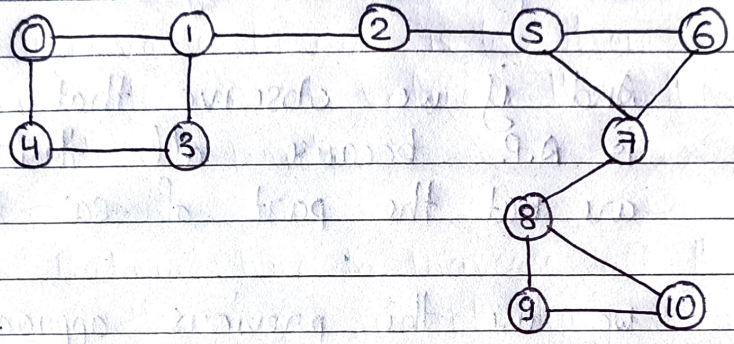
Page :
Date / /

Day - 219

Graph - 23

* Articulation Point :

⇒



⇒ Articulation Point is that node if we removed that node from the graph then our graph will be divided into two components.

⇒ So, here node — ①, ②, ⑤, ⑥, etc are A.P.

⇒ If we normally find ~~every~~ A.P.s then our T.C. will become very high.

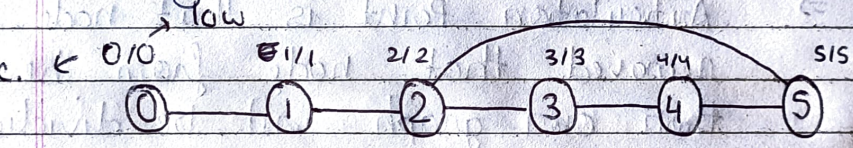
⇒ So, how to solve the question in good time complexity?

Page:
Date: / /

⇒ We can observe that, why ④ is not A.P. because it is the part of cycle or we can say all the neighbours of ④ are the part of the cycle.

⇒ And if we observe that why ① is A.P. because all their neighbours are not the part of ~~sa~~ the cycle.

⇒ We use the previous approach of finding bridges.



⇒ disc[node] <= low[neigh]
⇒ ↳ Articulation Point

⇒ Disc (discovery time)
⇒ At what time, the node is discovered.

⇒ Low : It store the lowest discovery time reachable from current node.

⇒ So, how to start?

⇒ we will apply DFS and fill the disc & low value.

⇒ Then, if we get any visited node, then it will not be a AP.

⇒ Also, we will compare the low of node with disc. of neigh. & select the min.

⇒ When we are returning back then we check for — AP (disc[node] <= low[neigh])
low[node] = min(low[node], low[neigh])

⇒ We will don't check for that node that parent is -1.

⇒ If the root node have more than one child then the root node is A.P.

⇒ Sometimes, AP can be counted twice so that we can maintain a AP vector.