1 Almost-Heap

An almost-heap is a binary tree that satisfies all heap-properties except that at one item the order-property may be violated. Thus, it consists of an array $A$ and one index $i$ such that $A[j] < A[parent(j)]$ for every $j \neq i$. In the figure below, the violated order-property is dashed.

Let $h$ be the height of the almost-heap, and let $l$ be the level that contains node $i$. Show how to turn an almost-heap into a heap in time $O((l + 1)(h - l + 1))$.

2 Sorting

Let $0 < \epsilon < 1$. Suppose that we have an array $A$ of $n$ items such that the first $n - n^\epsilon$ items are sorted among each other. Describe an $O(n)$ time algorithm to sort $A$. 
3 Bogo Sort

Give the best-case, worst-case, expected best-case, and expected worst-case running time for the following function. You can assume that the Shuffle operation requires $O(n)$ time. Note: The Shuffle() function produces each permutation equally likely.

Bogo(A):
    Shuffle(A)
    if A is sorted then
        return A
    else
        return Bogo(A)
    end if

4 Multi-Way Merge

Given a set of $k$ sorted arrays, where the combination of the $k$ arrays has $n$ elements in total, design an $O(n \log k)$ algorithm that produces a single sorted array containing all $n$ elements.