Name: Sample Soln

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.

| Question | Max Points | Score |
|---|---|---|
| 1 | 18 | |
| 2 | 6 | |
| 3 | 10 | |
| 4 | 6 | |
| 5 | 8 | |
| 6 | 8 | |
| 7 | 8 | |
| 8 | 14 | |
| 9 | 14 | |
| 10 | 8 | |
| **Total** | 100 | |

**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) $f(N) = N \log N + N \log \log N$ $O(N \log N)$

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

c) $f(N) = N \log_2(4^N)$ $O(N^2)$

d) *Push in a **stack** containing N elements implemented using linked list nodes (worst case)* $O(1)$

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

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

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)* $O(\log N)$

h) $T(N) = T(N-1) + N$ $O(N^2)$

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

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

$O(\log n)$

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;
}
```

$O(n^4)$

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;
}
```

$O(n^2)$

**3. (10 pts) Big-O, Big Ω, Big Θ**

(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) $50 N^2 + 10N + 20 = \Omega(N^2)$

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.

$10n^2 + 5n + 42 \le c \cdot n^2$ for $n \ge n_0$

$\boxed{\text{e.g. } c = 100, \; n_0 = 1}$

$10n^2 + 5n + 42 \le 100n^2$

$n=1$: $10 \cdot 1 + 5 \cdot 1 + 42 \le 100 \cdot 1$ ✓

$n=2$: $10 \cdot 4 + 5 \cdot 2 + 42 \le 100 \cdot 4$
$40 + 10 + 42 \le 400$ ✓

(Many possible values for c + $n_0$)

$n=3$: $10 \cdot 9 + 5 \cdot 3 + 42 \le 100 \cdot 9$
$90 + 15 + 42 \le 900$ ✓

⋮

**4. (6 pts) Recurrence Relationships -**
Suppose that the running time of an algorithm satisfies the recurrence relationship

$$T(1) = 5.$$

and

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

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).

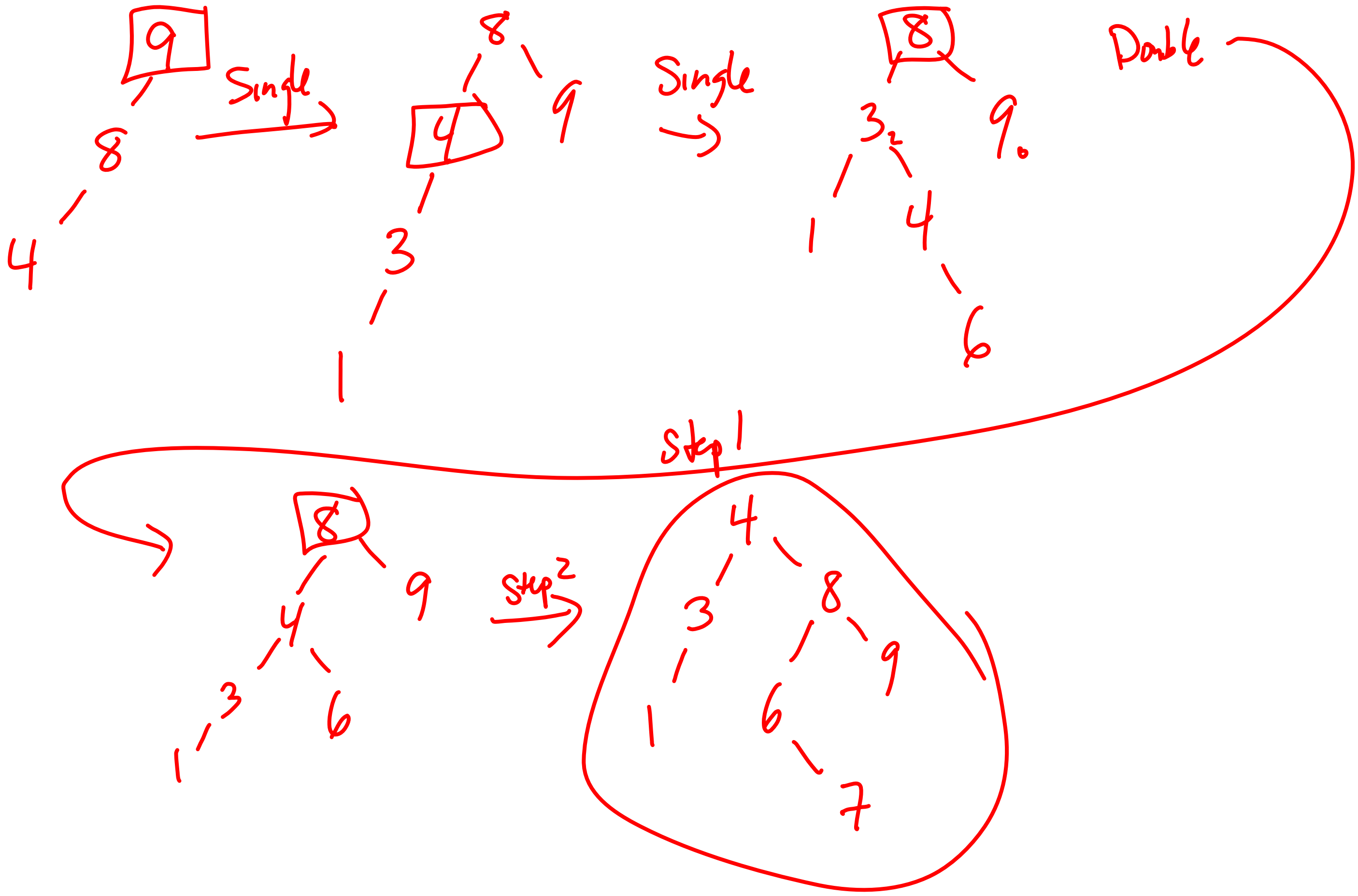
$$\begin{aligned}
T(N) &= T(N-1) + 7 \\
&= T(N-2) + 7 + 7 \\
&= T(N-3) + 7 + 7 + 7 \\
&= T(N-k) + k \cdot 7 \\
&= T(1) + (N-1)7 \\
&= 5 + 7N - 7
\end{aligned}$$

