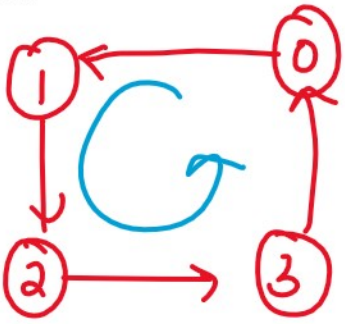
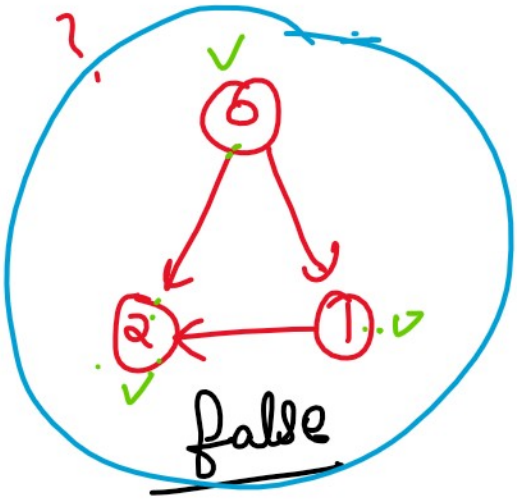
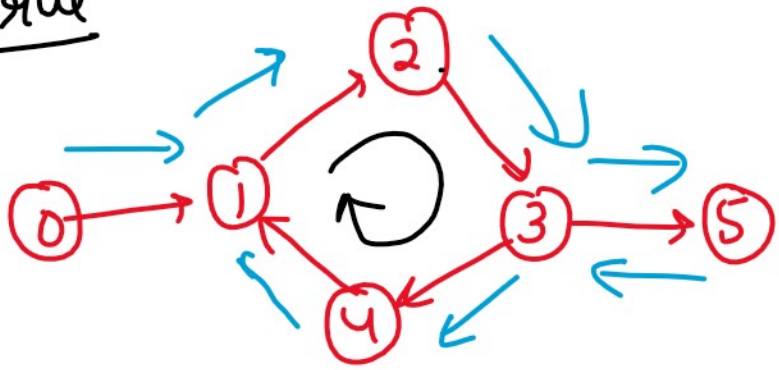
Detect cycle in directed graph

24 September 2023 13:05





True

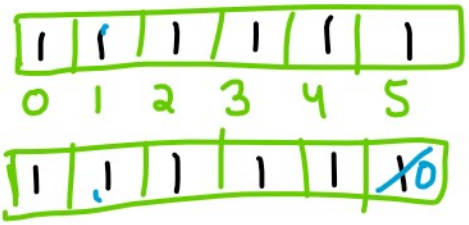


true

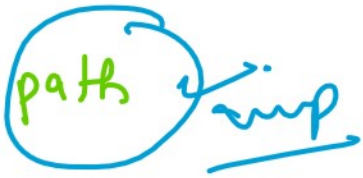
2 arrays.

