Práctica 2: Mapas y diccionarios

Ejercicio 1: Encontrar k números con el máximo de repeticiones en un array

Dado un array de en números y u n entero positivo k, se pide encontrar, dentro del array, los k números com más repeticiones, es decir los k números con mayor frecuencia de aparición en el array. Si dos números tienen la misma frecuencia, el mayor de los dos tendrá preferencia. Los números deben aparecer en orden decreciente de frecuencia. Ejemplos de funcionamiento:

Entrada:
7, 3, 5, 12, 4, 7, 2, 1, 3, 9
2
Salida:
7 3
// La frecuencia de 7 y de 3 es 2, el orden de aparición se debe a que 7 > 3

Entrada:
7, 10, 11, 5, 2, 5, 5, 7, 11, 8, 9
4
Salida:
5 11 7 10
// la frecuencia de 5 es 3, la de 11 y 7 2, y la de 10 es 1

Se recomienda crear un `HashMap` para almacenar pares elemento-frecuencia. Después ordenar el par elemento-frecuencia en orden decreciente de frecuencia, utilizar el método `sort` de `Collections` pasándole un `Comparator`. Finalmente imprimir los primeros k elementos.

Ejercicio 2: código Huffman

Se desea desarrollar una clase que permita codificar y decodificar texto a código binario comprimido utilizando la codificación Huffman.
Para ello se debe construir la clase `HuffmanTree` que tendrá el constructor y 2 métodos públicos:

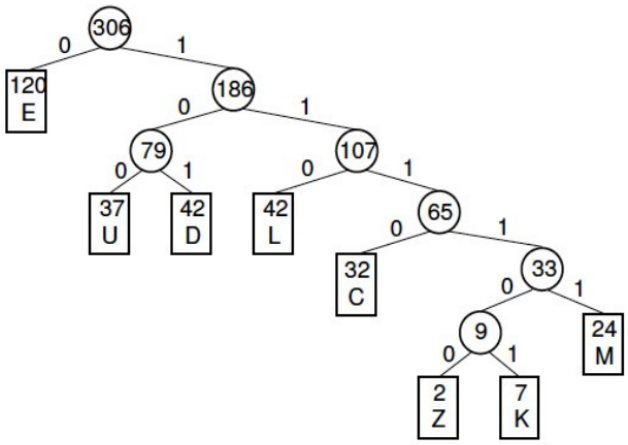
- `HuffmanTree(String)`
- `byte [] encoding(String)`
- `String decoding(byte [])`

El constructor `HuffmanTree(String)` construye un árbol Huffman utilizando el siguiente algoritmo:

1. Construya una tabla de frecuencias para las letras de un texto. De manera que para cada letra conozca el número de apariciones de dicha letra en el texto. Por simplicidad se supondrá que el texto contiene al menos una aparición de todas las posibles letras que puedan aparecer en la fase de codificación.

Letter	Z	K	M	C	U	D	L	E
Frequency	2	7	24	32	37	42	42	120

2. Prepare una colección de n árboles Huffman iniciales, cada uno de los cuales es un solo nodo de hoja. Ponga los n árboles en una cola de prioridad organizada por peso (frecuencia).
3. Elimine los dos primeros árboles de la cola (los de menor peso). Una estos dos árboles para crear un nuevo árbol cuya raíz tenga un nodo cuyo peso es la suma de los pesos de los dos árboles unidos.
4. Ponga este nuevo árbol en la cola de prioridades.
5. Repita los pasos 3 y 4 hasta que todos los árboles Huffman parciales se hayan combinado en uno solo.
6. Una vez construido el árbol de Huffman construya una tabla que asigne a cada letra del paso 1 un código que se obtiene por el camino utilizado para llegar a esa letra desde la raíz. Partiendo de la raíz, si la arista es a un hijo izquierdo se inserta un 0, y si es a un hijo derecho se inserta un 1.



Letter	Freq	Code	Bits
E	120	0	1
D	42	101	3
L	42	110	3
U	37	100	3
C	32	1110	4
M	24	11111	5
K	7	111101	6
Z	2	111100	6

El método `encoding()` recibe un texto y devuelve un array de bytes con la codificación Huffman.

El método `decoding()` recibe un array de bytes con una codificación Huffman y devuelve un texto.