Tecniche di Programmazione

Esercitazione 11

* Si considerino alberi binari con valori di tipo intero, **positivi**.
* Scrivere dei test nel main per verificare che le funzioni scritte siano corrette.
* Manipolare gli alberi solo tramite le funzioni del tipo astratto (vedere lista.h, albero.h), non accedere all’implementazione.
* Tutti gli esercizi devono essere risolti in modo ricorsivo, senza usare strutture dati ausiliarie (se non esplicitamente richiesto).

### Esercizio 11.1

Implementare la funzione

TipoInfoAlbero trova\_massimo(TipoAlbero a);

che, dato un albero binario, restituisca il valore massimo contenuto nei nodi dell'albero. Se l’albero è vuoto, si ritorni -1.

### Esercizio 11.2

Implementare la funzione:

TipoInfoAlbero somma\_foglie(TipoAlbero a);

che, dato un albero binario, restituisca la somma di tutti i valori contenuti nei nodi foglia.

### Esercizio 11.3

Implementare la funzione

int cerca\_livello(TipoAlbero a, TipoInfoAlbero v);

che, dati un albero binario a e un valore v, restituisca il livello dell’albero dove si trova v.

Se v non e’ presente all’interno dell’albero ritornare -1. Se sono presenti più nodi con lo stesso valore, si restituisca il livello del nodo più a sinistra.

### Esercizio 11.4

Implementare la funzione:

int conta\_dispari(TipoAlbero a);

Che, dato un albero binario, restituisca il numero di nodi che soddisfano la seguente condizione: la somma tra il valore del nodo e i suoi figli è dispari.

### Esercizio 11.5

Implementare la seguente funzione

int somma\_singoli(TipoAlbero a);

che, dato in input un albero binario, restituisca la somma dei valori dei nodi che hanno un solo successore.

### Esercizio 11.6

Implementare la seguente funzione:

TipoLista albero\_lista(TipoAlbero a);

Che, dato un albero binario di ricerca, restituisca la lista dei valori ordinati, in modo decrescente.

### 

### Esercizio 11.7

Implementare la seguente funzione

TipoLista percorso\_lungo(TipoAlbero a);

Che, dato un albero binario, restituisca la lista dei nodi contenuti nel percorso più lungo dalla radice a una delle foglie. Se esistono diversi percorsi di dimensione massima, si restituisca quello più a sinistra.

### Esercizio 11.8

Implementare la funzione:

void somma\_sottoalbero(TipoAlbero a);

che modifica il valore di ogni nodo dell’albero a, scrivendo come valore la somma dei valori dei sottoalberi di quel nodo.

### Esercizio 11.9

Implementare la funzione:

void scambia\_foglie(TipoAlbero a);

che scambia il contenuto di tutte le coppie di foglie che hanno lo stesso padre.