CMPUT 275 — Tangible Computing II **Winter 2018**

Binary Search Trees

Recall the simple insertion method into a binary search tree that inserts the key k into the tree:

1. If the tree is empty, create the root and set its key to be equal to k
2. Otherwise, find the insertion point by following the path from the root to a leaf where at each node n, the left child is taken if $k < n.key$, otherwise the right child is taken. The insertion point is the last node n which doesn't have the appropriate child.
3. Insert a new node as a left child of the insertion point n if $k < n.key$, otherwise as a right child.

Recall the simple method to remove key k:

1. Find the node n that holds the key k.
2. If there is no left child of n, replace n with its right child (may be nothing).
3. Otherwise, find the node m with the maximum key in the subtree rooted at $m.left$. Move $m.key$ to $n.key$. Then remove m and replace it with its left child (may be nothing).

Problems

1. Draw the binary search tree resulting from inserting the keys $49, 79, 44, 41, 64, 80, 48$ into the empty tree in this order with the simple insertion method. Let T be the resulting tree.

   The items being stored with the keys are not relevant for this exercise, just show the keys.

2. Give the range of numbers that can could insert T as a left child of the node whose key is 48:

3. Give the range of numbers that can be inserted into T as a right child of the node whose key is 64:

4. Draw the sequence of trees obtained by removing the keys in this order: $44, 46, 49, 79, 41, 80, 64$. Use the removal method described above.

5. What order could you insert keys $1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12$ to get a BST with the following shape?