UNIVERSIDAD MARIANO GALVEZ DE GUATEMALA
FACULTAD DE INGENIERIA EN SISTEMAS
COMPILADORES
ING. MIGUEL CATALAN

NOMBRE	NUMERO DE CARNE
Angel Enrique Juarez Castellanos	7590-21-21054

Guatemala, 31/05/2024

Informe de proyecto

Desafíos encontrados.

Adaptabilidad de la Gramática:

Una de las partes más complejas del proyecto fue desarrollar una gramática que pudiera adaptarse a distintos escenarios y no quedara sesgada a solo uno. La gramática del lenguaje define las reglas sintácticas que el compilador debe seguir, y lograr que estas reglas sean lo suficientemente flexibles para cubrir todos los casos posibles sin perder funcionalidad.

Compatibilidad de Versiones:

Otro de los inconvenientes importantes fue encontrar las versiones correctas y compatibles de JFlex y CUP con la versión de Java que estaba utilizando. No todas las versiones de JFlex y CUP son compatibles con todas las versiones de Java. Esto implicó un proceso de prueba y error para identificar las combinaciones adecuadas, lo que a veces llevó a problemas de configuración y dependencia que necesitaban ser resueltos.

Falta de Conocimiento y Documentación:

La falta de conocimiento profundo sobre JFlex y CUP y la limitada documentación disponible fueron obstáculos notables. Aunque ambas herramientas son poderosas para la construcción de compiladores, su uso efectivo requiere una comprensión detallada de sus funcionalidades y limitaciones. La documentación oficial y los recursos educativos disponibles no siempre proporcionan ejemplos claros o guías detalladas.

Optimización del Rendimiento:

Asegurarse que el compilador funcione de manera eficiente también fue un desafío. La optimización del rendimiento implica no solo asegurar que el código generado sea eficiente, sino también que el proceso de análisis y compilación sea rápido y consuma la menor cantidad de recursos posible. Esto requirió ajustes finos tanto en la gramática como en el código generado.

Decisiones de diseño

Definición de la Gramática del Lenguaje Fuente:

La gramática del lenguaje fuente fue definida según las especificaciones proporcionadas en el documento del proyecto. Para darle funcionalidad a estas gramáticas, utilicé clases de apoyo que facilitaron la implementación de las reglas gramaticales y la integración con los analizadores léxico y sintáctico. Estas clases de apoyo permitieron encapsular las estructuras y operaciones necesarias para la correcta interpretación y procesamiento de las distintas construcciones del lenguaje. Además, me aseguré de que la gramática fuera clara y comprensible.

Definición del Manejo de Errores:

El manejo de errores fue diseñado para identificar y reportar de manera precisa los errores léxicos y gramaticales que pudieran surgir durante el análisis del código fuente. En el analizador léxico, definí errores específicos para detectar tokens inválidos o mal formados, proporcionando mensajes de error detallados que incluyen la ubicación exacta del error en el código fuente. Para los errores gramaticales, utilicé las capacidades de CUP para capturar y reportar discrepancias en la estructura del código, especificando el tipo de error y el contexto en el que ocurrió.

Definición de la Gramática:

Para generar la gramática del lenguaje, utilicé todos los conceptos y técnicas vistos en clase, asegurándome de que la gramática fuera robusta y capaz de manejar una amplia variedad de construcciones sintácticas. Una de las consideraciones más importantes fue la factorización por la izquierda, una técnica esencial para evitar conflictos y ambigüedades en la gramática y garantizar que el analizador sintáctico generado por CUP pueda procesar el código de manera eficiente.

Lecciones aprendidas

Uno de los aspectos más importantes que aprendí durante este proyecto fue la importancia de no olvidar los conceptos básicos de la programación. A lo largo del desarrollo del compilador, utilicé una variedad de conceptos fundamentales que se enseñan al inicio de la carrera, como estructuras de datos, control de flujo, y buenas prácticas de codificación. Estos conocimientos básicos fueron cruciales para poder abordar y resolver los desafíos que surgieron de manera efectiva. Además, aprendí la importancia de dedicar tiempo suficiente para estudiar y comprender nuevas herramientas y tecnologías. En este proyecto, trabajé con JFlex y CUP, que eran nuevas para mí. Invertir tiempo en leer la documentación, seguir tutoriales y hacer pruebas me permitió comprender mejor cómo funcionan estas herramientas y cómo integrarlas en mi proyecto. Esta preparación previa fue esencial para evitar problemas mayores durante la implementación y para poder solucionar cualquier inconveniente que apareciera. Otra lección valiosa fue la importancia de planificar y realizar pruebas exhaustivas. Al tener un buen entendimiento de la gramática y de cómo manejar los errores, pude diseñar casos de prueba que me ayudaron a verificar que el compilador funcionara correctamente en diferentes escenarios. Esto no solo mejoró la calidad del proyecto, sino que también me dio la confianza para hacer ajustes y mejoras sin temor a introducir nuevos errores.