Università degli Studi dell'Aquila

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

Lunedì 4 Febbraio 2013 – 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. Siano $f(n)$ e $g(n)$ i costi dell'algoritmo Insertion Sort nel caso peggiore e Selection Sort in quello medio, rispettivamente. Quale delle seguenti relazioni asintotiche è vera:
   a) $f(n) = o(g(n))$ *b) $f(n) = \Theta(g(n))$ c) $f(n) = \omega(g(n))$ d) $g(n) = \omega(f(n))$
2. Un algoritmo ha una complessità temporale $O(f(n))$ se:
   a) Il tempo di esecuzione $T(n)$ dell'algoritmo su uno specifico input di dimensione n verifica $T(n) = O(f(n))$
   *b) Il tempo di esecuzione $T(n)$ dell'algoritmo su ogni input di dimensione n verifica $T(n) = O(f(n))$
   c) Il tempo di esecuzione medio $T(n)$ dell'algoritmo su un input di dimensione n verifica $T(n) = O(f(n))$
   d) Nel caso migliore, il tempo di esecuzione $T(n)$ dell'algoritmo su un input di dimensione n verifica $T(n) = O(f(n))$
3. Il numero di foglie dell'albero di decisione associato al problema dell'ordinamento di n elementi è:
   *a) $\Omega(n!)$ b) $O(n!)$ c) $\Theta(n \log n)$ d) $\Theta(n!)$
4. Dato un albero AVL T contenente n elementi, si supponga di cancellare una sequenza arbitraria di $\log n$ elementi contenuti in T. L'altezza dell'AVL risultante sarà pari a:
   a) $\Theta(n)$ b) $\Theta(n/\log n)$ c) $\Theta(1)$ *d) $\Theta(\log n)$
5. Si supponga di inserire la sequenza di chiavi $10, 22, 31$ (in quest'ordine) in una tavola hash di lunghezza $m = 3$ (ovvero con indici $0, 1, 2$) utilizzando l'indirizzamento aperto con funzione hash $h(k) = k \mod 3$, e risolvendo le collisioni con il metodo della scansione lineare. Quale sarà la tavola hash finale?
   a) $A = [10, 22, 31]$ b) $A = [22, 31, 10]$ c) $A = [31, 22, 10]$ *d) $A = [31, 10, 22]$
6. Si consideri il grafo $G = (V, E)$ con $V = \{1, 2, 3, 4\}$ ed $E = \{(1, 3), (1, 4), (2, 3), (2, 4)\}$. Quali delle seguenti affermazioni è <u>falsa</u>:
   a) G è bipartito b) G è connesso *c) G è aciclico d) G è bipartito completo
7. Si consideri il grafo di Domanda 6, e si pesino gli archi come segue (si osservi che gli archi non sono orientati): $w(1, 3) = 1, w(1, 4) = 5, w(2, 3) = 1, w(2, 4) = 2$. Applicando l'algoritmo di Dijkstra con node sorgente 1, qual è la sequenza di inserimento dei nodi nella soluzione finale?
   *a) $1, 3, 2, 4$ b) $1, 3, 4, 2$ c) $1, 2, 3, 4$ d) $1, 4, 3, 2$
8. Dato un grafo connesso di n nodi ed m archi, per quale valore (asintotico) di m si ha che l'implementazione ripetuta dell'algoritmo di Dijkstra con heap di Fibonacci ha la stessa complessità temporale dell'implementazione di Floyd e Warshall per trovare le distanze tra tutte le coppie di nodi del grafo (entrami implementati utilizzando liste di adiacenza)?
   a) $m = \Theta(n)$ *b) $m = \Theta(n^2)$ c) $m = \Theta(n \log n)$ d) mai
9. Dato un grafo connesso di n nodi ed m archi, per quale valore (asintotico) di m si ha che l'implementazione di Prim con heap di Fibonacci è <u>strettamente più efficiente</u> dell'implementazione di Kruskal con alberi QuickUnion con euristica di bilanciamento *union by size*?
   a) $m = \Theta(n)$ b) sempre c) mai *d) $m = \omega(n)$
10. Dato un grafo connesso con n vertici ed m archi, il numero <u>minimo</u> di archi che l'algoritmo di Borůvka aggiunge alla soluzione alla fine della prima passata è pari a:
    a) $n - 1$ b) $\log n$ c) 1 *d) $\lceil n/2 \rceil$

Griglia Risposte

	Domanda									
Risposta	1	2	3	4	5	6	7	8	9	10
a										
b										
c										
d										