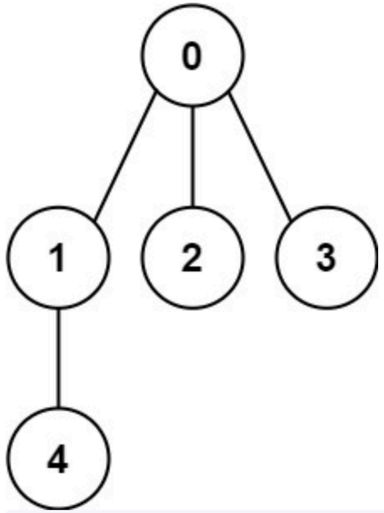
261. Graph Valid Tree

Medium 2528 68 Add to List Share

You have a graph of `n` nodes labeled from `0` to `n - 1`. You are given an integer n and a list of `edges` where `edges[i] = [a_i, b_i]` indicates that there is an undirected edge between nodes `a_i` and `b_i` in the graph.

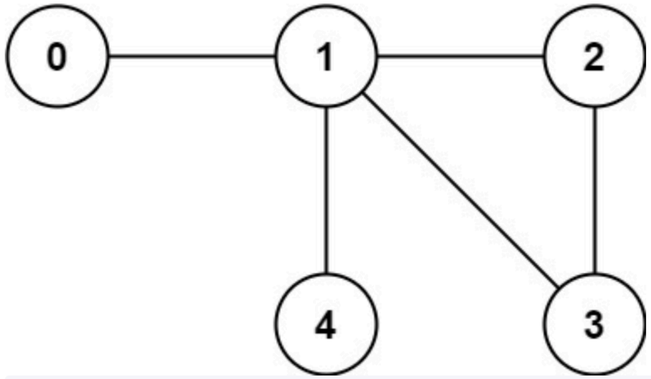
Return `true` *if the edges of the given graph make up a valid tree, and* `false` *otherwise*.

Example 1:




```
Input: n = 5, edges = [[0,1],[0,2],[0,3],[1,4]]
Output: true
```

Example 2:

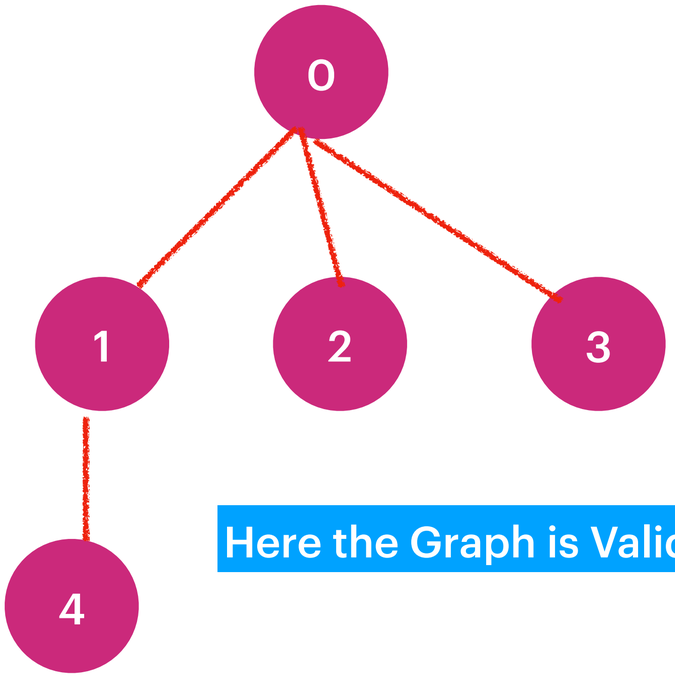



```
Input: n = 5, edges = [[0,1],[1,2],[2,3],[1,3],[1,4]]
Output: false
```

Constraints:

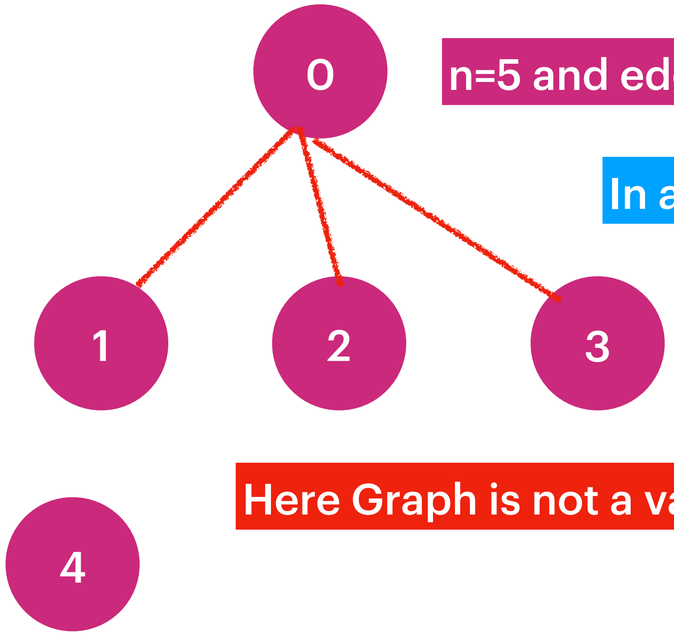
- `1 <= n <= 2000`
- `0 <= edges.length <= 5000`
- `edges[i].length == 2`
- `0 <= a_i, b_i < n`
- `a_i != b_i`
- There are no self-loops or repeated edges.

[[0,1],[0,2],[0,3],[1,4]]



Tree Properties :
1. In a Tree the child has only one immediate Parent.
1. In a Tree If we start from root , we can visit every Node.

Here the Graph is Valid Tree.

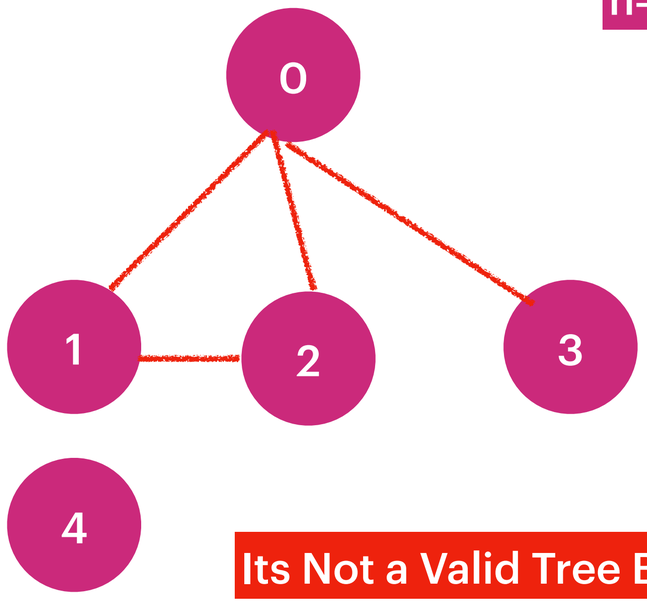


n=5 and edges = [[0,1],[0,2],[0,3]]

In a valid Tree no.of edges == n-1

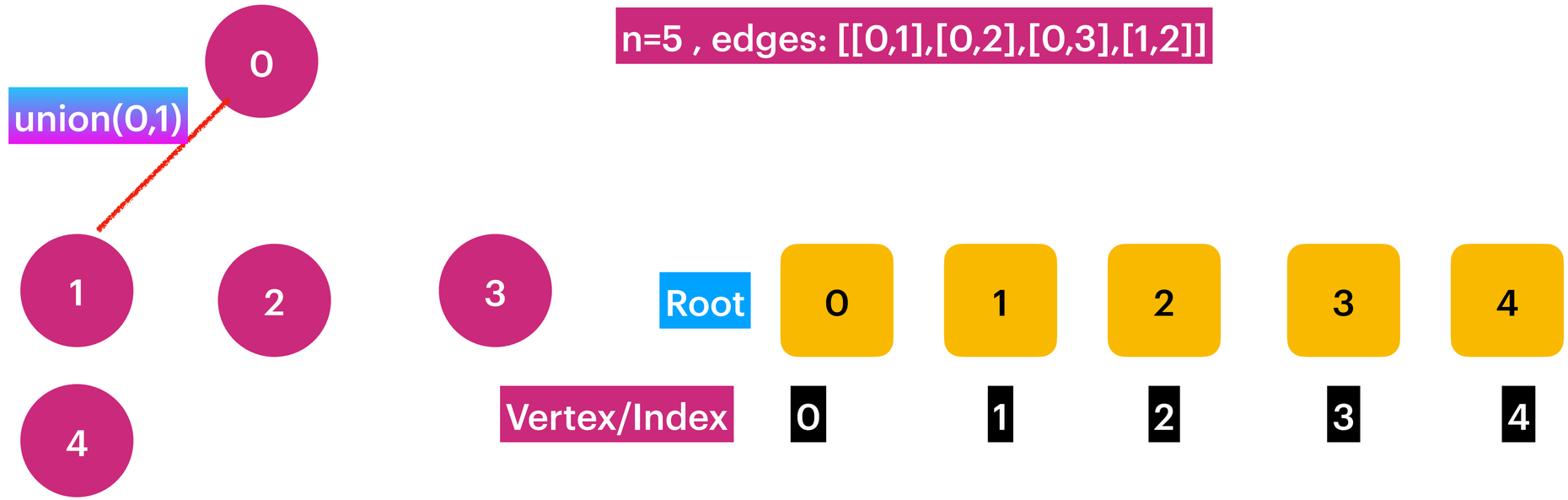
Here Graph is not a valid Tree : because no.of edges were 3, expected : 4

