Tree traversal

- Traversal is a process to visit all the nodes of a tree and may print their values too.

- All nodes are connected via edges (links) we always start from the root (head) node.

- There are three ways which we use to traverse a tree
  - In-order Traversal
  - Pre-order Traversal
  - Post-order Traversal

- Generally we traverse a tree to search or locate given item or key in the tree or to print all the values it contains.

Pre-order, In-order, Post-order

- Pre-order

<root><left><right>

- In-order

<left><root><right>

- Post-order

<left><right><root>

Pre-order Traversal

- The preorder traversal of a nonempty binary tree is defined as follows:
  - Visit the root node
  - Traverse the left sub-tree in preorder
  - Traverse the right sub-tree in preorder

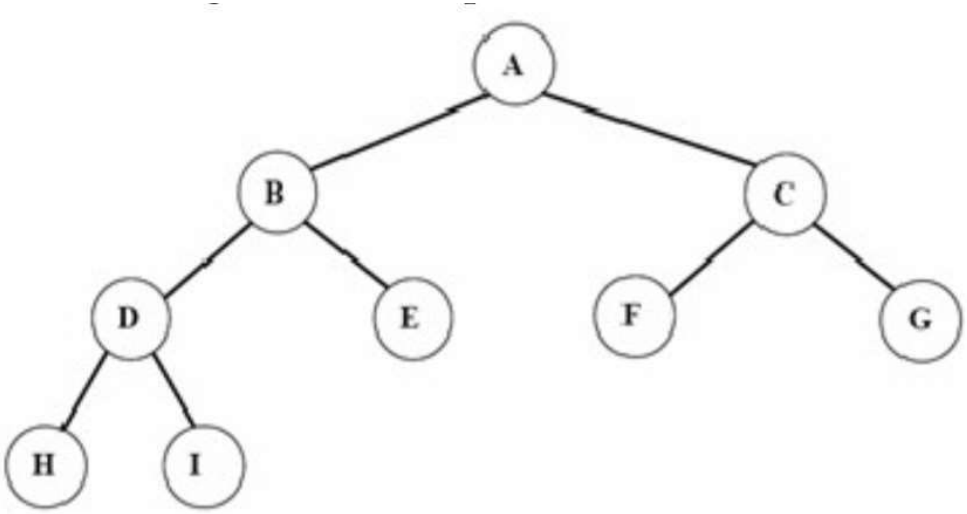


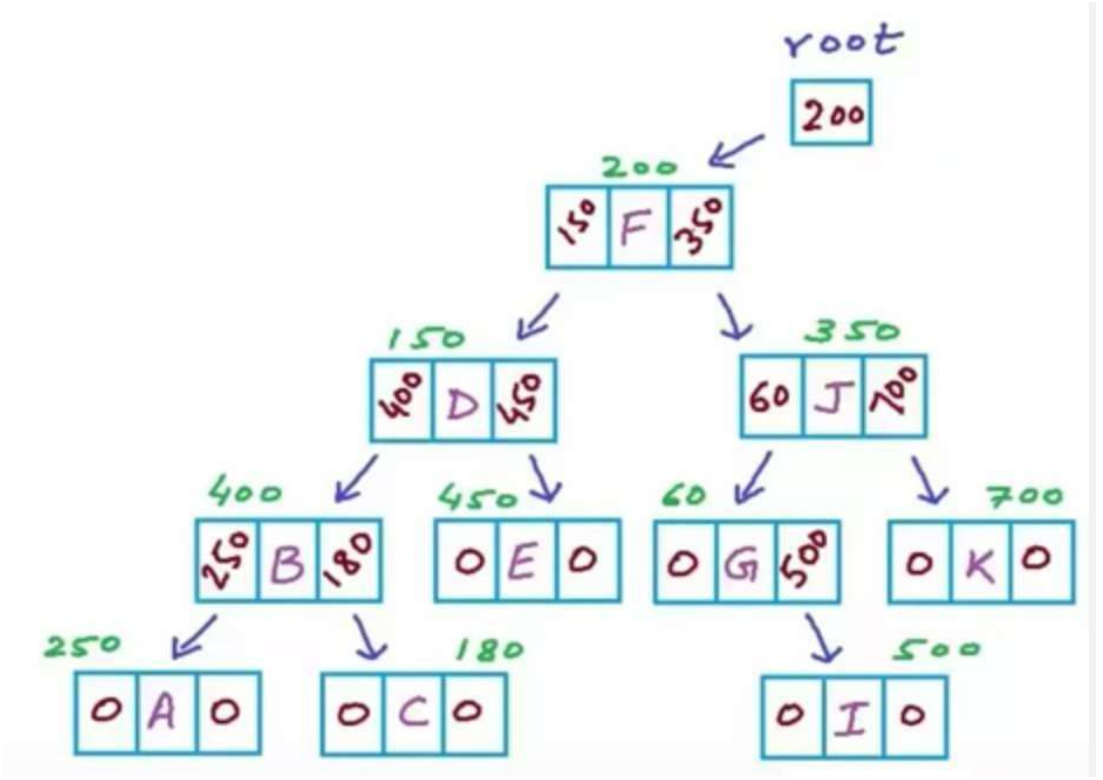
fig Binary tree

The preorder traversal output of the given tree is: A B D H I E C F G

The preorder is also known as depth first order.

Pre-order Pseudocode

```
struct Node{  char data;
   Node *left;  Node
   *right;
}
void Preorder(Node
   *root)
{
if (root==NULL) return;
   printf (“%c”, root-
   >data);
Preorder(root->left);
   Preorder(root->right);
}
```



In-order traversal

- The in-order traversal of a nonempty binary tree is defined as follows:
  - Traverse the left sub-tree in in-order
  - Visit the root node
  - Traverse the right sub-tree in inorder

- The in-order traversal output of the given tree is

H D I B E A F C G

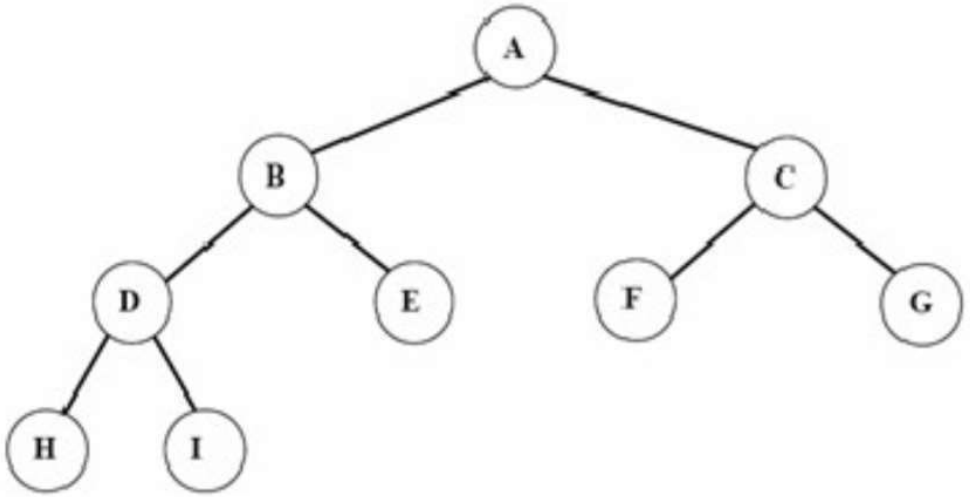
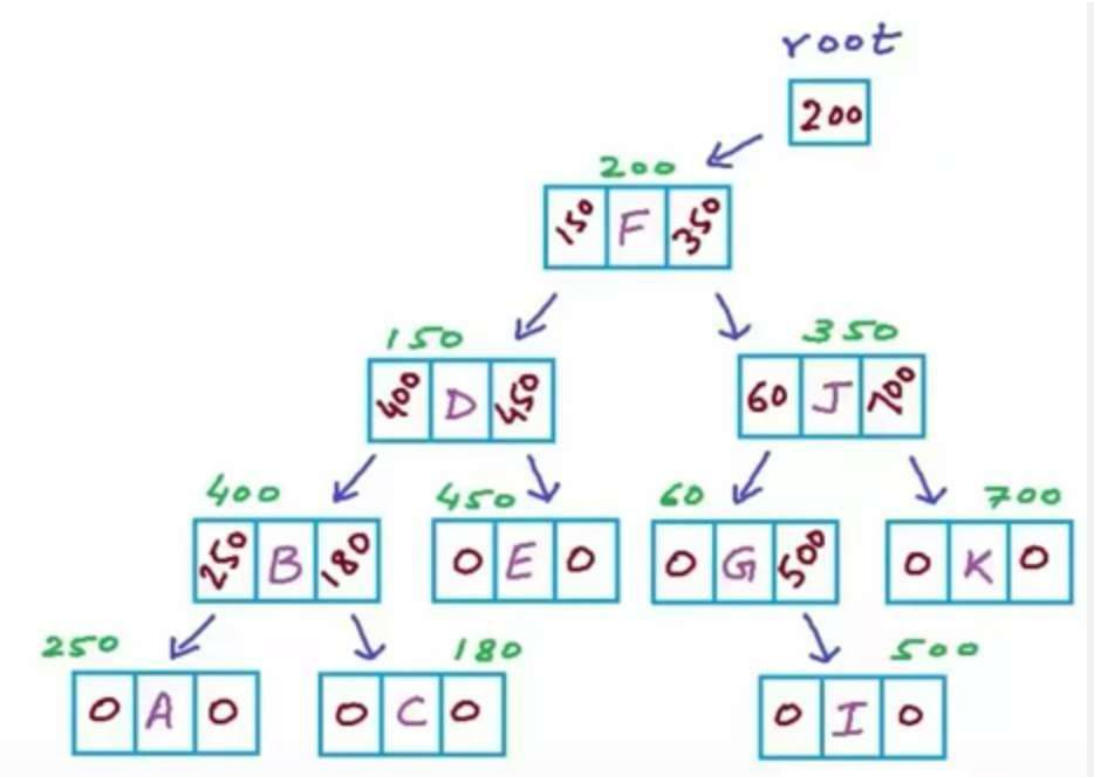


