**OBSERVACIONES DE LA PRACTICA**

Nombre Estudiante 1 Cod XXXX

Mateo Zambrano Burbano Cod 202321531

Nicolas Mario Romero Colmenares 202321309

Juan Eduardo Briceño Toro 202320095

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

Inputs: Son los datos o información que el usuario introduce en un programa para que este funcione. Estos datos pueden ser de diversos tipos, como:

Texto: Nombres, palabras, frases, etc.

Números: Enteros, decimales, etc.

Fechas: Días, meses, años, etc.

Opciones: Seleccionar entre diferentes posibilidades, como "Sí" o "No".

Archivos: Imágenes, vídeos, documentos, etc.

Outputs: Son las respuestas o resultados que el programa proporciona al usuario después de procesar los inputs. Los outputs también pueden ser de diversos tipos, como:

Mensajes en pantalla: Información textual que se muestra al usuario.

Gráficos: Visualizaciones de datos en forma de barras, líneas, etc.

Sonidos: Reproducción de audio.

Acciones: Control de dispositivos o sistemas externos.

Archivos: Creación o modificación de archivos.

En resumen, los inputs son la "materia prima" que el usuario proporciona al programa, y los outputs son el "producto final" que el programa entrega al usuario.

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

Los datos de los distintos CSVs son almacenados con la función newCatalog(): dentro de esta función se hacen sus respectivas asignaciones en un diccionario con las distintas llaves para asignarles los valores, se usan ARRAY\_LIST o SINGLE\_LINKED según necesite el programa.

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

Las funciones son el newController() es el que recibe el resultado de la función newCatalog(): y luego retornarlo al view.py mediante la misma función newController()

1. ¿Cuál es la función que permite crear una lista?, ¿Qué datos son necesarios?

La función que permite crear listas en el programa es Lt.newList() ubicada en el List.py. Esta es convocada en newCatalog(), newAuthor() y getBestBooks() ubicadas en el model del programa. Para crear una lista este requiere de unos parametros: datastructure, cmpfunction, filename, key y delimeter

1. ¿Para qué sirve el parámetro **datastructure** en la función **newList()**?, ¿Cuáles son los posibles valores para este parámetro?

SINGLES\_LINKED, DOUBLE\_LINKED y ARRAY\_LIST

Tipo de estructura de datos a utilizar para implementar la lista. Los tipos posibles pueden ser: ARRAY\_LIST, SINGLE\_LINKED y DOUBLE\_LINKED.

1. ¿Para qué sirve el parámetro **cmpfunction** en la función **newList()**?

El parámetro cmdstructure dentro de la función de newList() permite hacer que la función compare los elementos que están dentro de la lista en donde si no hay una función de comparación, se debe recorrer la función por defecto y la llave del parámetro debe de tener un valor, de lo contrario, la llave del parámetro debe decir None y se recorrerá una función de comparación.

1. ¿Qué hace la función addLast()?

Lo que hace la función addLast() es añadir un elemento en el último espacio de la lista, para esto se requiere la lista que se va a actualizar, y el elemento que se busca añadir. En la función se realiza un except con el objetivo de que los datos de los parámetros sean correctos, en caso de que sean así, el parámetro de elemento será agregado a la lista, para que después se aumente el tamaño en 1 y se retorne la lista actualizada

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

Lo que hace la funcion getElement() es buscar un elemento especifico dentro de la lista, para esto se requiere la lista y la posición del elemento que se solicita (que debe ser entre cero y el tamaño de la lista). Dentro de la función se hace un except con el objetivo de que los datos estén bien ingresados, en caso de que sean así, la función recorrerá cada elemento de la lista para buscar el elemento que está en la posición dada como parámetro y así retornar lo que hay dentro de esa posición en la lista.

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

La función subList() crea una lista mas pequeña a partir de la que se da por parámetro, adicionalmente, los otros parámetros son pos, que sirve como punto de inicio de la sublista y numelem, que cuenta la cantidad de elementos que incluirá en la sublista a desde el punto de inicio. Dentro de la función se hace un except con el objetivo de que los datos estén bien ingresados, en caso de que sea así, se creara una nueva lista y se irán añadiendo elementos de la lista original desde el punto pos hasta que la cantidad de numelem haya sido completado. Finalmente se retornará la sublista creada.

1. Revise el uso de la función **iterator()** en las funciones **printAuthorData(author)** y **printBestBooks(books)** en la Vista que aplican a una lista de libros. ¿Qué hace la función **iterator()**?

La función iterator() de la libreria DISClib.ADT, en general, crea un iterador que recorra una estructura de datos abstracta (ADT), es decir ARRAY\_LIST, DOUBLE LINKED, SINGLE\_LINKED, pilas o colas y particularmente en la primera función printAuthorData(author), el iterator() toma los valores de la lista “authors” para proporcionar el acceso a estos solo si el nombre del autor es encontrado. En el caso de printBestBooks(books) ocurre lo mismo, pero con la lista “books”.

1. ¿Observó algún cambio en el comportamiento del programa al cambiar el valor del parámetro **'datastructure'** en la creación de las listas?

El rendimiento del programa se vio muy afectado, demorándose minutos para realizar la carga de datos, por lo que se concluye que los tipos de datos que están cargando deben leerse de manera distinta por sus diferentes características, así que requieren de un tipo de lista diferente para manejarse de manera óptima y rápida.