# UNIDAD TEMÁTICA 4: Arboles Binarios

## TRABAJO DE APLICACIÓN 4

## Ejercicio #1

**Agregar funcionalidades al TDA Arbol Binario de Búsqueda.**

**PASO 1: EN SUBEQUIPOS:**

**Sub-Equipo A:**

a) Analizar **pre y post-condiciones** de un algoritmo para **obtener la cantidad de hojas** del árbol.
b) Desarrollar en seudocódigo un algoritmo para **obtener la cantidad de hojas** del árbol. Firmas:
    - de TArboBB **cantidadHojas** : entero
    - de TElementoAB **cantidadHojas** : entero

**Sub-Equipo B:**

a) Analizar **pre y post-condiciones** de un algoritmo para, dada una clave, **indicar en qué nivel se encuentra.**
b) Desarrollar en seudocódigo un algoritmo para, dada una clave, **indicar en qué nivel se encuentra**. Firmas:
    - de TArboBB **nivel (de tipoClave unaClave)**: entero
    - de TElementoAB **nivel (de tipoClave unaClave)**: entero

**PASO 2: EN SUBEQUIPOS**

**Sub-Equipo A:**

a) Escribir en lenguaje natural / seudocódigo un procedimiento para probar el algoritmo desarrollado por el sub-equipo B (dada una clave, **indicar en qué nivel se encuentra)**

**Sub-Equipo B:**

a) Escribir en lenguaje natural / seudocódigo un procedimiento para probar el algoritmo desarrollado por el sub-equipo A (**obtener la cantidad de hojas** que tiene el árbol)

**PASO 3: TODO EL EQUIPO**

Intercambiar seudocódigos de algoritmos y especificaciones de los casos de prueba desarrollados.

**REMITIR TODO LO REALIZADO A LA TAREA CORRESPONDIENTE EN LA WEBASIGNATURA. ESTE MATERIAL SERÁ NECESARIO PARA EL SIGUIENTE EJERCICIO.**

# UNIDAD TEMÁTICA 4: Arboles Binarios

## TRABAJO DE APLICACIÓN 4

## Ejercicio #2

**EN SUBEQUIPOS:**

**Sub-Equipo A:**

a) Revisar el seudocódigo desarrollado en el **Ejercicio 1 PASO 2** para probar el algoritmo desarrollado por el **sub-equipo B** (dada una clave, **indicar en qué nivel se encuentra)**
b) Implementar en JAVA el / los **test cases** para ese método
c) Enviar al Sub-Equipo B el código fuente de los test-cases generados (para que lo inserten en la clase TArbolBBTest)

**Sub-Equipo B:**

b) Revisar el seudocódigo desarrollado en el **Ejercicio 1 PASO 2** para probar el algoritmo desarrollado por el **sub-equipo A** (**obtener la cantidad de hojas** que tiene el árbol)
c) Implementar en JAVA el / los test cases para ese método
d) Enviar al Sub-Equipo B el código fuente de los test-cases generados (para que lo inserten en la clase TArbolBBTest)

## Ejercicio #3

**PASO1**

**EN SUBEQUIPOS:**

**Sub-Equipo A:**

a) Implementar en JAVA el algoritmo para **obtener la cantidad de hojas** que tiene el árbol.
b) Ejecutar los **test-cases** de este algoritmo (desarrollados en el Ejercicio 2 por el Sub-Equipo B) y corregir eventuales errores.
c) Enviar al Sub-Equipo B el código fuente generado (para que lo inserten en la clase TArbolBBTest)

**Sub-Equipo B:**

a) Implementar en JAVA el algoritmo para, dada una clave, **indicar en qué nivel se encuentra.**
b) Ejecutar los **test-cases** de este algoritmo (desarrollados en el Ejercicio 2 por el Sub-Equipo A) y corregir eventuales errores.
c) Enviar al Sub-Equipo B el código fuente generado (para que lo inserten en la clase TArbolBBTest)

**PASO 2: UNO DE LOS SUB-EQUIPOS ACTUALIZA EN EL REPOSITORIO EL PAQUETE UT4-TA4. SE CALIFICARÁ EL CONTENIDO DEL GIT HASTA LA HORA 1935.**

**Operaciones Complementarias** (una vez terminados los ejercicios 1 a 3)

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.