**Observaciones**

**Integrantes:**

**Alejandro Herrera Jiménez – 201915788**

**Hugo David Flórez Rodríguez - 201912429**

**PASO 3:**

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

Inputs:

La línea roja indica el input que hace el usuario, es decir, el valor que ingresa al programa.

Las líneas azules son las comparaciones usadas por el programa para saber el significado del input ingresado por el usuario.

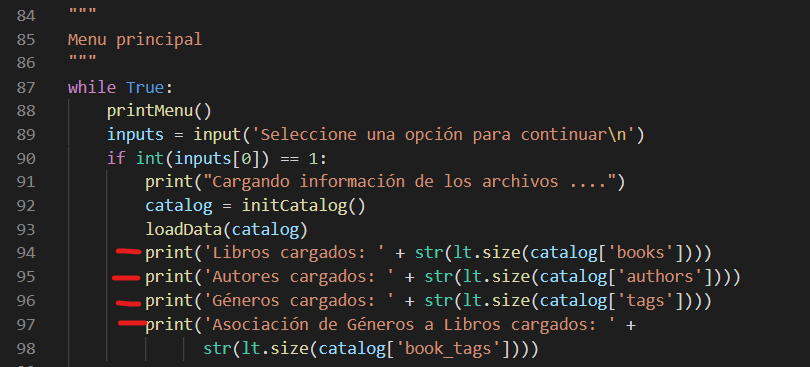
Texto

Descripción generada automáticamente

Outputs:

Las líneas rojas indican los outputs finales, es decir, los datos solicitados por el usuario con el input.

Primera función (cargar datos):



Segunda función (Best Books):

Texto

Descripción generada automáticamente

Tercera Función (Books by Author):

Texto

Descripción generada automáticamente

Cuarta función (Books by Tag):

Texto

Descripción generada automáticamente

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

**Crear catalogo:**

1. La función **newCatalog()** crea una lista vacía para los libros, una lista vacía de tipo 'ARRAY\_LIST' para los autores la cual ordena con la función **compareauthors**, otra lista vacía de tipo 'ARRAY\_LIST' para los tags la cual ordena con la función **comparetagnames** y finalmente otra lista vacía de tipo 'ARRAY\_LIST' pero esta vez sin ningún tipo de ordenamiento para los booktags.

**Cargar libros y autores:**

1. Se hace uso de la función **addBook(catalog, book)**, la cual tiene como parámetros el catálogo inicialmente vacía creado en el paso 1 y el parámetro book hace referencia a cada una de las líneas del archivo books.csv. Primero, la función **lt.addLast** añade el nombre del libro a la lista de libros en el catálogo. Segundo, se crea una lista con el(los) autor(es) del libro.
2. Se hace uso de **addBookAuthor(catalog, authorname, book),** los parámetros son el catálogo, el nombre del autor y el mismo libro del paso pasado. La función llama a la lista de autores del catálogo, que inicialmente está vacía. Luego, se guarda en una variable la posición del autor en dicha lista, si no está devuelve 0. En caso de estar toma el elemento (el autor) y si no está lo agrega con la función **newAuthor(name)** que recibe como parámetro el “authorname” inicial y lo agrega al final de la lista de autores en el catálogo. Después, se añade el nombre del libro en la lista de libros del autor (cada autor en la lista de autores del catalogo inicial tiene el nombre del autor y una lista relacionada con el nombre de todos los libros de dicho autor).

**Cargar Tags:**

1. **addTag(catalog, tag)** la función recibe como parámetro el catálogo creado al principio y el tag que se va a agregar. Luego, llama a la función **newTag(name, id)** con los parámetros nombre del tag y el id con el que está relacionado, la cual crea una estructura que tiene un valor relacionado a tag['name'] que es el nombre del tag y un valor relacionado a tag['tag\_id'] que es el id de cada tag. Luego se añade esa estructura de datos la cual se le pone como nombre tag y se añade al final de la lista de tags del catálogo.

**Cargar BookTags:**

1. **addBookTag(catalog, booktag)** la función recibe como parámetro el catalogo y un booktag. Luego llama a la función **newBookTag(tag\_id, book\_id)**  cuyos parámetros son el tag de un id y el id de un libro y lo relaciona en una estructura de datos con los nombres de 'tag\_id' y 'book\_id' respectivamente. Finalmente, lo agrega a la lista booktags de catálogo.

**Ordenamiento:**

1. Finalmente se usa la función **sortBooks(catalog)** que recibe el catalogo inicial y lo que hace es ordenar los libros teniendo en cuenta el rating de cada libro, esto los hace con la estructura de datos Shellsort.

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

**Todas estas funciones se encuentran en el archivo controller.py.**

**initCatalog():** Llama a la función newCatalog() ubicada en el modelo.

**loadData(catalog):** Carga los datos de los archivos .csv por medio de las funciones loadBooks(catalog), loadTags(catalog), loadBooksTags(catalog) y sortBooks(catalog), las cuales serán explicadas en breve y también se encuentran en controller.py. Por medio de dichas funciones se termina de comunicar con el model.py.

**loadBooks(catalog):** Carga los libros del archivo y los envía a modell.py por medio de la función addBook(catalog, book).

**loadTags(catalog):** Carga los tags del archivo y los envía a modell.py por medio de la función addTag(catalog, tag).

**loadBooksTags(catalog):** Carga los booktags del archivo y los envía a modell.py por medio de la función addBookTag(catalog, booktag).

**sortBooks(catalog):** Usa la función sortBooks(catalog) ubicada en modell.py para ordenar los libros por average\_rating.

**getBooksByAuthor(catalog, authorname**): Usa la función getBooksByAuthor(catalog, authorname) ubicada en modell.py y retorna los libros de un autor.

**getBestBooks(catalog, number):** Usa la función getBestBooks(catalog, number) ubicada en modell.py y retorna los mejores libros.

**countBooksByTag(catalog, tag):** Usa la función countBooksByTag(catalog, tag) ubicada en modell.py y retorna los libros que fueron etiquetados con el tag.

**PASO 4:**

**• ¿Cómo se crea una lista?**

Se crea haciendo uso de la función y parámetros especificados a continuación newList(datastructure='SINGLE\_LINKED', cmpfunction=None, key=None, filename=None, delimiter=","). Esta función se encuentra en el list.py

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

El parámetro cmpfunction es la función de comparación que se va a usar para ordenar la lista, esta función se debe especificar y escribir aparte en el código.

**• ¿Qué hace la función addLast()?**

Esta función añade el elemento deseado en la lista especificada, los parámetros se escriben de la siguiente forma (lista, elemento).

**• ¿Qué hace la función getElement()?**

Recibe como parámetros los siguientes datos en el mismo orden (lista, posición), lo que hace la función es retornar el elemento que se encuentre en la posición dada.

**• ¿Qué hace la función subList()?**

Recibe como parámetros los siguientes datos en el mismo orden (lista, posición, número de elementos), lo que hace es crear una sublista con los elementos de la lista dada iniciando en la posición especificada y con una longitud del número dado.

**PASO 5:**

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

Se observó que el tiempo de funcionamiento incrementó cuando se usó el parámetro “SINGLE\_LINKED” en comparación al uso del “ARRAY\_LIST”. Con el primero fue de aproximadamente 5 minutos y 36 segundos mientras que con el segundo fue de aproximadamente 5 minutos y 17 segundos.