Name: Key

Email address: ______________________

CSE 373 Winter 2009: Midterm #1

(closed book, closed notes, NO calculators allowed)

Instructions: Read the directions for each question carefully before answering. We may give partial credit based on the work you **write down**, so if time permits, 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.

Good Luck!

Total: 85 points. Time: 50 minutes.

Question	Max Points	Score
1	16	
2	8	
3	13	
4	14	
5	6	
6	18	
7	10	
Total	85	

1. (16 pts) **Big-O**

For each of the functions *f(N)* given below, indicate the tightest bound possible (in other words, giving $O(2^N)$ as the answer to every question is not likely to result in many points). Unless otherwise specified, all logs are base 2. You MUST choose your answer from the following (not given in any particular order), each of which could be re-used (could be the answer for more than one of a) – h)):

> $O(N^2)$, $O(N^3 \log N)$, $O(N \log N)$, $O(N)$, $O(N^2 \log N)$, $O(N^5)$, $O(2^N)$, $O(N^3)$, $O(\log N)$, $O(1)$, $O(N^4)$, $O(N^N)$

You do not need to explain your answer.

a) $f(N) = N \cdot (100N + 200N^3) + N^3$ $O(N^4)$

b) $f(N) = N^2 \log N + N^3 + 1000^4$ $O(N^3)$

c) $f(N) = 100N + (N/2) \log (N/2) + N/4$ $O(N \log N)$

d) $f(N) = (N/4) + N^{1/2}$ $O(N)$

e) $f(N) = N \log(N^4) + 3 N^2$ $O(N^2)$

f) $f(N) = (N^3 + N^1) / N$ $O(N^2)$

g) $f(N) = 400 N^2 – 20N$ $O(N^2)$

h) $f(N) = N^2 \log N + N \log (N^2)$ $O(N^2 \log N)$

2. **(8 pts) Big-Oh and Run Time Analysis:** Describe the worst case running time of the following pseudocode functions in Big-Oh notation in terms of the variable n. ***Showing your work is not required*** (although showing work ***__may__*** allow some partial credit in the case your answer is wrong – don’t spend a lot of time showing your work.). You MUST choose your answer from the following (not given in any particular order), each of which could be re-used (could be the answer for more than one of I. – IV.):

$O(n^2)$, $O(n^3 \log n)$, $O(n \log n)$, $O(n)$, $O(n^2 \log n)$, $O(n^5)$, $O(2^n)$, $O(n^3)$,
$O(\log n)$, $O(1)$, $O(n^4)$, $O(n^n)$

```
 I.   void silly(int n) {
         for (int i = 0; i < n; ++i) {
            for (int j = 0; j < i; ++j) {
               System.out.println(”j = ” + j);
               for (int k = 0; k < j; ++k)
                  System.out.println(”k = ” + k);
            }
         }
      }
```

Runtime:

$O(n^3)$

```
 II.  void silly(int n, int x, int y) {
         for (int i = 0; i < n; ++i) {
            if (x > y) {
               for (int j = 0; j < n; ++j)
                  System.out.println(”j = ” + j);
               for (int k = 0; k < n * n; ++k)
                  System.out.println(”k = ” + k);
            } else
               System.out.println(”i = ” + i);
         }
     }
```

$O(n^3)$

```
III. void silly(int n, int m) {
         if (m > n) return;
         System.out.println(”m = ” + m);
         silly(n, m+2);
      }
```

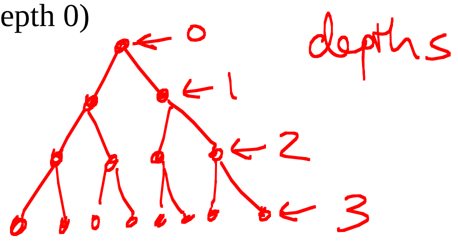
$O(n)$

```
 IV. void silly(int n) {
         for (int i = 0; i < n; i = i + 10) {
            for (int j = 0; j < i; ++j) {
               System.out.println(”i = ” + i);
               System.out.println(”j = ” + j);
            }
         }
   }
```

$O(n^2)$

3. (13 pts) **Read each question carefully.** Explanations are not necessary, but may yield partial credit. You must solve any summations or recurrences for full credit.

