Question statement (6):

Given pointers to two nodes a and b in a rooted binary tree (not BST) where every node has a pointer to its parent, determine whether the two nodes have a common ancestor besides the root.

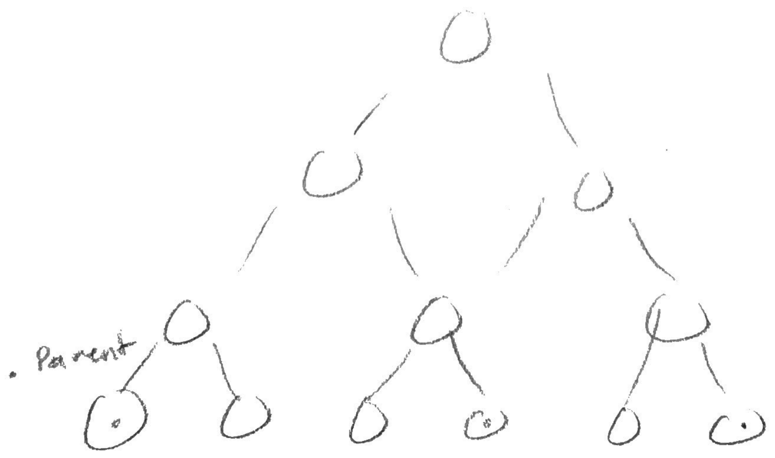
You can give the candidate a prototype of the function like:

```
boolean haveCommonAncestorsBesideRoot(
            Node node1, Node node2, Node root) {
   ...
}
```

Example

```
        1
      /   \
     2     3
    / \   /  \
   4   5 6    7
```

```
haveCommonAncestorsBesideRoot(4, 5, 1) -> true
haveCommonAncestorsBesideRoot(4, 6, 1) -> false
```



(compare nodes)

Question 6

I dentify the prob (what's going?)

•) we need to find out roots of 2 nodes

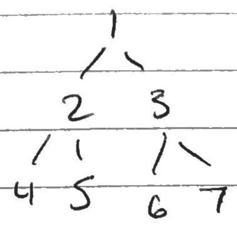
D efine the Goal

•) ~~the~~ we have to find out if the 2 nodes ~~are~~ have the same root i.e., are in the same subtree

sounds like a dynamic programming prob!

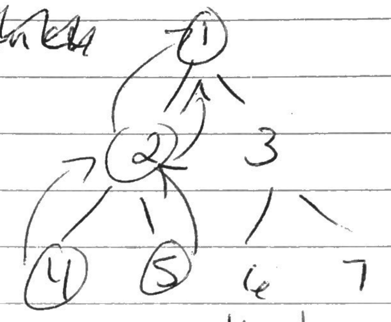
E - xplore / A - nticipate - ct through Duke's Approach

1) workout E.g.s by hand



//save all subtree of 1st node then check if 2nd node is inside of ~~that~~ the subtree of 1st haveCommonAncestorBesidesRoot (4, 5, 1)

~~Ancest~~



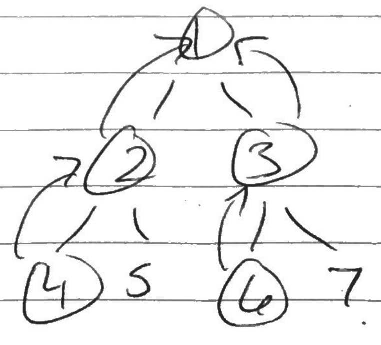
//helpper method
ancestorsOf(4) → 2, 1

ancestorsOf(5) → 2, 1

// look ~~if~~ the ^any one of^ numbers of
ancestersOf(4) are
contained in ancestorsOf(5)

return true

haveCommonAncesterBesidesRoot(4,6,1)



// looking for like elements,
returned 1, which is root

return false

2) write what I did

- •) Sub Prob 1 : ~~Looking~~ gathering all ancesters of a given node

- •) Sub Prob 2: seeing if 2 nodes have same node aside from root