Tuesday, August 18, 2020 9:06 PM

ALGORITHM

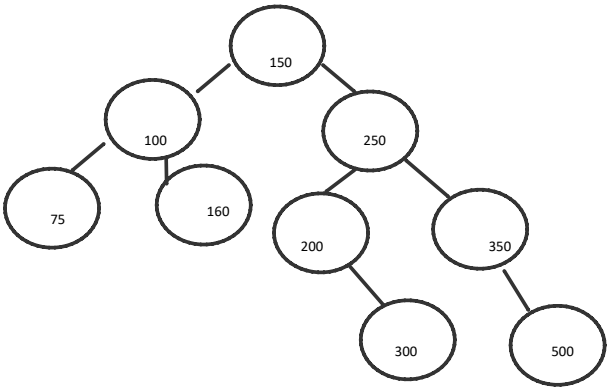
PROBLEM DOMAIN

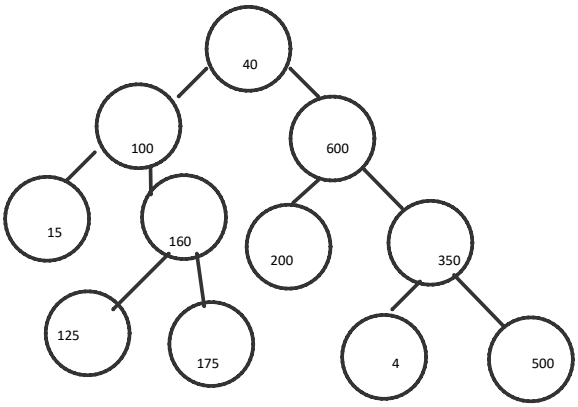
Write a function called tree_intersection that takes two binary tree parameters.
Without utilizing any of the built-in library methods available to your language, return a set of values found in both trees.

EDGE CASES
Null values
Non-repeat numbers

Create a Method Tree Insertion
Input <-- root1, root 2
push the Nodes of first tree in stack s1
push the Nodes of second tree in stack s2
Both root1 and root2 are NULL here
IF current keys in two trees are same
Move to the Inorder next
IF Node of first tree is smaller
Inorder and next have same value
Pop from s2
Root2 is set to NULL
Both roots and both stacks are empty

VISUALS





Output [100,160,125,175,200,350,500]

PSEUDO CODE

```
public static void TreeIntersection(Node<T> root1, Node<T> root2)
    {
      Stack<Node<T>> s1 = new Stack<Node<T>>();
      Stack<Node<T>> s2 = new Stack<Node<T>>();

      while (true)
      {
        if (root1 != null)
        {
       s1.Push(root1);
          root1 = root1.left;
        }


        else if (root2 != null)
        {
          s2.Push(root2);
          root2 = root2.left;
        }

        else if (s1.Count > 0 && s2.Count > 0)
        {
          root1 = s1.Peek();
          root2 = s2.Peek();

          if (root1.key == root2.key)
          {
            s1.Pop();
            s2.Pop();

            root1 = root1.right;
            root2 = root2.right;
          }

          else if (root1.key < root2.key)
          {
            s1.Pop();
            root1 = root1.right;

            root2 = null;
          }
          else if (root1.key > root2.key)
          {
            s2.Pop();
            root2 = root2.right;
            root1 = null;
          }
        }


        else
        {
          break;
        }
      }
    }
```