Name: _____

Email address: _____

CSE 373 Sample Midterm #1

(closed book, closed notes, calculators o.k.)

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 which were mentioned in the book so far.

Note: For questions where you are drawing pictures, please circle your final answer for any credit. There is one extra page at the end of the exam that you may use for extra space on any problem. If you detach this page it must still be turned in with your exam when you leave.

Advice You have 50 minutes, **do the easy questions first**, and work quickly!

1. **(16 pts) Big-Oh and Run Time Analysis:**

a. (10 points total) Describe the running time of the following pseudocode in Big-Oh notation in terms of the variable n. Assume all variables used have been declared. ***Show your work for partial credit.***

```
int foo(int k) {
   int cost;
   for (int i = 0; i < k; ++i)
     cost = cost + (i * k);
   return cost;
}
```

I.
```
answ = foo(n);
```

II.
```
int sum;
for (int i = 0; i < n; ++i) {
   if (n < 1000)
     sum++
   else
     sum += foo(n);
}
```

III.
```
for (int i = 0; i < n + 100; ++i) {
   for (int j = 0; j < i * n ; ++j){
     sum = sum + j;
   }
   for (int k = 0; k < n + n + n; ++k){
     c[k] = c[k] + sum;
   }
}
```

IV.
```
for (int j = 4; j < n; j=j+2) {
   val = 0;
   for (int i = 0; i < j; ++i) {
     val = val + i * j;
     for (int k = 0; k < n; ++k){
       val++;
     }
   }
}
```

V.
```
for (int i = 0; i < n * 1000; ++i) {
   sum = (sum * sum)/(n * i);
   for (int j = 0; j < i; ++j) {
     sum += j * i;
   }
}
```

1. (cont.) **Big-Oh and Run Time Analysis**

   b. (6 pts) Consider the following function:

```
int mystery(int n) {

   int answer;
   if (n > 0) {
      answer =(mystery(n-2)+3*mystery(n/2) + 5);
      return answer;
   }
   else
     return 1;
}
```

Write down the complete recurrence relation, T(n), for the running time of `mystery(n)`. Be sure you include a base case T(1). **You do not have to actually solve this relation, just write it down.**

2. (9 points) **Trees** –

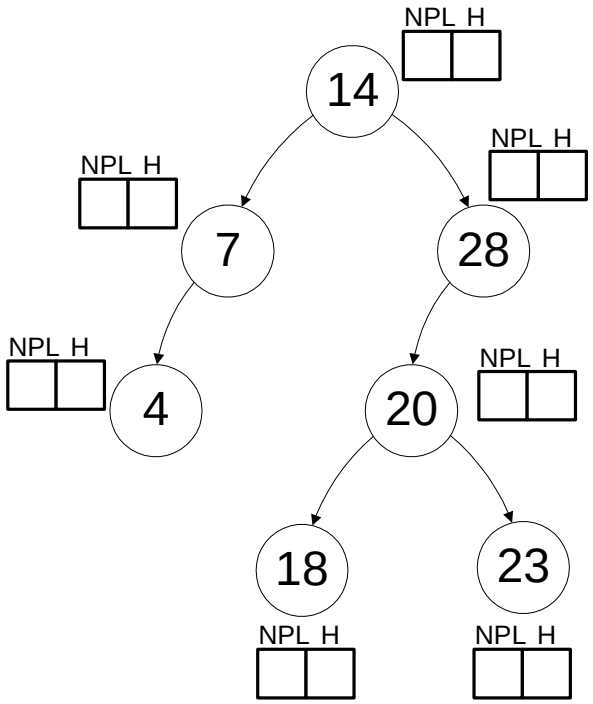
a) What is the minimum and maximum number of nodes in a ***complete*** binary tree of height h?

b) What is the minimum and maximum number of nodes in a ***perfect*** binary tree of height h?

c) What is the AVL balance property?

3. **(18 pts)**

a.) (10 pts) Mark the following properties for each node of the tree below in the space indicated for each node: **Height (H) (Ignore NPL)**



b.) (8 pts) Also, circle **yes** or **no** to indicate whether the tree above might represent each of the following data structures. If you circle **no,** give **one specific reason** why the tree could **not** be that data structure.

- AVL tree yes no
- Splay tree yes no
- Binary tree yes no
- Binary min heap yes no

4. (16 pts) Running Time Analysis

Give a O-bound on the *worst case* running time for each of the following in terms of *n.* **No explanation is required**, but an explanation may help for partial credit. Assume that all keys are distinct.

(a) insert in an *AVL tree* of size *n*

(b) insert in a *splay tree* of size *n*

(c) deleteMin in a *binary min heap* of size *n*

(d) Floyd's `buildHeap` to build a *binary heap* of *n* elements

5. a. (8 pts) Draw the AVL tree that results from inserting the keys 4, 10, 3, 8, 5, 6, 25 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.***

b. (2 pts) Give a *preorder* traversal of your final AVL tree you created in part a.

c. (8 pts) Draw the Splay tree that results from inserting the keys 4, 9, 3, 7, 5, 6 in that order into an initially empty Splay tree. You are only required to show the final tree, although if you draw intermediate trees, ***please circle your final result for ANY credit.*** You may continue your answer to this problem on the next page if needed.

d. (2 pts) Give a *postorder* traversal of your final Splay tree you created in part c.

6. **(20 pts) Heaps**

   a. (8 pts) Draw the binary min heap that results from inserting 11, 9, 12, 14, 3, 15, 7, 8, 1 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.***

6. (cont.) **Heaps:**
b. (4 pts) Draw the binary heap that results from doing 2 deletemins 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.***

7. Binary search trees

a) Insert the following values into a binary search tree in this order: 5, 9, 2, 1, 4, 8, 3, 7, 6

b) Show the tree you created above after deleting the value 4 using one of the (two) methods described in lecture. (do not use lazy deletion)

8. More Big-O – True or false

   - $3000N + 36 = O(N)$
   - $N \log N + N/5 = O(N^2)$
   - $N \log N + N^2 = O(N \log N)$
   - $N^2 \log N + N^2 = O(N^2)$
   - $N^{1/2} + \log N = O(\log N)$