📚 NOTES: Checking if a Binary Tree is a Max Heap

1. Max Heap ke liye 2 Conditions

1. **Heap Property** – Har parent node ka value apne dono children se **bada ya barabar** hona chahiye.
2. **Complete Binary Tree (CBT)** – Saare levels **poore fill hone chahiye** (except last level), aur last level me nodes **left se right order** me fill hone chahiye, beech me koi gap nahi hona chahiye.

2. Function 1: `size_of_tree(Node* root)`

Purpose

Binary tree me total number of nodes count karta hai.

Yeh recursive hai:

- Agar root `NULL` → size `0`.
- Warna size = `1 + size_of_tree(left) + size_of_tree(right)`.

Example:

```
      100
     /   \
    90    80
   / \    / \
  70 60  50  40
 /
30
```

Total size = `8`.

3. Function 2: `isCBT(Node* root)`

Purpose

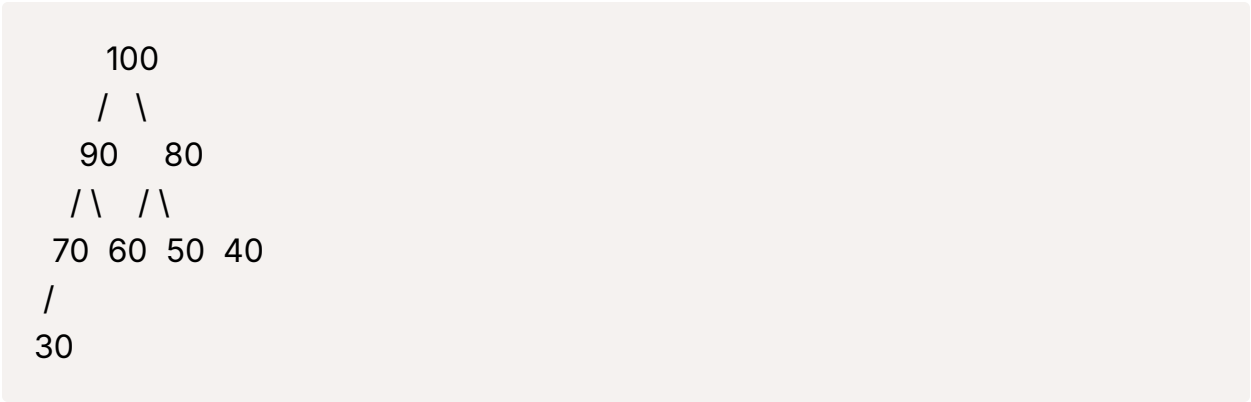
Check karta hai ki tree **Complete Binary Tree** hai ya nahi.

Logic

1. **Step 1:** Tree ka size nikal lo → `size`.
2. **Step 2:** Level Order Traversal ke liye queue use karo.
3. **Step 3:** Ek `count` variable rakho jo traverse huye nodes ka count kare.
4. **Step 4:** Jab tak `count < size`:
   - Queue se node nikalo.
   - `count++`.
   - Agar node `NULL` nahi hai:
     - Left aur Right push karo (chahe NULL ho ya value wala node).
5. **Step 5:** Jab `count == size`, ab queue me bache huye nodes check karo:
   - Agar koi **NULL nahi** mila → false (kyunki ek NULL ke baad koi value node aayi matlab gap hai → not CBT).
6. **Step 6:** Agar sab NULL → true.

Example Queue Flow (isCBT function ke liye)

Tree:



size = 8

Queue operations step-by-step:

- Start: `Q = [100]`, count = 0
- Pop 100 → push(90, 80) → Q = [90, 80], count=1
- Pop 90 → push(70, 60) → Q = [80, 70, 60], count=2
- Pop 80 → push(50, 40) → Q = [70, 60, 50, 40], count=3
- Pop 70 → push(30, NULL) → Q = [60, 50, 40, 30, NULL], count=4
- Pop 60 → push(NULL, NULL) → Q = [50, 40, 30, NULL, NULL, NULL], count=5
- Pop 50 → push(NULL, NULL) → Q = [40, 30, NULL, NULL, NULL, NULL, NULL], count=6
- Pop 40 → push(NULL, NULL) → Q = [30, NULL, NULL, NULL, NULL, NULL, NULL, NULL], count=7
- Pop 30 → push(NULL, NULL) → Q = [NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL], count=8

Ab count = size → ab bache huye sab NULL → ✅ **CBT**.

---

4. Function 3: `Max_heap_property(Node* root)`

Purpose

Check karta hai ki Max Heap property follow hoti hai ya nahi.

Logic

1. Agar node `NULL` hai → return true.
2. Agar left child hai aur parent < left → return false.
3. Agar right child hai aur parent < right → return false.
4. Warna, recursively left aur right subtree ke liye check karo.

Example:

```
   100
  /   \
 90    80
```

- 100 >= 90, 100 >= 80 → check children
- 90 >= children, 80 >= children → ✅.

---

5. Function 4: `is_max_heap(Node* root)`

Purpose

CBT aur Heap property dono satisfy ho to true return kare.

Logic

```
return Max_heap_property(root) && isCBT(root);
```

---

6. Output for Example Tree

```
isCBT: 1
Max Heap Property: 1
Is Max Heap: 1
```

✅ Summary Table

Condition	Function	Pass/Fail
Complete Binary Tree	`isCBT`	✅
Max Heap Property	`Max_heap_property`	✅
Both satisfied	`is_max_heap`	✅
