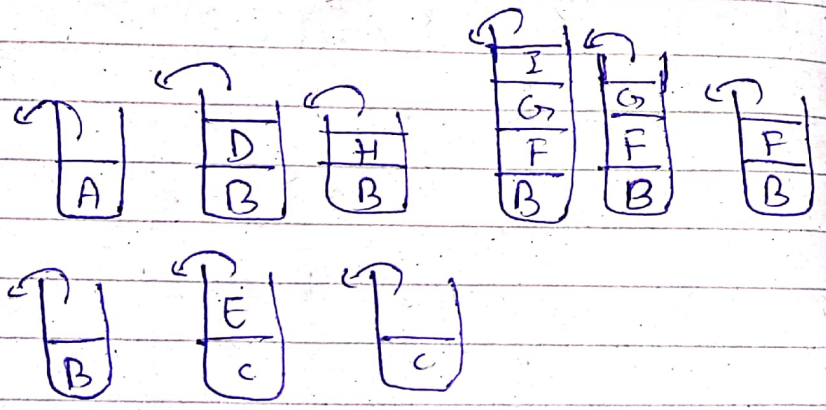
ALGO ASSIGNMENT 04

Bilal Ahmed Khan

20K0183 ; Sec: B

QUESTION 01

a)

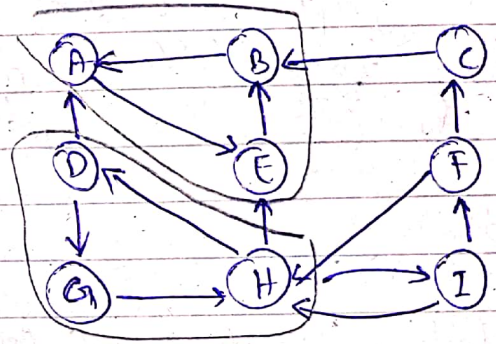


Visited: | A | D | H | I | G | F | B | E | C |

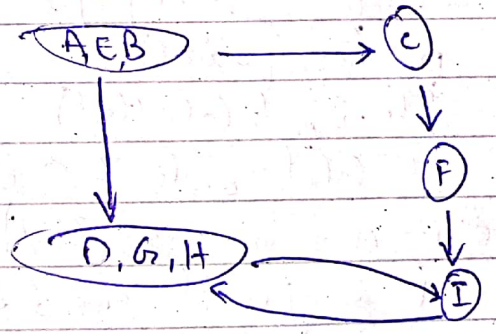
b) NO set given.

c) Strongly Connected components

↳ A-E-B
↳ D-G-H
↳ I
↳ F
↳ C



Component graph



QUESTION 02

a)

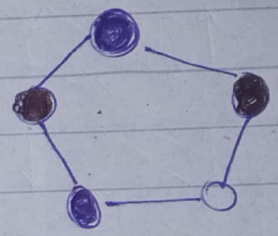
We can use two different marks (X and Y) while traversing the graph via BFS.
If we are able to mark the whole graph such that all adjacent nodes have different marks then its bipartite, otherwise not.

Algorithm:-

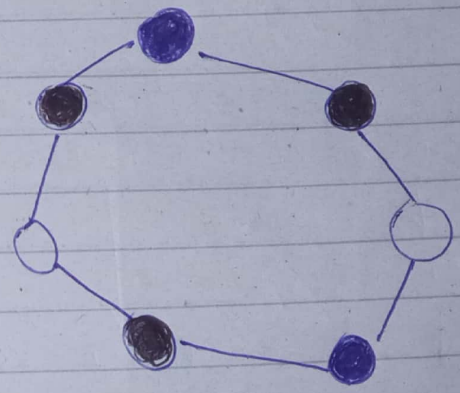
1) Select a random node (mark it X)
→ push node in queue
while (queue != empty())
{ n = top.queue()
if (n is not labelled)
(label node Y & continue)
if (n is same labelled)
{ give it same label;
return false i.e. not bipartite
}
}
return true i.e. bipartite

The running time complexity of this algorithm will be linear

b) We will need a minimum of 3 colors, we can show this by drawing a cycle of 5 or cycle of 7 nodes.



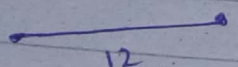
Black
Blue
& white
color used



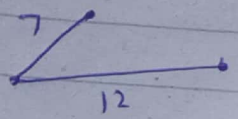
QUESTION 03

a)

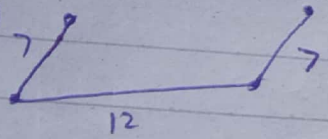
i)



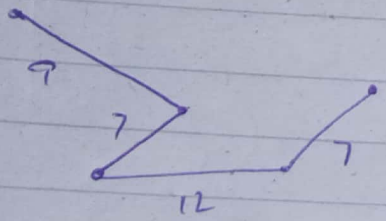
ii)



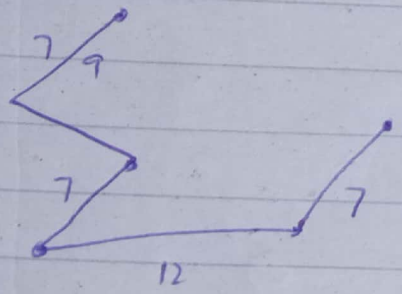
iii)



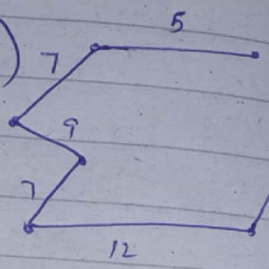
iv)



v)



iv)



MST Cost = 47

b)

It will depend on the graph

for eg: If each edge of a graph has different weights then both Prims and kruskal will give same MST.

