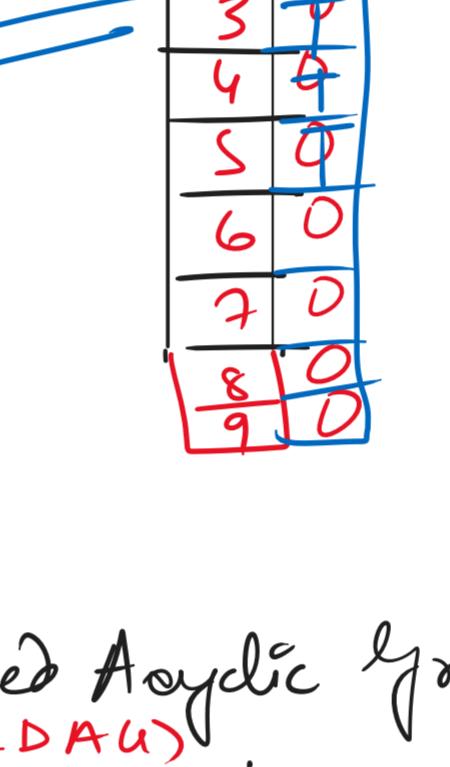


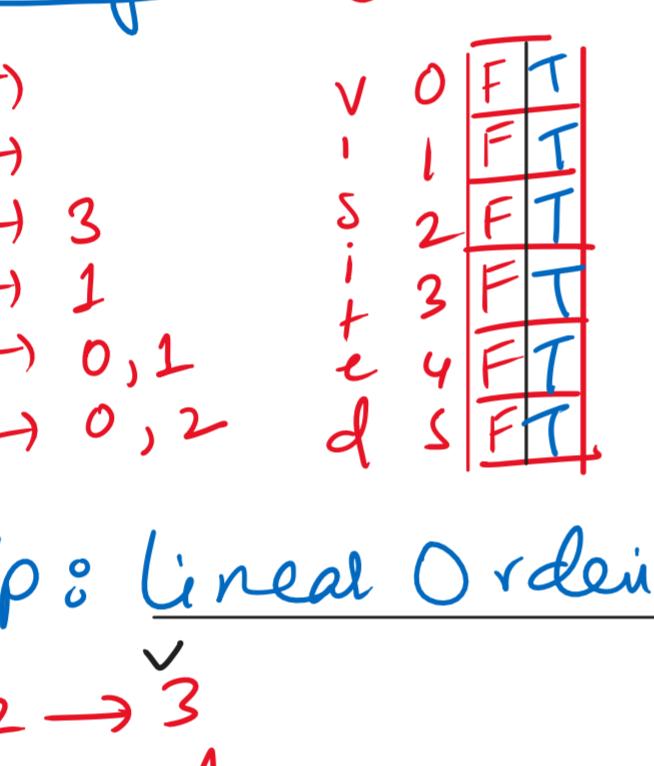
(JSCATE / RNS / SJBIT / DSIT / GATE) (2024/2025) (Synamedia OTT)

- \* Given a graph (undirected) calculate the number of edges of the given graph :

Node 0 → {1, 2}  
Node 1 → {0, 2, 3, 5}  
Node 2 → {0, 1, 4}  
Node 3 → {1, 4}  
Node 4 → {2, 3}



Graph Traversals (DFS & BFS) ⇒ Cycle Detection



recursion queue  
1 3 2 5 4  
1 2 3 4 5  
(who are your neighbors :)  
O/P: 1, 2, 3, 4, 5

adj list: 1 → {2, 3}  
2 → {1, 4, 5}  
3 → {0, 1, 4, 5}  
4 → {1, 2, 3, 5}  
5 → {3, 4}

Vis [ ]  
1 F  
2 F  
3 F  
4 F  
5 F  
S F

queue  
X X X X X

for (int i = 0; i < n; i++)  
 dfs(i);  
 recursion  
 dfs(1) ← dfs(6) → dfs(7)  
 dfs(2) ← dfs(6) → dfs(7)  
 dfs(3) ← dfs(6) → dfs(7)  
 dfs(4) ← dfs(6) → dfs(7)  
 dfs(5) ← dfs(6) → dfs(7)  
 dfs(6) ← dfs(6) → dfs(7)  
 O/P - 1, 2, 3, 4, 5, 6, 7, 8, 9

Adj List:  
1 → 2, 6  
2 → 3  
3 → 4, 5  
4 → 3, 5  
5 → 3, 4  
6 → 1, 7  
7 → 6, 8, 9  
8 → 7, 9  
9 → 7, 8

Visited  
1 0  
2 0  
3 0  
4 0  
5 0  
6 0  
7 0  
8 0  
9 0

Topological Sort: ⇒ (Directed Acyclic Graph) (u → v)  
(Definition: → Linear Ordering of nodes such that if there is an edge from "u" to "v", in the linear ordering "u" always comes before "v".)

Adjacency List: (We do a component-wise traversal) :

0 → v 0 F T  
1 → 1 F T  
2 → 3 S 2 F T  
3 → 1 i 3 F T  
4 → 0, 1 c 4 F T  
5 → 0, 2 d 5 F T

for (int i = 0; i < n; i++)  
 dfs(i);  
 push(i);  
 dfs(i);  
 dfs(2);  
 dfs(2);  
 dfs(1);  
 dfs(1);  
 dfs(0);  
 dfs(0);  
 O/P - 1, 2, 3, 4, 5, 6, 7, 8, 9

After  
dfs(i)  
finishes  
↓  
stack

O/p: Lineal Ordering: (S = 4 = 2 = 3 = 1 = 0) [TS == N]  
S = 4 2 3 1 0  
S = 4 2 3 1 0  
S = 4 2 3 1 0  
S = 4 2 3 1 0  
S = 4 2 3 1 0

Topological Sort: ⇒ (BFS) Kahn's Algorithm: ⇒  
Adj List  
Step 1: Insert all nodes with indegree 0 into the queue  
Step 2: Take out the nodes from the queue & reduce the indegree of neighbours.

(Indegree) → The number of incoming nodes to a particular node is its indegree.

Pseudo code: → if (node u → v) indegree[v]++;

Important Topo Sort Question: →

Given a cyclic graph, detect cycle using Topo sort: →

Adjacency List

empty queue for BFS

if TS.size != N → cycle

Topo Sort O/p: 1 (finished)

if TS.size != N → cycle

Single Source Distance Algo: →

1 Dijkstra's

2 Prim's

3 Kruskal's

4 Floyd Warshall

5 Bellman Ford → (-ve)

\* The edge weight for 1 → 2 & 2 → 4 are negative: →

In case of sales, marks, performance a graph can be negative

Problems: →

1 Number of Provinces

2 Number of Islands

3 Shortest Path

4 Minimum Spanning Tree

5 All Pairs Shortest Path

6 Shortest Path Between Two Vertices

7 Minimum Spanning Tree Using DFS

8 Shortest Path Using BFS

9 Shortest Path Using Dijkstra's

10 Shortest Path Using Bellman Ford

11 Shortest Path Using Floyd Warshall

12 Shortest Path Using Kruskal's

13 Shortest Path Using Dijkstra's

14 Shortest Path Using Prim's

15 Shortest Path Using Bellman Ford

16 Shortest Path Using Floyd Warshall

17 Shortest Path Using Dijkstra's

18 Shortest Path Using Bellman Ford

19 Shortest Path Using Floyd Warshall

20 Shortest Path Using Dijkstra's

21 Shortest Path Using Bellman Ford

22 Shortest Path Using Floyd Warshall

23 Shortest Path Using Dijkstra's

24 Shortest Path Using Bellman Ford

25 Shortest Path Using Floyd Warshall

26 Shortest Path Using Dijkstra's

27 Shortest Path Using Bellman Ford

28 Shortest Path Using Floyd Warshall

29 Shortest Path Using Dijkstra's

30 Shortest Path Using Bellman Ford

31 Shortest Path Using Floyd Warshall

32 Shortest Path Using Dijkstra's

33 Shortest Path Using Bellman Ford

34 Shortest Path Using Floyd Warshall

35 Shortest Path Using Dijkstra's

36 Shortest Path Using Bellman Ford

37 Shortest Path Using Floyd Warshall

38 Shortest Path Using Dijkstra's

39 Shortest Path Using Bellman Ford

40 Shortest Path Using Floyd Warshall

41 Shortest Path Using Dijkstra's

42 Shortest Path Using Bellman Ford

43 Shortest Path Using Floyd Warshall

44 Shortest Path Using Dijkstra's

45 Shortest Path Using Bellman Ford

46 Shortest Path Using Floyd Warshall

47 Shortest Path Using Dijkstra's

48 Shortest Path Using Bellman Ford

49 Shortest Path Using Floyd Warshall

50 Shortest Path Using Dijkstra's

51 Shortest Path Using Bellman Ford

52 Shortest Path Using Floyd Warshall

53 Shortest Path Using Dijkstra's

54 Shortest Path Using Bellman Ford

55 Shortest Path Using Floyd Warshall

56 Shortest Path Using Dijkstra's

57 Shortest Path Using Bellman Ford

58 Shortest Path Using Floyd Warshall

59 Shortest Path Using Dijkstra's

60 Shortest Path Using Bellman Ford

61 Shortest Path Using Floyd Warshall

62 Shortest Path Using Dijkstra's

63 Shortest Path Using Bellman Ford

64 Shortest Path Using Floyd Warshall

65 Shortest Path Using Dijkstra's

66 Shortest Path Using Bellman Ford

67 Shortest Path Using Floyd Warshall

68 Shortest Path Using Dijkstra's

69 Shortest Path Using Bellman Ford

70 Shortest Path Using Floyd Warshall

71 Shortest Path Using Dijkstra's

72 Shortest Path Using Bellman Ford

73 Shortest Path Using Floyd Warshall

74 Shortest Path Using Dijkstra's

75 Shortest Path Using Bellman Ford

76 Shortest Path Using Floyd Warshall

77 Shortest Path Using Dijkstra's

78 Shortest Path Using Bellman Ford

79 Shortest Path Using Floyd Warshall

80 Shortest Path Using Dijkstra's

81 Shortest Path Using Bellman Ford

82 Shortest Path Using Floyd Warshall

83 Shortest Path Using Dijkstra's

84 Shortest Path Using Bellman Ford

85 Shortest Path Using Floyd Warshall

86 Shortest Path Using Dijkstra's

87 Shortest Path Using Bellman Ford

88 Shortest Path Using Floyd Warshall

89 Shortest Path Using Dijkstra's

90 Shortest Path Using Bellman Ford

91 Shortest Path Using Floyd Warshall

92 Shortest Path Using Dijkstra's

93 Shortest Path Using Bellman Ford

94 Shortest Path Using Floyd Warshall

95 Shortest Path Using Dijkstra's

96 Shortest Path Using Bellman Ford

97 Shortest Path Using Floyd Warshall

98 Shortest Path Using Dijkstra's

99 Shortest Path Using Bellman Ford

100 Shortest Path Using Floyd Warshall

101 Shortest Path Using Dijkstra's

102 Shortest Path Using Bellman Ford

103 Shortest Path Using Floyd Warshall

104 Shortest Path Using Dijkstra's