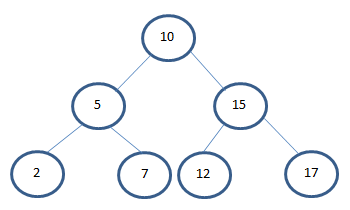
Binary Search Tree PreOrder traversal using recursion.

In previous blogs we studied what is tree and how to insert data into the tree.

The most important thing to solve problems on tree is to learn tree traversals. To name a few InOrder, PreOrder, PostOrder, Bread-First-Search.

In this post we will study recursive **PreOrder** traversal. Consider the following tree.



3 step description for PreOrder traversal

Display the current element.

Traverse left subtree by calling function preOrder recursively.

Traverse the right subtree by calling preOrder function recursively.

/\*\*

\* Pre Order traversal Recursion

\*

\* **@param** localRoot the local root

\*/

**public** **void** preOrder(Node localRoot) {

/\*\*

\* Base condition to end recursion.

\* \*/

**if** (localRoot != **null**) {

//Step 1: Print the current node.

localRoot.displayNode();

//Step 2: Recursively traverse left subtree.

preOrder(localRoot.leftChild);

//Step 3: Recursively traverse right subtree.

preOrder(localRoot.rightChild);

}

}

Output:

{ 10 } { 5 } { 2 } { 7 } { 15 } { 12 } { 17 }

In next post we will see how to traverse the BST (Binary Search Tree) using PreOrder without recursion.

Click here to see next post.