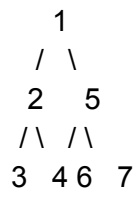
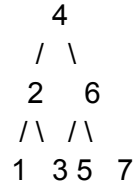
WEEK 3 Tutorial

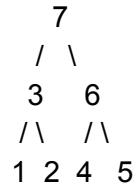
1. complele binary tree which preorder traversal is the sequence [1,2,3,4,5,6,7]



complele binary tree which inorder traversal is the sequence [1,2,3,4,5,6,7]



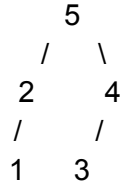
complele binary tree which post order traversal is the sequence [1,2,3,4,5,6,7]



2.Consider a binary tree with labels such that the postorder traversal of the tree lists the elements in increasing order. Let us call such a tree a post-order search tree. Describe how you will do search, min, max, insert and delete on this tree. Please write pseudo-code.

-The post order traversal follows the following order:
a)Visit the left subtree
b)Visit the right subtree
c)Visit the root node.

Since the post order traversal is in ascending order then this means all the elements of the left subtree is less than the elements in the right subtree which in tree is less than the root node element.



- **Search an element**

  Step1: A temporary variable is initialized to the root of the tree and if the item is greater than the root return NOT FOUND.

  Step2: If the node value is equal to the item to be searched return FOUND.

  Step 3: The item to be searched is checked with the left and right node value

  Step 4: If the element is less than the left node the left subtree is searched recursively for the item.

  Step 5: Else the right subtree is searched recursively.

  Step 6: If the temporary variable becomes NULL then the item is NOT FOUND.

- **Minimum element**

  The min element will be present at the left most leaf node in the tree.

  Step1:Two pointers are used to point the current node and its previous node.

  Step2:If the left subtree is not NULL then its left node is visited until the current node is NULL.

Step3:If the left subtree is NULL then the pointer is moved to the right node of the current node and repeat step 2.

Step 4: The minimum element is the value of the previous pointer

- **Max Element**

  The root is the max element in the array

- **Insert Element**

Step1:Input the data to be inserted
Step2:if the current node value is greater than the item to be inserted then make the current node as the left node of the item to be inserted.
Step3:If the item to be inserted is less than the right child of the current node then visit the left subtree recursively to find the position of the node in the left subtree.
Step4: Else visit the right subtree recursively.
Step5: If the item to be inserted is less than the current node value and greater than the left node of the current node and the right child is empty then enter it in the right child.
Step6: If the item is less than the right node of the current node value and the left child is empty, then make the item the left child of the current node

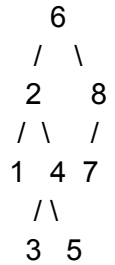
- **Delete Element**

step1:Input the data of the node to be deleted.
step2:If the node is a leaf node, delete the node directly.

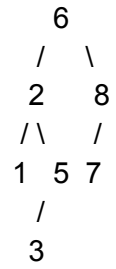
step3:Else if the node has one child, copy the child to the node to be deleted and delete the child node.

step4:Else if the node has two children replace it with its right node recursively.

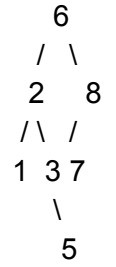
3. given that post order traversal = 1,3,5,4,2,7,8,6
in order traversal will be in the sorted order as this tree is BST
so, the in order traversal will be 1,2,3,4,5,6,7,8



deleting 4 and replacing with inorder successor



deleting 4 with inorder predecessor



4.Given a BST T and an element a, the task is to delete all elements b<a from T. Write pseudocode to do this. How much time does your algorithm take?

Step1:Traverse the BST from root to the left

Step2: If the value of the left child node is less than the value ‘a’ then replace the node with its right child and visit the

Step3:Move the pointer to the right child of the current node and repeat step 2.

Since we are traversing the height of the tree so the time complexity of the algorithm is O(height of the BST).