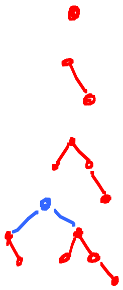
a) (3 pts) What is the minimum and maximum number of nodes *at depth d* in a **perfect** binary tree? Be sure to list the nodes **at** depth d. Do not include nodes at depth d-1 or d+1 or other depths. (Hint: the root is at depth 0)



Minimum = 2^d

Maximum = 2^d

b) (4 pts) What is the minimum and maximum number of nodes in a balanced **AVL tree** of height 5?



h	N ← Min # of nodes
0	1
1	2
2	4
3	7

Minimum = 20

Maximum = 63

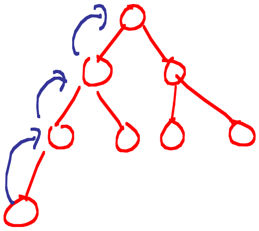
↑ a perfect tree = $2^{h+1} - 1$

Min# Nodes(h) = MinNodes(h-1) + MinNodes(h-2) + 1

c) (4 pts) You are given a **binary min heap** of height h. The minimum and maximum number of *comparisons* we might have to do when inserting the next value (in terms of h) is:

ex.

3 comparisons max

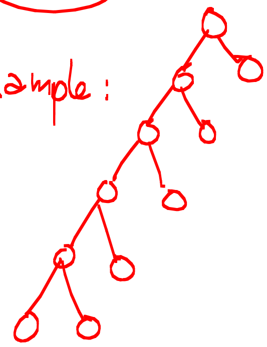


Minimum = 1

Maximum = h

d) (2 pts) A **full** binary tree satisfies the AVL balance condition at each node. (circle one) TRUE or [✓ FALSE]

Counterexample:



4. (14 pts) **Trees**

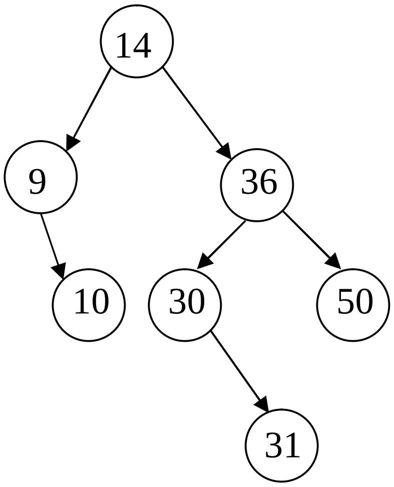
a.) (6 pts) Give traversals of the tree shown below:

In-Order: 9, 10, 14, 30, 31, 36, 50

Post-Order: 10, 9, 31, 30, 50, 36, 14

Pre-Order: 14, 9, 10, 36, 30, 31, 50

b.) (8 pts) Given the following tree:



Circle **yes** or **no** to indicate whether the tree above might represent each of the following data structures. If you circle **no, you must *draw a new picture* containing the same values that satisfies the given condition.**).

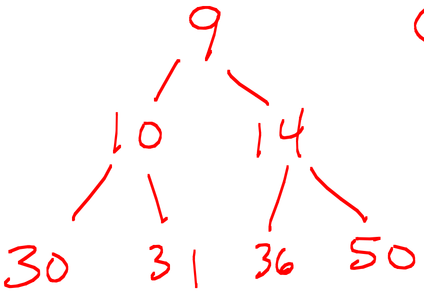
1) AVL tree [✓ yes] no

2) Binary Search Tree [✓ yes] no

3) Complete Tree yes [✓ no]

4) Binary Min Heap yes [✓ no]

Draw and label any necessary pictures here:

