Name: _____

UWNetID: _____

CSE 332 Autumn 2016: 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	8	
6	5	
7	8	
8	10	
9	12	
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. For questions about hash tables, assume that no values have been deleted (lazily or otherwise).

You do not need to explain your answer.

a) $T(N) = T(N - 1) + 50$ _____

b) *Printing all elements in a **binary search tree** containing N elements from largest to smallest. (worst case).* _____

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

*d) Finding the maximum value in a **binary min heap** containing N elements. (worst case)* _____

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) *Creating a **binary min heap** from the elements in a **binary search tree** containing N elements (worst case).* _____

g) *Dequeue in a **FIFOqueue** containing N elements implemented using linked list nodes (worst case)* _____

h) *Finding the minimum value in an **AVL tree** containing N elements (worst case)* _____

i) $T(N) = 2 T(N/2) + N$ _____

j) *f(N)* = $\log_{10} (2^N)$ _____

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***

I.
```
void haunted(int n, int sum) {
    for (int i = 1; i < n * n; i++) {
       for (int j = 0; j < n; j += 3) {
          sum++;
       }
    }
    for (int k = 0; k < n * n; k++) {
       sum++;
    }
}
```
Runtime:

II.
```
int sweet(int n, int sum) {
    if (n < 5) {
       return sum;
    } else {
       for (int i = 0; i < n; i++) {
          sum++;
       }
    }
    return sweet(n-2, sum);
}
```

III.
```
int treat(int n) {
    if (n < 10) {
       return n;
    } else if (n < 100) {
       return n + 1;
    }
    return n * treat(n / 2) + treat(n / 2);
}
```

IV.
```
void spooky(int n, int sum) {
    int k = n;
    while (k > 0) {
       for (int i = 0; i < n * n; i++) {
          if (i % 2 == 0) {
             for (int j = 0; j < i; j++) {
                sum++
             }
          }
       }
       k--;
    }
}
```

3. (15 pts) Big-O, Big Ω, Big Θ

(2 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) $N \log (N^2)$ is:

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

b) $N \log (\log N)$ is:

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

c) $5^N \log N + N^4$ is:

$\Omega (N^5)$ $O (N^5)$ $\Theta (5^N)$ $\Theta (N^N)$

d) $N \log N + N \log^3 N$ is:

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

e) $N^{1/2} + \log N$ is:

$\Omega (\log N^3)$ $O (N^{1/2})$ $O (\log N)$ $\Omega (N^{1/3})$

4. (10 pts) Draw the AVL tree that results from inserting the keys 6, 2, 5, 7, 9, 8 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.***

5. (8 pts) Heaps & Trees -

a) (4 pts) Given a 3-heap of height h, what are the minimum and maximum number of nodes in the heap? Give your answer in closed form (there should not be any summation symbols). The list of summations on the last page of the exam may be useful.

Minimum:

Maximum:

b) (2 pts) Give a **post-order** traversal of the AVL tree you drew on the previous page.

c) (2 pts) Given an AVL tree of height 5, what is the **maximum** number of rotations we might have to do when doing an insert? Give an exact number, not a formula. A single rotation = 1 rotation, a double rotation = 2 rotations. **Explain your answer in one sentence.**

6. (5 pts) Recurrences

Give a base case and a recurrence for the runtime of the following function. Feel free to use c1, c2 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 boo(int n) {
     if (n < 2)
        return n + 5;
     else {
        for (int i = 0; i < 1000; i++) {
          print i
        }
        return n * boo(n / 2) + 4 * boo(n / 3);
     }
}
```

T(1) =

T(N) =

7. (8 pts) Solving Recurrences

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

$$T(1) = 5.$$

and

$$T(N) = 2 * T(N/2) + 9 \qquad \text{for integers } N > 1$$

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. (10 pts) B-Trees

a) (5 pts) Given M=7 and L=20, what is the minimum number of data items in a B-Tree (as defined in lecture and in Weiss) of height 4? Give a single number for your answer, not a formula. **Explain briefly how you got your answer.**

b) (5 pts) Given the following parameters for a B-tree with M= 22 and L = 5:

Key Size = 8 bytes

Pointer Size = 4 bytes

Data Size = 50 bytes per record (***includes*** the key)

Assuming that M and L were chosen appropriately, **what is the likely size of a disk block** on the machine where this implementation will be deployed? Give a numeric answer and **a short justification *based on two equations*** using the parameter values above.

9. (12 points) Data Structure Analysis

A new Aviation Management System is about to go live and they want an analysis of its data structures. The system is designed to keep track of planes that are due to arrive at all airports. The proposed data structure is an AVL Tree where the key is a unique airport number and the value is a binary min heap of planes due to arrive at that airport. The priority of each plane is the scheduled arrival time (where the arrival time is represented as the number of seconds since January 1, 2016 12am). Not all airports necessarily have any incoming planes scheduled.

A = total number of airports in the system

P = total number of planes currently in the system

Since you have no idea how P relates to A, you will need to keep all factors of P and A in your answers (e.g. you should NOT assume that P > A, or that either one is a constant). Assume that all heaps have sufficient capacity for new inserts.

a) (3 points) What would be the worst case big-O run-time of a **insertPlane(airport_num, plane)** operation on this data structure? If the given airport_num is not yet in the system, it will be added and then the plane will be added. **Explain your answer briefly.**

b) (3 points) Assuming that our data structure is up to date, and any planes that have arrived have been removed from the system, what would be the worst case big-O run-time of a **findNextPlane(airport_num)** operation on this data structure (finds the next plane due to arrive at the given airport but does not remove it)? **Explain your answer briefly.**

c) (3 points) Every once in a while the data structure gets out of synch and we need to fix things (hey, we are only in Beta). What would be the worst case big-O run-time of an operation that removes all planes whose arrival time is earlier than the specified time: **removeOldPlanes**(time)? **Explain your answer briefly.**

d) (3 points) There is a secondary design idea for the system which involves using a sorted array instead of an AVL tree to hold the airports. What would be the worst case big-O run-time of **insertPlane(airport_num, plane)** as described above if this secondary design is used instead? **Explain your answer briefly.**