Università Degli Studi Di L'Aquila

Prova di recupero di **Algoritmi e Strutture Dati**

Martedì 12 Febbraio 2008 – Prof. Guido Proietti

Scrivi i tuoi dati ⟹	Cognome:	Nome:	Matricola:	PUNTI
ESERCIZIO 1	Risposte Esatte:	Risposte Omesse:	Risposte Errate:	

ESERCIZIO 1 (25 punti): Domande a risposta multipla

Premessa: Questa parte è costituita da 20 domande a risposta multipla. Per ciascuna domanda vengono fornite 4 risposte, di cui soltanto una è corretta. Per rispondere utilizzare la griglia annessa, barrando con una × la casella corrispondente alla risposta prescelta. È consentito omettere la risposta. In caso di errore, contornare con un cerchietto la × erroneamente apposta (ovvero, in questo modo ⊗) e rifare la × sulla nuova risposta prescelta. Se una domanda presenta più di una risposta, verrà considerata omessa. Per tutti i quesiti verrà attribuito un identico punteggio, e cioè: risposta esatta 3 punti, risposta omessa 0 punti, risposta sbagliata -1 punto. Il voto relativo a questa parte è ottenuto sommando i punti ottenuti e normalizzando su base 25. Se tale somma è negativa, verrà assegnato 0.

