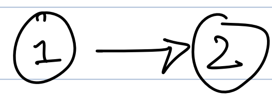
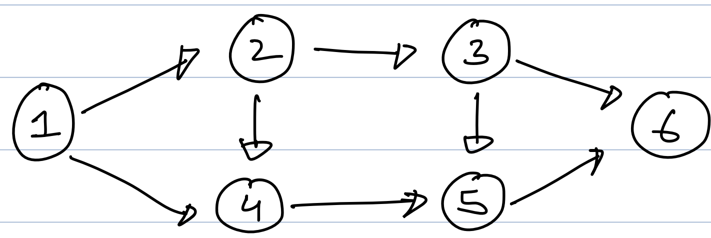
Topological Sorting → DAG [Directed Acyclic graphs]

Linear Ordering of nodes such that if there is a path from node i to node j, then i shall come before j in the ordering.

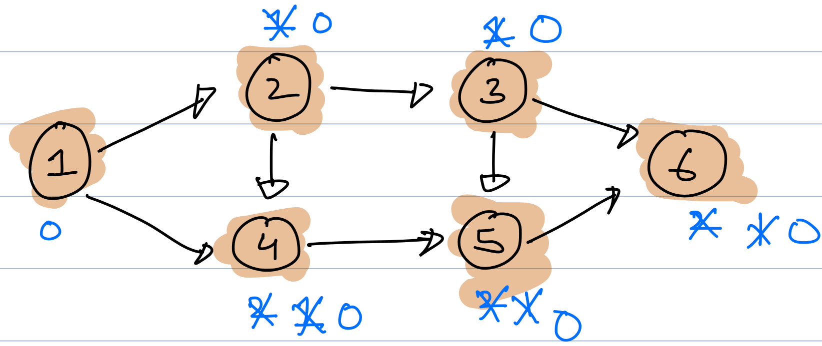


1 2



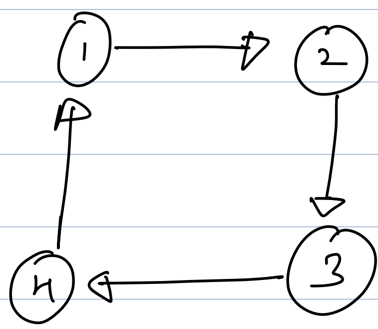
1, 2, 4, 3, 5, 6

1, 2, 3, 4, 5, 6

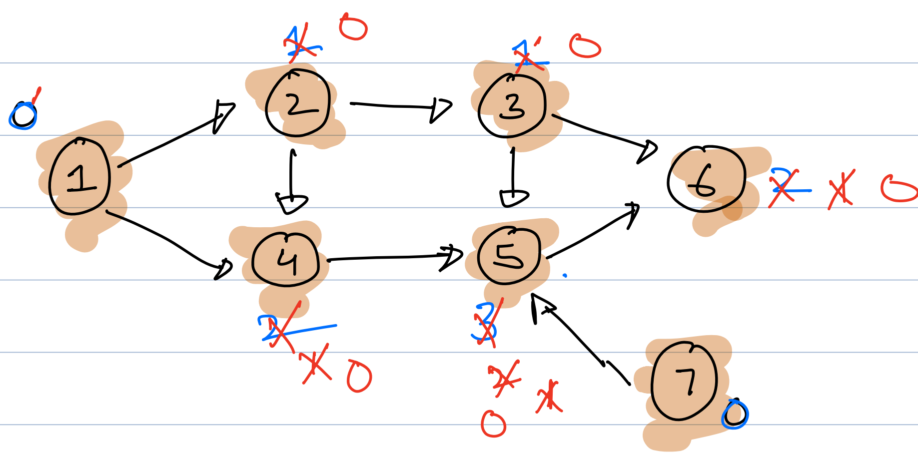


1, 2, 4, 3, 5, 6

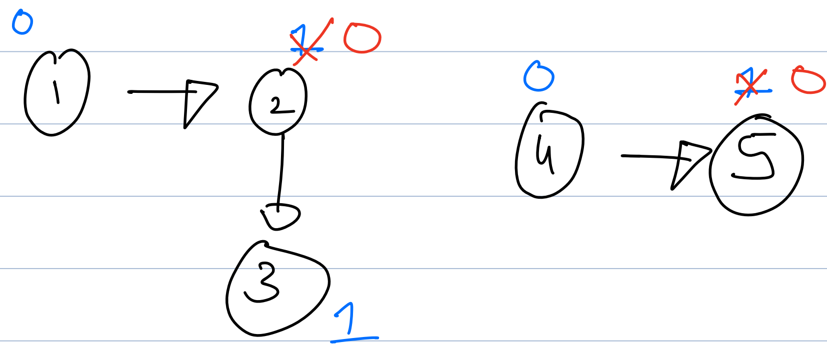
