CS 209
Data Structures and Mathematical Foundations

03 / 18 / 2024

Instructor: Michael Eckmann

Today's Topics

- Questions/Comments?
- Binary Search Trees
  - Finish implementing remove
- Recursion

Remove in binary search trees

- Given a piece of data to remove, we first search the tree to find the node containing that data
- Then we must figure out how to get that node out of the tree and still have a BST
- We should consider a replacement of the removed node with some other node in the tree
- Let's see some examples and talk it through and come up with a plan

Binary Search Trees

- Let’s write remove.
- If a leaf --- remove it easily (set it’s parent’s child to None)
- If it has only 1 child, (set its parent’s child to its only child)
- If it has 2 children, then we do:
  - Either replace the node to be removed’s data with that of the RMN in LST’s data OR LMN in RST’s data. (that is, either with the largest in its left subtree or with the smallest in its right subtree).
  - We settled on LMN in RST. We’ll then remove this LMN in RST which will be easier to remove than the one that had two children, as it has at most one child (a right child).
  - It was noted that the RMN in LST wouldn’t work if it was a duplicate.

Recursion

- 1. have at least one base case that is not recursive
- 2. recursive case(s) must progress towards the base case
- 3. trust that your recursive call does what it says it will do (without having to unravel all the recursion in your head.)
- 4. try not to do redundant work. That is, in different recursive calls, don't recalculate the same info.

Recursion

- some definitions are inherently recursive
- e.g.
- sum of the first n integers, n>= 1.
- S(1) = 1
- S(N) = S(N-1) + N

- recursive code for this
- iterative code for this
- simpler code for this ->   N*(N+1)/2
- any problems with the recursive code? n=0 or large n?

Recursion

- The fibonacci sequence is:
  0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, ...

  - Can you detect the pattern?