1. (20 points) Assume you are given an unsorted array $A[1..n]$, $n \geq 1$, of integers. In solving the following two problems, you may use algorithms from the lectures or any text. Use pairwise comparisons as the benchmark operation.

   a) In worst-case $O(n)$ time, determine $x, y \in A$ such that $|x - y| \geq |w - z|$ for all $w, z \in A$.

   b) In worst-case $O(n\lg n)$ time, determine $x, y \in A$, $x \neq y$, such that $|x - y| \leq |w - z|$ for all $w, z \in A$. 
2. (20 points) Suppose you want to find the MAXimum and the MINimum of a list of \( n \) distinct numbers, \( \{a, \ldots, a_n\} \), where the benchmark operation is pairwise comparisons. We know any algorithm to solve the problem using pairwise comparisons can be modeled using a decision tree. What lower bound do you get on the worst-case complexity of this problem using the decision tree model?
3. (25 points) Consider program fragment `while ¬B do S`, where $B$ is true with probability $p > 0$, and the successive tests of $B$ are independent.

a As a function of $p$ and $n$, what is the probability that $S$ will be executed more than $n$ times?

b What is the expected number of executions of $S$ if you are given that it is executed more than $n$ times? Your answer should be in closed form.
4. (10 points) In one variant of QUICKSORT, procedure PARTITION decomposes a subarray about a randomly chosen (from a uniform distribution) element $x$ of the subarray. If the $n$ elements to be QUICKSORTed are all distinct, then in one worst-case scenario of this algorithm:

- when PARTITION tries to partition $n$ elements, it chooses $x$ to be the smallest of the $n$ elements,
- in the next pass, when PARTITION tries to partition $n-1$ elements, it chooses $x$ to be the smallest of the $n-1$ elements,
- in the next pass, when PARTITION tries to partition $n-2$ elements, it chooses $x$ to be the smallest of the $n-2$ elements,
- in the final pass, when PARTITION tries to partition 2 elements, it chooses $x$ to be the smaller of the 2 elements.

a What is the probability of this worst-case scenario if the input array is sorted?
b What is the probability of this worst-case scenario if the input array is drawn from a uniform distribution over all permutations of $(1, 2, \ldots, n)$?
5. (25 points) Consider the following algorithm to find the median of $A[1..n]$, where you may assume that $n$ is a power of 3. We assume that function MEDIAN-OF-3 returns the median of three elements in time in $O(1)$. The following is invoked with MEDIAN(1, $n$).

MEDIAN(lo, hi)
   if $hi-lo=2$ then return MEDIAN-OF-3($A[lo], A[lo+1], A[lo+2]$)
   else $FirstThird \leftarrow lo + (hi-lo+1)/3$
       $SecondThird \leftarrow lo + 2*(hi-lo+1)/3$
   return MEDIAN-OF-3(MEDIAN(lo,$FirstThird$),
                        MEDIAN($FirstThird+1$, $SecondThird$),
                        MEDIAN($SecondThird+1$, hi))

a Does MEDIAN always return the median of $n$ elements? Justify your response.

b Analyze the time taken by MEDIAN. You may use asymptotic notation.
1. \(a\) \(x \leftarrow \text{MAX}(A)\) \(\Theta(n)\)
\(y \leftarrow \text{MIN}(A)\) \(\Theta(n)\)

\(b\) \(\text{HEAPSORT}(A)\) \(\Theta(n \log n)\)

\[\text{We know that } x \text{ and } y \text{ must now be adjacent in } A\]
\[\text{for } i \leftarrow 2 \text{ to } n-1 \text{ do}\]
\[\text{if } A[i+1] - A[i] < \delta\]
\[\text{then } \text{closest} \leftarrow i\]
\[\delta \leftarrow A[\text{closest} + 1] - A[\text{closest}]\]
\[\text{return } A[\text{closest}], A[\text{closest}+1]\]

2. We know that since there are \(\binom{n}{2} = \frac{n(n-1)}{2}\) possible answers, the decision tree must contain at least \(\frac{n(n-1)}{2}\) leaves. Thus, the decision tree must contain a path of length at least \(\log \frac{n(n-1)}{2} = \log n + \log(n-1) - 2 \approx 2 \log n - 1\).

3. \(a\) We need the probability that \(B\) will be false at least \(n\) consecutive times, which is \((1 - p)^n\).

\(b\) Let random variable \(X\) denote the number of executions of \(S\). We are given that \(X > n\). The expected value of \(X\) is \(\sum_{k=n}^{\infty} k \Pr\{X = k\} = \sum_{k=n}^{\infty} k (1-p)^{k-1} p\). But, because the geometric distribution is memoryless, this is \(n + \sum_{k=1}^{\infty} k \Pr\{X = k\} = n + \sum_{k=1}^{\infty} k (1-p)^{k-1} = n + 1/p\).

4. \(a, b\) \(\prod_{2 \leq k \leq n} \frac{1}{k} = \frac{1}{n!}\)

5. \(a\) \(\text{MEDIAN}\) fails on input \(\{1, 2, 3, 4, 7, 8, 5, 6, 9\}\). This input has median 5, but algorithm \(\text{MEDIAN}\) returns 6.

\(b\) Letting \(T(n)\) denote the time for \(\text{MEDIAN}\) on an input of \(n\) numbers, its recurrence is
\[T(n) = \begin{cases} c_1, & \text{if } n = 1 \\ 3T(n/3) + c_2, & \text{if } n > 1 \end{cases}\]

Applying the Master Theorem with \(a = b = 3\) and \(f(n) = c_2\) yields case 1, so \(T(n) \in \Theta(n)\).