Hoja de trabajo No. 7

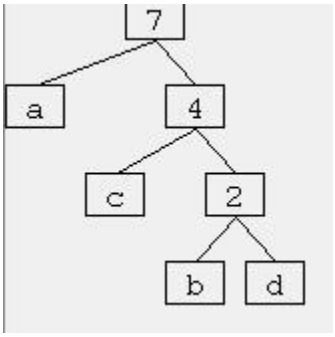
Realizar: Programa de implementación de **árboles binarios y heaps.**
Realizarse: en PAREJAS.

Objetivos:
a. Implementación de las operaciones de un árbol binario.
b. Utilización de Priority Queues implementados por medio de Heaps.
c. Codificación de mensajes empleando árboles de Huffman.

Programa a realizar:

El programa construye un árbol de Huffman dado un mensaje de entrada, conformado por una cadena de caracteres, muestra los códigos que le corresponden a cada caracter y la frecuencia de ese caracter en el mensaje.

Por ejemplo, si ingresa el mensaje: **abaccda**
Se obtendría como salida la tabla con el código asignado a cada carácter: (NO es necesario mostrar el árbol)



Caracter	frecuencia	código[1]
a	3	0
b	1	110
c	2	10
d	1	111

NOTA: puede usar como referencia del algoritmo, el documento que acompaña esta hoja de trabajo: "Algoritmo de Huffman.doc"

Diseño:
a. Debe implementarse las operaciones de un árbol binario.
b. Para la fase de construcción utilizar un heap que guarde todos los árboles que se van construyendo (Foresta) para seleccionar el que tiene menor frecuencia.

Tareas:
a. Construir el diagrama UML de clases, que muestre las clases que su grupo construirá para implementar el árbol binario, el heap y el algoritmo de Huffman.
b. Construir el diagrama UML de secuencia, para indicar la comunicación entre las clases para DECODIFICAR un mensaje usando el árbol de Huffman.
c. Construir el programa que construya el árbol de Huffman y muestre los códigos que corresponden a cada carácter del mensaje. Recuerde que la Foresta o colección de árboles intermedios, se debe almacenar en Priority Queue (que se implementa con un Heap).
d. Debe dejar evidencia de todo el desarrollo en el repositorio de Subversion.
e. Debe implementar casos de prueba, como mínimo para cada operación del árbol binario. Puede usar de referencia el capítulo 12 del libro de texto, Binary Trees. Estos casos de prueba también deben estar en el repositorio de Subversion.
f. Debe implementar los casos de prueba para los métodos del Heap. Puede usar de referencia el capítulo 13 del libro de texto, Priority Queues.

[1] NOTA: este es un posible código del carácter en un árbol de Huffam. Ya que el código depende de la frecuencia de ocurrencias del carácter en el mensaje, si existen varios caracteres con la misma frecuencia pueden tener distinto código.

g. Su programa lee un mensaje ya utilizando el código producido por su programa y lo muestra convertido a caracteres. Por ejemplo lee el mensaje: 110100 y produce: bca (si se sigue el ejemplo mostrado arriba). NOTA: su programa debe indicar si la cadena de 0 y 1 ingresada es un mensaje correcto, de acuerdo al árbol de Huffman que usted construyó.

Debe subir a Blacboard todos los productos elaborados y los enlaces a su repositorio de Subversion.

Calificación: su programa debe funcionar para ser calificado.

Aspecto	Puntos
Diagrama UML de clases. Muestra las clases necesarias para el árbol de Huffman, el árbol binario, priority queue y heap.	10
Diagrama UML de secuencia. Muestra la interacción entre las clases para DECODIFICAR un mensaje, con el código producido por su árbol de Huffman.	15
Implementación de las operaciones del árbol de Huffam. (Recordar que es un árbol binario).	10
Implementación del Priority Queue, usando un Heap (que se emplea para guardar la Foresta o conjunto de árboles que se generan en el transcurso del algoritmo).	15
Casos de prueba: debe existir como mínimo uno por cada operación del árbol binario. Usar JUnit.	10
Casos de prueba: debe existir como mínimo uno por cada operación del Heap.	10
Uso del repositorio: existen más de tres versiones guardadas, la última versión es igual a la colocada en el Blackboard	10
Funcionamiento del programa: generación de los códigos a emplear para cada carácter del mensaje de entrada. Lectura de un mensaje codificado y se traduce al mensaje expresado en caracteres.	20
TOTAL:	100