600.363 Introduction to Algorithms / 600.463 Algorithms I **Fall 2014**
Homework #2 **Due:** September 16, 2014, 1:30pm

Please start each problem on a new page, and include your name on each problem. You can submit on blackboard, under student assessment, or can bring your solutions to lecture.

Remember: you may work in groups of up to three people, but must write up your solution entirely on your own. Collaboration is limited to discussing the problems – you may not look at, compare, reuse, etc. any text from anyone else in the class. Please include your list of collaborators on the first page of your submission. You may use the internet to look up formulas, definitions, etc., but may not simply look up the answers online.

Please include proofs with all of your answers, unless stated otherwise.

1 Median of Sorted Arrays (33 points)

Let A and B be sorted arrays of n elements each. We can easily find the median of A or the median of B, since they are already sorted – it will be at index $\lfloor \frac{n+1}{2} \rfloor$. But what if we want to find the median element of $A \cup B$? We could just concatenate them and use the $O(n)$ time median algorithm, but is it possible to do better? In this problem you should give matching upper and lower bounds, i.e. you should find a function $f(n)$ and do the following:

(a) Design a deterministic algorithm whose running time (measured in terms of the number of comparisons) is $O(f(n))$, and

(b) Give a lower bound showing that any comparison-based algorithm must make $\Omega(f(n))$ comparisons in the worst case.

2 More Lower Bounds (33 pts)

Consider the following two-dimensional sorting problem: we are given an arbitrary array of n^2 numbers (unsorted), and have to output an $n \times n$ matrix of the inputs in which all rows and columns are sorted.

As an example, suppose $n = 3$ so $n^2 = 9$. Suppose the 9 numbers are just the integers $\{1, 2, \ldots, 9\}$. Then possible outputs include (but are not limited to)

```
1 4 7     1 3 5     1 2 6
2 5 8     2 4 6     3 4 8
3 6 9     7 8 9     5 7 9
```

It is obvious that we can solve this in $O(n^2 \log n)$ time by sorting the numbers and then using the first n as the first row, the next n as the second row, etc. For this question, you should prove a matching lower bound of $\Omega(n^2 \log n)$ in the comparison-based model. For simplicity, you can (as always) assume that n is a power of 2.

Hints: instead of reasoning directly about the decision tree, show that if we could solve this problem with $o(n^2 \log n)$ comparisons we could break the sorting lower bound. Useful facts to keep in mind are that $n! > (n/e)^n$ and that we can merge two sorted arrays of length n using $2n - 1$ comparisons.

3 Non-comparison sorting (33 pts)

(a) Suppose we are given an array of integers, but instead of all integers having the same length they can each have a different number of bits. So e.g. the number 0 or 1 takes one bit, the numbers $2, 3$ take 2 bits, the numbers $4, 5, 6, 7$ take three bits, etc. However, the *total* number of bits over *all* of the integers in the array is equal to n. Show how to sort the array in $O(n)$ time.

(b) Suppose now that we are given an array of strings (over some finite alphabet, say the letters `a-z`). Each string can have a different number of characters, but the total number of characters in all the strings (i.e. the sum over the strings of the length of the string) is equal to n. Show to sort the strings lexicographically in $O(n)$ time. Here lexicographic order is the standard alphabetic order, so for example `a` $<$ `ab` $<$ `b`.