Name: _____

Email address: _____

Quiz Section: _____

CSE 332 Spring 2014: Midterm Exam

(closed book, closed notes, no calculators)

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 that 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: 100 points. Time: 50 minutes.

Question	Max Points	Score
1	18	
2	6	
3	10	
4	6	
5	8	
6	6	
7	8	
8	15	
9	15	
10	8	
Total	100	

1. (18 pts) Big-Oh

(2 pts each) For each of the operations/functions 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. **Your answer should be as "tight" and "simple" as possible.** For questions that ask about running time of operations, assume that the most efficient implementation is used.

You do not need to explain your answer.

a) *Determine what the maximum value is in a **hash table** containing N elements where quadratic probing has been used to resolve collisions (worst case)* _____

b) $T(N) = T(N-2) + 12$ _____

c) *Dequeue in a **queue** containing N elements implemented using an array (worst case)* _____

d) $f(N) = \log (N + N) + \log^2 N$ _____

e) *Remove(k) on a **binary min heap** containing N elements. Assume you have a reference to the key k that should be removed. (worst case)* _____

f) *Find in a **separate chaining hash table** containing N elements where each bucket points to a binary search tree (worst case)* _____

g) $f(N) = \log^2 N + N \log N$ _____

h) *Merging two **binary search trees** containing N elements each. (worst case)* _____

i) *Finding the kth largest item in an **AVL tree** containing N elements (worst case)* _____

2. **(6 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`. Your answer should be as "tight" and "simple" as possible.
Showing your work is not required

Runtime:

I.
```
int funny (int n) {
    if (n < 10)
       return n - 2;
    else {
       for (int i = 0; i < n; i++) {
          print i
       }
       return funny (n - 3);
    }
}
```

II.
```
int happy (int n, int sum) {
    for (int k = 0; k < n; k = k++) {
       for (int i = 0; i < k; i++) {
          sum++;
       }
    }
    for (int j = 3 * n; j > 0; j--) {
          sum++;
    }
    return sum;
}
```

III.
```
void sunny (int n, int sum) {
    int i = n
    while (i > 0) {
       for (int j = 0; j < n * n; j++) {
          sum++;
       }
       i = i/2
    }
    return sum;
}
```

3. (10 pts) Big-O, Big Ω, Big Θ

(2 pts each) For parts (a) – (e) circle **ALL** of the items that are TRUE. You do not need to show any work or give an explanation.

a) $N^4 + N^3$ is:

Ω (N^6) O (N^6) Θ (N^6) None of these

b) $N \log N + 2 N^2 + 3 N^2 + 4 N + 50$ is:

Ω (N^2) O (N^2) Θ (N^2) None of these

c) $5 N^2 \log N$ is:

Ω (N^3) O (N^3) Θ (N^3) None of these

d) $8 N + \log N + N^4$ is:

Ω (N^3) O (N^3) Θ (N^3) None of these

e) $3 N + N \log \log N$ is:

Ω (N log N) O (N log N) Θ (N log N) None of these

4. (6 pts) Recurrence Relationships -

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

$$T(1) = 6.$$

and

$$T(N) = T(N-1) + N \quad \text{for integers } N > 1$$

Find the closed form for T(N) ***and show your work step by step***. In other words express T(N) as a function of N. Your answer should *not* be in Big-Oh notation – show the relevant *exact* constants in your answer (e.g. don't use "C" in your answer).

5. (8 pts) Draw the AVL tree that results from inserting the keys 7, 1, 2, 5, 8, 9, 4, 3 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. (6 pts) Trees

a) (2 pts) What is the **minimum number of nodes than must be visited in order to find the largest value** in an **AVL tree of height 4?** ***Give an exact number*** not a formula. (include the root node in your count)

b) (4 pts) Given a **complete binary** tree of height h, what are the minimum and maximum **number of nodes in the right subtree** of the root?

Minimum:

Maximum:

7. (8 pts) Trees

a) (4 pts) Given a B-tree (as defined in lecture and in Weiss) with M =8 and L = 4, if you have inserted 300 items into the tree, **what is maximum height of the tree**? Give an exact number, not a formula for your answer. **Explain briefly how you got your answer.**

b) (4 pts) Given an AVL tree, with 300 items inserted into the tree, **what is the maximum height of the tree**? Give both a fully simplified numeric answer (a *number*, not a formula containing things like summations or logs or exponents) and a **brief explanation of how you arrived at your answer**. Note: Your numeric answer does not have to be exact, the explanation of how you came up with your answer is more important.

8. (15 pts) Hash Tables

For a) and b) below, insert the following elements in this order: 70, 21, 49, 60, 59, 39. For each table, TableSize = 10, and you should use the primary hash function h(k) = k%10. If an item cannot be inserted into the table, please indicate this and continue inserting the remaining values.

a) Quadratic probing hash table

0	
1	
2	
3	
4	
5	
6	
7	
8	
9	

b) Separate chaining hash table – use a linked list for each bucket where the values are ordered by **increasing value**

0	
1	
2	
3	
4	
5	
6	
7	
8	
9	

c) What is the load factor in Table b)?

d) What is the worst case running time of a find operation in table b?

d) In a sentence or two, describe **double hashing**.

e) What is one advantage of **double hashing** over **quadratic probing**, **be specific**.

9. (15 pts) Min Heaps -

a) (8 pts) As discussed on homework 2, a three heap with *n* elements can be stored in an array A, where A[0] contains the root of the tree. Draw the **three** min heap that results from inserting 15, 2, 5, 7, 3, 6, 12, 1, 4 in that order into an **initially empty three 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.***

9. Min Heaps (continued)

b) (2 pts) Draw the result of doing 1 deletemin 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.***

c) (1 pts) Draw the array representation of your heap from part b.

d) (2 pts) What is the big-O worst case running time for a **findmin** operation in a **three** (min) heap containing N elements?

e) (2 pts) What is the big-O worst case running time for a **deletemin** operation in a **three** (min) heap containing N elements?

10. (8 pts) B-tree Insertion

a) (2pts) In the B-Tree shown below, please write in the appropriate values for the interior nodes.

b) (2 pts) Based on the picture below, what are the values for M and L?

M = L =

c) (4 pts) Starting with the B-tree shown below, insert **28**. Draw and circle the resulting tree (*including values for interior nodes*) below. Use the method for insertion described in lecture and used on homework.