Graphs 4

Take action in next 15 days, Work hard revise & give mock Interviews

Good Evening

Agenda

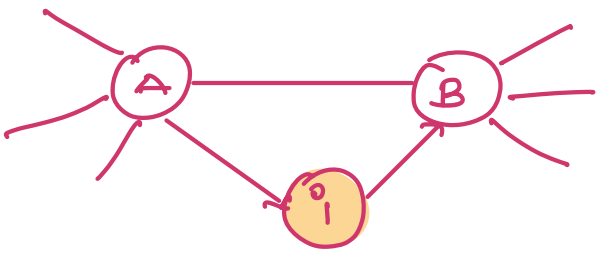
01. Floyd Warshall Algo
02. Graph coloring
03. Rotten oranges

Bellman ford Algo

If negative wt cycle is present, we can't find shortest dist

01. Find shortest distance from every node to every other node.

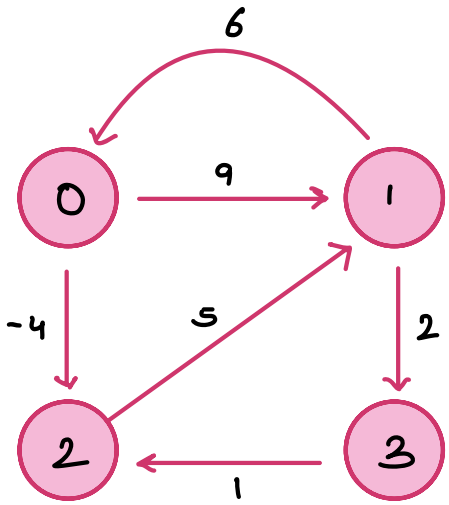
Floyd Warshall Algo → All pairs shortest path



Idea

↳ Consider every node as intermediate node & try to relax the edges with larger wts

ignore



	0	1	2	3
0	0	9	-4	∞
1	6	0	∞	2
2	∞	5	0	∞
3	∞	∞	1	0

Adjacency matrix

mat[i][j] = wt of direct edge b/w u to v

01. 0 → intermediate node

	0	1	2	3
0	0	9	-4	∞
1	6	0	2	2
2	∞	5	0	∞
3	∞	∞	1	0

$d[u][0] + d[0][v] < d[u][v]$

update it

✓ $d[1][0] + d[0][2] < d[1][2]$

× $d[1][0] + d[0][3] < d[1][3]$

× $d[2][0] + d[0][1] < d[2][1]$

× $d[2][0] + d[0][3] < d[2][3]$

$d[3][0] + d[0][1] < d[3][1]$

$d[3][0] + d[0][2] < d[3][2]$

* 1 - intermediate

	0	1	2	3
0	0	9	-4	11
1	6	0	2	2
2	11	5	0	7
3	∞	∞	1	0

$d[u][i] + d[i][v] < d[u][v]$

* 2 - intermediate

	0	1	2	3
0	0	1	-4	3
1	6	0	2	2
2	11	5	0	7
3	12	6	1	0

$d[v][2] + d[2][v] < d[v][v]$

$d[0][2] + d[2][1] < d[0][1]$

$-4 + 5 < 9$

* 3 - intermediate

	0	1	2	3
0	0	1	-4	3
1	6	0	2	2
2	11	5	0	7
3	12	6	1	0

```
for (i=0; i<n; i++) {
    for (u=0; u<n; u++) {
        for (v=0; v<n; v++) {
            if (d[u][i] + d[i][v] < d[u][v]) {
                d[u][v] = d[u][i] + d[i][v];
            }
        }
    }
}
return d[][];
```

TC: $O(n^3)$

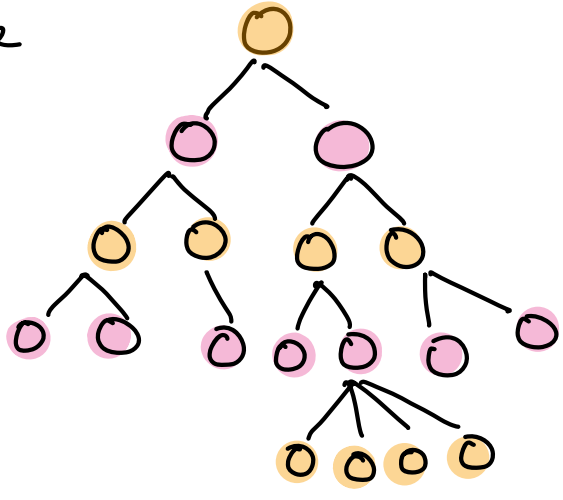
SC: $O(1)$

* Graph coloring

* Francis Guthrie (1852)

