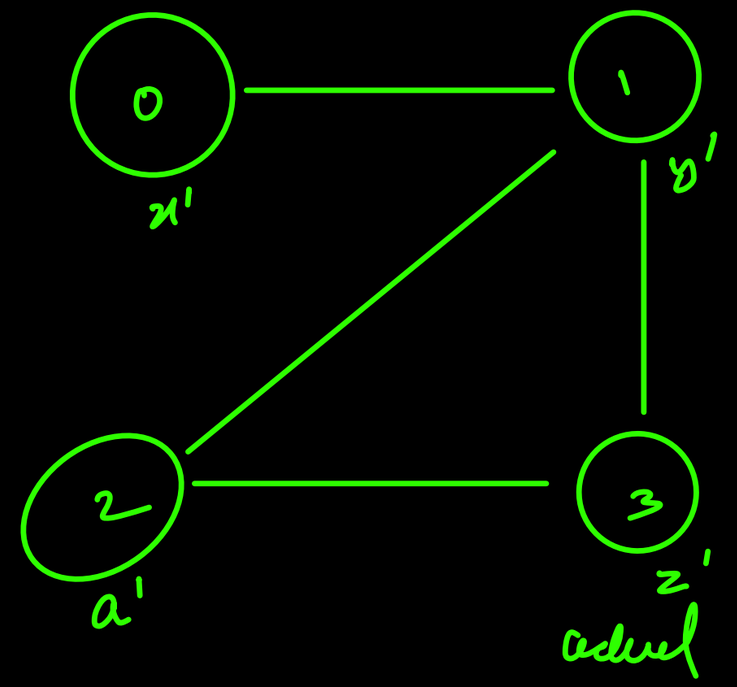
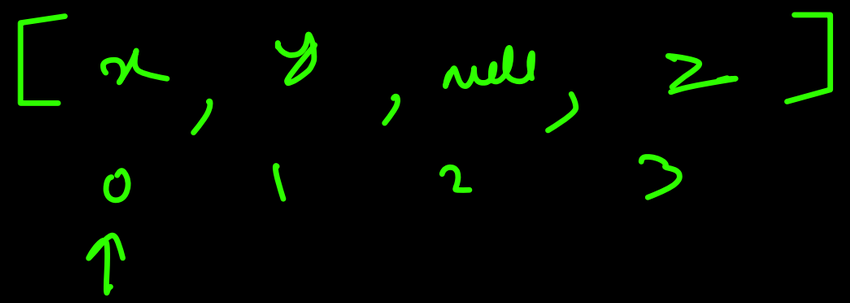
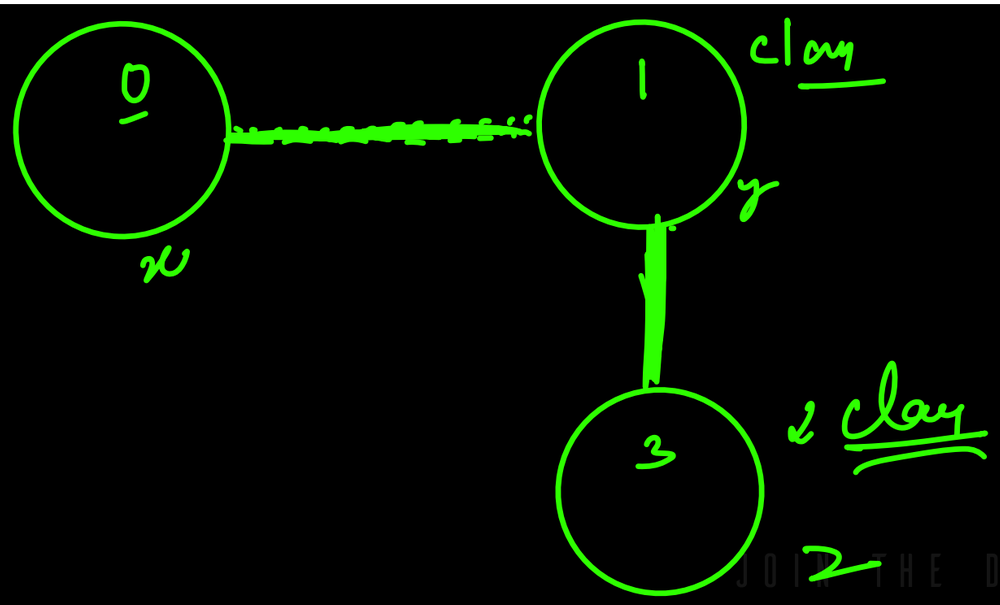
```cpp
class Solution {
public:
    vector<Node*> nodeRegister;

    void dfs(Node* actual, Node* clone) {
        for(auto neighbor : actual->neighbors) {
            if(not nodeRegister[neighbor->val]) {
                // create the neighbor for the first time
                Node* newNode = new Node(neighbor->val);
                nodeRegister[newNode->val]  = newNode;
                clone->neighbors.push_back(newNode);
                dfs(neighbor, newNode);
            } else {
                clone->neighbors.push_back(nodeRegister[neighbor->val]);
            }
        }
    }

    Node* cloneGraph(Node* node) {
        if(node == NULL) return NULL;
        Node* clone = new Node(node->val);
        nodeRegister.resize(110, NULL); // this array contains ref to the created nodes
        nodeRegister[clone->val] = clone;
        dfs(node, clone);
        return clone;
    }
};
```

