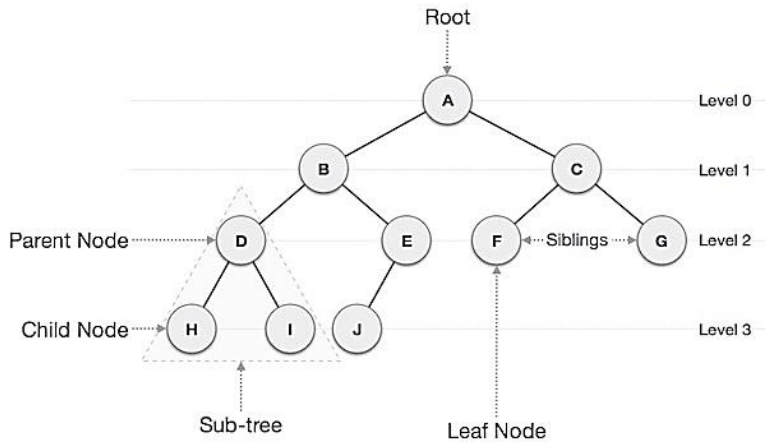
Practical – 9
Implementation of Tree and Searching

Tree

Tree represents the nodes connected by edges. We will discuss binary tree or binary search tree specifically.

Binary Tree is a special datastructure used for data storage purposes. A binary tree has a special condition that each node can have a maximum of two children. A binary tree has the benefits of both an ordered array and a linked list as search is as quick as in a sorted array and insertion or deletion operation are as fast as in linked list.



Tree Node

```
struct node {
   int data;
   struct node *leftChild;
   struct node *rightChild;
};
```

Binary Tree Traversal

Binary tree traversing is a process of accessing every node of the tree and exactly once. A tree is defined in a recursive manner. Binary tree traversal also defined recursively.

There are three techniques of traversal:

1. Preorder Traversal
2. Postorder Traversal
3. Inorder Traversal

1. Preorder Traversal

Algorithm for preorder traversal

Step 1 : Start from the Root.

Step 2 : Then, go to the Left Subtree.

Step 3 : Then, go to the Right Subtree.

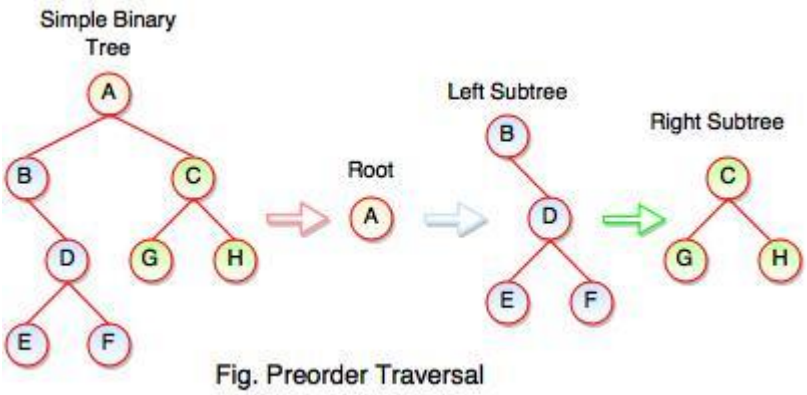


Fig. Preorder Traversal

The above figure represents how preorder traversal actually works.

Following steps can be defined the flow of preorder traversal:

Step 1 : A + B (B + Preorder on D (D + Preorder on E and F)) + C (C + Preorder on G and H)

Step 2 : A + B + D (E + F) + C (G + H)

Step 3 : A + B + D + E + F + C + G + H

Preorder Traversal : A B C D E F G H

2. Postorder Traversal

Algorithm for postorder traversal

Step 1 : Start from the Left Subtree (Last Leaf).

Step 2 : Then, go to the Right Subtree.

Step 3 : Then, go to the Root.

