OBSERVACIONES DEL LA PRACTICA

Estudiante 1 Cod 202011842

Estudiante 2 Cod 202021047

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

* En primer lugar, el view.py maneja un Output que es un menú de opciones (cargar información en el catálogo, consultar los Top x libros por promedio, consultar los libros de un autor, libros por género, salir) que le muestra al usuario con el fin de saber que operación se debe de realizar.
* En segundo lugar, están los resultados (Output) que el view.py muestra de acuerdo con la opción que el usuario seleccionó:

Si el usuario escogió la opción 1- cargar información en el catálogo, entonces el view.py mostrará los libros, autores, géneros, asociación de géneros a libros cargados.

Si el usuario escogió la opción 2- consultar los Top x libros por promedio se cargarán la cantidad de libros según el Top seleccionado por el usuario.

Si el usuario escogió la opción 3- consultar los libros de un autor se mostrarán todos los libros que se encontraron del autor indicado por el usuario.

Si el usuario escogió la opción 4- libros por género, se mostrarán todos los libros que se encontraron según la etiqueta escrita por el usuario.

Si el usuario escogió la opción 5-salir, se dejará de cargar las funciones.

* En tercer lugar, están los input que el view.py maneja, estos se encuentran las funciones de consultar los Top x libros por promedio, consultar los libros de un autor y libros por género, en los cuales se requiere una indicación del usuario para procesar las funciones siguientes.

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

En primer lugar, se crea el diccionario “Catalog” el cual contendrá listas vacías: “books”, “authors”, “tags”, “book\_tags”, luego a partir del uso de funciones como addtag, addBookAuthor se agregan los elementos a cada lista correspondiente del diccionario, estas funciones implementan una comparación con el fin de analizar si un elemento ya está en la lista, y cada vez que se agrega un nuevo elemento este se agrega en la última posición de su lista. Cabe recalcar que para usar las funciones anteriormente mencionadas previamente se crean los elementos a agregar a través de funciones como newBookTag, newTag, etc., los cuales se encargan de crear las listas o diccionarios correspondientes a agregar.

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

Dado que el view.py se comunica primero con el controller.py y este luego se comunica con el model.py hay que tener en cuenta las funciones en las que el view.py llama al comtroller.py, estas funciones son: initCatalog, loadData y el menú principal.

1. ¿Cómo se crea una lista?

Una lista se crea con la función newList, en la cual se debe de especificar la estructura de los datos (ARRAY\_LIST O SINGLE\_LINKED) y el cmpfunction para comparar los elementos de la lista.

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

El parámetro cmpfunction permite comparar los elementos de la lista a través de uso de otras funciones previamente creadas, si no se provee este se debe de dar un valor para Key, con el fin de que se use el comparador por defecto.

1. ¿Qué hace la funció **addLast()**?

Esta función agrega un elemento en la última posición de la lista, los parámetros son lst (lista a examinar) y element (elemento a insertar).

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

La función getElement retorna el elemento de una lista en una posición específica, los parámetros son lst (lista a examinar) y pos (posición del elemento a retornar).

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

La función subList() retorna una sublista de la lista principal, los argumentos son: lst (la lista principal), pos (la posición a partir de la cual se obtendrá la sublista) y numelem (número de elementos a copiar).

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