But if two or more edges have the same weights in graph then their MST will be different

QUESTION 04

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

D_0

	0	1	2	3	4
0	0	2	∞	∞	4
1	∞	0	3	∞	∞
2	∞	∞	0	5	1
3	8	10	∞	0	12
4	∞	∞	∞	7	0

D_1

	0	1	2	3	4
0	0	2	5	∞	4
1	∞	0	3	∞	∞
2	∞	∞	0	5	1
3	8	10	13	0	12
4	∞	∞	∞	7	0

D_2

	0	1	2	3	4
0	0	2	5	10	4
1	∞	0	3	8	4
2	∞	∞	0	5	1
3	8	10	13	0	12
4	∞	∞	∞	7	0

D_3

	0	1	2	3	4
0	0	2	5	10	4
1	16	0	3	8	4
2	13	15	0	5	1
3	8	10	13	0	12
4	15	17	20	7	0

D_4

	0	1	2	3	4
0	0	2	5	10	4
1	16	0	3	8	4
2	13	15	0	5	1
3	8	10	13	0	12
4	15	17	20	7	0

Floyd Warshall has $O(|V|)^3$ time complexity & $O(|V|^2)$ space complexity

QUESTION NO. 05

→ It is → All pairs shortest Path Algorithm which can be applied on directed graphs with +ve weights

→ We add a new vertex to the existing graph with a weight zero that is connected to all vertices of the graph.

→ The added vertex has no incoming edges and is used as the source vertex.

→ In this way we can use Bellman Ford Algo to complet the shortest path.

vertex weight

P_v = length of shortest s→v path

→ For each edge $e = (u,v)$,

$c'_e = c_e + P_u - P_v$

→ Once the re-weights are determined, all the weights become non-negative

→ It preserves all shortest paths

→ If there is a -ve cost cycle, the cycle has to be in the original graph.

→ Run Dijkstra algorithm after reweighting

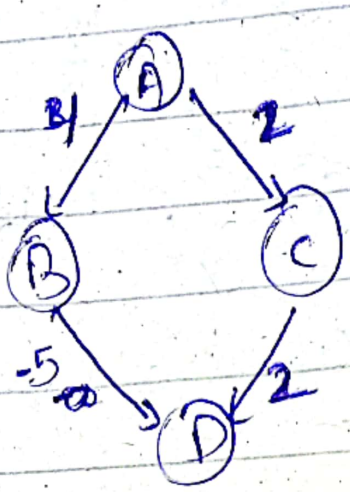
→ we get shortest path wrt new weight but subtract $(P_u + P_v)$ to get shortest path w.r.t. original graphs.

$$d(u,v) = d'(u,v) - (P_u + P_v)$$

→ running time = $O(mn \log(n))$

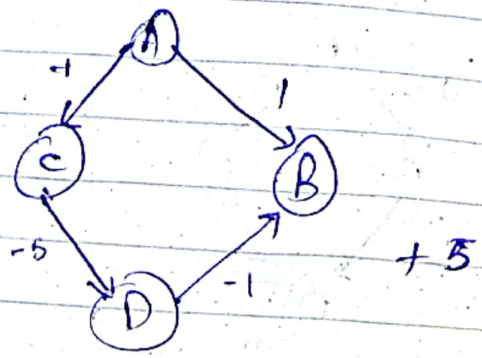
QUESTION 06

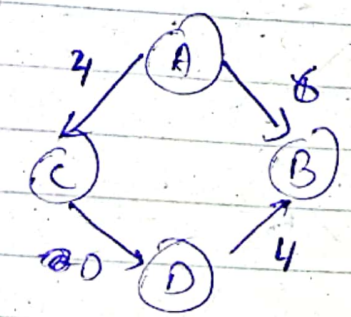
06:a)



If we apply Dijikstra algo to this graph wrong cost for A to D path (2+2=4) instead it could be -1 (4-5)

b) Its not a valid method and it will not work
Because it will affect the existing shortest paths and return the wrong answer.





The shortest path should be A→C→D→B but the algo will return A→B if we want to go from A to B. This is therefore a wrong solution

QUESTION 07

1) BFS

a) Adjacency Matrix

Time complexity $O(|V|^2)$

Space complexity: entire array of length $|V|$

b) Adjacency list:

Time complexity: $O(|V|+|E|)$

Space complexity: $|E|+|V|$

2) DFS

a) Adjacency Matrix Same as BFS

b) Adjcency List Same as BFS

3) KRUSKAL ALGORITHM:

a) Adjacency Matrix

Time complexity : $O(|V|^2 + E \log |E|)$

Space complexity:- Takes $O(|V|^2)$ to find all edges and $O(E \log E)$ to find smallest weights

b) Adjacency list

Time complexity : $O(E \log |V|)$

In we have to sort the edges into non decreasing order by weight 'w' for each edge.