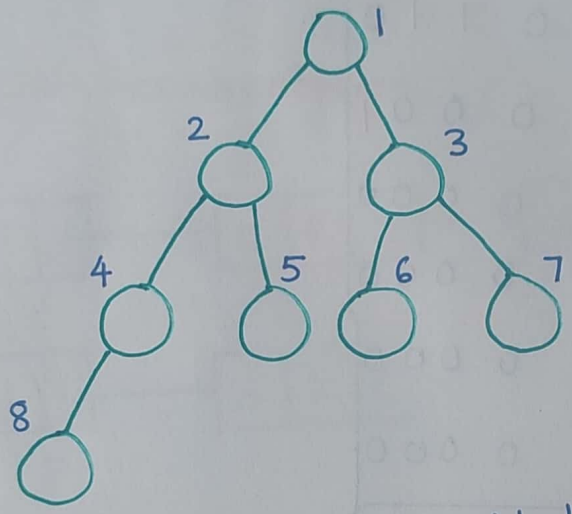
CS5300 Advanced Algorithms — Kishor Kumar Andekar

HW # 6 — 700744713 (CRN : 12008)

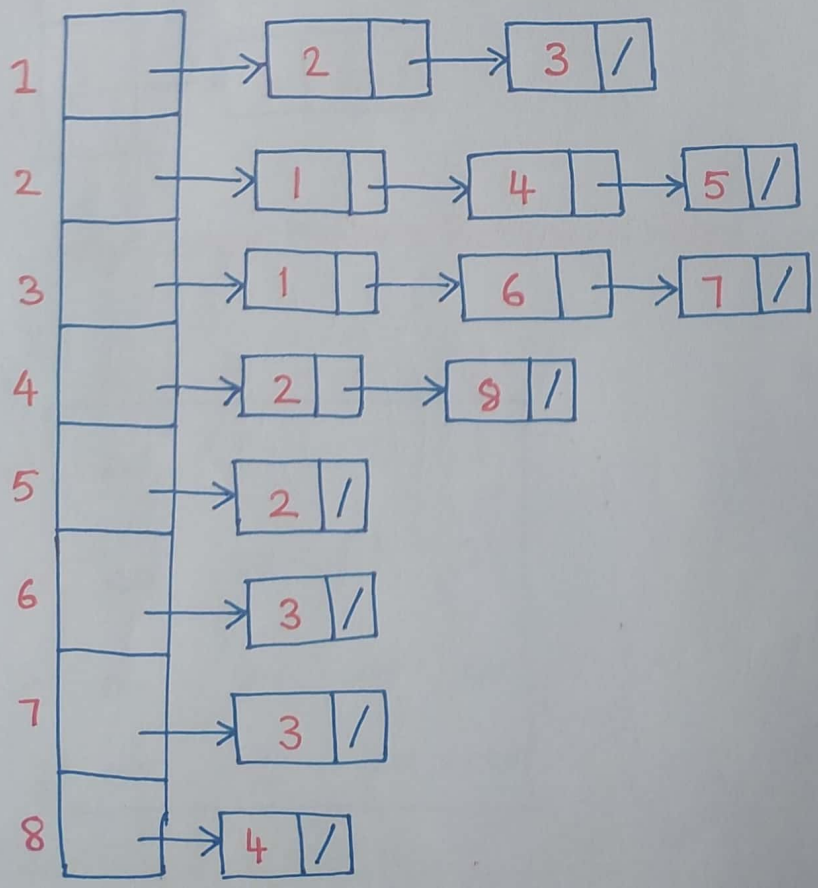
1. Give an adjacency-list representation for a complete binary tree on 8 vertices. Give an equivalent adjacency-Matrix representation. Assume that vertices are numbered from 1 to 8 as in a binary heap.



$|V| = 8$

We have 8 vertices, so we create 8 Lists.

Adjacency-List Representation

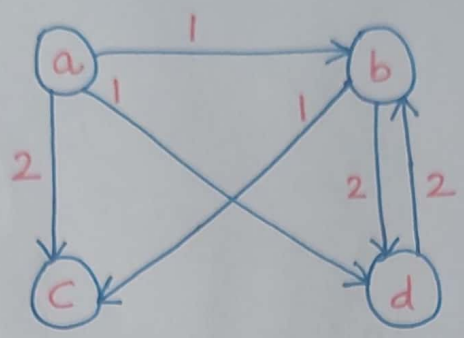


Adjacency-Matrix Representation :

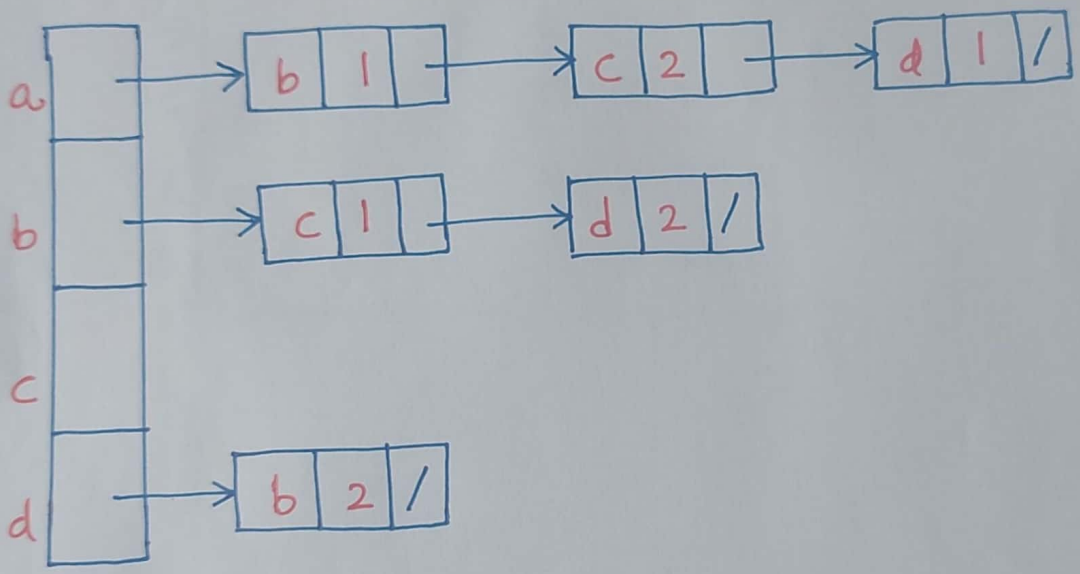
The size of the Matrix is 8 by 8.

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

2. Give an Adjacency List and Matrix Representation of the following Graph.



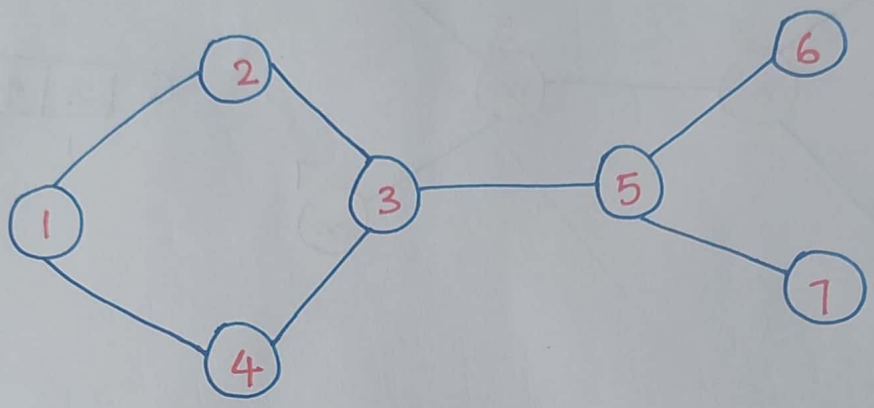
Adjacency List Representation:



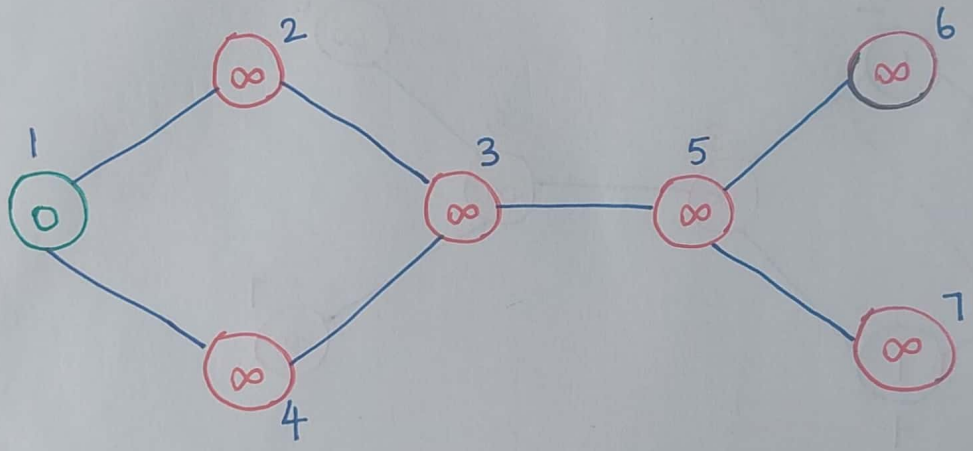
Adjacency Matrix Representation:

	a	b	c	d
a	∞	1	2	1
b	∞	∞	1	2
c	∞	∞	∞	∞
d	∞	2	∞	∞

3. Show the d and π values that result from running breadth-first search on the graph below using vertex 1 as the source. Show the Final graph produced by breadth-first search algorithm.



White = Red
Gray = Green

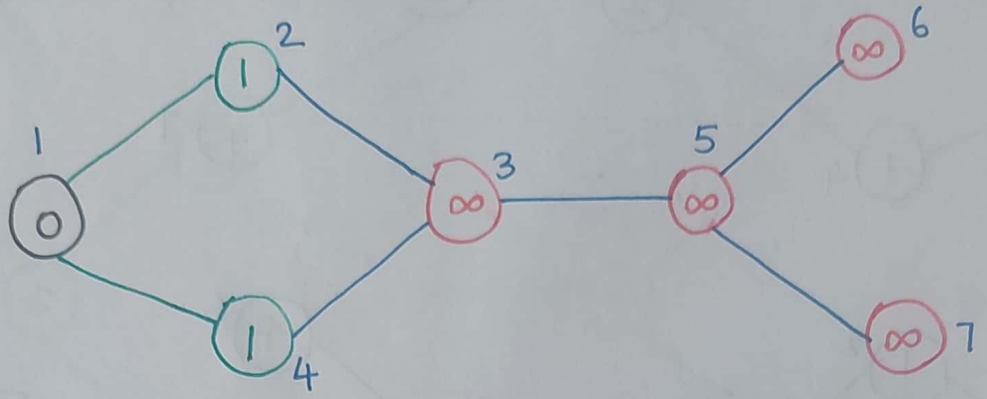


Q [1]
0

Vertex	Color	d	π
1	~~W~~ G B	0	Nil
2	~~W~~ ~~G~~ B	~~∞~~ 1	~~Nil~~ 1
3	~~W~~ ~~G~~ B	~~∞~~ 2	~~Nil~~ 2
4	~~W~~ ~~G~~ B	~~∞~~ 1	~~Nil~~ 1
5	~~W~~ ~~G~~ B	~~∞~~ 3	~~Nil~~ 3
6	~~W~~ ~~G~~ B	~~∞~~ 4	~~Nil~~ 5
7	~~W~~ ~~G~~ B	~~∞~~ 4	~~Nil~~ 5

