Proyecto #1

Bryam Steven López Miranda

Carlos Akion Garro Campos

Unidad desconcentrada de computación, Instituto Tecnológico de Costa Rica

Compiladores e Intérpretes

Prof. Óscar Víquez Acuña

12 de Septiembre de 2022

Análisis del lenguaje

En la parte de la gramática de Monkey Programming Language se logró introducir los tokens más comunes del lenguaje con los cuales se pueden crear algoritmos de distintos tipos así como operaciones sobre elementos creados en ellos, entre los cuales se encuentra la creación de variables con el token let, return,declaración de bloques if con else, creación de funciones, llamadas a funciones, comparadores, operadores aritméticos básicos, tipos de datos como Integer, String, Boolean, expresiones, creación de listas, diccionarios y acceso ítems de los mismos además de otras funciones sobre arrays, se puede crear código con una sintaxis similar a la de C++. No está incluido la creación de códigos de iteración, tales como ciclos while y for.

Soluciones e implementación

Para el proceso de solución del analizador léxico(lexer), analizador sintáctico(parser) y el AST se han realizado la creación de las clases respectivas por medio de la herramienta de autogenerado que brinda ANTLR4. Cada una de estas clases es sumamente importantes por lo que se detalla a continuación las implementaciones respectivas:

- **Lexer:** es el encargado de guardar todas las palabras reservadas del lenguaje, el tipo de tokens y los nombres específicos de cada uno y además de los métodos necesarios para reconocer cada de estas cuando se presentan en el lenguaje.
- **Parser:** es más compleja donde se implementa todas las reglas de sintaxis del lenguaje, esta clase está vinculada con el lexer, ya que tiene que hacer uso de todas las palabras reservadas para ser tomadas en cuenta en la sintaxis. Por otra parte se crea el método “program” que es el principal de la gramática, este llama a los métodos que tiene en el cuerpo y estos llaman a otros sucesivamente para poder formar todas las reglas del lenguaje.

El intérprete tipo REPL fue construido web utilizando la especificación estándar HTML5 con el uso de JavaScript, CSS3 y Bootstrap donde se incluyeron 2 áreas de texto para el REPL donde 1 textarea está destinado a escribir muchas líneas de código y además marca las líneas de código existentes, el otro textarea funciona para la visualización de errores y ejecución de líneas individuales de código. Además se incluye un botón para cargar un archivo de texto con código Monkey y el botón de correr código.

La conexión entre la interfaz gráfica y el programa en Python fue llevado a cabo utilizando la tecnología Eel que permite llamar funciones de python utilizando una etiqueta "@eel.expose" arriba del nombre de la función facilitando tanto la llamada de funciones así como el envió y obtención de información por parte del programa.

Resultados obtenidos

Para una mejor visualización de los resultados obtenidos en la realización del proyecto se muestra el cuadro 1 donde se puede ver una columna con los puntos a realizar del proyecto, otra columna con el estado (Finalizado, No finalizado) y una columna final de descripción.

Objetivo	Estado	Descripción
Reconocer tokens	Finalizado	Logrado
Devolver token según su posición en el archivo(tipo,lexema,fila y columna)	Finalizado	Logrado
Errores léxicos	Finalizado	Logrado
Creación del parser	Finalizado	Logrado

Gramática de Monkey	Finalizado	Logrado
Construir el Árbol de Sintaxis Abstracto	Finalizado	Logrado
Interfaz REPL (Read-Eval Print Loop)	Finalizado	Logrado
Apartado para ejecución de instrucciones y evaluación en el momento de las mismas.	Finalizado	Logrado
Opción para cargar archivos y compilar/correr los.	Finalizado	Logrado
Informar errores generados por el parser.	Finalizado	Logrado

Nota: Cuadro 1: análisis de resultados. Elaboración propia.

Conclusiones

En sintaxis todos los requerimientos del proyectos fueron llevados a cabo y finalizados con mucha satisfacción, durante el proceso se reforzó la utilización de la herramienta ANTLR4 para la creación de un Lexer, Parser y Árbol de forma óptima, además la implementación de la interfaz REPL se puso crear con las características esperadas, siendo esta muy intuitiva y sencilla de utilizar. Durante la ejecución quedaron bases bien establecidas sobre lo que implica la creación de un analizador de texto para un lenguaje de programación, los pasos que su creación implica, herramientas y los conocimientos necesarios para la correcta ejecución del mismo.

Bibliografía

GitHub - ChrisKnott/Eel: A little Python library for making simple Electron-like HTML/JS GUI apps. (s. f.). GitHub. Recuperado 11 de septiembre de 2022, de https://github.com/ChrisKnott/Eel

antlr4/index.md at master · antlr/antlr4. (s. f.). GitHub. Recuperado 11 de septiembre de 2022, de https://github.com/antlr/antlr4/blob/master/doc/index.md

Chui, C. (2022, 4 mayo). *Enable Line Numbering to any HTML Textarea - Weekly Webtips*. Medium. Recuperado 11 de septiembre de 2022, de https://medium.com/weekly-webtips/enable-line-numbering-to-any-html-textarea-35e15ea320e2

Cree una interfaz de usuario HTML usando Eel en Python – Acervo Lima. (s. f.). Recuperado 11 de septiembre de 2022, de https://es.acervolima.com/cree-una-interfaz-de-usuario-html-usando-eel-en-python/

ANTLR4 Terminate on Lexer/Parser error Python. (2015, 21 noviembre). Stack Overflow. Recuperado 11 de septiembre de 2022, de https://stackoverflow.com/questions/33847547/antlr4-terminate-on-lexer-parser-error-python

How to choose a file with Javascript? (2016, 17 mayo). Stack Overflow. Recuperado 11 de septiembre de 2022, de https://stackoverflow.com/questions/37285014/how-to-choose-a-file-with-javascript