Universidad del Valle de Guatemala

Facultad de Ingeniería

Departamento de Ciencias de la Computación

CC2003 - Algoritmos y Estructuras de datos

Proyecto No. 1

Fase 2

Jonathan Espinoza – 20022
Juan Galicia – 20298
Elisa Samayoa – 20710

Estructuras del Java Collections Framework

Las estructuras pertenecientes al JCF utilizadas para el traductor de Lisp a Java son las siguientes:

- Collection
    - List
        - ArrayList: usa un array dinámico para almacenar objetos y elementos, permite duplicados, mantiene el orden de la inserción y no está sincronizada, asimismo, se puede acceder a ellos al azar.

          Esta es una de las estructuras más utilizadas en todo el código. En primera instancia, es útil para ir separando por la expresión introducida e ir verificando los espacios entre cada String. Así también, para conservar el código ya balanceado y verificado, listo para su uso en las operaciones. Se usa también como parámetro para los métodos a lo largo del intérprete para ir conservando las respuestas o ir separando los elementos necesarios para operar.

          En la clase de Verificacion.java y Funciones.java se ve más de uso .

        - Stack: implementación del vector con la estructura de datos “último en entrar es el primero en salir”, contiene todos los métodos del vector, además de proveer métodos booleanos adicionales como “Push(), peek(), push(object o)”.

En el intérprete, los arraylist se usan para hacer las verificaciones de código para balanceo del código y validar que haya sido escrito correctamente, pues se debe asegurar que hayan sido introducidos la misma cantidad de paréntesis para poder leer la expresión.

Para observar con mayor detalle su implementación, se encuentra la clase Verificación.java; sin embargo, sus métodos provienen en Stack.java a partir de la interfaz iStack.java

- Dequeue
  - ArrayDeque: es una estructura de datos lineal que permite insertar y eliminar elementos por los dos extremos, por lo que, implementa en una única estructura las funcionalidades de las pilas y las colas.

    Es útil para la creación y definición de las funciones introducidas por el usuario (DEFUN) para poder conservarlas mientras el programa esté abierto y poder emplearlas en el momento que el usuario desee.

    Fue implementado en la clase de Funciones.java

- Map
  - LinkedHashMap: contiene valores únicos, puede incluir una llave nula y múltiples valores nulos. No está sincronizada, se diferencia del HashMap pues mantiene el orden de la inserción de elementos.

    Puesto que las variables creadas en setq deben ser guardadas para que el usuario pueda acceder a ellas en cuanto las necesite, se usó este LinkedHashMap para ir guardando el nombre de la variable como la key y el valor de la variable como el value. Así, poder mostrarlas en cuanto fueran llamadas desde el programa principal de nuevo.

    Ha sido empleado en la clase de Funciones.java

  - HashMap: utiliza la estructura de datos de Hash Table. No mantiene orden ni tolera los valores duplicados. Solo permite un valor de llave único y múltiples valores de nulos. No está sincronizada.

