Algoritmi e Strutture Dati (modulo I)

Testo della prova scritta del 15 settembre 2014

docente: Luciano Gualà

Cognome:..... Nome:..... Matr.:..... Corso di Laurea:.....

Esercizio 1 [10 punti]

(a) Si ordinino le seguenti funzioni in ordine non decrescente di tasso di crescita asintotica. Per ogni coppia di funzioni $f_i(n), f_{i+1}(n)$ adiacenti nell'ordinamento si specifichi se $f_i(n) = \Theta(f_{i+1}(n))$ o se $f_i(n) = o(f_{i+1}(n))$.

Le funzioni sono: $2^{n+\log n}$, $\frac{n^3\sqrt{n+1}}{\sqrt{n^4+n(n^2+1)}}$, $n\sqrt{n+6}+n\log n$, 2^n, $n^2\log\log n$, $n^2\sqrt{\log n}$, $\frac{n^3+1}{n^{0.9}-1}$, $2^{n\log n}$, $\sqrt{1+n^2\log\log n}$.

(b) Per un problema sono noti due algoritmi ricorsivi, A_1 e A_2 le cui complessità temporali sono descritte dalle seguenti equazioni di ricorrenza:

$T_1(n) = 2\,T_1(n-4)+1$;

$T_2(n) = 27\,T_2(n/3)+n^2\sqrt{n^2+1}$;

Dire, motivando la risposta, quale algoritmo è preferibile usare.

Esercizio 2 [12 punti] Sia T un albero binario rappresentato tramite una struttura dati collegata in cui il record di un nodo v contiene le seguenti informazioni: il colore del nodo $c(v) \in \{B, N\}$ che può essere bianco o nero, un puntatore $p(v)$ al padre e due puntatori $s(v)$ e $d(v)$ rispettivamente al figlio sinistro e al figlio destro. La profondità di un nodo v è la lunghezza (intesa come numero di archi) del cammino fra la radice e v. Il livello i dell'albero è il sottoinsieme di nodi di T che hanno tutti profondità i. Progettare un algoritmo con complessità lineare che verifica se l'albero ha o meno un livello interamente nero, ovvero un livello composto di soli nodi neri. Si fornisca lo pseudocodice dettagliato dell'algoritmo.

Esercizio 3 [13 punti]

Sonic si trova su una piattaforma composta da n caselle e deve raccogliere più anelli possibile. Le caselle, da sinistra a destra, sono numerate da 1 a n. Lui si trova nella casella 1 e il traguardo che deve superare è alla casella n. Ogni casella i ha un certo numero t_i di anelli a terra e un certo numero a_i di anelli in aria. Sonic ogni volta può fare due mosse: *spostarsi* a destra o *saltare* a destra. Se lui si trova nella casella i e si sposta a destra, allora finisce nella casella $i+1$ e raccoglie gli anelli a terra che si trovano in tale casella. Se invece decide si saltare a destra, allora si sposta nella casella $i+3$, raccoglie gli anelli che si trovano a terra nella casella $i+3$ e quelli che si trovano in aria nelle caselle $i+1$ e $i+2$. Progettare un algoritmo di programmazione dinamica che calcola in numero massimo di anelli che Sonic può raccogliere prima di tagliare il traguardo. Si assuma che non è necessario finire a terra nella casella n, ma è possibile tagliare il traguardo anche in volo.