Want: $N-k=1$

$K = N-1$

$$T(N) = 7N-2$$

**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.

$N = S(h-1) + S(h-2) + 1$

$= S(3) + S(2) + 1 = 7 + 4 + 1 = 12$

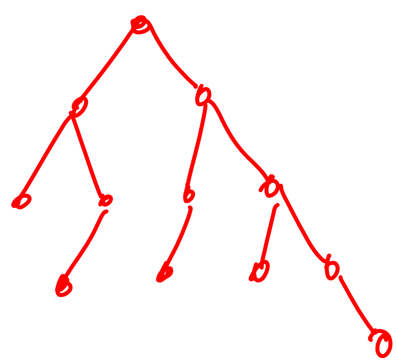
b) ( 2 pts) What is the **minimum number of nodes than must be examined in order to find the maximum value** in an **AVL tree of height 4?** ***Give an exact number*** not a formula.

3

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.

5

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.

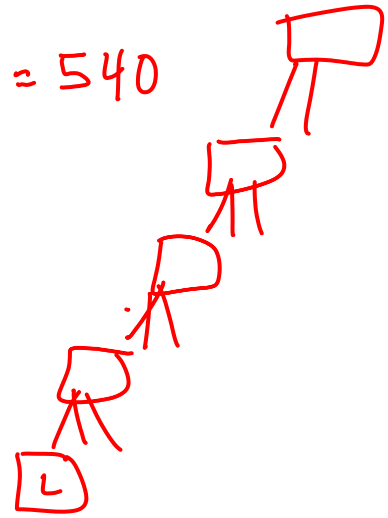


**7. (8 pts) B-trees**

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 = $2 \cdot 3^3 \cdot 10 = 2 \cdot 27 \cdot 10 = 540$

Maximum number of data items = $6^4 \cdot 20 = 25,920$




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 = 50

and

L = $\left\lfloor \frac{1000}{30} \right\rfloor = 33$

$$1000 \geq 16 \cdot M + 4(M-1)$$

$$1000 \geq 20M - 4$$

$$1004 \geq 20M$$

$$M \leq \left\lfloor \frac{1004}{20} \right\rfloor = 50$$

**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) = h%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 | | → 20 → 10 |
| 1 | | → 31 → 1 → 21 |
| 2 | | → 72 |
| 3 | | |
| 4 | | |
| 5 | | |
| 6 | | |
| 7 | | |
| 8 | | |
| 9 | | |

b) Quadratic probing hash table

| | | |
|---|---|---|
| 0 | $10_0$ | $20_0$ $31_3$ |
| 1 | $21_0$ | $20_1$ $1_0$ $31_0$ |
| 2 | $72_0$ | $1_1$ $31_1$ |
| 3 | | |
| 4 | $20_2$ | |
| 5 | $1_2$ | $31_2$ |
| 6 | | |
| 7 | $31_4$ | |
| 8 | | |
| 9 | | |

c) What is the load factor in Table a)? 6/10

d) What is the load factor in Table b)? 6/10

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?

