CSE 207 Class Test 1 Answers with Brief Explanation

15 February, 2011. Full marks: 20

Student No.. Name: ....

True/False ($40 * 0.5 = 20$) marks.

Write "T" for true and "F" for false for each of the following statement. Write in the left margin.

1. When we say "running time", we mean "worst case" running time. [T]
2. $f(n) = O(g(n))$ can also be written as $f(n) \in O(g(n))$; In fact, the later approach is more appropriate. [T, because $g(n)$ is a set of many functions like $f(n)$, i.e., there are many functions like $f(n)$ whose asymptotic upper bound is $g(n)$.]
3. If $f(n) = \Theta(g(n))$, then $f(n) = O(g(n))$. [T]
4. If $f(n) = \Omega(g(n))$, then $g(n)) = O(f(n))$. [T]
5. $f(n) = o(g(n))$ if and only if $g(n) = \omega(f(n))$. [T]
6. $2n^2 = o(n^2)$. [F]
7. The statement "The running time of algorithm A is at least $O(n^2)$" is meaningless, because it should be written as "The running time of algorithm A is at least $\Omega(n^2)$. [T]
8. o and ω are asymptotic tight bounds. [F, non-tight bounds.]
9. Θ is asymptotic perfect bound. [F, asymptotic tight bounds.]
10. $\lg n$ is polynomially larger than n^{ε}, for some small constant $\varepsilon > 0$. [F, try $LT_{n\to\infty} \frac{\lg n}{n^{\varepsilon}}$.]
11. $\lg n = O(n \lg n)$ but not $\lg n = \Omega(n \lg n)$ [T, first one is not tight.]
12. $\lg \lg n = \lg^2 n$. [F, $\lg \lg n = \lg(\lg n)$ and $\lg^2 n$ is $(\lg n)^2$; two are different.]
13. In order to say $f(n) = o(g(n))$, is suffices to find a constant $c > 0$ and an integer $n_0 > 0$ such that $0 \leq f(n) < cg(n)$. [F, it should be for *all* constant $c > 0$.]
14. If $f(n) = \omega(g(n))$, then $LT_{n\to\infty} \frac{f(n)}{g(n)} = 0$. [F, should be ∞]
15. $n^{\lg c} = \Theta(c^{\lg n})$. [T, take $\lg n = y$, then both solves to c^y]
16. Sub-linear means $O(n \lg n)$. [F, $o(n)$]
17. Strassen's algorithm improves on traditional divide and conquer algorithm, because it uses small number of addition. [F, small number of multiplication.]
18. The recurrence relation for traditional divide and conquer algorithm for matrix multiplication is: $T(n) = 6T(n/2) + \Theta(n)$. [F, 6 should be 7.]

19. The recurrence relation for Strassen's algorithm is: $T(n) = 7T(n/2) + O(1)$. [F, $O(1)$ should be $\Theta(n)$.]

20. In the above two recurrence relations, 6 and 7 stand for 6 and 7 additions respectively. [F, 6 and 7 are for multiplications.]

21. Master method can solve any recurrence relation. [F]

22. Asymptotic bounds achieved by Master theorem are always tight. [T, all cases give Θ.]

23. $T(n) = 2T(n/4) + \sqrt{n}$ solves to $\Theta(\sqrt{n} \lg n)$. [T, case 2 of Master method.]

24. The second condition in Case 3 of Master theorem indicates that the cost of conquer should be non-increasing. [T]

25. Master theorem can solve this recurrence: $T(n) = 4T(n/2)+n^2 \lg n$. [F, similar to the negative example given in the book.]

26. In quick sort, two elements are compared at least once. [F, at most once.]

27. In quick sort, assuming the numbering of the elements as $z_1, z_2, \ldots, z_n$ in their sorted sequence, the probability of two elements z_i and z_j are compared is: $\frac{2}{i+j+1}$. [F, $\frac{2}{j-i+1}$]

28. In quick sort, two elements z_i and z_j are compared if and only if one of them is selected as a pivot among the elements that are within the sorted sequence of z_i and z_j (inclusive). [F, "selected first"]

29. Average running time and expected running time are the same. [T]

30. Running time of a quick sort algorithm with $75\% - 25\%$ balanced partitioning is $O(n \lg_{0.25} n)$. [F, $O(n \lg_{4/3} n)$]

31. $\sum_{k=1}^{n} \frac{2}{k} = O(\lg^2 n)$. [F, $O(\lg n)$]

32. A lower bound for any sorting algorithm is $\Omega(n \lg n)$. [T, but not tight, e.g., $\Omega(n^2)$ for insertion sort.]

33. Lower bound for sorting is $\Omega(n \lg n)$. [T]

34. $\Omega(n)$ is another lower bound for sorting. [T, but not tight.]

35. $\lg n! \geq \lg(n/2)^{(n/2)} = \Omega(n \lg n)$. [T]

36. To determine the minimum running time from a decision tree, we need to take the shortest path in the tree. [F, longest path.]

37. An algorithm for a problem is asymptotically optimal when the upper bound matches the lower bound of the algorithm. [F, should match with the lower bound of the *problem*.]

38. Insertion sort has running time $\Theta(n^2)$, i.e., both upper and lower bounds are same as n^2. Therefore, insertion sort is an optimal sorting algorithm. [F, in the class test question, the second n^2 was written as $n \lg n$, it was a typo, but still the answer is "F".]

39. The leaves of a decision tree for sorting are the permutations of the elements. [T]

40. An exponential running time is infeasible. [T, because it would take too much time as n grows larger.]