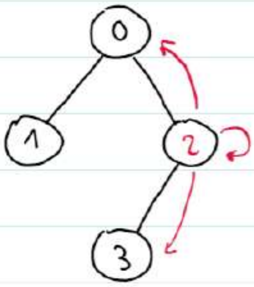
968. Binary Tree Cameras

Rules: A camera at a node covers:
- The node itself
- Its parent
- Its two immediate children

Goal: Minimize the amount of cameras needed to cover entire tree

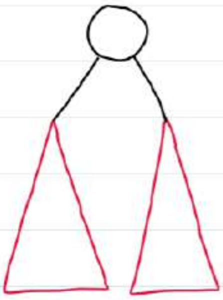
Example:



A camera on node 2 could cover nodes: 0, 2 and 3

Cases: For any given node (A.K.A. root)

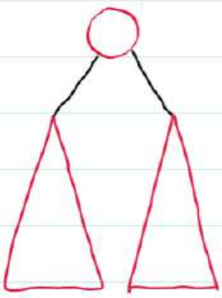
0:



Both left and right subtrees are covered except root node

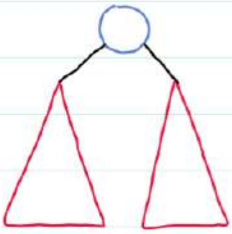
All nodes below current are covered

1:



Entire subtree is covered including its root

2:



All nodes are covered plus root has a camera

Base: null nodes are considered as covered and can't have cameras on them.

So leaf nodes can meet any of three cases above.

968. Binary Tree Cameras

Posible States:

$\forall$ u node, we define f(u) that returns d[3] :

d[0]: Min cameras needed to cover entire subtree excluding u

d[1]: Min cameras needed to cover entire subtree including u with no camera in u

d[2]: Min cameras needed to cover entire subtree with a camera on u

Recurrence

L = f(u.left)   R = f(u.right)

dp[0] = L[1] + R[1]   Children don't have camera

dp[1] = min(L[1]+R[2], L[2]+R[2], L[2]+R[2]) At least 1 camera

dp[2] = 1 + min(L) + min(R)   Min of each

Base Case:

$f(\text{null}) = (0, 0, \infty)$

Cameras not needed for a nonexistent node
A camera cannot be put on a null node

Final Result:

dp = f(root) => min(dp[1], dp[2])

Root must be covered, thus dp[0] cannot be used