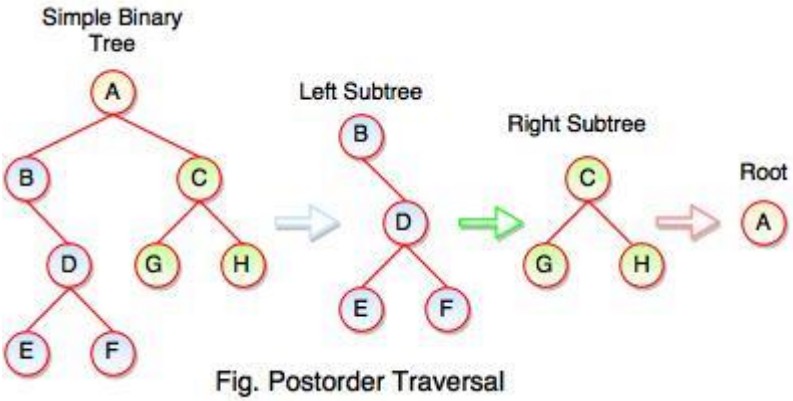


Fig. Postorder Traversal

The above figure represents how postorder traversal actually works.

Following steps can be defined the flow of postorder traversal:

Step 1 : As we know, preorder traversal starts from left subtree (last leaf) ((Postorder on E + Postorder on F) + D + B)) + ((Postorder on G + Postorder on H) + C) + (Root A)

Step 2 : (E + F) + D + B + (G + H) + C + A

Step 3 : E + F + D + B + G + H + C + A

Postorder Traversal : E F D B G H C A

3. Inorder Traversal

Algorithm for inorder traversal

Step 1 : Start from the Left Subtree.

Step 2 : Then, visit the Root.

Step 3 : Then, go to the Right Subtree.

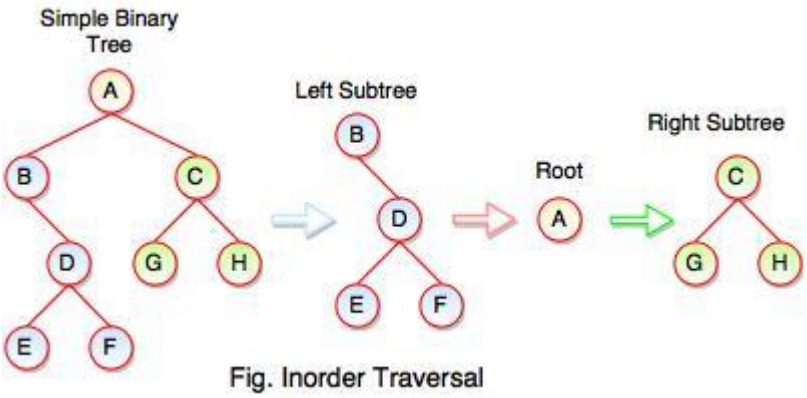


Fig. Inorder Traversal

The above figure represents how inorder traversal actually works.

Following steps can be defined the flow of inorder traversal:

Step 1 : B + (Inorder on E) + D + (Inorder on F) + (Root A) + (Inorder on G) + C (Inorder on H)

Step 2 : B + (E) + D + (F) + A + G + C + H

Step 3 : B + E + D + F + A + G + C + H

Inorder Traversal : B E D F A G C H

Searching

Searching is an operation or a technique that helps finds the place of a given element or value in the list. Any search is said to be successful or unsuccessful depending upon whether the element that is being searched is found or not. Some of the standard searching technique that is being followed in the data structure is listed below:

- Linear Search or Sequential Search
- Binary Search

Binary Search

```
# Input: Sorted Array A, integer key
# Output: first index of key in A,
# or -1 if not found
Algorithm: Binary_Search (A, left, right)
left = 0, right = n-1
while left < right
  middle = index halfway between left, right
  if A[middle] matches key
     return middle
  else if key less than A[middle]
     right = middle -1
  else
     left = middle + 1
return -1
```

Exercise

1. Write algorithm and implement program for creating a binary tree, traversing a binary tree using preorder, inorder and postorder
2. Write a program for searching an element in an array using linear search.
3. Write a program to implement binary search.