CSE 373, Fall 98 Name: ______________________

MIDTERM EXAM

This midterm is closed-book, closed notes. The point value of each of the questions is indicated. The points add up to 35, so budget your time accordingly. Don't spend too much time on any single question.

Keep your answers short and to-the-point. Concise, direct answers will receive more credit than longer, essay-like answers. **If you find a question ambiguous, state your assumptions precisely, and proceed.**
Whenever possible, express your answer using big-oh notation.
Remember this test is graded on "a curve" so if you can't answer every question–just do your best.

Good luck!

1. (5 points) What ADT or data structure would you use for the following:
   (a) to invert the contents of an array with n elements in $O(n)$ time and $O(n)$ space?
   Stack
   (b) to represent a perfect binary tree with maximal memory efficiency?
   Nodes with two child pointers. The question intended to ask about a **complete** *binary tree, in which case the answer would be an array.*
   (c) to efficiently check whether a long list of names contains any duplicates.
   Hash Table
   (d) to store a standard printer queue, where print job priority depends only on job arrival time.
   Queue
   (e) to represent a very sparse one-dimensional array?
   Linked list
2. (3 points) Are the following statements true or false:
   (a) If $f(n) = O(n^3)$, then there is no constant c such that $\forall n > 0 : \; c * f(n) > n^3$.
   False. Counterexample: $c = 2$.
   (b) Both the Queue and Stack ADTs can be implemented on top of array or linked list data structures.
   True.
   (c) There is *no* way to make sure that the depth of a binary search tree, in the worst case, is $O(\log n)$ where n is the number of nodes in the tree.
   False. AVL trees have this property.

3. (4 points) Suppose you are asked to insert the following items in order into an AVL tree. Draw the resulting tree: 5, 4, 3, 2, 1:

```
      4
     / \
    2   5
   / \
  1   3
```

4. (1 points) What is the advantage of closed hashing over open hashing (also known as seperate chaining)?

   Open hashing wastes space by storing pointers as well as data. Also, open hashing operations may be slower in practice, especially those involving memory allocation.

5. (3 points) Define a binary heap in 3 or 4 words. What data structure is used to implement a heap? Can a binary search tree be implemented using the same data structure without memory inefficiency?

   A binary heap is a type of priority queue. While conceptually is is a complete binary tree, it is usually implemented as an array. A binary search tree can support the same operations as a heap, but is more complex and may require more storage for the extra pointers.

6. (2 points) Suppose we are using double hashing for collision resolution, and our second hash function is $h_2(x) = x^2 \pmod 7$. What is the main problem with this hash function?

   If $x^2 \equiv 0 \pmod 7$ *then collisions will lead to an infinite loop.*

7. (3 points) If you are using linear probing and the load factor on your hash table is almost 1, what is the time complexity of the following operations: INSERT, DELETE, FIND?

   All are $O(n)$ because we may need to scan the entire table in order to find an empty spot (for INSERT) or the specified element (for DELETE and FIND).

8. (2 points) What is the ratio between the number of internal (non-leaf) nodes and the number of leaf nodes in a perfect binary tree?

   If there are n internal nodes, then there are $n + 1$ leaves, so the ratio is about 1:1.

9. (4 points) List the primary disadvantages of the following techniques: open hashing, linear probing, quadratic probing, and double hashing. Hint: Each technique has one primary disadvantage which can be described in a sentence. If you are uncertain as to which is the primary disadvantage, feel free to list at most two.

   - *Open Hashing requires lots of extra memory*
   - *Linear Probing suffers from primary clustering*
   - *Quadratic Probing may fail on tables which are not full*
   - *Double Hashing requires calculation of a second hash function*

10. (8 points) If you are required to represent a subset of the set of integers 1, 2, $\ldots n.$, please fill out the following table, contrasting three possible implementations: Sorted Linked List, an AVL Tree, and a Mystery Implementation that takes $O(1)$ for FIND, INSERT, and DELETE. Please explain what mystery data structure you chose in the space provided.

    For example, the union of $\{a, b\}$ and $\{b, c\}$ is $\{a, b, c\}$, and their intersection is $\{b\}$.

There are two possible choices for MYSTERY: either a hash table with m cells or an array A of booleans where $A[i]$ is true if i is in the set or false otherwise. In this case, m would be the maximum number allowed in the set.
The running times below for INTERSECTION and UNION assume both sets are of size $O(n)$.

	MYSTERY	SORTED LINKED LIST	AVL TREE
FIND	$O(1)$	$\mathrm{O}(n)$	$\mathrm{O}(\log n)$
INSERT	$O(1)$	$\mathrm{O}(n)$	$\mathrm{O}(\log n)$
DELETE	$O(1)$	$\mathrm{O}(n)$	$\mathrm{O}(\log n)$
UNION	$\mathrm{O}(m)$	$\mathrm{O}(n)$	$\mathrm{O}(n \log n)$
INTERSECTION	$\mathrm{O}(m)$	$\mathrm{O}(n)$	$\mathrm{O}(n \log n)$