

CST 370 – Spring A 2020
Homework 5

Name: Sara Kazemi

Class ID: 4444

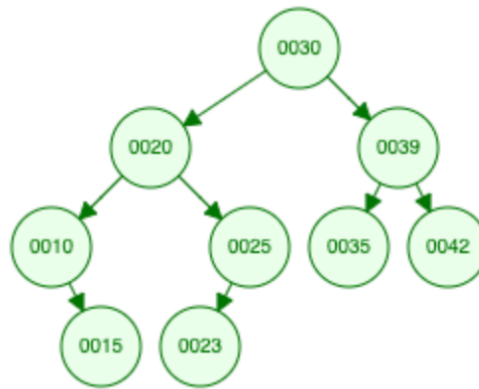
1. (5 points) The preorder traversal sequence of a binary search tree is 30, 20, 10, 15, 25, 23, 39, 35, 42. Which one of the following is the postorder traversal sequence of the same tree? Explain your answer.

- a) 10, 20, 15, 23, 25, 35, 42, 39, 30
- b) 15, 10, 25, 23, 20, 42, 35, 39, 30
- c) 15, 20, 10, 23, 25, 42, 35, 39, 30
- d) 15, 10, 23, 25, 20, 35, 42, 39, 30

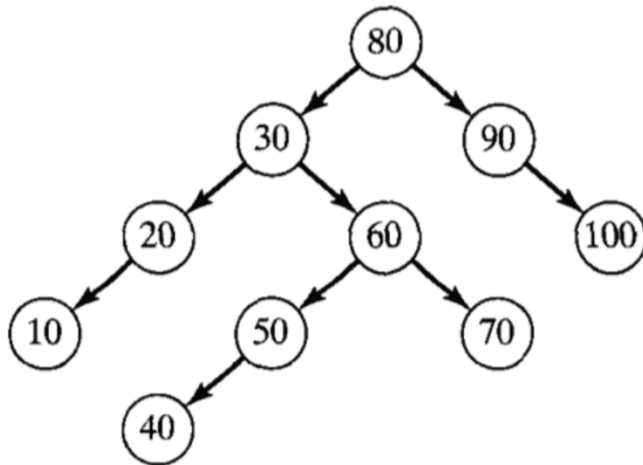
The postorder traversal sequence of the tree is:

d) 15, 10, 23, 25, 20, 35, 42, 39, 30

This is because post order first traverses left subtrees, then right subtrees, before finally reaching the root. First we traverse the left subtree rooted at node 20. We start at the descendant node 15 since it has no left sibling. Then we go up to node 10. Node 10's right sibling roots right subtree so we visit its left descendant, node 23. Then we go up to node 25. Having explored all child nodes in the subtree rooted at 20, we go up to node 20. Now we traverse the right subtree rooted at 39. We start at the left child node 35, go on to the right child 42, then go up to their parent node 39. Finally, we go to the root of the main tree, 30.

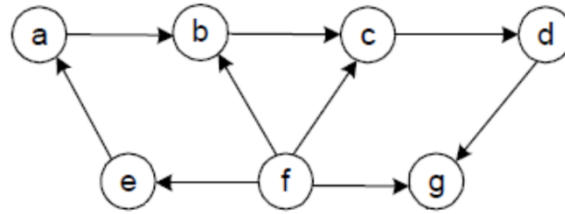


2. (10 points) Consider the following Binary Search Tree (BST):



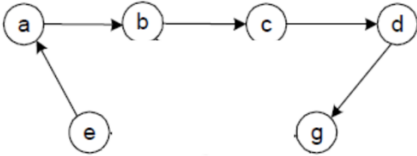
- a. Inorder traversal: 10, 20, 30, 40, 50, 60, 70, 80, 90, 100
- b. Preorder traversal: 80, 30, 20, 10, 60, 50, 40, 70, 90, 100
- c. Postorder traversal: 10, 20, 40, 50, 70, 60, 30, 100, 90, 80

3. (5 points) Apply the source-removal algorithm to solve the topological sorting problem for the following digraph. You should explain your answer and present the topological order clearly.

