Formando nuevas generaciones con sello de excelencia comprometidos con la transformación social de las regiones y un país en paz

Arboles Binarios de Búsqueda

Jesus Sebastian Delgado Sierra

Estructura de Datos Y Análisis De algoritmos

Profesor: Carlos Arturo Barrientos Suarez

17/05/2025

1. Introducción a los Árboles Binarios

Los Árboles Binarios de Búsqueda (ABB) son estructuras de datos que permiten almacenar elementos de forma ordenada, facilitando operaciones eficientes de búsqueda, inserción y eliminación. Su implementación en Java es común en aplicaciones que requieren manipulación dinámica de datos ordenados.

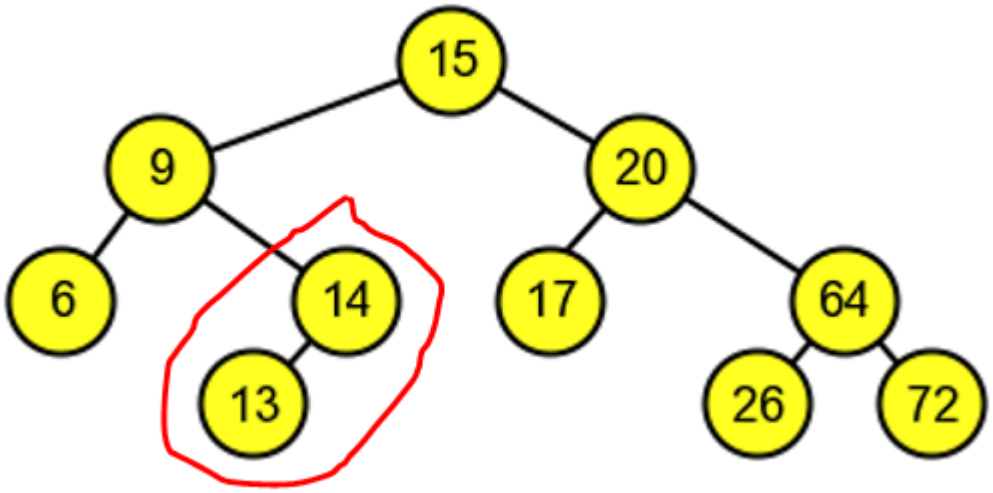
2. Definición de árbol binario de búsqueda

Un ABB es un tipo de árbol binario en el cual cada nodo cumple con la propiedad de que todos los elementos del subárbol izquierdo son menores que el nodo, y todos los elementos del subárbol derecho son mayores. Esta propiedad permite realizar búsquedas eficientes, ya que en cada paso se descarta la mitad del árbol.

