CS 325 - Algorithms - Summer 2014 - Assignment 2

Reading: Chapter 0 and Chapter 2

Directions: One or two of the questions on this assignment will appear on the quiz on Thursday, July 3rd.

1. Exercise 0.1 in *Algorithms*.
2. Solve the following recurrence relations (if this question appears on the quiz, I would like to see your trees and work):
   (a) $T(n) = T(\frac{n}{3}) + O(1)$
   (b) $T(n) = T(\frac{n}{3}) + O(n)$
   (c) $T(n) = 3T(\frac{n}{3}) + O(n)$
   (d) $T(n) = 2T(\frac{n}{3}) + O(n)$
   (e) $T(n) = 5T(\frac{n}{3}) + O(n^2)$
3. Exercise 2.4 in *Algorithms*.
4. Exercise 2.14 in *Algorithms*.
5. Suppose that you are given a sorted list of distinct integers $\{a_1, a_2, \ldots a_n\}$. Give a divide-and-conquer algorithm that determines whether there exists an index i such that $a_i = i$. For example, in $\{-10, -1, 3, 40, 41\}$, $a_3 = 3$, and in $\{4, 7, 19, 20\}$ there is no such i. Prove that this algorithm is correct. State the running time of your algorithm.
6. Give a divide-and-conquer algorithm for finding the kth largest element in the merge of two sorted sequences S_1 and S_2. Prove that this algorithm is correct. State the running time of your algorithm.
7. Suppose you are given k sorted arrays, each with n elements, and you want to combine these arrays into a single sorted array with kn elements. Design a divide-and-conquer algorithm that will do this and prove that this algorithm is correct. State the running time of your algorithm.
8. The *Towers of Hanoi* puzzle is defined as follows: Given a stack of n disks arranged from largest on the bottom to smallest on top placed on a rod, together with two empty rods, the Towers of Hanoi puzzle asks for the minimum number of moves required to move the stack from one rod to another, where moves are allowed only if they place smaller disks on top of larger disks. Give an algorithm to solve this puzzle and prove that this algorithm is correct. State the running time of your algorithm.
9. Suppose that you are given a stack of n pancakes of different sizes. You want to sort the pancakes so that smaller pancakes are on top of larger pancakes. The only operation that you can perform is a *flip*-insert a spatula under the top k pancakes and flip them all over. Give an algorithm to sort the pancake stack and prove that this algorithm is correct. State the running time of your algorithm.