Laboratorio 3, Reto 1

Tomas Angel Gallon - 202020366

Daniel Osorio Cardenas - 2020222996

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

Las interacciones I/O que se dan en el view.py se encuentran en el Menu Principal en la linea 87. Empieza por pedirle al usuario un numero, el cual hace referencia a cada uno de los comandos encontrados en el menu: “1- Cargar informacion en el catalogo”, “2- Consultar los Top x libros por promedio”, etc. Cada uno de los comandos hace una accion devuleve una informacion diferente dependiende del comando que el usuario pidio. Adicionalmente, para el segundo comando, se le pide al usuario una cantidad x para que el programa devuelva los x libros mas altos dentro del promedio. Asi mismol, el tercer comando le pide al usuario el nombre de uno de los autores para que este devuelva todos los libros que dicho autor ha escrito. Por ultimo, el cuarto comando pide unos de los ‘tags’ de los libros para que este mismo devuelva la cantidad de libros que estan clasificados dentro de esa misma etiqueta.

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

Los datos de Goodreads se almacenan en el model .py siguiendo unos modelos que se definen. Primero se crea el modelo del catálogo que es donde se almacenan todos los libros en una lista, al igual que los autores de los respectivos libros, los generos literarios y la asociacion libro-genero; cada uno en una lista por separado. Luego, se definen funciones que están destinadas a agregar información al catálogo. Para agregar un libro esta la función addBook, que recibe por parámetro el catálogo y el libro. Para agregar un autor a la lista de autores está la función addBookAuthor, que recibe por parámetro el catalogo, el nombre del autor y el libro. Para agregar un tag a la lista de tags está la función addTags, que recibe por parámetro el catalogo. De manera similar, está la funcion addBookTags. Por otro lado, están las funciones de creación de datos. La función newAuthor crea la estructura para modelar los libros de un autor y su promedio de ratings. Para esto, se crea un diccionario con las llaves "name", "books", y "average\_rating" con sus respectivos valores que varian dependiendo de lo recibido en los parametros de la función. De manera similar están estructuradas las funciones newTag y newBookTag, solo que con unas llaves de acuerdo a la información que se quiere guardar

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

Las funciones que comunican al view.py y al model.py se dan por medio de el controller.py. El view.py le pide informacion a el controller.py basandose en los comandos ingresados por el usuario y de este modo el controller.py le pide la informacion al model.py. Las funciones conectando a los dos son la del menu principal, initCatalog(), loadData(), loadBooks(), loadTags(), loadBookTags(), sortBooks(), getBooksByAuthor(), getBestBooks(), countBooksByTag().

1. ¿Cómo se crea una lista?

Una lista se crea a partir de varios argumentos dados por el usuario como el tipo de estructura de datos, una funcion para comprara los elementos, un identificador para comprar dos elementos de una lista, el nombre del rchivo con el cual se creara la nueva lista, y un delimitador para separar los valores (por defectos es una coma). Si no se tienen todos estos elementos, sube el error para que este se vea en pantalla.

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

cmp function es una función que sirve para comparar los elementos de la lista. Dependiendo de si el objeto es mayor, menor o igual devuelve -1, 1 o 0. Sirve para el ordenamiento de la lista por medio de esa comparación.

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

La funcoin addLast() adiciona un elemento en la ultima posicion de la lista y esta misma actualiza el apuntador a la ultima posicion. Adicioinalmente incrementa el tamaño de la lista por 1 puesto que se añadio un nuevo elemento. Los argumentos que esta funcion necesita son, la lista en la cual se queire añadir el elemento y el elemnto que se quiere insertar.

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

Esta funcion recorre la lista hasta llegar a la posicion del elemento, la cual debia ser mayor que cero y menor o igual al tamaño de la lista. Esta funcion despues retorna el elemento de la lista sin eliminarlo.

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

subList crea una una nueva lista con un conjunto menor de elementos a partir de la lista que recibe por parámetro, esta nueva lista toma los elementos desde la posición dada por el parámetro "pos" y con un tamaño dado por el parámetro "numelem" que determina el numero de elementos que van a entrar en la nueva sublista.

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

Si, el unico cambio notable en el funcionamiento del programa fue en tiempo de ejecucion de la funcion que carga la informacion en el catalogo. Al tomar el tiempo, el ‘ARRAY\_LIST’ se demoro 6 minutos y 56 segundos, mientras que el ‘SINGLE\_LINKED’ se demoro 6 minutos y 03 segundos.