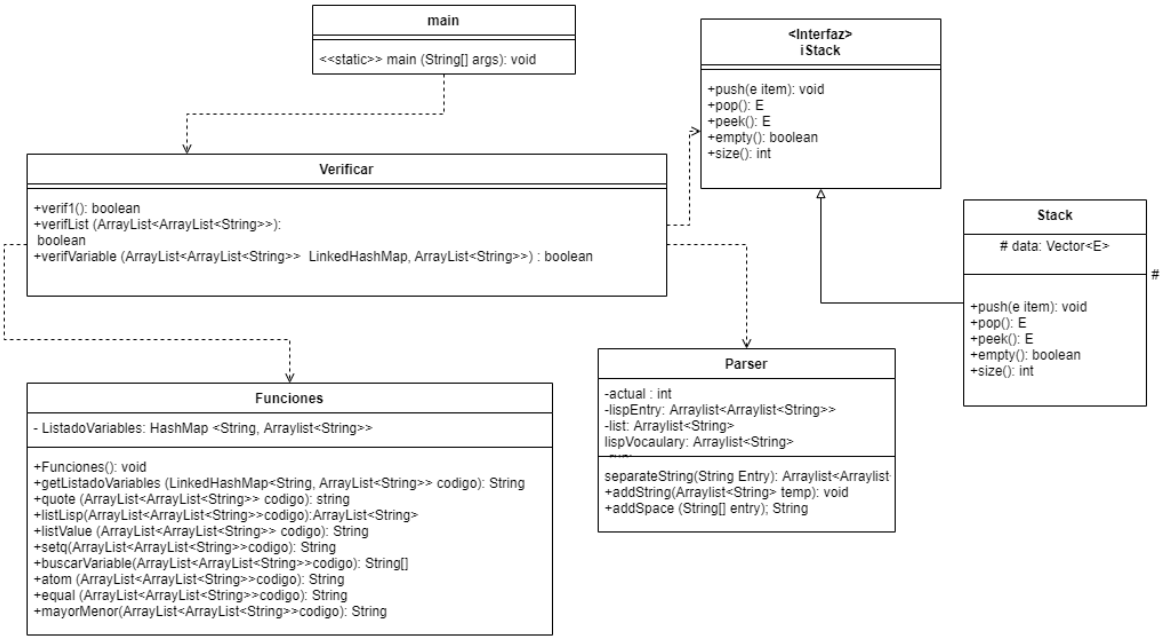
    Puesto que es necesario guardar a las funciones para un uso posterior según lo indique el usuario, el HashMap fue implementado para que se guardara el nombre de la función como el key y el proceso a realizar, el método o las instrucciones que se deben seguir en dicha funciones, para ser guardadas com value. De esta manera, sería más sencillo ubicar a la función y dar resultado en cuanto fuese necesario.

