14 July 2024 14:08

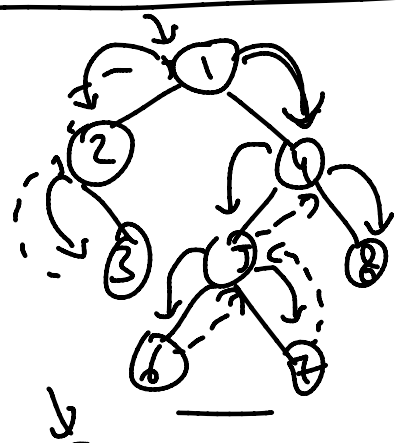
⇒ Graph Traversals :-

→ Visiting all the nodes in some order once.

DFS BFS

① DFS - Depth First Search :-

→



n = 8

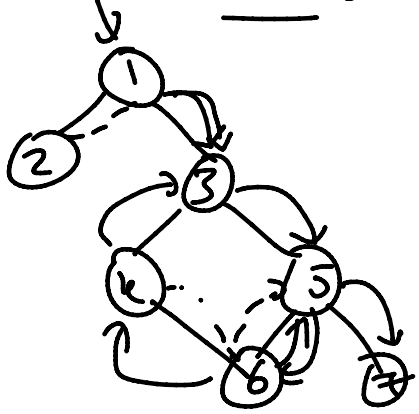
e = n-1 = 7

1, 2, 3, 4, 5, 6, 7, 8

1, 4, 8, 5, 6, 7, 2, 3

1, 4, 5, 6, 7, 8, 2, 3

→ Tree.



1, 4

1, 2

1 → 4, 2

1, 2

1, 4

1 → 2, 4

Multiple DFS are possible.

1, 2, 3, 4, 6, 5, 7

1, 2, 3, 5, 6, 4, 7

```
bool vis[V];
vis[i] = True;
```

⇒ Recursive :-

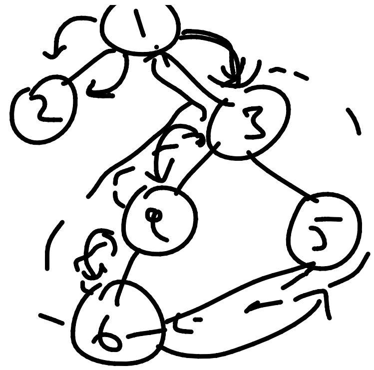
```
dfs(1);

void dfs(int i){
    vis[i] = T;
    print(i);

    for x in adj[i]:
        if(vis[x] == F){
```

1 → 3, 2

for x ...
if(vis[x]==F){
dfs(x);



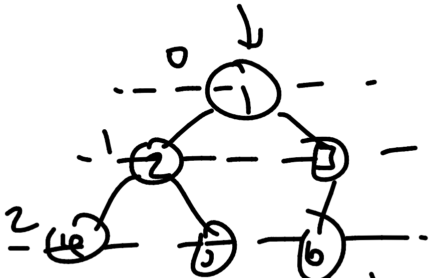
3
1, 2, 3, 4, 5, 6
~~F~~ ~~F~~ ~~F~~ ~~F~~ ~~F~~ ~~F~~
T T T T T T

3
3
1, 3, 4, 6, 5, 2

$TC = O(V+E)$. $AS = O(V)$

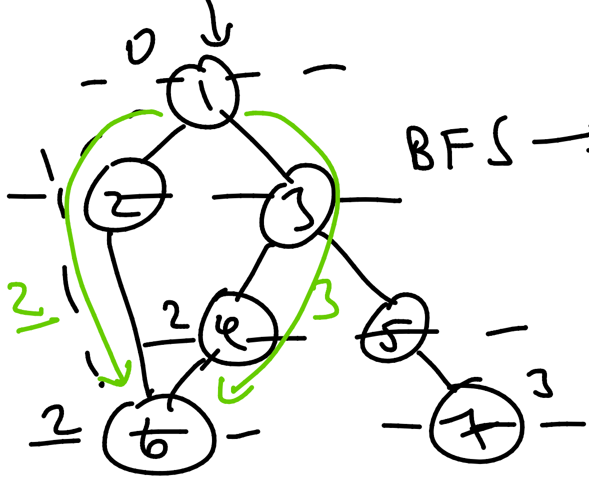
⇒ BFS – Breadth First Search :-

↳ level by level traversal.

