Algoritmos y Estructuras de Datos
Proyecto Segundo Parcial – 2015

Conteo y repetición de palabras en un archivo de texto.

Una forma simple de comprimir un archivo de texto es reemplazar cada palabra por un codigo binario que la represente. Por ejemplo, la palabra “casa” se reemplaza por el binario 1001 que ocupa 4 bits en lugar de 4 bytes. Para que la compresión sea efectiva, se debe reemplazar la parabra mas repetida por un código mínimo, y asi en forma creciente en el tamaño de código hasta la palabra menos repetida. Para ello es necesario contar cuantas veces se repite cada palabra en el texto.

La tarea consiste en contar cuantas veces se repite cada palabra. Para ello deberá leer un archivo de texto de al menos 500 palabras con al menos 100 palabras distintas e identificar cuales son las palabras que lo forman y cuantas veces están repetidas cada una de ellas. El archivo es de libre elección.

Para contar las palabras deberá armar un árbol binario de búsqueda donde cada nodo contenga la palabra y la cantidad de veces que figura repetida. Al leer una palabra del archivo, debera buscarla en el árbol. Si el mismo no existe, será agregada con el número de repetición 1. Si ya existiera, le sumará uno a las repeticiones previamente contadas. Deberá programar una clase abb como la dada en clase, pero en lugar de trabajar con nodos en memoria dinámica, deberá utilizar un arreglo de nodos de tamaño fijo en 1000, manteniendo los mismos métodos de la clase abb dada, es decir, debe modificar el código para que se adapte a un arreglo de nodos.

El árbol binario de búsqueda debe respetar las propiedades de AVL, por lo que al realizar la inserción de una nueva palabra, deberá cuidar el balance del árbol. Para esto debe adaptar el código respectivo dado en clases.

Al finalizar la lectura del archivo y tener completo el árbol, deberá indicar la altura del árbol creado y generar un listado de las palabras ordenadas de mayor a menor, según en número de repeticiones que tenga, indicando palabra y número de repetición, de forma tal que sea la entrada de la generación de los códigos de compresión. Debera usar el algoritmo de inserción y para evitar la duplicación de datos, la lista ordenada deberá tener referencias a los nodos del árbol, es decir, a los elementos del arreglo de soporte.

Opción de Promoción

Para alcanzar la opción de promoción, además de lo solicitado para la condición de regularidad, deberá realizar una segunda versión del proyecto que repita la misma funcionalidad de la primera parte, pero que difiera en la estructura de soporte del árbol AVL. Para este caso, el árbol debe estar basado en una lista simplemente enlazada que tenga un acceso directo con costo de órden O(lg n) por medio de un segundo árbol de direccionamiento, en lugar del arreglo utilizado en la primera parte. De esta forma, el armado del AVL dependerá de las posiciones en la lista, en lugar de las posiciones del arreglo. El algoritmo de ordenamiento para esta parte del proyecto es el algoritmo de quick-sort.