OBSERVACIONES DEL LA PRACTICA

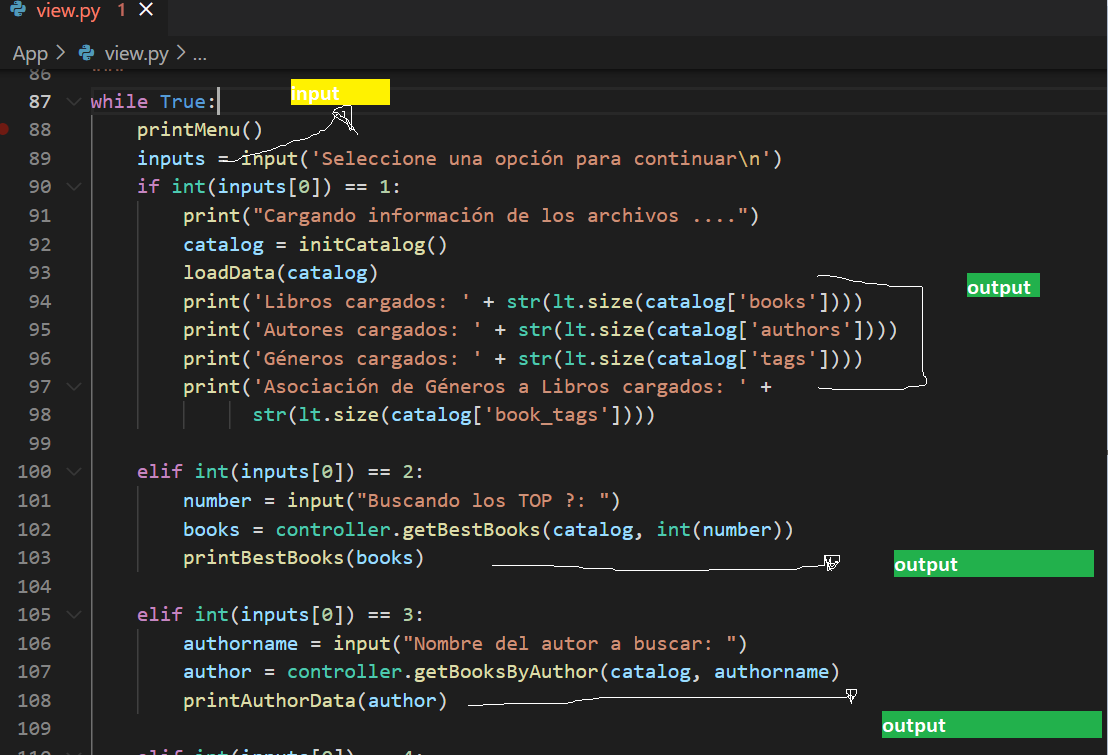
Estudiante 1 Ana Sofía Villa Benavides Cod 201923361

Estudiante 2 Daniela Alejandra Camacho Molano Cod 202110974

**NOTA**: Las preguntas las respondimos utilizando los archivos originales -small.csv ya que ambos de nuestros computadores presentaron problemas al intentar utilizar los archivos normales.

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

View.py tiene como mecanismo de interacción un menú. Inicialmente se imprimen las opciones del menú con la función printMenu(). Luego se tiene un input donde se le pide al usuario que seleccione una opción para continuar. Posteriormente a partir de if y elifs se ejecuta la función requerida y se imprimen los resultados de esta como output.



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

En primer lugar, en model.py se importa una lista TAD y a partir de esta se define una función en la que se crea un catálogo vacío para almacenar la información. Este catalogo lo dividen en cuatro listas vacías distintas: una para autores, otra para libros, otra para el género y otra que asocia libro y género.

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

No se encuentran funciones que se comuniquen directamente entre view y model, esto se debe posiblemente a la arquitectura MVC implementada. Por este motivo en view encontramos funciones que se comunican con controller (del modo *controller.funcion()*) y a su vez en controller funciones que se comunican con model (del modo *modelr.funcion()*).

1. ¿Cómo se crea una lista?

Las listas se pueden crear de las siguientes maneras:

- Se indica el nombre que va a adquirir la lista, se pone “:” y luego se ponen los elementos que esta va a tener. En caso de ser vacío se escribe “None” (ejemplo: “Book”: None)

- Se usa el siguiente código: lt.newlist(), dentro de los paréntesis se indica lo que esta lista va a tener como elementos

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

En newlist el parámetro cmpfunction es donde se indica la función utilizada para comparar los elementos de la lista. En este caso es = None por lo tanto se usa la función por defecto.

1. ¿Qué hace la funció **addLast()**?

Este adiciona el elemento indicado en la última posición de la lista. Este elemento puede ser de cualquier tipo (lista, variable, diccionario,etc)

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

La función recibe una lista y una posición como argumentos y devuelve o retorna el elemento de la lista que se encuentra en la posición indicada. La manera en la que funciona es recorriendo la lista hasta llegar al elemento de la posición que se requiere.

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

Esta tiene como tiene input una lista a examinar, una posición de la lista (desde esta posición la función examinara la lista), y el número de elementos que se quieren extraer. Como output retorna otra lista más pequeña con los parámetros dados en el input, esto quiere decir que genera una copia de la lista original, por lo que esta no es modificada.

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

Como mencionamos ambos de nuestro computadores tuvieron problemas al intentar utilizar los archivos normales (sin -small). Utilizando los archivos small, y haciendo el cambio en tags y books\_tags a SINGLE\_LINKES al correr el programa no percibimos ningun cambio significativo en su implementación. No surgieron erroes y tampoco hubo una diferencia en tiempo. Por otro lado, los cambios implementados se realizaron dentro del TAD, lo que significa una ventaja debido a que se puede utilizar array list o single linked según la necesidad y complejidad que el problema presente. Esto se puede determinar tomando en cuenta sus ventajas y desventajas. Por ejemplo: al utilizar la lista encadenada se utiliza más memoria y su complejidad de implementación es mayor; sin embargo, la memoria es utilizada de manera más eficiente ya que cada uno de sus elementos tiene una referencia al siguiente elemento. Asimismo, al utilizar el arreglo se puede acceder a cada elemento por medio de su posición y utiliza menos memoria; no obstante, para eliminar o insertar un nuevo elemento se necesita hacer un recorrido de toda la lista y modificar las posiciones de los elementos.