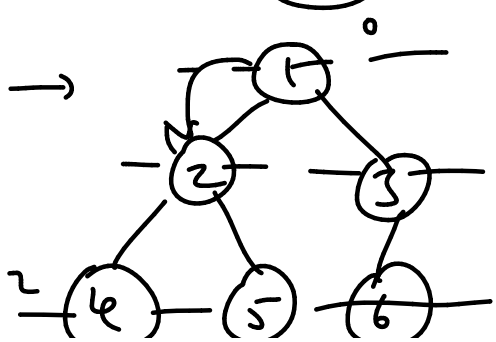


DFS → 1, 2, 4, 5, 3, 6

BFS → 1, 2, 3, 4, 5, 6 ✓
1, 3, 2, 6, 5, 4 ✓



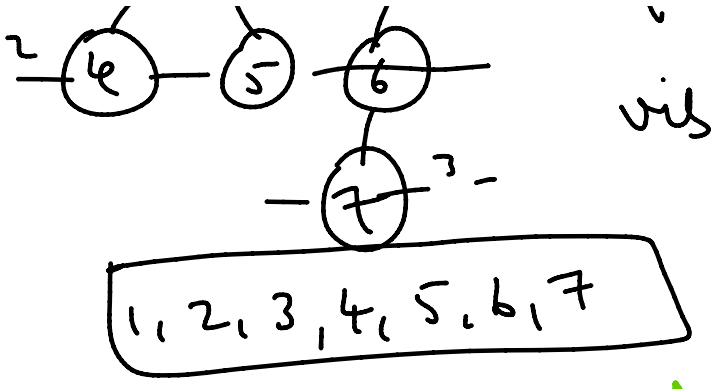
BFS → 1, 2, 3, 6, 4, 5, 7



1, 2, 3, 4, 5, 6, 7
0 1 2 3

q → ~~1~~, ~~2~~, ~~3~~, ~~4~~, ~~5~~, ~~6~~, ~~7~~

... 1, 2, 3, 4, 5, 6, 7

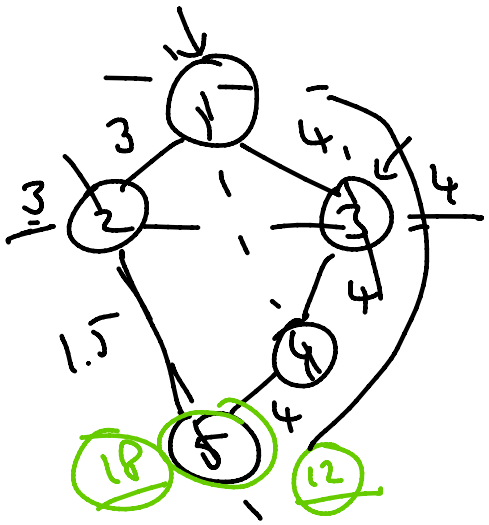


vis → 1, 2, 3, 4, 5, 6, 7

~~F~~ ~~F~~ ~~F~~ ~~F~~ F F F

T T T T T T T

When we push an element in queue, we mark it as true, assuming uniform edge weight.



→ queue<int> q;

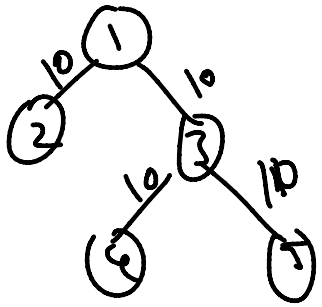
```
q.push(1);
vis[1] = T;
while (q.empty() == F)
{
    i = q.front();
    q.pop();
    print(i);

    for x in adj[i];
        if( vis[x] == F)
        {
            q.push(x);
            vis[x] = T;
```

TC = O(V+E)

AS = O(V)