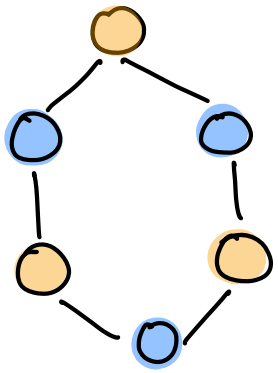
* Minimum no. of colors required to colour all the nodes of a graph such that no two nodes have the same color → Chromatic Number

01 Tree

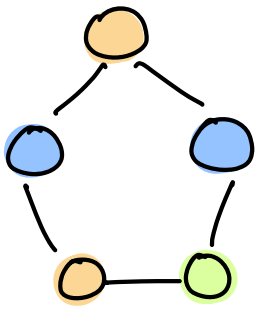


chromatic no. = 2

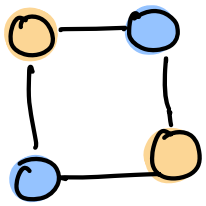
02. Cycle graph



CN = 2



CN = 3



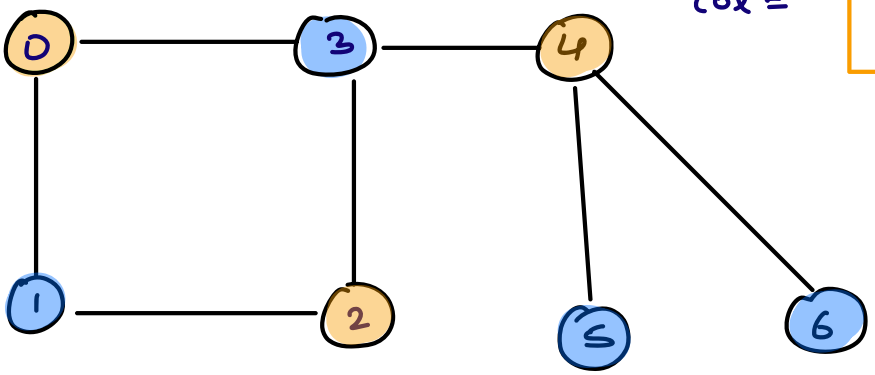
CN = 2

In general = CN ⇒ 2 + N % 2

* Bi-partite Graph

✓→ Any graph with chromatic no. ⇒ 2 Eg - tree, even length cycle graph

✓→ A graph is called bipartite if we can divide all the nodes in two sets such that all the edges are across the sets

