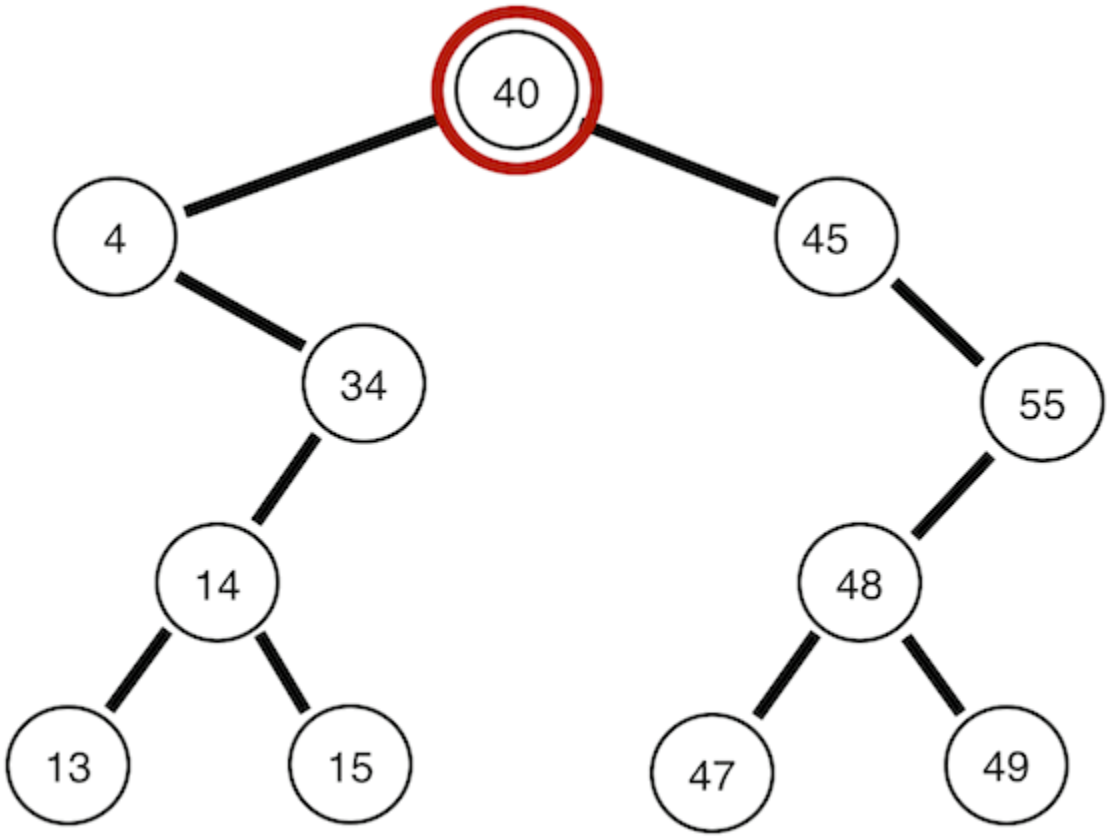
Traversal binary trees are a form of graph traversal that each node in the binary tree represent is visited from a certain data structure algorithm. In Computer Science, we have three type of traversal binary trees in which don't act like the same way with typical linear data structures such as Array, Linked List, Queues, Stacks, etc. The three types of traversal binary trees are **Pre-Order, In-Order, and Post-order traversal.**

Pre-Order Traversal:

In pre-ordered traversal, we create a copy of the binary tree to get a prefix expression on the expression tree. In this case, we go as far as we can to the left and thus as far as to the right in the binary tree.

In this example, the traversal goes like **40, 4, 34, 14, 13, 15, 45, 55, 48, 47, and then 49.**