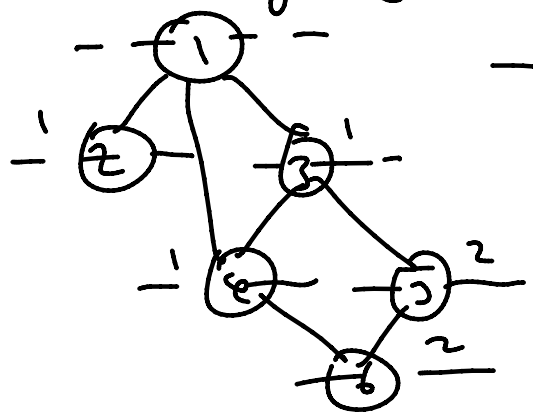
} } }



→ Applications :-

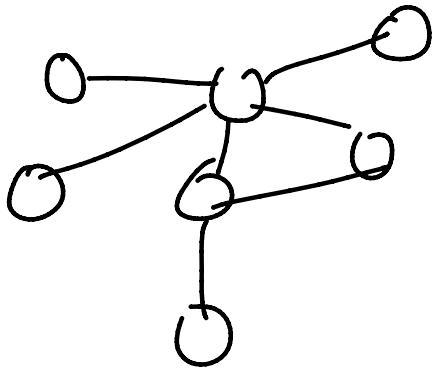
① Find min distance b/w 2 nodes when edge weights are uniform.

② → Find shortest distance from source node to all other nodes.

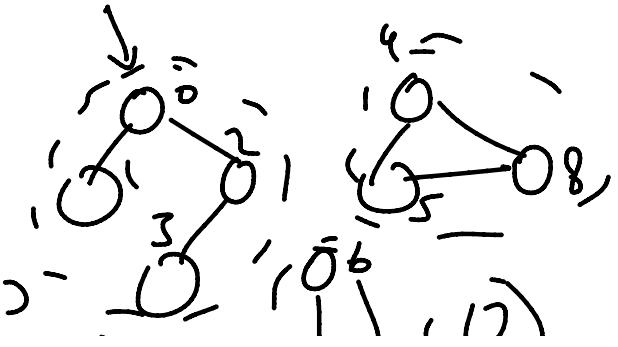


⇒ Connected graph :-

There is a path b/w each and every pair of vertices.

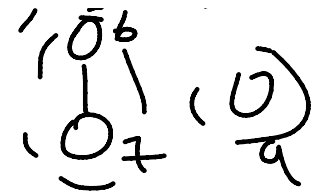


⇒ Unconnected graph



(4) connected

(4) connected components :-

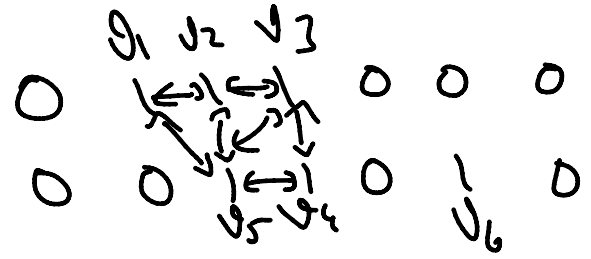


0 1 2 3 4 5 6 7 8 9

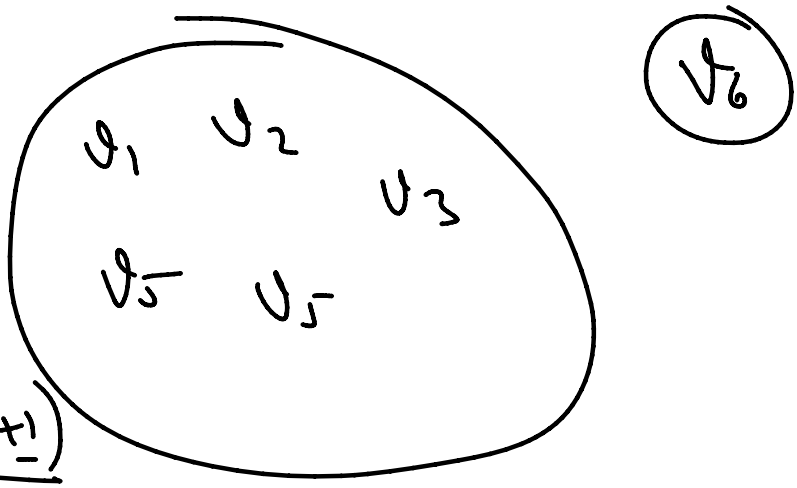
T_1 T_1 T_1 T_1 T_2 T_2 T_3 T_3 T_2 T_4

```
ans=0;
for (i=0 ; i<V ; i++)
{
  if(vis[i]==F)
  {
    dfs(i) ; ans++;
  }
}
```