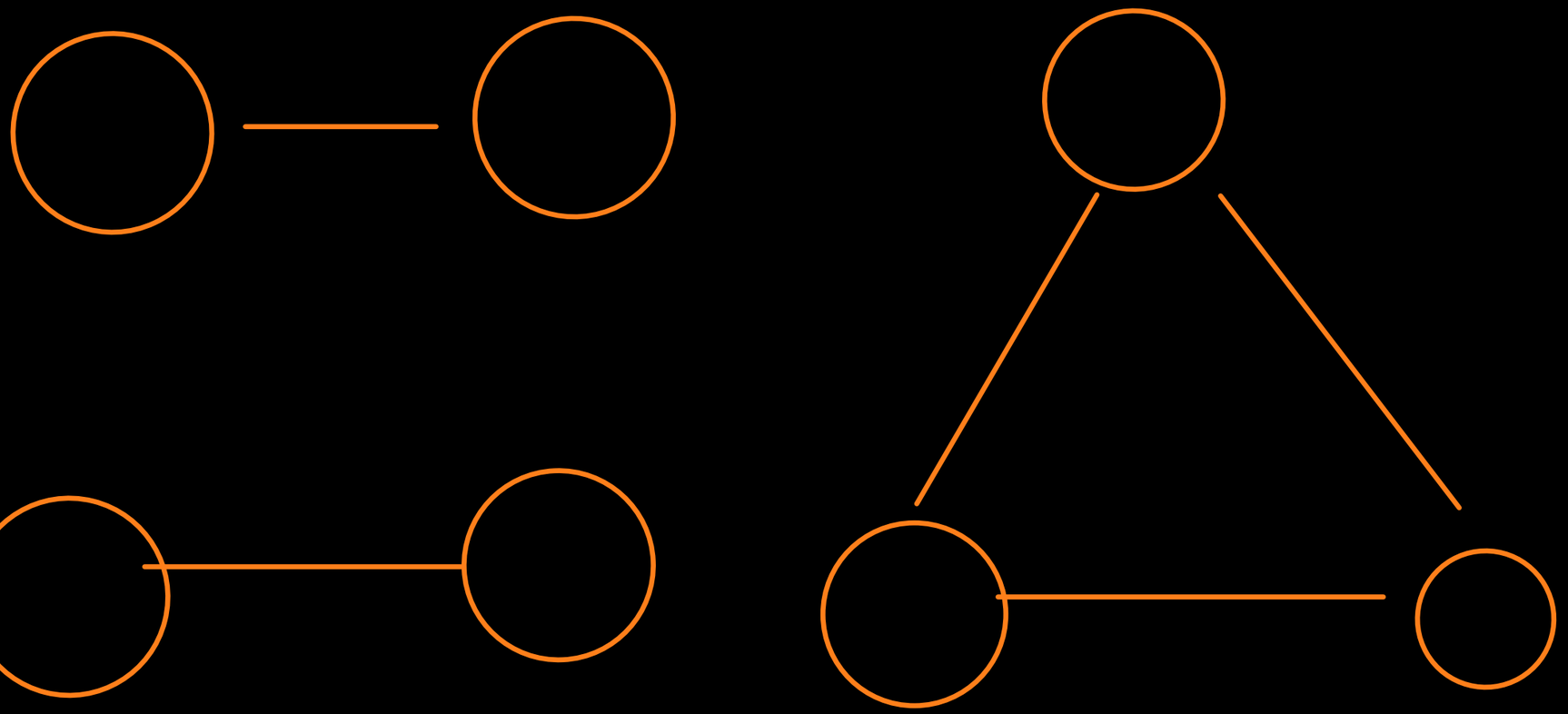




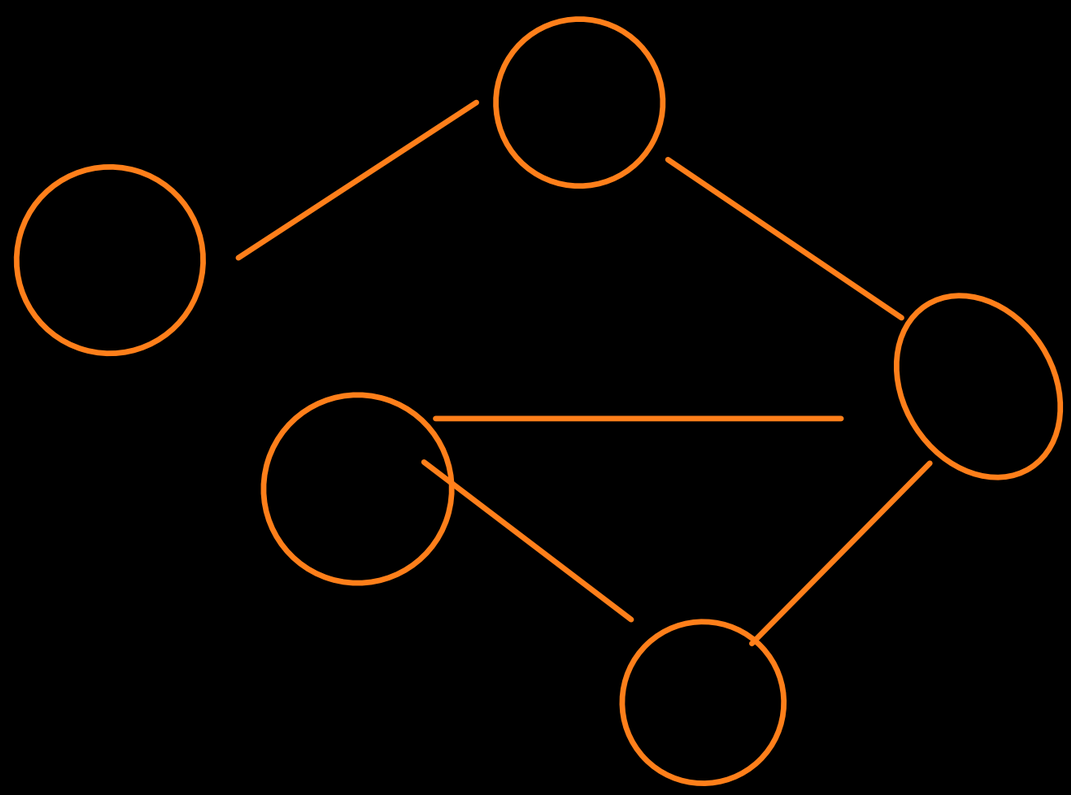


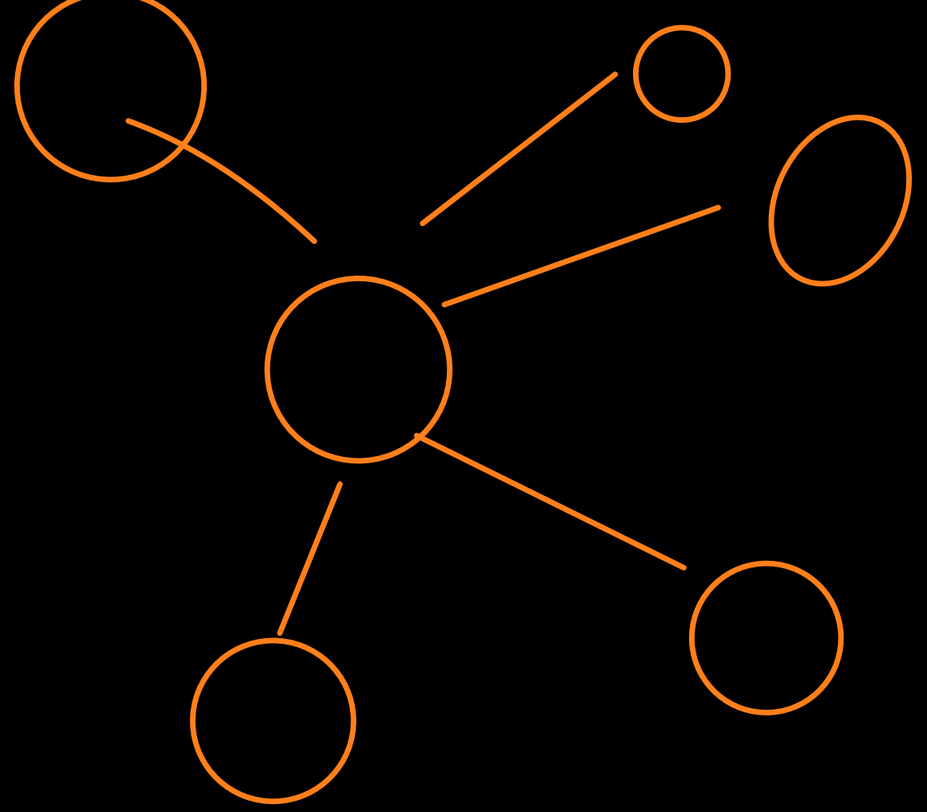
