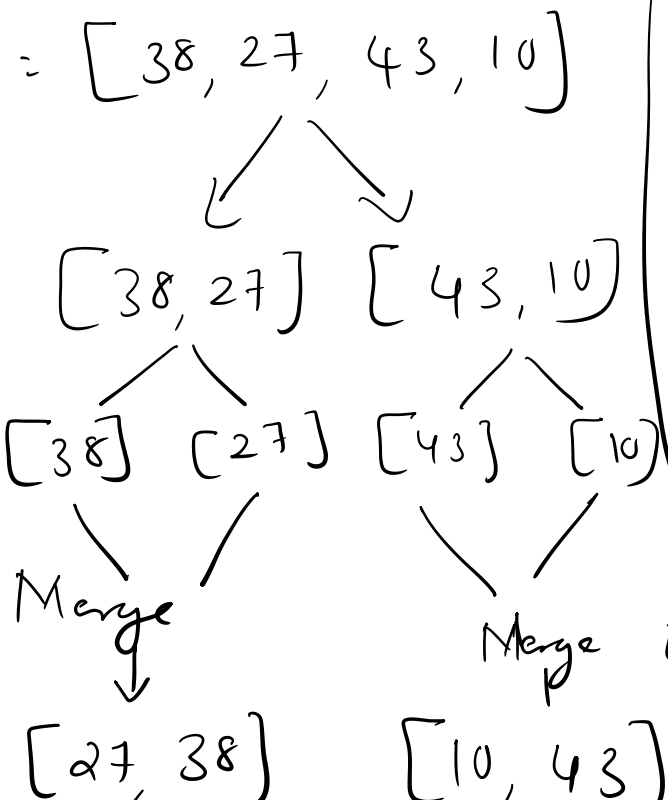
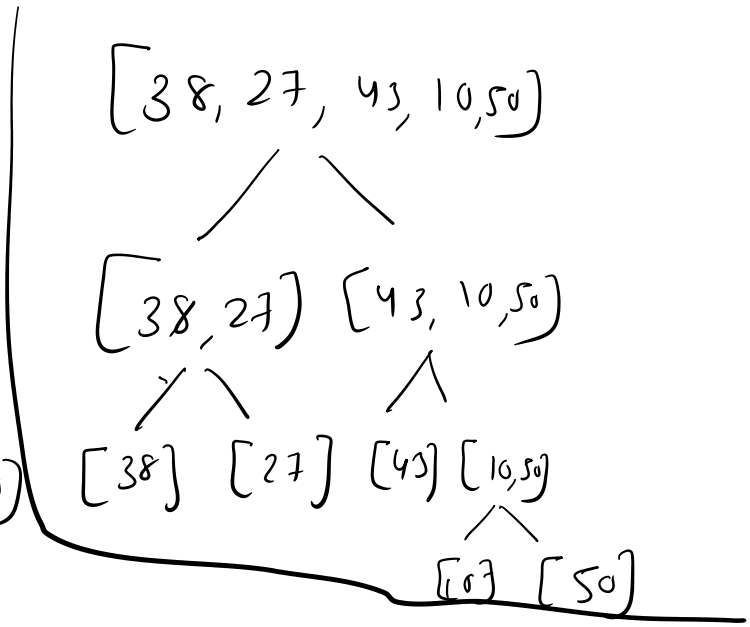
Merge Sort :-

lst = [38, 27, 43, 10]





