Università Degli Studi Di L'Aquila

Prova di recupero di **Algoritmi e Strutture Dati con Laboratorio** (parte di teoria)

Martedì 12 Febbraio 2008 – Proff. Guido Proietti e Giovanna Melideo

Scrivi i tuoi dati $\Longrightarrow$	Cognome: ..... Nome: ..... Matricola: .....

Da riconsegnare entro 40 minuti

ESERCIZIO: Domande a risposta multipla

Premessa: Questa parte è costituita da 10 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 $\times$ la casella corrispondente alla risposta prescelta. È consentito omettere la risposta. In caso di errore, contornare con un cerchietto la $\times$ erroneamente apposta (ovvero, in questo modo $\otimes$) e rifare la $\times$ 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 30. Se tale somma è negativa, verrà assegnato 0.

1. 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) $f(n) = \Omega(g(n))$
2. 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 $O(f(n))$
3. A quale delle seguenti classi <u>non</u> appartiene la complessità dell'algoritmo Merge Sort:
   *a) $o(n \log n)$ b) $O(n^2)$ c) $\Omega(n)$ d) $\Theta(n \log n)$
4. 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) $\Theta(n \log n)$
5. 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) $A = [12, 6, 9, 3, 5]$
6. 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) $\{B_0, B_0, B_1, B_2, B_3\}$
7. 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) $u \neq v \Rightarrow h(u) \neq h(v)$
8. 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
9. 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) $O(n + m \log n)$
10. 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
a										
b										
c										
d										