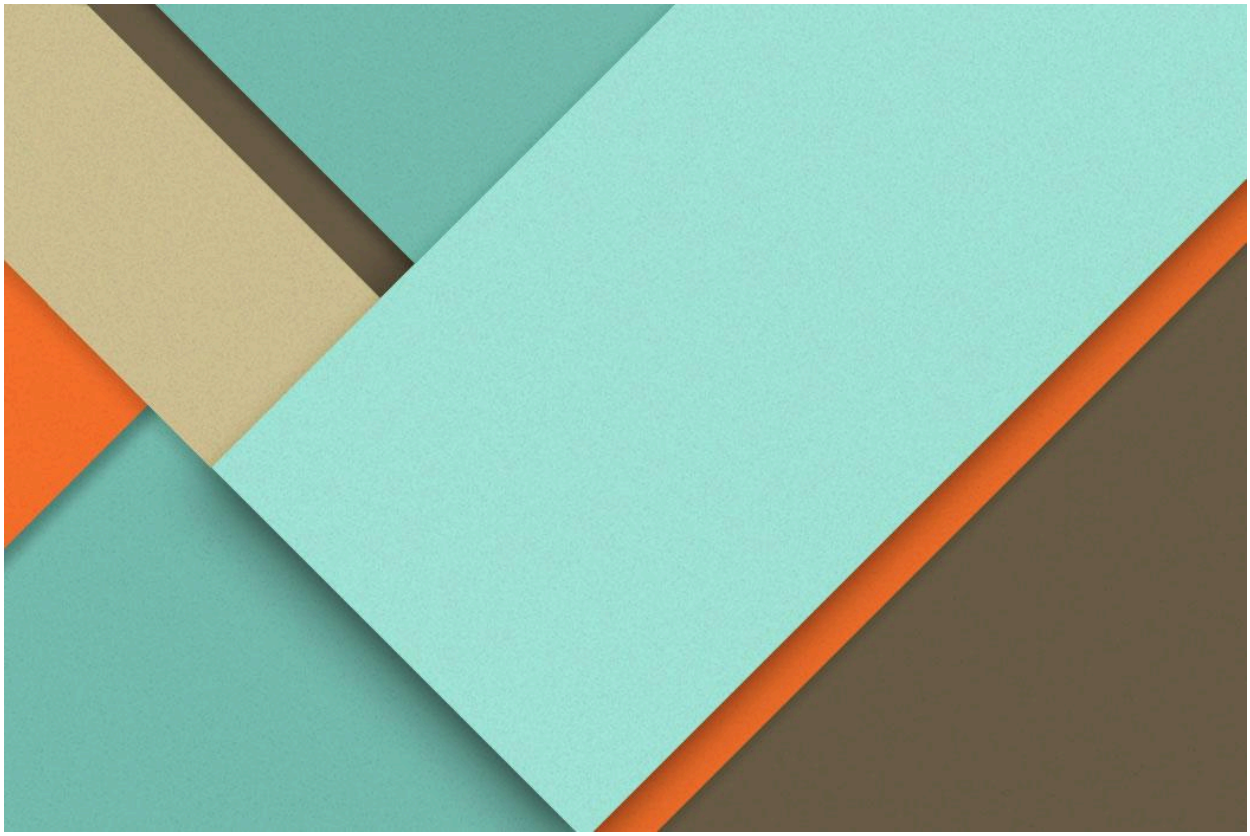

Trabajo Práctico 3 - Técnicas de Compilación

—

Bustos Rios Lucas Nicolas, de Aragón Juan Manuel
Profesor: Eschoyez, Maximiliano

UBP

Introducción

El objetivo de este trabajo práctico es agregar al Trabajo Práctico 1 la tabla de símbolos y el informe de errores. El programa a desarrollar debe tomar un archivo de código fuente en una versión reducida del lenguaje C, posiblemente con errores léxicos, sintácticos y semánticos, y generar como salida el árbol sintáctico (ANTLR) y la tabla de símbolos, además de un informe de errores .

Desarrollo

Definición del problema

La problemática abordada en esta etapa del proyecto consiste en extender las funcionalidades del analizador desarrollado previamente para que, además de generar el árbol sintáctico, sea capaz de:

- Reconocer bloques de código en cualquier parte del archivo fuente, controlando el equilibrio de llaves.
- Verificar declaraciones y asignaciones, operaciones aritméticas, y declaración/llamada a funciones.
- Generar un informe de errores sintácticos y semánticos.

Implementación de la Solución

Para realizar estas extensiones, utilizamos los contenidos adquiridos en clases impartidas por el profesor. Se definieron las expresiones regulares y reglas sintácticas necesarias para que el programa desarrollado pueda interpretar de forma léxica y sintáctica un código fuente en lenguaje C, y así generar el árbol sintáctico (ANTLR).

1. **Declaración de Expresiones Regulares** : Se declararon las expresiones regulares necesarias para la correcta interpretación del código, tales como palabras reservadas, variables, operadores y expresiones lógicas, entre otras.
2. **Desarrollo de Reglas Sintácticas** : Se desarrollaron las reglas sintácticas correspondientes para cada instrucción. Algunas de estas reglas incluyen bucles `while` y `for` operadores aritméticos y lógicos, sentencias `if` y funciones. Las

reglas sintácticas se anidaron para complejizarlas y alcanzar el resultado e

3. **Tabla de Símbolos** : Se implementó una tabla de símbolos que almacena información sobre los identificadores declarados en el código fuente. Esta tabla permite realizar verificaciones semánticas durante el análisis del código.

4. **Control de errores** :
    - **Errores Sintácticos** :
        - Falta de un punto y coma.
        - Falta de apertura de paréntesis.
        - Formato incorrecto en la lista de declaración de variables.
    - **Errores semánticos** :
        - Doble declaración del mismo identificador.
        - Uso de un identificador no declarado.
        - Uso de un identificador sin inicializar.
        - Identificador declarado pero no usado.
        - Tipos de datos incompatibles.

5. **Reporte de Errores** : Se desarrolló un sistema para reportar los errores detectados. Este sistema puede generar un informe en consola o en un archivo de texto, indicando el tipo de error (sintáctico o semántico) y proporcionando detalles para facilitar su corrección.

6. **Pruebas y Validación** : Se realizaron pruebas con diversos ejemplos para validar el correcto funcionamiento del parser, la generación del árbol sintáctico, la tabla de símbolos y la detección de errores. Las pruebas permitieron asegurar que todos los componentes funcionaran de manera integrada y precisa.

Hitos

El desarrollo de esta etapa del compilador resultó en un aprendizaje significativo sobre cómo un compilador analiza sintáctica y léxicamente un código fuente, y cómo se puede extender esta funcionalidad para incluir la generación de una tabla de símbolos y un informe de errores. El proyecto permitió profundizar en el manejo de errores sintácticos y semánticos, y en la importancia de estas herramientas en el desarrollo de compiladores eficientes y robustos.