**Santiago Rodriguez – Andrés Felipe Vargas Grupo: 09**

**Observaciones Lab 3**

**PASO 3**

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

Los mecanismos output del archivo view.py son todos aquellos que se despliegan en la interfaz del usuario que le permiten interactuar con el código, estos en su mayoría son todos los prints que le permiten al usuario conocer lo que hace cada opción y como realizarla basado en lo que busca y al generar una respuesta le explica detalladamente el resultad obtenido. Por otro lado, los inputs que recibe el programa son todas estas variables que el usuario introduce en la interfaz como lo son el numero de una opción que desea saber y posteriormente cualquier tipo de información adicional que requiera el programa para dar una respuesta basado en lo que el usuario solicito. Una vez el usuario selecciona una opción esta se ejecuta en el código sin que este sepa el proceso que hay poder detrás y posteriormente da una respuesta, un output basado en lo que solicito.

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

Los datos de GoodReads se almacena en una variable llamada ‘‘Catalog’’ la cual es un diccionario de listas. Dentro de este diccionario se almacenan 4 listas en las cuales se almacenan los siguientes datos: libros, autores, géneros y asociación de géneros. Estos datos se van cargando a medida que las funciones en model.py se van ejecutando y agregando datos a las listas.

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

Las funciones que comunica view.py y model.py se encuentran ubicadas en controller.py. Esto debido a que view.py ejecuta distintas funciones primero importando a controller y luego llamándolo con ‘‘controller. + la función deseada’’, una vez que view.py llama a controller a su vez controller llama a model.py con ‘‘model. + función deseada’’ y se ejecuta la función deseada. Por lo tanto view.py y model.py no se comunican entre si de manera directa, ya que en ninguno de estos archivos se encuentra importado en el otro (view.py no importa a model y model no importa a view.py). Usan a controller.py como intermediario y de esta manera view.py logra recibir la información generada en model.py.

**PASO 4**

**• ¿Cómo se crea una lista?**

En el ejemplo dado la lista se crea al ejecutar la función newList, en esta función se encuentran 5 argumentos o parámetros (datastructure, cmpfunction, key, filename, delimiter) que permiten crear una lista vacia. Sin embargo, de estos el más importante y el crea la lista en si, es el parametro de dataestructure el cual nos deja elegir la estructura de datos que vamos a usar.

**‘‘datastructure’’** = Permite elegir el tipo de lista que implementaremos puede ser ARRAY\_LIST y SINGLE\_LINKED.

A su vez estos parámetros permiten crear listas debido a que previamente se importó la librería de DISClib.DataStructures como lt.

**Sección de código donde se crea:**

Texto

Descripción generada automáticamente .

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

La función cmpfunction = None se encarga de realizar una comparación para los elementos de la lista. Si no se provee función de comparación se utiliza la función por defecto pero se debe proveer un valor para key. Si se provee una función de comparación el valor de Key debe ser None. De esta manera cmpfunction nos permite bien comparar 2 listas o 2 elementos de la lista y dar una respuesta basado en los argumentos analizados cuyo valor puede ser 1, 0 or -1, dependiendo si el elemento es mayor,menor o igual.

**• ¿Qué hace la función addLast()?**

La función addLast se encarga de adicionar un elemento en la última posición de la lista creada anteriormente y se actualiza el apuntador a la útima posición, a su vez se incrementa el tamaño de la lista en 1.

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

La función getElement permite conocer el elemento que se encuentra ubicado en una lista y en la posición buscada. En este caso el numero que debe ser mayor que cero y menor o igual al tamaño de la lista y se encuentra representado por ‘‘pos’’. Por lo tanto al digitar un numero la función getElement recorrerá toda la lista hasta llegar a la posición indicada y mostrara el numero que se encuentra en esta posición sin eliminarlo de la lista.

**• ¿Qué hace la función subList()?**

La función subList permite crear por así decirlo una lista alterna que comprende los elementos ubicados en la posición ‘‘pos’’ y termina al crear una lista con el numero de elementos dada por la variable ‘‘numelem’’. Por lo tanto la función retorna una lista nueva con copia de los elementos comprendidos entre las 2 variables mencionadas anteriormente.

**PASO 5**

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

Los resultados dados no cambiaron, se mantuvo igual tanto los numero correspondientes a los géneros y su asociación, al igual que los top 5 y los libros de autores. Sin embargo al ejecutar la primera función que se encarga de cargar los archivos demoro mucho mas que cuando se ejecuto con array. Esto se debe a que las Single linked lists usan mas memoria ya que la referencia al siguiente nodo tambien se almacena con la información. Debido a esto el programa demoro mucho mas en correr, que al realizar la misma operación pero con Array List.