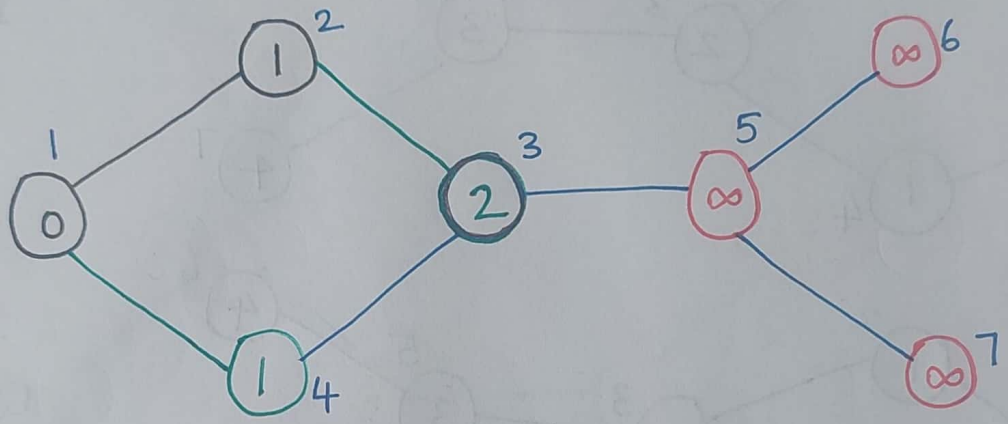
Q

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



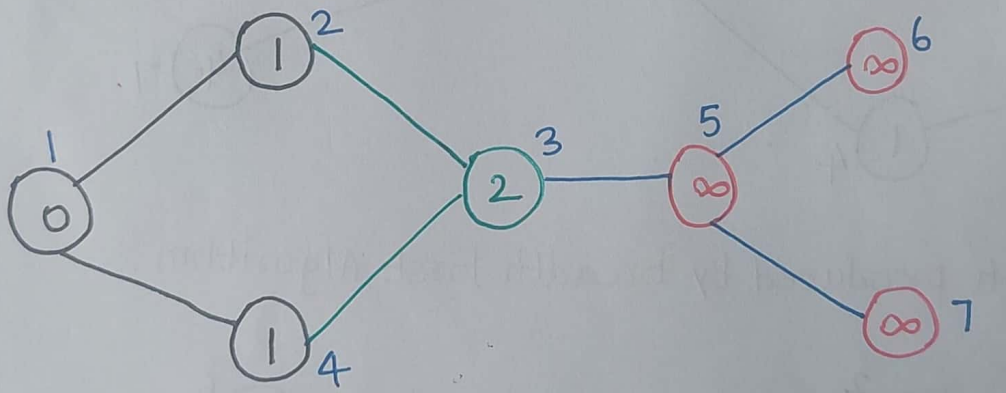
Q

2	4
1	1



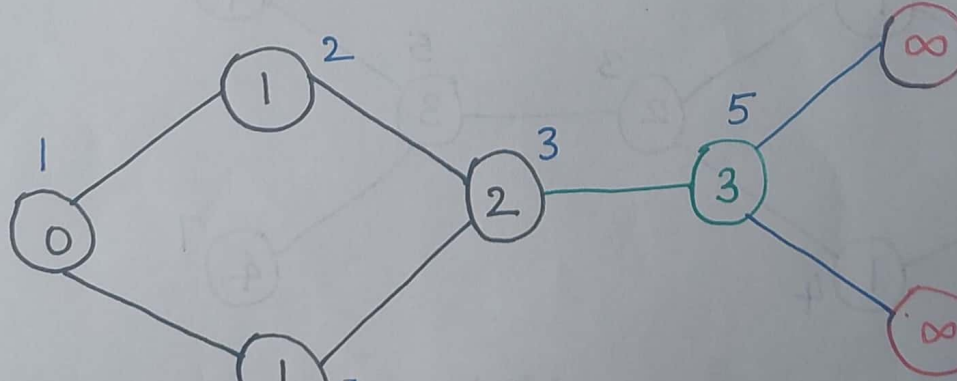
Q

4	3
1	2



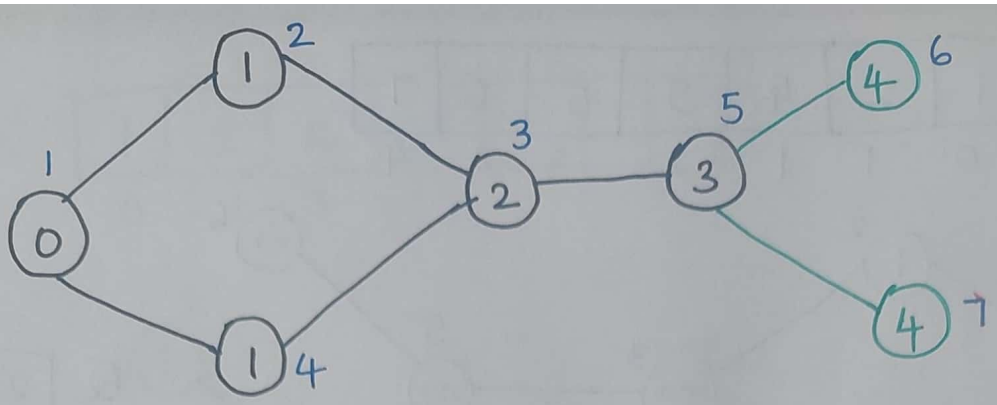
Q

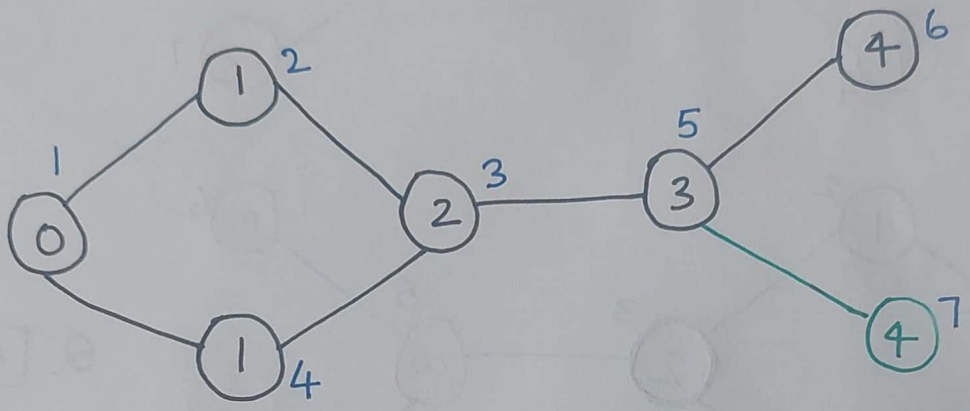
3
2

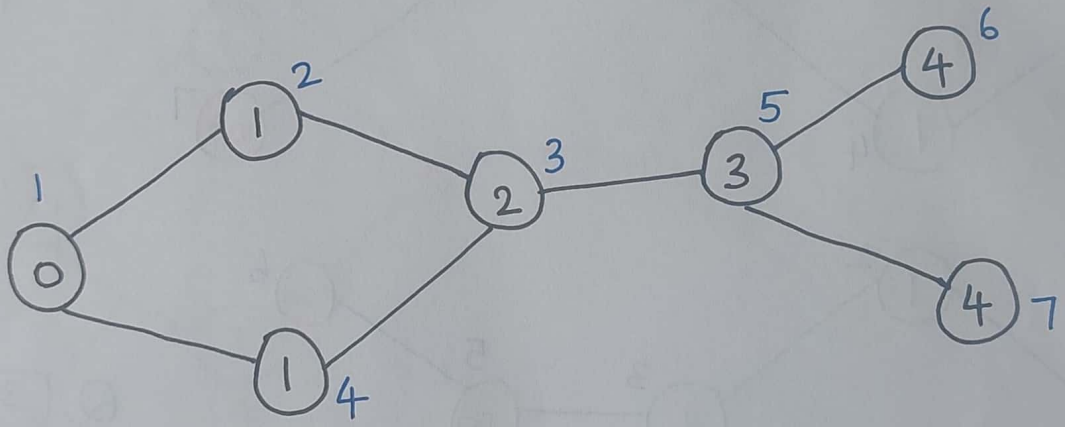


Q

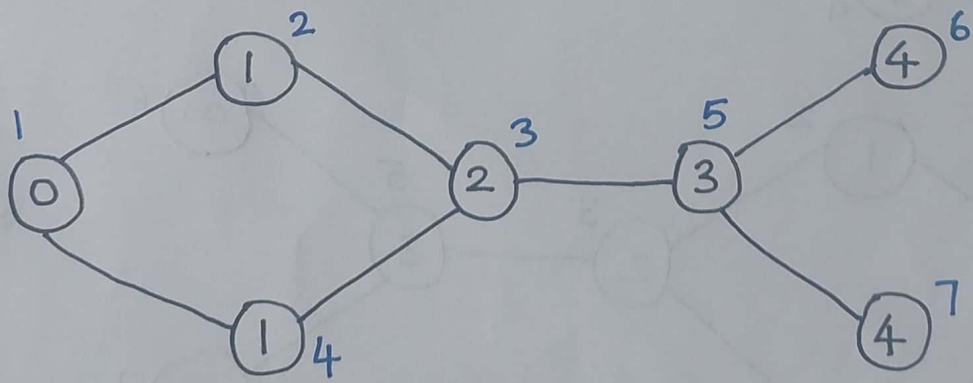
5
3



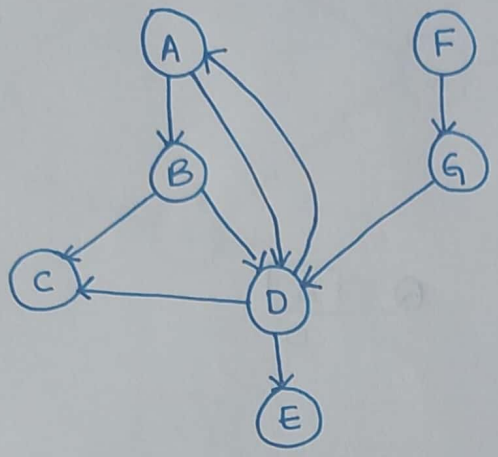




Final Graph produced by breadth First Algorithm :



4. Consider the following Graph.



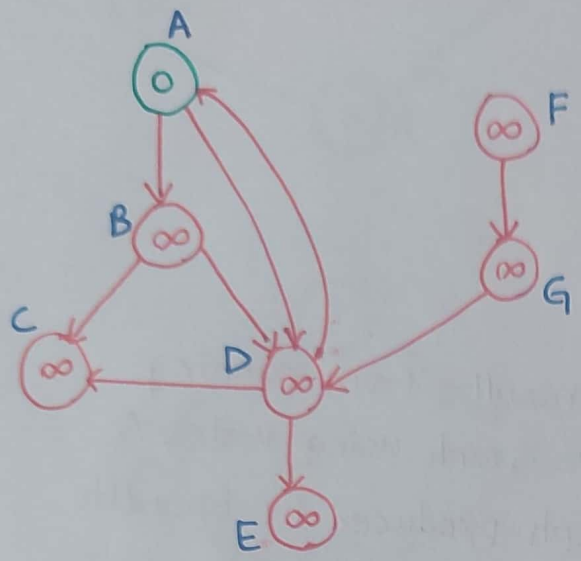
Show the d and π values that results from running breadth-first search on the above Graph using vertex A as the source. show the Final graph produced by breadth First search algorithm. What is the time complexity of breadth-first search algorithm?

Vertex A as the source.

