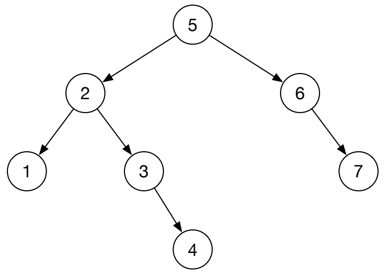
CS241

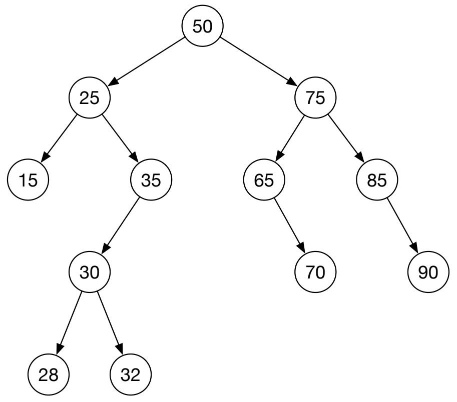
Homework 3: Binary trees and binary search trees

1. Assume you have the following binary search tree:



(a) List the order of the nodes for a preorder traversal.
(b) List the order of the nodes for an inorder traversal.
(c) List the order of the nodes for a postorder traversal.

2. Assume you are given the following binary search tree:



Show what the tree looks like after each of the following sequences of operations (draw the tree only after the last operation in each sequence).
RESET THE TREE BACK TO THE DRAWING ABOVE AT THE BEGINNING OF EACH SEQUENCE.
For deleting a node with two children, replace with the *inorder successor*.

(a) Insert 11, Insert 64, Insert 31, Insert 80, Insert 38, Insert 37, Insert 39
(b) Delete 35
(c) Delete 25
(d) Delete 50

3. Suppose a friend tells you they have a binary search tree with the following preorder traversal: 14, 7, 3, 10, 20, 16, 17, 22. Reconstruct the entire BST from this traversal and draw it. Hint: given a preorder traversal, how do you identify the root?

4. Suppose you have a *sorted* array A with n integers in it, indexed from 0 to n-1.

(a) Describe an algorithm that inserts the elements of array A into an empty binary search tree that is guaranteed to result in the tree being as "deep" as possible. (In other words, describe an algorithm that inserts the elements from A in such an order as to maximize the longest possible path from the root to any leaf node.) Note that this tree will be very unbalanced!

(b) Describe an algorithm that inserts the elements of A into an empty binary tree that is guaranteed to result in the tree being as "shallow" or "bushy" as possible. That means that the longest path from the root to any leaf node should be minimized. This tree will be very balanced!

Hint for (b): You may assume that the number of elements in array A is exactly a power of 2 minus 1 if that makes your life easier.

Hint for (a) and (b): These algorithms are not complicated, and do not have to be written formally. Just describe an ordering of the elements in array A that makes the tree as unbalanced as possible (part (a)) and as balanced as possible (part (b)).

5. This question involves doing "real" coding, but we're going to do it through a website called repl.it that will auto-grade your code so you get immediate feedback.

   Instructions:

   - Go to https://repl.it/classroom/invite/7uKf7p9
   - Sign up for an account (feel free to use the links to automatically sign in using google, facebook, or github)
   - Once you are logged in, you should see two problems, "Snippet A" and "Snippet B." I suggest doing Snippet A first, then B. Both A & B are part of this homework.
   - To use the repl.it environment, you write code in the left half of the window. Whenever you want to run your code, click the "Run" button at the top. The black area near the bottom of the screen is the console where you can type input in, and you'll see your program's output.
   - AUTO GRADING: Your snippets are automatically graded. I have written test cases for your programs that will automatically send input into your program and examine the output to make sure it matches mine. You can run the autograder as many times as you want before you finally submit, and it will give you feedback. To run the autograder, from the Run button near the top, drop down the arrow to the right and click "Run Tests."
   - When you finally are done (I suggest making sure you pass all the tests), click the green Submit button at the top right.

   Let me know about if you like/hate repl.it. I'm using this because people wanted more coding practice and this makes it easier to grade them.