Università degli Studi dell'Aquila

Prova di Recupero di **Algoritmi e Strutture Dati con Laboratorio**

Mercoledì 28 Novembre 2012 – Proff. Guido Proietti e Giovanna Melideo

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

ESERCIZIO 1: 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. 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. L'algoritmo di ordinamento crescente Insertion Sort applicato ad una sequenza di input ordinata in modo decrescente esegue un numero di confronti tra elementi pari a: a) $n - 1$ b) $n(n + 1)/2$ c) $n + 1$ *d) $n(n - 1)/2$
3. Sia $f(n) = 3n^3 - 4$; affinché sia $f(n) = \Omega(n^2)$, è sufficiente scegliere
   a) $n_0 = 1, c = 1$ b) $n_0 = 1, c = 2$ *c) $n_0 = 2, c = 1$ d) $n_0 = 2, c = 6$
4. A quale delle seguenti classi di complessità <u>non</u> appartiene la complessità dell'algoritmo Quicksort:
   a) $O(n^3)$ b) $\Theta(n^2)$ c) $\Omega(n)$ *d) $o(n^2)$
5. Qual è la complessità dell'algoritmo Radix Sort applicato ad un array A di n elementi in cui $A[i] = 2i^4$ per $i = 1, .., n$?
   a) $\Theta(n^4)$ b) $\Theta(1)$ *c) $O(n)$ d) $\Theta(n \log n)$
6. L'algoritmo *Heapify* per la costruzione di un heap(A) applicato ad $A = [3, 5, 4, 6, 7]$ restituisce:
   a) $A = [7, 6, 5, 3, 4]$ b) $A = [7, 6, 3, 4, 5]$ c) $A = [7, 5, 6, 4, 3]$ *d) $A = [7, 6, 4, 3, 5]$
7. Si consideri l'albero AVL di 7 elementi, nella cui radice compare la chiave 5, nel figlio sinistro della radice compare la chiave 3, nel figlio destro della radice compare la chiave 6, nel figlio sinistro del figlio sinistro della radice compare la chiave 2, nel figlio destro del figlio sinistro della radice compare la chiave 4, nel figlio sinitro del figlio sinistro del figlio sinistro della radice compare la chiave 1, e infine nel figlio destro del figlio destro della radice compare la chiave 7. Si supponga ora di cancellare l'elemento nella radice dell'AVL. Quale albero AVL risulterà?
   a) Nella radice compare la chiave 3, nel figlio sinistro della radice compare la chiave 2, nel figlio destro della radice compare la chiave 6, nel figlio sinistro del figlio sinistro della radice compare la chiave 1, nel figlio sinistro del figlio destro della radice compare la chiave 4, e infine nel figlio destro del figlio destro della radice compare la chiave 7.
   *b) Nella radice compare la chiave 4, nel figlio sinistro della radice compare la chiave 2, nel figlio destro della radice compare la chiave 6, nel figlio sinistro del figlio sinistro della radice compare la chiave 1, nel figlio destro del figlio sinistro della radice compare la chiave 3, e infine nel figlio destro del figlio destro della radice compare la chiave 7.
   c) Nella radice compare la chiave 4, nel figlio sinistro della radice compare la chiave 3, nel figlio destro della radice compare la chiave 6, nel figlio sinistro del figlio sinistro della radice compare la chiave 1, nel figlio destro del figlio sinistro della radice compare la chiave 2, e infine nel figlio destro del figlio destro della radice compare la chiave 7.
   d) Nella radice compare la chiave 4, nel figlio sinistro della radice compare la chiave 3, nel figlio destro della radice compare la chiave 6, nel figlio sinistro del figlio sinistro della radice compare la chiave 2, nel figlio sinitro del figlio sinistro del figlio sinistro della radice compare la chiave 1, e infine nel figlio destro del figlio destro della radice compare la chiave 7.
8. L'albero BFS (ovvero ottenuto mediante una *visita in ampiezza*) di un grafo connesso di n vertici ha altezza:
   a) $\Theta(n)$ *b) $O(n)$ c) $O(\log n)$ d) 1
9. Sia dato un grafo non diretto G con n vertici, numerati da 1 ad n, ed $n - 1$ archi, disposti in modo arbitrario, ma in modo tale da garantire la connessione. Si orientino ora gli archi in modo arbitrario, e si applichi l'algoritmo di ordinamento topologico rispetto al nodo sorgente etichettato 1. La complessità risultante è pari a:
   a) $\Theta(n^2)$ *b) $\Theta(n)$ c) $\Theta(n \log n)$ d) indefinita (non è detto che l'algoritmo possa essere applicato)
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										