Vertex	Color	d	π
A	~~G~~ B	0	Nil
B	~~W~~ ~~G~~ B	~~∞~~ 1	~~Nil~~ A
C	~~W~~ ~~G~~ B	~~∞~~ 2	~~Nil~~ B
D	~~W~~ ~~G~~ B	~~∞~~ 1	~~Nil~~ A
E	~~W~~ ~~G~~ B	~~∞~~ 2	~~Nil~~ D
F	W	∞	Nil
G	W	∞	Nil

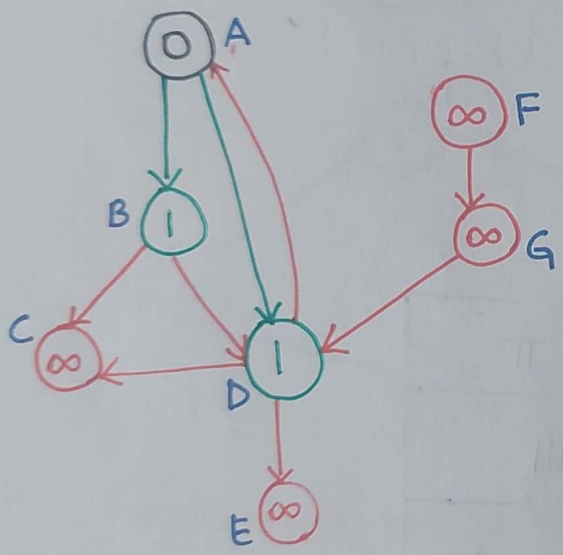
Q

A	B	D	C	E
0	1	1	2	2



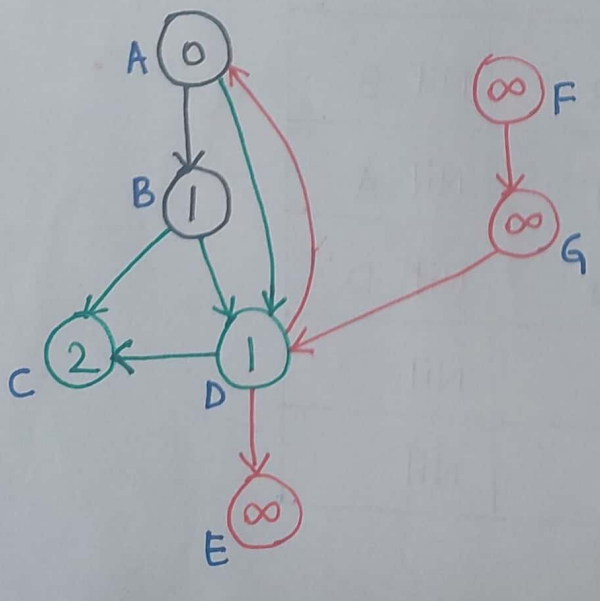
Q

A
0



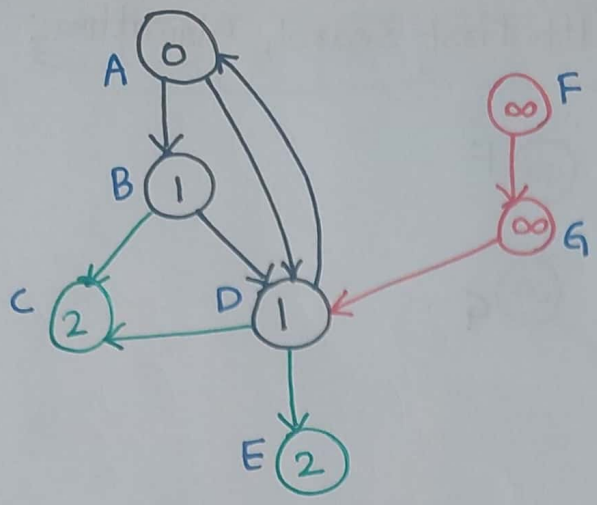
Q

B	D
1	1

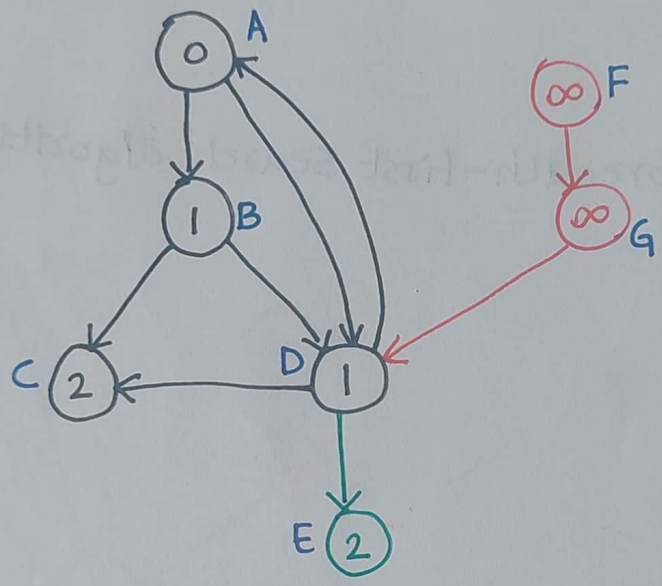


Q

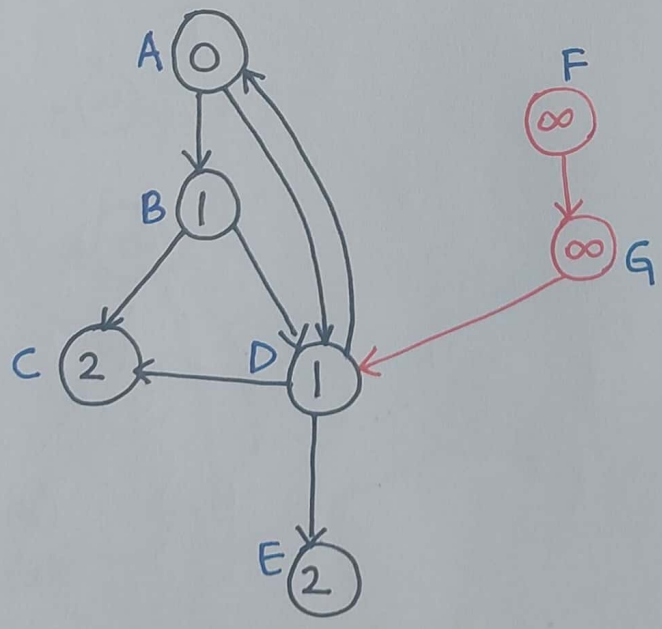
D	C
1	2



Q | C | E
2 2



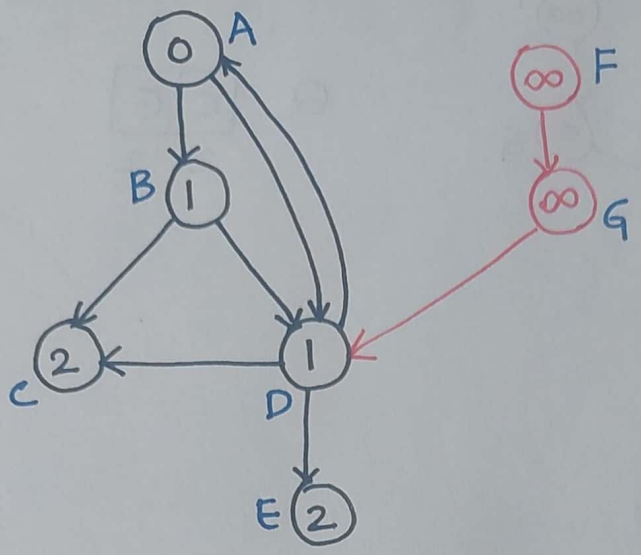
Q | E
2



Q ∅

In BFS Algorithm, we are unable to discover the vertex F and G. So, we leave them alone.