dfs

Connected Component → It is a subset of the given graph that has vertices between which there is always a path.



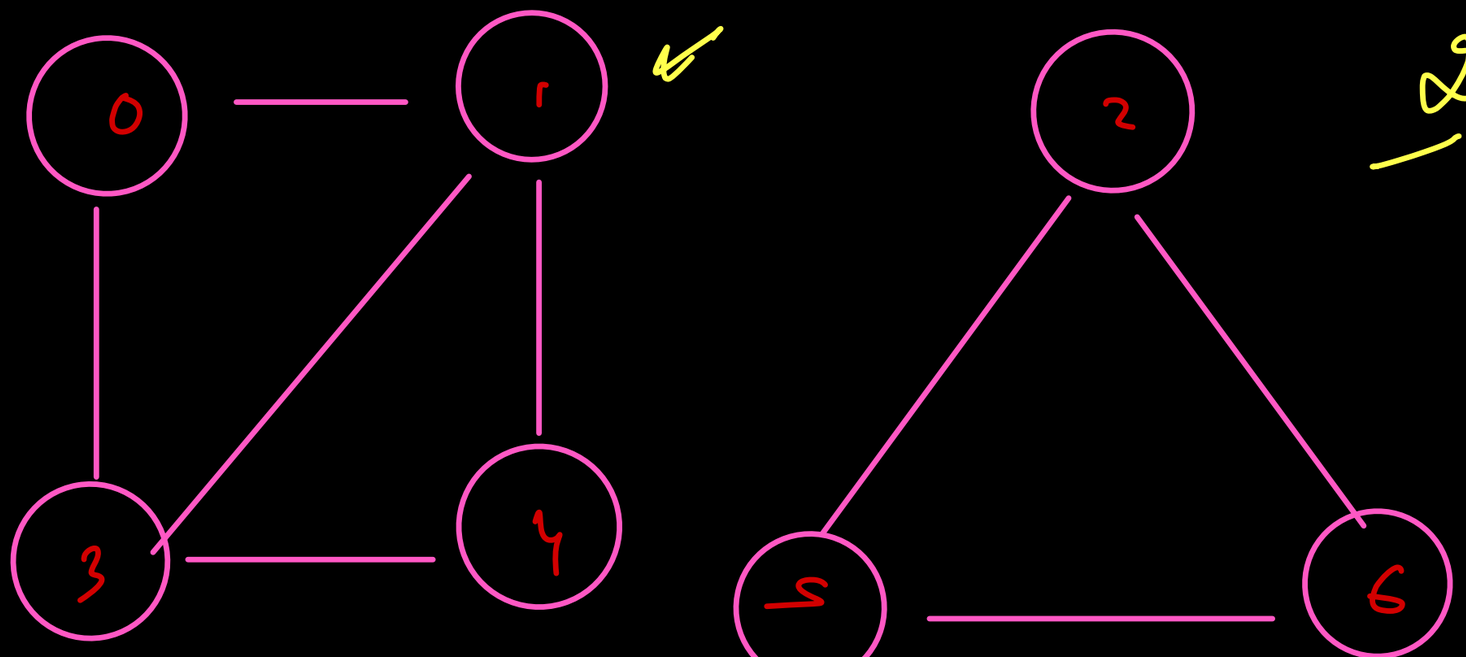
vertices present in 2 diff C.C. don't have a path