43 < 10 ✗

27 < 10 ✗ 38 < 10 ✗

27 < 43 ✓ 38 < 43 ✓

Trees :-

Nodes

Root

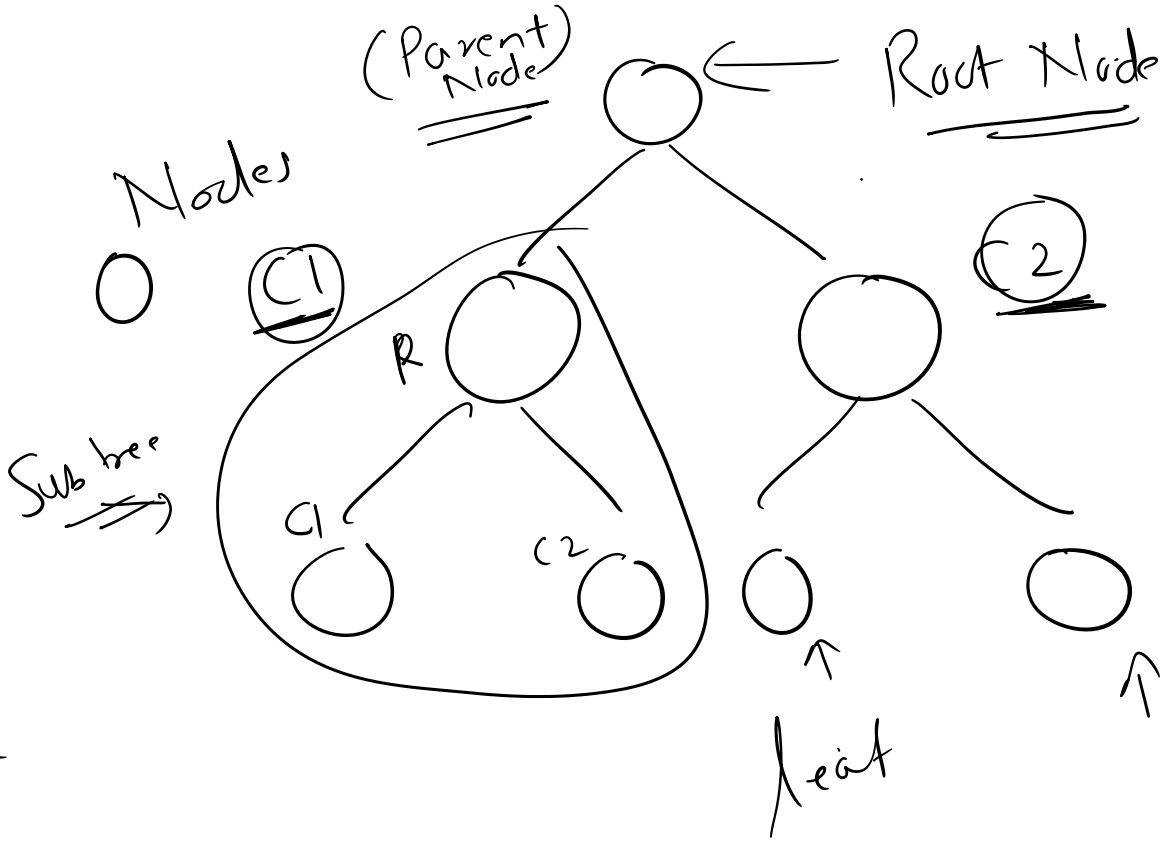
Parent

Child

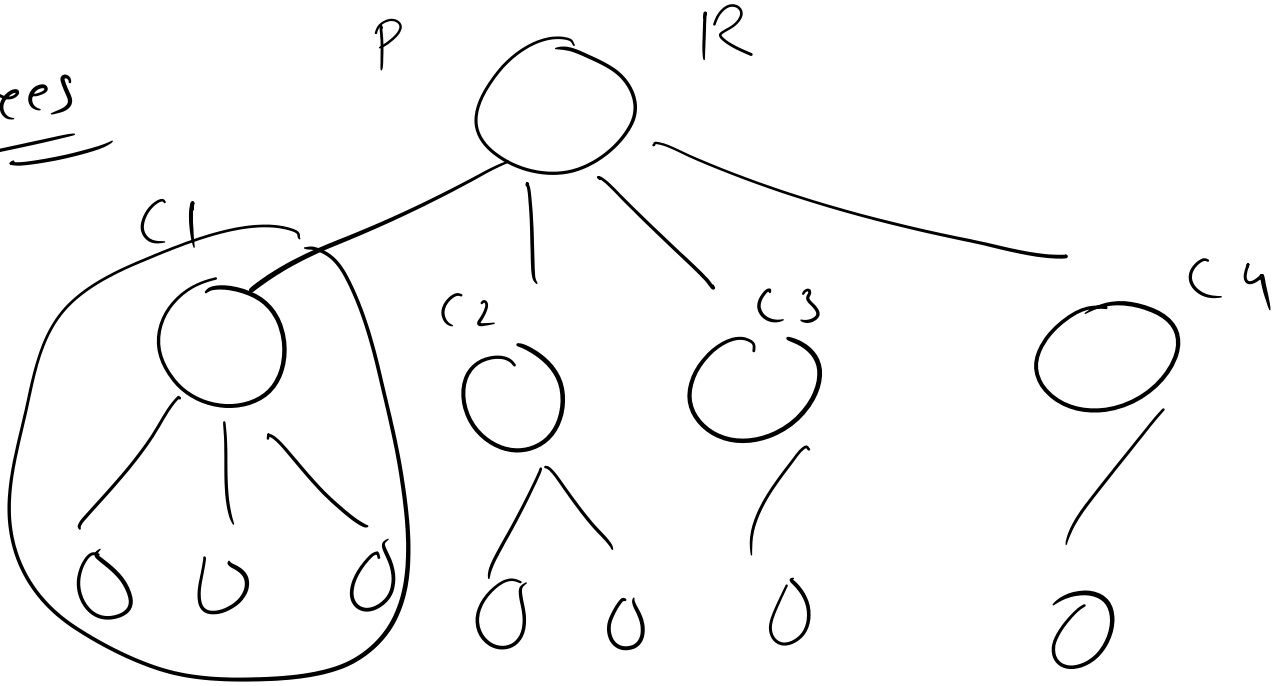
Leaf

Siblings

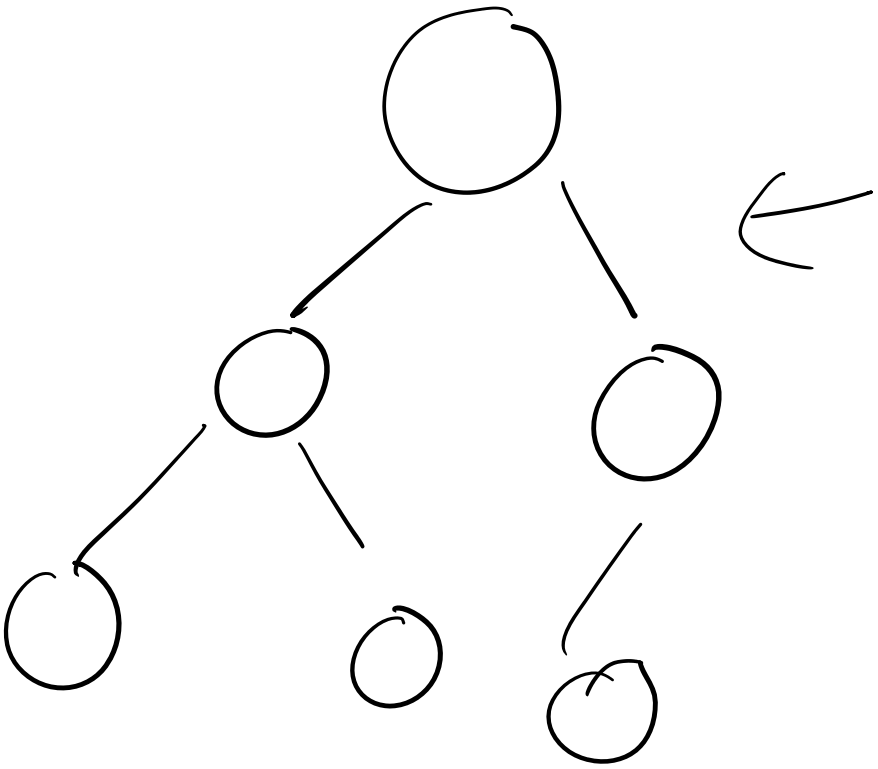
Sub tree



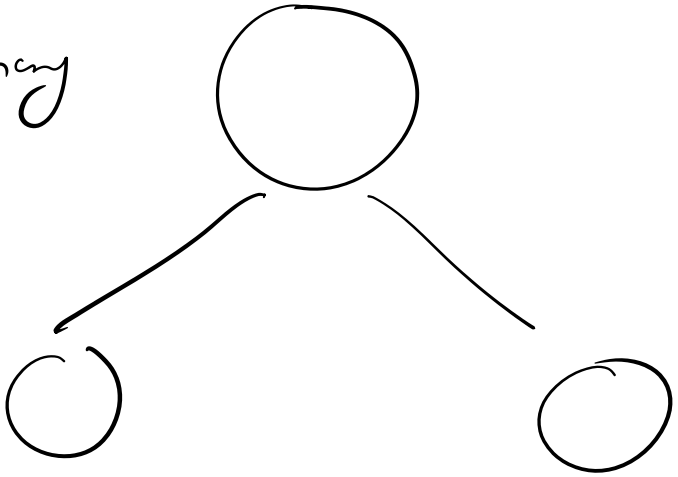
Trees

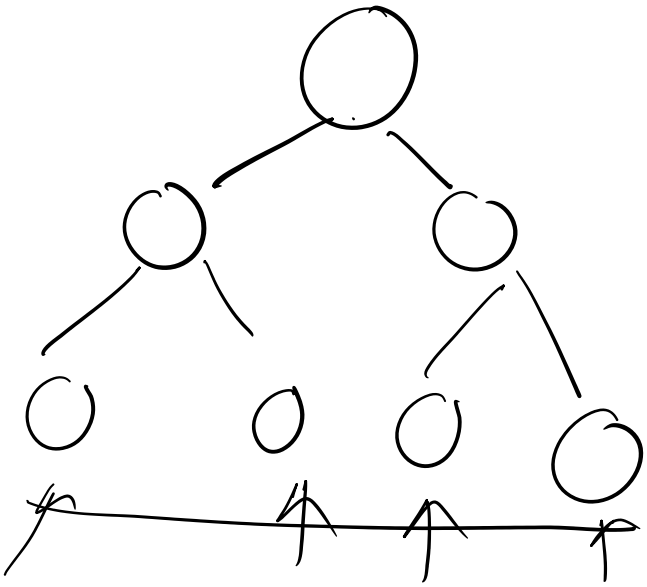


Binary tree :-



Perfect Binary tree





