Plan

Saturday, 25 September 2021 7:04 PM

Day 3 (Morning)

1. As far from land as possible
2. Mother vertex
3. Kosaraju Algorithm

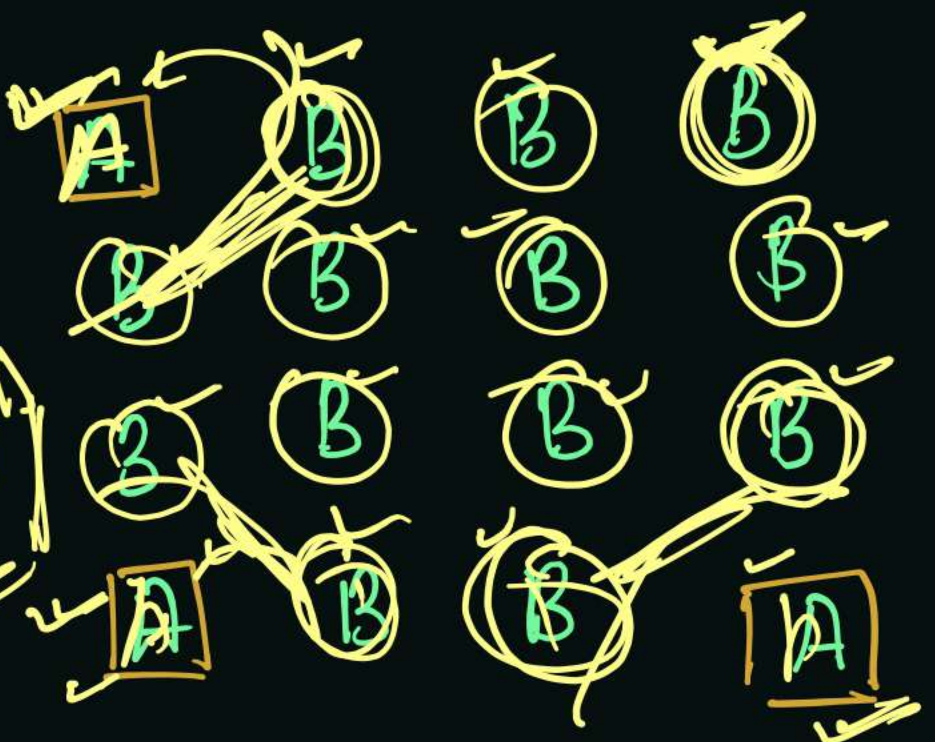
Day 4 (Evening)

1. Shortest bridge
2. Articulation point and bridge
3. Critical connection

As Far From Land As Possible

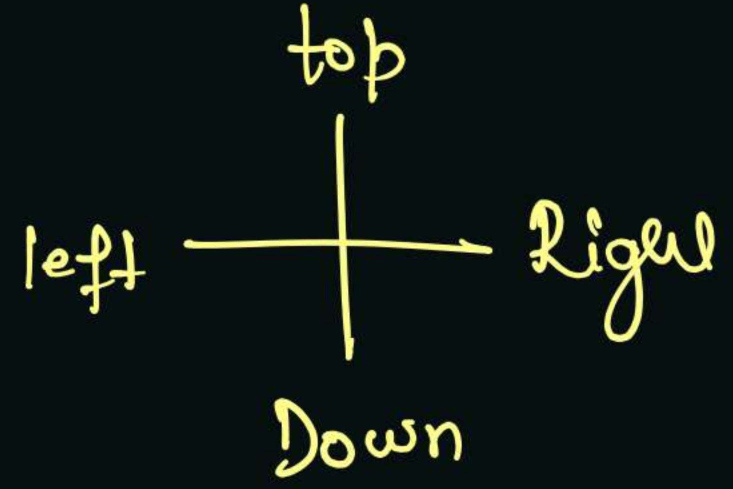
Saturday, 25 September 2021 7:04 PM

A ⇒ Land , B ⇒ water (Manhattan dist)



V(V+E)

Maximise min distance from B to A.



put all A's

All initial A's are at same level

Simultaneously BFS from land

Min distance from B to A →

A	1	2	3
1	2	3	2
1	2	2	1
A	1	1	A

Result = B

un

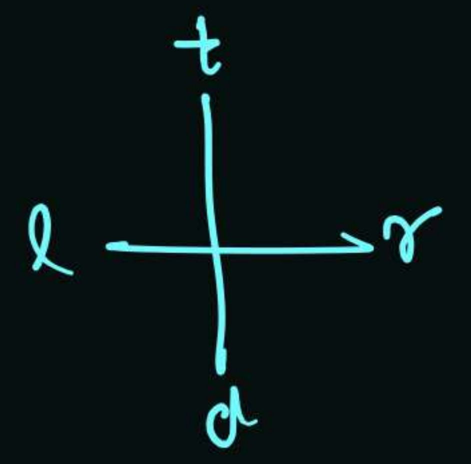
take all coordinate at level→0 (initial A's coordinate)

```
pair ← level X
while(
    while
    }
}
```

```
pair ← level
while(){ (V+E)
}
```

$$\begin{array}{c|cccc} & 0 & 1 & 2 & 3 \\ \hline 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 \\ 3 & 1 & 0 & 0 & 1 \end{array}$$

① add initial one's



level = ~~1~~ ~~0~~ ~~1~~ ~~2~~ 3

| ~~(0,0)~~ | ~~(3,0)~~ | ~~(3,3)~~ | ~~(1,0)~~ | ~~(0,1)~~ | ~~(2,0)~~ | ~~(3,1)~~ | ~~(2,3)~~ | ~~(3,2)~~ | ~~(1,1)~~ | ~~(0,2)~~ | ~~(2,1)~~ | ~~(1,3)~~ | ~~(2,2)~~

