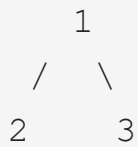


Min distance between two given nodes of a Binary Tree

Given a binary tree with **n** nodes and two node values, **a** and **b**, your task is to find the minimum distance between them. The given two nodes are guaranteed to be in the binary tree and all node values are **unique**.

Example 1:

Input:



a = 2, b = 3

Output:

2

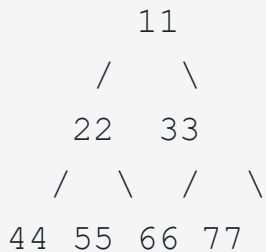
Explanation:

We need the distance between 2 and 3. Being at node 2, we need to take two steps ahead in order to reach node 3.

The path followed will be: 2 -> 1 -> 3. Hence, the result is 2.

Example 2:

Input:



a = 77, b = 22

Output:

3

Explanation:

We need the distance between 77 and 22. Being at node 77, we need to take three steps ahead in order to reach node 22. The path followed will be: 77 -> 33 -> 11 -> 22. Hence, the result is 3.

Your Task:

You don't need to read input or print anything. Your task is to complete the function **findDist()** which takes the **root** node of the tree and the two node values **a** and **b** as input parameters and returns the minimum distance between the nodes represented by the two given node values.

Expected Time Complexity: $O(n)$.

Expected Auxiliary Space: $O(\text{Height of the Tree})$.

Constraints:

$2 \leq n \leq 10^5$

$1 \leq \text{Data of a node} \leq 10^9$