Instructor: Dr. Sartaj Sahni
Fall, 2003

Advanced Data Structures
(COP 5536 /NTU AD 711R)
Exam 1

CLOSED BOOK
60 Minutes
Take one Week after Lecture 13 (Oct. 1st 2003)

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. (10) Suppose that we use a dynamic array to store a stack. The *push(x)* operation takes $O(1)$ time if the stack is not full. In case it is full, we square the size of the array by allocating a new array, copying all elements from the old array to a new array, and storing element x in the new array. This takes $O(k^2)$ time where k is the array size before the increase. Assume that we start with an empty stack using an array of size 2 and perform n *push()* operations. What is the amortized cost of a *push()* operation?
(You may use the equality N^2 + N + $\sqrt{N}$ +$\sqrt[4]{N}$ + ... = $O(N^2)$)

2. (12)

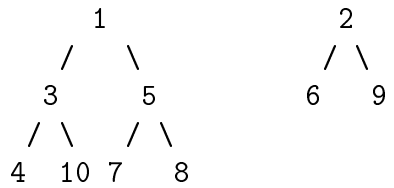
   (a) (3) Construct a *winner-tree* for the following sequence of elements:
   3, 2, 7, 1, 5, 8, 6, 4

   (b) (3) Construct a *loser-tree* for the sequence of elements given in (a).

   (c) (6) Give an application for a tournament tree (either winner tree or loser tree) and describe how to use the tree in this application.

3. (16)

(a) (6) *Meld* the following two *height-biased min leftist trees*, showing each step.



(b) (10) Let a node have fields $LeftChild$, $RightChild$, key, and *shortest*. Write a pseudo code to convert an arbitrary binary tree to a *height-biased leftist tree*. Assume that initially all nodes have *shortest* value *zero*. Your code should run in $O(n)$ time where n is the number of nodes in the tree.

4. (12)Perform the following operations on an initially empty *min binomial heap* (showing each step).

   Insert(10), Insert(3), Insert(7), Insert(5), RemoveMin, RemoveMin, Insert(6), Insert(20), RemoveMin, Insert(3), Insert(4)