UNIDAD TEMÁTICA 4: Arboles Binarios

TRABAJO DE APLICACIÓN 2

Ejercicio #1

El objetivo de este ejercicio es implementar varias operaciones del TDA *TArbolBB*, a fin de aplicarlo más tarde a situaciones reales. Utilizar para esto los seudocódigos publicados, los ejercicios domiciliarios individuales y **las interfaces publicadas.**

PASO 1: TODO EL EQUIPO:

a) Crear paquete en proyecto Netbeans: **UT4-TA2** **(IMPORTANTE: SOLO SE EVALUARÁ EL CÓDIGO CONTENIDO EN ESTE PAQUETE)**
b) Agregar la clase Manejador de Archivos Genérico
c) Crear el *Main*
d) Crear *TArbolBB* y *TElementoArbolBB* de acuerdo a las interfaces publicadas
e) **Sincronizar las máquinas para los sub-equipos**.

PASO 2: EN SUBEQUIPOS:

Sub-Equipo A:

a) Implementar y probar el algoritmo para ***Insertar*** un nuevo *ElementoAB* en el *Arbol*. El método debe implementar un contador que se incrementa con cada invocación al mismo y debe mostrar por consola el contenido final del contador.
b) En la clase *Main*, agregar funcionalidad que permita ***cargar*** las claves desde un archivo de texto "claves.txt" , en el cual habrá una clave por línea, y que escriba otro archivo en que, en cada línea, escriba la clave insertada y el valor del contador (**0** si no se pudo insertar).
c) Implementar un método que devuelva una *string* con el recorrido del árbol en **preorden;**

Sub-Equipo B:

a) Implementar y probar el algoritmo para **Buscar** un elemento en el árbol, dada una clave (comparable, string o entero).
b) En la clase *Main*, agregar funcionalidad que permita cargar las **consultas** desde un archivo de texto "*consultas.txt*", en el cual habrá una clave por línea, y que escriba otro archivo en que, en cada línea, escriba la clave buscada y el valor del contador (con signo negativo si la clave no fue encontrada en el árbol).
c) Implementar un método que devuelva una string con el recorrido del árbol en **postorden;**
d) Implementar un método que devuelva una string con el recorrido del árbol en **inorden;**

PASO 3: TODO EL EQUIPO

a) Integrar los cambios en el repositorio
b) Utilizando el programa realizado y los archivos que se liberarán, responder preguntas proyectadas en pantallaEntrega (cierre de la tarea 21:05)

SE CALIFICARÁ SOBRE EL CÓDIGO FUENTE EXISTENTE EN EL GIT, **EXCLUSIVAMENTE EN EL PAQUETE INDICADO**, HASTA LA HORA **21:05**

NOTA IMPORTANTE: verificar el cumplimiento de los requerimientos especificados en el documento "Lineamientos generales para entrega de Trabajos de Aplicación" disponible en la Webasignatura.

Ejercicio #2

Agregar funcionalidades al TDAArbol.

PASO 1: EN SUBEQUIPOS:

Sub-Equipo A:

a) Implementar el algoritmo para **obtener la altura** del árbol.
b) Implementar el algoritmo para **obtener la cantidad de hojas** del árbol.
c) Probar los algoritmos insertando en el árbol varias claves y verificar en papel

Sub-Equipo B:

a) Implementar el algoritmo para **obtener el tamaño** del árbol.
b) Implementar el algoritmo para, dada una clave, **indicar en qué nivel se encuentra**.
c) Probar los algoritmos insertando en el árbol varias claves y verificar en papel

PASO 2: TODO EL EQUIPO

a) Integrar los cambios en el repositorio
b) Utilizando el programa realizado y los archivos que se liberarán, responder preguntas proyectadas en pantalla

NOTA IMPORTANTE: verificar el cumplimiento de los requerimientos especificados en el documento "Lineamientos generales para entrega de Trabajos de Aplicación" disponible en la Webasignatura, Unidad 0.

Ejercicio #3

Operaciones Complementarias (una vez terminados los ejercicios 1 y 2)

1. Obtener la menor clave del árbol.
2. Obtener la mayor clave del árbol.
3. Obtener la clave inmediata anterior a una clave dada (pasada por parámetro)
4. Obtener la cantidad de nodos de un nivel dado (por parámetro)
5. Listar todas las hojas cada una con su nivel.
6. Verificar si el árbol es de búsqueda.