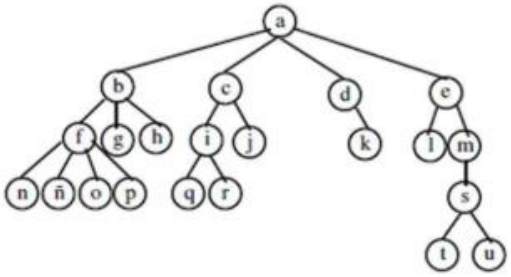
GUÍA DE EJERCICIOS - ÁRBOLES: ABB, AVL y ÁRBOL B.

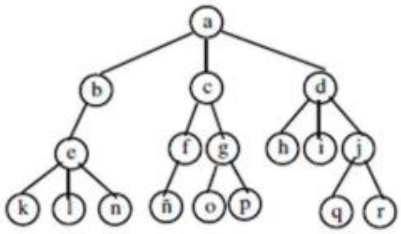
1. Explica la diferencia entre un árbol general y un árbol binario.
2. Revisa la especificación teórica del TAD ABB vista en clase. ¿Qué operaciones modificaría o agregarías si el tipo base (a) no admite comparación <, >, ==? ¿Cómo afectaría eso a su implementación?
3. Responda justificando conceptualmente:
    a. ¿Cuándo es más conveniente usar un ABB que un AVL?
    b. ¿Qué ventajas e inconvenientes presenta la implementación con nodos frente a la implementación con arreglo?
    c. ¿Qué relación hay entre la altura del árbol y la eficiencia de sus operaciones?
4. Dado los siguientes árboles escribe el orden en que se visitan los nodos en preorden, inorden y postorden:

i

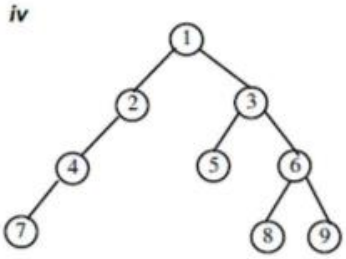


ii

iii



iv



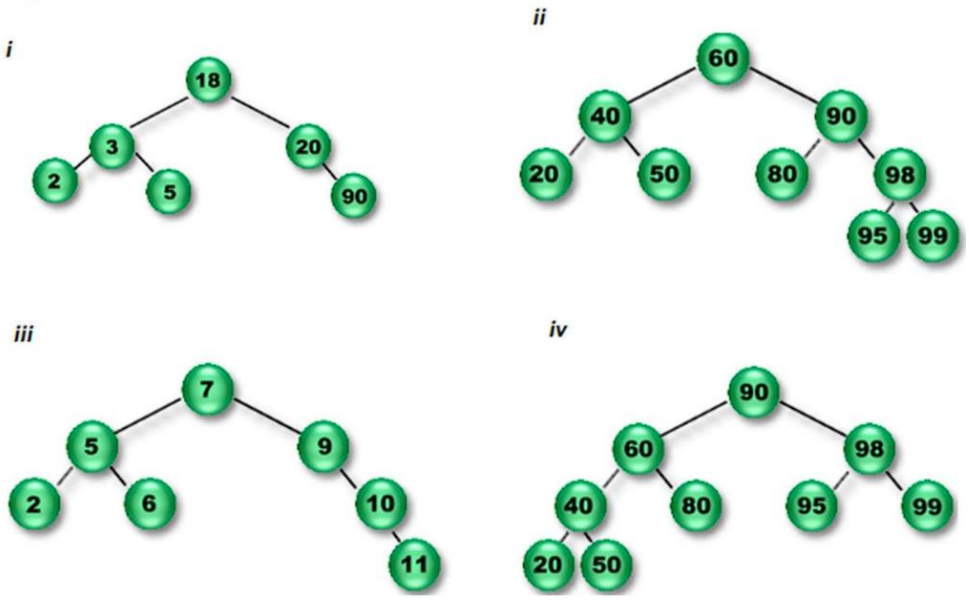
5. Construye un ABB para cada una de las siguientes secuencias:
    a. (20, 5, 40, 37, 45, 1, 7, 38)
    b. (120, 55, 42, 35, 72, 65, 126)
    c. (35, 17, 4, 50, 23, 8, 30, 15, 12, 41)

    Luego indica en cada caso la altura del árbol.
6. Para un árbol binario de altura *h*, ¿cuál es el máximo número de nodos posibles? Justifica.

