Name: _____

UWNetID: _____

CSE 332 Autumn 2017: 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 so far.

Note: For questions where you are drawing pictures, please circle your final answer.

Good Luck!

Total: 100 points. Time: 50 minutes.

Question	Max Points	Score
1	20	
2	12	
3	15	
4	10	
5	9	
6	6	
7	11	
8	9	
9	8	
Total	100	

1. (20 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. For array-based structures, assume that the underlying array is large enough.

You do not need to explain your answer.

a) *Post-order traversal of all elements in a **AVL tree** containing N elements. (worst case).* _____

b) *f(N)* = $\log_3 (2^N) + (\log N)^2$ _____

c) *Combining 2 **binary min heaps,** each containing N elements, into one binary min heap (worst case).* _____

d) $T(N) = T(N/2) + 7$ _____

e) *f(N)* = $\log \log (N^3) + \log^2 N$ _____

*f) Finding the 2nd largest value in a **AVL tree** containing N elements. (worst case)* _____

g) $T(N) = 2\ T(N/2) + 6$ _____

h) *Pop in a **stack** containing N elements implemented as an array (worst case)* _____

i) *Finding the maximum value in an **binary search tree** containing N elements (worst case)* _____

j) decreaseKey*(k, amount) on a **binary min heap** containing N elements. Assume you have a reference to the key k that should be updated. (worst case)* _____

2. **(12 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.*** Note: ^ is used here to denote exponentiation. e.g. 2^3 is 8.

I.
```
void scary(int n) {
    for (int i = 1; i < 2^n; i *= 2) {
       if (i % (2^(n / 2)) == 0) {
          for (int j = i; j < n; j++) {
             print "whew!";
          }
       } else {
          print "boo!";
       }
    }
}
```
Runtime:

II.
```
int boo(int n, int candy) {
    if (n < 50) {
       for (int i = 0; i < n * n* n; i++) {
          candy++;
       }
       return candy;
    } else if (n < 2000) {
       return n * n;
    }
    candy += boo(n / 2, candy);
    return candy * boo(n / 2, candy);
}
```

III.
```
int treat(int n, int candy) {
    for (int i = 0; i < n * n * n; i++) {
       if (i % 3 == 0) {
          int j = 0;
          while (j < i) {
             candy++;
             j++;
          }
       }
    }
    return candy;
}
```

IV.
```
int haunted(int n, int candy) {
    if (n < 15) {
       return candy + n;
    } else if (n < 100) {
       return haunted(n / 2, candy);
    } else {
       for (int i = 0; i < n * n; i++) {
          candy++;
       }
       return haunted(n - 3, candy);
    }
}
```

3. (15 pts) Big-O, Big Ω, Big Θ

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

a) 2^{N+2} is:

$\Omega(2^N)$ $\Theta(2^N)$ $O(N^3)$ $\Theta(N^N)$

b) log (log N) is:

O (log N) Ω (log N) $O(\log^2 N)$ $\Omega(\log^2 N)$

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

$\Omega(\log^2 N)$ O (log N) Ω (log N) Θ (log N)

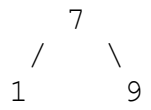
d) $N \log N + N^{3/2}$ is:

$\Omega(N^{3/2})$ $\Theta(N^{3/2})$ Ω (N log N) O (N log N)

e) log (N^2) is:

$\Omega(\log^2 N)$ Ω (log N) Θ (log N) Ω (N)

4. (10 pts) Draw the AVL tree that results from inserting the keys 2, 6, 8, 3, 4, 5 in that order into the AVL tree shown below. You are only required to show the final tree, although if you draw intermediate trees, ***please circle your final result for ANY credit.***



5. (9 pts) 4-Heaps & AVL Trees

a) (6 pts) Given a 4-heap of height h, what are the minimum and maximum number of nodes in **the left-most and right-most sub-trees** of the root? Give your answer in closed form (there should not be any summation symbols).