#



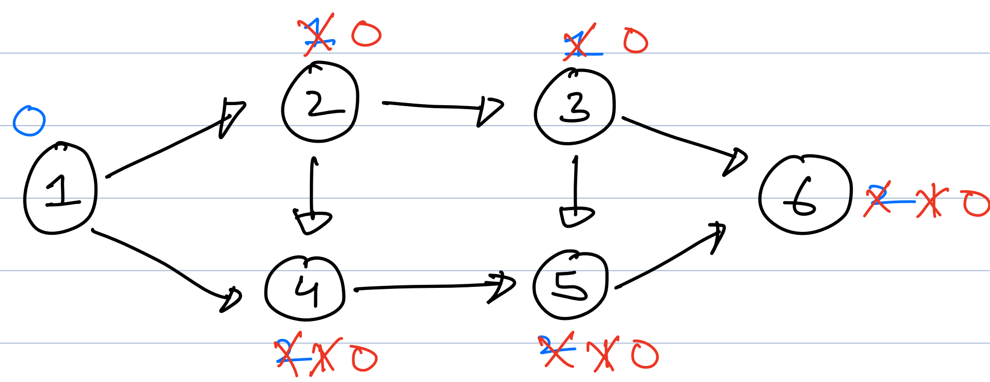
⇒ cyclic
no possible

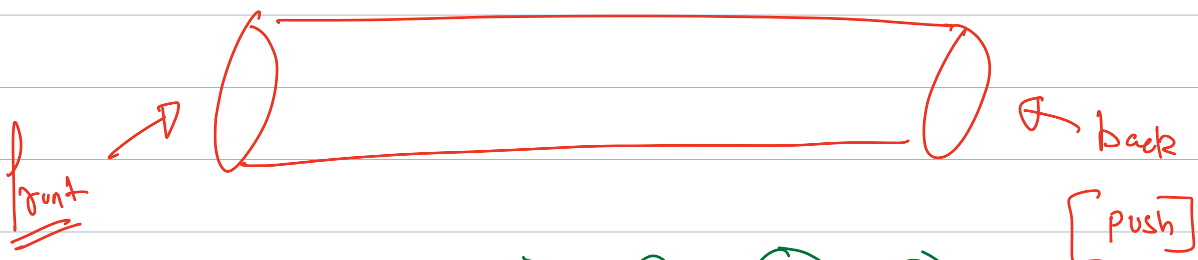


1, 2, 3, 7, 4, 5, 6



1, 4, 5, 2, 3





1 2 3 4 5 6

TODO: Solve using DFS

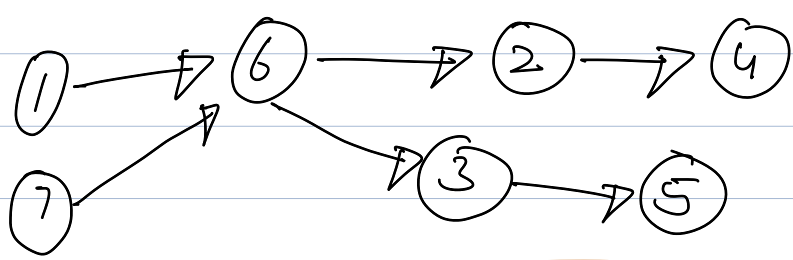
Pseudo Code:

1) Calculate indegree of each node.
2) Find nodes with indegree = 0 & push to Queue.
3) While (!Q.empty()) {

    → Pop the element, cnt++;
    → Reduce the indegree of it's neighbours.
    → if indegree of any neighbour becomes 0, push to Queue.

}

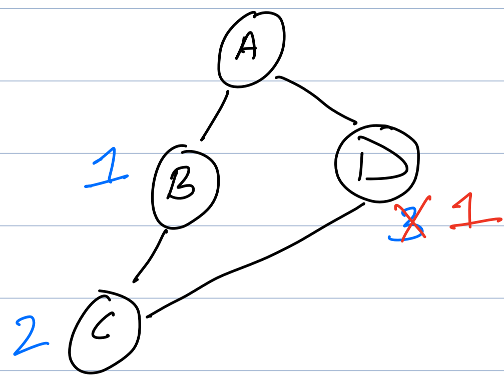
1, 2, 3, 4, 5, 6

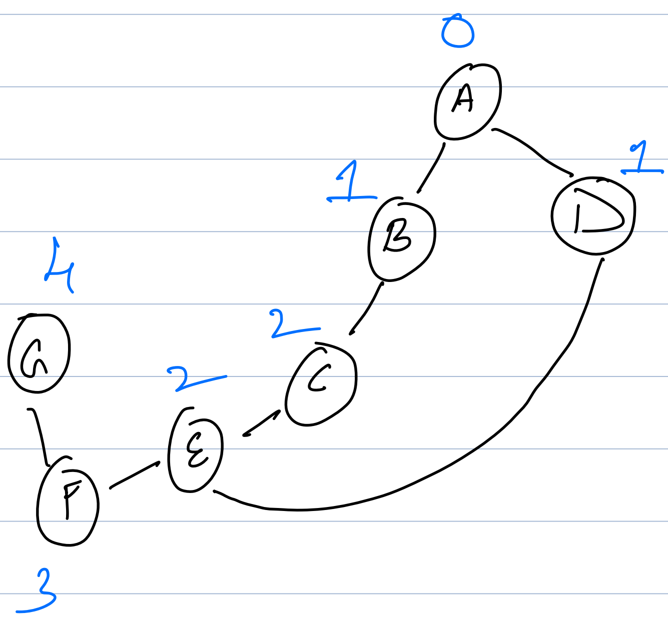


1, 6, 2, 3, 4, 5

Shortest Distance [UNWEIGHTED GRAPHS]

Source = A





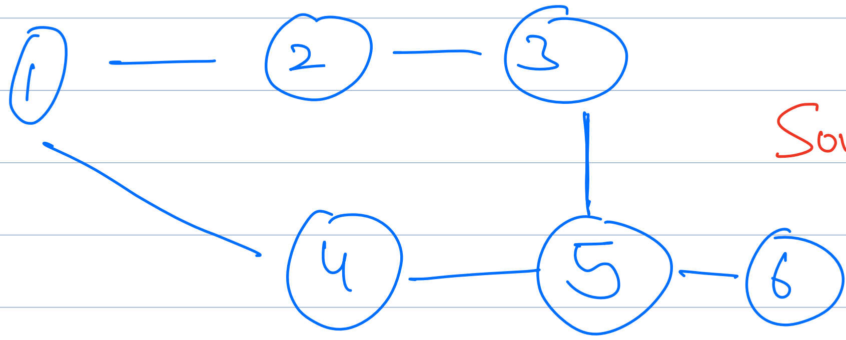
Shortest from source to destination = Level of destination w.r.t source.

Pseudo Code

```
void bfs ( int source. ) {.

    int visited [n] = {0};
    Queue <int > q;
    q. push (source)
    visited [source] = 1;   distance [source] =0;

    while( ! Q.empty() ) {.

        int n => Q.front();
        Q.pop();

        for (int i=0 ; i< adj [n].size(); i++ ) {.
            int neigh => adj [n] [i];
            if ( !visited [neigh]) {
                q.push (neigh);
                visited [neigh] = 1;
                distance [neigh] = distance [n] +1
            }
        }
}
```

TC: O (V+E)

SC: O (V+E)



