PRÁCTICO ÁRBOL AVL

1. **Inserciones y FE paso a paso (caso LL y RR)**
   Inserte en un AVL la secuencia: `30, 20, 10, 40, 50, 60`.
   a) Dibuje el árbol tras cada inserción.
   b) Calcule alturas y factor de equilibrio (FE) de cada nodo en cada paso.
   c) Indique qué rotación se aplica en cada desbalance (LL o RR) y por qué.

2. **Inserciones con rotación doble (caso LR y RL)**
   Inserte la secuencia: `30, 10, 20, 40, 35, 37`.
   a) Muestre el estado del árbol en cada paso.
   b) Identifique los desbalances (FE = ±2).
   c) Especifique cuándo corresponde **rotación doble** (LR o RL) y ejecútela.

3. **Secuencia ordenada y "efecto peinar"**
   Inserte `5, 10, 15, 20, 25, 30, 35` en un ABB y luego balancee hasta AVL (o inserte directamente en AVL, mostrando reequilibrios).
   a) Explique por qué un ABB puro se desbalancea con datos crecientes.
   b) Detalle las rotaciones que hacen que el AVL mantenga altura O(log n).

4. **Eliminación con rebalanceo**
   Dado el AVL resultante de insertar `50, 30, 70, 20, 40, 60, 80, 65, 75`, elimine: `20`, luego `70`.
   a) Dibuje el árbol tras cada borrado.
   b) Indique FE de los nodos afectados y rotaciones necesarias para restaurar el balance.

5. **Comprobador de AVL**
   Implemente un método `esAVL(Nodo r)` que:
   - Devuelva `(esAVL, altura)` en una sola pasada recursiva.
   - Verifique que para todo nodo `|altura(izq) - altura(der)| ≤ 1` y que además respete la propiedad de ABB.
     Pruebe con árboles válidos e inválidos pequeños.

6. **Factor de equilibrio completo**
   Inserte `10, 100, 20, 80, 40, 70` (o una variante equivalente) y:
   a) Liste para el árbol final `(valor, altura, FE)` de todos los nodos.
   b) Marque los nodos críticos donde surgieron FE = ±2 durante el proceso.

7. **Implementación guiada: rotación izquierda**
   Complete el código de una **rotación simple a izquierda** y úselo en `insertar`.
   a) Muestre antes/después sobre un subárbol donde ocurra caso RR.
   b) Actualice correctamente las **alturas** involucradas.

(Pueden partir de un esqueleto de `NodoAVL{ valor, altura, izq, der }`.)

8. **Implementación guiada: rotación doble izquierda-derecha (LR)**
   Programe `rotacionDobleIzquierdaDerecha(n)` usando dos rotaciones simples.
   a) Justifique por qué LR ≡ (rotación simple izquierda en hijo) + (rotación simple derecha en n).
   b) Valide con el caso del ejercicio 2.

9. **Costos y altura**
   a) Demuestre (informalmente) que la altura del AVL es O(log n).
   b) Explique por qué eso garantiza operaciones de búsqueda/insertar/eliminar en O(log n).
   c) Compare brevemente con ABBs sin balance y con rojinegros (solo conceptual).

10. **Secuencias “estresantes” y pruebas unitarias**
    a) Genere 3 secuencias de 20 números (creciente, decreciente, pseudoaleatoria con repetidos) e inserte en un AVL (ignore repetidos si su diseño no los admite).
    b) Escriba tests que verifiquen tras cada inserción: `esAVL == true`, alturas correctas y orden in-order creciente.
    c) Informe cuántas rotaciones totales se aplicaron en cada secuencia.