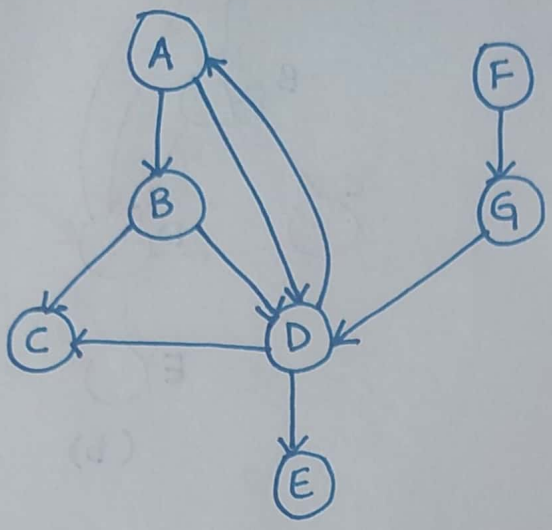
Final Graph produced by Breadth First Search Algorithm :



Time Complexity of breadth-first search algorithm is

$\Theta(|V|+|E|)$

5. Consider the following Graph



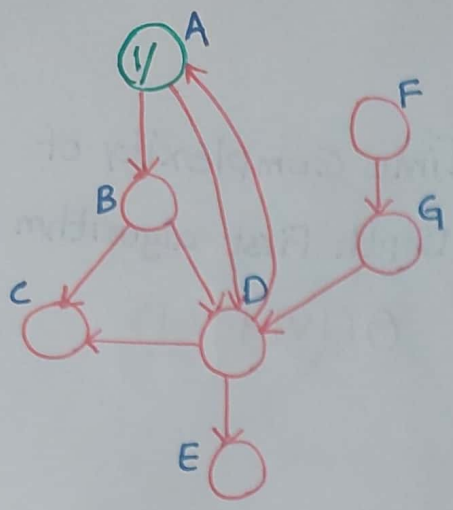
Time Complexity of Depth First algorithm is

$\Theta(|V|+|E|)$

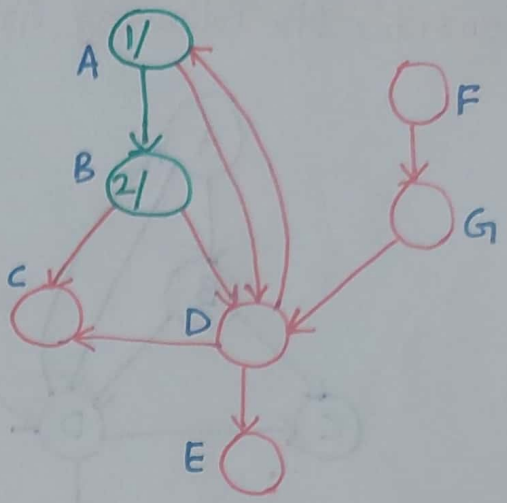
Show how depth First search works on the above Graph. Assume that the for loop of lines 5-7 of the DFS procedure consider the vertices in alphabetical order, and assume that each adjacency list is ordered alphabetically. Show the discovery and Finishing times for each vertex, and show the classification of each edge. What is the time complexity of depth first algorithm?

Time
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14

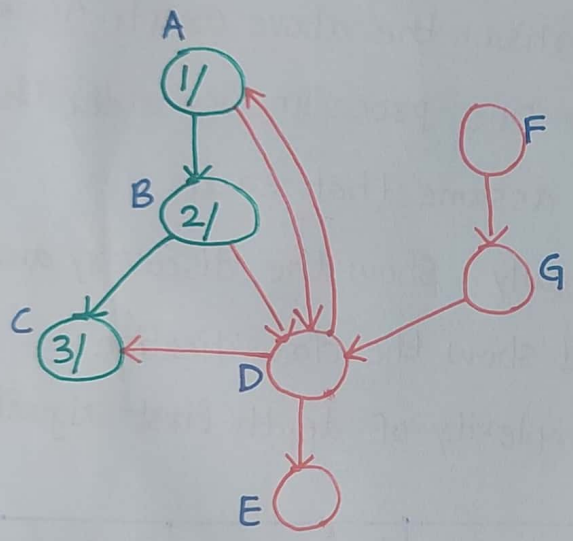
Vertex	Color	Parent(π)	Discovery Time (d)	Finishing Time (f)
A	~~W~~ ~~G~~ B	Nil	1	10
B	~~W~~ ~~G~~ B	~~Nil~~ A	2	9
C	~~W~~ ~~G~~ B	~~Nil~~ B	3	4
D	~~W~~ ~~G~~ B	~~Nil~~ B	5	8
E	~~W~~ ~~G~~ B	~~Nil~~ D	6	7
F	~~W~~ ~~G~~ B	Nil	11	14
G	~~W~~ ~~G~~ B	~~Nil~~ F	12	13


A
1/
B
C
D
E
F
G

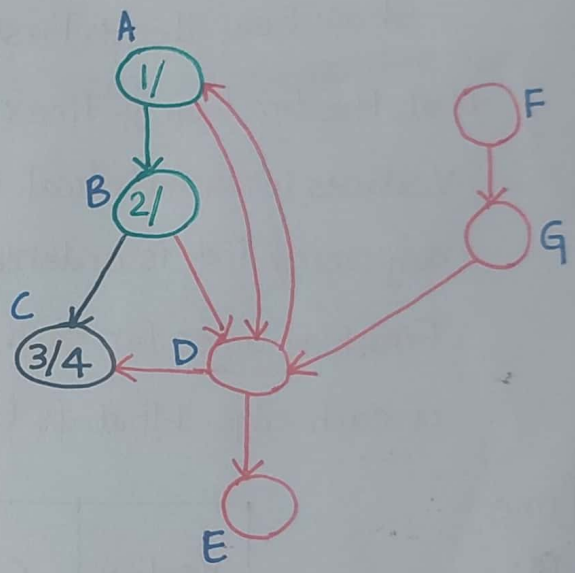
(a)


A
1/
B
2/
C
D
E
F
G

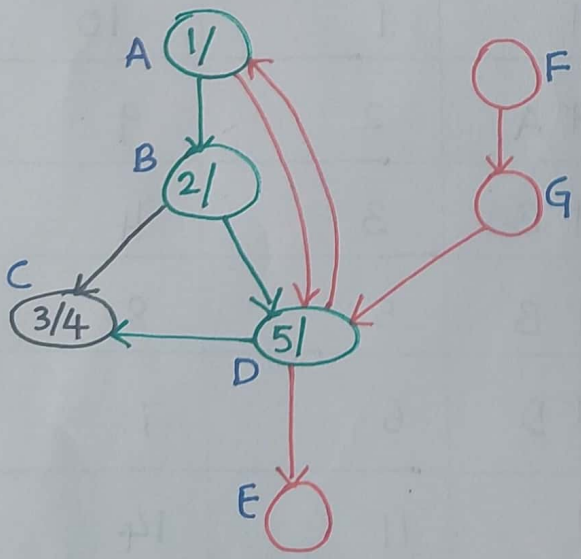
(b)


A
1/
B
2/
C
3/
D
E
F
G

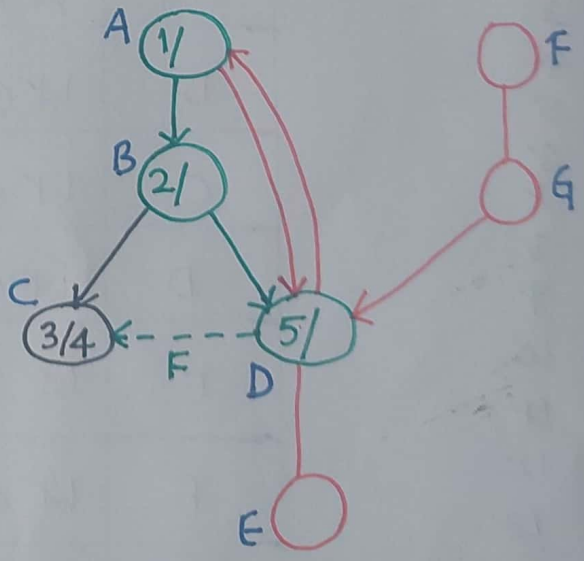
(c)


