Algorithm for Binary Search Tree **Remove** Method

```
public void remove(Comparable val) {
   root = remove(val, root);
}

private TreeNode remove(Comparable val, TreeNode tree) {

   if there's no node:
      return null;

   decide whether val is bigger or smaller than the current value

   if val is smaller:
      call remove() on the left child and make the result the new left child
   otherwise, if val is bigger:
      call remove() on the right child and make the result the new right child
   otherwise:
      if there is no right child:
         set tree to the left child
   otherwise:
      find the successor to this element, i.e. the smallest element in the right subtree
      set this node's value to that of the successor
      remove the successor from the right subtree and make the result the new right child

   return tree;

}
```