* Course Schedule

* Topological Sort

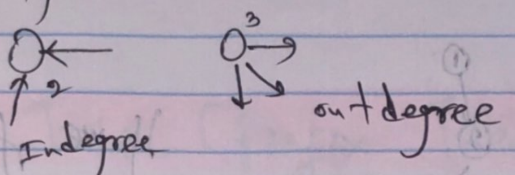
for directed acyclic graphs (DAG) only

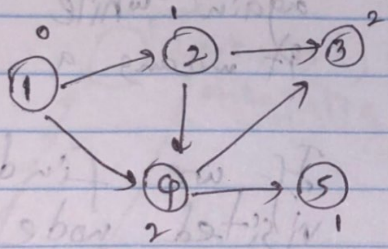
$O(V+E)$

$u \rightarrow v$; u must come before v

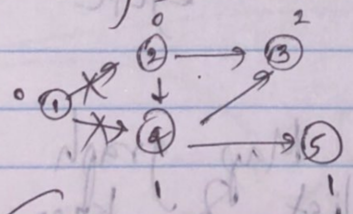
there should be atleast one topological ordering ~~sort~~

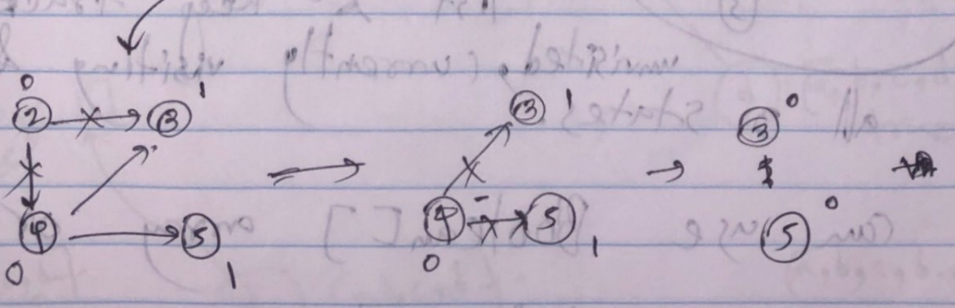
(1) indegree of each vertex





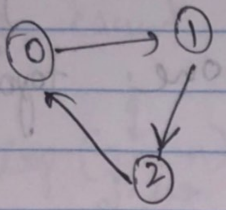
(2) take vertex with indegree 0 & delete those edges





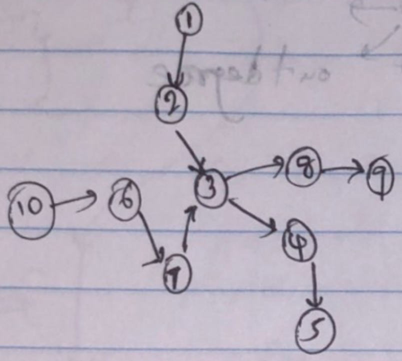
topological sortings: 1, 2, 4, 3, 5 or 1, 2, 4, 5, 3

for
- School class prerequisites
- program dependencies (one package inside another)
- Event Scheduling
- Assembly instructions
- Build systems
-



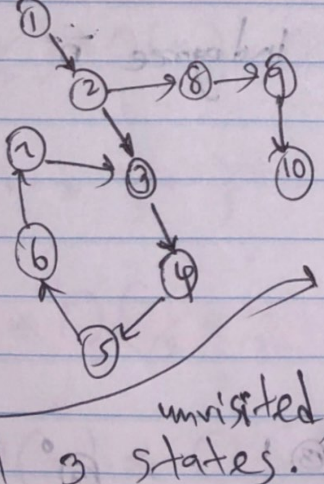
deadlock

•



If we find same element again while doing dfs, it means a cycle.

If we find previously visited node in current dfs, then it means no cycle.



DFS

Map graph to adjacency list & keep track of unvisited, currently visiting & visited all 3 states.

$O(V+E)$
$O(V+E)$

can use Boolean[] array or

set to visited, boolean[] for currently visitings.

No: Date: / /

BFS

use a queue & ~~that~~ keep checking for vertices of 0 indegrees. To that use an array of indegrees, & keep adding to that.

Then keep popping & removing Edges & if queue is empty & if there are remaining edges, ~~then~~ there should be cycles.

$O(V+E)$

$O(V+E)$