A
1/
B
2/
C
3/4
D
E
F
G

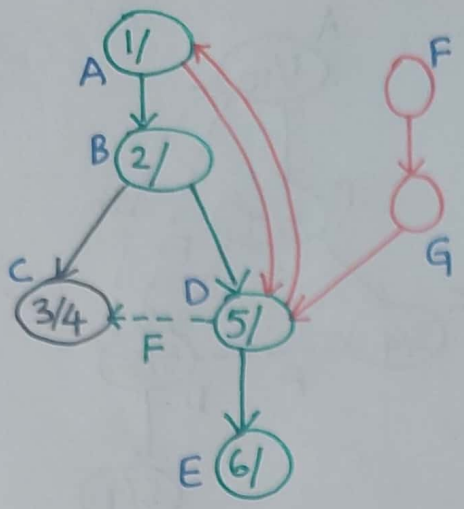
(d)


A
1/
B
2/
C
3/4
5/
D
E
F
G

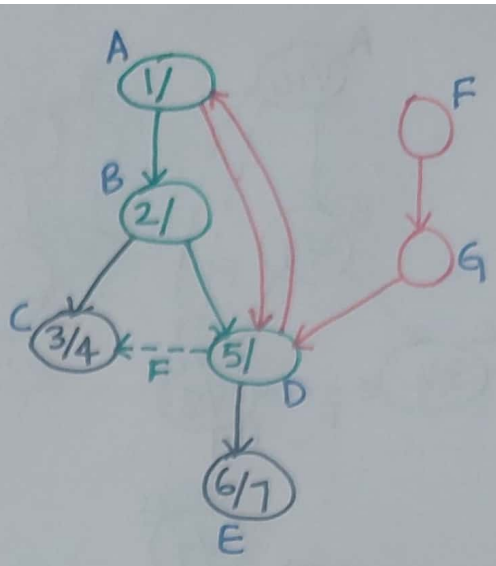
(e)


A
1/
B
2/
C
3/4
F
5/
D
E
F
G

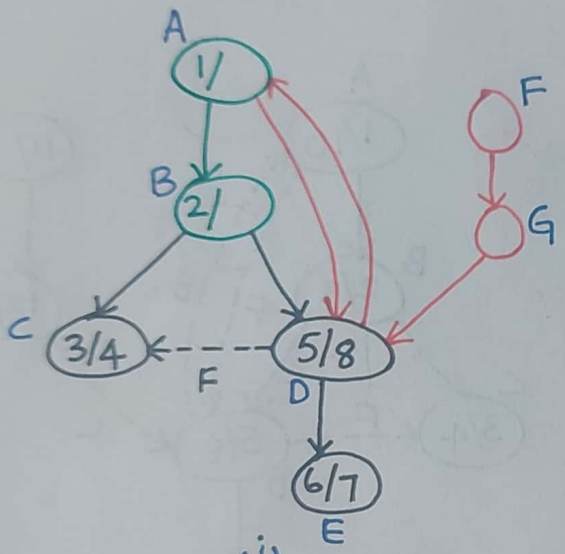
(f)



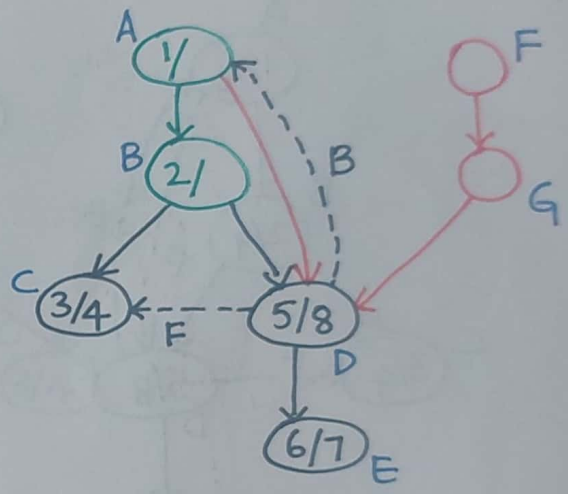
(g)



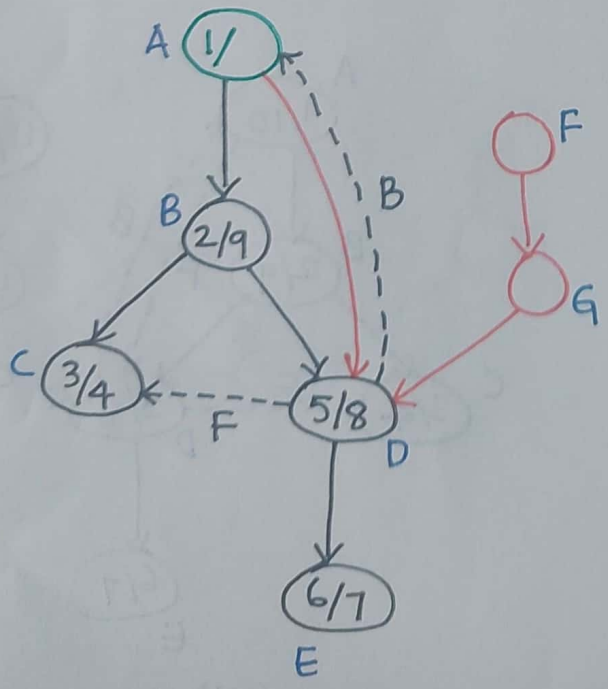
(h)



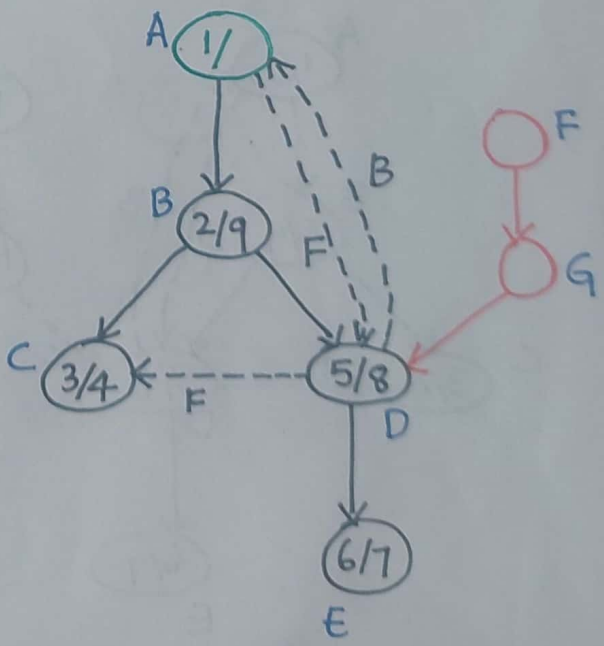
(i)



