Algoritmi e strutture dati

Allievi della Laurea Specialistica in Ingegneria Informatica
Tema d'esame - 26 Marzo 2010 – ore 9.00-10.30

NOME: **COGNOME:**
MATRICOLA: **FIRMA:**

Il valore degli
esercizi è riportato
nel prospetto a lato.

Esercizio	1	2	3	4	5	6a	6b	7a	7b	7c
Valore	2	2	2	3	4	3	3	5	3	3
Valutazione										

Esercizi

- 1) Definire i concetti di albero e foresta, puntualizzando la differenza fra gli stessi.
- 2) Uno heap d-ario è come uno heap binario ma i nodi (non foglia) dello heap hanno d figli invece di 2. Stabilire se la sequenza $\langle 23, 17, 14, 6, 13, 10, 1, 5, 7, 12 \rangle$ è uno heap ternario, motivando la risposta.
- 3) Illustrare l'esecuzione di RADIX-SORT sulla seguente lista di parole: DOG, RAG, DIM, BUG, NEW, MIX.
- 4) Si consideri una tabella hash con concatenazione di lunghezza $m = 17$, la cui funzione hash è costruita per moltiplicazione con $A = 0.618$. Effettuando i calcoli con troncamento dei valori inferiori a 10^{-3} , simulare l'inserimento prima della chiave 12, poi della chiave 22.
- 5) Scrivere e risolvere l'equazione di ricorrenza relativa al tempo di esecuzione del seguente algoritmo.

BOH(A, n)

1 $k \leftarrow 0$

2 **for** $i \leftarrow 1$ **to** n

3 **do** $k \leftarrow k + A[i] / n$

4 **if** $(k > 44)$

5 **then return** $3 * BOH(A, n - 2) + k$

6 **else return** k

- 6) Si consideri il problema definito come segue.

Input: un vettore A, contenente una sequenza di n ($n > 0$) valori interi, e un intero x .

Output: due interi distinti h e k , $1 \leq h \leq n$, $1 \leq k \leq n$, tali che $A[h] + A[k] = x$;

due valori interi identici se non esiste alcuna coppia di indici distinti h e k del vettore A, $1 \leq h \leq n$, $1 \leq k \leq n$, tale che $A[h] + A[k] = x$.

- a) Scrivere lo pseudocodice di un algoritmo che risolva tale problema.
- b) Analizzare la complessità temporale dell'algoritmo proposto.

- 7) Per profondità di un nodo di un albero si intende il numero di archi sul cammino che va dalla radice al nodo stesso. Si consideri il problema definito come segue.

Input: un albero binario T di n nodi ($n > 0$) e un intero k , $0 \leq k \leq n - 1$; ogni nodo di T contiene (come dato) un carattere.

Output: il contenuto dei nodi di T visitati come segue: dapprima tutti i nodi di T di profondità k , visitati da sinistra a destra, poi tutti i nodi di T di profondità $k - 1$, sempre visitati da sinistra a destra, e così via fino a visitare la radice.

- a) Definire in pseudocodice un algoritmo ricorsivo che risolva tale problema.
- b) Determinare la complessità temporale dell'algoritmo proposto.
- c) Ricordando che, a ogni nuova chiamata di una funzione ricorsiva, il contesto corrente della funzione (ovvero variabili locali e indirizzo dell'istruzione a cui ritornare) viene salvato su una pila, determinare la complessità spaziale dell'algoritmo proposto.