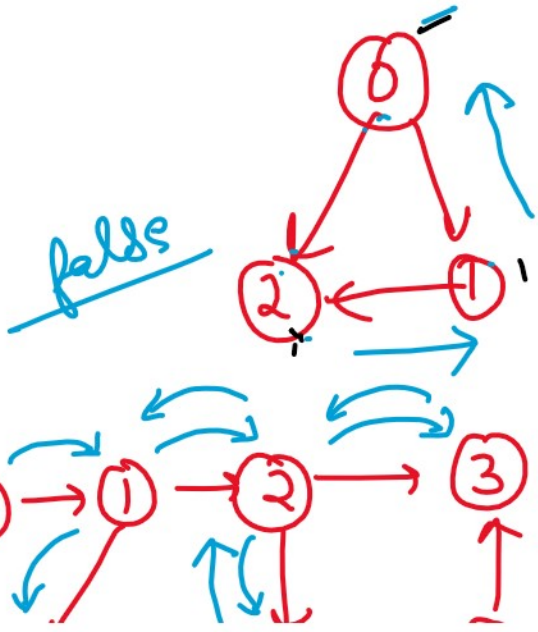
visited, path

true

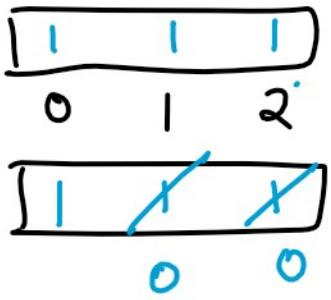


visited





false



visited

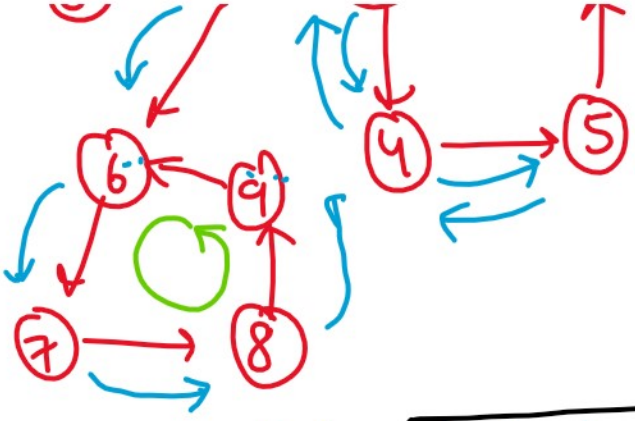
path

adj list

0 → 1

1 → 2, 6

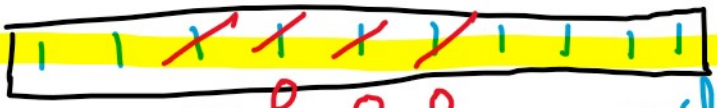
2 → 3, 4



2 → 3,4
3 → X
4 → 5
5 → 3
6 → 7
7 → 8
8 → 9
9 → 6

visited

path

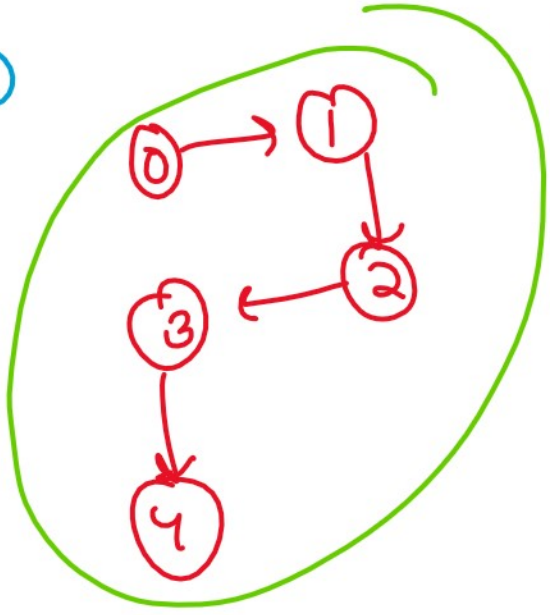


cyclic graph

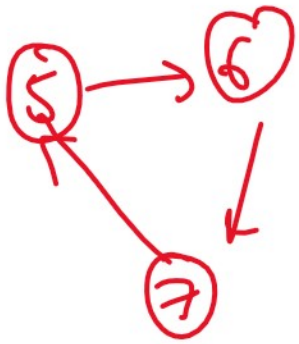
(V+E)

Time : O(V+E)
Space : O(V)

