601.433/633 Introduction to Algorithms **Fall 2025**
Homework #2 **Due:** September 29, 2025, 11:59pm

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 Jug Matching (33 points)

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 all the blue ones. Moreover, for every red jug there is a blue jug that holds exactly the same amount of water (and vice versa).

Your job is to find a matching between red jugs and blue jugs that hold the same amount of water. To do this, you are only allowed to use the following operation: pick a red jug and a blue jug, fill the red jug, and pour it into the blue jug. This will tell you whether the volume of the red jug is less than, equal to, or greater than the volume of the blue jug. In other words, you can compare any red jug and any blue jug. But you *cannot* compare two red jugs or two blue jugs.

Give a randomized algorithm that uses $O(n \log n)$ comparisons in expectation. Prove that your algorithm is correct and that it uses $O(n \log n)$ comparisons in expectation.

2 Costly Median (34 points)

Suppose that you are given n distinct numbers $x_1, x_2, \ldots, x_n \in \mathbb{R}^+$, each of which also has a *cost* $c_i \in \mathbb{R}^+$ so that $\sum_{i=1}^n c_i = 1$. The *costly median* is defined to be the number x_k such that

$$\sum_{i:x_i<x_k} c_i < \frac{1}{2} \qquad \text{and} \qquad \sum_{i:x_i>x_k} c_i \leq \frac{1}{2}.$$

Give a deterministic algorithm which finds the costly median and has $O(n)$ worst-cast running time (and prove correctness and running time).

3 More Lower Bounds (33 points)

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 that this is tight. Formally, you should prove that in the comparison model, any algorithm that solves this problem requires $\Omega(n^2 \log n)$ time. 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 there exists an algorithm making a number of comparisons that is not $\Omega(n^2 \log n)$ then we could break the sorting lower bound (we could sort an array of size n using fewer than $\log(n!)$ comparisons). 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. You might need to be careful with constants.