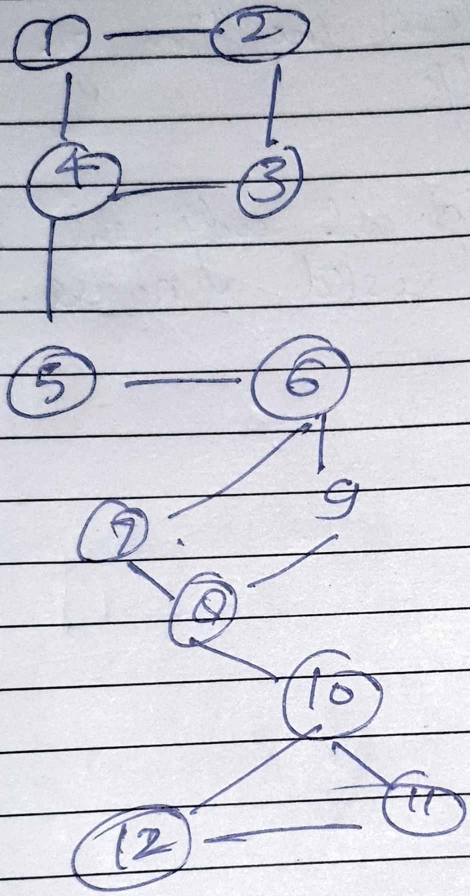
Bridges in graph

Dy ⇒ Pick up any edge and if we removed the picked up edge then the component gets broken into 2 compos

Ex ⇒



Bridges = [[4, 5], [5, 6], [8, 10]]

How to implement algo to find the bridges :

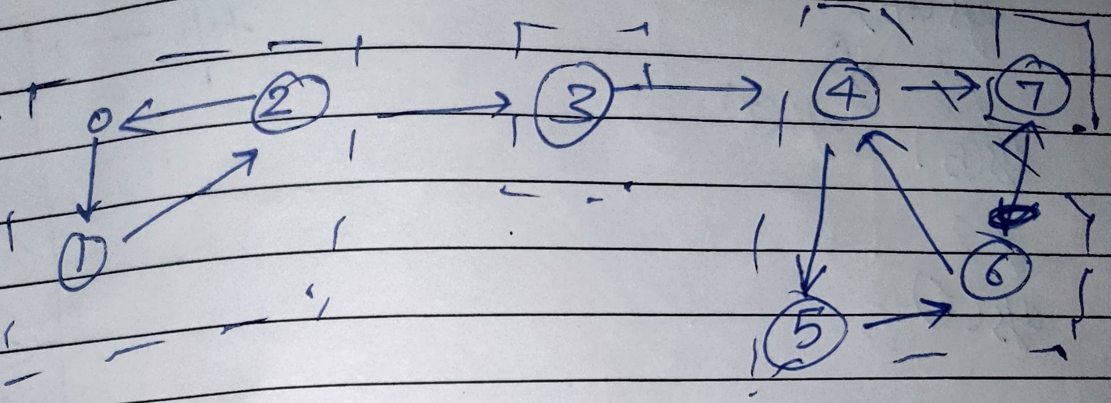
tin[]: the time of insertion on DFS

low[]: min lowest time insertion of all adjacent nodes apart from parent.

Strongly Connected Compo :- (→ Kosaraju's Algo)
(SCC)

Note :- Strongly Connected compo is only valid for ~~connected~~ directed graph.

Ex :-



4 strongly Con compo (SCC) are there in given Example.

