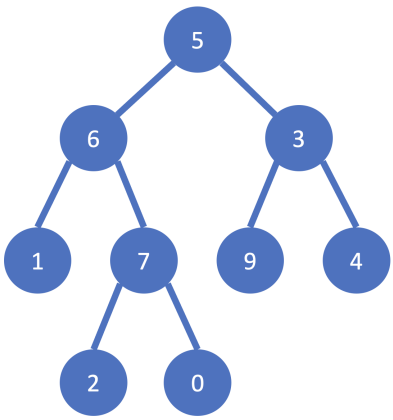
Lowest common ancestor of a BST P52853_en

The lowest common ancestor (LCA) of two nodes x and y in a tree is the lowest (i.e. deepest) node that has both x and y as descendants, where we define each node to be a descendant of itself.

For instance, in the following tree, 5 is the LCA of 1 and 9, and 6 is the LCA of 1 and 0:



Write a function *Tree lowest_common_ancestor_bst* (*Tree* t, **int** x, **int** y); that returns the node that corresponds to the LCA of x and y in a binary search tree of integers t. You can assume that t is a binary search tree and that t contains both x and y. Note that an efficient solution is expected, exploiting the fact that the tree is a binary search tree.

Most of the program is already writen for you. Download it! It reads several binary search trees in preorder (empty trees are marked with a -1 value) and, for each of these, reads severals pairs of values and prints their LCA. You just have to specify and implement the *lowest_common_ancestor_bst* () function (and other helper functions, should you need them). Also, write a comment with the time efficiency of your algorithm.

Sample input

```
2

7 2 1 -1 -1 -1 9 8 -1 -1 15 -1 -1

    1 15
    15 1
    8 15
    15 8
    2 15
    9 9
    -1 -1

10 -1 20 -1 30 -1 40 -1 50 45 -1 -1 60 -1 -1

    10 20
    10 60
```

```
20 60
45 60
-1 -1
```

Sample output

```
7
7
9
9
7
9

10
10
20
50
```

Problem information

Author : Jordi Petit
Generation : 2025-05-13 11:24:08