Università degli Studi dell'Aquila

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

Martedì 5 Settembre 2017 – 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: Questa parte 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 $\times$ 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 30. Se tale somma è negativa, verrà assegnato 0.

1. $f(n) = O(g(n))$ se:
   a) Esistono due costanti $c \geq 0$ e $n_0 \geq 0$ tali che $f(n) \geq c \cdot g(n)$ per ogni $n \geq n_0$.
   *b) Esistono due costanti $c > 0$ e $n_0 \geq 0$ tali che $f(n) \leq c \cdot g(n)$ per ogni $n \geq n_0$.
   c) Esistono due costanti $c > 0$ e $n_0 \geq 0$ tali che $f(n) \geq c \cdot g(n)$ per ogni $n \leq n_0$.
   d) Esistono due costanti $c > 0$ e $n_0 \geq 0$ tali che $f(n) \geq c \cdot g(n)$ per ogni $n \geq n_0$.
2. Un algoritmo ha una complessità temporale $O(f(n))$ se:
   a) Il tempo di esecuzione $T(n)$ dell'algoritmo su uno specifico input di dimensione n verifica $T(n) = O(f(n))$
   *b) Il tempo di esecuzione $T(n)$ dell'algoritmo su ogni input di dimensione n verifica $T(n) = O(f(n))$
   c) Il tempo di esecuzione medio $T(n)$ dell'algoritmo su un input di dimensione n verifica $T(n) = \Theta(f(n))$
   d) Nel caso migliore, il tempo di esecuzione $T(n)$ dell'algoritmo su un input di dimensione n verifica $T(n) = o(f(n))$
3. Siano $f(n)$ e $g(n)$ i costi dell'algoritmo Selection Sort nel caso migliore e in quello peggiore, rispettivamente. Quale delle seguenti relazioni asintotiche è falsa:
   *a) $f(n) = o(g(n))$ b) $f(n) = \Omega(g(n))$ c) $g(n) = O(f(n))$ d) $f(n) = \Theta(g(n))$
4. Quale istruzione va aggiunta al seguente pseudocodice affinché gli n elementi dell'array A siano ordinati in ordine non crescente (il primo elemento dell'array è indicizzato come $A[1]$):

```
Sort(A)
  for (k = 1) to n − 1 do
    m = k
    for (j = n) downto k + 1 do
      ...
    scambia A[m] con A[k]
```

   *a) if $(A[j] > A[m])$ then $m = j$ b) if $(A[j] < A[m])$ then $m = k$
   c) if $(A[j] > A[m])$ then $m = k$ d) if $(A[j] < A[m])$ then $m = j$
5. La procedura *Heap-Insert*$(A, 10)$ applicata al vettore $A = [12, 9, 3, 6, 5, 2, 1, nil]$ rappresentante un heap binario restituisce:
   a) $A = [10, 9, 12, 6, 5, 3, 2, 1]$ b) $A = [12, 9, 10, 6, 5, 3, 2, 1]$ c) $A = [12, 9, 10, 6, 5, 2, 3, 1]$ *d) $A = [12, 10, 3, 9, 5, 2, 1, 6]$
6. Dato un albero AVL contenente n elementi, si consideri su di esso una sequenza di $k < n$ cancellazioni di elementi. La nuova altezza dell'AVL diventa:
   a) $\Theta(n - k)$ b) $\Theta(\log(n) - k)$ *c) $\Theta(\log(n - k))$ d) $\Theta(\log n)$
7. Sia $h(\cdot)$ una funzione hash, e siano c_1 e c_2 due costanti positive. Quale delle seguenti funzioni descrive il metodo di scansione quadratico in una tabella hash di dimensione m per l'inserimento di un elemento con chiave k dopo l'i-esima collisione, per $0 \leq i < m$:
   a) $c(k, i) = (h(k) + c_1 \cdot i + c_2 \cdot i) \mod m$ b) $c(k, i) = (h^2(k) \cdot i) \mod m$ *c) $c(k, i) = (h(k) + c_1 \cdot i + c_2 \cdot i^2) \mod m$
   d) $c(k, i) = i^2 \mod m$
8. L'algoritmo di Bellman e Ford applicato ad un grafo pesato con un numero di archi $m = \Theta(n \log n)$, ha complessità:
   a) $\Theta(n^2)$ b) $\Theta(n + m)$ c) $\Theta(n^3)$ *d) $O(n^2 \log n)$
9. Dato un grafo connesso con n vertici ed m archi rappresentato tramite liste di adiacenza, l'algoritmo di Dijkstra realizzato con una *lista lineare ordinata* costa:
   *a) $O(mn)$ b) $\Theta(n + m)$ c) $\Theta(m \log n)$ d) $O(m \log n)$
10. Dato un grafo connesso e pesato con pesi distinti con n vertici ed m archi, l'algoritmo di Borůvka alla prima passata aggiunge alla soluzione un numero massimo di archi pari a:
    *a) $n - 1$ b) $\log n$ c) 1 d) $\lceil n/2 \rceil$

Griglia Risposte

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