1



2

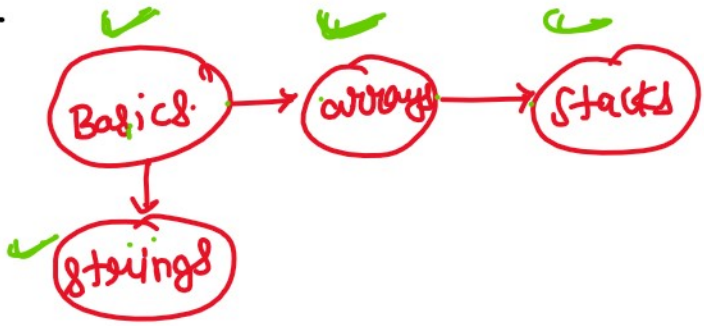


Topological sorting

24 September 2023 13:44

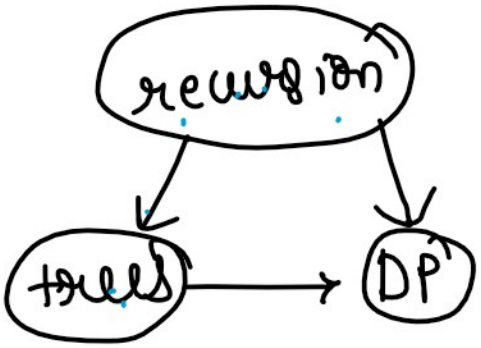
$x \longrightarrow y$

x should always come before y

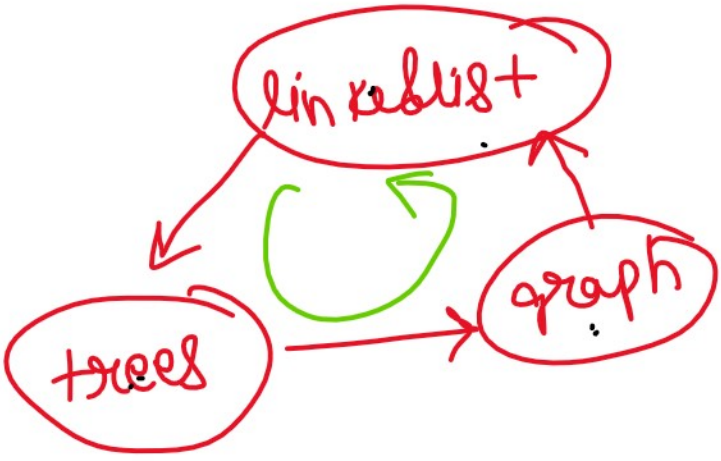


Basics → strings → arrays → stacks

basics → arrays → stacks → strings



recursion → trees → DP



DAG

Directed acyclic graph.

trees.

graph

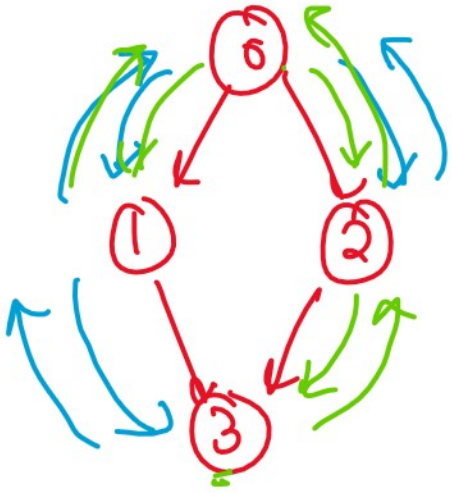
linkedlist

graph

linkedlist+

x → y

x should come before y.



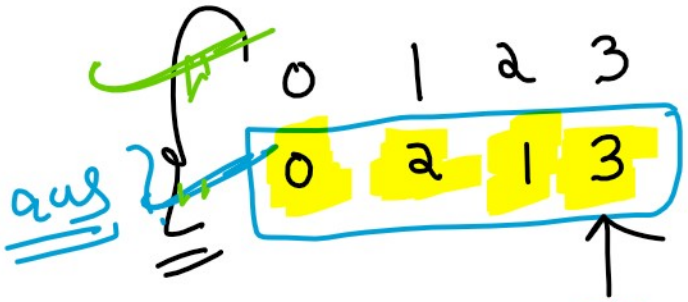
graph.

