Let's begin at 9:05 PM

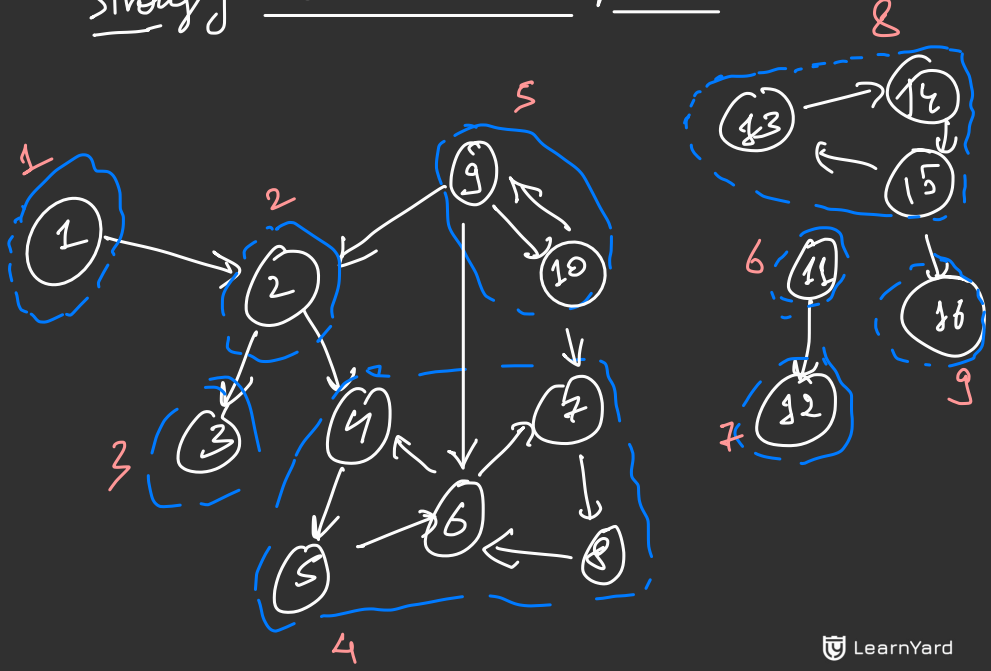
L94
Strongly Connected Components
(Journey back to Directed Graphs)

