Università degli Studi dell'Aquila

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

Giovedì 11 Febbraio 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 $\times$ 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 finale è ottenuto sommando i punti ottenuti e normalizzando su base 30. Se tale somma è negativa, verrà assegnato 0.

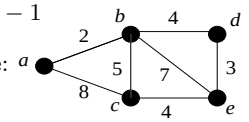
1. Siano $f(n)$ e $g(n)$ i costi dell'algoritmo Insertion Sort (basato sul ciclo `while`) nel caso medio e Selection Sort in quello migliore, rispettivamente. Quale delle seguenti relazioni asintotiche è falsa:
   *a) $f(n) = o(g(n))$ b) $f(n) = \Theta(g(n))$ c) $f(n) = O(g(n))$ d) $f(n) = \Omega(g(n))$
2. Durante l'esecuzione del QuickSort, applicando la procedura di partizione *in loco* al vettore $[23, 42, 7, 93, 15, 1, 27]$, con perno l'elemento 23, si ottiene
   *a) $[15, 1, 7, 23, 93, 42, 27]$ b) $[7, 1, 15, 23, 93, 42, 27]$ c) $[1, 7, 15, 23, 42, 93, 27]$ d) $[1, 7, 15, 23, 27, 42, 93]$
3. L'altezza dell'albero di decisione associato all'algoritmo HeapSort è:
   *a) $\Theta(n \log n)$ b) $\Omega(n!)$ c) $\omega(n \log n)$ d) $\Omega(n^2)$
4. Quale tra i seguenti rappresenta lo pseudocodice dell'algoritmo FixHeap di un max-heap binario:

   *a)
   FixHeap(nodo v, heap binario H)
   if (v è una foglia) **then** return
   else
   sia u il figlio di v con chiave massima
   if (`chiave`(v) < `chiave`(u)) **then**
   scambia `chiave`(u) e `chiave`(v)
   FixHeap(u, H)

   b)
   FixHeap(nodo v, heap binario H)
   if (v è una foglia) **then** return
   else
   sia u il figlio di v con chiave massima
   if (`chiave`(u) < `chiave`(v)) **then**
   scambia `chiave`(u) e `chiave`(v)
   FixHeap(u, H)

   c)
   FixHeap(nodo v, heap binario H)
   if (v è una foglia) **then** return
   else
   sia u il figlio di v con chiave minima
   if (`chiave`(v) < `chiave`(u)) **then**
   scambia `chiave`(u) e `chiave`(v)
   FixHeap(u, H)

   d)
   FixHeap(nodo v, heap binario H)
   if (v è una foglia) **then** return
   else
   sia u il figlio di v con chiave minima
   if (`chiave`(v) < `chiave`(u)) **then**
   scambia `chiave`(u) e `chiave`(v)
   FixHeap(v, H)
5. Come viene eseguita in un min-heap binomiale l'operazione *increaseKey*(e, Δ) di un elemento e con chiave k?
   a) *decreaseKey*(e, ∞) seguita da *deleteMin*() *b) *delete*(e) seguita da *insert*$(e, k + \Delta)$
   c) *decreaseKey*(e, ∞) seguita da *insert*$(e, k + \Delta)$
   d) Decrementando la chiave e ripristinando la proprietà dell'heap spingendo l'elemento verso il basso tramite ripetuti scambi.
6. Sia dato un AVL di n elementi nel quale si eseguono in successione $O(\log n)$ cancellazioni e $\Theta(1)$ inserimenti. Nel caso peggiore, quante rotazioni subirà l'AVL?
   a) $\Theta(1)$ b) $\Theta(n)$ c) $\Theta(\log n)$ *d) $\Theta(\log^2 n)$
7. Sia n_h il numero di nodi di T_h, ovvero l'albero di Fibonacci di ordine h, e sia F_h l'h-esimo numero della sequenza di Fibonacci. Risulta:
   a) $n_h = F_{h+1} + 3$ b) $n_h = F_{h+2} - 1$ c) $n_h = F_{h+2} + 1$ *d) $n_h = F_{h+3} - 1$
8. L'albero dei cammini minimi con sorgente a del grafo in figura ha peso totale:
   a) 18 *b) 14 c) 15 d) 21



9. 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 strettamente più efficiente (asintoticamente) applicare l'algoritmo di Floyd&Warshall rispetto ad un'applicazione ripetuta dell'algoritmo di Dijkstra con heap di Fibonacci?
   a) $m = \omega(n^2/\log n)$ b) $m = \Theta(n)$ c) per ogni valore di m *d) per nessun valore di m
10. 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 strettamente più efficiente dell'implementazione di Kruskal con alberi *QuickUnion* con euristica di bilanciamento *union by size*?
    a) $m = \Theta(n)$ b) sempre c) mai *d) $m = \omega(n)$

Griglia Risposte

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