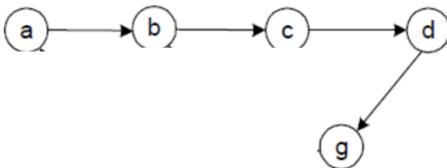


Solution: $f \rightarrow e \rightarrow a \rightarrow b \rightarrow c \rightarrow d \rightarrow g$

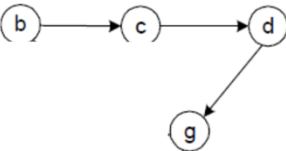
Explanation: We must first identify a source vertex—a vertex that has no incoming edges. The first one we encounter as we traverse this graph is vertex f. **We delete vertex f.**



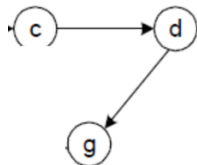
After we delete f, the only source we encounter is e. **We delete vertex e.**



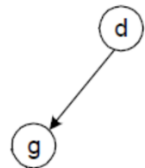
Now the earliest source we encounter is vertex a. **Delete vertex a.**



Next we encounter vertex b as a source. **Delete vertex b.**



Next we encounter vertex c as a source. **Delete vertex c.**



Next we encounter vertex d as a source. **Delete vertex d.**



Last we encounter vertex g as a source. **Delete vertex g.**

4. (10 points) A detachment of n soldiers must cross a wide and deep river with no bridge in sight. They notice two 12-year-old boys playing in a rowboat by the shore. The boat is so tiny that it can only hold two boys or one soldier. Note that the boat needs at least one boy or one soldier to pass from one shore to another.

- a. How can the soldiers get across the river?

Considering the case where there is just $n = 1$ soldiers, the two boys will first have to cross the river together. Then, only one boy will return to the shore where the soldiers remain. This boy will remain on the original shore while a soldier rows to the opposite shore. The boy on the opposite shore will then row the boat back to pick up the boy on the original shore. So, the total number of times the rowboat passes from shore to shore is four times ($4n$).

If there are $n = 2$ soldiers, the process is the same as the above, but after the boy on the opposite shore picks up the boy on the original shore, he must drop him off at the opposite shore again. Then the boy who remains in the boat can row back to the original shore, deboard, and allow the soldier to row across to the opposite shore. The boy on the opposite shore could then row back to the boy on the original shore. Now the total number of times the rowboat passes from one shore to another is eight times (still $4n$).

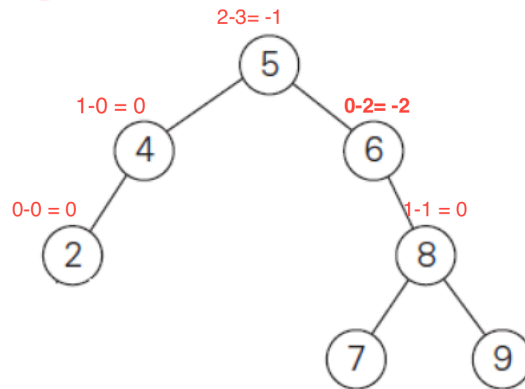
For each increasing number of n soldiers, the process of ensuring that one boy is delivered to the opposite shore before another soldier can cross must be repeated. This can be generalized to $4n$ crossings, where n is the number of soldiers that need to cross.

- b. How many times need the boat cross from shore to shore? **The boat must cross from one shore to another $4n$ times in order to allow n soldiers to cross the river.**

5. (10 points)

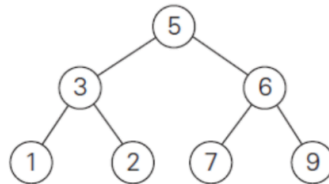
(a) Is this an AVL tree? (Yes/No)

No, because one of the nodes' balance factors is -2, which violates the definition of an AVL tree, which postulates that all nodes have a balance factor of 0, -1, or +1.