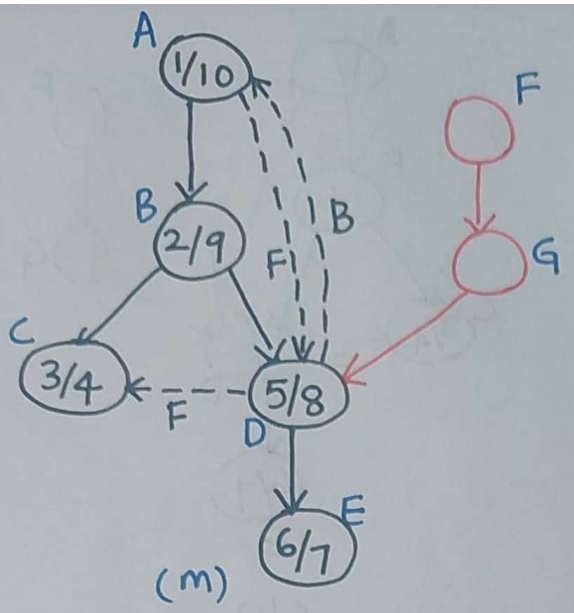
(j)



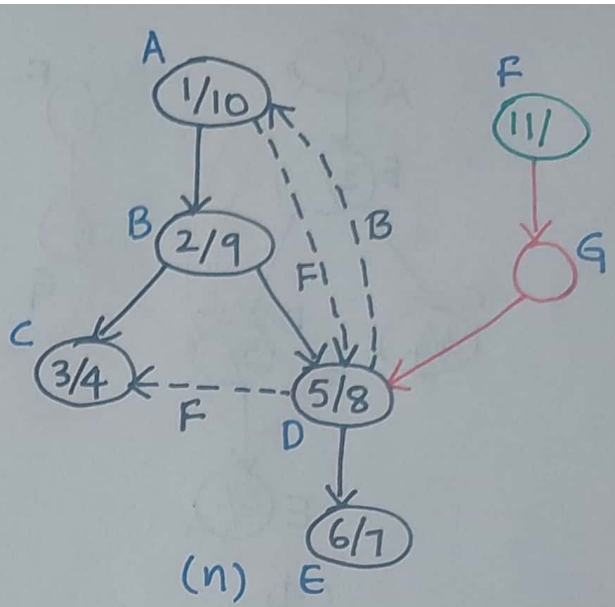
(K)



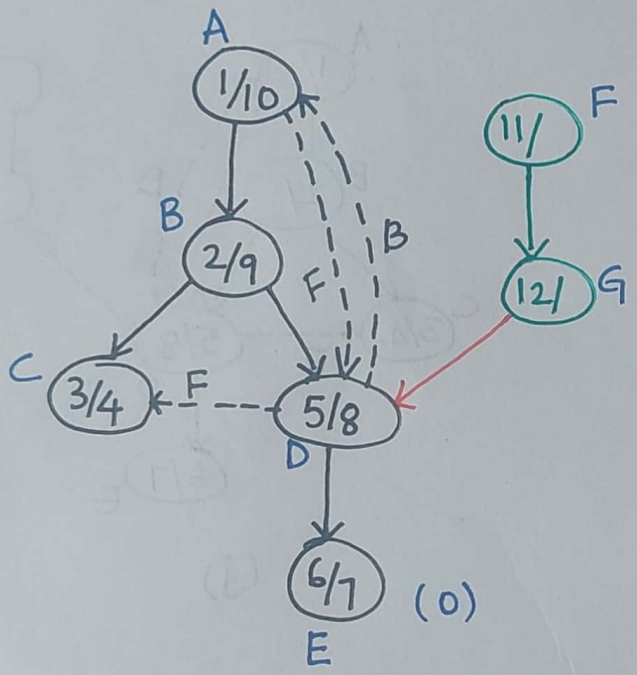
(L)


A
1/10
B
2/9
F
B
C
3/4
F
5/8
D
F
G
E
6/7

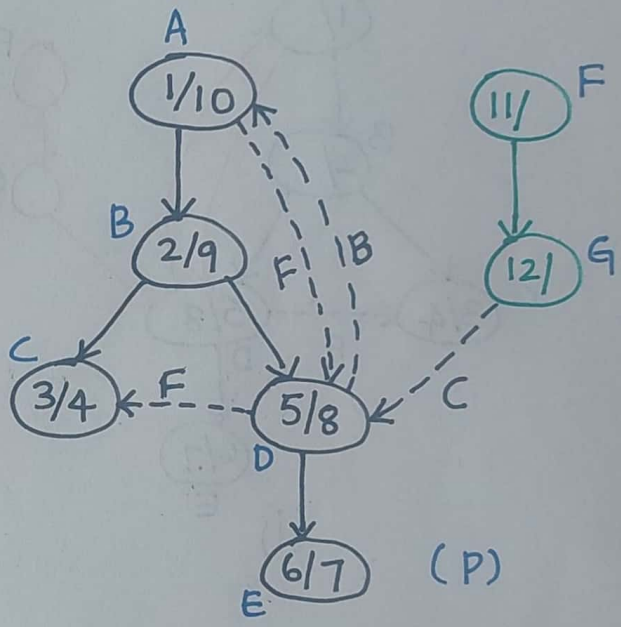
(m)


A
1/10
B
2/9
F
B
C
3/4
F
5/8
D
F
11/
G
6/7

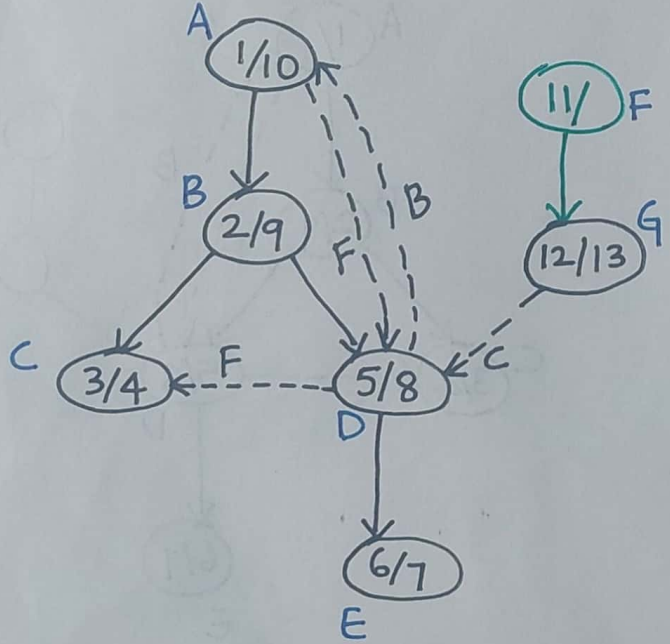
(n) E


A
1/10
B
2/9
F
B
C
3/4
F
5/8
D
11/
F
12/
G
6/7
E

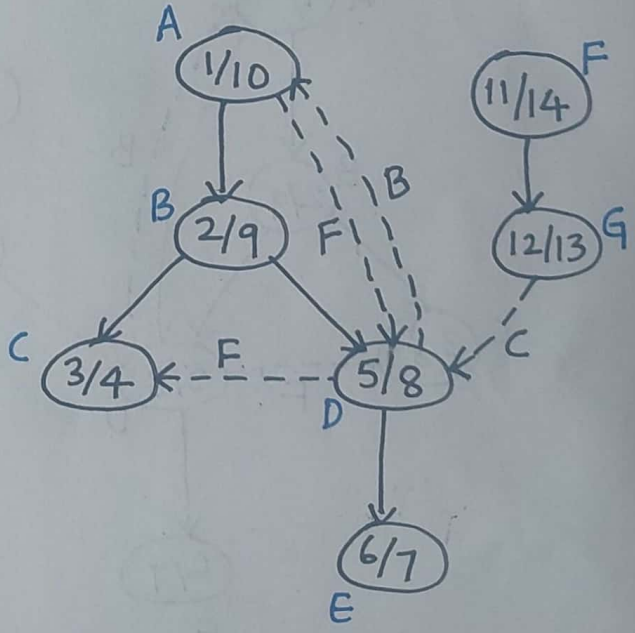
(o)


A
1/10
B
2/9
F
B
C
3/4
F
5/8
D
C
11/
F
12/
G
6/7
E

(P)

