Solution: Lowest Common Ancestor of a Binary Tree III

Let’s solve the Lowest Common Ancestor of a Binary Tree III problem using the Two Pointers pattern.

We'll cover the following

Statement

You are given two nodes, `p` and `q`. The task is to return their lowest common ancestor (LCA). Both nodes have a reference to their parent node. The tree’s root is not provided; you must use the parent pointers to find the nodes’ common ancestor.

> **Note:** The lowest common ancestor of two nodes, `p` and `q`, is the lowest node in the binary tree, with both `p` and `q` as descendants.
>
> In a tree, a descendant of a node is any node reachable by following edges downward from that node, including the node itself.

Constraints:

- $-10^4 \leq$`Node.data` $\leq 10^4$
- The number of nodes in the tree is in the range $[2, 500]$.
- All `Node.data` are unique.
- `p != q`
- Both `p` and `q` are present in the tree.

Solution

We will use the two pointer approach to find the LCA of the given nodes in a binary tree in which each node has a reference to its parent.

The two-pointer algorithm begins by setting two pointers at the given nodes and moving them toward the roo
If a pointer reaches the top of the tree, it is reset to start from the other node. This resetting balances the paths ensuring that both pointers cover the same distance in total. The pointers continue moving until they me the

meeting point is the lowest common ancestor.Let's go through the algorithm to see how we will reach the solution:

- Start with the two pointers, `ptr1` at node `p` and `ptr2` at node `q`.
- Move both pointers to their respective parent nodes:
  - If a pointer reaches the root, reset it to the starting position of the other node.
  - For example, if `ptr1` reaches root, set it to node `q`, and similarly, if `ptr2` reaches root, set it to node `p`.
- Continue this process until these two pointers meet. Return the node at which they meet as this is LCA of the given nodes.

Why reset the pointers?

- **Balancing the distance:** Imagine both pointers racing to the tree's root. If one starts higher up than the other, it will reach the top sooner. Resetting the faster pointer to start from where the slower one began gives the faster pointer a chance to *catch up* by traveling the same distance the slower one did.
- **Ensuring they meet:** This way, both pointers cover the same ground and eventually meet at the first common point they reach, the LCA.

Let's look at the following illustration to get a better understanding of the solution: