Name: Rahul Rathod
Net id: rr4080

04/08/2022 Assignment 8

Q1

Given the number of nodes in the graph to be V, number of edges in the graph to be E, running time:

Adjacency list : $O(V+E)$
Adjacency matrix: $O(V^2)$

In DFS, we go through every node once & go through (visit) the one adjacent to them. In each case we shall require $O(V)$ time.
In case of adjacency list, we only have to visit every edge in every list, thus $O(E)$. ∴ Total running time is $O(V)+O(E) = O(V+E)$
In case of adjacency matrix, we have only E edges, thus when we go through

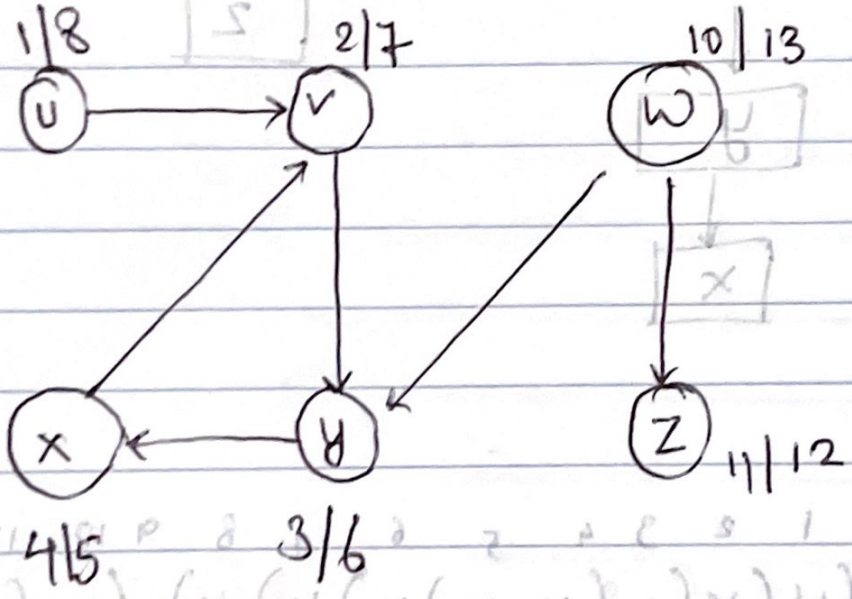
a node (for ex j^{th} node), we shall visit the element in j row for adjacency matrix. In all, we visit each element in the adjacency matrix, which is v^2 (for either 0 or 1),
Therefore, total running time is
$O(v^2) + O(v) = O(v^2)$

Q2

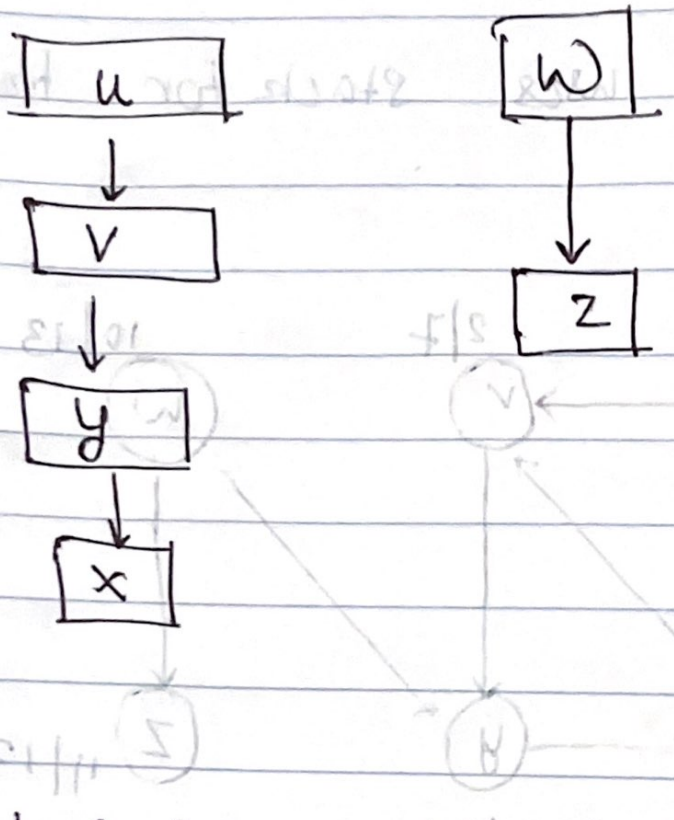
Running DFS on a tree T, setting the start node as the root & recording the discover time & finish time in each node. As T is a tree, no of edges are $n-1$. i.e. Total running time (number)
$= O(V+E) = O(n+n-1) = O(n)$.
As we know that, given any 2 nodes u, v in the DFS tree,

If $v.d < r.d < r.f < u.f$, u is the ancestor of r. Otherwise if, $v.d < u.d < v.f < r.f$ v is the ancestor of u. As we only need $O(n)$ time to complete DFS & store the discover time to finish time. Other operations have constant time $O(1)$, so total running time $= O(n)$. ∴ we only need $O(n)$ time to initialize & $O(1)$ to determine the relationship between 2 node.

Q3 DFS uses stack for tracing the graph



contd next page →



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

(u (v (y (x x) y) v) u) (w (z z) w)

Q4

```
def pPS(s, p, out=deque()):
if p<0:
    print → (list(out))
out.append(s[i])
pPS(s, p-1, out)
out.popl()
while p>0 and s[i] is equal to s[i-1]:
 p=p-1
pPS(s, i-1, out)
def PS(s):
sort s (s.sort())
pPS(s, len(s)-1)
```