Size = ~~3~~ ~~6~~ ~~5~~ 2

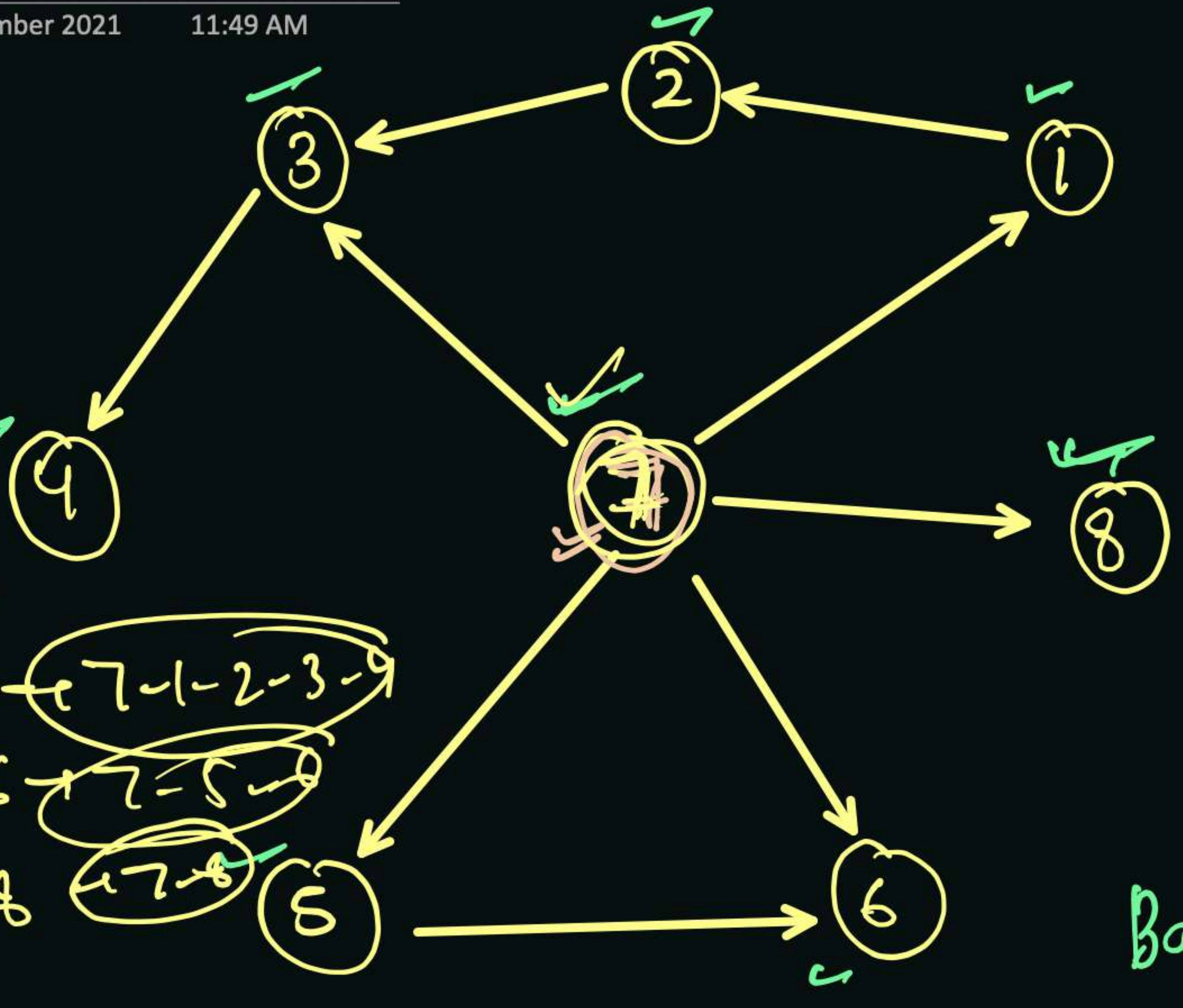
~~(1,2)~~ | ~~(0,3)~~ |

level = 3 → Max distance →

↳ Manhattan distance

Mother Vertex

Sunday, 26 September 2021 11:49 AM



log

7-4 → 7-1-2-3-4

7-6 → 7-5-6

7-8 → 7-8

We can visit all vertex from 7, so

'7' is mothervex.

① Brute force

→ Try.

Static

Djikstra's

(shortest path in tms of weight

Topological Sort

Back track → stack.push(vertex)

mark ① → DFS

2 → mark

3 → "

4 → "

