Settimana 8 Esercizi per casa (preparazione all'appello)

ES 1 Vogliamo gestire valori da 0 a 999 con dieci array, allocati dinamicamente, di massimo 100 elementi.
L'idea e' che il primo array deve contenere gli elementi tra 0 e 99 (se ce ne sono), il secondo contiene gli elementi tra 100 e 199, e cosi via.
Quindi ogni valore da 0 a 999 ha un suo posto nella struttura,
ad esempio il valore 103 e' l'elemento di indice 3 del secondo array.

La struttura dati che utilizziamo e' un array X di puntatori ad interi, inizializzati a NULL. Durante l'esecuzione del programma, gli elementi di X saranno degli array allocati dinamicamente.

Si chiede di realizzare le operazioni di inserimento, ricerca e cancellazione, per gestire questa struttura dati.

Inserimento. Vogliamo inserire il valore N, nell'array appropriato, alla posizione che gli compete (vedi esempio sopra).
Se l' array non esiste ancora, lo si deve allocare, inizializzando tutti gli elementi a -1, e poi fare l'inserimento di N.
Nel caso N sia gia' presente in X, la funzione non fa niente.

Ricerca. La funzione *ricerca* testa se un certo valore N e' presente in X.

Cancellazione. La funzione *cancella* elimina il valore dato N, se presente in X. Nella cancellazione puo' capitare che uno degli array di 100 elementi si svuoti, in questo caso l'array va deallocato.

ES 2 Un cammino in un albero binario è descritto da una sequenza di 0 e 1 che termina con la sentinella -1. Vogliamo realizzare una funzione che determini la lunghezza massima dei cammini di un albero dato (cioè l'altezza dell'albero) e che al contempo costruisca la sequenza di 0 ed 1 di un particolare cammino di lunghezza massima. Ovviamente il cammino parte dalla radice dell'albero e termina in una foglia.

Più precisamente, **si chiede di realizzare** la funzione ricorsiva CAM_MAX che riceve come parametro (il puntatore alla radice di) un albero R ed inserisce in ogni nodo N dell'albero la seguente coppia di valori:
a) l'altezza H dell'albero radicato in N, cioè la lunghezza massima dei cammini da N ad una foglia (nell'albero con radice N)
b) il valore 0, 1 o -1 a seconda che il cammino di lunghezza H che CAM_MAX ha costruito per l'albero radicato in N prosegua, rispettivamente, col figlio sinistro di N, col figlio destro di N, oppure termini in N (se N è una foglia)

Per esempio, se consideriamo l'albero: X(X(_,_),X(_,X(_,X(_,_)))) la funzione CAM_MAX deve etichettare i suoi nodi come segue:
[3,1]([0,-1](_,_),[2,1](_,[1,1](_,[0,-1](_,_)))
in cui per ogni nodo abbiamo evidenziato tra parentesi [] i campi (a) e (b) nell'ordine.

Quindi i nodi degli alberi che consideriamo avranno il seguente tipo:
struct nodo {int lung, direz; nodo* left, *right;};
il campo **lung** conterrà la lunghezza del cammino massimo (H del punto (a)) ed il campo **direz** (per direzione) conterrà 0,1 o -1 (come descritto nel punto (b)).

Vogliamo anche stampare il cammino di lunghezza massima determinato da CAM_MAX. A questo fine **si chiede quindi di realizzare** una funzione STAMPA_CAM (iterativa o ricorsiva a piacere) da invocare dopo CAM_MAX e che soddisfa le seguenti specifiche.
STAMPA_CAM riceve come parametro (il puntatore alla radice dell'albero) R e stampa la sequenza di coppie [lung, direz] di tutti i nodi del cammino di lunghezza massima costruito da CAM_MAX (nell'ordine dalla radice alla foglia).

ES 3 Scrivere una funzione di prototipo (obbligatoriamente)
*int min (nodo * T)*
che dato un ***albero non vuoto*** restituisce il minimo tra i valori registrati nei campi info di tale albero.