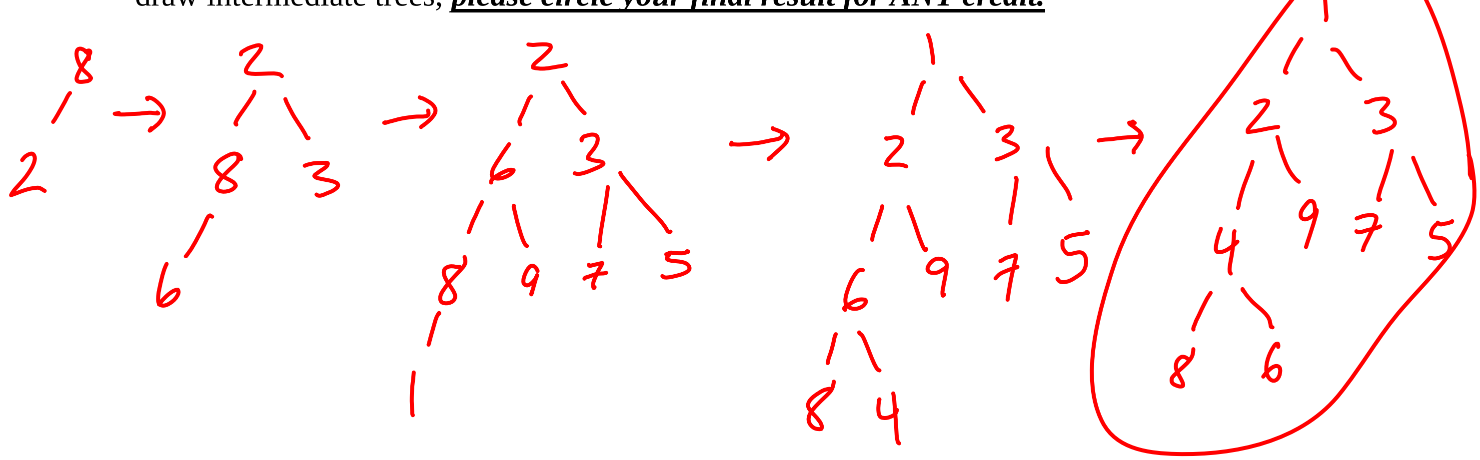
O(N)

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 <u>sorted</u> linked list**?

O(N)

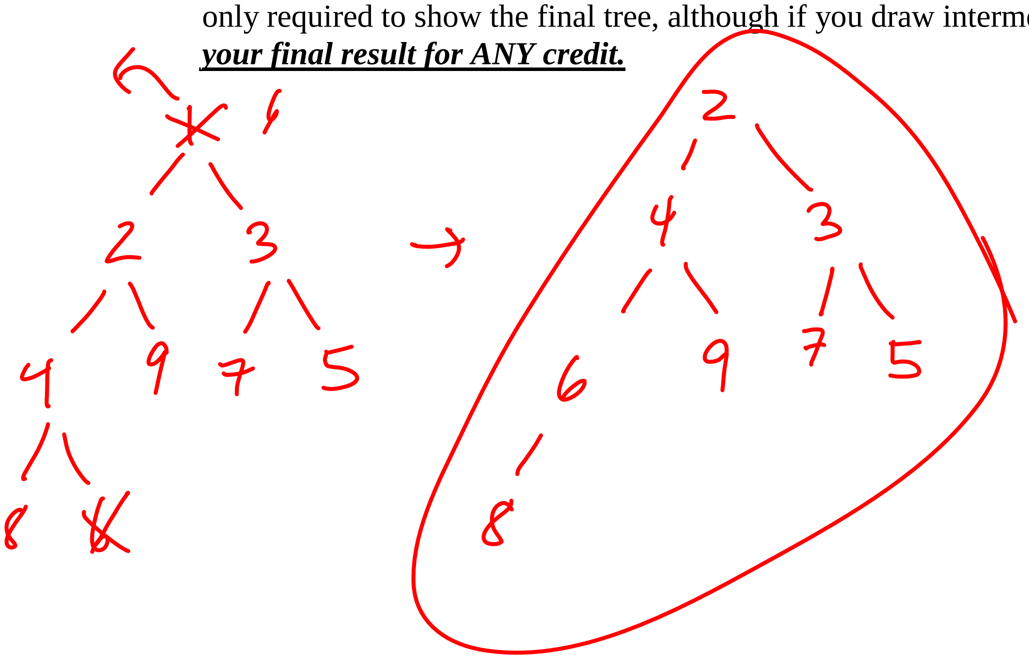
**9. (14 pts) Binary Min Heaps**

a) (8 pts) Draw the binary min heap that results from <u>inserting 8, 2, 3, 6, 9, 7, 5, 1, 4 in that order</u> 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, ***<u>please circle your final result for ANY credit.</u>***



## 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:

Min values in complete tree of height h-1 = $2^{h-1}$

Max values in left subtree:

Values in perfect tree of height h-1 = $2^{h}-1$

Min values in right subtree:

$2^{h-1}-1$

Max values in right subtree:

$2^{h}-1$

**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 = 4 L = 2

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.