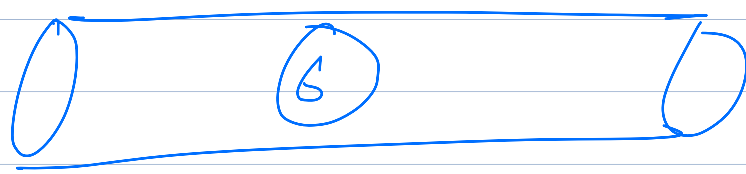
Source = 1

Visited =

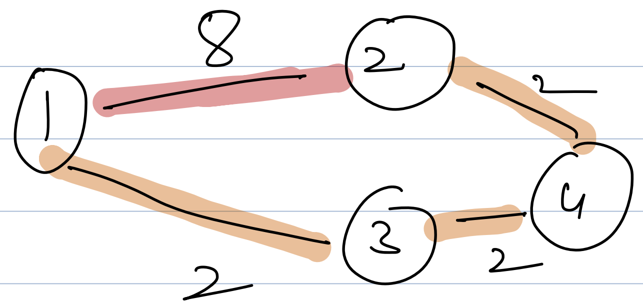
1	2	3	4	5	6
1	1	1	1	1	1

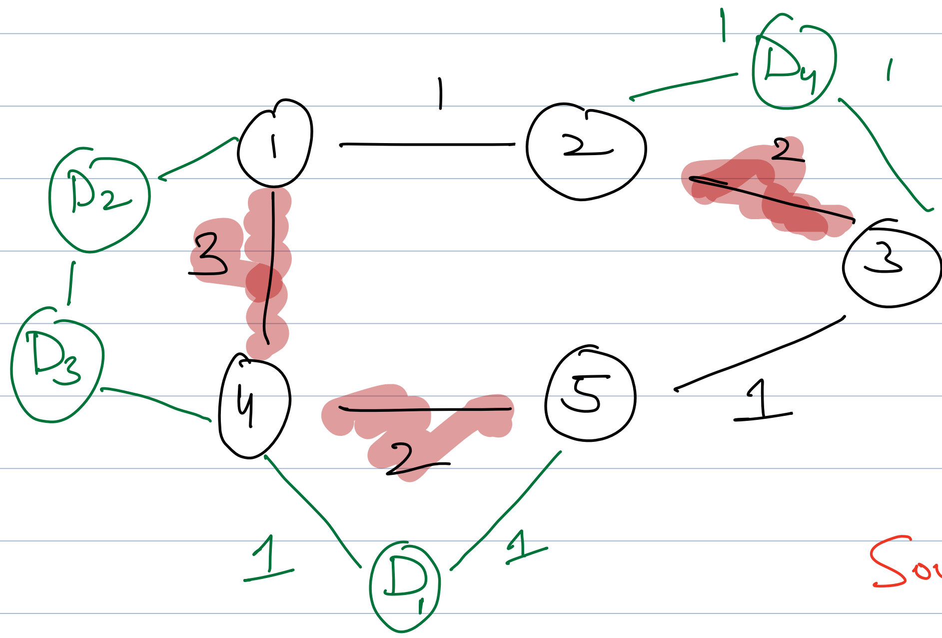
distance =

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



Shortest Distance [Weighted Graph].





Weight ≤ 3

Source = 1

Destination = 3

$TC: O(V+E)$

$O(V'+E')$

$V' \Rightarrow V+2E$

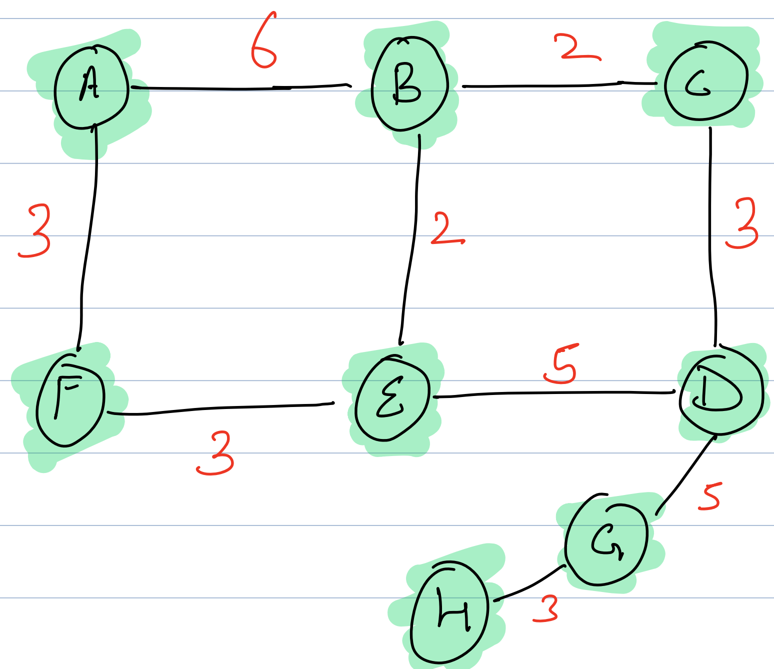
$E' \Rightarrow 3E$.

