Observaciones-lab3

Método MVC:

1. Mecanismos de interacción con el usuario:

Al iniciar el programa, se imprime un menú **printMenu()** con 5 opciones, pidiendo al usuario un input de qué opción desea utilizar:

1. Si el usuario ingresa la opción 1: crea una lista vacía **initCatalog()**, y luego le carga la información de los datos ingresados **loadCatalog()**. Aquí no se solicita más información, y retorna la cantidad de libros, autores, géneros, y asociación de géneros a libros, en caso de haber sido exitoso. Al finalizar vuelve a imprimir el menú y pide el input de opción inicial.
2. Al ingresar la opción 2: se le pide al usuario ingresar cuantos libros quiere obtener en la lista de los mejor valorados. Después de ingresar esto, retorna la información de todos los libros mejor valorados **getBestBooks()**, según el numero solicitado por el usuario. Si se ingresa algo distinto a un número, retorna un error. Al finalizar vuelve a imprimir el menú y pide el input de opción inicial.
3. Al ingresar la opción 3: se le pide al usuario que ingrese el nombre de algún autor de su interés. En caso de el nombre coincida con algún autor de la base de datos, retorna una lista con la información de todos los libros pertenecientes al autor **getBooksByAuthor()**. En caso de que no coincida, retorna un mensaje de “No se encontró el autor”. Al finalizar vuelve a imprimir el menú y pide el input de opción inicial.
4. Al ingresar la opción 4: se le pide al usuario que ingrese algún género de interés. Si encuentra el género, retorna el numero de libros encontrados que pertenecen a ese género **countBooksByTag()**. Al finalizar vuelve a imprimir el menú y pide el input de opción inicial.
5. Al ingresar la opción 0 o cualquier otro número: cierra la aplicación.
6. Almacenamiento de datos:

Primero se crea un diccionario compuesto de 4 listas vacías con la función **newCatalog()**.

Luego, carga cada una de las 4 listas con las funciones: **-loadBooks()**

**-loadTags()**

**-loadBooksTags()**

En la función **loadBooks()**, se carga la lista de libros y la lista de autores.

En la función **loadTags()**, se carga la lista de Tags

En la función **loadBooksTags()**, se carga la lista de books\_tags.

El tipo de estructura utilizado para cada lista dependerá del parámetro ARRAY\_LIST o SINGLE\_LINKED establecido en el código para cada lista dentro de la función **newCatalog()**.

1. Comunicación **model.py** y **view.py**:

El model y el view se comunican a través del **controller.py**.

Para crear el diccionario con listas vacías (catalogo), utiliza la función **initCatalog(),** llamando una función del model.

Para cargar los datos al catalogo vacío, utiliza la función **loadData()**, la cual a su vez utiliza 4 funciones dentro del controller. Cada una de estas funciones llama una función del model.

Para cada opción de consulta, del controller también tiene una función que llama a un método que pertenece a la clase del model.

Método ADT:

1. Creación de listas:

Para crear una lista vacía se utiliza la función **newList()**. Esta función crea el tipo de lista dependiendo del string ingresado en el parámetro **datastructure**, que puede ser SINGLE\_LINKED o ARRAY\_LIST, estos atributos se establecen dentro de la función de **initCatalog()** dentro del model.

1. Cmpfunction:

El parámetro **cmpfunction** también se utiliza a la hora de crear una nueva lista vacía. Este indica el tipo de comparación que se va a realizar a la hora de ordenar una lista. La función de comparación default compara el tamaño de dos ints. Si se quiere utilizar otro tipo de comparación, por ejemplo comparar si dos autores son el mismo, se necesita definir una función en el model.

1. addLast():

Se agrega un elemento al final de la lista. El tamaño de la lista aumenta por 1.

1. getElement():

A partir de una posición dada, retorna el elemento de la lista en dicha posición.

1. subList():

A partir de una posición dada, y una longitud establecida, retorna una copia de la lista con la longitud dada comenzando a partir de la posición ingresada.

Cambios tipo de lista:

No vimos ningún cambio, ni en el tiempo de implementación, ni en los resultados de la función.