Informatica Generale
Homework 3/2019: Alberi binari

Ivano Salvo – Sapienza Università di Roma – 29/5/2019

Esercizio 1. Scrivere una funzione C di prototipo:

```
binTreeRat calkinWilf(int n)
   /* PREC: n>=0 */
```

che genera i primi n livelli dell'albero dei razionali di Calkin-Wilf (vedi Homework **1**). Assumendo che la radice sia al primo livello e contando verso il basso, l'albero nella traccia dell'HW **1** ha 4 livelli. Se n fosse 0, la funzione restituisce l'albero vuoto. Nei file messi a disposizione, viene data la definizione del tipo `binTreeRat`, che definisce alberi binari in cui l'etichetta di ciascun nodo è una coppia di interi positivi (che rappresentano quindi un razionale).

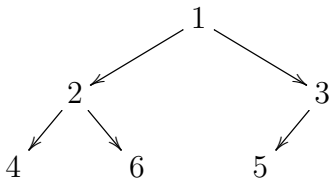
Suggerimento: Trovare una facile regola di generazione che permetta di calcolare i valori da memorizzare nei nodi figlio destro/sinistro a partire dal razionale memorizzato nel nodo corrente.

Esercizio 2. Scrivere una funzione C di prototipo:

```
binTree balancedTreeFromList(list L)
```

che mette gli interi contenuti in una lista L in un albero binario bilanciato. Procedere nel seguente modo: mettere il primo elemento della lista nella radice dell'albero e poi ricorsivamente gli elementi che in L occupano un posto pari nel sottoalbero sinistro e quelli che occupano un posto dispari (eccetto il primo, memorizzato nella radice) nel sottoalbero destro. **Non modificare** la lista L passata come parametro.

Esempio: Leggendo la lista $L = \langle 1, 2, 3, 4, 5, 6 \rangle$, deve essere prodotto il seguente albero:



Esercizio 3. Dati due alberi binari di interi A e B, diremo che A è *minore per tracce* di B (notazione $A \preceq B$) se per ogni sequenza di etichette p in un cammino di A, esiste un cammino in B in cui incontro la sequenza di etichette p. Gli alberi A e B sono *equivalenti per tracce* se e solo se $A \preceq B$ e $B \preceq A$.

Scrivere una funzione C di prototipo:

```
int eqTrace(binTree A, binTree B)
```

che ritorna 1 se gli alberi A e B sono equivalenti per tracce e 0 altrimenti.

Esempi: Presi gli alberi in Fig. 1, abbiamo chiaramente che $A \preceq C$ e $A \preceq B$. Osservate che la sequenza di etichette $\langle 11, 23, 19 \rangle$ si trova nell'albero B nel cammino contenente le etichette $\langle 11, 23, 19, 42 \rangle$. Questo cammino è il controesempio che mostra che $B \not\preceq A$ e $B \not\preceq C$. Infine anche $C \preceq A$ e quindi A e C sono equivalenti per tracce.

Suggerimento: Organizzare il codice in funzioni che risolvono sottoproblemi più semplici. Ovviamente è più semplice scrivere una funzione che implementa $\preceq$.

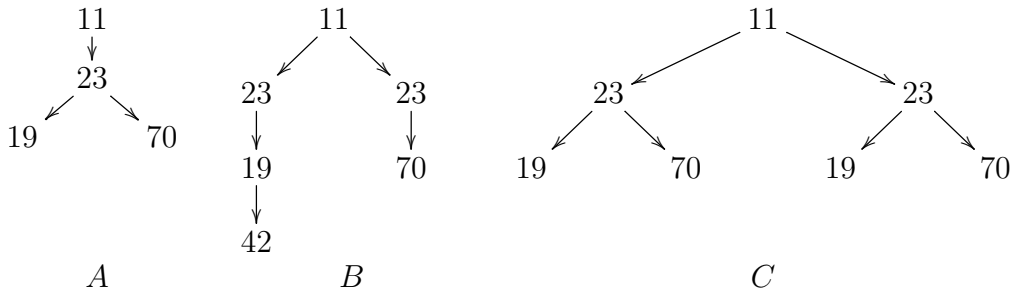


Figura 1: Alberi binari negli esempi

I comandi di compilazione saranno (per l'account twiki LilyEvans):

```
gcc -std=c99 main-2019-3-1.c binTreeRat.c LilyEvans.1.c
gcc -std=c99 main-2019-3-2.c binTree.c list.c LilyEvans.2.c
gcc -std=c99 main-2019-3-3.c binTree.c LilyEvans.3.c
```