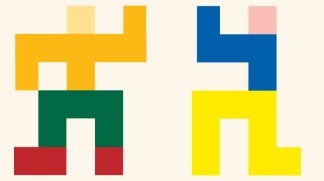


RECONSTRUCT



Statement:

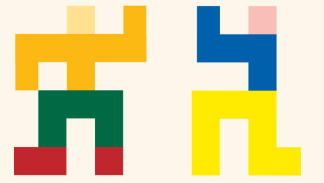
There is a hidden tree with N vertices. For each vertex there is also a hidden preorder starting from it (a preorder is a dfs-ordering of the vertices). We are allowed to ask queries about the j -th vertex in the i -th preorder. The aim is to find the tree with minimum number of queries ($\leq 3N$ for 100 points).

Constraints:

$$N \leq 2^{16} + 1$$



RECONSTRUCT

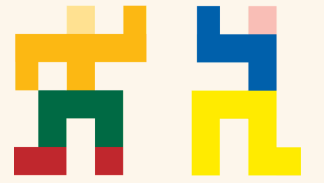


Solution for $O(N^2)$ queries - 57

- 1)** We start by querying one whole preorder (say from vertex 0).
- 2)** We iterate through the vertices from $\text{guess}(0,1)$ to $\text{guess}(0,n-1)$ and for each one we find an edge connecting it to the vertices before it in the preorder (to its parent if we root the tree at 0).
- 3)** Let $v = \text{guess}(0,i)$. We investigate v 's preorder again starting from $\text{guess}(v,1)$ until we find a vertex that we already processed in **2)**. This is v 's parent.



RECONSTRUCT



Solution for $O(N \log N)$ queries - 85

We can find the tree by repeatedly connecting subtrees we have already found. Initially all subtrees are the single vertices. Each preorder will start with vertices $a_1 a_2 \dots$ from a given subtree A followed by a vertex b_i from subtree B .

Claim: there will be two preorders of the form:

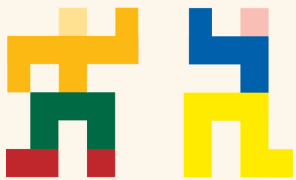
$a_1 a_2 a_3 \dots a_n b_i$

$b_1 b_2 b_3 \dots b_m a_j$

From them we can conclude that there is an edge between a_j and $b_i \rightarrow$ we can connect the two subtrees.



RECONSTRUCT



Solution for $O(N \log N)$ queries - 85

0 1 2 3 5 4
1 2 0 3 4 5 *
2 1 0 3 5 4 *
3 5 4 1 2 0 **
4 3 5 1 0 2
5 3 4 1 2 0 **

Found edges:
1 - 2
3 - 5

0 1 2 3 5 4 *
1 2 0 3 4 5 *
2 1 0 3 5 4
3 5 4 1 2 0
4 3 5 1 0 2 **
5 3 4 1 2 0 **

Found edges:
0 - 1
3 - 4

0 1 2 3 5 4 *
1 2 0 3 4 5
2 1 0 3 5 4
3 5 4 1 2 0 *
4 3 5 1 0 2
5 3 4 1 2 0

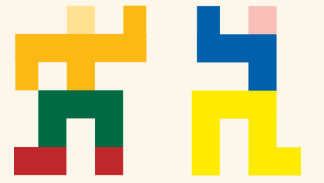
Found edges:
1 - 3

Implementation:

We keep prefixes beginning only with unique subtrees and when merging two subtrees we discard the shorter prefix. This way we achieve an optimal solution for $N \log N$ queries (similar to small to large idea). For keeping track of subtrees we can use DSU/union-find data structure.



