Subject: Can Prune Delete a Node When It's The Only One, or must v/rbst delete function do that?
Posted by davidmccoy on Mon, 27 Feb 2017 23:53:52 GMT

EDIT: I have a new question: is prune unable to cut off a node when its the only one? For the delete functions in vbst and rbst, you can set tree->tree->root to null, and it's not a problem; but in the prune operation alone, all you are passed in is a bst *node, so you can prune it off from its parent by setting it's parent left or right child to node, but you can't actually set the node itself to be "null" to make the tree empty (writing node = NULL simply changes the pointer you're using, but tree->tree->root will still point to the node you have).

Original post:

In testing my prune method for generic bst, I have come across this issue: it won't prune the root when it's the only node left. In my prune method there are three conditions, the first of which is that if a node is its own parent and has no children, then set that node pointer to null. But that apparently doesn't successfully get rid of the node; when display is run, the root still successfully points to the node I tried to prune! Any help?

Tree:
0: 3
1: 2 4
2: 1 5
Delete a value: 3
Tree:
0: 4
1: 2 5
2: 1
Delete a value: 2
Tree:
0: 4
1: 1 5
...deleting rest of nodes...

Tree:
0: 4
Delete a value: 4
Tree:
0: 4
Delete a value: 4
Tree:
0: 4
The deletion method should detect this case (note that bst does not have a deletion method).

the deletion fix up routine should start at the node that it is value swapped with the leaf right?, also on recognition to the the node to be prune, is the easiest way is to search for that node by value?.

Yes, the leaf to be pruned is sent to the deletion fixup routine (rbt specific).

To delete, find the node holding the value first, then send it to the swap-to-leaf routine.

I thought this should be

since it prefers to swap with a predecessor.
That is correct, this was a old post. Check out my latest testing files at
https://github.com/damccoy1/assign2testing.