

UTN - FRBA

Sintaxis y Semántica de los Lenguajes - 2022 Trabajo Práctico Integrador Analizador léxico, sintáctico y semántico de C

Implementar en C un programa que realice el análisis léxico, sintáctico y semántico de un archivo fuente del lenguaje C generando su respectivo reporte sobre el análisis realizado. El mismo será un **modelo simplificado** (tomar una GIC reducida) que solamente realizará **determinadas validaciones semánticas**.

Las validaciones semánticas serán:

- Al menos un control de tipos de datos en alguna operación binaria (a elección de cada grupo) -> Utilizar TS si operamos con variables
- Control de doble declaración de variables (se consideran que **todas las variables están en el mismo ámbito** para simplificar la validación) -> Utilizar TS
- Control de cantidad y tipos de parámetros en la invocación a funciones -> Utilizar TS

Las validaciones sintácticas (TP 4):

- Sentencias (todos los tipos: compuesta, If, While, Salto, For, Expresión, etc)
- Declaraciones de variables (puede agregar variables tipo puntero y tipo arreglo) de forma correcta almacenando en TS.
- Declaraciones de funciones de forma correcta almacenando en TS.
- Expresiones (que están incluidas dentro de las sentencias).

El programa deberá reportar un informe en pantalla con

- Lista de variables declaradas indicando el tipo de cada una. (Utilizar TS)
- Lista de funciones declaradas indicando el tipo de parámetro que devuelve y cantidad y tipos de parámetros que recibe. (Utilizar TS)
- Error léxicos encontrados (si los hay) - (Implementar en Flex, archivo.L)
- Errores sintácticos encontrados (si los hay) (Implementar en Bison, utilizar token error)
- Errores semánticos encontrados (si los hay) (Implementar Rutinas Semánticas con TS).

El criterio para el manejo de errores es el siguiente:

- Ante un error léxico, capturar la secuencia de caracteres no reconocida (pueden utilizar una última regla con comodín), en este caso, no le llegará ningún token al analizador sintáctico. Pero puede suceder que un error léxico, desencadene un error sintáctico en caso de que el Parser no reciba los tokens adecuados para la GIC implementada.

- Ante un error sintáctico, utilizar el token “error” para capturar el error y almacenar la información que consideren importante para registrar el mismo. Luego seguir procesando a partir de donde encuentran un punto y coma ‘;’, o bien un salto de línea ‘\n’.
- Ante un error semántico, no es necesario salvar el error dado que la gramática independiente de contexto seguirá operando sin problemas. Pero deben implementar dentro de las rutinas semánticas la impresión de algún mensaje en pantalla, o bien, almacenar los errores encontrados en una lista para imprimirlo al final del análisis en el reporte.

La entrega de este trabajo práctico es obligatoria y será en conjunto con defensa del mismo los días ***lunes 31 de octubre (K2001)*** y ***martes 1 de noviembre (K2052)***. Deben estar presentes con micrófono y cámara todos los integrantes del grupo. La defensa tendrá una duración aproximada de 15 minutos en los cuales todos los integrantes deberán responder preguntas referidas al trabajo práctico integrador.

El **entorno de programación queda a criterio de cada grupo de trabajo** (Eclipse, Dev, Codeblocks, Visual Studio Code). Se recomienda un IDE que esté integrado con Git para poder realizar el trabajo en equipo de una forma más práctica.

El **entorno de programación queda a criterio de cada grupo de trabajo** (Eclipse, Dev, Codeblocks, ***Visual Studio Code***). Se recomienda un IDE que esté integrado con Git para poder realizar el trabajo en equipo de una forma más práctica.

Formará parte de la entrega, como mínimo, el **archivo fuente (.c)** y el **archivo ejecutable (.exe)**. Todo debe estar en el repositorio GitHub.

La **entrega será a través del repositorio** de GitHub en la carpeta correspondiente a cada TP.

Las **consultas** podrán ser respondidas a través Discord. Es importante que los utilices para compartir sus dudas con el resto de los compañeros.

Muchos éxitos =)