OBSERVACIONES DEL LA PRACTICA

Obed Cabanzo Cod **201911749** jo.cabanzo @uniandes.edu.co

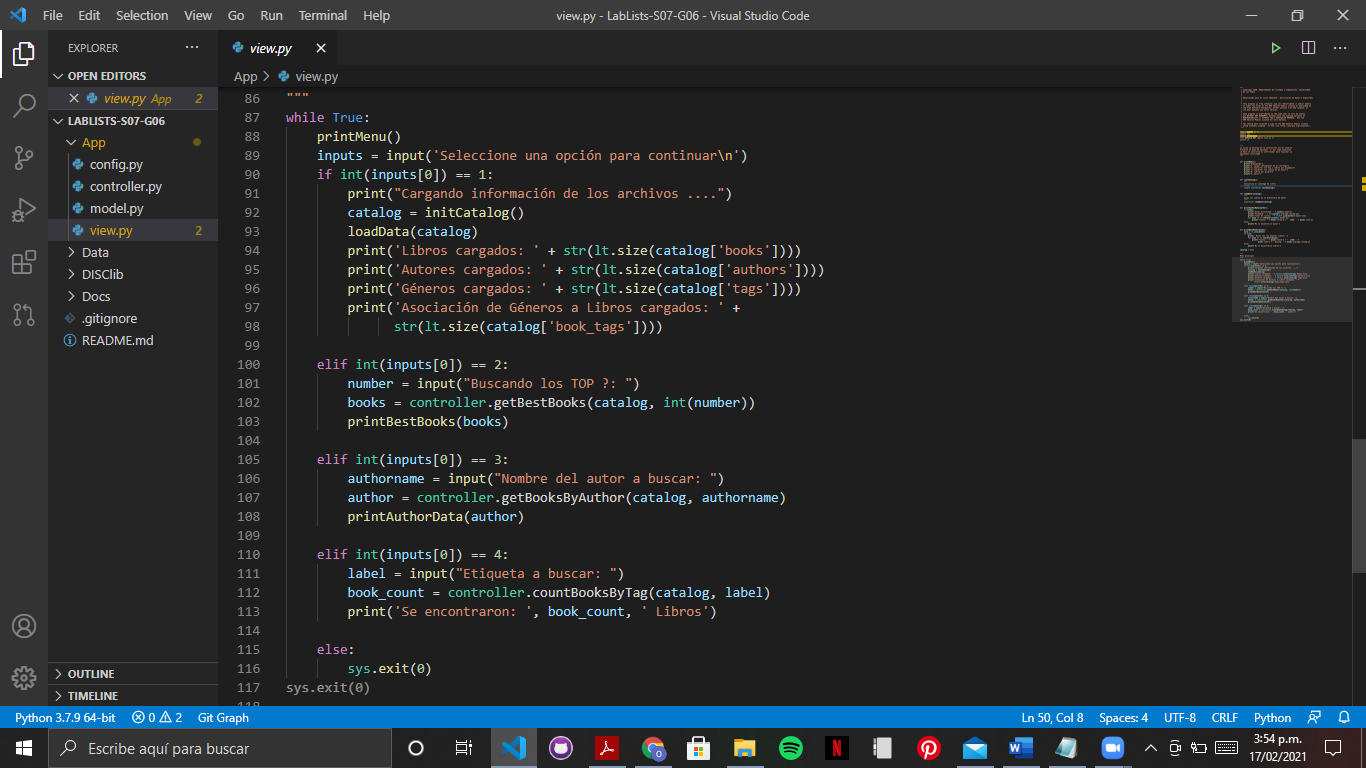
Alejandra Melo Cod **202021526** a.melo4@uniandes.edu.co

1. ¿Cuáles son los mecanismos de interacción (I/O: Input/Output) que tiene el **view.py** con el usuario?
2. ¿Cómo se almacenan los datos de **GoodReads** en el **model.py**?
3. ¿Cuáles son las funciones que comunican el el **view.py** y el **model.py**?
4. ¿Cómo se crea una lista?
5. ¿Qué hace el parámetro **cmpfunction=None** en la función **newList()**?
6. ¿Qué hace la funció **addLast()**?
7. ¿Qué hace la función **getElement()**?
8. ¿Qué hace la función **subList()**?
9. ¿Observó algún cambio en el comportamiento del programa al cambiar la implementación del parámetro **“ARRAY\_LIST”** a **“SINGLE\_LINKED”**?