    También fue mayormente implementado en la clase de Funciones.java

UML

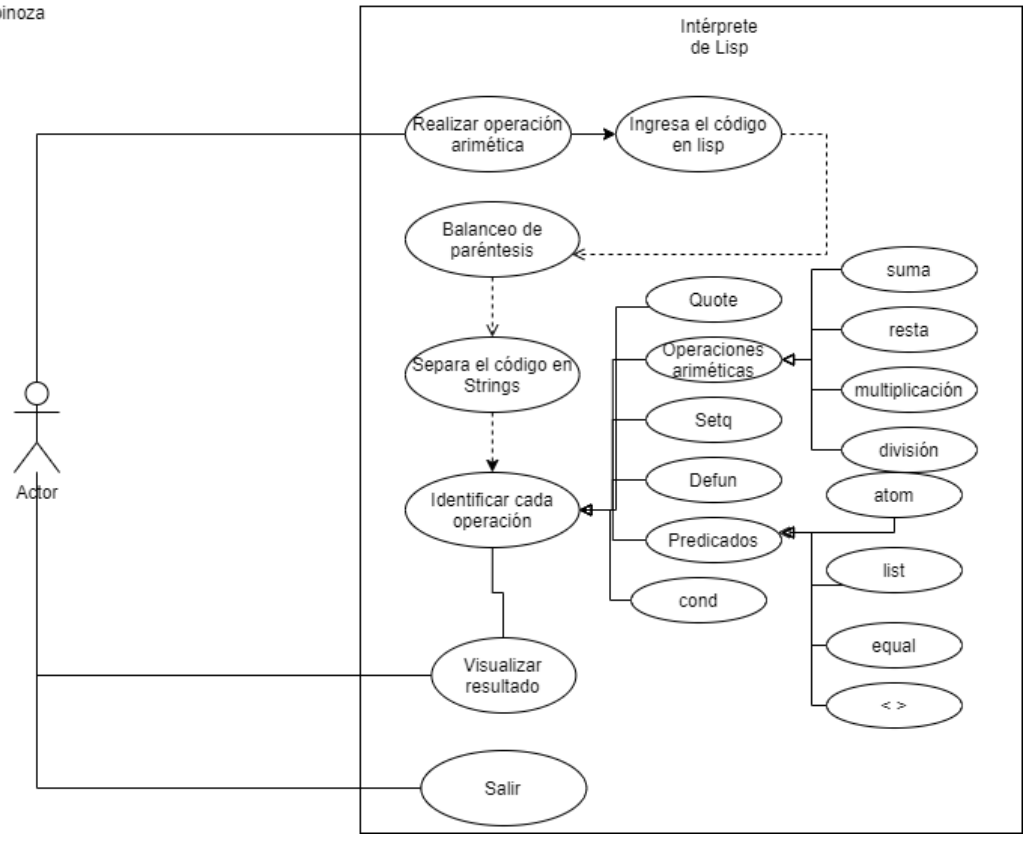
De clases

Elisa Samayoa
Juan Andrés Galicia
Jonathan Espinoza

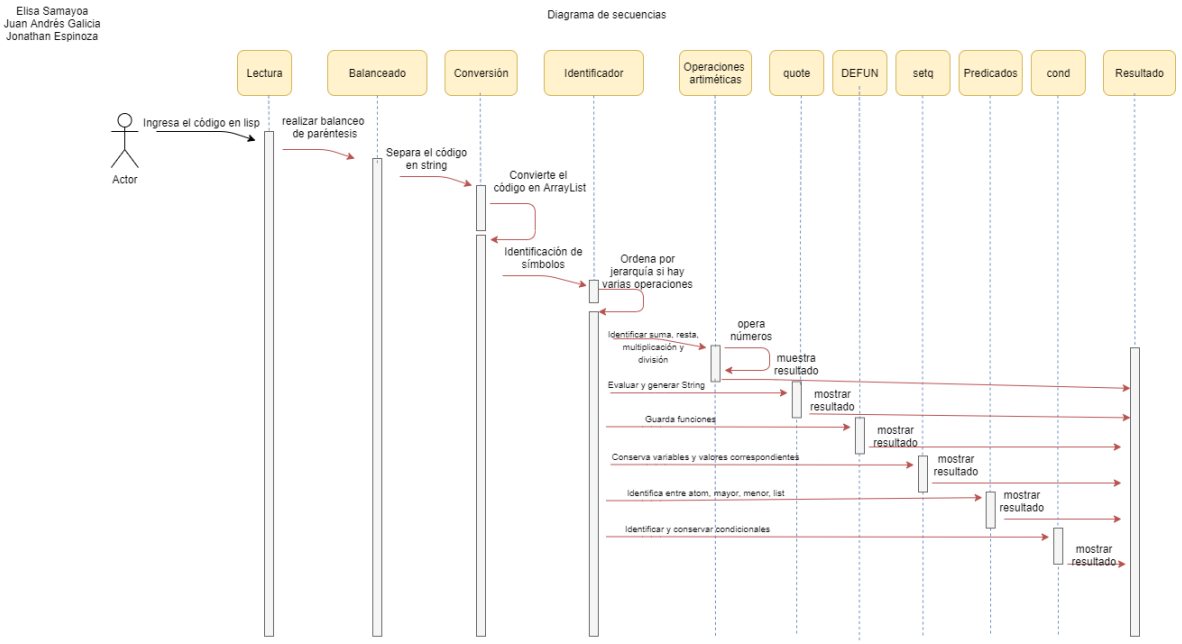


De caso de uso

Elisa Samayoa
Juan Andrés Galicia
Jonathan Espinoza



De secuencia



Bibliografía

5.3.- Implementación de interfaces. | PROG09.- Utilización avanzada de clases. (s.f.). Implementación de Interfaces. https://ikastaroak.birt.eus/edu/argitalpen/backupa/20200331/1920k/es/DAMDAW/PROG/PROG08/es_DAMDAW_PROG08_Contenidos/website_53_implementacin_de_interfaces.html

Collections Framework Overview. (s.f.). Oracle. https://docs.oracle.com/javase/8/docs/technotes/guides/collections/overview.html

Gupta, J. 2019. Java : Collection Framework Hierarchy. Facing Issues On IT https://facingissuesonit.com/2019/10/15/java-collection-framework-hierarchy/