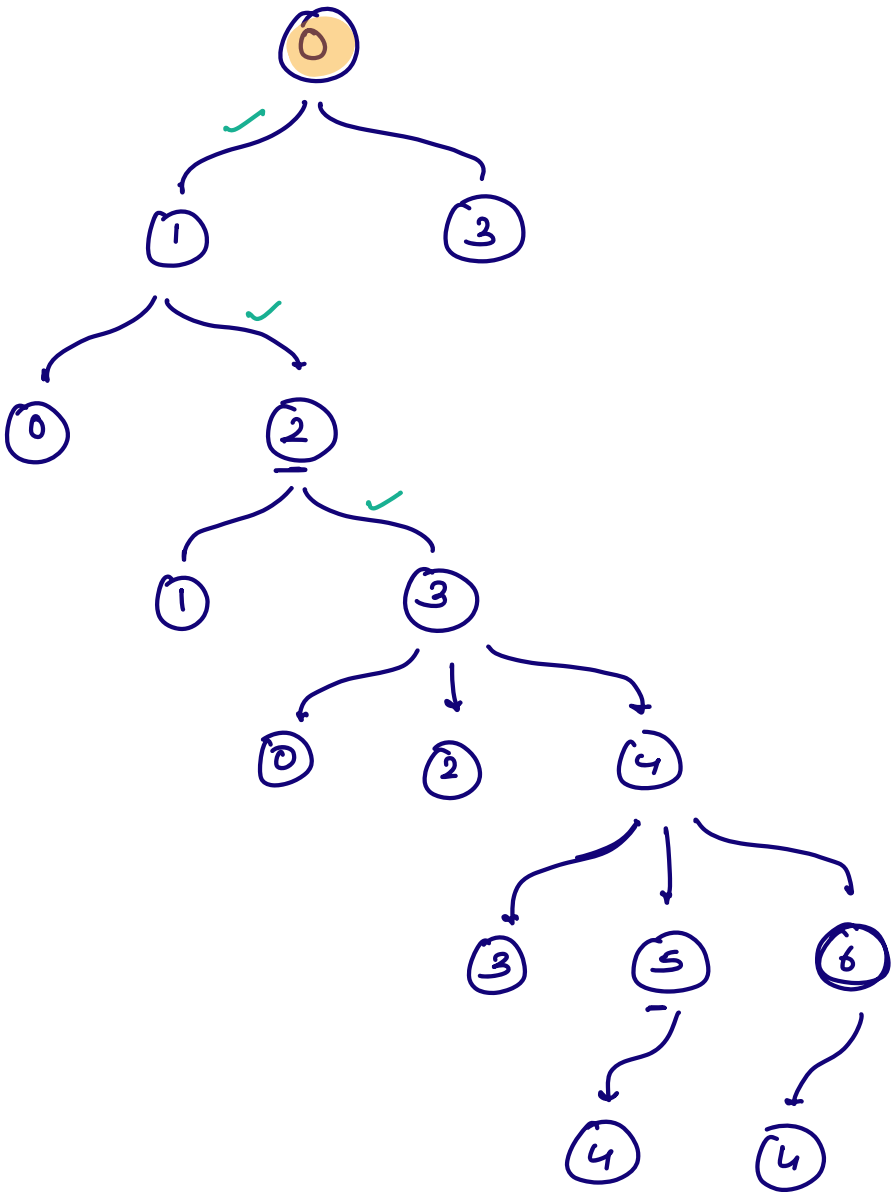


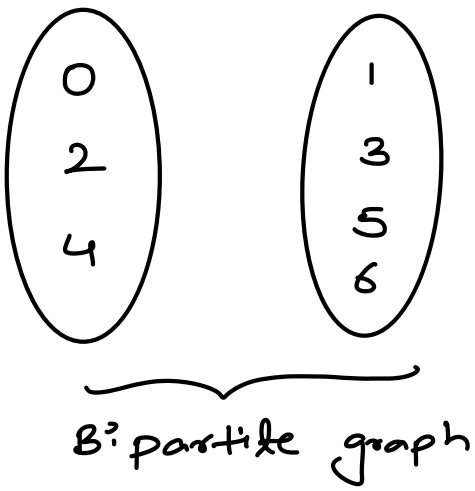
col =

0	1	2	3	4	5	6
0	1	0	1	0	1	1

0 → orange color

1 → blue color





Bipartite graph

* col[N] → col, fill it with -1.

col[src] = 0;

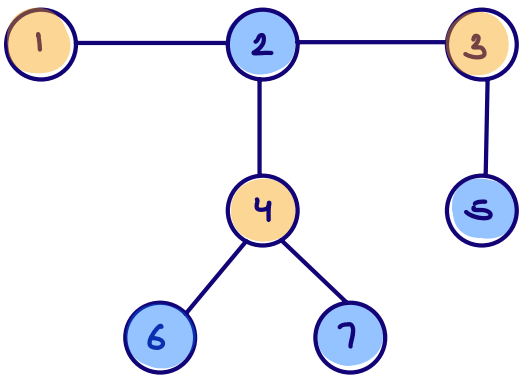
```
boolean dfs(src){
    for(int nbr : graph[src]){
        if(col[src] == col[nbr]) return false;
        else if(col[nbr] == -1){
            col[nbr] = 1 - col[src]
            if(dfs(nbr) == false){
                return false;
            }
        }
    }
    return true;
}

for(i=0; i<N; i++){
    if(col[i] == -1){
        col[i] = 0
        if(dfs(i) == false) return false
    }
}
return true;
```

checking for all components

10:36 pm → 10:46 pm

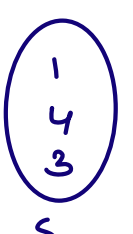
Q A country consist of N cities connected by (N-1) roads King of that country wants to construct maximum roads such that cities can be divided into two sets & there is no road between cities in same set. Find maximum no. of new roads that can be created ?

Note → All cities can be visited from any city.

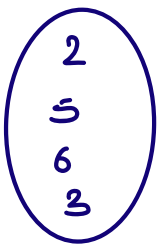


{ 7 - cities
6 - roads }

0	1	0	0	1	1	1
1	2	3	4	5	6	7



S_1



S_2

Max no. of rods possible = maximum no. of edges across two sets

Ans → set$_1$ * set$_2$ = { no. of nodes with col=0 * no. of nodes with col=1 } − (N−1)

Q Given a matrix of integers containing

0 → empty

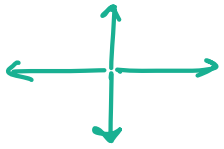
1 → fresh orange

2 → rotten oranges

Every minute, any fresh orange adjacent to a rotten orange becomes rotten.

