02/06/2022

**TRABAJO PRÁCTICO ESPECIAL**

**PROGRAMACIÓN 3**

Entrega N°1

TUDAI- EXACTAS - UNICEN

Autores:

* De Francesco German (*germandefrancesco@gmail.com*)
* Ballone Gabriel (*gabrielgeno@gmail.com*)

# Introducción

Se desea a implementar la lógica necesaria para obtener una colección de libros que contenga un género en particular, ingresado por el usuario. La herramienta comenzará llevando a memoria la colección completa de libros para luego realizar un filtrado por un género dado, presentando al usuario la colección de libros resultante. Para optimizar el proceso de búsqueda, se requiere implementar un índice por género, el cual simplificará el acceso a solo un subconjunto de todos los libros existentes.

Entrada: El programa tomará un archivo .csv de entrada, con un formato como el que se muestra a continuación.

Salida: El programa deberá generar un archivo .csv con los títulos de los libros que cumplen con el género dado.

Consignas

1. Discutir y analizar el costo de las operaciones sobre las distintas estructuras que se podrían utilizar para almacenar los libros en memoria.

* Justificar la elección de la estructura considerando la utilización de la misma dentro de la herramienta.

1. Considerando las siguientes estructuras como posibles índices de acceso por género:

* Una lista simplemente vinculada (implementada en el práctico 1).
* Alguna de las implementaciones conocidas de la interfaz List de Java.
* Un árbol binario de búsqueda.

1. Discutir y analizar el costo para realizar un filtrado utilizando cada una de las estructuras propuestas como índice de acceso.
2. Implementar el índice de acceso por géneros eligiendo una de las estructuras. Justificar su elección.
3. Realizar pruebas de ejecución utilizando los distintos .cvs provistos y analizar los resultados, considerando distintas métricas (tiempo de ejecución, cantidad de iteraciones, cantidad de nodos visitados, etc.).

# Desarrollo del trabajo

Se optó por utilizar una lista vinculada simple (LinkedList) tanto para el conjunto de libros como así también para el conjunto índice de géneros.

En el caso de los libros, dado que la consigna sólo solicita obtener todos los libros de un género, es decir, no hay que buscar por título o algo que requiera algún criterio de ordenamiento, la búsqueda siempre será lineal, de principio a fin (O(n)). Utilizando esta estructura estamos más que cubiertos y no tenemos que traer todos los libros a memoria (realizando un malgasto de ésta) como sería utilizando un ArrayList.

Además, la búsqueda no requiere recorrer la lista en forma reversa o acceder a un libro en particular, lo cual sería una de las desventajas de LinkedList (a menos que sea doblemente vinculada, pero esto representa mayor uso de memoria para guardar el puntero extra).

Una de las ventajas de las LinkedLists es que, al ser una estructura dinámica, puede crecer en tiempo de ejecución en base a la asignación y des asignación de memoria, por lo que no hay necesidad de darle un valor inicial.

También cabe mencionar que su implementación es más sencilla a pesar de que, en nuestro caso, estamos utilizando una estructura provista por las librerías de Java.

En el caso de los géneros, implementamos este mismo tipo de estructura dado que son un conjunto de tamaño reducido y la diferencia entre un tipo de estructura y otro no implica una diferencia notable.

Una solución más rápida habría sido utilizar un árbol binario de búsqueda. Sin embargo, según la consigna nos interesa un género en particular y encontrarlo en un ABB implicaría, en el peor de los casos, recorrer la mitad de sus elementos (O(h), siendo h la altura del árbol). Pero, como se mencionó anteriormente, al ser un conjunto acotado no va a presentar una diferencia notable y la implementación es más compleja.

# Conclusiones

Es importante saber o poder analizar el tipo de estructura con el cual se va a trabajar o procesar una solución. Es algo que a veces no se tiene en cuenta pero que a la larga o al crecer el proyecto puede traer consecuencias no deseadas en el rendimiento.