Università degli Studi dell'Aquila

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

Giovedì 7 Luglio 2016 – 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

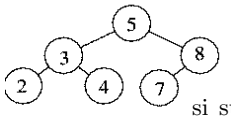
Premessa: Per ciascuna domanda vengono fornite 4 risposte, di cui soltanto una è corretta. Per rispondere utilizzare la griglia annessa, barrando con una × la casella corrispondente alla risposta prescelta. È consentito omettere la risposta. In caso di errore, contornare con un cerchietto la × erroneamente 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 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'altezza dell'albero di decisione associato al problema dell'ordinamento è:
   *a) $\Omega(n \log n)$ b) $\omega(n \log n)$ c) $O(n \log n)$ d) $\Theta(n!)$
3. 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 $(i \leq f)$ then return
   $m = (i + f)/2$
   MergeSort(A, i, m)
   MergeSort$(A, m + 1, f)$
   Merge(A, i, m, f)

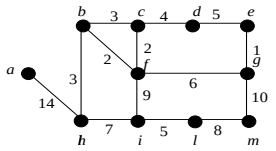
   *b) MergeSort(A, i, f)
   if $(i \geq f)$ then return
   $m = (i + f)/2$
   MergeSort(A, i, m)
   MergeSort$(A, m + 1, f)$
   Merge(A, i, m, f)

   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)
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 temporale dell'algoritmo Bucket Sort applicato ad un array A di n elementi in cui l'elemento massimo è pari a 2^{16}?
   a) $\Theta(2^{16})$ *b) $\Theta(n)$ c) $O(n + k)$ d) $\Theta(n \log n)$
6. Dato l'albero AVL 

   si supponga di inserire l'elemento con chiave 6. La visita in ordine anticipato dell'AVL ribilanciato restituisce:

   *a) 5,3,2,4,7,6,8 b) 2,3,4,5,6,7,8 c) 2,4,3,5,6,8,7 d) 8,7,6,5,4,3,2
7. Sia data una tavola ad accesso diretto di dimensione m e con un fattore di carico $\alpha = 1$. Nell'ipotesi che le collisioni tra gli n elementi vengano gestite utilizzando le liste di trabocco, quanto costa nel caso peggiore trovare un elemento?
   a) α b) $O(1)$ c) $O(\log n)$ *d) $O(n)$
8. Dato il grafo in figura, si applichi su di esso l'algoritmo di Kruskal.
   Qual è la sequenza di archi inserita nella soluzione?
   a) $(e, g), (b, f), (c, f), (b, h), (c, d), (d, e), (i, l), (h, i), (f, i), (a, h)$
   *b) $(e, g), (b, f), (c, f), (b, h), (c, d), (d, e), (i, l), (h, i), (l, m), (a, h)$
   c) $(e, g), (b, f), (c, f), (b, h), (b, c), (d, e), (i, l), (h, i), (l, m), (a, h)$
   d) $(e, g), (b, f), (c, f), (b, h), (c, d), (d, e), (f, g), (h, i), (l, m), (a, h)$



9. Dato il grafo in figura, si applichi su di esso l'algoritmo di Borůvka guardando i vertici in ordine alfabetico.
   In quante fasi convergerà l'algoritmo?
   a) 1 *b) 2 c) 3 d) 4
10. Sia dato un grafo pesato $G = (V, E)$ con n nodi ed m archi, senza cicli negativi, e si consideri il problema di trovare i cammini minimi in G tra tutte le coppie di nodi. Quando è conveniente (asintoticamente) applicare l'algoritmo di Floyd&Warshall rispetto ad un'applicazione ripetuta dell'algoritmo di Dijkstra con heap binomiali?
    *a) $m = \omega(n^2/\log n)$ b) $m = \Theta(n)$ c) per ogni valore di m d) per nessun valore di m

Griglia Risposte

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