Name: _____

Email address: _____

Quiz Section: _____

SECTION A

CSE 326 Winter 2007: Midterm Exam

(closed book, closed notes, calculators o.k.)

Instructions Read the directions for each question carefully before answering. We will give partial credit based on the work you **write down**, so show your work! Use only the data structures and algorithms we have discussed in class or which were mentioned in the book so far.

Note: For questions where you are drawing pictures, please circle your final answer for any credit. There is one extra page at the end of the exam that you may use for extra space on any problem. If you detach this page it must still be turned in with your exam when you leave.

Advice You have 50 minutes, **do the easy questions first**, and work quickly!

Total: 115 points. Time: 50 minutes.

Question	Max Points	Score
1	15	
2	10	
3	9	
4	15	
5	10	
6	16	
7	15	
8	15	
9	10	
Total	115	

1. (15 pts) **Big-O, Big Ω, Big Θ True/ False**

Indicate for each of the statements below whether the statement is true or false. You do not need to state the reason, but a reason may help you get partial credit.

a) $N^2 = O(N^3)$

b) $5\ N^2 = \Omega(N^3)$

c) $54\ N^2 = \Theta(N^3)$

d) $N \log N + 137\ N = \Theta(N \log N)$

e) $(3^N)^{1/3} = \Theta\ (3^N)$

2. (10 pts) **Recurrence Relationships -**
Please circle your answer. ***Be sure to keep track of constants exactly*** (e.g. don't use "C" in your answer). We want the actual function, not just the big-O class.

a) Suppose that the running time of an algorithm satisfies the recurrence relationship

$T(N)=T(N-1)+4$ for N and integer greater than 1.

and

$T(1)=10$.

Solve for T(N). In other words express T(N) as a function of N. For partial credit, please show your work.

b) Suppose that the running time of an algorithm satisfies the recurrence relationship

$T(N)=2T(N-1)+1$ for N and integer greater than 1.

and

$T(1)=1$.

Solve for T(N). In other words express T(N) as a function of N. For partial credit, please show your work.

3. (9 pts) **Trees.** No explanations are necessary, but explanations may yield partial credit. Please circle your answer.

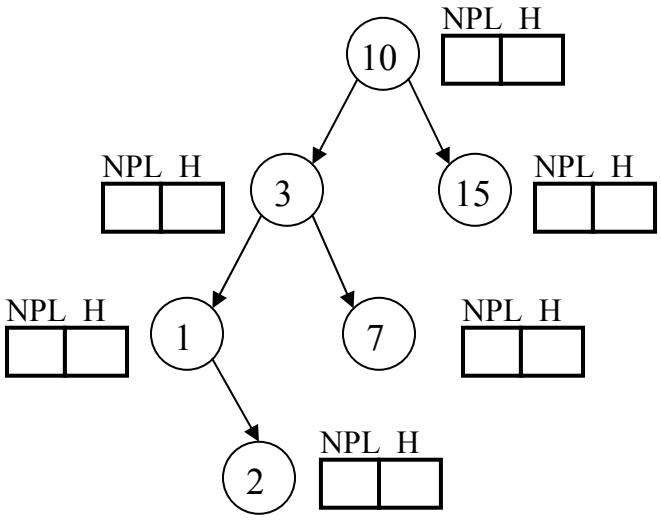
   a) How many leaves are in a perfect binary tree of height h?

   b) How many edges are in a perfect binary tree of height h?

   c) Does a complete binary tree satisfy the AVL balance condition? Yes or No.

4. (15 pts) **More Trees**

a.) (6 pts) Mark the following properties for each node of the tree below in the space indicated for each node: **Null Path Length (NPL)** and **Height (H).**



b.) (9 pts) Also, circle **yes** or **no** to indicate whether the tree above might represent each of the following data structures. If you circle **no,** give **one specific reason** why the tree could **not** be that data structure.

- AVL tree `yes` `no`
- Binary Search Tree `yes` `no`
- Leftist heap `yes` `no`

5. (10 pts) **AVL Trees** Draw the AVL tree that results from inserting the keys 1, 2, 4, 8, 5, 3, 15 in that order into an initially empty AVL tree. You are only required to show the final tree, although if you draw intermediate trees, ***please circle your final result for ANY credit.***

6. (16 pts) **Running Time Analysis**

Give a O-bound on the *worst case* running time for each of the following in terms of *n.* **No explanation is required**, but an explanation may help for partial credit. Assume that all keys are distinct.

