1. (20 points) Suppose you’re looking for a number $x$ in a sorted list $A[1..n]$, and the benchmark operation is pairwise comparisons of the form $x \geq A[k]$. That is, your algorithm can ask questions like $10 \geq A[1]$ or $3.2 \geq A[12]$. Use a decision tree argument to find a lower bound on the worst-case complexity of the problem.
2. (20 points) Suppose that you want to find the index of the maximum element of list \( A[1..n] \) using the program:

\[
\text{Max} \leftarrow 1 \\
\text{for } k \leftarrow 2 \text{ to } n \\
\quad \text{if } A[k] > A[\text{Max}] \text{ then } \text{Max} \leftarrow k
\]

Assuming that every element of \( A \) is distinct, what is the expected number of times that \( \text{Max} \leftarrow k \) is executed according to each of the following distributions.

(a) Every permutation of \( A \) is equally likely.

(b) \( \Pr \{ A[i] < A[k], 1 \leq i < k \} = \frac{1}{k \cdot H_n} \) for \( 2 \leq k \leq n \).
3. (20 points) Derive an exact solution to the recurrence

\[ T(n) = \begin{cases} 
T(n-1) + 2^n + n, & \text{if } n > 0 \\
1, & \text{if } n = 0 
\end{cases} \]
4. (20 points) A fixed point of an array $A[1..n]$ is a $k$ such that $A[k] = k$. Let $A[1..n]$ be a sorted array of distinct integers. Describe an algorithm to test if $A$ has a fixed point which executes in time in $o(n)$. Note that the time bound is little-oh, not big-oh.
5. (20 points) Assume that $A = a_1, ..., a_n$ is a permutation of $1, ..., n$. We say that $A$ is $k$-sorted if each index $i, 1 \leq i \leq n$, satisfies $|i - a_i| \leq k$. That is, every $a_i$ is at most $k$ positions from its final place after $A$ is sorted. For example, $3, 1, 4, 2, 5$ is 2-sorted but is not 1-sorted. For any fixed value of $k$, what is the asymptotic time taken by INSERTIONSORT on a $k$-sorted list $A$? Use $\Theta$-notation.
1. Because the key being sought can be $A[k]$ for any $1 \leq k \leq n$, or it could possibly not belong to $A$, any decision tree to solve the problem must branch to at least $n+1$ leaves. Hence, the height of the tree (corresponding to the worst-case number of comparisons) must be at least $\lceil \lg (n+1) \rceil$.

2. Letting $X_i = \begin{cases} 1, & \text{if } A[i] < A[k] \text{ for } 2 \leq i < k \\ 0, & \text{otherwise} \end{cases}$ for $2 \leq k \leq n$, we find that the number of executions of $Max \leftarrow k$ is $\sum X_i$. The problem asks for

$$E \left[ \sum X_i \right] = \sum_{2 \leq k \leq n} E[X_i] = \sum_{2 \leq k \leq n} \Pr\{X_i = 1\}.$$  

(a) $\Pr\{X_i = 1\} = 1/k$, so $E \left[ \sum X_i \right] = \sum_{2 \leq k \leq n} 1/k = \sum_{2 \leq k \leq n} 1/k - 1 = H_n - 1$

(b) $\Pr\{X_i = 1\} = \frac{1}{k \cdot H_n}$, so

$$E \left[ \sum X_i \right] = \sum_{2 \leq k \leq n} \frac{1}{k \cdot H_n} = \frac{1}{H_n} \sum_{2 \leq k \leq n} \frac{1}{k} = \frac{1}{H_n} \left( \sum_{1 \leq k \leq n} \frac{1}{k} - 1 \right) = \frac{H_n - 1}{H_n} = 1 - \frac{1}{H_n}$$

3. 

$$T(n) = T(n-1) + 2^n + n = T(n-2) + (2^n + 2^{n-1}) + (n + (n-1))$$

$$= (2^n + 2^{n-1} + \ldots + 2^1 + 2^0) + (n + (n-1) + \ldots + 1 + 0)$$

$$= \sum_{0 \leq k \leq n} 2^k + \sum_{0 \leq k \leq n} k = 2^{n+1} - 1 + \frac{n(n+1)}{2}$$

4. The following procedure, analogous to binary search, is invoked with $\text{FIXEDPOINT}(1,n)$.

\begin{algorithm}
\textbf{FIXEDPOINT}(lo,hi)
\begin{itemize}
  \item If there’s a fixed point, it’s within $A[lo,hi]$
  \item if $lo=hi$ then if $A[lo]=lo$ then return $lo$
  \item else return fail
  \item mid $\leftarrow \left\lfloor (lo + hi) / 2 \right\rfloor$
  \item if $A[mid] > mid$ then return $\text{FIXEDPOINT}(lo, mid-1)$
  \item else return $\text{FIXEDPOINT}(mid+1, hi)$
\end{itemize}
\end{algorithm}

The execution time of $\text{FIXEDPOINT}$ is in $O(\lceil \lg n \rceil)$, which is certainly $o(n)$.

5. If a list is $k$-sorted, then every element is involved in at most $k$ inversions. Since there are $n$ elements, there are no more than $kn$ inversions. Since the number of swaps done by $\text{INSERTIONSORT}$ is
proportional to the number of inversions, then for fixed $k$ the time taken by INSERTIONSORT on a $k$-sorted array is in $\Theta(n)$. 