

CSE 150 Fall 2009: Homework #4

Note: If you don't know the proper notation for something, just describe it as well as you can.

Problem 1

Solve the following recurrence relations (in all cases, $T(1) = 0$). You only need to find $\Theta()$ solutions, not exact solutions.

- $T(n) = T(\lfloor n/2 \rfloor) + 1$
- $T(n) = 2T(\lfloor n/2 \rfloor) + 1$
- $T(n) = 4T(\lfloor n/2 \rfloor) + 1$
- $T(n) = T(n - 1) + 1$
- $T(n) = 2T(n - 1) + 1$
- $T(n) = T(\lfloor n/2 \rfloor) + n$
- $T(n) = 4T(\lfloor n/2 \rfloor) + n$

Problem 2

Suppose you have a two-dimensional matrix $A[0, \dots, n - 1][0, \dots, m - 1]$ such that $A[i][j] \leq A[i][j + 1]$ and $A[i][j] \leq A[i + 1][j]$ for all i and j . Find the fastest algorithm you can to determine whether a given number x occurs in A .

Problem 3

Write two recursive functions that operate on linked lists:

- $\text{insert}(L, x)$. If L is sorted, then $\text{insert}(L, x)$ returns a sorted linked list containing all the elements of L , and x . For example, $\text{insert}([1, 3, 4], 2) = [1, 2, 3, 4]$ and $\text{insert}([1, 2, 3], 1) = [1, 1, 2, 3]$.
- $\text{sort}(L)$ returns a sorted version of L . For example, $\text{sort}([2, 1, 3, 0]) = [0, 1, 2, 3]$ and $\text{sort}([]) = []$.

What are the running times of your algorithms?

Problem 4

Let T be a full binary tree (i.e. a binary tree in which each node has 0 or 2 children) with n leaves. How many internal nodes are there? Prove your answer.

Problem 5

- Write an algorithm to remove the smallest element from a binary search tree. You should fix up the tree so that it is still a binary search tree and no elements get lost.
- Write an algorithm to remove the root of a binary search tree. Hint: you need to replace the root with a new root? use your algorithm from above to get the new root.