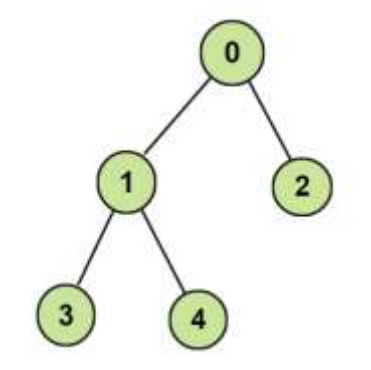
Tree Transformation

You are given a tree containing **N** nodes in the form of an array **P** where $\mathbf{P_i}$ represents the parent of the *i*-th node and $\mathbf{P_0}$ = -1 as the tree is rooted at node 0. In one move, you can merge any two adjacent nodes. Calculate the minimum number of moves required to turn the tree into a **star** tree.

-> Merging adjacent nodes means deleting the edge between them and considering both the nodes as a single one.
-> A Star tree is a tree with a center node, and all other nodes are connected to the center node only.

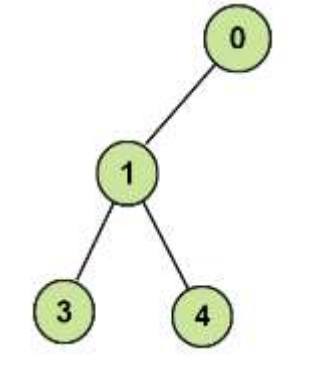
Example 1:

```
Input:
N = 5
p[] = {-1, 0, 0, 1, 1}
Output:
1
Explanation:
Tree looks like:
```



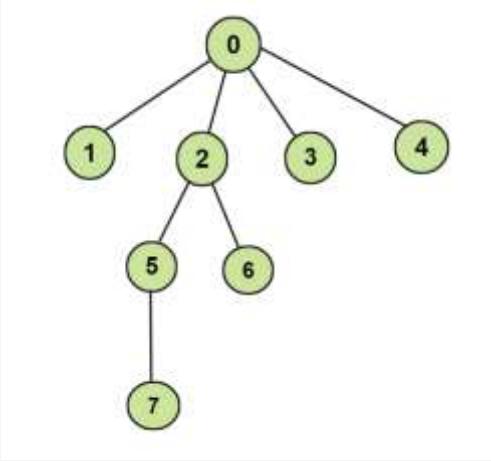

```
Merge the edge 0 - 2 in one operation

Our Tree will look like:
```

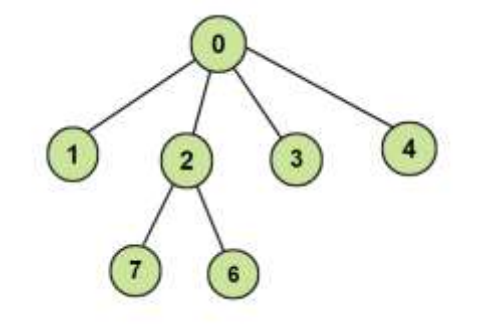


Example 2:

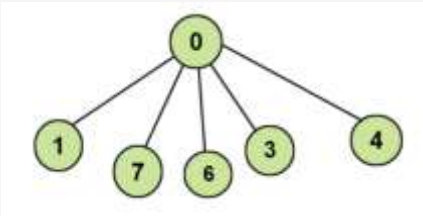
Input: N = 8
p[] = {-1 0 0 0 0 2 2 5}
Output:
2
Explanation:
Tree looks like:



Merge node 5 to 2, tree will look like



and then 2 to 0, finally the tree will be:



thus tree formed will be a star tree.

Your Task:
You don't need to read, input, or print anything. Your task is to complete the function ***solve( ),*** which takes integer **N,** and an array **p[ ]** as input parameters and returns the minimum number of moves required to turn the tree into a **star** tree.

Expected Time Complexity: O(N)
Expected Auxiliary Space: O(N)

Constraints:
$1 \leq N \leq 10^5$
$0 \leq p[i] < N$
p[0] = -1, 0 is the root node.