OBSERVACIONES DEL LA PRACTICA

Thais Tamaio Cod 202022213

Juan José Osorio Cod 202021720

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

View.py tiene como mecanismos de interacción las solicitudes del usuario (el input) y el output son las opciones del menú y también la información solicitada por el usuario que se le muestra a través del view.py.

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

Los datos de GoodReads se almacenan en model.py por medio del la estructura de datos con orden lineal llamado “arreglo”, la cual es retornada por la función “newCatalog()”.

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

Las funciones que comunican al view.py y el model.py se encuentran en el archivo controller.py y se llaman:

* initCatalog()
* loadData()
* getBestBooks()
* getBooksByAuthor()
* countBooksByTag()

1. ¿Cómo se crea una lista?

En el archivo list.py, se crea una lista nueva de tipo TAD List con la función newList(). Esta función crea una lista vacía.

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

El parámetro cmpfunction=None en la función newList() provee una función que permite comparar los elementos de la lista que se crea por medio de la función elegida. Tiene como valor por defecto None, es decir, ninguna función.

1. ¿Qué hace la funció **addLast()**?

La función addLast() agrega un elemento en la útlima posición de la lista. Tanto la lista como el elemento deben entrar como parámetros de la función.

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

La función getElement() retorna un elemento en una posición específica de un lista. Tanto la lista como la posición son parámetros que necesita la función para retornar el elemento.

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

La función subList() retorna una sublista que contiene ciertos elementos que son determinados por una posición y longitud de elementos que entran como parámetros. Retorna la sublista.

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

Para responder esta pregunta utilizamos la librería “time” para medir el tiempo en el que se demoraba en procesar la primera opción. Con la estructura de orden lineal “ARRAY\_LIST” tardó 336.716 segundos mientras que con “SINGLE\_LINKED” tardó 356.263 segundos. Por lo tanto, el cambio que observamos al al cambiar la estructura de datos se refleja en el tiempo debido a que al utilizar “SINGLE\_LINKED” se demoró 20 segundos más. Teniendo en cuenta estos resultados, se puede concluir que utilizar un “ARRAY\_LIST”puede llegar a ser mucho más eficiente (en casos similares al del laboratorio) cuando se trata un gran número de datos, es por esto que entre más grande sea la cantidad de datos, mayor será la diferencia de tiempo entre una lista de tipo “ARRAY\_LIST” y una de “SINGLE\_LINKED”.