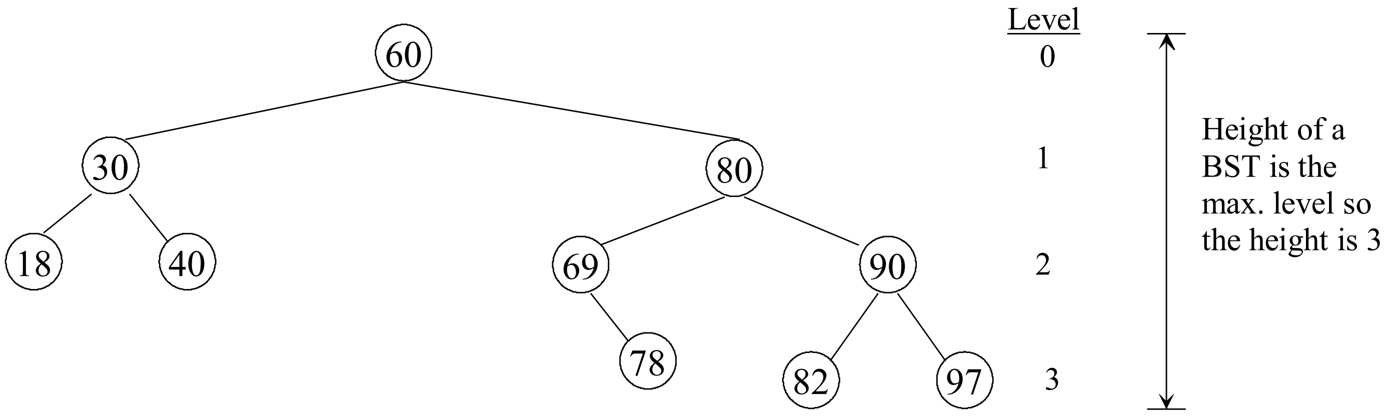
Data Structures (CS 1520) Lab 9 Name:___________________________

Objectives: You will gain experience BST performance and implementation

To start the lab: Download and unzip the lab9.zip file from eLearning.

Part A: Consider the Binary Search Tree (BST) below. For each node in a BST, all values in the left-subtree are < the node and all values in the right-subtree are > the node.



a) Review section 6.5.2 on Tree Traversals to determine the order nodes are processed in each tree traversal.

- What is the order of node processing in a preorder traveral of the above BST?
- What is the order of node processing in a inorder traveral of the above BST?
- What is the order of node processing in a postorder traveral of the above BST?

b) Starting with an empty BST, what would be the shape of the BST after `put`'s for keys:
50, 60, 30, 70, 90, 40, and 65?

Part B: Run the `timeBinarySearchTree.py` program that:

- creates a list, evenList, that holds 2,000 sorted, even values (e.g., evenList = [0, 2, 4, 6, 8, ..., 3996, 3998])
- puts (adds) all the evenList items into an initially empty BinarySearchTree object, bst
- times the searches (in) bst for target values 0, 1, 2, 3, 4, ..., 3998, 3999 so half of the searches are successful and half are unsuccessful

a) How long does it take to search for target values of 0, 1, 2, 3, 4, ..., 3998, 3999?

b) Explain why these searches take so long. (Hint: consider the shape of the BinarySearchTree `bst`)

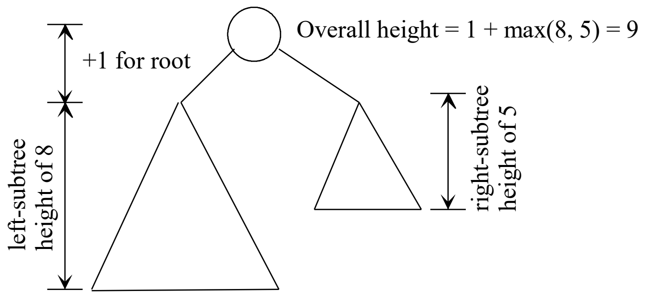
c) Uncomment the "`shuffle(evenList)`" which randomizes the items in evenList before adding them to the BinarySearchTree `bst`. Now how long does it take to search for target values from 0, 1, 2, 3, 4, ..., 3998, 3999?

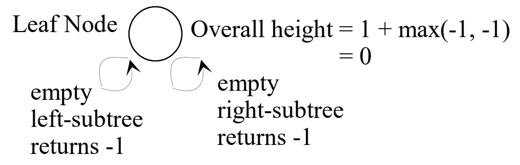
d) Explain why these searches take so little time.

e) What is the search time with the `timeOpenAddrHashDictSearch.py` program? Why is it faster?

Part C: a) **Complete the recursive `height` method in the BinarySearchTree class.** Model it after the postorder traversal, since the height of the whole BST can be determined after you know the height of the left-subtree and height of the right-subtree. For example if the left-subtree has a height of say 8 and the right-subtree has a height of 5, then the overall height including the root is 9 (i.e., one more than the tallest subtree's height). For the base case of the recursion, if we define the empty subtree's height to be -1 (i.e., `subtreeRoot` points to `None` since it has no `TreeNode` to point at), then the recursive definition still works for a leaf node which should have a height of 0.

b) Uncomment the call to the `height` method at the end of the `timeBinarySearchTree.py` program. What is the height of `bst` if we are shuffling the `evenList`?




Leaf Node
Overall height = 1 + max(-1, -1) = 0
empty left-subtree returns -1
empty right-subtree returns -1

c) What would be the shortest possible height for a binary tree with 2,000 items?

After you have answers and correct code for all parts of the lab, submit a lab9.zip containing your code on eLearning. If you do not get done today, then submit it by next week's lab period.

Remember to save your lab9 files for later usage on homework assignments!