1 对 remove 函数的阐述和分析

首先，讨论节点 t 为空的情况，此时没有节点可以删除所以直接返回。然后，如果 x 小于当前节点的值，那么递归地在 t 的左子树中删除 x。如果 x 大于当前节点的元素，则在右子树中删除 x。然后我进行分情况讨论：如果 t 的左右子树都不为空，那么通过 detachMin 函数找到 t 的右子树中的最小节点 minNode，将 minNode 左子树指向当前节点的左子树，右子树指向当前节点的右子树，删除当前节点 t，同时将 t 指向 minNode。如果其中一个子树为空，那么通上课代码的思路，保存 t 到 oldNode，将 t 指向非空的子树，删除 oldNode。

关于对程序结果的测试，我创建一个二叉搜索树，插入 5,10,15,20,25,30,35，并且打印出树的结构。然后我删除树最左侧的节点 5，再次打印树的结构，发现 5 被成功删除。同理我删除有一个子节点的节点 30，再次打印树的结构，发现 30 被成功删除。然后我删除有两个子节点的节点 10，再次打印树的结构，发现 10 被成功删除。最后我删除根节点 20，再次打印树的结构，发现仍成功。

2 测试的结果

测试结果一切正常。详细的结果见上文的测试程序的设计思路部分。