Compresión de archivos de texto por encriptación binaria

Tomás Mateo Núñez Ríos

Índice

1.- Introducción

Uno de los problemas de la informática es el tamaño de los archivos que utilizamos a veces, esto nos dificulta el simple hecho de enviar y recibir o tener guardado en memoria cualquier archivo, nos demorará mucho tiempo y espacio que podríamos dedicar a otros quehaceres.

La solución a este problema llega cuando se crean algoritmos que reducen el tamaño en bytes de los archivos para ocupar menos espacio.

Si nosotros tenemos un char, este ocupa en memoria un byte (8 bits), pero si transformamos ese char en una secuencia de bits menor que 8 ya hemos comprimido una letra de todo el texto, así se procedería con todos los distintos caracteres que tenga el texto, por ejemplo en la palabra “coco” solo tenemos dos caracteres distintos y si cada uno lo convertimos en una cadena de 4 bits en lugar de 8 obtenemos una compresión del 50%.

Hay varios algoritmos para comprimir: Huffman, RLE, Codificación de bytes pares, diccionarios, etc.

Yo he intentado de recrear Huffman, que emplea árboles binarios para que las letras que más frecuencia tienen en el archivo ocupen menos ceros y unos compensando así, la cantidad de bytes que se dan al multiplicar 8 bits por la frecuencia (si conseguimos asignar uno o dos bits reducimos notablemente el archivo).

Debido a la complejidad de implementar los árboles binarios de Huffman, he tenido que pensar en otro método para encriptar en binario y al final me he decantado por la tabla Núñez (porque me la he inventado yo).

2.- Objetivos

Necesitaremos recorrer todo el fichero de texto para obtener la frecuencia de cada letra, por lo que tendremos dos arrays (letra y frecuencia).

Al ordenar las letras por frecuencias podremos crear la tabla e introducir los ceros y unos.

Una vez hecha la tabla podremos recorrerla para generar el diccionario, la regla que tendrá la base de datos para cifrar y descifrar.

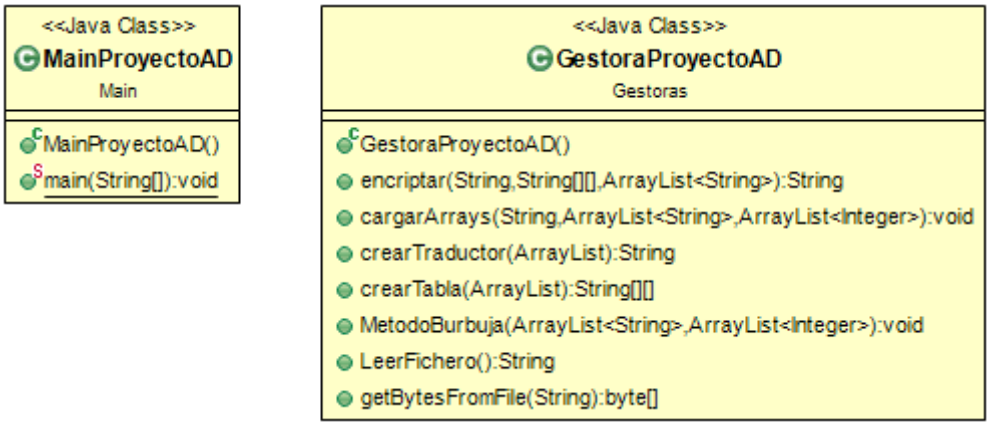
Para descifrar/descomprimir el archivo, lo que tendríamos que hacer es la inversa del cifrado, recorrer la información de la base de datos e ir traduciendo gracias al diccionario creado para esa encriptación.

Crear los ficheros de textos con la encriptación y guardar la traducción (diccionario) en la base de datos.

3.- Herramientas utilizadas

- Intellij IDEA
- Eclipse
- SQLite Studio

4.- Diagramas de clases y diagrama entidad relación



En el diagrama entidad-relación solo tengo una tabla con dos campos (no adjunto imagen porque me parece una tontería para dos columnas).

NombreFichero	Varchar(50)
Traduccion	Varchar(50000)

5.- Implementación de la aplicación

La implementación está adjuntada en el .rar

6.- Conclusiones

La conclusión más clara que he sacado con este trabajo es que a priori comprimir puede ser sencillo, pero detrás hay un método de encriptación muy potente que al implementar en un entorno de desarrollo, se convierte en un demonio y eso es solo el cifrado para comprimir, después necesitas, según la misma regla que has utilizado antes, descifrar el archivo comprimido para mostrar el texto original.

Otro punto que he visto es que hay que tener un buen manejo de los String, de los char, básicamente los tipos de datos que envuelven a los textos y de las estructuras de datos.