CSE 326: Data Structures
Assignment #6
May 17, 1996
due: Friday, May 24

1. (a) Page 254, problem 27(a). Show the splay tree after each of the four double rotations. In your diagrams, label the subtrees that hang off this path $T_1, T_2, \ldots, T_{10}$ in the order encountered during an inorder traversal.
   (b) Page 254, problem 27(b). Show the splay tree after each of the four double rotation. In your diagrams, label the subtrees that hang off this path $T_1, T_2, \ldots, T_{10}$ in the order encountered during an inorder traversal.
   (c) For the splay operation of part (a), suppose that the rank of the node with key 90 does not increase during the first and fourth double rotations. Specify two nodes whose ranks decrease as a result of the splay, and prove that they do decrease. Thus, money from these two nodes can be used to pay for these two double rotations. (Hint: see the proof of the Cost of Splay Steps Lemma.)
   (d) Page 254, problem 27(d). Note that the reference to part (c) means the textbook's version of part (c). (Hint: see the proof of the Cost of Splay Steps Lemma.)

2. Insert the integers 13, 5, 22, 8, 34, 19, 21 (in this order) into an initially empty hash table of size 7 using the hash function $h(x) = x \bmod 7$,
   (a) using separate chaining.
   (b) using open addressing with double hashing, where $h_2(x) = 1$.
   (c) using open addressing with double hashing, where $h_2(x) = 1 + (x \bmod 6)$.

3. Page 296, problem 32. Justify your answers. In part (a), keep in mind that these are *ordered* pairs of *distinct* numbers. In part (b), delete the word "maximum": you are determining the exact probability. Compare your answer in part (a) to the upper bound $N(N-1)/m$ proved in the Theorem, and your answer in part (b) to the upper bound $1/m$.

4. Page 331, problem 1. Show the partially ordered tree after each Insert and after each DeleteMin. Also show the heap in table form after the last Insert.

5. Another useful heap operation is ChangePriority(i, K, h), which changes the key stored in heap entry $(\text{Table}(h))[i]$ to K.
   (a) Write an algorithm that implements the operation ChangePriority.
   (b) Show that your algorithm runs in time $O(\log n)$, where n is the number of entries in the heap.