0 → 1, 2

1 → 3

2 → 3

3 → X

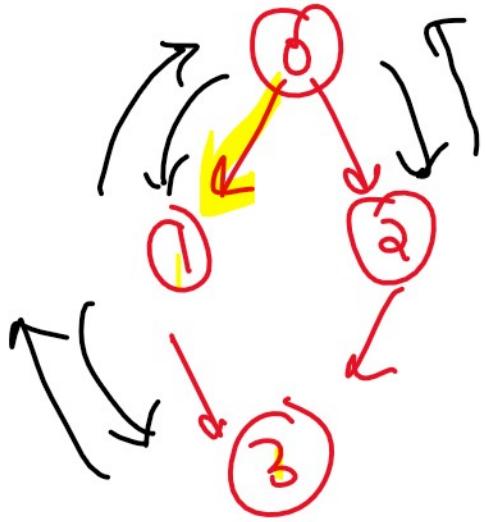


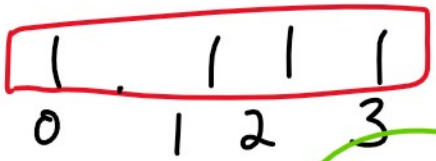
last in first out

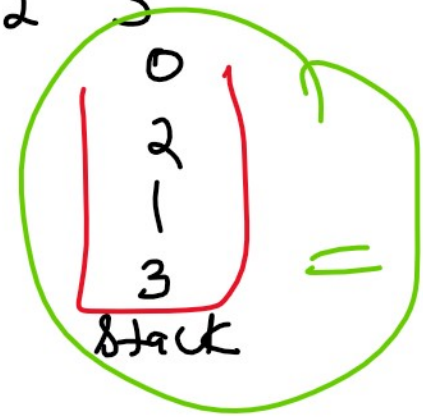
↓

0 1 2 3

stacks



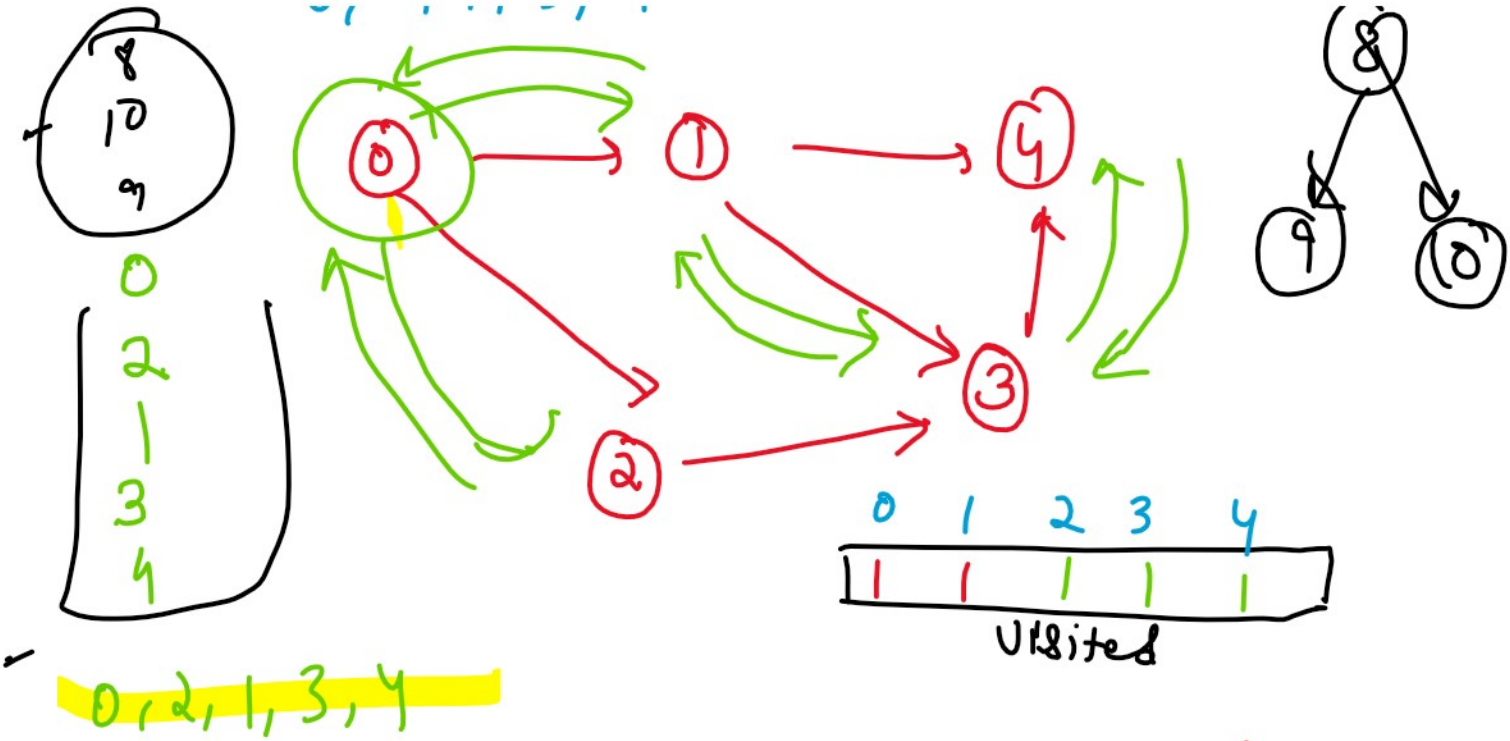




ans

0, 2, 1, 3

0, 2, 1, 3, 4



0, 2, 1, 3, 4

1. Cycle detection in undirect graph.
2. Course schedule - I, II

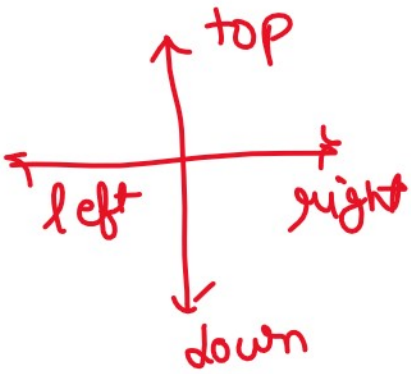
Rotting oranges

24 September 2023 14:15

find min time to rot all fresh oranges.

