Settimana 7 Esercizi per casa (alberi)

Per cominciare. E' disponibile un kit di lavoro (kit_alberi.c) che contiene una funzione per visualizzare gli alberi in modo "intuitivo", ed un primo albero su cui lavorare.

Se non lo hai mai fatto, costruisci un albero, stampalo, calcola la somma di tutti i campi info dell'albero, moltiplica per due il valore nel campo info, stampalo.

Consiglio: su alberi e liste, e' essenziale lavorare carta e penna.

ES 1 Considera un albero binario. Ad ogni nodo e' associato un punteggio (registrato nel campo info).

(i) Scrivi una funzione max_punti che calcola il ***punteggio massimo che possiamo totalizzare percorrendo un cammino*** dalla radice alle foglie.

(ii) Vogliamo percorrere l'albero dalla radice ad una foglia. Per vincere dobbiamo totalizzare esattamente un certo punteggio: ne' far meno, ne' sforare… Scrivi una funzione esiste_cammino che dati un albero ed un intero tot, dice se ***c'e' un cammino dalla radice ad una foglia che permette di totalizzare esattamente tot*** punti.

ES 2 Si consideri un albero binario i cui nodi hanno tipo
struct nodo {int info; int help; nodo * left, * right;}.
Scrivere una funzione scrivi_somma che ***inserisce nel campo help*** di ogni nodo n ***la somma di tutti i campi info dell'albero radicato in n*** (n compreso).
Nota: per stampare i campi help, e' necessaria una piccola modifica della funzione di stampa.

ES 3 Scrivere una funzione ricorsiva F che riceve il puntatore root alla radice di un albero dato T e lo traversa costruendo una lista L tale che abbia tanti nodi quante sono le foglie di T e che ogni nodo di L punti ad una diversa foglia di T. F deve restituire al chiamante il puntatore all'inizio di L.

Aiuto: i nodi dell'albero hanno ovviamente tipo
struct nodo {int info; nodo * left, * right;},
mentre i nodi di L avranno tipo: struct nodo_L{nodo * info; nodo_L * next;};