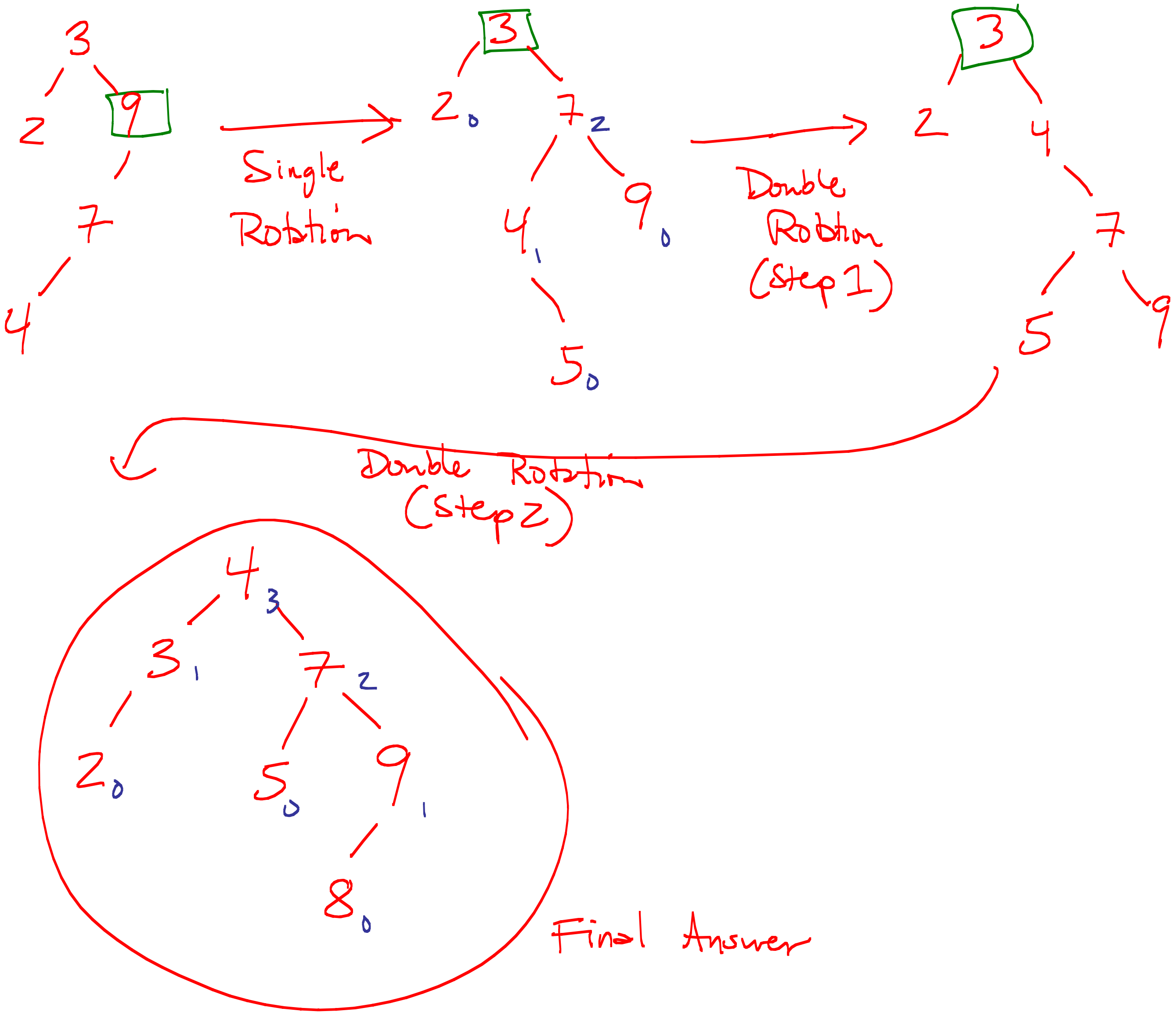


A complete tree also a binary min heap

Causes Single Rotation

Causes Double Rotation

5. (6 pts) **AVL Trees** Draw the AVL tree that results from inserting the keys: 3, 9, 2, 7, 4, 5, 8 in that order into an initially empty AVL tree. You are only required to show the final tree, although drawing intermediate trees may result in partial credit. If you draw intermediate trees, ***please circle your final tree for ANY credit.***



6. (18 pts) **Running Time Analysis:** Give the tightest possible upper bound for the *worst case* running time for each of the following in terms of *N*. You MUST choose your answer from the following (not given in any particular order), each of which could be re-used (could be the answer for more than one of a) – f)):

$O(N^2)$, $O(N^3 \log N)$, $O(N \log N)$, $O(N)$, $O(N^2 \log N)$, $O(N^5)$, $O(2^N)$, $O(N^3)$, $O(\log N)$, $O(1)$, $O(N^4)$, $O(N^N)$

****For any credit, you must explain your answer.** Assume that the most time-efficient implementation is used. Assume that all keys are distinct.

a) Inserting N values into an initially empty Binary Search Tree

Explanation: Worst case is values are given in increasing or decreasing order, resulting in a linked list + number of nodes that must be examined $= 1 + 2 + 3 + 4 + \cdots + N-1$

$$\cong \frac{N(N+1)}{2} = O(N^2)$$

a) $O(N^2)$

b) Finding the minimum value in a Binary Min Heap of size N

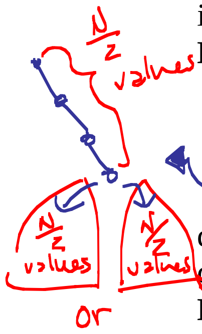
Explanation: The minimum value is always located at the root (location 1 in the array) – constant time access to find.

b) $O(1)$

c) Inserting a value into a Binary Search Tree of size N, where the value you are inserting is the median value when compared to values currently in the tree. Inserting the

Explanation: median value ("value in the middle – half values are > median, half are < median) in the worst case may require traversing half of the values similar to in a linked list). $\frac{N}{2} = O(N)$

