COMPSCI390.01: Algorithmic Foundations of Data Science
Midterm I

NAME:

Prob #	Score	Max Score
1		20
2		25
3		30
4		25
Total		100

Instructions:

1. If you write pseudocode, make sure to describe your idea in words.
2. Analyze the time complexity of your algorithms if asked.
3. You may use any algorithm covered in the class without detailed description, but you should be explicit about the input and output.
4. For any change that you make to an algorithm covered in class, you should describe the changes precisely.

Problem 1: (**20 pts**) Construct a binary Huffman encoding tree for the string

```
bubba ate a banana
```

and show its corresponding encoding.

Problem 2: (**25 pts**) Consider a special instance of *k-center* in $\mathbb{R}^1$ where we wish to partition a set $X = \{x_1, \ldots, x_n\}$ of n real numbers into 2 clusters so that the maximum distance between any point of X and the center of its assigned cluster is minimized. That is, compute a partition of X into two sets X_1 and $X_2 = X - X_1$, along with their centers c_1, c_2, respectively, such that the following objective function is *minimized*:

$$\max\{\max_{x \in X_1} |c_1 - x|, \max_{y \in X_2} |c_2 - y|\}.$$

Describe an $O(n \log n)$-time algorithm that achieves this goal, and briefly justify its time complexity and correctness. (**Hint:** *What property does a* 1*-dimensional optimal clustering have, and how do you use it to compute an optimal clustering efficiently?*)

Problem 3: (**30 pts**) We wish to construct the LSH functions with respect to the ℓ_1-metric in $\mathbb{R}^d$ for some fixed constant $d \geq 1$. (Recall $\|x - y\|_1 = \sum_{i=1}^{d} |x_i - y_i|$.)

Let $r > 0$ be a given integer, and let $c \geq 1$ be another integer. Consider building a randomly shifted d-dimensional grid with side length $\Delta = cr$, and define a hash function h that maps points lying within the same grid-cell to the same value. That is, we uniformly choose a random shift value $a = (a_1, \ldots, a_d) \in [\Delta]^d$ and for $x = (x_1, \ldots, x_d)$, set

$$h_a(x) = \left(\left\lfloor \frac{x_1 + a_1}{\Delta} \right\rfloor, \ldots, \left\lfloor \frac{x_d + a_d}{\Delta} \right\rfloor \right).$$

Consider two points $x = (x_1, \ldots, x_d)$ and $y = (y_1, \ldots, y_d)$.

(i) (**7 points**) What is the probability that a grid line in the i^{th} dimension separates x, y? In other words, what is the probability that the i^{th} coordinate of $h_a(x)$ and $h_a(y)$ differ, assuming $a = (a_1, \ldots, a_d)$ is chosen randomly from $[\Delta]^d$?

(ii) (**8 points**) Provide an upper bound of $\Pr_{a \sim [\Delta]^d}[h_a(x) \neq h_a(y)]$ using $\|x - y\|_1$ and other defined parameters. (**Hint:** *Use (i) and the union bound for probability.*)

(iii) (**15 points**) Let $\epsilon > 0$ be a fixed parameter. Obtain a lower bound for $\Pr_{a \sim [\Delta]^d}[h_a(x) = h_a(y) \mid \|x - y\|_1 \leqslant r]$ and an upper bound for $\Pr_{a \sim [\Delta]^d}[h_a(x) = h_a(y) \mid \|x - y\|_1 > (1 + \epsilon)r]$.

Problem 4: (**25pts**) Suppose we have two sets L and R, as well as two corresponding Bloom filters B_L, B_R storing L and R, respectively, that were constructed using the same hash function. Recall Bloom filters never have false negatives (if the query procedure for an item x returns no, then x is not in the set). Suppose the false positive rates (FPR) (i.e., the probability of the query procedure returning yes on an item x not in the set) of B_L, B_R are f_L, f_R, respectively.

(i) (**13 points**) Given B_L and B_R, how can one construct a Bloom filter $B_{L\cup R}$ that stores the union $L \cup R$ *without accessing individual items of the sets*? What is the runtime to construct $B_{L\cup R}$? Justify that $B_{L\cup R}$ never returns false negatives. Estimate its FPR.

(ii) (**12 points**) Suppose instead we want to construct a Bloom filter $B_{L\cap R}$ to store the intersection $L \cap R$. How will one construct it from B_L and B_R without having access to the set $L \cap R$? Does your construction return false negatives? Does your algorithm return the same Bloom filter as if one had constructed it directly for the set $L \cap R$ using the same hash function as for B_L and B_R (assuming we had access to the set $L \cap R$)? Justify your answers.