Introduzione agli Algoritmi
Appello del 14 novembre 2013
Proff. Emanuela Fachini ed Irene Finocchi

Le risposte non motivate non saranno prese in considerazione.

Negli esercizi di progettazione, prima di passare allo pseudocodice descrivete l'idea algoritmica soggiacente.

Per tutti gli algoritmi progettati è necessario analizzare tempo di esecuzione e correttezza (canale 1: usare gli invarianti per progettare e verificare il ciclo in caso di algoritmi iterativi).

Parte 1

Esercizio 1 (punti 14)

Si considerino le seguenti funzioni:

```
fun (intero n) {
        i=0;
        for a=1 to n do
            test(a);
}
```

```
test (intero x)
        if x≤1 then return
        for i ← 1 to 4 do
            test(x/2)
```

Scrivere la relazione di ricorrenza che descrive il tempo di esecuzione T(x) della funzione test e risolverla. Calcolare poi il tempo di esecuzione F(n) della funzione fun.

Esercizio 2 (punti 18)

Sia A un array di n valori distinti. Si dice che A "*richiede k scansioni*" se dobbiamo scandire A esattamente k volte da sinistra a destra per leggere tutti i suoi elementi in ordine crescente.

Ad esempio, sia A = 4 9 1 8 2 5 3 6 7. Per l'array A sono richieste 4 scansioni, ovvero k=4. Nella prima scansione vengono letti in ordine crescente 1 2 3, nella seconda scansione 4 5 6 7, nella terza scansione 8 e nella quarta scansione 9.

Si proponga un algoritmo, il più possibile efficiente, per calcolare il numero di scansioni richieste per A, ovvero il valore k. Analizzare la correttezza, utilizzando gli invarianti nel caso di cicli (canale 1) o per induzione (canale 2), e il tempo di esecuzione dell'algoritmo proposto.

Introduzione agli Algoritmi
Appello del 14 novembre 2013
Proff. Emanuela Fachini ed Irene Finocchi

Le risposte non motivate non saranno prese in considerazione.

Negli esercizi di progettazione, prima di passare allo pseudocodice descrivete l'idea algoritmica soggiacente.

Per tutti gli algoritmi progettati è necessario analizzare tempo di esecuzione e correttezza (canale 1: usare gli invarianti per progettare e verificare il ciclo in caso di algoritmi iterativi).

Parte 2

Esercizio 3. (punti 16)

1. Definire le proprietà dei maxHeap.
2. Dimostrare che l'altezza di un MaxHeap con n nodi è $\Theta(\log(n))$.
3. Discutere la veridicità delle seguenti affermazioni, fornendo una dimostrazione oppure un controesempio:
    a. L'inserimento in un maxHeap è commutativo, ovvero se si inserisce in H prima un intero k e poi un altro intero $h \neq k$, si ottiene lo stesso albero che sia otterrebbe inserendo prima h e poi k.
    b. Il maxHeap supporta le operazioni di inserimento, cancellazione e ricerca di un elemento con tempi di esecuzione analoghi a quelli degli alberi binari di ricerca.
    c. Se ci sono elementi uguali in un maxHeap, essi sono in una relazione di antenato – discendente.

Esercizio n. 4 (punti 16)

Dati un albero binario di ricerca T di altezza h ed un intero k, vogliamo individuare il k^{mo} elemento di T in tempo O(h). Si assuma che l'albero è rappresentato tramite puntatori ai figli sinistro e destro e che ogni nodo contenga anche un campo *dim* con l'indicazione del numero dei nodi nel sottoalbero in esso radicato. Analizzare la correttezza e il tempo di esecuzione dell'algoritmo proposto.