& Each node having exactly 2 children

← leaves at same level

Traversal :- The process of visiting each & every node of a tree in a specifice pre defined order is called as Traversal.

* Based on specific order there are 3 types of traversals

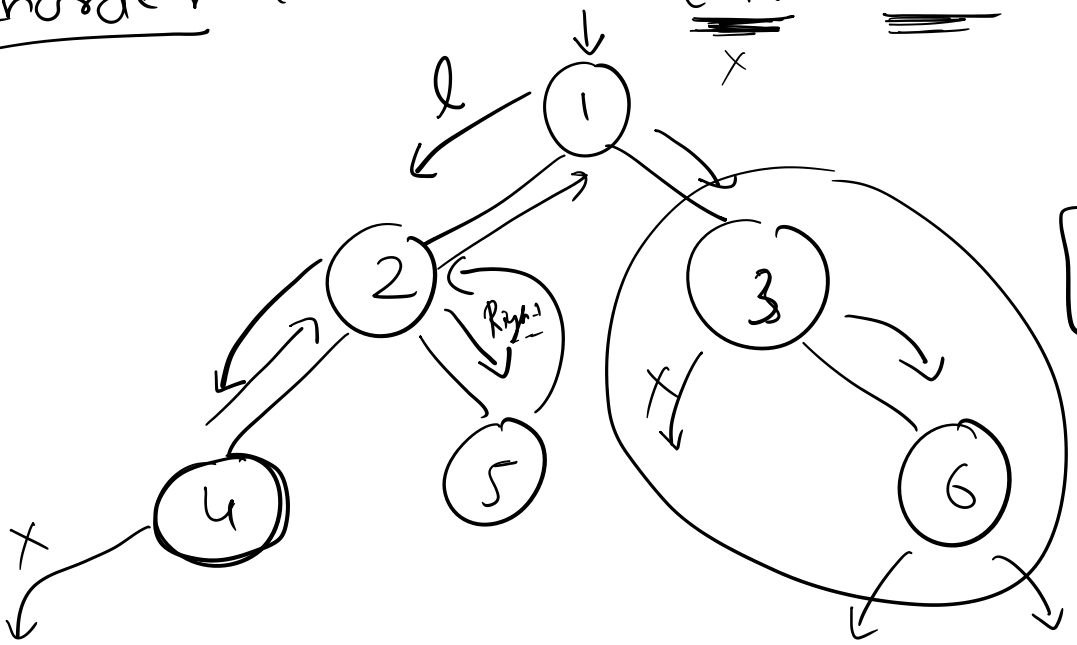
1. In-order traversal
2. Pre-order traversal
3. Post-order traversal.

Pre order traversal - Root → Left → right.

