10. Assume that each node in an AVL tree has the data member *lsize*. For any node, *a*, *a*→*lsize* is the number of nodes in its left subtree plus one. Write a C++ function Avl::*Find(k)* to locate the *k*th smallest key in the tree. Show that this can be done in O(log *n*) time if there are *n* nodes in the tree.

```
node* Find(node* n, int k){
    if(k-1 == n->lsize)
        return n
    else if(k-1 < n->lsize)
        if(n->lchild == NULL)
            return n
        else
            return Find(n->lchild, k)
    else
        if(n->rchild == NULL)
            return n
        else
            return Find(n->rchild, k-(n->lsize))
}
```

In AVL tree, after doing operation(insert, delete), the Left and Right will be adjusted balanced. Therefore, there will not be an extreme situation and since AVL tree is the best case in binary search tree, it only takes O(log n) to solve the problem.