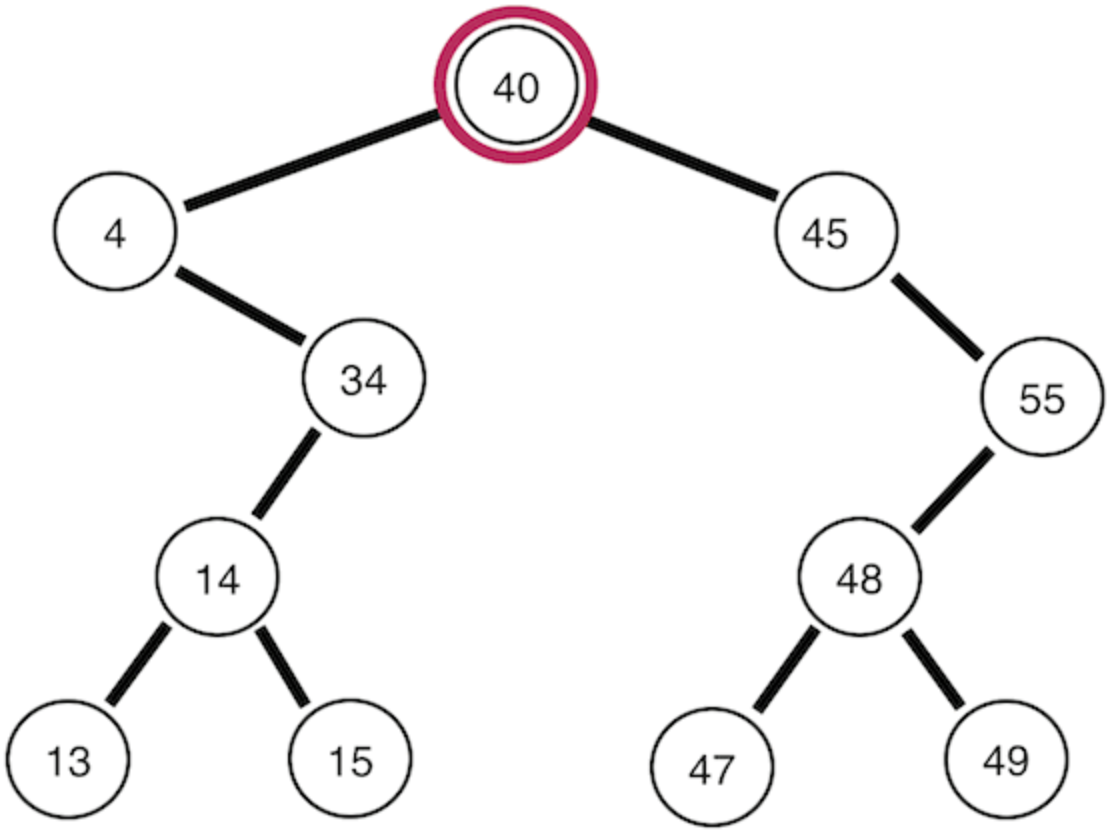
Example C++ Code:

```
//Given struct Node

struct Node
{
    int data;
    Node* left;
    Node* right;
};


//Preorder Traversal
```

```
void preOrder(struct Node* node)

{

     if (node == NULL)

          return;

 cout<<"  "<<data<<"  "<<endl;
 preOrder(node->left);
 preOrder(node->right);
}
```

In-Order Traversal:

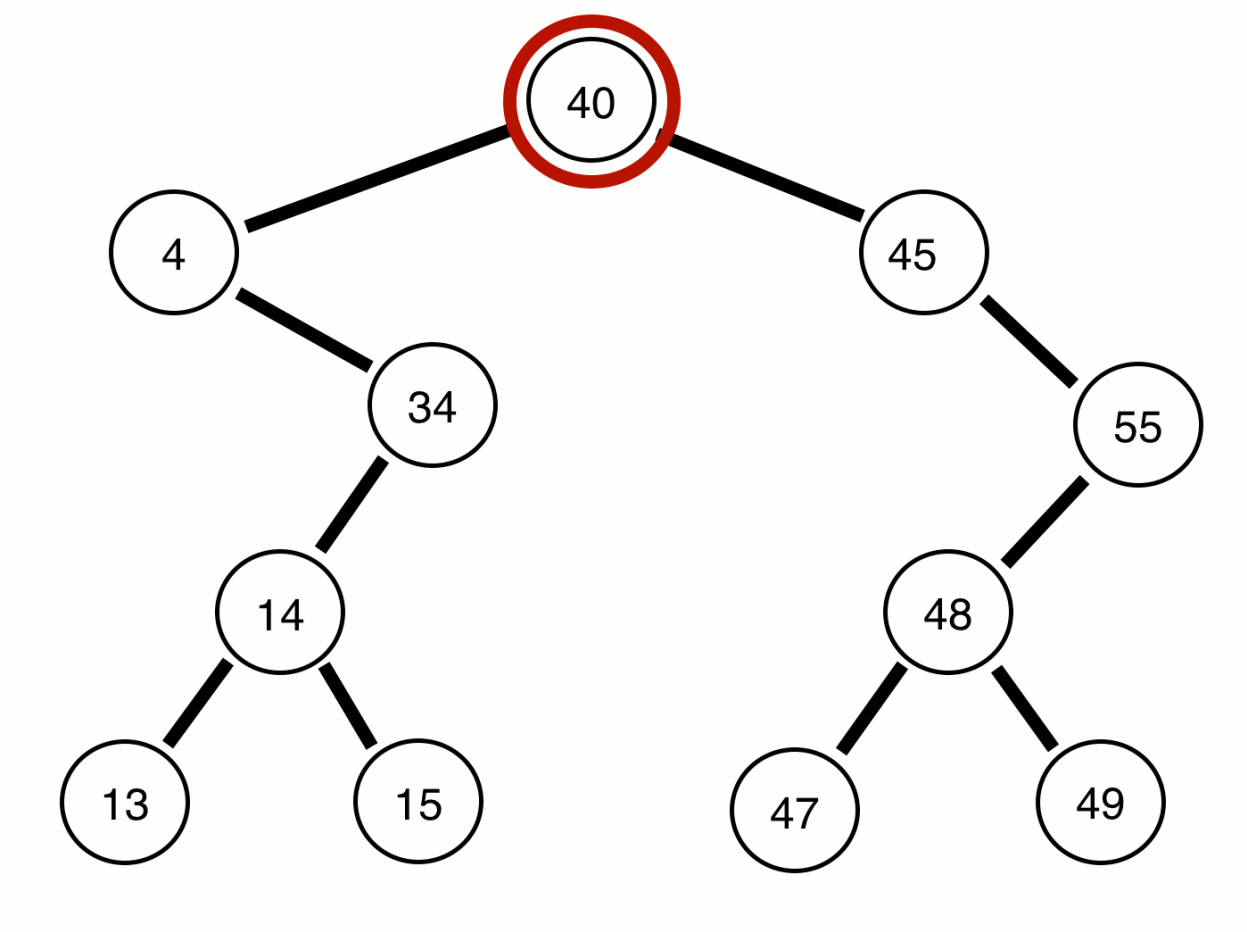
In-order traversal is a binary traversal that gives the nodes in a non-decreasing order manner. In order to get nodes of the binary search trees in
decreasing order, a variation of in order traversal can be used only where in-order traversal is reversed. In this example, the order in this scenario
goes from **4, 13, 14, 15, 34, 40, 45, 47, 48, 49, and then 55.**



Example C++ Code:

```
//Given struct Node

struct Node
{
   int data;
   Node* left;
   Node* right;
};


//Inorder Traversal

void inOrder(struct Node* node)

{

     if (node == NULL)

          return;



 inOrder(node->left);
 cout<<"  "<<data<<"  "<<endl;
 inOrder(node->right);

}
```

Post-order Traversal:
In post-order traversal, it is used to delete the tree one by one. Although the concept of post-order traversal is similar to pre-order traversal in
terms of direction, it starts from the bottom to the top of the binary tree in a left to right scenario. Thus, the order in this case is
13, 15, 14, 34, 4, 47, 49, 48, 55, 45, and then 40.



Example C++ Code:

```
//Given struct Node

struct Node
{
   int data;
   Node* left;
   Node* right;
};


//Post-order Traversal

void postOrder(struct Node* node)

{

      if (node == NULL)

            return;
```

```
 postOrder(node->left);
 postOrder(node->right);
 cout<<" "<<data<<" "<<endl;


}
```

Conclusion:

Pre-order Traversal: 40, 4, 34, 14, 13, 15, 45, 55, 48, 47, 49

In-order Traversal: 4, 13, 14, 15, 34, 40, 45, 47, 48, 49, 55.

Post-order Traversal: 13, 15, 14, 34, 4, 47, 49, 48, 55, 45, 40