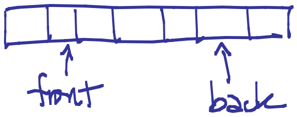
Ex.



c) $O(N)$

d) Dequeue from a queue containing N elements implemented as a circular array (as described in lecture)

Explanation:



Deque just requires moving the front pointer + returning the value pointed to. Constant time.

d) $O(1)$

e) Pre-order traversal of a Perfect Binary Tree containing N elements

Explanation: The order of the traversal does not matter, all N elements must be visited (and are only visited a constant # of times each).

e) $O(N)$

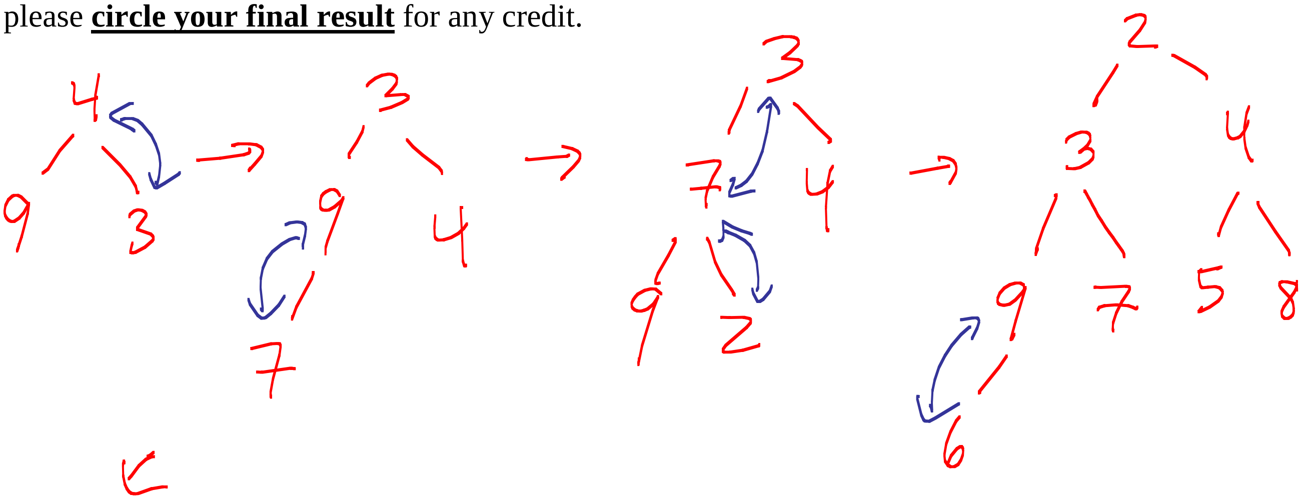
f) Finding the maximum value in an AVL Tree containing N elements.

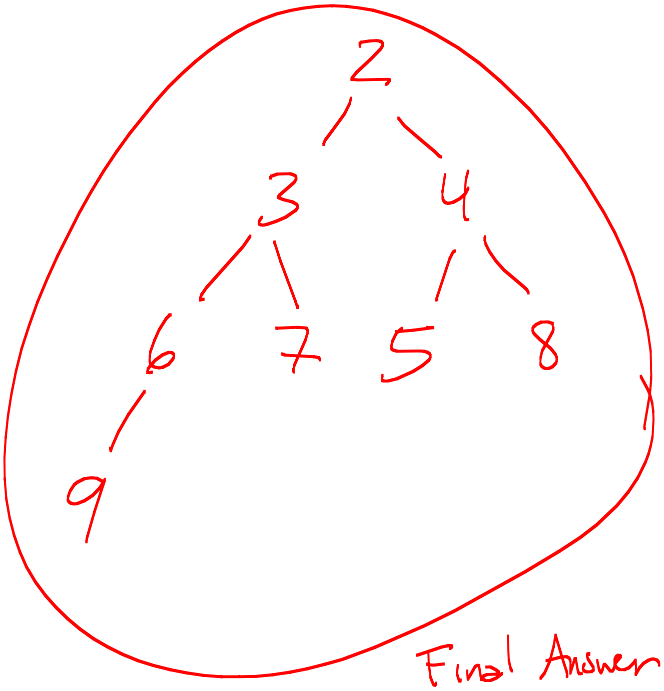
Explanation: Find the max in a BST (or AVL tree) by following right child pointers until you hit a NULL. Max height of an AVL tree is $O(\log N)$, so will only need to examine $O(\log N)$ elements at most.

f) $O(\log N)$

7. (10 pts total) **Binary Min Heaps**

(a) [6 points] Draw the binary min heap that results from inserting ~~4, 9, 3, 7, 2~~, 5, 8, 6 in that order into an initially empty binary min heap. You do not need to show the array representation of the heap. You are only required to show the final tree, although drawing intermediate trees may result in partial credit. If you draw intermediate trees, please **circle your final result** for any credit.

