Date / / Subject

BST - Operations

Creation: creating a tree from empty tree.

Insertion: Adding a value in a tree.

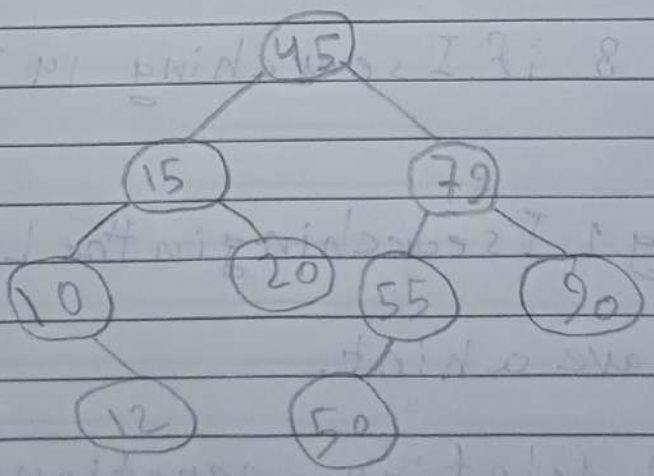
Searching: Finding a key value.

Deletion: Removing a value from tree.

Traversal: Accessing/visiting all values of a tree once

BST - Insert operation

Insert- 45, 15, 79, 90, 10, 55, 12, 20, 50

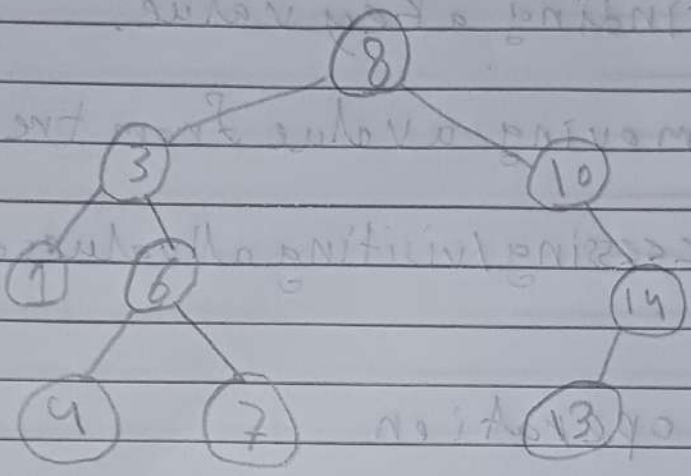


Date / / Subject

Binary Search Tree

value of left child is smaller than root

value of right child is greater or equal to the root



all values in left ≤ root

all values in right > root

ex:

the root is 8 if I searching 14 I searching in the right.

if I Searching 1 I searching in the Left.

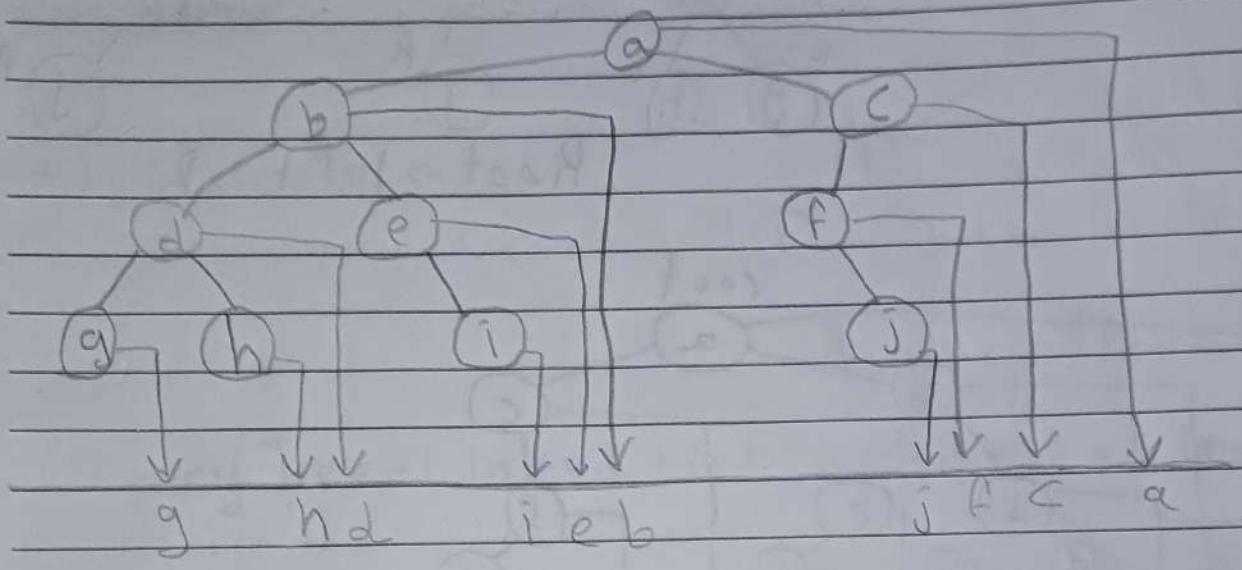
we always have a hint.

insertion and deletion operations are faster in BST.