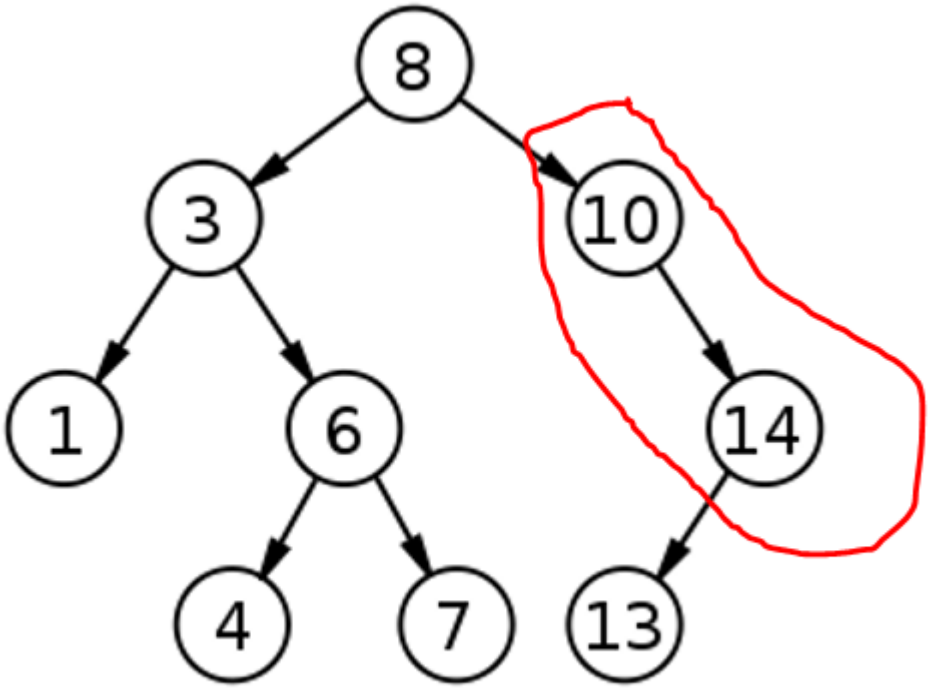
3. Estructura de un ABB

Valor: Dato almacenado en el nodo.

Referencia al hijo izquierdo: Apunta al subárbol con valores menores.

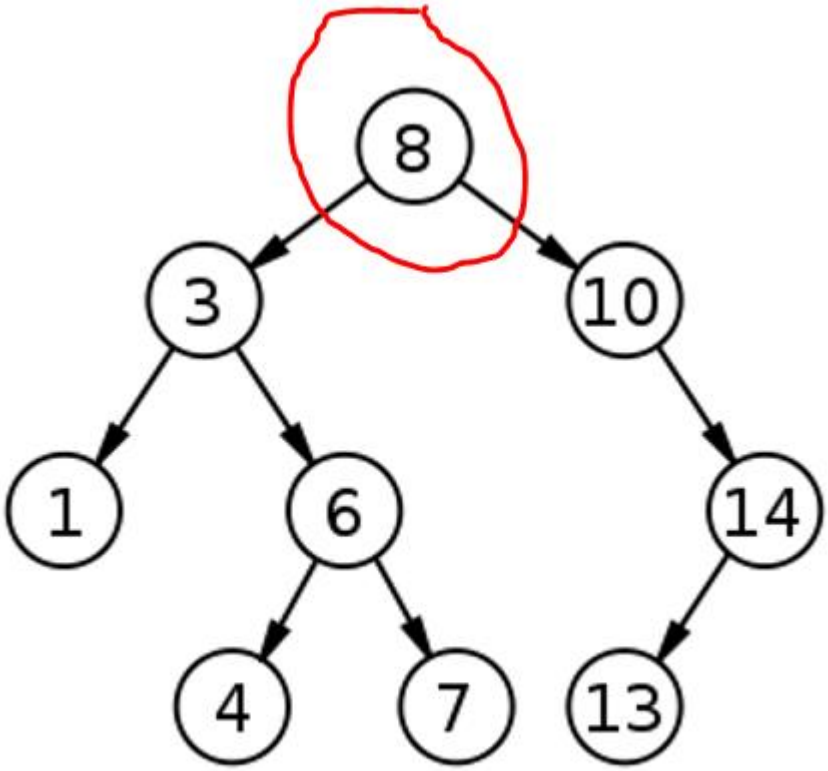


Referencia al hijo derecho: Apunta al subárbol con valores mayores.



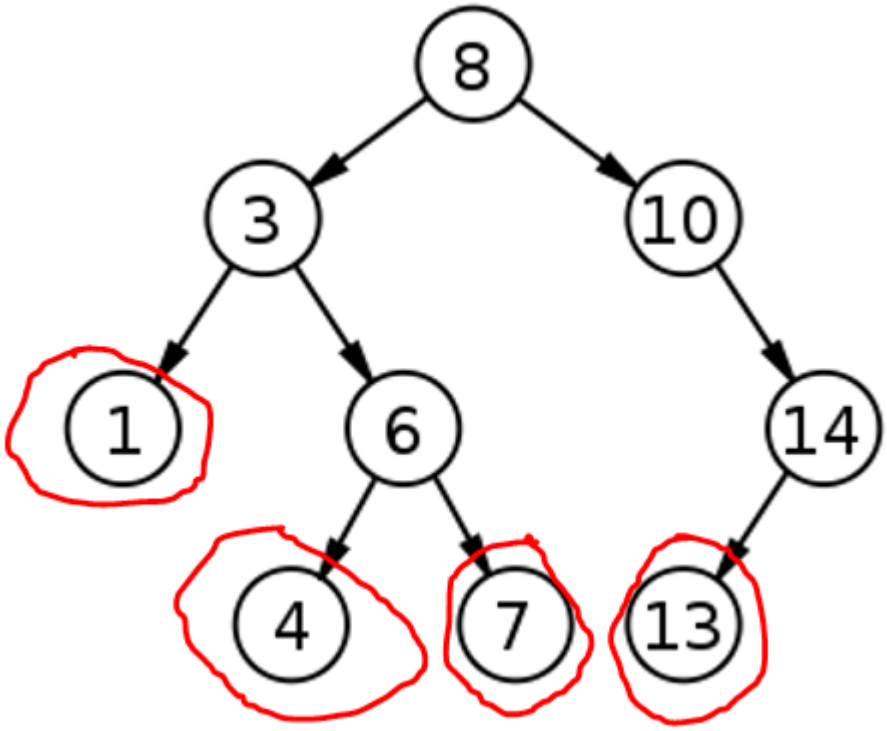
Además, se pueden identificar:

Nodo raíz: Primer nodo del árbol.



Nodos internos: Nodos con al menos un hijo.

Nodos hoja: Nodos sin hijos.



4. Operaciones Fundamentales

4.1 Inserción

Para insertar un nuevo valor:

1. Comenzar en la raíz.
2. Comparar el valor a insertar con el valor del nodo actual.
3. Si es menor, moverse al hijo izquierdo; si es mayor, al hijo derecho.
4. Repetir hasta encontrar una posición nula donde insertar el nuevo nodo.

4.2 Búsqueda

Para buscar un valor:

1. Comenzar en la raíz.
2. Comparar el valor buscado con el valor del nodo actual.
3. Si es igual, se ha encontrado el valor.
4. Si es menor, continuar en el subárbol izquierdo; si es mayor, en el subárbol derecho.
5. Repetir hasta encontrar el valor o llegar a un nodo nulo.

4.3 Eliminación

Eliminar un nodo puede implicar tres casos:

1. Nodo sin hijos: Se elimina directamente.
2. Nodo con un hijo: Se reemplaza el nodo por su único hijo.
3. Nodo con dos hijos: Se encuentra el sucesor inorden (el menor valor en el subárbol derecho), se reemplaza el valor del nodo a eliminar por el del sucesor, y luego se elimina el sucesor.

5. Implementación en Java

Una implementación básica en Java de un nodo de ABB:

```java
class Nodo {
    int valor;
    Nodo izquierdo, derecho;

    public Nodo(int valor) {
        this.valor = valor;
        izquierdo = derecho = null;
    }
}
```

Se pueden implementar métodos para insertar, buscar y eliminar nodos, así como para recorrer el árbol en diferentes órdenes (inorden, preorden, postorden).

6. Ventajas y desventajas

6.1 Ventajas:

- **Eficiencia:** Operaciones de búsqueda, inserción y eliminación en tiempo promedio O(log n).
- **Ordenamiento**: Los elementos se mantienen ordenados, facilitando operaciones como recorridos inorden.

Desventajas:

- **Desequilibrio**: Si los datos se insertan en orden ascendente o descendente, el árbol puede degenerar en una lista enlazada, perdiendo eficiencia.
- **Complejidad en balanceo**: Mantener el árbol balanceado puede requerir implementaciones más complejas, como árboles AVL o árboles rojo-negro.

7. Conclusiones

Los Árboles Binarios de Búsqueda (ABB) son estructuras fundamentales para la organización eficiente de datos, permitiendo búsquedas, inserciones y eliminaciones con un buen rendimiento cuando el árbol está balanceado. Su lógica sencilla y su implementación en Java los convierten en una herramienta accesible pero poderosa para programadores en formación y en entornos profesionales.

A pesar de sus ventajas, es importante tener en cuenta que su rendimiento puede verse afectado si el árbol se desequilibra, por lo que en casos más complejos se recomienda el uso de variantes balanceadas como los árboles AVL. En general, entender y saber implementar un ABB es clave para abordar problemas que requieren un manejo ordenado de información y una alta eficiencia en el acceso a los datos.

.