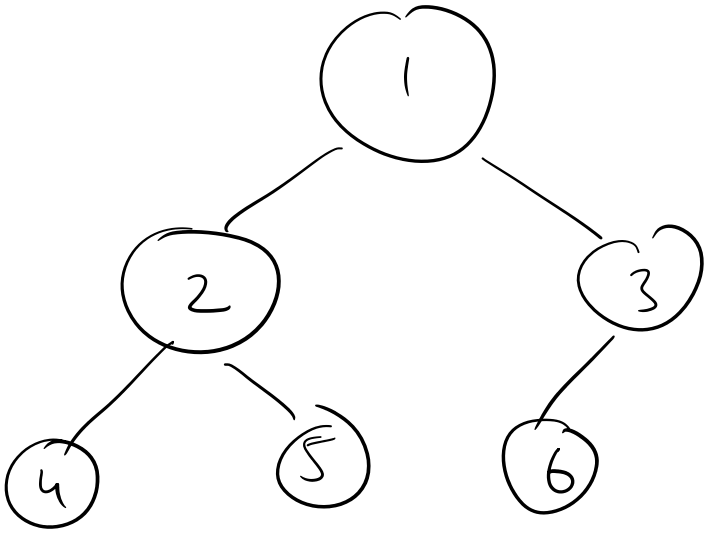
Inorder traversal - Left → Root → Right

Post traversal - Left → Right → Root.

Inorder traversal :- Left → Root → right.

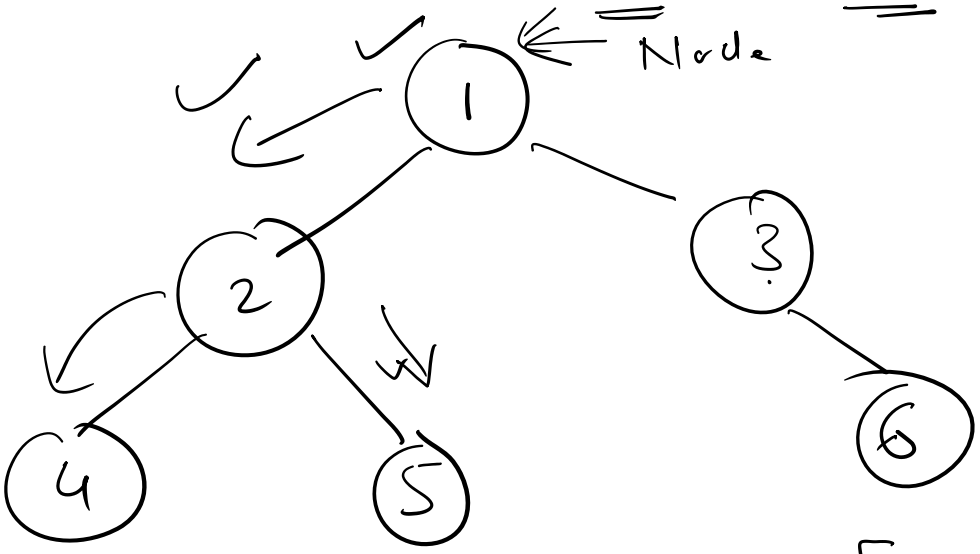


[4 → 2 → 5 → 1 → 3 → 6]



[4 → 2 → 5 → 1 → 6 → 3]