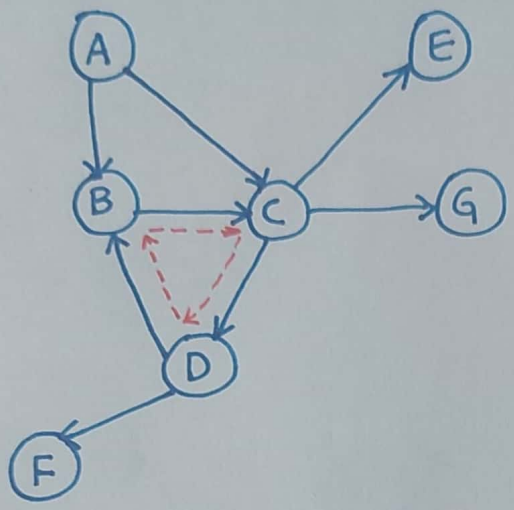

A
1/10
B
2/9
F
B
C
3/4
F
5/8
D
C
11/
F
12/13
G
6/7
E

(q)


A
1/10
B
2/9
F
B
C
3/4
F
5/8
D
C
11/14
F
12/13
G
6/7
E

(r)

6. List one Topological ordering of the following graph. If no ordering exists, briefly explain Why.



In our given directed Graph, We see a cycle 'BCD'

Cycle Graph means if you started from a vertex and you are able to come back to that particular vertex.

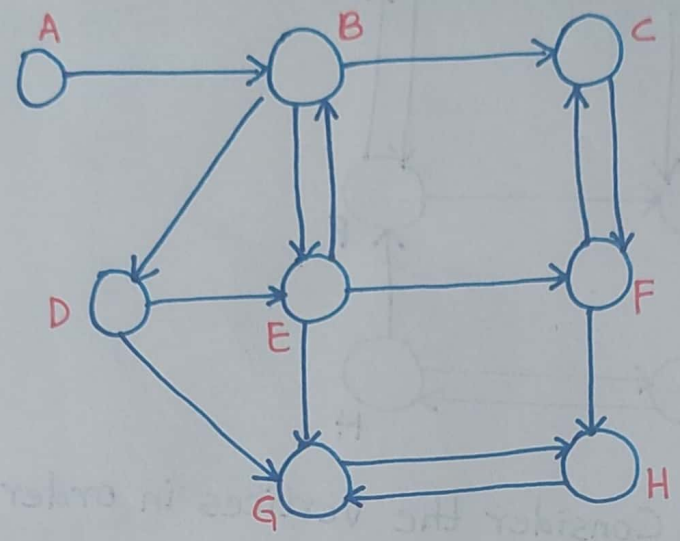
Directed Acyclic Graph (DAG) means graph has no cycles.

Topological ordering is applied only to DAG.

Above Given Graph is not a DAG, So Topological ordering is not possible. Also, our Graph consists cycle then no Topological sort.

7. For the Following Graph. Find the strongly Connected Components:

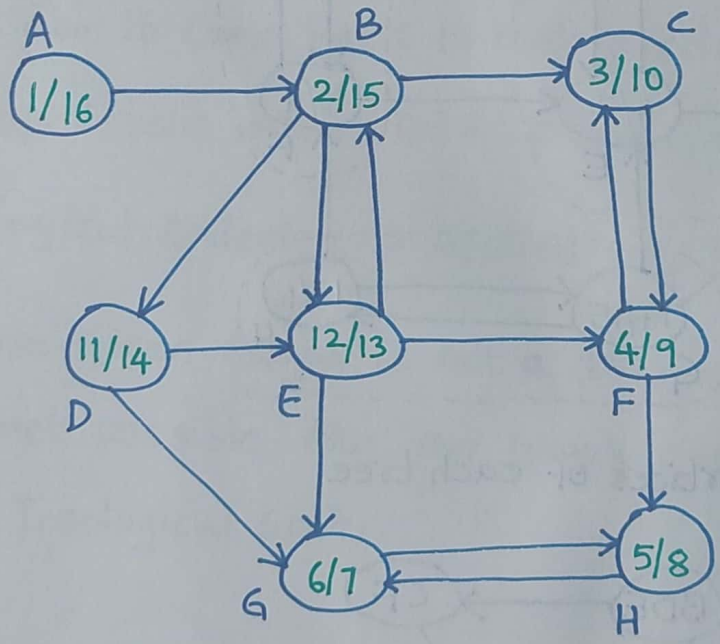
G:



Step 1: Call DFS(G)

Consider the vertices in alphabetical order and also also adjacency Lists in alphabetical order

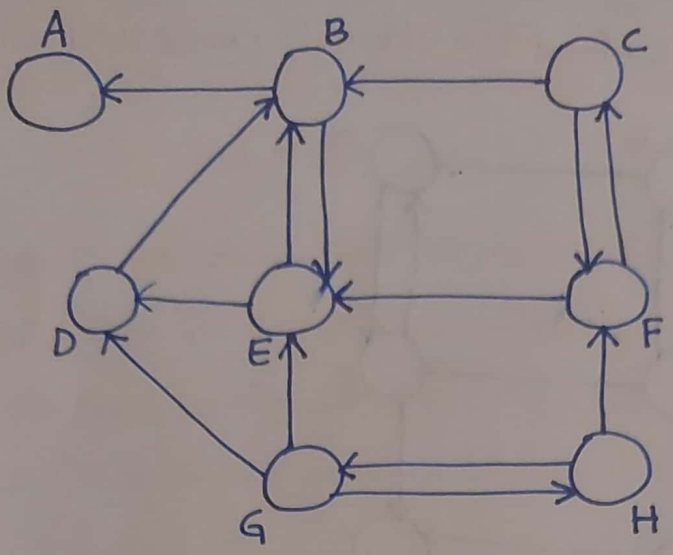
G:



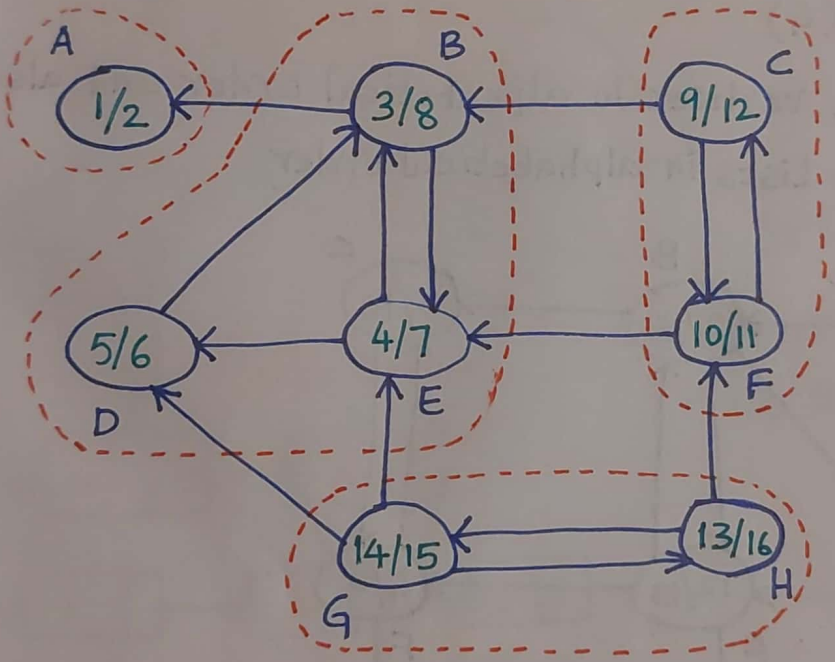
Step 2: Compute G^T

G^T means reverse the direction.

G^T:



Step 3: Call DFS(G^T), Consider the vertices in order of decreasing order of finishing time i.e., 16



Step 4: Output the vertices of each tree