# Università degli Studi dell'Aquila

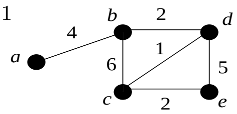
Prova di recupero di **Algoritmi e Strutture Dati**

Martedì 15 Luglio 2008 – Prof. Guido Proietti

| | | |
|---|---|---|
| Scrivi i tuoi dati $\Longrightarrow$ | Cognome: ........................ Nome: .................. Matricola: ................. | PUNTI |
| ESERCIZIO 1 | Risposte Esatte: Risposte Omesse: Risposte Errate: | |

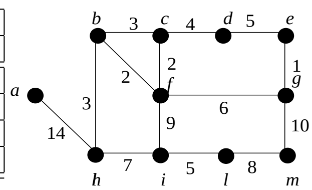
**ESERCIZIO 1 (25 punti): Domande a risposta multipla**

**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 $\times$ 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 25. Se tale somma è negativa, verrà assegnato 0.

1. Quale delle seguenti relazioni di ricorrenza definisce la sequenza di Fibonacci?
   a) $F_{n+1} = F_n + F_{n-1}$ se $n \geq 3$, $F_1 = F_2 = 1$  b) $F_{n-1} = F_{n-2} + F_{n-3}$ se $n \geq 3$, $F_1 = F_2 = 1$
   *c) $F_n = F_{n-1} + F_{n-2}$ se $n \geq 3$, $F_1 = F_2 = 1$  d) $F_n = F_{n-1} + F_{n-2}$ se $n \geq 2$, $F_1 = 1$
2. Sia $f(n) = n^2 + 2$; affinché sia $f(n) = O(n^2)$, è sufficiente scegliere:
   *a) $n_0 = 1, c = 3$  b) $n_0 = 1, c = 1$  c) $n_0 = 2, c = 1$  d) $n_0 = 1, c = 2$
3. Se $f(n) = n \log \sqrt{n}$ e $g(n) = n \log^2 n$, quale delle seguenti relazioni asintotiche è falsa:
   a) $f(n) = o(g(n))$  b) $f(n) = O(g(n))$  c) $g(n) = \Omega(f(n))$  *d) $f(n) = \Theta(g(n))$
4. 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
5. La delimitazione inferiore al problema dell'ordinamento ottenibile dagli alberi di decisione è:
   a) $o(n \log n)$  b) $\omega(n \log n)$  *c) $\Theta(n \log n)$  d) $\Theta(n)$
6. L'algoritmo di ordinamento non crescente Insertion Sort applicato ad una sequenza di input ordinata in modo non crescente esegue un numero di confronti tra elementi pari a:
   *a) $n-1$  b) $n$  c) $n+1$  d) $n(n-1)/2$
7. L'algoritmo Selection Sort, nel caso migliore costa:
   a) $o(n^2)$  *b) $\Theta(n^2)$  c) $O(n \log n)$  d) $\Theta(n \log n)$
8. A quale delle seguenti classi appartiene la complessità del caso medio dell'algoritmo Quicksort:
   a) $\Theta(n^2)$  *b) $\Theta(n \log n)$  c) $O(n)$  d) $\Omega(n^2)$
9. La delimitazione inferiore al problema della ricerca di un elemento in un insieme non ordinato di $n$ elementi è:
   a) $\Theta(\log n)$  b) $\Theta(n \log n)$  *c) $\Omega(n)$  d) $\Omega(n \log n)$
10. In un albero binario di ricerca di altezza $h$, il *successore* di un elemento può essere determinato in:
    a) $\Theta(\log h)$  b) $O(\log h)$  c) $\Theta(1)$  *d) $O(h)$
11. Dati due elementi $u, v$ appartenenti ad un universo totalmente ordinato $U$, una funzione hash $h(\cdot)$ si dice *perfetta* se:
    a) $u = v \Rightarrow h(u) \neq h(v)$  b) $u \neq v \Rightarrow h(u) = h(v)$  c) $u = v \Rightarrow h(u) = h(v)$  *d) $u \neq v \Rightarrow h(u) \neq h(v)$
12. Qual è la distanza tra le stringhe *tredici* e *sedici*?
    a) 0  b) 1  *c) 2  d) 3
13. Una coda di priorità realizzata con una lista lineare ordinata supporta l'estrazione del massimo in:
    *a) $O(1)$  b) $\Theta(n)$  c) $\Theta(\log n)$  d) $\Omega(\log n)$
14. Un grafo *non connesso* di $n$ vertici, ha un numero minimo di archi pari a:
    *a) 0  b) $n-1$  c) $n-2$  d) 1
15. La visita in ampiezza del grafo  eseguita partendo dal nodo $a$ genera un albero BFS di altezza pari a:
    
    a) 1  b) 2  *c) 3  d) 4
16. La *visita in ampiezza* di un grafo connesso con $n$ vertici ed $m$ archi rappresentato tramite liste di adiacenza, può essere eseguita in:  *a) $O(n+m)$  b) $\Omega(n^2)$  c) $O(n)$  d) $\Theta(n)$
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 con $n$ vertici ed $m = \Theta(n \log n)$ archi, l'algoritmo di Dijkstra realizzato con heap binomiali costa:
    a) $\Theta(n^2)$  b) $\Theta(n+m)$  c) $O(m)$  *d) $O(n \log^2 n)$
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 Dijkstra sul seguente grafo, assumendo il nodo $h$ quale nodo sorgente (ogni figura disegnata deve rappresentare lo stato dei nodi del grafo prima di visitare il prossimo nodo).