

1. **Problem 5.6:** Consider an n -node complete binary tree T , where $n = 2^d - 1$ for some d . Each node v of T is labeled with a real number x_v . You may assume that these numbers are all distinct. A node v of T is a *local minimum* if the label x_v is less than the label x_w for all nodes w that are joined to v by an edge.

You are given such a complete binary tree T , but the labeling is only specified in the following *implicit* way: for each node v , you can determine the value x_v by *probing* the node v . Show how to find a local minimum of T by using only $O(\log n)$ *probes* to the nodes of T .

2. We are doing a simulation in which we look at n consecutive days of a given stock, GOOG, at some point in the past. Let's number the days $i = 1, 2, \dots, n$; for each day i , we have a price $p(i)$ per share for the stock on that day. Suppose during this time period, we wanted to buy 1000 shares on some day and sell all these shares on some (later) day. We want to know: what is the best time to buy and sell? That is, given one chance to buy and one chance to sell, how to maximize our profit? If we can't make money at all, we should report this. Show an $O(n \log n)$ algorithm for the problem.

3. **Problem 5.2:** Recall the problem of finding the number of inversions. As in the text, we are given a sequence of n numbers a_1, a_2, \dots, a_n , which we assume are all distinct, and we define an inversion to be a pair $i < j$ such that $a_i > a_j$.

We motivated the problem of counting inversions as a good measure of how different two orderings are. However, one might feel that this measure is too sensitive. Let's call a pair a *significant inversion* if $i < j$ and $a_i > 2a_j$. Give a $\Theta(n \log n)$ divide-and-conquer algorithm to count the number of significant inversions of an ordering.