OBSERVACIONES DEL LA PRACTICA

Estudiante 1 Cod 201818326

Estudiante 2 Cod 201729994

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

Los mecanismos de interacción son:

* Seleccionar una opción del menú principal. Entre las opciones están: Cargar información en el catálogo, Consultar los Top x libros por promedio, Consultar los libros de un autor, consultar libros por género, y salir.
* En la opción ‘Buscar los Top x libros por promedio’ tenemos un input adicional para escoger la cantidad de libros en el top x.
* En la opción ‘Consultar los libros de un autor’ tenemos un input adicional para ingresar el nombre del autor.
* En la opción ‘Libros por género’ tenemos un input adicional para el género a buscar.

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

* Todos los datos se almacenan en un catálogo creado en **newCatalog()**, después de que se ejecuta esta función se agrega información mediante **load(books, tags, booktags)** las cuales leen el archivo CSV y llaman a **add(books, tags, booktags)** para añadirlo al catálogo.

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

* View y model se comunican mediante las funciones en controller, cuando el usuario inicia cualquier opción del menú principal este llama a las funciones de controller para que se complete el proceso. Por ejemplo, en la opción 1 se utiliza **loadData()** en controller para iniciar el proceso de crear el catálogo.

1. ¿Cómo se crea una lista?

* Una lista se crea mediante **newList(…)** en donde se explica: la estructura, la función de comparación, la llave, el nombre y un separador. Esta información se lleva a **lt.newlist(…)** donde dependiendo de la estructura que se elija se va a **alt** o **slt** para crear la lista con sus parámetros específicos.

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

* Este parámetro se utiliza para definir una función que compare los elementos de la lista facilitando el orden de esta, en el caso que no se defina nada se utiliza None.

1. ¿Qué hace la funció **addLast()**?

* Esta función utiliza un sistema parecido a **newList()** en donde el list.py se agrega la información, en liststructure.py se define que tipo de lista es y según eso se va a **alt** o **slt** para añadir el elemento sea por append o por un nuevo nodo.

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

* Esta función tiene como parámetros de entrada un entero **pos** que va a ser la posición de el elemento que se quiere extraer de la lista **lst**. Usa la librería liststructure para recorrer la lista hasta el elemento en la posición **pos** de la lista **lst** y lo retorna sin eliminarlo**.**
* Si la lista es vacía o **pos** es menor que cero o mayor que el tamaño de la lista crea un **Exception**.

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

* Esta función tiene como parámetros un lista **lst**, y dos enteros **pos** y **numelem** . Lo que hace esta función es retornar una nueva lista que contiene los elementos de la lista original lst desde la posición pos y con una cantidad numelem de elementos. Si pos es menor a cero o mayor al tamaño de lst se crea una excepción o si la cantidad pos+numelem es negativa o mayor al tamaño de lst.

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

* Los resultados en consola fueron idénticos al realizar la prueba de funcionamiento, por lo que no se observó ningún cambio en el comportamiento del programa. Lo que se cambió fue la estructura de datos en la que se almacenan los tags y los book\_tags. Lo que podría cambiar es la forma en la que se almacenan estos datos en la memoria interna del computador, en términos de funcionalidades, deben ser iguales pues son el mismo tipo abstracto de dato (lineal). Como solo se miraron funcionalidades, el usuario no percibe los cambios entre un array list y una lista single linked.