Min nodes in
left-**most** sub-tree:

Max nodes in
left-**most** sub-tree:

Min nodes in
right-**most** sub-tree:

Max nodes in
right-**most** sub-tree:

b) (3 pts) What is the minimum number of nodes in an AVL tree of height 4? Give an exact number, not a formula. Give a brief explanation of how you arrived at your answer.

6. (6 pts) Recurrences

Give a base case and a recurrence for the runtime of the following function. Use variables appropriately for constants (e.g. c1, c2, etc.) in your recurrence (you do not need to attempt to count the exact number of operations). **YOU DO NOT NEED TO SOLVE** this recurrence.

```
int spooky(int n) {
    if (n <= 10)
       return n + n;
    else {
       for (int i = n; i > 0; i--) {
         print i;
       }
       return 5 + spooky(n - 2) + n * spooky(n - 3);
    }
}
```

T(n) = _____ For n <= 10

T(n) = _____ For n > 10

Yipee!!!! YOU DO NOT NEED TO SOLVE *this* recurrence…

7. (11 pts) Solving Recurrences

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

$$T(1) = 9.$$

and

$$T(N) = 2 * T(N/2) + 7N \qquad \text{for integers } N > 1$$

Find the closed form for T(N). **You may assume that N is a power of 2**. Your answer should *not* be in Big-Oh notation – show the relevant ***exact*** constants in your answer (e.g. don't use "c1, c2" in your answer). You should not have any summation symbols in your answer. The list of summations on the last page of the exam may be useful. ***Show your work***.

8. (9 pts) B-Trees & Binary Min Heaps

a) Give a tight big-O running time of the worst case of a find operation on a **B-tree** (as described in lecture and in Weiss). Keep all factors of M, L, and N in your answer. Do not use any other variables in your answer. **Explain briefly how you got your answer.**

b) Given a **binary min heap** containing N values, calculate the maximum number of values that must be examined in order to find the 3 smallest values in the heap. If a value is examined more than once, only count it once. Assume that the heap does not contain duplicates and that N is much larger than 3. Assume that you have direct access to the heap array and the first value is stored at index 0. You should not modify the heap. Your answer should be **exact**, NOT in big-O.

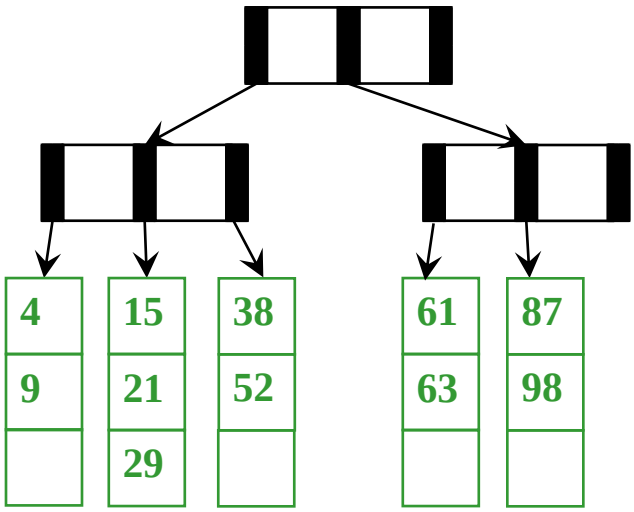
Maximum number of values that must be examined: _____

For any credit, briefly explain your answer:

9. (8 pts) B-trees

a) (1 pt) In the B-Tree shown below, **write in the values for the interior nodes**.

ORIGINAL:



After inserting 30:

b) (4 pts) Starting with the **ORIGINAL** B-tree shown above, in the box above, draw the tree resulting after inserting the value 30 (*including values for interior nodes*). Use the method for insertion described in lecture and in the book.

c) (3 pts) Starting with the **ORIGINAL** B-tree shown above, below, draw the tree resulting after deleting the value 87 (*including values for interior nodes*). Use the method for deletion described in lecture and in the book.

After deleting 87: