Sergio Barbero Báscones

Estructuras de Datos

Práctica 8

UNIVERSIDAD DE BURGOS

INTRODUCCION

La práctica consiste en implementar una estructura de Mapa mediante direccionamiento por dispersión abierta.

Usaremos la clase ArrayList como base para nuestra tabla.

LOS MÉTODOS

Métodos de la clase padre

- int getNumberBuckets(): Nos devuelve el número de cubetas del mapa.

Complejidad algorítmica: Complejidad simple O(1).

- V put(K key, V value): Introduce elementos al mapa, y devuelve el elemento reemplazado o null en caso de que el elemento sea nuevo.

Complejidad algorítmica: Nos encontramos con una invocación al método get de esta misma clase, cuya complejidad es de O(n^2).

Nos encontramos con una bifurcación, nos quedamos con el peor caso posible, en el caso del then nos encontramos con otra bifurcación mas, que estudiaremos mas adelante, en el caso del else nos encontramos con una invocación a la misma función, por lo que sería recursivo: O(n)

En el caso del if nos encontramos con una bifurcación, por lo que nos quedaremos con la salida mas compleja, la condición if se basa en una llamada al método indexOf que internamente hace una iteración sobre los elementos (O(n)), el interior de dicha estructura es simple

En resumen: nuestra función es de complejidad O(n^2 + n)

- Void redimensionar(): Redimensiona el mapa al 150%

Complejidad algorítmica: O(1), devuelve el número de cubetas multiplicado por 1.5.

- V remove(Object key): Elimina un elemento del mapa y lo devuelve

Complejidad algorítmica: Nos encontramos con un iterador y una llamada a la función indexOf, por lo que la complejidad de la función será de O(n^2)

- V get(Object key): Devuelve el valor para una determinada clave.

Complejidad algorítmica: Nos encontramos con un bucle for y una llamada al método indexOf(key) anidados. Por lo que la complejidad de nuestra función es de O(n^2)

- Set<Entry<K, V>> entrySet(): Devuelve un set de pares clave-valor

Complejidad algorítmica: Nos encontramos con dos iteradores anidados, por lo que podemos concluir directamente que la complejidad de nuestra función es de O(n^2).

- Int size(): Devuelve el tamaño del mapa

Complejidad algorítmica: Nos encontramos con un iterador. Concluimos que la complejidad de la función es de O(n).

- Collection<V> Values(): Devuelve la colección de valores del mapa

Complejidad algorítmica: Tenemos dos iteradores anidados. Nuestra función tendrá complejidad O(n^2)

- Set<K> keySet(): Devuelve el set de claves del mapa

Complejidad algorítmica: Funciona de la misma manera que el anterior solo que con claves. Complejidad O(n^2)

Métodos de la clase EntradaMultiple

ArrayList<K> keySet(): Devuelve el array de claves de una EntradaMultiple.

ArrayList<V> valueSet(): Devuelve el array de valores de una EntradaMultiple.

Complejidad algorítmica: Estos dos algoritmos son de complejidad simple O(1).