Instructor: Dr. Sartaj Sahni
Spring, 2003

Advanced Data Structures
(COP 5536 /AD 711R)
Exam 2

CLOSED BOOK
55 Minutes

Name: ________________________________

NOTE:

1. **For all problems, use only the algorithms discussed in class/text.**
2. All answers will be graded on correctness, efficiency, clarity, elegance and other normal criteria that determine quality.
3. The points assigned to each question are provided in parentheses.

1. (12) For the following min Fibonacci heap, assume that the *ChildCut* field of each node is TRUE except node 9. The ChildCut value of the root node is undefined.

```
      min
       |
       V
       1
    /  |   \
   14  7    9 (F)
       |   / \
      11  10 13 (T)
       |      |
      32      15 (T)
             /  \
            17  19(T)
                 |
                21 (T)
```

Figure 1. Fibonacci heap

(a) (5) Perform a *DecreaseKey* operation by changing 21 to 2. Draw the resulting Fibonacci heap, clearly label *ChildCut* values.

(b) (7) Perform a *DeleteMin* operation on the *resulting Fibonacci heap* of (a), clearly label *ChildCut* values. (Show each step)

2. (11) Consider the following *AVL* tree,

```
           35
       /        \
      20           45
   /     \       /     \
  14     30     40      50
 / \    /  \   /  \      \
12  16 26  34 37  43      55
       /          /
      24         41
```

(a) (5) Delete key 55, show each step and specify each rotation type.

(b) (6) Start with the original AVL tree(i.e., the tree before the deletion of key 55) and insert 21 and 28, in this order. Show each step and specify each rotation type.

3. (10) For 2-3 trees,

   (a) (5) Draw a 2-3 tree with 11 keys (keys from 1 to 11) and height 3, where all nodes at level 2 are 2-nodes (the root is at level 1).

   (b) (5) *Delete* the key in the leftmost node at level 2 and draw the resulting 2-3 tree. From the resulting tree, *delete* the *smaller* key of the root node. Draw the new 2-3 tree. (Note: When you delete a key from a nonleaf node, replace the key by one in its left subtree.)

4. (17) For red-black trees, use the *bottom − up* algorithm for this problem.

(a) (7) Construct a red-black tree by inserting the keys in the following sequence into an initially empty red-black tree : 12, 10, 7, 2, 6, and 9.
Show each step.

```
       5
   //      \\
   2        8
 /   \    /   \
1     3  6     9
         \\    \\
          7     10
```

Figure 2. Red-black tree

(b) (5) For the red-black tree of Figure 2, perform the *Delete* operation for key value 9, show the resulting tree. *Delete* again 10 and show the resulting tree. Double edge indicates a red pointer and single indicates a black pointer.

(c) (5) For the red-black tree of Figure 2, perform the *split* operation for key value 6, showing each step.