bison. Su uso queda a discreción del estudiante con conocimiento del profesor del curso.

Cabe mencionar que la gramática presentada en la documentación del curso no se encuentra escrita en ninguna sintaxis particular para alguna de esta herramienta (ni siquiera BNF). Es parte del trabajo traducir esta gramática a la sintaxis de la herramienta seleccionada para procesar el lenguaje, así como tomar en cuenta los conceptos de precedencia de operadores y la resolución de conflictos *desplazamiento-reducción* y *reducción-reducción.*

Scanner

- Para la construcción del scanner, se debe de seguir la especificación de la estructura léxica de YAPL.
- Los posibles errores léxicos encontrados deben resolverse de forma amigable sin que el programa termine abruptamente y que sea posible seguir procesando el resto de la entrada [Recuperación de errores]
- Se debe implementar un *token* tipo ERROR*,* el cual será trasladado al parser junto al resto de *tokens identificados.* Este token tendrá aparte del lexema que da el error, un mensaje legible del error para imprimirse en fases posteriores.
- Existe limitante de tamaño para el token *string*, si el lexema para este *token* es excedido, se debe de reportar y el *scanner* debe seguir procesando el resto de la entrada.
- Si un token tipo *string* posee un caracter de nueva línea (no escapado), reportar el error y continuar con el procesamiento en la siguiente línea del archivo.
- Para los comentarios, no se debe *tokenizar* el contenido de un comentario bajo ningún aspecto.
- Para todos los tokens, incluyendo errores, se debe de guardar la línea en dónde son encontrados.
- La salida del *scanner* es una lista de parejas <token, lexema>. La codificación interna de los tokens queda a discreción del desarrollador. Esta lista será consumida por el *parser* para la construcción del árbol de análisis sintáctico.

Parser

El principal resultado del *parser* es la generación y visualización de un árbol de análisis sintáctico (parse tree).

- La raíz del árbol debe ser el símbolo inicial de la gramática generada.
- Al encontrar uno o más errores sintácticos, se deben de reportar sin generar árbol alguno.
- El *parser* se encargará de procesar un sólo archivo de YAPL.
- El *parser* ya generado no debe de mostrar o generar mensajes de error tipo *desplazamiento-reducción* o *reducción-reducción*. Estos ya tienen que haberse resuelto de antemano.

C. Evaluación

La calificación se realizará de forma presencial, durante el periodo de clase. Para obtener el puntaje completo, el *scanner y parser* deberán de ser capaces de procesar sin problema los archivos de prueba (elaborados por los estudiantes). Cada estudiante deberá subir a Canvas un archivo comprimido con el código fuente respectivo. La falta del código fuente se tomará como falta de realización del laboratorio.