Directions: Use a separate page for each problem, and label each solution according to the problem attempted. Please make sure your pages are in order when you scan your work. Also make sure to upload your two-sided, handwritten notes page as well as any scratch work you used (but please make sure to clearly identify which work you want graded and which work is scratch).

You will have 90 minutes to complete this exam. Upon finishing, you may use up to 20 additional minutes to scan and upload your work to T-Square. You may not modify your exam once you have begun to scan and upload it. If you encounter any problems submitting your exam to T-Square, email your answers to `cpryby3@gatech.edu` within this 20-minute period.

1. In this problem, you will design a function `minDist` to solve the following problem.

   Input: A array of $n > 1$ integers, $A = [a_0, a_1, ..., a_{n-1}]$, sorted in increasing order

   Output: The smallest integer d satisfying $d = a_j - a_i$ for some pair of indices $j > i$

   Examples:

   - `minDist([1,2,3,4]) == 1` (obtained via $a_3 - a_2$, for example)
   - `minDist([0,1,1,2,3]) == 0` (obtained via $a_2 - a_1$)
   - `minDist([1,3,6,10,15]) == 2` (obtained via $a_1 - a_0$)

   (a) Describe the basic strategy of your algorithm in English.

   (b) Implement your algorithm in pseudocode.

   (c) Justify why your algorithm correctly solves the problem, using and proving a loop invariant if appropriate.

   (d) Analyze and give a tight estimate of your algorithm's worst-case performance, giving consideration to both runtime and auxiliary space.

2. You are given a *unimodal* array A of n distinct numbers; that is, there is an index p (called the *peak* index) such that the values in the array are increasing from position 0 to p (inclusive) and then decreasing from p to $n-1$ (inclusive). For example, the array $[-9, -8, 1, 2, 3, 0, -5]$ is unimodal with peak index $p = 4$.

   In this problem you will design and analyze a divide-and-conquer algorithm for finding the peak index p.

   (a) Describe the basic strategy of your algorithm in English.

   (b) Implement your algorithm in pseudocode.

   (c) Use an inductive argument to show that your algorithm properly finds the peak of a unimodal array.

   (d) Write a recurrence relation for the running time of your algorithm as a function of n, and then give a tight closed-form asymptotic bound on the running time.

3. Consider a *uniform* rooted tree of height h; that is, every leaf is at distance h from the root. Every internal node (including the root) has three children. Each leaf has a value of either 0 or 1 associated with it, unknown to us, and each transmits its value to its parent. Each internal node computes the value transmitted to it by the majority of its children and then transmits this value to its parent node. In the end, a final value will be computed by the root by majority vote of its children.

   The evaluation problem consists of determining what the value of the root will be by only reading the values at the children. At each step, an algorithm can choose one leaf whose value it wishes to read.

   Consider the recursive algorithm that evaluates the first two children of the root. If these values agree, the root takes the value of those two children. If the children's values disagree, the algorithm evaluates of the third subtree to determine which value has the majority vote.

   (a) Implement this algorithm in pseudocode.

   (b) Show that, assuming each leaf has a uniform, independent probability of having value 0 or 1, the expected number of leaves inspected by a run of the algorithm is $\Theta(2.5^h)$.