ans=1
ans=2
ans=3
ans=4

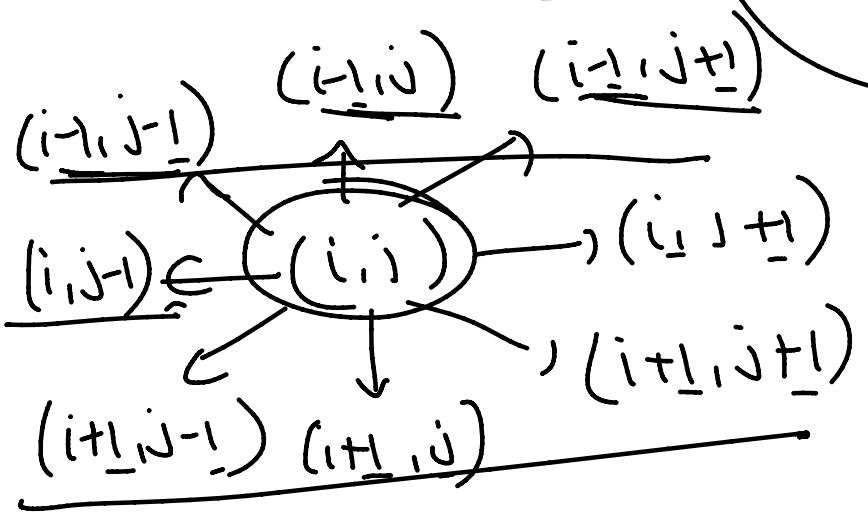


all → vertices.



i
i-1
i

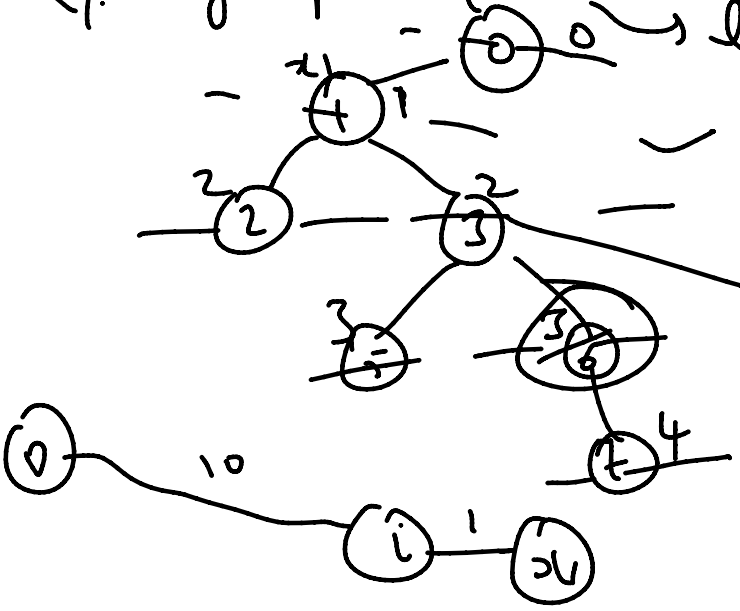
(2)




```
for(i1=-1; i1<=1; i1++)
{
  for(i2=-1; i2<=1; i2++)
  {
    (i+i1, i+i2)
  }
}
```

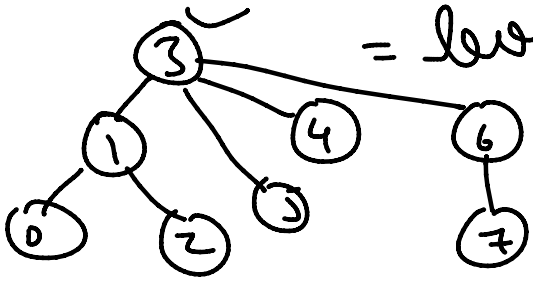
Q1 graph. x → node.

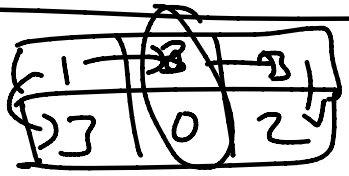
Q/ graph. x → node
0 → level.



x = 6

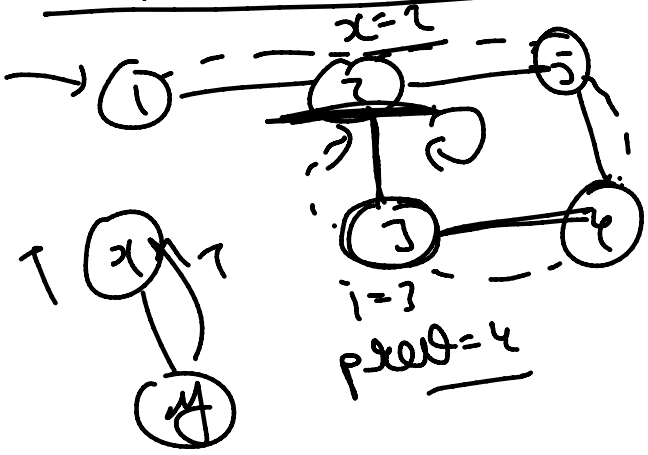
level(x)
= level[i] + 1;