n=5 , edges: [[0,1],[0,2],[0,3],[1,2]]

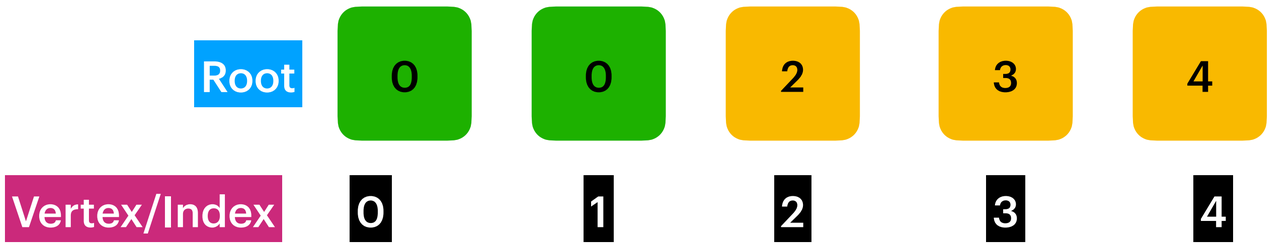


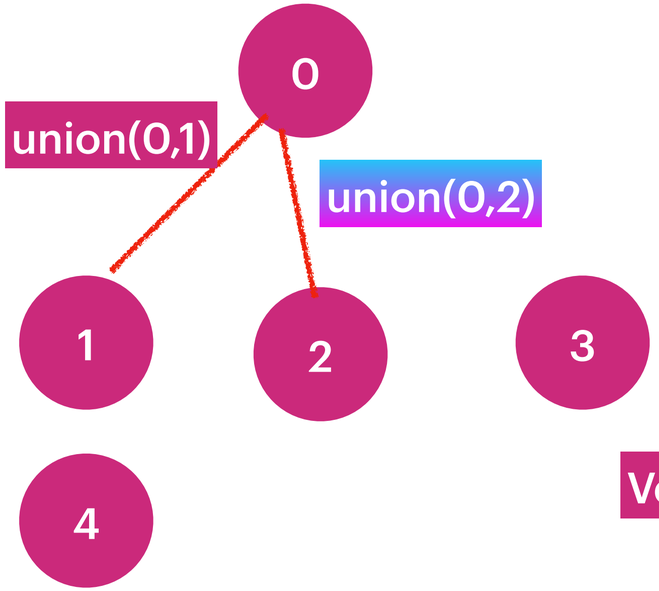
Tree Properties :
1. In a Tree the child has only one immediate Parent.
1. In a Tree If we start from root , we can visit every Node.

Its Not a Valid Tree Because node(2) is having two parents node(0), node(1)



