Complexity Analysis of searchTree

1. searchTree Function

The searchTree function processes user inputs iteratively and performs operations to find and display details about a node in a tree. At each iteration, it calls two key methods:

- searchNode: Finds a node based on its identifier.
- searchNodeE: Generates a detailed report about the node, including its type and parent operation.

The complexity of searchTree depends on the combined cost of these operations.

2. searchNode Method

searchNode searches for a node in the tree by comparing its identifier. The method checks two maps:

- nameMap: Stores nodes indexed by name.
- nodeMap: Stores nodes indexed by ID.

Both maps require traversing the tree to build, and lookups are proportional to the size of the tree:

Time Complexity: O(N), where N is the total number of nodes in the tree.

3. searchNodeE Method

searchNodeE generates a report about the target node and its parent operation. The process involves:

1. Retrieving node details like type or quantity, which is a constant-time operation.
2. Identifying the parent operation using findParentOperation, which traverses the tree recursively.

findParentOperation explores all child nodes at each level until the target node is located. In the worst case, it visits every node in the tree.

Time Complexity: O(N), where N is the total number of nodes.

Overall Complexity of searchNodeE: Dominated by findParentOperation, resulting in O(N).

4. searchTree Overall Complexity

The overall complexity of searchTree is determined by its two primary operations:

- searchNode: O(N)

- searchNodeE: O(N)

For each user input, the complexity is:

O(N) + O(N) = O(N).

If the user provides M inputs, the total complexity becomes:

O(M * N), where:

- M = Number of user inputs.

- N = Total number of nodes in the tree.

5. Space Complexity

The space complexity of searchTree is influenced by recursion in findParentOperation, which depends on the depth of the tree (D):

- Space Complexity: O(D), where D is the maximum depth of the tree.

For balanced trees, D is proportional to log(N). For skewed trees, D is proportional to N.

6. Summary Table

Operation	Time Complexity	Space Complexity
searchTree	O(M * N)	O(D)