⑤ → DFS

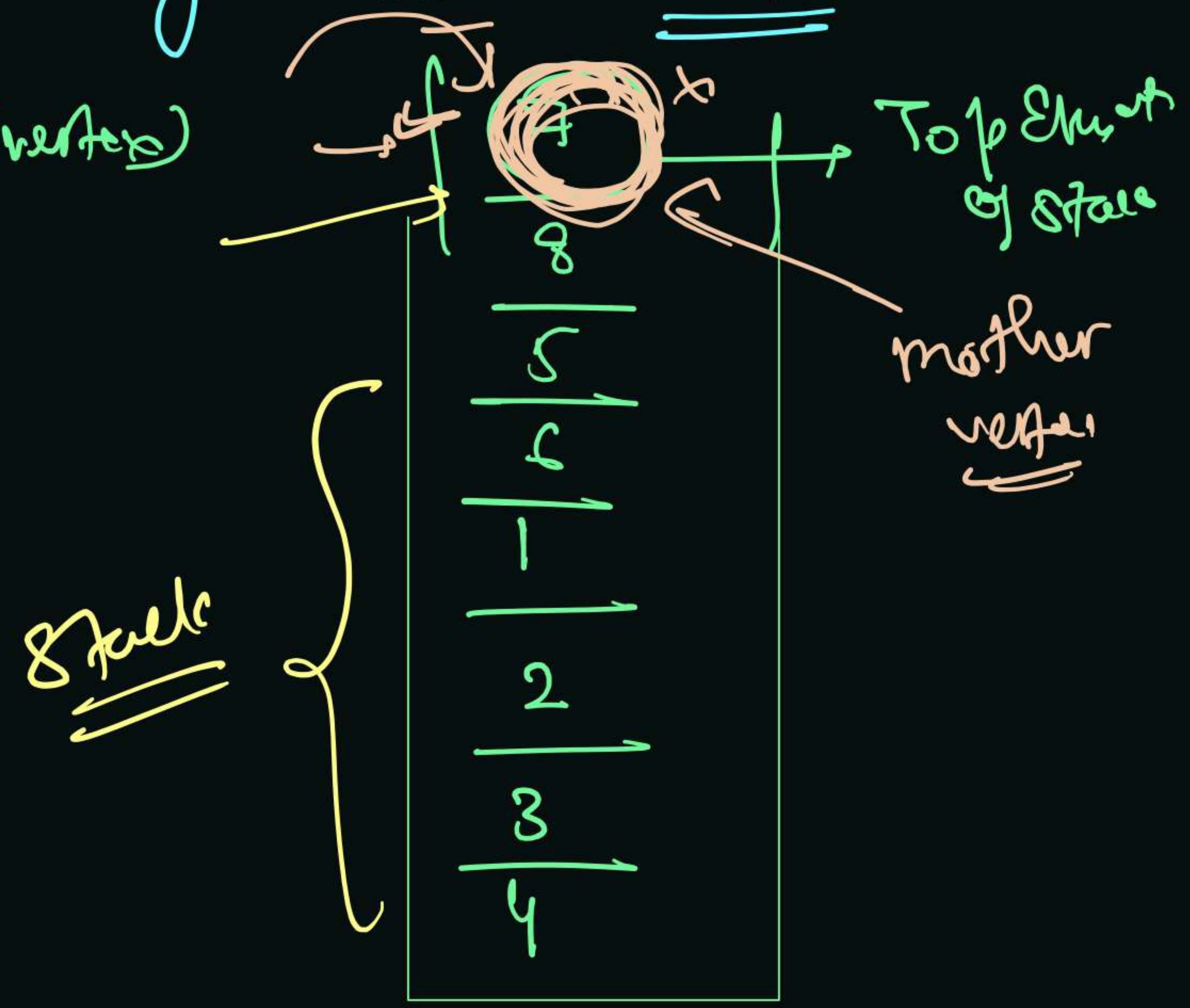
6 → mark

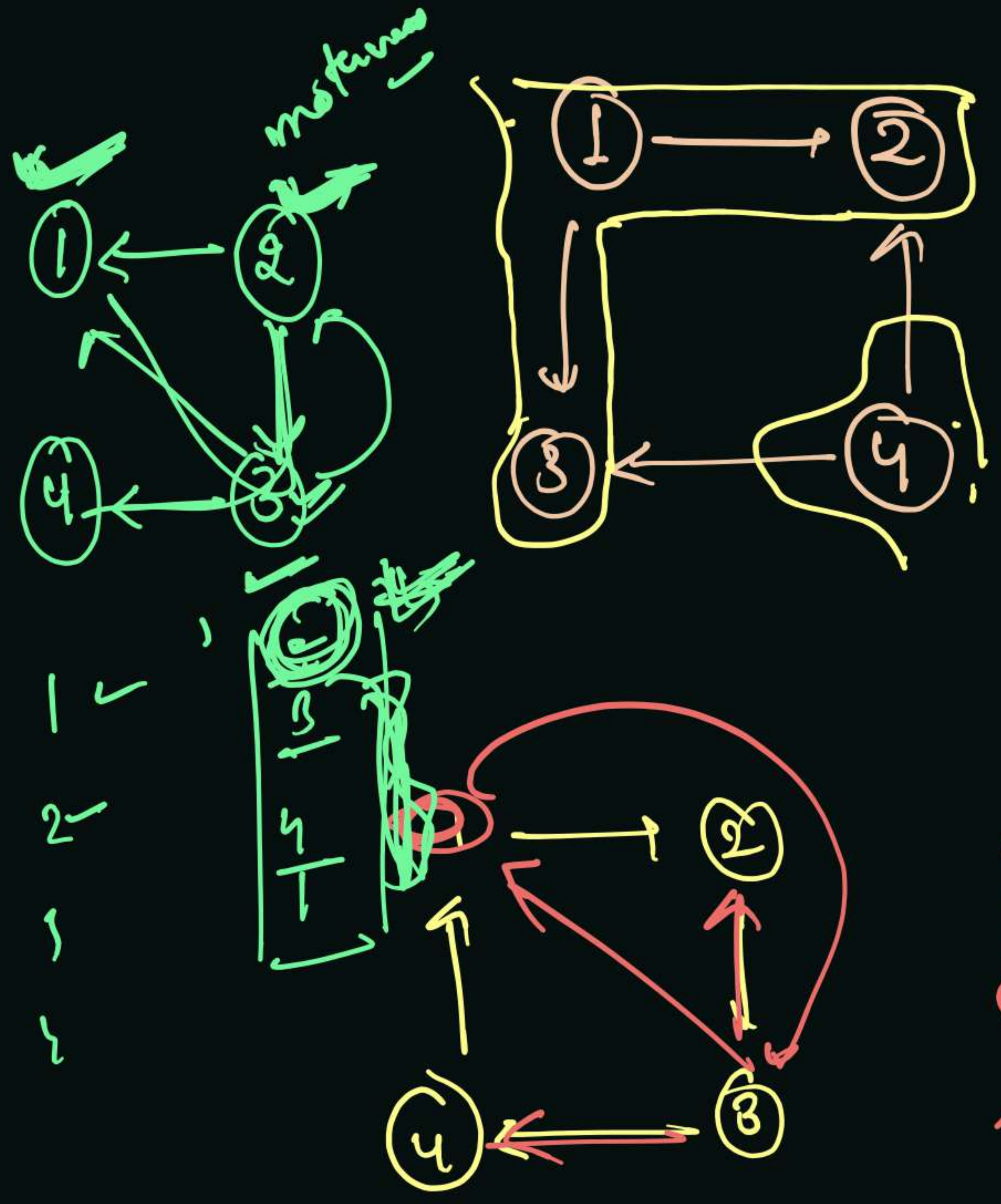
7 → mark

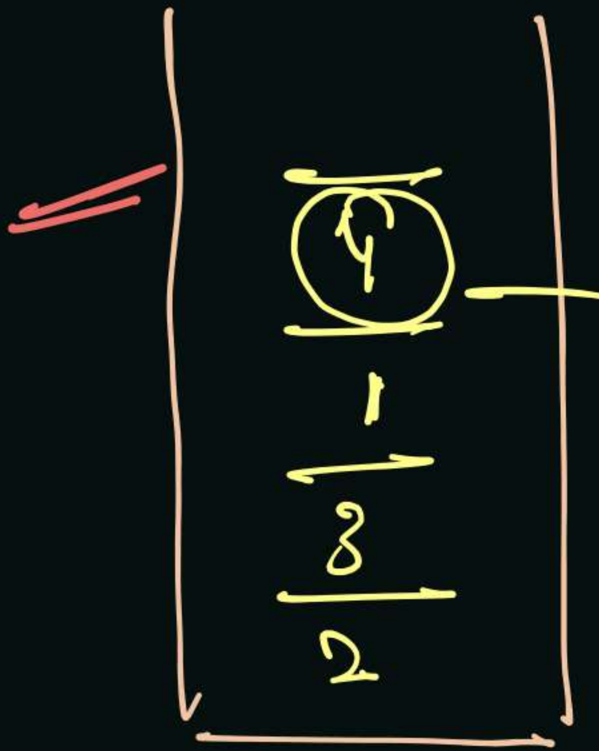
8 → "