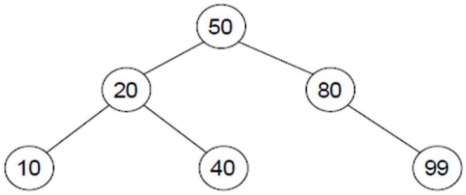
7. Para cada una de las siguientes secuencias construye un ABB y luego elimina los valores indicados. Describe cada caso y dibuja el árbol resultante
    a. (5, 10, 8, 3, 12, 0) - eliminar 0 y 5
    b. (50, 45, 17, 68, 99, 75, 5, 51, 78, 76) - eliminar 17, 68 y 50
    c. (3, 5, 13, 43, 92, 97, 25, 41) - eliminar 13, 25 y 43
8. Dada la siguiente estructura interna para un ABB en Go:

```
type Node struct {
 valor   int
 izquierda  *Node
 derecha *Node
}
type ABB struct {
 raiz *Node
}
```

    a. Implementa una función para insertar un valor en un ABB.
    b. Implementa una función en Go que busque un valor en un ABB y devuelva un booleano indicando si el valor está presente o no. Determina la cantidad de comparaciones realizadas en el peor caso.
    c. Implementa en Go la función eliminar(nodo, valor) -> nodo para eliminar un valor del ABB.
    d. Implementa funciones en Go para encontrar el mínimo de un ABB en forma recursiva e iterativa. Compara su rendimiento.
    e. Implementa una función para calcular la cantidad total de nodos de un ABB.
9. Menciona al menos tres aplicaciones prácticas donde se utilicen ABB y explica por qué son útiles.
10. Dado los siguientes árboles binarios determine para cada caso si se trata de un árbol AVL. Para cada nodo calcule su altura (H) y su factor de equilibrio (FE). En caso de que el árbol no sea AVL, indique el/los nodo/s desbalanceado/s y qué tipo de rotación (simple o doble, izquierda o derecha) se debe aplicar.



11. Partiendo del árbol AVL dado, realice las siguientes operaciones secuenciales Inserciones: 120, 45, 48, 30 Eliminaciones: 40, 50, 80, y para cada paso:
    a. Indique el factor de equilibrio (FE) de cada nodo del árbol después de cada operación.
    b. Si se produce un desequilibrio:
    c. Señale el nodo en el que se detecta el desbalance.
    d. Explique la causa del desequilibrio (tipo de inserción/eliminación que lo provocó y cómo afecta la altura de los subárboles).
    e. Describa con precisión la rotación utilizada (simple o doble, hacia qué dirección) para restaurar el equilibrio.
    f. Represente el estado del árbol resultante luego de aplicar la rotación.



12. Queremos construir un sistema de búsqueda rápida para una biblioteca digital. Cada libro está representado por su número de ISBN (entero). Se pide:
    a. Implementar un ABB o AVL que permita insertar los ISBN.
    b. Insertar los siguientes valores: [9780345391803, 9780131103627, 9780321573513, 9780596009205, 9780262033848]
    c. Implementar una función que indique si un ISBN dado está presente.

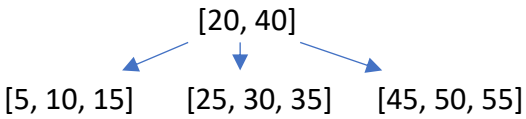
d. Comparar la cantidad de comparaciones necesarias para encontrar un ISBN si:
    i. a) se usa un ABB no balanceado
    ii. b) se usa un AVL
e. Justificar por qué un AVL puede ser preferido en este contexto.

