Solution for Stations

Let us refer to the station with label L given by Ali as node L. The SIB network forms a tree. Suppose we root the tree at the station with the smallest label. We observe that if Brenda can, given 2 nodes A and B, determine whether

- node A is an ancestor of node B
- node B is an ancestor of node A
- nodes A and B are not ancestors of each other

then Brenda can solve each query as follows:

1. For the source node S, check each neighbouring node to see if it is an ancestor of node S.
    - If it is, it must be the parent of node S, which we shall call node P. There can be at most 1 such node.
    - Otherwise, it is a child of node S. Check if it is an ancestor of the destination node T. If yes, return this child.
2. If no answer is returned by the steps above, either node T is an ancestor of node S or they are not ancestors of each other. In either case, the answer is node P.

Hence, if Ali can label the stations such that Brenda can check the above just from the labels of the station, then Brenda can solve all the queries.

Subtask 1

The network is a line. Ali can find a leaf station and label the stations 0 to $N-1$ in DFS order in $O(N)$ time. Then, node A is an ancestor of node B if and only if $A \leq B$, which can be checked in $O(1)$ time.

Subtask 2

The network is a binary tree. For each station X in the input, Ali can give the label $X+1$. That way, the parent of node L is node $\lfloor L/2 \rfloor$. Hence, node A is an ancestor of node B if and only if $A \leq B$ and repeatedly applying $B \to \lfloor B/2 \rfloor$ reaches A before reaching 0. This takes $O(\log N)$ time per check.

Subtask 3

Ali can give label 0 to the station with degree more than 2, or to any station if no such station exists, and label its neighbours 1 to M where M is the degree of node 0. If we remove node 0, the

remaining stations will form disjoint lines where one station in each line is adjacent to node 0. For each line, let the station adjacent to node 0 be node L. Ali can label each station X in the line with $(d(L, X) \times 1000 + L)$ where $d(L, X)$ is the number of edges in the unique path between station X and node L. After this, node A is an ancestor of node B if and only if $A \leq B$ and either $A == 0$ or $(A\%1000 == B\%1000)$ which takes $O(1)$ time to check.

Subtask 4

The network is small with at most 8 stations. Select a random station and label it 1, this will be the root of the tree. For each node L with M children, label it's children $(L \times 8)$ to $(L \times 8 + M - 1)$. That way, the parent of node L is node $\lfloor L/8 \rfloor$. Hence, node A is an ancestor of node B if and only if $A \leq B$ and repeatedly applying $B \to \lfloor B/8 \rfloor$ reaches A before reaching 0. The maximum possible label is $8^7 \leq 10^9$.

Subtask 5

In the general case, select a random station to root the tree and perform a DFS from the root:

```
counter = 0
dfs(station v):
    in[v] = counter++
    call dfs(c) for each child c of v
    out[v] = counter++
call dfs(root)
```

Suppose Brenda knows $in[v]$ and $out[v]$ for the source, destination and the neighbours of the source. For any 2 stations v_1 and v_2, she can deduce that v_1 is an ancestor of v_2 if and only if $in[v_1] < in[v_2] < out[v_1]$, and thus be able to solve the query using the method described in the previous section. One simple way to encode this is that Ali can label each station v with $L = in[v] \times 1000 + out[v]$, so Brenda can get $in[v] = L/1000$ and $out[v] = L\%1000$. The maximum label will be $999 \times 1000 + 999 = 999999$ which will get around 65 points if the first 4 subtasks are handled separately.

To increase the points further, we can make a few observations (*)

- Brenda can find the parent of the source station s by looking for the neighbouring station v with the smallest $in[v]$ or $out[v]$, unless $in[s] == 0$ in which case station s is the root.
- If we sort the m children $c[]$ of station s in ascending order of $in[v]$ to get $c_1, c_2, \ldots, c_m$, then $out[c_1] = in[c_2] - 1, out[c_2] = in[c_3] - 1$ and so on. Also, $in[s] = in[c_1] - 1$ and $out[c_m] = out[s] - 1$.

Hence, some minor optimisations can be made to get $70 - 80$ points by not encoding some values of $out[v]$ and deriving them from the $in[v]$ of the other neighbours instead.

In order to get 89points, we need Ali to use a maximum label of 2000 or $2n$. We label each station v

as follows:

- If the number of edges in the unique path from station v to the root is even, label the station with $in[v]$.
- Otherwise, label the station with $out[v]$

For Brenda, there are 2 possible cases to consider:

- If the label of the source station s is smaller than the label of its neighbours, then the label of station s is $in[s]$ and we can use the observations in (*) to find the parent of station s (if station s is not the root) and $in[v]$ and $out[v]$ for all its children v and be able to solve the problem as described above.
- Otherwise, the label of station s is $out[s]$ and we can use the same idea to also find the parent of station s (if station s is not the root) and $in[v]$ and $out[v]$ for all its children and do the same.

To get the full 100 points, we make a final observation that the exact values of the labels are not essential since the 89 point solution only needs to know how each label compares with each other. Since the labels are distinct, we can simply label the stations 0 to $N-1$ based on the rank of the labels in the 89 point solution.