DESARROLLO

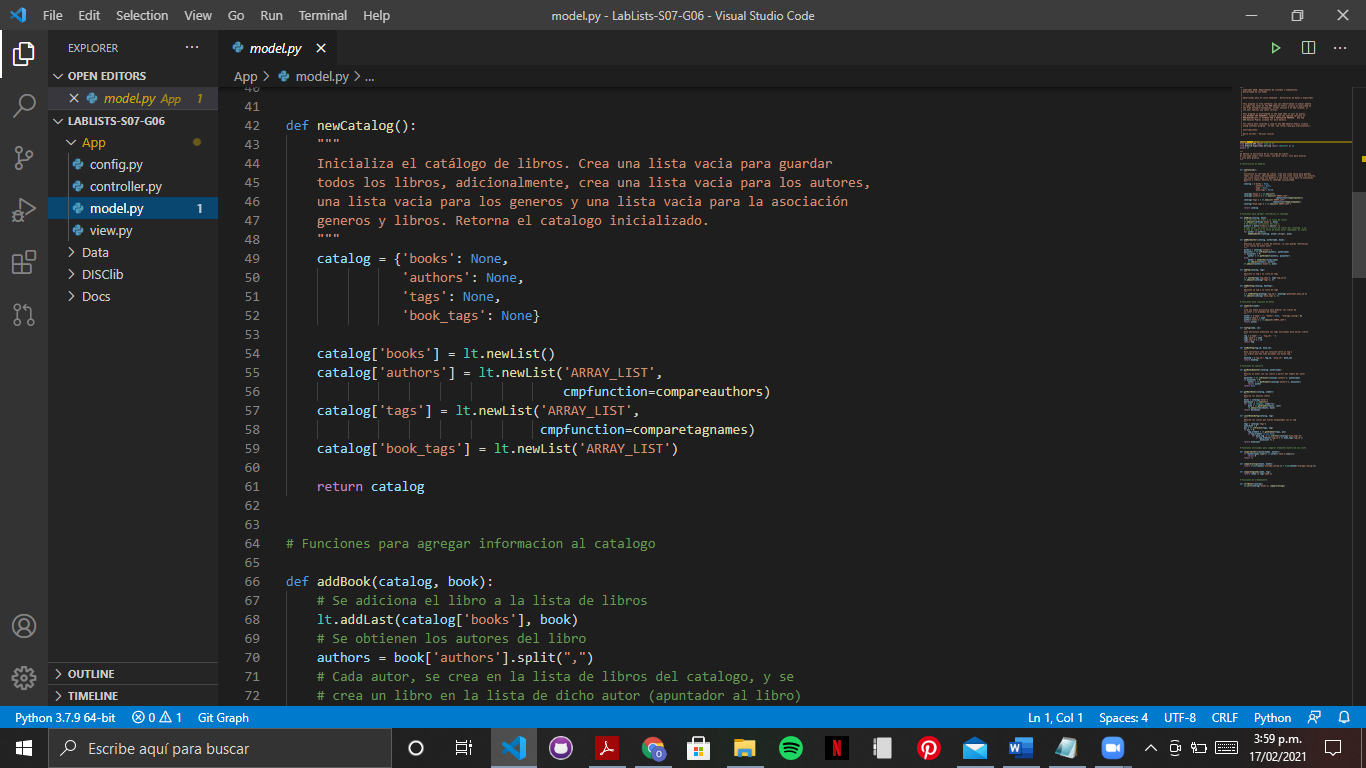
Los métodos de interacción que tiene el **view.py** son los siguientes:

I**nput/Output:**



En los que por medio del parametro de numeros enteros se mantiene la interaccion con el usuario y evidentemente se imprimen en consola los resultados de la busqueda que se solicite.

En primer lugar, crea un catálogo para almacenar todos los datos que se van a cargar. Catálogo que contiene todos los datos del archivo y los guarda en listase tipo **ARRAYLIST**.



Ademas de ello tiene todos los metodos que permitiran añadir elementos al catalogo

def addBook(catalog, book):

def addBookAuthor(catalog, authorname, book):

def addTag(catalog, tag):

def addBookTag(catalog, booktag):

Las funciones que comunican al view.py y el model.py son los siguientes:

def loadData(catalog):

Este metodo del **controller.py** conecta el metodo **loadData** del view con el model. Y se encarga de cargar los datos que posteriormente añadira con los metodos que agregan elementos a las listas del catalogo que tiene el model como: **loadBooks, loadBooksTags, loadTags.** Entre otros.

Para crear una lista se puede utilizar la función newList de la librería DISClib. 

Cmpfunction es una función de comparación para los elementos de la lista y se utiliza en vez de utilizar el key que tiene el identificador de comparación por defecto.

La función addLast permite agregar un elemento al final de una lista. Además, al utilizar esa función se debe actualizar el apuntador para que señale la última posición y se aumenta el tamaño de la lista en 1. 

La función getElement recibe una lista y una posición y retorna el elemento de la posición indicada. Para hacer esto la función recorre la lista hasta llegar al elemento en la posición y lo devuelve. No obstante, la lista no debe estar vacía y debe tener por lo menos un tamaño igual al valor de posición para poder utilizar la función. 

La función subList recibe una lista, una posición y un número de elementos (longitud). A partir de esto crea y retorna una lista con una copia de los elementos a partir de la posición y longitud indicadas. 

Al ejecutar nuevamente las funciones del terminal no se vieron muchos cambios, el rendimiento de los algoritmos era similar y no vimos cambios.

