CSCI 2270 Lecture Notes

2/25/19

<https://visualgo.net/bn/bst>

BST Continued:

* delete node
  + remove node in a way that preserves that validity of the BST (99% of the work)
  + Deallocate the node’s memory (super simple)
* High level Algorithm for delete
  + find the node (need search(value))
  + remove node from tree
    - based on where the node is in the tree, different actions need to be taken
    - root?
    - number of children it has? (0,1,2)
    - relationship to parent? (left child, right child, neither (root))
* Possible children scenarios of node to be deleted
  + No children (leaf)
    - most trivial case
    - delete the node, tell parent that child is now null
    - does not affect the validity of the bst
  + 2 children
    - Find the min node of right subtree
    - replace n (node being deleted) with min-node found before
    - if min is the right child, just “slide” that node and its subtree into its place
    - if min is somewhere else in the subtree, tree needs re-arranging
  + 1 child
    - replace n with root of the subtree
    - left or right child, no need to find min
* Additional consideration
  + is node a left or right child, is it root?

Search Function:

* For balanced trees, the search function has a time complexity of O(log(n))
* public function: search(int value)
  + this calls recursive function with root and value which is private
* private function: searchRecursive(node, int sKey)
  + check if key is found
    - return node
  + if sKey < n->key
    - go left
    - else, go right