Pre order traversal :- Root → left → right

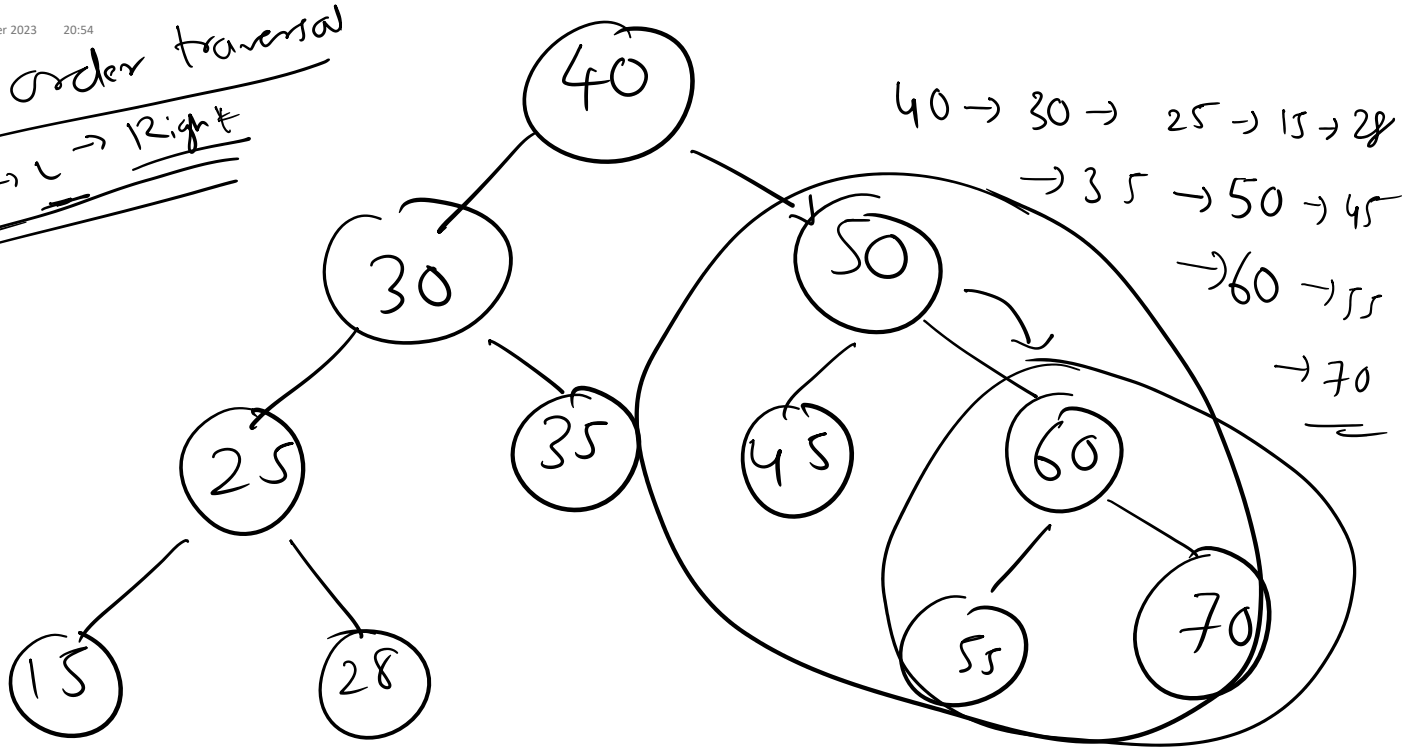


[1 → 2 → 4 → 5
→ 3 → 6]