union(0,1)
find(0) -> returns root of '0' = 0
find(1) -> returns root of '1' = 1



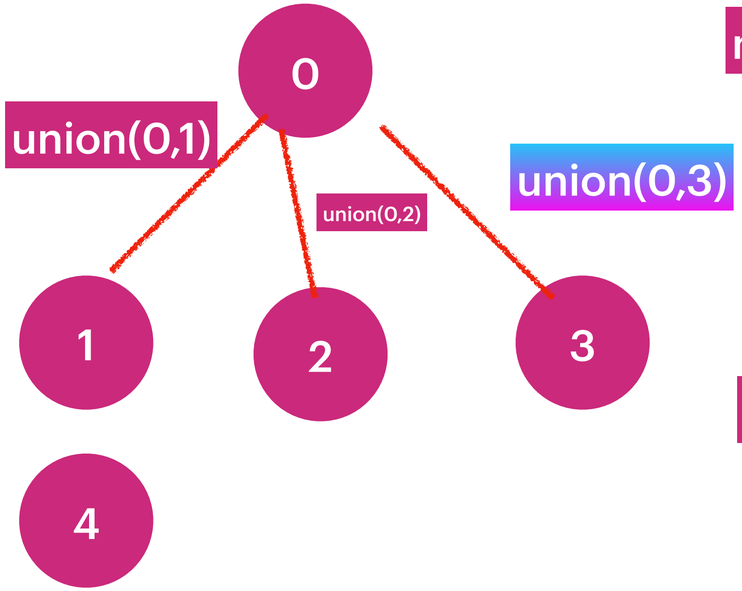


n=5 , edges: [[0,1],[0,2],[0,3],[1,2]]

Root	0	0	2	3	4
Vertex/Index	0	1	2	3	4

union(0,2)
find(0) -> returns root of ‘0’ = 0
find(2) -> returns root of ‘2’ = 2

Root	0	0	0	3	4
Vertex/Index	0	1	2	3	4

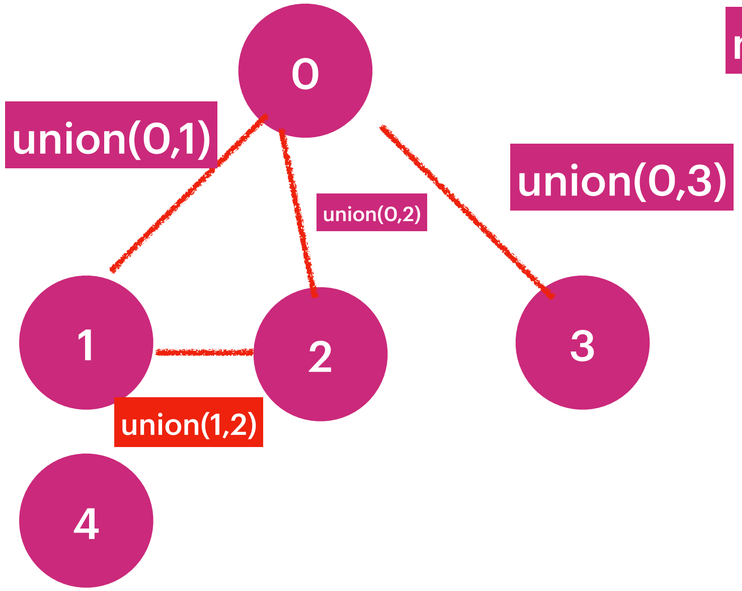


n=5 , edges: [[0,1],[0,2],[0,3],[1,2]]

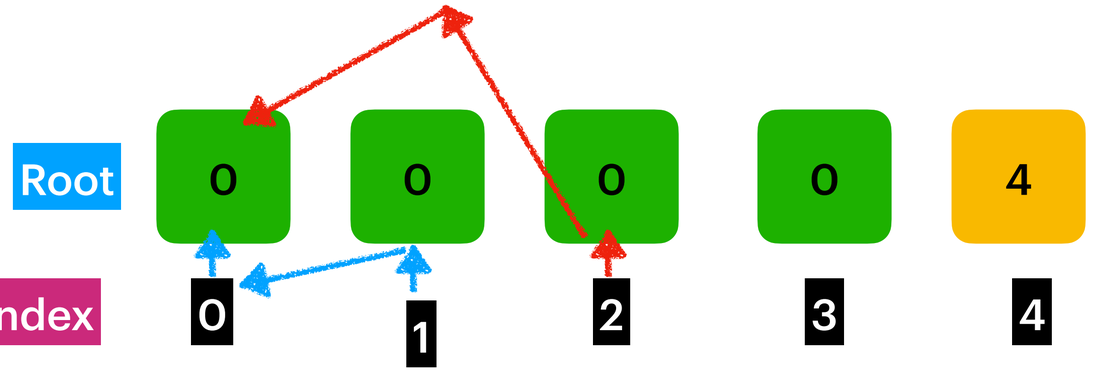
Root	0	0	0	3	4
Vertex/Index	0	1	2	3	4

union(0,3)
find(0) -> returns root of ‘0’ = 0
find(3) -> returns root of ‘3’ = 3

Root	0	0	0	0	4
Vertex/Index	0	1	2	3	4



n=5 , edges: [[0,1],[0,2],[0,3],[1,2]]



