1. Recall that a priority search tree is a binary tree in which every node has both a search key and a priority, arranged so that the tree is simultaneously a binary search tree for the keys and a min-heap for the priorities. A heater is a priority search tree in which the priorities are given by the user, and the search keys are distributed uniformly and independently at random in the real interval [0, 1]. Intuitively, a heater is a sort of anti-treap.

The following problems consider an n-node heater T whose priorities are the integers from 1 to n. We identify nodes in T by their priorities; thus, “node 5” means the node in T with priority 5. For example, the min-heap property implies that node 1 is the root of T. Finally, let i and j be integers with 1 ≤ i < j ≤ n.

(a) What is the exact expected depth of node j in an n-node heater? Answering the following subproblems will help you:

i. Prove that in a random permutation of the (i + 1)-element set {1, 2, . . . , i, j}, elements i and j are adjacent with probability 2/(i + 1).

ii. Prove that node i is an ancestor of node j with probability 2/(i + 1). [Hint: Use the previous question!]

iii. What is the probability that node i is a descendant of node j? [Hint: Do not use the previous question!]

(b) Describe and analyze an algorithm to insert a new item into a heater. Analyze the expected running time as a function of the number of nodes.

(c) Describe an algorithm to delete the minimum-priority item (the root) from an n-node heater. What is the expected running time of your algorithm?

2. Suppose we are given a coin that may or may not be biased, and we would like to compute an accurate estimate of the probability of heads. Specifically, if the actual unknown probability of heads is p, we would like to compute an estimate  \( \hat{p} \) such that

\[
\Pr[|\hat{p} - p| > \varepsilon] < \delta
\]

where \( \varepsilon \) is a given accuracy or error parameter, and \( \delta \) is a given confidence parameter.

The following algorithm is a natural first attempt; here FLIP() returns the result of an independent flip of the unknown coin.

```
MEANESTIMATE(\( \varepsilon \)):
  count ← 0
  for i ← 1 to N
    if FLIP() = Heads
      count ← count + 1
  return count/N
```

(a) Let \( \hat{p} \) denote the estimate returned by MEANESTIMATE(\( \varepsilon \)). Prove that \( E[\hat{p}] = p \).
(b) Prove that if we set \( N = \lceil \alpha / \epsilon^2 \rceil \) for some appropriate constant \( \alpha \), then we have \( \Pr[|\hat{p} - p| > \epsilon] < 1/4 \). [Hint: Use Chebyshev’s inequality.]

(c) We can increase the previous estimator’s confidence by running it multiple times, independently, and returning the median of the resulting estimates.

\[
\text{MedianOfMeansEstimate}(\delta, \epsilon) : \\
\quad \text{for } j \leftarrow 1 \text{ to } K \\
\quad \text{estimate}[j] \leftarrow \text{MeanEstimate}(\epsilon) \\
\quad \text{return Median(estimate[1..K])}
\]

Let \( p^* \) denote the estimate returned by \( \text{MedianOfMeansEstimate}(\delta, \epsilon) \). Prove that if we set \( N = \lceil \alpha / \epsilon^2 \rceil \) (inside \( \text{MeanEstimate} \)) and \( K = \lceil \beta \ln(1/\delta) \rceil \), for some appropriate constants \( \alpha \) and \( \beta \), then \( \Pr[|p^* - p| > \epsilon] < \delta \). [Hint: Use Chernoff bounds.]