

Name: \_\_\_\_\_

Email address: \_\_\_\_\_

Quiz Section: \_\_\_\_\_

## **CSE 332 Spring 2013: 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.

<b>Question</b>	<b>Max Points</b>	<b>Score</b>
1	18	
2	6	
3	10	
4	6	
5	8	
6	8	
7	8	
8	14	
9	14	
10	8	
<b>Total</b>	<b>100</b>	

**1. (18 pts) Big-Oh**

(2 pts each) 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. **Your answer should be as “tight” and “simple” as possible.**

You do not need to explain your answer.

a)  $f(N) = N \log N + N \log \log N$  \_\_\_\_\_

b)  $f(N) = 50 \log N^2 + 100 (\log N)^2$  \_\_\_\_\_

c)  $f(N) = N \log_2(4^N)$  \_\_\_\_\_

d) *Push in a **stack** containing  $N$  elements implemented using linked list nodes (worst case)* \_\_\_\_\_

e) *A preorder traversal in a **binary search tree** containing  $N$  elements (worst case)* \_\_\_\_\_

f) *Insert in a **separate chaining hash table** containing  $N$  elements where each bucket points to an AVL tree (worst case)* \_\_\_\_\_

g) *IncreaseKey( $k, v$ ) on a **binary min heap** containing  $N$  elements. Assume you have a reference to the key  $k$ .  $v$  is the amount that  $k$  should be increased. (worst case)* \_\_\_\_\_

h)  $T(N) = T(N-1) + N$  \_\_\_\_\_

i)  $T(N) = T(N/2) + 100$  \_\_\_\_\_

**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*

```
I. int sunny (int n) {  
    if (n < 10)  
        return n - 1;  
    else {  
        return sunny (n / 2);  
    }  
}
```

Runtime:

```
II. int funny (int n, int sum) {  
    for (int k = 0; k < n * n; ++k)  
        for (int j = 0; j < k; j++)  
            sum++;  
    return sum;  
}
```

```
III. int happy (int n, int sum) {  
    for (int k = n; k > 0; k = k - 1) {  
        for (int i = 0; i < k; i++)  
            sum++;  
        for (int j = n; j > 0; j--)  
            sum++;  
    }  
    return sum;  
}
```

3. (10 pts) Big-O, Big  $\Omega$ , Big  $\Theta$

(2 pts each) For parts (a) – (c) circle whether the statement is true or false. Part (d) has its own instructions. You do not need to show your work for parts (a) – (c).

TRUE / FALSE      a)  $5N^2 + 100N = \Omega(N)$

TRUE / FALSE      b)  $100N^2 + N^3 + 10N = \Theta(N^3)$

TRUE / FALSE      c)  $50N^2 + 10N + 20 = \Omega(N^2)$

d) (4 points) Demonstrate that:  $10n^2 + 5n + 42$  is  $O(n^2)$  by finding positive integers  $c$  and  $n_0$  such that the definition of Big Oh is satisfied. ***For full credit you must do more than just giving values for  $c$  and  $n_0$ .*** You do not need to give a proof, just briefly demonstrate why your constants are appropriate.

**4. (6 pts) Recurrence Relationships -**

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

$$T(1) = 5.$$

and

$$T(N) = T(N-1) + 7 \quad \text{for integers } N > 1$$

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 9, 8, 4, 3, 1, 6, 7 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. (8 pts) AVL Trees**

a) ( 2 pts) What is the **minimum number of nodes** in an **AVL tree of height 4**? (Hint: the height of a tree consisting of a single node is 0) *Give an exact number* not a formula.

b) ( 2 pts) What is the **minimum number of nodes than must be examined in order to find the maximum value** in an **AVL tree of height 4**? *Give an exact number* not a formula.

c) ( 2 pts) What is the **maximum number of nodes than must be examined in order to find the minimum value** in an **AVL tree of height 4**? *Give an exact number* not a formula.

d) ( 2 pts) **Draw an example** of an **AVL tree of height 4** with the **minimum number of nodes**. You do not need to put values in the tree, just show the shape using dots for nodes.

## 7. (8 pts) B-trees

a) (4 pts) Given  $M = 6$  and  $L = 20$ , what is the minimum and maximum number of data items in a B-tree (as defined in lecture and in Weiss) of height 4? Give a single number as your answer, partial credit may be given for showing your work.

Minimum number of data items =

Maximum number of data items =

b) (4 pts) Given the following parameters:

1 Page on disk = 1000 bytes

Disk access time = 2 milli-secs per byte

Key = 4 bytes

Pointer = 16 bytes

Data = 30 bytes per record (includes key)

Assuming you can place things where you want in memory (in other words this is not a question about Java implementation of B-trees), what are the best values in a B-tree for:

$M =$

and

$L =$



**8. (14 pts) Hash Tables**

For a) and b) below, insert the following elements in this order: 72, 21, 10, 20, 1, 31. 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) Separate chaining hash table – use a unsorted linked list for each bucket, insert at front

0	
1	
2	
3	
4	
5	
6	
7	
8	
9	

b) Quadratic probing hash table

0	
1	
2	
3	
4	
5	
6	
7	
8	
9	

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

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

The following two questions are about hashing generally, NOT about the two tables above.

e) What is the big-O worst case running time for a **find** operation in a **linear probing hash table** containing N elements?

f) What is the big-O worst case running time for an **insert** operation in a **separate chaining hash table** containing N elements **where each bucket points to a sorted linked list**?

**9. (14 pts) Binary Min Heaps**

a) (8 pts) Draw the binary min heap that results from inserting 8, 2, 3, 6, 9, 7, 5, 1, 4 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.*

### 9. Binary 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) (4 pts) Given a binary min heap of height  $h$ , what is minimum and maximum number of values in the left and right subtrees of the root?

Min values in left subtree:

Max values in left subtree:

Min values in right subtree:

Max values in right subtree:

**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 **25**. Draw and circle the resulting tree (*including values for interior nodes*) below. Use the method for insertion described in lecture and used on homework.

