Binary Search Tree

- Describe efficient recursive algorithms that can be used to compute the size as well as the height of a binary search tree that is given as input. Your algorithms should each use a number of steps that is **at most linear** in the size of the given binary search tree.

Height

```
int findHeight(TreeNode<T> aNode)
{
    if(aNode == 0)
        return -1;

    int lefth = findHeight(aNode.left);
    int righth = findHeight(aNode.right);

    if(lefth > righth)
        return lefth + 1;
    else
        return righth +1
}
```

Size

```
size(tree)
1. If tree is empty then return 0
2. Else
     (a) Get the size of left subtree recursively  i.e., call
          size( tree->left-subtree)
     (a) Get the size of right subtree recursively  i.e., call
          size( tree->right-subtree)
     (c) Calculate size of the tree as following:
            tree_size  =  size(left-subtree) + size(right-
                               subtree) + 1
     (d) Return tree_size
```

Proof ?

Write a recurrence

- Steps(T) = Steps(T/2) + c + Steps(T/2)