$TC: O(V+6E)$

$\simeq TC: O(V+E)$

Dijkstra → doesn't work for −ve edges

Source = E



Min heap

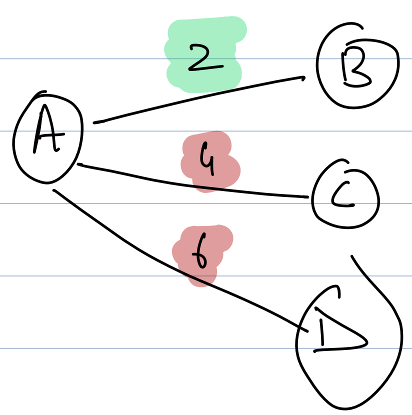
(0, E)

Visited :

A	B	C	D	E	F	G	H
1	1	1	1	1	1	1	1

Distance =

A	B	C	D	E	F	G	H
6	2	4	5	0	3	10	13



A → (B,6) → (F,3)

Pseudo Code!

$\boxed{0 — n-1}$

```
dis [n] = {-1}.
Minheap <Pair <int, int> > mh;
dis [source] = 0
mh.push ( make_pair ( 0, source ) );

while ( ! mh.empty() ) {

    d, node => mh.front();
    mh.pop();

    if ( dis [node] != -1 )
        continue;

    dis [node] = d;

    for ( int i=0 ; i < adj [node].size(); i++ ) {
        neigh, wei => adj [node] [i]

        if ( dis [neigh] == -1 ) {
            mh.push (make_pair (d + wei,
                                neigh ));
        }
    }
}
```

1

3

TC : No of Entries in min heap $\simeq E$

$$\underbrace{E \log E}_{\text{insertion}} + \underbrace{E \log E}_{\text{removal}} + V$$

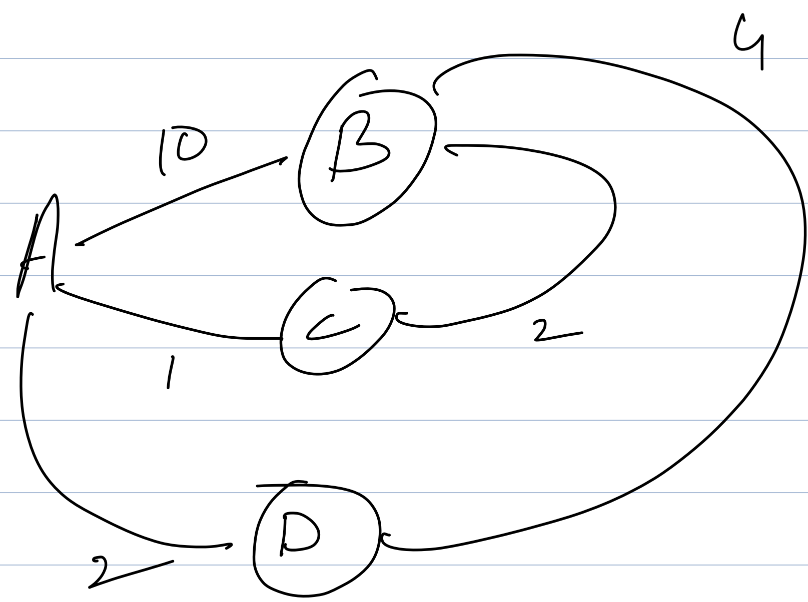
$$TC \Rightarrow O(V + E \log E)$$

$$SC \Rightarrow O(V+E)$$

10, B

(6, B)

(3, B)



1 2 3 4 5

V