fig Binary tree

In-order Pseudocode

```
struct Node{  char data;
    Node *left;  Node
    *right;
}
void Inorder(Node *root)
{
if (root==NULL) return;
    Inorder(root->left);
printf (“%c”, root->data);
    Inorder(root->right);
}
```



Post-order traversal

- The in-order traversal of a nonempty binary tree is defined as follows:
  - Traverse the left sub-tree in post-order
  - Traverse the right sub-tree in post-order
  - Visit the root node

- The in-order traversal output of the given tree is

H I D E B F G C A

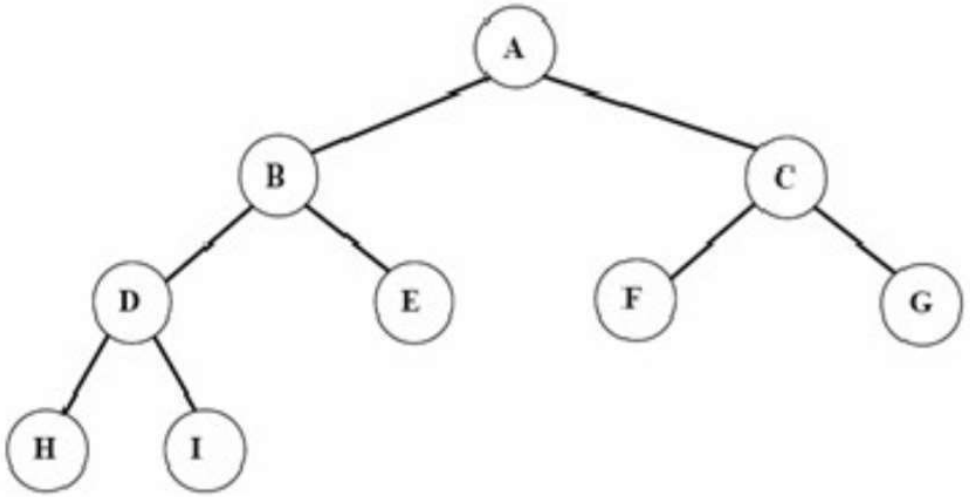


fig Binary tree

Post-order Pseudocode

```
struct Node{  char data;
   Node *left;  Node
   *right;
}
void Postorder(Node
   *root)
{
if (root==NULL) return;
   Postorder(root->left);
Postorder(root->right);
   printf (“%c”, root-
   >data);
}
```