0 → empty
1 → fresh
2 → rotten

2	2 1	0
2 1	2 1	0
0	2 1	2 1



Time = ~~0 1 2 3~~ 4

2	1 2	0
2 1	0	0
0	1	0

Time = ~~0~~ 1

-1

	0	1	2	3
0	0	~~1~~ 2	0	1 2
1	2	~~1~~ 2	1 2	2
2	0	0	0	1 2
3	2	1 2	2 1	~~1~~ 2

Time = ~~0~~ 1 2

bfs

dfs

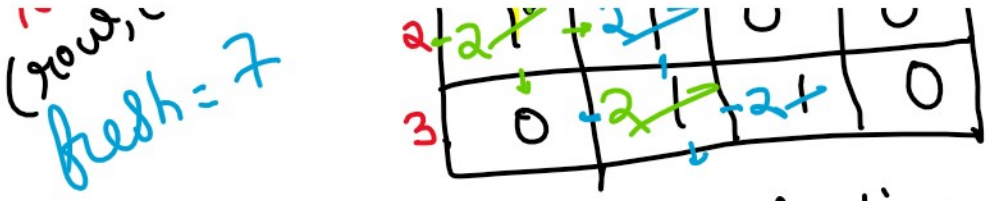
queue

insert all rotten oranges (row, col, time)

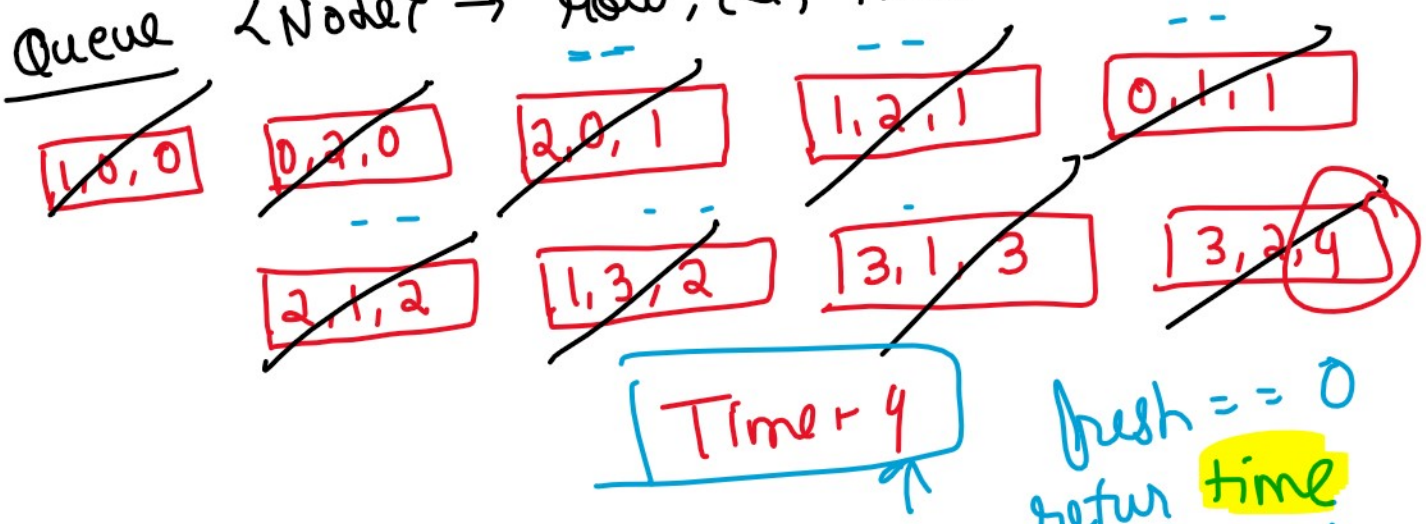
	0	1	2	3
0	0	2 1	2	0
1	2	0	1 2	1 2
2	2 1	2 1	0	0

Time = 4

= ~~0~~ 1 2



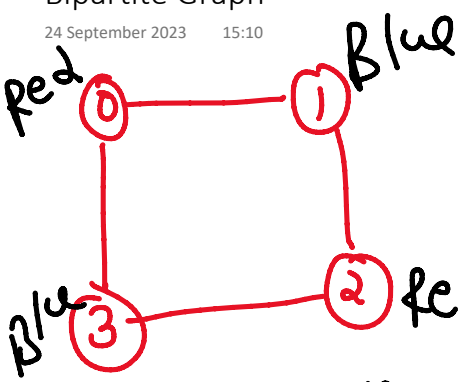
Queue <Node> → row, col, time.



