1. (30 pts.) 3/4-tree.

Consider a uniform rooted tree of height \( h \) (every leaf is at distance \( h \) from the root). The root, as well as any internal node, has exactly 4 children. Each leaf has a boolean value associated with it. An internal node returns 1 if and only at least three of its children evaluate to 1. The evaluation problem consists of determining the value of the root; at each step, an algorithm can choose one leaf whose value it wishes to read. This tree has \( n = 4^h \) leafs.

(A) (10 pts.) Consider a tree of height \( h = 1 \). Provide a randomized algorithm that in expectation (independent of the values stored in the 4 leafs!) evaluates strictly less than 4 leafs.

(B) (20 pts.) Extend the algorithm from (A) to evaluating a tree of larger depth. Describe your new algorithm, and prove that the expected number of leaves read by your algorithm on any instance is at most \( n^{0.954} \).

(For fun, you can try and prove that any deterministic algorithm has to read all leaves of the tree.)

2. (30 pts.) Concentrate.

Consider a full binary tree of height \( h \). You start from the root, and at every stage you flip a coin and go the left subtree with probability half (if you get a head), and to the right subtree with probability half (if you get a tail). You arrive to a leaf, and let assume you took \( k \) turns to the left (and \( h - k \) turns to the right) traversing from the root to this leaf. Then the value written in this leaf is \( k \), where \( \alpha < 1 \) is some parameter.

Let \( X_h \) be the random variable that is the returned value.

(A) (10 pts.) Prove that \( \mathbb{E}[X_h] = (1 + \alpha) \frac{h}{2} \) by stating a recursive formula on this value, and solving this recurrence.

(B) (10 pts.) Consider flipping a fair coin \( h \) times independently and interpret them as a path in the above tree. Let \( \mathcal{E} \) be the event that we get at most \( h/4 \) heads in these coin flips. Argue that \( \mathcal{E} \) happens if and only if \( X_h \geq \alpha^{h/4} \).

(C) (10 pts.) Markov’s inequality states that for a positive random variable \( X \) we have that \( \Pr[X \geq t] \leq \mathbb{E}[X]/t \). Let \( Y \) be the number of heads when flipping a fair coin \( h \) times. Using Markov’s inequality, (A) and (B) prove that

\[
\Pr[\text{Out of } h \text{ coin flips getting at most } h/4 \text{ heads}] \leq \left( \frac{1 + \alpha}{2\alpha^{1/4}} \right)^h.
\]

In particular, by picking the appropriate value of \( \alpha \), prove that

\[
\Pr[\text{Out of } h \text{ coin flips getting at most } h/4 \text{ heads}] \leq 0.88^h.
\]
What is your value of $\alpha$?

3. (40 pts.) Matrix search.

You are given a matrix $M$ of size $n \times n$. You can read any entry of the matrix in constant time. Furthermore, assume that $M[i][j] < M[i+1][j]$ and $M[i][j] < M[i][j+1]$ for any $i$ and $j$. (You can assume all the values in the matrix are distinct.)

(a) (5 pts.) Given an interval $I = [\alpha, \beta]$, we would like to figure out all the numbers in the matrix that fall in $I$. Given a row index $i$, describe how to compute, in $O(\log n)$ time, the two indices $l_i$ and $r_i$ such that all the numbers in the row $M[i]$ that fall in $I$ are $M[i][l_i] : : r_i$.

(b) (5 pts.) Compute $l_1, r_1, l_2, r_2, \ldots, l_n, r_n$ given $I = [\alpha, \beta]$ in linear time (i.e., $O(n)$ time).

(Hint: Draw the matrix and think about how the values of $l_i$ and $r_i$ change between the $i$th row and $(i+1)$th row.)

(c) (5 pts.) Given a value $\tau$, show how to compute the number of element in $M$ that are smaller than $\tau$ in $O(n)$ time.

(d) (25 pts.) Given $M$ and a number $k$, given an algorithm with expected running time $O(n \log n)$ that outputs the $k$th smallest element in $M$. Prove the correctness and running time of your algorithm.

Hint: Think about QuickSelect and how adapt it to this problem.