$$^nC_2 = \frac{n(n-1)\cancel{(n-2)!}}{\cancel{(n-2)!}(2)!}$$

$$\boxed{= V^2} \Big] \times \text{DFS}$$

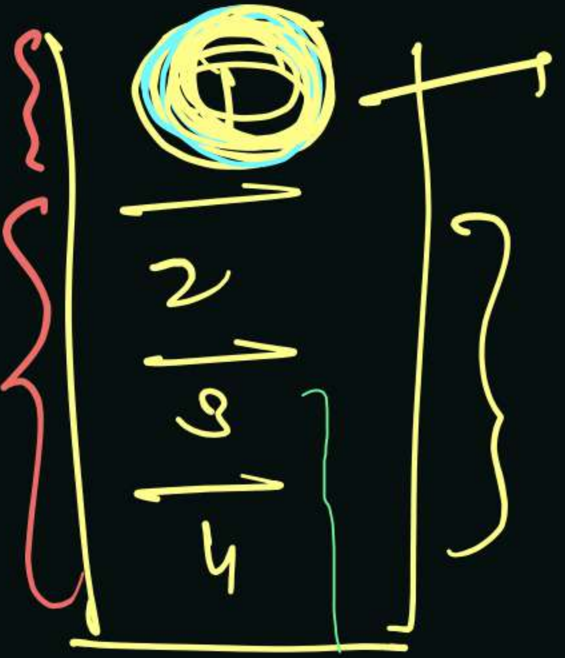




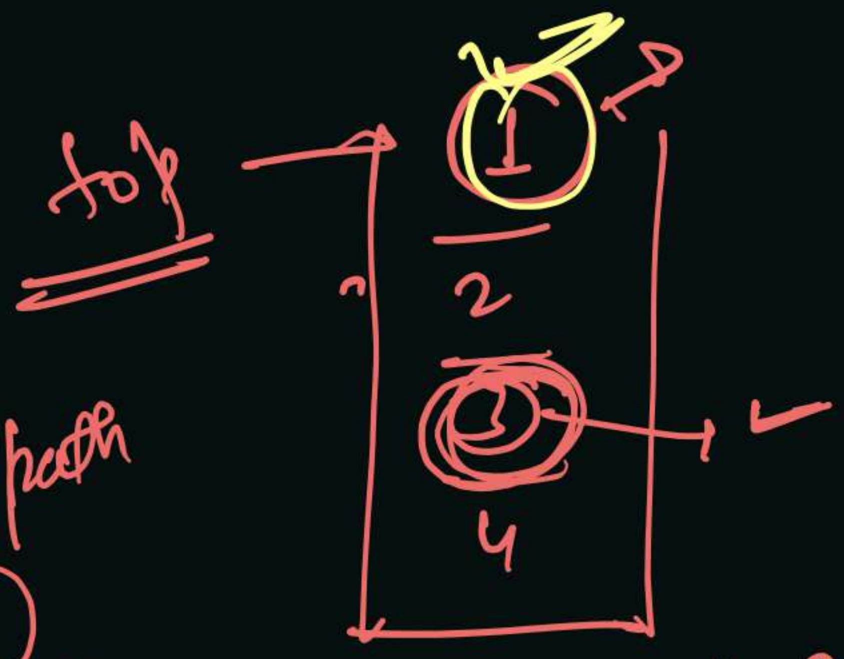


1 → DF
2 → mark
3 → mun
4 —

4 is at top but it is not mothervertex

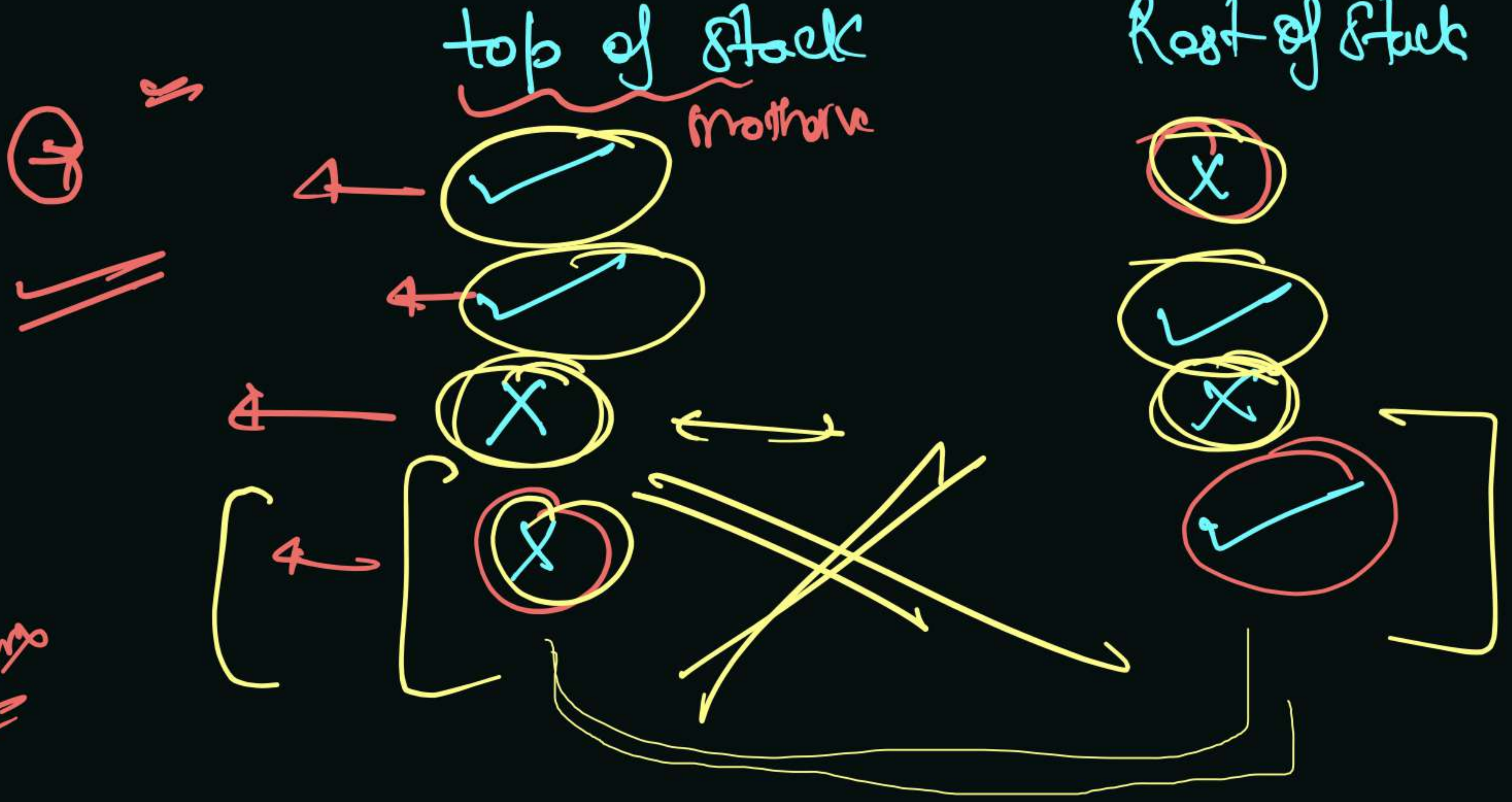


mothervert
1 → DF
2
3
4

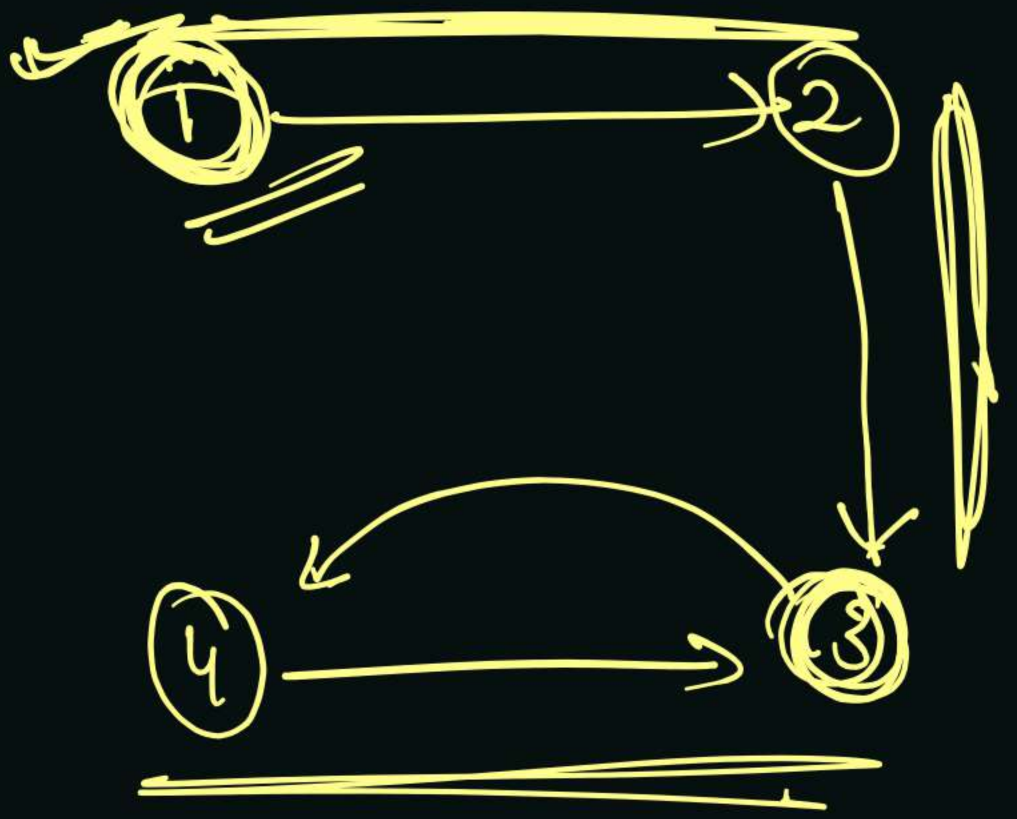


1 have path till 3 and 3 is [Motherverat] 1 is mother vertex

definitely



Example →



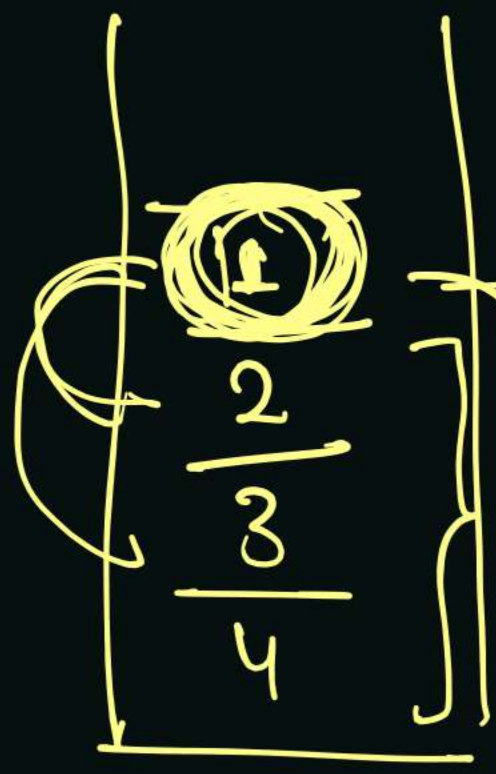
✓ 3 → DFS

✓ 2 → DFS

– 1 –

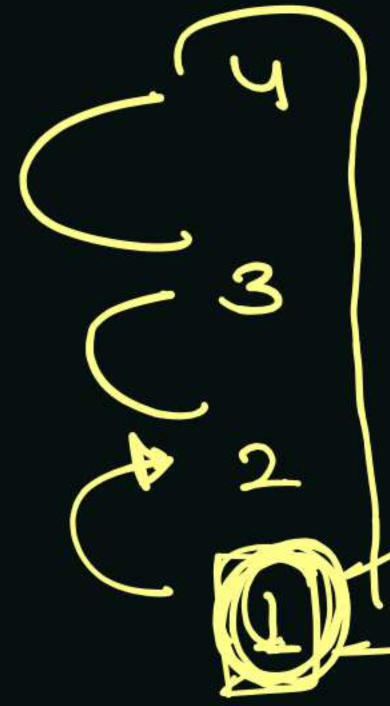
✓ 4 → already visit

Stack.



