

TRABAJO PRÁCTICO N ° 5

Diccionarios y mapeos

Departamento de Ciencias e Ingeniería de la Computación - U.N.S.

Ejercicio 1:

Suponiendo que cuenta con implementaciones para el TDADiccionario y el TDAMapeo, resuelva los siguientes problemas (sugerencia: resolver en papel):

- Escriba un método que reciba dos mapeos M1 y M2 de enteros en enteros (nros de libreta - nota materia) que devuelva una `PositionList<Entry<Integer,Integer>>` L con aquellos elementos E1 de M1 y E2 de M2 que coincidan en la clave, pero tengan un valor diferente. Por ejemplo, si E1= (LU: 29303, Nota: 8) pertenece a M1 y E2= (LU:29303, Nota: 7) pertenece a M2, entonces E1 y E2 deben estar en L.
- Escriba un metodo que dados dos mapeos M1 y M2, determine si todas las claves de M1 están contenidas en M2.
- Escriba un método cuya signatura sea: `public Dictionary<K,V> acomodar (Dictionary<K,V> d)` que reciba un diccionario d, y que retorne un nuevo diccionario igual a d pero sin claves repetidas. De esta manera, el diccionario resultante de este procedimiento no tendrá entradas con claves iguales. Utilice un mapeo auxiliar para resolver este ejercicio.
Por ejemplo: Si $d = \{(1,a), (2,b), (3,a), (2,c), (1,d), (4,b)\}$, entonces el diccionario resultante es $dRes = \{(1,d), (2,c), (3,a), (4,b)\}$.
- Escriba un método tal que reciba una `PositionList<Character>` y retorne un mapeo cuyas claves sean cada uno de los caracteres que aparecen en la lista y el valor la cantidad de veces que aparecen los caracteres en la lista. Por ejemplo: Si $l = \langle a, b, a, c, d, b \rangle$, entonces el mapeo resultante es $mRes = \{(a,2), (b,2), (c,1), (d,1)\}$.

Ejercicio 2: Implemente el TDAMapeo con hash abierto (arreglo de listas de entradas). Utilice la interfaz `Map<K,V>` provista por la cátedra. Corra el correspondiente tester JUnit para probar su implementación.

Ejercicio 3: Implemente el TDADiccionario con hash abierto (arreglo de listas de entradas). Utilice la interfaz `Dictionary<K,V>` provista por la cátedra. Corra el correspondiente tester JUnit para probar su implementación.

Ejercicio 4:

- Utilizando las implementaciones dadas en los ejercicios 2 y 3, utilice el entorno de programación para probar las soluciones propuestas a los problemas presentados en el ejercicio 1.
- Calcule, para cada inciso del ejercicio 1, el tiempo de ejecución e indique su orden. Indique en lenguaje natural las estructuras subyacentes utilizadas.

Ejercicio 5:

- Agregue una operación a la clase `Diccionario<K,V>` programada en el ejercicio 3 con la siguiente signatura: `Iterable<Entry<K,V>> eliminarTodas(K c,V v) throws InvalidKeyException`. Este método debe eliminar del diccionario todas aquellas entradas con clave c y valor v. Retorna un iterable con todas las entradas eliminadas. Recuerde que está agregando un método a la clase `Diccionario<K,V>`, por lo tanto tiene total acceso a su estructura.
- Calcule el orden del tiempo de ejecución de su solución. Indique en lenguaje natural las estructuras subyacentes utilizadas.