PARTE 2 - Equipos: Ejercicios de seudocódigo
Duración: 60 minutos

EJERCICIO 1

Los árboles binarios de búsqueda son usados para organizar un conjunto de elementos con el objetivo de que las búsquedas sobre él tengan un orden del tiempo de ejecución logarítmico en el mejor de los casos. Este mejor caso corresponde a que el árbol resulte armado completo, con todas sus hojas al mismo nivel.

Para asegurar un orden logarítmico en las búsquedas en todos los casos, se introduce el criterio de balance "AVL" que establece que, para cada nodo del árbol, la diferencia de alturas de sus dos subárboles no puede ser mayor a uno.

Insertar las siguientes claves en el orden dado en un árbol AVL:

72, 24, 12, 62, 32, 47, 17, 19, 21, 23, 40, 51, 63.

CONSIDERACIONES IMPORTANTES

- **Se debe mostrar paso a paso la inserción de cada clave.**
- **Cuando se produce un desbalance, se debe identificar el nodo desbalanceado, el tipo de desbalance y los distintos nodos que se ven involucrados en la operación de balanceo (k1, k2, k3).**
- En caso de rotaciones complejas, se puede obviar mostrar el estado intermedio luego de realizar la primera rotación.

EJERCICIO 2

Como una medida posible que da una idea sobre la forma de un árbol binario, se define como "longitud de trayectoria interna" (LTI) de un árbol a la suma de los niveles de todos los nodos del árbol, y como "longitud de trayectoria interna media" (LTIM) a ese valor dividido por la cantidad de nodos del árbol (tamaño).

$LTI = \sum_{i=1}^{n} h_i$ donde "n" es el tamaño del árbol y "hi" es el nivel del nodo "j"

$$LTI = \frac{LTI}{n}$$

Escribir un algoritmo (método de árbol y método de nodo) que, aplicado a un árbol binario de búsqueda, devuelva su longitud de trayectoria interna promedio, con la siguiente firma:

TArbolBB.longTrayIntMedia(): devuelve un número real

La firma del método de nodo es abierta, y el método de árbol sería más simple y conveniente que invocara a más de un método (por ejemplo, dos métodos), que deben desarrollarse completamente.

CONSIDERACIONES IMPORTANTES

- Deben desarrollarse en detalle todos los métodos que se invoquen.
- Se tendrá en cuenta la eficiencia de los algoritmos desarrollados, calificando mejor aquellos que tengan mejor tiempo de ejecución o mejor performance general.
- No es necesario desarrollar los métodos de los TDAs utilizados si estos corresponden a las variantes vistas en clase.
- Se deben cumplir todos los pasos estándar de desarrollo de algoritmos, Lenguaje Natural, Pre y Post Condiciones, Pseudocódigo y Análisis del Orden del Tiempo de Ejecución