CS525AD

MIDTERM EXAM

Name_____

Date: October 16, 2003
All documentation permitte

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 $x \overset{?}{\geq} A[k]$. That is, your algorithm can ask questions like $10 \overset{?}{\geq} A[1]$ or $3.2 \overset{?}{\geq} A[12]$. 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

```
Max ← 1
for k ← 2 to n
    if A[k]>A[Max] then Max ← k
```

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

(a) Every permutation of A is equally likely.

(b) $\Pr\{A[i] < A[k], 1 \le i < k\} = \dfrac{1}{k * H_n}$ for $2 \le k \le n$.

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 $A = a_1, \ldots, a_n$ is a permutation of $1, \ldots, n$. We say that A is *k-sorted* if each index i, $1 \le i \le n$, satisfies $|i - a_i| \le k$. 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 Θ-notation.

CS525DA

Solutions to Midterm Exam

1. Because the key being sought can be $A[k]$ for any $1 \le k \le n$, 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 rv $X_k = \begin{cases} 1, \text{if } A[i] < A[k] \text{ for } 2 \le i < k \\ 0, \text{otherwise} \end{cases}$ for $2 \le k \le n$, we find that the number of executions of $Max \leftarrow k$ is $\sum_{2 \le k \le n} X_k$. The problem asks for

$$E\left[\sum_{2 \le k \le n} X_k\right] = \sum_{2 \le k \le n} E[X_k] = \sum_{2 \le k \le n} \Pr\{X_k = 1\}.$$

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

(b) $\Pr\{X_k = 1\} = \dfrac{1}{k * H_n}$, so

$$E\left[\sum_{2 \le k \le n} X_k\right] = \sum_{2 \le k \le n} \frac{1}{k * H_n} = \frac{1}{H_n} \sum_{2 \le k \le n} \frac{1}{k} = \frac{1}{H_n}\left(\sum_{1 \le k \le n} \frac{1}{k} - 1\right) = \frac{H_n - 1}{H_n} = 1 - \frac{1}{H_n}$$

3.
$$\begin{aligned} T(n) &= T(n-1) + 2^n + n = T(n-2) + \left(2^n + 2^{n-1}\right) + \left(n + (n-1)\right) \\ &= \left(2^n + 2^{n-1} + \ldots + 2^1 + 2^0\right) + \left(n + (n-1) + \ldots + 1 + 0\right) \\ &= \sum_{0 \le k \le n} 2^k + \sum_{0 \le k \le n} k = 2^{n+1} - 1 + \frac{n(n+1)}{2} \end{aligned}$$

4. The following procedure, analagous to binary search, is invoked with FIXEDPOINT(1,n).

```
FIXEDPOINT(lo,hi)
    ▸ If there's a fixed point, it's within A[lo,hi]
    if lo=hi then if A[lo]=lo then return lo
                                  else return fail
    mid ← ⌊(lo+hi)/2⌋
    if A[mid]>mid then return FIXEDPOINT(lo,mid-1)
                  else return FIXEDPOINT(mid+1,hi)
```

The execution time of FIXEDPOINT is in $O(\lg n)$, 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 INSERTIONSORT is

proportional to the number of inversions, then for fixed k the time taken by INSERTIONSORT on a k-sorted array is in $\Theta(n)$.