Mother vertex

Valid candidate for mother Vertexs



if exist then it is mother vertex otherwise

↳ No mother vertex is here.

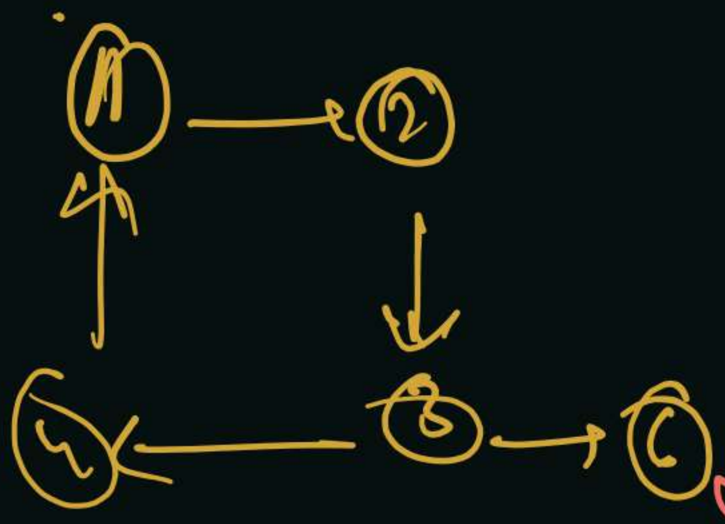
Travel all element

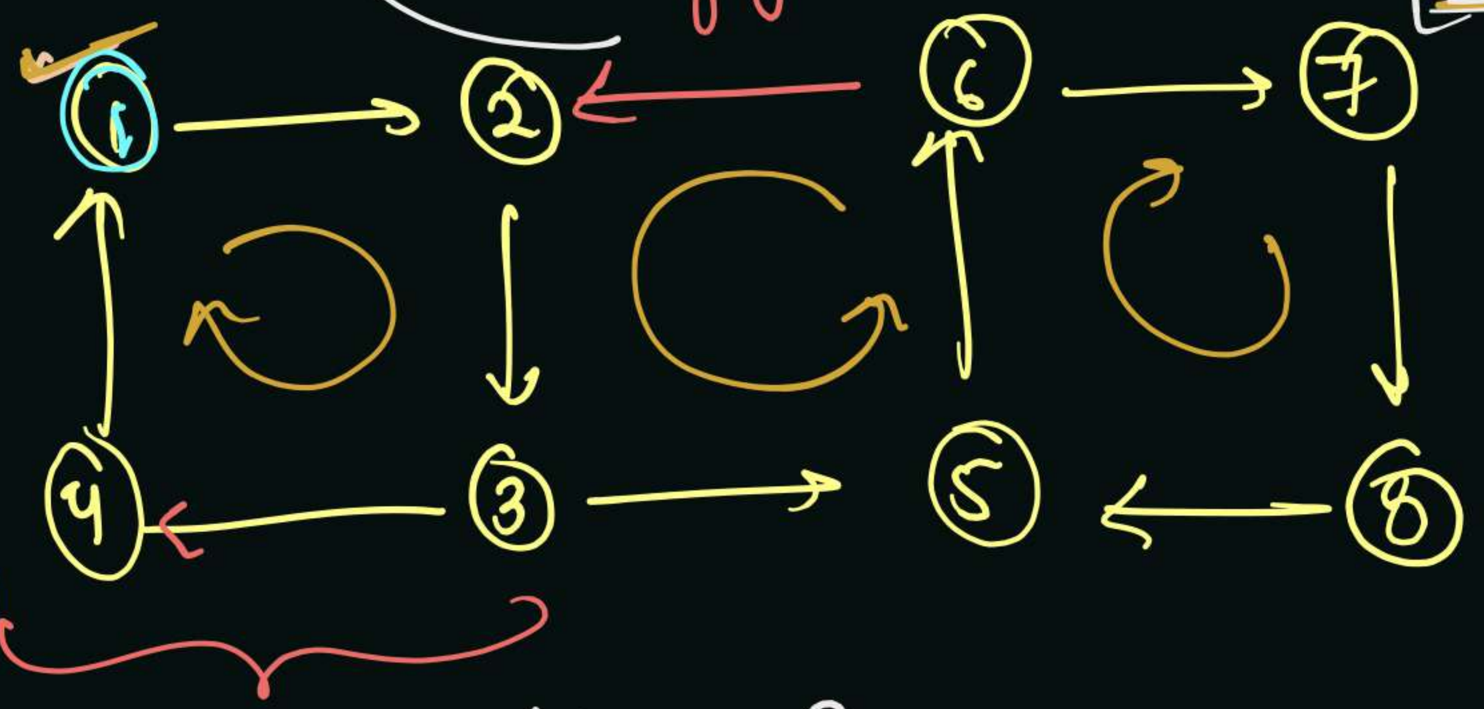
Kosaraju Algorithm

Sunday, 26 September 2021 12:44 PM

Red ~~edge~~ + Yellow Edge.

