Name: ____

Email address: ____

CSE 373 Autumn 2011: Midterm #1

(closed book, closed notes, NO calculators allowed)

Instructions: Read the directions for each question carefully before answering. We may give partial credit based on the work you **write down**, so if time permits, show your work! Use only the data structures and algorithms we have discussed in class or which 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: 73 points. Time: 50 minutes.

Question	Max Points	Score
1	16	
2	8	
3	17	
4	8	
5	6	
6	18	
Total	73	

1. (16 pts) **Big-O**

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. **You MUST choose your answer from the following** (not given in any particular order), each of which could be re-used (could be the answer for more than one of a) – h)):

$O(N^2)$, $O(N^{1/2})$, $O(N^{1/4})$, $O(\log^3 N)$, $O(N)$, $O(N^2 \log N)$, $O(N^5)$, $O(2^N)$, $O(N^3)$, $O(N^8)$, $O(\log^4 N)$, $O(N \log^3 N)$, $O(N^2 \log^2 N)$ $O(\log N)$, $O(1)$, $O(N^4)$, $O(N^N)$, $O(N^6)$, $O(N \log^2 N)$, $O(\log \log N)$

You do not need to explain your answer.

a) $f(N) = N^{1/2} + \log^3 N$ ____________________

b) $f(N) = 100 \log N + N^{1/4}$ ____________________

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

d) $f(N) = (N\cdot (100N + 5 + N^3))^2$ ____________________

e) $f(N) = 1000 \log \log N + \log N$ ____________________

f) $f(N) = \log_{16}(2^N)$ ____________________

g) $f(N) = N^2 \cdot (\log N^3 – \log N)+ N^2$ ____________________

h) $f(N) = (N \log N + 2N)^2$ ____________________

2. (8 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`. ***Showing your work is not required*** (although showing work ***may*** allow some partial credit in the case your answer is wrong – don't spend a lot of time showing your work.). You MUST choose your answer from the following (not given in any particular order), each of which could be re-used (could be the answer for more than one of I. – IV.):

   $O(n^2)$, $O(n^3 \log n)$, $O(n \log n)$, $O(n)$, $O(n^2 \log n)$, $O(n^5)$, $O(2^n)$, $O(n^3)$,
   $O(\log n)$, $O(1)$, $O(n^4)$, $O(n^n)$, $O(n^6)$, $O(n^8)$, $O(n^7)$

I.
```
void sunny(int n, int x) {
     for (int k = 0; k < n; ++k)
        if (x < 50) {
           for (int i = 0; i < n; ++i)
              for (int j = 0; j < i; ++j)
                 System.out.println("x = " + x);
        } else {
           System.out.println("x = " + x);
        }
}
```

Runtime:

II.
```
void warm(int n) {
     for (int i = 0; i < 2 * n; ++i) {
        j = 0;
        while (j < n) {
           System.out.println("j = " + j);
           j = j + 5;
        }
     }
}
```

III.
```
int silly(int n, int m) {
     if (n < 1) return m;
     else if (n < 10)
       return silly(n/2, m);
     else
       return silly(n - 2, m);
}
```

IV.
```
void happy(int n) {
      for (int i = n*n; i > 0; i--) {
         for (int k = 0; k < n; ++k)
             System.out.println("k = " + k);
         for (int j = 0; j < i; ++j)
            System.out.println("j = " + j);
         for (int m = 0; m < 5000; ++m)
           System.out.println("m = " + m);
      }
  }
```

3. (17 pts total) **Trees**.

a) (4 pts) What is the minimum and maximum number of nodes in a **complete tree of height 6?**
(Hint: the height of a tree consisting of a single node is 0)
Give an exact number (with no exponents) for both of your answers – not a formula.

Minimum =

Maximum =

b) (2 pts) What is the minimum number of nodes in a balanced **AVL tree of height 4?**
Give an exact number (with no exponents) for your answer – not a formula.

Minimum =

c) (2 pts) What is the maximum number of **leaf** nodes in a **binary tree of height h?**
Give a formula in terms of h for your answer.