27 September 2023 20:54

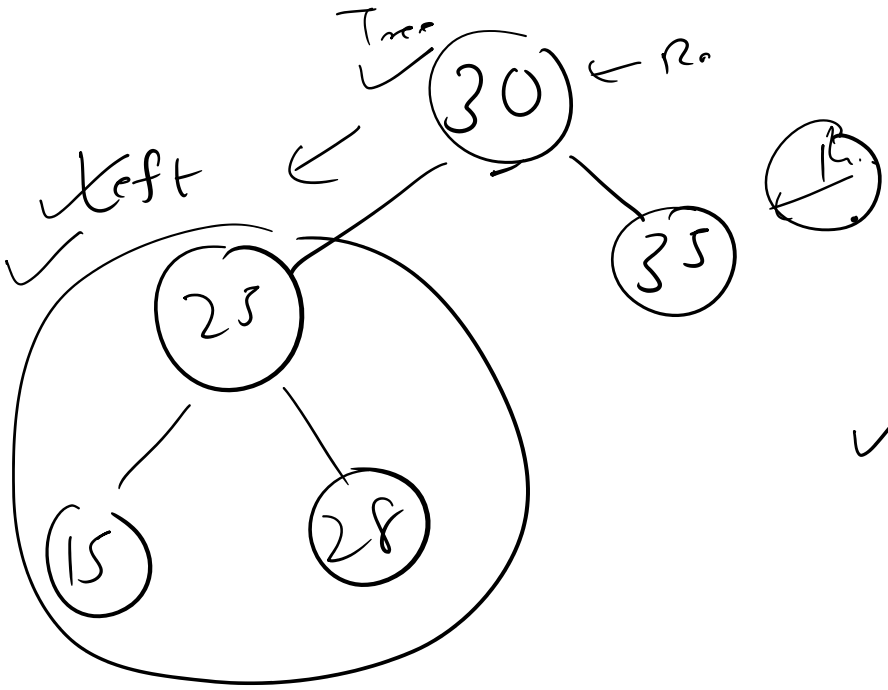
Pre order traversal

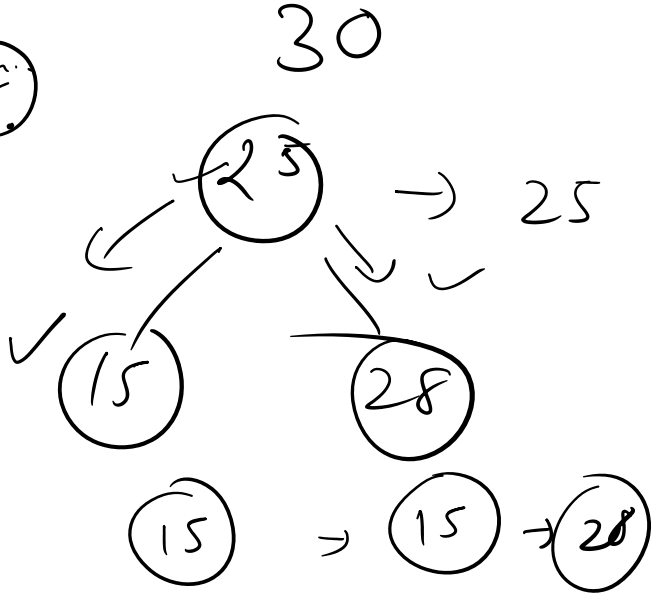
R → L → Right



40 → 30 → 25 → 15 → 28 → 35 → 50 → 45 → 60 → 55 → 70

R → L → Right

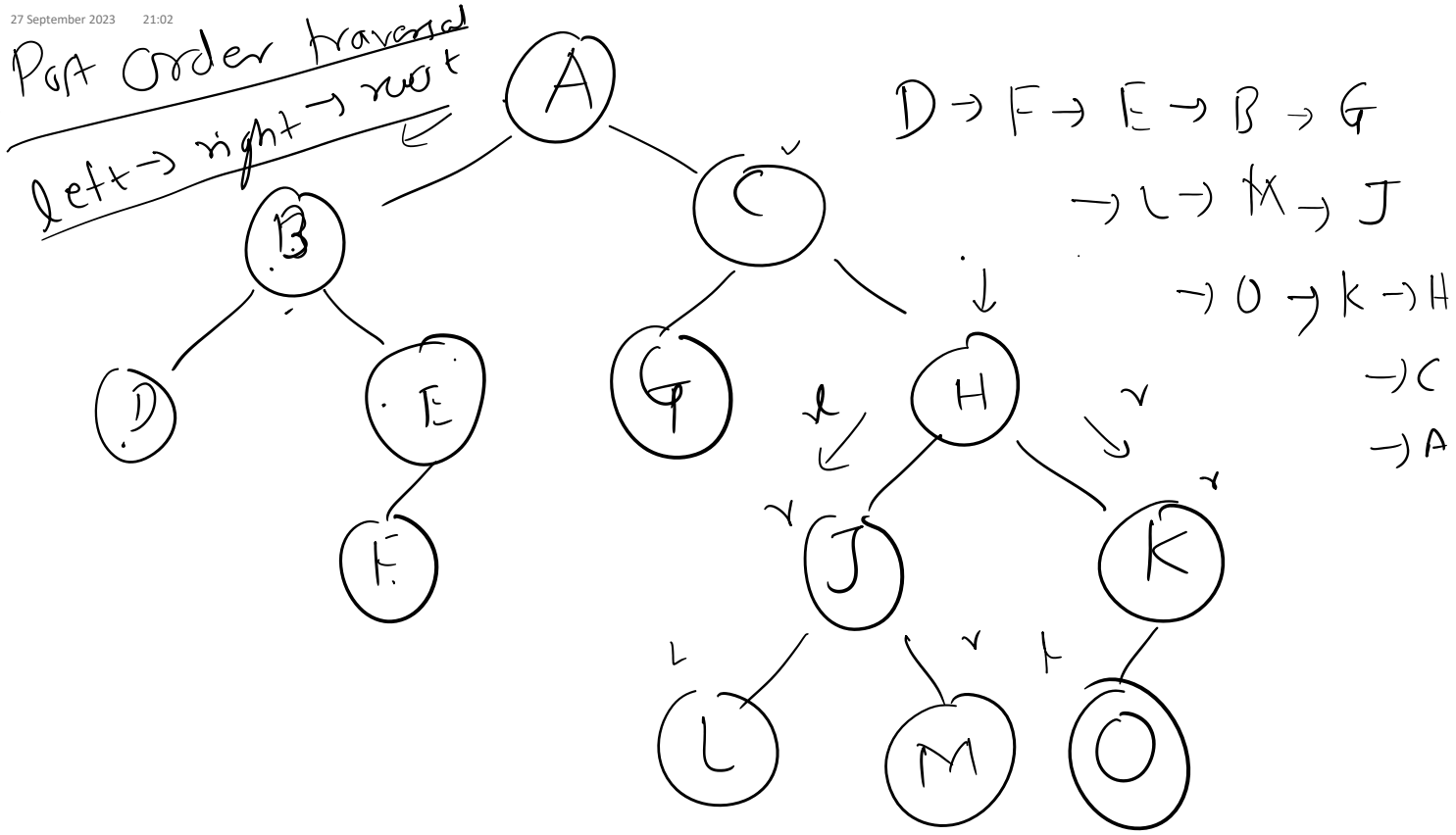




25 → 15 → 28

Post Order traversal

left → right → root



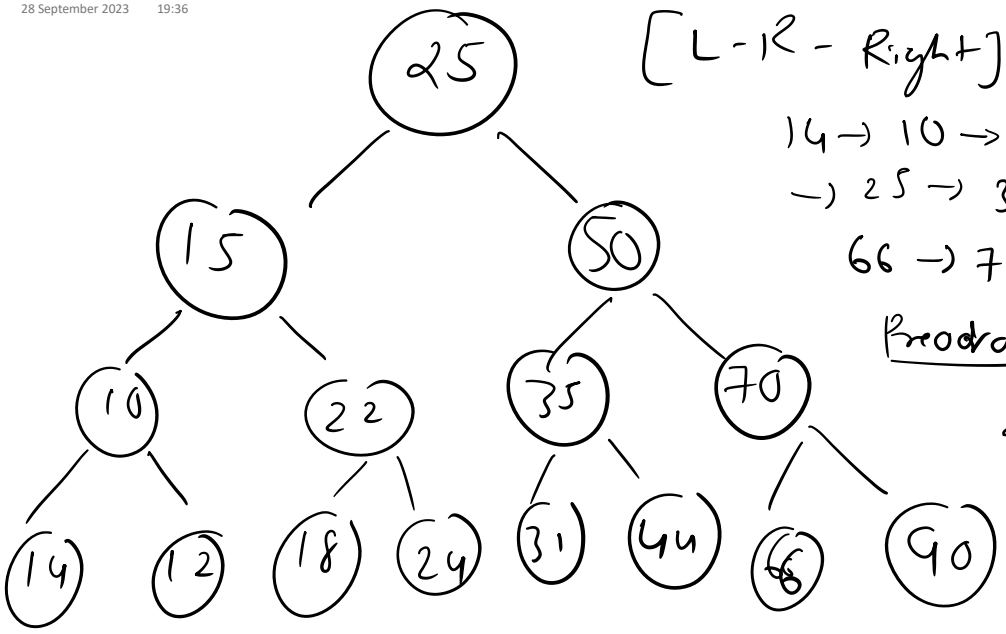
D → F → E → B → G

→ L → M → J

→ O → K → H

→ C

→ A



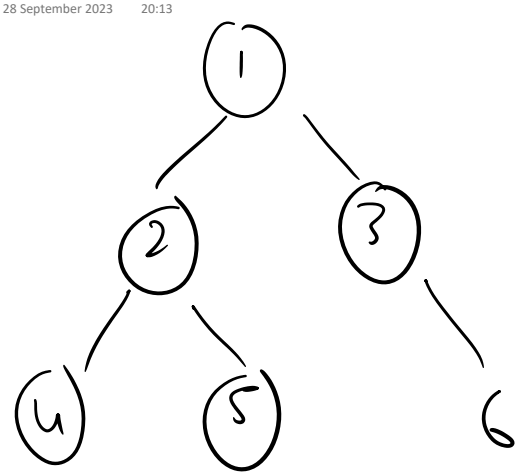
[L - R - Right] Inorder

14 → 10 → 12 → 15 → 18 → 22 → 24 → 25 → 31 → 35 → 44 → 50 → 66 → 70 → 90.