union(1,2)
find(1) -> returns root of '0' = 0
find(2) -> returns root of '0' = 0

V(1) , & V(2) were are already connected because root is same[0] so return False

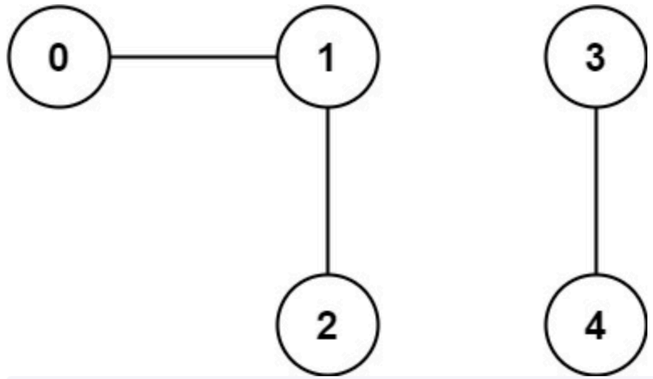
323. Number of Connected Components in an Undirected Graph

Medium 2067 70 Add to List Share

You have a graph of `n` nodes. You are given an integer `n` and an array `edges` where `edges[i] = [a_i, b_i]` indicates that there is an edge between `a_i` and `b_i` in the graph.

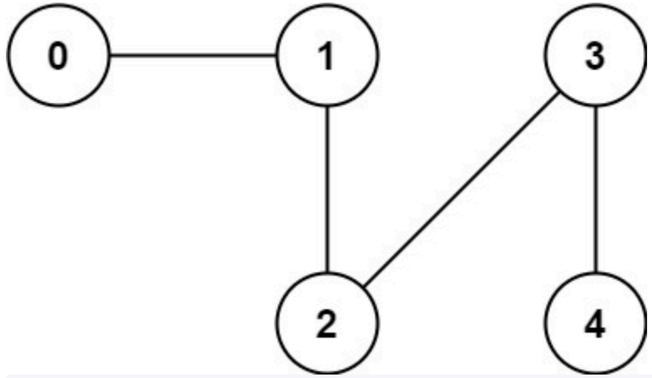
Return *the number of connected components in the graph*.

Example 1:




```
Input: n = 5, edges = [[0,1],[1,2],[3,4]]
Output: 2
```

Example 2:

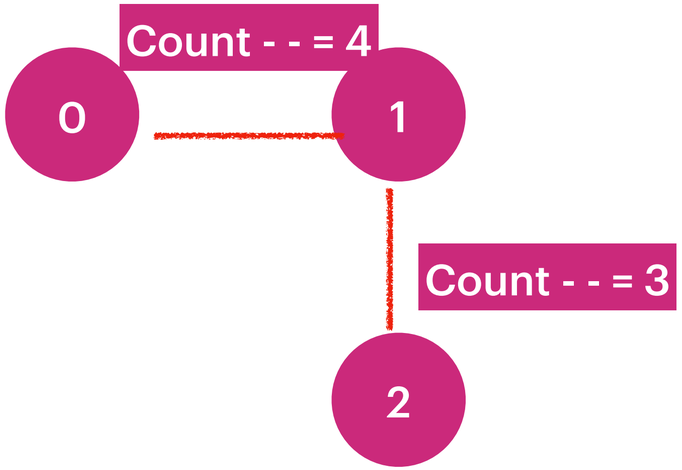


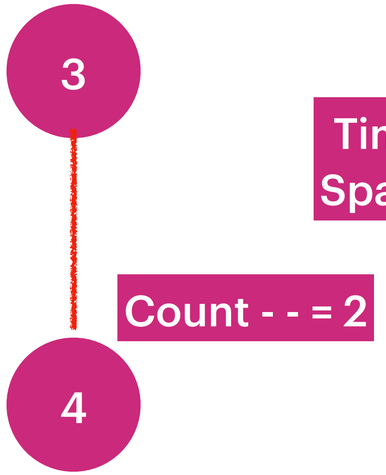

```
Input: n = 5, edges = [[0,1],[1,2],[2,3],[3,4]]
Output: 1
```

