Esame di Programmazione 1 22 marzo 2006

Il tempo a disposizione è 2 ore. Gli esercizi più importanti sono (1) e (2). Tentate il (3) solo dopo aver fatto i primi 2.

1)Parte iterativa:
Si scriva un programma che legge da cout i seguenti valori interi:
1)3 interi d1, d2, d3;
2) e successivamente una sequenza n0,..,nm di interi che si conclude con un doppio 0;
Il programma deve allocare un array int A[d1*d2*d3] interi e leggere in A i valori n0…nm finchè stanno in A o si incontrano i due 0 uno dopo l'altro.

Successivamente il programma deve trattare A come fosse una matrice int [d1][d2][d3] e stabilire quanti dei d1 strati (di dimensioni [d2][d3]) della matrice hanno tutte le righe composte da valori tutti diversi.

Esempio: sia d1=3, d2=2 e d3=2 ed i valori che seguono siano: 2 3 2 1 1 1 1 0 0, quindi il primo strato [2][2] contiene nella prima riga 2 e 3 e nella seconda riga, 2 e 1. Il secondo strato contiene nella prima riga, 1 e 1 e nella seconda riga solo 1. Quindi la seconda riga del secondo strato non è completa mentre il terzo strato è vuoto.
Il primo strato ha tutte e 2 le righe con elementi diversi tra loro, mentre nel secondo strato la prima riga contiene due volte 1. Quindi il risultato che il programma dovrebbe determinare è 1, cioè 1 solo strato della matrice ha ogni riga composta da elementi tutti diversi.

2)Parte ricorsiva:
Si scriva una funzione ricorsiva che riceve un albero binario e determina il nodo a profondità minima che abbia al più 1 figlio. La funzione deve restituire la profondità di questo nodo.

Esempio: si consideri l'albero binario, 34(10(2(_,5(_,_)),_),6(_,_)), contiene 2 nodi con al più 1 figlio ed a profondità minima, essi sono il nodo 10 ed il nodo 6 a profondità 1.
Quindi la funzione da fare deve restituire 1.

3)Domanda extra:
In quanto segue, per brevità, chiameremo **nodo+1** un nodo di un albero che possieda al più un figlio. Modificare la funzione della parte ricorsiva (2) in modo che costruisca una lista L di nodi che puntano ai nodi+1 dell'albero e tale che il primo nodo di L punti al nodo+1 a profondità minima, il secondo nodo di L punti al nodo+1 alla seconda profondità e così via. Il tipo dei nodi della lista L deve essere: *struct nodoL{int prof; nodo* info; nodoL* next;};*
il campo *prof* conterrà la profondità nell'albero del nodo puntato dal campo *info*.

Esempio: consideriamo l'albero dell'esempio precedente. La lista L deve contenere 4 nodi, il primo dei quali punta al nodo 10, il secondo al 6 (o viceversa visto che hanno entrambi profondità minimale = 1) il campo *prof* di entrambi questi nodi sarà 1, il terzo nodo di L punta a 2 con campo *prof*=2 e l'ultimo nodo di L punta a 5 con *prof*=3.