Strongly Con. Comp+ → {1, 2, 3, 4, 5, 6, 7, 8} {1, 2, 3, 4}





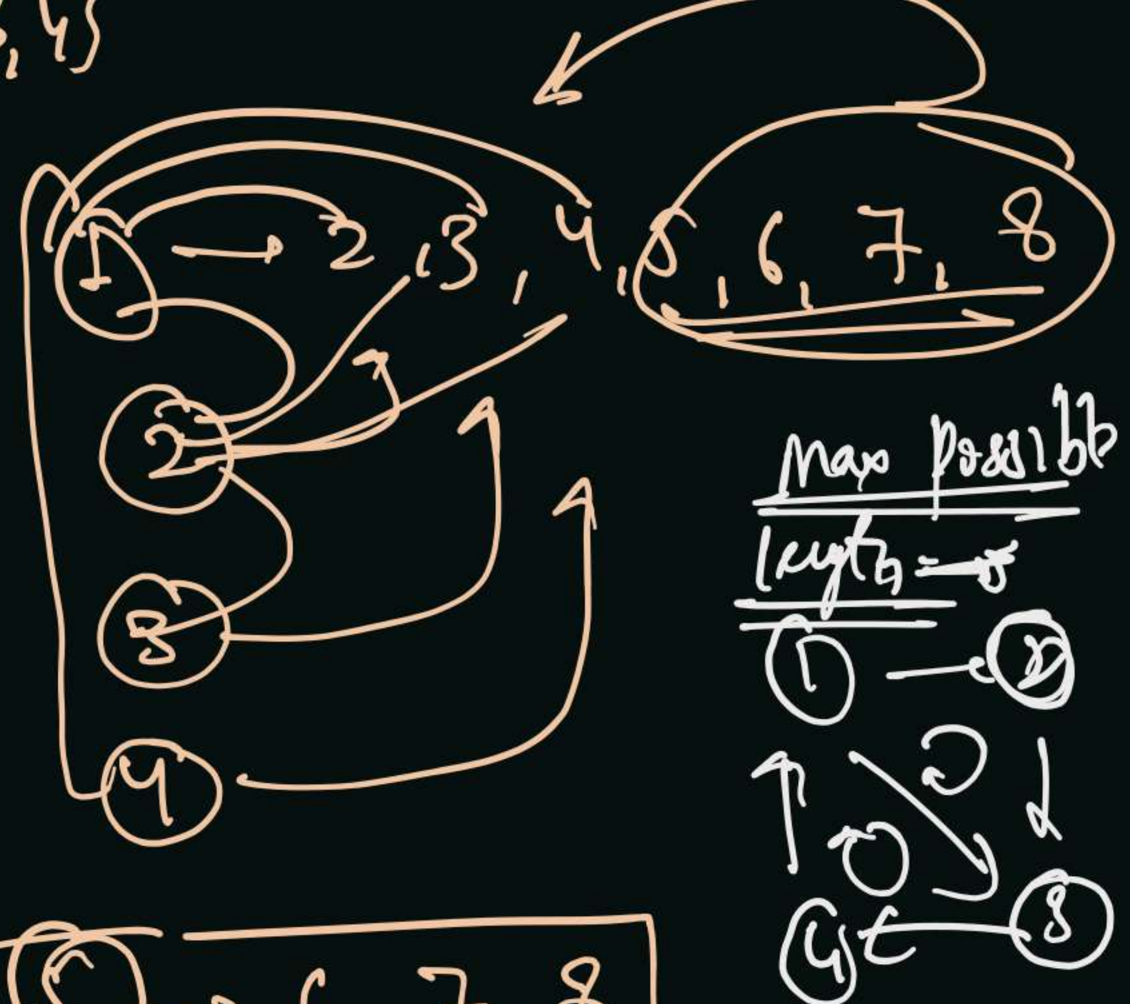
Cont=1

Yellow Edge

{1, 2, 5, 4} {5} {1 2 3 4} 1

{5, 6, 7, 8} 2

- from any vtx in this set, we can travel all vertex present in it



{5, 6, 7, 8}

5 → 6, 7, 8

6 → 7, 8, 5

7 → 8, 5, 6

8 → 5, 6, 7

{1, 2, 3, 4}

[1], [2 7 3 4]

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

strongly connected component is cyclic

[No. of cycle is no. of strongly ~~connect~~ component it have] loop. → Stack while in backtracking