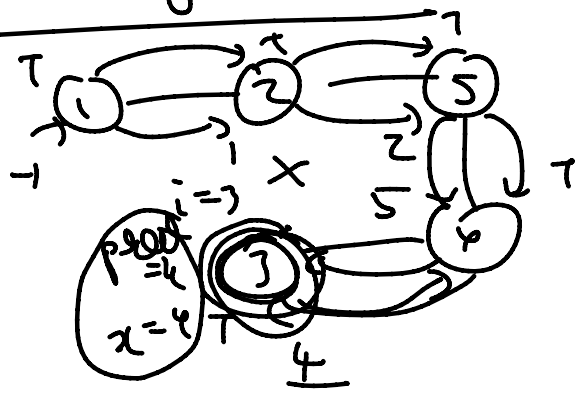


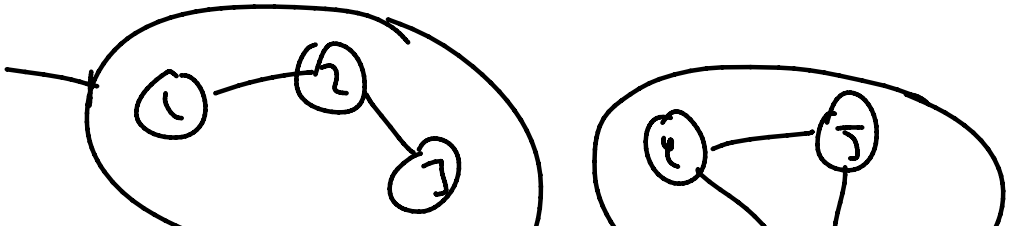


False
True

Q/. Cycle in undirected graph:-

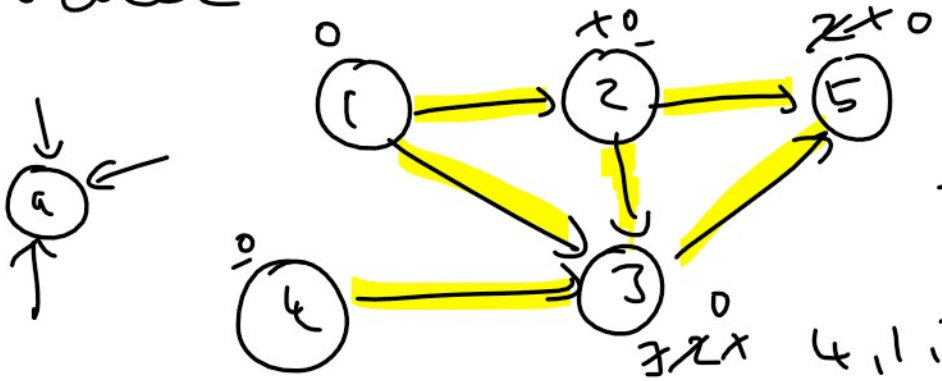






=) Topological sort :-

It is an ordering of graph s.t if $a \to b$, then in topological sorted order a must come before b.



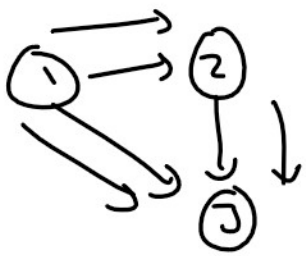
multiple topological sorted order are possible.

4, 1, 2, 3, 5 ✓

1, 2, 4, 3, 5 ✓

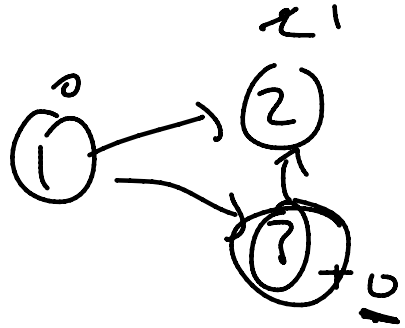
1, 4, 2, 3, 5

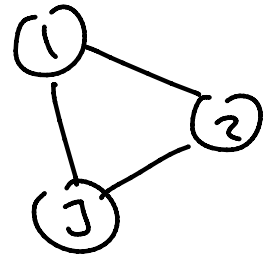
only defined for DAG.