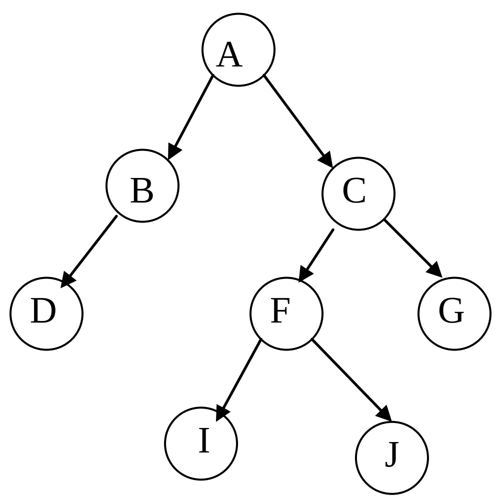
Maximum =

d) (1 pt) What is the **height** of node C in the tree shown below:

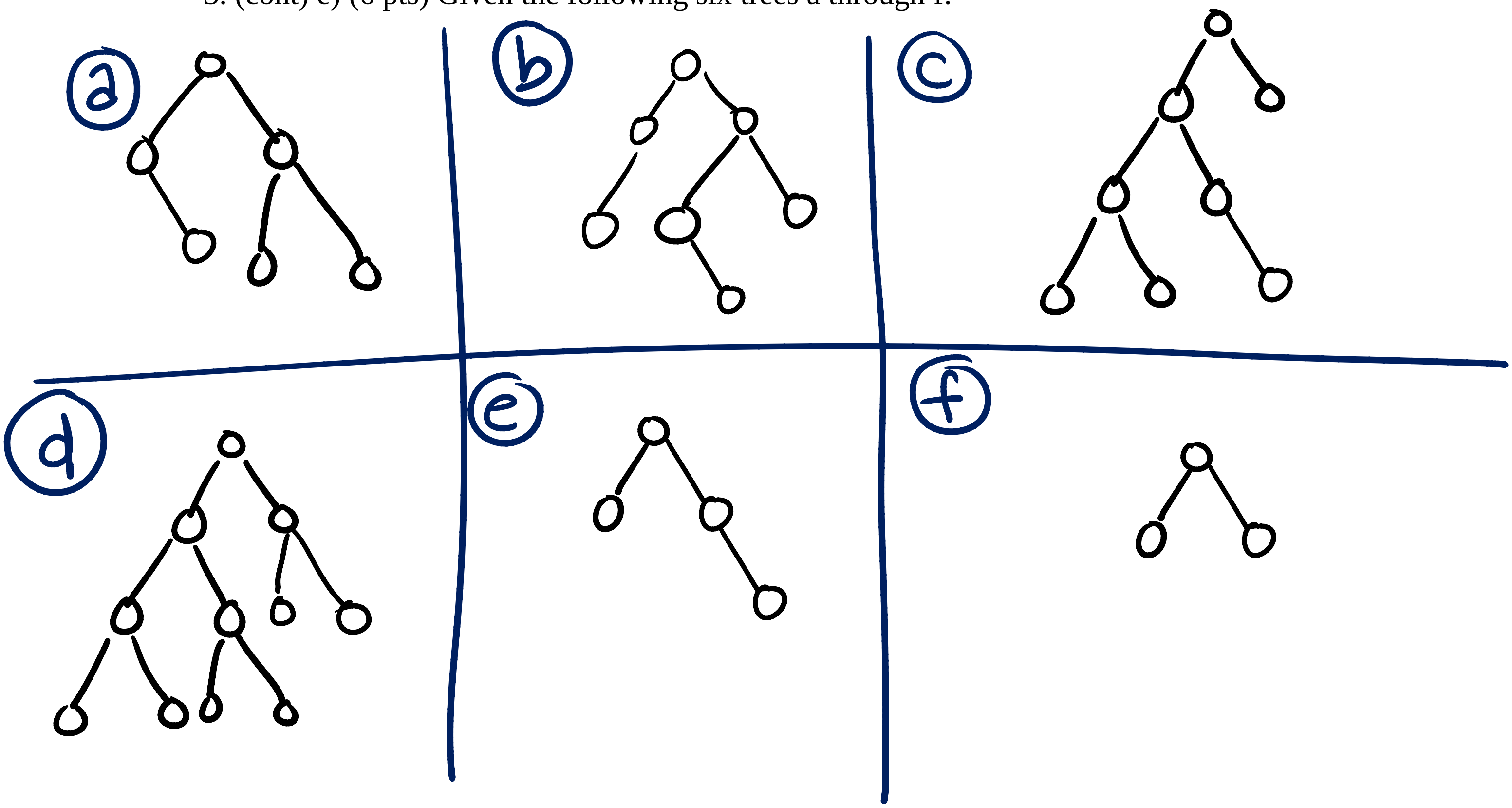
e) (1 pt) Give a Post-Order traversal of the tree shown below:

f) (1 pt) Is it AVL balanced (ignore the values, only look at the shape):