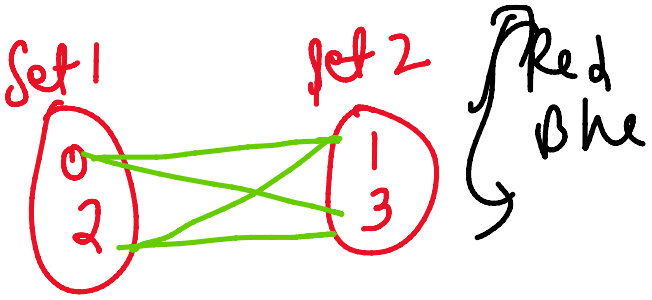
fresh == 0
retur time
else retur -1

Bipartite Graph

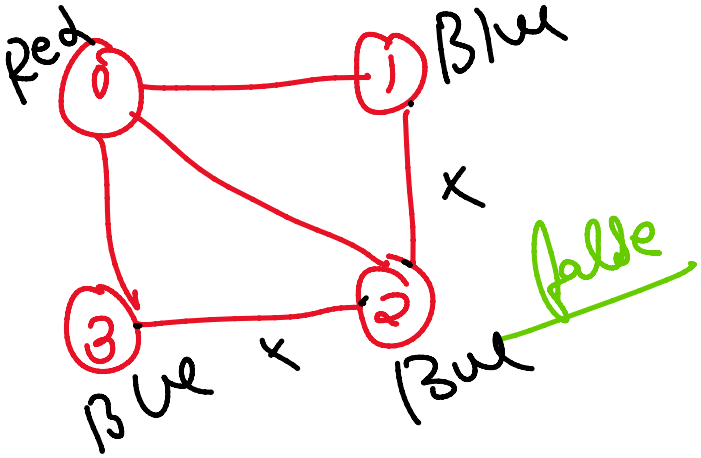
24 September 2023 15:10

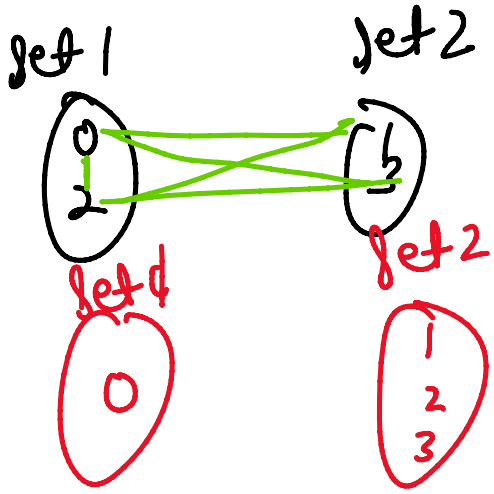


true

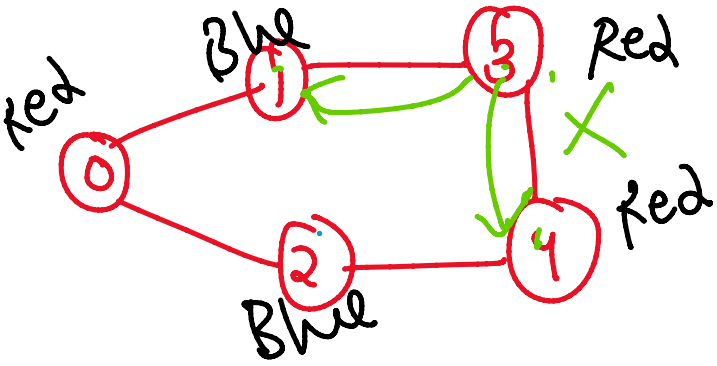


vertices on the same set shouldn't have any edges b/w them. (0-2) (1-3)