Constraints:

- `1 <= n <= 2000`
- `1 <= edges.length <= 5000`
- `edges[i].length == 2`
- `0 <= a_i <= b_i < n`
- `a_i != b_i`
- There are no repeated edges.

count=5

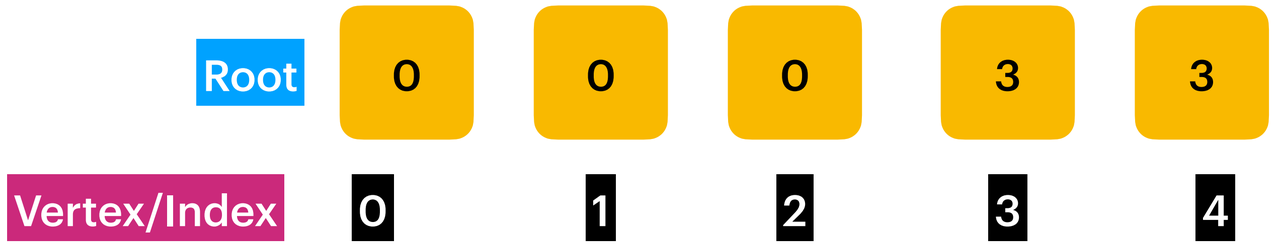




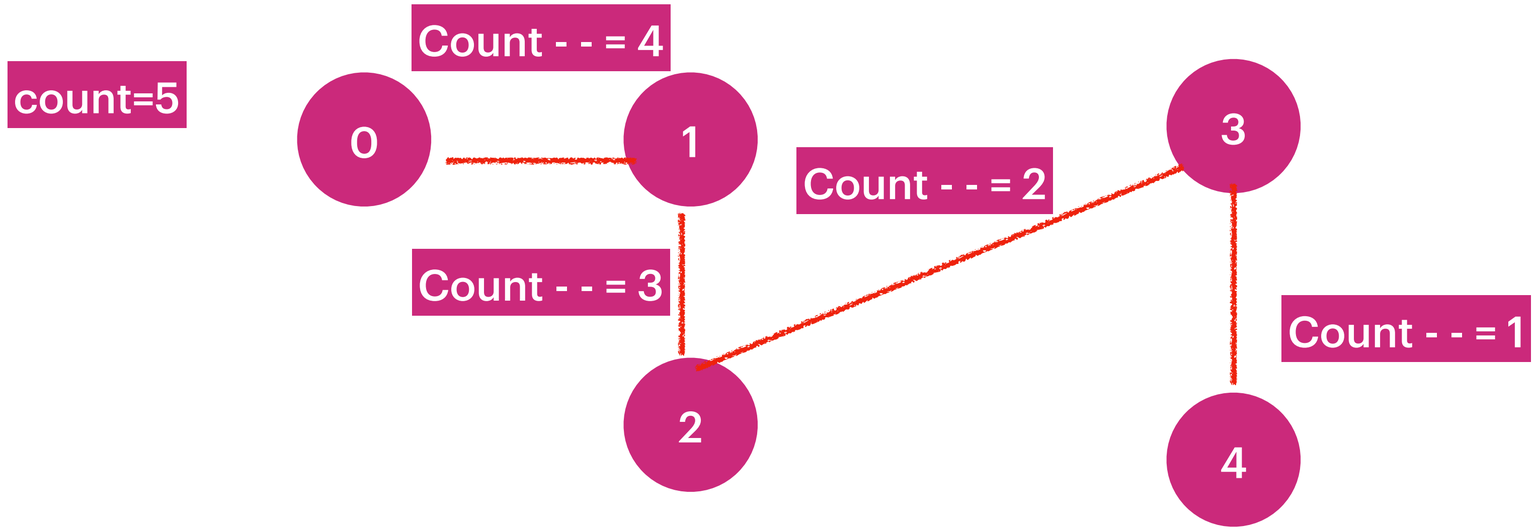
Time Complexity : O(n)
Space Complexity : O(n)

n = 5, edges = [[0,1],[1,2],[3,4]]

Count will be dicremented whenever we made a new Connection.



No Of distinct root = 2 : [0,3] So output is 2



n = 5, edges = [[0,1],[1,2],[2,3],[3,4]]

Root	0	0	0	0	0
Vertex/Index	0	1	2	3	4

No Of distinct root = 1 : [0] So output is 1