Preorder :- [Root - L - R]

25 → 15 → 10 → 14 → 12 → 22 → 18 → 24 → 50 → 35 → 31 → 44 → 70 → 66 → 90

Postorder :- 14 → 12 → 10 → 18 → 24 → 22 → 15 → 31 → 44 → 35 → 66 → 90 → 70 → 50 → 25



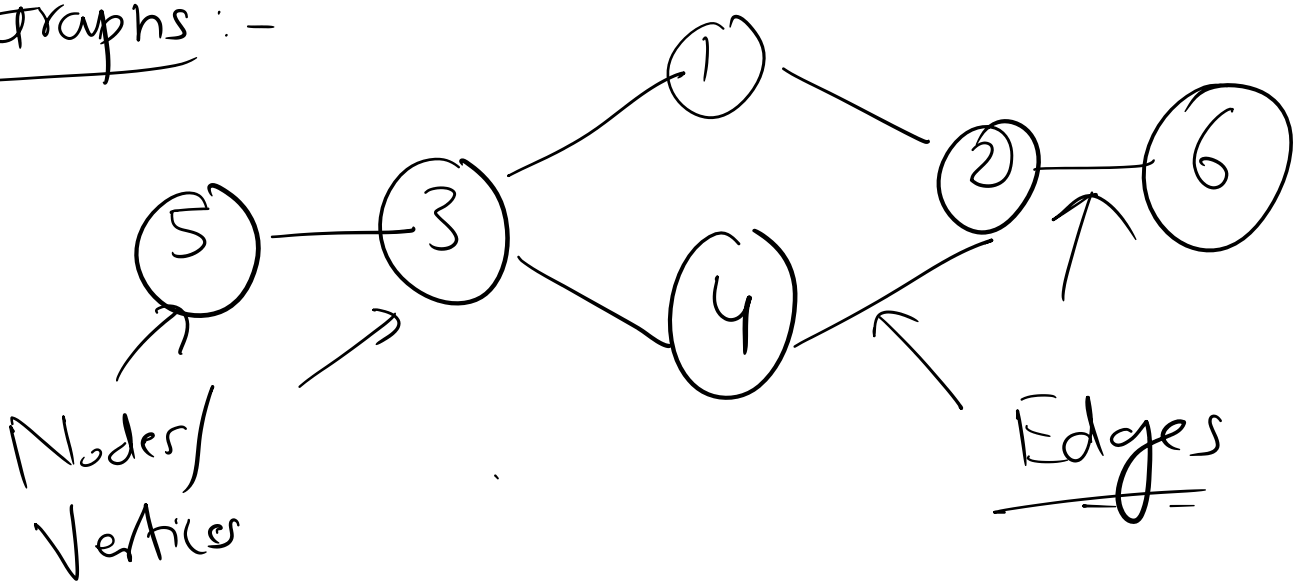
Pre order [Root - L - R]

1 → 2 → 4 → 5 → 3 → 6.

Post order [L - R - R]

4 → 2 → 5 → 6 → 3 → 1

Graphs :-



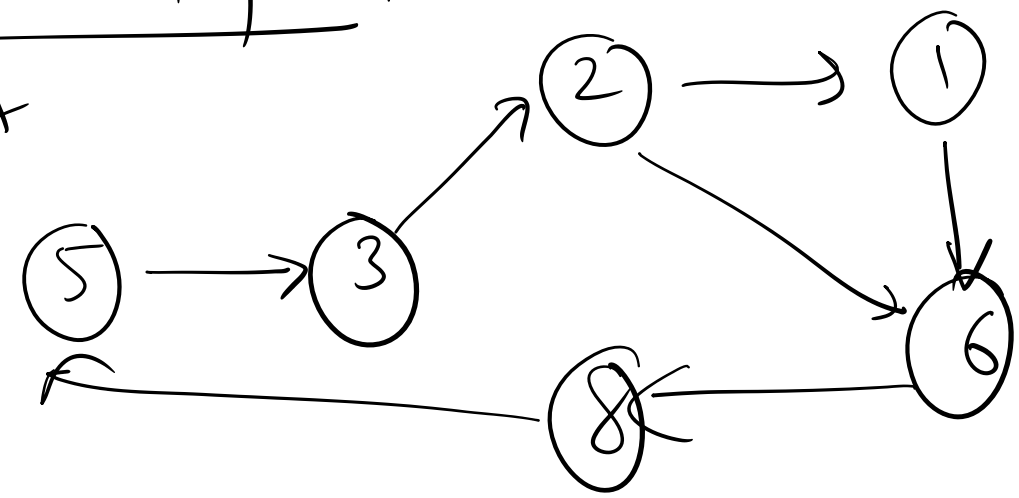
* A graph is a non-linear data structure containg of vertices & edges.