a) Merging two leftist heaps, the largest of size N

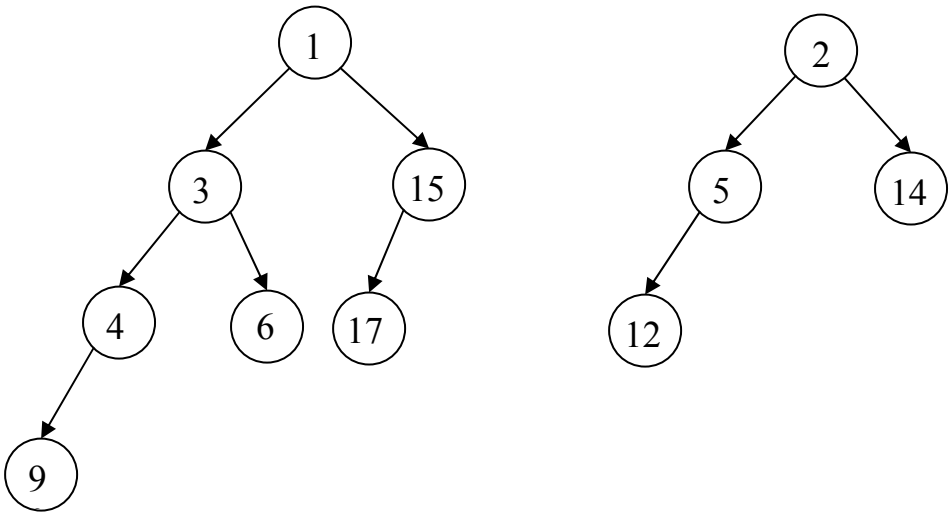
b) Finding the maximum of a binary min heap

c) Finding the maximum of an AVL tree

d) Finding the minimum of a binomial queue (assume the queue has binomial trees which obey the min-heap order property)

7. (15 pts) **Leftist Heap Merge**

Merge the following two leftist min heaps using the leftist heap merge described in class. You are only required to show the final tree, although if you draw intermediate trees, ***please circle your final result for ANY credit.*** You may continue your answer to this question on the next page if needed.



7. **Leftist Heap Merge** continued

8. (15 pts) **Binary Min Heaps**

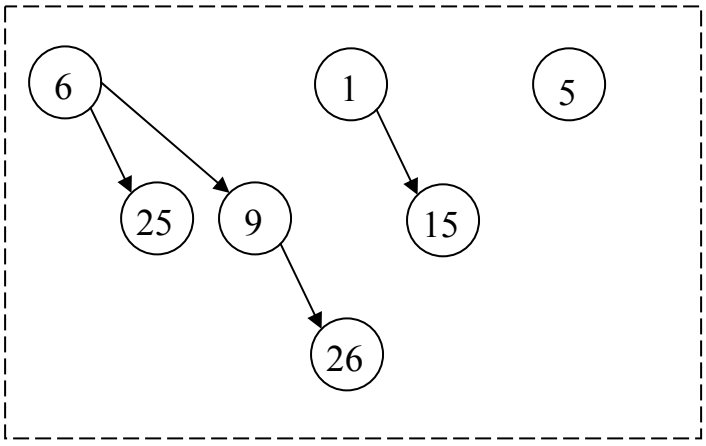
a) (10 pts) Draw the binary min heap that results from inserting 11, 2, 1, 14, 3, 15, 5, 8, 0 in that order into an **initially empty binary heap**. You do not need to show the array representation of the heap. You are only required to show the final tree, although if you draw intermediate trees, ***please circle your final result for ANY credit.***

8. **Binary Min Heaps continued**

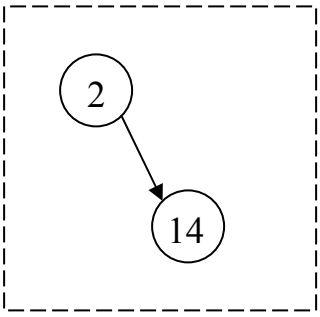
b) (5 pts) Draw the result of doing 2 deletemins on the heap you created in part a. You are only required to show the final tree, although if you draw intermediate trees, ***please circle your final result for ANY credit.***

9. (10 pts) **Binomial Queues** –

Merge the two Binomial Queues shown below using the algorithm described in class. You do not need to show the array representation of the heap. You are only required to show the final tree, although if you draw intermediate trees, ***please circle your final result for ANY credit.*** You may continue onto the next page if needed.



Queue 1



Queue 2

9. **Binomial Queues** continued

Scratch Paper Page (Please Turn In)

Scratch Paper Page (Please Turn In)