Ground Rules

- See HW1.
- All problems will be graded.

Problems

1. Let us modify binary search in the following way. Given a sorted list of \( n \) elements and a target to search for, instead of querying the median of the list, we compare the target against a random element, and then recurse. Prove that in the worst case the expected number of comparisons the algorithm makes is still \( O(\log n) \).

2. You are given a sorted circular linked list containing \( n \) integers, where every element has a “next” pointer to the next larger element. (The largest element’s “next” pointer points to the smallest element.) You are asked to determine whether a given target element belongs to the list. The only way you can access an element of the list is to follow the next pointer from a previously accessed element, or via the function RAND that returns a random element of the list. Develop a randomized algorithm for finding the target that makes at most \( O(\sqrt{n}) \) comparisons in expectation and always returns the correct answer.

3. Suppose I have two degree \( n \) polynomials, \( A \) and \( B \), with integer coefficients. I think that \( A \cdot B = C \) (for some other polynomial \( C \)) and want to efficiently verify that this identity holds. One way of doing so is to multiply \( A \) and \( B \) and compare the answer against \( C \). The standard way of doing so takes \( O(n^2) \) time, although \( O(n \log n) \) is possible (using FFT, for example). I want to verify the equation in \( O(n) \) time.

(a) Suppose that \( D \) and \( E \) are two distinct degree-\( k \) polynomials. Let \( p \) be a prime number larger than \( k \) and all of the coefficients in \( D \) and \( E \). Consider picking a number \( x \) uniformly at random from \( \{0, \ldots, p-1\} \), and evaluate both \( D(x) \) and \( E(x) \) modulo \( p \). What is the probability that you get the same answer? In other words, obtain an upper bound on \( \Pr[D(x) = E(x) \pmod p] \) over the random choice of \( x \).

\( \text{ Hint: How many roots can a degree } k \text{ polynomial have over the set } \{0, \ldots, p-1\} \pmod p? \text{ For example, the polynomial } x^2 + 3x + 2 \pmod 5 \text{ can be factorized as } (x+2)(x+1) \pmod 5 \text{ and therefore has two roots, namely } 3 \text{ and } 4, \text{ over the set } \{0, \ldots, 4\}. \)

(b) Use your answer to part (a) to design an \( O(n) \) time randomized algorithm for the problem of verifying the identity \( A \cdot B = C \) that returns the correct answer with probability at least \( 1/2 \). You may assume that basic arithmetic operations can be done in \( O(1) \) time.

\( \text{(Extra Credit)} \) Can you improve your algorithm from part (b) so that its error probability decreases to some small \( \epsilon > 0? \) What is the running time of your new algorithm in terms of \( n \) and \( \epsilon? \)