**66CS 3050 Quiz # 2, Aril 4, 2019**

**Time Limit: 75 Minutes**

**Name : Student ID:**

Note: (1) you can use a letter-sized sheet paper with notes; (2) closed-book quiz, no discussion, no use of cell phone; (3) use additional pages or reverse side of quiz pages if needed.

# Multiple Choices (5 points each, 20 points in total. Circle the correct selection, there is only one correct answer for each problem)

(1) Which of the following statements about binary trees is NOT true?

A. Every binary tree has at least one node.

B. Every non-empty tree has exactly one root node.

C. Every node has at most two children.

D. Every non-root node has exactly one parent.

(2) Which of the following is false about a binary search tree?  
A. The left child is always lesser than its parent  
B. The right child is always greater than its parent  
C. The left and right sub-trees should also be binary search trees  
D. None of the mentioned

(3) What is the special property of red-black trees?

A. A color which is either red or black and root should always be black color only

B. For any node, the height of its left subtree differs from the height of its right subtree by at most 1

C. A red-black trees may not be a binary search tree

D. A color which is either green or black

(4) What are the operations that could be performed in O(*log* *n*) time complexity by red-black tree?

A. Insertion, deletion, finding predecessor, successor

B. Only insertion

C. Only finding predecessor, successor

D. For sorting

# Short Answers (10 points each, 20 points in total)

(1) Here is a small binary tree:

14

/ \

2 11

/ \ / \

1 3 10 30

/ /

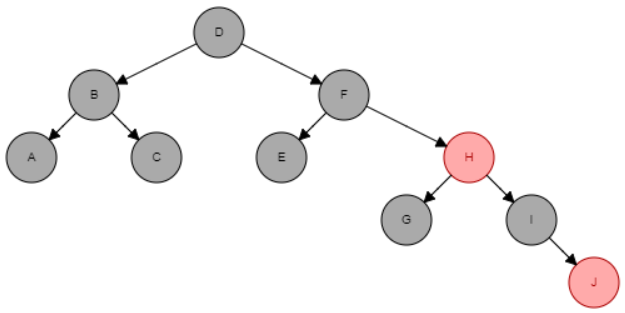
7 40

Write the order of the nodes visited in an in-order traversal (write the numbers in a sequence):

Sol :

1, 2, 3, 14, 7, 10, 11, 40, 30

(3) Design a “worst-case” red-black tree with 10 nodes, i.e., a red-black tree with the longest possible path from the root to a leaf. (You can use double circles to represent a black node, use a single circle to represent a red node. Use letters *A*, *B*, … to name the node)



C. Suppose you are given two Binary Search Trees T1 and T2. Present the psuedo code for an efficient algorithm to determine if they are identical trees. [Hint: Identical trees mean that they contain the same elements and have the same tree structure, e.g. Fig. (A) and Fig. (B) are identical trees]. (20 points)

14

/ \

2 11

/ \

1 3

(A)

14

/ \

2 11

/ \

1 3

(B)

Sol :

Compare (N1, N2)

{

If (both == Nil)

Return true;

Else if (both != Nil)

{

If (N1.value == N2.value)

Compare (N1.left, N2.left)

Compare (N2.left, N1.left)

}

Return false

}

D. Perform Insertion, for each element in Array A, in the Red and Black tree by drawing the tree for each step of insertion.

A = {A, L, G, O, R, I, T, H, M}.

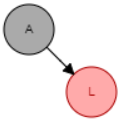
Note: The letters are ordered alphabetically. You can use double circles to indicate black nodes, and single circle to indicates red nodes. For each insertion step, please draw a tree.

(20 points)

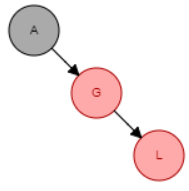
Solution :

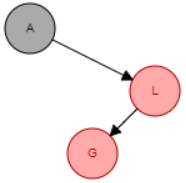


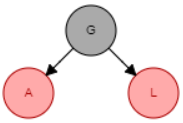
1. A :



1. L :



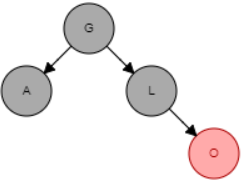




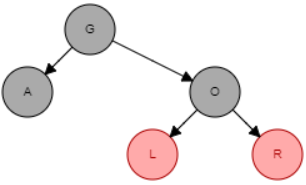
Left-Rotate (A)

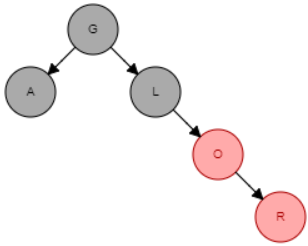
Right-Rotate (L)

3- G :



