ID: 20301357

Ans: to the question no: 04

For B.F.S:

For adjacency list,
As the ~~linear~~ time complexity depends on the number of edge and vertices we have. The more E and V we have more time ~~ca~~ will be needed to execute

So, we are adding new edges connected to vertices each time we get one.

and as, we can see on the Pseudocode there is one for loop so it should be the complexity around O(n) but as we are adding new edges connected to vertices so this n is proportional to (V+E), as it is traversing through all the

Vertiexes and edges.

∴ Time complexity = $O(V+E)$

Now,

for adjacency matrix, the time complexity is $O(n^2)$

As, for row and column, we have to iterate through nested - loops, therefor the time complexity is

$O(n^2)$

DFS

For adjacency list, the time complexity is ⇒ $O(\text{vertex} + 2 \cdot \text{Edge})$

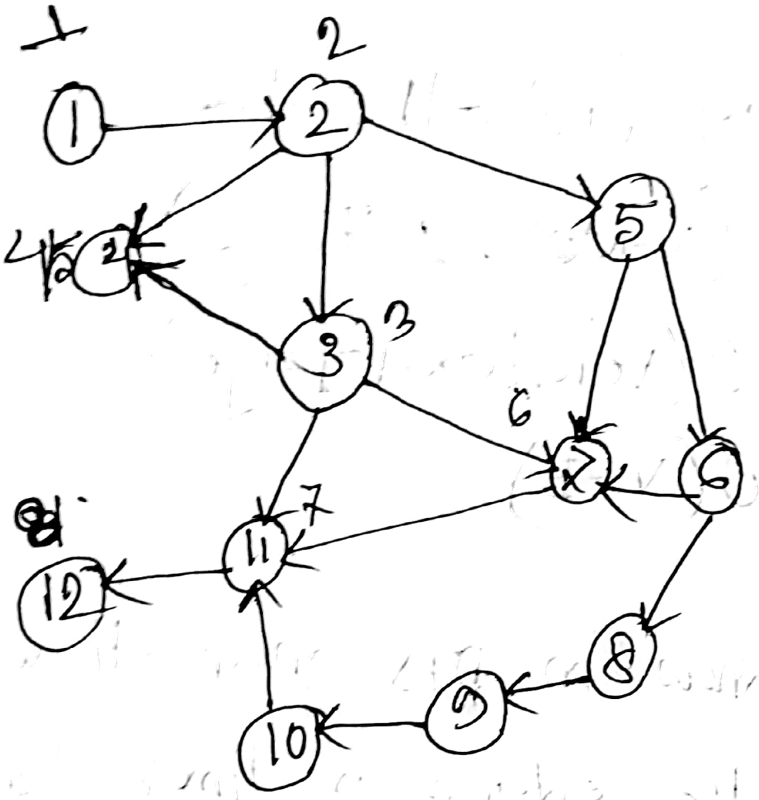
$\Rightarrow O(\text{vertex} + \text{Edge})$

$= O(V+E)$

Here we know in DFS graph traversal; it traverces the edges 2 times. for that we are multiplying it ~~by 2~~ the edge by 2.

For adjacency matrix, time complexity is $O(n^2)$ as for row and colum, we have to iterate through nested loops, therefore the time complexity $= O(n^2)$

Now,



for BFS,

Q → ~~1~~, ~~2~~, ~~3~~, ~~4~~, ~~5~~, ~~7~~, ~~11~~, ~~6~~, ~~12~~, 8

da → 1 2, 3, 4, 5, 7, 11, 6, 12 we got victory road.

for DFS,

1, 2, 3, 4, 7, 11, 12 we got victory road.

From the above simulation we can see those who are using DFS will gets to the victory road first

But ~~a~~ we know, we always use BFS for shortest Path. ~~But~~ But as we have to find through ~~it~~ the output, so, those who are using DFS will get to the victory road first .