Size of BST or Total nodes in BST

In previous post we saw how to insert nodes in BST and traverse the BST. In this post we will see how to get the number of nodes or size of BST.

For below tree the total nodes are 7. Or the size of tree is 7.

We can get size in 2 ways.

First, use recursion. Call size(node) for left sub-tree and call size(node) for right subtree. Add 1 for root node.

It looks something like this.

size(node.leftChild) + 1 + size(node.rightChild)

Code for recursion is as follows:

**public** **int** size(Node node) {

/\*\*

\* If node is null then return 0.

\* \*/

**if** (node == **null**) {

**return** 0;

}

/\*\*

\* size(node.leftChild) will calculate size of left subtree

\* size(node.rightChild) will calculate size of right subtree.

\* \*/

**return** (size(node.leftChild) + 1 + size(node.rightChild));

}

Second method is we can use any traversal technique and count all the traversed nodes. Let us use InOrder Traversal. Click [here](http://data-structure-learning.blogspot.com/2015/05/binary-search-tree-inorder-traversal.html) to learn InOrder traversal.

What we will do is whenever we pop the element from stack we will increment the count. Reason is when we pop the element means that element is now processed.

Below is the code for counting total number of nodes in tree.

**public** **int** countNodes(Node root) {

//Define a Stack to store nodes into it.

Stack<Node> s = **new** Stack<Node>();

//Assign root to temp. Do not move root.

Node temp = root;

**int** count=0;

/\*\*

\* Here we check

\* 1. Whether stack is empty or not

\* 2. temp is not null

\* \*/

**while** (!s.isEmpty() || temp != **null**) {

/\*\*

\* temp is not null

\* 1. Push temp in stack

\* 2. traverse left child

\*

\* This step is similar to Step 1 in InOrder Recursion.

\* \*/

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

s.push(temp);

temp = temp.leftChild;

}

/\*\*

\* temp is null

\* 1. Push temp in stack

\* 2. traverse right child

\*

\* This step is similar to Step 2 & 3 in InOrder Recursion.

\* \*/

**else** {

temp = s.pop();

count++;

//System.out.print(temp.iData+" ");

temp = temp.rightChild;

}

}

**return** count;

}