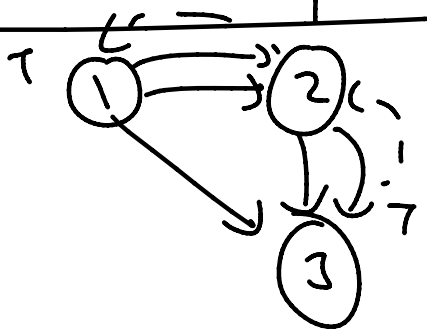


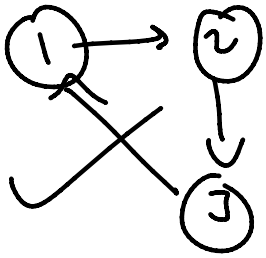

```
queue<int> q;
for(i=0; i<V; i++)
```

```
for(i=0;i<v;...)
{
    if(in_degree[i]==0)
    {
        q.push(i);
    }
}
while(!q.empty())
{
    i=q.front();
    q.pop();
    print(i);
    for(x in adj[i])
    {
        in_degree[x]--;
        if(in_degree[x]==0)
            q.push(x);
    }
}
```







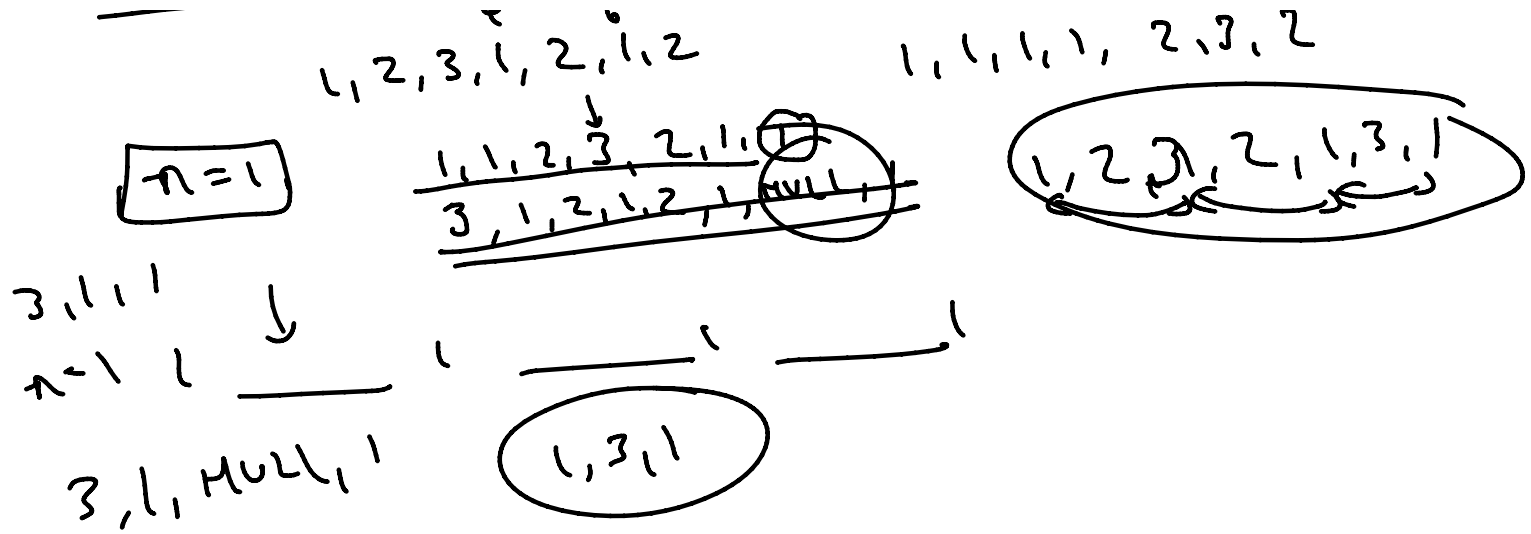


⇒ **Doubt :-**

1, 2, 3, 1, 2, 1, 2

1, 1, 1, 1, 2, 3, 2

n=2


1,2,3,1,2,1,2
1,1,1,1,2,3,2
n=1
1,1,2,3,2,1,1
3,1,2,1,2,1,NULL,1
1,2,3,2,1,3,1
3,1,1
n-1
3,1,NULL,1
1,3,1