✓1 →

✓2

✓3

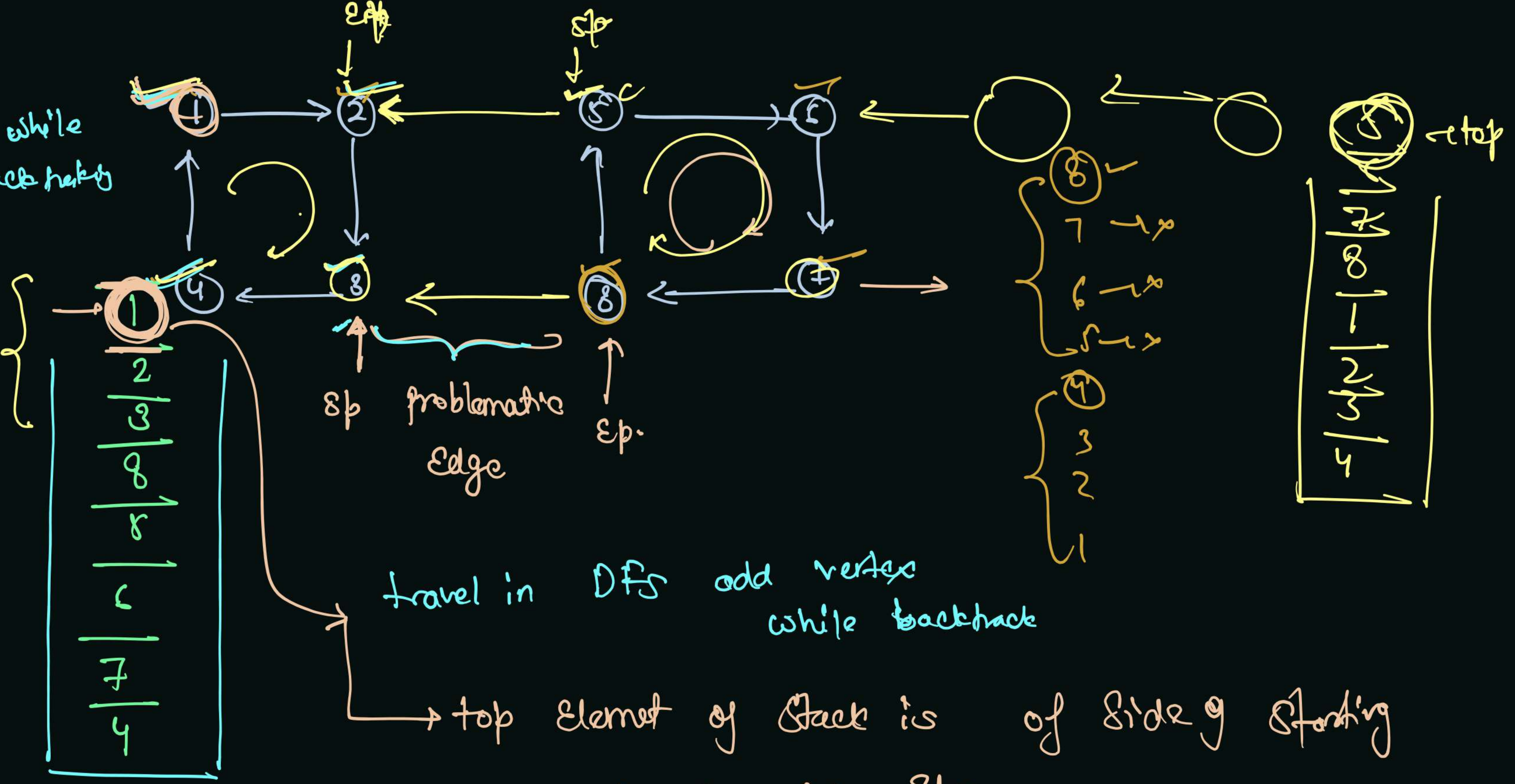
✓4

✓5

✓6

✓7

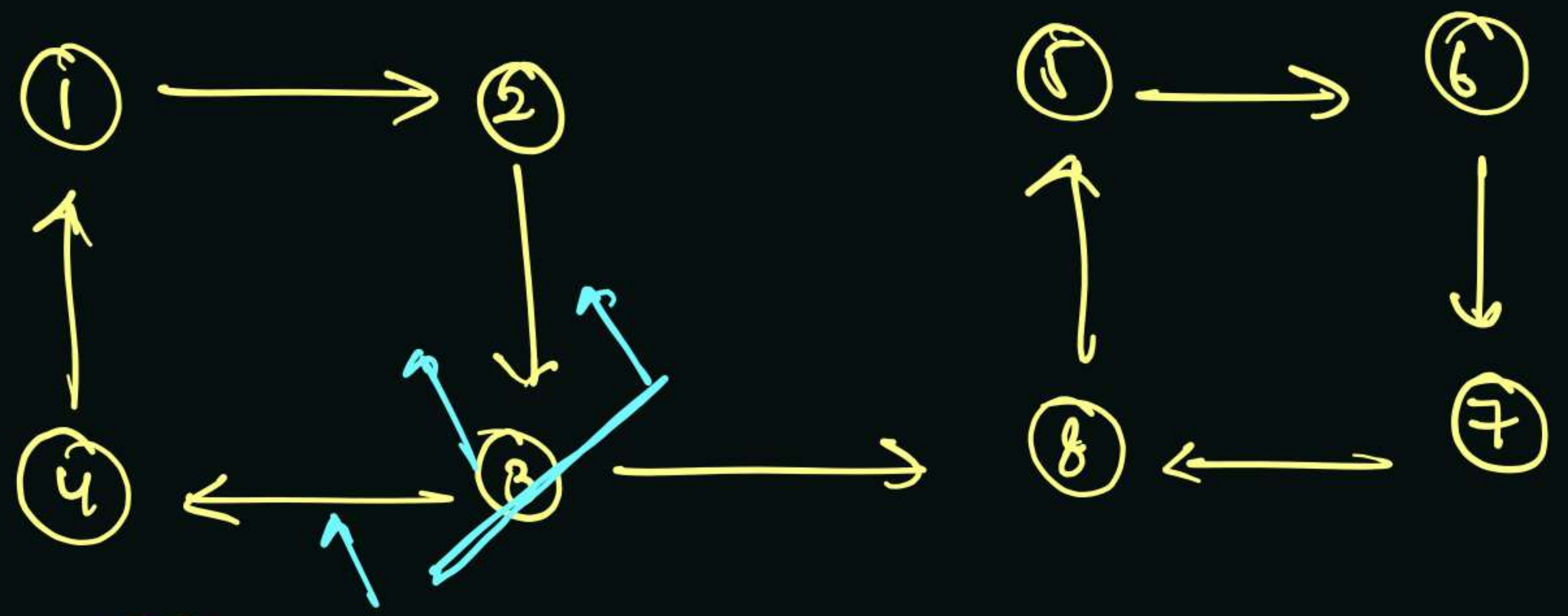
✓8



travel in DFS add vertex while backtrack

→ top Element of Stack is of Side g Starting in problematic Edge.

⇒ Starting point of problematic can be reachable from top element.



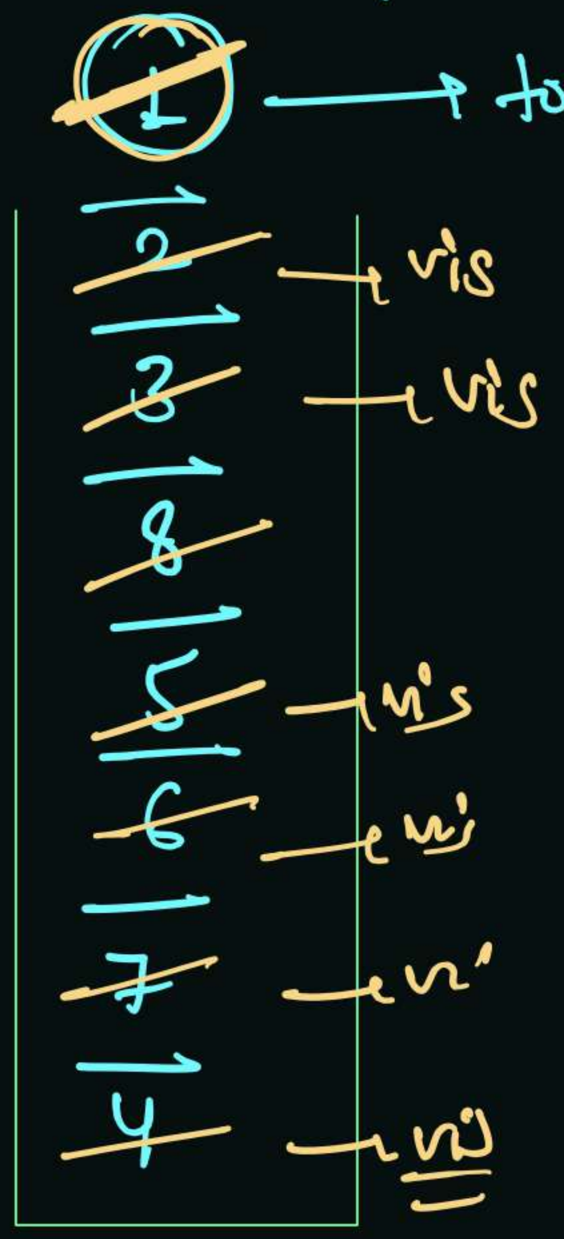
Step ① travel on vertex and call DFS from unvisited vertex and add vertex in stack in post order.

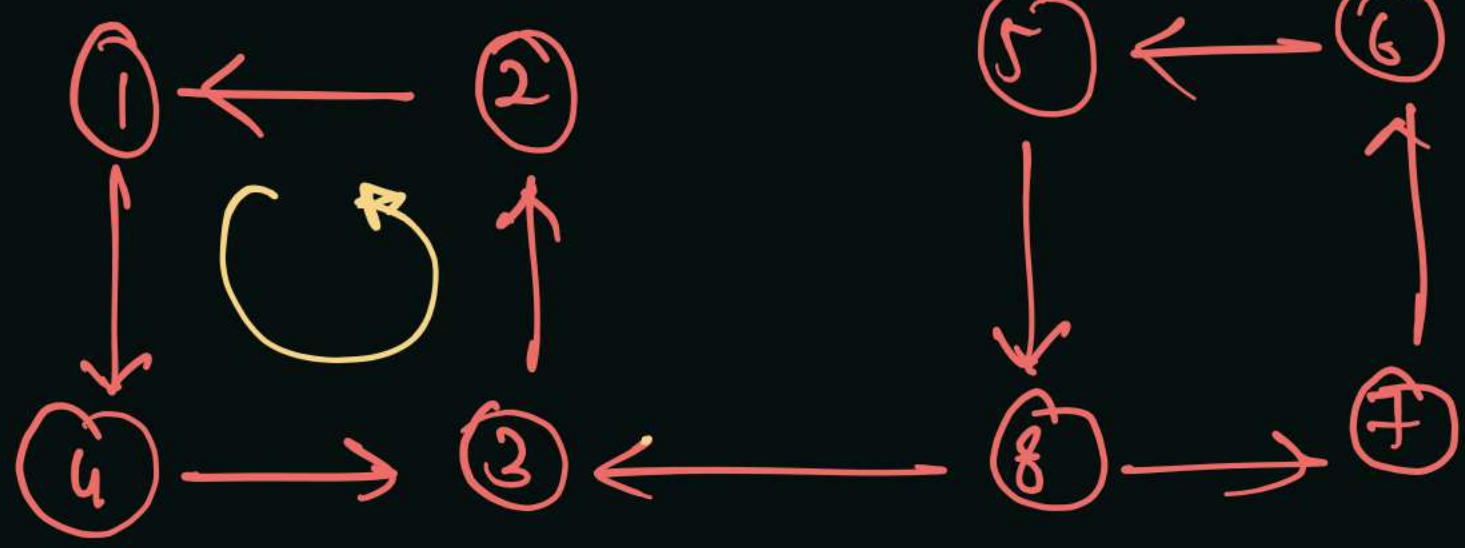
Step ② Reverse all Edge.

Step ③ Iterate DFS in order of Stack's Element and count No. of calls.

Mark.

