EDITORIAL

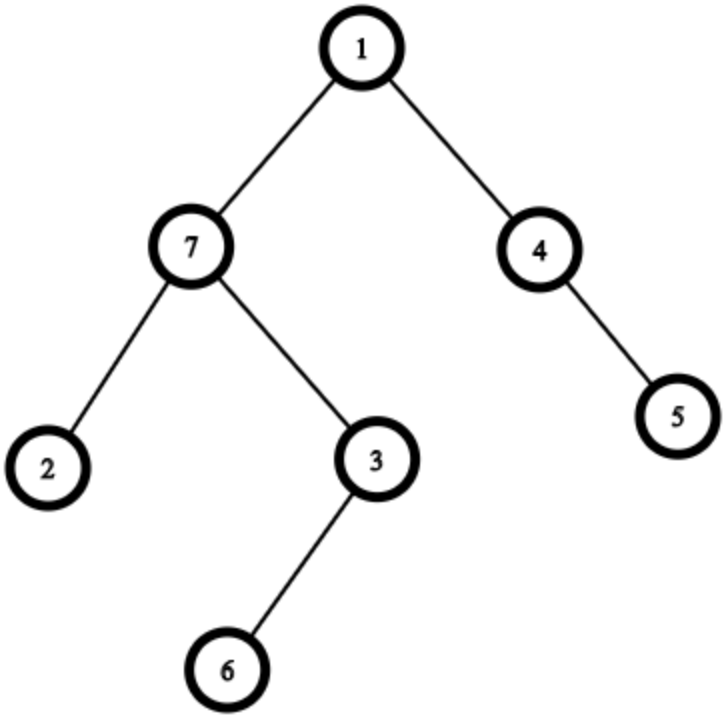
Choosing Subtree is Fun

Problem Link

Prerequisite - DFS , LCA , Basic Idea of Two Pointer Algorithm

This was one very good problem on LCA. I will recommend you to read the codeforces editorial of the problem and try to think and implement first before reading further .
Link to Editorial

Detailed Explanation - OK. I hope you have read the editorial. Now we shall understand the solution proposed there. First and foremost, we should observe that if two nodes indexed x and y are included in the subtree, then all the nodes on the simple path between x and y must be included. Consider the following tree -



Now suppose we include 3 and 4 in the subtree. Then clearly nodes 1 and 7 must be included.
Now how to approach the solution??
We can use the idea of two pointer algorithm which will help us to consider an interval [L , R] . Now we have to think how to add a vertex R+1 and remove a vertex L in less than O(n) time. An important thing to notice here is that we cannot maintain the whole subtree in a set as this will time out. What we need to do is to maintain the end points in the set and use it . Basically, we want to keep track of size of subtree only and not the whole subtree.

We can do this by storing the dfs order of the vertices. Like we maintain the set of some vertices in a STL set sorted on the basis of their arrival in DFS. To add some vertex x, we find the vertex present in the set with arrial time just less than that of current vertex x. Let this be prev. Similarly, we find a vertex in the set with arrival time just greater than x. Let this be next.
So the size of our set will be
size=size + (dist(x,next) + dist(x,prev) - dist(prev,next))/2
Similarly for the erase
size=size + (dist(prev,next) - dist(x,next) - dist(x,prev))/2

See code for better understanding.
Code