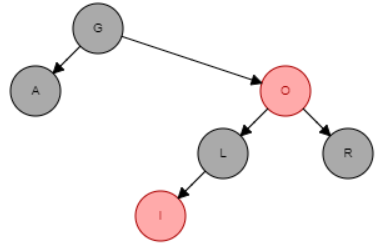
4- O :

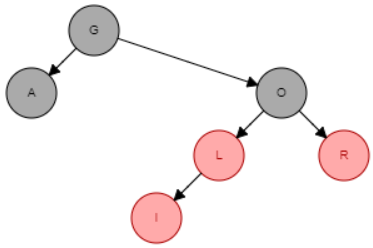




5- R :

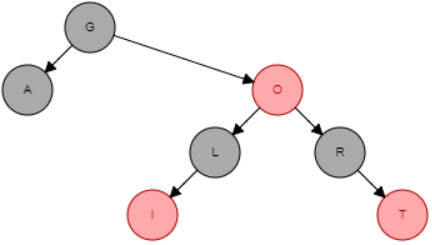
Left-Rotate (L)



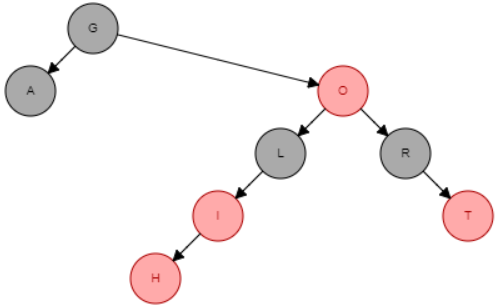


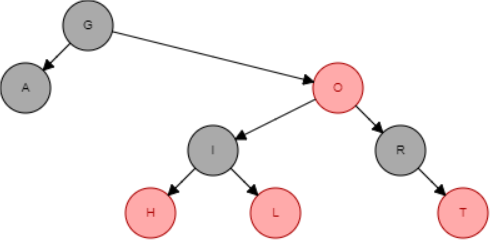
Re-Color

6- I :



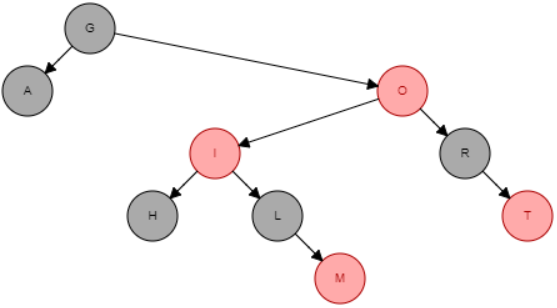
7- T :

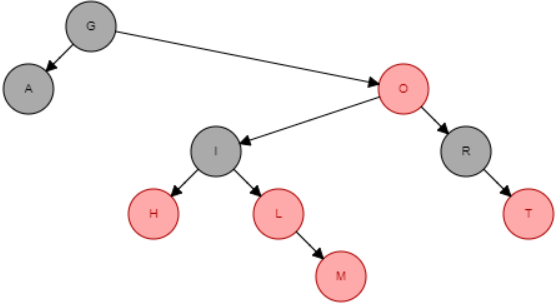




Right-Rotate(L)

8- H :

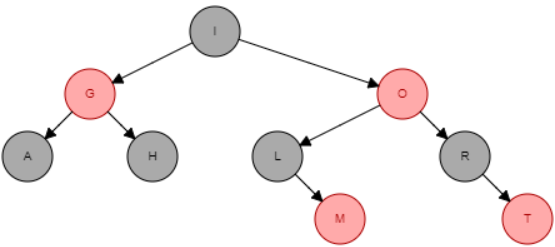


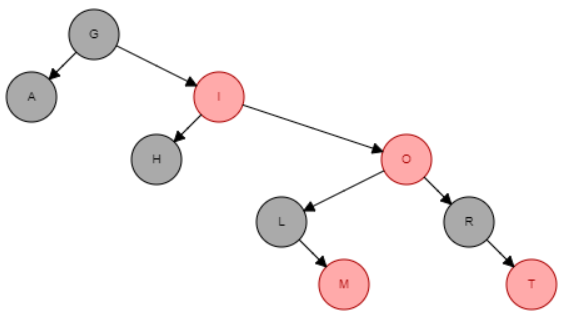


Re-Color (I, H, L)

9- M :

Right-Rotate (O)





Left-Rotate (G)

E. Given an element x in an n-node order-statistic tree and a natural number i, how can we determine the ith **predecessor** of x efficiently? [Hint: write a psuedo code to use a combination of OS-SELECT and OS-RANK]. (20 points)

Sol/

The pseudo code is :

Predecessor(T, x, i)

r = OS-RANK(T, x)

return OS-SELECT(T.root, r-i)