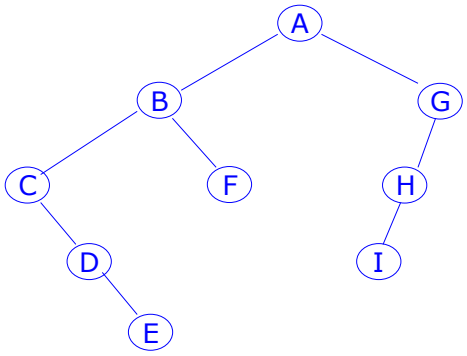
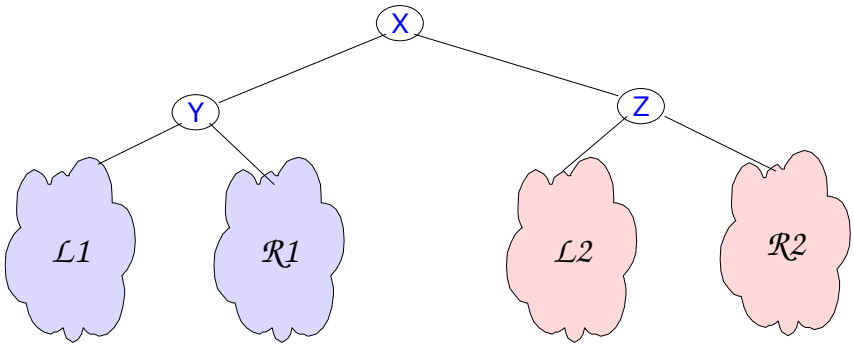
Solutions to Exercises on Trees

1. Pre-order traversal of a binary tree gives A B C D E F G H I and the in-order traversal gives C D E B F A I H G. Draw the tree.



2. In a Binary Search Tree (BST), show that if a node has two children, then its successor has no left child and its predecessor has no right child.

Suppose that node x has two children y and z.



Case 1. Predecessor

The predecessor of x is the maximum element, say M, in the subtree $\mathcal{R}1$. If M has a right child, then that child will be greater than M, which violates the stipulation that M is the largest in $\mathcal{R}1$, showing that M can not have a right child. Note that if $\mathcal{R}1$ is empty, then y is the predecessor, and we have nothing to prove.

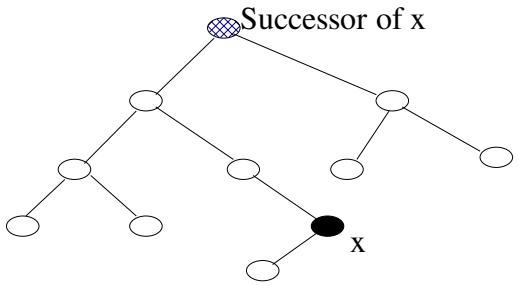
Case 2. Successor

The successor of x is the minimum element, say m, in the subtree $\mathcal{L}2$. (If $\mathcal{L}2$ is empty, z is the successor, and we have nothing to prove.) If m has a left child, then that child will be less than m, violating the stipulation that m is the smallest in $\mathcal{L}2$. This proves that m can not have a left child.

3. Suppose we have numbers from 1 to 1000 in a BST and want to search for 363. Which of the following sequences, if any, could NOT be the sequence of nodes examined?
a. 2, 252, 401, 398,330,344,397,363
b. 925, 202, 911,240,912,245,363
c. 935,278,347,621,299,392,358,363

a. is OK, b and c are not.

4. Prove that if the right subtree of a node x in a BST is empty and x has a successor y, then y is the lowest ancestor of x whose left child is also an ancestor of x.



5. Let B represent a BST. Write an algorithm Tree_MAX(x) that returns the node with maximum value.
Done Elsewhere
6. Let B represent a BST. Write an algorithm Tree_MIN(x) that returns the node with minimum value.
Done Elsewhere
7. Let B represent a BST and x a given node in B. Write an algorithm to find the successor (node following x in an in-order traversal) of x. (See question 4.)

Pseudo-code for finding successor of a node x:

```
successor(x)
  if right[x] != NULL
      then return Minimum(right[x])
  else
      y=parent[x]
      while ( (y != NULL) and x== right[y] )
        {
          x=y
          y=parent[y]
        }
        return y
```

8. Let B represent a BST and x a given node in B. Write an algorithm to find the predecessor of x.

This is symmetric to finding successor