YES / NO



3. (cont) e) (6 pts) Given the following six trees a through f:



List the letters of all of the trees that have the following properties: (Note: It is possible that none of the trees above have the given property, it is also possible that some trees have more than one of the following properties.)

Complete: ________________________

Full: ________________________

AVL balanced: ________________________

Perfect: ________________________

4. (8 pts total) **Binary Min Heaps**

(a) (6 pts) Draw the binary min heap that results from inserting 3, 4, 8, 7, 2, 6, 9, 5, 1 in that order into an initially empty binary min heap. *You do not need to show the array representation of the heap*. You are only required to show the final tree, although drawing intermediate trees may result in partial credit. If you draw intermediate trees, please **circle your final result** for any credit.

4. **(cont.)**

(b) (2 pts) Draw the result of one deletemin call on your heap drawn at the end of part (a).

5. (6 pts) **AVL Trees** Draw the AVL tree that results from inserting the keys: 1, 9, 5, 3, 6, 2, 4 in that order into an initially empty AVL tree. You are only required to show the final tree, although drawing intermediate trees may result in partial credit. If you draw intermediate trees, ***please circle your final tree*** for any credit.

6. (18 pts) **Algorithms & Running Time Analysis:**

- **Describe the most time-efficient way to implement the operations listed below**. Assume no duplicate values and that you can implement the operation as a member function of the class – with access to the underlying data structure, including knowing the number of values currently stored (N).
- Then, give the tightest possible upper bound for the ***worst case*** running time for each operation in terms of *N*. ****For any credit, you must explain *why* it gets this worst case running time.** You must choose your answer from the following (not listed in any particular order), each of which could be re-used (could be the answer for more than one of a) -f)).

  $O(N^2)$, $O(N^{½})$, $O(N^3 \log N)$, $O(N \log N)$, $O(N)$, $O(N^2 \log N)$, $O(N^5)$, $O(2^N)$, $O(N^3)$, $O(\log N)$, $O(1)$, $O(N^4)$, $O(N^{12})O(N^N)$, $O(N^6)$, $O(N^8)$, $O(\log \log N)$

a) Pushing a value onto a stack implemented as an array. Assume the array is of size 2N.
Explanation:

a)

b) Print out all leaf nodes in an AVL tree in descending order (from largest to smallest).
Explanation:

b)

c) Popping a value in a stack implemented as linked list. Be specific in explaining how you get the runtime you provide. **Explanation:**

c)

6. (continued) **Algorithms & Running Time Analysis:**

- **Describe the most time-efficient way to implement the operations listed below**. Assume no duplicate values and that you can implement the operation as a member function of the class – with access to the underlying data structure, including knowing the number of values currently stored (N).
- Then, give the tightest possible upper bound for the ***worst case*** running time for each operation in terms of *N*. ****For any credit, you must explain *why* it gets this worst case running time.** You must choose your answer from the following (not listed in any particular order), each of which could be re-used (could be the answer for more than one of a) -f)).

  $O(N^2)$, $O(N^{½})$, $O(N^3 \log N)$, $O(N \log N)$, $O(N)$, $O(N^2 \log N)$, $O(N^5)$, $O(2^N)$, $O(N^3)$, $O(\log N)$, $O(1)$, $O(N^4)$, $O(N^{12})O(N^N)$, $O(N^6)$, $O(N^8)$, $O(\log \log N)$

d) Given a binary min heap, find which value is the *minimum* value and delete it.
Explanation:

d)

e) Given a FIFO queue, find which value is the *minimum* value and delete it. When you finish, the rest of the values should be left in their original order. **Explanation:**

e)

f) Given a binary search tree, find which value is the *median* value, and delete that value. **Explanation:**

f)