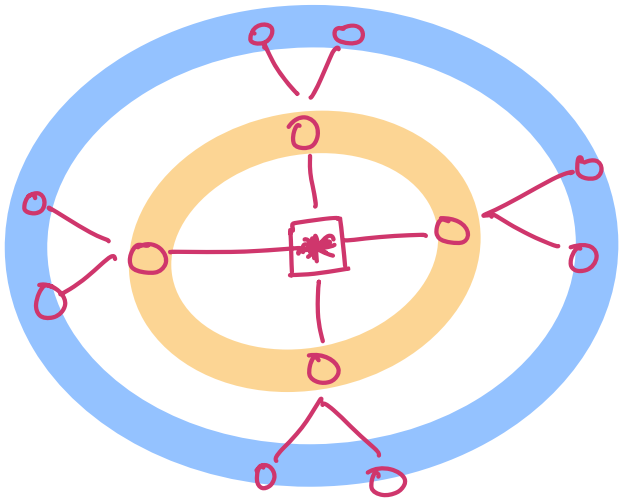
In how many minutes, will all oranges become rotten? If not possible, return -1.

	0	1	2	3	4
0	1	0	1	0	1
1	1	1	1	1	1
2	0	2	0	1	0
3	0	1	1	1	1
4	1	1	1	0	0



t = 0 1 2 3 4 5

t = 5 min

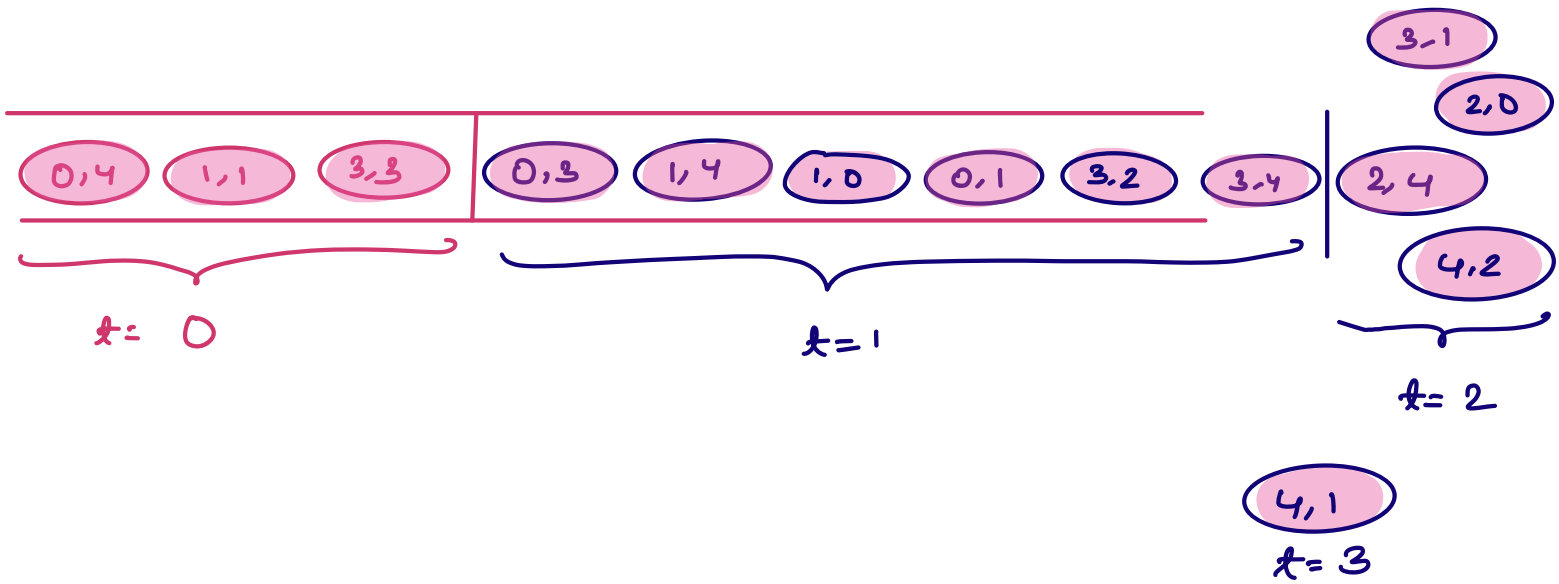


Multisource BFS

	0	1	2	3	4
0	0	2	0	2	2
1	2	2	0	0	2
2	2	0	0	0	2
3	0	2	2	2	2
4	0	2	2	0	0

$t = 0 \; or \; 1 \; or \; 2 \; or \; 3$

$t = 3$



// Insert every 2 in queue.

```
while (q.size()>0){
    int sz = q.size();
    for (i=1; i<=sz; i++){
        rp= q.deque();
        // explore all 4 nbrs
        // if any fresh orange = 1 is present,
           make orange = 2 & insert in your queue
    }
    t = t+1;
}
// check if any fresh orange is left or not.
return t-1;
```

* SQL → 10-11 classes

DSA

→ Mock Interviews

Contest → Leetcode

✓