CS2223

MIDTERM EXAM

Name_____

Date: November 16, 2006
All documentation permitted

1	(25)	_____
2	(30)	_____
3	(25)	_____
4	(20)	_____
TOTAL		_____

1. (25 points) How many times is the comparison $A[i] < A[j]$ executed in the following code fragment? Give an exact answer; do not use asymptotic notation.

```
for i ← 1 to n  do Count[i] ← 0
for i ← n downto 2 do
      for j ← i-1 downto 1 do
           if A[i]< A[j] then Count[j] ← Count[j] + 1
                         else  Count[i] ← Count[i] + 1
```

2. (30 points) ***a*** Either construct two different functions *f*(*n*) and *g*(*n*) such that $f(n) \in O(g(n))$ **and** $f(n) \in \Omega(g(n))$ or show that this is impossible.

b Is $3^{n+3} \in O(3^n)$? Justify your response.

c Does $f(n) \in O(g(n))$ and $g(n) \in O(h(n))$ imply that $f(n) \in O(h(n))$ **for all possible** functions *f*, *g* and *h*? Justify your response.

3. (25 points) We say a list $A[1..n]$ is *2-sorted* if each element is at most two positions from its final position. That is, if A is sorted then the element which is originally in $A[i]$, $1 \le i \le n$, will end up in one of the positions $A[i-2], A[i-1], A[i], A[i+1], A[i+2]$, where these positions might not exist for i=1, 2, n-1 and n. What is the fastest algorithm to sort a 2-sorted list for large n? Justify your response.

4. (20 points) Show that a worst-case lower bound for testing if an element x belongs to an array $A[1..n]$ is $\Omega(n)$, where the benchmark operation is a test of the form "Is $A[i] \stackrel{?}{=} x$?"

CS2223

Solutions to Midterm Exam

1. $\sum_{i=2}^{n}\sum_{j=1}^{i-1}1=\sum_{i=2}^{n}(i-1)=\sum_{i=2}^{n}i-\sum_{i=2}^{n}1=\frac{n(n+1)}{2}-1-(n-1)=\frac{n^2-n}{2}$

2. ***a*** $f(n)=n$ and $g(n)=42n$.

b $3^{n+3}=27*3^n$ so choosing c=27 and $n_0=1$ it follows that $3^{n+3}\le 27*3^n$ for all $n\ge n_0$.

c The claim is true. If $f(n)\in O(g(n))$ and $g(n)\in O(h(n))$, then there exist constants c_0,c_1,n_0,n_1 such that $f(n)\le c_0 g(n)$ for all $n\ge n_0$ and $g(n)\le c_1 h(n)$ for all $n\ge n_1$. But then $f(n)\le c_0c_1h(n)$ for all $n\ge \max(n_0,n_1)$, which implies that $f(n)\in O(h(n))$.

3. Each $A[i]$ can only have at most two elements to its left which are larger than it is (at most two inversions), and hence INSERTIONSORT will execute in time in $\Theta(n)$.

4. Assume that such an algorithm exists. It must return false on $A=(1,2,...,n)$ and x=n+1. Since the algorithm can not probe every element of A in $\Omega(n)$ operations, there must be some element of A , say i, which wasn't probed. Change A by replacing $A[i]$ by n+1, and leaving the rest unchanged.. The algorithm receives the same answers to the same questions and hence still returns false, but now the answer is incorrect. By contradiction, such an algorithm can not exist.