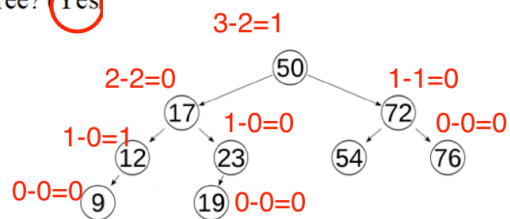


(b) Is this an AVL tree? (Yes/No)


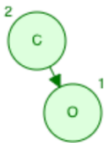
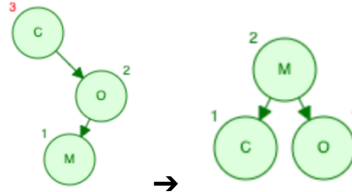
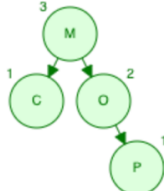
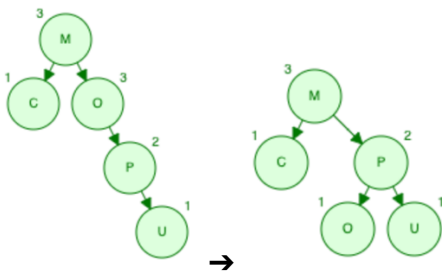
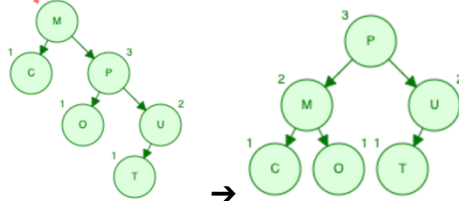
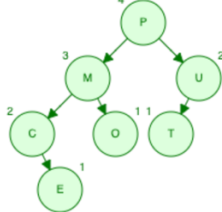
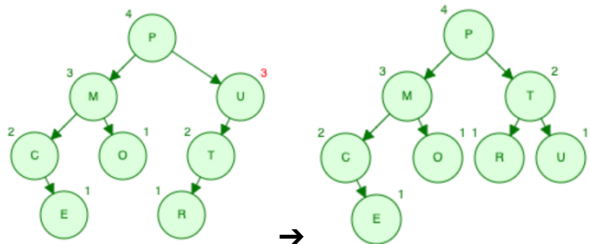
No, because it violates the basic requirements of a binary search tree. Node 7 is a left child of Node 6.



(c) Is this an AVL tree? Yes



6. (10 points) Construct an AVL tree for the list **C, O, M, P, U, T, E, R**. Use the alphabetical ascending order of the letters and insert them successively starting with the empty tree. Your answer should present the rotation operations clearly for each letter addition.

Insertion	AVL Tree	Rotations
Insertion 1: C		None
Insertion 2: O		None
Insertion 3: M		RL-Rotation
Insertion 4: P		None
Insertion 5: U		L-Rotation
Insertion 6: T		L-Rotation
Insertion 7: E		None
Insertion 8: R		R-Rotation

7. (5 points) Suppose you are given a list of N integers. All but one of the integers are sorted in numerical order. Identify a sorting algorithm from class which will sort this special case in $O(N)$ time and explain why this sorting algorithm achieves $O(N)$ runtime in this case.

Insertion sort is efficient for input data that is already mostly sorted since the number of key comparisons ($A[j] > v$ in the algorithm below) depends on the nature of this input. For highly sorted input, Insertion Sort has a best case of $O(n)$ efficiency.

ALGORITHM *InsertionSort*($A[0..n - 1]$)

```
//Sorts a given array by insertion sort
//Input: An array  $A[0..n - 1]$  of  $n$  orderable elements
//Output: Array  $A[0..n - 1]$  sorted in nondecreasing order
for  $i \leftarrow 1$  to  $n - 1$  do
     $v \leftarrow A[i]$ 
     $j \leftarrow i - 1$ 
    while  $j \geq 0$  and  $A[j] > v$  do
         $A[j + 1] \leftarrow A[j]$ 
         $j \leftarrow j - 1$ 
     $A[j + 1] \leftarrow v$ 
```