SCC1 → SCC2 → SCC3 → SCC3

* (reversal of edges)

SCC1 ← SCC2 ← SCC3 ← SCC4

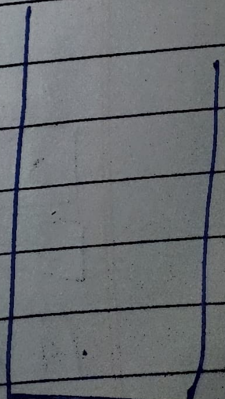
Kosaraju's Algo :- ① Sort all edges acc. to finishing time

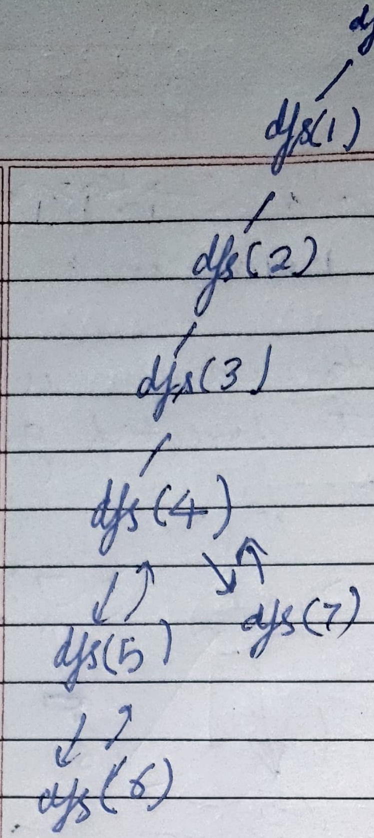
② Reverse Graph

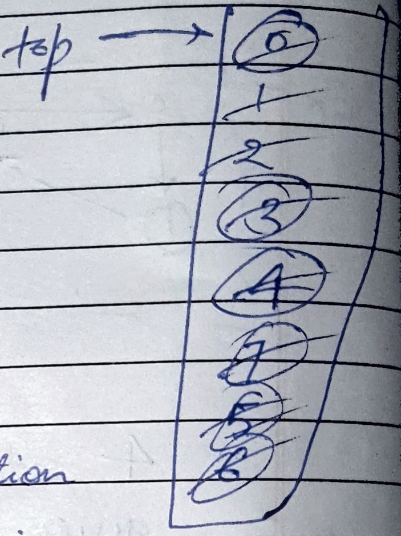
③ ~~St~~ Do a dfs.

Applying Kosaraju's Algo in above example:-

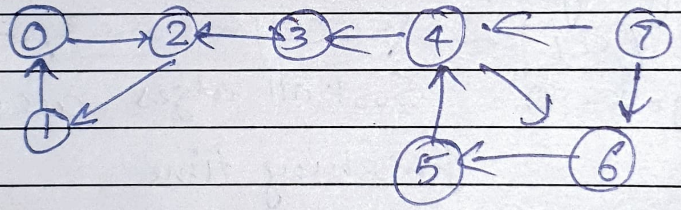
Step:- let's call dfs(0) and storing the nodes according to finishing time

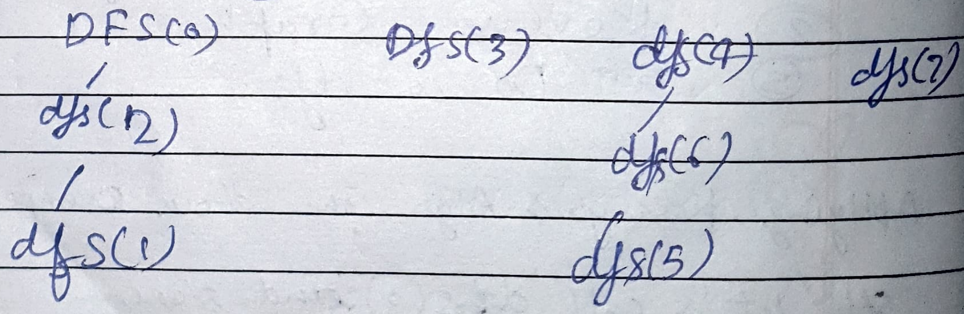






Step-2. Reversing Graph edge direction & applying dfs for node 0.





0 2 1

3

4 6 5

7

Articulation point :— Nodes on whose removal the graph breaks down into multiple compo.

tin[] :— Stores the time of insertion during dfs

low[] :→ min of all adjacent nodes apart from parent & visted nodes.