Data Structures and Programming

Midterm Exam. Spring, 2007

1. (10 pts) Show the resulting tree if we search for 70 in the splay tree below and perform the appropriate splay. Show your work in sufficient detail.

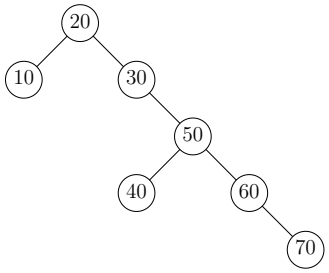


Figure 1: Tree 1

2. (18 pts) Consider the following AVL free. Answer the following two questions. You must show your derivations in detail in order to receive full credit.

   (a) Show the resulting tree after inserting 23 to the tree.

   (b) Show the resulting tree after deleting 91 from the original tree.

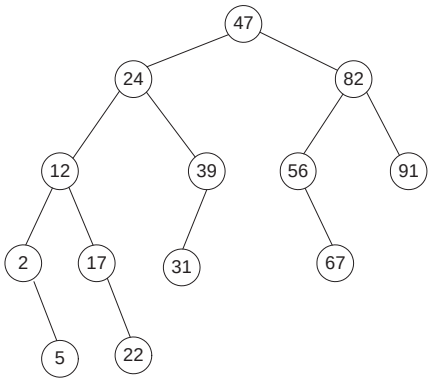


Figure 2: Tree 2

3. (8 pts) Suppose Tree 2 above is an ordered tree. Draw its binary tree representation (i.e., first-child/next-sibling representation).

4. (12 pts) Suppose you are given the following facts (procedures): preorder, inorder and postorder traversals of a binary tree of n nodes take $O(n)$ time; push and pop operations of a stack take $O(1)$ time; dequeue and enqueue operations of a queue take $O(1)$ time; insertion/deletion/search of an AVL tree take $O(logn)$ time; insertion/deletion/search of a splay tree may require $\Omega(n)$ in the worst case; finding the maximum/minimum/arbitrary element of an array takes $O(n)$ time; binary search of a sorted array takes $O(logn)$ time.

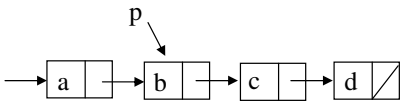
   (a) (6 pts) Can you use some of the above facts to design an efficient sorting algorithm? Explain your algorithm in sufficient detail. What is the running time of your algorithm?

   (b) (6 pts) Given two sets S_1 and S_2 of keys, design an algorithm (based on the above facts) to output (in increasing order) the set of keys that are in S_1 but not in S_2. (For instance, if $S_1 = \{2, 4, 6, 7, 8\}$ and $S_2 = \{2, 7\}$, then the output should be $4, 6, 8$.) Originally S_1 and S_2 are not sorted. What is the running time of your algorithm?

Note: Points received depend on the efficiency of your algorithm.

5. (6 pts) Write an algorithm (in Chinese or English) to print a singly linked list in reverse, using only constant extra space. (That is, if the list is $\rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow NIL$, then the output should be 5 4 3 2 1.) This means that you cannot use recursion. Assume that the linked list is constructed using dynamic memory allocation.

6. (6 pts) Suppose we have a pointer to a node in a singly linked list that is guaranteed not to be the last node in the list. We do not have pointers to any other nodes (except by following links). Describe an $O(1)$ algorithm (in Chinese or English) that logically removes the value stored in such a node from the linked list, maintaining the integrity of the linked list. For example, remove b (which is pointed to by p) from the following list.



7. (40 points) Answer the following questions. No explanations are needed.

   (a) (3 pts) Suppose that we start with an empty splay tree and insert n distinct keys. Then, in an arbitrary order, we search for each of these n keys. What is the total time (in big-Oh notation) required in the worst-case for this entire sequence of operations?

   (b) (3 pts) Suppose we define a variation on AVL trees in which the heights of the left and right subtrees of a node could differ by at most 2 (rather than at most 1). What recurrence relation would we get for the minimum number n_k of keys in a tree of height k?

   (c) (8 pts) Write in Θ-notation the amount of time used by each of the the code segments below.

   (1)
```
for (i = 0; i < n; i++) {
    for (j = 1; j < n; j = 3 * j) {
        x = x + 1;
    }
}
```

   (2)
```
sum = 0;
for (i = 0; i < n; i = i + 2) {
    for (j = n * n; j > 4; j = j - 1) {
        sum = sum + 1;
    }
}
```

   (d) (5 pts) Suppose that a certain binary tree on seven nodes has inorder traversal $PQRSTUV$ and postorder traversal $QRPTUVS$. Draw the tree.

   (e) (5 pts) Below is a post-order traversal of a binary search tree. Draw the tree.

   12 9 24 17 40 49 38

   (f) (6 pts) Given an AVL tree of n nodes. What is the number of rotations needed in the worst case to re-balance the tree when a (1) *deletion* (2) *insertion* is carried out?

   (g) (4 pts) Draw an AVL tree of height 3 with a minimum number of nodes.

   (h) (6 pts) Suppose T_1 and T_2 are two binary search trees of heights h_1 and h_2, respectively, and the largest key in T_1 is smaller than the smallest key in T_2. Can you design an algorithm in $O(h_1+h_2)$ time to combine the two trees into a single binary search tree? Explain why your algorithm takes $O(h_1 + h_2)$ time.