Solution for first Problem:

class Node

{

int value;

Node \*left;

Node \*right;

}

int FindSmallestValueLargerThanOrEqualToKey(Node root, int key)

{

if(root == NULL) {

return -1;

}

else if(root->value==key) {

return key;

}

else if(root->value<key) {

return FindSmallestValueLargerThanOrEqualToKey(node->right, key);

}

else {

int i = FindSmallestValueLargerThanOrEqualToKey(node->left, key);

if(i <= -1) {

return node.value;

} else {

return i;

}

}

}

Solution for 2nd Problem:

class Node

{

int value;

Node left;

Node right;

}

public class Arr{

public ArrayList<Integer> values(Node root,int ele)

{

ArrayList<Integer> numbers= new ArrayList<Integer>();

while(root!=NULL)

{

if(root.value==ele)

{

numbers.add(root.value);

return numbers;

}

else if(root.value>ele)

{

numbers.add(root.value);

root=root.left;

}

else

{

numbers.add(root.value);

root=root.right;

}

}

}

}

Node FindLowestCommonParent(Node root, int a, int b)

{

int e1,e2,temp;

Arr arr= new Arr();

ArrayList<Integer> list1= new ArrayList<Integer>();

ArrayList<Integer> list2= new ArrayList<Integer>();

list1=arr.values(root,a);

list2=arr.values(root,b);

int n1=list1.size();

int n2=list2.size();

for(;n1!=0;n1--)

{

e1=list1.get(n1-1);

temp=n2;

for(;temp!=0;temp--)

{

e2=list2.get(n2-1)

if(e1==e2)

{

System.out.println("Common Parent for both is "+e1);

break;

}

}

}

root =GetNode(root,e1);

return root;

}

Node GetNode(Node root, int key)

{

while(root!=NULL)

{

if(root.value==ele)

{

return root;

}

else if(root.value>ele)

{

root=root.left;

}

else

{

root=root.right;

}

}

}