CSE 326: Data Structures
Practice Problems/Suggestions for Midterm
April 21, 1999

**Global Comments:**

- Material to be covered: everything through Friday, May 7. Weiss, Chapters 2– 5, plus handout on splay trees.
- The exam will be closed books, closed notes, no calculators.
- Be sure you know:
  - how to analyze the running times of simple programs, notions of worst-case and average-case complexity
  - how to compare functions by Big Oh, Big Omega and Big Theta.
  - tree traversals (preorder, inorder, postorder)
  - how the basic dictionary operations are implemented, what their worst-case (and when appropriate, average case) running times are and what the tradeoffs are using
    * unsorted and sorted lists
    * binary search trees (without balancing)
    * AVL trees
    * splay trees
    * B-trees
    * hashing (with separate chaining or the various open addressing)

**Practice Problems on Recent Material (not to be turned in):**

1. Weiss, p. 175, problem 4.42
2. Weiss, p. 204, problem 5.1.
3. Suppose we use a random hash function $h$ to hash $n$ distinct keys into an array $T$ of length $m$. What is the expected number of collisions? More precisely, what is the expected cardinality of $\{(x, y) | x \neq y \text{ and } h(x) = h(y)\}$? What is the worst-case number of collisions?
4. What is the advantage of picking a hash function from a universal class of hash functions over using a fixed hash function?
5. Consider a separate chaining hashing scheme that starts out with a table of size 100, rehashes the elements into a table twice the current size whenever the load exceeds threshold $\lambda_{hi}$, and rehashes the elements into a table half the current size, whenever the load drops below $\lambda_{lo}$ (and the table size is above 100). Assuming that the hash function being used distributes keys at random throughout the table, give values of $\lambda_{hi}$ and $\lambda_{lo}$ that guarantee that the total expected cost to perform $M$ operations (insert, delete, lookup, rehash) is $O(M)$. Explain your answer.
6. Weiss, p. 205, problem 5.8.

7. When you buy a ticket in the State Lottery, you choose six different numbers between 1 and 36. The lottery officials keep a dictionary keyed on the set of six numbers chosen on each ticket. After the officials pick the winning numbers, they access this dictionary to identify the winning ticket or tickets, if any. Since millions of tickets are sold, the officials have decided to keep the dictionary in external storage with a directory in an internal hash table. Their computer consultant, S.L. Ow, has recommended that they use the hash function

$$h(x_1, x_2, x_3, x_4, x_5, x_6) = (x_1 + x_2 + x_3 + x_4 + x_5 + x_6) \bmod m$$

where $m$ is the number of external buckets in which the records will be stored. Give a critique of this recommendation, and suggest a better alternative.

## A Midterm From a Previous Quarter

1. (4 points) Suppose you want to implement a Stack. State a reason that could cause you to choose the dynamic ("Linked") implementation rather than the static ("Contiguous", or array) implementation.

2. (5 points) Show the result of deleting the key 34 from the following binary search tree. (Use ordinary binary search tree deletion exactly as presented in lecture, not AVL or splay tree deletion.)

```
      12
     /  \
    7    34
        /  \
      31    61
     /     /  \
    18   48    80
   /  \    \
  15  20    52
           /  \
         49    56
```

3. (9 points) Consider the following AVL tree $T$.

```
        50
       /  \
     20     74
    /  \      \
  12    31     91
 /     /  \
5    23    33
```

(a) In the figure above, label each node in $T$ with its balance.

(b) Show the result of inserting the key 46 into the AVL tree $T$. (I recommend that you show your intermediate work, for the possibility of partial credit in case you make a mistake.)

4. (5 points) During AVL insertion, we argued that no ancestor of the critical node ever changes balance or height. Given this fact, explain the exact circumstances under which the overall height of the AVL tree can ever increase.

5. (16 points) Consider the following splay tree $T$.

```
          6
         / \
        5   13
       /   /
      4   11
     /   / \
    3   9   12
   /   / \
  2   7  10
 /     \
1       8
```

(a) Show the results of each of the rotations to splay the key 9 to the root. (I recommend that you show your intermediate work, for the possibility of partial credit in case you make a mistake.)

(b) In part (a), let $T_1$ be your splay tree after the first Case I, II, or III rotation, and $T_2$ be your splay tree after the second. Go back and label the nodes of the 3 trees $T$, $T_1$, and $T_2$ with their ranks.

(c) Suppose the Money Invariant holds for $T$, and suppose you were only paid \$2 to splay the key 9 in $T$. According to the proof of the Cost of Splay Steps Lemma, for which of your two rotations ($T$ to $T_1$, or $T_1$ to $T_2$) would you have to take \$1 from the tree, and from which node would the \$1 come?

(d) Exactly how would the \$2 you are paid for the splay in part (c) be used?

6. (5 points) In class, a student asked why we bother to splay during a LookUp: once you've located the key in the splay tree, you know the value LookUp will return without having to splay. One reason to splay during LookUp is to achieve the self-organizing behavior (locality of reference), but there is a much more important reason. Explain what it is, and support your answer with a concrete example.

7. (6 points) In the handout on splay trees, it was proven that you can delete key $K$ from a splay tree if you are paid $7\lfloor\log n\rfloor + 2$ dollars (in addition to the dollars that the Money Invariant states are already in the tree). Recall that the breakdown for this figure was $3\lfloor\log n\rfloor + 1$ dollars to splay on $K$, another $3\lfloor\log n\rfloor + 1$ dollars to splay the left subtree on $+\infty$, and $\lfloor\log n\rfloor$ extra dollars to invest in the new root because it takes on new descendents. Explain why $6\lfloor\log n\rfloor + 2$ dollars is in fact sufficient for a Delete.