

Proyecto #2 Interprete V1

Bryam Steven López Miranda

Carlos Akion Garro Campos

Unidad desconcentrada de computación, Instituto Tecnológico de Costa Rica

Compiladores e Intérpretes

Prof. Óscar Víquez Acuña

16 de Octubre de 2022

**Soluciones e implementación**

Una vez ejecutadas correctamente las etapas anteriores del Scanner y Parser se procede a recorrer el árbol generado, esto para poder ejecutar las instrucciones que se encuentran en el código ingresado. Para la ejecución se hace uso del patrón de diseño Visitor que nos brinda la facilidad de poder recorrer toda la estructura generada por el Parser, esto por medio de visitas a las expresiones y por último llegando a los terminales. Dentro de las visitas toda la información importante para la ejecución de las instrucciones es agregada a las estructuras de almacenamiento creadas para la administración de memoria a nivel local.

Las estructuras utilizadas fueron un HashMap y una Pila que tienen las siguientes funcionalidades:

**· HashMap:** Es el encargado de guardar todas las asignaciones de memoria en todo el momento en el que el intérprete se encuentre en ejecución, puede guardar diferentes tipos de datos como String, Integer, HashLiteral, boolean y arrayLiteral. Esta estructura fue creada como una clase donde tiene diferentes métodos como agregar, eliminar, obtener un valor e imprimir información

**· Pila:** Es una estructura auxiliar que nos ayuda a la hora de evaluar expresiones en diferentes nodos y posteriormente mandar a guardar los datos, esta pila la estructura utilizada por Python fue un Array que nos proporciona todo lo necesario para hacerla funcionar como LIFO.

Por otra parte, se creó una clase REPL que fue implementada con un singleton para asegurarse de contar con una sola instancia siempre, dicha clase contiene un HashMap y una Pila y esta clase va a estar disponible en todo el momento en que el intérprete se esté ejecutando sin importar cuantas veces se crean y recorren los árboles ya que es totalmente independiente.

**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 |
| Recorrer el árbol manualmente | Completo | Se visitan de manera propia sin utilizar children |
| Creación de HashMap | Completo |  |
| Creación Pila | Completo |  |
| Creación REPL | Completo |  |
| Se agregan variables de diferentes tipos al almacén | Completo |  |
| Se imprime los valores | Completo |  |
| Creación y guardar de HashLiteral en Data | Completo |  |
| Creación y guardar de ArrayLiteral en Data | Completo |  |
| Creación de operaciones de lógicas entre las expresiones | Completo |  |
| Creación de IfExpression | Completo |  |
| Funciones de los Arreglos  (LEN | FIRST | LAST | REST | PUSH) | Completo |  |
| Mostrar en pantalla los resultados | Completo |  |

**Conclusiones**

Todos los requerimientos para el proyecto fueron realizados satisfactoriamente por el equipo de trabajo, se analizó, comprendió y ejecutó el funcionamiento del visitor junto con las estructuras del REPL HashMap y Pila sin mayores contratiempo implementando lo requerido para la entrega, además, se mejoró la interacción con la consola significativamente.

**Referencias**

GeeksforGeeks. (2021, 27 agosto). *Singleton Pattern in Python - A Complete Guide*. Recuperado 14 de octubre de 2022, de <https://www.geeksforgeeks.org/singleton-pattern-in-python-a-complete-guide/>

Urooj, W. (2022, 26 marzo). *Hash Tables and Hashmaps in Python: What are they and How to implement?* Edureka. Recuperado 14 de octubre de 2022, de <https://www.edureka.co/blog/hash-tables-and-hashmaps-in-python/>

*Hash Map in Python*. (2012, 2 enero). Stack Overflow. Recuperado 14 de octubre de 2022, de <https://stackoverflow.com/questions/8703496/hash-map-in-python>

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>