*Owen Lindsey*

*Professor Demland, David*

*CST-201 Exercise 9*

*11/03/2024*

*Exercise 5.3 - 8*

*Algorithm Explanation*

1. **Take two arrays as input:**

* Inorder traversal: 9, 3, 1, 0, 4, 2, 7, 6, 8, 5
* Postorder traversal: 9, 1, 4, 0, 3, 6, 7, 5, 8, 2

1. **Base Rules for Building Tree:**

* Last element of postorder is always the root node.
* In inorder, elements before root form left subtree.
* In inorder, elements after root form right subtree.

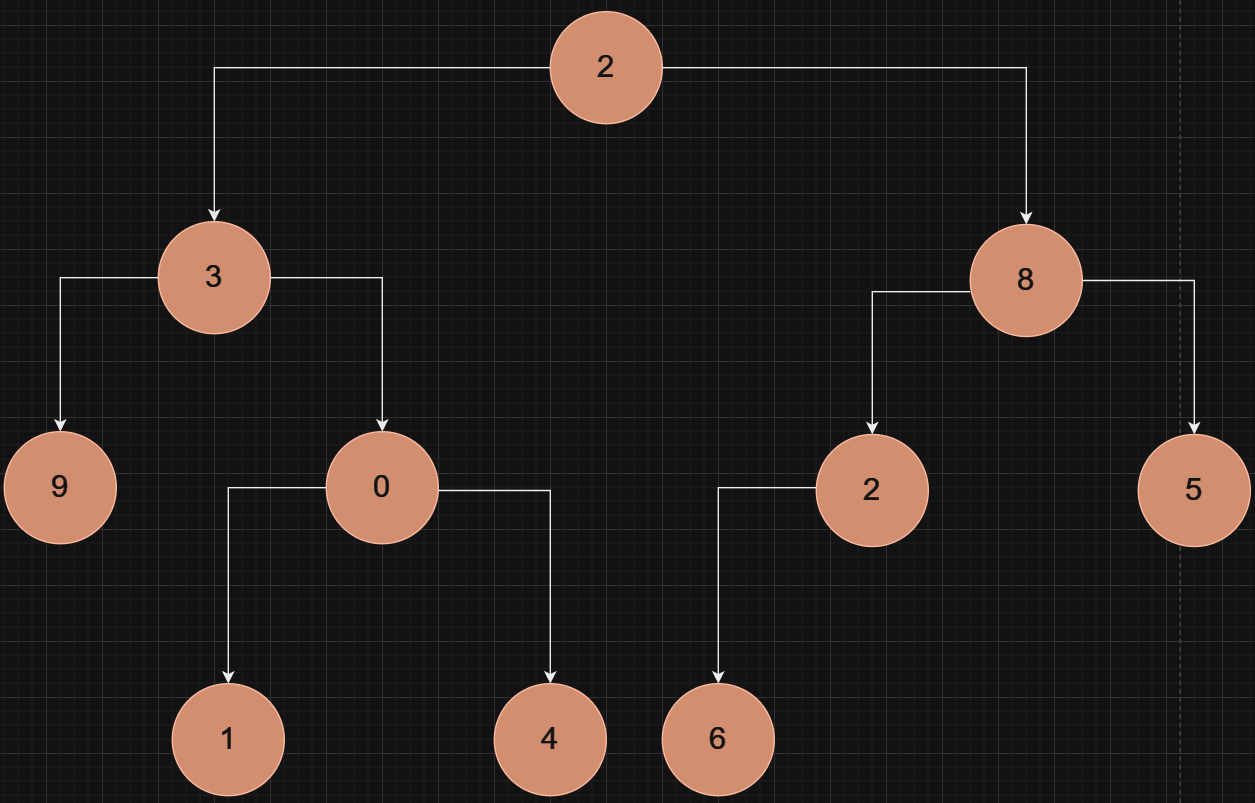
1. **Tree Construction Process:** Identify root (2) from postorder's last element. Then split inorder around root:

* Left subtree: [9, 3, 1, 0, 4]
* Right subtree: [7, 6, 8, 5] 3. For left subtree:
* Find its root (3) from postorder.
* Split remaining elements around 3.
* Build connections: 3 → 9, 3 → 0 4. For right subtree:
* Find its root (8) from postorder.
* Split remaining elements around 8.
* Build connections: 8 → 7, 8 → 5 5. Continue recursively until all nodes are placed

1. **Verification Process:**  Follow inorder traversal rules:

* Visit left subtree.
* Visit root.
* Visit right subtree.
* Should get: 9, 3, 1, 0, 4, 2, 7, 6, 8, 5 2. Follow postorder traversal rules:
* Visit left subtree.
* Visit right subtree.
* Visit root.
* Should get: 9, 1, 4, 0, 3, 6, 7, 5, 8, 2

1. **Tree Structure:**



1. ***Key Conditions:***

*1. Tree must be binary (max 2 children per node).*

*2. No requirement for node value ordering.*

*3. Solution must satisfy both traversal sequences.*

*4. All nodes must be used exactly once.*

*Exercise 5.5 - 1*

*Algorithm Explanation*

1. ***Divide-and-Conquer Approach:***

* *Sort array first: O(n log n).*
* *Divide array into halves recursively.*
* *Return minimum of:* 
  + *Closest pair in left half.*
  + *Closest pair in right half.*
  + *Pair across the middle.*

1. ***Time Complexity:***

* *Initial sort: O(n log n).*
* *Recurrence: T(n) = 2T(n/2) + O(1).*
* *Overall complexity: O(n log n).*

1. ***Is it a Good Algorithm?*** *No, because:*

* *A simple solution exists sort array and compare adjacent elements.*
* *Simple solution is still O(n log n) but more efficient in practice.*
* *Divide-and-conquer adds unnecessary overhead.*
* *No benefit gained from recursive approach.*

1. ***Better Solution:***

* *Sort array once.*
* *Single linear scan comparing adjacent elements.*
* *Same complexity but simpler and faster in practice.*