Universidad de los Andes José Nicolás Cárdenas (201922006) Estructuras de Datos y Algoritmos – Sección 6 Andrés Leonardo Beltrán (202014143) **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Laboratorio 3: Observaciones **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

1. ***¿Cuáles son los mecanismos de interacción (I/O: Input/Output) que tiene el view.py con el usuario?***

Hay tres funciones que generan interacción con el usuario:

1. *printMenu.*
2. *printAuthorData.*
3. *printBestBooks.*

Adicionalmente, el *While* relacionado a la impresión del menú y la escogencia de las opciones también genera una interacción con el usuario:

Texto

Descripción generada automáticamente

1. ***¿Cómo se almacenan los datos de GoodReads en el model.py?***

Los datos de *GoodReads* se almacenan en una estrucutura de datos que se genera mediante la función *newCatalog* de *Model.py*, la cual implementa el método *newList* de la clase *list*. Dicha estructura contiene la información de los libros, los autores y los tags. La función mencionada define el diccionario *catalog*, el cual será el que retorna la función. Las parejas llave-valor de este son cadenas de caracteres y arreglos vacíos (respectivamente); la única excepción es el elemento con identificador *authors*, ya que este no especifica si es una lista enlazada o un arreglo. Es importante notar que todas estas listas están vacías, ya que la información es cargada en el módulo *controller.py*.

1. ***¿Cuáles son las funciones que comunican el el view.py y el model.py?***

El proceso inicia con la función *newCatalog* de *controller.py,* ya que esta trae todas las estructuras de datos guardades en el módulo *model.py*. Adicionalmente, todas las funciones que inician con el prefijo *load* de *controller.py* son las encargadas de cargar a lista *catalog* toda la información guardada en la carpeta *Data*. Finalmente, el *view.py* se comunica con *controller.py* a través de las funciones *initCatalog* y *loadData*.

1. ***¿Cómo se crea una lista?***

Mediante la función *newList*. Este crea el TAD lista que se usará en el proyecto.

1. ***¿Qué hace el parámetro cmpfunction=None en la función newList()?***

Este parámetro especifica la función que se usará para comparar los elementos que se van a añadir a la lista con un orden establecido.

1. ***¿Qué hace la función addLast()?***

Se encargar de agregar un último elemento al final de la lista.

1. ***¿Qué hace la función getElement()?***

Esta función permite obtener un elemento de la lista de una posición especificada.

1. ***¿Qué hace la función subList()?***

Retorna una sublista que hace parte de la lista original; es decir, esta crea una lista que contiene los elemenos de la lista que van desde la posición ingresada por parámetro y hasta *numelem* menos uno (-1).

1. ***¿Observó algún cambio en el comportamiento del programa al cambiar la implementación del parámetro “ARRAY\_LIST” a “SINGLE\_LINKED”?***

No, el comportamiento del programa siguió siendo el mismo, y todas las operaciones realizadas funcionaron de la misma manera. Aun así, pese a que no se pueda percibir directamente, haber cambiado a la listas encadenadas implicó mayor uso de espacio, ya que fue necesario añadir y guardar información relacionada a las referencias a cada nodo/elmento de la lista.