1. Quale delle seguenti relazioni di ricorrenza descrive la complessità dell'algoritmo più efficiente per il calcolo della sequenza di Fibonacci basato sul prodotto di matrici?
   a) $T(n) = 2T(n/2) + O(1)$ se $n \geq 2$, $T(1) = O(1)$ se $n = 1$ b) $T(n) = 2T(n/4) + O(1)$ se $n \geq 2$, $T(1) = O(1)$ se $n = 1$
   *c) $T(n) = T(n/2) + O(1)$ se $n \geq 2$, $T(1) = O(1)$ se $n = 1$ d) $T(n) = 2T(n/2) + O(1)$ se $n \geq 2$, $T(1) = O(n)$ se $n = 1$
2. Siano $f(n)$ e $g(n)$ i costi dell'algoritmo Insertion Sort nel caso medio e Selection Sort in quello migliore, rispettivamente. Quale delle seguenti relazioni asintotiche è falsa:
   *a) $f(n) = o(g(n))$ b) $f(n) = \Theta(g(n))$ c) $f(n) = O(g(n))$ d) $f(n) = \Omega(g(n))$
3. Un problema ha una delimitazione inferiore alla complessità temporale $\Omega(f(n))$ se:
   *a) Tutti gli algoritmi per la sua risoluzione hanno una delimitazione inferiore alla complessità computazionale pari a $\Omega(f(n))$
   b) Tutti gli algoritmi per la sua risoluzione hanno una delimitazione superiore alla complessità computazionale pari a $O(f(n))$
   c) Esiste un algoritmo per la sua risoluzione che ha una delimitazione inferiore alla complessità computazionale pari a $\Omega(f(n))$
   d) Esiste un algoritmo per la sua risoluzione che ha una delimitazione superiore alla complessità computazionale pari a $O(f(n))$
4. L'algoritmo ottimale di fusione di due sequenze ordinate di lunghezza p e q rispettivamente, ha complessità:
   a) $\Theta(p \cdot q)$ b) $\Theta(p)$ c) $\omega(p + q)$ *d) $\Theta(p + q)$
5. A quale delle seguenti classi non appartiene la complessità dell'algoritmo Merge Sort:
   *a) $o(n \log n)$ b) $O(n^2)$ c) $\Omega(n)$ d) $\Theta(n \log n)$
6. Siano $f(n)$ e $g(n)$ i costi degli algoritmi Heapsort e Quicksort nel caso peggiore e in quello medio, rispettivamente. Quale delle seguenti relazioni asintotiche è vera:
   a) $g(n) = o(f(n))$ *b) $f(n) = \Theta(g(n))$ c) $f(n) = \omega(g(n))$ d) $g(n) = \omega(f(n))$
7. Sia dato un array A di n elementi in cui l'elemento massimo è pari a n^c, con c costante positiva. Qual è la complessità temporale dell'algoritmo Radix Sort applicato ad A?
   a) $\Theta(n^c)$ b) $\Theta(n \log_c n)$ *c) $O(n)$ d) $\Theta(n \log n)$
8. Un heap binario di altezza k contiene:
   a) tra 2^{k+1} e $2^{k+2} - 1$ elementi *b) meno di 2^{k+1} elementi c) più di 2^k elementi d) tra $2^k + 1$ e $2^{k+1} - 1$ elementi
9. La procedura *Heapify* per la costruzione di un heap applicata al vettore $A = [5, 6, 9, 3, 12]$ restituisce:
   a) $A = [12, 9, 3, 6, 5]$ b) $A = [12, 6, 5, 9, 3]$ c) $A = [12, 5, 3, 6, 9]$ *d) $A = [12, 6, 9, 3, 5]$
10. Una coda di priorità realizzata con un array non ordinato supporta l'estrazione del massimo in:
    a) $\Theta(\log n)$ b) $O(\log n)$ c) $\Theta(1)$ *d) $\Theta(n)$
11. Sia H_1 un heap binomiale di 15 elementi, e sia H_2 un heap binomiale di 1 elemento. Da quali alberi binomiali è formato l'heap binomiale ottenuto dalla fusione di H_1 e H_2?
    *a) $\{B_4\}$ b) $\{B_0, B_1, B_2, B_3\}$ c) $\{B_{16}\}$ d) $\{B_0, B_0, B_1, B_2, B_3\}$
12. L'altezza di un qualsiasi albero di decisione associato al problema della ricerca in un insieme ordinato di n elementi è:
    a) $\Theta(n \log n)$ *b) $\Omega(\log n)$ c) $\Theta(\log n)$ d) $\Omega(n)$
13. In un albero binario di ricerca di altezza h, il *successore* di un elemento può essere determinato in:
    a) $\Theta(\log h)$ b) $O(\log h)$ c) $\Theta(1)$ *d) $O(h)$
14. Dati due elementi u, v appartenenti ad un universo totalmente ordinato U, una funzione hash $h(\cdot)$ si dice *perfetta* se:
    a) $u = v \Rightarrow h(u) \neq h(v)$ b) $u \neq v \Rightarrow h(u) = h(v)$ c) $u = v \Rightarrow h(u) = h(v)$ *d) $u \neq v \Rightarrow h(u) \neq h(v)$
15. Sia $G = (V, E)$ un grafo *completo* di 6 vertici, in cui i vertici sono numerati da 1 a 6, ed il peso dell'arco (i, j) è pari al minimo tra i e j. Qual è la distanza tra il vertice 3 e il vertice 6 in G?
    a) 1 *b) 2 c) 3 d) 6
16. Il *minimo albero ricoprente* del grafo della domanda (15) ha peso totale: a) 6 b) 15 c) 0 *d) 5
17. Dato un grafo pesato con n vertici ed $m = O(n)$ archi, l'algoritmo di Dijkstra realizzato con heap di Fibonacci costa:
    a) $\Theta(n^2)$ b) $\Theta(n + m)$ c) $O(m)$ *d) $O(n \log n)$
18. Usando gli alberi *QuickUnion* e l'euristica dell'unione pesata *by size*, il problema della gestione di n insiemi disgiunti sottoposti ad $n - 1$ *Union* ed m *Find* può essere risolto in:
    a) $\Theta(n)$ b) $\Theta(m)$ c) $\Theta(m^2)$ *d) $O(n + m \log n)$
19. Dato il grafo di domanda 15, il primo arco scartato dall'algoritmo di Kruskal ha peso: a) 1 *b) 2 c) 3 d) 6
20. Dato un grafo completo con n vertici, l'algoritmo di Prim eseguito con un *heap binario* costa:
    a) $O(n^2)$ *b) $O(n^2 \log n)$ c) $\Theta(n^2 \log n)$ d) $\Theta(n^2)$

Griglia Risposte

	Domanda																			
Risposta	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
a																				
b																				
c																				
d																				

ESERCIZIO 2 (5 punti) (**Da svolgere sul retro della pagina!**)

Mostrare l'intera evoluzione, passo per passo, di un albero AVL inizialmente vuoto, sul quale vengono eseguite le seguenti operazioni di inserimento e cancellazione, specificando le eventuali rotazioni applicate: *Insert*(1), *Insert*(2), *Insert*(3), *Insert*(4), *Insert*(5), *Insert*(6), *Insert*(7), *Delete*(1), *Delete*(3).