1-2, 2-3



false

0 → 1, 2

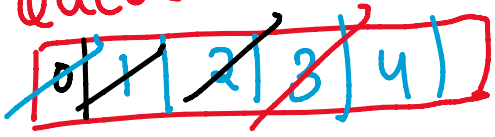
1 → 0, 3

2 → 0, 4

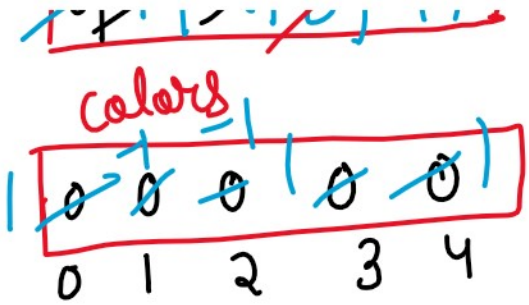
3 → 1, 4

4 → 2, 3

Queue



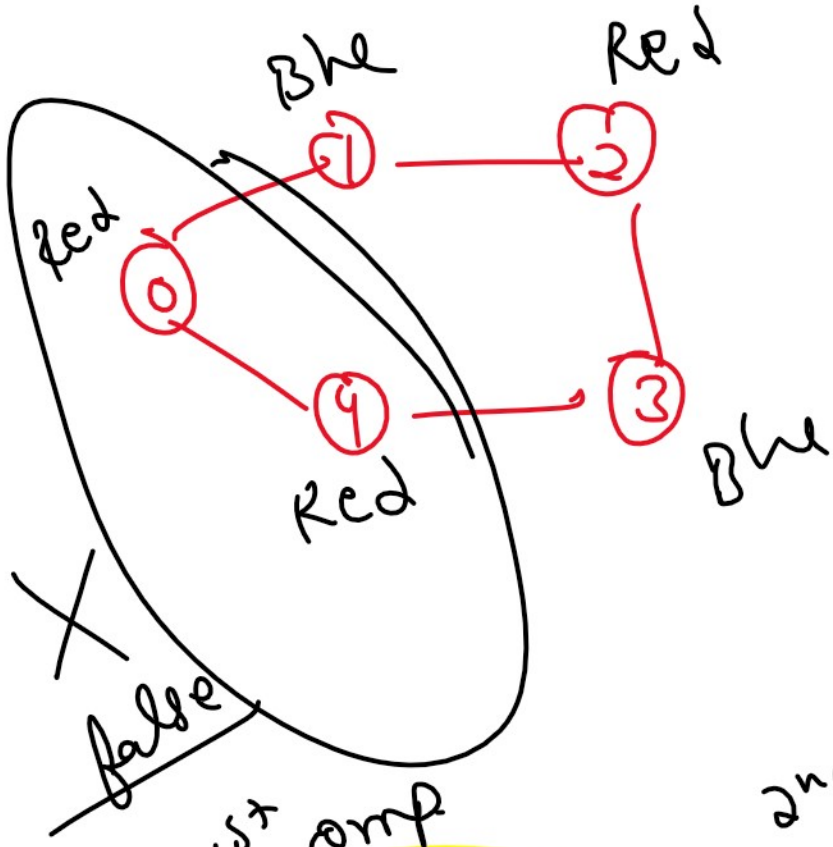
colors.