Join Discord - https://bit.ly/ly-discord

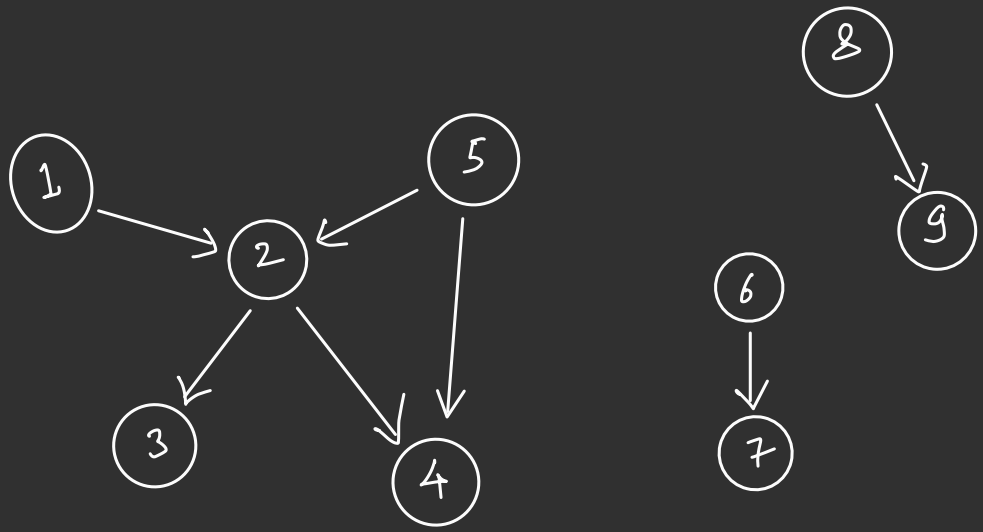
RECAP

Few terminologies &
observations

Strongly connected component



Condensation graph → Always a DAG



Kosaraju's Algorithm

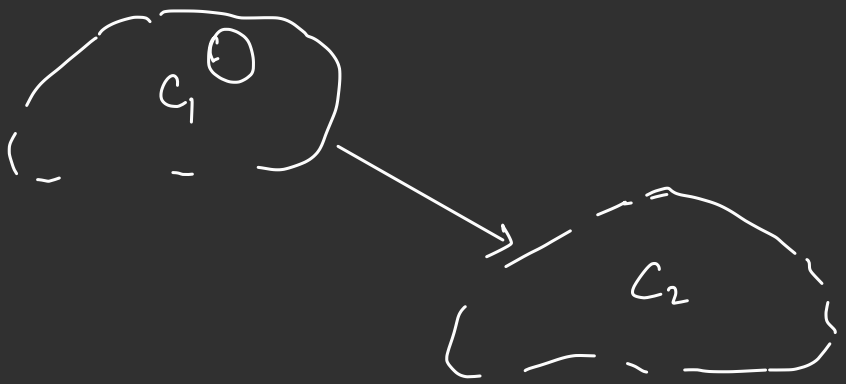
A claim

$tout[C] \Leftarrow$ Max. tout out of all the nodes in the SCC.

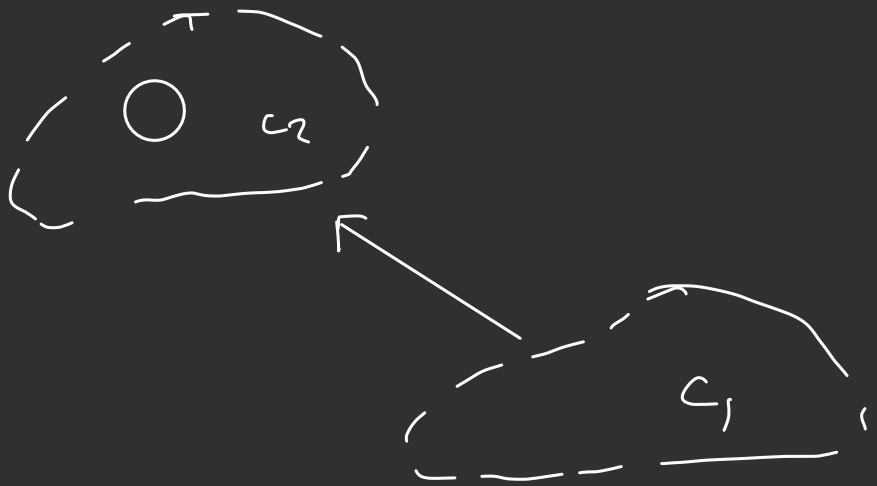
If 

, then $tout[C_1] > tout[C_2]$

Case 1: DFS call is done first from a node in C_1



Case 2 : C_2 first

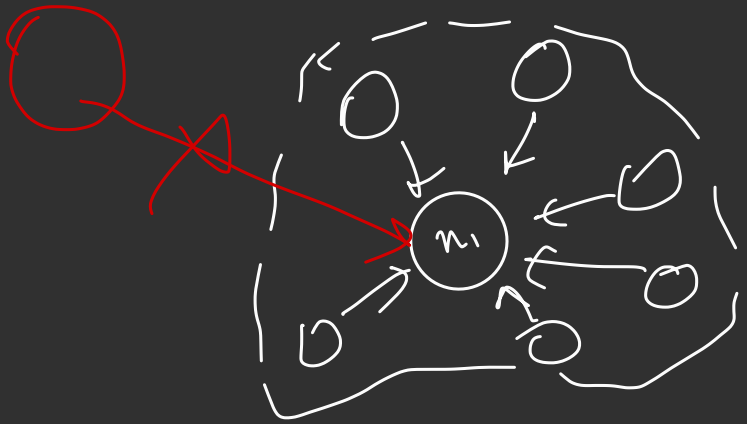


The Algorithm

All the nodes in dec order of tout time.

↳ 1st node ⇒ largest tout time

Apply dfs from n_1 on REVERSE graph



Pseudo-Code

```
{
    for(i : 1 to n) {
        if (!vis[i])
            dfs(i, adj);
    }
    order.reverse();

    for(cur : order) {
        if (!scc[cur])
            dfs2(cur, radj);
    }
}
```

Time Complexity?

$O(N+M)$

A problem (+ implementation)

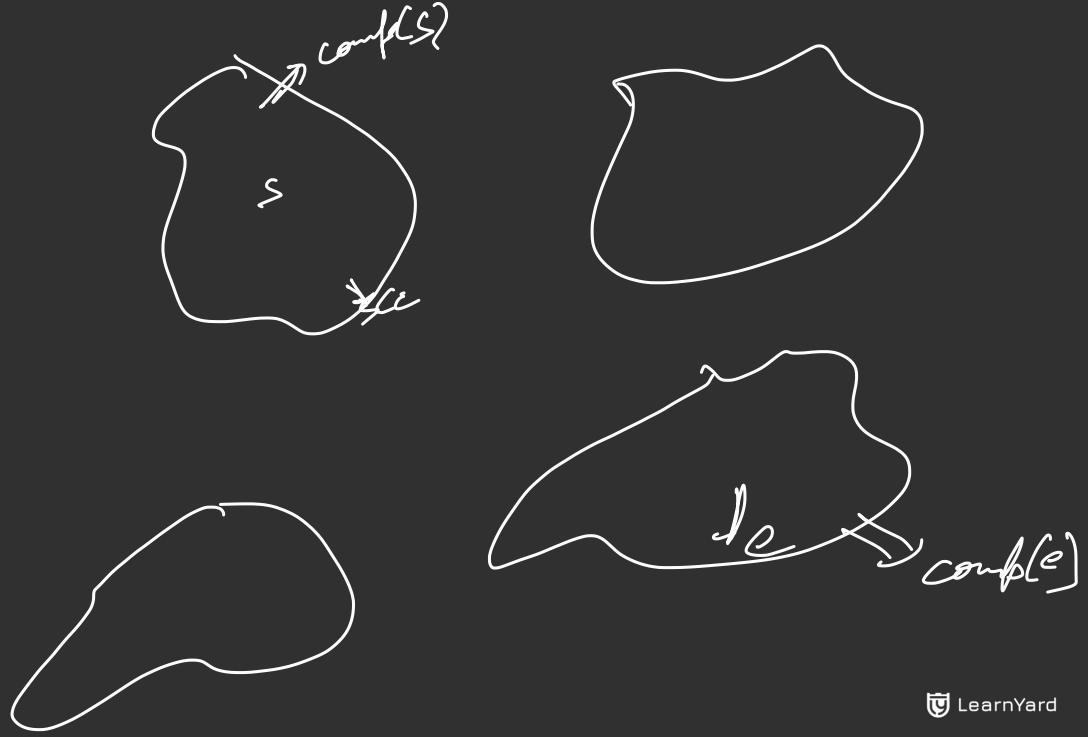
Good Travels

Intuition / Solution

```
int fun(        )
    if (cur == e)
        ans = f[e];

    dp[cur] = -2; // → not possible to reach.

    for(int nb : adj[cur]) {
        if (fun(nb) == -2)
            continue;
        dp[cur] = max(__, f[cur] + fun(nb));
    }
    return dp[cur];
```



$$f_2[comp] = \sum_{i \in C} f[i]$$

Let’s Implement

Thank You!

Reminder: Going to the gym & observing the trainer work out can help you know the right technique, but you'll muscle up only if you lift some weights yourself.

So, PRACTICE, PRACTICE, PRACTICE!