Video23 (1)

Conceptos Principales

1. a
2. b
3. c

Notas

Algoritmo de eliminación en Árbol Binario de Búsqueda (ABB)

La eliminación en un ABB es más compleja que la búsqueda o inserción, porque se deben **reorganizar los nodos** para mantener la propiedad de orden.

Existen **3 casos principales** según la cantidad de hijos del nodo a eliminar.

Pasos generales

1. **Buscar el nodo a eliminar** (igual que el algoritmo de búsqueda).
2. **Aplicar el caso correspondiente según hijos**:
    - Caso 1: nodo sin hijos.
    - Caso 2: nodo con un solo hijo.
    - Caso 3: nodo con dos hijos.

Caso 1: Nodo hoja (sin hijos)

- Ejemplo: eliminar `77`.
- El nodo no tiene ni hijo izquierdo ni derecho.
- **Acción**: simplemente eliminarlo (liberar memoria).

Regla: `if nodo.izq == NULL and nodo.der == NULL → free(nodo)`

Caso 2: Nodo con un solo hijo

- Ejemplo: eliminar `83` (tiene solo hijo izquierdo).
- **Acción**:
    - Reemplazar el nodo con su **único subárbol existente**.
    - Liberar el nodo eliminado.

Regla:

- Si `nodo.der == NULL → nodo = nodo.izq`.
- Si `nodo.izq == NULL → nodo = nodo.der`.

Importante manejar bien los **punteros** para no perder referencias (similar a listas enlazadas).

Caso 3: Nodo con dos hijos

- Ejemplo: eliminar `18` o `48`.
- El nodo tiene **subárbol izquierdo y derecho**.
- **Acción**:
    1. Buscar el **mínimo del subárbol derecho** (nodo más a la izquierda de la rama derecha).
    2. Reemplazar el valor del nodo actual con ese mínimo.
    3. Eliminar recursivamente el nodo mínimo encontrado (que tendrá caso 1 o 2).

Esto asegura que la propiedad del ABB se mantenga.

Ejemplo paso a paso (eliminando `18`):

1. Nodo `18` tiene hijo derecho → buscar mínimo en su subárbol derecho → `19`.
2. Reemplazar `18` por `19`.
3. Llamar recursivamente a `eliminar(19)` en el subárbol derecho.
4. `19` no tiene hijo izquierdo → aplicar caso 2 → liberar.

Pseudocódigo

```
func eliminar(nodo, valor):
    si nodo == NULL:
        return NULL

    si valor < nodo.valor:
        nodo.izq = eliminar(nodo.izq, valor)
    sino si valor > nodo.valor:
        nodo.der = eliminar(nodo.der, valor)
    sino:
        // Nodo encontrado
        si nodo.izq == NULL and nodo.der == NULL:
            free(nodo)
            return NULL
        si nodo.izq == NULL:
            temp = nodo.der
            free(nodo)
            return temp
        si nodo.der == NULL:
            temp = nodo.izq
            free(nodo)
            return temp

        // Caso 3: dos hijos
        temp = minimo(nodo.der)
```

```
            nodo.valor = temp.valor
            nodo.der = eliminar(nodo.der, temp.valor)

        return nodo
```

Resumen final

- **Caso 1**: eliminar directo si no tiene hijos.
- **Caso 2**: sustituir nodo por su único hijo.
- **Caso 3**: sustituir valor por el mínimo del subárbol derecho y eliminar ese mínimo.
- Siempre se mantiene la regla del ABB:
    - izquierda < raíz < derecha.