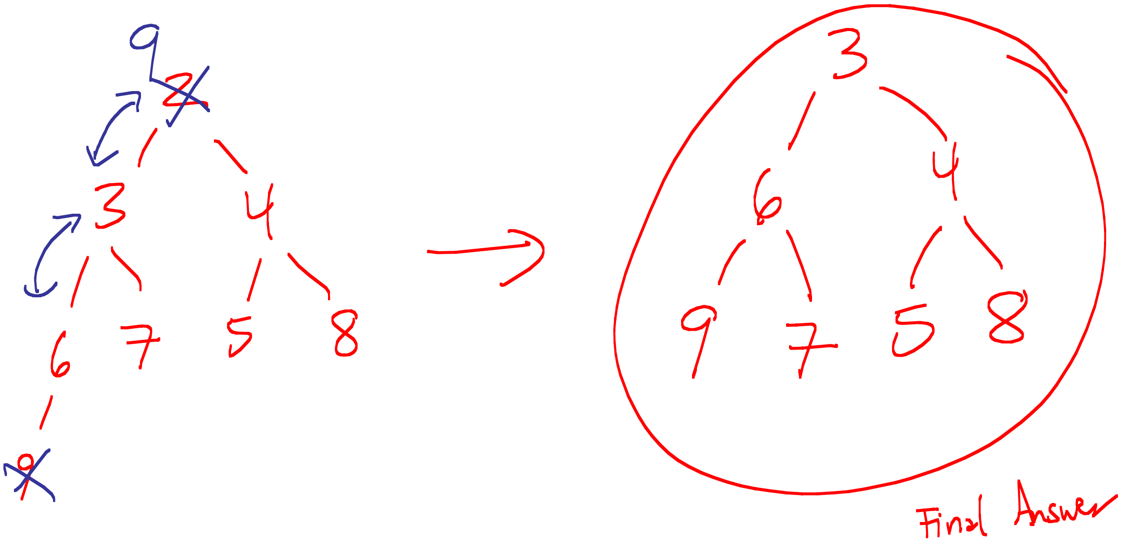




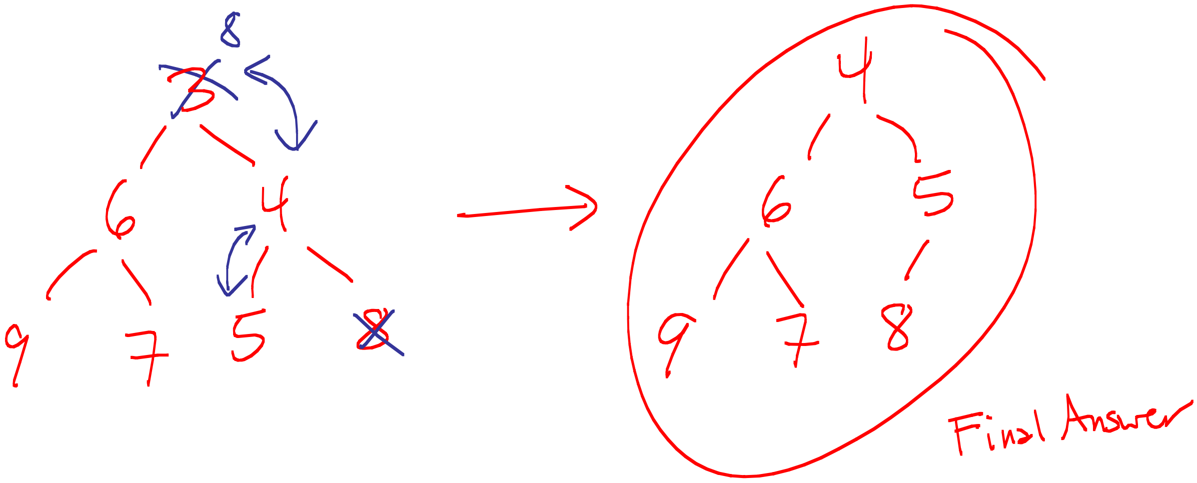
Final Answer

7. **(cont.)**

(b) [2 points] Draw the result of one deletemin call on your heap draw in part (a).



(c) [2 points] Draw the result of one deletemin call on your heap draw in part (b).



Scratch Paper Page