

CMPS 201: Summer 2003

Homework Assignment 4

1. (20 Points) Prove the correctness of RadixSort by (finite) induction on i , the column being sorted. Be sure to carefully state your induction hypothesis. It should be clear from your proof why the sort on line 2 must be stable, and why the algorithm must sort the digits from least to most significant.

RadixSort(A, d) (Pre: $A[1..n]$ consists of d digit numbers)

1. for $i \leftarrow 1$ to d
2. sort A on digit i using a stable sort

2. (20 Points) Prove that Counting Sort is stable. Also show that if we reverse the order in which the final loop is executed, the resulting algorithm is correct, but not stable. (This is problem 8.2-3 on page 170).
3. (20 Points) Consider the problem of determining an integer m in the range $1 \leq m \leq 10^9$ by asking a sequence of questions which have at most 5 possible answers. (For instance, you may ask to which of 5 subintervals m belongs.) Give a decision tree lower bound on the worst case number of questions which must be asked by any algorithm which solves this problem.
4. (20 Points) **Bar Weighing Problem:** Assume we are given 12 gold bars numbered 1 to 12 where 11 bars are pure gold and one is counterfeit: either gold-plated lead (which is heavier than gold), or gold-plated tin (lighter than gold). The problem is to find the counterfeit bar and what metal it is made of using only a balance scale. Any number of bars can be placed on each side of the scale, and each use of the scale produces one of three outcomes, indicating either that both sides are the same weight, or which of the two sides is heavier.
 - a. (10 Points) Give a decision tree argument to establish a lower bound on the (worst case) number of weighings which must be performed by any algorithm which solves this problem.
 - b. (10 Points) Design an algorithm which solves this problem with the fewest possible (worst case) number of weighings. You may present your algorithm by drawing a decision tree, rather than pseudo-code.
5. (20 Points) **Water Jug Problem** (problem 8-4 on page 179): Suppose that you are given n red and n blue water jugs, all of different shapes and sizes. All red jugs hold different amounts of water, as do the blue ones. Moreover, for every red jug, there is a blue jug that holds the same amount of water, and vice versa. It is your task to find a grouping of the jugs into pairs of red and blue jugs that hold the same volume of water. To do so, you may perform the following operation: pick a pair of jugs in which one is red and one is blue, fill the red jug with water, and then pour the water into the blue jug. This operation will tell you whether the red or the blue jug can hold more water, or if they are of the same volume. Assume that such a comparison takes one time unit. Your goal is to find an algorithm that makes a minimum number of comparisons to determine the grouping. Remember that you may not directly compare two red jugs or two blue jugs.
 - a. (10 Points) Describe an algorithm that uses $\Theta(n^2)$ comparisons to group the jugs into pairs.
 - b. (10 Points) Prove a lower bound of $\Omega(n \log n)$ for the worst case number of comparisons an algorithm solving this problem must perform.