13. Una empresa quiere gestionar el inventario de su depósito de forma eficiente. Cada producto está identificado por un código numérico único. Se desea:
    a. Implementar una estructura de datos (ABB o AVL) para almacenar los códigos de producto.
    b. Insertar los siguientes códigos en el orden indicado: [1012, 1020, 1005, 1030, 1008, 1015, 1025]
    c. Implementar una función para:
        i. Buscar un código en el inventario.
        ii. Obtener el código menor y el mayor.
    d. Calcular la cantidad de comparaciones necesarias para buscar el código 1025:
        i. si el árbol es un ABB no balanceado.
        ii. si el árbol es un AVL.
    e. Justificar cuál de las dos estructuras es más adecuada para este caso de uso.
14. Se debe diseñar una agenda digital para turnos médicos. Cada turno tiene un ID numérico que representa su orden en el día (cuanto menor el número, más temprano el turno). Se requiere:
    a. Implementar un ABB o AVL que permita:
        i. Insertar nuevos turnos.
        ii. Cancelar turnos (eliminar por ID).
        iii. Obtener el turno más próximo (mínimo).
    b. Insertar los siguientes turnos: [600, 900, 720, 660, 840, 780, 1020]
    c. Cancelar los turnos 720 y 600. Luego mostrar el estado del árbol.
    d. Determinar el impacto en la eficiencia de las operaciones si:
        i. Se utiliza un ABB sin balanceo.
        ii. Se utiliza un AVL balanceado.
    e. ¿Qué garantiza el uso de un AVL en este sistema?
15. Dado un árbol B de orden m = 5, responda:

a. ¿Cuál es el número mínimo y máximo de claves que puede tener un nodo interno (no raíz)?
b. ¿Cuál es el número mínimo y máximo de hijos que puede tener un nodo interno?
c. ¿Cuál es el número mínimo de claves que puede tener la raíz (si no es hoja)?
d. Justifique por qué los árboles B son especialmente adecuados para manejar grandes volúmenes de datos almacenados en disco.

16. Considere insertar las claves [10, 20, 5, 6, 12, 30, 7, 17] en un **árbol B de orden m = 4**, inicialmente vacío.
    a. a. Dibuje paso a paso el árbol B después de cada inserción, indicando claramente las divisiones de nodos cuando ocurran.
    b. b. Explique qué clave se promueve hacia el padre en cada división.
    c. ¿Qué propiedad del árbol B se preserva después de cada operación?

17. Considere el siguiente **árbol B de orden m = 5** ya construido:



a. Elimine los valores **30**, **25** y **35**, en ese orden.
b. Para cada eliminación:
    - Indique si es necesario realizar una **redistribución** o una **fusión**.
    - Dibuje el árbol después de cada operación.
    - Explique qué decisiones tomó:
        - ¿De qué hermano se toma la clave en una redistribución?
        - ¿Qué clave asciende o desciende en una fusión?
c. Después de eliminar los tres valores, ¿cuál es la altura del árbol resultante?

18. Responda justificando:
    a. ¿Qué estructura elegiría para administrar **10.000 registros** en memoria RAM que requieren acceso muy frecuente y rápido a claves?
    b. ¿Qué estructura elegiría para administrar **10 millones de registros** almacenados en disco duro, donde los accesos a disco son costosos?
    c. Complete la siguiente tabla comparativa:

Criterio	ABB	AVL	Árbol B
Altura en peor caso			
Eficiencia de búsqueda			
Coste de inserción/eliminación			

19. Imagine que debe diseñar la estructura de almacenamiento de un sistema de archivos en disco (similar a NTFS o ext4). ¿Qué tipo de árbol utilizaría (ABB, AVL, B o B+) para organizar las ubicaciones de archivos? Justifique ¿Qué ventaja ofrece un árbol B+ frente a un árbol B en esta aplicación específica?