✓ 1
✓ 2
✓ 3
✓ 4
✓ 5
✓ 6
✓ 7
✓ 8



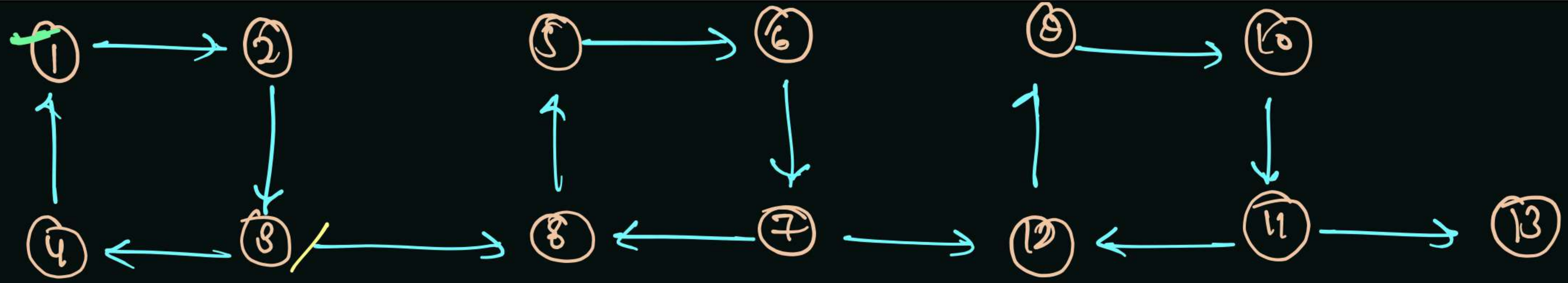


$\underbrace{1\ 4\ 3\ 2}_{\text{call 1}}$

$\underbrace{8\ 7\ 6\ 5}_{\text{call 2}}$

call 1 — call 2 → Count

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



Step-1 Stack fill

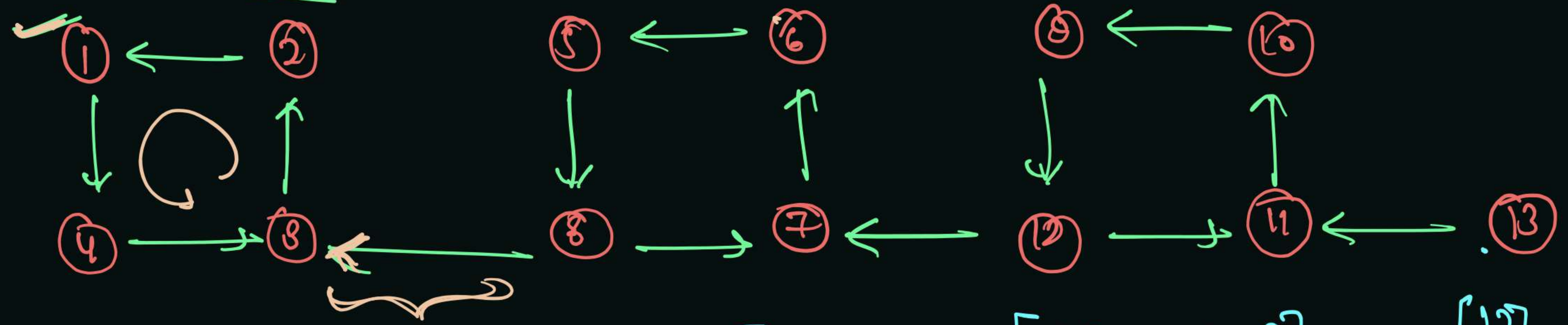
Step.2 - Reverse Edge

1 — DFS
2 ×
3 ×
4 ×
5 ×
6 ×
7 ×
8 ×
9 ×
10 ×
11 ×
12 ×
13 ×

~~1~~ DFS
~~2~~ vis
~~3~~ vis
~~8~~ DFS
~~5~~ vis
~~6~~ vis
~~7~~ vis
~~12~~ DFS
~~9~~ vis
~~10~~ vis
~~11~~ vis
~~13~~ DFS
~~4~~ vis

stack

Reverse Edge.



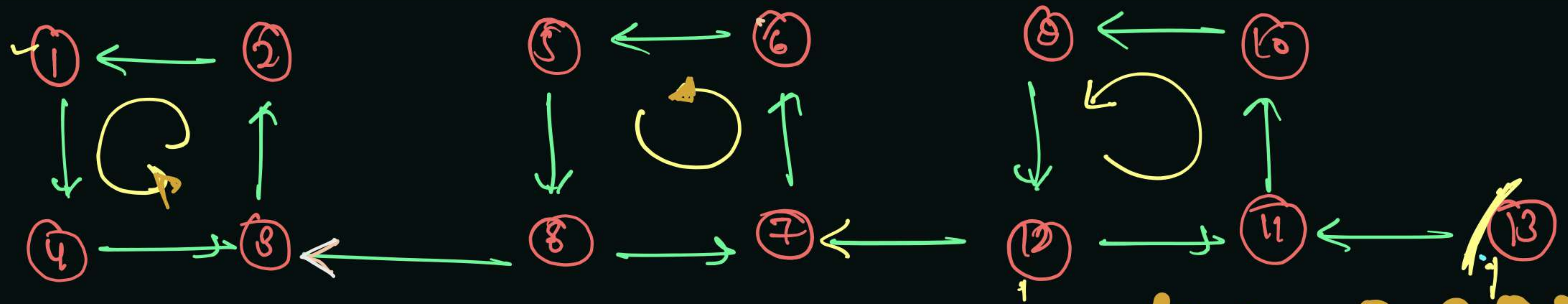
[1, 2, 3, 4] call-1

[8 7 6 5] call2

[12 11 10 9] call3

[13] call4

4 — 4 Strongly connected components.



KOSARAJU ALGO.

✓ ✓ 1 → DFS

✓ ✓ 2 → vis

✓ ✓ 3 → vis

✓ ✓ 4 → vis

✓ ✓ 5 → DFS

✓ ✓ 6 → vis

✓ ✓ 7 → vis

✓ ✓ 8 → vis

✓ ✓ 9 → DFS

✓ 10 → vis

✓ 11 → vis

✓ 12 → vis

✓ 13

Stack

~~13~~ → DFS

~~9~~ → DFS

~~12~~ vis

~~11~~ vis

~~10~~ vis

~~5~~ → DFS

~~8~~ vis

~~7~~ vis

~~6~~ vis

~~1~~ DFS

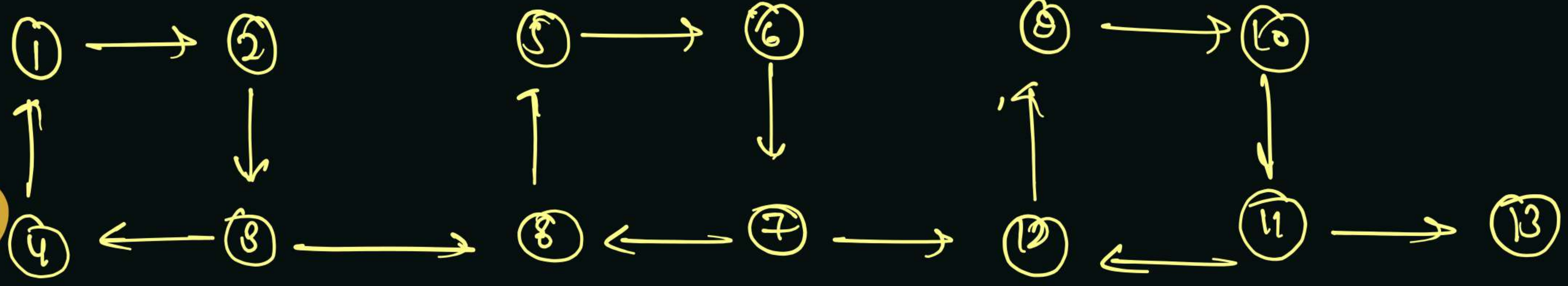
~~4~~ vis

~~3~~ vis

~~2~~ vis

~~stack~~

Reverse Edge



[1 2 3 4] call 4

[5 6 7 8] call 3

[9 10 11 12] call 2

[13] call 1

Total = 4 calls = 4 strongly connected comp