Universidad de los Andes
Estructuras de Datos y Algoritmos – Sección 6

José Nicolás Cárdenas (201922006)
Andrés Leonardo Beltrán (202014143)

Laboratorio 3: Observaciones

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

Hay tres funciones que generan interacción con el usuario:

1- *printMenu.*
2- *printAuthorData.*
3- *printBestBooks.*

Adicionalmente, el *While* relacionado a la impresión del menú y la escogencia de las opciones también genera una interacción con el usuario:

```
"""
Menu principal
"""
while True:
    printMenu()
    inputs = input('Seleccione una opción para continuar\n')
    if int(inputs[0]) == 1:
        print("Cargando información de los archivos ....")
        catalog = initCatalog()
        loadData(catalog)
        print('Libros cargados: ' + str(lt.size(catalog['books'])))
        print('Autores cargados: ' + str(lt.size(catalog['authors'])))
        print('Géneros cargados: ' + str(lt.size(catalog['tags'])))
        print('Asociación de Géneros a Libros cargados: ' +
              str(lt.size(catalog['book_tags'])))

    elif int(inputs[0]) == 2:
        number = input("Buscando los TOP ?: ")
        books = controller.getBestBooks(catalog, int(number))
        printBestBooks(books)

    elif int(inputs[0]) == 3:
        authorname = input("Nombre del autor a buscar: ")
        author = controller.getBooksByAuthor(catalog, authorname)
        printAuthorData(author)

    elif int(inputs[0]) == 4:
        label = input("Etiqueta a buscar: ")
        book_count = controller.countBooksByTag(catalog, label)
        print('Se encontraron: ', book_count, ' Libros')

    else:
        sys.exit(0)
sys.exit(0)
```

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

Los datos de *GoodReads* se almacenan en una estrucutura de datos que se genera mediante la función *newCatalog* de *Model.py*, la cual implementa el método *newList* de la clase *list*. Dicha estructura contiene la información de los libros, los autores y los tags. La función mencionada define el diccionario *catalog*, el cual será el que retorna la función. Las parejas llave-valor de este son cadenas de caracteres y arreglos vacíos (respectivamente); la única excepción es el elemento con identificador *authors*, ya que este no especifica si es una lista enlazada o un arreglo. Es importante notar que todas estas listas están vacías, ya que la información es cargada en el módulo *controller.py*.

3) ¿Cuáles son las funciones que comunican el el view.py y el model.py?

El proceso inicia con la función *newCatalog* de *controller.py,* ya que esta trae todas las estructuras de datos guardades en el módulo *model.py*. Adicionalmente, todas las funciones que inician con el prefijo *load* de *controller.py* son las encargadas de cargar a lista *catalog* toda la información guardada en la carpeta *Data*. Finalmente, el *view.py* se comunica con *controller.py* a través de las funciones *initCatalog* y *loadData*.

4) ¿Cómo se crea una lista?

Mediante la función *newList*. Este crea el TAD lista que se usará en el proyecto.

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

Este parámetro especifica la función que se usará para comparar los elementos que se van a añadir a la lista con un orden establecido.

6) ¿Qué hace la función addLast()?

Se encargar de agregar un último elemento al final de la lista.

7) ¿Qué hace la función getElement()?

Esta función permite obtener un elemento de la lista de una posición especificada.

8) ¿Qué hace la función subList()?

Retorna una sublista que hace parte de la lista original; es decir, esta crea una lista que contiene los elemenos de la lista que van desde la posición ingresada por parámetro y hasta *numelem* menos uno (-1).

9) ¿Observó algún cambio en el comportamiento del programa al cambiar la implementación del parámetro "ARRAY_LIST" a "SINGLE_LINKED"?

No, el comportamiento del programa siguió siendo el mismo, y todas las operaciones realizadas funcionaron de la misma manera. Aun así, pese a que no se pueda percibir directamente, haber cambiado a la listas encadenadas implicó mayor uso de espacio, ya que fue necesario añadir y guardar información relacionada a las referencias a cada nodo/elmento de la lista.