Date / / Subject

Post Order Left → Right → Root

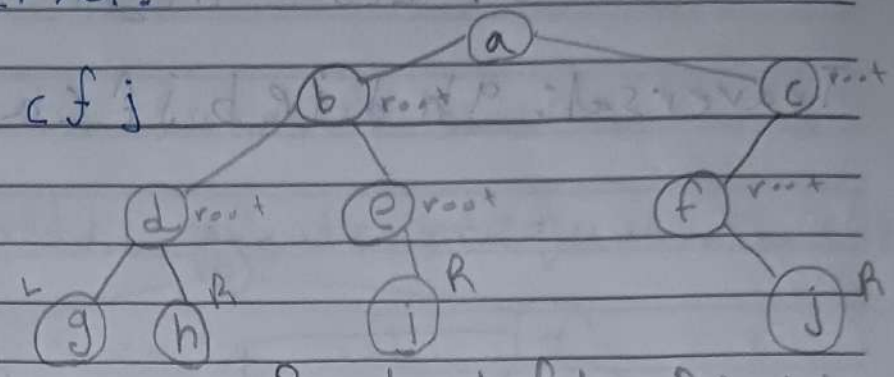
Traversal: g h d i e b j f c a



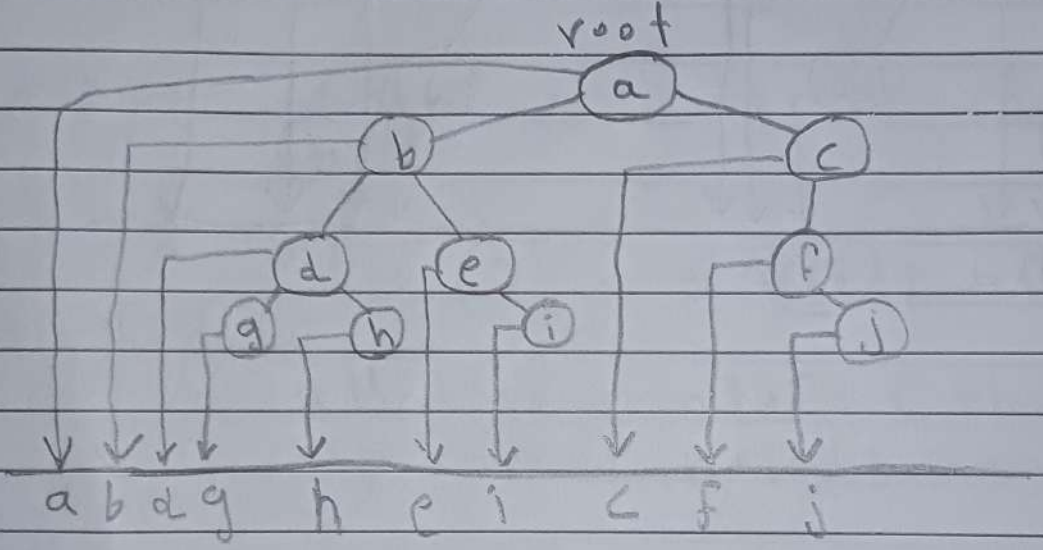
Date / / Subject

Pre Order Traversal:

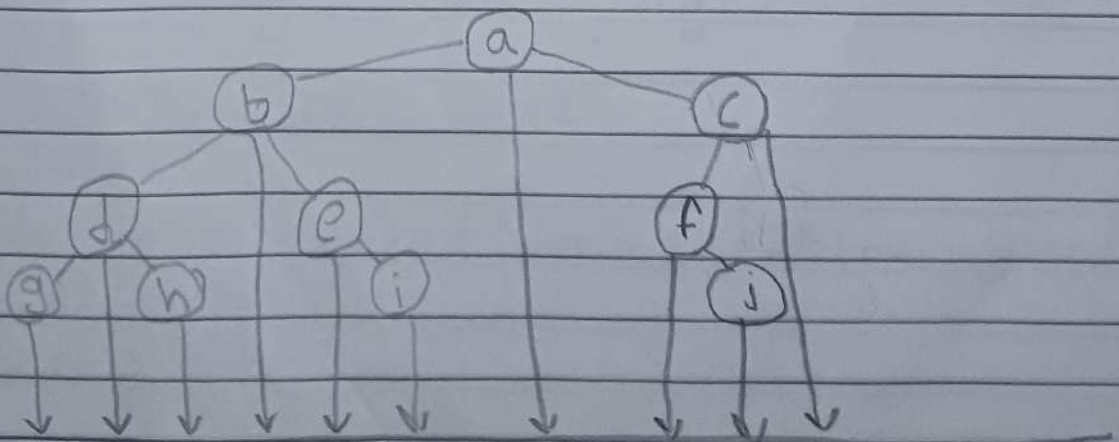
a b d g h e i c f j



Root -> left -> Right



In Order Traversal: g d h b e i a f j c

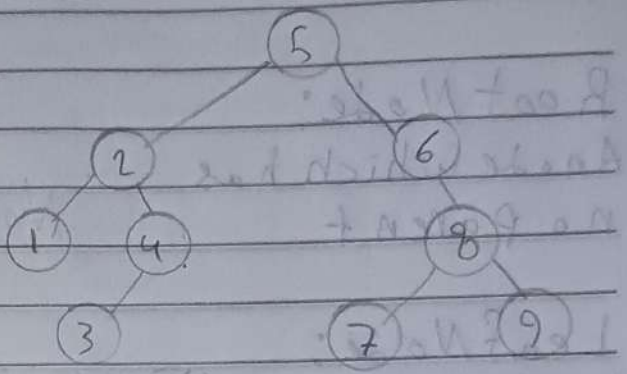


Left -> Root -> Right

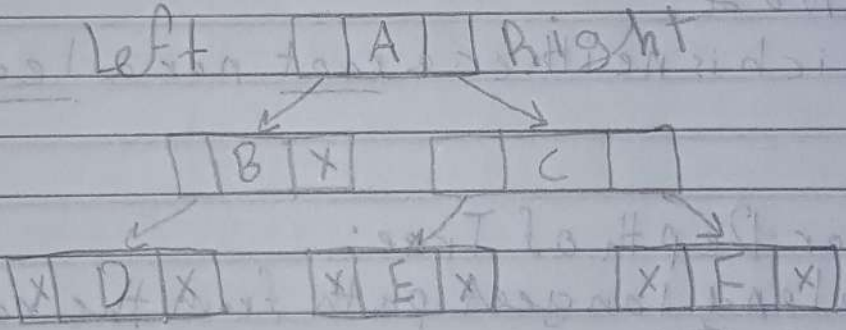
Date / / Subject

Binary Tree:

only one root
Maximum 2 children per node
Each node can have no child, one child or two children

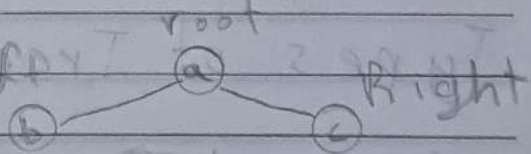


Binary Tree - Linked List Implementation



Binary Tree Traversal

3 - variants

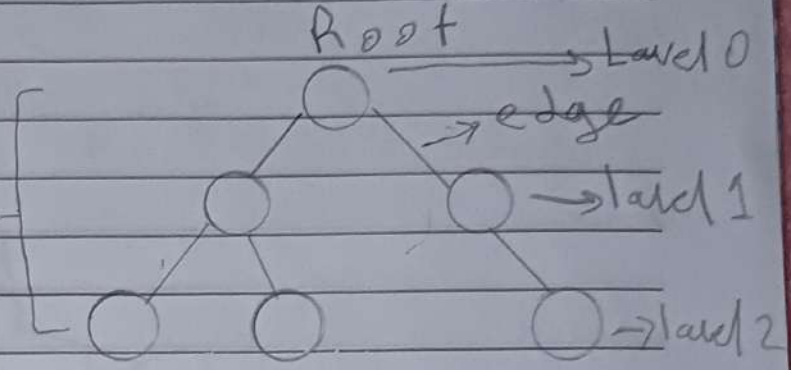


- Preorder: Root - Left - Right (a - b - c)
- In Order: Left - Root - Right (b - a - c)
- Post Order: Left - Right - Root (b - c - a)

Tree

Date / / Subject

A tree is a hierarchical data structure that organizes data elements, called nodes, by connecting them with links, called edges



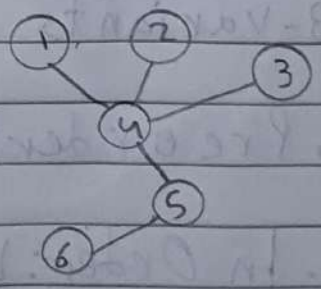
Root Node:
A node which has no parent

Leaf Node:
A node which has no child

Interior Node:
A node which is neither a root nor a leaf

Height or Depth of Tree:
of edges along longest path from the root to a leaf in that tree

Types of Tree



General Tree:
Any number of roots

Any number of children per node