Directed Graphs :-

BFS

DFS

Shortest path

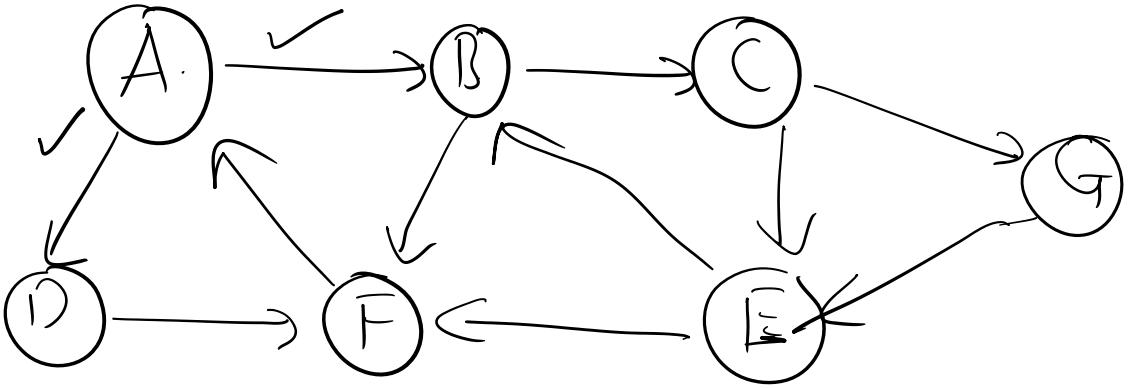


node 3 is neighbor for node 5

But node 5 is not a neighbor of node 3.

28 September 2023 20:27

Breadth First Search :



Task :- Start at Node A & reach Node E using shortest path.

BFS :- Breadth first Search 20:32 - 20:36

Idea of BFS to start with 2 queue

queue 1 & queue 2.

queue 1 (Q1) - hold all the nodes that are to be processed

queue 2 (Q2) - holds all the nodes that are already processed and deleted from queue 1

Step 1 :- First, add A to Q1 & NULL to Q2.

Q1 = [A] , Q2 = [NULL]

Step2 :- Now delete node A from Q1 ...

Step2 :- Now, delete node A from Q_1 & add it to Q_2., Insert all neighbors of node A to Q_1

$Q_1 = [B, D]$ $\quad Q_2 = [A]$

Step3 :- Now, del B from Q_1 & add it to Q_2. Insert neighbors of B to Q_1

$Q_1 = [D, C, F]$ $\quad Q_2 = [A, B]$

Step4 :- Now, del D from Q_1 & add it to Q_2 Insert neighbors of D to Q_1

$Q_1 = [C, F]$ $\quad Q_2 = [A, B, D]$

F is neighbor of D & since F is already in Q_1, ignore F.

Step5 :- Now, del C from Q_1 & add it to Q_2 Insert neighbor of C to Q_1

$Q_1 = [F, E, G]$ $\quad Q_2 = [A, B, D, C]$

Step6 :- Now, del F from Q_1 & add it to Q_2 Insert neighbors of F to Q_1

Insert neighbors of F to Q_1

$Q_1 = [E, G]$ $Q_2 = [A, B, D, E, F]$

neighbor of F is A. But do not add A as it is already visited.

Step7:- Del- E from q_1 & add it to Q_2

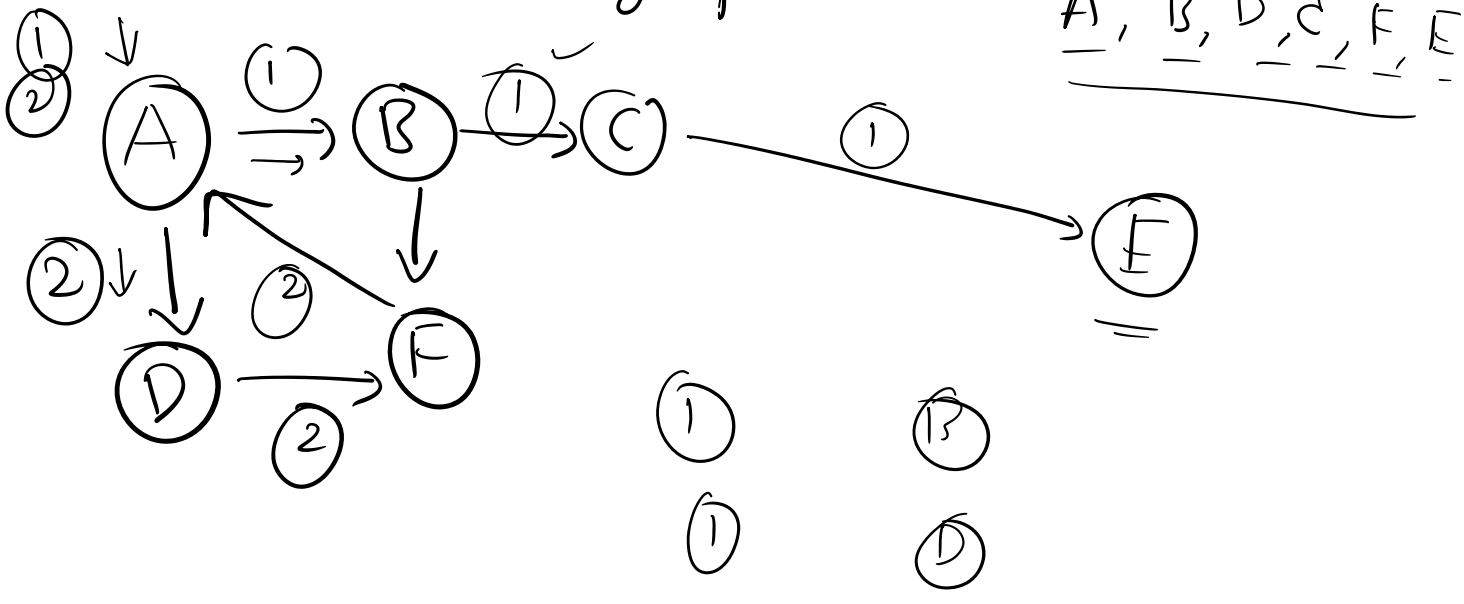
Insert neighbors of E to Q_1

$Q_1 = [G]$ $Q_2 = [A, B, D, C, F, E]$

$Q_2 = [A, B, D, C, F, E]$

Using Q_2 & my original graph construct a new graph

A, B, D, C, F, E



28 September 2023 21:12

DFS :- Depth First Search :-