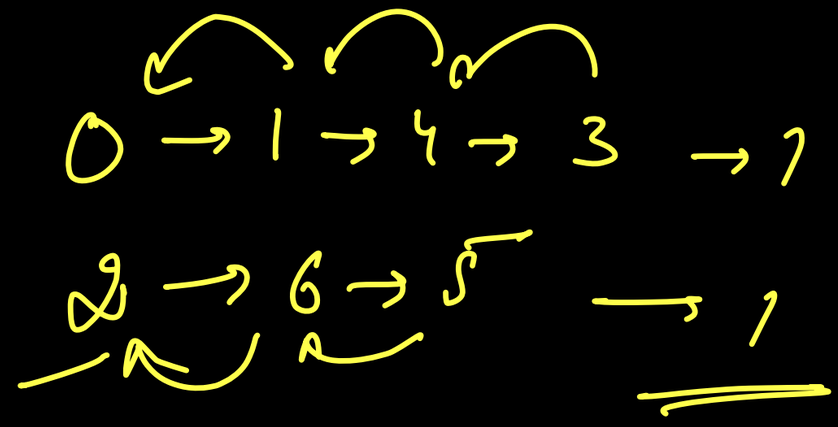




→ dfs / bfs →

No. of times dfs/bfs is called is the CC.





2, 5, 6

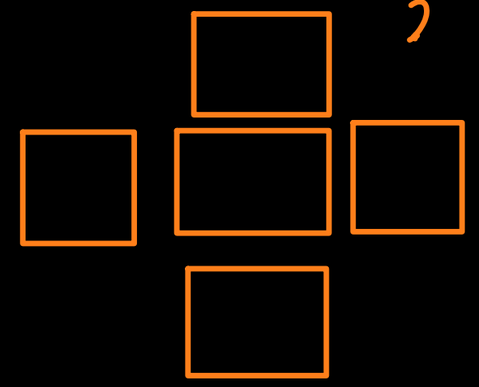
Q. leetcode 1034

row col

color = 4

grid[i][j] represent color of the cell

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



row = 3
col = 3

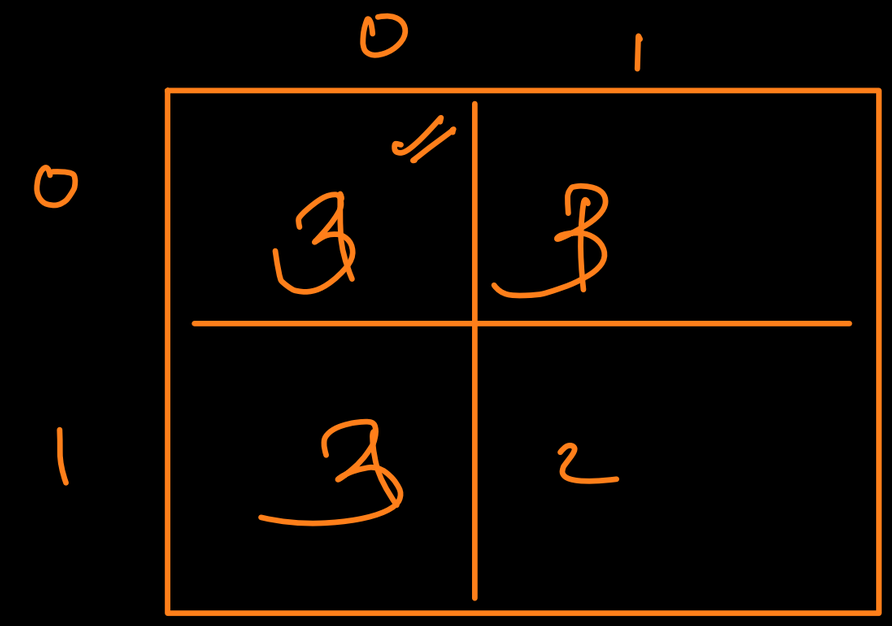
(i-1, j)

(i, j-1) ← (i, j) → (i, j+1)

(i+1, j)

color

[[4,3] [3,3]]



color=3