## **Binary Search Tree: Remove: The Algorithm**

**The goal for this exercise** is to make sure that you understand how the BST remove algorithm works, before you actually code up the remove algorithm in source code.

**What you need to do for this exercise:**

1. For each of the following cases, show what the resulting binary search tree should look like, after you add values in the specified order, and then remove the given value.  
     
   Note: For consistency (so that you can compare your answer to other students' answers easily), you should always pick the predecessor value to remove. Yes, this will work equally correctly by picking the successor value, but for this exercise, use the predecessor value.  
     
   Please draw out the trees on a sheet of paper.

|  |  |
| --- | --- |
| 1. **Insert:** 10, 4, 6, 8, 2, 1 **Remove:** 1 | 1. **Insert:** 10, 4, 6, 8, 2, 1 **Remove:** 8 |
| 1. **Insert:** 10, 4, 6, 8, 2, 1 **Remove:** 6 | 1. **Insert:** 10, 4, 6, 8, 2, 1 **Remove:** 2 |
| 1. **Insert:** 10, 4, 6, 8, 2, 1 **Remove:** 4 | 1. **Insert:** 10, 4, 6, 8, 2, 1 **Remove:** 40 |
| 1. **Insert:** 10, 4, 6, 8, 2, 1 **Remove:** 10 | 1. **Insert:** 10, 4, 6, 8, 2, 1, 3 **Remove:** 4 |
| 1. **Insert:** 10, 4, 6, 8, 2, 1, 3, 2.5 **Remove:** 4 |  |