RECONSTRUCT

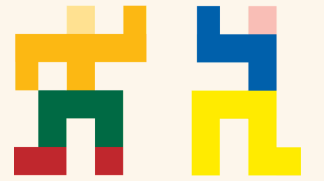


Solution for 3N queries - 100

- 1) We start by querying one whole preorder (say from vertex 0).
- 2) We iterate through the vertices from last to first: $\text{guess}(0, n-1)$ to $\text{guess}(0, 1)$, and for each one we find an edge connecting it to the vertices before it in the preorder (to its parent if we root the tree at 0).
- 3) Let $v = \text{guess}(0, i)$. We investigate v 's preorder starting from $\text{guess}(v, 1)$. There are two possibilities:
 - A) $\text{guess}(v, 1)$ is a child of v - we already know the whole subtree for that child so we can skip directly to $\text{guess}(v, 1 + \text{size_subtree})$
 - B) $\text{guess}(v, 1)$ is the parent of v



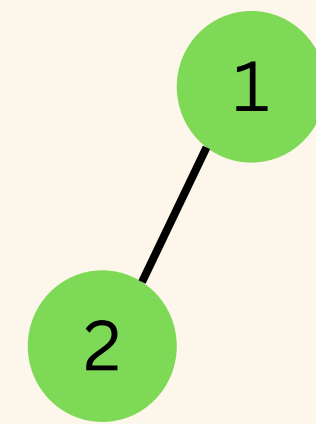
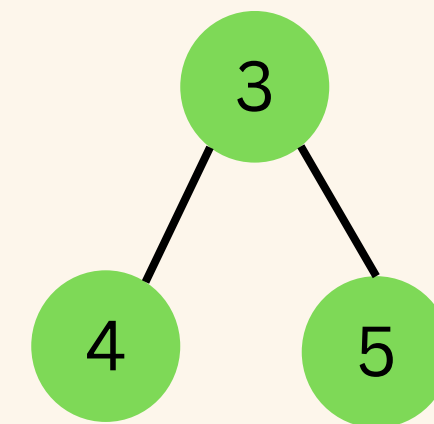
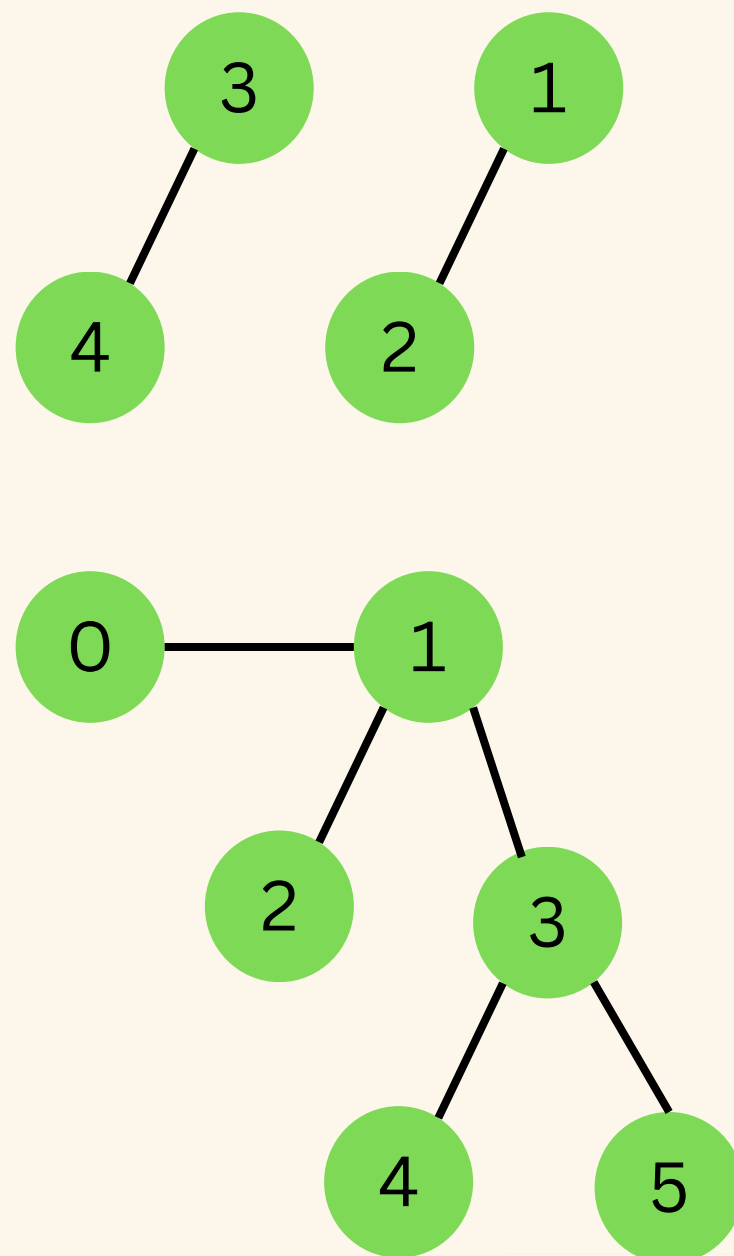
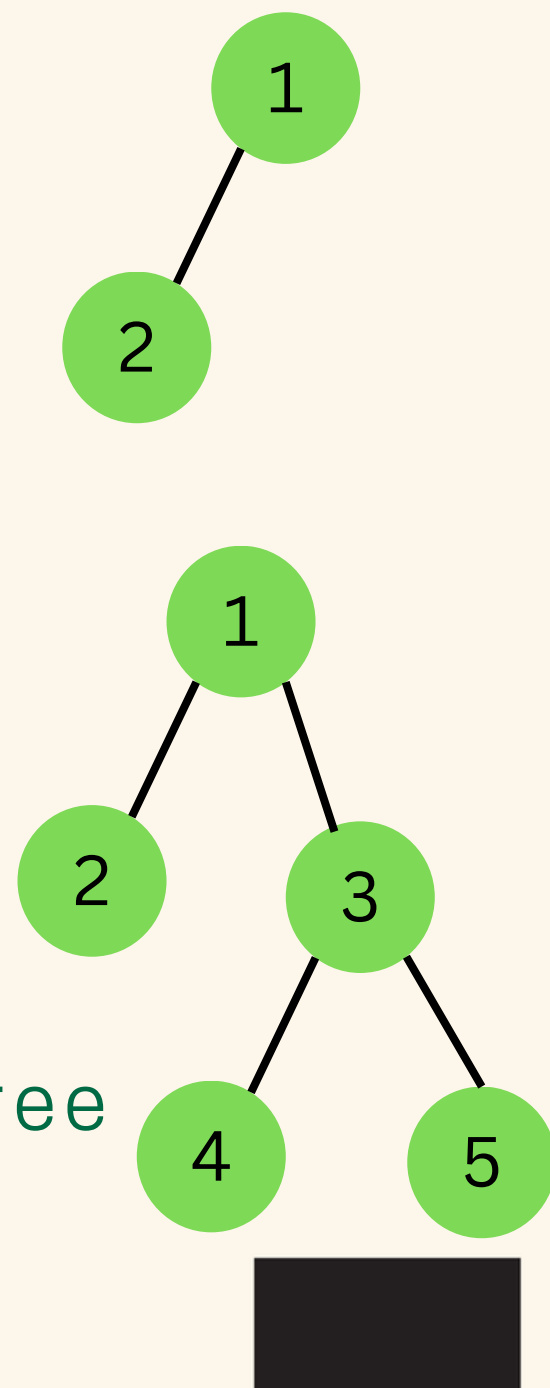
RECONSTRUCT



Solution for 3N queries:

0	1	3	5	4	2
1	3	4	5	2	0
2	1	0	3	5	4
3	5	4	1	2	0
4	3	5	1	0	2
5	3	4	1	2	0

- parent
- child
- skipped subtree



N queries for a full
preorder, 2 queries for
each tree edge: $N + 2(N-1)$
 $= 3N-2$ queries