Algoritmi e strutture dati

Allievi della Laurea Specialistica in Ingegneria Informatica
Tema d'esame - 26 Novembre 2009 – ore 9.00-10.30

NOME: **COGNOME:**

MATRICOLA: **FIRMA:**

Il valore degli
esercizi è riportato
nel prospetto a lato.

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

Esercizi

- 1) Quali sono gli algoritmi di ordinamento *per confronto* presentati durante il corso e perché si chiamano così? Cosa si intende quando si afferma che un algoritmo di ordinamento per confronto è *asintoticamente ottimo*? Quali algoritmi di ordinamento per confronto sono asintoticamente ottimi?
- 2)
 - a) Definire la proprietà di stabilità di un algoritmo di ordinamento.
 - b) Indicare se l'algoritmo HeapSort sia stabile o meno e dimostrare quanto affermato.
- 3) Si consideri una tabella hash di lunghezza $m = 13$ gestita mediante hashing doppio con
$$h_1(k) = k \bmod m$$
$$h_2(k) = 1 + k \bmod (m - 1)$$
Simulare l'inserimento delle chiavi 11, 22, 31, 9, 18, 44, 37 nell'ordine dato.
- 4) Un algoritmo ricorsivo impiega un tempo lineare per individuare il sottoproblema (del problema corrente) da affrontare mediante una chiamata ricorsiva (passo DIVIDE) e, una volta che tale chiamata è terminata, per utilizzare la soluzione da essa prodotta al fine di risolvere il problema corrente (passo COMBINA). La dimensione del sottoproblema affrontato dalla chiamata ricorsiva (passo IMPERA) è pari ai 7/10 della dimensione del problema corrente.
Scrivere e risolvere l'equazione di ricorrenza relativa al tempo di esecuzione dell'algoritmo.
- 5) Il problema di *ricerca* è definito come segue.
Input: una sequenza di n numeri $A = \langle a_1, a_2, \dots, a_n \rangle$ e un valore v .
Output: un indice i tale che $A[i] = v$, oppure il valore speciale NIL se v non è presente in A .
L'algoritmo di *ricerca binaria* assume che la sequenza di ingresso sia ordinata (precondizione) e risolve il problema di ricerca ripetendo il seguente passo: confronta v con l'elemento che occupa la posizione centrale ed elimina da ulteriori confronti metà della sequenza.
 - a) Scrivere lo pseudocodice di una versione ricorsiva dell'algoritmo di ricerca binaria.
 - b) Analizzare 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.

- 6) 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 (con radice $\text{root}[T]$) di n nodi ($n > 0$) e un intero k , $0 \leq k \leq n - 1$.

Output: il numero dei nodi di T che hanno profondità k .

- a) Definire in pseudocodice una procedura ricorsiva che risolva tale problema.
- b) Determinare la complessità temporale dell'algoritmo proposto in funzione delle due dimensioni n e k .