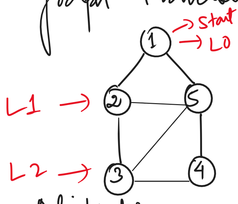
Graph Traversals: → Level Order (BFS) & DFS



Idea / Intuition:

O/P → 1, 2, 5, 3, 4
1, 5, 2, 4, 3

Who are your neighbours?

Starting → 1 2 5 3 4

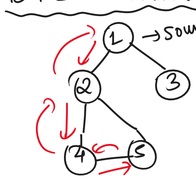
Adj List:

1 → 2, 5
2 → 1, 3, 5
3 → 2, 4, 5
4 → 3, 5
5 → 1, 2, 3, 4

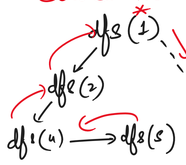
Visited Array

1 →	F T
2 →	F T
3 →	F T
4 →	F T
5 →	F T

DFS Traversal: →



Source Node: → 1, 2, 4, 5, 3
1, 2, 5, 4, 3
1, 3, 2, 4, 5
1, 3, 2, 5, 4



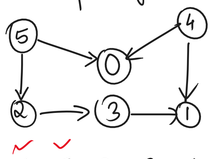
Adj List: →

1 → 2, 3
2 → 1, 4, 5
3 → 1
4 → 2, 5
5 → 2, 4

Visited Array: →

	1	2	3	4	5
Status	T	T	T	T	T
Nodes →	1	2	3	4	5

Topological Sort: → Directed Acyclic Graphs (DAGs)



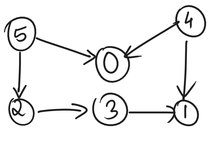
Edges: →

5 - 0
4 - 0
5 - 2
2 - 3
2 - 1
4 - 1

4 5 2 3 1 0

5 4 2 3 1 0

* Linear Ordering: (of vertices)
If there is an edge from u to v, then in the ordering, u always comes before v.



Component wise traversal →
node → node 0, 1, 2
for (0 → n-1) for(3)
dfs(node)

Topo Sort:

0	1	2	3	4	5
1	1	1	1	1	1
∅	∅	∅	∅	∅	∅

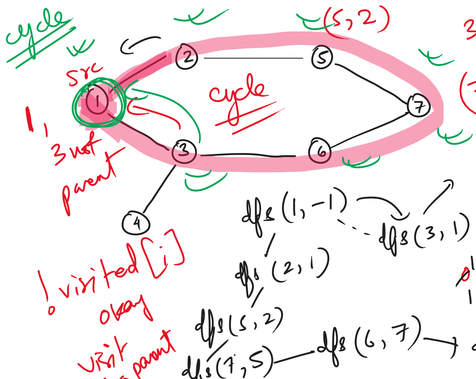
dfs(0)
dfs(1)
dfs(2)
dfs(3)
dfs(4)
dfs(5)

Stack: 5, 4, 2, 3, 1, 0

Adj List

0 →
1 →
2 → 3
3 → 1
4 → 0, 1
5 → 0, 2

O/P: 5 4 2 3 1 0



3 → 1 → visited

1 → 2, 3
2 → 1, 5
3 → 1, 4, 6
4 → 3
5 → 2, 7
6 → 3, 7
7 → 5, 6

(1, 3)
1, 3 parent

{visited but not parent} cycle

visited

1	2	3	4	5	6	7
T	T	T	∅	T	∅	∅