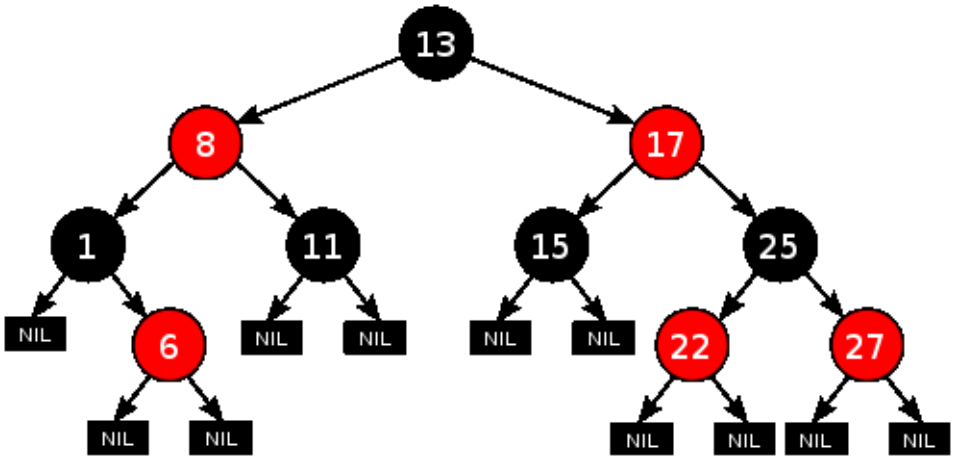
EECS 311: Data Structures and Data Management **Sample Final**
Name: _____ **Fall 2010**

Advice: Skip problems that might take a while and come back to them later. If an algorithm is requested, be as succinct as possible, write your algorithms in pseodocode (unless otherwise specified). You do not need to supply any proofs unless explicitly asked. All runtimes should be given using big-oh notation.

1. A *red-black* tree is a kind binary search tree. The following is an unedited portion of the Wikipedia entry and accompanying figure that defines the structural properties of red-black trees.

   > *A red-black tree is a binary search tree where each node has a color attribute, the value of which is either red or black. In addition to the ordinary requirements imposed on binary search trees, the following additional requirements of any valid red-black tree apply:*
   >
   > - *A node is either red or black.*
   > - *The root is black. (This rule is used in some definitions and not others. Since the root can always be changed from red to black but not necessarily vice-versa this rule has little effect on analysis.)*
   > - *All external nodes are black, even when the parent is black (The external nodes are the null value children.)*
   > - *Both children of every red node are black.*
   > - *Every simple path from a node to a descendant leaf contains the same number of black nodes, either counting or not counting the null black nodes. (Counting or not counting the null black nodes does not affect the structure as long as the choice is used consistently.)*



(Nodes 6, 8, 17, 22, and 27 are red; the remaining are black.)

Prove that any n-node red-black tree has height $O(\log n)$.

2. The following array represents a complete binary tree.

1	2	3	4	5	6
6	3	9	1	4	8

   (a) Draw the tree.

   (b) Is it a heap, a binary search tree, both, or neither?

3. We took advantage of the complete binary tree structure to implement binary heaps in an array. The structure of binomial heaps also permits a natural array based representation. Why would we never want to use an array based binomial heap to implement a binomial queue?

4. The following array represents a union-find data structure

1	2	3	4	5	6	7	8	9
8	6	4	6	6	0	6	4	6

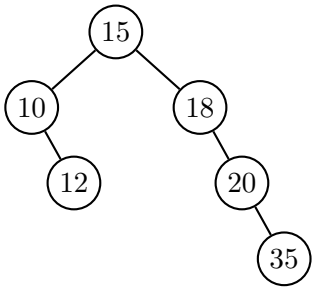
   (a) Draw the forest (or tree) the array represents.

   (b) Could this union-find instance have been created with union-by-rank?

   (c) Could this union-find instance have been created with union-by-size?

5. For the following trees (i) decide whether it is an AVL tree or not, (ii) decide whether it cam be made an AVL tree via rotations, and if so, (iii) describe rotations (e.g., "RotateLeft on node 15") should be performed and give the resulting tree.

   (a)

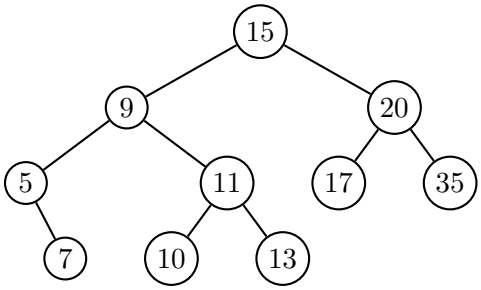


   i. Is it AVL? Yes or No.

   ii. Can it be made AVL by rotations? Yes or No.

   iii. If so, how? Give the resulting tree.

   (b)

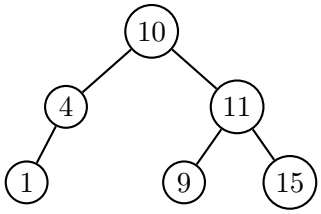


   i. Is it AVL? Yes or No.

   ii. Can it be made AVL by rotations? Yes or No.

   iii. If so, how? Give the resulting tree.

(c)

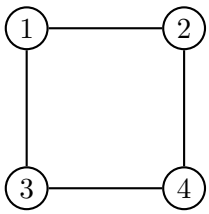


i. Is it AVL? Yes or No.

ii. Can it be made AVL by rotations? Yes or No.

iii. If so, how? Give the resulting tree.

6. Answer the true or false questions by (circling) either "True" or "False". Some questions refer to the graph G below. [Points: _____/10]



(a) True or False: All AVL trees are balanced binary trees.

(b) True or False: All Splay trees are balanced binary trees.

(c) True or False: All Binary heaps are balanced binary trees.

(d) True or False: All Binomial heaps are balanced binary trees.

(e) True or False: All trees maintained in union-find with union-by-height are balanced binary trees.

(f) True or False: All minimum spanning trees are balanced binary trees.

(g) True or False: A BFS in G starting from vertex 1 must visit 2 before 4.

(h) True or False: A DFS in G starting from vertex 1 must visit 2 before 4.

(i) True or False: A BFS in G starting from vertex 1 that visits 2 before 3 must also visit 4 before 3.

(j) True or False: A DFS in G starting from vertex 1 that visits 2 before 3 must also visit 4 before 3.