Università degli Studi dell'Aquila

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

Martedì 14 Luglio 2015 – Prof. Guido Proietti (Modulo di Teoria)

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

ESERCIZIO 1: Domande a risposta multipla

Premessa: La prova 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 × 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 finale è dato dalla somma dei punti ottenuti. Se tale somma è negativa, verrà assegnato 0.

1. Sia $f(n) = n \log^2 n$. Quale tra le seguenti relazioni asintotiche è falsa?
   a) $f(n) = \Omega(n \log^2 n)$ *b) $f(n) = \Theta(n \log n)$ c) $f(n) = o(n^{1,00001})$ d) $f(n) = \omega(n \log n^3)$
2. Siano $f(n)$ e $g(n)$ i costi dell'algoritmo Insertion Sort (basato sul ciclo `while`) 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. L'altezza dell'albero di decisione associato al problema della fusione ordinata di due sequenze ordinate è (suggerimento: lo si ricavi dalla complessità del relativo algoritmo ottimo studiato a lezione):
   *a) $\Omega(n)$ b) $\omega(n \log n)$ c) $O(1)$ d) $\Theta(n!)$
4. Quale tra i seguenti è lo pseudocodice del Merge Sort (si supponga che la chiamata principale sia MergeSort$(A, 1, n)$)

   *a) MergeSort(A, i, f)
   if $(i \geq f)$ then return
   $m = (i + f)/2$
   MergeSort(A, i, m)
   MergeSort$(A, m + 1, f)$
   Merge(A, i, m, f)

   b) MergeSort(A, i, f)
   if $(i \leq f)$ then return
   $m = (i + f)/2$
   MergeSort(A, i, m)
   MergeSort$(A, m + 1, f)$
   Merge(A, i, m, f)

   c) MergeSort(A, i, f)
   if $(i \geq f)$ then return
   $m = (i + f)/2$
   Merge(A, i, m, f)
   MergeSort(A, i, m)
   MergeSort$(A, m + 1, f)$

   d) MergeSort(A, i, f)
   if $(i \geq f)$ then return
   $m = (i + f)/2$
   MergeSort(A, i, m)
   MergeSort$(A, m + 1, f)$
   Merge(A, i, f, m)
5. La procedura *FixHeap*$(A, 1)$ per il mantenimento di un heap binario applicata al vettore $A = [12, 15, 21, 14, 9, 13]$ restituisce:
   a) $A = [21, 15, 12, 14, 9, 13]$ b) $A = [15, 12, 21, 14, 9, 13]$ c) $A = [15, 14, 21, 12, 9, 13]$ *d) $A = [21, 15, 13, 14, 9, 12]$
6. Come si esegue l'operazione `increaseKey`(elem e, chiave Δ) di un elemento con chiave k in un heap binomiale?
   a) eseguendo `insert`$(e, k + \Delta)$ b) eseguendo `decreaseKey`$(e, -\Delta)$
   c) eseguendo `delete`(e) seguita da `insert`(e, Δ) *d) eseguendo `delete`(e) seguita da `insert`$(e, k + \Delta)$
7. Si consideri il semplice albero AVL di 5 elementi, nella cui radice compare la chiave 5, 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, e infine nel figlio destro del figlio sinistro della radice compare la chiave 3. Si supponga ora di inserire l'elemento 4 nell'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 5.
   *b) Nella radice compare la chiave 3, nel figlio sinistro della radice compare la chiave 2, nel figlio destro della radice compare la chiave 5, 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 6.
   c) Nella radice compare la chiave 4, nel figlio sinistro della radice compare la chiave 3, nel figlio destro della radice compare la chiave 5, 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 sinistro del figlio destro della radice compare la chiave 6.
   d) Nella radice compare la chiave 5, 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 3, e infine nel figlio destro del figlio destro della radice compare la chiave 4.
8. L'algoritmo di Bellman e Ford applicato ad un grafo pesato con un numero di archi $m = \Theta(n)$, ha complessità:
   *a) $\Theta(n^2)$ b) $\Theta(n + m)$ c) $\Theta(n^3)$ d) $O(m \log n)$
9. Sia dato un grafo completo di 5 nodi, e si assegnino arbitrariamente ma senza ripetizioni ai suoi archi dei pesi interi compresi tra 1 e 10. Quale tra i seguenti è un costo ammissibile per un *minimo albero ricoprente* del grafo?
   a) 0 b) 4 c) 5 *d) 10
10. Dato un grafo pesato con n vertici ed m archi, il costo di una fase dell'algoritmo di Borůvka è pari a:
    *a) $O(m)$ b) $O(n)$ c) $\Theta(m + n \log n)$ d) $\Theta(m \log n)$

Griglia Risposte

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