Instructor: Dr. Sartaj Sahni
Summer, 2005

Advanced Data Structures
(NTU AD 711R)
Exam 02

CLOSED BOOK
60 Minutes

Name: _____

PLEASE READ THE FOLLOWING INSTRUCTIONS CAREFULLY

1. **For all problems, use only the algorithms discussed in class/text.**
2. **Write your name at the top of every exam sheet.**
3. **Write your answers directly on the exam question sheet.** You may use scrap paper for work, but these will not be graded.
4. All answers will be graded on correctness, efficiency, clarity, elegance and other normal criteria that determine quality.
5. The points assigned to each question are provided in parentheses.
6. You may use only a pen or a pencil. No calculators allowed.
7. Do not write on the reverse side of the exam sheet.
8. Do not write close to the margins since those areas do not always make it through when faxed.

1. (50) For *min Fibonacci heaps*:

   (a) (25) Perform the following sequence of operations on an initially empty *min Fibonacci heap* in this order:

   Insert(18), Insert(10), Insert(16), Insert(6), Insert(4), Insert(15), RemoveMin

   Draw min Fibonacci heaps before and after performing the *remove min* operation.

   (b) (25) For the *min Fibonacci heap* shown in Figure 1, perform a *DecreaseKey* operation by changing 33 to 4. T represents a *ChildCut* of $TRUE$ and F represents $FALSE$.

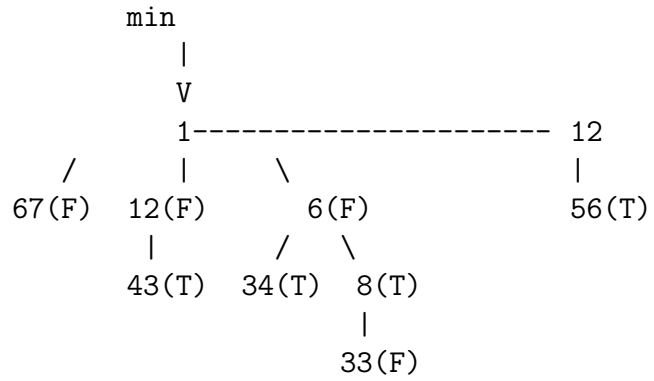

Figure 1. Min Fibonacci heap

   Draw the resulting *min Fibonacci heap* and clearly label *ChildCut* values.

Name:

Continue work here if necessary.

Name:

2. (40) For the *two-pass max pairing heap* shown in Figure 2,

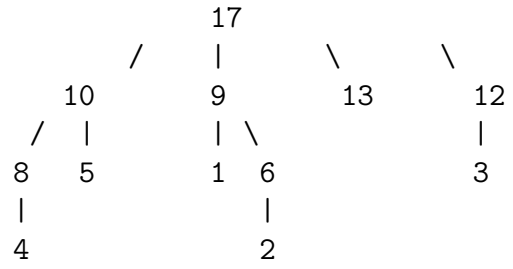

Figure 2. Max pairing heap

(a) (20) Perform a *RemoveMax* operation. Show each step.

(b) (20) Perform *remove(9)* from the original max pairing heap shown in Figure 2. Show each step.

Name:

Continue work here if necessary.

Name:

3. (30) Start with an empty AVL tree, perform *insert* operations using the following keys in the order: 12, 10, 8, 14, 16, and 11. Show each step and specify rotation type if applied.

Name:

Continue work here if necessary.

Name:

4. (60) For *2-3 trees*,

   (a) (30) Perform *Insert(30)* on the 2-3 tree shown in Figure 3. Draw the resulting *2-3 tree* and explain how you get your result.

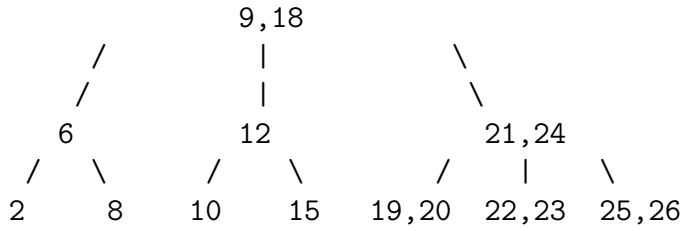

Figure 3. 2-3 tree

   (b) (30) Construct a *2-3 tree* with height 3 (i.e., the tree has 3 levels) that consists of only 2-nodes. Delete the key value of the *root* node. Show each step.(*Root node is at height 1.*)

Name:

Continue work here if necessary.

Name:

5. (70) For *red-black* trees, use the *bottom-up* algorithm for this problem. Double lines indicate a red edge and single line a black edge.

   (a) (40) Perform the following sequence of operations on the red-black tree shown in Figure 4 in this order:
   Insert(7), Insert(4), Delete(10)
   Show each step and specify rotation type/color flip/rebalancing strategy if applied.

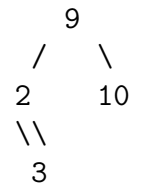

Figure 4. Red-black tree

   (b) (30) Consider the red-black tree shown in Figure 5. Perform *Split(18)* operation, showing each step.

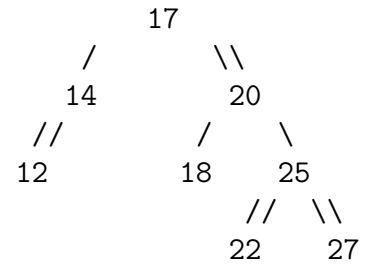

Figure 5. Red-black tree

Name:

Continue work here if necessary.