Università degli Studi dell'Aquila

Prova di recupero di **Algoritmi e Strutture Dati**

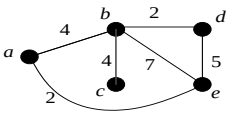
Martedì 27 Gennaio 2009 – Prof. Guido Proietti

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

ESERCIZIO 1 (25 punti): Domande a risposta multipla

Premessa: Questa parte è costituita da 20 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 relativo a questa parte è ottenuto sommando i punti ottenuti e normalizzando su base 25. Se tale somma è negativa, verrà assegnato 0.

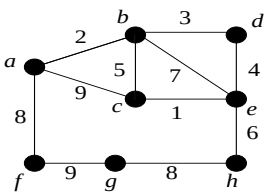
1. L'algoritmo più efficiente per il calcolo dell' n-esimo numero della sequenza di Fibonacci ha complessità
   a) $\Omega(n)$ b) $\Theta(n)$ *c) $O(\log n)$ d) $\Theta(n \log n)$
2. L'algoritmo Insertion Sort, nel caso medio costa: a) $O(n)$ b) $\omega(n^2)$ *c) $\Omega(n^2)$ d) $\Theta(n)$
3. Se $f(n) = \Omega(g(n))$ e $g(n) = \Omega(h(n))$, allora:
   a) $h(n) = \Omega(f(n))$ *b) $f(n) = \Omega(h(n))$ c) $f(n) = \Theta(h(n))$ d) $f(n) = \omega(h(n))$
4. Il numero di foglie dell'albero di decisione associato al problema dell'ordinamento di n elementi è:
   a) $\Theta(n \log n)$ b) $\omega(n!)$ c) $O(n \log n)$ * d) $\Omega(n!)$
5. Dato un problema con una delimitazione inferiore alla complessità temporale pari a $\Omega(f(n))$, un algoritmo per la sua risoluzione non può avere tempo di esecuzione $g(n)$ pari a:
   a) $g(n) = \Theta(f(n))$ *b) $g(n) = o(f(n))$ c) $g(n) = \omega(f(n))$ d) $g(n) = O(f(n))$
6. A quale delle seguenti classi non appartiene la complessità dell'algoritmo Merge Sort:
   a) *$o(n \log n)$ b) $\Omega(n)$ c) $O(n^2)$ d) $\Theta(n \log n)$
7. Nel caso medio, assumendo che le istanze siano equidistribuite, la ricerca di un elemento in un insieme non ordinato di n elementi richiede un numero di confronti pari a: a) n b) $(n-1)/2$ *c) $(n+1)/2$ d) 1
8. 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) $[1, 7, 15, 23, 27, 42, 93]$
9. 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 $n^4 + \log n$?
   a) $\Theta(n + k)$ b) $\Theta(n)$ *c) $O(n^4)$ d) $\Theta(n \log n)$
10. Un heap binario di altezza 3 contiene:
    a) esattamente 15 elementi b) almeno 16 elementi c) al più 7 elementi *d) tra 8 e 15 elementi
11. Una coda di priorità realizzata con una lista non ordinata supporta l'estrazione del massimo in:
    a) $\Theta(\log n)$ b) $O(\log n)$ c) $\Theta(1)$ *d) $\Theta(n)$
12. Dato un albero binario di ricerca di n elementi ed altezza h, l'inserimento di un elemento restituisce un albero avente al massimo altezza: *a) $h + 1$ b) $\Theta(\log h)$ c) $\Theta(\log n)$ d) h
13. In un albero AVL di n elementi, l'inserimento di un elemento nel caso peggiore induce un numero di rotazioni pari a:
    a) 0 *b) 2 c) $\Theta(\log n)$ d) 1
14. Siano X e Y due stringhe di lunghezza m ed n. Qual è la complessità dell'algoritmo per la determinazione della distanza tra X e Y basato sulla tecnica della programmazione dinamica?
    *a) $O(mn)$ b) $O(n)$ c) $O(m + n)$ d) $O(m)$
15. Quanti archi contiene il sottografo indotto da $\{a, c, e\}$ nel seguente grafo?
    a) 0 *b) 1 c) 2 d) 3



16. La visita in ampiezza dell'albero di cui alla Domanda 15 eseguita partendo dal nodo c restituisce un BFS di altezza:
    a) 1 *b) 2 c) 3 d) 4
17. L'algoritmo di Bellman e Ford applicato ad un grafo pesato con un numero di archi $m = \Theta(n^2)$, ha complessità:
    a) $\Theta(n^2)$ b) $\Theta(n + m)$ *c) $\Theta(n^3)$ d) $O(m \log n)$
18. Dato un grafo pesato $G = (V, E)$ con n vertici ed $m > n$ archi, e presi 2 vertici u, v, trovare il cammino minimo tra u e v applicando l'algoritmo di Dijkstra che usa l'heap binario costa:
    a) $\Theta(n)$ b) $\Theta(m)$ c) $\Theta(1)$ *d) $\Theta(m \log n)$
19. Usando gli alberi *QuickUnion* e l'euristica dell'unione pesata *by size*, il problema della gestione di n insiemi disgiunti sottoposti ad $n - 1$ *Union* ed $m = n^2$ *Find* può essere risolto in:
    a) $\Theta(n)$ b) $\Theta(n + m)$ c) $\Theta(n^2)$ *d) $O(n^2 \log n)$
20. Dato un grafo connesso con n vertici ed m archi, l'algoritmo di Kruskal esegue un numero di operazioni *Union*(u, v) pari a:
    a) $2m$ *b) $n - 1$ c) $\Theta(m \log n)$ d) $\Theta(\log n)$

Griglia Risposte

	Domanda																			
Risposta	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
a																				
b																				
c																				
d																				



ESERCIZIO 2 (5 punti) (**Da svolgere sul retro della pagina!**)

Mostrare l'intera esecuzione, passo per passo, dell'algoritmo di Kruskal sul seguente grafo: