PRÁCTICA 2

Estructura de datos

Grupo 201 GII

María Guzmán Valdezate
Guillermo López de Arechavaleta Zapatero

Contenido

Introducción

En esta práctica, hemos trabajado con estructuras de datos de tipo Map, que permiten almacenar pares clave-valor y realizar búsquedas eficientes. Se han implementado varios métodos para la creación, inversión y transformación de mapas, lo que nos ha permitido explorar distintas formas de manipulación de estos elementos en Java.

El objetivo de este informe es analizar la complejidad algorítmica de los métodos desarrollados, justificando su eficiencia en términos de operaciones realizadas. Además, se analizará cómo estas implementaciones afectan al rendimiento y se destacarán los beneficios de emplear estructuras de datos apropiadas para resolver problemas computacionales.

Descripción y Análisis de Métodos

creaMapa (Collection<K> keys, Collection<V> values)

Descripción: Crea un mapa a partir de dos colecciones, una con las claves y otra con los valores. Si las colecciones no tienen el mismo tamaño, lanza una excepción.

Complejidad algorítmica: En el bucle for se recorre toda la colección de claves y valores una vez, lo que resulta en una complejidad de O(n).

creaMapa (Iterator<K> keyIterator, Iterator<V> valueIterator)

Descripción: Genera un mapa utilizando dos iteradores, uno para las claves y otro para los valores. Detiene la inserción cuando uno de los iteradores se queda sin elementos.

Complejidad algorítmica: Se recorre cada iterador una única vez, por lo que la complejidad es O(n).

mapaInverso (Map<K, V> map)

Descripción: Invierte un mapa intercambiando las claves y los valores. Si existen valores repetidos en el mapa original, el último par clave-valor sobrescribirá los anteriores.

Complejidad algorítmica: Se recorre la colección de entradas entera una vez, por lo que la complejidad es O(n).

multiMapaInverso (Map<K, V> map)

Descripción: Crea un multi-mapa en el que cada valor original del mapa se convierte en clave y almacena una colección con todas las claves que tenían dicho valor.

Complejidad algorítmica: Se recorre la colección de entradas una vez y se realizan inserciones en listas, lo que da como resultado una complejidad O(n).

Conclusiones

La implementación y análisis de los distintos métodos de manipulación de mapas nos ha permitido comprender mejor el funcionamiento de esta estructura de datos y su impacto en la eficiencia de los algoritmos. En general, la mayoría de los métodos presentan una complejidad óptima para las operaciones realizadas, garantizando un acceso rápido a los elementos almacenados.

Se ha identificado que la inversión de un mapa estándar puede presentar problemas cuando hay valores repetidos, lo que sugiere la necesidad de utilizar un enfoque diferente, como un multimap. Además, se ha destacado la importancia de elegir la estructura de datos adecuada para cada caso, ya que pequeñas diferencias en la implementación pueden afectar significativamente el rendimiento del programa.

Este análisis confirma la relevancia de los Map en el desarrollo de aplicaciones eficientes y su papel fundamental en la optimización de algoritmos que requieren almacenamiento y recuperación rápida de información.