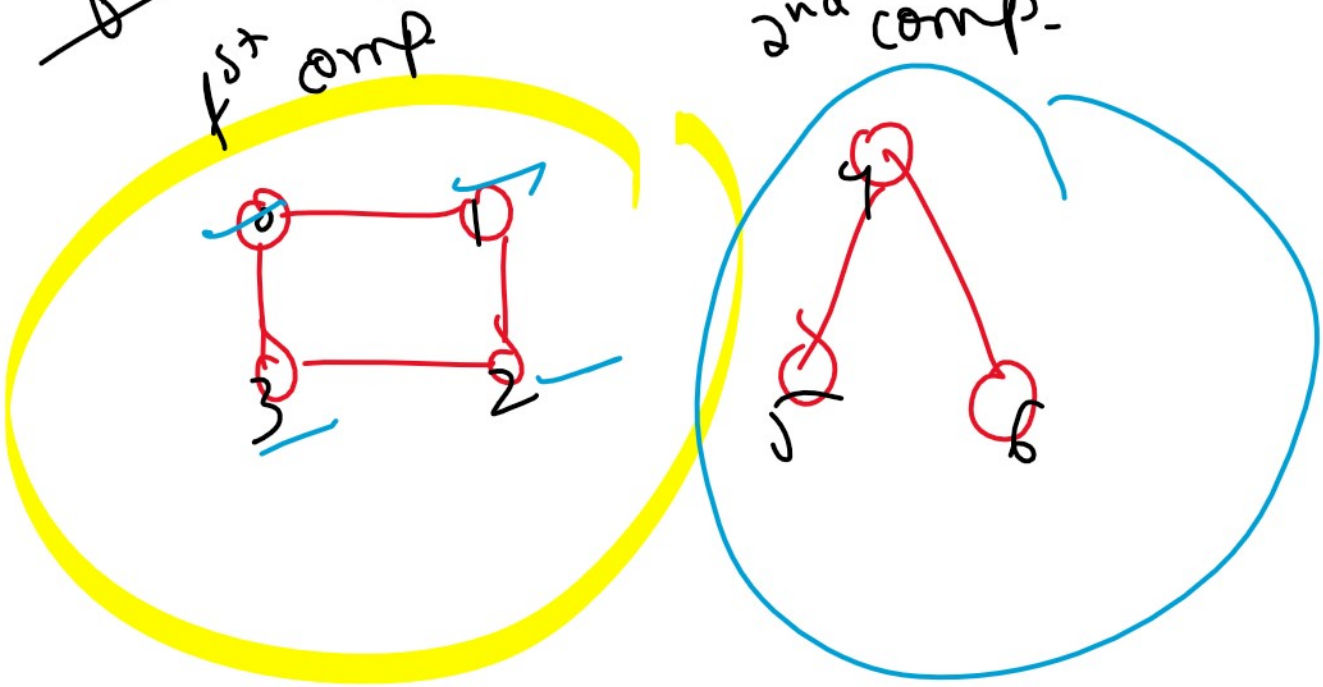


4 → 2, 3

adj list

0 → unvisited
1 → Red
-1 → Blue





Dijkstra algorithm

24 September 2023 15:31

source = 2

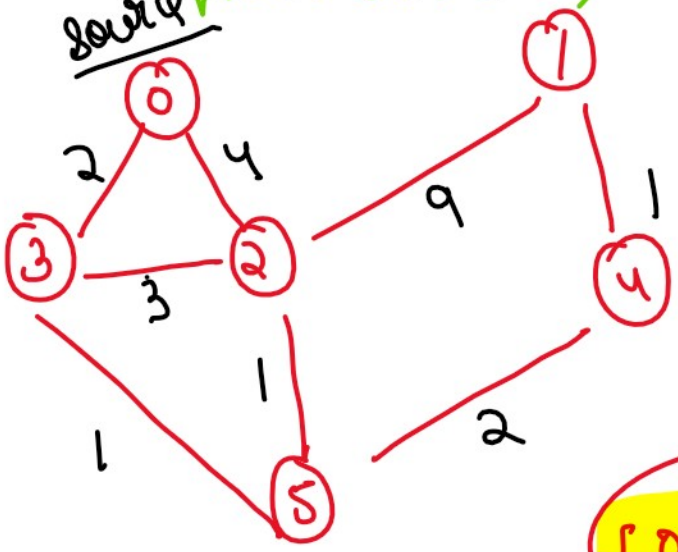
vertices	Distance
0	4
1	1
2	0

min (6, 3+1)

min (1, 6+3)

[4, 1, 0] and

source = 0



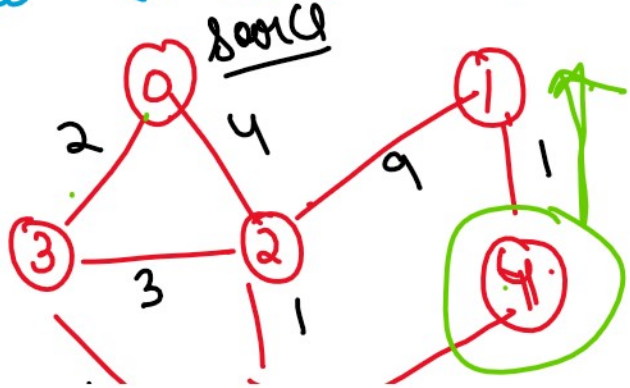
vertex	dist
0 →	0
1 →	6
2 →	4
3 →	2
4 →	5
5 →	3

[0, 6, 4, 2, 5, 3] and

path1 = (0-2) + (2-1) = 4+9 = 13

path2 = (0-2) + (2-5) + (5-4) + (4-1) = 4 + 1 + 2 + 1 = 8

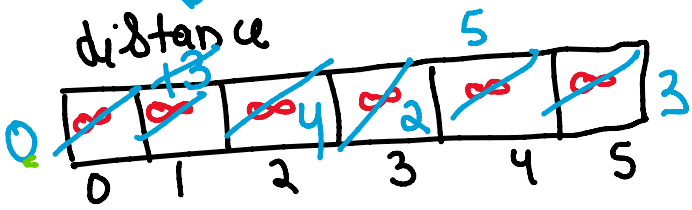
path3 = (0-3) + (3-5) + (5-4) + (4-1) = 2 + 1 + 2 + 1 = 6



min Heap (type)?

weight, vertex

{2, 3}, {4, 2}



4+1=5

2+1=3

3+2=5

4+9=13

5+1=6

[0, 6, 4, 2, 5, 3]